POLITECNICO DI TORINO

Master of Science in Automotive Engineering

Master Thesis

Modelling of a Heavy Duty Diesel engine and its conversion to Real-Time model

Supervisors:
prof. Federico MILLO - POLITECNICO DI TORINO
prof. Mika LAUREN — TURKU UNIVERSITY OF APPLIED SCIENCES

Candidate:
Marco BRASACCHIO

April 2020

"A mia mamma ed a mio papà, ai quali sarò sempre grato per gli innumerevoli sacrifici.
A mio fratello ed a mia sorella, amorevoli guide e miei secondi genitori.
Alla mia neonata nipotina Giulia, che la vita ti
sorrida sempre, gioia di zio!"

Marco

Abstract

This thesis work is focused on the realization of the Real-Time model of a heavy duty engine by means of the by now established CFD zero/mono-dimensional numerical modelling software GT-Suite (Gamma Technologies). Standard models, in fact, require a remarkable computational effort for the calculation of the several variables that characterize a complex system such as an internal combustion engine, and so very long times. This is for sure a big handicap for the research process since it limits the productivity of the researcher during the many experiments to conduct but, more than anything else, it hinders totally the use of Hardware-in-the-Loop investigation strategies (HiL). Such methodologies allow the testing of the ECUs (Electronic Control Units) without a real engine available. Substantially, the model developed in this thesis work emulates part of the test bench (the engine with all his sensors and actuators) "deceiving" the ECU. Parallel scopes have been to provide a full load torque curve typical of these engines, real maps of fuel consumption and of the engine energy balance, and to enable the model to run transient simulations. The engine simulated in this thesis work is a turbocharged AGCO Diesel 4 cylinders available at Turku University of Applied Sciences, Finnish University in which the whole thesis work was carried out. First of all, a detailed GT-Power model of such an engine was built, acquiring 6 experimental working points, the macro-geometries of the engine and the compressor map, calibrating then the whole intake system inclusive of the intercooler through available experimental pressure and temperature data in many points. Subsequently, since the aim of the final model was exclusively to emulate an already existing engine, it was decided to use a non-predictive or a semi-predictive combustion model. In absence of the measuring instruments for acquiring the burn rate profile, necessary for a non-predictive model, the second model prevailed, creating a MATLAB code capable of providing injection profiles as a function of the rail pressure and of the injected quantities, starting from injection profiles acquired experimentally for a similar injector. After having started the conversion process to Fast Running Model, that should have ended once reached a Factor of Real-Time smaller or equal to one, the need of converting it to a Mean Value model has emerged because the semi-predictive nature of the combustion model was limiting considerably the running time of the model. The thesis work ended verifying the accuracy of the Real-Time model compared to the detailed one.

Sommario

Questo lavoro di tesi è incentrato sullo sviluppo del modello Real-Time di un motore Diesel heavy duty mediante l'ormai ben affermato codice di simulazione numerica CFD zero/monodimensionale GT-Suite (Gamma Technologies). Modelli standard, infatti, richiedono un notevole sforzo computazionale per il calcolo delle numerose variabili che caratterizzano un sistema complesso come lo è un motore a combustione interna, quindi tempi molto lunghi. Questo è sì un grande ostacolo per la ricerca in quanto limita la produttività dello studioso del fenomeno durante i molti esperimenti da condurre ma, in misura più grave, nega totalmente l'accesso a strategie di investigazione Hardware-in-the-Loop (HiL). Tali metodologie permettono il testing delle ECU (Electronic Control Unit) prima (o senza) che il motore reale sia disponibile. In sostanza, il modello sviluppato in questo lavoro di tesi emula parte del banco prova (il motore con tutti i suoi relativi sensori e attuatori) "ingannando" la centralina elettronica. Scopi paralleli sono stati il fornire una curva di coppia a pieno carico tipica di questo tipo di motori, mappe reali dei consumi e del bilancio energetico, e di abilitare il modello a simulazioni di natura transitoria. Il motore simulato in questo lavoro di tesi è un AGCO Diesel 4 cilindri sovralimentato disponibile fisicamente alla Turku University of Applied Sciences, Università finlandese nella quale è stato condotto l'intero studio di tesi. È stato innanzitutto costruito un modello GT-Power dettagliato di suddetto motore, acquisendo 6 punti sperimentali di funzionamento, le macro-geometrie dello stesso e la mappa del compressore, per poi calibrare l'intero sistema di aspirazione comprensivo dell'intercooler attraverso le misure sperimentali di pressione e temperatura disponibili in vari punti. Successivamente, sicché lo scopo del modello finale era esclusivamente di emulare un motore già esistente, si è optato per usare un modello di combustione non-predittivo o semi-predittivo. In assenza degli strumenti di misura necessari per acquisire il burn rate, necessario per un modello non-predittivo, è prevalso il secondo modello, generando un codice MATLAB capace di fornire profili di iniezione al variare della pressione e della quantità di massa iniettata partendo da dei profili acquisiti sperimentalmente disponibili per lo stesso modello di iniettore. Iniziato il processo di conversione a Fast Running Model, che si sarebbe concluso al raggiungimento di un Factor of Real-Time più piccolo o uguale a uno, è infine emersa la necessità di convertire il modello in Mean Value model poiché la natura semi-predittiva del modello di combustione limitava notevolmente la velocità di esecuzione del modello. Lo studio di tesi si è concluso verificando la precisione del modello Real-Time a confronto con il modello dettagliato.

Contents

Abstract IV
Sommario VI
List of Figures XII
List of Tables XV
1 Introduction 1
2 GT-Suite simulation environment 4
2.1 What is GT-Suite 4
2.2 GT-Suite structure and applications 4
2.3 Flow solver 5
2.4 Combustion models 6
2.4.1 Non-predictive combustion models for Diesel applications 8
2.4.2 Semi-predictive combustion model for Diesel applications 9
2.5 Heat transfer models 9
2.5.1 WallTemperature 9
2.5.2 Heat transfer object 10
3 Detailed GT-Power model 11
3.1 Engine specifications 11
3.2 PC specifications 12
3.3 Model building 12
3.3.1 The test bench 14
3.3.2 Experimental runs 14
3.3.3 Intercooler pressure drop calibration 15
3.3.4 Intercooler effectiveness calibration 16
3.3.5 Turbocharger group 17
3.3.6 Intermediate optimization of the model 18
3.3.7 Combustion model 20
3.3.8 Injector model 21
3.3.9 Heat transfer model 24
3.3.10 Results of the detailed model 25
4 Mean Value (real-time) Model 29
4.1 Introduction to Fast Running Models (FRM) and Mean Value Models 29
4.2 Procedure followed for the conversion to Fast Running Model 30
4.3 Conversion to Mean Value model 37
4.3.1 DOE setup and Neural Networks training 37
4.3.2 Neural networks training 39
4.3.3 Neural networks configuration 40

4.3.4 Additional changes 42
4.4 Transients 43
4.4.1 Transients initialization 43
4.5 Accuracy check and results of the Mean Value model 43
Conclusions 47
Appendix 48
References 52
Thanks 54

Acronyms

bmep	Brake mean effective pressure
bsfc	Brake specific fuel consumption
CFD	Computational Fluid Dynamics
CI	Compression Ignition
DI	Direct Injection
DoE	Design of Experiments
fmep	Friction mean effective pressure
h	Heat transfer coefficient
imep	Indicated mean effective pressure
$\dot{m}$	Boundary mass flux into the volume
$\dot{m}_{air}$	Air mass flow
p	Pressure
P_{compr}	Pressure after compressor
PFI	Port Fuel Injection
pmep	Pumping mean effective pressure
T_{ai}	Air temperature at intercooler inlet
T_{ao}	Air temperature at intercooler outlet
T_c	Intercooler coolant temperature
T_{compr}	Temperature after compressor
T_{fluid}	Fluid temperature
T_{wall}	Wall temperature
T_{eng}	Engine brake torque
V	Volume
WG	WasteGate
ρ	Density

List of Figures

Figure 1-1: ECUs development process. 2
Figure 2-1: The libraries (on the left) and the template-object-part hierarchy (on the right). 4
Figure 2-2: Staggered grid approach. 6
Figure 2-3: Heat transfer models setting. 9
Figure 3-1: Test bench and engine. 11
Figure 3-2: Preliminary detailed GT-Suite model with schematization. The black dots represent the position of the pressure and temperature sensors. 13
Figure 3-3:Model for intercooler pressure drop calibration. 15
Figure 3-4: Model for intercooler heat transfer calibration. 16
Figure 3-5: TurbineSimple connection. 18
Figure 3-6: Model optimization: before. 19
Figure 3-7: Factor of real-time before and after the intermediate optimization. 19
Figure 3-8: EMI curves of the injector under investigation. 22
Figure 3-9: Experimental injection profiles. 23
Figure 3-10: Injection rate profiles generated with MATLAB. 24
Figure 3-11: Full load torque and power curves. 26
Figure 3-12: Detailed model engine map. 26
Figure 3-13: Engine energy balance at full load. 27
Figure 3-14: Engine energy balance at part load. 28
Figure 3-15: Experimental - simulated data comparison. 28
Figure 3-16: Experimental - simulated data comparison (2). 29
Figure 4-1: Tags predefined in GT-Suite. 32
Figure 4-2: Components tagging. 33
Figure 4-3: Workflow for the conversion to FRM. 35
Figure 4-4: Result of the conversion of the boost pipes. 37
Figure 4-5: Accuracy check of step 2, spreadsheet. 37
Figure 4-6: Graphical accuracy check of step 2. 37
Figure 4-7: DoE setup. 38
Figure 4-8: RMSE vs. number of experiments. 38
Figure 4-9: Neural networks input. 39
Figure 4-10: Neural networks filtering. 40
Figure 4-11: Neural networks fitting check. 40
Figure 4-12: Neural networks subassembly. 41
Figure 4-13: Control system of the Mean Value model. 41

Figure 4-14: Model with active accelerator and torque, power, bsfc and heat rejection monitors. 42
Figure 4-15: Mean Value model: factor of Real-Time in the engine map. 44
Figure 4-16: Factor of Real-Time for the three evolution steps of the model averaged in the six experimental cases. 44
Figure 4-17: Performances comparison of detailed and MV models. 45
Figure 4-18: Temperature and pressure comparison of detailed and MV models. 46

List of Tables

Table 3-1: Engine characteristics. 12
Table 3-2: Computer characteristics. 12
Table 3-3: Experimental data. 14
Table 3-4: Intercooler pressure drops. 15
Table 3-5: Temperatures after compressor. 16
Table 3-6: IntercoolerEff map. 17
Table 4-1: Key RLTs for accuracy check. 34

1 Introduction

Computer Aided Engineering (CAE) is nowadays one of the key components for the success of companies and in general for the research and development processes. From its first applications in the late '50s, it was clear that such "fast" computers, if correctly used, were going to solve the big issue of making computations by hand, which required very long times and the total recomputing if mistakes were made. Its further applications enabled also to reduce the huge costs of prototyping and of the experiments. This last case matches exactly the topic of this thesis, CAE applied to the engines field for reducing experiments number and costs.

Software capable of fully simulating engines are nowadays widespread in the R&D field. Computational Fluid Dynamics programs (such as GT-Suite, used for this thesis) are indeed highly employed for investigation strategies of all kinds. The recent shortening of the time-to-market of a product to 2 years on average requires the research process to be speeded up considerably. For example, through these software, the spark-timing map can be now created before the first real engine is available and fully working, or the best cam profile can be found by means of "free" tests rather than manufacturing many of them and physically test each one. Further, CAE application eliminates the huge costs of experimental campaigns due to the installation of real engines in the test bench, gasoline, electricity for the electric motor brake, time losses for its start-up, warm-up and runs, training courses for researchers aimed to use it, space that should be given to each test bench etc. Finally, even when a real engine is already available, testing algorithms on a model can avoid damages to the engine caused by wrong control strategies.

Current systems endowed with engines interact largely with electric/electronic components for their control as this can guarantee huge benefits in terms of performance, fuel economy, safety. Indeed, dedicated ECUs (Electronic Control Units) receive and send signals from the engine: for example, the throttle controller in drive-by-wire systems send and receive hundreds of times per second respectively the signal to control the DC motor to regulate the throttle opening angle and the position signal from a potentiometer, in a closed-loop strategy, until the wanted opening angle is achieved. Such control strategies need to be thoroughly tested before the series production starts. The recent shortening of the time-to-market above mentioned doesn't allow anymore wastes of time such as waiting for the real engine to be ready for starting the sensors/actuators and ECUs testing. Model/Software/Hardware-in-the-Loop (MiL, SiL, HiL) strategies have been then introduced in order to simulate systems not yet physically available, such as the case of this work, an internal combustion engine.

The ECUs development process has usually the following steps:

- MiL: both the controller (ECU) and the engine are models designed in a PC, usually in the form of MathWorks Simulink assemblies of blocks. In this step the controller logic is tested.
- SiL: a code of the controller assembly is generated and replaces the previous block. This step gives the idea of whether the controller model can be converted to code and if it is hardware implementable.
- HiL: in this step, the controller model is put on the real ECU hardware and so the final processor can be tested. In case of glitches, the algorithm is rectified re-starting from the MiL step.

In Figure 1-1 it's possible to see a graphical representation of the first three phases of the process for a better understanding. Finally, obviously, the real ECU is tested with the real engine.

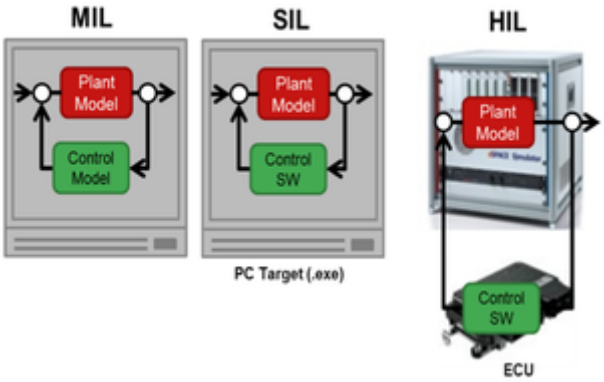

Figure 1-1: ECUs development process.

Whilst MiL and SiL steps can have a non-Real-Time model since all the system runs at the same speed, HiL requires a Real-Time one because in this phase a real hardware with real delays comes in. Standard engine models can have very high complexities such as advanced combustion models, the presence of turbochargers, of PID (Proportional-Integral-Derivative) controllers, which need high computational times that can lead to run a cycle tens of times slower than a real engine, while Real-Time models substantially compute all the cases in advance and store them into a map, so that a much lower computational time is required.

This thesis work aimed then first at creating an engine model representing accurately an already existing engine (physically available in the University in which the thesis studies have been carried out) and then in converting into a particular kind of model called

Mean Value model which allows it to be run as fast as a real engine (if not faster), to be used in a HiL test bench.

2 GT-Suite simulation environment

2.1 What is GT-Suite

The software used for this thesis work is GT-Suite, which is a set of software produced by Gamma Technologies LLC dealing with computational fluid dynamics, which main advantage is that of having libraries containing standard components that can be modelled then just setting the main characteristics of the component object of the investigation. In other words, if the aim is modelling an engine cylinder, thanks to this software there's no anymore need of creating a fluid dynamic model including CAD file of the system, of setting of the dependencies with the crank angle, mesh creation etc., it's enough to insert the bore, stroke, compression ratio and other few parameters to launch the simulation. Even though the work is highly simplified, this doesn't mean that the suite solves in complete autonomy our investigation. As with all the numerical analyses, the set parameters are very important and the logical link with the reality must never be lost.

2.2 GT-Suite structure and applications

GT-Suite is made of many libraries which are sets of *components* (or *templates*), *connections* between the components, and *references* (fluid-dynamic models and tables) regarding exactly that library. As an example, in the engine library, a component can be the engine cylinder, a connection can be the throttle valve, and a reference can be a combustion model. When a template is created with some specific values referring to the case under examination, it becomes an *object*, and a name can be given to it, for identification. When this object is then put in the project map, it will be a *part*. It is clearly illustrated in Figure 2-1.

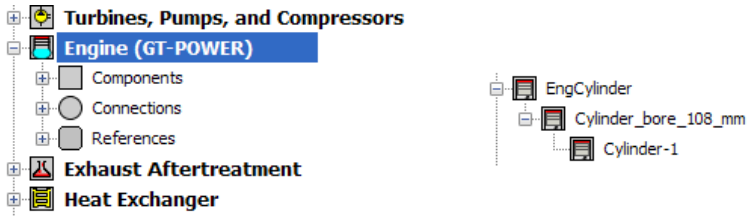

Figure 2-1: The libraries (on the left) and the template-object-part hierarchy (on the right).

Even if GT-Suite can be used with all what is a fluid circuit, its most common application is regarding engine simulations. In fact, after the common flow library, there are libraries about

- Turbines, pumps and compressors
- Engines

- Exhaust aftertreatments
- Heat exchangers
- A/C and waste heat recovery
- Cooling
- Fuel injection
- Hydraulics/pneumatics
- Lubrication
- GEM3D/COOL3D

It can then be used for many purposes, as intake/exhaust piping tuning, turbocharger matching, valve lift design, ATS design etc., since the program, after each simulation, provides crank angle or time-resolved results about power, torque, volumetric efficiency, flows in all passages, noxious emissions, noise, burn rates and much more. This allows first of all to test the influence of different parameters on an engine which is not existing either as prototype, so in the earliest phases of the product development process. Then, it allows saving huge costs that would be coming from experimental tests.

2.3 Flow solver

GT-Suite flow solution is based on a 1-dimensional model, so it provides results about the flow averaged in the cross section. The system is discretized in many volumes. Thus, every pipe volume is discretized in several subvolumes according to the set *discretization length*. A coarse discretization will result in a faster run time, but in a lower accuracy as well. However, a lower and lower discretization length will not always result in better accuracy, there's a threshold under which the accuracy will not increase anymore that much. In order to respect the *Courant condition*, the time step must be smaller than the time needed for the flow to cross the two boundaries of discretized part of the circuit, otherwise big errors in the results will be made [1]. Since the exhaust temperature are higher, the speed of sound will be higher as well, and the exhaust pipes discretization length must be smaller than the intake pipes one. Common values are 40% of the bore for the intake system pipes and 55% for the exhaust ones, to keep the time step equal all over the system, avoiding bottlenecks.

