IDENTIFICACIÓN Y CONTROL ADAPTABLE

FERNANDO DI SCIASCIO

PRÁCTICA DE IDENTIFICACIÓN RECURSIVA 2007

INTRODUCCIÓN TEÓRICA

El algoritmo de identificación recursiva mas conocido y empleado es el de mínimos cuadrados recursivos (RLS) que surge de la ecuación normal al volverla recursiva[1].

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

Definiendo $R(t) = P^{-1}(t)$ otra forma usual de presentar el algoritmo en la bibliografía es:

$$\boxed{\begin{aligned}\hat{\theta}(t) &= \hat{\theta}(t-1) + P(t)\varphi(t)\left[y(t) - \varphi^T(t)\hat{\theta}(t-1)\right] \\ P^{-1}(t) &= \lambda(t)P^{-1}(t-1) + \varphi(t)\varphi^T(t)\end{aligned}} \qquad (1')$$

Tras aplicar el lema de inversión de matrices nos queda el algoritmo siguiente que evita la inversión de una matriz de dxd (d es la dimensión del vector de parámetros θ y del regresor φ). Para pocos parámetros la (1) y la (2) son equivalentes, pero para muchos parámetros la (2) es mucho mas rápida)

$$\boxed{\begin{aligned}\hat{\theta}(t) &= \hat{\theta}(t-1) + L(t)\left[y(t) - \varphi^T(t)\hat{\theta}(t-1)\right] \\ L(t) &= \frac{P(t-1)\varphi(t)}{\lambda(t) + \varphi^T(t)P(t-1)\varphi(t)} \\ P(t) &= \left[P(t-1) - \frac{P(t-1)\varphi(t)\varphi^T(t)P(t-1)}{\lambda(t) + \varphi^T(t)P(t-1)\varphi(t)}\right] / \lambda(t)\end{aligned}} \qquad (2)$$

---

[1] La ecuación normal es: $\hat{\theta} = \left[\sum_{t=1}^{N}\varphi(t)\varphi^T(t)\right]^{-1}\left[\sum_{t=1}^{N}\varphi(t)y(t)\right]$

Recordar que el algoritmo RLS es válido para modelos linealmente parametrizados $\hat{y}(t) = \varphi^T(t)\hat{\theta}(t-1)$ como el ARX y no es en general válido para modelos no linealmente parametrizados (ver apuntes).

En la (1) y (2) $\lambda(t)$ es el factor de olvido, este puede ser variable, generalmente se adopta un valor constante λ. El factor de olvido es necesario cuando la planta que se quiere identificar varía en el tiempo en forma abrupta pero esporádicamente o cuando varía en forma lenta (con respecto al algoritmo). Si la planta es fija se adopta $\lambda = 1$. Al implementar el algoritmo RLS ((1) o (2)) en Simulink, las entradas son $y(t)$ y $\varphi(t)$ y la salida es $\theta(t)$.

Algoritmo PEM recursivo (RPEM)

Para modelos no linealmente parametrizados (por ejemplo modelos ARMAX) no es posible emplear el algoritmo RLS, para estos casos el algoritmo mas eficiente es el RPEM.

$$\boxed{\begin{aligned}\varepsilon(t) &= y(t) - \hat{y}(t)\\ \hat{\theta}(t) &= \hat{\theta}(t-1) + R^{-1}(t)\,\psi(t)\,\varepsilon(t)\\ R(t) &= \lambda(t)R(t-1) + \psi(t)\,\psi^T(t)\end{aligned}} \qquad (3)$$

donde el vector $\psi(t)$ se define como[2]: $\psi(t) \triangleq \dfrac{d\,\hat{y}(t,\theta)}{d\theta}$

Aplicando el lema de inversión de matrices se llega al siguiente algoritmo:

$$\boxed{\begin{aligned}\hat{\theta}(t) &= \hat{\theta}(t-1) + L(t)\,\varepsilon(t)\\ \varepsilon(t) &= y(t) - \hat{y}(t)\\ L(t) &= \frac{P(t-1)\psi(t)}{\lambda(t) + \psi^T(t)P(t-1)\psi(t)}\\ P(t) &= \left[P(t-1) - \frac{P(t-1)\psi(t)\psi^T(t)P(t-1)}{\lambda(t) + \psi^T(t)P(t-1)\psi(t)}\right]/\lambda(t)\end{aligned}} \qquad (4)$$

$\psi(t) = \psi(t,\hat{\theta}(t-1))$, $\varphi(t) = \varphi(t,\hat{\theta}(t-1))$,
$\hat{y}(t) = \varphi^T(t,\hat{\theta}(t-1))\,\hat{\theta}(t-1) = \varphi^T(t)\,\hat{\theta}(t-1)$

[2] En realidad operacionalmente es: $\psi(t,\hat{\theta}(t-1)) \triangleq \left.\dfrac{d\,\hat{y}(t,\theta)}{d\theta}\right|_{\theta=\hat{\theta}(t-1)} = \dfrac{d\,\hat{y}(t,\hat{\theta}(t-1))}{d\hat{\theta}(t-1)}$

Ver el apéndice sobre la estructura ARMAX para un ejemplo del cálculo de $\psi(t)$.

Si el modelo está linealmente parametrizado (p.ej. ARX) $\hat{y}(t) = \varphi^T(t)\hat{\theta}(t-1)$

$$\psi(t) \triangleq \frac{d\,\hat{y}(t,\theta))}{d\theta} = \varphi(t)$$

entonces el RPEM (Ec.4) coincide con RLS (Ec.2).

Observación Importante: Lo anterior implica que el mismo soft desarrollado para RLS (en este caso bloques de Simulink) puede ser reutilizado y empleado para RPEN. Para el caso RLS como se dijo anteriormente las entradas son $y(t)$ y $\varphi(t)$ y la salida es $\theta(t)$.

Si se quiere emplear el mismo bloque para resolver RPEM las entradas deben ser $z(t)$ y $\psi(t)$, donde

$$z(t) = y(t) - \hat{y}(t,\hat{\theta}(t-1)) + \psi^T(t)\hat{\theta}(t-1)$$

debe ingresar $z(t)$ y no $y(t)$ ya que el algoritmo asume que ingresa $\varphi(t)$ y calcularía el error de predicción como $\varepsilon(t) = y(t) - \psi^T(t)\hat{\theta}(t-1)$ en lugar del correcto $\varepsilon(t) = y(t) - \hat{y}(t,\hat{\theta}(t-1))$. Al ingresar $z(t)$ y $\psi(t)$ se tiene:
$\varepsilon(t) = z(t) - \psi^T(t)\hat{\theta}(t-1) = y(t) - \hat{y}(t,\hat{\theta}(t-1))$

Algoritmos aproximados

En el algoritmo RLS la actualización de la estima $\hat{\theta}$ involucra menos carga computacional que el cálculo de $R(t)$ y su inversa (o equivalentemente $P(t) = R^{-1}(t)$) . Por lo que se proponen un par de algoritmos mas sencillos pero con inferiores velocidades de convergencia.

