MODEL VALIDATION

"HOW WELL ARE WE DOING?"

- VARIOUS CODES EXIST IN THE TOOLBOX

COMPARE - COMPARE MODEL'S SIMULATED OR PREDICTED OUTPUT WITH ACTUAL OUTPUT

IDSIM - SIMULATE A MODEL

PE - COMPUTE PREDICTION ERRORS

PREDICT - PREDICT FUTURE OUTPUTS

RESID - COMPUTE AND TEST RESIDUALS.

⇒ TRY AT LEAST ONE, IF NOT TWO, OF THESE.

- THE LONGER THE PREDICTION HORIZON, THE MORE DEMANDING THE TASK FOR THE MODEL.

VALIDATION - DETAILS

- USUALLY WE DO NOT KNOW THE "ACTUAL SYSTEM" DYNAMICS - SO HOW DO WE ESTABLISH IF OUR MODEL IS GOOD?

- VARIOUS TYPES OF TESTS CAN BE PERFORMED
  - PREDICTION AND SIMULATION ERRORS
  - FREQUENCY RESPONSE FIT

  ⇒ MAKE SURE YOU USE DIFFERENT DATA TO VALIDATE (IF POSSIBLE).

- CAN ALSO PERFORM A VERY DETAILED ANALYSIS OF THE <u>RESIDUALS</u> (LL 511)

$$\begin{aligned} \epsilon(t) &= y(t) - \hat{y}(t|t-1) \\ &= y(t) - (1-H^{-1})\,y(t) - H^{-1}G\,u(t) \\ &= H^{-1}\left(y(t) - G\,u(t)\right) \end{aligned}$$

⇒ CALLED THE "INNOVATIONS PROCESS" AND IT CONTAINS A LOT OF INFORMATION ABOUT THE QUALITY OF OUR FIT

- DESIRABLE <u>PROPERTIES FOR THE RESIDUALS:</u>

  ① NORMALLY DISTRIBUTED (AT LEAST SYMMETRIC)

  ② ZERO MEAN

  ③ WHITE NOISE PROCESS

  ④ INDEPENDENT (UNCORRELATED) WITH PAST INPUTS

①-③: BASICALLY WANT $\epsilon(t)$ TO LOOK LIKE WHAT WE ASSUMED FOR $e(t)$

④: IF THERE ARE TRACES OF PAST INPUTS IN THE RESIDUALS, THEN THERE IS A PART OF $y(t)$ THAT ORIGINATES FROM THE INPUT AND WAS NOT CAPTURED WELL IN OUR MODEL ⇒ BAD!

- ANALYZE ① WITH A HISTOGRAM OF $\epsilon(t)$
- ANALYZE ③ WITH $\hat{R}_\epsilon(\tau) = \frac{1}{N} \sum_{t=\tau}^{N} \epsilon(t)\epsilon(t-\tau)$
  - RESIDUAL AUTOCORRELATION.
  - DESIRED SHAPE?
- ANALYZE ④ WITH $\hat{R}_{\epsilon u}(\tau) = \frac{1}{N} \sum_{t=\tau}^{N} \epsilon(t)u(t-\tau)$
  - CROSS-CORRELATION
  - $\tau > 0$ CORRELATES $\epsilon(t)$ WITH OLD $u(t-\tau)$
  - DESIRED SHAPE?

- BOTH ANALYSIS TESTS OF THE CORRELATION GRAPH NEED A MEASURE OF "SMALL ENOUGH"
  - MUST DEVELOP THIS FROM THE DATA AS WELL.
  - ⇒ CAN DEVELOP THIS BY ANALYZING THE STATISTICS OF THE RESIDUALS

- <u>WHITENESS</u> LET $\Gamma = \frac{1}{\hat{R}_\epsilon(0)} \begin{bmatrix} \hat{R}_\epsilon(1) \\ \vdots \\ \hat{R}_\epsilon(m) \end{bmatrix}$

  THEN, ACCORDING TO THE CENTRAL LIMIT THEOREM, AS $N \to \infty$

  $$\sqrt{N}\,r \sim N(0, I)$$

  - I.E., IN THE LIMIT, $\sqrt{N}\,r$ WILL BE NORMALLY DISTRIBUTED WITH UNIT VARIANCE.

- CAN CONTINUE THE THIS ANALYSIS AND SHOW THAT $\gamma_{N,m} = N r^T r$ WILL LIMIT TO A $\chi^2(m)$ DISTRIBUTION WHICH GIVES A SIMPLE OVERALL TEST

- MORE INSTRUCTIVE IS TO LOOK AT THE <u>CONFIDENCE INTERVALS</u> FOR A NORMAL DISTRIBUTION

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}$$

PROB $\{ \lvert x \rvert > \sigma \}$	CONFIDENCE LEVEL	CONFIDENCE INTERVAL
0.001	99.9%	$\mu \pm 3.29\sigma$
0.005	99.5%	$\mu \pm 3.09\sigma$
0.01	99%	$\mu \pm 2.58\sigma$
0.05	95%	$\mu \pm 1.96\sigma$

- SO, FOR A 95% CONFIDENCE LEVEL WE CAN USE THE $\pm 1.96/\sqrt{N}$ BOUNDS TO DECIDE IF THE ϵ AUTOCORRELATION IS SMALL FOR $\tau > 0$
  - $\Rightarrow$ PLOT $r(k)$ $1 \le k \le M$
  - $\Rightarrow$ TEST FOR NORMALITY BY ENSURING THAT $r(k)$ WITHIN THE CONFIDENCE INTERVAL $\forall$ K.

RESID.M

- <u>CROSSCORRELATION TEST.</u>
  - CAN SHOW THAT, AS $N \to \infty$

$$\sqrt{N}\,\hat{R}_{\epsilon u}(\tau) \sim N(0, P_r)$$

WITH

$$P_r = \sum_{k=-\infty}^{\infty} R_\epsilon(k)\, R_u(k)$$

  - $\Rightarrow$ CAN PERFORM A NORMALITY TEST ON $\hat{R}_{\epsilon u}(\tau)$ BY CHECKING IF $\left|\hat{R}_{\epsilon u}(\tau)\right| \le 1.96\sqrt{\frac{P_r}{N}}$ $\forall \tau$
  - $\Rightarrow$ IF $\hat{R}_{\epsilon u}(\tau)$ IS OUTSIDE THESE BOUNDS, THEN FOR THOSE VALUES OF τ, $\epsilon(t)$ AND $u(t-\tau)$ ARE PROBABLY DEPENDENT.

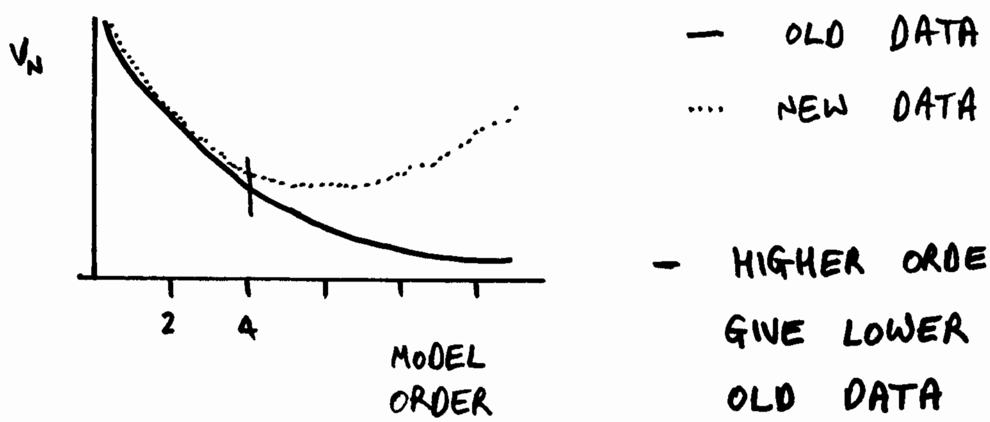
- DEPENDENCY FOR SMALL τ COULD IMPLY THE NEED FOR LESS DELAY IN THE MODEL.

MODEL SELECTION

- BE CAREFUL COMPARING MODELS USING THE SAME SET OF DATA USED TO MAKE THEM.
  - A LARGER MODEL WILL ALWAYS GIVE A BETTER FIT (LOWER $V_N(\hat{\theta})$)

$\Rightarrow$ MUST USE NEW DATA TO COMPARE
  - GOOD MODELS WILL STILL GIVE GOOD PREDICTIONS ON THE NEW DATA AS WELL.

- TYPICAL SCENARIO



- ACTUAL ORDER 4

- HIGHER ORDER MODELS GIVE LOWER V_N ON OLD DATA
- BUT "OVERFIT" THE DATA BY INCLUDING KNOWLEDGE OF PARTICULAR NOISE MEASURED

- THIS EXTRA "NOISE" INFORMATION IS NOT USEFUL TO US SINCE WE PLAN TO USE IT ON DATA WITH DIFFERENT NOISE.

- SO PEOPLE HAVE DEVELOPED MODIFIED COST FUNCTIONS OF THE FORM

$$J = V_N(\theta)\left(1 + U_N\right)$$

FIRST TERM: STANDARD COST

SECOND TERM: PROVIDES A MEASURE OF THE COMPLEXITY OF THE MODEL.

⇒ TYPICALLY HAVE $V_N \downarrow$ WITH MODEL SIZE INCREASE, BUT $U_N \uparrow$.

⇒ GIVES US A WAY TO TRADE OFF IMPROVEMENTS IN V_N AGAINST MODEL COMPLEXITY

- STANDARD CRITERIA:

  AKAIKE INF. CRITERION (AIC) – $U_N = \dfrac{2d}{N}$

  MIN DESCRIPTION LENGTH (MDL) – $U_N = \dfrac{\log N}{N} d$

  d ~ DIMENSION OF θ

- OBJECTIVE NOW: $\min\limits_{d,\theta} J$

  ⇒ FOR FIXED d, $\min\limits_{\theta} J = J_d^*$

  ⇒ PLOT J_d^* VS. d AND SELECT LOWEST VALUE.

- ACCESSIBLE FOR ARX MODELS IN ARXSTRUC.M.

