Least Squares and Kalman Filtering

Questions: Email me, namrata@ece.gatech.edu

Recall: Weighted Least Squares

- $y = Hx + e$
- Minimize

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

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

Recall: Proof

- 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

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

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

- Solution: Use 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

- When number of equations much larger than number of variables
  - Storage
  - Invert big matrices
  - Getting data sequentially
- Use a recursive algorithm

  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

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

Use Matrix Inversion identity:
$(A + BCD)^{-1} = A^{-1} + A^{-1}B(C^{-1} + DA^{-1}B)^{-1}DA^{-1}$

$$
P_i = P_{i-1} - \frac{P_{i-1} h_i^T h_i P_{i-1}}{1 + h_i P_{i-1} h_i^T}
$$

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

If $W_i \neq I$, this modifies to (replace y_i by $w_i^{1/2} y_i$ & h_i by $w_i^{1/2} h_i$):

$$
\hat{x}_i = \hat{x}_{i-1} + P_{i-1} h_i^T ({w_i}^{-1} + h_i P_{i-1} h_i^T)^{-1} (y_i - h_i \hat{x}_{i-1})
$$

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

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

Connection with Kalman Filtering

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

$$
\begin{aligned}
x_i &= x_{i-1} \\
y_i &= h_i x_i + v_i, \;\; v_i \sim \mathcal{N}(0, R_i), \; R_i = w_i^{-1}
\end{aligned}
\tag{4}
$$

Kalman Filter

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 $p(x_i | Y_i)$

Kalman filtering algorithm

At $i = 0$, $\hat{x}_0 = 0$, $P_0 = \Pi_0$.

For any i, assume that we know $\hat{x}_{i-1} = E[x_i|Y_{i-1}]$. Then

$$\begin{aligned} E[x_i|Y_{i-1}] &= F_i \hat{x}_{i-1} \stackrel{\triangle}{=} \hat{x}_{i|i-1} \\ Var(x_i|Y_{i-1}) &= F_i P_{i-1} F_i^T + Q_i \stackrel{\triangle}{=} P_{i|i-1} \end{aligned} \tag{5}$$

This is the **prediction step**

Filtering or correction step: Now $x_i|Y_{i-1}$ & $y_i|x_i, Y_{i-1}$ jointly Gaussian

$$\begin{aligned} x_i|Y_{i-1} &\sim \mathcal{N}(\hat{x}_{i|i-1}, P_{i|i-1}) \\ y_i|x_i, Y_{i-1} = y_i|x_i &\sim \mathcal{N}(h_i x_i, R_i) \end{aligned}$$

Using formula for the conditional distribution of $Z_1|Z_2$ when Z_1 and Z_2 are jointly Gaussian,

$$\begin{aligned} E[x_i|Y_i] &= \hat{x}_{i|i-1} + P_{i|i-1} h_i^T (R_i + h_i P_{i|i-1} h_i^T)^{-1} (y_i - h_i \hat{x}_{i|i-1}) \\ Var(x_i|Y_i) &= P_{i|i-1} - P_{i|i-1} h_i^T h_i P_{i|i-1} (R_i + h_i P_{i|i-1} h_i^T)^{-1} \end{aligned}$$

$\hat{x}_i = E[x_i|Y_i]$ and $P_i = Var(x_i|Y_i)$

Summarizing the algorithm

$$
\begin{aligned}
\hat{x}_{i|i-1} &= F_i \hat{x}_{i-1} \\
P_{i|i-1} &= F_i P_{i-1} F_i^T + Q_i \\
\hat{x}_i &= \hat{x}_{i|i-1} + P_{i|i-1} h_i^T (R_i + h_i P_{i|i-1} h_i^T)^{-1} (y_i - h_i \hat{x}_{i|i-1}) \\
P_i &= P_{i|i-1} - P_{i|i-1} h_i^T h_i P_{i|i-1} (R_i + h_i P_{i|i-1} h_i^T)^{-1}
\end{aligned}
$$

For $F_i = I$, $Q_i = 0$, get the RLS algorithm.

Example Applications

- RLS:
  - adaptive noise cancelation, given a noisy signal d_n assumed to be given by $d_n = \underline{u}_n^T \underline{w} + v_n$, get the best estimate of the weight w. Here $y_n = d_n$, $h_n = \underline{u}_n$, $x = \underline{w}$
  - 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),
- Kalman filter: Track a moving object (estimate its location and velocity at each time)

Suggested Reading

- Chapters 2, 3 & 9 of Linear Estimation, by Kailath, Sayed, Hassibi
- Chapters 4 & 5 of An Introduction to Signal Detection and Estimation, by Vincent Poor