

Master of Science in Automotive Engineering

Vehicle Performance Simulations with **OpenModelica**

A Master's Thesis in collaboration with:



**Politecnico
di Torino**



Author: André Fernandes Reduto

Politecnico di Torino Supervisor: Prof. Federico Millo

PUNCH Torino Supervisors: Marcello Rimondi & Paolo Ferreri

April 2023



Acknowledgements

Trying to decide what to do in life can be tricky, and often-times frightening. Often times one might know what subjects they like but be less aware of what applications they may have. Such was my case when arriving at the conclusion of my studies. For this reason, I would like to thank the work done by Professor Federico Millo not only in finding me and proposing to me this thesis, but also new careers opportunities for my future. Through this thesis, my eyes have been opened wide to the world of simulation, and a new passion has been created within me to find future career options in such domains.

I would also like to thank everyone at PUNCH, particularly those of the 1D analysis & virtual calibration group of Ing. Marcello Rimondi for welcoming me into the team with such open arms. The nine months spent at PUNCH have been an amazing experience with a working environment so pleasant, and a team so friendly, that it was always a joy to come into the office. A special thank-you goes out to Ing. Paolo Ferreri, Dr. Francesco Sapio and Ing. Filippo Aglietti for always being available to help me out whenever a new roadblock was reached.

I secondary thank-you goes out to my family continued to support and encourage me to get my engineering degree. My parents, Ana Paula Queirós Fernandes & José Fernando Alves da Costa Pereira, and grandparents, Emilia Queirós Fernandes & Gaspar Fernandes, have endured the financial burden of my studies, on top of the results of my outlandish decisions which include the adoption of my two large dogs Oli and Barney. My best friends, Tomás Dominguez Chiozza and Olivier Ian Newman, people I have now known for over a decade, have also constantly motivated me and helped me with studying for my degree. I owe everything to these six individuals, for without them, I would not be finishing this degree. I am, truly, from the bottom of my heart, thank-full.

In loving memory of Prof. Elvio Bonisoli... a brilliant PoliTO professor, who passed away way too young, with whom I did my bachelor final project. He will forever be remembered for his joyful personality, his contribution in educating the next generation of engineers, and for his contributions to the world of academia.



Abstract

In a world characterized by themes like decarbonization and pollution reduction receiving increasing attention, the automotive industry is experiencing a revolution. Manufacturers are striving to achieve new eco-friendly solutions capable of reducing environmental impact whilst remaining cost-competitive for the consumer. To achieve such goals, increasingly complex propulsion systems are being developed that aim to minimize energy consumption whilst ensuring good, or improved performance. Traditional approaches heavily based on procurement and testing of physical hardware are no longer a viable option for the development of these new solutions since they would be too slow and expensive. This is especially true when all the steps normally needed to find the best solution starting from draft designs are considered. In this context, numerical simulation represents a key element capable of supporting the entire development process from the concept phase up to the validation one, reducing the need for physical prototypes. This is possible through CAE tools capable of facilitating the testing of several powertrain solutions and predicting a wide range of performance indicators, from fuel consumptions to failure points. With the help of computers, complex calculations and simulations can be performed, allowing the exploration of trade-offs for a huge set of solutions in a reduced timeframe.

This work aims to explore the viability of an open-source software such as OpenModelica as an alternative to costly commercial licenses. OpenModelica is a tool developed in the Modelica programming language, which has been developed by the Modelica Association for simulation environments. Since the software is open source, information is openly shared by OpenModelica users online, whilst pre-existing models and libraries can be found online for a variety of applications. An advantage of such is that users can easily build upon pre-existing libraries built by previous developers such as to create models for systems and set-ups that hadn't previously been considered or built. Consequently, the focus of this work is to create an ad-hoc model library for PUNCH Torino for modeling battery electric vehicles and fuel cell electric vehicles. The custom library aims to evaluate vehicle performance parameters such as acceleration times and fuel economy, whilst the choice to create a library from the ground up allows to optimize models running time, whilst simultaneously ensuring an appropriate model accuracy. In order to validate the tool, vehicle models are created, assembled and simulated, with results being compared directly with those from models already available and developed in GT-SUITE. The slight deviations detected between the simulation results, with errors ranging from zero to ten percent due to differences between the software and the modeling approaches taken, confirm the reliability of the models developed and provide an initial point from which the project can be expanded upon.



Table of Contents

Abstract.....	3
List of Acronyms.....	11
List of Symbols	12
#1 - Introduction	13
Thesis Objective	13
What is OpenModelica?	13
Modelica Standard Library.....	13
Commercial Libraries	13
Determining Viability.....	13
#2 - Library Development.....	15
Quick “Classes” Overview.....	15
Modelling.....	15
The Library.....	15
#3 Modeling Components.....	17
Driver Models	17
Simple Driver Model.....	17
Simple Driver Model for Accelerations	18
Target Speed Traces.....	19
SORT 1, SORT 2 and SORT 3.....	19
Braunschweig	20
Old MLTB.....	20
RDC #1	21
Energy Storage.....	23
Battery Model.....	23
Fuel Cell	26
Balance of Plant	27
Vehicle.....	28
Simple Bus – Variant 1	28
Simple Bus – Variant 2	29
Torque Source.....	30
SMPM Motor & Inverter	30
Electronics	33
Simple Boost	33
Constant Auxiliaries	33
Motor Control Unit.....	34
Hybrid Control Unit	35
Icons	38
#4 Model Overview & Results	39



BEV Model	39
BEV Model Assembly	39
Driver Calibration.....	40
BEV Model Results.....	41
FC Hybrid Model	54
FCH Model Assembly	54
FCH Model Results.....	55
#5 Conclusions.....	68
Bibliography	70
Books.....	70
Websites	70
Other	70
ANNEX A – Equations.....	71
Battery Current	71
ANNEX B – MATLAB Scripts.....	72
Power Limited Motor Operation.....	72
Post Processing Scripts.....	73
ANNEX C – Some BEV Results Comparisons	81
BEV Fuel Economy Tests	81
BEV Vehicle Acceleration/Slope Tests.....	89
ANNEX D – Some FCH Results Comparisons.....	92
FCH Fuel Economy Tests (PAUX = 7 kW)	92
FCH Fuel Economy Tests (PAUX = 0 kW)	100
FCH Vehicle Acceleration/Slope Tests.....	103



Table of Figures & Tables

Figure 2.01 – PUNCH_Torino Library	15
Figure 3.01 – Simple Driver Model	17
Figure 3.02 – Simple Driver Model for Accelerations	18
Figure 3.03 – Speed Cycles	19
Figure 3.04 – SORT 1, SORT 2 & SORT 3	19
Figure 3.05 – Braunschweig	20
Figure 3.06 – MLTB	21
Figure 3.07 – RDC #1	21
Figure 3.08 – PUNCH Battery Pack – Variant 1	23
Figure 3.09 – PUNCH Battery Pack – Variant 2	23
Figure 3.10 – Battery Cells	23
Figure 3.11 – MSL 4.0 Battery Cell	24
Table 3.01 – Battery Layouts	25
Figure 3.12 – Fuel Cell Model	26
Figure 3.13 – Balance of Plant Model	27
Figure 3.14 – Bus Longitudinal Dynamics, Variant 1	28
Figure 3.15 – Bus Longitudinal Dynamics, Variant 2	29
Figure 3.16 – Motor & Inverter Lumped Model	30
Figure 3.17 – Inverter Efficiency Map	30
Figure 3.18 – Motor Efficiency Map	30
Figure 3.19 – Inverted Motor & Inverter Map	31
Figure 3.20 – Simple Boost Model	33
Figure 3.21 – Constant Auxiliaries Model	33
Figure 3.22 – Motor Control Unit Model	34
Figure 3.23 – State Graph Hybrid Control Unit Model	35
Figure 3.24 – RBC Maps for Charge Level One and Two	37
Figure 3.25 – Driver Icon	38
Figure 3.26 – Battery Icon	38
Figure 3.27 – Bus Icon	38
Figure 3.28 – Control Unit Icon	38
Figure 4.01 – BEV Powertrain Configuration	39
Figure 4.02 – OpenModelica BEV Model	39
Figure 4.03 – GT-SUITE BEV Model	40
Table 4.01 – Raw BEV Results – 475.2 kWh Battery, $P_{aux} = 7$ kW	41
Table 4.02 – Comparison of GT-SUITE & OpenModelica BEV Results ($P_{aux} = 7$ kW)	42
Figure 4.04 – Zoom on Comparison of BEV Speed Profiles (SORT3, 16 000 kg, $SOC_0 = 80$ % & $P_{aux} = 7$ kW)	43
Figure 4.05 – Zoom on Comparison of BEV Speed Profiles (SORT3, 16 000 kg, $SOC_0 = 80$ % & $P_{aux} = 7$ kW)	44
Table 4.03 – GT-SUITE & OpenModelica BEV Error Trackers ($SOC_0 = 80$ %, $P_{aux} = 7$ kW)	44



Figure 4.06 – Zoom on Comparison of BEV Speed Profiles (Braunschweig, 16 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW)	45
Figure 4.07 – Zoom on Comparison of BEV Speed Profiles (RDC #1 x3, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW)	46
Figure 4.08 – Zoom on Inclination Profile (RDC #1 x3, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW)	46
Figure 4.09 – Zoom on Comparison of BEV Inclination Profiles (RDC #1 x3, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW)	47
Figure 4.10 – Comparison of BEV Motor Torque Traces (SORT3, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW)	48
Figure 4.11 – Comparison of BEV Motor Torque Traces (First 500 s of MLTB, 16 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW)	49
Table 4.04 – GT-SUITE & OpenModelica BEV Acceleration Times ($P_{aux} = 7.5$ kW)	49
Figure 4.12 – Comparison of BEV Acceleration (Case #1).....	50
Figure 4.13 – Comparison of BEV Motor Torque during Acceleration Maneuver (Case #1).....	51
Figure 4.14 – Comparison of BEV Motor Torque during Acceleration Maneuver (Case #5).....	51
Figure 4.15 – Comparison of BEV Motor Torque during Acceleration Maneuver (Case #5).....	52
Figure 4.16 – Comparison of BEV Motor Torque during Acceleration Maneuver (Case #16)	52
Figure 4.17 – Simple Series Hybrid Powertrain.....	54
Figure 4.18 – OpenModelica FC Hybrid EV Model.....	54
Figure 4.19 – GT-SUITE FC Hybrid EV Model.....	55
Table 4.05 – Raw FC Hybrid Vehicle Results – 39.6 kWh Battery	55
Table 4.06 – Comparison of GT-SUITE & OpenModelica FCH RAW Results ($SOC_0 = 62\%$, $P_{aux} = 7$ kW)	56
Table 4.06 – Comparison of GT-SUITE & OpenModelica FCH Equivalent Results ($SOC_0 = 62\%$, $P_{aux} = 7$ kW)	57
Table 4.07 – GT-SUITE & OpenModelica FCH Error Trackers ($SOC_0 = 62\%$)	58
Figure 4.20 – Comparison of FCH Fuel Cell Gross Power (SORT1, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)	59
Figure 4.21 – Comparison of FCH SOC Discharge Profiles (SORT1, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)	59
Figure 4.23 – Comparison of FCH Hybridization Mode (SORT1, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)	60
Figure 4.22 – Comparison of FCH Fuel Cell Gross Power (MLTB, 16 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)	61
Figure 4.24 – Zoom on Comparison of FCH Speed Profiles (Braunschweig, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)	61
Table 4.08 – Comparison of GT-SUITE & OpenModelica FCH RAW Results ($SOC_0 = 62\%$, $P_{aux} = 0$ kW)	63
Figure 4.25 – Comparison of FCH Hybridization Mode (MLTB, 16 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 0$ kW) ..	63
Figure 4.26 – Comparison of FCH Hydrogen Consumed (MLTB, 16 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 0$ kW) ..	64
Table 4.09 – Comparison of GT-SUITE & OpenModelica FCH Equivalent Results ($SOC_0 = 62\%$, $P_{aux} = 0$ kW)	64
Figure 4.27 – Comparison of FCH Vehicle Fuel Cell Gross Power (MLTB, 16 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 0$ kW)	65
Table 4.10 – GT-SUITE & OpenModelica FCH Acceleration Times ($P_{aux} = 7.5$ kW)	65



Figure 4.28 – Comparison of FCH Acceleration (Case #4, 19 000 kg, 0 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW).....	66
Figure 4.29 – Comparison of FCH Torque Trace (Case #4, 19 000 kg, 0 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW).....	66
Figure 4.30 – Comparison of FCH Fuel Cell Gross Power (Case #4, 19 000 kg, 0 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW).....	67
Figure 4.31 – Comparison of FCH Hybridization Model (Case #4, 19 000 kg, 0 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW).....	67
Figure A.01 – Simple Battery & Load.....	71
Figure C.01 – Comparison of BEV Speed Profiles (SORT1, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW).....	81
Figure C.02 – Comparison of BEV SOC Discharge Profiles (SORT1, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW).....	81
Figure C.03 – Comparison of BEV Torque Trace (SORT1, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW).....	82
Figure C.04 – Comparison of BEV Battery Power (SORT1, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW).....	82
Figure C.05 – Comparison of BEV Speed Profiles (Braunschweig, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW).....	83
Figure C.06 – Comparison of BEV SOC Discharge Profiles (Braunschweig, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW).....	83
Figure C.07 – Comparison of BEV Torque Trace (Braunschweig, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW).....	84
Figure C.08 – Comparison of BEV Battery Power (Braunschweig, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW).....	84
Figure C.09 – Comparison of BEV Speed Profiles (MLTB, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW).....	85
Figure C.10 – Comparison of BEV SOC Discharge Profiles (MLTB, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW).....	85
Figure C.11 – Comparison of BEV Torque Trace (MLTB, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW).....	86
Figure C.12 – Comparison of BEV Battery Power (MLTB, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW).....	86
Figure C.13 – Comparison of BEV Speed Profiles (RDC #1, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW).....	87
Figure C.14 – Comparison of BEV SOC Discharge Profiles (RDC #1, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW).....	87
Figure C.15 – Comparison of BEV Torque Trace (RDC #1, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW).....	88
Figure C.16 – Comparison of BEV Battery Power (RDC #1, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW).....	88
Figure C.17 – Comparison of BEV Acceleration (Case #9, 19 000 kg, 0 % inclination, $SOC_0 = 85\%$ & $P_{aux} = 7.5$ kW).....	89
Figure C.18 – Comparison of BEV Torque Trace (Case #9, 19 000 kg, 0 % inclination, $SOC_0 = 85\%$ & $P_{aux} = 7.5$ kW).....	89
Figure C.19 – Comparison of BEV Acceleration (Case #11, 19 000 kg, 0 % inclination, $SOC_0 = 65\%$ & $P_{aux} = 7.5$ kW).....	90
Figure C.20 – Comparison of BEV Torque Trace (Case #11, 19 000 kg, 0 % inclination, $SOC_0 = 65\%$ & $P_{aux} = 7.5$ kW).....	90
Figure C.21 – Comparison of BEV Acceleration (Case #16, 19 000 kg, 4 % inclination, $SOC_0 = 25\%$ & $P_{aux} = 7.5$ kW).....	91
Figure C.22 – Comparison of BEV Torque Trace (Case #16, 19 000 kg, 4 % inclination, $SOC_0 = 25\%$ & $P_{aux} = 7.5$ kW).....	91
Figure D.01 – Comparison of FCH Speed Profiles (SORT3, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW).....	92
Figure D.02 – Comparison of FCH SOC Discharge Profiles (SORT3, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW).....	92
Figure D.03 – Comparison of FCH Torque Trace (SORT3, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW).....	93



Figure D.04 – Comparison of FCH Battery Power (SORT3, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)	93
Figure D.05 – Comparison of FCH Hybridization Mode (SORT3, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW).....	94
Figure D.06 – Comparison of FCH Hydrogen Consumed (SORT3, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)	94
Figure D.07 – Comparison of FCH Fuel Cell Gross Power (SORT3, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)	95
Figure D.08 – Comparison of FCH Speed Profiles (Braunschweig, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)	95
Figure D.09 – Comparison of FCH SOC Discharge Profiles (Braunschweig, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)	96
Figure D.10 – Comparison of FCH Hybridization Mode (Braunschweig, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)	96
Figure D.11 – Comparison of FCH Hydrogen Consumed (Braunschweig, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)	97
Figure D.12 – Comparison of FCH Fuel Cell Gross Power (Braunschweig, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)	97
Figure D.13 – Comparison of FCH Speed Profiles (RDC #1, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)	98
Figure D.14 – Comparison of FCH SOC Discharge Profiles (RDC #1, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)	98
Figure D.15 – Comparison of FCH Hybridization Mode (RDC #1, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)	99
Figure D.16 – Comparison of FCH Hydrogen Consumed (RDC #1, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)	99
Figure D.17 – Comparison of FCH Fuel Cell Gross Power (RDC #1, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)	100
Figure D.18 – Comparison of FCH Speed Profiles (Braunschweig, 16 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 0$ kW)	100
Figure D.19 – Comparison of FCH SOC Discharge Profiles (Braunschweig, 16 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 0$ kW)	101
Figure D.20 – Comparison of FCH Hybridization Mode (Braunschweig, 16 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 0$ kW)	101
Figure D.21 – Comparison of FCH Hydrogen Consumed (Braunschweig, 16 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 0$ kW)	102
Figure D.22 – Comparison of FCH Fuel Cell Gross Power (Braunschweig, 16 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 0$ kW)	102
Figure D.23 – Comparison of FCH Acceleration (Case #2, 16 000 kg, 4 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW)	103
Figure D.24 – Comparison of FCH Torque Trace (Case #2, 16 000 kg, 4 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW)	103
Figure D.25 – Comparison of FCH Fuel Cell Gross Power (Case #2, 16 000 kg, 4 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW).....	104
Figure D.26 – Comparison of FCH Hybridization Model (Case #2, 16 000 kg, 4 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW).....	104
Figure D.27 – Comparison of FCH Acceleration (Case #3, 16 000 kg, 8 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW)	105



Figure D.28 – Comparison of FCH Torque Trace (Case #3, 16 000 kg, 8 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5\text{ kW}$)	105
Figure D.29 – Comparison of FCH Fuel Cell Gross Power (Case #3, 16 000 kg, 8 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5\text{ kW}$)	106
Figure D.30 – Comparison of FCH Hybridization Model (Case #3, 16 000 kg, 8 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5\text{ kW}$)	106
Figure D.31 – Comparison of FCH Acceleration (Case #6, 19 000 kg, 8 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5\text{ kW}$)	107
Figure D.32 – Comparison of FCH Torque Trace (Case #6, 19 000 kg, 8 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5\text{ kW}$)	107
Figure D.33 – Comparison of FCH Fuel Cell Gross Power (Case #6, 19 000 kg, 8 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5\text{ kW}$)	108
Figure D.34 – Comparison of FCH Hybridization Model (Case #6, 19 000 kg, 8 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5\text{ kW}$)	108

List of Acronyms

<u>Acronym</u>	<u>Definition</u>
EU	European Union.
OMedit	Open Modelica Editor.
MSL	Modelica Standard Library.
MSL 4.0	Modelica Standard Library version 4.0.
CAE	Computer-Aided Engineering.
EV	Electric Vehicle.
BEV	Battery Electric Vehicle.
FCH	Fuel Cell Hybrid Vehicle.
MLTB	Millbrook London Transport Bus (Driving Cycle).
RDC	Real Driving Cycle.
PID	Proportional-Integral-Derivative Controller.
PIDF	Proportional-Integral-Derivative Controller with a Filter.
SOC	State of Charge.
OC	Open Circuit.
OCV	Open Circuit Voltage.
RC	Resistive-Capacitive.
FC	Fuel Cell.
BOP	Balance of Plant.
SG	State Graph.
RBC	Rule Based Controller.
eCMS	Equivalent Consumption Management Strategy.
LVL	Level.
HV	High Voltage.
LV	Low Voltage.
LHV	Lower Heating Value.
DOE	Design of Experiment.
FMU	Functional Mock-up Unit.

List of Symbols

<u>Symbol</u>	<u>Definition</u>	<u>Symbol</u>	<u>Definition</u>
$G(s)$	Controller Transfer Function in Laplace domain.	τ	Low Pass filter Time Constant [s].
K_p	Proportional Gain [-].	K_i	Integral Gain [-].
K_D	Derivative Gain [-].	$e(s)$	Velocity Error.
s	Laplace Variable.	V_{oc}	Open Circuit Voltage [V].
$I_{dis,max}$	Maximum Discharge Current [A].	R_{eq}	Equivalent Internal Voltage [Ω].
$I_{ch,max}$	Maximum Charge Current [A].	P_{dis}	Maximum Discharge Battery Power [W].
N_s	Number of Cells in Series [-].	P_{ch}	Maximum Charging Battery Power [W].
N_p	Number of Units in Parallel [-].	N_{sm}	Number of Modules in Series [-].
R_g	Universal Gas Constant [kg m ² /K/mol].	F	Faradays Constant [C/mol].
α	Charge Transfer Coefficient [-].	V_{ic}	Internal Voltage Loss [V].
j	Current Density [A/cm ²].	V_{act}	Activation Voltage Loss [V].
j_0	Activation Current Density [A/cm ²].	j_1	Limiting Current Density [A/cm ²].
C_{mt}	Mass Transfer Coefficient.	V_{mt}	Mass Transfer Voltage Loss [V].
T	Temperature [K].	H_2O	Water Produced [kg].
H_2	Hydrogen Consumed [kg].	O_2	Oxygen Consumed [kg].
i	Current [A].	n_s	Number of Fuel Cell Stacks.
\mathcal{M}	Molar Mass [kg/mol].	F_t	Traction Force [N].
ρ	Ambient Air Density [kg/m ³].	S	Vehicle Projected Surface Area [m ²].
f	Constant Tire Rolling Resistance [-].	C_x	Vehicle Drag Coefficient.
K	Speed Dependence of Tire Rolling Resistance [-].	V	Vehicle Velocity [m/s].
g	Acceleration due to Gravity [m/s ²].	m	Vehicle Mass [kg].
θ	Road Slope [deg].	η	Motor or Inverter Efficiency [-].
ω	Motor Angular Velocity [rad/s].	T	Motor Torque [Nm].

#1 - Introduction

Punch Torino is an engineering firm owned by the Punch Group, a firm that specializes in solutions for sustainable mobility. Having previously been owned by General Motors under the name “General Motors Global Propulsion Systems”, Punch Torino developed, and continues to develop, many diesel combustion engines for large clients in various sectors. Furthermore, through the Punch Hydrocells brand, the Punch Group is now developing H₂ solutions for mobility and technologies for H₂ production, storage and distribution. One of the tools utilized for such activities is GT-SUITE, by Gamma Technologies, a commercial software package typically for CAE of internal combustion engines, powertrains, and vehicle simulations.

Thesis Objective

In order to reduce an engineering firms operating costs, it is possible to reduce reliance on commercial licenses and obtain certain preliminary results from free open-source software. With this in mind, an open-source software can be explored via a case study to determine its viability. A list of objectives for the study can be listed as follows:

- Build a software package with OpenModelica, capable to evaluate:
 - 1) Vehicle performance indexes (ie; maximum velocity, 0-50km/h acceleration , etc...).
 - 2) Range/Fuel economy for BEV and/or Fuel Cell Hybrid vehicles.
- Compare these results with ones obtained by a commercial software (such as MATLAB or GT-SUITE).

What is OpenModelica?

OpenModelica is an open-source simulation environment developed by the nonprofit known as the *Modelica Association*. It utilizes the Modelica programming language, an object-oriented programming language specifically created for optimizing simulations. To utilize OpenModelica, an editor such as OMEdit is used, which allows for both text programming and graphical programming. OMEdit also compiles and runs simulations, generates graphs, animations, etc. for analysis. Other editors such as Dymola by *Dassault Systems* also exist but are typically commercial products.

Modelica Standard Library

The Modelica Standard Library (MSL) is a toolbox produced by the *Modelica Association* which comes pre-installed on OMEdit. It contains both basic and complex models of mechanical, electrical, fluid etc. components. Components within the library can represent anything from stand-alone equations to complex physical systems such as an electric motor.

Commercial Libraries

Although OpenModelica was originally developed to be a free platform, several engineering firms have been able to successfully commercialize the software by developing large commercially available libraries containing more complex base models than those available in the MSL. These can be installed onto a user's OMEdit and utilized alongside the MSL and other user created models to save time or run more complex simulations. Such libraries can include but are not limited to:

- The Vehicle Dynamics Library by *Modelon AB*
- The Electrified Powertrains Library by *Claytex Technica Company*
- The Hydrogen Library by *Claytex Technia Company*
- Real Time Car Simulator Library by *Altair*

As the scope of this project is to analyze the potential for savings using free software's, none of the above are considered.

Determining Viability

To determine the success of the project, baseline results generated in GT-SUITE are used due to their availability from other PUNCH projects. Furthermore, these projects can also be used for sourcing data



throughout the simulations in OpenModelica. Via comparing the numerical results obtained, OpenModelica's capabilities can be objectively measured. The success of OpenModelica's implementation is also determined via its ease of use, simulation speed and capabilities, as these factors also determine how easily it can become a substitute for its commercial counterparts. By analyzing the problems encountered throughout the project, OpenModelica's viability can be subjectively measured.

#2 - Library Development

Quick “Classes” Overview

Modelica files are structured into objects defined as “*classes*”, whose definitions can be found in Modelica support pages online. For the sake of brevity, only three *classes* deemed to be of fundamental importance will be explained in this text:

- *Package* – behaves like a folder into which other classes can be stored, including other packages themselves. These packages can be bestowed properties and directly imported into models.
- *Model* – is a standalone model which can be simulated by itself.
- *Block* – is a component that acts like a graphical programming function, with an input and an output, and cannot be simulated by itself.

Other classes include but are not limited to *functions*, *types*, *connector*, etc... Certain properties can also be imposed onto classes such as *partial* to facilitate a parent class importing characteristics of the aforementioned partial class.

Modelling

Modelling in Modelica can be done in one of two ways; either graphically in manners similarly to Simulink/Simscape, or textually in manners more resembling traditional programming. That being said, a key characteristic of programming in Modelica is that it is a non-sequential language, with lines of code being simulated all at once independently of the order in which they are written. Whilst the project was originally tackled using a graphical approach, as it proved easier to work with, it’s limitations became quite evident as the system became progressively more complex.

The first issue encountered is that graphical programming required more overall variables and equations, meaning the system was slower to simulate. This is because calculations and interpolations that were unnecessary, such as those occurring for regenerative braking, were also being performed during instances where they were not being used. Further issues involved numerical chattering and Booleans not updating correctly despite the conditions having been met (numerical issues).

For this reason the entire library was re-built once familiarity with the Modelica programming language was sufficiently high, allowing for more compact models (often with one hundred or more equations less) which simulated much faster.

The Library

The “software package” mentioned in the thesis objectives was built and named “PUNCH_Torino” after the company name. The file is a *package* class or “library” containing several other *package* classes each containing *models* or *blocks* that sought to tackle a variety of issues or modelling requirements encountered with the software during the project. Not all *models* or *blocks* were used into the final vehicle simulations, as often other solutions were found to a problem, however, all of the components were kept in the library in case they may be of help for future users.

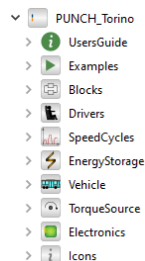


Figure 2.01 – PUNCH_Torino Library



The UsersGuide package simply contains documents such as library version, development data and contact information of the library, and is always the first package present in all Modelica libraries. The Examples package contains the final working models used for all graphs and results which will be displayed in this paper. The Blocks package contain certain custom blocks which were built for a variety of applications, mostly to be implemented in various driver models. The Drivers package contains driver models with varying behaviors and logics. The SpeedCycles package contains trivial models that import desired driving cycles into the simulations. The EnergyStorage package contains and battery, fuel cell models. The Vehicles package currently contains simplified longitudinal models of a bus' longitudinal dynamics, including a variant with slip. The TorqueSources package contains models of two map based electric motors developed for the project. The Electronics package contains models of inverters, boost DC/DC converters, and control units for the motor and hybrid systems. The final Icons package contains partial models of custom graphics to be imported into other packages.

#3 Modeling Components

Driver Models

In order to run forward simulations of the vehicle, a driver is necessary to provide adequate control signals to certain actuators. In the case of longitudinal dynamics, a throttle pedal position and a brake pedal position are necessary. In the case where a certain speed cycle must be followed, the driver must control said actuators such as to follow the driving cycle and minimizing the vehicles energy consumption. Alternatively, in the case of a pure acceleration or braking test with no tire slip, the driver can be substituted all together by constant throttle/brake signals.

Simple Driver Model

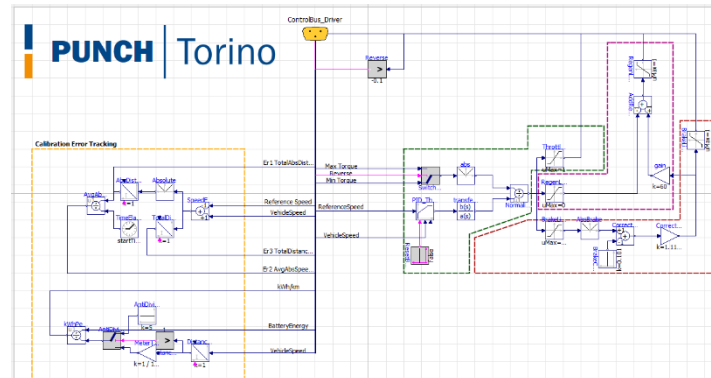


Figure 3.01 – Simple Driver Model

The simplest of the driver models included, shown in Figure 3.01, compares a reference speed provided by a speed cycle to the vehicles current speed. This driver model operates under the assumption that there is no tire slip, no gears, and that the vehicle motion is limited to one direction (forwards – even if there is a provision for reverse). For improved legibility, the model has been highlighted into four colored sections; the throttle and PIDF calculations in green, the regen calculations in purple, the brake calculations in red and the error tracking in yellow. The difference between the reference and actual speeds is evaluated and fed into a PID controller, whose parameters, shown in Equation 3.01 as K_p , K_i and K_d can be defined by the user to provide different controller characteristics. The output of the controller can be then fed through a low pass filter to remove the noise generated by the derivate gain, thus creating a PIDF controller.

$$G(s) = \frac{1}{1 + \tau s} \left(K_p e(s) + \frac{1}{s} K_i e(s) + s K_d e(s) \right) \quad \text{Eq 3.01}$$

To normalize the output, the signal is divided by the maximum torque the motor can provide at the current rotational velocity. This data is taken from the control bus and is provided by the motor model. If the resulting signal is positive, it is interpreted as a throttle, passed through a saturation block such as to ensure the output is always between zero and one, and fed back to the control bus. The saturation block is necessary to enforce the upper limit, and ignore the PIDF when it is negative, as throttle signals outside this range are invalid. If it is negative, and below a certain threshold, it is converted into a brake signal that actuates the friction brakes by taking the absolute value of said signal and correcting for the threshold such that the output is once between zero and one. This threshold value is included for two reasons; firstly, it allows for the driver to “coast” the vehicle using the resistive forces to slow down the vehicle, and secondly, it allows for the introduction of the regenerative braking effect normally found on EV’s when the driver let’s off the accelerator. The interpretation of the PID signal as both the accelerator and brake signal ensures both pedals are mutually exclusive and cannot be pressed simultaneously.

The third signal generated as a result, labelled “regen throttle” in the models, controls the regen intensity of the motors and how much power is put back into the battery. This signal is not representative of a real physical signal generated by the driver, but a consequence of how the motor is modelled. This regen signal is composed of the sum of two signals, one being the absolute value of the PIDF when negative, and one being an amplification of the brake signal. The latter represents the desire to brake as much as possible with the electric motor over the friction brakes, such as to minimize energy consumption. This signal is also later passed through a negative one gain and a saturation block to ensure the output is always between zero and one.

To help calibrating the driver, a few values referred to as errors are also evaluated in the driver model. These aim to provide numerical feedback to the user when calibrating such as to be allow them to make an accurate assessment as to if a modification to the PIDF resulted in the desired driver performance modification. This is necessary since when tuning the driver as the differences in performance may not be easily perceived. These calculations include:

- **Error1:** The integral of the absolute value of the difference in speeds between the vehicle and the reference trajectory. It is a distance and represents how far the driver was from the reference trajectory.
- **Error2:** The average absolute speed difference. This value is simply error1 normalized for time, such as to provide a different visualization of the difference.
- **Error3:** The difference in total distance covered by the reference cycle and the vehicle. This value can be both positive or negative depending on if the vehicle was on average travelling faster than it should’ve been, or slower. It is expected that this value does not surpass a few meters and should, ideally, be even less.
- **The Energy Consumption:** The power consumption associated to the battery’s difference in SOC, measured in kWh/km.

Simple Driver Model for Accelerations

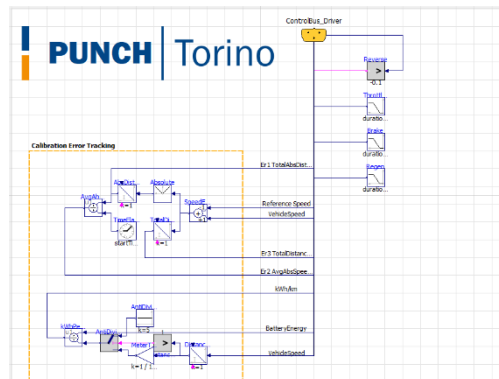


Figure 3.02 – Simple Driver Model for Accelerations

The driver model utilized for acceleration testing and inclination testing is an oversimplification of the simple driving model. It generates ramp signals of the relative to simulate a pedal being progressively pressed down to full throttle (or full brake) according to the time settings selected by the user. The error tracking features have been left in the model for being able to compare the results with other speed curves, possibly imported from other software.

Target Speed Traces

The speed cycles *package* contains the *models* that import reference cycles into the final vehicle models. This was deemed necessary since Modelica does not have the possibility to simulate “cases” like GT-SUITE wherein a parameter is changed and a batch of simulations occur with varying end times and cycles. Consequently, every time a new speed trace is needed, both files containing the reference speed cycles and inclinations need to be changed as well as the final simulation time of the model and its timestep. Thus, it was deemed easier to simply have different assemblies with different end times and associated timesteps to different speed cycle models.

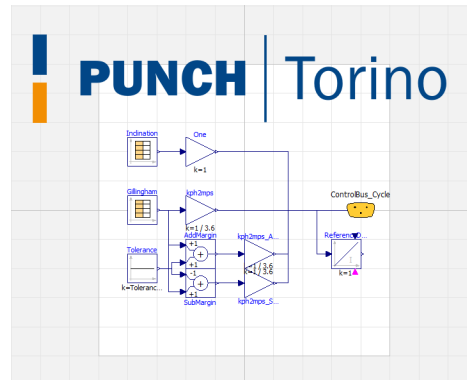


Figure 3.03 – Speed Cycles

As previously mentioned, all speed cycles models have the same structure with different target files. They provide a few parameters to the control bus useful for running other models or for analysis afterwards, such as the reference speed cycle, the inclination, an upper and lower speed margin to be used when calibrating the driver, and an integrator for determining the reference cycle distance.

SORT 1, SORT 2 and SORT 3

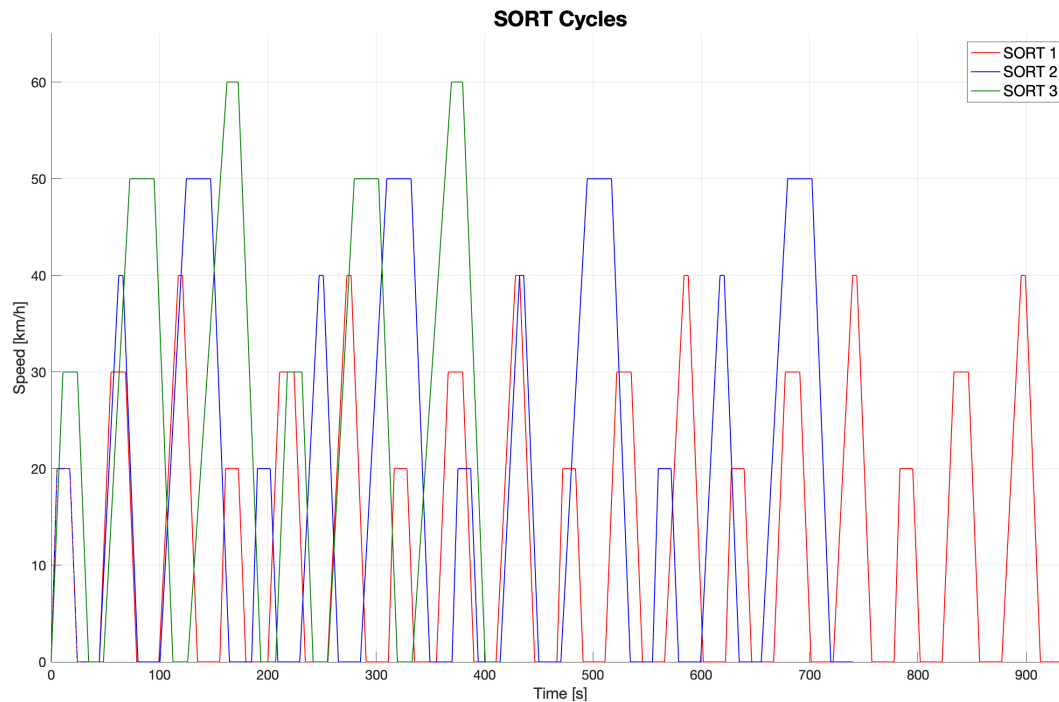


Figure 3.04 – SORT 1, SORT 2 & SORT 3

The SORT cycles are artificial driving cycles on a flat road wherein the vehicle accelerates at a constant rate to a target speed, stays at said speed for a certain time, then decelerates at a constant rate. As the SORT number increases, so too does the top speed value, whilst the acceleration is reduced. It is predominantly useful for calibration and testing of the models, as they are easier to analyze.

Braunschweig

The Braunschweig cycle is a more realistic driving cycle also performed on a flat surface with high top speeds and relatively hard accelerations. As it is both the most realistic and more reasonable of the six cycles displayed in this text, it was the one most used during the calibration procedure. This is down to the reasonably high velocities experienced, the duration and the relative flatness, which while not characteristic of every environment, is certainly less extreme than the RDC #1 seen later in Figure 3.07.

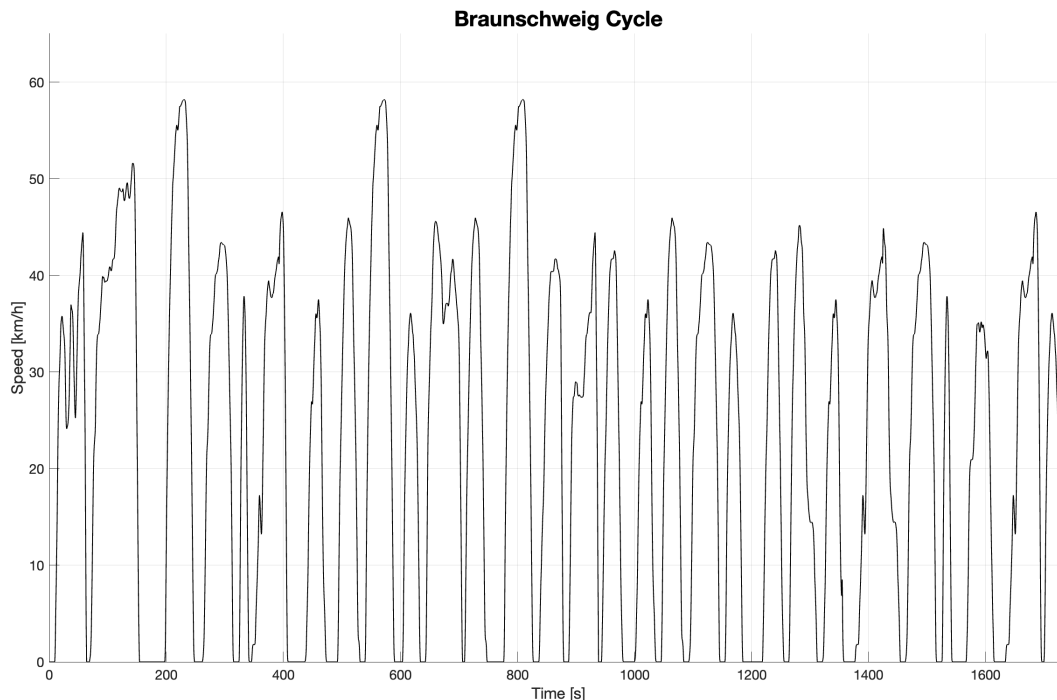


Figure 3.05 – Braunschweig

Old MLTB

The old Millbrook London Transport Bus driving cycle is a very calm driving cycle with lower average and top speed than the other realistic driving cycles included in this study. It is representative of a speed profile of a double decker bus driving around the congested streets of London along a designated route. It has recently received an update to make it more realistic.

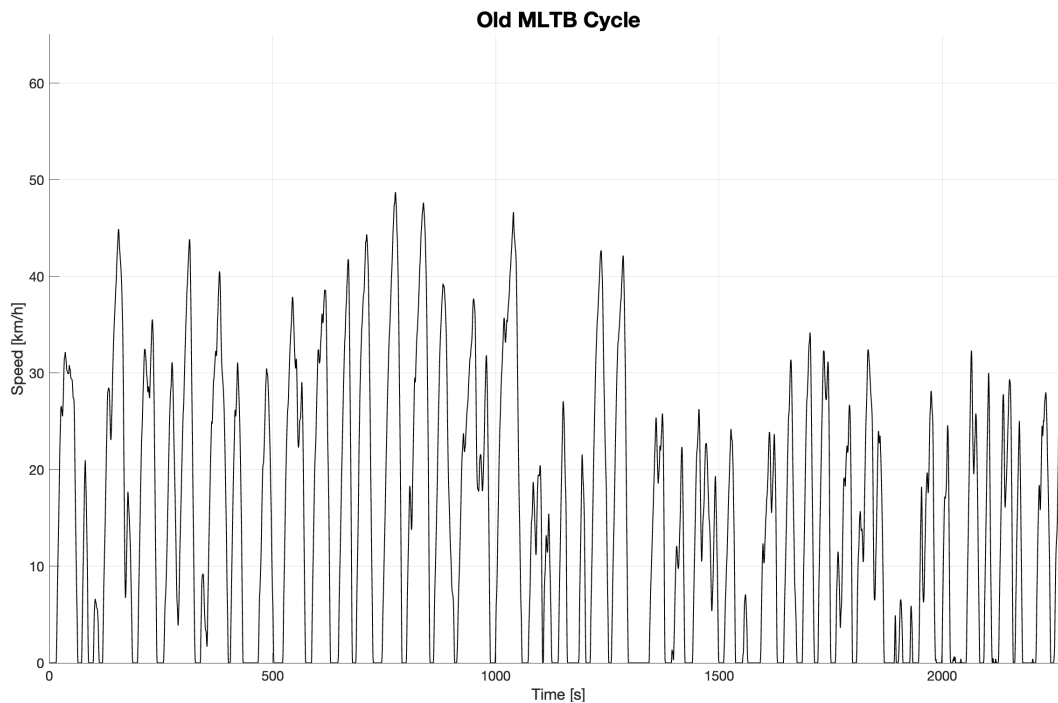


Figure 3.06 – MLTB

RDC #1

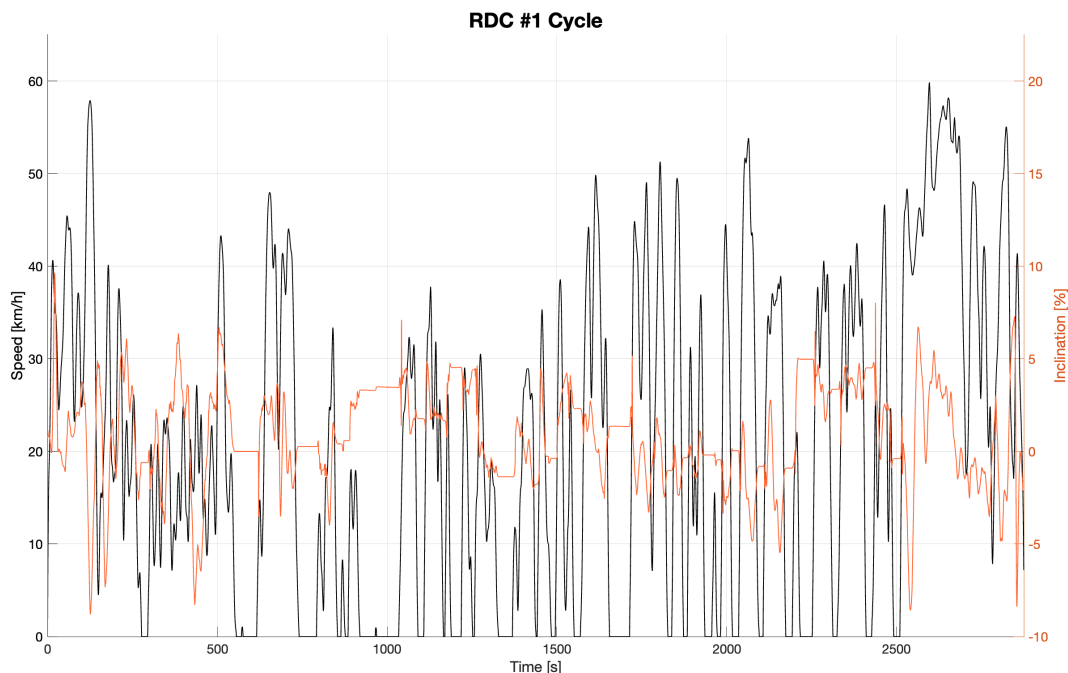


Figure 3.07 – RDC #1

The RDC #1 cycle is the only cycle present to not be performed on a flat surface, with very severe changes in the slope, ranging between over 9 % inclines and 8 % declines. It was found to be the cycle that the bus struggles with the most, as one would expect, both due the inclinations but also due to its longer length and thus higher overall battery discharge.



When testing the battery models, a longer variant of the RDC #1 cycle which will be referred to as “RDC #1 (x3)” was also used, in which three RDC #1 cycles were stitched together but with the central RDC #1 cycle done in reverse.

Note: The cycles shown above become progressively longer and thus the accelerations experienced by the vehicle are not directly comparable across graphs due to variations of the dimensions of the x-axis.

