# Constrained convex optimization: algorithms and applications

Source: Ahmadi, ORF 363 slides.

# Convex optimization problems

A convex optimization problem is an optimization problem of the form

$$\begin{aligned} \min \ & f(x) \\ \text{s.t.} \ & g_i(x) \leq 0, \quad i = 1, \ldots, m \\ & h_j(x) = 0, \quad j = 1, \ldots, k \end{aligned}$$

where $f, g_i : \mathbb{R}^n \to \mathbb{R}$ are convex functions and $h_i : \mathbb{R}^n \to \mathbb{R}$ are affine functions

# Convex optimization problems

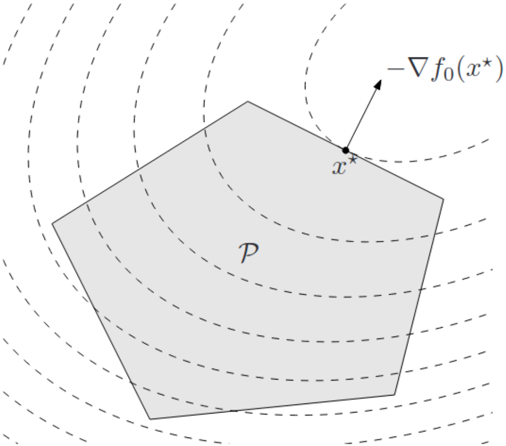
A convex optimization problem is an optimization problem of the form

$$\begin{aligned} \min \ & f(x) \\ \text{s.t.} \ & g_i(x) \leq 0, \quad i = 1, \ldots, m \\ & h_j(x) = 0, \quad j = 1, \ldots, k \end{aligned}$$

where $f, g_i : \mathbb{R}^n \to \mathbb{R}$ are convex functions and $h_i : \mathbb{R}^n \to \mathbb{R}$ are affine functions

The **feasible set** $\Omega = \{x \in \mathbb{R}^n \mid g_i(x) \leq 0, h_j(x) = 0\}$ is a convex set

# Geometry




$f_0$ is the objective, $\mathcal{P}$ is the feasible region

# Application of constrained convex optimization

## The Frank-Wolfe algorithm

# Constrained least-squares

Back to the diet problem

| Ingredients needed |  |  |  |  |
|---|---|---|---|---|
|  | 3 oz | 2 oz | 2 oz | 50 cts |
|  | 0 oz | 4 oz | 5 oz | 80 cts |
| Requirements | 6 oz | 10 oz | 8 oz | |

Suppose that we want to meet the nutrition requirements within our budget of 300 cents

# Constrained least-squares

Back to the diet problem

| Ingredients needed |  |  |  |  |
|---|---|---|---|---|
|  | 3 oz | 2 oz | 2 oz | 50 cts |
|  | 0 oz | 4 oz | 5 oz | 80 cts |
| Requirements | 6 oz | 10 oz | 8 oz | |

We can formulate this as a least-squares problem

$$\begin{aligned} \min_{x_1, x_2} \ & (3x_1 - 6)^2 + (2x_1 + 4x_2 - 10)^2 + (2x_1 + 5x_2 - 8)^2 \\ \text{s.t.} \ & 50x_1 + 80x_2 \leq 300 \\ & x_1, x_2 \geq 0 \end{aligned}$$

# Constrained least-squares

Back to the diet problem

| Ingredients needed |  |  |  |  |
|---|---|---|---|---|
|  | 3 oz | 2 oz | 2 oz | 50 cts |
|  | 0 oz | 4 oz | 5 oz | 80 cts |
| Requirements | 6 oz | 10 oz | 8 oz | |

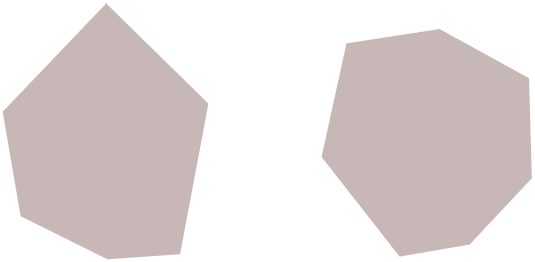
or in the matrix form:

$$\min_{x_1,x_2} \left\| \begin{bmatrix} 3 & 0 \\ 2 & 4 \\ 2 & 5 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - \begin{bmatrix} 6 \\ 10 \\ 8 \end{bmatrix} \right\|^2$$

$$\text{s.t.} \quad 50x_1 + 80x_2 \leq 300$$

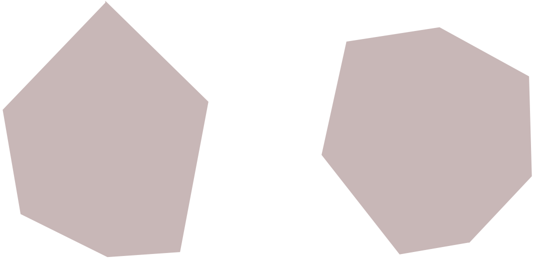
$$x_1, x_2 \geq 0$$

# Distance between polyhedra



$(\mathcal{P}_1) \qquad A_1 x \leq b_1 \qquad\qquad A_2 x \leq b_2 \quad (\mathcal{P}_2)$

