

# Control Inteligente

## Identificación de Sistemas con Redes Neuronales L9

Luis Moreno, Santiago Garrido, Dorin Copaci

Dpto. Ing. de Sistemas y Automática  
Universidad Carlos III  
Madrid

Oct 2019



# Table of contents

- 1 Identificación de Sistemas con Redes Neuronales

# Introducción

- La identificación estudia cómo obtener los modelos matemáticos de aquellos sistemas dinámicos que nos resultan de interés a partir de unos datos observados experimentalmente. Requiere tres cosas:
  - ① *Los datos.* El sistema es sometido a unos experimentos concebidos para realizar la identificación, donde se miden una serie de datos de entrada y de salida en el sistema.
  - ② *Un conjunto de modelos.* Los modelos se obtienen especificando entre los diferentes tipos de estructuras de modelos, cuál de ellos vamos a utilizar para ajustar los datos.
  - ③ *Un criterio* para determinar cual de los modelos que se evalúa es el mejor de acuerdo con los datos disponibles.

## Preliminar

- Una Red Neuronal representa una familia específica de mapas parametrizados.
- Si hay  $n_o$  elementos de entrada y  $n_L$  elementos de salida la red define un mapeo continuo

$$NN : \mathbb{R}^{n_o} \rightarrow \mathbb{R}^{n_L}.$$

- Para que el mapeo sea suprayectivo, se toma la capa de salida lineal.
- Las redes feedforward multicapa son aproximadores universales. [Cybenko(1989)] y [Hornik(1989)] demostraron que cualquier mapeo sobre un dominio compacto puede ser aproximado con la precisión necesaria por una red neuronal feedforward con una capa oculta.

## Preliminar

- Esto implica que dado un  $\epsilon > 0$  cualesquiera, una red neuronal  $NN_f$  con un número suficientemente grande de nodos puede determinarse para que

$$\|f(u) - NN_f(u)\| < \epsilon \text{ para todos } u \in \mathcal{D} \quad (1)$$

donde  $f$  es la función a aproximar y  $\mathcal{D}$  es un dominio compacto de dimensión finita sobre un espacio vectorial.

- El error de aproximación de la red viene dado por:

$$e(u) = f(u) - NN_f(u). \quad (2)$$

- El entrenamiento de la  $NN_f(\cdot)$  para aproximar  $f$  en el dominio  $\mathcal{D}$  es equivalente a minimizar:

$$I = \int_{\mathcal{D}} \|e(u)\| du \quad (3)$$

## Preliminar

- Si  $\theta$  denota los parámetros o pesos de la red. Si los pesos de la red se ajustan después de cada ejemplo de entrenamiento de acuerdo a:

$$\theta(k+1) = \theta(k) - \nu(k) \frac{\partial I}{\partial \theta} \Big|_{\theta=\theta(k)} \quad (4)$$

- La teoría de aproximaciones estocásticas (Ljung 1977) garantiza que si el paso  $\nu(k)$  cumple ciertas condiciones,  $I$  converge a un mínimo local con probabilidad 1.
- Si la hipersuperficie de prestaciones es **unimodal** esto implica que **se alcanza el mínimo global**.

## Selección de la estructura del modelo

- Buscamos ser capaces de identificar sistemas dinámicos no lineales en entornos deterministas o estocásticos.
- En una primera aproximación al problema, para ser capaces de describir mapeos no lineales se pueden utilizar redes feedforward multicapa o redes de funciones de base radial. En nuestro caso usaremos **redes feedforward** (perceptrón multicapa).
- Tomaremos como punto de partida las estructuras de modelos lineales.

## Respuesta temporal

- La salida de un sistema en tiempo discreto ante una cierta secuencia de entrada  $u(k)$ ,  $k = 1, \dots, \infty$  se puede expresar

$$y(k) = \sum_{i=0}^{\infty} g(i)u(k-i) \quad k = 0, 1, 2, \dots$$

- Se define la respuesta impulsional del sistema como la respuesta a una señal de tipo impulso de Dirac. Es decir, la respuesta impulsional de un sistema será la secuencia  $g(k)$ ,  $k = 0, \dots, \infty$ .
- Los sistemas cuya respuesta impulsional tiene un número finito de coeficientes distintos de cero, se denominan sistemas con **respuesta impulsional finita (FIR- finite impulse response)**.



# Modelos paramétricos

Dentro de los modelos paramétricos los más frecuentemente empleados son:

- las funciones de transferencia
- los modelos en variables de estado.

## Modelos de función de transferencia

Es usual utilizar las expresiones de la función de transferencia en tiempo discreto, dado que el proceso de identificación se realiza por parte de un computador que es el que obtiene y analiza los datos obtenidos.

- Un sistema discreto cuya ecuación lineal en diferencias es

$$y(k) = - \sum_{i=1}^p a_i y(k-i) + \sum_{i=0}^q b_i u(k-i)$$

donde  $y(k)$  es la secuencia de salida y  $u(k)$  es la secuencia de entrada al sistema.

