AIR FUEL RATIO CLOSED LOOP CONTROL

Applied Automatic Control

Advanced Automotive Engineering

University of Bologna

A.A. 2020-2021

Lorenzo Ferrari, Matteo Fiorani, Luca Freddi, Tommaso Trunfio

June 4, 2021

Abstract

Increased air pollution is one of the hardest challenges facing States in recent decades and one of its main causes are the emissions coming from the automotive internal combustion engines. International Governments have taken several measures to regulate the number of emissions produced by this category of vehicles. This has involved new techniques and devices that could limit and reduce exhaust pollutants and, one of the most effective way to achieve this purpose, is to design a feedback control system with the goal of keeping a stoichiometric air-fuel ratio or at least in the range of it (slightly lean or slightly rich).

In this study, an aspirated spark ignition engine (1275 cc British Leyland) is implemented in a MATLAB program in order to design a feedback control by means of a MIMO closed loop control to ensure the stability and anti-disturbance of the system. The dynamic model is evaluated through the mean value engine model (MVEM), and it consists of three engine variables (states): fuel film mass flow in the intake runners, intake manifold pressure and the AFR dynamics. The addressed control variables are the throttle valve angle and the injected fuel mass flow through the injector. To implement the control algorithm, some reference conditions are selected in the first place and, a subsequent linearization of the plant around them is carried out. Then, the LQR (Linear Quadratic Regulator) command is utilized to put in place the optimal control strategy, together with the design of the optimal observer through the *kalman* command. Eventually, the engine rotational speed represents the external disturbance affecting the plant.

Chapter 1

Introduction

1.1 Motivations

Internal Combustion Engines are one of the major causes of air pollution and, during the last two decades, several solutions have been designed to considerably decrease the amount of them, such as the three-way catalytic converter (TWC) for gasoline engines. The three-way catalyst, to work properly, needs an air-to-fuel ratio (AFR) as close as possible to its stoichiometric value, therefore, the control of the AFR represents a key point for engine management. This parameter impacts the quality of the combustion process and engine performance related to fuel consumption and emissions. As a result, closed-loop control of air-to-fuel ratio is of paramount importance.

1.2 Contributions

The aim of this study is to design a closed-loop control scheme to properly regulate the air-to-fuel ratio to exploit the maximum conversion efficiency of the TWC. The first part of this work is dedicated to the description of the dynamic model of a spark ignition engine through the mean value engine model (MVEM), enabling the dynamic estimation of the main engine variables (fuel film mass flow rate in the intake runners, intake manifold pressure and lambda dynamics) with respect to variations of the control inputs (throttle plate angle and the injected fuel). Then, the second part of the work focuses on the corresponding MATLAB and Simulink simulation, in which, the closed loop scheme is developed by implementing and tuning a MIMO controller, designed to satisfy the objectives of the study.

1.3 State of art and literature comparison

The study has been elaborated thanks to the contribution coming from several works. The idea of describing the engine variables through dynamic models is handled by papers [1],[2] where the Mean Value Engine Model has been utilized as main reference, since it has a good predictive property without being too complex [1]. One of its most important elements is the manifold pressure state equation, because it can estimate the air mass flow through the throttle during throttle opening and closing. The air mass flow significantly affects the air/fuel ratio and consequently the performances and the emissions of a SI engine [2], [3]. References [4] and [5] tackle the problem from another point of view, in the specific, the manipulated variable is the fuel mass flow into the combustion chamber, the controlled variable is lambda, and the throttle plate angle is considered as a disturbance.

Looking into the literature, several strategies of AFR control are addressed. A first group deals with linear tools. Guzzella et al. [6] develop a feedback linearization and then, a robust controller is designed. The main drawbacks are a lack of robustness according to noise measurements and, besides, the time delay introduced by the lambda sensor is not considered. A second family exploits non-linear tools. In Li and Yurkovich's work [7], adaptive sliding mode control is used and, also in these approaches, the control law design does not take into account the time delay since, the strength of the method is supposed to compensate this drawback. Their objective is to ensure a fast AFR regulation around 1 even during fast transient phases. Another proposed solution by Powell et al. [8] uses an accurate estimation, which relies on the linear observer theory, to control the engine with a specific focus on observers, based on exhaust measurements. This method presents a delay between the plant and the sensor measurements and, the primary disturbance is represented by the throttle.

1.4 Organization of the manuscript

Chapter 2 describes the setup of the plant dynamic model of the internal combustion engine. The main formulas are described and briefly discussed and, moreover, the reference conditions are defined together with the reference control variable and disturbance. Subsequently, the reachability and observability studies are performed, followed by a scheme of the control system architecture with its implementation through a MATLAB code. The theory behind the control strategy is briefly recalled and adapted to the purpose.

Chapter 3 includes the Simulink block scheme and the description of all its elements, the closed loop, the integral action, and the observer block. Then, the analysis of the main script file is carried out, in which it is shown where the control parameters can be changed for user purposes. Finally, the simulation results are plotted, commented and, besides, the tuning logic path is also introduced.

Chapter 4, the last one, summarizes the highlights of the results and opens a discussion about additional and future studies, which can further improve the AFR management in gasoline engines.

1.5 List of the symbols

The following symbols are used in this project and there are shown in the Appendix:

t	Time [s]
α	Throttle plate angle [degrees]
α_0	Closed throttle plate angle [degrees]
$\dot{m}_{cyl}$	Air mass in the cylinder [kg/s]

$\dot{m}_f$	Fuel film mass flow in the combustion chamber [kg/s]
$\dot{m}_{fi}$	Injected fuel mass flow [kg/s]
$\dot{m}_{fv}$	Fuel mass vaporization rate [kg/s]
$\ddot{m}_{ff}$	Fuel film mass flow rate [kg/s^2]
p_{amb}	Ambient pressure [Pa]
p	Intake manifold pressure [Pa]
T_{amb}	Ambient temperature [K°]
T_{man}	Intake manifold temperature [K°]
V_d	Engine displacement [m^3]
V	Intake manifold and port passage volume [m^3]
n	Engine speed [rpm]
R	Gas global constant [J/(kg·K)]
D	Throttle bore diameter [m]
η_v	Volumetric efficiency [-]
λ_{cyl}	Relative air/fuel ratio inside cylinder [-]
λ_s	Stochiometric air/fuel ratio [-]
τ_f	Vaporization time constant [s]
χ	Film fuel fraction [-]
C_t	Flow coefficient of throttle body throat
k	Ratio of specific heats (C_p/C_v; 1.4 for the air)

Chapter 2

Main body

2.1 Model and problem formulation

In order to pursue the development of a suitable and well-tuned controller capable of steering the air fuel ratio (AFR) to the desired value, the first element that has to be defined is the set of equations that describes the system.
To fully understand and track all the parameters that influence the engine functioning, the typical MVEM (Mean Value Engine Model) approach has been considered. Thanks to this model the cycle's evolution can be mapped with good accuracy (up to 5000 rpm) despite some little approximations. Above all the assumptions made, certainly the most important refers to the combustion process, which is considered only as a time delay without accounting separately every single reaction.

After this brief clarification, the Internal Combustion System can be described considering 3 main contributes:

- Intake Manifold Dynamics
- Fuel Film Mass Flow Dynamics
- Lambda Dynamics

2.1.1 Intake manifold dynamics

In order to choose the most complete model for control purposes, two main MVEMs has been considered: the isothermal and the adiabatic.
The isothermal model is obtained by considering mass conservation and by knowing both the temperature of the manifold time by time (equal to the ambient one) and EGR temperatures.
The adiabatic indeed, through the mass and energy conservation, describes the dynamics of the intake manifold also considering the temperature variation and, it is defined by 2 coupled nonlinear differential equations.

For project purposes, the isothermal model has been adopted assuming the intake temperature known and equal to $T_{man} = 297° K$.
The intake manifold pressure dynamics is described by [3]:

$$\dot{p}(t) = -\frac{n(t)V_d\eta_v p(t)}{120V} + \frac{RT_{man}(t)\dot{m}(t)}{V} \tag{1}$$

In the equation (1) the term $\dot{m}(t)$ describes the dynamics of the air mass flow through the intake manifold and it is a function of the throttle plate angle and the pressure ratio:

$$\dot{m}(t) = C_t \frac{\pi}{4} D^2 \frac{p_{amb}}{\sqrt{RT_{amb}}} \beta_1(\alpha)\beta_2(p(t)) \tag{2}$$

where:

$$\begin{aligned}\beta_1(\alpha) = \left(1 - \frac{\cos(\alpha)}{\cos(\alpha_0)}\right) + \frac{2}{\pi}\left(\frac{a}{\cos(\alpha)}\sqrt{(\cos^2(\alpha) - a^2\cos^2(\alpha_0)}\right) \\ + \frac{2}{\pi}\left(\frac{\cos(\alpha)}{\cos(\alpha_0)} \arcsin\left(a\frac{cos(\alpha_0)}{cos(\alpha)}\right) - a\sqrt{1-a^2} - \arcsin(a)\right)\end{aligned} \tag{3}$$

and

$$\beta_2(p(t)) = \begin{cases} \sqrt{\frac{2k}{k-1}\left(\left(\frac{p(t)}{p_{amb}}\right)^{\frac{2}{k}} - \left(\frac{p(t)}{p_{amb}}\right)^{\frac{k+1}{k}}\right)} & , if \left(\frac{p(t)}{p_{amb}}\right) \geq \left(\frac{2}{k+1}\right)^{\frac{k}{k-1}} \\ \sqrt{k}\left(\frac{2}{k+1}\right)^{\frac{k+1}{2(k-1)}} & , otherwise \end{cases} \tag{4}$$

The term k appearing in (4) for the air is set equal to 1.4.
Finally, the volumetric efficiency is defined as:

$$\eta_v = \eta_{v_{n0}} + \eta_{v_{n1}} N + \eta_{v_{n2}} N^2 + \eta_{v_{p1}} p \tag{5}$$

Further details regarding the definition of the aforementioned efficiency are given in the Appendix section.

2.1.2 Fuel film mass flow dynamics

As known, for a gasoline engine, the fuel injected in the runners is not completely mixed with the air entering the cylinder. Indeed, part of it due to the wall wetting phenomenon, settles on the runners' walls as a liquid fuel film, which vaporizes with a certain time constant mixing with the coming fresh air.
It's evident that considering the fuel entering the cylinder equal to the one injected would lead to an error in the estimated lambda inside the combustion chamber, so it's necessary to account the phenomenon by the means of the Aquino model.

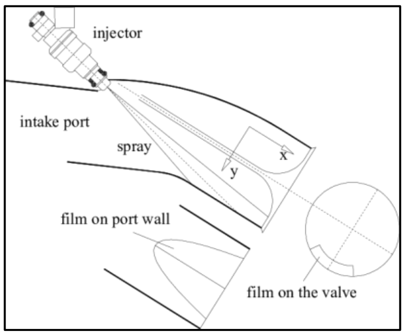


Figure 1 Fuel deposition in the intake runner

The Aquino model consists of three equations that consider the above-mentioned facts:

$$\ddot{m}_{ff} = \frac{1}{\tau_f}(-\dot{m}_{ff} + \chi \dot{m}_{fi})$$
$$\dot{m}_{fv} = (1 - \chi)\dot{m}_{fi} \tag{7}$$
$$\dot{m}_f = \dot{m}_{fv} + \dot{m}_{ff}$$

The first equation refers to the time evolution of the fuel film mass flow rate and, it depends on the amount of fuel which has been injected. Varying the injected fuel, the film mass changes following a certain dynamics having a time constant τ_f. The fuel fraction χ and the time constant τ_f are usually mapped for each kind of engine and they can be estimated according to the engine rotational speed. In eq. (8) is reported the definition of the above-mentioned parameters [4].

$$\begin{gathered} \tau_f = \sigma_3 N^{-\sigma_4} \\ \chi = \sigma_5 N + \sigma_6 \end{gathered} \tag{8}$$

2.1.3 Lambda dynamics

The relation that defines the λ for the cylinder is shown in eq. (9), in which the AFR depends on the air mass pumped in the cylinder and the fuel mass together with the stoichiometric AFR, equal to 14.67.

$$\lambda_{cyl} = \frac{m_{cyl}}{\lambda_s m_f} \tag{9}$$

The dynamics of lambda in the cylinder is obtained through the differentiation of eq. (9), considering that $\dot{m}_f = \frac{N}{30} m_f$ for a four-stroke engine. So, the derived equation is:

$$\dot{\lambda}_{cyl} = \frac{N}{30} \frac{1}{\lambda_s} \left(\frac{\dot{m}_{cyl}}{\dot{m}_f} - \dot{\lambda}_{cyl} \right) \tag{10}$$

Nevertheless, the actual value of lambda does not refer to the ongoing combustion cycle but, it exists a certain time delay in the measurement on the back of the exhaust gases transport through the exhaust manifold where the lambda sensor is placed. This delay introduces a further model for the lambda sensor dynamics, which has not been accounted for this project [4].

2.1.5 Plant definition

Once the fundamental equations have been described, the plant can be then defined. The plant vector x (eq. 11) has a dimension equal to 3 in which, the first equation regards the fuel film mass flow rate, the second one is referring to the lambda dynamics and, the last one defines the intake manifold time evolution. As for the control vector u, the throttle plate angle α and the injected fuel mass flow rate $\dot{m}_{fi}$ have been selected. Eventually, to insert a realistic disturbance d to the plant, the engine rotational speed N has been chosen, assuming that the engine is run on a test bench in which the speed is set by the brake dynamometer.

$$x = \begin{bmatrix} m_{ff} \\ \lambda_{cyl} \\ p \end{bmatrix} \qquad u = \begin{bmatrix} \dot{m}_{fi} \\ \alpha \end{bmatrix} \qquad d = [N] \tag{11}$$

Eventually the plant results in:

$$\dot{x} = \begin{bmatrix} \dot{m}_{ff} \\ \dot{\lambda}_{cyl} \\ \dot{p} \end{bmatrix} = \begin{bmatrix} \frac{1}{\tau_f}\left(-\dot{m}_{ff} + \chi \dot{m}_{fi}\right) \\ \frac{N}{30}\left(\frac{1}{\lambda_s}\frac{\eta_v\, N\, V_d\, p}{120\, R\, T_{man}\left[(1-\chi)\dot{m}_{fi} + \dot{m}_{ff}\right]} - \lambda_{cyl}\right) \\ -\frac{N\, V_d\, \eta_v\, p}{120\, V} + \frac{R\, T_{man}}{V} C_t \frac{\pi}{4} D^2 \frac{p_{amb}}{\sqrt{R\, T_{amb}}}\, \beta_1(\alpha)\, \beta_2(p) \end{bmatrix} \tag{12}$$

The following initial conditions have been considered for simulation purposes.

$$x_0(t_o = 0) = \begin{bmatrix} 3 * 10^{-5}\frac{kg}{s} \\ 0.8 \\ 0.5 * p_{atm} \end{bmatrix} \tag{13}$$

2.2 Model analysis

2.2.1 Linearisation

The plant linearisation is the first step in the design of a general control system but, before going through it, proper reference conditions need to be defined. Specifically, in this project, steady-state conditions have been considered as main reference and, moreover, the reference values of lambda and the intake manifold pressure (reported in Table 1) have been selected according to [3]. Going through some calculations, the corresponding values of the control variables $\dot{m}_{fi}$ and α can be then defined once the reference disturbance N_{ref} has been also set. As for α_{ref}, a MATLAB command has been exploited to solve numerically the equation regarding the $\beta_1(\alpha)$ definition.

Reference conditions	$\lambda_{ref} = 1$
	$p_{ref} = 0.37\ bar$
	$\dot{x} = \begin{bmatrix} \dot{m_{ff}} \\ \dot{\lambda_{cyl}} \\ \dot{p} \end{bmatrix} = 0$
Reference control variables	$\dot{m_{f_{i_{ref}}}} = \frac{1}{\lambda_s} \frac{N_{ref} V_d \eta_{v_{ref}} p_{ref}}{R T_{man} \lambda_{ref} 120}$
	$\beta_{1_{ref}}(\alpha) = \frac{N_{ref} V_d \eta_{v_{ref}} p_{ref}}{R T_{man} \lambda_{ref} 120} \frac{4\sqrt{R T_{amb}}}{C_t \pi D^2 p_{amb} \beta_{2_{ref}}}$
Reference disturbance	$N_{ref} = 870\ rpm$

Table 1 Control system reference conditions and corresponding control variables

Starting from the reference conditions, the linearisation procedure can be carried out and more details are given in the Appendix. The outcome is the definition of the matrix A, which considers the partial derivatives of the plant with respect to the plant variables, the matrix B$_1$, where the partial differentiation of the plant with respect to the control variables is performed, and, eventually, the matrix B$_2$ which takes into account the partial derivative of the state with regard to the disturbance. So, a new linear plant in terms of $\tilde{x}$ results from the linearisation as shown in eq. 14.

$$\dot{\tilde{x}} = x - x_{ref} = A\tilde{x} + B_1\tilde{u} + B_2\tilde{d}$$
$$\tilde{u} = u - u_{ref} \tag{14}$$
$$\tilde{d} = d - d_{ref}$$

2.2.2 Plant eigenvalues

Once all the partial derivatives have been performed, the A matrix, shown in Table 2, is then defined.

-48.75	0.00	0.00
-220008.58	-29.00	0.00
0.00	0.00	-7.52

Table 2 A matrix

The corresponding eigenvalues, reported in Table 3, are calculated by means of the MATLAB command eig(A).

-29.00
-48.75
-7.52

Table 3 A matrix eigenvalues

Actually, there would not be the need to calculate them because, looking better at the A matrix, it can be noticed that it is a lower triangular one so, its eigenvalues are located along the diagonal as confirmed by the MATLAB calculation. They are all negative ones, meaning that the plant is self-stable according to the automatic control theory and, this behaviour, has been supported by the Simulink open loop simulation as shown in Figure 2.

