**CS 709 : Convex Optimization** **Autumn 2015**

# Second-Order Cone Programming

*Instructor: Ganesh Ramakrishnan By: Aditya Kumar Akash, 120050046, Manohar Kumar, 120050044*

## 1 Introduction

### 1.1 Cones

A set $C$ is called cone if $\forall x \in C, \theta \geq 0$ we have $\theta x \in C$. C is said to be convex cone if $\forall x_1, x_2 \in C$ and $\theta_1, \theta_2 \geq 0$ we have $\theta_1 x_1 + \theta_2 x_2 \in C$.
A *Second Order Cone* in $\mathcal{R}^{p+1}$ is defined as

$$C = \{(x, y) \in \mathcal{R}^{p+1} | \|x\|_2 \leq y\}$$

A second-order cone inequality on a vector $x \in \mathbf{R}^n$ states that a vector $(y, t)$ that is some affine combination of $x$ belongs to a second-order cone.

This is a constraint of the form $\|Ax + b\|_2 \leq c^T x + d$, where $A \in \mathbf{R}^{m \times n}$, $bin\mathbf{R}^m$, $cin\mathbf{R}^n$, and $d$ is a scalar.

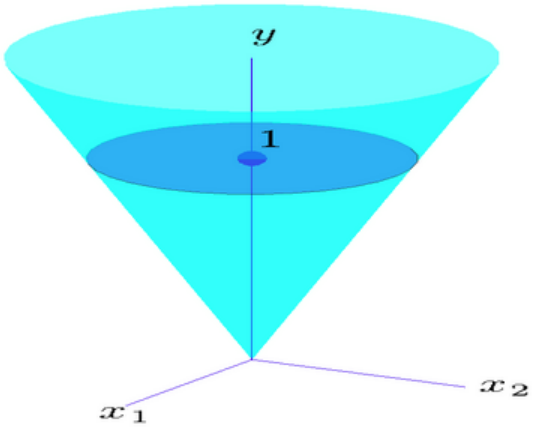


Figure 1: Second Order Cone

### 1.2 Second Order Cone Programming

A second-order cone program (or **SOCP**) is an optimization problem of the form

$$\min_x : c^T x \ : \ \|A_i x + b_i\|_2 \leq c_i^T x + d_i, ; ; i = 1, \ldots, m,$$

where $A_i \in \mathbf{R}^{p_i \times n}$'s are given matrices, $b_i \in \mathbf{R}_i^p$, $c_i \in \mathbf{R}^n$ vectors, and $d_i$'s scalars.

The problem is convex, since the constraint functions of the corresponding standard form

$$f_i(x) = \|A_i x + b_i\|_2 - (c_i^T x + d_i), ; ; i = 1, \ldots, m,$$

are of convex functions.

### 1.3 Methods to Solve SOCP

In general interior-point methods are used for solving convex optimization problems that include inequality constraints. Interior-point methods solve the problem by applying Newtons method to a sequence of equality constrained problems, or to a sequence of modified versions of the KKT conditions.
**In this report we present a formulation of Robust least squares problem into second order cone programming and solve it** using following algorithms

1. Barrier Method
2. Primal Dual Interior Point Method

## 2 Robust Least-Squares Problem

Many real world optimization problems involve data which is noisy, or uncertain, due to measurement or modelling errors. So adressing data uncertainty in mathematical models is a central problem in optimization. A normal least square problem is

$$\min_x \|Ax - y\|_2$$

, where $A \in \mathbb{R}^{m \times n}, y \in \mathbb{R}^m$.
Now it might be the case that the matrix A is known with some uncertainty. Like, we can assume that it is known to be within some distance (matrix distance) to a given nominal matrix $\hat{A}$. Let's assume that $\|A - \hat{A}\| \leq \rho$ , where $\|.\|$ denotes matrix 2-norm, and $\rho \geq 0$ measures the size of uncertainty.
So, the robust least squares problem can be written as