# Distance between polyhedra

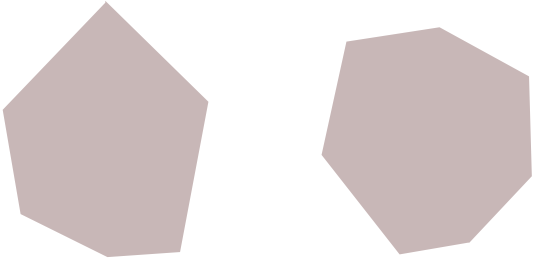


$(\mathcal{P}_1) \qquad A_1 x \leq b_1 \qquad\qquad\qquad A_2 x \leq b_2 \quad (\mathcal{P}_2)$

The distance between two polyhedra is

$$\min_{x_1 \in \mathcal{P}_1, x_2 \in \mathcal{P}_2} \|x_1 - x_2\|$$

# Distance between polyhedra



$(\mathcal{P}_1) \qquad A_1 x \leq b_1$ $\qquad\qquad$ $A_2 x \leq b_2 \quad (\mathcal{P}_2)$

This can be formulated as

$$\min_{x_1, x_2} \|x_1 - x_2\|^2$$
$$\text{s.t. } Ax_1 \leq b_1, Ax_2 \leq b_2$$

# Linear program with random cost

We go back to an LP

$$\begin{aligned} \text{minimize} \quad & c^T x \\ \text{subject to} \quad & Ax = b \\ \text{and} \quad & x \geq 0 \end{aligned}$$

But now suppose that the cost $c \in \mathbb{R}^n$ is random with mean $\bar{c}$ and covariance $\mathbb{E}[(c - \bar{c})(c - \bar{c})^T] = \Sigma$

# Linear program with random cost

We go back to an LP

$$\begin{aligned} \text{minimize} \quad & c^T x \\ \text{subject to} \quad & Ax = b \\ \text{and} \quad & x \geq 0 \end{aligned}$$

But now suppose that the cost $c \in \mathbb{R}^n$ is random with mean $\bar{c}$ and covariance $\mathbb{E}[(c - \bar{c})(c - \bar{c})^T] = \Sigma$ The mean and variance of the cost $c^T x$ is

$$\mathbb{E}[c^T x] = \bar{c}x, \quad \text{Var}[c^T x] = x^T \Sigma x$$

# Linear program with random cost

In general there is a trade-off between expected cost and variance

One way to take variance into account is to minimize a linear combination of the expected value and the variance of the cost

$$\bar{c}x + \gamma x^T \Sigma x$$

This is called **risk-sensitive** cost

The parameter $\gamma \geq 0$ is called the **risk-aversion parameter**

# Linear program with random cost

We replace the linear cost by the risk-sensitive cost

$$\begin{aligned} &\min_x \; \bar{c}x + \gamma x^T \Sigma x \\ &\text{s.t. } \; Ax = b \\ &\text{and } \; x \geq 0 \end{aligned}$$

This is a convex optimization problem

# Markowitz portfolio optimization

- We consider a classical portfolio problem with $n$ stocks held over a period of time

# Markowitz portfolio optimization

- We consider a classical portfolio problem with $n$ stocks held over a period of time

- Let $x_i$ denote the amount of asset $i$ held throughout the period, at the price at the beginning of the period

# Markowitz portfolio optimization

- We consider a classical portfolio problem with $n$ stocks held over a period of time

- Let $x_i$ denote the amount of asset $i$ held throughout the period, at the price at the beginning of the period
    - A normal long position in asset $i$ corresponds to $x_i > 0$

# Markowitz portfolio optimization

- We consider a classical portfolio problem with $n$ stocks held over a period of time

- Let $x_i$ denote the amount of asset $i$ held throughout the period, at the price at the beginning of the period
    - A normal long position in asset $i$ corresponds to $x_i > 0$
    - A short position in asset i (i.e., the obligation to buy the asset at the end of the period) corresponds to $x_i < 0$