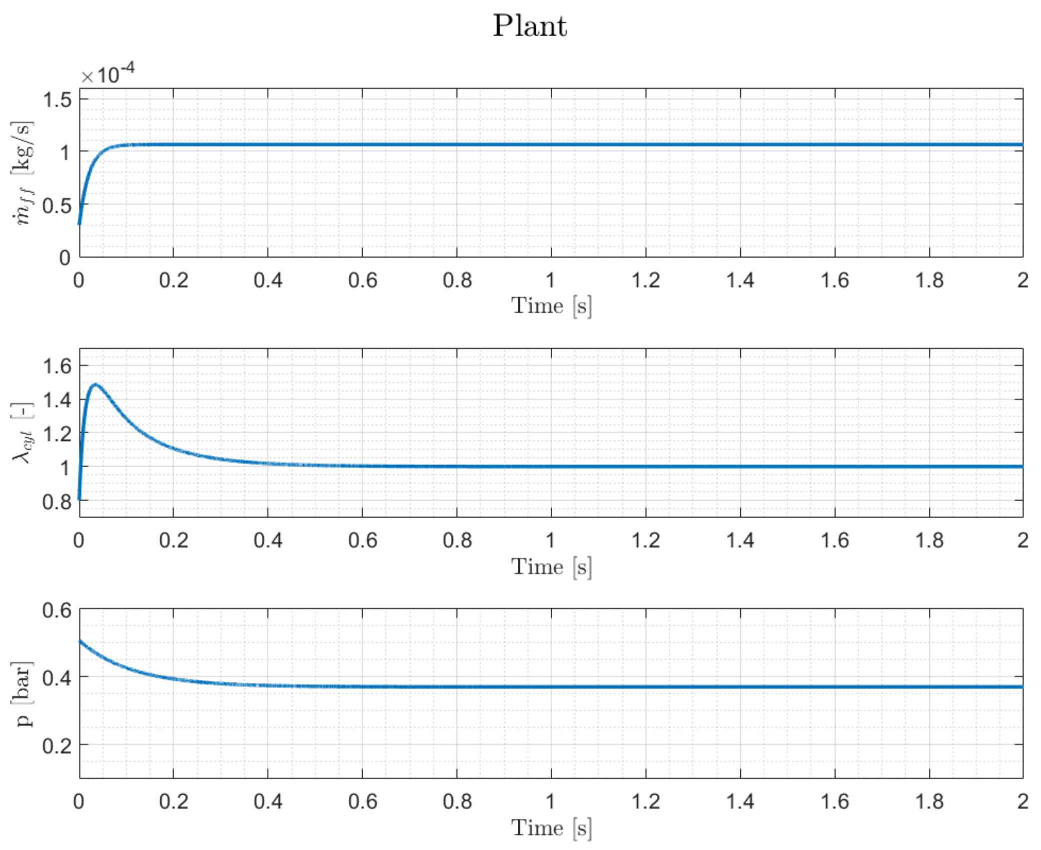


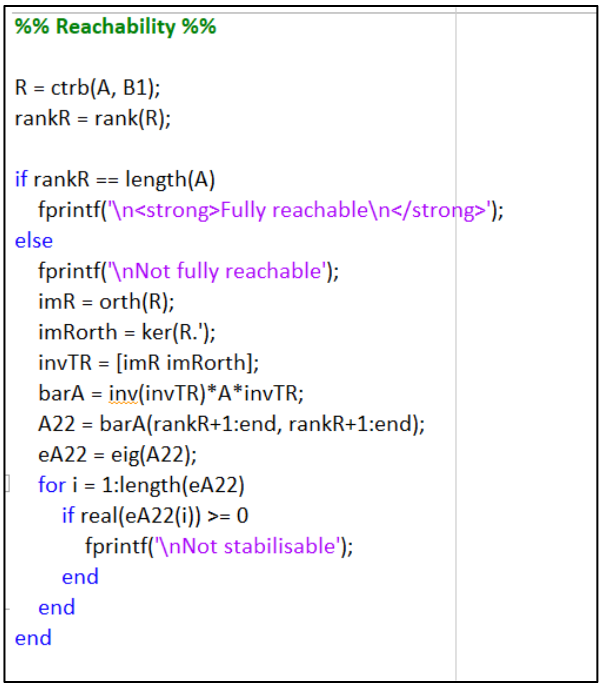
Figure 2 Open loop simulation

2.2.3 Reachability analysis

To study the reachability of the control system, it is necessary to the build the reachability matrix R and check if its rank is equal to the number of the plant states, 3 in this case. Making reference to the control theory, the R matrix is built in the following way:

$$R = [B_1 \quad AB_1 \dots A^{n-1}B_1] = [B_1 \quad AB_1 \quad A^2B_1] \tag{15}$$

In the MATLAB environment, the reachability matrix is generated through the command *ctrb*, which returns the R matrix once the A and B_1 have been provided (Figure 3).



```
%% Reachability %%

R = ctrb(A, B1);
rankR = rank(R);

if rankR == length(A)
   fprintf('\n<strong>Fully reachable\n</strong>');
else
   fprintf('\nNot fully reachable');
   imR = orth(R);
   imRorth = ker(R.');
   invTR = [imR imRorth];
   barA = inv(invTR)*A*invTR;
   A22 = barA(rankR+1:end, rankR+1:end);
   eA22 = eig(A22);
   for i = 1:length(eA22)
      if real(eA22(i)) >= 0
         fprintf('\nNot stabilisable');
      end
   end
end
```

Figure 3 MATLAB script to build the reachability matrix and check its rank. If the rank is not maximum, the system stabilisation is addressed, and the script warns if the plant is not stabilisable.

The resulting R matrix is shown in Table 4 and, its rank is equal to 3 meaning that the system, with the chosen control variables $\dot{m}_{fi}$ and α, is fully reachable.

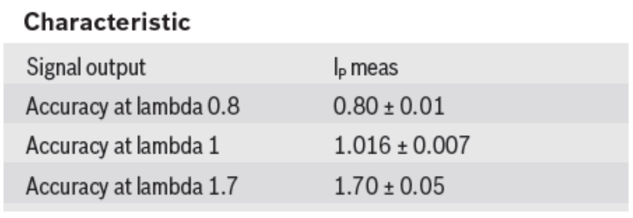
39.35	0.00	-1918.14	0.00	93508.53	0.00
-42435.25	0.00	-7426005.92	5995.05	637361326.05	-218919.53
0.00	7648861.32	0.00	-57494144.13	0.00	432165844.03

Table 4 Reachability matrix

2.2.4 Observability analysis

Once the reachability study has been performed, the final step in the plant characterization is the observability analysis, whose outcome tells if it is possible to estimate the plant with the chosen set of sensors. In this project, 2 sensors have been selected: the lambda sensor and the intake manifold pressure sensor. One remark regarding the lambda sensor is that its measurement, actually, occurs with a certain time delay due to the exhaust gases transport through the exhaust manifold but, in the project, this fact is not considered, leading to an instantaneous lambda measurement.

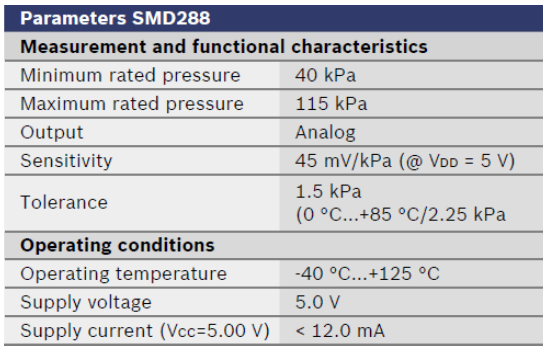
As known, every sensor has its own accuracy and to model it, a proper noise power has been selected. Having had a look at the main datasheet parameters of the Bosch lambda sensor and the Bosch barometric pressure sensor [9-10], both reported in Figure 4, the corresponding noise power of the Band-Limited White Noise Simulink block is then chosen to match, as much as possible, the sensor tolerances as shown in Figure 5-6.



Characteristic

Signal output	I_p meas
Accuracy at lambda 0.8	0.80 ± 0.01
Accuracy at lambda 1	1.016 ± 0.007
Accuracy at lambda 1.7	1.70 ± 0.05

A)



Parameters SMD288	
Measurement and functional characteristics	
Minimum rated pressure	40 kPa
Maximum rated pressure	115 kPa
Output	Analog
Sensitivity	45 mV/kPa (@ V_{DD} = 5 V)
Tolerance	1.5 kPa (0 °C...+85 °C/2.25 kPa
Operating conditions	
Operating temperature	-40 °C...+125 °C
Supply voltage	5.0 V
Supply current (V_{CC}=5.00 V)	< 12.0 mA

B)

Figure 4 Sensor suite main parameters: A) Bosch Lambda Sensor LSU 4.9, B) Bosch barometric pressure sensor SMD288

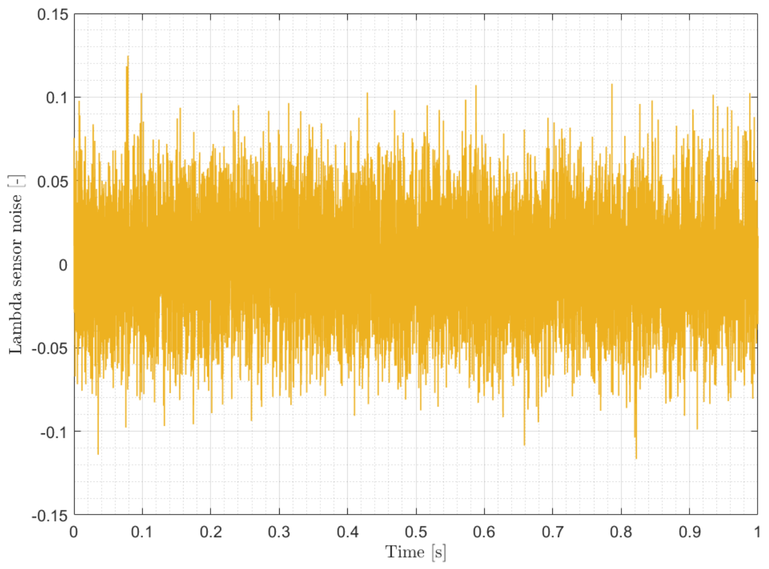


Figure 5 Lambda sensor noise

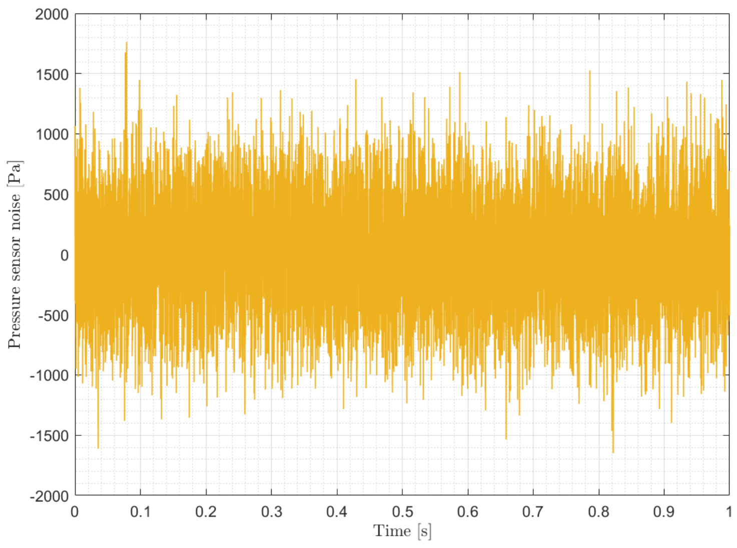


Figure 6 Pressure sensor noise

From the automatic control theory, the linearized sensor output vector $\tilde{y}$ must be defined to proceed with the observability analysis, as shown in Eq. 16.

$$\tilde{y} = y - y_r + v$$

$$y_r = \begin{bmatrix} \lambda_{cyl_{ref}} \\ p_{ref} \end{bmatrix} = \begin{bmatrix} 1 \\ 0.37\ bar \end{bmatrix} \tag{16}$$

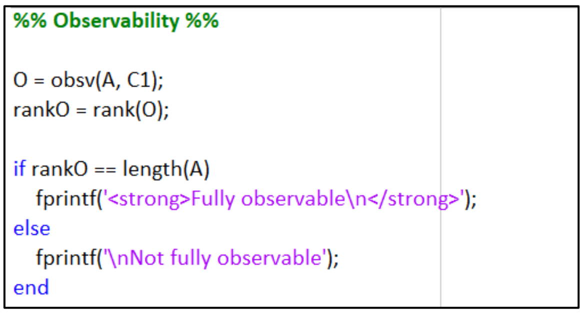
$$v = \begin{bmatrix} lambda\ sensor\ noise \\ pressure\ sensor\ noise \end{bmatrix}$$

In this specific case, $y - y_r$ can be written as:

$$y - y_r = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tilde{x} = C_1 \tilde{x} \tag{17}$$

The definition of the matrix C_1 is crucial because it let us build the O matrix that is, the observability matrix (Eq. 18). MATLAB has a special command to setup the O matrix, the *obsv* one which returns the observability matrix once the A and C_1 have been provided as displayed in Figure 7. Moreover, it has been also shown the complete MATLAB script to look into the observability problem.

$$O = \begin{bmatrix} C_1 \\ C_1 A^2 \\ \dots \\ C_1 A^{n-1} \end{bmatrix} = \begin{bmatrix} C_1 \\ C_1 A \\ C_1 A^2 \end{bmatrix} \tag{18}$$



```
%% Observability %%

O = obsv(A, C1);
rankO = rank(O);

if rankO == length(A)
   fprintf('<strong>Fully observable\n</strong>');
else
   fprintf('\nNot fully observable');
end
```

Figure 7 MATLAB script to investigate the observability problem

The result of the *obsv* command, the O matrix, is shown in Table 5 and its rank is 3, meaning that the system is not only fully reachable but also fully observable.

0.00	1.00	0.00
0.00	0.00	1.00
-220008.58	-29.00	0.00
0.00	0.00	-7.52
17105578.78	841.00	-0.03
0.00	0.00	56.50

Table 5 Observability matrix

2.3 Proposed solution

To implement the lambda control, the solution, represented as a block diagram in Figure 8, has been adopted.

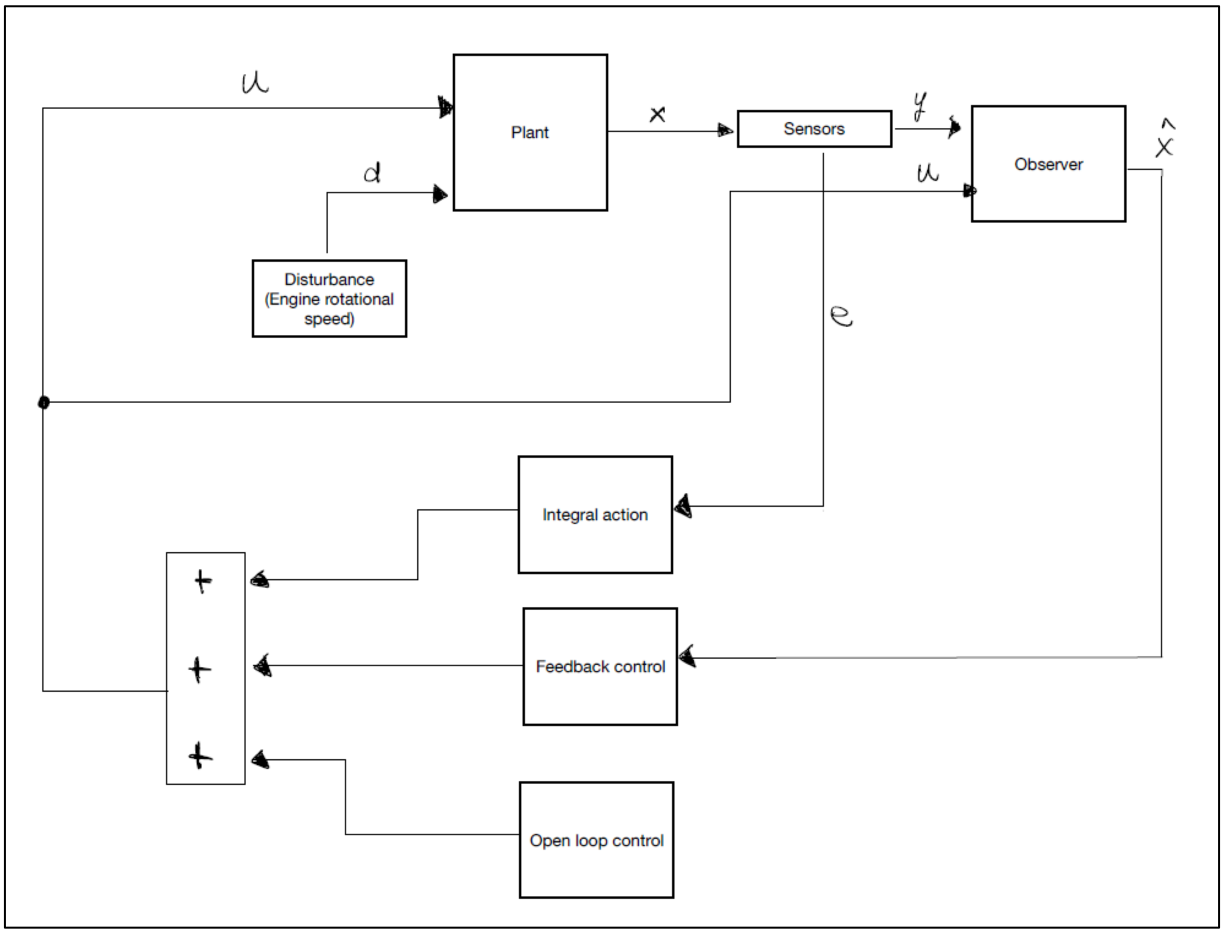


Figure 8 Proposed control system block diagram

As it can be noticed, there are 3 main contributions for the control purpose: the open loop control, whose main goal is to drive the plant towards the reference conditions defined by the user, the feedback, which provides stability of the plant and increases the performance regarding the achievement of the reference conditions, and, eventually, the integral action, which takes the error with respect to x_r to comply with the presence of an external disturbance.

