POLITECNICO DI TORINO

Department of

Control and Computer Engineering (DAUIN)

Master Degree in Mechatronic Engineering

Autonomous PID controller regulation in nonlinear environments: an application to an automotive engine test bench

Supervisor
Prof. Alessandro Rizzo

Co-supervisors
Eng. Fabio Thione
Eng. Emanuele Baffo

Candidate
Luigi Spasiano

Academic Year 2020/2021

20th October, 2021

Abstract

Two of the most important topics nowadays, dealing with development of the society, are the concepts of standardization and automation. They are the key through which the humanity can make a step towards a future where the machines will deal with the majority of the jobs, helping the mankind to get rid of boring, dangerous, repetitive or unwanted tasks; giving to many people the possibility to spend their life only doing what they love or desire to do. The steps that are needed to reach this goal are the replacement of human work with robot work and the development of technologies that are able to take care of this change. In this regard, a multidisciplinary approach will be very important in order to handle all the aspects of the process.

In this thesis, this multidisciplinary approach will be evident dealing with a complex system like an automotive motor test bench. The aim of this thesis is to standardize many aspects of the actual test that until now are operator-dependent, creating a software environment that will manage these aspects and, at the same time, will propose some solutions based on its internal logic, this will be a software where the experience of the engineers that are working with the test bench will flow into. This thesis work sees as a goal, in practice, the development of a tool that will deal with the regulation of the PID controller of the test bench, this means that in every situation it will be able to compute the controller values to carry out the engine test in the desired way. This is crucial because the cell is a nonlinear system and it will be necessary to adapt many times the controller. The plant where the whole project has been conducted is the FPT Industrial SPA in Turin.

Contents

1. Introduction

1.1 Thesis overview

This chapter is intended to be a small overview and explanation of everything that will be met during the analysis. FPT Industrial is a part of CNH Industrial group that is dedicated to production and testing in industrial vehicles field (both on-road and off-road) and marine applications. The six kind of motors (diesel or natural gas) that can be found here are called R22, F1, F5, Nef, Cursor and Vector. Testing area in Turin is divided in test benches and offices where engineers are grouped in teams with different objectives. Several tests on real motor prototypes are performed weekly; generally, the Automation group, collecting a dataset which comes from motor's producer department, generates cycles that are going to be followed in the cell during the test cycle while the Homologation group performs the actual test in the bench. These cycles are the activities that the engine shall accomplish in order to fit some requirements, like to measure its emissions in a particular condition.

The aim of the test is to homologate the engine for the future sales. In order to do this, the engine has to pass tests with very stringent requirements about emissions, fuel consumption, quality and so on. This obviously requires a good realization in mechanical and electronics terms, but also a big effort in automation domain to make the test bench able to provide results with best efficiency; furthermore, the post-processing will be another fundamental automation task. In this regard, the goal of this thesis work will be the role of the automation during the test, its help in post-processing to fit the law standard requirements by manipulating the data and the role of the controllers during and after a test, moving between Automation and Homologation group in order to follow a test from its birth to the full homologation performed in the presence of an external examiner. Technical aspects will be discussed and all efforts will be connected to new solutions in automation field that can make easier and more efficient the test, in terms of data collection and analysis. A detailed view of the cell can be seen in figure 1.1 and 1.2.

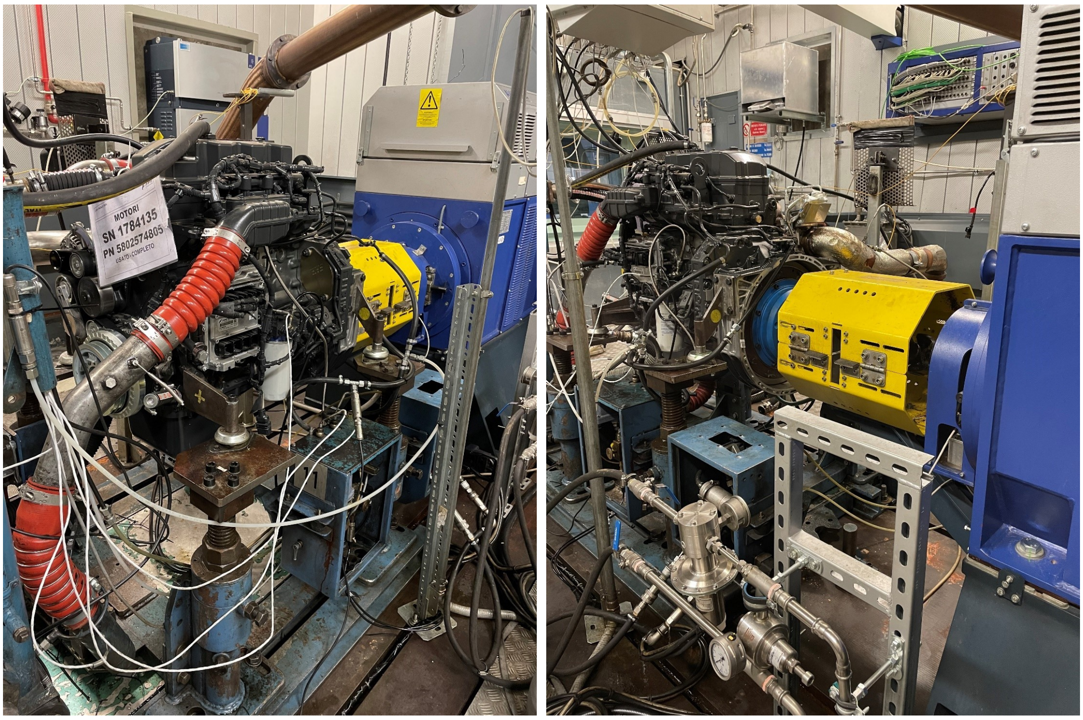


Figure 1.1: On the left, the test bench. On the right, a detail of the connection between engine and dynamo.

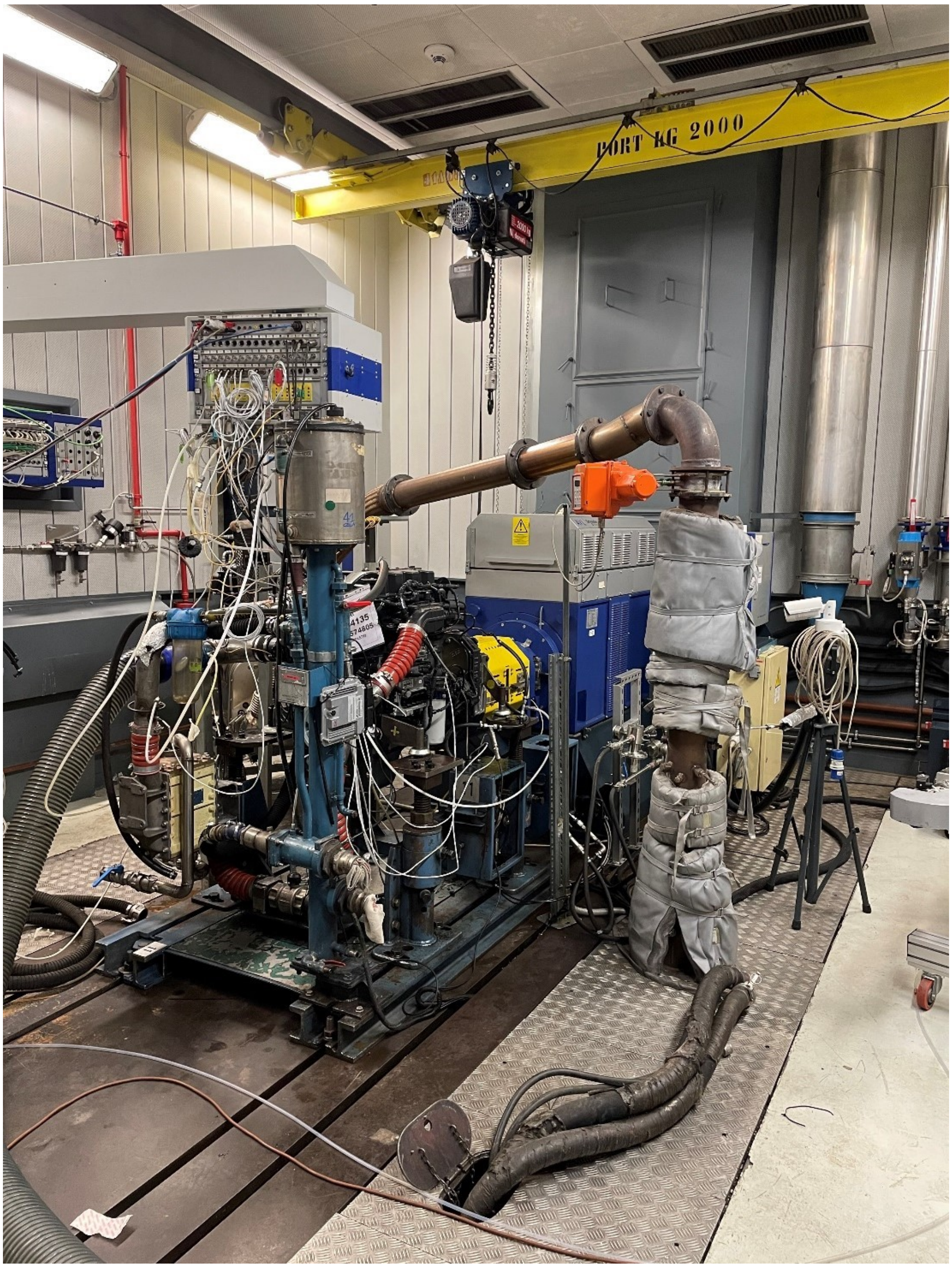


Figure 1.2: Engine and dynamo system.

Dealing with tests that will be met in on-road applications, most important ones are WHSC (world harmonized stationary cycle) and WHTC (world harmonized transient cycle), tests will be discussed in details later. When a test has to be performed, as said, the Automation group prepares a cycle that has to be followed during the test by the engine and one of their job is to check, after the conclusion, in post-processing, if it has been correctly followed as desired. In general, a regression analysis is employed, there are standard requirements that have to be checked between the cycle curve created by automation and the feedback that come back from the bench. To see this in details, some concepts about the work environment of the test have to be explained. The test bench is composed by a room where the engine and a dynamo are placed together with mechanical elements for their connection, the ECU and several actuators and sensors, some elements are also present to measure the quantities required by the law. This room is controlled in terms of temperature, pressure and more or less everything that can be important to replicate some standard conditions imposed by law. Engine parameters and room parameters are sent in feedback outside where an operator, behind a window, is able to see all those values on an Industrial PC where different software applications run. The PUMA open software by AVL is one of the most important: it deals with the whole control of the bench providing all the controlled values and giving the possibility to modify some of them thanks to a control panel, so an hardware with which on the basis of the used mode you can basically modify one of the control values. A mode is a way to control the engine and the dynamo by imposing a parameter for each of them. Together with PUMA, INCA software will provide the ECU flash and its parameters can be seen on another monitor: the operator will see both the ECU parameters and the PUMA ones.

A detailed view of the hardware can be seen in figure 1.3.

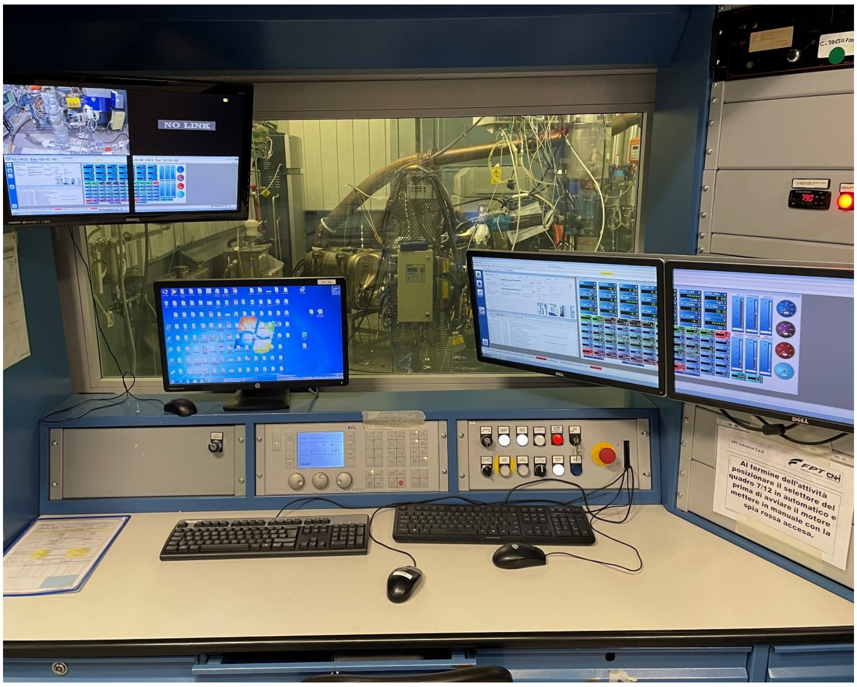

NO LINK
FPT CNH
Al termine dell'attività posizionare il selettore del quadro 7/12 in automatico e prima di avviare il motore mettere in manuale con la spia rossa accesa.

Figure 1.3: Puma

After this brief introduction, everything can be seen in details: from automation group a reference is created, let's take as example a reference for torque and a reference for rpm (speed). The PUMA environment takes them and feeds the engine with a command able to generate that torque, for example a value of the acceleration in 0-100% while the dynamo is fed by the rpm values. Sensors in the room will collect the real torque on engine and the velocity of the dynamo and through a communication line system those two values are sent back to the operator. A regression analysis is performed to check if the feedback values are not too far from reference, this is important because law requirements are very stringent, differences smaller than even 1% in some cases have to be guaranteed. This makes crucial the regression analysis; it is a very important part of the test, if it is not passed, the test can't go on, this means that a software that helps reaching this goal is of crucial importance. The aim of this thesis will be to make this analysis easier for the homologation group that has to perform the test, this can be possible through the automatization of some procedures based on the experience of engineers that with this conceptual work will be in some part translated into a software. When a regression is failed, what it's done is based on experience, generally the engineers will act on the PID of the PUMA hardware, the ones that are responsible for creating the variables that feed engine and dynamo. In practice, the human operator looks for some motivations for failure and together with what he knows about past cycles, he suggests some new values for the PID actions, but until now no systematic procedure exists to directly link what is going on during the regression analysis with a certain new value of the controller gains. Generally, let's say 100% of cases, problems are met on engine side, the dynamo is a system that has not the delays of an engine and it has no problem when it has to follow a reference. A very important instrument within testing environment is another AVL software: Concerto [1]. Another duty of automation group is the one to try to make life easier to everyone, in the sense that they have to translate every procedure into software to make everything faster, more standard and then to give a better efficiency to all the testing area. This is done using Concerto that is able to provide an environment based on a peculiar programming language that can be used to create formulas and graphical layouts. A formula is no more than a piece of code that helps achieving a particular need, for example it can take as input all the values met during the feedback discussed dealing with regression analysis and through a suitable formula written by automation the operator can check in almost zero time if regression analysis has passed or not. A formula has been written for any aspect of testing, from this big one to smallest one like a simple mean of an array of values. Everything is implemented in formulas that follow a LEGO-model structure so that a final big formula will just be an operation on many results coming from different formulas. The need for Concerto formula is never satisfied and everyday it is useful to create new ones for new aspects that engineers are called to face. This thesis work is totally realized working with Concerto 5 [2].

This is what will be analysed and developed, the experience of the operators will meet the software procedure and many formulas will finally give a further step to automation, not only the operator will know if regression has passed or not, but in negative situations, the new values for the PID will be automatically given by software leading to smaller waste of time and bigger efficiency. Furthermore, PID controller will be compared to others control techniques to make a comparison about best fit. This thesis work will integrate the industrial automation to software development and control techniques given a big overview on many mechatronic fields in a big industrial context like FPT that is a worldwide reality. Simulation tools like Simulink by MATLAB will also be employed to simulate the test bench with a big freedom that will lead to many analysis with helpful considerations about the bench control.

Simulink is a block diagram environment for multidomain simulation and Model-Based Design. It supports system-level design, simulation, automatic code generation, and continuous test and verification of embedded systems. Simulink provides a graphical editor, customizable block libraries, and solvers for modeling and simulating dynamic systems. It is integrated with MATLAB, enabling to incorporate MATLAB algorithms into models and export simulation results to MATLAB for further analysis [3].

In Concerto, a tool has also been developed, it is called Unigen and its purpose is to collect all data from the test and prepare a report where using the formulas every law-required value is compared to actual one. In reality, this operation is part of post-processing work that is a long part of the test where everything is put together to get the final information in terms of emissions, torque and everything is important in that actual test.

1.2 Tests performed in compliance with EURO VI standard

This section has been developed using an official guide provided by FPT that describes in details all their activities, many elements have been taken by this guide to develop this introductory section [4]. Concerning EURO VI regulation, there are some tests (with involved regulations) that have to be performed for certification purposes:

- Measurement of net power (R85);
- Smoke test (R24);
- Transient cycle WHTC (R49);
- Steady state cycle with transition ramps WHSC (R49);
- Emission off cycle Test WNTE (R49);
- Test OBD;
- Fuel consumption mapping cycle FCMC.

This thesis work will mainly deal with WHTC and WHSC.
The world harmonised transient test cycle (WHTC) is listed as a second-by-second sequence of normalized speed and torque values for a total of 1800 seconds. This can be seen in the following table.

Table 1.1: WHTC cycle.

Time [s]	**Normalized speed** [%]	**Normalized torque** [%]
1	0.0	0.0
2	0.0	0.0
3	0.0	0.0
4	0.0	0.0
5	0.0	0.0
6	0.0	0.0
7	1.5	8.9
8	15.8	30.9
9	27.4	1.3
10	32.6	0.7
11	34.8	1.2
12	36.2	7.4
13	37.1	6.2
14	37.9	10.2
15	39.6	12.3
...	...	...
1800	0.0	0.0

Relevant pollutant to be measured and limits.

Table 1.2: Pollutant WHTC data.

	WHTC (CI)	**WHTC (PI)**
CO [mg/kWh]	4000	4000
THC [mg/kWh]	160	
NMHC [mg/kWh]		160
CH4 [mg/kWh]		500
NOX [mg/kWh]	460	460
NH3 [ppm]	10	10
PM mass [mg/kWh]	10	10
PM number [/kWh]	$6 * 10^{11}$	$6 * 10^{11}$

Sequence of operations for Transient cycle – WHTC.

- Generate engine mapping (maximum torque curve);
- Generate reference test cycle;
- Run one or more practice cycles as necessary to check engine/test cell/emissions systems;
- Natural or forced engine cool-down;
- Ready all systems for sampling and data collection;
- Cold start exhaust emissions test WHTC;
- Hot soak period;
- Hot start exhaust emissions test.

In order to perform the WHTC test on an engine test cell, the normalized values shall be converted to the actual values for the individual engine under test based on the engine mapping curve as previously determined with some conditions for NNR Engine use. The conversion is referred to denormalization, and the test cycle so developed is the reference cycle of the engine to be tested. With those references speed and torque values, the cycle shall be run on the test cell.
For generating the reference cycles, the normalized speed shall be denormalized using

the following equation:

$$n_{ref} = n_{norm} * (0,45 * n_{lo} + 0,45 * n_{pref} + 0,1 * n_{hi} - n_{idle}) * 2,0327 + n_{idle} \quad (1.1)$$

For generating the reference cycles, the torque values for each individual reference speed value shall be denormalized, using the mapping curve previously determined using the following equation:

$$M_{ref,i} = \frac{M_{norm,i}}{100} * M_{max,i} + M_{f,i} - M_{r,i} \quad (1.2)$$

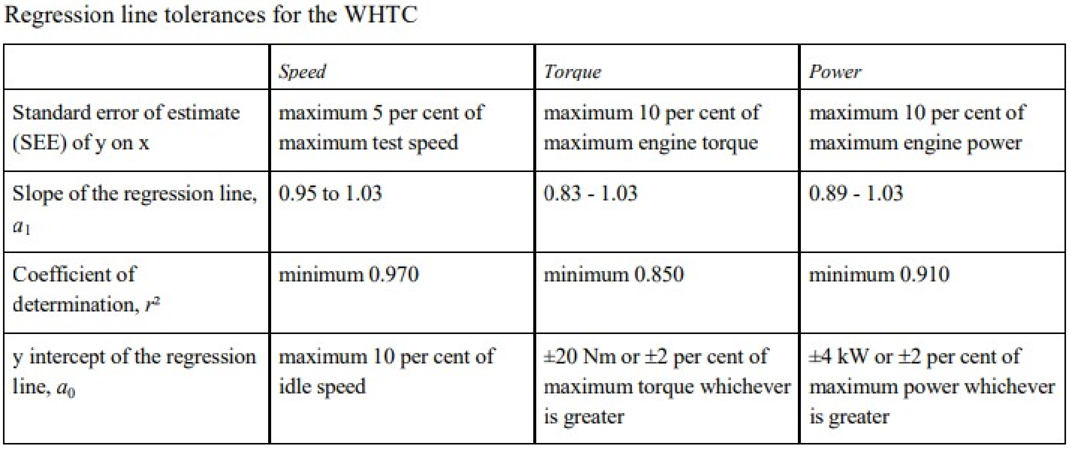
Furthermore, reference cycle work W_{ref} is calculated. Reference cycle work shall be determined over the test cycle by synchronously calculating instantaneous values for engine power from reference speed and reference torque. After generating the reference cycle with n_{ref}, T_{ref}, steps of “official certification sequence” shall be performed:

- Run one or more practice cycles as necessary to check engine/test cell/emission system;
- Engine cool-down;
- Cold start exhaust emission test WHTC (20-30 °C) ;
- Hot soak period;
- Hot start exhaust emission test WHTC.

