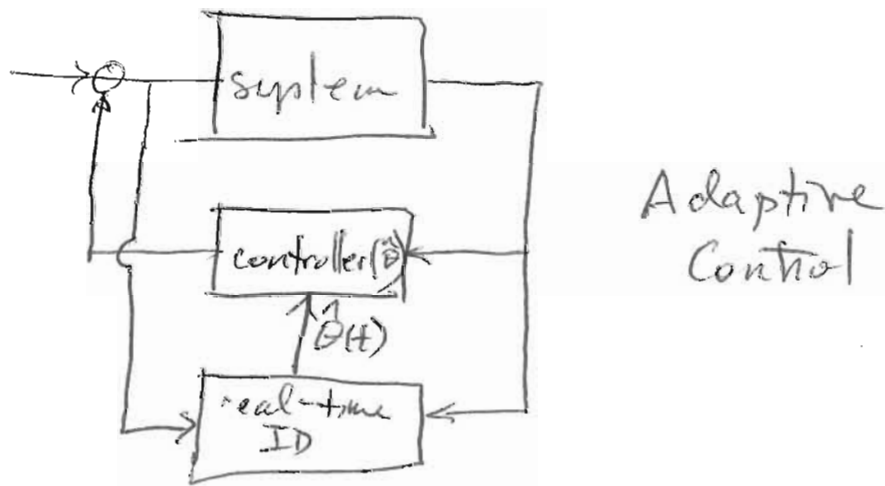


# Recursive Identification

(Ljung, Ch. 11)

It is often desirable to have an identification algorithm that can be implemented in real-time. The parameter estimate  $\hat{\Theta}(t)$  is then time-varying, presumably improves with time, and can be used for, e.g., a parameterized controller:



Necessary properties of a real-time identification algorithm:

- (1) Bounded computational requirements that can be met between samples (within  $st$ ).
- (2) Bounded identifier state that can fit within available memory
- (3) Competitive performance to what is achievable off-line (where "competitive" depends on the application).

General System Identification Method, where the model is ~~defined~~ <sup>determined</sup> by a parameter vector:

$$\hat{\theta}_t = F(t, z^t)$$

↑  
Size of  $z$  increases as  $t \uparrow$   
Computational complexity increases as  $t \uparrow$ .

Introduce "identifier state":

$$\underline{x}(t) = \underline{H}(t, \underline{x}(t-1), y(t), u(t))$$

$$\hat{\theta}_t = \underline{h}(\underline{x}(t))$$

$\underline{x}(t)$  — state information carried from one time step to the next

$\underline{H}$  — function mapping state + most recent measurements to next/new state

$\underline{h}$  — function mapping state to parameters.

If each measurement represents a small increment in known information about the system, an incremental form of the identifier is:

$$\hat{\underline{\theta}}_t = \hat{\underline{\theta}}_{t-1} + \gamma_t \underline{\varphi}_0(\underline{x}(t), y(t), u(t))$$

$$\underline{x}(t) = \underline{x}(t-1) + \mu_t \underline{\varphi}_x(\underline{x}(t-1), y(t), u(t))$$

—

## Recursive Least Squares

L.S. estimate minimizes the weighted LS criterion:

$$\hat{\underline{\theta}}_t = \arg \min_{\underline{\theta}} \sum_{k=1}^t \beta(t, k) [y(k) - \underline{\varphi}^T(k) \underline{\theta}]^2$$

$\hat{\underline{\theta}}_t$  is a solution to

$$\bar{R}(t) \hat{\underline{\theta}}_t = \underline{f}(t)$$

with

$$\bar{R}(t) = \sum_{k=1}^t \beta(t, k) \underline{\varphi}(k) \underline{\varphi}^T(k)$$

$$\underline{f}(t) = \sum_{k=1}^t \beta(t, k) \underline{\varphi}(k) y(k)$$

Suppose

$$\beta(t, k) = \lambda(t) \beta(t-1, k); \quad 0 \leq k < t-1$$

$$\beta(t, t) = 1$$

Exploring these equations, starting at  $t=0$ :

$$\beta(0, 0) = 1$$

$$\beta(1, 0) = \lambda_1 \beta(0, 0) = \lambda_1$$

$$\beta(1, 1) = 1$$

$$\beta(2, 0) = \lambda_2 \beta(1, 0) = \lambda_2 \lambda_1 \beta(0, 0) = \lambda_2 \lambda_1$$