- Podemos escribirlo como:

$$y(k) = \sum_{i=0}^{\infty} g(i) u(k-i) \quad k = 0, 1, 2, \dots$$

y como

$$G(q) = \frac{y(k)}{u(k)} = \frac{1 + b_1 z^{-1} + \dots + b_m z^{-q}}{1 + a_1 z^{-1} + \dots + a_n z^{-p}}$$

## Modelos de función de transferencia

- puede escribirse como

$$\begin{aligned}y(k) &= \sum_{j=1}^{\infty} g(j)u(k-j) \\ &= \sum_{j=1}^{\infty} g(j)[u(k)q^{-j}] \\ &= \left[ \sum_{j=1}^{\infty} g(j)q^{-j} \right] u(k) \\ &= G(q)u(k)\end{aligned}$$

donde

$$G(q) = \sum_{k=1}^{\infty} g(k)q^{-k}$$

## Modelos de función de transferencia

- Si suponemos que el sistema está sometido a ruido, para el ruido puede hacerse lo mismo obteniendo unas expresiones equivalentes

$$v(k) = H(q)e(k)$$

con

$$H(q) = \sum_{k=1}^{\infty} h(k)q^{-k}$$

- con lo que un modelo lineal con perturbación aditiva podrá expresarse de la siguiente forma

$$y(k) = G(q)u(k) + H(q)e(k) \quad (5)$$

- Dependiendo de la forma que adoptan las funciones  $G(q)$  y  $H(q)$  podemos hablar de cuatro tipos de modelos básicos: ARX, ARMAX, OE y Box-Jenkins.

## Estructuras de Modelos Lineales

- De acuerdo con [Ljung(2001)] un sistema se dice lineal si es posible describirlo por un modelo de la forma

$$y(t) = G(q^{-1})u(t) + H(q^{-1})e(t) \quad (6)$$

donde:

- $G$  y  $H$  son funciones de transferencia en el operador retardo temporal  $q^{-1}$ .
- El operador retardo opera de la forma siguiente

$$q^{-d}x(t) = x(t - d) \quad (7)$$

donde  $d$  es un múltiplo del tiempo de muestreo.

- $e(t)$  es una señal de tipo ruido blanco que es independiente de sus valores anteriores y que puede ser caracterizado por alguna función de densidad de probabilidad.
- En el caso multivariable (MIMO)  $u(t)$ ,  $y(t)$  y  $e(t)$  son vectores y  $G$  y  $H$  matrices polinómicas en  $q^{-1}$ .

## Estructuras de Modelos Lineales

- Si el sistema es lineal el objetivo de la identificación es determinar una buena estimación de las funciones de transferencia  $H$  y  $G$ .
- El criterio para definir cuando una estimación es buena lo relacionaremos con la capacidad del modelo para producir una predicción para el ciclo siguiente (one-step-ahead prediction) con errores de baja varianza.
- Para el sistema lineal general descrito por la ecuación 6 la predicción de mínima varianza viene dada por la expresión

$$\hat{y}(t|t-1) = H^{-1}(q^{-1})G(q^{-1})u(t) + [1 - H^{-1}(q^{-1})]y(t) \quad (8)$$

## Estructuras de Modelos Lineales

Veamos el significado de algunos conceptos:

- El **sistema verdadero** (true system) se asume que puede ser descrito por

$$y(t) = G_0(q^{-1})u(t) + H_0(q^{-1})e_0(t) \quad (9)$$

siendo  $e_0(t)$  una señal del tipo ruido blanco independiente de la señal de entrada  $u(t)$ .

- La **estructura del modelo**,  $\mathcal{M}$  es un conjunto parametrizado de modelos candidatos

$$\begin{aligned} y(t) &= G(q^{-1}, \theta)u(t) + H(q^{-1}, \theta)e(t) \\ \mathcal{M} &: \{G(q^{-1}, \theta), H(q^{-1}, \theta) | \theta \in \mathcal{D}_m\} \end{aligned} \quad (10)$$

donde  $\theta$  denota los  $p$  parámetros ajustables del modelo y  $\mathcal{D}_m$  es el algún subconjunto de  $\mathcal{R}^p$  en el que se busca el modelo.

## Estructuras de Modelos Lineales

- La estructura del modelo en forma de predictor tiene la forma

$$\hat{y}(t|t-1, \theta) = H^{-1}(q^{-1}, \theta)G(q^{-1}, \theta)u(t) + [1 - H^{-1}(q^{-1}, \theta)]y(t) \quad (11)$$

por sencillez notacional no representaremos el  $t-1$ .

- La estructura del modelo a menudo se representa en la forma

$$\hat{y}(t|\theta) = \phi^T \theta \quad (12)$$

donde  $\theta$  es el *vector de parámetros* y  $\phi$  es el *vector de regresores* (información disponible de entradas, salidas y señales derivadas de estas de los instantes anteriores).



## Estructuras de Modelos Lineales

- Un **modelo** es simplemente una elección particular del vector de parámetros, es decir  $\theta = \hat{\theta}$ .
- El modelo general que se ha comentado puede reescribirse en la forma

$$A(q^{-1})y(t) = q^{-d} \frac{B(q^{-1})}{F(q^{-1})} u(t) + \frac{C(q^{-1})}{D(q^{-1})} e(t) \quad (13)$$

donde

$$\begin{aligned} A(q^{-1}) &= 1 + a_1 q^{-1} + \dots + a_n q^{-n} \\ B(q^{-1}) &= 1 + b_1 q^{-1} + \dots + b_m q^{-m} \\ C(q^{-1}) &= 1 + c_1 q^{-1} + \dots + c_k q^{-k} \\ D(q^{-1}) &= 1 + d_1 q^{-1} + \dots + d_l q^{-l} \\ F(q^{-1}) &= 1 + f_1 q^{-1} + \dots + f_r q^{-r} \end{aligned} \quad (14)$$

