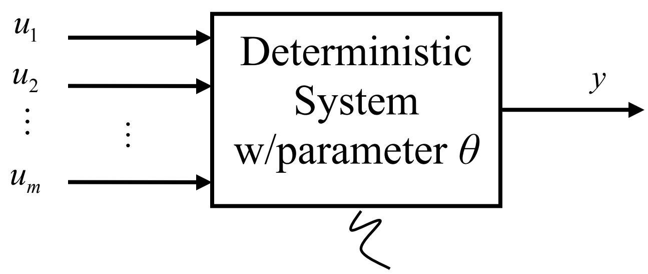
2.160 Identification, Estimation, and Learning

Lecture Notes No. 2

February 13, 2006

2. Parameter Estimation for Deterministic Systems

2.1 Least Squares Estimation



Linearly parameterized model

Input-output $\quad y = b_1 u_1 + b_2 u_2 + \ldots + b_m u_m$

Parameters to estimate: $\quad \theta = [b_1 \quad \ldots \quad b_m]^T \in R^m$

Observations:
$$\begin{cases} \varphi = [u_1 \quad \ldots \quad u_m]^T \in R^m \\ y = \varphi^T \theta \end{cases} \tag{1}$$

The problem is to find the parameters $\theta = [b_1 \quad \ldots \quad b_m]^T$ from observation data:

$$\left.\begin{matrix} \varphi(1), y(1) \\ \varphi(2), y(2) \\ \vdots \\ \varphi(N), y(N) \end{matrix}\right\} \rightarrow \theta$$

The system may be

a linear dynamic system, e.g. $\quad y(t) = b_1 u(t-1) + b_2 u(t-2) + \cdots\cdots + b_m u(t-m)$

$$\varphi(t) = [u(t-1), u(t-2), \cdots, u(t-m)]^T \in R^m$$

or

a nonlinear dynamic system, e.g. $\quad y(t) = b_1 u(t-1) + b_2 u(t-2) u(t-1)$

$$\varphi(t) = [u(t-1), u(t-2)u(t-1)]^T$$

Note that the parameters, b_1, b_2, are ***linearly*** involved in the input-output equation.

Using an estimated parameter vector $\hat{\theta}$, we can write a predictor that predicts the output from inputs: $\quad \hat{y}(t|\theta) = \varphi(t)^T \hat{\theta}$ (2)

We evaluate the predictor's performance by the squared error given by

$$V_N(\theta) = \frac{1}{N}\sum_{t=1}^{N}(\hat{y}(t\,|\,\theta) - y(t))^2 \tag{3}$$

Problem: Find the parameter vector $\hat{\theta}$ that minimizes the squared error:

$$\hat{\theta} = \arg\min_{\theta} V_N(\theta) \tag{4}$$

Differentiating $V_N(\theta)$ and setting it to zero,

$$\frac{dV_N(\theta)}{d\theta} = 0 \quad \Longrightarrow \quad \frac{2}{N}\sum_{t=1}^{N}(\varphi^T(t)\theta - y(t))\varphi(t) = 0 \tag{5}$$

$$\underbrace{\left[\sum_{t=1}^{N}(\varphi(t)\varphi^T(t)\right]}_{\Phi\Phi^T}\theta = \sum_{t=1}^{N} y(t)\varphi(t) \tag{6}$$

Consider $m \times N$ matrix

$$\Phi = [\varphi(1) \quad \varphi(2) \quad \dots \quad \varphi(N)] \tag{7}$$

$$m \le N$$

If vectors $\varphi(1) \quad \varphi(2) \quad \dots \quad \varphi(N)$ span the whole *m*-dimensional vector space

