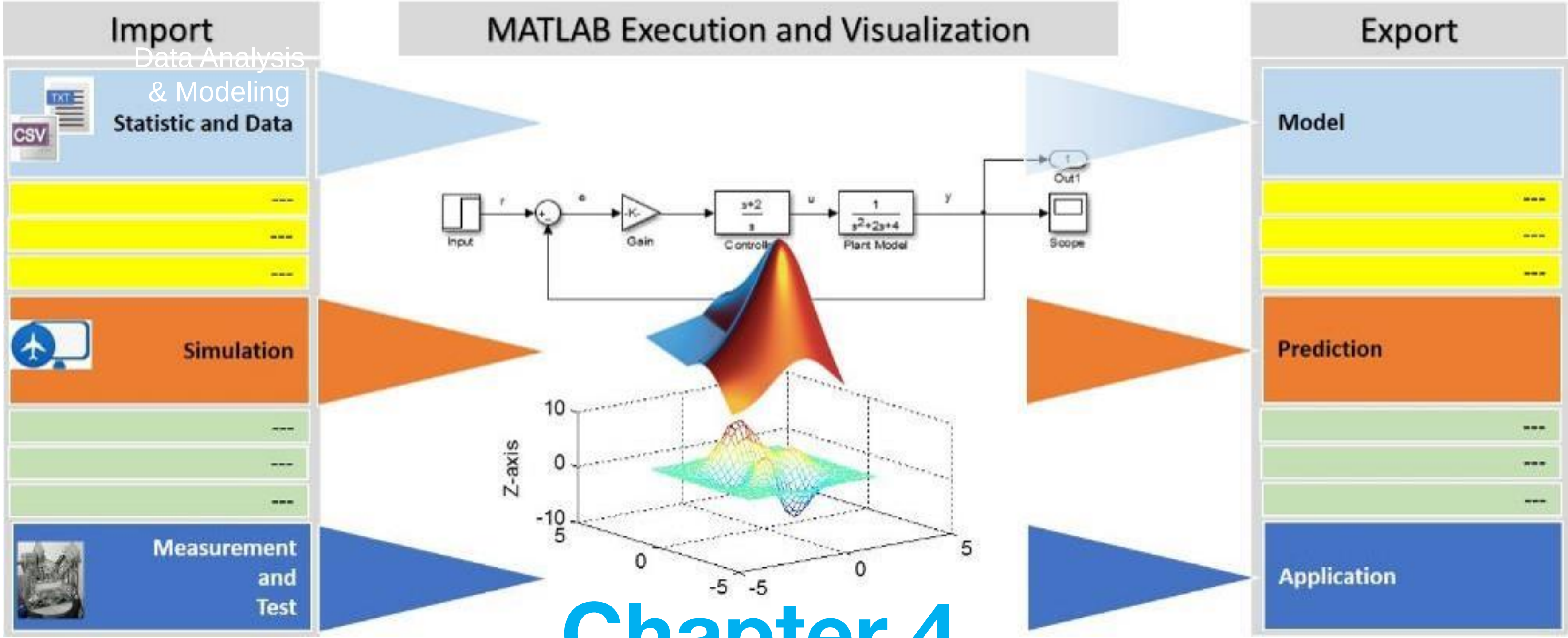


Chapter 4

MATLAB Data Handling and Visualization

- Exercises -

Exercise 1 – API Integration and Data Export

1. Complete the function ***GetNextTrain***, that loads train information from mat file and stores them in the output file ***NextTrain.txt***.

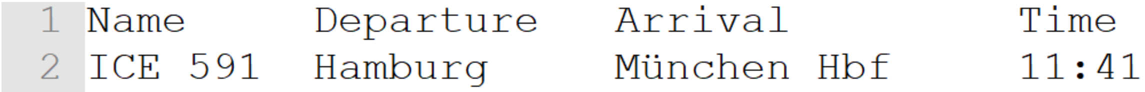
Steps to do:

a) Load the data file

b) Find the next train whose origin matches *departStation*, neglect possible intermediate stops

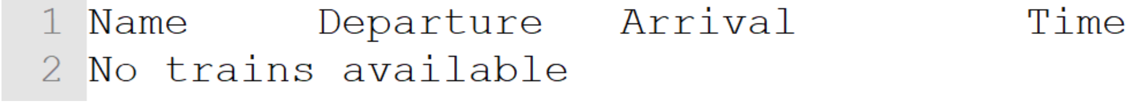
c) Store *trainName, arrivalStation* and *arrivalTime* of this certain train

d) Store the train information with a suitable header in the output file *NextTrain.txt*



```
Name      Departure   Arrival          Time
ICE 591   Hamburg     München Hbf      11:41
```

e) If no train was found, an error message should be written into the output file



