CS/ECE/ISYE524: Introduction to Optimization – Convex Optimization Models

Jeff Linderoth

Department of Industrial and Systems Engineering
University of Wisconsin-Madison

April 1, 2024

Quadratic forms

- **Linear functions:** sum of terms of the form $c_i x_i$ where the c_i are parameters and x_i are variables. General form:

$$c_1 x_1 + \cdots + c_n x_n = c^{\mathsf{T}} x$$

- **Quadratic functions:** sum of terms of the form $q_{ij} x_i x_j$ where q_{ij} are parameters and x_i are variables. General form:

$$q_{11} x_1^2 + q_{12} x_1 x_2 + \cdots + q_{nn} x_n^2 \qquad (n^2 \text{ terms})$$

$$= \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} q_{11} & \cdots & q_{1n} \\ \vdots & \ddots & \vdots \\ q_{n1} & \cdots & q_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = x^{\mathsf{T}} Q x$$

Quadratic forms

Any quadratic function $f(x_1, \ldots, x_n)$ can be written in the form $x^\mathsf{T} Q x$ where Q is a **symmetric matrix** ($Q = Q^\mathsf{T}$).

Proof: Suppose $f(x_1, \ldots, x_n) = x^\mathsf{T} R x$ where R is *not* symmetric. Since it is a scalar, we can take the transpose:

$$x^\mathsf{T} R x = \left(x^\mathsf{T} R x\right)^\mathsf{T} = x^\mathsf{T} R^\mathsf{T} x$$

Therefore:

$$x^\mathsf{T} R x = \tfrac{1}{2}\left(x^\mathsf{T} R x + x^\mathsf{T} R^\mathsf{T} x\right) = x^\mathsf{T} \tfrac{1}{2}(R + R^\mathsf{T}) x$$

So we're done, because $\frac{1}{2}(R + R^\mathsf{T})$ is symmetric!

Eigenvalue decomposition

Theorem. Every real symmetric matrix $Q = Q^{\mathsf{T}} \in \mathbb{R}^{n \times n}$ can be decomposed into a product:

$$Q = U \Lambda U^{\mathsf{T}}$$

where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ is a real diagonal matrix, and $U \in \mathbb{R}^{n \times n}$ is an orthogonal matrix. i.e. it satisfies $U^{\mathsf{T}} U = I$.

Eigenvalues and eigenvectors

If $A \in \mathbb{R}^{n \times n}$ and there is a vector v and scalar λ such that

$$Av = \lambda v$$

Then v is an *eigenvector* of A and λ is the corresponding *eigenvalue*.
Some facts:

- Any square matrix has n eigenvalues (but some may be repeated — they may not all be different).
- Each eigenvalue has at least one corresponding eigenvector.
- In general, eigenvalues & eigenvectors can be complex.
- In general, eigenvectors aren't orthogonal, and may not even be linearly independent. i.e. $V = \begin{bmatrix} v_1 & \cdots & v_n \end{bmatrix}$ may not be invertible. If it is, we say that A is *diagonalizable* and then $A = V \Lambda V^{-1}$. Otherwise, Jordan Canonical Form.

Symmetric matrices are **much** simpler!

Symmetric matrices; Eigenvalues and Eigenvectors

- Every symmetric $Q = Q^\mathsf{T} \in \mathbb{R}^{n \times n}$ has n **real** eigenvalues λ_i.
- There exist n mutually orthogonal eigenvectors $u_1, \ldots, u_n$:

$$Qu_i = \lambda_i u_i \qquad \text{for all } i = 1, \ldots, n$$

$$u_i^\mathsf{T} u_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

- If we define $U = \begin{bmatrix} u_1 & \cdots & u_n \end{bmatrix}$ then $U^\mathsf{T} U = I$ and

$$Q = U \begin{bmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_n \end{bmatrix} U^\mathsf{T}$$

Positive definite matrices

For a matrix $Q = Q^\mathsf{T}$, the following are equivalent:

1. $x^\mathsf{T} Q x \geq 0$ for all $x \in \mathbb{R}^n$
2. all eigenvalues of Q satisfy $\lambda_i \geq 0$

A matrix with this property is called *positive semidefinite* (PSD). The notation is $Q \succeq 0$.

Note: When we talk about PSD matrices, we *always* assume we're talking about a symmetric matrix.

Positive definite matrices

Name	Definition	Notation
Positive semidefinite	all $\lambda_i \geq 0$	$Q \succeq 0$
Positive definite	all $\lambda_i > 0$	$Q \succ 0$
Negative semidefinite	all $\lambda_i \leq 0$	$Q \preceq 0$
Negative definite	all $\lambda_i < 0$	$Q \prec 0$
Indefinite	everything else	(none)

Some properties:

- If $P \succeq 0$ then $-P \preceq 0$
- If $P \succeq 0$ and $\alpha > 0$ then $\alpha P \succeq 0$
- If $P \succeq 0$ and $Q \succeq 0$ then $P + Q \succeq 0$

Difference of positive definite matrices

Claim: Every symmetric matrix R can be written as $R = P - Q$ for some appropriate choice of symmetric matrices $P \succeq 0$ and $Q \succeq 0$.

Write eigenvalue decomposition as $R = \sum_{i=1}^{n} \lambda_i v_i v_i^{\mathsf{T}}$. Let

$$\mathcal{P} = \{i = 1, 2, \ldots, n \,:\, \lambda_i > 0\},$$
$$\mathcal{N} = \{i = 1, 2, \ldots, n \,:\, \lambda_i < 0\}.$$

Then

$$R = \sum_{i \in \mathcal{P}} \lambda_i v_i v_i^T - \sum_{i \in \mathcal{N}} (-\lambda_i) v_i v_i^T.$$

Get result by setting

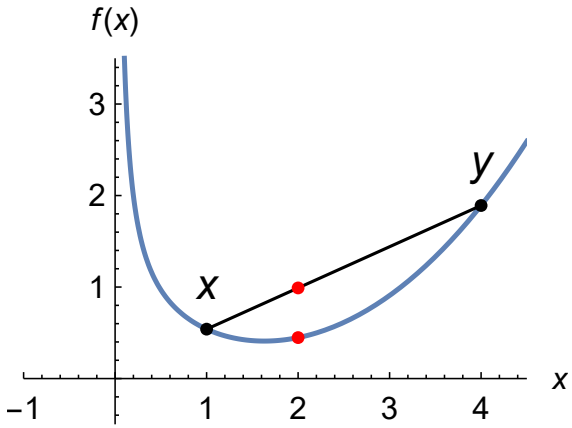
$$P = \sum_{i \in \mathcal{P}} \lambda_i v_i v_i^T, \quad Q = \sum_{i \in \mathcal{N}} (-\lambda_i) v_i v_i^T.$$

Convex functions

- A function $f : \mathbb{R}^n \to \mathbb{R}$ is *convex* for all $x, y \in \mathbb{R}^n$ and $0 \leq \alpha \leq 1$, we have:

$$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha) f(y)$$

- f is *concave* if $(-f)$ is convex.



- "Holds water"
- All local minima are global minima

Convex and concave functions on $\mathbb{R}$

Convex functions on $\mathbb{R}$

- Affine: $ax + b$.
- Absolute value: $|x|$.
- Quadratic: ax^2 for any $a \geq 0$.
- Exponential: a^x for any $a > 0$.
- Powers: x^α for $x > 0$, $\alpha \geq 1$ or $\alpha \leq 0$.
- Negative entropy: $x \log x$ for $x > 0$.

Concave functions on $\mathbb{R}$

- Affine: $ax + b$.
- Quadratic: ax^2 for any $a \leq 0$.
- Powers: x^α for $x > 0$, $0 \leq \alpha \leq 1$.
- Logarithm: $\log x$ for $x > 0$.

Convex and concave functions

Convex functions on $\mathbb{R}^n$

- Affine: $a^\mathsf{T} x + b$.
- Norms: $\|x\|_2$, $\|x\|_1$, $\|x\|_\infty$
- Quadratic form: $x^\mathsf{T} Q x$ for any $Q \succeq 0$

Concave functions on $\mathbb{R}^n$

- Affine: $a^\mathsf{T} x + b$.
- Quadratic form: $x^\mathsf{T} Q x$ for any $Q \preceq 0$

Building convex functions

1. Nonnegative weighted sum: If $f(x)$ and $g(x)$ are convex and $\alpha, \beta \geq 0$, then $\alpha f(x) + \beta g(x)$ is convex.
2. Composition with an affine function:
   If $f(x)$ is convex, so is $g(x) := f(Ax + b)$
3. Pointwise maximum: If $f_1(x), \ldots, f_k(x)$ are convex, then $g(x) := \max\{f_1(x), \ldots, f_k(x)\}$ is convex.

- N.B.: A composition of two convex functions is not necessarily convex!
  - *Example* in $\mathbb{R}$: $g(x) = |x|$ and $h(x) = x^2 - 1$ are both convex, but $g(h(x)) = |x^2 - 1|$ is not convex.

Least squares

- We often are given a linear system $Ax = b$, $A \in \mathbb{R}^{m \times n}$, with $m > n$ (overdetermined).
- If there is no solution to $Ax = b$, we try instead to have $Ax \approx b$.
- The least-squares approach: make Euclidean norm $\|Ax - b\|$ as small as possible.
  - Recall: $\|x\| := \sqrt{x_1^2 + \cdots + x_n^2} = \sqrt{x^\mathsf{T} x}$
- Equivalently: make $\|Ax - b\|^2$ as small as possible.

Standard form:

$$\underset{x}{\text{minimize}} \quad \|Ax - b\|^2$$

This is an **unconstrained optimization problem**.

Positive Definite

- Note that the quadratic form is PSD:

$$\|Ax - b\|^2 = (Ax - b)^\mathsf{T}(Ax - b) = x^\mathsf{T}A^\mathsf{T}Ax - 2b^\mathsf{T}Ax + b^\mathsf{T}b$$

- For any matrix A, $A^\mathsf{T}A$ is PSD, since for any $y \in \mathbb{R}^n$, let $z = Ay$, and then

$$0 \leq z^T z = (A^\mathsf{T}y)^\mathsf{T}(Ay) = (y^\mathsf{T}A^\mathsf{T})(Ay) = y^\mathsf{T}(A^\mathsf{T}A)y.$$

- So regression is a convex optimization problem
- Of course, you know there is a faster solution in terms of the solution of the normal equations:

More least squares

- Solving the least squares optimization problem:

$$\underset{x}{\text{minimize}} \quad \|Ax - b\|^2$$

- Is equivalent to solving the normal equations:

$$A^\mathsf{T} A \hat{x} = A^\mathsf{T} b$$

- If $A^\mathsf{T} A$ is invertible (A has linearly independent columns)

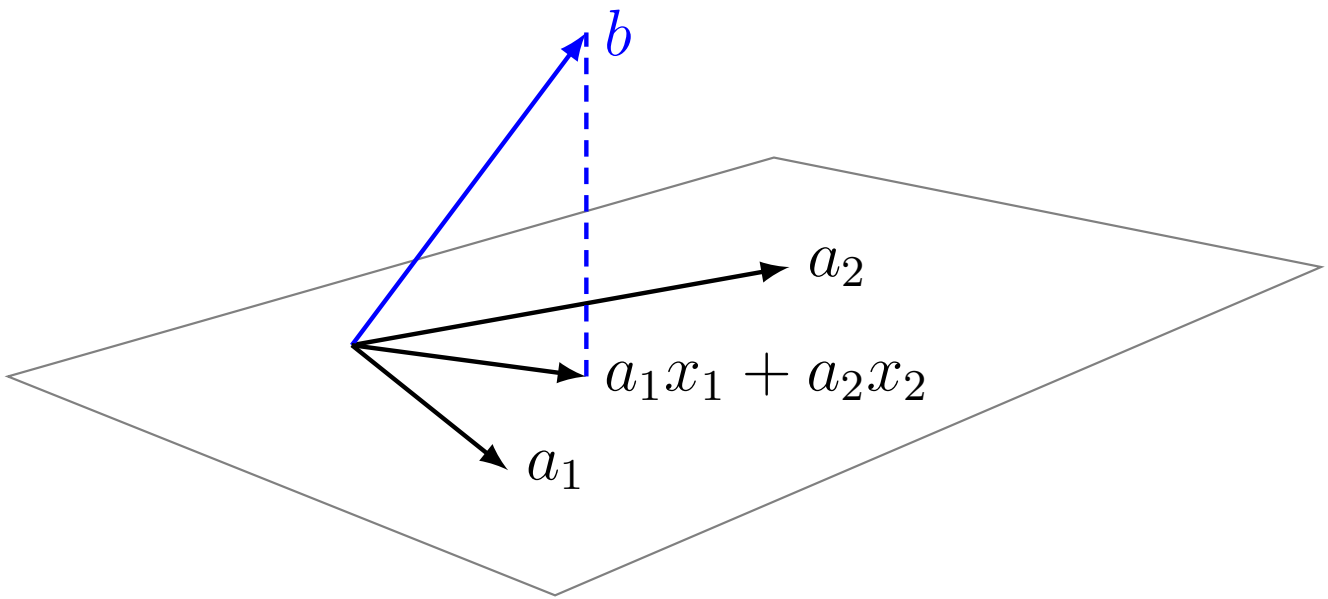
$$\hat{x} = (A^\mathsf{T} A)^{-1} A^\mathsf{T} b$$

- $A^\dagger := (A^\mathsf{T} A)^{-1} A^\mathsf{T}$ is called the *pseudoinverse* of A. `pinv(A)` in Julia (LinearAlgebra)

Geometry of Least Squares

- **column interpretation**: find the linear combination of columns $\{a_1, \ldots, a_n\}$ that is closest to b.

$$\|Ax - b\|^2 = \left\|(a_1x_1 + \cdots + a_nx_n) - b\right\|^2$$



Geometry of Least Squares

- **row interpretation**: If $\tilde{a}_i^\mathsf{T}$ is the ith row of A, define $r_i := \tilde{a}_i^\mathsf{T} x - b_i$ to be the i^{th} residual component.

$$\|Ax - b\|^2 = (\tilde{a}_1^\mathsf{T} x - b_1)^2 + \cdots + (\tilde{a}_m^\mathsf{T} x - b_m)^2$$

  We minimize the sum of squares of the residuals.
- Solving $Ax = b$ would make all residual components zero.
- Least squares attempts to make all of them small.

Regression Example: curve-fitting

- We are given noisy data points (x_i, y_i).
- We suspect they are related by $y = px^2 + qx + r$
- Find the p, q, r that best agrewe es with the data.

Writing all the equations:

$$\begin{aligned} y_1 &\approx px_1^2 + qx_1 + r \\ y_2 &\approx px_2^2 + qx_2 + r \\ &\vdots \\ y_m &\approx px_m^2 + qx_m + r \end{aligned} \quad \Longrightarrow \quad \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \approx \begin{bmatrix} x_1^2 & x_1 & 1 \\ x_2^2 & x_2 & 1 \\ \vdots & \vdots & \vdots \\ x_m^2 & x_m & 1 \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

Example: curve-fitting

- **More complicated**: $y = pe^x + q\cos(x) - r\sqrt{x} + s\,x^3$
- Find the p, q, r, s that best agrees with the data.

Writing all the equations:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \approx \begin{bmatrix} e^{x_1} & \cos(x_1) & -\sqrt{x_1} & x_1^3 \\ e^{x_2} & \cos(x_2) & -\sqrt{x_2} & x_2^3 \\ \vdots & \vdots & \vdots & \vdots \\ e^{x_m} & \cos(x_m) & -\sqrt{x_m} & x_m^3 \end{bmatrix} \begin{bmatrix} p \\ q \\ r \\ s \end{bmatrix}$$

- Julia notebook: Regression.ipynb
- We probably want to install Gurobi (for its better ability to solve quadratic optimization problems)
- Also will start doing some plotting: See `https://matplotlib.org/stable/tutorials/pyplot.html`

Optimal tradeoffs

We often want to optimize several different objectives simultaneously, but these objectives are **conflicting**.

- Risk vs Expected return (finance)
- Power vs Fuel economy (automobiles)
- Quality vs Memory (audio compression)
- Space vs Time (computer programs)

Tradeoff Applications

1. Radiation Treatment Planning
   - Maximize dose at cancerous tissue sites
   - Minimize dose to critical normal tissue
2. Military Planning
   - Maximize number of targets hit. (Targets themselves have different priorities)
   - Minimize time to reach targets.
   - Minimize number of high-value weapons
3. Network design (retail network)
   - Minimize cost of network
   - Maximize customer service
4. Portfolio Planning
   - Maximize Return
   - Minimize Risk
   - Minimize Tax implications
   - Minimize market impact from large trades

Optimal tradeoffs

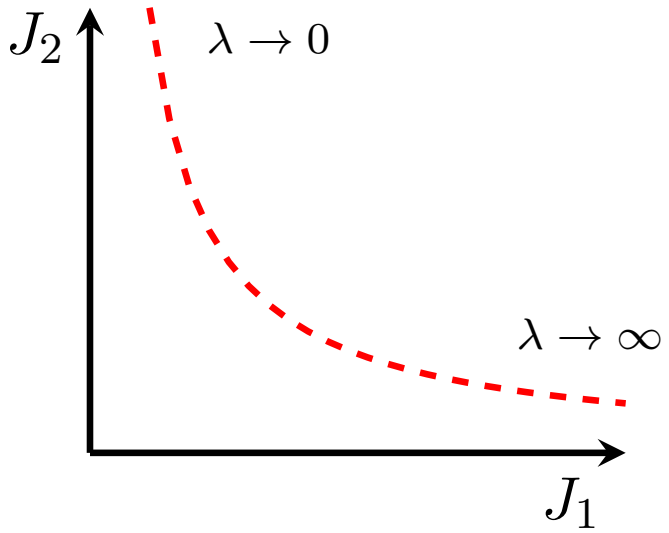
- Suppose $J_1 = f_1(x) = \|Ax - b\|^2$ and $J_2 = f_2(x) = \|Cx - d\|^2$.
- We would like to make **both** J_1 and J_2 small.
- A sensible approach: solve the optimization problem:

$$\underset{x}{\text{minimize}} \quad J_1 + \lambda J_2$$

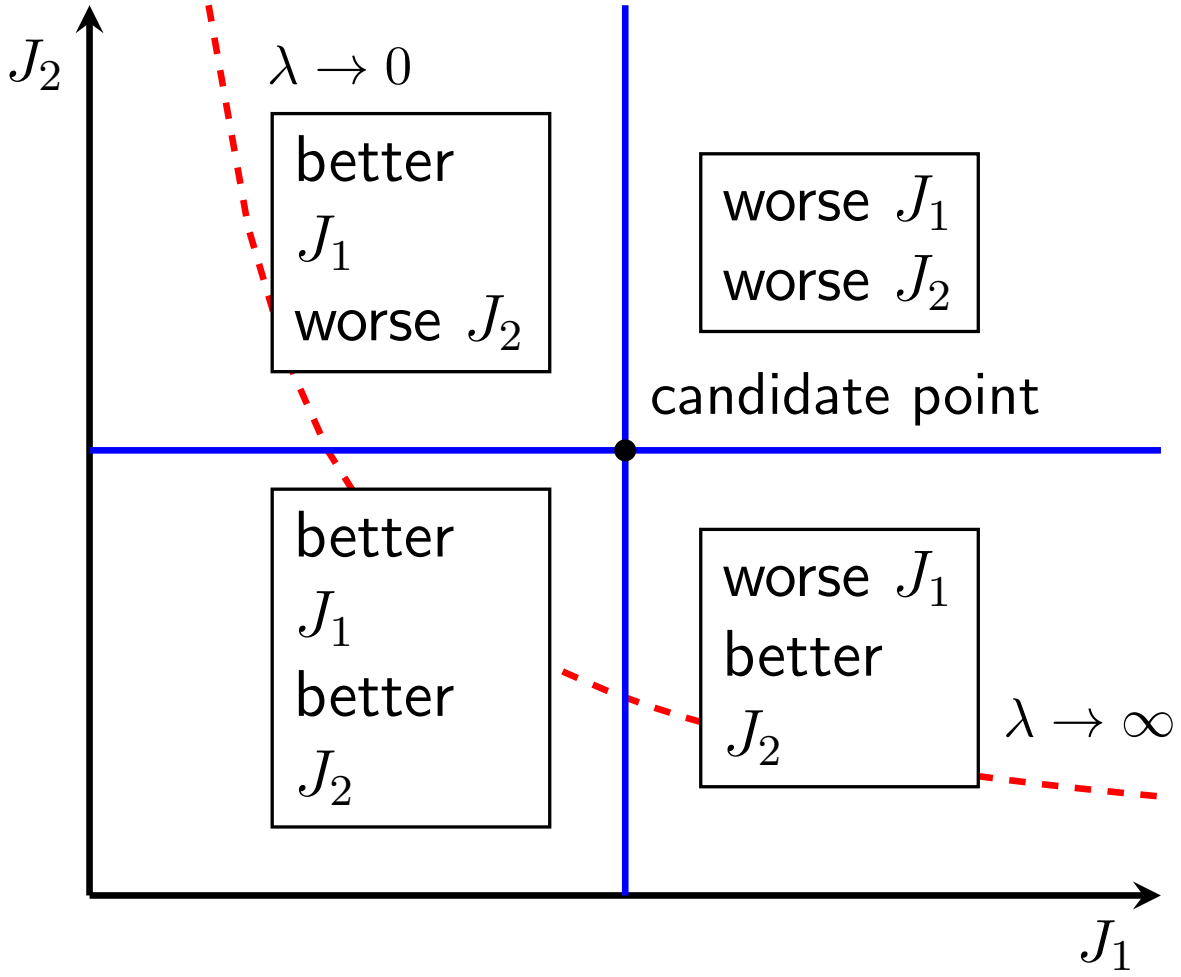
  where $\lambda > 0$ is a (fixed) *tradeoff parameter*.
- Then tune λ to explore possible results.
  - When $\lambda \to 0$, we place more weight on J_1
  - When $\lambda \to \infty$, we place more weight on J_2

