# POLITECNICO DI TORINO

**Master's Degree in Mathematical Engineering**




**Master's Degree Thesis**

# Data-Driven Turbulence Closure Models for Steady and Unsteady Reynolds–Averaged Navier–Stokes Equations

**Supervisors**

**Dr. Maria STRAZZULLO**

**Prof. George Em KARNIADAKIS**

**Dr. Zhen ZHANG**

**Candidate**

**Luigi DONNARUMMA**

**September 2026**

# Summary

Reynolds-Averaged Navier–Stokes (RANS) simulations remain the workhorse of industrial computational fluid dynamics, and their accuracy is bounded by the turbulence closure. This thesis develops data-driven closures for the Reynolds stress tensor in the wake of a circular cylinder and asks two questions: whether a closure gains from reading beyond the point, and whether, in the unsteady problem, it gains from a memory of the shedding cycle. Both questions are judged on four counts: the accuracy on reference data, the coupling with the solver, the accuracy of the coupled solution and the convergence of the iteration.

Reference data come from direct and large-eddy simulations at four Reynolds numbers between $3.9 \times 10^3$ and $1.4 \times 10^5$. Three are used for training and the highest is held out. The closures span three levels of description. A tensor-basis neural network reads the point and returns a realizable stress by construction. A vector-cloud network reads a neighbourhood whose radius is a closed-form function of the wall distance and of the angular position, obtained by symbolic regression. Two graph networks read the mesh through message passing with an equivariant vector channel, in a single-scale form and in a multiscale form with a coarse level of fixed node count. Every closure is coupled to the same finite element solver, written for this work, in which the treatment of the closure term is varied one ingredient at a time. For the unsteady problem the solver marches in time and the tensor-basis closure is extended with rate-of-change features and with a recurrent head over past phases.

Four main results follow. First, non-locality helps. A closure that reads a neighbourhood of the point instead of the point alone is more accurate on reference data, couples to the solver under the same numerics as the local closure, and, when its reach is set by the domain rather than by the mesh, produces the most accurate coupled mean field of the work: the multiscale graph closure reaches an $L^2$ relative error of 0.037 on the velocity field and 0.345 on the stress tensor field, against 0.048 and 0.402 for the local closure and 0.103 and 0.640 for the classical model. Second, the memory of the flow helps. In the phase-resolved problem the two closures that read past phases of the shedding cycle reproduce the reference vortex street, with a cycle more periodic than the classical model, the shedding frequency within one per

cent of the reference and the stresses within 0.24 to 0.44 of it, where the classical model leaves them at 0.78 to 0.92. The closures without memory either never reach a limit cycle or reach it at the wrong amplitude. Third, a method to couple a learned closure to the solver is established, and its key element is the filtering of the predicted stress before it enters the momentum equation. Through the split of the closure term the filter introduces a stabilizing dissipation, proportional to the closure's own eddy viscosity and acting on the smallest resolved scales only. This single term is what makes the coupling work, both in the steady and unsteady solver. Fourth, accuracy on reference data does not predict accuracy inside the solver: the coupling amplifies the error of every closure, by a factor between 3.3 and 10.7, and reverses the ordering of the closures, so a closure has to be selected in the solver it will be used in.

Beyond these, the architecture-specific ablations yield observations that are worth recording. Symbolic regression finds a simple analytic relation, depending only on the wall distance and on the angular position around the cylinder, that models the advection length of this flow. That length is the reference scale on which the neighbourhood of the non-local closure is built, and it is what makes the coupling of the vector cloud possible. A stress tensor reconstruction from the net output parameters can be the most accurate on reference data but unusable in the solver. Data augmentation made trough reflection-and-reversal symmetry of the steady problem doubles the training set at no cost. A tensor form of the implicit coefficient in the split improves the convergence of the iteration by up to a factor of five. In the unsteady problem periodicity and accuracy are separate properties, since a closure without memory can reproduce the shedding cycle as those with memory but still overshoot the amplitude by a factor of 3.5. Still in the unsteady case, a similarity analysis in feature space suggests that the information the closure lacks lies in past phases rather than in close neighbouring cells, an indication that only a coupled test can confirm.

Future closures should be trained against the coupled solution rather than against reference data alone, whether with the solver in the loop, through the adjoint of the discrete equations or through a residual term in the loss. The natural continuation is then a single closure, non-local in space and recurrent in time, as a unique general model that crosses the steady and the unsteady problem.

# Acknowledgements

First and foremost, I would like to express my deepest gratitude to my advisor, Dr. Maria Strazzullo, for her invaluable guidance, constant support, and unwavering presence throughout this entire work. Her profound expertise, along with the research method and academic rigor she has instilled in me, have been fundamental to my growth as a researcher and to the successful completion of this thesis.

I am also sincerely grateful to Professor George Em Karniadakis for welcoming me into the CRUNCH group at Brown University. This experience provided me with the extraordinary opportunity to collaborate with brilliant researchers who share my same passions, allowing me to conduct research within one of the most prominent and prestigious hubs in this field.

Special thanks go to Dr. Zhen Zhang, who served as my primary mentor during my time at Brown University. I am deeply thankful for his constant availability, for guiding me through every doubt and challenge, and for sharing a visionary approach to research that I will cherish and rely on throughout my future academic and professional endeavors.

# Table of Contents

**List of Tables** x

**List of Figures** xvii

**Acronyms** xxvii

**1 State of the art and motivations** 1
- 1.1 State of the art . . . 2
  - 1.1.1 Classical closures and their limits . . . 2
  - 1.1.2 Data-driven closures . . . 3
  - 1.1.3 The coupling with the solver . . . 4
  - 1.1.4 Non-locality . . . 5
  - 1.1.5 Unsteady closures and the phase-resolved equations . . . 6
- 1.2 Motivations . . . 7

**2 Reynolds-Averaged Navier–Stokes Equations** 10
- 2.1 The Navier–Stokes Equations . . . 10
- 2.2 Reynolds-Averaged Navier–Stokes Equations . . . 13
  - 2.2.1 Reynolds Decomposition . . . 13
  - 2.2.2 The Reynolds Stress Tensor and the Closure Problem . . . 14
  - 2.2.3 Steady versus Unsteady Statistics: Time, Ensemble, and Phase Averages . . . 14
- 2.3 Turbulence Closure Models . . . 17
  - 2.3.1 Classical Eddy-Viscosity Models . . . 17
  - 2.3.2 Machine-Learning-Based Closure Models . . . 18

**3 Machine Learning Local Closures for RANS** 23
- 3.1 Machine Learning . . . 23
  - 3.1.1 Dataset . . . 24
  - 3.1.2 Per-point loss weighting . . . 27
  - 3.1.3 Loss function . . . 27

3.1.4 Evaluation Metrics . . . 30
3.2 Machine Learning Local Models . . . 32
3.2.1 The easiest set up: Multi Layer Perceptron . . . 32
3.2.2 Tensor Basis Neural Network . . . 33
3.2.3 Physics-informed training . . . 38
3.2.4 Ablations . . . 44
3.3 FEM Solver . . . 55
3.3.1 Governing equations and function spaces . . . 55
3.3.2 Picard linearization . . . 57
3.3.3 Explicit–implicit splitting of the closure term . . . 58
3.3.4 Discretized matrix form and stabilization . . . 59
3.3.5 Mesh . . . 62
3.3.6 Boundary conditions . . . 64
3.3.7 Near-wall stress damping . . . 64
3.3.8 Feature and stress filtering . . . 65
3.4 Results Solver Coupling . . . 67
3.4.1 Setting of the comparison . . . 67
3.4.2 Three stabilization operators . . . 69
3.4.3 The first configuration that works, and its limits . . . 71
3.4.4 The adopted configuration: the filtered split as a stabilization 79
3.4.5 What the coupling costs . . . 84
3.4.6 Summary . . . 93

**4 Non-Local closure for RANS: VCNN** 97
4.1 VCNN . . . 98
4.1.1 The cloud . . . 98
4.1.2 Frame-dependent and frame-independent inputs . . . 99
4.1.3 Bilinear pooling and the invariance argument . . . 101
4.1.4 Reconstruction . . . 102
4.1.5 Training and results . . . 103
4.2 Coupling with the solver . . . 106
4.2.1 The numerical setting . . . 107
4.2.2 Results . . . 107
4.2.3 Why the neighbourhood makes the coupling harder . . . 110
4.2.4 What the neighbourhood costs . . . 111
4.2.5 How this configuration was reached . . . 112
4.3 Symbolic Regression for radius length . . . 112
4.3.1 Symbolic regression . . . 113
4.3.2 Dataset and search setup . . . 113
4.3.3 Selection from the Pareto front . . . 114
4.4 Ablations . . . 116

4.4.1 What the closure reads . . . 117
4.4.2 How the stress is rebuilt . . . 119
4.4.3 How the network is trained . . . 120
4.4.4 The role of the solver . . . 120

**5 Non-Local closure for RANS: GNN** 123
5.1 Graph closures . . . 124
5.1.1 The graph . . . 124
5.1.2 Node state and the equivariant vector channel . . . 124
5.1.3 Readout . . . 126
5.1.4 Data augmentation . . . 127
5.1.5 Reaching far: the multiscale variant . . . 129
5.1.6 Training and results . . . 134
5.2 Coupling with the solver . . . 138
5.2.1 The numerical setting . . . 138
5.2.2 Results . . . 139
5.2.3 Reading the numbers through the graph mismatch . . . 140
5.3 The receptive field . . . 142
5.3.1 What range costs . . . 142
5.3.2 What does not transfer, and what would settle it . . . 142
5.4 Ablations . . . 143
5.4.1 What the closure reads . . . 143
5.4.2 How the stress is rebuilt . . . 143
5.4.3 How much of the coarse branch is used . . . 146
5.4.4 How the network is trained . . . 146

**6 Local closure for URANS** 148
6.1 Solver FEM . . . 149
6.1.1 Governing equations and function spaces . . . 149
6.1.2 Time discretization and linearization . . . 149
6.1.3 Explicit–implicit splitting of the closure term . . . 151
6.1.4 Discretized matrix form and stabilization . . . 151
6.1.5 Mesh . . . 154
6.1.6 Boundary and initial conditions . . . 155
6.1.7 Near-wall stress damping . . . 156
6.1.8 Feature and stress filtering . . . 158
6.1.9 Force coefficients and shedding diagnostics . . . 159
6.1.10 Implementation and cost . . . 159
6.1.11 Time-step convergence . . . 160
6.2 The phase-resolved problem . . . 160
6.2.1 The phase-resolved dataset . . . 161

6.2.2 What a usable unsteady solution has to satisfy . . . . . 161
6.2.3 The reference and the control . . . . . 163
6.2.4 The four closures . . . . . 164
6.3 Retraining the spatial closure: TBNN for URANS . . . . . 165
6.3.1 *A priori* . . . . . 165
6.3.2 In the solver . . . . . 167
6.4 Rate-of-change features: RateTBNN . . . . . 170
6.4.1 *A priori* . . . . . 170
6.4.2 In the solver . . . . . 171
6.5 A recurrent head: TBNN-MAMBA . . . . . 172
6.5.1 *A priori* . . . . . 172
6.5.2 In the solver . . . . . 174
6.6 A recurrent head with rate features: RateTBNN-MAMBA . . . . . 175
6.6.1 *A priori* . . . . . 175
6.6.2 In the solver . . . . . 178
6.7 The four closures side by side . . . . . 178
6.7.1 *A priori* . . . . . 178
6.7.2 In the solver . . . . . 181
6.8 What the closures get wrong . . . . . 185
6.8.1 The coupling amplifies the error . . . . . 185
6.8.2 When closure learns the mean, not the dynamics of the cycle 186
6.8.3 How much of the fluctuation is learnable at all . . . . . 189
6.9 Ablations . . . . . 193
6.9.1 Learning rate . . . . . 193
6.9.2 Numerical setting . . . . . 195
6.9.3 Filtering and splitting . . . . . 197
6.9.4 Corrector sub-iterations . . . . . 199
6.9.5 Length of the window . . . . . 199
6.10 Summary . . . . . 201

**7 Conclusions** 205
7.1 The work in brief . . . . . 207
7.2 Main findings . . . . . 207
7.3 Outlook . . . . . 211

**Bibliography** 213

# List of Tables


3.1 Simulation parameters (Zhang et al. [21]) and exported analysis mesh used in this thesis. . . . . 25

3.2 MLP results on the Re = 11 000 training simulation, evaluated in-sample. . . . . 34

3.3 MLP results on the held-out Re = 140 000 test set. . . . . 34

3.4 Test performance of the adopted network on the held out Reynolds number. . . . . 38

3.5 MLP closure trained in a supervised setting and inside a physics-informed formulation, at identical network topology and parameter count, held-out Re = 140 000 test set. The two settings differ also in the activation and in the standardization of the closure inputs. . 41

3.6 Tensor-basis closure trained in a supervised setting and inside a physics-informed formulation, held-out Re = 140 000 test set. The two configurations differ in both the training strategy and the reconstruction. . . . . 43

3.7 Accuracy of the adopted network against an extended feature set and a reduced basis set, all using the capped reconstruction. . . . . 47

3.8 MLP [24,24]: effect of a post-hoc eigenvalue-clipping projection onto the nearest PSD tensor, held-out Re = 140 000 test set. No retraining involved. . . . . 48

3.9 Accuracy of the adopted network against the earlier, unit-normalized-basis, linear-reconstruction configuration of Ling et al. [4]. . . . . 49

3.10 Realizability diagnostics and accuracy of the reconstruction attempts, compared against the earlier, unconstrained configuration, on the held-out Re = 140 000 test set. CS viol. is the fraction of test points at which the predicted stress violates the Cauchy–Schwarz condition $\overline{u'v'}^2 \leq \overline{u'u'}\,\overline{v'v'}$. Neg. $\overline{u'u'}$ and neg. $\overline{v'v'}$ are the fractions at which the predicted normal stresses are negative. A realizable prediction has all three at zero. . . . . 50

3.11 MLP: best network overall vs. the smaller network adopted, held-out Re = 140 000 test set. . . . . 51

3.12 MLP [24,24]: composite physics-weighted loss vs. plain MSE, held-out Re = 140 000 test set. . . . . 52

3.13 Final relative velocity update $\delta u$, Equation (3.40), at Re = 11 000 with $c_S = 1$. Bold marks a configuration with $\delta u < 10^{-3}$, italics one with $10^{-3} \leq \delta u < 10^{-2}$, and a value in upright type one with $\delta u \geq 10^{-2}$, whose field is an arbitrary iterate of a sequence that is still moving. An asterisk marks the runs that reached the solver tolerance $10^{-5}$ before the iteration budget was exhausted. The superscript "s" marks a run stopped early by a singular momentum matrix. All runs use the scalar split of Equation (3.43). . . . . 73

3.14 *A posteriori* accuracy at Re = 11 000, isotropic diffusion, $c_S = 1$. Accuracy is reported only where the iteration reached at least a partial fixed point. Bold marks a configuration with $\delta u < 10^{-3}$, italics one with $10^{-3} \leq \delta u < 10^{-2}$, and a configuration with $\delta u \geq 10^{-2}$ is left blank, its field being an arbitrary iterate of a sequence that is still moving. The values of $\delta u$ are those of Table 3.13. Both errors are computed on the window of Equation (3.67). All runs use the scalar split of Equation (3.43). . . . . 75

3.15 Effect of the stabilization scale under isotropic diffusion. For each closure the table reports the best value over the four filtering configurations, separately for the two criteria. The errors are computed on the window of Equation (3.67). All runs use the scalar split of Equation (3.43). . . . . 78

3.16 *A posteriori* accuracy of the local closures at Re = 11 000 under the adopted configuration, consistent streamline-upwind stabilization with the filtered split of Equation (3.77), 150 iterations. Relative $L^2$ errors are on the window of Equation (3.67), coefficients of determination over the whole domain. . . . . 80

3.17 Dependence on the iteration budget, tensor-basis closure under the adopted configuration. For each filtering configuration the first two rows are the final relative velocity update of Equation (3.40) and the final closure increment of Equation (3.79), and the third is the relative $L^2$ error of the velocity magnitude on the window of Equation (3.67). Only the iterations budget changes between columns. 83

3.18 Median relative velocity update over the last third of the iteration budget, without stabilization and with the grad-div term alone, at Re $= 11\,000$ and 150 iterations. For the transported closure the output filter is not applied, so the first and third columns coincide, as do the second and fourth. All runs use the scalar split of Equation (3.43). . . . . 85
3.19 Velocity update and closure increment at the last iteration, tensor-basis closure, grad-div stabilization only, 150 iterations. The two columns are the two unknowns of the coupled problem, and they are not at the same stage of convergence. All runs use the scalar split of Equation (3.43). . . . . 87
3.20 Median relative velocity update, tensor-basis closure, grad-div stabilization only, with the full predicted tensor and with its Boussinesq projection. The two rows differ only in what is imposed on the momentum equation. The last row is the ratio between them. All runs use the scalar split of Equation (3.43). . . . . 89
3.21 Median relative velocity update and errors over the near-wake window, tensor-basis closure, grad-div stabilization only, 150 iterations. The scalar and tensor splits differ only in whether the orthogonal part of the prediction is treated implicitly. Errors are the relative $L^2$ error of the velocity vector and the aggregate stress error of Section 3.1.4, both restricted to the window of Section 3.4.1. . . . . 91
3.22 Median relative velocity update of the tensor-basis closure against the bound $c_{\mathbb{M}}$ of Equation (3.86). The value $c_{\mathbb{M}} = 0$ is the scalar split of Equation (3.43). . . . . 92
3.23 Improvement of the median relative velocity update over the scalar split, tensor-basis closure. A value below one is a degradation. . . . 92

4.1 VCNN closure on the held-out Re $= 140\,000$ test set, with the local tensor-basis closure of Table 3.4 reported alongside on the two metrics the two share. Bold marks every entry on which the non-local closure is ahead. . . . . 105
4.2 *A posteriori* accuracy of the non-local closure at Re $= 11\,000$ under the adopted configuration, with the local tensor-basis closure and the classical reference reported under the same setting. Relative $L^2$ errors are on the window of Equation (3.67), coefficients of determination over the whole domain. . . . . 108
4.3 Pareto front of the symbolic regression for the cloud radius. The loss is the mean-squared error on $\log r$, and complexity is the number of nodes in the expression tree. Here $d$ is the wall distance and $\alpha$ the angular position. . . . . 115

4.4 Neighbourhood ablations on the held-out Re $= 140\,000$ test set. Within each block a single ingredient varies. The two blocks use different feature sets and are not to be compared row by row. . . . 118
4.5 The three reconstructions at parity in every other respect, held-out Re $= 140\,000$ test set. . . . . 119
4.6 The non-local closure under three formulations of the coupling, held at Re $= 11\,000$. The input filter alone returns no solution in any of the three. Relative $L^2$ errors are on the window of Equation (3.67). 121

5.1 The two graph closures on the held-out Re $= 140\,000$ test set, with the local tensor-basis closure and the vector-cloud closure reported alongside. Every row is the network deployed in the solver. Relative $L^2$ errors are over the whole domain. . . . . 135
5.2 *A posteriori* accuracy of the two graph closures at Re $= 11\,000$ under the adopted configuration, with the closures of the previous chapters reported under the same setting in their own adopted filtering configuration. Relative $L^2$ errors are on the window of Equation (3.67), coefficients of determination over the whole domain. 139
5.3 GNN: tensor-basis against direct readout, held-out Re $= 140\,000$ test set. The first two rows share every setting except the readout. . 143
5.4 Trace-normalized against capped reconstruction on both graph architectures, at parity in every other respect. The three coefficients of determination are on the held-out Re $= 140\,000$ test set; the final update and the velocity error are measured at Re $= 11\,000$ under the adopted configuration, with both filters on the single-scale architecture and the output filter on the multiscale one. The dagger marks the head adopted for each architecture. . . . . 144
5.5 The two reconstruction heads on the multiscale architecture at Re $= 11\,000$, across the four filtering configurations. The update is reported both at the last iteration and at the lowest value reached along the iteration. The dagger marks the filtering configuration adopted for this architecture. . . . . 145
5.6 MS-GNN: test loss against the number of coarse rounds, at two learning rates, with everything else held fixed. The starred entry is the test loss recorded by a run that stopped because got stuck. . . . 146

6.1 The two full-domain meshes considered for the unsteady computations. 155
6.2 Steady tensor-basis closure evaluated as is on phase-resolved data, and the same architecture retrained on phase-resolved data in the configuration carried into the solver, held-out Re $= 140\,000$, with $\rho$ the correlation with the true field. . . . . 167

6.3 The retrained spatial closure in the solver, hyperviscous split, both filters. Upper block: diagnostics of Section 6.2.2 on the two windows, with the control and the reference of Section 6.2.3. Lower block: relative $L^2$ error of the phase-averaged coupled solution against the reference, over the near wake $x \in [-2,8]D$, $y \in [-3,3]D$, averaged over the phases. . . . . 168

6.4 Tensor-basis closure of Section 6.3 compared with the same closure but with the six rate features of Equations (6.20)–(6.21). Metrics on held-out $\mathrm{Re} = 140\,000$. . . . . 171

6.5 The closure with the rate features in the solver, hyperviscous split, both filters. Blocks as in Table 6.3. . . . . 171

6.6 Recurrent head on a frozen steady spatial network, receiving the coefficient sequence only, configuration carried into the solver, held-out $\mathrm{Re} = 140\,000$. . . . . 174

6.7 The recurrent closure in the solver, hyperviscous split, output filter. Blocks as in Table 6.3. . . . . 175

6.8 Recurrent head receiving the coefficient sequence, the local features and the six rate features, configuration carried into the solver, held-out $\mathrm{Re} = 140\,000$. . . . . 178

6.9 The closure with the rate features and the recurrent head in the solver, hyperviscous split, output filter. Blocks as in Table 6.3. . . . 179

6.10 *A priori* accuracy of the four closures deployed in the solver, held-out $\mathrm{Re} = 140\,000$, with $\rho$ the correlation with the true field. The shear stress is the worst-predicted component of every one of them. . . . . 181

6.11 The four closures in their adopted configurations. Upper block: diagnostics on the window $t \in [60,120]$, the last column being the number of the conditions of Equation (6.18) that are met. Lower block: phase-averaged errors over the near wake on $t \in [130,180]$, with the classical closure continued over the same window and phase averaged in the same way. . . . . 184

6.12 Relative $L^2$ error of each deployed closure at $\mathrm{Re} = 11\,000$ over the whole domain, component by component: *a priori*, on the reference fields with exact features, and *a posteriori*, on the phase-averaged coupled solution, with the ratio of the two. . . . . 186

6.13 Decomposition of Equation (6.23) over the wake, 40 phases, area-weighted. The first block is a property of the reference data alone. The second is the relative error of the four deployed closures on each part; the two recurrent closures are evaluated with the head having seen a full period before the phase, which is its condition in the solver. 187

6.14 Integrated wake production of Equation (6.24) over the cycle, and its correlation with the reference cycle. The predicted production has very nearly the right mean and almost no variation within the cycle. . . . . 187

6.15 Relative error on the phase fluctuation of the stress over the wake, for the two feature sets of Sections 6.3 and 6.4. The floor is the right-hand side of Equation (6.26), normalized in the same way. The last column is the trained closure of Table 6.13. . . . . 190

6.16 Floor of Equation (6.26) against the information given to the closure, over the whole wake. The history is a number of past phases of the same cell, the stencil a number of neighbouring cells at the same phase. The mean distance between neighbours is reported because it grows with the dimension and works against the estimate. . . . . 191

6.17 Tensor-basis closure trained directly on phase-resolved data, learning-rate sweep, held-out Re $= 140\,000$. All configurations use two hidden layers of 24 units and 1000 epochs. . . . . 193

6.18 Tensor-basis closure trained on phase-resolved data with the six rate features of Equations (6.20)–(6.21) added to the coefficient-network input, learning-rate sweep, held-out Re $= 140\,000$. Everything else is identical to Table 6.17. . . . . 194

6.19 Recurrent head on a frozen steady spatial network, held-out Re $= 140\,000$. Upper block: the head receives the coefficient sequence only. Lower block: the head additionally receives the local features and the six rate features of Equations (6.20)–(6.21). . . . . 194

6.20 Recurrent head receiving the coefficient sequence, the local features without the pressure-gradient norm, and the six rate features, learning-rate sweep, held-out Re $= 140\,000$. . . . . 195

6.21 One closure, four numerical settings, window $t \geq 60$. The first row is the consistent stabilization alone and the second adds the entropy viscosity, so the two differ only in the added dissipation, and the last two only in whether the isotropic part of the predicted stress is retained. The transition between the second and the third row changes three ingredients at once and is not a single-variable comparison. . . . . 195

6.22 Cycle-to-cycle dispersion of Equation (6.17), window $t \in [60,120]$, for the four closures under the exact and the hyperviscous form of the split and the four filtering configurations. Bold marks the runs that satisfy Equation (6.18); the starred one does so on this window and drifts on the next. The classical closure gives 2.1%. . . . . 198

6.23 Cycle-to-cycle dispersion at one, three and five corrector sub-iterations per step, exact split, no filter. The cost per step is 0.41, 1.30 and 2.1 seconds on sixteen processes. . . . . 199
6.24 Cycle statistics of the four adopted runs on consecutive windows of 30 convective time units. The peak fraction is not reported because its spectral band is not resolved on a window this short. . . . . 200
6.25 Relative $L^2$ error of the phase-averaged fields over the near wake, mean over the phases, on the two halves of the continuation window. 200

7.1 The steady closures, what each one reads and how it performs on the four counts. Grey cells mark an ingredient the closure does not have or a metric that does not apply. Errors are relative $L^2$ errors: *a priori* on the stress tensor over the whole domain of the held-out flow, Table 5.1; *a posteriori* on the velocity magnitude and on the stress tensor of the coupled solution at the training Reynolds number, over the near-wake window of Equation (3.67), Table 5.2. The last column is the final velocity update of the iteration. *A post.* stands for *a posteriori*. Best learned value in bold. . . . . 208
7.2 The phase-resolved closures, what each one reads and how it performs. Grey cells as in Table 7.1. Errors are relative $L^2$ errors: *a priori* on the stress tensor of the held-out flow, Table 6.10; *a posteriori* on the phase-averaged velocity magnitude and shear stress of the coupled solution over the near wake at the training Reynolds number, mean over the phases, Table 6.11. The retrained spatial closure does not meet the admission criterion of Equation (6.18), and its errors are those of a cycle that is still growing. Best learned value in bold. . . 209

# List of Figures

3.1 Meshes of the reference simulations used in this thesis (data from Zhang et al. [21]). . . . . 26

3.2 MLP architecture: input features, hidden layers, and output targets. 33

3.3 MLP: training and validation loss history. . . . . 34

3.4 MLP: predicted versus true Reynolds stress components, training ($Re = 11\,000$) and held-out ($Re = 140\,000$) test data. Neither scatter is satisfactory, but the two point clouds are distributed similarly. This indicates that accuracy is retained at a test Reynolds number more than twice the highest seen in training ($Re = 60\,000$). Despite its simplicity and limited accuracy, the plain MLP generalizes to unseen Reynolds numbers. . . . . 35

3.5 TBNN architecture: invariant scalar features, tensor bases, coefficient network, and the capped reconstruction of the Reynolds stress. . . . 37

3.6 TBNN, adopted network: predicted versus true Reynolds stress components, held-out $Re = 140\,000$ test set. . . . . 38

3.7 Physics-informed formulation. One state network per Reynolds number supplies the mean fields, and a single closure network shared across Reynolds numbers supplies the stress. The residual, boundary and data terms are evaluated on three distinct sets of points, and the momentum residual differentiates the state network twice. . . . 39

3.8 MLP closure trained inside the physics-informed formulation: predicted versus true Reynolds stress components. The shear stress is almost entirely lost. . . . . 42

3.9 Physics-informed training: history of the four loss terms, reported separately. The residual term settles orders of magnitude below the terms that impose an external target. . . . . 43

3.10 Tensor-basis closure trained inside the physics-informed formulation: predicted versus true Reynolds stress components, held-out test set, coloured by the area of the associated mesh cell on a logarithmic scale. The largest cells, in yellow, depart from the diagonal by little in absolute terms and by a great deal in relative terms, at a true stress close to zero. They carry 98% of the domain area and therefore dominate the area-weighted error, while the points farthest from the diagonal belong to the small cells near the cylinder and dominate instead the coefficient of determination, which counts points rather than area. . . . . 44

3.11 MLP: the same predicted versus true Reynolds stress components as Figure 3.4, after the post-hoc positive-semidefinite projection. Every $\overline{u'u'}$ and $\overline{v'v'}$ prediction is now non-negative, visible as the absence of points below zero for these two components. . . . . 48

3.12 Adaptive weighting of the loss terms. The residual weight saturates upward while the others stay of order one. The faint curves are the control trained on a single Reynolds number, which was run for fewer optimization steps and therefore ends earlier on the same axis. 53

3.13 Training with the residual and boundary terms weighted to zero. The data terms are fitted an order of magnitude better than in any configuration where the physical terms are active, while the physical residuals, measured but not enforced, grow throughout training. . . 56

3.14 RANS solver mesh: anisotropic boundary layer at the cylinder wall, continuous wake refinement, and coarsening toward the far field. . . 63

3.15 Convergence history of the Picard iteration at $\mathrm{Re} = 11\,000$, with isotropic diffusion (top row) and consistent streamline-upwind stabilization (bottom row). Columns are closures, curves within each panel are the four filtering configurations of Section 3.3.8. The physics-informed MLP is omitted: it never approaches convergence in any configuration. In the reference panels the four curves coincide, the transported closure being insensitive to the filters. The vertical range differs between the two rows, since no run of the upper row exceeds 10. The dashed line marks the $9 \times 10^{-4}$ threshold and the shaded band the artificial-viscosity ramp of Equation (3.41). Curves that stop before the budget have reached the solver tolerance. . . . 74

3.16 Converged fields at Re = 11 000 under isotropic diffusion, each closure in the configuration stated above its column. Top row, velocity magnitude; middle row, shear stress; bottom row, local error on the shear stress against the reference. Each row shares one colour scale. The dashed rectangle is the window of Equation (3.67): the error of the bottom row is concentrated inside it and vanishes outside. The columns are nearly indistinguishable on the velocity and clearly distinct on the stress. . . . . 77

3.17 Wake profiles at four streamwise stations, converged runs under isotropic diffusion, each closure in the configuration of Figure 3.16. Top row, streamwise velocity; bottom row, shear stress. The horizontal scale is set independently in each panel, since the shear stress falls by an order of magnitude between the first station and the last; the panels therefore show shape and not magnitude. The velocity profiles nearly collapse; the stress profiles do not. . . . . 77

3.18 Converged fields under the adopted configuration, compared against the reference DNS. Each closure is shown in its best filtering configuration of Table 3.16. The multilayer closure is absent because it produces no field under this stabilization. The dashed rectangle is the window of Equation (3.67), over which the errors of the tables are computed. . . . . 82

3.19 The predicted deviatoric stress in the two-dimensional space of symmetric traceless tensors. A scalar eddy viscosity can only represent the component parallel to the strain rate, so the orthogonal component is left in the explicit residual. The tensor coefficient of Equation (3.83) represents both, within the coercivity cone $|b| \leq \nu_M$. 88

3.20 Converged fields of the best configuration of each local closure against the reference, over the near-wake window. Rows, from top to bottom: velocity magnitude, the two velocity components, the three Reynolds stress components and the pressure. All the maps in a row share one colour scale. Each column is labelled with its own configuration and with its velocity and stress errors, and the two are not the same: the multilayer closure produces no field under the adopted stabilization, so its best deployable run is the one of Section 3.4.3. This is the only figure in which its converged field appears. . . . . 94

4.1 Geometry of the vector cloud. The sampling radius is set by the wall distance and the angular position alone, so it is fixed once for a given mesh. Clouds close to the cylinder are markedly smaller than clouds in the far field. . . . . 99

4.2 Architecture of the non-local closure. The cloud is described by a feature matrix whose first four columns carry true vectors and whose remaining columns carry frame invariants. Only the invariant block is embedded, and the two are recombined by a bilinear pooling whose entries depend on the cloud through pairwise inner products alone. The fitting network outputs the coefficients of a fixed tensor basis and a scale, from which the stress is reconstructed in a form that is realizable by construction. . . . . 104
4.3 Training and validation loss of the non-local closure. The steps in both curves coincide with the scheduled reductions of the learning rate. . . . . 104
4.4 Non-local closure: predicted versus true Reynolds stress components, training ($\mathrm{Re} = 11\,000$) and held-out ($\mathrm{Re} = 140\,000$) test data, on the same axes as Figure 3.6. The scatter is visibly tighter than that of the local closures, and the margin is largest on the first component. 106
4.5 Area-weighted relative error against wall distance, for the closures compared in this work. The shaded band marks the near-wall region where the solver replaces the raw prediction by the damped one of Section 3.3.7. . . . . 106
4.6 Picard history of the non-local closure under the adopted configuration. The update stops descending well before the end of the budget and settles into a band. The projection error of the stress drifts upward when only the output filter is applied and stays flat when both are. . . . . 109
4.7 Mean fields at $\mathrm{Re} = 11\,000$ under the adopted configuration, reference data against the non-local and the local tensor-basis closures. The velocity fields are close to each other and to the reference, and the two closures separate on the stress components. . . . . 110
4.8 Pareto front of the symbolic regression. The selected expression sits at the largest single drop in loss along the front, before the search begins to introduce the Reynolds number and to fit the four available simulations individually. . . . . 114
4.9 Cloud radius from the original flow-dependent expression and from the geometric surrogate obtained by symbolic regression. The surrogate reproduces the spatial structure of the target without using any flow quantity, and is therefore computable once per mesh. . . . 116

5.1 From the reference mesh to the graph used by the closure. Nodes are mesh nodes and edges are element sides. The inset shows the set of nodes reachable within the number of message-passing rounds used here, which is the region of influence of the highlighted node. . 125

5.2 GNN architecture. Each node carries an invariant scalar state and an equivariant vector channel. Messages are built from invariants alone, and the only vector-valued quantity entering the vector update is the fixed relative position of the edge, which is what makes the update equivariant. The vector channel is discarded before the readout, and the stress is reconstructed from the invariant state through a fixed tensor basis. . . . . 128
5.3 The two augmentations. Re-meshing changes the discretization and leaves the field unchanged. Reflection with flow reversal maps the field onto a solution of the same equations, with the shear stress changing sign. . . . . 129
5.4 The two transfers between levels and the path a node state follows through one cycle; the layout of the scheme follows Fortunato et al. [36], the meshes are schematic. The downsampling carries the fine node states onto the coarse level, the coarse rounds move information across the domain, and the upsampling returns it to the fine level, where it is combined with the state the fine rounds had already produced. The numbered badges give the order of the operations within one cycle. . . . . 130
5.5 The auxiliary coarse level. Coarse nodes are a subset of the fine mesh nodes chosen by farthest-point sampling, connected by a Delaunay triangulation. The inset shows the bipartite links that carry messages between the two levels. . . . . 131
5.6 MS-GNN forward pass. A single round on the fine graph is followed by a downsampling onto the coarse level, eleven rounds there, an upsampling back, and a final fine round. The coarse branch carries almost no cost, since the coarse graph has two orders of magnitude fewer edges than the fine one. . . . . 133
5.7 GNN: training and test loss history. The dashed level marks the local minimum discussed later in this section, which the reported training passes through in its first epochs before escaping. . . . . 134
5.8 MS-GNN: training and test loss history, with the test loss of the single-scale architecture shown for comparison. . . . . 135
5.9 GNN: predicted versus true Reynolds stress components, training and held-out test data; the legend of the third panel applies to all three. The streamwise normal stress is visibly more scattered than the other two components. . . . . 136
5.10 MS-GNN: predicted versus true Reynolds stress components, training and held-out test data; the legend of the third panel applies to all three. The scatter on the streamwise normal stress is markedly tighter than in Figure 5.9. . . . . 136

5.11 Spatial distribution of the error on the streamwise normal stress for the two graph architectures, and their difference. The contour on the last panel encloses the region where the cell size is below the median. . . . . 137

5.12 *A priori* against *a posteriori* accuracy of the four deployed closures at the same Reynolds number, Re $= 11\,000$, and over the same window, that of Equation (3.67). On the abscissa, the relative $L^2$ error on the stress tensor of each closure evaluated on the reference fields of the training flow, with the definition used in Table 5.2; on the ordinates, the *a posteriori* error of the coupled solution of that table, on the stress (left, with the bisector) and on the velocity (right). The *a priori* errors are those of a fit on the training data, 0.041 for the vector cloud, 0.087 and 0.103 for the single-scale and the multiscale graph closures, 0.100 for the local tensor basis, and the coupling amplifies every one of them, by a factor of 3.3 for the multiscale graph closure and of 10.7 for the vector cloud. The ordering does not survive: the closure with the tightest fit, the vector cloud, is the least accurate *a posteriori* on the stress, 0.442; the multiscale graph closure, last of four *a priori*, is first on both the stress, 0.345, and the velocity, 0.037; the local tensor basis, *a priori* within three thousandths of it, is second on both. A smaller error on the training flow itself does not imply a smaller error once the closure is coupled, so the disagreement met in Chapters 3 and 4 is not an effect of the change of Reynolds number between the two measurements but a property of the coupling. The *a priori* error is computed on the grid of the reference data, the *a posteriori* one on the solver mesh, over the same region. . . . . 141

6.1 Domain and mesh for the unsteady computations. The symmetry condition used by the steady solver removes the antisymmetric mode from the discrete solution space, and vortex shedding is that mode, so the full domain is required. . . . . 156

6.2 Lift and drag histories over a full computation, and the power spectral density of the lift coefficient on the statistically steady window. The dashed line marks the reference Strouhal number. . . . . 157

6.3 Near-wall behaviour of the predicted stress with and without damping, and with the damping applied before and after the output filter. The dashed lines are the asymptotic slopes required by Equation (3.62). 158

6.4 Why the spectral diagnostics are not sufficient. The control and a closure under the exact split both pass the peak and high-frequency conditions. Folding the two signals onto the shedding period separates them: the cycles of the control collapse onto one curve, those of the closure do not, and the adopted configuration collapses like the control. . . . . 163
6.5 The four unsteady tensor-basis closures compared in this section, drawn as one shared pipeline with two optional additions. The rate features enter the input of the coefficient network; the recurrent head corrects its output. . . . . 165
6.6 Retrained spatial closure: lift coefficient from the start of the march to the end of the continuation. The dashed lines mark the amplitude of the reference, $\pm\sqrt{2}\,C_{l,\mathrm{rms}}$, and the dotted lines the start of the evaluation window and of the continuation. . . . . 168
6.7 Retrained spatial closure: lift and drag coefficients over one shedding period, phase average of the coupled solution over $t \in [130,180]$ against the reference and against the classical closure on the same window, after phase alignment of each to the reference. . . . . 169
6.8 Retrained spatial closure: velocity magnitude and shear stress over the near wake at two opposite phases of the cycle, coupled solution against the reference at the same phase. . . . . 169
6.9 Closure with the rate features: lift coefficient from the start of the march to the end of the continuation. The dashed lines mark the amplitude of the reference, $\pm\sqrt{2}\,C_{l,\mathrm{rms}}$, and the dotted lines the start of the evaluation window and of the continuation. . . . . 172
6.10 Closure with the rate features: lift and drag coefficients over one shedding period against the reference, as in Figure 6.7. . . . . 172
6.11 Closure with the rate features: velocity magnitude and shear stress over the near wake at two opposite phases, against the reference. . . 173
6.12 Recurrent closure: lift coefficient from the start of the march to the end of the continuation. The dashed lines mark the amplitude of the reference, $\pm\sqrt{2}\,C_{l,\mathrm{rms}}$, and the dotted lines the start of the evaluation window and of the continuation. . . . . 175
6.13 Recurrent closure: lift and drag coefficients over one shedding period against the reference, as in Figure 6.7. . . . . 176
6.14 Recurrent closure: velocity magnitude and shear stress over the near wake at two opposite phases, against the reference. . . . . 176
6.15 Recurrent closure with the rate features: lift coefficient from the start of the march to the end of the continuation. The dashed lines mark the amplitude of the reference, $\pm\sqrt{2}\,C_{l,\mathrm{rms}}$, and the dotted lines the start of the evaluation window and of the continuation. . . 179

6.16 Recurrent closure with the rate features: lift and drag coefficients over one shedding period against the reference, as in Figure 6.7. . . 180

6.17 Recurrent closure with the rate features: velocity magnitude and shear stress over the near wake at two opposite phases, against the reference. . . . . 180

6.18 Training and validation loss of the four unsteady tensor-basis families, at the best learning rate of each. The two families carrying a recurrent head train for fewer epochs, since only the head is trained. . . . . 182

6.19 Recurrent closure: predicted against true phase-averaged Reynolds stress components, training (Re = 11 000) and held-out (Re = 140 000) test data, with the head having seen a full period before each phase; the legend of the third panel applies to all three. The shear stress is the component on which the held-out cloud opens most. 182

6.20 Phase-by-phase accuracy of the four unsteady families over one shedding period. The horizontal lines are the period-averaged values of Table 6.10. . . . . 183

6.21 Phase-averaged shear stress at four phases of the shedding period: reference, frozen steady closure, and the same closure corrected by the recurrent head. The comparison shows what the correction of Equation (6.22) actually supplies. . . . . 183

6.22 Left, integrated wake production over one shedding period, normalized by its own mean, for the reference and the four closures. Right, relative error of the four closures on the full stress, on its phase mean and on its fluctuation, against the share of the energy each part carries in the reference. . . . . 188

6.23 Left, the floor of Equation (6.26) against the size of the feature set, separately for past phases of the same cell and for neighbouring cells at the same phase. Right, the distribution of the phase separation between a sample and its nearest neighbours in feature space, with the quarter-cycle mark. . . . . 192

6.24 Phase-averaged fields in the near wake, at one phase of the cycle, for the reference, the classical control and the recurrent closure in its adopted configuration. Rows, from top to bottom: velocity magnitude, the two velocity components, the three Reynolds stress components and the pressure. . . . . 203

7.1 The thesis in one scheme. The two questions, the closures compared under each, in order of increasing reach on the left and of increasing memory on the right, the four counts on which every closure is judged, and the two answers with their key numbers. The two boxes at the bottom are the common ground of both halves: a solver whose numerics are built for the learned closure, and one protocol under which every closure is realizable and frame-invariant by construction. The thick border marks the closure adopted in each half. . . . . 206
7.2 *A posteriori* accuracy of the coupled closures, classical closure in grey, best learned value with a black border. Top, steady problem: relative $L^2$ errors on the velocity magnitude and on the stress tensor over the near-wake window of Equation (3.67), Table 5.2. Bottom, phase-resolved problem: relative $L^2$ errors on the phase-averaged velocity magnitude and shear stress over the near wake, mean over the phases on $t \in [130,180]$, and Strouhal number on $t \in [60,120]$ against the reference value, Table 6.11. The hatched bars are the retrained spatial closure, which does not reach a sustained limit cycle, so its values describe a cycle that is still growing. . . . . 210

# Acronyms

**NSE**
Navier–Stokes Equations

**RANS**
Reynolds-Averaged Navier–Stokes

**URANS**
Unsteady Reynolds-Averaged Navier–Stokes

**DNS**
Direct Numerical Simulation

**LES**
Large Eddy Simulation

**PIV**
Particle-Image-Velocimetry

**CFD**
Computational Fluid Dynamics

**SEM**
Spectral Element Method

**DOF**
Degrees Of Freedom

**GPU**
Graphics Processing Unit

**PDE**

Partial Differential Equation

**SST**

Shear-Stress Transport

**RSM**

Reynolds Stress Models

**PSD**

Positive Semi-Definite

**RNG**

Renormalization Group

**MLP**

Multi Layer Perceptron

**TBNN**

Tensor Basis Neural Network

**VCNN**

Vector-Cloud Neural Network

**GNN**

Graph Neural Network

**MS-GNN**

Multi-Scale Graph Neural Network

**RNN**

Recurrent Neural Network

**LSTM**

Long Short-Term Memory

**SSMs**

State-Space Models

**PINN**
: Physics-Informed Neural Network

**FEM**
: Finite Element Method

**UFL**
: Unified Form Language

**SUPG**
: Streamline-Upwind Petrov–Galerkin

**PSPG**
: Pressure-Stabilization Petrov–Galerkin

**CG**
: Continuous Galerkin

**GELU**
: Gaussian Error Linear Unit

**SiLU**
: Sigmoid Linear Unit

**ReLU**
: Rectified Linear Unit

**MSE**
: Mean Squared Error

**RMSE**
: Root Mean Square Error

**MAE**
: Mean Absolute Error

**GMRES**
: Generalized Minimal Residual method

# Chapter 1

# State of the art and motivations

Turbulent flows are governed by the Navier–Stokes Equations (NSE), and for the regimes of engineering interest their direct solution is out of reach. The number of degrees of freedom that a Direct Numerical Simulation (DNS) has to resolve grows steeply with the Reynolds number, and Large Eddy Simulation (LES), which resolves only the largest scales, remains too expensive for routine use on complex geometries [1, 2]. Industrial Computational Fluid Dynamics (CFD) therefore rests on the Reynolds-Averaged Navier–Stokes (RANS) equations, in their steady form and, when the flow carries an organized unsteadiness, in their unsteady form, the Unsteady Reynolds-Averaged Navier–Stokes (URANS) equations. Averaging removes the fluctuations from the unknowns and introduces the Reynolds stress in their place. That stress is not known, and the equations are closed only once a model expresses it in terms of the averaged field. The closure is the single model ingredient that decides the accuracy of a RANS prediction, and it is the subject of this thesis.

The classical closures are calibrated on canonical flows and are accurate within the regime of that calibration [3]. Outside it, and in particular in flows with curvature, impingement and separation, the linear relation between the Reynolds stress and the mean strain rate on which most of them rest does not hold, and the predicted anisotropy is wrong [4, 5]. Wakes behind bluff bodies are the case in point: the flow separates, the near wake is dominated by the vortex street, and the momentum transfer after separation is exactly the quantity that the classical models predict worst [2]. The wake of a circular cylinder, the benchmark of this work, has been the reference configuration for the study of these dynamics for decades [6].

The availability of high-fidelity data has opened a different route. A closure can

be learned from DNS or LES fields instead of being derived and calibrated by hand, and the last decade has produced a large body of work in this direction, reviewed in [7, 8]. This chapter surveys that body of work in the parts that matter for the present thesis. Section 1.1 covers, in order, the classical closures and their limits, the data-driven closures and how they are evaluated, the problem of coupling a learned closure to a flow solver, the non-local closures that read a neighbourhood rather than a point, and the unsteady formulation with the models that carry memory in time. Section 1.2 states the gaps that the survey leaves open and the questions that the following chapters address.

## 1.1 State of the art

This section reviews the closures in use so far and the questions they leave open. It starts from the classical models and their limits, moves to the data-driven closures and to the problem of coupling them with a solver, and ends with the two directions this thesis follows, non-locality in space and memory in time.

### 1.1.1 Classical closures and their limits

The closures used in practice belong to a small number of families. The eddy-viscosity models express the deviatoric Reynolds stress as the product of a scalar turbulent viscosity and the mean strain rate, and differ in how that viscosity is obtained: from one or two transported turbulence quantities, as in the $k$-$\varepsilon$ model [9], the $k$-$\omega$ model [10] and the shear-stress transport model that blends the two [11]. Their formulation, calibration and typical behaviour are described in Chapter 2 and in the standard reference [1], and are not repeated here. What matters for the survey is the assumption they share: the stress is aligned with the strain rate at every point, and its magnitude is set by a local scalar.

That assumption fails where the flow departs from simple shear. The linear eddy-viscosity model does not reproduce the correct Reynolds stress anisotropy in flows with curvature, impingement and separation, and not even in simple shear flows [4]. Nonlinear eddy-viscosity models were proposed to repair this by adding quadratic and cubic products of the strain and rotation rate tensors to the constitutive relation, and they perform better than the linear ones on flows with strong streamline curvature and stagnation [5]. The general form that such relations can take was established by Pope, who showed that the most general effective-viscosity hypothesis expresses the anisotropy as a linear combination of a finite set of tensor bases built from the strain and rotation rate tensors, with scalar coefficients that are functions of their invariants [12]. That representation is the backbone of most of the data-driven closures reviewed below. At the top of the classical hierarchy stand the Reynolds stress models, which transport the six

components of the stress tensor; they are recommended for complex flows in the industrial practice, at a cost in robustness and in the modelling of the remaining unclosed terms [13].

The alternative to a better closure is to resolve more of the flow. Spalart placed the strategies on a scale from steady RANS to DNS, with LES in between and unsteady RANS and hybrid RANS-LES methods as intermediate options. He argued that, for several decades, practical methods would necessarily be RANS, possibly unsteady, or RANS-LES hybrids, and that the newer strategies threaten the complex RANS models most for bluff-body flows [2]. The same conclusion is drawn in the nuclear engineering practice, where RANS and LES coexist with different roles and the production of DNS and wall-resolved LES data is encouraged for the validation of RANS models [13]. The present thesis stays within the RANS family and asks what a learned closure can add to it.

### 1.1.2 Data-driven closures

The first data-driven approaches used high-fidelity data to correct an existing model rather than to replace it. In the field inversion and machine learning paradigm, a spatial correction to the source terms of a transport-equation model is first inferred by solving an inverse problem with the flow solver, and the map from local flow features to that correction is then learned by a regression model [14]. A related line learns the discrepancy of the modelled Reynolds stress with respect to DNS data as a function of mean-flow features, with random forests trained on one flow and applied to a different flow of the same family [15]; the discrepancy so learned improves the Reynolds stress, and its propagation to the velocity field was demonstrated on the square duct with an integrity basis of invariants as input [16]. A third line searches a space of algebraic expressions for the stress-strain relation with an evolutionary algorithm, producing explicit algebraic models that can be read and implemented as classical closures [17]. All three keep the structure of a classical model and learn a correction to it.

The second generation learns the closure itself, with the physical symmetries built into the architecture. The tensor-basis neural network of Ling, Kurzawski and Templeton embeds Galilean invariance by writing the anisotropy tensor as a combination of the integrity basis of Pope, with coefficients produced by a network that reads the invariants of the strain and rotation rates; the invariant architecture is more accurate than a generic network, and its predictions, propagated through the RANS equations, improve on both linear and nonlinear eddy-viscosity models on two test flows [4]. The same construction has been carried onto random forests, which guarantee the invariance with the same basis and produce realizable predictions close to the reference data on the square duct and the backward-facing step [18], and onto unsteady flows, where the tensor-basis network is adapted to

phase-resolved data [19]. The tensor basis is also the representation adopted in the most recent closures trained across families of bluff-body wakes [3]. Chapter 3 builds on this line, with a pointwise network and a tensor-basis network as the local closures of this work.

The data these closures learn from set what they can be asked to generalize to. Reference fields come from DNS and wall-resolved LES, increasingly produced with spectral element solvers accelerated on Graphics Processing Units (GPU) [20], and from measurements, since particle image velocimetry supplies mean fields and Reynolds stresses on geometries that no simulation can afford. The closure of the cylinder wake has been trained on a cross-validated combination of the two [21]. Generalization is then tested in one of two ways. The closure is trained on several Reynolds numbers of one geometry and evaluated at a Reynolds number outside the training range, as in the early discrepancy models on the square duct [16], or it is trained on several geometries and evaluated on one left out, as in the leave-one-shape-out protocol across bluff-body wakes [3]. This thesis uses the first protocol: every closure is trained on the cylinder wake at several Reynolds numbers and evaluated at a held-out one.

The evaluation of a learned closure has two levels, and the reviews insist on the distinction [7]. The *a priori* evaluation compares the predicted stress with the reference stress on the reference field, the same field the closure was trained on or one held out from training. The *a posteriori* evaluation inserts the closure in a flow solver and compares the resulting mean field with the reference. The two do not measure the same thing. A closure can be accurate *a priori* and useless *a posteriori*, and the reason is the subject of the next subsection.

### 1.1.3 The coupling with the solver

The difficulty was quantified by Wu and coauthors, who showed that the RANS equations with an explicit data-driven Reynolds stress can be ill-conditioned: substituting a Reynolds stress with an error below half a per cent with respect to DNS into the mean momentum equation produces velocity errors of up to thirty-five per cent, the conditioning worsens with the Reynolds number, and a local condition number is needed to capture the problem, since the global condition number of the discrete system does not [22]. The same study found that an implicit treatment of the stress, in which part of it is carried by an eddy-viscosity term absorbed into the diffusion operator, is better conditioned than the explicit treatment. Every closure that is deployed in a solver has to face this amplification, and the strategies found in the literature act on it in three ways.

The first way is numerical. The coupling of a random-forest closure needed a continuation method and a modified turbulent kinetic energy equation to be stable [18]. The closures trained across bluff-body wakes are deployed frozen in a

finite element solver stabilized by a smoothing of the input velocity gradients, by a Lipschitz constraint on the network imposed at training, by the streamline-upwind stabilization of the convective term [23] and by an under-relaxation of the predicted stress [3]. The implicit and explicit split of the stress, the filtering of the closure inputs and outputs, and the stabilization of the discrete problem are the ingredients that Chapter 3 studies one at a time for the solver of this work.

The second way is to put the solver inside the training. The field inversion paradigm does so through the adjoint of the flow solver [14], and the evolutionary search does so by scoring every candidate expression with a RANS simulation [17]. The closure obtained in this way is consistent with the solver by construction, at the price of tying the model to one discretization and of paying many forward or adjoint solves per iteration [3].

The third way removes the solver from the training and imposes the governing equations in another form. Physics-informed neural networks encode the residual of a differential equation in the loss of a network through automatic differentiation [24, 25]; applied to the incompressible Navier–Stokes equations they solve laminar and turbulent flows in the velocity-pressure and in the vorticity-velocity formulations [26], and their use for the reconstruction of flows from sparse data has been reviewed recently [27]. Their training has known failure modes, analysed through the neural tangent kernel, with the different terms of the loss converging at different rates [28]. For the closure problem, the unclosed RANS equations are imposed as a residual, the mean field and the Reynolds stress forcing are inferred from boundary and sparse data, and the closure so obtained is then embedded in a forward solver. This has been done for the cylinder wake over a range of Reynolds numbers, with a cross-validated dataset of particle image velocimetry and spectral element simulations, and the resulting closure improves the RANS prediction of both the mean flow and the turbulence statistics [21]. The same route trains closures without a solver in the loop across six bluff-body shapes, and the authors argue that removing the solver makes the training cheaper and freer, that only the final closure needs to be solver-stable, and that arbitrary closure forms are admitted without an adjoint [3]. The solver of this work follows the numerical setting of these studies and is used to measure, for each closure, how much of the *a priori* accuracy survives the coupling.

### 1.1.4 Non-locality

The closures above, classical and learned, are local: the stress at a point is a function of the mean field at that point. The evidence against this assumption is direct. From the Karman–Howarth equation for filtered fields it follows that the two-point correlation between the subfilter stress and the filtered strain rate governs the evolution of the velocity correlations, and DNS data of isotropic and channel turbulence show that local eddy-viscosity models do not reproduce the

long tails of that correlation. A stress built on a fractional gradient of the velocity, of order between zero and one, reproduces them better, which confirms a strong non-locality in the interaction between stresses and resolved deformation rates [29]. In wall-bounded flows, a non-local fractional derivative of variable order, learned from DNS data, represents the Reynolds stress across the whole mean profile, and the order has a universal form across Reynolds numbers and flow types [30]. The premise of the non-local closures is therefore physical, not only a modelling convenience.

The learned non-local closures differ in how they represent the neighbourhood. The vector-cloud neural network reads a cloud of points around the target, each carrying a feature vector, and maps the cloud to the closure variable with a network that is invariant to the translation and rotation of the frame and to the ordering of the points, so that it works on arbitrary and unstructured grids [31]. Its equivariant extension predicts tensorial quantities, with an output that co-rotates with the frame, and was evaluated *a priori* as an emulator of the Reynolds stress transport model and on DNS data [32]. The nonlocal kernel network interprets a deep network as a discrete nonlocal diffusion-reaction equation, which makes it resolution independent and lets it capture long-range dependencies [33]. The non-local closures trained across bluff-body wakes obtain their range through a transported turbulent kinetic energy with a learned length scale, and the closure with the learned length scale is the most accurate on the stress fields [3]. Chapter 4 adopts the vector-cloud construction and studies the one parameter it leaves free, the radius of the cloud.

Graph neural networks offer a second representation [34]. A mesh is a graph, and message passing along its edges is a natural way to read a neighbourhood whose extent is counted in cells. Mesh-based graph networks learn the dynamics of physical systems on meshes, adapt the discretization during the simulation and generalize across resolutions [35]. Their range is the limiting factor at high resolution, because points at equal distance in space become far apart in graph space; the multiscale extension passes messages on two resolutions, a fine one and a coarse one, and improves the accuracy while requiring less computation [36]. The equivariant graph networks of Satorras, Hoogeboom and Welling make the message passing equivariant to rotations, translations, reflections and permutations without expensive higher-order representations, by carrying a vector channel alongside the invariant node state [37]. Chapter 5 combines these three ingredients into a single-scale and a multiscale graph closure for the Reynolds stress.

### 1.1.5 Unsteady closures and the phase-resolved equations

When the flow carries an organized unsteadiness, the wake of a bluff body being the canonical case, the average that defines the closure has to be chosen with care. The

triple decomposition of Hussain and Reynolds separates the mean, the organized wave and the turbulent fluctuation, and the phase average is the operator that isolates the organized motion [38]. Phase-averaged measurements of the near wake of a circular cylinder have provided the reference picture of the entrainment and transport in the vortex street [39], and the phase-averaged turbulence properties of that wake have since been obtained also from proper orthogonal decomposition of measured fields [40]. In the phase-averaged equations the closure has to supply the turbulent stress at every phase of the cycle, and the organized motion is resolved by the equations themselves.

Whether the URANS equations do that legitimately has been questioned. Spontaneous unsteadiness in RANS simulations has been observed since the 1990s and proposed as a way of resolving large turbulent structures, but a dynamical-system analysis of the modelled equations shows that URANS can predict incorrect decay rates, that its solution tends towards the steady RANS solution over time, and that in forced turbulence the modelled energy settles at a fixed fraction of the total regardless of the model parameters [41]. The concept is questioned also from the side of the industrial practice [13]. The formulation used in this work, described in Chapter 6, sidesteps part of the objection: the organized motion is the periodic shedding of the wake, the closure is trained on phase-averaged reference data, and the closure is asked to supply the turbulent part only.

The closure of a phase-resolved flow can carry memory. In reduced-order modelling, the Mori–Zwanzig formalism represents the effect of the unresolved variables on the resolved ones as a memory functional of the resolved trajectory. A selective state-space sequence model can then be trained to predict the closure from this trajectory. Such a model has been shown to outperform Markovian and gated recurrent alternatives in both accuracy and long-term stability [42]. The sequence models available for this purpose range from the long short-term memory network [43] to the selective state-space models, which reach linear time in the sequence length [44]. For the Reynolds stress, the adaptation of the tensor-basis network to unsteady flows is recent [19]. Chapter 6 follows this direction, with rate-of-change features and a recurrent head on the coefficients of the tensor basis as the two ways of giving the closure a memory of the cycle.

## 1.2 Motivations

The survey leaves two questions open, and the thesis is organized around them. The first is whether non-locality helps. The closures of Section 1.1.4 read a neighbourhood of the point, through a cloud of fixed radius or through the graph of the mesh, and can represent the part of the stress that a local closure cannot; whether that helps is asked here on four counts: the accuracy on reference data, the

coupling with the solver, that is whether the closure can be coupled at all and what the coupling requires, the accuracy of the coupled solution, and the convergence of the iteration that produces it. The second is whether the history of the flow helps. In the phase-resolved problem the stress at one phase depends on where in the cycle the flow stands, and a closure that reads one instant cannot know it; whether a memory of the cycle helps is asked on the same four counts, with the convergence of the iteration read, for a time march, as the existence and the regularity of the limit cycle.

Non-locality is pursued at a controlled cost. The non-local closures of the literature either fix the neighbourhood by a physical radius, which has to be chosen and which may depend on the flow, or read the mesh through message passing, whose range is counted in cells and does not transfer across resolutions. This work asks how the radius of a vector cloud can be fixed before the solver starts, and how a graph closure can obtain a range set by the domain rather than by the mesh, and it measures what each choice buys inside the solver.

Memory is pursued through the coefficients of the tensor basis. This work asks whether rate-of-change features or a recurrent head on the coefficients of the tensor basis let the closure reproduce the cycle, and how much of the information the cycle needs is available in the features at all.

Answering either question on those four counts requires two things that the survey does not provide, and they are the two instrumental questions of the work. The first is the relation between the two evaluations. The reviews separate the *a priori* and the *a posteriori* accuracy, and the ill-conditioning result explains why they can differ, but the closures of the literature are usually selected on reference data and then coupled, and the comparison between the two orderings across several architectures, on the same flow and with the same solver, is rarely reported. This work evaluates every closure in both ways, at a held-out Reynolds number on reference data and at the training Reynolds number inside the solver, and asks whether a smaller error on the reference data implies a smaller error once the closure is coupled.

The second is the numerics of the coupling. The stabilizations reported in the literature are presented as ingredients of a working configuration, and the effect of each one is seldom isolated. This work builds a finite element solver in which the treatment of the closure term can be varied one ingredient at a time, the implicit and explicit split of the stress, its filtered form, the stabilization of the discrete problem, the filtering of the closure inputs and outputs, and the damping near the wall, and asks which of them a learned closure actually needs.

The thesis proceeds as follows. Chapter 2 states the averaged equations and the classical closures. Chapter 3 describes the local closures, the finite element solver and the numerical conditions under which the two can be coupled. Chapter 4 presents the vector-cloud closure and the symbolic regression that fixes its radius.

Chapter 5 presents the single-scale and the multiscale graph closures and compares all the steady closures at the same Reynolds number. Chapter 6 extends the formulation to the phase-resolved equations and to the closures with memory. Chapter 7 states what has been established and what remains to be done.

# Chapter 2

# Reynolds-Averaged Navier–Stokes Equations

From a mathematical standpoint, turbulent flows are solutions of the NSE that exhibit chaotic, multi-scale behaviour in space and time. For most flows of engineering interest, resolving all of these scales directly is computationally prohibitive. The RANS framework circumvents this cost by describing only the statistical mean of the flow, at the price of introducing additional unknown terms that must be modelled. This chapter lays the theoretical groundwork needed for the rest of the thesis. Section 2.1 briefly recalls the NSE and the mathematical setting in which they are posed. Section 2.2 derives the RANS equations through Reynolds decomposition and shows how averaging gives rise to the Reynolds stress tensor. Section 2.3 discusses the resulting closure problem, reviewing both classical eddy-viscosity models and the more recent data-driven, machine-learning-based approaches that motivate the architecture developed for this work.

## 2.1 The Navier–Stokes Equations

The NSE describe the motion of a fluid regarded as a continuum. This idealization is valid whenever the characteristic length scale of the flow is much larger than the mean free path of the constituent molecules. Macroscopic quantities such as velocity, pressure, and density can be treated as smooth fields in space and time. Under this continuum hypothesis, the equations follow from the conservation laws of mass, momentum, and energy applied to a fluid element. Restricting to a Newtonian fluid, for which the viscous stress tensor is linearly related to the rate-of-strain tensor, the constitutive relation reads

$$\boldsymbol{\tau} = \mu \left( \nabla \mathbf{u} + \nabla \mathbf{u}^T \right) + \mu' (\nabla \cdot \mathbf{u}) \, \mathbf{I}, \tag{2.1}$$

where $\mu$ is the (first, or shear) dynamic viscosity and $\mu'$ is the second coefficient of dynamic viscosity, an independent material property relating the isotropic part of the viscous stress to the local rate of volumetric expansion or compression. No relation between $\mu$ and $\mu'$ is assumed at this stage: the commonly used closure $\mu' = -\frac{2}{3}\mu$, known as Stokes' hypothesis [45], is exact only for monatomic ideal gases and is, in general, a separate empirical approximation. Since $\mu'$ plays no role once the incompressible limit is taken below, it is left unspecified here rather than fixed by that hypothesis. With this constitutive relation, the momentum and continuity equations for a general compressible Newtonian fluid read

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0, \tag{2.2}$$

$$\rho \left( \frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} \right) = -\nabla p + \nabla \cdot \left[ \mu \left( \nabla \mathbf{u} + \nabla \mathbf{u}^T \right) + \mu' (\nabla \cdot \mathbf{u}) \mathbf{I} \right] + \rho \mathbf{f}. \tag{2.3}$$

In this general compressible form, the system is not closed: an equation of state relating pressure, density, and temperature, together with the energy equation, is required to determine all thermodynamic variables. Throughout this thesis, attention is restricted to the incompressible form of Equations (2.2)–(2.3), which reduces to

$$\nabla \cdot \mathbf{u} = 0, \tag{2.4}$$

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\frac{1}{\rho} \nabla p + \nu \nabla^2 \mathbf{u} + \mathbf{f}, \tag{2.5}$$

being $\nu = \mu/\rho$ the kinematic viscosity. This simplification is not merely a mathematical convenience, but relies on specific physical conditions being satisfied by the flow under consideration.

- The first is a low-Mach-number condition. A fluid's compressibility can be quantified by a coefficient $\epsilon$ defined through $\mathrm{d}\rho/\rho = \mathrm{d}p/\epsilon$, so that a given pressure fluctuation $\mathrm{d}p$ produces a relative density change $\mathrm{d}\rho/\rho$ that is small whenever $\epsilon$ is large; the local speed of sound is related to this same coefficient by $c = \sqrt{\epsilon/\rho}$, so that equivalently $\mathrm{d}\rho/\rho = \mathrm{d}p/(\rho c^2)$. In a flow with characteristic velocity $U$, pressure fluctuations scale dynamically as $\mathrm{d}p \sim \rho U^2$, so that the relative density variation scales as $\mathrm{d}\rho/\rho \sim (U/c)^2 = Ma^2$, with $Ma = U/c$ the Mach number: a flow can therefore be treated as incompressible whenever this induced density variation is negligible, i.e., whenever $Ma \ll 1$. For $Ma \lesssim 0.3$, relative density variations remain below a few percent and the constraint $\nabla \cdot \mathbf{u} = 0$ introduces negligible error, whereas beyond this threshold compressibility effects become non-negligible and the full compressible system must be retained.

- The second condition requires the fluid to be homogeneous and to have effectively constant properties: the flow is assumed isothermal, or characterized by sufficiently small temperature variations that $\mu$ and the thermal conductivity can be treated as constants, which, together with the low-Mach-number condition, decouples the momentum equation from the energy equation, so that Equations (2.5)–(2.4) alone determine $\mathbf{u}$ and $p$.

- Finally, the continuum regime must hold, i.e., the mean free path of the fluid molecules must remain much smaller than any characteristic length scale of the flow, expressed through the Knudsen number $Kn = \lambda/L \ll 1$, so that the fluid can be described by continuous fields rather than kinetic theory.

Under these conditions, pressure ceases to be a thermodynamic state variable and instead acts as a Lagrange multiplier enforcing the divergence-free constraint at every instant; taking the divergence of Equation (2.5) yields a Poisson equation for $p$, so that the pressure field is instantaneously determined by the velocity field once suitable boundary conditions are prescribed. This elliptic character of the pressure field reflects the fact that pressure disturbances propagate at infinite speed within the incompressible approximation, consistent with the low-Mach-number assumption above. From a mathematical standpoint, Equations (2.5)–(2.4) form a nonlinear system of second-order partial differential equations, parabolic in the viscous terms and hyperbolic in the advective term.

For a comprehensive, rigorous treatment of these equations within the appropriate functional-analytic setting, as weak formulations, function spaces, and the associated existence theory, the reader is referred to Temam [46]. The question of existence and smoothness of solutions is, in fact, neither closed nor purely academic: proving existence and smoothness of solutions to the three-dimensional incompressible Navier–Stokes equations, for smooth, sufficiently decaying initial data, is one of the Clay Mathematics Institute's Millennium Prize Problems [47]. What is rigorously known can be summarized as follows. A smooth solution is guaranteed to exist locally in time, over an interval whose length may depend on the initial data; whether this interval can always be extended indefinitely (global existence and smoothness) is precisely the open part of the problem. Global existence of a *weak* solution, in the sense of Leray, is instead known unconditionally, but such weak solutions are not known to be unique, nor is their smoothness established. Partial progress toward closing this gap has come from partial-regularity theory: the set of possible space-time singularities of a suitable weak solution has been shown to be small, in the sense of a parabolic analogue of Hausdorff measure, by Caffarelli, Kohn and Nirenberg, with a simplified proof later given by Lin [47]. This open regularity question, however, has little direct bearing on the turbulence-modelling problem addressed in this thesis, and the two should not be conflated: even if global existence and smoothness were established, the practical obstacle to a direct

numerical solution of Equations (2.5)–(2.4) would remain unchanged. This obstacle is a separate, purely computational one: turbulent solutions of these equations exhibit an active range of spatial and temporal scales that grows rapidly with the Reynolds number, making their direct numerical resolution intractable at the Reynolds numbers of engineering interest. It is this practical intractability, rather than any open question of well-posedness, that motivates the statistical, averaged description developed in the next section.

## 2.2 Reynolds-Averaged Navier–Stokes Equations

This section develops the RANS equations from the Navier–Stokes equations introduced above and introduces the terminology used throughout the remainder of the thesis. The Reynolds decomposition, presented first (Section 2.2.1), splits every instantaneous flow quantity into a mean and a fluctuating component and is substituted directly into the Navier–Stokes equations to obtain the RANS momentum equation. This substitution introduces an additional unknown term, the Reynolds stress tensor, whose physical origin and the resulting closure problem are discussed next (Section 2.2.2). The section closes by examining how the choice of averaging operator – time, ensemble, or phase average – affects the interpretation of the resulting mean fields (Section 2.2.3), a distinction that underlies the separation between the steady and unsteady architectures developed later in this thesis.

### 2.2.1 Reynolds Decomposition

Rather than tracking the instantaneous, chaotic velocity and pressure fields, engineering practice is typically concerned with their statistical mean behaviour. This is not merely a matter of convenience. Resolving every active scale of a turbulent flow directly, as DNS attempts to do, demands a mesh fine enough to capture the smallest, dissipative eddies and a time step small enough to track the fastest fluctuations. At the Reynolds numbers of engineering interest, this can push even a single simulation to months of wall-clock time on the most powerful supercomputers available today. This motivates the Reynolds decomposition, in which any instantaneous flow quantity $\phi(\mathbf{x}, t)$ is split into a mean component $\overline{\phi}$ and a fluctuating component $\phi'$:

$$\phi(\mathbf{x}, t) = \overline{\phi}(\mathbf{x}, t) + \phi'(\mathbf{x}, t), \qquad \overline{\phi'} = 0. \tag{2.6}$$

The averaging operator $\overline{(\cdot)}$ may represent, depending on the context, a time average (for statistically stationary flows), an ensemble average over independent realizations, or a spatial average. In all cases it is assumed to be a Reynolds operator, satisfying

linearity and the commutation properties, i.e.,

$$\overline{\overline{\phi}} = \overline{\phi}, \qquad \overline{\frac{\partial \phi}{\partial t}} = \frac{\partial \overline{\phi}}{\partial t}, \qquad \overline{\frac{\partial \phi}{\partial x_i}} = \frac{\partial \overline{\phi}}{\partial x_i}. \tag{2.7}$$

Applying the decomposition of Equation (2.6) to the velocity and pressure fields gives $u_i = \overline{u}_i + u_i'$ and $p = \overline{p} + p'$, where $u_i$ denotes the $i$-th Cartesian component of the velocity vector $\mathbf{u}$ ($i = 1, 2, 3$). Substituting into Equations (2.5)–(2.4) and averaging yields the incompressible RANS equations (in index notation, with implicit summation over repeated indices):

$$\frac{\partial \overline{u}_i}{\partial t} + \overline{u}_j \frac{\partial \overline{u}_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \overline{p}}{\partial x_i} + \nu \frac{\partial^2 \overline{u}_i}{\partial x_j \partial x_j} - \frac{\partial \overline{u_i' u_j'}}{\partial x_j} + \overline{f}_i, \tag{2.8}$$

$$\frac{\partial \overline{u}_i}{\partial x_i} = 0. \tag{2.9}$$

### 2.2.2 The Reynolds Stress Tensor and the Closure Problem

Equation (2.8) has the same structure as the original momentum equation, with one crucial addition: the divergence of the *Reynolds stress tensor* $\boldsymbol{\tau}$. Its components are defined as

$$\tau_{ij} \equiv -\rho\, \overline{u_i' u_j'}, \qquad i, j = 1,2,3, \tag{2.10}$$

and arise entirely from the nonlinearity of the advection term, representing the mean transport of momentum by turbulent velocity fluctuations.

Being symmetric, $\tau_{ij}$ introduces six independent unknown quantities (in three dimensions) in addition to the four already present in $\overline{u}_i$ and $\overline{p}$. Since the averaging procedure provides no new equations for these components, the RANS system is not self-contained. Indeed, deriving a transport equation for $\overline{u_i' u_j'}$ from the Navier–Stokes equations introduces yet higher-order unclosed correlations, such as triple velocity products and pressure–velocity correlations. This is the well-known *turbulence closure problem*: additional modelling assumptions relating $\tau_{ij}$ to the mean flow quantities are required in order to close the system of equations. The physical modelling of $\tau_{ij}$, and specifically the design of a neural architecture able to respect its tensorial symmetries, is the central concern of this thesis and is discussed at length in the following chapters.

### 2.2.3 Steady versus Unsteady Statistics: Time, Ensemble, and Phase Averages

The averaging operator introduced in Equation (2.6) was left deliberately generic, since the exact, unclosed RANS equations do not depend on the specific Reynolds

average used to derive them [41]: any linear operator that commutes with differentiation and satisfies the idempotency property $\overline{\overline{\phi}} = \overline{\phi}$ produces the same averaged momentum equation, Equation (2.8). The choice of average does, however, have direct consequences for how the resulting mean field, and hence the Reynolds stress $\tau_{ij}$ that this thesis is concerned with modelling, should be interpreted, and this distinction is central to how the steady and unsteady architectures developed later in this thesis are set up and trained.

For flows that are statistically stationary, the natural choice is a *time average*,

$$\overline{\phi}(\mathbf{x}) = \lim_{T\to\infty} \frac{1}{T} \int_0^T \phi(\mathbf{x}, t)\, \mathrm{d}t, \tag{2.11}$$

which produces a single, time-independent mean field $\overline{\boldsymbol{u}}(\mathbf{x})$ and a single Reynolds stress field $\boldsymbol{\tau}(\mathbf{x})$ for the entire flow. This is the situation addressed by the steady architectures developed in this thesis: for a given geometry and Reynolds number, the network is trained to predict one closure field from one mean flow field, consistent with the steady RANS equations, Equations (2.8)–(2.9), solved for a time-independent solution.

However, not all flows are statistically stationary in the strict sense: the wake of a bluff body, such as the cylinder considered in this work, is dominated by the periodic, large-scale vortex-shedding phenomenon, superimposed on the smaller-scale stochastic turbulent fluctuations proper.

In this setting, the relevant average is the *phase average*. Operationally, the shedding period is first divided into $n$ discrete phases; the flow is simulated (or measured) over several complete shedding periods, each of which is subdivided into the same $n$ phases; and the flow quantity of interest is then averaged, at each phase, only over the corresponding phase index across all the simulated periods, rather than over the whole time record. The result is not a single mean field but a sequence of $n$ instantaneous snapshots, $\overline{u}_i(\mathbf{x}, t_1), \ldots, \overline{u}_i(\mathbf{x}, t_n)$, which together resolve one full period of the periodic mean motion while still averaging out the underlying stochastic fluctuations at each phase. Equations (2.8)–(2.9), closed with a phase-averaged Reynolds stress $\boldsymbol{\tau}$, then become explicitly time-dependent, and it is this formulation, commonly referred to as URANS, that motivates the unsteady architectures developed later in this work, which must predict a time-dependent closure field rather than a single static one.

A time average and a phase (or ensemble) average are, in general, distinct mathematical objects, and coincide only under specific conditions, broadly grouped under the assumption of *ergodicity*: loosely, that a sufficiently long time record of a single realization of the flow explores the same statistics as an ensemble of independent realizations, or, equivalently here, as the same phase sampled over many shedding periods. Whenever this assumption fails, for instance if the shedding process itself drifts slowly in period or amplitude over the simulated time window,

the time average and the phase average need not agree, and only the latter correctly isolates the periodic mean motion from the stochastic fluctuations.

It is worth stressing, at this point, an important caveat that will recur throughout this thesis: while the phase-average interpretation of URANS is common practice and is the one adopted here, it does not follow from the RANS equations themselves. As discussed at length by Israel [41], the RANS momentum equation is, formally, identical regardless of whether the averaging operator is interpreted as a time, ensemble, or phase average; a URANS simulation is obtained simply by solving the same, unmodified steady closure model (e.g. $k$-$\varepsilon$) in a time-marching fashion and allowing the solution to remain unsteady, without the model itself containing any explicit parameter that distinguishes resolved, periodic scales from modelled, stochastic ones. The identification of the resulting unsteady solution with a phase-averaged quantity is, in this sense, an *ex post facto* reinterpretation of a model that was not derived with any such decomposition in mind, and a rigorous theoretical justification for it remains, to date, an open question in the turbulence-modelling literature. This thesis adopts the phase-averaged interpretations for practical and organizational reasons, consistent with common practice in the vortex-shedding literature (e.g., [38] [39] [40]). but it was better to make the choice of this paradigm clear and explicit, since it is not an obvious theoretical constraint but rather an interpretive distinction related to the meaning of steady and unsteady problems.

Finally, this steady/unsteady distinction carries over differently to the two families of closure models reviewed in Section 2.3. For the machine-learning-based closures that are the focus of this thesis, the steady/unsteady character is a property of the *architecture* and its training data:

- A steady model such as the Tensor Basis Neural Network (TBNN) or the Vector-Cloud Neural Network (VCNN) family is trained on, and predicts, a single time-independent closure field per flow condition,

- An unsteady model, by contrast, must additionally represent the phase (or time) dependence of the closure field, typically by conditioning the network on a phase index or by processing a temporal sequence of mean-flow snapshots.

On the other hand, for the classical eddy-viscosity closures reviewed in Section 2.3, the time-derivative terms in the transport equations for $k$, $\varepsilon$, or $\omega$ are, in practice, never dropped, irrespective of whether the flow of interest is steady or periodic-unsteady. These models are implemented in solvers that march the governing equations forward in time and are run either to a steady state, for statistically stationary flows, or to a statistically periodic state, for vortex-shedding-dominated flows such as the one considered here, with the same underlying numerical machinery in both cases.

## 2.3 Turbulence Closure Models

Closure models can be broadly classified into two families: classical models, based on physically motivated algebraic or differential relations between $\boldsymbol{\tau}$ and the mean flow, and, more recently, data-driven models that learn this relation directly from high-fidelity data. Both are reviewed here, since the architecture developed in this thesis is best understood as an attempt to combine the physical consistency of the former with the flexibility of the latter.

### 2.3.1 Classical Eddy-Viscosity Models

The large majority of engineering closure models rest on the Boussinesq eddy-viscosity hypothesis, which postulates a linear relationship between the anisotropic part of the Reynolds stress and the mean strain-rate tensor $\overline{\mathbf{S}} = \frac{1}{2}\left(\nabla\overline{\mathbf{u}} + \nabla\overline{\mathbf{u}}^T\right)$, in direct analogy with the constitutive law of a Newtonian fluid:

$$\boldsymbol{\tau} - \frac{2}{3}k\,\mathbf{I} = 2\,\mu_t\,\overline{\mathbf{S}}, \tag{2.12}$$

where $k = \frac{1}{2}\overline{u_i' u_i'}$ is the turbulent kinetic energy and $\mu_t$ is a scalar *eddy viscosity*, which, unlike the molecular viscosity $\nu$, is not a fluid property but depends on the local state of turbulence and must itself be modelled. This assumption reduces the closure problem to that of determining a single scalar field, $\mu_t$, and forms the basis of the two-equation models that remain the workhorse of CFD.

