Training Unit 2: NEDC Simulation

1 INTRODUCTION TO TRAINING UNIT

After successfully completing Training Unit 1, this unit serves as a logical continuation, building directly on your MATLAB knowledge. In this exercise, you will learn how to import, manipulate, and display data in MATLAB, applying the commands you learned in the first unit.
Additionally, be sure to consult the MATLAB cheat sheet for data import / export and generic Matlab functions for extra assistance. You can also explore the comprehensive Matlab documentation online or access it via the "F1" key in MATLAB, where you'll find a wealth of examples and copy-paste code.
By the end of this unit, you will advance from a beginner to an intermediate level in MATLAB, gaining the skills to handle data, perform calculations, and visualize data with properly labeled axes and correct plotting layouts.

2 PREPARATION

- Successfully finished the Matlab Onramp tutorial
- Read through section 3 (technical background)

3 TECHNICAL BACKGROUND

To give you a "taste" of how MATLAB can be applied, this exercise is based on a representative example of an electric vehicle. You will import datasets, perform calculations on the provided data, and analyze the results. The main objective of this training unit is to determine the energy consumption and performance of the electric vehicle.
The foundation of this investigation is the test bench data, which you will need to import into your workspace. The data represents an electric vehicle driven over a specified drive cycle.
The drive cycle used in our example is the NEDC (New European Driving Cycle). This drive cycle was last updated in 1997 and was used until September 2018. This cycle was last updated in 1997 and was used until September 2018, when it was replaced by the more modern WLTP (Worldwide Harmonized Light Vehicles Test Procedure). The WLTP cycle is now widely accepted in the US, Japan, China, and many other countries. The NEDC cycle became outdated as it no longer reflects modern driving styles, with changes in distances and road conditions since the test was originally designed in the 1960s.

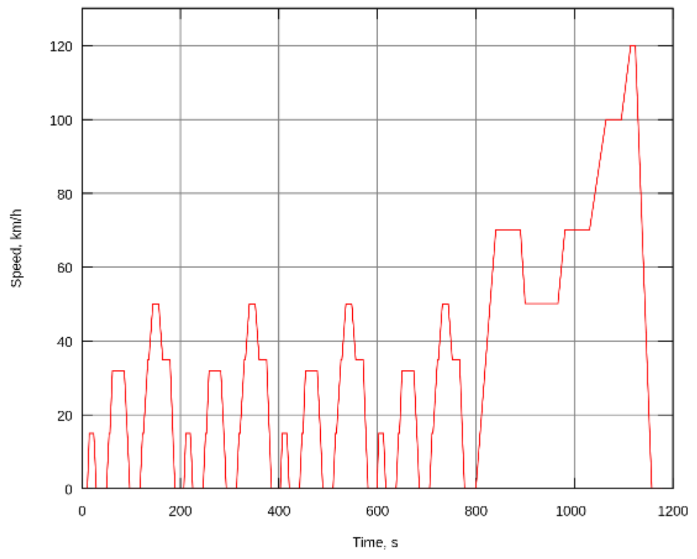

Figure 1: NEDC profile

The collected data includes information on **motor speed, motor torque, vehicle speed and driven distance**. Since the data was collected by different devices, the formats vary. One of your first tasks will be to **import the data and make it available and compatible within Matlab**.
Motor speed and motor torque are the key data points for assessing the vehicle's performance. Since these measurements were taken at the motor's output, they reflect the power required to propel the vehicle. For combustion engine vehicles, performance is typically measured by fuel consumption per 100 kilometers. In the case of the electric vehicle under investigation, our focus is on the **energy consumed per 100 kilometers**, drawn from the battery.

To estimate electric energy demand, the efficiency chain of an e-vehicle must be considered as in depicted in Figure 2.