## Modelo FIR (Finite Impulse Response)

- Es el modelo más simple,

$$G(q^{-1}, \theta) = q^{-d}B(q^{-1}) \quad H(q^{-1}, \theta) = 1 \quad (15)$$

el predictor queda en la forma

$$\hat{y}(t|\theta) = q^{-d}B(q^{-1})u(t) \quad (16)$$

- O en forma de regresión

$$\hat{y}(t|\theta) = \phi^T \theta \quad (17)$$

con

$$\begin{aligned} \phi(t) &= [u(t-d) \dots u(t-d-m)]^T \\ \theta &= [b_0 \dots b_m]^T \end{aligned} \quad (18)$$

## Modelo ARX (AutoRegressive, eXternal input)

- Es el modelo más simple, se obtiene como

$$y(k) + a_1y(k-1) + \dots + a_ny(k-n) = b_1u(k-1) + \dots + b_mu(k-m) + e(k)$$

donde  $e(k)$ , es un ruido blanco, y entra como un error directo en la ecuación en diferencias (se denomina también de *error en ecuación*).

- Denominando:

$$A(q^{-1}) = 1 + a_1q^{-1} + \dots + a_nq^{-n} \quad (19)$$

$$B(q^{-1}) = b_1q^{-1} + \dots + b_mu(k-m) \quad (20)$$

tenemos que

$$G(q^{-1}) = \frac{B(q^{-1})}{A(q^{-1})} \quad (21)$$

$$H(q^{-1}) = \frac{1}{A(q^{-1})} \quad (22)$$

## Modelo ARX (AutoRegressive, eXternal input)

- Puesto en forma general, es un modelo de la forma

$$G(q^{-1}, \theta) = q^{-d} \frac{B(q^{-1})}{A(q^{-1})} \quad H(q^{-1}, \theta) = \frac{1}{A(q^{-1})} \quad (23)$$

el predictor queda en la forma

$$\hat{y}(t|\theta) = q^{-d} B(q^{-1})u(t) + [1 - A(q^{-1})]y(t) \quad (24)$$

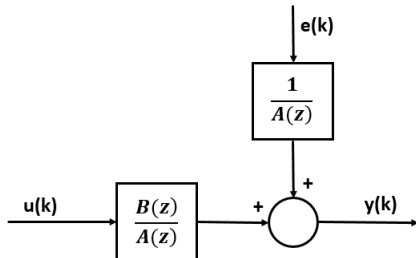
que en forma de regresión

$$\hat{y}(t|\theta) = \phi^T \theta \quad (25)$$

con

$$\begin{aligned} \phi(t) &= [y(t-1), \dots, y(t-n), u(t-d), \dots, u(t-d-m)]^T \\ \theta &= [-a_1, \dots, -a_n, b_0, \dots, b_m]^T \end{aligned} \quad (26)$$

## Estructura de modelo ARX



- Este modelo se denomina ARX, donde AR se refiere a la parte autorregresiva de  $A(q)y(k)$  y X a la entrada  $B(q)u(k)$ .
- En el caso especial de que  $n_a = 0$ ,  $y(k)$  es modelado como un sistema con respuesta impulsional finita (FIR).

## Modelo ARMAX (AutoRegressive, Moving Average, eXternal input)

- La mayor desventaja del modelo anterior estriba en la ausencia de libertad para describir las propiedades del término de perturbaciones.
- Es posible añadirle algo de flexibilidad al modelo anterior si consideramos que la perturbación es un ruido blanco de media móvil. Con esta consideración se obtiene un modelo con una estructura del siguiente tipo,

$$\begin{aligned}y(t) + a_1y(t-1) + \dots + a_ny(t-n) = \\ b_1u(t-1) + \dots + b_mu(t-m) + \\ e(t) + c_1e(t-1) + \dots + c_ke(t-k)\end{aligned}\quad (27)$$

con

$$C(q) = 1 + c_1q^{-1} + \dots + c_kq^{-k}$$

## Modelo ARMAX

- y la ecuación puede reescribirse como

$$A(q^{-1})y(t) = B(q^{-1})u(t) + C(q^{-1})e(t) \quad (28)$$

donde

$$G(q^{-1}) = \frac{B(q^{-1})}{A(q^{-1})} \quad (29)$$

$$H(q^{-1}) = \frac{C(q^{-1})}{A(q^{-1})} \quad (30)$$

## Modelo ARMAX (AutoRegressive, Moving Average, eXternal input)

- Es un modelo de la forma

$$G(q^{-1}, \theta) = q^{-d} \frac{B(q^{-1})}{A(q^{-1})} \quad H(q^{-1}, \theta) = \frac{C(q^{-1})}{A(q^{-1})} \quad (31)$$

el predictor queda en la forma

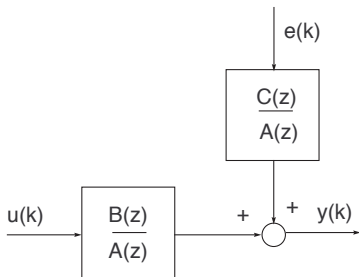
$$\begin{aligned} \hat{y}(t|\theta) &= q^{-d} \frac{B(q^{-1})}{C(q^{-1})} u(t) + \left[1 - \frac{A(q^{-1})}{C(q^{-1})}\right] y(t) \\ &= q^{-d} B(q^{-1}) u(t) + [1 - A(q^{-1})] y(t) \\ &\quad + [C(q^{-1}) - 1] \epsilon(t, \theta) \end{aligned} \quad (32)$$

donde  $\epsilon(t, \theta)$  representa el error de predicción o residuo, y en forma de regresión  $\hat{y}(t|\theta) = \phi^T \theta$  con

$$\begin{aligned} \phi(t) &= [y(t-1), \dots, y(t-n), u(t-d), \dots, u(t-d-m), \\ &\quad \epsilon(t, \theta), \dots, \epsilon(t-k, \theta)]^T \\ \theta &= [-a_1, \dots, -a_n, b_0, \dots, b_m, c_1, \dots, c_k]^T \end{aligned} \quad (33)$$