Emission shall be measured in this time. If the engine stops during the WHTC cold test, the whole test is cancelled. If the engine stops during the WHTC hot test, it is needed to repeat the stabilization of the engine and to repeat the WHTC hot test. Concerning test conditions, the parameter f_a describing the laboratory test condition shall be within the following limits: $0,96 \leq f_a \leq 1,04$. The supply air temperature, which is recorded at rated speed and at full load, must correspond to an approximation of $\pm 5K$ with the maximum supply air temperature specified by the manufacturer. Before calculating actual cycle work, any point recorded during engine starting phase shall be omitted. Actual cycle work shall be determined over the test cycle by synchronously using actual speed and actual torque to calculate instantaneous values for engine power. Instantaneous engine power values shall be integrated over the test cycle to calculate the actual cycle work $W_{act}[kWh]$.

W_{act} is valid if: $85\% W_{ref} \leq W_{act} \leq 105\% W_{ref}$

In order to validate a WHTC cycle a linear regression analysis of actual values (speed, torque, power) on the reference values is employed. The method of least squares shall be used with equation: $y = a_1 x + a_0$
Regression limits for WHTC are collected from EURO VI manual and they can be seen in figure 1.4



Regression line tolerances for the WHTC

	Speed	*Torque*	*Power*
Standard error of estimate (SEE) of y on x	maximum 5 per cent of maximum test speed	maximum 10 per cent of maximum engine torque	maximum 10 per cent of maximum engine power
Slope of the regression line, a_1	0.95 to 1.03	0.83 - 1.03	0.89 - 1.03
Coefficient of determination, r^2	minimum 0.970	minimum 0.850	minimum 0.910
y intercept of the regression line, a_0	maximum 10 per cent of idle speed	±20 Nm or ±2 per cent of maximum torque whichever is greater	±4 kW or ±2 per cent of maximum power whichever is greater

Figure 1.4: WHTC tolerance

The specific emissions (including CO2 and fuel consumption) related to the WHTC test shall be calculated by weighting the results obtained in the cold start test and the hot start test, as follows:

$$e = \frac{(0.14 * m_{cold}) + (0.86 * m_{hot})}{(0.14 * W_{act,cold}) + (0.86 * W_{act,hot})} \tag{1.3}$$

In this equation, m_{cold} is the mass emission of the component on the cold start test, [g/test]; m_{hot} is the mass emission of the component on the hot start test, [g/test]; $W_{act,cold}$ is the actual cycle work on the cold start test, [kWh]; $W_{act,hot}$ is the actual cycle work on the hot start test, [kWh]. Is the calculation described above sufficient to determine the final emission values that must comply with the limits imposed by the legislation? The answer is NO. What shall be considered is a kind of regeneration associated with the engine and deterioration factor, the emission must be guaranteed throughout the useful life of the engine. Due to the periodic regeneration, the regeneration factors $k_{r,u}$ or $k_{r,d}$ shall be multiplied with or be added to, respectively, the specific emissions result e (weighted Cold/Hot).

$$e = k_r * \frac{(0.14 * m_{cold}) + (0.86 * m_{hot})}{(0.14 * W_{act,cold}) + (0.86 * W_{act,hot})} \tag{1.4}$$

$$e = k_r + \frac{(0.14 * m_{cold}) + (0.86 * m_{hot})}{(0.14 * W_{act,cold}) + (0.86 * W_{act,hot})} \tag{1.5}$$

The multiplicative adjustment factor shall be calculated as follows:

$k_{r,u} = \frac{e_w}{e}$ (upward)

$k_{r,d} = \frac{e_w}{e_r}$ (downward)

The additive adjustment factor shall be calculated as follows:

$k_{r,u} = e_w - e$ (upward)

$k_{r,d} = e_w - e_r$ (downward)

For a test without regeneration, $k_{r,u}$ shall be multiplied with or be added to, respectively, the specific emission e. For a test with regeneration $k_{r,d}$ shall be multiplied with or be added to, respectively, the specific emission e.
Procedure to determine K_r – WHTC

- The manufacturer provides an after-treatment system with a load that determines the activation of regeneration during the WHTC test;
- Engine warm-up;
- Preconditioning engine with hot WHTC cycle;
- Hot soak period;
- Hot WHTC (cycle official);
- Hot soak period;
- Official WHTC cycle(s): cycle(s) required to complete the regeneration event;
- Hot soak period;
- Hot WHTC (cycle official) .

The final results of the specific emission values previously calculated (considering already the regeneration event in the calculation), are obtained with the application of the deterioration factors (DF). The engines shall meet the respective emission limits for each pollutant after application (previously application of k_r) of the deterioration factors to the test result. Depending on the type of DF, the following previsions apply:

-Multiplicative: (specific emission, including k_r if applicable) * DF $\leq$ emission limits;
-Additional: (specific emission, including k_r if applicable) + DF $\leq$ emission limits.

The final results of emission for each pollutant are determined as follow:

$$e = \frac{(0.14 * m_{cold}) + (0.86 * m_{hot})}{(0.14 * W_{act,cold}) + (0.86 * W_{act,hot})} \tag{1.6}$$

Where k_r takes into account regeneration events and DF is deterioration factor.

Table 1.3: Regeneration and deterioration factors.

$(e * k_r) * DF \leq$limit applicable	First possible case
$(e * k_r) + DF \leq$limit applicable	Second possible case
$(e + k_r) * DF \leq$limit applicable	Third possible case
$(e + k_r) + DF \leq$limit applicable	Fourth possible case

The ramped steady state test cycle WHSC consists of a number of normalized speed and load modes. The engine shall be operated for the prescribed time in each mode, whereby engine speed and load shall be changed linearly within 20 ± 1 seconds.

In order to perform the WHSC test on an engine test cell, the normalized values shall be converted to the actual values for the individual engine under test based on the engine-mapping curve as previously determined with same condition for NNR Engine use.

Mode time duration includes a 20 seconds ramp.

Table 1.4: WHSC cycle.

Mode [ ]	Normalized speed [%]	Normalized torque [%]	Mode time [s]
1	0.0	0.0	210
2	55	100	50
3	55	25	250
4	55	70	75
5	35	100	50
6	25	25	200
7	45	70	75
8	45	25	200
9	55	50	125
10	75	100	50
11	35	50	200
12	35	25	250
13	0	0	210
Sum			1895

Denormalization of engine speed:

$n_{ref} = n_{norm} * (0,45 * n_{lo} + 0,45 * n_{pref} + 0,1 * n_{hi} - n_{idle}) * 2,0327 + n_{idle}$

Denormalization of torque values:

$M_{ref,i} = \frac{M_{norm,i}}{100} * M_{max,i} + M_{f,i} - M_{r,i}$

Table 1.5: Pollutant WHSC data.

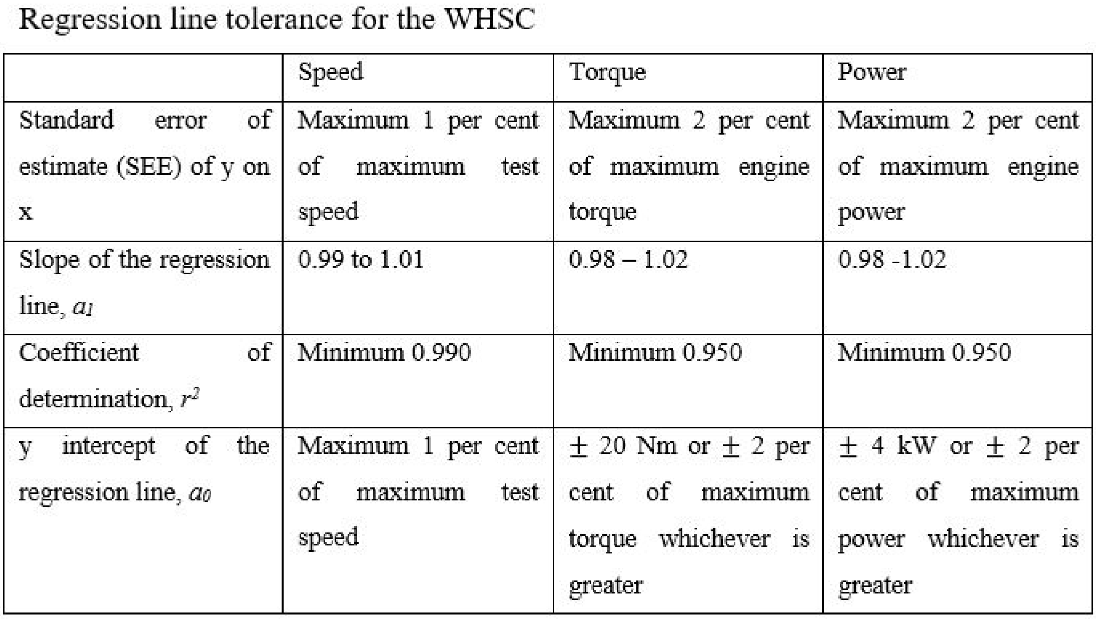
	WHSC (CI)
CO [mg/kWh]	1500
THC [mg/kWh]	130
NMHC [mg/kWh]	
CH4 [mg/kWh]	
NOX [mg/kWh]	400
NH3 [ppm]	10
PM mass [mg/kWh]	10
PM number [/kWh]	$8 * 10^{11}$

Official Certification Sequence for steady state cycle – WHSC

- Generate engine map (maximum torque curve);
- Generate reference test cycle;
- Starting the dilution system and heating engine;
- Engine preconditioning at mode 9 (minimum 10 ±1min);
- Engine stop;
- Engine starting 5 ± 1 minutes after completion of preconditioning at mode 9;
- Run test WHSC.

How to validate a WHSC cycle?
Linear regressions of the actual values $(n_{act}, M_{act}, P_{act})$ on the reference values $(n_{ref}, M_{ref}, P_{ref})$ shall be performed for both. The check of linear regression passes through the method of least squares shall be used with follow equation: $y = a_1x + a_0$

Regression limits for WHSC are collected from EURO VI manual and they can be seen in figure 1.5 from EURO VI standard.



Regression line tolerance for the WHSC

	Speed	Torque	Power
Standard error of estimate (SEE) of y on x	Maximum 1 per cent of maximum test speed	Maximum 2 per cent of maximum engine torque	Maximum 2 per cent of maximum engine power
Slope of the regression line, a_1	0.99 to 1.01	0.98 – 1.02	0.98 -1.02
Coefficient of determination, r^2	Minimum 0.990	Minimum 0.950	Minimum 0.950
y intercept of the regression line, a_0	Maximum 1 per cent of maximum test speed	± 20 Nm or ± 2 per cent of maximum torque whichever is greater	± 4 kW or ± 2 per cent of maximum power whichever is greater

Figure 1.5: WHSC tolerance

The specific emissions (including CO2 and fuel consumption) related to the WHSC test are calculated as follows:

$e = \frac{m}{W_{act}}$

In this equation m is the mass emission of the component and W_{act} is the actual cycle work. The final results of emission for each pollutant are determined as follow. Depending on the type of DF, the following previsions apply:

-Multiplicative: (specific emission e) * DF $\leq$ emission limits;

-Additional: (specific emission e) + DF $\leq$ emission limits.

1.3 PID controller

Automatic control techniques are employed when a particular physical quantity must behave in a particular manner without human intervention. Nowadays society is full of examples regarding this technology, one of them is the air conditioning system in a house: it is a system that has to keep the room temperature at a desired level and it only needs an input, then every other action is performed by system controller. The full scheme of interest is represented in figure 1.6

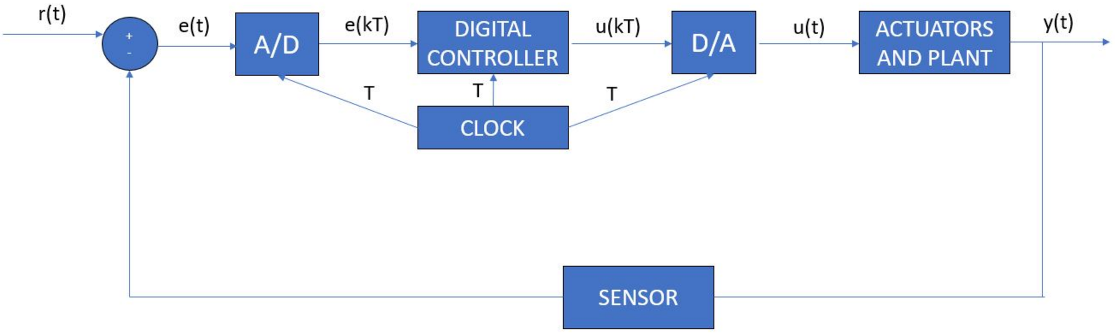


Figure 1.6: Automatic Control Scheme

An explanation of the figure is now provided.

- Plant and actuators. It is the system under control, actuators are needed since the controller provides a low power signal that is not sufficient to cause the desired action on the plant: the role of the actuators is to amplify that signal till the desired action on the plant is met;
- Digital controller is the controller device that is responsible to provide the control input to the plant as its output. Digital devices are now replacing analog controllers that were affected by many drawbacks like components degradation and hard re-tuning. Controller input is based on the employed architecture: there are open-loop controllers where no information about the plant is known, in this context only the reference signal will drive the controller action. The reference signal r(t) gives the desired value of the controlled output. Dealing with closed-loop controllers, a sensor is employed to get information about the plant status, this will surely enhance the control action, for this reason in the majority of applications, when possible, the closed-loop solution should be preferred. In this context, the input of the controller will be different, it is called tracking error e(t) and it is equal to the difference between the reference signal and the controlled output, in this way the controller will know how far the output of the system is from reference signal and it will compensate this difference, hence the objective of the controller architecture is to have a tracking error as small as possible: ideally at e(t)=0 the controller is perfectly acting on the system and the output is exactly the desired quantity. This process is called regulation. This is not an easy task in a real environment

because of the presence of disturbances that will make very hard a perfect match between r(t) and y(t). A system with a good regulation even in the presence of disturbance signals has a good disturbance rejection. A system with good regulation in the face of changes in the plant parameters has a low sensitivity to these parameters. A system with both good disturbance rejection and low sensitivity is called robust [5];

- A/D and D/A. The digital controller, as said by its name, is nothing but a computer, then it works only with digital signals, this means that the signals in the controller architecture are not continuous, they are called discrete-time signals since they are composed by a sequence of real numbers, these signals are only defined in discrete-time instants, defined by the Sampling Time. To better explain this concept, the situation can be summed up by saying that the digital controller, at each sampling time, compute the control input on the basis of the tracking error. The role of the Analog to Digital Converter (A/D) and the Digital to Analog Converter (D/A) is then the one to convert a physical signal (like an electrical voltage) into a stream of bits (A/D) as input of the digital controller, while the vice versa is performed by D/A that takes as input a stream of bits coming from the digital computer and it is able to convert this stream into a physical quantity with which the analog parts of the controller architecture can deal with;

- Transfer Functions. All those blocks, during the design phase, are studied through a function that is called Transfer Function (TF). Since they are single input single output (SISO) systems, they can be resumed by this TF that is the ratio between the system output and the system input;

$$H(z) = \frac{N(z)}{D(z)} = \frac{b_m z^m + b_{m-1} z^{m-1} + ... + b_1 z + b_0}{z^n + a_{n-1} z^{n-1} + ... + a_1 z + a_0} \tag{1.7}$$

The degree of the numerator is always equal or smaller than the degree of the denominator, $m \leq n$. The roots of nominator function are called zeros, while the roots of denominator function are called poles

To cope with discrete-time signals, those functions are obviously not function of time as well, but they are defined in Z-domain. The Z transform will help solving problems that deal with that kind of signals.

$$F(z) = \sum_{k=0}^{\infty} f(kT)z^{-k} \tag{1.8}$$

Another transform that is widely used when dealing with continuous system is the Laplace Transform, it is helpful to solve differential equations, typical of a continuous problem, without the need to perform complex integrals.

$$F(s) = \int_{-\infty}^{+\infty} f(t)e^{-st}\,dt \tag{1.9}$$

Among all the families of controllers, probably the most common one is the PID controller. PID regulators react proportionally to the current error value, take into consideration the past error signal history with the integral of the error signal, and count the future error signal trend by the differential of the error. Dealing with PID regulator, the control action is now computed summing a contribution that is proportional to the error (P), a contribution that is related to the integral of the error (I), and a contribution that is related to the differential of the error (D). Applications of PID techniques are mostly related to industrial control systems, they are the most frequently applied controllers due to the simplicity in realization, all the industrial needs they can fulfil and their simple re-tuning operation [6].
The control action performed by the PID can be represented as:

$$u(t) = K_P\, e(t) + K_I \int_0^t e(\tau)d(\tau) + K_D \frac{de(t)}{dt} \tag{1.10}$$

Alternatively, the control law can be written as:

$$u(t) = K_P \left[e(t) + \frac{1}{T_I} \int_0^t e(\tau)d(\tau) + T_D \frac{de(t)}{dt} \right] \tag{1.11}$$

In this way, the regulator parameters are Kp (the proportional gain), T_I (the integrating time constant) and T_D (the differentiating time constant).
As said, the controller can be expressed by means of a transfer function (TF). As it can be seen from its block representation, the controller TF is the ratio between the control action expression (that is its output) and the tracking error expression (that is its input).

e(kT)
DIGITAL CONTROLLER
u(kT)

Figure 1.7: Controller Scheme

The controller TF expression in Laplace domain is the following one:

$$H(s) = K_P + \frac{K_I}{s} + sK_D \tag{1.12}$$

$$H(s) = K_P \left[1 + \frac{1}{sT_I} + sT_D\right] \tag{1.13}$$

$$H(s) = \frac{K_P}{T_I} \frac{1 + sT_I + s^2 T_I T_D}{s} \tag{1.14}$$

The problem is that the degree of the numerator is bigger than the one of denominator, for this reason this expression is not feasible and for practical applications, to make feasible the derivative action, it is needed to add an high frequency pole. The TF of the new approximate PID regulator is given by the following equation:

$$H(s) = K_P \left[1 + \frac{1}{sT_I} + \frac{sT_D}{1 + sT}\right] \tag{1.15}$$

$$H(s) = \frac{K_P}{T_I} \frac{1 + s(T_I + T) + s^2 T_I (T_D + T)}{s(1 + sT)} \tag{1.16}$$

Now the regulator has two poles, one at the origin and the second at $-\frac{1}{T}$ in the complex plane.