1- Algoritmo de Kaczmarz o de proyección

$$\boxed{\begin{array}{l}\hat{\theta}(t) = \hat{\theta}(t-1) + \dfrac{\mu\varphi(t)}{\alpha + \varphi^T(t)\varphi(t)}\left[y(t) - \varphi^T(t)\hat{\theta}(t-1)\right] \\ \alpha \geq 0\ ,\ 0 < \mu < 2\end{array}} \qquad (6)$$

2- Aproximación Estocástica (Algoritmo de Robbins-Monro)

Definiendo como antes:

$$\varepsilon(t,\hat{\theta}(t-1)) = y(t) - \hat{y}(t,\hat{\theta}(t-1))$$

$$\psi(t,\hat{\theta}(t-1)) \triangleq \left.\frac{d\,\hat{y}(t,\theta)}{d\theta}\right|_{\theta=\hat{\theta}(t-1)} = -\left.\frac{d\,\varepsilon(t,\theta)}{d\theta}\right|_{\theta=\hat{\theta}(t-1)}$$

y considerando en el algoritmo RPEM (Ec.(3)) $R(t) = I$ (matriz identidad) se obtiene el siguiente algoritmo:

$$\boxed{\hat{\theta}(t) = \hat{\theta}(t-1) + \mu(t)\,\psi(t,\hat{\theta}(t-1))\varepsilon(t,\hat{\theta}(t-1))} \qquad (7)$$

donde $\{\mu(t)\}$ es una secuencia de números positivos que satisface determinadas condiciones que aseguran la convergencia del algoritmo, típicamente

i) $\displaystyle\sum_{t=1}^{\infty}\mu(t) = \infty$

ii) $\displaystyle\lim_{t\to\infty}\mu(t) = 0$

Si el modelo es linealmente parametrizado $\hat{y}(t,\hat{\theta}(t-1)) = \varphi^T(t)\,\hat{\theta}(t-1)$, entonces

$$\psi(t,\hat{\theta}(t-1)) = \left.\frac{d\,\hat{y}(t,\theta)}{d\theta}\right|_{\theta=\hat{\theta}(t-1)} = \left.\frac{d\left\{\varphi^T(t)\,\hat{\theta}(t-1)\right\}}{d\theta}\right|_{\theta=\hat{\theta}(t-1)} = \varphi(t)$$

$$\boxed{\hat{\theta}(t) = \hat{\theta}(t-1) + \mu(t)\,\varphi(t)\varepsilon(t,\hat{\theta}(t-1))} \qquad (8)$$

frecuentemente se adopta $\mu(t) = \dfrac{1}{t}$, en otras ocasiones por simplicidad se adopta $\mu(t) = \mu$ constante, $0 < \mu < 1$ que claramente no verifica la segunda condición y por lo tanto es mas difícil asegurar la convergencia del algoritmo.

$$\boxed{\hat{\theta}(t) = \hat{\theta}(t-1) + \mu\,\psi(t,\hat{\theta}(t-1))\varepsilon(t,\hat{\theta}(t-1))} \qquad (9)$$

y para el caso linealmente parametrizado[3]

$$\boxed{\hat{\theta}(t) = \hat{\theta}(t-1) + \mu\,\varphi(t)\varepsilon(t,\hat{\theta}(t-1))} \qquad (10)$$

Para el algoritmo (10) se verifica que el algoritmo es estable si μ verifica la relación $0 < \mu < {}^{2}\!/_{\lambda_{\max}}$, donde $\lambda_{\max}$ es el autovalor mayor de la matriz (definida positiva) $R = E[\varphi(t)\varphi(t)^T]$ generalmente se adopta $\mu\,\lambda_{\max} \ll 1$. Los algoritmos (8) y (10) son muy conocidos en la ingeniería en particular en el área de procesamiento adaptable de señales. Se denomina algoritmo LMS (least mean squares). Fueron desarrollados por Bernard Widrow a fines de los 60.

[3] Los algoritmos simplificados (9) y (10) son los mas simples posibles. En la página 22 del apunte **Parametrica-di Sciascio.pdf** se los presenta como “Estimación de parámetros por el método del gradiente”.

DESARROLLO DE LA PRÁCTICA

Con el fin de su posterior empleo en las prácticas de Control Adaptable se pide:

1) Desarrollar en Simulink el algoritmo RLS (2) y el aproximado (10) para identificar en forma recursiva modelos ARX (modelos linealmente parametrizados)

$$\boxed{\begin{aligned}\hat{\theta}(t) &= \hat{\theta}(t-1) + L(t)\left[y(t) - \varphi^T(t)\hat{\theta}(t-1)\right] \\ L(t) &= \frac{P(t-1)\varphi(t)}{\lambda(t) + \varphi^T(t)P(t-1)\varphi(t)} \\ P(t) &= \left[P(t-1) - \frac{P(t-1)\varphi(t)\varphi^T(t)P(t-1)}{\lambda(t) + \varphi^T(t)P(t-1)\varphi(t)}\right] / \lambda(t)\end{aligned}} \quad (2)$$

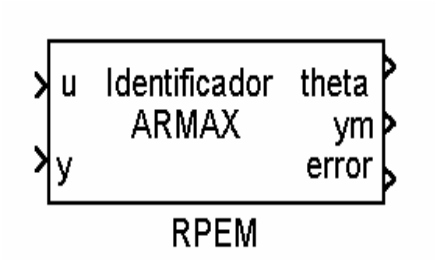
$$\boxed{\hat{\theta}(t) = \hat{\theta}(t-1) + \mu\,\varphi(t)\varepsilon(t, \hat{\theta}(t-1))} \quad (10)$$

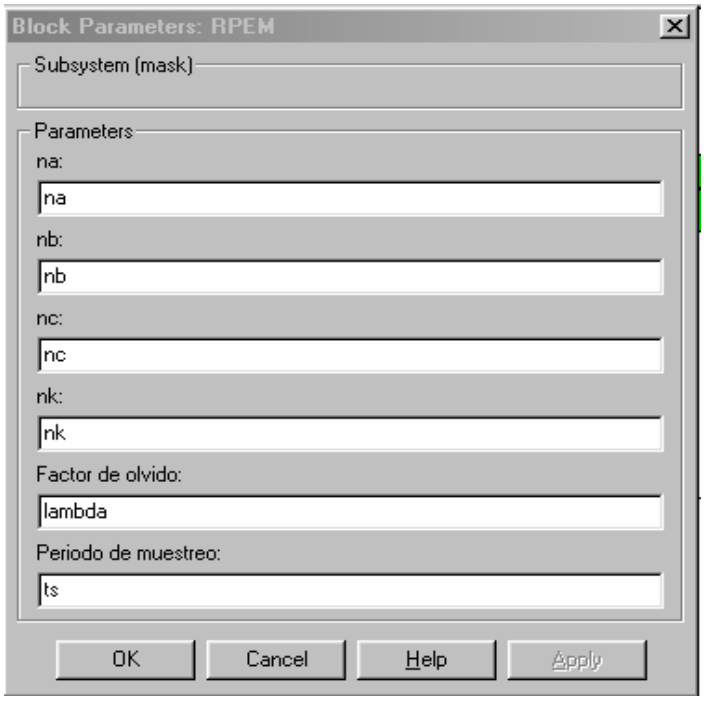
2) Desarrollar en Simulink el algoritmo RPEM (4) y el aproximado (9) para identificar en forma recursiva modelos ARMAX (modelos no-linealmente parametrizados)