## Estructura de modelo ARMAX



Este modelo es estándar en control y también en econometría. Una versión de este es el modelo ARIMA(X), donde la I indica integración, y es muy útil para describir sistemas con perturbaciones lentas. Se obtiene sustituyendo en la ecuación  $y(t)$  por  $\Delta I(t) = y(t) - y(t - 1)$ . Este modelo se suele utilizar cuando las perturbaciones aparecen en la entrada del sistema, lo que es frecuente en muchos sistemas.

## Modelo OE (Output Error)

- En los modelos anteriores todas las descripciones de  $G$  y de  $H$  tenían al polinomio  $A$  en el denominador. Desde un punto de vista físico, es más natural parametrizar estas funciones independientemente.
- Si consideramos que la perturbación es un ruido blanco, y un modelo de estructura

$$\omega(t) + f_1\omega(t-1) + \dots + f_r\omega(t-r) = b_1u(t-1) + \dots + b_mu(t-m) \quad (34)$$

$$y(t) = \omega(t) + e(t) \quad (35)$$

con

$$F(q^{-1}) = 1 + f_1q^{-1} + \dots + f_rq^{-r}$$

el modelo quedará como

$$y(t) = \frac{B(q^{-1})}{F(q^{-1})}u(t) + e(t) \quad (36)$$

## Modelo OE (Output Error)

- Es un modelo de la forma

$$G(q^{-1}, \theta) = q^{-d} \frac{B(q^{-1})}{F(q^{-1})} \quad H(q^{-1}, \theta) = 1 \quad (37)$$

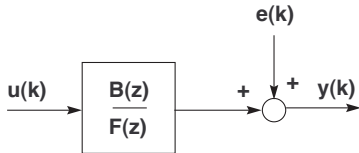
el predictor queda en la forma

$$\begin{aligned} \hat{y}(t|\theta) &= q^{-d} \frac{B(q^{-1})}{F(q^{-1})} u(t) \\ &= q^{-d} B(q^{-1}) u(t) + [1 - F(q^{-1})] \hat{y}(t|\theta) \end{aligned} \quad (38)$$

y en forma de regresión  $\hat{y}(t|\theta) = \phi^T \theta$  con

$$\begin{aligned} \phi(t) &= [\hat{y}(t-1), \dots, \hat{y}(t-r), u(t-d), \dots, u(t-d-m)]^T \\ \theta &= [-f_1, \dots, -f_r, b_0, \dots, b_m]^T \end{aligned} \quad (39)$$

## Estructura de modelo OE



Este modelo presenta mayores dificultades a la hora de realizar su estimación que los modelos ARX o ARMAX. Permite obtener una descripción correcta de la función de transferencia determinista sin importar la forma que tomen las perturbaciones.

## Modelo en el Espacio de Estados (State Space Innovation Form)

- Si el sistema lo describimos por su representación de estado, es decir en la forma

$$\begin{aligned}x(t+1) &= A(\theta)x(t) + B(\theta)u(t) + w(t) \\ y(t) &= C(\theta)x(t) + v(t)\end{aligned}\quad (40)$$

donde  $w(t)$  y  $v(t)$  son ruidos blancos independientes de la señal de control  $u(t)$ , con

$$E \left\{ \begin{bmatrix} w(t) \\ v(t) \end{bmatrix} \begin{bmatrix} w^T(t) & v^T(t) \end{bmatrix} \right\} = \begin{bmatrix} R_w(\theta) & R_{wv}(\theta) \\ R_{wv}^T(\theta) & R_v(\theta) \end{bmatrix} \quad (41)$$

- El predictor óptimo tiene la forma (Soderstrom y Stoica 1989)

$$\begin{aligned}\hat{x}(t+1, \theta) &= A(\theta)\hat{x}(t, \theta) + B(\theta)u(t) + K(\theta)\epsilon(t, \theta) \\ \hat{y}(t, \theta) &= C(\theta)\hat{x}(t, \theta)\end{aligned}\quad (42)$$

## Modelo en el Espacio de Estados (State Space Innovation Form)

- El término  $K(\theta)$  tiene la siguiente expresión

$$K(\theta) = [APC^T + R_{ww}][CPC + R_w]^{-1} \quad (43)$$

donde  $P(\theta)$  representa la solución de la ecuación de Ricatti (covarianza de la estimación de estado)

$$P(\theta) = APA^T + R_w - [APC^T + R_{ww}][CPC + R_w]^{-1}[APC^T + R_{ww}]^T \quad (44)$$

- A este predictor óptimo también se le denomina **Filtro de Kalman** y la matriz  $K(\theta)$  se le denomina ganancia de Kalman.
- A la expresión 42 se le denomina **state space innovation form**.

