3/12/08 Brian Fitzgibbon

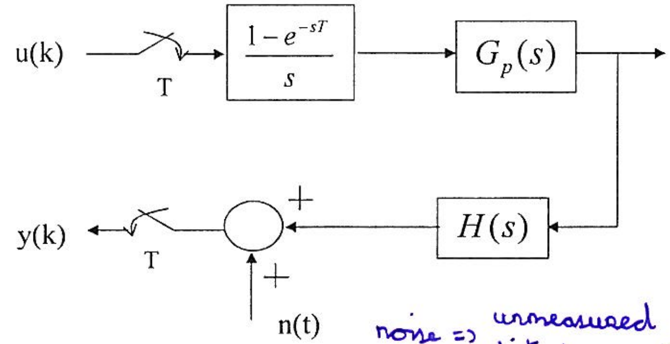
Chapter 8. System Identification

8.1 Introduction

Identify G(z) from experimental results:

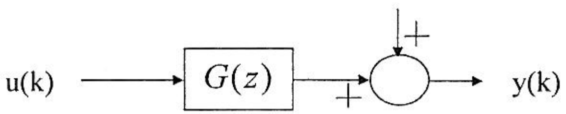
EXCITATION SEQUENCE

SAMPLED BY COMPUTER



noise => unmeasured disturbances etc

Could be represented as:



n(k) → noise sequence

Could identify G(z):

- A convolution model: $G(z) = g_0 + g_1 z^{-1} + \ldots + g_N z^{-N}$

Robust to miscalculation of the order

Simple test (could be generated easily from a step response

Many parameters needed 30-100 depends on T_s

- A transfer function:

$G(z) = B(z)/A(z)$ IIR

Order structure is important

8.2 The Least Squares Algorithm

Consider that the SISO process can be represented by the transfer function:

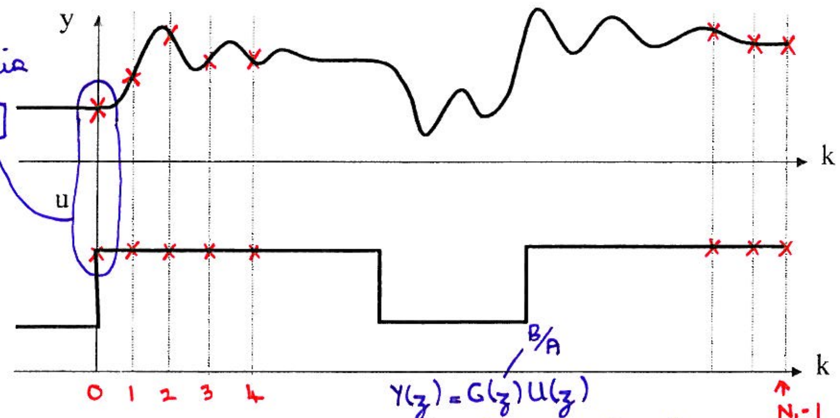
pure time delay

$$\frac{Y(z)}{U(z)} = G(z) = \frac{z^{-d}\left(b_1 z^{-1} + b_2 z^{-2} + \cdots + b_m z^{-m}\right)}{1 - a_1 z^{-1} - a_2 z^{-2} - \cdots - a_n z^{-n}} = \frac{B(z)}{A(z)}$$

order

Carry out the experiment – excite the process, collect N_1 data pairs:

data pair $[u(0), y(0)]$



$Y(z) = G(z)U(z)$ using inverse Z transform $AY = BU$

Specify the candidate model:

$$\hat{y}(k+1) = \hat{a}_1 y(k) + \hat{a}_2 y(k-1) + \cdots \hat{a}_n y(k-n+1) + \hat{b}_1 u(k-d) + \hat{b}_2 u(k-d-1) + \cdots \hat{b}_m u(k-d-m+1)$$

ESTIMATE

The first valid "test" equation is then:

$$\hat{y}(m+d) = \hat{a}_1 y(m+d-1) + \hat{a}_2 y(m+d-2) + \cdots \hat{a}_n y(m+d-n) + \hat{b}_1 u(m-1) + \hat{b}_2 u(m-2) + \cdots \hat{b}_m u(0)$$

N.B

We can repeat this, to generate output estimates over the valid dataset: USING OUR GUESS FOR $\hat{a}_1 \ldots \hat{b}_m$

$$\hat{y}(m+d)=\hat{a}_1 y(m+d-1)+\hat{a}_2 y(m+d-2)+\cdots\hat{a}_n y(m+d-n)+\hat{b}_1 u(m-1)+\hat{b}_2 u(m-2)+\cdots\hat{b}_m u(0)$$
$$\hat{y}(m+d+1)=\hat{a}_1 y(m+d)+\hat{a}_2 y(m+d-1)+\cdots\hat{a}_n y(m+d-n+1)+\hat{b}_1 u(m)+\hat{b}_2 u(m-1)+\cdots\hat{b}_m u(1)$$
$$\hat{y}(m+d+2)=\hat{a}_1 y(m+d+1)+\hat{a}_2 y(m+d)+\cdots\hat{a}_n y(m+d-n+2)+\hat{b}_1 u(m+1)+\hat{b}_2 u(m)+\cdots\hat{b}_m u(2)$$
$$\hat{y}(m+d+3)=\hat{a}_1 y(m+d+2)+\hat{a}_2 y(m+d+1)+\cdots\hat{a}_n y(m+d-n+3)+\hat{b}_1 u(m+2)+\hat{b}_2 u(m+1)+\cdots\hat{b}_m u(3)$$
$$\vdots \qquad \vdots$$
$$\vdots \qquad \vdots$$
$$\hat{y}(N_1-1)=\hat{a}_1 y(N_1-2)+\hat{a}_2 y(N_1-3)+\cdots\hat{a}_n y(N_1-1-n)+\hat{b}_1 u(N_1-d-2)+\hat{b}_2 u(N_1-d-3)+\cdots\hat{b}_m u(N_1-d-1-m)$$

