# Least Squares Estimation

*Namrata Vaswani, namrata@iastate.edu*

## Recall: Geometric Intuition for Least Squares

- Minimize $J(x) = ||y - Hx||^2$
- Solution satisfies: $H^T H\hat{x} = H^T y$, i.e. $\hat{x} = (H^T H)^{-1} H^T y$
- So $H^T (y - H\hat{x}) = 0$
- The least error $(y - H\hat{x})$ is $\perp$ to column space of $H$
- Think 3D: minimum error is always $\perp$ to plane of projection

# Weighted Least Squares

- $y = Hx + e$
- Minimize

$$J(x) = (y - Hx)^T W (y - Hx) \triangleq ||y - Hx||_W^2 \tag{1}$$

Solution:

$$\hat{x} = (H^T W H)^{-1} H^T W y \tag{2}$$

- Given that $E[e] = 0$ and $E[ee^T] = V$,
  Min. Variance Unbiased Linear Estimator of $x$: choose $W = V^{-1}$ in (2)
  Min. Variance of a vector: variance in any direction is minimized

## Proof (skip if you want to)

- Given $\hat{x} = Ly$, find $L$, s.t. $E[Ly] = E[LHx] = E[x]$, so $LH = I$
- Let $L_0 = (H^T V^{-1} H)^{-1} H^T V^{-1}$
- Error variance $E[(x - \hat{x})(x - \hat{x})^T]$

$$\begin{aligned} E[(x-\hat{x})(x-\hat{x})^T] &= E[(x - LHx - Le)(x - LHX - Le)^T] \\ &= E[Lee^T L^T] = LVL^T \end{aligned}$$

Say $L = L - L_0 + L_0$. Here $LH = I,\ L_0 H = I$, so $(L - L_0)H = 0$

$$\begin{aligned} LVL^T &= L_0 V L_0^T + (L - L_0)V(L - L_0)^T + 2L_0 V (L - L_0)^T \\ &= L_0 V L_0^T + (L - L_0)V(L - L_0)^T + (H^T V^{-1} H)^{-1} H^T (L - L_0)^T \\ &= L_0 V L_0^T + (L - L_0)V(L - L_0)^T \geq L_0 V L_0^T \end{aligned}$$

Thus $L_0$ is the optimal estimator (Note: $\geq$ for matrices)

# Regularized Least Squares

- Minimize

$$J(x) = (x - x_0)^T \Pi_0^{-1} (x - x_0) + (y - Hx)^T W (y - Hx) \tag{3}$$

$$\begin{aligned}
x' &\triangleq x - x_0, \;\; y' \triangleq y - Hx_0 \\
J(x) &= x'^T \Pi_0^{-1} x' + y'^T W y' \\
&= z \, M^{-1} \, z \\
z &\triangleq \begin{pmatrix} 0 \\ y' \end{pmatrix} - \begin{bmatrix} I \\ H \end{bmatrix} x' \\
M &\triangleq \begin{bmatrix} \Pi_0^{-1} & 0 \\ 0 & W \end{bmatrix}
\end{aligned}$$

- Solution: Use weighted least squares formula with $\tilde{y} = \begin{pmatrix} 0 \\ y' \end{pmatrix}$,

  $\tilde{H} = \begin{bmatrix} I \\ H \end{bmatrix}, \tilde{W} = M$

  Get:

$$\hat{x} = x_0 + (\Pi_0^{-1} + H^T W H)^{-1} H^T W (y - H x_0)$$

- Advantage: improves condition number of $H^T H$, incorporate prior knowledge about distance from $x_0$

# Recursive Least Squares

- Use in one of following situations:
  - When number of equations much larger than number of variables: Storage problem
  - Getting data sequentially, do not want to re-solve the full problem again
  - The dimension of $x$ is large, want to avoid inverting matrices
- **Goal:** At step $i-1$, have $\hat{x}_{i-1}$: minimizer of
  $(x-x_0)^T \Pi_0^{-1}(x-x_0) + ||H_{i-1}x - Y_{i-1}||^2_{W_{i-1}}$, $Y_{i-1} = [y_1, ...y_{i-1}]^T$

  Find $\hat{x}_i$: minimizer of $(x-x_0)^T \Pi_0^{-1}(x-x_0) + ||H_i x - Y_i||^2_{W_i}$,

$H_i = \begin{bmatrix} H_{i-1} \\ h_i \end{bmatrix}$ ($h_i$ is a row vector), $Y_i = [y_1, ...y_i]^T$ (column vector)

For simplicity of notation, assume $x_0 = 0$ and $W_i = I$.

$$\begin{aligned} H_i^T H_i &= H_{i-1}^T H_{i-1} + h_i^T h_i \\ \hat{x}_i &= (\Pi_0^{-1} + H_i^T H_i)^{-1} H_i^T Y_i \\ &= (\Pi_0^{-1} + H_{i-1}^T H_{i-1} + h_i^T h_i)^{-1} (H_{i-1}^T Y_{i-1} + h_i^T y_i) \end{aligned}$$

Define

$$\begin{aligned} P_i &= (\Pi_0^{-1} + H_i^T H_i)^{-1}, \;\; P_0 = \Pi_0 \\ \text{So } P_i &= [P_{i-1}^{-1} + h_i^T h_i]^{-1} \end{aligned}$$

