Linköpings tekniska högskola, ISY, Fordonssystem

Project compendium Modelling and Control of Engines and Drivelines TSFS09

Linköping, October 24, 2022

Contents

Part I

Project 1

Chapter 1
General description

Objective

The idea behind this project is to provide an insight into how you can build a realistic model of a vehicle, and to provide an understanding of how torque is generated inside an engine. The goal is that you in your project groups should, as far as possible, both put up a theoretical model together and implement it in Simulink. In addition, the model is both adapted and verified against real measurement data. Therefore you should set up appropriate measurements to do this, and conduct the measurements on your own. Finally, the developed model will be used in a computer experiment for a real driving cycle which examines the engine emissions. This project has great potential to bring an understanding of how real modeling work goes on and is therefore an important part of the course.

It is good to emphasize that:
Important: Project 1 takes a lot of time!

The report should be submitted as a *.pdf* to Lisam for project 1 according to the course homepage. The file should have the following name format according to your student LiU-ID's: *LiUID1_LiUID2_Project1.pdf*

1.1 The different project parts:

The different parts in this project are:

1. Project 1A :
    - Divide a vehicle model into suitable sub-components and construct models

for the components or complete the models which are incomplete.

- Identify all unknown parameters of the constructed models, and design experiments to determine these parameters. Moreover, experiments are designed to validate the different sub-models. The experiments should be designed so that they can be implemented in engine laboratory of Vehicular Systems.
- Perform measurements under the supervision of an assistant. For more information on the engine laboratory and various types of measurements see chapter 2.3.

2. Project 1B :

   - Analyse how engine operates and generates work, according to the measured data (projects 1B).
   - Implement each previously developed sub-model in Simulink as well as: i) adapt the unknown parameters to the measured data and ii) verify each sub-model against measured data.

3. Project 1C :

   - Connect the various sub-model in Simulink to a large coherent model of a complete vehicle, and also perform some computer experiments (simulation) to see how it manages regulatory requirements for emissions (EURO-3 and EURO-4).

First hand in, Chapter 2 of the report template.

Second hand in, Chapter 3 of the report template, do not forget to attach your Matlab code!

Third hand in, Chapter 4-7 of the report template, do not forget to attach the Matlab-Simulink files!

Chapter 2

Project 1A: Engine Measurements

2.1 Objective

The objective of project 1A is to illustrate how measurements in a modern engine test cell works and generate data which can be used in the evaluation stage (project 1B).

Project description

Project 1A, with the preceding assignments, correspond to item 1 in chapter 1.1. We remind you that it is you in the project team that designs the measurements in the sub-projects, whose purpose is to generate data needed to estimate the unknown parameters and/or verify the accuracy of sub-models in project 1B.

2.2 Examination Requirements

1. A **model description** of a mean value model for a turbocharged engine. Note that the turbocharger is not modeled in project 1. The description shall show subsystems and model equations specifying what are inputs, outputs and parameters in the models that are used in the part 1B&C of the project. If the first hand in for model description is not approved, the new version of the report addressing the comments must be handed together with the first version.

2. Together with the engine model a written **experiment plan** should be handed in and approved before performing engine measurements. The experiment plan should describe which measurements shall be performed and specify what is

going to be measured, and what signals should we vary during the measurement (control signals). The purpose of the measurements is to provide data for the identification of the parameters in the mean value model, and to show what happens inside the cylinders. A brief description of how the various test series shall be used for the evaluation shall be given. You need not to specify the engine operating points, but only what to measure and how. If the experiment plan is not approved on your first attempt, the original experimental plan must be submitted again along with the additional information requested.

3. To pass part 1A you require an approved **model description** and **experiment plan** before measurement session, and **active presence** in the measurement session. For the written parts (model description and experiment plan) the report template available on the course website should be used.

2.3 Description of the research laboratory

Vehicular Systems Research Laboratory consists of an engine test cell and a control room. In the engine test cell there are two engines and two electric brakes. The brakes are often used to generate a loading torque, but they can also do the reverse, ie drive engines. The engines, brakes and sensors are controlled and monitored from the control room.

An introduction to the lab is given below. At the engine measurement session there will be assistants avaiable for your guidance, therefore you do not need to know how the lab works in detail.

2.3.1 Engine test cell

At present, the engine test cell includes two turbocharged gasoline engines. One of them is prepared with extra sensors, and one is a typical production setup. You are going to model the engine with extra sensors. Many of the sensors (the ones used in production engines) where installed on the engines when they were delivered, since they are used by the engine control system. The extra sensors gives you the opportunity to retrive data that makes it possible to isolate the components that are needed to build the engine model.

The two engines are coupled to one brake respectively. The brakes are big, controllable, electric machines, which make it possible to perform repeated tests with great accuracy. Brakes are normally controlled so that they (and therefore engine) keep a constant speed, while the engines are controlled to maintain a constant throttle opening.

Since the engines are in an engine test cell, it is possible to send control signals which a regular control system normally does not produce. It is for example possible to make the amount of injected fuel vary independent of air mass flow. We can also change the throttle position without affecting the engine speed.

Something that is important to note in terms of engine control, is that the accelerator pedal is not directly connected to the throttle plate. The throttle angle can not easily be expressed as a function of accelerator pedal position. The accelerator pedal usually corresponds to a desired torque, which translates into a reference air mass flow. The throttle angle is controlled so that the actual air mass flow rate becomes equal to the reference air mass flow.

Another important thing to note is that the air/fuel ratio measured with the continuous sensor is not reliable for $\lambda < 0.95$. The reason is that λ sensor measures the oxygen content in exhaust gas, and for fuel rich mixtures, there exists little amount of oxygen in the exhaust gas to be measured.

2.3.2 Control room

In the control room there are computers, measurement instruments and control devices to the brakes. One of the computers is connected to the engine control system, through which it can read the measurement signals and actuate some control signals to the engine. Another computer controls a high speed measurement system, and the brakes (when necessary).

Measurement data collection is handled via Matlab. Matlab takes in measured data and saves it in a .mat file. Manual readings of measurements can in some cases be appropriate.

The measurements performed in the lab can be divided into three different categories:

- Stationary (engine map)
- Dynamic
- Crankshaft based

Physical measurements will also be needed, callipers are used to determine *compressor wheel diameter needed in project 2*. The three categories of measurements can be briefly described as follows:

Stationary The engine is controlled to a desired operating point, where it is given time to stabilize, i.e the dynamics will have time to subside. Thereafter, data collection is performed for some time (usually a couple of seconds). What is returned is the average value over this period of time. The result is then one measurement per operating point and sensor.

Usually various operating points are measured in series after each other. Going from a working point to the next, allowing the engine to stabilize, measuring, moving on to the next, allowing the engine to stabilize, and so on. This is called making an **enginemap**.

Dynamic Data collection is performed in continuous time, both the signal values and time are recorded. The result is thus a vector of time resoluted data for each signal that is recorded.

Crankshaft based Crankshaft measurement is used when you want to see what happens during a cycle. An ordinary resolution is 1 sample/degree.

2.4 Project tips

This section contains suggestions for some of the model components that is to be implemented.

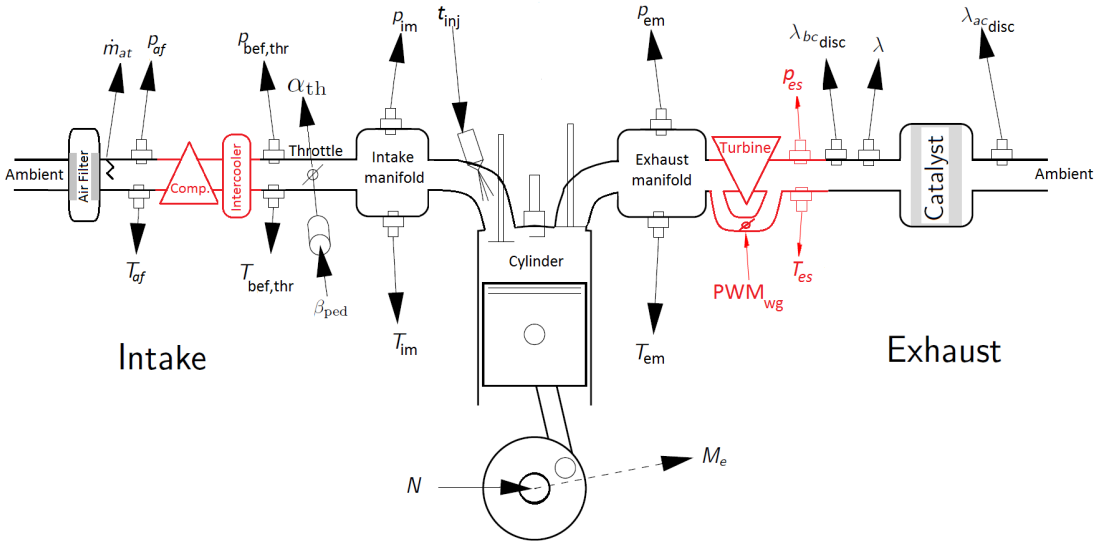

Figure 2.1 *An overview of the air path with signals and components to be modeled in project 1 (shown in black) and project 2 (shown in red+black)*

For your convenience, a complete model description and experimental plan for the accelerator pedal, and most of model description for the throttle are already included in the report template. You can use this example as a template when you write model description and experimental plan for other components.You should have replaced every ???? in the report template before submitting your report.

2.4.1 Engine Map

When estimating parameters data from engine maps are often used. It is therfore recomended to include a section where it is explaned what an engine map is and how it is created. When using data fomr the map you can then simply refer to this section.

2.4.2 Suggested throttle model

The accelerator pedal is not connected directly to the throttle, but its position is translated in the control system into a reference angle for the throttle. A controller makes

the throttle angle follow the reference. The details of the throttle control will not be included in this project, but assume there is a working controller. If we see the pedal position as the reference angle, the dynamics to the throttle angle, including the controller, can be approximated by a first order system according to

$$\dot{\alpha}_{thr} = \frac{1}{\tau_{thr}}(\beta_{ped} - \alpha_{thr}) \tag{2.1}$$

2.4.3 Suggested throttle airflow model

Components with significant pressure drop and a small flow area, such as the throttle in most operating points, can often be described well by the equations for compressible flow. These are given by

$$\dot{m}_{at} = \frac{p_{amb}}{\sqrt{RT_{amb}}} A_{eff}(\alpha_{thr})\Psi(\Pi), \qquad \Pi = \frac{p_{im}}{p_{amb}} \tag{2.2}$$

$$\Psi = \sqrt{\frac{2\gamma}{\gamma - 1}\left(\Pi_{lim}^{\frac{2}{\gamma}} - \Pi_{lim}^{\frac{\gamma+1}{\gamma}}\right)}, \qquad \Pi_{lim} = max\left(\Pi, \left(\frac{2}{\gamma+1}\right)^{\frac{\gamma}{\gamma-1}}\right) \tag{2.3}$$

in which $A_{eff} = A(\alpha_{thr})C_D$ describes the effective flow area as a function of throttle angle. This can be parameterized in different ways, but here we suggest a second degree polynomial in α_{thr} according to (2.4). See the course book for other options.

$$A_{eff}(\alpha_{thr}) = a_0 + a_1\alpha_{thr} + a_2\alpha_{thr}^2 \tag{2.4}$$

This model gives a good description of the mass flow rate for pressure ratios that are not too close to 1, so do only use points with $\Pi < 0.8$ in the estimation of these model parameters.

2.4.4 Intake Manifold

The intake manifold pressure is modeled as a volume where the pressure dynamic behavior is calculated from the inflow and outflow.

2.4.5 Suggestions for implementation of the volumetric efficiency

There are many ways to implement the volumetric efficiency. One way is to use a table in Simulink, *Look-Up Table (2-D)*. However, some of the drawbacks of using look-up tables (maps) are that these require a lot of memory, especially if they depend on many variables (have many dimensions), and that the simulation could end up outside the map. Another option, which is what we recommend, is that you adapt a function. A number of variants are described in the textbook, and the one suggested is:

$$\eta_{vol}(N, p_{im}) = c_0 + c_1\sqrt{p_{im}} + c_2\sqrt{N}$$

2.4.6 Cylinders

In the cylinder model there are four sub-models; one sub-model for the generated torque, one sub-model for the cylinder temperature, one sub-model for the air mass-flow into the cylinders and another one for the air/fuel ratio inside the cylinder (λ_{cyl})

2.4.7 Suggested torque model

The engine output torque can be modeled in many different ways. Equation 7.55 in the book is recommended for torque modeling, and is based on efficiency, since this may be the most physical model. Read chapter 7.9.[1,2,3] before doing this part. Assume that the ignition is optimal i.e. $\eta_{ign} = 1$. For friction modeling use the Heywood model on page 191 but you have to find the model constants.

When estimating the paramterers for this model the enigne map could be used. However, since the engine is using VVT and the ignition is not always optimal many of the points in the map are not suitable for this model. Therfore other measurements will be used. The measurements that will be used are cylinder pressure trace for some opeating points and engine friction measurements. The friction measurements has been created by dragging the engine and measuring the torque required to drive it at different speeds. In project 1A you do not have to fully specifi how the unknown parameters will be estimated, it is enough to specify what measurements will be used for each parameter.

2.4.8 Temperatures in the engine

Temperatures on the intake side of the engine can be assumed constant in each part and estimated as the mean of the corresponding temperature in the engine map. The ambient temperature is set equal to the temperature after the air filter. Note that you should only use these mean values when using the model, when you estimate other parameters in the model you should use the entire map.

The temperature in the exhaust manifold is assumed to be equal to the cylinder out temperature which is mainly a function of the mass flow. Hence, this temperature can be modeled as a linear function of the square root of mass flow out of the engine

$$T_e = T_0 + k\sqrt{\dot{m}_{exh}}$$

where T_0 and k are model parameters.

2.4.9 Exhaust system and exhaust backpressure

The exhaust system is comprised of a model for exhaust manifold pressure and also the light-off time for the catalyst. The exhaust manifold pressure is modeled as a volume where the pressure is calculated from the inflow and outflow, just as the intake

manifold. The output mass flow from the exhaust manifold can be modeled as an incompressible turbulent restriction, see chapter 7.2 in the book.

$$\Delta p = C_2 \frac{RT_{us}}{p_{us}} \dot{m}^2 \text{(see the course book).}$$

Considering Figure 2.1, the fact that the Enginemap data is obtained from a turbocharged engine and that a naturally aspirated engine is going to be modeled, which one of P_{em} or P_{es} signals in the Enginemap should be used in this task and why? This is an important question so you shold make sure that you understand it.