2.3.1 Closed loop control

As previously mentioned, the closed loop control aim is to provide stability of the plant and, at the same time, it can improve the performance in reaching the reference conditions. It is based on the existence of the K_r matrix which, multiplied by the linearised plant vector $\tilde{x}$, furnishes the feedback control vector $\widetilde{u_{fb}}$, as reported in Eq. 19.

$$\widetilde{u_{fb}} = K_r\tilde{x} \tag{19}$$

To proper design the K_r matrix, the optimal control theory has been exploited and it is based on the so-called cost function J (Eq. 20). In the equation 2 additional matrices, both positive defined, are needed to design the cost function: Q, which tells, intuitively speaking, how much the control is interested in minimizing the error so, the higher Q the stronger will be the feedback, and R which, on the other hand, is related to the power consumption linked to the control action. Therefore, if R gets higher the feedback gets smoother since the "price" for the control actuation increases.

$$J = \int_0^\infty (\tilde{e}^T Q \tilde{e} + \tilde{u}^T R \tilde{u})\, dt$$

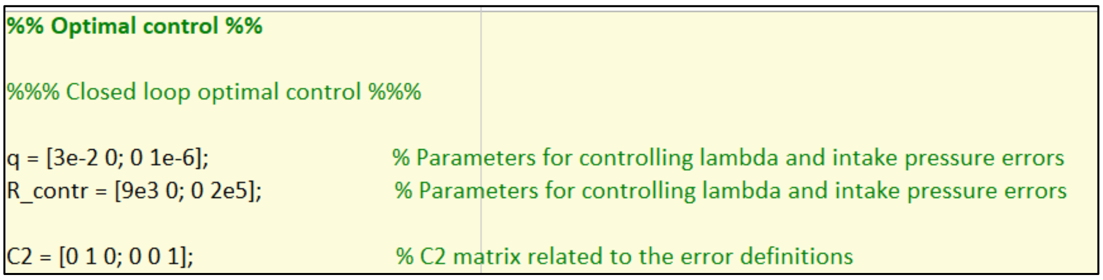
$$\tilde{e} = e - e_r = \begin{bmatrix} \lambda_{cyl} - \lambda_{cyl_{ref}} \\ p - p_{ref} \end{bmatrix} - \begin{bmatrix} \lambda_{cyl_{ref}} - \lambda_{cyl_{ref}} \\ p_{ref} - p_{ref} \end{bmatrix} = \begin{bmatrix} x_2 - x_{2_{ref}} \\ x_3 - x_{3_{ref}} \end{bmatrix} = \begin{bmatrix} \widetilde{x_2} \\ \widetilde{x_3} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tilde{x} = C_2 \tilde{x} \tag{20}$$

$$Q = \begin{bmatrix} q_{11} & q_{12} \\ q_{21} & q_{22} \end{bmatrix} = \begin{bmatrix} q_{11} & 0 \\ 0 & q_{22} \end{bmatrix}$$

$$R = \begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{bmatrix} = \begin{bmatrix} r_{11} & 0 \\ 0 & r_{22} \end{bmatrix}$$

For sake of simplicity, the Q and R matrices are considered as purely diagonal ones where, q_{11} and r_{11} indicate, respectively, how much the feedback control is interested in minimizing the AFR error and the actuation cost (fuel injection) related to it. The same applies for q_{22} and r_{22} as for the intake manifold pressure in which the throttle valve opening represents the actuation.

Once the matrices have been setup, to get to K_r, the *lqr* MATLAB command has been used (Figure 9). The Linear-Quadratic Regulator solves the associated Riccati equation to provide K_r, such that it minimizes the cost function J. Therefore, to achieve the desired performances, it is necessary to tune the Q and R matrices.



```
%% Optimal control %%

%%% Closed loop optimal control %%%

q = [3e-2 0; 0 1e-6];               % Parameters for controlling lambda and intake pressure errors
R_contr = [9e3 0; 0 2e5];           % Parameters for controlling lambda and intake pressure errors

C2 = [0 1 0; 0 0 1];                % C2 matrix related to the error definitions
```

Figure 9 MATLAB script for designing the optimal closed loop control

2.3.2 Integral action

The closed loop is not enough to provide a robust control since, some external disturbances could have an impact on the system. For this reason, a countermeasure needs to be taken, that is the integral action. It relies on the theory internal model, based on the assumption of knowing the disturbance model and, in particular, in the project a constant disturbance (a constant engine rotational speed different from the reference one) has been assumed. From the sensor measurements, the errors with respect to the reference conditions can be calculated and, subsequently, integrated to generate the new extended plant where, the integrated errors become part of it, as shown in Eq. 21.

$$\dot{\tilde{x_e}} = \begin{bmatrix} \dot{\tilde{x}} \\ \dot{\eta} \end{bmatrix} = \begin{bmatrix} A & 0 \\ C_2 & 0 \end{bmatrix} \begin{bmatrix} \tilde{x} \\ \eta \end{bmatrix} + \begin{bmatrix} B_1 \\ 0 \end{bmatrix} \tilde{u} + \begin{bmatrix} B_2 \\ 0 \end{bmatrix} \tilde{d}$$

$$\dot{\eta} = \tilde{e} \tag{21}$$

$$\tilde{u} = \begin{bmatrix} K_r & H \end{bmatrix} \begin{bmatrix} \tilde{x} \\ \eta \end{bmatrix}$$

The matrix H is in charge of steering the errors to zero and, the performance whereby does it, depends on the tuning of Q_e and R matrices of the extended plant optimal closed loop control (Eq. 22).

$$J = \int_0^{\infty} \left(\tilde{e_e}^T Q_e \tilde{e_e} + \tilde{u}^T R \tilde{u} \right) dt$$

$$\tilde{e_e} = \begin{bmatrix} C_2 & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \tilde{x} \\ \eta \end{bmatrix} \tag{22}$$

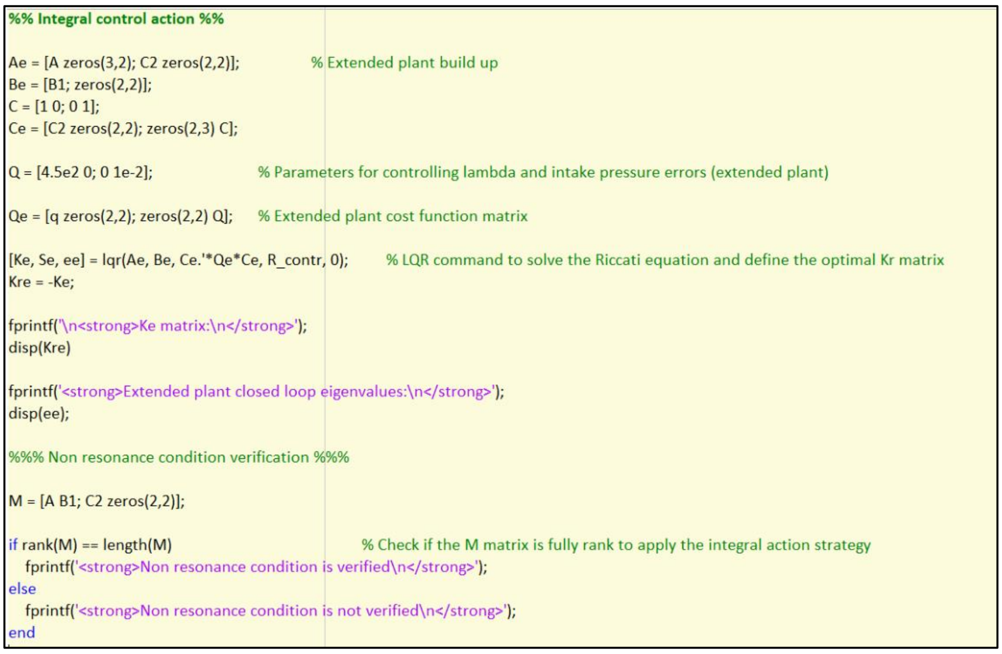
$$Q_e = \begin{bmatrix} Q & 0 & 0 \\ 0 & q_{e_1} & 0 \\ 0 & 0 & q_{e_2} \end{bmatrix}$$

By means of the *lqr* MATLAB command (Figure 10), K_r and H are generated at the same time and, once again, they are such that the cost function J is minimized. This time Q_e consist of 2 main parts: Q which is the same matrix as the one of Eq.20, and q_{e_1}, q_{e_2} which denote the "importance" of minimizing the errors (lambda and intake manifold pressure errors) given by the presence of the external disturbance. The higher those values, the stronger will be the integral action and the consequent disturbance control. The impact of the selection of all the parameters of the optimal control will be shown in the Chapter 3 where, all the simulation results will be plotted.

Moreover, an additional check must be performed to verify the possibility of putting in practise the integral action that is, the non-resonance condition verification. It is related to the inversion of the matrix written in Eq. 23 and to do that, the matrix must be full rank (5), otherwise the integral action control for MIMO systems cannot be implemented.

$$M = \begin{bmatrix} A & B_1 \\ C_2 & 0 \end{bmatrix} \tag{23}$$

Always in Figure 10 this kind of check is performed, and it is successful.



```
%% Integral control action %%

Ae = [A zeros(3,2); C2 zeros(2,2)];            % Extended plant build up
Be = [B1; zeros(2,2)];
C = [1 0; 0 1];
Ce = [C2 zeros(2,2); zeros(2,3) C];

Q = [4.5e2 0; 0 1e-2];                % Parameters for controlling lambda and intake pressure errors (extended plant)

Qe = [q zeros(2,2); zeros(2,2) Q];    % Extended plant cost function matrix

[Ke, Se, ee] = lqr(Ae, Be, Ce.'*Qe*Ce, R_contr, 0);        % LQR command to solve the Riccati equation and define the optimal Kr matrix
Kre = -Ke;

fprintf('\n<strong>Ke matrix:\n</strong>');
disp(Kre)

fprintf('<strong>Extended plant closed loop eigenvalues:\n</strong>');
disp(ee);

%%% Non resonance condition verification %%%

M = [A B1; C2 zeros(2,2)];

if rank(M) == length(M)                      % Check if the M matrix is fully rank to apply the integral action strategy
   fprintf('<strong>Non resonance condition is verified\n</strong>');
else
   fprintf('<strong>Non resonance condition is not verified\n</strong>');
end
```

Figure 10 MATLAB script to implement the integral control action strategy and to verify the non-resonance condition

2.3.3 Observer

The last step in the control framework design is the observer, whose main purpose is to estimate the real plant considering the measurements of the inputs and outputs of the real plant itself. The observer model refers to the Luenberger one, whose dynamics is reported in Eq. 24, where the "hat" symbols refer to the estimated ones. If the system is observable, the estimated plant $\hat{x}$ will converge asymptotically to the physical one $\tilde{x}$ once the K_o matrix has been properly designed.

$$\dot{\hat{x}} = A\hat{x} + B_1\tilde{u} + K_o(\tilde{y} - \hat{y}) \qquad \textit{Luenberger observer dynamics}$$

$$\tilde{u} = K_r\tilde{x} \approx K_r\hat{x} \tag{24}$$

$$\hat{y} = C_1\hat{x}$$

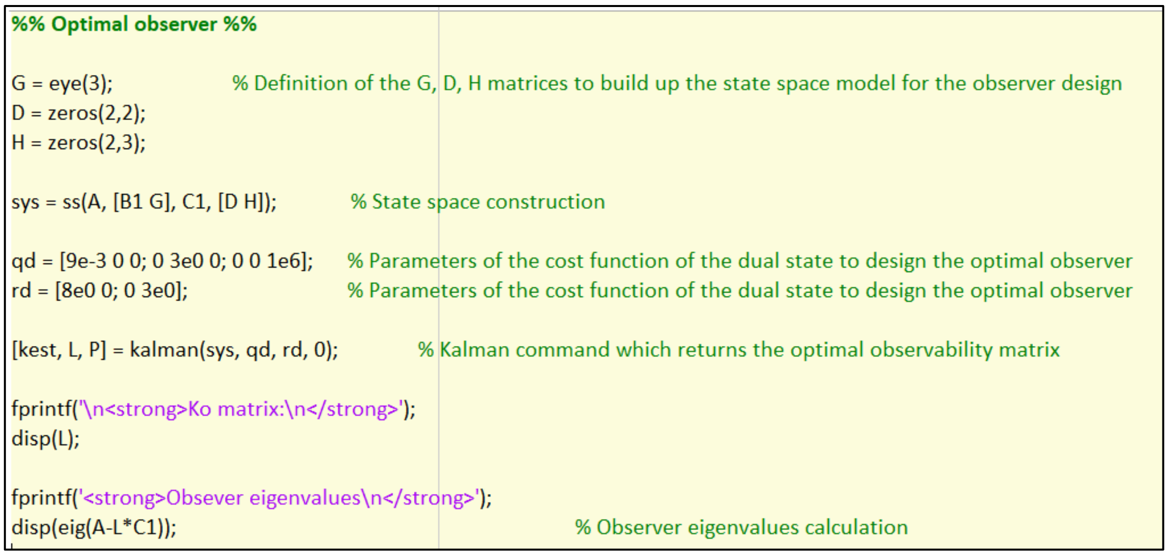
Given the state space model of the plant together with the process and measurement noise covariance data, the *kalman* MATLAB command returns the K_o matrix, which represents the Kalman filter optimal solution. In Eq. 25 is reported the state space model from the MATLAB point of view so to speak, in which some new variables turn up, such as w, the process the noise, which represents the possibility of the system to change its state over time without knowing the exact details of when and how this will occur, and v, the white measurement noise already defined in Eq. 16. To comply with the plant definition, D and H are set to 0 while G is an identity matrix 3x3.

$$\dot{\tilde{x}} = A\tilde{x} + B_1\tilde{u} + Gw \qquad (state\ equation)$$
$$\tilde{y} = C_1\tilde{x} + D\tilde{u} + Hw + v \qquad (measurement\ equation) \tag{25}$$

Also in this case, for simplicity, the 2 covariance matrices Q_d and R_d (Eq. 26) are considered as diagonal ones. To give further details, the Q_d matrix is related to the process noise, in particular the higher its elements the higher the degree of uncertainty given to the plant evolution over time, that means the Kalman filter will push more on observation ($\hat{x}$ will converge to $\tilde{x}$ faster) and the observer eigenvalues will be higher. However, there is a side effect because the measurement noise will not be filtered out and it will be included in the feedback control making the algorithm less performant. So, a trade-off must be found with the R_d matrix, which has an impact on the measurement noise v: the higher $r_{d_{11}}$ and $r_{d_{22}}$ more enhanced will be the noise filtering, making smoother the quantity read-outs.

$$Q_d = \begin{bmatrix} q_{d_{11}} & q_{d_{12}} & q_{d_{13}} \\ q_{d_{21}} & q_{d_{22}} & q_{d_{23}} \\ q_{d_{31}} & q_{d_{32}} & q_{d_{33}} \end{bmatrix} = \begin{bmatrix} q_{d_{11}} & 0 & 0 \\ 0 & q_{d_{22}} & 0 \\ 0 & 0 & q_{d_{33}} \end{bmatrix}$$
$$R_d = \begin{bmatrix} r_{d_{11}} & r_{d_{12}} \\ r_{d_{21}} & r_{d_{22}} \end{bmatrix} = \begin{bmatrix} r_{d_{11}} & 0 \\ 0 & r_{d_{22}} \end{bmatrix} \tag{26}$$

Moreover, in Figure 11 is displayed the MATLAB script regarding the design of the optimal observer.



```
%% Optimal observer %%

G = eye(3);              % Definition of the G, D, H matrices to build up the state space model for the observer design
D = zeros(2,2);
H = zeros(2,3);

sys = ss(A, [B1 G], C1, [D H]);          % State space construction

qd = [9e-3 0 0; 0 3e0 0; 0 0 1e6];      % Parameters of the cost function of the dual state to design the optimal observer
rd = [8e0 0; 0 3e0];                     % Parameters of the cost function of the dual state to design the optimal observer

[kest, L, P] = kalman(sys, qd, rd, 0);          % Kalman command which returns the optimal observability matrix

fprintf('\n<strong>Ko matrix:\n</strong>');
disp(L);

fprintf('<strong>Obsever eigenvalues\n</strong>');
disp(eig(A-L*C1));                                   % Observer eigenvalues calculation
```

Figure 11 MATLAB script to implement the optimal observer

Chapter 3

Application

3.1 Simulator description

In the following chapter the simulator description will be carried out, describing the MATLAB code and the Simulink scheme, shown in Figure 12, block by block.

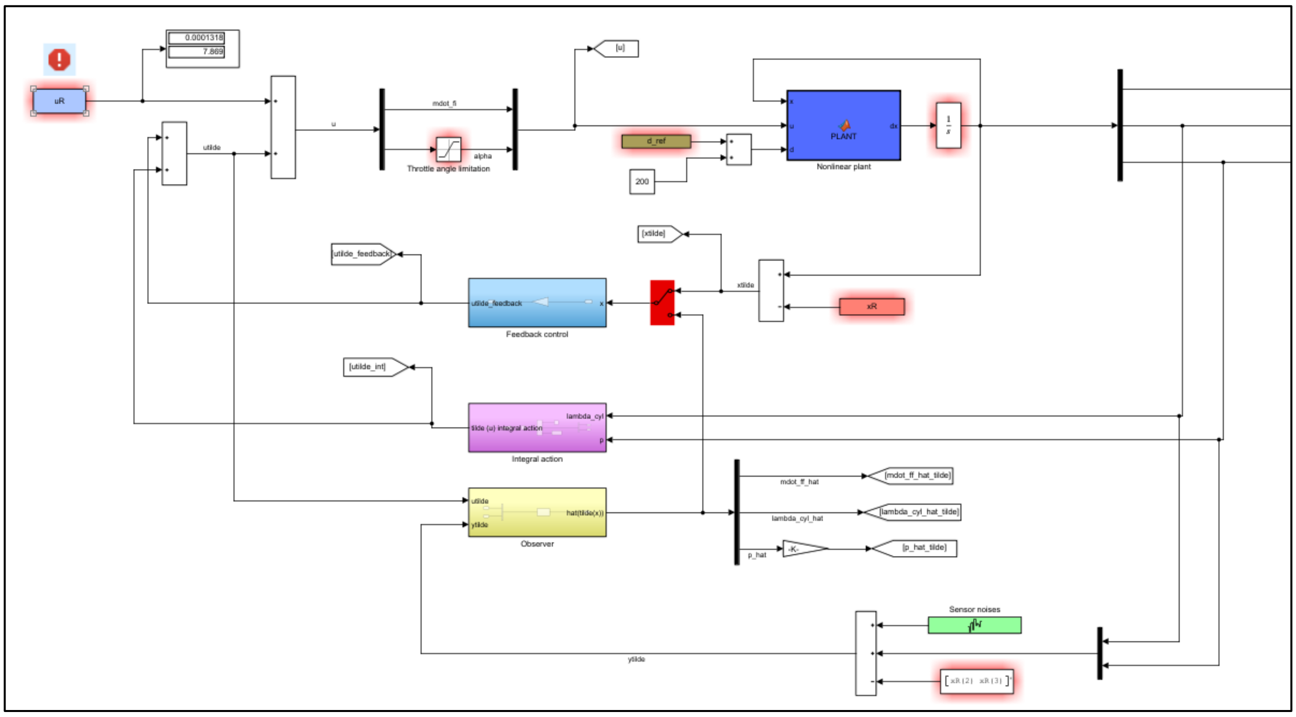


Figure 12 Simulink bird's eye view

3.1.1 Plant block

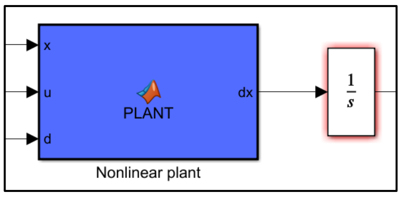
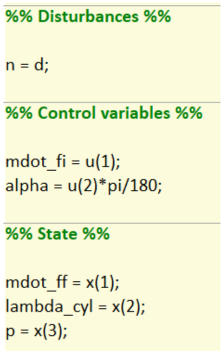


Figure 13 Nonlinear plant block

The plant block (Figure 13) implements the equations, reported in Eq. 12 so that the dynamics of the plant can be simulated. Based on the control theory the numerical computation of the states is achieved by taking as inputs the state itself x, the control vector u and the disturbance d, as shown in Figure 14.