STATE SPACE FORM

- DISCRETE TIME MODELS WRITTEN IN TERMS OF STATE SPACE DIFFERENCE EQUATION

$$x_{k+1} = A x_k + B u_k \qquad k \geq 0$$
$$y_k = C x_k \qquad \text{ASSUME } x_0 = 0$$

- CONSIDER RESPONSE TO A UNIT DISCRETE IMPULSE (I.E. $u_k = 1$ IFF $k=0$)

- RESPONSE

$$y_0 = C x_0 = 0 \qquad x_1 = B$$
$$y_1 = C x_1 = CB \qquad x_2 = AB$$
$$y_2 = C x_2 = CAB \qquad x_3 = A^2 B$$
$$\vdots$$
$$y_k = CA^{k-1}B \qquad k \geq 1$$

- THE TERMS $h_k = CA^{k-1}B$ ARE CALLED THE MARKOV PARAMETERS OF THE SYSTEM
  - $\Rightarrow$ THE MARKOV PARAMETERS ARE THE VALUES OF THE DISCRETE-TIME IMPULSE RESPONSE

(SEE KAILATH , CHEN)

HANKEL MATRIX

- AN IMPORTANT MATRIX ASSOCIATED WITH MARKOV PARAMETERS

$$M_{ij} = \begin{bmatrix} h_i & h_{i+1} & h_{i+2} & \cdots & h_{i+j} \\ h_{i+1} & h_{i+2} & & & \\ h_{i+2} & & \ddots & & \\ \vdots & & & \ddots & \\ h_{i+j} & & & & h_{i+2j} \end{bmatrix}$$

– ELEMENTS OF THE HANKEL MATRIX ARE THE MARKOV PARAMETERS – CONSTANT ALONG ANTI-DIAGONALS

- REALLY IMPORTANT TO NOTE THAT:

$$\begin{bmatrix} C \\ CA \end{bmatrix} \begin{bmatrix} B & AB \end{bmatrix} = \begin{bmatrix} CB & CAB \\ CAB & CA^2B \end{bmatrix} = \begin{bmatrix} h_1 & h_2 \\ h_2 & h_3 \end{bmatrix}$$

$\begin{bmatrix} C \\ CA \end{bmatrix} \rightarrow M_o$ <u>OBSERVABILITY MATRIX</u>

$\begin{bmatrix} B & AB \end{bmatrix} \rightarrow M_c \sim$ <u>CONTROLLABILITY MATRIX</u>

$\Rightarrow$ CLOSE CONNECTION BETWEEN BETWEEN HANKEL MATRIX AND M_o, M_c.

- INTERESTING CONNECTION, BUT HOW USE THIS? NOTE THAT:

  - $M_{1(j-1)} = M_{o_j} M_{c_j}$ $\qquad M_{o_j} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{j-1} \end{bmatrix} \quad M_{c_j} = \begin{bmatrix} B & AB & \dots & A^{j-1}B \end{bmatrix}$

  - $M_{2(j-1)} = M_{o_j} A M_{c_j}$

⇒ THESE TWO LEAD TO A NATURAL <u>SYSTEM REALIZATION PROCESS</u> (I.E. HOW TO GET A, B, C)

- PROVIDED THAT WE CAN FIND THE MARKOV PARAMETERS FROM THE MEASURED DATA

SYSTEM REALIZATION

- $M_{1(j-1)} = M_{oj} M_{cj}$ $\Rightarrow$ USE SVD OF $M_{1(j-1)}$ TO FIND A, B, C (SQUARE)

- SVD $M_{1(j-1)} = U \Sigma V^*$ $U^* U = I$, $V^* V = I$

  Σ - $n \times n$ DIAGONAL MATRIX

  WHERE $n = \text{RANK}(M_{1(j-1)})$

  $\rightarrow$ HAVE $j \geq n$

- WITH NON-SINGULAR T, WRITE $M_{1(j-1)} = (U \Sigma^{1/2} T)(T^{-1} \Sigma^{1/2} V^*)$

  $\Rightarrow$ $M_{oj} = U \Sigma^{1/2} T$

  $M_{cj} = T^{-1} \Sigma^{1/2} V^*$ (CAN USE $T = I$)

- CAN GET - C FROM FIRST n_y ROWS OF $U \Sigma^{1/2} T$

  - B FROM FIRST n_y COLS OF $T^{-1} \Sigma^{1/2} V^*$

- FIND A BY SOLVING

$$M_{2(j-1)} = M_{oj} A M_{cj} = U \Sigma^{1/2} T A T^{-1} \Sigma^{1/2} V^*$$

$$\Rightarrow A = T^{-1} \Sigma^{-1/2} U^* M_{2(j-1)} V \Sigma^{-1/2} T$$

ISSUES WITH THIS ALGORITHM

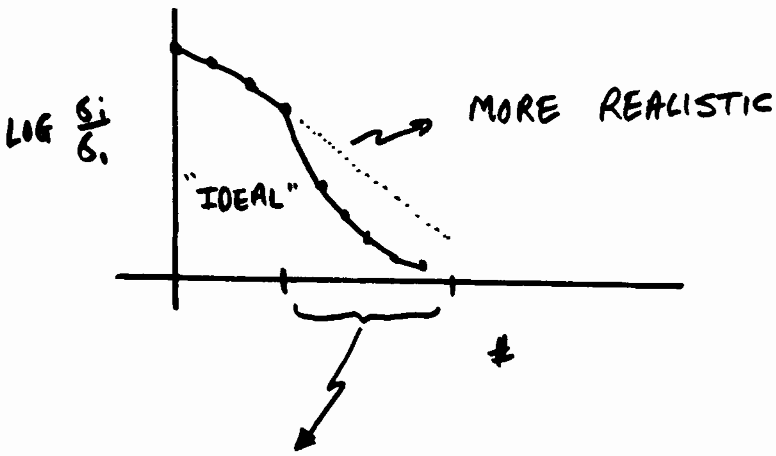
- NEED TO RECORD THE MARKOV PARAMETERS
- IN THEORY $M_{1(j-1)}$ WOULD BE OF RANK n_A

  ⇒ WOULD THEN KNOW DIMENSION OF THE SYSTEM (A MATRIX)

- PROBLEM: $M_{1(j-1)}$ IS USUALLY FULL RANK
  - DUE TO SENSOR NOISE AND NONLINERITIES.

⇒ WHEN YOU CALCULATE THE SINGULAR VALUES YOU FIND THAT

$$\text{DIAG}(\tilde{\Sigma}_M) = (\sigma_1, \sigma_2, \ldots, \sigma_n, \underbrace{\sigma_{n+1}, \ldots, \sigma_p}_{\neq 0,\ \text{BUT "SMALL"}})$$



⇒ TRUNCATE MODEL SIZE AT "n" SO THAT

$$\text{DIAG}(\Sigma_M) \simeq (\sigma_1, \sigma_2, \ldots, \sigma_n, 0, 0, \ldots)$$

– DYNAMICS ASSOCIATED WITH THESE S.V.'S ARE A BLEND OF NOISE, NONLINEARITIES, ETC.

DO YOU WANT THESE IN THE MODEL?

Lecture #9

State Space Models

Subspace ID

Thanks to Bart deMoor, P. Van Overschee, Bo Wahlberg, and M. Jansson

LL 208-211 & section 10.6

Introduction

- Assumed truth model form:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + w_k \\ y_k &= Cx_k + Du_k + v_k \end{aligned}$$

  – x is $n \times 1$, y is $m \times 1$ and u is $r \times 1$

  – w (process noise) and v (sensor noise) are assumed to be stationary, zero-mean, white Gaussian noises.

$$R = \mathcal{E}\left\{ \begin{bmatrix} w_k \\ v_k \end{bmatrix} \begin{bmatrix} w_k^T & v_k^T \end{bmatrix} \right\}$$

  i.e. in this case we explicitly include the noises.

- **Objectives:** Use the measured data y_k, u_k, $k = 1, \ldots, N$ to

  1. Estimate the system order n
  2. Estimate a model that is similar to the true description,
  3. Estimate the noise covariances so that we can design a Kalman Filter.

- **Basic point:** given the state response of the system (x_k), it is a simple *linear regression* to find the plant model matrices A, B, C, D.
  - **Reason:** If x_k known $\forall k$, then we can rewrite

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + w_k \\ y_k &= Cx_k + Du_k + v_k \end{aligned}$$

as

$$\overline{Y}_k = \Theta\Phi_k + E_k$$

where

$$\overline{Y}_k = \begin{bmatrix} x_{k+1} \\ y_k \end{bmatrix}, \quad \Theta = \begin{bmatrix} A & B \\ C & D \end{bmatrix}, \quad \Phi_k = \begin{bmatrix} x_k \\ u_k \end{bmatrix}, \quad E_k = \begin{bmatrix} w_k \\ v_k \end{bmatrix},$$

- Could then estimate the covariance matrix using the square of the model residuals (as we did before)

$$\hat{R} = \frac{1}{N}\sum_{k=1}^{N} E_k E_k^T$$

and then use this to solve for the Kalman filter gain K

- **Primary motivation for Subspace approach:**
If we can develop a reasonable estimate for the state x_k from the measured data, then it is relatively easy to develop a model of the plant model matrices A, B, C, D.

Subspace Identification

- Subspace ID based on the development of predictors for **future** outputs using old values of the **inputs** and **outputs**.
  - Predictors will depend on several unknown matrices.
  - Difference these predictions with measured data (over all time) to form the *prediction error*.
  - Define a cost function that minimizes these prediction error

  ⇒ Minimize this cost to solve for the unknowns.

- Solution allows us to define one possible set of system states x_k, $\forall k$
  - Can then solve for the model matrices.