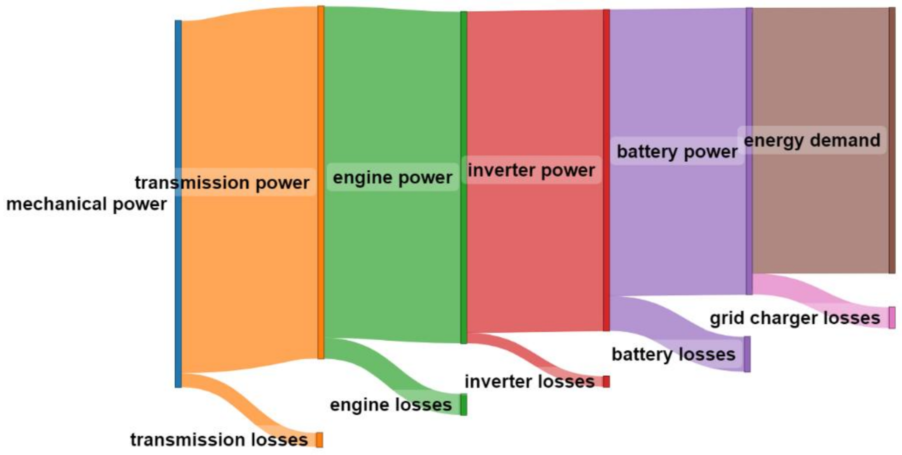

Figure 2: Vehicle efficiency chain

$\eta_{transmission}$ [%]	η_{engine} [%]	$\eta_{inverter}$ [%]	$\eta_{battery}$ [%]	$\eta_{grid\ charger}$ [%]
96	94	97	90	94

In terms of the efficiency chain, it is crucial to consider the direction of energy flow, as it can change depending on the driving conditions. When the vehicle is accelerating or going uphill, energy flows from the battery to the wheels. In this case, the energy demand from the battery is at its highest, reducing the state of charge (SOC). Conversely, when the vehicle is decelerating or going downhill, energy (normally lost as heat in combustion engine cars) can be used to charge the battery, reversing the energy flow—this process is known as recuperation. For a video on recuperating brake energy, check out following LINK. At this point, the energy on the electric motor side is higher than that of the battery, which requires a different approach to calculating the energy flow. Please consider it in your implementation!
On a later point of the analysis, it will be required to identify areas, where the vehicle is recuperating energy, and vice versa demanding energy, as the imported data provide a mixture of both.

The vehicle under investigation has following parameters:

parameter	value
vehicle weight	1250 kg
e-motor power	60 kW
battery capacity	22 kWh
transmission efficiency	96 %
engine efficiency	94 %
inverter efficiency	97 %
battery efficiency	90 %
charger efficiency	94 %

4 LABORATORY WORK

4.1 Load the workspace with the necessary data

4.1.1 Task description

- Download the files from the Moodle training unit section and store them locally.
- Create a live script (*.mlx), that will perform all following tasks of this TU automatically.
- Create a data store variable with the vehicle parameters mentioned above. The structure shall start with the name "VEHICLE_PARAMETER." and continue with the dedicated parameter name. e.g. VEHICLE_PARAMETER.VEHICLE_WEIGHT.data, and so on. Make sure to store not only the value itself, but also the unit of the value in a "string" or 'char-array'.
- Write a Matlab function (stored in a separate *.m) that performs following tasks:
    - Import all the data included in the *.xls file to your local workspace table (one column for every parameter).
    - Consider storing the measurement data within a new workspace variable which is of type [Matlab **TABLE**]
    - Display the size of the table directly in the Matlab console to check the table dimensions.
    - Name the columns in your Matlab table according to the names in the *.xls file (including the unit in brackets e.g. Voltage[V]).

    - Write a 2nd Matlab function (new *.m file) that does following tasks:
    - Read the file "Torque.mat" (contains a time series of data).
    - Open the data manually and try to investigate the timeseries data. What can you find out by examining the data? Is there additional information, that you did not know before? Explain in the report.
    - Store the torque data as 4th column in your already existing data table. Name the column as the "Name" field suggests in the dataseries array. Also append the unit in square brackets, as previously done.
    - Add the sampling time series as first column to your table, resulting in following table structure:

sampling_time [s]	motor_speed [rad/s]	distance [m]	vehicle_speed [km/h]	torque [Nm]

    - Remove the imported torque time series variable from your workspace, as well as additional artefacts that were created during the import process. Only leave the Matlab table and struct remaining in your Matlab workspace.

4.1.2 Presenting your results

- Plot all 4 variables of your table in one figure (using 4 subplots)
    - Make sure that the presentation of your results is self-explanatory.
    - Use axis names including the correct unit.
    - Use titles for every plot / diagram as well as a grid.
    - Use legends when plotting several lines within one plot.

4.1.3 Result:

After running the live script, created in this section, all data shall be read automatically and converted to a data table with correct column names and units. The table shall be presented in one descriptive plot in your live script.