## Modelo en el Espacio de Estados (State Space Innovation Form)

- La relación entre esta representación de estado y la de entrada-salida viene dada por (Ljung 1999)

$$\begin{aligned}G(q^{-1}, \theta) &= C(\theta)[qI - A(\theta)]^{-1}B(\theta) \\H(q^{-1}, \theta) &= C(\theta)[qI - A(\theta)]^{-1}K(\theta) + I\end{aligned}\quad (45)$$

## Modelos No Lineales basados en NN

- En esta aproximación se usará una estructura de red del tipo perceptron multicapa (feedforward) que permite un buen aprendizaje de relaciones no lineales a partir de un conjunto de datos.
- Para buscar una familia de estructuras de modelos adecuadas para la identificación de sistemas dinámicos no lineales usaremos la red ya comentada y habrá que determinar:
  - Las entradas a la red
  - La arquitectura interna de la red.
- Con frecuencia se usan unas entradas similares a las entradas vistas para los modelos lineales manteniéndose la estructura de la red feedforward sin ninguna modificación.



## Modelos No Lineales basados en NN

Este enfoque presenta algunas ventajas:

- Es una extensión natural de los métodos lineales ya conocidos.
- Su arquitectura interna puede expandirse gradualmente a medida que se requiera mayor flexibilidad para modelar relaciones no lineales más complejas.
- Las decisiones estructurales que es necesario tomar por el diseñador son reducidas a un nivel razonable de manejar.
- Son adecuadas para el diseño de sistemas de control.

## Modelos No Lineales basados en NN

- En estos modelos no lineales asociados a las estructuras vistas para los lineales se obtendrán modelos de la forma:

$$y(t) = g[\phi(t, \theta), \theta] + e(t) \quad (46)$$

o en forma de predictor

$$\hat{y}(t|\theta) = g[\phi(t, \theta), \theta] \quad (47)$$

donde:

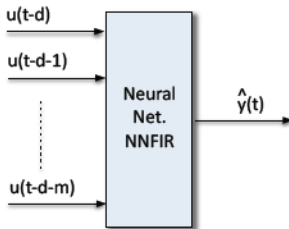
- $\phi(t, \theta)$  es el vector de regresores.
- $\theta$  es el vector que contiene los parámetros ajustables del modelo que en las redes neuronales son los pesos de las conexiones.
- $g$  es la función que aproximamos mediante la red neuronal que se supone de estructura feedforward.
- Según el vector de regresores que se tome tendremos una estructura de modelo no lineal u otra.

# NNFIR

- El modelo no lineal es de la forma

$$\hat{y}(t|\theta) = NN_g[\phi(t, \theta), \theta] \quad (48)$$

$$\phi(t) = [u(t-d) \dots u(t-d-m)]^T \quad (49)$$

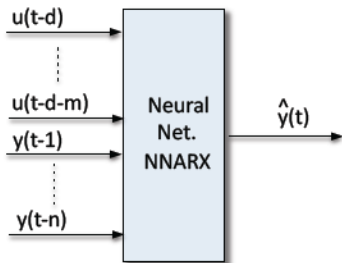


## NNARX

- El modelo no lineal es de la forma

$$\hat{y}(t|\theta) = NN_g[\phi(t, \theta), \theta] \quad (50)$$

$$\phi(t) = [y(t-1), \dots, y(t-n), u(t-d), \dots, u(t-d-m)]^T \quad (51)$$



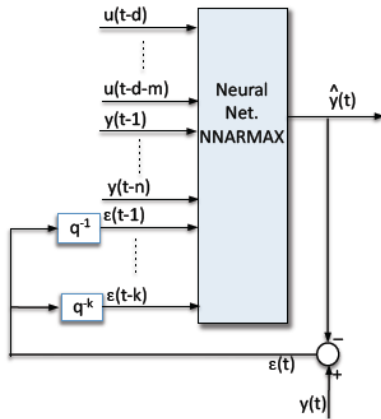
# NNARX

- El modelo NNARX es una buena elección de partida cuando:
  - el sistema es determinista
  - el sistema presenta un nivel de ruido muy bajo

## NNARMAX

- El modelo no lineal es de la forma  $\hat{y}(t|\theta) = NN_g[\phi(t, \theta), \theta]$  con

$$\phi(t) = [y(t-1), \dots, y(t-n), u(t-d), \dots, u(t-d-m), \epsilon(t, \theta), \dots, \epsilon(t-k, \theta)]^T \quad (52)$$

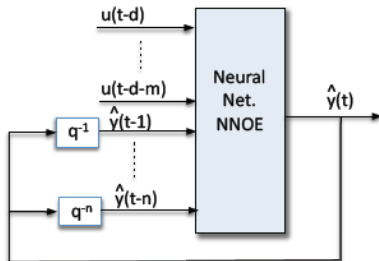


# NNARMAX

- Aunque la función  $g$  se aproxima por una red neuronal feedforward el predictor tiene realimentación ya que los regresores realimentan el error de predicción.
- En el modelo ARMAX los errores de predicción dependen de la salida del modelo por lo que se establece una realimentación, tal y como se puede observar en el esquema del NNARMAX.
- A este tipo de estructura de red se le suele denominar redes recurrentes.
- Puede ocurrir que estos modelos sean estables en ciertas zonas e inestables en otras.

