ESTIMATION, FILTERING AND SYSTEM IDENTIFICATION - FORMULARY

Sample mean estimator: $\hat{\theta}_N = \frac{1}{N}\sum\limits_{i=1}^{N} d_i$; **Weighted sample mean estimator:** $\hat{\theta}_N = \sum\limits_{i=1}^{N} \alpha_i d_i$, $\alpha_i = \frac{\alpha}{Var\left[d_i\right]}$, $\alpha = \left[\sum_{i=1}^{N} \frac{1}{Var\left[d_i\right]}\right]^{-1}$

Least Squares (LS) estimator for a linear regression problem $y(t)=\varphi(t)^T\theta$, $t=1,\ldots,N$, with $\varphi(t) = \begin{bmatrix} u_1(t) \\ \vdots \\ u_n(t) \end{bmatrix} \in \mathbb{R}^n$, $\theta = \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_n \end{bmatrix} \in \mathbb{R}^n$:

$$\hat{\boldsymbol{\theta}}_{\mathbf{LS}} = \left[\sum_{t=1}^{N} \varphi(t)\,\varphi(t)^T\right]^{-1} \left[\sum_{t=1}^{N} \varphi(t)\,y(t)\right] = \left[\boldsymbol{\Phi}^T\boldsymbol{\Phi}\right]^{\mathbf{-1}} \boldsymbol{\Phi}^T\mathbf{y},\ \Phi = \begin{bmatrix} \varphi(1)^T \\ \vdots \\ \varphi(N)^T \end{bmatrix} = \begin{bmatrix} u_1(1) & \ldots & u_n(1) \\ \vdots & & \vdots \\ u_1(N) & \ldots & u_n(N) \end{bmatrix} \in \mathbb{R}^{N\times n},\ y = \begin{bmatrix} y(1) \\ \vdots \\ y(N) \end{bmatrix} \in \mathbb{R}^N$$

identifiability condition: $\Phi^T\Phi$ is nonsingular

Weighted LS (WLS) estimator: $\hat{\boldsymbol{\theta}}_{\mathbf{WLS}} = \left[\boldsymbol{\Phi}^T\mathbf{Q}\boldsymbol{\Phi}\right]^{\mathbf{-1}}\boldsymbol{\Phi}^T\mathbf{Q}\mathbf{y}$, $Q=\text{diag}(q(t))\in\mathbb{R}^{N\times N}$; $Var[\hat{\theta}_{WLS}] = \left[\Phi^T Q\Phi\right]^{-1}\Phi^T Q\Sigma_v Q\Phi\left[\Phi^T Q\Phi\right]^{-1}$

Gauss-Markov (GM) estimator: $\hat{\boldsymbol{\theta}}_{\mathbf{GM}} = \left[\boldsymbol{\Phi}^T\boldsymbol{\Sigma}_v^{-1}\boldsymbol{\Phi}\right]^{\mathbf{-1}}\boldsymbol{\Phi}^T\boldsymbol{\Sigma}_v^{-1}\mathbf{y}$, $Var[\hat{\theta}_{GM}] = \left[\Phi^T\Sigma_v^{-1}\Phi\right]^{-1}$; if $\Sigma_v = \sigma_v^2 I_N \Rightarrow \hat{\theta}_{GM} = \hat{\theta}_{LS}$

Maximum Likelihood (ML) estimator $\hat{\theta}_{ML} = \arg\max_{\theta\in\mathbb{R}^n} L(\theta)$, $L(\theta) = \left. f(q,\theta)\right|_{q=\delta}$ = likelihood function

$\hat{\theta}_{GM}$ is unbiased ($E(\hat{\theta}_{GM}) = \theta_o$), efficient ($\Sigma_{\hat{\theta}_{GM}} \leq \Sigma_{\hat{\theta}}$, $\forall\hat{\theta}$), consistent ($\lim_{N\to\infty}\Sigma_{\hat{\theta}_{GM}} = 0$), Gaussian

Bayesian (or conditional mean) estimator: $\hat{\boldsymbol{\theta}} = \mathbf{E}\left[\boldsymbol{\theta}\right|\mathbf{d}]$