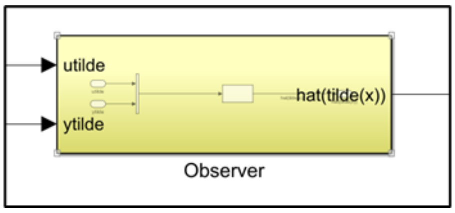
```
%% Disturbances %%

n = d;

%% Control variables %%

mdot_fi = u(1);
alpha = u(2)*pi/180;

%% State %%

mdot_ff = x(1);
lambda_cyl = x(2);
p = x(3);
```

Figure 14 Plant inputs

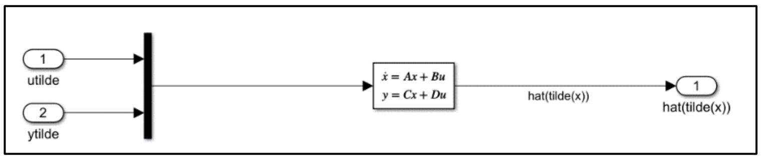
Moreover, inside the block all the parameters describing the engine with the related time constants are listed (see Appendix) and used for simulation purposes. The output is the dynamics of the state vector x which needs to be integrated to get the actual state.

3.1.2 Observer block

The observer block estimates the state adopting the Luenberger strategy, as reported in Eq. 24. The inputs are the linearised control vector $\tilde{u}$ and the sensor measurements $\tilde{y}$ which enter the state space Simulink block as shown in Figure 15 B. By means of the results of the *kalman* command, the state estimation $\hat{x}$ is then carried out.



A)



B)

Figure 15 Left: observer block. Right: observer explosion

3.1.3 Feedback Control Loop

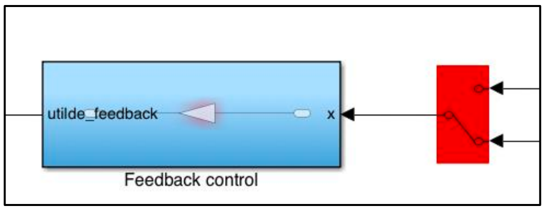
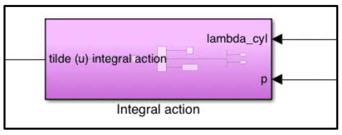


Figure 16 Feedback Control Block

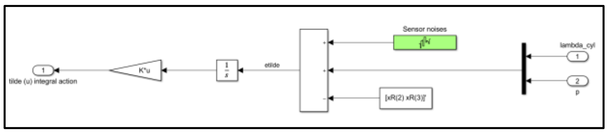
Through the feedback control block, shown in Figure 16, the simulator builds part of the control vector $\tilde{u}$. As known, the control can be defined in more general ways as the sum of a closed loop and an open loop contribute, which, in the simulator, has been neglected for sake of simplicity. The approach to define mathematically the feedback block has been carried out by the means of the optimal control strategy through the definition and the tuning of the cost function parameters.
Finally, the *switch* in the Simulink model let the possibility to build the vector $\tilde{u}$ taking as input the real state or the estimated one. Obviously, the results, which are reported in Chapter 3.2, have been obtained simulating a real control scheme by computing from the observed state. In the results section, reported in Chapter 3.2, a simulation comparison will be shown between the closed loop performances achieved taking as input both the real state and the observed one.

3.1.4 Integral Action

The last block of the control scheme is the Integral Action Block (Figure 17 A) which contributes to steer the state to the reference one by accounting the history of the errors, calculated from the measured variables, p and λ_{cyl}, and the reference conditions. To achieve a better simulation of the real environment, the block considers also the noises associated to the sensors (Figure 17 B), whose values has been fetched by their datasheets.
To build the block, the MATLAB script implements the extended system and performs the validation of the '*Non resonance Condition*' and, eventually, the tuning of the 2 additional parameters q_{e_1} and q_{e_2} of cost function (Eq. 22) is tackled to meet the desired performances.
In the end, the integral action contribution is added to complete the linearised control vector $\tilde{u}$.



A)



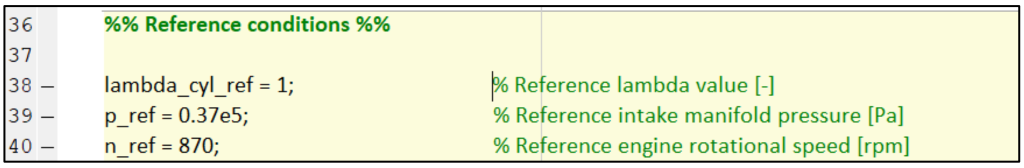
B)

Figure 17 A): integral action block. B): under the block

3.1.5 Main Script

The Simulink environment, to properly run the simulation, needs all the parameters used to design the control algorithm, which cannot be stored inside Simulink itself, but they have to be loaded from MATLAB. To do that, is it recommended to write the main script file, where all the parameters, for example of the engine, are saved and then retrieved by Simulink.

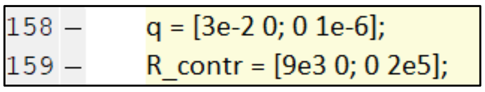
The main script is divided into 9 sections, entirely shown in the Appendix. In section 1, the parameters of the engine are reported together with some constant needed to estimate the fuel film vaporisation constant, the fuel film fraction constant, and the engine volumetric efficiency.
In section 2, the initial conditions are written, and they are essential to run the simulation in the very first seconds.
Section 3 deals with the calculation of the u_R vector, that is the reference control action to enable the system to achieve the reference conditions define by the user from line 38 to line 40 (Figure 18). The algorithm automatically computes the reference fuel mass flow rate $m_{\dot{f_i}_{ref}}$ and the throttle plate angle α_{ref} to reach the reference pressure and lambda values.



```
36      %% Reference conditions %%
37
38 –    lambda_cyl_ref = 1;                % Reference lambda value [-]
39 –    p_ref = 0.37e5;                    % Reference intake manifold pressure [Pa]
40 –    n_ref = 870;                       % Reference engine rotational speed [rpm]
```

Figure 18 Reference condition setup in the main script

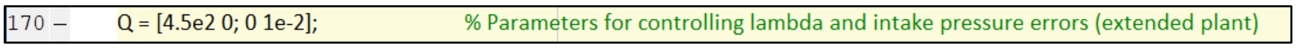
Section 4 builds the A, the B$_1$, and the B$_2$ matrices resulting from the linearisation procedure carried out in the Appendix and it calculates the open loop eigenvalues, reported in Table 3.
Section 5 and 6 investigate the reachability and observability of the plant respectively, considering both the reachability and the observability matrices. They check if the rank of the 2 matrices is maximum to verify whether the system is fully reachable and observable otherwise, the Kalman decomposition is performed.
Section 7 is related to the closed loop optimal control design, where the Q and R matrices of the cost function (Eq. 20) are built and tuned through an iterative process. The control designer can adjust the values of the matrices from line 158 to line 159 in the main script file, as shown in Figure 19. Moreover, the Linear-Quadratic-Function is recalled to find the K_r matrix, such that the cost function J is minimized.



```
158 –        q = [3e-2 0; 0 1e-6];
159 –        R_contr = [9e3 0; 0 2e5];
```

Figure 19 Q and R matrices of the cost function

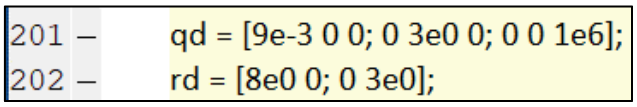
Section 8 expands the optimal control to the extended plant (Eq. 21-22), so the A_e, the B_e, the C_e and the Q_e matrices are defined and then, the same MATLAB command is utilised to design the additional contribution to the closed loop control that is, the integral action. At line 184 (Figure 20) the designer can change the parameters of the Q_e matrix to achieve the demanded performances. Furthermore, the non-resonance condition check is performed, and the systems warns if it is not fulfilled.



```
Q = [4.5e2 0; 0 1e-2];            % Parameters for controlling lambda and intake pressure errors (extended plant)
```

Figure 20 Main script line at which the designer can change the control parameters for the integral control action

Finally, Section 9, addresses the observer design and in particular the G, D, and the H matrices are defined to match the plant equation (Eq. 14) with the one the *kalman* command can process (Eq. 25). Subsequently, the state space model is built through the *ss* command which takes as inputs the A, the B_1, the C_1, the D, the G and the H matrices. Then, once the $Q_d$and the R_d (Eq. 26) have been defined by the user (Figure 21), the *kalman* command is recalled giving as a result the K_o matrix, which enables to estimate the system through the Luenberger observer from the sensor measurements. Eventually, the observer eigenvalues are calculated and displayed in the command window.



```
qd = [9e-3 0 0; 0 3e0 0; 0 0 1e6];
rd = [8e0 0; 0 3e0];
```

Figure 21 Q_d and R_d matrices of the cost function related to the dual systems to define the optimal observer

3.2 Simulation results

In this Chapter the main simulation results will be shown making a comparison between the real state and the estimated one through the observer.

3.2.1 Closed loop

Looking at Figure 2, the plant open loop evolution shows a quite important overshoot during the transient, in which λ_{cyl} becomes greater than 1.3, which represents the lean limit for the combustion stability. To make up for it, and to provide further stability of the plant, a proper feedback control action must be put in practise. Following the theory guidelines reported in Section 2.3.1, a suitable tuning of the Q and R matrices has been carried out to improve the performance in the fulfilment of the chosen reference conditions, as displayed in Figure 22.

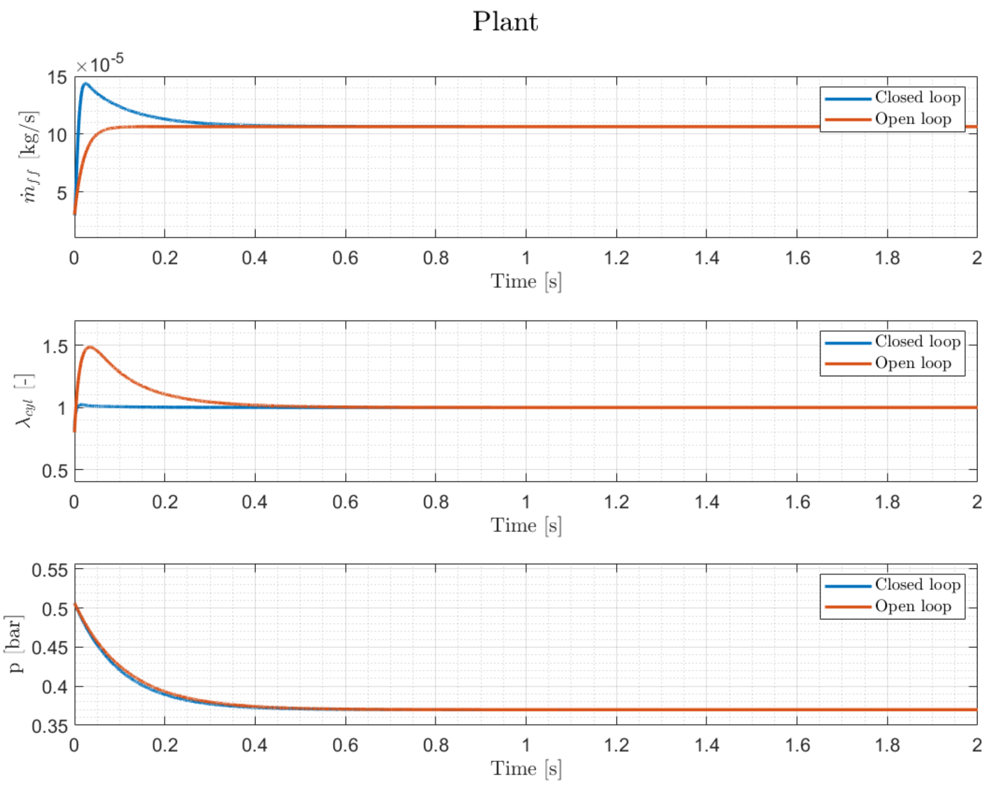


Figure 22 Open loop vs closed loop simulation

As it can be seen, the performance has been notably improved, indeed, the overshoot has been reduced so that the maximum value reached by lambda is about 1.02, a value which prevents an unstable combustion. Moreover, the reference λ_{cyl} is reached in a faster way because, to compensate for the excessive lean mixture in the open loop case, the injector introduces more fuel and the fuel film mass flow rate, being related to the injected fuel, presents an overshoot.
Eventually, in Table 6 are shown the closed loop eigenvalues and, as it can be noticed, their real part is more negative with respect to the open loop case, and this provides more stability of the plant. They have been obtained through the optimal control considering as parameters the Q and R matrices reported in Figure 19.

Open loop eigenvalues	Closed loop eigenvalues
-29.00	-103.11 + 108.43i
-48.75	-103.11 - 108.43i
-7.52	-107.11

Table 6 Comparison between closed and open loop eigenvalues

3.2.2 Integral action

The integral action is crucial to manage properly the presence of an external disturbance (in this case the engine rotational speed). As a matter of fact, the closed loop control only is not sufficient to drive the plant at its reference conditions, as shown in Figure 23.

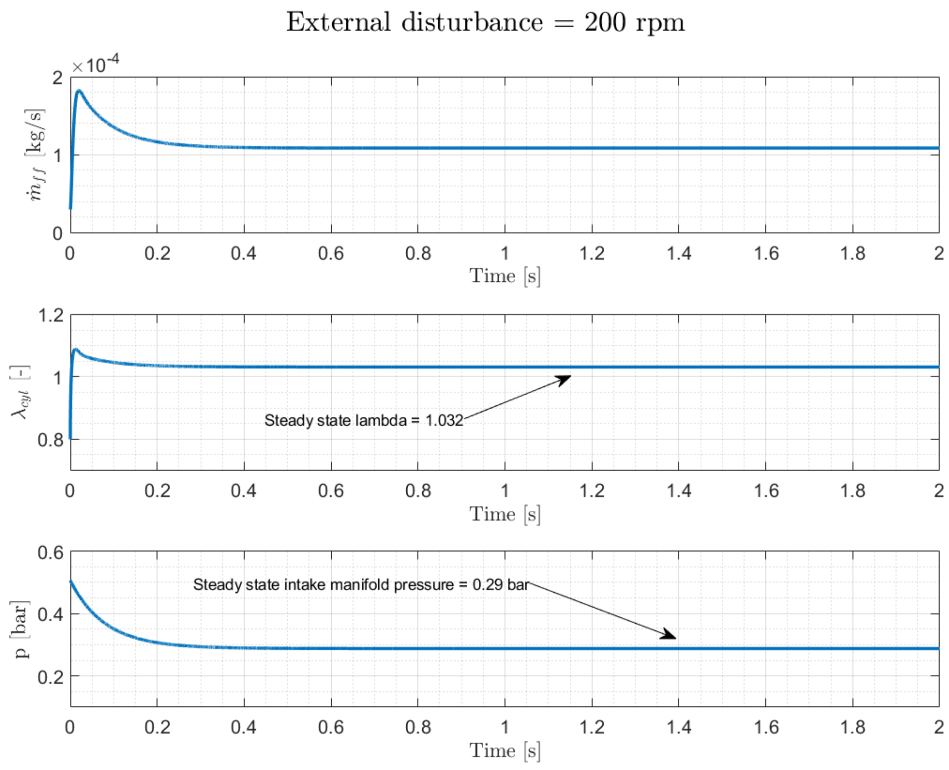


Figure 23 Simulation output in case of an external disturbance (200 rpm) without the integral action

As it can be noticed, the steady state values for λ_{cyl} and p are not corresponding to the reference ones and, in particular, the steady state value of the AFR is out of the window to obtain the maximum conversion efficiency of the three-way catalyst (TWC), as Figure 24 proves [11]. Therefore, it is necessary to arrange a suitable integral action to minimize the steady state error, thanks to the implementation of the extended plant as discussed in Section 2.3.2. The outcome, shown in Figure 25, proves that the system comes back to its own reference conditions within 0.5 seconds.

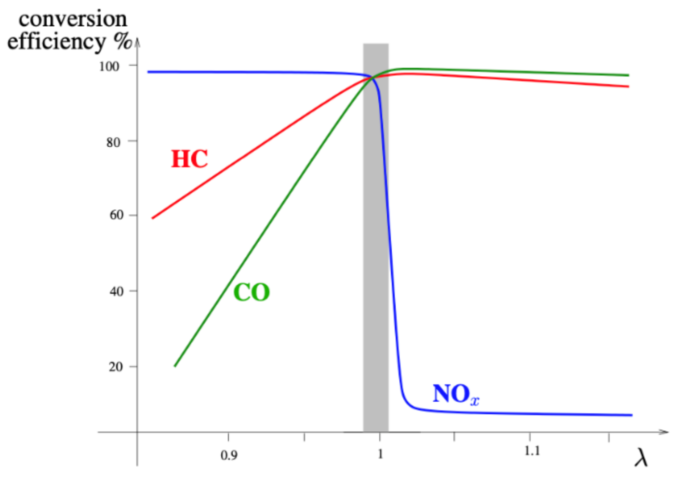


Figure 24 Three-way catalyst conversion efficiency as a function of the AFR

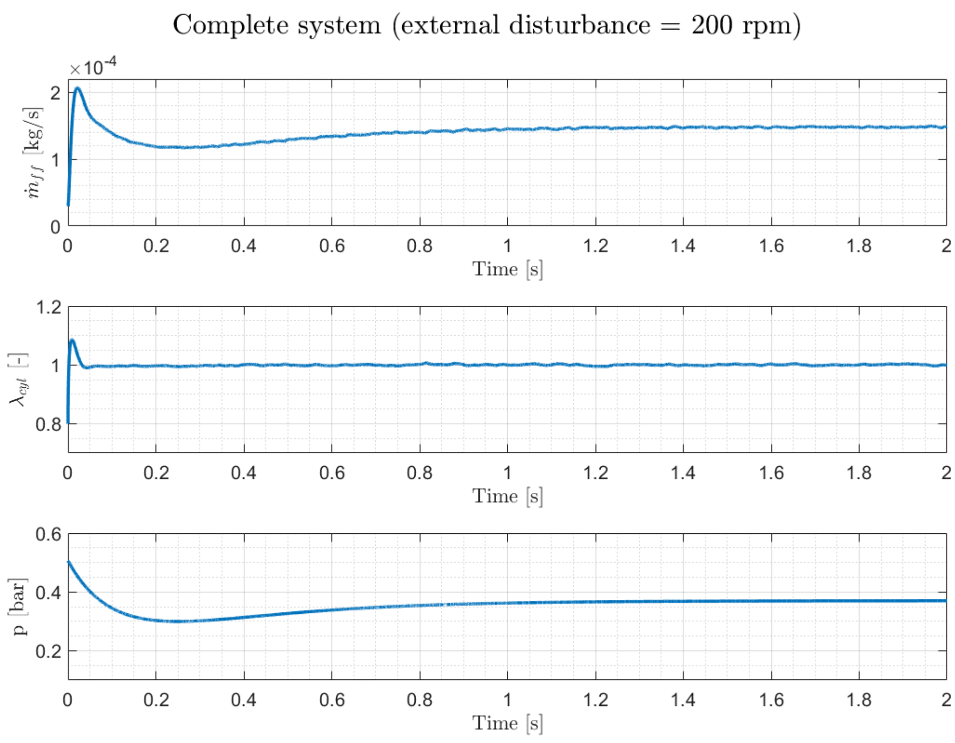


Figure 25 Complete system simulation (with the integral action) considering an external disturbance equal to 200 rpm

Moreover, in Table 7 are reported the extended closed loop eigenvalues, which have been got applying the optimal control strategy taking into account the Q_e and R matrices shown in Figure 20. The last 2 eigenvalues are referred to the additional contribution on the back of the 2 errors, related to λ_{cyl} and p, as shown in Equation 20-21.