$rank\ \Phi = m$; full rank

If there are *m* linearly independent (column) vectors in this matrix Φ,

$rank\Phi = m$; full rank

$rank\Phi = rank\Phi\Phi^T = m$; full rank, hence invertible

Under this condition, the optimal parameter vector is given by

$$\hat{\theta} = PB \tag{8}$$

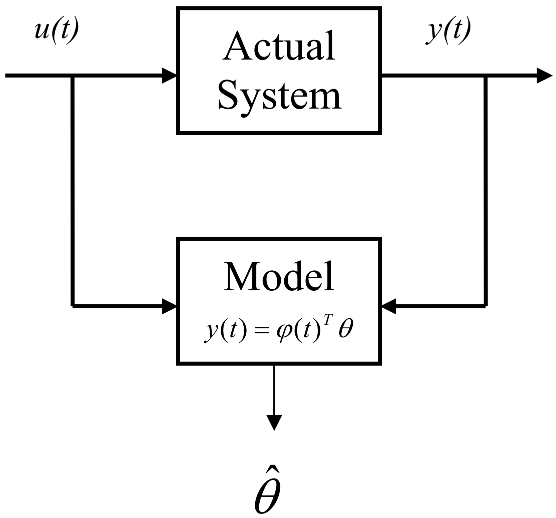
$$\text{where} \quad P = \left[\sum_{t=1}^{N}(\varphi(t)\varphi^T(t))\right]^{-1} = \left(\Phi\Phi^T\right)^{-1} \tag{9}$$

$$B = \sum_{t=1}^{N} y(t)\varphi(t) \tag{10}$$

2.2 The Recursive Least-Squares Algorithm

While the above algorithm is for batch processing of whole data, we often need to estimate parameters in real-time where data are coming from a dynamical system.

A recursive algorithm for computing the parameters is a powerful tool in such applications.



$\hat{\theta} = PB$: Batch Processing

$\Downarrow$

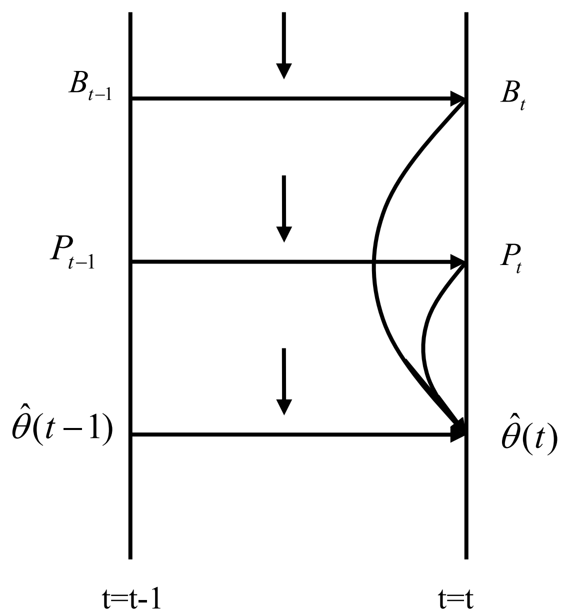
On-Line Estimation
Based on $\{y(t), \varphi(t)\}$ the latest data

We update the estimate $\hat{\theta}$

Recursive Formula

- Simple enough to complete within a given sampling period
- No need to store the whole observed data

$\hat{\theta}(t) = P_t B_t$



$\hat{\theta} = PB$

$$P = \left[\sum_{t=1}^{N} (\varphi(t)\varphi^T(t)) \right]^{-1} = \left(\Phi\Phi^T\right)^{-1} \qquad P^{-1} = \left[\sum_{t=1}^{N} (\varphi(t)\varphi^T(t)) \right] = \left(\Phi\Phi^T\right)$$

$$B = \sum_{t=1}^{N} y(t)\varphi(t)$$

Three steps for obtaining a recursive computation algorithm

a) Splitting B_t and P_t

From (10)

$$B_t = \sum_{i=1}^{t} y(i)\varphi(i) = \sum_{i=1}^{t-1} y(i)\varphi(i) + y(t)\varphi(t)$$

$$B_t = B_{t-1} + y(t)\varphi(t) \tag{11}$$

From (11)

$$P_t^{-1} = \sum_{i=1}^{t} (\varphi(i)\varphi^T(i))$$

$$P_t^{-1} = P_{t-1}{}^{-1} + \varphi(t)\varphi^T(t) \tag{12}$$

b) The Matrix Inversion Lemma (An Intuitive Method)

Premultiplying P_t and postmultiplying P_{t-1} to (12)yield

$$P_t P_t^{-1} P_{t-1} = P_t P_{t-1}{}^{-1} P_{t-1} + P_t\varphi(t)\varphi^T(t)P_{t-1}$$

$$P_{t-1} = P_t + P_t\varphi(t)\varphi^T(t)P_{t-1} \tag{13}$$

Postmultiplying $\varphi(t)$

$$P_{t-1}\varphi(t) = P_t\varphi(t) + P_t\varphi(t)\varphi^T(t)P_{t-1}\varphi(t) = P_t\varphi(t)\left(1 + \varphi^T(t)P_{t-1}\varphi(t)\right)$$

$$P_t\varphi(t) = \frac{P_{t-1}\varphi(t)}{\left(1 + \varphi^T(t)P_{t-1}\varphi(t)\right)}$$