# Markowitz portfolio optimization

- We consider a classical portfolio problem with $n$ stocks held over a period of time
- Let $x_i$ denote the amount of asset $i$ held throughout the period, at the price at the beginning of the period
  - A normal long position in asset $i$ corresponds to $x_i > 0$
  - A short position in asset i (i.e., the obligation to buy the asset at the end of the period) corresponds to $x_i < 0$
- We let $p_i$ denote the relative price change of asset $i$ over the period, i.e., its change in price over the period divided by its price at the beginning of the period

# Markowitz portfolio optimization

- The overall return on the portfolio is

  $$r = p^T x$$

  The unknown variable is the portfolio vector $x$

# Markowitz portfolio optimization

- The overall return on the portfolio is

$$r = p^T x$$

  The unknown variable is the portfolio vector $x$

- A simple set of constraints is $x_i \geq 0$ (i.e., no short positions) and $\sum_i x_i = B$ (i.e., the total budget to be invested is $B$)

# Markowitz portfolio optimization

- The overall return on the portfolio is

  $$r = p^T x$$

  The unknown variable is the portfolio vector $x$

- A simple set of constraints is $x_i \geq 0$ (i.e., no short positions) and $\sum_i x_i = B$ (i.e., the total budget to be invested is $B$)

- The prices $p \in \mathbb{R}^n$ is usually random with known mean $\bar{p}$ and covariance $\Sigma$

# Markowitz portfolio optimization

- The overall return on the portfolio is

  $$r = p^T x$$

  The unknown variable is the portfolio vector $x$

- A simple set of constraints is $x_i \geq 0$ (i.e., no short positions) and $\sum_i x_i = B$ (i.e., the total budget to be invested is $B$)

- The prices $p \in \mathbb{R}^n$ is usually random with known mean $\bar{p}$ and covariance $\Sigma$

- Therefore with portfolio $x \in \mathbb{R}^n$, the return $r$ is a random variable with mean $\bar{p}^T x$ and variance $x^T \Sigma x$

# Markowitz portfolio optimization

- We want to minimize the risk $x^T \Sigma x$, while maintaining the mean return $r_m$

# Markowitz portfolio optimization

- We want to minimize the risk $x^T \Sigma x$, while maintaining the mean return $r_m$

- The classical portfolio optimization problem by Markowitz is

$$\begin{aligned} \min_x \ & x^T \Sigma x \\ \text{s.t.} \ & \bar{p}^T x \geq r_m \\ & \sum_i x_i = B, \quad x \geq 0 \end{aligned}$$

# Markowitz portfolio optimization

- We want to minimize the risk $x^T \Sigma x$, while maintaining the mean return $r_m$

- The classical portfolio optimization problem by Markowitz is

$$
\begin{aligned}
\min_x \; & x^T \Sigma x \\
\text{s.t. } & \bar{p}^T x \geq r_m \\
& \sum_i x_i = B, \quad x \geq 0
\end{aligned}
$$

- One extension is to allow short positions, i.e., $x_i < 0$. To do this we introduce variables $x_{\text{long}}$ and $x_{\text{short}}$ with

$$x_{\text{long}} \geq 0 \quad x_{\text{short}} \geq 0 \quad x = x_{\text{long}} - x_{\text{short}} \quad \Sigma x_{\text{short}} \leq \gamma \Sigma x_{\text{long}}$$

where $\gamma$ limits the total short position as a fraction of the total long position

# Markowitz portfolio optimization

- As another extension we can include linear transaction costs

# Markowitz portfolio optimization

- As another extension we can include linear transaction costs

- Starting from a given initial portfolio $x_{\text{init}}$, we buy and sell assets to achieve the portfolio $x$, which we then hold over the period

# Markowitz portfolio optimization

- As another extension we can include linear transaction costs
- Starting from a given initial portfolio $x_{\text{init}}$, we buy and sell assets to achieve the portfolio $x$, which we then hold over the period
- We are charged a transaction fee for buying and selling assets, which is proportional to the amount bought or sold

# Markowitz portfolio optimization

- As another extension we can include linear transaction costs

- Starting from a given initial portfolio $x_{\text{init}}$, we buy and sell assets to achieve the portfolio $x$, which we then hold over the period

- We are charged a transaction fee for buying and selling assets, which is proportional to the amount bought or sold