$$\beta(2, 1) = \lambda_2 \beta(1, 1) = \lambda_2$$

$$\beta(2, 2) = 1$$

$$\beta(3, 0) = \lambda_3 \beta(2, 0) = \lambda_3 \lambda_2 \lambda_1$$

$$\beta(3, 1) = \lambda_3 \beta(2, 1) = \lambda_3 \lambda_2$$

$$\beta(3, 2) = \lambda_3 \beta(2, 2) = \lambda_3$$

$$\beta(3, 3) = 1$$

The pattern:

$t$	$k \setminus 0$	1	2	3	4
0	1	—	—	—	—
1	$\lambda_1$	1	—	—	—
2	$\lambda_2 \lambda_1$	$\lambda_2$	1	—	—
3	$\lambda_3 \lambda_2 \lambda_1$	$\lambda_3 \lambda_2$	$\lambda_3$	1	—
4	$\lambda_4 \lambda_3 \lambda_2 \lambda_1$	$\lambda_4 \lambda_3 \lambda_2$	$\lambda_4 \lambda_3$	$\lambda_4$	1

$$\left. \begin{array}{l} \\ \\ \\ \\ \\ \end{array} \right\} \beta(t, k) = \prod_{j=k+1}^t \lambda(j) \quad \text{for } t > k.$$

Since

$$\underline{\bar{R}}(t) = \sum_{k=1}^t \varphi(t,k) \varphi(k) \varphi^T(k)$$

and

$$\beta(t,k) = \lambda(t) \beta(t-1,k) \quad \text{for } 0 \leq k < t-1$$

then

$$\underline{\bar{R}}(t) = \sum_{k=1}^{t-1} \lambda(t) \beta(t-1,k) \varphi(k) \varphi^T(k) + \varphi(t) \varphi^T(t)$$

$$\text{or } \boxed{\underline{\bar{R}}(t) = \lambda(t) \underline{\bar{R}}(t-1) + \varphi(t) \varphi^T(t)} \quad (\star)$$

Similarly,

$$\boxed{\underline{f}(t) = \lambda(t) \underline{f}(t-1) + \varphi(t) y(t)}$$

Looking at the solution,

$$\begin{aligned} \hat{\theta}_t &= \underline{\bar{R}}^{-1}(t) \underline{f}(t) = \underline{\bar{R}}^{-1}(t) [\lambda(t) \underline{f}(t-1) + \varphi(t) y(t)] \\ &= \underline{\bar{R}}^{-1}(t) [\lambda(t) \overbrace{\underline{\bar{R}}^{-1}(t-1) \hat{\theta}_{t-1}}^{\underline{f}(t-1)} + \varphi(t) y(t)] \\ &= \underline{\bar{R}}^{-1}(t) \left[ [\underline{\bar{R}}(t) - \varphi(t) \varphi^T(t)] \hat{\theta}_{t-1} + \varphi(t) y(t) \right] \quad \text{from } (\star) \\ &= \hat{\theta}_{t-1} + \underbrace{\underline{\bar{R}}^{-1}(t) \varphi(t)}_{\text{gain!}} \underbrace{[y(t) - \varphi^T(t) \hat{\theta}_{t-1}]}_{\text{note: innovations process!}} \end{aligned}$$

Summarizing,

$$\begin{aligned}\hat{\theta}_t &= \hat{\theta}_{t-1} + \bar{R}(t)^{-1} \varphi(t) [y(t) - \varphi^T(t) \hat{\theta}_{t-1}] \\ \bar{R}(t) &= \lambda(t) \bar{R}(t-1) + \varphi(t) \varphi^T(t)\end{aligned}$$

is the recursive LS identifier.

Here, the identifier's state  $\underline{\lambda}(t)$  is  $[\bar{R}(t), \hat{\theta}_t]$ .

Note that  $\bar{R}(t)$  is symmetric, so only its upper triangular part is needed in storage.

—

Avoiding inversion of  $\bar{R}(t)$  at each step:

$$\text{Define } \underline{P}(t) = \bar{R}^{-1}(t)$$

Apply the matrix inversion lemma.

