Reinforcement learning and Optimal control for Robotics (ROB-GY 6323)

New York University, Fall 2024

Due on 10/27, 11:59 PM

Name: Raman Kumar Jha
NYU ID: N13866145

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 $\text{m} \cdot \text{s}^{-1}$. 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.

[1]https://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/

Solution: The solution to all the first three questions has been provided in the notebook named Series2_Question1.ipynb.

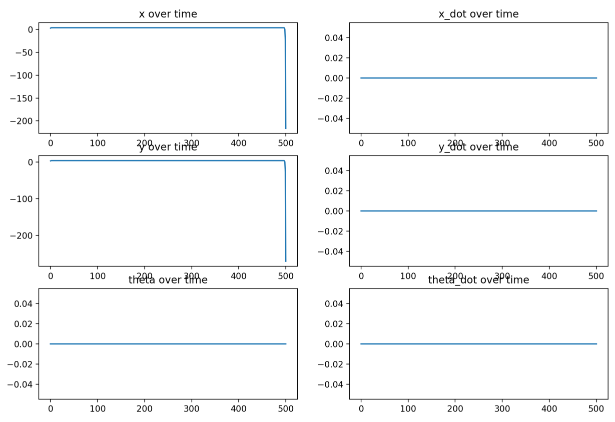

Figure 1: Contour plot of $f_3(x, y)$

1. Dynamics: The first plot named Figure 1 with no constraints shows a static system with no dynamics, while the second image named Figure 2 depicts a dynamic system with active changes in the environment.
2. State Changes: The second plot reflects more realistic behavior with variations in state variables and their rates of change over time.

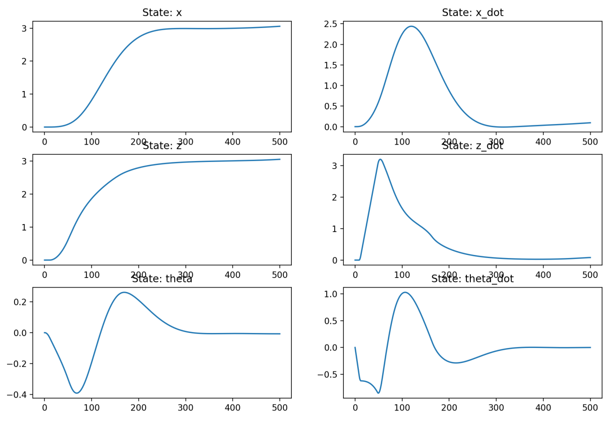

Figure 2: Contour plot of $f_3(x, y)$

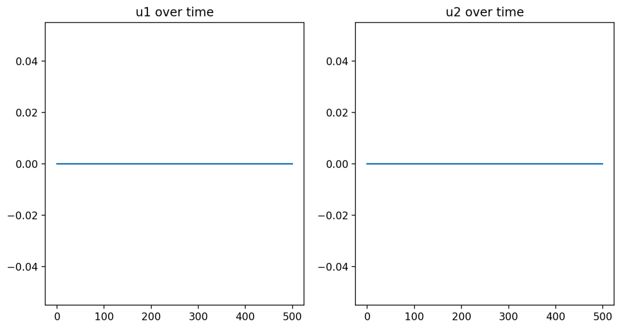

Figure 3: Contour plot of $f_3(x, y)$

1. Dynamics: The first plot named Figure 3 with no constraints shows a static system with no dynamics, while the second image named Figure 4 depicts a dynamic system with active changes in the environment.
2. State Changes: The second plot reflects more realistic behavior with variations in optimal control variables and their rates of change over time.

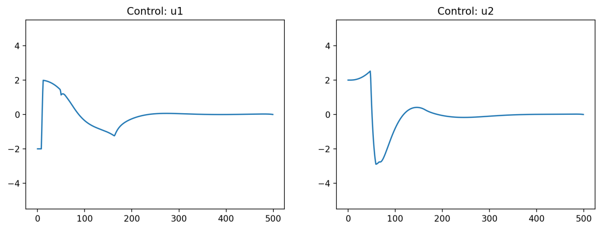

Figure 4: Contour plot of $f_3(x, y)$

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:

- $-e^{-(x-1)^2}$, starting with $x_0 = 0$
- $(1-x)^2 + 100\left(y - x^2\right)^2$, 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 $x_0 = \binom{10}{10}$

- $\frac{1}{2}x^T \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 4 \end{bmatrix} x - \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} x$, staring with $x_0 = \begin{pmatrix} -10 \\ -10 \\ -10 \end{pmatrix}$

For each function:

- 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.

