ECE-GY 9243 / ME-GY 7973
Optimal and learning control for robotics

Exercise series 1

For questions requesting a written answer, please provide a detailed explanation. Typesetted answers (e.g. using LaTeX[1]) are strongly preferred. Include plots where requested, either in a Jupyter Notebook or in the typesetted answers. For questions requesting a software implementation, please provide your code in a python file or in a Jupyter Notebook such that it can be run directly. Include comments explaining how the functions work and how the code should be run if necessary. Any piece of code that does not run out of the box or does not contain instructions to execute it will be considered invalid.

Exercise 1 [Convex optimization with linear equalities]

Consider the following optimization problem

$$\min_{\mathbf{x}} \frac{1}{2}\mathbf{x}^T\mathbf{Q}\mathbf{x} \tag{1}$$

$$\text{subject to } \mathbf{A}\mathbf{x} = \mathbf{b} \tag{2}$$

where $\mathbf{Q} \in \mathbb{R}^{n\times n} > 0$ and $\mathbf{A} \in \mathbb{R}^{m\times n}$ is full rank with $m < n$ and $\mathbf{b} \in \mathbb{R}^m$ is an arbitrary vector.

- Write the Lagrangian of the optimization problem as well as the KKT conditions for optimality
- Solve the KKT system and find the optimal Lagrange multipliers as a function of $\mathbf{Q}$, $\mathbf{A}$ and $\mathbf{b}$.
- Use the above results to compute the minimum the function

$$\frac{1}{2}\mathbf{x}^T \begin{bmatrix} 10 & 1 & 0 \\ 1 & 100 & 2 \\ 0 & 2 & 1 \end{bmatrix} \mathbf{x} \tag{3}$$

  under the constraint that the sum of the components of the vector $\mathbf{x} \in \mathbb{R}^3$ should be equal to 1. What is the value of $\mathbf{x}$ and of the Lagrange multipliers? Verify that the constraint is indeed satisfied for your result. Hint: use python for all your numerical computation.[2]

Exercise 2 [Netwon's method]

In the companion Jupyter Notebook called *Gradient Descent*, we provide an example of gradient descent.

- Use this example to implement Newton's method to find the minimum of a function.

[1] `https://en.wikibooks.org/wiki/LaTeX`, NYU provides access to Overleaf to all the community `https://www.overleaf.com/edu/nyu`

[2] in particular the numpy function solve `https://docs.scipy.org/doc/numpy/reference/generated/numpy.linalg.solve.html?highlight=solve#numpy.linalg.solve` is generally more robust and efficient when computing the solution of $Ax = b$ that taking the inverse of A explicitly.

- Use the algorithm to compute the minimum of the functions
  - $f(x) = \mathrm{e}^{\frac{x^2}{100}}$ with $x_0 = 6.5$ as an initial guess
  - $f(\mathbf{x}) = \mathbf{x}^T \begin{bmatrix} 10 & 1 \\ 1 & 10 \end{bmatrix} \mathbf{x}$ with $\mathbf{x}_0 = \begin{bmatrix} -9, -9 \end{bmatrix}$ as an initial guess.
- Compare the convergence results with the gradient descent algorithm. In particular, provide similar plots as the ones shown in the Jupyter Notebook for the Newton method.

Exercise 3 [Linear Least Squares]

In the companion Jupyter Notebook called *Linear Least Square Problems*, we describe how to fit arbitrary polynomial functions from data.

- Use the examples provided in the Notebook and especially the *do_regression* function to find the best polynomial fit for the dataset given in the file *regression_dataset* and answer the following questions
  - What is the order of the polynomial that best fits the data? Why?
  - What are the polynomial coefficient?
  - Plot the function you found and the data from the dataset. Does the fit look good?

  Note that you can read the data file into a numpy array by calling *data = np.loadtxt('regression_dataset')*
- It is in fact possible to functions to the data as long as the function is of the form $y = \sum_k a_k f_k(x)$ where f_k is an arbitrary function. Following the same reasoning as for polynomial fitting, we can write the problem as a quadratic program. Assume now that we want to fit periodic functions of the form

$$y = a_0 + \sum_{k=1}^{K} a_k \cos(kT2\pi x) + b_k \sin(kT2\pi x) \tag{4}$$

  where T is the period of the function, which we assume known.
  - Write down the optimization problem that allows to find the coefficients a_k and b_k.
  - Using the *do_regression* function as a model, implement a function that solves the problem for periodic functions.
  - Use this function find the best fit for the data stored in the file *regression_dataset2* assuming that the period of the function is $T = 1$.
  - Plot the function you found and the data from the dataset. Does the fit look good?