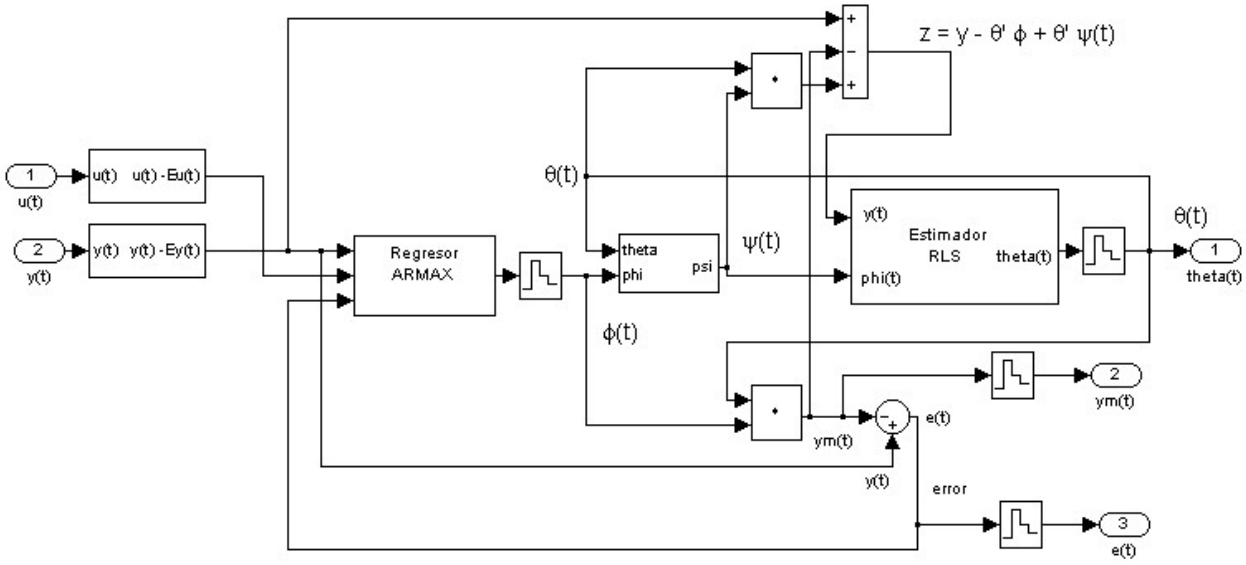
$$\boxed{\begin{aligned}\hat{\theta}(t) &= \hat{\theta}(t-1) + L(t)\,\varepsilon(t) \\ \varepsilon(t) &= y(t) - \hat{y}(t) \\ L(t) &= \frac{P(t-1)\psi(t)}{\lambda(t) + \psi^T(t)P(t-1)\psi(t)} \\ P(t) &= \left[P(t-1) - \frac{P(t-1)\psi(t)\psi^T(t)P(t-1)}{\lambda(t) + \psi^T(t)P(t-1)\psi(t)}\right] / \lambda(t)\end{aligned}} \quad (4)$$

$$\boxed{\hat{\theta}(t) = \hat{\theta}(t-1) + \mu\,\psi(t, \hat{\theta}(t-1))\varepsilon(t, \hat{\theta}(t-1))} \quad (9)$$

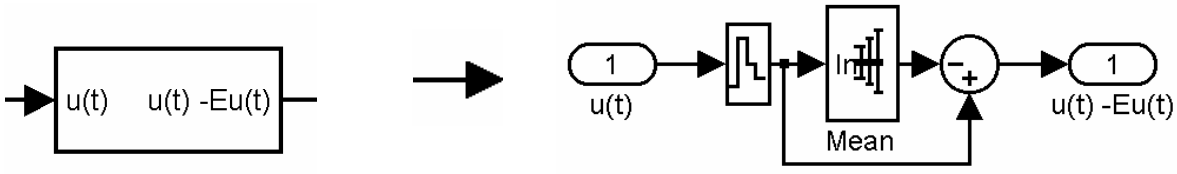
Ejemplo de implementación en Simulink del RPEM







Observación: Los bloques que aparecen a la entrada remueven los valores medios.



El bloque regresor ARMAX (ver el apéndice B sobre la estructura ARMAX) se construye siguiendo las líneas del apéndice A.

Apéndice A: Diseño de regresores con Simulink

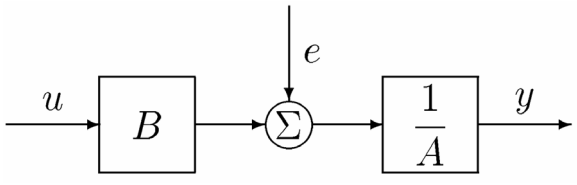
En esta sección se dan las guías para el diseño de regresores en Simulink ya que estos constituyen una parte importante en la identificación recursiva. Supongamos que deseamos construir un bloque que genere el regresor para la estructura ARX.

La estructura ARX se define mediante los polinomios $A(q)$ y $B(q)$

$$y(t) = \frac{B(q)}{A(q)}u(t) + \frac{1}{A(q)}e(t)$$

con

$$A(q) = 1 + a_1 q^{-1} + a_2 q^{-2} + \cdots + a_{na} q^{-na}$$
$$B(q) = \left[b_1 + b_2 q^{-1} + \cdots + b_{nb} q^{-nb+1}\right] q^{-nk} \quad (4)$$



La mejor estima de la salida es:

$$\hat{y}(t) = [1 - A(q)]y(t) + B(q)u(t)$$

reemplazando $A(q)$ y $B(q)$ en el segundo miembro

$$\hat{y}(t) = \left[-a_1 q^{-1} - a_2 q^{-2} - \cdots - a_{na} q^{-na}\right] y(t) + \left[b_1 + b_2 q^{-1} + \cdots + b_{nb} q^{-nb+1}\right] q^{-nk} u(t)$$

$$\hat{y}(t) = -a_1 y(t-1) - a_2 y(t-2) - \cdots - a_{na} y(t-na) + b_1 u(t-nk) + b_2 u(t-1-nk) + \cdots + + b_{nb} u(t-nb+1-nk)$$

se define ahora el vector de parámetros θ y el vector regresor $\varphi(t)$

$$\varphi(t,\theta) = \left[-y(t-1) \; -y(t-2) - \cdots - y(t-na) \quad u(t-nk) \quad u(t-1-nk) \; \cdots \; u(t-nb-nk)\right]^T$$

$$\theta = \left[a_1 \;\; a_2 \;\; \cdots \; a_{na} \;\; b_1 \;\; b_2 \;\; \cdots \; b_{nb}\right]^T$$