- Gaussian scalar case $\begin{bmatrix} d \\ \theta \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \Sigma = Var\begin{bmatrix} d \\ \theta \end{bmatrix} = \begin{bmatrix} \sigma_{dd} & \sigma_{d\theta} \\ \sigma_{\theta d} & \sigma_{\theta\theta} \end{bmatrix}\right) \Rightarrow \hat{\theta} = \frac{\boldsymbol{\sigma}_{\theta d}}{\boldsymbol{\sigma}_{dd}}d$, with $Var[\hat{\theta}] = \frac{\sigma_{\theta d}^2}{\sigma_{dd}}$, $Var[\theta-\hat{\theta}] = \sigma_{\theta\theta} - \frac{\sigma_{\theta d}^2}{\sigma_{dd}} = \sigma^2$
- optimal linear estimator for the scalar case $\begin{bmatrix} d \\ \theta \end{bmatrix} \sim \left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \Sigma = Var\begin{bmatrix} d \\ \theta \end{bmatrix} = \begin{bmatrix} \sigma_{dd} & \sigma_{d\theta} \\ \sigma_{\theta d} & \sigma_{\theta\theta} \end{bmatrix}\right)$: $\hat{\theta} = \frac{\sigma_{\theta d}}{\sigma_{dd}}d$
- if $\begin{bmatrix} d \\ \theta \end{bmatrix} \sim \left(\begin{bmatrix} E[d]=\bar{d}\in\mathbb{R}^N \\ E[\theta]=\bar{\theta}\in\mathbb{R}^n \end{bmatrix}, \Sigma = Var\begin{bmatrix} d \\ \theta \end{bmatrix} = \begin{bmatrix} \Sigma_{dd} & \Sigma_{d\theta} \\ \Sigma_{\theta d} & \Sigma_{\theta\theta} \end{bmatrix}\right) \Rightarrow \hat{\theta} = \bar{\theta} + \Sigma_{\theta d}\Sigma_{dd}^{-1}\left(d-\bar{d}\right)$, with $Var[\theta-\hat{\theta}] = \Sigma_{\theta\theta} - \Sigma_{\theta d}\Sigma_{dd}^{-1}\Sigma_{d\theta}$
- $\hat{\theta}^{prior} = E\left[\theta\right] = \bar{\theta}$, with $Var[\hat{\theta}^{prior}] = \Sigma_{\theta\theta}$; $\hat{\theta}^{posterior} = \hat{\theta}^{prior} + \frac{\sigma_{\theta d}}{\sigma_{dd}}\left(d-\bar{d}\right)$, with $Var[\theta - \hat{\theta}^{posterior}] = \sigma_{\theta\theta}\left(1-\rho^2\right)$, $\rho = \frac{\sigma_{\theta d}}{\sqrt{\sigma_{\theta\theta}\sigma_{dd}}}$

Recursive Bayesian estimator:

- scalar case $\begin{bmatrix} \theta \\ d(1) \\ d(2) \end{bmatrix} \sim \left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \Sigma = Var\begin{bmatrix} \theta \\ d(1) \\ d(2) \end{bmatrix} = \begin{bmatrix} \sigma_{\theta\theta} & \sigma_{\theta 1} & \sigma_{\theta 2} \\ \sigma_{1\theta} & \sigma_{11} & \sigma_{12} \\ \sigma_{2\theta} & \sigma_{21} & \sigma_{22} \end{bmatrix}\right)$, $\begin{cases} \sigma_{\theta 1} = \sigma_{1\theta} \\ \sigma_{\theta 2} = \sigma_{2\theta} \\ \sigma_{12} = \sigma_{21} \end{cases} \Rightarrow E[\theta|\,d(1),d(2)] = E[\theta|\,d(1)] + E[\theta|\,e]$,