Predictor Representation

- General model input/output form

$$x_{k+1} = Ax_k + Bu_k + w_k \text{ and } y_k = Cx_k + Du_k + v_k$$

- For future outputs

$$\begin{aligned} y_{k+1} &= Cx_{k+1} + Du_{k+1} + v_{k+1} \\ &= C\left[Ax_k + Bu_k + w_k\right] + Du_{k+1} + v_{k+1} \\ &= CAx_k + \left[\begin{array}{cc} CB & D \end{array}\right] \left[\begin{array}{c} u_k \\ u_{k+1} \end{array}\right] + (Cw_k + v_{k+1}) \end{aligned}$$

$$\begin{aligned} y_{k+2} &= Cx_{k+2} + Du_{k+2} + v_{k+2} \\ &= C\left[Ax_{k+1} + Bu_{k+1} + w_{k+1}\right] + Du_{k+2} + v_{k+2} \\ &= C\left[A(Ax_k + Bu_k + w_k) + Bu_{k+1} + w_{k+1}\right] + Du_{k+2} + v_{k+2} \\ &= CA^2x_k + \left[\begin{array}{ccc} CAB & CB & D \end{array}\right] \left[\begin{array}{c} u_k \\ u_{k+1} \\ u_{k+2} \end{array}\right] + (CAw_k + Cw_{k+1} + v_{k+2}) \end{aligned}$$

- Collecting terms we get

$$\left[\begin{array}{c} y_k \\ y_{k+1} \\ y_{k+2} \end{array}\right] = \left[\begin{array}{c} C \\ CA \\ CA^2 \end{array}\right] x_k + \left[\begin{array}{ccc} D & 0 & 0 \\ CB & D & 0 \\ CAB & CB & D \end{array}\right] \left[\begin{array}{c} u_k \\ u_{k+1} \\ u_{k+2} \end{array}\right] + \left[\begin{array}{c} \eta_k \\ \eta_{k+1} \\ \eta_{k+2} \end{array}\right]$$

- The full form is then

$$\mathbf{y}_\alpha(k) = \mathcal{M}_o^\alpha x_k + S_\alpha \mathbf{u}_\alpha(k) + \eta_\alpha(k) \qquad \textbf{(KP \#1)}$$

where

$$\mathbf{y}_\alpha(k) = \begin{bmatrix} y_k \\ \vdots \\ y_{k+\alpha-1} \end{bmatrix}, \mathbf{u}_\alpha(k) = \begin{bmatrix} u_k \\ \vdots \\ u_{k+\alpha-1} \end{bmatrix}$$

$$\mathcal{M}_o^\alpha = \begin{bmatrix} C \\ \vdots \\ CA^{\alpha-1} \end{bmatrix}, \quad S_\alpha = \begin{bmatrix} D & 0 & \cdots & 0 \\ CB & D & & 0 \\ \vdots & & \ddots & 0 \\ CA^{\alpha-2}B & CA^{\alpha-3}B & \cdots & D \end{bmatrix}$$

and $\mathcal{M}_o^\alpha$ is the *extended observability matrix*

- Notes:
  - $\mathbf{y}_\alpha(k)$, $\mathbf{u}_\alpha(k)$, and $\eta_\alpha(k)$ all contain present and future data
  - All past information needed to predict the future response is embedded in the present state x_k.

- **(KP #2)** Since x_k contains all past information, can show that the *mean-square* optimal prediction of $\mathbf{y}_\alpha(k)$ **given data upto time** $k-1$ is

$$\hat{\mathbf{y}}_\alpha(k) = \mathcal{M}_o^\alpha x_k$$

  - noises white, so our best estimate of the future values is zero.
  - $\mathbf{u}_\alpha(k)$ contains future inputs.

Algorithm - First Cut

- Assume $\hat{\mathbf{y}}_\alpha(k)$ known $\forall k = 1, \ldots, N$, could write

$$\begin{aligned} \hat{\mathbf{Y}}_\alpha &= \begin{bmatrix} \hat{\mathbf{y}}_\alpha(1) & \hat{\mathbf{y}}_\alpha(2) & \cdots & \hat{\mathbf{y}}_\alpha(N) \end{bmatrix} \qquad (\alpha m \times N) \\ \mathbf{X} &= \begin{bmatrix} x_1 & x_2 & \cdots & x_N \end{bmatrix} \qquad (n \times N) \end{aligned}$$

$$\Rightarrow \hat{\mathbf{Y}}_\alpha = \mathcal{M}_o^\alpha \mathbf{X}$$

- Interesting, but what does $\hat{\mathbf{Y}}_\alpha$ look like? Let $\alpha = 3$, then

$$\begin{aligned} \hat{\mathbf{y}}_3(1) &= \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \hat{y}_3 \end{bmatrix} \\ \Rightarrow \hat{\mathbf{Y}}_3 &= \begin{bmatrix} \hat{\mathbf{y}}_3(1) & \hat{\mathbf{y}}_3(2) & \hat{\mathbf{y}}_3(3) \end{bmatrix} \quad \text{a \textbf{block Hankel matrix}} \\ &= \left[\begin{array}{c|c|c} \hat{y}_1 & \hat{y}_2 & \hat{y}_3 \\ \hat{y}_2 & \hat{y}_3 & \hat{y}_4 \\ \hat{y}_3 & \hat{y}_4 & \hat{y}_5 \end{array}\right] \end{aligned}$$

- If $\hat{\mathbf{Y}}_\alpha$ not know, but we can estimate it (e.g. using least squares) as $\hat{\hat{\mathbf{Y}}}_\alpha$ then:
  - $\hat{\hat{\mathbf{Y}}}_\alpha$ is rank deficient (why?) $\rightarrow$ determine the system order.
  - form *low-rank factorization* of $\hat{\hat{\mathbf{Y}}}_\alpha$ to estimate $\mathcal{M}_o^\alpha$ and $\mathbf{X}$

$$\hat{\hat{\mathbf{Y}}}_\alpha = \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \approx \begin{bmatrix} \cdot \\ \cdot \\ \cdot \end{bmatrix} \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} = \mathcal{M}_o^\alpha \mathbf{X}$$

$\Rightarrow$ Can do this factorization using an SVD (again).

Low-rank Factorizations

- Assume that we do an SVD of a matrix and get

$$\begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} = \begin{bmatrix} (1) & (2) & (3) \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix} \begin{bmatrix} 4 & 0 & 0 & 0 & 0 \\ 0 & 10^{-5} & 0 & 0 & 0 \\ 0 & 0 & \epsilon & 0 & 0 \end{bmatrix} \begin{bmatrix} (1) & \cdot & \cdot & \cdot & \cdot \\ (2) & \cdot & \cdot & \cdot & \cdot \\ (3) & \cdot & \cdot & \cdot & \cdot \\ (4) & \cdot & \cdot & \cdot & \cdot \\ (5) & \cdot & \cdot & \cdot & \cdot \end{bmatrix}$$

$$\approx \begin{bmatrix} (1) & (2) & (3) \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix} \begin{bmatrix} 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} (1) & \cdot & \cdot & \cdot & \cdot \\ (2) & \cdot & \cdot & \cdot & \cdot \\ (3) & \cdot & \cdot & \cdot & \cdot \\ (4) & \cdot & \cdot & \cdot & \cdot \\ (5) & \cdot & \cdot & \cdot & \cdot \end{bmatrix}$$

$$= \begin{bmatrix} 2 \times (1) \\ \cdot \\ \cdot \end{bmatrix} \begin{bmatrix} 2 \times (1) & \cdot & \cdot & \cdot & \cdot \end{bmatrix}$$

- This is a rank-one representation of a 3×5 matrix.
- How big an error is there in this approximation?
- Other form:

$$Y = U\Sigma V^T = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix}$$

$$\approx U_1 \Sigma_1 V_1^T = \left[U_1 \Sigma_1^{1/2} \right] \cdot \left[\Sigma_1^{1/2} V_1^T \right]$$

- Note that the number of singular values retained determines the number of columns in U_1

Subspace Algorithm

- Previous Algorithm focused on finding an estimate for the state, but it turns out to be better to instead focus on finding $\mathcal{M}_o^\alpha$
  - in fact subspace estimation refers to the estimation of the extended observability matrix $\mathcal{M}_o^\alpha$

- Key remaining component then is to develop an algorithm to solve for an estimate of $\hat{\mathbf{Y}}_\alpha$ from the measured data.

- Three main steps:
  1. Develop an estimate for the state x_k that can be used in the equation
$$\mathbf{y}_\alpha(k) = \mathcal{M}_o^\alpha x_k + S_\alpha \mathbf{u}_\alpha(k) + \eta_\alpha(k)$$
$$\Rightarrow \hat{x}_k$$
  2. Use $\hat{x}_k$ in the expression for our estimator $\hat{\mathbf{y}}_\alpha(k)$
  3. Form block Hankel matrices (measured data and predicted responses), difference these to develop the prediction error, and select parameters to optimize
$$\min \|\mathbf{Y}_\alpha^{data} - \hat{\mathbf{Y}}_\alpha^{pred}\|_F^2$$

$\|A\|_F^2 = \text{Trace}(A^\star A)$

- **Step #1:** best linear mean-square estimate for x_k given

$$\mathbf{y}_\beta(k-\beta) = \begin{bmatrix} y_{k-\beta} \\ y_{k-\beta+1} \\ \vdots \\ y_{k-1} \end{bmatrix}, \mathbf{u}_\beta(k-\beta) = \begin{bmatrix} u_{k-\beta} \\ u_{k-\beta+1} \\ \vdots \\ u_{k-1} \end{bmatrix}, \mathbf{u}_\alpha(k) = \begin{bmatrix} u_k \\ u_{k+1} \\ \vdots \\ u_{k+\alpha-1} \end{bmatrix}$$

  is

$$\hat{x}_k = K_1 \mathbf{y}_\beta(k-\beta) + K_2 \mathbf{u}_\beta(k-\beta) + K_3 \mathbf{u}_\alpha(k)$$

  - $\mathbf{y}_\beta(k-\beta)$ and $\mathbf{u}_\beta(k-\beta)$ contain (truncated) past data
  - $\mathbf{u}_\alpha(k)$ contains future input data

- β is a design parameter – typically will set $\beta = \alpha$. Corresponds to the memory of the estimator.
  - expect performance to improve as β increased (usually the case)
  - numerical complexity clearly balloons with α and β

- Estimate $\hat{x}_k$ is non-causal since it uses future inputs
  - the past input sequence is truncated to length β. If past and future inputs are correlated, then it would be advantageous to use future inputs as well (i.e. non-causal filter)
    $\rightarrow$ should improve our estimate of $\hat{x}_k$
  - not a big deal since we are not working in real-time

- **Step #2:** Use this $\hat{x}_k$ to develop $\hat{\mathbf{y}}_\alpha(k)$. If we start with

$$\mathbf{y}_\alpha(k) = \mathcal{M}_o^\alpha x_k + S_\alpha \mathbf{u}_\alpha(k) + \eta_\alpha(k)$$

and replace x_k with $\hat{x}_k$ to get

$$\begin{aligned}
\Rightarrow \mathbf{y}_\alpha(k) &= \mathcal{M}_o^\alpha \hat{x}_k + S_\alpha \mathbf{u}_\alpha(k) + \mathbf{e}_\alpha(k) \\
&= \mathcal{M}_o^\alpha \left[K_1 \mathbf{y}_\beta(k-\beta) + K_2 \mathbf{u}_\beta(k-\beta) + K_3 \mathbf{u}_\alpha(k)\right] \\
&\qquad + S_\alpha \mathbf{u}_\alpha(k) + \mathbf{e}_\alpha(k) \\
&= L_1 \mathbf{y}_\beta(k-\beta) + L_2 \mathbf{u}_\beta(k-\beta) + L_3 \mathbf{u}_\alpha(k) + \mathbf{e}_\alpha(k)
\end{aligned}$$

- $\mathbf{e}_\alpha(k)$ consists of the future process and sensor noises, as well as the future state estimation error. Thus our best estimate is zero.

$$\Rightarrow \hat{\mathbf{y}}_\alpha(k) = L_1 \mathbf{y}_\beta(k-\beta) + L_2 \mathbf{u}_\beta(k-\beta) + L_3 \mathbf{u}_\alpha(k)$$

or, for example, if $k = \beta + 1$

$$\hat{\mathbf{y}}_\alpha(1+\beta) = L_1 \mathbf{y}_\beta(1) + L_2 \mathbf{u}_\beta(1) + L_3 \mathbf{u}_\alpha(\beta+1)$$

- So the best estimate of the future outputs is a linear combination of the measured data.