The two most widely used members of this family are the $k$-$\varepsilon$ and $k$-$\omega$ models, presented in turn below.

The $k$-$\varepsilon$ model, originally developed by Launder and Spalding [9], obtains $\mu_t$ from two additional transported quantities: the turbulent kinetic energy $k$ and its dissipation rate $\varepsilon$, related through $\mu_t = \rho\, C_\mu\, k^2/\varepsilon$, with $C_\mu$ an empirical constant. The evolution of $k$ and $\varepsilon$ is governed by their own modelled transport equations, which include production, dissipation, and diffusion terms calibrated against canonical flows. The $k$-$\varepsilon$ model performs well for free-shear and fully turbulent flows, but is known to be less accurate near solid walls and under adverse pressure gradients, which motivated the development of alternative formulations.

The $k$-$\omega$ model, in the form popularised by Wilcox [10], instead transports $k$ together with the specific dissipation rate $\omega \sim \varepsilon/k$, giving $\mu_t = \rho\, k/\omega$. This formulation integrates more naturally through the viscous sublayer and is generally preferred for wall-bounded and separated flows, at the cost of a stronger sensitivity to the free-stream value of $\omega$. The Shear-Stress Transport (SST) blend of Menter [11] combines the two approaches, using $k$-$\omega$ near walls and $k$-$\varepsilon$ in the free stream, and is today one of the most widely adopted RANS models in industrial practice.

More elaborate closures, such as Reynolds Stress Models (RSM), abandon the Boussinesq hypothesis altogether and instead solve a modelled transport equation

directly for each component of $\boldsymbol{\tau}$, at a significantly higher computational and modelling cost.

Regardless of the specific formulation, all of these models share a common limitation: they rely on a fixed, empirically calibrated functional form, built on strong simplifying assumptions rather than on the actual, general physical behaviour of the fluid. Such behaviour, however, is in principle encoded in the data itself, a realization that has motivated the shift towards machine-learning-based approaches.

### 2.3.2 Machine-Learning-Based Closure Models

As explained in Section 1.1.2, the limitations of fixed functional forms and the availability of high-fidelity data have motivated a substantial body of work aimed at replacing or augmenting classical closures with data-driven models. Their promise is a relation between the Reynolds stress and the mean flow closer to the true one than any hand-derived expression; their limitations, several of which were encountered firsthand during the present study, are discussed in the relevant sections. The works on which this thesis builds are described here in more detail than in the survey.

#### Steady Local and Nonlocal Closures

An important line of work in this direction is due to Ling et al. [4], who proposed a TBNN that predicts the Reynolds stress anisotropy tensor as a learned linear combination of a fixed set of tensor invariants built from the mean strain-rate and rotation-rate tensors, following the classical representation theorem of Pope [12]. By construction, the coefficients of this expansion are learned as a neural network mapping of the tensor invariants, so that the resulting model automatically satisfies Galilean invariance and, crucially, rotational invariance of the predicted stress tensor with respect to the reference frame, a property that purely feature-based regression models, using raw tensor components as network inputs, generally fail to guarantee. This work established the importance of embedding known physical symmetries directly into the network architecture rather than relying on the training data to induce them empirically, a principle that strongly informs the design choices discussed later in this thesis.

Subsequent research extended this idea beyond purely local, pointwise mappings. Zhou, Han, and Xiao [31] introduced a frame-independent VCNN for nonlocal constitutive modelling, in which the closure variable at a given location is predicted from a cloud of neighbouring points, each carrying relative coordinates, velocity, and scalar flow features. Invariance to translation, rotation, and to the ordering of points within the cloud is achieved by first computing pairwise inner products of the feature vectors, and then projecting this information onto a set of learned,

permutation- and rotation-invariant embedding bases produced by a dedicated scalar embedding network, before the resulting invariant descriptor is passed to a fitting network that predicts the closure variable. This construction is directly relevant to the present work, which adopts a closely related vector-cloud approach for the case under study.

A further relevant class of nonlocal architectures for problems posed on unstructured meshes is the Graph Neural Network (GNN), first introduced by [34]. In this framework, the computational mesh is treated as a graph, with mesh points represented as nodes and their connectivity (e.g. to immediate neighbours) represented as edges. Information is propagated through the graph by message passing: for each edge, a learned edge network computes a message from the features of the two connected nodes (typically including their relative position and other local flow quantities); the incoming messages at each node are then aggregated (e.g. by summation or averaging) and fed, together with the node's own features, into a learned node-update network that produces the updated node representation. Stacking $L$ of these message-passing layers allows each node to progressively incorporate information from nodes up to $L$ edges away, so that the effective receptive field of the network grows with its depth. A key practical advantage of this construction is that the edge and node networks are shared across all edges and all nodes of the graph, respectively: the same, comparatively small set of weights is applied everywhere, which keeps the model size largely independent of the mesh resolution, makes it straightforward to train, and allows it to generalize across meshes of different size and connectivity.

This depth-dependent growth of the receptive field is, however, also the source of a well-documented limitation: as the number of message-passing layers increases, node features computed from different parts of the graph tend to become increasingly similar, a phenomenon known as *oversmoothing*, which degrades the network's ability to represent spatially localized features in its output. This is particularly problematic on fine meshes, where resolving nonlocal effects at the scale of the domain would, in principle, require many layers, yet increasing the depth beyond a fairly limited range washes out the very features the additional layers were meant to capture: with a bounded number of usable layers, each node can only gather information from a correspondingly bounded neighbourhood, however fine the mesh is.

To address this limitation, an alternative GNN-based architecture, the Multi-Scale Graph Neural Network (MS-GNN), has been proposed [36]. This architecture operates simultaneously on two graphs, one built on the original, fine mesh and one on an auxiliary, coarser mesh, and combines four distinct GNN modules: one performing message passing on the fine graph, one on the coarse graph, and two mapping networks that respectively pool node features from the fine graph onto the coarse graph and interpolate them back from the coarse graph onto the fine one.

Because the coarse graph spans the same physical domain with far fewer nodes and edges, a small number of message-passing layers on it already corresponds to a much larger effective receptive field in physical space, without requiring a correspondingly deep, oversmoothing-prone stack of layers on the fine mesh itself. This approach has been shown to restore the spatial convergence lost by standard message-passing GNNs at high mesh resolutions, at a lower computational cost than a single-scale GNN with the same number of message-passing steps [36].

A complementary strategy couples data-driven closures with physics-informed learning. Zhang et al. [21] assembled a cross-validated dataset of flow past a circular cylinder, combining hydrodynamic and aerodynamic Particle-Image-Velocimetry (PIV) measurements with high-fidelity DNS/LES across a wide range of Reynolds and Mach numbers, and applied a physics-informed correction procedure, based on a Helmholtz decomposition, to make every dataset exactly satisfy the incompressible RANS continuity and momentum equations, including an explicit Reynolds-forcing term equal to the divergence of the Reynolds-stress tensor. Analysis of this corrected dataset revealed a self-similar distribution of the Reynolds forcing across Reynolds numbers, observed consistently in both the simulated and the experimental data, which the authors used as physical justification for a data-driven closure: a compact feedforward neural network, trained on the time-averaged DNS data alone, that maps local mean-velocity gradients to the Reynolds-forcing vector and generalises well to Reynolds numbers outside its training set. Crucially, this closure is not only validated offline against held-out data but embedded directly within the unclosed RANS momentum equation and trained *a posteriori*, first within a forward Physics-Informed Neural Network (PINN) solver, and then, more significantly, within the OpenFOAM RANS solver via a discrete-adjoint procedure that differentiates through the fully converged flow solution, outperforming both a baseline eddy-viscosity model and a state-of-the-art neural-network-augmented closure. The same unclosed-RANS, physics-informed framework is further shown, in a related but distinct use, to reconstruct full velocity, pressure and Reynolds-forcing fields directly from sparse boundary measurements alone, including real experimental PIV data, without invoking the trained closure at all, illustrating the versatility of embedding governing equations directly into neural-network training.

### Unsteady, Sequence-Based Closures

The architectures reviewed so far predict a closure field from a single, time-independent set of mean-flow features, consistent with the steady RANS setting introduced in Section 2.2.3. The unsteady, phase-averaged closures developed later in this thesis instead require a model capable of representing an explicitly time-dependent closure field across a full shedding period, and this work explores two alternative sequence-modelling backbones for that purpose.

The first is the Long Short-Term Memory (LSTM) network [43], a gated variant of the Recurrent Neural Network (RNN) originally introduced to overcome the vanishing- and exploding-gradient problems that prevent plain RNNs from learning dependencies spanning more than a few time steps. At each step, an LSTM cell maintains an internal cell state that is updated, rather than simply overwritten, through three learned gates: a forget gate that decides how much of the previous cell state to discard, an input gate that decides how much new information to write in, and an output gate that decides how much of the (updated) cell state to expose as the hidden output. This additive, gated update allows gradients to be propagated across many time steps largely unattenuated, in contrast to the purely multiplicative recurrence of a vanilla RNN. Applied to phase-averaged closure modelling, an LSTM processes the sequence of per-phase mean-flow features one phase at a time and produces a correspondingly time-dependent closure prediction, naturally exploiting the fact that the Reynolds stress at a given phase of vortex shedding depends primarily on a bounded recent history rather than the entire period. Its principal drawback is that this recurrence is inherently sequential: each phase must be processed after the previous one, which limits parallelisation over the temporal dimension and can make training slower than architectures that process a full sequence at once. Closely related to the present work, Myklebust, Fuhrman, and Li [19] very recently combined a TBNN with an LSTM (TBNN–LSTM) to predict the anisotropic Reynolds-stress tensor of an unsteady, periodically forced turbulent boundary layer, an oscillatory wave boundary layer, closely analogous in spirit to the periodic vortex-shedding wake considered in this thesis, and reported substantially improved accuracy over a baseline $k$-$\omega$ RANS model across the oscillation cycle.

The second backbone considered is Mamba [44], a more recent sequence architecture based on selective structured State-Space Models (SSMs). Its starting point is a continuous-time linear dynamical system, $\dot{x} = Ax + Bu$, $y = Cx$, discretised and unrolled along the sequence; unlike earlier, time-invariant SSMs, Mamba makes the system matrices and the discretisation step themselves functions of the current input, a *selectivity* mechanism that lets the model decide, at every step, what to retain in or discard from its internal state, conceptually similar to LSTM gating but realised through a linear state-space recursion rather than a nonlinear gated cell. Because the underlying recursion is linear, it admits a hardware-efficient parallel-scan algorithm at training time, giving Mamba near-linear cost in sequence length together with a degree of parallelism across the sequence that a standard LSTM cannot offer, while retaining constant per-step memory at inference time, unlike attention-based sequence models. This makes it an attractive candidate for the relatively long phase sequences involved in resolving a full vortex-shedding period at fine temporal resolution. Its main limitations, in the present context, are its comparative novelty, with far less accumulated practice on physical time series or on time series governed by Partial Differential Equations (PDE) than

the three-decade-old LSTM, and the added implementation complexity of the selective scan relative to a standard recurrent cell. A very recent and closely related application uses Mamba to model the memory effect that unresolved variables exert on a reduced-order dynamical system: rather than deriving this memory term analytically, a Mamba-based sequence model is trained to predict it directly from the history of the resolved variables, and is then coupled, step by step, with the governing equations of the reduced system to advance the solution in time. This approach was shown to track the true dynamics substantially more accurately, and over much longer rollout horizons, than both a memory-free baseline and a Gated Recurrent Unit sequence model [42].

Taken together, these contributions frame two challenges that any machine-learning-based closure model must address:

- Ensuring that the predicted Reynolds stress tensor respects the physical symmetries of the problem: frame invariance, permutation invariance with respect to neighbouring points, and, for the stress tensor itself, positive semi-definiteness and correct equivariance under rotation of the reference frame.
- Capturing the nonlocal, history-dependent nature of turbulent transport that local eddy-viscosity closures neglect by construction.

The architectures developed in this thesis are motivated precisely by these two requirements, and build on the principles introduced by the works reviewed above, extending, combining, and actually implementing them within a working solver, and testing them on the considerably more demanding problem of the high-Reynolds-number cylinder wake.

# Chapter 3

# Machine Learning Local Closures for RANS

This chapter addresses the *local* closure problem for RANS, in which the Reynolds stress tensor at a given point is predicted purely from the mean-flow quantities available at that same point, with no explicit dependence on neighbouring points. Section 3.1 introduces the machine-learning setting common to both architectures considered here: the input features and dataset construction pipeline, the per-point loss weighting scheme, the composite loss function, and the metrics used throughout this thesis to compare model performance. Section 3.2 presents the two local architectures compared in this thesis, the Multi-Layer Perceptron (MLP) and the TBNN, together with their training results. Section 3.4 embeds both closures within the Finite Element Method (FEM) RANS solver developed for this thesis and compares the resulting velocity fields against DNS and the classical $k$-$\ell$ closure.

## 3.1 Machine Learning

Both the MLP and the TBNN architectures considered in this chapter are trained primarily in a supervised-learning scenario. In this setting, a model with a set of learnable parameters is shown many paired examples of inputs and known outputs, and its parameters are adjusted so that the model's predictions match those outputs as closely as possible. Two ingredients are needed for this: a dataset, and a way to measure how far the model's predictions are from the true outputs. Some of the experiments presented later in this chapter extend this scenario with a physics-informed loss term, which does not require a paired target at every point and is therefore not purely supervised; this distinction is noted explicitly wherever it applies.

Out of the many quantities available in the reference simulations dataset, a

subset is chosen as input features and another as the prediction target: this choice defines in a certain sense the closure problem. The input features are a set of local mean-flow and geometry quantities, specific to each architecture and detailed in Section 3.2; the target is the corresponding components of the two dimensional Reynolds stress tensor ($\overline{u'u'}$, $\overline{v'v'}$, and $\overline{u'v'}$), the unclosed term that a RANS solver cannot obtain from the mean flow alone, as discussed in Section 2.3. Both architectures are *local* closures in this sense: the target at a given point is predicted from input features evaluated at that same point alone, with no dependence on neighbouring points.

Training consists of finding the parameter values of a model that minimize a loss function, a scalar measure of the discrepancy between the model's predictions and the true targets across the dataset. This section introduces the elements of the training setup shared by both architectures: the dataset, the per-point loss weighting, and the composite loss function.

### 3.1.1 Dataset

The reference data are taken from Zhang et al. [21]: the DNS and LES simulations, togheter with the resulting mean-field data used throughout this thesis, were generated for that paper and are reused here as-is, not produced as part of this work. The simulations were run with NekRS [20], a GPU-accelerated spectral-element solver for the incompressible Navier–Stokes equations. Standard boundary conditions are used throughout:

- inlet ($\Gamma_{\text{in}}$): uniform free-stream velocity, $\mathbf{u} = (U_\infty, 0{,}0) = (1, 0{,}0)$;
- outlet ($\Gamma_{\text{out}}$): homogeneous Neumann condition on velocity, $\dfrac{\partial \mathbf{u}}{\partial x} = 0$, reference pressure $p = 0$, combined with a damping function that relaxes the flow toward its free-stream value near the outlet, a standard technique in simulations of open flows to suppress spurious reflections back into the domain;
- top and bottom lateral boundaries ($\Gamma_{\text{top}}, \Gamma_{\text{bottom}}$): free-slip, $v = 0$, $\dfrac{\partial u}{\partial y} = 0$, $\dfrac{\partial w}{\partial y} = 0$;
- cylinder wall ($\Gamma_{\text{wall}}$): no-slip, $\mathbf{u} = \mathbf{0}$;
- spanwise ($z$) direction: periodic over the finite span $L_z$ reported in Table 3.1, $\phi(x, y, 0) = \phi(x, y, L_z)$ for any field $\phi$.

All quantities can be considered non-dimensionalized with the cylinder diameter $D = 1$ and the free-stream velocity $U_\infty = 1$.

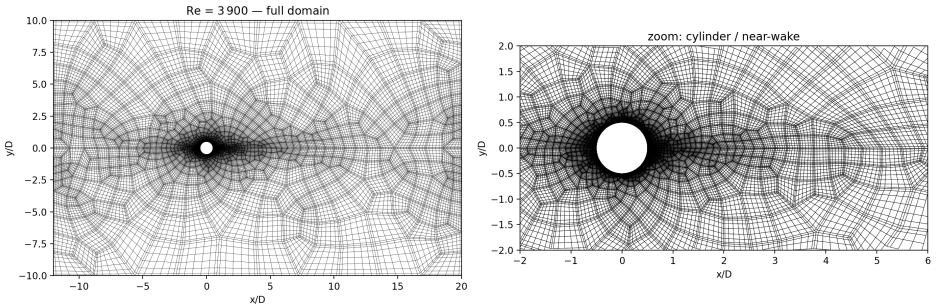
Unlike standard low-order finite-volume or finite-element discretizations, whose accuracy improves only algebraically with the number of elements ($h$-refinement), a spectral element method represents the solution within each element as a high-order Lagrange polynomial interpolant on a set of Gauss–Lobatto–Legendre nodes (polynomial order $N$, given per simulation in Table 3.1). For a sufficiently smooth solution, the resulting discretization error decreases exponentially with $N$ (spectral convergence rather than algebraic), so that a comparatively coarse element mesh can still resolve fine-scale flow structures accurately: the resolution is provided by the polynomial order within each element, not only by the number of elements itself. For a comprehensive treatment of the spectral element method, the reader is referred to Patera [48], who first introduced it, and to Karniadakis and Sherwin [49] for a general reference.

The two lower Reynolds numbers (3 900 and 11 000) are genuine DNS; the two higher ones (60 000 and 140 000) are wall-resolved LES using an entropy-viscosity subgrid-scale model. All four simulations are run in the full three-dimensional domain, statistically homogeneous in the spanwise ($z$) direction. Then are averaged both in time and along $z$ before being reported on the resulting two-dimensional $(x, y)$ mesh used for training, consistently with the steady (time-independent) two dimensional problem addressed in this chapter. Therefore, each data point stores the streamwise and cross-stream mean velocity components $\overline{u}$, $\overline{v}$, the (near-zero, by spanwise symmetry) mean spanwise velocity $\overline{w}$, the mean pressure $\overline{p}$, the components of the Reynolds stress tensor ($\overline{u'u'}$, $\overline{v'v'}$, $\overline{w'w'}$, $\overline{u'v'}$), the pressure variance $\overline{p'p'}$, and the wall shear-stress components $F_x, F_y$ on cylinder-boundary points. The four meshes are shown in Figure 3.1.

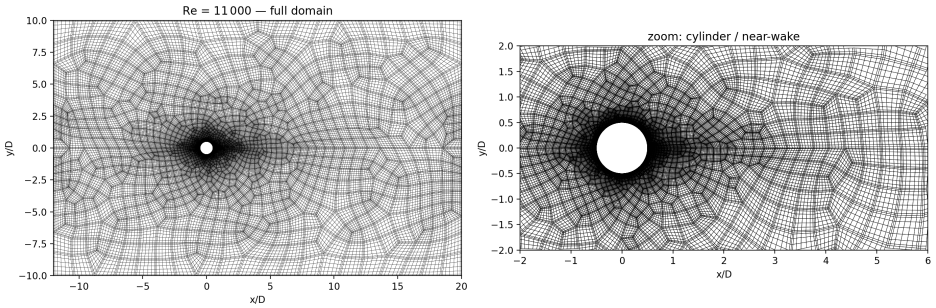
Table 3.1 summarizes the four simulations. Columns up to the Degrees Of Freedom (DOFs) are the native Spectral Element Method (SEM) simulation parameters reported by Zhang et al.; the remaining columns describe the exported two-dimensional analysis mesh actually read by the dataset builders used in this thesis. Re = 140 000 is held out entirely as the test set. The remaining three Reynolds numbers are used for training. Cell size is $\sqrt{\text{area}}$ of each quadrilateral cell of the exported mesh, in units of $D$.

**Table 3.1:** Simulation parameters (Zhang et al. [21]) and exported analysis mesh used in this thesis.

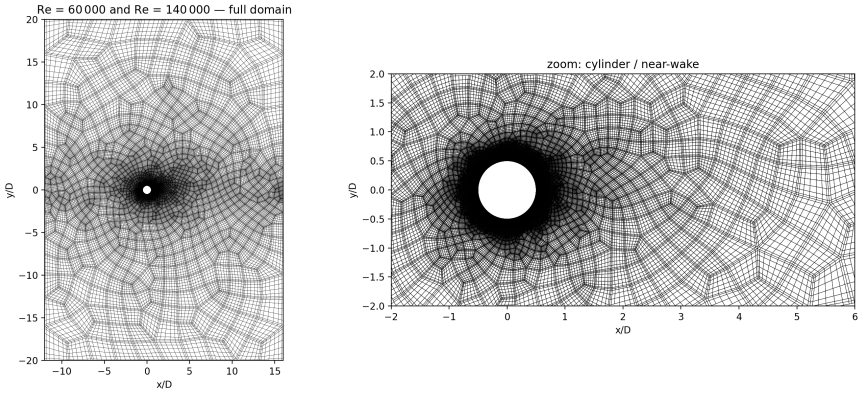
| Re | Split | Type | SEM N | $L_z/D$ | $N_z$ | $L_r/D$ | DOF (M) | Elements | Min size | Max size | Mean size |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 900 | Train | DNS | 7 | 6.4 | 64 | 0.002 | 35.3 | 78 743 | 5.1e-4 | 6.1e-1 | 4.9e-2 |
| 11 000 | Train | DNS | 7 | 6.4 | 72 | 0.001 | 54.8 | 108 780 | 3.6e-4 | 3.4e-1 | 4.9e-2 |
| 60 000 | Train | LES | 6 | 4.0 | 72 | 0.001 | 21.8 | 155 085 | 4.8e-4 | 6.4e-1 | 4.2e-2 |
| 140 000 | Test | LES | 7 | 4.0 | 48 | 0.0012 | 34.7 | 155 085 | 4.8e-4 | 6.4e-1 | 4.2e-2 |




**(a)** Re = 3900.




**(b)** Re = 11 000.




**(c)** Re = 60 000 / 140 000 (identical mesh).

**Figure 3.1:** Meshes of the reference simulations used in this thesis (data from Zhang et al. [21]).

Note that Re = 60 000 and Re = 140 000 have identical exported analysis meshes (element count, node count, and cell-size statistics), despite differing native simulation parameters (SEM order and degrees of freedom).

### 3.1.2 Per-point loss weighting

Not every point in the domain carries the same importance for training: regions near the cylinder wall, under strong adverse pressure gradients, or inside the wake velocity deficit are comparatively rare in the actual domain, yet are also the regions where the eddy-viscosity assumption underlying classical closures is most likely to break down. To give them more importance then the free-stream field, every point $i$ is assigned a training weight

$$W_i = \min\Big(1 + f_i^{\text{wall/APG}} + f_i^{\text{wake}},\ w_{\max}\Big)\,, \tag{3.1}$$

combining a near-wall/adverse-pressure-gradient term and a wake-deficit term,

$$f_i^{\text{wall/APG}} = \alpha\, e^{-\eta_i/\eta_0} \left(1 + \beta\, \frac{(\partial p/\partial t)_i^+}{\text{apg}_0}\right), \tag{3.2}$$

$$f_i^{\text{wake}} = \gamma\, \left(\frac{U_\infty - U_{x,i}}{U_\infty}\right)^+ \left(1 + \delta\, \frac{k_i}{k_0}\right), \tag{3.3}$$

where $\eta_i$ is the wall-normal distance, $(\partial p/\partial t)_i^+$ is the positive part of the tangential pressure gradient, $k_i$ is the local turbulent kinetic energy, and $\eta_0$, $\text{apg}_0$, $U_\infty$, $k_0$ are per-simulation normalization scales (respectively the 25th, 90th, 95th, and 90th percentile of the corresponding quantity over the simulation). The weight is capped at $w_{\max} = 6$ to prevent a small number of extreme points from dominating the loss, with $\alpha = \gamma = 2$ and $\beta = \delta = 1$.

The constants above were not tuned through a dedicated hyperparameter study; different choices could change the resulting per-point weights substantially. This weighting scheme is therefore better understood as an exploratory idea than as an established technique: the turbulence-modelling literature typically trains on a plain, unweighted mean-squared error, since unjustified point weights can degrade both training and generalization if they are not well motivated. Although this weighting has a physical rationale, will be shown in the following sections that it does not help consistently across *all* the architectures considered later in this thesis.

### 3.1.3 Loss function

Before defining the loss terms, the following notation is fixed.

Let $\mathbf{y}_i = (\overline{u'u'}, \overline{v'v'}, \overline{u'v'})_i \in \mathbb{R}^3$ denote the true Reynolds stress components at point $i$, and $\hat{\mathbf{y}}_i$ the corresponding network prediction, both expressed in physical (denormalized) units. Let $W_i$ be the per-point weight defined above, and $N$ the number of points in a training batch. For a generic per-point quantity $q_i$, define the weighted mean $\langle q \rangle_W \equiv \sum_i W_i q_i / \sum_i W_i$. Denote by $k_i \equiv \frac{1}{2}(\overline{u'u'}_i + \overline{v'v'}_i)$ the (in-plane) turbulent kinetic energy implied by a stress field, computed identically from $\mathbf{y}_i$ and from $\hat{\mathbf{y}}_i$.

Depending on the experiment, the network is trained with a plain mean-squared error, with an MSE-plus-PINN residual loss, or with the composite, physics-informed loss described below; the specific choice is stated for each result presented in this thesis. The composite loss was developed as an exploratory attempt to encode, as soft penalties, several physical constraints known to hold for the true Reynolds stress field, in addition to the raw per-point regression error. As shown in the results that follow, this composite loss helps substantially in some training regimes and is close to indifferent in others. It is not adopted unconditionally, and its effect on each result reported in this thesis is discussed explicitly rather than assumed.

The composite loss reads

$$\mathcal{L} = \mathcal{L}_{\text{local}} + \lambda_{\text{mag}}\mathcal{L}_{\text{mag}} + \lambda_\mu \mathcal{L}_\mu + \lambda_k \mathcal{L}_k + \lambda_+ \mathcal{L}_+ + \lambda_{uv}\mathcal{L}_{uv} + \lambda_{\text{CS}}\mathcal{L}_{\text{CS}} + \lambda_a \mathcal{L}_a, \quad (3.4)$$

with each term defined explicitly as follows.

- **Local regression term**: the weighted mean-squared error between predicted and true stress components,

$$\mathcal{L}_{\text{local}} = \left\langle \tfrac{1}{3} \textstyle\sum_c (\hat{y}_{i,c} - y_{i,c})^2 \right\rangle_W , \quad (3.5)$$

  where $c$ ranges over the three stress components.

- **Global magnitude term**: matches the root-mean-square magnitude of the predicted and true fields, component-wise,

$$\mathcal{L}_{\text{mag}} = \left\| \sqrt{\langle \hat{y}_c^2 \rangle} - \sqrt{\langle y_c^2 \rangle} \right\|^2, \quad \text{averaged over } c. \quad (3.6)$$

  The square root gives the root-mean-square amplitude of each component, computed from the squared values so that positive and negative fluctuations do not cancel; this term constrains the overall scale of the predicted field, regardless of sign, and is insensitive to whether the field is, on average, biased high or low. This is not properly the $L^2$ relative error of Section 3.1.4: it matches only the aggregate magnitude of the two fields, not their pointwise difference.

- **Global mean term**: matches the weighted mean of each predicted component to its true counterpart,

$$\mathcal{L}_\mu = \left\| \langle \hat{y}_c \rangle_W - \langle y_c \rangle_W \right\|^2, \quad \text{averaged over } c. \tag{3.7}$$

  Unlike the global magnitude term above, this is computed from the raw (signed) values, so positive and negative deviations can cancel: it is blind to the overall spread of the field and checks only for a systematic bias. The two terms are therefore complementary. A model could get the mean exactly right while predicting the wrong amplitude of variation, or vice versa, and each term alone would miss the other failure mode.

- **Energy term**: matches the mean and root-mean-square of the implied turbulent kinetic energy,

$$\mathcal{L}_k = \left( \langle \hat{k} \rangle_W - \langle k \rangle_W \right)^2 + \left( \sqrt{\langle \hat{k}^2 \rangle_W} - \sqrt{\langle k^2 \rangle_W} \right)^2. \tag{3.8}$$

  The first term matches the average predicted energy to the true one; the second matches its root-mean-square amplitude, the overall scale of its variation across the domain. $k$ governs the production–dissipation balance and sets the eddy-viscosity magnitude in the classical closures of Section 2.3, so matching both keeps the predicted turbulent energy budget physically consistent.

- **Realizability (positivity) term**: penalizes negative predicted normal stresses,

$$\mathcal{L}_+ = \frac{1}{2} \left\langle \left[ \max(0, -\hat{y}_{\overline{u'u'}}) \right]^2 + \left[ \max(0, -\hat{y}_{\overline{v'v'}}) \right]^2 \right\rangle. \tag{3.9}$$

  This is basic realizability requirement on the Reynolds stress tensor (Section 2.3); this term imposes it only indirectly, as a soft penalty in the loss, rather than as a hard structural constraint built into the architecture.

- **Shear-sign term**: a hinge loss penalizing shear-stress predictions of the wrong sign,

$$\mathcal{L}_{uv} = \left\langle \frac{|y_{\overline{u'v'}}|}{\langle |y_{\overline{u'v'}}| \rangle} \max\left(0,\, -\hat{y}_{\overline{u'v'}}\, y_{\overline{u'v'}}\right) \right\rangle, \tag{3.10}$$

- **Cauchy–Schwarz term**: penalizes violation of the realizability inequality $\overline{u'v'}^2 \leq \overline{u'u'}\,\overline{v'v'}$,

$$\mathcal{L}_{\mathrm{CS}} = \left\langle \max\left(0,\, \hat{y}^2_{\overline{u'v'}} - \hat{y}_{\overline{u'u'}}\, \hat{y}_{\overline{v'v'}}\right) \right\rangle. \tag{3.11}$$

- **Anisotropy term**: matches the predicted and true normal-stress difference,

$$\mathcal{L}_a = \left\langle \left[ (\hat{y}_{\overline{u'u'}} - \hat{y}_{\overline{v'v'}}) - (y_{\overline{u'u'}} - y_{\overline{v'v'}}) \right]^2 \right\rangle_W . \tag{3.12}$$

  $\overline{u'u'} - \overline{v'v'}$ measures how unevenly the turbulent fluctuations are distributed between the streamwise and cross-stream directions.

All inputs and targets are normalized with a global z-score before training. For a generic quantity $q$, with mean $\mu_q$ and standard deviation $\sigma_q$ computed once over the entire training set,

$$\tilde{q} = \frac{q - \mu_q}{\sigma_q}, \tag{3.13}$$

so that every normalized quantity has zero mean and unit variance over the training set; *global* refers to this single, training-set-wide statistic, not a per-Reynolds-number or per-point one. The loss terms above are evaluated after denormalizing the network output back to physical units.

These additional terms were introduced incrementally, trial after trial, in an attempt to steer training toward more physically consistent solutions. Not all of them proved equally effective across the different architectures considered in this thesis. However, one observation recurred consistently. Tuning the weights $\lambda$ to bring these terms to the same order of magnitude as the local regression term made training highly unstable and prevented convergence to a good solution. By contrast, leaving them several orders of magnitude smaller than the local term let them act as mild regularizers across several architectures. Even without matching the scale of the dominant term, they still improved results, as if gently steering the optimizer toward better solutions but if their magnitude approach that of the local term, the resulting loss landscape became too complex to optimize reliably.

### 3.1.4 Evaluation Metrics

The architectures presented in this work are compared using a common set of metrics, defined once here and used throughout the remainder of this thesis. Unless stated otherwise, metrics are computed per Reynolds stress component on the held-out test set. Let $y_i$ denote the true value of a given quantity at test point $i$, $\hat{y}_i$ the corresponding prediction, $N$ the number of test points, and $\bar{y} = \frac{1}{N}\sum_i y_i$ the mean of the true values over that same test set.

**Coefficient of determination, $R^2$:**

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}. \tag{3.14}$$

$R^2 = 1$ is a perfect prediction, zero residual error. $R^2 = 0$ means the model performs exactly as well as the trivial baseline of always predicting the test-set

mean $\bar{y}$, regardless of the input meaning it has learned nothing about how the quantity varies across the domain. $R^2 < 0$ means the model does *worse* than that trivial baseline: its predictions are actively misleading, not merely uninformative. Because $R^2$ is measured relative to the variability of the set being evaluated, it is not an absolute error scale; throughout this thesis, $R^2 \geq 0.8$ on the held-out test set is taken as a good result, a threshold set empirically from the range of values observed across the architectures presented here, not derived from first principles.

**Relative $L^2$ error:** predicted fields are compared against their true (or reference) counterparts using a relative error in the continuous $L^2(\Omega)$ sense,

$$\varepsilon_{L^2} = \frac{\|\hat{y} - y\|_{L^2(\Omega)}}{\|y\|_{L^2(\Omega)}} = \left( \frac{\int_\Omega (\hat{y} - y)^2 \, d\Omega}{\int_\Omega y^2 \, d\Omega} \right)^{1/2}, \tag{3.15}$$

On the discrete test mesh, this integral is approximated by a cell-area-weighted sum, not a plain average over test points,

$$\varepsilon_{L^2} \approx \left( \frac{\sum_{i=1}^{N} A_i \, (\hat{y}_i - y_i)^2}{\sum_{i=1}^{N} A_i \, y_i^2} \right)^{1/2}, \tag{3.16}$$

where $A_i$ is the area of the mesh cell associated with test point $i$. Weighting by cell area matters here because the mesh is highly non-uniform (Table 3.1, cell sizes spanning three orders of magnitude): without it, densely meshed regions such as the near-wall would be implicitly over-represented relative to the wake and sparser far field, and the metric would not be a faithful discretization of the domain integral above. $\varepsilon_{L^2} = 0$ is exact, while $\varepsilon_{L^2} = 1$ means the error is as large as the field being predicted. Unlike $R^2$, it has no built-in trivial-baseline reference. It is a direct, dimensionless measure of error magnitude, which makes it easier to compare directly against the solver-level errors reported later in this chapter. Because the true norm $\|y\|_{L^2(\Omega)}$ sits in the denominator, this metric is sensitive to cases where that norm is close to zero: a modest absolute error can then produce a disproportionately large, even explosive, relative value.

**Aggregate (tensor) relative error:** when a quantity has several components, such as the Reynolds stress tensor components $(\overline{u'u'}, \overline{v'v'}, \overline{u'v'})$, the same area-weighted definition is applied to the full stacked tensor at once,

$$\varepsilon_{L^2}^{\text{tensor}} = \left( \frac{\sum_{i=1}^{N} A_i \sum_c (\hat{\tau}_{i,c} - \tau_{i,c})^2}{\sum_{i=1}^{N} A_i \sum_c \tau_{i,c}^2} \right)^{1/2}, \qquad c \in \{\overline{u'u'}, \overline{v'v'}, \overline{u'v'}\}. \tag{3.17}$$

This is the single aggregate number reported for each architecture. It can differ from a simple average of the per-component relative errors, since each component is weighted by its own magnitude rather than treated as equally important.

**Root Mean Square Error (RMSE) and Mean Absolute Error (MAE):**

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_i (y_i - \hat{y}_i)^2}, \qquad \mathrm{MAE} = \frac{1}{N}\sum_i |y_i - \hat{y}_i|. \tag{3.18}$$

Both are reported in the physical units of the quantity itself, unlike $R^2$ or the relative $L^2$ error, which are dimensionless. RMSE penalizes large individual errors more than MAE, since errors are squared before averaging; comparing the two gives a rough sense of whether the error is spread evenly across the domain or concentrated in a few points.

## 3.2 Machine Learning Local Models

### 3.2.1 The easiest set up: Multi Layer Perceptron

The MLP is the simplest architecture considered in this thesis. One plain feedforward network that maps a local feature vector directly to the three in-plane Reynolds stress components ($\overline{u'u'}, \overline{v'v'}, \overline{u'v'}$). Two aspects of this mapping are worth stressing before presenting results. First, the input features are the mean-flow quantities themselves, each normalized with a global $z$-score before training and, for the wall distance $\eta$, further compressed with a $log1p$ transform to reduce its dynamic range near the wall. Therefore, there is no embedding or preprocessing designed to guarantee invariance to symmetries such as rotation of the reference frame. Second, the prediction is direct: there is no reconstruction step, and nothing in the architecture itself enforces realizability or any other physical constraint on the output. The positivity of the normal stresses and the Cauchy–Schwarz inequality between the shear and normal stresses are not architecturally guaranteed, only softly encouraged through the composite loss of Section 3.1. The following sections progressively add physical structure and complexity, and the MLP's results are the reference against which the benefit of doing so is measured.

The input feature vector comprises six quantities, $(U, V, U_x, U_y, V_x, \eta)$ (Figure 3.2): the two mean-velocity components, three components of the mean-velocity gradient, and the wall-normal distance $\eta$ from the cylinder surface. Only three of the four in-plane velocity-gradient components are included: the fourth, $V_y$, carries no independent information, since the incompressibility constraint $\nabla \cdot \mathbf{u} = 0$, Equation (2.4), fixes it exactly as $V_y = -U_x$ once $U_x$ is already given. Including it would add nothing to the input. These six features feed a feedforward network with two hidden layers of 24 neurons each (Rectified Linear Unit (ReLU) activations, 843 parameters total) and a linear output layer of size three. Section 3.2.4 reports the capacity sweep behind this choice.

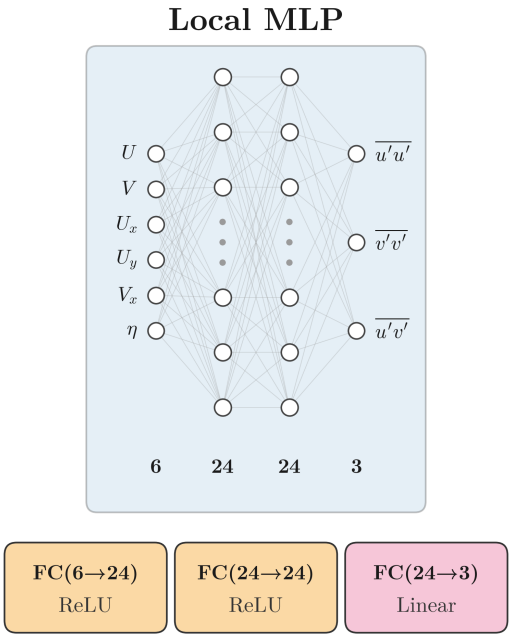



**Figure 3.2:** MLP architecture: input features, hidden layers, and output targets.

The reported training was run for 2000 epochs, starting from a learning rate of $5 \times 10^{-3}$. Optimization uses Adam, together with a step learning-rate schedule that multiplies the learning rate by 0.7 every 100 epochs. Training and validation loss history are shown in Figure 3.3, while Table 3.3 summarizes its accuracy on the held-out $\mathrm{Re} = 140\,000$ test set. These metrics compare the predicted stress components directly against the time-averaged DNS/LES targets described in Section 3.1. They say nothing yet about the accuracy of the mean-velocity field once this closure is embedded and solved for inside the FEM solver, reported separately in Section 3.4. Table 3.2 reports the same metrics on the $\mathrm{Re} = 11\,000$ training simulation alone. These are in-sample metrics: the network has already seen these points during training. This case is reported here for a specific reason. Section 3.4 embeds this same closure inside the FEM solver and solves the same $\mathrm{Re} = 11\,000$ case. The resulting *a posteriori* error will be compared against this baseline stress-field accuracy.

Figure 3.4 shows the predicted against true components.

### 3.2.2 Tensor Basis Neural Network

Whether the physical constraints are better imposed in the architecture than on the output of a network that ignores them is the question the tensor-basis network

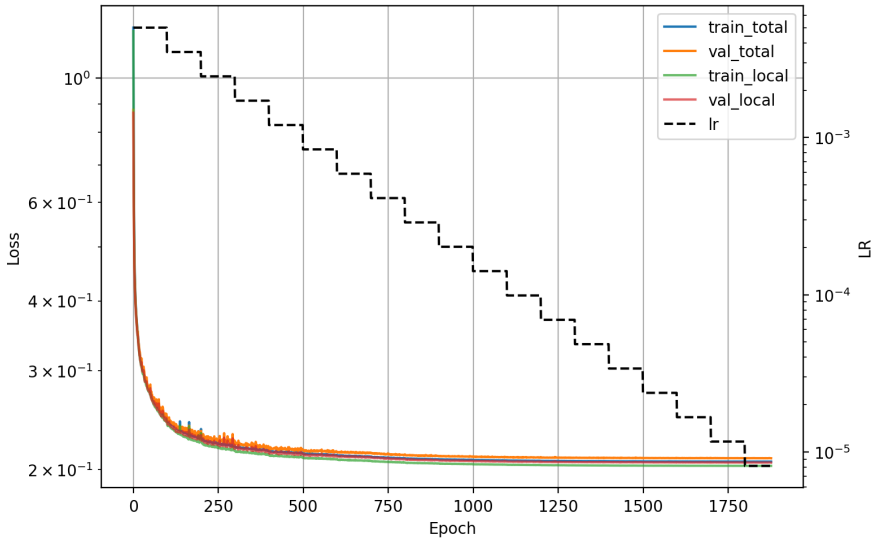



**Figure 3.3:** MLP: training and validation loss history.

**Table 3.2:** MLP results on the Re = 11 000 training simulation, evaluated in-sample.

| | $R^2$ | Rel. $L^2$ error | RMSE | MAE |
|---|---|---|---|---|
| $\overline{u'u'}$ | 0.854 | 0.375 | 0.0230 | 0.0114 |
| $\overline{v'v'}$ | 0.894 | 0.256 | 0.0351 | 0.0172 |
| $\overline{u'v'}$ | 0.827 | 0.443 | 0.0098 | 0.0048 |
| $k$ (turbulent kinetic energy) | 0.903 | 0.265 | 0.0240 | 0.0124 |

**Table 3.3:** MLP results on the held-out Re = 140 000 test set.

| | $R^2$ | Rel. $L^2$ error | RMSE | MAE |
|---|---|---|---|---|
| $\overline{u'u'}$ | 0.803 | 0.434 | 0.0357 | 0.0180 |
| $\overline{v'v'}$ | 0.786 | 0.274 | 0.0469 | 0.0248 |
| $\overline{u'v'}$ | 0.775 | 0.565 | 0.0130 | 0.0070 |
| $k$ (turbulent kinetic energy) | 0.823 | 0.305 | 0.0337 | 0.0193 |

answers by construction. Instead of predicting each Reynolds stress component directly, as the MLP closure of the previous section does, it predicts a set of scalar coefficients that multiply a fixed collection of tensors, and forms the predicted stress from their linear combination. This restructuring is adopted for two reasons.

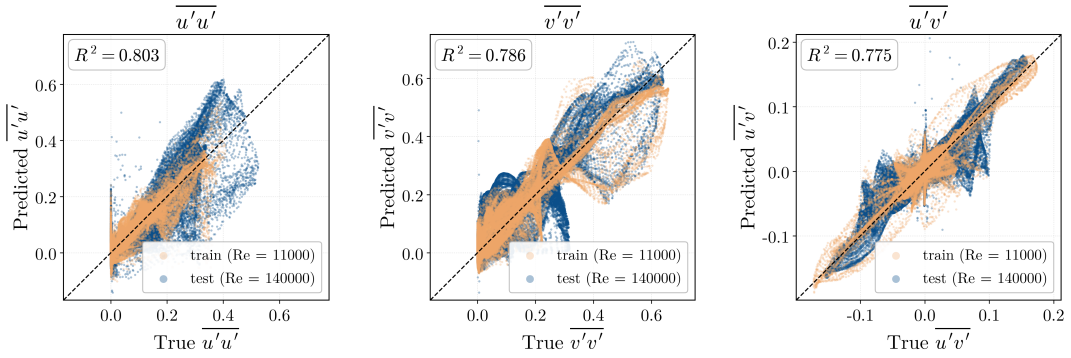



**Figure 3.4:** MLP: predicted versus true Reynolds stress components, training (Re = 11 000) and held-out (Re = 140 000) test data. Neither scatter is satisfactory, but the two point clouds are distributed similarly. This indicates that accuracy is retained at a test Reynolds number more than twice the highest seen in training (Re = 60 000). Despite its simplicity and limited accuracy, the plain MLP generalizes to unseen Reynolds numbers.

First, it embeds physical structure into the reconstruction itself, rather than leaving the network to discover it from data. The tensors entering the combination are fixed in advance and already carry the relevant physics. The network's remaining task is to weight them correctly for the local flow state, a simpler task than reproducing the tensor structure from scratch.

Second, both the individual tensors and their linear combination are rotationally equivariant. This is what guarantees, by construction, that the reconstructed Reynolds stress tensor itself transforms correctly under a rotation of the reference frame, as made precise below. A tensor valued function $\mathbf{F}$ of the mean strain rate $\mathbf{S} \equiv \frac{1}{2}(\nabla\overline{\mathbf{u}} + \nabla\overline{\mathbf{u}}^\top)$, the mean rotation rate $\mathbf{\Omega} \equiv \frac{1}{2}(\nabla\overline{\mathbf{u}} - \nabla\overline{\mathbf{u}}^\top)$, and any other frame-dependent vector argument $\mathbf{v}$ such as the mean velocity or the mean pressure gradient, is equivariant under a rotation $\mathbf{Q}$ of the reference frame if

$$\mathbf{F}(\mathbf{Q}\mathbf{S}\mathbf{Q}^\top, \mathbf{Q}\mathbf{\Omega}\mathbf{Q}^\top, \mathbf{Q}\mathbf{v}) = \mathbf{Q}\,\mathbf{F}(\mathbf{S}, \mathbf{\Omega}, \mathbf{v})\,\mathbf{Q}^\top, \tag{3.19}$$

for every orthogonal $\mathbf{Q}$. Evaluating $\mathbf{F}$ on the rotated inputs gives the same result as rotating the output of $\mathbf{F}$ evaluated on the original inputs. A scalar valued function $g$ of the same arguments is instead invariant if $g(\mathbf{Q}\mathbf{S}\mathbf{Q}^\top, \mathbf{Q}\mathbf{\Omega}\mathbf{Q}^\top, \mathbf{Q}\mathbf{v}) = g(\mathbf{S}, \mathbf{\Omega}, \mathbf{v})$, unchanged by the rotation.

Each tensor basis $\mathbf{T}_n$ used by the TBNN is built directly from $\mathbf{S}$, $\mathbf{\Omega}$, the mean velocity, and the mean pressure gradient through products and outer products, and is equivariant in the sense of Equation (3.19) by construction. The scalar coefficients $\beta_n$ predicted by the coefficient network are functions of rotation invariant features only, and are therefore themselves invariant. A linear combination of equivariant

tensors with invariant coefficients is itself equivariant. Writing $\mathbf{C} = \sum_n \beta_n \mathbf{T}_n$ and using invariance of each $\beta_n$ together with equivariance of each $\mathbf{T}_n$,

$$\begin{aligned}\mathbf{C}(\mathbf{Q}\mathbf{S}\mathbf{Q}^\top, \mathbf{Q}\boldsymbol{\Omega}\mathbf{Q}^\top, \mathbf{Q}\mathbf{v}) &= \sum_n \beta_n(\mathbf{S}, \boldsymbol{\Omega}, \mathbf{v})\, \mathbf{Q}\mathbf{T}_n(\mathbf{S}, \boldsymbol{\Omega}, \mathbf{v})\mathbf{Q}^\top \\ &= \mathbf{Q}\left(\sum_n \beta_n(\mathbf{S}, \boldsymbol{\Omega}, \mathbf{v})\, \mathbf{T}_n(\mathbf{S}, \boldsymbol{\Omega}, \mathbf{v})\right)\mathbf{Q}^\top \\ &= \mathbf{Q}\,\mathbf{C}(\mathbf{S}, \boldsymbol{\Omega}, \mathbf{v})\,\mathbf{Q}^\top,\end{aligned}$$

so the reconstructed stress transforms correctly under a change of reference frame regardless of what the coefficient network has learned.

The classical representation theorem of Pope [12] further guarantees that, for a two-dimensional mean flow, every symmetric, equivariant tensor built from $\mathbf{S}$ and $\boldsymbol{\Omega}$ alone can be written as a linear combination of only three independent tensor bases, with coefficients depending on the flow only through invariant scalar features. The reconstruction adopted here uses five bases in total, built from the raw, dimensional flow fields rather than from unit-normalized counterparts. $\mathbf{T}_1$ through $\mathbf{T}_3$ below are the classical Pope basis. $\mathbf{T}_4$ and $\mathbf{T}_5$ are two additional bases built from the mean velocity and the mean pressure gradient.

- $\mathbf{T}_1 = \mathbf{I}$, the identity tensor.
- $\mathbf{T}_2 = \mathbf{S}$, the mean strain rate tensor.
- $\mathbf{T}_3 = \mathbf{S}\boldsymbol{\Omega} - \boldsymbol{\Omega}\mathbf{S}$, the coupling between strain and rotation.
- $\mathbf{T}_4 = \overline{\mathbf{u}} \otimes \overline{\mathbf{u}} - \frac{1}{2}|\overline{\mathbf{u}}|^2\mathbf{I}$, the traceless part of the outer product of the mean velocity with itself.
- $\mathbf{T}_5 = \nabla\overline{p} \otimes \nabla\overline{p} - \frac{1}{2}|\nabla\overline{p}|^2\mathbf{I}$, the traceless part of the outer product of the mean pressure gradient with itself.

$\mathbf{T}_4$ and $\mathbf{T}_5$ are not part of the classical Pope basis. They were added empirically, after observing that including them improves the accuracy of the predicted stress relative to the three-basis formulation. This comparison, and a possible explanation for it, are discussed later in this section.

The coefficient network takes as input five rotation invariant scalar features,

$$\Big(\|\overline{\mathbf{u}}\|,\quad \eta,\quad \|\boldsymbol{\Omega}\|,\quad \|\mathbf{S}\|,\quad \|\nabla\overline{p}\|\Big), \tag{3.20}$$

the velocity magnitude, the wall-normal distance $\eta$ from the cylinder surface, and the normalized vorticity, strain rate, and pressure gradient magnitudes. Two hidden layers of 24 units each map these five features to the five scalar coefficients $\beta_1, \ldots, \beta_5$.

The five coefficients are not combined linearly. The first coefficient $\beta_1$ is passed through a softplus function, giving a non-negative isotropic scale $\sigma \geq 0$. The remaining four coefficients combine with the four traceless bases into an anisotropic part $\mathbf{a} = \beta_2\mathbf{T}_2 + \beta_3\mathbf{T}_3 + \beta_4\mathbf{T}_4 + \beta_5\mathbf{T}_5$, whose norm is capped through a hyperbolic tangent, following the realizability construction of Zhang et al. [3], so that it can never exceed $\sigma$ in magnitude. The reconstructed Reynolds stress $\widehat{\boldsymbol{\tau}}$ is the sum of $\sigma$ on the diagonal and this capped anisotropic part,

$$\widehat{\boldsymbol{\tau}} = \sigma\,\mathbf{I} + \sigma\tanh\left(\frac{\|\mathbf{a}\|}{\sigma}\right)\frac{\mathbf{a}}{\|\mathbf{a}\|}. \tag{3.21}$$

Because the function tanh is strictly smaller than one for any finite, nonzero argument, the capped anisotropic norm is always strictly smaller than $\sigma$, so the resulting tensor is symmetric positive definite for any coefficients the network can produce, without any post-hoc correction. Figure 3.5 summarizes the full pipeline, from the invariant input features of Equation (3.20) through the tensor bases and coefficient network to this reconstruction.

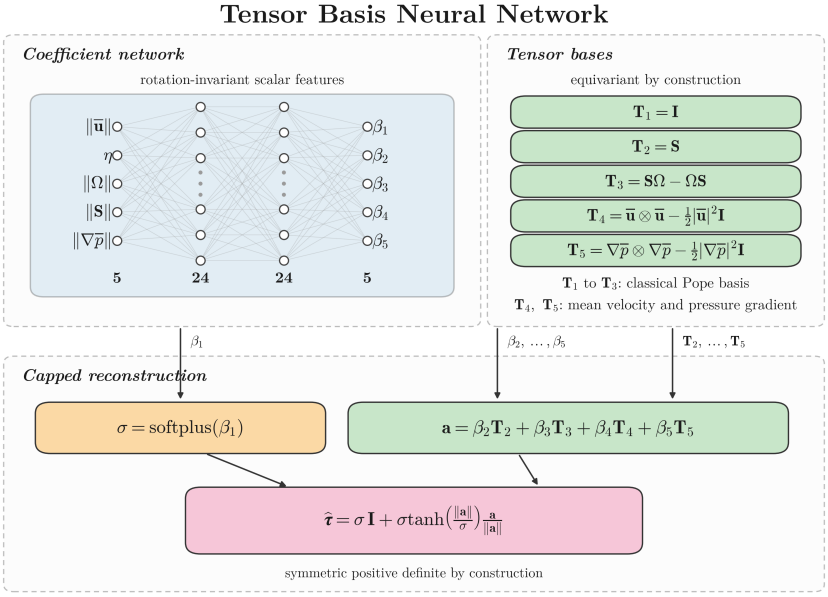



**Figure 3.5:** TBNN architecture: invariant scalar features, tensor bases, coefficient network, and the capped reconstruction of the Reynolds stress.

The alternatives to this reconstruction, and to the basis and the feature set it uses, are compared in Section 3.2.4. On the held out Reynolds number, the adopted network reaches the performance summarized in Table 3.4 and shown component by component in Figure 3.6 through their scatter plots.

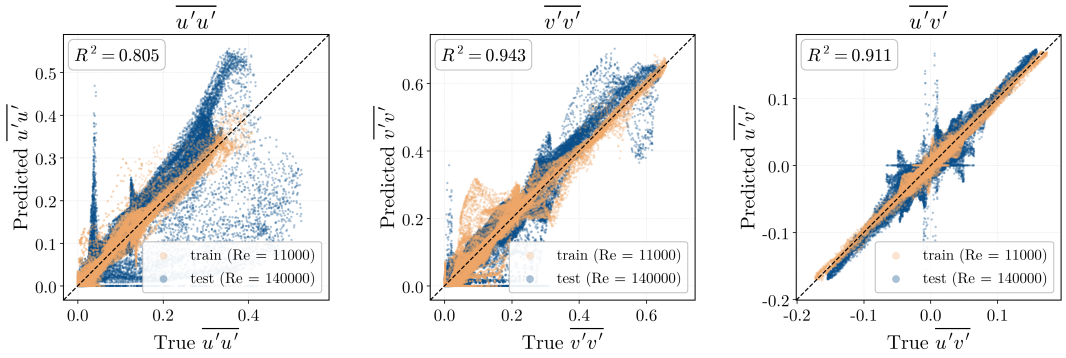



**Figure 3.6:** TBNN, adopted network: predicted versus true Reynolds stress components, held-out Re = 140 000 test set.

| | $\overline{u'u'}$ | $\overline{v'v'}$ | $\overline{u'v'}$ |
|---|---|---|---|
| $R^2$ | 0.805 | 0.944 | 0.911 |
| $\varepsilon_{L^2}^{\text{tensor}}$ (per component) | 0.215 | 0.116 | 0.173 |

**Table 3.4:** Test performance of the adopted network on the held out Reynolds number.

### 3.2.3 Physics-informed training

#### The formulation and the pointwise closure

Everything above fits the closure pointwise against stress labels. Nothing in that procedure requires the fitted field to be compatible with a momentum balance. This matters for two reasons, and the first is the one that decides the outcome. Inside the Picard iteration of Section 3.3.2 the closure is not evaluated on the reference mean field. It is evaluated on whatever the solver currently holds, and the early iterates of that sequence are not physical fields at all: they carry the artificial viscosity of Equation (3.41), they do not satisfy the momentum balance, and their gradients lie far outside the distribution the network was fitted on. The stress the closure returns there is therefore wrong for a reason no amount of supervised accuracy can repair, and the iteration either fails to contract or contracts towards a fixed point that inherits that error. The second reason is that the solver never uses the predicted stress itself. It uses its divergence, and a field can match its target closely while its derivatives do not, since fitting controls the value at each point and not the variation between neighbouring points. Zhang et al. [3] report exactly this failure for closures fitted *a priori*, which reproduce the stress well and then produce cell-scale striations once their divergence is taken inside a solver.

What both reasons ask for is the same thing: a training signal that constrains

the closure on the field the solver will actually produce, not only on the reference one. Embedding the solver in the loss would supply exactly that, at the cost of a full Picard cycle per epoch, which is prohibitive. The physics-informed formulation below is the tractable surrogate. A state network carries the mean field, the closure is required to be consistent with it through the momentum residual, and the pair is optimized together, so the closure is trained against a field that is free to move rather than against a fixed target.

A physics-informed formulation [24] addresses the objection directly by making the momentum residual part of the training objective. The construction used here follows Zhang et al. [21] and couples two networks, laid out in Figure 3.7:

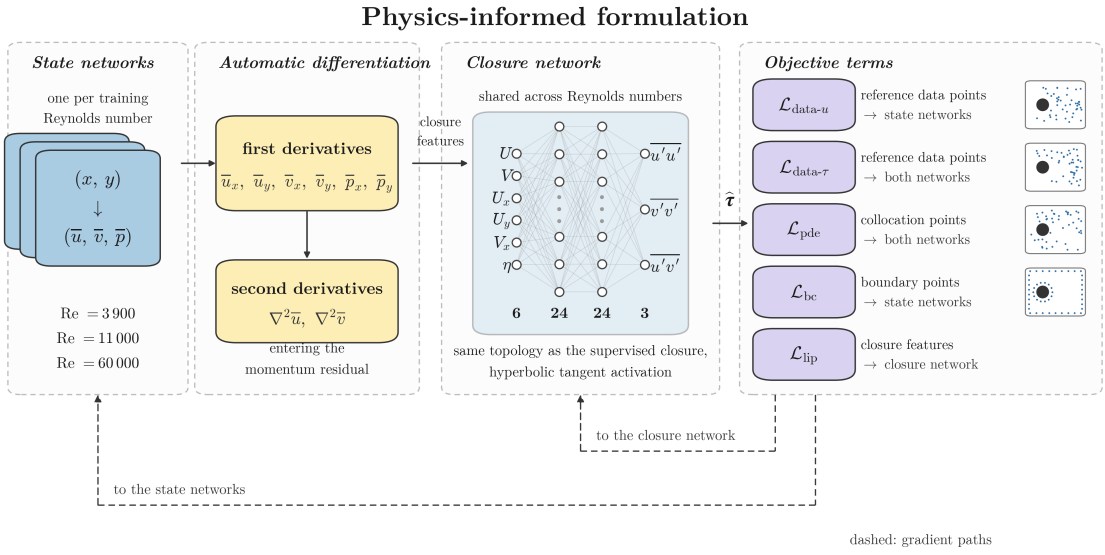



**Figure 3.7:** Physics-informed formulation. One state network per Reynolds number supplies the mean fields, and a single closure network shared across Reynolds numbers supplies the stress. The residual, boundary and data terms are evaluated on three distinct sets of points, and the momentum residual differentiates the state network twice.

- A state network maps position to the mean fields, $\mathcal{N} : (x, y) \mapsto (\overline{u}, \overline{v}, \overline{p})$. One such network is trained per training Reynolds number, since the mean field is a different function of position in each case.
- A closure network maps the local features to the Reynolds stress, exactly as in the supervised setting. A single closure network is shared across all training Reynolds numbers, since a closure is meant to be a Reynolds-independent constitutive relation.

The two are trained jointly against a loss with four contributions,

$$\mathcal{L} = \sum_{\mathrm{Re}} \Big[ w_{\mathrm{data}} \big( \mathcal{L}_{\text{data-}u} + \mathcal{L}_{\text{data-}\tau} \big) + w_{\mathrm{pde}}\, \mathcal{L}_{\mathrm{pde}} + w_{\mathrm{bc}}\, \mathcal{L}_{\mathrm{bc}} \Big] + \lambda_{\mathrm{lip}}\, \mathcal{L}_{\mathrm{lip}}. \tag{3.22}$$

The residual term evaluates the steady RANS equations at collocation points where no target is available,

$$r_u = \overline{u}\,\overline{u}_x + \overline{v}\,\overline{u}_y + \overline{p}_x - \nu \nabla^2 \overline{u} + \partial_x \overline{u'u'} + \partial_y \overline{u'v'}, \tag{3.23}$$

$$r_v = \overline{u}\,\overline{v}_x + \overline{v}\,\overline{v}_y + \overline{p}_y - \nu \nabla^2 \overline{v} + \partial_x \overline{u'v'} + \partial_y \overline{v'v'}, \tag{3.24}$$

$$\mathcal{L}_{\mathrm{pde}} = \big\langle r_u^2 + r_v^2 + (\overline{u}_x + \overline{v}_y)^2 \big\rangle, \tag{3.25}$$

where every derivative is taken by automatic differentiation through the two networks and the stress divergence is the derivative of the closure output. The boundary term imposes the conditions of Section 3.1 at points sampled on the domain boundary and on the cylinder,

$$\mathcal{L}_{\mathrm{bc}} = \big\langle (\overline{u}-1)^2 + \overline{v}^2 \big\rangle_{\Gamma_{\mathrm{in}}} + \big\langle \overline{p}^{\,2} \big\rangle_{\Gamma_{\mathrm{out}}} + \tfrac{1}{2} \sum_{\Gamma_{\mathrm{top}}, \Gamma_{\mathrm{bottom}}} \Big[ \big\langle \overline{u}_y^2 \big\rangle + \big\langle \overline{v}^2 \big\rangle \Big] + \big\langle \overline{u}^2 + \overline{v}^2 \big\rangle_{\Gamma_{\mathrm{wall}}}, \tag{3.26}$$

and the two data terms are mean-squared errors against the reference velocity and stress at points drawn from the reference dataset. Collocation and boundary points are resampled at every optimization step and are independent of the data points. Collocation points are drawn uniformly over the bounding box of the data with the cylinder interior rejected, two thousand and forty-eight per Reynolds number, matching the number of data points. Boundary points are drawn on each of the four sides of the box and on the cylinder, five hundred and twelve per segment.

One ingredient of the reference formulation is implemented here and is used with the MLP closure but not with the tensor-basis closure. The momentum residual can be enforced along a pseudo-time trajectory rather than instantaneously: each component is augmented by the change of the corresponding velocity since the previous update, the continuity residual is left untouched, and the added term is annealed away as training proceeds. Its purpose is to keep the optimization away from the spurious steady solutions a stationary residual admits while both networks are still far from any solution [3, 50, 51].

The last term of Equation (3.22) penalizes the sensitivity of the closure to its own inputs,

$$\mathcal{L}_{\mathrm{lip}} = \Big\langle \big\| \mathbf{J}_{\mathcal{C}} \big\|_F^2 \Big\rangle, \qquad \big( \mathbf{J}_{\mathcal{C}} \big)_{kj} = \frac{\partial\, \widehat{\tau}_k}{\partial f_j}, \tag{3.27}$$

with $\mathbf{f}$ the closure feature vector. It acts on the closure parameters alone, since the features are detached from the state network before the Jacobian is taken, so it cannot compete with the velocity data term through a shared path.

One deviation from the supervised network of Section 3.2.1 is forced rather than chosen. The activation is a hyperbolic tangent instead of a rectified linear unit. Equations (3.23)–(3.24) differentiate the state network twice, and the second derivative of a piecewise-linear activation is zero almost everywhere, so a rectified state network would make the viscous term vanish identically instead of raising an error. The closure network is differentiated once for the stress divergence, and since its features already contain first derivatives of the state network, the assembled graph contains second derivatives there as well. The closure network is otherwise the one of the previous section, with the same six features, the same two hidden layers of twenty-four units and the same parameter count, so the comparison in Table 3.5 holds the capacity of the closure fixed. It does not isolate the training strategy on its own. Besides the activation, the two settings differ in the standardization applied to the closure inputs and in the fact that the supervised network also standardizes its targets, whereas the physics-informed one works in physical units throughout because its output is differentiated inside the momentum residual.

**Table 3.5:** MLP closure trained in a supervised setting and inside a physics-informed formulation, at identical network topology and parameter count, held-out $\mathrm{Re} = 140\,000$ test set. The two settings differ also in the activation and in the standardization of the closure inputs.

| Training | $R^2_{\overline{u'u'}}$ | $R^2_{\overline{v'v'}}$ | $R^2_{\overline{u'v'}}$ |
|---|---|---|---|
| Supervised | 0.803 | 0.786 | 0.775 |
| Physics-informed | 0.597 | 0.652 | 0.171 |

The physics-informed closure is markedly less accurate *a priori*, and the loss is not uniform. The shear stress is almost entirely lost, with a coefficient of determination of 0.17 against 0.78, and Figure 3.8 shows the loss as a widening of the cloud around the diagonal rather than as a bias. This outcome is expected in part, since the two settings do not optimize the same objective and only one of them targets the stress labels directly, but the size of the gap is not explained by that alone.

A diagnostic isolates the mechanism. The state network reproduces the mean velocity components very accurately, with coefficients of determination above 0.98, yet the derivatives obtained from it by automatic differentiation degrade sharply close to the cylinder, where the mean field varies over the shortest length scale. The closure is fed those derivatives during training. Feeding it the reference derivatives instead, at deployment, gives a worse prediction than feeding it the state network's own derivatives, consistently across every training Reynolds number. The closure has therefore learned a mapping that compensates the bias of the state network rather than the underlying physical relation, and the two coincide only where that

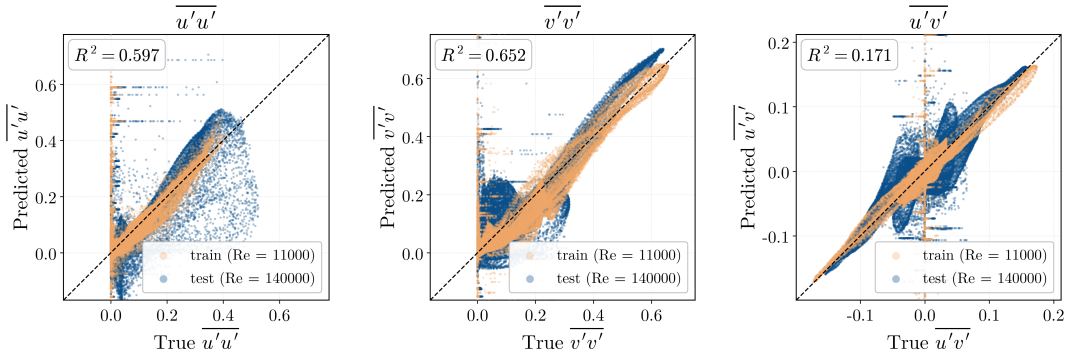



**Figure 3.8:** MLP closure trained inside the physics-informed formulation: predicted versus true Reynolds stress components. The shear stress is almost entirely lost.

bias is small. This is a distribution mismatch between training and deployment rather than overfitting in the usual sense, and it does not require the closure to have memorized anything.

The mismatch itself is left open. The residual is evaluated on derivatives obtained from a network that reproduces the field it was fitted to almost exactly, so the discrepancy is not a failure of the fit but a property of the pair formed by these data and this residual. Which of the two is at fault is not established here, and it is the single most useful thing to settle before the physics-informed route is judged: the same formulation is reported to work in the reference study [21], on the same geometry and at comparable Reynolds numbers. Figure 3.9 shows the history of the four loss terms.

### The tensor-basis closure

The formulation of Section 3.2.3 applies here unchanged in its state network, its loss terms and its sampling. Two things differ, and only one of them is a choice.

The coefficient network receives the same five invariant features of Equation (3.20), computed live from the derivatives of the state network rather than read from a dataset, and its activation is again a hyperbolic tangent for the reason given above. The reconstruction is not the one of Equation (3.21). The network emits three quantities instead of five, a turbulent kinetic energy, an eddy viscosity and one further coefficient, from which the stress is assembled following the reduced model of Zhang et al. [3],

$$k = \mathrm{softplus}(c_0), \qquad \nu_t = \mathrm{softplus}(c_1), \qquad g_2 = c_2, \tag{3.28}$$

$$\mathbf{b} = -2\nu_t\,\mathbf{S} + g_2\left(\mathbf{S\Omega} - \mathbf{\Omega S}\right), \quad \sigma = \tfrac{2}{3}k, \quad \widehat{\boldsymbol{\tau}} = \sigma\,\mathbf{I} + \sigma\tanh\!\left(\frac{\|\mathbf{b}\|_2}{\sigma}\right)\frac{\mathbf{b}}{\|\mathbf{b}\|_2}. \tag{3.29}$$

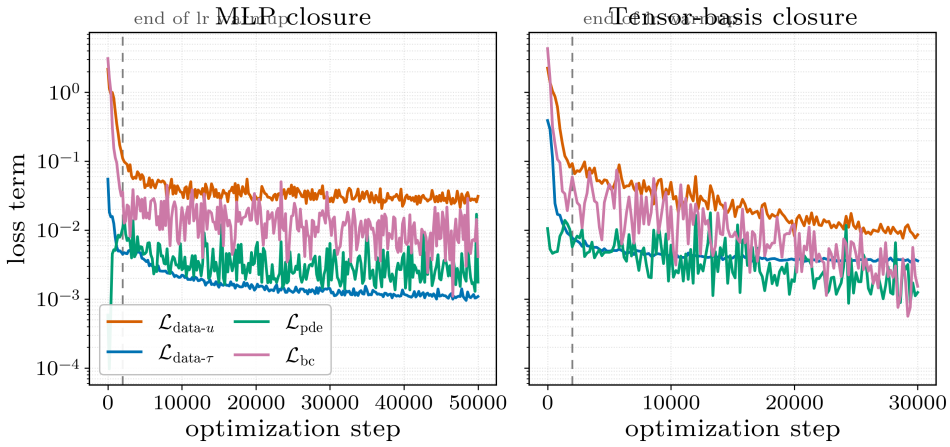



**Figure 3.9:** Physics-informed training: history of the four loss terms, reported separately. The residual term settles orders of magnitude below the terms that impose an external target.

The cap is the same as in Equation (3.21) and the tensor is again positive definite by construction. What changes is the anisotropic part, which is now a two-term expansion around an eddy-viscosity form rather than a combination of five bases. This is not a free parameter of the comparison. It is the reconstruction under which the physics-informed training converged, and it is worth stating plainly that the resulting comparison against the supervised network of Table 3.4 therefore varies two things at once, the training strategy and the stress parametrization, unlike the corresponding comparison for the MLP in Table 3.5, which varies only the first.

The training setting is the one of Section 3.2.3 and is not repeated here: the same state network per Reynolds number, the same shared closure network, the same four loss terms of Equation (3.22) summed over Reynolds numbers before a single backward pass. What changes is the closure network itself and its reconstruction, described above.

**Table 3.6:** Tensor-basis closure trained in a supervised setting and inside a physics-informed formulation, held-out $\mathrm{Re} = 140\,000$ test set. The two configurations differ in both the training strategy and the reconstruction.

| Training | $R^2_{\overline{u'u'}}$ | $R^2_{\overline{v'v'}}$ | $R^2_{\overline{u'v'}}$ | $\varepsilon^{\overline{u'u'}}_{L^2}$ | $\varepsilon^{\overline{v'v'}}_{L^2}$ | $\varepsilon^{\overline{u'v'}}_{L^2}$ |
|---|---|---|---|---|---|---|
| Supervised | 0.805 | 0.944 | 0.911 | 0.215 | 0.116 | 0.173 |
| Physics-informed | 0.660 | 0.592 | 0.389 | 2.222 | 1.187 | 0.550 |

Table 3.6 is the clearest instance in this work of the two metrics of Section 3.1.4 disagreeing, and it is worth reading carefully. The coefficients of determination are

mediocre but not absurd, and taken alone they would describe a model that explains roughly two thirds of the variance of the normal stresses. The area-weighted relative errors on those same components exceed one. Since the denominator of that metric is by definition the norm of the true field, a value above one means the prediction is further from the truth, in the area-weighted $L^2$ sense, than predicting identically zero stress everywhere. The aggregate value is 1.415.

The two readings are not contradictory. The coefficient of determination counts every test point once, while the area-weighted error counts each point in proportion to the physical area it represents, and on a mesh whose cell sizes span three orders of magnitude these two weightings are close to opposite. A value above one therefore requires the error to be concentrated in the cells carrying most of the area, which are those of the far field and the wide wake, and to be comparatively small in the densely meshed near-wall region. This is precisely the situation the area-weighted metric was introduced to expose. It is reported here rather than resolved in favour of the more flattering number, and Figure 3.10 is the direct check of the spatial reading just given.

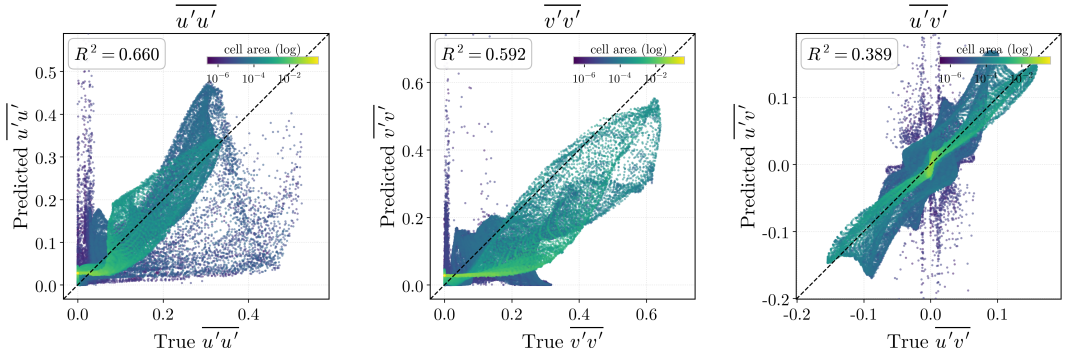



**Figure 3.10:** Tensor-basis closure trained inside the physics-informed formulation: predicted versus true Reynolds stress components, held-out test set, coloured by the area of the associated mesh cell on a logarithmic scale. The largest cells, in yellow, depart from the diagonal by little in absolute terms and by a great deal in relative terms, at a true stress close to zero. They carry 98% of the domain area and therefore dominate the area-weighted error, while the points farthest from the diagonal belong to the small cells near the cylinder and dominate instead the coefficient of determination, which counts points rather than area.

### 3.2.4 Ablations

This section collects the comparisons that fixed the two closures above. Each one changes a single ingredient of the adopted configuration and is measured with the

metrics of Section 3.1.4 on the held-out Reynolds number.

### What the closure reads

Many combinations of input features were also tried beyond the six adopted here. The wall-normal distance $\eta$ proved essential in every configuration tested, consistently the single most important feature for accuracy. Rotation and reflection invariant alternatives to the raw velocity and gradient components were also tried, following the same reasoning that motivates the tensor-basis construction of the TBNN (Section 3.2). None of these variations, however, moved the results meaningfully away from what is reported above.

The two additional bases are not there to enlarge what the closure can represent. It is fundamental to keep in mind that Pope theory already guarantees that the three bases, rescaled by some function that depends on the used invariants, span every admissible Reynolds stress tensor in two dimensions, so nothing is missing in exact arithmetic. They are there as numerical insurance. The bases are built from $\mathbf{S}$ and $\mathbf{\Omega}$ evaluated cell by cell, and in a cell where the local strain rate or vorticity is an outlier those tensors can be numerically small enough that the reconstructed stress collapses towards zero in a region where the true stress is not small. The two extra bases are built from directions that do not vanish with $\mathbf{S}$ and $\mathbf{\Omega}$, so they keep a non-degenerate direction available where the Pope set degenerates.

The price is paid on the coefficients rather than on the stresses. With five bases in a three-dimensional space the expansion is redundant, so the coefficients are no longer determined by the stress they reconstruct. The question that follows is whether the Pope invariants used before, which are enough to determine three coefficients, are still enough to determine five. The experiments below test exactly that, by enlarging the invariant set with quantities built from the pressure gradient and from the velocity, which are the two sources of information the Pope invariants do not carry.

Two configurations were tested against this question. The first extends the coefficient network input with four additional rotation invariant features that encode this orientation information, while keeping the raw bases and the capped reconstruction unchanged. Writing $\mathbf{e}_u = \overline{\mathbf{u}}/\|\overline{\mathbf{u}}\|$ and $\mathbf{e}_p = \nabla\overline{p}/\|\nabla\overline{p}\|$ for the unit velocity and pressure gradient directions, and $\omega$ for the scalar mean vorticity, the four added features are

$$s_u = \mathbf{e}_u^\top \mathbf{S}\, \mathbf{e}_u, \qquad s_p = \mathbf{e}_p^\top \mathbf{S}\, \mathbf{e}_p, \qquad c_{up} = \mathbf{e}_u \cdot \mathbf{e}_p, \qquad \chi = \mathbf{e}_u^\top \mathbf{\Omega}\, \mathbf{e}_p. \tag{3.30}$$

The first two measure how the strain rate is oriented relative to the velocity and to the pressure gradient, each fixing the corresponding angle only up to a sign. The third is the cosine of the angle between the velocity and the pressure gradient. The fourth follows the same construction as the first two with $\mathbf{\Omega}$ in place of $\mathbf{S}$,

and is equal to $\omega$ times the sine of the same angle; it is the only one of the four that changes sign under a mirror reflection of the frame. It is not established whether these four features, together, are sufficient to capture whatever directional dependence the true stress may have. They are a natural candidate, built from the same quantities already used elsewhere in the closure, not a proven complete description of the missing information.

The second configuration instead removes the two potentially ill-posed bases entirely, retaining only the identity, the strain rate tensor, and the strain-rotation coupling term, following the reduced basis set proposed by Zhang et al. [3]. The best performing configuration found for this reduced basis set is reported.

Table 3.7 compares the two against the adopted network. The extended input does not improve on the adopted network consistently. It reaches a higher $R^2$ on $\overline{u'u'}$ and a lower area weighted error on the same component, but a lower $R^2$ and a higher area weighted error on $\overline{u'v'}$. Adding the missing orientation information does not yield a consistent improvement, despite requiring nearly twice the number of input features: five invariants are not made sufficient to determine five coefficients by adding four more, and the redundancy survives the enlarged input. The reduced configuration measures what the two extra bases are worth in the other direction. Removing $\mathbf{T}_4$ and $\mathbf{T}_5$ costs eight points of $R^2$ on $\overline{u'u'}$, which is the price of giving up the numerical insurance they provide where the Pope set degenerates.

One can think that adding input features to a net of fixed parameters introduces a shortage of network capacity problem. This is not the case. Widening the coefficient network to two hidden layers of 32 units instead of 24, and, separately, deepening it to three hidden layers of 32 units, both increased this configuration's aggregate relative error, a single metric combining all three stress components rather than the per-component values of Table 3.7, from 0.319 for the two-layer, 24-unit network to 0.380 and 0.353 respectively. If the smaller network were too limited to exploit the four extra features, more capacity should have closed the gap; instead it widened it, the pattern expected when a fixed amount of data is spread across a higher dimensional input, not when the network lacks the capacity to represent the mapping. The extended configuration may need more data to realize the improvement the added features make possible in principle, not a larger network.