Energy Storage

The Energy Storage *package* was created to contain *models* of any component capable of storing and providing energy to the vehicle.

Battery Model

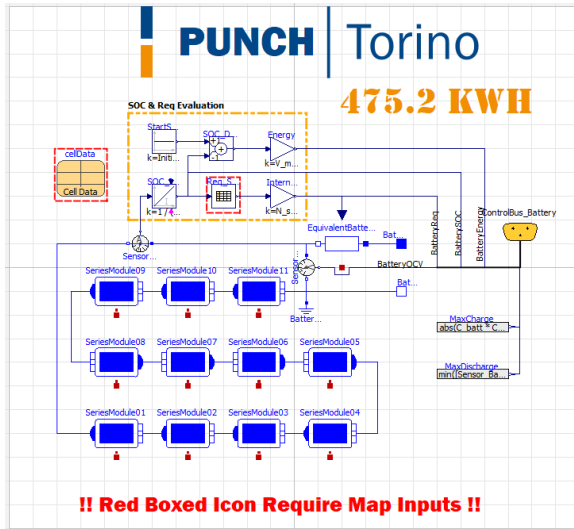


Figure 3.08 – PUNCH Battery Pack – Variant 1

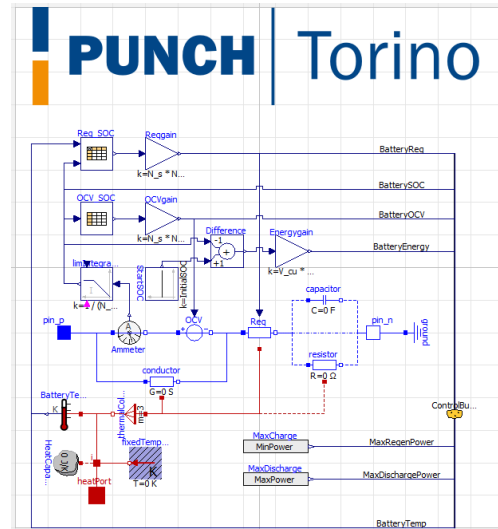


Figure 3.09 – PUNCH Battery Pack – Variant 2

Two battery models were created of varying complexities for the models, both capable of being used for vehicle simulations. Figure 3.08 displays the first variant, based on MSL4.0 cells, whilst Figure 3.09 represents an alternative battery pack model built from the ground up. Both battery packs operate under certain assumptions, such as uniform battery pack temperature of 20°C and have varying advantageous and limitations. Both battery packs are built the Thevenin battery model, utilizing several cells set up in series to create a “parallel unit”, then set up in parallel to create a “series module”, and once again set up in series to create the overall battery pack.

Battery Cells

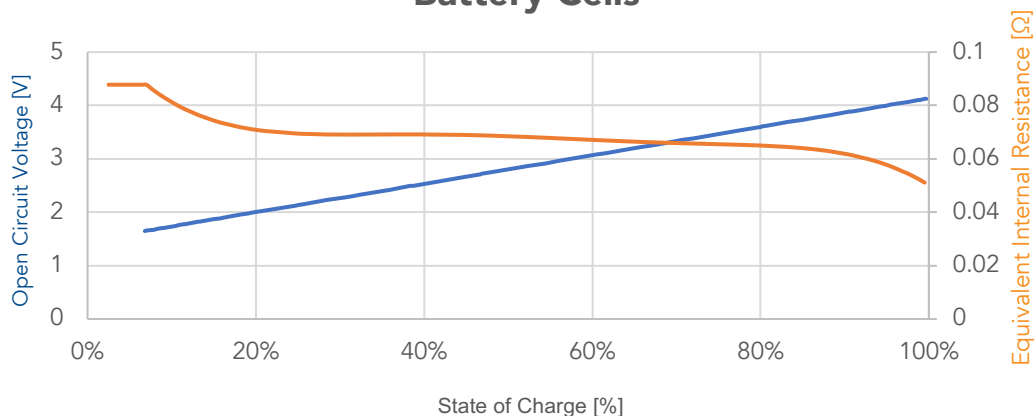


Figure 3.10 – Battery Cells

In both variants, certain calculations and data of the battery models is fed back into the control bus for calculations needed by other components, or data requested by controllers (such as the driver or the motor control unit). Examples of such data includes the temperature, the open circuit voltage, the internal resistance,

the state of charge, the maximum discharge power, and the maximum charging power. These latter two values can be calculated according to Equation 3.02 and Equation 3.03, in the absence of the RC components.

$$P_{dis}(SOC) = \min \left(\frac{v_{OC}^2(SOC)}{4 \cdot R_{eq}(SOC)}, v_{OC}(SOC) \cdot I_{max,dis} - R_{eq}(SOC) \cdot I_{max,dis}^2 \right) \quad \text{Eq 3.02}$$

$$P_{ch}(SOC) = -(v_{OC}(SOC) \cdot I_{max,ch} + R_{eq}(SOC) \cdot I_{max,ch}^2) \quad \text{Eq 3.03}$$

In both models, the useable battery capacity does not correspond to the full battery capacity, such as to preserve the components life. For this reason, the system warns the user if the battery is used outside of its acceptable range, whether it be overcharged or over depleted.

Variant 1, although not used in the final simulations, takes advantage of Modelica's built-in library to model individual series modules as voltage sources dependent on the state of charge of the battery pack. This is the main advantage present in this first variant, as it models series modules separately and allows for varying characteristics in their behavior if necessary (eg; a series module is damaged). MSL4.0 cells are however, limited in scope and do not allow for state of charge dependent resistance. This can be seen by entering the cell model as shown in Figure 3.11 to observe the constant resistance values.

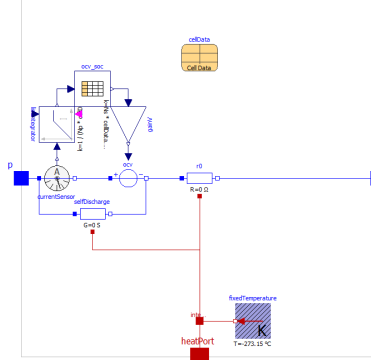


Figure 3.11 – MSL 4.0 Battery Cell

In order to implement an internal resistance capable of varying with the state of charge, the first variant of the battery pack lumps together the resistance of all the series modules together into one SOC dependent resistance. This is done by evaluating the internal resistance according to Equation 3.04, resulting from an algebraic manipulation of the rules of equivalent resistances for parallel and series circuits.

$$R_{eq,batt}(SOC) = \frac{N_s \cdot N_{sm}}{N_{pu}} \cdot R_{cell}(SOC) \quad \text{Eq 3.04}$$

Since the self-discharge characteristics of the battery play a negligible role in the battery performance during normal operation, and are more interesting when evaluating battery storage, the bypass conductor is set to have a null conductivity. As a consequence of both the internal resistance and self-discharge conductors having null values, no heat is generated by either component, whether the thermal port of the MSL4.0 cells be activated or disabled. Thus, for this version of the battery pack to consider thermal effects, a thermal port would need to be eventually connected to the equivalent resistor located outside the MSL4.0 cells and could be considered an upgrade and improvement to bring to the model further down the line.

To evaluate the SOC of the battery, MSL 4.0 cells use an integrator such as to implement Equation 3.05.

$$SOC(t) = SOC_0 - \int_{t_0}^t \frac{I_{batt}}{C_{batt}} dt \quad \text{Eq 3.05}$$

Another consequence of this battery model construction is the strict adherence to the number of series modules selected when creating the system. This is a result of each series module being modelled by the MSL4.0 cells, which can model parallel units and series cells internally, but do not account for the possibility of putting the resulting modules in series to create the larger voltages necessary in electric vehicles to improve efficiency.

This removes the possibility of easily modifying the number of series modules to study their effects, but as previously mentioned, allows the user to study effects of damaged series modules.

The number of cells in series and units in parallel can however be modified to create varying battery pack dimensions for various applications, as shown in Table 3.01. Naturally, in cases where the battery pack is the sole energy source, the battery must be bigger.

Application	Number of Series Modules [-]	Number of Parallel Units [-]	Number of Cells in Series [-]	Battery Size [kWh]
Hybrid	11	20	20	39.6
BEV	11	240	20	475.2

Table 3.01 – Battery Layouts

Lastly, as a consequence of using MSL4.0 cells with many features disabled, this model is inefficient from a numerical and simulation point of view, containing more equations and constraints than necessary by comparison to the subsequent variant (over six hundred equations). This naturally translates to longer simulation times.

Variant 2 improves on some of the downsides of its counterpart, namely the possibility to vary the number of series modules whilst facilitating the importation of data into the model. It also adds *optional* features to the user such as RC units and thermal modelling for more advanced modeling (if desired). The second variant expands upon the structure of the MSL4.0 cells, however now representing a whole battery pack. Thus, the open circuit voltage can be evaluated as the sum of all the cell voltages in series, resulting in Equation 3.06.

$$V_{OC,batt}(SOC) = N_s \cdot N_{sm} \cdot V_{OC,cell}(SOC) \quad \text{Eq 3.06}$$

The equivalent internal resistance of the entire battery pack is once again applied to a single resistor but is now also dependent on temperature. If the user chooses to disable this feature, the temperature remains constant, and battery pack internal resistance is once again evaluated as shown previously in Equation 3.04, else it is evaluated as shown in Equation 3.07.

$$R_{eq,batt}(SOC, T) = \frac{N_s \cdot N_{sm}}{N_{pu}} \cdot R_{cell}(SOC, T) \quad \text{Eq 3.07}$$

Once again, the bypass conductor is set to have a null conductance as the self-discharge characteristics of the battery are deemed to have a negligible effect on the simulation, though a value can be attributed if other studies are desired.

If the RC circuit is enabled, the short circuit path connecting the equivalent resistor to the negative battery pack terminal is turned into an open circuit, and the RC components are connected in parallel in place of the short circuit. This is represented on the graphical model via the dashed lines to symbolize the ability to activate this feature. Naturally, if the RC circuit is disabled, the RC elements are removed from the system and the short circuit replaces them. It should be stated that this feature was added for completions sake and was not used in the final simulations due to not having the data to run it.

If the thermal options are enabled, the two resistors and one conductor generate heat and vary the uniform temperature of the whole battery pack. A Modelica thermal capacitor is used to model the thermal capacitance of the battery pack, whilst a temperature sensor then feeds data to Equation 3.07 for ensuring the application of the correct resistance. Note that contrarily to conductors, the battery pack internal resistance decreases with increasing temperature.

The last practical upgrade brought on by the second variant is the possibility of loading OCV data directly from an external ASCII file, something not possible to be done with MSL4.0 cell components as they require the OCV data be present locally in the “cellData” parameters list. This allows the user to easily swap between cells in a battery pack to evaluate differences.

The disadvantages of the second variant are purely limited to their inability to evaluate the effects on the battery pack, and thus vehicle performance, if one cell were to be damaged. This is due to simulating the whole

battery pack as one giant unit rather than individual series modules. Naturally, this does however result in a reduced simulation time due to a system that is much smaller than its counterpart (about one hundred equations).

Fuel Cell

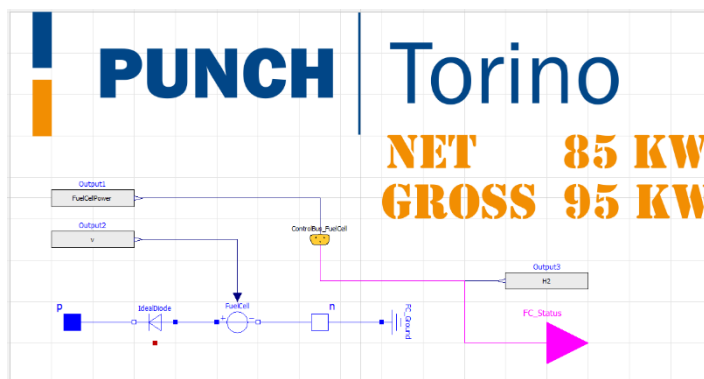


Figure 3.12 – Fuel Cell Model

The fuel cell, similarly, to the battery pack, is modeled by a Thevenin style circuit, with the negative terminal being grounded. Contrarily to the battery however, the fuel cell is not always connected to the system and must be able to be disabled if the energy management strategy decides to do so. For this reason, a Boolean parameter is imported from the data bus capable of switching on and off the whole system.

During the modeling, some simplifying assumptions were also made. As the goal was to evaluate the validity of OpenModelica as a tool, it was deemed appropriate to remove pressure effects and thermal effects from the model, under the hypothesis that these were kept constant by the control systems such as to ensure optimal performance. This was done after preliminary simulations in GT-SUITE showed thermal variations of the fuel cell model during operation were minimal. Furthermore, this model doesn't not simulate transient effects within the fuel cell and is assumed to be capable of supplying any power request by the user (within its limits), immediately. Further simplifications include the neglect of transient effects, with the fuel cell capable of supplying the power request immediately upon being requested. It is worth noting that the simplification related to the ability to supply immediate power wasn't performed in GT-SUITE and could be a source of variation down the line; however, with limited initial access to GT-SUITE, this was not immediately understood.

The fuel cell model is built based upon the GT-SUITE model, modeling a voltage source subject to various losses. Due to the fuel cells uni-directional nature (it can only discharge power and not absorb power), an ideal diode is included in the circuit. Thus, the controllable voltage source imposes Equation 3.08 onto the system.

$$v_{FC} = (v_{OC} - v_{act} - v_{mt} - v_{ic} - i \cdot R) \cdot n_s \quad \text{Eq 3.08}$$

...where R is the individual internal resistance of a single fuel cell, and not of the whole stack.

As both the open circuit voltage and the internal resistance are constants, and not dependent on an external data map, the calculation for the fuel cell voltage can be performed via text-based programming, taking account for the other losses such as the activation voltage loss, the mass transport voltage loss, and the internal voltage loss, the last of which is constant whilst the first two can be evaluated according to the expressions shown in Equation 3.09 and Equation 3.10.

$$v_{act} = \frac{R_{gas} \cdot T}{2 \cdot \alpha \cdot F} \cdot \ln \left(\frac{j}{j_0} \right) \quad \text{Eq 3.09}$$

$$v_{mt} = -C_{mt} \cdot \ln\left(1 - \frac{j}{j_1}\right) \quad \text{Eq 3.10}$$

The hydrogen, oxygen and air consumed, are calculated once again in the text-based programming via the first determining their consumption rates, which are proportional to the discharged current, and then integrating

said rates to evaluate the final consumption. This relationship can be observed via Equation 3.11 and Equation 3.12. Note it assumes a 21 % content of oxygen in the air at all points in time.

$$\dot{H}_2 = \frac{i \cdot n_s \cdot \mathcal{M}_{H_2}}{2 \cdot 1000 \cdot F} \quad \text{Eq 3.11}$$

$$\dot{O}_2 = \frac{1}{1.21} \cdot \frac{i \cdot n_s \cdot \mathcal{M}_{H_2}}{4 \cdot 1000 \cdot F} \quad \text{Eq 3.12}$$

The water production rate can then be calculated using a mass balance obtained via performing a mass balance obtained from the chemical equation relating to the combustion of hydrogen.

Balance of Plant

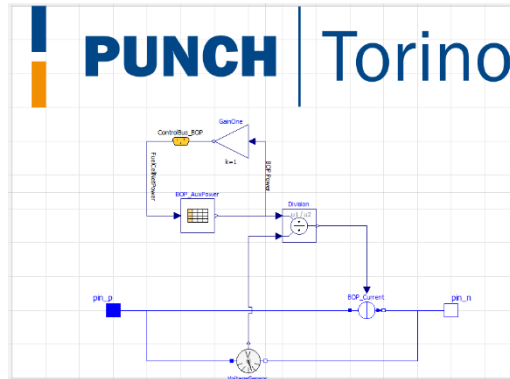


Figure 3.13 – Balance of Plant Model

The Balance of Plant model intends to model the consumption of the air compressor and auxiliaries typically associated to the operation of a fuel cell. This is done via a simple circuit wherein a user (the compressor) requests current from the fuel cell according to how much power is being produced by said fuel cell. Via interpolating a data map associated to the fuel cell, different current requests can be obtained for the various operation points.

Vehicle

The vehicle *package* was created to store a variety of vehicle *models* of varying complexities depending on the desired application. Two vehicle models were created for simulating basic longitudinal dynamics. The first model presented is sufficient and is the one that will be used for comparison with GT-SUITE. The second model removed some approximations for the sake of testing the software in non-linear conditions. In both, the following approximations were made:

- Rigid elements.
- No suspension.
- Left and right wheels travel at the same speed.
- No thermal modeling.
- Null vertical drag coefficient.

Simple Bus – Variant 1

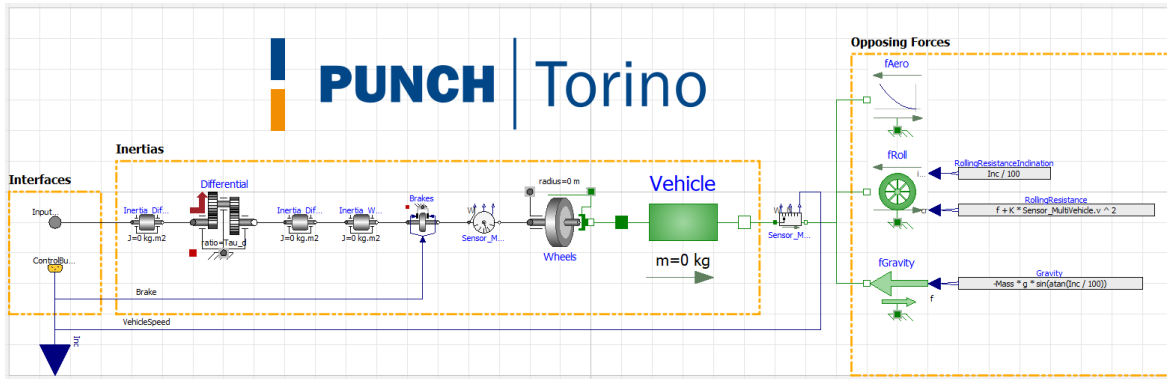


Figure 3.14 – Bus Longitudinal Dynamics, Variant 1

The first vehicle model represents the vehicle by a singular mass whose motion is opposed upon by three forces: aerodynamic drag, rolling resistance, and gravity in case of a slope. This opposing force can be summarized as shown in Equation 3.13.

$$F_t = m\ddot{x} + \frac{1}{2}\rho SC_x V^2 + mg \cdot \cos(\vartheta) \cdot (f + KV^2) + mg \cdot \sin(\vartheta) \quad \text{Eq 3.13}$$

This vehicle is propelled forwards via a transmission line, containing all the relevant inertias for the tires, wheels, semi-axis, etc. Note that the vehicle does not contain a gearbox and contains a gear ratio to model the differential with a constant gear ratio and efficiency. The presence of a brake element serves to slow the vehicle down when the signal line is acted upon by the driver. It is also worth noting that since there is no tire slip, all wheels and tires are lumped into a single unit and can be assumed to be travelling at the same rotational speed.

To reduce the number of equations and variables being simulated by the model, some constraints such as the resistive forces were applied via equation blocks. This required a technique wherein variables of interest such as the inclination profile were pulled out of the data bus and fed back into the model using an input interface. This is visible in Figure 3.14 as the connection between the dark blue triangle and the control bus. Its necessity is explained due to the inability for Modelica to utilize variables inside the control bus directly, as they do not truly exist within a model. Importing variables in this way allows the variable to be used in the text-based section of the modelling environment, and in the equation blocks seen on the right side of Figure 3.14. This is of stark contrast to GT-SUITE wherein it is possible to connect “receiveSignal” blocks directly to the system to be actuated. This technique of importing fictitious signals from the data bus into a variable to be used within individual models is also used in later models, most notable in motor control units and hybridization control units.

Simple Bus – Variant 2

The second model builds on the first by adding tire slip. Naturally to use this model, tire data is needed, something which may not always, and in this case, wasn't available. Nonetheless, as the purpose of the second variant is to test the software rather than obtain results from them, simulations can be carried out by loading generic passenger vehicle tire data from example data present on MATLAB's help page.

As tire slip is now modelled, the driven axle and the undriven axles speeds must be uncoupled, leading to the separation of the undriven wheels and tire inertias from the driven wheels and tires inertias. Naturally, the brakes are also separated, and the amount of torque they can produced is also reduced. Furthermore, to ensure the front axle always locks up before the rear axle, gains are present on the brake lines to obtain the desired braking ratio.

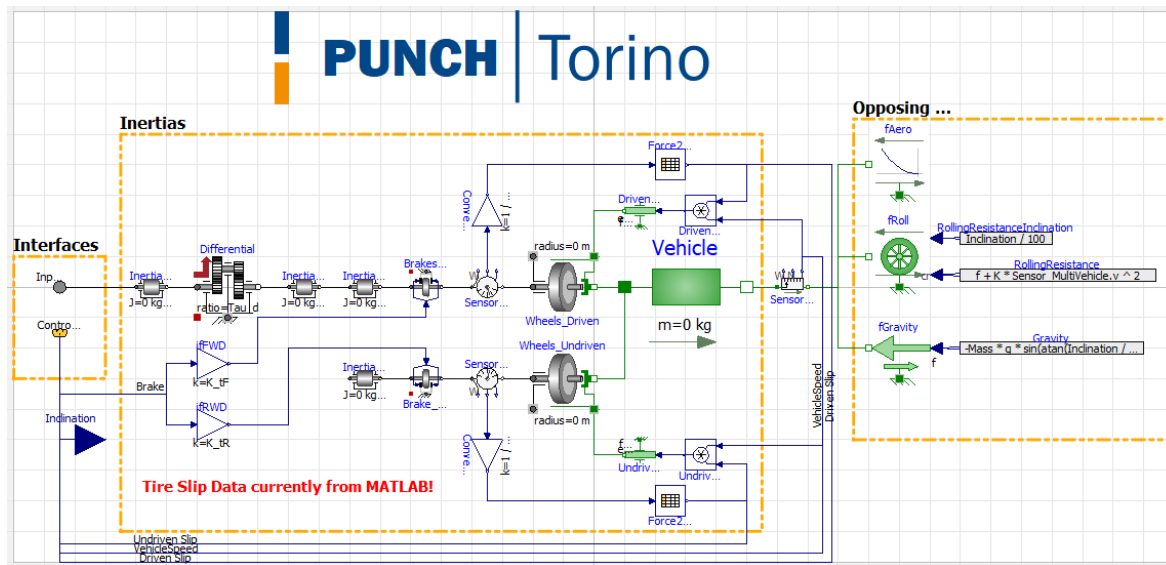


Figure 3.15 – Bus Longitudinal Dynamics, Variant 2

The tire slip is imposed upon the model via evaluating the torque (whether braking or traction) being sent to the wheel and converting this value into a force that needs to be balanced. Using tire data with slip vs longitudinal friction coefficient, it is then possible to determine the slip necessary to balance the driving/braking torque. This slip is imposed on the model at the wheel ground contact point, uncoupling the rotation speed of the inertias and the vehicle.

Torque Source

SMPM Motor & Inverter

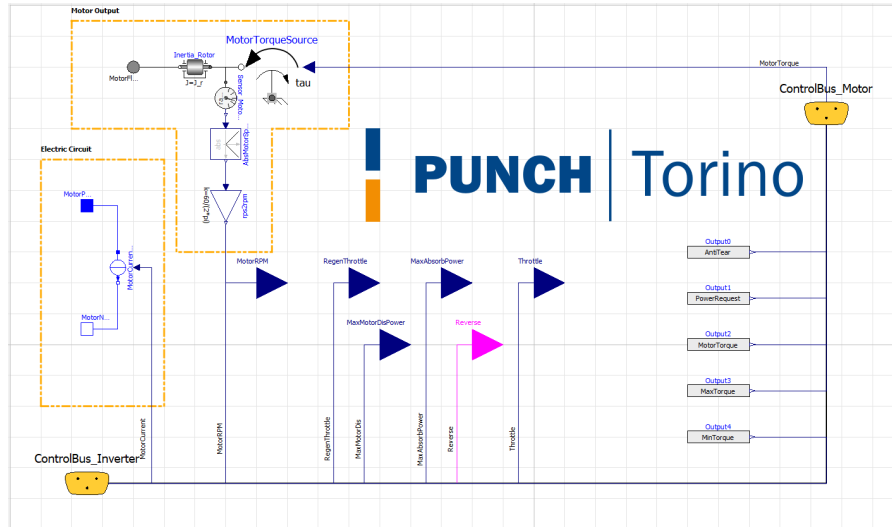


Figure 3.16 – Motor & Inverter Lumped Model

The model is characterized by one electrical circuit, and one physical circuit. The electrical circuit is used to control the current to be drained from the power source by the motor via a signal which modifies the current source. The physical circuit uses a signal-controlled torque source to accelerate the motor inertia, and anything connected to it down the line. The rotational speed sensor provides a reading for the motor's angular velocity. Unlike GT-SUITE, this motor model lacks the ability to impose both torque and speed on the motor, and its rotational velocity is calculated as a consequence of the downstream load.

For simplicity, and due to available data, the model shown is a map-based models which utilize torque request and motor speed to interpolate two 2D efficiency maps. It is also for this reason that the motor and inverter models have been lumped together into one model. Further data is also obtained from the data bus for the correct operation of the model, such as the throttle position, if the vehicle is in reverse, etc. This model also outputs some data to the data bus such as the maximum and minimum torque the motor can provide at a given instant for the current motor angular velocity. The use of map-based models means the model is approximative and interpolates every point as if the model were running at steady state.

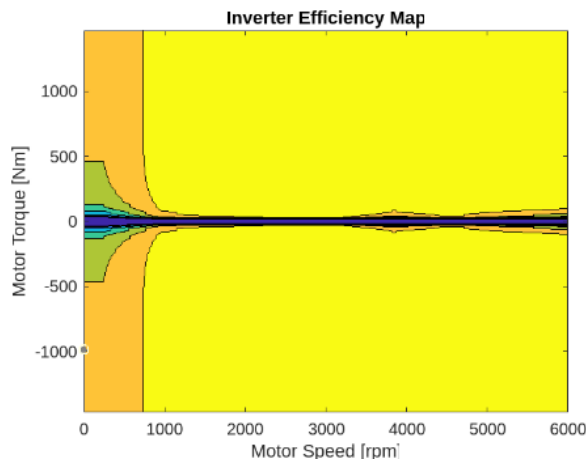


Figure 3.17 – Inverter Efficiency Map

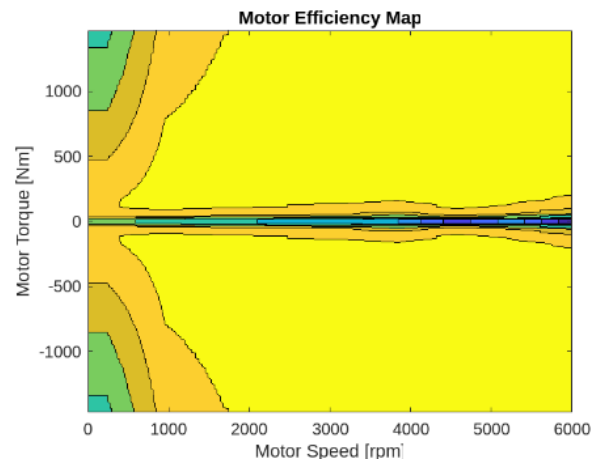


Figure 3.18 – Motor Efficiency Map

The efficiency maps inserted are those shown in Figure 3.17 and Figure 3.18. Naturally, not all the data points shown in Figure 3.17 and Figure 3.18 are valid, as the torque request must always be below the torque limit curve. Nonetheless, the whole map is used as an input as the torque limits are imposed afterwards. It is, however, worth noting that this model requires a preprocessing of the efficiency maps for it to run. In case the power source is not capable of satisfying the power request by the user, the torque output becomes limited. Using a MATLAB script, it is possible to convert the two efficiency maps into one where electrical power and motor rotational speed are the inputs, and motor torque is the output. This is an inherent dis-advantage with respect to GT that does not require such pre-processing of the motor maps. Furthermore, as Modelica doesn't accept "NaN" as a valid input, the white space of the map where the user isn't allowed to operate must be replaced with real numbers ideally in such a way as to not cause a change in the map interpolation and vehicle behavior along the map extremities. Consequently, the whole map cannot be exploited and thus only 98 % of the electrical power is ever provided at the input of such a map, to avoid interpolation errors. Note that the resulting map contains bubbles or pockets on non-linear behavior. These are errors that occur during the generation of such a map using the algorithm depicted in ANNEX B. Furthermore, it is worth also noting that not all the points on the map are valid, similarly to the torque curves.

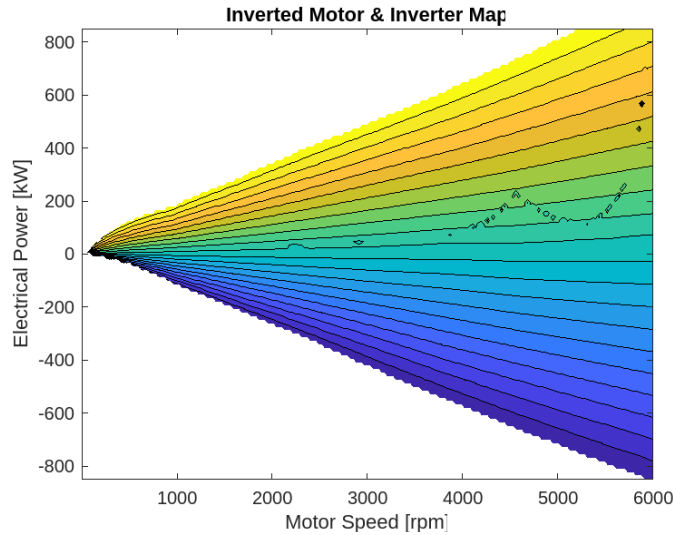


Figure 3.19 – Inverted Motor & Inverter Map

With these three data maps, the logic governing this lumped model begins as follows; if battery limitations are not met, throttle signals provided by the driver are converted into torque requests according to Equation 3.14.

$$T_{mot} = h_t \cdot T_{max}(\omega) \quad \text{Eq 3.14}$$

The knowledge of the motor rotational speed via the rotational speed sensor and the torque request allows for the interpolation of the efficiency maps. Using the efficiency points of both the electrical motor and the inverter, the electrical power request can be evaluated via Equation 3.15.

$$P_{mot,el} = (\eta_{inv}\eta_m)^k \left(T_{mot} \cdot \frac{60}{2\pi} \cdot \omega \right) \quad \text{Eq 3.15}$$

...where k is an exponent equal to one in regen conditions and equal to minus one in traction conditions.

If battery limitations are met, the MATLAB generated map, shown in Figure 3.19, is used. In such a case, the torque the motor applies is evaluated using the battery power limitations and the rotational speed of the motor as inputs. Once the correct values of torque and power have been determined, the signal-controlled current source and signal-controlled torque source impose their values, obtained from calculations done in either the motor control unit or the hybridization control unit.



Note that if the motor is used in a hybrid model, the battery limitations get altered, as the maximum power that can be fed to the motor is no longer limited by the battery discharge limits minus the auxiliary consumptions, but by the battery discharge limits minus the auxiliary consumptions plus the extra power added to the circuit by the fuel cell after the step-up DC/DC converter. Likewise, the maximum electrical power the system can absorb is reduced for analogous reasons. These statements can be summarized into Equation 3.16 and Equation 3.17.

$$P_{el,max} = P_{batt,max} - P_{aux} + P_{boost,o} \quad \text{Eq 3.16}$$

$$P_{el,min} = P_{batt,min} - P_{aux} + P_{boost,o} \quad \text{Eq 3.17}$$

...where both the auxiliary power consumption and output boost power are both defined with positive values.

Electronics

Simple Boost

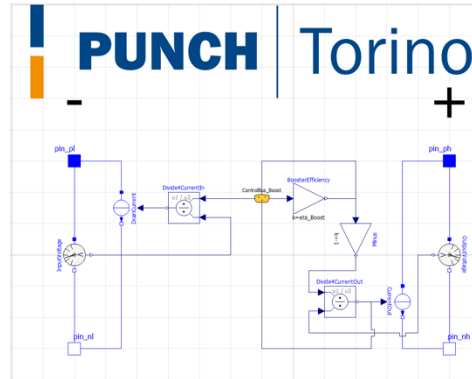


Figure 3.20 – Simple Boost Model

The model for the DC/DC step-up is performed assuming a constant efficiency. It is modeled using two controllable current sources, and two voltage sensors. The current source attached to the lower voltage circuit (left) acts like a user, requesting current such as to match the net power request calculated by the hybridization unit. The current source connected to the high voltage circuit (right) supplies a current such as to produce a power equivalent to the net fuel cell power request multiplied by the boost efficiency, such as to respect Equation 3.18.

$$P_{HV} = \eta_{DC/DC} \cdot P_{LV} \quad \text{Eq 3.18}$$

Data such as the power out is also fed back into the data bus for calculations necessary in other models, such as the maximum power available to the motor.

Constant Auxiliaries

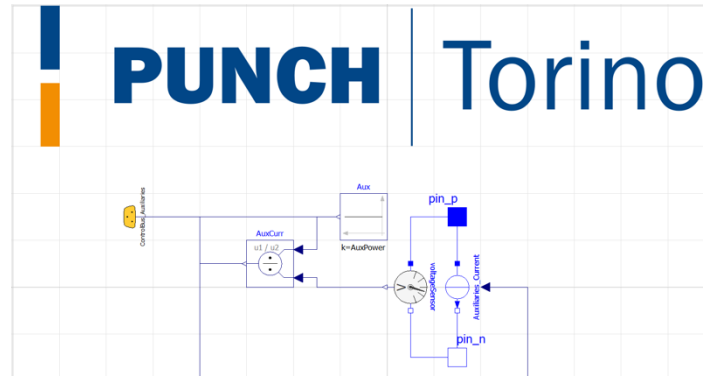


Figure 3.21 – Constant Auxiliaries Model

The constant power auxiliaries' model is a model aimed to represent the 12 V circuit and auxiliaries present on the bus, such as the doors, lighting and air conditioning unit usually connected to the BEV battery via a DC/DC converter. Here, they are modelled via a controllable current source, which requests a current such as to ensure the produce of the current produced and the voltage difference across the positive and negative pins is equal to the auxiliary power request. The auxiliary power request is also fed to the data bus for calculations performed in other models.

Motor Control Unit

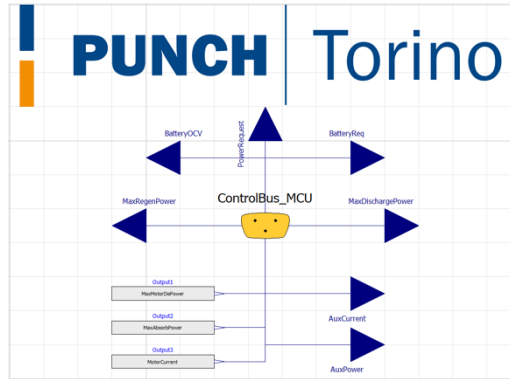


Figure 3.22 – Motor Control Unit Model

The motor control unit is a model created to ensure the electrical motor model shown previously works adequately for both BEV and FCH applications. This is done via ensuring that the motor model is fed with the correct information from the data bus regarding what current it should be discharging, and what power it can be requested from the onboard energy sources.

By comparing the power request associated to the drivers throttle pedal position with what the power the on-board energy sources can provide, the Motor Control Unit model can decide if the motor is being limited by constraints further upstream of the motor itself. If this is the case, the torque that the motor can supply is re-evaluated using the map shown in Figure 3.18, as previously discussed.

The power associated to said torque is converted into a current request for each component by first determining the battery current using Equation 3.19. This battery current can then be converted into a current request for each node via considering Equation 3.20.

$$I_{batt} = \frac{v_{OC}(SOC) - \sqrt{v_{OC}^2(SOC) - 4 \cdot (P_{mot} + P_{aux}) \cdot R_{eq}(SOC)}}{2 \cdot R_{eq}(SOC)} \quad \text{Eq 3.19}$$

$$I_{mot} = I_{batt} - I_{aux} \quad \text{Eq 3.20}$$

The derivation of Equation 3.19 can be found in ANNEX A.

Hybrid Control Unit

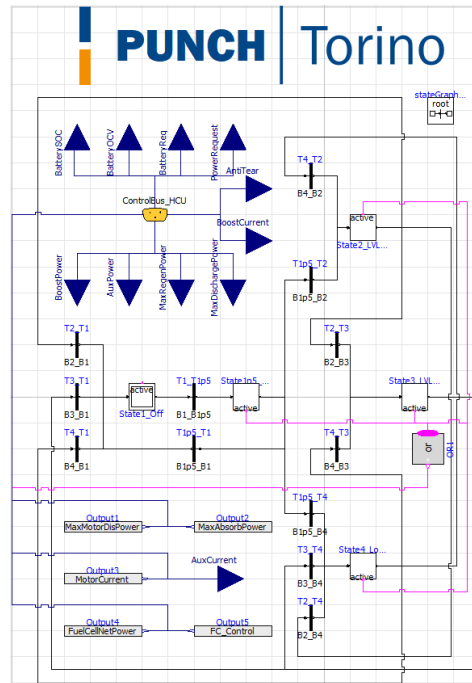


Figure 3.23 – State Graph Hybrid Control Unit Model

The hybridization control unit here developed aims to control a series hybrid electric vehicle with an onboard fuel cell and high voltage battery. It models a rule based controlled using MSL's "State Graph" library which operates similarly to MATLAB/Simulinks Stateflow. From Figure 3.23, it is possible to note five different system states represented by the five white squares. Transitioning to and from the five states is controlled by the thick black elements which act as Boolean controlled gates. Whilst the Booleans of the gates may be true or false at any time, the gates are only "open" if the gate is "enabled", a condition which is only met if the active state is one preceding the gate. Furthermore, it is worth noting that only one state can be active at a time, as each state aims to control the fuel cell and battery systems power split and are thus mutually exclusive.

In order to ensure optimal fuel cell and battery operation, the system aims to keep the battery charged just over 50 %, and thus defines an upper SOC limit and a lower SOC limit to act upon.

State 1: Fuel Cell Off. When the fuel cell is off, the system acts like exactly like the motor control unit described before, with two extra signals being generated for the control bus: a Boolean declaring the fuel cell to be off, and a scalar declaring the fuel cell net power to be null. Thus, the equations governing this state are still Equation 3.19, and Equation 3.20. As visible in Figure 3.23, this state can be entered from any other state, provided the gates leading to it are enabled and their condition Booleans are true, all of which require the battery to have reached the upper SOC limit (plus a minor anti hysteresis value of 2 %). This state may be left if the total power requested by the system surpasses the battery maximum power output, or if the battery SOC decreases below the upper SOC limit. Furthermore, if the fuel cell is in this state, it should ideally remain in this state for a minimum duration of time, to ensure the Hybridization unit isn't constantly asking it to switch on and off. In either case, the Fuel Cell is forced into the only possible next state.

State 1.5: Fuel Cell Warm-Up. Once the necessity for the fuel cell to be turned on has been confirmed, the fuel cell net power request is slowly increased according to Equation 3.21. This stage of the hybridization unit aims to model the transients of the fuel cell when switching on. Note the fuel cell must remain in this state for a minimum amount of time referred to as the fuel cell start-up time.

$$P_{FC,net} = 0.15 \cdot \frac{t}{\tau_{su}} \cdot P_{FC,max,net} \quad \text{Eq 3.21}$$

The Hybridization Unit must consequently adapt the calculations being fed to the motor and auxiliaries to account for the new power source present in the circuit, thus having a battery current conforming to Equation 3.22.

$$I_{batt} = \frac{v_{oc}(SOC) - \sqrt{v_{oc}^2(SOC) - 4 \cdot (P_{mot} + P_{aux} - P_{boost}) \cdot R_{eq}(SOC)}}{2 \cdot R_{eq}(SOC)} \quad \text{Eq 3.22}$$

...where P_{boost} corresponds to the power at the output of the step-up DC/DC converter.

Furthermore, the equations governing the signal-control sources for the motor and the auxiliaries must also be adapted to become those shown in Equation 3.22 and Equation 3.23.

$$I_{mot} = I_{batt} - I_{aux} + I_{boost,o} \quad \text{Eq 3.23}$$

From the warm-up state, the fuel cell may only enter state one, two or four, according to the conditions met. It may return to the off state if the battery SOC has surpassed the upper limit (plus the anti-hysteresis threshold). It may transition to state two if the battery SOC is below the upper limit, and it may transition to state four if the total power request surpasses the power capable of being supplied by the battery system by itself.

State 2: Charge Level 01. If charging level one is entered, the Fuel Cell produces a power according to an externally loaded data map. This can be represented via Equation 3.24.

$$P_{FC,net} = P_{LVL01}(SOC) \quad \text{Eq 3.24}$$

From here, a new timer is generated keeping track of how long the fuel cell has been on for. This is relevant for the same logic governing the fuel cell minimum off time: to avoid unnecessary transients. Thus, the fuel cell may actively transition to states three or four at any time but may only switch back to state one if the minimum on time has been satisfied. The equations governing the currents being requested by the system are the same as in the warm-up state and are once again Equation 3.22 and Equation 3.23. The system will transition to state three if the lower SOC limit is reached, or transition to state four if the total system power request surpasses the capabilities power capabilities of the battery pack.

State 3: Charge Level 02. This state corresponds to a more intense variant of state two, with a governing logic identical to that found in state two, simply using a different P_{FC} vs. SOC data map, as evidenced by Equation 3.25. It should however be noted that this state can only be entered from states two and four, and only allows transitioning back to state one or four. Once again, this state contributes to the minimum on time of the fuel cell in case the system wishes to switch it off.

$$P_{FC,net} = P_{LVL02}(SOC) \quad \text{Eq 3.25}$$

State 4: Load Follower. The final state the system can be in corresponds to the fuel cell providing a mean power to the system, whilst also providing any excess power that may be needed to satisfy the power request. This can be formalized mathematically into Equation 3.26.

$$P_{FC,net} = \max\left(\frac{P_{LVL01}(SOC) + P_{LVL02}(SOC)}{2}, \frac{P_{req.} - P_{batt,max}}{\eta_{DC/DC}}\right) \quad \text{Eq 3.26}$$

Naturally, the equations governing the currents in this state are once again those described earlier by Equation 3.22 and Equation 3.23. This state also contributed to the minimum on time of the fuel cell in case the hybridization unit wishes to switch off the fuel cell. From this state, the system can transition to all other states except for the warm-up state.

It is worth noting that this model does have an issue, wherein the total time the fuel cell has been on isn't properly kept track of. Instead, the model tracks time spent in each state, whereas it should accumulate the time spent in State 1.5, State 2, State 3 and State 4. This is, however, an acceptable flaw with the model, since the

fuel cell stays on once started in all simulations, even with the minimum on time feature removed. This approximation is necessary in the model to prevent issues with Modelicas' StateGraph library, wherein if the user keeps track of the total fuel cell on time externally to the StateGraph, the Booleans controlling the transitions may be by-passed altogether. It was found, for example, in certain attempts, that State 2 could be entered directly from State 1 despite the existence of the intermediary State 1.5 between them.

Further issues with this model include the inability for the user to select the starting state, as it is defined by the models' construction with the StateGraph blocks. This is particularly crucial in a situation such as testing the vehicles maximum acceleration, where the fuel cell should be operating at full power but can't, due to the switching on time. Naturally, this can be countered by removing the minimum on time and start-up time for these simulations, however it isn't ideal. To solve this, different SG RBC models with different initial states could be made.

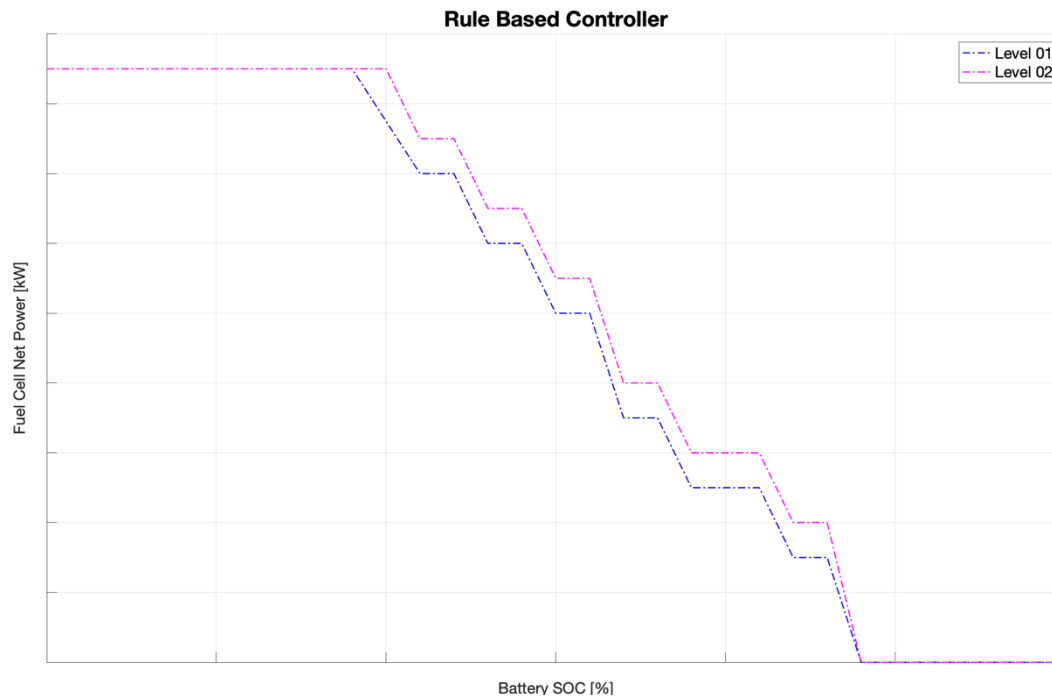


Figure 3.24 – RBC Maps for Charge Level One and Two

Note: The axes of Figure 3.24 have been obscured for confidentiality reasons.

Icons

The Icons *package* was made primarily to contain the *partial models* associated to the graphical icons of all the models created. In this way, if one were to modify the icon of say, the battery model, all battery models icons using the *partial model* would be modified accordingly.