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

with  $\underline{A} = \lambda(t) \bar{R}(t-1)$ ;  $\underline{B} = \underline{D}^T = \varphi(t)$ ,  $\underline{C} = 1$

to

$$\underline{\bar{R}}(t) = \lambda(t) \underline{\bar{R}}(t-1) + \varphi(t) \varphi^T(t) :$$

$$\underline{P}(t) = \frac{1}{\lambda(t)} \left[ \underline{P}(t-1) - \frac{\underline{P}(t-1) \varphi(t) \varphi^T(t) \underline{P}(t-1)}{\lambda(t) + \varphi^T(t) \underline{P}(t-1) \varphi(t)} \right]$$

and

$$\underline{\bar{R}}^{-1}(t) \varphi(t) = \frac{\underline{P}(t-1) \varphi(t)}{\lambda(t) + \varphi^T(t) \underline{P}(t-1) \varphi(t)}$$

The Recursive LS ID Algorithm is then :

$$(11.12a-c) \left\{ \begin{array}{l} \underline{\hat{\theta}}(t) = \underline{\hat{\theta}}(t-1) + \underline{L}(t) [y(t) - \varphi^T(t) \underline{\hat{\theta}}(t-1)] \\ \underline{L}(t) = \frac{\underline{P}(t-1) \varphi(t)}{\lambda(t) + \varphi^T(t) \underline{P}(t-1) \varphi(t)} \\ \underline{P}(t) = \frac{1}{\lambda(t)} \left[ \underline{P}(t-1) - \frac{\underline{P}(t-1) \varphi(t) \varphi^T(t) \underline{P}(t-1)}{\lambda(t) + \varphi^T(t) \underline{P}(t-1) \varphi(t)} \right] \end{array} \right.$$

## Normalized Gain Version

Define

$$\underline{R}(t) = \lambda(t) \bar{R}(t) \quad \text{and} \quad \gamma(t) = \left[ \sum_{k=1}^t P(t, k) \right]^{-1}$$

Then

$$\frac{1}{\delta(t)} = \frac{\lambda(t)}{\gamma(t-1)} + 1$$

and

$$\underline{R}(t) = \underline{R}(t-1) + \delta(t) \left[ \varphi(t) \varphi^T(t) - \underline{R}(t-1) \right]$$

then

$$\begin{aligned} \varepsilon(t) &= y(t) - \varphi^T(t) \hat{\theta}(t-1) \\ \hat{\theta}(t) &= \hat{\theta}(t-1) + \delta(t) \underline{R}^{-1}(t) \varphi(t) \varepsilon(t) \\ \underline{R}(t) &= \underline{R}(t-1) + \delta(t) \left[ \varphi(t) \varphi^T(t) - \underline{R}(t-1) \right] \end{aligned}$$

where  $\varepsilon(t)$  is the innovations process. In this version,  $\underline{R}(t)$  is normalized, and  $\delta(t)$  is an update gain.

## Initial Conditions / Start-up :

At  $t=0$ ,  $\bar{P}(0) = \underline{0}$ ,  $\hat{\theta}_0$  is arbitrary, and  $\underline{P}(t)$  is not defined. This doesn't work!

Instead, it is often better to interpret  $\theta_0$  as a random vector with known mean (becoming  $\hat{\theta}_0$ ) and covariance (used for  $\bar{P}(0)$ ). The book provides a couple of other alternatives.

## Multivariable Case

Suppose  $y(k)$  is a vector. Then the weighted LS parameter identification problem minimizes, over  $\theta$ ,

$$\frac{1}{2} \sum_{k=1}^t \beta(t,k) [y(k) - \phi^T(k)\theta]^T \Lambda_k^{-1} [y(k) - \phi^T(k)\theta]$$

with  $\beta(t,k)$  as for the previous case (11.6).