Extended plant closed loop eigenvalues
-103.11 + 108.43i
-103.11 - 108.43i
-107.11
-30.70 + 27.71i
-30.70 - 27.71i

Table 7 Extended plant closed loop eigenvalues

3.2.3 Observer

The last step in control design is the implementation of the observer based on the Luenberger theory. In practise, from the sensor measurements the observer estimates the plant but, it must be carefully designed to meet the desired performances. As a matter of fact, the $r_{d_{11}}$ values of the R_d matrix, shown in Equation 26, has been selected higher than $r_{d_{22}}$ to filter more the measurement noise coming from the lambda sensor, since λ_{cyl} value is crucial for both the combustion and the aftertreatment conversion efficiency. Instead, as far as the Q_d matrix is concerned, it has been assumed that the first state has not a large "unpredictability", so the $q_{d_{11}}$ value has an order of magnitude much lower than the other 2 values constituting the Q_d matrix.

Considering the input parameters for the *kalman* command, shown in Figure 21, and performing the closed loop simulation without the integral action, the observer performance is plotted in Figure 26.

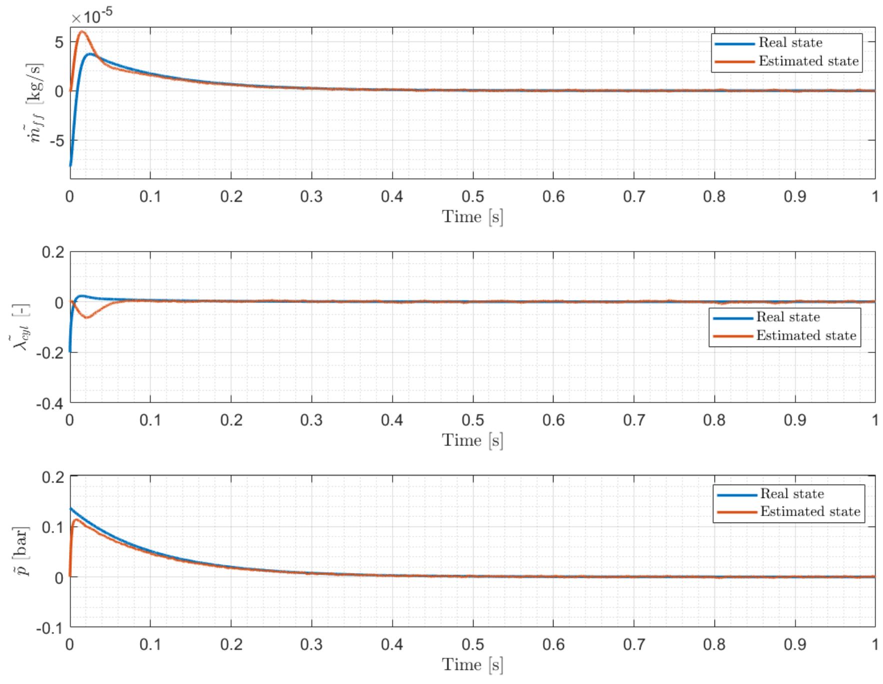


Figure 26 Observer performance

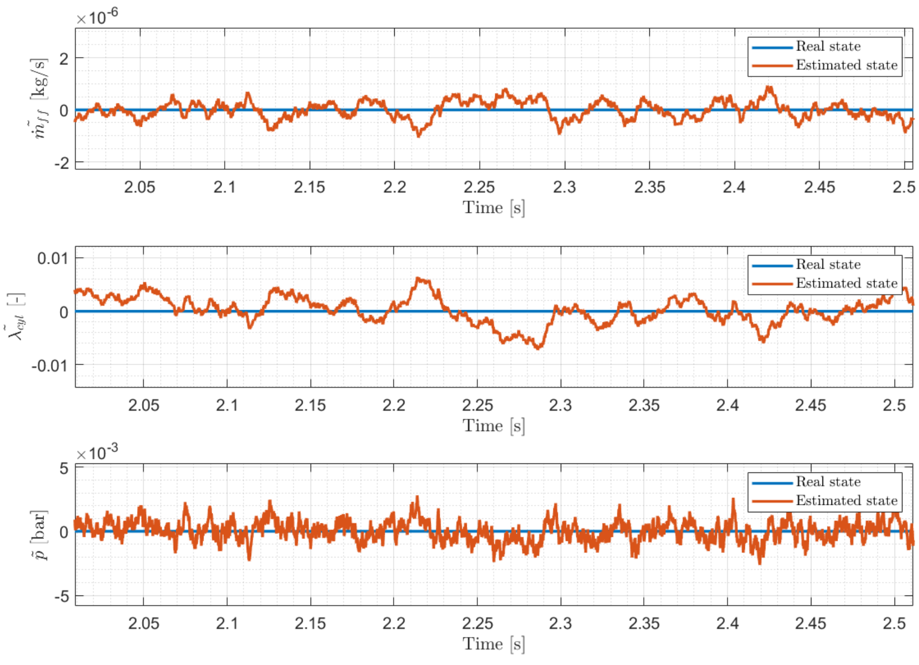


Figure 27 Observer performance at steady state zoomed

As it can been seen from Figure 26, the observer converges very quickly to the reference conditions, and this represents an upside regarding the performance of control itself as it will be shown in the following Section. Moreover, looking at the plots more closely, it is possible to appreciate the sensor noise effects on the estimation of the state, so the Q_d and R_d matrices have to be tuned such that the noise effects do not have an impact on the control performance.

Furthermore, considering the presence of a constant external disturbance (200 rpm), the observer does not converge to the real plant but, despite following its trend (Figure 28), it presents a steady-state error due to the fact that the disturbance model is not included in the observer definition.

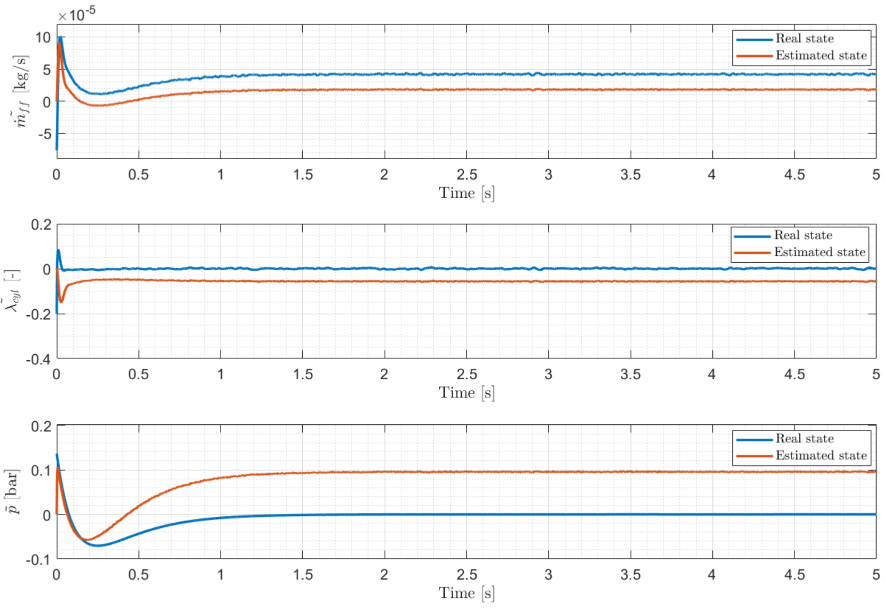


Figure 28 Observer steady state offset

Nevertheless, the observer steady state error does not affect the achievement of the plant reference conditions thanks to the compensating effect of the integral action, as shown in Figure 29.

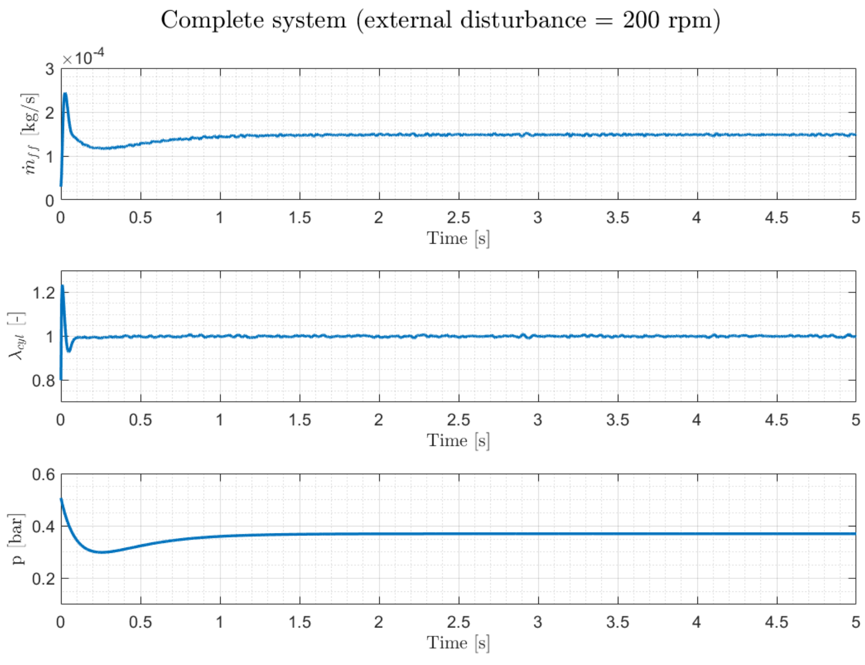
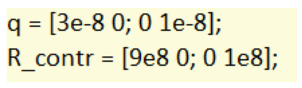


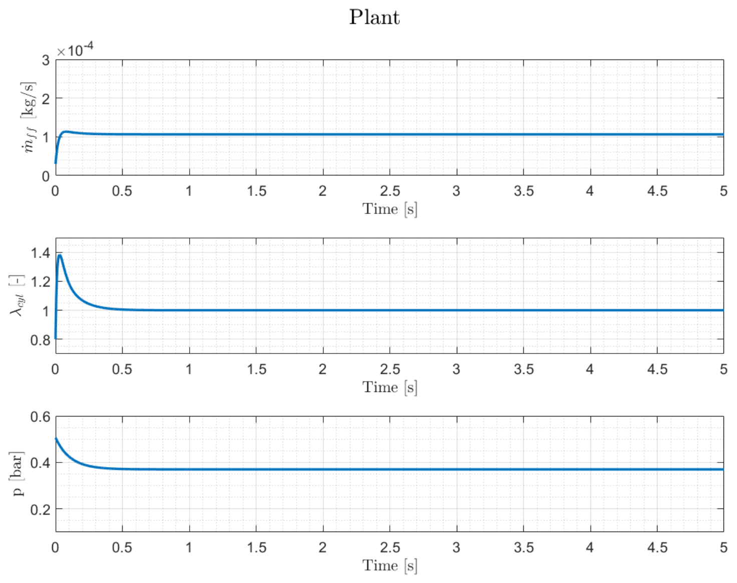
Figure 29 Complete simulation with the estimated state

3.2.4 Tuning

To properly setup the control algorithm, it is necessary to tune in the best possible way the control parameters, that is the Q, Q_e, R, Q_d and R_d matrices. Starting from the open loop simulation, an iterative process has been put in practise to properly select the aforementioned parameters. The first step was the closed loop tuning, involving the Q and R matrices. At the very beginning the closed loop control has been kept quite "gentle" choosing the parameters shown in Figure 30 to better see the weight of each parameter in the control performance.



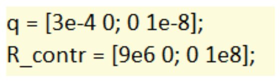
```
q = [3e-8 0; 0 1e-8];
R_contr = [9e8 0; 0 1e8];
```



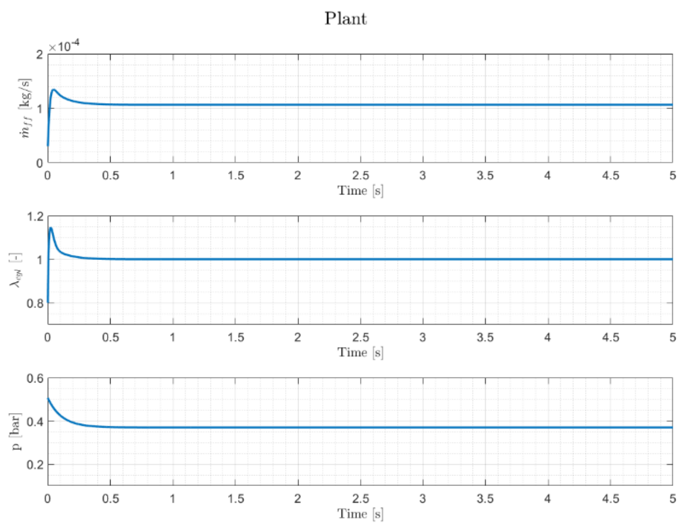
A) B)

Figure 30 Initial setting for the optimal closed loop control

The initial overshoot of lambda cannot be tolerated since, the lambda value overcomes the lean limit for the combustion stability, as previously mentioned. So, a more powerful action is needed to keep lambda under control, and this has led to a new set of parameters (Figure 31).



```
q = [3e-4 0; 0 1e-8];
R_contr = [9e6 0; 0 1e8];
```

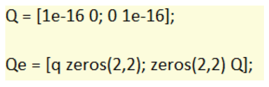


A) B)

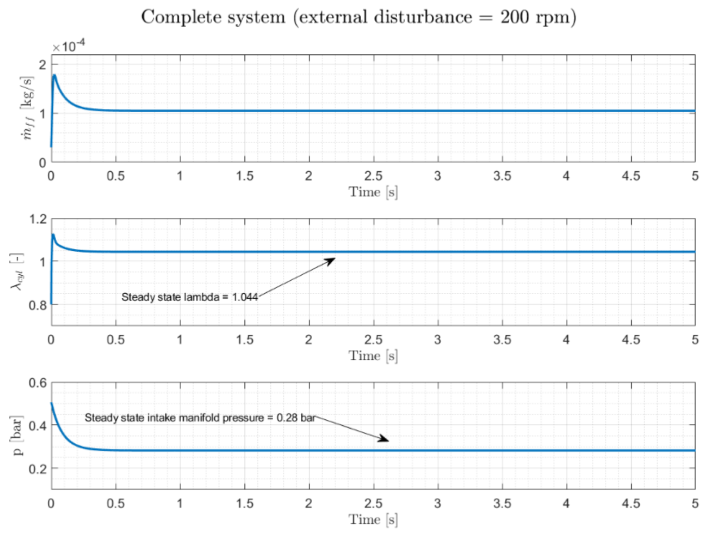
Figure 31 Finer setting for the optimal closed loop control

In this second attempt, the q_{11} and the r_{11} parameters have been respectively increased and reduced so that, the closed loop control is allowed to act more on the fuel injection and the initial overshoot of lambda is then limited, leading to a more stable combustion since the lean lambda limit is not overcome. In spite of that, it is not acceptable for the TWC a lambda value above 1 since, the NO_x conversion efficiency would dramatically drop (Figure 24). To make up for it, a further tuning of the parameters is needed and the final values which have been selected are those represented in Figure 19 so that, the achieved control performance is the one of Figure 22.

Once the closed loop control has been tuned properly, in the presence of an external disturbance it is needed to introduce the integral action as previously mentioned. As a result, an additional tuning of the Q_e matrix has to be performed and, to do that, the same approach of the closed loop tuning has been followed. In the first attempt (Figure 32), the section of the Q_e matrix relative to the integral action has been "shut off" and, the simulation result is similar to Figure 23.



```
Q = [1e-16 0; 0 1e-16];

Qe = [q zeros(2,2); zeros(2,2) Q];
```

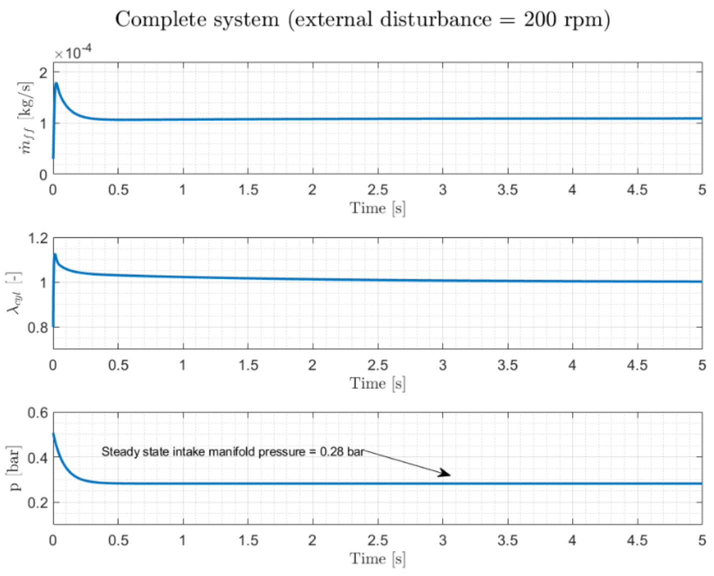


A) B)

Figure 32 Simulation result in the integral action tuning starting point

Since λ_{cyl} is a key parameter and it must be carefully controlled, the second tuning attempt was meant to improve the performance in the reduction of the lambda error, leading to a new selection of the q_{e_1} value of the Q_e matrix (Equation 22). In the specific, it has been increased up to 0.01 so that the integral action concerning lambda is more powerful.

```
Q = [1e-2 0; 0 1e-16];

Qe = [q zeros(2,2); zeros(2,2) Q];
```