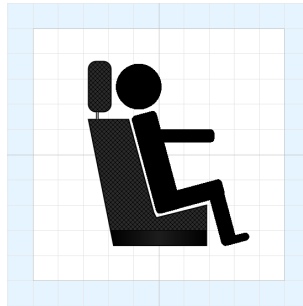


Figure 3.25 – Driver Icon

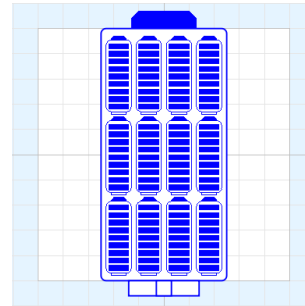


Figure 3.26 – Battery Icon

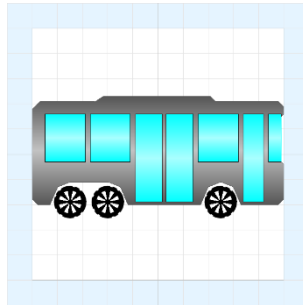


Figure 3.27 – Bus Icon

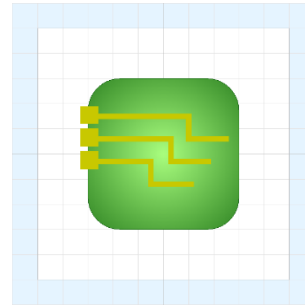


Figure 3.28 – Control Unit Icon

#4 Model Overview & Results

BEV Model

BEV Model Assembly

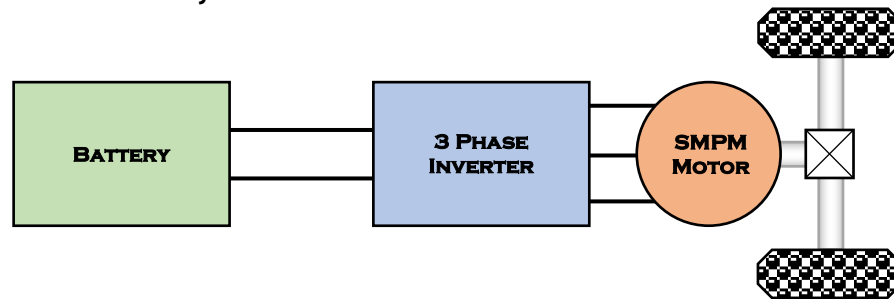


Figure 4.01 – BEV Powertrain Configuration

The battery electric vehicle powertrain model is assembled according to the schematic seen in Figure 4.01, with illustrates the standard two-wheel drive BEV architecture, without a gearbox. This is translated to the Modelica graphical programming environment according to the Figure 4.02, with a few extra elements to control the mechanism, such as the driver, the speed cycle, and the control unit. It is worth noting that in the case of testing top speed/acceleration curves, the reference speed cycle can be replaced with generic signals to force the driver to keep the throttle pedal constantly fully actuated.

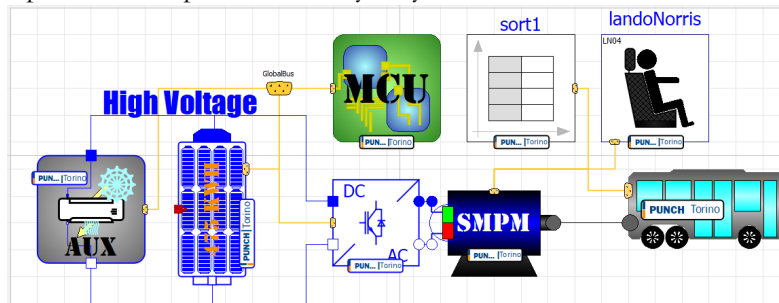


Figure 4.02 – OpenModelica BEV Model

Figure 4.02 illustrates the three connections present on the model, a torque connection from the motor to the bus, electrical connections across the battery, auxiliaries, and inverter, and finally a control bus connection across all elements for sharing data amongst the components. It is worth re-stating that despite the auxiliaries being connected to the high voltage battery, they are not directly fed by it, and should theoretically comprise also of a DC/DC converter. Nonetheless, for the sake of determining the energy consumptions, this set-up is equivalent. It is worth noting that no ground is visible on this model, as the negative pin of the battery is defined as the ground within the battery model.

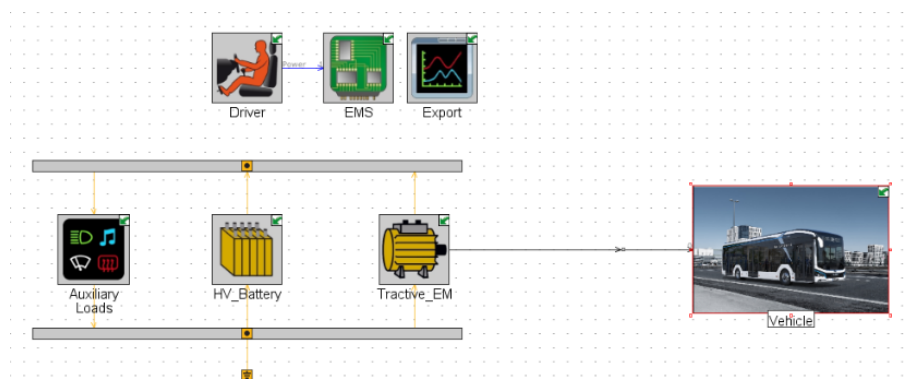


Figure 4.03 – GT-SUITE BEV Model

Driver Calibration

To calibrate the driver, some target parameters had to be defined to clearly define whether one calibration was superior to another. These quantities include those described in Chapter 3 during the driver model creation, as well as a tolerance margin of two kilometers per hour above and below to the target velocity. For a calibration to be deemed successful, it had to remain within the tolerance margins, ideally perfectly tracing the reference profile, whilst minimizing the three errors measured by the driver model. A calibration was deemed to be superior to another if at least two of the measured error quantities being tracked were reduced. Naturally this calibration aims to improve the driver accuracy as much as possible, and not necessarily minimize consumption. It was however, often observed that some calibrations would have significant energy consumption improvements.

BEV Model Results

Speed Cycle	Mass [kg]	SOC ₀ [%]	SOC _f [%]	ΔSOC [%]	Battery Energy [kWh]	Consumption [kWh/km]
SORT 1 Dist = 14.50 km t _r = 933.2 s	16 000	80	79.06	0.94	4.446	1.434
		50	48.78	1.22	5.803	1.871
		25	23.33	1.67	7.957	2.564
	19 000	80	78.93	1.07	5.078	1.639
		50	48.58	1.42	6.736	2.173
		25	23.05	1.95	9.259	2.996
SORT 2 Dist = 14.05 km t _r = 740 s	16 000	80	79.09	0.91	4.348	1.183
		50	48.80	1.20	5.703	1.551
		25	23.34	1.66	7.883	2.147
	19 000	80	78.95	1.05	4.970	1.353
		50	48.60	1.40	6.673	1.816
		25	23.07	1.93	9.189	2.522
SORT 3 Dist = 6.17 km t _r = 414.2 s	16 000	80	79.36	0.64	3.059	1.056
		50	49.15	0.85	4.046	1.397
		25	23.81	1.19	5.639	1.952
	19 000	80	79.26	0.74	3.520	1.216
		50	48.98	1.01	4.823	1.667
		25	23.60	1.40	6.646	2.320
Braunschweig Dist = 10.87 km t _r = 1740 s	16 000	80	77.33	2.67	12.677	1.168
		50	46.39	3.61	17.169	1.583
	19 000	80	76.84	3.16	15.002	1.388
		50	45.70	4.30	20.428	1.899
Old MLTB Dist = 9.00 km t _r = 2281 s	16 000	80	77.53	2.47	11.748	1.310
		50	46.75	3.25	15.427	1.720
	19 000	80	77.15	2.85	13.523	1.508
		50	46.21	3.79	18.012	2.009
RDC #1 Dist = 16.59 km t _r = 2875 s	16 000	80	75.17	4.83	22.945	1.386
		50	43.58	6.42	30.517	1.848
	19 000	80	74.34	5.56	26.884	1.629
		50	42.47	7.53	35.802	2.189
RDC #1 (x3) Dist: 49.780 km t _r = 8625.5 s	16 000	80	64.98	15.02	71.377	1.437
		50	29.38	20.62	97.986	1.983
	19 000	80	62.30	17.70	84.104	1.700
		50	25.65	24.35	115.72	2.376

Table 4.01 – Raw BEV Results – 475.2 kWh Battery, P_{aux} = 7 kW

Note: The following results were obtained with the bus auxiliaries draining 7 kW of power constantly, and are thus affected by the simulation duration, since the “idling” of the vehicle continues to consume a considerable amount of energy. Furthermore, it is interesting to highlight the effects of the variable internal resistance and OCV of the battery highlighted via the varying consumption rates of the RDC #1 and RDC #1 (x3) cycles.

From the raw results, it is easy to observe a degradation in system performance as the battery becomes progressively more depleted, with energy consumptions nearly doubling when the battery is at a low level of charge. Whilst these energy consumptions are in line with expectations for a modern electric bus, it is worth highlighting that the several simplifications made, such as the lack of RC circuits in the battery model, lead the results to indicate a lower energy consumption than one would expect from the final vehicle.

Simulation		GT-SUITE Batt. Energy [kWh]	OpenModelica Batt. Energy [kWh]	% Variation
Speed Cycle	Mass	SOC ₀ = 80 %	SOC ₀ = 80 %	SOC ₀ = 80 %
SORT 1 Dist = 14.50 km t _r = 933.2 s	16 000	4.182	4.446	06.33 %
	19 000	4.737	5.079	07.21 %
SORT 2 Dist = 14.05 km t _r = 740 s	16 000	4.067	4.348	06.90 %
	19 000	4.541	4.970	09.43 %
SORT 3 Dist = 6.17 km t _r = 414.2 s	16 000	2.819	3.059	08.50 %
	19 000	3.182	3.520	10.63 %
Braunschweig Dist = 10.87 km t _r = 1740 s	16 000	12.325	12.677	02.86 %
	19 000	14.270	15.002	05.13 %
Old MLTB Dist = 9.00 km t _r = 2281 s	16 000	11.359	11.748	03.42 %
	19 000	12.822	13.523	05.47 %
RDC #1 Dist = 16.59 km t _r = 2875 s	16 000	22.203	22.945	03.34 %
	19 000	25.1723	26.884	06.80 %
RDC #1 (x3) Dist: 49.780 km t _r = 8625.5 s	16 000	68.887	71.377	03.61 %
	19 000	75.009	84.104	12.12 %

Table 4.02 – Comparison of GT-SUITE & OpenModelica BEV Results ($P_{aux} = 7$ kW)

Upon comparing the results obtained in OpenModelica and GT-SUITE, as shown in Figure 4.02, it is easy to conclude that OpenModelica's BEV model is consuming more energy than the GT-SUITE BEV model. The percentage difference obtained does however paint an interesting picture, in that the average error in the real cycles (Braunschweig, MLTB and RDC #1) is lower than the average error in the artificial test cycles (SORT1, SORT2 and SORT3). Although it can be argued that due to the lower energy consumption of the smaller cycles, smaller absolute differences in energy consumption cause larger relative errors, the non-linear nature of the battery internal resistance can be expected to amplify errors for longer and more intense cycle. The difference can therefore not be attributed solely to varying component behavior. The sole exception to this rule is the RDC #1 (x3) cycle wherein the non-linearity of the battery causes the error to grow as the test duration increases.

It is possible to plot and analyze the speed profiles (reported in their entirety in ANNEX C) to determine the causes for such differences. Taking an example from an artificial cycle such as the one reported in Figure 4.04, it is possible to observe the varying controller behavior. This can be explained in two parts; the Modelica controller created is very simple; utilizing a PIDF that can lead to oscillations such as those seen in the speed trace, regardless of the filtering effect. Furthermore, the Simple Driver model created for Modelica lacks foresight, whilst the GT-SUITE driver is allowed to see up to five seconds into the future to see what the target speed will be. This naturally leads to the Simple Driver model having a rougher speed profile, but one

that remains truer to the target reference speed. This is evidenced by the fact that the GT Driver overshoots the braking moments and has a larger setting time. In fact, the GT driver is often on the borders of the tolerance regions.

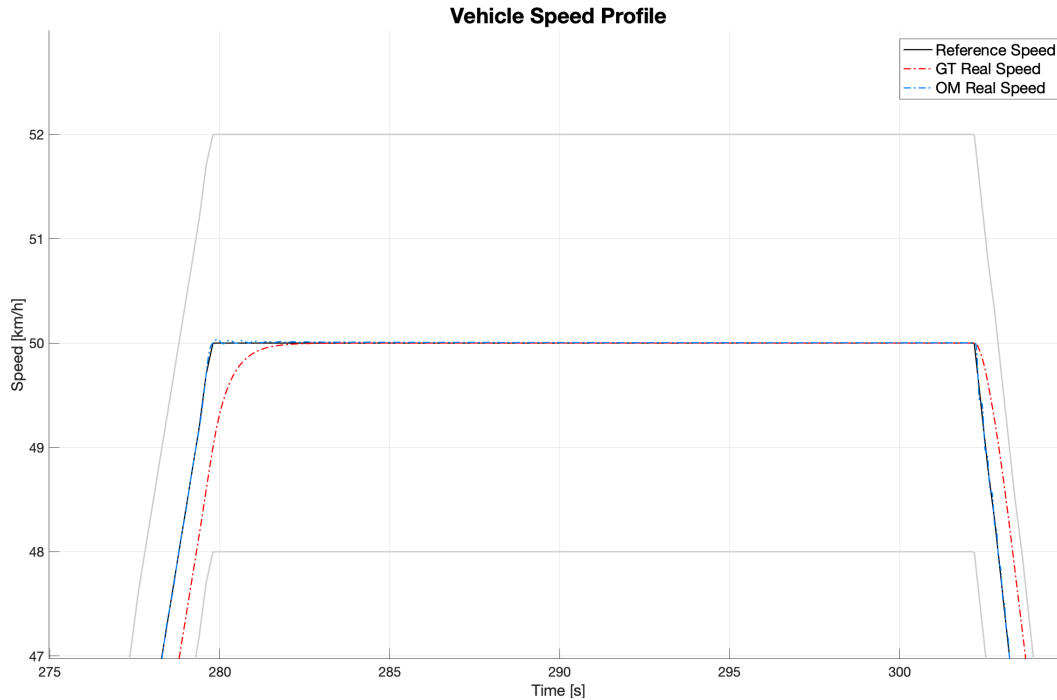


Figure 4.04 – Zoom on Comparison of BEV Speed Profiles
(SORT3, 16 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW)

It is possible to take the GT Drivers Speed trace and analyze similarly to the errors computed by the Modelica driver to thus determine if the GT Driver travels shorter or longer distances, at higher or lower average speeds, etc, to help identify further causes for the differences in the results. As the error tracking isn't performed on the GT models, this analysis can be performed on MATLAB to obtain Table 4.03.

The consistently lower Error 1 and Error 2 values presented in Table 4.03 indicate that, as Figure 4.04 seems to indicate, the Modelica built driver remains significantly closer to the target speed profile throughout the whole simulation, regardless of the cycle performed. It is however interesting to note that, on the artificial cycles, whilst the Modelica driver deteriorates when the vehicle load increases, indicating a worse performance of the system, GT-Suite's driver tends to improve. Though individually these effects are minor, they undoubtedly contribute to the larger relative consumption differences seen as mass increases within Table 4.02 for the artificial cycles. This as opposed to real cycles wherein both driver performances deteriorate and thus the relative energy consumptions don't drastically change and is instead lower than on the artificial cycles.

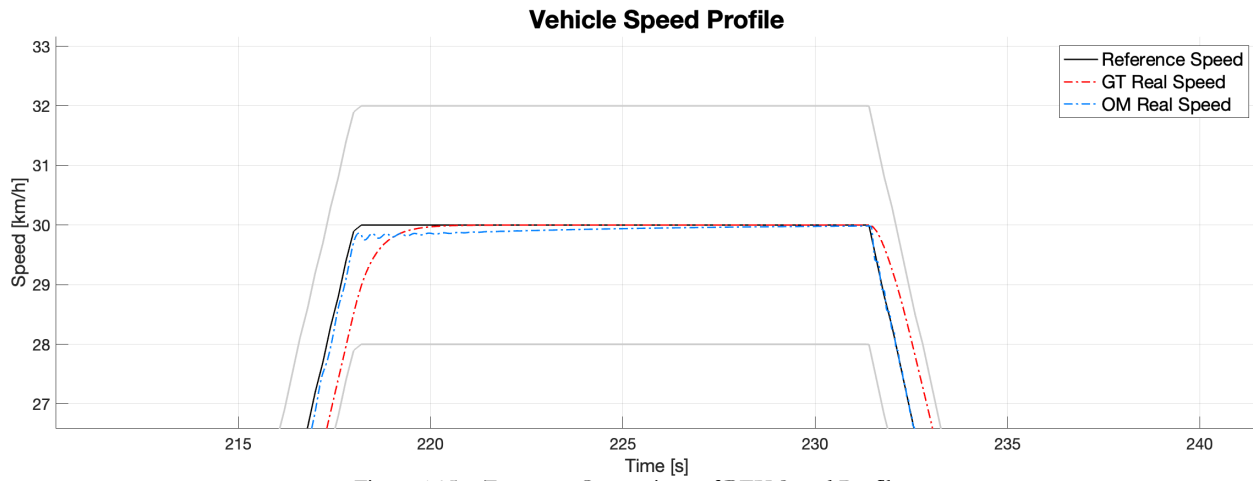


Figure 4.05 – Zoom on Comparison of BEV Speed Profiles
(SORT3, 16 000 kg, SOC₀ = 80 % & P_{aux} = 7 kW)

Simulation		GT-Suite Driver Tracking Errors			OpenModelica Driver Tracking Errors		
Speed Cycle	Mass	Error 1 [m]	Error 2 [m/s]	Error 3 [m]	Error 1 [m]	Error 2 [m/s]	Error 3 [m]
SORT 1 Dist = 14.50 km t _r = 933.2 s	16 000	153.5	0.164	7.180	18.7	0.020	16.8
	19 000	151.2	0.162	5.753	21.3	0.023	19.1
SORT 2 Dist = 14.05 km t _r = 740 s	16 000	124.6	0.168	5.135	9.8	0.013	8.31
	19 000	122.9	0.166	4.157	11.7	0.016	10.1
SORT 3 Dist = 6.17 km t _r = 414.2 s	16 000	79.4	0.192	4.119	3.4	0.008	2.35
	19 000	78.3	0.189	3.497	4.6	0.011	3.63
Braunschweig Dist = 10.87 km t _r = 1740 s	16 000	382.3	0.220	6.756	27.4	0.016	21.5
	19 000	390.1	0.224	38.40	77.4	0.044	65.7
Old MLTB Dist = 9.00 km t _r = 2281 s	16 000	454.5	0.199	77.65	3.72	0.002	0.08
	19 000	448.8	0.197	74.11	5.45	0.002	1.32
RDC #1 Dist = 16.59 km t _r = 2875 s	16 000	477.7	0.166	4.911	47.5	0.017	40.2
	19 000	489.6	0.170	14.18	102.0	0.035	90.6
RDC #1 (x3) Dist: 49.780 km t _r = 8625.5 s	16 000	1454	0.169	164.5	133.6	0.015	110.6
	19 000	1502	0.174	190.1	333.5	0.038	295.0

Table 4.03 – GT-SUITE & OpenModelica BEV Error Trackers (SOC₀ = 80 %, P_{aux} = 7 kW)

Whilst the first two errors measure how far the driver is, on average, from the target profile, the third tracking error measures how far the vehicle's total distance error is. Interestingly, despite the Modelica driver being closer on average to the desired speed profile, it often falls short of the target distance by a larger amount than the GT driver. This can be hypothesized as being caused by instances such as those shown in Figure 4.05, where the PID fails to reach the target velocity in an adequate time frame, thus struggling in acceleration, but performs exceptionally well in braking. This can be justified due to the driver having more torque available from the combined effect of the brake pedal and motor during deceleration, than acceleration. Naturally this can also be a source of difference between the Modelica model and the GT model, wherein the harsher braking of the Modelica models lead to more friction being applied than in the GT model.

The analysis carried out on Figure 4.04 depicting the vehicle on SORT 3 can be extended onto any other cycle, such as in Figure 4.06 depicting a zoom of the vehicle speed profile on the Braunschweig cycle. Here once again, the Modelica driver sticks closer on average to the target speed profile than the slower reacting GT model.

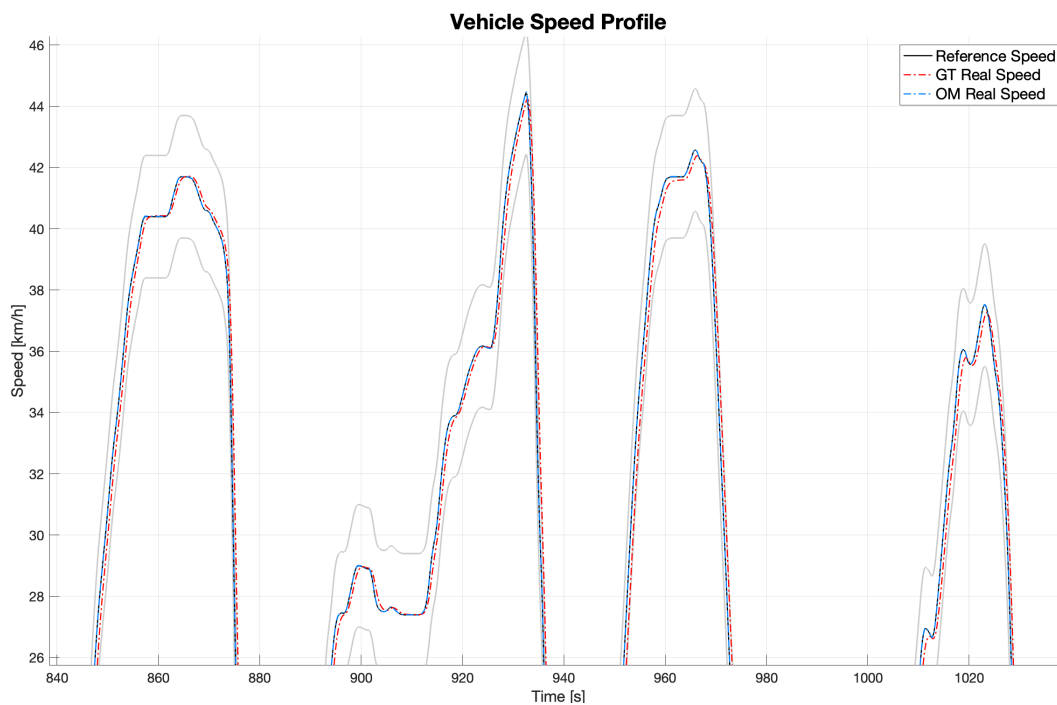


Figure 4.06 – Zoom on Comparison of BEV Speed Profiles
(Braunschweig, 16 000 kg, SOC₀ = 80 % & P_{aux} = 7 kW)

As already touched upon though before, the Modelica driver does struggle at times, and this can be further highlighted when observing the RDC #1 x3 cycle which aims to stress test the vehicle. When observing Figure 4.07, both Modelica and GT drivers are unable to reach the target velocity, with the Modelica driver falling short by roughly 3 km/h, whilst the GT driver falls short by 1 km/h. This occurs both due to faster depletion of the battery on the Modelica model than the GT model, and due to the Modelica motor map the aforementioned interpolation issues when the model becomes power limited. As the difference in battery state of charge is augmented, the maximum power output of both batteries becomes significantly different. Thus, in a high load demand such as climbing a steep hill, shown by the combination of Figure 4.07 and Figure 4.08, both vehicles achieve a different top speed.

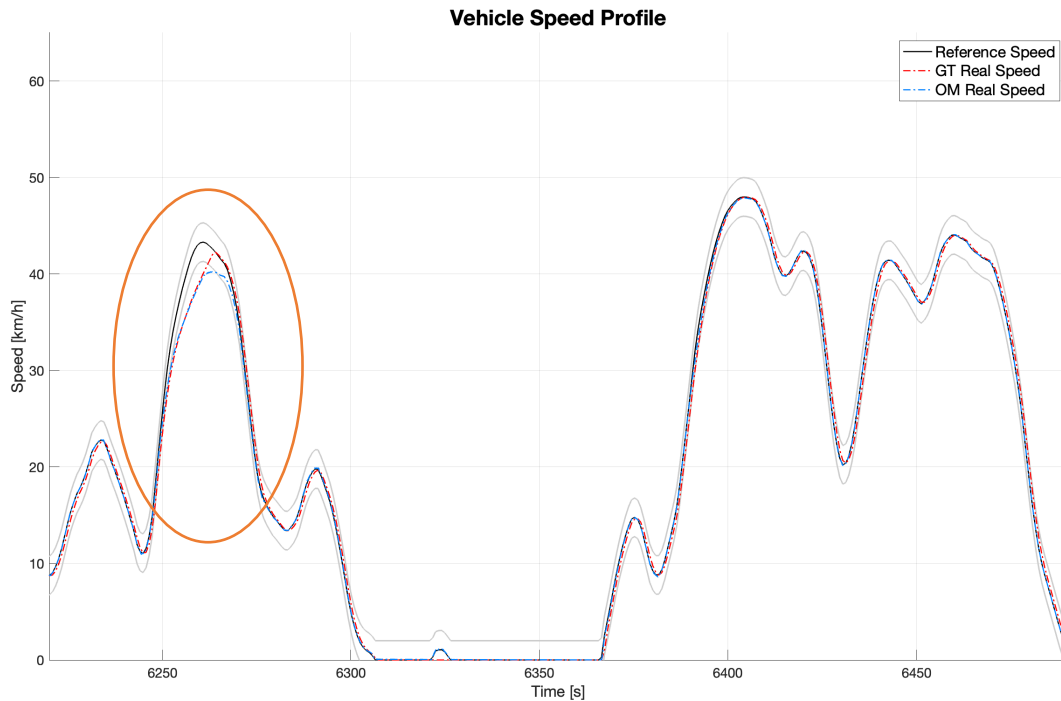


Figure 4.07 – Zoom on Comparison of BEV Speed Profiles
(RDC #1 x3, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7\text{ kW}$)

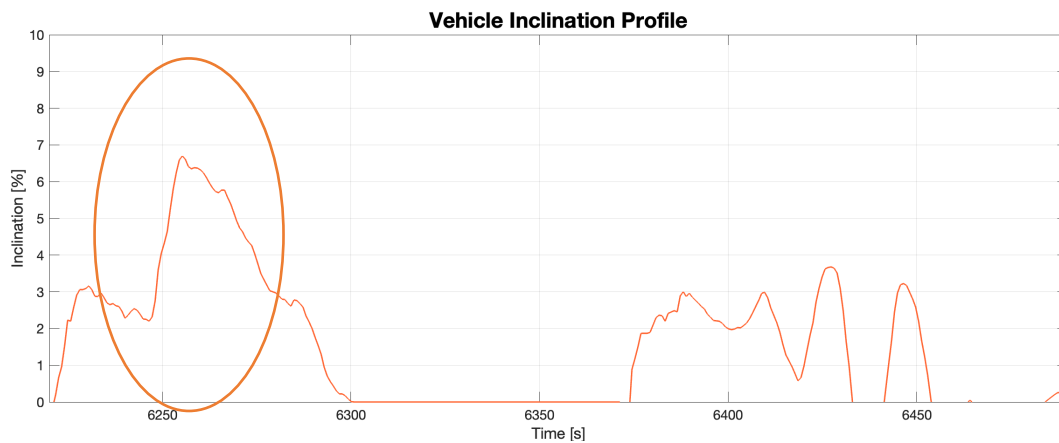


Figure 4.08 – Zoom on Inclination Profile
(RDC #1 x3, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7\text{ kW}$)

The high-power demand can also be seen in the battery state of charge curves in Figure 4.09, who suddenly begin to reduce at an accelerated rate. In the moment where the speed curves begin to deviate, the GT battery has already over 1 % more battery SOC than the Modelica battery. Although small, this difference is enough to cause the powertrain performance to begin to deteriorate significantly.

As both the GT model and the Modelica models fail to reach the target speed after just over an hour of running, starting from an 80 % initial state of battery charge, a near full state of charge when accounting for a 10 % unusable upper limit, it can be stated that the battery powertrain is insufficient. Hence, in later in the chapter another powertrain configuration is proposed which aims to simultaneously solve insufficient battery performance, as well as the inherent cost issues of such a large vehicle battery (over 70 000 € at 150 €/kWh, roughly one quarter of the purchase price of a bus).

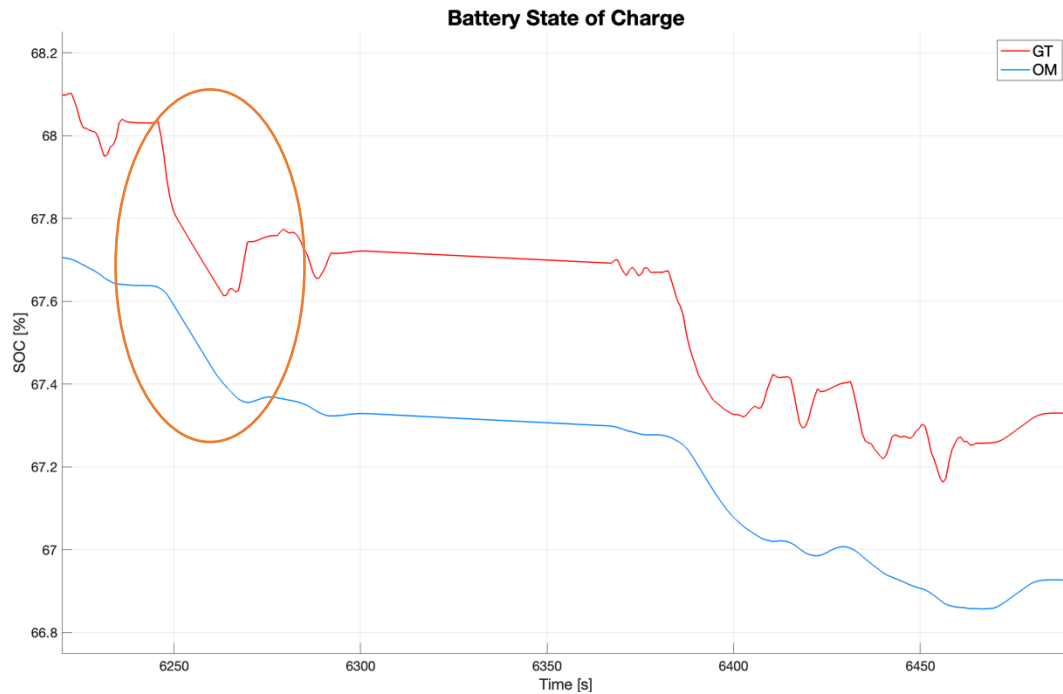


Figure 4.09 – Zoom on Comparison of BEV Inclination Profiles
(RDC #1 x3, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7\text{ kW}$)

It is also possible to look at the torque traces for the vehicles to further understand the differences between the two models. Observing Figure 4.10 referring to SORT 3 simply as the artificial nature of the cycle makes it easier to analyze, it is possible to observe four distinct phases. In a first phase, the vehicle is stationary, and thus the motor torque is null. This phase is nearly perfectly identical in both models, neglecting small overshoots, and different acceleration/braking times that were already previously elaborated upon. There are two bizarre instances where the Modelica model applies traction torque at standstill, yet does not accelerate, a clear error with the calibration, and a source of different consumption as this inherently drains the battery more. In a second phase, characterized by a hard constant vehicle acceleration, the motor torque increases nearly instantaneously to a target steady state value to achieve the desired acceleration. In both models, the driver is unable to reach the target perfectly, and an overshoot occurs. Furthermore, in the Modelica model, the driver model can be seen struggling to reach the steady state value, and the motor aggressively oscillates around the steady state torque value. In a third phase, the driver reaches the steady state velocity, and thus the load request reduces to a new lower steady state torque value. It can be observed that here, both models achieve this steady state velocity and maintain it with fewer torque oscillations. Still, the Modelica model does experience a small instance of regenerative braking as it overshoots its target velocity and must let off the throttle to slow the vehicle down to the steady state velocity. This differs from the GT model who reaches the steady state velocity by simply letting off the throttle. During the fourth and final phase, the electric motor is used to recover kinetic energy and thus a negative torque is registered in both models. The torque trace indicates a smoother regeneration phase during the braking of the GT model, as the Modelica model's torque oscillates uncontrollably due to being directly proportional to the brake pedal. Unfortunately, this was a design choice necessary to be imposed during modeling to avoid numerical chattering, who here has severe repercussions on the model behavior.

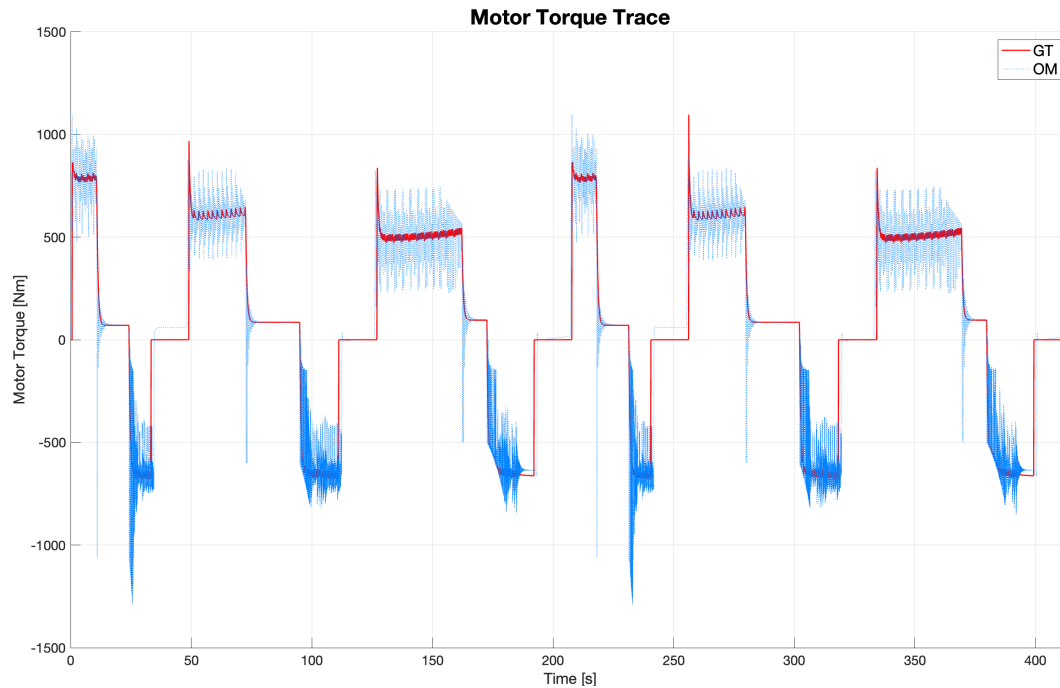


Figure 4.10 – Comparison of BEV Motor Torque Traces
(SORT3, 19 000 kg, SOC₀ = 80 % & P_{aux} = 7 kW)

The differences in torque control are seemingly reduced when considering a more dynamic cycle such as the MLTB indicated in Figure 4.11. Here, torque peaks may vary between models by significant amounts, but the overall shape of the torque traces are nearly identical. Naturally, as the cycle is much more irregular, further analysis becomes tricky, although the braking and regen control observed in the Modelica model continues to be inferior. The seemingly fewer differences in torque control are also a potential reason for why the Modelica driver appears to be closer to the GT driver during the less artificial driving cycles.

Whilst in Figure 4.02 and Figure 4.03 the models regarding fuel economy are shown, the models built for fuel economy are shown, it was also possible to alter settings in both to simulate top speed, acceleration or maximum slope tests. In Table 4.04, it is possible to observe a comparison of Modelica and GT-SUITE results in the case of maximum acceleration event from a standstill, on a few inclinations at a variety of initial states of charge. As expected, one can observe progressively longer acceleration times as both the initial state of charge decreases (due to lower OCV and higher internal equivalent resistance), and as the slope increases (due to higher resistive forces). The reason for choosing to simulate to fifty kilometers per hour is simply due to the vehicle being a city bus, and thus is being deemed both a representative velocity and a realistic one. It is interesting to note how the Modelica model underestimates the acceleration time with respect to the GT-SUITE model when at a near full initial state of charge. This behavior is inverted as soon as the initial battery state of charge is reduced, with the GT-model providing quicker acceleration times. It is worth noting that whilst there are modeling related reasons for these differences, these differences aren't the sole culprits for the variation in acceleration times.

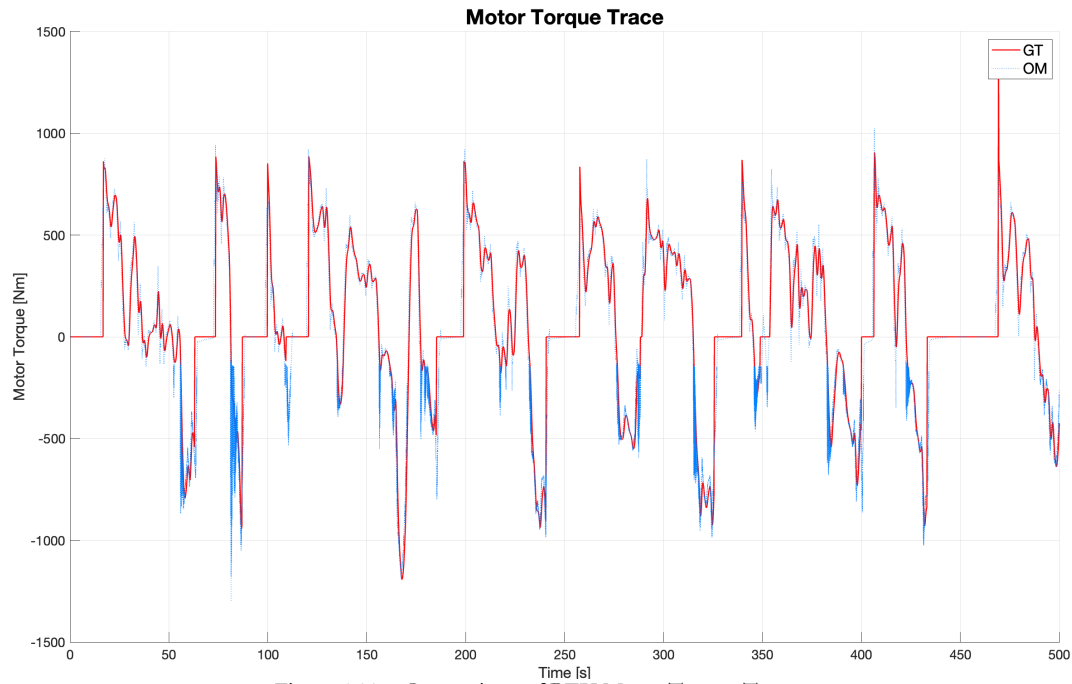


Figure 4.11 – Comparison of BEV Motor Torque Traces
(First 500 s of MLTB, 16 000 kg, SOC₀ = 80 % & P_{aux} = 7 kW)

Mass	SOC ₀	Inclination	GT-SUITE 0 – 50 km/h [s]	OpenModelica 0 – 50 km/h [s]	Percent Variation [%]
16 000	85	0	10.50	10.37	-1.24 %
		4	15.70	15.25	-2.87 %
	65	0	10.60	10.80	1.85 %
		4	15.90	16.26	2.26 %
	45	0	12.30	12.67	3.03 %
		4	20.70	22.08	6.67 %
	25	0	15.4	15.93	3.44 %
		4	No Data	40.85	No Data
19 000	85	0	12.90	12.68	-1.71 %
		4	22.10	21.31	3.57 %
	65	0	13.00	13.23	1.77 %
		4	22.30	23.01	3.18 %
	45	0	15.10	15.57	3.11 %
		4	≈31.7	35.08	No Data
	25	0	19.00	19.67	3.52 %
		4	Not Possible	Not Possible	No Data

Table 4.04 – GT-SUITE & OpenModelica BEV Acceleration Times (P_{aux} = 7.5 kW)



Figure 4.12 – Comparison of BEV Acceleration (Case #1)

Observing Figure 4.12, an example of the acceleration curves, it is possible to observe the same overall acceleration curve, with a “knee” and subsequent reduction in acceleration when the vehicle reaches roughly 30 km/h. This example, a zoom of Figure B.49, is not the only acceleration curve to have such a knee, as all other results displayed in ANNEX C contain such feature. Due to the lack of a gearbox, the vehicle's motor is brought to the base speed of the electric motor, and subsequently experiences a drop in torque. This is especially evident when plotting the motor torque for the same acceleration event, as shown in Figure 4.13. Here, it is possible to observe the maximum torque the motor can deliver varying through time, which, given the vehicle's ever-increasing speed, also shows the maximum motor torque curve in a slightly deformed manner.

From Figure 4.12 and Figure 4.13, the two main causes of differences in acceleration times can be identified. The first, is that GT-SUITE's model lower exploitation of the maximum motor torque at low speed, once again producing a smoother torque profile (like those observed in the driving cycles simulations performed previously). As a consequence, the OpenModelica model results in a slightly faster acceleration time, whilst the first is simply a small source of variation.

As Table 4.04 reports, not all Modelica models result in a faster acceleration. Observing a case where battery limitations play a stronger role in the vehicle performance, such as the one displayed in Figure 4.14, where the inclination is still null, but the battery initial state of charge is at 45 %, it is clear to see the maximum torque curve is not as straightforward as before. In the Modelica model upon reaching base speed, the torque drops off significantly. This vertical drop, as well as a dip in the torque at about 30 s into the cycle, are both consequences of the way in which battery limitations are imposed, via the graph displayed in Figure 3.18. Despite a very high interpolation density when converting the maps from Figure 3.16 and Figure 3.17 into Figure 3.18, some errors are generated in the map, visible by the little bubbles in the torque map. Consequently, the dips in torque visible as simply effects of the motor entering a “bubble” of lower torque in the battery limited maps. This, combined with only ever being able to use 98 % of the maximum battery power to avoid edge effects of the map interpolation, lead to the slower acceleration times despite a higher initial motor torque. In fact, observing Figure 4.15, containing the speed trace of this same case, it is possible to observe that the initial higher torque off-sets the GT-SUITES imposed initial speed, whilst the reduction in battery power then causes a divergence in the speed trace.

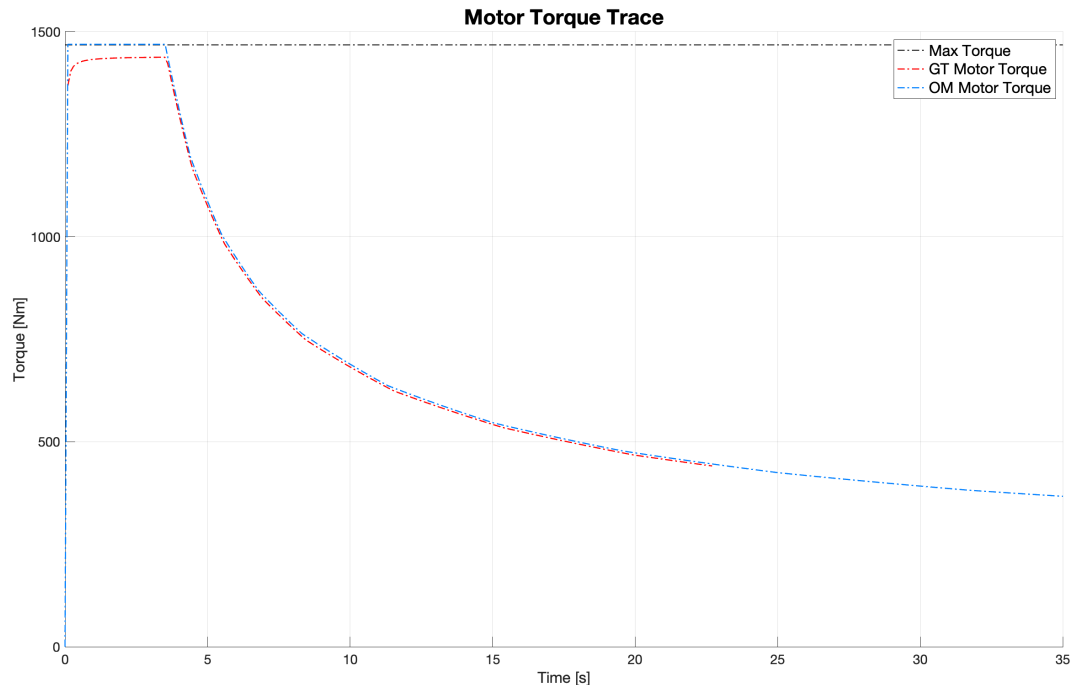


Figure 4.13 – Comparison of BEV Motor Torque during Acceleration Maneuver (Case #1)

It is worth also noting that whilst the very large dip in torque at around 30 s in Figure 4.14 doesn't play a role in the acceleration times evaluated here, it undoubtedly contributes to differences present in the cycle tests shown before. This torque dip is also visible in the velocity curves in the extended graphs as a small instance of constant velocity in the midst of the acceleration. This issue is not present in GT-SUITE where the software doesn't require battery limited maps, and only the efficiency maps, correctly managing the battery limitations internally, leading to the smoother torque traces.