# NNOE

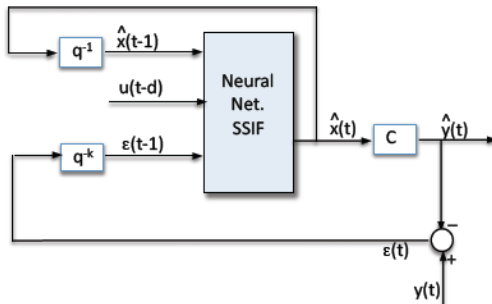
- En la estructura NNOE algunos de los regresores son las predicciones de salidas pasadas y tiene problemas similares a la estructura NNARMAX





# NNSSIF

- La estructura NNSSIF para un predictor es la siguiente:



## NNSSIF

- En general los modelos SSIF son más complejos de identificar que los modelos de E/S.
- El modelo que se ha adoptado en el esquema implementa

$$\begin{aligned}\hat{x}(t+1, \theta) &= NN_g[\phi(t, \theta), \theta] \\ \hat{y}(t, \theta) &= C(\theta)\hat{x}(t, \theta)\end{aligned}\quad (53)$$

donde

$$\phi(t, \theta) = \begin{bmatrix} \hat{x}(t, \theta) \\ u(t) \\ \epsilon(t, \theta) \end{bmatrix}\quad (54)$$

- Al igual que en los modelos NARMAX y NNOE el predictor contiene una realiimentación dado que el estado previo y los vectores de error de predicción forman parte del vector de regresión.

## Estructuras de modelo modificadas

- Se han propuesto diferentes variaciones sobre estas estructuras de modelo de forma que se crean en ocasiones estructuras con mezclas de modelos lineales y no lineales (Narendra 1992, Bittani 1993).

## Esquemas NNARX modificados

- El modelo no lineal es de la forma

$$\hat{y}(t|\theta) = NN_{g_y}[\phi_y(t), \theta_y] + NN_{g_u}[\phi_u(t), \theta_u] \quad (55)$$

$$\hat{y}(t|\theta) = NN_g[\phi_y(t), \theta_y] + q^{-d}B(q^{-1})u(t) \quad (56)$$

$$\hat{y}(t|\theta) = [1 - A(q^{-1})]y(t) + NN_g[\phi_u(t), \theta_u] \quad (57)$$

donde

$$\begin{aligned} \phi_y(t) &= [y(t-1), \dots, y(t-n)]^T \\ \phi_u(t) &= [u(t-d), \dots, u(t-d-m)]^T \end{aligned} \quad (58)$$

## Esquemas NNOE modificados

- Un esquema de estructura NNOE modificada que se ha propuesto es

$$\hat{y}(t|\theta) = [1 - A(q^{-1})]\hat{y}(t, \theta) + NN_g[\phi_u(t), \theta_u] \quad (59)$$

este esquema es interesante ya que resuelve los problemas con el desconocimiento de las características de estabilidad típicas de los modelos OE.

## Esquemas NNARMAX modificados

- Un esquema de estructura NNARMAX modificada que se ha propuesto es

$$\hat{y}(t|\theta) = NN_g[\phi_{uy}(t), \theta_{uy}] + C(q^{-1})\epsilon(t, \theta) \quad (60)$$

donde

$$\phi_{uy}(t) = [y(t-1), \dots, y(t-n), u(t-d), \dots, u(t-d-m)]^T \quad (61)$$

en este esquema ocurre algo similar al anterior, resuelve los problemas con el desconocimiento de las características de estabilidad típicas de los modelos ARMAX.

## Estructuras híbridas lineal-no lineal

- Es posible implementar estructuras que contengan una parte lineal como subconjunto del esquema (Narendra).
- Si suponemos en primer lugar sistemas SISO (controlables y observables) descritos por una ecuación de E/S de la forma:

$$y_p(k+1) = \sum_{i=0}^{n-1} \alpha_i y_p(k-i) + \sum_{j=0}^{m-1} \beta_j u(k-j) \quad (62)$$

donde los parámetros  $\alpha$  y  $\beta$  son constantes y desconocidos.

# Estructuras híbridas lineal-no lineal

Identificación en modo paralelo y serie-paralelo

- Es posible implementar dos estructuras de identificación diferentes cuya forma es:
  - Modo de identificación paralelo

$$\hat{y}_p(k+1) = \sum_{i=0}^{n-1} \hat{\alpha}_i \hat{y}_p(k-i) + \sum_{j=0}^{m-1} \hat{\beta}_j u(k-j) \quad (63)$$

- Modo de identificación serie-paralelo

$$\hat{y}_p(k+1) = \sum_{i=0}^{n-1} \hat{\alpha}_i y_p(k-i) + \sum_{j=0}^{m-1} \hat{\beta}_j u(k-j) \quad (64)$$

donde los parámetros  $\hat{\alpha}$  y  $\hat{\beta}$  son parámetros ajustables.



# Modelos híbridos lineal-no lineal

Sistemas SISO (serie-paralelo)

- Modelo tipo I

$$y_p(k+1) = \sum_{i=0}^{n-1} \alpha_i y_p(k-i) + g[u(k), u(k-1), \dots, u(k-m+1)] \quad (65)$$

- Modelo tipo II

$$y_p(k+1) = f[y_p(k), y_p(k-1), \dots, y_p(k-n+1)] + \sum_{i=0}^{m-1} \beta_i u(k-i) \quad (66)$$

# Modelos híbridos lineal-no lineal

Sistemas SISO (serie-paralelo)

- Modelo tipo III

$$y_p(k+1) = f[y_p(k), y_p(k-1), \dots, y(k-n+1)] \\ + g[u(k), u(k-1), \dots, u(k-m+1)] \quad (67)$$