- To handle this, we introduce variables $u_{\text{buy}}$ and $u_{\text{sell}}$, which determine the amount of each asset we buy and sell before the holding period. We have the constraints

$$x = x_{\text{init}} + u_{\text{buy}} - u_{\text{sell}} \quad u_{\text{buy}} \geq 0 \quad u_{\text{sell}} \geq 0$$

# Markowitz portfolio optimization

- We replace the simple budget constraint $\sum_i x_i = B$ with the condition that the initial buying and selling, including transaction fees, involves zero net cash

$$(1-f_{\mathrm{sell}})\sum u_{\mathrm{sell}} = (1+f_{\mathrm{buy}})\sum u_{\mathrm{buy}}$$

- The constants $f_{\mathrm{buy}} \geq 0$ and $f_{\mathrm{sell}} \geq 0$ are the transaction fee rates for buying and selling

- The left-hand side is the total proceeds from selling assets, less the selling transaction fee

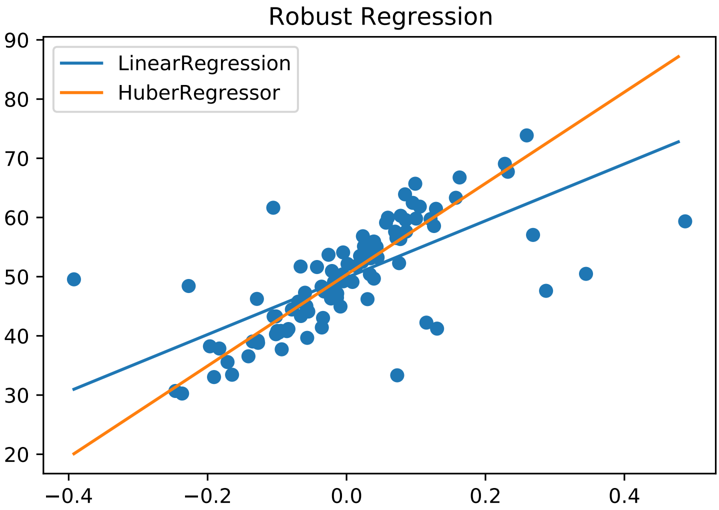
- The right-hand side is the total cost, including transaction fee, of buying assets

# Markowitz portfolio optimization

In summary, the optimization problem with transaction costs is

$$
\begin{aligned}
\min_x \; & x^T \Sigma x \\
\text{s.t.} \; & \bar{p}^T x \geq r_m \\
& x = x_{\text{init}} + u_{\text{buy}} - u_{\text{sell}} \\
& (1 - f_{\text{sell}}) \sum u_{\text{sell}} = (1 + f_{\text{buy}}) \sum u_{\text{buy}} \\
& u_{\text{buy}} \geq 0, u_{\text{sell}} \geq 0
\end{aligned}
$$

# Robust linear regression



# Robust linear regression

$X =$ matrix of independent variables, $y =$ dependent variable

The linear regression problem (finding the coefficient vector $\beta$) can be written as

$$\min_{\beta, r} r_1^2 + r_2^2 + \ldots + r_p^2$$
$$\text{s.t. } r = X\beta - y$$

Idea: minimize the residuals $r$

# Robust linear regression

$X =$ matrix of independent variables, $y =$ dependent variable

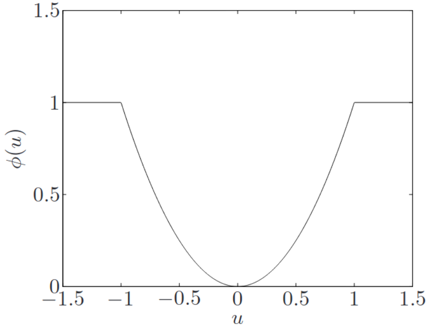
The linear regression problem (finding the coefficient vector $\beta$) can be written as

$$\begin{aligned} \min_{\beta, r} \ & \phi(r_1) + \phi(r_2) + \ldots + \phi(r_p) \\ \text{s.t.} \ & r = X\beta - y \end{aligned}$$

where $\phi$ is a **penalty function** (e.g. $\phi(u) = u^2$)

How can we design $\phi$ so that the regression is **robust to outliers**?

# First idea

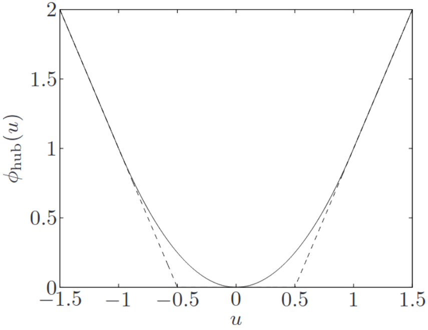



$$\phi_M(u) = \begin{cases} u^2 & |u| \leq M \\ M^2 & |u| > M \end{cases}$$

This penalty function agrees with LR for any residual smaller than $M$

but puts a fixed weight on any residual larger than $M$, no matter how much larger it is. However, this function is not convex!