The reduced three basis configuration performs worse than the adopted network on every component and every metric. This is consistent with the redundancy established above. Removing $\mathbf{T}_4$ and $\mathbf{T}_5$ costs nothing wherever $\mathbf{S}$ and the vorticity are both well away from zero, since $\mathbf{T}_2$ and $\mathbf{T}_3$ already span every reconstruction they could produce there, but it removes the only remaining source of reconstruction directions exactly where $\mathbf{S}$ or the vorticity is small: $\mathbf{T}_3$ is proportional to both and vanishes whenever either does, and $\mathbf{T}_2 = \mathbf{S}$ vanishes with it. Near such points, for instance close to the wake centerline where the mean vorticity changes sign, $\mathbf{T}_4$

and $\mathbf{T}_5$ may be the only bases still able to reach the correct traceless stress. This is a plausible, and provable, partial explanation for why removing them hurts and why including them helps; it does not by itself establish how much of the adopted network's advantage it accounts for.

Taken together, these results point away from orientation ambiguity explanation and toward the structural redundancy established above. The extended input does not help, consistent with $\beta_4$ and $\beta_5$ not being well defined quantities to begin with, and the reduced basis hurts in exactly the way the redundancy argument predicts, near points where $\mathbf{S}$ or the vorticity is small. The raw, non-normalized form of the bases and the capped reconstruction still appear to matter more, for accuracy, than any hypothesis about missing orientation information.

| Configuration | $R^2_{\overline{u'u'}}$ | $R^2_{\overline{v'v'}}$ | $R^2_{\overline{u'v'}}$ | $\varepsilon^{\overline{u'u'}}_{L^2}$ | $\varepsilon^{\overline{v'v'}}_{L^2}$ | $\varepsilon^{\overline{u'v'}}_{L^2}$ |
|---|---|---|---|---|---|---|
| Adopted, 5 features | 0.805 | 0.944 | 0.911 | 0.215 | 0.116 | 0.173 |
| Extended, 9 features | 0.830 | 0.943 | 0.878 | 0.183 | 0.128 | 0.177 |
| Reduced, 3 bases | 0.724 | 0.919 | 0.879 | 0.286 | 0.159 | 0.215 |

**Table 3.7:** Accuracy of the adopted network against an extended feature set and a reduced basis set, all using the capped reconstruction.

### How the stress is rebuilt

As already noted, the network's raw output has no architectural guarantee of realizability. In particular, the main mathematical constraint is that the Reynolds stress tensor must be Positive Semi-Definite (PSD). This can be enforced *a posteriori* through a simple projection onto the space of PSD tensors: finding the closest PSD tensor in Frobenius norm. The Frobenius norm of a matrix is defined as the square root of the sum of its squared entries.

The Frobenius norm is invariant under orthogonal change of basis, so the closest PSD tensor can be found directly in the eigenbasis of the predicted tensor, where the tensor is diagonal. In this basis, any off-diagonal entry in a candidate PSD tensor only increases the distance, and the closest non-negative value to each diagonal entry is the eigenvalue itself if already non-negative, or zero otherwise [52]. Rotating back to the original basis, the closest PSD tensor keeps the same eigenvectors as the predicted tensor, with only the eigenvalues clipped to zero where negative. Because each predicted tensor is only $2 \times 2$, this eigendecomposition requires no iterative algorithm and is obtained in closed form. This post-hoc projection was applied to the trained network's held-out predictions, with no retraining involved.

Figure 3.11 shows the scatter of Figure 3.4 after the projection, and Table 3.8 shows the projection is essentially free: every accuracy metric improves slightly,

rather than trading accuracy for realizability. The improvement is modest but consistent across every metric tested. A plausible reading is that part of the raw network's non-realizable output was itself prediction noise rather than useful signal, so removing it costs nothing.

**Table 3.8:** MLP [24,24]: effect of a post-hoc eigenvalue-clipping projection onto the nearest PSD tensor, held-out Re = 140 000 test set. No retraining involved.

| | $\overline{u'u'}$ $R^2$ | $\overline{v'v'}$ $R^2$ | $\overline{u'v'}$ $R^2$ |
|---|---|---|---|
| Raw | 0.803 | 0.786 | 0.775 |
| PSD-projected | 0.808 | 0.790 | 0.781 |

This shows that some constraints can be imposed *a posteriori* without losing accuracy. Whether it is instead preferable to impose them directly in the architecture is the question the tensor-basis closure of Section 3.2.2 addresses.

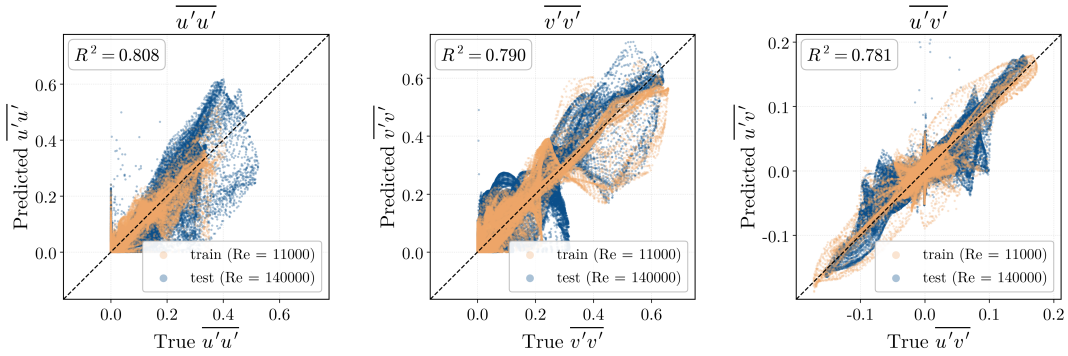



**Figure 3.11:** MLP: the same predicted versus true Reynolds stress components as Figure 3.4, after the post-hoc positive-semidefinite projection. Every $\overline{u'u'}$ and $\overline{v'v'}$ prediction is now non-negative, visible as the absence of points below zero for these two components.

The adopted network is compared against an earlier configuration that combines the same coefficients and bases linearly, following the original formulation of Ling et al. [4], using unit-normalized bases instead of their raw form. Table 3.9 reports both networks' performance on the held out Reynolds number.

Judged by $R^2$ alone the earlier configuration is more accurate on two of the three components. The area weighted relative error used throughout this chapter tells a different story: the earlier configuration's error is higher than the adopted network's on all three components. The discrepancy follows the same pattern discussed in Section 3.2.1, the coefficient of determination is dominated by the

| Configuration | $R^2_{\overline{u'u'}}$ | $R^2_{\overline{v'v'}}$ | $R^2_{\overline{u'v'}}$ | $\varepsilon^{\overline{u'u'}}_{L^2}$ | $\varepsilon^{\overline{v'v'}}_{L^2}$ | $\varepsilon^{\overline{u'v'}}_{L^2}$ |
|---|---|---|---|---|---|---|
| Standard reconstruction | 0.830 | 0.954 | 0.921 | 0.269 | 0.144 | 0.285 |
| Capped reconstruction | 0.805 | 0.944 | 0.911 | 0.215 | 0.116 | 0.173 |

**Table 3.9:** Accuracy of the adopted network against the earlier, unit-normalized-basis, linear-reconstruction configuration of Ling et al. [4].

dense mesh near the cylinder, while the area weighted error better reflects accuracy over the full domain.

Beyond accuracy, the earlier configuration does not guarantee a physically admissible stress tensor. Its predictions violate the Cauchy-Schwarz-type realizability condition $\overline{u'v'}^2 \leq \overline{u'u'}\,\overline{v'v'}$ on 36.5% of the held out domain, and predict a negative normal stress on 22.0% of the domain for $\overline{u'u'}$ and on 24.2% for $\overline{v'v'}$. A high $R^2$ does not imply a realizable prediction.

The raw, non-normalized bases make this worse in the absence of any cap. A network trained with these bases and a plain linear reconstruction fails to converge to a physically meaningful solution. It violates the realizability condition on 99.4% of the domain and predicts negative stress on over half of it. With raw bases, the capped reconstruction is not only a realizability safeguard. It is necessary for stable training.

As with the MLP (Table 3.8), a post-hoc projection onto the nearest realizable tensor can also be applied to this baseline, linear-reconstruction configuration: it removes nearly all realizability violations without degrading the evaluation metrics.

Another reconstruction attempt changes the reconstruction itself, independently of the raw or normalized form of the bases. Instead of a linear combination, the coefficients and bases build a matrix $\mathbf{A}$, and the reconstructed stress is

$$\mathbf{A} = \beta_1\mathbf{T}_1 + \beta_2\mathbf{T}_2 + \beta_3\mathbf{T}_3 + \beta_4\mathbf{T}_4 + \beta_5\mathbf{T}_5, \qquad \widehat{\boldsymbol{\tau}} = \frac{2k\,\mathbf{A}\mathbf{A}^\top}{\mathrm{tr}(\mathbf{A}\mathbf{A}^\top)},$$

proportional to $\mathbf{A}\mathbf{A}^\top$ and normalized by its trace, with an additional scalar output $k$ setting the overall scale. Since $\mathbf{A}\mathbf{A}^\top$ is symmetric positive semi-definite by construction, this form always satisfies the Cauchy-Schwarz-type condition regardless of the sign of $k$. Positive diagonal entries are not guaranteed in the same way: a negative $k$ yields a negative semi-definite tensor, which still satisfies the determinant based condition but predicts negative stress on both normal components at once. A post-hoc PSD projection, clamping $k$ to non-negative values only at deployment and without retraining, removes these violations entirely, at a marginal cost to accuracy (Table 3.10). Enforcing $k \geq 0$ throughout training instead, through a softplus reparametrization, was also attempted, but training

proved unstable across the hyperparameter configurations tried, and no usable results were obtained.

The reconstruction ultimately implemented in the adopted network instead caps the anisotropic contribution against a non-negative isotropic scale. Following Zhang et al. [3], this guarantees a symmetric positive definite tensor by construction, for every coefficient the network can output, without any deployment-only correction or retraining step, and gave good overall metrics among the attempts compared. Table 3.10 summarizes all four configurations.

| Configuration | CS viol. | neg. $\overline{u'u'}$ | neg. $\overline{v'v'}$ | $R^2_{\overline{u'u'}}$ | $R^2_{\overline{v'v'}}$ | $R^2_{\overline{u'v'}}$ |
|---|---|---|---|---|---|---|
| Earlier configuration | 36.5% | 22.0% | 24.2% | 0.830 | 0.954 | 0.921 |
| + PSD projection | 2.1% | 0.0% | 0.0% | 0.832 | 0.956 | 0.922 |
| Quadratic, raw $k$ | 0.0% | 12.8% | 12.8% | 0.813 | 0.950 | 0.921 |
| + PSD projection | 0.0% | 0.0% | 0.0% | 0.814 | 0.951 | 0.921 |
| Adopted network | 2.1% | 0.0% | 0.0% | 0.805 | 0.944 | 0.911 |

**Table 3.10:** Realizability diagnostics and accuracy of the reconstruction attempts, compared against the earlier, unconstrained configuration, on the held-out Re = 140 000 test set. CS viol. is the fraction of test points at which the predicted stress violates the Cauchy–Schwarz condition $\overline{u'v'}^2 \leq \overline{u'u'}\,\overline{v'v'}$. Neg. $\overline{u'u'}$ and neg. $\overline{v'v'}$ are the fractions at which the predicted normal stresses are negative. A realizable prediction has all three at zero.

### How the network is trained

Considerably larger networks were also trained at this stage. The best result in terms of solution accuracy were obtained overall, across every capacity tested, using three hidden layers of 128 neurons each (34 307 parameters, roughly 40 times more than the network adopted here), reaching a higher accuracy on every component (Table 3.11). Two considerations motivated adopting the smaller network instead. First, this closure is eventually embedded, and evaluated at every solver iteration, inside the FEM solver of Section 3.4; a lighter network directly reduces that cost. Second, and more importantly, the dataset used here is limited to three training Reynolds numbers, and reducing the number of parameters was found not to degrade accuracy anywhere near proportionally to the reduction in size: most of the gap between the two networks in Table 3.11 is concentrated in $\overline{v'v'}$ and $\overline{u'v'}$. Given this, a modest, deliberate accuracy trade-off in exchange for a substantially lighter model was preferred over pursuing the highest achievable accuracy in isolation.

**Table 3.11:** MLP: best network overall vs. the smaller network adopted, held-out $\mathrm{Re} = 140\,000$ test set.

| Network | Parameters | $R^2$ ($\overline{u'u'}/\overline{v'v'}/\overline{u'v'}$) | $\varepsilon_{L^2}^{\mathrm{tensor}}$ |
|---|---|---|---|
| [128,128,128] (best overall) | 34 300 | 0.828 / 0.865 / 0.836 | 0.343 |
| [24,24] (adopted) | 843 | 0.803 / 0.786 / 0.775 | 0.313 |

A sweep over training length at fixed learning rate showed no benefit from training longer and shrinking the [128,128,128] network by a factor of 14, down to [32,32,32], cost no measurable accuracy, reinforcing that network capacity is not the limiting factor for this problem.

The choice of loss function, by contrast, mattered considerably more at the much smaller [24,24] size adopted here than at larger capacities. Table 3.12 compares the composite, physics-weighted loss of Section 3.1 against a plain, unweighted mean-squared error at otherwise identical settings. The gap is large and consistent across every metric: $R^2$ on $\overline{u'u'}$ and $\overline{u'v'}$ roughly halves without the composite loss, and the turbulent-kinetic-energy $R^2$ drops from 0.82 to 0.67. This is consistent with the general observation already made in Section 3.1: even though the physics-consistency terms are individually several orders of magnitude smaller than the local regression term, they act as a form of regularization that is particularly valuable when the network has very limited capacity, biasing the optimizer toward solutions that are also globally and physically consistent.

However, Table 3.12 tells the opposite story on the $\varepsilon_{L^2}^{\mathrm{tensor}}$ error. Plain Mean Squared Error (MSE) reaches a lower value than the composite loss. This does not contradict the previous point. The two metrics answer different questions. $R^2$ measures how much of the field's variance a model explains, and that variance is concentrated almost entirely in the wake. The aggregate relative error instead sums the squared error over the domain, weighted by the physical area of each mesh cell. Its value is therefore dominated by whichever region occupies the most physical area, not by whichever region varies the most.

A visual inspection of the spatial error field explains why the two metrics disagree here. The composite loss assigns a larger per-point training weight to wall, adverse-pressure-gradient, and wake points, as described in Section 3.1. With only 24 neurons per hidden layer, the network has little spare capacity: pushed to fit the wake well, it fits the near-laminar region in front of the cylinder more poorly. The resulting front-field error is small in absolute terms, but the front covers a large fraction of the domain's area, so it still drags the area-weighted metric up. Plain MSE, with uniform point weights, does not push the network toward the

wake in this way. Capacity is spread more evenly across the domain instead. The front field comes out visibly cleaner, and the area-weighted error improves as a result. The wake, however, comes out worse, and since the wake is where most of the field's variance lives, $R^2$ drops sharply.

The composite loss is nonetheless kept as the reference configuration. This choice follows common practice in the turbulence-modelling literature, which typically evaluates a closure primarily on its accuracy in the wake. Close to the cylinder, the flow is affected by numerical singularities that data-driven closures typically struggle to represent well. The front and far field are simply not of particular interest. The area-weighted metric in Table 3.12 rewards accuracy in exactly that region, which is why it favours the plain MSE model even though the composite loss captures the wake better.

**Table 3.12:** MLP [24,24]: composite physics-weighted loss vs. plain MSE, held-out $\mathrm{Re} = 140\,000$ test set.

| Loss | $\overline{u'u'}$ $R^2$ | $\overline{v'v'}$ $R^2$ | $\overline{u'v'}$ $R^2$ | $k$ $R^2$ | $\varepsilon_{L^2}^{\text{tensor}}$ |
|---|---|---|---|---|---|
| Composite (adopted) | 0.803 | 0.786 | 0.775 | 0.823 | 0.313 |
| Plain MSE | 0.454 | 0.749 | 0.431 | 0.674 | 0.261 |

**Balancing the loss terms.** Equation (3.22) carries three weights, and the natural first move is to set them adaptively rather than by hand. The standard device rescales each group so that its gradient contribution matches the average across groups,

$$\begin{aligned} \hat{w}_i &= \frac{\bar{g}}{\|\nabla_\theta \mathcal{L}_i\| + \epsilon\,\bar{g}}, \quad \bar{g} = \frac{1}{3}\sum_j \left\|\nabla_\theta \mathcal{L}_j\right\|, \\ w_i &\leftarrow \beta w_i + (1-\beta)\,\mathrm{clip}\Big(\hat{w}_i,\ 0.01,\ 100\Big), \end{aligned} \tag{3.31}$$

applied to the three groups formed by the data, residual and boundary terms, with the weights recomputed periodically and smoothed by an exponential moving average. The premise of Equation (3.31) is that a small gradient identifies a term the optimizer is neglecting, which should therefore be amplified.

On this problem that premise is false, and the evidence is consistent across every architecture in this work.

- The residual weight always saturates upward, as Figure 3.12 shows. It reaches values between 73 and 98 while the data and boundary weights remain of order one, in every configuration tested.

- The saturation is not caused by heterogeneity across Reynolds numbers. Restricting training to a single Reynolds number, so that one state network serves one flow, still drives the residual weight to between 73 and 79.
- Decomposing the gradient by sub-network shows why. Measured on the state network, the residual term contributes a gradient norm between seventeen and thirty-five times smaller than the data and boundary terms, consistently through training. Measured on the closure network, the boundary term contributes exactly zero, which is the expected value since Equation (3.26) does not involve the closure at all, and serves as a check that the decomposition is correct.
- The small residual gradient is not an artifact of the way the residual is enforced. Residuals evaluated with and without the pseudo-time relaxation of Section 3.2.3 differ by less than five per cent, with the unrelaxed value sometimes the larger of the two.
- Across more than twenty configurations spanning three architectures, adaptive weighting never outperformed fixed weights, and in several cases collapsed the model to negative coefficients of determination.

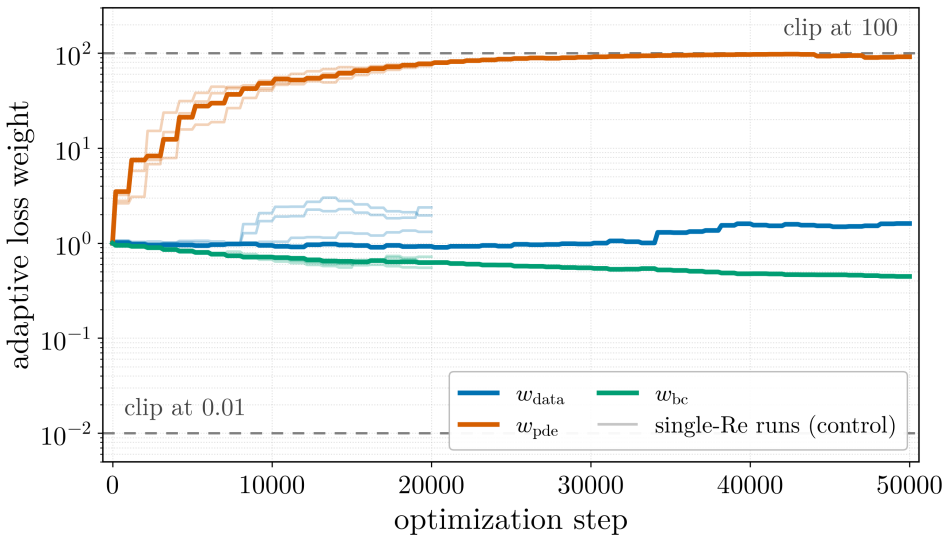



**Figure 3.12:** Adaptive weighting of the loss terms. The residual weight saturates upward while the others stay of order one. The faint curves are the control trained on a single Reynolds number, which was run for fewer optimization steps and therefore ends earlier on the same axis.

The explanation consistent with all of the above is structural rather than numerical. The residual term is a self-consistency condition. It is satisfied whenever

the state network and the closure network agree with each other, and the state field can always adjust itself to satisfy the momentum balance with whatever stress the closure happens to supply. The data and boundary terms impose fixed external targets that neither network can move. A small residual gradient therefore means that the term is already satisfied, not that it is being neglected, and amplifying it drives the pair towards a mutually consistent solution that need not be the physical one.

Fixing the weight schedule in advance, instead of inferring it from the gradients, avoids the failure mode by construction. The residual weight is ramped linearly from zero over a fixed number of steps,

$$w_{\mathrm{pde}}(s) = w_{\mathrm{pde}}^{\max} \min\Big(1,\ \frac{s}{S_{\mathrm{ramp}}}\Big), \tag{3.32}$$

with the data weight constant from the start, so that the state network is first pulled towards the data and the residual is enforced only once it describes something close to the reference field. This costs nothing, since no gradient norms are computed, and it produces no collapse. It does not, however, improve on fixed weights, reaching $R^2 = 0.607/0.566/0.181$ against $0.700/0.690/0.533$. That comparison used a fifth of the optimization steps of the fixed-weight configuration and is therefore not conclusive on its own.

**The role of the physical terms.** A final experiment removes the residual and boundary terms entirely, keeping the data terms alone, and is the most direct available test of whether those terms do anything.

Without them, the supervised part of the objective is fitted better than in any configuration where they are active. The velocity data term falls to $5.8 \times 10^{-3}$ and the stress data term to $6.3 \times 10^{-4}$, against values between $1.4 \times 10^{-2}$ and $7.7 \times 10^{-2}$ for the velocity term when the physical terms are present. Neither data availability nor network capacity is the binding constraint.

The physical residuals, which are still evaluated but carry zero weight and are therefore never corrected, grow monotonically throughout training. The momentum residual rises from 0.30 to 5.32 and the boundary residual from 4.69 to 8.76, while the data terms fall. The held-out evaluation confirms the reading, with coefficients of determination of $-0.162$, $0.453$ and $-2.578$ on the three components, so a configuration that fits its training targets an order of magnitude better than any other generalizes worse than predicting a constant. This reproduces the argument of Zhang et al. [3] on the consistency between a closure and the equations it is meant to close, with the difference that the divergence here is visible during training, in the growth of a residual that is being measured and not enforced, rather than only after deployment in a solver.

The reading that survives all of the above is that something in the pairing between these data and this residual is wrong, and that it is worth finding out what. The same formulation, on the same geometry, is the configuration the reference study reports as the one that makes everything work. Here it is the configuration that fits the data best and satisfies the physics worst. The discrepancy is stated rather than resolved, and it is left as the first thing to settle in any continuation of this work, for the reason already given in Section 3.2.3: until it is settled, a negative result on the physics-informed route is a result about this implementation and not about the route.

One qualification belongs with this result. The configuration just described is not the supervised baseline of Sections 3.2.1 and 3.2.2. It is the physics-informed formulation with its physical terms switched off, on a restricted domain and with the state network still in the loop, so the stress it is fitted against is evaluated at positions produced by that network. The supervised closures of Table 3.3 and Table 3.4 do not show this behaviour, and reach coefficients of determination between 0.78 and 0.94 on held-out data. What the experiment establishes is that the physical terms are load-bearing within the physics-informed formulation, not that supervised fitting is unsound. Figure 3.13 shows the training with the residual and boundary terms weighted to zero.

## 3.3 FEM Solver

### 3.3.1 Governing equations and function spaces

The FEM solver developed for this thesis solves the steady, incompressible RANS momentum and continuity equations for the mean velocity $\overline{\mathbf{u}} \equiv (\overline{u}, \overline{v})$ and mean pressure $\overline{p}$, closed by the Reynolds stress tensor $\boldsymbol{\tau}$ predicted by the ML closure under test. Quantities are non-dimensionalized as in Section 3.1, with $D = 1$ and $U_\infty = 1$; the kinematic viscosity is then $\nu = 1/\mathrm{Re}$. The strong form reads

$$(\overline{\mathbf{u}} \cdot \nabla)\overline{\mathbf{u}} - \nu \Delta \overline{\mathbf{u}} + \nabla \overline{p} = \nabla \cdot \boldsymbol{\tau}, \tag{3.33}$$

$$\nabla \cdot \overline{\mathbf{u}} = 0. \tag{3.34}$$

$\boldsymbol{\tau}$ is not assumed to follow the Boussinesq eddy-viscosity form here: it enters as a general symmetric tensor forcing, the same object predicted directly by the network during training (Section 3.1). This keeps the solver agnostic to the internal structure of the closure, so the same code path serves the MLP, the TBNN, and every other architecture compared later in this thesis.

The corresponding weak form is discretized with a Taylor–Hood element: velocity in $P_2$ (continuous, piecewise-quadratic, vector-valued), pressure in $P_1$ (continuous, piecewise-linear). This pair is inf-sup stable, so no separate pressure stabilization

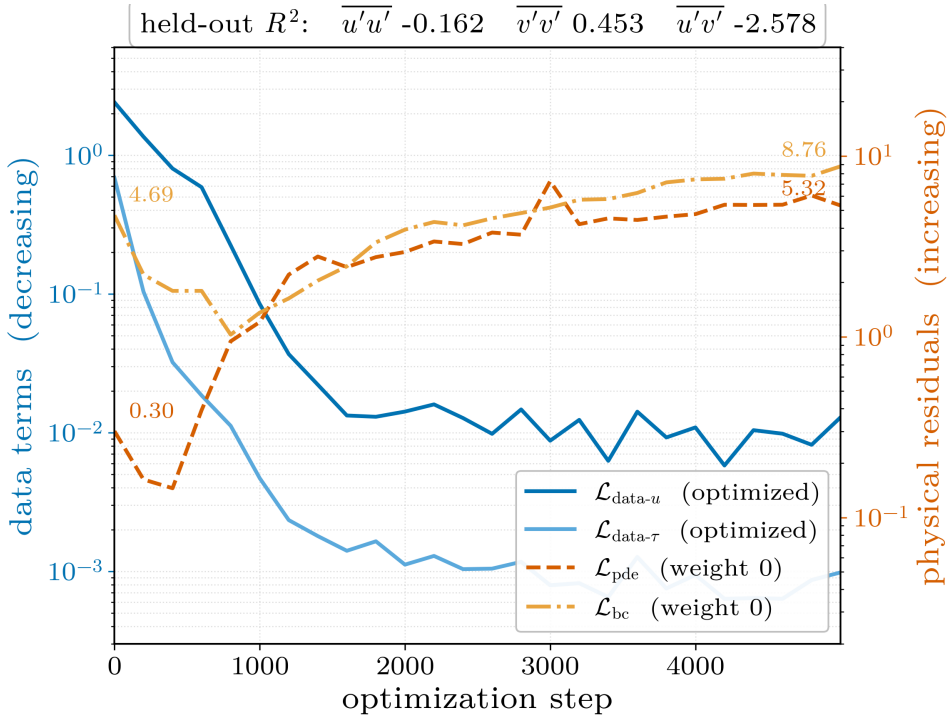



**Figure 3.13:** Training with the residual and boundary terms weighted to zero. The data terms are fitted an order of magnitude better than in any configuration where the physical terms are active, while the physical residuals, measured but not enforced, grow throughout training.

term is required. Let $W = V \times Q$ be the mixed space, $(\overline{\mathbf{u}}, \overline{p}) \in W$ the trial functions, and $(\mathbf{w}, q) \in W$ the corresponding test functions. Testing Equation (3.33) against $\mathbf{w}$ and Equation (3.34) against $q$, and integrating the pressure and viscous terms by parts, gives

$$\int_\Omega (\overline{\mathbf{u}} \cdot \nabla)\overline{\mathbf{u}} \cdot \mathbf{w} + \nu \nabla \overline{\mathbf{u}} : \nabla \mathbf{w} - \overline{p} \nabla \cdot \mathbf{w} \, d\Omega = \int_\Omega \boldsymbol{\tau} : \nabla \mathbf{w} \, d\Omega, \tag{3.35}$$

$$\int_\Omega q \nabla \cdot \overline{\mathbf{u}} \, d\Omega = 0. \tag{3.36}$$

There are two mathematically equivalent ways to write the viscous term for a Newtonian fluid. One is the Laplacian form, $\nu \Delta \overline{\mathbf{u}}$. The other is the divergence of the strain rate, $2\nu \nabla \cdot \mathbf{S}(\overline{\mathbf{u}})$, with $\mathbf{S}(\overline{\mathbf{u}}) \equiv \frac{1}{2}(\nabla \overline{\mathbf{u}} + \nabla \overline{\mathbf{u}}^\top)$ the mean strain-rate tensor. These two forms are identical only if $\nu$ is constant in space. Expanding the divergence form gives

$$2\nu \nabla \cdot \mathbf{S}(\overline{\mathbf{u}}) = \nu \Delta \overline{\mathbf{u}} + \nu \nabla (\nabla \cdot \overline{\mathbf{u}}). \tag{3.37}$$

The second term vanishes by incompressibility, Equation (3.34), leaving exactly the Laplacian form. This step requires $\nu$ to be a constant that can be pulled outside the divergence.

The molecular viscosity $\nu$ is indeed constant, so the solver uses the simpler Laplacian form for it. The eddy viscosity $\nu_{t,\text{eff}}$ produced by the closure (Section 3.3.3) is not constant: it varies from point to point. The two forms are no longer equivalent for it, since $\nu_{t,\text{eff}}$ cannot be pulled outside the divergence. Only the divergence form, $2\nu_{t,\text{eff}}\,\nabla\cdot\mathbf{S}(\overline{\mathbf{u}})$, is physically correct for a spatially varying coefficient. This is why the weak form below uses two different-looking viscous terms: the Laplacian form for the constant molecular viscosity, and the symmetric-gradient form for the variable eddy viscosity.

### 3.3.2 Picard linearization

The convective term in Equation (3.35) is nonlinear in $\overline{\mathbf{u}}$. It is linearized with a Picard fixed-point iteration, not a Newton iteration: the velocity multiplying the gradient is frozen at the previous iterate $\mathbf{u}_n$, while the velocity being differentiated remains the unknown,

$$(\overline{\mathbf{u}}\cdot\nabla)\overline{\mathbf{u}} \;\longrightarrow\; (\mathbf{u}_n\cdot\nabla)\overline{\mathbf{u}}. \tag{3.38}$$

With $\mathbf{u}_n$ frozen, Equations (3.35)–(3.36) become linear in the unknowns $(\overline{\mathbf{u}},\overline{p})$, so a single linear solve produces the next iterate. Picard is preferred to Newton here. Newton's method needs the exact derivative of the momentum residual with respect to the unknowns $(\overline{\mathbf{u}},\overline{p})$. That derivative must include how $\nabla\cdot\boldsymbol{\tau}$ changes as $\overline{\mathbf{u}}$ changes, since the closure depends on the mean-flow gradients. The closure is evaluated as a plain numerical forward pass, outside the symbolic weak form. The Unified Form Language (UFL), the symbolic language used to express the weak form in the FEM library, differentiates the rest of the equation automatically, but it cannot differentiate through this external evaluation. The exact derivative of $\boldsymbol{\tau}$ with respect to $\overline{\mathbf{u}}$ is therefore not available. Picard avoids this problem entirely: at each iteration it evaluates the closure once, on the current velocity guess $\mathbf{u}_n$, and treats the result as fixed data for that iteration's linear solve. No derivative of the closure is needed, which makes Picard a more natural fit than Newton for coupling an external, black-box closure to the momentum solve.

One Picard iteration proceeds as follows.

1. The closure is evaluated on the current velocity iterate $\mathbf{u}_n$, producing the raw Reynolds stress $\boldsymbol{\tau}$. This is split into an implicit eddy viscosity $\nu_{t,\text{eff}}$ and an explicit residual stress $\boldsymbol{\tau}_{\text{res}}$ (Section 3.3.3).

2. The resulting linear system is assembled with $\mathbf{u}_n$ frozen in the convective coefficient, in the Streamline-Upwind Petrov–Galerkin (SUPG) stabilization (Section 3.3.4), and in the closure evaluation, and solved once for $(\overline{\mathbf{u}},\overline{p})$.

3. The velocity is under-relaxed,

$$\mathbf{u}_{n+1} = (1-\alpha)\,\mathbf{u}_n + \alpha\,\overline{\mathbf{u}}, \qquad \alpha = 0.4, \tag{3.39}$$

and the closure fields $\nu_{t,\mathrm{eff}}$, $\boldsymbol{\tau}_{\mathrm{res}}$ are under-relaxed in the same way with a separate factor, $\alpha_{\mathrm{closure}} = 0.3$. This is a standard numerical tool [53] [54].

4. Convergence is checked on the relative velocity update,

$$\delta u = \frac{\|\mathbf{u}_{n+1} - \mathbf{u}_n\|}{\|\mathbf{u}_{n+1}\|}, \tag{3.40}$$

against a tolerance of $10^{-5}$. The loop runs for at most 150 iterations and never declares convergence before iteration 55, so an accidentally small early update cannot be mistaken for convergence.

"The Picard terms" are every quantity evaluated at the frozen iterate $\mathbf{u}_n$: the convection coefficient in Equation (3.38), the closure inputs $\nabla\mathbf{u}_n$, and the resulting $\nu_{t,\mathrm{eff}}(\mathbf{u}_n)$, $\boldsymbol{\tau}_{\mathrm{res}}(\mathbf{u}_n)$. Everything else, the trial functions $(\overline{\mathbf{u}}, \overline{p})$, is the genuine unknown solved for linearly at that iteration.

A further numerical device eases the first iterations, before the velocity field and the closure have stabilized: an artificial viscosity $\nu_{\mathrm{art}}$ is added to the molecular viscosity and ramped down linearly over the first 50 iterations,

$$\nu_{\mathrm{art}}(n) = \max\Big(\nu_{\mathrm{floor}},\ \nu_{\mathrm{art},0}\,(1 - n/50)\Big), \qquad \nu_{\mathrm{art},0} = 0.05,\ \ \nu_{\mathrm{floor}} = 10^{-6}. \tag{3.41}$$

This is a continuation (homotopy) strategy: the first iterations solve a more diffusive, better-conditioned problem, and $\nu_{\mathrm{art}}$ is withdrawn once the Picard loop has a reasonable velocity field to linearize around. It is a purely numerical tool [53] [55], unrelated to the physical eddy viscosity produced by the closure.

### 3.3.3 Explicit–implicit splitting of the closure term

Embedding a data-driven $\boldsymbol{\tau}$ as a fully explicit source term, evaluated only at $\mathbf{u}_n$ and added to the right-hand side, is known to be a poorly conditioned way to couple a data-driven Reynolds stress to a RANS solve: a fully explicit source has no counterpart in the momentum system's own matrix, so a small stress-prediction error has no implicit, self-correcting feedback and is amplified through the coupled solve rather than damped by it [22][21]. This thesis instead reconstructs, at every Picard iteration, the scalar eddy viscosity that best explains $\boldsymbol{\tau}$ in a least-squares sense, and keeps only the remainder explicit.

Let $\mathbf{S} = \mathbf{S}(\mathbf{u}_n)$ be the mean strain-rate tensor at the current iterate. The eddy viscosity minimizing $\|\boldsymbol{\tau} - (-2\nu_t\mathbf{S})\|_F^2$ over the scalar $\nu_t$ is

$$\nu_{t,\mathrm{eff}} = \max\left(0,\ -\frac{\boldsymbol{\tau} : \mathbf{S}}{2\,\mathbf{S} : \mathbf{S}}\right), \tag{3.42}$$

clipped to zero so that the implicit contribution is always dissipative (a negative $\nu_t$ would make the implicit operator indefinite). The remainder is defined so that the split reconstructs $\boldsymbol{\tau}$ *exactly*,

$$\boldsymbol{\tau}_{\text{res}} \equiv \boldsymbol{\tau} + 2\nu_{t,\text{eff}}\mathbf{S}, \qquad \text{so that} \qquad \boldsymbol{\tau} = -2\nu_{t,\text{eff}}\mathbf{S} + \boldsymbol{\tau}_{\text{res}}. \tag{3.43}$$

This is a post-hoc reparametrization of whatever the already-trained network outputs, not a change to the network or to $\boldsymbol{\tau}$ itself: Equation (3.43) holds pointwise as an identity. The anisotropic, non-eddy-viscosity part of the prediction, including the full shear component, is preserved intact in $\boldsymbol{\tau}_{\text{res}}$; nothing about the closure's own anisotropy is discarded, unlike a Boussinesq-only implicit treatment that would keep just an isotropic diffusivity and drop the rest of the tensor. An optional scalar $\beta_{\text{boost}}$ (default 1) rescales $\nu_{t,\text{eff}}$ before Equation (3.43) is applied, shifting how much of $\boldsymbol{\tau}$ is represented implicitly versus explicitly while leaving the reconstructed $\boldsymbol{\tau}$ unchanged.

$\nu_{t,\text{eff}}$ enters the bilinear (implicit) side of the weak form exactly like a Boussinesq eddy viscosity,

$$+\int_\Omega 2\nu_{t,\text{eff}}\,\mathbf{S}(\overline{\mathbf{u}}):\mathbf{S}(\mathbf{w})\,d\Omega, \tag{3.44}$$

while $\boldsymbol{\tau}_{\text{res}}$ replaces $\boldsymbol{\tau}$ on the right-hand side of Equation (3.35). Both quantities are under-relaxed between iterations with $\alpha_{\text{closure}} = 0.3$ (Section 3.3.2); no separate coefficient blends implicit against explicit beyond $\beta_{\text{boost}}$, since Equation (3.43) already fixes their sum. [21] and [3] report the same explicit-versus-implicit trade-off for a data-driven cylinder RANS closure and recommend an implicit treatment for deployment in practical CFD solvers, which motivates adopting the split above for all closures throughout this work.

### 3.3.4 Discretized matrix form and stabilization

Two further terms are added to the Taylor–Hood weak form for numerical stability, on top of the physical terms already introduced. A grad-div term penalizes the discrete divergence of the velocity trial and test functions,

$$+\int_\Omega \gamma_{\text{gd}}\,(\nabla\cdot\overline{\mathbf{u}})(\nabla\cdot\mathbf{w})\,d\Omega, \qquad \gamma_{\text{gd}} = 0.2, \tag{3.45}$$

which improves mass conservation and pressure robustness without affecting consistency (it vanishes on the exact, divergence-free solution).

A SUPG term stabilizes the convection-dominated regime typical of this flow,

$$+\int_\Omega \tau_{\text{S}}\left[\mathcal{A}(\overline{\mathbf{u}},\overline{p}) - \mathbf{f}_{\text{expl}}\right]\cdot(\mathbf{u}_n\cdot\nabla)\mathbf{w}\,d\Omega, \tag{3.46}$$

with $\mathcal{A}$ collecting every term of the momentum equation that depends on the unknowns and $\mathbf{f}_{\text{expl}}$ its known data, both in strong form and with $\mathbf{u}_n$ frozen,

$$\mathcal{A}(\overline{\mathbf{u}},\overline{p}) = (\mathbf{u}_n\cdot\nabla)\overline{\mathbf{u}}+\nabla\overline{p}-(\nu+\nu_{\text{art}})\Delta\overline{\mathbf{u}}-2\nu_{t,\text{eff}}\nabla\cdot\mathbf{S}(\overline{\mathbf{u}}), \qquad \mathbf{f}_{\text{expl}} = -\nabla\cdot\boldsymbol{\tau}_{\text{res}}, \tag{3.47}$$

so that the bracket is the residual of the linear problem actually solved at the current iteration. Restricting it to the convective term alone would not be admissible, since the remaining terms do not vanish at the converged solution and the stabilization would then act as a spurious body force. The parameter is

$$\tau_{\text{S}} = c_{\text{S}}\left(\frac{2|\mathbf{u}_n|}{h_p} + \frac{4(\nu+\nu_{\text{art}}+\nu_{t,\text{eff}})}{h_p^2}\right)^{-1}, \qquad h_p = \frac{h}{p}, \tag{3.48}$$

$h$ the local cell diameter, $p = 2$ the polynomial degree of the velocity space, and $c_{\text{S}}$ an explicit scale factor on the stabilization strength. The expression is the classical streamline-upwind parameter, and the subscript is written S rather than SUPG because all three stabilization forms compared in Section 3.4.2 use the same $\tau_{\text{S}}$, and only one of them is the streamline-upwind form. The value $c_{\text{S}} = 1$ is the standard calibration and the one used throughout this thesis unless stated otherwise for a specific case. Rescaling $\tau_{\text{S}}$ is legitimate because the SUPG term is *consistent*: it is built entirely from the residual of the linear problem solved at the current iterate, tested against the streamline derivative $(\mathbf{u}_n \cdot \nabla)\mathbf{w}$. At the converged solution this residual vanishes identically, so the term contributes exactly zero there, whatever positive value $c_{\text{S}}$ takes; it only changes how strongly spurious, convection-driven oscillations are damped along the way, not which solution the discrete problem converges to. In this sense $\tau_{\text{S}}$ plays a role close to that of a Lagrange multiplier attached to a soft constraint, here the vanishing of that residual at the true solution: as with a Lagrange multiplier enforcing an equality constraint, rescaling it does not move the constrained solution itself, only the conditioning and robustness of the discrete system used to reach it. This is the same structural role played by the pressure $\overline{p}$ against the divergence-free constraint in the saddle-point system below (Equation (3.57)), which is why the analogy is drawn here rather than left as a purely numerical tuning choice.

A reduced scale was used in an earlier version of this solver, and the concern that motivated it is real. If the stabilization adds enough streamline diffusion to dominate the momentum balance over both the modeled implicit eddy viscosity $\nu_{t,\text{eff}}$ and the explicit residual stress $\boldsymbol{\tau}_{\text{res}}$, the velocity fields obtained with different closures become visibly harder to tell apart, since the numerical stabilization, not the closure, sets the balance. Some SUPG is required regardless, because without it the Picard loop does not always converge. How far the scale can be reduced before convergence is lost has not been re-measured against the residual of Equation (3.47), which is not the one used when the earlier value was calibrated, so the standard calibration is retained here.

No separate Pressure-Stabilization Petrov–Galerkin (PSPG) term is needed, since the Taylor–Hood pair is already inf-sup stable.

Expanding the trial and test functions in the Taylor–Hood basis, $\overline{\mathbf{u}}_h = \sum_i U_i\,\boldsymbol{\varphi}_i$, $\mathbf{w}_h = \boldsymbol{\varphi}_k$ ($P_2$, vector-valued), and $\overline{p}_h = \sum_j P_j\,\psi_j$, $q_h = \psi_l$ ($P_1$, scalar), turns the weak form of Section 3.3.4 into an algebraic system for the velocity and pressure degree-of-freedom vectors $\mathbf{U}$, $\mathbf{P}$. Every bilinear term contributes one matrix, evaluated with $\mathbf{u}_n$ frozen at its current Picard value:

$$C(\mathbf{u}_n)_{ki} = \int_\Omega (\mathbf{u}_n \cdot \nabla)\boldsymbol{\varphi}_i \cdot \boldsymbol{\varphi}_k \, d\Omega, \tag{3.49}$$

(Picard-linearized convection)

$$K_{\nu ki} = \int_\Omega (\nu + \nu_{\text{art}})\, \nabla\boldsymbol{\varphi}_i : \nabla\boldsymbol{\varphi}_k \, d\Omega, \tag{3.50}$$

(molecular + artificial viscosity)

$$K_{\nu_t}(\mathbf{u}_n)_{ki} = \int_\Omega 2\nu_{t,\text{eff}}(\mathbf{u}_n)\, \mathbf{S}(\boldsymbol{\varphi}_i) : \mathbf{S}(\boldsymbol{\varphi}_k) \, d\Omega, \tag{3.51}$$

(implicit eddy viscosity)

$$K_{\text{gd}\,ki} = \gamma_{\text{gd}} \int_\Omega (\nabla \cdot \boldsymbol{\varphi}_i)(\nabla \cdot \boldsymbol{\varphi}_k) \, d\Omega, \tag{3.52}$$

(grad-div)

$$K_{\text{SUPG}}(\mathbf{u}_n)_{ki} = \int_\Omega \tau_{\text{S}}(\mathbf{u}_n)\, (\mathbf{u}_n \cdot \nabla)\boldsymbol{\varphi}_i \cdot (\mathbf{u}_n \cdot \nabla)\boldsymbol{\varphi}_k \, d\Omega, \tag{3.53}$$

(SUPG)

$$B_{li} = \int_\Omega \psi_l \, \nabla \cdot \boldsymbol{\varphi}_i \, d\Omega, \tag{3.54}$$

(discrete divergence)

(3.55)

with $B^\top$ the same matrix coupling pressure test functions to velocity trial functions through the $-\overline{p}\,\nabla \cdot \mathbf{w}$ term. The right-hand side collects the explicit residual stress and the SUPG residual term, both evaluated at $\mathbf{u}_n$,

$$F(\mathbf{u}_n)_k = \int_\Omega \boldsymbol{\tau}_{\text{res}}(\mathbf{u}_n) : \nabla\boldsymbol{\varphi}_k \, d\Omega - \int_\Omega \tau_{\text{S}}(\mathbf{u}_n)\, \mathbf{R}_m(\mathbf{u}_n) \cdot (\mathbf{u}_n \cdot \nabla)\boldsymbol{\varphi}_k \, d\Omega. \tag{3.56}$$

Collecting $A(\mathbf{u}_n) = C(\mathbf{u}_n) + K_\nu + K_{\nu_t}(\mathbf{u}_n) + K_{\text{gd}} + K_{\text{SUPG}}(\mathbf{u}_n)$, the system solved once per Picard iteration is the block saddle-point system

$$\begin{bmatrix} A(\mathbf{u}_n) & B^\top \\ B & 0 \end{bmatrix} \begin{bmatrix} \mathbf{U} \\ \mathbf{P} \end{bmatrix} = \begin{bmatrix} F(\mathbf{u}_n) \\ \mathbf{0} \end{bmatrix}. \tag{3.57}$$

The zero (2,2) block is the classical Stokes-type saddle-point structure: the pressure $\mathbf{P}$ has no equation of its own and instead acts as the Lagrange multiplier enforcing

the discrete divergence-free constraint $B\mathbf{U} = \mathbf{0}$ on the velocity field, the same role referenced above for $\tau_{\mathrm{S}}$. Equation (3.57) is assembled and solved as a single coupled system, rather than through a segregated velocity/pressure splitting: $\mathbf{U}$ and $\mathbf{P}$ are concatenated into one unknown vector and solved for together, which is why the method is called a monolithic solver. It is solved directly (LU factorization via MUMPS) once per Picard iteration. No Krylov iteration or Schur-complement preconditioning is used, since a 2D direct factorization at this mesh size remains affordable at every one of the outer Picard iterations.

### 3.3.5 Mesh

The computational mesh is generated with Gmsh, restricted to the half-domain $x \in [-15{,}30]D$, $y \in [0{,}15]D$. The mean flow past a cylinder is statistically symmetric about $y = 0$: this makes the restriction possible, since a symmetry boundary condition at $y = 0$ (Section 3.3.6) fully replaces the lower half of the domain without approximation, for a solver that only targets the time-averaged flow. This restriction is also a deliberate choice, for two reasons:

- It roughly halves the number of cells, and with it the size of the matrix in Equation (3.57), factorized directly at every Picard iteration: the resulting solver is substantially cheaper.
- More importantly, it removes a spurious instability. The instantaneous flow behind a cylinder in this Reynolds range sheds vortices asymmetrically, alternating above and below the wake centerline. Only its time average is symmetric. A steady solver run on the full domain, with no symmetry constraint, has no mechanism to suppress this antisymmetric mode. Small numerical asymmetries can rise from the mesh or from round-off and nothing prevents them from growing along that direction and biasing the Picard iterates away from the symmetric mean flow the solver is actually meant to find. Imposing the symmetry line directly removes that antisymmetric mode from the discrete solution space entirely, rather than relying on it staying small.

The mesh is triangular throughout, including the boundary layer, since the FEM library used does not support mixed triangle/quadrilateral meshes.

The wall-normal resolution at the cylinder is sized directly from the reference DNS data, so that the wall distance in viscous units, $y^+ \equiv y\,u_\tau/\nu \lesssim 1$, stays below one at the wall, with $y$ the physical wall-normal distance and $u_\tau \equiv \sqrt{\tau_w/\rho}$ the friction velocity built from the wall shear stress $\tau_w$. The friction velocity is estimated at each wall angle $\theta$ from the tangential mean-velocity DNS profile,

$$u_\tau(\theta) = \sqrt{\nu \left.\frac{\partial u_t}{\partial r}\right|_{\mathrm{wall}}(\theta)}, \qquad y_1(\theta) = \frac{\nu}{u_\tau(\theta)}, \tag{3.58}$$

with $u_t$ the velocity component tangential to the wall and $y_1(\theta)$ the wall-normal spacing that gives exactly $y^+ = 1$ at that angle. A single global wall-normal first-cell height $h_{\text{wall}} = 6 \times 10^{-4} D$ is then chosen, so that $y^+ = h_{\text{wall}}/y_1(\theta) \leq 1$ over 93.5% of the cylinder wall. Angle-varying wall spacing was deliberately not used, to keep the boundary-layer mesh a simple structured annulus. The boundary layer grows geometrically away from the wall with ratio 1.15 over a radial extent of $0.796D$, reaching a spacing of about $0.06D$ at its outer edge. Tthe wall-tangential spacing is left much coarse, about $0.02D$, since tangential resolution is not the bottleneck for wall-normal wall-shear accuracy. This anisotropy keeps the total element count, and hence the cost of the direct LU factorization repeated at every Picard iteration, affordable. Isotropic mesh at the same wall-normal spacing was estimated to cost roughly 6× more per solve. Away from the boundary layer, the resulting mesh has a maximum triangle aspect ratio of 1.64; inside the anisotropic boundary-layer annulus, by design, up to 33.6.

Away from the cylinder, the background element size is set by a continuous field built directly from the same reference DNS data, rather than by a small number of discrete refinement zones. A monotonic streamwise envelope and a lateral half-width profile are defined through a small set of anchor points. They were chosen informally, based on a visual inspection of the DNS wake extent, and then refined by trial and error against the resulting mesh quality and cost. This keeps the mesh fine across roughly the wake's real physical extent, rather than a single hand-picked box, but it remains a pragmatic, hand-tuned choice, not a data-driven derivation. Every refinement transition in this field, including a shorter isotropic taper immediately around the boundary-layer annulus, is built to respect the same growth-rate cap already used for the boundary layer itself (ratio 1.15, i.e. at most a 15% size increase per element ring). Figure 3.14 shows the resulting mesh.

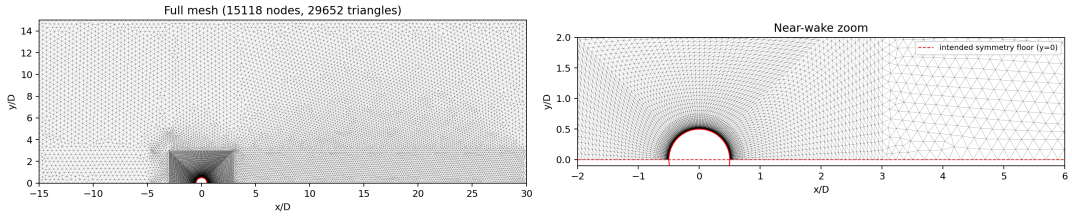



**Figure 3.14:** RANS solver mesh: anisotropic boundary layer at the cylinder wall, continuous wake refinement, and coarsening toward the far field.

### 3.3.6 Boundary conditions

Boundary conditions mirror those of the reference DNS (Section 3.1) where the two solvers overlap: uniform inflow $\overline{\mathbf{u}} = (1, 0)$ at the inlet, no-slip $\overline{\mathbf{u}} = \mathbf{0}$ on the cylinder wall, and free-slip, $\overline{v} = 0$, on the symmetry line and on the upper far-field boundary. The outlet carries no explicit boundary condition. This is the standard *do-nothing* outflow condition arising directly from the weak formulation [56, 57]. Integrating the viscous and pressure terms of the momentum equation by parts gives

$$\int_\Omega \left[ -\nu \nabla^2 \overline{\mathbf{u}} \right] \cdot \mathbf{w} \, d\Omega = \int_\Omega \nu \, \nabla \overline{\mathbf{u}} : \nabla \mathbf{w} \, d\Omega - \int_{\partial\Omega} \nu \, (\nabla \overline{\mathbf{u}} \, \mathbf{n}) \cdot \mathbf{w} \, dS, \tag{3.59}$$

$$\int_\Omega \nabla \overline{p} \cdot \mathbf{w} \, d\Omega = - \int_\Omega \overline{p} \, (\nabla \cdot \mathbf{w}) \, d\Omega + \int_{\partial\Omega} \overline{p} \, (\mathbf{w} \cdot \mathbf{n}) \, dS. \tag{3.60}$$

No boundary integral is added back on $\Gamma_{\text{out}}$, where $\mathbf{w}$ is otherwise unconstrained. The inlet, the wall, and the symmetry line fix $\mathbf{w}$ through Dirichlet conditions instead, so the corresponding boundary term already vanishes there. The discrete variational equation can only hold for every admissible $\mathbf{w}$ if the missing boundary terms at the outlet cancel out on their own:

$$-\overline{p} \, \mathbf{n} + \nu \, (\nabla \overline{\mathbf{u}}) \, \mathbf{n} = \mathbf{0} \qquad \text{on } \Gamma_{\text{out}}. \tag{3.61}$$

This natural condition ties the outlet pressure to the local viscous stress along the whole boundary segment. A fully Neumann velocity boundary would leave the pressure undetermined up to an additive constant. The inlet, the wall, and the symmetry line all carry Dirichlet conditions instead, so this degeneracy does not arise here: the natural condition alone fixes the pressure gauge, without pinning its value at an isolated point. This FEM solver does not include a sponge or damping term near the outlet; the damping function mentioned in Section 3.1 is specific to the NekRS DNS/LES runs that generated the training data, not to the solver described here.

### 3.3.7 Near-wall stress damping

Even where the mesh satisfies $y^+ \leq 1$ (Section 3.3.5), nothing guarantees that a data-driven closure's raw prediction respects the asymptotic near-wall behaviour of the true Reynolds stress. A Taylor expansion of the fluctuating velocity about a no-slip wall, combined with continuity, gives $u' \sim y$ and $v' \sim y^2$ as the wall distance $y \to 0$ [1], so that

$$\overline{u'u'} \sim y^2, \qquad \overline{v'v'} \sim y^4, \qquad \overline{u'v'} \sim y^3, \qquad y \to 0. \tag{3.62}$$

A closure prediction that violates this scaling in the near-wall band corrupts the wall shear stress, and with it the whole downstream solution, independently of how

fine the mesh is there. An optional damping is applied to enforce it: with $y$ the analytic distance to the cylinder surface and a threshold $y_{\mathrm{thr}}$,

$$f(y) = \mathrm{clip}\left(\frac{y}{y_{\mathrm{thr}}},\, 0,\, 1\right), \qquad \overline{u'u'} \mapsto f^2\,\overline{u'u'}, \quad \overline{v'v'} \mapsto f^4\,\overline{v'v'}, \quad \overline{u'v'} \mapsto f^3\,\overline{u'v'}, \tag{3.63}$$

recovering the raw prediction unchanged once $y \geq y_{\mathrm{thr}}$. The recommended threshold, $y_{\mathrm{thr}} \approx 0.046D$, is taken from the same DNS wall-shear extraction as Equation (3.58), as the 95th percentile (rather than the maximum, which is singular at the mean separation point where $u_\tau \to 0$) of $y_1(\theta)$ at $y^+ = 5$. This last value is the limit up to which this approximation still holds [1]. This damping is applied to the closure's own prediction *after* the output filter of Section 3.3.8. The order matters and is not interchangeable, because the filter length near the wall is not small compared with the width of the damping band, so filtering last would smear undamped values from outside the band back into it and undo most of the damping. It is active by default.

### 3.3.8 Feature and stress filtering

A diffusive (Helmholtz-type) filter is used throughout to smooth fields evaluated on the mesh, both the mean-flow features fed into the closure and, optionally, the Reynolds stress it predicts. Given a raw field $g_{\mathrm{raw}}$, possibly discontinuous across mesh cells, the filtered field $g$ solves

$$g - \ell^2 \Delta g = g_{\mathrm{raw}} \quad \text{in } \Omega, \qquad \nabla g \cdot \mathbf{n} = 0 \quad \text{on } \partial\Omega, \tag{3.64}$$

a homogeneous-Neumann diffusion problem with length scale $\ell$. It damps variation on scales below $\ell$ while leaving smoother, larger-scale structure essentially unchanged. Testing equation (3.64) against $w$ and integrating the Laplacian by parts (the Neumann condition removes the resulting boundary term) gives the discrete weak form actually solved, on Continuous Galerkin piecewise-linear elements ($\mathrm{CG}_1$),

$$\int_\Omega \left[ g\, w + \ell^2\, \nabla g \cdot \nabla w \right] d\Omega = \int_\Omega g_{\mathrm{raw}}\, w\, d\Omega, \qquad \forall\, w \in \mathrm{CG}_1(\Omega). \tag{3.65}$$

The associated bilinear form is coercive in the $H^1$ norm directly from the mass term alone, $a(g,g) = \|g\|_{L^2}^2 + \ell^2 \|\nabla g\|_{L^2}^2 \geq \min(1, \ell^2)\, \|g\|_{H^1}^2$. Both $a(\cdot,\cdot)$ and the right-hand side are bounded on $H^1(\Omega)$ for any $g_{\mathrm{raw}} \in L^2(\Omega)$, including a fully discontinuous, cell-wise constant field. By the Lax–Milgram theorem, Equation (3.65) therefore has a unique solution $g \in H^1(\Omega)$ for any such input, and its $\mathrm{CG}_1$ discretization is itself continuous, so $g_h \in \mathrm{CG}_1(\Omega) \subset C^0(\Omega) \subset H^1(\Omega)$ automatically. The length scale is set either mesh-relatively, $\ell = c\,h$ with $h$ the local cell diameter, or as a fixed physical length; the default is mesh-relative.

The filter can be applied in two distinct places, sharing the same operator but not necessarily the same length scale $\ell$. The *input* filter smooths each component of $\nabla\overline{\mathbf{u}}$ and $\nabla\overline{p}$ before they are evaluated by the closure, with $c = 2.0$ by default. Its output is projected back to the closure's native cell-centered representation (Section 3.3.8 below) before being packed into the closure's feature vector. The *output* filter instead smooths each component of the raw predicted $\boldsymbol{\tau}$ *after* the closure has been evaluated, and is off by default, enabled explicitly per experiment.

The closure is always evaluated at cell centroids, i.e. on cell-wise constant ($\mathrm{DG}_0$) fields, both for its inputs and its raw output $\boldsymbol{\tau}$. Differentiating the continuous ($P_2$) velocity field gives a gradient that is piecewise linear on each cell but discontinuous across cells: a broken $P_1$ field, in $L^2(\Omega)$ but not $H^1(\Omega)$. Projecting this onto $\mathrm{DG}_0$ is a purely local, per-cell operation: since the $\mathrm{DG}_0$ test and trial functions are each supported on a single cell, with no overlap between cells, the projection's weak form decouples into one independent problem per cell $K$,

$$u|_K = \frac{1}{|K|}\int_K f\, dK, \tag{3.66}$$

the integral average of the field $f$ over $K$. No cross-cell coupling is involved. The closure's inputs and output are kept at this $\mathrm{DG}_0$ level throughout. Reconstructing a globally continuous approximation from data that is not actually smooth enough is known to introduce spurious oscillations into iterative algorithms [58], and was found to destabilize the Picard loop in this solver family. Working directly with cell averages also has a practical advantage: the closure was trained on cell-averaged quantities, so that the same network can be deployed unchanged in finite-volume solvers, which likewise operate natively on cell averages.

When the output filter of Equation (3.65) is applied to $\boldsymbol{\tau}$, its solution is genuinely $H^1$, both in theory (by Lax–Milgram theorem) and in practice, because the filter equation is solved with continuous piecewise-linear ($\mathrm{CG}_1$) elements rather than cell-based ($\mathrm{DG}_0$) ones. The implicit/explicit split and the momentum weak form are then applied directly to this $\mathrm{CG}_1$ stress, without projecting it back down to $\mathrm{DG}_0$ first, which would discard exactly the regularity the filter was solved for. No modification to the weak form of Equations (3.35)–(3.36) is needed for this: UFL integrates a $\mathrm{CG}_1$ coefficient in exactly the same way as a $\mathrm{DG}_0$ one, so only the function space of $\boldsymbol{\tau}_{\mathrm{res}}$ and $\nu_{t,\mathrm{eff}}$ changes, not the form itself. Keeping the closure output in $H^1$ rather than only $L^2$ matters for three reasons:

- The $\nu_{t,\mathrm{eff}}$ coefficient enters the momentum bilinear form as a variable coefficient. Cell-wise discontinuous coefficient is the classical case of an elliptic operator with a rough coefficient, which worsens the conditioning of the assembled matrix [53, 59].
- The Picard map itself depends on $\mathbf{u}_n \mapsto \nu_{t,\mathrm{eff}}(\mathbf{u}_n)$ at every iteration. Smoother dependence improves the effective contraction of that fixed-point map [60].

- If $\boldsymbol{\tau}_{\text{res}}$ is only $\text{DG}_0$, its distributional divergence is not a function but a measure concentrated on cell interfaces, so the right-hand side of Equation (3.35) is only as regular as an $H^{-1}$ datum. Even the improved $L^2$-norm error estimate obtained via the Aubin–Nitsche duality argument does not recover the optimal rate in this case: for elliptic problems with measure-type right-hand side data, the convergence rate is provably below the smooth-data optimum, even in $L^2$ [61]. Once $\boldsymbol{\tau}_{\text{res}} \in H^1$, its divergence is a genuine $L^2$ function, elliptic regularity gains two orders, the velocity field reaches $H^2$ regularity, and the $P_2$ discretization is no longer limited by insufficiently smooth forcing data. This means less error at the same mesh size.

However, this is not sufficient on its own: the strain rate $\mathbf{S}$ entering Equation (3.42)–(3.43) must also be filtered with the output filter's own length scale in this *output filtering* setting. Since Equation (3.43) defines $\boldsymbol{\tau}_{\text{res}}$ as an algebraic identity in terms of both $\boldsymbol{\tau}$ and $\mathbf{S}$, smoothness of $\boldsymbol{\tau}$ alone does not distribute to an arbitrary decomposition of it: a noisy $\mathbf{S}$ still makes $\nu_{t,\text{eff}}$ noisy through Equation (3.42), and $\boldsymbol{\tau}_{\text{res}}$ inherits that noise back in order to keep the sum in Equation (3.43) exact. Both the output-filtered stress and the correspondingly filtered strain rate are therefore required together to keep the implicit coefficient $\nu_{t,\text{eff}}$ itself smooth.

All this filtering idea originates with Zhang et al. [3]. They apply a Helmholtz filter only to the closure's input and report a clear gain in Picard convergence. Here the same filtering is applied to the closure's output as well, the residual stress $\boldsymbol{\tau}_{\text{res}}$ itself. Output filtering yields the largest gains among all combinations tested, in both convergence speed and accuracy for several architectures.

## 3.4 Results Solver Coupling

The two local closures of Section 3.2 are embedded in the solver of Section 3.3.4. Everything reported so far is an *a priori* accuracy: the network is handed the true mean field and its output is compared against the true stress. This section reports the *a posteriori* accuracy instead. The closure is now the only source of the Reynolds stress, the mean field is the unknown, and the two are found together as the fixed point of the Picard iteration of Section 3.3.2. The two accuracies are separate properties, and the first does not imply the second.

### 3.4.1 Setting of the comparison

All the results below are obtained at $\text{Re} = 11\,000$, on the half-domain mesh of Section 3.3.5. The reference is the DNS field of Table 3.1 interpolated onto that

same mesh, so that solver output and reference share node coordinates and cell connectivity exactly and no interpolation error enters the comparison.

$\mathrm{Re} = 11\,000$ belongs to the training set of every closure considered in this thesis. The *a posteriori* error reported here is therefore not a generalization test. It measures how much accuracy is lost between handing a network the true mean field and asking a solver to reconstruct that mean field from the network alone. The in-sample *a priori* accuracy of Table 3.2 is the natural baseline against which that loss is read.

The fields produced by the solver are compared against the reference with the metrics already defined in Section 3.1.4, the relative $L^2$ error and the coefficient of determination, applied to whichever field is under discussion and always area weighted. Two further quantities appear here and nowhere else, because they measure convergence rather than accuracy: the relative velocity update $\delta u$ of Equation (3.40) and the closure increment $\delta R$ of Equation (3.79), better discussed in Section 3.4.5, that tells whether the closure map is contracting with the solver or not.

Convergence is measured by the relative velocity update $\delta u$ of Equation (3.40). The solver tolerance is $10^{-5}$ and the iteration budget is 150. Under the stabilization adopted for the earlier part of this chapter no machine-learning closure reaches that tolerance, so a coarser operational threshold is used wherever that is the case: a configuration is called *converged* when its final $\delta u$ is below $1 \times 10^{-3}$. Below that value the velocity iterate moves by less than one part in a thousand per iteration, and the field can be read as the fixed point of the coupled problem. Above it, the reported field is one particular iterate of a sequence that is still moving, and its error is not a property of the closure alone. Both cases are reported, and the distinction is always stated.

What does need stating here is the region the errors are integrated over, because it changes them by a factor of five and it is the single choice that most affects how the tables below read. Every one of these quantities depends strongly on the region it is integrated over, and the choice of region is not a detail. The computational domain reaches $30D$ downstream and $15D$ across, while the flow the closures are meant to model develops within a few diameters of the cylinder. Outside that region the solution is nearly uniform, every closure reproduces it, and the error density there is very small; but the area is enormous, so an area-weighted error over the whole domain is dominated by the region where nothing happens. Two closures that differ by a factor of three in the wake can then differ by a few per cent on the whole domain, and the metric reports agreement where there is none.

The errors reported in this work are therefore computed on the near-wake window

$$\Omega_{\mathrm{w}} = \{\, (x, y) : -1 \le x/D \le 5,\ 0 \le y/D \le 2 \,\}, \tag{3.67}$$

which contains the recirculation region and the near wake and covers 1.7% of the

mesh area. This is the primary metric throughout, and whole-domain values are given only where the comparison with them is itself the point.

Four filtering configurations of Section 3.3.8 are run for every closure: no filter, input filter only, output filter only, and both. Every other solver setting is held fixed across all the runs compared here, namely the mesh, the Reynolds number, the iteration budget, the two under-relaxation factors $\alpha = 0.4$ and $\alpha_{\text{closure}} = 0.3$, the artificial-viscosity ramp of Equation (3.41) with $\nu_{\text{floor}} = 10^{-6}$, and the near-wall damping band of Section 3.3.7. Each comparison below therefore varies one ingredient at a time, and the exceptions are stated where they occur.

### 3.4.2 Three stabilization operators

The single largest effect measured in this study is not attributable to any closure. It is attributable to the choice of stabilization operator. Three forms are compared, all sharing the same parameter $\tau_{\text{S}}$ of Equation (3.48) and differing only in which quadratic form is added to the momentum bilinear form.

The first is the consistent streamline-upwind form of Section 3.3.4, whose bilinear part is

$$s_{\text{SUPG}}(\overline{\mathbf{u}}, \overline{p}; \mathbf{w}) = \int_{\Omega} \tau_{\text{S}}\, \mathcal{A}(\overline{\mathbf{u}}, \overline{p}) \cdot (\mathbf{u}_n \cdot \nabla)\mathbf{w}\, d\Omega, \tag{3.68}$$

with $\mathcal{A}$ given by Equation (3.47).

The second is textbook streamline diffusion, which keeps the same streamline test operator but replaces the full residual by the convective term alone,

$$s_{\text{SD}}(\overline{\mathbf{u}}; \mathbf{w}) = \int_{\Omega} \tau_{\text{S}} \left[(\mathbf{u}_n \cdot \nabla)\overline{\mathbf{u}}\right] \cdot \left[(\mathbf{u}_n \cdot \nabla)\mathbf{w}\right] d\Omega. \tag{3.69}$$

The third acts on the streamwise projection of the velocity rather than on its streamline derivative. Writing $\varphi = \mathbf{u}_n \cdot \overline{\mathbf{u}}$ and $\psi = \mathbf{u}_n \cdot \mathbf{w}$ for the projections of the trial and test velocities onto the frozen iterate, the form is

$$s_{\text{ID}}(\overline{\mathbf{u}}; \mathbf{w}) = \int_{\Omega} \tau_{\text{S}}\, \nabla\varphi \cdot \nabla\psi\, d\Omega. \tag{3.70}$$

Equation (3.70) is a Laplacian acting on a single scalar field, so it diffuses $\varphi$ equally in every direction. It is referred to below as *isotropic diffusion*, in contrast with the two streamline forms, which act only along $\mathbf{u}_n$.

Read at the fixed point, Equation (3.70) is not a generic diffusion. When the iteration stops moving, $\mathbf{u}_n = \overline{\mathbf{u}}$ and the projected field becomes $\varphi = \|\overline{\mathbf{u}}\|^2$, twice the kinetic energy of the mean flow. What the term penalises is therefore the squared gradient of the kinetic energy, and by the Lamb decomposition of the convective acceleration,

$$(\overline{\mathbf{u}} \cdot \nabla)\overline{\mathbf{u}} = \tfrac{1}{2}\nabla\|\overline{\mathbf{u}}\|^2 - \overline{\mathbf{u}} \times \boldsymbol{\omega}, \tag{3.71}$$

that gradient is exactly the irrotational half of the term the stabilization exists to control. The rotational half, which carries the vorticity of the wake, does not enter.

The same reading fixes what the form cannot do. Substituting $\mathbf{w}$ for $\overline{\mathbf{u}}$ gives $\int_\Omega \tau_\mathrm{S}\|\nabla(\mathbf{u}_n \cdot \mathbf{w})\|^2 d\Omega$, which vanishes whenever $\mathbf{u}_n \cdot \mathbf{w}$ is constant. Its kernel is therefore $\{\mathbf{w} : \mathbf{u}_n \cdot \mathbf{w} = \text{const}\}$, and a velocity perturbation orthogonal to the frozen iterate costs nothing at all. The operator is isotropic in space and of rank one in the velocity components, which is a different statement from the streamline forms of Equations (3.68) and (3.69): those select a direction in space and act on both components, this one selects a component and acts in every direction.

The mechanism that follows is a matter of scales. A cell-scale oscillation of the velocity produces a cell-scale oscillation of $\varphi$, whose gradient scales as $h^{-1}$, so the penalty it pays grows as $h^{-2}$, while a field that is smooth on the scale of the wake is barely charged. This is what the explicit forcing of a cell-wise constant stress produces, and it is the reason a term with no consistency to recommend it damps that forcing more effectively than either streamline form. The argument explains the behaviour reported in Section 3.4.3; it does not make the form consistent, and the price is the one quantified there.

The three differ on two properties that turn out to matter in opposite ways.

**Coercivity.** Setting $\mathbf{w} = \overline{\mathbf{u}}$ in Equations (3.69) and (3.70) gives

$$s_\mathrm{SD}(\overline{\mathbf{u}}; \overline{\mathbf{u}}) = \int_\Omega \tau_\mathrm{S} \left\|(\mathbf{u}_n \cdot \nabla)\overline{\mathbf{u}}\right\|^2 d\Omega \;\geq\; 0, \tag{3.72}$$

$$s_\mathrm{ID}(\overline{\mathbf{u}}; \overline{\mathbf{u}}) = \int_\Omega \tau_\mathrm{S} \left\|\nabla\varphi\right\|^2 d\Omega \;\geq\; 0, \tag{3.73}$$

since $\tau_\mathrm{S} > 0$ pointwise. Both forms are positive semidefinite for any velocity field, so both add unconditionally to the coercivity of the block $A(\mathbf{u}_n)$ of Equation (3.57). The same substitution in Equation (3.68) does not produce a sign. Of the four terms collected in $\mathcal{A}$, only the convective one gives a positive quadratic form; the pressure gradient, the viscous term and the eddy-viscosity term are indefinite when tested against the streamline derivative. The consistent form therefore adds no guaranteed coercivity to $A(\mathbf{u}_n)$. The standard way to recover it is a constraint on the coefficients of the formulation, that the reaction coefficient minus half the divergence of the advecting field be bounded below by a positive constant. Neither half of that quantity is available here: the problem is steady, so the operator of Equation (3.47) carries no zeroth-order term, and $\mathbf{u}_n$ comes from a Taylor–Hood solve in which the divergence is enforced only weakly, so it is neither pointwise zero nor of one sign. Whether the consistent form stabilizes the iteration at all therefore depends on the field it is evaluated on.

**Consistency.** The converse holds for the other property. Equation (3.68) is built from the residual of the linear problem actually solved, so it vanishes identically

at the exact discrete solution and does not move the converged answer, whatever the value of $c_{\mathrm{S}}$. Equations (3.69) and (3.70) are not residual based. They add to the discrete problem a term of order $\tau_{\mathrm{S}}$ that does not vanish at its solution, so the field they converge to is not the solution of the unstabilized problem. Reducing $c_{\mathrm{S}}$ reduces that perturbation, which is the mechanism behind the dose–response of Section 3.4.3.

Left as they stand, both forms would act as a pure numerical diffusion with no counterpart on the right-hand side. Each is therefore paired with a lagged term built from the same operator applied to the previous iterate, so that for the isotropic form the pair added to the system is

$$\int_{\Omega} \tau_{\mathrm{S}}\, \nabla\varphi \cdot \nabla\psi \, d\Omega \;-\; \int_{\Omega} \tau_{\mathrm{S}}\, \nabla\varphi_n \cdot \nabla\psi \, d\Omega, \qquad \varphi_n = \mathbf{u}_n \cdot \mathbf{u}_n, \tag{3.74}$$

and analogously for Equation (3.69). The difference of the two integrals vanishes when the iteration stops moving, so what the stabilization contributes at the fixed point is a discretization of a differential operator applied to the converged field rather than an arbitrary body force. It does not make the form consistent, since that limit is still not zero, but it removes the part of the perturbation that depends on the iteration history.

### 3.4.3 The first configuration that works, and its limits

The configuration reported in this section is the first one under which the coupled problem is solvable at all. It is not the first that was tried: it is the outcome of the stabilization study above, and every ingredient in it was selected because the alternatives failed. What this section establishes is that it works, and where it stops working. Its convergence and its accuracy are reported first, then the two experiments that characterise it, namely how much stabilization it needs and what the two filters contribute. Section 3.4.4 then changes one ingredient of it and recovers most of what is missing here.

Table 3.13 reports the final $\delta u$ for the two consistent-versus-isotropic alternatives at the standard calibration $c_{\mathrm{S}} = 1$. The classical closure is included as a reference, and the choice of a one-equation model over a two-equation one is deliberate.

The standard $k$-$\varepsilon$ model of Section 2.3.1 is known to perform poorly for the class of flow considered here. It is calibrated on thin shear layers in local equilibrium, and a bluff body produces the opposite: a stagnation region where the eddy viscosity is systematically overpredicted, a separation point whose location it is unable to place, and a massively separated wake in which production and dissipation are nowhere in balance. This is precisely what motivated the long list of more elaborate variants, from the low-Reynolds-number damping functions to the realizable and Renormalization Group (RNG) formulations to the $k$-$\omega$ blends. Implementing one

of those was outside the scope of this work, and implementing the standard form would have provided a reference known in advance to be unrepresentative. The one-equation model below was chosen instead for three reasons: it is simple, it is the classical baseline adopted by the reference formulation on which this solver is built, and it was already known to work on this implementation.

A single transport equation is solved for the turbulent kinetic energy,

$$(\overline{\mathbf{u}} \cdot \nabla)k = \nabla \cdot \Big[(\nu + \nu_t)\nabla k\Big] + 2\nu_t\, \mathbf{S}\!:\!\mathbf{S} - C_D\, \frac{\sqrt{k}}{\ell}\, k, \tag{3.75}$$

whose three terms are diffusion, production and destruction. The eddy viscosity follows from the Prandtl–Kolmogorov relation, with a von Kármán mixing length capped in the far field,

$$\nu_t = C_\mu^{1/4} \sqrt{k + k_\epsilon}\; \ell, \qquad \ell = \min\Big(\kappa\, d,\ \ell_{\max}\Big), \tag{3.76}$$

with $d$ the wall distance, $\kappa = 0.41$, $C_\mu = 0.09$, $C_D = C_\mu^{3/4}$, $\ell_{\max} = 1.0\, D$ and $k_\epsilon = 10^{-6}$ a floor inside the square root that keeps $\nu_t$ Lipschitz where $k$ vanishes. The diffusivity of $k$ carries no turbulent Prandtl number, the destruction term is linearized around the previous iterate, and $k = 0$ is imposed both at the wall and at the inlet. The stress follows from Equation (2.12), so the eddy viscosity is a smooth field obtained from a differential problem rather than a pointwise network evaluation.

The difference from the two-equation form is where the length scale comes from. A $k$-$\varepsilon$ model transports a second quantity and recovers the length scale from it implicitly, through $\nu_t = C_\mu k^2/\varepsilon$. Here it is prescribed: $\ell$ is the smaller of a wall-proportional length $\kappa d$ and a fixed cap $\ell_{\max}$, so the model is wall-driven within $d < \ell_{\max}/\kappa \approx 2.4\, D$ of the cylinder surface and uses a single constant length equal to one diameter everywhere beyond it.

What is bought with this simpler model is robustness. The $\varepsilon$ equation is the fragile half of a two-equation model, with a stiff production–destruction balance and a delicate near-wall limit, and a two-equation low-Reynolds-number implementation of it was in fact tried first on this configuration and abandoned: it produced a slowly building and suddenly releasing oscillation in the wake, which survived a cap on the eddy viscosity, a raised floor on $k$ and a seeded initial condition.

None of the constants above was tuned against the present data. This reference is therefore an uncalibrated baseline, and every margin reported against it in this chapter should be read as an upper bound on the advantage of the data-driven closures over a classical model that had been fitted to this flow.

Three things follow directly from Table 3.13.

- No machine-learning closure converges under the consistent form. The best value any of them reaches in the lower half of the table is $5.2 \times 10^{-3}$, and

**Table 3.13:** Final relative velocity update $\delta u$, Equation (3.40), at $\mathrm{Re} = 11\,000$ with $c_\mathrm{S} = 1$. Bold marks a configuration with $\delta u < 10^{-3}$, italics one with $10^{-3} \leq \delta u < 10^{-2}$, and a value in upright type one with $\delta u \geq 10^{-2}$, whose field is an arbitrary iterate of a sequence that is still moving. An asterisk marks the runs that reached the solver tolerance $10^{-5}$ before the iteration budget was exhausted. The superscript "s" marks a run stopped early by a singular momentum matrix. All runs use the scalar split of Equation (3.43).

| Stabilization | Closure | No filter | Input | Output | Input+output |
|---|---|---|---|---|---|
| Isotropic diffusion | $k$-$\ell$ | **1.2e-5**$^*$ | **2.0e-5**$^*$ | **1.2e-5**$^*$ | **2.0e-5**$^*$ |
| | MLP | 1.9e-2 | *9.9e-3* | *1.3e-3* | *1.3e-3* |
| | MLP+PINN | 1.0$^\mathrm{s}$ | 1.7$^\mathrm{s}$ | 1.0$^\mathrm{s}$ | 9.9e-1$^\mathrm{s}$ |
| | TBNN | 2.0e-1 | *7.7e-3* | *3.2e-3* | **8.7e-4** |
| | TBNN+PINN | *6.6e-3* | *4.4e-3* | **3.5e-4** | **3.4e-4** |
| Consistent SUPG | $k$-$\ell$ | **1.4e-5**$^*$ | **1.3e-5**$^*$ | **1.4e-5**$^*$ | **1.3e-5**$^*$ |
| | MLP | 2.4 | 1.9 | 2.5 | 7.6e-1 |
| | MLP+PINN | 9.7e-1 | 1.2e2 | 3.4 | 1.7 |
| | TBNN | 2.1e-1 | 4.9e-2 | 6.0e-2 | 2.9e-2 |
| | TBNN+PINN | *5.2e-3* | 1.5e-2 | 2.1e-2 | *8.3e-3* |
| Streamline diffusion | TBNN | 2.3e-1 | 2.7e-2 | 2.1e-2 | 1.8e-2 |

the plain MLP does not merely fail to converge but leaves the physical range entirely, with $\delta u$ above unity in three configurations out of four.

