Numerical Optimization
Re-exam Handin 6

Dmitry Serykh (qwl888)

June 10, 2020

1 The Setup

In this assignment, I have implemented a coordinate descent minimizer for a quadratic problem with box-constraints, which can be written as:

$$\begin{aligned} &\min_{\vec{x}} f(\vec{x}) := \tfrac{1}{2}\vec{x}^\top A\vec{x} + \vec{b}^\top\vec{x} \\ &\text{s.t. } m_i \le x_i \le M_i, \quad \forall i = 1, \dots, n \end{aligned}$$

1.1 Solution to the 1D Problem

While the basic steps of the algorithm are described in the assignment text, some important details were omitted. I would especially focus on the solution to the 1D problem, which can be formulated as:

$$\begin{aligned} &\alpha_k = \arg\min_\alpha f(\vec{x}_k + \alpha\vec{e}_i) \;\text{ s.t. } m_i \le x_{k,i} + \alpha \le M_i \\ &\vec{x}_{k+1} = \vec{x}_k + \alpha_k\vec{e}_i \end{aligned}$$

The step direction for the quadratic problem can be found analytically by using the formula (3.55, p. 56 in the book):

$$\alpha_k = -\frac{\nabla f_k^T p_k}{p_k^T Q p_k}$$

where p_k is the search direction and Q is the matrix. It can be reformulated to:

$$\alpha_k = -\frac{\nabla f(x_k)^T\vec{e}_i}{\vec{e}_i^T A\vec{e}_i} = -\frac{\nabla f(x_k)_i}{A_{i,i}}$$

The downside of this approach is that the endpoint of the step can be outside of the feasible region. We can solve this by adding:

$$\alpha_k = \max\left(\min\left(\alpha_{min},\, -\frac{\nabla f(x_k)_i}{A_{i,i}}\right), \alpha_{max}\right)$$

where $\alpha_{min} = x_i - m_i$ and $\alpha_{max} = x_i - M_i$.

1.2 Parameters

I used following parameter values in my implementation:

- I used the KKT error as stopping criterion, by following the guidelines in the assignment text. The threshold for the norm of h was set to $\varepsilon = 1e-5$.
- Starting point was placed in the center of the feasible area, s.t $x_0 = \frac{\vec{M}-\vec{m}}{2}$
- `max_iter` was set to 1000

2 Testing protocol

2.1 Validation

I validated my implementation by implementing the exact 2D problem solver using the method, described in the text, where we first try to make a newton step, and if the endpoint lies outside the feasible region, we check all the box sides, by solving 4 problems in one dimension and finding the minimum of the four.

My coordinate descent implementation find the minima, which lay in the proximity of 10^{-4} of the exact solution. It was important to test the cases, where the global minima of the function lies inside the feasible region and ones, where it is not the case. It was also important to both test the cases where the minimizer had to take positive steps and cases with negative steps.

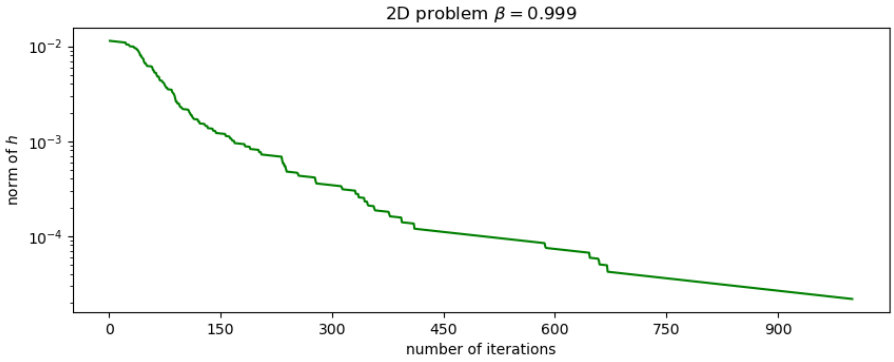

Figure 1: Convergence plot (average norm of $\vec{h}$) of my implementation of the coordinate descent for the 2D problem with $\beta = 0.999$

2.2 Problem Generation

In order to generate the quadratic problems, I have used two approaches, both of which were described in the assignment text:

- The first is for generation of hard 2D problems, where:

$$A = \begin{bmatrix} 1 & \beta \\ \beta & 1 \end{bmatrix}$$
$$-1 < \beta < 1$$
$$\beta = \frac{999}{1000}$$

  For the value of $\vec{b}$, I used a vector of normally distributed values with $\sigma = 0.01$ and $\mu = 0$, since the assignment text suggests using small values of $\vec{b}$.

