Midterm last night, but now redo it like HW.
Can collaborate! Go to HW Party. One Friday.
Read solutions after redos, then submit redo.
Can resubmit after seeing solutions to get back redo.
If redo complete (i.e., 80% after redos), double credit.

Our Goal: Your Mastery

Motivating Example:

\[ u(t) \xrightarrow{\text{system}} y(t) \]

Related to HW 6

Demo: System problem

Example 4.

Motivating Examples:

The model could be:

\[ y(t) = b \cdot u(t) \]

Called auto-regressive model of order \( b \).

Markov of order \( b \):

\[
\begin{align*}
\dot{y}(t) &= y(t-1) + \ldots + y(t-b) + \epsilon(t) \\
&= y(t) + \epsilon(t)
\end{align*}
\]

Want to do: System ID

(a) Try to fit all the models

Gold standard: works in practice

Silver standard: predicts well on test data

To do (b), need to solve a family of nested least-square problems

Approach: \( P \times S \xrightarrow{\text{set up, approximate}} \)

\[ P = \begin{bmatrix} b & a_1 & a_2 & \ldots \end{bmatrix} \]

\[ S = \begin{bmatrix} y(t) \\ y(t-1) \\ \vdots \end{bmatrix} \]
Other places this can happen:
1) Polynomial fitting
2) OMP

How do you solve least-squares?

Can we make things faster?

Thoughts:
1) $(Q^T Q)^{-1}$ is the painful step.
2) Can we change coordinates somehow?
When is \( D^T D \) the identity?

\[
\begin{align*}
\langle \vec{d}_i, \vec{d}_i \rangle &= 1 \quad \text{diagonal entries of} \\
\langle \vec{d}_i, \vec{d}_j \rangle &= 0 \quad \text{if } i \neq j \quad \text{aka} \quad \langle \vec{d}_j, \vec{d}_j \rangle = 0 \\
\| \vec{d}_i \| &= 1 \\
\| \vec{d}_j \| &= 1 \quad \text{normalized} = \frac{1}{\sqrt{\langle \vec{d}_i, \vec{d}_i \rangle + \langle \vec{d}_j, \vec{d}_j \rangle + \cdots + \langle \vec{d}_k, \vec{d}_k \rangle}}
\end{align*}
\]

Desired property: Orthonormal columns.

\[
Q = \begin{bmatrix} \vec{q}_1 & \vec{q}_2 & \cdots & \vec{q}_k \end{bmatrix}
\]

Know if \( Q \) is orthonormal matrix then \( Q^T Q = I \) \( k \times k \) 

"Orthonormal matrix"

How can we project onto the span of the columns of \( Q \)?

Want to find \( x \) s.t.

\[
\| \vec{y} - Q \hat{x} \|_2 \text{ is as close as possible to } \| \vec{y} \|
\]

\[
\hat{x} = (Q^T Q)^{-1} Q^T \vec{y}
\]

\[
\hat{y} = Q \hat{x} = Q Q^T \vec{y}
\]

\( Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \) \( Q^T \) \( \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} y_1' \\ y_2' \\ y_3' \end{bmatrix} \)
Given a basis expressed as an orthonormal $Q$ for a subspace, projection onto it is very fast.

**Goal (Subgoal of making things fast):**
Given a sequence of vectors $\vec{d}_1, \vec{d}_2, \ldots, \vec{d}_l$, return another sequence of vectors $\bar{\vec{d}}_1, \bar{\vec{d}}_2, \ldots, \bar{\vec{d}}_l$ such that $Q = [\bar{\vec{d}}_1, \ldots, \bar{\vec{d}}_l]$ is orthonormal, and $\forall k \leq l$, $\text{span}\{\vec{d}_1, \ldots, \vec{d}_k\} = \text{span}\{\bar{\vec{d}}_1, \ldots, \bar{\vec{d}}_k\}$.

**Orthonormalization** is the name given to this process.

Start with simplest case: $l=1$ - just one vector.
Need $\bar{\vec{d}}_1 = \alpha \vec{d}_1 \in \mathbb{R}$ so it spans the same subspace.

**Q: How do pick $\alpha$?**

**Guess:** Divide by norm?

$$\bar{\vec{d}}_1 = \frac{\vec{d}_1}{\|\vec{d}_1\|}$$

**Compute** $\bar{\vec{d}}_1^T \vec{d}_1 = \frac{\vec{d}_1^T \vec{d}_1}{\|\vec{d}_1\|^2} = \frac{\|\vec{d}_1\|^2}{\|\vec{d}_1\|^2} = 1$

**First nontrivial case:** $\{\vec{d}_1, \vec{d}_2\}$

Want $\bar{\vec{d}}_1$

Want $\bar{\vec{d}}_2$

Projection of $\vec{d}_2$ onto subspace spanned by $\vec{d}_1$. 