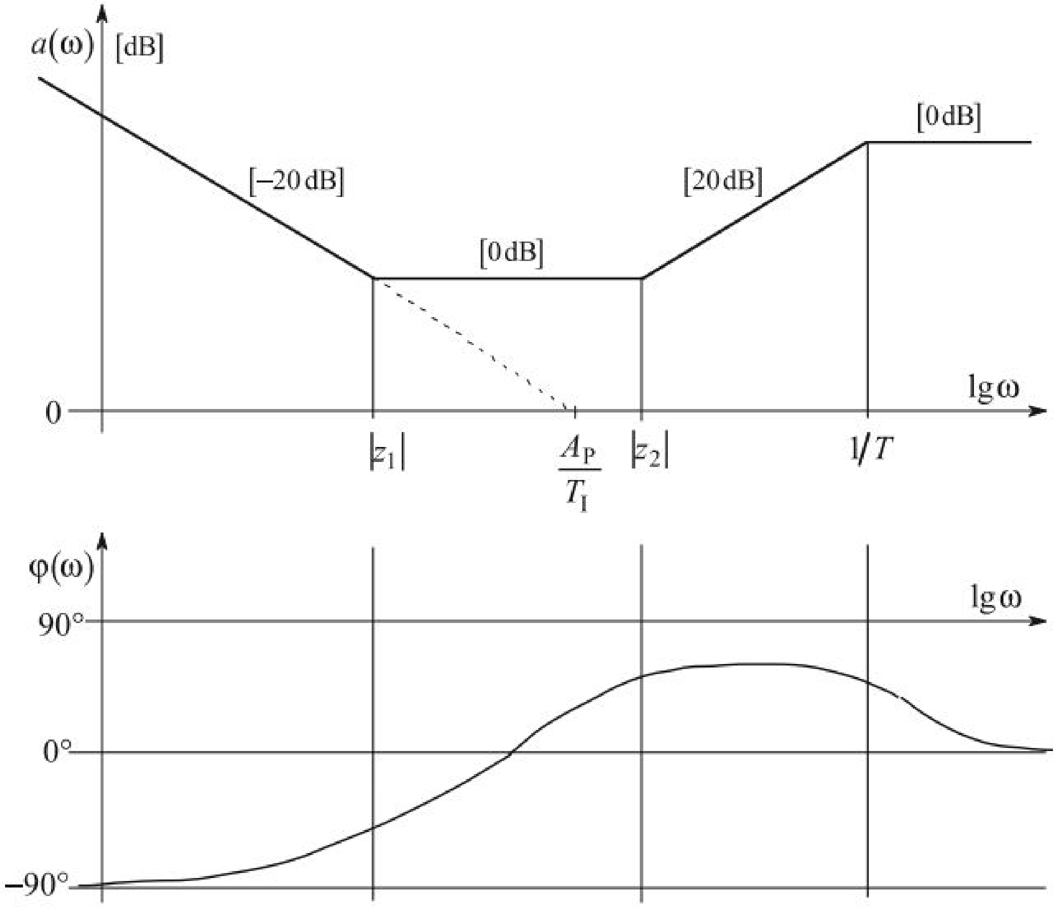


Figure 1.8: Asymptotic BODE diagram of the approximate PID regulator from "Control Engineering" [6]

As it can be seen, now the controller shows two zeros, for this reason the design of the PID can me summed up by saying that it is reduced to a choice of a coherent gain Kp and a choice of a coherent position for those two zeros: generally the gain is chosen to impose certain dynamics while the zeros are chosen to cancel the plant transfer function poles.

1.4 Regression analysis

Regression analysis is a statistical tool for investigation of relationships between variables [7]. Among all the regression techniques, in this context the least square method is used. This technique leads to a regression curve or a line that shall

approximate all the points in a plane, where those points are the available data. This curve has to be the one that is able to minimize the sum of the squares of the distances among the points and the curve itself.
The EURO VI standard explicitly tells to make usage of a regression line that follows the equation:

$y = a_1 x + a_0$

The parameter a_1 is the line's slope. It is called "m" in the literature. The law wants this parameter to be as close as possible to 1 because the line that has unitary slope is the plane bisector, this means that a value on the y-axis is equal to a value on the x-axis, and since the actual torque values have been placed on y-axis and the reference values have been placed on x-axis, this unitary slope would reflect in a perfect match among those values. However, the regulator knows that it is almost impossible, for this reason it is just sufficient to reach a slope value that is inside a range around 1, as specified by the standards. Dealing with WHTC, this range is $0.83 <$ slope < 1.03.

The parameter a_0 is called intercept. It is called "q" in the literature. This value is the offset between point (0,0) and the point where the line crosses the y-axis (0,q). Obviously, this time the standards want this value to be as close as possible to 0, for the same reason that pushes the slope to be 1, that is a perfect match between actual and reference values. As for the slope, a range is also provided for the intercept, because it is not easy to reach perfectly 0. This time the choice of the range is left to the engine tester, it is possible to choice between the range -20 < intercept < +20 or -2% of the maximum torque < intercept < +2% of the maximum torque. In this thesis, the range -20 < intercept < +20 has been chosen since it was the one employed by FPT Industrial.

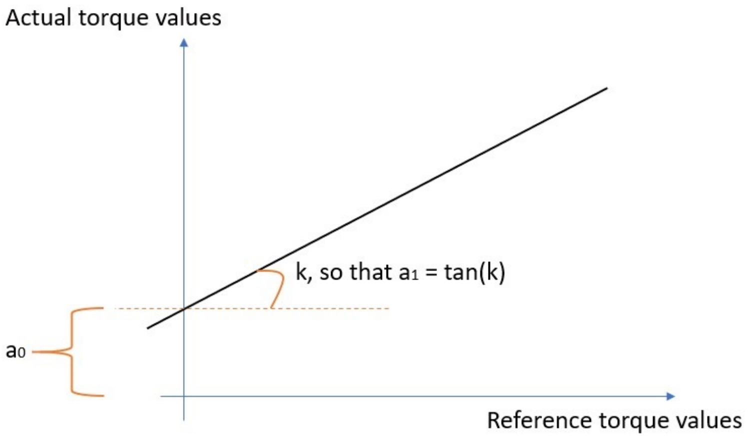
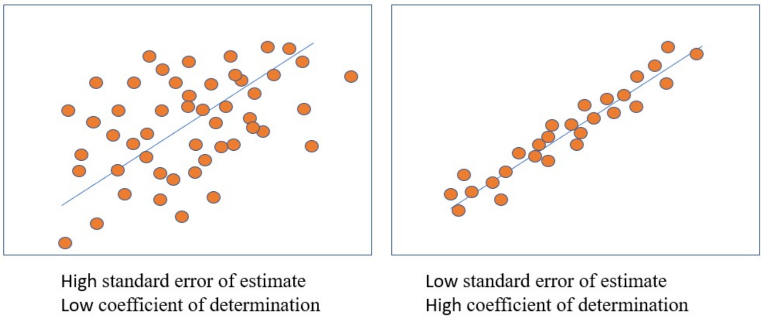


Figure 1.9: Regression line

The law requirements are not only expressed in terms of slope and intercept of the regression line. Two other quantities are specified: SEE and r^2.

The standard error of estimate (SEE) is a way to measure the distance between the data and the regression line, so it is a way to understand how bad your approximation line is and then, since it is a distance estimate, it is preferable to have SEE as low as possible, to be closer to the line. For this reason, the standards only provide a maximum value that SEE doesn't have to cross: in WHTC it is equal to 10% of the maximum engine torque.

The coefficient of determination, (r^2), is basically a measure of the model, in the sense that it tells how good the chosen line approximates the data. For low r^2, the data are far from the line, for high r^2 the data points are very close to the line. The coefficient r^2 explains the percentage of variable variation that this linear model is able to explain through its expression. As it is clear, this value can be as close to maximum as it wants, the law only requires the satisfaction of a minimal threshold cross, that is to say that only a minimum value is provided, again, referred to WHTC, this value is 0.85 (85%).



High standard error of estimate
Low coefficient of determination

Low standard error of estimate
High coefficient of determination

Figure 1.10: SEE and r^2

2. PID Analysis Tool

This chapter will face every criticality that has been met during the coding of the software in terms of achievement and graphical representation of the results. A huge number of lines of code has been written to create this software and a big overview will be provided, starting from the empty Concerto editor.

2.1 Software environment

From this moment, after this introductory part, all the effort will be devoted to the software tool creation. The starting point is the Concerto environment, that is a professional software provided by AVL (an independent company for development, simulation and testing in the automotive industry), it is helpful to analyse and to handle data coming from a test bench. When a cycle run on a bench, the fundamental measurements are collected and sent over a server that is connected to Concerto, in almost real time it will provide these results to Automation group that can work on them. Usually the data are managed through formulas that are written in a peculiar programming language, for example speed and torque values are collected and power values are computed after the test by a simple formula that multiplies them; everything can be implemented by a formula or many of them, from this basic power computation to law requirements check based on thousands of lines of code. Furthermore, Concerto gives the chance to build a graphical layout where formulas results are shown together with curves, diagrams, tables and more or less everything that can be useful to implement a kind of report of your activity. The software tool will then be based on formulas written in Concerto "language" and a graphical layout where results are shown to user that has only to upload the desired test and the software, through the layout and the formulas that will be created, will provide many useful results, from regression analysis check to PID modification suggestions. Opening the Concerto software, a choice among many features has to be done. Generally the Concerto Application Development Toolbox is enough to write formulas and create a layout. Many other features are available like a Python support for programming with Python, graphical elements that generate very important diagrams in vehicle analysis like a pV diagram and so on.

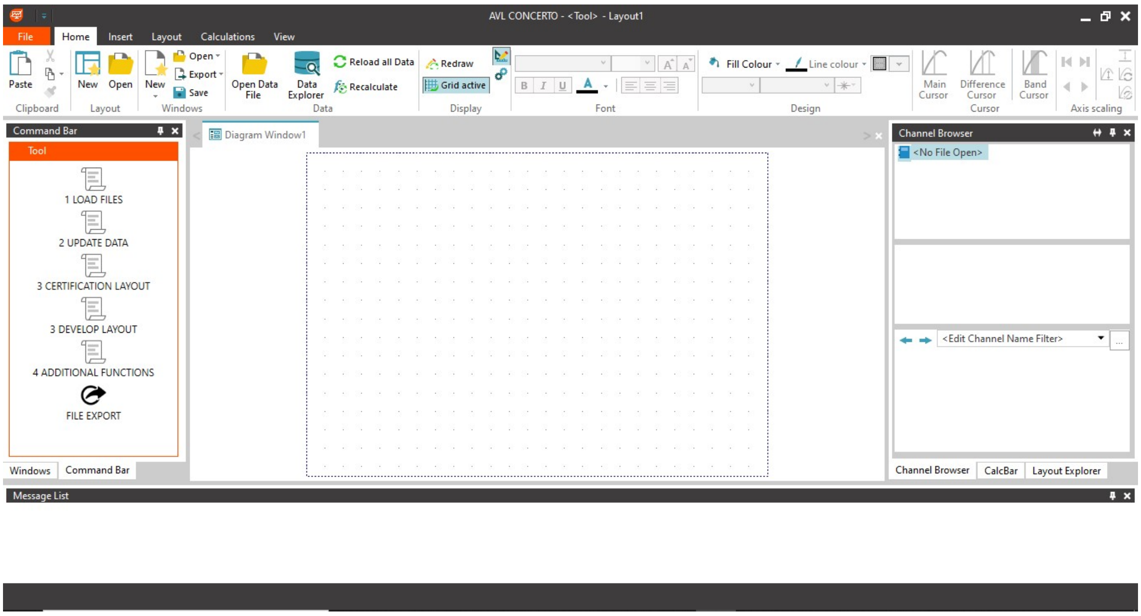


Figure 2.1: Concerto environment

This is the Concerto environment once opened. On the left there is the Unigen Tool that gives chance to load a test, to modify it for post-processing purposes and to generate the certification layout that compares every law requirement with test results. This is fundamental to homologate and it shows the role of automation in nowadays industrial reality: a click on that “certification layout” button followed by a few minutes wait would be comparable to several days of calculation and the constant fear to commit errors, without the standardization support. On the right, once uploaded a test, every measurement about it will be present and ready to be used in a formula. On the top there are many windows: in “home” a new layout can be started and analysed by the commands on the top-right, in “insert” diagrams, tables, texts and many other things can be inserted in the layout, in “layout” there are some management actions that can be performed on the layout itself like how to arrange diagrams, in “calculation” there is the formula/script editor while in “view” there are some settings to Concerto general visualization that can be personalized.

Due to the importance of Unigen, it will be very helpful to integrate the new-born software tool as a sub-tool within Unigen, this can enhance user's experience that will not have to switch between Unigen and the new tool, due to the fact that Unigen is however always employed.

For this reason the tool will be realised within option 4 of Unigen Tool, "Additional Functions". A new empty tool will then be coded modifying the existing code of Addition Functions section by right-clicking on it and selecting "Edit".

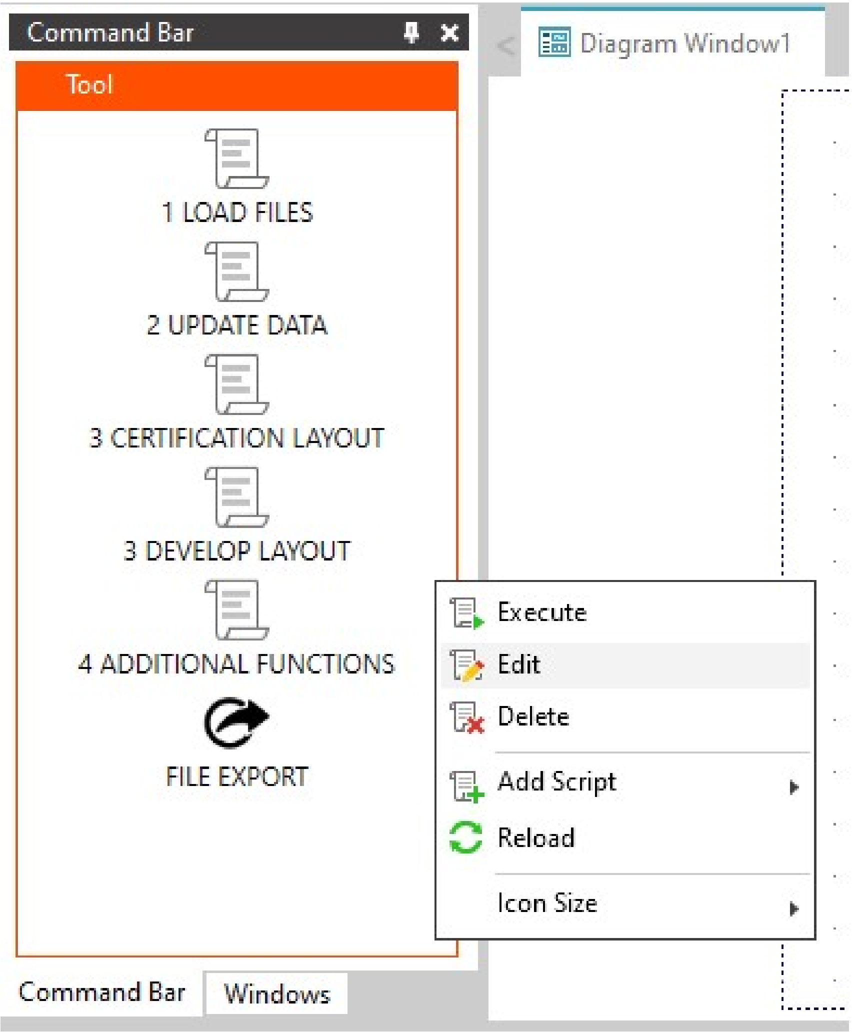


Figure 2.2: Software birth

Looking at the code, it will be sufficient to add within the user's choices a new one that will point to a new sub-tool among the others that is "PID Analysis Tool", this is the official birth of the software that now can be selected by the user even if it is empty in this moment.

2.2 Software creation

To really create the software now it will be sufficient to create a new layout that in other words will be like a software interface for the user, so first of all a Cover Main Page is created with the FPT logo.

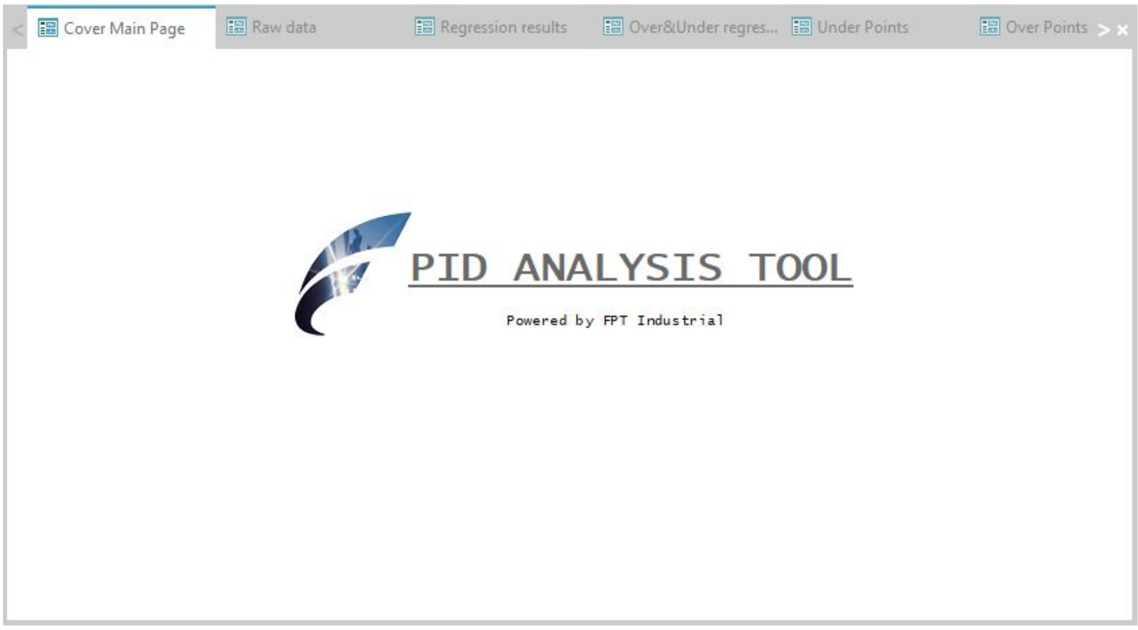


Figure 2.3: Cover main page

Then the idea will be to show the raw data in order to give a first overview to software user about what is going on during the test so at the same time the actual and the demanded values for both speed and torque are plotted, together with alpha, that is the curve of accelerator values (0-100%) , the mode used to perform the test and a flag, a formula that has been implemented to be 1 when the used mode is 12, 0 otherwise. This is done because mode 12 is the most used one. From this moment on, the layout screens will be adapted to better fit the view of the results and only a particular with the relevant information will be displayed in the thesis, instead of placing the whole screen of the software's layout. For software realization purposes, an old test with failed regression is used, just to have some data to work on. It is a F1C diesel engine with turbocharger, with a nominal power of 152 kW at 3500 rpm and a nominal torque of 470 N at 1400 rpm. It has a 2999 cm^3 displacement.

