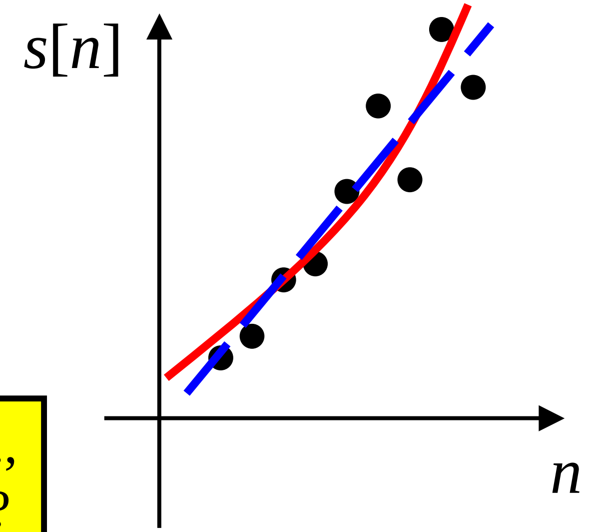
8.6 Order-Recursive LS



Motivate this idea with <u>*Curve Fitting*</u>

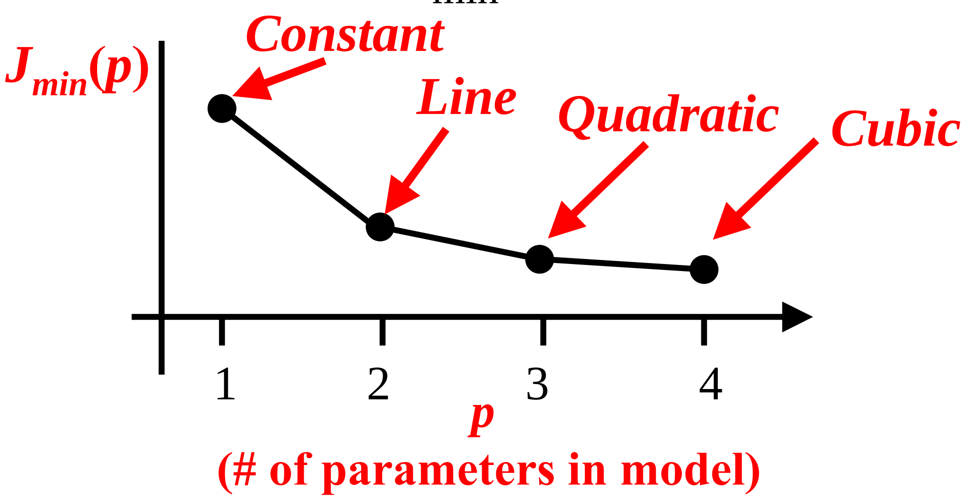
Given data: $n = 0, 1, 2, \ldots, N-1$

$s[0], s[1], \ldots, s[N-1]$

Want to fit a polynomial to data.., but which one is the right model?

- Constant
- Quadratic
- Linear
- Cubic, Etc.

Try each model, look at J_{min} … which one works "best"

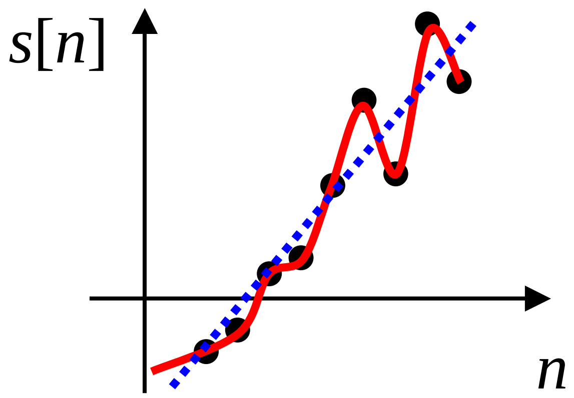


Choosing the Best Model Order

Q: Should you pick the order p that gives the smallest J_{min}??
A: NO!!!!

Fact: $J_{min}(p)$ is monotonically non-increasing as order p increases

If you have any N data points…
you can **perfectly** fit a $p = N$ model to them!!!!



