Derivation of recursive least squares(continued)

Matrix inversion Lemma: If $\mathbf{A}$, $\mathbf{C}$, $\mathbf{BCD}$ are nonsingular square matrix (the inverse exists) then

$$[\mathbf{A}+\mathbf{BCD}]^{-1} = \mathbf{A}^{-1}-\mathbf{A}^{-1}\mathbf{B}[\mathbf{C}^{-1}+\mathbf{DA}^{-1}\mathbf{B}]^{-1}\mathbf{DA}^{-1}$$

The best way to prove this is to multiply both sides by $[\mathbf{A}+\mathbf{BCD}]$.

$$\begin{aligned}
& [\mathbf{A}+\mathbf{BCD}][\mathbf{A}^{-1}-\mathbf{A}^{-1}\mathbf{B}[\mathbf{C}^{-1}+\mathbf{DA}^{-1}\mathbf{B}]^{-1}\mathbf{DA}^{-1}] \\
= \; & \mathbf{I}+\mathbf{BCDA}^{-1}-\mathbf{B}[\mathbf{C}^{-1}+\mathbf{DA}^{-1}\mathbf{B}]^{-1}\mathbf{DA}^{-1} \\
& -\mathbf{BCDA}^{-1}\mathbf{B}[\mathbf{C}^{-1}+\mathbf{DA}^{-1}\mathbf{B}]^{-1}\mathbf{DA}^{-1} \\
= \; & \mathbf{I}+\mathbf{BCDA}^{-1}-\mathbf{B}\underbrace{\mathbf{CC}^{-1}}_{\mathbf{I}}[\mathbf{C}^{-1}+\mathbf{DA}^{-1}\mathbf{B}]^{-1}\mathbf{DA}^{-1} \\
& -\mathbf{BCDA}^{-1}\mathbf{B}[\mathbf{C}^{-1}+\mathbf{DA}^{-1}\mathbf{B}]^{-1}\mathbf{DA}^{-1} \\
= \; & \mathbf{I}+\mathbf{BCDA}^{-1} \\
& -\mathbf{BC}\underbrace{\{\mathbf{C}^{-1}+\mathbf{DA}^{-1}\mathbf{B}\}[\mathbf{C}^{-1}+\mathbf{DA}^{-1}\mathbf{B}]^{-1}}_{\mathbf{I}}\mathbf{DA}^{-1} \\
= \; & \mathbf{I}
\end{aligned}$$

Now look at the derivation of RLS algorithm so far and consider applying the matrix inversion lemma to (2) below

$$\begin{aligned}
\varepsilon(n) &= y(n) - \boldsymbol{\phi}^T(n)\hat{\boldsymbol{\theta}}(n-1) && (1)\\
\mathbf{P}(n) &= (\mathbf{P}^{-1}(n-1) + \boldsymbol{\phi}(n)\boldsymbol{\phi}^T(n))^{-1} && (2)\\
\mathbf{K}(n) &= \mathbf{P}(n)\boldsymbol{\phi}(n) && (3)\\
\hat{\boldsymbol{\theta}}(n) &= \hat{\boldsymbol{\theta}}(n-1) + \mathbf{K}(n)\varepsilon(n) && (4)
\end{aligned}$$

with

$$\mathbf{A} = \mathbf{P}^{-1}(n-1), \qquad \mathbf{B} = \boldsymbol{\phi}(n)$$

$$\mathbf{C} = 1, \qquad \mathbf{D} = \boldsymbol{\phi}^T(n)$$

$$\begin{aligned}
&\mathbf{P}(n)\\
=\ &(\mathbf{P}^{-1}(n-1) + \boldsymbol{\phi}(n)\boldsymbol{\phi}^T(n))^{-1}\\
=\ &\mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}[\mathbf{C}^{-1} + \mathbf{D}\mathbf{A}^{-1}\mathbf{B}]^{-1}\mathbf{D}\mathbf{A}^{-1}\\
=\ &\mathbf{P}(n-1) - \mathbf{P}(n-1)\boldsymbol{\phi}(n)\\
&\times[1 + \boldsymbol{\phi}^T(n)\mathbf{P}(n-1)\boldsymbol{\phi}(n)]^{-1}\boldsymbol{\phi}^T(n)\mathbf{P}(n-1)\\
=\ &\mathbf{P}(n-1) - \frac{\mathbf{P}(n-1)\boldsymbol{\phi}(n)\boldsymbol{\phi}^T(n)\mathbf{P}(n-1)}{1 + \boldsymbol{\phi}^T(n)\mathbf{P}(n-1)\boldsymbol{\phi}(n)}
\end{aligned}$$