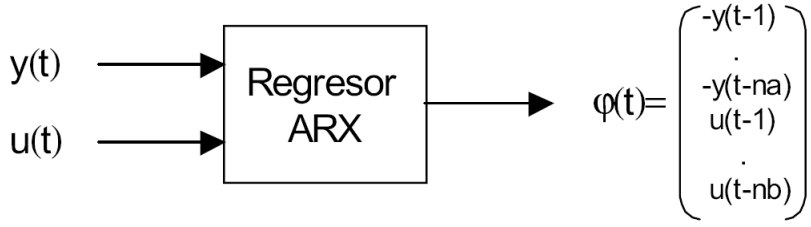
$$\boxed{\hat{y}(t) = \theta^T \varphi(t)}$$

esta ecuación expresa la estima del modelo como una parametrización lineal. Para el caso de la identificación recursiva los parámetros se estiman a partir de los datos disponibles

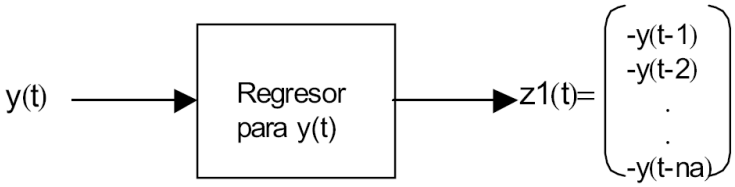
$$\boxed{\hat{y}(t)) = \hat{\theta}(t-1)^T \varphi(t)}$$

[4] Acá se adopta la nomenclatura del Toolbox de identificación de Matlab. Por defecto el toolbox asume $nk = 1 \;\; B(q) = b_1 q^{-1} + b_2 q^{-2} + \cdots + b_{nb} q^{-nb}$. También es usual en la bibliografía definir el polinomio $B(q)$ como $B(q) = \left[b_0 + b_1 q^{-1} + \cdots + b_{nb} q^{-nb}\right] q^{-nk}$.

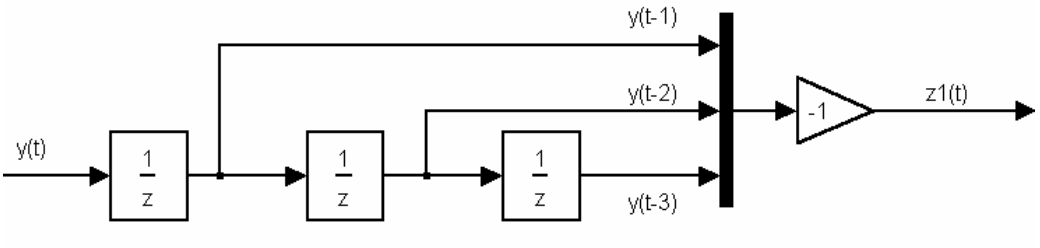
Luego el vector regresor está constituido por las $y(t)$ y $u(t)$ pasadas



Por simplicidad se implementa el regresor por partes, la primera involucra las $y(t)$ pasadas.



La forma mas sencilla es emplear una cadenas de bloques de retardo unitarios por ejemplo para $na = 3$ la estructura es la siguiente:



el problema con esta forma de implementar el regresor es que es poco flexible y se debe modificar para distintos na. La forma mas elegante para construir un regresor genérico para cualquier na es mediante un sistema dinámico discreto en su representación en el espacio de estados en el que la entrada al sistema es $y(t)$ y la salida son los estados del mismo, donde los estados son las entradas retrasadas con el signo cambiado.

$$\begin{cases} x(t+1) = Ax(t) + By(t) \\ \quad z_1(t) = Cx(t) + Dy(t) \end{cases}$$

Ejemplo para $na = 3$

$$x_1(t+1) = -y(t) \quad \rightarrow \quad x_1(t) = -y(t-1)$$

$$x_2(t+1) = x_1(t) \quad \rightarrow \quad x_2(t) = x_1(t-1) = -y(t-2)$$

$$x_3(t+1) = x_2(t) \quad \rightarrow \quad x_3(t) = x_2(t-1) = x_1(t-2) = -y(t-3)$$

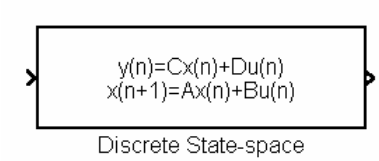
$$\begin{cases} \begin{bmatrix} x_{1(t+1)} \\ x_{2(t+1)} \\ x_{3(t+1)} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_{1(t)} \\ x_{2(t)} \\ x_{3(t)} \end{bmatrix} + \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} y(t) \\ z_1(t) = \begin{bmatrix} z_{1(t)} \\ z_{2(t)} \\ z_{3(t)} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{1(t)} \\ x_{2(t)} \\ x_{3(t)} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} y(t) \end{cases}$$

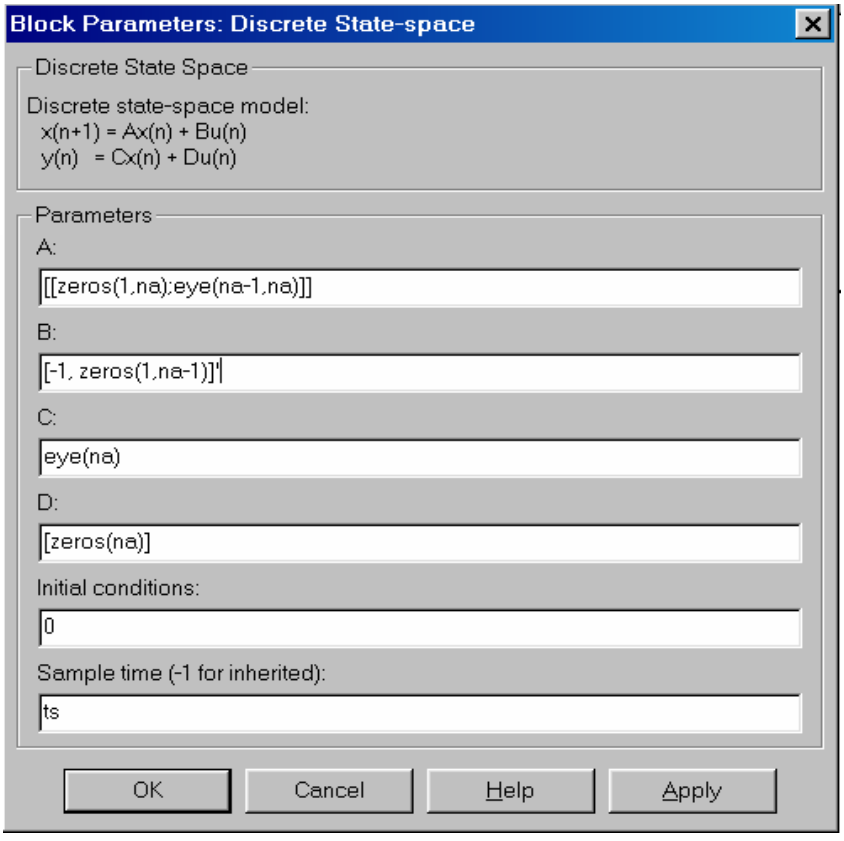
que es de la forma

$$\begin{cases} x(t+1) = Ax(t) + By(t) \\ z_1(t) = Cx(t) + Dy(t) \end{cases}$$

con $A = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$, $B = \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix}$, $C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$, $D = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

La anterior se implementa con un bloque Discrete State-Space de la librería Discrete de Simulink en forma genérica (función de na).