we now have $(N_1 - M - d)$ valid equations

$\Rightarrow$ N equations

Could be written in matrix form as:

$$N\updownarrow\underbrace{\begin{bmatrix} \hat{y}(m+d) \\ \hat{y}(m+d+1) \\ \hat{y}(m+d+2) \\ \vdots \\ \hat{y}(N_1-2) \\ \hat{y}(N_1-1) \end{bmatrix}}_{\hat{Y}_N} = \underbrace{\left[\begin{array}{ccc|ccc} \multicolumn{3}{c|}{\overbrace{\qquad\qquad\qquad}^{n}} & \multicolumn{3}{c}{\overbrace{\qquad\qquad\qquad}^{M}} \\ y(m+d-1) & \cdots & y(m+d-n) & u(m-1) & \cdots & u(0) \\ y(m+d) & \cdots & y(m+d-n+1) & u(m) & \cdots & u(1) \\ y(m+d+1) & \cdots & y(m+d-n+2) & u(m+1) & \cdots & u(2) \\ \vdots & \vdots & & \vdots & \ddots & \vdots \\ y(N_1-3) & \ddots & y(N_1-n-2) & u(N1-d-3) & \cdots & u(N_1-d-m-2) \\ y(N_1-2) & \cdots & y(N_1-n-1) & u(N_1-d-2) & \cdots & u(N_1-d-m-1) \end{array}\right]}_{\Phi} \underbrace{\begin{bmatrix} \hat{a}_1 \\ \hat{a}_2 \\ \\ \hat{a}_n \\ \hline \hat{b}_1 \\ \hat{b}_2 \\ \\ \hat{b}_m \end{bmatrix}}_{\hat{\Theta}} \begin{array}{l} \}\, n \\ \}\, M \end{array}$$

This could be rewritten as:

$$\underline{\hat{Y}}_N = \Phi\underline{\hat{\theta}}$$

where:

Φ : Regressor matrix - contains measured samples of i/o

$\underline{\hat{\Theta}} = [\hat{a}_1 \ldots \hat{a}_n ; \hat{b}_1 \ldots \hat{b}_m]^T$ GUESS OR ESTIMATE OF PARAMETERS

We need a measure of how good our candidate model fits the data-set.

Define the Least-Squares cost function:

$$J = \sum_{i=m+d}^{N_1-1} e^2(i)$$

Sum of squared errors over the N valid data points

where: $e(i) = y(i) - \hat{y}(i)$ ← ESTIMATE

Now we will define the error vector $\underline{E}$ over the valid data-set:

$$\underline{E} = \underbrace{\begin{bmatrix} e(m+d) \\ e(m+d+1) \\ \vdots \\ e(N_1-1) \end{bmatrix}}_{\text{ERROR VECTOR}} = \underbrace{\begin{bmatrix} y(m+d) \\ y(m+d+1) \\ \vdots \\ y(N_1-1) \end{bmatrix}}_{\text{MEASUREMENT VECTOR}} - \underbrace{\begin{bmatrix} \hat{y}(m+d) \\ \hat{y}(m+d+1 \\ \vdots \\ \hat{y}(N_1-1) \end{bmatrix}}_{\text{ESTIMATE VECTOR}}$$

$$\underline{E} = \underline{Y}_N - \underline{\hat{Y}}_N$$

Then we can write the least-squares cost function as:

$$J = \sum_{i=m+d}^{N_1-1} e^2(i) = \underline{E}^T\underline{E} = [e(m+d) \ldots e(N_1-1)] \begin{bmatrix} e(m+d) \\ \vdots \\ e(N_1-1) \end{bmatrix}$$

$$= e(m+d)^2 + e(m+d+1)^2 + \ldots + e(N_1-1)^2$$

Could be rewritten as: $J = \underline{E}^T\underline{E}$

$$J = \underbrace{\left(\underline{Y}_N - \underline{\hat{Y}}\right)^T}_{E^T} \underbrace{\left(\underline{Y}_N - \underline{\hat{Y}}\right)}_{E}$$

But we know:

$$\underline{\hat{Y}}_N = \Phi\underline{\hat{\theta}}$$

$$\therefore J = (\underline{Y}_N - \Phi\underline{\hat{\Theta}})^T(\underline{Y}_N - \Phi\underline{\hat{\Theta}})$$

Revision: $(A+B)^T = A^T + B^T$

$(AB)^T = B^T A^T$

$\Rightarrow (\underline{Y}_N - \Phi\hat{\underline{\theta}})^T = \underline{Y}_N^T - (\Phi\hat{\underline{\theta}})^T$

$= \underline{Y}_N^T - \hat{\underline{\theta}}^T \Phi^T$

The cost can be expanded as follows:

N.B. SCALAR →

$$J = \left(\underline{Y}_N^{\ T} - \hat{\underline{\theta}}^T \Phi^T\right)\left(\underline{Y}_N - \Phi\hat{\underline{\theta}}\right) = (\underline{Y}_N^T - \hat{\underline{\theta}}^T \Phi^T)(\underline{Y}_N - \Phi\hat{\underline{\theta}})$$

$$= \underline{Y}_N^T \underline{Y}_N \underbrace{- \underline{Y}_N^T \Phi \hat{\underline{\theta}} - \hat{\underline{\theta}}^T \Phi^T \underline{Y}_N}_{\text{SAME}} + \hat{\underline{\theta}}^T \Phi^T \Phi \hat{\underline{\theta}}$$