2 points define a… line
3 points define a… quadratic
4 points define a… cubic
…
***N* points define… $a_N x^N + a_{N-1} x^{N-1} + \ldots + a_1 x + a_0$**

Warning: Don't "Fit the Noise"!!

Choosing the Order in Practice

Practice: use simplest model that adequately describes the data

Scheme: Only increase order if cost reduction is "significant"

- Increase to order p+1 only if $J_{min}(p) - J_{min}(p=1) > \varepsilon$ ← user-set threshold

- Also, in practice you may have some idea of the expected level of error

  $\Rightarrow$ thus have some idea of expected J_{min}

  $\Rightarrow$ use order p such that $J_{min}(p) \approx$ Expected J_{min}

Wasteful to independently compute the LS solution for each order

Drives Need for:

Efficient way to compute LS for many models

Q: If we have computed p-order model, can we use it to recursively compute (p+1)-order model?

A: YES!! $\Rightarrow$ **Order-Recursive LS**

Define General Order-Increasing Models

Define: $\mathbf{H}_{p+1} = [\ \mathbf{H}_p\ \mathbf{h}_{p+1}\ ]$ $\Rightarrow$ $\mathbf{h}_1,\ \mathbf{h}_2,\ \mathbf{h}_3, \ldots$

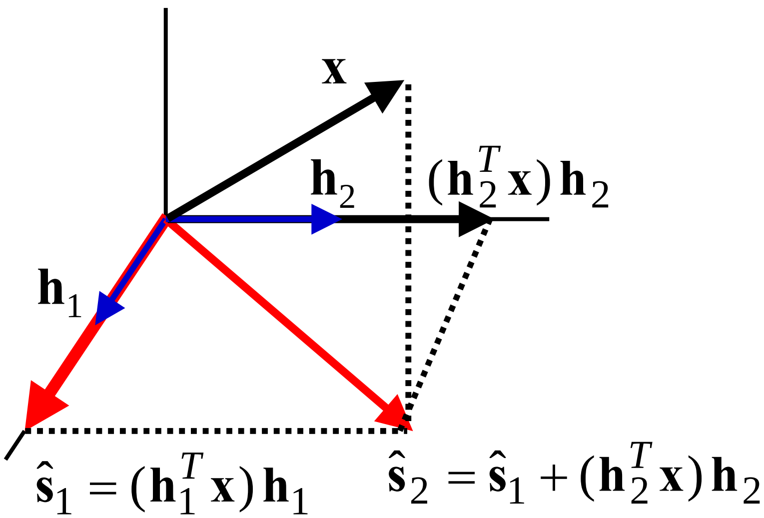
$\underbrace{\mathbf{h}_1}_{\mathbf{H}_1}$

$\underbrace{\mathbf{h}_1,\ \mathbf{h}_2}_{\mathbf{H}_2}$

$\underbrace{\mathbf{h}_1,\ \mathbf{h}_2,\ \mathbf{h}_3}_{\mathbf{H}_3}$ Etc.

Order-Recursive LS with Orthonormal Columns

If all $\mathbf{h}_i$ are $\perp$ $\Rightarrow$ EASY !!



$p = 1 \qquad \hat{\mathbf{s}}_1 = (\mathbf{h}_1^T \mathbf{x})\,\mathbf{h}_1$

$p = 2 \qquad \hat{\mathbf{s}}_2 = \hat{\mathbf{s}}_1 + (\mathbf{h}_2^T \mathbf{x})\,\mathbf{h}_2$

$p = 3 \qquad \hat{\mathbf{s}}_3 = \hat{\mathbf{s}}_2 + (\mathbf{h}_3^T \mathbf{x})\,\mathbf{h}_3$