- The second is for generation of quadratic problems of higher dimensions d, where:

$$A = B^T B$$
$$-1 \leq x_i \leq 1$$

  The same method for generation of $\vec{b}$ was used.

2.3 Metrics

In order to further test the effectiveness of my implementation, I came up with a testing protocol, where I used following metrics:

- The convergence plots with norm of the vecor $\vec{h}$ on the y-axis and iteration number on the x-axis for the 2D quadratic problems. The resulting plot can be seen on Figure 1.
- The convergence plots with norm of the $\vec{h}$ for the multidimensional quadratic problems for $d \in \{5, 10, 15\}$. The resulting plot can be seen on Figure 3.
- I measured the relationship between the dimensionality of the quadratic problem and the efficiency of my implementation, which is measured by the number of iterations until the magnitude of $\vec{h}$ reaches 10^{-5}. The resulting plot can be seen on Figure 2.

Each experiment was repeated 100 times for all metrics and the mean was taken.

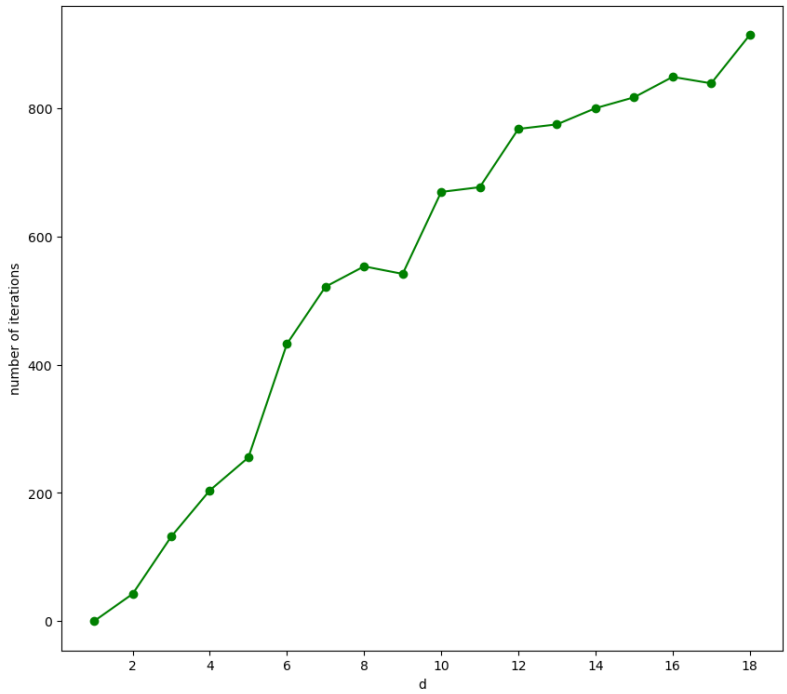

Figure 2: Average efficiency of my implementation of the coordinate descent as function of quadratic problem dimensionality

3 Theoretical Part

3.1

I will reformulate the First-Order Necessary Conditions in terms of the box constraints. I will use index i for the dimension of the problem, hence there would be two constraints for each i:

- Lower bound: $x_i^* \geq m_i \leftrightarrow x_i^* - m_i \geq 0$
- Upper bound: $x_i^* \leq M_i \leftrightarrow M_i - x_i^* \geq 0$

Furthermore, there would be two sets of Lagrangian multipliers: $\lambda_{m,i}$, related to lower bounds and $\lambda_{M,i}$, corresponding to the upper bounds. The First-Order Necessary Conditions can therefore be expressed as:

$\vec{x}^*$ is local minimum for the quadratic problem with box constraints, if:

$$x_i^* - m_i \geq 0, \qquad \forall i = 1, \ldots, n \tag{1}$$
$$M_i - x_i^* \geq 0, \qquad \forall i = 1, \ldots, n \tag{2}$$
$$\lambda_{m,i}^* \geq 0, \qquad \forall i = 1, \ldots, n \tag{3}$$
$$\lambda_{M,i}^* \geq 0, \qquad \forall i = 1, \ldots, n \tag{4}$$
$$\lambda_{m,i}^*(x_i^* - m_i) = 0, \qquad \forall i = 1, \ldots, n \tag{5}$$
$$\lambda_{M,i}^*(M_i - x_i^*) = 0, \qquad \forall i = 1, \ldots, n \tag{6}$$
$$\nabla f(x^*) - \sum_{j \in \mathcal{A}_m(x^*)} \lambda_{m,j}^* \vec{e_j} x_i^* + \sum_{j \in \mathcal{A}_M(x^*)} \lambda_{M,j}^* \vec{e_j} x_j^* = 0 \tag{7}$$

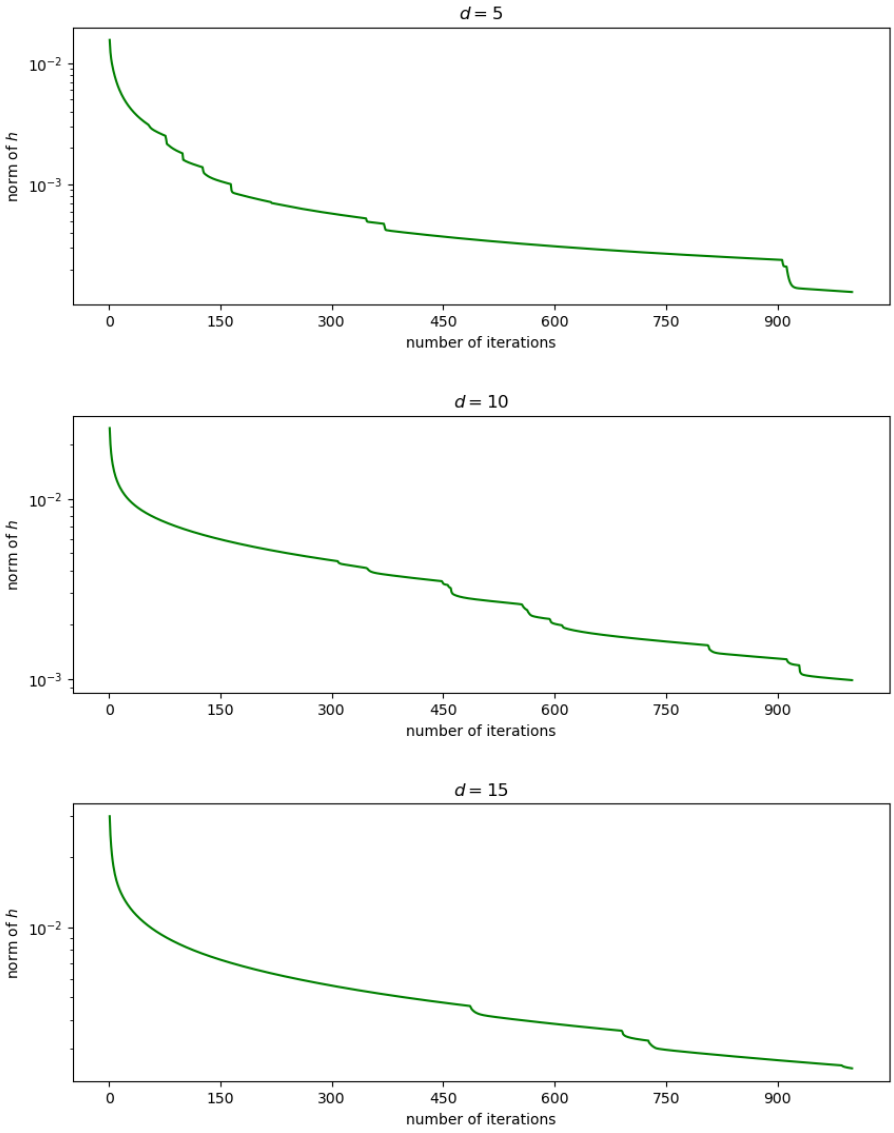

Figure 3: Convergence plot (average norm of $\vec{h}$) of my implementation of the coordinate descent for the multi-dimensional problem $d \in \{5, 10, 15\}$

3.2

In order to determine the value of λ_i, s.t. the two conditions are fulfilled, and the LHS of (7) is minimized, I must look at two cases: one, where the value of $x_{k,i} = m_i$ and where it is not the case. I will only look at the lower bound, hence inequalities (1), (3) and (5), but a symmetrical argument can be applied to the upper bound.

- If $x_{k,i} \neq m_i$, (5) would only be fulfilled if $\lambda_i = 0$, hence this value must be chosen.
- If $x_{k,i} = m_i$, (5) is fulfilled, and I can choose any non-negative value of λ_i s.t. (3) holds. The active sets $\mathcal{A}_M$ and $\mathcal{A}_m$ are disjoint, hence I must choose the value of λ_i, s.t. $\lambda_i x_{k,i} = g_{k,i}$:

  $$\lambda_i = \frac{\vec{A_i}\vec{x_k} + b_i}{m_i}$$

  Since (3) must hold, the inequality would only have a solution when $g_k > 0$, otherwise I should use $\lambda_i = 0$. In that case, (5) would still hold.

3.3

For the optimal λ_i (that we found s.t. (3) - (6) hold), we can minimize the norm of our solution, by minimizing the LHS of (7). If the constraint of index i is active and we can find the optimal value of λ_i, (m_i or M_i) is subtracted from the gradient and we get: $\lambda_i x_{k,i} - g_{k,i} = 0$ for that index.