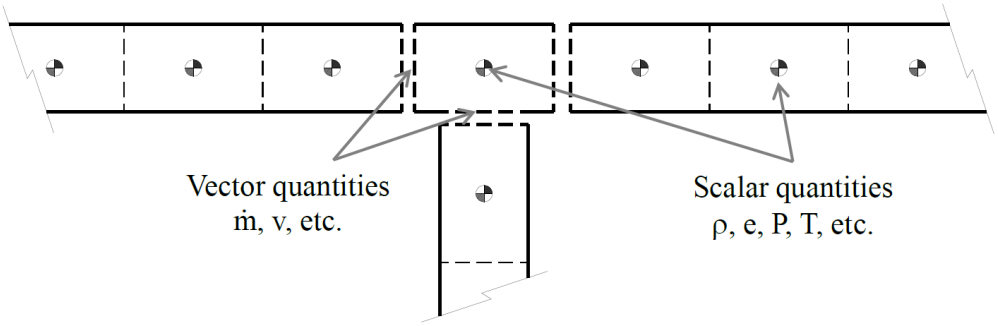

Figure 2-2: Staggered grid approach.

As shown in Figure 2-2, the scalar quantities such as pressure, temperature, density, enthalpy and species concentrations are calculated at the centroid of each subvolume and are considered uniform in the whole volume, while the vector variables such as mass flux, speed, mass fraction fluxes, are computed at each of the boundary of the pipes [2].

It is important to remark that the solution is not based on an iterative approach, indeed results are obtained by using the Navier-Stokes equations in each of the volume (continuity, energy, momentum) [2]

Continuity equation $\frac{dm}{dt} = \sum \dot{m}$

Energy equation $\frac{d(me)}{dt} = -p\frac{dV}{dt} + \sum(\dot{m}H) - hA_s\left(T_{fluid} - T_{wall}\right)$

Momentum $\frac{d\dot{m}}{dt} = \frac{dpA + \sum(\dot{m}u) - 4C_f\frac{\rho u|u|}{2}\frac{dxA}{D} - K_p\left(\frac{1}{2}\rho u|u|\right)A}{dx}$

where

- m is the mass of the volume
- e is the total specific internal energy
- H is the total specific enthalpy
- A_s is the heat transfer surface area
- A is the cross-sectional flow area
- D is the equivalent diameter
- K_p is the pressure loss coefficient
- u is the velocity at the boundary
- C_f is the fanning friction factor

2.4 Combustion models

The most difficult part/phenomenon to model in GT-Suite is the heat addition to the gas, and so the *combustion process*. It must be specified into the cylinder template

(EngCylinder) and it's in most of the cases a *two-zone combustion model*, which as the name says features a *burnt zone,* which is made of burnt, residuals and inert gases, and an *unburnt zone*, which only contains the fresh mixture of air and fuel. Two-zones combustion models are quite accurate unless we are dealing with certain phenomena such as the NO_x production, which highly depend on the temperature distribution inside the cylinder. In these cases, a multi-zone (3+) model is required [3].
One important remark must be made: during the combustion process, charge motions such as swirl, tumble and squish are very important for the fuel mixing, especially for diesel engines [4], but these can't be predicted by standard GT-Suite one-dimensional simulations.
Before starting the dissertation, it's useful to introduce some important definitions for the understanding of the different models

- **Burn rate**: it's physically the instantaneous rate of fuel consumption during the combustion. Talking about a discretized process, it's the rate at which the mixture molecules are transferred from the unburnt zone and get involved in a chemical reaction that will transform them in burnt gases.
- **Heat Release Rate**: it's the rate at which the energy in the fuel molecules is released in the cylinder as thermal energy. It differs from the burn rate because reactions, after their start, can take some time to complete and then to release/absorb heat. The HRR in fact always lags with respect to the burn rate.
- **Apparent burn rate**: it's the burn rate that would need to be set in a non-predictive simulation which has as input exactly this burn rate and as output the pressure trace to be emulated.
- **Forward Run Combustion Calculation**: it's a combustion calculation where as input is set the burn rate and the output is the cylinder pressure. It's the working principle of typical GT-Power simulations.
- **Reverse Run Combustion Calculation**: in this case the input is the in-cylinder pressure and the result is the apparent burn rate that would generate the set pressure. It's an iterative procedure which varies the amount of fuel transferred from the unburnt to the burnt zone step by step.

Depending on the intended use of the GT-Suite model and also on the sensors and tools to calibrate the combustion model, a *non-predictive*, *predictive*, or *semi-predictive* model must be chosen.

In a non-predictive combustion, what happens is a simple imposition of a burn rate as a function of the crank angle. Until the fuel inside the cylinder will be enough to let the combustion occur, this prescribed rate will be followed no matter the other variables

such as injection timing, residual fraction etc. For this reason, such combustion model can be used as long as the variable to study doesn't influence significantly the burn rate.

In a predictive combustion, instead, the burn rate is predicted by appropriate inputs such as pressure, temperature, residuals fraction, air-fuel ratio etc. This adds a big complexity to the calculations which result in longer computational times but also in a big calibration effort. For these reasons, non-predictive combustion models must always be preferred.

The semi-predictive combustion models instead can represent a compromise in some cases. These models use a non-predictive (Wiebe) methodology where the combustion rate is imposed as function of significant input variables through look-up tables with the Wiebe coefficients.

It must be noted that GT-Suite offers also the possibility of using a code developed by the user itself for the combustion.

Since the aim of this thesis work was not to study the behavior of the engine varying some parameters but only to simulate the behavior of the engine, a non-predictive combustion model would have been the preferred option.

2.4.1 Non-predictive combustion models for Diesel applications

The combustion models available in the GT-Power library for this case are the "EngCylCombProfile", "EngCylCombDIWiebe" and "EngCylCombMultiWiebe".

Imposed Combustion Profile (EngCylCombProfile)

This template is intended for the imposition of a burn rate as a function of the crank angle. It can be actually used both for spark-ignition and compression-ignition engines. It's particularly useful when the measured cylinder pressure is available, so that the burn rate profile can be generated by means of a reverse run combustion calculation.

Direct-Injection Diesel Wiebe Model (EngCylCombDIWiebe)

This model is meant to impose the burn rate for DI, CI engines to generate, by means of the proper coefficients of a three-term Wiebe function, the burn rate of a typical single injection engine. This superposition of three Wiebe functions is to make possible the shaping of the premixed and diffusion portions of the combustion process.

MultiWiebe Model (EngCylCombMultiWiebe)

This template imposes the burn rate by means of multiple Wiebe functions. The main usage of this model is for engines with a multiple injection working strategy.

2.4.2 Semi-predictive combustion model for Diesel applications

The "EngCylCombDIWiebe" template explained as non-predictive in subsection 0 becomes semi-predictive when its attributes are set to "def". These parameters will be now calculated from the injection profile, the air-fuel ratio, pressure and temperature. Thus, injector geometry and injector pressure profile must be carefully specified.

2.5 Heat transfer models

Since one of the requests of the work was to provide the engine energy balance profiles as function of speed and load, the choice of the heat transfer model was very important, even because the exhaust gas temperature also affects all the performance of the engine because of the turbine.

The heat transfer in the cylinder and in the crankcase is modeled using the combinations of two objects: one defines the temperatures of the engine block (*WallTemperature*), and the other defines how the heat transfer occurs between the various components (*Heat transfer object)*, as shown in Figure 2-3.

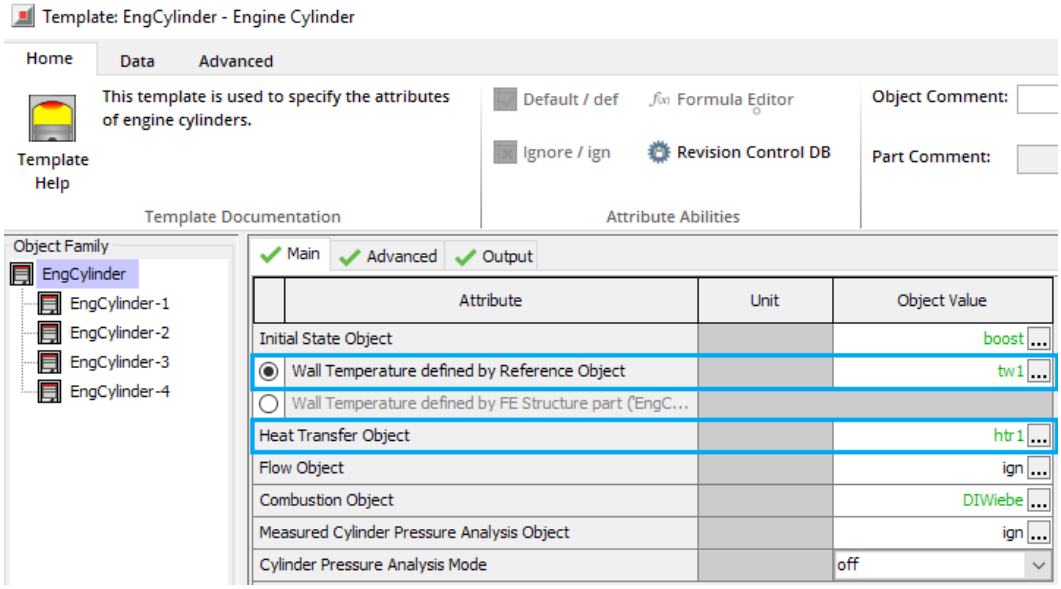

Figure 2-3: Heat transfer models setting.

2.5.1 WallTemperature

For what concerns the former object, GT-Suite provides the two following options:

EngCylTWallSoln and EngCylTWallDetail

They are both very accurate models: the first one also gives as a result the cylinder

chamber wall temperature distribution. But they're both also very demanding: they require in fact, besides common measurements such as coolant and oil temperatures in the boundaries, a very complex geometry of the whole block decomposed in its many subcomponents, each of which will have its temperature. These models are more important for PFI engines, where fuel evaporation occurs inside the ports.

EngCylTWall

This model instead consists basically in just imposing the temperatures of head, piston and cylinder walls, using some typical values if measurements are not available.

2.5.2 Heat transfer object

About the heat transfer model, instead, the following models are available in the GT-Suite library:

WoschniClassic

It uses the classical Woschni correlation without swirl, in fact it was highly recommended when swirl data were not available. before the introduction of the WoschniGT. The formula of the convective heat transfer coefficient for every Woschni model (WoschniClassic, WoschniGT, WoschniSwirl, WoschniHuber) is the following

$$h_{c(Woschni)} = \frac{K_1 p^{0.8} w^{0.8}}{B^{0.2} T^{K_2}}$$

where B is the bore and w, K_1 and K_2 are respectively the average cylinder gas velocity and two constants. These three last factors have formulas/values different from model to model.

WoschniGT

It's the most recommended nowadays when swirl data are not available. It's a variant of the WoschniClassic which, among the various characteristics, features the increase in heat transfer when the valves are open, since the flow velocity is higher.

WoschniSwirl, WoschniHuber and Flow

These are heat transfer models which use different formulas and methodologies but all based on the available data for what concerns swirl. They're not dealt thoroughly because swirl data was not available in the case of this thesis work.

Hohenberg

Hohenberg model uses a correlation quite similar to the Woschni one, but it has been observed predicting more accurate results for certain direct injection engines. In rotational regimes close to zero, the minimum heat transfer coefficient is in this case

about 5 times higher than with the other models.
The convective heat transfer coefficient in this case is defined as

$$h_c = 130V^{-0.06}(p/100000)^{0.8}T^{-0.4}(\tilde{S}_p + 1.4)^{0.8}$$

where $\tilde{S}_p$ is the mean piston speed.

Hgprofile

Heat transfer calculated from an array of heat transfer coefficients versus the crank angle. It's in fact known that the chamber has a variable volume, the charge motion intensity changes during the cycle, elastic ring move in their seat, presence of oil inside the air [4], [5].

3 Detailed GT-Power model

3.1 Engine specifications

The engine kindly made available for tests by Turku University of Applied Sciences was an in-line 4 cylinders 4.4 liters Diesel AGCO Sisu Power 3rd generation, model 44 CWA 4V. The test bench is visible in Figure 3-1.

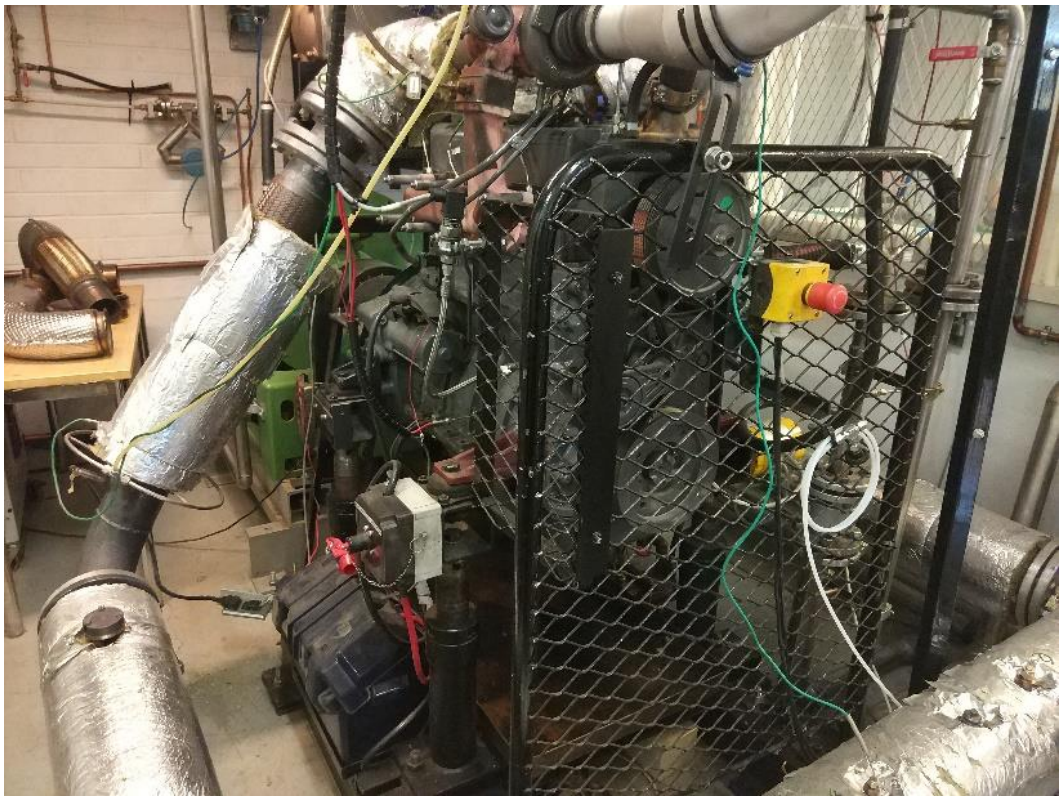

Figure 3-1: Test bench and engine.

In Table 3-1 are reported its specifications.

Number of cylinders	4
Total displacement	4.4 L
Cycle	4 strokes
Method of ignition	Compression ignition
Fuel feeding	Direct injection, common rail
Air supply	Turbocharged
Bore	108 mm
Stroke	120 mm
Turbocharger group	Single scroll, WGT controlled
Valves per cylinder	2 intake, 2 exhaust
Compression ratio	17.4:1
Piston offset (wrist pin to crank)	0.5 mm

Table 3-1: Engine characteristics.

The engine has neither internal neither external exhaust gas recirculation, no Helmoltz resonator for intake manifold tuning and has been deprived of any after-treatment system. One self-regulating flap is inserted in the intake duct right before the compressor in order to simulate intake air filters pressure drop and one flap is positioned in the exhaust pipe for simulating the ATS presence. At full load, the first flap provokes a pressure drop of 30 mbar while the second one a Δp of 150 mbar.

3.2 PC specifications

On of the purposes of this work was also to check that the Mean Value engine model was capable of running in Real-Time even on computer that are not necessarily huge workstations typical of high level laboratories. In Table 3-2 are reported the main characteristics of the laptop used for this project.

Brand and model	Dell, precision series
CPU	Intel i7-8750H
Clock	2.20 GHz (3.9 GHz w/TurboBoost)
Cores	6
Cache	9 MB
RAM	32 GB

Table 3-2: Computer characteristics.

3.3 Model building

After acquiring all the geometrical and functional characteristics of the engine, a preliminary main model has been built in GT-Suite and can be seen in Figure 3-2.

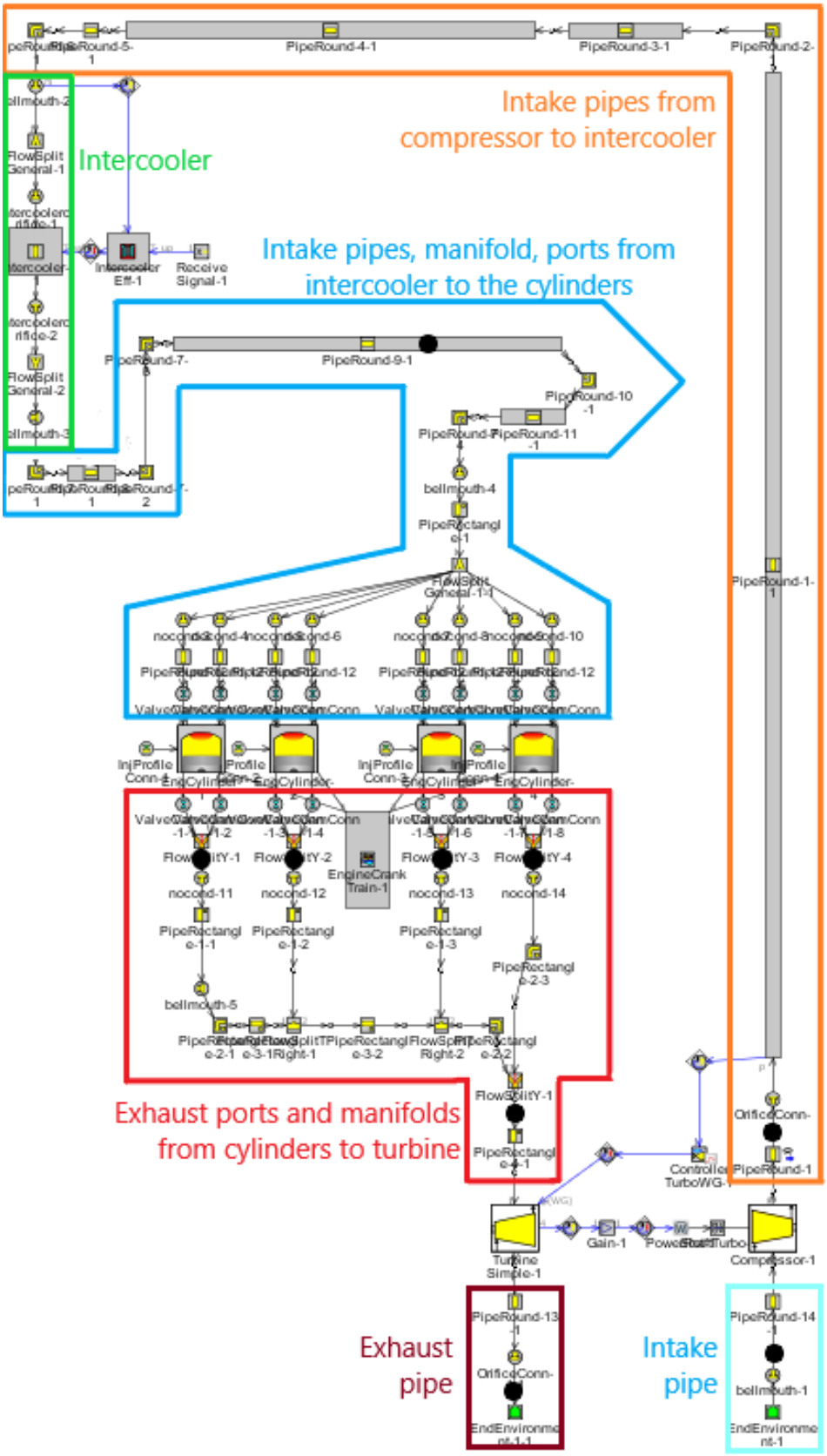

Figure 3-2: Preliminary detailed GT-Suite model with schematization. The black dots represent the position of the pressure and temperature sensors.