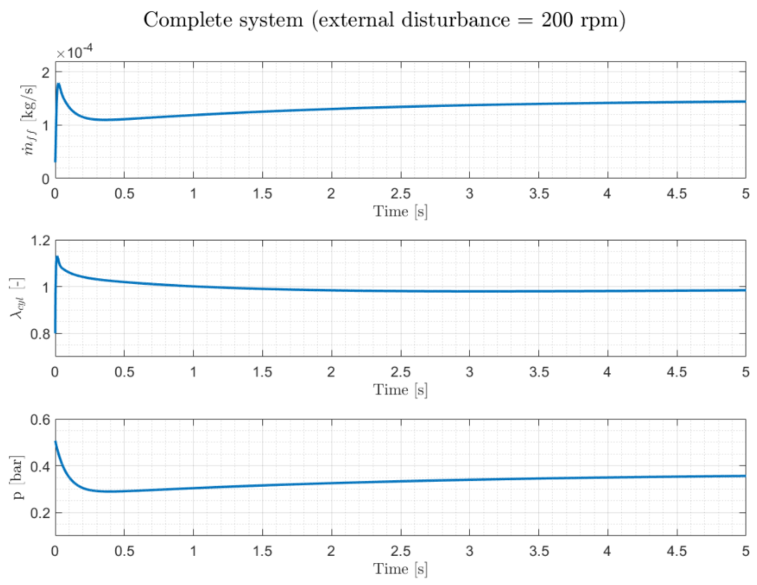


A) B)

Figure 33 Simulation result for integral action tuning (2nd attempt)

As it can been noticed, lambda tends to its reference value in 5 s unlike the previous example. This outcome is not acceptable since, the TWC environment would be in excess of oxygen for too long time resulting in a poor NO_x conversion efficiency. Moreover, the intake manifold pressure shows the same behaviour as Figure 32, because q_{e_2} has not changed. So, in the next attempt, also the pressure error has been considered, resulting in a further new selection of q_{e_2}.

```
Q = [1e-2 0; 0 2e-4];

Qe = [q zeros(2,2); zeros(2,2) Q];
```



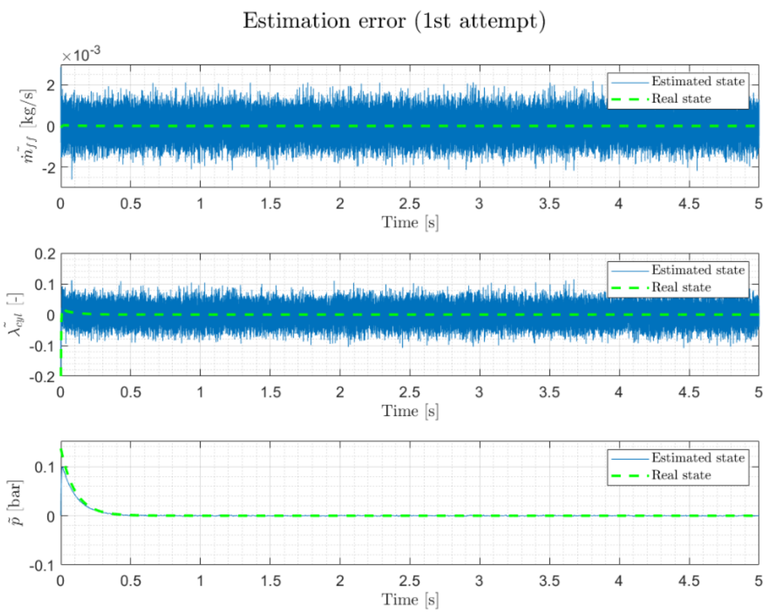
A) B)

Figure 34 Simulation result for integral action tuning (3rd attempt)

As it can be seen, the intake manifold pressure tends to its reference value (0.37 bar) but, λ_{cyl} crosses 1 and it remains slightly rich for too long time, worsening the CO and HC aftertreatment performance. As a consequence, an iterative process has been carried out to set the final values of the Q_e matrix, reported in Figure 20, whose simulation results are plotted in Section 3.2.2.

The last step in the tuning process regards the observer. The initial values for the Q_d and R_d matrices and the performance outcome are respectively shown in Figure 35 A and 35 B.

```
qd = [1e8 0 0; 0 1e8 0; 0 0 1e8];
rd = [1e3 0; 0 1e3];
```

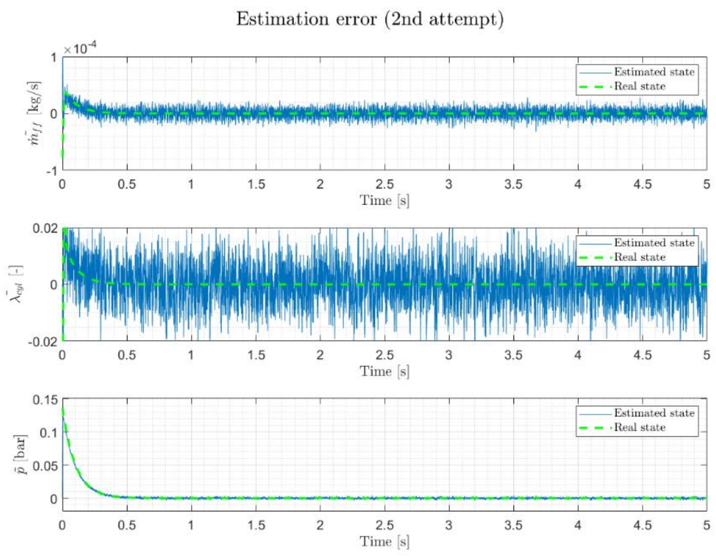


A) B)

Figure 35 Observer tuning starting point

As it can be spotted, the convergence performance is quite satisfactory, but the observer precision is not that high, due to the sensor noises effect. To filter more the measurement noise, r_{d_1} and r_{d_2} have been increased and q_{d_1} and q_{d_2} have been reduced, leading to a new set of parameters (Figure 36).

```
qd = [1e2 0 0; 0 1e2 0; 0 0 1e8];
rd = [1e2 0; 0 1e2];
```



A) B)

Figure 36 Simulation result for observer tuning (2nd attempt)

From Figure 36, the noise filtering has been significantly improved since, for example, $\widehat{\lambda_{cyl}}$ oscillates between -0.2 and 0.2. To further increase the observer performance, also in this case, an iterative process has been performed and the final set of parameters for the observer are the ones of Figure 21.

In the following graphs is plotted the simulation results carried out considering the estimated state and not the real one.

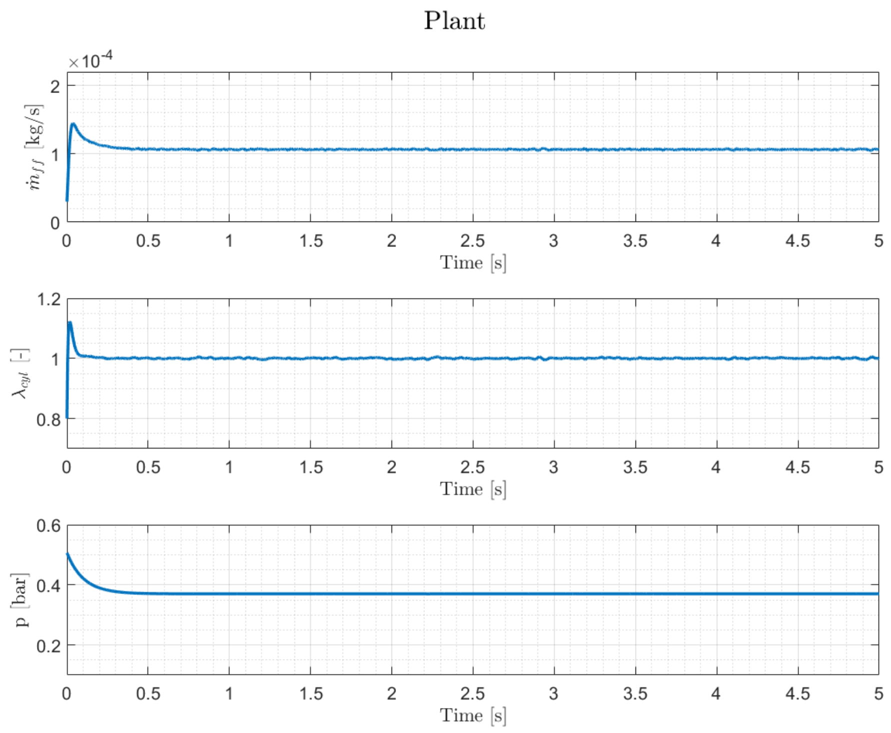


Figure 37 Closed loop simulation result with the estimated state

With respect to the simulation done with the real state, at the very beginning during the transient, the lambda overshoot is a bit more emphasised, since the observer needs some time to converge to the reference value. Moreover, some oscillations can be spotted due to the presence of the measurement noise.

As far as regards the simulation carried out with the external disturbance, it has been already shown in Figure 29.

3.3 Transient response

To further investigate the plant, an analysis in case of a not constant disturbance has been performed. Indeed, a linear time varying engine rotational speed (from 870 rpm to 1070 rpm in 3 seconds) has been applied to the plant to see how the control reacts. However, the implemented control algorithm is not suitable to deal with time varying disturbance but, for sake of curiosity this kind of simulation has been carried out (Figure 38).

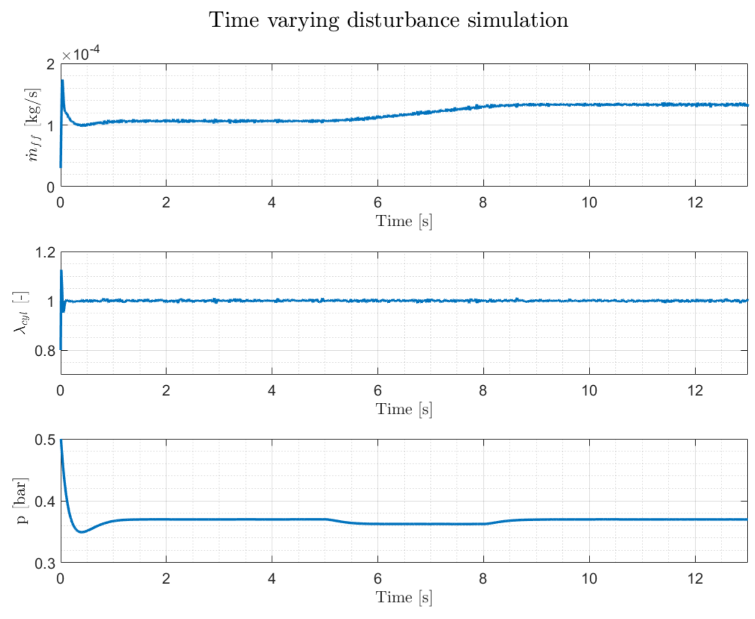


Figure 38 Simulation result in case of a linear time varying disturbance

Between 5 and 8 seconds of simulation (time interval in which the time varying disturbance is applied), the intake manifold pressure drops and settles to a new lower value, 0.3625 bar, since the engine is accelerating and more air for the same interval time is demanded. Despite this pressure drop, λ_{cyl} stays around 1 even if the control is not designed to deal with transient conditions. When the speed reaches the new steady state value (1070 rpm), the integral action brings the intake pressure back to its reference value (0.37 bar) opening the throttle valve and, to compensate for the more incoming air, the injector introduces more fuel to keep lambda equal to 1, as shown in Figure 39.

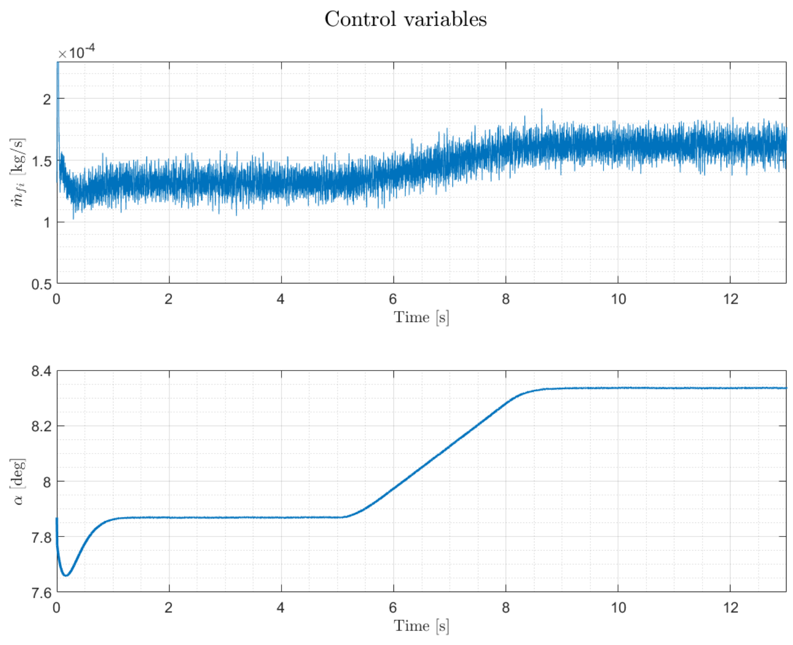


Figure 39 Control variables time evolution

Chapter 4

Conclusions and further investigations

The purpose of this study was to develop a control system for the regulation of the AFR of a spark ignition engine in order to minimize, as much as possible, the exhaust pollutants and gas emissions. The objective has been reached through a case study by controlling lambda using a MIMO controller. The engine, simulated in MATLAB and Simulink, was a 1275cc British Leyland and the MVEM (Mean Value Engine Model) has been exploited to proper model the engine plant. The lambda control algorithm is made of three contributions:

- **Open loop control** action, which leads the system to the reference conditions
- **Closed loop control** action, which increases both the stability and the performance of the control system
- **Integral** action, which tackles the presence of external disturbances

Furthermore, the Luenberger observer has been introduced to estimate the real plant from the sensor measurements and the linearized control inputs.

Finally, the results of the simulations have been illustrated, and it appears clear that the control system can drive lambda, towards the reference values, with a satisfactory accuracy. Indeed, the introduction of the closed loop control and of the integral action has significantly improved the performances in terms of speed and stability, also in the presence of an external disturbance.

In the development of the control system, the lambda sensor delay has not been considered so, future works could investigate the impact of the retarded measurement in the control performance. Furthermore, it could be useful to investigate the operation of the controller under full load conditions, in which the exhaust gas temperature raises up significantly and it could be detrimental for the materials of the exhaust system, on one hand and, on the other hand, for knock occurrence. So, to cool down the combustion temperature fuel enrichment is put in practice and, therefore the target lambda value changes.

Bibliography

1. Siviero, C., Scattolini, R., Gelmetti, A., Poggio, L. & Serra, G. Analysis & Validation of Mean Value Models for SI IC-Engines. IFAC Proc. Vol. 28, 1–6 (1995).

2. Hendricks, E. et al. Modelling of the intake manifold filling dynamics. SAE Tech. Pap. 105, 122–146 (1996).

3. Acquati, F., Battarola, L., Scattolini, R. & Siviero, C. An Intake Manifold Model for Spark Ignition Engines. IFAC Proc. Vol. 29, 7945–7950 (1996).

4. Lauber, J., Guerra, T. M. & Dambrine, M. Air-fuel ratio control in a gasoline engine. Int. J. Syst. Sci. 42, 277–286 (2011).

5. Chalepoudis, I. PID control of Lambda in internal combustion engines. (2019).

6. Guzzella, L., Simons, M. & Geering, H. P. Feedback linearizing air/fuel-ratio controller. Control Eng. Pract. 5, 1101–1105 (1997).

7. Li, X. & Yurkovich, S. I-C engine air/fuel ratio prediction and control using discrete-time nonlinear adaptive techniques. Proc. Am. Control Conf. 1, 212–216 (1999).

8. David Powell, J., Fekete, N. P. & Chang, C. F. Observer-Based Air-Fuel Ratio Control. IEEE Control Syst. 18, 72–83 (1998).

9. Motorsport, Bosch. "Lambda Sensor LSU 4.9." (2016).

10. Motorsport, Bosch. "Barometric pressure sensor for engine management systems SMD288".

11. Thomas Leroy, Jonathan Chauvin, Gue´nae¨l Le Solliec and Gilles Corde, Air Path Estimation for a Turbocharged SI Engine with Variable Valve Timing. New York City, USA, July 11-13, 2007

Appendix

As mentioned in Chapter 2, a general MVEM for SI engines includes 3 nonlinear differential equations and, in this section, there will be shown the parameters used to define completely these equations.

Figure 40 1275cc British Leyland engine

Intake manifold air mass flow sub-system

For the intake manifold the parameters utilized in the equations are referring to the 1275cc British Leyland engine and are the following:

- $T_{man} = 308\ K$
- $V = 614.6 * 10^{-6}\ m^3$
- $V_d = 1.275\ litres$
- $R = 0.287 \frac{kJ}{kg\ K}$
- $C_t = 0.83$
- $D = 5\ cm$
- $p_{amb} = 1.013\ bar$
- $T_{amb} = 297\ K$
- $\alpha_0 = 5.4\ deg$
- $k = 1.4$

As for the volumetric efficiency, a black box approach has been adopted by Siviero et al. [1], where the dependence of η_v upon N and p is assumed to be given by the following model:

$$\eta_v = \eta_{vn0} + \eta_{vn1}N + \eta_{vn2}N^2 + \eta_{vp1}p$$

where,

$$\eta_{vn0} = 0.133$$

$$\eta_{vn1} = 0.391 * 10^{-3} \tag{27}$$

$$\eta_{vn2} = -0.0636 * 10^{-6}$$

$$\eta_{vp1} = 0.202 * 10^{-5}$$

Fuel vapor and fuel film subsystem

Regarding the model for the fuel dynamics, to estimate the vaporization constant and the fuel fraction parameters Lauber et al. [4] has been considered as main reference. The 2 parameters have been estimated depending on the engine rotational speed, as shown below.

$$\tau_f = \sigma_3 N^{-\sigma_4}$$
$$\chi = \sigma_5 N + \sigma_6$$

where,

$$\sigma_3 = 1.67$$

$$\sigma_4 = 0.65 \tag{28}$$

$$\sigma_5 = 9.6 * 10^{-5}$$

$$\sigma_6 = 0.7236$$

Linearisation procedure

To put in practise the control algorithm, it is necessary to perform a proper linearisation procedure to end up with the A matrix, the B_1 matrix and the B_2 matrix. In the following is reported the complete calculus of those matrices.