2.4.10 Fuel injectors

When modeling the fuel injectors, use the equation for injected mass per stroke as base model for parameter estimation and validation,

$$m_f = c_{fi}(t_{inj} - t_0)$$

which is (7.28) in the course book, where ρ_{fuel} and Δp_{inj} have been assumed constant and lumped together with C_0 to form c_{fi}. Also U_{batt} is assumed constant in the project.

2.4.11 Lambda sensor

Models for continuous and discrete lambda sensors, before catalyst are required. Assume that the time delay, $\tau_{d_{bc}}$, to the lambda sensor is constant over the whole engine operating region. In addition to the pure time delay, the exhaust manifold and sensor adds a certain dynamic. Assume that this can be described by a first order system, with time constant τ_λ that should be estimated. Note that this time constant is a lumped model of the dynamics i the exhaust and the sensor dynamics.

2.4.12 Complete engine diagram

In the Section **Complete engine**, you shold include a figure of a Simulink diagram with blocks for each of the components in the complete engine model. You do not have to implement any of your models, so the Simulink blocks can be empty. However you shold be able to clearly see how the signals (the physical quantities) flow between the various sub-components. Having the Simulink diagram in this figure right will later, in project 1C, help you when implementing the models in Simulink.

Chapter 3

Project 1B: Analysis and Validation

3.1 Objective

There are two objectives of Project 1B. The first is to provide an understanding of how an engine generates mechanical work through the analysis of torque, work, and air and fuel supply. The second objective is to select, parametrize and validate models of each of the components of the engine.

3.2 Examination requirements

To pass, all tasks in Section 3.3 shall be addressed and the solutions properly described in your report. Remember to explain, not just describe, the content of all of the figures.

The project report should include:

- Comment on all validation figures and discuss the correlation with measurement data, and argue why it is good enough.
- Discuss the choices of parameters. In cases where you can choose from estimating your own parameter values or using values already given, the choice shall be discussed
- Attach the **matlab code** that performs all calculations and creates all required figures in section 3.3 in a **separate .m file**.
- After validation of models, all estimated parameters in different sub models must be presented in a table. This makes it easier for us to correct your reports, and for you to reach and find different values during model implementation in Simulink.

If the report is not approved on the first hand-in, each subsequent hand-in of a new version should also have all previous, commented, versions attached.

3.3 Project components

In the analysis you will look at data from two different operating points. How to get data and necessary project files is described among the Matlab tips in Section 3.6 below. Download and use especially the available skeleton file `Project1b.m`. We strongly recommend you to use the skeleton file to collect all the Matlab commands you use in the analysis.

Observe: It is the models (equations and parameter) that should be validated, not your Simulink implementations of these models.

Data package: The data package to be used during this course is named `ProjektFilerTSFS09.zip` and is available at the project part of the course homepage:
`www.fs.isy.liu.se/Edu/Courses/TSFS09/Projekt/`
The data package is password protected with the password shown below:

```
Password: TSFS09_HT17
```

In the subsequent modelling and validation, you will use the static measurement data and the transient data that you collected during the measurements in Module 1A.

3.3.1 Analysis

In this part you will use measured cylinder pressure curves from the SVC engine, along with the aforementioned engine map. These are available in the data package `ProjektFilerTSFS09.zip`, files to be used are named `pCylHigh.mat` and `pCylLow.mat`.

Tips: The required engine parameters are included in the two .mat files.

Tips: Use the .m file **Project1b.m** for this part of Project 1b.

1. Draw PV diagrams for the engine in the two operating points. Compare the PV diagrams for the two operating points with each other and explain the differences. What is the main difference between the two operating points? (what is it that is high and low?)

   Calculate the work performed during one cycle at each operating point, along with the average torque, by determining the enclosed area. Do not forget to report if the calculated numbers are for one cylinder or for the entire engine. Compare the calculated average torque with the measured torque. What are the main causes of the difference?

To integrate the enclosed area you can use the `trapz` function, according to the Matlab tips in section 3.6.

2. This task deals with the ideal Otto cycle

   (a) In Matlab, plot and present an ideal Otto cycle for one of the measuring points above. Use the equations for isentropic compression, combustion under constant volume, and isentropic expansion. Assume that the temperature at point 1 is equal to the inlet temperature. Start with using the inlet pressure as the pressure at point 1 and adjust this so that the measured and ideal compression pressure tracks coincide. Assume that all injected gasoline burns instantaneously and that there are no residual gases. Also assume that all gases are ideal.

   (b) Plot the ideal cycle and a measured cycle in the same figure and explain the differences. Which peak pressure is obtained from the Otto cycle?

   (c) Adjust the amount of energy drawn from the fuel for the ideal cycle (by adjusting q_{LHV}) so that the enclosed areas become the same. Present this new figure and answer the following questions:

      i. How much (percentage) of the energy from the fuel is not transformed into mechanical work?

      ii. What do you compensate for when you reduce the amount of added energy? Most of the fuel has, after all, burned when the cycle is complete, so where has the rest of the energy gone?

3. Do the same as in 2, but now first remove the pumping work from the total work of the measured cycle (this can be done in different ways, think of which is best). How can the result from this be used in your models?

4. Compile the results from task 1-3 and make a table or diagram (or something similar) that shows where the energy in the fuel goes.

5. Use the function `sfc_plot.m` to make a contour plot of specific fuel consumption as a function of torque and speed. Use `help sfc_plot` to see what arguments and what units the function requires. Which operating point would you have chosen if you were to design a generator to be used for stationary operation with a constant speed and fixed load? Do not forget to explain the figure! (Use data from `EngineMap.mat` during this task)

3.3.2 Modelling

In this part you should select, parametrize and validate all necessary sub-models for the VEP-engine. All the required data for the VEP engine are available in the data package at the course homepage. In some cases the validations will also require the models to be implemented in Simulink, in the other cases the Simulink implementations may wait until Module 1C. Remember that the validation is of the models and parameters, not of your implementations.

Tips: The required engine parameters are included in file `EngineMapTSFS09.mat`

Tips: Use the .m file **Init_Project1.m** for this part of Project 1b.

To pass, all sub-models must be validated. Validation of a model means showing that the model and the data are consistent by comparing the output from the models (static or simulated) to the data from the measurements. In cases where a model can not be validated against measured data, the implementation of the model should be verified by simple experiments.

Identify the inputs, outputs and parameters for all sub-models. Adapt the model parameters for the constituent sub-models to the measurement data and show how well the model and measurement data agree. Describe how the validation has been performed for each sub-model and the parameter values you use. **Use the tips given in Sections 3.5 and 3.6**

To get started, we suggest that you begin with the following:

1. Estimate the parameters c_{fi} and t_0 in the linear model $m_f = c_{fi}(t_{inj} - t_0)$, which describes how the fuel per injection depends on the injection time (t_{inj}). Use the data from all operating points when estimating parameters, see Matlab tips in Section 3.6 below for tips on the least squares method. Plot both the modelled and the measured values for the total injected fuel quantity per engine operation cycle as a function of injection time. Plot the measured values with markers instead of a line. Compare the model with the measured data and report the comparison.
2. Make a 3D plot of the engine volumetric efficiency as a function of engine speed and intake pressure, and state the maximum value of the volumetric efficiency. You may use the `nvol_plot.m` described under Matlab tips in section 3.6.
3. The Matlab script `eng_val.m` produces a range of figures that might or might not be suitable for the validation of the models. The figures are intended as inspiration for you, so to help you come up with figure types and/or layouts that are suitable for the different static model validations.

3.4 Proposed preparatory tasks

We strongly recomend that you have studied, solved and feel confident about the solutions for the following preparatory tasks **before** the computer lab session. You will need all the results and our experience is that solving these issues beforehand does save time in the end and makes it a lot easier to fully utilize the assistance available at the computer sessions. This preparatory tasks do not need to be explained in the report, but be aware that they can provide valuable clues as you e.g. explain figures.

1. Derive a model for cylinder volume $V(\theta)$ as a function of crank angle θ.
2. Suggest a method to calculate the engine's average output torque from the enclosed area in the pV-diagram.

3. Describe an ideal Otto cycle by specifying how the pressures and temperatures in the cycle's 4 points can be calculated. Calculate the chemical energy based on measured air mass flow.

4. What is the specific fuel consumption and how can it be calculated from measurable variables in steady state conditions (assuming that λ is known)?

5. What does the volumetric efficiency mean and how can it be calculated from measurable variables in steady state conditions?

3.5 Project tips

Below are some tips on modeling of certain components and how, for example, the filling ratio and the torque can be modeled using polynomials. For tips on which other subsystems to include, see the headings in the report template.

3.5.1 Validation of the static and dynamic models

To make sure that your sub-models works as intended, these should be validated individually or in as small groups as possible. Do not confuse the validations of your models to the (also necessary) verifications of your Simulink implementations.

For static models, validation is preferably done with static plots/maps. If the model depends solely on one variable, this can be used as x-axis, and the modeled and measured signals can be compared on the y-axis. For models that depend on several variables, one can plot the modeled signal against the measured signal for each point. If the model is correct, all points will lie on a straight line with a 1/1-slope passing through the origin, since this means that the modeled and measured signal are equal. You should also check your Simulink implementation of the model, and the easiest way is by testing a number of constant inputs and ensure that you get the correct outputs.

For dynamic models you need to make simulations and compare the behavior to the measurement data. One good option is to use measurement data (using `From Workspace`) as inputs to the simulation model, and plot the simulated output together with the measured output. When you compare the outputs, remember that you are validating the dynamic behavior i.e. rise times and time delays. The static levels are static and should be validated as suggested above. Use the Simulink blocks `From Workspace` and `To Workspace` to send data to and from you models. Use the Matlab help for these blocks for more detailed instructions.

3.6 Matlab tips

Below is a collection of Matlab tips that may be helpful during this part of the project.

- Measurement data is given in the data package named `ProjektFilerTSFS09.zip` that is downloaded through the project part of the course homepage. `www.fs.isy.liu.se/Edu/Courses/TSFS09/Projekt/` As mentioned earlier, the data package is password protected and the password is shown below:

  ```
  Password: TSFS09_HT17
  ```

- In the data package you can find the skeleton files `Project1b.m` and `Init_Project1.m`. It is highly recommended that you collect all the Matlab commands for the exercises in Section 3.3.1 in `Project1b.m`, and the Matlab commands for the exercises in Section 3.3.2 in `Init_Project1.m`.

- When commands should be executed, run the entire file or use the Matlab *cell-mode* to execute part of the code. A cell begins with `%%` and with `Ctrl+Enter` the code is executed until the next `%%` or to the end of the file. This is so that you easily be able to rehearse and make changes to the command sequences.

- Measurement data is stored in structures (`struct`). A structure is a variable which consists of a number of fields. A field may for example contain a scalar, a vector or a string (text). If a structure is named `X`, access to the field `y` is gained by writing `X.y`. Which measurement signals `X` contains can be seen by writing `X` in the Matlab-prompt.

- To make elements-wise operations when working with vectors or matrices a "." is put before the operator (e.g .* for multiplication ./ for division etc). If only the operator is used, matrix operation will be performed.

- Two functions, `nvol_plot` and `sfc_plot`, are given to facilitate the drawing of volumetric efficiency and specific fuel consumption. Write `help` followed by the function name in the Matlab-prompt to see details on how to call the functions. The functions are retrieved in the same manner as the skeleton file from the website and added together with the other project files.

- When calculating the energy during a cycle the integral $\oint p dv$ should be calculated. The integral value can be approximated with e.g trapezoidal method (`trapz` i Matlab).

- It is often good to use the least squares method to determine the best values for model parameters. By writing `x=A\B` in Matlab the line that best describes a variety of points in the least squares sense is obtained, meaning the x which minimizes $\|Ax - B\|_2$. The sought vector x should be a vector consisting of the unknown parameters, or functions of these.

  Suppose that an affine model is sought under $y = k(z + m)$ and that data for z and y are the vectors `z` and `y` in Matlab. The parameters to be estimated are k and m. One can then do as the following example shows:

```
% On the form Ax=B (with x = [k k*m])
A = [z,ones(size(y))];
B = y;
% Perhaps remove measurements
index = ... %index for good measurements, see tips below
A = A(index,:);
B = B(index);
% solve with with the least squares method
x = A\B;
% x = [k k*m]
k = x(1);
m = x(2)/k;
```

- Some times the data containts unvanted entries that need to be removed before using the data, this can easily be done in matlab. For example, the following example shows how we can extract the positive values in a vector

```
y = sin(1:10) % create vector with some values
index = y >= 0; % find entries with positive values
x = y(index); % x now only contains positive values
```

- To save a figure from a Matlab plot the commands `print` or `save` can be used. You can use this for getting your script to automatically save the generated figures.

- Use the Simulink block `From Workspace` in `Simulink Library Browser` → `Sources` to get a data series from Matlab workspace to Simulink. Similarly, the block `To Workspace` in `Simulink Library Browser` → `Sinks` is used for exporting signals from Simulink to the Matlab workspace, and from there be used in the validations that should be included in the report. Read matlab help for these blocks for more detailed instructions on how they work.

Chapter 4

Project 1C: Dynamics and Emissions

4.1 Objective

The objective of project 1C is to study the dynamics and emission formation for an engine coupled to a vehicle, and to illustrate the main control functions in an engine control system. Furthermore, the project aims to give a sense of the general considerations that arise when choosing a model-structure-complexity and the choice between using given parameter values and adjusting your own.

4.2 Examination Requirements

To pass, all tasks in Section 4.3 shall be addressed and the solutions properly described in your report. The purpose of the project report is to show that you have performed and understood all components of the project. Active presence in the project is not enough to demonstrate this. Remember to explain, not just describe, the content of all of the figures.

The project report should include:

- Report on all equations, Matlab code and Simulink figures.
- Print names and units on the axes of all figures.
- An attached executable **Matlab code** in an **separate .m file** performing all parameter estimates for your engine model.
- An attached **Simulink model** which should be executable after the attached .m file has been run.

Observe: Measurement files and available files that you have not modified (`nvol_plot.m`, `init_drivecycle`, `celcEmissions` etc.) does **not** have to be attached.

4.3 Project components

The project includes implementation and simulation of a complete vehicle, the design of a fuel controller and investigating vehicle acceleration capability, fuel consumption and emissions. A Simulink project template is provided in the data package that is downloaded through the course homepage. The complete engine model and your controllers should be inserted into this frame when they are ready and validated. The frame model is executable to facilitate testing but you are expected to change the *Engine* and *ECU* $\rightarrow$ *Fuel Controller* blocks.

The major work is to implement the engine model, so do not be alarmed if the following task 1 takes a long time for you to finish.

4.3.1 Tasks

1. Implement a mean value model for the naturally aspirated engine in Simulink using the sub-models you have prepared and validated in the previous Module. Also implement the feedforward part of the fuel (or lambda) controller from the sensors available e.g. $\dot{m}_{at}$ or p_i and N. The lab assistant can give you a brief review for those who have not read about feedforward.