3.3.1 The test bench

The test bench had the following sensors:

- Cycle-averaged pressure and temperature sensors in the intake pipe, right after the compressor outlet, after the intercooler and slightly before and after the turbine (Figure 3-2)
- Torsiometer for the brake torque
- Engine speed sensor
- Intake air and exhaust gases mass flow rate sensors
- Rail pressure sensor
- Oil and cooling fluid thermocouples
- Cooling fluid mass flow rate sensor
- Fuel temperature sensor
- Coriolis based sensor for injected fuel quantity measurement

3.3.2 Experimental runs

A set of operating points (shown in Table 3-3) was acquired at the test bench

Test (case) #	1	2	3	4	5	6	7
RPM	2100	2100	1500	1500	1000	1000	848
Load	100%	50%	100%	50%	100%	50%	Idle
Torque (Nm)	416	209	520	261	538	268	2
Power (kW)	92	46	81	41	57	28	0.1

Table 3-3: Experimental data.

3.3.3 Intercooler pressure drop calibration

In order to calibrate the whole intake section, a simple model with continuous flow just for the circuit calibration was built, as visible in Figure 3-3. For the sake of a better view, the "intake908" pipe was graphically shortened.

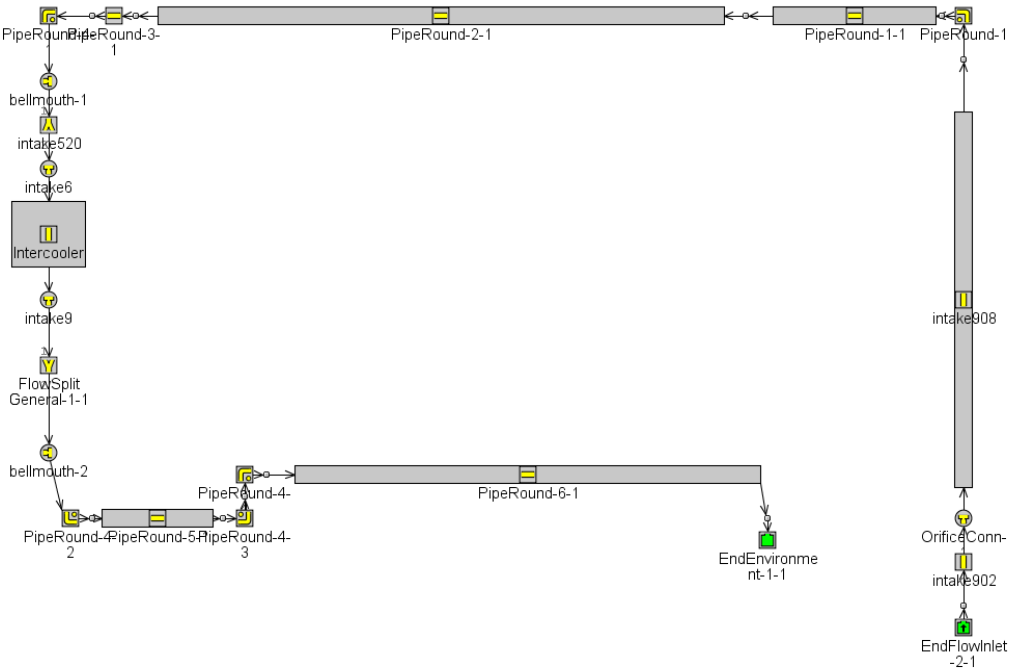

Figure 3-3:Model for intercooler pressure drop calibration.

The "EndFlowInlet" template was used to impose the air mass flow measured at the test bench, while the "EndEnvironment" one was used to impose the pressure and the temperature after the compressor. The usefulness of the various orifices is explained later. The intercooler was simulated as a bundle of 80 pipes with a diameter of 7 mm and a length of 300 mm each [6], as standard size ones, because the measure of the real ones was unfeasible. An optimization run with the integrated *advanced optimizer* tool in GT-Suite was run with the friction multiplier of each tube as *independent variable* and the pressure just after the compressor (in the "intake902" element) as target (*dependent variable*), in order to match only the biggest flow rate case pressure drop. A friction multiplier of 2.71 was found to be fulfilling the request and it was checked in all the cases, as shown in Table 3-4.

Case #	1	2	3	4	5	6
Exp. avg. pressure (bar)	2.26	2.11	2.26	1.9	1.89	1.36
Simulated avg. pressure (bar)	2.25	2.09	2.24	1.89	1.85	1.34
Error (%)	0.4	0.9	0.8	0.5	2.1	1.5

Table 3-4: Intercooler pressure drops.

3.3.4 Intercooler effectiveness calibration

Regarding the calibration of the temperature side of the intercooler, firstly a model which featured the real intake ambient and the real compressor was created, in order to have the predicted compressor-outlet temperatures (Figure 3-4).

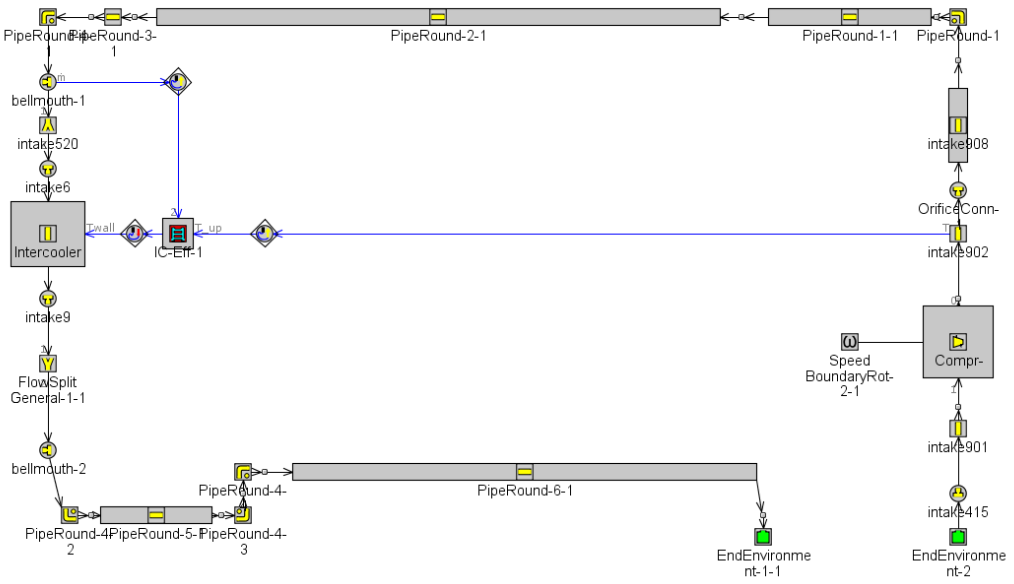

Figure 3-4: Model for intercooler heat transfer calibration.

The operating points of the compressor were then targeted, imposing the desired flow to be had as dependent variable, the shaft rpm as independent variable and the compressor outlet pressure as a constraint, since the compressor map is a 3-variable dependent map. In this way, also the turbocharger shaft rotational regimes were found, useful to set as initial conditions for the complete engine model, speeding up the convergence.

The temperatures and the rotational regimes shown in Table 3-5 were found.

Case #	1	2	3	4	5	6
Compressor RPM	148678	141252	140553	123577	120052	79101
Exp. avg. T (°C)	126	116	125	101	110	68
Simulated avg. T (°C)	129	117	127	101	115	62
ΔT (°C)	-3	+1	+2	0	+5	-6

Table 3-5: Temperatures after compressor.

Afterwards, an "IntercoolerEff" template was used for setting the temperature lowering, which is composed of a map-based intercooler with the inlet mass air flow rate and temperature as input and a temperature to impose to each of the 80 tubes constituting

the intercooler as wall temperature, computed on the basis of the map inserted in the template according to the equation of the intercooler effectiveness [6], [7]

$$\varepsilon_{IC} = \frac{T_{ai} - T_{ao}}{T_{ai} - T_c} \times 100$$

For this purpose the "bellmouth" orifices were set just to acquire the mass flow without causing any pressure drop, while the two "nocond" orifices were used to isolate the heat exchange inside the intercooler, since the *heat transfer multiplier* inside the object representing the intercooler pipes was set to 50, in order to allow the tubes to impose the desired temperature on the air flow but not on the adjacent tubes.

Although the error in the prediction of the compressor outlet temperatures was acceptable in most of the cases, the effectiveness table of the intercooler was computed taking into account the experimental temperatures for a better precision, and is shown in Table 3-6. In fact, the temperature is very important because of the influence on the propagation speed of the pressure waves [7], affecting the volumetric efficiency. The pressure waves, in fact, travel at the sound speed in air, and this speed varies with the law

$$c(T) = 331{,}45 + 0{,}62T$$

where T is expressed in Celsius degrees [8].

Air mass flow rate (Kg/s)	T_{ai}(sim.) (°C)	T_{ao}(exp.) (°C)	Effectiveness (%)
0.03	61	31	76
0.05	107	34	85
0.06	104	38	79
0.1	128	43	79
0.11	119	46	75
0.16	133	49	75

Table 3-6: IntercoolerEff map.

3.3.5 Turbocharger group

The compressor and turbo maps were not provided by the OEM. For what concerns the compressor map, it had been acquired by colleagues who formerly worked with it and has been provided only in graphical form. Its conversion to coordinates, in terms of mass flow, pressure ratio and compressor speed, was performed very fast by using the freeware PlotDigitizer. Even though the map didn't have so many curves but only 4 rotational regimes stepped by 20000 rpm, GT-Suite had an internal tool which extrapolated the map, building the surge line as well. Finally, the "Create Compressor Map External File" option in the "CompressorMap" object was selected, so that to create a file with

extension .cmp that contained the map itself, so the software didn't have to extrapolate the map in each run, saving computational time.

For what concerns the turbine instead, the map was not available at all, and so the "TurbineSimple" template was used. It consists basically in a composite of two orifices in series: the diameter of the first one determines the backpressure due to the turbine presence, while the second one is controlled by the wastegate actuator. Typical values of the first diameter ("Turbine orifice diameter") are in the order of 15÷25 mm while the second one has values in the 2÷15 mm range. It's important to remark that when a turbinesimple template is connected to a normal compressor, a more complex connection must be made, and it's shown in Figure 3-5.

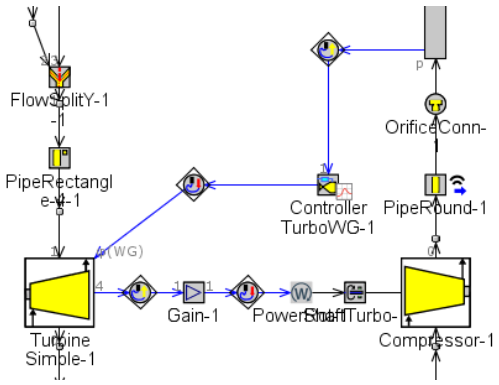

Figure 3-5: TurbineSimple connection.

The gain has to be set to 1000 and inside the WG controller it must be specified the turbine object in the "Special configurations" tab, since it's not a conventional turbine. The turbine orifice diameter was calibrated running an optimization in order to target the experimental backpressure measured in the highest mass-flow rate case.

3.3.6 Intermediate optimization of the model

After adding the turbine and the related WG controller, the model became noticeably slower because of the loop control strategy, up to not anymore sustainable levels. By looking at the simulation process user interface, it was possible to see run by run which were the elements restricting the timestep and the related percentages, as shown for example in Figure 3-6.

Variable	Unit	Flow 1 (Column 1)
Cumulative Simulation Timesteps		35789
Average timestep (time)	s	1.523E-05
Timesteps Restricted by		FlowSplitY-1-1
Percentage of Timesteps	%	98.0
Timesteps Restricted by		PipeRectangle-3-1
Percentage of Timesteps	%	1.5
Cluster Name		1 (Column 1)
Timesteps per Cycle		5252
Average timestep (angle)	deg	0.137

Figure 3-6: Model optimization: before.

It's also visible the time step, which was extremely small. The too high number of subvolumes and their size is very deleterious for the running speed.

The model has been then reviewed by correcting the elements suggested by GT-Suite, reducing too huge volumes but especially lengthening/shortening many pipes in order to have an integer number of subvolumes (so to have a total length which was a multiple of the discretization length), avoiding too small subvolumes which required the model to run very slow in order to compute all the variables for such close centroids.

The result is that the timestep in the highest speed case passed from 0.17 to about 0.57 deg, and the Real-time factor (which indicates how slower or how faster the model runs with respect to the real rotational regime) passed from 30 (which means that the model runs 30 times slower than the real engine!) to 10 (all values averaged on the six cases), as can be seen in Figure 3-7.

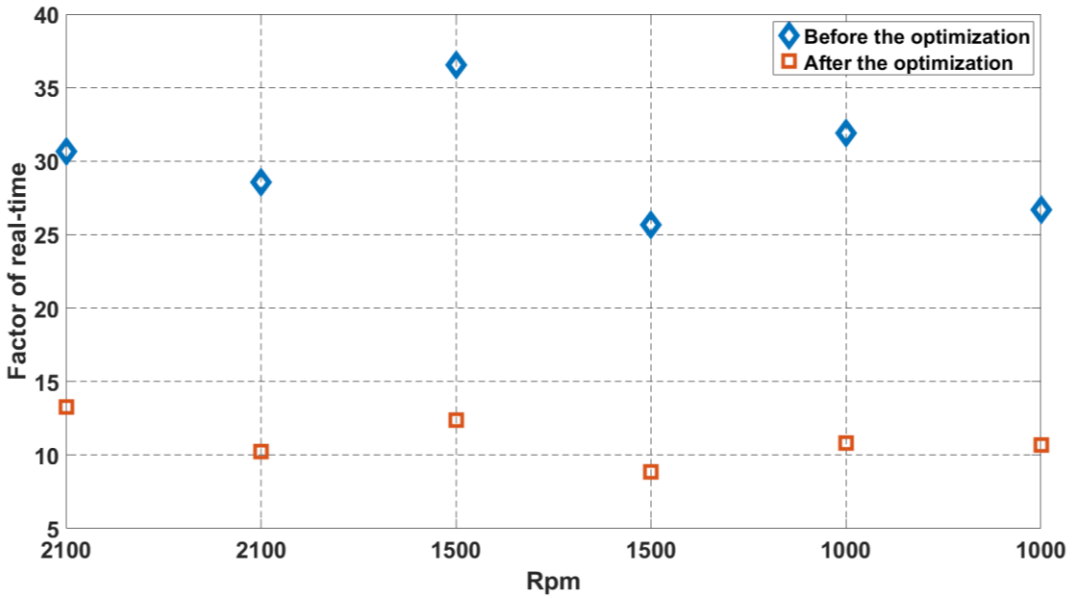

Figure 3-7: Factor of real-time before and after the intermediate optimization.

It must be reminded that for the scope of this thesis work, it had to be finally be reduced at least until 1, if not lower (because it might be that the machine on which the model will be run will not be as fast as the one gave in use by TUAS), but this was not achievable with a detailed model. In fact, one of the characteristics of the detailed models is the limit of the time step to 1 deg, set in the Run Setup folder for a better precision (constraint that was then removed during the conversion to FRM).

3.3.7 Combustion model

As briefly explained in section 2.4, the choice of the combustion model highly depends on the intended use of the complete simulation model. The predictive models were not dealt thoroughly because they were not at all the needed ones for this work, as explained below.

Engine 1D models can be used for infinite scopes: some examples are the effects of injection timing on the NOx production, or the analysis of the maximum EGR quantity over the whole operating map. These applications need a predictive combustion model, because the investigated variables highly influence the burn rate. Such models are very rich of information and can be used for several scopes. However, they need a very long experimental calibration and many measuring instruments.

When the intended use of the model instead is for example the analysis of the effects of the intake pipes length on the volumetric efficiency, or to be used in a HiL environment such in this case, the preferred combustion model is always a non-predictive one. The burn rate will be imposed and respected no matter the other variables values. A useful and detailed example can be the study of the valve timing: if the variations in charge motion are neglected, what changes in the working of the engine is only the pumping work loop, because the combustion remains exactly identical. In this way, the increase of the bmep (and so the decrease of the bsfc) due to the decrease of the pmep can be evaluated for each lift profile.

Still, it can be that the engine under investigation is not equipped with the necessary sensors to conduct such analysis, such as in this case, and they could not be implemented because the engine was used by different researchers groups at the same time, thus the engine couldn't be disassembled for long periods. Moreover, this was the same reason for which the volumetric efficiency of the engine was not calibrated in the various combinations with and without the turbine and the compressor, as is usually suggested [6], [9].

For the reason just explained, the used combustion model for this work was finally the "EngCylCombDIWiebe", used in its semipredictive version, so setting all its parameters

as default. This was possible because injection profiles, even if not totally accurate, were available.
A huge problem of the usage of this model is that it doesn't allow multi-injection profiles, as is the case of this engine.

As shown in hundreds of studies, pilot injections are very important for the reduction of the noise of the engine and for the ignition delay [4], [7], [10], [11]. The absence of accuracy on the noise emission was not a concern in this work obviously, but the ignition delay reduction directly affects the engine efficiency. Thus, one could have thought to set the main injection timing between the original pilot and main injection timings, in order to have the same bsfc, basically. However, post injections are mainly known for oxidizing the remained soot at the end of the combustion process, but as side effect they also increase the temperature in the exhaust, changing the temperature at the turbine inlet and so the power that can be given by the turbine [4] , [7], [10], [11]. Thus, following this logic, the main injection timing should instead have been set after the original injection timing.