Tradeoff analysis

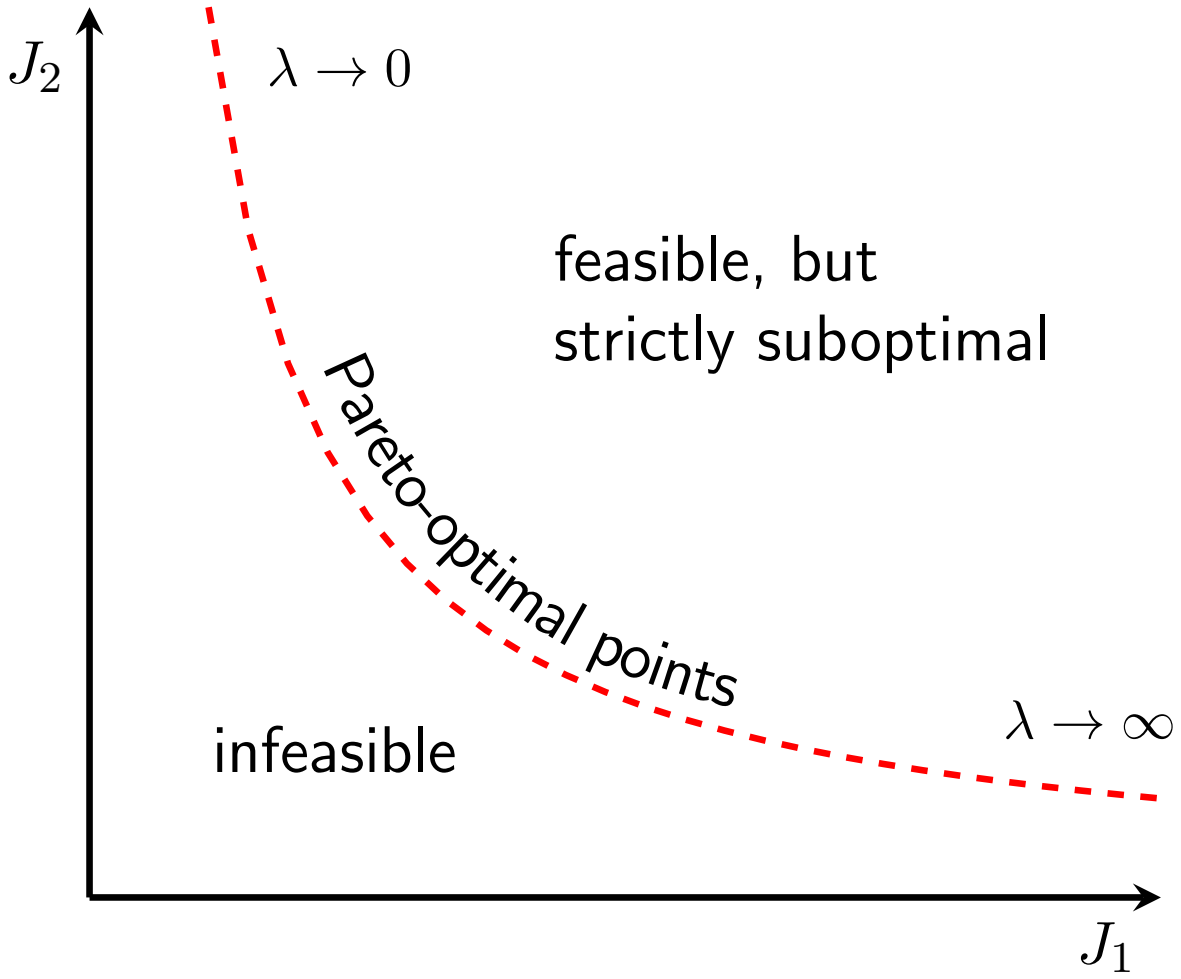
1. Choose values for λ (usually log-spaced). A useful command: `lambda = logspace(p,q,n)` produces n points logarithmically spaced between 10^p and 10^q.
2. For each λ value, find $\hat{x}_\lambda$ that minimizes $J_1 + \lambda J_2$.
3. For each $\hat{x}_\lambda$, also compute the corresponding J_1^λ and J_2^λ.
4. Plot $(J_1^\lambda, J_2^\lambda)$ for each λ and connect the dots.



Pareto curve



Pareto curve



CS/ECE/ISYE524: Introduction to Optimization – Convex Optimization Models

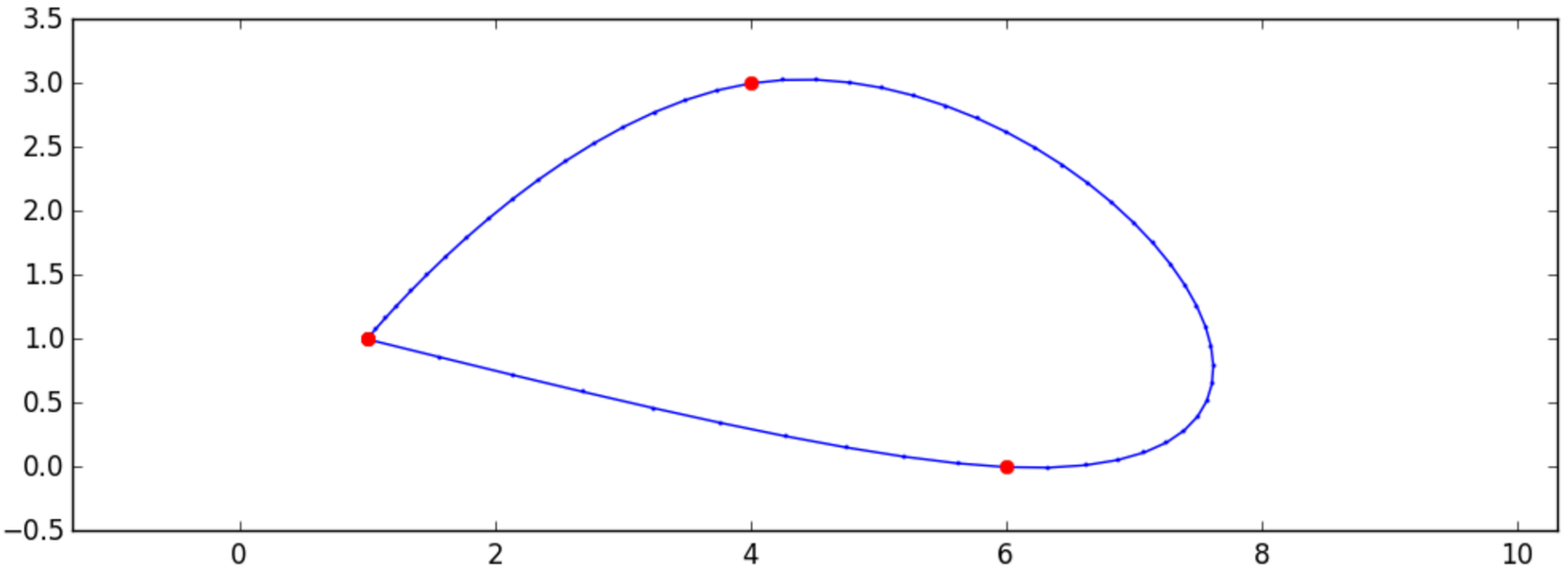
Jeff Linderoth

Department of Industrial and Systems Engineering
University of Wisconsin-Madison

April 1, 2024

Example: Anakin Goes Podracing

- Anakin Skywalker has a Pod, and he is practicing his pod-racing.
- We are given a set of k waypoint locations and times.
- The objective is to hit the waypoints at the prescribed times while minimizing fuel use.



Anakin's Dynamics

- Discretize time: $t = 0, 1, 2, \ldots, T$.
- Important variables: position x_t, velocity v_t, thrust u_t. (These are vectors in $\mathbb{R}^2$)
- Simplified model of the dynamics:

$$\begin{aligned} x_{t+1} &= x_t + v_t \\ v_{t+1} &= v_t + u_t \end{aligned} \qquad \text{for } t = 0, 1, \ldots, T-1$$

- We must choose $u_0, u_1, \ldots, u_T$.
- Initial position and velocity: $x_0 = 0$ and $v_0 = 0$.
- Waypoint constraints: $x_{t_i} = w_i$ for $i = 1, \ldots, k$.
- Minimize fuel use: $\|u_0\|^2 + \|u_1\|^2 + \cdots + \|u_T\|^2$

Anakin's First Model

First model: Hit the waypoints exactly

$$
\begin{aligned}
\underset{x_t, v_t, u_t}{\text{minimize}} \quad & \sum_{t=0}^{T} \|u_t\|^2 \\
\text{subject to:} \quad & x_{t+1} = x_t + v_t \quad \text{for } t = 0, 1, \ldots, T-1 \\
& v_{t+1} = v_t + u_t \quad \text{for } t = 0, 1, \ldots, T-1 \\
& x_0 = v_0 = 0 \\
& x_{t_i} = w_i \quad \text{for } i = 1, \ldots, k
\end{aligned}
$$

Julia model: Podracing.ipynb

Tradeoffs in Podracing

Second model: We are allowed to miss the waypoints (by a bit):

$$\begin{aligned} \underset{x_t, v_t, u_t}{\text{minimize}} \quad & \sum_{t=0}^{T} \|u_t\|^2 + \lambda \sum_{i=1}^{k} \|x_{t_i} - w_i\|^2 \\ \text{subject to:} \quad & x_{t+1} = x_t + v_t \quad \text{for } t = 0, 1, \ldots, T-1 \\ & v_{t+1} = v_t + u_t \quad \text{for } t = 0, 1, \ldots, T-1 \\ & x_0 = v_0 = 0 \end{aligned}$$

- λ controls the tradeoff between making u small and hitting all the waypoints.
- Tradeoff-Podracing.ipynb

Multi-objective tradeoff

- We can use a similar procedure if we have more than two costs we'd like to make small, e.g. J_1, J_2, J_3
- Choose parameters $\lambda > 0$ and $\mu > 0$. Then solve:

$$\begin{array}{ll} \underset{x}{\text{minimize}} & J_1(x) + \lambda J_2(x) + \mu J_3(x) \\ \text{subject to:} & \text{constraints} \end{array}$$

- Each $\lambda > 0$ and $\mu > 0$ yields a solution $\hat{x}_{\lambda,\mu}$.
- Can visualize tradeoff by plotting $J_3(\hat{x}_{\lambda,\mu})$ vs $J_2(\hat{x}_{\lambda,\mu})$ vs $J_1(\hat{x}_{\lambda,\mu})$ on a 3D plot. You then obtain a *Pareto surface*.

Minimum-norm as a regularization

- When $Ax = b$ is underdetermined (A is wide), we can resolve ambiguity by adding a cost function, e.g. *min-norm* LS:

$$\begin{array}{ll} \underset{x}{\text{minimize}} & \|x\|^2 \\ \text{subject to:} & Ax = b \end{array}$$

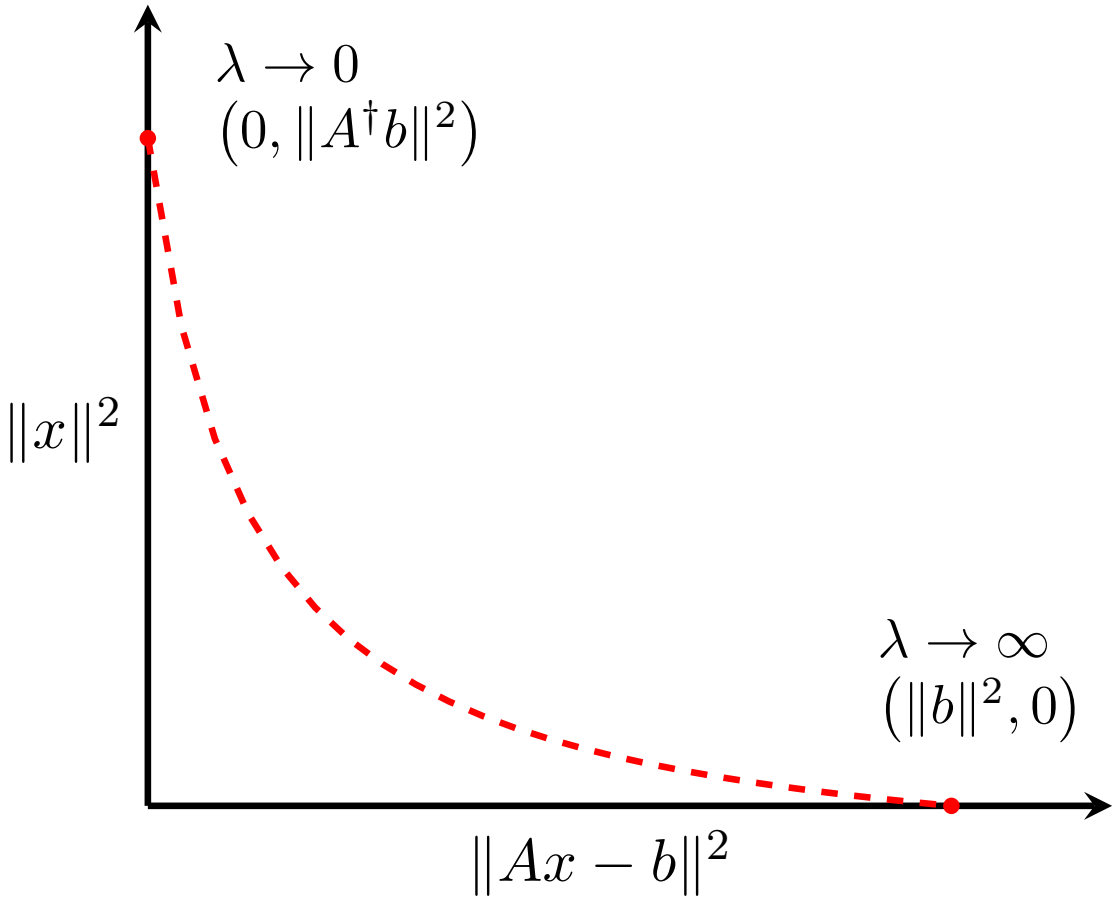
- Alternative approach: express it as a tradeoff!

$$\underset{x}{\text{minimize}} \quad \|Ax - b\|^2 + \lambda \|x\|^2$$

- Tradeoffs of this type are called regularization and λ is called the *regularization parameter* or *regularization weight*
- If we let $\lambda \to \infty$, we just obtain $\hat{x} = 0$
- If we let $\lambda \to 0$, we obtain the minimum-norm solution!

Tradeoff visualization

$$\underset{x}{\text{minimize}} \quad \|Ax - b\|^2 + \lambda \|x\|^2$$



Regularization

Regularization: Additional penalty term added to the cost function to encourage a solution with desirable properties.

Regularized least squares:

$$\underset{x}{\text{minimize}} \quad \|Ax - b\|^2 + \lambda R(x)$$

- $R(x)$ is the regularizer (penalty function)
- λ is the regularization parameter
- The model has different names depending on $R(x)$.

Regularized least squares turns out to be important in many contexts!

Regularization

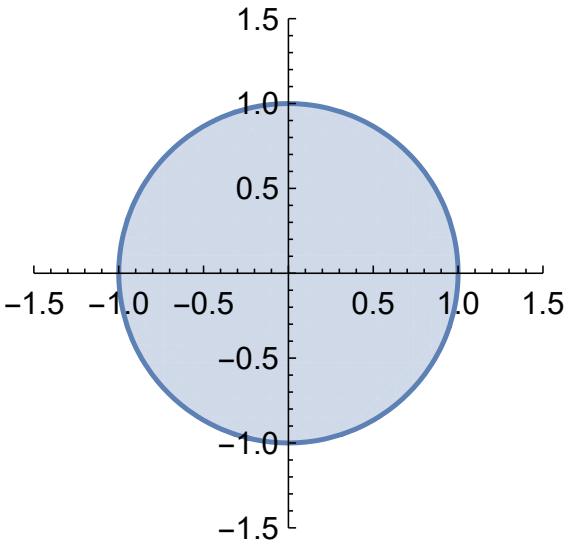
$$\underset{x}{\text{minimize}} \quad \|Ax - b\|^2 + \lambda R(x)$$

1. If $R(x) = \|x\|^2 = x_1^2 + x_2^2 + \cdots + x_n^2$
   It is called: L_2 *regularization*, *Tikhonov regularization*, or *Ridge regression* depending on the application. It has the effect of *smoothing* the solution.
2. If $R(x) = \|x\|_1 = |x_1| + |x_2| + \cdots + |x_n|$
   It is called: L_1 *regularization* or *LASSO*. It has the effect of *sparsifying* the solution ($\hat{x}$ will have few nonzero entries).
3. $R(x) = \|x\|_\infty = \max\{|x_1|, |x_2|, \ldots, |x_n|\}$
   It is called L_∞ *regularization* and it has the effect of *equalizing* the solution (makes many components tie for the max absolute value).