$\vdots \qquad\qquad \vdots$

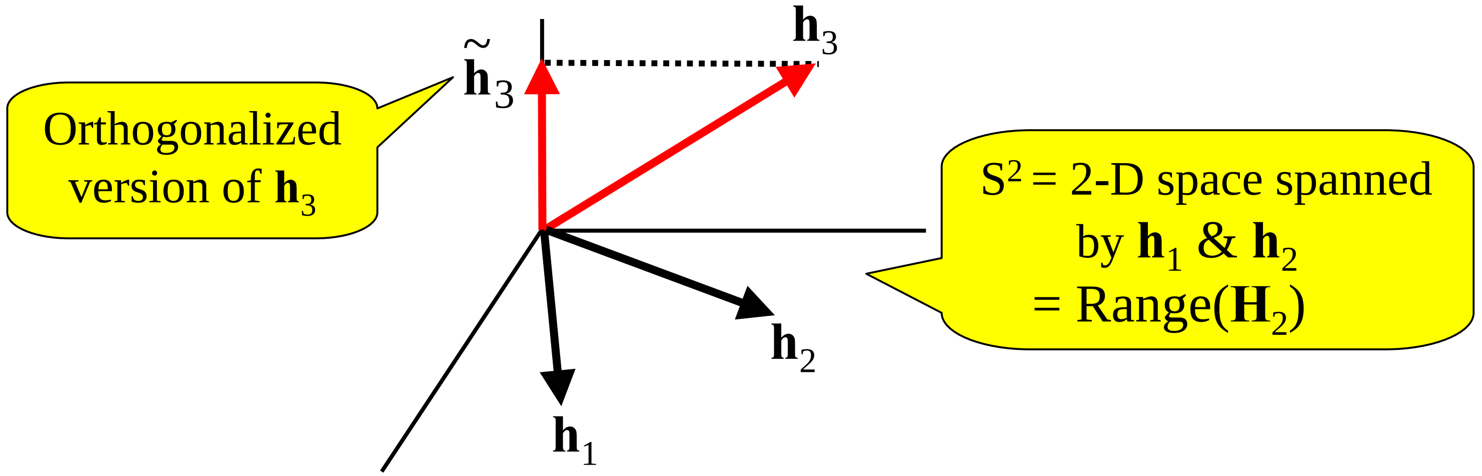
Order-Recursive Solution for General H

If $\mathbf{h}_i$ are *Not* ⊥ ⇒ Harder, but Possible!

Basic Idea: Given current-order estimate:

- map new column of **H** into an ON version
- use it to find new "estimate,"
- then transform to correct for orthogonalization

Quotes here because this estimate is for the orthogonalized model



Note: **x** is not shown here… it is in a higher dimensional space!!

Geometrical Development of Order-Recursive LS

The Geometry of Vector Space is indispensable for DSP!

See App. 8A for *Algebraic* Development

Yuk! Geometry is Easier!

Current-Order = k

$$\Rightarrow \quad \mathbf{H}_k = [\mathbf{h}_1\ \mathbf{h}_2\ \ldots\ \mathbf{h}_k] \qquad \text{(not necessarily } \perp)$$

Recall: $$\underbrace{\mathbf{P}_k = \mathbf{H}_k(\mathbf{H}_k^T\mathbf{H}_k)^{-1}\mathbf{H}_k^T}$$

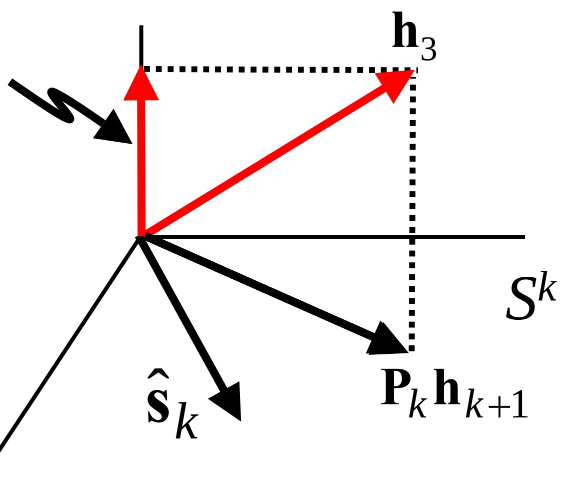
Projector onto $S^k = \text{Range}(\mathbf{H}_k)$

Given next column: $\mathbf{h}_{k+1}$ Find $\tilde{\mathbf{h}}_{k+1}$, which is $\perp$ to S^k

$$\tilde{\mathbf{h}}_{k+1} = \mathbf{h}_{k+1} - \mathbf{P}_k\mathbf{h}_{k+1} = \underbrace{(\mathbf{I} - \mathbf{P}_k)}_{\mathbf{P}_k^\perp}\mathbf{h}_{k+1}$$

$$\tilde{\mathbf{h}}_{k+1} = \mathbf{P}_k^\perp \mathbf{h}_{k+1}$$

$$\tilde{\mathbf{h}}_{k+1} \perp S^k \quad \Rightarrow \quad \tilde{\mathbf{h}}_{k+1} \perp \hat{\mathbf{s}}_k$$