The final decision was to run the engine with the only main injection and with its default timing copied from the map of the test bench, which may have carried some inaccuracies but still the general model was providing a very real torque curve, which was one of the objectives of the work.

3.3.8 Injector model

It's known that Diesel combustion is very sensitive to the injection profile, and especially in this case in which the combustion model was set with *def* parameters, that then the software was going to calculate exactly through the injection profiles: they had to be quite precise.

A very long process of experimental investigation or a GT-Suite model of the injector were then the best ways to produce such data. Nevertheless, neither the instruments to conduct the experimental investigation, neither the injector internal geometry for building an injector model were available. All what was available were the EMI curves (which give the cumulative injected mass as a function of the energizing time and of the rail pressure) of the same injector but with a different nozzle [12] and some injection profiles acquired experimentally at 400, 800 and 1200 bar for different energizing times [13].

Having this data, the injection profiles for different injected quantities and rail pressures were found scaling the experimental ones [14].

After digitizing the EMI curves as done with the compressor map, they were scaled along the y axis to pass from the nozzle present in the journal article to the nozzle of the case under investigation. Considering various energizing times and rail pressures, this factor resulted to be an average of 0.714 with a quite good precision all over the EMI curves.

The EMI curves extension was only up to 1200 bar, while the ECU operates the engine under investigation up to 1570 bar. Since the injected volume at constant energizing time doesn't vary linearly with the rail pressure, the ratios of the injected volume at a certain rail pressure over the injected volume at a 200 bar lower rail pressure were computed for the experimental available curves, and extrapolating the curve of these ratios over 1200 bar, the EMI curves of 1400 and 1600 bars were found.

Then, after having imported them in MATLAB, the experimental points acquired in the 7 experimental cases conducted with the AGCO engine were added (rail pressure, energizing time and injected quantity were available), since the EMI curves had injection times lower than 1 ms while the experimental cases recorded injection times up to 1.4 ms, and the curves were interpolated.

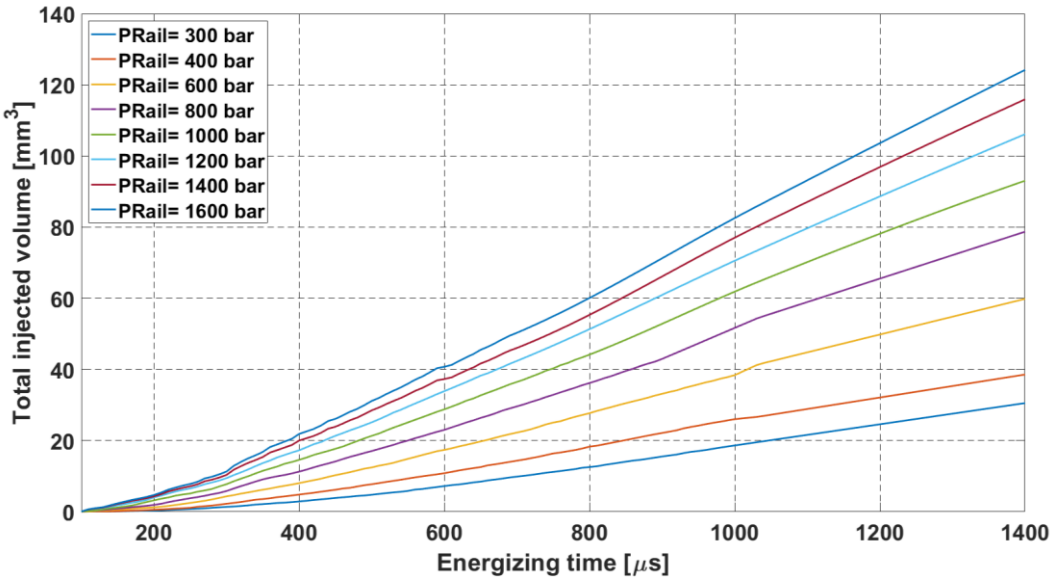

Figure 3-8: EMI curves of the injector under investigation.

To interpolate/extrapolate the injection profiles at different rail pressures and different energizing times, the following assumptions were made [14]:

- The hydraulic delay is constant at a given rail pressure
- The rising slope of the mass flow rate is constant at a given rail pressure
- Once the injector is at full lift (maximum mass flow rate), the mass flow rate is constant (the controlling part is the nozzle)

- The descendent slope of the mass flow rate is constant at a given rail pressure

Figure 3-9 shows the origin of these assumptions.

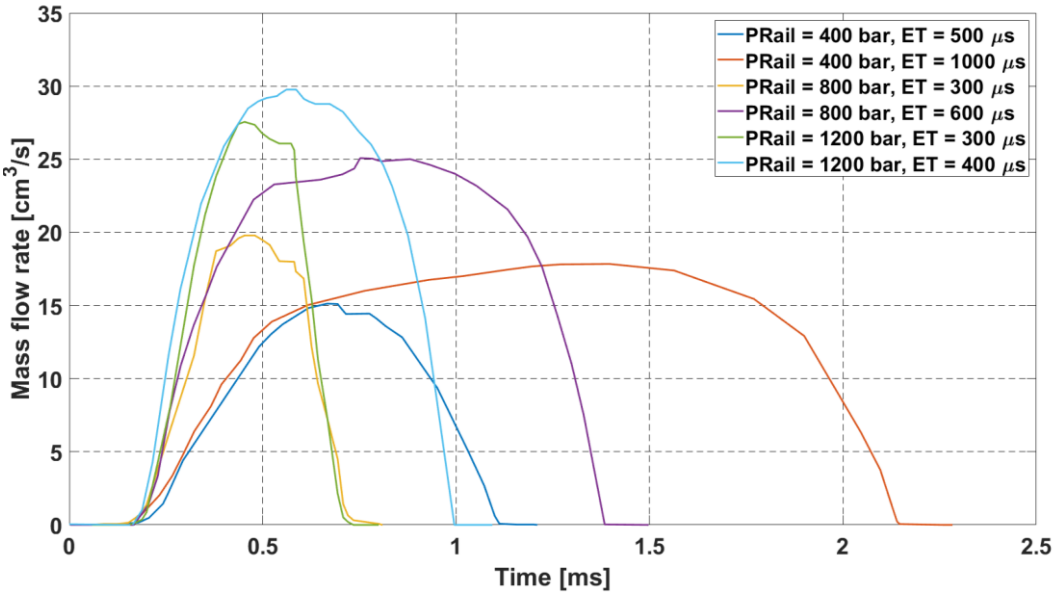

Figure 3-9: Experimental injection profiles.

A program computing all the needed injection profiles having as input the injected mass and the rail pressure was then made using MATLAB. The program was capable of recognizing if the input rail pressure was closer to 400, 800 or 1200 bar (where the experimental injection profiles were available) and then scale the closest injection profile plus lengthen the constant phase time in order to have the input injected mass. For example, if the injection pressure was 700 bar, the software was first going to scale the experimental injection profile of 800 bar and then to lengthen/shorten the constant

phase. In Figure 3-10Figure 3-10: Injection rate profiles generated with MATLAB. it's possible to see all the injector profiles from 350 to 1400 bar, from 10 to 100 mg.

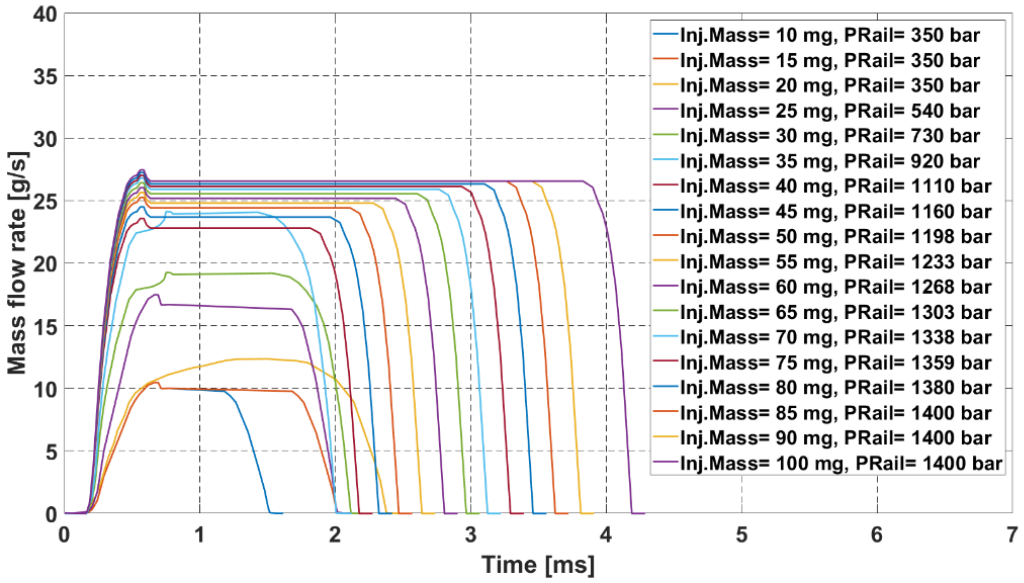

Figure 3-10: Injection rate profiles generated with MATLAB.

The injection profiles for all the operating conditions of the engine were then generated and inserted in a three coordinates map (input: injected mass, rpm; output: injection profile) inside the InjProfileConn object.

The MATLAB script can be found in the Appendix.

3.3.9 Heat transfer model

Regarding the heat transfer model, due to the lack of complex data, the simple combination of EngCylTWall and WoschniGT were used, imposing 600, 600 and 400 K respectively as temperatures for head, piston and cylinder, as suggested by GT-Suite for the full load case.

The accuracy of this model was checked only for the full load, 1000 rpm operating point because it was the only point with no post injection. In fact, depending on its timing, the exhaust temperature can vary of about 100 °C, so it has a very high influence [15].

To check whether the simulated temperature was correct, a thermocouple object was added in the thermal folder in one of the runners of the exhaust manifold. The simulated temperatures are indeed very different from the measured ones because of the influence of the instrument itself: the flow impacting on the thermocouple transforms in fact part of its kinetic energy in heat raising the recorded temperature, while the heat transfer

from the inner part of the thermocouple to the adjacent parts which are in contact with the exhaust manifold and with the surrounding air, raise or lower the recorded temperature according to the operating condition (transient from low to high load or vice versa). The parameters for the thermocouple object were the ones of the most common thermocouples, considering that even if the thermocouple was different, the difference in the measured temperature would have been anyway lower than the difference between the experimental temperature and the temperature simulated without any thermocouple. The simulated temperature was then found to be lower than the experimental one only of 2%, then the heat transfer model was assumed correct and no further calibration was performed. The thermocouple was finally removed from the model to have shorter computational times.

Also, talking about heat transfer, it must be noticed that in part load operation of the model there could be inaccuriacies due to the wall temperature object, which in absence of experimental data has been set with values for head, piston and liner (as previously said) typical of a full-load operation. Part-load modelling would instead require an EngCyllTWallSoln object.

3.3.10 Results of the detailed model

In the following graphs can be seen some results of the model and the comparison with the measured data. It must be reminded that many values might show big differences because the injection strategy was completely different and, furthermore, the turbine model is far from being representative of the actual one. The trends of pressure and temperature, in fact, go diverging because of the flow calibration done for the highest mass flow rate case (turbine orifice diameter). However, the tenor is very similar and so will be the torque and power curves.

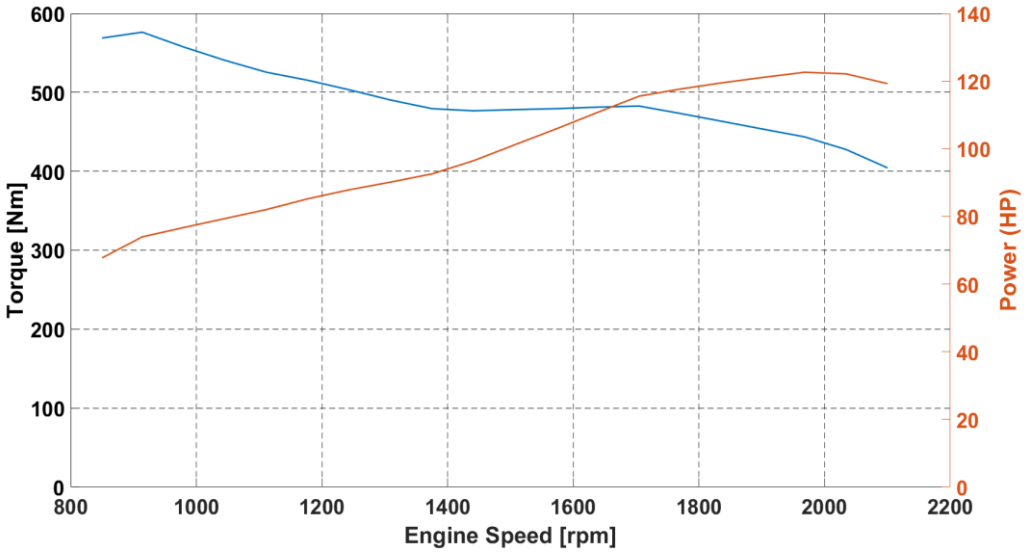

Figure 3-11: Full load torque and power curves.

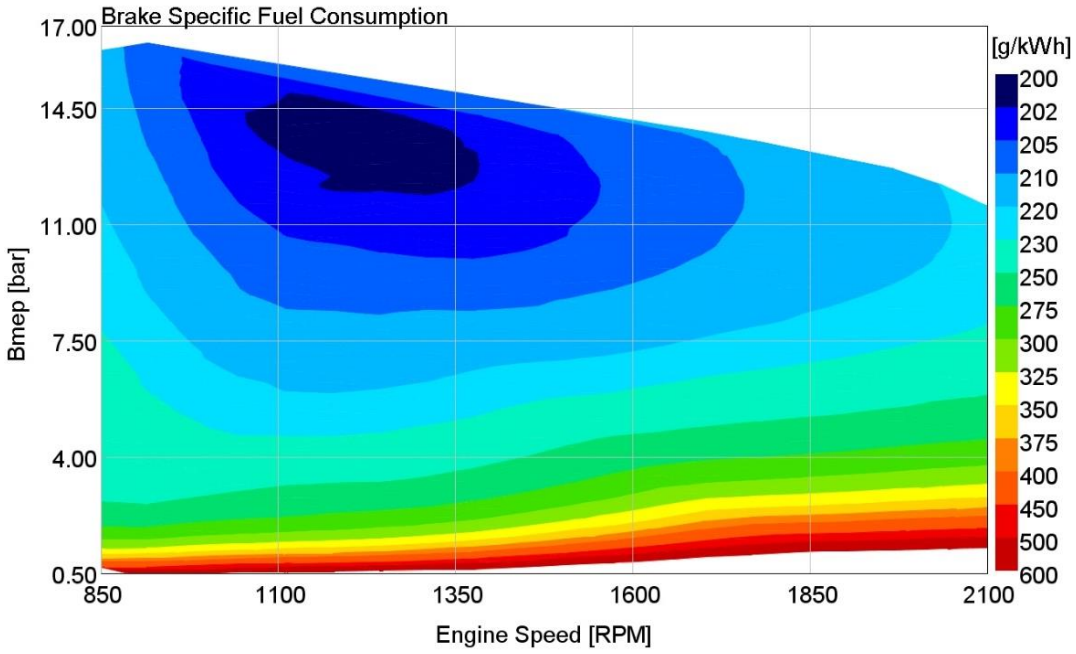

Figure 3-12: Detailed model engine map.

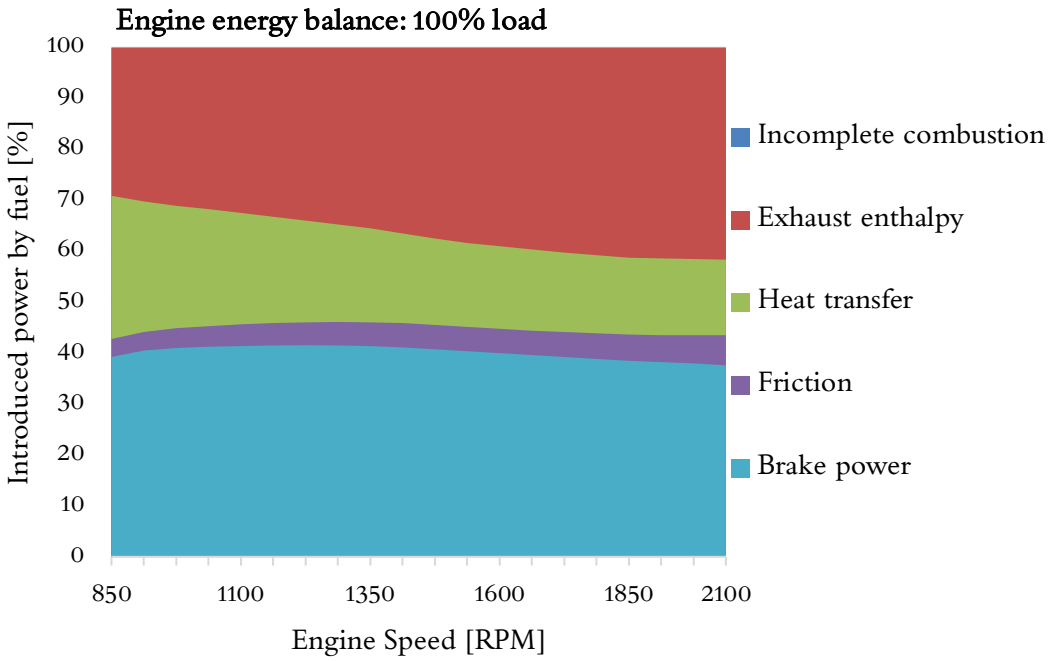

Figure 3-13: Engine energy balance at full load.