Para este ejemplo ($na = 3$)

A=[[zeros(1,na);eye(na-1,na)]] $\rightarrow \quad A = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$

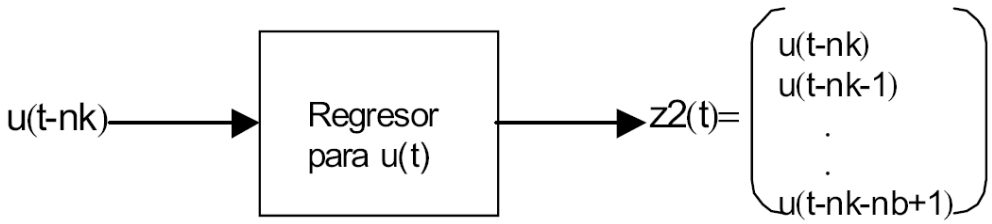
B= [-1, zeros(1,na-1)]' $\rightarrow \quad B = \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix}$

C= eye(na) $\rightarrow \quad C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

D =[zeros(na,1)] $\rightarrow \quad D = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

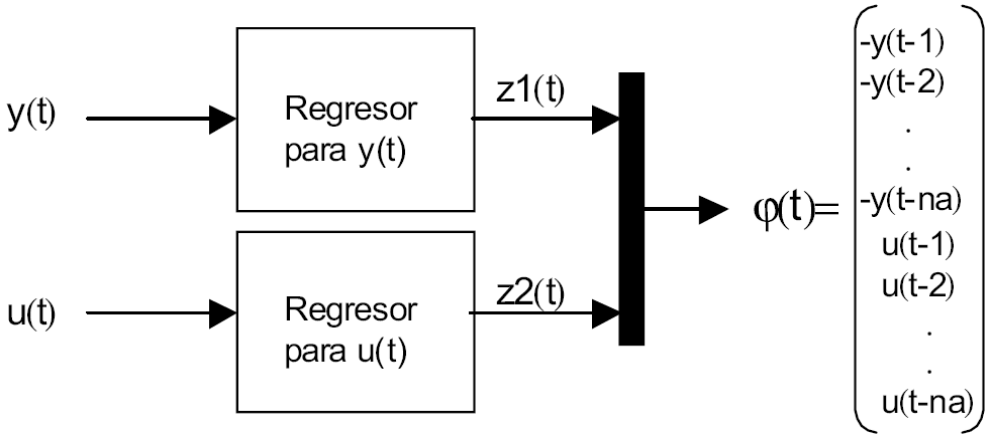
Para la parte del regresor que involucra las $u(t)$ pasadas el procedimiento es el mismo, teniendo en cuenta que la entrada al sistema ahora es $u(t-nk)$ y que las $u(t)$ pasadas no están invertidas por lo que

B= [1, zeros(1,nb-1)]' $\rightarrow \quad B = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$

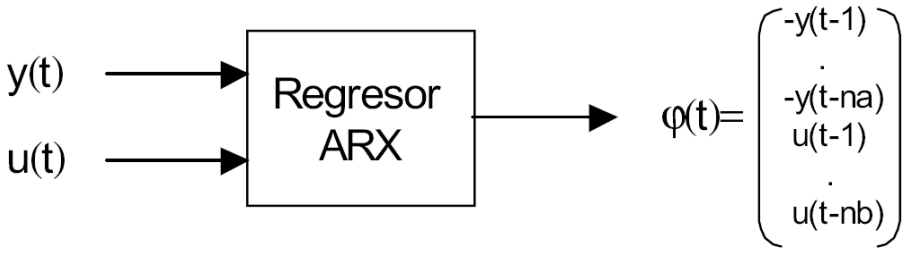


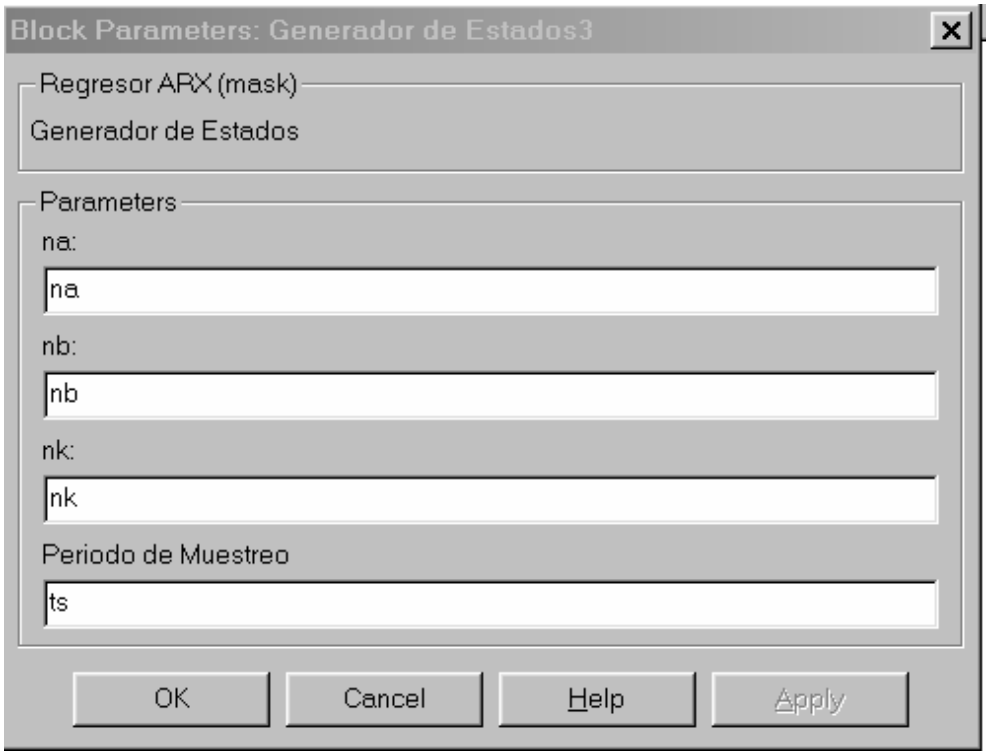
Otra posibilidad es considerar un sistema con $nb + nk$ estados donde la entrada es $u(t)$.

Finalmente con un multiplexor (bloque mux de Simulink) se forma el regresor completo.