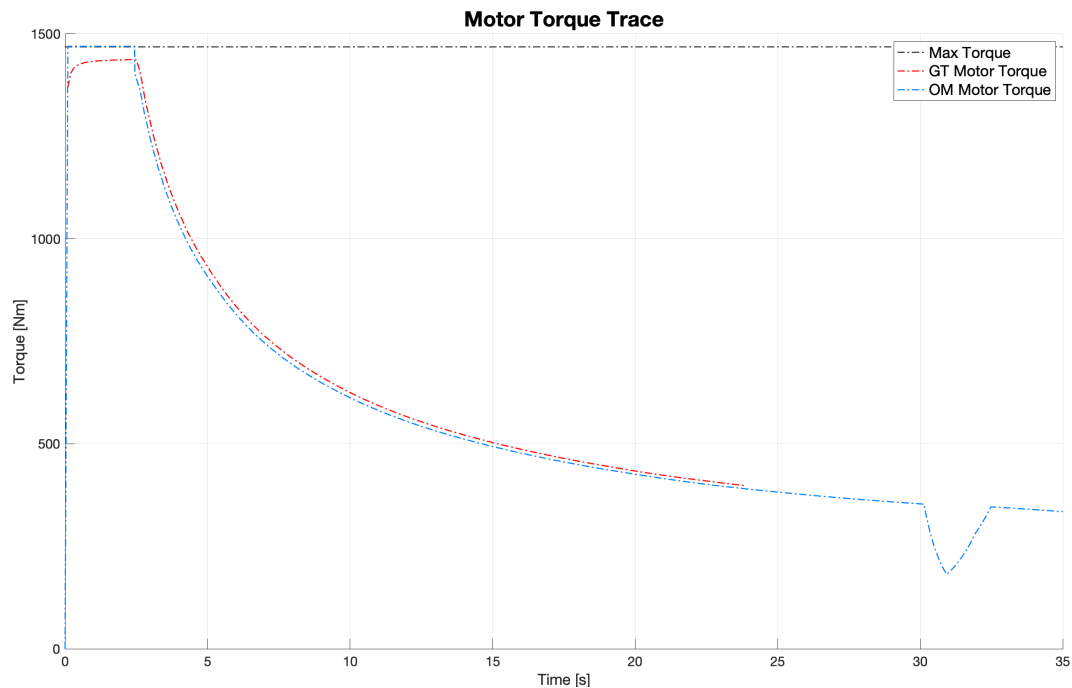


Figure 4.14 – Comparison of BEV Motor Torque during Acceleration Maneuver (Case #5)



Figure 4.15 – Comparison of BEV Motor Torque during Acceleration Maneuver (Case #5)

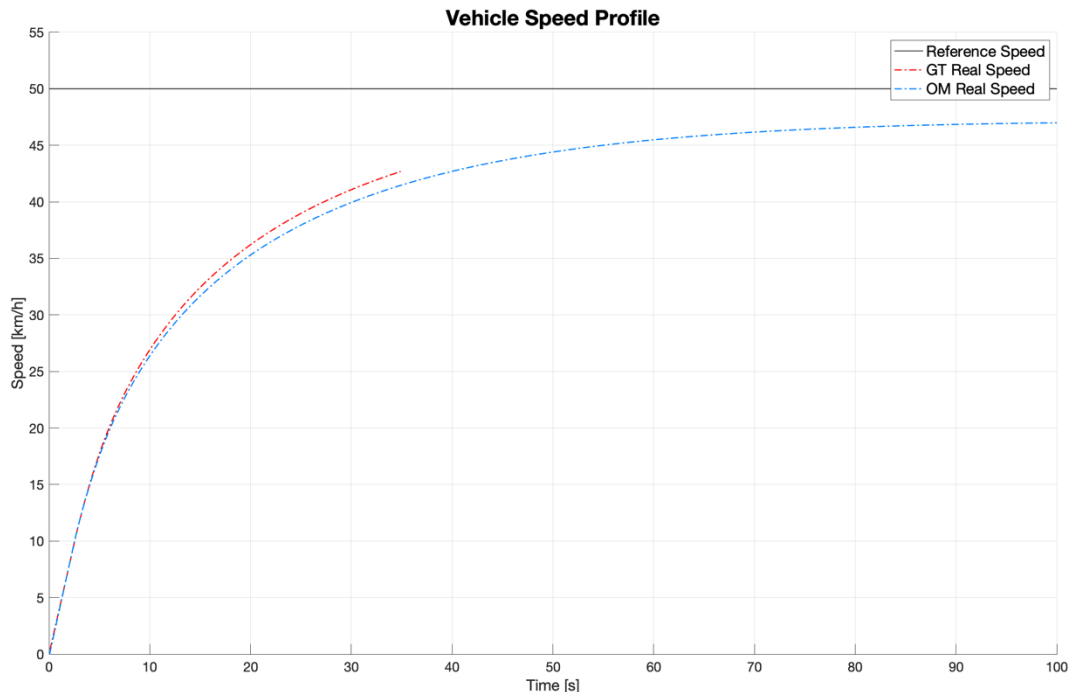


Figure 4.16 – Comparison of BEV Motor Torque during Acceleration Maneuver (Case #16)

Lastly, it is also possible to extend these graphs and observe the maximum speed of the vehicle on a designated inclination and initial state of charge. Considering Figure 4.16, which displays a speed trace of the vehicle climbing a 4 % inclination with a low initial state of charge (25 %) at maximum vehicle load, it is clear to see that the vehicle is unable to reach 50 km/h, and tops out at roughly 47 km/h. Though the vehicle is still experiencing a slight acceleration at the moment in which the graph ends, it is clear that as the battery SOC is



reduced due to hard acceleration, the maximum power the battery can deliver will be simultaneously reduced and the vehicle will begin to decelerate, despite being at full throttle.

FC Hybrid Model

FCH Model Assembly

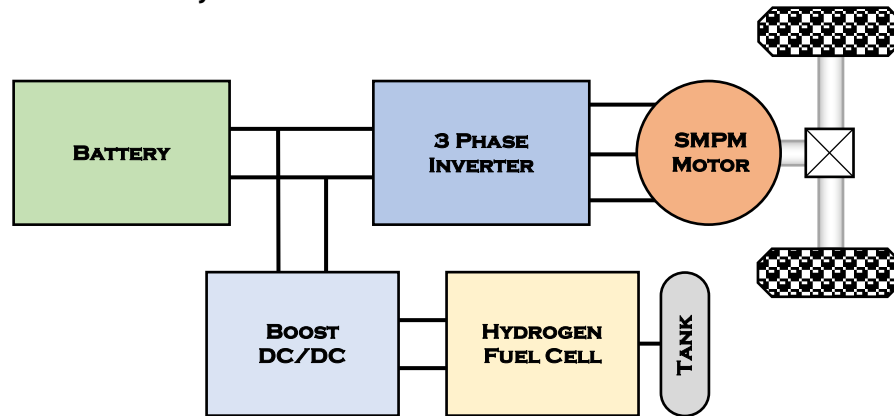


Figure 4.17 – Simple Series Hybrid Powertrain

A simple series fuel cell hybrid of order two and index one can be built following the schematic shown in Figure 4.17. This model introduces the passive DC/DC converter shown in Chapter 3 such as to compensate the voltage difference between the battery and the secondary energy source: the hydrogen fuel cell. To differentiate between the two voltage levels, text indicating “High Voltage” for the battery and “Low Voltage” for the fuel cell has been placed on the schematic.

The schematic shown in Figure 4.18 is assembled using *PUNCH_Torino* library similarly to how the BEV is assembled, with one component change and the addition of three others. The Motor Control Unit gets replaced by the Hybridization Control Unit, whilst the DC/DC converter, Fuel cell balance of plant and fuel cell all get added.

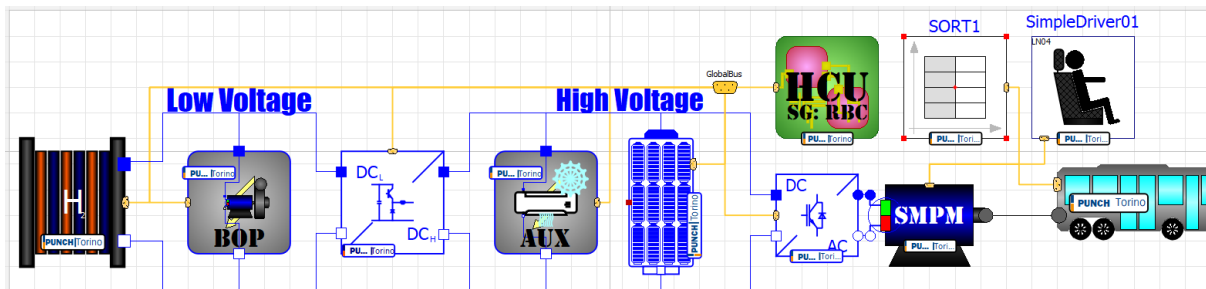


Figure 4.18 – OpenModelica FC Hybrid EV Model

Figure 4.18 illustrates the three connections present on the model, a torque connection from the motor to the bus, a high voltage electrical connection across the battery, auxiliaries, DC/DC converter and inverter, a low voltage electrical connection across the DC/DC converter, BOP and fuel cell, and finally a control bus connection across all elements for sharing data amongst the components. It is once again worth noting that no ground is visible on this model, as the negative pin of the battery, and the negative pin of the fuel cells are defined as the ground within their respective models.

Much like with the BEV model, it is possible to note the similarity in structure when comparing the structure of the model in OpenModelica and the GT-SUITE model shown in Figure 4.19.

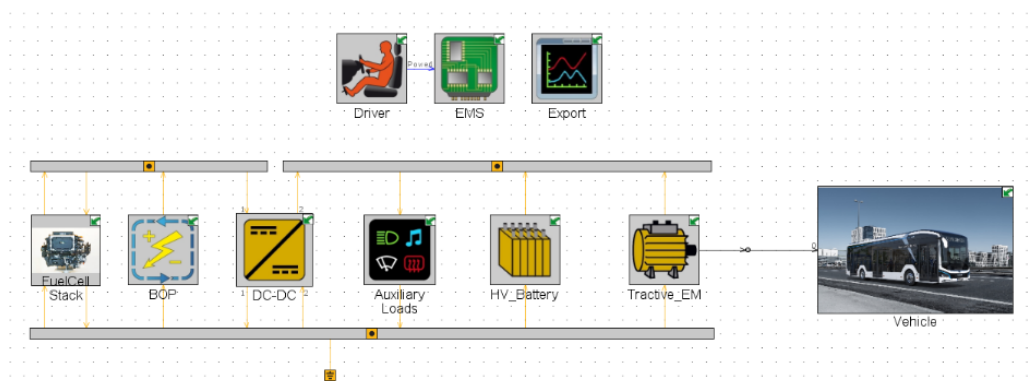


Figure 4.19 – GT-SUITE FC Hybrid EV Model

FCH Model Results

Speed Cycle	Mass [kg]	SOC ₀ [%]	SOC _f [%]	ΔSOC [%]	Battery Energy [kWh]	H ₂ [kg]	Consumption	
							[kWh/km]	[g/km]
SORT 1 Dist = 14.50 km t _r = 933.2 s	16 000	62	61.54	00.46	0.182	0.262	0.059	84.58
		48	60.43	-12.90	-5.108	0.539	-1.587	167.61
	19 000	62	60.90	01.10	0.436	0.314	0.140	101.33
		48	59.23	-11.23	-4.449	0.585	-1.436	188.82
SORT 2 Dist = 14.05 km t _r = 740 s	16 000	62	60.23	01.77	0.699	0.249	0.190	67.75
		48	58.92	-10.92	-4.325	0.525	-1.177	142.80
	19 000	62	58.63	03.37	1.334	0.288	0.363	78.27
		48	58.18	-10.18	-4.032	0.591	-1.098	160.92
SORT 3 Dist = 6.16 km t _r = 414.2 s	16 000	62	58.85	03.15	1.248	0.153	0.431	52.80
		48	56.65	-08.65	-3.426	0.422	-1.183	145.59
	19 000	62	57.75	04.25	1.683	0.188	0.581	64.87
		48	55.15	-07.15	-2.831	0.471	-0.978	162.53
Braunschweig Dist = 10.87 km t _r = 1740 s	16 000	62	58.83	03.17	1.255	0.820	0.116	75.60
		48	58.83	-10.83	-4.288	1.139	-0.395	104.97
	19 000	62	57.78	04.22	1.671	1.046	0.155	96.92
		48	57.78	-09.78	-3.872	1.388	-0.359	128.61
Old MLTB Dist = 9.00 km t _r = 2281 s	16 000	62	61.61	00.39	0.155	0.679	0.017	75.65
		48	61.61	-13.61	5.389	0.973	-0.600	108.37
	19 000	62	60.90	01.10	0.435	0.828	0.049	92.37
		48	60.87	-12.87	-5.097	1.122	-0.568	125.17
RDC #1 Dist = 16.59 km t _r = 2875 s	16 000	62	56.86	05.14	2.037	1.487	0.123	89.87
		48	56.86	-08.86	-3.507	1.816	-0.212	109.80
	19 000	62	54.16	07.84	3.105	1.893	0.189	114.96

Table 4.05 – Raw FC Hybrid Vehicle Results – 39.6 kWh Battery

Note: The following results were obtained with the bus auxiliaries draining 7 kW of power constantly, and are thus affected by the simulation duration, since the “idling” of the vehicle continues to consume a considerable amount energy.

Simulation		GT-Suite		OpenModelica		% Variation Batt Energy	% Variation H ₂
Speed Cycle	Mass	Batt. Energy [kWh]	H ₂ [kg]	Batt. Energy [kWh]	H ₂ [kg]		
SORT 1 Dist = 14.50 km t _r = 933.2 s	16 000	0.119	0.265	0.182	0.262	52.26 %	-1.05 %
	19 000	0.313	0.317	0.436	0.314	39.58 %	1.07 %
SORT 2 Dist = 14.05 km t _r = 740 s	16 000	0.460	0.257	0.699	0.249	52.13 %	-2.92 %
	19 000	1.186	0.286	1.334	0.288	12.53 %	0.42 %
SORT 3 Dist = 6.16 km t _r = 414.2 s	16 000	1.096	0.155	1.248	0.153	13.87 %	-1.29 %
	19 000	1.609	0.182	1.683	0.188	4.63 %	3.07 %
Braunschweig Dist = 10.87 km t _r = 1740 s	16 000	1.236	0.848	1.255	0.820	1.55 %	-3.30 %
	19 000	1.632	1.051	1.671	1.046	2.36 %	-0.43 %
Old MLTB Dist = 9.00 km t _r = 2281 s	16 000	0.119	0.712	0.155	0.679	30.08 %	-4.71 %
	19 000	0.294	0.864	0.435	0.828	48.31 %	-4.19 %
RDC #1 Dist = 16.59 km t _r = 2875 s	16 000	1.858	1.488	2.037	1.487	9.66 %	-0.08 %
	19 000	2.552	1.814	3.105	1.893	21.65 %	4.28 %

Table 4.06 – Comparison of GT-SUITE & OpenModelica FCH RAW Results (SOC₀ = 62 %, P_{aux} = 7 kW)

Comparing the overall energy consumption results of OpenModelica and GT-SUITE separately, it is possible to make some preliminary comments. Building upon what was seen with the BEV model it is clear that, on average, the Modelica Model has a higher battery discharge rate. It is worth noting however, that whilst the relative difference observed is quite high for the battery energy consumption, the absolute error is quite low. Such is also the case with the hydrogen consumption, wherein the absolute error is, for the most part, quite low.

To have a more complete understanding of the error, it is possible to convert the hydrogen consumed throughout the cycle into an equivalent electrical energy, add it to the battery energy consumed, and thus obtain a final single value to be used when comparing the two software. Naturally, this process is a gross approximation since the fuel cell efficiency varies throughout the cycle, and so too does the battery. That being said, as the fuel cells operate for large portions of time (as defined by the RBC) in situations where the power output is dependent on the battery SOC, and in both models the SOC remains relatively constant; it is fair to assume that the fuel cell efficiencies are at least comparable. As most PEM fuel cells operate in an efficiency window ranging from 40 % to 60 %, a 50% average efficiency will be taken, such as to convert the hydrogen energy into electrical energy. Doing so allows an estimated equivalent battery energy to be evaluated according to Equation 4.01.

$$E_{eq} = m_{H_2} \cdot LHV \cdot \eta_{FC} \cdot \eta_{DC/DC} + E_{batt} \quad \text{Eq 4.01}$$

The resulting Table 4.06 shows the results, with a clear increase in total energy consumption in case of the hybrid with respect to the pure BEV. This is expected due to the increased number of dissipative components along the transmission line. It is not however, a critique of the hybrid architecture, as the target of a load

follower hybrid fuel cell vehicle is not necessary to minimize energy consumption but rather ensuring adequate range and performance of the vehicle, as well as reducing downtime due to charging.

Simulation		GT-Suite	OpenModelica	% Variation Approximate Equivalent Energy
Speed Cycle	Mass	Approximate Equivalent Energy [kWh]	Approximate Equivalent Energy [kWh]	
SORT 1 Dist = 14.50 km t _r = 933.2 s	16 000	4.501	4.517	0.36 %
	19 000	5.558	5.626	1.21 %
SORT 2 Dist = 14.05 km t _r = 740 s	16 000	4.701	4.816	2.46 %
	19 000	5.919	6.087	2.84 %
SORT 3 Dist = 6.16 km t _r = 414.2 s	16 000	3.657	3.776	3.25 %
	19 000	4.622	4.789	3.60 %
Braunschweig Dist = 10.87 km t _r = 1740 s	16 000	15.257	14.813	-2.24 %
	19 000	18.997	18.961	-0.19 %
Old MLTB Dist = 9.00 km t _r = 2281 s	16 000	11.887	11.369	-4.36 %
	19 000	14.580	14.124	-2.65 %
RDC #1 Dist = 16.59 km t _r = 2875 s	16 000	26.454	26.614	0.60 %
	19 000	32.527	34.392	5.73 %

Table 4.06 – Comparison of GT-SUITE & OpenModelica FCH Equivalent Results (SOC₀ = 62 %, P_{aux} = 7 kW)

When comparing the relative differences of an equivalent energy consumption, it is possible to observe a smaller average relative percentage difference between the models simulated in GT-SUITE and in OpenModelica, despite the increased complexity of the system. Naturally these values are only indicative as they assume a constant efficiency of the fuel cell, however, a potential reason for such a reduction could be related to the fuel cell modelling in OpenModelica having been much more strictly related to the fuel cell modelling of GT-SUITE, due to the GT Model having become available at the time of modeling. This combined with the fuel cell being the primary energy source of the system, since the battery acts more like a buffer, should lead to smaller overall differences. Another observation that can be made is how, similarly to the BEV model, the relative differences increased when running high speed, high load cycles, such as the three artificial ones, and the RDC #1 cycle.

It is also possible to once again analyze the tracking errors generated to observe the different drivability of the powertrain with respect to the battery electric vehicle. Note that due to some data corruption issues, the GT-SUITE column for low mass simulations has been left empty, and the associated speed traces have been omitted from ANNEX D. It is possible to observe via the tracking error 2, much like in the BEV case, that the Modelica driver sticks much closer to the target speed profile than the GT model. Furthermore, much like in the BEV case, the Modelica model also worsens with high load request profiles, which can once again be attributed to the driver modeling & calibration.

Simulation		GT-Suite Driver Tracking Errors			OpenModelica Driver Tracking Errors		
Speed Cycle	Mass	Error 1 [m]	Error 2 [m/s]	Error 3 [m]	Error 1 [m]	Error 2 [m/s]	Error 3 [m]
SORT 1 Dist = 14.50 km $t_r = 933.2$ s	16 000	-	-	-	18.7	0.020	16.8
	19 000	152.1	0.163	5.782	21.2	0.023	18.8
SORT 2 Dist = 14.05 km $t_r = 740$ s	16 000	-	-	-	9.81	0.013	8.33
	19 000	133.6	0.181	6.954	11.7	0.016	9.82
SORT 3 Dist = 6.16 km $t_r = 414.2$ s	16 000	-	-	-	3.43	0.008	2.37
	19 000	92.3	0.223	6.443	4.58	0.011	3.51
Braunschweig Dist = 10.87 km $t_r = 1740$ s	16 000	-	-	-	27.2	0.016	21.9
	19 000	390.1	0.224	38.40	77.4	0.044	65.7
Old MLTB Dist = 9.00 km $t_r = 2281$ s	16 000	-	-	-	3.72	0.002	0.08
	19 000	449.1	0.197	74.23	5.47	0.002	1.35
RDC #1 Dist = 16.59 km $t_r = 2875$ s	16 000	-	-	-	54.3	0.019	46.53
	19 000	756.6	0.263	261.8	140.3	0.049	125.9

Table 4.07 – GT-SUITE & OpenModelica FCH Error Trackers ($SOC_0 = 62\%$)

It is however by comparing Figure 4.20, alongside Figure 4.21 and Figure 4.22 that it becomes possible to understand the cause for the underuse of the battery energy. Figure 4.20 displays how the lack of constraint on Modelica fuel cell power generation, combined with a very aggressive driver calibration creates a situation where the fuel cell power aggressively oscillates between maximum power and the average power of levels one and two. This occurs as the driver accelerates aggressively to meet the target speed profile, then completely releases the throttle once he reaches it. Due to the aggressive driver calibration, the driver inevitably overshoots the target, activating regenerative braking and slowing the vehicle back below the target velocity. This occurrence in rapid succession leads to the blue spikes present in Figure 4.20, as the fuel cell power oscillates between the two terms of Equation 3.26. To fix this, it is recommended that future models introduce a low pass filter at the output of the fuel cell power output in the hybridization power output. This will reduce drivability as power requests will be met later but improve the power delivery.

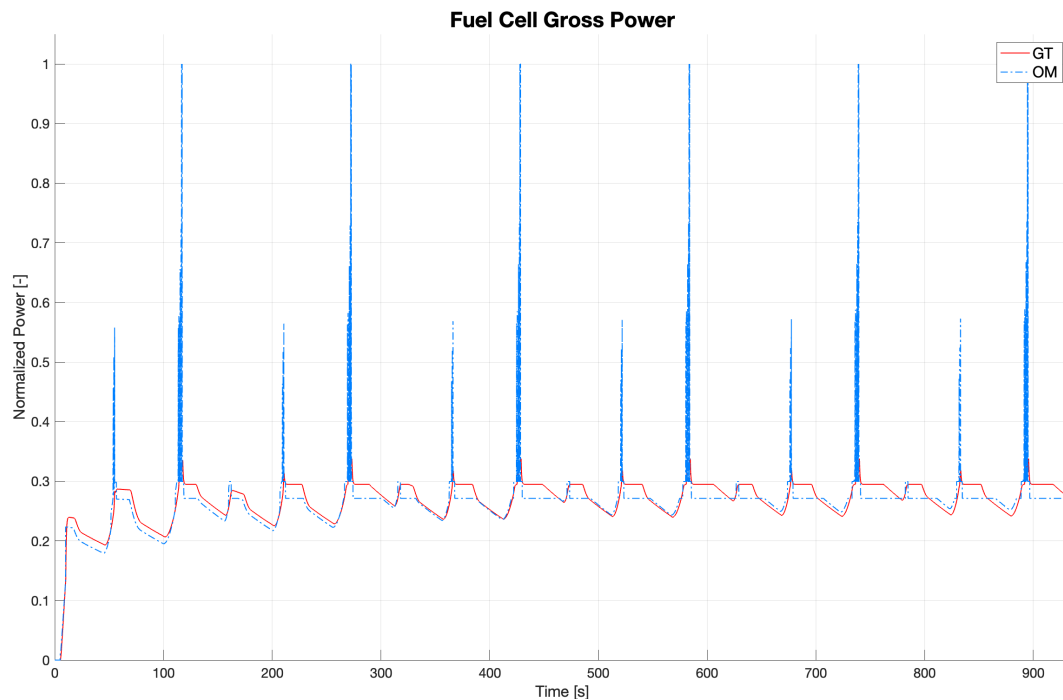


Figure 4.20 – Comparison of FCH Fuel Cell Gross Power
(SORT1, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7\text{ kW}$)

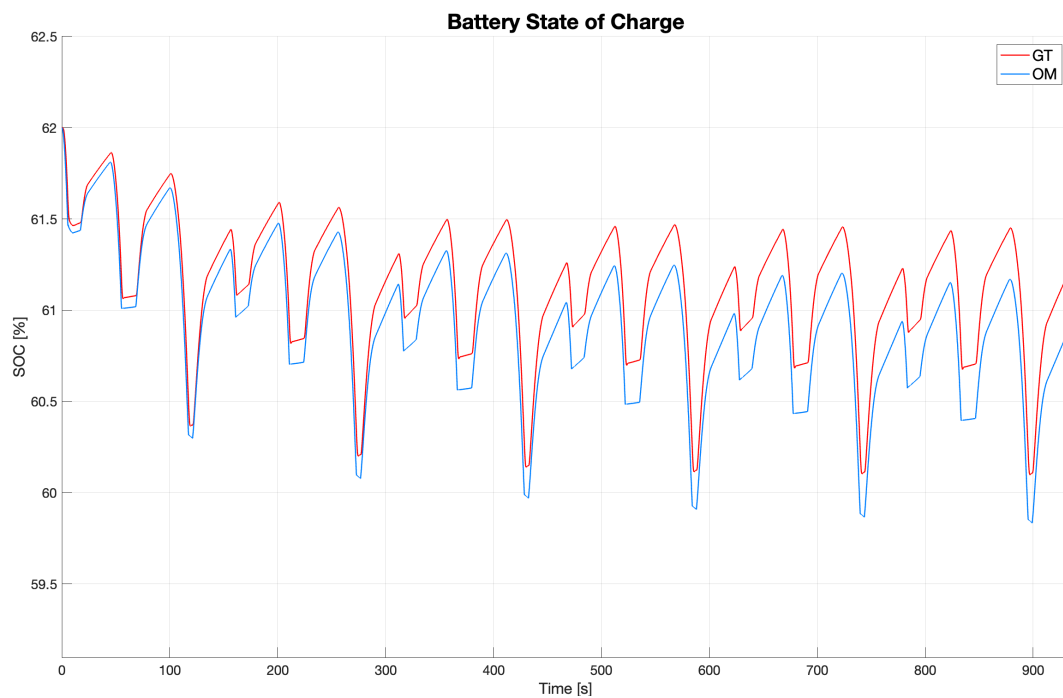


Figure 4.21 – Comparison of FCH SOC Discharge Profiles
(SORT1, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7\text{ kW}$)

It is also worth noting an important difference in level ones' power generation shortly after the second fuel cell power peak (shortly after 100 s) due to the hybridization mode switching to state four, fuel cell as a load follower. The power output should be dependent on the map shown in Figure 3.24 and on the Balance of Plant

map which in semi-linear steps up to 10 kW. However, given the battery states of charge are always relatively similar, and the map of Figure 3.24 indicates a constant maximum net power between 59 % and 61 % battery SOC of 25 kW, with a BOP power between 0 kW and 1 kW, in a system without power transients, the fuel cell shouldn't be able to surpass 26 kW whilst in charge level one and over 59 % battery SOC. This isn't the case with the GT-SUITE model, that partly simulates some transients via removing the assumption that fuel cell is capable of providing the power request instantaneously.

In addition, in the case of the MLTB cycle with a low vehicle mass, such as the one displayed in Figure 4.22 the lower hydrogen power production of the Modelica model with respect to the GT-Model despite no switching for several hundred seconds indicates that it is also likely that the maps provided for OpenModelica were not identical, and thus differences in the results are to be expected.

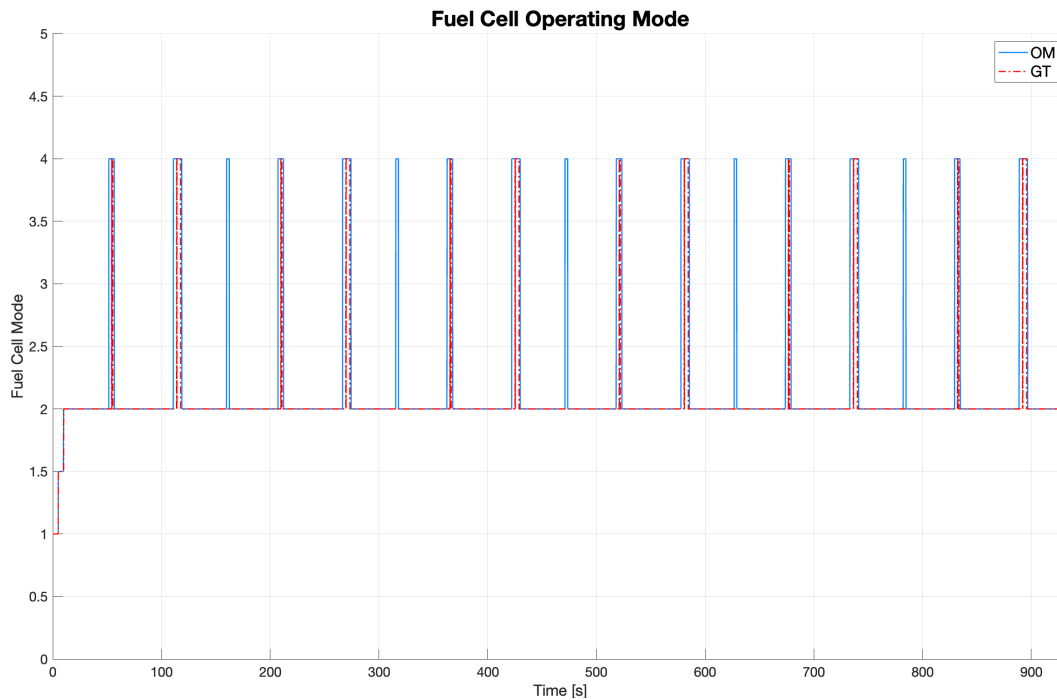


Figure 4.23 – Comparison of FCH Hybridization Mode
(SORT1, 19 000 kg, SOC₀ = 62 % & P_{aux} = 7 kW)

Thus, the lack of transients, and less aggressive recharge level one state of the fuel cell vehicle modelled in Open Modelica force a lower hydrogen consumption, but a larger battery energy depletion, with respect to the GT-SUITE model. It can also be reasoned that it is for this reason that the fuel cell vehicle model in Modelica switches more often to more aggressive fuel cell strategies, as indicated in Figure 4.23, due to its lower battery state of charge and the lower fuel cell power contribution when in charge level one.

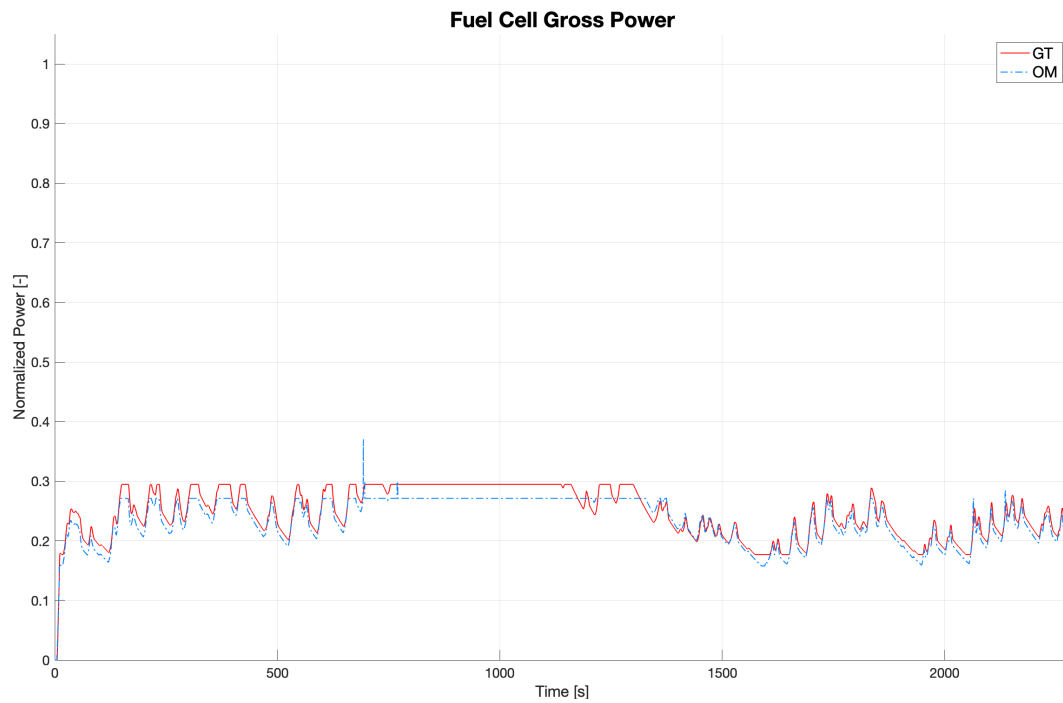


Figure 4.22 – Comparison of FCH Fuel Cell Gross Power
(MLTB, 16 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)



Figure 4.24 – Zoom on Comparison of FCH Speed Profiles
(Braunschweig, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)

Despite the extra power provided by the fuel cell, some instances on the driving cycles still prove too challenging for the bus to overcome, as highlighted by Figure 4.24. Here the more dynamic behavior of the OpenModelica fuel cell allows for the vehicle to more closely follow the cycle profile but fails to provide the



acceleration power requested. Such is also the case with the GT-SUITE model, that reaches roughly the same top speed slightly later due to the slower fuel cell response. To solve such cases, one can adopt a more aggressive hybridization strategy, a larger battery, or a larger fuel cell.

It is also possible to re-run these simulations with no auxiliary consumptions and observe the effects of the auxiliaries upon the vehicle. Doing so results in the consumptions reported in Table 4.08, with a preliminary analysis indicating the energy consumptions being markedly reduced. Once again, the relative difference in battery energy consumption indicates large percentile differences due to how small the absolute values in the variations are. Hence this table is re-evaluated assuming a constant fuel cell efficiency of 50 % over the whole cycle, as Equation 4.01 and the previous case, to determine an equivalent energy consumption.

The result of such an operation leads to the results shown in Table 4.09 which show indicates the system is under-estimating the energy consumptions on real driving cycles and slightly over-estimating on the artificial cycles, with respect to the GT model. The consequence of the different recharge maps is particularly evident in the case of the low vehicle mass running the MLTB cycle, wherein the vehicle nearly remains in charge level one throughout the entire driving cycle in both simulations (Figure 4.25). Since both maps lead to charge sustaining behaviors, the lower fuel cell power demand by Modelica model leads to a lower hydrogen consumption (Figure 4.26). However, as the fuel cell is operating at a lower power level, it requests a lower current, leading to fewer losses and thus lower overall energy consumption. This latter statement is also complimented by a second contribution, as the fuel cell operates during most of its cycle producing 15 kW or less, as seen in Figure 4.26, thus leading to no extra consumptions from the BOP (according to the map that was provided).

Simulation		GT-Suite		OpenModelica		% Variation Batt Energy	% Variation H ₂
Speed Cycle	Mass	Batt. Energy [kWh]	H ₂ [kg]	Batt. Energy [kWh]	H ₂ [kg]		
SORT 1 Dist = 14.50 km t _r = 933.2 s	16 000	-0.495	0.192	-0.392	0.188	-20.82 %	- 2.25 %
	19 000	-0.027	0.232	0.037	0.229	-237.76 %	-1.26 %
SORT 2 Dist = 14.05 km t _r = 740 s	16 000	0.017	0.198	0.094	0.194	457.57 %	-1.94 %
	19 000	0.343	0.246	0.410	0.244	19.69 %	-0.79 %
SORT 3 Dist = 6.16 km t _r = 414.2 s	16 000	0.463	0.139	0.567	0.136	22.59 %	-2.36 %
	19 000	1.129	0.159	1.172	0.161	3.79 %	1.75 %
Braunschweig Dist = 10.87 km t _r = 1740 s	16 000	0.342	0.696	0.400	0.661	17.03 %	-5.03 %
	19 000	1.170	0.873	1.154	0.860	-1.36 %	-1.53 %
Old MLTB Dist = 9.00 km t _r = 2281 s	16 000	-0.508	0.494	-0.485	0.461	-4.53 %	-6.83 %
	19 000	-0.128	0.633	-0.007	0.591	-94.43 %	-6.55 %
RDC #1 Dist = 16.59 km t _r = 2875 s	16 000	1.380	1.183	1.517	1.163	9.92 %	-1.67 %
	19 000	1.992	1.513	2.303	1.562	15.62 %	3.25 %

Table 4.08 – Comparison of GT-SUITE & OpenModelica FCH RAW Results (SOC₀ = 62 %, P_{aux} = 0 kW)

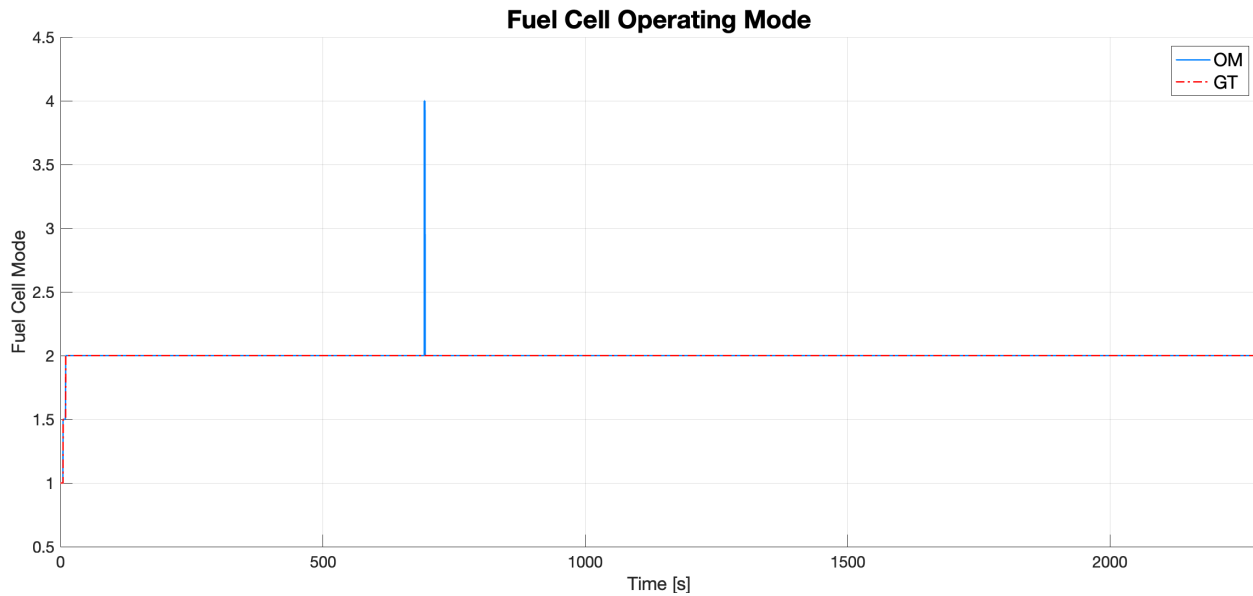


Figure 4.25 – Comparison of FCH Hybridization Mode
(MLTB, 16 000 kg, SOC₀ = 62 % & P_{aux} = 0 kW)

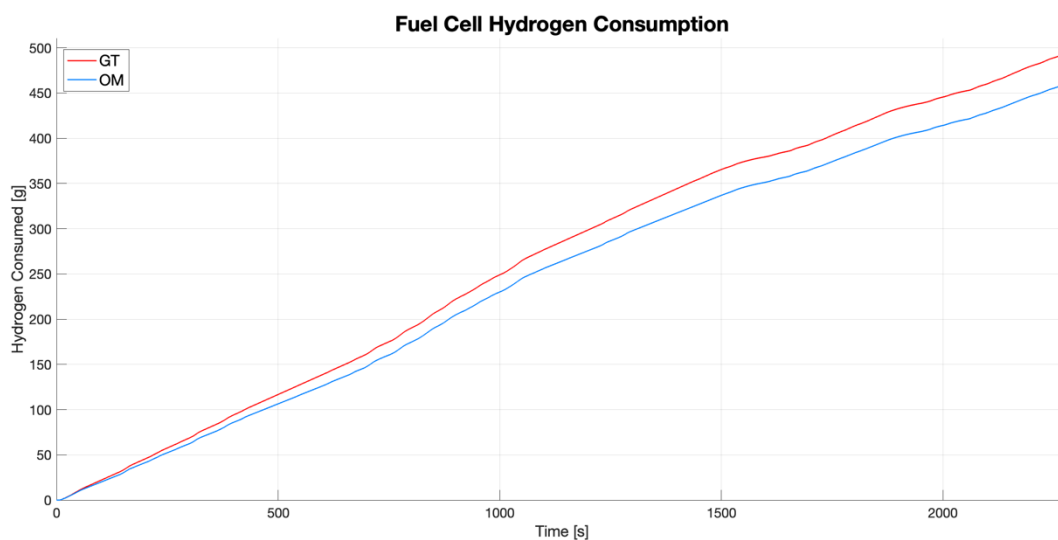


Figure 4.26 – Comparison of FCH Hydrogen Consumed
(MLTB, 16 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 0$ kW)

Simulation		GT-Suite	OpenModelica	% Variation Approximate Equivalent Energy
Speed Cycle	Mass	Approximate Equivalent Energy [kWh]	Approximate Equivalent Energy [kWh]	
SORT 1	16 000	2.613	2.646	1.27 %
	19 000	3.737	3.753	0.42 %
SORT 2	16 000	3.217	3.231	0.46 %
	19 000	4.331	4.367	0.83 %
SORT 3	16 000	2.720	2.771	1.88 %
	19 000	3.696	3.784	2.37 %
Braunschweig	16 000	11.610	11.101	-4.38 %
	19 000	15.308	15.075	-1.52 %
MLTB	16 000	7.500	6.976	-6.99 %
	19 000	10.121	9.569	-5.45 %
RDC #1	16 000	20.533	20.351	-0.89 %
	19 000	26.497	27.606	4.19 %

Table 4.09 – Comparison of GT-SUITE & OpenModelica FCH Equivalent Results ($SOC_0 = 62\%$, $P_{aux} = 0$ kW)

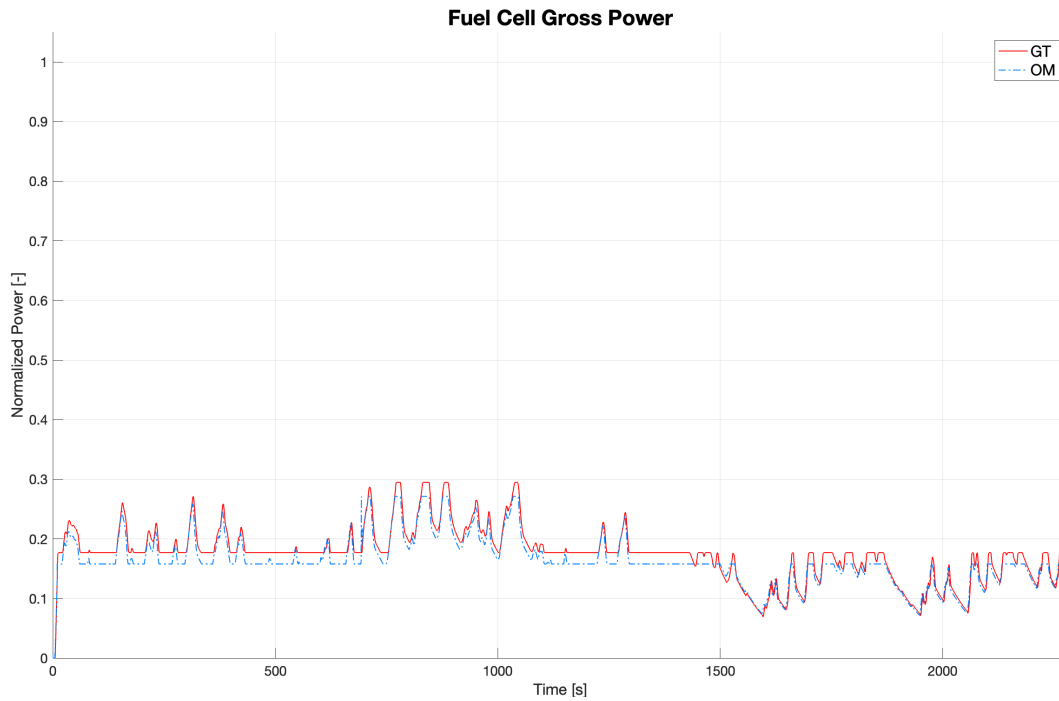


Figure 4.27 – Comparison of FCH Vehicle Fuel Cell Gross Power
(MLTB, 16 000 kg, SOC₀ = 62 % & P_{aux} = 0 kW)

It is also possible to evaluate the vehicle simulate vehicle accelerations and hill climb maneuvers, and obtain the results indicated in Table 4.10. It is clear to see that whilst the results obtained vary drastically. Despite the minimum off time condition for the fuel cell being removed, and the throttle trace being (nearly) identical in each model, with the GT-SUITE model being a constant 100 % actuation as opposed to Modelicas very fast ramp to 100 % actuation, the fuel cells begin to contribute power at different instances in time. This fact, combined with the lack of simulated transient in Modelica leads to the drastically different results.

Mass	SOC ₀	Inclination	GT-SUITE 0 – 50 km/h [s]	OpenModelica 0 – 50 km/h [s]
16 000	62	0	15.50	11.89
		4	20.40	17.23
		8	<i>Not Possible</i>	<i>Not Possible</i>
19 000	62	0	18.90	14.11
		4	23.65	49.74
		8	<i>Not Possible</i>	<i>Not Possible</i>

Table 4.10 – GT-SUITE & OpenModelica FCH Acceleration Times (P_{aux} = 7.5 kW)

Analyzing the results of the fourth case from Table 4.10, these differences can be observed in Figure 4.28 to Figure 4.31. Starting with Figure 4.31, the GT-SUITE model delays switching on the fuel cell until after the Modelica model. Observing Figure 4.29, it can be reasoned that this is a consequence of its reduced torque output and thus lower power demand. In Figure 4.29 it is also possible to note that both vehicles rapidly begin to operate in a battery power limited scenario, as the torque curve becomes very irregularly shape, in Modelicas case as a consequence of using the map in Figure 3.19. It is later possible to observe the sudden increase in acceleration of both FCH vehicles through Figure 4.27 as the fuel cells switch into the load follower modes. As the Modelica model simulates supplying the load request instantaneously (Figure 4.30), the sudden increase in available power translate to a spike in the torque delivered in Figure 4.29 roughly six seconds into the maneuverer. Thus, the acceleration is stronger in this second phase and leading to a faster time.