  where e = innovation of $d(2)$ given $d(1) = d(2) - E\left[d(2)|\,d(1)\right] = d(2) - \frac{\sigma_{21}}{\sigma_{11}}d(1)$
- vector case $\begin{bmatrix} \theta \\ d(1) \\ d(2) \end{bmatrix} \sim \left(\begin{bmatrix} \bar{\theta} \\ \overline{d}(1) \\ \overline{d}(2) \end{bmatrix}, \Sigma = Var\begin{bmatrix} \theta \\ d(1) \\ d(2) \end{bmatrix} = \begin{bmatrix} \Sigma_{\theta\theta} & \Sigma_{\theta 1} & \Sigma_{\theta 2} \\ \Sigma_{1\theta} & \Sigma_{11} & \Sigma_{12} \\ \Sigma_{2\theta} & \Sigma_{21} & \Sigma_{22} \end{bmatrix}\right)$, $\begin{cases} \Sigma_{\theta 1} = \Sigma_{1\theta}^T \\ \Sigma_{\theta 2} = \Sigma_{2\theta}^T \\ \Sigma_{12} = \Sigma_{21}^T \end{cases} \Rightarrow E[\theta|\,d(1),d(2)] = E[\theta|\,d(1)] + E[\theta|\,e] - \bar{\theta}$,

  where e = innovation of $d(2)$ given $d(1) = d(2) - \overline{d}(2) - E[d(2) - \overline{d}(2)\,|\,d(1) - \overline{d}(1)] = d(2) - \overline{d}(2) - \Sigma_{21}\Sigma_{11}^{-1}[d(1) - \overline{d}(1)]$

Set Membership estimation framework: $y = \Phi\cdot\theta_o + e$, $e\in\mathcal{B}_e$, using $\hat{\theta} = \left[\hat{\theta}_j\right] = Ay$, $A = [a_{jk}] = (\Phi^T\Phi)^{-1}\Phi^T$ (i.e., $\hat{\theta} = \hat{\theta}^{LS}$)

- if $\mathcal{B}_e = \mathcal{B}_e^{\infty} = \left\{\tilde{e}\in\mathbb{R}^N\colon |\tilde{e}_i| \leq \varepsilon,\ i=1,\ldots,N\right\} = \left\{\tilde{e}\in\mathbb{R}^N\colon \|\tilde{e}\|_{\infty} = \max_{i=1,\ldots,N}|\tilde{e}_i| \leq \varepsilon\right\} \Rightarrow$

  $MUS^{\infty} = y\oplus\mathcal{B}_e^{\infty} = \left\{\tilde{y}\in\mathbb{R}^N : |\tilde{y}_i - y_i| \leq \varepsilon,\ i=1,\ldots,N\right\} \subset \mathbb{R}^N$, $EUS^{\infty} = A[MUS^{\infty}] = \text{conv}\left\{A\bar{y}_k, k=1,\ldots,2^N\right\} \subset \mathbb{R}^n$

  $EUI_j^{\infty} = \left[\min_{\theta\in EUS^{\infty}}\theta_j, \max_{\theta\in EUS^{\infty}}\theta_j\right] = \left[\hat{\theta}_j^m, \hat{\theta}_j^M\right] \subset \mathbb{R}, j=1,\ldots,n$, where $\hat{\theta}_j^m = \sum_{k=1}^{N} a_{jk}\left[y_k - \varepsilon\cdot\text{sign}\left(a_{jk}\right)\right]$, $\hat{\theta}_j^M = 2\hat{\theta}_j - \hat{\theta}_j^m$

  $FPS^{\infty} = \left\{\tilde{\theta}\in\mathbb{R}^n : \left|y_i - \varphi_i^T\tilde{\theta}\right| \leq \varepsilon,\ i=1,\ldots,N\right\}$, $PUI_j^{\infty} = \left[\min_{\theta\in FPS^{\infty}}\theta_j, \max_{\theta\in FPS^{\infty}}\theta_j\right] = \left[\theta_j^m, \theta_j^M\right] \subset \mathbb{R}, j=1,\ldots,n \Rightarrow$

  $\theta_j^m = \min\limits_{F\cdot\theta\leq f} c_j^T\cdot\theta$, $\theta_j^M = \max\limits_{F\cdot\theta\leq f} c_j^T\cdot\theta = -\min\limits_{F\cdot\theta\leq f} -c_j^T\cdot\theta$, with c_j = j-th column of $I_{n\times n}$, $F = \begin{bmatrix} \Phi \\ -\Phi \end{bmatrix}$, $f = \begin{bmatrix} y \\ -y \end{bmatrix} + \varepsilon$