También es posible generar en un solo paso el regresor completo donde las matrices A, B, C y D son funciones de na, nb y nk





Ejemplo para $na = 3$, $nb = 2$, $nk = 1$

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} -1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \qquad D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

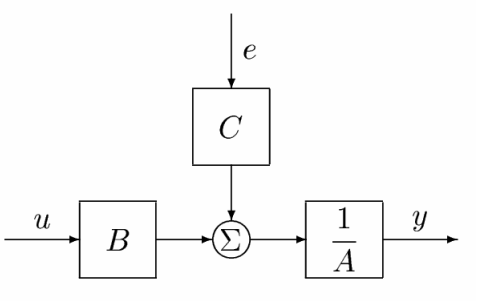
Apéndice B: Sobre la estructura ARMAX

La estructura ARMAX se define mediante los polinomios $A(q)$, $B(q)$ y $C(q)$

$$y(t) = \frac{B(q)}{A(q)}u(t) + \frac{C(q)}{A(q)}e(t) \tag{11}$$

con

$$A(q) = 1 + a_1 q^{-1} + a_2 q^{-2} + \cdots + a_{na} q^{-na}$$
$$B(q) = \left[b_1 + b_2 q^{-1} + \cdots + b_{nb} q^{-nb+1}\right] q^{-nk} \quad \text{(nota[5])}$$
$$C(q) = 1 + c_1 q^{-1} + c_2 q^{-2} + \cdots + c_{nc} q^{-nc}$$



La mejor estima de la salida es:

$$\hat{y}(t) = \frac{B(q)}{C(q)}u(t) + \left[1 - \frac{A(q)}{C(q)}\right]y(t) \tag{12}$$

pasando $C(q)$ al primer miembro

$$C(q)\hat{y}(t) = B(q)u(t) + [C(q) - A(q)]y(t) \tag{13}$$

sumando $[1 - C(q)]\hat{y}(t)$ en ambos miembros de la (13) y reordenando

$$\hat{y}(t) = -A(q)y(t) + B(q)u(t) + C(q)[y(t) - \hat{y}(t)] + \hat{y}(t)$$

sumando y restando $y(t)$ en el segundo miembro y reordenando queda:

$$\hat{y}(t) = [1 - A(q)]y(t) + B(q)u(t) + [C(q) - 1]\varepsilon(t) \tag{14}$$

con $\varepsilon(t) \triangleq y(t) - \hat{y}(t)$ que es el error de predicción, reemplazando $A(q)$, $B(q)$ y $C(q)$ en el segundo miembro de la (14)

$$\hat{y}(t) = \left[-a_1 q^{-1} - a_2 q^{-2} - \cdots - a_{na} q^{-na}\right]y(t) + \left[b_1 + b_2 q^{-1} + \cdots + b_{nb} q^{-nb+1}\right]q^{-nk}u(t) +$$
$$+\left[c_1 q^{-1} + c_2 q^{-2} + \cdots + c_{nc} q^{-nc}\right]\varepsilon(t)$$

[5] Ver nota 4

$$\hat{y}(t) = -a_1 y(t-1) - a_2 y(t-2) - \cdots - a_{na} y(t-na) + b_1 u(t-nk) + b_2 u(t-1-nk) + \cdots + \\ + b_{nb} u(t-nb+1-nk) + c_1 \varepsilon(t-1) + c_2 \varepsilon(t-2) + \cdots + c_{nc} \varepsilon(t-nc) \tag{15}$$

se define ahora el vector de parámetros θ y el vector regresor $\varphi(t)$

$$\varphi(t,\theta) = \begin{bmatrix} -y(t-1) & -y(t-2) & \cdots & -y(t-na) & u(t-nk) & u(t-1-nk) & \cdots & u(t-nb-nk) & \varepsilon(t-1) & \varepsilon(t-2) & \cdots & \varepsilon(t-nc) \end{bmatrix}^T$$

$$\theta = \begin{bmatrix} a_1 & a_2 & \cdots & a_{na} & b_1 & b_2 & \cdots & b_{nb} & c_1 & c_2 & \cdots & c_{nc} \end{bmatrix}^T$$

$$\boxed{\hat{y}(t) = \theta^T \varphi(t,\theta)} \tag{16}$$

la (16) expresa la estima del modelo como una parametrización pseudolineal o no lineal.[6] Para el caso de la identificación recursiva los parámetros se estiman a partir de los datos disponibles y la (16) es:

$$\boxed{\hat{y}(t,\hat{\theta}(t-1)) = \hat{\theta}(t-1)^T \varphi(t,\hat{\theta}(t-1))} \tag{16´}$$

En lo que sigue se calcula el vector $\psi(t) \triangleq \dfrac{\partial \hat{y}(t)}{\partial \theta}$, este vector es necesario para los algoritmos de identificación recursivos (en particular el RPEM). Se rescribe la (14)

$$\hat{y}(t) = [1 - A(q)]\, y(t) + B(q) u(t) + [C(q) - 1]\, \varepsilon(t) \tag{14}$$

y a partir de esta ecuación se calculan las distintas derivadas parciales de $\hat{y}(t)$ con respecto a los parámetros.

$$\frac{\partial \hat{y}(t)}{\partial a_i} = \frac{\partial [[1 - A(q)] y(t) + B(q) u(t) + [C(q) - 1] \varepsilon(t)]}{\partial a_i} = \frac{\partial [1 - A(q)]}{\partial a_i} y(t) + [C(q) - 1] \frac{\partial \varepsilon(t)}{\partial a_i} \tag{17}$$

$$\frac{\partial [1 - A(q)]}{\partial a_i} = \frac{\partial \left[-a_1 q^{-1} - a_2 q^{-2} - \cdots - a_i q^{-i} - \cdots - a_{na} q^{-na} \right]}{\partial a_i} = -q^{-i} \tag{18}$$

$$\frac{\partial \varepsilon(t)}{\partial a_i} = \frac{\partial [y(t) - \hat{y}(t)]}{\partial a_i} = -\frac{\partial \hat{y}(t)}{\partial a_i} \tag{19}$$

reemplazando las (18) y (19) en la (17)

[6] Por simplicidad en la notación se ha escrito $\hat{y}(t)$ y $\psi(t)$ en lugar de $\hat{y}(t,\theta)$ y $\psi(t,\theta)$

$$\frac{\partial \hat{y}(t)}{\partial a_i} = -q^{-i} y(t) - [C(q) - 1] \frac{\partial \hat{y}(t)}{\partial a_i}$$

reordenando

$$C(q) \frac{\partial \hat{y}(t)}{\partial a_i} = -y(t-i) \qquad \Rightarrow \quad \frac{\partial \hat{y}(t)}{\partial a_i} = -\frac{1}{C(q)} y(t-i) \tag{20}$$

repitiendo el cálculo para los parámetros b_i

$$\frac{\partial \hat{y}(t)}{\partial b_i} = \frac{\partial \left[[1 - A(q)] y(t) + B(q) u(t) + [C(q) - 1] \varepsilon(t) \right]}{\partial b_i} = \frac{\partial B(q)}{\partial b_i} u(t) + [C(q) - 1] \frac{\partial \varepsilon(t)}{\partial b_i} \tag{21}$$

$$\frac{\partial B(q)}{\partial b_i} = \frac{\partial \left[b_1 + b_2 q^{-1} + \cdots + b_i q^{-i+1} + \cdots + b_{nb} q^{-nb+1} \right] q^{-nk}}{\partial b_i} = q^{-i+1-nk} \tag{22}$$

$$\frac{\partial \varepsilon(t)}{\partial b_i} = \frac{\partial [y(t) - \hat{y}(t)]}{\partial b_i} = -\frac{\partial \hat{y}(t)}{\partial b_i} \tag{23}$$

reemplazando las (22) y (23) en la (21)

