Predictive Control and Optimization
Exercises

Prof. Dr.-Ing. O. Nelles
Automatic Control – Mechatronics
University of Siegen

12.04.2022

Task 1: Least Squares - Linear Regression

a) In Tab. 1, you can see the power and consumption data of different cars.

Table 1: Power and consumption data of different cars

Car	Power [PS]	Consumption [l/100 km]
Opel Corsa 1.6 Turbo OPC	207	7.5
Honda Civic Type R2.0	310	7.3
BMW X6 M	575	11.1
Porsche Panamera GTS	440	10.3
BMW M6 Coupé M6	560	13.9
Jaguar F-Type R Coupé R	505	10.7
BMW M4 Coupé M4	431	8.3
Toyota GT86	200	7.1
Ferrari 488 GTB Coupé	670	11.4

Plot the data with power on the x-axis and consumption on the y-axis.

b) It is assumed that there is a linear (affine) relationship between power and consumption. To describe this relationship, a linear regression is to be performed. Set up the regression matrix $\underline{X}$ and the output vector $\underline{y}$.

c) Calculate the optimal parameters $\underline{\theta}$ using the least squares (LS) equation. Plot the resulting regression line into the data plot.

Implementation: The most efficient way in MATLAB to calculate a inverse of a matrix is the backslash operator

Example for a square matrix $\underline{B}$: $\underline{a} = \underline{B}^{-1}\underline{c}$

```
        a = B \ c;
```

d) Reformulate the linear regression problem into the standard formulation of a quadratic program (without constraints). Some helpful remarks can found in the lecture notes. Create a anonymous function with the name f_quad of the standard cost function.

Implementation: Anonymous functions are important in MATLAB while dealing with optimization problems. They can be defined in the following way:

Example function: $f(\underline{\theta}) = 5 \cdot \theta_1 + 7 \cdot \theta_2^2$

```
        f = @(theta) 5 * theta(1) + 7 * theta(2)^2;
```

e) Plot the resulting quadratic loss function in the parameter space (θ_1, θ_2)

Implementation: To plot a two-dimensional function different options exist:

```
        surf(X,Y,Z);
        contour(X,Y,Z);
```

`X`,`Y` and `Z` must be matrices. `X` and `Y` consists of the variables the function depends on and `Z` contains the corresponding output of the function.

To create such matrices the following functions are helpful:

```
        [X,Y] = meshgrid(range1,range2);
        x = reshape(X,dimx,dimy);
```

Task 2: Least Squares - Linear Regression 2

This task deals with data from different start-up companies (see `50_Startups.csv`). The influence of different expenses on the profit of the company shall be investigated. R&D and marketing spending are to be examined as potential influencing factors.

a) Calculate the parameters of two individual linear regression models – one for each of the expenses (R&D and marketing). Calculate the error between the resulting model and the collected data. Use the root-mean-squared-error (RMSE) for this purpose.

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}(i) - y(i))^2}$$

b) Now a linear model with two inputs shall be estimated. Use R&D and marketing investments as inputs of a two dimensional model. Calculate the parameters of the regression model. Calculate the error between the resulting model and the collected data. Of which dimension is the resulting loss function in standard formulation?

c) Plot the resulting model and the data points.

Implementation: To plot data points in a three-dimensional space, use the following MATLAB command:

```
        plot3(x,y,z,'o');
```

d) Corresponding to the calculated model, which of the expenses promises a higher profit?

Task 3: Quadratic Program (QP)

In this task, the following quadratic cost function is investigated:

$$J(\underline{\theta}) = \frac{1}{2}\underline{\theta}^T \underbrace{\begin{bmatrix} 3 & 0.5 \\ 0.5 & 1 \end{bmatrix}}_{\underline{H}} \underline{\theta} + \underbrace{\begin{bmatrix} -5 \\ -2 \end{bmatrix}^T}_{\underline{g}^T} \underline{\theta}\,. \tag{1}$$

a) Define an anonymous function of the cost function and plot it in the parameter space. Take a range of $[-10 \dots 10]$ for both θ_1 and θ_2.

b) Optimize the unconstrained cost function with the least squares (LS) solution.

c) Apart from the analytical least squares solution, it is also possible to optimize the loss function numerically. MATLAB provides different tools to solve such a problem. Use the following two methods to solve the unconstrained loss function numerically. Are the solutions of all of the algorithms the same? What could be a reason for possible deviations?

```
        quadprog(H,g,A,b);
        fmincon(@functionHandle,theta_0,A,b);
        % A and b can be choosen empty [] for unconstrained
           optimization
```

d) Now, constraints shall be taken into account during the optimization. The following constraints shall be satisfied.

$$\theta_1 \geq 3 \tag{2}$$
$$0.5\theta_1 - \theta_2 \leq 5 \tag{3}$$
$$-\theta_1 - \theta_2 \geq -6 \tag{4}$$

Plot these constraints into the same plot as the loss function before.

e) Reformulate the constraints into the following format.

$$\underline{A}\,\underline{\theta} - \underline{b} \leq 0 \tag{5}$$

f) Now the constraints will be taken into account for optimization. Use the two methods given above to solve the constraint problem. Plot the optimal $\underline{\theta}$ into the plot with the constraints and cost function.

g) Which of the constraints are active and therefore directly influence the optimal solution?