So our approach is now: project $\mathbf{x}$ onto $\tilde{\mathbf{h}}_{k+1}$ and then add to $\hat{\mathbf{s}}_k$

The projection of $\mathbf{x}$ onto $\tilde{\mathbf{h}}_{k+1}$ is given by

Divide by norm to normalize

$$\Delta\hat{\mathbf{s}}_{k+1} = \left\langle \mathbf{x}, \frac{\tilde{\mathbf{h}}_{k+1}}{\left\|\tilde{\mathbf{h}}_{k+1}\right\|} \right\rangle \frac{\tilde{\mathbf{h}}_{k+1}}{\left\|\tilde{\mathbf{h}}_{k+1}\right\|} \qquad \textit{use } \tilde{\mathbf{h}}_{k+1} = \mathbf{P}_k^{\perp}\mathbf{h}_{k+1}$$

$$= \frac{\mathbf{x}^T \tilde{\mathbf{h}}_{k+1}}{\left\|\tilde{\mathbf{h}}_{k+1}\right\|^2} \tilde{\mathbf{h}}_{k+1} = \underbrace{\left[\frac{\mathbf{x}^T \mathbf{P}_k^{\perp}\mathbf{h}_{k+1}}{\left\|\mathbf{P}_k^{\perp}\mathbf{h}_{k+1}\right\|^2} \right]}_{\textit{scalar}!} \mathbf{P}_k^{\perp}\mathbf{h}_{k+1}$$

Now add this to current signal estimate: $\hat{\mathbf{s}}_{k+1} = \hat{\mathbf{s}}_k + \Delta\hat{\mathbf{s}}_{k+1}$

$$= \mathbf{H}_k \hat{\boldsymbol{\theta}}_k + \Delta\hat{\mathbf{s}}_{k+1}$$

Now we have:

$$\hat{\mathbf{s}}_{k+1} = \mathbf{H}_k\hat{\boldsymbol{\theta}}_k + \left[\frac{\mathbf{x}^T\mathbf{P}_k^{\perp}\mathbf{h}_{k+1}}{\left\|\mathbf{P}_k^{\perp}\mathbf{h}_{k+1}\right\|^2}\right]\mathbf{P}_k^{\perp}\mathbf{h}_{k+1}$$

Scalar… can move here and transpose

$$= \mathbf{H}_k\hat{\boldsymbol{\theta}}_k + \frac{(\mathbf{I}-\mathbf{P}_k)\ \mathbf{h}_{k+1}\mathbf{h}_{k+1}^T\ \mathbf{P}_k^{\perp}\ \mathbf{x}}{\underbrace{\mathbf{h}_{k+1}^T\ \mathbf{P}_k^{\perp}\ \mathbf{h}_{k+1}}}$$

Write out $\mathbf{P}_k^{\perp}$

Write out $\|.\|^2$ and use that $\mathbf{P}_k^{\perp}$ is idempotent

scalar… define as b for convenience

Finally: $\hat{\mathbf{s}}_k = \mathbf{H}_k\hat{\boldsymbol{\theta}}_k + \mathbf{h}_{k+1}b - \mathbf{H}_k(\mathbf{H}_k^T\mathbf{H}_k)^{-1}\mathbf{H}_k^T\mathbf{h}_{k+1}b$

$$= \underbrace{\left[\mathbf{H}_k \quad \mathbf{h}_{k+1}\right]}_{=\mathbf{H}_{k+1}} \underbrace{\begin{bmatrix} \hat{\boldsymbol{\theta}}_k - (\mathbf{H}_k^T\mathbf{H}_k)^{-1}\mathbf{H}_k^T\mathbf{h}_{k+1}b \\ \\ b \end{bmatrix}}_{\text{Clearly this is } \hat{\boldsymbol{\theta}}_{k+1}}$$