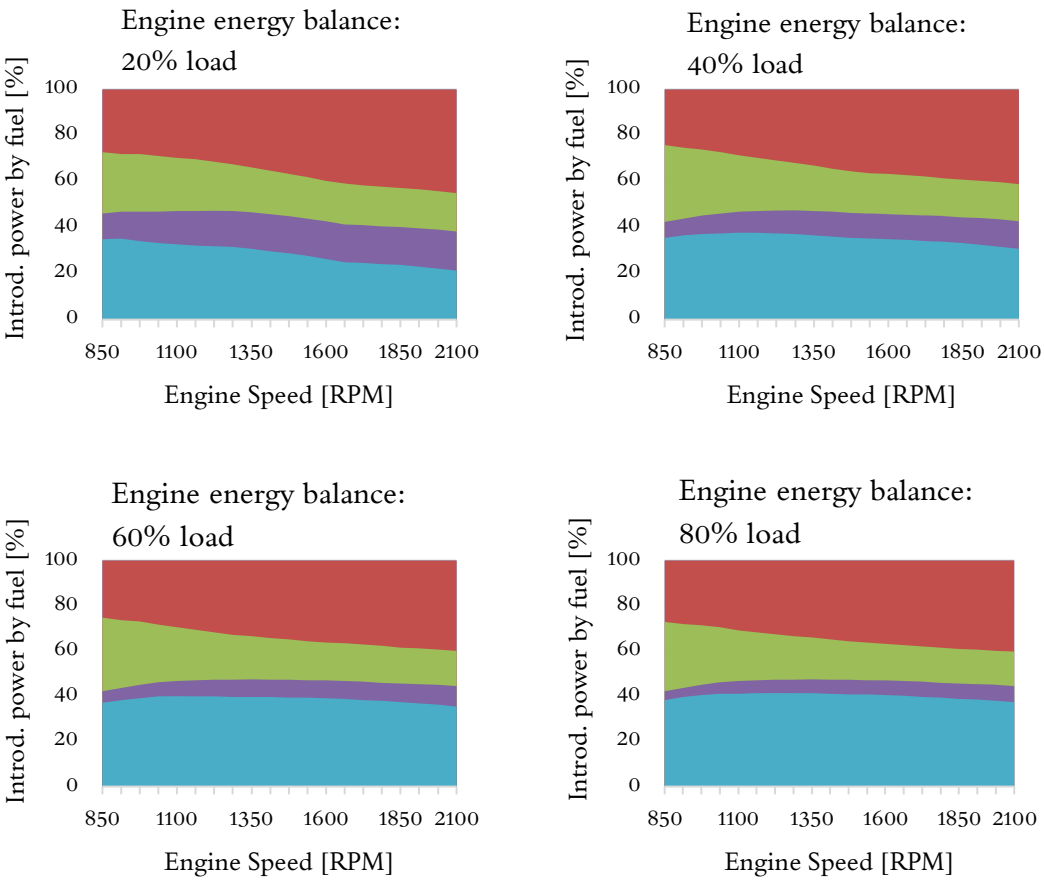

Figure 3-14: Engine energy balance at part load.

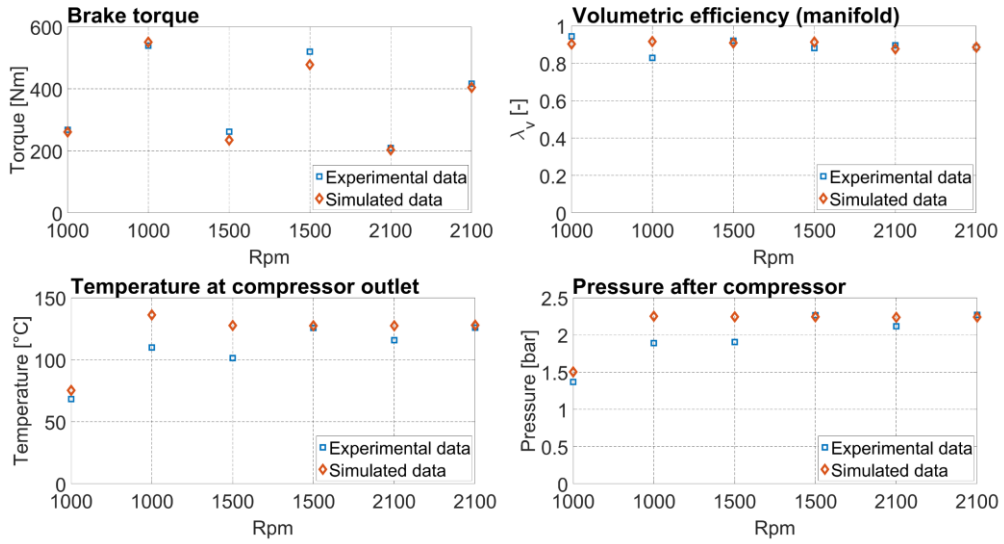

Figure 3-15: Experimental - simulated data comparison.

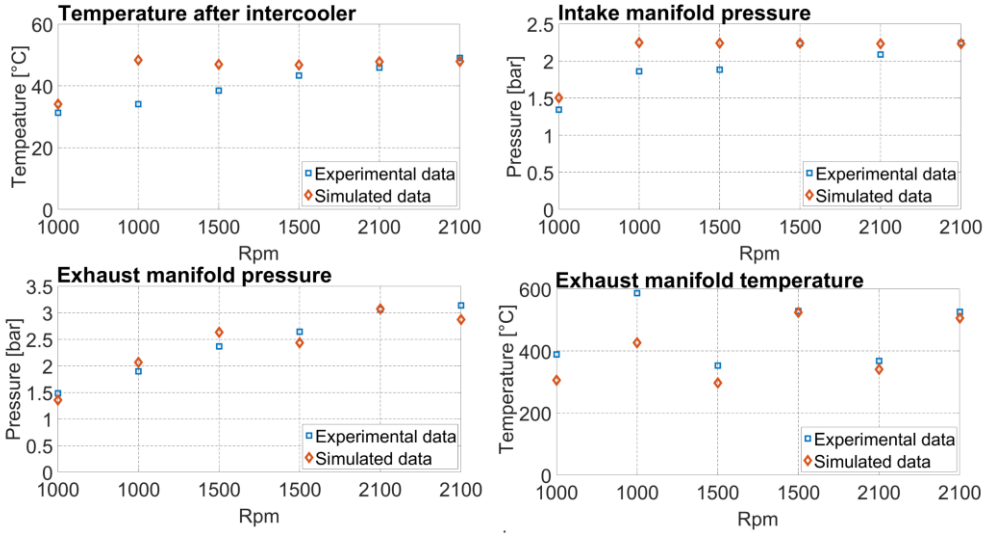

Figure 3-16: Experimental - simulated data comparison (2).

4 Mean Value (real-time) Model

4.1 Introduction to Fast Running Models (FRM) and Mean Value Models

The running time of standard models is generally significantly high, resulting in models that can't perform calculations for thousands of cycles per minute, that is the rotational speed that common engines have. For some applications such as HiL environments for testing the ECUs, as in the case of this thesis, or for fast transients modelling, the running time is of key importance. Thus, some accuracy can be sacrificed in favor of a faster speed.

Although *Fast Running Models* and *Mean Value Models* are often believed to be the same thing, there's a huge difference between them.

Fast Running Models are still very similar to the detailed engine models, they just have the combination of all the pipes in a section of the engine (e.g. compressor outlet pipes) into a single pipe which has the same characteristics, and so pressure drop and heat transfer. FRMs in fact keep the EngCylinder template, so the model is still predictive and, wave dynamics, that are necessary for example for EGR studies and turbo pulse efficiency, are still present. The lengthening of subvolumes allows contemporarily larger time step sizes and less calculations per time step. Running time can be in this way reduced by a factor of 10 or more but the model can be not running in Real-Time (that is to say, if the engine speed we are running the model is 3500 RPM and it's a 4-stroke

engine, the model must be able to perform 1750 cycles per minute) [9].

Mean Value Models instead are further simplifications of the FRMs: in addition to all the procedures made to simplify the model used for the FRMs, the cylinder is substituted with a mean value cylinder. Substantially it's just a big map (with data stored previously) that receives as input some parameters for the conditions of the engine in that cycle (imep, volumetric efficiency etc.) and gives as output the torque, power and other variables of interest. No calculations inside this new cylinder are made at all, so the model is not predictive anymore. Still, since there's not a cylinder with a heat transfer model anymore, thermal managements models can't be done with mean value models. In fact, it's enough to imagine that no matter how many cylinders the initial model had, there will be only one cylinder in the mean value model [9].

	Fast Running Model	Mean Value Model
Advantages	- Heat rejection from the cylinder computable - Wave dynamics - Combustion modeling	- Very stable solution - Very fast run time
Disadvantages	- Some accuracy may be lost - Slower than Mean Value Models	- Longer process to create the model (DoE) - Limited predictive capabilities - No heat rejection from the cylinder
Usages	- Thermal Management Models - Integrated System Models - Control Models (HiL)	- Control models (HiL) - Vehicle Models only requiring performance input from the engine

4.2 Procedure followed for the conversion to Fast Running Model

The model was converted first into a FRM and then into a Mean Value one because got to a certain point of the conversion to FRM, the software warned that the factor limiting the time step was not anymore the flow circuit but the combustion. In particular, it suggested to set a finite value to a parameter in the combustion model, which was set as *def* because of the semi-predictive nature of the model, that calculates the parameters of the combustion from the injection profiles, taking long time. Received the warning

from the software, it was converted into a MV model and after that the simplifying of the flow circuit continued. Hereafter is explained the process followed to simplify the flow circuit (FRM conversion), while later is shown the conversion to MV model.

In spite of the fact that the procedure described is standard and to be followed, the degree of simplification varies from case to case. For example, if the heat rejection to the coolant is of primary concern, it´d be better to leave the ports as they are, since they are in contact with the cooling fluid. Moreover, if a finite element based heat transfer model inside the cylinder was used (EngCylTWallSoln, for example) in order to have very accurate heat rejection profiles, this can be still used in the FRM.

The conversion starts usually with the identification of the most time-step restricting component, which commonly is the exhaust system because of the higher temperatures, and its simplification to a single volume. This is because it depends on the factor of Real-Time that is targeted, it can be that a general decrease of 1 or 2 units in this factor is enough for the intended use of the model but, since this work is aiming at carrying it to the absolute value of 1 (at least), the simplification of the only exhaust ducts would have not been enough and starting from the conversion of the exhaust manifold rather than from the intake one wouldn't have made any difference.

The volumes can be combined in two different ways: with the tool present in the FRM converter wizard or with the Combine Flow Volume Wizard.

FRM converter wizard tool

The conversion starts with the tagging of the components in groups, so that they can be simplified with an easier procedure. From the Tools menu, FRM Tags window can be opened and the elements of interest selected and grouped. GT-Suite has already inside this window a schematic subdivision as aid, as can be seen in Figure 4-1.

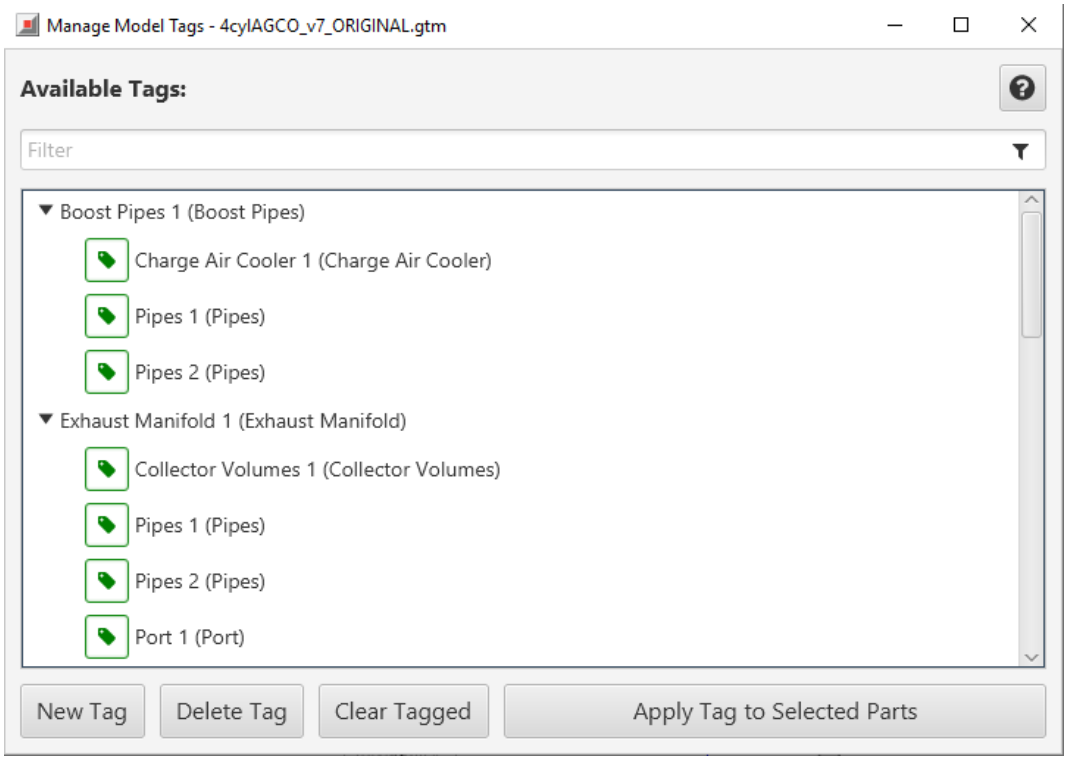

Figure 4-1: Tags predefined in GT-Suite.

For example, inside the Boost Pipes, that go from the compressor to the intake manifold, it's clear that there will always be an intercooler and some pipes. By clicking twice on "Pipes", the tags "Pipes 1" and "Pipes 2" can be created, because the whole fluid circuit will be simplified in three total components: the pipes from the compressor to the intercooler, then the intercooler itself, and finally the pipes that carry from the intercooler exit to the intake manifold inlet.

It's important to remark that during the tagging of the components, every group has to start with a pipe/flow-split and end with an orifice, that is the connection to the next group, as shown in Figure 4-2.

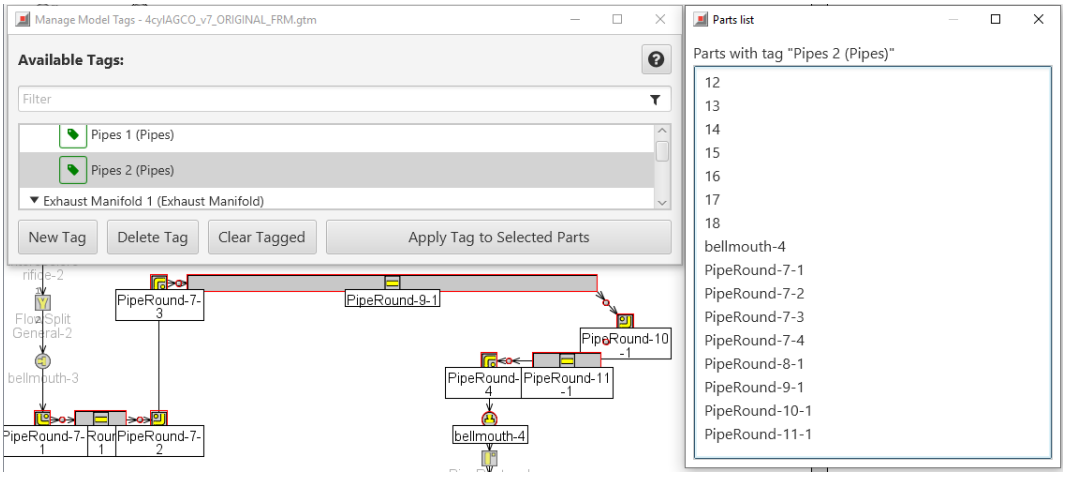

Figure 4-2: Components tagging.

The procedure then continues by being guided: the tool knows exactly of which part of the circuit the simplification was going to be done (whether intake pipes, rather than boost or exhaust ones), and provides appropriate hints for the conversion. For example, for what concerns the exhaust manifold, a heat transfer multiplier is usually set in the port because they are in the hot head of the engine and together with the heat transfer from the exhaust valve, the heating phenomenon is significant. If there is a turbine, the temperature of the gases that will reach the its inlet is very important for the enthalpy gap, so the temperature of the inlet gas must be matched. Since the ports were going to be combined with the runners, the software suggested to find a heat transfer coefficient that along the total length of the new pipe would have led to the same temperature that was had in the detailed model, at the turbine inlet. The wizard also provides in the beginning the option if to simplify for having a compromise which focuses a bit more on the accuracy or one that focuses on speed.

Combine Flow Volume Wizard

With this tool there's no need to tag the components, they must be selected case by case. It requires more expertise as it must be known which parameters to calibrate, which components to lump together.

In conclusion, both methods were used.
Whether was used one method or the other for combining the volumes, the model had then to be run with all the cases in which the calibration was wanted, and all the points except from the idle were analyzed in this case. The FRM converter, in fact, needs a .gdx file in order to start the process.

Once launched the FRM converter, some RLT variable had to be chosen because the software was going to compare the result from the detailed model and the one from the

simplified one and check that the difference was within the target percentage. For the intended use of this model (providing an accurate full load torque curve, bsfc calculation and engine energy balance), the variables shown in Table 4-1 were chosen.

Variable	Allowed error (%)
Brake torque	3
Exhaust temperature in the exhaust manifold	3
Brake specific fuel consumption	3
Volumetric efficiency	3

Table 4-1: Key RLTs for accuracy check.

As briefly mentioned, the simplification and calibration of each subsystem consisted basically in adding an orifice with a diameter to calibrate in order to have the same pressure drop, or by specifying a friction multiplier different from 1. Additionally, in parts of the circuits with high heat addition/rejection, a heat transfer multiplier bigger or smaller than one was added too. These calibrations were performed for the highest flow rate case and then the diameters/multipliers were set for all the cases in order to check whether the results were in the accuracy limits (allowed error) set in the beginning. As the workflow in Figure 4-3**Errore. L'origine riferimento non è stata trovata.** shows, only then it was possible to start the simplification of the other systems, and it must be remarked that the simplifications had to be cumulative. In other words, the simplification could not be done singularly and then the accuracy checked with respect to the detailed model, but had to be done in a sequence: in fact, the error coming from all the simplifications had to be lower than not 3%, not the error deriving from each subsystem simplification.