- **Step #3:** Form block Hankel matrices
- Collect all possible α-ahead predictors using data (first starts at $\beta+1$ to leave enough room to populate the *old* data columns).

$$\hat{\mathbf{Y}}_\alpha^{pred} \equiv \begin{bmatrix} \hat{\mathbf{y}}_\alpha(\beta+1) & \hat{\mathbf{y}}_\alpha(\beta+2) & \cdots & \hat{\mathbf{y}}_\alpha(N-\alpha+1) \end{bmatrix}$$

$$= \left[\begin{array}{c|c|c|c} \hat{y}(\beta+1) & \hat{y}(\beta+2) & \cdots & \hat{y}(N-\alpha+1) \\ \hat{y}(\beta+2) & \hat{y}(\beta+3) & \cdots & \hat{y}(N-\alpha+2) \\ \vdots & & & \vdots \\ \hat{y}(\beta+\alpha) & \hat{y}(\beta+\alpha+1) & \cdots & \hat{y}(N) \end{array}\right]$$

$$\rightarrow \hat{\mathbf{Y}}_\alpha^{pred} = L_1\mathbf{Y}_\beta + L_2\mathbf{U}_\beta + L_3\mathbf{U}_\alpha$$

$$\mathbf{Y}_\alpha^{data} = \text{similar form, but populated with data}$$

where

$$\mathbf{Y}_\beta = \begin{bmatrix} \mathbf{y}_\beta(1) & \mathbf{y}_\beta(2) & \cdots & \mathbf{y}_\beta(N-\alpha-\beta+1) \end{bmatrix}$$

$$= \left[\begin{array}{c|c|c|c} y(1) & y(2) & \cdots & y(N-\alpha-\beta+1) \\ y(2) & y(3) & \cdots & y(N-\alpha-\beta) \\ \vdots & & & \vdots \\ y(\beta) & y(\beta+1) & \cdots & y(N-\alpha) \end{array}\right]$$

$$\mathbf{U}_\beta = \begin{bmatrix} \mathbf{u}_\beta(1) & \mathbf{u}_\beta(2) & \cdots & \mathbf{u}_\beta(N-\alpha-\beta+1) \end{bmatrix}$$

$$\mathbf{U}_\alpha = \begin{bmatrix} \mathbf{u}_\alpha(\beta+1) & \mathbf{u}_\alpha(\beta+2) & \cdots & \mathbf{u}_\alpha(N-\alpha+1) \end{bmatrix}$$

- Clearly these are all just block Hankel matrices populated with the measured input and output data.

Solution Algorithm

- Now pick L_i to optimize

$$\min_{L_1,\ L_2,\ L_3} \|\mathbf{Y}_\alpha^{data} - \hat{\mathbf{Y}}_\alpha^{pred}\|_F^2$$

- Note that L_3 unconstrained and in step #2 we showed that

$$\begin{bmatrix} L_1 & L_2 \end{bmatrix} = \mathcal{M}_o^\alpha \begin{bmatrix} K_1 & K_2 \end{bmatrix}$$

so we must have that

$$\text{Rank}\left(\begin{bmatrix} L_1 & L_2 \end{bmatrix}\right) = n$$

- Given $\begin{bmatrix} L_1 & L_2 \end{bmatrix}$, can do a low-rank factorization and solve for $\mathcal{M}_o^\alpha$.

$$\begin{bmatrix} L_1 & L_2 \end{bmatrix} = \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \approx \begin{bmatrix} \cdot \\ \cdot \\ \cdot \end{bmatrix} \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} = \mathcal{M}_o^\alpha \begin{bmatrix} K_1 & K_2 \end{bmatrix}$$

Note: number of columns of $\mathcal{M}_o^\alpha \equiv$ system order (why?)

- Given $\mathcal{M}_o^\alpha$ can solve for the matrix C. To find the A, note that

$$J_1 = \begin{bmatrix} I_{(\alpha-1)m} & 0_{(\alpha-1)m\times m} \end{bmatrix} \text{ then } J_1\mathcal{M}_o^\alpha = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{\alpha-2} \end{bmatrix}$$

$$J_2 = \begin{bmatrix} 0_{(\alpha-1)m\times m} & I_{(\alpha-1)m} \end{bmatrix} \text{ then } J_2\mathcal{M}_o^\alpha = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{\alpha-1} \end{bmatrix}$$

$$\Rightarrow J_1\mathcal{M}_o^\alpha A = J_2\mathcal{M}_o^\alpha$$

which gives us

$$\hat{A} = (J_1\mathcal{M}_o^\alpha)^\dagger J_2\mathcal{M}_o^\alpha$$

- Similar techniques can be used to solve for B and D
  - these are much easier to find since the transfer function from u_k to y_k is linear in B and D

Solution Algorithm

- Now pick L_i to optimize

$$\min_{L_1,\, L_2,\, L_3} \|\mathbf{Y}_\alpha^{data} - \hat{\mathbf{Y}}_\alpha^{pred}\|_F^2$$

- Note that L_3 unconstrained and in step #2 we showed that

$$\begin{bmatrix} L_1 & L_2 \end{bmatrix} = \mathcal{M}_o^\alpha \begin{bmatrix} K_1 & K_2 \end{bmatrix}$$

so we must have that

$$\text{Rank}\left(\begin{bmatrix} L_1 & L_2 \end{bmatrix}\right) = n$$

- Given $\begin{bmatrix} L_1 & L_2 \end{bmatrix}$, can do a low-rank factorization and solve for $\mathcal{M}_o^\alpha$.

$$\begin{bmatrix} L_1 & L_2 \end{bmatrix} = \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \approx \begin{bmatrix} \cdot \\ \cdot \\ \cdot \end{bmatrix} \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} = \mathcal{M}_o^\alpha \begin{bmatrix} K_1 & K_2 \end{bmatrix}$$

Note: number of columns of $\mathcal{M}_o^\alpha \equiv$ system order (why?)

Core Algorithm

- Let $\theta_c = \|\mathbf{Y}_\alpha^{data} - (L_1\mathbf{Y}_\beta + L_2\mathbf{U}_\beta + L_3\mathbf{U}_\alpha)\|_F^2$
- And $\overline{L} = \begin{bmatrix} L_1 & L_2 \end{bmatrix}$

- Since L_3 unconstrained, we can solve for that directly

$$\begin{aligned} L_3 &= \left[\mathbf{Y}_\alpha^{data} - (L_1\mathbf{Y}_\beta + L_2\mathbf{U}_\beta)\right]\mathbf{U}_\alpha^\dagger \\ &= \left[\mathbf{Y}_\alpha^{data} - \begin{bmatrix} L_1 & L_2 \end{bmatrix}\begin{bmatrix} \mathbf{Y}_\beta \\ \mathbf{U}_\beta \end{bmatrix}\right]\mathbf{U}_\alpha^\dagger \\ &= \left[\mathbf{Y}_\alpha^{data} - \overline{L}\mathcal{P}_\beta\right]\mathbf{U}_\alpha^\dagger \end{aligned}$$

- Substitute in for L_3 and use $\mathbf{U}_\alpha^\perp = I - \mathbf{U}_\alpha^\dagger\mathbf{U}_\alpha$

$$\begin{aligned} \tilde{\theta}_c &= \|\mathbf{Y}_\alpha^{data} - (\overline{L}\mathcal{P}_\beta + \left[\mathbf{Y}_\alpha^{data} - \overline{L}\mathcal{P}_\beta\right]\mathbf{U}_\alpha^\dagger\mathbf{U}_\alpha)\|_F^2 \\ &= \|\mathbf{Y}_\alpha^{data}\mathbf{U}_\alpha^\perp - \overline{L}\mathcal{P}_\beta\mathbf{U}_\alpha^\perp\|_F^2 \end{aligned}$$

$$\min_{\overline{L}} \tilde{\theta}_c \Rightarrow \overline{L} = \mathbf{Y}_\alpha^{data}\mathbf{U}_\alpha^\perp(\mathcal{P}_\beta\mathbf{U}_\alpha^\perp)^\dagger$$

- Then we can do an SVD of $\overline{L}$ and look for the largest singular values. By selecting n of them, we define the **order of the system.** (see 9-13)