The LS parameter recursive identification is then

$$(11.21) \quad \left\{ \begin{aligned} \hat{\underline{\theta}}(t) &= \hat{\underline{\theta}}(t-1) + \underline{\lambda}(t) [y(t) - \underline{\varphi}^T(t) \hat{\underline{\theta}}(t-1)] \\ \underline{\lambda}(t) &= \underline{P}(t-1) \underline{\varphi}(t) [\lambda(t) \underline{\Lambda}_t + \underline{\varphi}^T(t) \underline{P}(t-1) \underline{\varphi}(t)]^{-1} \\ \underline{P}(t) &= \frac{\underline{P}(t-1) - \underline{P}(t-1) \underline{\varphi}(t) [\lambda(t) \underline{\Lambda}_t + \underline{\varphi}^T(t) \underline{P}(t-1) \underline{\varphi}(t)]^{-1} \underline{\varphi}^T(t) \underline{P}(t-1)}{\lambda(t)} \end{aligned} \right.$$

or

$$(11.22) \quad \left\{ \begin{aligned} \underline{\varepsilon}(t) &= y(t) - \underline{\varphi}^T(t) \hat{\underline{\theta}}(t-1) \\ \hat{\underline{\theta}}(t) &= \hat{\underline{\theta}}(t-1) + \delta(t) \underline{R}^{-1}(t) \underline{\varphi}(t) \underline{\Lambda}_t^{-1} \underline{\varepsilon}(t) \\ \underline{R}(t) &= \underline{R}(t-1) + \delta(t) [\underline{\varphi}(t) \underline{\Lambda}_t^{-1} \underline{\varphi}^T(t) \otimes \underline{R}(t-1)] \end{aligned} \right.$$

### Kalman Filter Interpretation

Eqs. (11.21) is the KF for the system

$$\underline{\theta}(t+1) = \underline{\theta}(t)$$

$$y(t) = \underline{\varphi}^T(t) \underline{\theta}(t) + v(t) \quad ; \quad E[v(t)v^T(t)] = \underline{R}_2(t)$$

with  $\lambda(t) \equiv 1$  and  $\underline{\Lambda}_t = \underline{R}_2(t)$ .

Observations:

- (1) If the measurement noise is white and Gaussian,  $P(\theta(t) | Z^{t-1})$  is Gaussian with mean value  $\hat{\theta}(t)$  and covariance  $P(t)$
- (2) Using the KF to interpret the initial conditions,  $\hat{\theta}(0)$  &  $P(0)$  are the mean and covariance of the prior distribution of  ~~$\theta(0)$~~   $\theta$ .
- (3) The norm weighting matrix  $\underline{\Lambda}_t$  is the measurement (prediction) error noise covariance in the KF.

Forgetting Factors, Recursive Identification Schemes  
the nose Touch with Reality, and Time-Varying  
Systems.

$\lambda(t)$  is a Forgetting Factor. It determines how quickly past data are discounted.

- (1) If past data are not discounted, since there is no process noise ( $\theta(t)$  is assumed constant), the id. gain  $\rightarrow 0$  as  $t \rightarrow \infty$ . The algorithm begins to ~~ignore~~ ignore new data!

(2) One often encounters slowly time varying systems — which can be modeled

by  $\Phi(t) = \Phi(t_0) + w(t)$

or by  $\lambda(t) < 1$ .

Example : Consider the discrete-time system :

$$x_k = x_{k-1}^2 + w_{k-1}$$

$$z_k = x_k^3 + v_k$$

$$E v_k = E w_k = 0$$

$$E v_{k_1} v_{k_2} = 2 \Delta (k_2 - k_1)$$

$$E w_{k_1} w_{k_2} = \Delta (k_2 - k_1)$$

$$E x(0) = \hat{x}_0 = 2$$

$$x_x^{nom.} = 2$$

$$P_0 = 1$$

Find the EKF time & measurement update eqns) ? for k=1

Ans. :  $A_1 = \left. \frac{\partial (x^2)}{\partial x} \right|_{x=x^{nom.}} = 2x \Big|_{x=2} = 4$

$$H_1 = \left. \frac{\partial (x^3)}{\partial x} \right|_{x=x^{nom.}} = 3x^2 \Big|_{x=2} = 12$$

EKF Time update eqns) : (for k=1)

$$\hat{x}_1^- = \hat{x}_{k-1}^2$$

$$P_1^- = A_1 P_0 A_1^T + Q_0 = 16 P_0 + 1$$

EKF measurement update eqns) : (for k=1)

$$K_1 = \frac{12 P_{11}^-}{144 P_{11}^- + 2}$$

$$\hat{x}_1 = \hat{x}_{k-1}^2 + K_1 (z_1 - (\hat{x}_1^-)^3)$$

$$P_1 = [1 - 12 K_1] P_1^-$$

(From : Kalman Filtering, Mohinder Grewal & Angus Andrews)