$$\frac{\partial \hat{y}(t)}{\partial b_i} = u(t - i + 1 - nk) - [C(q) - 1] \frac{\partial \hat{y}(t)}{\partial b_i}$$

reordenando

$$C(q) \frac{\partial \hat{y}(t)}{\partial b_i} = u(t - i + 1 - nk) \qquad \Rightarrow \quad \frac{\partial \hat{y}(t)}{\partial b_i} = \frac{1}{C(q)} u(t - i + 1 - nk) \tag{24}$$

finalmente para los parámetros c_i

$$\frac{\partial \hat{y}(t)}{\partial c_i} = \frac{\partial \left[[1 - A(q)] y(t) + B(q) u(t) + [C(q) - 1] \varepsilon(t) \right]}{\partial c_i} =$$

$$= \frac{\partial \left[[C(q) - 1] \varepsilon(t) \right]}{\partial c_i} = \frac{\partial [C(q) - 1]}{\partial c_i} \varepsilon(t) + [C(q) - 1] \frac{\partial \varepsilon(t)}{\partial c_i} \tag{25}$$

$$\frac{\partial [C(q) - 1]}{\partial c_i} = \frac{\partial \left[c_1 q^{-1} + c_2 q^{-2} + \cdots + c_i q^{-i} + \cdots + c_{nc} q^{-nc} \right]}{\partial c_i} = q^{-i} \tag{26}$$

$$\frac{\partial \varepsilon(t)}{\partial c_i} = \frac{\partial [y(t) - \hat{y}(t)]}{\partial c_i} = -\frac{\partial \hat{y}(t)}{\partial c_i} \tag{27}$$

reemplazando las (26) y (27) en la (25)

$$\frac{\partial \hat{y}(t)}{\partial c_i} = \varepsilon(t-i) - [C(q)-1]\frac{\partial \hat{y}(t)}{\partial c_i}$$

reordenando

$$C(q)\frac{\partial \hat{y}(t)}{\partial c_i} = \varepsilon(t-i) \qquad \Rightarrow \quad \frac{\partial \hat{y}(t)}{\partial c_i} = \frac{1}{C(q)}\varepsilon(t-i) \tag{28}$$

las (20), (24) y (28) son las derivadas parciales de $\hat{y}(t)$ respecto de los parámetros

$$\boxed{\begin{aligned}\frac{\partial \hat{y}(t)}{\partial a_i} &= -\frac{1}{C(q)}y(t-i)\\ \frac{\partial \hat{y}(t)}{\partial b_i} &= \frac{1}{C(q)}u(t-i+1-nk)\\ \frac{\partial \hat{y}(t)}{\partial c_i} &= \frac{1}{C(q)}\varepsilon(t-i)\end{aligned}}$$

que se pueden agrupar en forma vectorial

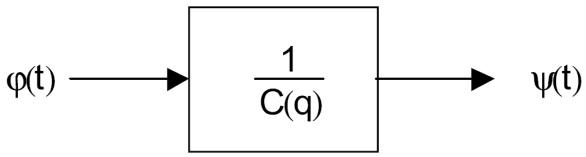
$$\begin{bmatrix}\frac{\partial \hat{y}(t)}{\partial a_i}\\ \frac{\partial \hat{y}(t)}{\partial b_i}\\ \frac{\partial \hat{y}(t)}{\partial c_i}\end{bmatrix} = \frac{1}{C(q)}\begin{bmatrix}-y(t-i)\\ u(t-i+1-nk)\\ \varepsilon(t-i)\end{bmatrix}$$

de la definición del vector de parámetros θ, del regresor $\varphi(t)$ y de $\psi(t)$ se tiene para la estructura ARMAX que:

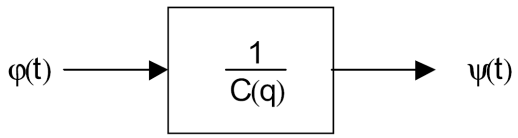
$$\boxed{\psi(t) = \frac{\partial \hat{y}(t)}{\partial \theta} = \frac{1}{C(q)}\varphi(t)} \tag{29}$$

se llega a que $\psi(t)$ se obtiene filtrando el regresor $\varphi(t)$ con el filtro $\frac{1}{C(q)}$

$$\frac{1}{C(q)} = \frac{1}{1 + c_1 q^{-1} + c_2 q^{-2} + \cdots + c_{nc} q^{-nc}}$$



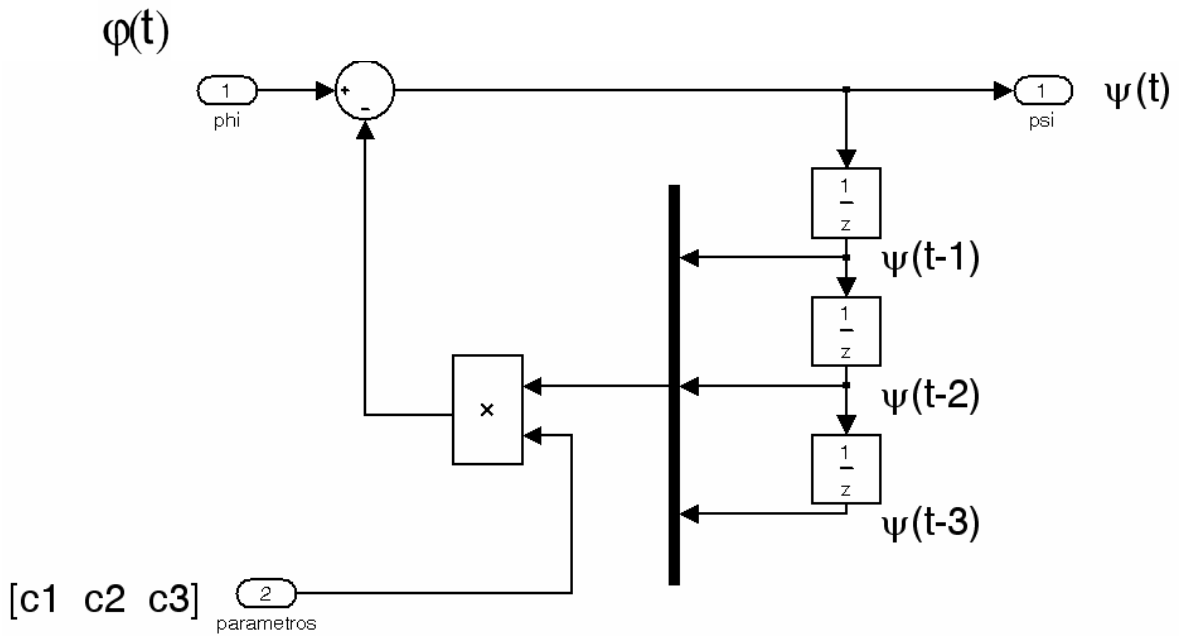
Implementación del filtro $\frac{1}{C(q)}$