$$\min_x \max_{\|\Delta A\| \leq \rho} \|(\hat{A} + \Delta A)x - y\|_2$$

It can be interpreted as trying to minimize the worst case value of the residual norm.
By triangle inequality,

$$\|(\hat{A} + \Delta A)x - y\|_2 \leq \|\hat{A}x - y\|_2 + \|(\Delta A)x\|_2$$

Also, by definition of norm,

$$\|(\Delta A)x\|_2 \leq \|\Delta A\| \|x\|_2 \leq \rho \|x\|_2$$

Hence, we have a bound on the objective value of the robust problem,

$$\max_{\|\Delta A\| \leq \rho} \|(\hat{A} + \Delta A)x - y\|_2 \leq \|\hat{A}x - y\|_2 \leq \|\hat{A}x - y\|_2 + \rho \|x\|_2$$

The upper bound is attained by the following value of $\Delta A$

$$\Delta A = \frac{\rho}{\|\hat{A}x - y\|_2 \cdot \|x\|_2} (\hat{A}x - y)x^T$$

Finally the robust least squares is equivalent to the following problem,

$$\min_x \|\hat{A}x - y\|_2 + \rho \|x\|_2$$

This is an SOCP,

$$\min_{x,u,v} u + \rho v \quad : \quad u \geq \|\hat{A}x - y\|_2, v \geq \|x\|_2$$

# 3 Algorithms

Consider the SOCP in general form

$$\min_x : c^T x \ : \ \|A_i x + b_i\|_2 \leq c_i^T x + d_i, ; ; i = 1, \ldots, m,$$

We describe the methods used to solve this formulation of optimization problem.

## 3.1 Barrier Method

We consider the following logarithmic barrier function

$$\phi(x) = -\sum_{i=1}^{m} \log((c_i^T x + d_i)^2 - \|A_i x + b_i\|^2)$$

with domain of $\phi$ as $\{x | \|A_i x + b_i\| \leq c_i^T x + d_i \forall i\}$

The psuedo code can be written as
Given strictly feasible $x$, $t = t(0) > 0$, $\mu > 1$, tolerance $\epsilon > 0$.
*Repeat*

1. Centering step. Compute $x^*(t)$ by minimizing $tf_0 + \phi$. This step we have implemented using BFGS algorithm which is a quassi newton algorithm to find optimum value
2. Update. $x = x^*(t)$.
3. Stopping criterion. quit if $m/t < \epsilon$.
4. Increase $t$. $t = \mu t$. This is done as in formulation as $t \to \infty$ we get optimum answer.

## 3.2 Primal Dual Interior Point Method

These methods are based on reducing the gap between the primal and dual solutions.

**Primal-dual search direction**

As in the barrier method, we start with the modified KKT conditions (11.15), expressed as $r_t(x, \lambda, \nu) = 0$, where we define

$$r_t(x, \lambda, \nu) = \begin{bmatrix} \nabla f_0(x) + Df(x)^T \lambda + A^T \nu \\ -\mathbf{diag}(\lambda) f(x) - (1/t)\mathbf{1} \\ Ax - b \end{bmatrix}, \qquad (11.53)$$

and $t > 0$. Here $f : \mathbf{R}^n \to \mathbf{R}^m$ and its derivative matrix $Df$ are given by

$$f(x) = \begin{bmatrix} f_1(x) \\ \vdots \\ f_m(x) \end{bmatrix}, \qquad Df(x) = \begin{bmatrix} \nabla f_1(x)^T \\ \vdots \\ \nabla f_m(x)^T \end{bmatrix}.$$

