Assignment 2

Numerical Optimization / Optimization for CS WS2021

Christian Kopf, christian.kopf@icg.tugraz.at
Lukas Erlbacher, lukas.erlbacher@student.tugraz.at
Thomas Wedenig, thomas.wedenig@student.tugraz.at

December 3, 2021

Deadline: Dec 7$^{\text{th}}$, 2021 at 23:59

Submission: Upload your report and your implementation to the TeachCenter. Please use the provided framework-file for your implementation. Make sure that the total size of your submission does not exceed 50MB. Include **all** of the following files in your submission:

- `report.pdf`: This file includes your notes for the individual tasks. Keep in mind that we must be able to follow your calculation process. Thus, it is not sufficient to only present the final results. You are allowed to submit hand written notes, however a compiled LaTeX document is preferred. In the first case, please ensure that your notes are well readable.
- `main.py`: This file includes your python code to solve the different tasks. Please only change the marked code sections. Also please follow the instructions defined in `main.py`.
- `figures.pdf`: This file is generated by running `main.py`. It includes a plot of all mandatory figures on separate pdf pages. Hence, you do not have to embed the plots in your report.

1 Lagrange Multiplier Problem (12 P.)

Given the following constrained optimization problems over $\mathbf{x} = (x_1, x_2)^{\mathrm{T}}$, compute the optimal solution using Lagrangian formulations.

a)
$$\min_{\mathbf{x}} \; x_2 - x_1 \quad \text{s.t.} \quad x_1 \geq 4x_2, \quad x_2 = \frac{1}{10}x_1^2 - 3$$

b)
$$\min_{\mathbf{x}} \; \|\mathbf{x}\|_2^2 \quad \text{s.t.} \quad x_2 \geq 3 - x_1, \quad x_2 \geq 2$$

c)
$$\min_{\mathbf{x}} \; (x_1 - 1)^2 + x_1 x_2^2 - 2 \quad \text{s.t.} \quad x_1^2 + x_2^2 \leq 4$$

For each example - **In Python**:

1. Plot the level sets of the above functions using a (filled) contour plot[1].
2. Draw the equality-/inequality-constraints on top.
3. Add markers to all points which fulfill the KKT-conditions (computed with Pen & Paper). Highlight the overall optimal solution with a point in a different color.

For each example - **With Pen & Paper**:

4. Formulate the Lagrangian equation. As shown in the exercise.

[1] `https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.contourf.html`

5. Formulate the KKT-conditions.
6. Discuss every case with respect to the given constraints. Compute all possible solutions and identify valid ones.
7. Show which solution is the optimal one.

2 Lagrange Augmentation (8 P.)

Consider an optimization problem with linear constraints of the form

$$\min_{\mathbf{x}} \; f(\mathbf{x}) \quad \text{s.t.} \quad \mathbf{A}\mathbf{x} - \mathbf{b} = h(\mathbf{x}) = 0 \;, \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{b} \in \mathbb{R}^m$. Recall from the lecture that the Lagrangian function provides a lower bound to the original constrained optimization problem which can be written in the form of a min-max problem

$$\min_{\mathbf{x}} \{ \max_{\lambda} f(\mathbf{x}) + \lambda^{\mathrm{T}} h(\mathbf{x}) = \mathcal{L}(\mathbf{x}, \lambda) \} \;. \tag{2}$$

Due to the non-smoothness of the inner (maximization) problem w.r.t. $\mathbf{x}$, solving eq. (2) can be difficult. Instead it is also possible to solve for the optimal solution iteratively by augmenting the Lagrangian function and solving a sequence of unconstrained problems directly. This can be done by introducing state variables $\lambda_k \in \mathbb{R}^m$ and $\mathbf{x}_k \in \mathbb{R}^n$ at iteration k and augmenting the Lagrangian to read

$$\min_{\mathbf{x}_k} \{ \max_{\lambda_k} f(\mathbf{x}_k) + \lambda_k^{\mathrm{T}} h(\mathbf{x}_k) - \frac{1}{2\alpha} \|\lambda_k - \lambda_{k-1}\|_2^2 = \mathcal{L}_a(\mathbf{x}_k, \lambda_k, \alpha) \} \;, \tag{3}$$

with $\alpha > 0$ and λ_{k-1} denoting the prior state of λ. Maximizing eq. (3) w.r.t λ_k leads to

$$\min_{\mathbf{x}_k} f(\mathbf{x}_k) + \lambda_{k-1}^{\mathrm{T}} h(\mathbf{x}_k) + \frac{\alpha}{2} \|h(\mathbf{x}_k)\|_2^2 \;. \tag{4}$$

Algorithm 1: Iterative update procedure on augmented Lagrangian

Initialize λ_{k-1}
Choose $\alpha > 0$
for $k < K$ **do**
    Update $\mathbf{x}_k$ using the update rule computed in 2.3
    Update λ_k using the results from 2.2
end

Tasks: Given the optimization problem over $\mathbf{x} = (x_1, x_2)^{\mathrm{T}}$

$$\min_{\mathbf{x}} f(\mathbf{x}) = (x_1 - 1)^2 - x_1 x_2 \quad \text{s.t.} \quad -x_1 + 4 = x_2$$

With Pen & Paper:

1. Solve for $\mathbf{x}^*$ and λ^* using the exact method as in Task 1. State the KKT-conditions and compute the solution to the given problem.
2. Prove that eq. (3) leads to eq. (4) by solving the inner optimization problem.
3. Solve eq. (4) to attain an update rule in $\mathbf{x}$.

In Python:

4. Plot the level sets of the above function using a filled contour plot[2].
5. Draw the constraints on top.
6. Add a marker to the optimal solution which fulfills the KKT-conditions (computed with Pen & Paper).

[2] https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.contourf.html

7. Implement Algorithm 1 and compute the optimal solution iteratively. Choose any λ_{k-1} and a $\alpha > \frac{1}{2}$. Also set $K = 20$.
8. Plot the points $\mathbf{x}_k$ over the iterations
9. What is the effect of choosing a larger value for α?

3 Least Squares Fitting (5 P.)

Assume a given dataset comprised of N points (x_n, y_n, z_n), $n = 1, \ldots, N$. The data indicates that it can be modelled by a polynomial function $f : \mathbb{R}^2 \mapsto \mathbb{R}$. We thus assume that

$$z_n \approx f(x_n, y_n) = \sum_{i=0}^{d} \sum_{j=0}^{d} w_{ij} x_n^i y_n^j \quad n = 1, ..., N,$$

where d denotes the degrees of the polynomial. Furthermore w_{ij} are the coefficients of the polynomial.

In Python:

1. Plot the data using a 3D scatter plot. By looking at the data determine d.
2. Formulate an linear system of the form
$$A\mathbf{x} \approx \mathbf{b}. \tag{5}$$
3. Implement in python a least squares solver to approximately solve (5) for the provided N samples. Plot the estimated polynomial function using the matplotlib `ax.plot_wireframe` function for $x, y \in [-4, 4]$. Report the resulting coefficients w_{ij}.