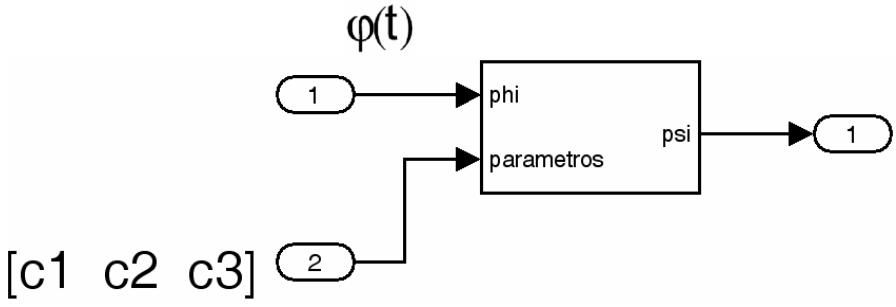


Desarrollando la (29)

$$C(q)\psi(t) = \varphi(t)$$
$$\left(1 + c_1 q^{-1} + c_2 q^{-2} + \cdots + c_{nc} q^{-nc}\right)\psi(t) = \varphi(t)$$
$$\psi(t) + c_1\psi(t-1) + c_2\psi(t-2) + \cdots + c_{nc}\psi(t-nc) = \varphi(t)$$
$$\psi(t) = \varphi(t) - c_1\psi(t-1) - c_2\psi(t-2) - \cdots - c_{nc}\psi(t-nc)$$

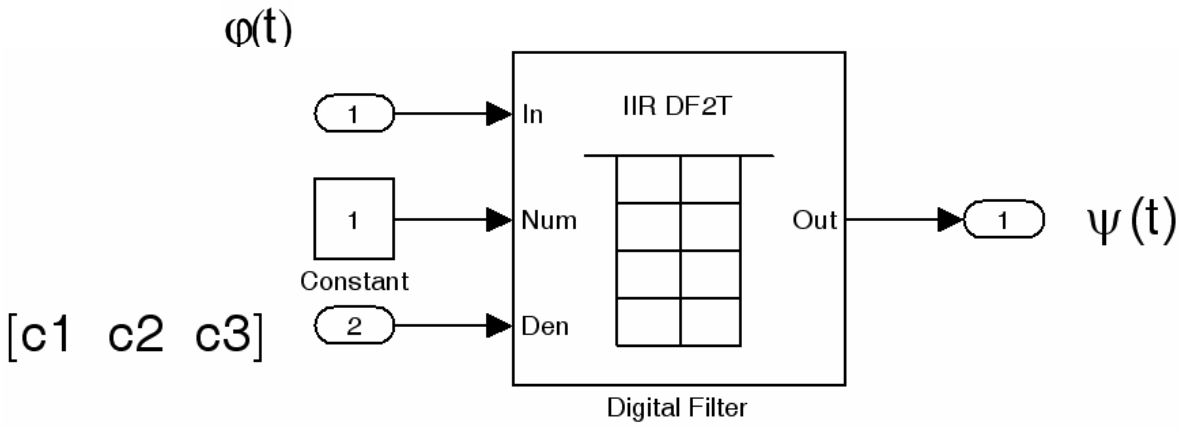
Se implementa en Simulink la última ecuación para $nc = 3$. El vector con los coeficientes del polinomio $C(q)$ es parte del vector de pará,etros que se está identificando.

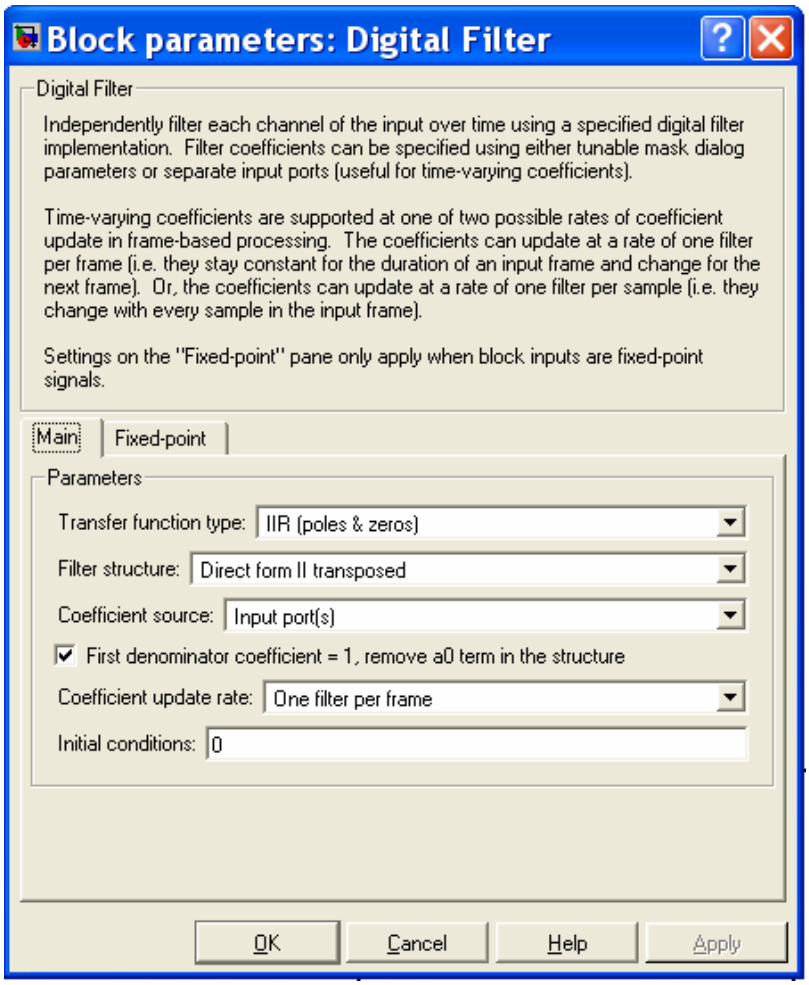




Observación

En realidad lo anterior se diseño para las versiones de Simulink anteriores al Matlab 6.0 que no disponían de un filtro adaptable incorporado en sus librerías. En el Simulink 5 (R13) de Matlab 6.5 en adelante ya se cuenta en **DSP Blockset/Filtering/Filter Designs**, el bloque Digital Filter y la implementación del filtro es directa.





BIBLIOGRAFÍA

Åström, K., Wittenmark, B., Adaptive Control, Second Edition, Editorial: Addison-Wesley Publishing Company, Inc., 1995.

di Sciascio F., *Identificación Paramétrica de Sistemas Lineales*, Apuntes de Cátedra, 1998.

Goodwin, G.C. y Sin, Kyai Sang, *Adaptive Filtering, Prediction and Control*, Prentice-Hall, Inc. Englewood Cliffs, 1984.

Ljung, L., *System Identification: Theory for the User*, 2nd Edition, Prentice Hall, Englewood Cliffs, N.J., 1999.

Ljung, L., Sodestrom, T., Theory and Prectice of Recursive *System Identification*, 2nd Edition, Prentice The MIT Press, 1983.

Ljung, L., *System Identification Toolbox, for use with Matlab*, User's Guide Version 5, The MathWorks, Inc, Natick, MA, 2002.

Söderström, T. & Stoica, P.: *System Identification*, Prentice Hall, Englewood Cliffs, N.J., 1989.