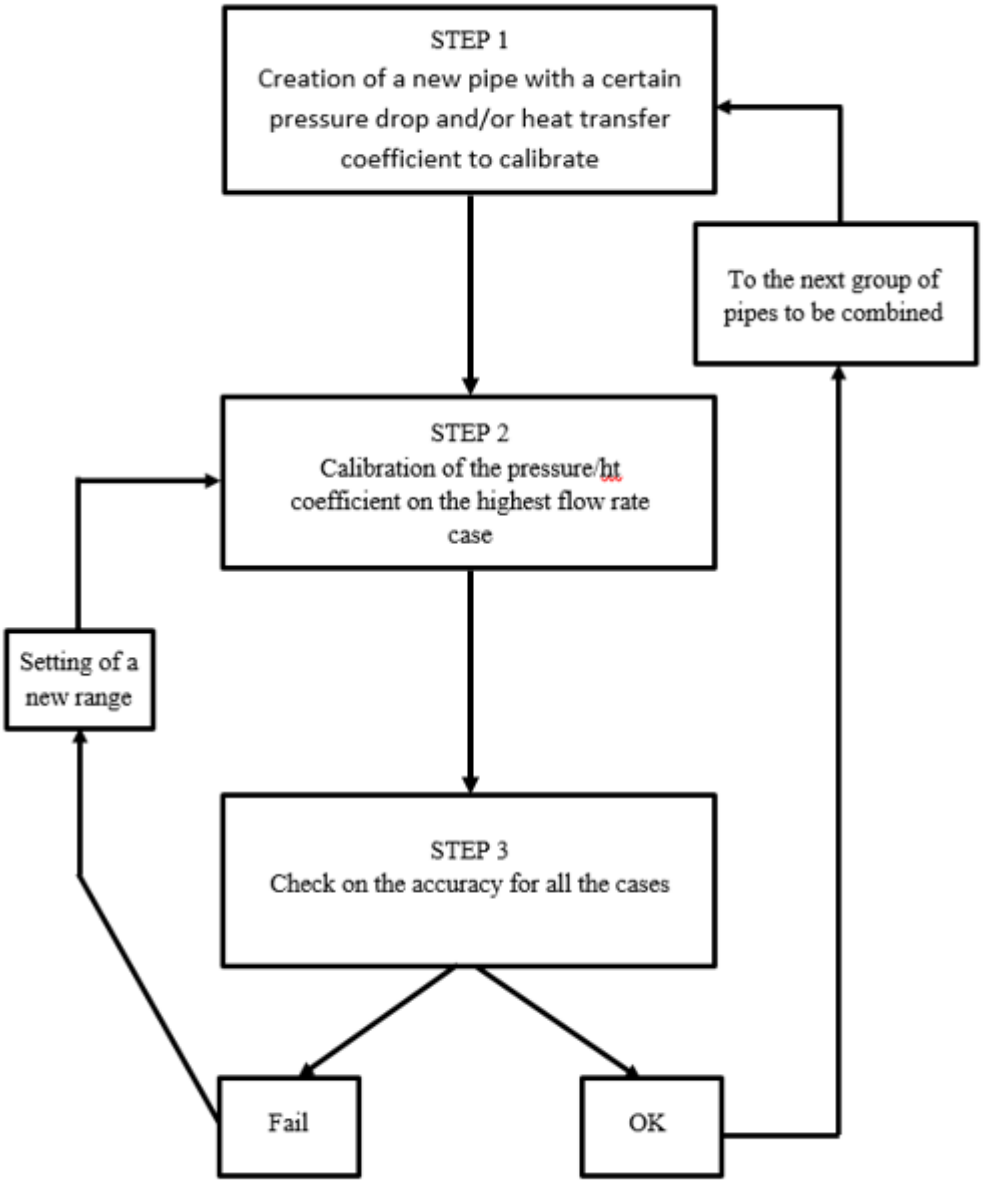

Figure 4-3: Workflow for the conversion to FRM.

In Figure 4-4 can be seen the result of the conversion of the boost pipes, while in Figure 4-5 and Figure 4-6 is visible the accuracy check done after the cumulative conversion of intake pipe and of boost pipes (step 2): the error was smaller than 3%, as set.

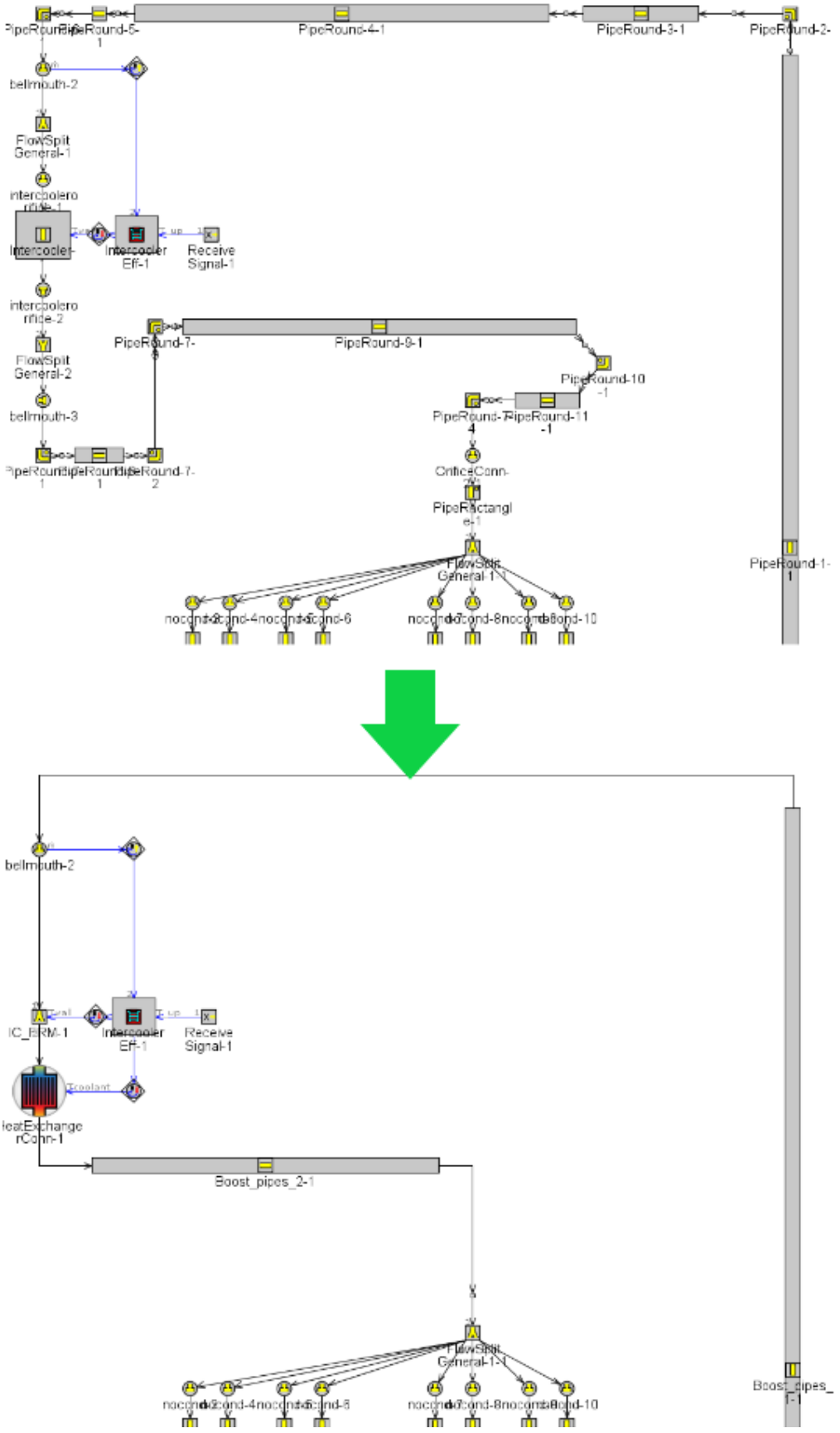
PipeRound-4-1
PipeRound-3-1
bellmouth-2
FlowSplit General-1
Intercooler Eff-1
Receive Signal-1
PipeRound-9-1
PipeRound-10 -1
PipeRound-11 -1
OrificeConn-
PipeRectangl
FlowSplit General-1-1
PipeRound-1-1
bellmouth-2
Intercooler Eff-1
Receive Signal-1
Coolant
Boost_pipes_2-1
FlowSplit General-1-1
Boost_pipes_ 1-1

Figure 4-4: Result of the conversion of the boost pipes.

Case Setup Number	rpm+ [RPM]	Baseline Value [N-m]	Previous Value [N-m]	Previous Error [%]	Current Value [N-m]	Current Error [%]
1	2100.0	404.90747	404.02676	-0.21750815	409.88754	1.2299284
2	2100.0	229.65685	229.0433	-0.2671554	234.73006	2.20904
3	1500.0	467.93625	467.5752	-0.07715869	471.80426	0.8266108
4	1500.0	236.85493	236.65593	-0.08401983	238.09225	0.5223957
5	1000.0	565.7431	565.6738	-0.01224494	569.30566	0.62971354
6	1000.0	230.58972	230.5655	-0.010501638	231.23248	0.2787467

Figure 4-5: Accuracy check of step 2, spreadsheet.

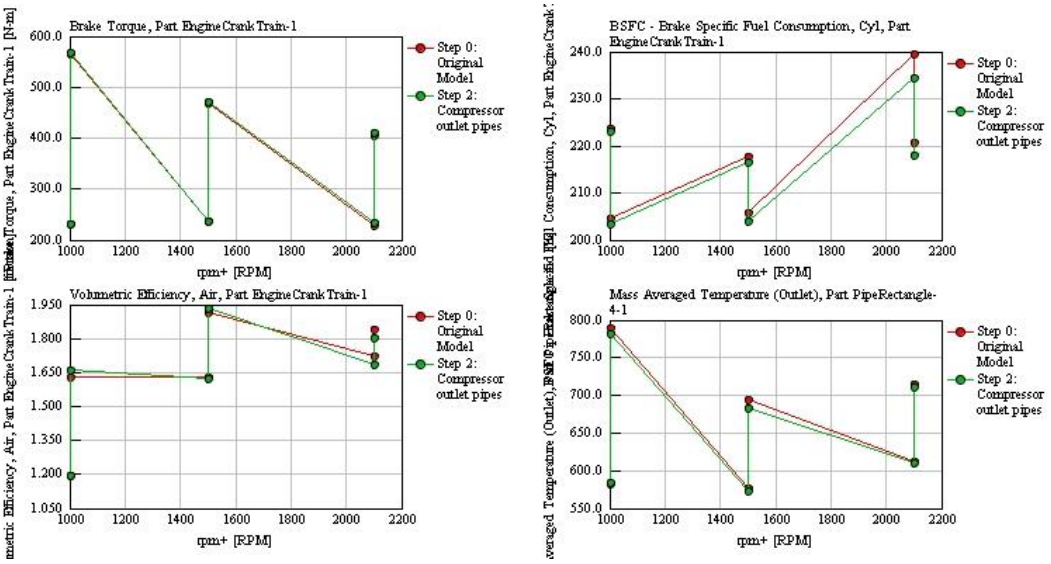

Figure 4-6: Graphical accuracy check of step 2.

Additionally, the constraint of maximum 1 deg as time step had to be relaxed and set to 720 in the Time Step and Solution Control Object (Run Setup) in order to reach the Real-Time running speed.

4.3 Conversion to Mean Value model

As said, after a certain point the model needed to be converted to a Mean Value one because of the slowness of the combustion computation.

4.3.1 DOE setup and Neural Networks training

Starting from the detailed model, the turbocharger group was removed and two endenvironments were added: one to simulate the boost pressure and one to simulate the backpressure. The following data were then parametrized (inserted in square brackets in order to be used in the DoE):

1. Boost pressure
2. Boost temperature
3. Back pressure

4. Injected mass
5. Start of injection
6. Engine speed

This was done because all the possible combinations of these parameters had to be computed and stored. The removal of the turbocharger brought the additional advantage of having a much faster model, very important since the number of cases to be computed is in the order of thousands, depending on the number of variables involved. The DoE setup is visible in Figure 4-7.

Factor	Unit	Min	Max
rpm	RPM	848.0	2100.0
PBOOST	bar	1.0	2.3
PBACK	bar	1.0	3.13
T_BOOST	K	273.0	398.0
injected_mass	mg	0.0	100.0
SOI	deg	-9.5	5.0

Figure 4-7: DoE setup.

The number of experiments had to be high enough to provide good sets of data for building the maps that will control the mean value cylinder. A study conducted on a model similar to this showed that the root mean square error of imep and exhaust temperature for a 6 variable investigation is acceptable with 4000 cases and anyway not decreasing much anymore while increasing the number of experiments, as visible in Figure 4-8 [6]. For the volumetric efficiency the problem is less relevant (the reason why imep, exhaust temperature, volumetric efficiency and fmep are the variable of main concern is explained just below).

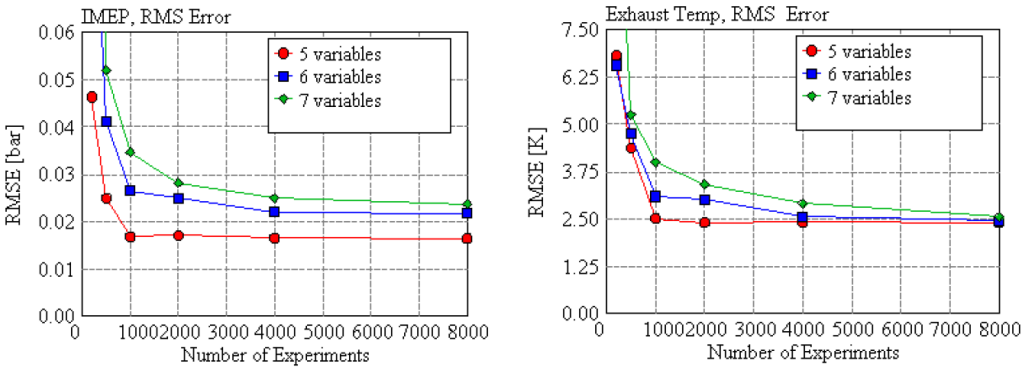

Figure 4-8: RMSE vs. number of experiments.

4.3.2 Neural networks training

After calculating all these cases, the *neural networks* had to be trained. Substantially they are a subsystem that receives as input the six variables listed above, depending on the case that is investigated, and give as output three inputs for the mean value cylinder (calculated and stored in the DoE run):

- Volumetric efficiency, it determines the mass flow rate through the cylinder
- Indicated efficiency (that can also be the IMEP), it will impose how much of the fuel energy contained in the cylinder will be converted in work done on the piston
- Exhaust energy fraction (that can also be the exhaust temperature), it imposes how much of the energy of the fuel is used to heat up the exhaust gases
- fmep, since the engine friction object in this case included a dependence on cylinder pressure (through the Peak Cylinder Pressure Factor), a neural network had to be trained to take into account this relation. In fact, mean value cylinders do not predict combustion and gas exchange events, so the maximum pressure inside it is much lower than the real one.

Neural networks are needed because all these 4 variables can't be put in a look-up table, as they are not function of only two parameters. Figure 4-9 shows how the inputs for the neural networks are chosen, while in Figure 4-10 constraints to the maximum input intake and exhaust pressures were set in order to filter the DoE data saving space and possible errors. Similar constraints were set also in the OutputRLT and ConstraintRLT folders.

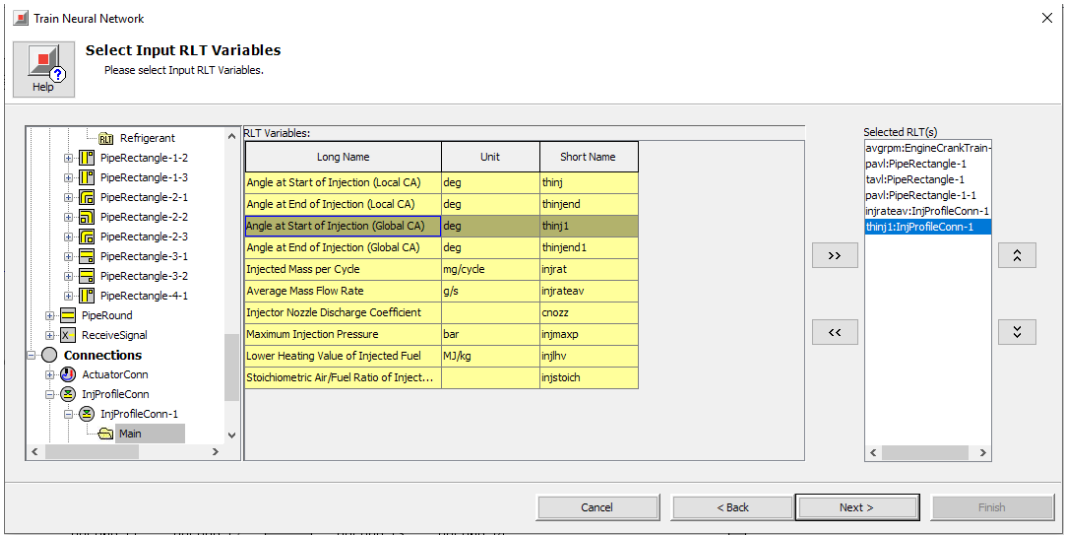

Figure 4-9: Neural networks input.

InputRLT | OutputRLT | ConstraintRLT

RLT Variable	Min	Max
avgrpm:EngineCrankTr...	ign	ign
pavl:PipeRound-7-2	ign	2.3
tavl:PipeRound-7-2	ign	ign
pavl:Pipe_exh_manifold	ign	3.2
injrateav:InjProfileConn-1	ign	ign
thinj:InjProfileConn-1	ign	ign

Figure 4-10: Neural networks filtering.

After the training, the accuracy of the training had to be checked. In Figure 4-11 it's possible to see that there was a very good fitting of the neural network output data with respect to the original DoE data.

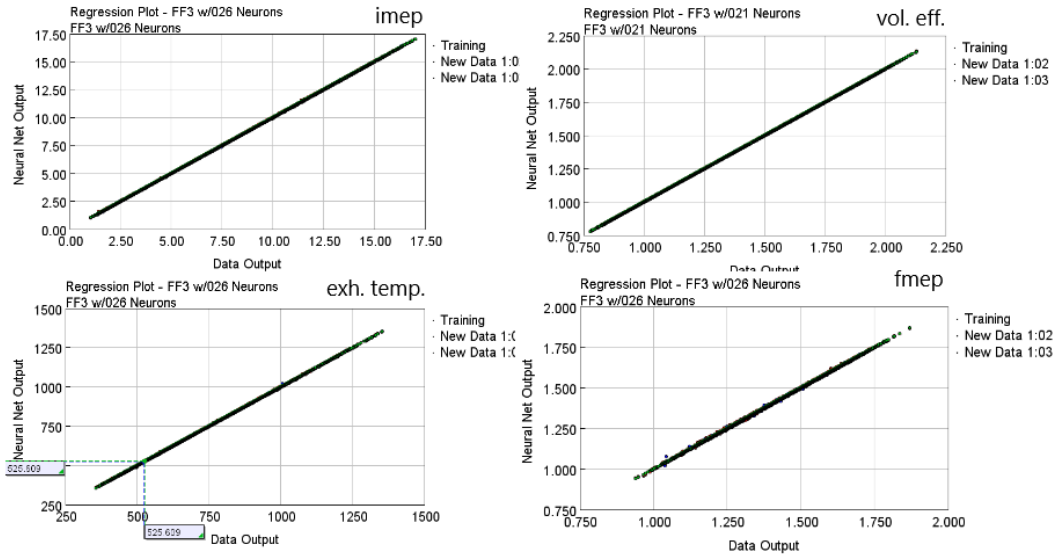

Figure 4-11: Neural networks fitting check.