The RLS algorithm is

$$
\begin{aligned}
\varepsilon(n) &= y(n) - \boldsymbol{\phi}^T(n)\hat{\boldsymbol{\theta}}(n-1) \\
\mathbf{P}(n) &= \mathbf{P}(n-1) - \frac{\mathbf{P}(n-1)\boldsymbol{\phi}(n)\boldsymbol{\phi}^T(n)\mathbf{P}(n-1)}{1+\boldsymbol{\phi}^T(n)\mathbf{P}(n-1)\boldsymbol{\phi}(n)} \\
\mathbf{K}(n) &= \mathbf{P}(n)\boldsymbol{\phi}(n) \\
\hat{\boldsymbol{\theta}}(n) &= \hat{\boldsymbol{\theta}}(n-1) + \mathbf{K}(n)\varepsilon(n)
\end{aligned}
$$

Here the term $\varepsilon(n)$ should be interpreted as a prediction error. It is the difference between the measured output $y(n)$ and the one-step-ahead prediction

$$\hat{y}(n|n-1, \hat{\boldsymbol{\theta}}(n-1)) = \boldsymbol{\phi}^T(n)\hat{\boldsymbol{\theta}}(n-1)$$

made at time $t = (n-1)$. If $\varepsilon(n)$ is small $\hat{\boldsymbol{\theta}}(n-1)$ is good and should not be modified very much.

$\mathbf{K}(n)$ should be interpreted as a weighting factor showing how much the value of $\varepsilon(n)$ will modify the different elements of the parameter vector.

The algorithm also needs initial values of $\hat{\boldsymbol{\theta}}(0)$ and $\mathbf{P}(0)$. It is convenient to set the initial values of $\hat{\boldsymbol{\theta}}(0)$ to zeros and the initial value of $\mathbf{P}(0)$ to $LN \times \mathbf{I}$, where $\mathbf{I}$ is the identity matrix and LN is a large number.

Example 1: (see Lecture 3 Example 2) Estimation of a constant. Assume that a model is

$$y(t) = b$$

This means a constant is to be determined from a number of noisy measurements $y(t)$, $t = 1, ..., n$.

Since $\boldsymbol{\phi}(n) = 1$

$$\mathbf{P}(n) = \mathbf{P}(n-1) - \frac{\mathbf{P}(n-1)\boldsymbol{\phi}(n)\boldsymbol{\phi}^T(n)\mathbf{P}(n-1)}{1 + \boldsymbol{\phi}^T(n)\mathbf{P}(n-1)\boldsymbol{\phi}(n)}$$

$$= \mathbf{P}(n-1) - \frac{\mathbf{P}^2(n-1)}{1 + \mathbf{P}(n-1)}$$

$$= \frac{\mathbf{P}(n-1)}{1+\mathbf{P}(n-1)}$$

i.e.

$$\mathbf{P}^{-1}(n) = \mathbf{P}^{-1}(n-1) + 1 = n$$

$$\mathbf{K}(n) = \mathbf{P}(n) = \frac{1}{n}$$

Thus $\hat{\boldsymbol{\theta}}(n) = \hat{\boldsymbol{\theta}}(n-1) + \frac{1}{n}\varepsilon(n)$ coincides with the results of Example 2 in Lecture 3.

Example 2: The system input/output of a process is measured as in the following Table. Suppose the process can be described by a model $y(t) = ay(t-1) + bu(t-1) = \boldsymbol{\phi}^T(t)\boldsymbol{\theta}$. If a recursive least squares algorithm is applied for on-line parameter estimation and at time

$t = 2, \hat{\boldsymbol{\theta}}(2) = [0.8, 0.1]^T, \mathbf{P}(2) = \begin{bmatrix} 1000 & 0 \\ 0 & 1000 \end{bmatrix}$,

find $\hat{\boldsymbol{\theta}}(3)$, $\mathbf{P}(3)$.

t	1	2	3	4
$y(t)$	0.5	0.6	0.4	0.5
$u(t)$	0.3	0.4	0.5	0.7

Note $\boldsymbol{\phi}(t) = [y(t-1), u(t-1)]^T$

$$\begin{aligned}\varepsilon(3) &= y(3) - \boldsymbol{\phi}^T(3)\hat{\boldsymbol{\theta}}(2) \\ &= 0.4 - [0.6, 0.4]\begin{bmatrix} 0.8 \\ 0.1 \end{bmatrix} = -0.12\end{aligned}$$

$$\mathbf{P}(3) = \mathbf{P}(2) - \frac{\mathbf{P}(2)\boldsymbol{\phi}(3)\boldsymbol{\phi}^T(3)\mathbf{P}(2)}{1+\boldsymbol{\phi}^T(3)\mathbf{P}(2)\boldsymbol{\phi}(3)}$$

$$= \begin{bmatrix} 1000 & 0 \\ 0 & 1000 \end{bmatrix} - \frac{\begin{bmatrix} 1000 & 0 \\ 0 & 1000 \end{bmatrix}\begin{bmatrix} 0.6 \\ 0.4 \end{bmatrix}[0.6,0.4]\begin{bmatrix} 1000 & 0 \\ 0 & 1000 \end{bmatrix}}{1+[0.6,0.4]\begin{bmatrix} 1000 & 0 \\ 0 & 1000 \end{bmatrix}\begin{bmatrix} 0.6 \\ 0.4 \end{bmatrix}}$$