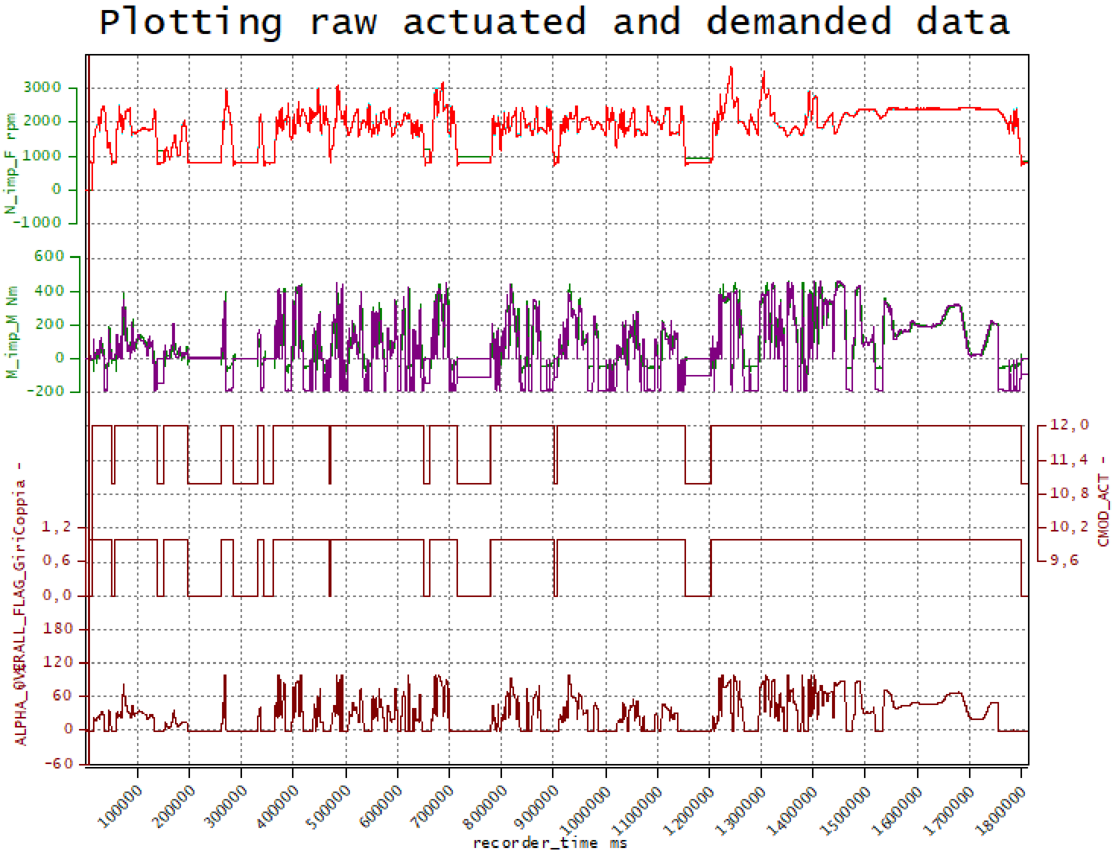


Figure 2.4: Raw data

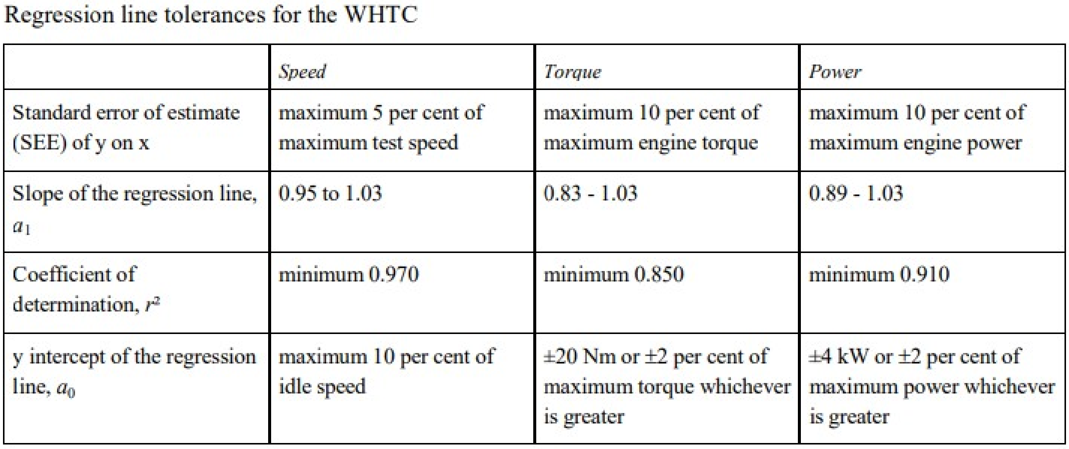
Several modes are used within the same test, they are collected inside the CMOD_ACT array, that collects the mode value for each recorded time, 18130 values in log-points or from 88 ms to 1812988 ms with a 100 ms step among them. Calling N the engine speed, T the engine torque, alpha the accelerator value, these are the most important modes:

- CMOD_ACT=8 -> N/alpha
- CMOD_ACT=10 -> idle controlled
- CMOD_ACT=11 -> T/alpha
- CMOD_ACT=12 -> N/T
- CMOD_ACT=13 -> T/N

Then this configuration x/y means x imposed to dynamo while y imposed to engine in control terms, that is to say, for example, considering mode 12 that is the most

common one and most important one, the demanded speed is imposed to dynamo (so actual speed is collected from a dynamo sensor) while demanded torque is imposed to engine (so actual torque is collected from an engine sensor). The connection between dynamo and engine in test bench can be seen in first figures of first chapter of this thesis work.
Created the first two windows of the software layout, one of the most important is realized. Window 3 shows the regression results compared to law requirements to the user. Hundreds of lines of code have been written to fulfil this objective taking into account the EURO VI regulation that has been read and applied from a practical point of view, in the sense that every aspect of the regression had to be extrapolated from the little freedom provided by the law. There are 12 values in this picture: slope results for speed, torque and power, intercept results for speed, torque and power, SEE results for speed, torque and power and r^2 results for speed, torque and power. They have been compared with law threshold and if their value is below that value, "PASSED" can be read, otherwise "FAILED" will appear. It is sufficient to have just one failed result to fail all the regression analysis.

Firstly, the raw vectors have been taken into account: actual values of speed, torque and power together with their demanded values. The demanded values have been filtrated creating a new demanded vector that now takes into account the correspondent mode, because originally the demanded values are not mode dependent but they are general values coming from every sensor of the bench, this means that if for example a test has been performed half in mode 12 and half in mode 13, the reference values are one for all the test in mode 12 and one for all the test in mode 13, using CMOD_ACT array a new reference vector that contains mode 12 vector values until CMOD_ACT is set to 12 and mode 13 vector values for the other half. Usually more than two modes are used in a test, so the previous one has to be taken as a simple example to explain what has been done. This filter is applied to speed, torque and power separately. Actual values and new created demand values are just some of the inputs for regression formula: alpha values, idle speed, test frequency and many others are inserted too. First things to do now is to carefully read the EURO VI regulation to understand how the standard requires the regression analysis to be performed [8]. The method of least squares shall be used with equation: $y = a_1x + a_0$.



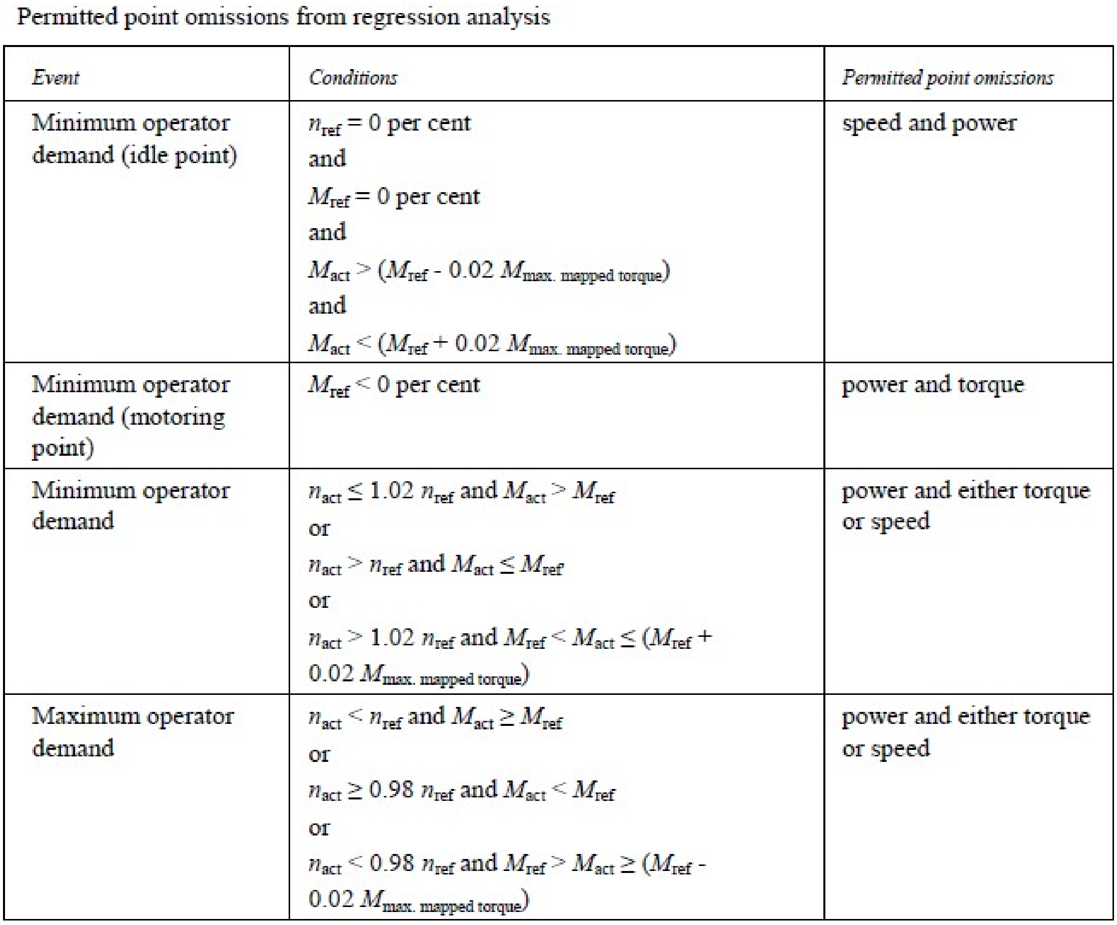
Regression line tolerances for the WHTC

	Speed	*Torque*	*Power*
Standard error of estimate (SEE) of y on x	maximum 5 per cent of maximum test speed	maximum 10 per cent of maximum engine torque	maximum 10 per cent of maximum engine power
Slope of the regression line, a_1	0.95 to 1.03	0.83 - 1.03	0.89 - 1.03
Coefficient of determination, r^2	minimum 0.970	minimum 0.850	minimum 0.910
y intercept of the regression line, a_0	maximum 10 per cent of idle speed	±20 Nm or ±2 per cent of maximum torque whichever is greater	±4 kW or ±2 per cent of maximum power whichever is greater

Figure 2.5: Tolerances of WHTC

Furthermore, the law guarantees a time shift that can be applied to actual values, the standard says that due to the delays that can be met between an imposed value like a reference torque and a collected value like the actual torque from sensor, the actual values can be shifted in time in a range that goes from -3 s to +3 s, for this reason the formula has been implemented inside an infinite while loop letting the delay changes from to -3 and +3 at every iteration. Most important values are the ones closer to 0s so it would be useless to go from -3 to +3, a smarter way to operate is to get the delay value at each iteration from the result of a particular alternating numerical series.

In this way every value will be exploited with a 0,1 step, so the results will be 0.1 , -0.1 , 0.2 , -0.2 , ... 3, -3. Obviously, this is not mandatory and it is not mandatory too, once applied the shift, to reach 3/-3. This means that a condition that will stop the while execution will be added when the regression has finally passed or when the delay will be over 3 or under -3, saying to the user that regression analysis has not passed. In addition, another shift is needed because starting and finishing points of every curve are not needed and maybe their weight can ruin the computations that will be performed later, because they will be based on average values too. The EURO VI standard also gives the possibility to delete some points from the curves, this is done if they satisfy particular conditions under which their influence is not that important for regression calculation purposes, this will help achieving satisfactory results, because there will be less points to analyse.



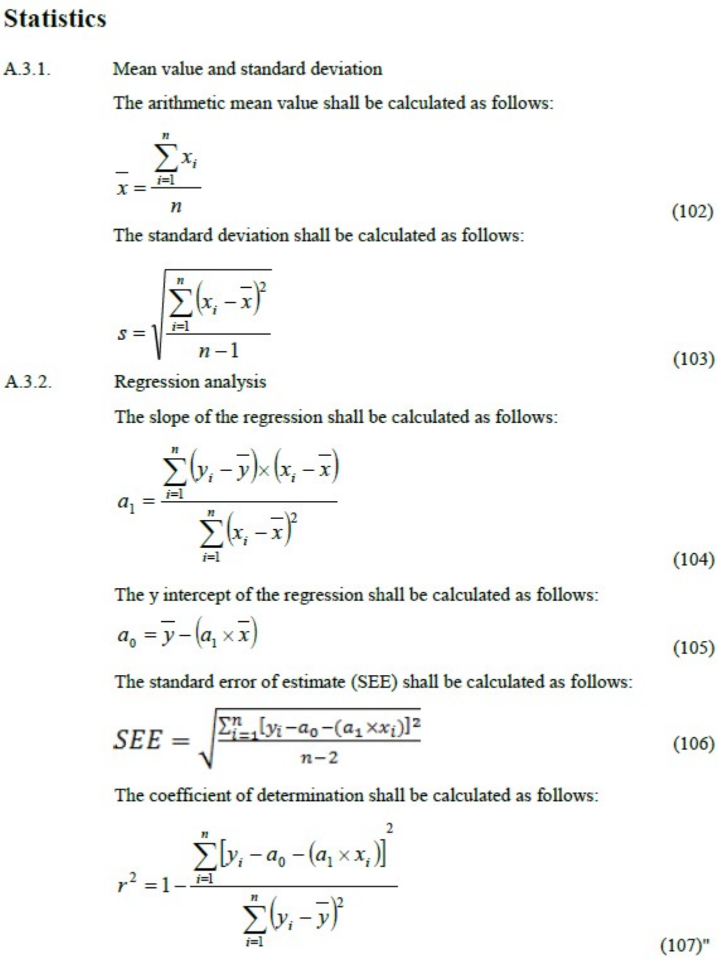
Permitted point omissions from regression analysis

Event	*Conditions*	*Permitted point omissions*
Minimum operator demand (idle point)	n_{ref} = 0 per cent and M_{ref} = 0 per cent and $M_{\text{act}} > (M_{\text{ref}} - 0.02\ M_{\text{max. mapped torque}})$ and $M_{\text{act}} < (M_{\text{ref}} + 0.02\ M_{\text{max. mapped torque}})$	speed and power
Minimum operator demand (motoring point)	$M_{\text{ref}} < 0$ per cent	power and torque
Minimum operator demand	$n_{\text{act}} \leq 1.02\ n_{\text{ref}}$ and $M_{\text{act}} > M_{\text{ref}}$ or $n_{\text{act}} > n_{\text{ref}}$ and $M_{\text{act}} \leq M_{\text{ref}}$ or $n_{\text{act}} > 1.02\ n_{\text{ref}}$ and $M_{\text{ref}} < M_{\text{act}} \leq (M_{\text{ref}} + 0.02\ M_{\text{max. mapped torque}})$	power and either torque or speed
Maximum operator demand	$n_{\text{act}} < n_{\text{ref}}$ and $M_{\text{act}} \geq M_{\text{ref}}$ or $n_{\text{act}} \geq 0.98\ n_{\text{ref}}$ and $M_{\text{act}} < M_{\text{ref}}$ or $n_{\text{act}} < 0.98\ n_{\text{ref}}$ and $M_{\text{ref}} > M_{\text{act}} \geq (M_{\text{ref}} - 0.02\ M_{\text{max. mapped torque}})$	power and either torque or speed

Figure 2.6: Omitted points

To do this, four different flags have been implemented whose value is 1 when the condition has not been met, while it is 0 when the condition has been met, in this way it is sufficient to reduce the vectors with curve values deleting the ones that are set to 0. When the law standard leaves to operator the choice between points of power and another value to delete, the one that has more satisfactory values is chosen, in order to delete more points.

After this introductory part that is performed in order to follow the regulation requirements, the real regression analysis can be carried out. The standard error estimate of y on x, the slope of the regression line, the coefficient of determination and the intercept of the regression line are computed according to the following equations from this EURO VI manual page:



Statistics

A.3.1. Mean value and standard deviation

The arithmetic mean value shall be calculated as follows:

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n} \qquad (102)$$

The standard deviation shall be calculated as follows:

$$s = \sqrt{\frac{\sum_{i=1}^{n} \left(x_i - \bar{x}\right)^2}{n-1}} \qquad (103)$$

A.3.2. Regression analysis

The slope of the regression shall be calculated as follows:

$$a_1 = \frac{\sum_{i=1}^{n} \left(y_i - \bar{y}\right) \times \left(x_i - \bar{x}\right)}{\sum_{i=1}^{n} \left(x_i - \bar{x}\right)^2} \qquad (104)$$

The y intercept of the regression shall be calculated as follows:

$$a_0 = \bar{y} - \left(a_1 \times \bar{x}\right) \qquad (105)$$

The standard error of estimate (SEE) shall be calculated as follows:

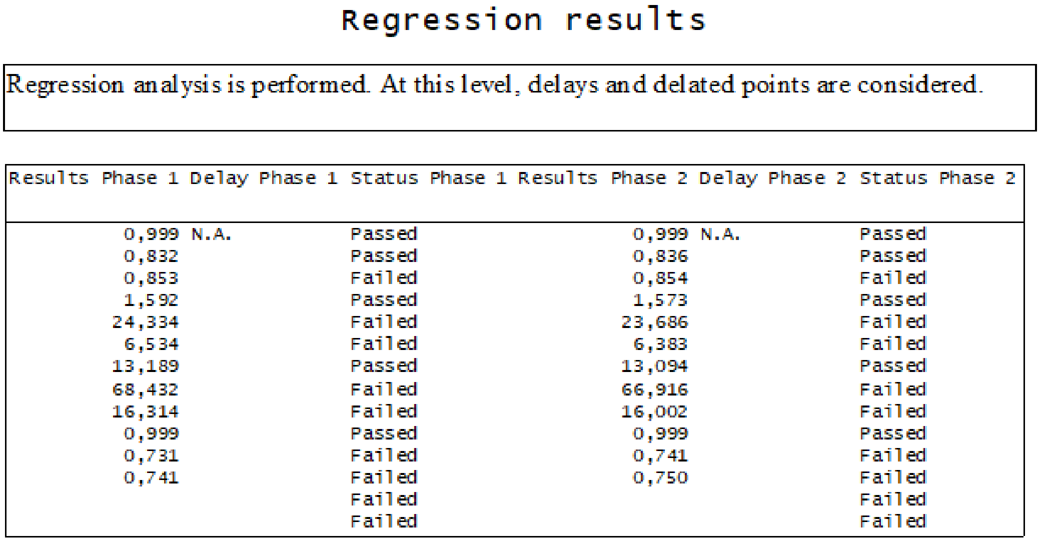
$$SEE = \sqrt{\frac{\sum_{i=1}^{n} [y_i - a_0 - (a_1 \times x_i)]^2}{n-2}} \qquad (106)$$

The coefficient of determination shall be calculated as follows:

$$r^2 = 1 - \frac{\sum_{i=1}^{n} \left[y_i - a_0 - \left(a_1 \times x_i\right)\right]^2}{\sum_{i=1}^{n} \left(y_i - \bar{y}\right)^2} \qquad (107)"$$

Figure 2.7: Statistics

These values are compared to law's threshold inside a while loop, if the regression analysis has passed, the problem is terminated, otherwise the delay is increased with the alternating sequence explained before until -3s/3s, always checking if the new delay condition leads to the satisfaction of the requirements before augmenting it again. The third page of the software layout will tell the regression results to the user. The columns of the results are, in right order, related to: slope for speed, torque and power, intercept for speed, torque and power, SEE for speed, torque and power, r^2 for speed, torque and power. When the regression is failed, the results are shown with delay not applied (N.A.) so equal to 0.



Regression results

Regression analysis is performed. At this level, delays and delated points are considered.

Results Phase 1	Delay Phase 1	Status Phase 1	Results Phase 2	Delay Phase 2	Status Phase 2
0,999	N.A.	Passed	0,999	N.A.	Passed
0,832		Passed	0,836		Passed
0,853		Failed	0,854		Failed
1,592		Passed	1,573		Passed
24,334		Failed	23,686		Failed
6,534		Failed	6,383		Failed
13,189		Passed	13,094		Passed
68,432		Failed	66,916		Failed
16,314		Failed	16,002		Failed
0,999		Passed	0,999		Passed
0,731		Failed	0,741		Failed
0,741		Failed	0,750		Failed
		Failed			Failed
		Failed			Failed

Figure 2.8: Regression results

At the moment, there is the possibility to discriminate between passed and failed tests and it is necessary to look at which are the differences between these tests in order to find out what can be done to improve the failed ones. What is immediately notable is that speed has no regression problems, as said it is generally imposed to dynamo that is perfectly capable to follow the reference, so the attention from now is just about torque. A first approach to this problem was led by computing the regression results in every possible condition, so they have been computed for mode 12 only, for points where the actual torque is bigger than demand one and vice versa, in order to find any systematic behaviour.