However, as mentioned earlier, we can only find that optimal $\lambda_i \neq 0$, if $g_{k,i} < 0$ and $g_{k,i} > 0$ for the upper and lower bounds respectively. Luckily for us, this condition is maintained by the safeguards in the suggested program. Intuitively, this also makes sense. If the derivative of the objective function with respect to i is negative, we would like to increase the value of $x_{k,i}$. But, if $x_{k,i} = M_i$, we can not increase it any further while preserving (2), hence we must stop. If $x_{k,i} = m_i$ and $g_{k,i} < 0$, nothing is stopping us from increasing the value of $x_{k,i}$. The same logic applies to the lower bound. Therefore, the norm of my solution is computed by the suggested program.

3.4

In order to see if we are close to the minimum, we can check that the values of ∇f that correspond to the inactive constraints is close to zero, and we can do it by checking if $\|\vec{h}\| < \varepsilon$.

It can also be explained in a following way: for the active constraints, where $x_{k,i} = m_i$ or $x_{k,i} = M_i$ and the gradient is positive/negative, we have "extracted" as much minimization in direction $\vec{e_i}$ as possible, without violating (1) and (2). Therefore, we must concentrate on optimizing the remaining directions, only include them in the norm of our solution and exclude the active constraints by setting $h_i = 0$. If we have "extracted" all the possible decrease in the objective function by reaching the upper and lower bounds, we can not minimize any further, hence $\|\vec{h}\| = 0 < \varepsilon$. Therefore, I can conclude that this test constitutes a good stopping criterion.

4 Conclusion

I have implemented a working implemenation of a quadratic problem minimizer using coordinate descent with box constraints and KKT error as a stopping criterion. I validated my solution by comparing the found minima with an exact problem solver and argued for why the KKT error is a good stopping criterion using Theorem 12.1. I have also plotted the convergence of the method and can conclude that it exhibits a linear convergence, simmilarly to a steepest descent minimizer. Moreover, I can also conclude that the time to reach the optimum scales linearly with the problem dimensionality.