4.3.3 Neural networks configuration

After the simplification of the last parts of the flow circuit (hindered prior to the conversion to MV model), the four cylinders with the relative valves were substituted by a unique Mean Value cylinder, ready to be controlled exclusively by the neural networks. The process of adding this control system to the MV cylinder was accomplished by adding as many sensors as input variables (6, in this case) and as many actuators as the input for the cylinder (4). Between them, "NeuralNet" objects were inserted. This template receives all the 6 inputs of the case under investigation and through the data stored from the training of the neural network gives as output the imep, fmep, volumetric efficiency and exhaust temperature that were had with the same 6 inputs in the detailed model. All this system was grouped in a subassembly in order to have a better view of the model that otherwise would have resulted too confusional. In Figure

4-12 can be seen the neural network subassembly while in Figure 4-13 it's possible to see its connections to the mean value cylinder.

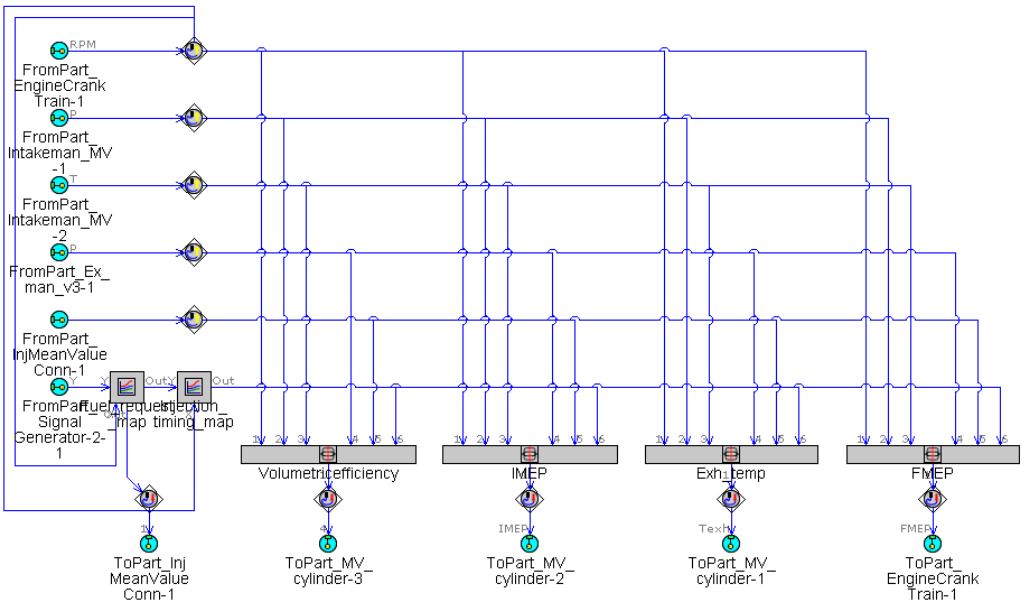

Figure 4-12: Neural networks subassembly.

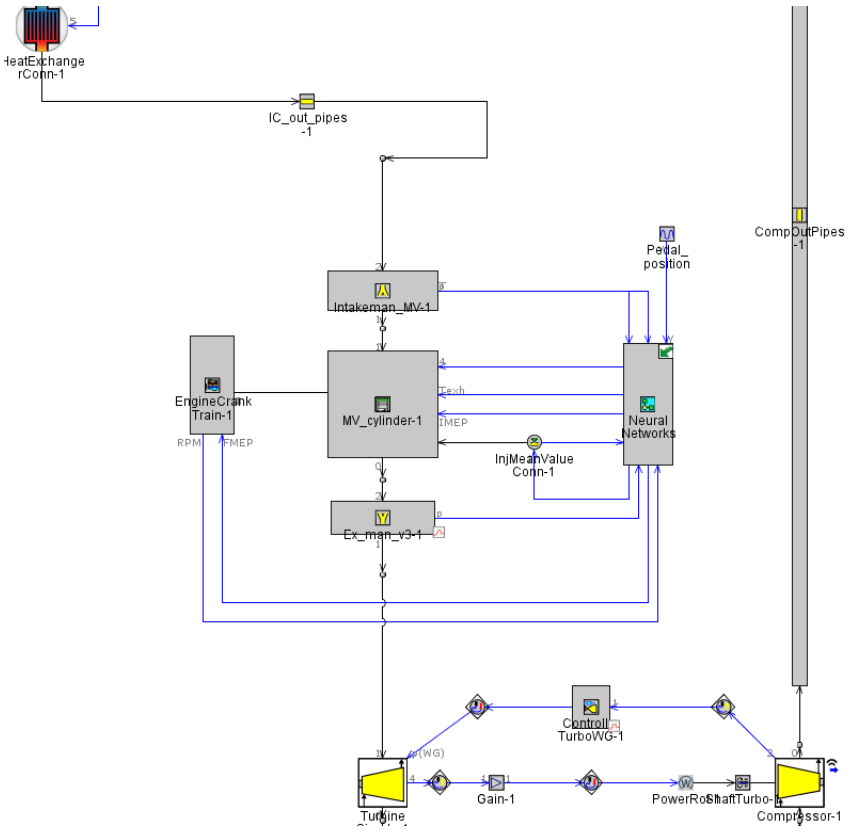

Figure 4-13: Control system of the Mean Value model.

4.3.4 Additional changes

After the whole conversion to mean value model, there were still some cases of the whole map that had a factor of Real-Time bigger than one. To solve this issue, the *real gas option* in the time step and solution control object was set to *off*. In fact, in the detailed model, since peak pressures (especially in direct injection engines) are high enough that the variation from a perfect gas to a real one becomes significant, it was necessary to have it on. Nevertheless, in mean value models, as already said dealing with the fmep, the in-cylinder pressure is totally different and this extra detail was not needed anymore.

The mean value model was finally used to compute the injection quantities map for the whole operating range of the engine: starting from the full load curve injection quantities map taken from the laboratory ECU, the injection quantities for part load operations were obtained using the standard optimizer targeting percentages of the maximum torque from 10 to 90 %, for each rotational regime. For what concerns the idle condition (0% load), the optimization was carried targeting bmep=0, and so engine providing only the necessary work to overcome frictions and pumping work to keep the engine rotating but producing no useful power at the shaft (imep=fmep). Knowing that optimization processes take very long times, this would have been practically infeasible with the detailed model running with a factor of Real-Time equal to 10 ca. The fuel request map having as inputs the rpm and the load percentage was then created and added to the detailed model as well, having the engine now controlled by the *pedal position* object (signal generator template). A model with an active controller for the pedal position signal and some monitors was finally realized in order to control the engine behavior actively and in real-time, as shown in Figure 4-14.

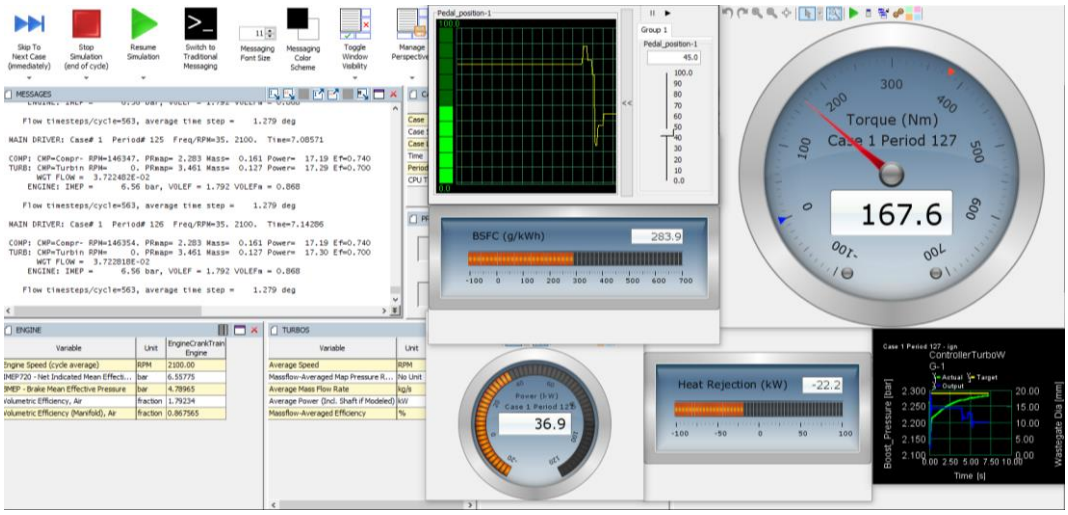

Figure 4-14: Model with active accelerator and torque, power, bsfc and heat rejection monitors.

4.4 Transients

The model now obtained is capable of running transients. However, there are some inaccuracies:

- Inertias: inertias of engine and turbocharger were not available, thus a value of $0.25\ kg*m^2$ for the former and of $0.005\ kg*m^2$ for the latter were assumed looking at engine models with similar geometries in the GT-Suite examples libraries, but still there are obviously some inaccuracies

- Temperatures: temperature variations are instantaneous because the thermal solver is set to steady state and not to transient, so the thermal capacitance of each object is not taken into account. However, this could be calibrated through the detailed engine model but still, it was not the purpose of the project

- Heat transfer object: as already said, the set temperatures for engine head, piston and liner are not realistic for the whole operating range

Despite the fact that these inaccuracies are present, the model can be used for transients with many other aims.

To set transient simulations, "ProfileTransient" objects can be set both into the cranktrain object, Engine Speed section (specifying how the engine speed must vary during the time), and/or in the pedal position object (indicating how the load varies during the transient to be simulated).

4.4.1 Transients initialization

This kind of simulations requires special attention to the initial conditions, differently from the steady-state ones. For steady-state simulations, a reasonable value close enough to the actual result is used in order to assure convergence in a not too long time. For transient simulations instead, initial conditions values must be precise. The suggested procedure is, therefore, to set an initial case with the same conditions of the starting point of the transient so that during this first case the values for the initial conditions are found and then the transient can start. Obviously, the initialization state (Run Setup folder) must be set to "previous_case". To make the model faster, the thermal solver can be left to steady for the first case and only then set to transient (if the thermal behavior is of interest).

4.5 Accuracy check and results of the Mean Value model

The Mean Value model was run in the whole engine operating map, and in Figure 4-15 is shown that finally a factor of Real-Time smaller than one for each operating point was

achieved, while Figure 4-16 shows the comparison of the Factor of Real-Time for the evolution steps of the model. The average factor of real time was strongly reduced: the MV model can run now about 32 times faster then the optimized detailed one.

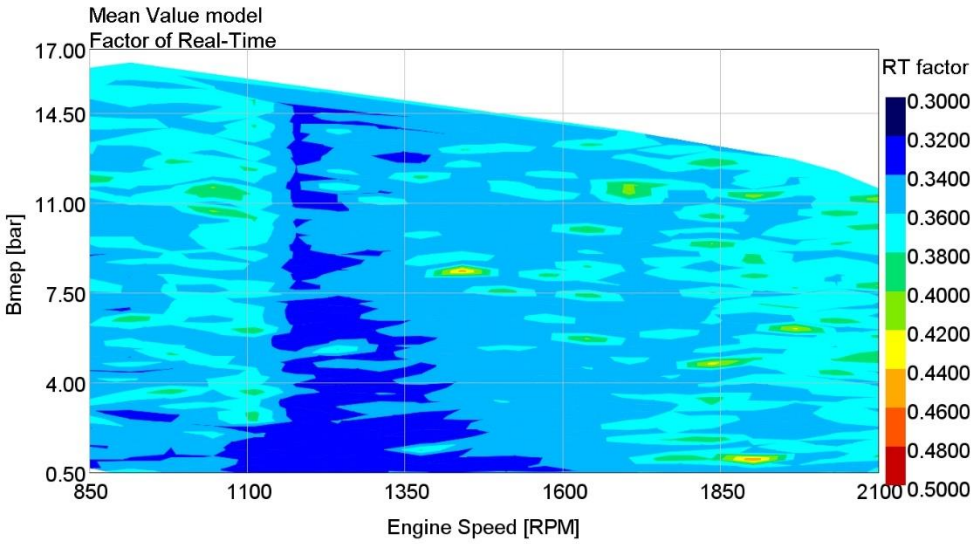

Figure 4-15: Mean Value model: factor of Real-Time in the engine map.

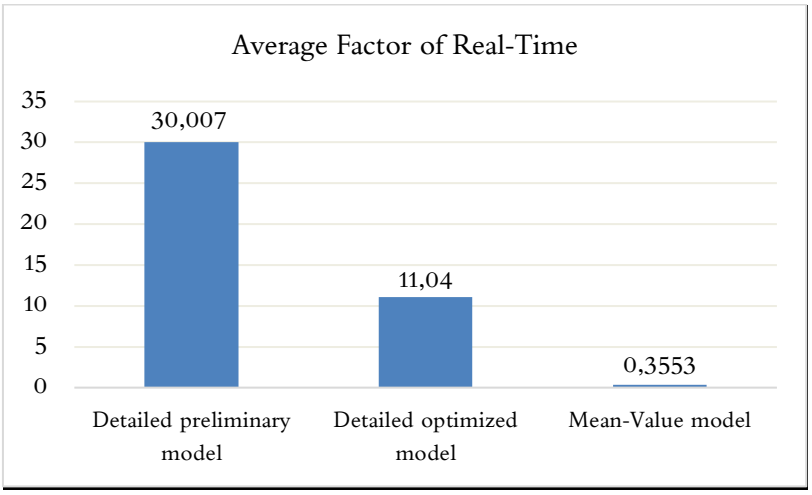

Figure 4-16: Factor of Real-Time for the three evolution steps of the model averaged in the six experimental cases.

The accuracy of the MV model with respect to the detailed one was checked, it's important to remark that it was done in operating points different from the ones for which the neural networks were trained. As the following figures show, there were some zones of the engine map with accuracy problems, reaching the 10% of difference. The biggest problems come from the pressure in intake and exhaust and from the consequent volumetric efficiency. The root cause was the calibration of the intake and exhaust lumped volumes, which pressure drop was done for the highest mass flow rate (in fact, at 2100 rpm, full load, the error is 0%). Nevertheless, this calibration method was the best one [6] and the model showed a very good behavior, having the error lower than the 2% for most of the map in all the variables.

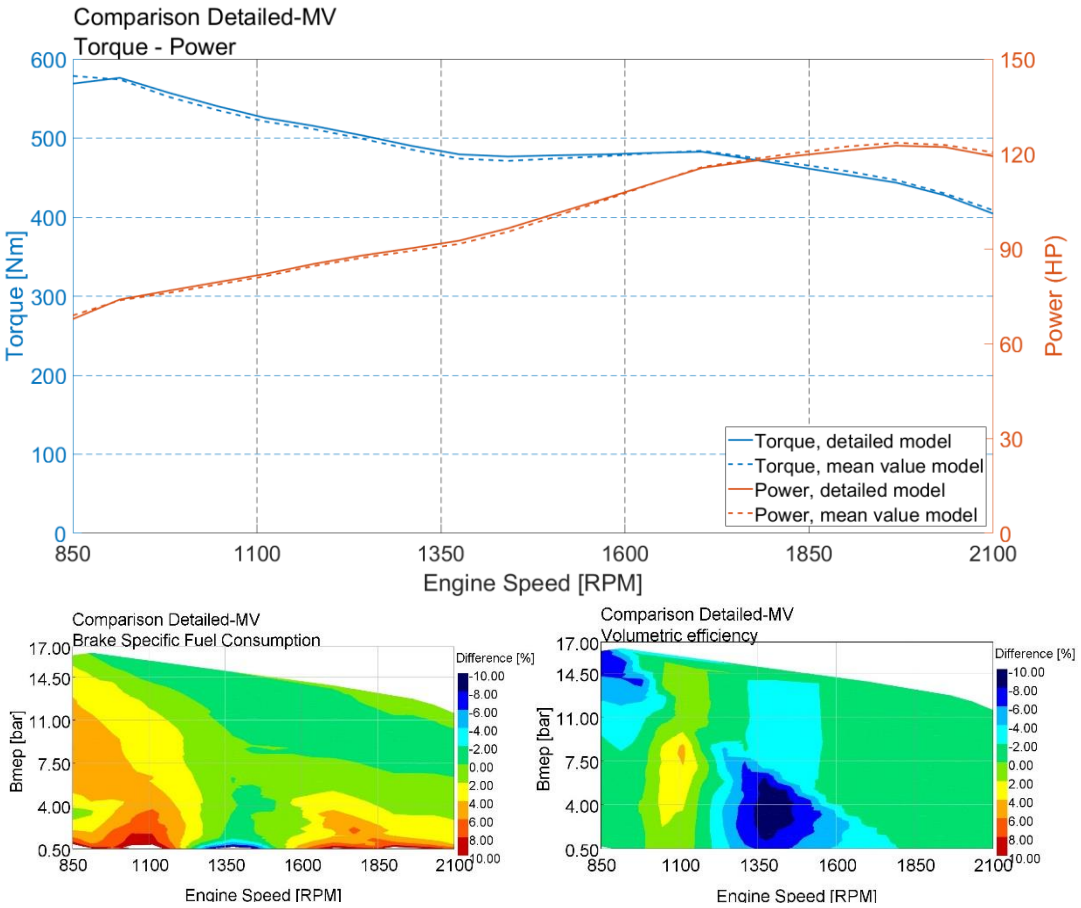

Figure 4-17: Performances comparison of detailed and MV models.