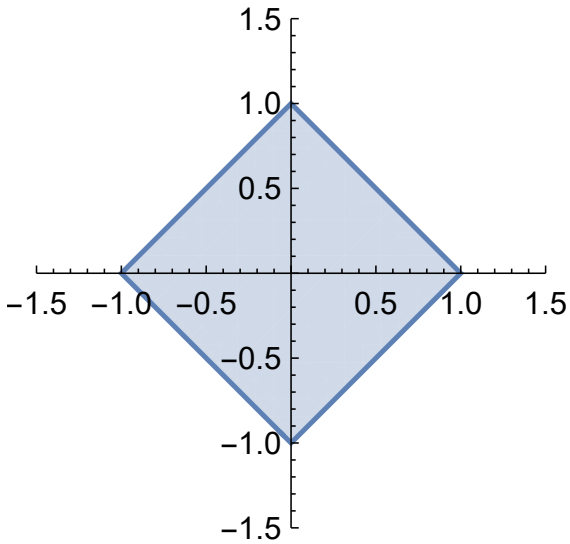
Norm balls

For a norm $\|\cdot\|_p$, the **norm ball** of radius r is the set:

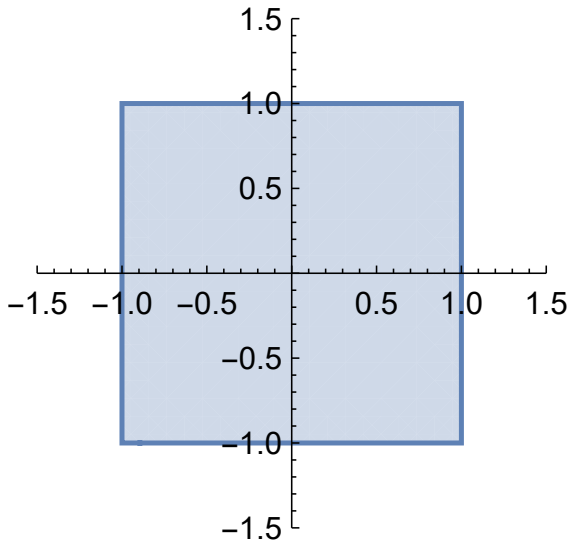
$$B_r = \{x \in \mathbb{R}^n \mid \|x\|_p \leq r\}$$



$$\|x\|_2 \leq 1$$
$$x^2 + y^2 \leq 1$$



$$\|x\|_1 \leq 1$$
$$|x| + |y| \leq 1$$



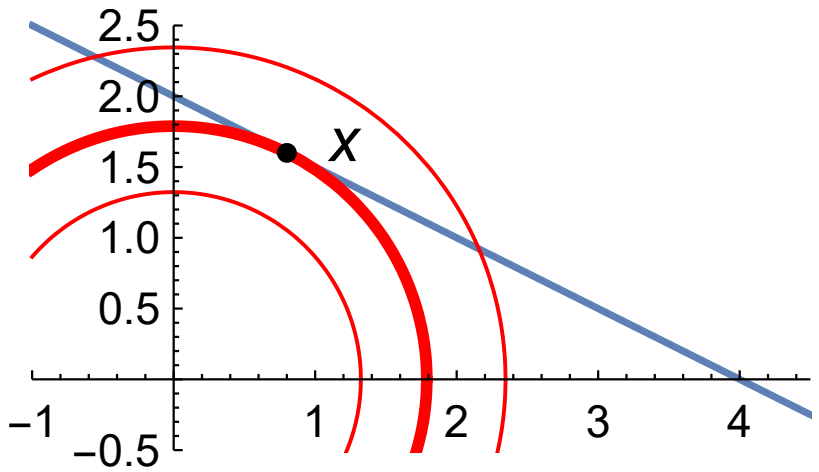
$$\|x\|_\infty \leq 1$$
$$\max\{|x|, |y|\} \leq 1$$

Simple example

Consider the minimum-norm problem for different norms:

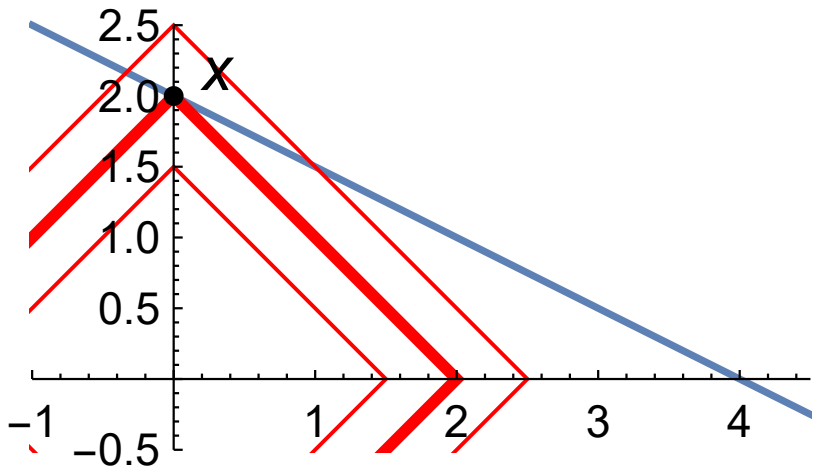
$$\begin{array}{ll} \underset{x}{\text{minimize}} & \|x\|_p \\ \text{subject to:} & Ax = b \end{array}$$

- set of solutions to $Ax = b$ is an affine subspace
- solution is point belonging to smallest norm ball
- for $p = 2$, this occurs at the perpendicular distance

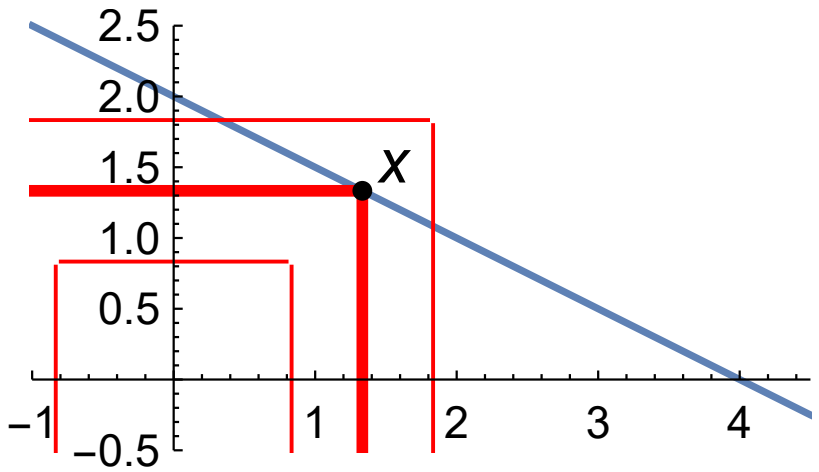


Simple example

- for $p = 1$, this occurs at one of the axes.
- *sparsifying* behavior



- for $p = \infty$, this occurs at corners or edges, where many components tie for max value
- *equalizing* behavior



Example: hovercraft revisited (simpler 1D case)

One-dimensional version of the hovercraft problem:

- Start at $x_1 = 0$ with $v_1 = 0$ (at rest at position zero)
- Finish at $x_{50} = 100$ with $v_{50} = 0$ (at rest at position 100)
- Same simple dynamics as before:

$$\begin{aligned} x_{t+1} &= x_t + v_t \\ v_{t+1} &= v_t + u_t \end{aligned} \qquad \text{for: } t = 1, 2, \ldots, 49$$

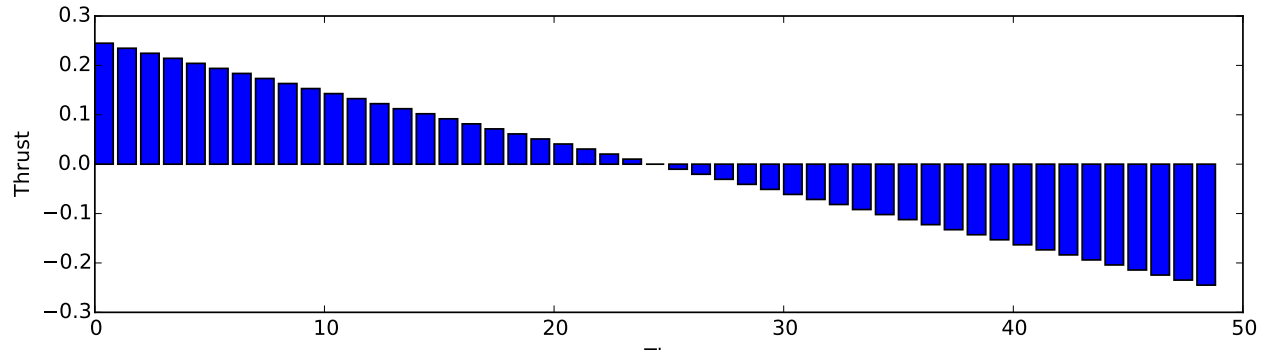
- Decide thruster inputs $u_1, u_2, \ldots, u_{49}$.
- This time: minimize $\|u\|_p$

Example: hovercraft revisited

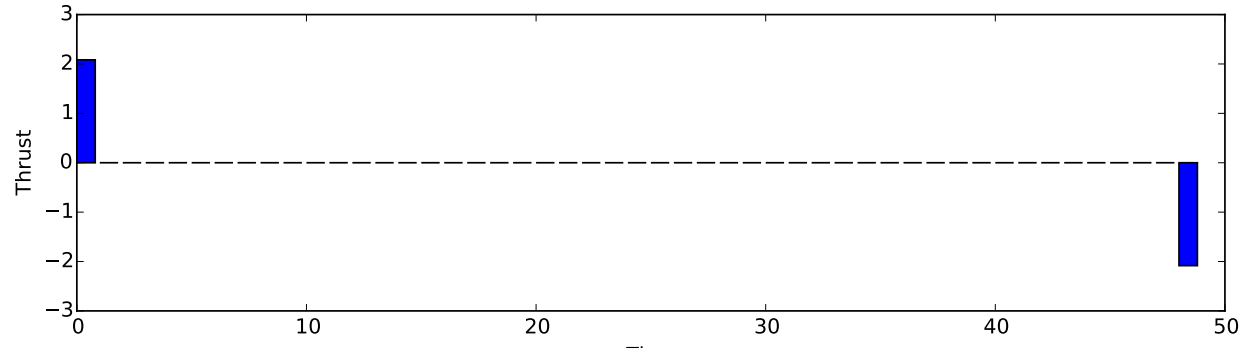
$$
\begin{array}{lll}
\underset{x_t, v_t, u_t}{\text{minimize}} & \|u\|_p & \\
\text{subject to:} & x_{t+1} = x_t + v_t & \text{for } t = 1, \ldots, 49 \\
& v_{t+1} = v_t + u_t & \text{for } t = 1, \ldots, 49 \\
& x_1 = 0, \quad x_{50} = 100 & \\
& v_1 = 0, \quad v_{50} = 0 &
\end{array}
$$

- This model has 150 variables, but very easy to understand.

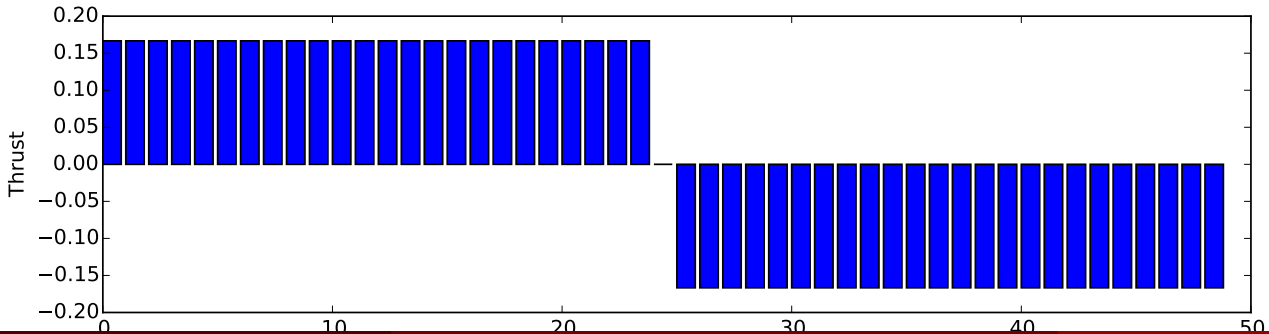
1. Minimizing $\|u\|_2^2$ (smooth)



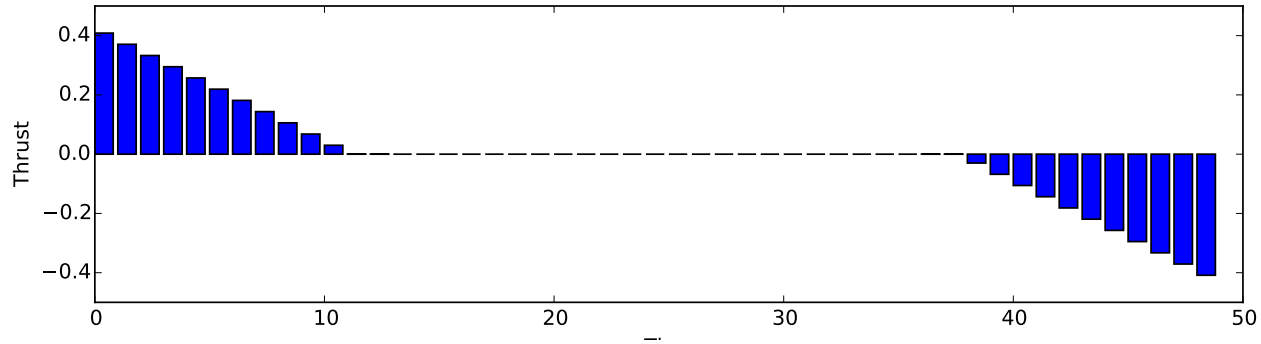
2. Minimizing $\|u\|_1$ (sparse)



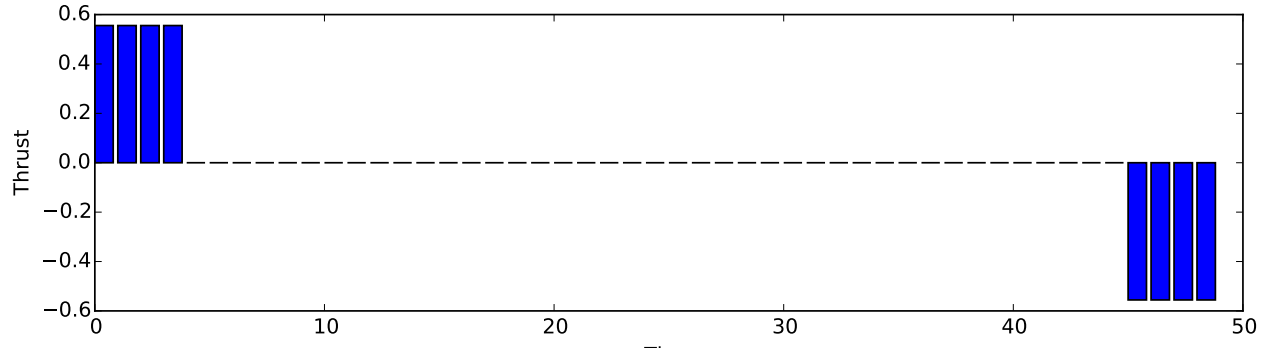
3. Minimizing $\|u\|_\infty$ (equalized)



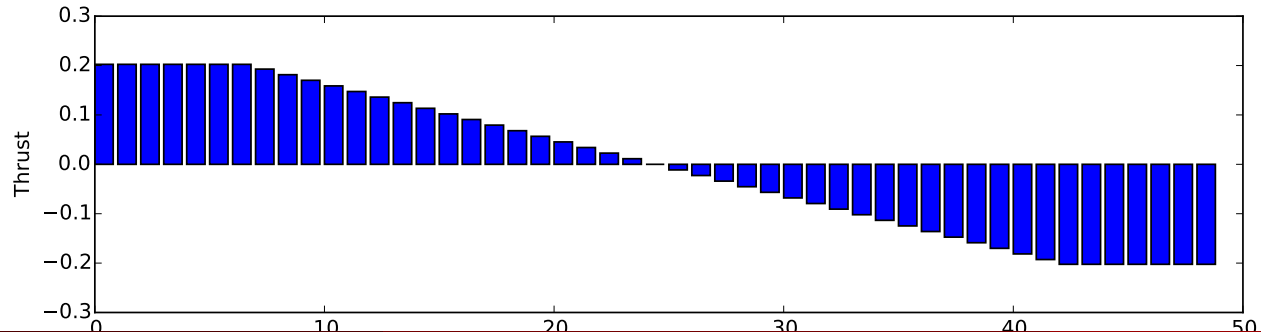
1. Minimizing $\|u\|_2^2 + \lambda\|u\|_1$ (smooth and sparse)



2. Minimizing $\|u\|_\infty + \lambda\|u\|_1$ (equalized and sparse)



3. Minimizing $\|u\|_2^2 + \lambda\|u\|_\infty$ (equalized and smooth)



Hierarchical Optimization

- One popular way for dealing with multiple objectives or goals is to combine then into objective using weights or penalties.
- This can work.
- But I don't recommend it in some cases...
  - If the quantities you are combining are in different units
  - If you have significantly more than two objectives

- Instead, often decision-makers have priorities for the individual goals
- In this case you can solve the problem in a hierarchical manner

Hierarchical Optimization

- Suppose we wish to make some decisions $x \in X$, but we have three different goals/objectives:
  1. $\min f_1(x)$
  2. $\max f_2(x)$
  3. $\min f_3(x)$
- In hierarchical optimization, we solve the objectives in the given order.
- As we move down the hierarchy, we impose the constraint that we do (almost?) as well with respect to the previous objectives.
- This of course generalizes to any number of goals that you can put in a priority order

Hierarchical Optimization

1. First, solve for the highest priority objective

$$z_1^* := \min_{x \in X} f_1(x)$$

2. Next, constrained to do (almost) as well with respect to first objective, solve for second priority:

$$z_2^* := \max_{x \in X} \{ f_2(x) : f_1(x) \leq (1+\varepsilon) z_1^* \}$$

3. Finally, solve final objective constrained to do (almost) as well with respect to first two objectives:

$$\min_{x \in X} \{ f_3(x) : f_1(x) \leq (1+\varepsilon) z_1^*, f_2(x) \geq (1-\varepsilon) z_2^* \}$$

The Upshot

Final solution is best with respect to the third objective that is within ε% of the best for the first two objectives

Annakin Again—Constrained Waypoints

- Suppose now that we only wish to make sure that we get "close enough" to the waypoints, which we measure by saying that each component of the position is within β units away

$$
\begin{aligned}
F^* := \underset{x,v,u}{\text{minimize}} \quad & \sum_{t=0}^{T} \|u_t\|^2 \\
\text{subject to:} \quad & x_{t+1} = x_t + v_t && \text{for } t = 0, 1, \ldots, T-1 \\
& v_{t+1} = v_t + u_t && \text{for } t = 0, 1, \ldots, T-1 \\
& \|x_{t_i} - w_i\|_\infty \leq \beta && \text{for } i = 1, \ldots, k \\
& x_0 = v_0 = 0
\end{aligned}
$$

Tradeoff-Podracing-Constraint.ipynb

Hierarchical—Finish Slow

- Now suppose that for a fixed value of β, we have a secondary objective to be going as slow as possible at the end of the race
- But we also want to make sure that we won't use too much more fuel than we do in an optimal fuel plan
- Tradeoff-Podracing-Hierarchical.ipynb

$$
\begin{array}{lll}
\underset{x,v,u}{\text{minimize}} & \|v_T\|^2 & \\
\text{subject to:} & x_{t+1} = x_t + v_t & \text{for } t = 0, 1, \ldots, T-1 \\
& v_{t+1} = v_t + u_t & \text{for } t = 0, 1, \ldots, T-1 \\
& \|x_{t_i} - w_i\|_\infty \leq \beta & \text{for } i = 1, \ldots, k \\
& x_0 = v_0 = 0 & \\
& \displaystyle\sum_{t=1}^{T} \|u_t\|^2 \leq (1+\varepsilon) F^* &
\end{array}
$$

CS/ECE/ISYE524: Introduction to Optimization – Convex Optimization Models

Jeff Linderoth

Department of Industrial and Systems Engineering
University of Wisconsin-Madison

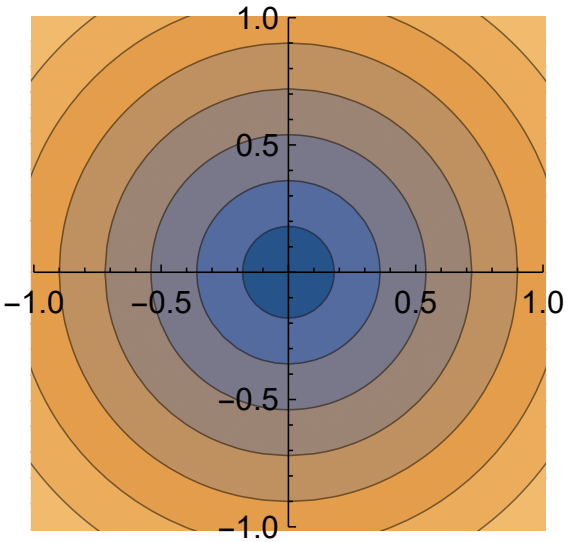
April 1, 2024

Ellipsoids

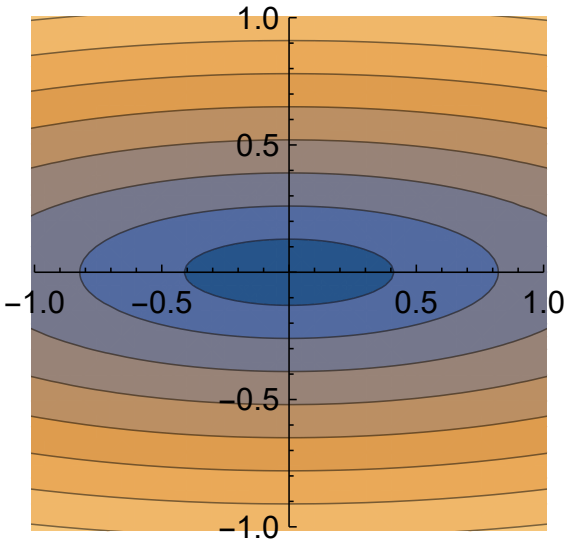
- For linear constraints, the set of x satisfying $c^{\mathsf{T}}x = b$ is a *hyperplane* and the set $c^{\mathsf{T}}x \leq b$ is a *halfspace*.
- For quadratic constraints, the set of x satisfying $x^{\mathsf{T}}Qx \leq b$ is an *ellipsoid* if $Q \succ 0$.
- If $Q \succ 0$, then $x^{\mathsf{T}}Qx \leq b \iff \|Q^{1/2}x\|^2 \leq b$.
- (Recall that if we write the eigenvalue decomposition $Q = U\Lambda U^T$, then $Q^{1/2} = U\Lambda^{1/2}U^T$, where $\Lambda^{1/2}$ is the diagonal matrix whose diagonal entries are the square roots of the diagonals of Λ.)