Task 4: Model Predictive Control: DMC - Part I

1.) Change the prediction horizon N_p, the control horizon N_u and the weighting factor λ. How do these factors influence the result/performance of the control?

2.) So far, only the analytical calculation of the optimal manipulated variable sequence has been implemented. Extend the source code to use also constraints during the Optimization. The constraints are defined in the variable `Constraints`. Proceed as follows:

1) Define a cost function
   `I = @(du_plus) ....`
   Hint: Use for $\underline{H}^{+}_{N_p \neq N_u} = \underline{H}^{+}\tilde{\underline{H}}$
2) Optimize `I` using `fmincon` (first without constraints)
   Hint: Set `useConstraints= 1;(line 16)` *if you want to check your code.*
3) Define the matrices `A, b`. For this use the values in `Constraints`:
   - `Constraints.u_max`
   - `Constraints.u_min`
   - `Constraints.du_max`
   - `Constraints.du_min`

   Hint: You can use the functions `eye(), ones(), zeros(), tril(), triu()`.
4) Optimize `I` using `fmincon` with constraints.
5) Change the constraints. How do these factors influence the control?
6) What happens with the controlled variable when the `u_max` constraint is too low and why?
   Hint: Rewriting of A, b, example $N_u = 2$:
   Notation: $A \cdot \Delta\underline{u}^{+} < b$

$$\begin{aligned} u(k-1) + \Delta u(k) &< u_{\max} \\ u(k-1) + \Delta u(k) + \Delta u(k+1) &< u_{\max} \end{aligned}$$

$$A = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, b = \begin{bmatrix} 1 \\ 1 \end{bmatrix}(u_{\max} - u(k-1))$$

$$\begin{aligned} u(k-1) + \Delta u(k) &> u_{\min} \\ u(k-1) + \Delta u(k) + \Delta u(k+1) &> u_{\min} \end{aligned}$$

$$A = -\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, b = -\begin{bmatrix} 1 \\ 1 \end{bmatrix}(u_{\max} - u(k-1))$$

$$\begin{aligned} \Delta u(k) &> \Delta u_{\min} \\ \Delta u(k+1) &> \Delta u_{\min} \end{aligned}$$

$A = ?, b = ?$

$$\begin{aligned} \Delta u(k) &< \Delta u_{\max} \\ \Delta u(k+1) &< \Delta u_{\max} \end{aligned}$$

$A = ?, b = ?$

3.) Make some changes that the model is not equal to the process, i.e. change some coefficients of the transfer function of the model. How the MPC performance is influenced? Why the reference value can be reached, even with a wrong model?

4.) Add output noise (white Gaussian noise) on the measurement of the controlled variable. Use the predefined variance variable `sigma2`. How does the figure change?

Task 5: Model Predictive Control: State Space - Part II

The goal of this task is to implement a model predictive controller (MPC) in state space. In this exercise we focus on the control of a lane change scenario. As a car model a single-track model is used, where only the lateral direction (Y-position) should be controlled. The steering angle is used as input u for the car model and a constant velocity in longitudinal direction is assumed. The task is to implement an MPC which optimizes the steering angle to follow a given reference trajectory.

1.) Get familiar with the code. How does the car behave, if a step signal is given to the manipulated variable (steering angle)? Is this behavior also visible in the given scopes?

2.) Now we want to control the Y-position of the car. For this purpose, add an `MPC Controller` block in SIMULINK. In the script `MPC_init` we already defined an MPC Controller named `MPC`. Also make sure that we do not have any additional input ports, since we do not measure any disturbance. Please connect all the input and output ports of the MPC block. How does the MPC behave, if only the first entry of the reference is given to the MPC. Now also the future reference values should be used in the MPC block. How does the behavior change?

3.) Add the estimated states in the MPC (this is done via Kalman filter) as an additional output port of the MPC Controller and compare them to the real ones of the system. Can you see any difference?

4.) Open the MATLAB script `MPC_init`. A lane change maneuver to $Y = 4$ should be performed. Do all necessary changes in the reference value.