Which could be rearranged to yield:

$$J = \hat{\underline{\theta}}^T \Phi^T \Phi \hat{\underline{\theta}} - 2\underline{Y}_N^{\ T} \Phi \hat{\underline{\theta}} + \underline{Y}_N^{\ T} \underline{Y}_N$$

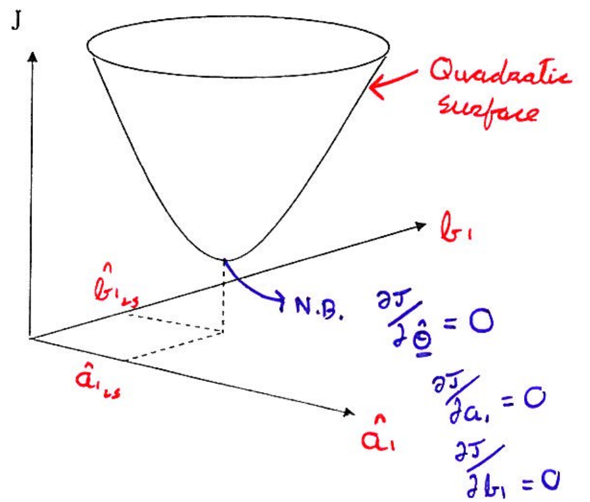
This is a quadratic function of the parameter vector:-

We will choose $\hat{\underline{\theta}}$ to minimise J "best fit" IN A LEAST SQUARES SENSE

Consider a simple example where we are trying to identify the parameters of the following first order model:

$$G(z) = \frac{b_1 z^{-1}}{1 - a_1 z^{-1}}$$

The least-squares cost function could be plotted for various choices of parameters:



N.B. minimum cost $\neq 0$ due to "noise"

INSTEAD OF SOLVING USING $\partial J / \partial \hat{\underline{\theta}} = 0$ WE WILL USE COMPLETING THE SQUARE

Solution by completing the square:

First consider

$$J = x^2 + 3x + 1$$

could easily solve for minimum point $\partial J / \partial x = 2x + 3$ min occurs when $x = -3/2$

Completing the square yields,

$J = (x + 3/2)^2 + (1 - 9/4)$

The minimum occurs at:

$x = -3/2$

Now consider:

$$J = 5x^2 + 3x + 1$$

Try the following candidate for completing the square:

$J = (x+\alpha)\,5(x+\alpha) + \beta$

Multiplying out yields,

$J = 5x^2 + 10\alpha x + 5\alpha^2 + \beta$

$\therefore 10\alpha = 3$

$\alpha = 3/10$

MIN WHEN $x = -\alpha = -3/10$

N.B. MIN COST $J_{MIN} = \beta$

Now consider the following cost based on vector $\underline{x}$

SCALAR →

$$\boxed{J = \underline{x}^T M \underline{x} - G\underline{x} + J_0} \qquad (1)$$

Consider the candidate for completing the square:

$$J = (\underline{x} - \underline{\alpha})^T M (\underline{x} - \underline{\alpha}) + \beta \qquad (2)$$

The minimum of this cost function then occurs when:

$\underline{x} = \underline{\alpha}$

Multiplying out equation (2) yields:

SAME

SCALAR → $$J = \underline{x}^T M \underline{x} - \underline{\alpha}^T M \underline{x} - \underline{x}^T M \underline{\alpha} + \underline{\alpha}^T M \underline{\alpha} + \beta \qquad (3)$$

Comparing equation (3) with the cost equation (1):

$$J = \underline{x}^T M \underline{x} - G\underline{x} + J_0 = \underline{x}^T M \underline{x} - 2\underline{\alpha}^T M \underline{x} + \left(\underline{\alpha}^T M \underline{\alpha} + \beta\right)$$

$J = \underline{x}^T M \underline{x} - 2\underline{\alpha}^T M \underline{x} + (\underline{\alpha}^T M \underline{\alpha} + \beta)$

$G\underline{x} = 2\underline{\alpha}^T M \Rightarrow G = 2\underline{\alpha}^T M$

$\tfrac{1}{2}G = \underline{\alpha}^T M \Rightarrow \tfrac{1}{2}GM^{-1} = \underline{\alpha}^T M M^{-1}$

$\underline{\alpha}^T = \tfrac{1}{2}GM^{-1}$

Hence the cost of equation (1) is a minimum when:

$\underline{x} = \underline{\alpha}$ or $\underline{x}^T = \underline{\alpha}^T = \tfrac{1}{2}GM^{-1}$

Now we will return to the least squares cost for system identification:

POSITIVE DEFINITE AND SYMMETRIC

$$J = \underline{\hat{\theta}}^T \underbrace{\Phi^T \Phi}_{M} \underline{\hat{\theta}} - \underbrace{2\underline{Y}_N \Phi}_{G} \underline{\hat{\theta}} + \underbrace{\underline{Y}_N{}^T \underline{Y}_N}_{J_0}$$

$= \underline{\hat{\Theta}}^T M \underline{\hat{\Theta}} - G\underline{\hat{\Theta}} + J_0$

$\underline{\hat{\Theta}}_{LS}^T = \underline{\alpha}^T$

This will then be minimised when:

$$\underline{\hat{\theta}}_{LS}^T = \tfrac{1}{2}GM^{-1} = \tfrac{1}{2}(\underbrace{2\underline{Y}_N \Phi}_{G})(\underbrace{\Phi^T \Phi}_{M})^{-1}$$

