ROB6323: Reinforcement learning and optimal control for robotics

Exercise series 2

For questions requesting a written answer, please provide a detailed explanation and typeset answers (e.g. using LaTeX[1]). Include plots where requested in the answers (or in a Jupyter notebook where relevant). For questions requesting a software implementation, please provide your code in runnable Jupyter Notebook. Include comments explaining how the functions work and how the code should be run if necessary. Code that does not run out of the box will be considered invalid.

Exercise 1 [20 points]

Consider the optimal control problem of Series 1 - Exercise 4 (control of a drone).

- Reusing the notebook of Series 1, write code to solve the same problem when the control is limited to $|5|$ for both rotors and the horizontal and vertical velocities are bounded to $2\mathrm{m}\cdot\mathrm{s}^{-1}$. To solve the resulting QP, use the *cvxopt* solver available with the *qpsolvers* library[2]
- Show plots of all the states of the robot as a function of time
- Show plots of the optimal control as a function of time
- Compare the results with the results of Series 1 where no bounds were used.

Exercise 2 [40 points]

We have seen in class two methods to minimize an arbitrary (twice continuously differentiable) function: gradient descent and Newton's method. Implement in a Jupyter Notebook both gradient descent and Newton's method using a backtracking line search where $c = 10^{-4}$. Use these algorithms to minimize the following functions:

- $-\mathrm{e}^{-(x-1)^2}$, starting with $x_0 = 0$
- $(1-x)^2 + 100(y-x^2)^2$, starting with $x_0 = y_0 = 1.2$
- $x^T \begin{bmatrix} 3 & 1 \\ 1 & 3 \end{bmatrix} x + \begin{bmatrix} -1 & 1 \end{bmatrix} x$, staring with $x_0 = \begin{pmatrix} 10 \\ 10 \end{pmatrix}$
- $\frac{1}{2} x^T \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 4 \end{bmatrix} x - \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} x$, staring with $x_0 = \begin{pmatrix} -10 \\ -10 \\ -10 \end{pmatrix}$

For each function:

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

[2] `https://pypi.org/project/qpsolvers/`

- Show the convergence of each method by plotting (in the same plot) the norm of the distance to the optimum as a function of the number of iterations
- Plot the value of α in the backtracking line search as a function of the number of iterations
- Print the optimum and explain which criteria you used to stop the algorithms

Do not invert the Hessian matrix, but instead use the solve function from NumPy (cf. Series 1) to find a step p_k that satisfies $\nabla^2 f(x_k)p_k = -\nabla f(x_k)$. Note also that when the Hessian is not positive definite, there will be an issue and you can add a small multiple of the identity to it until it is positive definite, i.e. use $\nabla^2 f(x_k) + \gamma I$ where γ is small instead of the Hessian.

Exercise 3 [40 points]

Implementation of a SQP for nonlinear optimal control. Please answer the questions in the notebook *exercise 3_pendulum.ipynb*.