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

Jeff Linderoth

Department of Industrial and Systems Engineering
University of Wisconsin-Madison

March 6, 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:

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

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

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

$$Qu_i = \lambda_i u_i \qquad \text{for all } i = 1, \ldots, n$$
$$u_i^\top u_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

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

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

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

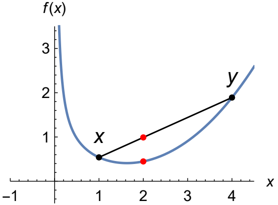
$$P = \sum_{i \in \mathcal{P}} \lambda_i v_i v_i^T, \quad Q = \sum_{i \in \mathcal{N}} (-\lambda_i) v_i v_i^T.$$

Convex functions

- A function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ 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:

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

This is an **unconstrained optimization problem**.

Positive Definite

- Note that the quadratic form is PSD:

$$\|Ax - b\|^2 = (Ax - b)^\top (Ax - b) = x^\top A^\top Ax - 2b^\top Ax + b^\top b$$

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

$$0 \leq z^T z = (A^\top y)^\top (Ay) = (y^\top A^\top)(Ay) = y^\top (A^\top A)y.$$

- 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:

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

- Is equivalent to solving the normal equations:

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

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

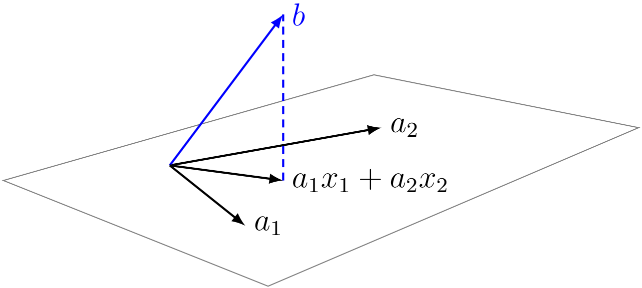
$$\hat{x} = (A^\mathsf{T} A)^{-1} A^\mathsf{T} b$$

- $A^\dagger := (A^\mathsf{T} A)^{-1} A^\mathsf{T}$ 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.

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



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

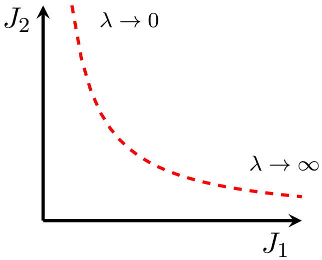
- Suppose $J_1 = f_1(x) = \|Ax - b\|^2$ and $J_2 = f_2(x) = \|Cx - d\|^2$.
- We would like to make **both** J_1 and J_2 small.
- A sensible approach: solve the optimization problem:

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

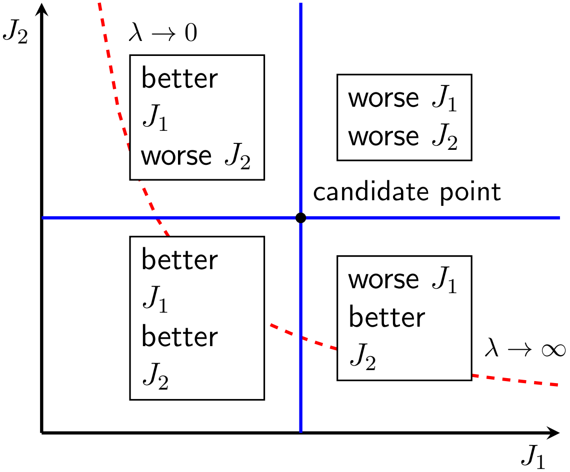
  where $\lambda > 0$ is a (fixed) *tradeoff parameter*.
- Then tune λ to explore possible results.
  - When $\lambda \to 0$, we place more weight on J_1
  - When $\lambda \to \infty$, we place more weight on J_2