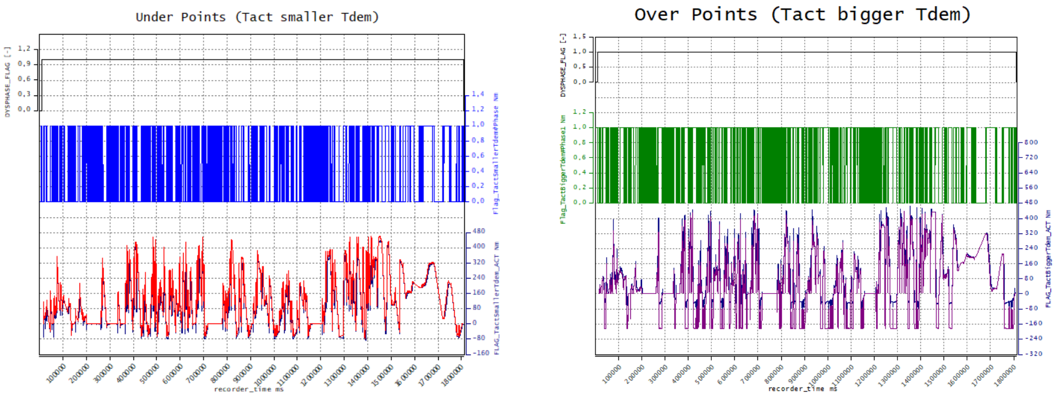
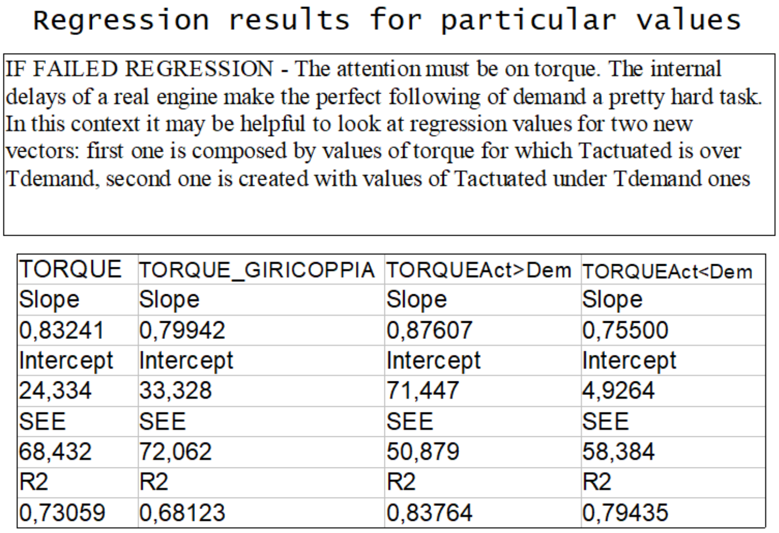


Figure 2.9: Over points are points for which the actuated torque is bigger than demand one, while vice versa holds for Under points



Regression results for particular values

IF FAILED REGRESSION - The attention must be on torque. The internal delays of a real engine make the perfect following of demand a pretty hard task. In this context it may be helpfil to look at regression values for two new vectors: first one is composed by values of torque for which Tactuated is over Tdemand, second one is created with values of Tactuated under Tdemand ones

TORQUE	TORQUE_GIRICOPPIA	TORQUEAct>Dem	TORQUEAct<Dem
Slope	Slope	Slope	Slope
0,83241	0,79942	0,87607	0,75500
Intercept	Intercept	Intercept	Intercept
24,334	33,328	71,447	4,9264
SEE	SEE	SEE	SEE
68,432	72,062	50,879	58,384
R2	R2	R2	R2
0,73059	0,68123	0,83764	0,79435

Figure 2.10: Over and Under regression results

Unfortunately, this approach has given no important idea since the behaviour in this case is similar between cycles whose regression results are positive and cycles whose regression results are not good, but at least it becomes clear that to find noticeable results it is important to not focus only on torque values. The human experience is the key in a case like this where it is hard to find a starting point, the point of view of engineers that work at test bench has been collected. What usually happens during a regression-failed test, is a lack of acceleration due to a wrong PID gains setting, this means that even if a high value of acceleration is set, the engine isn't able to pick up speed in the desired way reaching the wanted torque behaviour. In other words, a way to link acceleration of the engine and reached torque has to be found. The first thing to check is the maximum acceleration that is reached during the test. A new window is created inside the software to show (at the same time reference) the speed, the torque and a flag that is programmed to be equal to 1 every time the acceleration value is equal or bigger than 99.5% of max value. This showed that no cycle has problems reaching at least few times the maximum alpha value.

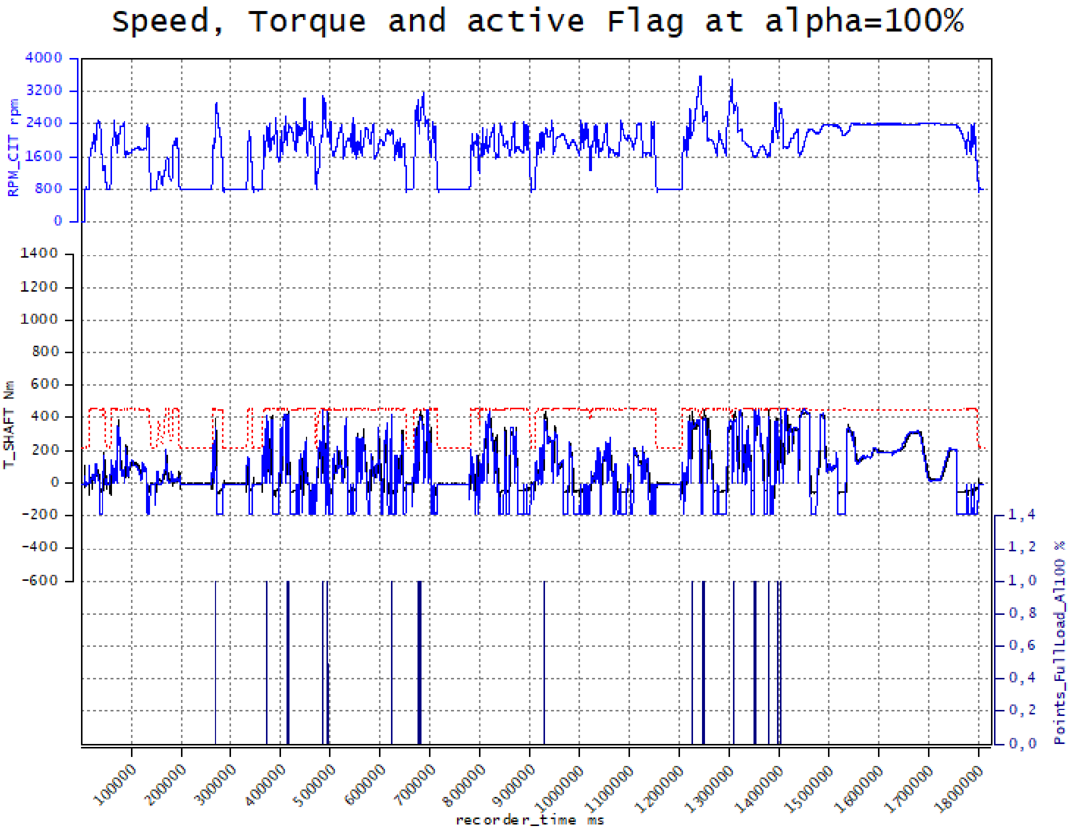


Figure 2.11: Alpha 100 percentage Flag

The just created flag is used to create another diagram of the layout, in fact the time at which the flag value is 1 has been collected and used to get the correspondent speed and torque values at max alpha. These points have been compared with the full load curve and this gave some interesting suggestions, because once seen previously that there are no problem reaching max alpha, comparing these points to full load gives a sort of confirmation because both in regression-passed cycles and regression-failed cycles there is a good number of points that satisfy the max alpha condition.

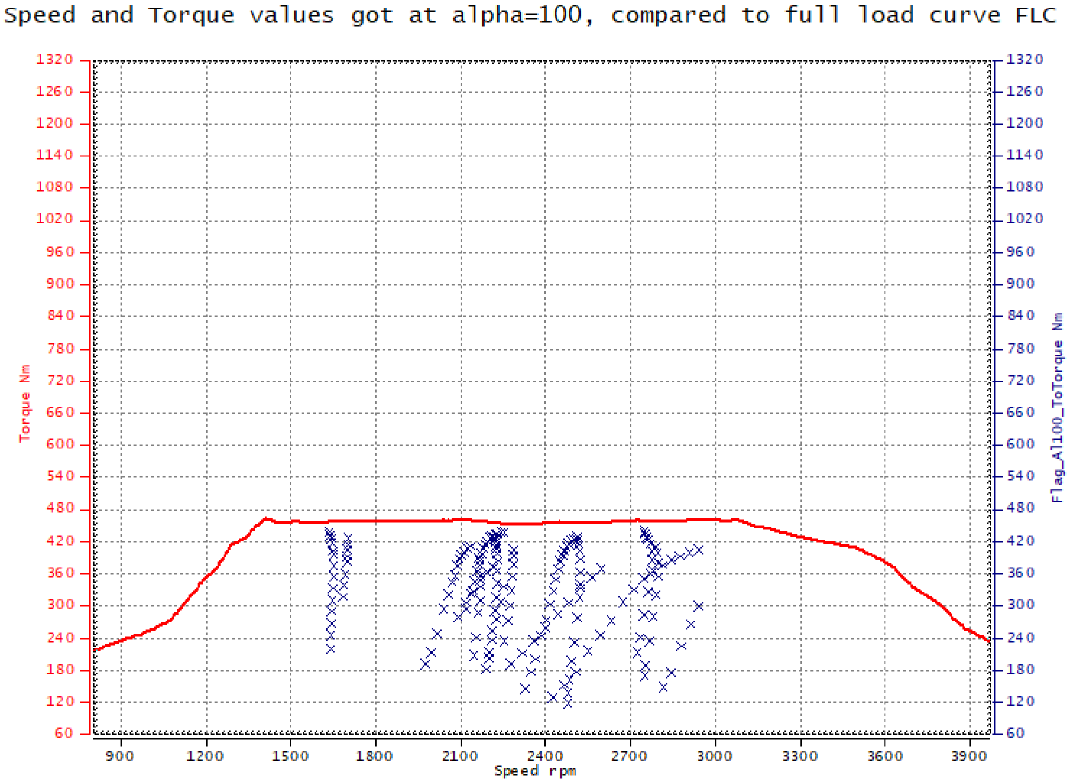


Figure 2.12: Full Load curve

Until now only some generic aspects have been debated and inserted within software's layout, they can be useful information for an user that wants to know how speed, torque and alpha behave. From this time, only PID related considerations can be done, in fact, a very relevant analysis can be performed. Since every cycle has the possibility to reach top acceleration, it is obvious that the problems must be found in the way they reach the top acceleration, this means to get the most critical rise inside the torque curve and look at values that acceleration assumes at that time. This is not an easy task since it can be seen that the torque curve has no regular shape, as it is a real collection of measurements. It is not practical to try working on the real curve. The best way to proceed is to have all torque values inside a vector and work on numbers. In other words, the highest solicitation will be found and analysed because it is very likely the highest acceleration demand will be needed going from 10% to 90% of max torque in minor time (since the irregular shape of the curve will go several times from 10% to 90% of maximum value, the one that goes in less time will be selected), it would be interesting to see how the engine will speed up during this time interval. Again, for the umpteenth time a Concerto formula will be written to bring all these information and related plots inside the layout of the software. The starting point is the vector that contains each value of the torque, collected directly in the test bench with a sampling time equal to 100 ms, going from 88 ms to 1812988 ms, that means 18130 samples: more or less 30 minutes as set by standards. What is done in software is to collect in a new matrix every vector (as a column of the new matrix) composed by any possible rise time computation, this will lead to a matrix whose columns are torque values that go from 10% to 90% of maximum value. This is not a very easy task, since the shape of the curve is so irregular that no common rise time could be computed, maybe the function was going from 10% to 30%, then back to 20%, then up again until 50% and so on, reaching the very high value of 90% of peak so late that the related solicitation was not that meaningful. A solution is also not easy to find by an optimization point of view, in the sense that since a perfect 10% couldn't be found or at least it was not so smart to look perfectly for it (because maybe this would lead to lose a 10.37% - 89.97% rise time that could be important for the analysis), a tolerance must be imposed, it is chosen to look for every point that are below 11% of max value, up to every point that is between 89% and 93% of maximum. The problem is that it is impossible from an optimization point of view to take each one of these points and to see if up to second threshold there is a noticeable situation, the algorithm (only for this formula) deals with almost 20000 points, this would take several minutes to process them all, it has to bear in mind that only for this PID Analysis Tool there will be about a hundred formulas and this has to be a sub-tool inside a bigger software: the complexity is a crucial point, every some seconds time save is a big conquest in this context, this also made the regression formula very difficult to code, because in that circumstance there were also more computations to care about. For this reason, a smart way to process this huge size of data has to be found. First of all, the problem of considering ineffective rise time (like the ones that goes like 10% - 30% - 20% - 50% - 25% - ... - 90%) should be solved by not processing every possible time. This is done by a simple flag whose value is 1 if the successive point is bigger than previous one inside the torque values vector that will be very useful later. The starting point is the vector access, divided between two variables, let's call them i and m and this was a critical point to speed up the code because the access is performed as vector[i+m], if a relevant value is met, only m is augmented up to find 90% of maximum and after that i is updated to i=i+m (so that the next value to check is maybe 300 or 400 values after), otherwise i is augmented until a relevant value is met. The flag that has been computed is useful in this moment, since only the rise time whose sum of correspondent flag values is bigger than 85% of the number of values that it contains will be considered. This will avoid the condition explained before where the torque values go up and down leading to no interesting result. Tens of condition loops are also present to manage every other possibility that can be met during torque vector check. The formula provides very good results in less than one second. Around this formula many other are built to deal with other issues, for example computing the time at which the rise times in the columns of the matrix are computed. Two very important formulas are the one that collects all those information about rise times and timing and select the one that provides highest solicitation in minor time, that is the one that will be used within the real analysis and within the software layout in order to show those information to the user and the other formula is the one that takes this highest solicitation and creates a flag whose value is 1 during that rise time and 0 otherwise, so that plotted within the layout close to torque curve it can enhance the user's experience by showing exactly where this highest demand is located inside the torque curve.

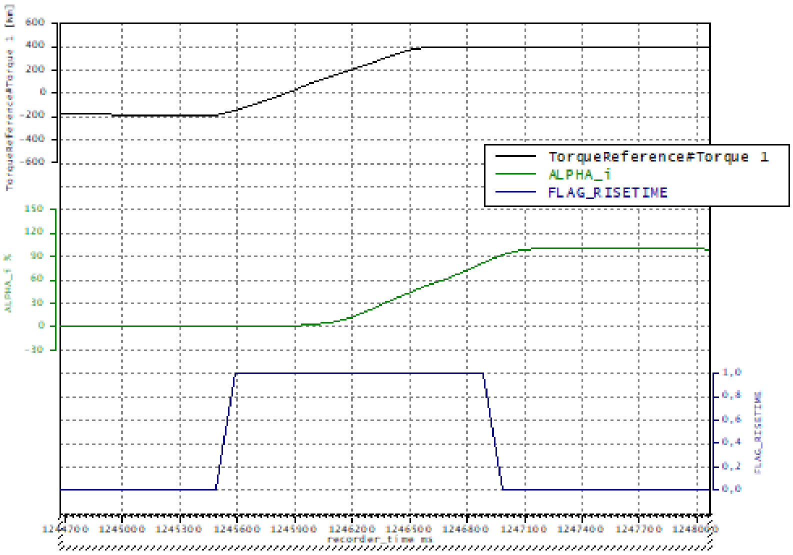
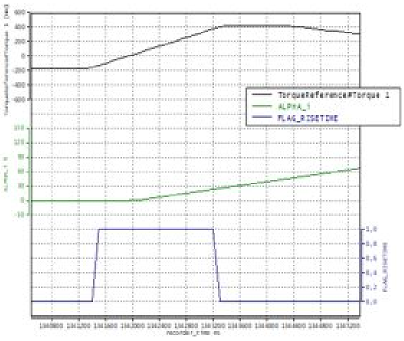


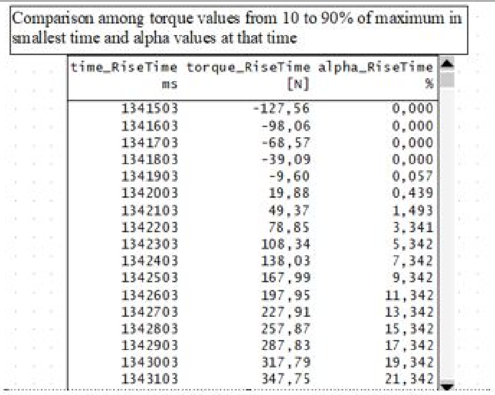
Figure 2.13: Flag rise time

To validate the results, a comparison among several tests is done, considering many regression-passed tests and many regression-failed tests and the results are quite clear: a very good point has been found, every failed test isn't able to get 100% acceleration during this rise time, that is where it is needed the most, more or less every failed-cycle reaches 100% acceleration a bit later, this is a kind of delay that is clearly linked to the PID controller.

STATUS | **FLAG RISETIME** | **RISETIME COMPARISON**

FAILED

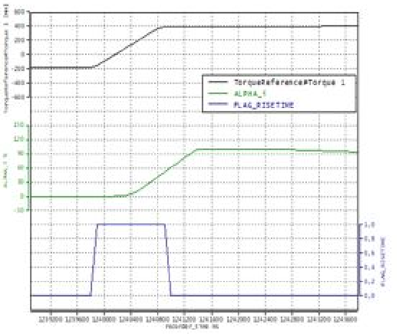


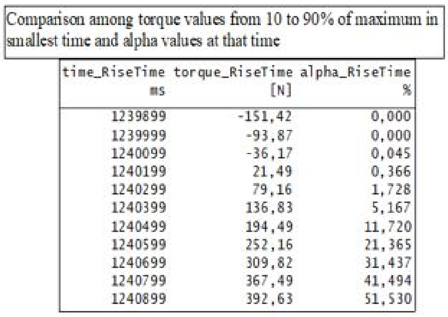


Comparison among torque values from 10 to 90% of maximum in smallest time and alpha values at that time

time_RiseTime ms	torque_RiseTime [N]	alpha_RiseTime %
1341503	-127,56	0,000
1341603	-98,06	0,000
1341703	-68,57	0,000
1341803	-39,09	0,000
1341903	-9,60	0,057
1342003	19,88	0,439
1342103	49,37	1,493
1342203	78,85	3,341
1342303	108,34	5,342
1342403	138,03	7,342
1342503	167,99	9,342
1342603	197,95	11,342
1342703	227,91	13,342
1342803	257,87	15,342
1342903	287,83	17,342
1343003	317,79	19,342
1343103	347,75	21,342

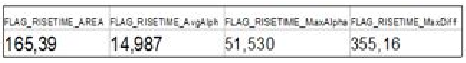
FAILED





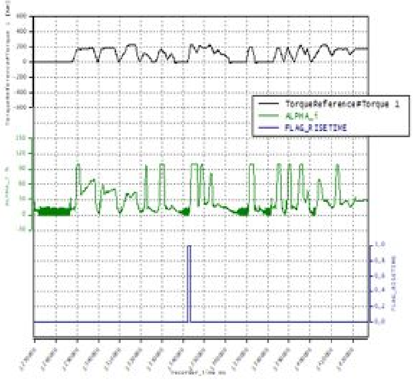
Comparison among torque values from 10 to 90% of maximum in smallest time and alpha values at that time

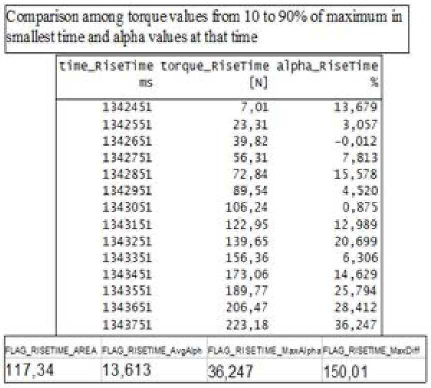
time_RiseTime ms	torque_RiseTime [N]	alpha_RiseTime %
1239899	-151,42	0,000
1239999	-93,87	0,000
1240099	-36,17	0,045
1240199	21,49	0,366
1240299	79,16	1,728
1240399	136,83	5,167
1240499	194,49	11,720
1240599	252,16	21,365
1240699	309,82	31,437
1240799	367,49	41,494
1240899	392,63	51,530



FLAG_RISETIME_AREA	FLAG_RISETIME_AvgAlph	FLAG_RISETIME_MaxAlpha	FLAG_RISETIME_MaxDiff
165,39	14,987	51,530	355,16

FAILED





Comparison among torque values from 10 to 90% of maximum in smallest time and alpha values at that time

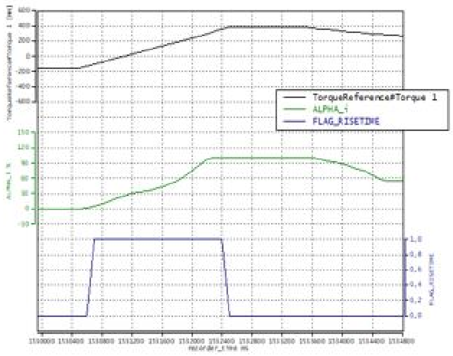
time_RiseTime ms	torque_RiseTime [N]	alpha_RiseTime %
1342451	7,01	13,679
1342551	23,31	3,057
1342651	39,82	-0,012
1342751	56,31	7,813
1342851	72,84	15,578
1342951	89,54	4,520
1343051	106,24	0,875
1343151	122,95	12,989
1343251	139,65	20,699
1343351	156,36	6,306
1343451	173,06	14,629
1343551	189,77	25,794
1343651	206,47	28,412
1343751	223,18	36,247