- Replacing the consistent form by the isotropic one improves the final $\delta u$ of every machine-learning closure, by between one and three orders of magnitude, and the improvement is largest exactly where the consistent form fails hardest: a factor of about 580 for the MLP, of 33 for the tensor-basis closure and of 15 for its physics-informed variant. The classical reference is not affected by the choice, agreeing to two significant figures between the two halves of the table and reaching the solver tolerance in all eight runs.

- Convergence is therefore not a property of the closure alone, nor of the stabilization alone, but of the pair. The same tensor-basis checkpoint converges or fails to converge depending only on which quadratic form is added to the momentum operator, with every other setting identical.

The mechanism is the one anticipated in Section 3.4.2. A network closure is evaluated pointwise on the current iterate and re-enters the momentum equation through its divergence, so the operator whose fixed point is sought has a data-driven, non-smooth coefficient. What the iteration needs from the stabilization

is coercivity of the linear operator at every step, which Equation (3.70) provides unconditionally and Equation (3.68) does not provide at all. Consistency, the property the streamline-upwind form is designed around, is only meaningful once the iteration reaches a fixed point, and is therefore of no use in the regime where these closures actually operate. The last row of Table 3.13 separates the two ingredients on the tensor-basis closure: dropping consistency while keeping the streamline direction is worth a factor of less than two, from $2.9 \times 10^{-2}$ to $1.8 \times 10^{-2}$, and the remaining factor of twenty down to $8.7 \times 10^{-4}$ belongs to the shape of the operator. The streamline-diffusion form is reported for that measurement alone and is not used again. Figure 3.15 shows the same runs as histories, where the separation by row rather than by column is immediate.

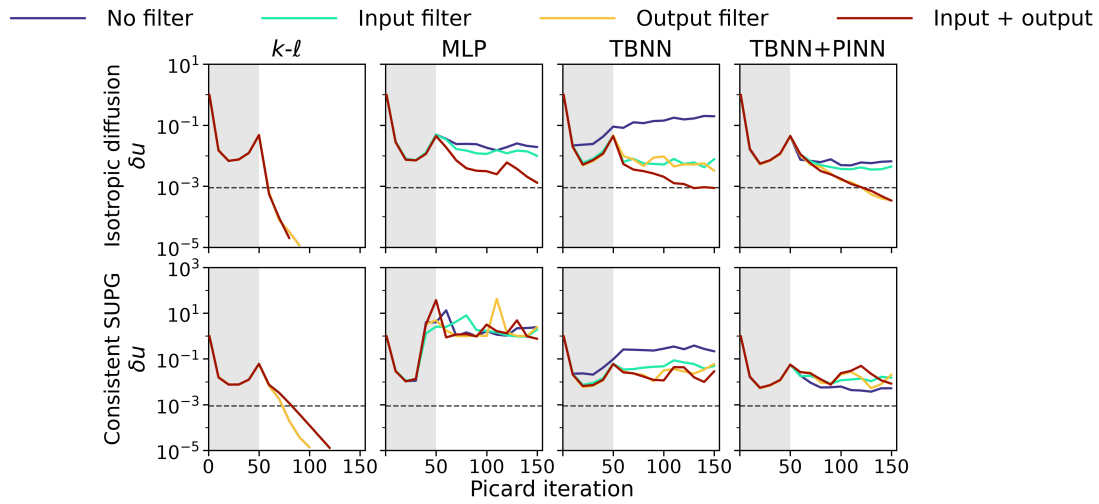



**Figure 3.15:** Convergence history of the Picard iteration at Re $= 11\,000$, with isotropic diffusion (top row) and consistent streamline-upwind stabilization (bottom row). Columns are closures, curves within each panel are the four filtering configurations of Section 3.3.8. The physics-informed MLP is omitted: it never approaches convergence in any configuration. In the reference panels the four curves coincide, the transported closure being insensitive to the filters. The vertical range differs between the two rows, since no run of the upper row exceeds 10. The dashed line marks the $9 \times 10^{-4}$ threshold and the shaded band the artificial-viscosity ramp of Equation (3.41). Curves that stop before the budget have reached the solver tolerance.

Table 3.14 reports the accuracy of the same runs. The physics-informed MLP is absent from it: its momentum matrix becomes singular between the fortieth and the seventy-ninth iteration in every configuration of the isotropic form, and under the consistent form it leaves the physical range instead, so it produces no

field to measure. That this closure fails where its purely supervised counterpart merely underperforms is itself informative, since the physics-informed variant is the supervised one plus two additional constraints and should be at least as deployable. What is measured is not the cost of adding physics but the cost of adding a residual that, on these data, is not consistent with the labels the same objective is fitting.

**Table 3.14:** *A posteriori* accuracy at Re $= 11\,000$, isotropic diffusion, $c_{\mathrm{S}} = 1$. Accuracy is reported only where the iteration reached at least a partial fixed point. Bold marks a configuration with $\delta u < 10^{-3}$, italics one with $10^{-3} \leq \delta u < 10^{-2}$, and a configuration with $\delta u \geq 10^{-2}$ is left blank, its field being an arbitrary iterate of a sequence that is still moving. The values of $\delta u$ are those of Table 3.13. Both errors are computed on the window of Equation (3.67). All runs use the scalar split of Equation (3.43).

| Closure | Metric | No filter | Input | Output | In+out |
|---|---|---|---|---|---|
| $k$-$\ell$ | $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | **0.085** | **0.083** | **0.085** | **0.083** |
| | $\varepsilon_{L^2}^{\mathrm{tensor}}$ | **0.481** | **0.584** | **0.481** | **0.584** |
| MLP | $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | | *0.092* | *0.093* | *0.093* |
| | $\varepsilon_{L^2}^{\mathrm{tensor}}$ | | *0.371* | *0.294* | *0.294* |
| TBNN | $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | | *0.083* | *0.116* | **0.091** |
| | $\varepsilon_{L^2}^{\mathrm{tensor}}$ | | *0.387* | *0.429* | **0.417** |
| TBNN+PINN | $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | *0.101* | *0.129* | **0.121** | **0.127** |
| | $\varepsilon_{L^2}^{\mathrm{tensor}}$ | *0.513* | *0.510* | **0.491** | **0.509** |

The table records a negative result before it records anything else. Of the sixteen runs, seven never come within a factor of ten of the convergence criterion and their columns are left blank: an error computed on a field that is still moving measures where the iteration happened to stop rather than what the closure predicts. Only four runs, all in bold, correspond to a field that can be called a solution, and the classical reference supplies all of its own.

On accuracy the converged runs say two things. On the velocity no network improves on the reference, which stands at 0.083 against 0.091 for the tensor-basis closure and 0.121 for its physics-informed variant. On the stress the ordering reverses, the tensor-basis closure reaching 0.417 against 0.481 for the reference in its own best configuration. The margin the *a priori* metrics of Section 3.2 would predict is therefore not visible here in either direction, and no closure beats the reference on both quantities. Figure 3.16 shows the same runs as fields, the columns being nearly indistinguishable on the velocity and clearly distinct on the stress. That the velocity fields should agree so closely while the stresses that produce them do not suggests that the isotropic diffusion of Equation (3.70) is itself large

enough to set the velocity, so that what the top row displays is in good part the effect of the stabilization rather than of the closure. The following configuration of Section 3.4.4 supports that reading: with the stabilization replaced and nothing else changed, the same closures separate on the velocity as well. This is offered as a reading and not as a measurement, and it is one more reason to be wary of ranking closures by a velocity error alone. Figure 3.17 makes the same point station by station, and adds where the classical reference actually fails: from two diameters downstream its shear stress peaks at roughly twice the reference value and in the wrong place, half a diameter too far from the axis, while both networks stay closer to it. The same figure carries the sharpest evidence for the reading above. At the last station the multilayer closure has the shear stress of the wrong sign and the wrong curvature, peaking positive where the reference has a negative minimum, and its velocity profile there is nevertheless indistinguishable from the other three. A closure can therefore be wrong about the direction of the momentum transfer in the far wake and leave no trace whatever on the velocity, which settles that the agreement of the top row is not a property of the closures.

Two runs illustrate, in opposite directions, why both the window and its complement are reported throughout. The unfiltered tensor-basis run is one of those excluded from the table, and it is instructive for that reason: on the whole domain its velocity error is 0.062, which would read as a usable field, while on the window it is 0.351, with 3.12 on the cross-stream component and a stress error of 4.42. The MLP closure fails the same test from the other side, and it is in the table: on the window its stress error is 0.294, the lowest of the local closures, and outside the window its shear-stress error is 5.09, against 0.84 for the tensor-basis closure and 0.45 for the reference. It is accurate where it is measured and wrong everywhere else.

The scale factor $c_{\mathrm{S}}$ of Equation (3.48) was then varied over 1, 0.1 and 0.05 with the isotropic form held fixed. Table 3.15 reports the effect of the scale on both criteria.

The two criteria move in opposite directions along $c_{\mathrm{S}}$, which is the expected consequence of the non-consistency established in Section 3.4.2. Reducing $c_{\mathrm{S}}$ reduces the perturbation the stabilization adds to the discrete problem, so the fixed point moves closer to the solution of the unstabilized problem; it also reduces the coercivity the same term supplies, so the iteration is less able to reach that fixed point.

The exchange rate is not the same for every closure, and it is not always worth taking. At $c_{\mathrm{S}} = 0.1$ the tensor-basis closure gains little for what it gives up, five per cent off the velocity error against a fourfold degradation of a residual already above the threshold, while its physics-informed variant gains nearly a third for a degradation of less than two. The plain MLP buys nothing: it worsens on both criteria at once, and is the one closure whose two criteria move together instead of

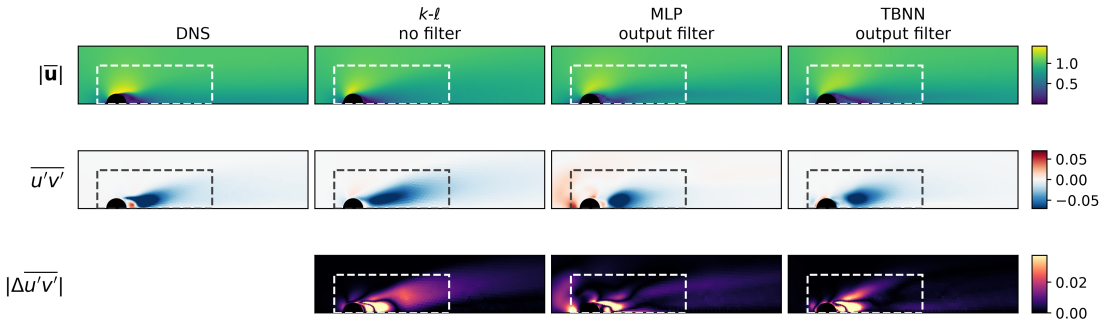



**Figure 3.16:** Converged fields at Re = 11 000 under isotropic diffusion, each closure in the configuration stated above its column. Top row, velocity magnitude; middle row, shear stress; bottom row, local error on the shear stress against the reference. Each row shares one colour scale. The dashed rectangle is the window of Equation (3.67): the error of the bottom row is concentrated inside it and vanishes outside. The columns are nearly indistinguishable on the velocity and clearly distinct on the stress.

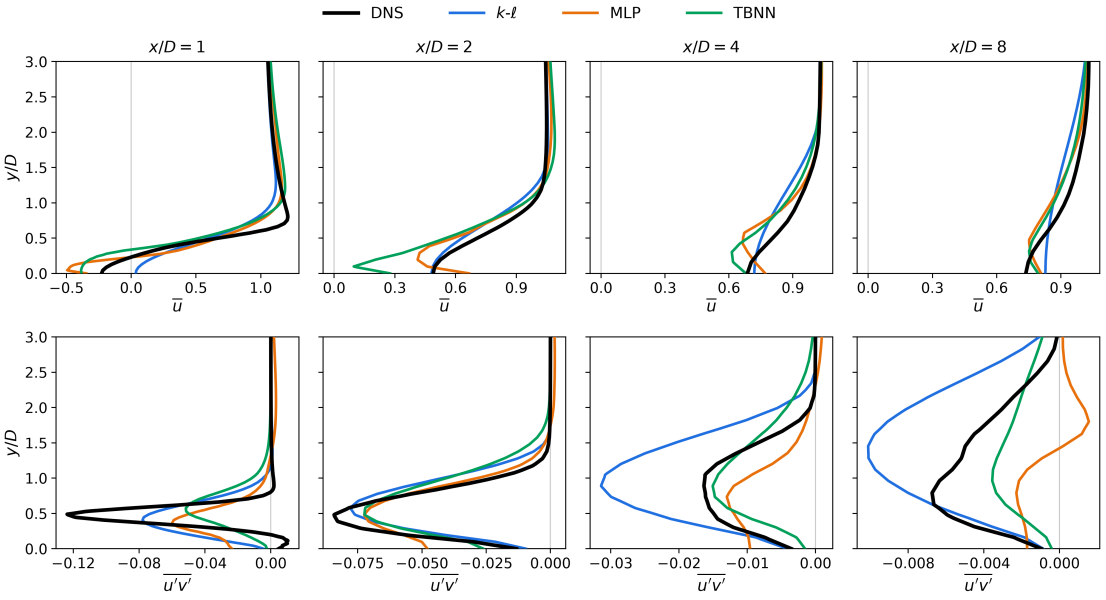



**Figure 3.17:** Wake profiles at four streamwise stations, converged runs under isotropic diffusion, each closure in the configuration of Figure 3.16. Top row, streamwise velocity; bottom row, shear stress. The horizontal scale is set independently in each panel, since the shear stress falls by an order of magnitude between the first station and the last; the panels therefore show shape and not magnitude. The velocity profiles nearly collapse; the stress profiles do not.

**Table 3.15:** Effect of the stabilization scale under isotropic diffusion. For each closure the table reports the best value over the four filtering configurations, separately for the two criteria. The errors are computed on the window of Equation (3.67). All runs use the scalar split of Equation (3.43).

| | best $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | | | best $\delta u$ | | |
|---|---|---|---|---|---|---|
| Closure | $c = 1$ | $c = 0.1$ | $c = 0.05$ | $c = 1$ | $c = 0.1$ | $c = 0.05$ |
| $k$-$\ell$ | 0.083 | 0.069 | 0.078 | 1.2e-5 | 9.9e-6 | 1.1e-5 |
| MLP | 0.092 | 0.122 | 0.104 | 1.3e-3 | 4.3e-2 | 3.6e-2 |
| TBNN | 0.083 | 0.079 | 0.075 | 8.7e-4 | 3.5e-3 | 4.4e-3 |
| TBNN+PINN | 0.101 | 0.073 | 0.069 | 3.4e-4 | 5.3e-4 | 6.6e-4 |

against each other. Whether a closure can be deployed at all is therefore settled by the numerics before any comparison between closures is meaningful.

The two filters of Section 3.3.8 act on different objects and their contributions are not symmetric.

The output filter is the one that governs convergence. Its effect on $\delta u$ is large in every row of the upper half of Table 3.13, a factor of 63 for the tensor-basis closure, of 19 for its physics-informed variant and of 15 for the MLP, and every configuration that meets the convergence criterion includes it.

On accuracy the two filters have no consistent ordering. The input filter is the better of the two for the tensor-basis closure, 0.083 against 0.116 on the velocity, and the output filter is the better one for the MLP, 0.294 against 0.371 on the stress, and marginally so for the physics-informed variant. What the input filter does unambiguously is repair a closure fed a gradient field it cannot handle: on the unfiltered tensor-basis run it takes the window velocity error from 0.351 to 0.083 and the stress error from 4.42 to 0.387. It is a repair rather than an improvement, and it does nothing comparable to a configuration that is already working.

The two effects can therefore oppose each other. For the tensor-basis closure the best convergence is reached with both filters and the best accuracy with the input filter alone, which does not converge. Choosing a configuration by $\delta u$ and choosing it by the error give different answers, and the criterion has to be declared.

This is the whole of what this configuration offers, and its limits are structural. It converges only by adding a term that does not vanish at the solution and that perturbs the discrete problem at first order in the cell size, its coefficient trades the two criteria against each other rather than improving both, and no filtering configuration is best on both. The next section replaces that term with one that has none of those properties.

### 3.4.4 The adopted configuration: the filtered split as a stabilization

Everything reported so far uses the isotropic non-consistent form of Equation (3.70). That form converges, but it is also the form this section replaces.

The objection is the one already established in Section 3.4.2. The form is not residual based, so the term it adds does not vanish at the solution of the unstabilized problem and the fixed point the iteration reaches is not that of the problem one meant to solve. Its weight is $O(h)$, since $\tau_{\mathrm{S}} \simeq h/2\|\overline{\mathbf{u}}\|$ in the convection-dominated regime that covers between 61% and 73% of the cells of the window. It is therefore a first-order perturbation of the discrete problem, and it is not a form with standing in the literature: it emerged from a sequence of trials that produced good convergence.

The consistent form of Equation (3.68) has neither defect and, on its own, does not converge. What makes it usable is a second ingredient, obtained by reading the output filter of Section 3.3.8 differently.

In the formulation used above, the filter is applied to the predicted stress and the split of Equation (3.43) is then computed from the filtered field, so that the residual and the implicit coefficient see the same strain rate. The alternative is to build the residual against the filtered strain rate while the imposed stress is still referred to the raw one. The stress the momentum equation receives is then

$$\boldsymbol{\tau}_{\mathrm{imp}} = \boldsymbol{\tau} + 2\nu_t\Big(\mathbf{S}_{\mathrm{filt}} - \mathbf{S}_{\mathrm{raw}}\Big) \;\simeq\; \boldsymbol{\tau} + 2\nu_t\,(c\,h)^2\,\nabla^2\mathbf{S}, \tag{3.77}$$

since the filter of Section 3.3.8 is a Helmholtz smoother of length $c\,h$ and its action on a field is, to leading order, the addition of $(ch)^2$ times its Laplacian. The difference between the two formulations is therefore a biharmonic term of coefficient $\nu_t(ch)^2$, and it is a second stabilization rather than a variant of the filter.

Three properties justify it.

**It is second order in the mesh size.** The added term is $O(h^2)$, one order higher than the non-consistent form it replaces, and it vanishes in the continuum limit at the same rate as the discretization error of the scheme. It is not consistent in the strict sense, since at finite $h$ it does not vanish on the exact solution, but the perturbation it introduces is of the same order as the one the discretization introduces in any case, so it does not degrade the formal order of the method.

**It cannot dominate the closure it stabilizes.** At wavelength $k$ the ratio between the added term and the closure's own dissipation is $(c\,h\,k)^2$, independently of the closure. On the mesh of Section 3.3.5 that ratio is 39.5 at the cell scale and 0.061 at the scale of the wake: forty times the closure where the noise lives, six per cent of it where the flow lives. The term is proportional to the eddy viscosity the closure itself predicts, so it should not overwhelm the closure it is added to, and

every closure receives the same dose relative to its own dissipation. The comparison between architectures is fairer for it.

**It has no free coefficient.** The length $c\,h$ is the filter scale already declared, $h$ is the mesh, and $\nu_t$ comes from the network. Nothing here is tuned, which is precisely what the dose–response of Section 3.4.3 had to establish for the non-consistent form.

The configuration adopted from here on is therefore the consistent form of Equation (3.68) together with the split of Equation (3.77). Table 3.16 reports the local closures under it.

**Table 3.16:** *A posteriori* accuracy of the local closures at $\mathrm{Re} = 11\,000$ under the adopted configuration, consistent streamline-upwind stabilization with the filtered split of Equation (3.77), 150 iterations. Relative $L^2$ errors are on the window of Equation (3.67), coefficients of determination over the whole domain.

| Closure | Metric | No filter | Input | Output | In+out |
|---|---|---|---|---|---|
| $k$-$\ell$ | $\delta u$ | 1.4e-5 | 1.3e-5 | 1.4e-5 | 1.3e-5 |
| | $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | 0.105 | 0.103 | 0.105 | 0.103 |
| | $\varepsilon_{L^2}(\overline{u})$ | 0.114 | 0.110 | 0.114 | 0.110 |
| | $\varepsilon_{L^2}(\overline{v})$ | 0.449 | 0.415 | 0.449 | 0.415 |
| | $\varepsilon_{L^2}^{\mathrm{tensor}}$ | 0.521 | 0.640 | 0.521 | 0.640 |
| | $R^2(\overline{u'v'})$ | 0.46 | 0.50 | 0.46 | 0.50 |
| MLP | $\delta u$ | *9.5e-1* | *1.8e+0* | *6.6e-1* | *8.3e-1* |
| | $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | *163* | *32.3* | *7.23* | *0.168* |
| | $\varepsilon_{L^2}(\overline{u})$ | *162* | *30.3* | *7.15* | *0.186* |
| | $\varepsilon_{L^2}(\overline{v})$ | *277* | *99.1* | *12.5* | *1.229* |
| | $\varepsilon_{L^2}^{\mathrm{tensor}}$ | *152* | *3133* | *67.7* | *0.435* |
| | $R^2(\overline{u'v'})$ | — | — | — | *0.50* |
| TBNN | $\delta u$ | *2.4e-1* | *3.0e-2* | *6.9e-3* | *1.7e-3* |
| | $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | *0.466* | *0.083* | *0.060* | *0.048* |
| | $\varepsilon_{L^2}(\overline{u})$ | *0.586* | *0.087* | *0.064* | *0.054* |
| | $\varepsilon_{L^2}(\overline{v})$ | *3.99* | *0.440* | *0.276* | *0.268* |
| | $\varepsilon_{L^2}^{\mathrm{tensor}}$ | *2.58* | *0.393* | *0.327* | *0.402* |
| | $R^2(\overline{u'v'})$ | −51.6 | −0.73 | *0.24* | *0.57* |
| TBNN+PINN | $\delta u$ | *4.0e-3* | 2.2e-5 | 1.1e-5 | 2.0e-5 |
| | $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | *0.057* | 0.085 | 0.068 | 0.070 |
| | $\varepsilon_{L^2}(\overline{u})$ | *0.056* | 0.086 | 0.068 | 0.069 |
| | $\varepsilon_{L^2}(\overline{v})$ | *0.193* | 0.334 | 0.263 | 0.235 |
| | $\varepsilon_{L^2}^{\mathrm{tensor}}$ | *0.493* | 0.530 | 0.532 | 0.544 |
| | $R^2(\overline{u'v'})$ | *0.72* | 0.17 | 0.49 | 0.44 |

Some immediate observations follow.

The tensor-basis closure beats the classical reference, and by a wide margin. Under the adopted configuration it reaches a velocity error of 0.048 on the window against 0.105 for the one-equation model, that is less than half, and 0.402 against 0.521 on the stress. This is the first configuration in this chapter in which a machine-learning closure is better than the classical one on both quantities at once: under the stabilization of Section 3.4.3 the reference was still ahead on the velocity. Figure 3.18 shows where that margin lies, the velocity fields being barely distinguishable and the closures separating on the stress components.

Part of that reversal is the reference itself doing worse here: the velocity error $\varepsilon_{L^2}(|\overline{\mathbf{u}}|)$ of the classical closure is 0.105 under the adopted configuration, against 0.083 under the isotropic form. Still a big win but a bit less impressive. However, the other configuration introduced a shift of the fixed point, unnecessary for a closure that already converges as the classical one, which for that case luckily make the solver solution get closer to the real one. In general, it is not proven that the shift is always in the right direction. Therefore, the configuration under which the classical closure is actually used is the one presented here, with a consistent classical stabilizer.

The MLP does not converge under a consistent stabilization, as already seen in the previous try. Its final update lies between 0.66 and 1.84 across the four filtering configurations, and its velocity error reaches 163 where the iteration leaves the physical range altogether. In the same conditions the tensor-basis closure produces a field with a velocity error of 0.048. The usability the MLP closure showed in Table 3.14 was therefore borrowed from the form of stabilization and not supplied by the closure: remove that support and the closure stops producing a field, while the tensor bases keep producing one, and a better one.

The cross-stream component carries most of the velocity error. Its relative error lies between 0.19 and 0.44 in the network configurations that produce a field, three to six times the error on the velocity magnitude, and the two do not move together. This can also be considered as an artifact of the relative metric since $v$ is almost 0 everywhere, making the relative error grow more even for smaller error. Indeed, visually the field is comparable.

The physics-informed variant is the exception on both counts, and it is worth stating separately. It is the only closure that reaches the tolerance under both stabilization forms, $1.2 \times 10^{-5}$ and $1.1 \times 10^{-5}$, and in three filtering configurations out of four it stops before the iteration budget, which no other closure does. That a closure constrained by the equations is the only one that does not need the non-consistent form to stand up is a result in favour of coupling physics-informed training with data for closure modelling. Its accuracy does not follow. Its stress error outside the window is 2.77 in all eight runs, four filters by two stabilizations, constant to four digits, against 0.35 to 1.01 for the other closures and 0.48 for the

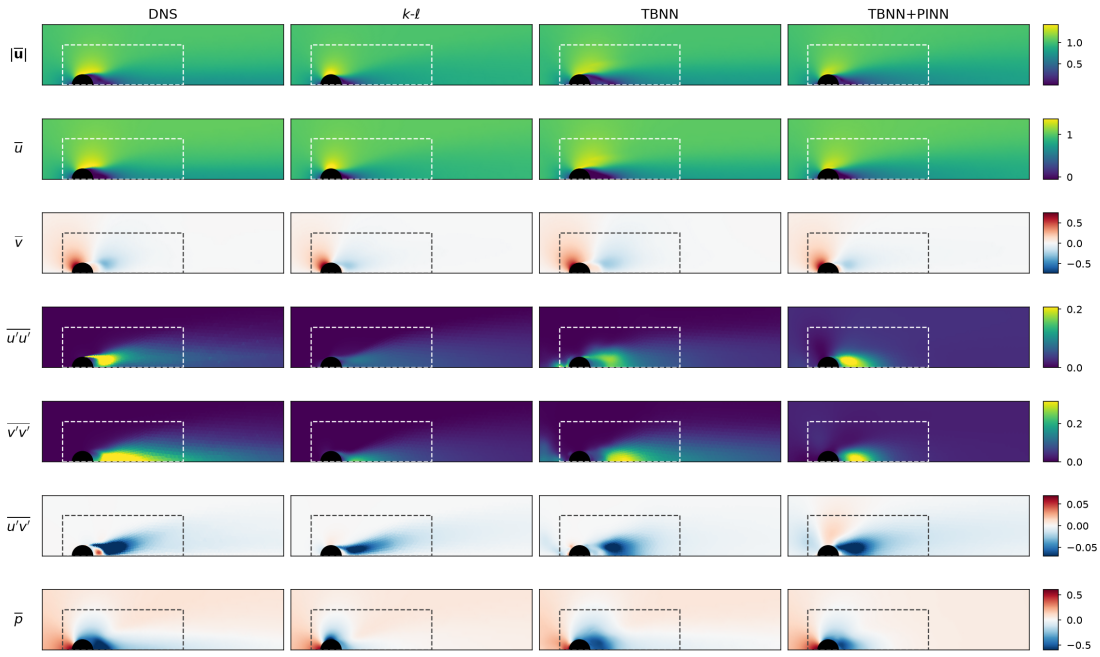



**Figure 3.18:** Converged fields under the adopted configuration, compared against the reference DNS. Each closure is shown in its best filtering configuration of Table 3.16. The multilayer closure is absent because it produces no field under this stabilization. The dashed rectangle is the window of Equation (3.67), over which the errors of the tables are computed.

reference. That the number does not move with either the filter or the stabilization places the discrepancy in the network rather than in the coupling, which is where the mismatch of Section 3.2.3 would place it. The closure is not usable as it stands, and the combination of unconditional convergence with a network-level error is the reason it is worth repairing rather than discarding.

Two ingredients buy different things and are not interchangeable. The non-consistent form buys convergence, by one to two orders of magnitude on every closure that survives it. The consistent form with the filtered split buys accuracy: the velocity error of the tensor-basis closure falls from 0.091 to 0.048 and its shear-stress $R^2$ rises from 0.52 to 0.57. Only the second of the two rests on an argument rather than on an observation, which is why it is the one adopted.

The comparison is made at a final update of order $10^{-3}$ rather than $10^{-5}$, and the residual indeterminacy this leaves is quantified rather than assumed. Table 3.17 repeats the adopted configuration at four iteration budgets, and the measurement is reported below. Every number reported at this level of convergence carries that qualification, and the configurations that do reach the tolerance are marked as

such.

**Table 3.17:** Dependence on the iteration budget, tensor-basis closure under the adopted configuration. For each filtering configuration the first two rows are the final relative velocity update of Equation (3.40) and the final closure increment of Equation (3.79), and the third is the relative $L^2$ error of the velocity magnitude on the window of Equation (3.67). Only the iterations budget changes between columns.

| Filtering | Quantity | 100 | 150 | 200 | 400 |
|---|---|---|---|---|---|
| Input | $\delta u$ | 3.4e-2 | 3.0e-2 | 4.4e-2 | 5.1e-2 |
| | $\delta R$ | 1.1e-1 | 1.0e-1 | 1.2e-1 | 1.3e-1 |
| | $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | 0.081 | 0.083 | 0.084 | 0.077 |
| Output | $\delta u$ | 2.9e-2 | 6.9e-3 | 1.5e-2 | 1.9e-2 |
| | $\delta R$ | 5.7e-2 | 1.4e-2 | 2.5e-2 | 3.7e-2 |
| | $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | 0.065 | 0.060 | 0.064 | 0.065 |
| Input + output | $\delta u$ | 2.4e-3 | 1.7e-3 | 9.9e-4 | 1.4e-3 |
| | $\delta R$ | 1.7e-2 | 1.5e-2 | 1.1e-2 | 1.9e-2 |
| | $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | 0.044 | 0.048 | 0.050 | 0.050 |

Three things follow from Table 3.17, and the first two together are what licenses the comparisons of this section.

Wherever the output filter is present, the error settles while the update does not. Between 200 and 400 iterations the velocity error changes in the fourth decimal place, 0.050 to 0.050 with both filters and 0.064 to 0.065 with the output filter alone, while the final update over the same interval moves by a factor of two and moves in both directions. The iterate is therefore not converging to a point, it is confined to a neighbourhood, and the size of that neighbourhood is what the update measures; the field itself has stopped moving in any way the error metric can resolve. This is why a final update of order $10^{-3}$ does not prevent the comparison: the residual indeterminacy of the field is an order of magnitude smaller than the quantity being compared.

The input filter on its own behaves differently, and the table is the cleanest evidence of it. Its final update grows monotonically with the budget, from $3.4\times10^{-2}$ to $5.1\times10^{-2}$, and its velocity error moves by eight per cent without settling on a value. That configuration is not approaching a fixed point at all, and the numbers reported for it anywhere in this chapter are iterates rather than solutions. It is also the configuration whose update is an order of magnitude larger than the other two at every budget, so the two readings agree.

The closure increment settles least of the three quantities, and it settles nowhere. It stays between $1.1\times10^{-2}$ and $1.3\times10^{-1}$ at every budget and in every filtering

configuration, one to two orders of magnitude above the velocity update measured on the same iterate, and it neither falls nor converges as the budget grows. The velocity field stops moving in a way the error metric can resolve long before the closure output does, which is the statement of Section 3.4.5 read along a single axis.

For the two configurations that do settle, the whole excursion is smaller than the differences the tables are used to resolve. Across a fourfold change of budget the velocity error moves by 0.006 at most, from 0.044 to 0.050 with both filters, which is twelve per cent of its own value. The differences between filtering configurations at any fixed budget are between 0.02 and 0.04, and the differences between closures reported earlier in this section are larger still. The uncertainty introduced by stopping the iteration is therefore not of a size that could reorder any of the conclusions drawn from these tables.

The natural next step, left to future exploration, is to use the term of Equation (3.77) on its own, without filtering the predicted field at all, so that the only stabilization acting on the closure is one whose magnitude is set by the closure itself.

### 3.4.5 What the coupling costs

The four experiments that follow are diagnostics rather than configurations. They were measured under the implementation of Section 3.4.3, and none of them is an attempt to improve on the one adopted above: each isolates one reason why coupling a network closure to this solver is harder than coupling a transported one, and their conclusions are properties of that pair rather than of the stabilization used to measure them.

#### Without stabilization

The stabilization study of Section 3.4.2 varies the operator that is added to the momentum form. It cannot say how much of the difficulty belongs to the discretization and how much to the closure, because no configuration in it removes the stabilization altogether. A control run supplies that missing reference. The stabilization is switched off entirely, $\tau_{\mathrm{S}} = 0$, or leaving only the grad-div term, which vanishes exactly on a divergence-free field and therefore does not move the continuous fixed point.

Table 3.18 reports the result. The classical reference reaches $1.1 \times 10^{-4}$ with no stabilization at all, with a minimum of $1.3 \times 10^{-5}$ at iteration 100 and a residual still falling at the end of the budget. Adding the grad-div term changes it by two per cent. The mesh of Section 3.3.5, the Taylor–Hood pair and the Picard iteration therefore reach the solver tolerance on this problem without any stabilization

whatsoever. Every difficulty documented in this chapter is a property of the pair closure and solver, and not of the discretization on its own.

**Table 3.18:** Median relative velocity update over the last third of the iteration budget, without stabilization and with the grad-div term alone, at Re $= 11\,000$ and 150 iterations. For the transported closure the output filter is not applied, so the first and third columns coincide, as do the second and fourth. All runs use the scalar split of Equation (3.43).

| Stabilization | Closure | No filter | Input | Output | Input+output |
|---|---|---|---|---|---|
| None | $k$-$\ell$ | **1.1e-4** | **8.3e-5** | **1.1e-4** | **8.3e-5** |
| | TBNN | 1.2 | 1.2 | 2.5e-2 | 2.9e-2 |
| | MLP | 1.3 | 1.4 | 2.9 | 1.1 |
| Grad-div only | $k$-$\ell$ | **1.1e-4** | **8.2e-5** | **1.1e-4** | **8.2e-5** |
| | TBNN | 2.6e-2 | 2.4e-2 | 2.3e-2 | 3.2e-2 |
| | MLP | 2.4 | 1.5 | 4.0 | 1.0 |

Two further readings follow from Table 3.18.

The grad-div term is not a generic stabilizer here. It leaves the classical reference untouched and improves convergence of the unfiltered tensor-basis closure by a factor of about forty-five, from 1.2 to $2.6 \times 10^{-2}$, which is the difference between a diverging run and a bounded one. It is a repair for a specific defect of an unfiltered network prediction, and not a uniform gain.

The pattern is sharper than that, and it points at the mechanism. The grad-div term acts only where the output filter is absent: it recovers the two diverging columns, with and without the input filter, and leaves the two filtered ones unchanged to within ten per cent. A plausible reading is that the two act on the same defect from opposite ends. The predicted stress is evaluated cell by cell, so the explicit forcing it produces is a cell-wise constant field whose divergence concentrates on the cell interfaces; that forcing drives a spurious cell-scale compressible component in the velocity iterate, and the grad-div term penalizes exactly that component. The output filter removes the same forcing at the source by lifting the stress to a continuous space before its divergence is taken, and once it has done so there is nothing left for the grad-div term to penalize. This is offered as a reading and not as a measurement: it would be isolated by monitoring the discrete divergence of the iterate in the two cases, which is left for further exploration.

The value at which the velocity update $\delta u$ of the machine-learning closures settles, of order $10^{-2}$, is set neither by the stabilization nor by the architecture. What the closures have in common, and the transported reference does not, is that their eddy viscosity is a pointwise evaluation rather than the solution of a differential problem, and that they carry a part of the stress which the split of

Equation (3.43) cannot represent implicitly. Section 3.4.5 isolates the second of the two.

A separate measurement settles a further question about the stabilization, which the earlier sections leave open. If the flow in the region of interest were diffusion dominated, a streamline stabilization would have nothing to stabilize and its effect would be a numerical artefact. The cell Péclet number,

$$\mathrm{Pe} = \frac{\|\overline{\mathbf{u}}\|\, h}{2\,(\nu + \nu_{t,\mathrm{eff}})}, \tag{3.78}$$

measured on the final fields over the near-wake window of Equation (3.67), has a median between 2.1 and 3.7 across the closures, a ninety-fifth percentile between 13 and 371, and exceeds unity in between the 61% and the 73% of the cells. The problem is therefore convection dominated in the region where the closures are compared, as obviously expected, and the Péclet number is the numerical evidence that a streamline stabilization is needed there rather than an assumption.

Two experiments follow from this and are left for future work: a finer mesh, and a lower Reynolds number. Either would bring the cell Péclet number below unity throughout the window and remove the need for stabilization altogether. The question they would answer is whether the closure increment and the solver convergence do better.

### Net Output Updates

Every convergence number reported so far is the relative velocity update of Equation (3.40). That quantity compares two consecutive velocity iterates and says nothing directly about the second unknown of the coupled problem, which is the stress the closure returns. A closure that is a transport equation converges together with the velocity field, so for the classical reference the two are the same statement. A closure that is a network evaluated on $\nabla\mathbf{u}_n$ has no such guarantee. A second diagnostic is therefore introduced, the relative increment of the raw closure prediction between two consecutive iterates,

$$\delta R = \frac{\|\boldsymbol{\tau}(\mathbf{u}_{n+1}) - \boldsymbol{\tau}(\mathbf{u}_n)\|}{\|\boldsymbol{\tau}(\mathbf{u}_{n+1})\|}, \tag{3.79}$$

with the norm taken over the whole field and over the three components. The two predictions entering Equation (3.79) are the raw network outputs, taken before the closure under-relaxation of Section 3.3.2 is applied. Relaxation shrinks the field the solver assembles by construction, so a diagnostic measured after it would fall as the relaxation factor falls, whether or not the network output is settling.

Table 3.19 reports the two quantities side by side for the tensor-basis closure. Its second block anticipates the tensor split introduced in Section 3.4.5 and is

placed here so that the two diagnostics can be read together. The gap between the two columns is one order of magnitude wide throughout. At a velocity update of $1.0 \times 10^{-2}$ the prediction the solver is being handed still changes by 13% from one iteration to the next.

**Table 3.19:** Velocity update and closure increment at the last iteration, tensor-basis closure, grad-div stabilization only, 150 iterations. The two columns are the two unknowns of the coupled problem, and they are not at the same stage of convergence. All runs use the scalar split of Equation (3.43).

| Split | Filter | $\delta u$ | $\delta R$ |
|---|---|---|---|
| Scalar, Equation (3.43) | none | 3.1e-2 | 2.5e-1 |
| | input | 4.0e-2 | 1.3e-1 |
| Tensor, Equation (3.83) | none | 1.0e-2 | 1.3e-1 |
| | input | 1.8e-2 | 4.5e-2 |

Three consequences follow, and they qualify every convergence table of this thesis.

- The velocity update measures the consistency between the momentum solve and the closure it is currently being given. It does not measure the convergence of the coupled problem, which requires both unknowns to stop moving.
- The tables ordered by $\delta u$ remain valid as relative comparisons between configurations measured the same way. None of them establishes that a fixed point has been reached.
- An accuracy reported at a given iteration is a photograph of a field that is still moving, which is one reason why the error can change appreciably between neighbouring filtering configurations of the same closure.

The target that follows from Equation (3.79) is $\delta R \to 0$, and no machine-learning closure of this work comes close to it. What that target asks for is a closure whose output cannot move faster than the field it is evaluated on, which is a Lipschitz bound on the map from the mean-flow features to the stress. Two ways of imposing it were tried. Enforcing it as a hard constraint, by spectral normalization of every linear layer so that each has operator norm at most one [62], does produce the bound but costs too much: the closure loses enough expressive power that its *a priori* accuracy is no longer usable, and a closure that is smooth and wrong is of no interest to the solver. Enforcing it softly, as the penalty on the gradient of the coefficient network with respect to its own inputs described in Section 3.2.3, is the compromise adopted here for the PINNs architectures: it retains the accuracy and

buys part of the regularity. Neither is sufficient if the goal is the perfect classical convergence, and the gap between the two is the most concrete open direction this diagnostic points to.

### *Anisotropy splitting* costs convergence but gains accuracy

The split of Equation (3.43) sends into the momentum matrix the quantity $-2\nu_{t,\mathrm{eff}}\mathbf{S}$. That quantity is a scalar multiple of $\mathbf{S}$, so it is parallel to $\mathbf{S}$ at every point. A scalar can rescale a tensor and cannot rotate it. Figure 3.19 shows the two cases side by side. Writing $\mathbf{S}_{\mathrm{dev}}$ for the deviatoric strain rate and decomposing the predicted stress into the part parallel to it and the part orthogonal to it,

$$\boldsymbol{\tau}_{\mathrm{dev}} = \boldsymbol{\tau}_{\parallel} + \boldsymbol{\tau}_{\perp}, \qquad \boldsymbol{\tau}_{\parallel} = \frac{\boldsymbol{\tau}_{\mathrm{dev}} : \mathbf{S}_{\mathrm{dev}}}{\mathbf{S}_{\mathrm{dev}} : \mathbf{S}_{\mathrm{dev}}}\, \mathbf{S}_{\mathrm{dev}}, \qquad \boldsymbol{\tau}_{\perp} : \mathbf{S}_{\mathrm{dev}} = 0, \tag{3.80}$$

the least-squares eddy viscosity of Equation (3.42) absorbs $\boldsymbol{\tau}_{\parallel}$ exactly and leaves $\boldsymbol{\tau}_{\perp}$ untouched. The orthogonal part is therefore not a remainder that a better fit could reduce. It is the part of the prediction the scalar split is structurally unable to represent, and it goes into the right-hand side in full.

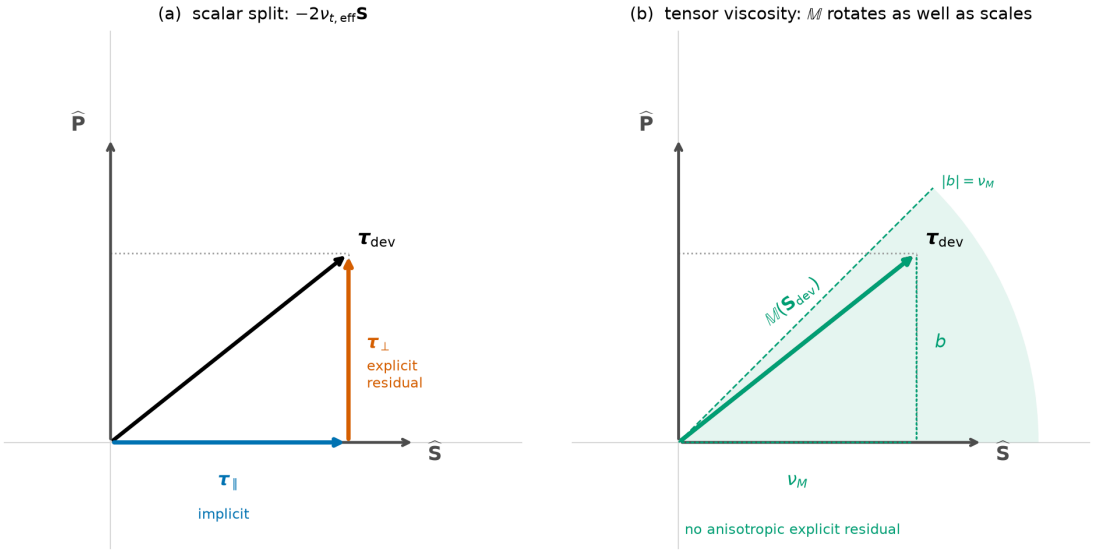



**Figure 3.19:** The predicted deviatoric stress in the two-dimensional space of symmetric traceless tensors. A scalar eddy viscosity can only represent the component parallel to the strain rate, so the orthogonal component is left in the explicit residual. The tensor coefficient of Equation (3.83) represents both, within the coercivity cone $|b| \leq \nu_M$.

The Picard map enters as the divergence of a second-order operator evaluated at the previous iterate. The amplification factor of such an explicit treatment scales

as

$$\frac{\|\boldsymbol{\tau}_\perp\|}{2\left(\nu+\nu_{t,\mathrm{eff}}\right)\|\mathbf{S}\|}, \tag{3.81}$$

i.e. the ratio between the diffusion treated explicitly and the diffusion treated implicitly, which is the classical stability criterion of a semi-implicit splitting of an anisotropic diffusion.

Equation (3.81) is testable directly. The closure is run a second time with its own anisotropy discarded, imposing the Boussinesq projection of its own prediction, $\boldsymbol{\tau} = \frac{2}{3}k\mathbf{I} - 2\nu_t\mathbf{S}$ with $k$ and $\nu_t$ taken from the network. This makes the closure formally identical, from the numerical point, to the transported reference and leaves nothing in the explicit residual except the isotropic part. Table 3.20 reports the comparison.

**Table 3.20:** Median relative velocity update, tensor-basis closure, grad-div stabilization only, with the full predicted tensor and with its Boussinesq projection. The two rows differ only in what is imposed on the momentum equation. The last row is the ratio between them. All runs use the scalar split of Equation (3.43).

| Imposed stress | No filter | Input | Output | Input+output |
|---|---|---|---|---|
| Full tensor | 1.5e-1 | 1.7e-1 | 6.4e-2 | 3.2e-2 |
| Boussinesq projection | 1.8e-2 | 2.6e-2 | 9.4e-3 | 5.2e-3 |
| Ratio | 8.4 | 6.7 | 6.9 | 6.2 |

Discarding the anisotropy improves the iteration by a factor between six and eight and a half, uniformly across the four filtering configurations. This is a diagnostic and not a configuration to adopt, since it throws away exactly what the tensor bases of Section 3.2 exist to represent, and the field pays for it. On the two filtered configurations with the smallest residual the stress error over the near-wake window rises from about 0.46 to about 0.66 once the anisotropy is removed. The measurement therefore establishes the cost of the orthogonal part and, at the same time, that removing it is not the remedy.

### A tensor eddy viscosity

The obstacle just identified is a property of the parametrization, not of the closure. It disappears if the implicit coefficient is allowed to be a tensor.

In two dimensions the symmetric traceless tensors form a space of dimension two. An orthonormal basis of that space is

$$\mathbf{E}_1 = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}, \qquad \mathbf{E}_2 = \frac{1}{\sqrt{2}}\begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}, \qquad \mathbf{E}_i : \mathbf{E}_j = \delta_{ij}. \tag{3.82}$$

For an incompressible flow $\mathbf{S}$ is traceless, so both $\mathbf{S}$ and the deviatoric part of the predicted stress live in the plane spanned by Equation (3.82). A linear map of that plane into itself is a two by two matrix, and a two by two matrix can rotate. Let $\widehat{\mathbf{S}} = \mathbf{S}_{\mathrm{dev}}/\|\mathbf{S}_{\mathrm{dev}}\|$ and let $\widehat{\mathbf{P}}$ be the unit tensor orthogonal to it in that plane. The implicit coefficient is replaced by the fourth-order tensor

$$\mathbb{M}(\mathbf{A}) = \nu_M\,\mathbf{A} + (\mathbf{A}:\mathbf{B})\,\widehat{\mathbf{S}} + (\mathbf{A}:\widehat{\mathbf{S}})\,\mathbf{B}, \qquad \mathbf{B} = b\,\widehat{\mathbf{P}}, \tag{3.83}$$

so that $\mathbb{M}(\mathbf{S}_{\mathrm{dev}}) = \nu_M \mathbf{S}_{\mathrm{dev}} + b\,\|\mathbf{S}_{\mathrm{dev}}\|\,\widehat{\mathbf{P}}$. Choosing

$$b = -\frac{\boldsymbol{\tau}_\perp : \widehat{\mathbf{P}}}{2\,\|\mathbf{S}_{\mathrm{dev}}\|}, \tag{3.84}$$

makes $-2\,\mathbb{M}(\mathbf{S}_{\mathrm{dev}})$ reproduce the orthogonal part of the prediction as well as the parallel one, so the anisotropic explicit residual vanishes identically. The split becomes

$$\boldsymbol{\tau} = -2\,\mathbb{M}\Big(\mathbf{S}(\overline{\mathbf{u}})\Big) + \boldsymbol{\tau}^{\mathbb{M}}_{\mathrm{res}}, \qquad \boldsymbol{\tau}^{\mathbb{M}}_{\mathrm{res}} \equiv \boldsymbol{\tau} + 2\,\mathbb{M}\Big(\mathbf{S}(\mathbf{u}_n)\Big), \tag{3.85}$$

which reduces to Equation (3.43) when $b = 0$.

Three properties make Equation (3.85) usable.

**The fixed point does not move.** For any $\mathbb{M}$ whatsoever, $-2\mathbb{M}(\mathbf{S}(\mathbf{u}_h)) + [\boldsymbol{\tau} + 2\mathbb{M}(\mathbf{S}(\mathbf{u}_n))] = \boldsymbol{\tau}$ at $\mathbf{u}_h = \mathbf{u}_n$. This is the same algebraic identity that already justifies the scalar split and the boost factor $\beta_{\mathrm{boost}}$ of Section 3.3.3. The reparametrization moves stress between the matrix and the right-hand side and never changes the stress that is imposed at convergence, so the accuracy of the converged field is that of the closure and any gain is read entirely on the convergence.

**Coercivity is an explicit condition.** In the basis $(\widehat{\mathbf{S}}, \widehat{\mathbf{P}})$ the map of Equation (3.83) is symmetric with eigenvalues $\nu_M \pm |b|$, so the bilinear form $2\int_\Omega \mathbb{M}(\mathbf{S}(\overline{\mathbf{u}})) : \mathbf{S}(\mathbf{w})\,d\Omega$ is positive semidefinite if and only if $|b| \le \nu_M$. The condition $|b| \le \nu_M$ is the exact tensor analogue of the clipping $\nu_{t,\mathrm{eff}} \ge 0$ of Equation (3.42), and like it, it is a condition on the implicit operator and not on the closure.

**One dimensionless parameter closes the construction.** Equation (3.84) diverges where $\|\mathbf{S}_{\mathrm{dev}}\| \to 0$. The stress it produces stays bounded, since $2|b|\|\mathbf{S}_{\mathrm{dev}}\| = \|\boldsymbol{\tau}_\perp\|$, but the viscosity does not, and an unbounded viscosity in a far-field cell destroys the solution there. The bound is read from Equation (3.81) itself,

$$|b| \le c_{\mathbb{M}}\,(\nu + \nu_{t,\mathrm{eff}}), \qquad \nu_M = \max\Big(\nu_{t,\mathrm{eff}},\ |b|\Big), \tag{3.86}$$

that is, the anisotropic viscosity is worth at most $c_{\mathbb{M}}$ times the diffusion already present in the matrix. The bound is written on $\nu + \nu_{t,\mathrm{eff}}$ and not on $\nu_{t,\mathrm{eff}}$ alone,

since a bound on the latter would switch $b$ off exactly in the cells where the clipping has set $\nu_{t,\text{eff}} = 0$, which are the counter-gradient cells where the explicit anisotropy does the most damage. At $c_{\mathbb{M}} = 0$ Equation (3.85) reduces to the scalar split identically, which is used below as a single-variable reference.

Every closure is deployed with the reconstruction whose eddy viscosity is non-negative by construction rather than obtained from the least-squares fit of Equation (3.42). The fit is clipped at zero, so it vanishes identically wherever the predicted stress is counter-gradient, which is a substantial fraction of the domain; a reconstruction that produces a strictly positive eddy viscosity everywhere leaves the anisotropic coefficient a margin in those cells as well.

Table 3.21 compares the two splits on the tensor-basis closure, with the same checkpoint, the same stabilization and the same iteration budget, so that the splitting is the only variable.

**Table 3.21:** Median relative velocity update and errors over the near-wake window, tensor-basis closure, grad-div stabilization only, 150 iterations. The scalar and tensor splits differ only in whether the orthogonal part of the prediction is treated implicitly. Errors are the relative $L^2$ error of the velocity vector and the aggregate stress error of Section 3.1.4, both restricted to the window of Section 3.4.1.

| Split | Quantity | No filter | Input | Output | Input+output |
|---|---|---|---|---|---|
| Scalar | $\delta u$ | 2.6e-2 | 2.4e-2 | 2.3e-2 | 3.2e-2 |
| | $\varepsilon_{L^2}(\overline{\mathbf{u}})$ | 0.136 | 0.105 | 0.651 | 0.301 |
| | $\varepsilon_{L^2}^{\text{tensor}}$ | 0.642 | 0.553 | 0.812 | 0.572 |
| Tensor, $c_{\mathbb{M}} = 4$ | $\delta u$ | 1.3e-2 | 1.8e-2 | 8.6e-3 | **6.7e-3** |
| | $\varepsilon_{L^2}(\overline{\mathbf{u}})$ | **0.073** | 0.126 | 0.439 | 0.297 |
| | $\varepsilon_{L^2}^{\text{tensor}}$ | 0.589 | 0.647 | 0.694 | **0.489** |

The velocity update improves in all four filtering configurations, by a factor of 2.0, 1.3, 2.7 and 4.8 respectively, and the gain grows with the amount of filtering. The accuracy does not pay for it, which is what the invariance of the fixed point predicts. Six of the eight error entries improve or stay unchanged, the two that degrade are both in the input-filter column, and the unfiltered configuration improves on both criteria at once, from 0.136 to 0.073 on the velocity while its residual halves. The closure increment of Equation (3.79) improves as well, from $2.5 \times 10^{-1}$ to $1.3 \times 10^{-1}$ without filters and from $1.3 \times 10^{-1}$ to $4.5 \times 10^{-2}$ with the input filter, so the gain is on the coupled problem and not on the velocity iterate alone.

Table 3.22 sweeps the single parameter of Equation (3.86). The behaviour is monotone from 1 to 4 and degrades at 16, so the adopted value is bracketed on both sides rather than chosen.

Above $c_{\mathbb{M}} = 1$ the bound of Equation (3.86) sets $\nu_M = |b| > \nu_{t,\text{eff}}$ wherever it

**Table 3.22:** Median relative velocity update of the tensor-basis closure against the bound $c_{\mathbb{M}}$ of Equation (3.86). The value $c_{\mathbb{M}} = 0$ is the scalar split of Equation (3.43).

| $c_{\mathbb{M}}$ | No filter | Input | Output | Input+output |
|---|---|---|---|---|
| 0 | 2.6e-2 | 2.4e-2 | 2.3e-2 | 3.2e-2 |
| 1 | 2.5e-2 | 2.5e-2 | 1.2e-2 | **6.3e-3** |
| 2 | 2.1e-2 | 2.1e-2 | 1.1e-2 | 6.7e-3 |
| 4 | 1.3e-2 | **1.8e-2** | **8.6e-3** | 6.7e-3 |
| 16 | **1.2e-2** | 2.8e-2 | 9.4e-3 | 8.3e-3 |

bites, so the construction does two things at once. It makes the rotation implicit, which is the mechanism it was built for, and it raises the isotropic viscosity in the matrix, which stabilizes any Picard iteration whatever the reason. On the tensor-basis closure the second effect is not small, with $\nu_M/\nu_{t,\text{eff}}$ close to 3.5 in the median. Two controls separate them. The first raises the isotropic viscosity by the same factor through $\beta_{\text{boost}} = 3.5$ and leaves the excess in the explicit residual, which is the added viscosity without the rotation. The second sets $c_{\mathbb{M}} = 1$, where Equation (3.86) gives $\nu_M = \nu_{t,\text{eff}}$ almost everywhere, which is the rotation without the added viscosity.

**Table 3.23:** Improvement of the median relative velocity update over the scalar split, tensor-basis closure. A value below one is a degradation.

| Ingredient | No filter | Input | Output | Input+output |
|---|---|---|---|---|
| Added isotropic viscosity only, $\beta_{\text{boost}} = 3.5$ | **1.9** | 0.8 | 0.8 | 0.8 |
| Rotation only, $c_{\mathbb{M}} = 1$ | 1.0 | 0.9 | 1.9 | **5.1** |
| Both, $c_{\mathbb{M}} = 4$ | **2.0** | **1.3** | **2.7** | 4.8 |

Table 3.23 separates the two cleanly, and the separation is monotone in the amount of filtering. Without filters the gain comes from the added viscosity and the rotation contributes nothing. As the filters are switched on the added viscosity stops helping and becomes a small penalty, while the rotation grows to a factor of five. The reading is mechanical. The filters already smooth the fields, so extra diffusion becomes first redundant and then over-damping, whereas the misalignment between the predicted stress and the strain rate is a structural property of the closure that no filter touches. Only the change of parametrization removes it.

**A diagnostic available before the run.** The construction also measures the quantity it was designed to remove. The ratio $|b|/\nu_M$ is the fraction of the implicit coefficient that is anisotropic, and it says how much of the predicted stress the

scalar split was forced to leave explicit. For the tensor-basis closure it is 1.000 in the median, with the bound of Equation (3.86) active in 31% of the cells: its predicted stress is almost entirely orthogonal to the strain rate. This is the *a priori* counterpart of the factor of six to eight measured in Table 3.20, and the two measurements agree without sharing anything but the closure.

**The plain MLP is unaffected.** The MLP closure diverges under both splits and in all four filtering configurations, with a velocity update between 1.0 and 4.0. Its reconstruction carries no eddy viscosity of its own, so $\nu_{t,\mathrm{eff}}$ comes from the least-squares fit of Equation (3.42) and vanishes over a large part of the domain, which leaves Equation (3.86) with almost no margin. A change of parametrization redistributes a prediction between matrix and right-hand side and cannot repair a prediction that the momentum equation cannot absorb in any arrangement.

### 3.4.6 Summary

The coupling experiments of this chapter concern the two local closures and their physics-informed variants, and they support four statements.

The tensor bases survive the coupling, and the multilayer closure does not. Under the adopted stabilization the tensor-basis closure produces a field whose velocity error on the window is 0.048, half that of the one-equation reference, while the multilayer closure does not produce a field at all: its final update lies between 0.66 and 1.84 across the four filtering configurations, and its velocity error reaches 163 where the iteration diverges outright. The margin measured *a priori* in Section 3.2 is therefore not merely preserved by the coupling, it is widened by it. Figure 3.20 puts the best field each of the two produces beside the reference, and the two closures are told apart by the stress components and not by the velocity.

A metric restricted to the near wake is necessary and is not sufficient. The unfiltered tensor-basis run reports 0.062 on the whole domain and 0.351 on the window; the multilayer closure, in the configuration where it does produce a field, reports the lowest stress error of this chapter on the window, 0.294, and a shear-stress error of 5.09 outside it. Both the window and its complement are reported for every configuration for this reason.

The physics-informed variant is the only closure that reaches the tolerance under both stabilization forms, $1.2 \times 10^{-5}$ and $1.1 \times 10^{-5}$, and in three filtering configurations out of four it stops before the iteration budget, which no other closure does. Its accuracy does not follow: its stress error outside the window is 2.77 in all eight runs, four filters by two stabilizations, constant to four digits. That the number does not move with either the filter or the stabilization places the discrepancy in the network rather than in the coupling, which is where the mismatch between the data and residual terms of Equation (3.22) would place

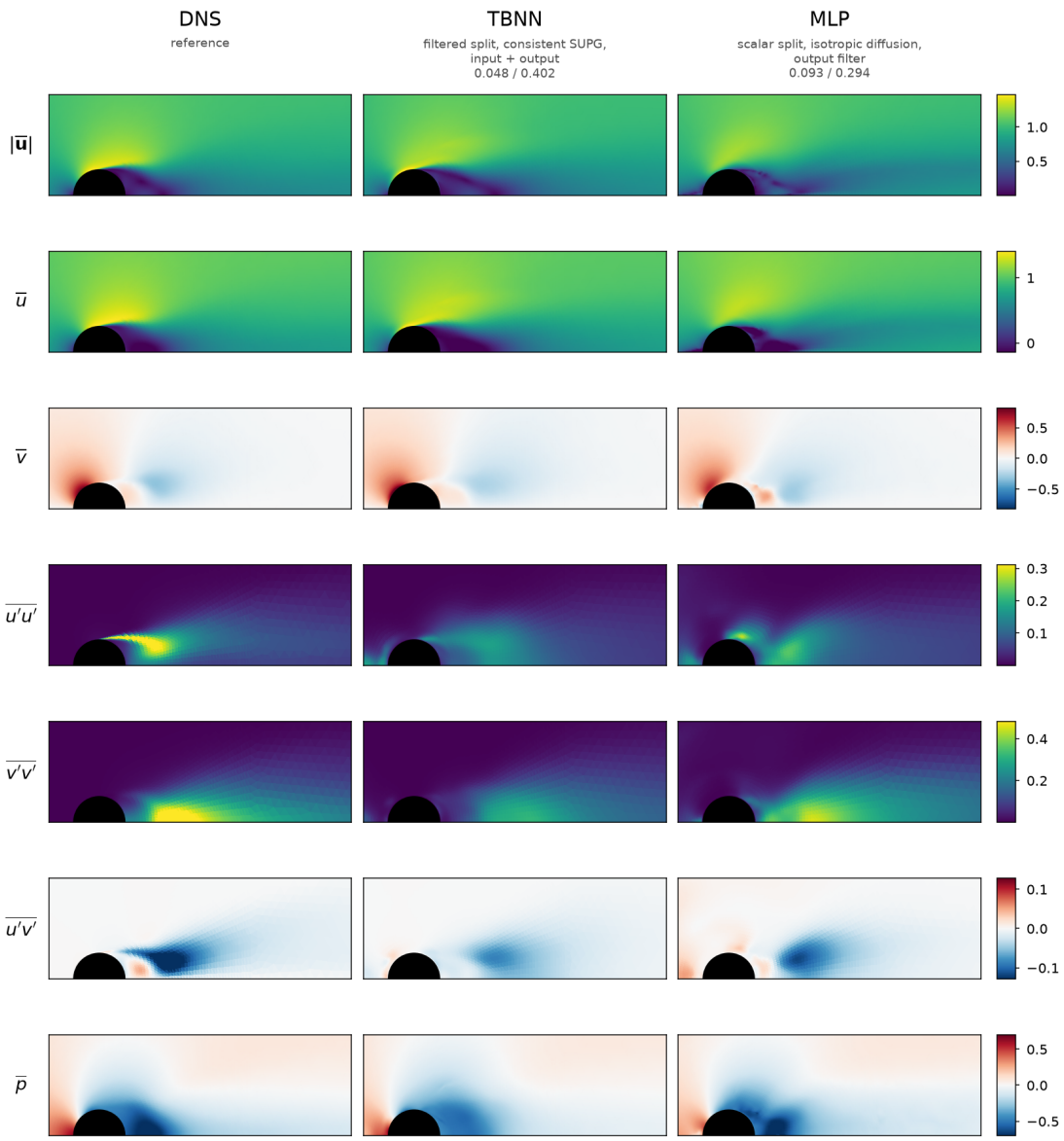



**Figure 3.20:** Converged fields of the best configuration of each local closure against the reference, over the near-wake window. Rows, from top to bottom: velocity magnitude, the two velocity components, the three Reynolds stress components and the pressure. All the maps in a row share one colour scale. Each column is labelled with its own configuration and with its velocity and stress errors, and the two are not the same: the multilayer closure produces no field under the adopted stabilization, so its best deployable run is the one of Section 3.4.3. This is the only figure in which its converged field appears.

it. The closure is not usable as it stands, and the combination of unconditional convergence with a network-level error is the reason it is worth repairing rather than discarding.

The configuration adopted for the comparisons of the following chapters is fixed here. It is the consistent streamline-upwind form of Equation (3.68), the filtered split of Equation (3.77), both filters active, the Taylor–Hood discretization of Section 3.3.4 on the half-domain mesh at $\mathrm{Re} = 11\,000$, at most 150 Picard iterations, and the grad-div coefficient at $\gamma = 0.2$. Both filters are active because that configuration is simultaneously the most accurate and the best converged of the four, so the choice does not trade one criterion against the other. The tensor-basis closure of Section 3.2.2 in that configuration is the local reference against which every non-local closure of the following chapters is measured, and its measured behaviour is $\delta u = 1.7 \times 10^{-3}$, $\varepsilon_{L^2}(|\overline{\mathbf{u}}|) = 0.048$ and $\varepsilon_{L^2}^{\mathrm{tensor}} = 0.402$ on the window, with $R^2(\overline{u'v'}) = 0.567$ computed over the whole domain.

On the numerics, six findings hold across every configuration tested.

- Whether the coupled problem is solvable is decided by the stabilization operator, not by the closure. Under the consistent streamline-upwind form alone no local machine-learning closure converges; under an isotropic, non-consistent, unconditionally coercive form two of the four do, with everything else identical; and under the consistent form combined with the filtered split of Equation (3.77) all of them except the multilayer closure produce a bounded field.

- Weakening the stabilization improves accuracy and degrades convergence, because the non-consistent form perturbs the discrete problem by a term proportional to its own strength. The two criteria therefore rank configurations differently, and the ranking criterion has to be stated.

- The output filter is a precondition for convergence and the input filter is a repair for a closure fed a noisy gradient field. Every converged configuration includes the former; only the closure whose unfiltered stress is unusable benefits substantially from the latter.

- The discretization is not the obstacle. With no stabilization at all the classical reference reaches the solver tolerance on the same mesh and with the same iteration, so the whole of the gap belongs to the pair closure and solver.

- A scalar implicit coefficient cannot represent the part of the predicted stress orthogonal to the strain rate, and that part is what the iteration has to carry explicitly. Discarding it improves the residual by a factor between six and eight; representing it with a tensor coefficient improves the residual by up to

a factor of five and costs no accuracy, because the fixed point of the split is invariant.

- The velocity update is not the convergence of the coupled problem. Measured at the same iteration, the closure prediction still moves by one order of magnitude more than the velocity field.

On the closures, and on the window that measures them, the local family reaches a velocity error between 0.048 and 0.085 against 0.105 for the uncalibrated classical reference, and a stress error between 0.29 and 0.53 against 0.521. The stress margin is smaller than the *a priori* metrics of Section 3.2 would suggest, and it disappears entirely for the physics-informed variant. The *a priori* ordering of the closures is not preserved *a posteriori*, and the reasons are that in the solver the net sees unphysical, out of distribution, fields and then the solver does not use the predicted stress but its divergence, weighted by the physical area of each cell.

Whether spatial non-locality changes this picture is the question the next two chapters address, with the same solver, the same mesh, the same Reynolds number and the same convergence criterion.

# Chapter 4

# Non-Local closure for RANS: VCNN

The closures of Chapter 3 predict the Reynolds stress at a point from mean-flow quantities evaluated *at that same point*. This locality is an assumption, not a property of the tensor. Since it represents turbulent fluctuations of the flow, and turbulence is highly non local, the stress at a point depends on the flow in a neighbourhood of that point rather than on the point alone. A local closure can only recover that dependence indirectly, through whatever correlation happens to exist between the local mean flow and its surroundings in the training data.

This chapter relaxes that assumption. The Vector-Cloud Neural Network predicts the stress at a point from a cloud of neighbouring points, and is built so that the prediction does not depend on how the reference frame is oriented. The input of the closure is therefore a neighbourhood rather than a point, and every choice described below follows from that one fact. The neighbourhood is an unordered set, so the network must be invariant to how its members are numbered, and that requirement fixes its form. The neighbourhood must not move as the solution changes, or the fixed-point iteration of the solver would evaluate the closure on a different set at every step, which fixes how its size is chosen. And the neighbourhood carries the unconverged field into the closure through as many points as it contains, which is what makes the coupling much harder than it is for a local closure.

Section 4.1 describes the architecture, the invariance argument that underpins it, the reconstruction that maps the network output onto an admissible stress tensor, and the accuracy the closure reaches against reference data. Section 4.2 embeds it in the FEM solver of Section 3.4 under the configuration adopted in Chapter 3, and compares the resulting mean fields against DNS and against the local closures. Section 4.3 focuses on the size of the cloud, which is the parameter that sets how much non-local information reaches the network, and derives a closed-form

expression for it by symbolic regression. Section 4.4 collects the comparisons that motivate each choice, both on reference data and inside the solver.

## 4.1 VCNN

The construction follows Zhou et al. [31], who introduced the vector-cloud representation as a frame-independent surrogate for the Reynolds stress transport equations. The starting observation is that a non-local closure needs a way to describe a neighbourhood that does not depend on how that neighbourhood is labelled. Two requirements follow.

- The description must be invariant to the ordering of the neighbouring points. A neighbourhood is a set, and any indexing of it is arbitrary.
- The description must be invariant to rigid rotations and reflections of the reference frame, in the sense of Equation (3.19), so that the predicted stress transforms as a tensor rather than as an array of numbers.

Both are obtained here by the same mechanism, a bilinear pooling of the cloud against a matrix of frame-dependent vectors, described in the rest of this section.

### 4.1.1 The cloud

Around every point $\mathbf{x}_c$ at which the stress is required, a circular cloud of radius $r(\mathbf{x}_c)$ is collected from the mesh. The point itself is always retained and placed first. The remaining members are drawn uniformly from the mesh points falling inside the disc, up to a fixed cloud size $K = 150$. When the disc contains fewer than $K$ points, the missing rows are zero-padded and flagged by a binary mask, and every subsequent operation is masked and normalized by the number of actual members, so that padded rows contribute nothing. On the four reference simulations the clouds contain on average between 122 and 129 actual points out of 150, and no cloud is ever discarded for being too small.

The radius $r(\mathbf{x}_c)$ is the only genuinely free geometric parameter of the construction, and Section 4.3 is devoted to analyze it. A first formulation made it a function of the local flow state, as done in the reference, through a ratio between the velocity magnitude and the sum of the strain and rotation rates, which is an advection length, the distance a fluid element travels while the local structure deforms appreciably. That choice is physically sounded. However, it is also unusable inside a fixed-point iteration. A radius that depends on the current field must be recomputed at every iteration, and the set of mesh points falling inside the disc must be resampled with it, but a point entering or leaving the cloud in a single

step makes the closure discontinuous and the coupling harder. The radius adopted here is therefore obtained by distilling that same expression onto quantities known before the solver starts, the wall distance and the angular position around the cylinder, as described in Section 4.3. Two properties already matter at this stage. It is computable once per mesh and never recomputed as the solver iterates. Its range is $[0.068D,\ 0.505D]$, so the largest cloud has a diameter equal to the cylinder diameter, the characteristic length of the flow. Figure 4.1 shows how the disc varies across the domain.

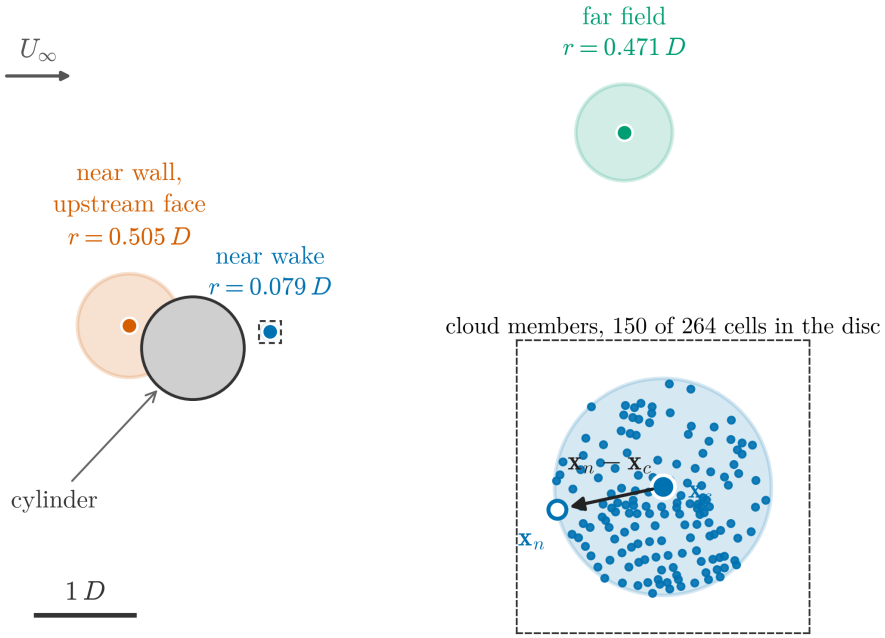



**Figure 4.1:** Geometry of the vector cloud. The sampling radius is set by the wall distance and the angular position alone, so it is fixed once for a given mesh. Clouds close to the cylinder are markedly smaller than clouds in the far field.

### 4.1.2 Frame-dependent and frame-independent inputs

Each member $n$ of the cloud contributes a row $\mathbf{q}_n$ of a feature matrix $\mathbf{Q} \in \mathbb{R}^{K\times F}$, with $F$ the dimension of the Feature space. The row is split according to how its entries behave under a change of reference frame. Its first four components are the frame-dependent block,

$$\big(\mathbf{q}_n\big)_{1:4} = \big(\hat{\mathbf{x}}_n,\ \overline{\mathbf{u}}_n\big), \qquad \hat{\mathbf{x}}_n = \frac{\mathbf{x}_n - \mathbf{x}_c}{\|\mathbf{x}_n - \mathbf{x}_c\|}, \tag{4.1}$$

i.e. the unit vector pointing from the cloud centre to the member, followed by the mean velocity at the member. Both are genuine two-dimensional vectors, so under a change of frame $\mathbf{R}$ every row transforms as $\big(\mathbf{q}_n\big)_{1:4} \mapsto \big(\mathbf{q}_n\big)_{1:4}\tilde{\mathbf{R}}$, with $\tilde{\mathbf{R}}$ block diagonal. This block is the only place where frame-dependent information enters, and it carries true vectors alone. That restriction is what the invariance argument of the next section rests on.

The remaining components of $\mathbf{q}_n$, from the fifth onward, are invariant under rotations and reflections. Eight of them are magnitudes and geometric ratios,

$$\Big(\theta_n,\ \|\overline{\mathbf{u}}_n\|,\ \eta_n,\ \rho_n,\ \rho'_n,\ \|\mathbf{\Omega}_n\|,\ \|\mathbf{S}_n\|,\ \|\nabla\overline{p}_n\|\Big), \tag{4.2}$$

where $\theta_n$ is the cell volume of the member normalized by the mean cell volume of the cloud, $\eta_n$ the wall distance, $\rho_n$ a monotone compression of the distance from the cloud centre, and

$$\rho'_n = \frac{\|\overline{\mathbf{u}}_n\|}{U_\infty}\Big(\varepsilon_1 - \cos\alpha_n\Big), \qquad \cos\alpha_n = \frac{\overline{\mathbf{u}}_n \cdot (\mathbf{x}_n - \mathbf{x}_c)}{\|\overline{\mathbf{u}}_n\|\,\|\mathbf{x}_n - \mathbf{x}_c\|}, \qquad \varepsilon_1 = 1.05, \tag{4.3}$$

a velocity-weighted measure of the alignment between the member's velocity and its position relative to the cloud centre, which is largest when the velocity points towards the centre and smallest when it points away from it. Two further invariants complete the row,

$$s_n^{\mathrm{proj}} = \hat{\mathbf{x}}_n^{\top}\,\mathbf{S}_n\,\hat{\mathbf{x}}_n, \tag{4.4}$$

$$\omega_n^{\mathrm{proj}} = \omega_n\Big(\hat{\mathbf{x}}_n \times \overline{\mathbf{u}}_c\Big)_z, \tag{4.5}$$

with $\omega_n$ the scalar mean vorticity at the member and $\overline{\mathbf{u}}_c$ the mean velocity at the cloud centre. Both are invariant, and both are signed. Equation (4.4) is a vector-tensor-vector contraction, invariant under any orthogonal change of frame by construction. Equation (4.5) is the product of two pseudoscalars, the vorticity and the out-of-plane component of a cross product between two in-plane vectors, and is therefore a true scalar. Their purpose is to restore, in invariant form, orientation information that the magnitudes of Equation (4.2) discard. The magnitudes alone cannot distinguish a strain field aligned with the direction of a neighbour from one perpendicular to it, whereas Equation (4.4) can.

The inputs whose distribution spans several orders of magnitude are first compressed with a signed logarithm, $x \mapsto \mathrm{sign}(x)\log(1+|x|)$. These are the wall distance and the three norms of vorticity, strain rate and pressure gradient, all of which develop a heavy tail close to the wall, where a standardization applied to the raw values would be set by the extremes alone. Every input is then standardized on statistics computed on the training split alone, as described in Section 3.1. The frame-dependent block of Equation (4.1) is standardized differently from the rest,

and the difference matters for the argument of the next section. Each of its two vector pairs is divided by the single scalar $\sqrt{\mathbb{E}[\|\cdot\|^2]}$ and no mean is subtracted, so the operation commutes with any rotation of the frame. Standardizing those columns one at a time, as the invariant components are standardized, would subtract a fixed vector and apply a different scale to each component, and both operations single out the axes of the training frame. The mask is reapplied after standardization, so padded rows are exactly zero.

### 4.1.3 Bilinear pooling and the invariance argument

The invariant components of $\mathbf{q}_n$ are mapped through a shared embedding network of two hidden layers, $10 \to 32 \to 48$, with Gaussian Error Linear Unit (GELU) activations [63], collecting the outputs in a matrix $\mathbf{G} \in \mathbb{R}^{K\times 48}$. A narrower copy $\mathbf{G}^* \in \mathbb{R}^{K\times 4}$ is taken as the first four columns of $\mathbf{G}$. The truncation is what keeps the model small. The descriptor assembled below has one row per embedding channel and one column per component of $\mathbf{G}^*$, so its size, and with it the input width of the fitting network, grows with the number of columns retained. Keeping four instead of all forty-eight reduces that width by a factor of twelve. The reference formulation reports that the leading columns already carry most of the information the embedding extracts from the invariant inputs [31], which is the ground on which the truncation is adopted here.

Writing $n_{\text{eff}}$ for the number of real cloud members and $\mathbf{Q}^{\text{vec}}$ for the frame-dependent block of $\mathbf{Q}$, the two pooled matrices and their product are

$$\mathbf{L} = \frac{\mathbf{G}^\top \mathbf{Q}^{\text{vec}}}{n_{\text{eff}}} \in \mathbb{R}^{48\times 4}, \qquad \mathbf{L}^* = \frac{\mathbf{G}^{*\top} \mathbf{Q}^{\text{vec}}}{n_{\text{eff}}} \in \mathbb{R}^{4\times 4}, \qquad \mathbf{D} = \mathbf{L}\,\mathbf{L}^{*\top} \in \mathbb{R}^{48\times 4}. \tag{4.6}$$

The matrix $\mathbf{D}$ is flattened into a 192-dimensional vector and fed to a fitting network of one hidden layer, $192 \to 64 \to 6$.

Both required invariances follow from Equation (4.6) directly. Permutation invariance holds because every entry of $\mathbf{L}$ and $\mathbf{L}^*$ is a sum over cloud members, and a sum does not depend on the order of its terms. Frame invariance follows by substitution. The embedding acts on invariant components, so $\mathbf{G}$ and $\mathbf{G}^*$ are unchanged by a rotation. Only $\mathbf{Q}^{\text{vec}}$ transforms, and it appears in $\mathbf{D}$ exactly twice,

$$\mathbf{D} = \frac{1}{n_{\text{eff}}^2}\, \mathbf{G}^\top\, \mathbf{Q}^{\text{vec}}\, \mathbf{Q}^{\text{vec}\,\top}\, \mathbf{G}^*, \tag{4.7}$$

so that under $\mathbf{Q}^{\text{vec}} \mapsto \mathbf{Q}^{\text{vec}}\tilde{\mathbf{R}}$ the product $\mathbf{Q}^{\text{vec}}\tilde{\mathbf{R}}\tilde{\mathbf{R}}^\top\mathbf{Q}^{\text{vec}\,\top} = \mathbf{Q}^{\text{vec}}\mathbf{Q}^{\text{vec}\,\top}$ is recovered unchanged for any orthogonal $\tilde{\mathbf{R}}$. The entries of $\mathbf{Q}^{\text{vec}}\mathbf{Q}^{\text{vec}\,\top}$ are the pairwise inner products of the cloud rows, and inner products are invariant under rotations and reflections alike. Nothing in $\mathbf{D}$ moves.