N4SID Algorithm

```
function [TH,bestchoice,nchoice,failflag] = ...
          n4sid(z,order,l,auxord,dkx,maxsize,Tsamp,refine,arg,trace)
%N4SID   Estimates a state-space model using a sub-space method.
%    TH=N4SID(Z) or [TH,AO]=N4SID(Z,ORDER,NY,AUXORD,DKX,MAXSIZE,TSAMP)
%
%    TH: Returned as the estimated state-space model in the THETA format.
%        No model covariances are given.
%    Z : The output input data [y u], with y and u as column vectors
%        For multi-variable systems, Z=[y1 y2 ... yp u1 u2 ... un]
%    ORDER: The order of the model (Dimension of state vector). If entered
%        as a vector (e.g. 3:10) information about all these orders will be
%        given in a plot, Default; ORDER=1:10;
%        If ORDER is entered as 'best', the default order among 1:10 is
%        chosen.
%    NY: The number of outputs in the data matrix. Default NY =1.
%    AUXORD: An auxiliary order, that is used for the selection of state
%        variables. Default 1.2*ORDER+3. If AUXORD is entered as a row vector
%        the best value (min pred error) in this vector will be selected.
%    DKX: This is a vector defining the structure: DKX =[D,K,X]
%        D=1 indicates that a direct term from input to output will be
%                estimated, while D=0 means that a delay from input to output
%            is postulated.
%        K=1 indicates that the K-matrix is estimated, while K=0 means that
%                K will be fixed to zero.
%        X=1 indicates that the initial state is estimated, X=0 that the
%            initial state is set to zero.
%        To define an arbitrary input delay structure NK, where NK(ku) is
%        the delay from input number ku to any of the outputs, let
%        DKX=[D,K,X,NK]. NK is thus a row vector of length=no of input
%        channels. When NK is specified, it overrides the value of D.
%            Default: DKX = [0, 1, 1]
%        TRACE: Letting the last given argument be 'trace' gives info to screen
%        about fit and choice of AUXORD
%        MAXSIZE: See also AUXVAR.
%
%    AO: The chosen value of AUXORD.
%
%    The algorithm implements Van Overschee's and De Moor's method for
%    identification of general multivariable linear systems in state space.
%    See also CANSTART, PEM.

%    M. Viberg, 8-13-1992, T. McKelvey, L. Ljung 9-26-1993.
%    Copyright (c) 1986-98 by The MathWorks, Inc.
%    $Revision: 3.5 $  $Date: 1997/12/02 03:40:05
```

Notes

- Need to select α and β (typically set $\alpha = \beta \approx 1.5\hat{n}$)

- No nonlinear optimizations

- Then need to determine where to **cut** when we do the approximate low-rank factorizations $\rightarrow$ same as seleting the model order.
  - The model order includes the dynamics for both G and H.

- Note that N4SID explicitly allows you to try various model orders (e.g. $n = 1 : 10$)

- Note from the manual:

  auxord: An auxiliary order used by the algorithm. This can be seen as a prediction horizon, and it should be larger than the order. The default value is auxord = 1.2× order+3. The choice of auxord could have a substantial influence on the model quality, and there are no simple rules for how to choose it.

- Note distinction from OKID - we never once mentioned Markov parameters.

- Many researchers in this area (Larimore [CVA], Verhaegen [MOESP], and Overschee/DeMoor [N4SID])

- **Example:** robot arm data that you already analyzed.

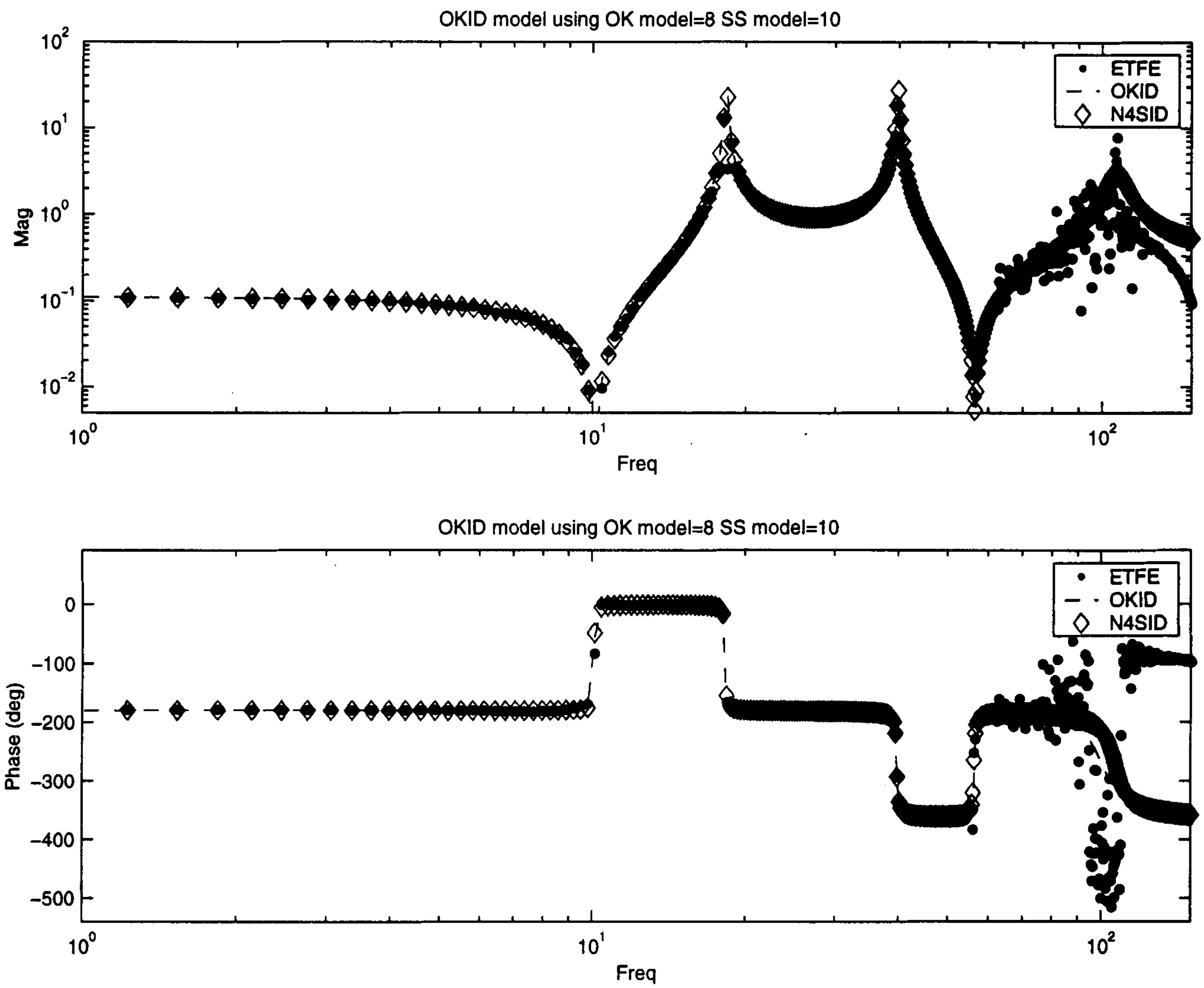
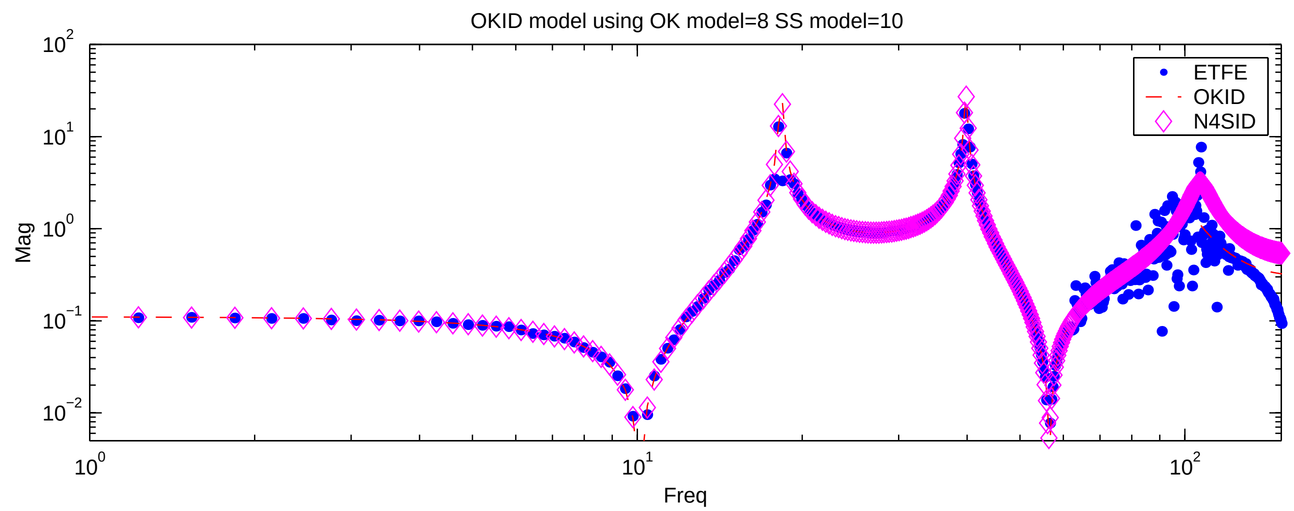
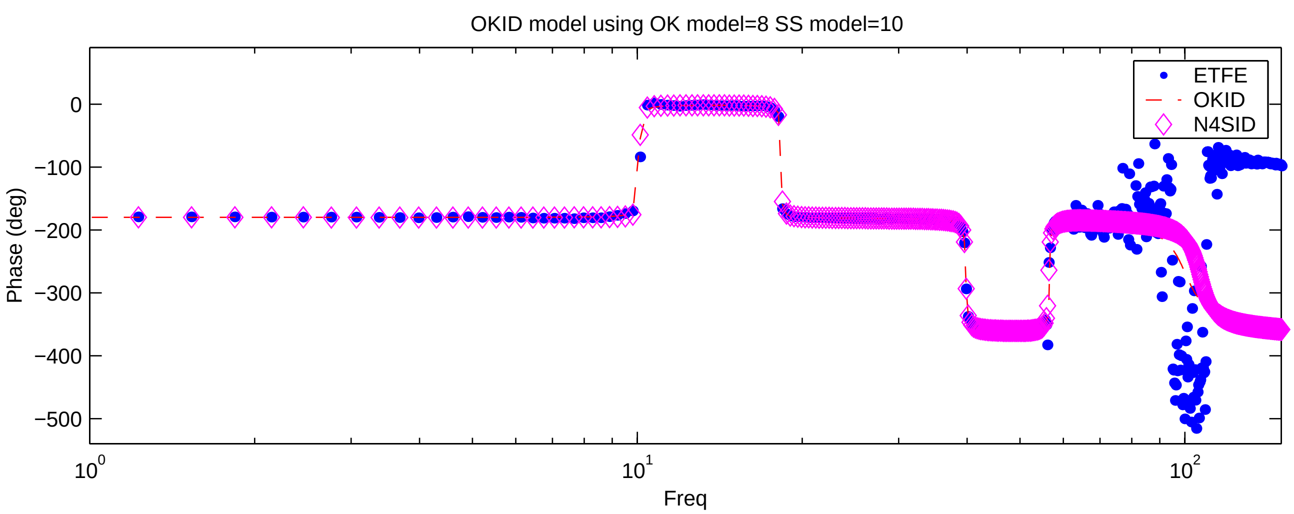


Figure 1: TF's

- Seems to provide a very reasonable fit to the data with a 10th order model.