$\underline{\hat{\Theta}}_{LS}^T = \underline{Y}_N^T \Phi(\Phi^T \Phi)^{-1}$

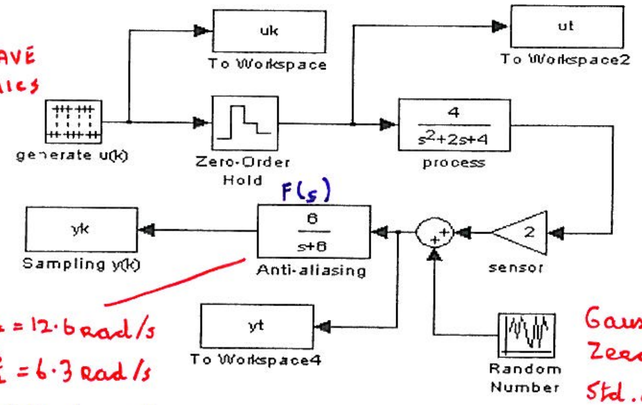
$\underline{\hat{\Theta}}_{LS}^T = (\Phi^T \Phi)^{-1} \Phi^T \underline{Y}_N$

Revision: $(ABC)^T = C^T B^T A^T$ $\quad (\Phi^T \Phi)^{-1}$ is symmetric

EXAMPLE: Sample time = 0·5 seconds

$\therefore \omega_s = \frac{2\pi}{T} = 12{\cdot}6$ rad/s

FILTER AND PLANT HAVE QUITE SIMILAR DYNAMICS
FILTER WILL DISTORT RESPONSE



Gaussian
Zero mean
Std. dev σ

$\omega_s = 12{\cdot}6$ rad/s

$\frac{\omega_s}{2} = 6{\cdot}3$ rad/s

Cutoff = 6 rad/s

The antialising filter is chosen as:

$\frac{\omega_s}{2} \approx 6$ rad/s $\qquad F(s) = \frac{6}{s+6}$

The transfer function G(z) is then determined as:

SINCE F(s) HAS AN EFFECT ON THE BASEBAND

$$G(z) = Z\left\{ \underbrace{\frac{1-e^{-sT}}{s}}_{ZOH} \underbrace{\frac{4}{s^2+2s+4}}_{G} \underbrace{2}_{H} \underbrace{\frac{6}{s+6}}_{F} \right\}$$

$$= \frac{0.3975z^{-1} + 0.652z^{-2} + 0.0565z^{-3}}{1 - 0.835z^{-1} + 0.407z^{-2} - 0.0183z^{-3}}$$ → NOTE 3rd ORDER

Hence:

TRUE $$\underline{\theta}_{true} = [a_1 \quad a_2 \quad a_3 \quad b_1 \quad b_2 \quad b_3]^T =$$
$$= [0.835 \quad -0.407 \quad 0.0183 \quad 0.3975 \quad 0.652 \quad 0.0565]^T$$

The first valid equation is: (3rd order)

$$\hat{y}k(3) = \hat{a}_1 yk(2) + \hat{a}_2 yk(1) + \hat{a}_3 yk(0) + \hat{b}_1 u(2) + \hat{b}_2 u(1) + \hat{b}_3 u(0)$$

The regressor matrix Φ is:

$$\Phi = \left[\begin{array}{ccc|ccc} yk(2) & yk(1) & yk(0) & u(2) & u(1) & u(0) \\ yk(3) & yk(2) & yk(1) & u(3) & u(2) & u(1) \\ yk(4) & yk(3) & yk(2) & u(4) & u(3) & u(2) \\ yk(5) & yk(4) & yk(3) & u(5) & u(4) & u(3) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{array}\right]$$

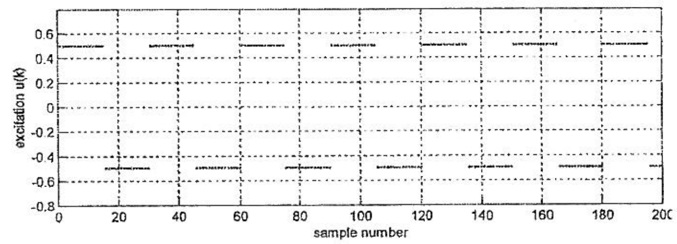
The least squares solution could be implemented as an M file:

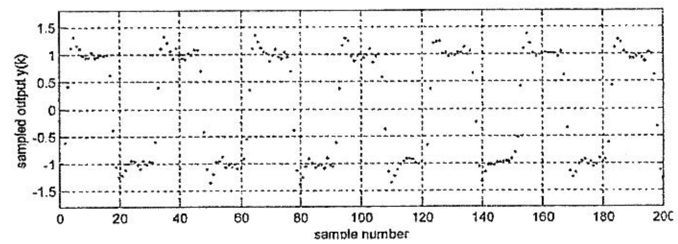
```
n=length(yk);
y=yk-1;
u=uk-0.5;
y3=y(1:n-3);
y2=y(2:n-2);
y1=y(3:n-1);
u3=u(1:n-3);
u2=u(2:n-2);
u1=u(3:n-1);
phi=[y1,y2,y3,u1,u2,u3];
yn=y(4:n);
theta=inv(phi'*phi)*phi'*yn
```