  Under MATLAB R2014A: `PUI_j_min = c_j'*lp(c_j, F, f); PUI_j_max = c_j'*lp(-c_j, F, f); % with lp`
  `options_old = optimset('linprog'); options_new = optimset(options_old, 'Algorithm', 'simplex'); % to use linprog`
  `PUI_j_min = c_j'*linprog(c_j, F, f, [], [], [], [], [], options_new); PUI_j_max = c_j'*linprog(-c_j, F, f, [], [], [], [], [], options_new);`
- if $\mathcal{B}_e = \mathcal{B}_e^2 = \left\{\tilde{e}\in\mathbb{R}^N\colon \tilde{e}^T\cdot\tilde{e} = \sum_{i=1}^{N}\tilde{e}_i^2 \leq \varepsilon^2\right\} = \left\{\tilde{e}\in\mathbb{R}^N\colon \|\tilde{e}\|_2 = \sqrt{\sum_{i=1}^{N}\tilde{e}_i^2} \leq \varepsilon\right\} \Rightarrow$

  $MUS^{\,2} = y\oplus\mathcal{B}_e^2 = \left\{\tilde{y}\in\mathbb{R}^N : (\tilde{y}-y)^T\cdot(\tilde{y}-y) \leq \varepsilon^2\right\} \subset \mathbb{R}^N$, $EUS^2 = A\left[MUS^2\right] = \left\{\tilde{\theta}\in\mathbb{R}^n : (\tilde{\theta}-\hat{\theta})^T\Phi^T\Phi\,(\tilde{\theta}-\hat{\theta}) \leq \varepsilon^2\right\} \subset \mathbb{R}^n$

  $EUI_j^2 = \left[\min_{\theta\in EUS^2}\theta_j, \max_{\theta\in EUS^2}\theta_j\right] = \left[\hat{\theta}_j^m, \hat{\theta}_j^M\right] = \left[\hat{\theta}_j - \varepsilon\cdot\sigma_j, \hat{\theta}_j + \varepsilon\cdot\sigma_j\right] \subset \mathbb{R}, j=1,\ldots,n$, with $\sigma_j = \sqrt{\left[(\Phi^T\Phi)^{-1}\right]_{j,j}}$

  $FPS^{\,2} = \left\{\tilde{\theta}\in\mathbb{R}^n : (\tilde{\theta}-\hat{\theta})^T\left[\Phi^T\Phi\right](\tilde{\theta}-\hat{\theta}) \leq \varepsilon^2 - \alpha^2\right\}$, with $\alpha^2 = (y-\Phi\hat{\theta})^T(y-\Phi\hat{\theta}) = \|\,y-\Phi\hat{\theta}\,\|_2^2 \leq \varepsilon^2$ (otherwise $FPS^{\,2} = \emptyset$)

  $PUI_j^2 = \left[\min_{\theta\in FPS^2}\theta_j, \max_{\theta\in FPS^2}\theta_j\right] = \left[\theta_j^m, \theta_j^M\right] = \left[\hat{\theta}_j - \sigma_j\sqrt{\varepsilon^2-\alpha^2}, \hat{\theta}_j + \sigma_j\sqrt{\varepsilon^2-\alpha^2}\right] \subset \mathbb{R}, j=1,\ldots,n$, with $\sigma_j = \sqrt{\left[(\Phi^T\Phi)^{-1}\right]_{j,j}}$

Central estimate: $\hat{\theta}^C = \left[\hat{\theta}_j^C\right]$, $j=1,\ldots,n$, with $\hat{\theta}_j^C = (\theta_j^m + \theta_j^M)/\,2$ ($\hat{\theta}^C$ is optimal for $\mathcal{B}_e^2$ and $\mathcal{B}_e^{\infty}$, while $\hat{\theta}^{LS}$ is optimal only for $\mathcal{B}_e^2$)