You should now have a working mean value model for a naturally aspirated engine that runs at $\lambda = 1$ in stationary points. Test the implementation by running a constant operating point, constant speed and throttle angle, somewhere in the middle of the engine map. Check that for example intake pressure, p_{im}, air flow $\dot{m}_{at}$, and torque, M, gives similar values as in the engine map. Also check that the engine runs at $\lambda = 1$ to confirm that the feedforward part of the lambda controller is correct.

2. Design a simple feedback for the λ-controller from a discrete λ-sensor mounted before the catalytic converter.. Make a step in accelerator position at a constant engine speed (this is most easily done with the help of I/O-blocks, see the Matlab tips in section 4.6). How fast is the step response for the torque?

3. Connect the engine model to the longitudinal vehicle model with a clutch and gearbox model (see Section 4.4.3). Turn all switches in the Driver Model block to the preset manual values (full throttle in 3rd gear) and simulate (note: make sure you have not set constant speed etc. in the I/O block). How fast does the vehicle accelerate from 70 km/h to 110 km/h in 3rd gear?

4. Plug the driver model in (see section 4.4.2) and test drive this model in the first 30 seconds of the driving cycle. **Include a figure plotting the vehicle speed vs.**

drive cycle speed, intake pressure, engine speed, engine torque and continuous λ in the cylinder. Based on the speed of the vehicle, determine if the other signals appear to be reasonable and explain why.

5. Simulate an entire cycle **and include a figure plotting the vehicle speed vs. drive cycle speed, intake pressure, engine speed, engine torque and continuous λ in the cylinder** and calculate the emissions in [g/km] for HC, CO and NO_x. Your λ-controller must at least meet the requirements given by EURO 3. There is a complete Matlab function (see Section 4.6) which, given λ calculates HC, CO and NO_x. The function also prints the emission limits of the EURO-3 and EURO-4 for this cycle.

   When you calculate the emissions you will do it for a hot and a cold engine. For the cold engine you should set the cold start time to the time you measured and for the hot engine the catalyst should ignite immediately by setting the cold start time to 0.

   The simulation model includes several different lambda signals, e.g. in the cylinder, before the catalyst, and measured lambda with both discrete and continuous sensor. Describe the lambda signal you use in the emission calculations, and why this signal is appropriate.

6. Calculate the fuel consumption , in [$dm^3/100$ km], in the European driving cycle. The density of petrol may be set to 0.75 kg/dm^3.

The project report shall report on the above points. In addition, we would like you to treat and report on the following points. We recommend that you at least carry out and report on the first of the points:

- Examine how vehicle size and engine size affects fuel consumption and emissions. Do this by e.g. reduce vehicle weight by 25%, reducing the vehicle cross area by 20% and reduce motor size by 50%. How well is this vehicle able to follow the cycle? What will be the emissions in [g/km]? What is the fuel consumption in [$dm^3/100$ km]?
- Investigate what would happen if the vehicle fuel tank is filled with a slightly different fuel. In order to simulate this, change the stoichiometric A/F_s ratio in the Fuel controller to 15.5. Test it with and without the fuel controller feedback enabled, what are the differences in emissions and fuel consumption?
- Illustrate the problem with the time delay for closed-loop control of λ.
- Illustrate how a poor control system and engine design can bring increased emissions.

4.4 Subcomponents in the vehicle model

The vehicle model is divided into a number of blocks; engine model with controllers, driver model, clutch and gearbox model, vehicle model and emissions model. All

models except the emissions model is implemented as blocks in Simulink, while the emissions model is available as an m-file that can be run after the simulation. A top view of the Simulink model is shown in Figure 4.1.

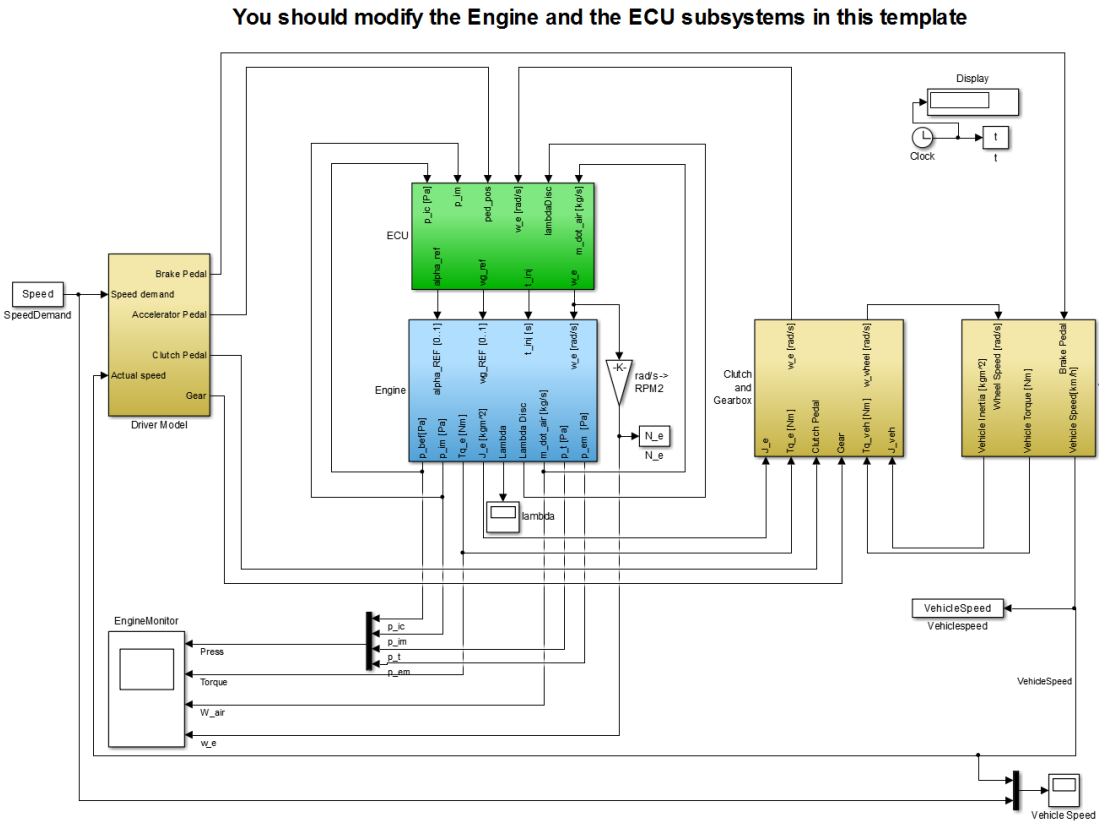

Figure 4.1 *Model for the complete vehicle. You should change the* Engine *and* ECU-*blocks*

4.4.1 Engine model with controllers

The engine model and lambda controller are not given, but should be implemented. The engine model should be of a mean-value type. The model parameters should be identified from the measurements made on the engine that the model is supposed to describe.

Figure 4.2 shows a top view of the engine and the ECU. These blocks contains some connections already at the start, but the contents are only made to make the Simulink model executable, and must therefore be replaced with ”real connections”.

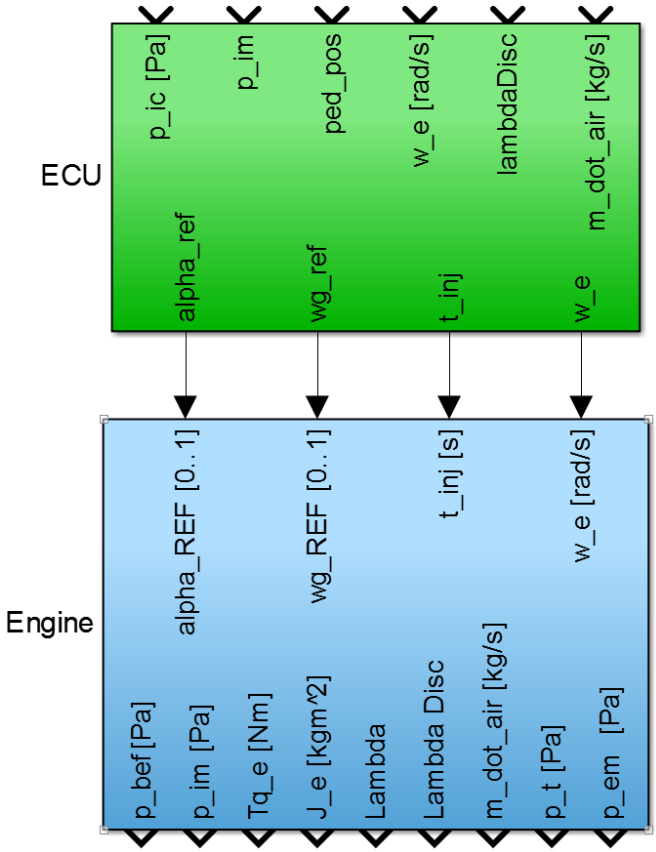

Figure 4.2 *Model of engine and ECU.*

Input signals Engine model	
alpha_ref	Accelerator pedal position reference: 0–1, with 0=idle, 1=full throttle.
wg_ref	Wastegate reference, used in Project 2.
w_e	Engine speed [rad/s]
tinj	Injection time for the fuel injectors [s]

The Boost control block is not enabled in Project 1. It will require some subsystems to be implemented once you are working with Project 2.
You are expected to implement the Fuel controller block for Project 1.

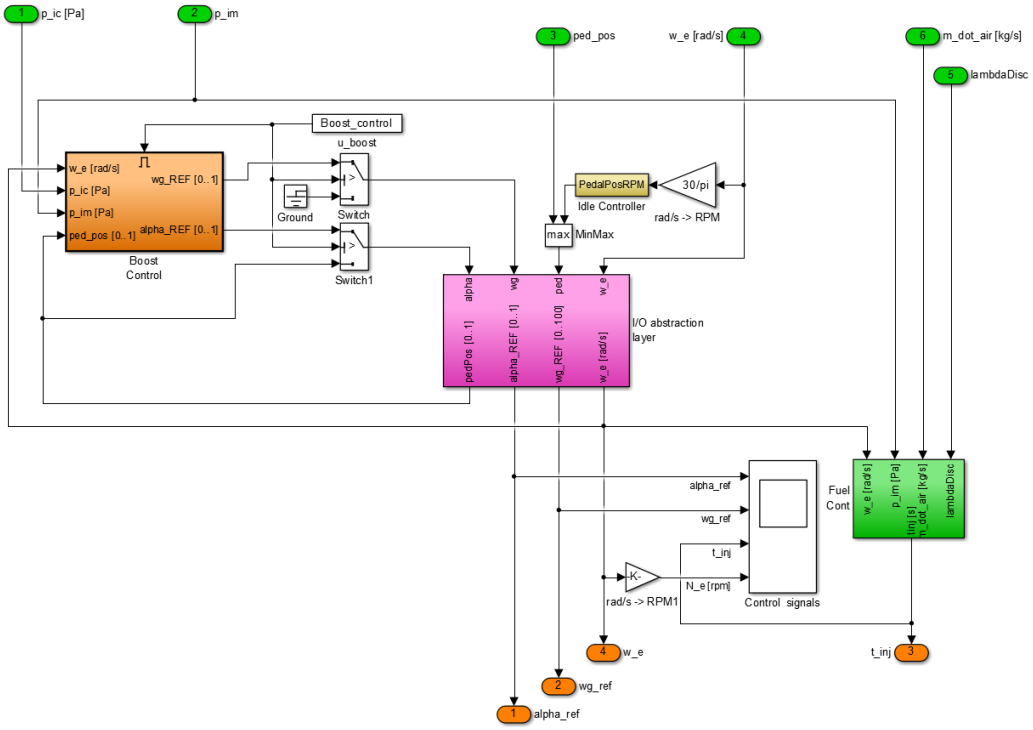

Figure 4.3 *The ECU contains idle, fuel and Boost pressure. The later will not be used until Project 2. Here is also an I/O block that you can use to manually set the input signals to the engine block. The fuel control is also a lambda controller which ensures that the engine runs at $\lambda = 1$. Before the I/O block input for throttle position is a* `max` *block that ensures that idling can only support the driver, i.e. provide extra torque but never give less torque than what the driver desires.*

Output signals Engine model	
p_bef	Pressure before the throttle [Pa]
p_im	Pressure in the intake manifold [Pa]
Tq_e	Engine torque [Nm]
J_e	Engine inertia [kgm^2]
Lambda	Normalized air-/fuel relation, used for the emission calculation
Lambda Disc	Discrete lambda signal
m_dot_air	Air mass flow past the throttle [kg/s]
p_t	Pressure after turbine [Pa]
p_em	Pressure in the exhaust manifold [Pa]

Fuel controller

The fuel controller ensures that the engine gets as much fuel as it needs by calculating how long the injectors should be open. The opening time of the injection valve determines the mass of fuel injected. The desired injected fuel mass is calculated from how much air the engine gets, either by measuring the air mass flow or by calculating it by using the volumetric efficiency (η_{vol}). To compensate for the modeling errors, etc. the fuel mass estimation is corrected by means of feedback from the lambda sensor via the lambda regulator.

Input signals Fuel and lambda controller	
w_e	Engine speed [rad/s]
p_im	Intake manifold pressure [Pa]
lambdaDisc	Discrete lambda signal
$\dot{m}$_air	Air mass flow past the throttle [kg/s]

Output signal Fuel and lambda controller	
t_inj	Injection time for the fuel injectors [s]

Input signal Idle controller	
RPM	Actual engine speed [RPM]

Output signal Idle controller	
PedalPos	Accelerator pedal position [-]

4.4.2 Driver model

A model of a driver is included, see Figure 4.4. The drivers job is to follow the velocity profile in a cycle as close as possible with the help of gas pedal, brake pedal, clutch and gear position. The outputs of the gas, brake and clutch pedals are in the range $[0, 1]$, i.e. between 0 and 1.

Input signals	
Speed Demand	Driving cycle speed profile, comes from the Matlab workspace when the script `Init_Project1.m` is run.
Actual Speed	The actual speed from the vehicle model.

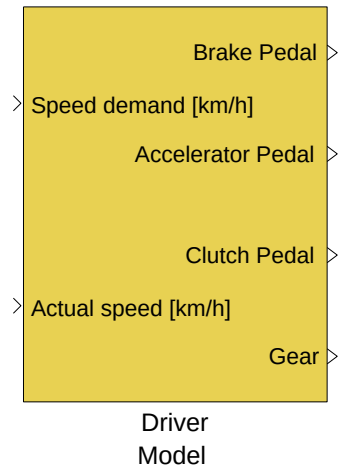

Figure 4.4 *Simulink block of the driver.*

Output signals	
Brake Pedal	Brake pedal postion: 0–1, with 0=no brake, 1=full brake. This should be connected to the vehicle model.
Accelerator Pedal	Accelerator pedal postion: 0–1, with 0=zero thottle, 1=full throttle. This should be connected to the engine model.
Clutch Pedal	Clutch signal to be connected to the gearbox. Position: 0–1, with 0=clutch fully depressed, 1=clutch fully released.
Gear	Selected gear. Should be be connected to the gearbox.