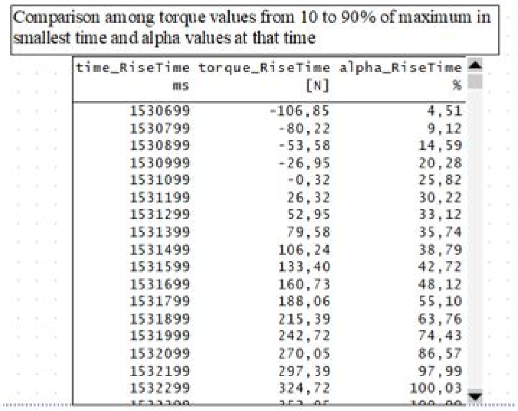
FLAG_RISETIME_AREA	FLAG_RISETIME_AvgAlph	FLAG_RISETIME_MaxAlpha	FLAG_RISETIME_MaxDiff
117,34	13,613	36,247	150,01

Figure 2.14: Comparative among failed cycles

STATUS **FLAG RISETIME** **RISETIME COMPARISON**

PASSED

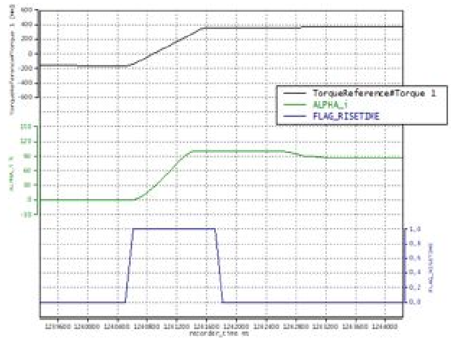


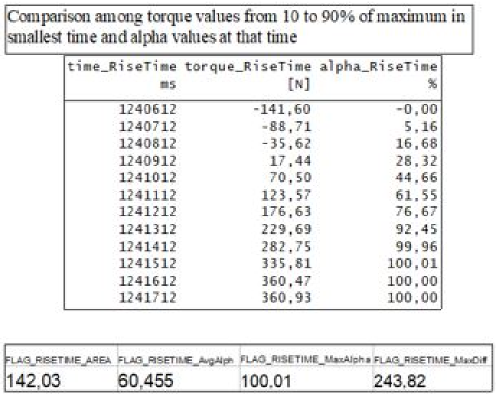


Comparison among torque values from 10 to 90% of maximum in smallest time and alpha values at that time

time_RiseTime ms	torque_RiseTime [N]	alpha_RiseTime %
1530699	-106,85	4,51
1530799	-80,22	9,12
1530899	-53,58	14,59
1530999	-26,95	20,28
1531099	-0,32	25,82
1531199	26,32	30,22
1531299	52,95	33,12
1531399	79,58	35,74
1531499	106,24	38,79
1531599	133,40	42,72
1531699	160,73	48,12
1531799	188,06	55,10
1531899	215,39	63,76
1531999	242,72	74,43
1532099	270,05	86,57
1532199	297,39	97,99
1532299	324,72	100,03

PASSED



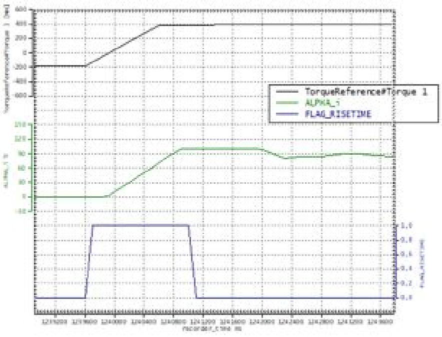


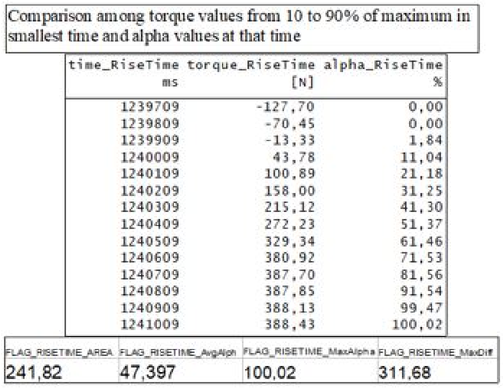
Comparison among torque values from 10 to 90% of maximum in smallest time and alpha values at that time

time_RiseTime ms	torque_RiseTime [N]	alpha_RiseTime %
1240612	-141,60	-0,00
1240712	-88,71	5,16
1240812	-35,62	16,68
1240912	17,44	28,32
1241012	70,50	44,66
1241112	123,57	61,55
1241212	176,63	76,67
1241312	229,69	92,45
1241412	282,75	99,96
1241512	335,81	100,01
1241612	360,47	100,00
1241712	360,93	100,00

FLAG_RISETIME_AREA	FLAG_RISETIME_AvgAlph	FLAG_RISETIME_MaxAlpha	FLAG_RISETIME_MaxDiff
142,03	60,455	100,01	243,82

PASSED





Comparison among torque values from 10 to 90% of maximum in smallest time and alpha values at that time

time_RiseTime ms	torque_RiseTime [N]	alpha_RiseTime %
1239709	-127,70	0,00
1239809	-70,45	0,00
1239909	-13,33	1,84
1240009	43,78	11,04
1240109	100,89	21,18
1240209	158,00	31,25
1240309	215,12	41,30
1240409	272,23	51,37
1240509	329,34	61,46
1240609	380,92	71,53
1240709	387,70	81,56
1240809	387,85	91,54
1240909	388,13	99,47
1241009	388,43	100,02

FLAG_RISETIME_AREA	FLAG_RISETIME_AvgAlph	FLAG_RISETIME_MaxAlpha	FLAG_RISETIME_MaxDiff
241,82	47,397	100,02	311,68

Figure 2.15: Comparative among cycles

2.3 Simulation

Now the architecture of the PID has to be analysed in details to understand which parameter can be related to this phenomenon and how it has to be modified in order to enhance acceleration response speed during the transients. Then other problems about the test results should be found and related to other PID parameters. To do this, the EMCON 400/401 manual by AVL (the test bench manual) has to be carefully read to understand the nature of the control of a so nonlinear system [9]. This manual is an user's guide that deals with operational safety for the hardware usage, general information about the test cell and the PUMA software and operative instructions about how to use them. Chapter 6 deals with the engine controllers. A generalized ENPAC structure can be found in order to provide easier parameterization for various control applications, such as engine speed control, torque control and control of an external quantity. The PID controller is in reality a PI controller since no derivative contribution is present, furthermore additional nonlinearities must be considered, it is like a dynamic adaptation of the proportional and integral gain. Looking at the general ENPAC structure in the guide, a simple simulation of the test bench can be carried out using MATLAB R2014a and in particular Simulink, that is a MATLAB-based graphical programming environment for modelling, simulating and analysing multidomain dynamical systems. This will be important to start being familiar with the test bench and its parameters.

To take into account these nonlinearities in a bench simulation, the chosen control technique is Gain Scheduling. In control theory, gain scheduling is a way to model a non-linear system that uses some linear controllers, each of that provides satisfactory control for a different working point of the system. An arbitrary number of variables, called the scheduling variables, are used to determine what operating region the system is currently in and to enable the appropriate linear controller. It is widely used in aeronautics where a very common control system is the one that employs altitude and Mach number as the scheduling variables, with different linear controller parameters available that will be used inside the controller, in this way a nonlinear system will be studied through a linear approach. Dealing with this test bench, a filtered version of the error signal is chosen as scheduling variable, this means that the controller gains are error-dependent, and this makes sense since it would be useful to have a less aggressive control when the error is not that high, while otherwise an aggressive control could be necessary. The error is the difference between the demand value (imposed) and the actual value (what is collected from sensors). From EMCON user guide, a general overview of the controller structure can be seen.

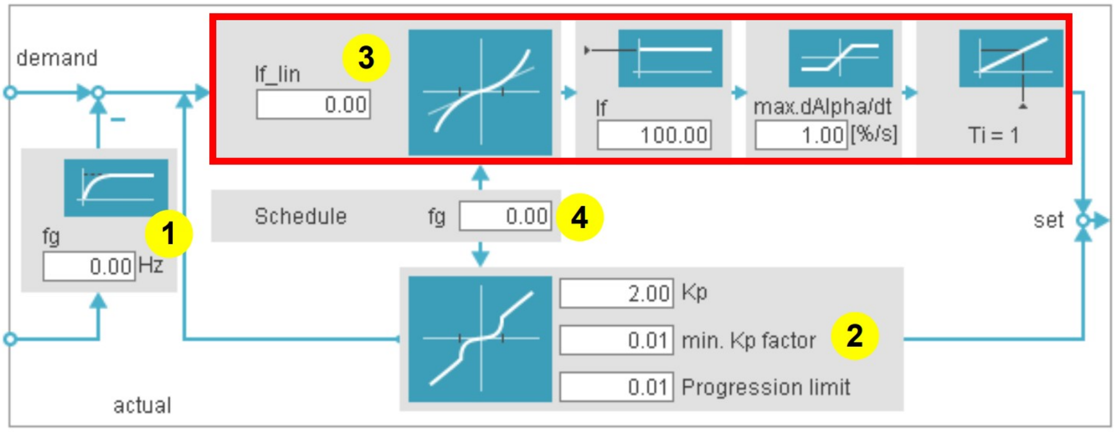


Figure 2.16: Emcon

The part that is selected in red is the integral part, while below on the other branch there is the proportional part. It can be seen that the actual contribution is filtered by a lowpass second order filter in order to cut the unwanted measuring noise from the actual value: a correct choice of the cutoff frequency is specified by the guide to be around 15 Hz, while for speed control applications, the filter is usually not used. Concerning all the other parameters in the scheme, the experience of the engineers working on the test bench and the user manual have been mixed together to have a clear understanding of their role. Starting from proportional part, there are three parameters to look at:

- "Kp" is the proportional gain as used during a linear approach, it directly reacts to the error with a reaction that is stronger for growing Kp values. Usually, Kp value goes from 0 to 1000;

- "min. Kp factor" is a reduction factor of the proportional gain implemented as Kp* min. Kp factor, this is active if the scheduling variable is within the range defined by the Progression limit parameter. A meaningful value range per "min. Kp factor" is between 0 and 1;

- "Progression limit" defines the range of the scheduling variable in which the gain reduction is active. It is set as a percentage of the input range that is the difference between maximum and minim value of the controlled quantity. Meaningful value range for progression limit is between 0 and 10%. Setting it to 0 means to deactive.

In other words, the proportional part will be simulated by using a linear proportional gain outside a range (defined by progression limit) where a softer action is needed on the error and a nonlinear function inside this range where an harder action is needed and this nonlinearity will be approximated due to gain scheduling control.
Looking at integral part, the parameters used for the controller's integral part are as follows:

- "If" is the integral gain that will multiply the nonlinear function that is the white function in the box marked with 3 on the previous figure. This nonlinear function is called FInt by the EMCON manual's author;
- "If_lin" is used to parameterize the nonlinear function FInt, in the sense that if the scheduling variable is within the range defined by If_lin (this range goes from -If_lin to If_lin), FInt has a linear behaviour, otherwise it is a nonlinear function. This is similar to what is done in the proportional part. Value is between 0 and 1 where 0 means deactivated and 1 means linear relationship over the entire range;
- "max.dAlpha/dt" is the maximum rate of change of the integral part (rate limiter), this is implemented as a limitation of the input signal to the integrator. Value is between 0 and 100%;
- "Ti" represents the integrator.

To ensure compatibility of the controller settings for various types of engines, normalization is needed. This is helpful in an environment like a test bench where many engines are placed, in fact, if an engine is replaced, the controller doesn't need to be returned. Quantities in the normalized format no longer have physical units. In this context, ENPAC is used as an engine torque controller, this means that, being the two input, as said, the torque demanded and the actual torque, the output will be the set value for the throttle position. On the user manual there is also the procedure to adjust an ENPAC engine torque controller:

- Run the engine at the speed with the highest torque;

- Set If to 0;
- Set the noise filter cutoff frequency fg to 0;
- Set max.dAlpha/dt to 0;
- Perform a series of step tests and gradually increase Kp to the point at which the control loop starts to oscillate;
- Reduce Kp back to a stable point;
- Gradually increase max.dAlpha/dt and If simultaneously until the control loop starts to oscillate again;
- Reduce If back to a stable point and then set max.dAlpha/dt to a small value;
- Check if the control loop is stable over the entire operating range; If necessary, repeat the tuning procedure.
- Set the noise filter cutoff frequency fg between 15 and 30 Hz.

The actual Simulation has been carried out on Simulink and here is the scheme:

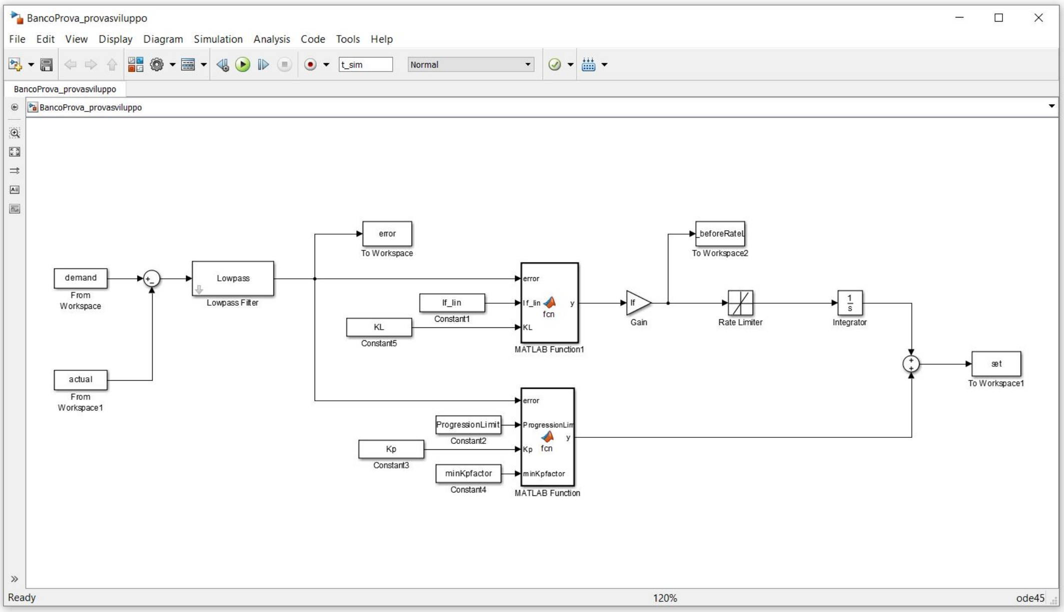


Figure 2.17: Simulink Scheme

A bench simulation is really important due to the costs that are behind a real simulation in the test bench. It is not possible to test every little change using the real hardware, so a totally virtual domain gives the chance to test everything so that the tests in the real cell will be maximized and performed only when many virtual simulations have been carried out with good results. From Concerto environment, the demand and actual values are saved in a .mat file and they are imported on a MATLAB script, from which they are used in the simulation. The MATLAB script is used to represent all the PI parameters just described (Kp, minKpfactor, ProgressionLimit, If_lin, max.dAlpha/dt, Ti). The error is computed by simply subtracting the actual values from demand ones. This result is usually filtered between 1 and 5 Hz because the scheduling variable should change relatively slowly. There was no way to simulate the two nonlinear functions (the one in proportional part and the one in integral part) using blocks since there are not related to common applications. Best way to simulate them is to use a MATLAB function and manually code the behaviour of the two curves, that are respectively the behaviour of proportional and integral gain when the scheduling variable varies.

To code the proportional part, three states have been considered. In first state an high action is needed since the scheduling variable is below the negative value of the progression limit, this means that the action will be equal to Kp*error, without any reduction. In second state a soft action is needed since the scheduling variable is between its negative and positive value, this means that the action will be equal to Kp*minKpfactor*error, with a significant reduction. In third state an high action is needed since the scheduling variable is above the positive value of the progression limit, this means that the action will be equal to Kp*error, without any reduction.

To code the integral part, three states have been considered. In first state an high action is needed since the scheduling variable is below the negative value of the If_lin, this means that the action will be equal to |error|*error, without any reduction. In second state a soft action is needed since the scheduling variable is between its negative and positive value, this means that the action will be equal to KL*error, with a significant reduction. KL is calculated to ensure a smooth transition between the states. In third state an high action is needed since the scheduling variable is above the positive value of the If_lin, this means that the action will be equal to |error|*error, without any reduction.

Furthermore, the integral part is multiplied by gain If. The maximum rate of change

of the integral part is limited simulating the max.dAlpha/dt action by coding this maximum rate of change through another MATLAB function. Ti is modelled using an integrator block. The integral part and the proportional part are then joint together to compute the set values.
The limits of this simulation scheme are related to the variety of modes that are used in a real test. As said in introductory chapter 1, the test is composed by many modes that determine the way in which engine and dynamo are controlled. This scheme only deals with dynamo controlled in terms of speed and engine controlled in terms of torque, making quite weak the application to this simulation to a complex analysis of the system, however is has to be considered a starting point to play with the EMCON parameters just to understand how they change the physical responses of the system engine.
The results are quite good in the corresponding mode, while they are totally different at the beginning, at the end and in few parts in the middle where different modes are used to control the bench, nevertheless in the corresponding mode, when the real acceleration is almost 0%, the simulation provides a very low acceleration, while where it is almost 100%, the simulation provides high acceleration, this means that this scheme is a quite good starting point.

One important aspect to deal with is the fact that the PI parameters that will be used in MATLAB script can't be directly taken into account, there is no way to collect these parameters and import them on MATLAB because they are not present in Concerto environment either. In this regard, a very important formula has been written to collect these parameters directly from the PUMA interface. This piece of code takes into account all the results coming from the test bench and it shows to the user a matrix where the PI EMCON parameters are displayed.

2.4 Software development

At this point, a way to get all PI parameters, a way to simulate the test bench and a starting point for PI parameters correction (related to acceleration behaviour during the most critical torque augmentation) have been found. Now it is time to develop the crucial point of the software, this means to compute the right correction for the controller. In this regard, every parameter of the EMCON scheme will be analysed and many corrections will be proposed related to the particular condition that the tool will be capable to understand from the test it is dealing with.

The best way to understand every effect of a parameter correction is to simulate many times the same test, every time with a different PI configuration. This enhances the importance of the simulation that now becomes an indispensable tool for going on with the thesis work. The simulations are carried out using a particular AVL hardware simulator bench that is capable to replicate all the engine dynamics, in this way it is possible to collect every time the engine curves related to the new set of PID parameters. At this point the Simulink simulation is not used anymore since it is too weak for precise simulations, it has been created just to have a small idea of the test bench, trying to play with the parameters to understand the general behaviour. It has not been required to prepare a more precise simulation since the plant has advanced simulators. The simulated engine is part of Iveco F5H family. All the results in terms of speed and consequently power regression analysis will be poor since the controller of the dynamo is out of the scope of this thesis, so it will not be considered and tuned properly. This is a choice performed together with FPT engineers because the dynamo is perfectly capable to follow the reference during a real test, for this reason a tool that looks also for dynamo tuning is useless. Last thing to point out before the simulations can start is the fact that they are, as said, simulations and since the simulated systems are very complex entities, the results will not be 100% precise but it doesn't matter since what is important in this phase is to collect information about the behaviour of the physical quantities involved in the regression analysis. For example, values like r^2 equal to 1 (100%) will be met, this is very difficult to find in a real physical system, because it means that the line perfectly fits the data. Nevertheless, it is a clear indication that the results are very good.

First choice of the parameters is performed. This test is performed with a very low proportional contribution, while the integral part shows a large max.dAlpha/dt. The other values in integral part are not modified from the standard configuration, this is because the attention is now on the effect of the proportional part. The results of the test show a poor acceleration response during the critical rise time, this is the expected behaviour, because the effect of the proportional part is clearly directly linked to the acceleration response. For sure, the internal logic of the software will deal with proportional gain augmentation when the alpha response during critical solicitation is that poor. That's a good first point, it is now evident how to deal with the proportional part.