OKID model using OK model=8 SS model=10
Mag
Freq
ETFE
OKID
N4SID
10^{2}
10^{1}
10^{0}
10^{-1}
10^{-2}


OKID model using OK model=8 SS model=10
Phase (deg)
Freq
ETFE
OKID
N4SID
0
−100
−200
−300
−400
−500

- **Example:** Consider the nonwhite noise example from before.
- 6th order system model in OKID.
- 4th order system model from N4SID.

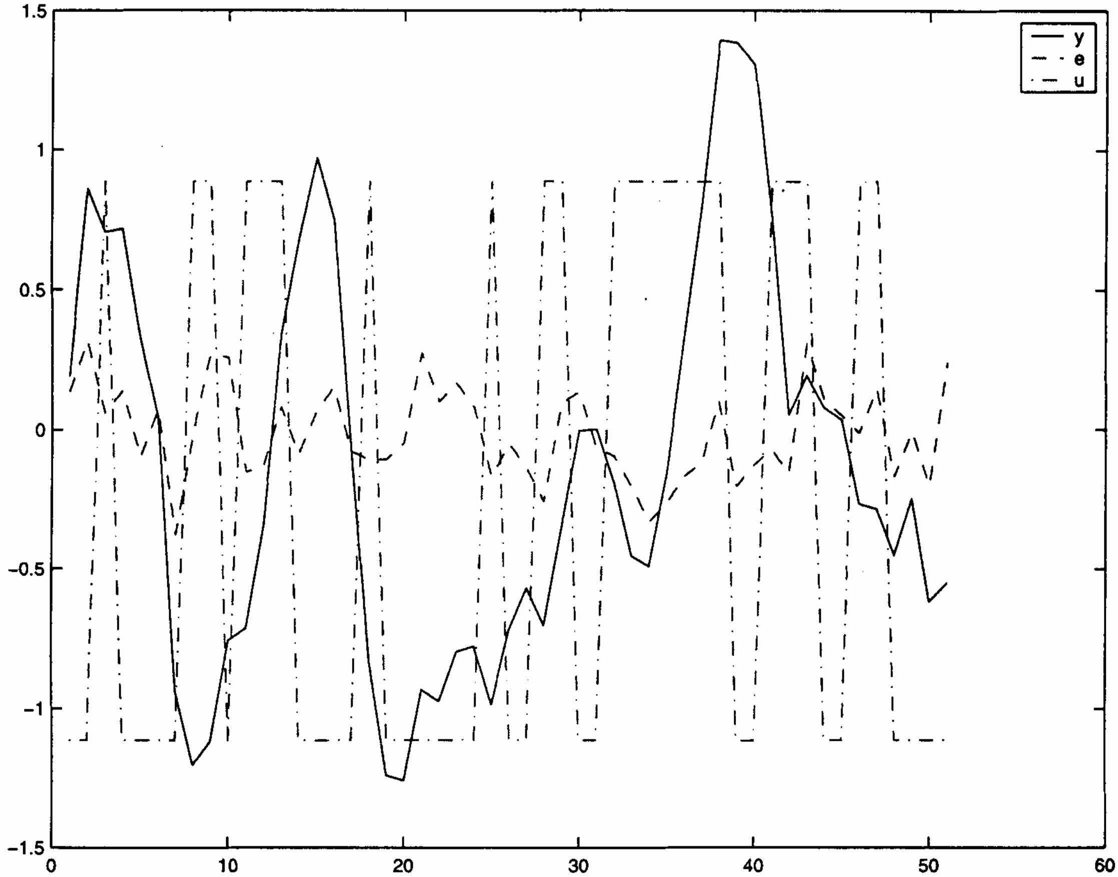


Figure 2: SIGNALS - NONWHITE NOISE EXAMPLE

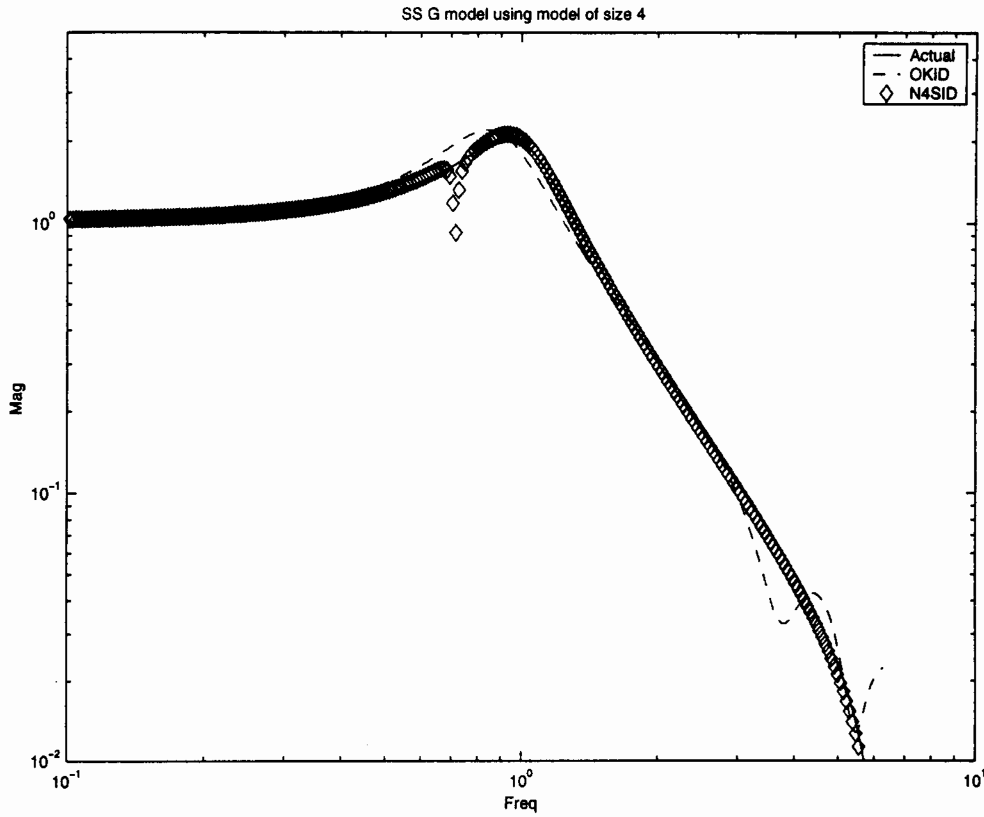


Figure 3: ESTIMATE AND ACTUAL G (NOTE EFFECT OF IMPERFECT POLE/ZERO CANCELATION OF THE DYNAMICS THAT ARE ASSOCIATED WITH H)