Postmultiplying $\varphi^T(t)P_{t-1}$

$$\underbrace{P_t\varphi(t)\varphi^T(t)P_{t-1}}_{P_{t-1} - P_t} = \frac{P_{t-1}\varphi(t)\varphi^T(t)P_{t-1}}{\left(1 + \varphi^T(t)P_{t-1}\varphi(t)\right)}$$

(18)

Therefore,

$$P_t = P_{t-1} - \frac{P_{t-1}\varphi(t)\varphi^T(t)P_{t-1}}{\left(1 + \varphi^T(t)P_{t-1}\varphi(t)\right)} \tag{14}$$

Note that no matrix inversion is needed for updating P_t!

This is a special case of the Matrix Inversion Lemma.

$$\left[A + BCD\right]^{-1} = A^{-1} - A^{-1}B\left[DA^{-1}B + C^{-1}\right]^{-1}DA^{-1} \tag{15}$$

Exercise 1 Prove (15) and use (15) to obtain (14) from (12)

c) Reducing $\hat{\theta}(t) = P_t B_t$ (16)

to the following recursive form:

$$\hat{\theta}(t) = \hat{\theta}(t-1) + \mathrm{K}_t \left(\underbrace{y(t) - \varphi^T(t)\hat{\theta}(t-1)}_{\text{Prediction Error}} \right) \tag{17}$$

A type of gain for correcting the error

From (16) $\hat{\theta}(t) = P_t B_t \quad \hat{\theta}(t-1) = P_{t-1} B_{t-1}$

$$\hat{\theta}(t) - \hat{\theta}(t-1) = P_t B_t - P_{t-1} B_{t-1}$$

$$= \left(P_{t-1} - \frac{P_{t-1}\varphi(t)\varphi^T(t)P_{t-1}}{\left(1 + \varphi^T(t)P_{t-1}\varphi(t)\right)} \right) \left(B_{t-1} + y(t)\varphi(t)\right) - P_{t-1}B_{t-1}$$

$$= P_{t-1}y(t)\varphi(t) - \frac{P_{t-1}\varphi(t)\varphi^T(t)P_{t-1}}{\left(1 + \varphi^T(t)P_{t-1}\varphi(t)\right)} \left(B_{t-1} + y(t)\varphi(t)\right)$$

$$= -\frac{P_{t-1}y(t)\varphi(t)\left(1 + \varphi^T(t)P_{t-1}\varphi(t)\right) - P_{t-1}\varphi(t)\varphi^T(t)P_{t-1}\varphi(t)y(t) - P_{t-1}\varphi(t)\varphi^T(t)P_{t-1}B_{t-1}}{\left(1 + \varphi^T(t)P_{t-1}\varphi(t)\right)}$$

$$= \frac{P_{t-1}y(t)\varphi(t) - P_{t-1}\varphi(t)\varphi^T(t)\hat{\theta}(t-1)}{\left(1 + \varphi^T(t)P_{t-1}\varphi(t)\right)}$$

$$= \underbrace{\frac{P_{t-1}\varphi(t)}{\left(1 + \varphi^T(t)P_{t-1}\varphi(t)\right)}} \left[y(t) - \varphi^T(t)\hat{\theta}(t-1)\right]$$

Replacing this by $\mathrm{K}_t \in R^{m \times 1}$, we obtain (17)

The Recursive Least Squares (RLS) Algorithm

$$\hat{\theta}(t) = \hat{\theta}(t-1) + \frac{P_{t-1}\varphi(t)}{\left(1 + \varphi^T(t)P_{t-1}\varphi(t)\right)} \left(y(t) - \varphi^T(t)\hat{\theta}(t-1)\right) \tag{18}$$

$$P_t = P_{t-1} - \frac{P_{t-1}\varphi(t)\varphi^T(t)P_{t-1}}{\left(1 + \varphi^T(t)P_{t-1}\varphi(t)\right)} \quad t\text{=1,2,… w/initial conditions} \tag{14}$$

with

$\hat{\theta}(0)$: arbitrary

P_o: positive definite matrix

This Recursive Least Squares Algorithm was originally developed by Gauss (1777 – 1855)