Order-Recursive LS Solution

$$\hat{\boldsymbol{\theta}}_{k+1} = \begin{bmatrix} \hat{\boldsymbol{\theta}}_k - (\mathbf{H}_k^T \mathbf{H}_k)^{-1} \mathbf{H}_k^T \mathbf{h}_{k+1} \left(\dfrac{\mathbf{h}_{k+1}^T \mathbf{P}_k^{\perp} \mathbf{x}}{\mathbf{h}_{k+1}^T \mathbf{P}_k^{\perp} \mathbf{h}_{k+1}} \right) \\ \dfrac{\mathbf{h}_{k+1}^T \mathbf{P}_k^{\perp} \mathbf{x}}{\mathbf{h}_{k+1}^T \mathbf{P}_k^{\perp} \mathbf{h}_{k+1}} \end{bmatrix}$$

Drawback: Needs Inversion Each Recursion

See Eq. (8.29) and (8.30) for a way to avoid inversion

Comments:

1. If $\mathbf{h}_{k+1} \perp \mathbf{H}_k$ $\Rightarrow$ simplifies problem as we've seen

   (This equation simplifies to our earlier result)

2. Note: $\mathbf{P}^{\perp}_k \, \mathbf{x}$ above is residual of k-order model

   = part of **x** not modeled by k-order model

   $\Rightarrow$ Update recursion works solely with this

   Makes Sense!!!

8.7 Sequential LS

In Last Section:
- Data Stays Fixed
- Model Order Increases

In This Section:
- Data Length Increases
- Model Order Stays Fixed

Say we have $\hat{\boldsymbol{\theta}}[N-1]$ based on $\{x[0], \ldots, x[N-1]\}$

If we get $x[N]$… can we compute $\hat{\boldsymbol{\theta}}[N]$ based on $\hat{\boldsymbol{\theta}}[N-1]$ and $x[N]$?
(w/o solving using full data set!)

We want… $\hat{\boldsymbol{\theta}}[N] = f(\hat{\boldsymbol{\theta}}[N-1], x[N])$

Approach Here:
1. Derive for DC-Level case
2. Interpret Results
3. Write Down General Result w/o Proof

Sequential LS for DC-Level Case

We know this: $$\hat{A}_{N-1} = \frac{1}{N}\sum_{n=0}^{N-1} x[n]$$

Re-Write

… and this: $$\hat{A}_N = \frac{1}{N+1}\sum_{n=0}^{N} x[n] = \frac{1}{N+1}\left[N\left(\frac{1}{N}\sum_{n=0}^{N-1} x[n]\right) + x[N]\right]$$

$$= \underbrace{\frac{N}{N+1}}_{=1-\frac{1}{N+1}} \hat{A}_{N-1} + \frac{1}{N+1} x[N]$$

$$\hat{A}_N = \underbrace{\hat{A}_{N-1}}_{\text{old estimate}} + \frac{1}{N+1}\underbrace{\Big(x[N] - \underbrace{\hat{A}_{N-1}}_{\text{prediction of the new data}}\Big)}_{\text{prediction error}}$$

Weighted Sequential LS for DC-Level Case

This is an even better illustration…

Assumed model: $x[n] = A + w[n] \quad \text{var}\{w[n]\} = \sigma_n^2$

$w[n]$ has unknown PDF but has known time-dependent variance

Standard WLS gives: $$\hat{A}_{N-1} = \frac{\displaystyle\sum_{n=0}^{N-1} \frac{x[n]}{\sigma_n^2}}{\displaystyle\sum_{n=0}^{N-1} \frac{1}{\sigma_n^2}}$$

With manipulations similar to the above case we get:

$$\hat{A}_N = \underbrace{\hat{A}_{N-1}}_{\text{old estimate}} + \underbrace{\left[\frac{\dfrac{1}{\sigma_N^2}}{\displaystyle\sum_{n=0}^{N} \frac{1}{\sigma_n^2}}\right]}_{\triangleq k_N} \underbrace{\left(x[N] - \hat{A}_{N-1}\right)}_{\text{prediction error}}$$

k_N is a "Gain" term that reflects "goodness" of new data

Exploring The Gain Term

We know that $\operatorname{var}(\hat{A}_{N-1}) = \dfrac{1}{\displaystyle\sum_{n=0}^{N-1}\left(\frac{1}{\sigma_n^2}\right)}$ … and using it in k_N…