4.4.3 Clutch and gearbox model

An example of a clutch and gearbox is implemented in Simulink, as shown in Figure 4.5. The model simulates the engaging and disengaging of the transmission with a variety of gears and will be paired with your engine model, vehicle model, and a driver model to simulate the whole car. The cycle contains data about gear and clutch sequence entered for use by the clutch model.

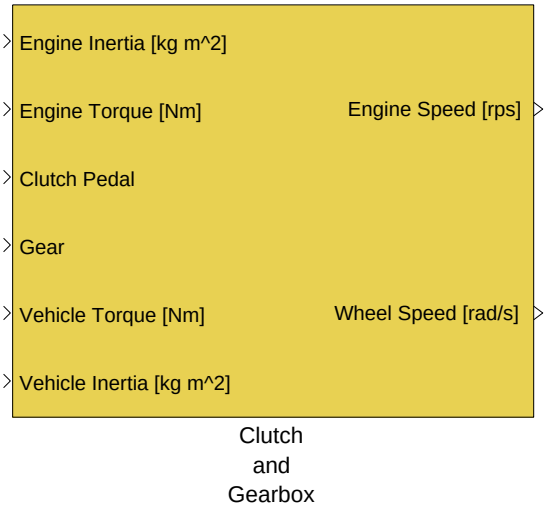

Figure 4.5 *Simulink block for clutch and gearbox*

Input signals	
Engine Inertia	Engine inertia, a constant.
Engine Torque	Engine propulsive torque from the engine model.
Clutch Pedal	Clutch pedal signal from the driver model, given by the driving cycle.
Gear	Gear position from the driver model, given by the driving cycle.
Vehicle Torque	The sum of all braking torque from the vehicle, ie aerodynamic drag, rolling resistance and driver's braking torque.
Vehicle Inertia	Vehicle weight converted via the car's wheels to a rotating inertia, a constant.

Output signals	
Engine Speed	Engine speed.
Wheel Speed	Wheel speed, corresponds to vehicle speed.

Note that some of the input signals (eg, inertia torque) actually are parameters. It stands just as (input) signals because in some cases it is a simple way, purely for implementation, to describe a parameter in Simulink.

4.4.4 Vehicle model

A model of the vehicle is implemented in Simulink, and shown in Figure 4.6. The vehicle is modeled as a mass that is accelerated by the engine (or rather the gearbox) driving torque and braked by the air and rolling resistance, including the driver brakes.

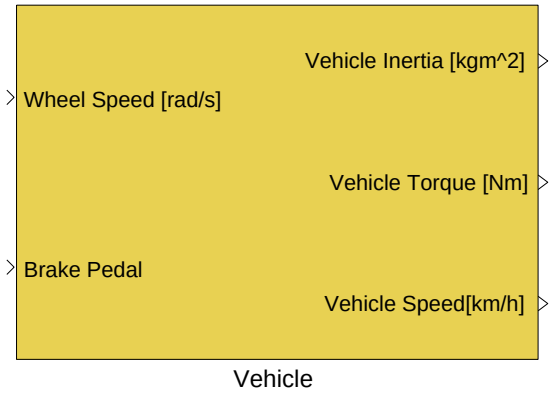

Figure 4.6 *Simulink block of the vehicle*

Input signals	
Wheel Speed	Calculated in the Clutch model and used to calculate the air and rolling resistance and the vehicle speed.
Brake Pedal	The driver's brake signal. The brake signal is translated into braking torque by amplification in the block Brake Gain in the Vehicle block.

Output signals	
Vehicle Inertia	Estimated from vehicle mass transformed via the wheels to a rotating inertia.
Vehicle Torque	Total vehicle braking torque, which is the sum of the air and rolling resistance, and braking torque.
Vehicle Speed	Wheel speed converted to vehicle speed.

4.4.5 Emissions model

After the burnt gas leaves the cylinder it passes through the exhaust pipe and the catalyst. If the catalyst is warm and the λ-value of the exhaust gas is sufficiently close to 1, the catalyst will clean the emissions to consist as much as possible, of carbon dioxide, water and nitrogen.

There are many emissions standards defined in the world. Here in Europe, these include EURO-3 and EURO-4, which are two differently stringent requirements. The table below lists the emission requirements listed as emissions in grams per kilometer driven for petrol vehicles.

	CO [g/km]	HC [g/km]	NO_x [g/km]
EURO-3	2,3	0,2	0,15
EURO-4	1,0	0,1	0,08

In this project, the vehicle shall be driven according to the operating cycle in Figure 4.7. It is implemented in `init_drivecycle.m`. Note that if the engine is started

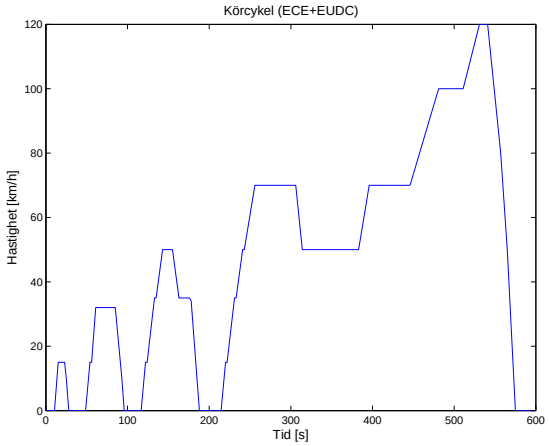

Figure 4.7 *Driving cycle (ECE+EUDC)*

cold we need to know the light-off time, i.e. the time it takes before the catalyst has reached its operating temperature. Emission calculations are performed using the `calcEmissions`.

4.5 Project tips

λ-controller When you implement the λ controller, it is important to remeber that you can not suck fuel back through the valve, i.e., you can not have negative fuel flow. Also study the course compendium for a proposal on how to implement a λ-controller.

4.6 Matlab and Simulink tips

In this section we want to provide some small tips as a guide to a solution of the project components.

- If you are unable to simulate your model, or if it takes long time, Appendix B contains some tips.
- Follow and complete each step of the project requirements, Section 4.3.1, before starting on the next step. This facilitates fault finding in the models. In project 1C, start with torque from only the engine model that was developed in the computer lesson. This corresponds to the engine block in the given Simulink structure. When the engine model is to be developed, it is easier if it is developed as a separate Simulink model.
- Finish each subcomponent before you move on to the next. Becoming finished means adapting parameters to measurement data, and to verify that the model for the component works. During the adjustment and verification phase of each component the measured data should be used as inputs (where possible). After verification the inputs shall come from the rest of the model. The blocks `Binary Switch` and `From Workspace` can be useful for switching between these two different cases.
- The models and some useful Matlab files are available in the data package on the course homepage:
  The file: `init_drivecycle.m` initiates the velocity vector, the clutch pedal position and selected gear along with `Init_Projekt1.m`. We strongly recommend that you place your commands which calculate model parameters and initializes the simulation model in `Init_Projekt1.m` and run this before each simulation.
- Measurement data from the dynamic and static experiments are available in the data package at the course homepage.
- The block "I/O abstraction layer" of the frame model, in which different signals to the engine can be controlled "manually", has been included in the model to simplify the simulation of specific driving cases or tests. The block is prepared for controlling N_e, $\alpha_{thr,ref}$, $pos_{wg,ref}$ och ped_{pos} ($pos_{wg,ref}$ will not be used until Project 2).

- Implement and evaluate the air mass flow model first, then go ahead with the fuel model and finally the torque model.
- A comparison block is available in the data package from the course website to help in the validation of dynamic sub-models.

- In Simulink you can find the block `Polynomial` which is good for evaluating polynomials.
- The simulink block `Scope` is useful for assessing the plausibility of the simulation data (do not use figures from `Scope` in the report). The downside is that they take a lot of computing power from the simulation.
- The time vector from the simulation is stored in the variable `tout`. Signals can be exported to Matlab workspace with the simulink block `To Workspace` (the time can also be included in each of these).
- Components can be grouped into `Subsystems`, by selecting a number of components, right-click and choose `Create Subsystem from Selection`. You can also insert an empty subsystem from:
  `Simulink Library Browser` → `Ports & Subsystems` → `Subsystem`.
- Signals in Simulink can be named by double-clicking on the signal and writing the desired name. To see the name of a signal named earlier in the model, double-click on the signal and type <> (this is good for seing the name of a signal that has been named outside of the subsystem).
- When interpreting a Simulink scheme the dimension of the signals can be seen by activating the `Signal Dimensions` contained in the `format` menu.
- A helpful function called `calcEmissions` has been created to calculate out the emissions and show the limits of the EURO-3 and EURO 4. Note that the function requires the light-off time.
- During the project, you will need to estimate some parameters with the least squares method. Suppose that an affine model is sought under $y = k(x + m)$ and that data for x and y are the vectors `x` and `y` in Matlab. The parameters to be estimated are k and m. One can then do as the following example shows:

```
% On the form Av=b (with v = [k k*m])
A = [x,ones(size(y))];
b = y;
% Perhaps remove measurements
index = ... %index for good measurements
A = A(index,:);
b = b(index);
% solve with with the least squares method
v = A\b;
```

```
% v = [k k*m]
k = v(1);
m = v(2)/k;
```

Part II

Project 2

Chapter 5

General description

Objective

One main objectives of this project is to give basic knowledge about turbocharging and how to model a turbo. Another is to show how, and why, downsizing and turbocharging can reduce fuel consumption and emissions compared to a naturally aspirated engine with the equal performance. The project should also give insight into the main control loops for air control of a turbocharged engine.

The project goal is to model a downsized (1.2 L) turbocharged petrol engine. A torque based control structure with coordinated control of the actuators should then be developed for the engine model. The turbocharged engine performance and emissions will be compared to the earlier projects naturally aspirated, and bigger (2.0 L) petrol engine.

The template for Project 1 is prepared with a turbocharger block. You should connect this to the other engine components and implement your submodels for the turbocharger inside it.

5.1 The different project parts:

The different parts in this project are:

1. Project 2A :
    - Estimate all parameters in the compressor , turbine and wastegate model needed to implement the turbocharged engine model in part 2B.
2. Project 2B :
    - Implement a 1.2 L turbocharged engine model based on the model you developed in Project 1, with the parameters estimated in part 2A, in the ENGINE block.
    - Implement a feed forward for the throttle control system and tune controller parameters for the feedback part of the throttle and wastegate controller in the ECU block.
    - Investigate vehicle acceleration, fuel consumption and emissions for the turbocharged engine and compare with the 2.0 L naturally aspirated engine form Project 1.

First hand in, Chapter X of the report template, do not forget to attach your Matlab code!

Second hand in, Chapter X of the report template, do not forget to attach your Matlab code!

5.2 Components in the turbocharged engine model

This section gives an overview of the components in the turbocharged engine, and how they connect to each other. It also cover measurements of turbo performance and how corrected quantities are used in the compressor and turbine maps. An overview of the signal flow in the engine model is given in Figure 5.1.

The figure shows how the submodels from Project 1 communicates with the submodels for the turbo to create a component based turbocharged engine structure. The abbreviations in the figure are AMB = ambient, C = compressor, IC = Intercooler, THR = throttle, IM = intake manifold, CYL = cylinder, EM = exhaust manifold, T = turbine, WG = wastegate, S = turbo axis, ES = exhaust system. Squares indicate that the submodel includes state variables. For example the turbine (C) has in signals from the surrounding control volumes (pressure and temperature before and pressure after) and from the block containing the state for the turbo speed (S). The turbine model calculates the mass flow that is taken from the control volume before (V_{em}) and is feed to the control volume after (V_{es}). Since also the temperature of the flow changes through the turbine, torque is produced the is feed to the turbo shaft.

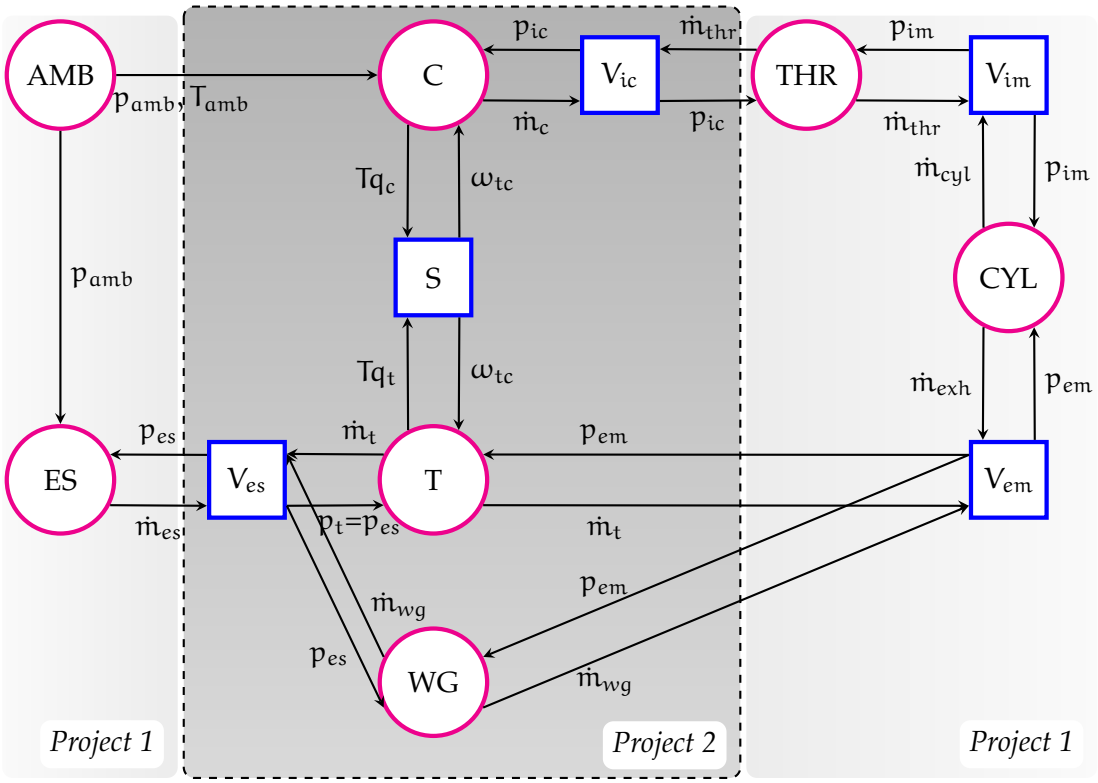

Figure 5.1 *Overview of the components of the turbo system, and how they connect to the other components of the engine model (from Project 1). The main focus of Project 2 thus lies between components that you have already built in Project 1. The only change is to the displacement volume of the engine but apart from changing that parameter value, you only have to extend the model with new components. The arrows in the figure indicates how information is passed between the component models. For example, squares in the figure indicate control volumes, they pass their internal pressure state to the surrounding components, that calculates the mass flow in and out of the control volume.*

5.2.1 Turbo maps and corrected quantities

Measurement data for the main components of the turbo (compressor/turbine) are normally collected in turbo maps. One important thing is that the data in the maps are given in corrected quantities and not actual measured values (rotational speed, mass flow). Without the correction the maps could only be applied to surrounding conditions at the time of measurement. By using corrected quantities the compressor performance can be calculated for any surrounding condition.