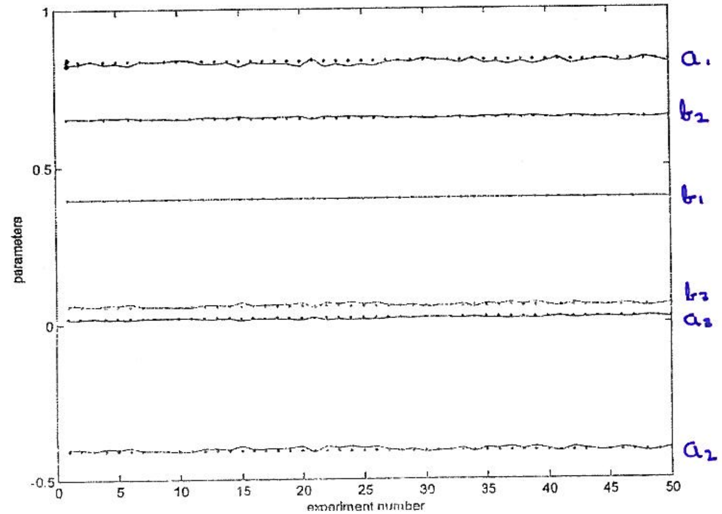
VALID MEAS →

Expt 1: Low standard deviation noise

- 50 different data sets collected each consisting of 200 points
- Parameter vector estimated for each data-set:





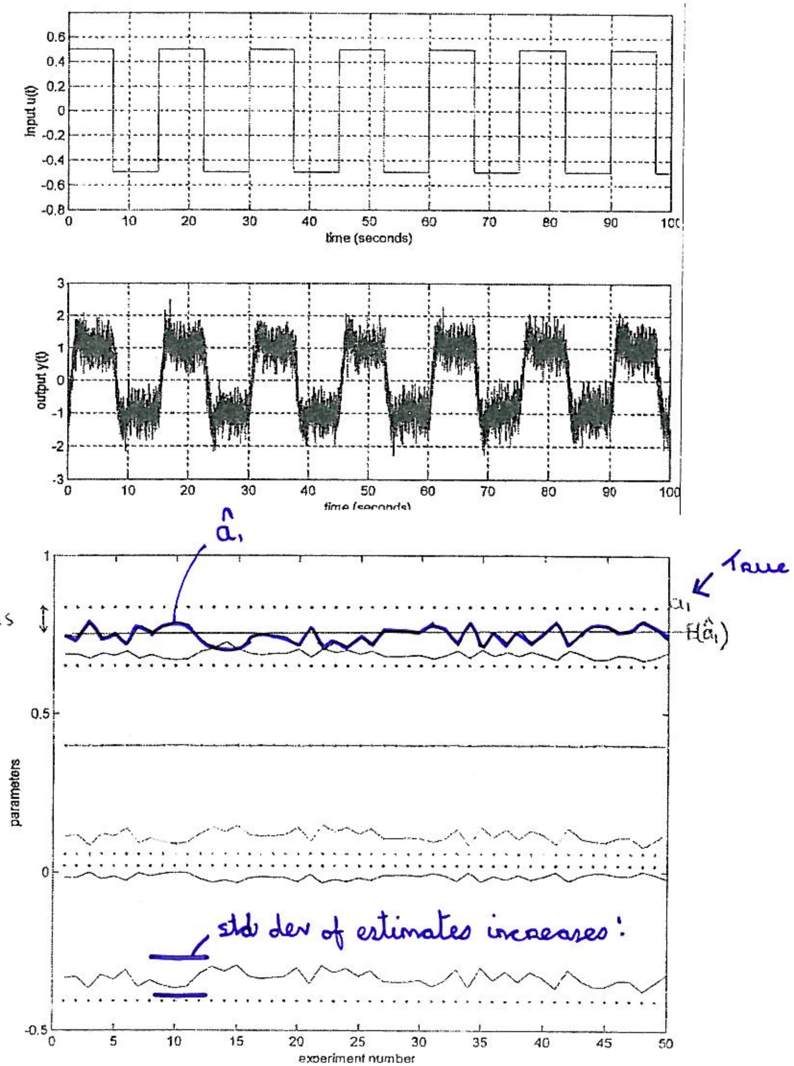


N.B THERE IS A SLIGHT BIAS

EACH DATASET IS DIFFERENT AND YIELDS A DIFFERENT PARAMETER VECTOR

Expt 2: Large standard deviation noise

- 50 different data sets collected each consisting of 200 points
- Parameter vector estimated for each data-set:



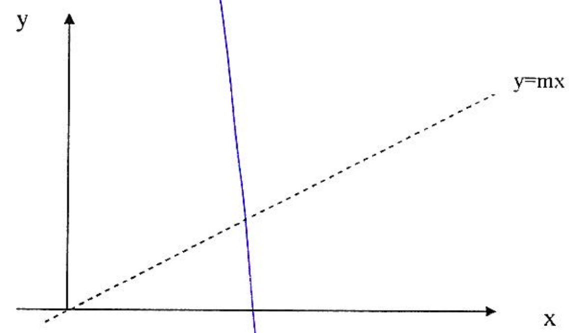
Bias ↑ as σ ↑

as noise σ ↑ → bias ↑ and std dev of parameters ↑

accuracy ↓ and uncertainty ↑

8.2.1 A Note on Bias

Consider the simple line fitting problem:



The true system is y=mx, but the data is corrupted by noise, hence the measurements x and y are related by:

$$y = mx + \varepsilon$$

If the measurement noise is random with zero mean:

Now consider the system identification problem of obtaining unbiased estimates of the A(z) and B(z) polynomials.

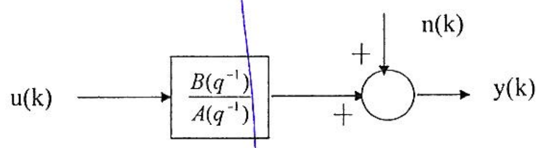
If the measurements are related via the ARX equation:

$$A(q^{-1})y(k) = B(q^{-1})u(k) + \varepsilon(k)$$

and the residual obeys:

then the parameter estimates will be unbiased.