Table 2.1: First test results

PID	RiseTime [ms]	Torque_RiseTime [N]	Alpha_RiseTime [%]
Kp=	1247684	-106.01	0.061
0.30	1247784	-67.10	1.422
max.dAlpha/dt=	1247884	-27.92	4.246
90%	1247984	11.24	8.194
	1248084	50.39	13.121
	1248184	89.54	18.535
	1248284	128.70	24.386
	1248384	167.85	30.618
	1248484	207.00	37.174
	1248584	246.16	44.035
	1248684	266.38	49.716

Second choice of parameters is performed. This test is performed with the same low proportional contribution and this time even the integral part will show a small contribution, this can be seen by max.dAlpha/dt that is now lowered to 20%. The expected result for this test is a poorer response by the acceleration pedal because the max.dAlpha/dt deals with the maximum rate of change of the integral part, to lower this value literally means to press with less energy the acceleration pedal of a car. The results of the simulation show what has been said. The performance is even poorer than before, the acceleration is very slow and clearly it is not sufficient to let throttle follow the ideal reference. It is now possible to understand how to deal with a low acceleration during the transients, the software will have to look at these alpha values and make a choice based on this reasoning: if the alpha values are too low during transients, the max.dAlpha/dt should be augmented; if the alpha values are not that low, but the actual torque curve can't manage to follow the reference one, there is a direct lack of acceleration, that 70% for example doesn't carry to a strong response in terms of torque, this means that the proportional gain should be augmented. This task in not so easy because if the proportional part is augmented too much, the simulation will surely show the presence of overshoots, that are a direct response to a smaller transient and in general are not part of a desired behaviour. Particular attention will be paid on that expect later on, when the final parameters computation will be addressed.

Table 2.2: Second test results

PID	**RiseTime** [ms]	**Torque_RiseTime** [N]	**Alpha_RiseTime** [%]
Kp=	1247649	-97.68	0.827
0.30	1247749	-58.47	2.708
max.dAlpha/dt=	1247849	-19.30	5.514
20%	1247949	19.85	9.733
	1248049	59.00	14.338
	1248149	98.16	18.536
	1248249	137.31	22.509
	1248349	176.46	26.417
	1248449	215.62	30.323
	1248549	254.56	34.255
	1248649	266.72	36.072

Third choice of parameters is performed. This test is performed as a proof of the conclusions about dealing with acceleration that has been found out after previous simulation. In this regard, a high proportional contribution is used while the same low value is chosen for the maximum rate of change of the integral part. Looking at the results, it is confirmed that an higher acceleration is present, as it is desired, but the low rate of change of the integral part leads to a poor general performance, because the rise time is not sufficient at that rate of change to keep track of the reference curve.

Table 2.3: Third test results

PID	**RiseTime** [ms]	**Torque_RiseTime** [N]	**Alpha_RiseTime** [%]
Kp=	1247614	-94.97	3.363
0.80	1247714	-55.72	10.317
max.dAlpha/dt=	1247814	-16.56	16.207
20%	1247914	22.59	21.237
	1248014	61.74	26.269
	1248114	100.90	31.600
	1248214	140.05	37.027
	1248314	179.20	43.273
	1248414	218.36	48.740
	1248514	256.88	54.030
	1248614	266.76	52.884

It is now time to deal with min.Kpfactor and Progression limit, but since their effect is not so evident on the overall performance, at least in a first analysis their modification will be omitted.

Lastly, the contributions of the integral gain of the controller (If) and the nonlinear function parameterization (If_lin) should be taken into account. Two simulations are carried out. First test parameters are reused (max.dAlpha/dt = 90% and Kp=0.30), but this time a value equal to 0.70 is chosen to be assigned to If_lin. The effect of the nonlinear function on the overall test is very hard to find, for this reason the software will follow a simplified approach: the effect of those two parameters are computed on the regression result and not on a physical quantity like the acceleration or the torque, the effect of every change of these parameters is linked to regression analysis and it will be found how to modify them in order to improve the results in terms of regression line slope, intercept, standard error of estimate (SEE) and coefficient of determination (r^2). The regression results before lowering If_lin from the maximum value to 0.70 is:

Table 2.4: Regression test standard results

PID	Name	Value	Limits
If_lin=	SlopeRPM	0.11	0.95<x<1.03
Maximum	SlopeTORQUE	0.97	0.83<x<1.03
If=	SlopePOWER	0.96	0.89<x<1.03
1	InterceptRPM	960.76	-inf<x<0
Kp=	InterceptTORQUE	2.20	-20<x<+20
0.30	InterceptPOWER	0.57	-4<x<+4
max.dAlpha/dt=	SEE_RPM	101.08	-inf<x<76.854
90%	SEE_TORQUE	15.88	-inf<x<36.569
minKpfactor=	SEE_POWER	1.75	-inf<x<4.311
0.10	R2_RPM	0.65	0.97<x<+inf
ProgressionLimit=	R2_TORQUE	0.97	0.85<x<+inf
0.50	R2_POWER	0.97	0.91<x<+inf

The regression results after lowering If_lin from the maximum value to 0.70 is:

Table 2.5: Regression test results with If_lin=0.70

PID	Name	Value	Limits
If_lin=	SlopeRPM	0.11	0.95<x<1.03
0.70	SlopeTORQUE	0.88	0.83<x<1.03
If=	SlopePOWER	0.87	0.89<x<1.03
1	InterceptRPM	960.68	-inf<x<0
Kp=	InterceptTORQUE	11.28	-20<x<+20
0.30	InterceptPOWER	1.65	-4<x<+4
max.dAlpha/dt=	SEE_RPM	101.26	-inf<x<76.854
90%	SEE_TORQUE	25.78	-inf<x<36.569
minKpfactor=	SEE_POWER	2.98	-inf<x<4.311
0.10	R2_RPM	0.64	0.97<x<+inf
ProgressionLimit=	R2_TORQUE	0.90	0.85<x<+inf
0.50	R2_POWER	0.91	0.91<x<+inf

The regression results taking back If_lin to its maximum and lowering If gain from standard bench (that is 1) value to 0.50 is:

Table 2.6: Regression test results with If=0.50

PID	Name	Value	Limits
If_lin=	SlopeRPM	0.11	0.95<x<1.03
Maximum	SlopeTORQUE	0.86	0.83<x<1.03
If=	SlopePOWER	0.86	0.89<x<1.03
0.50	InterceptRPM	960.80	-inf<x<0
Kp=	InterceptTORQUE	12.60	-20<x<+20
0.30	InterceptPOWER	1.82	-4<x<+4
max.dAlpha/dt=	SEE_RPM	101.15	-inf<x<76.854
90%	SEE_TORQUE	26.91	-inf<x<36.569
minKpfactor=	SEE_POWER	3.12	-inf<x<4.311
0.10	R2_RPM	0.64	0.97<x<+inf
ProgressionLimit=	R2_TORQUE	0.89	0.85<x<+inf
0.50	R2_POWER	0.90	0.91<x<+inf

Looking at numbers, it can be seen that for a lower value of both If and If_lin the regression results in terms of torque are worse. Basically, both the values with a minimum limit and a maximum limit change: the slope value is lowered, the intercept value is increased, while the SEE (that has only a maximum limit) is increased and the r^2 (that has only a minimum limit) is decreased; for this reason, the overall behaviour is worse than before, since SEE and r^2 are brought closer to their limit. However, the software will make use of lower If and If_lin when SEE and r^2 are sufficiently "safe" and intercept parameter of the regression line needs to be augmented. The vice versa will be much more useful, it is expected to have a better behaviour in terms of slope, SEE, r^2 with a value of if_lin that is a bit higher, this means to have a function that is more linear for the integral contribution. This is tested through another simulation, where again all parameters are set as default and only If_lin is chosen to be an high value, for example 0.95. The expected results are obtained as showed in table.

Table 2.7: Regression test results with If_lin=0.95

PID	Name	Value	Limits
If_lin=	SlopeRPM	0.11	0.95<x<1.03
0.95	SlopeTORQUE	0.92	0.83<x<1.03
If=	SlopePOWER	0.92	0.89<x<1.03
1	InterceptRPM	960.86	-inf<x<0
Kp=	InterceptTORQUE	6.38	-20<x<+20
0.30	InterceptPOWER	1.07	-4<x<+4
max.dAlpha/dt=	SEE_RPM	101.07	-inf<x<76.854
90%	SEE_TORQUE	20.94	-inf<x<36.569
minKpfactor=	SEE_POWER	2.39	-inf<x<4.311
0.10	R2_RPM	0.64	0.97<x<+inf
ProgressionLimit=	R2_TORQUE	0.94	0.85<x<+inf
0.50	R2_POWER	0.94	0.91<x<+inf

The outcomes of the previous simulation (If_lin=0.70) and actual one (If_lin=0.95) are compared to emphasise the concept in following table:

Table 2.8: Comparison table about If_lin parameter augmentation

	If_lin=0.70	**If_lin=0.95**	**Outcome**
SlopeTORQUE	0.88	0.92	better
InterceptTORQUE	11.28	6.38	lower
SEE_TORQUE	25.78	20.94	better
R2_TORQUE	0.90	0.94	better

Lastly, again a simulation is carried out, in order to validate the behaviour of the If gain too. As expected, since it is just a gain that multiplies the nonlinear function parametrized by If_lin value, the effect of a bigger gain should reflect into the same effect of a bigger If_lin. For this purpose, coming back to standard configuration, only the If value is chosen to be different and equal to 2.

Table 2.9: Regression test results with If=2.00

PID	**Name**	**Value**	**Limits**
If_lin=	SlopeRPM	0.11	0.95<x<1.03
Maximum	SlopeTORQUE	1.01	0.83<x<1.03
If=	SlopePOWER	1.00	0.89<x<1.03
2.00	InterceptRPM	960.84	-inf<x<0
Kp=	InterceptTORQUE	-2.68	-20<x<+20
0.30	InterceptPOWER	0.00	-4<x<+4
max.dAlpha/dt=	SEE_RPM	101.04	-inf<x<76.854
90%	SEE_TORQUE	6.09	-inf<x<36.569
minKpfactor=	SEE_POWER	0.52	-inf<x<4.311
0.10	R2_RPM	0.65	0.97<x<+inf
ProgressionLimit=	R2_TORQUE	1.00	0.85<x<+inf
0.50	R2_POWER	1.00	0.91<x<+inf

Now everything is ready to perform, at least in a simulation mode, a complete PID set correction, in order to understand the logic that will then be implemented in software. In this context, two simulations are carried out: an actual PID correction made by a correct calibration and a correction where every parameter is set as high as

possible, in order to show that this task requires attention and a carefully performed design, because exceeding with parameters augmentation or reduction can anyway lead to a failed regression result.
In the first one, the set of parameters is composed by parameters that showed a good result in previous simulations:

- If_lin=0.95;
- If=2;
- Max.dAlpha/dt = 90%;
- Kp=0.80.

Table 2.10: Regression test results with best parameters set

PID	Name	Value	Limits
If_lin=	SlopeRPM	0.11	0.95<x<1.03
Maximum	SlopeTORQUE	0.98	0.83<x<1.03
If=	SlopePOWER	0.97	0.89<x<1.03
2.00	InterceptRPM	960.92	-inf<x<0
Kp=	InterceptTORQUE	0.73	-20<x<+20
0.30	InterceptPOWER	0.37	-4<x<+4
max.dAlpha/dt=	SEE_RPM	100.80	-inf<x<76.854
90%	SEE_TORQUE	8.31	-inf<x<36.569
minKpfactor=	SEE_POWER	0.75	-inf<x<4.311
0.10	R2_RPM	0.65	0.97<x<+inf
ProgressionLimit=	R2_TORQUE	0.99	0.85<x<+inf
0.50	R2_POWER	0.99	0.91<x<+inf

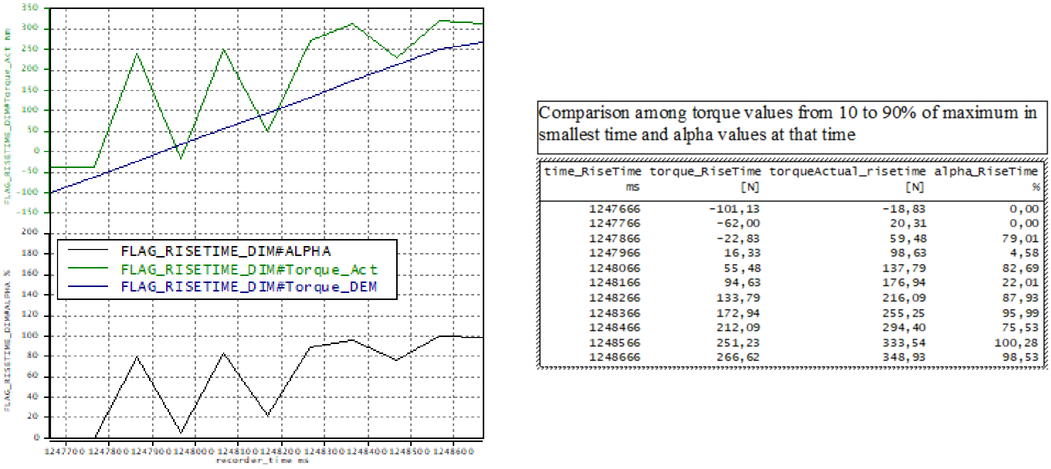
This leads to an interesting result: the combination of the best parameters doesn't lead to the best outcome. This torque behaviour is worse than the one of previous test, where only If=2 was set. In this regard, it is evident that the best approach will be to specifically modify few things but in doing this, it is very important to understand clearly which is the focused action to perform and luckily this was the approach followed during software realization, every action is justified by a particular behaviour of a physical quantity involved in the test, avoiding an approach just focused on a good regression results without taking into account many dynamics during the engine cycle. As last simulation, it is important to point out that the maximum choice of the parameters leads to a failed regression analysis, this means that every time an action is needed, it must be carefully weighted. The set of parameters is composed by:

- If_lin=1.00;
- If=50;
- Max.dAlpha/dt = 100%;
- Kp=1000.

Table 2.11: Regression test results with best parameters set

PID	Name	Value	Limits
If_lin=	SlopeRPM	0.11	0.95<x<1.03
Maximum	SlopeTORQUE	0.93	0.83<x<1.03
If=	SlopePOWER	0.89	0.89<x<1.03
2.00	InterceptRPM	958.08	-inf<x<0
Kp=	InterceptTORQUE	8.45	-20<x<+20
0.30	InterceptPOWER	1.84	-4<x<+4
max.dAlpha/dt=	SEE_RPM	102.90	-inf<x<76.854
90%	SEE_TORQUE	93.44	-inf<x<36.569
minKpfactor=	SEE_POWER	11.35	-inf<x<4.311
0.10	R2_RPM	0.59	0.97<x<+inf
ProgressionLimit=	R2_TORQUE	0.43	0.85<x<+inf
0.50	R2_POWER	0.40	0.91<x<+inf

Regression analysis is failed, as expected. There are problems in terms of SEE and R2. The high Kp value implies that the engine is not able to reply to a so intense input, for this reason the response in terms of torque and acceleration sees ripples and overshoot problems, as a consequence of a so short transient imposed: this can be seen in following figure coming from the software results. The Kp factor is then directly related to the transient behaviour, this conclusion is perfectly related to the role Kp has until now, since it was modified as a consequence of needs coming from rise time operations.



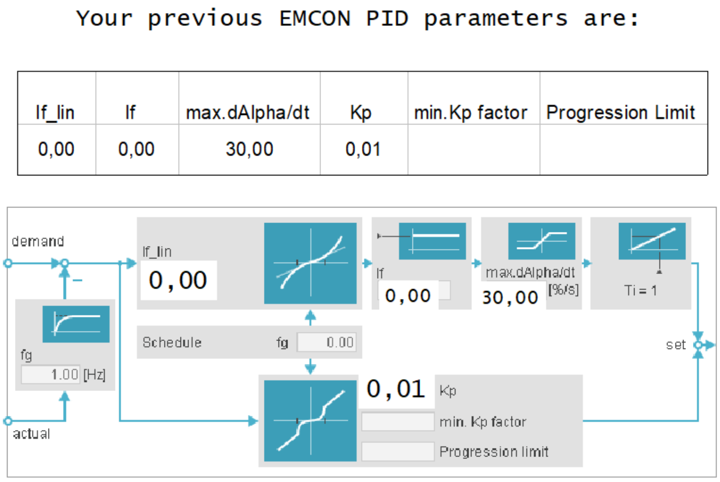
Comparison among torque values from 10 to 90% of maximum in smallest time and alpha values at that time

time_RiseTime	torque_RiseTime	torqueActual_risetime	alpha_RiseTime
ms	[N]	[N]	%
1247666	-101,13	-18,83	0,00
1247766	-62,00	20,31	0,00
1247866	-22,83	59,48	79,01
1247966	16,33	98,63	4,58
1248066	55,48	137,79	82,69
1248166	94,63	176,94	22,01
1248266	133,79	216,09	87,93
1248366	172,94	255,25	95,99
1248466	212,09	294,40	75,53
1248566	251,23	333,54	100,28
1248666	266,62	348,93	98,53

Figure 2.18: On the left, the curves related to physical quantities. On the right, the numerical results.

Everything has been said as conclusion of these simulations should now be implemented in software.

New formulae are written. First of all, the layout was updated inserting the PID values formula, that is capable to show to the user the parameters before any modification, as used by test bench automation.



```
Your previous EMCON PID parameters are:
```

lf_lin	lf	max.dAlpha/dt	Kp	min.Kp factor	Progression Limit
0,00	0,00	30,00	0,01		

Figure 2.19: Initial PID parameters set

The user, even if not so expert in terms of PI bench control, can easily recognize the parameters due to a picture of the controller on which the values are placed. It is important to stay more general as possible and to be clear because not every user will be aware of the bench control, probably there are users that will just launch some tests and maybe they have never seen the controller of the cell. This enhances the usage of the software to any level of the user.

The idea is now to split the PI modification in two big formulae. This is again due to the variety of users that are going to make usage of this tool. There are two kind of users: the ones that are not aware of the controller and they are just trying to homologate the engine by performing a certain test and the ones that are expert in terms of bench hardware, so they know about the internal dynamics of the cell. For this reason, the tool will provide both the new PI values for a less expert user and it will also provide some suggestions, that are sentences that explain why a certain parameter will change and what are the physical considerations that lead to this. This will be helpful for an expert user because maybe he has to change some other settings on the control panel and in this way he will be aware of the problems that are linked to that engine test. The suggestion-formula is coded and then, on the basis of the suggestions, the right control action will be computed in another formula that will end up producing as output a matrix that contains the new PI values.

The piece of code that deals with suggestions starts computing the acceleration values during some specific instants of time, like the rise time, the time related to most significant torque and speed during the cycle. The acceleration curve is carefully checked looking for unwanted overshoot and its characteristic is studied to find out if it is behaving as expected or not. On the basis of this analysis, some suggestions about Kp and max.dAlpha/dt are deduced. Furthermore, the suggestions can be summed up as:

- "Must increment a lot";
- "Must increment";
- "Must increment a bit";
- "Do not modify";
- "Reduce a bit";
- "Reduce";
- "Reduce a lot".

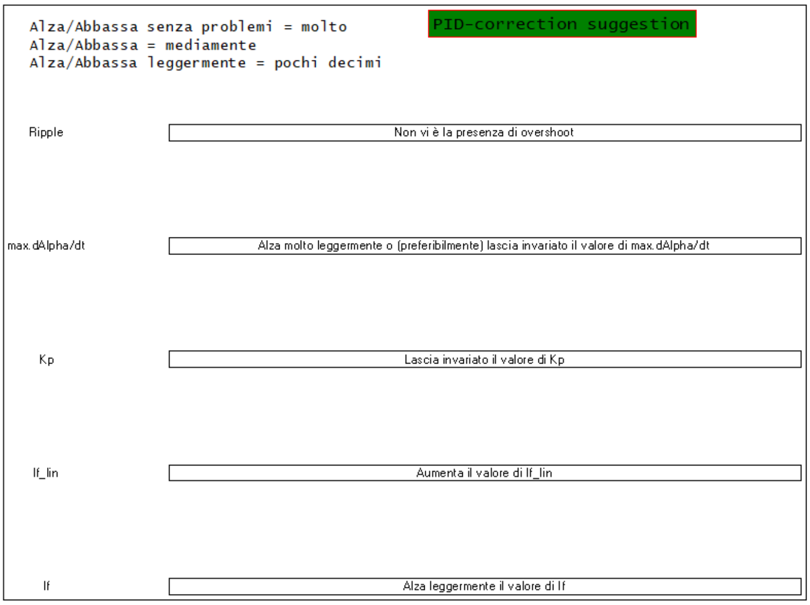
Generally, Kp is augmented if the alpha response is very poor overall. Kp is unvaried if , in the same situation, the original value of Kp is beyond a threshold set to 0.5 by simulation and a reduction of Kp is linked to the presence of overshoot in an environment where there is no problem in reaching high torque values when needed. In addition, max.dAlpha/dt is generally augmented if its original value is not too high: otherwise it would stay unchanged. As said, the original values are also considered by the software logic, in this way the modification will be more coherent. The tool will compute the regression enhancement provided by this improvement in the acceleration emanation due to previous parameters modification, this means that this software is capable to understand how the regression result will come better and if further actions are needed, its logic will be able to modify in a coherent manner also the other parameters. In a first analysis the values that parameterize the nonlinear behaviour of the proportional gain are not modified, since generally the test bench comes out with good parameters and however the effect of their change has not been so evident during the simulations. This enhances the code effect because concentrating on less parameters is a way to lower the number of variables under analysis and then the software complexity.