This is why the composition of the frame-dependent block is restricted so tightly. A pseudoscalar placed among its columns would enter $\mathbf{Q}^{\mathrm{vec}}\mathbf{Q}^{\mathrm{vec}\,\top}$ as a product of two sign-flipping quantities in some entries and as a product of one sign-flipping and one invariant quantity in others, and the second kind would not survive a reflection.

The construction is therefore invariant under rotations and, in addition, under reflections. The reason is not in the reconstruction, which is equivariant under any orthogonal transformation, but in the inputs: the frame-dependent block of $\mathbf{q}_n$ carries true vectors alone, and every invariant component entering the embedding is a true scalar. In two dimensions the mean vorticity is a pseudoscalar and changes sign under reflection, which is why it enters only through the product of Equation (4.5), where it is multiplied by a second pseudoscalar and the product is a true scalar.

### 4.1.4 Reconstruction

The outputs of the fitting network are combined with a fixed set of tensor bases, following the same principle as the TBNN of Section 3.2. The bases are built from the raw mean-flow quantities at the cloud centre.

- $\mathbf{B}_0 = \mathbf{I}$, the identity.
- $\mathbf{B}_1 = \mathbf{S}/\|\mathbf{S}\|_F$, the normalized mean strain rate.
- $\mathbf{B}_2 = \mathbf{e}_u\mathbf{e}_u^\top - \frac{1}{2}\mathbf{I}$, built from the unit mean-velocity direction.
- $\mathbf{B}_3 = \mathbf{e}_p\mathbf{e}_p^\top - \frac{1}{2}\mathbf{I}$, built from the unit mean-pressure-gradient direction.
- $\mathbf{B}_4 = (\mathbf{S\Omega} - \mathbf{\Omega S})/\|\mathbf{S\Omega} - \mathbf{\Omega S}\|_F$, the normalized strain-rotation coupling.

The network emits six numbers. Five of them are the coefficients $\beta_0, \ldots, \beta_4$ of the bases, assembled into a single tensor $\mathbf{A}$, and $\kappa$ sets the scale,

$$\mathbf{A} = \sum_{k=0}^{4} \beta_k \, \mathbf{B}_k, \qquad \widehat{\boldsymbol{\tau}} = \frac{2\,\mathrm{softplus}(\kappa)\,\mathbf{A}\mathbf{A}^\top}{\mathrm{tr}\left(\mathbf{A}\mathbf{A}^\top\right)}. \tag{4.8}$$

Realizability is a property of the form itself and costs nothing to enforce. The product $\mathbf{A}\mathbf{A}^\top$ is symmetric and positive semidefinite for any $\mathbf{A}$, so both eigenvalues of $\widehat{\boldsymbol{\tau}}$ are non-negative, and they are strictly positive whenever $\mathbf{A}$ is non-singular. The normalization by the trace fixes the scale independently of the direction, so that

$$\mathrm{tr}\,\widehat{\boldsymbol{\tau}} = 2\,\mathrm{softplus}(\kappa) > 0, \qquad k = \tfrac{1}{2}\,\mathrm{tr}\,\widehat{\boldsymbol{\tau}} = \mathrm{softplus}(\kappa), \tag{4.9}$$

and the sixth output is therefore the predicted turbulent kinetic energy, while the five coefficients set the shape of the tensor alone. The reconstructed stress is

admissible for every coefficient the network can produce, with no post-hoc correction and no penalty term in the loss. The realizability terms of the composite loss of Section 3.1 are consequently switched off for this architecture, since they can never be active.

Two alternative heads were implemented on the same network, at parity in features, cloud, cloud size, embedding and fitting widths, activation, learning rate, number of epochs and loss, so that the head is the only difference between the three. The first bounds the anisotropic part against the isotropic one instead of normalizing the trace. The second is the learned equivariant reconstruction of the reference formulation, in which the tensor structure is fitted from data rather than prescribed. All three reach comparable accuracy on reference data, and the comparison is reported in Section 4.4, anyway accuracy is not what decides between them.

What decides is the solver. The learned head emits one coefficient per embedding channel plus a scalar, so the number of quantities the network has to predict grows with the width of the embedding, and every one of those coefficients enters the same reconstructed tensor. Inside the fixed-point iteration this proved impossible to stabilize: the coupled problem either diverged or returned fields outside the physical range under every numerical setting tried. The reading retained here is that predicting a larger number of quantities is harder to keep consistent when the input field is itself the output of the previous iterate, since the error on each predicted coefficient enters the same tensor and the errors accumulate rather than cancel. Predicting fewer quantities is what made the coupling possible. Figure 4.2 collects the whole construction in one diagram.

### 4.1.5 Training and results

The adopted network has 14 678 trainable parameters, of which 1 936 belong to the embedding and the remainder to the fitting network. As the local ones, it is trained on the three simulations at $\mathrm{Re} = 3\,900$, 11 000 and 60 000 and tested on the one at $\mathrm{Re} = 140\,000$, which is never seen during training, so that every figure reported below measures extrapolation in Reynolds number. Training uses Adam [64] with weight decay $10^{-6}$, an initial learning rate of $5 \times 10^{-3}$ decayed by a factor 0.7 every 100 epochs, batches of 4096 clouds, gradient-norm clipping at 1.0, and 1000 epochs. The composite loss of Section 3.1 is used together with the per-point weights, and the realizability terms are switched off for the reason given above. Ten per cent of the training clouds are held out for validation, and the reported network is the one minimizing the validation loss. Figure 4.3 shows the two curves.

Table 4.1 reports the accuracy on the held-out Reynolds number, using the metrics of Section 3.1.4, with the local tensor-basis closure of Table 3.4 alongside. As for the local closures, these are *a priori* metrics. They compare the predicted

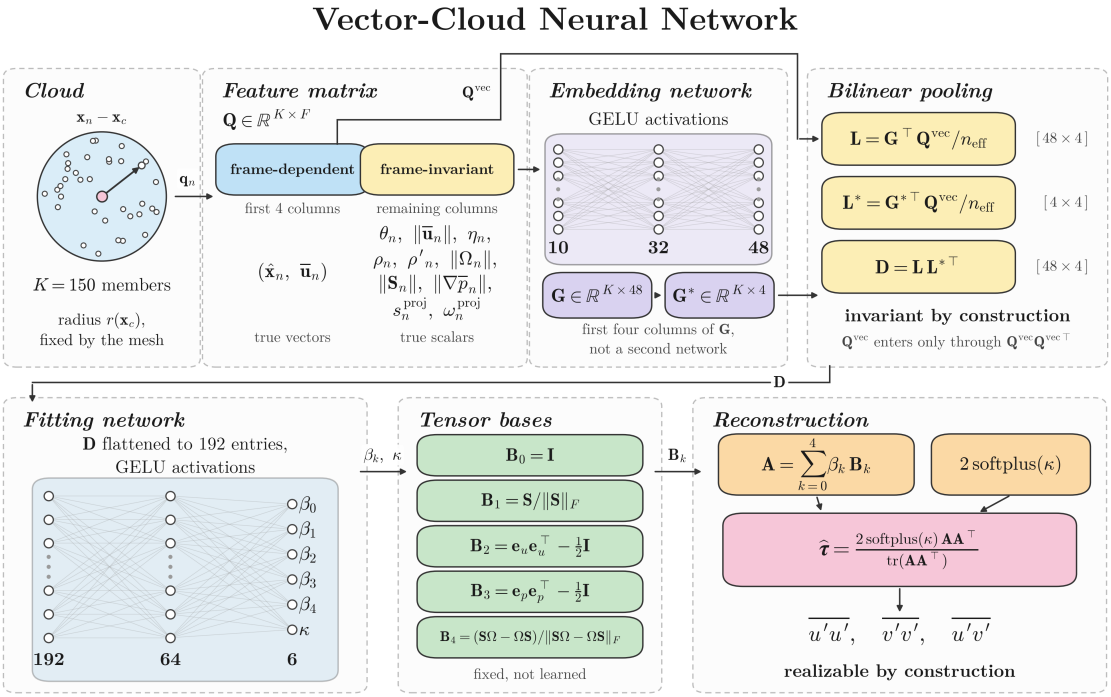



**Figure 4.2:** Architecture of the non-local closure. The cloud is described by a feature matrix whose first four columns carry true vectors and whose remaining columns carry frame invariants. Only the invariant block is embedded, and the two are recombined by a bilinear pooling whose entries depend on the cloud through pairwise inner products alone. The fitting network outputs the coefficients of a fixed tensor basis and a scale, from which the stress is reconstructed in a form that is realizable by construction.

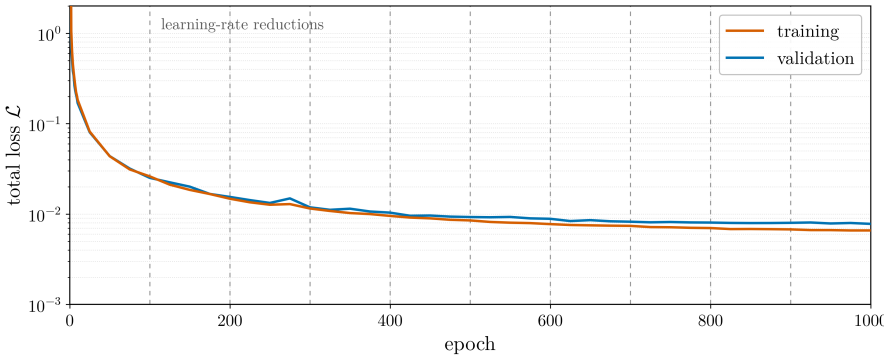



**Figure 4.3:** Training and validation loss of the non-local closure. The steps in both curves coincide with the scheduled reductions of the learning rate.

stress against the time-averaged reference data, and say nothing yet about the mean velocity obtained once the closure is solved for inside the FEM solver.

**Table 4.1:** VCNN closure on the held-out $\mathrm{Re} = 140\,000$ test set, with the local tensor-basis closure of Table 3.4 reported alongside on the two metrics the two share. Bold marks every entry on which the non-local closure is ahead.

| | $R^2$ | | $\varepsilon_{L^2}$ | | VCNN | |
|---|---|---|---|---|---|---|
| | VCNN | TBNN | VCNN | TBNN | RMSE | MAE |
| $\overline{u'u'}$ | **0.911** | 0.805 | **0.145** | 0.215 | 0.0240 | 0.0103 |
| $\overline{v'v'}$ | **0.975** | 0.944 | **0.086** | 0.116 | 0.0159 | 0.0075 |
| $\overline{u'v'}$ | **0.959** | 0.911 | **0.171** | 0.173 | 0.0056 | 0.0028 |
| $k$ | 0.954 | — | 0.087 | — | 0.0171 | 0.0085 |
| Aggregate | | | $\varepsilon_{L^2}^{\mathrm{tensor}} = 0.101$ | | | |

Every component clears the $R^2 \geq 0.8$ threshold of Section 3.1.4 with margin, and the non-local closure is ahead of the local tensor-basis closure on every component and on both metrics at once. This is the central *a priori* result of this chapter. The margin is not uniform. It is largest on $\overline{u'u'}$, the component the local closure predicts worst, where the coefficient of determination rises by more than a tenth and the relative error falls by a third. It is smallest on $\overline{u'v'}$, where the two are close on the relative error and separate on the coefficient of determination.

That ordering is the one the non-local hypothesis predicts. The streamwise normal stress in the wake is produced upstream and convected downstream, so its value at a point reflects the history of the flow along the streamline rather than the local strain rate, and it is the component a purely local closure is least equipped to represent. The shear stress, by contrast, is the component most directly tied to the local mean shear, which is exactly the quantity a local closure already has. Figure 4.4 shows the same comparison point by point.

The improvement is not bought with parameters. The local closures were also trained in wider variants, above thirty thousand parameters, and those variants are comparable to or worse than the smaller ones adopted in Chapter 3, so the local family does not improve when given more capacity. The closure used here has fewer parameters than the wide local variants and predicts better than all of them, so what separates it is the non-locality of its input and not the size of its network. Figure 4.5 shows where in the domain the margin lies.

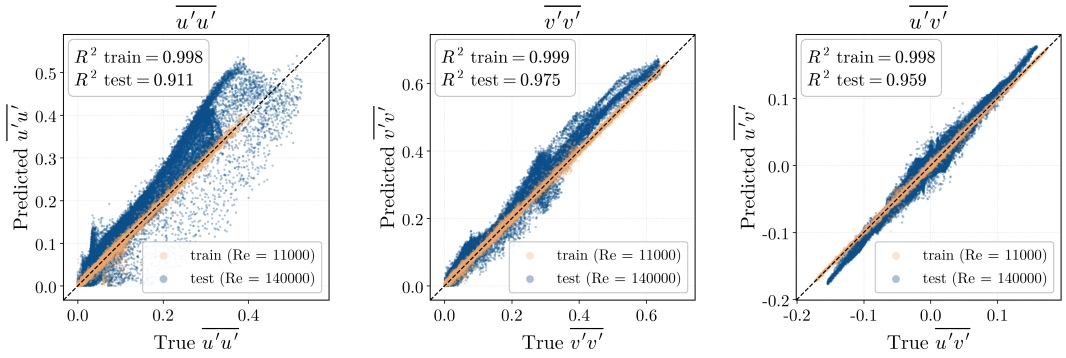



**Figure 4.4:** Non-local closure: predicted versus true Reynolds stress components, training (Re = 11 000) and held-out (Re = 140 000) test data, on the same axes as Figure 3.6. The scatter is visibly tighter than that of the local closures, and the margin is largest on the first component.

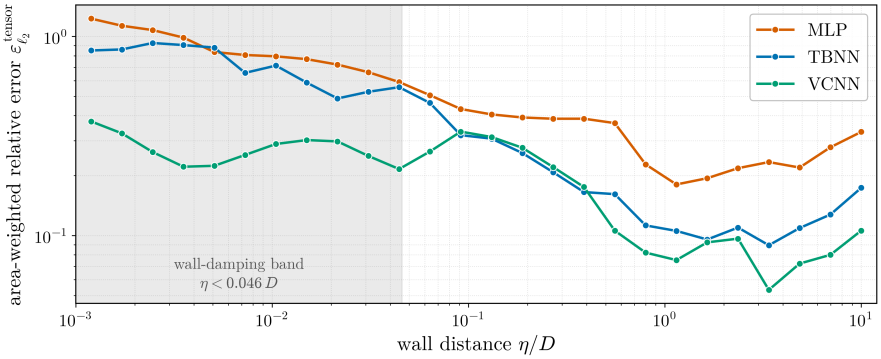



**Figure 4.5:** Area-weighted relative error against wall distance, for the closures compared in this work. The shaded band marks the near-wall region where the solver replaces the raw prediction by the damped one of Section 3.3.7.

## 4.2 Coupling with the solver

The non-local closure is embedded in the same solver, on the same mesh and at the same Reynolds number as the local closures of Chapter 3, under the same numerical configuration and with the same convergence criterion. Nothing about the discretization, the stabilization or the treatment of the stress is changed for this chapter. The comparison that follows therefore varies one thing, the closure, and everything the two chapters report can be read side by side.

What does change is the nature of the fixed-point problem. The stress at a cell now depends on the mean field over a neighbourhood of that cell rather than at the cell alone, so the map the Picard iteration is asked to converge reads the current

iterate at $K$ points for every cell instead of one. That difference is invisible in the *a priori* metrics of Table 4.1 and it dominates everything reported below.

### 4.2.1 The numerical setting

Everything reported in this section uses the following setting, held fixed.

- Closure: the network of Section 4.1, with the quadratic reconstruction of Equation (4.8), evaluated at every cell of the mesh at every iteration.
- Cloud: circular, $K = 150$ members, radius from the geometric law of Equation (4.11). The neighbour lists and the relative coordinates are built once before the Picard loop and are never rebuilt.
- Stabilization: the consistent streamline-upwind form of Equation (3.68), at the standard calibration $c_S = 1$, together with the grad-div term at the same scale used in Chapter 3.
- Stress treatment: the filtered implicit and explicit split of Equation (3.77), which is the form adopted in Chapter 3 for the same reason it is adopted here, that it is second order in the mesh size and carries no free coefficient.
- Filtering: the four configurations of Section 3.3.8, no filter, input only, output only, and both, are run and reported, since the choice is a property of the coupling and not of the closure.
- Iteration: under-relaxation $\alpha_{\text{closure}} = 0.3$ on the predicted stress, near-wall damping of Section 3.3.7, a floor on the eddy viscosity, and at most 150 Picard iterations with the convergence threshold of Section 3.4.1.

This is item by item the configuration adopted at the end of Chapter 3, so the coupling results of the local closures can be compared directly with the following.

### 4.2.2 Results

Table 4.2 reports the four filtering configurations, together with the local tensor-basis closure and the classical reference under the identical setting, taken from Table 3.16. Relative errors are on the evaluation window of Equation (3.67) and coefficients of determination over the whole domain, as in that table.

Four conclusions follow, and the first is about the filter rather than the closure.

The output filter is essential. Without it the velocity error is half again as large, the stress error rises by more than a half, and the coefficient of determination on the shear stress is negative, which means that the predicted field is further from the reference than a constant would be. With it the closure produces a usable field in

**Table 4.2:** *A posteriori* accuracy of the non-local closure at Re = 11 000 under the adopted configuration, with the local tensor-basis closure and the classical reference reported under the same setting. Relative $L^2$ errors are on the window of Equation (3.67), coefficients of determination over the whole domain.

| Closure | Metric | No filter | Input | Output | In+out |
|---|---|---|---|---|---|
| Non-local | $\delta u$ | 2.9e-2 | — | 1.3e-2 | 8.5e-3 |
| | $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | 0.089 | — | 0.059 | 0.061 |
| | $\varepsilon_{L^2}(\overline{u})$ | 0.100 | — | 0.078 | 0.074 |
| | $\varepsilon_{L^2}(\overline{v})$ | 0.538 | — | 0.463 | 0.400 |
| | $\varepsilon_{L^2}^{\text{tensor}}$ | 0.632 | — | 0.370 | 0.442 |
| | $R^2(\overline{u'v'})$ | $-0.79$ | — | 0.46 | 0.43 |
| Tensor-basis | $\delta u$ | *2.4e-1* | *3.0e-2* | *6.9e-3* | *1.7e-3* |
| | $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | *0.466* | *0.083* | *0.060* | *0.048* |
| | $\varepsilon_{L^2}(\overline{u})$ | *0.586* | *0.087* | *0.064* | *0.054* |
| | $\varepsilon_{L^2}(\overline{v})$ | *3.99* | *0.440* | *0.276* | *0.268* |
| | $\varepsilon_{L^2}^{\text{tensor}}$ | *2.58* | *0.393* | *0.327* | *0.402* |
| | $R^2(\overline{u'v'})$ | $-51.6$ | $-0.73$ | *0.24* | *0.57* |
| $k$-$\ell$ | $\delta u$ | *1.4e-5* | *1.3e-5* | *1.4e-5* | *1.3e-5* |
| | $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | *0.105* | *0.103* | *0.105* | *0.103* |
| | $\varepsilon_{L^2}(\overline{u})$ | *0.114* | *0.110* | *0.114* | *0.110* |
| | $\varepsilon_{L^2}(\overline{v})$ | *0.449* | *0.415* | *0.449* | *0.415* |
| | $\varepsilon_{L^2}^{\text{tensor}}$ | *0.521* | *0.640* | *0.521* | *0.640* |
| | $R^2(\overline{u'v'})$ | *0.46* | *0.50* | *0.46* | *0.50* |

both of the configurations that carry it. The input filter alone behaves differently from every other case in this work: the factorization of the momentum matrix fails and no field is produced at all. It is the only one of the four configurations, across all the formulations tried for this closure, that does not return a solution. However, this outcome was expected, because filtering an input that already reads spatial differences degrade those discrepancies between points of a neighbrood that the net learned to handle during training.

Against the classical reference the non-local closure wins, and by a wide margin on both quantities at once. It reaches a velocity error of 0.061 against 0.103, that is two fifths less, and 0.442 against 0.640 on the stress, that is a third less. This is the comparison that matters for the purpose of the chapter, since the one-equation model is what the closure is meant to replace.

Against the local tensor-basis closure it loses. The velocity error is 0.061 against 0.048 and the stress error 0.442 against 0.402, so the ordering of the *a priori* metrics of Table 4.1, where the non-local closure is ahead on every component, does not

survive the coupling. The *a priori* ranking and the *a posteriori* ranking are not the same ranking, and this is the second time in this work that they disagree; Figure 5.12 in Chapter 5 returns to the question comparing all the architectures with both errors measured at the same Reynolds number. Figure 4.7 shows where the two fields separate.

The iteration does not converge. The final update is $8.5 \times 10^{-3}$ with both filters and $1.3 \times 10^{-2}$ with the output filter alone, against a threshold three orders of magnitude smaller. The update does not decrease towards the end of the budget either: from roughly half of the iterations onward it oscillates within a fixed band instead of descending, so the iteration has reached a limit cycle and a larger budget would not move it. The local tensor-basis closure under the same setting stops at $1.7 \times 10^{-3}$, so it is a factor of five better and equally short of the threshold, while the classical reference converges. The fields reported above are therefore the fields at the end of the budget and not the fields at a fixed point. Therefore, one can ask if reaching better convergence, at least the one of the TBNN, the VCNN could have won on the *a posteriori* metrics, too. This motivates further exploration in both the net and the numerical setting. Some of those follows in the ablation studies of Section 4.4, but obtaining good results in the coupling with this architecture is experienced to be extremely difficult. It remains the main problem to face in future works. Figure 4.6 shows the update and the projection error along the iteration.

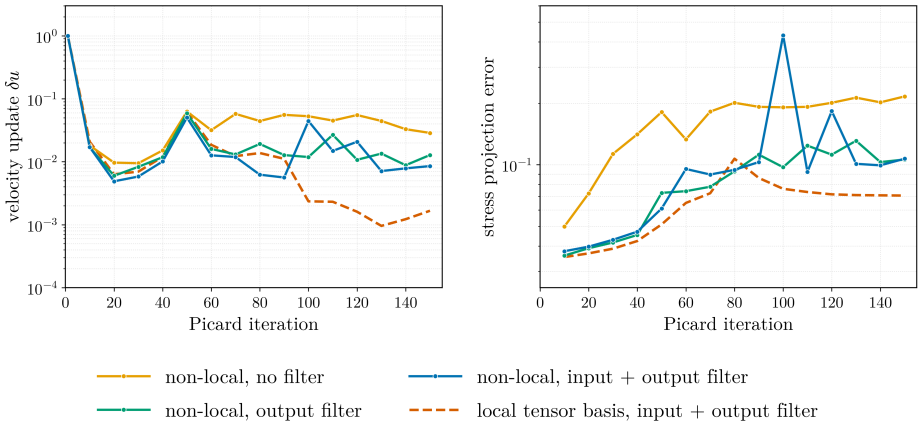



**Figure 4.6:** Picard history of the non-local closure under the adopted configuration. The update stops descending well before the end of the budget and settles into a band. The projection error of the stress drifts upward when only the output filter is applied and stays flat when both are.

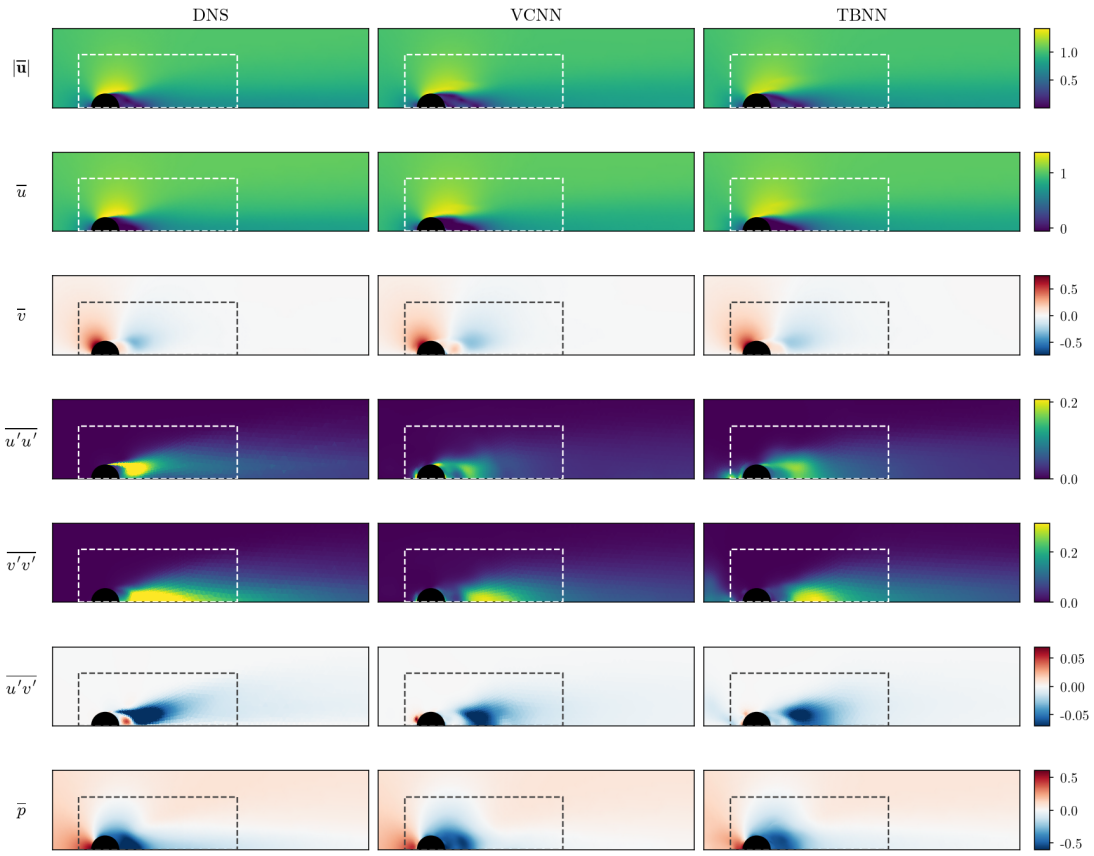



**Figure 4.7:** Mean fields at Re $= 11\,000$ under the adopted configuration, reference data against the non-local and the local tensor-basis closures. The velocity fields are close to each other and to the reference, and the two closures separate on the stress components.

### 4.2.3 Why the neighbourhood makes the coupling harder

The result that has to be explained is the one in the two paragraphs above: a closure that is ahead of the local one on every component of the stress when measured against reference data ends up behind it once the stress is solved for, and does not converge. The explanation proposed here is structural, and it is specific to cloud non-locality: it does not apply to the closures of Chapter 3.

A fixed-point solver feeds the closure its own previous iterate. In the first iterations that iterate is far from any physical field, and it is far from anything the network saw during training. A local closure evaluated on such a field receives one unphysical input per cell. The non-local closure receives $K$ of them, because the cloud gathers the same unconverged field at $K$ neighbouring points, and the stress

it returns at a cell is a pooled function of all of them. The probability that the input at a cell resembles anything in the training distribution therefore falls as the number of points the closure reads grows, and it falls fastest exactly where the field is furthest from the solution.

The same argument has a second form, on the side of the network rather than the input. The mapping the network has learned is constrained at $K$ points at once rather than at one, so the region of input space on which it returns a sensible stress is correspondingly narrower. A closure of this kind is selective: it behaves well on fields close to those it was trained on, and the iterates of a fixed-point solver are, by construction, far from those fields until the very end. This is a reading and not a measurement, and it is offered as such.

What the iteration does is consistent with both forms. The projection error of the predicted stress, that is the part of the prediction the discrete space cannot represent, sits at about a tenth for most of the iteration and jumps by a factor of four twice, at the two iterations where the update also jumps, before falling back. Those are the two moments at which the field entering the clouds leaves the range the network was trained on, and the iteration recovers from them but does not settle. It is also what the filters act on, which is consistent with the output filter being the ingredient no working configuration does without, and with the input filter, applied on its own, being the one that removes the solution entirely.

### 4.2.4 What the neighbourhood costs

The closure gathers $K$ neighbours for every cell at which it is evaluated, and it is evaluated at every cell at every Picard iteration. The cost of one evaluation is therefore proportional to $K$, and it is paid once per cell per iteration, which is the reason the cloud size is not chosen on accuracy alone. As Section 4.4 reports, sampling the same disc with more members is marginally more accurate, since it recovers the small loss introduced by the fixed geometric radius, but the cost grows in the same proportion, so $K = 150$ is retained as the compromise between the two.

The neighbourhood also complicates a distributed implementation, and this is the reason the solver used here runs on a single process. The gather is a nearest-neighbour query, so the cloud of a cell near a partition boundary reaches into cells that a different process owns, and a distributed run would have to exchange a halo whose width is set by the largest cloud radius rather than by the stencil of the discretization. The local closures of Chapter 3 have no such requirement. Building that exchange is the natural next step for a production implementation and it was not undertaken here.

### 4.2.5 How this configuration was reached

The setting reported above is the end point of a long search. The procedure was a cycle: train the network, measure it against reference data, put it in the solver, read what the solver produced, and then change either the architecture and retrain, or the numerical setting and solve again. Most of the settings tried failed outright, the iteration either leaving the physical range or settling on an error of the same order as the field it was predicting, and each failure had to be diagnosed on intermediate quantities before the next change could be chosen.

That cycle is expensive, because one turn of it costs a training and a set of solver runs, and the two are not interchangeable: a change that improves the *a priori* metrics need not improve the coupling, and the two results of this chapter that disagree are precisely of that kind.

## 4.3 Symbolic Regression for radius length

The result of the previous section depends on one parameter more than on any other. The radius of the cloud sets how much of the surrounding flow the network sees, and with it how much of the non-locality it can actually use. It is also the parameter that had to be changed for the coupling to work at all, from an expression depending on the flow to one depending on the mesh, for the reason given in Section 4.2. This section derives that expression.

The cloud radius is the one parameter of the construction that is neither learned nor derived. The formulation this work started from set it from the local flow state,

$$r_{\mathrm{old}}(\mathbf{x}) = \mathrm{clip}\left( \|\overline{\mathbf{u}}\| \, \frac{c_S + c_\Omega}{\|\mathbf{S}\| + \|\mathbf{\Omega}\| + c_0},\ r_{\min},\ r_{\max} \right), \tag{4.10}$$

with hand-calibrated constants. The physical reading of Equation (4.10) is a length of advection. The reciprocal of $\|\mathbf{S}\| + \|\mathbf{\Omega}\|$ is a local deformation time scale, and multiplying it by the local velocity magnitude gives the distance a fluid particle travels before the mean-flow gradients change appreciably. Bounding a turbulence time scale by the reciprocal of the strain rate in this way is a device used in the realizability bounds of classical two-equation models.

Equation (4.10) has one property that is acceptable during training and problematic at deployment. It depends on the solution. Inside the FEM solver of Section 3.4 the velocity and its gradients change at every Picard iteration, so the cloud would have to be rebuilt at every iteration, and a nearest-neighbour search over the whole mesh is by far the most expensive operation in the closure evaluation. More importantly, the region of influence of a point would then depend on the current iterate, so the closure would no longer be a fixed function of the mesh and

the fixed-point map the Picard iteration relies on would change from one iteration to the next.

The aim of this section is therefore to replace Equation (4.10) by an expression that reproduces its behaviour but depends on the geometry alone.

### 4.3.1 Symbolic regression

Symbolic regression searches for a closed-form algebraic expression that fits a dataset, treating both the structure of the expression and its constants as unknowns. Unlike a regression over a fixed functional form, the space being searched is the set of expression trees built from a chosen set of operators and input variables. The search is combinatorial and is carried out here by genetic programming, as implemented in PySR [65]. Populations of candidate expressions are evolved by mutation and crossover, and each candidate is scored on both its accuracy and its complexity, the latter measured by the number of nodes in its tree.

The output is not a single expression but a Pareto front. For each complexity level the search returns the most accurate expression found at that complexity, so that the trade-off between fit and interpretability is exposed rather than resolved automatically. Selecting a member of the front is a modelling decision, and it is made explicitly below.

### 4.3.2 Dataset and search setup

The target of the regression is Equation (4.10) itself, re-clipped to $[0.05D,\ 0.5D]$. The upper bound was raised from the original value so that the largest admissible cloud has a diameter equal to the cylinder diameter. The regression therefore distils an existing expression into a surrogate that uses a restricted set of inputs, rather than fitting the radius to an independently defined ground truth. Since the radius spans more than an order of magnitude, the regression is performed on its logarithm, which makes the fit relative rather than absolute and prevents the large far-field radii from dominating the objective.

The candidate inputs are restricted to three, and this restriction is the point of the exercise.

- The wall distance, measured from the cylinder surface.
- The angular position around the cylinder.
- The Reynolds number, rescaled by $10^3$.

The local flow quantities appearing in Equation (4.10) are deliberately excluded, so that any expression the search returns is computable from the mesh alone.

Seed points are drawn from the reference simulations with a sampling density weighted towards the near-wall and wake regions, and snapped to actual data points. The search uses the binary operators $+$, $-$, $\times$, $\div$ and exponentiation, and the unary operators $\sqrt{\cdot}$, exp, log, sin, cos, tanh and squaring, with 100 iterations, 30 populations of 33 individuals each, and a maximum expression size of 30 nodes. Two Reynolds numbers are used for the search and the remaining two are held out.

### 4.3.3 Selection from the Pareto front

Table 4.3 reports the front returned by the search, restricted to the complexity levels where the loss changes appreciably, and Figure 4.8 shows it.

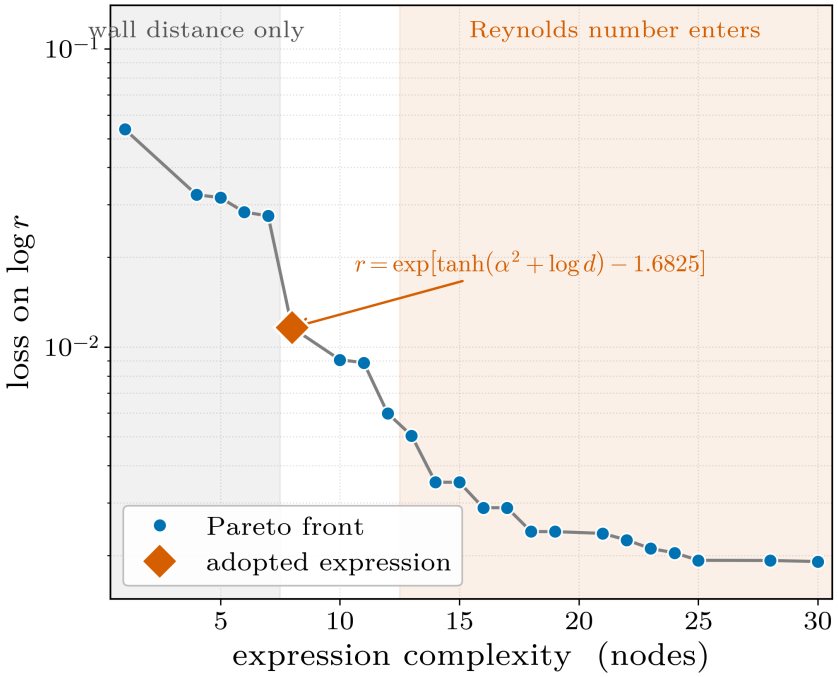



**Figure 4.8:** Pareto front of the symbolic regression. The selected expression sits at the largest single drop in loss along the front, before the search begins to introduce the Reynolds number and to fit the four available simulations individually.

Three features of the front drive the selection. The loss falls by a factor 2.4 between complexity seven and complexity eight, the largest single improvement anywhere on the front, and the corresponding expression attains the highest score assigned by the search over all complexity levels. Below complexity eight the angular position does not appear at all, and every such expression is a monotone function of the wall distance alone. The simplest of them cannot produce a radius below $0.18D$, and therefore removes exactly the adaptivity near the wall that

**Table 4.3:** Pareto front of the symbolic regression for the cloud radius. The loss is the mean-squared error on $\log r$, and complexity is the number of nodes in the expression tree. Here $d$ is the wall distance and $\alpha$ the angular position.

| Complexity | Expression | Loss |
|---|---|---|
| 1 | const | 0.0537 |
| 4 | $\tanh(d) - 1.700$ | 0.0324 |
| 5 | $\tanh(d^2) - 1.700$ | 0.0317 |
| 6 | $-0.909/(\tanh(d) + 0.298)$ | 0.0283 |
| 7 | $2.211\sqrt{\tanh(d)} - 2.903$ | 0.0275 |
| **8** | $\mathbf{\tanh(\alpha^2 + \log d) - 1.682}$ | **0.0116** |
| 10 | $\tanh\Big(1.919(\alpha^2 + \log d)\Big) - 1.698$ | 0.0091 |
| 12 | $\tanh\Big(\log(d^2(-\alpha^2 - 0.812)^4)\Big) - 1.693$ | 0.0060 |

motivated a variable radius in the first place. Above complexity eight the gains become marginal, and from complexity thirteen onwards the expressions begin to introduce the Reynolds number inside trigonometric functions, which is the signature of fitting four discrete Reynolds numbers rather than a trend.

The expression at complexity eight is therefore adopted,

$$r(\mathbf{x}) = \exp\Big[\tanh\Big(\alpha^2 + \log d\Big) - 1.6825\Big], \tag{4.11}$$

with $d$ and $\alpha$ in units of the cylinder diameter. The Reynolds number does not appear, so the same expression serves all four simulations.

Equation (4.11) carries a single free constant, an additive offset inside the exponential, which acts as a multiplicative scale on the radius. Everything else is structure. The saturation of the hyperbolic tangent bounds the result without any explicit clipping, and the resulting range is $[0.068D,\ 0.505D]$, confirmed on the actual meshes. Figure 4.9 shows the field it produces and the one it was distilled from. The logarithm of the wall distance inside the tangent makes the radius grow smoothly away from the wall and collapse close to it, which is the behaviour Equation (4.10) produced through the strain rate. The quadratic dependence on the angular position makes the cloud largest in the wake and smallest on the upstream face, where the boundary layer is thin and attached.

Replacing Equation (4.10) by Equation (4.11) is not for free. What it costs, and what enlarging the cloud recovers of it, is reported with the other neighbourhood comparisons in Section 4.4.

What is gained in exchange is structural rather than numerical. The region of influence of every point becomes a property of the mesh, fixed before the solver starts

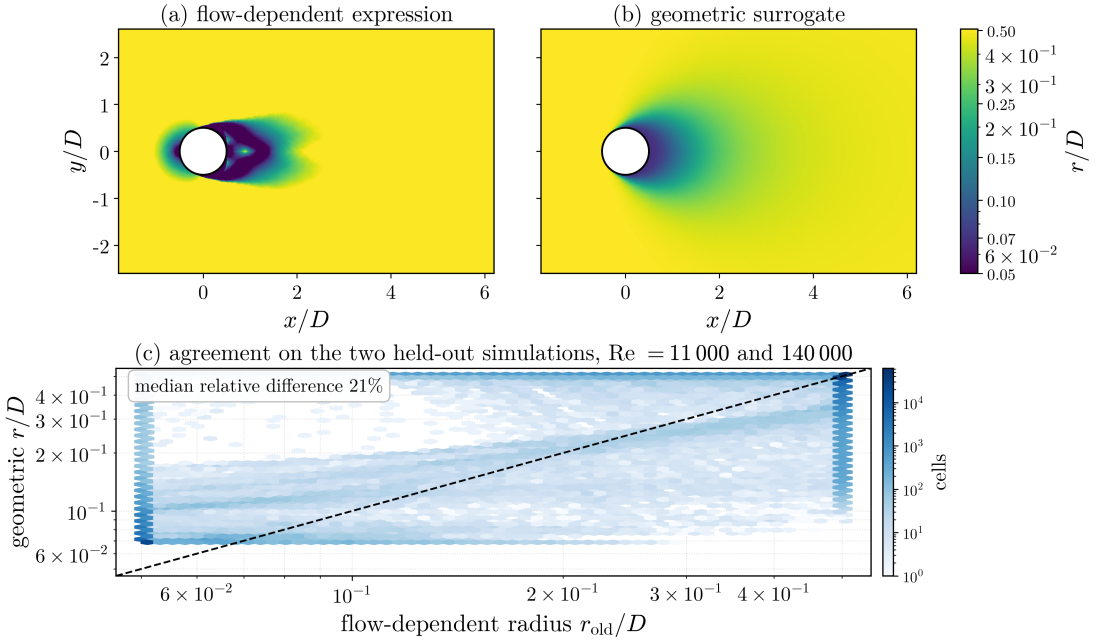



**Figure 4.9:** Cloud radius from the original flow-dependent expression and from the geometric surrogate obtained by symbolic regression. The surrogate reproduces the spatial structure of the target without using any flow quantity, and is therefore computable once per mesh.

and unchanged by it, so the closure remains a fixed function of the discretization throughout the fixed-point iteration.

## 4.4 Ablations

The comparisons collected here are grouped by which step of the construction they vary, in the same order in which the chapter introduced them: what the closure reads, how it rebuilds the stress from what it has read, how it is trained, and what the solver does to it once it is coupled. Each comparison states what it isolates and what it does not, because several of them vary more than one ingredient at a time.

### 4.4.1 What the closure reads

**Standardizing the frame-dependent block.**

The two vector pairs of Equation (4.1) can be standardized in two ways. Treating their four columns as four independent scalars, which is what the invariant components receive, subtracts a different mean from each and divides each by a different deviation. Dividing each pair by the single scalar $\sqrt{\mathbb{E}[\|\cdot\|^2]}$ and subtracting no mean treats them as the vectors they are.

The difference is not cosmetic, and it can be measured on the trained network without retraining it. Rotating the frame and recomputing the pooled descriptor of Equation (4.7), which the argument of Section 4.1 says should not move at all, produces a relative drift of order unity under the per-column treatment and of order $10^{-15}$ under the other, that is machine precision. The per-column treatment subtracts a fixed vector and applies an anisotropic scaling, and both single out the axes of the frame the training data happened to be written in, so the invariance the whole chapter rests on is lost at the first operation applied to the input.

Retraining shows that nothing is paid for restoring it. The two networks reach $R^2 = 0.911/0.975/0.959$ under the per-column treatment and $0.920/0.969/0.964$ under the isotropic one, and the second is ahead on the aggregate relative error. The exact treatment is therefore adopted. This comparison was run only on the VCNN introduced in this chapter, so it establishes the direction of the effect for this network and not for every variant of it. Different results may be expected since not all of the following restore 100% equivariance.

**The shape and the size of the neighbourhood.**

The reference formulation collects an elliptical neighbourhood whose semi-axes are set by the local flow state, following the same principle this work started from. The cloud used here is circular and its radius is a function of the mesh alone, for the reason given in Section 4.2.

Table 4.4 collects what is available on the two remaining choices, the law that sets the radius and the number of members. It must be read with one caveat stated first. The comparison between the flow-dependent radius and the geometric one exists only on a different feature set, the one that does not enforce invariance under reflections but only under rotation. This other variant exists because adding the signed vorticity and the anisotropy component of the Strain tensor between the features improves net performance at the cost of losing reflection invariance. The two radii are therefore compared with each other under a setting that differs from the adopted on. While the fixed geometric version is compared varying $K$ under the network setting actually used. For this reason, the table as a whole is not a single-variable sweep and no row of it should be read against a row of a different

block.

**Table 4.4:** Neighbourhood ablations on the held-out Re $= 140\,000$ test set. Within each block a single ingredient varies. The two blocks use different feature sets and are not to be compared row by row.

| Feature set | Radius | $K$ | $R^2$ ($\overline{u'u'}/\overline{v'v'}/\overline{u'v'}$) |
|---|---|---|---|
| Invariant, adopted | Geometric | 150 | 0.911 / 0.975 / 0.959 |
| Invariant, adopted | Geometric | 300 | 0.920 / 0.972 / 0.957 |
| Unconstrained, earlier | Flow-dependent | 150 | 0.943 / 0.977 / 0.953 |
| Unconstrained, earlier | Geometric | 150 | 0.921 / 0.973 / 0.956 |
| Unconstrained, earlier | Geometric | 300 | 0.947 / 0.972 / 0.953 |

Three considerations follow. The geometric radius does not cost accuracy, and enlarging the cloud recovers whatever small difference remains. The larger cloud is not adopted, for the reason given in Section 4.2: the cost of gathering the neighbourhood is proportional to the cloud size and is paid at every cell and at every iteration of the solver, so doubling $K$ doubles the most expensive operation of the closure in exchange for a difference of a few thousandths in $R^2$. And within the adopted feature set the cloud size makes no difference worth having either, which is the same conclusion reached independently.

**Enforcing invariance under reflections.**

The best feature set tried carries the mean vorticity and one component of the strain rate as raw columns of the frame-dependent block. Both change sign under a reflection, so that the network is invariant under rotations but not under reflections, and Section 4.1 replaced them by the two invariant projections of Equations (4.4) and (4.5).

The constraint is not for free, and the direction of what it costs is informative. At either cloud size the unconstrained network is ahead on $\overline{u'u'}$, by a hundredth at $K = 150$ and by almost three hundredths at $K = 300$, where it reaches the highest value of that component recorded anywhere in this work. It is behind on $\overline{u'v'}$ at both sizes. The observation is that the raw pseudoscalars carry the orientation information the network can use, and that the projections recover most but not all of it, at the price of the equivariance property that then holds exactly rather than approximately. The trade is accepted here because a closure whose prediction depends on the orientation of the frame it was written in is not considered to be a reliable closure.

### 4.4.2 How the stress is rebuilt

Three heads were implemented on the same network, at parity in features, cloud, cloud size, embedding and fitting widths, activation, learning rate, number of epochs and loss. The adopted one normalizes the trace, Equation (4.8). The second bounds the anisotropic part against the isotropic one instead,

$$\sigma = \mathrm{softplus}(\beta_0), \qquad \mathbf{a} = \sum_{k=1}^{4} \beta_k \mathbf{B}_k, \qquad \widehat{\boldsymbol{\tau}} = \sigma \mathbf{I} + \sigma \tanh\left(\frac{\|\mathbf{a}\|_2}{\sigma}\right) \frac{\mathbf{a}}{\|\mathbf{a}\|_2}, \quad (4.12)$$

where $\|\cdot\|_2$ is the spectral norm, which for a symmetric traceless two-dimensional tensor is the modulus of either eigenvalue. Since the hyperbolic tangent is strictly smaller than one for any finite argument, both eigenvalues of the anisotropic part are strictly smaller than $\sigma$, so this form is admissible by construction as well. The third is the learned equivariant reconstruction of the reference formulation,

$$\widetilde{\mathbf{X}} = \frac{\mathbf{G}^\top \mathbf{X}}{n_{\mathrm{eff}}} \in \mathbb{R}^{48\times 2}, \qquad \widehat{\boldsymbol{\tau}} = \widetilde{\mathbf{X}}^\top \mathrm{diag}(\hat{\mathbf{e}})\, \widetilde{\mathbf{X}} + \hat{\gamma}\, \mathbf{I}, \quad (4.13)$$

where $\mathbf{X}$ collects the relative position vectors of the cloud members and the coefficients $\hat{\mathbf{e}}$ and $\hat{\gamma}$ come from the fitting network. The tensor structure is learned from data rather than prescribed. Passing the coefficients through a softplus makes the result a sum of two positive semidefinite terms and restores realizability without touching equivariance, since they are elementwise functions of already-invariant quantities while $\widetilde{\mathbf{X}}$, the only equivariant object, is left intact.

**Table 4.5:** The three reconstructions at parity in every other respect, held-out $\mathrm{Re} = 140\,000$ test set.

| Reconstruction | Parameters | $R^2$ ($\overline{u'u'}$/$\overline{v'v'}$/$\overline{u'v'}$) | $\varepsilon_{L^2}^{\mathrm{tensor}}$ |
|---|---|---|---|
| Trace-normalized, adopted, Eq. (4.8) | 14 678 | 0.911 / 0.975 / 0.959 | 0.101 |
| Capped anisotropy, Eq. (4.12) | 14 613 | 0.926 / 0.970 / 0.960 | 0.095 |
| Learned embedding, Eq. (4.13) | 17 473 | 0.912 / 0.955 / 0.938 | — |

All three reconstructions are good, and they are close. Table 4.5 separates them by at most a hundredth and a half on any component, which is of the order of the spread between seeds. The capped form is marginally the most accurate of the three and the adopted one is marginally behind it. On *a priori* grounds alone there would be no reason to prefer one over another. This is a point for the non locality argument because the gain in the performance with respect local architectures does not depend on the reconstruction. It comes entirely from spatial information.

The choice was not made on *a priori* grounds. The capped checkpoint does not work inside the solver, and the learned one could not be stabilized there at

all, for the reason given in Section 4.1. The last part of this section reports what the solver does to the first two, and it is the comparison that actually decided the architecture of this chapter.

The three differ in what they constrain, which is the reason they behave differently once coupled. Equation (4.8) fixes the trace through Equation (4.9) and lets the network set only its magnitude, so the anisotropy is entirely determined by the direction of $\mathbf{A}$. Equation (4.12) leaves the isotropic scale and the anisotropic direction as separate degrees of freedom and constrains only their ratio. Equation (4.13) constrains neither, and emits one coefficient per embedding channel rather than a fixed handful.

### 4.4.3 How the network is trained

**The composite loss.**

The composite loss of Section 3.1 was compared against a plain unweighted mean-squared error at otherwise identical settings. The composite loss reaches $R^2 = 0.911/0.975/0.959$ against $0.929/0.950/0.912$ for the plain error. The pattern matches the one already observed for the smallest local closure in Section 3.2: the plain error is better on $\overline{u'u'}$ alone and clearly worse on the two components that carry the anisotropy and the shear sign, which are precisely the ones the composite terms constrain.

**The activation.**

GELU is used in place of a rectified linear unit because the latter left a fraction of the embedding units permanently inactive on this input distribution. The classic dead neurons phenomena that easily happens with ReLU. The one comparison available at parity, on a wider variant of the architecture, does not show an accuracy benefit, so the change is reported as what it was, a fix for inactive units rather than a validated improvement.

### 4.4.4 The role of the solver

The last group is the one that decided the architecture, and it varies two things: the form in which the stress is fed to the momentum equation and the stabilization that accompanies it. Three formulations were run, each with the four filtering configurations of Section 3.3.8, on the deployed checkpoint. The first uses the exact implicit and explicit split of Equation (3.43) with the isotropic diffusion of Equation (3.70). The second replaces the exact split by the filtered one of Equation (3.77), keeping the same stabilization. The third keeps the filtered split

and replaces the stabilization by the consistent form of Equation (3.68), which is the configuration adopted in Section 4.2 and in Chapter 3.

**Table 4.6:** The non-local closure under three formulations of the coupling, held at Re $= 11\,000$. The input filter alone returns no solution in any of the three. Relative $L^2$ errors are on the window of Equation (3.67).

| Formulation | Metric | No filter | Output | In+out |
|---|---|---|---|---|
| Exact split, isotropic diffusion | $\delta u$ | 9.9e-3 | 5.1e-4 | 9.3e-3 |
| | $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | 0.094 | 0.089 | 0.091 |
| | $\varepsilon_{L^2}^{\text{tensor}}$ | 0.642 | 0.372 | 0.408 |
| Filtered split, isotropic diffusion | $\delta u$ | 1.2e-2 | 1.7e-3 | **9.5e-6** |
| | $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | 0.092 | 0.081 | 0.071 |
| | $\varepsilon_{L^2}^{\text{tensor}}$ | 0.599 | 0.384 | 0.366 |
| Filtered split, consistent stabilization | $\delta u$ | 2.9e-2 | 1.3e-2 | 8.5e-3 |
| | $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | 0.089 | **0.059** | 0.061 |
| | $\varepsilon_{L^2}^{\text{tensor}}$ | 0.632 | 0.370 | 0.442 |

Table 4.6 reproduces, on a non-local closure, the trade-off that Chapter 3 established on local ones, and it is worth stating plainly because the two chapters reached it independently.

The one configuration that converges is the one built from a stabilization that is not consistent. With the filtered split and isotropic diffusion the iteration reaches the threshold and stops of its own accord after 123 iterations, at a final update three orders of magnitude below anything the consistent form achieves. It is also, of the three, the one whose predicted stress is closest to the reference. However, it is a formulation far to be a standard one. The term it adds does not vanish on the exact solution, so the field it converges to is not the solution of the unstabilized problem, and the shift it introduces is neither measured nor guaranteed to be in the right direction.

The most accurate velocity field in terms of error comes from the consistent stabilization, at 0.059 with the output filter and 0.061 with both, against 0.071 for the best of the non-consistent runs. That form does not fully converge, and it is nonetheless the one adopted, for the reason given in Chapter 3: an almost converged field obtained from a formulation that leaves the fixed point where it belongs is preferred to a converged field obtained from one that moves it.

Replacing the exact split by the filtered one is what separates the first block from the second, it is a single-variable change. It improves every column it touches: the update falls from $9.3 \times 10^{-3}$ to $9.5 \times 10^{-6}$ with both filters, the velocity error from 0.091 to 0.071 and the stress error from 0.408 to 0.366. The filtered split was introduced in Chapter 3 on local closures and carries over to this one unchanged.

**The best reconstruction does not work in the solver.**

The capped head of Equation (4.12) is marginally the most accurate of the three in Table 4.5. Run through the same solver, under both the formulations of the second and third blocks above and under all four filtering configurations, it never produces a usable field: the velocity error on the window stays between 0.41 and 0.82, that is of the same order as the field it is predicting, the predicted stress is orders of magnitude away from the reference, and the smallest final update reached across the eight runs, one of which returns no field at all, is $9.0 \times 10^{-2}$, two orders of magnitude worse than the adopted head under the same settings. This are the numbers behind the choice declared along the chapter. It is the reason why the deployed checkpoint is the trace-normalized one, and the whole of the cost of that decision is the difference between the first two rows of Table 4.5, which is a few thousandths of $R^2$.

It is also the sharpest instance in this work of a result that both this chapter and the previous one keep running into. *A priori* accuracy, measured against reference data on a held-out Reynolds number, does not predict what a closure does once the equations are solved with it. Two heads that differ by less than the spread between seeds on every *a priori* metric differ by two orders of magnitude in the solver, and the ordering reverses. Any selection made on *a priori* metrics alone, in this class of models, is a selection made on the wrong quantity.

# Chapter 5

# Non-Local closure for RANS: GNN

The closure of Chapter 4 reads a neighbourhood rather than a point, and defines that neighbourhood as a disc of prescribed physical radius. This chapter keeps the first half of that choice and replaces the second. A graph closure also reads a neighbourhood, but it defines it as a number of message-passing rounds on the connectivity of the mesh, so its region of influence is measured in cells rather than in diameters.

That single difference organizes the whole chapter. Measuring range in cells is what makes the architecture cheap, because a round of message passing is a sparse product on edges that already exist and costs nothing to build. It is also what makes it fragile, because the connectivity is a property of one mesh and does not travel with the trained weights. A closure trained on one mesh and deployed on another keeps its number of hops and loses the physical distance those hops used to span. Everything reported below, from the way the node state is built to the size of the gap between accuracy measured against reference data and accuracy measured inside the solver, follows from that.

Two architectures are described, and what separates them is how they reach far. The first propagates information one cell at a time and pays one round per cell of range. The second adds a coarse level, so that distant nodes are connected directly and range is bought at a cost that does not grow with distance. Section 5.1 describes both, together with the accuracy they reach against reference data. Section 5.2 embeds them in the FEM solver of Section 3.4, under the configuration adopted in Chapter 3 and used again in Chapter 4, so that the three chapters can be read side by side. Section 5.3 returns to range itself, which is the design problem specific to this family, and measures what it costs and what the coarse level buys. Section 5.4 collects the comparisons behind each choice.

## 5.1 Graph closures

### 5.1.1 The graph

The graph is the mesh of the reference simulation, taken as it is. Nodes are the mesh nodes and edges are the sides of the quadrilateral elements. Each element contributes its four sides, self-loops are discarded, duplicate sides shared by adjacent elements are merged, and every remaining undirected edge is stored in both directions so that messages can travel either way. The resulting graphs range from about $10^5$ to $2 \times 10^5$ nodes and have a mean degree of 3.50 on all four reference meshes, so the connectivity a node sees is the same regardless of which simulation it comes from.

Nothing in this construction is a hyperparameter. There is no radius, no neighbour count and no sampling, and the region a node can see after a fixed number of message-passing rounds is determined by the mesh alone. The price is that the graph carries the resolution of the mesh into the model, so that two points which are physically close but separated by many small cells are far apart on the graph. Section 5.1.5 is the response to that. Figure 5.1 shows the graph on one of the reference meshes.

Mean-flow gradients are reconstructed at the nodes by a local least-squares fit over the one-hop neighbours,

$$\mathbf{G}_i = \arg\min_{\mathbf{G}} \sum_{j \in \mathcal{N}(i)} \left\| \mathbf{G}\,(\mathbf{x}_j - \mathbf{x}_i) - (f_j - f_i) \right\|^2, \tag{5.1}$$

from which the strain rate, the vorticity and the pressure gradient follow.

### 5.1.2 Node state and the equivariant vector channel

Each node carries two kinds of state, separated exactly as in Section 4.1, and for the same reason.

The invariant part is a vector of five scalars,

$$\mathbf{h}_i = \Big( \|\overline{\mathbf{u}}_i\|,\ \eta_i,\ \|\boldsymbol{\Omega}_i\|,\ \|\mathbf{S}_i\|,\ \|\nabla \overline{p}_i\| \Big), \tag{5.2}$$

compressed with a logarithm on the four heavily skewed entries and then standardized. These are the same five invariants that feed the coefficient network of the local tensor-basis closure of Section 3.2, which makes the two architectures comparable on their inputs. They are passed to the first layer raw, with no learned embedding in front of them, so the width of the invariant state is fixed at five.

The equivariant part is a two-dimensional vector $\mathbf{v}_i$, initialized from the mean velocity at the node itself. It follows the equivariant graph network layer of Satorras et al. [37], adapted to a mesh that does not move, so that node positions are never

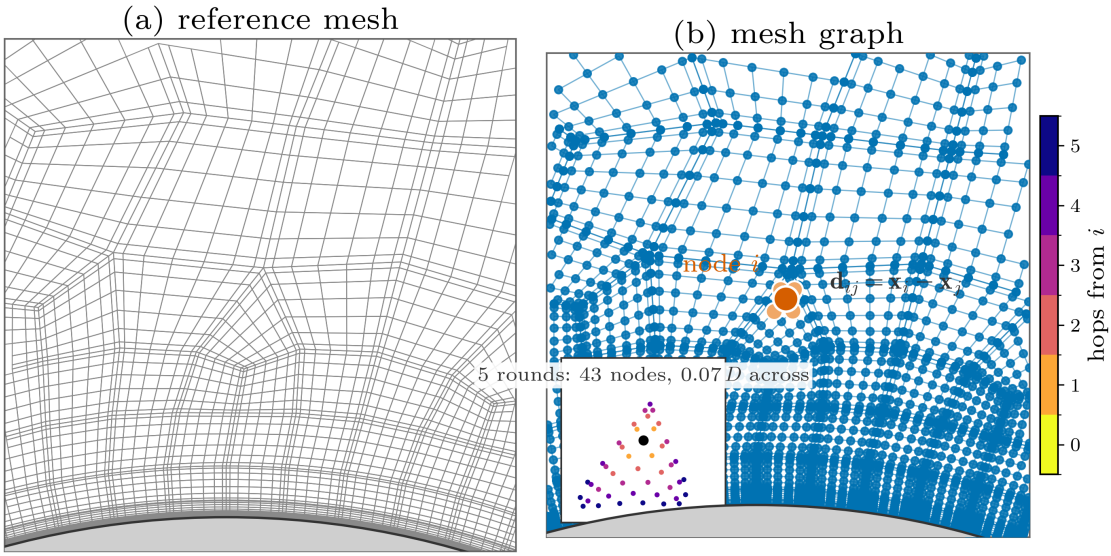



**Figure 5.1:** From the reference mesh to the graph used by the closure. Nodes are mesh nodes and edges are element sides. The inset shows the set of nodes reachable within the number of message-passing rounds used here, which is the region of influence of the highlighted node.

updated and only $\mathbf{v}_i$ evolves. Writing $\mathbf{d}_{ij} = \mathbf{x}_i - \mathbf{x}_j$ for the fixed relative position of a sender $j$ with respect to a receiver $i$, one round of message passing reads

$$\delta_{ij} = \frac{\|\mathbf{d}_{ij}\|^2}{\left\langle \|\mathbf{d}_{ik}\|^2 \right\rangle_{k\in\mathcal{N}(i)}}, \tag{5.3}$$

$$\mathbf{m}_{ij} = \phi_e\Big(\mathbf{h}_i,\ \mathbf{h}_j,\ \delta_{ij},\ \mathbf{v}_i \cdot \mathbf{v}_j\Big), \tag{5.4}$$

$$a_{ij} = \underset{j\in\mathcal{N}(i)}{\mathrm{softmax}}\ \phi_a\left(\mathbf{m}_{ij},\ \frac{\mathbf{d}_{ij}\cdot\mathbf{v}_i}{\|\mathbf{d}_{ij}\|\,\|\mathbf{v}_i\|}\right), \tag{5.5}$$

$$\mathbf{v}_i \leftarrow \mathbf{v}_i + \sum_{j\in\mathcal{N}(i)} a_{ij}\,\phi_v(\mathbf{m}_{ij})\,\mathbf{d}_{ij}, \tag{5.6}$$

$$\mathbf{h}_i \leftarrow \phi_h\Big(\mathbf{h}_i,\ \sum_{j\in\mathcal{N}(i)} a_{ij}\,\mathbf{m}_{ij}\Big), \tag{5.7}$$

with $\phi_e$, $\phi_a$, $\phi_v$ and $\phi_h$ small multilayer perceptrons with Sigmoid Linear Unit (SiLU) activations, each round carrying its own weights.

Equations (5.3)–(5.7) are equivariant under any orthogonal change of frame, and the argument is short enough to give in full. Every argument of $\phi_e$ is an invariant scalar, since $\|\mathbf{d}_{ij}\|$ is a length and $\mathbf{v}_i \cdot \mathbf{v}_j$ an inner product, so $\mathbf{m}_{ij}$ is invariant. The alignment fed to $\phi_a$ is a normalized inner product between two vectors and is

therefore invariant as well, so $a_{ij}$ and $\phi_v(\mathbf{m}_{ij})$ are invariant scalars. In Equation (5.6) the only vector-valued object is $\mathbf{d}_{ij}$, multiplied by invariant scalars, so $\mathbf{v}_i$ transforms exactly as $\mathbf{d}_{ij}$ does and remains equivariant round after round. Equation (5.7) combines invariants, so $\mathbf{h}_i$ stays invariant. The construction therefore respects rotations and reflections alike, by the same reasoning that supports Equation (4.7) for the vector cloud.

Two ingredients of Equations (5.3)–(5.5) deserve a word. The normalization in Equation (5.3) divides the squared edge length by its mean over the edges incident on the receiver. Without it the edge feature would span orders of magnitude across the domain, since the mesh is graded and wall cells are far smaller than far-field cells, and a single learned function would have to cover that whole range. The alignment entering Equation (5.5) is the cosine of the angle between the edge direction and the locally accumulated flow direction, and is positive when the sender lies upstream of the receiver. It is not used as a fixed weight. It is one input of a learned logit, so the network decides how much upstream information is worth relative to downstream information, and the resulting attention weights sum to one over the neighbours of each node. This replaces a plain sum, whose magnitude would grow with the degree of the receiver and would therefore encode mesh resolution rather than physics.

The vector channel is internal. It is never read after the last round, and the prediction is built from $\mathbf{h}_i$ alone. Its role is to carry into the messages a quantity a scalar-only network cannot express, namely how the locally accumulated flow directions of two nodes relate to each other and to the edge joining them.

### 5.1.3 Readout

After the last round, the invariant node state is decoded by a fitting network of three hidden layers with ReLU activations into five coefficients and one scale. These are combined with the same five tensor bases $\mathbf{B}_0$ to $\mathbf{B}_4$ used in Section 4.1, built here from the raw mean-flow quantities of the node itself. The stress is reconstructed in the capped form of Equation (4.12), which bounds the anisotropic part against the isotropic one: the first coefficient sets a non-negative isotropic scale through a softplus, the remaining four assemble the anisotropic part, and a hyperbolic tangent keeps its two eigenvalues strictly below that scale. The reconstruction is therefore realizable for every coefficient the network can produce. Since the coefficients are invariant and the bases transform as tensors, the reconstructed stress is equivariant, and the equivariance of the message passing established above carries through to the output. Figure 5.2 collects the whole construction in one diagram.

The trace-normalized form of Equation (4.8), which is the one adopted for the vector-cloud closure, was implemented on this architecture as well. The two graph architectures adopt different ones, the capped form described above on the

single-scale architecture and the trace-normalized form on the multiscale variant of Section 5.1.5. On the single-scale architecture the two are close on reference data and separate inside the solver, in the opposite direction to the one they separate in Chapter 4. The comparison on both architectures is reported in Section 5.4, together with a training instability that the choice of head interacts with.

Targets are normalized by a Mahalanobis whitening rather than by a per-component z-score. Writing $\mathbf{\Sigma}$ for the covariance of the three stress components over the training nodes and $\mathbf{\Sigma} = \mathbf{L}\mathbf{L}^\top$ for its Cholesky factorization, the normalized target is

$$\tilde{\mathbf{y}} = \mathbf{L}^{-1}\big(\mathbf{y} - \boldsymbol{\mu}\big). \tag{5.8}$$

A per-component z-score divides by the standard deviations alone and leaves the components correlated, so a network that improves one component is implicitly penalized on another. Equation (5.8) maps the three components onto uncorrelated coordinates of unit variance, and the regression term of the loss is evaluated there. On this dataset the two normal stresses are strongly correlated while the shear stress is very nearly uncorrelated with both, so the whitening mostly acts by removing the correlation between $\overline{u'u'}$ and $\overline{v'v'}$. This kind of normalization is introduced for this architecture only, after having observed systematic improvement of only one component at a time during hyperparameter search.

### 5.1.4 Data augmentation

Only three simulations are available for training, so the closure sees three flow states. Two augmentations enlarge that set, and they are different in kind.

The first re-discretizes the same physical field. A local element-size field is extracted from the original mesh, a new quadrilateral mesh is generated from that field, and the reference velocity, pressure and stress fields are interpolated onto it. The physics is unchanged and the discretization is not, so this augmentation targets robustness to the mesh rather than coverage of new flow states. The resulting meshes are coarser than the originals, with about two thirds of the nodes.

The second exploits a symmetry of the governing equations. Reflecting the domain about the vertical axis alone is not a symmetry here, because the domain is not symmetric in the streamwise direction and the inlet and the outlet carry different kinds of boundary condition. Reflecting the domain and reversing the flow direction together is. Writing $\tilde{x} = -x$, $\tilde{u} = -u$, $\tilde{v} = v$ and $\tilde{p} = p$, and using $\partial_{\tilde{x}} = -\partial_x$ throughout, every term of the steady incompressible equations reproduces itself. Continuity gives $\partial_{\tilde{x}}\tilde{u} + \partial_y\tilde{v} = \partial_x u + \partial_y v = 0$, and both momentum components have their left and right sides multiplied by $-1$ simultaneously, so the transformed fields solve the same equations. The boundary conditions transform consistently as well. The uniform inflow at the upstream boundary becomes a uniform inflow of the same speed entering from the opposite side, the no-slip

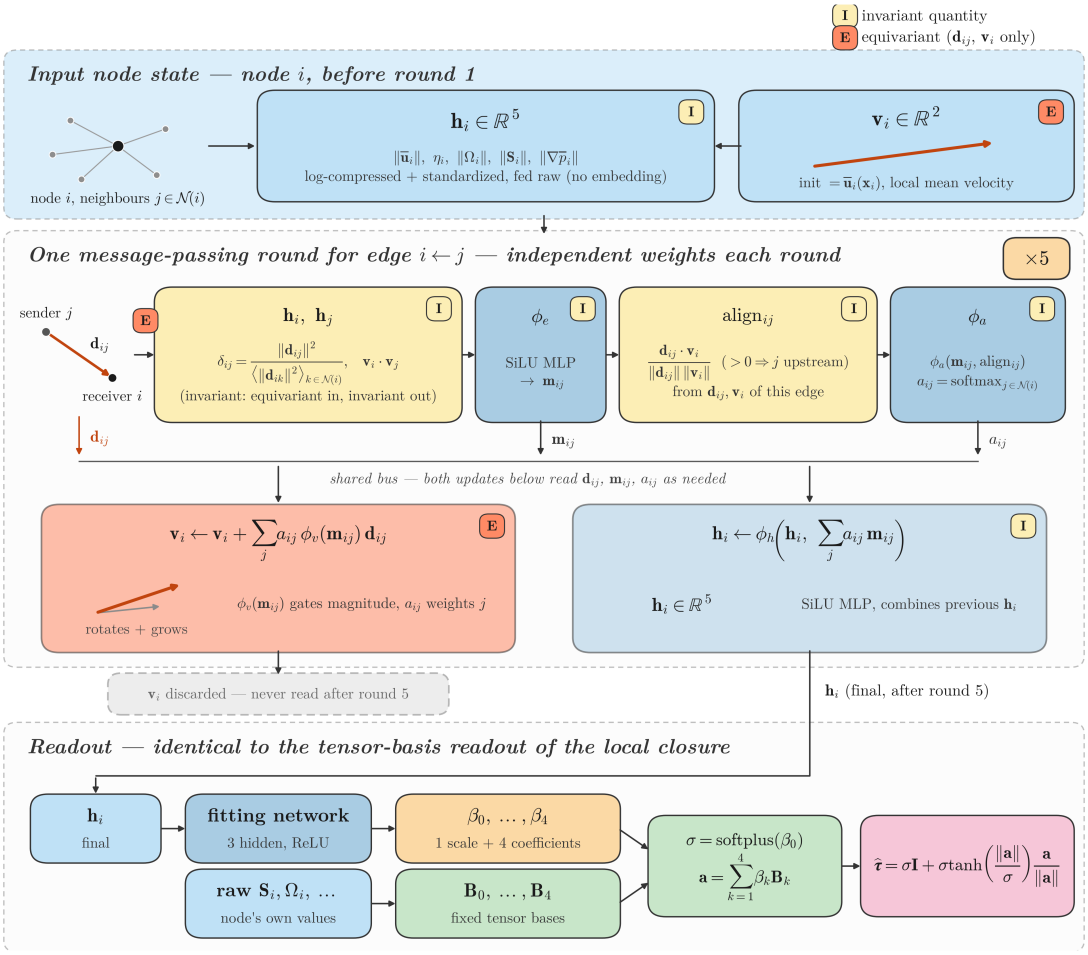



**Figure 5.2:** GNN architecture. Each node carries an invariant scalar state and an equivariant vector channel. Messages are built from invariants alone, and the only vector-valued quantity entering the vector update is the fixed relative position of the edge, which is what makes the update equivariant. The vector channel is discarded before the readout, and the stress is reconstructed from the invariant state through a fixed tensor basis.

condition on the cylinder is unchanged, and the cylinder is symmetric under the reflection by construction.

The transformation is applied to the raw fields alone, and the derived quantities are recomputed from them through Equation (5.1), so the following sign pattern is reproduced rather than imposed,

$$\begin{aligned} &\text{unchanged:} \quad \overline{u'u'},\ \overline{v'v'},\ S_{xx},\ \partial_y\overline{p},\ \overline{v},\ \overline{p}, \\ &\text{sign reversed:} \quad \overline{u'v'},\ S_{xy},\ \partial_x\overline{p},\ \overline{u},\ \omega. \end{aligned} \tag{5.9}$$

Applying both augmentations to the three training simulations yields twelve graphs, shown in Figure 5.3.

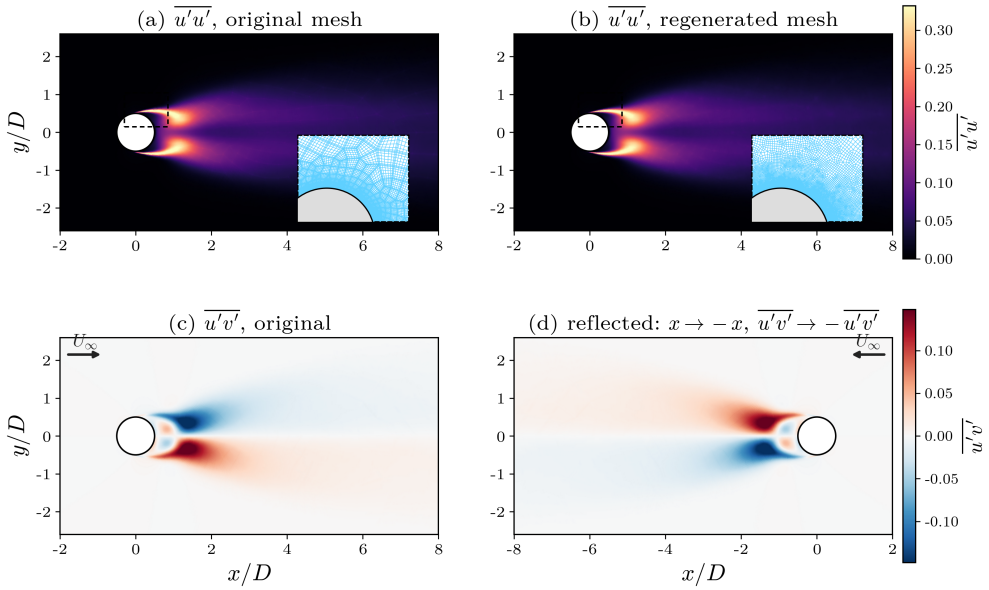



**Figure 5.3:** The two augmentations. Re-meshing changes the discretization and leaves the field unchanged. Reflection with flow reversal maps the field onto a solution of the same equations, with the shear stress changing sign.

### 5.1.5 Reaching far: the multiscale variant

The architecture described so far pays one round of message passing per cell of range. Reaching a point $L$ away therefore costs $L/h$ rounds, and every round adds parameters, memory and latency. On the meshes of this work a range of one cylinder diameter is of the order of a hundred cells in the near wake, so the range that the vector-cloud closure of Chapter 4 obtains by drawing a disc is not reachable by hops at any practical depth. Moreover, it is well known that GNNs are severely affected by oversmoothing effect. That is, the variance of the output predicted by the net hardly decrease as the number of hops increases. The problem is also described in [36], where they introduced a Multiscale Graph Neural Network to overcome it.

The multiscale variant buys range differently. A second, much sparser graph is built on a subset of the nodes, so that one hop on it spans many cells of the fine

one, and information travels between distant regions in a fixed number of rounds rather than in a number proportional to the distance. The fine graph keeps the local detail, the coarse graph carries the range, and the two are coupled in both directions once per cycle.

Those two couplings are the operations the reference formulation [36] calls *downsampling* and *upsampling*, and they are not the restriction and prolongation of a multigrid cycle. Instead of fixed operators chosen once from the geometry, they are message-passing layers with their own weights, trained end to end with the rest of the network, so what is sent up and what is brought back is learned rather than prescribed. They are also not symmetric. Downsampling writes onto a coarse state that is still empty, so the coarse level is built entirely out of the fine one and holds nothing of its own. Upsampling writes onto fine nodes that already carry the state produced before the excursion, so what returns from the coarse level is combined with the local description instead of replacing it. Figure 5.4 shows the two transfers and the path a node state follows through one cycle. The rest of this section builds the construction in that order: first the coarse graph and the links between the levels, then the sequence in which rounds and transfers are applied.

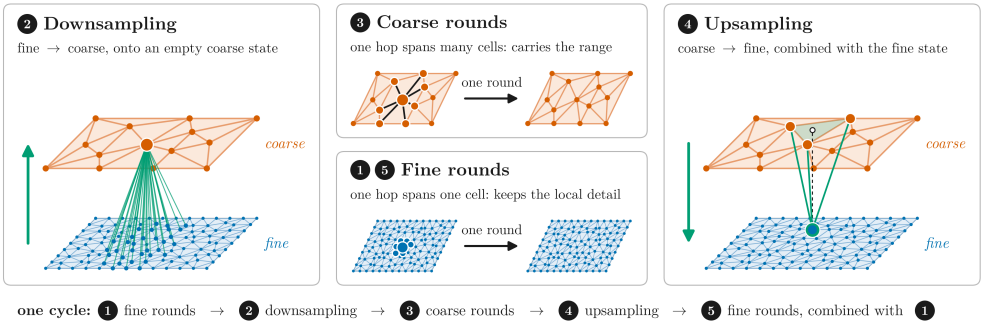



**Figure 5.4:** The two transfers between levels and the path a node state follows through one cycle; the layout of the scheme follows Fortunato et al. [36], the meshes are schematic. The downsampling carries the fine node states onto the coarse level, the coarse rounds move information across the domain, and the upsampling returns it to the fine level, where it is combined with the state the fine rounds had already produced. The numbered badges give the order of the operations within one cycle.

### The coarse level

A second graph is built once per mesh, from the node positions alone.

- $M = 500$ nodes are selected from the fine mesh by farthest-point sampling. Starting from one node, the node maximizing the distance to the set already

chosen is added repeatedly. The coarse nodes are therefore a genuine subset of the fine nodes rather than synthesized points, so relative positions between the two levels remain exact and the equivariance argument of Section 5.1 carries over unchanged.

- The coarse edges are the sides of the Delaunay triangulation of the coarse nodes, stored in both directions. This yields about 2 900 directed edges, so the coarse level is smaller than the fine one by two orders of magnitude and its cost is negligible.
- Two bipartite graphs connect the levels. Each fine node is linked to the three vertices of the coarse triangle containing it, and to its three nearest coarse nodes when it falls outside the coarse hull. The downward graph carries messages from fine to coarse and the upward graph is its transpose.

The ratio $M/N$ is between two and seven parts per thousand across the meshes used, so the coarse level is deliberately far coarser than any mesh on which this flow could be solved. Figure 5.5 shows the coarse level it produces.

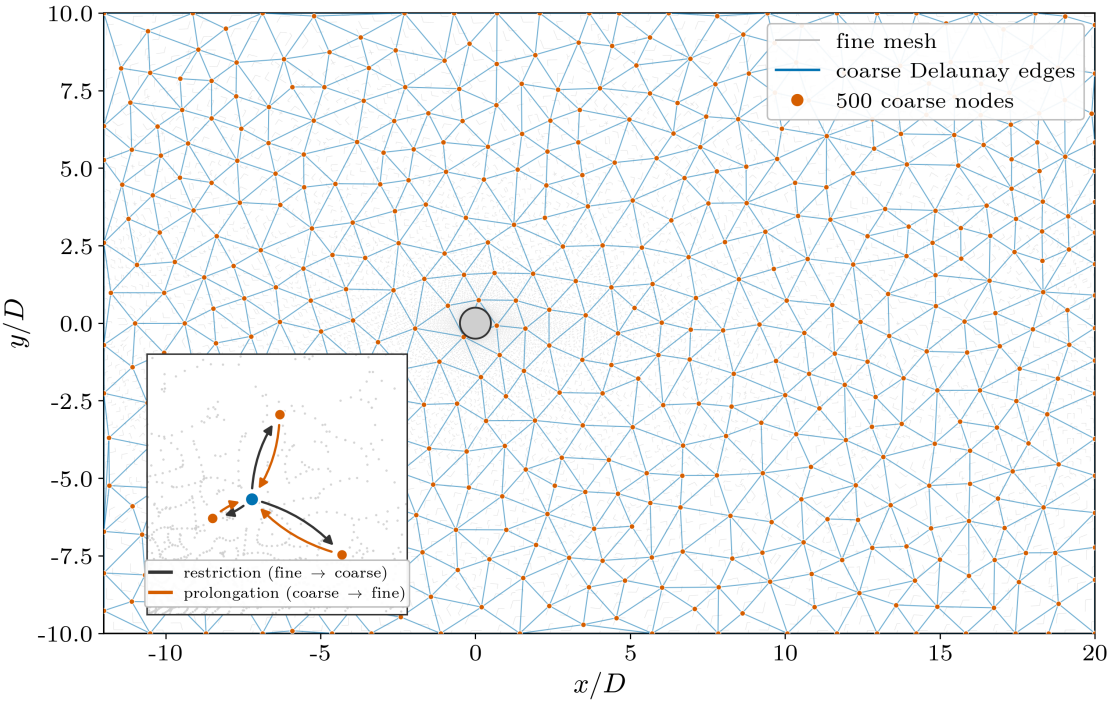



**Figure 5.5:** The auxiliary coarse level. Coarse nodes are a subset of the fine mesh nodes chosen by farthest-point sampling, connected by a Delaunay triangulation. The inset shows the bipartite links that carry messages between the two levels.