Figure 5.2 shows a compressor and turbine map measured in a gas stand, together with the models suggested in section 6.2.1. It can be seen that the models describe the measurement points well in the region where the turbo normally operates, but not so good for higher pressure ratios.

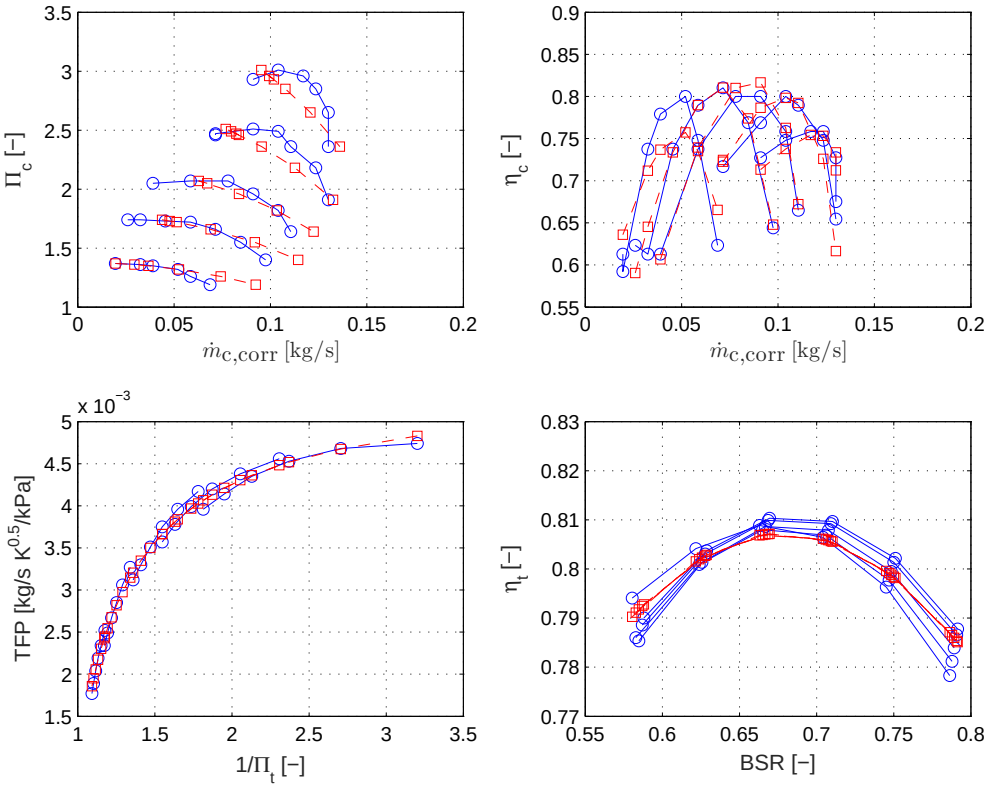

Figure 5.2 *Compressor and turbine maps in solid lines with blue circles, models fitted to the respective quantity in dash dotted lines with red squares.*

The compressor map

A compressor map describes how corrected mass flow ($\dot{m}_{c,corr}$) is connected to the pressure ratio over the compressor (Π_c) for constant corrected speed ($N_{c,corr}$), for a series of steady state measurement points. For each point in the map also the adiabatic compressor efficiency (η_c) is given. The corrected quantities for the compressor are calculated as:

$$\dot{m}_{c,corr} = \dot{m}_c \frac{\sqrt{T_{af}/T_{ref,c}}}{p_{af}/p_{ref,c}}$$

$$N_{c,corr} = \frac{N_c}{\sqrt{T_{af}/T_{ref,c}}}$$

The turbine map

For the turbine map normally the relation between the turbine flow parameter (TFP) and the turbine pressure ratio (Π_t) for constant reduce turbine speed (TSP) is given. Also the adiabatic turbine efficiency (η_t) is given for each operating point. The cor-

rected quantities for the turbine are calculated as:

$$TFP = \dot{m}_t \frac{\sqrt{T_{03}[K]}}{p_{03}[kPa]}$$
$$TSP = \frac{N_t[rpm]}{\sqrt{T_{03}[K]}}$$

OBS: The pressure correction for the turbine flow parameter TFP is normally *NOT* in the SI-unit Pa, instead kPa is used. The speed correction for the turbine speed parameter TSP is *NOT* in the SI-unit rps, instead rpm is used.

Measurement of turbo maps

Figure 5.3 shows an overview of a turbo mounted in a gas stand for mapping. Normally the compressor inlet receives preconditioned air at roughly room temperature (298K) and atmospheric pressure ($\approx$ 100kPa). The turbine is fed with pressurized air where diesel fuel has been burned. Normal turbine inlet temperature is regulated to 873K by controlling the diesel fuel pressure, and the turbine outlet normally has approximately the same pressure as the compressor inlet.

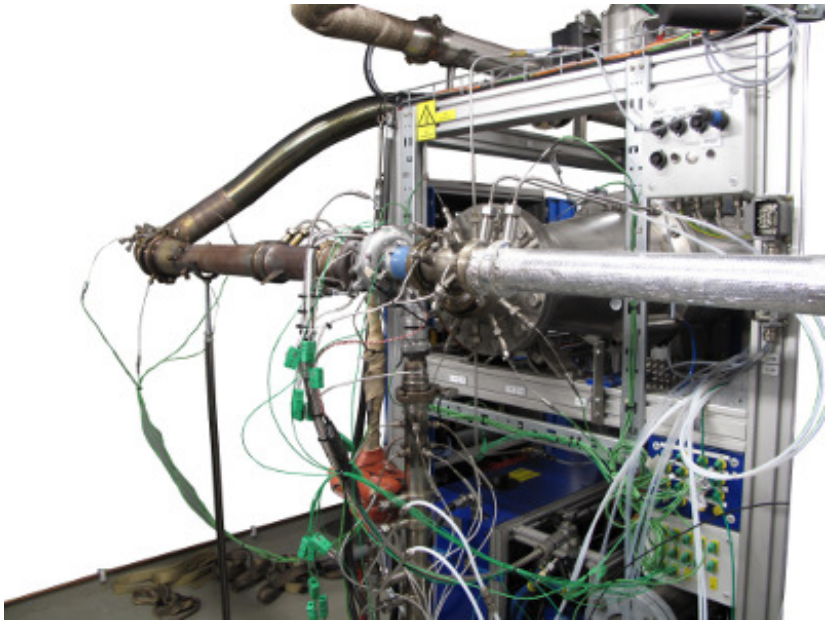

Figure 5.3 *Overview of a turbo (in the middle of the figure) mounted in a gas stand used for turbo mapping. The air to the compressor is feed in from the right in the figure in the horizontal pipe, and goes out in the bottom of the figure. The turbine gas comes from a diesel burner, and goes through the turbine and out to the left in the horizontal pipe.*

Chapter 6

Project 2A: Modeling and parameter estimation

Project goal

The goal of this part is to estimate and validate all parameters are needed for the implementation of project 2 in Simulink. The questions in Section 6.1 also aim at giving basic knowledge about turbocharging. There are five main properties which will be modeled in this chapter which are the mass flow and efficiency for the compressor and turbine, and a model for *bmep*.
The template file *project2A.m* on the course homepage should be used in this project.

Examination Requirements

In the project report you should:

1. Answer questions in section 6.1, perform the tasks in section 6.2 and 6.3.

2. Clearly present all model equations that are used, and what parameters are going to be estimated.

3. Present values of estimated parameter, and describe how they have been calculated.

4. Include validation plots where the models are compared against the measurements.

5. Comment all validation plots and discuss the agreement with measurements, and argue in which operating regions the model is good enough.

6. All plots **must**: be clear and zoomed to the regions of interest with as small blank space as possible, have proper axis names with units, legends, captions.

7. Attach executable **Matlab code** in a **separate.m file** performing all parameter estimation and plotting all validation figures in matlab. Also, the matlab codes should be included at the end of the submitted report.

8. A *.pdf* file as the report should be submitted to Lisam for project 2A according to the course homepage. The file should have the following name format according to your student LiU-ID's: *LIUID1_LIUID2_Project2A.pdf*

6.1 Tasks

1. What are the main components of a turbo and what are their purpose? Is there any limitations in the operating region of the components?

2. How are the compressor- and turbine maps measured? What is corrected quantities and why are they used? What is reference states in this context?

3. What benefits/drawbacks has a large/small naturally aspirated engine? How can turbocharging be used to remove the drawbacks? What other drawbacks is a consequence of turbo charging?

4. What extra actuators are present on a turbocharged engine compared to a naturally aspirated engine and what is their purpose?

5. Given a compressor flow ($\dot{m}_c$), compressor pressure ratio (Π_c), compressor efficiency (η_c) and compressor inlet temperature (T_{01});

   (a) How can the compressor outlet temperature (T_{02}) be calculated?

   (b) Given the temperature increase over the compressor, $\Delta T_c = T_{02} - T_{01}$, how can the compressor power be calculated?

   (c) Given the turbo speed (ω_{tc}) and the compressor power (P_c); how can the breaking torque of the compressor (Tq_c) on the turbo shaft be calculated?

6. Given a turbine mass flow ($\dot{m}_t$), turbine pressure ratio (Π_t), turbine efficiency (η_t) och turbine inlet temperature (T_{03});

   (a) How can the turbine outlet temperature (T_{04}) be calculated?

   (b) Given the temperature drop over the turbine, $\Delta T_t = T_{03} - T_{04}$, how can the turbine power be calculated?

   (c) Given the turbo speed (ω_{tc}) and the turbine power (P_t); how can the driving torque of the turbine (Tq_t) on the turbo shaft be calculated?

7. The pressure in the intake manifold is governed both by the throttle and the wastegate. Given a desired intake pressure, how should the control of these two actuators be coordinated for:

   (a) Maximum performance?

   (b) Best fuel efficiency?

8. A continuation on the previous exercise. The control system you develop should aim for a small pressure drop over the throttle during steady state operation ($\Delta p_{thr} = 10$kPa) what is the consequence on fuel consumption and response compared to the strategies in the previous exercise:

   (a) The most fuel efficient strategy?

   (b) The maximum performance strategy?

6.2 Turbocharger parameter estimation and model validation

1. Estimate the parameters in your model for:

   (a) The turbine mass flow

   (b) The turbine efficiency

   Plot and validate your models against the measured turbine map (reproducing the lower part of Figure 5.2). Present all estimated parameters and discuss your models performance. Are your models better/worse in some areas of the operating range? What areas are important?

2. Estimate the parameters in your model for:

   (a) The compressor mass flow

   (b) The compressor efficiency

   Plot and validate your models against the measured compressor map (reproducing the upper part of Figure 5.2). Present all estimated parameters and discuss your models performance. Are your models better/worse in some areas of the operating range? What areas are important?

6.2.1 Suggested turbocharger component models

This section presents suggestions for component models of the main parts of the turbocharger that you should implement: the compressor, turbine and the wastegate. It also describes the turbo shaft model that is already implemented.

Suggested compressor mass flow model

A model developed by Lars Eriksson is suggested for the compressor flow, described by the equations

$$\Pi_{c,max} = \left(\frac{U_2^2 \Psi_{max}}{2 c_p T_{af}} + 1 \right)^{\frac{\gamma}{\gamma-1}}, U_2 = r_c \, \omega_{tc} \tag{6.1}$$

$$\dot{m}_{c,corr} = \dot{m}_{c,corr,max} \sqrt{1 - \left(\frac{\Pi_c}{\Pi_{c,max}} \right)^2} \tag{6.2}$$

$$\dot{m}_c = \dot{m}_{c,corr} \frac{p_{af}/p_{ref,c}}{\sqrt{T_{af}/T_{ref,c}}} \tag{6.3}$$

where $p_{ref,c}$ and $T_{ref,c}$ are the reference states used for the map, and p_{af} and T_{af} are the pressure and temperature at the compressor inlet. The compressor blade speed U_2 is easily calculated using the rotational speed of the compressor and the compressor diameter. The model parameters to be estimated are Ψ_{max} and $\dot{m}_{c,corr,max}$. Ψ_{max} should be in the order of 1.

Suggested compressor efficiency model

The model from the course book, **Model 8.5** is used to describe the compressor efficiency, with a small change that was introduced in a dissertation from 2005[1]

$$\eta_c = \eta_{c,max} - \chi^T \, Q_\eta \, \chi \tag{6.4}$$

$$Q_{\eta_t} = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{12} & Q_{22} \end{bmatrix}, \chi = \begin{bmatrix} \dot{m}_{c,corr} - \dot{m}_{c,corr@\eta_{c,max}} \\ \sqrt{\Pi_c - 1} - \left(\Pi_{c@\eta_{c,max}} - 1 \right) \end{bmatrix} \tag{6.5}$$

In total there are six parameters in the model that you should estimate from the compressor map, using a nonlinear least squares solver (see Section 6.4):
$\eta_{c,max}$, $\dot{m}_{c,corr@\eta_{c,max}}$, $\Pi_{c@\eta_{c,max}}$, Q_{11} , Q_{12} and Q_{22}.

The model parameters $\dot{m}_{c,corr@\eta_{c,max}}$ and $\Pi_{c@\eta_{c,max}}$ are describing where in the map the maximum compressor efficiency $\eta_{c,max}$ is, and Q_{ij} is describing how the compressor efficiency changes when moving away from this point. Observe that the Q-matrix is symmetric.

OBS: Since the model for the efficiency is decreasing quadratically it will give unreasonably low or even negative efficiencies outside the normal operating region. You will have to put a lower saturation limit to the modeled efficiency, a suggestion is to use half the minimum value found in the map. It is also important to not get negative values in any square root calculation, good praxis is to limit the input to a minimum of zero before any square root calculation.

[1] P. Andersson, Air Charge Estimation in Turbocharged Spark Ignition Engines, 2005, p.112-113. The complete dissertation can be downloaded from `www.fs.isy.liu.se`.