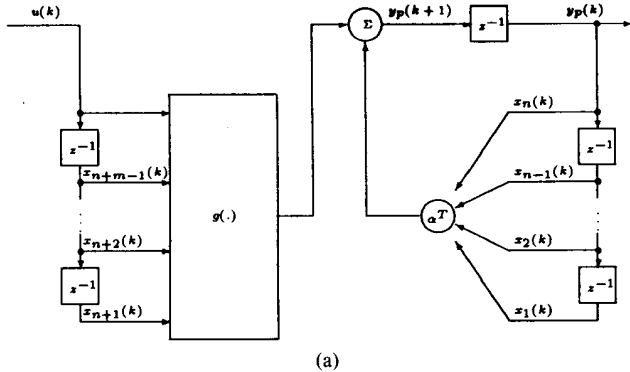
- Modelo tipo IV

$$y_p(k+1) = f[y_p(k), y_p(k-1), \dots, y(k-n+1), \\ u(k), u(k-1), \dots, u(k-m+1)] \quad (68)$$

# Modelos híbridos lineal-no lineal

Sistemas SISO

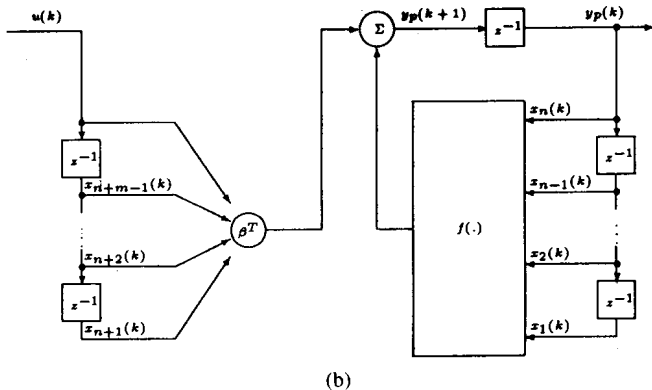
- Modelo tipo I



# Modelos híbridos lineal-no lineal

Sistemas SISO

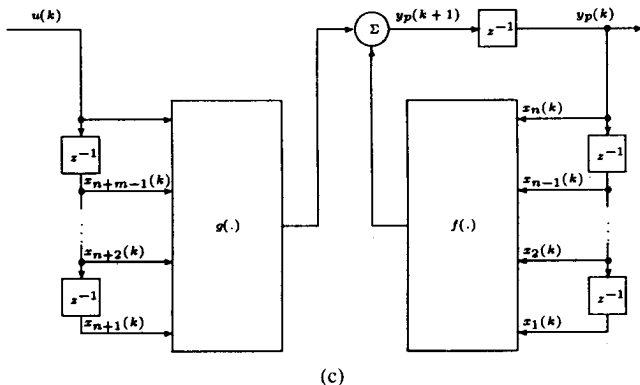
- Modelo tipo II



# Modelos híbridos lineal-no lineal

Sistemas SISO

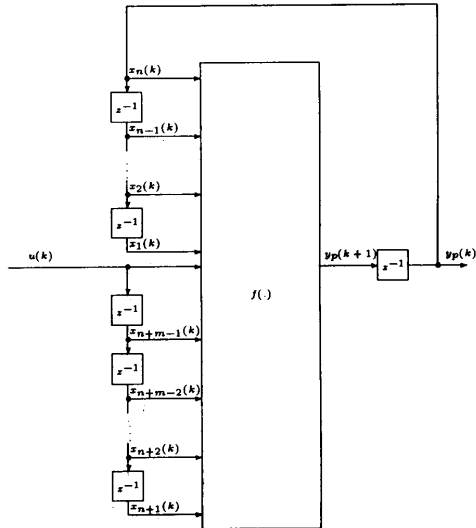
- Modelo tipo III



# Modelos híbridos lineal-no lineal

Sistemas SISO

- Modelo tipo IV



(d)

# Estructuras de redes neuronales para identificación de sistemas dinámicos

- En lo anterior se han planteado esquemas clásicos de red neuronal, que fundamentalmente utilizaban la red neuronal como un mapa estático y si era necesario se añadía una realimentación externa.
- Pero se han desarrollado redes neuronales específicas para estos tipos de problemas que incluyen algún tipo de realimentación interna a la red (lo que requiere algunas modificaciones en el aprendizaje).

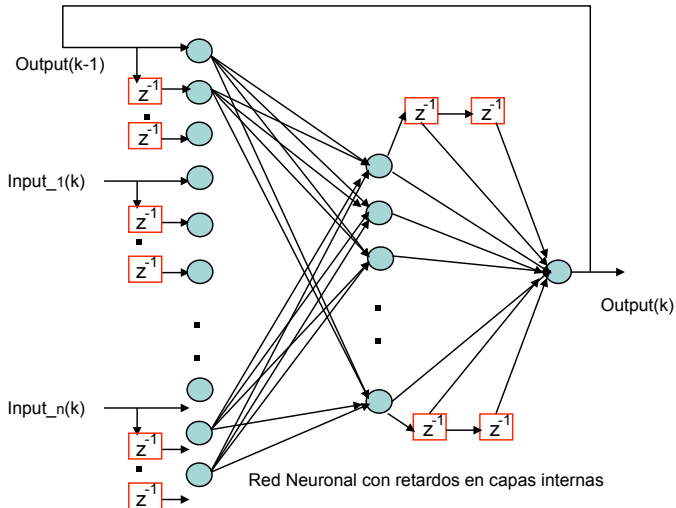
## Redes con representación del tiempo

Dentro de las redes con representación del tiempo utilizadas en aplicaciones de sistemas dinámicos podemos ver dos grandes grupos:

- Utilización de *arquitecturas de red dinámicas* para aplicaciones específicas.
  - Los retardos son introducidos en la capa de entrada, y cada neurona tiene información de un instante de tiempo.
  - Introducir los retardos en las neuronas de las capas intermedias.
  - Se pueden combinar ambas estrategias.
- *Modificación de estructuras y algoritmos de aprendizaje estáticos* para incluir de forma explícita el tiempo y poder modelar la dinámica del sistema.