Concerning the integral part, if further modifications are needed, the software will take care of improving the parameters If and If_lin.

If_lin is a value between 0 and 1 and closer it is to 1, more linear will be the integral gain response. If this value is equal to 1, the gain characteristic will show a linear relationship over the entire range. Obviously, a bigger value of this integral function is something good because this leads to a linear action, but there are also nonlinearities that shall be taken into account: for this reason, the software has to carefully choose this value on the basis on considerations related to the fit of the regression line to the available data. This means that for low SEE and high r^2 it is possible to lower this value, because it has been proven by simulation that this will augment the intercept value. The best situation is when a lower intercept value is needed. In this case, the parameter modification (through an higher value) can seize the moment to enhance the overall regression since it will lower the intercept and it will provide a better fit because SEE coefficient will go down and r^2 coefficient will go up.

Lastly, the If parameter is modified to amplify the effect of the nonlinear function just described for integral gain, because this If is just a gain that multiply this function so it is used by the software if the effect of If_lin has be to played up.

This formula provides then every possible suggestion to understand what is going on with the regression analysis of this engine and it also tells about how the PI parameters shall be modified, everything is translated into sentences that can be read on the layout page that is created to accommodate this new solution.



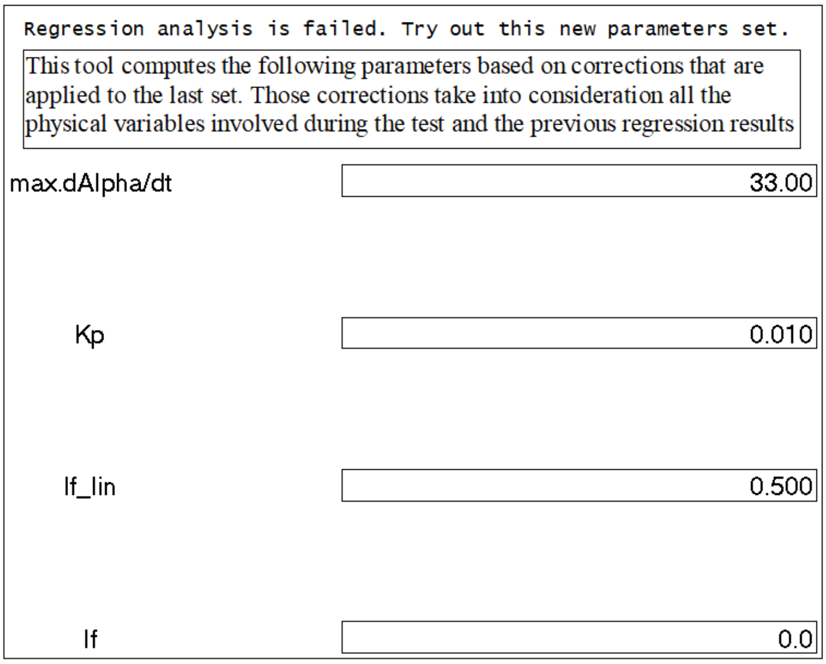
PID-correction suggestion

Alza/Abbassa senza problemi = molto
Alza/Abbassa = mediamente
Alza/Abbassa leggermente = pochi decimi

Ripple	Non vi è la presenza di overshoot
max.dAlpha/dt	Alza molto leggermente o (preferibilmente) lascia invariato il valore di max.dAlpha/dt
Kp	Lascia invariato il valore di Kp
If_lin	Aumenta il valore di If_lin
If	Alza leggermente il valore di If

Figure 2.20: PID parameters suggestions

The true modification is left to the last big formula of this thesis, where an algorithm is implemented to understand exactly how much a certain parameter shall increase or decrease or if it is better to left it unchanged. The logic behind this algorithm takes as input all the suggestions coming from the last formula and gives as output a new set of PI values that the user can consult on a specific layout page created for this purpose.



Regression analysis is failed. Try out this new parameters set.

This tool computes the following parameters based on corrections that are applied to the last set. Those corrections take into consideration all the physical variables involved during the test and the previous regression results

max.dAlpha/dt 33.00

Kp 0.010

If_lin 0.500

If 0.0

Figure 2.21: New PID parameters set

The choice of the correct modification is performed collecting every aspects of the test and the associated physical quantities, that are compared with the suggestion to understand how much it is needed to “listen” to that suggestion, in the sense that this formula will understand, starting from a suggestion like “increase Kp”, how much it has be to actually increased, giving the final value to the user that will set it on the test cell automation hardware.

3. Software Application

In this chapter, the software is employed to solve a real problem in the plant, since a test is having troubles in terms of regression analysis.

3.1 Test bench preparation

The engine comes to the plant and it is placed on a pallet, then everything is installed within the cell. It is important to have everything ready within the cell to welcome the engine, in the sense that before installing the motor it is important to prepare all instruments and sensors inside the cell and set them ready for a new engine.

To perform a WHTC, there is no choice between a normal brake or a dynamo, an electrical machine (provided to FPT by AVL) able to simulate every working condition of the vehicle with the relative frictions: a dynamo has to be chosen because WHTC is a particular cycle that tests every situation an engine can live, this means that it will also test conditions like driving on a road that goes downhill, for example, where even if the driver doesn't press heavily the acceleration pedal, the truck can easily gain speed and a dynamo is able to simulate this. For this reason, the automation will mostly impose the torque to the engine and the relative speed to the dynamo.

These two entities are linked using a Cardan Joint, a particular joint that is used to transmit rotary motion. In this context, a variant of the classic Cardan joint is used: it is called Double Cardan joint with damper. The simple Cardan joint has a drawback: even if first shaft rotates at a constant speed, the output shaft rotates at a variable speed, this will surely cause vibration at high rate, that is not good in motor testing environment, for this reason a Double Cardan joint solution is addressed.

This double configuration uses the two joints connected to an intermediate shaft, with the second element phased in relation to the first one to cancel the angular velocity shifting. In this configuration, the angular velocity of the driven shaft will be equal to the one of the driving shaft, provided some conditions about the positioning with respect to intermediate shaft.

When all entities are linked in the test room, every sensor is connected, together with the engine ECU and the instruments that are needed for law check purposes, like the after-treatment system that is fundamental to compute the right quantities that are going to be compared to law requirements. In this moment, every human operator has to leave the room for security purposes and after that the engine is started from the EMCON automation hardware.

Now some concepts expressed in the introduction to the tests will be reiterated and enhanced in order to introduce the software application on a real engine problem. The operator, in order to perform a correct WHTC, has now to perform two operations on the engine: the engine mapping and the preconditioning operation.

The engine mapping is important because as said during the introduction about those test cycles, the regulator provides some conditions in terms of normalized values, for this purpose a first action is represented by the engine mapping that is a particular cycle to be performed by a hot engine (not as soon as it is started). It is a full load cycle from minimum (idle speed) to maximum governed speed, in this way it is possible to obtain the characteristic velocities of the engine that are useful for denormalization purposes. This is the only thing to do to test several engines on the same test bench, without this operation everything in the cell should be designed for a specific motor and this would be too costly and time consuming considering the number of engines that have to be tested. The standardization principle is again a key point inside this plant.

The second operation is called preconditioning. The law regulator asks to perform the official WHTC in a standard situation for everyone, someone could think to save on some pollutant by starting the test on certain advantageous situation: for this reason, the law requires a preconditioning operation that basically is at least one phase of a WHTC so that the engine will test the real WHTC after that and not in another moment.

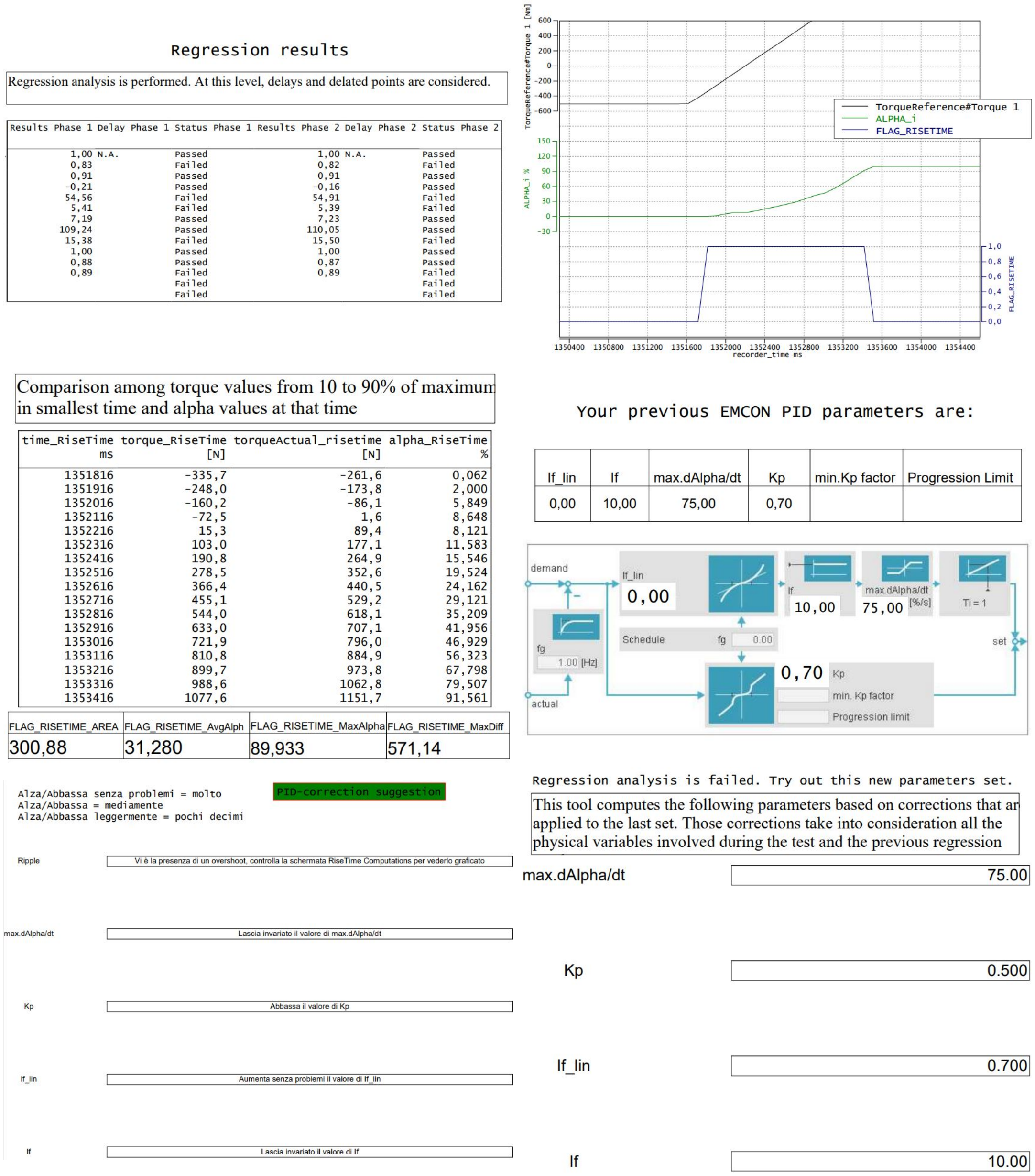
WHTC is a cold-hot cycle, this means that it starts with a cold engine status. An engine is defined as “cold” by the regulator when the temperature of water, oil and air is between 20 and 30 Celsius degrees. When this condition is respected, the cold phase can start, lasting 1800 seconds. When the cold phase ends, there is a small period called “hot soak” that lasts few minutes: obviously, this durations is again determined by the law requirements to enhance standardization for this test. Lastly, the hot phase is performed, again for 1800 seconds. The emissions are computed both during cold and hot phase and at the end they are mixed together using a weighted average, because it is very likely that the engine will be cold only the first minutes of a trip and then it will easily become hot, for this reason what happens during hot phase is more important and it has a heavier weight in an average. To that results, the DF and the regeneration factors are added to take into account the fact that this engine shall operate for several years and then it is not fair to compute the emissions testing a new one that is used only for that test, some factors will be added to take into account the effect of the years and of the kilometres on every part of the engine.

3.2 Application to real engine test

Once the tool has been developed, the automation group has received a call from a test bench because an engine needed to execute a WHTC. All the procedures explained in last section have been followed to the letter. The engine starts the test and in post processing it is evident that the regression analysis is failed. The automation group has now to solve this problem for the operator. Looking at the engine, its characteristics are:

- F4H; NEF engine for on-road applications;
- Certification: EURO VI;
- Turbocharged;
- Engine Displacement: 6.7 l;
- Maximum Power: 210 kW at 2500 rpm;
- Applications: Goods or people transport.

From Concerto, the tool is launched and the situation is studied.



Regression results

Regression analysis is performed. At this level, delays and delated points are considered.

Results Phase 1	Delay Phase 1	Status Phase 1	Results Phase 2	Delay Phase 2	Status Phase 2
1,00	N.A.	Passed	1,00	N.A.	Passed
0,83		Failed	0,82		Failed
0,91		Passed	0,91		Passed
-0,21		Passed	-0,16		Passed
54,56		Failed	54,91		Failed
5,41		Failed	5,39		Failed
7,19		Passed	7,23		Passed
109,24		Passed	110,05		Passed
15,38		Failed	15,50		Failed
1,00		Passed	1,00		Passed
0,88		Passed	0,87		Passed
0,89		Failed	0,89		Failed
		Failed			Failed
		Failed			Failed

Comparison among torque values from 10 to 90% of maximum in smallest time and alpha values at that time

time_RiseTime ms	torque_RiseTime [N]	torqueActual_risetime [N]	alpha_RiseTime %
1351816	-335,7	-261,6	0,062
1351916	-248,0	-173,8	2,000
1352016	-160,2	-86,1	5,849
1352116	-72,5	1,6	8,648
1352216	15,3	89,4	8,121
1352316	103,0	177,1	11,583
1352416	190,8	264,9	15,546
1352516	278,5	352,6	19,524
1352616	366,4	440,5	24,162
1352716	455,1	529,2	29,121
1352816	544,0	618,1	35,209
1352916	633,0	707,1	41,956
1353016	721,9	796,0	46,929
1353116	810,8	884,9	56,323
1353216	899,7	973,8	67,798
1353316	988,6	1062,8	79,507
1353416	1077,6	1151,7	91,561

FLAG_RISETIME_AREA	FLAG_RISETIME_AvgAlph	FLAG_RISETIME_MaxAlpha	FLAG_RISETIME_MaxDiff
300,88	31,280	89,933	571,14

Your previous EMCON PID parameters are:

If_lin	If	max.dAlpha/dt	Kp	min.Kp factor	Progression Limit
0,00	10,00	75,00	0,70		

```
Alza/Abbassa senza problemi = molto
Alza/Abbassa = mediamente
Alza/Abbassa leggermente = pochi decimi
```

PID-correction suggestion

Ripple	Vi è la presenza di un overshoot, controlla la schermata RiseTime Computations per vederlo graficato
max.dAlpha/dt	Lascia invariato il valore di max.dAlpha/dt
Kp	Abbassa il valore di Kp
If_lin	Aumenta senza problemi il valore di If_lin
If	Lascia invariato il valore di If

Regression analysis is failed. Try out this new parameters set.

This tool computes the following parameters based on corrections that ar applied to the last set. Those corrections take into consideration all the physical variables involved during the test and the previous regression

max.dAlpha/dt	75.00
Kp	0.500
If_lin	0.700
If	10.00

Figure 3.1: The selected test is studied through the PID Analysis Tool software

The regression analysis shows many problems. The first one is evident by looking at the acceleration response because there is an overshoot that is noticed by the software, for this reason it is important to act on the alpha response. Moreover, the regression results are very poor and it is very likely that working only on a better alpha supply will not solve the problem: for this reason it is important to also work on the integral contribution of the controller, also this aspects is correctly highlight by the software. It is noticeable that the software perfectly acted against the problems and it also provided a new set of PI parameters. These values are tested and the regression improvements are remarkable: the test can go on and the problem has been solved.

4. Conclusions

This chapter will conclude this thesis work. The objectives have been achieved, since a software able to correct the PID parameters of the test bench controller on the basis of a given engine under test has been developed. In addition, a powerful layout is able to show both the results and many other interesting considerations about the test under analysis. However, this work can also be enhanced. This means that, before discussing the conclusions of this work, it is needed to spend some time dealing with interesting future works that can be related to that software.

4.1 Future work

The developed software is a very good result from a practical point of view: it will be very useful during the plant activities and, mainly, it will save some time for the Automation Group. However, there are some points that could be enhanced:

- The simulation can be enhanced. A very long and complicated work could be made in terms of Simulink schemes, in the sense that the PID test bench simulation could be improved and added as sub-system is a bigger scheme where also a simulation of the engine is present as second sub-system, where, through some input blocks, the operator could select the right engine under test. This could be the main topic of a future next thesis in the plant. The results that this work can get are again in terms of time to save, because a single phase of WHTC, for example, lasts 30 minutes. This means that the operator may lose half an hour to end up with a failed regression and also with an engine that was running for nothing, that is never a good idea. The plant has already a simulator for the engine, that is the one used during this thesis, but it is a very sophisticated system that is very costly and for this reason it is impossible to have too much of them, moreover it also takes 30 minutes to perform the simulation. A Simulink scheme, even if a bit less precise, will give an idea in few minutes of what will be going on.

- The automotive environment lives as a very complicated moment the 21th century. Even if it is true that the electronics made possible to reach very good results in terms of pollutant emissions, the standards are continuously evolving and nobody knows what will be required in the future to face the environmental problems of the world.
  For this reason, it is very likely that this software will need to become more adaptive, in the sense that engine and law requirements will become different in a time that is going to be always shorter. To deal with this problem, in the future the software could be based on an artificial intelligence (AI) algorithm. A definition of AI could be based on four categories [10]:

  – "Systems that think like humans";

  – "Systems that act like humans";

  – "Systems that think rationally";

  – "Systems that act rationally".

  However, a tension exists between human-centered approaches and rationality-centered approaches, because to really emulate a human also the mistakes should be taken into account since nobody is infallible. For this reason, the human-centered approach must be an empirical science with hypothesis and confirmation while a rationality-centered approach must involve mathematics and engineering. The rationality-centered approaches are then preferred.
  At this level, it has not been done to ensure robustness, since it has been preferred to base the software on precise corrections that were studied to cover every possible case, but in the future, it is very likely that the AI will become more robust even in control design and it will be a key through which implement a more adaptive algorithm for the future.

4.2 Conclusions

This thesis is entirely dedicated to a software. All the steps of a software creation have been correctly followed. Starting from a primary study of the editor and of the engineering field under test (automotive industry and controller techniques), the software has been developed in layers called formulae, to enhance the reuse, the development and the edit. After that, this tool has been validated through simulations and real tests, from which a real case has also been taken and discussed during the thesis work. A powerful tool has been developed for the plant that will save time and then money, by performing on its own many considerations on the regression analysis imposed by the law standards for an engine test. Lastly, it is important to remember that this thesis could also be a starting point for many other automation projects discussed in last section.

List of Figures

List of Tables

Bibliography

[1] AVL, "AVL Concerto 2015." https://www.avl.com/documents/10138/885893/CONCERTO+2015+Release+Notes.pdf.

[2] AVL, "AVL Concerto 5, Experience the harmony." https://www.avl.com/documents/10138/3932186/AVL+CONCERTO+5%E2%84%A2+-+Solution+Brochure.

[3] MathWorks, "Simulink." https://it.mathworks.com/help/simulink/.

[4] FPT Industrial, "Engine Power, Emissions and CO_2 BenchTesting," 2019.

[5] G. Franklin, D. Powell, and M. Workman, *Digital Control of Dynamic Systems.* 1997.

[6] L. Keviczky, R. Bars, J. Hetthessy, and C. Banyasz, *Control Engineering.* 2017.

[7] A. Sykes, "An introduction to regression analysis," 1993.

[8] Economic Commission for Europe of the United Nations (UN/ECE), "Regulation No 49," 2018.

[9] AVL, "Emcon 400/401," 2015.

[10] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach.* 1995.