### The cycle

The forward pass performs one downward and one upward traversal, arranged like a multigrid V-cycle,

$$\begin{aligned} &1 \text{ round on } G^h \longrightarrow \text{ downsampling } \longrightarrow 11 \text{ rounds on } G^l \\ &\longrightarrow \text{ upsampling } \longrightarrow 1 \text{ round on } G^h. \end{aligned} \tag{5.10}$$

The downsampling and the upsampling are the same message-passing layer of Equations (5.3)–(5.7), applied to a bipartite graph with distinct sender and receiver sets, so that only the receiving side is updated. The coarse state is zero-initialized and is populated exclusively by the downsampling, so no physical quantity is measured on the coarse level.

The budget is redistributed rather than added. The two fine rounds cost what two rounds of the architecture of Section 5.1 cost, the two bipartite layers cost the same order again, and the eleven coarse rounds are almost free because the coarse graph has two orders of magnitude fewer edges. What the cycle buys is reach. Information leaves any fine node, crosses the coarse level in eleven hops of a graph whose edges span a substantial fraction of the domain, and returns, so the region of influence of a node is no longer bounded by a small number of fine mesh cells.

One further difference from Section 5.1 is required rather than optional. The vector update of Equation (5.6) is additive and unbounded, and Equation (5.10) applies fifteen layers in sequence against the five of the single-scale architecture. Accumulating fifteen unbounded additions drives the magnitude of the vector channel beyond the range of single-precision arithmetic, and gradient clipping does not prevent it, since clipping bounds the update to the weights and not the magnitude of the activations. The vector channel is therefore renormalized to unit length after every update,

$$\mathbf{v}_i \leftarrow \frac{\mathbf{v}_i + \Delta\mathbf{v}_i}{\|\mathbf{v}_i + \Delta\mathbf{v}_i\|}. \tag{5.11}$$

Equivariance is preserved, because the denominator is a rotation-invariant scalar built from the vector itself, so numerator and denominator transform consistently. The direction of the vector channel is retained and only its magnitude is discarded, which is consistent with its role as a direction along which upstream neighbours are identified. This is a genuine second difference between the two architectures, and the reconstruction head is a third, since the two adopt different ones. The comparison between them in Table 5.1 therefore does not isolate the multiscale topology alone. Figure 5.6 shows the cycle.

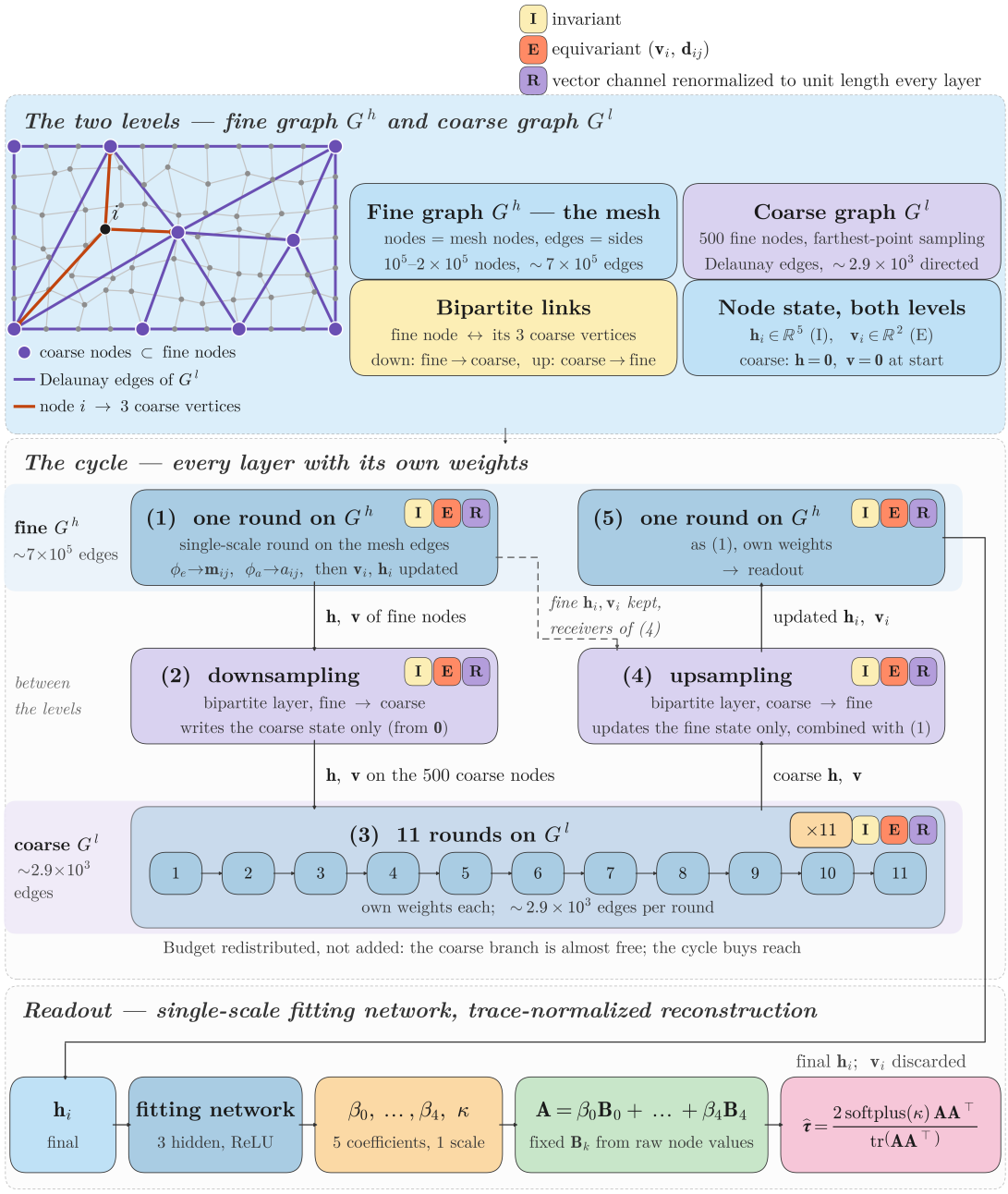



**Figure 5.6:** MS-GNN forward pass. A single round on the fine graph is followed by a downsampling onto the coarse level, eleven rounds there, an upsampling back, and a final fine round. The coarse branch carries almost no cost, since the coarse graph has two orders of magnitude fewer edges than the fine one.

### 5.1.6 Training and results

Both architectures are trained on the twelve graphs built from the three simulations at Re $= 3\,900$, $11\,000$ and $60\,000$ together with their reflected and augmented copies, and are tested on the graph of the simulation at Re $= 140\,000$. The single-scale network uses five message-passing rounds with a hidden width of 16 in every message network, an invariant state of width five, and a fitting network of three hidden layers of width 16, for a total of $8\,280$ parameters, of which $7\,555$ belong to the five rounds. The multiscale network keeps node features, activations, readout, target normalization and augmentation identical, adopts the trace-normalized reconstruction rather than the capped one for the reason given in Section 5.4, and reaches $23\,407$ parameters over fifteen layer instances and the shared readout. Training uses Adam [64] with weight decay $10^{-6}$, an initial learning rate of $10^{-2}$ on the single-scale network and $5 \times 10^{-3}$ on the multiscale one, decayed by a factor 0.7 every 100 epochs, gradient clipping at 1.0, and 1200 epochs. Each graph is processed whole, without minibatching, since the message passing needs the full connectivity. Figure 5.7 and Figure 5.8 show the two training histories.

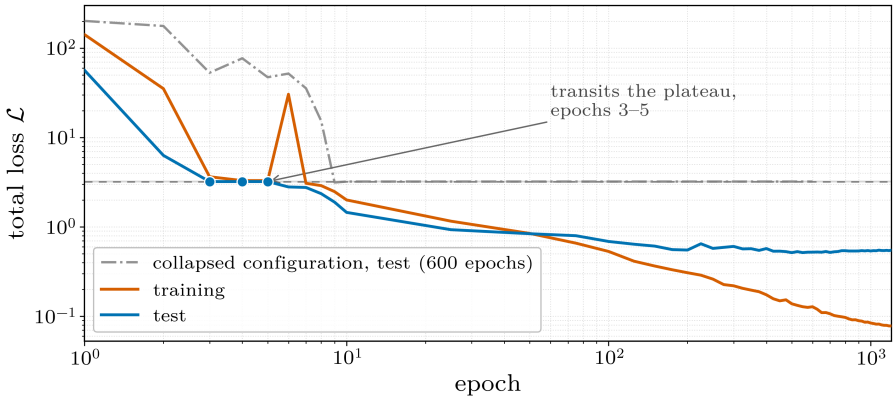



**Figure 5.7:** GNN: training and test loss history. The dashed level marks the local minimum discussed later in this section, which the reported training passes through in its first epochs before escaping.

Table 5.1 reports both networks on the held-out Reynolds number, with the local tensor-basis closure of Chapter 3 and the vector-cloud closure of Chapter 4 alongside. All four rows are the networks that are chosen as the best inside the solver later in this chapter and in the previous ones.

Three readings follow, and the first is the one that matters most for what comes after.

The two graph closures do not sit on the same side of the local tensor-basis closure on reference data. The single-scale one is behind it on the streamwise normal stress and on the shear stress and matches it on the cross-stream one; the

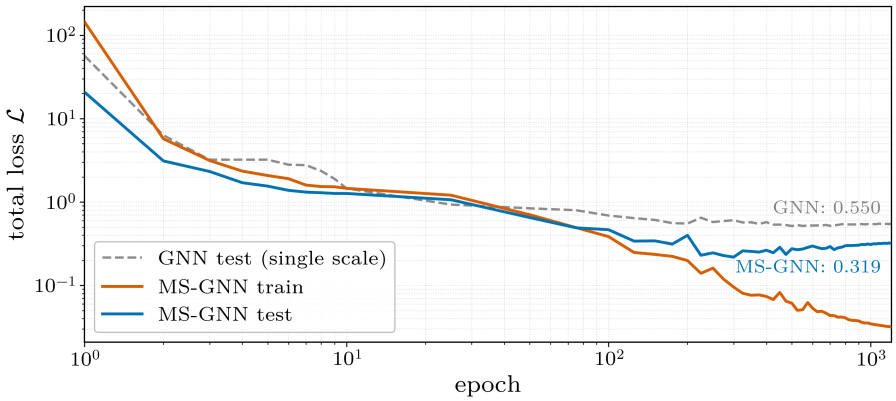



**Figure 5.8:** MS-GNN: training and test loss history, with the test loss of the single-scale architecture shown for comparison.

**Table 5.1:** The two graph closures on the held-out Re $= 140\,000$ test set, with the local tensor-basis closure and the vector-cloud closure reported alongside. Every row is the network deployed in the solver. Relative $L^2$ errors are over the whole domain.

| Closure | $R^2$ $(\overline{u'u'}/\overline{v'v'}/\overline{u'v'})$ | $\varepsilon_{L^2}$ $(\overline{u'u'}/\overline{v'v'}/\overline{u'v'})$ | $R_k^2$ | $\varepsilon_{L^2}^{\text{tensor}}$ |
|---|---|---|---|---|
| Graph, single scale | 0.773 / 0.944 / 0.893 | 0.209 / 0.118 / 0.205 | 0.899 | 0.140 |
| Graph, multiscale | 0.926 / 0.945 / 0.925 | 0.181 / 0.143 / 0.225 | 0.949 | 0.152 |
| Vector cloud | 0.911 / 0.975 / 0.959 | 0.145 / 0.086 / 0.171 | 0.954 | 0.101 |
| Tensor basis, local | 0.805 / 0.944 / 0.911 | 0.215 / 0.116 / 0.173 | 0.916 | 0.140 |

multiscale one is ahead on the streamwise normal stress and on the shear stress and level with it on the cross-stream one. This proves the real need of a multiscale graph since the small amount of neighborhood information given by a simple GNN is not enough to do better because the amount of non local information entering the net is physically to close to the point in which the prediction is made. To get a real gain, the net needs real physical spatial inputs.

The vector-cloud closure is ahead of both on every relative error, and its aggregate is 0.101 against 0.140 and 0.152, about thirty per cent smaller than both, although the multiscale closure is ahead of it on the coefficient of determination for the streamwise normal stress. The vector cloud gathers a hundred and fifty points at a prescribed physical radius and describes them by a pooled invariant; a five-hop graph reaches five cells and describes them by repeated local averaging. On these meshes the second is a much smaller region, and Section 5.3 measures how much smaller. Figure 5.9 and Figure 5.10 show the two comparisons point by point, and

Figure 5.11 shows where in the domain the two graph closures differ from each other.

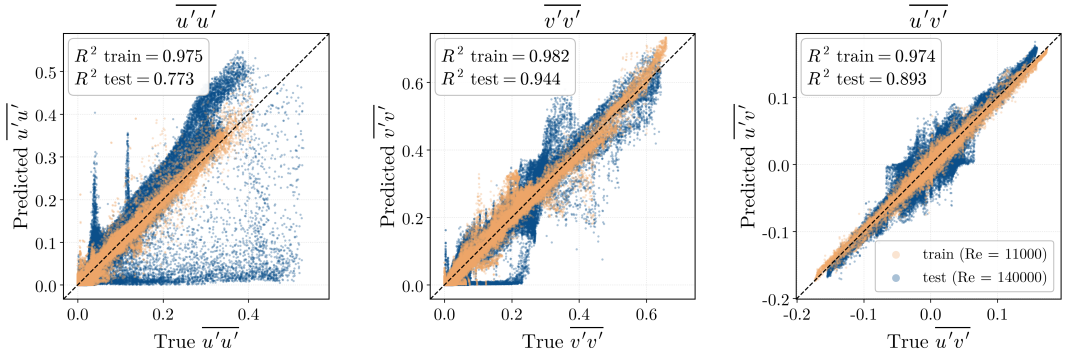



**Figure 5.9:** GNN: predicted versus true Reynolds stress components, training and held-out test data; the legend of the third panel applies to all three. The streamwise normal stress is visibly more scattered than the other two components.

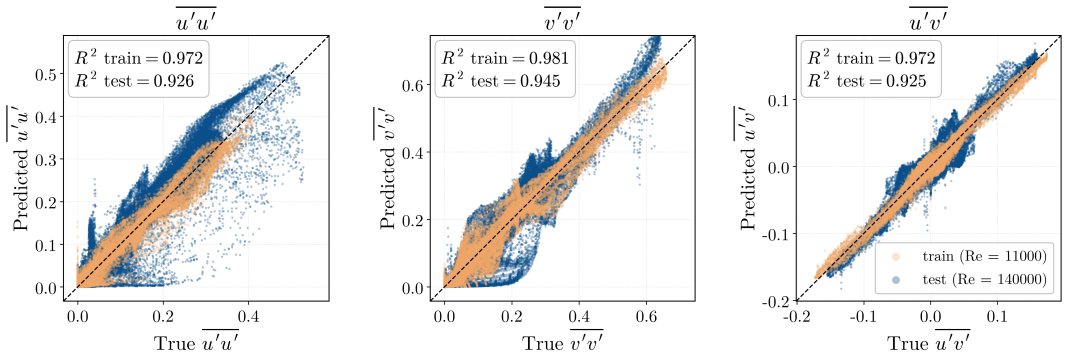



**Figure 5.10:** MS-GNN: predicted versus true Reynolds stress components, training and held-out test data; the legend of the third panel applies to all three. The scatter on the streamwise normal stress is markedly tighter than in Figure 5.9.

The two GNN variants do not totally agree on which is the more accurate here, and the answer depends on the measure. The multiscale one is ahead by fifteen hundredths on the streamwise normal stress, the component that depends most on upstream history, and by three on the shear stress, while the two are level on the cross-stream one. On relative error it is ahead on that same component and behind on the other two, and its aggregate is marginally larger, 0.152 against 0.140, for almost three times the parameters. What the coarse level buys is therefore already visible on reference data but not on every measure, and the rest of the chapter is where it shows most clearly.

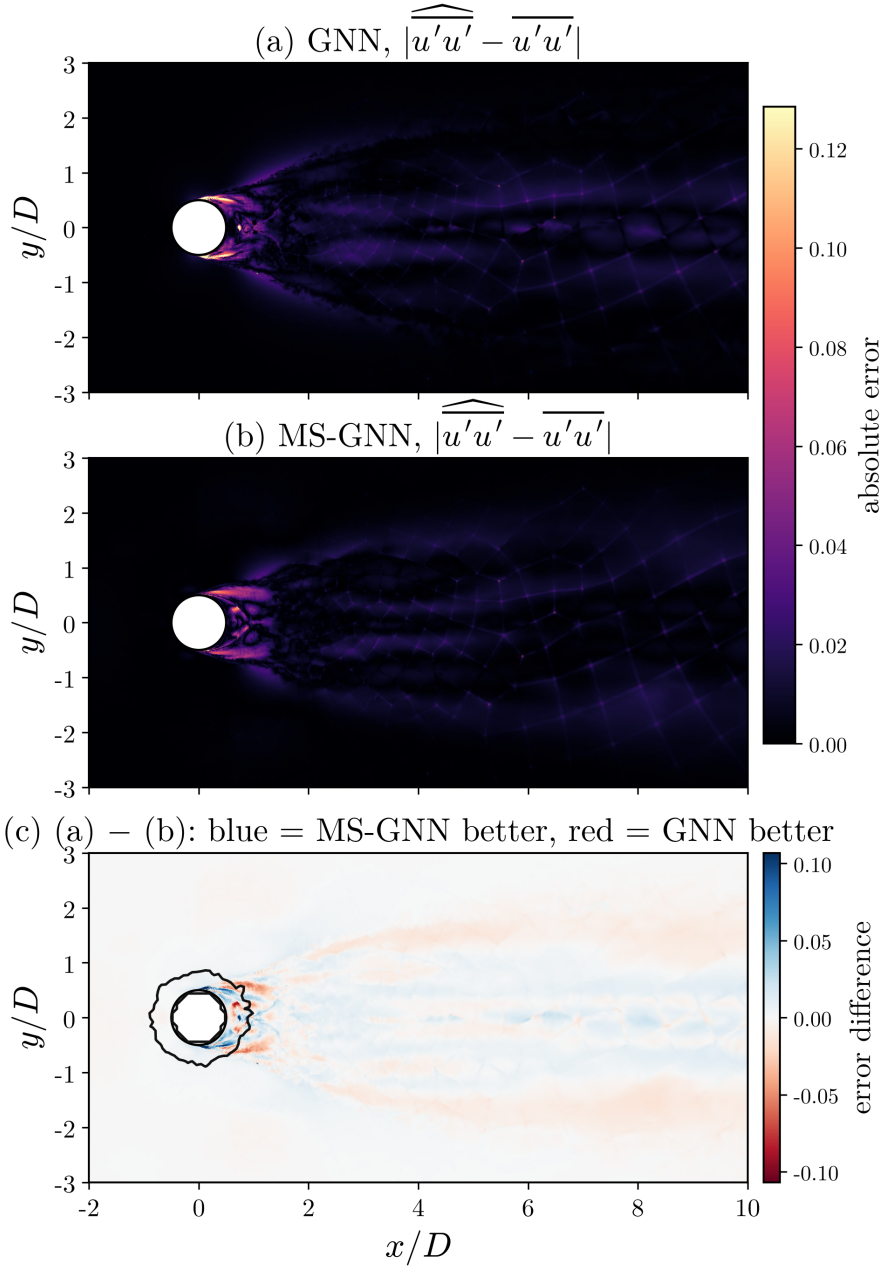



**Figure 5.11:** Spatial distribution of the error on the streamwise normal stress for the two graph architectures, and their difference. The contour on the last panel encloses the region where the cell size is below the median.

## 5.2 Coupling with the solver

The two graph closures are embedded in the same solver, on the same mesh and at the same Reynolds number as the closures of the previous two chapters, under the same numerical configuration and with the same convergence criterion. Nothing about the discretization, the stabilization or the treatment of the stress is changed for this chapter, so the rows quoted below from Chapter 3 and Chapter 4 are the same runs.

One difficulty carries over from the previous chapter and is not restated here. A closure that reads a neighbourhood receives the unconverged iterate at many points instead of one, so the probability that its input resembles anything seen in training falls with the size of that neighbourhood. That argument applies to a graph closure exactly as it applies to a vector-cloud one. What follows is the difficulty that does not carry over, and it is specific to this family. The graph is not the one they were trained on.

Both architectures were trained on the connectivity of the reference simulations' own quadrilateral meshes, Table 3.1, with between 78 743 and 155 085 elements. The solver runs on the triangular mesh of Section 3.3.5, coarser by more than an order of magnitude and different in cell topology, so the deployed closure rebuilds an analogous graph from the solver's own $P_1$ connectivity and, for the multiscale architecture, an analogous coarse level by the procedure of Section 5.1.5. Both constructions reproduce the *procedure* used in training; neither reproduces the *object* the network was trained on. Five rounds still reach five cells, but five cells of the solver mesh cover a much larger region. The coarse level compounds this, since the number of coarse nodes is a property of the training data rather than of the weights and cannot be read from the checkpoint: the value used, 500, is the one used in training, applied to a much smaller fine graph.

What follows therefore measures these networks outside the distribution of graphs they were trained on, and every comparison against the local and vector-cloud closures inherits that asymmetry in the direction unfavourable to them. Section 5.3 returns to it.

### 5.2.1 The numerical setting

Everything reported in this section uses the configuration adopted at the end of Chapter 3 and used again in Chapter 4: the consistent streamline-upwind stabilization of Equation (3.68) at $c_S = 1$, the filtered implicit and explicit split of Equation (3.77), under-relaxation $\alpha_{\text{closure}} = 0.3$ on the predicted stress, the near-wall damping of Section 3.3.7, and at most 150 Picard iterations. Both closures are evaluated at every node of the solver mesh at every iteration, with the graph built once before the loop and never rebuilt, and with the coarse level of the multiscale

architecture built once by the same procedure used in training. All four filtering configurations of Section 3.3.8 are run and reported, since the choice interacts with the architecture rather than being a property of the solver alone.

### 5.2.2 Results

Table 5.2 reports the four filtering configurations for both graph closures, with the local tensor-basis closure, the vector-cloud closure and the classical reference under the identical setting. Relative errors are on the evaluation window of Equation (3.67) and coefficients of determination over the whole domain, as in the corresponding tables of the previous two chapters.

**Table 5.2:** *A posteriori* accuracy of the two graph closures at $\mathrm{Re} = 11\,000$ under the adopted configuration, with the closures of the previous chapters reported under the same setting in their own adopted filtering configuration. Relative $L^2$ errors are on the window of Equation (3.67), coefficients of determination over the whole domain.

| Closure | Filter | $\delta u$ | $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | $\varepsilon_{L^2}^{\mathrm{tensor}}$ | $R^2(\overline{u'v'})$ |
|---|---|---|---|---|---|
| Graph, single scale | No filter | 2.8e-2 | 0.334 | 0.704 | $-0.35$ |
| | Input | 1.2e-2 | 0.059 | 0.321 | 0.53 |
| | Output | 2.4e-3 | 0.255 | 0.699 | 0.05 |
| | In+out | **1.6e-3** | 0.085 | 0.439 | 0.35 |
| Graph, multiscale | No filter | 3.5e-2 | 0.121 | 0.477 | 0.18 |
| | Input | 1.1e-2 | 0.073 | 0.360 | 0.16 |
| | Output | **1.1e-3** | **0.037** | **0.345** | **0.59** |
| | In+out | 1.0e-3 | 0.054 | 0.407 | 0.44 |
| *Reference closures, adopted filtering configuration* | | | | | |
| Tensor basis, local | In+out | 1.7e-3 | 0.048 | 0.402 | 0.57 |
| Vector cloud | In+out | 8.5e-3 | 0.061 | 0.442 | 0.43 |
| $k$-$\ell$ | In+out | 1.3e-5 | 0.103 | 0.640 | 0.50 |

The filtering configuration is not a detail for these two closures, and the two do not agree on it. The single-scale closure reaches its smallest update with both filters and its most accurate field with the input filter alone, and the two are an order of magnitude apart on the update, so neither configuration is good at both. The configuration adopted for it is therefore the one with both filters, which is also the one the previous two chapters adopt. The multiscale closure has no such conflict: the output filter alone gives the smallest update but one, the most accurate velocity

field, the most accurate stress and the highest shear correlation at the same time, and that configuration is adopted for it.

Under that configuration the multiscale graph closure produces the most accurate mean field of this thesis. Its velocity error is 0.037 against 0.048 for the local tensor-basis closure and 0.061 for the vector-cloud one, its stress error 0.345 against 0.402 and 0.442, and its shear correlation 0.59 against 0.57 and 0.43. It is ahead on all three at once, by a wide margin on the velocity and by a narrow one on the shear correlation, and it converges better than both. Against the classical reference the margin is a factor of three on the velocity and almost a half on the stress.

The single-scale closure is the weakest of the deployable closures. Under its adopted configuration it reaches 0.085 on the velocity, behind every other machine-learning closure of this work except the plain pointwise one, and 0.439 on the stress, behind the local tensor-basis closure. Its best field, obtained with the input filter, is competitive on the stress at 0.321 but is produced by an iteration whose final update is an order of magnitude above the adopted one.

Both graph closures converge better than the vector-cloud closure of Chapter 4, which under the same configuration stops at $8.5 \times 10^{-3}$. None of the three reaches the tolerance of the solver, fixed at $10^{-5}$, and the classical reference remains the only closure that does so.

### 5.2.3 Reading the numbers through the graph mismatch

The result of the previous paragraphs has to be read together with the caveat stated at the beginning of this section. The *a posteriori* numbers measure networks evaluated on a graph they were not trained on, which should cost accuracy, and both graph closures pay that cost. The multiscale closure is the stronger of the two on the coefficients of determination and stays the stronger inside the solver, where the distance between them widens instead of narrowing.

The explanation proposed here is in how the two architectures carry range, and it is a consequence of the same fact the chapter started from. The single-scale closure measures range in hops, and a hop is a cell: when the mesh changes, its receptive field changes with the cell size and nothing in the trained weights compensates. The multiscale closure carries its range on the coarse level, and the coarse level is built by sampling a fixed number of nodes spread over the domain, so its spacing is set by the size of the domain rather than by the size of a cell. Its long-range channel is therefore closer to a physical scale than to a mesh scale, and it survives the change of mesh better. That is a reading of the measurements and not a separate measurement, and Section 5.3 reports what can be quantified about it.

The consequence for how the chapter is read is straightforward. The *a priori* ordering of Table 5.1 and the *a posteriori* ordering of Table 5.2 do not conflict on the two graph closures. The multiscale one is ahead on the coefficients of determination

and ahead inside the solver, and behind only on the aggregate relative error, where the two are within a tenth of each other. What changes between them is the size of the gap, which is wider inside the solver than on reference data, and that is what the reading above accounts for. Figure 5.12 puts the same question to all four deployed closures with both errors measured on the training flow and over the same window, so that neither the Reynolds number nor the region differs between the two: the *a priori* ordering does not survive the coupling there either.

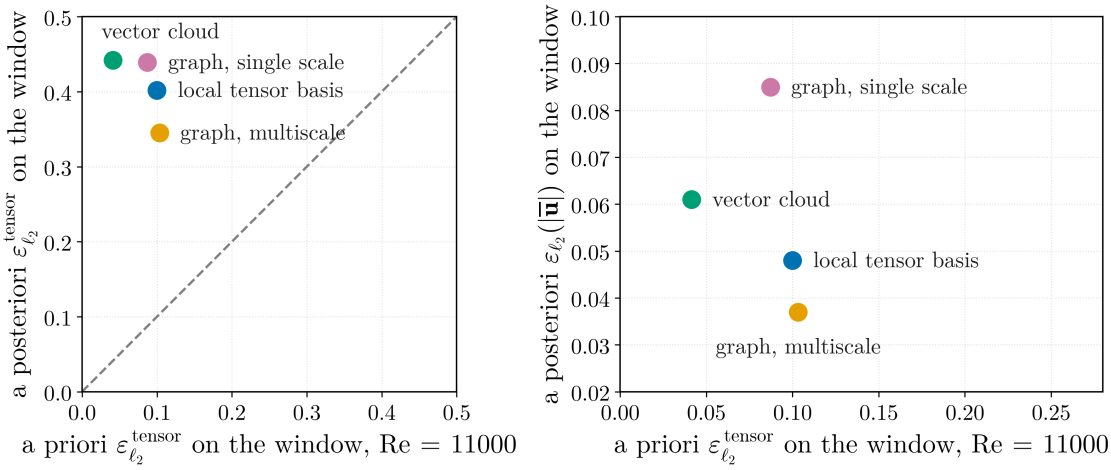



**Figure 5.12:** *A priori* against *a posteriori* accuracy of the four deployed closures at the same Reynolds number, Re = 11 000, and over the same window, that of Equation (3.67). On the abscissa, the relative $L^2$ error on the stress tensor of each closure evaluated on the reference fields of the training flow, with the definition used in Table 5.2; on the ordinates, the *a posteriori* error of the coupled solution of that table, on the stress (left, with the bisector) and on the velocity (right). The *a priori* errors are those of a fit on the training data, 0.041 for the vector cloud, 0.087 and 0.103 for the single-scale and the multiscale graph closures, 0.100 for the local tensor basis, and the coupling amplifies every one of them, by a factor of 3.3 for the multiscale graph closure and of 10.7 for the vector cloud. The ordering does not survive: the closure with the tightest fit, the vector cloud, is the least accurate *a posteriori* on the stress, 0.442; the multiscale graph closure, last of four *a priori*, is first on both the stress, 0.345, and the velocity, 0.037; the local tensor basis, *a priori* within three thousandths of it, is second on both. A smaller error on the training flow itself does not imply a smaller error once the closure is coupled, so the disagreement met in Chapters 3 and 4 is not an effect of the change of Reynolds number between the two measurements but a property of the coupling. The *a priori* error is computed on the grid of the reference data, the *a posteriori* one on the solver mesh, over the same region.

## 5.3 The receptive field

Range is the quantity this family pays for and the quantity it fails to carry across meshes. This section collects what can be said about it.

### 5.3.1 What range costs

On a single-scale graph the receptive field after $R$ rounds is $R$ cells, so reaching a physical distance $L$ costs $L/h$ rounds. Each round carries its own message networks, so parameters, memory and latency all grow linearly with the range, while the region reached grows only as the square of it in two dimensions. The comparison with the previous chapter is direct. A vector cloud of a hundred and fifty members at a radius of half a cylinder diameter reaches that distance at a fixed cost set by the number of members, and a five-hop graph on the same mesh reaches five cells. The two closures are both non-local, and the neighbourhoods they actually read differ by more than an order of magnitude in extent.

The multiscale variant changes the exponent rather than the constant. One hop on the coarse graph spans many cells of the fine one, so a fixed number of coarse rounds carries information across the domain regardless of how many cells lie in between. The price is paid once, in the construction of the coarse level and in the two transfer operators, and it does not grow with the distance reached. On this problem that price is about three times the parameters of the single-scale network, 23 407 against 8 280. The work done per evaluation is of the same order, since the cycle spends two rounds on the fine graph instead of five, the two transfers act on three edges per fine node, and the eleven coarse rounds run on a graph two orders of magnitude sparser.

### 5.3.2 What does not transfer, and what would settle it

That the coarse level is defined by a fixed number of nodes spread over the domain by farthest-point sampling, and the single-scale receptive field by a count of cells, is the reason proposed in Section 5.2 for why the multiscale closure loses less in the transfer. It is consistent with every measurement reported above, but it is not isolated by any of them.

Two experiments would settle it. Training both architectures directly on the connectivity of the solver mesh would remove the mismatch entirely and measure what the architectures do when the object they read is the object they learned on. However, the mesh used for training was build specifically for a Spectral Element Solver, and transporting it to the FEM solver implemented in this thesis is not straightforward. Alternatively, deploying the single-scale closure with the number of rounds rescaled so that its receptive field spans the same physical distance as at

training would separate the effect of the changed hop length from the effect of the changed connectivity. Both experiments are left for future studies.

## 5.4 Ablations

The comparisons are grouped by which step of the construction they vary, in the same order the chapter introduced them. Several were run before the architecture was final and vary more than one ingredient; where that is the case it is said.

### 5.4.1 What the closure reads

Widening the invariant node state through a learned embedding was tested at four widths, from five up to ninety-six. Every width tried was worse than the raw five-dimensional state, and two of them collapsed onto the local minimum described below. Each width was run at a different learning rate in order to find the best loss for each. What it establishes is that no configuration with an embedding matched the configuration without one.

Depth behaves differently depending on what consumes the node state. On the plain scalar variant of this architecture, which carries no vector channel, increasing the number of rounds beyond three at otherwise identical settings degrades the loss from 0.752 to 2.67 and beyond, a collapse rather than a gradual loss of accuracy. Depth is therefore useful only up to the point where the optimization remains well behaved. Where that point lies has been measured on the scalar variant and not on the architecture adopted here, which is stable at five rounds, so the two should not be conflated.

### 5.4.2 How the stress is rebuilt

**A direct readout cannot predict the shear stress.**

Replacing the tensor-basis reconstruction by a direct prediction of the three stress components from the invariant node state gives the results of Table 5.3.

**Table 5.3:** GNN: tensor-basis against direct readout, held-out $\mathrm{Re} = 140\,000$ test set. The first two rows share every setting except the readout.

| Readout | Rounds | $R^2_{\overline{u'u'}}$ | $R^2_{\overline{v'v'}}$ | $R^2_{\overline{u'v'}}$ |
|---|---|---|---|---|
| Tensor basis | 3 | 0.773 | 0.815 | 0.772 |
| Direct | 3 | 0.616 | 0.802 | −0.001 |
| Direct | 5 | 0.710 | 0.766 | −0.000 |

The direct readout fails on the shear stress completely, and does so at every learning rate and depth tested, with $R^2$ indistinguishable from zero in every case. This is a structural failure rather than a training failure, and it is the empirical counterpart of the invariance argument given above. The node state $\mathbf{h}_i$ is built from norms and inner products alone, so it is invariant under a reflection of the frame. The shear stress changes sign under that same reflection, as Equation (5.9) records. No function of an invariant argument can produce a sign-flipping output, so a direct readout is provably unable to predict $\overline{u'v'}$ from $\mathbf{h}_i$, and the value it converges to is the best available constant, which is what a vanishing $R^2$ means. The tensor bases supply exactly the missing structure, since $\mathbf{B}_1$ and $\mathbf{B}_4$ carry the sign behaviour of the strain rate and of the strain-rotation coupling.

**The two architectures deploy different heads, each the better one on reference data.**

Both fixed-basis heads of Chapter 4 were trained on both graph architectures, at parity in everything else, and then run through the solver under the configuration of Section 5.2. Table 5.4 reports the two measurements side by side, each head in the filtering configuration adopted for its architecture.

**Table 5.4:** Trace-normalized against capped reconstruction on both graph architectures, at parity in every other respect. The three coefficients of determination are on the held-out $\mathrm{Re} = 140\,000$ test set; the final update and the velocity error are measured at $\mathrm{Re} = 11\,000$ under the adopted configuration, with both filters on the single-scale architecture and the output filter on the multiscale one. The dagger marks the head adopted for each architecture.

| | Graph, single scale | | Graph, multiscale | |
|---|---|---|---|---|
| | Trace-norm. | Capped$^\dagger$ | Trace-norm.$^\dagger$ | Capped |
| $R^2(\overline{u'u'})$ | 0.709 | **0.773** | **0.926** | 0.815 |
| $R^2(\overline{v'v'})$ | 0.900 | **0.944** | **0.945** | 0.929 |
| $R^2(\overline{u'v'})$ | 0.869 | **0.893** | **0.925** | 0.901 |
| $\delta u$ | 5.7e-3 | **1.6e-3** | 1.1e-3 | 1.2e-3 |
| $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | 0.125 | **0.085** | 0.037 | 0.037 |

The two architectures disagree on which head is better on reference data, and the disagreement is not small: on the single-scale one the capped head is ahead on all three components, on the multiscale one the trace-normalized head is ahead by eleven hundredths on the streamwise normal stress. Inside the solver the same head wins on each architecture, so the two measurements agree. On the single-scale

architecture the capped head is ahead by a factor of three on the final update and by a third on the velocity error, and there is nothing to trade. On the multiscale one the two are close in the configuration adopted for it, $1.1 \times 10^{-3}$ against $1.2 \times 10^{-3}$ on the update and the same velocity error to three digits, so the advantage on reference data is what decides. Each architecture therefore deploys the head that is better on its own reference data, and on neither of them does the solver contradict that choice.

One quantity does not follow that ordering. Table 5.5 reports both heads on this architecture across the four filtering configurations, with the update at the last iteration and at its lowest value along the way. In the adopted configuration and with both filters the last update is also the lowest one for both heads, so the iteration is still descending when it stops. Without a filter and with the input filter alone the trace-normalized head passes through a lower update and then climbs back, while the capped head reaches its lowest value at the last iteration with the input filter and, without a filter, settles at about half the update of the other. Neither head converges without a filter. The head deployed on this architecture is therefore the accurate one and not the robust one, and the choice rests on the adopted configuration using the output filter, where both converge.

**Table 5.5:** The two reconstruction heads on the multiscale architecture at Re = 11 000, across the four filtering configurations. The update is reported both at the last iteration and at the lowest value reached along the iteration. The dagger marks the filtering configuration adopted for this architecture.

| | No filter | Input | Output$^\dagger$ | In+out |
|---|---|---|---|---|
| *Trace-normalized, the head deployed* | | | | |
| $\delta u$ at the last iteration | 3.5e-2 | 1.1e-2 | 1.1e-3 | 1.0e-3 |
| $\delta u$ at its lowest | 6.6e-3 | 7.1e-3 | 1.1e-3 | 1.0e-3 |
| $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | 0.121 | 0.073 | 0.037 | 0.054 |
| *Capped* | | | | |
| $\delta u$ at the last iteration | 1.7e-2 | 9.4e-4 | 1.2e-3 | 1.5e-3 |
| $\delta u$ at its lowest | 1.5e-2 | 9.4e-4 | 1.2e-3 | 1.5e-3 |
| $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | 0.123 | 0.073 | 0.037 | 0.068 |

Read together with Chapter 4 this is the reverse of what that chapter found. There the capped head was the more accurate on reference data and could not be coupled at all, while the trace-normalized one was the only one that converged, so reference data pointed at the head that had to be discarded. Here it points at the head that is deployed, on both architectures. Neither head is intrinsically the one that couples, and the ordering established on one family does not carry to the next,

which is why both heads were run through the solver on both architectures rather than chosen from the reference tables.

### 5.4.3 How much of the coarse branch is used

The depth of the coarse branch was swept at fixed width, and Table 5.6 reports the outcome at two learning rates. At each fixed learning rate the comparison isolates the number of coarse rounds alone. The networks of the sweep carry the reconstruction adopted for this architecture, so the comparison is on the deployed configuration.

**Table 5.6:** MS-GNN: test loss against the number of coarse rounds, at two learning rates, with everything else held fixed. The starred entry is the test loss recorded by a run that stopped because got stuck.

| Coarse rounds | $\eta = 10^{-3}$ | $\eta = 10^{-2}$ |
|---|---|---|
| 5 | 1.188 | $3.217^{*}$ |
| 8 | 1.260 | 0.349 |
| 11 | 0.681 | **0.205** |

Eleven coarse rounds is the best depth at both learning rates, and at the higher one it improves on eight by forty per cent. At both depths that ran to completion the loss is markedly worse at the lower learning rate, where the dependence on depth is also not monotone. This interaction between depth and learning rate is not incidental. The coarse branch is where the network accumulates long-range information, and a learning rate too small to move the coarse weights appreciably within the training budget leaves that branch close to its initialization, in which case the extra rounds add parameters without adding reach. The coarse branch is therefore used, and it is used more the more the optimizer is allowed to train it. Eleven rounds is the depth deployed, at a learning rate between the two shown. The shallowest configuration at the higher learning rate is the one entry that is not a measurement, since the run stopped, and nothing is concluded from it here.

### 5.4.4 How the network is trained

**A reproducible local minimum.**

This family has a failure mode worth documenting, because it interacts with the choice made above. In several configurations the composite loss converges to a strong local minimum in which the network predicts a field that is nearly constant in the whitened coordinates of Equation (5.8). That solution simultaneously satisfies

the global magnitude, global mean, energy and anisotropy terms, since all of them compare aggregate statistics, while leaving the regression term at the variance of the data. The signature is unambiguous: the coefficient of determination becomes negative on both normal stresses, around $-0.25$, and the correlation with the true field is essentially zero.

The first diagnosis attributed the collapse to inactive units in the message networks, whose derivative is exactly zero for negative arguments. Replacing every such unit by one whose derivative is small but never exactly zero did not change the outcome. The run collapsed again, on the same value of the loss to the digits reported. With the same seed the initial weights are identical between the two versions, but the trajectories should separate quickly once an activation changes, so reaching the identical value is evidence that the value is a property of the loss landscape rather than of the activation. Rerunning at a lower learning rate avoids the collapse but plateaus well above the reported configuration. The consequence is that the configuration reported here occupies a narrow region of hyperparameter space that has been explored, and changes altering the gradient path through the readout move the problem outside it. The capped head inserts two saturating functions in series on that path, which is a reason to expect it to be the more fragile of the two to train, and it is the one the solver requires on the single-scale architecture.

**Augmentation, weight decay and the composite loss.**

The augmentations were tested by doubling the training set from six to twelve graphs at fixed architecture, learning rate and number of epochs, which lowered the test loss from 0.639 to 0.505. This comparison isolates the reflected copies alone. The architecture is equivariant under exactly the transformation those copies apply, so in principle they carry no information the network cannot already deduce, and the measured improvement is reported here as an observation without a validated mechanism behind it. Weight decay was swept at fixed everything else, giving losses of 0.845, 1.063 and 2.622 at $10^{-6}$, $10^{-4}$ and $10^{-3}$ respectively; the largest value collapses the model onto the minimum described above, and the smallest is retained. Removing the auxiliary terms of the composite loss and keeping only the local regression term leaves that term at 0.601, above the value the composite loss reaches on the same term, so the auxiliary terms help even by the reduced loss's own measure, which is the pattern already observed for the local closures in Section 3.2.

# Chapter 6

# Local closure for URANS

Every closure considered so far predicts a time-averaged Reynolds stress and is embedded in a steady solver. That combination is only really meaningful when the mean flow is genuinely stationary. Behind a circular cylinder in the Reynolds range considered here it is not. The instantaneous flow sheds vortices alternately from the two sides of the body, and the time average is stationary only because the shedding is periodic and the averaging window spans many periods. A steady solver reproduces that average directly and never represents the periodic motion that produces it.

The unsteady formulation resolves the shedding and averages only the turbulent fluctuations, so the closure is now required to supply the stress at every instant of a phase, not once for the whole flow. Two things change together. The solver must march in time, and the closure must be evaluated against a mean field that itself varies in time. Section 6.1 describes the first of the two, following the same structure as Section 3.3.4 for the steady case, and reports the numerical decisions that the transition to time marching forces. Section 6.2 states what the closure is asked to predict, what a coupled solution has to satisfy to be usable, and the two reference points against which it is read. Sections 6.3 to 6.6 then take four ways of training the local tensor-basis closure in the unsteady setting one at a time, each first against phase-resolved reference data and then inside the solver. Section 6.7 reads the four side by side, Section 6.8 measures what the coupled solutions still get wrong, and Section 6.9 collects the comparisons behind each choice, including the one on the form of the split that decides whether a limit cycle exists at all.

## 6.1 Solver FEM

### 6.1.1 Governing equations and function spaces

The solver advances the unsteady, incompressible URANS equations for the phase-averaged velocity $\overline{\mathbf{u}}$ and pressure $\overline{p}$, closed by the Reynolds stress tensor $\boldsymbol{\tau}$ supplied by the closure under test,

$$\frac{\partial \overline{\mathbf{u}}}{\partial t} + (\overline{\mathbf{u}} \cdot \nabla)\overline{\mathbf{u}} - \nu \Delta \overline{\mathbf{u}} + \nabla \overline{p} = \nabla \cdot \boldsymbol{\tau}, \tag{6.1}$$

$$\nabla \cdot \overline{\mathbf{u}} = 0. \tag{6.2}$$

Equations (6.1)–(6.2) differ from their steady counterparts, Equations (3.33)–(3.34), by the time derivative alone. Non-dimensionalization, the treatment of $\boldsymbol{\tau}$ as a general symmetric tensor forcing rather than an eddy-viscosity term, and the split of the viscous term into a Laplacian form for the constant molecular viscosity and a symmetric-gradient form for the variable eddy viscosity are all carried over unchanged from Section 3.3.4.

The spatial discretization is likewise unchanged, a Taylor–Hood pair with velocity in $P_2$ and pressure in $P_1$ on a triangular mesh, so that the pair is inf-sup stable, assembled and solved as one coupled system..

### 6.1.2 Time discretization and linearization

The time derivative is discretized with a second-order backward differentiation formula. Writing $\mathbf{u}^n$ for the velocity at time level $n$ and $\Delta t$ for the step,

$$\left.\frac{\partial \overline{\mathbf{u}}}{\partial t}\right|^{n+1} \approx \frac{3\mathbf{u}^{n+1} - 4\mathbf{u}^n + \mathbf{u}^{n-1}}{2\Delta t}. \tag{6.3}$$

Equation (6.3) needs two known past levels, and at the first step only the initial condition exists. A single backward Euler step supplies the missing level,

$$\left.\frac{\partial \overline{\mathbf{u}}}{\partial t}\right|^{1} \approx \frac{\mathbf{u}^1 - \mathbf{u}^0}{\Delta t}, \tag{6.4}$$

after which Equation (6.3) takes over for the whole run.

The convective term is nonlinear, exactly as in the steady case, but the remedy is different. The steady solver freezes the advecting velocity at the previous Picard iterate and iterates to convergence at a single time level, because there is no time level to advance to. Here there is, and iterating to a self-consistent state within each step is unnecessary if the advecting velocity is extrapolated from the past

levels accurately enough. The convection is therefore linearized in Oseen form around a second-order extrapolation,

$$\mathbf{u}^{*} = 2\mathbf{u}^{n} - \mathbf{u}^{n-1}, \qquad (\overline{\mathbf{u}} \cdot \nabla)\overline{\mathbf{u}} \;\longrightarrow\; (\mathbf{u}^{*} \cdot \nabla)\overline{\mathbf{u}}, \tag{6.5}$$

with $\mathbf{u}^{*} = \mathbf{u}^{n}$ at the bootstrap step. The extrapolation and the mass term switch to their bootstrap forms on the same condition, so the two are always consistent. First-order extrapolation would be simpler and would reduce the whole scheme to first order in time regardless of Equation (6.3), which is why the second-order form is used.

With Equations (6.3) and (6.5) the problem solved at each step is linear in the unknowns, so a single linear solve advances the solution by one step. This is the structural difference from the steady solver, where a fixed-point iteration is unavoidable. Corrector sub-iterations, in which the extrapolated velocity is replaced by the solution just computed and the step is repeated, restore a fixed-point structure within each step at the cost of one more linear solve per sub-iteration. Three sub-iterations per step are used for every computation reported in this chapter. The closure is re-evaluated at each of them, on the re-extrapolated velocity, while the quantities that carry the time history of the closure, the rate features of Section 6.4.1 and the internal state of the recurrent head of Section 6.5.1, advance once per step and not once per sub-iteration. A single solve per step would already be second-order consistent, since Equation (6.5) provides a second-order estimate of the advecting velocity, and the sub-iterations are a robustness choice rather than a requirement of the scheme.

A continuation device eases the start of the run. An artificial viscosity is added to the molecular one and ramped down linearly,

$$\nu_{\mathrm{art}}(n) = \max\Big(\nu_{\mathrm{floor}},\; \nu_{\mathrm{art},0}\big(1 - n/n_{\mathrm{ramp}}\big)\Big), \qquad \nu_{\mathrm{art},0} = 0.05, \qquad n_{\mathrm{ramp}} = 10, \tag{6.6}$$

with $\nu_{\mathrm{floor}} = 0$, so that the artificial viscosity vanishes identically after ten steps. The ramp is much shorter than its steady counterpart of Equation (3.41), and deliberately so. In the steady solver the ramp is a homotopy: the first iterations solve an easier, more diffusive version of the same stationary problem, and the artificial viscosity is withdrawn as the iterate approaches the target. That reading does not transfer to time marching. After a few steps there is already a genuine, physically evolving solution, and blurring it with a large artificial viscosity would contaminate the very dynamics the simulation is meant to capture. The only role left for Equation (6.6) is to carry the bootstrap step and the first few second-order steps through the transient produced by an impulsive start, without a spurious startup oscillation. Disabling it altogether after the bootstrap step was considered and rejected as a default, because the far-field cells are large enough that an abrupt jump at the first step risks precisely the single-step transient the ramp exists to avoid.

### 6.1.3 Explicit–implicit splitting of the closure term

The splitting of Section 3.3.3 is carried over unchanged. At every step the closure is evaluated once, on the extrapolated velocity $\mathbf{u}^{*}$ rather than on a Picard iterate, and the resulting stress is decomposed into the eddy viscosity that best explains it in a least-squares sense and an explicit remainder,

$$\nu_{t,\mathrm{eff}} = \max\!\left(0,\; -\frac{\boldsymbol{\tau} : \mathbf{S}}{2\,\mathbf{S} : \mathbf{S}}\right), \qquad \boldsymbol{\tau}_{\mathrm{res}} = \boldsymbol{\tau} + 2\nu_{t,\mathrm{eff}}\mathbf{S}, \tag{6.7}$$

so that the sum reconstructs $\boldsymbol{\tau}$ exactly. The first term enters the bilinear side of the weak form and the second the right-hand side.

One detail of Equation (6.7) deserves stating explicitly, because it is easy to get wrong and its consequence is a spurious stress rather than a loss of accuracy. When the output filter of Section 6.1.8 is active, two different strain-rate fields are available, the filtered one and the raw one. The eddy viscosity is fitted against the filtered strain rate, since that is the field the implicit term will multiply, while the explicit remainder is constructed against the raw strain rate, since that is the field with respect to which the identity must hold. Using a single field for both leaves an unbalanced stress of magnitude $2\nu_{t,\mathrm{eff}}(\mathbf{S}_{\mathrm{filt}} - \mathbf{S}_{\mathrm{raw}})$ in the momentum equation. With the filter length proportional to the cell size that term is of order $h^{2}$ and acts as a hyperviscosity of coefficient $\nu_{t,\mathrm{eff}}$ times the square of the filter length, so it vanishes under mesh refinement but not on any given mesh. The steady solver adopts exactly that term in its reference configuration, as a declared stabilization, Section 3.4.4. The unsteady computations reported in this chapter keep the two strain rates separate, so that the stress imposed on the flow is the closure's own, and the effect of adopting the hyperviscous form here as well is measured in Section 6.9.3.

The quantity that is relaxed between steps is the closure prediction itself rather than the two halves of the split separately. Relaxing the halves independently imposes a stress carrying an extra term $2(1-w)\,\nu_{t,\mathrm{eff}}(\mathbf{S}^{n} - \mathbf{S}^{n+1})$, which is not present in either the previous or the current prediction. That term was measured at about three per cent of the median magnitude of the predicted stress, small but systematic, and it is removed by relaxing the prediction and splitting once.

### 6.1.4 Discretized matrix form and stabilization

Let $(\mathbf{u}, p)$ be the trial functions and $(\mathbf{w}, q)$ the test functions of the mixed space. The time derivative of Equation (6.3) contributes one term depending on the unknown and one depending on known past levels,

$$\mathbf{M}_{\mathrm{trial}} = \frac{c_{n+1}}{\Delta_c}\,\mathbf{u}, \qquad \mathbf{M}_{\mathrm{known}} = \frac{c_{n}\mathbf{u}^{n} + c_{n-1}\mathbf{u}^{n-1}}{\Delta_c}, \tag{6.8}$$

with $(c_{n+1}, c_n, c_{n-1}, \Delta_c) = (3, -4, 1, 2\Delta t)$ for the second-order formula and $(1, -1, 0, \Delta t)$ for the bootstrap step. The two must be assembled as separate terms, the first into the bilinear form and the second, with reversed sign, into the linear form.

A grad-div term and a Temam skew-symmetrization are added to the physical terms. The first has the same form as in the steady case, Section 3.3.4, and is used here with a coefficient of one rather than the value adopted there. Like the stabilization discussed below, it vanishes on a divergence-free solution and therefore does not alter the solution the discrete problem converges to. The second term is specific to the Oseen linearization,

$$+ \int_\Omega \tfrac{1}{2}\big(\nabla \cdot \mathbf{u}^*\big)\, \mathbf{u} \cdot \mathbf{w}\; d\Omega. \tag{6.9}$$

Its purpose is to restore the skew symmetry of the discrete convection operator. For a pointwise divergence-free advecting field the operator $(\mathbf{u}^* \cdot \nabla)$ is skew-symmetric, hence neutral in energy, and the discrete system inherits its coercivity from the viscous term alone. The Taylor–Hood pair enforces the divergence-free constraint only weakly, against the pressure test space, so the discrete advecting field is not pointwise divergence free and the convection operator can inject energy. Equation (6.9) cancels that injection exactly. The price is stated rather than hidden: unlike the stabilization terms discussed below, Equation (6.9) does not vanish on the discrete solution, so it adds a consistency error of order $\|\nabla \cdot \mathbf{u}_h\|$ instead of removing one. The trade is accepted here and would not be in the steady solver. The advecting field is an extrapolation of two only weakly divergence-free fields, and the simulation marches thousands of steps, so any energy the convection operator injects accumulates over the run rather than being annihilated at a fixed point.

The streamline-upwind term is built from the strong residual of the linear problem actually solved at each step. Collecting every trial-dependent term of the momentum equation into one operator,

$$\mathcal{A}(\mathbf{u}, p) = \frac{c_{n+1}}{\Delta_c}\mathbf{u} + (\mathbf{u}^* \cdot \nabla)\mathbf{u} + \tfrac{1}{2}(\nabla \cdot \mathbf{u}^*)\,\mathbf{u} + \nabla p - (\nu + \nu_{\text{art}})\Delta\mathbf{u} - 2\nu_{t,\text{eff}}\,\nabla \cdot \mathbf{S}(\mathbf{u}), \tag{6.10}$$

and writing $\mathbf{f}_{\text{expl}} = -\mathbf{M}_{\text{known}} - \nabla \cdot \boldsymbol{\tau}_{\text{res}}$ for the known data, the step solves $\mathcal{A}(\mathbf{u}, p) = \mathbf{f}_{\text{expl}}$ and the stabilization term is

$$+ \int_\Omega \tau_{\text{SUPG}} \Big[\mathcal{A}(\mathbf{u}, p) - \mathbf{f}_{\text{expl}}\Big] \cdot (\mathbf{u}^* \cdot \nabla)\mathbf{w}\; d\Omega. \tag{6.11}$$

Equation (6.11) is consistent in the strict sense. Its integrand is the residual of the equation being solved, so it vanishes identically on the discrete solution whatever value $\tau_{\text{SUPG}}$ takes, and the term changes how strongly convection-driven oscillations are damped along the way without changing which solution the discrete problem

converges to. Consistency requires the full operator of Equation (6.10) rather than the convective term alone. Retaining only convection would leave a residual that does not vanish on the exact solution, since there it equals the remaining terms rather than zero, and the stabilization would then act as a spurious body force.

Splitting Equation (6.11) between the two sides of the linear system puts the whole operator $\mathcal{A}$ into the bilinear form and the known data into the linear form. Unlike the steady case, the linear form always carries a stabilization contribution here, because $\mathbf{M}_{\text{known}}$ is a genuinely non-zero known field built from the past levels.

The stabilization parameter follows the standard transient form, with the element time scale entering alongside the convective and diffusive ones,

$$\tau_{\text{SUPG}} = c_{\text{SUPG}} \left( \frac{c_{n+1}}{\Delta_c} + \frac{2\|\mathbf{u}^*\|}{h_p} + \frac{4(\nu + \nu_{\text{art}} + \nu_{t,\text{eff}})}{h_p^2} \right)^{-1}, \qquad h_p = \frac{h}{p}, \tag{6.12}$$

with $h$ the local cell diameter and $p = 2$ the polynomial degree of the velocity space [66, 23]. Three features of Equation (6.12) are worth isolating.

- The first term is the reciprocal of the element time scale. Omitting it, as a formula transplanted from a steady solver would, overestimates $\tau_{\text{SUPG}}$ by a factor of three in the near-wall cells and by two orders of magnitude in the far field, measured at $\Delta t = 0.01$ on this mesh. Since the operator of Equation (6.10) contains a mass term proportional to $1/\Delta t$, an overestimated parameter lets the stabilized mass contribution dominate the bilinear form and destroy its coercivity.

- The coefficient of the trial velocity in the mass term is used rather than the classical constant $2/\Delta t$. It equals $1/\Delta t$ at the bootstrap step and $3/(2\Delta t)$ thereafter, so the parameter adapts to the change of scheme without a separate branch.

- The length scale is the cell diameter divided by the polynomial degree. The velocity space is quadratic, so the resolved length within a cell is half the cell size, and using the full cell size would overestimate the parameter by a factor between two and four across the mesh.

Two further terms appear when the closure output is filtered onto a continuous space, as described in Section 6.1.8. The argument that the divergence of the explicit stress vanishes holds only for a cell-wise constant tensor. Once that tensor is continuous and piecewise linear, the identity

$$\nabla \cdot \left( 2\nu_{t,\text{eff}} \mathbf{S} \right) = 2\nu_{t,\text{eff}} \, \nabla \cdot \mathbf{S} + 2\, \mathbf{S} \cdot \nabla \nu_{t,\text{eff}} \tag{6.13}$$

contributes a second term to Equation (6.10), and $\nabla \cdot \boldsymbol{\tau}_{\text{res}}$ contributes a second term to the known data. Omitting either destroys the consistency established

above, since the stabilization would then no longer vanish on the exact solution of the step, and would act as a body force of magnitude proportional to the gradient of the closure output.

The resulting system has the same block saddle-point structure as Equation (3.57), with the mass, convection, skew, grad-div and stabilization matrices collected into the velocity block. It is solved monolithically by a direct factorization at every step.

### 6.1.5 Mesh

The mesh differs from the steady one in a way that is not a matter of resolution but of topology, and the reason is the physics of the problem.

The steady solver restricts the domain to the upper half-plane and imposes a symmetry condition on the wake centreline, on the grounds that the time-averaged flow is symmetric. That argument is exactly what fails here. Vortex shedding is the antisymmetric mode of this flow, and a hard symmetry condition on the centreline removes the antisymmetric mode from the discrete solution space entirely. On a half domain the shedding cannot emerge, no matter how long the simulation runs or how finely the numerics are tuned. The unsteady solver therefore uses the full domain, with slip conditions on the upper and lower far-field boundaries and no symmetry condition anywhere.

Two full meshes were built and compared. The first is the direct extension of the steady mesh, retaining its anisotropic boundary-layer annulus with a first-cell height of $6 \times 10^{-4} D$ sized so that $y^+ \leq 1$ over 93.5% of the cylinder wall, on a domain $x \in [-15,30]D$ and $y \in [-15,15]D$, giving a blockage ratio of 3.3%. The second follows the construction used by the reference unsteady computations against which this solver is checked, with isotropic wall cells of size $0.02D$ and no boundary-layer field, on the smaller domain $x \in [-10,20]D$ and $y \in [-10,10]D$, giving a blockage ratio of 5%. The first cell centroid then sits at about $0.0058D$ from the wall, corresponding to $y^+$ of order two over most of the cylinder and reaching eight to ten at the peak of the wall shear stress.

On wall resolution alone the first mesh is strictly better, and it was the natural choice. The measured shedding characteristics reversed that expectation. On the boundary-layer mesh the dominant frequency of the lift coefficient corresponded to a Strouhal number of 0.067 to 0.083, which turned out on inspection to be a modulation frequency rather than the shedding frequency, and no clean shedding tone was present. On the second mesh the Strouhal number is between 0.17 and 0.20 against a reference value of 0.196. The second mesh is therefore adopted for all unsteady computations reported here.

Five things change together between the two computations, namely the wall resolution, the wake refinement, the domain size, the blockage ratio and the form of

**Table 6.1:** The two full-domain meshes considered for the unsteady computations.

| | Boundary-layer mesh | Adopted mesh |
|---|---|---|
| Streamwise extent | $[-15{,}30]D$ | $[-10{,}20]D$ |
| Cross-stream extent | $[-15{,}15]D$ | $[-10{,}10]D$ |
| Blockage ratio | 3.3% | 5% |
| Wall treatment | anisotropic annulus | isotropic |
| First cell height | $6 \times 10^{-4} D$ | $2 \times 10^{-2} D$ |
| Wall-normal resolution | $y^+ \leq 1$ on 93.5% of the wall | $y^+ \approx 2$ to 10 |
| Cells | 38 542 | 25 944 |
| Nodes | 19 486 | 13 135 |

the near-wall damping discussed in Section 6.1.7, so the comparison above does not isolate which of them is responsible. A consequence of the reduced wall resolution is that the algebraic length scale of the classical reference, which is proportional to the wall distance and therefore assumes the near-wall layer to be resolved, loses its justification on the adopted mesh. To its place, another damping function is introduced in Section 6.1.7. The domain and the mesh are shown in Figure 6.1.

### 6.1.6 Boundary and initial conditions

Boundary conditions are those of Section 3.3.6, transposed to the full domain. A uniform inflow is imposed at the inlet, no slip on the cylinder, and a slip condition on both far-field boundaries. The outlet carries no essential condition, so the natural do-nothing condition arising from the weak formulation applies there, and the pressure gauge is fixed by it without pinning the pressure at a point.

The initial condition is a genuine degree of freedom in the unsteady case, and it is worth being explicit about what it must and must not satisfy. The fields at the two initial levels are historical data feeding the extrapolation of Equation (6.5) and the mass term of Equation (6.8). They are not the constrained unknown, which is subject to the boundary conditions at every step regardless. The initial condition therefore need not satisfy the boundary conditions, and choosing it so that it does is not required.

Two initial conditions are considered. The first is a fluid at rest, which is the simplest choice. It leaves the antisymmetric mode to be excited by rounding alone, so its initial amplitude is far below anything physical and the transient is correspondingly long. On the earlier mesh configuration this was observed to be too weak to reach a self-sustained limit cycle within the simulated time. The second condition seeds the asymmetry deliberately, prescribing a uniform streamwise velocity of $1.01\,U_\infty$ in the upper half of the domain and $U_\infty$ in the lower half, which

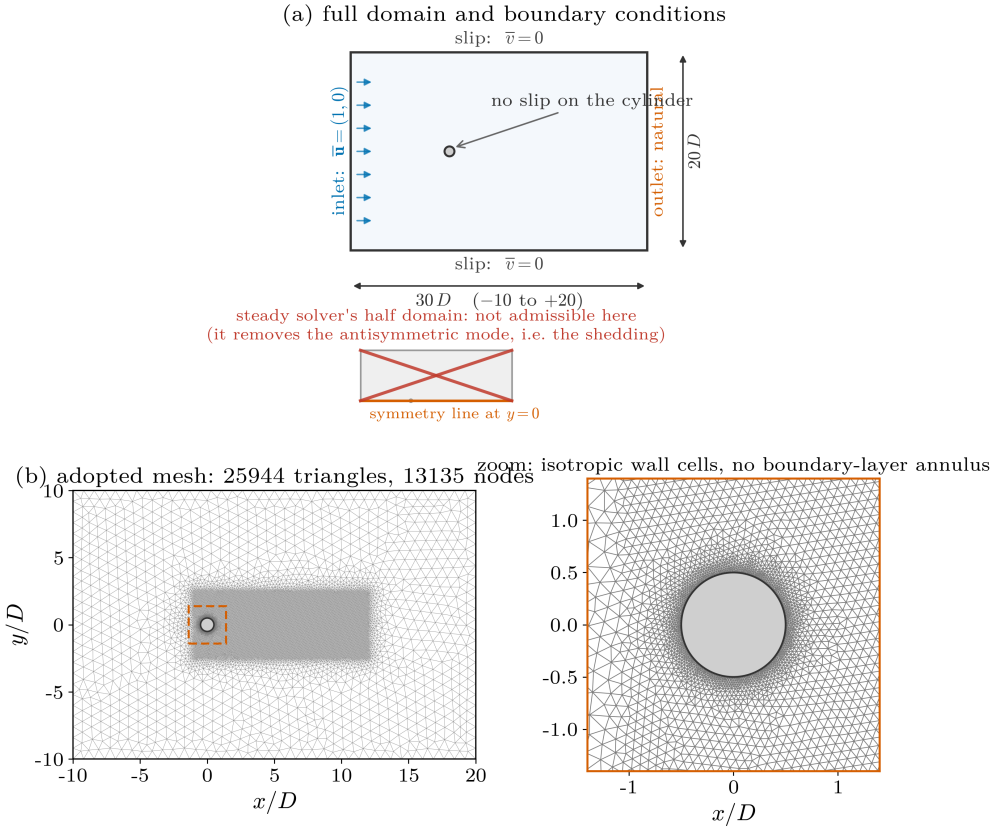



**Figure 6.1:** Domain and mesh for the unsteady computations. The symmetry condition used by the steady solver removes the antisymmetric mode from the discrete solution space, and vortex shedding is that mode, so the full domain is required.

starts the antisymmetric mode at a macroscopic amplitude. The second condition is the one used for every computation reported in this chapter, and the length of the initial transient in Figure 6.2 is to be read with that in mind.

### 6.1.7 Near-wall stress damping

The requirement of Equation (3.62) carries over unchanged, and some form of near-wall damping is active for every closure. Two forms are used in this work. The first is the component-wise damping of Equation (3.63), which reproduces the three asymptotic exponents individually. It is only been tested but the near wall

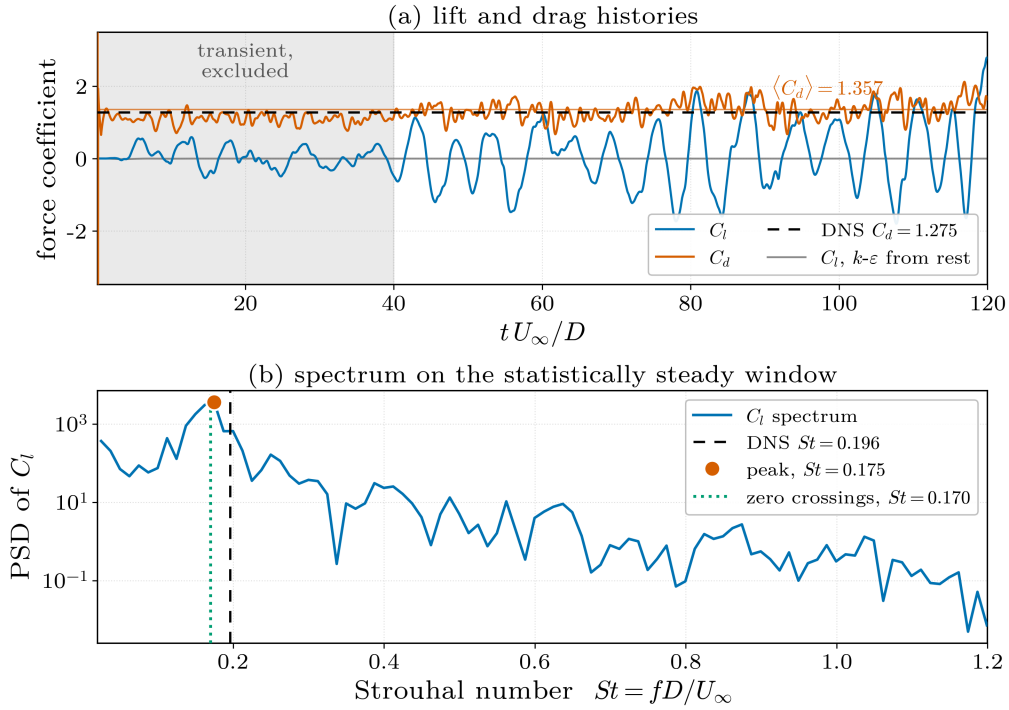



**Figure 6.2:** Lift and drag histories over a full computation, and the power spectral density of the lift coefficient on the statistically steady window. The dashed line marks the reference Strouhal number.

mesh in this case is not small enough for it, as already explained in Section 6.1.5. The second applies a single factor to all three components,

$$\overline{u'u'} \mapsto \varphi\,\overline{u'u'}, \quad \overline{v'v'} \mapsto \varphi\,\overline{v'v'}, \quad \overline{u'v'} \mapsto \varphi\,\overline{u'v'}, \qquad \varphi = \tanh\!\left(\frac{y}{d_0}\right), \tag{6.14}$$

which vanishes linearly rather than quadratically as the wall is approached and therefore does not reproduce the exponents of Equation (3.62), but damps more gently over a wider band. Equation (6.14) with $d_0 = 0.2D$ is the form used for the computations reported in this chapter. This damping function, with this same constant, is taken from the reference solver implemented in [3] and it is motivated by a practical reason rather than a physical one. The near wall region is the most sensitive one, but it is also the region where the net makes more mistake. Therefore, closure predictions are better to be masked there, in order to get reasonable results.

One ordering detail matters enough to record independently of which form is used, because getting it wrong removes the damping almost entirely while leaving it apparently switched on.

The damping must be applied after the output filter of Section 6.1.8, not before it. Filtering after damping smears values from outside the damping band back into

it, and the filter length near the band line is not small enough, so the smearing is not a perturbation. The effect was measured directly on the resulting stress field. With the filter applied last, the fitted near-wall exponent of $\overline{u'u'}$ is 0.07 where the asymptotic behaviour requires 2, the ratio $\overline{v'v'}/\overline{u'u'}$ stays flat at about 0.68 across the band where the two exponents should separate them by three orders of magnitude, and the shear stress at the first cell reaches 8.5 times the true wall shear stress. With the damping applied last, the fitted exponent is 1.88 against the required 2, and the stress at the wall falls from 169 times the wall shear stress with no damping to 1.5 times. Figure 6.3 shows the near-wall behaviour of the stress with and without the damping.

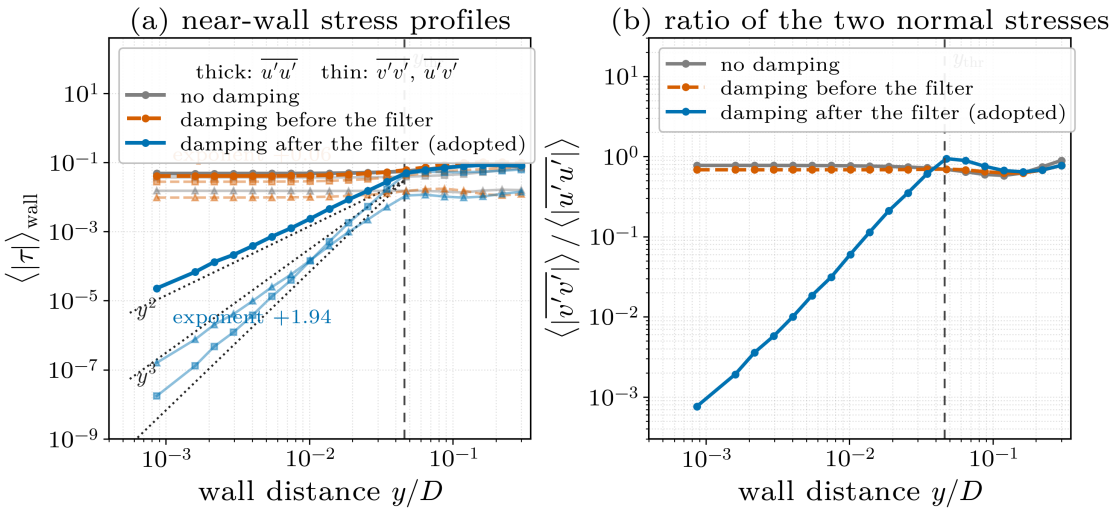



**Figure 6.3:** Near-wall behaviour of the predicted stress with and without damping, and with the damping applied before and after the output filter. The dashed lines are the asymptotic slopes required by Equation (3.62).

### 6.1.8 Feature and stress filtering

The Helmholtz filter of Equation (3.65) is available in the same two places as in the steady solver, on the mean-flow gradients feeding the closure and on the predicted stress leaving it, with independent length scales. The discussion of Section 3.3.8 on why the filtered stress is kept in the continuous space rather than projected back onto cell averages applies here without change, and the two additional stabilization terms this entails are those already given in Equation (6.13) and the paragraph following it.

### 6.1.9 Force coefficients and shedding diagnostics

The quantities of interest in the unsteady case are integral rather than pointwise. The force on the cylinder is obtained from the traction of the total stress on its surface,

$$\boldsymbol{\sigma} = -\overline{p}\,\mathbf{I} + 2\Big(\nu + \nu_{t,\mathrm{eff}}\Big)\mathbf{S}(\overline{\mathbf{u}}) - \boldsymbol{\tau}_{\mathrm{res}}, \qquad \mathbf{F} = -\int_{\Gamma_{\mathrm{wall}}} \boldsymbol{\sigma}\,\mathbf{n}\; d\Gamma, \tag{6.15}$$

with $\mathbf{n}$ the outward normal of the fluid domain, from which the drag and lift coefficients follow as $C_d = 2F_x/(U_\infty^2 D)$ and $C_l = 2F_y/(U_\infty^2 D)$.

Both halves of the closure contribution must appear in Equation (6.15). The implicit half is already accounted for by the eddy viscosity added to the molecular one, and it is the explicit remainder that is easy to omit. Omitting it discards the entire anisotropic part of the predicted stress, which after the split of Equation (6.7) is the larger part of the prediction for every data-driven closure considered here. The artificial viscosity of Equation (6.6) is deliberately excluded from Equation (6.15), since it is a continuation device rather than physics, and it has decayed to zero long before any reported force value.

The shedding period is estimated from the upward zero crossings of the lift coefficient, averaged over a sliding window of the most recent crossings, and is used only for reporting. Two properties of this estimator are worth stating. It is crude, being sensitive to noise and to asymmetric lobes, with no spectral refinement. It must also be re-evaluated continuously rather than frozen once a first estimate becomes available, since crossings produced during the initial transient otherwise contaminate the estimate for the remainder of the run. A window of eight crossings at the shedding periods observed here spans a time comparable with a substantial fraction of the total simulated time, so the estimate produced during a run is a running diagnostic and the values reported in this chapter are computed offline from the complete lift history.

Phase averaging is performed offline for the same reason, from instantaneous snapshots written at a prescribed interval, rather than accumulated during the run. Binning a step into a phase requires the period, and a period estimate that is still contaminated by the transient assigns a large fraction of the run to the wrong phase.

### 6.1.10 Implementation and cost

The solver is parallelized by domain decomposition over distributed-memory processes. The momentum system is factorized directly at every step with the distributed-memory multifrontal solver MUMPS [67, 68], called through PETSc [69]. The transport equation of the $k$-$\ell$ baseline is solved iteratively with the Generalized Minimal Residual (GMRES) method and a block-Jacobi preconditioner, since the

native factorization of PETSc is a serial one and cannot be applied to a matrix distributed over processes, and the filter problems are solved with a conjugate-gradient method.

One closure cannot be run in parallel. The vector-cloud closure of Chapter 4 builds its clouds by a nearest-neighbour search over the cell centroids, and under domain decomposition each process holds only its own partition, so clouds near a partition boundary would be built from the wrong neighbours. The result would be a silently wrong stress rather than an error, so this configuration is rejected explicitly rather than left available.

The cost per step is dominated by the direct factorization of the coupled system, and the remaining operations are arranged so that they do not compete with it. The variational forms, the filter problems and the wall-force functionals are assembled once and reused across steps, with the coefficients that change in time entering as mutable constants, so that the cost per step is one factorization and not one assembly and one factorization. Sampling of fields onto the cell-centred representation used by the closure is performed by interpolation at the centroids rather than by an $L^2$ projection, which for a cell-wise constant target differs at order $h^2$ and avoids assembling and factorizing a mass matrix at every step. On the adopted mesh, with sixteen processes and the three sub-iterations of Section 6.1.2, this gives a cost of about 1.3 seconds per step, of which about 0.1 is spent in the closure and the rest in the three factorizations, stable from the fiftieth step to the last, and a total of about 100 minutes for a computation spanning 120 convective time units.

### 6.1.11 Time-step convergence

The time step is an accuracy parameter rather than a stability one, since the scheme of Equations (6.3) and (6.5) is unconditionally stable, and it therefore requires a convergence study rather than a stability estimate.

## 6.2 The phase-resolved problem

The closure required by Equation (6.1) is a stress field that varies with the phase of the shedding cycle. The tensor-basis closure of Section 3.2 predicts a single, time-independent field, so it cannot supply one without modification. This section fixes what is common to every attempt at modifying it: the data the closures are trained and tested on, the four closures themselves, the criterion a coupled solution has to satisfy, and the two reference points against which it is read.

### 6.2.1 The phase-resolved dataset

The reference data are the same simulations of Table 3.1, processed with the phase average of Section 2.2.3 instead of the time average. The shedding period is divided into 40 phases, giving a phase spacing of $0.125\,D/U_\infty$ and a period of $5\,D/U_\infty$, consistent with a Strouhal number of 0.2. Each phase carries its own mean field and its own Reynolds stress, so the dataset is a sequence of 40 snapshots per Reynolds number rather than a single field.

Two differences with respect to the steady setting of Chapter 3 matter for reading the results below.

- Training uses a single Reynolds number, $\mathrm{Re} = 11\,000$, and testing uses the same held-out $\mathrm{Re} = 140\,000$ as the steady closures. The steady closures were trained on three Reynolds numbers. Every unsteady model reported here therefore extrapolates over a factor of about thirteen in Reynolds number from a single training condition, which is a substantially harder generalization problem than the one Chapter 3 poses.

- The quantity being predicted is different. A phase-averaged stress retains only the stochastic turbulent fluctuations, since the periodic large-scale motion is resolved rather than averaged out. It is therefore smaller in magnitude and differently distributed than the time-averaged stress of Chapter 3, and the numbers of this section are not comparable with those of Table 3.4.

Of the 108 780 cells of the training simulation, 20 000 are retained, with all 40 phases kept for each retained cell, giving 720 000 training and 80 000 validation samples. All the metrics reported below are those of Section 3.1.4, computed on the held-out Reynolds number over the whole period.

### 6.2.2 What a usable unsteady solution has to satisfy

All the diagnostics below are computed on the lift history over a window that excludes the initial transient, and most of them on its coherent part, which is the signal low-pass filtered below $3.5f_0$ with $f_0$ the dominant frequency. Five quantities are used.

- The *peak fraction*, the share of the lift variance falling within $\pm 15\%$ of $f_0$. It measures how much of the signal is the shedding tone.

- The *high-frequency fraction*, written HF in the tables below, the share of the variance above $3.5f_0$. It measures the roughness a noisy closure injects into the force.