Note: Consider the effect of sensor measurement noise:



Then:

$$y(k) = \frac{B(q^{-1})}{A(q^{-1})}u(k) + n(k)$$

Hence the estimates will be biased:

8.3 Recursive Least Squares

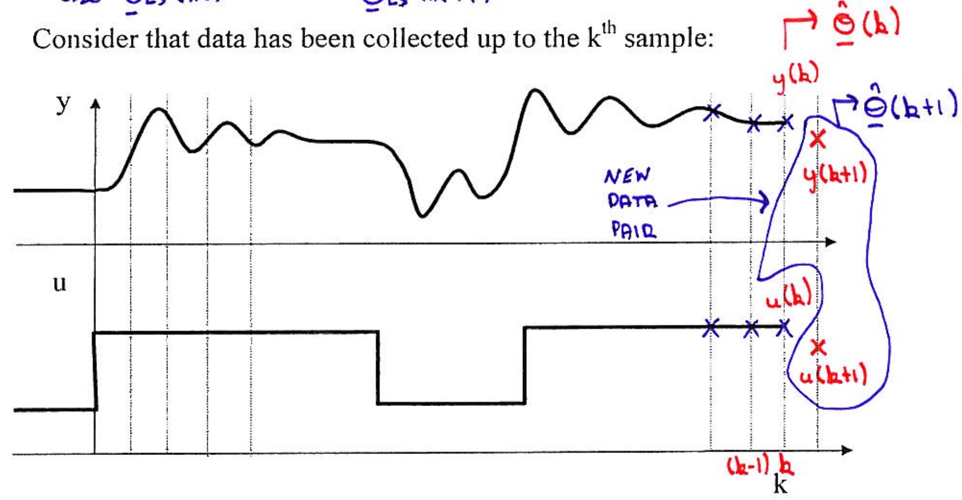
The least squares method is a batch algorithm:

$$\hat{\underline{\Theta}}_{LS} = (\Phi^T\Phi)^{-1}\Phi^T\underline{Y}$$

Recursive least Squares (RLS) is an on-line recursive algorithm – which will use each new sample pair {u(k+1), y(k+1)} to generate an updated estimate for the parameter vector.

use $\hat{\underline{\Theta}}_{LS}(k) \xrightarrow{\text{UPDATE}} \hat{\underline{\Theta}}_{LS}(k+1)$

Consider that data has been collected up to the k^{th} sample:



Define the measurement vector at time k:

$$\underline{Y}(k) = \begin{bmatrix} y(m+d) & y(m+d+1) & \cdots & y(k-1) & y(k) \end{bmatrix}^T$$

Now define the regressor matrix Φ(k) as being formed from all the valid input and output data up to time k.

Hence the estimator equation becomes:

$$\underline{\hat{Y}}(k) = \begin{bmatrix} \hat{y}(m+d) \\ \hat{y}(m+d+1) \\ \vdots \\ \hat{y}(k) \end{bmatrix} = \Phi(k)\underline{\hat{\theta}}$$

And the least squares solution to the parameters at time k is :

N.B.
as time unfolds $k \uparrow$
$\underline{Y}(k)$ and $\underline{\Phi}(k)$
become longer

$$\underline{\hat{\theta}}_{LS}(k) = \left(\Phi(k)^T\Phi(k)\right)^{-1}\Phi(k)^T\underline{Y}(k)$$

$(m+n) \times (m+n)$ → INVERSE IS NUMERICALLY INTENSIVE

Now consider that we have available the input/output pair sampled at the (k+1)th sample.

The measurement vector then becomes:

$$\underline{Y}(k+1) = \begin{bmatrix} y(m+d) \\ y(m+d+1) \\ \vdots \\ y(k) \\ \hline y(k+1) \end{bmatrix} = \begin{bmatrix} \underline{Y}(k) \\ \hline y(k+1) \end{bmatrix}$$

← new appended data (target)

The estimated output vector is:

$$\underline{\hat{Y}}(k+1) = \begin{bmatrix} \hat{y}(m+d) \\ \hat{y}(m+d+1) \\ \vdots \\ \hat{y}(k) \\ \hline \hat{y}(k+1) \end{bmatrix} = \begin{bmatrix} \underline{\hat{Y}}(k) \\ \hline \hat{y}(k+1) \end{bmatrix}$$

— new prediction

Where:

$$\hat{y}(k+1) = \hat{a}_1 y(k) + \cdots \hat{a}_n y(k-n+1) + \hat{b}_1 u(k-d) + \cdots \hat{b}_m u(k-d-m+1)$$

Or :

$$\hat{y}(k+1) = \underline{\psi}^T(k+1)\,\underline{\hat{\theta}}$$

$$\psi(k+1) = \begin{bmatrix} y(k) \\ y(k-1) \\ y(k-n+1) \\ \hline u(k-d) \\ u(k-d-n+1) \end{bmatrix} \qquad \underline{\hat{\Theta}} = \begin{bmatrix} \hat{a}_1 \\ \vdots \\ \hat{a}_n \\ \hline \hat{b}_1 \\ \vdots \\ \hat{b}_m \end{bmatrix}$$