<u>A MATRIX</u>

$$A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix}$$

$$A_{11} = \left(\frac{\partial f_1}{\partial \dot{m_{ff}}}\right)\bigg|_{ref} = -\frac{1}{\tau_{ref}\left(N_{ref}\right)}$$

$$A_{12} = \left(\frac{\partial f_1}{\partial \lambda_{cyl}}\right)\bigg|_{ref} = 0$$

$$A_{13} = \left(\frac{\partial f_1}{\partial p}\right)\bigg|_{ref} = 0$$

$$A_{21} = \left(\frac{\partial f_2}{\partial \dot{m_{ff}}}\right)\bigg|_{ref} = -\frac{N_{ref}}{30}\left(\frac{1}{\lambda_s}\frac{N_{ref}V_d\eta_{v_{ref}}\left(N_{ref}, p_{ref}\right)p_{ref}}{120RT_{man}}\right)\frac{1}{\left(\left(1-\chi_{ref}\left(N_{ref}\right)\right)\dot{m_{fi_{ref}}} + \dot{m_{ff_{ref}}}\right)^2}$$

$$A_{22} = \left(\frac{\partial f_2}{\partial \lambda_{cyl}}\right)\bigg|_{ref} = -\frac{N_{ref}}{30}$$

$$A_{23} = \left(\frac{\partial f_2}{\partial p}\right)\bigg|_{ref} = \frac{N_{ref}}{30}\left(\frac{1}{\lambda_s}\frac{N_{ref}V_d\eta_{v_{ref}}\left(N_{ref}, p_{ref}\right)}{120RT_{man}\left(\left(1-\chi_{ref}\left(N_{ref}\right)\right)\dot{m_{fi_{ref}}} + \dot{m_{ff_{ref}}}\right)}\right)$$

$$A_{31} = \left(\frac{\partial f_3}{\partial \dot{m_{ff}}}\right)\bigg|_{ref} = 0$$

$$A_{32} = \left(\frac{\partial f_3}{\partial \lambda_{cyl}}\right)\bigg|_{ref} = 0$$

$$A_{33} = \left(\frac{\partial f_3}{\partial p}\right)\bigg|_{ref} = -\frac{N_{ref}V_d\eta_{v_{ref}}\left(N_{ref}, p_{ref}\right)}{120V} + \frac{RT_{man}}{V}C_t\frac{\pi}{4}D^2\frac{p_{amb}}{\sqrt{RT_{amb}}}\beta_{1_{ref}}\left(\alpha_{ref}\right)\left(\frac{\partial \beta_2}{\partial p}\right)\bigg|_{ref}$$

Since, $\frac{p_{ref}}{p_{amb}} < \left(\frac{2}{k+1}\right)^{\frac{k}{k-1}}, \left(\frac{\partial \beta_2}{\partial p}\right)\Big|_{ref} = 0$. Putting together all these equations, the A matrix is then set and reported in Chapter 2.

B1 MATRIX

$$B_1 = \begin{bmatrix} B_{1_{11}} & B_{1_{12}} \\ B_{1_{21}} & B_{1_{22}} \\ B_{1_{31}} & B_{1_{32}} \end{bmatrix}$$

$$B_{1_{11}} = \left(\frac{\partial f_1}{\partial \dot{m_{fi}}}\right)\bigg|_{ref} = \frac{\chi_{ref}\left(N_{ref}\right)}{\tau_{ref}\left(N_{ref}\right)}$$

$$B_{1_{12}} = \left(\frac{\partial f_1}{\partial \alpha}\right)\bigg|_{ref} = 0$$

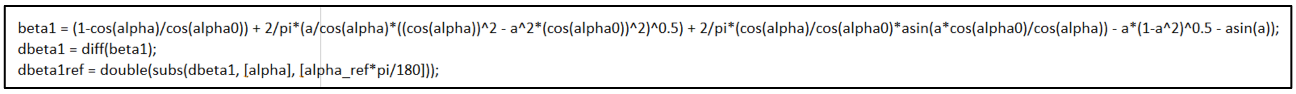
$$B_{1_{21}} = \left(\frac{\partial f_2}{\partial \dot{m_{f_\iota}}}\right)\bigg|_{ref} = -\frac{N_{ref}}{30}\left(\frac{1}{\lambda_s}\frac{N_{ref}V_d\eta_{v_{ref}}(N_{ref},p_{ref})p_{ref}}{120RT_{man}}\right)\frac{1-\chi_{ref}}{\left(\left(1-\chi_{ref}(N_{ref})\right)m_{f\dot{\iota}_{ref}} + m_{f\dot{f}_{ref}}\right)^2}$$

$$B_{1_{22}} = \left(\frac{\partial f_2}{\partial \alpha}\right)\bigg|_{ref} = 0$$

$$B_{1_{31}} = \left(\frac{\partial f_3}{\partial \dot{m_{f_\iota}}}\right)\bigg|_{ref} = 0$$

$$B_{1_{32}} = \left(\frac{\partial f_3}{\partial \alpha}\right)\bigg|_{ref} = \frac{RT_{man}}{V}C_t\frac{\pi}{4}D^2\frac{p_{amb}}{\sqrt{RT_{amb}}}\beta_{2_{ref}}(p_{ref})\left(\frac{\partial \beta_1}{\partial \alpha}\right)\bigg|_{ref}$$

$\left(\frac{\partial \beta_1}{\partial \alpha}\right)\Big|_{ref}$ has been computed through the MATLAB symbolic computation, as shown in Figure 41.



```
beta1 = (1-cos(alpha)/cos(alpha0)) + 2/pi*(a/cos(alpha)*((cos(alpha))^2 - a^2*(cos(alpha0))^2)^0.5) + 2/pi*(cos(alpha)/cos(alpha0)*asin(a*cos(alpha0)/cos(alpha)) - a*(1-a^2)^0.5 - asin(a));
dbeta1 = diff(beta1);
dbeta1ref = double(subs(dbeta1, [alpha], [alpha_ref*pi/180]));
```

Figure 41 MATLAB script for calculating the β_1 derivative with respect to the throttle position angle

B_2 MATRIX

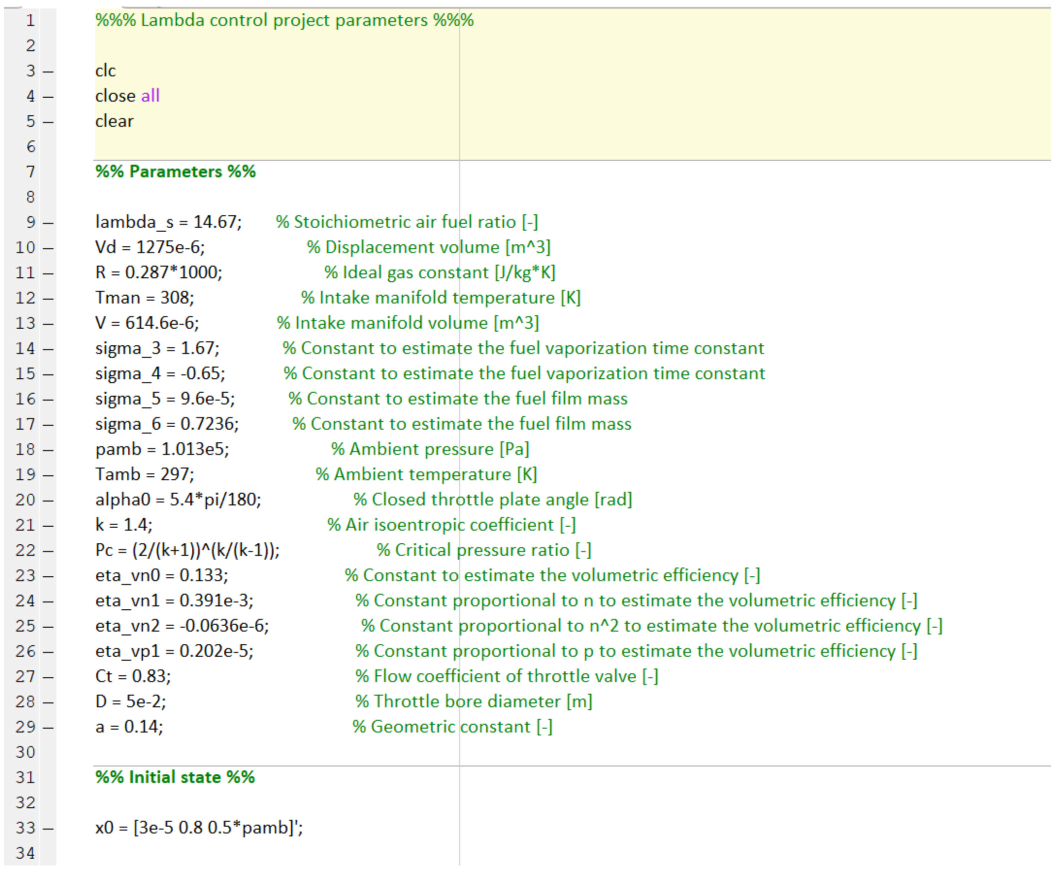
$$B_2 = \begin{bmatrix} B_{2_{11}} \\ B_{2_{21}} \\ B_{2_{31}} \end{bmatrix}$$

$$B_{2_{11}} = \left(\frac{\partial f_1}{\partial N}\right)\bigg|_{ref} = 0$$

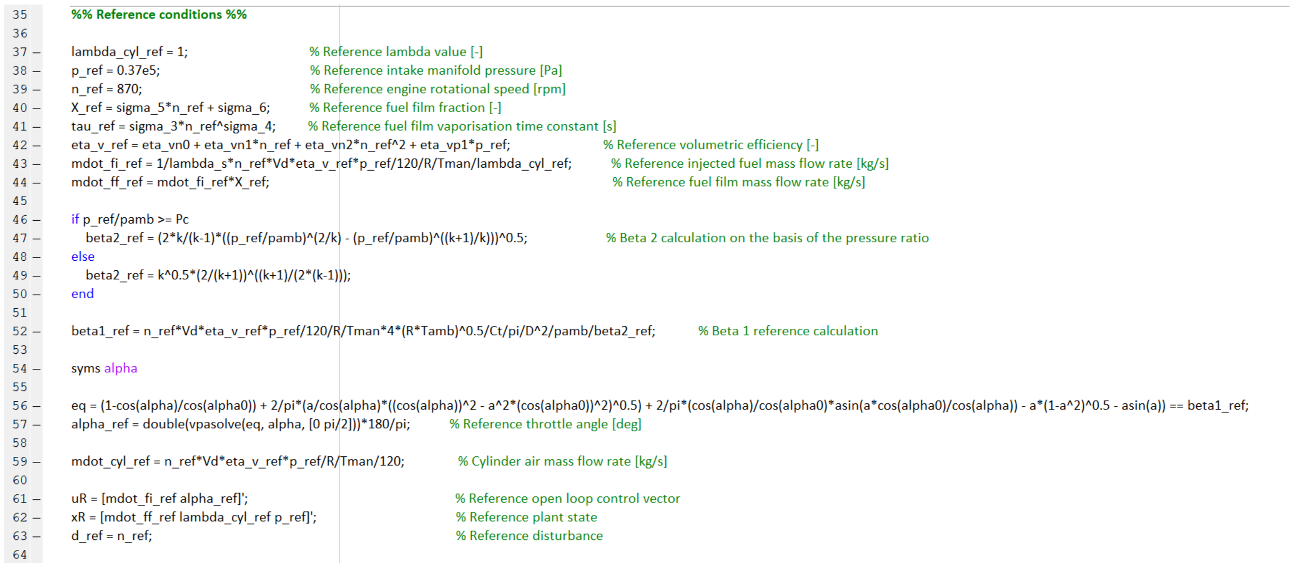
$$B_{2_{21}} = \left(\frac{\partial f_2}{\partial N}\right)\bigg|_{ref} = \frac{N_{ref}}{30}\left(\frac{1}{\lambda_s}\frac{N_{ref}V_d p_{ref}}{120RT_{man}\left(\left(1-\chi_{ref}(N_{ref})\right)m_{f\dot{\iota}_{ref}} + m_{f\dot{f}_{ref}}\right)}\right)$$

$$B_{2_{31}} = \left(\frac{\partial f_3}{\partial N}\right)\bigg|_{ref} = -\frac{V_d\eta_{v_{ref}}(N_{ref},p_{ref})p_{ref}}{120V}$$

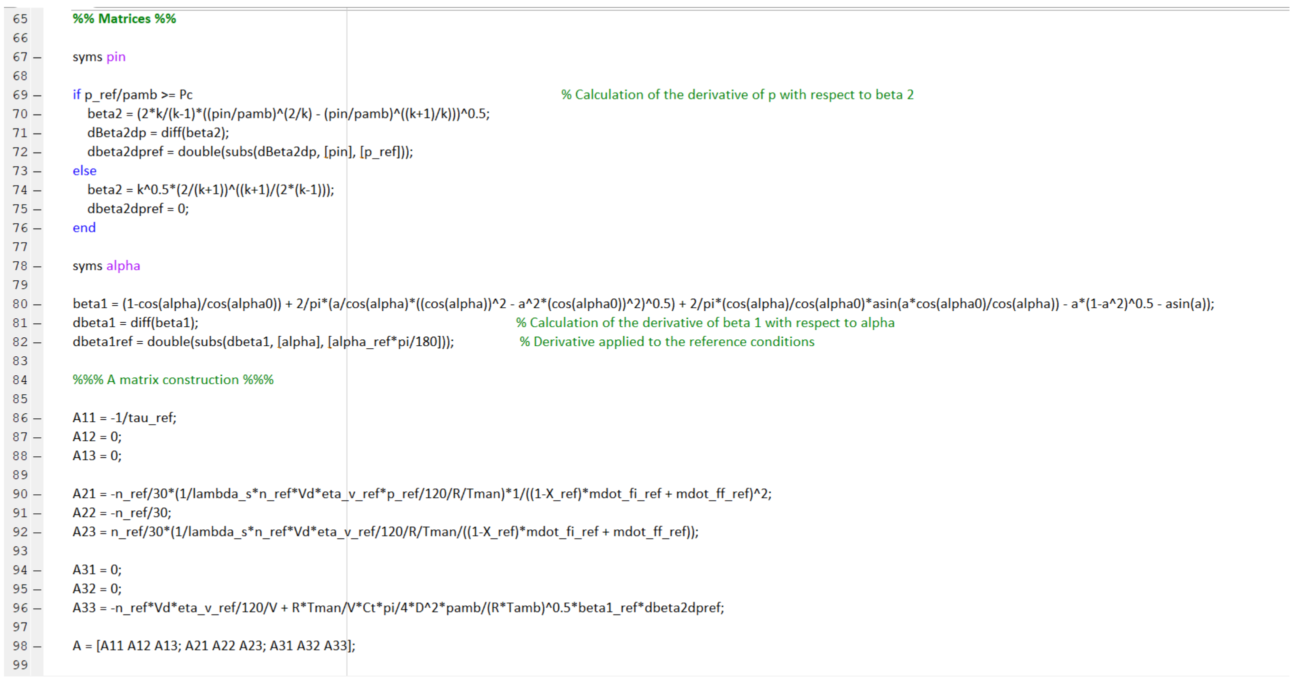
MATLAB complete script



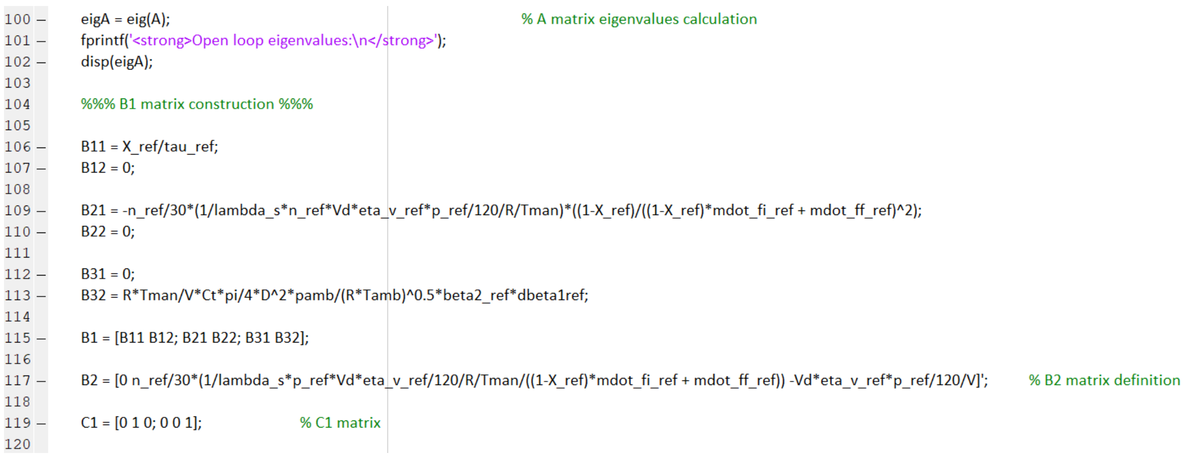
```
%%% Lambda control project parameters %%%

clc
close all
clear

%% Parameters %%

lambda_s = 14.67;     % Stoichiometric air fuel ratio [-]
Vd = 1275e-6;             % Displacement volume [m^3]
R = 0.287*1000;            % Ideal gas constant [J/kg*K]
Tman = 308;             % Intake manifold temperature [K]
V = 614.6e-6;           % Intake manifold volume [m^3]
sigma_3 = 1.67;         % Constant to estimate the fuel vaporization time constant
sigma_4 = -0.65;        % Constant to estimate the fuel vaporization time constant
sigma_5 = 9.6e-5;       % Constant to estimate the fuel film mass
sigma_6 = 0.7236;       % Constant to estimate the fuel film mass
pamb = 1.013e5;             % Ambient pressure [Pa]
Tamb = 297;               % Ambient temperature [K]
alpha0 = 5.4*pi/180;          % Closed throttle plate angle [rad]
k = 1.4;                    % Air isoentropic coefficient [-]
Pc = (2/(k+1))^(k/(k-1));         % Critical pressure ratio [-]
eta_vn0 = 0.133;              % Constant to estimate the volumetric efficiency [-]
eta_vn1 = 0.391e-3;             % Constant proportional to n to estimate the volumetric efficiency [-]
eta_vn2 = -0.0636e-6;            % Constant proportional to n^2 to estimate the volumetric efficiency [-]
eta_vp1 = 0.202e-5;             % Constant proportional to p to estimate the volumetric efficiency [-]
Ct = 0.83;                      % Flow coefficient of throttle valve [-]
D = 5e-2;                       % Throttle bore diameter [m]
a = 0.14;                      % Geometric constant [-]

%% Initial state %%

x0 = [3e-5 0.8 0.5*pamb]';