Use Matrix Inversion identity:

$$(A + BCD)^{-1} = A^{-1} - A^{-1}B(C^{-1} + DA^{-1}B)^{-1}DA^{-1}$$

$$P_i = P_{i-1} - K_i h_i P_{i-1}$$

where

$$K_i = P_{i-1}h_i^T(1 + h_i P_{i-1} h_i^T)^{-1} \tag{4}$$

Thus

$$\begin{aligned}
\hat{x}_0 &= 0 \\
\hat{x}_i &= P_i H_i^T Y_i \\
&= [P_{i-1} - K_i h_i P_{i-1}][H_{i-1}^T Y_{i-1} + h_i^T y_i] \\
&= \hat{x}_{i-1} + K_i(y_i - h_i \hat{x}_{i-1})
\end{aligned}$$

The last equality uses the facts that (i) $\hat{x}_{i-1} = P_{i-1}H_{i-1}^T Y_{i-1}$, (ii) $[P_{i-1} - K_i h_i P_{i-1}]h_i^T y_i = K_i y_i$ (expand $K_i$, obtain this after a few manipulations).

Here we considered the weight $W_i = I$. If $W_i \neq I$, the equation for $K_i$ modifies to (replace $y_i$ by $w_i^{1/2} y_i$ & $h_i$ by $w_i^{1/2} h_i$, where $w_i = (W_i)_{i,i}$)

$$K_i = P_{i-1} h_i^T ({w_i}^{-1} + h_i P_{i-1} h_i^T)^{-1} \tag{5}$$

Also, here we considered $y_i$ to be a scalar and $h_i$ to be a row vector. In general: $y_i$ can be a $k$-dim vector, $h_i$ will be a matrix with $k$ rows, and the above formulae still apply, replace $1$ by $I$ everywhere

**RLS with Forgetting factor**

Weight older data with smaller weight $J(x) = \sum_{j=1}^{i} (y_j - h_j x)^2 \beta(i, j)$

Exponential forgetting: $\beta(i, j) = \lambda^{i-j}, \quad \lambda < 1$

Moving average: $\beta(i, j) = 0$ if $|i - j| > \Delta$ and $\beta(i, j) = 1$ otherwise

# Summarizing Recursive LS

- In general can assume that $y_i$ is $k$ dimensional and so $h_i$ has $k$ rows. Weight matrix $(W_i)_{i,i} = w_i$. Solution is:

$$\begin{aligned} \hat{x}_0 &= x_0, \;\; P_0 = \Pi_0 \\ K_i &= P_{i-1} h_i^T ({w_i}^{-1} + h_i P_{i-1} h_i^T)^{-1} \\ P_i &= (I - K_i h_i) P_{i-1} \\ \hat{x}_i &= \hat{x}_{i-1} + K_i (y_i - h_i x_i) \end{aligned} \tag{6}$$

- This is a recursive way to get the Regularized LS solution

$$\hat{x}_i = (\Pi_0^{-1} + H_i^T W_i H_i)^{-1} Y_i \tag{7}$$

with $H_i = [h_1^T, h_2^T, ...h_i^T]^T$, $Y_i = [y_1^T, y_2^T, ...y_i^T]^T$

## Connection with Kalman Filtering

The above is also the Kalman filter estimate of the state for the following system model:

$$\begin{aligned} x_i &= x_{i-1} \\ y_i &= h_i x_i + v_i, \;\; v_i \sim \mathcal{N}(0, R_i), \;\; w_i = R_i^{-1} \end{aligned} \tag{8}$$

# Kalman Filter Motivation

RLS was for static data: estimate the signal $x$ better and better as more and more data comes in, e.g. estimating the mean intensity of an object from a video sequence

RLS with forgetting factor assumes slowly time varying $x$

Kalman filter: if the signal is time varying, and we know (statistically) the dynamical model followed by the signal: e.g. tracking a moving object

$$\begin{aligned} x_0 &\sim \mathcal{N}(0, \Pi_0) \\ x_i &= F_i x_{i-1} + v_{x,i}, \;\; v_{x,i} \sim \mathcal{N}(0, Q_i) \end{aligned}$$

The observation model is as before:

$$y_i = h_i x_i + v_i, \;\; v_i \sim \mathcal{N}(0, R_i)$$

Goal: get the best (minimum mean square error) estimate of $x_i$ from $Y_i$

Cost: $J(\hat{x}_i) = E[(x_i - \hat{x}_i)^2 | Y_i]$

Minimizer: conditional mean $\hat{x}_i = E[x_i | Y_i]$

This is also the MAP estimate, i.e. $\hat{x}_i$ also maximizes $p(x_i | Y_i)$

# Example Applications

- Recursive LS:
  - Adaptive noise cancelation
  - Channel equalization using a training sequence
  - Object intensity estimation: $x$ = intensity, $y_i$ = vector of intensities of object region in frame $i$, $h_i = 1_m$ (column vector of $m$ ones)
  - Keep updating estimate of location of an object that is static, using a sequence of location observations coming in sequentially
- Recursive LS with forgetting factor: object not static but drifts very slowly (e.g. floating object) or object intensity changes very slowly
- Kalman filter: Track a moving object (estimate its location, velocity at each time), when acceleration is assumed i.i.d. Gaussian

## Material adapted from

- Chapters 2, 3 of Linear Estimation, by Kailath, Sayed, Hassibi