Tradeoff analysis

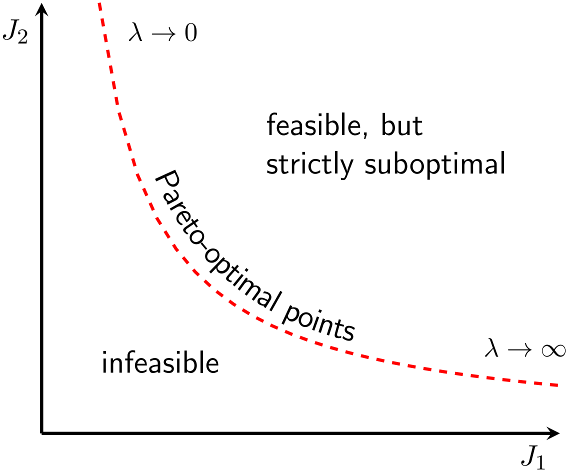
1. Choose values for λ (usually log-spaced). A useful command: `lambda = logspace(p,q,n)` produces n points logarithmically spaced between 10^p and 10^q.
2. For each λ value, find $\hat{x}_\lambda$ that minimizes $J_1 + \lambda J_2$.
3. For each $\hat{x}_\lambda$, also compute the corresponding J_1^λ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



Pareto curve

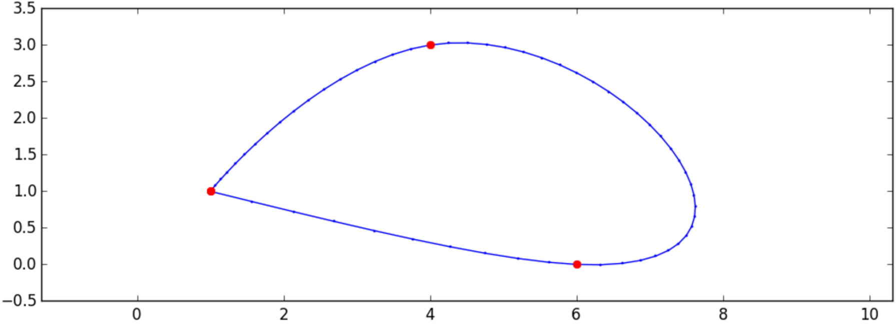


Pareto curve



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

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

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:

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

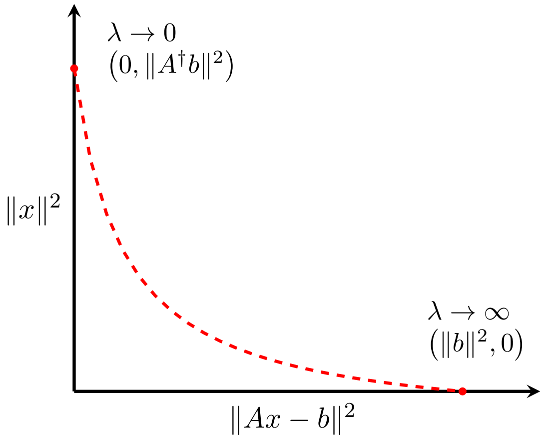
- Alternative approach: express it as a tradeoff!

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

- 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

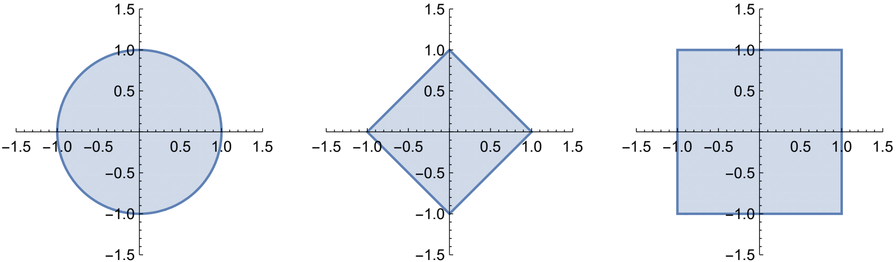
$$\underset{x}{\text{minimize}} \quad \|Ax - b\|^2 + \lambda R(x)$$

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:

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



$$\|x\|_2 \leq 1 \qquad \|x\|_1 \leq 1 \qquad \|x\|_\infty \leq 1$$

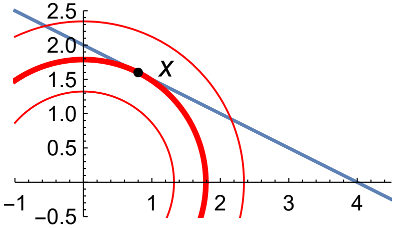
$$x^2 + y^2 \leq 1 \qquad |x| + |y| \leq 1 \qquad \max\{|x|, |y|\} \leq 1$$