Figure 4.28 – Comparison of FCH Acceleration
(Case #4, 19 000 kg, 0 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW)

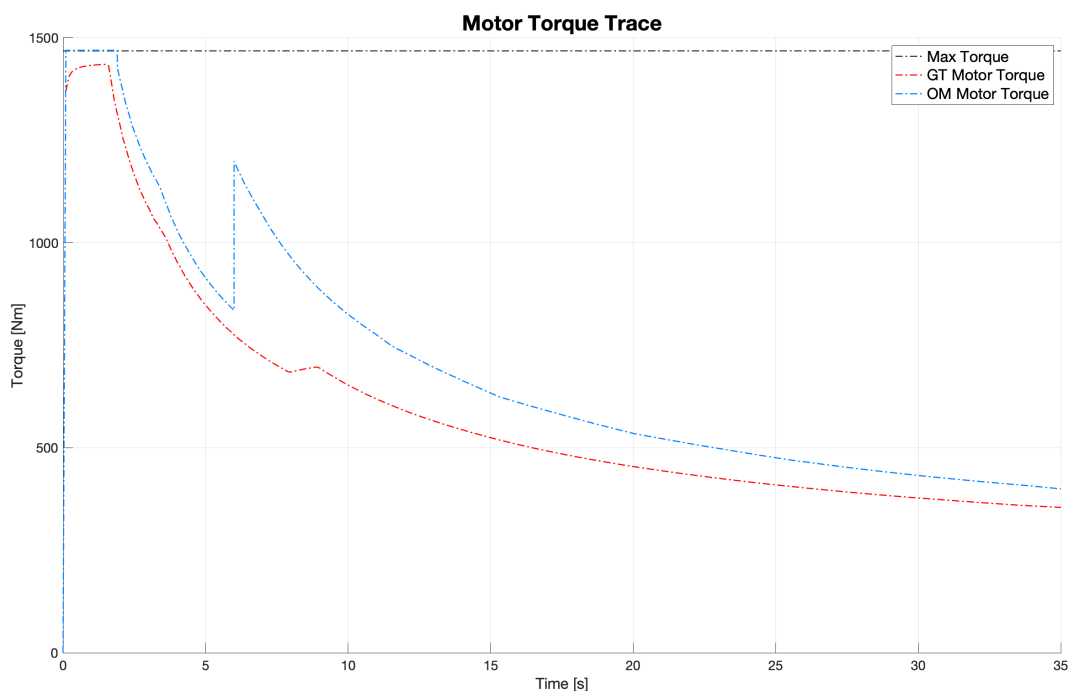


Figure 4.29 – Comparison of FCH Torque Trace
(Case #4, 19 000 kg, 0 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW)

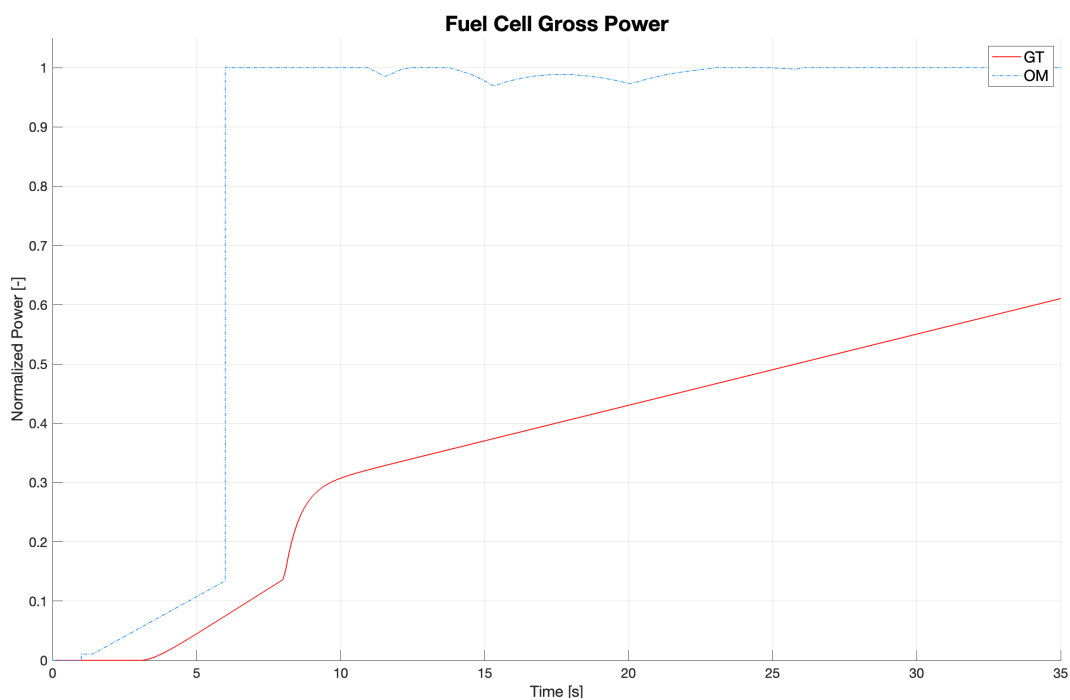


Figure 4.30 – Comparison of FCH Fuel Cell Gross Power
(Case #4, 19 000 kg, 0 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW)

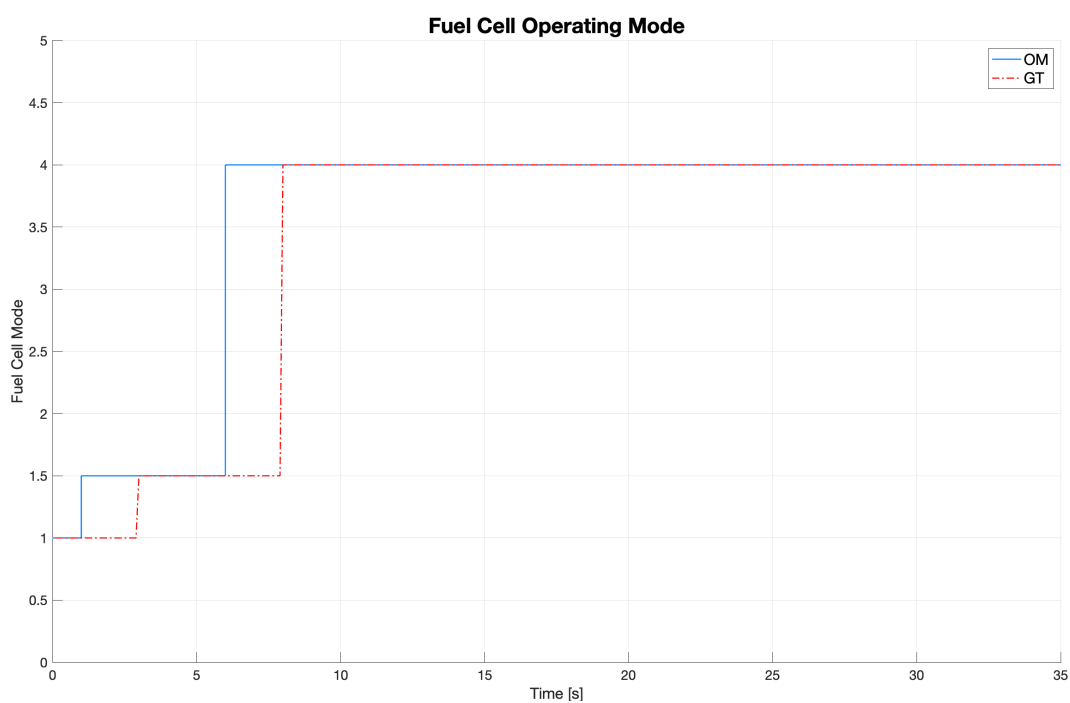


Figure 4.31 – Comparison of FCH Hybridization Model
(Case #4, 19 000 kg, 0 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW)

#5 Conclusions

OpenModelica, as a platform it is very powerful but it is very demanding, both from a software knowledge point of view, and from a technical knowledge point of view. Whilst the same can be said for many other simulation environments, the advantage of having a good software is that it can help to compensate for any gaps present in the user. With a complex and nuanced modeling language, and limited support, modeling in OpenModelica is inherently slower to GT-SUITE if a component hasn't been created yet. This is, however, OpenModelica's strongest feature, as its ability to create components and assemblies not present within its standard library provides it with a versatility not easily found in commercial software such as GT-SUITE. The presence of online libraries and a decent quantity of pre-built models helps OpenModelica, but they lack dedicated support, compatibility and are often not up to date with respect to OpenModelica releases. Despite this, it must be mentioned that thanks to the open nature of OpenModelica, it was still possible to obtain a relatively high level of confidence in working with the software over the span of eight-ish months working on this project.

However, the absence of many features present in other software can make OpenModelica a hassle to deal with. The lack of easy case setups or DOE's means the user must manually create such features themselves, which can be both laborious and frustrating. Other features such as separated results files for easy loading of model results were noted, which in case of long models that take several hours to run can prove problematic. The lack of choice of file formats with which to export results is also awkward, as even simple models such as those created for the purpose of this work can generate several GB's of CSV files. Furthermore, the fact that OMedit requires an external application for comparing results, makes it tricky to validate models, as one needs to export them and use an external comparison tool. These problems are solved via more advanced editors such as Dassault Dymola, but then one is reverting to a commercial editor of an open-source software, which limits the financial benefits of OpenModelica. Whilst Microsoft Excel is readily available to most users due to its relative affordable price point, it is unable to handle generation of plots with several hundred thousand data points with ease. This naturally leads the user to Mathworks MATLAB, as it is still relatively affordable compared to other software, but also due to its ability to handle and post process large quantities of data, a strength that was inherently useful when creating efficiency maps for this project.

If one were to consider the MATLAB's price point, being roughly four times cheaper than GT-SUITE with all its libraries included, many of which aren't needed in powertrain or vehicle simulation applications, it becomes hard to justify the use of OpenModelica as a viable alternative, given MATLAB, with Simulink and Simscape has equal if not superior versatility due to its constant development. Furthermore, OpenModelica's relative obscurity in the world of system simulation also results in another issue: lack of user familiarity. This naturally means longer time training an employee to become proficient in the software, and given its slower modeling and validation speed, more money spent by the manufacturer during development of new systems. Given the focus of automotive industries to reduce time to market and thus boost profits, a simulation software with slow model building, incapable of easily validation of models, and lacking features such as DOE's or case studies results in an undesirable environment.

OpenModelica does however remain very suitable for academic purposes, in a vacuum where agreements for university software licenses are not present, or alternatively to test novel concepts which one may find easier to implement in the Modelica environment. Although the models generated were never tested in co-simulation with other software and controllers, the fact that it is possible to model in OpenModelica and then export an FMU of the model to be implemented and used in other environments at least ensures OpenModelica can be used in conjunction with other software or other departments, in a case of parallel development of two models that later need to be simulated in tandem.

The case study presented indicates that it is possible to use OpenModelica as an alternative to GT-SUITE. Though the results do diverge by a significant amount, this is in large part due to the different modeling approaches used, since the two models being compared were created by different users and feature varying levels of details. Given the gross model differences, both from the control aspect and the considerations of fuel



cell transients, the fact that two vehicle architecture models are already within a 5 % error margin is a promising start. However, more complex models need to be developed for the library such as to ensure results are within the 2 % tolerance margin normally acceptable for simulation environments. In specific, were the project to be continued, focus would need to be concentrated upon improving the driver and fuel cell models.

This work also provides a basis for showing the power and use of hybrid vehicles in society moving forward. With a current obsession to ban ICE vehicles by the European Union, the use of Fuel Cell hybrid vehicles, especially on commercial vehicles that must be capable of running non-stop all day in congested city environments, can prove effective. This proves especially true for commercial vehicles as fewer recharge stations are necessary due to the ability to group them together at a depot. Furthermore, as this work demonstrates, the hybrid form factor also drastically reduces the battery size necessary on board the vehicle. In the case of the EU, a continent mostly void of Lithium and Cobalt, two of the most important metals currently in use in battery technologies, the possibility to create more vehicles from fewer rare earth metals should not be looked down upon.



Bibliography

Books

- [1] John G. Hayes, G. Abas Goodarzi, “Electric Powertrain: Energy Systems, Power Electronics and Drives for Hybrid, Electric and Fuel Cell Vehicles”, ISBN 9781119063643, 2018.
- [2] Giancarlo Genta, Lorenzo Morello, “The Automotive Chassis, Volume 2: System Design”, ISBN 978-1-4020-8673-1, 2009.

Websites

- [3] OpenModelica Website, User Documentation: “<https://openmodelica.org>” accessed during May and June 2022.
- Model Documentation: <https://build.openmodelica.org/Documentation/Modelica.html> accessed throughout May 2022 to December 2022.

Other

- [5] Ezio Spessa, Slides entitled “Heuristic Optimization: Rule Based Controllers (RBC)” from his course on “Energy Management in Hybrid and Electric Vehicles” taught for MSc in Automotive Engineering at Politecnico di Torino, 2020/2021.

ANNEX A – Equations

Battery Current

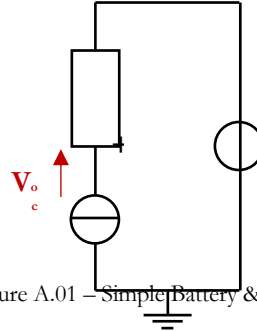


Figure A.01 – Simple Battery & Load

With reference to Figure A.01, the following passages are possible.

$$P_{user} = P_{batt} - P_{loss} \quad \text{Eq A.01}$$

$$P_{user} = v_{oc} \cdot I_{batt} - R_{eq} \cdot I_{batt}^2 \quad \text{Eq A.02.}$$

$$R_{eq} \cdot P_{user} = v_{oc} \cdot R_{eq} \cdot I_{batt} - R_{eq}^2 \cdot I_{batt}^2 \quad \text{Eq A.03}$$

$$-4 \cdot R_{eq} \cdot P_{user} = -4 \cdot v_{oc} \cdot R_{eq} \cdot I_{batt} + 4 \cdot R_{eq}^2 \cdot I_{batt}^2 \quad \text{Eq A.04}$$

$$v_{oc}^2 - 4 \cdot R_{eq} \cdot P_{user} = v_{oc}^2 - 4 \cdot v_{oc} \cdot R_{eq} \cdot I_{batt} + 4 \cdot R_{eq}^2 \cdot I_{batt}^2 \quad \text{Eq A.05}$$

$$v_{oc}^2 - R_{eq} \cdot P_{user} = (v_{oc} - 2 \cdot R_{eq} \cdot I_{batt})^2 \quad \text{Eq A.06}$$

$$\sqrt{v_{oc}^2 - R_{eq} \cdot P_{user}} = v_{oc} - 2 \cdot R_{eq} \cdot I_{batt} \quad \text{Eq A.07}$$

$$2 \cdot R_{eq} \cdot I_{batt} = v_{oc} - \sqrt{v_{oc}^2 - R_{eq} \cdot P_{user}} \quad \text{Eq A.08}$$

$$I_{batt} = \frac{v_{oc} - \sqrt{v_{oc}^2 - R_{eq} \cdot P_{user}}}{2 \cdot R_{eq}} \quad \text{Eq A.09}$$

ANNEX B – MATLAB Scripts

Power Limited Motor Operation

```
%% Preliminaries

clc
clear
close all

%% Algorithm

A1 = load("OriginalEta_TvsN.mat");
A2 = load("OriginalTT_TvsN.mat");
A3 = load("OriginalNN_TvsN.mat");
A4 = load("OriginalEta_Inverter.mat");
eta_mat = A1.eta_mat; % Motor
Efficiency
eta_mot = eta_mat; % Save
Motor Efficiency Map
n_vect = A2.n_vect; % Motor
Speeds [rpm]
T_vect = A3.T_vect; % Motor
Torques [Nm]
eta_inv = A4.eta_inverter_mat; %
Inverter Efficiency
eta_mat = eta_inv .* eta_mat;

figure;
contourf(n_vect, T_vect, eta_mot)
title('Motor Efficiency Map')
xlabel('Motor Speed [rpm]')
ylabel('Motor Torque [Nm]')
figure;
contourf(n_vect, T_vect, eta_inv)
title('Inverter Efficiency Map')
xlabel('Motor Speed [rpm]')
ylabel('Motor Torque [Nm]')

figure;
contourf(n_vect, T_vect, eta_mat)
title('Motor & Inverter Combo Efficiency
Map')
xlabel('Motor Speed [rpm]')
ylabel('Motor Torque [Nm]')
clear A1 A2 A3 A4 eta_inv
n_vect2 = linspace(0,6000, 201);
eta_mat2 = zeros(size(T_vect, 1),
size(n_vect2, 2));
for count_n = 1:201
    for count_T = 1:size(T_vect, 1)
        eta_mat2(count_T, count_n) =
interp2(n_vect, T_vect, eta_mat,
n_vect2(count_n), T_vect(count_T));
    end
end
n_vect = n_vect2;
eta_mat = eta_mat2;
clear eta_mat2 n_vect2
n_vect(1) = 1;
P_el = zeros(size(eta_mat, 1),
size(eta_mat, 2));
% Initialize Electrical Power Matrix
eta_mat2 = eta_mat;
for count_T = 1:size(T_vect, 1)
    eta_mat2(count_T, 1) = interp2(n_vect,
T_vect, eta_mat, 1, T_vect(count_T));
    end
    eta_mat = eta_mat2;
    clear eta_mat2
    for count_n = 1:size(eta_mat, 2)
        for count_T = 1:size(eta_mat, 1)
            eta = interp2(n_vect, T_vect,
eta_mat, n_vect(count_n), T_vect(count_T));
            if T_vect(count_T) >= 0
                k = 1;
            else
                k = -1;
            end
            P_el(count_T, count_n) =
(2*pi/60)*n_vect(count_n)*T_vect(count_T)/(e
ta^k);
        end
    end
    clear k
    figure
    contourf(n_vect, T_vect, abs(P_el))
    title('Motor & Inverter Combo Power Map')
    xlabel('Motor Speed [rpm]')
    ylabel('Motor Torque [Nm]')
    P_vect = linspace(min(P_el, [], 'all'),
max(P_el, [], 'all'), size(T_vect, 1));
    T_mat = zeros(size(P_vect, 1),
size(eta_mat, 2));
    for count_n = 1:size(eta_mat, 2)
        for count_P = 1:size(P_vect, 1)
            T_mat(count_P, count_n) =
interp1(P_el(:, count_n), T_vect,
P_vect(count_P));
        end
    end
    %%
    hold on
    figure
    contourf(n_vect, P_vect/1000, T_mat, 30)
    ylim([-850,850])
    title('Inverted Motor & Inverter Map')
    xlabel('Motor Speed [rpm]')
    ylabel('Electrical Power [kW]')
    Output = zeros(size(T_mat, 1) + 1,
size(T_mat, 2) + 1);
    Output(1, 2:size(Output, 2)) = n_vect;
    Output(2:size(Output, 1), 1) = P_vect;
    Output(2:size(Output, 1), 2:size(Output, 2))
= T_mat;
    OutputLimits = zeros(size(n_vect, 2), 3);
    for count_n = 1:size(n_vect, 2)
        [~, idx_max] = max(T_mat(:, count_n));
        [~, idx_min] = min(T_mat(:, count_n));
        OutputLimits(count_n, :) = [n_vect(1,
count_n), P_vect(idx_max, 1), P_vect(idx_min,
1)];
    end
```

Post Processing Scripts

```
%% Import Results GT SUITE

% The following code aims to import the
results generated in GT-SUITE and
% Open Modelica generated during a Masters
Thesis entitled "Vehicle
% Performance Simulations with
OpenModelica"

%% Preliminaries

clear
close all
clc

%% Select File Title

disp("Select A Software from the Menu...")
choice1 = menu("What Software is the Data
From?", ["GT SUITE", "OMedit", "Both"]);
if choice1 == 1
    s1 = "GT";
    s0 = ".txt";
elseif choice1 == 2
    s1 = "OM";
    s0 = ".csv";
else
    s1 = "Both";
    s0 = ".txt";
end
disp(join(["Selection: " s1]))

disp("Select the Powertrain
Architecture...")
choice2 = menu("What Transmission are we
Referring to?", ["BEV", "FCH"]);
if choice2 == 1
    s2 = "BEV";
    disp(join(["Selection: " s2]))
    disp(" ")
    disp("Conditions: ")
    choice3 = menu("What is the Simulation
Goal?", ["Fuel Economy",
"Acceleration/Slopes"]);
    if choice3 == 1
        s3 = "";
        choice4 = menu("What is the
Simulated Driving Cycle?", ["SORT1", "SORT2",
"SORT3", "Braunschweig", "MLTB",
"Gillingham", "Gillingham x3"]);
        if choice4 == 1
            s4 = "SORT1";
        elseif choice4 == 2
            s4 = "SORT2";
        elseif choice4 == 3
            s4 = "SORT3";
        elseif choice4 == 4
            s4 = "Braunschweig";
        elseif choice4 == 5
            s4 = "MLTB";
        elseif choice4 == 6
            s4 = "Gillingham";
        elseif choice4 == 7
            s4 = "Gillinghamx3";
        end
        disp(join(["Selection: " s4
"(Speed Cycle)"]))
        s5 = "80";
```

```
disp(join(["Selection: (def) " s5
"% (SOC0)"]))
s6 = "";
disp("Selection: (def) 7000 W
(Auxiliaries)")
else
    s3 = "Accel_";
    choice4 = menu("What is the
Slope?", ["0 deg", "4 deg", "8 deg"]);
    if choice4 == 1
        s4 = "0deg";
    elseif choice4 == 2
        s4 = "4deg";
    elseif choice4 == 3
        s4 = "8deg";
    end
    disp(join(["Selection: " s4
"(Slope)"]))

    choice5 = menu("Initial SOC?", ["85
%", "65 %", "45 %", "35 %", "25 %"]);
    if choice5 == 1
        s5 = "85";
    elseif choice5 == 2
        s5 = "65";
    elseif choice5 == 3
        s5 = "45";
    elseif choice5 == 4
        s5 = "35";
    elseif choice5 == 5
        s5 = "25";
    end
    disp(join(["Selection: " s5 "%
(SOC0)"]))
    s6 = "_Aux75";
    disp("Selection: 7500 W
(Auxiliaries)")
end
else
    s2 = "FCH";
    disp(join(["Selection: " s2]))
    disp(" ")
    disp("Conditions: ")

    choice3 = menu("What is the Simulation
Goal?", ["Fuel Economy",
"Acceleration/Slopes"]);
    if choice3 == 1
        s3 = "";
        choice4 = menu("What is the
Simulated Driving Cycle?", ["SORT1", "SORT2",
"SORT3", "Braunschweig", "MLTB",
"Gillingham", "Gillingham x3"]);
        if choice4 == 1
            s4 = "SORT1";
        elseif choice4 == 2
            s4 = "SORT2";
        elseif choice4 == 3
            s4 = "SORT3";
        elseif choice4 == 4
            s4 = "Braunschweig";
        elseif choice4 == 5
            s4 = "MLTB";
        elseif choice4 == 6
            s4 = "Gillingham";
        elseif choice4 == 7
            s4 = "Gillinghamx3";
        end
        end
```



```

disp(join(["Selection:      "      s4
"(Speed Cycle)"]))

choice5 = menu("Initial SOC?", ["62
%", "30 %"]);
if choice5 == 1
    s5 = "62";
else
    s5 = "30";
end
disp(join(["Selection:      "      s5      "%
(SOC0)"]))

choice6 = menu("Auxiliaries?",
["7k W", "0 W"]);
if choice6 == 1
    s6 = "";
disp("Selection:      7000      W
(Auxiliaries)")
else
    s6 = "_Aux0";
disp("Selection:      0      W
(Auxiliaries)")
end
else
    s3 = "Accel _";
choice4 = menu("What is the
Slope?", ["0 deg", "2 deg", "4 deg", "6 deg",
"8 deg", "10 deg", "12 deg"]);
if choice4 == 1
    s4 = "0deg";
elseif choice4 == 2
    s4 = "2deg";
elseif choice4 == 3
    s4 = "4deg";
elseif choice4 == 4
    s4 = "6deg";
elseif choice4 == 5
    s4 = "8deg";
elseif choice4 == 6
    s4 = "10deg";
elseif choice4 == 7
    s4 = "12deg";
end
disp(join(["Selection:      "      s4
"(Slope)"]))

choice5 = menu("Initial SOC?", ["62
%", "30 %"]);
if choice5 == 1
    s5 = "62";
else
    s5 = "30";
end
disp(strjoin(["Selection:      " s5      "%
(SOC0)"]))

s6 = "_Aux75";
disp("Selection:      7500      W
(Auxiliaries)")
end
end

choice7 = menu("Vehicle Mass?", ["15 570
kg", "19 000 kg"]);
if choice7 == 1
    s7 = "15t";
disp("Selection: 15 570 kg")
else
    s7 = "19t";
disp("Selection: 19 000 kg")
end

temp = strjoin([s3 s2 "_" s1 "_" s4 "_" s7
"_" s5 s6 s0]);
fileName = strrep(temp, " ", "");

%% Define Variables

if choice1 == 1
    Data1 = importdata(fileName, ',', 4);
elseif choice1 == 2
    Data1 = importdata(fileName, ',', 2);
else
    Data1 = importdata(strrep(strjoin([s3
s2 "_GT_" s4 "_" s7 "_" s5 s6 ".txt"]), " ",
""));
    Data2 = importdata(strrep(strjoin([s3
s2 "_OM_" s4 "_" s7 "_" s5 s6 ".csv"]), " ",
""));
end
clear temp s0 s2 s3 s4 s6 s7

time1 = Data1.data(:, 1); %
[s] Time Elapsed
RefSpeed1 = Data1.data(:, 2); %
[m/s] Reference Velocity
Incl1 = Data1.data(:, 3); %
[deg] Inclination
RealSpeed1 = Data1.data(:, 4); %
[m/s] Real Velocity
Throttle1 = Data1.data(:, 5); %
[-] Throttle Pedal
Brake1 = Data1.data(:, 6); %
[-] Brake Pedal
SOC1 = Data1.data(:, 7); %
[-] State of Charge
OCV1 = Data1.data(:, 8); %
[V] Open Circuit Voltage
Req1 = Data1.data(:, 9); %
[Ω] Equivalent Internal Resistance
Vbatt1 = Data1.data(:, 10); %
[V] Battery Terminal Voltage
Ibatt1 = Data1.data(:, 11); %
[A] Battery Current
if size(Data1.data, 2) > 11
    Iaux1 = Data1.data(:, 12); %
[A] Auxiliary Current
Torque1 = Data1.data(:, 13); %
[Nm] Motor Torque
RPM1 = Data1.data(:, 14); %
[RPM] Motor Rotational Velocity
torque_flag = 0;
else
    torque_flag = 1;
disp(' ')
disp('### MISSING TORQUE DATA ###')
end
if choice2 == 2
    State1 = Data1.data(:, 15); %
[-] Fuel Cell Control State
H2_dot1 = Data1.data(:, 16); %
[g/s] Hydrogen Consumption Rate
H21 = Data1.data(:, 17); %
[g] Hydrogen Consumption

```

```
Vfc1= Data1.data(:,18); %
[V] Fuel Cell Voltage
    Ifc1 = Data1.data(:, 19); %
[A] Fuel Cell Current
end

if choice1 == 3
    time2 = Data2.data(:, 1); %
[s] Time Elapsed
    RefSpeed2 = Data2.data(:, 2); %
[m/s] Reference Velocity
    Inc2 = Data2.data(:, 3); %
[deg] Inclination
    RealSpeed2 = Data2.data(:, 4); %
[m/s] Real Velocity
    Throttle2 = Data2.data(:, 5); %
[-] Throttle Pedal
    Brake2 = Data2.data(:, 6); %
[-] Brake Pedal
    SOC2 = Data2.data(:, 7); %
[-] State of Charge
    OCV2 = Data2.data(:, 8); %
[V] Open Circuit Voltage
    Req2 = Data2.data(:, 9); %
[Ω] Equivalent Internal Resistance
    Vbatt2 = Data2.data(:, 10); %
[V] Battery Terminal Voltage
    Ibatt2 = Data2.data(:, 11); %
[A] Battery Current
    Iaux2 = Data2.data(:, 12); %
[A] Auxiliary Current
    Torque2 = Data2.data(:, 13); %
[Nm] Motor Torque
    RPM2 = Data2.data(:, 14); %
[RPM] Motor Rotational Velocity
    if choice2 == 2
        State2 = Data2.data(:,15); %
[-] Fuel Cell Control State
        H2_dot2 = Data2.data(:,16); %
[kg/s] Hydrogen Consumption Rate
        H22 = Data2.data(:, 17); %
[kg] Hydrogen Consumption
        Vfc2 = Data2.data(:,18); %
[V] Fuel Cell Voltage
        Ifc2 = Data2.data(:, 19); %
[A] Fuel Cell Current
    end
end

if choice1 == 1 || choice1 == 3
    RealSpeed1 = RealSpeed1/3.6;
    RefSpeed1 = RefSpeed1/3.6;
end

flag = 0;
% The following if condition performs a
check where some files had a wrong data
exported in place of the velocity.
dt = timel(2, 1) - timel(1, 1);
if (choice1 == 1 || choice1 == 3) &&
abs(sum(RealSpeed1)/size(RealSpeed1,1)) < 1
&& choice3 == 1
    flag = 1;
    disp(" ")
    disp("DATA CONTAINED ERRORS!")
    disp(".....")
    disp(".....")
    disp(".....")
```

```

disp(":")
disp("!!")
disp("!!!")
disp("!!!!")
disp("!!!!!")
disp("!!!!!!")
disp("!!!!!!!")
disp("!!!!!!!!")
disp("!!!!!!!")
disp("!!!!!!")
disp("!!!!!")
disp("!!!!")
disp("!!!")
disp("!!")
disp(":")
disp("DATA CONTAINED ERRORS!")
end
clear count1

%% Error Analysis

if choice3 == 1
    disp(" ")
    cycledist = 
sum(RefSpeed1.*time1)/1000;
    disp(strjoin(["Cycle Reference Distance" num2str(cycledist) " km"]))

    sz1 = size(time1,1);
    disp(" ")
    if choicel ~= 3
        disp(strjoin(["Driver Errors in " s1]))
    else
        disp("Driver Errors in GT")
    end
    Err1a = sum(abs((time1(2:sz1) - time1(1:(sz1 - 1)))).*(RefSpeed1(2:sz1)-RealSpeed1(2:sz1))));
    disp(strjoin(["Err1:" num2str(Err1a)]))
    Err2a = sum(abs((time1(2:sz1) - time1(1:(sz1 - 1)))).*(RefSpeed1(2:sz1)-RealSpeed1(2:sz1)))/(time1(sz1));
    disp(strjoin(["Err2:" num2str(Err2a)]))
    Err3a = sum((time1(2:sz1) - time1(1:(sz1 - 1)))).*(RefSpeed1(2:sz1)-RealSpeed1(2:sz1)));
    disp(strjoin(["Err3:" num2str(Err3a)]))

    if choicel == 3
        sz2 = size(time2,1);
        disp(" ")
        disp("Driver Errors in OM")
        Err1b = sum(abs((time2(2:sz2) - time2(1:(sz2 - 1)))).*(RefSpeed2(2:sz2)-RealSpeed2(2:sz2))));
        disp(strjoin(["Err1:" num2str(Err1b)]))
        Err2b = sum(abs((time2(2:sz2) - time2(1:(sz2 - 1)))).*(RefSpeed2(2:sz2)-RealSpeed2(2:sz2)))/(time2(sz2));
        disp(strjoin(["Err2:" num2str(Err2b)]))
        Err3b = sum((time2(2:sz2) - time2(1:(sz2 - 1)))).*(RefSpeed2(2:sz2)-RealSpeed2(2:sz2)));

```

[illegible]

```

if choicel == 1
    BattEnergy1 = (SOC0 - SOC1(szl))*475.4;
    disp(['GT Battery Energy: ' num2str(BattEnergy1) ' kWh'])
elseif choicel == 2
    BattEnergy1 = (SOC0 - SOC1(szl))*475.4;
    disp(['OM Battery Energy: ' num2str(BattEnergy1) ' kWh'])
else
    BattEnergy1 = (SOC0 - SOC1(szl))*475.4;
    disp(['GT Battery Energy: ' num2str(BattEnergy1) ' kWh'])
    BattEnergy2 = (SOC0 - SOC2(sz2))*475.4;
    disp(['OM Battery Energy: ' num2str(BattEnergy2) ' kWh'])
    pervarl = ((BattEnergy2 - BattEnergy1)/BattEnergy1)*100;
    disp(' ')
    disp(['Battery Energy Variation: ' num2str(pervarl) ' %'])
end
else
    if choicel == 1
        BattEnergy1 = (SOC0 - SOC1(szl))*39.6;
        disp(['GT Battery Energy: ' num2str(BattEnergy1) ' kWh'])
        H21f = H21(szl);
        disp(['GT Hydrogen Consumed: ' num2str(H21f) ' kg' ])
    elseif choicel == 2
        BattEnergy1 = (SOC0 - SOC1(szl))*39.6;
        disp(['OM Battery Energy: ' num2str(BattEnergy1) ' kWh'])
        H21f = H21(szl)*1000;
        disp(['OM Hydrogen Consumed: ' num2str(H21f) ' kg' ])
    else
        BattEnergy1 = (SOC0 - SOC1(szl))*39.6;
        disp(['GT Battery Energy: ' num2str(BattEnergy1) ' kWh'])
        H21f = H21(szl);
        disp(['GT Hydrogen Consumed: ' num2str(H21f) ' g' ])
        BattEnergy2 = (SOC0 - SOC2(sz2))*39.6;
        disp(['OM Battery Energy: ' num2str(BattEnergy2) ' kWh'])
        H22f = H22(sz2)*1000;
        disp(['OM Hydrogen Consumed: ' num2str(H22f) ' g'])
        disp(' ')
        pervarl = ((BattEnergy2 - BattEnergy1)/BattEnergy1)*100;
        disp(['Battery Energy Variation: ' num2str(pervarl) ' %'])
        pervar2 = ((H22f - H21f)/H21f)*100;
        disp(['Hydrogen Variation: ' num2str(pervar2) ' %'])
        EqEnergy1 = BattEnergy1 + (0.98*0.5*119000*H21f)/(1000*3600);
        disp(' ')

```




```
disp(['GT Equivalent Energy  
Consumption: ' num2str(EqEnergy1) ' kWh'])  
EqEnergy2 = BattEnergy2 +  
(0.98*0.5*119000*H22f)/(1000*3600);  
disp(['OM Equivalent Energy  
Consumption: ' num2str(EqEnergy2) ' kWh'])  
disp(' ')  
pervar3 = ((EqEnergy2 -  
EqEnergy1)/EqEnergy1)*100;  
disp(['Equivalent Energy  
Variation: ' num2str(pervar3) ' %'])  
end  
end
```

%% Fuel Economy Graphs

```
if choice3 == 1  
% Speed Plot  
figure  
hold on;  
grid on;  
if choicel ~= 3  
if choicel == 1  
plot(time1, 3.6*RefSpeed1, 'k')  
plot(time1, 3.6*RefSpeed1 + 2,  
'Color', "#CCCCCC")  
plot(time1, 3.6*RefSpeed1 - 2,  
'Color', "#CCCCCC")  
plot(time1, 3.6*RealSpeed1, 'r--')  
set(gca, 'fontsize', 14)  
else  
plot(time1, 3.6*RefSpeed1, 'k')  
plot(time1, 3.6*RefSpeed1 + 2,  
'Color', "#CCCCCC")  
plot(time1, 3.6*RefSpeed1 - 2,  
'Color', "#CCCCCC")  
plot(time1, 3.6*RealSpeed1, 'b--')  
set(gca, 'fontsize', 14)  
end  
title(strjoin([s1 "Vehicle Speed  
Profile"]), 'FontSize', 22)  
legend('Reference Speed', '', '', 'Real  
Speed', 'FontSize', 16)  
else  
if flag == 0  
plot(time1, 3.6*RefSpeed1 + 2,  
'Color', "#CCCCCC", 'LineWidth', 1.5)  
plot(time1, 3.6*RefSpeed1 - 2,  
'Color', "#CCCCCC", 'LineWidth', 1.5)  
plot(time1, 3.6*RefSpeed1, 'k',  
'LineWidth', 1.25)  
plot(time1, 3.6*RealSpeed1, 'r-.',  
'LineWidth', 1.25)  
plot(time2, 3.6*RealSpeed2, '-  
,', 'LineWidth', 1.25, 'Color', [0 0.5 1])  
set(gca, 'fontsize', 14)  
title("Vehicle Speed Profile",  
'FontSize', 22)  
legend('', '', 'Reference Speed',  
'GT Real Speed', 'OM Real Speed', 'FontSize',  
16)  
else  
plot(time1, 3.6*RefSpeed1 + 2,  
'Color', "#CCCCCC", 'LineWidth', 1.5)  
plot(time1, 3.6*RefSpeed1 - 2,  
'Color', "#CCCCCC", 'LineWidth', 1.5)  
plot(time1, 3.6*RefSpeed1, 'k',  
'LineWidth', 1.25)
```

```
plot(time2, 3.6*RealSpeed2, '-  
,', 'LineWidth', 1.25, 'Color', [0 0.5 1])  
set(gca, 'fontsize', 14)  
title("Vehicle Speed Profile",  
'FontSize', 22)  
legend('', '', 'Reference Speed',  
'OM Real Speed', 'FontSize', 16)  
end  
end  
xlim([0, max(time1)])  
ylim([0, 65]);  
xlabel('Time [s]', 'FontSize', 16)  
ylabel('Speed [km/h]', 'FontSize', 16)
```

% Battery Power

```
figure  
hold on;  
grid on;  
if choicel ~= 3  
if choicel == 1  
plot(time1, Vbatt1.*Ibatt1/1000,  
'r', 'LineWidth', 0.75)  
set(gca, 'fontsize', 14)  
else  
plot(time1, Vbatt1.*Ibatt1/1000,  
'-', 'LineWidth', 0.75, 'Color', [0 0.5 1])  
set(gca, 'fontsize', 14)  
end  
title(strjoin([s1 "Battery State of  
Charge"]), 'FontSize', 22)  
else  
plot(time1, Vbatt1.*Ibatt1/1000, 'r-',  
'LineWidth', 1.25)  
plot(time2, Vbatt2.*Ibatt2/1000,  
'-', 'LineWidth', 0.75, 'Color', [0 0.5 1])  
set(gca, 'fontsize', 14)  
title("Battery Power", 'FontSize', 22)  
legend('GT', 'OM', 'FontSize', 16)  
end  
xlim([0, max(time1)])  
ylim([-250, 300]);  
xlabel('Time [s]', 'FontSize', 16)  
ylabel('Battery Power [kW]', 'FontSize',  
16)
```

% SOC Plot

```
figure  
hold on;  
grid on;  
if choicel ~= 3  
if choicel == 1  
plot(time1, SOC1*100, 'r',  
'LineWidth', 0.75)  
set(gca, 'fontsize', 14)  
else  
plot(time1, SOC1*100, '-  
,', 'LineWidth', 0.75, 'Color', [0 0.5 1])  
set(gca, 'fontsize', 14)  
end  
title(strjoin([s1 "Battery State of  
Charge"]), 'FontSize', 22)  
else  
plot(time1, SOC1*100, 'r-',  
'LineWidth', 1.25)  
plot(time2, SOC2*100, '-  
,', 'LineWidth',  
1.25, 'Color', [0 0.5 1])  
set(gca, 'fontsize', 14)
```



```
title("Battery State of Charge",
'FontSize', 22)
legend('GT', 'OM', 'FontSize', 16)
end
xlim([0, max(time1)])
ylim([min(SOC1)*100 - 1, max(SOC1)*100 +
0.5]);
xlabel('Time [s]', 'FontSize', 16)
ylabel('SOC [%]', 'FontSize', 16)

% Torque Request Plot
figure
hold on;
grid on;
if choice1 ~= 3
    if choice1 == 1
        plot(time1, Torque1, 'r',
'LineWidth', 0.75)
        set(gca, 'fontsize', 14)
    else
        plot(time1, Torque1, '-
', 'LineWidth', 0.75, 'Color', [0 0.5 1])
        set(gca, 'fontsize', 14)
    end
    title(strjoin([s1 "Motor Torque"])),
'FontSize', 22)
    else
        plot(time1, Torque1, 'r-',
'LineWidth', 1.25)
        plot(time2, Torque2,
':', 'LineWidth', 0.75, 'Color', [0 0.5 1])
        set(gca, 'fontsize', 14)
        title("Motor Torque Trace",
'FontSize', 22)
        legend('GT', 'OM', 'FontSize', 16)
    end
    xlim([0, max(time1)])
    ylim([-1500, 1500]);
    xlabel('Time [s]', 'FontSize', 16)
    ylabel('Motor Torque [Nm]', 'FontSize', 16)

% Fuel Cell Operating Mode
if choice2 == 2
    figure
    hold on;
    grid on;
    if choice1 ~= 3
        if choice1 == 1
            plot(time1, State1, 'r',
'LineWidth', 0.75)
            set(gca, 'fontsize', 14)
        else
            plot(time1, State1, '-
', 'LineWidth', 0.75, 'Color', [0 0.5 1])
            set(gca, 'fontsize', 14)
        end
        title(strjoin([s1 "Fuel Cell
Operating Mode"])), 'FontSize', 22)
        else
            plot(time2, State2, '-
', 'LineWidth', 1.25, 'Color', [0 0.5 1])
            plot(time1, State1, 'r-',
'LineWidth', 1.25)
            set(gca, 'fontsize', 14)
            title("Fuel Cell Operating Mode",
'FontSize', 22)
            legend('OM', 'GT', 'FontSize', 16)
        end

        xlim([0, max(time1)])
        ylim([0, 5])
        xlabel('Time [s]', 'FontSize', 16)
        ylabel('Fuel Cell Mode', 'FontSize',
16)
    end

% Fuel Cell Hydrogen Consumed
if choice2 == 2
    figure
    hold on;
    grid on;
    if choice1 ~= 3
        if choice1 == 1
            plot(time1, H21, 'r',
'LineWidth', 0.75)
            set(gca, 'fontsize', 14)
            ylim([0, H21(sz1)+50])
        else
            plot(time1, H21*1000, '-
', 'LineWidth', 0.75, 'Color', [0 0.5 1])
            set(gca, 'fontsize', 14)
            ylim([0, H21(sz1)*1000+50])
        end
        title(strjoin([s1 "Fuel Cell
Hydrogen Consumption"])), 'FontSize', 22)
        else
            plot(time1, H21, 'r-',
'LineWidth', 1.25)
            plot(time2, H22*1000, '-
', 'LineWidth', 1.25, 'Color', [0 0.5 1])
            set(gca, 'fontsize', 14)
            title("Fuel Cell Hydrogen
Consumption", 'FontSize', 22)
            legend('GT', 'OM', 'FontSize', 16)
            ylim([0, H22(sz2)*1000+50])
        end
        xlim([0, max(time1)])
        xlabel('Time [s]', 'FontSize', 16)
        ylabel('Hydrogen Consumed [g]',
'FontSize', 16)
    end

% Fuel Cell Power
if choice2 == 2
    figure
    hold on;
    grid on;
    if choice1 ~= 3
        if choice1 == 1
            plot(time1, Vfc1.*Ifc1/1000,
'r', 'LineWidth', 0.75)
            set(gca, 'fontsize', 14)
        else
            plot(time1, Vfc1.*Ifc1/1000,
'b', 'LineWidth', 0.75)
            set(gca, 'fontsize', 14)
        end
        title(strjoin([s1 "Fuel Cell
Hydrogen Consumption"])), 'FontSize', 22)
        else
            plot(time1, (Vfc1.*Ifc1)/1000, 'r-
', 'LineWidth', 1)
            plot(time2, (Vfc2.*Ifc2)/1000, '-
.', 'LineWidth', 1, 'Color', [0 0.5 1])
            set(gca, 'fontsize', 14)
            title("Fuel Cell Gross Power",
'FontSize', 22)
```



```
        legend('GT', 'OM', 'FontSize', 16)
    end
    xlim([0, max(time1)])
    ylim([0, 100])
    xlabel('Time [s]', 'FontSize', 16)
    ylabel('Power [kW]', 'FontSize', 16)
end

%% Acceleration Graphs
else
    figure
    hold on;
    grid on;
    if choice1 ~= 3
        if choice1 == 1
            plot(time1, 3.6*RefSpeed1, 'k-.',
'LineWidth', 1.25)
            plot(time1, 3.6*RealSpeed1, 'r--',
'LineWidth', 1.25)
            set(gca, 'fontsize', 14)
        else
            plot(time1, 3.6*RefSpeed1, 'k-.',
'LineWidth', 1.25)
            plot(time1, 3.6*RealSpeed1, '--
', 'LineWidth', 1.25, 'Color', [0 0.5 1])
            set(gca, 'fontsize', 14)
        end
        title(strjoin([s1 "Vehicle Speed
Profile"]), 'FontSize', 22)
        legend('Reference Speed', 'Real
Speed', 'FontSize', 16)
    else
        plot(time2, 3.6*RefSpeed2, 'k',
'LineWidth', 0.75)
        plot(time1, 3.6*RealSpeed1, 'r-.',
'LineWidth', 1.25)
        plot(time2, 3.6*RealSpeed2, '-
.', 'LineWidth', 1.25, 'Color', [0 0.5 1])
        set(gca, 'fontsize', 14)
        title("Vehicle Speed Profile",
'FontSize', 22)
        legend('Reference Speed', 'GT Real
Speed', 'OM Real Speed', 'FontSize', 16)
    end
    xlim([0,
35])%0.5*(max(time1)+max(time2))]
    ylim([0, 55]);
    xlabel('Time [s]', 'FontSize', 16)
    ylabel('Speed [km/h]', 'FontSize', 16)

    figure
    hold on;
    grid on;

    if torque_flag == 1
        if choice1 == 1
            disp(' ')
            disp('### ERROR DISPLAYING GT
TORQUE GRAPHS ###')
        else
            plot([0, 100], [1467.9 1467.9], 'k-
.', 'LineWidth', 0.75)
            plot(time2, Torque2, '-
.', 'LineWidth', 0.75, 'Color', [0 0.5 1])
            set(gca, 'fontsize', 14)
```

```
        title("Motor Torque Trace",
'FontSize', 22)
        legend('Max Torque', 'OM Motor
Torque', 'FontSize', 16)
    end
    else
        if choice1 ~= 3
            if choice1 == 1
                plot([0, 100], [1467.9
1467.9], 'k:');
                plot(time1, Torque1, 'r--',
'LineWidth', 1.25)
                set(gca, 'fontsize', 14)
                title("GT Motor Torque Trace",
'FontSize', 22)
            else
                plot([0, 100], [1467.9
1467.9], 'k-.')
                plot(time1, Torque1, '--
', 'LineWidth', 1.25, 'Color', [0 0.5 1])
                set(gca, 'fontsize', 14)
                title("OM Motor Torque Trace",
'FontSize', 22)
            end
            legend('Max Torque', 'Motor
Torque', 'FontSize', 16)
        else
            plot([0, 100], [1467.9
1467.9], 'k-.', 'LineWidth', 1)
            plot(time1, Torque1, 'r-.',
'LineWidth', 1.25)
            plot(time2, Torque2, '-
.', 'LineWidth', 1.25, 'Color', [0 0.5 1])
            set(gca, 'fontsize', 14)
            title("Motor Torque Trace",
'FontSize', 22)
            legend('Max Torque', 'GT Motor
Torque', 'OM Motor Torque', 'FontSize', 16)
        end
    end
    xlim([0,
35])%0.5*(max(time1)+max(time2))]
    ylim([0, 1500]);
    xlabel('Time [s]', 'FontSize', 16)
    ylabel('Torque [Nm]', 'FontSize', 16)

    if choice2 == 2
        figure
        hold on;
        grid on;
        if choice1 ~= 3
            if choice1 == 1
                plot(time1,
Vfc1.*Ifc1/1000, 'r', 'LineWidth', 0.75)
                set(gca, 'fontsize', 14)
            else
                plot(time1,
Vfc1.*Ifc1/1000, '-.', 'LineWidth', 0.75,
'Color', [0 0.5 1])
                set(gca, 'fontsize', 14)
            end
            title(strjoin([s1 "Fuel Cell
Hydrogen Consumption"]))
        else
            plot(time1, (Vfc1.*Ifc1)/1000,
'r-', 'LineWidth', 1)
            plot(time2, (Vfc2.*Ifc2)/1000,
'-.', 'LineWidth', 0.75, 'Color', [0 0.5 1])
```



```
set(gca, 'fontsize', 14)
title("Fuel Cell Gross Power",
'FontSize', 22)
legend('GT', 'OM', 'FontSize',
16)
end
xlim([0, 35])
ylim([0, 100])
xlabel('Time [s]', 'FontSize', 16)
ylabel('Power [kW]', 'FontSize',
16)
end

if choice2 == 2
figure
hold on;
grid on;
if choice1 ~= 3
if choice1 == 1
plot(time1, State1, 'r',
'LineWidth', 0.75)
set(gca, 'fontsize', 14)
else
plot(time1, State1, '-
', 'LineWidth', 0.75, 'Color', [0 0.5 1])
```

```
set(gca, 'fontsize', 14)
end
title(strjoin([s1 "Fuel Cell
Operating Mode"])))
else
plot(time2, State2, '-
', 'LineWidth', 1.25, 'Color', [0 0.5 1])
plot(time1, State1, 'r-.',
'LineWidth', 1.25)
set(gca, 'fontsize', 14)
title("Fuel Cell Operating
Mode", 'FontSize', 22)
legend('OM', 'GT', 'FontSize',
16)
end
xlim([0, 35])
ylim([0, 5])
xlabel('Time [s]', 'FontSize', 16)
ylabel('Fuel Cell Mode',
'FontSize', 16)
end
end
```