…we get that

$$k_N = \frac{\operatorname{var}(\hat{A}_{N-1})}{\operatorname{var}(\hat{A}_{N-1}) + \underbrace{\sigma_N^2}_{\substack{\text{variance of}\\ \text{the new data}}}}$$

"poorness" of current estimate

"poorness" of new data

Note: $0 \le K[N] \le 1$

$\Rightarrow$ Gain depends on Relative Goodness Between:

- Current Estimate
- New Data Point

Extreme Cases for The Gain Term

$$\hat{A}[N] = \underbrace{\hat{A}[N-1]}_{\text{old estimate}} + K[N]\underbrace{(x[N] - \hat{A}[N-1])}_{\text{prediction error}}$$

If $\quad \text{var}(\hat{A}[N-1]) << \sigma_n^2$

$\Rightarrow K[N] \approx 0$

$\Rightarrow$ New Data Has Little Use

$\Rightarrow$ Make Little "Correction" Based on New Data

Good Estimate
Bad Data

If $\quad \text{var}(\hat{A}[N-1]) >> \sigma_n^2$

$\Rightarrow K[N] \approx 1$

$\Rightarrow$ New Data Very Useful

$\Rightarrow$ Make Large "Correction" Based on New Data

Bad Estimate
Good Data

General Sequential LS Result

See App. 8C for derivation

At time index *n*-1 we have:

$$\mathbf{x}_{n-1} = \left[x[0] \quad x[1] \quad \cdots \quad x[n-1] \right]^T$$

$$\mathbf{x}_{n-1} = \mathbf{H}_{n-1}\boldsymbol{\theta} + \mathbf{w}_{n-1} \qquad \mathbf{C}_{n-1} = \text{diag}\{\sigma_0^2, \sigma_1^2, \cdots, \sigma_{n-1}^2\}$$

Diagonal Covariance

(Sequential LS requires this)

$\hat{\boldsymbol{\theta}}_{n-1}$ LS Estimate using $\mathbf{x}_{n-1}$

$$\boldsymbol{\Sigma}_{n-1} \triangleq \text{cov}\left\{ \hat{\boldsymbol{\theta}}_{n-1} \right\} \quad \text{quality measure of estimate}$$

At time index *n* we get *x*[*n*]:

$$\mathbf{x}_n = \mathbf{H}_n\boldsymbol{\theta} + \mathbf{w}_n = \begin{bmatrix} \mathbf{H}_{n-1} \\ \\ \mathbf{h}_n^T \end{bmatrix} \boldsymbol{\theta} + \mathbf{w}_n$$

Tack on row at bottom to show how θ maps to *x*[*n*]

Iterate these Equations:

Given the Following: $\hat{\boldsymbol{\theta}}_{n-1} \quad \boldsymbol{\Sigma}_{n-1} \quad x[n] \quad \mathbf{h}_n \quad \sigma_n^2$

Update the Estimate:

$$\hat{\boldsymbol{\theta}}_n = \hat{\boldsymbol{\theta}}_{n-1} + \mathbf{k}_n \left(x[n] - \underbrace{\mathbf{h}_n^T \hat{\boldsymbol{\theta}}_{n-1}}\right)$$

Prediction of $x[n]$ using current parameter estimate

Compute the Gain:

$$\mathbf{k}_n = \frac{\boldsymbol{\Sigma}_{n-1}\mathbf{h}_n}{\sigma_n^2 + \mathbf{h}_n^T \boldsymbol{\Sigma}_{n-1}\mathbf{h}_n}$$

Gain has same kind of dependence on Relative Goodness between:

- Current Estimate
- New Data Point

Update the Est. Cov.:

$$\boldsymbol{\Sigma}_n = (\mathbf{I} - \mathbf{k}_n \mathbf{h}_n^T)\, \boldsymbol{\Sigma}_{n-1}$$

Initialization: (Assume p parameters)

- Collect first p data samples $x[0], \ldots, x[p-1]$
- Use "Batch" LS to compute: $\hat{\boldsymbol{\theta}}_{p-1} \quad \boldsymbol{\Sigma}_{p-1}$
- Then start sequential processing

Sequential LS Block Diagram