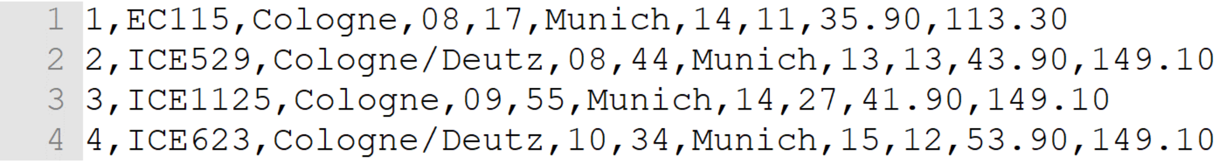
```
Name      Departure   Arrival          Time
No trains available
```

Exercise 2 – Import Data

2. The script ***AnalyzeTrains*** should analyses the ticket price and travel time of a certain train connection:

i. Import the train data from the file ***cologne_munich.txt***

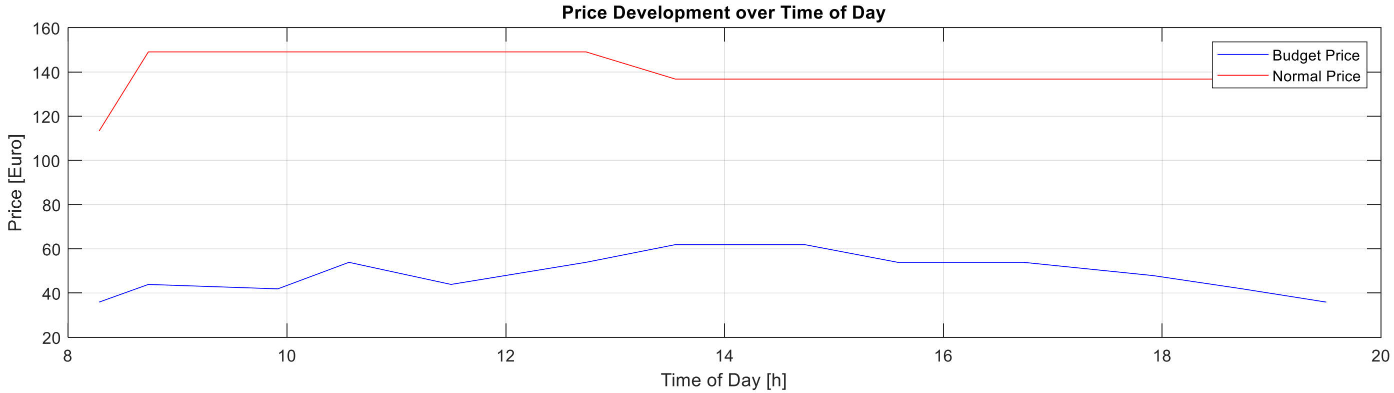
Data format: *Index, Train, Departure, hh, ss, Arrival, hh, ss, Budget Price, Normal Price*



```
1,EC115,Cologne,08,17,Munich,14,11,35.90,113.30
2,ICE529,Cologne/Deutz,08,44,Munich,13,13,43.90,149.10
3,ICE1125,Cologne,09,55,Munich,14,27,41.90,149.10
4,ICE623,Cologne/Deutz,10,34,Munich,15,12,53.90,149.10
```

ii. Create one subplot that shows the price development over time of day for the budget price (blue line) and normal price (red line)

- Give the plot a title and label the axes
- Create a legend for the budget and normal price