Degenerate Ellipsoids

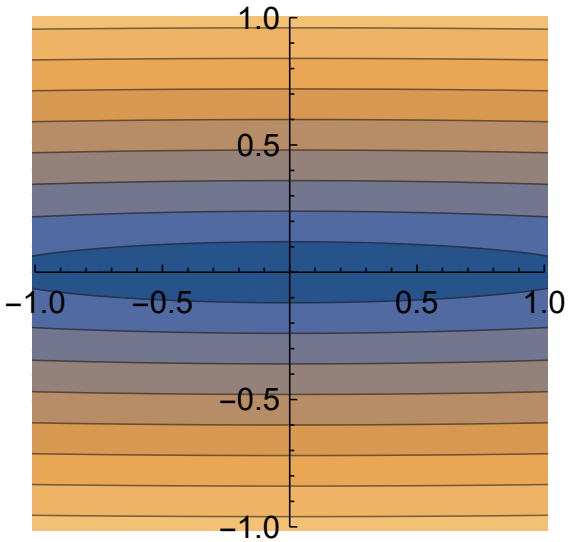
Ellipsoid axes have length $\frac{1}{\sqrt{\lambda_i}}$. When an eigenvalue is close to zero, contours are stretched in that direction.



$x^2 + y^2$



$\frac{1}{10}x^2 + y^2$



$\frac{1}{100}x^2 + y^2$

- Warmer colors = larger values
- If $\lambda_i = 0$, then $Q \succeq 0$. The ellipsoid $x^\mathsf{T} Q x \leq 1$ is *degenerate* (stretches out to infinity (is constant) in direction u_i).

Ellipsoids with linear terms

If $Q \succ 0$, then the quadratic form with extra linear term:

$$x^\mathsf{T} Q x + r^\mathsf{T} x + s$$

defines an *shifted* ellipsoid, whose center is not at 0. To see why, complete the square!

For scalars (ellipsoids in $\mathbb{R}^1$ are not very interesting), we have:

$$qx^2 + rx + s = q\left(x + \frac{r}{2q}\right)^2 + \left(s - \frac{r^2}{4q}\right)$$

In the matrix case, we have:

$$x^\mathsf{T} Q x + r^\mathsf{T} x + s = \left(x + \tfrac{1}{2}Q^{-1}r\right)^\mathsf{T} Q \left(x + \tfrac{1}{2}Q^{-1}r\right) + \left(s - \tfrac{1}{4}r^\mathsf{T} Q^{-1} r\right)$$

Ellipsoids with linear terms

- Therefore, the inequality $x^\mathsf{T}Qx + r^\mathsf{T}x + s \leq b$ is equivalent to:

$$\left(x + \tfrac{1}{2}Q^{-1}r\right)^\mathsf{T} Q \left(x + \tfrac{1}{2}Q^{-1}r\right) \leq \left(b - s + \tfrac{1}{4}r^\mathsf{T}Q^{-1}r\right)$$

- This is an ellipse centered at $-\frac{1}{2}Q^{-1}r$ — but its shape is still defined by the matrix Q.
  - Note that if $b - s + \frac{1}{4}r^\mathsf{T}Q^{-1}r < 0$, the original constraint makes the problem infeasible
- Writing this using the matrix square root, we have:

$$\left\|Q^{1/2}x + \tfrac{1}{2}Q^{-1/2}r\right\|^2 \leq \left(b - s + \tfrac{1}{4}r^\mathsf{T}Q^{-1}r\right)$$

$$\left\|Q^{1/2}x + \tfrac{1}{2}Q^{-1/2}r\right\| \leq \sqrt{\left(b - s + \tfrac{1}{4}r^\mathsf{T}Q^{-1}r\right)}$$

Quadratic programs

Quadratic program (QP) is like an LP, but with quadratic cost:

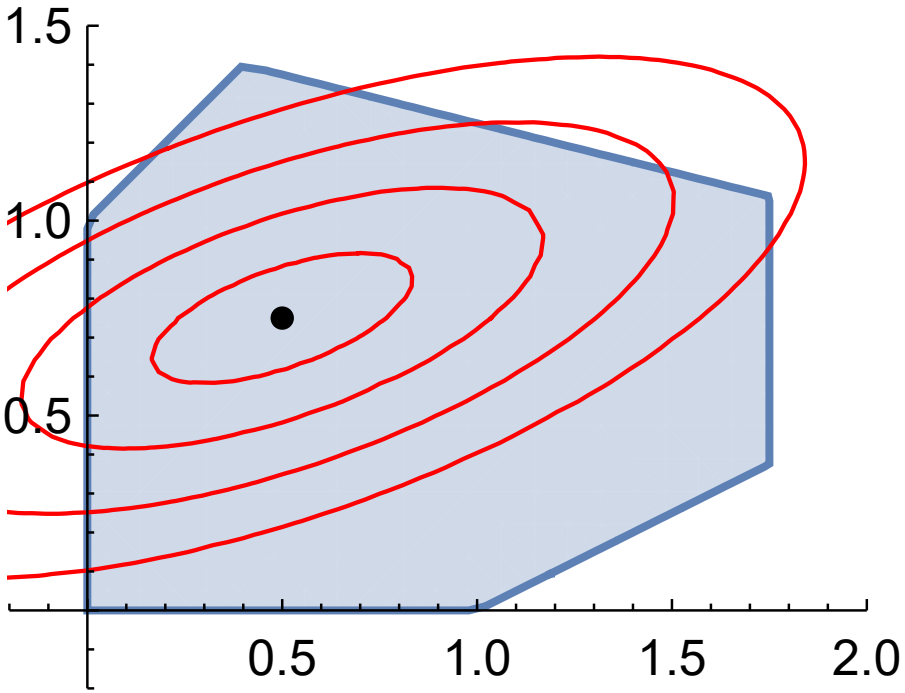
$$\begin{array}{ll} \underset{x}{\text{minimize}} & x^{\mathsf{T}} P x + q^{\mathsf{T}} x + r \\ \text{subject to:} & Ax \leq b \end{array}$$

Note: Constraints are still linear!

- If $P \succeq 0$, it is a *convex QP*
  - feasible set is a polyhedron
  - solution can be on boundary or in the interior
  - relatively easy to solve
- If $P \not\succeq 0$, it is **very hard** to solve in general.

Quadratic programs

$$\begin{array}{ll} \underset{x}{\text{minimize}} & x^T P x + q^T x + r \\ \text{subject to:} & Ax \leq b \end{array}$$

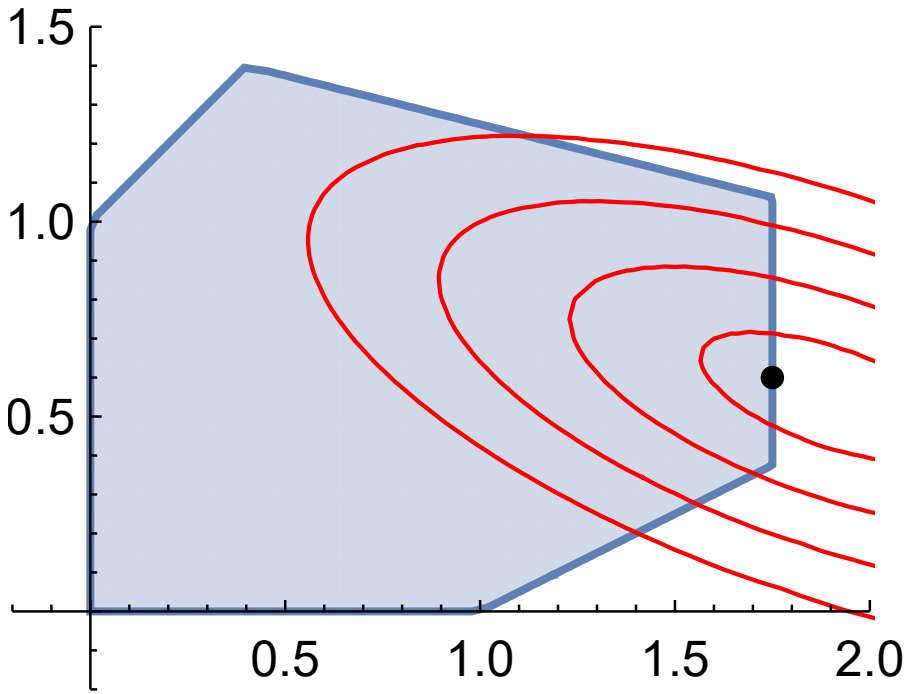


First case:

If the ellipsoid center is feasible, then it is also the optimal point.

Quadratic programs

$$\begin{array}{ll} \underset{x}{\text{minimize}} & x^T P x + q^T x + r \\ \text{subject to:} & Ax \leq b \end{array}$$



Second case:

If the ellipsoid center is infeasible, optimal point is on the boundary. Unlike LP, there may be no vertex solutions!

QCQPs

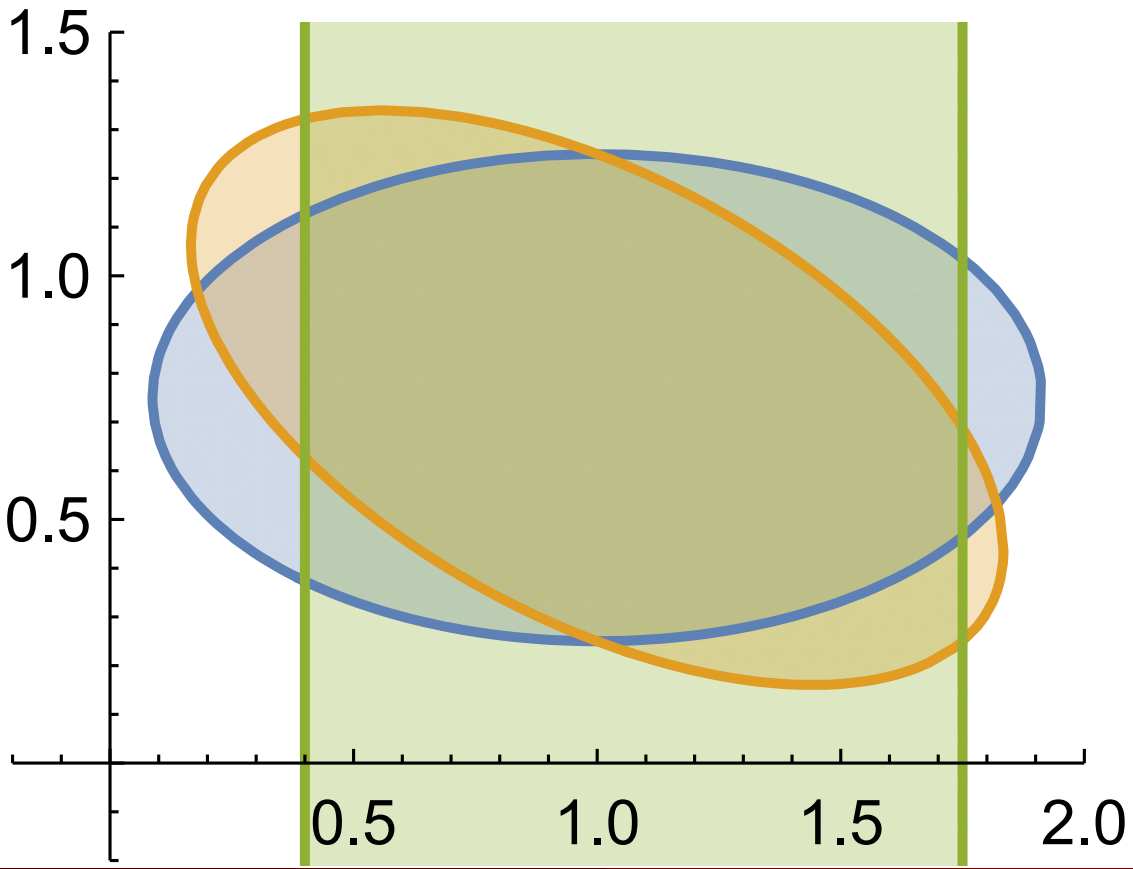
Quadratically constrained quadratic program (QCQP) has both a quadratic cost and quadratic constraints:

$$\begin{array}{ll} \underset{x}{\text{minimize}} & x^\mathsf{T} P_0 x + q_0^\mathsf{T} x + r_0 \\ \text{subject to:} & x^\mathsf{T} P_i x + q_i^\mathsf{T} x + r_i \leq 0 \quad \text{for } i = 1, \ldots, m \end{array}$$

- If $P_i \succeq 0$ for $i = 0, 1, \ldots, m$, it is a *convex QCQP*
  - feasible set is convex
  - solution can be on boundary or in the interior
  - relatively easy to solve
- If any $P_i \not\succeq 0$, the QCQP becomes **very hard** to solve.

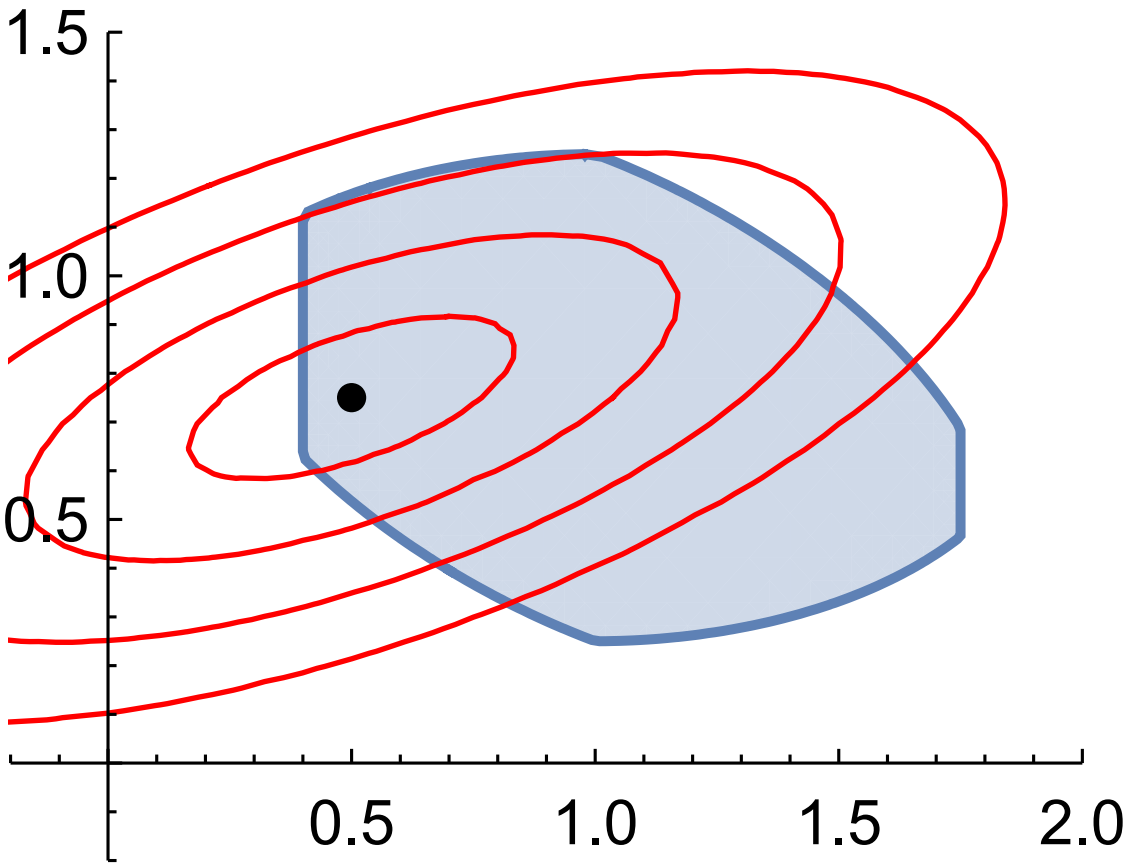
QCQPs

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & x^\mathsf{T} P_0 x + q_0^\mathsf{T} x + r_0 \\ \text{subject to:} \quad & x^\mathsf{T} P_i x + q_i^\mathsf{T} x + r_i \leq 0 \quad \text{for } i = 1, \ldots, m \end{aligned}$$



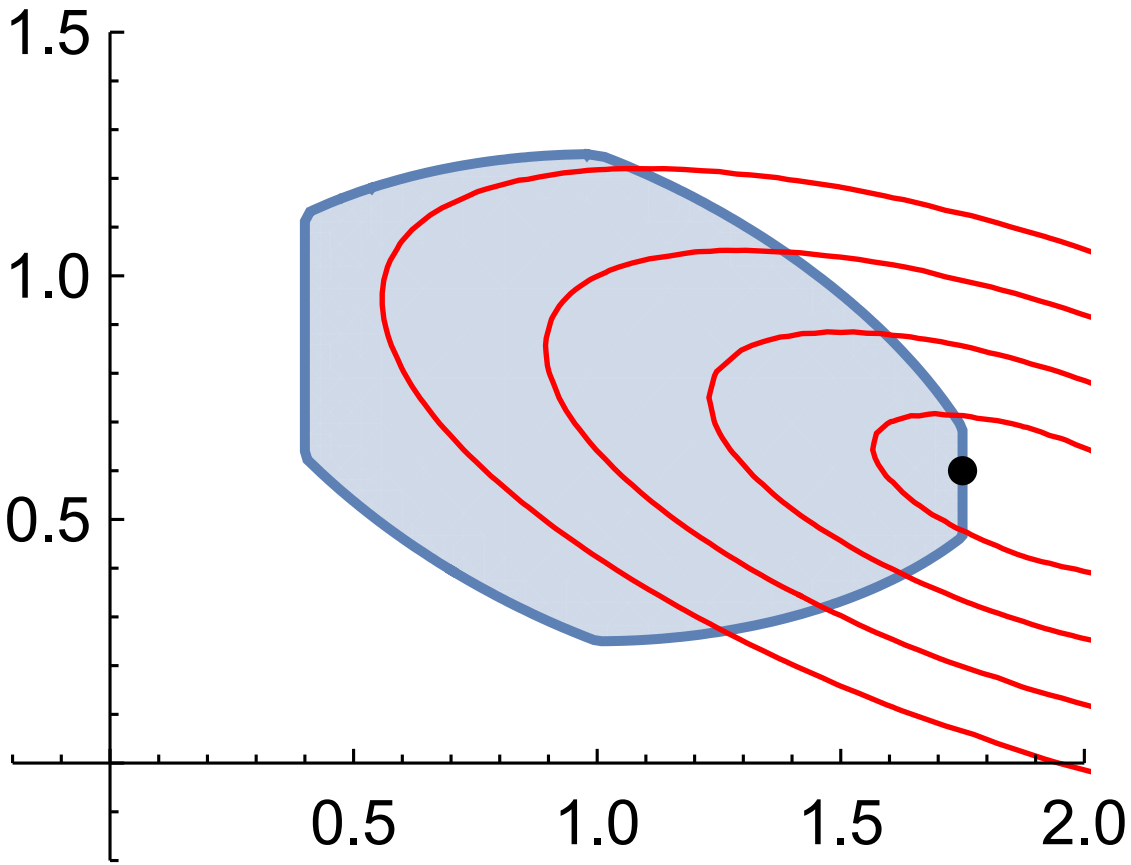
QCQPs

$$\begin{array}{ll} \underset{x}{\text{minimize}} & x^T P_0 x + q_0^T x + r_0 \\ \text{subject to:} & x^T P_i x + q_i^T x + r_i \leq 0 \quad \text{for } i = 1, \ldots, m \end{array}$$



QCQPs

$$\begin{array}{ll} \underset{x}{\text{minimize}} & x^T P_0 x + q_0^T x + r_0 \\ \text{subject to:} & x^T P_i x + q_i^T x + r_i \leq 0 \quad \text{for } i = 1, \ldots, m \end{array}$$

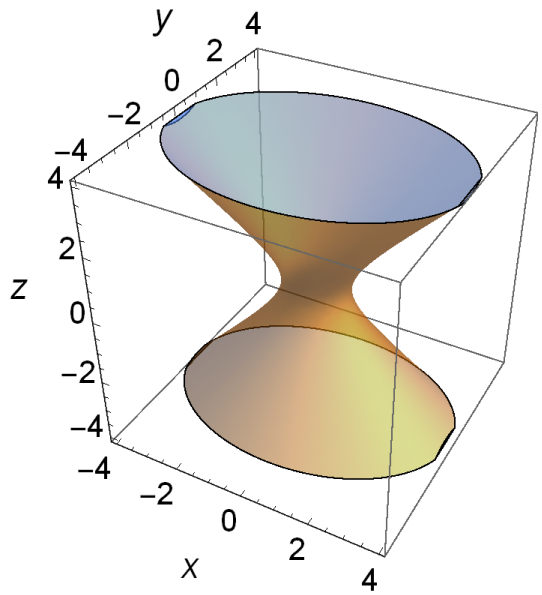


Difficult quadratic constraints

The following types of quadratic constraints make a problem nonconvex and generally difficult to solve (but not always).