Suggested turbine flow model

A version of the turbine model presented in the course book in **Model 8.14** is suggested, described by the equations

$$TFP_{model} = k_0\sqrt{1-\Pi_t^{k_1}} \qquad \Pi_t = \frac{p_{04}}{p_{03}} = \frac{p_{es}}{p_{em}} \tag{6.6}$$

$$\dot{m}_t = \frac{p_{em}}{\sqrt{T_{em}}} TFP_{model} \tag{6.7}$$

The parameters to be estimated are k_0 and k_1. Note that the definition of Π_t can differ from the model in the book and is given according to values in the turbomap. Also note that you often don't use SI-units for correction of the turbine flow, see Section 5.2.1.

Suggested turbine efficiency model

The turbine efficiency is modeled according to p.252 in the course book

$$BSR = \frac{\omega_{tc} r_t}{\sqrt{2c_{p,exh} T_{em}\left(1-\Pi_t^{\frac{\gamma_{exh}-1}{\gamma_{exh}}}\right)}} \tag{6.8}$$

$$\eta_t(BSR) = \eta_{t,max}\left(1-\left(\frac{BSR-BSR_{max}}{BSR_{max}}\right)^2\right) \tag{6.9}$$

where the model parameters $\eta_{t,max}$ and BSR_{max} should be estimated from the measure turbine map.

OBS: See OBS for the compressor efficiency above.

6.3 BMEP model

In project 2B, a model is required to calculate BMEP as a function of intake pressure, $p_{im,ref}$, which looks as follows:

$$bmep_{mod} = -C_{P0} + C_{P1} \cdot p_{im} \tag{6.10}$$

Estimate the parameters C_{P0} and C_{P1}, and include the estimated values in the report together with a validation where you plot measured $bmep$ and $bmep_{mod}$ against p_{im}. The model is parametrized using the measured engine map.

6.4 Tips for project 2A

1. Linear least squares problems can be easily solved in Matlab using the backslash operator. In this project you need to solve nonlinear least squares problems for the parameter estimation. There are several ways and the one suggestion is exemplified, and that is to use the function `lsqcurvefit`. Here is an example on how to use it when two constant parameters $a(1)$ and $a(2)$ should be calculated using x_1 and x_2 and y as measured data while we have $y = f(a, x)$ as follows:

```
% Define the nonlinear function
f = @(a,x)(a(1).*sqrt(x(:,1)-1./x(:,2).^a(2)));
% Define reasonable start values on the parameters
a_0 = [0.05,2]; %starting initial guess for a=[a(1),a(2)]
X = [x_1,x_2];
k = lsqcurvefit(f, a_0, X, y);
```

   To ease the nonlinear parameter estimation reasonable starting values should be given to the function. If it is hard to guess reasonable values they can be set to zero initially considering that this should not produce division by zero or negative square root values in the f function. Then using the obtained $a(1)$ and $a(2)$ values from a first try with lsqcurvefit, lsqcurvefit is used again and this is repeated until values do not differ after new iterations. Since the estimation does not guarantee convergence to a global optimum, it is very important to validate the model.

2. Assume $\gamma_{air} = 1.4$, $\gamma_{exh} = 1.3$ and $R_{air} = R_{exh} = 280 J/kg/K$ (which then gives c_p and c_v). These should already be implemented in matlab template.

Chapter 7

Project 2B: Implementation and evaluation

Project goal

At the end of this project you will have a working Simulink model for a vehicle equipped with a turbocharged engine. The main goal of this part is to implement your models estimated in Project 2A. It is recommended that you start with (a copy of) your model from project 1C, which already includes a turbocharger block with empty subsystems for the models you need to implement.
Exercise 1 and 2 should be implemented in the Simulink inside the red `Turbocharger` block. Exercise 3 and parts of 4 should be implemented in Simulink inside `Boost Control` block. The template file *Project2B.m* on the course homepage should be used for exercise 4-8.
Remember to check the course homepage before starting this project. Last minute changes in the Turbocharger and Boost Control blocks are common and will be announced there.

Examination Requirements

To get a pass in this project a report with the following properties should be submitted before the deadline:

1. Perform the tasks and answer all mandatory questions in the exercise section (Section 7.1).

2. Present parameter values used in the controllers or estimated in the throttle feed-forward, and describe how they have been calculated.

3. All presented plots should be followed by a discussion explaining why your

model and controller behave satisfactory.

4. All plots **must**: be clear and zoomed to the regions of interest with as small blank space as possible, have proper axis names with units, legends, captions.

5. A complete submission of report includes: The report in `.pdf` format, *project2B.m*, *project2B.m*, Complete Simulink model which should be executable after running *project2B.m*. All files should be submitted as a single .zip package.

6. The *.zip* package including the report should be submitted to the responsible person for project 2B according to the course homepage. The .zip file should have the following name format according to student LiU-IDs: *LiUID1_LiUID2_Project2B.zip* and *LiUID1_LiUID2_Project2B.pdf* for the report.

7.1 Exercises

The exercises marked with a $(\star)$ are not mandatory. Please do these tasks if you have time, or at least think of a possible answer and discuss with your course assistant.

1. Implement your compressor model in Simulink. The wastegate and turbo shaft models described in 7.2 and 7.3 should also be implemented inside the *Turbocharger* block. To ensure that the model is correct before running the engine, you are required to first run the model in a separate file, with constant inputs corresponding to a few points in the map. For at least three points, do the following and include it in the report:

   (a) Compare the modeled mass flow and efficiency with the measured values, and the validation plot from exercise 2 in Project 2A.

   (b) Ensure that the torque (Tq_c) is reasonable, it should be in the order of single newton meters or below.

2. Implement your turbine model in Simulink. To ensure that the model is correct before running the engine, you are required to first run the model in a separate model file, with constant inputs corresponding to a few points in the map. For at least three points, do the following and include it in the report:

   (a) Compare the modeled mass flow and efficiency with the measured values, and the validation plot from exercise 1 in Project 2A.

   (b) Ensure that the torque Tq_t is reasonable, it should be in the order of single newton meters or below.

3. Implement the driver gas pedal interpretation from gas pedal position to intake pressure reference. Instructions on how the interpretation should work is found in Section 7.4.1. The transformation from torque to intake pressure reference requires the model for $bmep$ which was introduced in 6.3.

4. Implement the feedforward part of the throttle controller according to the tips in Section 7.4.2. For the throttle feedback, a PI-controller with "tracking" is already implemented. You should tune the K_p and K_i-parameters to give a reasonable behavior with not too large overshoots and oscillations. Remember that the throttle is the main actuator for air control when no boost pressure is required. Therefore it is suggested to parametrize the throttle controller for operating conditions where $p_{im,ref} < p_{amb}$. This corresponds to small gas pedal position values. For validation of throttle controller, perform a step in gas pedal position with fixed engine speed, and plot the response in intake pressure p_{im} together with the reference $p_{im,ref}$. For this, after implementing the feedforward part, use the lines of code provided in *Project2B.m* for exercise 4.

   (a) What K_p and K_i values are used in the feedback part of the throttle controller?

(b) How good does the intake manifold pressure follow the reference when using both feedback and feedforward? Comment on the step response plot.

(c) How does the step response look like without including the throttle feedback (only using feedforward)? Present and describe about the step response for this case also.

(d) why a feedback is needed to guarantee correct intake pressure? Why only a feedforward is not good enough ?

5. The same structure as for the throttle is used for the wastegate feedback. As for the throttle, you should tune the K_p and K_i-parameters to give a reasonable behavior with not too large overshoots and oscillations. Since the wastegate is the main actuator for air control when boost pressure is needed, it is suggested to parametrize the wastegate controller for operating conditions with $p_{im,ref} > p_{amb}$ which is obtained by large gas pedal position values. For validation, include a step response in pedal position with fixed engine speed under this condition, and plot the response in both intake pressure (p_{im}) and boost pressure (p_{ic}) along with its references and the control signals for throttle and wastegate. Additionally, the control system should aim for a fully open wastegate for operating points where no boost pressure is needed, why?

6. To the project you will get interpolation curves with maximum and minimum torque as function of engine speed. These are used in the interpretation of the gas pedal position (see Figure 7.1).

   (a) Plot the maximum torque curve at different engine speeds once from the torque model in the engine block and another time from the provided interpolation curves used in gas pedal interpretation. (i.e. plot $Tq_{e,max}(N_e)$). This can be done for example by doing a slow ramp response in engine speed with maximum gas pedal position (see the lines of code related to this exercise in *project2B.m*). Explain why there is a difference between engine torque and torque from interpolation data.

   (b) Also plot the the compressor operating points corresponding to part (a) simulations on the compressor map (i.e. plot $\Pi_c(\dot{m}_{c,corr})$) and comment the results. In which regions does the compressor operate and what is the relation between the maximum torque and compressor operating region ?

7. In this exercise you should compare performance for your larger naturally aspirated engine from Project 1 with your smaller, turbocharged engine.

   (a) Compare acceleration performance, 70-110 km/h on fourth gear.

   (b) Compare fuel consumption and emissions for the driving cycle.

8. What maximum torque do you get from the engine if you do a step in pedal position from $ped = 0\%$ to $ped = 100\%$ for an engine speed of $N_e = 2500$rpm. Explain the behavior, i.e. the shape of the transient.

9. ($\star$) It is not only the drivers gas pedal position that controls the desired brake torque from the engine. What other systems of the engine can "require" torque?

10. ($\star$) How does the wastegate actuator system normally work, from control signal (usually a PWM signal), to actual valve position.

11. ($\star$) A single stage turbocharger has limitations on maximum torque and power. Where does these limitation originate from? How can it be remedied?

7.2 Wastegate model

In petrol engines, the most common way to control boost pressure is using a wastegate valve (WG). The valve lets exhaust gas bypass the turbine, which reduces the turbine power. The model for flow through a restriction that was used for the throttle is also applicable for the wastegate valve (that is, using Cd_{wg} och A_{wg} etc, in (2.2)). However, for the WG the effective area is given by

$$A_{\text{eff,wg}} = wg_{\text{pos}} \cdot \left(C_{\text{D,wg}} \cdot A_{\text{max,wg}}\right) \tag{7.1}$$

where $wg_{\text{pos}} \in [0..1]$, and $A_{\text{max,wg}}$ is the maximum opening area of the valve. Thus the model scales the effective area linearly with position, for example $wg_{\text{pos}} = 50\%$ gives an effective flow area of $0.5 \cdot \left(C_{\text{D,wg}} \cdot A_{\text{max,wg}}\right)$. For the wastegate position the following model can be used

$$\dot{wg}_{\text{pos}} = \frac{1}{\tau_{\text{wg}}} \left(wg_{\text{pos,ref}} - wg_{\text{pos}}\right) \tag{7.2}$$

where the time constant $\tau_{\text{wg}} = 0.1$ s.

7.3 Turbo shaft model

Friction on the turbo shaft is modeled as

$$Tq_{\text{tc,fric}} = c_{\text{tc,fric}} \cdot \omega_{\text{tc}} \tag{7.3}$$

where $c_{\text{tc,fric}} = 1 \cdot 10^{-6}\text{Nm}/(\text{rad/s})$. Newtons second law for a rotating system is then used to model the rotational speed of the shaft

$$\frac{d\omega_{\text{tc}}}{dt} = \frac{1}{J_{\text{tc}}} \left(Tq_{\text{t}} - Tq_{\text{c}} - Tq_{\text{tc,fric}}\right) \tag{7.4}$$

where ω_{tc} is the rotational speed of the turbocharger, Tq_{t} is the driving torque from the turbine, Tq_{c} is the breaking torque from the compressor, $Tq_{\text{tc,fric}}$ is the friction torque from the model above, and J_{tc} is the inertia for the shaft, turbine and compressor wheels. In the simulation model also a reasonable starting speed is required.

7.4 Components of the engine control structure

The maximum available torque for a modern engine is shaped by the control system. Most turbocharged engines have a flat torque curve over a relatively wide engine speed interval. The goal for this project is to create a very downsized engine, while maintaining a good output torque. Maximum and minimum engine torque is given by two interpolation tables in engine speed, that is:

$$Tq_{e,max} = f_{Tq_{e,max}}(N_e) \tag{7.5}$$
$$Tq_{e,min} = f_{Tq_{e,min}}(N_e) \tag{7.6}$$

where the right hand side is given by 1D-interpolation functions.

7.4.1 Gas pedal interpretation

To work with the driver model from Project 1 (PI-controller), the pedal position signal is interpreted in % of maximum available torque in the torque based control structure according to: (i) pedal positions where $ped \in [0..10]\%$ is interpreted as $[100..0]\%$ of maximum *braking* torque for a given engine speed. (ii) $ped \in [10..100]\%$ is interpreted as $[0..100]\%$ of maximum available *driving* torque for the given engine speed. Thus the engine tries to control the torque to zero for $ped = 10\%$, and tries to for example give half of the available torque if $ped = 55\%$. Mathematically this can be written as

$$Tq_{e,ref}(ped, N_e) = \begin{cases} Tq_{e,max}(N_e)\frac{ped-0.1}{1-0.1} & ped > 0.1 \\ Tq_{e,min}(N_e)\frac{0.1-ped}{1-0.9} & \text{else} \end{cases} \tag{7.7}$$

where ped is the pedal position signal($[0..1]$). In a production control system this translation from pedal position to desired torque is more complex, and is important for how the engine is perceived. The desired torque is then translated to a $bmep$ reference

$$bmep_{ref} = \frac{2\pi n_r Tq_{e,ref}}{V_D} \tag{7.8}$$

To get to an intake pressure reference, $p_{im,ref}$, that can be used in a feedback loop, the inverse of the affine model for $bmep$, use the model from 6.3, as function of p_{im} is used

$$bmep_{mod} = -C_{P0} + C_{P1} \cdot p_{im} \tag{7.9}$$
$$p_{im,ref} = \frac{bmep_{ref} + C_{P0}}{C_{P1}} \tag{7.10}$$

The methodology for calculating $p_{im,ref}$ given $Tq_{e,ref}$ thus becomes $bmep_{ref} = f(Tq_{e,ref}, V_D)$, $p_{im,ref} = f(bmep_{ref}, C_{P0}, C_{P1})$. Figure 7.1 and 7.2 show two examples of how a control system implementation can look in Simulink.