5.) Implement a lane limit in Y-direction of $2\,\mathrm{m}$. Also slow down the lane change maneuver.

6.) What happens, if you increase the control horizon? Try it in the simulation.

7.) Add measurement noise to the Y-position. This measured Y-position should also be used in the MPC controller (`Band-Limited White Noise` block). Try different noise powers, start with 0.0001. How robust is the MPC?

8.) Now we want to improve the lane change trajectory by changing the reference trajectory. Implement for example filtering (`filter`), ramp (`linspace`), etc.

Task 6: Optimization: Nonlinear Problems – Part I

The Rosenbrock function has the form

$$f(\underline{\theta}) = f(\theta_1, \theta_2) = (a - \theta_1)^2 + b(\theta_2 - \theta_1^2)^2 \,,$$

where commonly $a = 1$ and $b = 100$.

1.) Visualize the Rosenbrock function in the interval $\theta_1 \times \theta_2 \in [-2.25, 2.25] \times [-1.25, 3.25]$. For later ease of use, define the Rosenbrock function as anonymous function:

```
rosenbrock = @(theta) (1-theta(1,:)).^2+100*(theta(2,:)-theta(1,:)
   .^2).^2;
```

The function can then be evaluated as:

```
y = rosenbrock([theta_1;theta_2]);
```

2.) Search for the minimum of the Rosenbrock function in the interval $\theta_1 \times \theta_2 \in [-2.25, 2.25] \times [-1.25, 3.25]$ with a) grid search and b) random search. Analyze your obtained result with respect to different numbers of (grid) points that you choose. For the random search, use uniformly distributed data with `rand()` and shift the data points according to the interval.

3.) Implement the steepest gradient descent method. For that, first calculate the gradient $\underline{g}(\underline{\theta})$ analytically and implement it as anonymous function in MATLAB. Use a step size of $\eta = 0.002$, start at the initial parameter vector $\underline{\theta}_0 = \begin{bmatrix} -1.5 \\ 0.5 \end{bmatrix}$, and run the algorithm for 1000 iterations. Analyze the convergence behavior of steepest descent when the initial vector, the step size and the number of iterations changes.

Optional: Implement a learning rate scheduling with an exponentially decaying learning rate.

4.) Implement the Newton method. First, calculate analytically the Hessian matrix $\underline{H}(\underline{\theta})$ and implement it as anonymous function in MATLAB. Use a the initial parameter vector $\underline{\theta}_0 = \begin{bmatrix} -1.5 \\ 0.5 \end{bmatrix}$, and run the algorithm for 5 iterations. Analyze the convergence behavior of the Newton method when the initial vector or the number of iterations changes.

Task 7: Optimization: Nonlinear Problems – Part II

The Rosenbrock function has the form

$$f(\underline{\theta}) = f(\theta_1, \theta_2) = (a - \theta_1)^2 + b(\theta_2 - \theta_1^2)^2 \,,$$

where commonly $a = 1$ and $b = 100$.

1.) Find the minimum of the Rosenbrock function using MATLAB's `fminsearch` function. `fminserach` uses the Nelder-Mead-Algorithm, which is a special case of the downhill simplex method. Initialize `fminserach` with the parameter vector $\underline{\theta}_0 = \begin{bmatrix} -1.5 \\ 0.5 \end{bmatrix}$.

2.) Find the minimum of the Rosenbrock function using MATLAB's `fminunc` function. Initialize fminunc with $\underline{\theta}_0 = \begin{bmatrix} -1.5 \\ 0.5 \end{bmatrix}$.

3.) Now let's assume that we want to find the minimum of the Rosenbrock function under the constraint that the solution has to lie inside a circle with the center $\underline{c} = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$ and radius $r = 1.5$ in parameter space. How does the minimum change, when the location and radius of the circle changes? Incorporate in a second step an additional equality constraint $\theta_2 - \theta_1 = 0$. Use the MATLAB function `fmincon` to solve this problem. The constraints can be implemented with the `nonlcon` option. You can use the `deal` function to return two output arguments within an anonymous function

```
    ineq_c = @(theta) deal(⟨inequality constraint⟩,0);
```

4.) Use MATLAB's predifined dataset polydata (get data with `load polydata`) and fit to it a third-order polynomal function

$$y = \theta_1 x^3 + \theta_2 x^2 + \theta_3 x + \theta_4$$

using `fminunc`. For this, use the vectors `x` and `y` from the dataset. In a second step, use `fmincon` to incorporate the constraint that the squared sum of all parameters $\sum_{i=1}^{n_\theta} \theta_i^2 < t$. How does the function y differ for different values of t?