# Second idea




$$\phi_{\text{hub}}(u) = \begin{cases} u^2 & |u| \le M \\ M(2M - |u|) & |u| > M \end{cases}$$

This penalty function agrees with the LR for residuals smaller than $M$

and then reverts to linear growth for larger residuals. This works!

# Huber regression (1964)

Data points: $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$; $\quad x_i \in \mathbb{R}^p, y \in \mathbb{R}$

# Huber regression (1964)

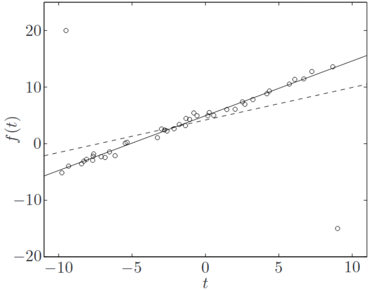
Data points: $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$; $\quad x_i \in \mathbb{R}^p, y \in \mathbb{R}$

The coefficient vector $\beta$ of the Huber regression is given by solving the least-squares problem:

$$\min_{\beta} \sum_{i=1}^{n} \phi_{\text{hub}}(y_i - A\beta)$$

# 1D example

Below is an example of 42 data points: $(x_1, y_1), \ldots, (x_{42}, y_{42})$




The solid line shows the Huber regression with one independent variable:

$$\hat{y}_i = a + bx_i$$

The coefficients $a, b$ and be found by solving

$$\min_{a,b} \; \phi_{\text{hub}}(y_i - a - bx_i)$$

Application of constrained convex optimization

## The Frank-Wolfe algorithm

# The Frank-Wolfe algorithm

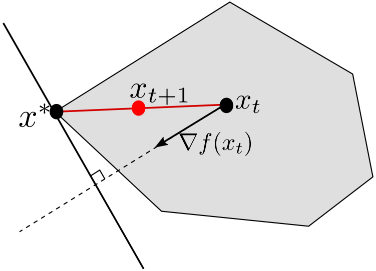
We go back to the constrained convex optimization problem

$$\begin{aligned} \min \ & f(x) \\ \text{s.t. } & g_i(x) \leq 0, \quad i = 1, \ldots, m \\ & h_j(x) = 0, \quad j = 1, \ldots, k \end{aligned}$$

There are many algorithms that solve this problem

One simple algorithm is the **Frank-Wolfe** algorithm

# The Frank-Wolfe algorithm

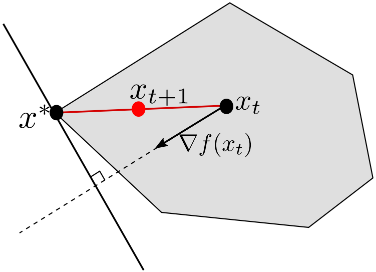



At step $t$, we approximate the objective by a linear function

$$f(x) \approx f(x_t) + \nabla f(x_t)(x - x_t)$$

$$\begin{aligned} \min\ & f(x_t) + \nabla f(x_t)(x - x_t) \\ \text{s.t.}\ & g_i(x) \leq 0, \quad i = 1, \dots, m \\ & h_j(x) = 0, \quad j = 1, \dots, k \end{aligned}$$

# The Frank-Wolfe algorithm

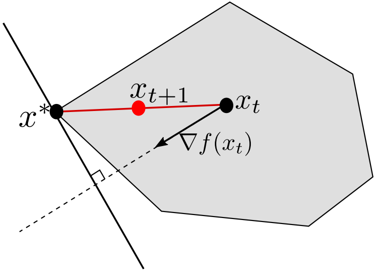



Since $x_t$ is fixed, $f(x_t)$ is a constant, so we can remove the term

$$\begin{aligned} \min\ & \nabla f(x_t)(x - x_t) \\ \text{s.t.}\ & g_i(x) \leq 0, \quad i = 1, \ldots, m \\ & h_j(x) = 0, \quad j = 1, \ldots, k \end{aligned}$$

Let $x^*$ be the solution to this LP (obtained via e.g. simplex method)

# The Frank-Wolfe algorithm




Use the line search to optimize the objective between $x^*$ and $x_t$

$$\min_{\alpha \in [0,1]} \; f(x_t + \alpha(x^* - x_t))$$

Go back to the first step, repeat until convergence