We have been looking at solving regularized problems:

\[
\min_x \|Ax - b\|_2^2 + \lambda r(x)
\]

where \(r(x)\) is chosen to be: (other choices exist also!)

\[
r(x) = \begin{cases} 
    \|x\|_1 : \text{lasso, promotes sparsity} \\
    \|x\|_2^2 : \text{tikhonov/ridge regression, small norm solution.} \\
    \|x\|_\infty : \text{equalized/quantized solution.}
\end{cases}
\]

Important to note:

\(\Rightarrow\) although we have a formula for the L2 case, i.e.
\[\hat{x} = A^+b \quad \text{when } \lambda \rightarrow 0 \quad \text{and } \quad \hat{x} = V_i \Sigma_i (\Sigma_i + \lambda I)^{-1} U_i^T b\]

in general, there is no formula for the L1 and L\(\infty\) cases!

Instead, we must use iterative methods to find the solution.

Note: iterative methods are often used to solve L2 problems even though we have a formula because iterative methods are fast! (we will discuss iterative methods soon).

\(\Rightarrow\) although L2 problem always has a unique solution, the same is not true for L1 and L\(\infty\) cases.

Example:
\[
\min_x \| [1, 1]^T x - 1 \|_2^2 + \lambda \|x\|_1.
\]
Example 1: predicting breast cancer from gene markers.

- we have m patients, m ≈ 200.
- each has been screened and tested. For each patient, we know the activity levels of \( n \approx 5000 \) genes, and we also know whether they have breast cancer or not.

\[
\begin{bmatrix}
1 \\
1 \\
1 \\
\vdots \\
1
\end{bmatrix}
= \begin{bmatrix}
0.3 & -0.21 & \cdots \\
0.6 & -0.1 & \cdots \\
\vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots \\
x_1 & x_2 & \cdots & x_n
\end{bmatrix}
\]

- disease state
- activity level for each of \( n \) genes
- in all \( m \) patients.

we'd like a design for \( \mathbf{x} \) that predicts disease state.
- i.e. we want to make a linear classifier.

**Problem:** in previous examples, e.g. iris classification, we had a large number of samples (flowers) and a small number of characteristics (sepal length, petal width,...) but this time, \( \mathbf{A} \mathbf{x} = \mathbf{b} \) has exact (and as many) solutions!

**Important note:**

Matlab's \( \mathbf{A} \backslash \mathbf{b} \) is not the same as \( \text{pinv}(\mathbf{A}) \times \mathbf{b} \) when \( \mathbf{A} \) is not full column rank!
Ex 1, cont'd

Regularization can help us choose a solution. In this case, we expect a relatively small number of genes to be involved, so it makes sense to look for a sparse $x$.

So we can use the lasso and adjust $\lambda$ and find solutions.

For each $\lambda$, solve min $||Ax-b||^2 + \lambda ||x||_1$.

(again, we'll see how to do this later!) and plot the trade-off curve:

![Graph showing $||Ax-b||^2$ vs. $||x||_1$](image)

(surrogate for sparsity).

If we plot actual sparsity:

![Graph showing number of non-zero entries in $x$ vs. $||Ax-b||^2$](image)

genre trend, but $\text{nnz}(x)$ is not a convex function, so it's not perfect.
Example 2: Hovercraft path planning (a control problem)

- Position at time $t$.
- Desired final position.

At $t=0$, craft is at $x=0$ (not moving).

At $t=40$, we want $x=10$ (again not moving).

We can apply thruster force at $t=0, 1, 2, \ldots$.

Define $\{x_0, x_1, \ldots, x_t, \ldots\} =$ position at time $t$.

$\{v_0, v_1, \ldots, v_t, \ldots\} =$ velocity at time $t$.

$\{u_0, u_1, \ldots, u_t, \ldots\} =$ thrust at time $t$.

Suppose equations of motion are

\[
\begin{align*}
\frac{d}{dt} x(t) &= v(t) \\
\frac{d}{dt} v(t) &= u(t)
\end{align*}
\]

"Double integrator".

Discretized equations:

\[
\begin{bmatrix}
v_{t+1} \\
x_{t+1}
\end{bmatrix} =
\begin{bmatrix}
1 & 0 \\
1 & 1
\end{bmatrix}
\begin{bmatrix}
v_t \\
x_t
\end{bmatrix} +
\begin{bmatrix}
0 \\
1
\end{bmatrix} u_t.
\]

\[
Z_{t+1} = A Z_t + B u_t.
\]

This is a linear dynamical system! (Ref. HW#1).
initial condition: \( Z_0 = [x_0, y_0] = [0, 0] \)

terminal condition: \( Z_{40} = [x_{40}, y_{40}] = [0, 0] \)

variables: \( u_0, u_1, \ldots, u_{39} \)

\[
U = \begin{bmatrix}
    u_0 \\
    u_1 \\
    \vdots \\
    u_{39}
\end{bmatrix}
\]

constraint...

\[
Z_1 = A Z_0 + B u_0 \\
Z_2 = A Z_1 + B u_1 = A^2 Z_0 + A B u_0 + B u_1 \\
\vdots \\
Z_{40} = A^{40} Z_0 + A^{39} B u_0 + A^{38} B u_1 + \ldots + A B u_{39} + B u_{39}
\]

\[
b = \text{zero.}
\]

\[
\begin{bmatrix}
    A^{39} B & \cdots & B
\end{bmatrix}
\begin{bmatrix}
    u_0 \\
    \vdots \\
    u_{39}
\end{bmatrix}
\]

problem: find \( U \) such that \( P U = b \).

\( \uparrow \quad \uparrow \quad \uparrow \)

2x40 \quad 40x1 \quad 2x1

highly under-determined!

how should one choose \( U \)?

- minimize \( \|U\|_2 \): higher thrust is increasingly more costly. Maybe appropriate if engine is less efficient at higher thrust. (more fuel required), Or cost is proportional to "input energy"

- minimize \( \|U\|_1 \): cost is proportional to thrust. More use also promotes sparsity.

- minimize \( \|U\|_{\infty} \): cost is proportional to maximum thrust required. (i.e. engine is free to operate, you pay more for bigger engine!)
problem: \[ \text{minimize } \| u \|_\alpha \]
\[ \text{subject to } Pu = b. \]

first approach: \[ \text{minimize } \| Pu - b \|_2^2 + \lambda \| u \|_\alpha \]
(regularize)
and take the limit of small \( \lambda > 0 \).

second approach: if \( Pu = b \), then \( u = \frac{P^* b + V_2 w}{\text{particular solution}} \) \( \frac{\text{general element}}{\text{solution of } N(P)} \).

then, \[ \text{minimize } \| P^* b + V_2 w \|_\alpha \]
and \( \hat{u} = P^* b + V_2 \hat{w} \).

\[ \text{show demo!} \]

- **L2**: (linear)
- **L0**: (constant)
- **L1**: (sparsel)