Indefinite quadratic constraints.

- Example: $x^2 + 2y^2 - z^2 \leq 1$ corresponds to the nonconvex region on the right.
- **Note:** Be mindful of $\leq$ vs $\geq$! e.g. $x^2 + y^2 \geq 1$ is nonconvex.



Quadratic equalities.

- Using quadratic equalities, you can encode Boolean constraints. Example: $x^2 = 1$ is equivalent to $x \in \{-1, 1\}$. (There are many interesting problems with these kinds of variables!)

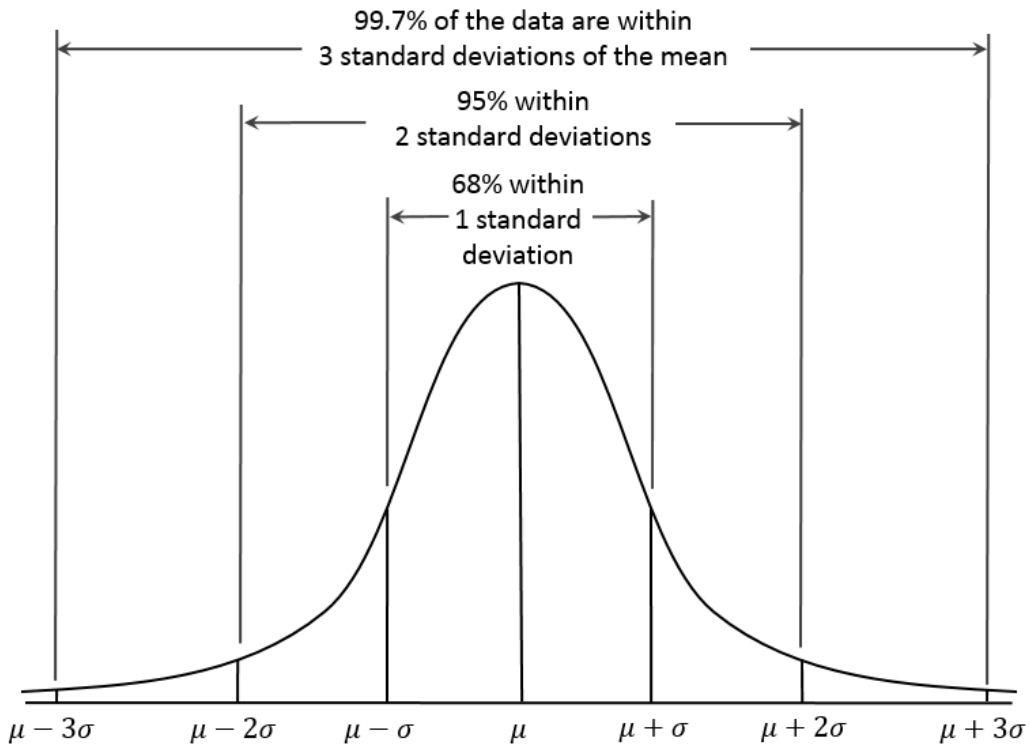
Where do quadratics commonly occur?

1. As a regularization or penalty term
   - $(\text{cost}) + \lambda \|x\|^2$: standard L_2 regularizer
   - $(\text{cost}) + \lambda\, x^{\mathsf{T}} Q\, x$ (with $Q \succ 0$) : weighted L_2 regularizer
2. Hard norm bounds on a decision variable
   - $\|x\|^2 \leq r$: a way to ensure that x doesn't get too big.
3. Allowing some tolerance in constraint satisfaction
   - $\|Ax - b\|^2 \leq e$: we allow a tolerance e.
4. Energy quantities (physics/mechanics)
   - examples: $\underset{\text{(kinetic)}}{\frac{1}{2}mv^2}$, $\underset{\text{(spring)}}{\frac{1}{2}kx^2}$, $\underset{\text{(capacitor)}}{\frac{1}{2}CV^2}$, $\underset{\text{(rotational)}}{\frac{1}{2}I\omega^2}$, $\underset{\text{(strain)}}{\frac{1}{2}VE\varepsilon^2}$.
5. Covariance constraints (statistics)

Example: portfolio optimization

We must decide how to invest our money, and we can choose between $i = 1, 2, \ldots, N$ different assets.

- Each asset can be modeled as a random variable (RV) with an expected return μ_i and a standard deviation σ_i.



- Standard deviation is a measure of uncertainty.

Example: portfolio optimization

If Z is the RV representing an asset:

- The expected return is $\mu = \mathbf{E}(Z)$ (expected value)
- The variance is $\mathbf{var}(Z) = \sigma^2 = \mathbf{E}\left((Z-\mu)^2\right)$
- The standard deviation is the square root of the variance.
- Sometimes use the notation $Z \sim (\mu, \sigma^2)$.

If $Z_1 \sim (\mu_1, \sigma_1^2)$ and $Z_2 \sim (\mu_2, \sigma_2^2)$ are two RVs

- The covariance is $\mathbf{cov}(Z_1, Z_2) = \mathbf{E}\left((Z_1-\mu_1)(Z_2-\mu_2)\right)$.
- Note that: $\mathbf{var}(Z) = \mathbf{cov}(Z, Z)$
- covariance measures tendency of RVs to move together.

Example: portfolio optimization

If $Z_1 \sim (\mu_1, \sigma_1^2)$ and $Z_2 \sim (\mu_2, \sigma_2^2)$, what is $x_1 Z_1 + x_2 Z_2$?

Calculating the mean:

$$\begin{aligned}\mathbf{E}(x_1 Z_1 + x_2 Z_2) &= x_1\mu_1 + x_2\mu_2 \\ &= \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}\end{aligned}$$

Calculating the variance:

$$\begin{aligned}\mathbf{var}(x_1 Z_1 + x_2 Z_2) &= \mathbf{E}\left(x_1(Z_1 - \mu_1) + x_2(Z_2 - \mu_2)\right)^2 \\ &= x_1^2\,\mathbf{var}(Z_1) + 2x_1x_2\,\mathbf{cov}(Z_1, Z_2) + x_2^2\,\mathbf{var}(Z_2) \\ &= \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} \mathbf{cov}(Z_1, Z_1) & \mathbf{cov}(Z_1, Z_2) \\ \mathbf{cov}(Z_2, Z_1) & \mathbf{cov}(Z_2, Z_2) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\end{aligned}$$

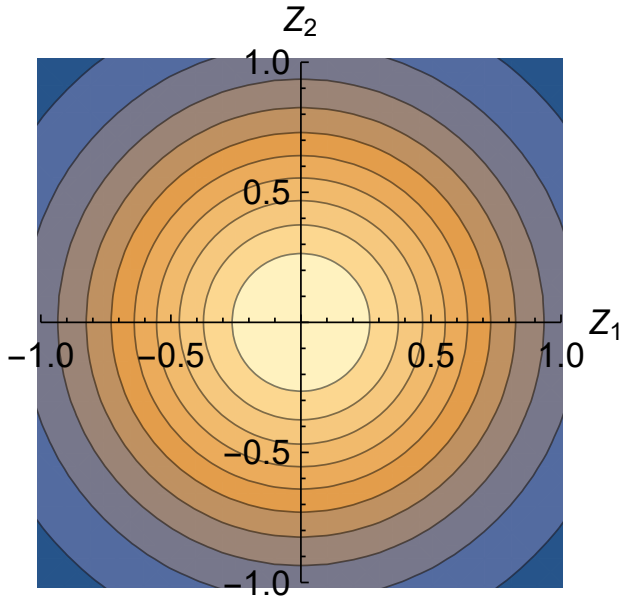
Example: portfolio optimization

If $Z_1, \ldots, Z_n$ are **jointly distributed** with:

- mean $\mu = \begin{bmatrix} \mathbf{E}(Z_1) \\ \vdots \\ \mathbf{E}(Z_n) \end{bmatrix}$
- covariance matrix $\Sigma = \begin{bmatrix} \mathbf{cov}(Z_1, Z_1) & \ldots & \mathbf{cov}(Z_1, Z_n) \\ \vdots & \ddots & \vdots \\ \mathbf{cov}(Z_n, Z_1) & \ldots & \mathbf{cov}(Z_n, Z_n) \end{bmatrix}$
- short form: $Z \sim (\mu, \Sigma)$.

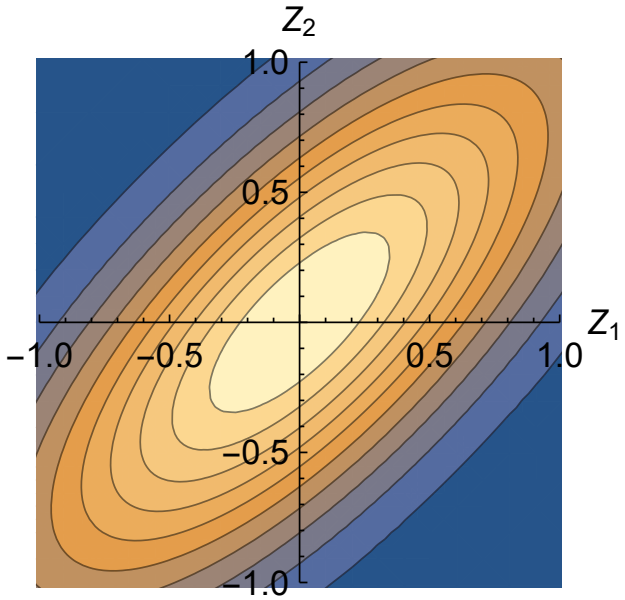
$$\sum_{i=1}^{n} x_i Z_i \sim \left(x^\mathsf{T}\mu\,,\, x^\mathsf{T}\Sigma x\right)$$

Example: portfolio optimization



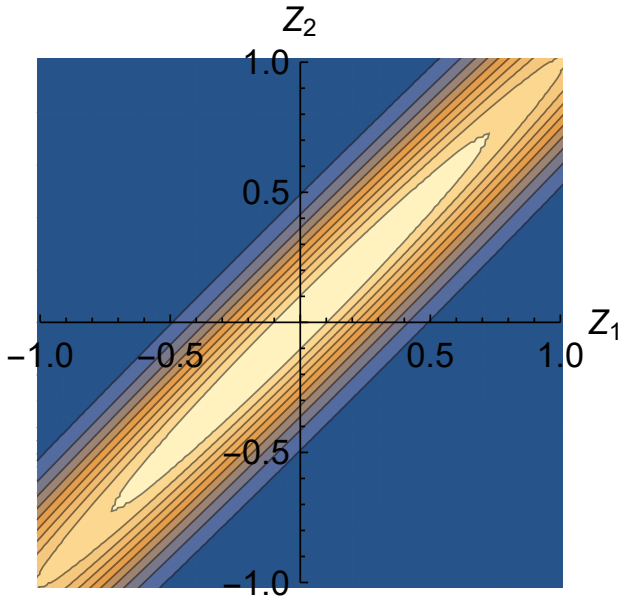
uncorrelated

$$\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$



somewhat correlated

$$\Sigma = \begin{bmatrix} 1 & .5 \\ .5 & 1 \end{bmatrix}$$



highly correlated

$$\Sigma = \begin{bmatrix} 1 & .99 \\ .99 & 1 \end{bmatrix}$$

Correlation is modeled by a **confidence ellipsoid**

Example: portfolio optimization

Example:

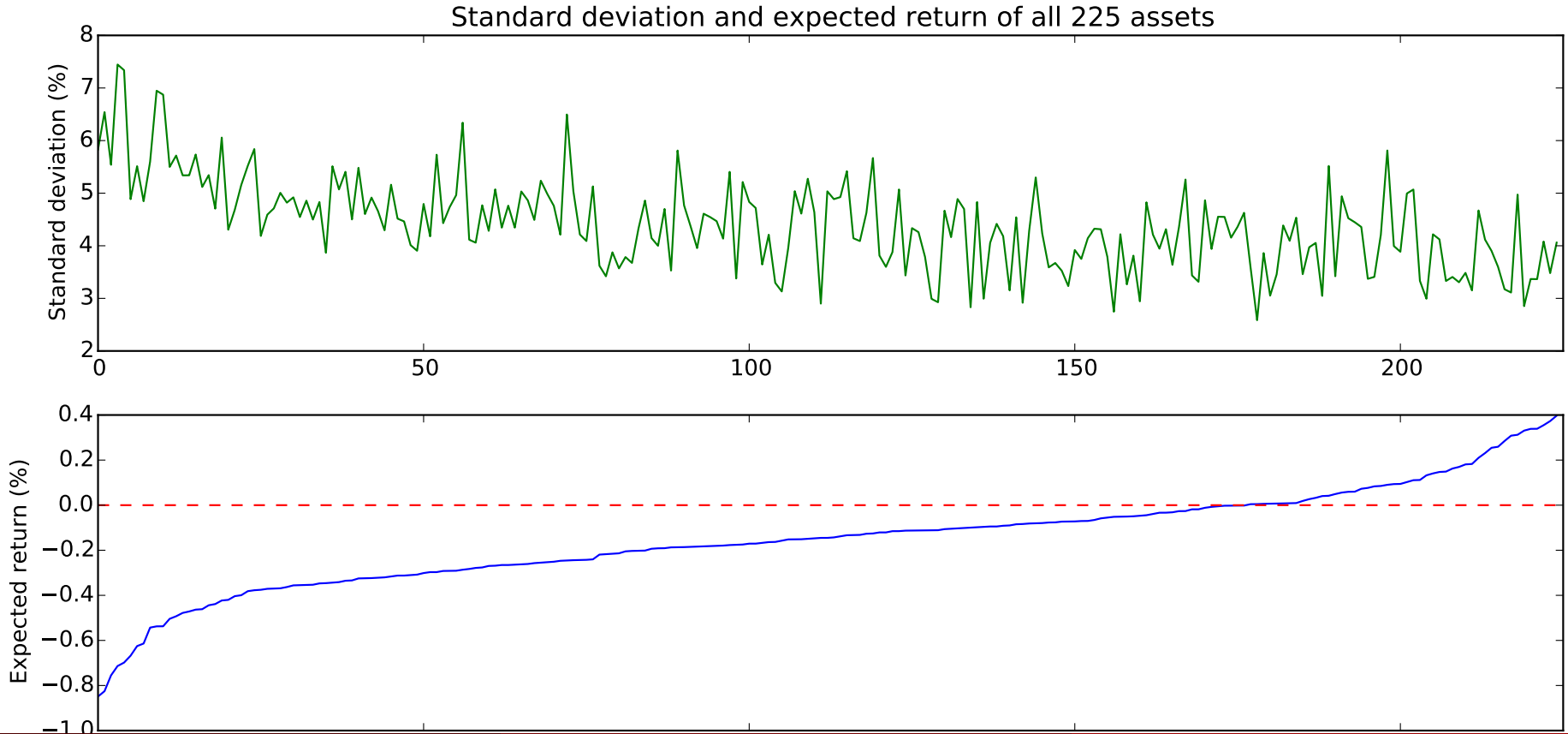
- There are 16 different stocks: $Z_1, \ldots, Z_{16}$. Each has expected return of 2% with standard deviation of 5%.
  You have \$100 in total to invest.
- If you invest in just one of them, you will earn $\$102 \pm \$5$.
- If the stocks are all correlated (e.g. all the same industry) and you invest evenly in all stocks, you still earn: $\$102 \pm \$5$.
- If the stocks are **uncorrelated** (e.g. very diverse) and you invest evenly in all stocks, the new variance is $16 \times (\frac{5}{16})^2$. Therefore, you will earn $\$102 \pm \$1.25$.

Julia code: Portfolio.ipynb

Example: portfolio optimization

Dataset containing 225 assets. How should we invest?

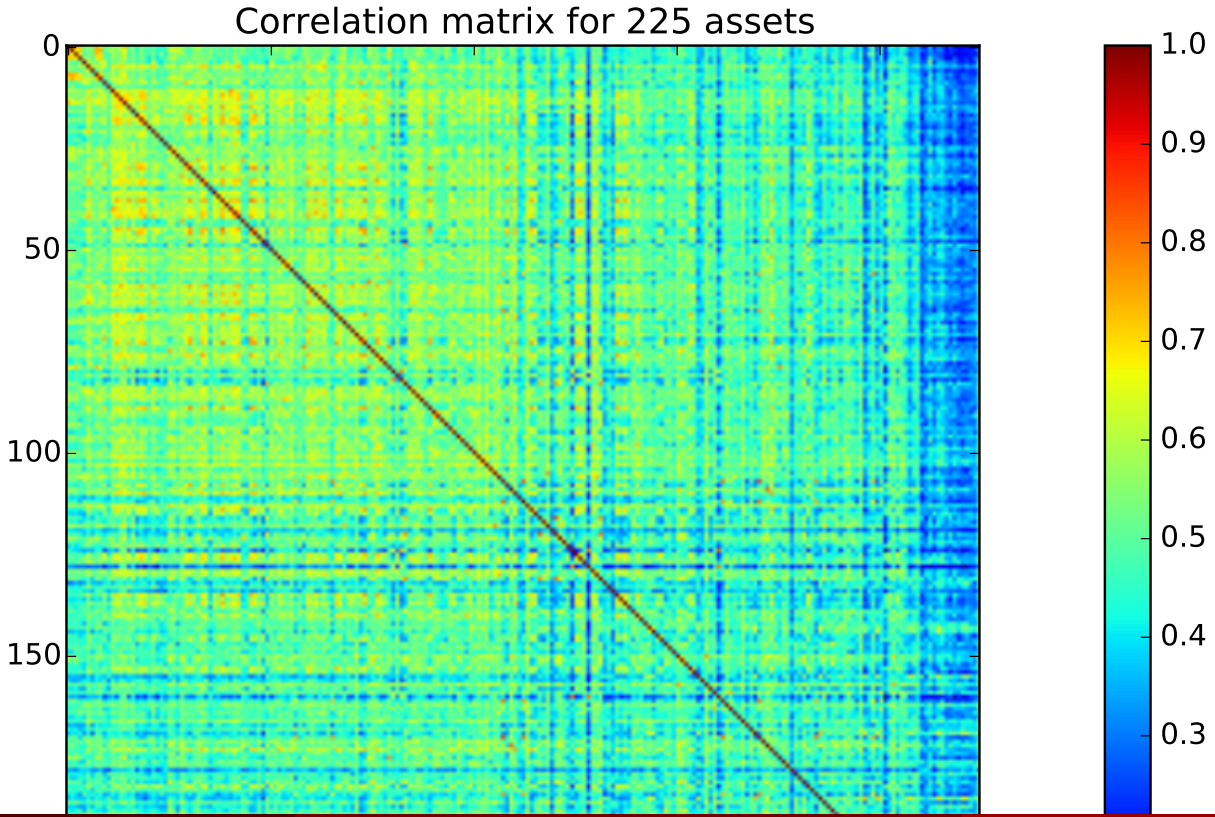
- We know the expected return μ_i for each asset
- We know the covariance Σ_{ij} for each pair of assets



Example: portfolio optimization

Dataset containing 225 assets. How should we invest?