- The *envelope modulation* and the *drift*. Let $A(t)$ be the modulus of the analytic signal of the coherent part. Then

$$\mathrm{env}_{cv} = \frac{\sigma(A)}{\langle A \rangle}, \qquad \mathrm{drift} = \frac{\langle A \rangle_{\text{last third}} - \langle A \rangle_{\text{first third}}}{\langle A \rangle}, \tag{6.16}$$

  both zero for a pure sinusoid. The first measures how much the amplitude breathes. The second measures whether the amplitude is still growing or still decaying at the end of the window, and a run with a large drift has not reached saturation within the simulated time.

- The *cycle-to-cycle dispersion*. The coherent signal is folded onto $N_\varphi = 72$ phase bins over $n_c$ consecutive cycles, giving $x_i(\varphi)$ for the $i$-th cycle and the phase mean $\bar{x}(\varphi)$. Then

$$\mathrm{cyc} = \frac{\left\langle \left( x_i(\varphi) - \bar{x}(\varphi) \right)^2 \right\rangle_{i,\varphi}^{1/2}}{\left\langle \bar{x}(\varphi)^2 \right\rangle_{\varphi}^{1/2}}. \tag{6.17}$$

  The folding period is not taken from the spectral peak, whose resolution is the reciprocal of the window length, but refined over a range of $\pm 6\%$ around it by minimizing Equation (6.17) itself, so that a run is not penalized for a period estimate that is half a per cent off.

The last of the five is the one that decides, and this is worth stating because the first four are the natural choices and none of them is sufficient. The peak fraction and the high-frequency fraction do not see an amplitude modulation: when the modulation is slow its sidebands fall inside the $\pm 15\%$ band and are counted as part of the peak. The envelope modulation does not see a loss of periodicity at constant amplitude, since a signal whose waveform or period changes from one cycle to the next can have a perfectly flat envelope. Equation (6.17) is the condition that every cycle be equivalent to every other. It is also exactly the condition that makes a phase average meaningful, because averaging cycles that are not equivalent mixes different states of the flow.

In this work, a Lifting coefficient is considered to be smooth and periodic enough when

$$\mathrm{peak} \geq 80\%, \quad \mathrm{HF} \leq 1\%, \quad \mathrm{env}_{cv} \leq 10\%, \quad |\mathrm{drift}| \leq 10\%, \quad \mathrm{cyc} \leq 10\%. \tag{6.18}$$

Therefore, a run is called admissible if all the condition of 6.18 hold. Figure 6.4 shows why the spectral diagnostics alone are not sufficient.

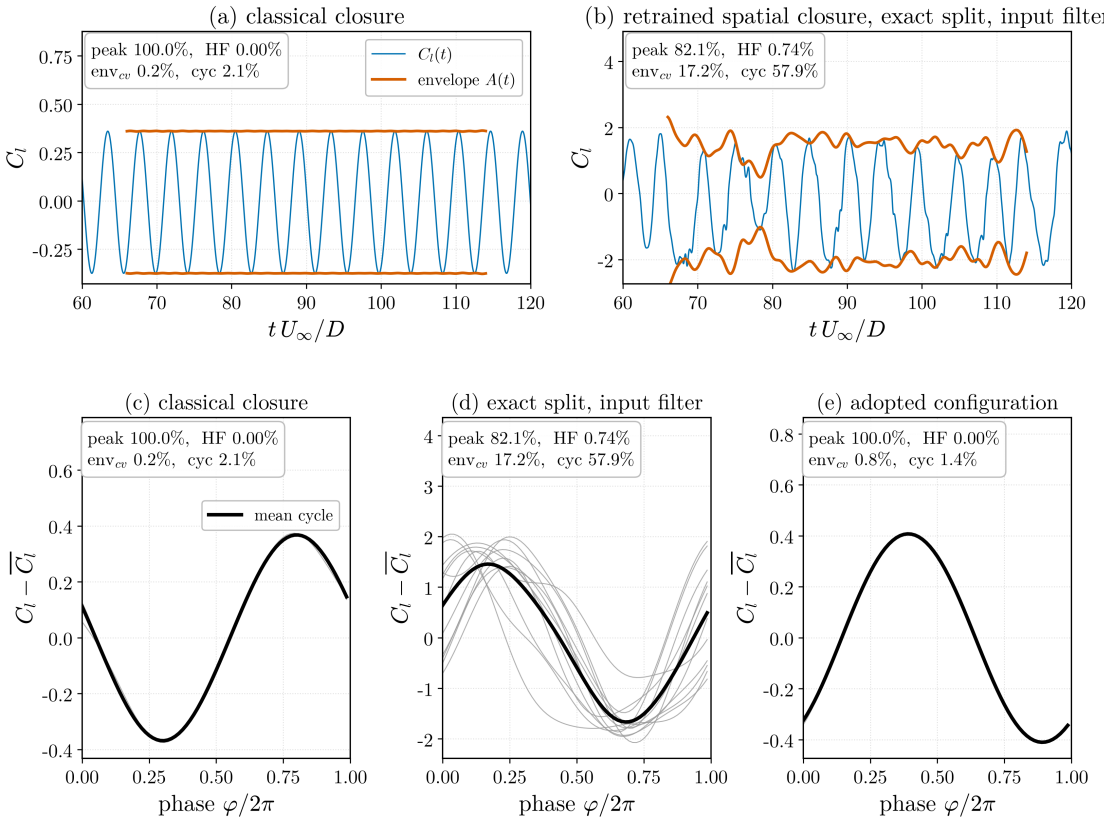



**Figure 6.4:** Why the spectral diagnostics are not sufficient. The control and a closure under the exact split both pass the peak and high-frequency conditions. Folding the two signals onto the shedding period separates them: the cycles of the control collapse onto one curve, those of the closure do not, and the adopted configuration collapses like the control.

### 6.2.3 The reference and the control

Two reference points are needed, and they answer different questions.

The first is the reference flow. The force coefficients are recomputed from the phase-resolved data of Section 6.2.1 by integrating the traction of Equation (6.15) over the cylinder surface at each of the 40 phases, rather than taken from the literature. This gives

$$C_d = 1.120, \qquad C_{l,\mathrm{rms}} = 0.244, \qquad \mathrm{St} = 0.200. \tag{6.19}$$

The integration is checked on quantities whose value is known in advance: the integrated perimeter reproduces $\pi$, the pressure coefficient at the stagnation point is 1.08, and the mean lift over the cycle is $-3 \times 10^{-4}$. One caveat applies to Equation (6.19) and is carried through the rest of this section. The fields are

phase averages, so the lift they produce is the coherent component alone, and the instantaneous lift fluctuation of the reference simulation is larger. Equation (6.19) is the correct target for a solver that resolves the coherent motion and models everything else, which is what Equation (6.1) does, and it is not the total lift fluctuation of the flow.

The second is a control on the numerics. The classical $k$-$\ell$ closure is run on the same mesh, with the same time step, the same time integration and the same stabilization as every closure below. It gives cyc = 2.1%, a high-frequency fraction of 0.00%, an envelope modulation of 0.2% and a drift of 0.0%, with the whole of the variance inside the peak. It is the only run of this chapter that satisfies Equation (6.18), and it satisfies it with a wide margin. The time marching, the mesh and the stabilization are therefore able to produce a clean, periodic limit cycle on this problem, and everything reported below is a property of the closure.

However, $k - \ell$ control is not accurate. Its drag coefficient is 0.678 against 1.120 and its Strouhal number is 0.234 against 0.200, which is expected of an uncalibrated one-equation model on a mesh whose first cell does not lie in the viscous sublayer, as noted in Section 6.1.5. It is used here as a reference for the quality of the cycle and not for its value.

### 6.2.4 The four closures

Four closures are obtained from the steady one by two additions that can be made independently, so that the four form a shared pipeline with two optional stages, Figure 6.5. The first addition supplies the coefficient network with six rate-of-change features, so that two cells with identical instantaneous invariants can be told apart by whether the flow at them is accelerating or decelerating. The second places a recurrent head after the coefficient network, so that the prediction at a phase can depend on the phases before it. The closure with neither addition is the steady architecture retrained on phase-resolved data. The four are presented in the order in which they were built, since each was motivated by what the previous one did not do, and each is presented first against the reference data and then inside the solver. The configuration of each family carried into the solver is identified in its own section, together with the numbers of that network on the reference data, and the sweeps from which it was chosen are collected in Section 6.9.1.

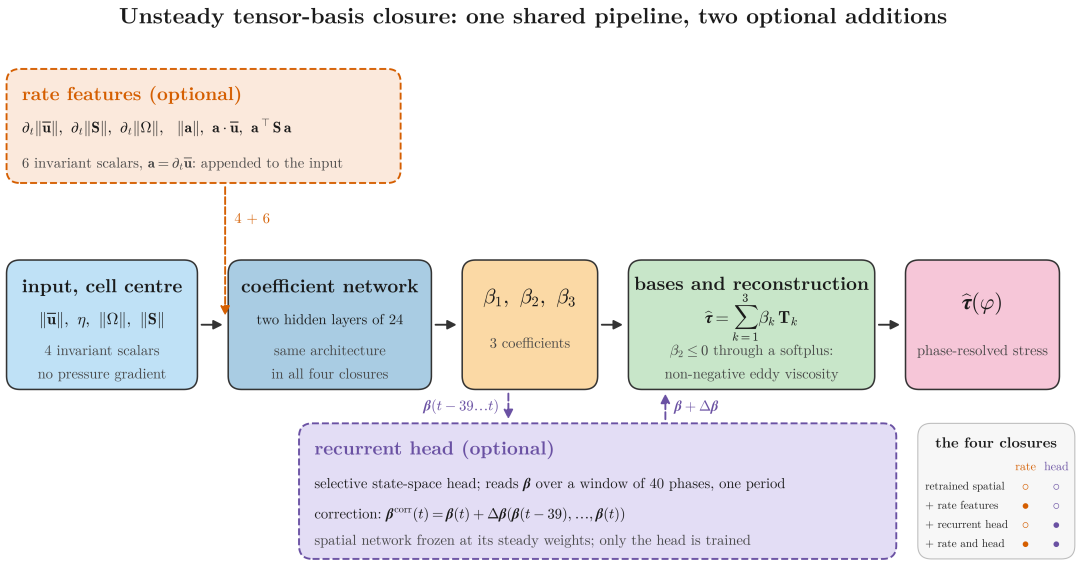



**Figure 6.5:** The four unsteady tensor-basis closures compared in this section, drawn as one shared pipeline with two optional additions. The rate features enter the input of the coefficient network; the recurrent head corrects its output.

## 6.3 Retraining the spatial closure: TBNN for URANS

### 6.3.1 *A priori*

The first attempt changes nothing about the architecture and everything about what it is shown. Each pair (cell, phase) is treated as an independent sample, and the coefficient network is trained from a random initialization on all 40 phases at once. No sequence, no memory, no time index enters anywhere. The resulting closure is unsteady only in the sense that its input varies with the phase.

Two changes to the closure of Section 3.2 were nevertheless required, and both are consequences of results already established in this thesis rather than free choices.

The first is the basis set. The unsteady closure retains only $\mathbf{T}_1$, $\mathbf{T}_2$ and $\mathbf{T}_3$, dropping the two velocity- and pressure-based bases of Section 3.2. In two dimensions the symmetric tensors span a three-dimensional space, so these three bases already saturate the reachable set, and any further basis is a pointwise linear combination of them. This is the redundancy argument of Table 3.7, and it has a specific consequence in the unsteady setting. During temporal marching with a monolitic solver, pressure values can oscillates a lot and they tend to be less

accurate. Introducing a base which depends on it, could alterate the distributions the net learned during training, making the solver less stable. Since stability is the crucial point here, it's been preferred to use Pope's basis only.

The second is the reconstruction. With the reduced basis set, the capped reconstruction of Equation (3.21) is replaced by the variant in which the coefficient multiplying $\mathbf{T}_2$ is forced to be non-positive through a softplus, so that the closure carries an explicit non-negative eddy viscosity. With three bases this reduces exactly to the two-term eddy-viscosity scheme of Equation (3.29). The reason for adopting it here is the one established in Section 3.4.4. Among the local closures, the one carrying an explicit non-negative eddy viscosity is the one that couples most readily, and it reaches the smallest final update of the whole steady campaign. The unsteady solver marches thousands of steps rather than iterating a fixed point, so a dissipative implicit operator matters more there. Consistently with the reduced basis set, the pressure-gradient magnitude is dropped from the input of Equation (3.20), since neither the retained bases nor the reconstruction involves the pressure, leaving four invariant features.

**Training stability.** The configuration is stable at the network size used in Chapter 3, two hidden layers of 24 units, and is not stable at a larger one. Trained at three hidden layers of 64 units with the same reconstruction and the same reduced basis set, the model collapses: the aggregate error reaches 1.335, above unity and therefore worse than predicting a uniformly zero stress, with negative coefficients of determination on all three components. The two configurations differ only in the size of the coefficient network. A plausible mechanism is the saturation of the softplus that enforces the eddy-viscosity sign: once it saturates, the gradient reaching the coefficient network through that path vanishes, and the composite loss admits a nearby minimum in which the prediction is almost constant in space. This reading is not isolated by a dedicated experiment and is offered as such. What the two runs do establish is that additional capacity does not help this configuration, so the small network is not a compromise here but the condition for training at all.

**Results.** The reference point for these numbers is not the steady closure of Table 3.4, which predicts a different quantity, but the same steady closure evaluated directly on phase-resolved data. Table 6.2 reports both, on the held-out Reynolds number.

The steady closure evaluated as is gives an aggregate error of 1.065 and negative coefficients of determination on all three components. A closure trained on time-averaged statistics is therefore not merely less accurate on phase-resolved data, it is worse than predicting nothing. This is totally expected since the turbulence closure in general needs to predict different fluctuations, depending on the steady

**Table 6.2:** Steady tensor-basis closure evaluated as is on phase-resolved data, and the same architecture retrained on phase-resolved data in the configuration carried into the solver, held-out Re $= 140\,000$, with $\rho$ the correlation with the true field.

| Closure | $R^2_{\overline{u'u'}}$ | $R^2_{\overline{v'v'}}$ | $R^2_{\overline{u'v'}}$ | $\rho(\overline{u'v'})$ | $\varepsilon_{L^2}(\overline{u'v'})$ | $\varepsilon^{\text{tensor}}_{L^2}$ |
|---|---|---|---|---|---|---|
| Steady closure as is | $-1.888$ | $-1.080$ | $-0.473$ | 0.628 | 1.788 | 1.065 |
| Adopted | 0.623 | 0.603 | 0.421 | 0.651 | 0.787 | 0.464 |

or unsteady nature of the simulation. Retraining the same architecture on phase-resolved data takes the aggregate error from 1.065 to 0.464 and every coefficient of determination above 0.4. The configuration carried into the solver is the one with the highest coefficient of determination on the shear stress across the learning-rate sweep of Table 6.17, rather than the one with the lowest aggregate error, which reaches 0.448 at a shear-stress coefficient of 0.284. The two metrics do not agree on which configuration is best, which is the same disagreement documented throughout Chapter 3 and has the same cause, and the shear stress is preferred here because it is the component that transports momentum across the wake. Whether it is also the component that decides the behaviour inside the solver is examined in Section 6.7.2.

### 6.3.2 In the solver

The retrained spatial closure is marched with the numerical setting of Section 6.1 in the configuration that Section 6.9.3 identifies as the best for it: the split of the closure term in the hyperviscous form of the steady solver, and the Helmholtz filter of Section 6.1.8 at $2h$ on both the features and the predicted stress. Table 6.3 reports the diagnostics of Section 6.2.2 on the window of 120 convective time units used throughout, and on the next 60 units of a continuation of the same run, and the phase-averaged errors of the coupled solution against the reference over the near wake. The closure does not satisfy Equation (6.18) on either window, and it fails on one condition only. The lift is spectrally as clean as the control produces, with nothing above $3.5f_0$ and all the variance inside the shedding peak, but its amplitude is still growing at the end of the continuation, with a drift of $+36\%$ on the last window, and it has doubled from one window to the next. The oscillation is therefore a shedding mode that is still emerging at $t = 180$, not a limit cycle at low amplitude, and no statement about its saturated state can be made from these runs. Its frequency is within seven per cent of the reference and its phase-averaged fields, read on a mode that has not saturated, are already within six per cent on the velocity magnitude and within 0.52 on the shear stress. Figure 6.6 shows the lift from the start of the march, and Figure 6.7 and Figure 6.8 show the force

coefficients over the cycle and the fields at two opposite phases. The lift curve of the figure lies close to the reference one, and this is an artefact of the phase averaging: an amplitude growing from 0.18 to 0.26 over the window is averaged to a value near the reference 0.24, and the mean drag, 0.97 against 1.12, is the quantity that shows the wake is not yet developed.

**Table 6.3:** The retrained spatial closure in the solver, hyperviscous split, both filters. Upper block: diagnostics of Section 6.2.2 on the two windows, with the control and the reference of Section 6.2.3. Lower block: relative $L^2$ error of the phase-averaged coupled solution against the reference, over the near wake $x \in [-2,8]D$, $y \in [-3,3]D$, averaged over the phases.

| Window | cyc | HF | peak | $\mathrm{env}_{cv}$ | drift | $C_{l,\mathrm{rms}}$ | St |
|---|---|---|---|---|---|---|---|
| $t \in [60,120]$ | 25.2% | 0.00% | 99.5% | 20.2% | +46.4% | 0.122 | 0.179 |
| $t \in [120,180]$ | 20.8% | 0.00% | 99.5% | 16.0% | +36.0% | 0.223 | 0.187 |
| $k$-$\ell$ control | 2.1% | 0.00% | 100% | 0.2% | +0.0% | 0.260 | 0.234 |
| Reference | — | — | — | — | — | 0.244 | 0.200 |

| Phase-averaged error, near wake | $\lvert\overline{\mathbf{u}}\rvert$ | $\overline{u'u'}$ | $\overline{v'v'}$ | $\overline{u'v'}$ | $k$ |
|---|---|---|---|---|---|
| $t \in [130,180]$, mean over the 40 phases | 0.061 | 0.413 | 0.371 | 0.515 | 0.270 |

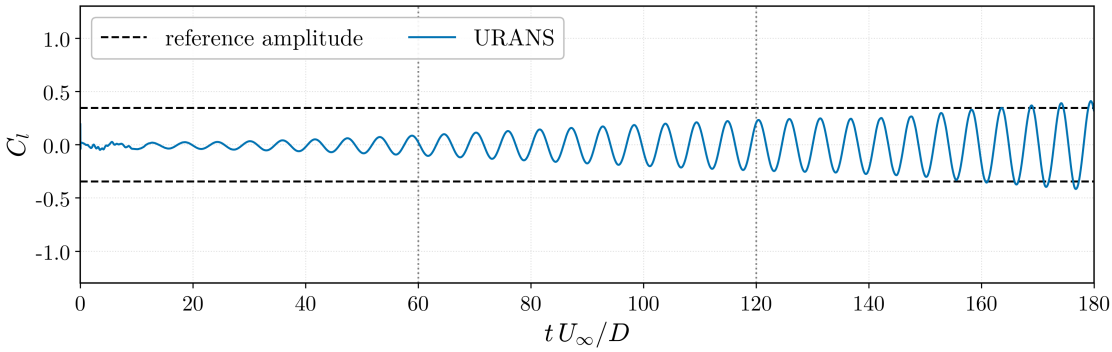



**Figure 6.6:** Retrained spatial closure: lift coefficient from the start of the march to the end of the continuation. The dashed lines mark the amplitude of the reference, $\pm\sqrt{2}\,C_{l,\mathrm{rms}}$, and the dotted lines the start of the evaluation window and of the continuation.

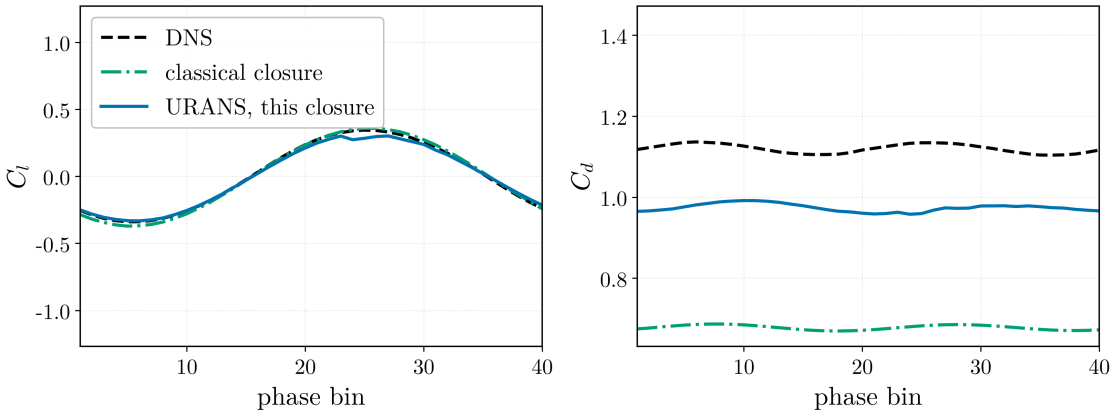



**Figure 6.7:** Retrained spatial closure: lift and drag coefficients over one shedding period, phase average of the coupled solution over $t \in [130,180]$ against the reference and against the classical closure on the same window, after phase alignment of each to the reference.

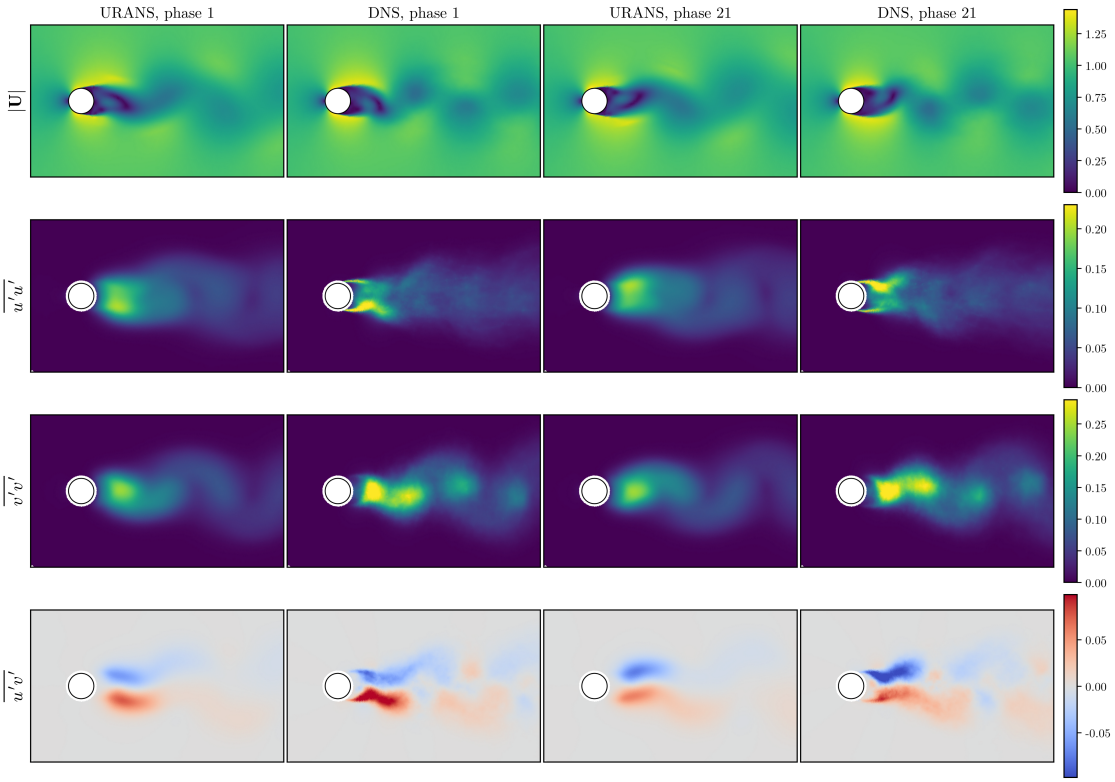



**Figure 6.8:** Retrained spatial closure: velocity magnitude and shear stress over the near wake at two opposite phases of the cycle, coupled solution against the reference at the same phase.

## 6.4 Rate-of-change features: RateTBNN

### 6.4.1 *A priori*

The closure above sees no time. Two cells with identical instantaneous invariants receive identical predictions, whether the flow at one is accelerating and at the other decelerating. The second attempt supplies that information as additional scalar inputs to the coefficient network.

Six features are added. The first three are rates of change of quantities that are already invariant scalars,

$$\dot{q} = \frac{\partial q}{\partial t}, \qquad q \in \Big\{ \|\overline{\mathbf{u}}\|,\ \|\mathbf{S}\|,\ \|\mathbf{\Omega}\| \Big\}, \tag{6.20}$$

and the remaining three are built from the mean acceleration $\mathbf{a} = \partial\overline{\mathbf{u}}/\partial t$,

$$\|\mathbf{a}\|, \qquad \mathbf{a}\cdot\overline{\mathbf{u}}, \qquad \mathbf{a}^{\top}\mathbf{S}\,\mathbf{a}. \tag{6.21}$$

All six are invariant under rotations and under reflections of the frame. The first three are so because they are time derivatives of invariant scalars, and the last three because they are, respectively, a norm, an inner product of two true vectors, and a vector–tensor–vector contraction. Feeding the components of $\mathbf{a}$ directly would instead destroy the equivariance the tensor-basis construction exists to guarantee, which is why no raw component appears among them. During training the derivatives are evaluated by a central difference over the cyclic phase sequence, which is available because the reference data cover a full period.

**Results.** Table 6.4 reports the configuration carried into the solver. It does not differ from the closure of Section 6.3 by the rate features alone: the network carried forward was retrained after the learning-rate sweep of Table 6.18, with the loss weighted by cell area and written on the three stress components separately, and it is the best of that retraining on the aggregate error. The single-variable comparison, the six features added to an otherwise identical network at equal learning rate, is the one between Table 6.17 and Table 6.18 in Section 6.9.1. Its outcome is that on the aggregate error the six features buy nothing, that they improve the cross-stream normal stress at every learning rate, and that they leave the shear stress unresolved, better at some learning rates and worse at others. The network carried forward is consistent with that reading. Against the retrained spatial closure of Table 6.2 it gains on both normal stresses and loses on the shear stress, from 0.421 to 0.266. Globally, these *a priori* metrics are comparable, so one might think that adding temporal rates does not help the net. However, too few experiments are made to make this conclusion. The reason might be in the few data available for training or in the specific rates chosen here.

**Table 6.4:** Tensor-basis closure of Section 6.3 compared with the same closure but with the six rate features of Equations (6.20)–(6.21). Metrics on held-out $\mathrm{Re} = 140\,000$.

| Closure | $R^2_{\overline{u'u'}}$ | $R^2_{\overline{v'v'}}$ | $R^2_{\overline{u'v'}}$ | $\rho(\overline{u'v'})$ | $\varepsilon_{L^2}(\overline{u'v'})$ | $\varepsilon^{\mathrm{tensor}}_{L^2}$ |
|---|---|---|---|---|---|---|
| TBNN | 0.623 | 0.603 | 0.421 | 0.651 | 0.787 | 0.464 |
| TBNN + Temporal Features | 0.665 | 0.703 | 0.266 | 0.608 | 0.809 | 0.436 |

### 6.4.2 In the solver

The closure with the rate features is marched in the same configuration, hyperviscous split and both filters, which is the one under which it saturates; with the output filter alone it also passes the criterion on the first window but its amplitude is still drifting on the second, as Section 6.9.3 reports. Table 6.5 shows a limit cycle that satisfies Equation (6.18) on both windows, with a cycle-to-cycle dispersion of about four per cent, an envelope modulation of two per cent and no drift. The frequency is within one per cent of the reference. The amplitude is not: the lift fluctuation is 3.5 times the reference one, and the phase-averaged fields carry the largest errors of the four closures, ten per cent on the velocity magnitude and 0.64 on the shear stress. The mean drag is 1.30 against 1.12, and it oscillates at twice the shedding frequency with an amplitude of 0.3 where the reference oscillates by 0.015. The closure therefore sustains a clean and periodic cycle at the right frequency and the wrong amplitude, and Section 6.8 traces the amplitude to what this closure learns and what it does not. Figure 6.9 shows the lift from the start of the march, and Figure 6.10 and Figure 6.11 show the force coefficients and the fields.

**Table 6.5:** The closure with the rate features in the solver, hyperviscous split, both filters. Blocks as in Table 6.3.

| Window | cyc | HF | peak | $\mathrm{env}_{cv}$ | drift | $C_{l,\mathrm{rms}}$ | St |
|---|---|---|---|---|---|---|---|
| $t \in [60,120]$ | 4.4% | 0.00% | 99.9% | 2.1% | +1.2% | 0.850 | 0.198 |
| $t \in [120,180]$ | 4.3% | 0.00% | 99.8% | 1.9% | +0.0% | 0.844 | 0.198 |
| $k$-$\ell$ control | 2.1% | 0.00% | 100% | 0.2% | +0.0% | 0.260 | 0.234 |
| Reference | — | — | — | — | — | 0.244 | 0.200 |

| Phase-averaged error, near wake | $\lvert\overline{\mathbf{u}}\rvert$ | $\overline{u'u'}$ | $\overline{v'v'}$ | $\overline{u'v'}$ | $k$ |
|---|---|---|---|---|---|
| $t \in [130,180]$, mean over the 40 phases | 0.104 | 0.471 | 0.422 | 0.642 | 0.395 |

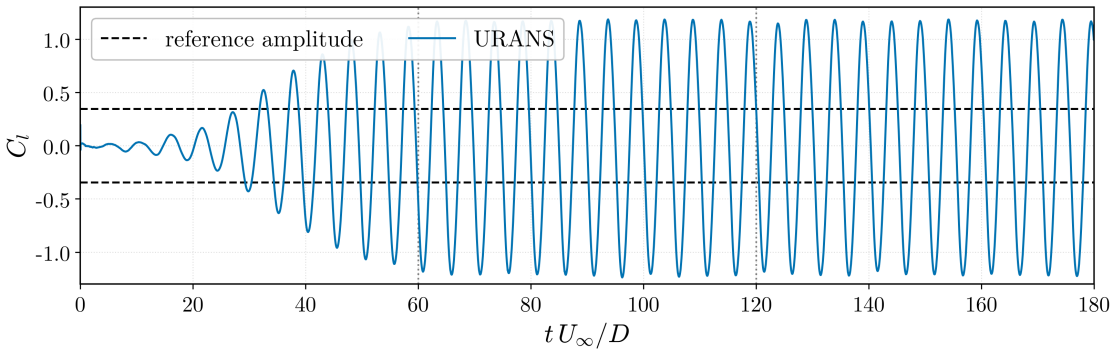



**Figure 6.9:** Closure with the rate features: lift coefficient from the start of the march to the end of the continuation. The dashed lines mark the amplitude of the reference, $\pm\sqrt{2}\,C_{l,\mathrm{rms}}$, and the dotted lines the start of the evaluation window and of the continuation.

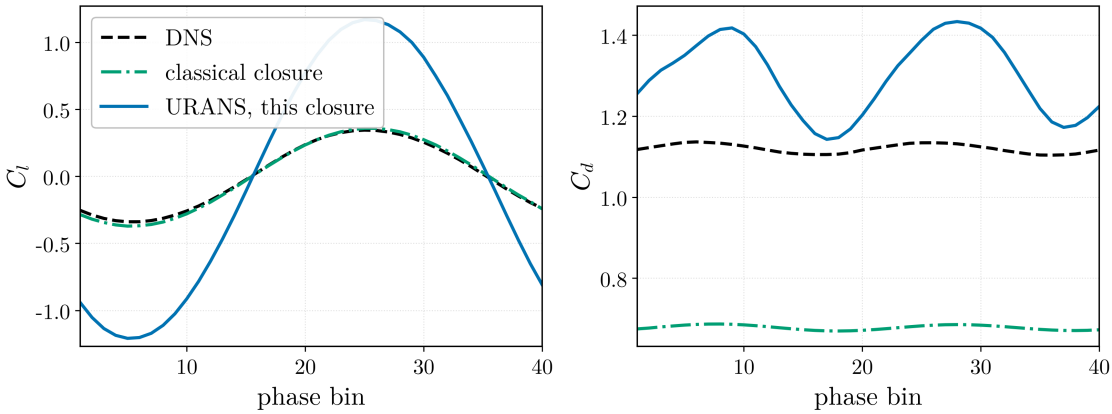



**Figure 6.10:** Closure with the rate features: lift and drag coefficients over one shedding period against the reference, as in Figure 6.7.

## 6.5 A recurrent head: TBNN-MAMBA

### 6.5.1 *A priori*

The third attempt introduces memory. A selective state-space head of the family described in Section 2.3 is placed after the coefficient network. It receives the sequence of coefficient vectors produced by the spatial network over a window of 40 phases, one full period, and returns a correction,

$$\boldsymbol{\beta}^{\mathrm{corr}}(t) = \boldsymbol{\beta}(t) + \Delta\boldsymbol{\beta}\Big(\boldsymbol{\beta}(t-39),\ \ldots,\ \boldsymbol{\beta}(t)\Big), \tag{6.22}$$

which is reconstructed into a stress through exactly the same bases and the same reconstruction as before. The spatial network is the steady closure of Chapter 3,

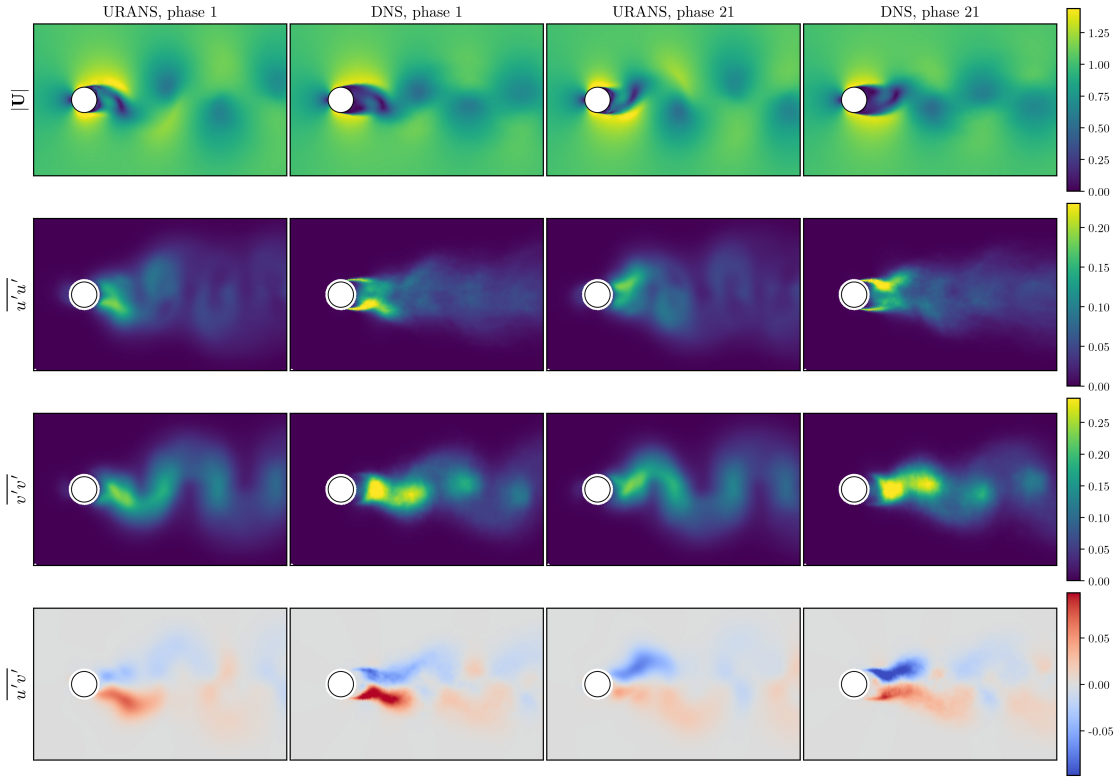



**Figure 6.11:** Closure with the rate features: velocity magnitude and shear stress over the near wake at two opposite phases, against the reference.

with its weights frozen, so the head is the only trained component. This is the arrangement that motivates the whole attempt: it asks whether a memory mechanism can adapt a closure trained on time-averaged statistics to phase-resolved data without retraining the closure itself.

**The head recovers what retraining the spatial network already recovers.** Table 6.6 reports the result. The frozen spatial network alone stands at an aggregate error of 1.065. Adding the head takes it to 0.472. Retraining the spatial network on the same data without any head, Section 6.3, takes it to 0.464. The two routes land within two per cent of each other, and the one without memory is marginally ahead. Adding a memory mechanism on top of a frozen steady backbone is therefore, to the accuracy this comparison resolves, the same thing as retraining the backbone on unsteady data. The configuration carried into the solver is the best of the learning-rate sweep of Table 6.19 on every metric at once.

**Table 6.6:** Recurrent head on a frozen steady spatial network, receiving the coefficient sequence only, configuration carried into the solver, held-out Re = 140 000.

| Closure | $R^2_{\overline{u'u'}}$ | $R^2_{\overline{v'v'}}$ | $R^2_{\overline{u'v'}}$ | $\rho(\overline{u'v'})$ | $\varepsilon_{L^2}(\overline{u'v'})$ | $\varepsilon^{\text{tensor}}_{L^2}$ |
|---|---|---|---|---|---|---|
| Steady closure as is | −1.888 | −1.080 | −0.473 | 0.628 | 1.788 | 1.065 |
| Adopted | 0.688 | 0.696 | 0.441 | 0.730 | 0.784 | 0.472 |

**A conjecture for why.** The head of Equation (6.22) reads a sequence whose every element is a deterministic function of the instantaneous invariant features at that cell and phase, since the frozen spatial network is memoryless. If the map from those instantaneous features to the true phase-averaged stress were single-valued, then a correction that depends only on the current element of the sequence would already be sufficient, and the memory would have nothing left to contribute. The head would then be spending its capacity learning a static re-mapping of the coefficients, which is exactly the function that retraining the spatial network learns directly. The measured equivalence between the two routes is what that situation predicts.

The conjecture makes a testable prediction: giving the head an input that is *not* a function of the instantaneous invariant features should break the equivalence, whereas any amount of additional memory over the same input should not. The rate features of Section 6.4.1 are exactly such an input, since a time derivative at a phase is not recoverable from the invariants at that phase alone.

### 6.5.2 In the solver

The recurrent closure is marched with the hyperviscous split and the output filter alone, the configuration under which it does best. Table 6.7 shows the cleanest limit cycle of this chapter. Its cycle-to-cycle dispersion on the first window, 1.4%, is lower than that of the classical control, all the lift variance is inside the shedding peak, the envelope modulation is below one per cent and the drift is zero, and the continuation confirms it, with a dispersion of 3.7% on the second window and the same amplitude and frequency to three digits. The Strouhal number is 0.199 against 0.200, where the control gives 0.234, and the lift amplitude is 0.29 against 0.24, a fifth too large. The phase-averaged fields are the most accurate of the four closures on the velocity magnitude, 0.044, and on the shear stress, 0.44, and the error moves by less than two hundredths across the phases of the cycle, so the accuracy is uniform along it rather than concentrated at one phase. The mean drag is 1.07 against 1.12. The closure that under the exact split did not shed at all, Section 6.9.3, is under the adopted split the one that reproduces the reference

cycle most closely. Figure 6.12 shows the lift from the start of the march, and Figure 6.13 and Figure 6.14 show the force coefficients and the fields.

**Table 6.7:** The recurrent closure in the solver, hyperviscous split, output filter. Blocks as in Table 6.3.

| Window | cyc | HF | peak | $\mathrm{env}_{cv}$ | drift | $C_{l,\mathrm{rms}}$ | St |
|---|---|---|---|---|---|---|---|
| $t \in [60,120]$ | 1.4% | 0.00% | 100% | 0.8% | −0.0% | 0.291 | 0.199 |
| $t \in [120,180]$ | 3.7% | 0.00% | 100% | 2.1% | −3.7% | 0.295 | 0.199 |
| $k$-$\ell$ control | 2.1% | 0.00% | 100% | 0.2% | +0.0% | 0.260 | 0.234 |
| Reference | — | — | — | — | — | 0.244 | 0.200 |

| Phase-averaged error, near wake | $\lvert\overline{\mathbf{u}}\rvert$ | $\overline{u'u'}$ | $\overline{v'v'}$ | $\overline{u'v'}$ | $k$ |
|---|---|---|---|---|---|
| $t \in [130,180]$, mean over the 40 phases | 0.044 | 0.322 | 0.320 | 0.439 | 0.242 |

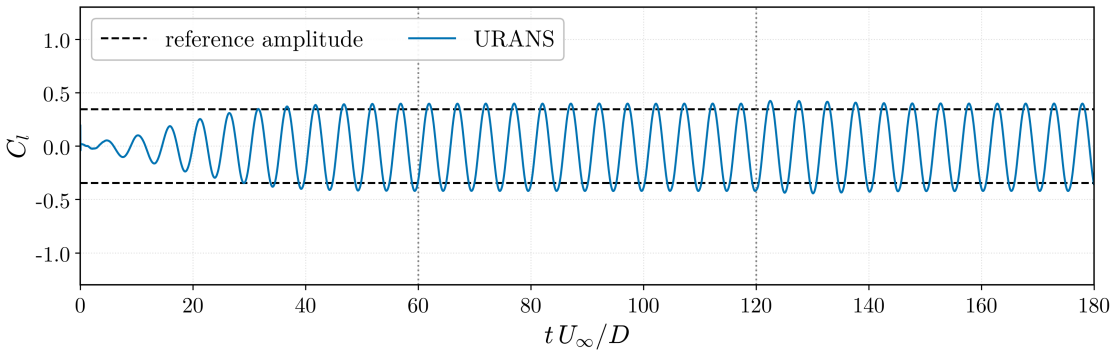



**Figure 6.12:** Recurrent closure: lift coefficient from the start of the march to the end of the continuation. The dashed lines mark the amplitude of the reference, $\pm\sqrt{2}\,C_{l,\mathrm{rms}}$, and the dotted lines the start of the evaluation window and of the continuation.

## 6.6 A recurrent head with rate features: RateTBNN-MAMBA

### 6.6.1 *A priori*

The fourth closure combines the two additions. The head of Section 6.5 receives, besides the coefficient sequence, the local features and the six rate features of Section 6.4. The conjecture of Section 6.5 predicts that this input, and not any

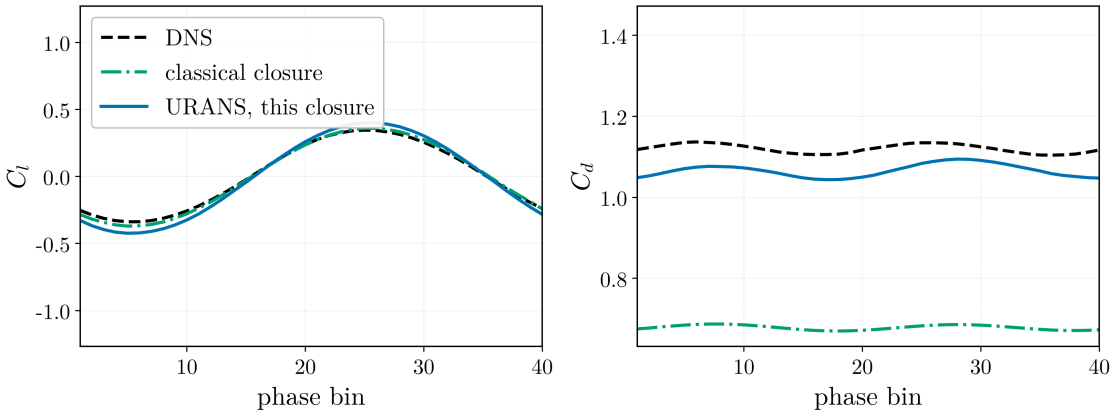



**Figure 6.13:** Recurrent closure: lift and drag coefficients over one shedding period against the reference, as in Figure 6.7.

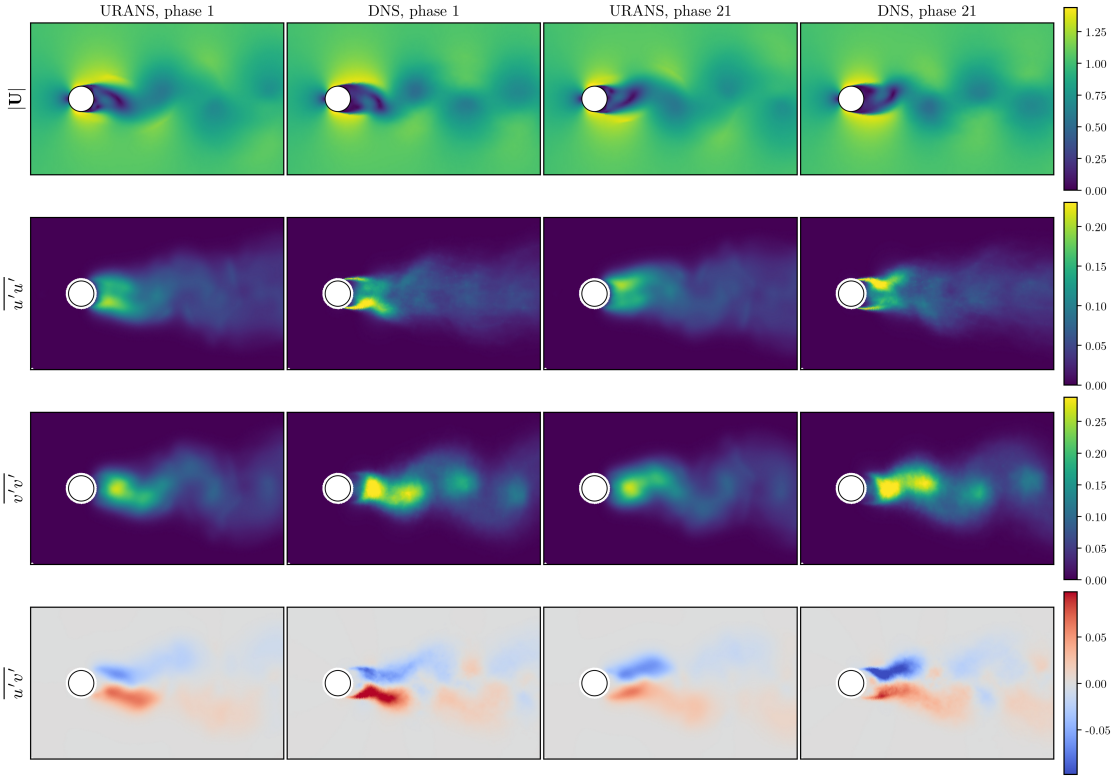



**Figure 6.14:** Recurrent closure: velocity magnitude and shear stress over the near wake at two opposite phases, against the reference.

further memory over the same input, is what should break the equivalence between the head and retraining.

**The prediction holds.** The lower block of Table 6.19 feeds the head the local features together with the six rate features. The aggregate error drops to 0.424, against 0.472 for the same head without them and 0.448 for the retrained spatial network of the same sweep. This is an improvement of about ten per cent over the head alone and of about five per cent over the best configuration of either of the two preceding families, and the improvement is monotone across the sweep, which none of the other three families shows. The result is therefore an interaction rather than an effect of either ingredient. Rate features without memory, Section 6.4.1, give nothing. Memory without rate features gives nothing beyond retraining. The two together give about ten per cent. This is consistent with the conjecture above: the memory becomes useful only once the sequence it reads carries information that the instantaneous prediction does not already contain.

**The cost is on the shear stress, again.** The same trade-off observed in Section 6.4.1 appears here amplified. Adding the rate features to the head takes the streamwise normal stress from 0.688 to 0.718 and the cross-stream one from 0.696 to 0.737, and takes the shear stress from 0.441 to 0.131, with one configuration turning it negative. Together with the partial evidence of Section 6.4.1, this establishes that the rate features improve the normal stresses, and that they degrade the shear stress whenever a memory mechanism is present to exploit them across the cycle. The aggregate error improves regardless, because the normal stresses carry most of the magnitude of the tensor, but a closure whose shear-stress coefficient of determination is 0.13 is producing the component that drives the momentum balance almost by chance. Whether this closure is preferable to the one of Section 6.3.1 therefore depends on the criterion, and the two candidates are carried forward together rather than resolved here.

**The configuration carried into the solver.** The network carried forward, Table 6.8, belongs to a second sweep of the same head, Table 6.20, in which the pressure-gradient norm is removed from the head input so that the closure is pressure-free on both sides, as the steady one is. It is the best of that sweep on the aggregate error, at 0.422, and it carries the same trade as the first sweep: the two normal stresses close to those of the head alone, and the shear stress well below, at 0.168 against 0.441.

**A caveat on the deployment.** The head is trained on windows of 40 phases starting from a zero internal state, whereas a solver marching in time never resets

**Table 6.8:** Recurrent head receiving the coefficient sequence, the local features and the six rate features, configuration carried into the solver, held-out Re = 140 000.

| Closure | $R^2_{\overline{u'u'}}$ | $R^2_{\overline{v'v'}}$ | $R^2_{\overline{u'v'}}$ | $\rho(\overline{u'v'})$ | $\varepsilon_{L^2}(\overline{u'v'})$ | $\varepsilon^{\text{tensor}}_{L^2}$ |
|---|---|---|---|---|---|---|
| Steady closure as is | −1.888 | −1.080 | −0.473 | 0.628 | 1.788 | 1.065 |
| Adopted | 0.674 | 0.675 | 0.168 | 0.617 | 0.807 | 0.422 |

that state. The configuration in which this closure is deployed is therefore outside the distribution the head was trained on, by construction, and this is not resolvable with the runs available.

### 6.6.2 In the solver

The closure with both additions is marched with the hyperviscous split and the output filter alone. Table 6.9 shows a limit cycle that satisfies Equation (6.18) on both windows, with a dispersion of about five per cent on the first and two per cent on the second, an envelope modulation of two to three per cent and no drift. This is the closure that reproduces the amplitude of the reference: the lift fluctuation is 0.237 against 0.244, three per cent low, where the control gives 0.260 and the recurrent closure without the rate features gives 0.291. The frequency is within four per cent and the mean drag is 1.04 against 1.12. The phase-averaged fields are as accurate as those of the recurrent closure on the velocity magnitude and on the shear stress, and better on the cross-stream normal stress, 0.24 against 0.32. The deployment caveat of Section 6.6.1 applies to this run as to the previous one, and neither shows the consequence it was expected to have: the recurrent state is never reset during the march, and the cycle it produces is periodic and stable over 120 time units. Figure 6.15 shows the lift from the start of the march, and Figure 6.16 and Figure 6.17 show the force coefficients and the fields.

## 6.7 The four closures side by side

### 6.7.1 *A priori*

Table 6.10 collects the four closures carried into the solver, component by component. These are the numbers of the four networks actually used, and not the best of each family in the sweeps of Section 6.9.1.

Three findings follow.

- A closure trained on time-averaged statistics is unusable on phase-resolved data, with an aggregate error above unity and negative coefficients of determination

**Table 6.9:** The closure with the rate features and the recurrent head in the solver, hyperviscous split, output filter. Blocks as in Table 6.3.

| Window | cyc | HF | peak | $\mathrm{env}_{cv}$ | drift | $C_{l,\mathrm{rms}}$ | St |
|---|---|---|---|---|---|---|---|
| $t \in [60,120]$ | 4.8% | 0.00% | 98.8% | 2.2% | +0.4% | 0.237 | 0.193 |
| $t \in [120,180]$ | 2.2% | 0.00% | 98.7% | 3.4% | +0.0% | 0.235 | 0.193 |
| $k$-$\ell$ control | 2.1% | 0.00% | 100% | 0.2% | +0.0% | 0.260 | 0.234 |
| Reference | — | — | — | — | — | 0.244 | 0.200 |

| Phase-averaged error, near wake | $\lvert\overline{\mathbf{u}}\rvert$ | $\overline{u'u'}$ | $\overline{v'v'}$ | $\overline{u'v'}$ | $k$ |
|---|---|---|---|---|---|
| $t \in [130,180]$, mean over the 40 phases | 0.041 | 0.327 | 0.243 | 0.437 | 0.240 |

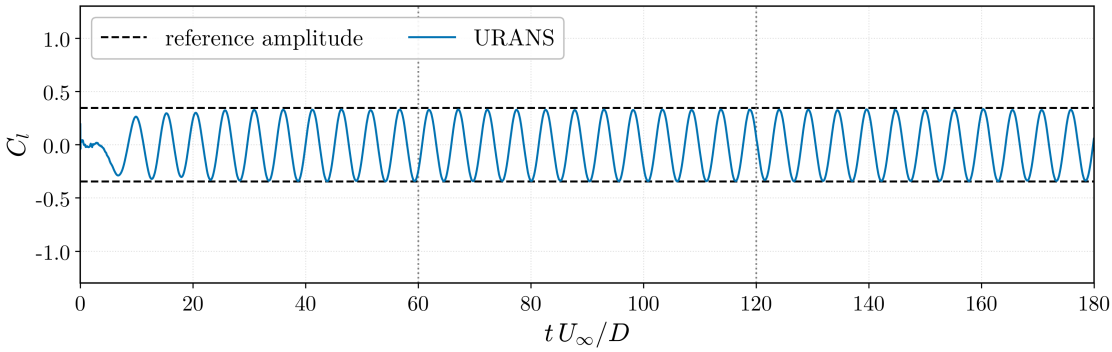



**Figure 6.15:** Recurrent closure with the rate features: lift coefficient from the start of the march to the end of the continuation. The dashed lines mark the amplitude of the reference, $\pm\sqrt{2}\,C_{l,\mathrm{rms}}$, and the dotted lines the start of the evaluation window and of the continuation.

on every component. Some form of unsteady training is not an improvement but a requirement.

- Which form of unsteady training is used barely matters on the aggregate error. Retraining the spatial network, or freezing it and adding a memory mechanism, land within two per cent of each other, and the second is not the better of the two. This is the equivalence conjectured and tested in Section 6.5.
- What does matter is supplying information the instantaneous invariants do not contain. The rate features are without effect on their own, and give about ten per cent once a memory mechanism is present to use them across the cycle. Their cost, once that memory mechanism is present, is a marked loss of accuracy on the shear stress, which is reported here as an open trade-off rather than as a settled preference.

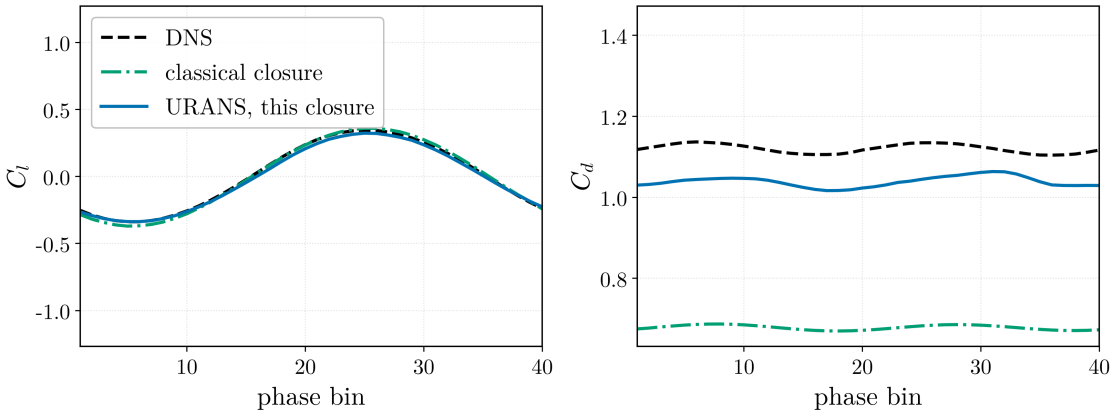



**Figure 6.16:** Recurrent closure with the rate features: lift and drag coefficients over one shedding period against the reference, as in Figure 6.7.

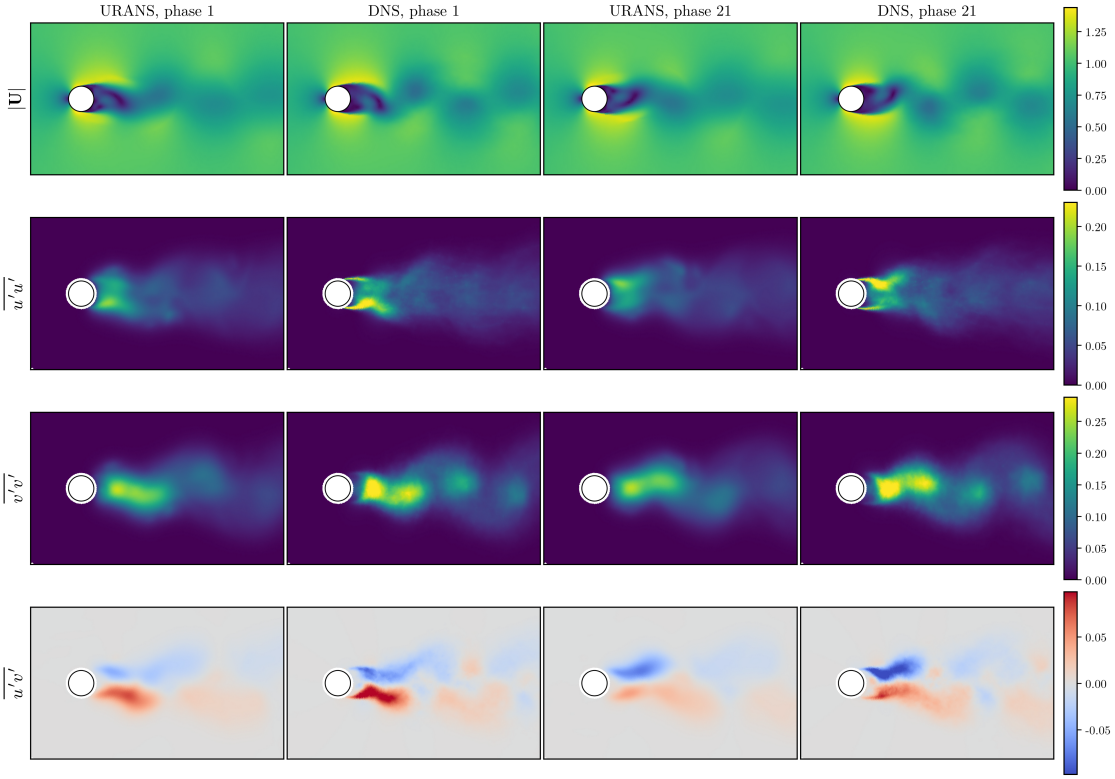



**Figure 6.17:** Recurrent closure with the rate features: velocity magnitude and shear stress over the near wake at two opposite phases, against the reference.

**Table 6.10:** *A priori* accuracy of the four closures deployed in the solver, held-out $\mathrm{Re} = 140\,000$, with $\rho$ the correlation with the true field. The shear stress is the worst-predicted component of every one of them.

| Closure | $R^2_{\overline{u'u'}}$ | $R^2_{\overline{v'v'}}$ | $R^2_{\overline{u'v'}}$ | $\rho(\overline{u'v'})$ | $\varepsilon_{L^2}(\overline{u'v'})$ | $\varepsilon^{\mathrm{tensor}}_{L^2}$ |
|---|---|---|---|---|---|---|
| Retrained spatial | 0.623 | 0.603 | 0.421 | 0.651 | 0.787 | 0.464 |
| + rate features | 0.665 | 0.703 | 0.266 | 0.608 | 0.809 | 0.436 |
| + recurrent head | 0.688 | 0.696 | 0.441 | 0.730 | 0.784 | 0.472 |
| + rate features and head | 0.674 | 0.675 | 0.168 | 0.617 | 0.807 | 0.422 |

The spread between the four families, from 0.422 to 0.472, is about eleven per cent. The gap between all four and the steady closure evaluated as is, a factor of 2.3, is an order of magnitude larger. The choice among unsteady training strategies is therefore a second-order decision, and the first-order one is simply to train on phase-resolved data at all. The shear stress is the component every one of the four predicts worst, with a coefficient of determination between 0.17 and 0.44 where the two normal components sit between 0.60 and 0.70; Section 6.8 returns to it. Figure 6.18 shows the training histories of the four networks, Figure 6.19 the scatter of the recurrent closure, Figure 6.20 the accuracy phase by phase, and Figure 6.21 what the recurrent head adds to the frozen backbone on the shear stress.

### 6.7.2 In the solver

Table 6.11 collects the four closures in their adopted configurations, that is the first rows of Tables 6.3 to 6.9, with the phase-averaged errors alongside. Three of the four satisfy Equation (6.18), and the fourth fails it only because its amplitude has not saturated within the simulated time. The classical closure is given the same treatment, so that the data-driven closures are read against it and not only against the reference.

Five statements are supported by Table 6.11.

**The classical closure reproduces the shedding and not the turbulence.** Its lift is periodic and has the amplitude of the reference, and Figures 6.7 to 6.16 show its lift curve on top of the reference one once folded on its own period; its frequency is 17% too high. Its mean drag is 0.68 against 1.12, and its phase-averaged stresses carry errors between 0.78 and 0.92 over the near wake, one and a half to two times those of the worst data-driven closure and two to three and a half times those of the best. The three data-driven closures that couple are therefore better than the control on every stress component, and the two with memory are better on the velocity and on the drag as well.

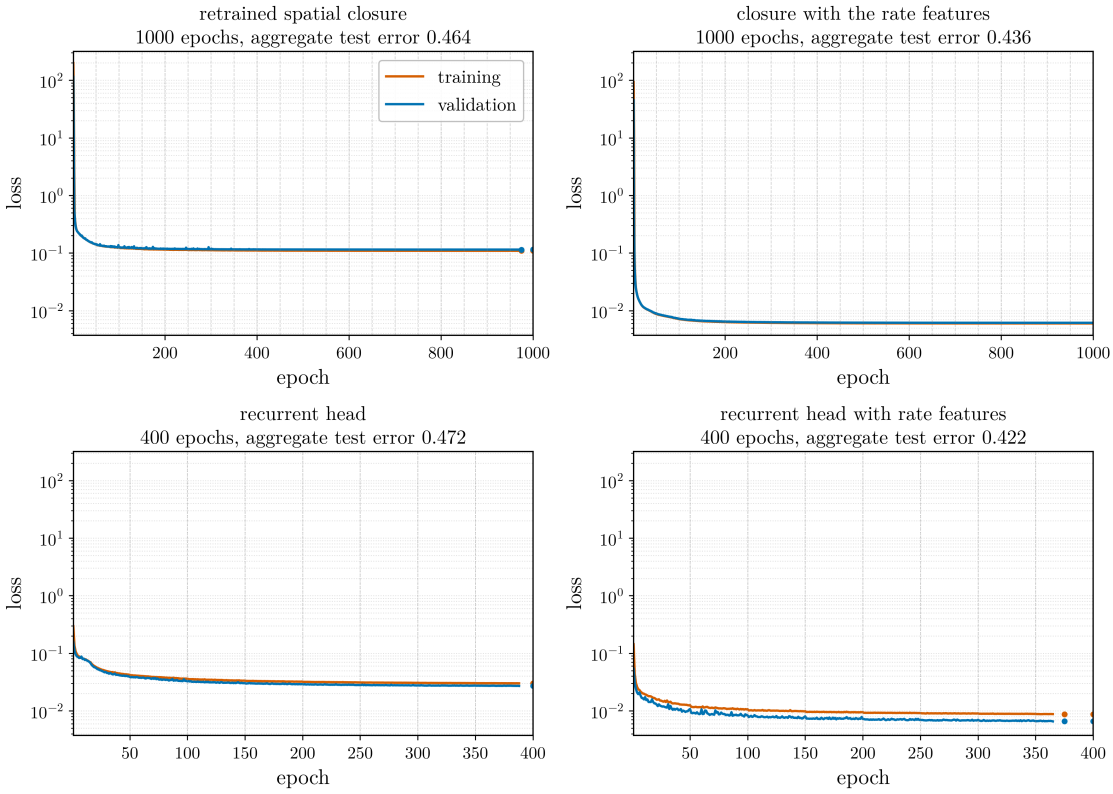



**Figure 6.18:** Training and validation loss of the four unsteady tensor-basis families, at the best learning rate of each. The two families carrying a recurrent head train for fewer epochs, since only the head is trained.

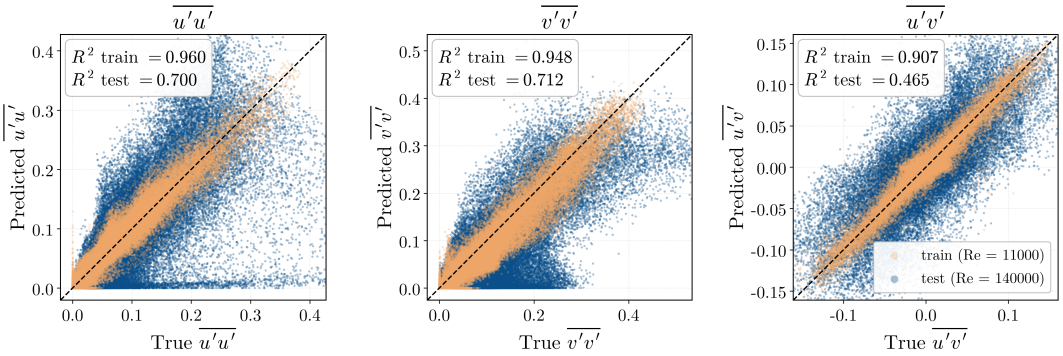



**Figure 6.19:** Recurrent closure: predicted against true phase-averaged Reynolds stress components, training ($\mathrm{Re} = 11\,000$) and held-out ($\mathrm{Re} = 140\,000$) test data, with the head having seen a full period before each phase; the legend of the third panel applies to all three. The shear stress is the component on which the held-out cloud opens most.

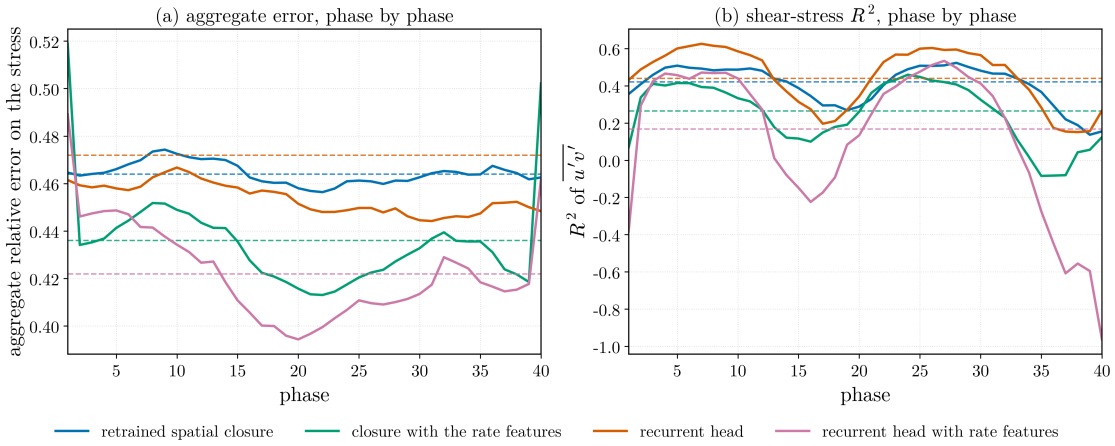



**Figure 6.20:** Phase-by-phase accuracy of the four unsteady families over one shedding period. The horizontal lines are the period-averaged values of Table 6.10.

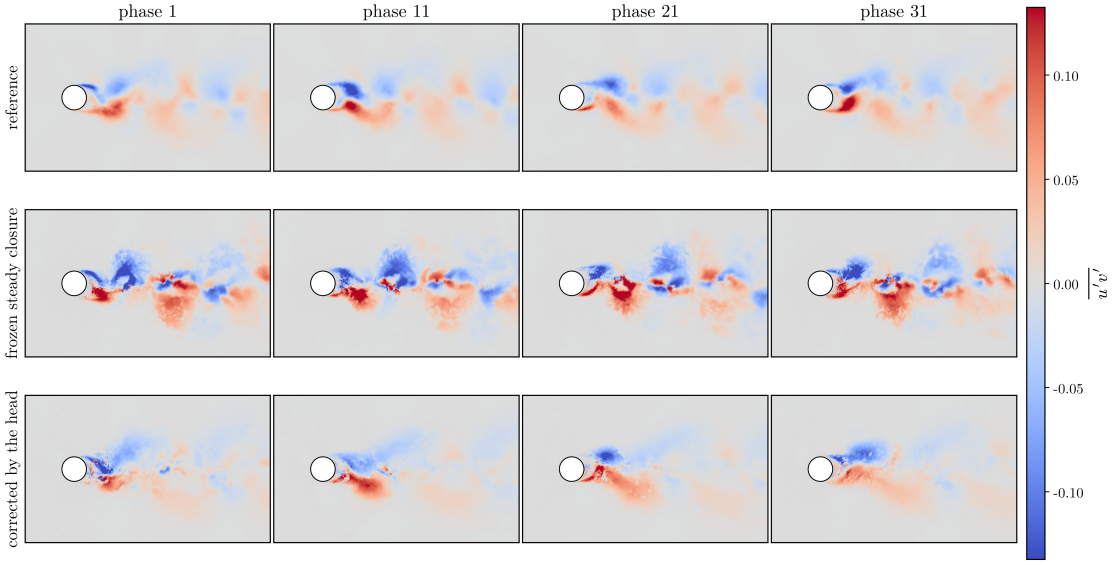



**Figure 6.21:** Phase-averaged shear stress at four phases of the shedding period: reference, frozen steady closure, and the same closure corrected by the recurrent head. The comparison shows what the correction of Equation (6.22) actually supplies.

**Table 6.11:** The four closures in their adopted configurations. Upper block: diagnostics on the window $t \in [60,120]$, the last column being the number of the conditions of Equation (6.18) that are met. Lower block: phase-averaged errors over the near wake on $t \in [130,180]$, with the classical closure continued over the same window and phase averaged in the same way.

| Closure | cyc | $\mathrm{env}_{cv}$ | drift | $C_{l,\mathrm{rms}}$ | St | met |
|---|---|---|---|---|---|---|
| Retrained spatial | 25.2% | 20.2% | +46.4% | 0.122 | 0.179 | 3/5 |
| + rate features | 4.4% | 2.1% | +1.2% | 0.850 | 0.198 | 5/5 |
| + recurrent head | **1.4%** | **0.8%** | −0.0% | 0.291 | **0.199** | 5/5 |
| + rate features and head | 4.8% | 2.2% | +0.4% | **0.237** | 0.193 | 5/5 |
| $k$-$\ell$ | 2.1% | 0.2% | +0.0% | 0.260 | 0.234 | 5/5 |
| Reference | — | — | — | 0.244 | 0.200 | |

| Closure | $\lvert\overline{\mathbf{u}}\rvert$ | $\overline{u'u'}$ | $\overline{v'v'}$ | $\overline{u'v'}$ |
|---|---|---|---|---|
| Retrained spatial | 0.061 | 0.413 | 0.371 | 0.515 |
| + rate features | 0.104 | 0.471 | 0.422 | 0.642 |
| + recurrent head | 0.044 | **0.322** | 0.320 | 0.439 |
| + rate features and head | **0.041** | 0.327 | **0.243** | **0.437** |
| $k$-$\ell$ | 0.085 | 0.780 | 0.871 | 0.918 |

**The two closures carrying memory are the two that reproduce the reference.** They are the only ones within five per cent of the reference on the frequency and within a fifth on the amplitude at once, and they are the two most accurate on every phase-averaged field. They reach half of the error on the velocity and a third of the error on the stresses compared to the classical reference. Between them, the recurrent head alone has the cleaner cycle and the better streamwise stress, and the recurrent head with the rate features has the better amplitude and the better cross-stream stress; on the velocity and on the shear stress they are the same to the second digit. The recurrent head alone is the closure this chapter retains, for the reasons given in Section 6.10.

**Periodicity and accuracy are separate properties.** The closure with the rate features is as periodic as the two with memory, with a dispersion of four per cent and no drift, and it is the least accurate of the four on every field and on the amplitude, which it overshoots by a factor of 3.5. A clean limit cycle is a necessary condition for reading an unsteady solution and not a measure of its accuracy, which is why the diagnostics of Section 6.2.2 and the phase-averaged errors are reported separately and neither replaces the other.

**The *a priori* ordering does not predict the coupling.** On reference data the shear stress is predicted best by the recurrent head, then by the retrained spatial closure, and worst by the closure with the rate features and the head, Table 6.10. Inside the solver the last of these is among the two best on every field and the retrained spatial closure is the one that does not saturate. The ordering established on reference data survives on one point only, the recurrent head alone being first on both, and it reverses on the others. This is the third time in this work that the two disagree, after Chapter 4 and Chapter 5, and here as there the selection of a closure for the coupled problem cannot be made from the *a priori* reference tables.

**The shear stress remains the worst-predicted component after the coupling.** *A priori* it is the worst component of every closure, Section 6.7.1. *A posteriori* it is again the worst of the four fields for every closure, between 0.44 and 0.64 against 0.24 to 0.47 for the normal stresses, so the ordering of the components survives the coupling even where the ordering of the closures does not. The next section measures what this error is made of.

## 6.8 What the closures get wrong

### 6.8.1 The coupling amplifies the error

The residual error of the coupled solutions could be the error of the closures carried through unchanged, or an amplification by the feedback between the stress and the velocity it acts on. Table 6.12 separates the two by comparing, component by component, the *a priori* error of each deployed closure with the *a posteriori* error of the same closure, both at the Reynolds number of the coupled runs and over the whole domain. The *a priori* figure is obtained by evaluating the deployed network on the phase-resolved reference fields at $\mathrm{Re} = 11\,000$, with the exact features and no solver; the *a posteriori* one by phase averaging the coupled solution and aligning it with the reference cycle. Two caveats apply. The reference flow at this Reynolds number is the training flow, so the *a priori* figure measures the fit on the training data and not the generalization that Table 6.10 measures at $\mathrm{Re} = 140\,000$.

Every component is amplified, and the shear stress is amplified most in every closure. The amplification is smallest, a factor of 1.3 to 1.5, for the retrained spatial closure and for the recurrent head, and largest, 2.2 to 3.0, for the closure with the rate features and the head, which is also the closure with by far the tightest fit on the training flow: its *a priori* errors are half those of the others and its coefficients of determination exceed 0.97, where at $\mathrm{Re} = 140\,000$ its shear stress had the lowest of the four, 0.17. A fit that tight is not reproduced inside the solver, where the same closure lands at the errors of the recurrent head alone, 0.33, 0.27 and 0.49 against 0.33, 0.32 and 0.49. The *a priori* error therefore sets neither the level nor

**Table 6.12:** Relative $L^2$ error of each deployed closure at $\mathrm{Re} = 11\,000$ over the whole domain, component by component: *a priori*, on the reference fields with exact features, and *a posteriori*, on the phase-averaged coupled solution, with the ratio of the two.

| Closure | *a priori* | | | *a posteriori* | | | ratio | | |
|---|---|---|---|---|---|---|---|---|---|
| | $\overline{u'u'}$ | $\overline{v'v'}$ | $\overline{u'v'}$ | $\overline{u'u'}$ | $\overline{v'v'}$ | $\overline{u'v'}$ | $\overline{u'u'}$ | $\overline{v'v'}$ | $\overline{u'v'}$ |
| Retrained spatial | 0.318 | 0.281 | 0.390 | 0.444 | 0.407 | 0.580 | 1.4 | 1.4 | 1.5 |
| + rate features | 0.228 | 0.218 | 0.316 | 0.451 | 0.422 | 0.682 | 2.0 | 1.9 | 2.2 |
| + recurrent head | 0.246 | 0.255 | 0.329 | 0.333 | 0.323 | 0.492 | 1.4 | 1.3 | 1.5 |
| + rate features and head | 0.113 | 0.121 | 0.165 | 0.327 | 0.267 | 0.491 | 2.9 | 2.2 | 3.0 |

the ordering of the *a posteriori* one beyond the separation between the two closures with memory and the two without, which survives the coupling. What the ratio measures is the feedback: inside the solver the closure reads features computed from a velocity that its own stress has already altered, and the error it makes on the shear stress, the component through which the stress acts on the mean momentum, is the one that returns to it amplified. The shear stress is the worst component of every closure before the coupling and after it, and the rest of this section is about what that error is made of.

### 6.8.2 When closure learns the mean, not the dynamics of the cycle

The two closures without memory leave the amplitude of the cycle wrong, one by a factor of 3.5 and the other by a growth that has not ended at $t = 180$, and the two with memory bring it within a fifth of the reference. What the four closures have in common, and what separates them, appears once the predicted stress is split into its phase mean and its fluctuation about that mean,

$$\boldsymbol{\tau}(\mathbf{x},\varphi) = \overline{\boldsymbol{\tau}}(\mathbf{x}) + \boldsymbol{\tau}'(\mathbf{x},\varphi), \qquad \overline{\boldsymbol{\tau}}(\mathbf{x}) = \frac{1}{40}\sum_{\varphi} \boldsymbol{\tau}(\mathbf{x},\varphi), \tag{6.23}$$

and the two parts are compared separately against the corresponding split of the reference. Table 6.13 reports the result over the wake.

The error on the fluctuation is three to six times the error on the mean, in all four closures. The two with memory lower it, from 0.77 and 0.64 to 0.58 and 0.35, and leave the ratio where it is. The reason is structural rather than accidental. The training loss is a pointwise squared error over all the samples, every sample is a pair of a cell and a phase, and the phase mean carries 92.5% of the energy. Nothing in the objective asks the network to reproduce $\boldsymbol{\tau}'$, and it does not.

**Table 6.13:** Decomposition of Equation (6.23) over the wake, 40 phases, area-weighted. The first block is a property of the reference data alone. The second is the relative error of the four deployed closures on each part; the two recurrent closures are evaluated with the head having seen a full period before the phase, which is its condition in the solver.

| *Energy of the reference* | | | | |
|---|---|---|---|---|
| $\|\overline{\boldsymbol{\tau}}\|$ | | | 0.1165 | |
| $\|\boldsymbol{\tau}'\|$ | | | 0.0333 | |
| share of the energy in the fluctuation | | | 7.5% | |
| *Relative error of the closure* | Retrained spatial | + rate features | + recurrent head | + rate features and head |
| on $\boldsymbol{\tau}$ | 0.297 | 0.230 | 0.227 | 0.112 |
| on $\overline{\boldsymbol{\tau}}$ | 0.219 | 0.153 | 0.169 | 0.059 |
| on $\boldsymbol{\tau}'$ | **0.767** | **0.642** | **0.576** | **0.352** |

The consequence is dynamical, and it closes the argument. Consider the quantity through which the closure acts on the energy of the oscillation, the turbulent production integrated over the wake,

$$P(\varphi) = -\int_{\Omega_{\text{wake}}} \boldsymbol{\tau}(\varphi) : \mathbf{S}(\varphi)\, d\Omega. \tag{6.24}$$

Table 6.14 compares its variation within the cycle against the reference.

**Table 6.14:** Integrated wake production of Equation (6.24) over the cycle, and its correlation with the reference cycle. The predicted production has very nearly the right mean and almost no variation within the cycle.

| | mean | rms amplitude | amplitude / mean | $\rho$ with reference |
|---|---|---|---|---|
| Reference | 0.4363 | 0.0191 | 0.044 | — |
| Retrained spatial | 0.4620 | 0.0030 | 0.007 | 0.51 |
| + rate features | 0.4552 | 0.0036 | 0.008 | 0.04 |
| + recurrent head | 0.4480 | 0.0033 | 0.007 | 0.08 |
| + rate features and head | 0.4527 | 0.0062 | 0.014 | −0.09 |

The predicted production oscillates three to six times less than the true one in all four closures, and its correlation with the reference cycle is 0.51 for the retrained spatial closure and below 0.1 in absolute value for the other three. The memory that lowers the error on $\boldsymbol{\tau}'$ by a quarter and by a half does not reach the production cycle: the two recurrent closures reproduce the mean of the production within three per cent, as the two local ones do, and its variation within the cycle not at

all. This also reads back onto the *a priori* comparison of Section 6.4.1: the rate features improve what the loss weighs and do not improve what the solver needs. Figure 6.22 shows the production cycle and the two parts of the error.