```



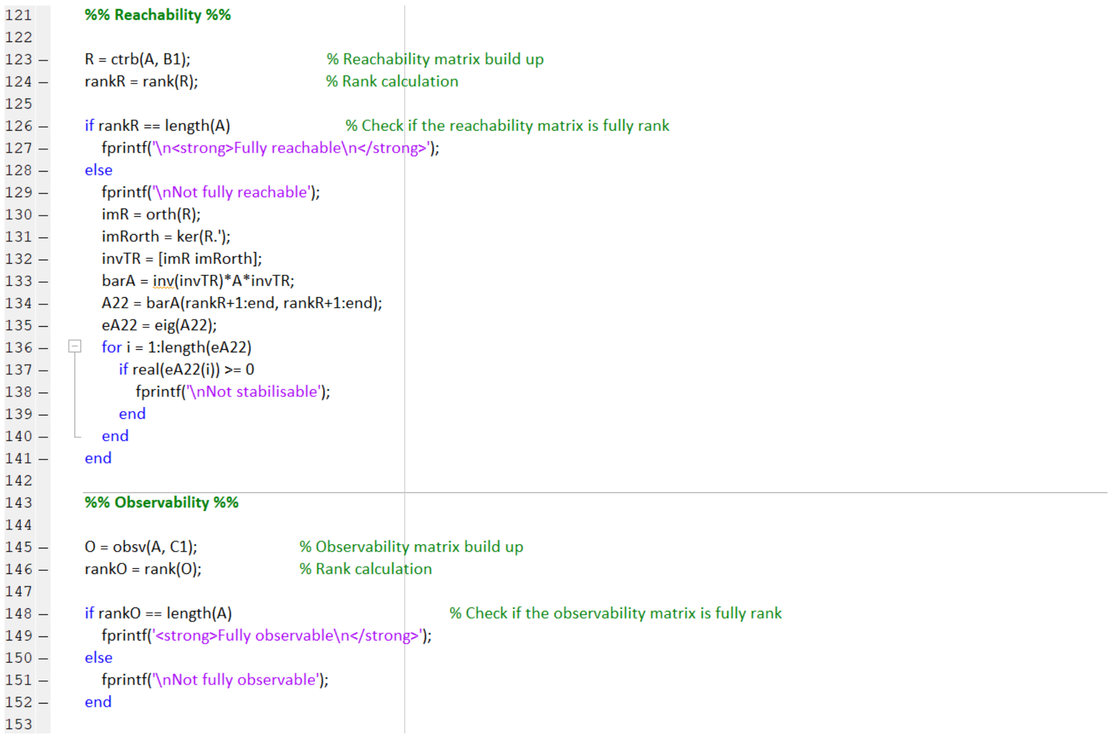
```
%% Reference conditions %%

lambda_cyl_ref = 1;                       % Reference lambda value [-]
p_ref = 0.37e5;                           % Reference intake manifold pressure [Pa]
n_ref = 870;                              % Reference engine rotational speed [rpm]
X_ref = sigma_5*n_ref + sigma_6;          % Reference fuel film fraction [-]
tau_ref = sigma_3*n_ref^sigma_4;         % Reference fuel film vaporisation time constant [s]
eta_v_ref = eta_vn0 + eta_vn1*n_ref + eta_vn2*n_ref^2 + eta_vp1*p_ref;              % Reference volumetric efficiency [-]
mdot_fi_ref = 1/lambda_s*n_ref*Vd*eta_v_ref*p_ref/120/R/Tman/lambda_cyl_ref;         % Reference injected fuel mass flow rate [kg/s]
mdot_ff_ref = mdot_fi_ref*X_ref;                                                     % Reference fuel film mass flow rate [kg/s]

if p_ref/pamb >= Pc
   beta2_ref = (2*k/(k-1)*((p_ref/pamb)^(2/k) - (p_ref/pamb)^((k+1)/k)))^0.5;                % Beta 2 calculation on the basis of the pressure ratio
else
   beta2_ref = k^0.5*(2/(k+1))^((k+1)/(2*(k-1)));
end

beta1_ref = n_ref*Vd*eta_v_ref*p_ref/120/R/Tman*4*(R*Tamb)^0.5/Ct/pi/D^2/pamb/beta2_ref;          % Beta 1 reference calculation

syms alpha

eq = (1-cos(alpha)/cos(alpha0)) + 2/pi*(a/cos(alpha)*((cos(alpha))^2 - a^2*(cos(alpha0))^2)^0.5) + 2/pi*(cos(alpha)/cos(alpha0)*asin(a*cos(alpha0)/cos(alpha)) - a*(1-a^2)^0.5 - asin(a)) == beta1_ref;
alpha_ref = double(vpasolve(eq, alpha, [0 pi/2]))*180/pi;          % Reference throttle angle [deg]

mdot_cyl_ref = n_ref*Vd*eta_v_ref*p_ref/R/Tman/120;            % Cylinder air mass flow rate [kg/s]

uR = [mdot_fi_ref alpha_ref]';                                 % Reference open loop control vector
xR = [mdot_ff_ref lambda_cyl_ref p_ref]';                      % Reference plant state
d_ref = n_ref;                                                 % Reference disturbance

```



```matlab
%% Matrices %%

syms pin

if p_ref/pamb >= Pc                                                  % Calculation of the derivative of p with respect to beta 2
    beta2 = (2*k/(k-1)*((pin/pamb)^(2/k) - (pin/pamb)^((k+1)/k)))^0.5;
    dBeta2dp = diff(beta2);
    dbeta2dpref = double(subs(dBeta2dp, [pin], [p_ref]));
else
    beta2 = k^0.5*(2/(k+1))^((k+1)/(2*(k-1)));
    dbeta2dpref = 0;
end

syms alpha

beta1 = (1-cos(alpha)/cos(alpha0)) + 2/pi*(a/cos(alpha)*((cos(alpha))^2 - a^2*(cos(alpha0))^2)^0.5) + 2/pi*(cos(alpha)/cos(alpha0)*asin(a*cos(alpha0)/cos(alpha)) - a*(1-a^2)^0.5 - asin(a));
dbeta1 = diff(beta1);                                            % Calculation of the derivative of beta 1 with respect to alpha
dbeta1ref = double(subs(dbeta1, [alpha], [alpha_ref*pi/180]));             % Derivative applied to the reference conditions

%%% A matrix construction %%%

A11 = -1/tau_ref;
A12 = 0;
A13 = 0;

A21 = -n_ref/30*(1/lambda_s*n_ref*Vd*eta_v_ref*p_ref/120/R/Tman)*1/((1-X_ref)*mdot_fi_ref + mdot_ff_ref)^2;
A22 = -n_ref/30;
A23 = n_ref/30*(1/lambda_s*n_ref*Vd*eta_v_ref/120/R/Tman/((1-X_ref)*mdot_fi_ref + mdot_ff_ref));

A31 = 0;
A32 = 0;
A33 = -n_ref*Vd*eta_v_ref/120/V + R*Tman/V*Ct*pi/4*D^2*pamb/(R*Tamb)^0.5*beta1_ref*dbeta2dpref;

A = [A11 A12 A13; A21 A22 A23; A31 A32 A33];

```



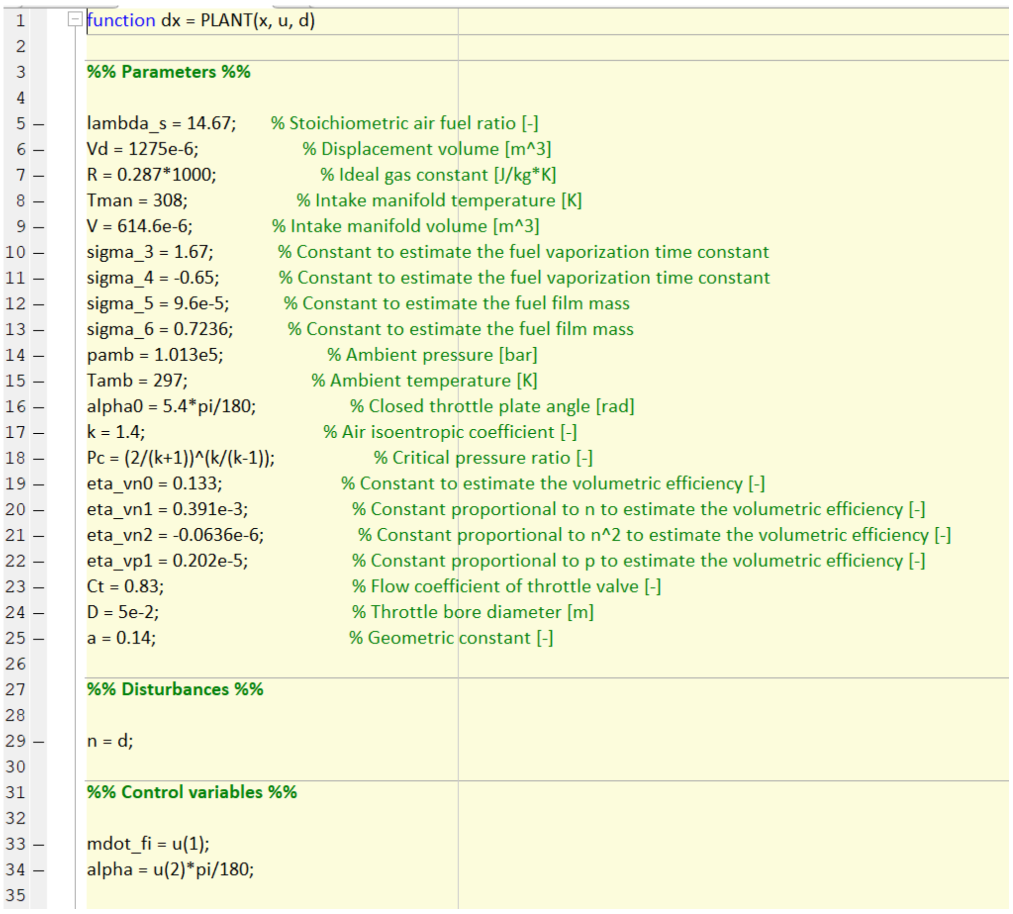
```matlab
eigA = eig(A);                                          % A matrix eigenvalues calculation
fprintf('<strong>Open loop eigenvalues:\n</strong>');
disp(eigA);

%%% B1 matrix construction %%%

B11 = X_ref/tau_ref;
B12 = 0;

B21 = -n_ref/30*(1/lambda_s*n_ref*Vd*eta_v_ref*p_ref/120/R/Tman)*((1-X_ref)/((1-X_ref)*mdot_fi_ref + mdot_ff_ref)^2);
B22 = 0;

B31 = 0;
B32 = R*Tman/V*Ct*pi/4*D^2*pamb/(R*Tamb)^0.5*beta2_ref*dbeta1ref;

B1 = [B11 B12; B21 B22; B31 B32];

B2 = [0 n_ref/30*(1/lambda_s*p_ref*Vd*eta_v_ref/120/R/Tman/((1-X_ref)*mdot_fi_ref + mdot_ff_ref)) -Vd*eta_v_ref*p_ref/120/V]';         % B2 matrix definition

C1 = [0 1 0; 0 0 1];                     % C1 matrix

```



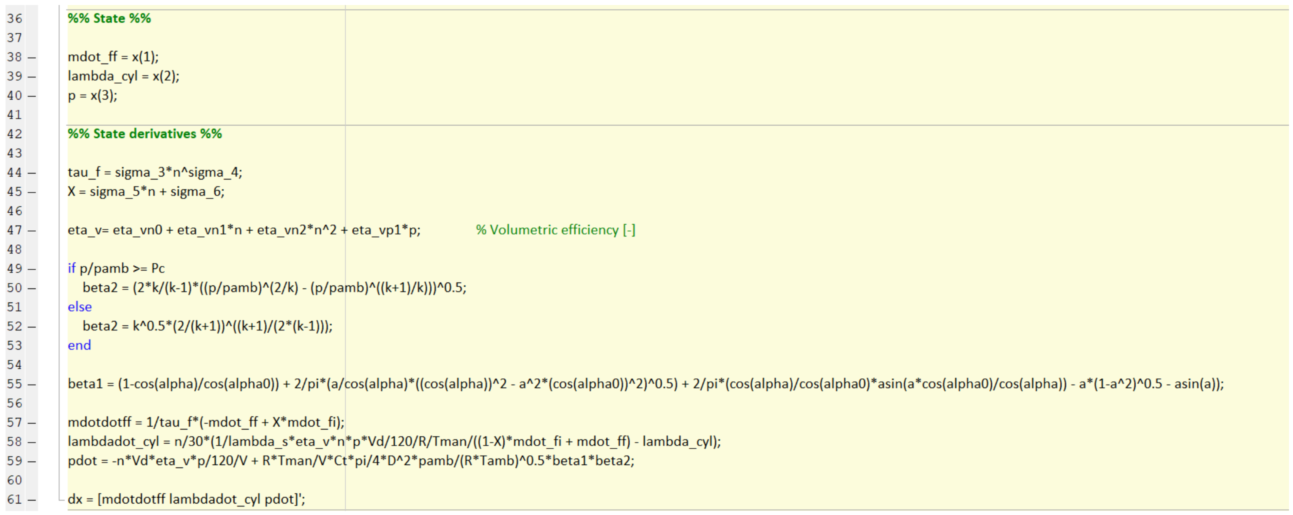
```matlab
%% Reachability %%

R = ctrb(A, B1);                          % Reachability matrix build up
rankR = rank(R);                          % Rank calculation

if rankR == length(A)                      % Check if the reachability matrix is fully rank
    fprintf('\n<strong>Fully reachable\n</strong>');
else
    fprintf('\nNot fully reachable');
    imR = orth(R);
    imRorth = ker(R.');
    invTR = [imR imRorth];
    barA = inv(invTR)*A*invTR;
    A22 = barA(rankR+1:end, rankR+1:end);
    eA22 = eig(A22);
    for i = 1:length(eA22)
        if real(eA22(i)) >= 0
            fprintf('\nNot stabilisable');
        end
    end
end

%% Observability %%

O = obsv(A, C1);                        % Observability matrix build up
rankO = rank(O);                        % Rank calculation

if rankO == length(A)                                    % Check if the observability matrix is fully rank
    fprintf('<strong>Fully observable\n</strong>');
else
    fprintf('\nNot fully observable');
end

```

```matlab
%% Optimal control %%

%%% Closed loop optimal control %%%

q = [3e-2 0; 0 1e-6];                    % Parameters for controlling lambda and intake pressure errors
R_contr = [9e3 0; 0 2e5];                % Parameters for controlling lambda and intake pressure errors

C2 = [0 1 0; 0 0 1];                     % C2 matrix related to the error definitions

%% Integral control action %%

Ae = [A zeros(3,2); C2 zeros(2,2)];          % Extended plant build up
Be = [B1; zeros(2,2)];
C = [1 0; 0 1];
Ce = [C2 zeros(2,2); zeros(2,3) C];

Q = [4.5e2 0; 0 1e-2];                   % Parameters for controlling lambda and intake pressure errors (extended plant)

Qe = [q zeros(2,2); zeros(2,2) Q];       % Extended plant cost function matrix

[Ke, Se, ee] = lqr(Ae, Be, Ce.'*Qe*Ce, R_contr, 0);        % LQR command to solve the Riccati equation and define the optimal Kr matrix
Kre = -Ke;

fprintf('\n<strong>Ke matrix:\n</strong>');
disp(Kre)

fprintf('<strong>Extended plant closed loop eigenvalues:\n</strong>');
disp(ee);

%%% Non resonance condition verification %%%

M = [A B1; C2 zeros(2,2)];

if rank(M) == length(M)                            % Check if the M matrix is fully rank to apply the integral action strategy
   fprintf('<strong>Non resonance condition is verified\n</strong>');
else
   fprintf('<strong>Non resonance condition is not verified\n</strong>');
end

%% Optimal observer %%

G = eye(3);               % Definition of the G, D, H matrices to build up the state space model for the observer design
D = zeros(2,2);
H = zeros(2,3);

sys = ss(A, [B1 G], C1, [D H]);          % State space construction

qd = [9e-3 0 0; 0 3e0 0; 0 0 1e6];       % Parameters of the cost function of the dual state to design the optimal observer
rd = [8e0 0; 0 3e0];                     % Parameters of the cost function of the dual state to design the optimal observer

[kest, L, P] = kalman(sys, qd, rd, 0);           % Kalman command which returns the optimal observability matrix

fprintf('\n<strong>Ko matrix:\n</strong>');
disp(L);

fprintf('<strong>Obsever eigenvalues\n</strong>');
disp(eig(A-L*C1));                                         % Observer eigenvalues calculation
```

Plant script



```matlab
function dx = PLANT(x, u, d)

%% Parameters %%

lambda_s = 14.67;     % Stoichiometric air fuel ratio [-]
Vd = 1275e-6;              % Displacement volume [m^3]
R = 0.287*1000;              % Ideal gas constant [J/kg*K]
Tman = 308;                % Intake manifold temperature [K]
V = 614.6e-6;         % Intake manifold volume [m^3]
sigma_3 = 1.67;        % Constant to estimate the fuel vaporization time constant
sigma_4 = -0.65;       % Constant to estimate the fuel vaporization time constant
sigma_5 = 9.6e-5;       % Constant to estimate the fuel film mass
sigma_6 = 0.7236;        % Constant to estimate the fuel film mass
pamb = 1.013e5;                % Ambient pressure [bar]
Tamb = 297;                 % Ambient temperature [K]
alpha0 = 5.4*pi/180;              % Closed throttle plate angle [rad]
k = 1.4;                     % Air isoentropic coefficient [-]
Pc = (2/(k+1))^(k/(k-1));              % Critical pressure ratio [-]
eta_vn0 = 0.133;               % Constant to estimate the volumetric efficiency [-]
eta_vn1 = 0.391e-3;              % Constant proportional to n to estimate the volumetric efficiency [-]
eta_vn2 = -0.0636e-6;             % Constant proportional to n^2 to estimate the volumetric efficiency [-]
eta_vp1 = 0.202e-5;              % Constant proportional to p to estimate the volumetric efficiency [-]
Ct = 0.83;                       % Flow coefficient of throttle valve [-]
D = 5e-2;                        % Throttle bore diameter [m]
a = 0.14;                        % Geometric constant [-]

%% Disturbances %%

n = d;

%% Control variables %%

mdot_fi = u(1);
alpha = u(2)*pi/180;

```



```matlab
%% State %%

mdot_ff = x(1);
lambda_cyl = x(2);
p = x(3);

%% State derivatives %%

tau_f = sigma_3*n^sigma_4;
X = sigma_5*n + sigma_6;

eta_v= eta_vn0 + eta_vn1*n + eta_vn2*n^2 + eta_vp1*p;           % Volumetric efficiency [-]

if p/pamb >= Pc
   beta2 = (2*k/(k-1)*((p/pamb)^(2/k) - (p/pamb)^((k+1)/k)))^0.5;
else
   beta2 = k^0.5*(2/(k+1))^((k+1)/(2*(k-1)));
end

beta1 = (1-cos(alpha)/cos(alpha0)) + 2/pi*(a/cos(alpha)*((cos(alpha))^2 - a^2*(cos(alpha0))^2)^0.5) + 2/pi*(cos(alpha)/cos(alpha0)*asin(a*cos(alpha0)/cos(alpha)) - a*(1-a^2)^0.5 - asin(a));

mdotdotff = 1/tau_f*(-mdot_ff + X*mdot_fi);
lambdadot_cyl = n/30*(1/lambda_s*eta_v*n*p*Vd/120/R/Tman/((1-X)*mdot_fi + mdot_ff) - lambda_cyl);
pdot = -n*Vd*eta_v*p/120/V + R*Tman/V*Ct*pi/4*D^2*pamb/(R*Tamb)^0.5*beta1*beta2;

dx = [mdotdotff lambdadot_cyl pdot]';
```