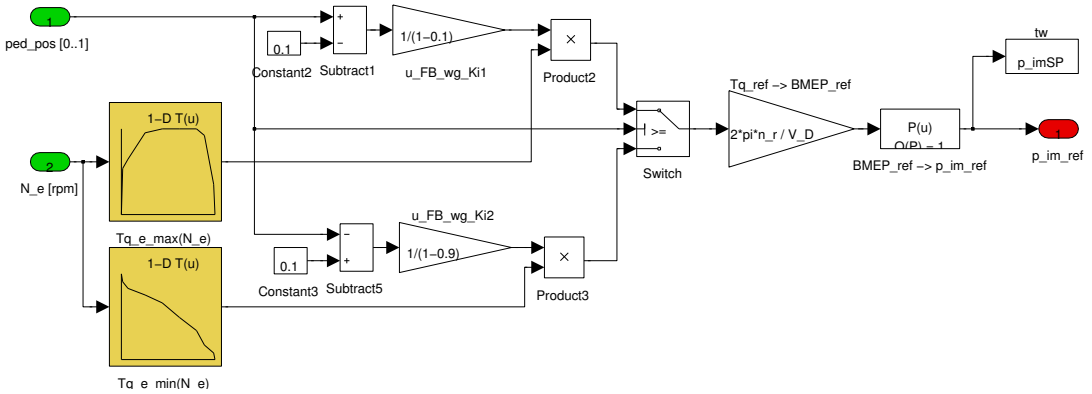

Figure 7.1 *Interpretation of the gas pedal position to an intake manifold pressure reference* $p_{im,ref}$*. The shaded blocks to the left are 1D maps for maximum and minimum torque as function of engine speed. These maps are given for the project and can be downloaded form the project homepage. The calculation order is to first compute a desired torque from engine speed and gas pedal position. The desired torque is translated to a* bmep *reference which given the displacement volume of the engine can be used to calculate an intake pressure reference.*

7.4.2 Suggested throttle feedforward

A model based throttle feed forward can be created, given an engine speed and intake manifold pressure. The feedforward uses your model for the air flow through the engine and your model for flow past the throttle.

In a real throttle controller, the inverse of a throttle model can be used to design a feedforward controller. Since models are not perfect, the feedforward part would still require feedback to control the throttle properly. In this exercise we are tying to build a controller for the throttle model, so, using an inverse of the model would make the controller perfect. In order to have a more realistic scenario, we want to avoid that the feedforward part is the exact inverse of the model. A way to prevent this is to use a constant temperature, e.g. $T = 293K$, when you implement in on Simulink instead of the "measured" constant intake manifold temperature T_{im}.

$$\dot{m}_{air,ref} = f(\eta_{vol}(T, N_e), p_{im,ref}, N_e, V_D, T, R_{air}, n_r) \tag{7.11}$$

$$\Pi_{thr,ref} = \frac{p_{im,ref}}{\max(p_{im,ref}, p_{ic})} \tag{7.12}$$

$$A_{thr,eff,ref} = f(\dot{m}_{air,ref}, \Psi(\Pi_{thr,ref}), T, R_{air}) \tag{7.13}$$

We assume here that the volumetric efficiency is independent of engine geometry, that means that you can use your model for the volymetric efficiency from Project 1

OBS: In the calculation of $A_{thr,eff,ref}$ you divide with Ψ. Since $\mathtt{Psi}(1) = 0$ you will have to put a lower limit, > 0 but close to 0, for Ψ in the feedforward to avoid simulation problems. A suggestion is to use $\geq$ *eps* (*eps* in Matlab is the precision for floating point numbers).

When the reference for the throttle effective area, $A_{thr,eff,ref}$, is determined, $\alpha_{thr,ref}$ can

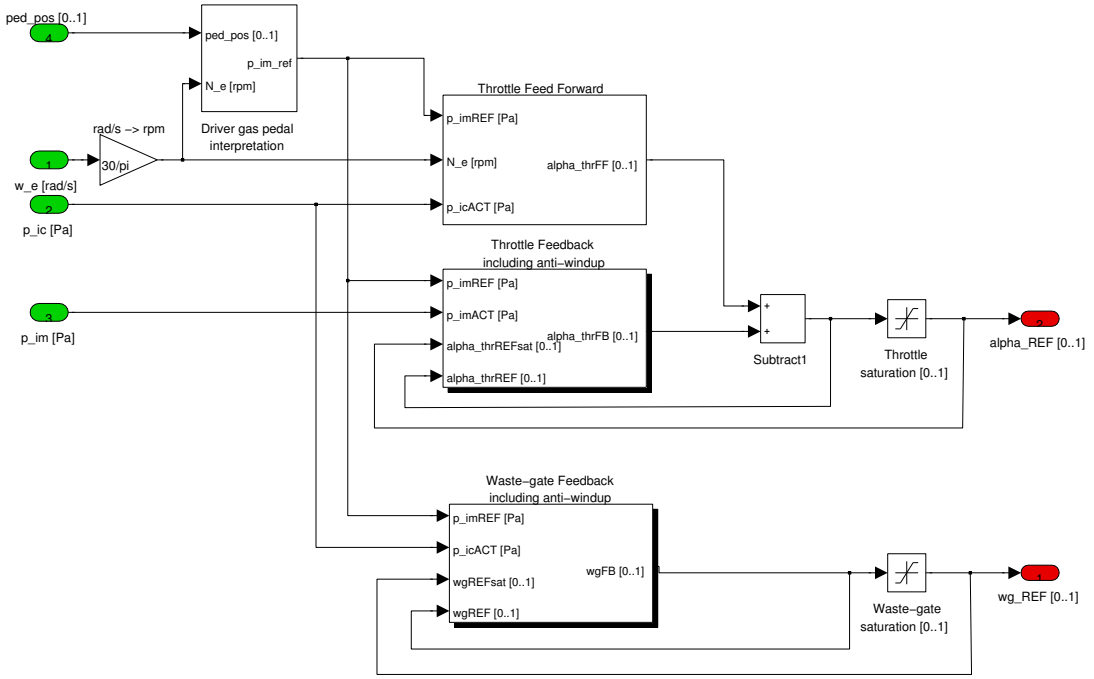

Figure 7.2 *Overview of the air control of a turbocharged engine. The first block is the gas pedal interpretation that was presented in Figure 7.1. This is followed by a throttle feedforward link above a feedback loop with tracking to avoid integrator wind-up. The block in the bottom is the wastegate controller that only consists of a feedback loop with tracking.*

be calculated by inverting the model for the effective area (given that the reference is between the minimum and maximum area for your throttle model, what should you do otherwise?) according to:

$$\alpha_{\text{thr,ref}} = -\frac{a_1}{2a_2} \pm \sqrt{\frac{A_{\text{thr,eff,ref}} - a_0}{a_2} + \left(\frac{a_1}{2a_2}\right)^2} \tag{7.14}$$

where the "correct" solution to the second order equation has to be chosen, that gives the $\tilde{\alpha}_{\text{thr,ff,n}}$ in the control structure below. The tilde symbol is added since the signal later in the structure will be filtered through a saturation, see below.

7.4.3 Throttle feedback controller

The throttle feedback is a PI-controller with tracking [1], implemented in discrete time:

$$e_{im,n} = p_{im,ref} - p_{im} \tag{7.15}$$

$$I_{thr,n} = I_{thr,n\text{-}1} + K_{p,thr}\frac{T_s}{T_{i,thr}}e_{im,n} \tag{7.16}$$

$$\tilde{\alpha}_{thr,fb,n} = K_{p,thr}e_{im,n} + I_{thr,n} \tag{7.17}$$

$$\tilde{\alpha}_{thr,n} = \tilde{\alpha}_{thr,fb,n} + \tilde{\alpha}_{thr,ff,n} \tag{7.18}$$

$$\alpha_{thr,ref,n} = \begin{cases} \alpha_{thr,max} & \text{if } \tilde{\alpha}_{thr,n} > \alpha_{thr,max} \\ \tilde{\alpha}_{thr,n} & \text{if } \alpha_{thr,min} < \tilde{\alpha}_{thr,n} < \alpha_{thr,max} \\ \alpha_{thr,min} & \text{if } \tilde{\alpha}_{thr,n} < \alpha_{thr,min} \end{cases} \tag{7.19}$$

$$I_{thr,n} = I_{thr,n} + \frac{T_s}{T_{t,thr}}\left(\alpha_{thr,ref,n} - \tilde{\alpha}_{thr,n}\right) \tag{7.20}$$

where the tracking constant is chosen according to the rule of thumb $T_{t,thr} = T_{i,thr}$, $\tilde{\alpha}_{thr,ff,n}$ is the contribution from the feed forward link, and T_s is the sampling time. The implementation of the control structure is given in the project template, and you are expected to tune the controllers $K_{p,thr}$ och $T_{i,thr}$ parameters.

Since the throttle is the main actuator for air control in operating points where no boost pressure is required, it is suggested to parameterize the throttle controller for operating conditions with $p_{im,ref} < p_{amb}$.

7.4.4 Wastegate feedback controller

For the WG controller only a feedback loop is used. The control structure is given in the template, and follows the structure that where presented for the throttle above. For the wastegate though, the contribution from the feedforward (`ff`) is of course removed. The control error for the wastegate controller includes the desired pressure drop over the throttle, $\Delta p_{thr,ref}$, and the reference value for the pressure before the throttle (that is your pressure in the intercooler control volume) is then given by

$$p_{ic,ref} = p_{im,ref} + \Delta p_{thr,ref} \tag{7.21}$$

$$e = p_{ic} - p_{ic,ref} \tag{7.22}$$

where p_{ic} is measured. The WG controller that is implemented in the template has, exactly as the throttle controller, tracking to avoid integrator wind-up. The tracking constant for this controller is also chosen to $T_{i,wg}$ and you are expected to tune controller parameters $K_{p,wg}$ and $T_{i,wg}$.

Since the wastegate is the main actuator for air control in operating points where boost pressure is needed, it is suggested to parameterize the wastegate controller for operating conditions with $p_{im,ref} > p_{amb}$.

[1] See for example the course compendium in "Industriell reglerteknik", p.95, for more details.

7.5 Tips for project 2B

This section includes tips that are useful for Project 2B. Figure 7.3 shows an overview of the simulink template for the turbocharged engine model.

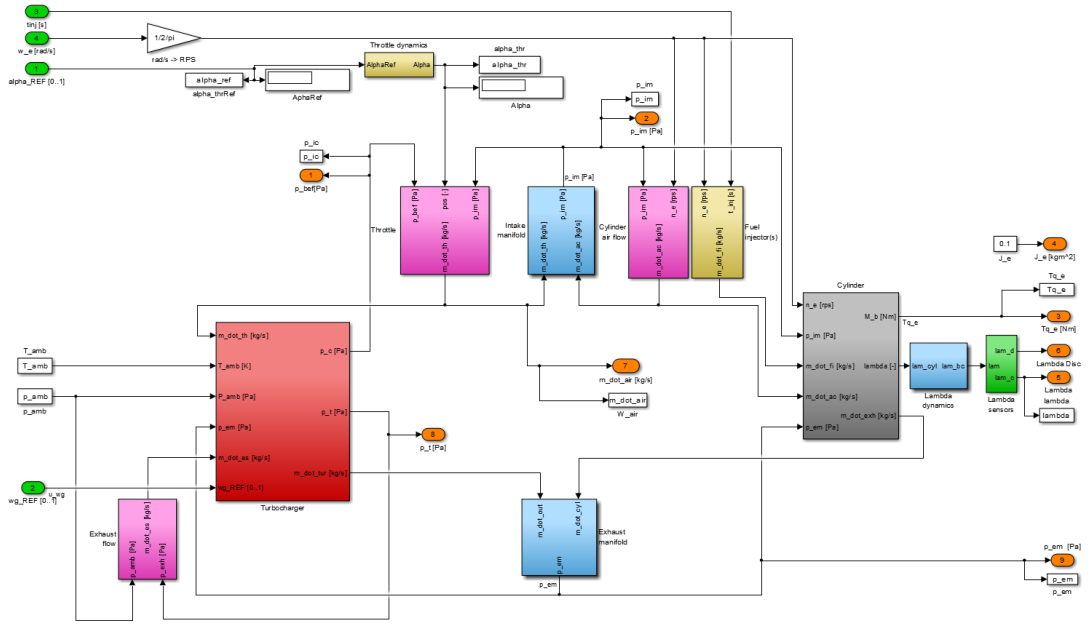

Figure 7.3 *Overview of the template for the turbocharged engine model. The air enters the turbocharger (the big red block), containing the compressor, in the left of the figure. It then moves through the throttle, intake manifold, cylinders, exhaust manifold, turbine and/or wastegate (also in the turbocharger block) and eventually exits through the exhaust in the bottom left.*

7.5.1 Matlab tips

1. In project 2B you will extend your model from Project 1 with new component models. The template from Project 1 is prepared with empty subsystem blocks for the components that you should implement in Project 2. Make a copy of the model from Project 1 before starting.

2. Use a well defined interface between your component models, and validate each component individually before you put them together. Ensure that your sub-models give reasonable outputs before you connect them, to avoid unit and scaling errors. A good step for implementation could be:

   (a) Test the compressor/turbine model separately with constant inputs form a few points in the map. Compare efficiency and mass flow with measured data and ensure that the torque is reasonable in size (in the order of single Nm).

(b) Connect the model in the complete engine model and test with constant engine speed, throttle angle and wastegate control signal. Check that you get reasonable pressure and torques etc.

(c) Implement the controller

(d) Run with constant engine speed and constant pedal position (let the control system handle throttle angle and wastegate control). Check that you get reasonable behavior.

(e) Run a drive cycle and check that you get reasonable behavior.

3. To make it easy to simulate specific drive scenarios with your model there is a block called "I/O abstraction layer" in the template. This can be used to set manual inputs to the engine for specific investigations. The block is prepared to control N_e, $\alpha_{thr,ref}$, $wg_{pos,ref}$ och ped_{pos}. For example to get the maximum torque for the engine with your control system, you could fix the pedal position to 100% and do a slow ramp in engine speed over the whole speed range of the engine. If the ramp is made slow enough, the operation could be considered stationary and you will get the maximum torque as function of engine speed.

4. Additional help can be found on the project homepage:
`www.fs.isy.liu.se/Edu/Courses/TSFS09/Projekt`

7.5.2 Relevant data and assumptions

1. The boost control system should try to achieve a pressure drop over the throttle, equal to $\Delta p_{thr,ref} = 10$kPa.

2. The maximum and minimum torque shaped by the control system is given for the project as functions of engine speed, see Figure 7.1. These curves can be downloaded from the project home page.

3. Since both the naturally aspirated engine (Project 1) and the turbocharged engine (Project 2) has similar maximum torque, the friction model should be reasonably similar (due to similar demands on bearings, piston rings, etc.) You can therefore use the same friction model for the smaller turbocharged engine, as for your engine in Project 1.