Given $\mathcal{S}$: $\begin{cases} x(t+1) = A(t)x(t) + B(t)u(t) + v_1(t) \\ y(t) = C(t)x(t) + v_2(t) \end{cases}$ $t = 1,2,\ldots,$ $\begin{cases} x(t)\in\mathbb{R}^n, y(t)\in\mathbb{R}^q, u(t)\in\mathbb{R}^p, A(t)\in\mathbb{R}^{n\times n}, B(t)\in\mathbb{R}^{n\times p}, C(t)\in\mathbb{R}^{q\times n} \\ v_1(t)\sim WN(0,V_1(t))\in\mathbb{R}^n, v_2(t)\sim WN(0,V_2(t))\in\mathbb{R}^q, x(t=1)\sim(\bar{x}_1,P_1) \\ V_1(t)\in\mathbb{R}^{n\times n}, V_2(t)\in\mathbb{R}^{q\times q}, V_{12}(t)\in\mathbb{R}^{n\times q}, \bar{x}_1\in\mathbb{R}^n, P_1\in\mathbb{R}^{n\times n} \end{cases}$

Dynamic Kalman one-step predictor and filter:

$$\begin{cases} \hat{x}(N+1|N) = A(N)\hat{x}(N|N-1) + B(N)u(N) + K(N)e(N) \\ \hat{y}(N|N-1) = C(N)\hat{x}(N|N-1) \\ \hat{x}(N|N) = \hat{x}(N|N-1) + K_0(N)e(N) \\ \hat{y}(N|N) = C(N)\hat{x}(N|N) \\ e(N) = y(N) - \hat{y}(N|N-1) \end{cases},\ \begin{cases} K_0(N) = P(N)C(N)^T\left[C(N)P(N)C(N)^T + V_2(N)\right]^{-1} \in\mathbb{R}^{n\times q} \\ K(N) = \left[A(N)P(N)C(N)^T + V_{12}(N)\right]\left[C(N)P(N)C(N)^T + V_2(N)\right]^{-1} \\ P(N+1) = A(N)P(N)A(N)^T + V_1(N) + \\ \qquad -K(N)\left[C(N)P(N)C(N)^T + V_2(N)\right]K(N)^T \in\mathbb{R}^{n\times n}\ \text{(DRE)} \end{cases}$$

Initialization: $\hat{x}(1|0) = E[x(1)] = \bar{x}_1$, $P(1) = E[(x(1)-\bar{x}_1)(x(1)-\bar{x}_1)^T] = P_1$

Predictor/corrector form of dynamic predictor and filter for a LTI system $\mathcal{S}$ (with $V_{12} = 0_{n\times q}, V_2 > 0, K(N) = AK_0(N)$)**:**

$$\begin{cases} K_0(N) = P(N)C^T\left[CP(N)C^T + V_2\right]^{-1} \\ P_0(N) = \left[I_n - K_0(N)C\right]P(N)\left[I_n - K_0(N)C\right]^T + K_0(N)V_2K_0(N)^T = \left[I_n - K_0(N)C\right]P(N) \\ e(N) = y(N) - C\hat{x}(N|N-1) \\ \hat{x}(N|N) = \hat{x}(N|N-1) + K_0(N)e(N) \\ P(N+1) = AP_0(N)A^T + V_1 \\ \hat{x}(N+1|N) = A\hat{x}(N|N) + Bu(N) \end{cases}$$

Steady-state Kalman one-step predictor and filter for a LTI system $\mathcal{S}$ (with $V_{12}=0_{n\times q}, V_2>0$)**:**

$$\begin{cases} \hat{x}(N+1|N) = A\hat{x}(N|N-1)+Bu(N)+\bar{K}e(N) \\ \hat{y}(N|N-1) = C\hat{x}(N|N-1) \\ \hat{x}(N|N) = \hat{x}(N|N-1)+\bar{K}_0e(N) \\ \hat{y}(N|N) = C\hat{x}(N|N) \\ e(N) = y(N)-\hat{y}(N|N-1) \end{cases} \text{, with } \begin{cases} \bar{K}_0 = \bar{P}C^T\left[C\bar{P}C^T+V_2\right]^{-1} \in \mathbb{R}^{n\times q} \\ \bar{K} = A\bar{P}C^T(C\bar{P}C^T+V_2)^{-1} = A\bar{K}_0 \in \mathbb{R}^{n\times q} \\ \bar{P} \in \mathbb{R}^{n\times n} : \bar{P} = A\bar{P}A^T+V_1-A\bar{P}C^T(C\bar{P}C^T+V_2)^{-1}C\bar{P}A^T \qquad \text{(ARE)} \end{cases}$$