ANNEX C – Some BEV Results Comparisons

BEV Fuel Economy Tests

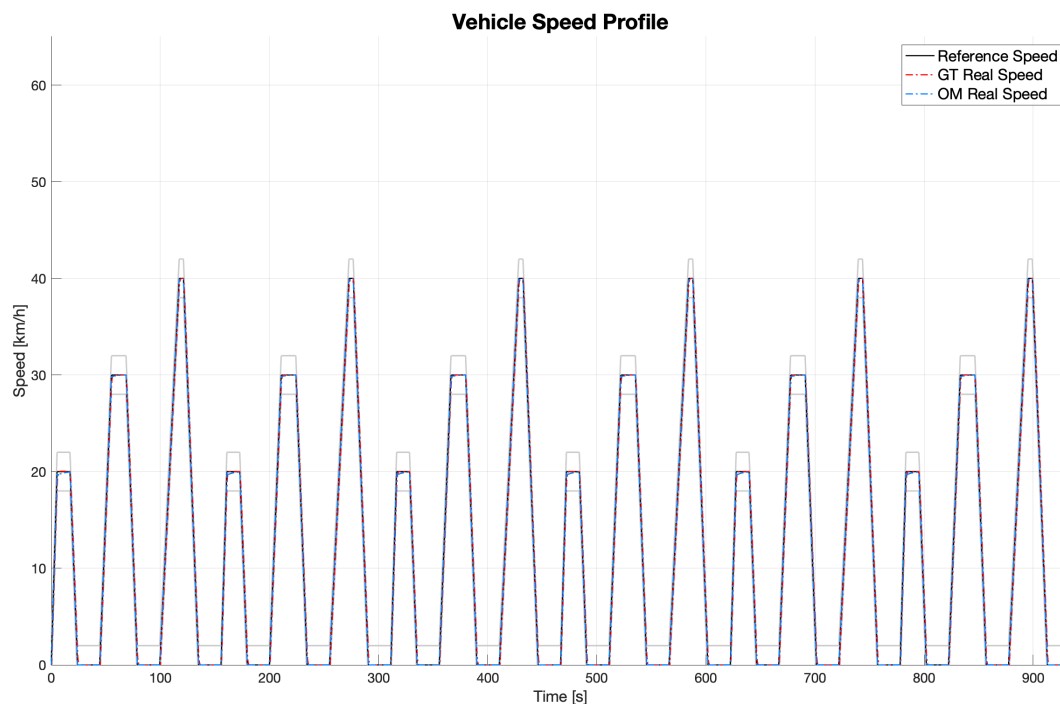


Figure C.01 – Comparison of BEV Speed Profiles
(SORT1, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7\text{ kW}$)

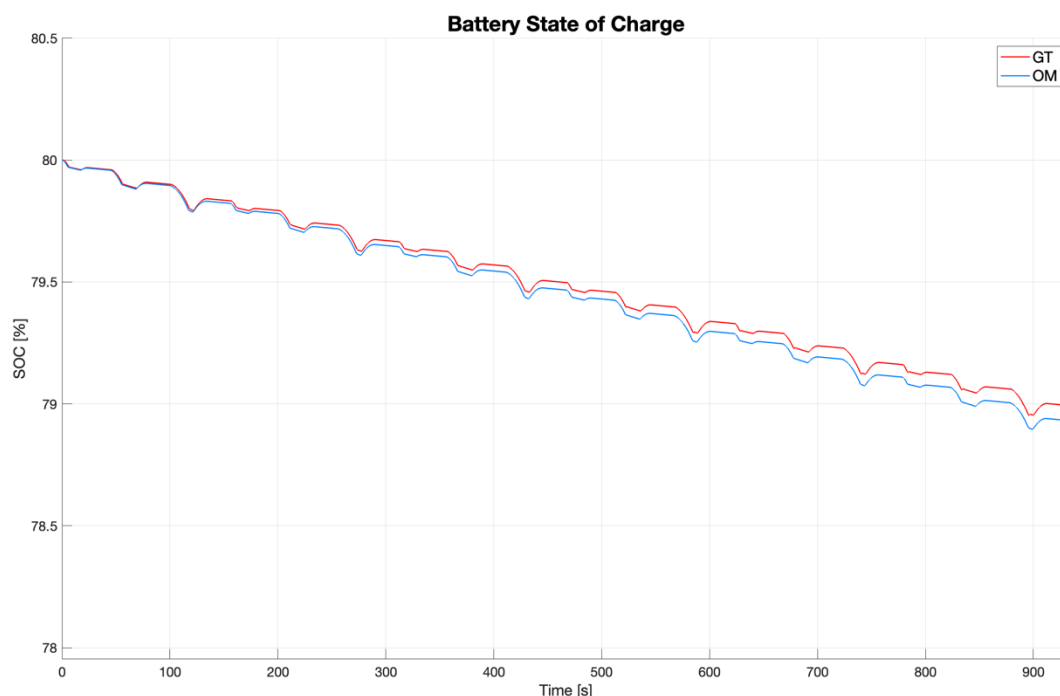


Figure C.02 – Comparison of BEV SOC Discharge Profiles
(SORT1, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7\text{ kW}$)

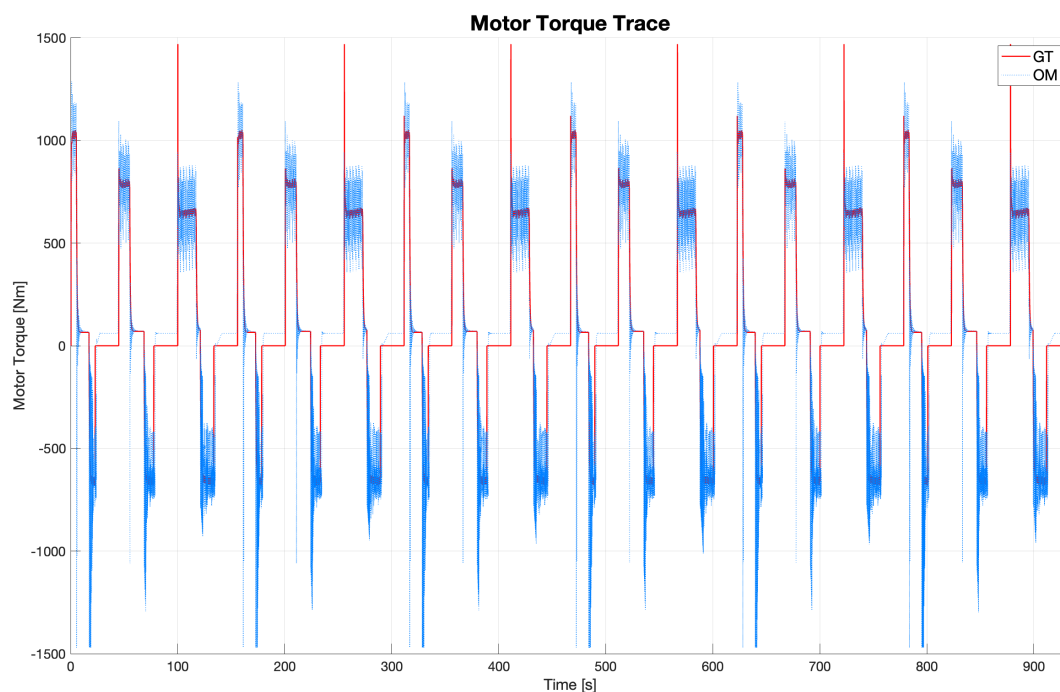


Figure C.03 – Comparison of BEV Torque Trace
(SORT1, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW)

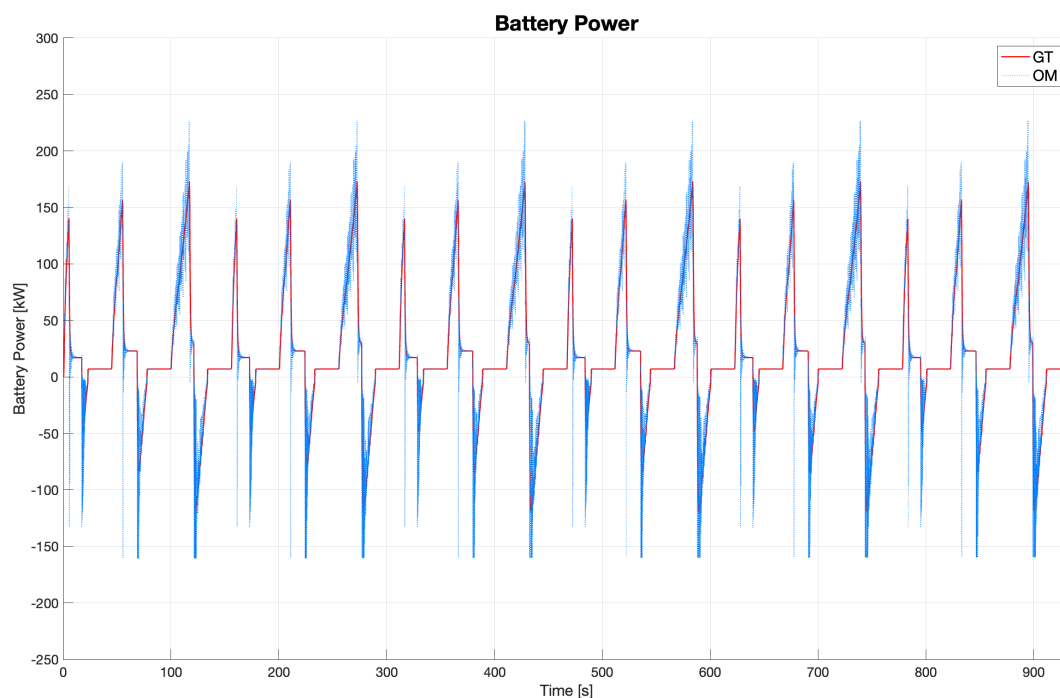


Figure C.04 – Comparison of BEV Battery Power
(SORT1, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW)

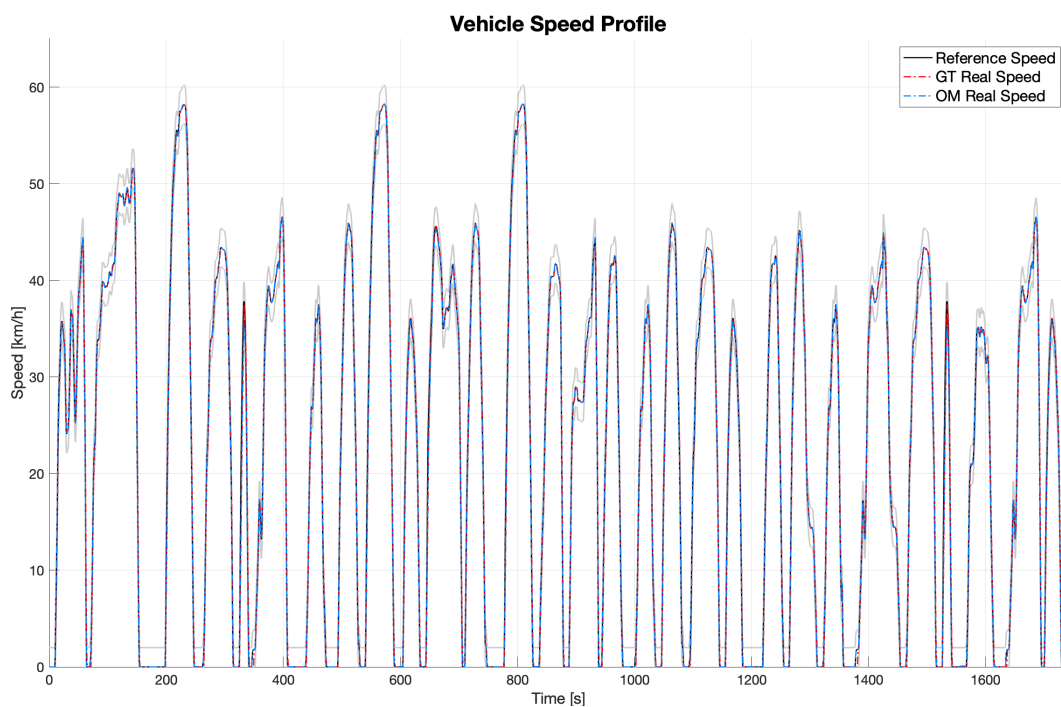


Figure C.05 – Comparison of BEV Speed Profiles
(Braunschweig, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7\text{ kW}$)

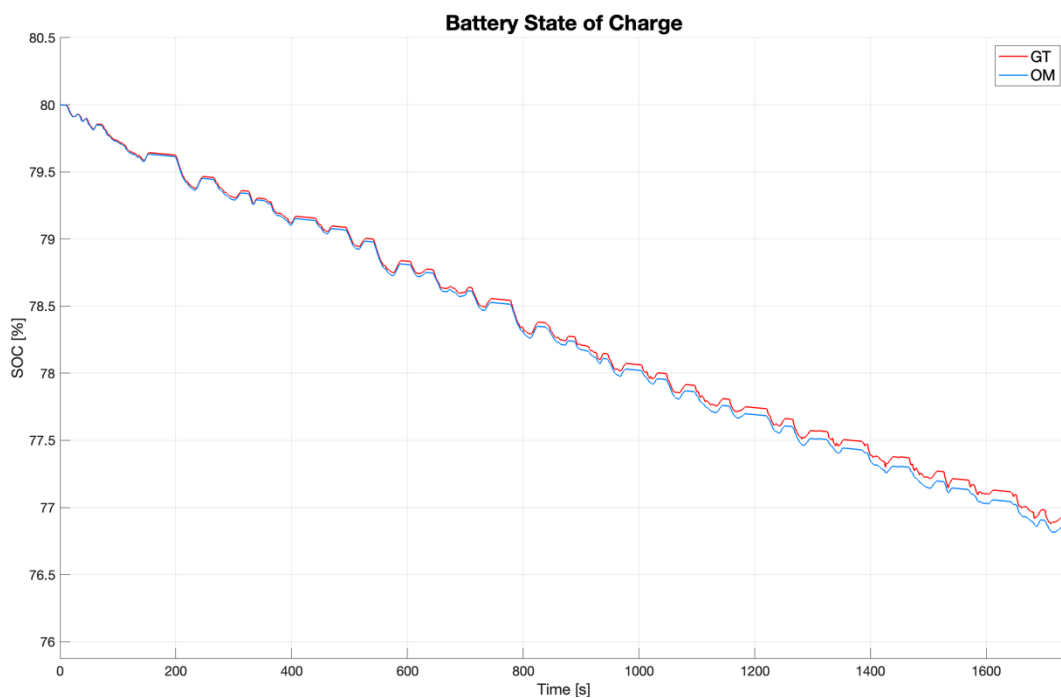


Figure C.06 – Comparison of BEV SOC Discharge Profiles
(Braunschweig, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7\text{ kW}$)

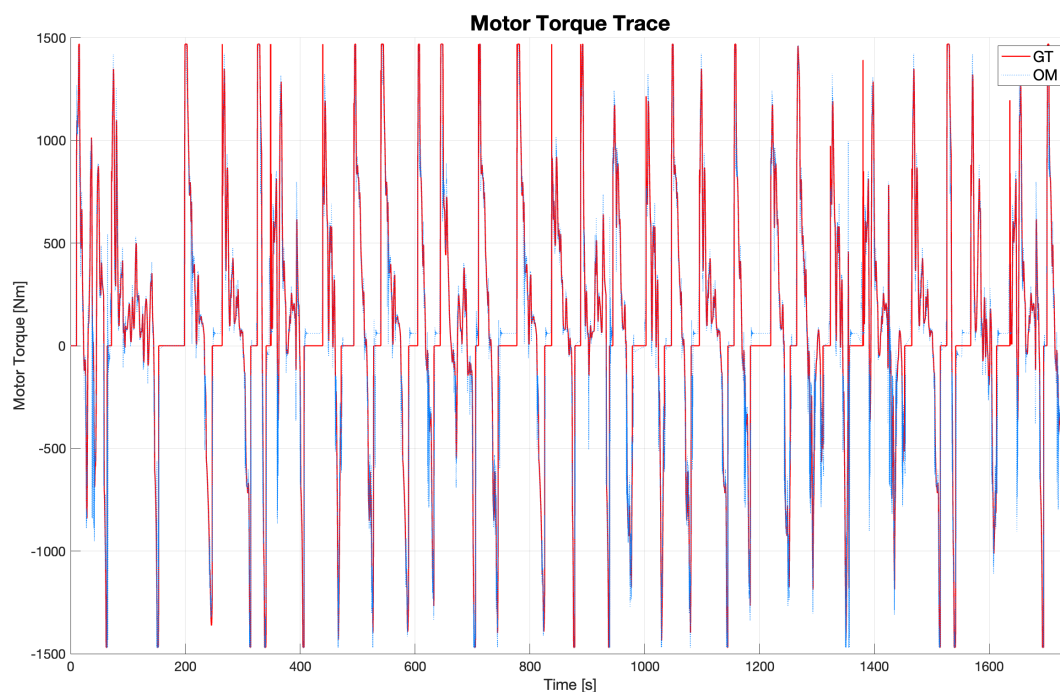


Figure C.07 – Comparison of BEV Torque Trace
(Braunschweig, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7\text{ kW}$)

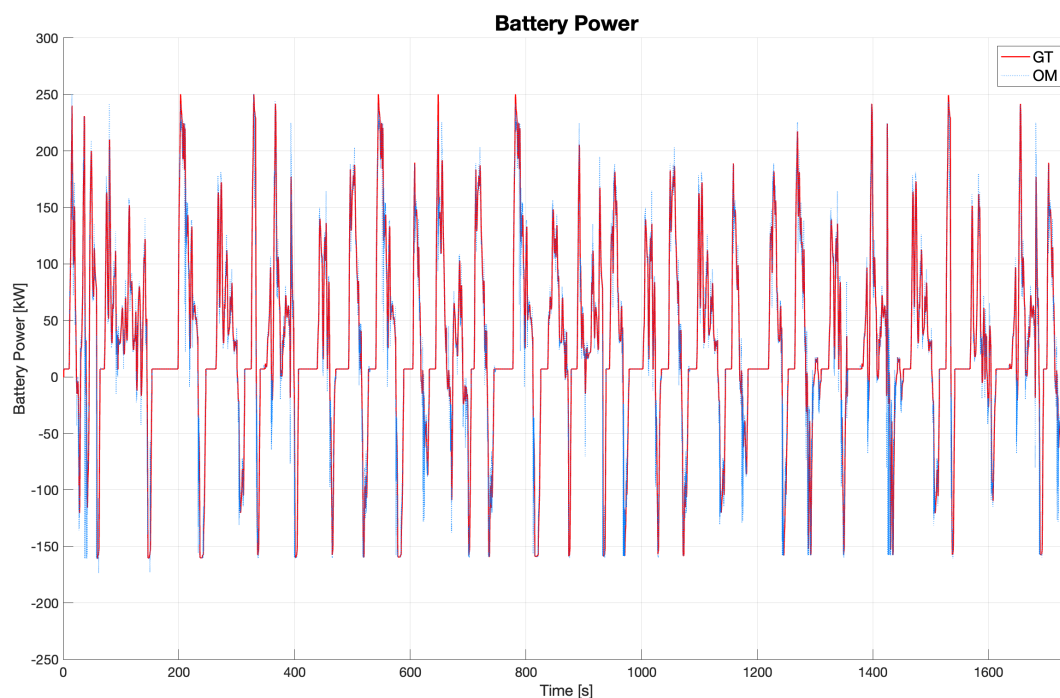


Figure C.08 – Comparison of BEV Battery Power
(Braunschweig, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7\text{ kW}$)

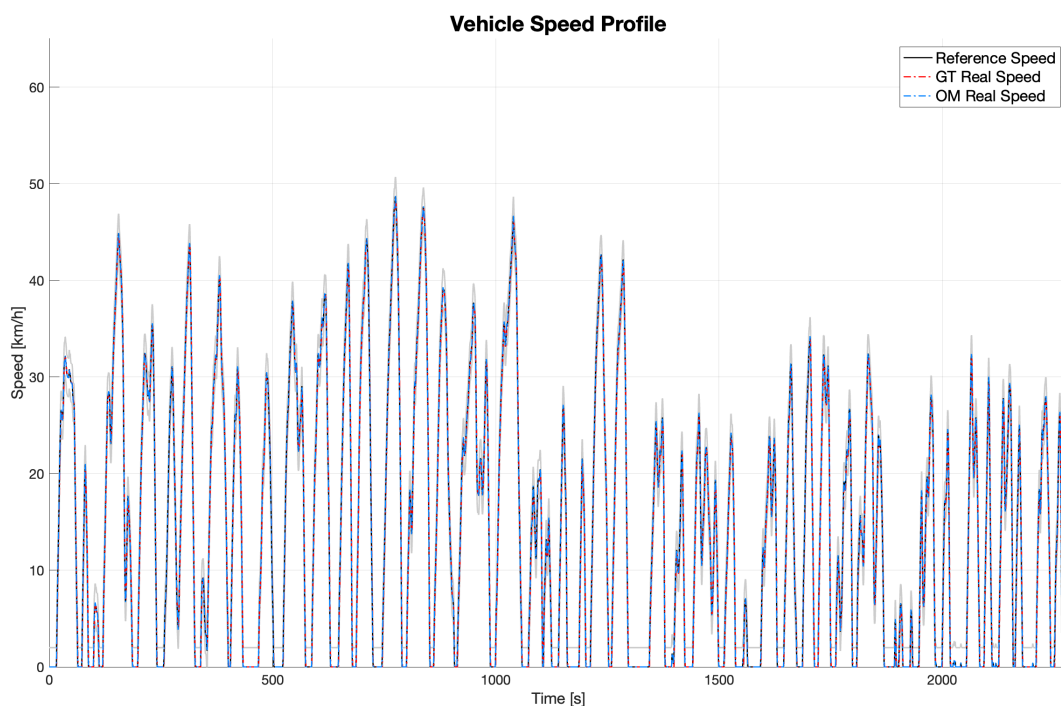


Figure C.09 – Comparison of BEV Speed Profiles
(MLTB, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7\text{ kW}$)

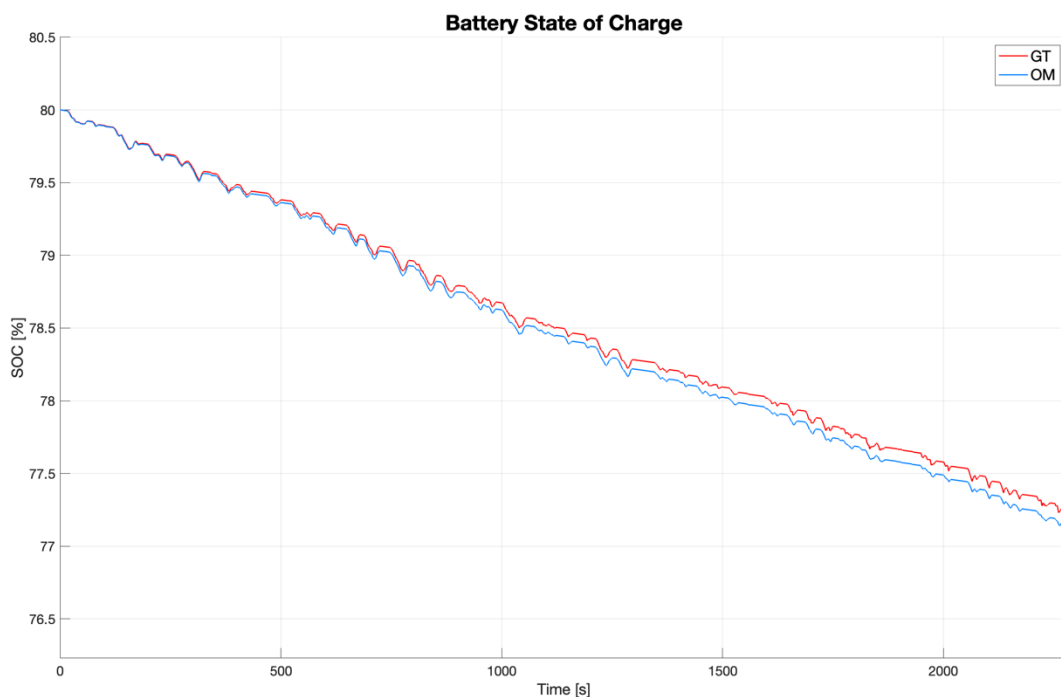


Figure C.10 – Comparison of BEV SOC Discharge Profiles
(MLTB, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7\text{ kW}$)

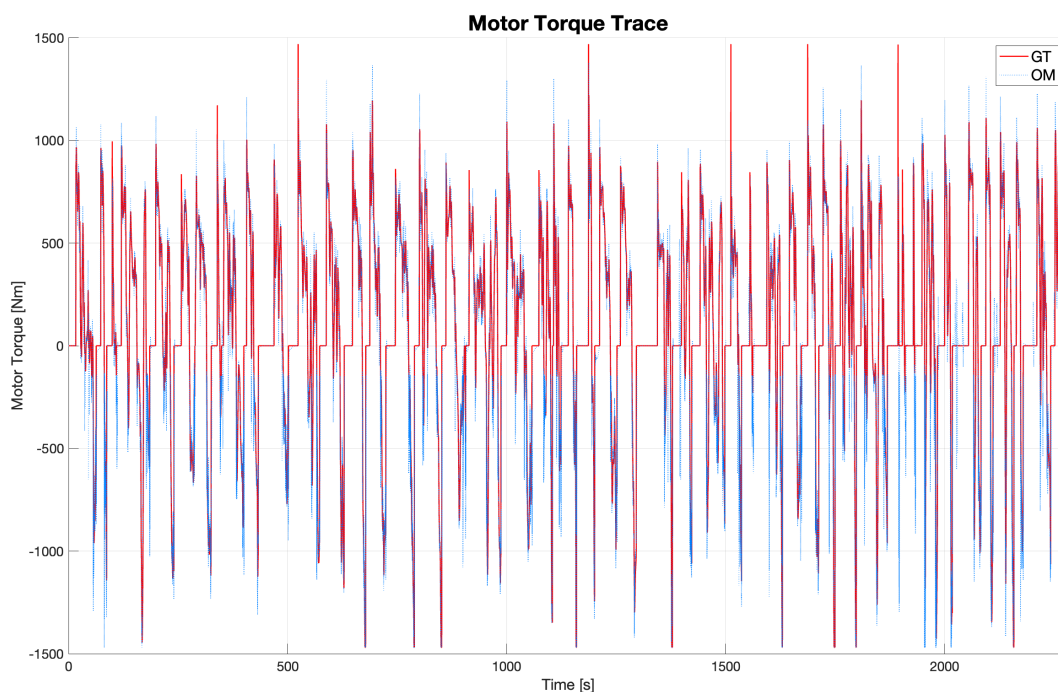


Figure C.11 – Comparison of BEV Torque Trace
(MLTB, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7\text{ kW}$)

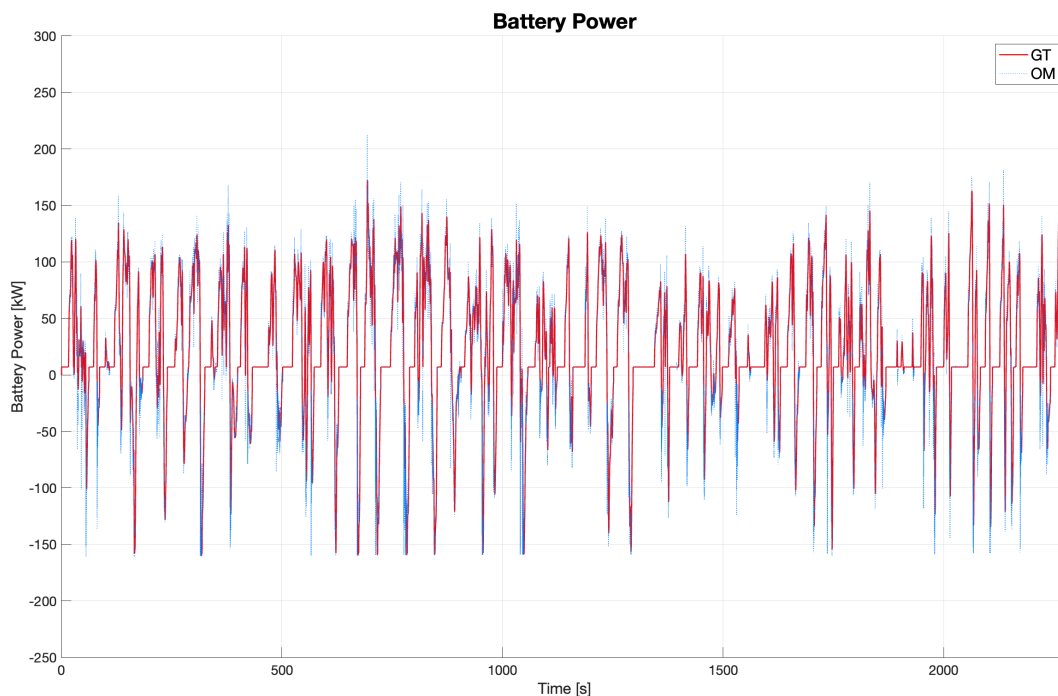


Figure C.12 – Comparison of BEV Battery Power
(MLTB, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7\text{ kW}$)

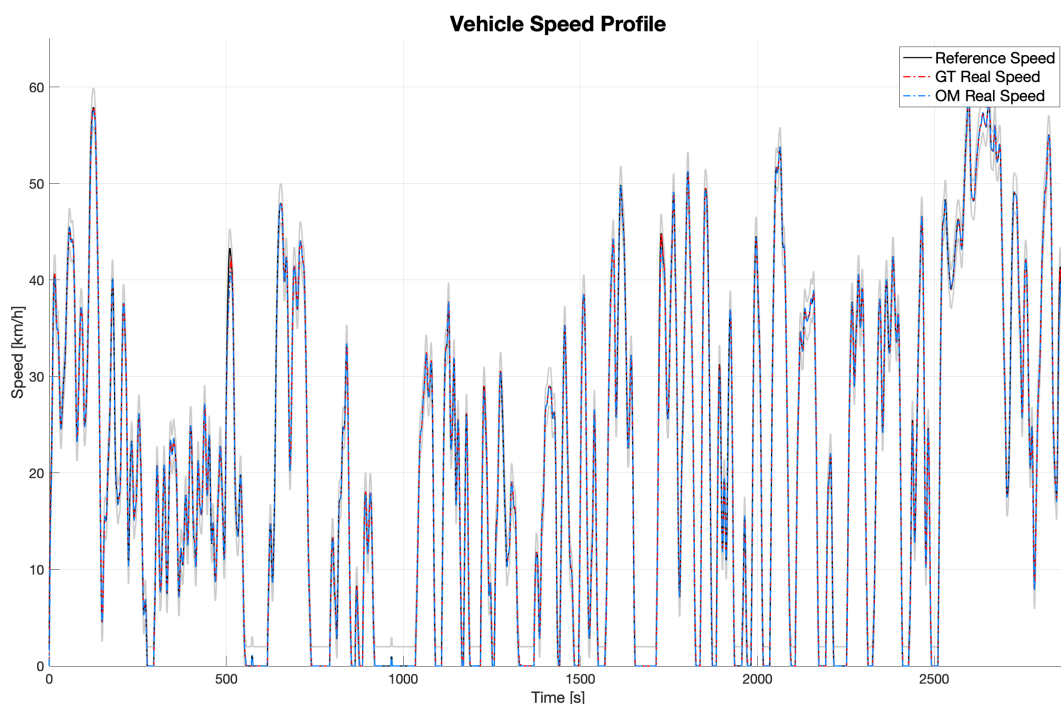


Figure C.13 – Comparison of BEV Speed Profiles
(RDC #1, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7\text{ kW}$)

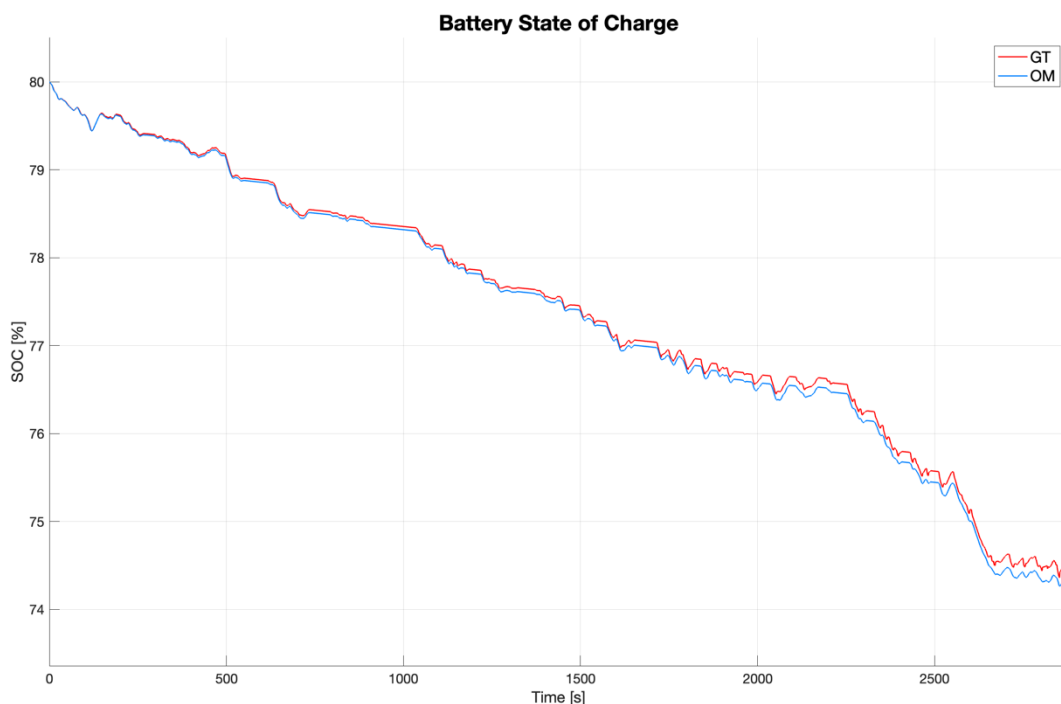


Figure C.14 – Comparison of BEV SOC Discharge Profiles
(RDC #1, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7\text{ kW}$)

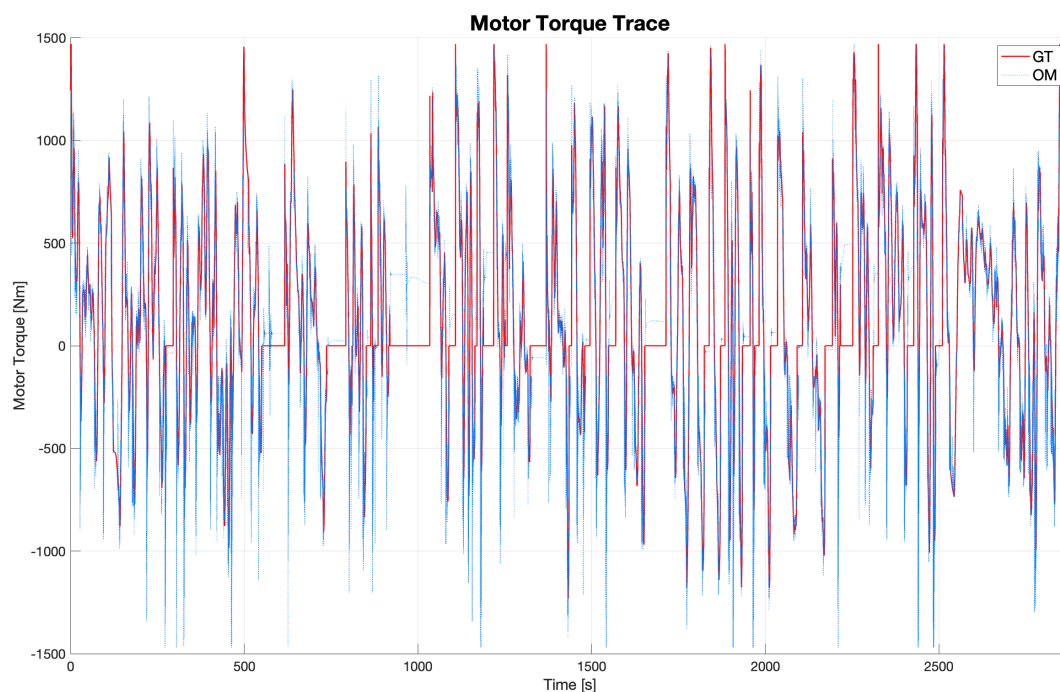


Figure C.15 – Comparison of BEV Torque Trace
(RDC #1, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW)

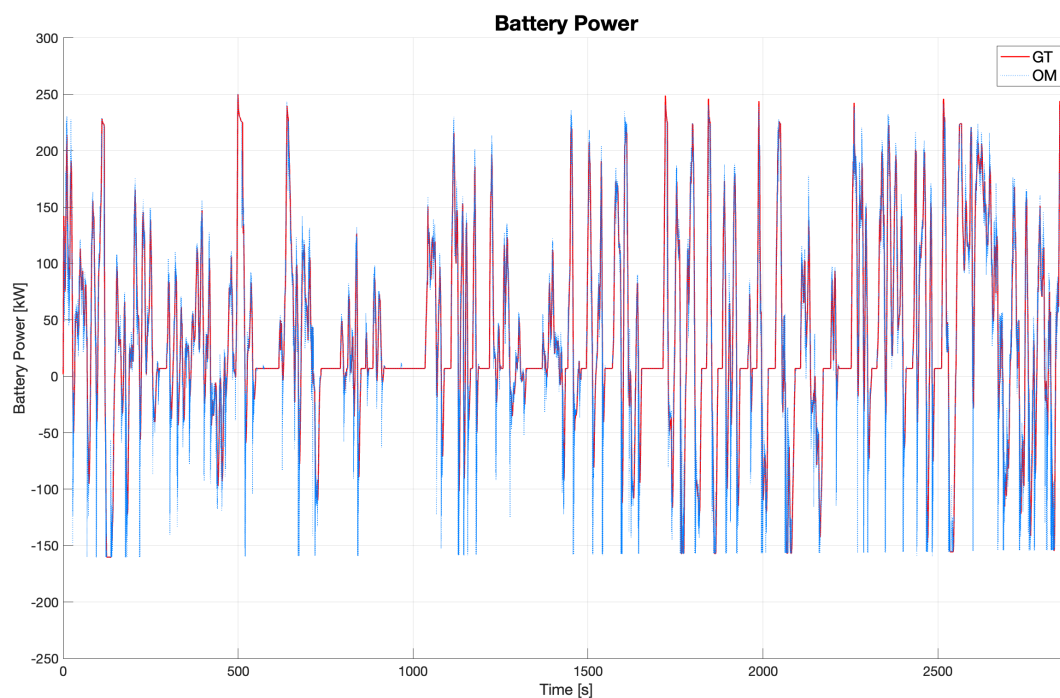


Figure C.16 – Comparison of BEV Battery Power
(RDC #1, 19 000 kg, $SOC_0 = 80\%$ & $P_{aux} = 7$ kW)

BEV Vehicle Acceleration/Slope Tests

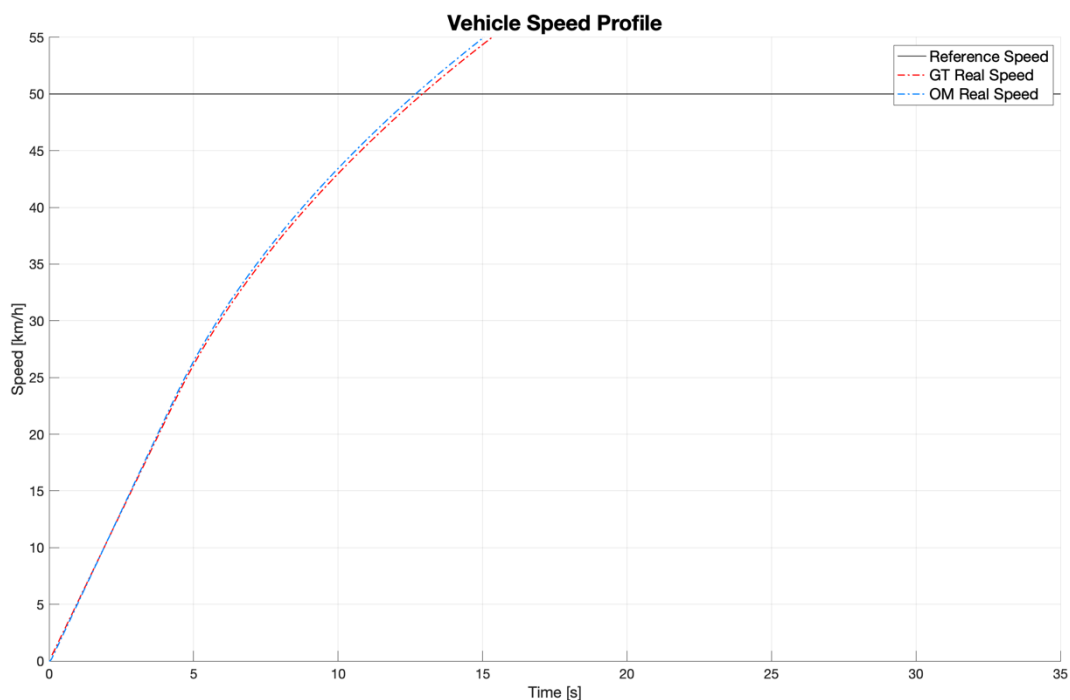


Figure C.17 – Comparison of BEV Acceleration
(Case #9, 19 000 kg, 0 % inclination, $SOC_0 = 85\%$ & $P_{aux} = 7.5$ kW)

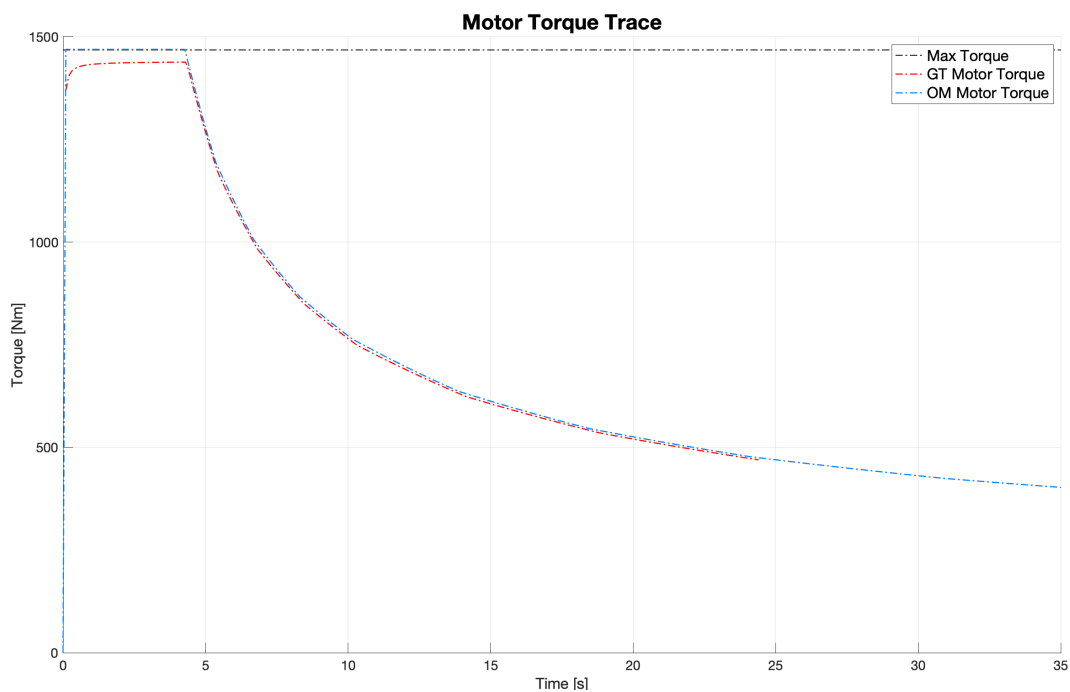


Figure C.18 – Comparison of BEV Torque Trace
(Case #9, 19 000 kg, 0 % inclination, $SOC_0 = 85\%$ & $P_{aux} = 7.5$ kW)