The estimator equation can then be written as:

$$\begin{array}{c} \hat{\underline{Y}}(k) \\ \begin{bmatrix} \hat{y}(m+d) \\ \hat{y}(m+d+1) \\ \vdots \\ \vdots \\ \hat{y}(k) \\ \hline \hat{y}(k+1) \end{bmatrix} \end{array} = \begin{array}{c} \underline{\Phi}(k) \\ \begin{bmatrix} y(m+d-1) & \cdots & y(m+d-n) & u(m-1) & \cdots & u(0) \\ y(m+d) & \cdots & y(m+d-n+1) & u(m) & \cdots & u(1) \\ y(m+d+1) & \cdots & y(m+d-n+2) & u(m+1) & \cdots & u(2) \\ \vdots & \vdots & & \vdots & \ddots & \vdots \\ y(k-1) & \ddots & y(k-n) & u(k-d-1) & \cdots & u(k-d-m) \\ \hline y(k) & \cdots & y(k-n+1) & u(k-d) & \cdots & u(k-d-m+1) \end{bmatrix} \end{array} \begin{bmatrix} \hat{a}_1 \\ \hat{a}_2 \\ \vdots \\ \hat{a}_n \\ \hat{b}_1 \\ \hat{b}_2 \\ \vdots \\ \hat{b}_m \end{bmatrix}$$

$\underline{\psi}^T(k+1)$

$\hat{Y}(k+1) = \underline{\Phi}(k+1)\,\underline{\hat{\Theta}}$

$\uparrow \underline{\hat{\Theta}}$

Which could be written as:

$$\underline{\hat{Y}}(k+1) = \begin{bmatrix} \underline{\hat{Y}}(k) \\ \hline \hat{y}(k+1) \end{bmatrix} = \begin{bmatrix} \Phi(k) \\ \hline \underline{\psi}^T(k+1) \end{bmatrix}\underline{\hat{\theta}}$$

The updated parameter vector could be calculated as:

$$\underline{\hat{\theta}}_{LS}(k+1) = \left(\Phi(k+1)^T\Phi(k+1)\right)^{-1}\Phi(k+1)^T\underline{Y}(k+1)$$

as time progresses this becomes ridiculous!!!
⇒ memory and calc. time

Revision $\begin{bmatrix} A \\ \hdashline B \end{bmatrix}^T = [A^T \,\vdots\, B^T]$

Consider:

$$\Phi(k+1)^T\Phi(k+1) = \begin{bmatrix} \Phi(k) \\ \hdashline \underline{\psi}^T(k+1) \end{bmatrix}^T \begin{bmatrix} \Phi(k) \\ \hdashline \underline{\psi}^T(k+1) \end{bmatrix} = [\phi(k)^T \,\vdots\, \psi(k+1)] \begin{bmatrix} \phi(k) \\ \hdashline \psi^T(k+1) \end{bmatrix}$$

$$= \phi(k)^T\phi(k) + \psi(k+1)\psi^T(k+1)$$

Now we will define:

$$P(k) = \left(\Phi(k)^T\Phi(k)\right)^{-1}$$
$$P(k+1) = \left(\Phi(k+1)^T\Phi(k+1)\right)^{-1}$$

Hence we can now write:

$$P(k+1) = \left[\Phi(k)^T\Phi(k) + \underline{\psi}(k+1)\underline{\psi}^T(k+1)\right]^{-1}$$

$$= [P(k)^{-1} + \psi(k+1)\psi^T(k+1)]^{-1}$$

Householder's Matrix Inversion lemma states:

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

Choose: $A = P(k)^{-1}$ $\quad B = \psi(k+1)$

N.B. $\rightarrow C = 1$ $\quad D = \psi^T(k+1)$

sets dimension for inverse

Then:

$$P(k+1) = \Big[\underbrace{P(k)^{-1}}_{A} + \underbrace{\underline{\psi}(k+1)}_{B}\,\underbrace{\vphantom{\psi}}_{C}\,\underbrace{\underline{\psi}^T(k+1)}_{D}\Big]^{-1}$$

Could be written as:

$$P(k+1) = A^{-1} - A^{-1}B(C^{-1} + DA^{-1}B)^{-1}DA^{-1}$$
$$= P(k) - P(k)\psi(k+1)[1 + \psi^T(k+1)P(k)\psi(k+1)]^{-1}\psi^T P(k)$$

But $(1+\psi^T(k+1)P(k)\psi(k+1))^{-1} = \dfrac{1}{1+\psi^T P\psi}$ SCALAR !!!

NO INVERSION OF A LARGE MATRIX IS REQUIRED → if we know $P(k)$ → we get $P(k+1)$

This can be simplified to:

$$P(k+1) = P(k) - \frac{P(k)\underline{\psi}(k+1)\underline{\psi}^T(k+1)P(k)}{1+\underline{\psi}^T(k+1)P(k)\underline{\psi}(k+1)}$$

The updated parameter vector is then generated from:

$$\underline{\hat{\theta}}_{LS}(k+1) = \left(\Phi(k+1)^T\Phi(k+1)\right)^{-1}\Phi(k+1)^T\underline{Y}(k+1)$$

$$\underline{\hat{\theta}}_{LS}(k+1) = P(k+1)\underbrace{\phi^T(k+1)\underline{Y}(k+1)}_{\text{memory!!}}$$

But:

$$\underbrace{\Phi(k+1)^T}_{\phi(k+1)}\underbrace{\underline{Y}(k+1)}_{\underline{Y}(k+1)} = \begin{bmatrix} \Phi(k) \\ \hdashline \underline{\psi}^T(k+1) \end{bmatrix}^T \begin{bmatrix} \underline{Y}(k) \\ \hdashline y(k+1) \end{bmatrix} = [\phi(k)^T \,\vdots\, \psi(k+1)] \begin{bmatrix} \underline{Y}(k) \\ \hdashline y(k+1) \end{bmatrix}$$

$$= \phi(k)^T\underline{Y}(k) + \psi(k+1)y(k+1)$$

The parameter estimate can then be written as:

$$\underline{\hat{\theta}}_{LS}(k+1) = \overbrace{\left[P(k) - \frac{P(k)\underline{\psi}(k+1)\underline{\psi}^T(k+1)P(k)}{1+\underline{\psi}^T(k+1)P(k)\underline{\psi}(k+1)}\right]}^{P(k+1)}\overbrace{\left[\Phi(k)^T\underline{Y}(k) + \underline{\psi}(k+1)y(k+1)\right]}^{\phi(k+1)\underline{Y}(k+1)}$$