Figure 2: Modified KKT conditions

$$y = (x, \lambda, \nu), \qquad \Delta y = (\Delta x, \Delta\lambda, \Delta\nu),$$

respectively. The Newton step is characterized by the linear equations

$$r_t(y + \Delta y) \approx r_t(y) + Dr_t(y)\Delta y = 0,$$

*i.e.*, $\Delta y = -Dr_t(y)^{-1} r_t(y)$. In terms of $x$, $\lambda$, and $\nu$, we have

$$\begin{bmatrix} \nabla^2 f_0(x) + \sum_{i=1}^m \lambda_i \nabla^2 f_i(x) & Df(x)^T & A^T \\ -\mathbf{diag}(\lambda) Df(x) & -\mathbf{diag}(f(x)) & 0 \\ A & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta\lambda \\ \Delta\nu \end{bmatrix} = - \begin{bmatrix} r_{\text{dual}} \\ r_{\text{cent}} \\ r_{\text{pri}} \end{bmatrix}. \tag{11.54}$$

The *primal-dual search direction* $\Delta y_{\text{pd}} = (\Delta x_{\text{pd}}, \Delta\lambda_{\text{pd}}, \Delta\nu_{\text{pd}})$ is defined as the solution of (11.54).

Figure 3: Finding Search Direction

We first compute the largest positive step length, not exceeding one, that gives $\lambda^+ \succeq 0$, *i.e.*,

$$\begin{aligned} s^{\max} &= \sup\{s \in [0,1] \mid \lambda + s\Delta\lambda \succeq 0\} \\ &= \min\{1, \ \min\{-\lambda_i/\Delta\lambda_i \mid \Delta\lambda_i < 0\}\}. \end{aligned}$$

We start the backtracking with $s = 0.99 s^{\max}$, and multiply $s$ by $\beta \in (0,1)$ until we have $f(x^+) \prec 0$. We continue multiplying $s$ by $\beta$ until we have

$$\|r_t(x^+, \lambda^+, \nu^+)\|_2 \leq (1 - \alpha s)\|r_t(x, \lambda, \nu)\|_2.$$

Figure 4: Line Search

*Primal-dual interior-point method*
Given x that satisfies $f_1(x) < 0, ..., f_m(x) < 0, \lambda \geq 0, \mu \geq 1, \epsilon > 0$,
*Repeat*

1. Determine $t$. Set $t = \mu m/\eta$

2. Compute primal-dual search direction $\Delta y$. This we have achieved using linear equation solver of numpy - a library in python.

3. *Line search and update.* Determine step length $s > 0$ and set $y := y + s\Delta y$. This was done using backtracking line search.

until $\eta \leq \epsilon$. $\eta$ is the duality gap.

# 4 Implementation and Results

We have implemented both the algorithms which are desribed in brief in above section. We used **python** programming language. We have used following packages to help us

1. cvxopt for matrix usage
2. numpy for solving system of linear equations

## 4.1 Problem Instance

Following problem instance which was generated randomly is used for analysis -

$$A = \begin{bmatrix} 58.2773294123 & 87.3081998769 & 73.6471184519 \\ 23.4473874681 & 25.3244933598 & 82.2292368154 \\ 66.3470558911 & 86.5022884107 & 69.2654461429 \end{bmatrix}$$

$$y = \begin{bmatrix} 55.1374437908 \\ 92.3665392038 \\ 2.44408187789 \end{bmatrix}$$

$\rho$ =4.77260793985, This is the uncertainty.

Now we solve this instance using our algorithm

## 4.2 Barrier

We plot the iteration number Vs a quantity which stands for dulality gap for this problem instance with $\epsilon = 10^{-5}$. The above figure shows better than linear convergence.

