REPORT 1

Mathieu Gaillard, Yichen Sheng

April 1, 2019

What has been done

We have read the first chapter of the book: Primal-dual interior-point methods by Stephen J. Wright, SIAM 1997. It gives a nice introduction about interior point solvers for linear programming. It exposes the same method as in class with more details. Each sub section shows a potential issue and introduce a chapter in which it is solved.

With the code we have seen in class and the explanations in the first chapter of the book, we managed to get a working code, which can solve the 2D problem seen in class. We give more details about this example later in this section. We set up a GitHub repository with clear explanations in the Readme.md file. Our solver is located in a separate package so that a potential user would just need to import it in order to use it.

We require the user to provide a problem in standard form as we did not implement a conversion method yet. To get a feasible starting point, we still rely on another LP solver. In fact, we use the same method as shown in class that uses a log barrier term to get a point around the center of the feasible region. Actually, we may not need this because we implemented the infeasible-interior-point method. However, we did not investigate that yet. Once a descent direction is found, we use a backtracking algorithm to find a suitable value for alpha.

Following is a description of how our code solves a very basic 2D LP problem.

$$\begin{array}{ll} \text{minimize} & -x_1 - 2x_2 \\ \text{subject to} & -2x_1 + x_2 \leq 2 \\ & -x_1 + 2x_2 \leq 7 \\ & x_1 \leq 3 \\ & x_1, x_2 \geq 0 \end{array}$$

The solution to this problem is: $x_1 = 3, x_2 = 5$. With a tolerance of 10^{-8}, our program finds the following solution in 30 iterations:

$$\mathbf{x} = \begin{pmatrix} 2.999999995343387 \\ 4.999999993015081 \end{pmatrix}$$

Our starting point is:

$$\mathbf{x}_0 = \begin{pmatrix} 2.19 \\ 2.19 \\ 4.19 \\ 4.80 \\ 0.80 \end{pmatrix}$$

$$\lambda_0 = \begin{pmatrix} -2.38 \\ -2.08 \\ -12.41 \end{pmatrix}$$

$$\mathbf{s}_0 = \begin{pmatrix} 4.55 \\ 4.55 \\ 2.38 \\ 2.08 \\ 12.41 \end{pmatrix}$$

Future work

The most important thing to do next is to modify the code to be able to run the problems from the online matrix repository. The objective by next week would be to manage to solve at least one problem from the list given in the project description. To achieve that result, we would need to use the sparse matrix class and write some code to convert the problem into standard form.

On the long term, we still need to do the following things:

Our algorithm to find the value of alpha could be perfected. Right now, we use the same principle as backtracking, however we could simply solve the equations to find the best alpha possible.

```
# This code could be improved
alpha = 1.0
while (any((x + alpha * dx) .< 0) || any((s + alpha * ds) .< 0))
    alpha /= 2;
end
```

We still rely on an external LP solver to find a starting point, we would need to find an alternative solution for that.

Although we did not encounter accuracy problem and terribly ill matrix yet, we would need a more robust algorithm to deal with those bad situations.

We could implement the Mehrotra's Predictor-Corrector Algorithm as it seems to be a well-established method.

Right now, we do not check if the problem is unbound in the opposite of the gradient direction. When that happens, our algorithm will run optimization process in max allowed steps and return a value that does not satisfy KKT condition. We don't know yet a method to test that.