Multiplying this out: $[\phi(k)^T\phi(k)]^{-1}\phi(k)^T\underline{Y}(k)$

$$\underline{\hat{\theta}}_{LS}(k+1) = \underbrace{P(k)\Phi(k)^T\underline{Y}(k)}_{\underline{\hat{\theta}}_{LS}(k)} + P(k)\underline{\psi}(k+1)y(k+1)$$
$$- \frac{P(k)\underline{\psi}(k+1)\underline{\psi}^T(k+1)P(k)}{1+\underline{\psi}^T(k+1)P(k)\underline{\psi}(k+1)}\Phi(k)^T\underline{Y}(k) - \frac{P(k)\underline{\psi}(k+1)\underline{\psi}^T(k+1)P(k)}{1+\underline{\psi}^T(k+1)P(k)\underline{\psi}(k+1)}\Phi(k)^T\underline{\psi}(k+1)y(k+1)$$

TUTORIAL:

Show that the parameter update algorithm can be expressed as:
NEW DATA BECOMES AVAILABLE $[u(k+1), y(k+1)] \rightarrow \psi(k+1)$

$$L(k+1)=\frac{P(k)\underline{\psi}(k+1)}{1+\underline{\psi}^T(k+1)P(k)\underline{\psi}(k+1)}$$

$$\hat{y}(k+1)=\underline{\psi}^T(k+1)\underline{\hat{\theta}}_{LS}(k)$$

$$\underline{\hat{\theta}}_{LS}(k+1)=\underline{\hat{\theta}}_{LS}(k)+L(k+1)\overbrace{\left(y(k+1)-\hat{y}(k+1)\right)}^{\text{PREDICTION ERROR}}$$

$$P(k+1)=\left[P(k)-\frac{P(k)\underline{\psi}(k+1)\underline{\psi}^T(k+1)P(k)}{1+\underline{\psi}^T(k+1)P(k)\underline{\psi}(k+1)}\right]$$

8.3.1 A Note on the Choice of Excitation Signal

The need to excite the dynamics of the process over the frequency range of interest – consider the solution to batch least squares:

$$\boxed{\underline{\hat{\theta}}_{LS}=\left(\Phi^T\Phi\right)^{-1}\Phi^T\underline{Y}}$$

If the data collected is not rich enough in information then:
$\phi^T\phi$ will become rank deficient and we will not get inverse

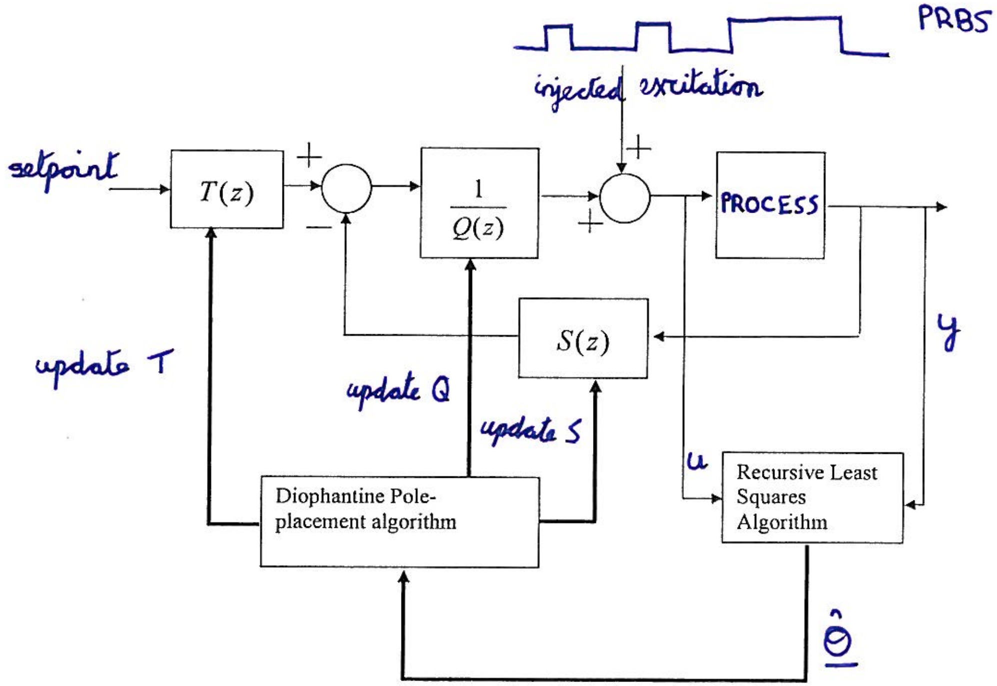
Choices of excitation include:

* sinusoids - mixture of freq range
* chirp
* square wave - contains odd harmonics
  - careful about frequency
* Pseudo Random Binary Sequence PRBS - it looks like white noise

8.3.2 Adaptive Pole Placement Control

Use RLS algorithm to provide on-line updates of the parameters – hopefully this will allow controller to track slow changes in the process.

Each new update of the parameter vector is then used, to design the controller polynomials, based on the polynomial pole-placement technique.



* note non linearity
* need for "packetting software" to protect the RLS estimator