$$= \begin{bmatrix} 1000 & 0 \\ 0 & 1000 \end{bmatrix} - \frac{\begin{bmatrix} 600 \\ 400 \end{bmatrix} [600,\ 400]}{1+[600,400]\begin{bmatrix} 0.6 \\ 0.4 \end{bmatrix}}$$

$$= \begin{bmatrix} 1000 & 0 \\ 0 & 1000 \end{bmatrix} - \frac{\begin{bmatrix} 360000 & 240000 \\ 240000 & 360000 \end{bmatrix}}{521}$$

$$= \begin{bmatrix} 309.0211 & -460.6526 \\ -460.6526 & 309.0211 \end{bmatrix}$$

$$\begin{aligned} \mathbf{K}(3) &= \mathbf{P}(3)\boldsymbol{\phi}(3) \\ &= \begin{bmatrix} 309.0211 & -460.6526 \\ -460.6526 & 309.0211 \end{bmatrix} \begin{bmatrix} 0.6 \\ 0.4 \end{bmatrix} \\ &= \begin{bmatrix} 1.1516 \\ -152.7831 \end{bmatrix} \\ \hat{\boldsymbol{\theta}}(3) &= \hat{\boldsymbol{\theta}}(2) + \mathbf{K}(3)\varepsilon(3) \\ &= \begin{bmatrix} 0.8 \\ 0.1 \end{bmatrix} + \begin{bmatrix} 1.1516 \\ -152.7831 \end{bmatrix} (-0.12) \\ &= \begin{bmatrix} 0.6618 \\ 18.4340 \end{bmatrix} \end{aligned}$$

RLS algorithm with a forgetting factor

The RLS algorithm can be modified for tracking time varying parameters. One approach is the RLS algorithm with a forgetting factor. For a linear regression model

$$y(t) = \boldsymbol{\phi}^T(t)\boldsymbol{\theta} + e(t)$$

The loss function to be minimized in a least squares algorithm is

$$V(\boldsymbol{\theta}) = \sum_{t=1}^{n} [y(t) - \boldsymbol{\phi}^T(t)\boldsymbol{\theta}]^2$$

then we minimize this with respect to $\boldsymbol{\theta}$. Consider modifying the loss function to

$$V(\boldsymbol{\theta}) = \sum_{t=1}^{n} \lambda^{(n-t)} [y(t) - \boldsymbol{\phi}^T(t)\boldsymbol{\theta}]^2$$

where $\lambda < 1$ is the forgetting factor. e.g. $\lambda = 0.99$ or 0.95. This means that as n increases, the measurements obtained previously are discounted. Older data has less effect on the coefficient estimation, hence "forgotten".

The RLS algorithm with a forgetting factor is

$$\begin{aligned}
\varepsilon(n) &= y(n) - \boldsymbol{\phi}^T(n)\widehat{\boldsymbol{\theta}}(n-1) \\
\mathbf{P}(n) &= \frac{1}{\lambda}\{\mathbf{P}(n-1) - \frac{\mathbf{P}(n-1)\boldsymbol{\phi}(n)\boldsymbol{\phi}^T(n)\mathbf{P}(n-1)}{\lambda + \boldsymbol{\phi}^T(n)\mathbf{P}(n-1)\boldsymbol{\phi}(n)}\} \\
\mathbf{K}(n) &= \mathbf{P}(n)\boldsymbol{\phi}(n) \\
\widehat{\boldsymbol{\theta}}(n) &= \widehat{\boldsymbol{\theta}}(n-1) + \mathbf{K}(n)\varepsilon(n)
\end{aligned}$$

When $\lambda = 1$ this is simply the RLS algorithm.

The smaller the value of λ, the quicker the information in previous data will be forgotten.

Using the RLS algorithm with a forgetting factor, we may speak about "real time identification"". When the properties of the process may change (slowly) with time, the algorithm is able to track the time-varying parameters describing such process.

Summary of the RLS algorithm:

1. Initialization: Set n_a, n_b, λ, $\hat{\boldsymbol{\theta}}(0)$ and $\mathbf{P}(0)$. Step 2-5 are repeated starting from $t = 1$.

2. At time step $t = n$, measure current output $y(n)$.

3. Recall past y's and u's and form $\boldsymbol{\phi}(n)$.

4. Apply RLS algorithm for $\hat{\boldsymbol{\theta}}(n)$ and $\mathbf{P}(n)$

5. $\hat{\boldsymbol{\theta}}(n) \rightarrow \hat{\boldsymbol{\theta}}(n-1)$ and $\mathbf{P}(n) \rightarrow \mathbf{P}(n-1)$

6. $t = n + 1$, go to step 2.