4.2 Perform calculations / conversions on the data table

4.2.1 Task description

- Continue in your live script from section 1 (add a new section to your script) with following calculations.
- Convert the motor speed provided from rad/s into rpm. (research online for the formular)
    - Important: For the calculated values prepare a separate column at the end of the table. Also define the unit for those calculated values, as it improves readability of the complete table.
- Calculate the **mean vehicle speed** and save it to a **new data structure** similar like the “VEHICLE_PARAMETER” structure, but with a different name.
- Calculate the mechanical power demanded by the vehicle and save it to your table. (research online how the mechanical power can be calculated by using speed and torque)
    - Check the dimensions of torque and speed before performing calculations on them. (*hint: matrix multiplication vs. element wise multiplication*).
    - This calculated power demand equals the required power to propel the vehicle.
    - Is the engine power sufficient (look at your motor spec that you created) to support the calculated mechanical power demand? Be sure to document it in your report.
- Convert this mechanical power demand into the corresponding instantaneous energy (energy = integral of power, check some possibilities of how to perform a continuous integration on the mechanical power → result shall be a vector). Which unit does the calculated energy has? Append it to your table again with the correct unit.
    - Calculate the different instantaneous **energies** demanded at certain levels of the efficiency chain, (use the efficiencies specified) without taking care of the energy flow direction.
      NOTE: That isn’t completely correct as there will be an influence of the energy flow direction. For a first estimation, it should be ok.
    - Calculate the complete energy lost at each part of the efficiency chain and save it to your new data structure, where you stored the mean vehicle speed.

4.2.2 Presenting your results

- Plot the **vehicle speed**, the **mean vehicle speed** and the calculated **motor speed** in one plot.
    - Try to find a method of plotting the scalar variable “mean_speed” in a meaningful way.
    - How can you improve visibility for the curves, as they have different units and different maximum values? Try to use two y-axis in one plot (hint look-up yyaxis function).
    - Take care of correct axis description and scaling.
- Plot mechanical power over time.
    - Add a dashed line to your plot, to visualize the maximum available power of the engine.
- Plot the results of different energy demands of the electric vehicle and depict them within a single plot.
    - Visualize this task better, by colouring the area between the single curves (sum of area between two curves equals the total power loss).
    - Plot the complete energy lost at each part of the efficiency chain in this plot as text field.

4.2.3 Result:

After running this section of your live script, your table should now have some new values appended like the engine speed in rpm or the mechanical power and all the energy levels of the efficiency chain.
There should be a data struct, that holds the mean vehicle speed and the total energies lost at different stages of the efficiency chain.

4.3 Taking care of correct energy flow direction

4.3.1 Task description

For a better understanding of the correct energy flow and how much energy can be recuperated, a differentiation of the energy flow must be performed. Consider the sign of the torque for identifying the two modes:

- Negative torque implies recuperation (charging the battery).
- Positive torque means acceleration (discharging the battery).

- Calculate the total equivalent chain efficiency (combining all necessary efficiencies together).
- Compute the recuperated and demanded energy separately (two new columns in your table).
  - Use the torque to decide if energy was demanded, recuperated or engine was in idle mode.
  - Whenever energy was demanded, the recuperation energy must be zero and vice versa.
- Calculate the overall energy used during this drive cycle and save it to your data struct.
- Calculate the overall energy recuperated during this drive cycle and save it to your data struct.
- Calculate the money saved in the next recharging process (= overall energy recuperated * € / kWh), assuming a price of:
  - approx. 30 ct. / kWh (price for slow charging (AC charging)) and save it to your data struct.
  - approx. 40 ct. / kWh (price for fast charging (DC charging)) and save it to your data struct.
- Calculate the following SOC levels of the vehicle, assuming the vehicle was fully charged at test start:
  - SOC after drive cycle finished with energy recuperation and save it to your data struct.
  - SOC after drive cycle finished without energy recuperation and save it to your data struct.

4.3.2 Presenting your results

- Plot the two energy profiles (demanded / recuperated) together with the vehicle speed using the subplot feature.
- Display the final data struct in your live script. Be sure to add explanations on how this data can be interpreted within your script. Use the “Live Editor” Control possibilities or check-out the “Insert” tab for different visualisation options.

4.3.3 Result

After running this live script section, you should have two new columns in your table and four new values in your data struct. A new figure shows the correct energy flow (demanding / recuperating) which is aligned with the vehicle speed. Your final data struct is printed on the screen showing all our calculations stored during this exercise.

5 REPORT

- The report for this training unit is your **Matlab live script** including all your **functions** that you additionally created. All the files that are necessary to run your script shall be uploaded via Moodle within on zipped folder.
- Make sure to use explain your findings properly in your live script. Use the “**Publish**” tab of the live script editor, to be able to format your text better. Divide your script in sections as mentioned in the task description above.
- Be sure to check the correct function of your live script before uploading it to Moodle. If your script **does not work**, you **will not receive any points** for this training unit.
- Make sure, that the script shows **your own work** on this training unit task. Using solutions from your fellow students will result in zero points for you and your fellow students.

- **Caution (!)** be sure to check the **due date** that is set for uploading the files on Moodle!