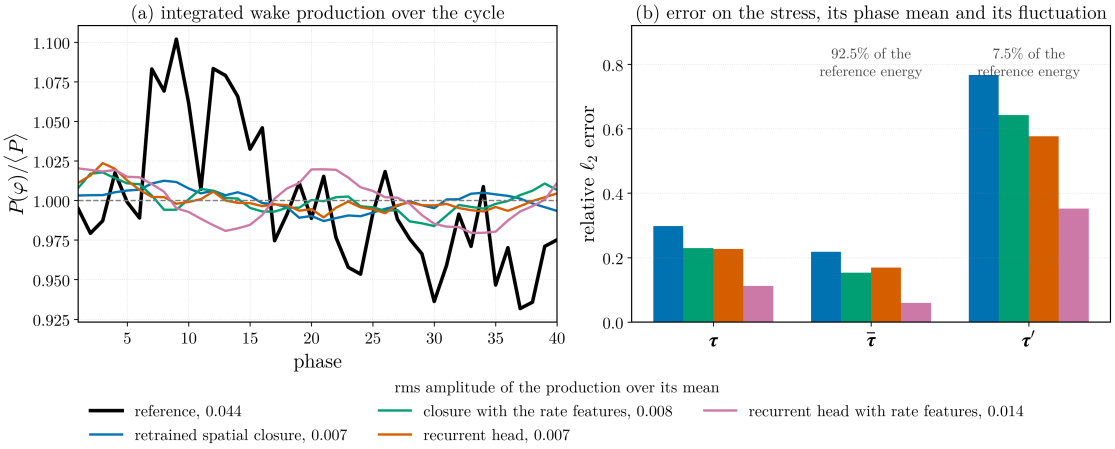



**Figure 6.22:** Left, integrated wake production over one shedding period, normalized by its own mean, for the reference and the four closures. Right, relative error of the four closures on the full stress, on its phase mean and on its fluctuation, against the share of the energy each part carries in the reference.

The missing ingredient is exactly the one that fixes the amplitude of a limit cycle. Writing $A$ for the amplitude of the shedding mode, its slow evolution near onset obeys a Landau equation [70, 71],

$$\frac{dA}{dt} = \sigma A - \mu |A|^2 A, \qquad \text{so that} \qquad |A|_\infty^2 = \frac{\sigma}{\mu}, \tag{6.25}$$

where $\sigma$ is the linear growth rate and $\mu$ the nonlinear saturation coefficient, produced by the distortion of the mean flow that the oscillation itself creates. The Reynolds stress is one of the terms producing that distortion. A closure whose prediction is almost constant over the cycle contributes to $\sigma$, through its mean eddy viscosity, and contributes almost nothing to $\mu$. Equation (6.25) then predicts a saturation amplitude that is too large even when the mean dissipation is right, which is the observation of Section 6.9.2 and of the closure with the rate features in Section 6.7.2, restated in the language of the amplitude equation. It is not, however, what separates the closures that reach the amplitude of the reference from the one that overshoots it: Table 6.14 shows the production of the two recurrent closures as flat over the cycle as that of the local ones, so the saturation term is missing in all four and the amplitude they reach is decided elsewhere.

The same behaviour was visible *a posteriori* in the campaign under the exact split, Section 6.9.3, where the eddy viscosity supplied by the closure over the wake

varied by about five per cent over the whole statistically steady window while the lift swung between $-1.5$ and $+1.6$: what the momentum equation received was very nearly a frozen viscosity field, spatially structured and temporally inert. What the decomposition does separate is the level of the eddy viscosity. The median eddy viscosity of the predicted stress over the wake is $264\nu$ and $263\nu$ for the two local closures, against $209\nu$ for the reference stress, a quarter too large; for the two recurrent closures it is $215\nu$ and $218\nu$, within four per cent of the reference, and its variation within the cycle, 10 to 12% of the mean against 5% for the local closures and 7% for the reference, is the one place where the head is seen to modulate the stress. The two closures that reach the amplitude of the reference are therefore the two that supply the eddy viscosity of the reference, on average over the cycle. Whether that is the cause of their amplitude, through the coefficient of the hyperviscous term of Section 6.9.3 which scales with the same eddy viscosity, is a reading these measurements do not isolate.

The same level does not show in the frequency. A larger eddy viscosity lowers the effective Reynolds number of Equation (6.27) and the shedding frequency with it, and the excess of the local closures would predict a deficit of a few per cent; the Strouhal numbers of Table 6.11 are between one and four per cent below the reference for all three admissible closures, with the closure with the rate features, at $264\nu$, no further from the reference than the recurrent head at $215\nu$. The frequency is set by the numerics of Section 6.9.2 as much as by the closure, and the eddy-viscosity level is not read from it.

Whether a better training objective could repair this is a separate question, and it has a quantitative answer. Section 6.8.3 measures how much of the fluctuation any closure built on these features could reproduce.

### 6.8.3 How much of the fluctuation is learnable at all

Section 6.8.2 shows that the closures reproduce the phase mean and not the fluctuation, and attributes it to the weighting of the training objective. That attribution is incomplete on its own, because it does not say how much of the fluctuation any closure of this form could reproduce with a better objective. This section measures that limit.

**The inequality.** A closure of the family considered here is a function of a feature vector $\mathbf{q}$ evaluated at one cell and one instant. Two samples carrying the same features therefore receive the same prediction, whatever the architecture. For any function $f$,

$$\mathbb{E}\Big[\big\|\boldsymbol{\tau}' - f(\mathbf{q})\big\|^2\Big] \;\geq\; \mathbb{E}\Big[\operatorname{Var}\big(\boldsymbol{\tau}' \,\big|\, \mathbf{q}\big)\Big], \tag{6.26}$$

with $\boldsymbol{\tau}'$ the phase fluctuation of Equation (6.23). The right-hand side depends on the data and on the choice of features alone. It does not depend on the architecture, on the loss or on the training, so it is a floor that no amount of capacity or re-weighting can cross.

The floor is estimated by nearest neighbours in feature space. Two samples drawn independently from the same conditional distribution satisfy $\mathbb{E}\|\boldsymbol{\tau}'_i - \boldsymbol{\tau}'_j\|^2 = 2\,\mathrm{Var}(\boldsymbol{\tau}' \mid \mathbf{q})$, so the mean squared difference between a sample and its $k$ nearest neighbours in $\mathbf{q}$ estimates twice the conditional variance. All the values below use $k = 8$ over the wake, on all 40 phases.

**The floor is already almost reached.** Table 6.15 compares it with what several models of increasing flexibility achieve on the same features.

**Table 6.15:** Relative error on the phase fluctuation of the stress over the wake, for the two feature sets of Sections 6.3 and 6.4. The floor is the right-hand side of Equation (6.26), normalized in the same way. The last column is the trained closure of Table 6.13.

| Features | floor | linear | nearest neighbours | trained closure |
|---|---|---|---|---|
| Four invariants | 0.656 | 0.978 | 0.669 | 0.767 |
| + six rate features | 0.587 | 0.923 | 0.594 | 0.642 |

The gap between the trained closure and the floor is 17% on the first row and 9% on the second, and a nearest-neighbour regressor on the same features closes it almost entirely. Whatever the training objective weighs, the fluctuation of the stress is simply not a function of these features. Re-weighting the loss can recover the gap and nothing beyond it.

**The ambiguity is an ambiguity of phase.** The neighbours that produce the floor are not neighbours in the flow. With the four invariants alone their median phase separation is 8 phases out of 40, and 45% of them lie more than a quarter of a cycle apart. Adding the rate features reduces the median separation to 6 phases and the fraction beyond a quarter cycle to 44%. Two states of the wake that are almost half a period apart can therefore present the same instantaneous invariants and require different stresses, which is the precise sense in which an instantaneous closure is inadequate for this flow.

**What lowers the floor, and what does not.** Equation (6.26) can be evaluated on any feature set, which turns it into a way of asking what information the closure is missing. The argument is one-sided and this is what makes it usable. Enlarging

the feature set can only lower the true floor, while it degrades the estimate, since neighbours in a higher-dimensional space are further apart. A floor that falls is therefore evidence, and a floor that rises says nothing. Table 6.16 reports the sweep.

**Table 6.16:** Floor of Equation (6.26) against the information given to the closure, over the whole wake. The history is a number of past phases of the same cell, the stencil a number of neighbouring cells at the same phase. The mean distance between neighbours is reported because it grows with the dimension and works against the estimate.

| History | Stencil | Dimension | Neighbour distance | Floor | Change |
|---|---|---|---|---|---|
| 0 | 0 | 4 | 0.045 | 0.663 | — |
| 0 | 4 | 20 | 0.165 | 0.661 | −0% |
| 0 | 8 | 36 | 0.272 | 0.661 | −0% |
| 1 | 0 | 8 | 0.088 | 0.554 | −16% |
| 2 | 0 | 12 | 0.124 | 0.507 | −24% |
| 4 | 0 | 20 | 0.184 | 0.473 | −29% |
| 8 | 0 | 36 | 0.281 | 0.439 | −34% |
| 12 | 0 | 52 | 0.360 | 0.420 | **−37%** |
| 1 | 4 | 24 | 0.197 | 0.586 | −12% |
| 2 | 4 | 28 | 0.228 | 0.522 | −21% |
| 4 | 4 | 36 | 0.279 | 0.479 | −28% |

Three conclusions follow from Table 6.16, and they are the most transferable result of this chapter.

- **Close spatial information does not help.** A stencil of four or eight neighbouring cells leaves the floor at 0.661 against 0.663. Whatever the closure is missing, it is not spatial non-locality, at least not the close spatial one. To observe a descent of the floor, far spatial information are probably need. Since the improvement given by non local architectures of this thesis.
- **Temporal information helps, monotonically, and is not saturated.** One past phase lowers the floor by 16%, two by 24%, four by 29%, twelve by 37%, and twelve phases are less than a third of a shedding period. This is the quantitative justification of the rate features and of the recurrent head, and it is the direction in which the closure has room to improve.
- **Adding the stencil on top of the history costs rather than pays.** The rows combining both are worse than the corresponding history-only rows, by one to six per cent, which is the expected behaviour of the estimator in a

higher-dimensional space and not an argument that spatial information is harmful.

**Reconciliation with the *a priori* comparison.** Section 6.5.1 finds that the recurrent head, on the aggregate error, does no better than retraining the spatial network. The two results are consistent, and the reason is a matter of weighting. The fluctuation carries 7.5% of the energy of the stress, so a metric aggregated over the whole tensor barely sees it. It carries a much larger share of the *error*: decomposing the aggregate error of the closures of Table 6.13 into its two parts gives 50% and 59% of the squared error on the fluctuation alone for the two local closures, and 49% and 74% for the two recurrent ones, because the relative error there is three to six times larger. A closure that reached the four-phase floor of Table 6.16 would improve the aggregate error from 0.297 to about 0.247, a gain of 17%, and would improve the part of the prediction on which the amplitude of the cycle depends by a factor of 1.6. The natural metric on which the unsteady closures were selected is therefore not the metric on which they are used. This means that implementing some of the other variants tested during the sweep could bring better results when coupled inside the solver. Figure 6.23 shows the floor against the size of the feature set and the phase separation of the nearest neighbours.

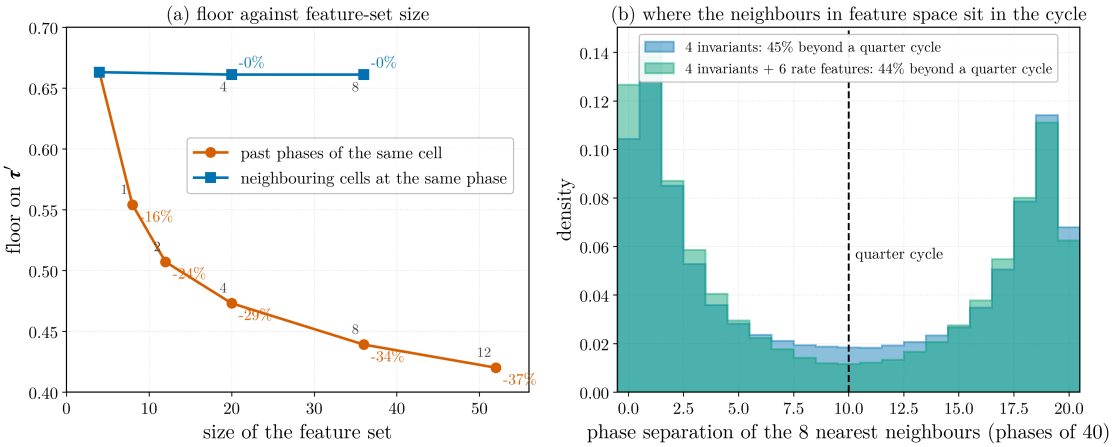



**Figure 6.23:** Left, the floor of Equation (6.26) against the size of the feature set, separately for past phases of the same cell and for neighbouring cells at the same phase. Right, the distribution of the phase separation between a sample and its nearest neighbours in feature space, with the quarter-cycle mark.

## 6.9 Ablations

The comparisons behind the choices of the previous sections are collected here, grouped by what they vary: the learning setting of each family, the numerical setting of the solver, and the filtering and splitting of the closure term.

### 6.9.1 Learning rate

**Retrained spatial closure.** Table 6.17 reports the learning-rate sweep of the closure of Section 6.3. All configurations use two hidden layers of 24 units and 1000 epochs.

**Table 6.17:** Tensor-basis closure trained directly on phase-resolved data, learning-rate sweep, held-out Re $= 140\,000$. All configurations use two hidden layers of 24 units and 1000 epochs.

| Learning rate | $R^2_{\overline{u'u'}}$ | $R^2_{\overline{v'v'}}$ | $R^2_{\overline{u'v'}}$ | $\varepsilon^{\text{tensor}}_{L^2}$ |
|---|---|---|---|---|
| $3 \times 10^{-4}$ | 0.623 | 0.603 | **0.421** | 0.465 |
| $5 \times 10^{-4}$ | 0.585 | 0.645 | 0.283 | 0.468 |
| $1 \times 10^{-3}$ | 0.550 | 0.611 | 0.164 | 0.472 |
| $2 \times 10^{-3}$ | 0.616 | 0.621 | 0.212 | 0.460 |
| $5 \times 10^{-3}$ | **0.669** | 0.632 | 0.284 | **0.448** |

The aggregate error varies between 0.448 and 0.472 across a sweep spanning more than an order of magnitude in learning rate, a spread of about five per cent. The coefficients of determination vary much more, and the shear stress varies most of all, from 0.164 to 0.421. The two metrics do not agree on which configuration is best, which is the same disagreement documented throughout Chapter 3 and has the same cause. The aggregate error is used to order the configurations here, since it is the stable one.

**Rate features.** Table 6.18 repeats the sweep with the six rate features of Section 6.4 added to the coefficient-network input, everything else being identical.

On the aggregate error the six features buy nothing. The best value moves from 0.448 to 0.454, that is, slightly in the wrong direction, and the range of the sweep narrows from $[0.448, 0.472]$ to $[0.454, 0.470]$. The honest summary of Table 6.18 is that the rate features make the model less sensitive to the learning rate and no more accurate.

Under the coefficients of determination the picture is different, and only partly consistent. Comparing the two tables at equal learning rate, the cross-stream normal stress improves at every learning rate, by up to 0.10, and the streamwise

**Table 6.18:** Tensor-basis closure trained on phase-resolved data with the six rate features of Equations (6.20)–(6.21) added to the coefficient-network input, learning-rate sweep, held-out Re $= 140\,000$. Everything else is identical to Table 6.17.

| Learning rate | $R^2_{\overline{u'u'}}$ | $R^2_{\overline{v'v'}}$ | $R^2_{\overline{u'v'}}$ | $\varepsilon^{\text{tensor}}_{L^2}$ |
|---|---|---|---|---|
| $3 \times 10^{-4}$ | 0.677 | **0.707** | 0.305 | 0.470 |
| $5 \times 10^{-4}$ | **0.679** | 0.702 | 0.211 | 0.457 |
| $1 \times 10^{-3}$ | 0.669 | 0.643 | 0.320 | 0.457 |
| $2 \times 10^{-3}$ | 0.613 | 0.654 | 0.292 | **0.454** |
| $5 \times 10^{-3}$ | 0.627 | 0.642 | **0.321** | 0.462 |

one at three of the five. The shear stress is mixed: it degrades at the two smallest learning rates, by up to 0.12, and improves at the other three. The evidence from this experiment alone is therefore that the rate features help the normal stresses and leave the shear stress unresolved. The same comparison repeated with a memory mechanism present, Section 6.5.1, is unambiguous, and it is negative on the shear stress.

**Recurrent head.** Table 6.19 reports the sweep of the head of Section 6.5, receiving the coefficient sequence only in the upper block and the local features together with the six rate features in the lower block. The local features of the lower block include the pressure-gradient norm, so that head is not pressure-free.

**Table 6.19:** Recurrent head on a frozen steady spatial network, held-out Re $= 140\,000$. Upper block: the head receives the coefficient sequence only. Lower block: the head additionally receives the local features and the six rate features of Equations (6.20)–(6.21).

| Head input | Learning rate | $R^2_{\overline{u'u'}}$ | $R^2_{\overline{v'v'}}$ | $R^2_{\overline{u'v'}}$ | $\varepsilon^{\text{tensor}}_{L^2}$ |
|---|---|---|---|---|---|
| coefficients only | $1 \times 10^{-4}$ | 0.675 | 0.685 | 0.425 | 0.491 |
| | $3 \times 10^{-4}$ | **0.688** | **0.696** | **0.441** | **0.472** |
| | $1 \times 10^{-3}$ | 0.657 | 0.662 | 0.357 | 0.477 |
| | $3 \times 10^{-3}$ | 0.658 | 0.677 | 0.346 | 0.487 |
| + local and rate features | $1 \times 10^{-4}$ | 0.701 | **0.737** | 0.135 | 0.449 |
| | $3 \times 10^{-4}$ | **0.718** | 0.717 | 0.131 | 0.436 |
| | $1 \times 10^{-3}$ | 0.600 | 0.644 | $-0.108$ | 0.435 |
| | $3 \times 10^{-3}$ | 0.658 | 0.668 | **0.210** | **0.424** |

**Recurrent head without the pressure feature.** Table 6.20 repeats the lower block of Table 6.19 with the pressure-gradient norm removed from the head input, which is the sweep from which the closure of Section 6.6 is taken.

**Table 6.20:** Recurrent head receiving the coefficient sequence, the local features without the pressure-gradient norm, and the six rate features, learning-rate sweep, held-out Re $= 140\,000$.

| Learning rate | $R^2_{\overline{u'u'}}$ | $R^2_{\overline{v'v'}}$ | $R^2_{\overline{u'v'}}$ | $\varepsilon^{\text{tensor}}_{L^2}$ |
|---|---|---|---|---|
| $1 \times 10^{-4}$ | 0.692 | **0.739** | 0.130 | 0.452 |
| $3 \times 10^{-4}$ | **0.703** | 0.719 | **0.172** | 0.434 |
| $1 \times 10^{-3}$ | 0.689 | 0.680 | 0.152 | **0.422** |
| $3 \times 10^{-3}$ | 0.674 | 0.675 | 0.168 | **0.422** |

## 6.9.2 Numerical setting

When comparing closures it is necessary to know how much of the outcome the numerical setting decides. Table 6.21 holds the closure fixed, at the family carrying the rate features, and varies the stabilization and the treatment of the isotropic part of the stress; all four settings use the exact split and three corrector sub-iterations, which were the settings of the first campaign, and none of them is filtered. It is the only family for which every setting was marched long enough for the amplitude to saturate, which the drift column certifies.

**Table 6.21:** One closure, four numerical settings, window $t \geq 60$. The first row is the consistent stabilization alone and the second adds the entropy viscosity, so the two differ only in the added dissipation, and the last two only in whether the isotropic part of the predicted stress is retained. The transition between the second and the third row changes three ingredients at once and is not a single-variable comparison.

| Numerical setting | cyc | HF | peak | env$_{cv}$ | drift | $C_{l,\text{rms}}$ | St |
|---|---|---|---|---|---|---|---|
| Nothing added | 117.5% | 0.42% | 75.5% | 11.5% | +4.2% | 1.156 | 0.191 |
| + entropy viscosity | 43.1% | 0.02% | 98.0% | 6.7% | +2.1% | 0.960 | 0.176 |
| Isotropic diffusion + filter | 30.5% | 0.02% | 90.9% | 4.0% | +0.9% | 0.966 | 0.220 |
| + isotropic part removed | **14.9%** | 0.02% | 97.3% | 4.5% | +1.9% | 0.950 | 0.219 |
| $k$-$\ell$ control | 2.1% | 0.00% | 100% | 0.2% | +0.0% | 0.260 | 0.234 |
| Reference | — | — | — | — | — | 0.244 | 0.200 |

The second stabilization of Table 6.21 is the isotropic-diffusion form of Equation (3.70), used here without the lagged right-hand side described in Section 3.4.2, so that it contributes only its positive semidefinite bilinear form. It is non-consistent, and it therefore perturbs the solution the discrete problem converges to, which is the reason the consistent form is preferred wherever it is usable.

Three readings follow.

**Under the exact split, the stabilization decides the periodicity.** The cycle-to-cycle dispersion moves from 117.5% to 14.9% across the table, a factor of eight, with the closure held fixed. Every ingredient that helps does so by removing something the momentum equation cannot absorb rather than by adding accuracy.

**Removing the isotropic part of the stress halves the dispersion.** This is the cleanest single-variable comparison of the chapter, from 30.5% to 14.9% with every other setting identical, and its mechanism is a property of the function spaces. Writing $\boldsymbol{\tau} = \frac{1}{2}\mathrm{tr}(\boldsymbol{\tau})\mathbf{I} + \boldsymbol{\tau}_{\mathrm{dev}}$, the divergence entering Equation (6.1) is $\frac{1}{2}\nabla\,\mathrm{tr}(\boldsymbol{\tau}) + \nabla\cdot\boldsymbol{\tau}_{\mathrm{dev}}$. The first term is a pure gradient, and in the continuous problem a pure gradient is absorbed by the pressure and leaves the velocity untouched. In the discrete problem it is absorbed only to the extent that it lies in the gradient of the pressure space, which is $P_1$, while the stress is evaluated cell by cell. What is left over is a forcing at the scale of the cell, and it is largest where the flow is quietest, since the isotropic part of the prediction does not decay to zero in the free stream. Removing it is admissible for the same reason: it changes the pressure and not the velocity in the continuous limit.

**These four ingredients never touch the amplitude.** Across the four settings the lift fluctuation stays between 0.95 and 1.16, that is within twenty per cent of itself and about four times the reference value of Equation (6.19), while the dispersion changes by a factor of eight. None of these four ingredients touches the amplitude; the form of the split does, Section 6.9.3, and what the two closures without memory then still get wrong is located in the closure by Section 6.8.2.

The frequency behaves in the third way again. Without added dissipation the Strouhal number is 0.191 against a reference of 0.200, the closest value of the table. The entropy viscosity lowers it to 0.176 and the non-consistent form with the output filter raises it to 0.220. Each device that improves the regularity of the cycle distorts its frequency, in one direction or the other.

**The added dissipation extinguishes the recurrent closure, and the cause is identified.** Under the exact split of Section 6.9.3, with the entropy viscosity of Table 6.21 added, the recurrent closure does not shed at all. It supplies an eddy

viscosity of $249\nu$ in the wake on its own, against $148\nu$ for the closure with the rate features on which the dose of entropy viscosity was calibrated. Adding $42\nu$ on top of the first gives an effective Reynolds number

$$\mathrm{Re}_{\mathrm{eff}} = \frac{\mathrm{Re}}{1 + (\nu_t + \nu_E)/\nu} \approx 38, \tag{6.27}$$

below the critical value of about 47 at which the von Kármán street appears behind a circular cylinder, and no oscillation can exist there. The same calculation gives 52 and 65 for the two closures that do shed. Equation (6.27) does not explain the extinctions observed without added dissipation under the exact split, whose effective Reynolds numbers are all above 85, so it is one identified mechanism and not a general account. What it does establish is methodological: closures whose intrinsic eddy viscosities differ by a factor of four cannot be compared at a fixed dose of added dissipation, since part of what is then measured is the mismatch between the dose and the closure.

### 6.9.3 Filtering and splitting

The steady solver of Chapter 3 adopts a filtering configuration for the closure term and a form of the explicit–implicit split, Section 3.4.4, that the first campaign of this chapter did not use. That campaign kept the two strain rates of Equation (6.7) separate, so that the stress imposed on the flow is the closure's own, and under that split no closure produced an admissible cycle in any filtering configuration, with or without added dissipation, at one, three or five corrector sub-iterations. Table 6.22 reports the cycle-to-cycle dispersion of the four closures under the two forms of the split and the four filtering configurations of the steady solver, with every other setting held at the values of Section 6.1.

Three readings follow.

**The form of the split decides whether a cycle exists.** Under the exact split the dispersion never falls below 58%, under the hyperviscous one six of the sixteen runs are admissible. The two forms differ in one term only, $2\nu_{t,\mathrm{eff}}(\mathbf{S}_{\mathrm{filt}} - \mathbf{S}_{\mathrm{raw}})$, which the exact split removes and the hyperviscous one keeps, and which acts as a biharmonic dissipation of coefficient $\nu_{t,\mathrm{eff}}$ times the square of the filter length. That term is where the two forms part, so it is what sustains the cycle: a dissipation that is proportional to the closure's own eddy viscosity, that acts on the smallest resolved scales only, and that the closure cannot switch off where it is active. The entropy viscosity of Table 6.21 adds dissipation too and does not produce a cycle, so it is not the amount that matters but where it acts; the entropy viscosity is large where the residual is large, that is in the coherent vortex cores, while the hyperviscous term is large where the solution is roughest.

**Table 6.22:** Cycle-to-cycle dispersion of Equation (6.17), window $t \in [60,120]$, for the four closures under the exact and the hyperviscous form of the split and the four filtering configurations. Bold marks the runs that satisfy Equation (6.18); the starred one does so on this window and drifts on the next. The classical closure gives 2.1%.

| Split | Closure | No filter | Input | Output | In+out |
|---|---|---|---|---|---|
| Exact | Retrained spatial | 124.9% | 57.9% | 114.0% | 97.7% |
| | + rate features | 75.1% | 120.8% | 181.1% | 165.4% |
| | + recurrent head | 118.8% | 160.2% | 228.0% | 239.6% |
| | + rate features and head | 154.6% | 103.8% | 185.9% | 95.4% |
| Hyperviscous | Retrained spatial | 75.2% | 39.6% | 26.4% | 25.2% |
| | + rate features | 63.1% | 87.7% | 8.6%* | **4.4%** |
| | + recurrent head | 142.7% | 37.6% | **1.4%** | **4.1%** |
| | + rate features and head | 83.2% | 61.6% | **4.8%** | **5.1%** |

**The output filter is what the hyperviscous split needs, and the input filter is what it does not.** Under the hyperviscous split every admissible run carries the output filter, and adding the input filter on top of it changes the dispersion by a few points in either direction and the amplitude for the worse in every closure. The input filter alone helps the two closures it helps under the exact split as well, by a factor of two on the retrained spatial closure and by four on the recurrent head, and leaves both far from admissible. This is the reverse of the steady result of Chapter 3, where the input filter is what the coupled problem needs; a steady iteration and a time march are stabilized by different terms.

**The closure with the worst reference accuracy is the one the filter helps most.** The recurrent head goes from a dispersion above one hundred per cent without a filter to the lowest of the campaign with the output filter, and the closure with both additions from 83% to 5%; the retrained spatial closure moves the least. The stress these closures predict is rougher than the one the spatially local closures predict, since the head corrects the coefficients phase by phase, and a rough stress is exactly what the output filter acts on. Under the exact split the same roughness is what prevented the cycle, Table 6.12 of the first campaign having shown, on the best run of that campaign, *a posteriori* errors of 1.23, 0.83 and 0.87 on the three stress components against 0.32, 0.32 and 0.44 for the recurrent closure now.

### 6.9.4 Corrector sub-iterations

Every computation of this chapter uses one corrector sub-iteration per step, that is a single linear solve, Section 6.1.2. Table 6.23 reports the four closures at one, three and five sub-iterations under the exact split without a filter, which is the setting of the first campaign. The middle column is measured on a computation of 100 time units and its window is $t \in [60{,}100]$; the other two on 120 units and $t \in [60{,}120]$.

**Table 6.23:** Cycle-to-cycle dispersion at one, three and five corrector sub-iterations per step, exact split, no filter. The cost per step is 0.41, 1.30 and 2.1 seconds on sixteen processes.

| Closure | 1 | 3 | 5 |
|---|---|---|---|
| Retrained spatial | 124.9% | 79.4% | 119.4% |
| + rate features | 75.1% | 87.1% | 79.6% |
| + recurrent head | 118.8% | 110.3% | 143.9% |
| + rate features and head | 154.6% | 112.9% | 72.1% |

The number of sub-iterations moves the dispersion in both directions and never below 70%, and the differences between columns are of the same size as the differences between runs of the same closure at different filter settings. Re-linearizing the convection within a step is therefore not what the coupled problem lacks, and it multiplies the cost of the march by three and by five. One sub-iteration is retained.

### 6.9.5 Length of the window

The diagnostics of Section 6.2.2 depend on the window they are computed on, and a limit cycle that is still evolving would show it as a change of its statistics from one window to the next. Table 6.24 reports the four adopted runs on four consecutive windows of 30 time units, six shedding periods each, from the end of the transient to the end of the continuation.

For the three admissible closures nothing moves: the amplitude is the same to the second digit on the four windows, the Strouhal number to the third, and the dispersion stays below seven per cent everywhere, with no trend from the first window to the last. The prediction does not degrade over the 120 time units after the transient, and the values reported in the previous sections are those of a saturated cycle and not of a window chosen well. The retrained spatial closure is the exception, and Table 6.24 is where its state is read: the amplitude grows by a factor of 2.7 across the four windows and the drift stays above ten per cent on each, so the run is on the growing branch of its cycle throughout. The dispersion

**Table 6.24:** Cycle statistics of the four adopted runs on consecutive windows of 30 convective time units. The peak fraction is not reported because its spectral band is not resolved on a window this short.

| Closure | | [60,90] | [90,120] | [120,150] | [150,180] |
|---|---|---|---|---|---|
| Retrained spatial | cyc | 18.9% | 22.6% | 21.8% | 15.0% |
| | drift | +29.9% | +18.6% | +10.5% | +17.1% |
| | $C_{l,\mathrm{rms}}$ | 0.094 | 0.144 | 0.183 | 0.258 |
| + rate features | cyc | 2.8% | 2.2% | 4.0% | 2.5% |
| | drift | +0.1% | +0.1% | −0.5% | +0.4% |
| | $C_{l,\mathrm{rms}}$ | 0.846 | 0.854 | 0.842 | 0.847 |
| + recurrent head | cyc | 0.6% | 0.4% | 3.9% | 0.7% |
| | drift | +0.2% | +0.0% | −3.9% | −0.2% |
| | $C_{l,\mathrm{rms}}$ | 0.291 | 0.291 | 0.299 | 0.291 |
| + rate features and head | cyc | 1.2% | 6.3% | 3.6% | 5.1% |
| | drift | −0.4% | −0.6% | +0.0% | −1.9% |
| | $C_{l,\mathrm{rms}}$ | 0.233 | 0.240 | 0.235 | 0.236 |

computed on the whole window $t \in [60,180]$ rises to seven per cent for the two closures with memory, against under four on each of its parts, because folding twenty-four periods on a single estimate of the period accumulates the phase error of that estimate; the consecutive windows are the fair reading, and Equation (6.17) is to be read on windows of ten periods or so. The same test on the phase-averaged fields is in Table 6.25, which recomputes the errors of Tables 6.3 to 6.9 on the two halves of the continuation, five periods each.

**Table 6.25:** Relative $L^2$ error of the phase-averaged fields over the near wake, mean over the phases, on the two halves of the continuation window.

| | $t \in [130,155]$ | | | $t \in [155,180]$ | | |
|---|---|---|---|---|---|---|
| Closure | $\lvert\overline{\mathbf{u}}\rvert$ | $\overline{u'u'}$ | $\overline{u'v'}$ | $\lvert\overline{\mathbf{u}}\rvert$ | $\overline{u'u'}$ | $\overline{u'v'}$ |
| Retrained spatial | 0.068 | 0.431 | 0.520 | 0.058 | 0.398 | 0.513 |
| + rate features | 0.104 | 0.471 | 0.643 | 0.104 | 0.470 | 0.642 |
| + recurrent head | 0.044 | 0.322 | 0.440 | 0.044 | 0.322 | 0.439 |
| + rate features and head | 0.041 | 0.328 | 0.437 | 0.041 | 0.327 | 0.437 |

For the three admissible closures the two halves agree to the third digit on every field, so five periods are enough to converge the phase average and the errors reported earlier do not depend on the window. For the retrained spatial closure the

errors fall from the first half to the second, by one hundredth on the velocity and by three on the streamwise stress, which is the growing amplitude of Table 6.24 approaching the reference; its errors are those of a state in transit and would be read again once the growth ends.

## 6.10 Summary

The unsteady coupling produces a usable solution for three of the four closures, and the conditions under which it does are now measured rather than conjectured.

- The form of the explicit–implicit split decides whether a limit cycle exists at all. Under the exact split, which imposes the closure's stress and nothing else, no closure sustains a periodic cycle in any filtering configuration, at any number of corrector sub-iterations, with or without added dissipation. Under the hyperviscous split of the steady solver, with the predicted stress filtered on its way out, three of the four do. The term that separates the two forms is a dissipation proportional to the closure's own eddy viscosity and to the square of the filter length, acting on the smallest resolved scales, and it is what a time march of this problem needs from the numerics.
- The numerics are otherwise not the obstacle. The classical control produces a clean and periodic cycle on the same mesh, time integration and stabilization, and the number of corrector sub-iterations, the entropy viscosity and the isotropic-diffusion form of the stabilization move the dispersion without ever producing a cycle under the exact split.
- The two closures carrying memory reproduce the reference. The recurrent head gives the cleanest cycle of the chapter, more periodic than the control, with the shedding frequency within one per cent of the reference and the amplitude a fifth too large; the recurrent head with the rate features gives the amplitude within three per cent and the frequency within four. Both bring the phase-averaged velocity within four per cent of the reference over the near wake and the stresses within 0.24 to 0.44, where the classical closure, periodic and at the right lift amplitude, leaves the stresses at 0.78 to 0.92 and the drag 40% low; and both are stable over 120 time units after the transient.
- Periodicity and accuracy are separate properties. The closure with the rate features and no memory is as periodic as the two with memory and is the least accurate of the four, with an amplitude 3.5 times the reference. The retrained spatial closure is spectrally clean and still growing at $t = 180$.
- The *a priori* ordering does not predict the coupling, for the third time in this work, and the coupling amplifies the error of every closure, by a factor of 1.3

to 3 on the same flow and the same domain, most on the shear stress and most for the closure with the tightest fit on the training data.

- What remains wrong is the shear stress, the worst-predicted component of every closure before and after the coupling. It is wrong in the way Section 6.8 measures: the closures reproduce the phase mean of the stress, which carries 92.5% of the energy the training objective sees, and not its variation within the cycle. A floor derived from the features alone sits within 9 to 17 per cent of what the two spatially local closures achieve on that variation. Feeding the closure past phases lowers that floor by up to 37% and feeding it neighbouring cells lowers it by nothing, which is the *a priori* counterpart of the *a posteriori* result that the two closures with memory are the two that couple.

Figure 6.24 shows the phase-averaged fields of the reference, of the classical control and of the recurrent closure at one phase of the cycle.

Among the four, the recurrent head alone is the closure retained. It is the most periodic, the most accurate on the velocity and on the shear stress, and the closest on the frequency, and it does so with the smaller of the two heads and without the rate features, whose contribution inside the solver is a better amplitude and a better cross-stream stress at a worse cycle. Its amplitude, a fifth above the reference, is the quantity its successor should improve, and the closure with the rate features and the head shows that the amplitude is reachable by this family.

Four directions follow from these measurements, in the order in which the evidence supports them.

The first is a deeper temporal memory. Table 6.16 shows the floor on the fluctuation still falling at twelve past phases, less than a third of a shedding period, and the two closures that carry memory are the two that couple, so the information the closure needs is there and is only partly used. The rate features amount to a memory of one phase, and one phase buys 16% of the 37% available.

The second is the split itself. The hyperviscous form was adopted here because it is what the steady solver uses and it is the form under which the cycle exists, and its effect has been measured on the cycle and on the phase-averaged fields but not isolated from the filter it needs. A study of the filter length under this split, and of a hyperviscosity applied with a coefficient of its own rather than through the filtered strain, would say how much of the accuracy of Table 6.11 is the closure's and how much is the term's.

The third is a loss term on the fluctuation, as in the previous version of this argument: a penalty on the part of the predicted stress that varies within the cycle changes what the objective weighs without changing the architecture, and the dataset of Section 6.2.1 already provides every phase of a cell in one batch.

The fourth concerns the recurrent closures. They are trained on windows starting from a zero internal state and deployed in a march that never resets it, and they

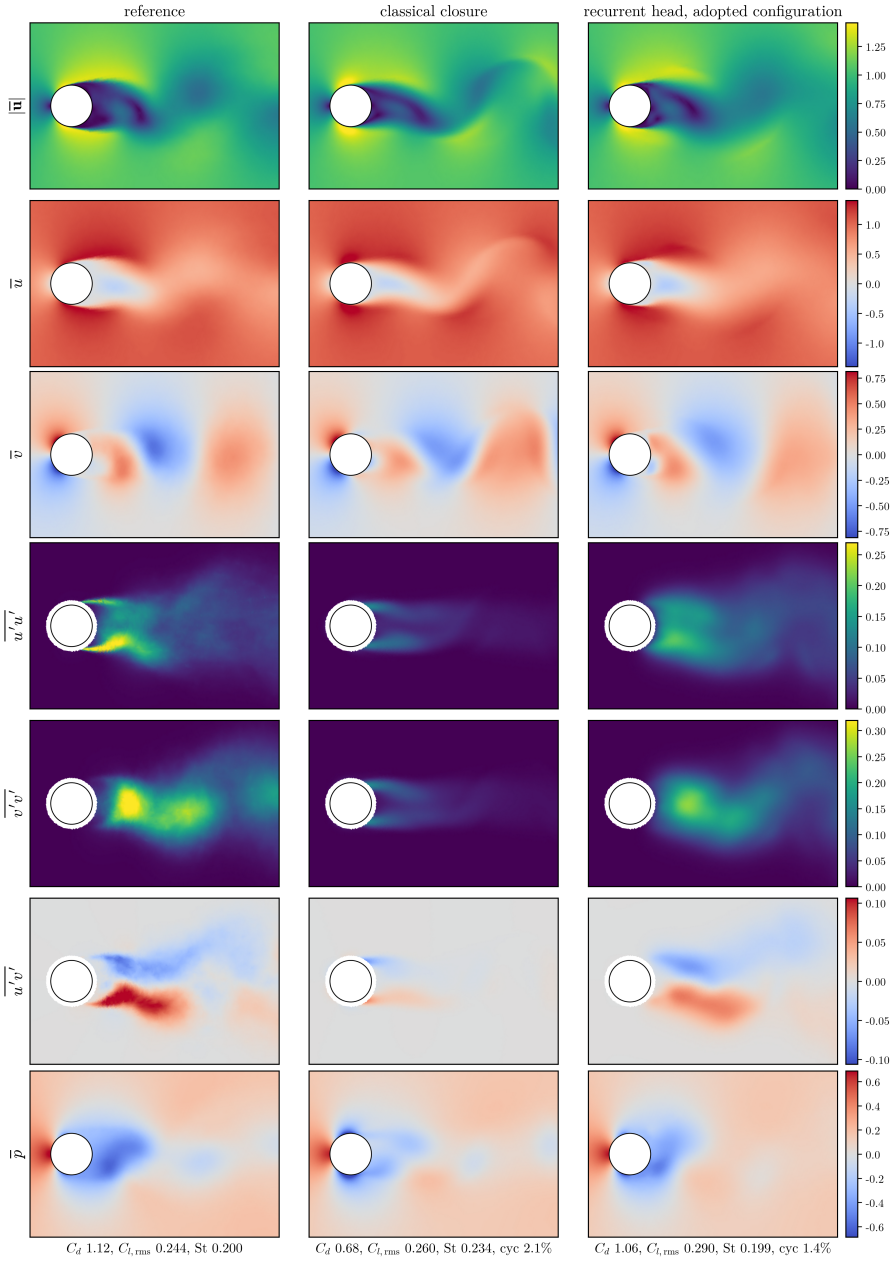



**Figure 6.24:** Phase-averaged fields in the near wake, at one phase of the cycle, for the reference, the classical control and the recurrent closure in its adopted configuration. Rows, from top to bottom: velocity magnitude, the two velocity components, the three Reynolds stress components and the pressure.

couple regardless; whether their accuracy would improve if the two were made the same object is a question the present runs leave open.

# Chapter 7

# Conclusions

Data-Driven Turbulence Closures ranging from pointwise to non-local in space, and from instantaneous to recurrent in time, were coupled to a finite element solver of the RANS equations and measured on the turbulent wake of a circular cylinder, in its steady and in its unsteady form. Every closure was judged on four counts: the accuracy on reference data, the coupling with the solver, the accuracy of the coupled solution and the convergence of the iteration. Figure 7.1 summarizes the comparison. The findings are stated below, with their evidence in Section 7.2, in Tables 7.1 and 7.2 and in Figure 7.2.

**Does the closure gain from reading spatial information beyond the single point?** ***Yes, on all four counts, when its reach is far enough.*** A graph closure with a coarse level of fixed node count produces the most accurate mean field of the work, with a relative $L^2$ error on the velocity of 0.037 over the near wake against 0.048 for the local closure and 0.103 for the classical one, and converges further than any other learned closure. This is the missing half of the evidence reviewed in Section 1.1.4: non-local closures were known to fit reference data better, and it is now shown that the gain can survives the coupling under certain numerical conditions.

**Does the closure gain from a memory of the shedding cycle?** ***Yes, modestly on reference data and decisively inside the solver.*** A recurrent head over the past phases is what separates a closure that reproduces the reference vortex street, with a velocity within five per cent of the reference over the near wake, from one that never sustains a limit cycle. This answers the question left open in Section 1.1.5, whether a phase-resolved closure with memory can be deployed in a time march and not only fitted to a trajectory.

**Underneath both, can a learned closure be used inside a solver at all?** ***Yes, if the numerics of the coupling are built for it.*** The filtered split of the stress, which introduces *a new stabilization term* for data driven closures, couples every architecture of this work, local and non-local, steady and phase-resolved, and without it none does. This is a fourth answer to the ill-conditioning reviewed in Section 1.1.3, and it needs neither a modified transport equation nor a constraint at training.

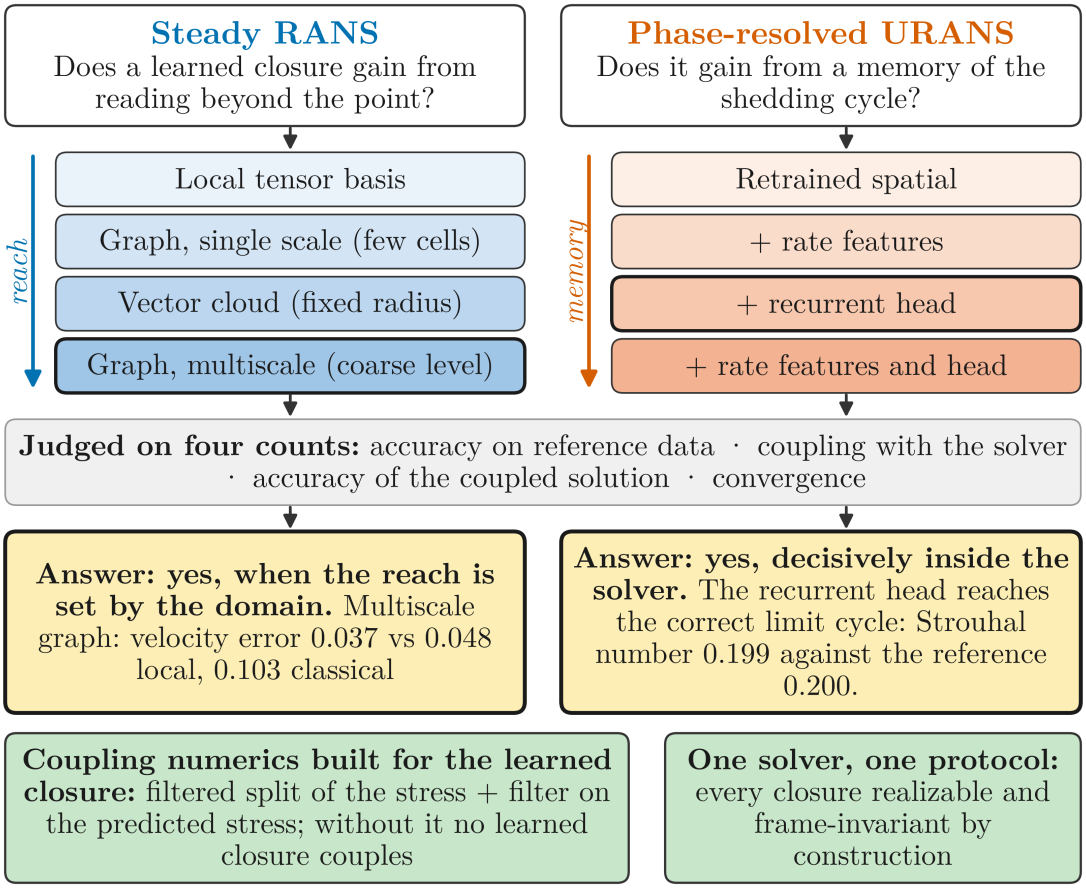



**Figure 7.1:** The thesis in one scheme. The two questions, the closures compared under each, in order of increasing reach on the left and of increasing memory on the right, the four counts on which every closure is judged, and the two answers with their key numbers. The two boxes at the bottom are the common ground of both halves: a solver whose numerics are built for the learned closure, and one protocol under which every closure is realizable and frame-invariant by construction. The thick border marks the closure adopted in each half.

## 7.1 The work in brief

The work proceeded in four steps, each one built on the previous.

- *A solver in which a learned closure replaces the classical one.* Section 3.3 builds a finite element solver for the averaged equations in which a learned Reynolds stress enters the momentum equation, with the numerical treatment of that term worked out ingredient by ingredient, the implicit and explicit split of the stress, its filtered form, the consistent stabilization, the filters on the closure input and output, and the near-wall damping. Every closure of this work is coupled through this one solver and judged with one protocol, on reference data at a held-out Reynolds number and inside the solver at the training one, over the same near-wake window.

- *Local closures as the baseline.* Section 3.2.2 trains a pointwise network and a tensor-basis network on three Reynolds numbers, and retains the tensor-basis one after its reconstruction, its loss and its per-point weighting had been settled by ablation.

- *Two routes to non-locality.* Chapters 4 and 5 build the vector-cloud closure, which reads a neighbourhood of fixed size whose radius is a closed-form function of the wall distance and of the angular position, obtained by symbolic regression, and the graph closures, which read the mesh itself through message passing with an equivariant vector channel, in a single-scale form and in a multiscale form that adds a coarse level of fixed node count.

- *The extension to the phase-resolved equations.* Chapter 6 marches the solver in time, defines an admission criterion for a limit cycle and trains four closures on phase-resolved data, from the retrained tensor basis to a recurrent head fed with rate-of-change features, each evaluated on reference data and then in the time march against the classical model and the phase-averaged reference.

The rest of this chapter states what these steps established, what follows from it for data-driven closures in general, and what remains to be done.

## 7.2 Main findings

**Large range non-locality is the best approach.** ***The multiscale graph closure is the most accurate of the work inside the solver and the one that converges furthest. The vector cloud, the most accurate on reference data, is less accurate than the local TBNN once coupled.*** Table 7.1 and Figure 7.2 collect the evidence. On reference data the non-local

closures are the most accurate of the work: the vector cloud reaches coefficients of determination of 0.911, 0.975 and 0.959 on the three stress components and the multiscale graph 0.926, 0.945 and 0.925, against 0.805, 0.944 and 0.911 for the local tensor basis, Table 5.1. All three couple under the same numerical configuration as the local closure, with no ingredient added for them. Inside the solver the answer separates: the multiscale graph closure of Chapter 5 produces the most accurate mean field of the work, an $L^2$ relative error of 0.037 on the velocity and of 0.345on the stress tensor against 0.048 and 0.402 for the local tensor basis and 0.103 and 0.640 for the uncalibrated classical closure, and it converges further, to a final update of $1.1 \times 10^{-3}$, Table 5.2. The vector cloud, the most accurate on reference data, is less accurate than the local closure once coupled, 0.061 and 0.442, and converges an order of magnitude less far, 8.5e-3. The single-scale graph sits in between. What carries the gain is a coarse level of a fixed number of nodes spread over the domain, so that information crosses the wake in a fixed number of rounds and the reach of the closure does not change with the mesh it is deployed on, at about three times the parameters of a single-scale graph and the same cost per evaluation. The closed-form radius of Equation (4.11), derived by symbolic regression from the wall distance and the angular position alone, is what makes the neighbourhood of the vector cloud a fixed function of the mesh and its coupling possible.

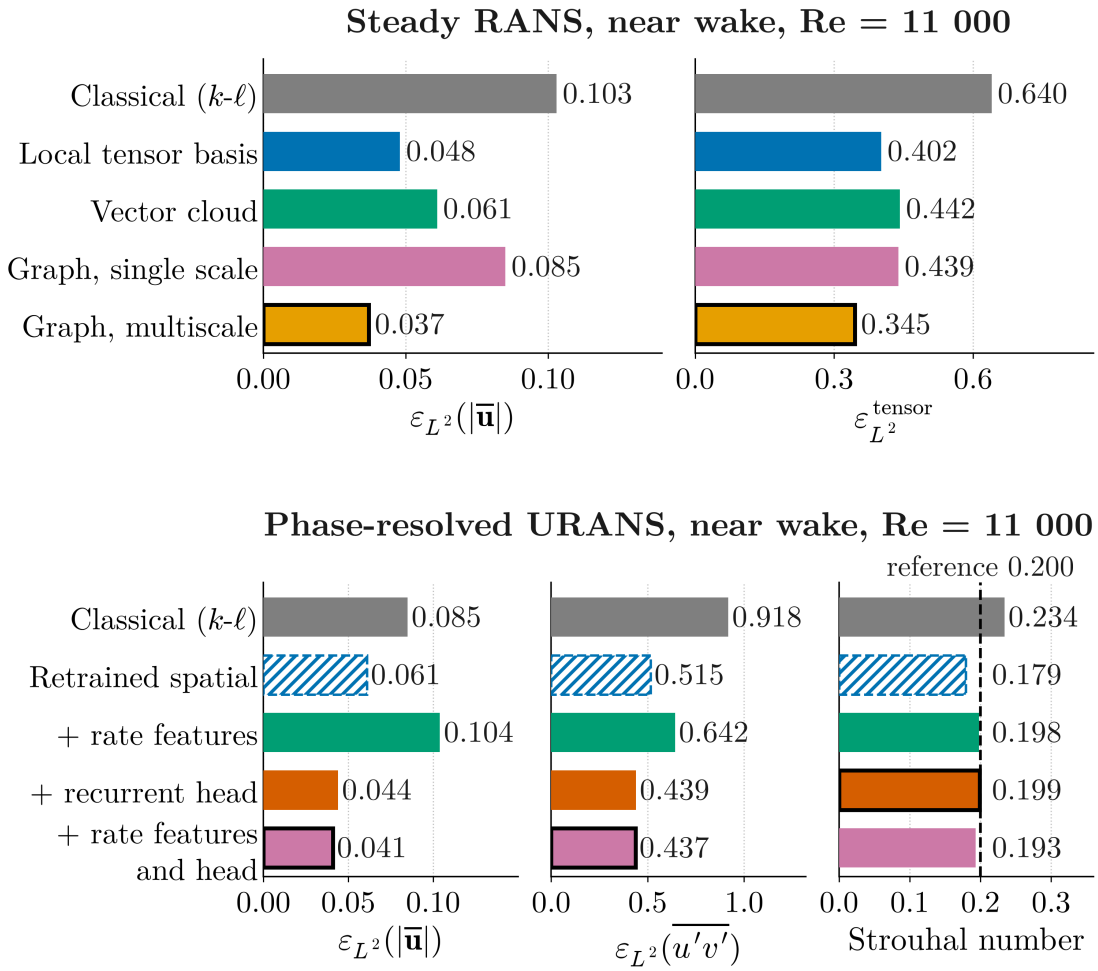
**Table 7.1:** The steady closures, what each one reads and how it performs on the four counts. Grey cells mark an ingredient the closure does not have or a metric that does not apply. Errors are relative $L^2$ errors: *a priori* on the stress tensor over the whole domain of the held-out flow, Table 5.1; *a posteriori* on the velocity magnitude and on the stress tensor of the coupled solution at the training Reynolds number, over the near-wake window of Equation (3.67), Table 5.2. The last column is the final velocity update of the iteration. *A post.* stands for *a posteriori*. Best learned value in bold.

| | What the closure reads | | | | Result on the four counts | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Closure | point | a few cells | fixed-radius neigh-bourhood | extended reach (coarse level) | *a priori* $\varepsilon^{\text{tensor}}_{L^2}$ | couples | *a post.* $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | *a post.* $\varepsilon^{\text{tensor}}_{L^2}$ | final update |
| Classical ($k$-$\ell$) | ✓ | | | | | ✓ | 0.103 | 0.640 | 1.3e-5 |
| Local tensor basis | ✓ | | | | 0.140 | ✓ | 0.048 | 0.402 | 1.7e-3 |
| Vector cloud | ✓ | | ✓ | | **0.101** | ✓ | 0.061 | 0.442 | 8.5e-3 |
| Graph, single scale | ✓ | ✓ | | | 0.140 | ✓ | 0.085 | 0.439 | 1.6e-3 |
| Graph, multiscale | ✓ | ✓ | | ✓ | 0.152 | ✓ | **0.037** | **0.345** | **1.1e-3** |

**The flow time history increases the accuracy of the coupled solver.** ***The two closures that carry a memory of the cycle are the two that reproduce the reference vortex street; the one without it never sustains a limit cycle.*** Table 7.2 and Figure 7.2 collect the evidence. On reference data a recurrent head over the past phases of the cycle leaves the aggregate error almost unchanged from the one of the retrained spatial closure, 0.472 against 0.464, and the same is for the shear stress: from 0.421 to 0.441 in the coefficient of determination. The gain is bounded by the features themselves. A floor derived from them shows that feeding the closure past phases lowers the error on the fluctuation of the stress by up to 37% while feeding it close neighbouring cells lowers it by nothing. Inside the solver the answer is decisive. Under the adopted numerics three closures out of four sustain a limit cycle, and the two that carry memory are the two that reproduce the reference. The recurrent head gives a cycle more periodic than the classical closure, a Strouhal number of 0.199 against the reference 0.200, phase-averaged velocity within five per cent of the reference over the near wake and stresses within 0.24 to 0.44, where the classical closure leaves them at 0.78 to 0.92 and the drag 40% low, Chapter 6. Convergence, read for a time march as the existence of a saturated cycle, follows the same line: the two closures with memory reach their cycle within thirty time units and hold it unchanged for the rest of the march, the closure with the rate features and no memory is as periodic but overshoots the amplitude by a factor of 3.5, while the retrained spatial closure is still growing at the end of the march.

**Table 7.2:** The phase-resolved closures, what each one reads and how it performs. Grey cells as in Table 7.1. Errors are relative $L^2$ errors: *a priori* on the stress tensor of the held-out flow, Table 6.10; *a posteriori* on the phase-averaged velocity magnitude and shear stress of the coupled solution over the near wake at the training Reynolds number, mean over the phases, Table 6.11. The retrained spatial closure does not meet the admission criterion of Equation (6.18), and its errors are those of a cycle that is still growing. Best learned value in bold.

| | What the closure reads | | | Result on the four counts | | | |
|---|---|---|---|---|---|---|---|
| Closure | point | rate features | past phases | *a priori* $\varepsilon_{L^2}^{\text{tensor}}$ | sustained cycle | *a post.* $\varepsilon_{L^2}(\lvert\overline{\mathbf{u}}\rvert)$ | *a post.* $\varepsilon_{L^2}(\overline{u'v'})$ |
| Classical ($k$-$\ell$) | ✓ | | | | ✓ | 0.085 | 0.918 |
| Retrained spatial | ✓ | | | 0.464 | × | 0.061 | 0.515 |
| + rate features | ✓ | ✓ | | 0.436 | ✓ | 0.104 | 0.642 |
| + recurrent head | ✓ | | ✓ | 0.472 | ✓ | 0.044 | 0.439 |
| + rate features and head | ✓ | ✓ | ✓ | **0.422** | ✓ | **0.041** | **0.437** |




*hatched bars: no sustained limit cycle*

**Figure 7.2:** *A posteriori* accuracy of the coupled closures, classical closure in grey, best learned value with a black border. Top, steady problem: relative $L^2$ errors on the velocity magnitude and on the stress tensor over the near-wake window of Equation (3.67), Table 5.2. Bottom, phase-resolved problem: relative $L^2$ errors on the phase-averaged velocity magnitude and shear stress over the near wake, mean over the phases on $t \in [130,180]$, and Strouhal number on $t \in [60,120]$ against the reference value, Table 6.11. The hatched bars are the retrained spatial closure, which does not reach a sustained limit cycle, so its values describe a cycle that is still growing.

**The coupling needs a stabilization built for the learned closure.** ***The filtered split of the stress with a filter on the predicted stress couples every closure of this work; the consistent stabilization alone couples none.*** With the consistent stabilization alone no learned closure of Chapter 3 converges, while the classical closure reaches the solver tolerance on the same mesh with no stabilization at all, so the whole of the gap belongs to the pair of closure and solver and not to the discretization. The filtered split of Equation (3.77) is adopted together with a filter on the predicted stress. In this formulation, the closure introduces a dissipation proportional to its own eddy viscosity on the smallest resolved scales. Under this configuration, all closures of the steady problem except the simple MLP produce a bounded field. The configuration is therefore fixed and applied unchanged to all closures. In the phase-resolved problem the same term is what separates a march that never sustains a limit cycle, under the exact split, in any filtering configuration and at any number of corrector sub-iterations, from one in which three closures out of four do. The numerics are therefore part of the model, a comparison of closures is meaningful only once they are fixed, and a learned closure should be designed with the term through which it will act on the momentum equation in view. This remedy belongs to the numerical route of Section 1.1.3. Where the literature combines a continuation method, a modified transport equation or a constraint imposed at training, here a single term serves every closure of this work and both forms of the averaged equations.

## 7.3 Outlook

**One closure, non-local in space and recurrent in time.** The two halves of this thesis reached their best results with two different ingredients, a receptive field that spans the domain at a fixed cost in the steady problem and a memory of past phases in the unsteady one, and nothing in either result excludes the other. A closure that reads a multiscale graph, or a vector cloud, at every phase and carries a recurrent state across phases would combine the two, and its pieces exist: the graph and cloud constructions are frame-invariant by design and evaluated once per mesh, and the recurrent head is a module that reads the coefficients of whatever network precedes it. Trained on phase-resolved data, such a closure would be the most physically sound among all the current machine learning closure, ideally accurate on the stress and not only on the mean field. It is the definitive model this work argues for, and the one its results make the natural next step.

**The remaining experiments.** These same closures but trained with the solver in the loop, or through the adjoint of the discrete equations, on the fields the iteration produces rather than on reference fields alone, is the direct answer to the

amplification of the error. The physics-informed variant, explored in this thesis, is a natural cheap try in this direction, and should be investigated further. A study of the filter length under the filtered split, and of a hyperviscosity with a coefficient of its own, would say how much of the accuracy of the phase-resolved closures belongs to the closure and how much to the term. A loss on the fluctuation of the stress within the cycle and a deeper temporal window are the two changes the floor analysis singles out, together with a deployment of the recurrent state that matches its training. Training the graph closures on the connectivity of the solver mesh, or rescaling their number of rounds to a fixed physical reach, would settle what the transfer between meshes costs, and a halo exchange for the cloud and for the coarse level is what a distributed implementation requires. A dataset spanning more geometries than the one studied here is the condition under which these comparisons could be repeated with generalization to a new flow as the test.

The closure problem still waits for a single model that holds across geometries and Reynolds numbers. This work argues that such a model will read beyond the point and remember the cycle, since both properties were found to pay wherever they were tested, and that a closure carrying them both is now within reach.

# Bibliography


[1] Stephen B. Pope. *Turbulent Flows*. Cambridge University Press, 2000 (cit. on pp. 1, 2, 64, 65).

[2] Philippe R. Spalart. «Strategies for turbulence modelling and simulations». In: *International Journal of Heat and Fluid Flow* 21.3 (2000), pp. 252–263. DOI: 10.1016/S0142-727X(00)00007-2 (cit. on pp. 1, 3).

[3] Zhen Zhang, Theo Käufer, Louise Ronglan, Michael S. Triantafyllou, and George Em Karniadakis. *Generalizable Turbulence Closures across Bluff-Body Shapes by PINN-Based Solver-Agnostic Training*. Under consideration for the Journal of Fluid Mechanics. 2026. arXiv: 2607.04491 (cit. on pp. 1, 4–6, 37, 38, 40, 42, 46, 50, 54, 59, 67, 157).

[4] Julia Ling, Andrew Kurzawski, and Jeremy Templeton. «Reynolds averaged turbulence modelling using deep neural networks with embedded invariance». In: *Journal of Fluid Mechanics* 807 (2016), pp. 155–166 (cit. on pp. 1–3, 18, 48, 49).

[5] T. J. Craft, B. E. Launder, and K. Suga. «Development and application of a cubic eddy-viscosity model of turbulence». In: *International Journal of Heat and Fluid Flow* 17.2 (1996), pp. 108–115. DOI: 10.1016/0142-727X(95)00079-6 (cit. on pp. 1, 2).

[6] C. H. K. Williamson. «Vortex Dynamics in the Cylinder Wake». In: *Annual Review of Fluid Mechanics* 28 (1996), pp. 477–539. DOI: 10.1146/annurev.fl.28.010196.002401 (cit. on p. 1).

[7] Karthik Duraisamy, Gianluca Iaccarino, and Heng Xiao. «Turbulence Modeling in the Age of Data». In: *Annual Review of Fluid Mechanics* 51 (2019), pp. 357–377. DOI: 10.1146/annurev-fluid-010518-040547 (cit. on pp. 2, 4).

[8] Steven L. Brunton, Bernd R. Noack, and Petros Koumoutsakos. «Machine Learning for Fluid Mechanics». In: *Annual Review of Fluid Mechanics* 52 (2020), pp. 477–508. DOI: 10.1146/annurev-fluid-010719-060214 (cit. on p. 2).

[9] B. E. Launder and D. B. Spalding. «The Numerical Computation of Turbulent Flows». In: *Computer Methods in Applied Mechanics and Engineering* 3.2 (1974), pp. 269–289. DOI: 10.1016/0045-7825(74)90029-2 (cit. on pp. 2, 17).

[10] David C. Wilcox. «Reassessment of the Scale-Determining Equation for Advanced Turbulence Models». In: *AIAA Journal* 26.11 (1988), pp. 1299–1310. DOI: 10.2514/3.10041 (cit. on pp. 2, 17).

[11] Florian R. Menter. «Two-Equation Eddy-Viscosity Turbulence Models for Engineering Applications». In: *AIAA Journal* 32.8 (1994), pp. 1598–1605. DOI: 10.2514/3.12149 (cit. on pp. 2, 17).

[12] Stephen B. Pope. «A More General Effective-Viscosity Hypothesis». In: *Journal of Fluid Mechanics* 72.2 (1975), pp. 331–340. DOI: 10.1017/S0022112075003382 (cit. on pp. 2, 18, 36).

[13] Sofiane Benhamadouche. «On the use of (U)RANS and LES approaches for turbulent incompressible single phase flows in nuclear engineering applications». In: *Nuclear Engineering and Design* 312 (2017), pp. 2–11. DOI: 10.1016/j.nucengdes.2016.11.002 (cit. on pp. 3, 7).

[14] Eric J. Parish and Karthik Duraisamy. «A paradigm for data-driven predictive modeling using field inversion and machine learning». In: *Journal of Computational Physics* 305 (2016), pp. 758–774. DOI: 10.1016/j.jcp.2015.11.012 (cit. on pp. 3, 5).

[15] Jian-Xun Wang, Jin-Long Wu, and Heng Xiao. «Physics-informed machine learning approach for reconstructing Reynolds stress modeling discrepancies based on DNS data». In: *Physical Review Fluids* 2 (2017), p. 034603. DOI: 10.1103/PhysRevFluids.2.034603 (cit. on p. 3).

[16] Jian-Xun Wang, Jin-Long Wu, Julia Ling, Gianluca Iaccarino, and Heng Xiao. «Physics-informed machine learning for predictive turbulence modeling: Towards a complete framework». In: *Proceedings of the Summer Program 2016, Center for Turbulence Research*. Stanford University, 2016 (cit. on pp. 3, 4).

[17] Jack Weatheritt and Richard D. Sandberg. «A novel evolutionary algorithm applied to algebraic modifications of the RANS stress–strain relationship». In: *Journal of Computational Physics* 325 (2016), pp. 22–37. DOI: 10.1016/j.jcp.2016.08.015 (cit. on pp. 3, 5).

[18] Mikael L. A. Kaandorp and Richard P. Dwight. «Data-driven modelling of the Reynolds stress tensor using random forests with invariance». In: *Computers & Fluids* 202 (2020), p. 104497. DOI: 10.1016/j.compfluid.2020.104497 (cit. on pp. 3, 4).

[19] Fredrik Dubay Myklebust, David R. Fuhrman, and Yuzhu Pearl Li. «Tensor basis neural networks for unsteady turbulent flow prediction». In: *Physics of Fluids* 37.7 (2025), p. 075122. DOI: 10.1063/5.0275400 (cit. on pp. 4, 7, 21).

[20] Paul Fischer et al. «NekRS, a GPU-Accelerated Spectral Element Navier-Stokes Solver». In: *Parallel Computing* 114 (2022), p. 102982. DOI: 10.1016/j.parco.2022.102982 (cit. on pp. 4, 24).

[21] Zhen Zhang et al. «Turbulence closure in Reynolds-averaged Navier–Stokes and flow inference around a cylinder using physics-informed neural networks and sparse experimental data». In: *Journal of Fluid Mechanics* 1034 (2026) (cit. on pp. 4, 5, 20, 24–26, 39, 42, 58, 59).

[22] Jin-Long Wu, Heng Xiao, Rui Sun, and Qiqi Wang. «Reynolds-averaged Navier–Stokes equations with explicit data-driven Reynolds stress closure can be ill-conditioned». In: *Journal of Fluid Mechanics* 869 (2019), pp. 553–586 (cit. on pp. 4, 58).

[23] Alexander N. Brooks and Thomas J. R. Hughes. «Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier–Stokes equations». In: *Computer Methods in Applied Mechanics and Engineering* 32.1–3 (1982), pp. 199–259. DOI: 10.1016/0045-7825(82)90071-8 (cit. on pp. 5, 153).

[24] Maziar Raissi, Paris Perdikaris, and George E. Karniadakis. «Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations». In: *Journal of Computational Physics* 378 (2019), pp. 686–707. DOI: 10.1016/j.jcp.2018.10.045 (cit. on pp. 5, 39).

[25] Maziar Raissi, Paris Perdikaris, Nazanin Ahmadi, and George Em Karniadakis. *Physics-Informed Neural Networks and Extensions*. 2024. arXiv: 2408.16806 [cs.LG] (cit. on p. 5).

[26] Xiaowei Jin, Shengze Cai, Hui Li, and George Em Karniadakis. «NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations». In: *Journal of Computational Physics* 426 (2021), p. 109951. DOI: 10.1016/j.jcp.2020.109951 (cit. on p. 5).

[27] Mouhammad El Hassan, Ali Mjalled, Philippe Miron, and Martin Mönnigmann. «Machine Learning in Fluid Dynamics: Physics-Informed Neural Networks (PINNs) Using Sparse Data: A Review». In: *Fluids* 10.9 (2025), p. 226. DOI: 10.3390/fluids10090226 (cit. on p. 5).

[28] Sifan Wang, Xinling Yu, and Paris Perdikaris. «When and why PINNs fail to train: A neural tangent kernel perspective». In: *Journal of Computational Physics* 449 (2022), p. 110768. DOI: 10.1016/j.jcp.2021.110768 (cit. on p. 5).

[29] Patricio Clark Di Leoni, Tamer A. Zaki, George Karniadakis, and Charles Meneveau. «Two-point stress–strain-rate correlation structure and non-local eddy viscosity in turbulent flows». In: *Journal of Fluid Mechanics* 914 (2021), A6. DOI: 10.1017/jfm.2020.977 (cit. on p. 6).

[30] Fangying Song and George Em Karniadakis. *A Universal Fractional Model of Wall-Turbulence*. 2018. arXiv: 1808.10276 [physics.flu-dyn] (cit. on p. 6).

[31] Xu-Hui Zhou, Jiequn Han, and Heng Xiao. «Frame-independent vector-cloud neural network for nonlocal constitutive modeling on arbitrary grids». In: *Computer Methods in Applied Mechanics and Engineering* (2021) (cit. on pp. 6, 18, 98, 101).

[32] Jiequn Han, Xu-Hui Zhou, and Heng Xiao. «An equivariant neural operator for developing nonlocal tensorial constitutive models». In: *Journal of Computational Physics* 488 (2023), p. 112243. DOI: 10.1016/j.jcp.2023.112243. arXiv: 2201.01287 (cit. on p. 6).

[33] Huaiqian You, Yue Yu, Marta D'Elia, Tian Gao, and Stewart Silling. «Nonlocal kernel network (NKN): A stable and resolution-independent deep neural network». In: *Journal of Computational Physics* 469 (2022), p. 111536. DOI: 10.1016/j.jcp.2022.111536 (cit. on p. 6).

[34] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. «The Graph Neural Network Model». In: *IEEE Transactions on Neural Networks* 20.1 (2008), pp. 61–80. DOI: 10.1109/TNN.2008.2005605 (cit. on pp. 6, 19).

[35] Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W. Battaglia. «Learning Mesh-Based Simulation with Graph Networks». In: *International Conference on Learning Representations (ICLR)*. 2021. arXiv: 2010.03409 (cit. on p. 6).

[36] Meire Fortunato, Tobias Pfaff, Peter Wirnsberger, Alexander Pritzel, and Peter Battaglia. «MultiScale MeshGraphNets». In: *2nd AI4Science Workshop at ICML*. 2022. arXiv: 2210.00612 (cit. on pp. 6, 19, 20, 129, 130).

[37] Víctor Garcia Satorras, Emiel Hoogeboom, and Max Welling. «E(n) Equivariant Graph Neural Networks». In: *Proceedings of the 38th International Conference on Machine Learning*. Vol. 139. PMLR. 2021. arXiv: 2102.09844 (cit. on pp. 6, 124).

[38] A. K. M. F. Hussain and W. C. Reynolds. «The Mechanics of an Organized Wave in Turbulent Shear Flow». In: *Journal of Fluid Mechanics* 41.2 (1970), pp. 241–258. DOI: 10.1017/S0022112070000605 (cit. on pp. 7, 16).

[39] Brian Cantwell and Donald Coles. «An Experimental Study of Entrainment and Transport in the Turbulent Near Wake of a Circular Cylinder». In: *Journal of Fluid Mechanics* 136 (1983), pp. 321–374. DOI: 10.1017/S0022112083002189 (cit. on pp. 7, 16).

[40] R. Perrin, M. Braza, E. Cid, S. Cazin, A. Barthet, A. Sevrain, C. Mockett, and F. Thiele. «Obtaining Phase Averaged Turbulence Properties in the Near Wake of a Circular Cylinder at High Reynolds Number Using POD». In: *Experiments in Fluids* 43.2–3 (2007), pp. 341–355. DOI: 10.1007/s00348-007-0347-6 (cit. on pp. 7, 16).

[41] Daniel M. Israel. «The Myth of URANS». In: *Journal of Turbulence* 24.8 (2023), pp. 367–392. DOI: 10.1080/14685248.2023.2225140 (cit. on pp. 7, 15, 16).

[42] Zhi-Feng Wei, Saad Qadeer, and Panos Stinis. «Mamba-Assisted Non-Markovian Closure for Reduced-Order Modeling». In: *arXiv preprint arXiv:2606.05371* (2026) (cit. on pp. 7, 22).

[43] Sepp Hochreiter and Jürgen Schmidhuber. «Long Short-Term Memory». In: *Neural Computation* 9.8 (1997), pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735 (cit. on pp. 7, 21).

[44] Albert Gu and Tri Dao. «Mamba: Linear-Time Sequence Modeling with Selective State Spaces». In: *arXiv preprint arXiv:2312.00752* (2023) (cit. on pp. 7, 21).

[45] Renzo Arina. *Fondamenti di Aerodinamica*. Levrotto & Bella, 2015 (cit. on p. 11).

[46] Roger Temam. *Navier-Stokes Equations: Theory and Numerical Analysis*. Vol. 2. American Mathematical Society, 2001 (cit. on p. 12).

[47] Charles L. Fefferman. *Existence and Smoothness of the Navier–Stokes Equation*. Tech. rep. Official problem description, Millennium Prize Problems. Clay Mathematics Institute, 2000 (cit. on p. 12).

[48] Anthony T. Patera. «A Spectral Element Method for Fluid Dynamics: Laminar Flow in a Channel Expansion». In: *Journal of Computational Physics* 54.3 (1984), pp. 468–488. DOI: 10.1016/0021-9991(84)90128-1 (cit. on p. 25).

[49] George Em Karniadakis and Spencer J. Sherwin. *Spectral/hp Element Methods for Computational Fluid Dynamics*. 2nd. Oxford University Press, 2005 (cit. on p. 25).

[50] Wenbo Cao and Weiwei Zhang. *TSONN: Time-stepping-oriented neural network for solving partial differential equations*. 2023. arXiv: 2310.16491 (cit. on p. 40).

[51] Sifan Wang, Shawn Koohy, Yiping Lu, and Paris Perdikaris. *When PINNs go wrong: Pseudo-time stepping against spurious solutions*. 2026. arXiv: 2604.23528 [cs.LG] (cit. on p. 40).

[52] Nicholas J. Higham. «Computing a Nearest Symmetric Positive Semidefinite Matrix». In: *Linear Algebra and its Applications* 103 (1988), pp. 103–118. DOI: 10.1016/0024-3795(88)90223-6 (cit. on p. 47).

[53] Howard C. Elman, David J. Silvester, and Andrew J. Wathen. *Finite Elements and Fast Iterative Solvers: With Applications in Incompressible Fluid Dynamics*. 2nd. Oxford, UK: Oxford University Press, 2014 (cit. on pp. 58, 66).

[54] Suhas V. Patankar. *Numerical Heat Transfer and Fluid Flow*. Washington, DC: Hemisphere Publishing Corporation, 1980 (cit. on p. 58).

[55] Eugene L. Allgower and Kurt Georg. *Numerical Continuation Methods: An Introduction*. Vol. 13. Springer Series in Computational Mathematics. Berlin, Heidelberg: Springer-Verlag, 1990 (cit. on p. 58).

[56] Jørgen S. Dokken. *The FEniCSx Tutorial*. Accessed 2026. 2023. URL: https://jsdokken.com/dolfinx-tutorial/ (cit. on p. 64).

[57] John G. Heywood, Rolf Rannacher, and Stefan Turek. «Artificial boundaries and flux and pressure conditions for the incompressible Navier–Stokes equations». In: *International Journal for Numerical Methods in Fluids* 22.5 (1996), pp. 325–352. DOI: 10.1002/(SICI)1097-0363(19960315)22:5<325::AID-FLD307>3.0.CO;2-Y (cit. on p. 64).

[58] David Gottlieb and Chi-Wang Shu. «On the Gibbs Phenomenon and Its Resolution». In: *SIAM Review* 39.4 (1997), pp. 644–668 (cit. on p. 66).

[59] Philippe G. Ciarlet. *The Finite Element Method for Elliptic Problems*. SIAM, 2002 (cit. on p. 66).

[60] Vivette Girault and Pierre-Arnaud Raviart. *Finite Element Methods for Navier-Stokes Equations: Theory and Algorithms*. Springer, 1986 (cit. on p. 66).

[61] L. Ridgway Scott. «Finite Element Convergence for Singular Data». In: *Numerische Mathematik* 21 (1973), pp. 317–327 (cit. on p. 67).

[62] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. «Spectral Normalization for Generative Adversarial Networks». In: *International Conference on Learning Representations (ICLR)*. 2018. arXiv: 1802.05957 (cit. on p. 87).

[63] Dan Hendrycks and Kevin Gimpel. *Gaussian Error Linear Units (GELUs)*. 2016. arXiv: 1606.08415 [cs.LG]. URL: https://arxiv.org/abs/1606.08415 (cit. on p. 101).

[64] Diederik P. Kingma and Jimmy Ba. «Adam: A Method for Stochastic Optimization». In: *3rd International Conference on Learning Representations (ICLR)*. 2015. arXiv: 1412.6980 [cs.LG] (cit. on pp. 103, 134).

[65] Miles Cranmer. *Interpretable Machine Learning for Science with PySR and SymbolicRegression.jl*. 2023. arXiv: 2305.01582 [astro-ph.IM]. URL: https://arxiv.org/abs/2305.01582 (cit. on p. 113).

[66] Tayfun E. Tezduyar. «Stabilized Finite Element Formulations for Incompressible Flow Computations». In: *Advances in Applied Mechanics* 28 (1991), pp. 1–44. DOI: 10.1016/S0065-2156(08)70153-4 (cit. on p. 153).

[67] Patrick R. Amestoy, Iain S. Duff, Jean-Yves L'Excellent, and Jacko Koster. «A fully asynchronous multifrontal solver using distributed dynamic scheduling». In: *SIAM Journal on Matrix Analysis and Applications* 23.1 (2001), pp. 15–41 (cit. on p. 159).

[68] Patrick R. Amestoy, Alfredo Buttari, Jean-Yves L'Excellent, and Theo Mary. «Performance and scalability of the block low-rank multifrontal factorization on multicore architectures». In: *ACM Transactions on Mathematical Software* 45.1 (2019), 2:1–2:26 (cit. on p. 159).

[69] Satish Balay, William D. Gropp, Lois Curfman McInnes, and Barry F. Smith. «Efficient management of parallelism in object oriented numerical software libraries». In: *Modern Software Tools in Scientific Computing*. Ed. by Erlend Arge, Are Magnus Bruaset, and Hans Petter Langtangen. Birkhäuser, 1997, pp. 163–202 (cit. on p. 159).

[70] M. Provansal, C. Mathis, and L. Boyer. «Bénard–von Kármán instability: transient and forced regimes». In: *Journal of Fluid Mechanics* 182 (1987), pp. 1–22. DOI: 10.1017/S0022112087002222 (cit. on p. 188).

[71] K. R. Sreenivasan, P. J. Strykowski, and D. J. Olinger. «Hopf bifurcation, Landau equation, and vortex shedding behind circular cylinders». In: *Forum on Unsteady Flow Separation*. Vol. 52. New York: ASME, Fluids Engineering Division, 1987, pp. 1–13 (cit. on p. 188).