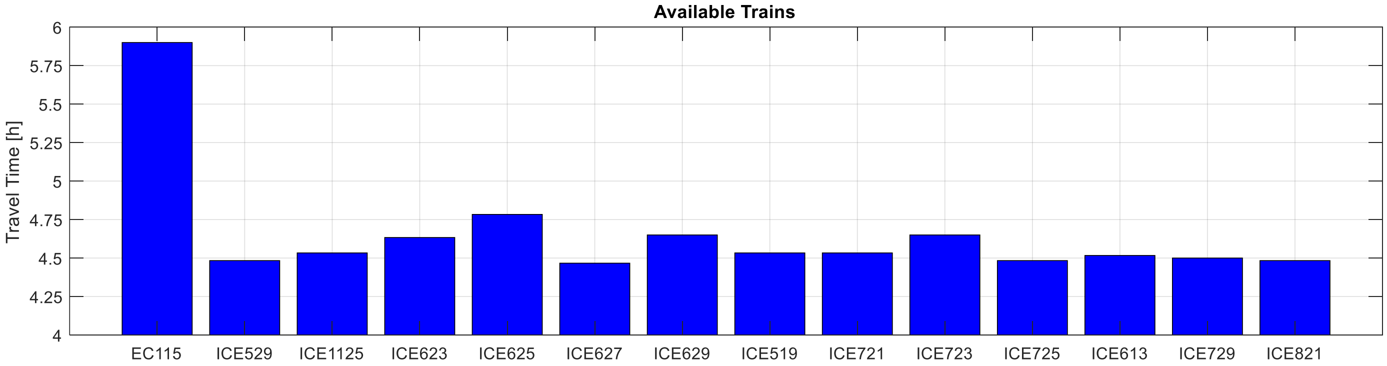


Exercise 2 – Plot Data

2. The script ***AnalyzeTrains*** should analyses the ticket price and travel time of a certain train connection:

   iii. Create a second subplot containing a bar graph that shows the travel time of each train

   - Give the plot a title and label the axes
   - The bars should be named with the appropriate train numbers
   - The y-values should be limited between 4 hours and 6 hours with an increment of 15min



Exercise 3 – GUI Syntax

GUI Plot Syntax

- Plot the data *x_values* over *y_values* in the figure *app.plot*

```
>> plot(app.plot, x_values, y_values, 'b-')
```

- Add a title, label and a legend to the figure *app.plot*

```
>> app.plot.Title.String = 'Title';
>> app.plot.XLabel.String = 'Label';
>> legend(app.plot, 'Curve 1', 'Curve 2')
```

- Retain *app.plot* when adding new plots

```
>> hold(app.plot, 'on')
```

- Display grid lines in the figure *app.plot*

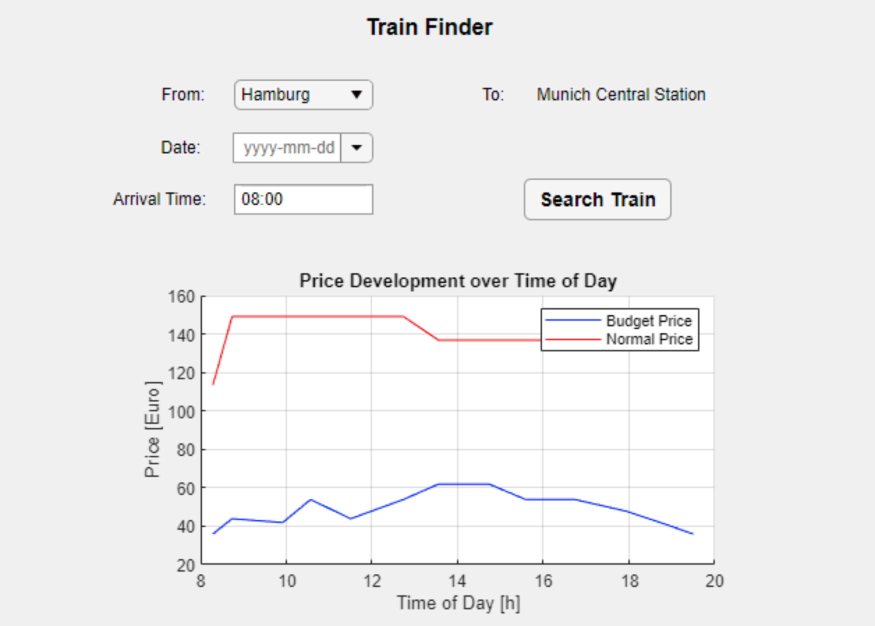
```
>> grid(app.plot, 'on')
```

Exercise 3 – GUI

3. Create a GUI that searches the **next** available train between a custom ***departure station*** and ***Munich Central Station***. The train details should be stored in the file ***NextTrain.txt***.

   i. By starting the GUI, the ***first plot of "exercise 2"*** should be integrated

   ii. If the button **"Search Train"** is pressed, the function ***GetNextTrain*** should be called with the values of the GUI fields



Train Finder

From: Hamburg

To: Munich Central Station

Date: yyyy-mm-dd

Arrival Time: 08:00

Exercise 3 – GUI

Steps to do:

a) Create the **GUI Design** for Departures from ‘**Hamburg, Berlin** and **Dortmund’**

b) Callback **app.*SearchButton***: Store the values of the GUI and call the function *GetNextTrain*

c) Callback ***app.UIFigure*** : Insert the code of “exercise 2”

d) Adapt the plot commands to fit the GUI syntax