Part III

Project 3

Chapter 8

Driveline Oscillations

8.1 Introduction and Purpose

The purpose of the project is to study the resonances in a drive line, and to study how the oscillations can be damped. One important case where drive line oscillations are introduced is during tip-in and tip-out maneuvers, i.e. during changes in accelerator pedal position causing the drive line torque to go from negative to positive and vice versa. The task in this project is to design and implement a controller that damps oscillations during tip-in/tip-out. The controller will then be tested on a realistic drive line model provided on the course website. The model is of a typical passenger car. To resemble a real case only the engine speed sensor will be available for "measurements" in the simulation model. This sensor is chosen because it is typically the one in the drive line with sufficient resolution for the given task. Data for the studied car is listed in Section 8.6, note that the wheel assembly is assumed to me friction free (b_w in the course literature equals 0). The Model with drive shaft flexibility[1] should be used to describe the vehicle driveline. The drive shaft flexibility model is used to include the effects of having inertia's in the final drive, transmission and the engine. These inertias are lumped together and connected to the wheel inertia by a damped torsional flexibility shaft[2]. To damp the oscillations in the driveline, a state feedback controller will be implemented. For this, all state variables of the driveline as state-space model should be available from measurements. However, to decrease the costs and reduce the number of required sensors, an observer will be used to estimate necessary variables.

[1] See model 14.2 in Modeling and Control of Engines and Drivelines.
[2] See figure 14.3 in Modeling and Control of Engines and Drivelines.

8.1.1 Observer

An observer can be used to reconstruct the states inside the system, using input control signals and measured output signals of the system and a system model. Observers are frequently used in vehicle industry to develop cost efficient diagnostics and controller systems, instead of using extra sensors. A system given by

$$\begin{aligned}\dot{x} &= A\,x + B\,u\\ y &= C\,x\end{aligned}$$

has the observer equation

$$\begin{aligned}\dot{\hat{x}} &= A\,\hat{x} + B\,u + K(y - C\,\hat{x})\\ y_{observed} &= C\,\hat{x}\end{aligned}$$

where the observer gain K has to be chosen. $y_{observed}$ is the output signal from the observer, containing the estimated states $\hat{x}$. In the project, reference signals for the states will be given. Note that the reference states are time varying and not constant.

8.2 Examination

To achieve the grade passed on this project written solutions to the prerequisites and the exercises have to be handed in before the deadlines given on the course website. Append matlab code and simulink block diagrams of your implementation in the report. Figures in too poor quality or bad axis adjustments which makes them either hard to read or not show the results, will be rejected and you will have to recreate the figures.

8.3 Prerequisites

1. Write down equations for a driveline including the four states drive shaft torsion, engine speed, wheel speed, and engine torque. Consequently, you have to extend the driveline models in the chapter "Driveline Modeling" in the course compendium with a state for the engine torque. This state can be modeled as a first order system with a time constant. The input signal to the complete model describing the four states should be demanded engine torque and the output signal should be engine speed. All the parameters in Section 8.6 must be considered in the model except for the parameter "Engine maximum torque" that you will consider in the implementation of the controller. It is sufficient to assume rolling condition for the wheels. The friction torques in the system are assumed to be proportional to the rotational speeds. The rolling resistance can be assumed to be constant, the air drag can be neglected, and the vehicle can be assumed to drive at a horizontal road.

2. Put the equations in standard linear statespace form and let constant terms be represented by a known constant input signal l according to:

$$\dot{x} = A\,x + B\,u + H\,l$$
$$y = C\,x$$

The order of the states should be: drive shaft torsion, engine speed, wheel speed, and engine torque.

3. Draw a block scheme for a control system of the driveline consisting of an observer and a state feedback controller using the observed states. As mentioned above only engine speed is available for measurements. In the project reference signals for the states will be given. Note that the reference states are time varying and not constant. Since we have several states entering the controller, the residual $\bar{r} = x - \hat{x}$ has to be created.

4. Write down the equations for an observer of the drive line model.

5. Write the observer on the form

$$\dot{\hat{x}} = A_{obs}\hat{x} + B_{obs}u_{obs}$$
$$y_{obs} = C_{obs}\hat{x}$$

where u_{obs} are all inputs to the observer and y_{obs} are all outputs from the observer. What is A_{obs}, B_{obs}, C_{obs}, u_{obs}, and y_{obs} expressed in A, B, H, C, u, l, y, $\hat{x}$, and observer gain?

This prerequisite will help you to implement the observer in the project in exercise 1 below.

6. How can the observer gain be chosen?

7. Write down the equations for a state feedback controller that uses the observed states. Remember that you will be given reference values for the states drive shaft torsion, engine speed, wheel speed, and engine torque.

8. How can the feedback gain be chosen?

8.4 Exercises

Download the driveline model from the course homepage. All your implementations should be done in the "car/car_model/controller/controller"-block. There is also a script "DoSim.m" that initializes the model, simulates it, and plots relevant signals.

1. Implement an observer for the driveline using a "state space"-block together with the solution from prerequisite 5.

2. Set the observer gain to zero, simulate, and demonstrate the performance of the observer with plots of the observed and the real drive line variables in the same plot.

3. Use your calculated observer gain, simulate, and demonstrate the performance of the observer with plots of the observed and the real drive line variables in the same plot. Is the performance improved compared to exercise 2?
4. Implement a state feedback controller that that damps the drive line oscillations. In the controller block reference states are already given. Use them to try to damp the oscillations in vehicle acceleration and drive shaft torsion. The requirements for the controller are:
   - The vehicle acceleration should be around $1.5\mathrm{m/s^2}$ when the feedback controller is active.
   - The maximum allowed rise time and overshoot of the acceleration is displayed in the plot produced by the Matlab script.
5. Display the final result according to the following questions:
   (a) Plot the vehicle acceleration with and without the state feedback. Attach the plot with the vehicle acceleration and demanded torques for both cases, comment the results.
   (b) Demonstrate the performance of the controller with plots of the drive line variables both in damped and undamped mode. Comment the results.

8.5 Hints

- If you use Kalman estimator design for the observer: The parameters for the process noise and the measurement noise are unknown. Consider these parameters as tuning parameters when you are evaluating the observer design.
- There are different ways to design the state feedback controller. One way is to choose the feedback gain such that the poles of the closed loop system is placed at desired places. When doing this it has to be assured that the commanded control is of reasonable size and can be actuated. One way to take this into account is to use LQ-design which finds the feedback gain that minimizes a criteria that is quadratic in states and control.
- There is a torque limitation of 400Nm in the car model. This means that if your controller tries to acquire to much torque, your observer will be inaccurate. Therefore it is wise to include this torque limitation on your controller output.
- You can select initial conditions for the estimated states $\hat{x}$ in the state space block in Simulink.

8.5.1 Useful matlab commands

Type *help control* in matlab to see the help for the control toolbox.

Specific useful commands are:

- *care*: Solves the continuous algebraic riccati equation.
- *kalman*: Continuous- or discrete-time Kalman estimator
- *ss*: Cretates state space model object. SS(A,B,C,D) creates a SS object representing the continuous-time state-space model.
- *place*: Pole placement.
- *lqr*: Linear-quadratic regulator design.
- *lqe*:Kalman estimator design.

8.6 Vehicle Data

Parameter	Description	Value	Unit
m	Vehicle mass	1500	[kg]
r_w	Wheel radius	0.3	[m]
J_w	Wheel inertia	0.6	[kgm^2]
c_{rl}	Rolling resistance coefficient	0.147	[m/s^2]
J_m	Engine inertia	0.2	[kgm^2]
τ_e	Engine step response time constant	0.1	[s]
M_{max}	Engine maximum torque	400	[Nm]
i_t	Transmission ratio 1:st gear	3.4	[-]
J_t	Transmission inertia 1:st gear	0.13	[kgm^2]
b_t	Transmission friction (viscous damping coefficient)	0.1	[Nms/rad]
i_f	Final drive gear ratio	3.4	[-]
J_f	Final drive inertia	0.1	[kgm^2]
b_f	Final drive friction (viscous damping coefficient)	0.1	[Nms/rad]
k	Drive shaft stiffness	1000	[Nm/rad]
c	Drive shaft damping	1	[Nms/rad]

Table 8.1 *Data for the passenger car studied in the project.*

Part IV

Appendix

Appendix A

Technical data on engines and measurement system

This section includes technical data about the engines in Vehicular Systems engine lab as well as information on sensors and measurement system. Th engine you will model in the project are the VEP-engine, but you will also use data from the SVC engine.

The dynamic measurements (except for the light-off time measurement) will be done on VEP engine. The SVC engine will be used to study the p-V diagram in real time, and how it changes with engine operating conditions, as well as cylinder pressure data analysis in Project 1b.

A.1 Engines that generated test data

The test data comes from two different engines. When collecting data the engines are connected to highly dynamic dynamometers, the dynamometers are also are used as starter motors for the engines. The dynamometers can consequently be used to power the engines even if there are no combustion, if desired. In the following section the two engines are described.

SVC Saab Variable Compression engine is an experimental engine where the compression ratio can be varied during operation. The engine is equipped with a mechanical compressor that can be engaged and disengaged with a magnetic clutch. The engine has a maximum brake torque of about 300 Nm and maximum speed of 6200,rpm. Other parameters needed for the Project assignment are included in the data files. The most important measurement and control signals for the course are:

Measurement signals: Engine speed, torque, intake pressure and temperature, cylinder pressure, crank angle.

Control signals: Throttle position reference, mechanical compressor clutch signal, injection length and time, ignition angle and dwell angle, compression ratio reference.

VEP An engine from engine from Volvos new engine family, with approximately 350 Nm maximum torque. The engine is a direct injected turbocharged engine, equipped with variable intake and exhaust camshafts.

Measurement signals: Control system: Pressure and temperature after air filter and before throttle, pressure in the intake manifold, air mass flow, throttle position, gas pedal position, continuous lambda, engine speed, crank angle, camshaft angles, knock sensors, fuel rail pressure, coolant water temperature, oil pressure and temperature, system voltage. The most important measurement and control signals for the course are:

Test cell measurement system: Pressure: after air filter, before and after intercooler, intake manfold, before and after turbine. Temperature: Ambient, after air filter, before and after intercooler, intake manifold, before and after turbine. Turbocharger speed.

Control signals: Gas pedal position, throttle position reference, injection time, wastegate control signal, intake and exhaust cam angle reference,

Appendix B

Simulink Troubleshooting

This sectiond contains some useful information when troubleshooting a simulink model that takes long time to simulate or crashes during simuation. If the simulation chrases you should always read the error message given by Simulink and trie to understand it. However, if this does not help you can trie the following:

- Make sure the solver is set up correctly. We recomend the solver *ode23tb* and relative- and absolute tolerance of 10^{-6}. These settings are available in the *Configuration Parameters* menu which you enter by clicking on the cogwheel in Simulink, see Figure B.2. To see al options you might have to press Additional Options.
- If you, for example, put a vecor instead of a scalar in a *Constat* block, Simulink will start as many simulations as the length of the vector and this will make you simulaiton slower. One way to see if this is the case is to look at the *Signal Dimensions* in simulink. To make simulink show signal dimensions press

  ```
  Display -> Signals & Ports -> Signal Dimensions
  ```

  see Figure B.1.

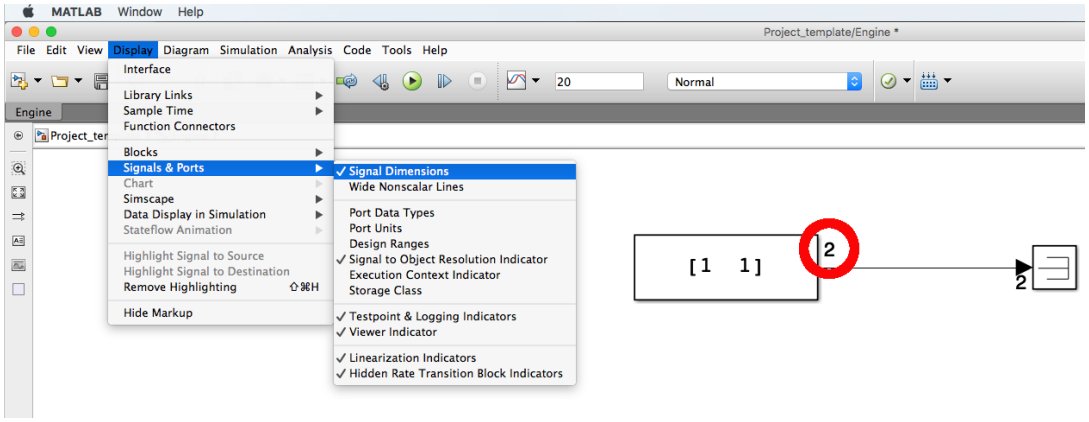

Figure B.1 *Figure showing how to make simulink show signal dimensions. The red cirkle mark where the singal dimension is shown.*

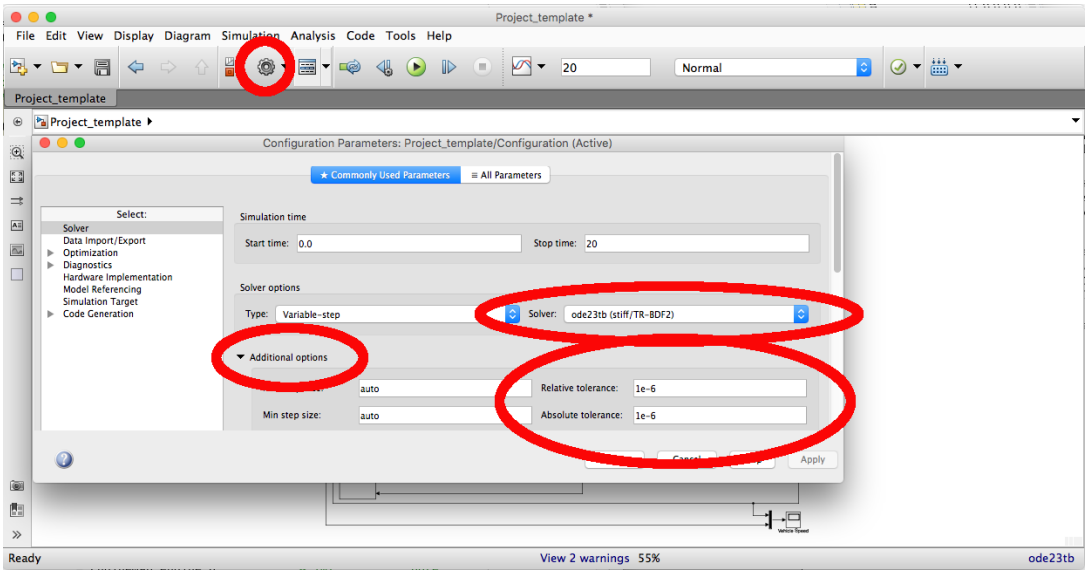

Figure B.2 *Figure showing how to wiew and change solver parameters. The red ellipses marks relevant fields.*