Under MATLAB: `S=ss(A,[B,eye(n)],C,[D,zeros(1,n)],1); [Kalman_predictor,Kbar,Pbar,K0bar]=kalman(S,V1,V2,0);`

ARX(n_a,n_b,n_k) : $y(t)+a_1y(t-1)+a_2y(t-2)+\cdots+a_{n_a}y(t-n_a) = b_1u(t-n_k)+\cdots+b_{n_b}u(t-n_k-n_b+1)+e(t)$

$y(t)=\dfrac{B(z)}{A(z)}u(t)+\dfrac{1}{A(z)}e(t)$, with $A(z)=1+a_1z^{-1}+a_2z^{-2}+\cdots+a_{n_a}z^{-n_a}$, $B(z)=b_1z^{-n_k}+b_2z^{-n_k-1}+\cdots+b_{n_b}z^{-n_k-n_b+1}$

ARX model $\hat{\mathcal{M}}(\theta)$ in predictor form: $\hat{y}(t)=[1-A(z)]\,y(t)+B(z)u(t)=\left(-a_1z^{-1}-\cdots-a_{n_a}z^{-n_a}\right)y(t)+\left(b_1z^{-n_k}+\cdots+b_{n_b}z^{-n_k-n_b+1}\right)u(t)$
$=\varphi(t)^T\theta$, with $\varphi(t)=[-y(t-1)\ \cdots\ -y(t-n_a)\ u(t-n_k)\ \cdots\ u(t-n_k-n_b+1)]^T\in\mathbb{R}^{n_a+n_b}$, $\theta=[a_1\cdots a_{n_a}\ b_1\cdots b_{n_b}]^T\in\mathbb{R}^{n_a+n_b}$

ARMAX(n_a,n_b,n_c,n_k) : $y(t)+a_1y(t-1)+a_2y(t-2)+\cdots+a_{n_a}y(t-n_a)=b_1u(t-n_k)+\cdots+b_{n_b}u(t-n_k-n_b+1)+e(t)+c_1e(t-1)+\cdots+c_{n_c}e(t-n_c)$

$y(t)=\dfrac{B(z)}{A(z)}u(t)+\dfrac{C(z)}{A(z)}e(t)$, with $A(z)=1+a_1z^{-1}+\cdots+a_{n_a}z^{-n_a}$, $B(z)=b_1z^{-nk}+\cdots+b_{n_b}z^{-n_k-n_b+1}$, $C(z)=1+c_1z^{-1}+\cdots+c_{n_c}z^{-n_c}$

ARMAX model $\hat{\mathcal{M}}(\theta)$ in predictor form: $\hat{y}(t)=\left[1-\dfrac{A(z)}{C(z)}\right]y(t)+\dfrac{B(z)}{C(z)}u(t)$

OE(n_b,n_f,n_k) : $y(t)=w(t)+e(t)$, where: $w(t)+f_1w(t-1)+\cdots+f_{n_f}w(t-n_f)=b_1u(t-n_k)+\cdots+b_{n_b}u(t-n_k-n_b+1)$

$y(t)=\dfrac{B(z)}{F(z)}u(t)+e(t)$, with $B(z)=b_1z^{-n_k}+b_2z^{-n_k-1}+\cdots+b_{n_b}z^{-n_k-n_b+1}$, $F(z)=1+f_1z^{-1}+f_2z^{-2}+\cdots+f_{n_f}z^{-n_f}$

OE model $\hat{\mathcal{M}}(\theta)$ in predictor form: $\hat{y}(t)=\dfrac{B(z)}{F(z)}u(t)$

RLS-1 algo: $S(t)=S(t-1)+\varphi(t)\,\varphi(t)^T$, $K(t)=S(t)^{-1}\varphi(t)$, $\varepsilon(t)=y(t)-\varphi(t)^T\,\hat{\theta}_{t-1}$, $\hat{\theta}_t=\hat{\theta}_{t-1}+K(t)\varepsilon(t)$