- We know the expected return μ_i for each asset
- We know the covariance Σ_{ij} for each pair of assets



Example: portfolio optimization

Suppose we buy x_i of asset Z_i. We want:

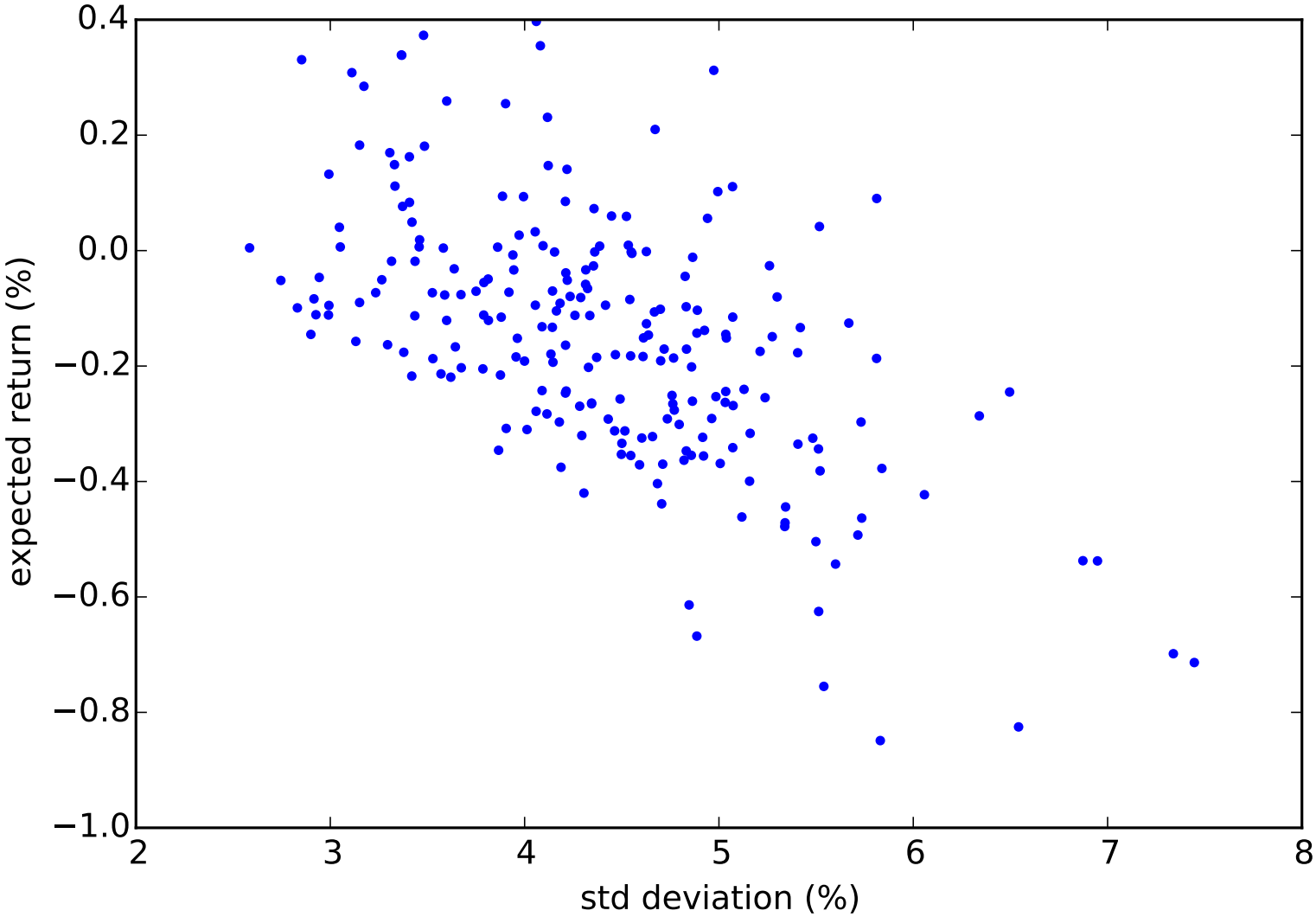
- A high total return. Maximize $x^\mathsf{T}\mu$.
- Low variance (risk). Minimize $x^\mathsf{T}\Sigma x$.

Pose the optimization problem as a tradeoff:

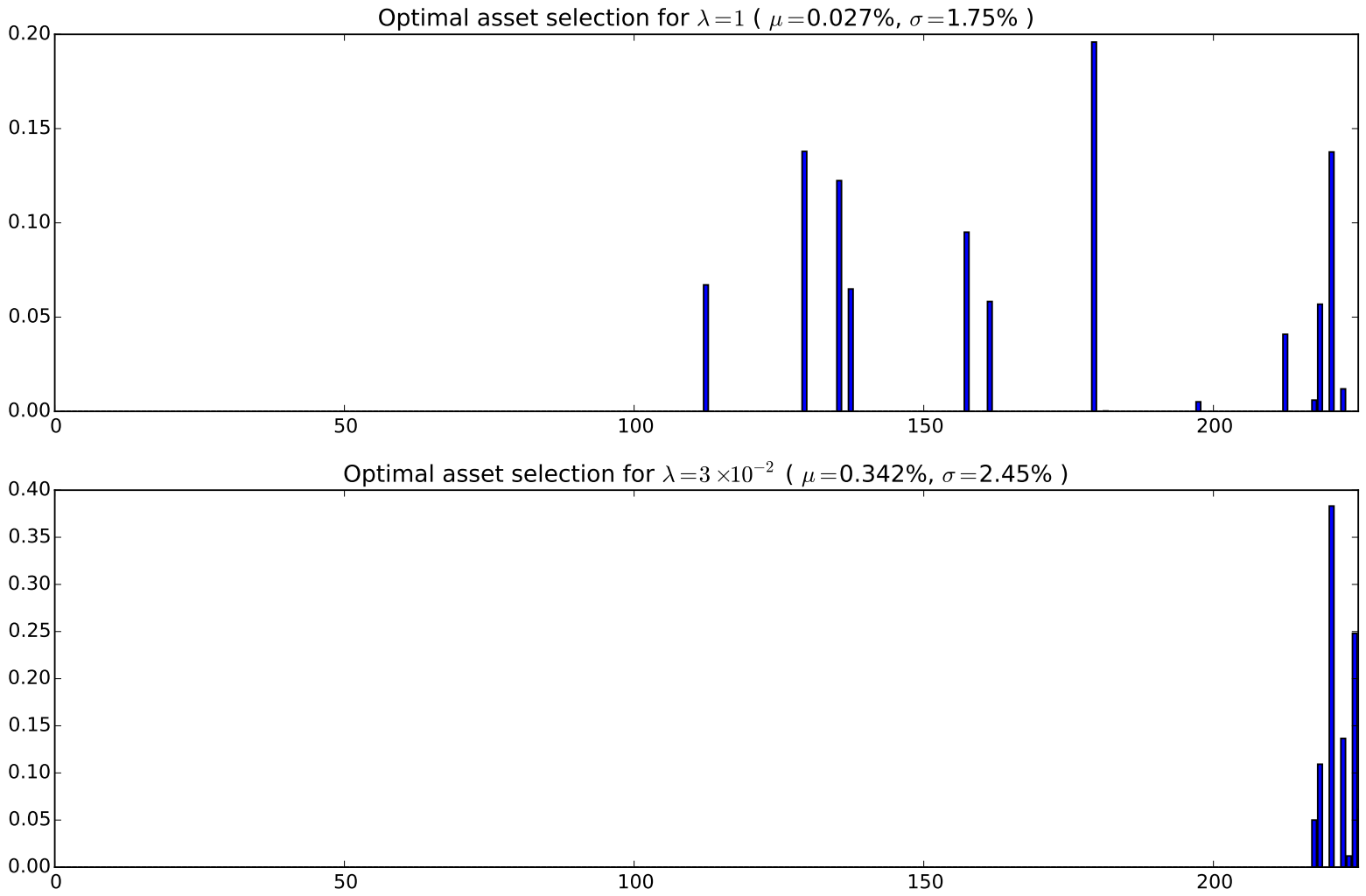
$$\begin{array}{ll} \underset{x}{\text{minimize}} & -x^\mathsf{T}\mu + \lambda x^\mathsf{T}\Sigma x \\ \text{subject to:} & x_1 + \cdots + x_{225} = 1 \\ & x_i \geq 0 \end{array}$$

Fun fact: This is the basic idea behind "Modern portfolio theory". Introduced by economist Harry Markowitz in 1952, for which he was awarded the Nobel Memorial Prize in Economics in 1990.

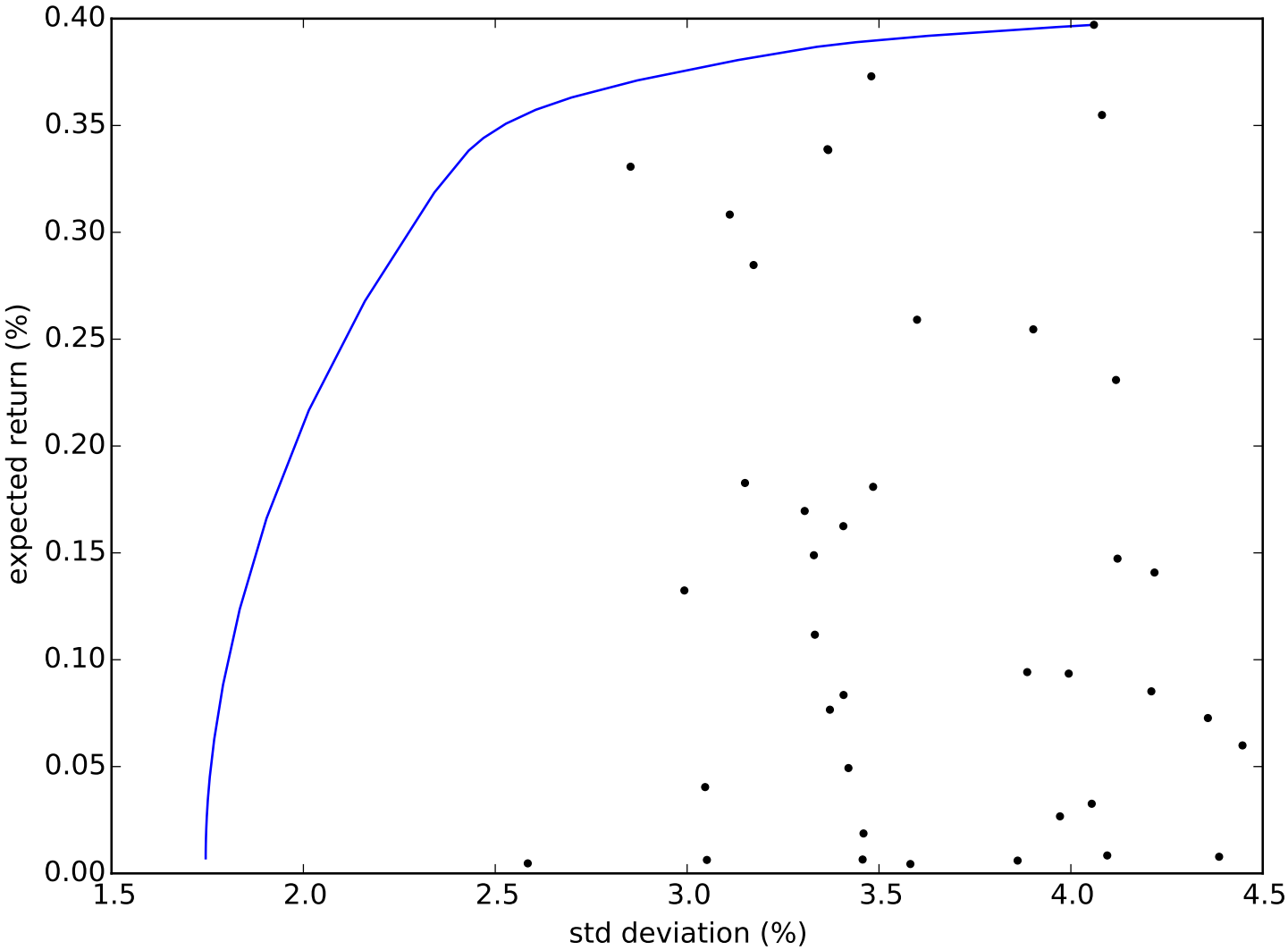
Asset Quality



Some Solutions



Efficient Frontier



CS/ECE/ISYE524: Introduction to Optimization – Convex Optimization Models

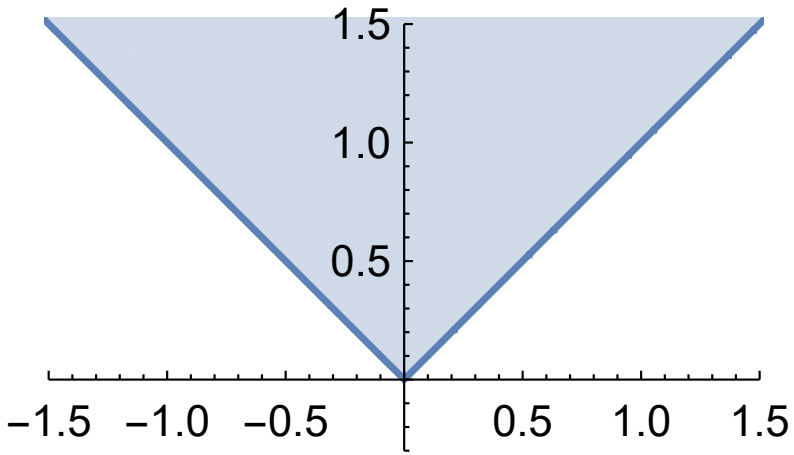
Jeff Linderoth

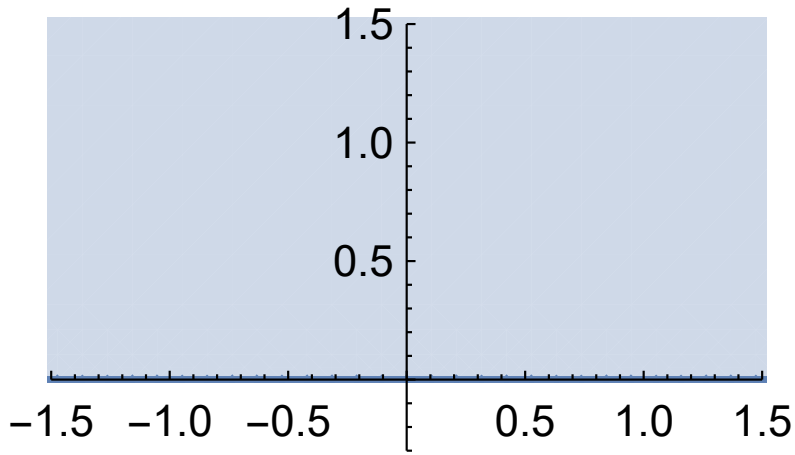
Department of Industrial and Systems Engineering
University of Wisconsin-Madison

April 1, 2024

What is a cone?

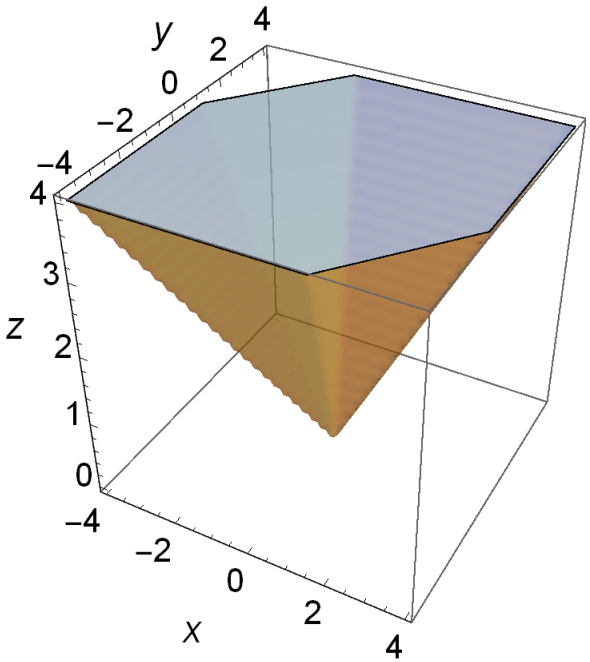
- A set of points $C \in \mathbb{R}^n$ is called a *cone* if $\alpha x \in C$ whenever $x \in C$ and $\alpha > 0$.
- A cone C is a *convex cone* if it is a convex set: $\lambda x + (1-\lambda)y \in C \; \forall \lambda \in [0,1]$ and $x, y \in C$
- Simple examples: $|x| \leq y$ and $y \geq 0$ are both cones.

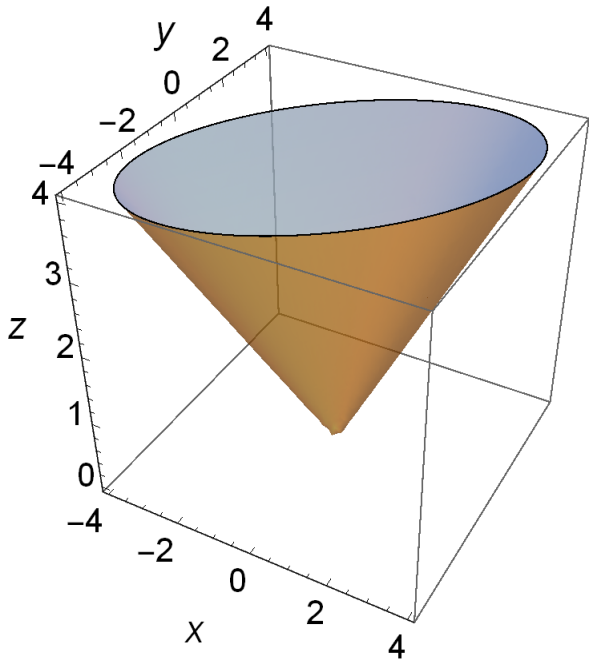


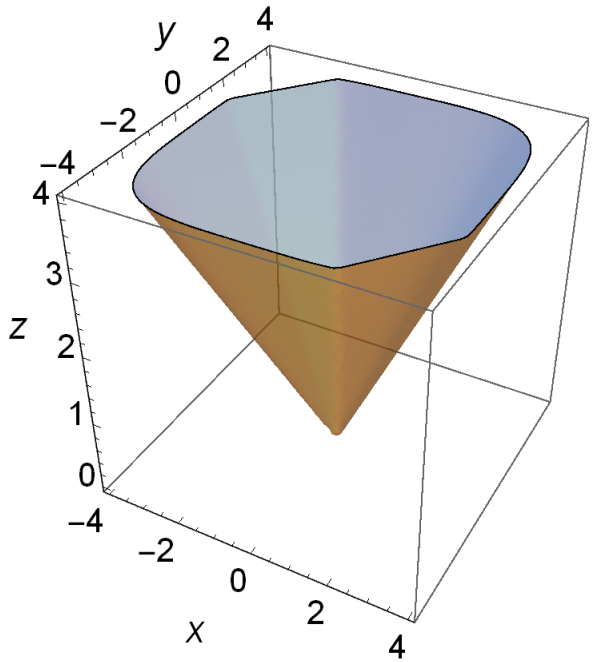


What is a cone?

- A *slice* of a cone is its intersection with an (affine) subspace.
- We are interested in *convex cones*, so all slices will be convex.
- Can be polyhedral, ellipsoidal, or something else...