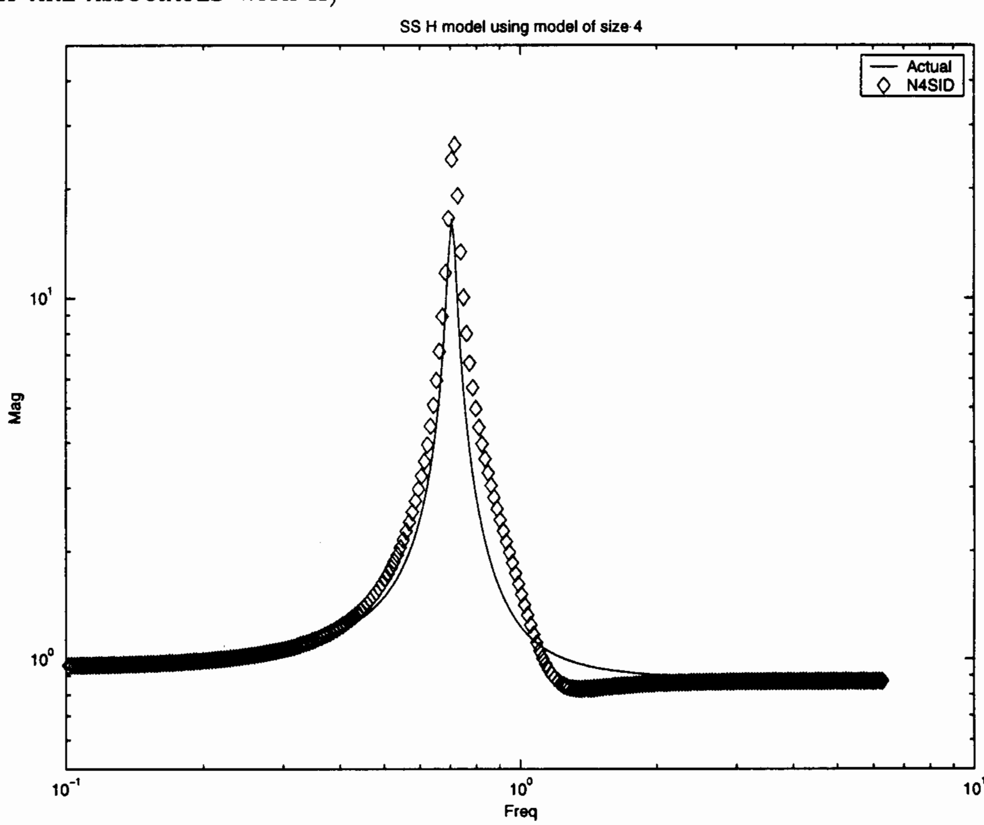


Figure 4: ESTIMATE AND ACTUAL H

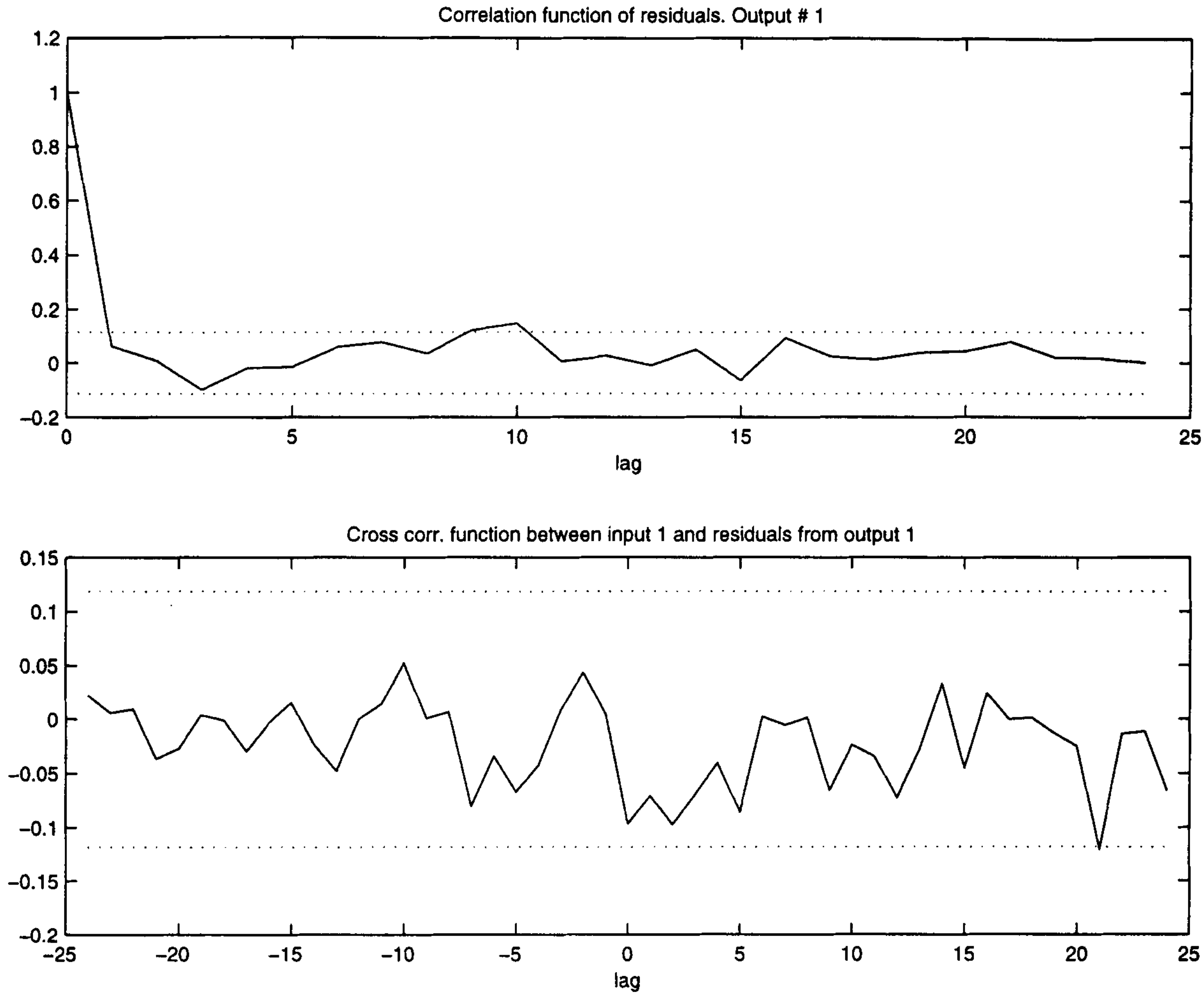
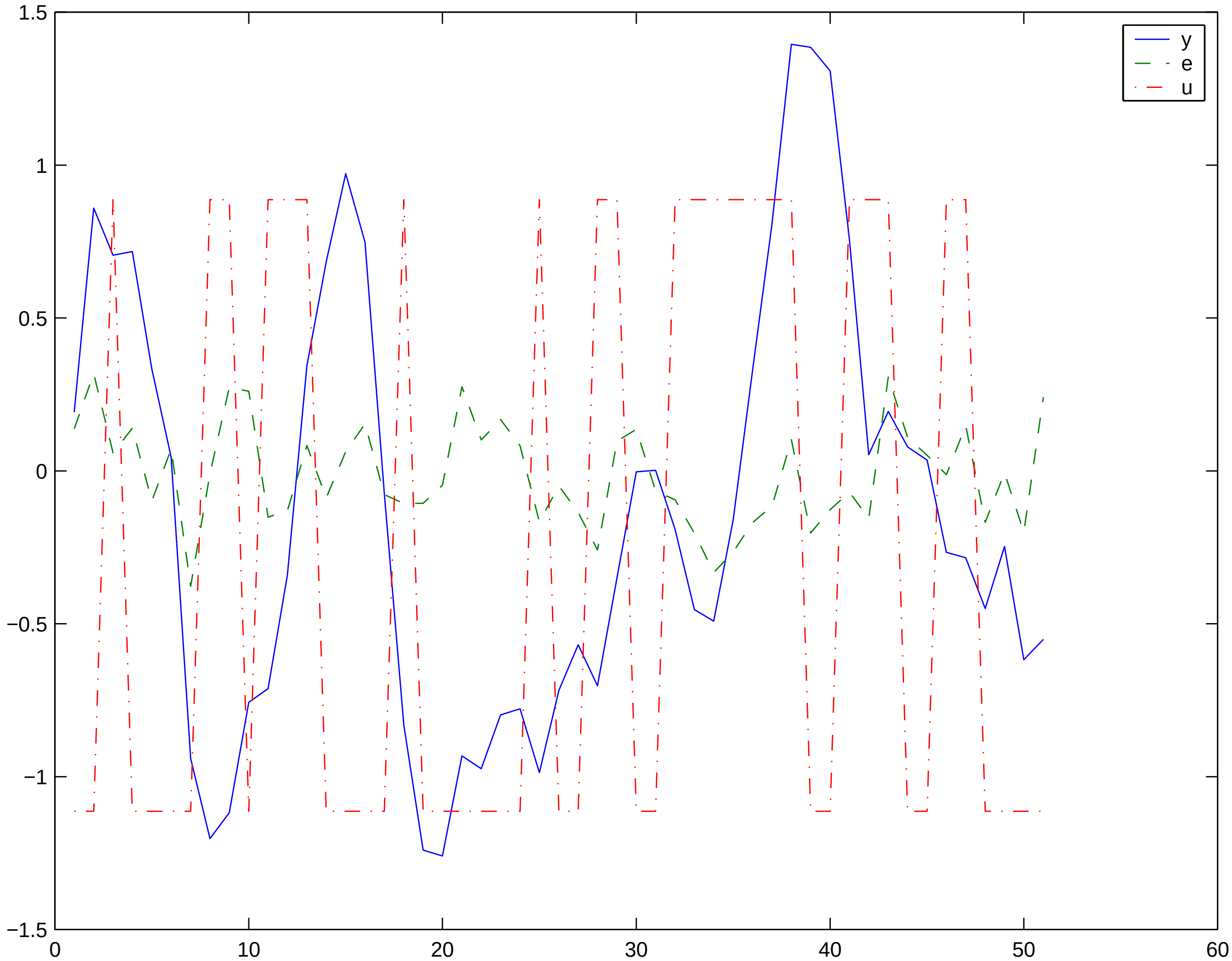


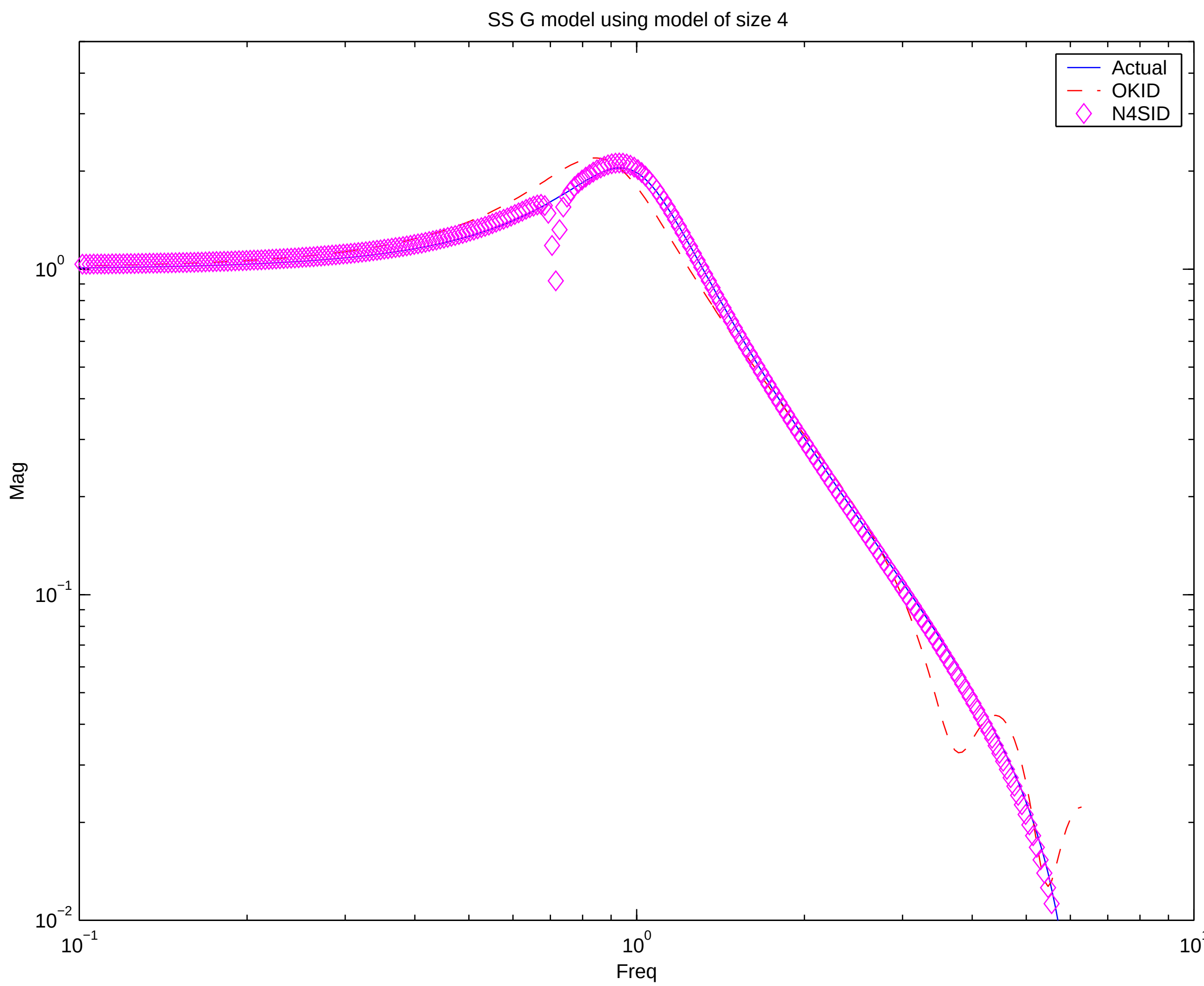
Figure 5: Residuals on a validation set of data

- Reasonable TF fit and residuals are pretty good.