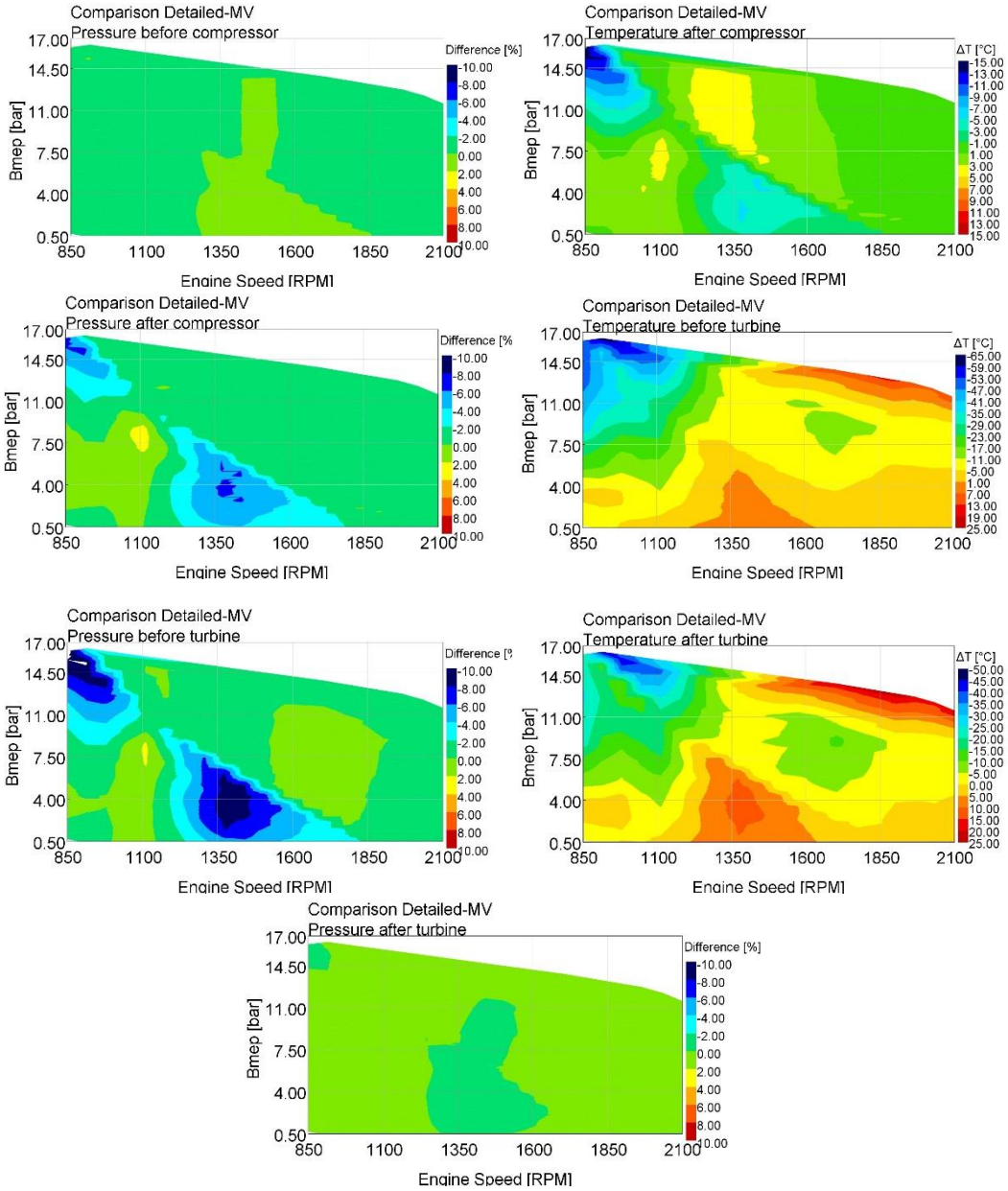

Figure 4-18: Temperature and pressure comparison of detailed and MV models.

Conclusions

This master thesis demonstrated primarily that it's possible to run a high fidelity model in Real-Time with a standard computer, having found a very low Factor of Real-Time and a very good accuracy compared to the detailed model in the whole operating range, with pedal position stepped by 2% and engine rpm divided in 20 intervals from 850 to 2100 rpm. Secondarily, it demonstrated that quite reliable detailed engine models can be built even without very complex measuring instruments. Starting in fact from rough data such as the geometries of intake and exhaust pipes and making similarity assumptions for what concerns the turbine and the fuel injection system, the engine GT-Suite model was built and calibrated step by step following the path that an air particle crosses from intake bellmouth to the tail-pipe. The most challenging process was calibrating the combustion model: its semi-predictive nature forced the use of a single-injection strategy (instead of the multi-injection used in the real engine) and, after having tried to simulate the injector with a dedicated 1D model (with no success, since the internal geometries of the injector were not available and are very critical for such a model), the injection profiles were obtained by generating a MATLAB code which interpolated and extrapolated experimental curves over the whole operating range of pressure and injection quantities of the injection map stored in the laboratory ECU. After having validated the heat transfer model through a thermocouple object inserted in one of the exhaust runners showing a temperature very close to the experimental one, the conversion to Real-Time started. Beginning from a detailed model with an average Factor of Real-Time equal to 9.7, the model was first converted to a Fast Running Model (which lumps the flow circuit volumes in order to have less variables to calculate and a faster convergence) and then to a Mean Value Model, which further transformed the whole cylinders subsystem into a map. The model was in this way speeded up of a factor of 32 circa, since the average final Factor of Real-Time was 0.35. Finally, inertias were added to make it capable of running transient simulations as well and a model with live controllers and displayers was created to show effectively its capabilities. Further developments of this thesis work are the real implementation of such a model in a HiL test bench, that will be done by a colleague from Turku University of Applied Sciences in the near future.

Appendix

MATLAB code realized for the generation of the injection profiles.

```
clear all
close all
clc
%% EMI curves: data acquisition
EMI_data=xlsread('EMI_curves_excel','EMI_curves_8holes');
injectiontiming300bar=EMI_data(1:63,1);
injectedflow300bar=EMI_data(1:63,2);
injectiontiming400bar=EMI_data(1:71,3);
injectedflow400bar=EMI_data(1:71,4);
injectiontiming600bar=EMI_data(1:78,5);
injectedflow600bar=EMI_data(1:78,6);
injectiontiming800bar=EMI_data(1:75,7);
injectedflow800bar=EMI_data(1:75,8);
injectiontiming1000bar=EMI_data(1:70,9);
injectedflow1000bar=EMI_data(1:70,10);
injectiontiming1200bar=EMI_data(1:67,11);
injectedflow1200bar=EMI_data(1:67,12);
injectiontiming1400bar=EMI_data(1:67,13);
injectedflow1400bar=EMI_data(1:67,14);
injectiontiming1600bar=EMI_data(1:67,15);
injectedflow1600bar=EMI_data(1:67,16);

%% EMI curves: data interpolation in order to have a matrix with
no NaN values and to plot it in an easier way
linsp_injtiming=linspace(100,1400,131);
interpinjtiming300bar=interp1(injectiontiming300bar,injectedflow30
0bar,linsp_injtiming,'linear','extrap');
interpinjtiming400bar=interp1(injectiontiming400bar,injectedflow40
0bar,linsp_injtiming,'linear','extrap');
interpinjtiming600bar=interp1(injectiontiming600bar,injectedflow60
0bar,linsp_injtiming,'linear','extrap');
interpinjtiming800bar=interp1(injectiontiming800bar,injectedflow80
0bar,linsp_injtiming,'linear','extrap');
interpinjtiming1000bar=interp1(injectiontiming1000bar,injectedflow
1000bar,linsp_injtiming,'cubic','extrap');
interpinjtiming1200bar=interp1(injectiontiming1200bar,injectedflow
1200bar,linsp_injtiming,'cubic','extrap');
interpinjtiming1400bar=interp1(injectiontiming1400bar,injectedflow
1400bar,linsp_injtiming,'cubic','extrap');
interpinjtiming1600bar=interp1(injectiontiming1600bar,injectedflow
1600bar,linsp_injtiming,'cubic','extrap');

injectedflow_matr(1,:)=interpinjtiming300bar;
injectedflow_matr(2,:)=interpinjtiming400bar;
injectedflow_matr(3,:)=interpinjtiming600bar;
injectedflow_matr(4,:)=interpinjtiming800bar;
injectedflow_matr(5,:)=interpinjtiming1000bar;
injectedflow_matr(6,:)=interpinjtiming1200bar;
injectedflow_matr(7,:)=interpinjtiming1400bar;
injectedflow_matr(8,:)=interpinjtiming1600bar;

figure(1)
plot(linsp_injtiming,injectedflow_matr)
xlim([100 1400])
xlabel('Energizing time [micros]');
```

```
ylabel('Total injected volume [mm^3]');
grid on
legend({'PRail= 300 bar','PRail= 400 bar','PRail= 600 bar','PRail=
800 bar','PRail= 1000 bar','PRail= 1200 bar',...
    'PRail= 1400 bar','PRail= 1600 bar'},'Location','Northwest');

%% Experimental profiles: data acquisition
injexpprofiles=xlsread('EMI_curves_excel','nozzle8_profiles');
profile_time(1,:)=injexpprofiles(:,1)/1000; %400bar, 300 micros
profile_flowrate(1,:)=injexpprofiles(:,2);
profile_time(2,:)=injexpprofiles(1:30,3)/1000; %400bar, 1000
micros
profile_flowrate(2,:)=injexpprofiles(1:30,4);
profile_time(3,:)=injexpprofiles(1:30,5)/1000; %800 bar, 300
micros
profile_flowrate(3,:)=injexpprofiles(1:30,6);
profile_time(4,:)=injexpprofiles(1:30,7)/1000; %800 bar, 600
micros
profile_flowrate(4,:)=injexpprofiles(1:30,8);
profile_time(5,:)=injexpprofiles(1:30,9)/1000; %1200 bar, 300
micros
profile_flowrate(5,:)=injexpprofiles(1:30,10);
profile_time(6,:)=injexpprofiles(1:30,11)/1000; %1200 bar, 400
micros
profile_flowrate(6,:)=injexpprofiles(1:30,12);

%% Experimental profiles: subdivision in ascending, constant,
descending phase
for i=1:6
profile_ascphase(i,:)=cumtrapz(profile_time(i,1:18),profile_flowra
te(i,1:18))*1000; %injcted fuel during ascending phase
profile_constphase(i,:)=cumtrapz(profile_time(i,18:19),profile_flo
wrate(i,18:19))*1000;
profile_descphase(i,:)=cumtrapz(profile_time(i,19:end),profile_flo
wrate(i,19:end))*1000;
end

%% Case
PRail=[350  350 350 540 730 919.9   1109.9  1159.6  1198.1  1233.2
1268.4  1303.6  1338.7  1359.2  1379.6  1400    1400    1400];
injectedmass=[10    15  20  25  30  35  40  45  50  55  60  65  70
75  80  85  90  100];
injectedvolume=injectedmass/0.85; %mm3
experimentalpressures=[400,-100,800,-100,1200,-100];
k=1; %index for writing the matrix (to be fast copied in an excel
file)
l=2;
for j=1:length(PRail)
    clear minDiff
clear index1
clear index
clear index4
clear multiplier
clear newvector_time
clear newvector_flowrate

index4=find(linsp_injtiming==800); %for finding the multiplier
based on the same injection time
if (PRail(j)>=300) && (400>=PRail(j))
```

```
multiplier=1/((injectedflow_matr(2,index4)/injectedflow_matr(1,ind
ex4))+(1-
(injectedflow_matr(2,index4)/injectedflow_matr(1,index4)))/100*(PR
ail(j)-300));
elseif (PRail(j)>400) && (600>=PRail(j))

multiplier=1+((injectedflow_matr(3,index4)/injectedflow_matr(2,ind
ex4))-1)/200*(PRail(j)-400);
elseif (PRail(j)>600) && (800>=PRail(j))

multiplier=1/((injectedflow_matr(4,index4)/injectedflow_matr(3,ind
ex4))+(1-
(injectedflow_matr(4,index4)/injectedflow_matr(3,index4)))/200*(PR
ail(j)-600));
elseif (PRail(j)>800) && (1000>=PRail(j))

multiplier=1+((injectedflow_matr(5,index4)/injectedflow_matr(4,ind
ex4))-1)/200*(PRail(j)-800)
elseif (PRail(j)>1000) && (1200>=PRail(j))

multiplier=1/((injectedflow_matr(6,index4)/injectedflow_matr(5,ind
ex4))+(1-
(injectedflow_matr(6,index4)/injectedflow_matr(5,index4)))/200*(PR
ail(j)-1000))
else
multiplier=1+((injectedflow_matr(8,index4)/injectedflow_matr(6,ind
ex4))-1)/400*(PRail(j)-1200)
end

 % Finding the closest experimental curve to use in order to
approximate the new profile
for i=1:6
    diff(i)=abs(PRail(j)-experimentalpressures(i));
end
minDiff=min(diff);
index1=find(diff==minDiff);%in order to find which profile to
multiply, to have a more realistic case
index2=index1(1); %in case, for example, 600 bar would give 2
indices
for i=1:6
    diff_injvol(i)=150;
end
for p=index2:index2+1

diff_injvol(p)=abs(profile_ascphase(p,end)+profile_constphase(p,en
d)+profile_descphase(p,end)-injectedvolume(j));
end
minDiff_injvol=min(diff_injvol);
index3=find(diff_injvol==minDiff_injvol);
index=index3(1);
index5=index3-1;

injvolconstphase=injectedvolume(j)-
multiplier*profile_ascphase(index,end)-
multiplier*profile_descphase(index,end); %volume to be injected
during the constant phase
if injvolconstphase<0
   index=index5;
```

```
   injvolconstphase=injectedvolume(j)-
multiplier*profile_ascphase(index,end)-
multiplier*profile_descphase(index,end);
else index=index;
end
timenewconstphase=injvolconstphase/(multiplier*(profile_flowrate(i
ndex,18)*1000)); %milliseconds
newvector_time=profile_time(index,:);
newvector_time(19)=profile_time(index,18)+timenewconstphase;
newvector_time(20:end)=timenewconstphase+profile_time(index,20:end
);
newvector_flowrate_mass=multiplier*profile_flowrate(index,:)*.85;
newvector_flowrate_mass(19)=multiplier*profile_flowrate(index,19)*
.85;
newvector_flowrate_mass(20:length(profile_time))=multiplier*profil
e_flowrate(index,20:end)*.85;

matrixnewvector(k,:)=newvector_time*1000;
matrixnewvector(l,:)=newvector_flowrate_mass;
k=k+2;
l=l+2;

figure(2)
plot(newvector_time*1000,newvector_flowrate_mass)
hold on
end

figure(2)
grid on
xlabel('Time [ms]');
ylabel('Mass flow rate [g/s]');
xlim([0 9])
ylim([0 40])
legend({'Inj.Mass= 10 mg, PRail= 350 bar','Inj.Mass= 15 mg, PRail=
350 bar','Inj.Mass= 20 mg, PRail= 350 bar'...
    ,'Inj.Mass= 25 mg, PRail= 540 bar', 'Inj.Mass= 30 mg, PRail=
730 bar','Inj.Mass= 35 mg, PRail= 920 bar'...
    ,'Inj.Mass= 40 mg, PRail= 1110 bar','Inj.Mass= 45 mg, PRail=
1160 bar','Inj.Mass= 50 mg, PRail= 1198 bar'...
    ,'Inj.Mass= 55 mg, PRail= 1233 bar','Inj.Mass= 60 mg, PRail=
1268 bar','Inj.Mass= 65 mg, PRail= 1303 bar'...
    ,'Inj.Mass= 70 mg, PRail= 1338 bar','Inj.Mass= 75 mg, PRail=
1359 bar','Inj.Mass= 80 mg, PRail= 1380 bar'...
    ,'Inj.Mass= 85 mg, PRail= 1400 bar','Inj.Mass= 90 mg, PRail=
1400 bar','Inj.Mass= 100 mg, PRail= 1400 bar'});
```

References

[1] Courant, R., Friedrichs, K., Lewy, H.,, "On the partial difference equations of mathematical physics.", IBM J. 11, pp. 215-234, 1967.

[2] Gamma Technologies, GT-SUITE Flow manual, 2018.

[3] Spessa, E., Design of Engine and Control System course, Politecnico di Torino, 2018/2019.

[4] Millo, F., Engine Emissions Control course, Politecnico di Torino, 2018/2019.

[5] Delprete, C., Powertrain components design course, Politecnico di Torino, 2018.

[6] Gamma Technologies, GT-SUITE Engine Performance Tutorials, 2018.

[7] D'Ambrosio, S., Internal Combustion Engines and Their Application to Vehicle course, Politecnico di Torino, 2018/2019.

[8] Fragiacomo, P., Corso di Fondamenti di Fluidodinamica, Università della Calabria, 2015/2016.

[9] Gamma Technologies, GT-Suite Engine Performance manual, 2018.

[10] Springer Vieweg, "Diesel Engine Management Systems and Components," *Bosch Professional Automotive Information,* 2014.

[11] D'Ambrosio, S., Ferrari, A., "Diesel engines equipped with piezoelectric and solenoid injectors: hydraulic performance of the injectors and comparison of the emissions, noise and fuel consumption," *Applied Energy,* pp. 1324-1342, 2018.

[12] A. Piano and et al., "Experimental and Numerical Assessment of Multi-Event Injection Strategies in a Solenoid Common-Rail Injector," *SAE Int. J. Engines,* vol. 10, no. 4, 2017.

[13] A. Piano and et al., "Numerical and Experimental Assessment of a Solenoid Common-Rail Injector Operation with Advanced Injection Strategies," *SAE Int. J. Engines,* vol. 9, no. 1, pp. 572-573, 2016.

[14] A. Piano and et al., "Numerical Simulation of the Combustion Process of a High EGR, High Injection Pressure, Heavy Duty Diesel Engine," *SAE Int. J. Engines,* 2017.

[15] Jeftic, M., Yu, S., Han, X., Reader, G., Wang, M., Zheng M., "Effects of Postinjection Application with Late Partially Premixed Combustion on Power Production and Diesel Exhaust Gas Conditioning," *Journal of Combustion,* vol. Volume 2011, p. 5, 2011, doi:10.1155/2011/891096.

Thanks

After the most important thanks to my dears written in the first pages of this thesis, there are many other people and parties to be grateful to.

The main thanksgiving goes to my home University tutor, Professor Federico Millo, who got me passionate with engines through his wonderful lectures of the Engine Emission Controls course. Very reluctant to internal combustion engines in the beginning because of the unpractical (but extremely necessary) thermodynamics lessons, he made spontaneously a great job in changing my mind and make me fall in love with the study of such a complex but amazing energy and motion source, the internal combustion engine. Moreover, many thanks go to him for having put his trust in me accepting to be my tutor for the thesis and also for always being ready to help me live and via phone at 2000 km of distance, even in non-working days, despite his many commitments.

Big thanks go also to my host University tutor, the very kind Professor Mika Lauren, for agreeing and supporting me in the changes of direction I had during the thesis work, to get the best out of it.

Gratefulness must be expressed to the Turku University of Applied Sciences, who made me feel home from the first day with its amazing hospitality and guidance, and for providing me the dream workplace: private office with an excellent workstation inside and a GT-Suite 2018 license.

Thanks go also to Dr. Andrea Piano and Dr. Alessandro Zanelli, Polytechnic of Turin's PhD. researchers who gave me assistance in developing the engine model.

Last but not least, several thanks go to all the professors I had from the high school to Università della Calabria, who gave me more than the necessary knowledge for starting my Master degree journey and trained me to study even during the nights and the weekends, to achieve satisfying academical results.