Cone Programming

Cone Program

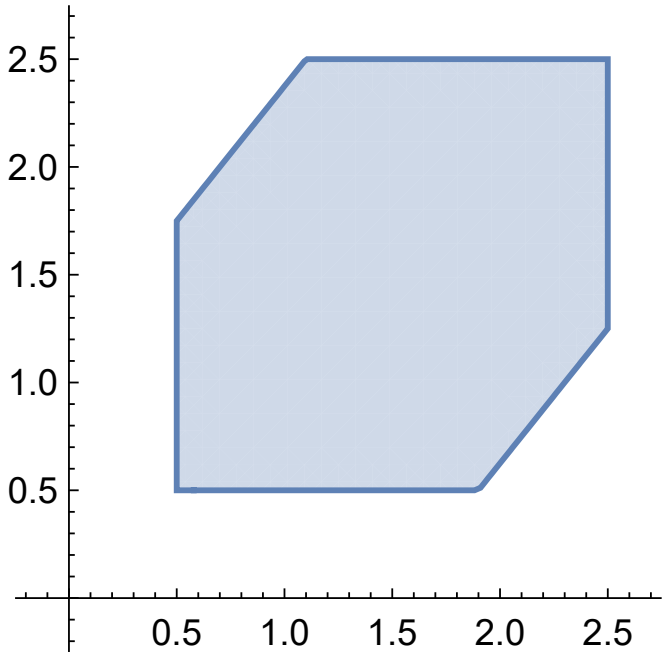
$$\min_{x \in \mathbb{R}^n} \quad c^T x$$
$$\text{s.t.} \quad A_i x + b_i \in C_i \quad \forall i = 1, \ldots, m$$

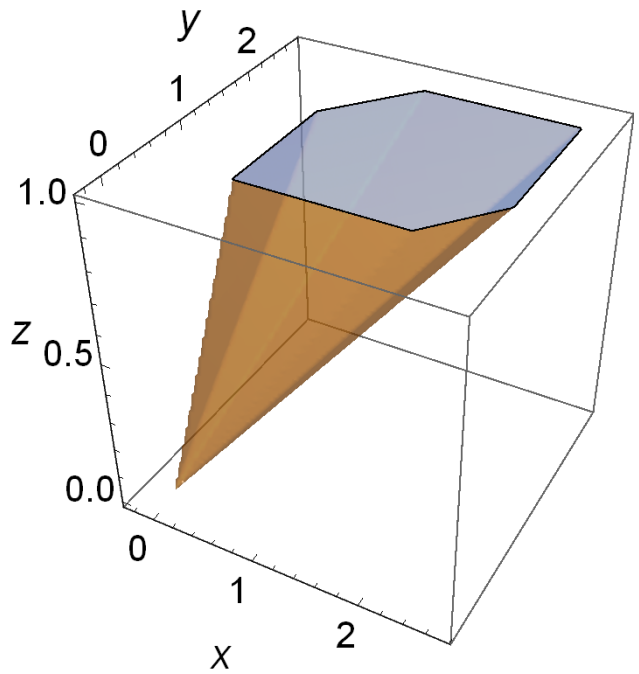
where C_i is a (closed) convex cone for each $i = 1, \ldots, m$

- We're going to learn lots o'cones

Recipe for a polyhedral cone:

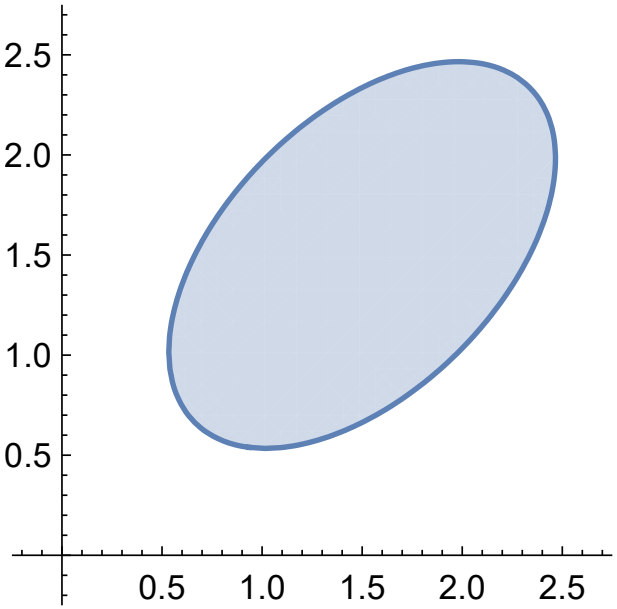
1. Begin with your favorite polyhedron $Ax \leq b$ where $x \in \mathbb{R}^n$
2. $\{Ax \leq bt, t \geq 0\}$ is a polyhedral cone in $(x,t) \in \mathbb{R}^{n+1}$
3. The slice $t = 1$ is the original polyhedron.

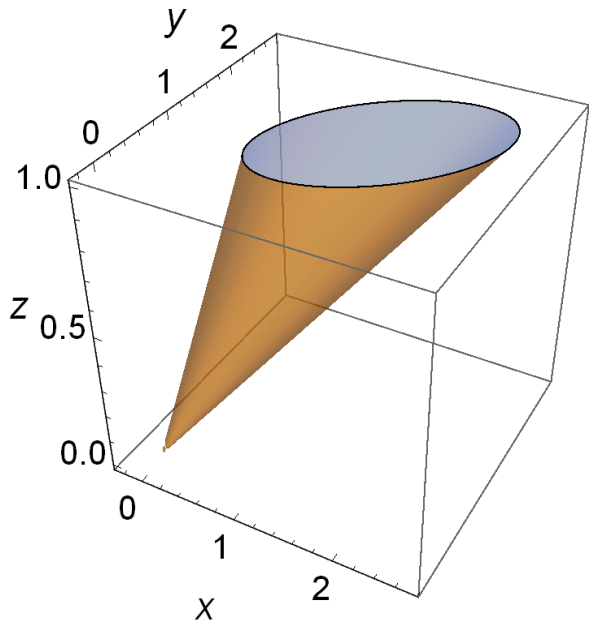




Recipe for an ellipsoidal cone:

1. Ellipsoid $x^\mathsf{T} P x + q^\mathsf{T} x + r \leq 0$ where $P \succ 0$ and $x \in \mathbb{R}^n$
2. Complete the square (see previous lecture) and write in the form $\|Ax + b\| \leq c$
3. $\{\|Ax + bt\| \leq ct\}$ is an ellipsoidal cone in $(x, t) \in \mathbb{R}^{n+1}$
4. The slice $t = 1$ is the original ellipsoid.





Second-order cone constraint

- A *second-order cone constraint* is the set of points $x \in \mathbb{R}^n$:

$$\|Ax + b\| \leq c^\mathsf{T} x + d$$

- If you square both sides...

$$\|Ax + b\| \leq c^\mathsf{T} x + d \quad \Longleftrightarrow \quad \begin{cases} \|Ax + b\|^2 \leq (c^\mathsf{T} x + d)^2 \\ c^\mathsf{T} x + d \geq 0 \end{cases}$$

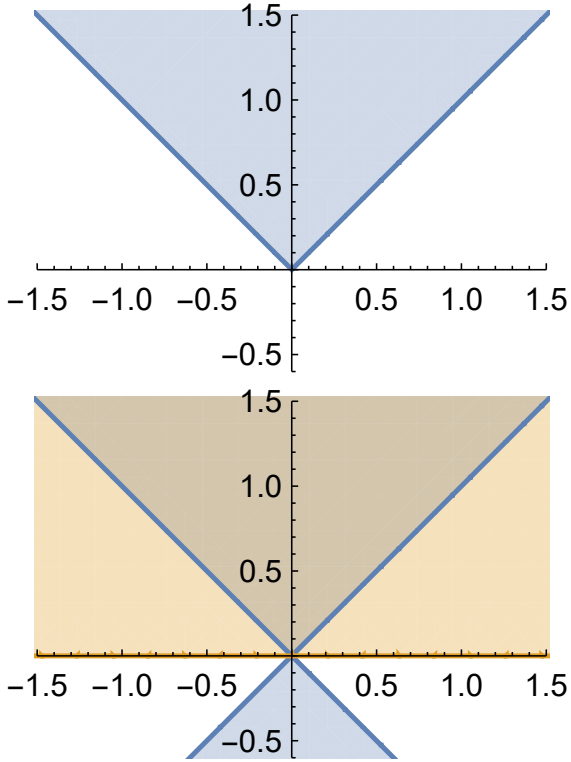
- The quadratic inequality is:

$$x^\mathsf{T}(A^\mathsf{T}A - cc^\mathsf{T})x + 2(b^\mathsf{T}A - dc^\mathsf{T})x + (b^\mathsf{T}b - d^2) \leq 0$$

- This may be **nonconvex**! You need to enforce the constraint $c^\mathsf{T} x + d \geq 0$ too!

Second-order cone constraint

$$\|Ax + b\| \leq c^{\mathsf{T}} x + d$$



Example:

If $A = \begin{bmatrix} 1 & 0 \end{bmatrix}$ and $c = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ and $b = d = 0$:

$$|x| \leq y$$

Squaring both sides leads to:

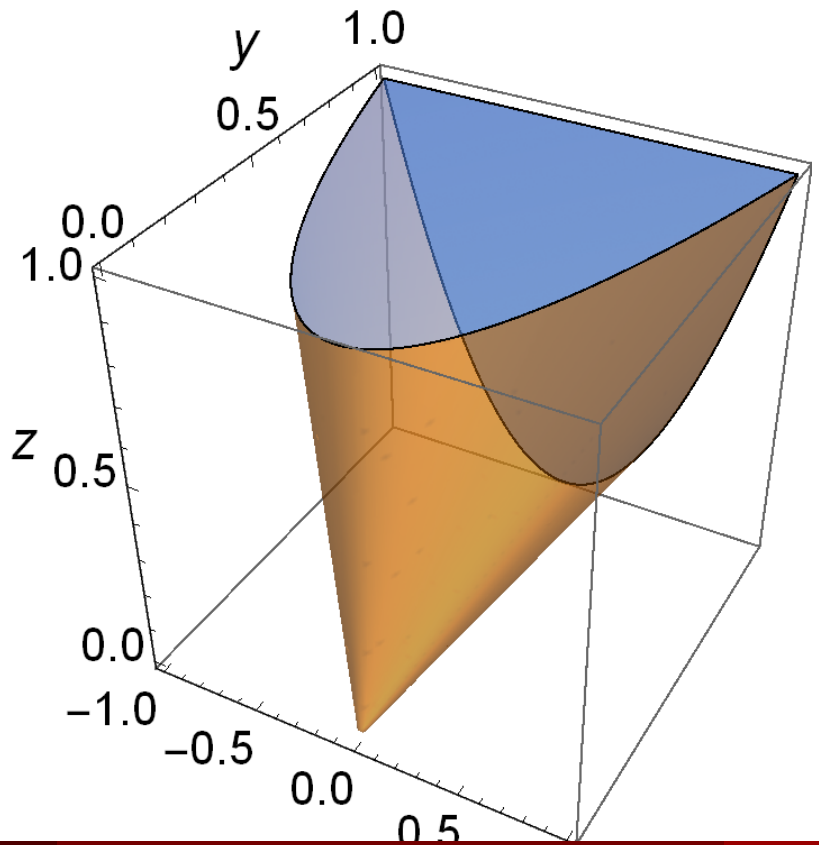
$$x^2 - y^2 \leq 0 \quad \text{and} \quad y \geq 0$$

Special case: Rotated Second-Order Cone

- A rotated second-order cone is the set $x \in \mathbb{R}^n$, $y, z \in \mathbb{R}$:

$$x^\mathsf{T} x \leq yz, \quad y \geq 0, \quad z \geq 0$$

- With $n = 1$, this looks like:



Rotated Second-Order Cone

- A rotated second-order cone is the set $x \in \mathbb{R}^n$, $y, z \in \mathbb{R}$:

$$x^\mathsf{T} x \leq yz, \quad y \geq 0, \quad z \geq 0$$

- Can put into standard (unrotated) SOC form:

$$\begin{aligned}
4x^\mathsf{T} x &\leq 4yz \\
4x^\mathsf{T} x + y^2 + z^2 &\leq 4yz + y^2 + z^2 \\
4x^\mathsf{T} x + (y-z)^2 &\leq (y+z)^2 \\
\sqrt{4x^\mathsf{T} x + (y-z)^2} &\leq y + z \\
\left\| \begin{bmatrix} 2x \\ y - z \end{bmatrix} \right\| &\leq y + z
\end{aligned}$$

SOCPs

A second-order cone program (SOCP) has the form:

$$\begin{array}{ll} \underset{x}{\text{minimize}} & c^{\mathsf{T}} x \\ \text{subject to:} & \|A_i x + b_i\| \leq c_i^{\mathsf{T}} x + d_i \quad \text{for } i = 1, \ldots, m \end{array}$$

- Every LP is an SOCP (just make each $A_i = 0$)
- Every convex QP and QCQP is an SOCP
  - convert quadratic cost to epigraph form (add a variable)
  - convert quadratic constraints to SOCP (complete square)

Julia and JuMP

- For scalar variable t and vector of variables x, JuMP has the notation:
  `@constraint(model, [t; x] in SecondOrderCone())`
- For scalar variables u and t and vector of variables$x$, JuMP has the notation:
- `@constraint(model, [t; u; x] in RotatedSecondOrderCone())`
  - N.B: This implements algebra $2ut \leq \|x\|_2^2$
- A good tutorial for cone programs in JuMP is: `https://jump.dev/JuMP.jl/stable/tutorials/conic/tips_and_tricks/`
- Check Out SOCP.ipynb

Rational Powers with SOC

- Modeling with cones gives you power to write nonlinear (and non-quadratic) things in a "convex" way
- Example: When I was at working in finance, our clients wanted to model "market impact" of making large trades:

$$\{(t, x) \in \mathbb{R}^2 : t \geq x^{3/2}\}$$

- This can be modeled with rotated second order cones:
- You can see this by

$$2ut \geq x^2, 2\frac{1}{8}x \geq u^2 \Rightarrow 4u^2t^2 \cdot \frac{1}{4}x \geq x^4u^2 \Rightarrow t \geq x^{3/2}$$

Example: Robust LP

Consider a linear program with each linear constraint separately written out:

$$\begin{array}{ll} \underset{x}{\text{maximize}} & c^T x \\ \text{subject to:} & a_i^T x \leq b_i \quad \text{for } i = 1, \ldots, m \end{array}$$

Suppose there is **uncertainty** in some of the a_i vectors. Say for example that $a_i = \bar{a}_i + \rho u$ where $\bar{a}_i$ is a nominal value and u is the uncertainty.

- box constraint: $\|u\|_\infty \leq 1$
- ball constraints: $\|u\|_2 \leq 1$

Robust LP with box constraint

Substituting $a_i = \bar{a}_i + \rho u$ into $a_i^\mathsf{T} x \leq b_i$, obtain:

$$\bar{a}_i^\mathsf{T} x + \rho u^\mathsf{T} x \leq b_i \quad \text{for all uncertain } u$$

box constraint:

If this must hold for **all** u with $\|u\|_\infty \leq 1$, then it holds for the worst-case u. Therefore:

$$u^\mathsf{T} x = \sum_{i=1}^n u_i x_i \leq \sum_{i=1}^n |u_i||x_i| \leq \sum_{i=1}^n |x_i| = \|x\|_1$$

Then we have

$$\bar{a}_i^\mathsf{T} x + \rho \|x\|_1 \leq b_i$$

Robust LP with box constraint

With a box constraint $a_i = \bar{a}_i + \rho u$ with $\|u\|_\infty \leq 1$

$$\begin{array}{ll} \underset{x}{\text{maximize}} & c^T x \\ \text{subject to:} & a_i^T x \leq b_i \quad \text{for } i = 1, \ldots, m \end{array}$$

Is equivalent to the optimization problem

$$\begin{array}{ll} \underset{x}{\text{maximize}} & c^T x \\ \text{subject to:} & \bar{a}_i^T x + \rho \|x\|_1 \leq b_i \quad \text{for } i = 1, \ldots, m \end{array}$$

Robust LP with box constraint

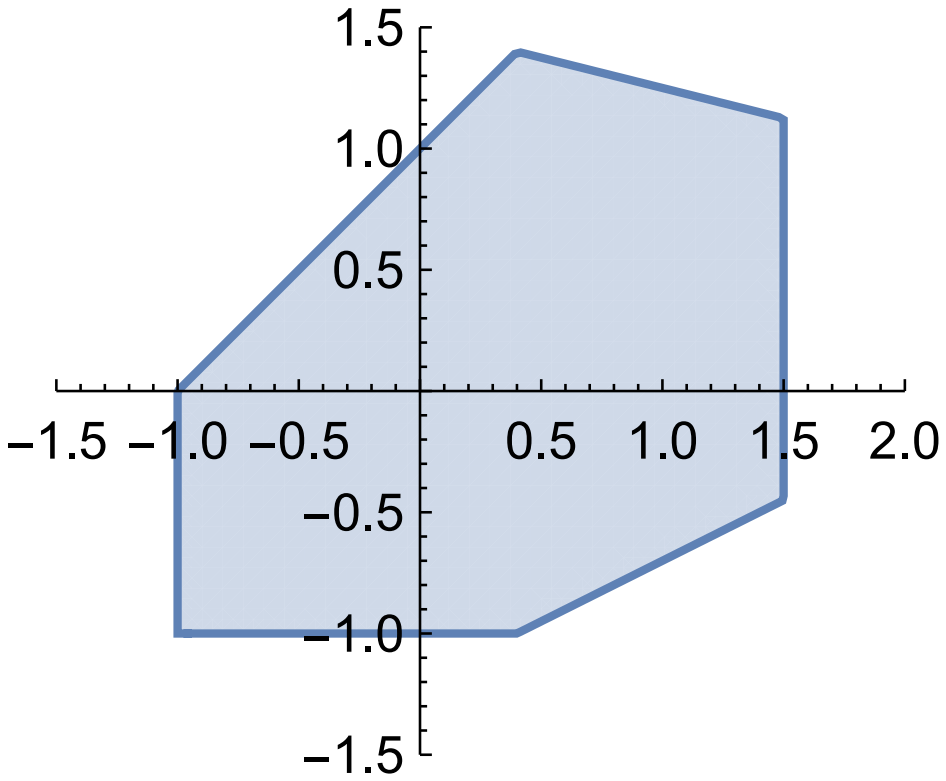
With a box constraint $a_i = \bar{a}_i + \rho u$ with $\|u\|_\infty \leq 1$

$$\begin{array}{ll} \underset{x}{\text{maximize}} & c^T x \\ \text{subject to:} & a_i^T x \leq b_i \quad \text{for } i = 1, \ldots, m \end{array}$$

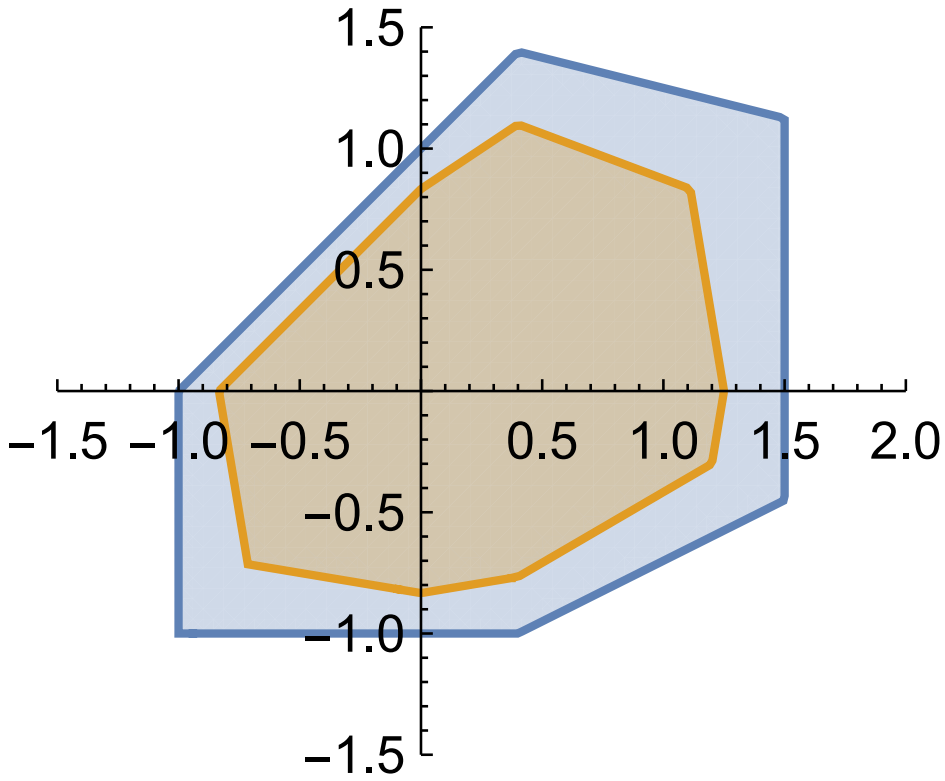
... which is equivalent to the linear program:

$$\begin{array}{ll} \underset{x,t}{\text{maximize}} & c^T x \\ \text{subject to:} & \bar{a}_i^T x + \rho \sum_{j=1}^{n} t_j \leq b_i \quad \text{for } i = 1, \ldots, m \\ & -t_j \leq x_j \leq t_j \quad \text{for } j = 1, \ldots, n \end{array}$$

Robust LP with box constraint



$$a_i^T x \leq b_i$$



$$a_i^T x + 0.2\|x\|_1 \leq b_i$$

- New region is smaller, still a polyhedron
- More robust to uncertain constraints

Robust LP with ball constraint

Substituting $a_i = \bar{a}_i + \rho u$ into $a_i^\mathsf{T} x \leq b_i$, obtain:

$$\bar{a}_i^\mathsf{T} x + \rho u^\mathsf{T} x \leq b_i \quad \text{for all uncertain } u$$

ball constraint:

If this must hold for **all** u with $\|u\|_2 \leq 1$, then it holds for the worst-case u. Using Cauchy-Schwarz inequality:

$$u^\mathsf{T} x \leq \|u\|_2 \|x\|_2 \leq \|x\|_2$$

Then we have

$$\bar{a}_i^\mathsf{T} x + \rho \|x\|_2 \leq b_i$$

(a second-order cone constraint!)