Solution: The solution to all the questions has been provided in the notebook named Series2_Question2.ipynb.

The code for question 2 contains code for functions named backtracking_line_search, gradient_descent_step, newton_step, and minimize for implementing the minimization using the Gradient Descent and Newton method. For each function, there is a cell dedicated to providing the plot of convergence of each method, as well as the value of α has been shown in the notebook.

The optimum for Gradient Descent, as well as the Newton method, has been printed at the end of each cell of the function in the notebook.
The stopping criterion: for both algorithms is based on the norm of the gradient being less than a tolerance of 1e-6.

Exercise 3 [40 points]

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

The solution to all the questions has been provided in the notebook named exercise 3_pendulum_solution.ipynb.

1. Algorithm Outline

1. **Initialization**: - An initial guess for x (state and control trajectory) and Lagrange multipliers λ for constraints, Convergence tolerance and maximum

iterations.
2. **Iterate until convergence or max iterations**:
Step 1: The gradients of the running cost and constraints at the current guess $\bar{x}$.
Step 2: The Hessian of the running cost.
Step 3: Linearization of the constraints at the current $\bar{x}$.
Step 4: Formulation the KKT system with:
- First-order optimality condition: gradient of the Lagrangian should be zero.
- Primal feasibility: constraints satisfied at each step.
Step 5: KKT system to obtain the step Δx and multiplier update $\Delta\lambda$.
Step 6: Filter line search to decide step acceptance based on cost reduction and constraint violation reduction.
Step 7: Updating the guess for x and λ.
Step 8: Termination conditions: - If KKT conditions $\nabla_x L$ and $\nabla_\lambda L$ are close to zero.
- Or if the maximum iterations are reached.

2. Gradient of Running Cost

Let the running cost be represented by $f(x)$, with x being the stacked state and control variables over the time horizon.

The gradient of the running cost, $\nabla f(x)$ at a given guess $\bar{x}$, is calculated as:

$$\nabla f(\bar{x}) = \begin{bmatrix} \frac{\partial f}{\partial \theta_0} & \frac{\partial f}{\partial \omega_0} & \frac{\partial f}{\partial u_0} & \cdots & \frac{\partial f}{\partial \theta_{300}} & \frac{\partial f}{\partial \omega_{300}} & \frac{\partial f}{\partial u_{300}} \end{bmatrix}^T$$

3. Hessian of Running Cost

The Hessian of the running cost H_f at a given guess $\bar{x}$ can be expressed as a block matrix:

$$H_f(\bar{x}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_0^2} & \cdots & \frac{\partial^2 f}{\partial x_0 \partial x_{300}} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_{300} \partial x_0} & \cdots & \frac{\partial^2 f}{\partial x_{300}^2} \end{bmatrix}$$

4. Linear Approximation of Constraints

Given a constraint $c(x) = 0$, the linearization at $\bar{x}$ can be written as:

$$G(\bar{x})\Delta x = g(\bar{x})$$

where: - $G(\bar{x})$ is the Jacobian of the constraints evaluated at $\bar{x}$, - $g(\bar{x}) = c(\bar{x})$.

5. Construct Inner Linear KKT System

Formation of the KKT system with blocks:

$$\begin{bmatrix} H_f & G^T \\ G & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \end{bmatrix} = \begin{bmatrix} -\nabla f \\ -g \end{bmatrix}$$

6. Constraint Violation Function

A function to return the sum of the absolute values of each constraint, providing a measure of step constraint violation.

7. Filter Line Search

A filter line search, so that:
It Accepts a step if it reduces both the objective function and the constraint violation.
If neither is reduced, reduce the step size and retry until improvement or reaching a minimum step size threshold.

8. Termination Condition

Terminate the algorithm when either:
1. The KKT conditions $\nabla_x L$ and $\nabla_\lambda L$ are close to zero, i.e., 10^{-4}.
2. The maximum iteration count (i.e., 100) is reached.