Simple example

Consider the minimum-norm problem for different norms:

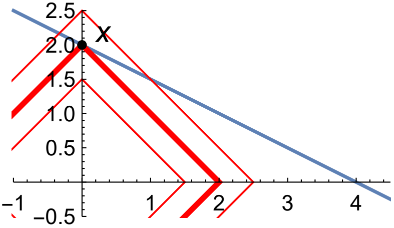
$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \|x\|_p \\ \text{subject to:} \quad & Ax = b \end{aligned}$$

- 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

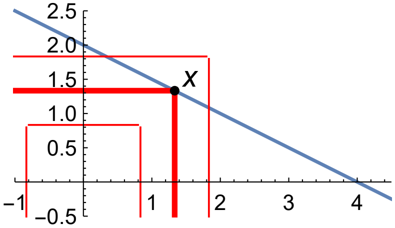


Simple example

- 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:

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

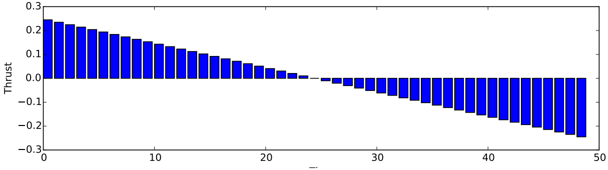
- Decide thruster inputs $u_1, u_2, \ldots, u_{49}$.
- This time: minimize $\|u\|_p$

Example: hovercraft revisited

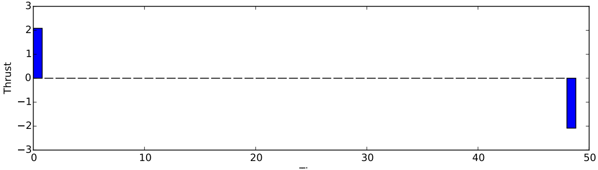
$$
\begin{aligned}
\underset{x_t, v_t, u_t}{\text{minimize}} \quad & \|u\|_p \\
\text{subject to:} \quad & x_{t+1} = x_t + v_t && \text{for } t = 1, \ldots, 49 \\
& v_{t+1} = v_t + u_t && \text{for } t = 1, \ldots, 49 \\
& x_1 = 0, \quad x_{50} = 100 \\
& v_1 = 0, \quad v_{50} = 0
\end{aligned}
$$

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

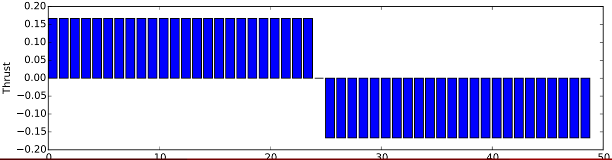
1. Minimizing $\|u\|_2^2$ (smooth)



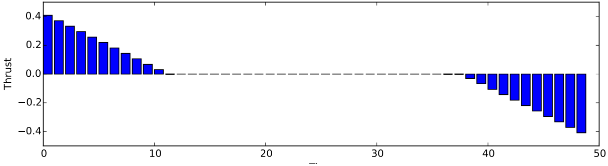
2. Minimizing $\|u\|_1$ (sparse)



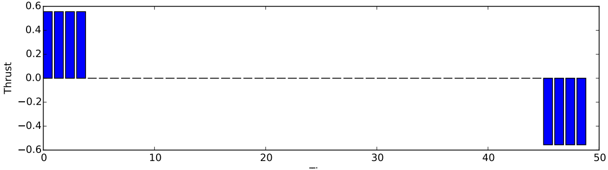
3. Minimizing $\|u\|_\infty$ (equalized)



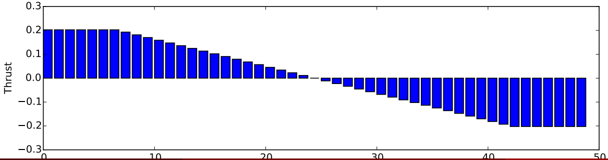
1. Minimizing $\|u\|_2^2 + \lambda\|u\|_1$ (smooth and sparse)



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 my 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

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