Orthonormalization of $\vec{d}_1, \vec{d}_2$: $Q = [\bar{\vec{d}}_1, \bar{\vec{d}}_2]$. 

new vector $\bar{\vec{d}}_2 = \vec{d}_2 - \text{proj}_{\bar{\vec{d}}_1}(\vec{d}_2)$.
\[ \mathbf{e}_r = \mathbf{d}_r - \frac{\mathbf{q}_r (\mathbf{q}_r^T \mathbf{d}_r)}{\| \mathbf{q}_r \|^2} \]

**Step 1** Project \( \mathbf{d}_r \) out of \( \mathbf{e}_r \)

**Step 2** Subtract \( \mathbf{q}_r \), projection of \( \mathbf{d}_r \), to get a residual \( \mathbf{e}_r \) that is 0

**Step 3** Normalize it.

Breaks if \( \| \mathbf{e}_r \| = 0 \) or \( \mathbf{e}_r = \emptyset \)

Happens only if \( \mathbf{d}_r \) is a multiple of \( \mathbf{q}_r \).

**Generalize** Given \( \{ \mathbf{d}_1, ..., \mathbf{d}_k, \mathbf{d}_{k+1} \} \)

How can we get \( \mathbf{q}_{k+1} \) given that we already have \( \mathbf{Q} = [\mathbf{q}_1, ..., \mathbf{q}_k] \) with \( \text{nullspace}(\mathbf{Q}_k) \)

Let \( \mathbf{D}_k = [\mathbf{d}_1, ..., \mathbf{d}_k] \)

Want \( \mathbf{e}_{k+1} = \mathbf{d}_{k+1} - \text{Projection of } \mathbf{d}_{k+1} \text{ onto subspace spanned by columns of } \mathbf{D}_k \)

\[ = \mathbf{d}_{k+1} - \mathbf{D}_k (\mathbf{D}_k^T \mathbf{D}_k)^{-1} \mathbf{D}_k^T \mathbf{d}_{k+1} \]

Since \( \text{Projection onto } \text{colspan } (\mathbf{D}_k) \) is the same as projection onto \( \text{nullspace } (\mathbf{Q}_k) \)

\[ = \mathbf{d}_{k+1} - \mathbf{Q}_k \mathbf{Q}_k^T \mathbf{d}_{k+1} \]

\[ = \mathbf{d}_{k+1} - \sum_{j=1}^{k} \mathbf{q}_j (\mathbf{q}_j^T \mathbf{d}_{k+1}) \]

Can normalize to get \( \mathbf{q}_{k+1} = \frac{\mathbf{e}_{k+1}}{\| \mathbf{e}_{k+1} \|} \)
Is $Q_{k+1} = \left[ Q_k ; \tilde{q}_{k+1} \right]$ orthonormal?

Need to check: $\| \tilde{q}_{k+1} \| = 1$? Yes because

Is $\tilde{q}_{k+1} \perp \tilde{q}_i$ for $i = 1, \ldots, k$?

Check:

$\tilde{q}_i^T \tilde{q}_{k+1} = \tilde{q}_i^T \frac{\tilde{z}_{k+1}}{\| \tilde{z}_{k+1} \|} = \frac{1}{\| \tilde{z}_{k+1} \|} \left( \tilde{q}_i^T \tilde{d}_{k+1} - \sum_{j=1}^{k} \tilde{q}_i^T \tilde{d}_j \tilde{q}_j^T \tilde{d}_{k+1} \right) = 0$

So $\tilde{q}_{k+1} \perp \tilde{q}_i$ for $i = 1, \ldots, k$.

Subgoal Achieved: Gram-Schmidt Orthonormalization

Is this actually any faster?

New Idea:

How many operations to set $\tilde{q}_i$?

For $k = 1, \ldots, d$

Recall $\tilde{d}_i$ are n-dim

So total cost of setting $\tilde{q}_1, \ldots, \tilde{q}_k$ is $O(d^2)$
Because

What about the old way? \[ (Q^T D)^T D^{-1} y \]

\[ F = 0 \text{ being } \mathbf{e} \]

\[ n > d. \text{ So both ways for the } d\text{-dim case} \]

\[ \text{cost the same.} \]

Savings exist relative to the \( \ell^4 \) term

is computing all the LS problem

Get significant speedup:

\[ (100)^4 = 10^8 \leq 100 \text{ million} \]

\[ (100)^3 = 10^6 \leq 1 \text{ million}. \]

**Final Approach:** To solve nested least-square problems:

1) Orthornomalize the columns \([d_1, \ldots, d_e]\)

   to get \([\tilde{q}_1, \ldots, \tilde{q}_e]\)

2) Compute \(Q^T \tilde{y}\) to get solutions to all nested problems at once.
So in the new coordinates: to see the solution for just $k$ we
look at $\hat{q}_1, \hat{q}_2, \ldots, \hat{q}_k, \theta$.

This gives you a projection 

$\sum_{i=1}^{k} \hat{q}_i \cdot (q_i, \theta)$

that best approximates $\bar{q}$ in the subspace spanned by $\hat{q}_1, \ldots, \hat{q}_k, \theta$. 