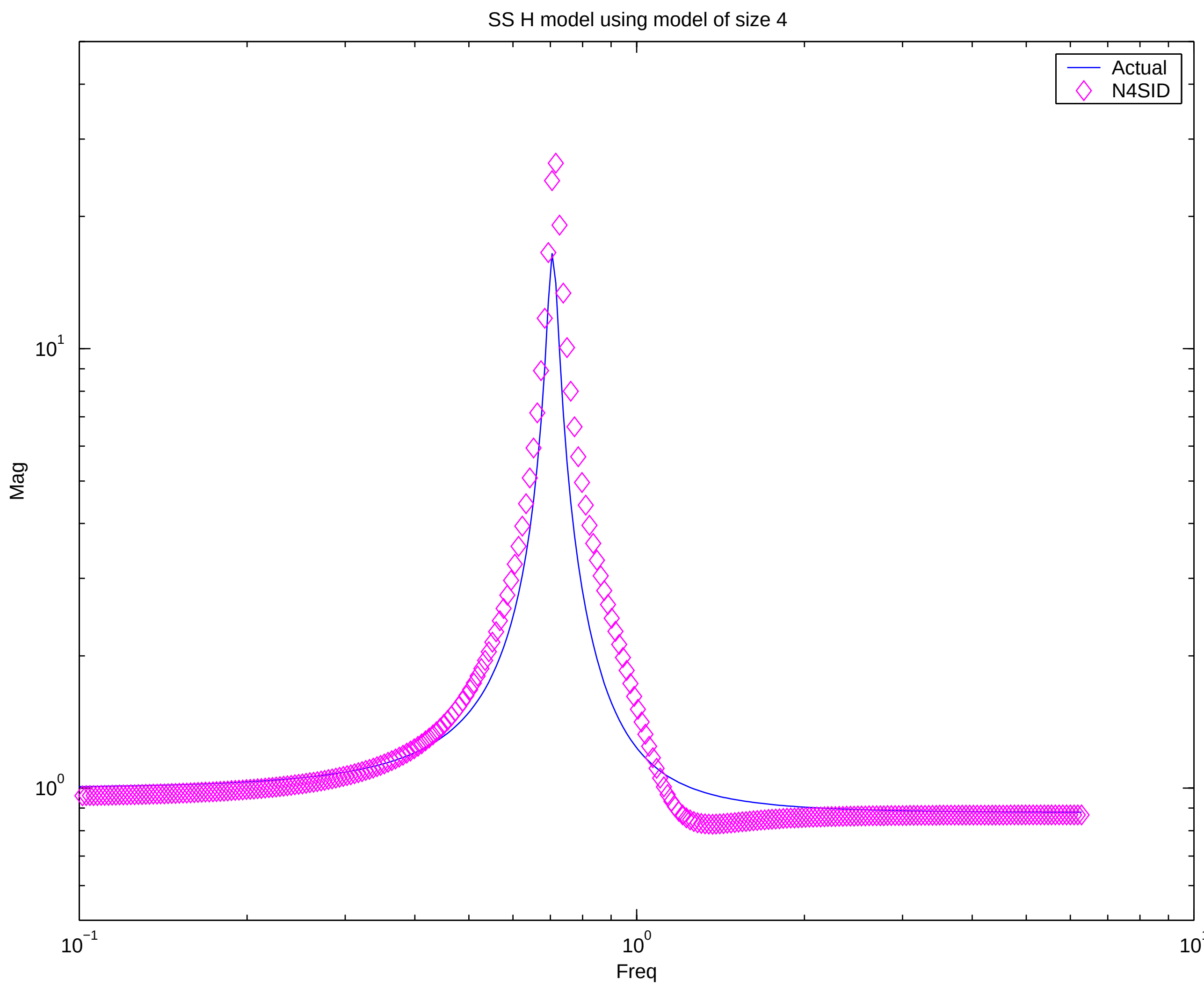
- Great thing is that this approach easily handles MIMO models



1.5
1
0.5
0
−0.5
−1
−1.5
0
10
20
30
40
50
60
y
e
u



SS G model using model of size 4
Actual
OKID
N4SID
Mag
Freq
10^{0}
10^{-1}
10^{-2}
10^{-1}
10^{0}
10^{1}


SS H model using model of size 4
Actual
N4SID
Mag
Freq
10^1
10^0
10^{-1}
10^0
10^1


Correlation function of residuals. Output # 1
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
0
5
10
15
20
25
lag


Cross corr. function between input 1 and residuals from output 1
0.15
0.1
0.05
0
−0.05
−0.1
−0.15
−0.2
−25
−20
−15
−10
−5
0
5
10
15
20
25
lag

Summary

- What this indicates is that the state space methods are very good ways of getting initial models
  - few user inputs required
  - simpler calculations (no local minima)
  - easily handle MIMO systems.

- Problems with the state space methods is that there are few *knobs*
  - can get a good model, but how get a great one?

- Suggest that you use the state space methods as a starting point for the Box-Jenkins (PEM) optimizations.

```
\begin{verbatim}
%
% E211 System ID
% Jonathan How
% Fall 1999
% Use the N4SID algorithm for the robot data
%
clear all;close all;
randn('state',44);
Ny=40;

load hw3_robot_arm
y=z(:,1);
u=z(:,2);
y=dtrend(y);u=dtrend(u);
z=[y u];

fig=0;fig=fig+1;figure(fig);clf
plot([z]);setlines;legend('y','u')

Nest=6;
[Aok,Bok,Cok,Dok,Gok]=okid(size(y,2),size(u,2),Ts,u,y,'batch',(Nest)+2);
%
% [TH,AO]=N4SID(Z,ORDER,NY,AUXORD,DKX,MAXSIZE,TSAMP)
%
[th_ss,AO]=n4sid(z,4:10,1,[],[1 1 1],[],Ts,'trace');
[A_ss,B_ss,C_ss,D_ss,K_ss,X0_ss]=th2ss(th_ss);

Npts=512;
ghat=etfe([y u],[128*4],Npts,Ts);
[wa,ghm,ghp]=getff(ghat,1,1);
%
% models of G
%
[mag1,ph1]=dbode(Aok,Bok,Cok,Dok,Ts,1,wa);
[mag2,ph2]=dbode(A_ss,B_ss,C_ss,D_ss,Ts,1,wa);
fig=fig+1;figure(fig);clf
subplot(211)
hh=loglog(wa,ghm,'b.',wa,mag1,'r--',wa,mag2,'md');
set(hh(1),'MarkerSize',12)
legend('ETFE','OKID','N4SID');
axis([1 150 .005 100])
ylabel('Mag');xlabel('Freq')
title(['OKID model using OK model=',num2str(size(Aok,1)),' SS
model=',num2str(size(A_ss,1))])
subplot(212)
hh=semilogx(wa,ghp,'b.',wa,ph1-360,'r--',wa,ph2-360,'md');
set(hh(1),'MarkerSize',12)
legend('ETFE','OKID','N4SID');
axis([1 150 -540 90])
ylabel('Phase (deg)');xlabel('Freq')
title(['OKID model using OK model=',num2str(size(Aok,1)),' SS
model=',num2str(size(A_ss,1))])

%return
figure(2);print -dpsc robot.ps
```

E211

LECTURE #7

- SOME CLARIFICATIONS
- MODEL QUALITY
  - BIAS, VARIANCE
- SOME ANALYTIC METHODS TO STUDY BIAS

LL 7.1, 7.2 (FIRST HALF), 7.3 (UPTO 206)
BITS OF 8 (SEE PAGE REFS)

CLARIFICATION #1

- NOTE THAT THERE IS A VERY WELL DEFINED FREQUENCY VECTOR THAT YOU <u>MUST</u> USE WHEN PLOTTING THE OUTPUTS FROM FFT, ETFE, SPA ... (MANUAL 4-32)

- DEFAULT IN ETFE IS TO USE N=128

  ⇒ $G(e^{jw})$ IS THEN ESTIMATED AT THE SPECIFIC FREQUENCIES

$$w = \frac{[1:N]}{N} \frac{\pi}{T}$$

(TRY "TYPE ETFE" IN MATLAB - IT IS THE THIRD LAST LINE IN THE PROGRAM)

⇒ YOU HAVE TO USE THIS FREQUENCY VECTOR TO GET BODE PLOTS THAT ARE USEFUL.

- CAN DO BODEPLOT(ETFE(Z))

  OR

```
g = ETFE(z)
[w,m,p] = GETFF(g,1,1)
```

SEE CODE FOR LECTURE 4

↳ CORRECT w TO USE FOR PLOTTING

CLARIFICATION #2

- MODEL DESCRIPTIONS – ON 3-10 HE DEFINES "ARX 221" THE FOLLOWING:

$$G = q^{-n_k} \frac{B(q)}{A(q)} \qquad A(q) = 1 + a_1 q^{-1} + \dots + a_{n_a} q^{-n_a}$$

$$B(q) = b_1 + b_2 q^{-1} + \dots + b_{n_b} q^{-n_b+1}$$

NOT THE SAME AS THE BOOK OR THE NOTES, BUT CLOSE.

- GET THE SAME FORM WE ASSUMED PROVIDED $n_k \geq 1$ AND $(-n_b+1)$ TERM UNDERSTOOD.

"221" → $n_a = n_b = 2$, $n_k = 1$

$$q^{-n_k} B(q) = q^{-1}(b_1 + b_2 q^{-1}) = b_1 q^{-1} + b_2 q^{-2}$$

THIS WAS WHAT WE ASSUMED THE $B(q)$ WOULD LOOK LIKE. (THEN ADDED MORE DELAYS IF NEEDED)

⇒ IN THE TOOLBOX, ALWAYS USE $n_k \geq 1$