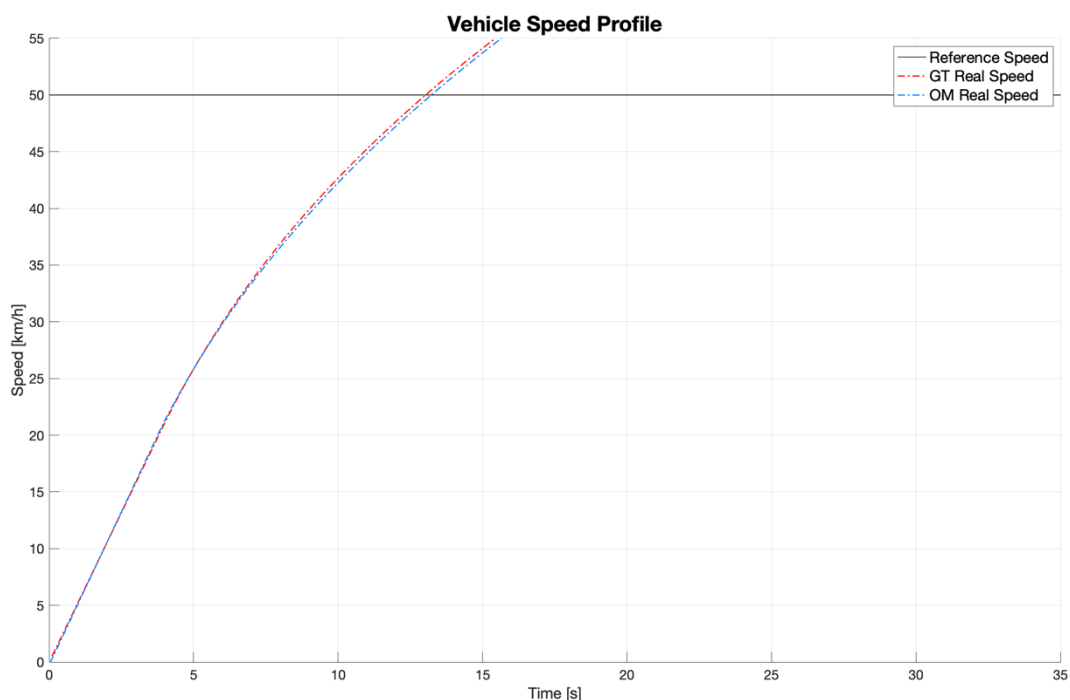


Figure C.19 – Comparison of BEV Acceleration
(Case #11, 19 000 kg, 0 % inclination, $SOC_0 = 65\%$ & $P_{aux} = 7.5$ kW)

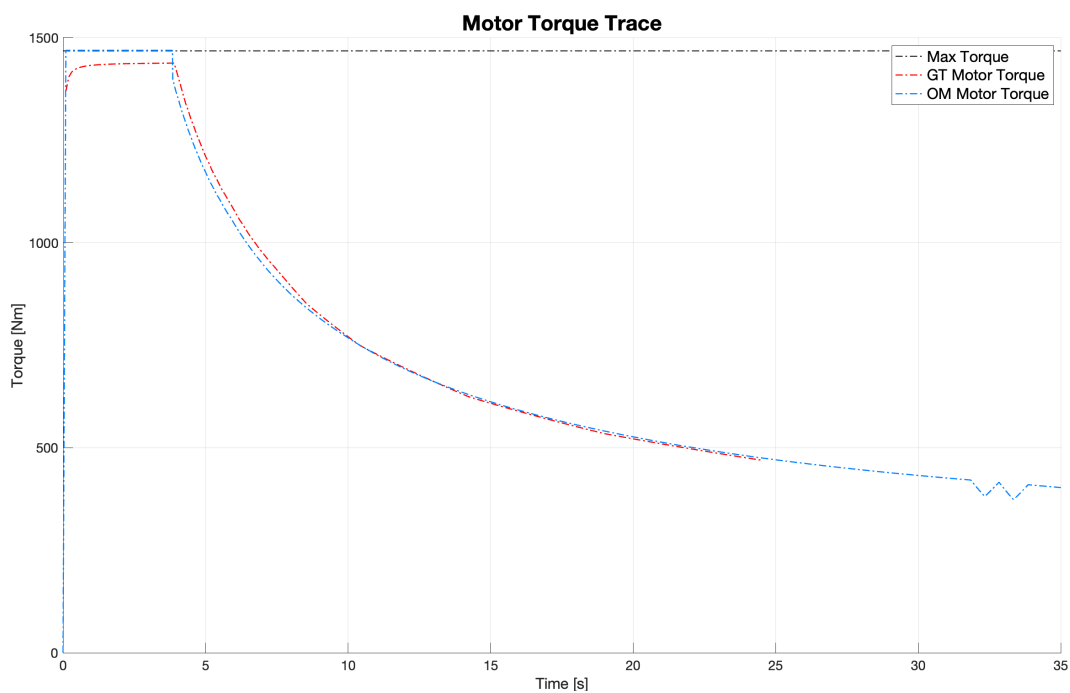


Figure C.20 – Comparison of BEV Torque Trace
(Case #11, 19 000 kg, 0 % inclination, $SOC_0 = 65\%$ & $P_{aux} = 7.5$ kW)

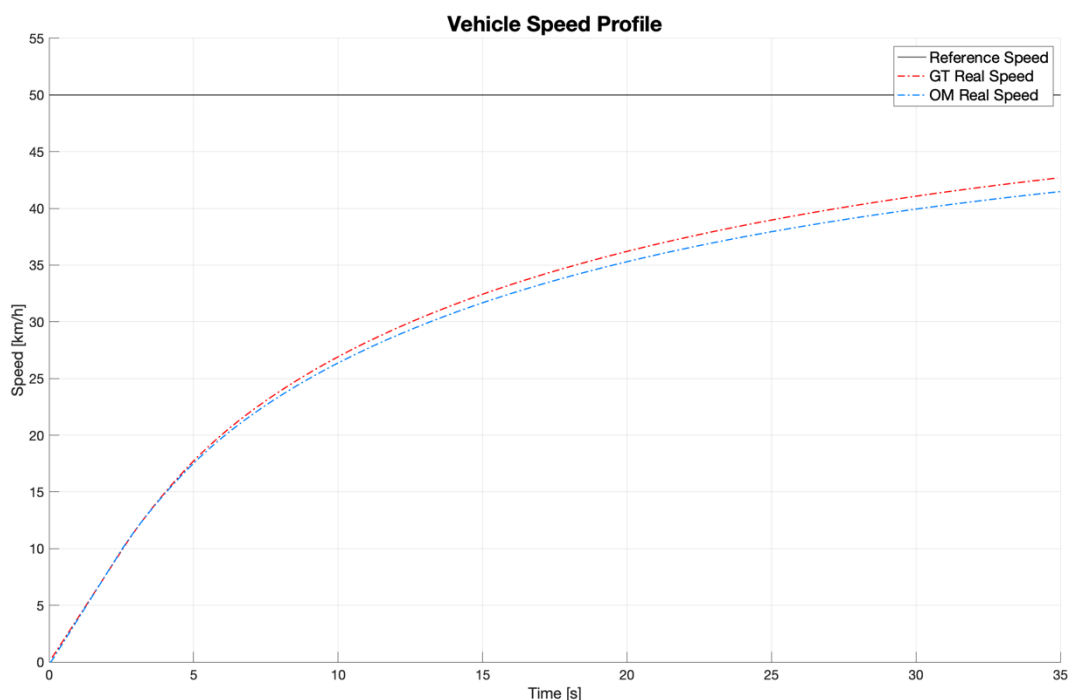


Figure C.21 – Comparison of BEV Acceleration
(Case #16, 19 000 kg, 4 % inclination, $SOC_0 = 25\%$ & $P_{aux} = 7.5$ kW)

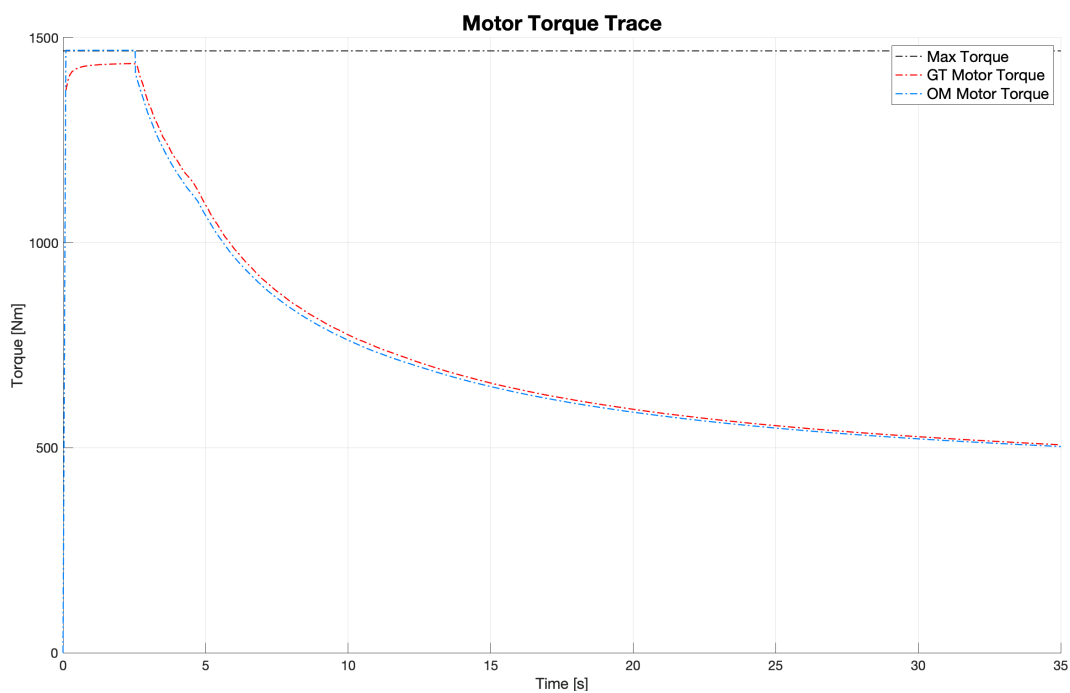


Figure C.22 – Comparison of BEV Torque Trace
(Case #16, 19 000 kg, 4 % inclination, $SOC_0 = 25\%$ & $P_{aux} = 7.5$ kW)

ANNEX D – Some FCH Results Comparisons

FCH Fuel Economy Tests ($P_{\text{aux}} = 7 \text{ kW}$)

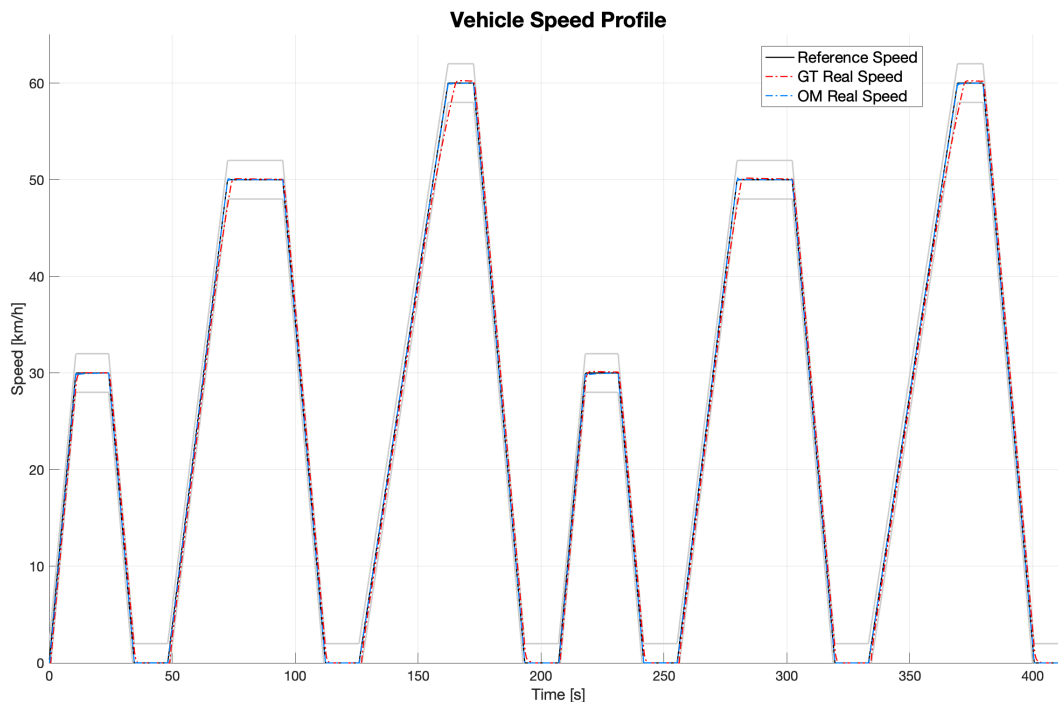


Figure D.01 – Comparison of FCH Speed Profiles
(SORT3, 19 000 kg, $\text{SOC}_0 = 62 \%$ & $P_{\text{aux}} = 7 \text{ kW}$)

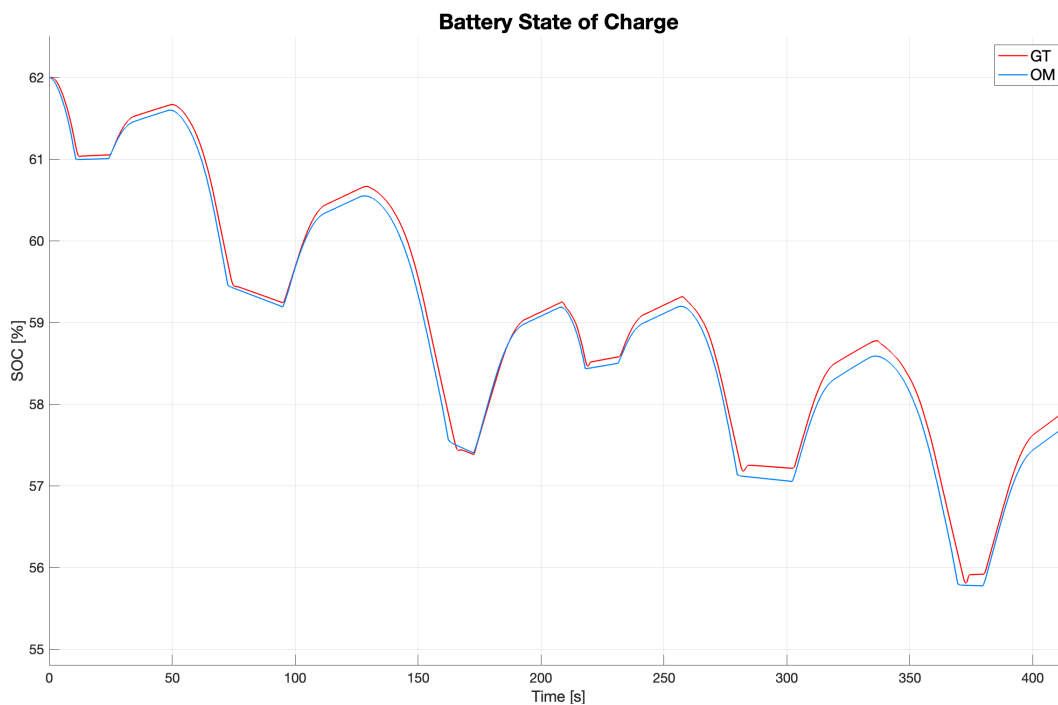


Figure D.02 – Comparison of FCH SOC Discharge Profiles
(SORT3, 19 000 kg, $\text{SOC}_0 = 62 \%$ & $P_{\text{aux}} = 7 \text{ kW}$)

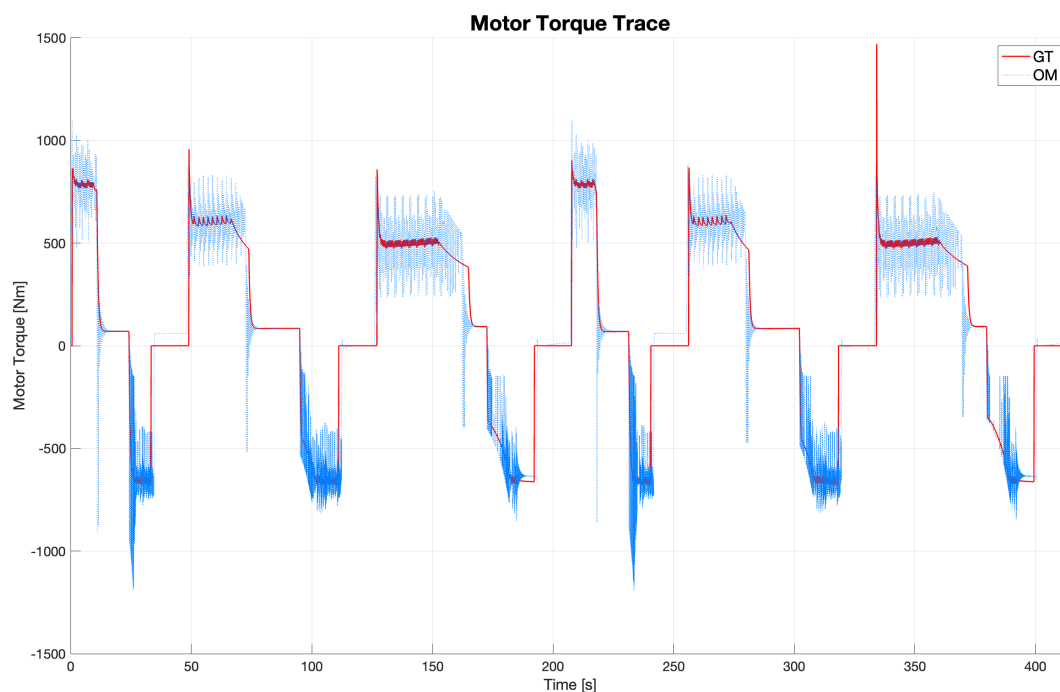


Figure D.03 – Comparison of FCH Torque Trace
(SORT3, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7\text{ kW}$)

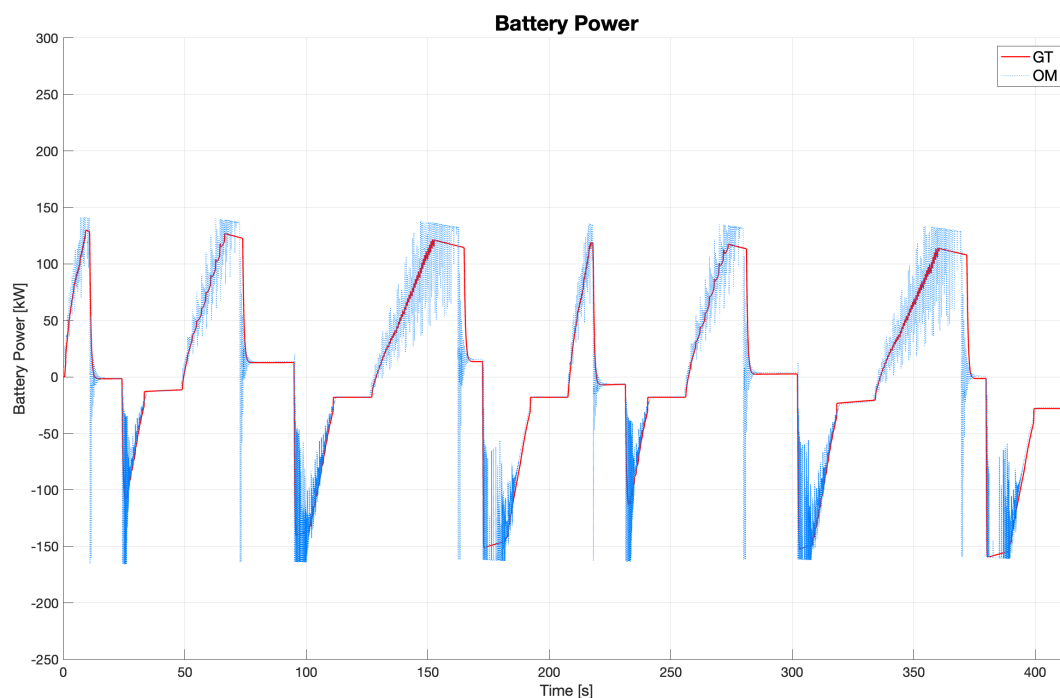


Figure D.04 – Comparison of FCH Battery Power
(SORT3, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7\text{ kW}$)

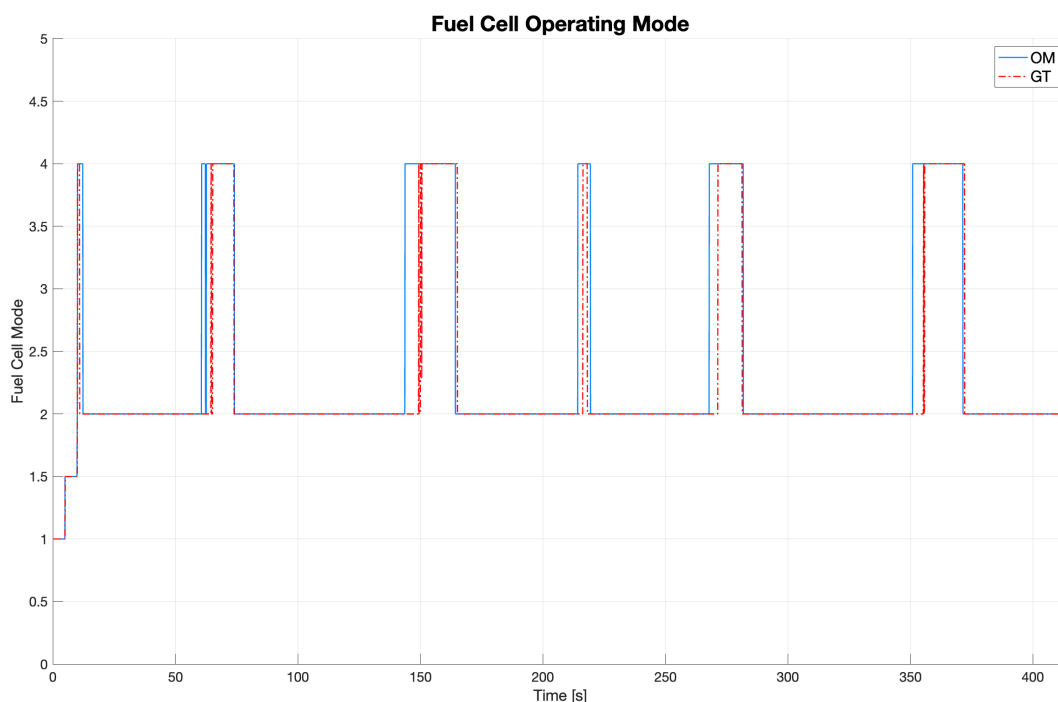


Figure D.05 – Comparison of FCH Hybridization Mode
(SORT3, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)

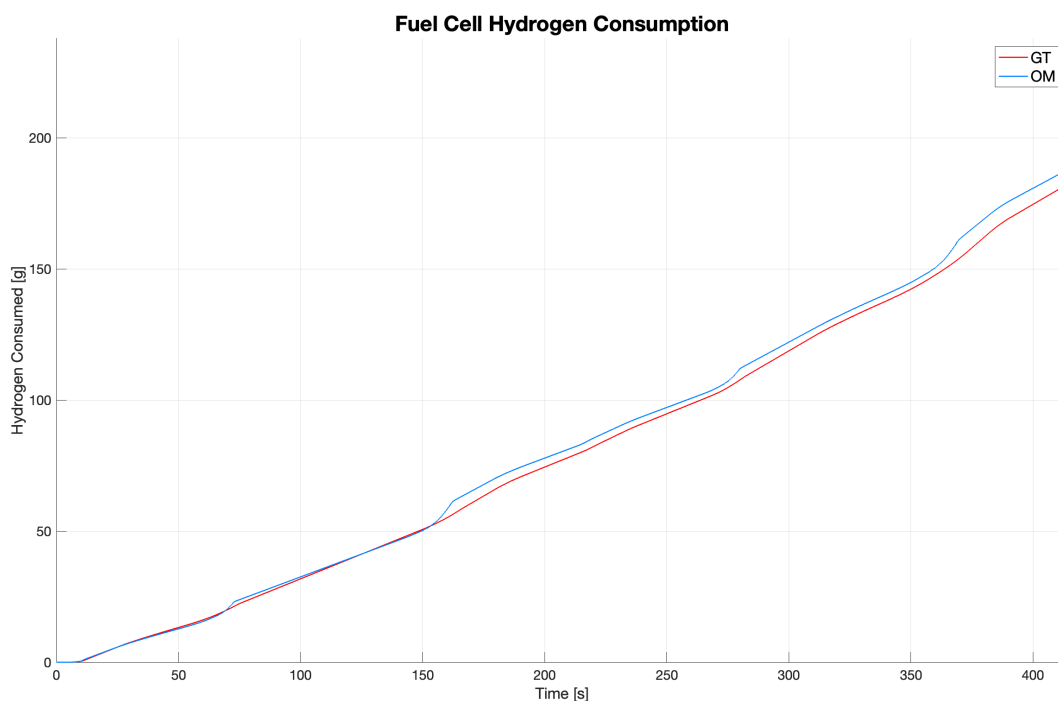


Figure D.06 – Comparison of FCH Hydrogen Consumed
(SORT3, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)

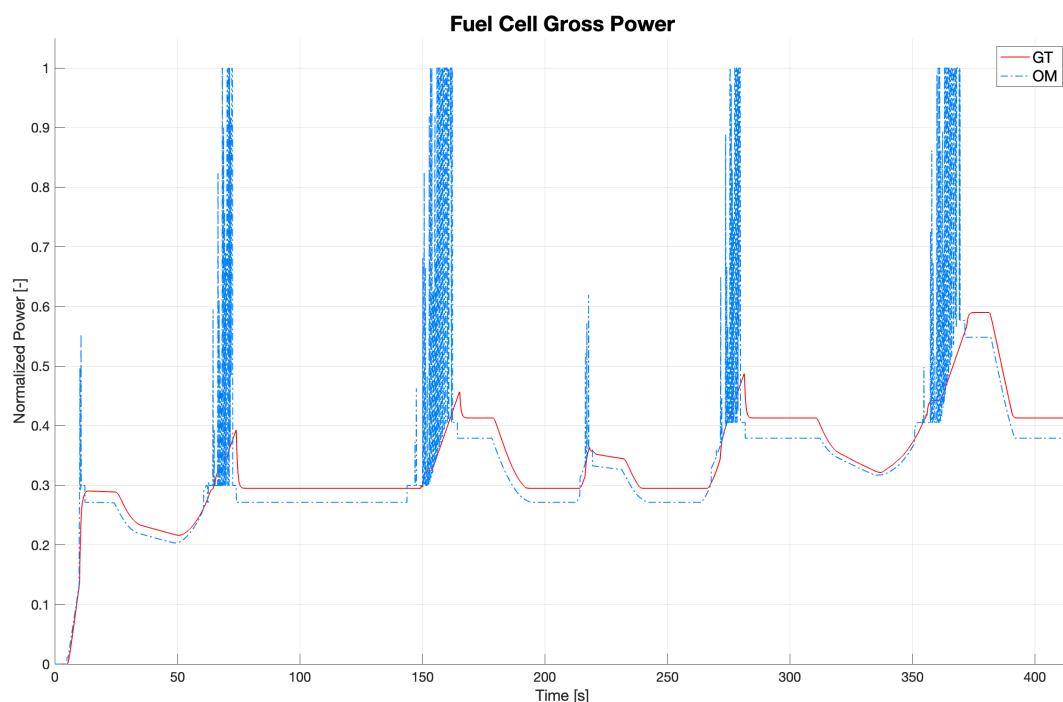


Figure D.07 – Comparison of FCH Fuel Cell Gross Power
(SORT3, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)



Figure D.08 – Comparison of FCH Speed Profiles
(Braunschweig, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)

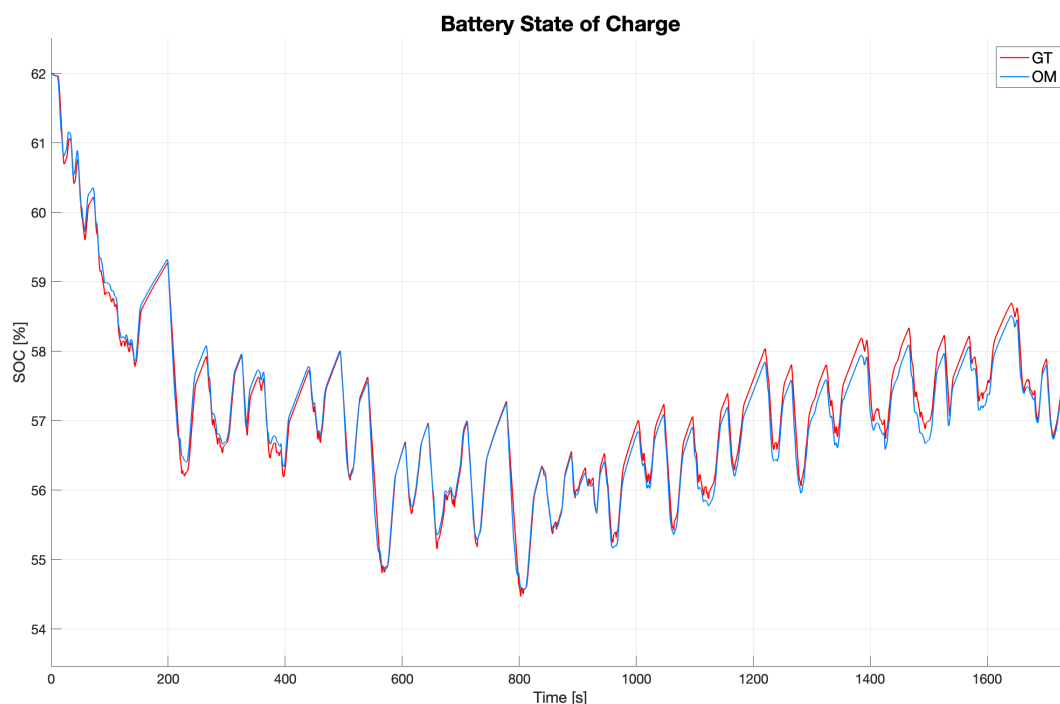


Figure D.09 – Comparison of FCH SOC Discharge Profiles
(Braunschweig, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7\text{ kW}$)

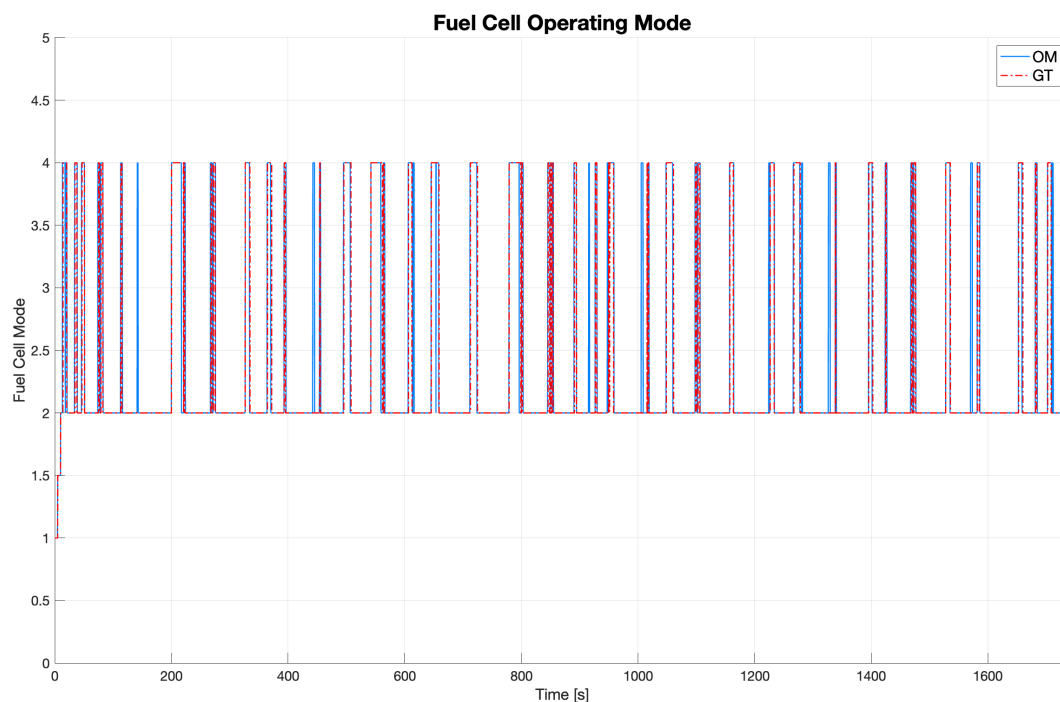


Figure D.10 – Comparison of FCH Hybridization Mode
(Braunschweig, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7\text{ kW}$)

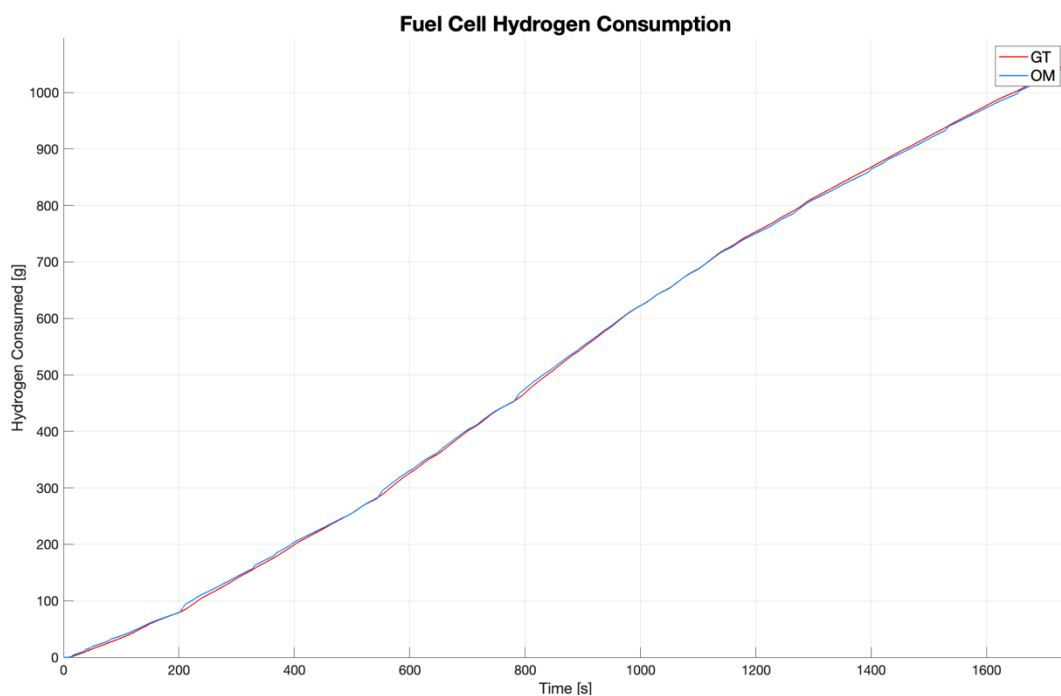


Figure D.11 – Comparison of FCH Hydrogen Consumed
(Braunschweig, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)

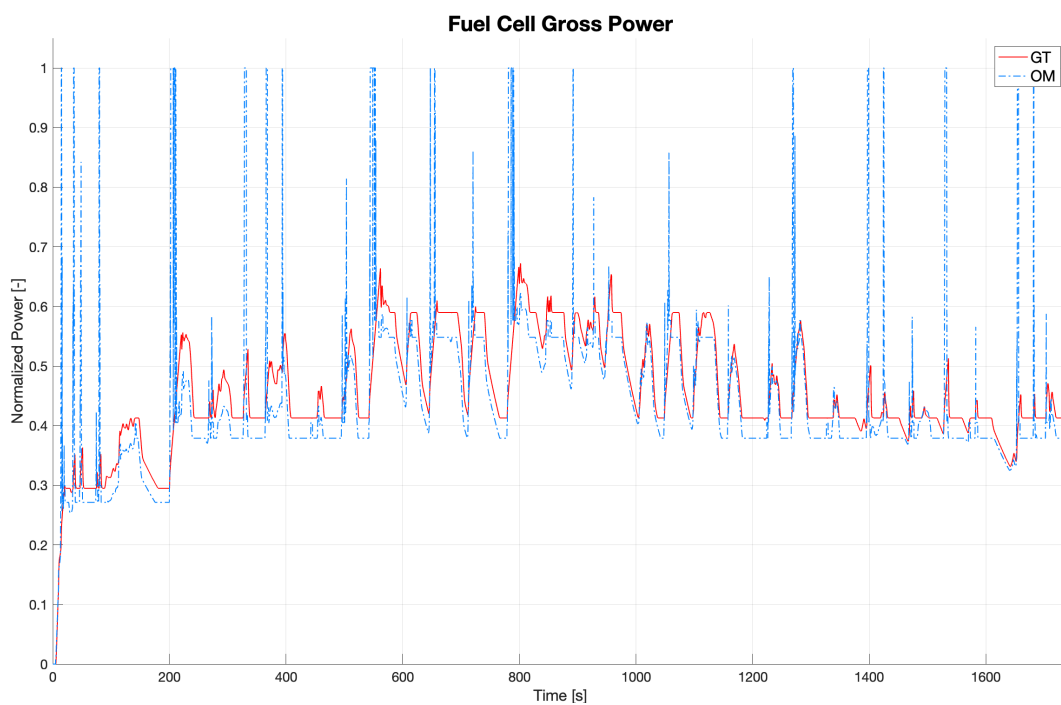


Figure D.12 – Comparison of FCH Fuel Cell Gross Power
(Braunschweig, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)

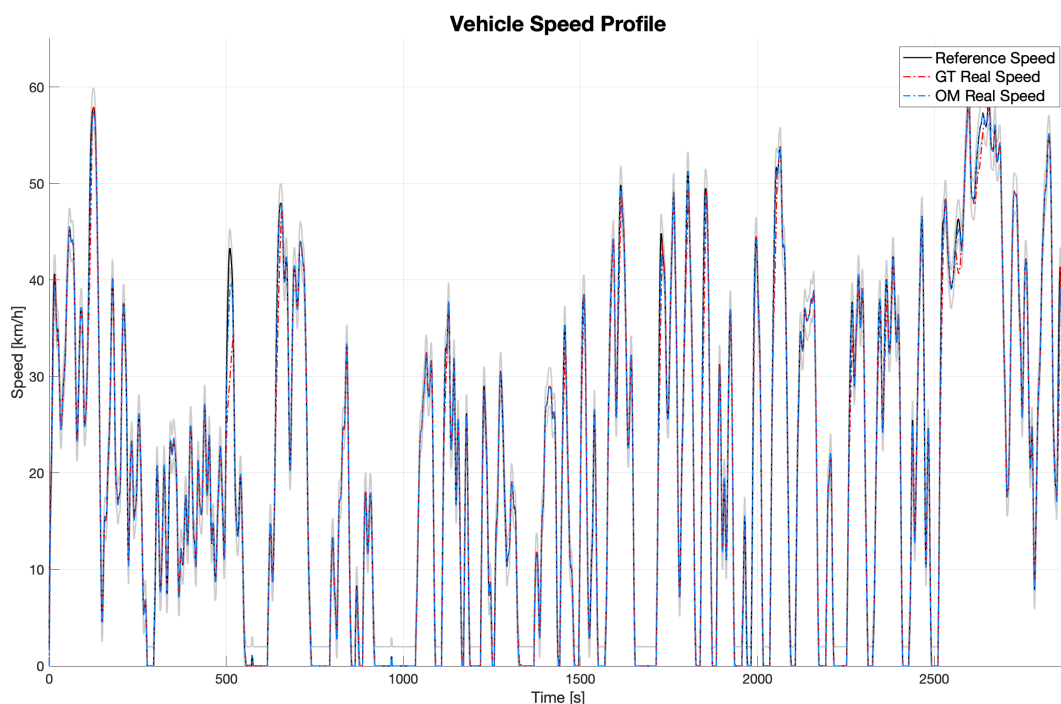


Figure D.13 – Comparison of FCH Speed Profiles
(RDC #1, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7\text{ kW}$)

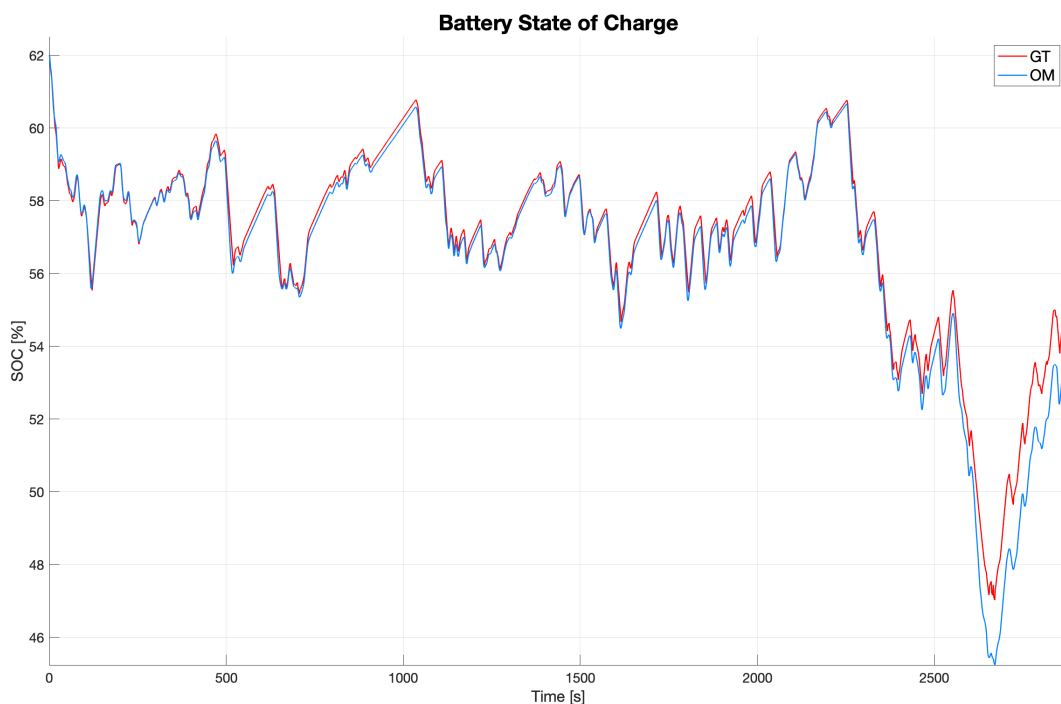


Figure D.14 – Comparison of FCH SOC Discharge Profiles
(RDC #1, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7\text{ kW}$)

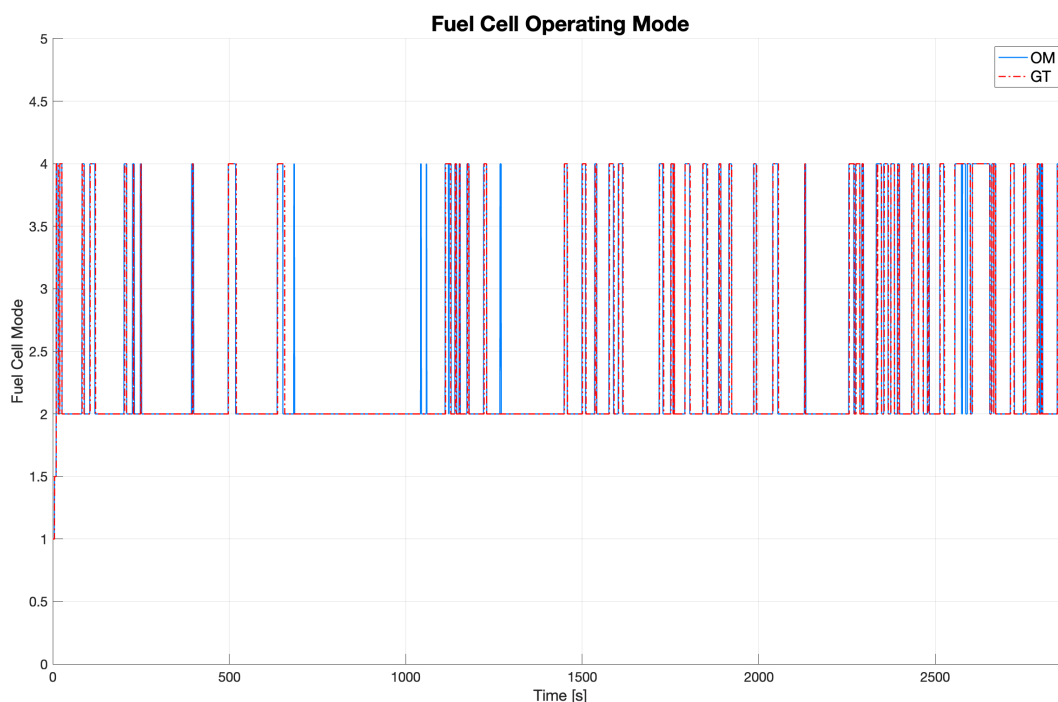


Figure D.15 – Comparison of FCH Hybridization Mode
(RDC #1, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)