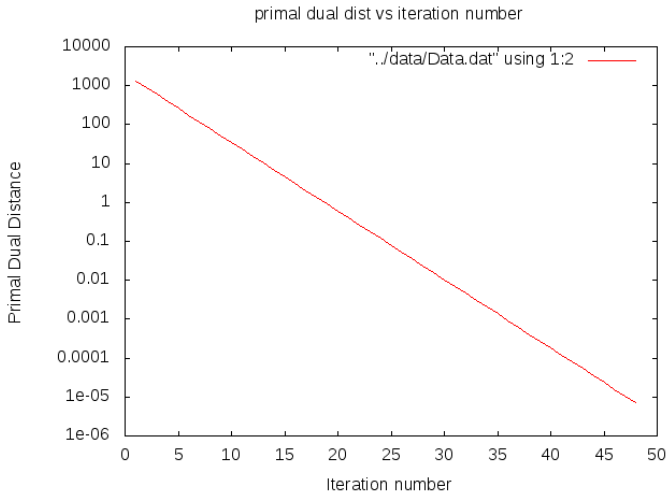


Figure 5: Quantity representing primal dual Distance Vs Iteration Number

### 4.3 Primal Dual Interior Point Algorithm

We plot the iteration number Vs dulality gap for this problem instance with $\epsilon = 10^{-5}$. The figure shows

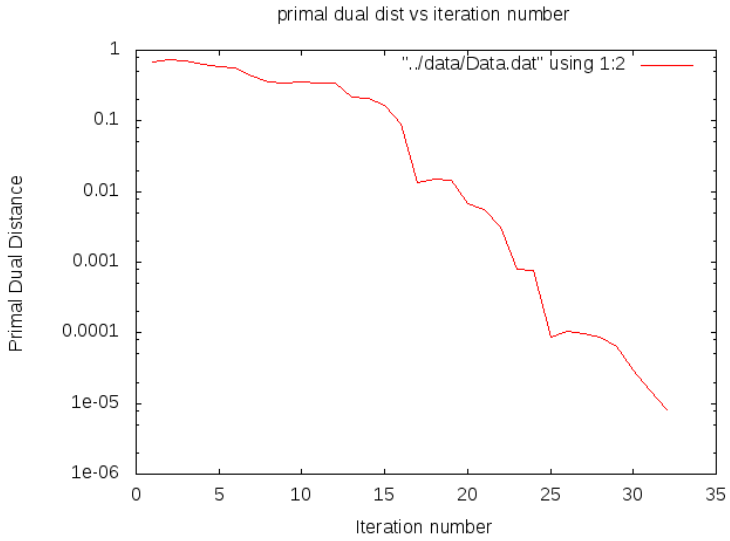


Figure 6: Primal Dual Distance Vs Iteration Number

better than linear convergence for primal dual algorithm.

### 4.4 Variation of Minimum value with uncertainty

We plotted using the barrier method the variation of minimum value for the sample instance with the uncertainty in data. As per expectation the optimum increases with the uncertainty.

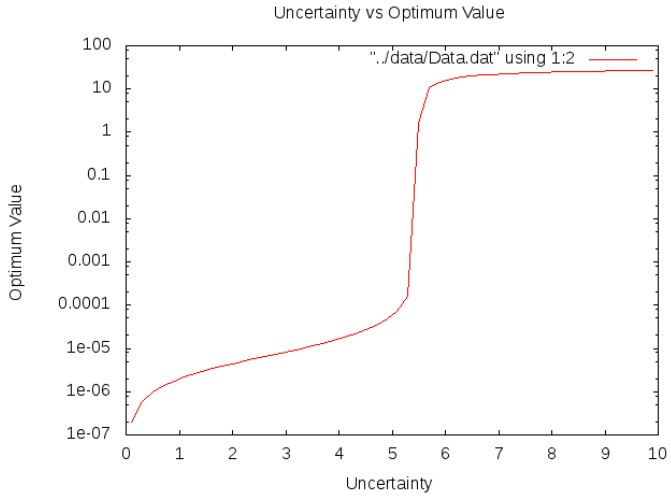


Figure 7: Uncertainty vs Minimum Value

# 5 Conclusion

We learnt how to formulate a problem into SOCP. We also got our hands on implementation of major algorithms in convex optimization namely - Barrier and Primal Dual Interior Point method.
It was a learning experience to code in python and learn to use available packages. We also played around with the other parameters available. Though the problem we formulated was small we still faced challenges in coding hessian and solving of intermediate system of equations.
We would like to thank Prof. Ganesh for giving us this opportunity.

# 6 References

We have refered following resources -

1. Wikipedia
2. Convex Notes from cs709
3. Book by Boyd