RLS-2 algo: $R(t)=\dfrac{t-1}{t}R(t-1)+\dfrac{1}{t}\varphi(t)\,\varphi(t)^T$, $K(t)=\dfrac{1}{t}R(t)^{-1}\varphi(t)$, $\varepsilon(t)=y(t)-\varphi(t)^T\,\hat{\theta}_{t-1}$, $\hat{\theta}_t=\hat{\theta}_{t-1}+K(t)\varepsilon(t)$

RLS-3 algo: $\beta_{t-1}=1+\varphi(t)^TV(t-1)\varphi(t)$, $V(t)=V(t-1)-\beta_{t-1}^{-1}V(t-1)\varphi(t)\varphi(t)^TV(t-1)$, $K(t)=V(t)\varphi(t)$, $\varepsilon(t)=y(t)-\varphi(t)^T\hat{\theta}_{t-1}$,
$\hat{\theta}_t=\hat{\theta}_{t-1}+K(t)\varepsilon(t)$, with initialization: $\hat{\theta}_1=0_{n\times1}$, $V(1)=S(1)^{-1}=R(1)^{-1}=\alpha I_n$, $\alpha>0$, $n=\dim(\theta)$

Given a model $\hat{\mathcal{M}}(\theta)$ with complexity n (given by $\dim(\theta)$ or the degree of the model t.f., according to the problem context) $\Rightarrow$

$$\textbf{MSE}=\frac{1}{N-N_0}\textstyle\sum_{t=N_0+1}^{N}\varepsilon(t)^2=\frac{1}{N-N_0}\sum_{t=N_0+1}^{N}\left(y(t)-\hat{y}(t,\theta)\right)^2 \Rightarrow \textbf{RMSE}=\sqrt{MSE};$$

$$\textbf{FPE}=\frac{N-N_0+n}{N-N_0-n}\cdot MSE;\ \textbf{AIC}=n\frac{2}{N-N_0}+\ln(MSE);\ \textbf{MDL}=n\frac{\ln(N-N_0)}{N-N_0}+\ln(MSE);$$

$$\textbf{Best Fit}=1-\sqrt{\frac{MSE}{\frac{1}{N-N_0}\sum_{t=N_0+1}^{N}(y(t)-\bar{y})^2}},\quad \text{with } \bar{y}=\frac{1}{N-N_0}\sum_{t=N_0+1}^{N}y(t)$$

For a neural network with n inputs, r hyperbolic tangent neurons only in the unique hidden layer and just one linear output neuron:

$$\hat{y}(t)=\sum_{i=1}^{r}\left[W_2\right]_i\tanh\left(\textstyle\sum_{j=1}^{n}\left[W_1\right]_{i,j}\varphi_j\left(t\right)+\left[W_1\right]_{i,n+1}\right)+\left[W_2\right]_{r+1},\quad \text{where}$$

- for a **NNARX**(n_a,n_b,n_k) : $\varphi(t)=[y(t-1)\ \cdots\ y(t-n_a)\ u(t-n_k)\ \cdots\ u(t-n_k-n_b+1)]^T\in\mathbb{R}^{n_a+n_b}$, $n=n_a+n_b$
- for a **NNOE**(n_b,n_f,n_k) : $\varphi(t)=[\hat{y}(t-1)\ \cdots\ \hat{y}(t-n_f)\ u(t-n_k)\ \cdots\ u(t-n_k-n_b+1)]^T\in\mathbb{R}^{n_f+n_b}$, $n=n_f+n_b$

Under MATLAB: $\texttt{NetDef}=[\texttt{'}\underbrace{\texttt{H}\cdots\texttt{H}}_{r\text{ times}}\texttt{'};\texttt{'L}\underbrace{-\cdots-}_{r-1\text{ times}}\texttt{'}]$; `trparms = settrain; trparms = settrain(trparms,'maxiter',500);`

`[W1,W2] = nnarx(NetDef,[na,nb,nk],[],[],trparms,y,u); % or: [W1,W2] = nnoe(NetDef,[nb,nf,nk],...`

Free space for other formulas or schemes (<u>NO</u> MATLAB code!); **since A.A. 2015/16, this space has to remain <u>free</u>**