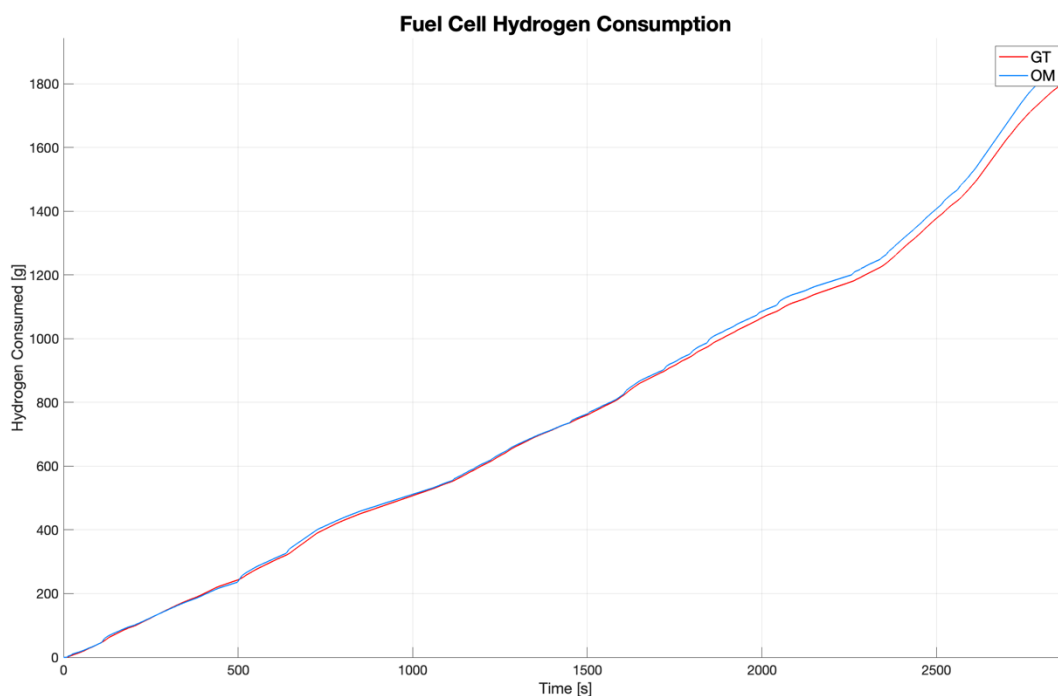


Figure D.16 – Comparison of FCH Hydrogen Consumed
(RDC #1, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)

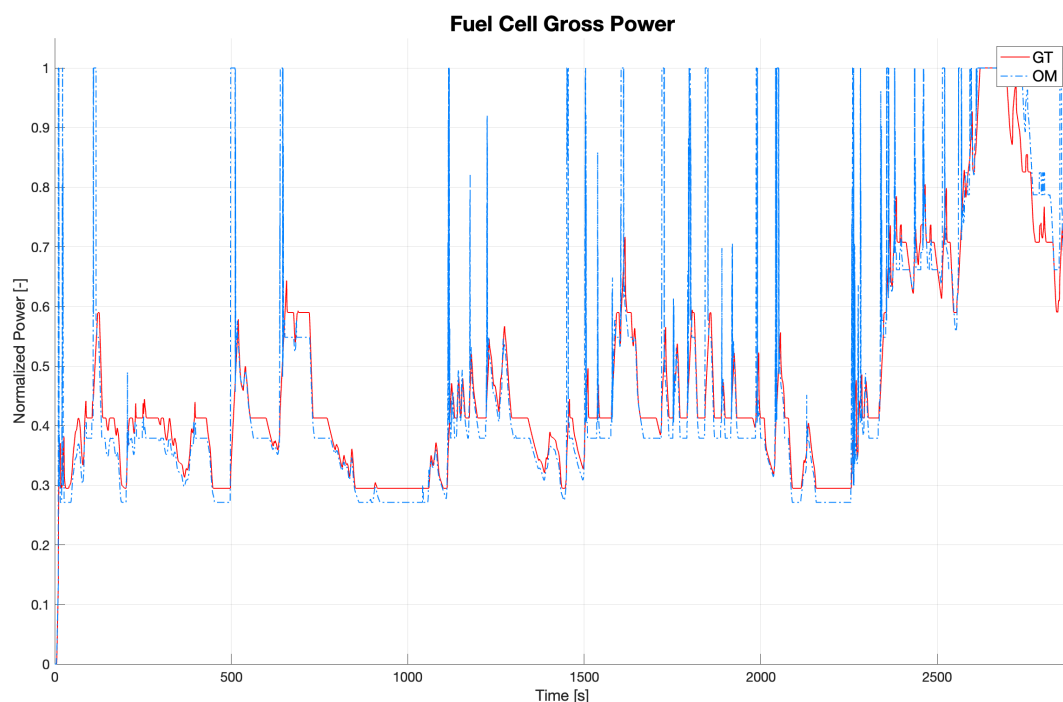


Figure D.17 – Comparison of FCH Fuel Cell Gross Power
(RDC #1, 19 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 7$ kW)

FCH Fuel Economy Tests ($P_{aux} = 0$ kW)

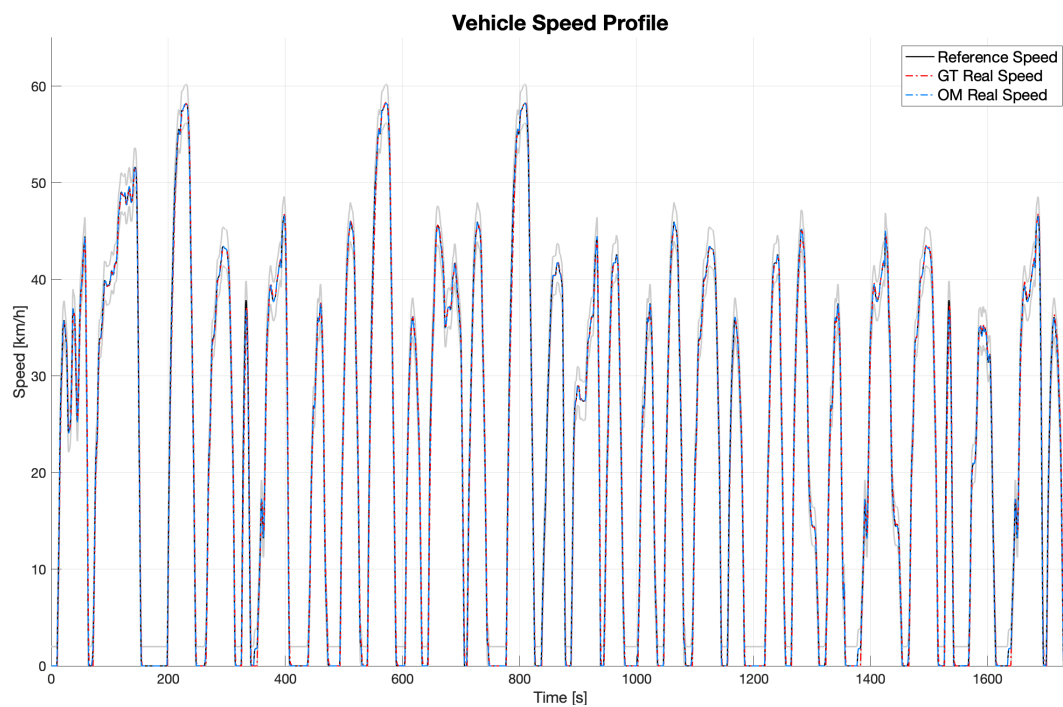


Figure D.18 – Comparison of FCH Speed Profiles
(Braunschweig, 16 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 0$ kW)

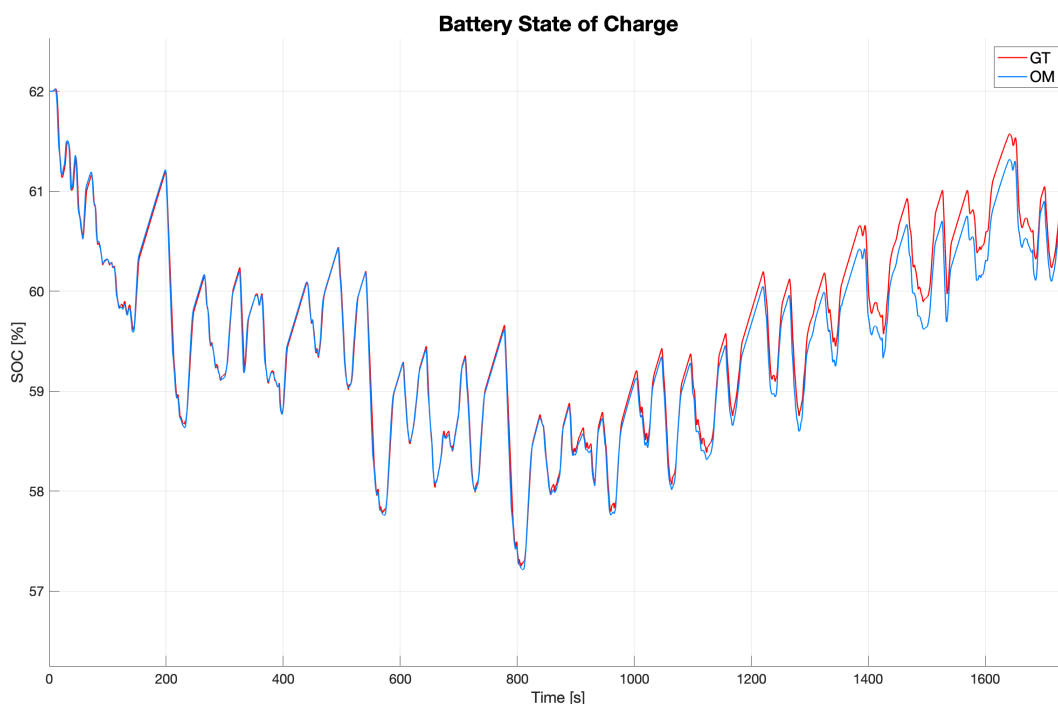


Figure D.19 – Comparison of FCH SOC Discharge Profiles
(Braunschweig, 16 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 0$ kW)

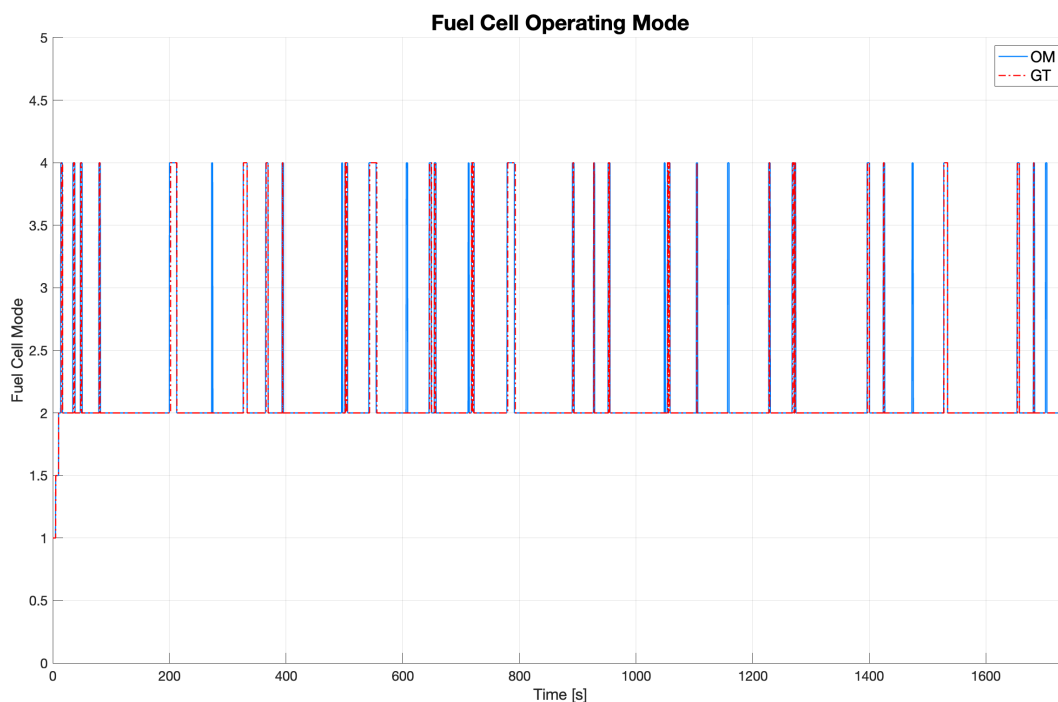


Figure D.20 – Comparison of FCH Hybridization Mode
(Braunschweig, 16 000 kg, $SOC_0 = 62\%$ & $P_{aux} = 0$ kW)

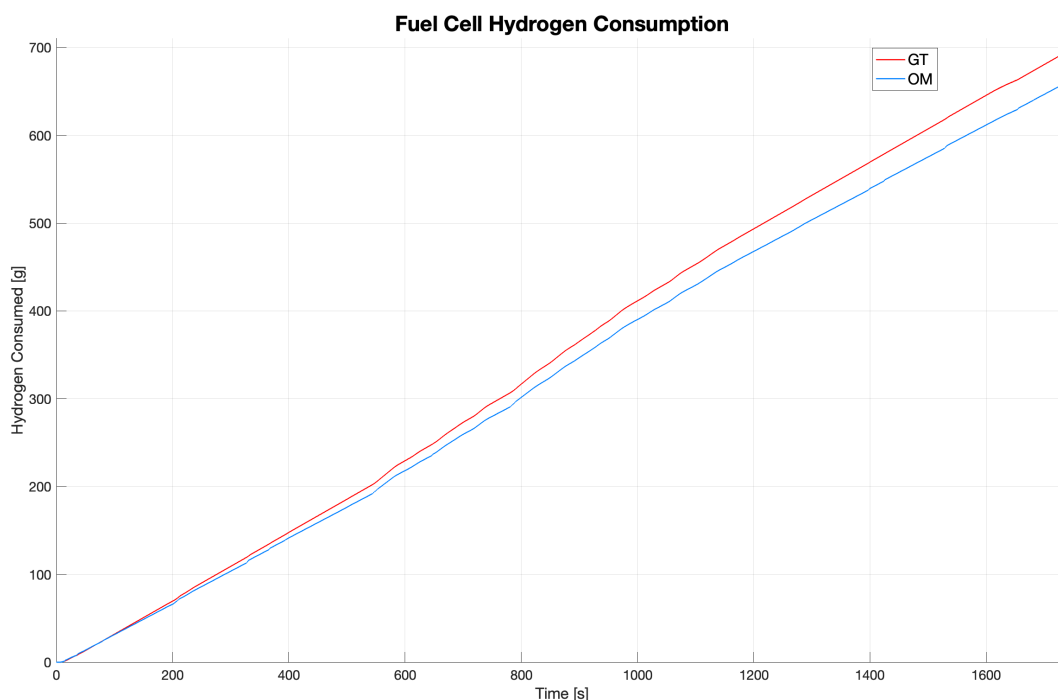


Figure D.21 – Comparison of FCH Hydrogen Consumed
(Braunschweig, 16 000 kg, SOC₀ = 62 % & P_{aux} = 0 kW)

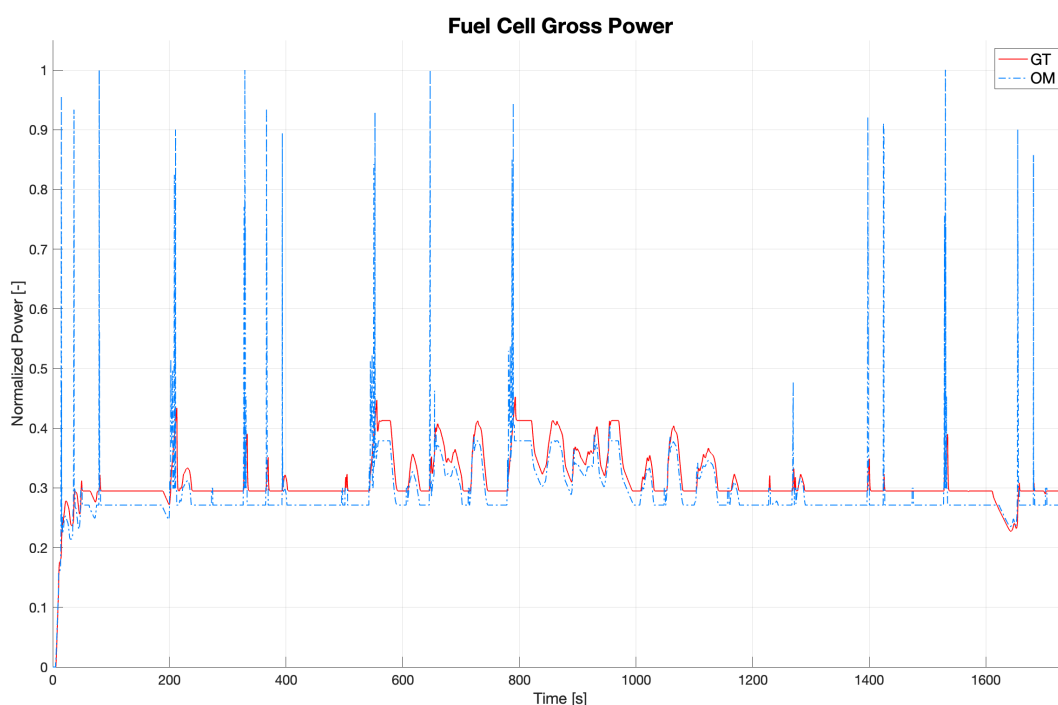


Figure D.22 – Comparison of FCH Fuel Cell Gross Power
(Braunschweig, 16 000 kg, SOC₀ = 62 % & P_{aux} = 0 kW)

FCH Vehicle Acceleration/Slope Tests

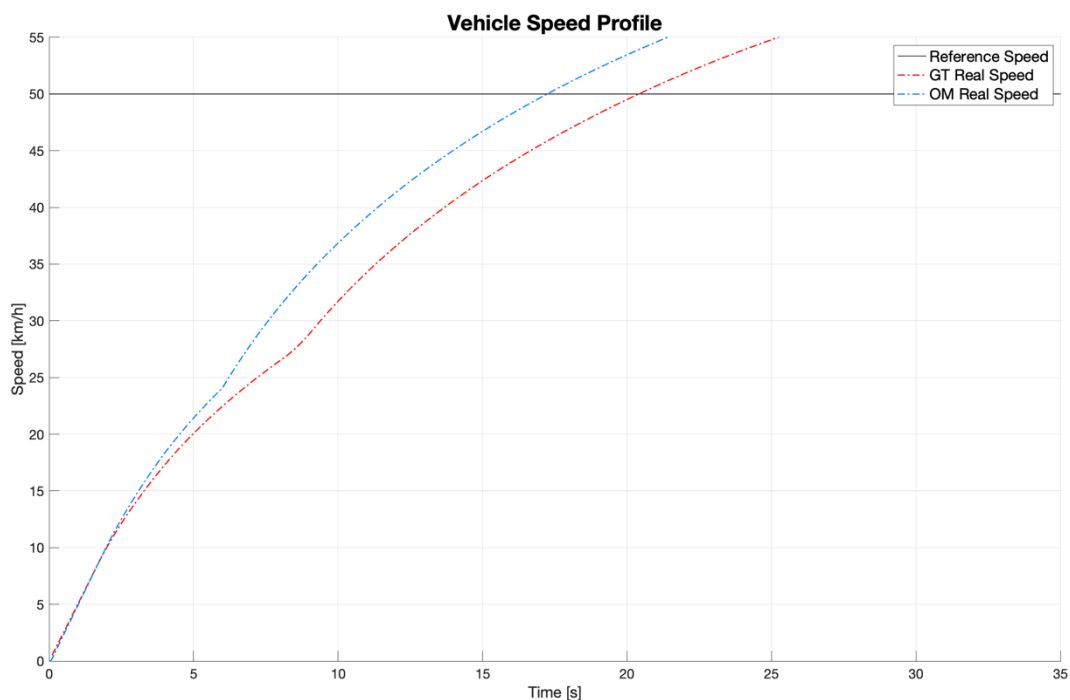


Figure D.23 – Comparison of FCH Acceleration
(Case #2, 16 000 kg, 4 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW)

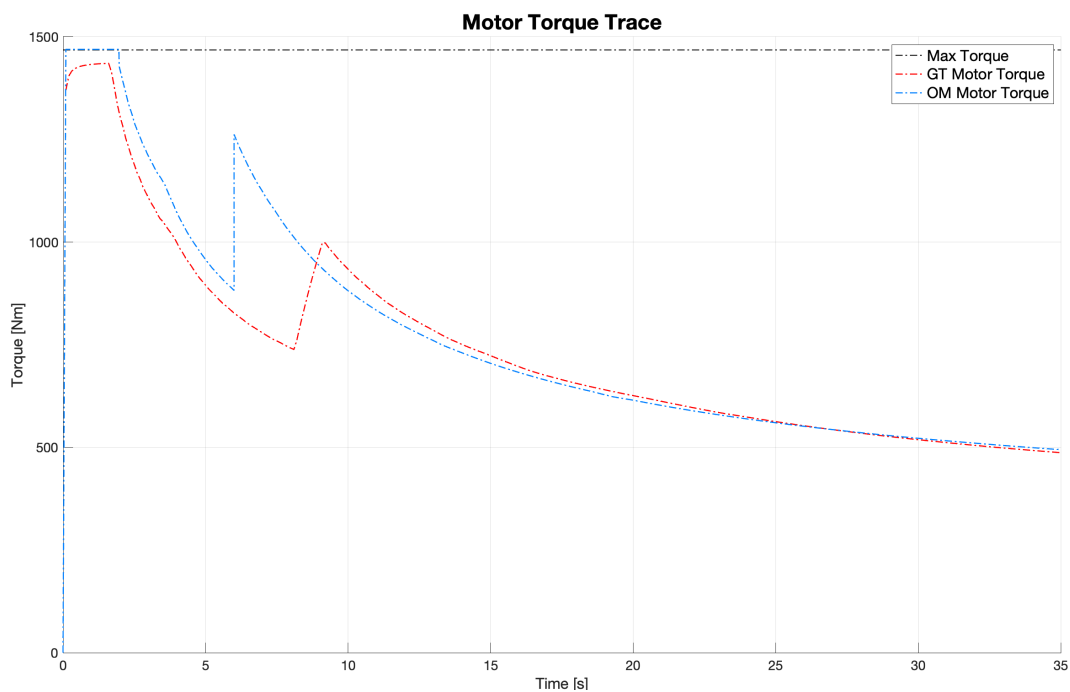


Figure D.24 – Comparison of FCH Torque Trace
(Case #2, 16 000 kg, 4 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW)

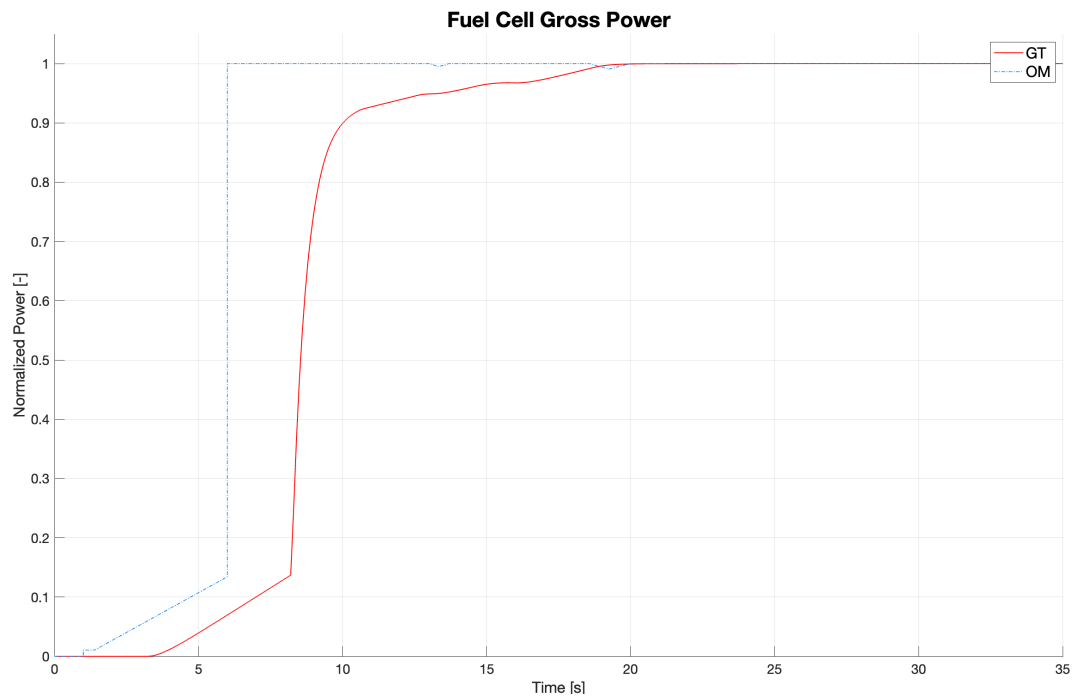


Figure D.25 – Comparison of FCH Fuel Cell Gross Power
(Case #2, 16 000 kg, 4 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW)

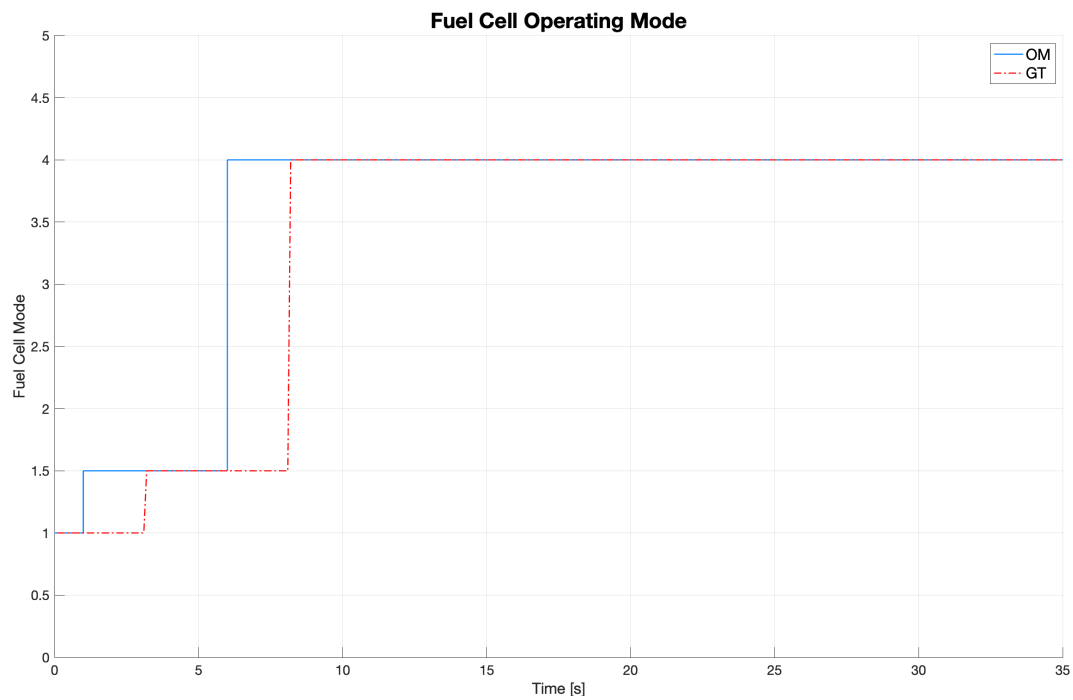


Figure D.26 – Comparison of FCH Hybridization Model
(Case #2, 16 000 kg, 4 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW)

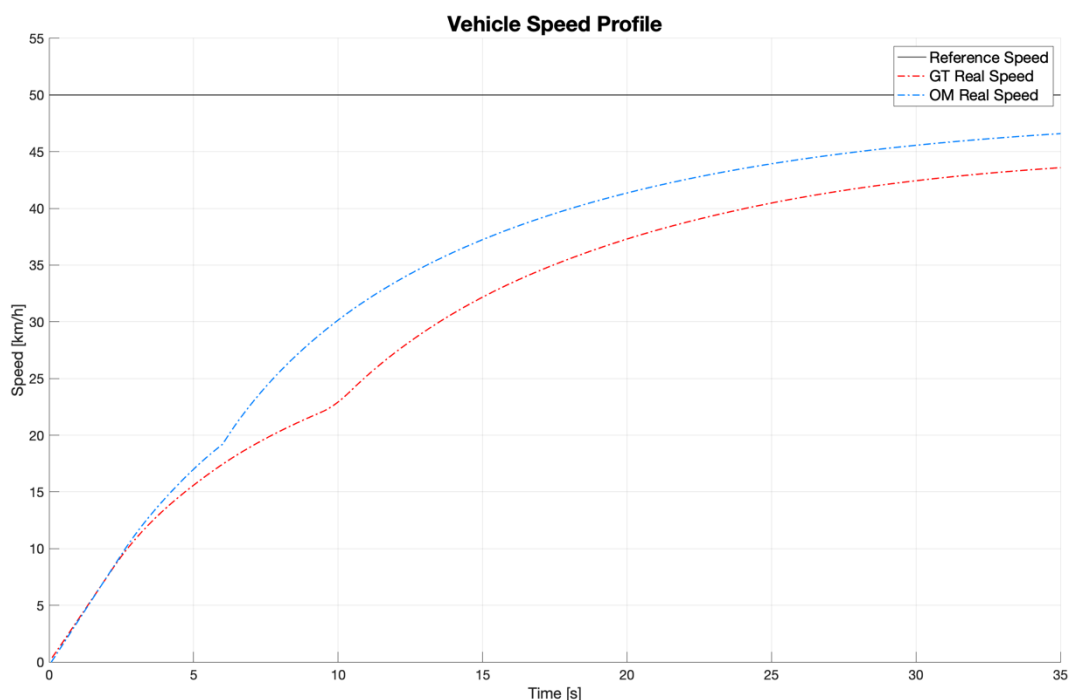


Figure D.27 – Comparison of FCH Acceleration
(Case #3, 16 000 kg, 8 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW)

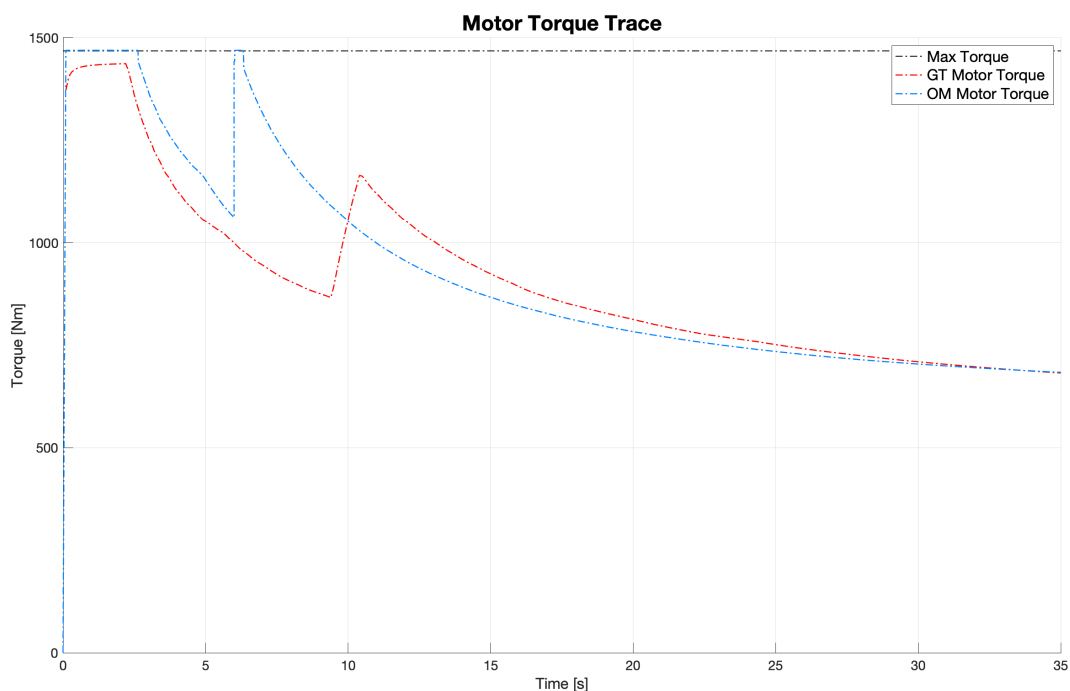


Figure D.28 – Comparison of FCH Torque Trace
(Case #3, 16 000 kg, 8 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW)

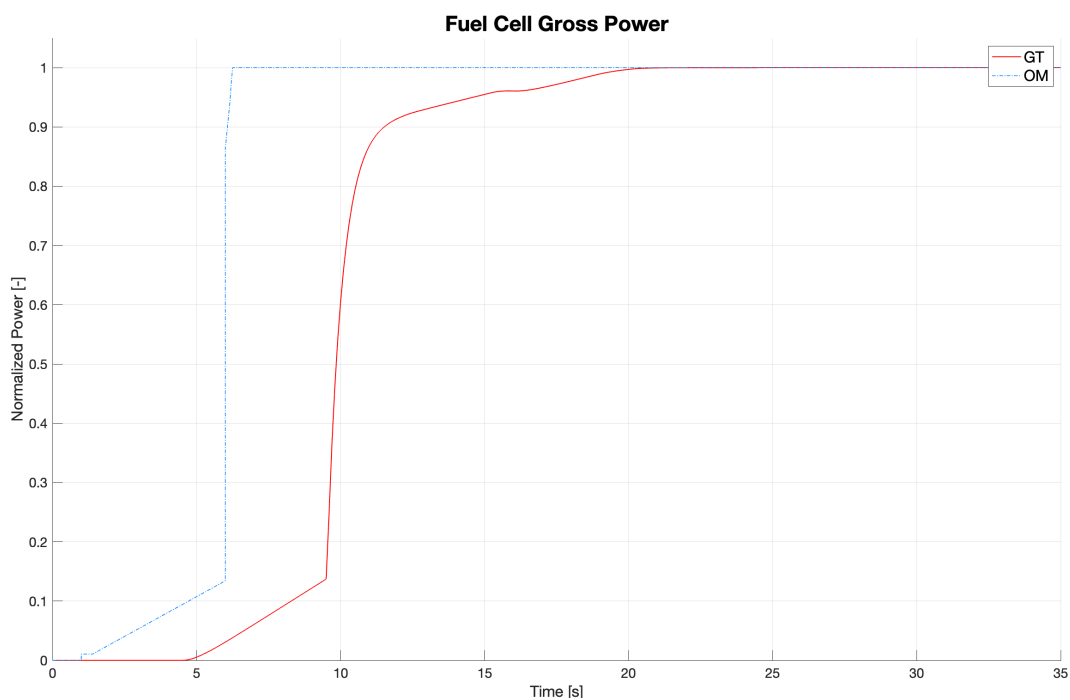


Figure D.29 – Comparison of FCH Fuel Cell Gross Power
(Case #3, 16 000 kg, 8 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW)

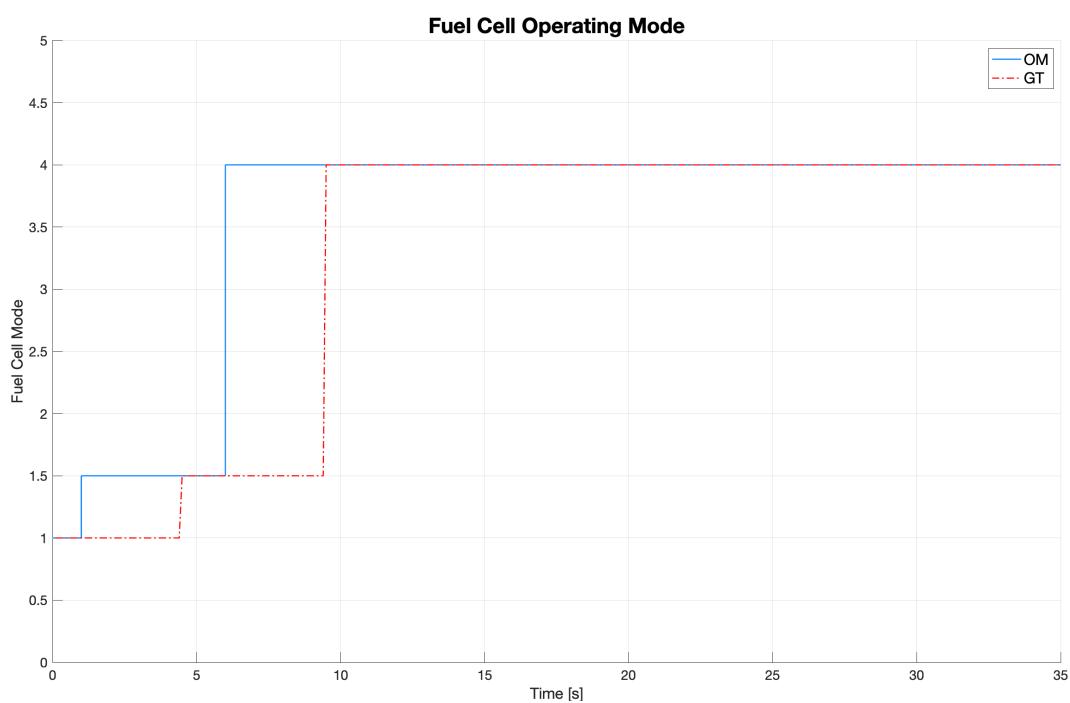


Figure D.30 – Comparison of FCH Hybridization Model
(Case #3, 16 000 kg, 8 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW)

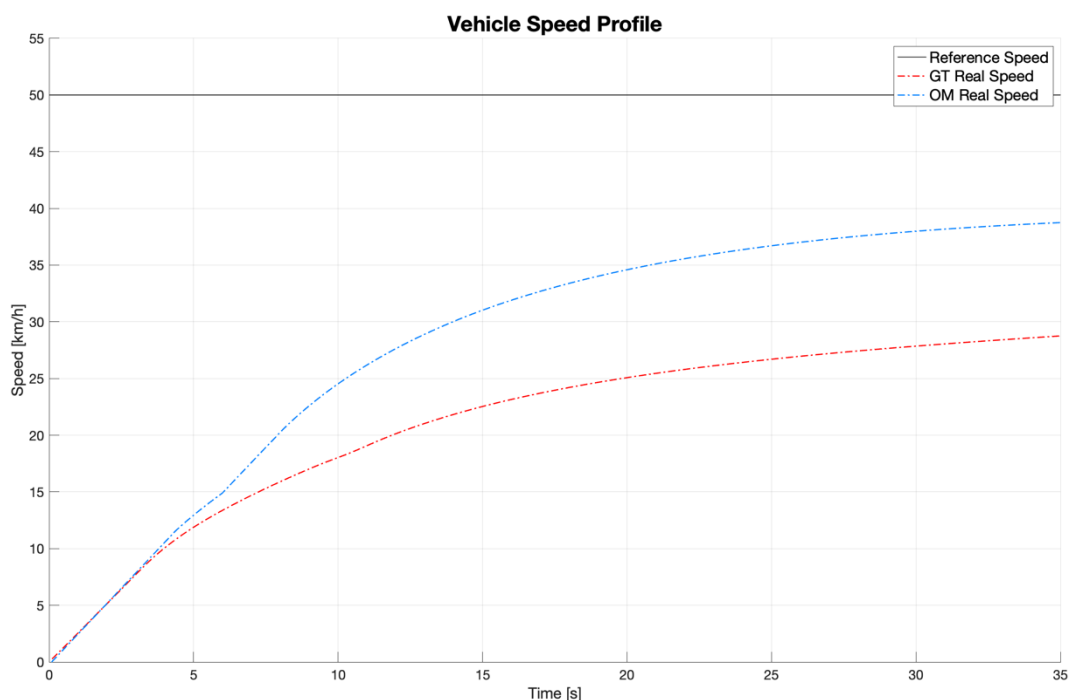


Figure D.31 – Comparison of FCH Acceleration
(Case #6, 19 000 kg, 8 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW)

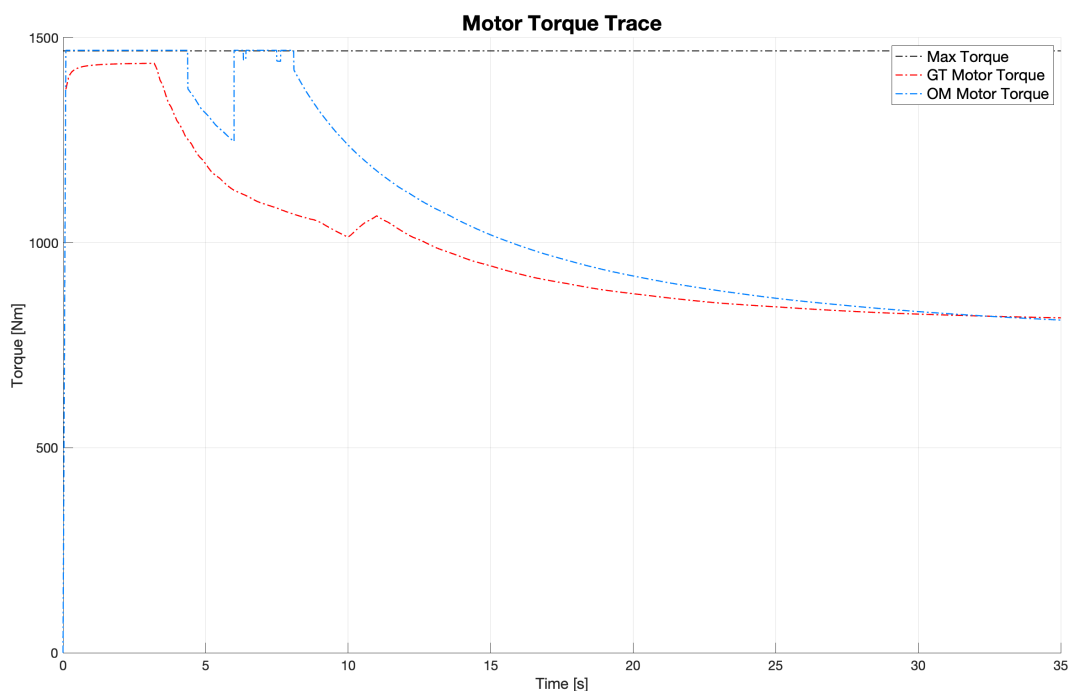


Figure D.32 – Comparison of FCH Torque Trace
(Case #6, 19 000 kg, 8 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW)

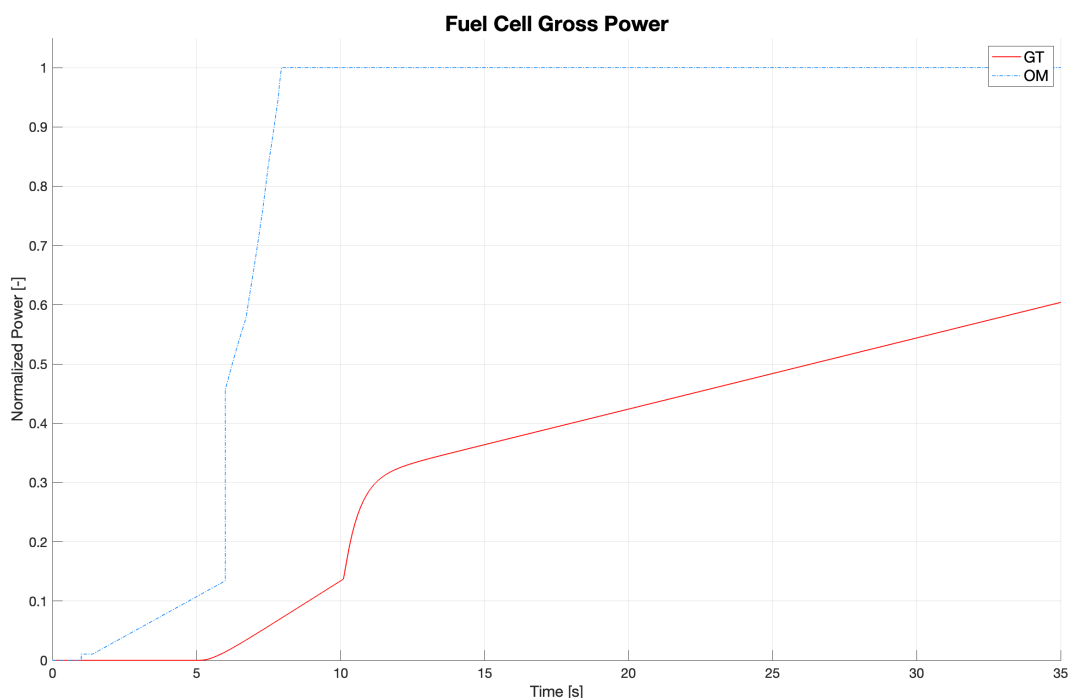


Figure D.33 – Comparison of FCH Fuel Cell Gross Power
(Case #6, 19 000 kg, 8 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW)

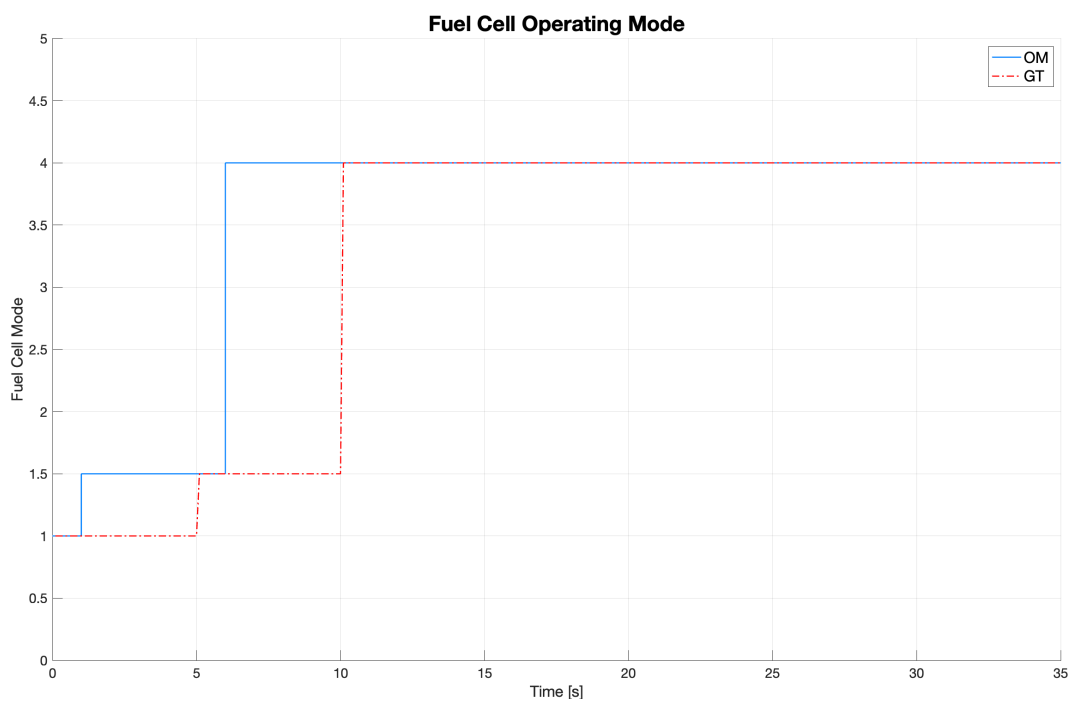


Figure D.34 – Comparison of FCH Hybridization Model
(Case #6, 19 000 kg, 8 % inclination, $SOC_0 = 62\%$ & $P_{aux} = 7.5$ kW)