Robust LP with ball constraint

With a ball constraint $a_i = \bar{a}_i + \rho u$ with $\|u\|_2 \leq 1$

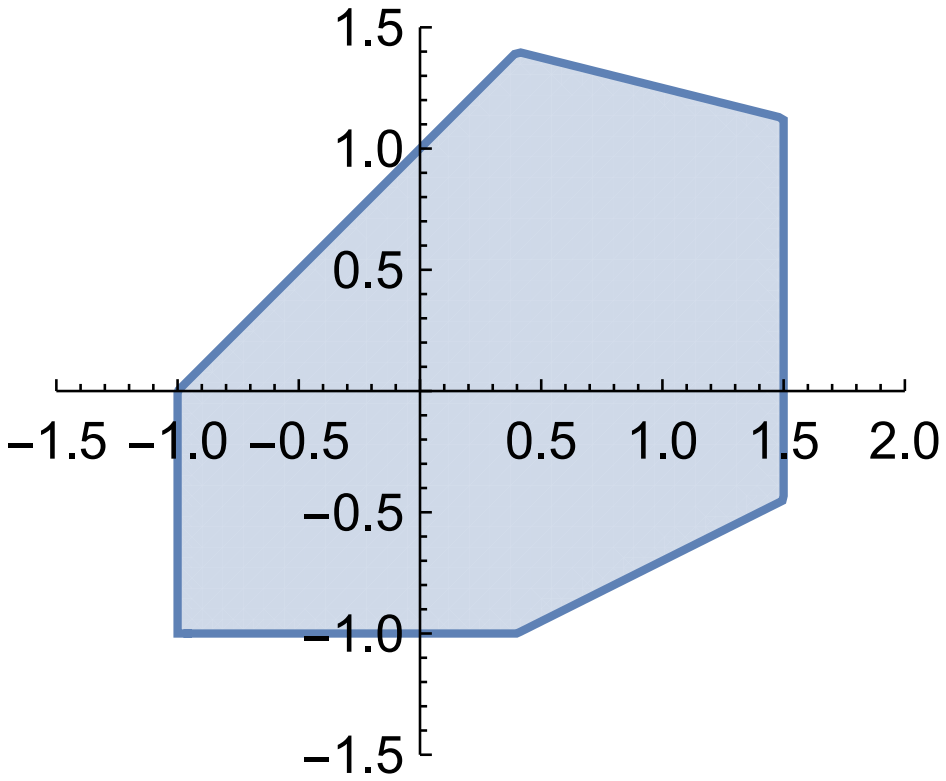
$$\begin{array}{ll} \underset{x}{\text{maximize}} & c^T x \\ \text{subject to:} & a_i^T x \leq b_i \quad \text{for } i = 1, \ldots, m \end{array}$$

Is equivalent to the optimization problem

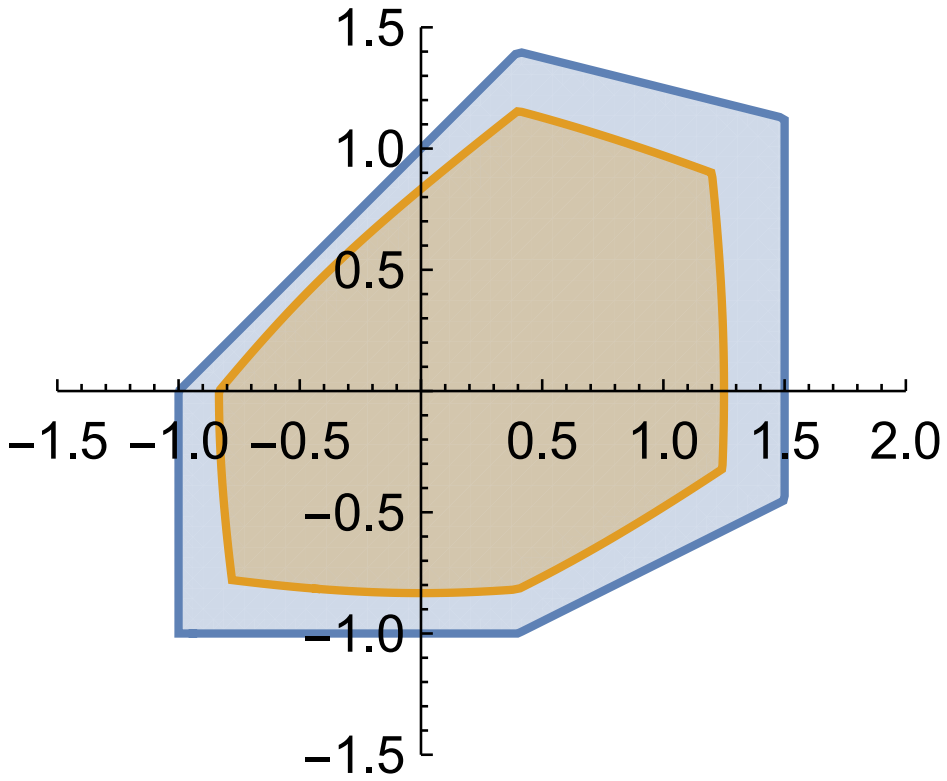
$$\begin{array}{ll} \underset{x}{\text{maximize}} & c^T x \\ \text{subject to:} & \bar{a}_i^T x + \rho \|x\|_2 \leq b_i \quad \text{for } i = 1, \ldots, m \end{array}$$

which is an SOCP

Robust LP with ball constraint



$$a_i^T x \leq b_i$$



$$a_i^T x + 0.2\|x\|_2 \leq b_i$$

- New region is smaller, no longer a polyhedron
- More robust to uncertain constraints

Matrix variables

Sometimes, the decision variable is a **matrix** X.

- Can always just think of $X \in \mathbb{R}^{m \times n}$ as $x \in \mathbb{R}^{mn}$.
- Linear functions:

$$\sum_{k=1}^{mn} c_k x_k = c^{\mathsf{T}} x$$

$$\sum_{i=1}^{m} \sum_{j=1}^{n} C_{ij} X_{ij} = \mathrm{trace}(C^{\mathsf{T}} X) = \langle C, X \rangle.$$

- Linear program:

$$\begin{array}{ll} \underset{X}{\text{maximize}} & \langle C, X \rangle \\ \text{subject to:} & \langle A_i, X \rangle \leq b_i \quad \text{for } i = 1, \ldots, k \end{array}$$

Matrix variables

If a decision variable is a symmetric matrix $X = X^\mathsf{T} \in \mathbb{R}^{n \times n}$, we can represent it as a vector $x \in \mathbb{R}^{n(n+1)/2}$.

$$\begin{bmatrix} x_1 & x_2 & x_3 \\ x_2 & x_4 & x_5 \\ x_3 & x_5 & x_6 \end{bmatrix} \quad \Longleftrightarrow \quad \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix}$$

The constraint $X \succeq 0$ is called a *semidefinite* constraint. What does it look like geometrically?

The PSD cone

The set of matrices $X \succeq 0$ are a **convex cone** in $\mathbb{R}^{n(n+1)/2}$

Example: The set $\begin{bmatrix} x & y \\ y & z \end{bmatrix} \succeq 0$ of points in $\mathbb{R}^3$ satisfy:

$$xz \geq y^2, \quad x \geq 0, \quad z \geq 0$$

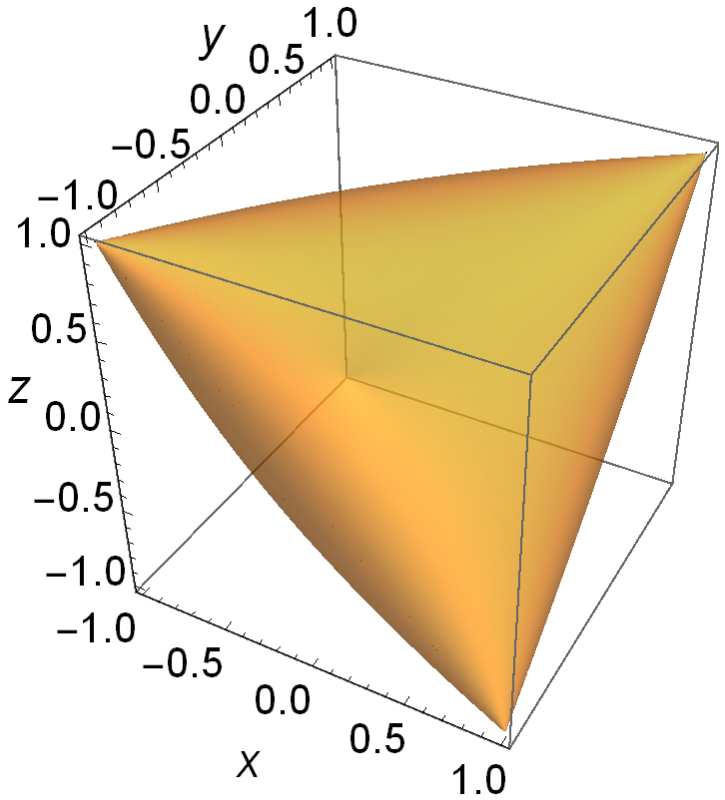
This is a rotated second-order cone! Equivalent to:

$$\left\| \begin{bmatrix} 2y \\ x - z \end{bmatrix} \right\| \leq x + z$$

More complicated example

The set of (x, y, z) satisfying $\begin{bmatrix} 1 & x & y \\ x & 1 & z \\ y & z & 1 \end{bmatrix} \succeq 0$ is the solution of:

$$\left\{ X \in \mathbb{R}^{3\times 3}, \quad X \succeq 0, \quad X_{11} = 1, \quad X_{22} = 1, \quad X_{33} = 1 \right\}$$



Spectrahedra

- Two common set representations:
  - variables $x_1, \ldots, x_k$, constants $Q_i = Q_i^T$, and constraint:

$$Q_0 + x_1 Q_1 + \ldots x_k Q_k \succeq 0 \qquad \text{(linear matrix inequality)}$$

  - variable $X \succeq 0$ and the constraints:

$$\langle A_i, X \rangle \leq b_i \qquad \text{(linear constraint form)}$$

- These sets are called *spectrahedra*.
- Very rich set, lots of possible shapes.

Semidefinite program (SDP)

Standard form #1: (looks like the standard form for an LP)

$$\begin{array}{ll} \underset{X}{\text{maximize}} & \langle C, X \rangle \\ \text{subject to:} & \langle A_i, X \rangle \leq b_i \quad \text{for } i = 1, \ldots, m \\ & X \succeq 0 \end{array}$$

Standard form #2:

$$\begin{array}{ll} \underset{x}{\text{maximize}} & c^{\mathsf{T}} x \\ \text{subject to:} & Q_0 + \sum_{i=1}^{m} x_i Q_i \succeq 0 \end{array}$$

Relationship with other programs

- Every LP is an SDP:

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

  is the same as:

$$x_1 \begin{bmatrix} a_{11} & 0 \\ 0 & a_{21} \end{bmatrix} + x_2 \begin{bmatrix} a_{12} & 0 \\ 0 & a_{22} \end{bmatrix} \preceq \begin{bmatrix} b_1 & 0 \\ 0 & b_2 \end{bmatrix}$$

- Every SOCP is an SDP too
- SDP.ipynb for some examples of SDPs

Exponential Cone

- The exponential cone is the set of points in $\mathbb{R}^3$ with

$$K_{exp} := \{(x, y, z) : y \exp(x/y) \leq z, y > 0\}$$

- It is useful for things modeling exponential and logarithms.
- Epigraph of exponential:

$$t \geq e^x \Leftrightarrow (x, 1, t) \in K_{exp}$$

- Hypograph of log:

$$t \leq \log(x) \Leftrightarrow (t, 1, x) \in K_{exp}$$

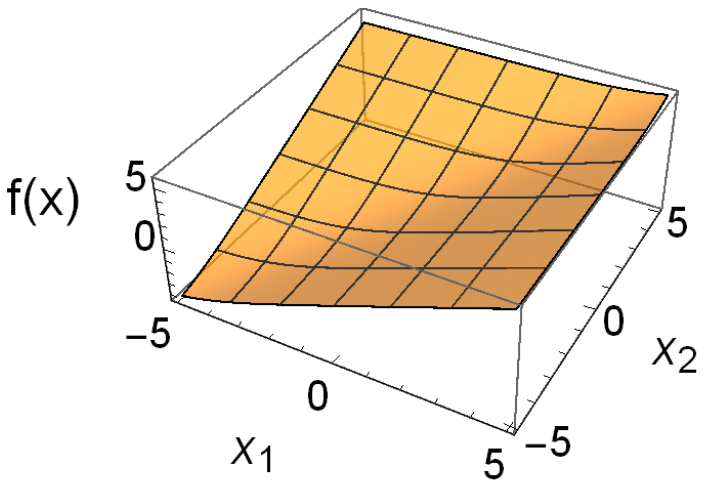
- Entropy:

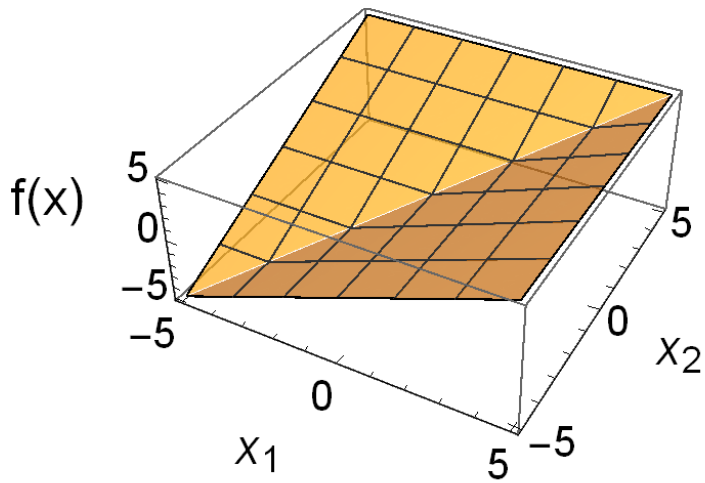
$$t \leq -x \log(x) \Leftrightarrow t \leq x \log(1/x) \Leftrightarrow (t, x, 1) \in K_{exp}$$

Log-Sum-Exp and Geometric Programming

- The *log-sum-exp* function (shown left) is convex:

$$f(x) := \log\left(\sum_{k=1}^{n} \exp x_k\right)$$





- It's a smoothed version of $\max\{x_1, \ldots, x_k\}$ (shown right).
- Sometimes called *softmax* — ubiquitous in machine learning

Example: Geometric Programming

- Suppose we have positive decision variables $x_i > 0$, and constraints of the form (with each $c_j > 0$ and $\alpha_{jk} \in \mathbb{R}$):

$$\sum_{j=1}^{n} c_j x_1^{\alpha_{j1}} x_2^{\alpha_{j2}} \cdots x_n^{\alpha_{jn}} \leq 1$$

- Then by using the substitution $y_i := \log(x_i)$, we have:

$$\log\left(\sum_{j=1}^{n} \exp\left(a_{j0} + a_{j1}y_1 + \cdots + a_{jn}y_n\right)\right) \leq 0$$

  (where $a_{j0} := \log c_j$).
- This is a log-sum-exp function composed with an affine function (convex!)
-

Modeling Log-Sum-Exp with Exponential Cone

- We wish to model the set

$$L = \{(t, x_1, \ldots, x_n) \in \mathbb{R}^{n+1} : t \geq \log(e^{x_1} + e_{x^2} + \ldots e^{x_n})\}$$

- Take $\exp()$ of both sides, divide both sides by e^t are rearrange to

$$e^{x_1 - t} + e^{x_2 - t} + \ldots e^{x_n - t} \leq 1.$$

- Introduce n new variables u_i (that we constrain to be $u_i \geq e^{x_i - t}$) and $\sum_i u_i \leq 1$
- So the set L can be modeled as

$$L \text{“} = \text{”} \{(t, x, u) \in \mathbb{R}^{2n+1} : \sum_{i=1}^{n} u_i \leq 1, (x_i - t, 1, u_i) \in K_{exp}\}$$

Example: Box-Making

- We want to design a box of height h, width w, and depth d with maximum volume (hwd) subject to the limits:
  - total wall area: $2(hw + hd) \leq A_{\text{wall}}$
  - total floor area: $wd \leq A_{\text{flr}}$
  - height-width aspect ratio: $\alpha \leq \frac{h}{w} \leq \beta$
  - width-depth aspect ratio: $\gamma \leq \frac{d}{w} \leq \delta$

We can make some of the constraints linear, but not all of them. This appears to be a nonconvex optimization problem...

Example: Box-Making

- total wall area: $2(hw + hd) \leq A_{\text{wall}}$
- total floor area: $wd \leq A_{\text{flr}}$
- height-width aspect ratio: $\alpha \leq \frac{h}{w} \leq \beta$
- width-depth aspect ratio: $\gamma \leq \frac{d}{w} \leq \delta$

$$\begin{array}{ll} \underset{h,w,d>0}{\text{minimize}} & h^{-1}w^{-1}d^{-1} \\ \text{subject to:} & \frac{2}{A_{\text{wall}}}hw + \frac{2}{A_{\text{wall}}}hd \leq 1, \quad \frac{1}{A_{\text{flr}}}wd \leq 1 \\ & \alpha h^{-1}w \leq 1, \qquad \frac{1}{\beta}hw^{-1} \leq 1 \\ & \gamma wd^{-1} \leq 1, \qquad \frac{1}{\delta}w^{-1}d \leq 1 \end{array}$$

Example: geometric programming

$$
\begin{array}{ll}
\underset{h,w,d>0}{\text{minimize}} & h^{-1}w^{-1}d^{-1} \\
\text{subject to:} & \frac{2}{A_{\text{wall}}}hw + \frac{2}{A_{\text{wall}}}hd \leq 1, \quad \frac{1}{A_{\text{flr}}}wd \leq 1 \\
& \alpha h^{-1}w \leq 1, \quad \frac{1}{\beta}hw^{-1} \leq 1 \\
& \gamma wd^{-1} \leq 1, \quad \frac{1}{\delta}w^{-1}d \leq 1
\end{array}
$$

- Define: $x := \log h$, $y := \log w$, and $z := \log d$.
- Express the problem in terms of the new variables x, y, z. Note: h, w, d are positive but x, y, z are unconstrained.

Example: geometric programming

$$
\begin{aligned}
\underset{x,y,z}{\text{minimize}} \quad & \log(e^{-x-y-z}) \\
\text{subject to:} \quad & \log(e^{\log(2/A_{\text{wall}})+x+y} + e^{\log(2/A_{\text{wall}})+x+z}) \leq 0 \\
& \log(e^{\log(1/A_{\text{flr}})+y+z}) \leq 0 \\
& \log(e^{\log\alpha - x + y}) \leq 0, \qquad \log(e^{-\log\beta + x - y}) \leq 0 \\
& \log(e^{\log\gamma + y - z}) \leq 0, \qquad \log(e^{-\log\delta - y + z}) \leq 0
\end{aligned}
$$

- this is a convex model, but it can be simplified!
- most of the constraints are actually linear. (Only the first one is not.)

Example: geometric programming

$$
\begin{aligned}
\underset{x,y,z}{\text{minimize}} \quad & -x-y-z \\
\text{subject to:} \quad & \log(e^{\log(2/A_{\text{wall}})+x+y} + e^{\log(2/A_{\text{wall}})+x+z}) \leq 0 \\
& y+z \leq \log A_{\text{flr}} \\
& \log\alpha \leq x-y \leq \log\beta \\
& \log\gamma \leq z-y \leq \log\delta
\end{aligned}
$$

- There is one log-sum-exp constraint
- Remaining constraints are all linear
- BoxDesign.ipynb