# Redes con retardos en capas internas



## Redes con retardos

Dentro de las redes con retardos en capas internas tenemos:

- Redes TDNN (Time Delay Neural Networks).
  - Hopfield y Tank 1986: Toma de decisiones y Optimización.
  - Guyon y Albrecht 1990: Reconocimiento de caracteres escritos a mano.
- Redes Recurrentes.
  - Rumelhart, Hinton, y Williams 1986: Completar secuencias.
  - Robinson y Fallside 1988: Reconocimiento de voz.
  - Williams y Zipser 1989: Reconocimiento de caracteres.
- Memory Neuron Networks.
  - Poddar 1991: Reconocimiento de caracteres.

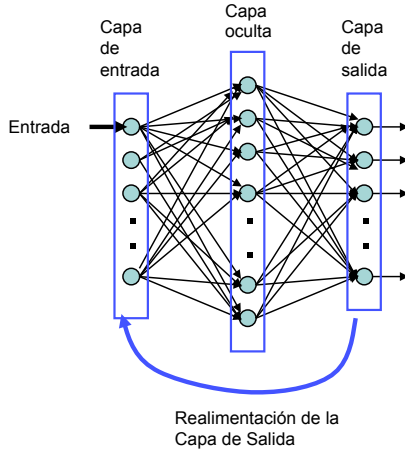
## Redes clásicas modificadas

Son redes clásicas y algoritmos de aprendizaje clásicos con modificaciones, que en su mayoría derivan del algoritmo de aprendizaje backpropagation :

- **Red de Elman** (1990), se realimenta en la entrada la capa oculta.
  - Reconoce secuencias de periodos cortos de tiempo. Elman la usó para reconocer palabras generadas por una gramática formal.
- **Red de Jordan**, se realimenta en la entrada la capa de salida.
  - Reconocimiento y clasificación de secuencias. Generación de trayectorias de robots.

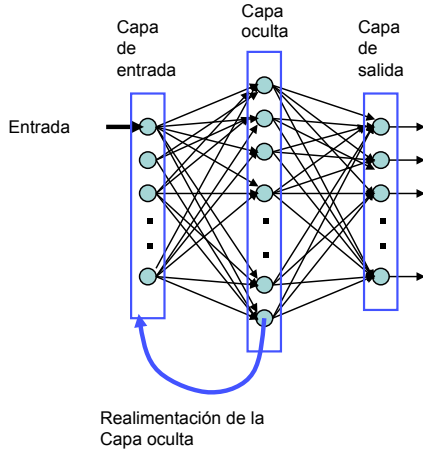
# Redes clásicas modificadas

- Red de Jordan



# Redes clásicas modificadas

- Red de Elman



# Algoritmos de entrenamiento para Redes clásicas modificadas

Son redes clásicas y algoritmos de aprendizaje clásicos con modificaciones, que en su mayoría derivan del algoritmo de aprendizaje backpropagation :

- **Backpropagation in Dynamic Systems (BPD)** (Narendra 1991).
  - Identificación y control por modelo de referencia.
- **Backpropagation Through Time (BTT)** (Werbo 1990).
  - Identificación de un Camión-remolque.

# Redes recurrentes

## Aprendizaje

- Cuando se interconectan varios bloques de redes neuronales feedforward usando conexiones realimentadas se obtienen lo que se denominan **redes recurrentes** o estructuras recurrentes.
- La conducta de estas redes *no puede describirse mediante un mapeo estático entre el espacio de entrada y salida*.
- En estas redes recurrentes la salida exhibe *conductas temporales complejas* que dependen de los estados actuales de las neuronas así como de las entradas.
- Mientras que el **algoritmo de backpropagation estática** o clásico nos permite entrenar las redes feedforward con mapeo estático, las redes recurrentes pueden ser entrenadas para emular un mapeo estático mediante lo que se denominará **algoritmo de backpropagation dinámica** que lo que busca es entrenar un red recurrente para seguir secuencias temporales.[NP90, NP91] .

# Identificación de sistemas dinámicos

- Tipos de representación de un sistema dinámico:
  - **Entrada/Salida**  
Es una relación que expresa la salida que da un sistema para una entrada
  - **Variables de estado**  
Son un conjunto de ecuaciones de 1er orden que expresan como varia el estado interno del sistema en función de las entradas que se le introducen, y un conjunto de ecuaciones de salida que indican que salida dará el sistema cuando está en un cierto estado.



## Identificación de modelos en el Espacio de Estados

- Si suponemos que el sistema está descrito por una ecuación de estado de la forma

$$\begin{aligned}x(k+1) &= f(x(k), u(k)) \\ y(k) &= h(x(k))\end{aligned}\tag{69}$$

una forma natural de identificar este modelo mediante redes neuronales consiste en usar redes neuronales que tengan la misma estructura que el modelo.

- Utilizando las propiedades de las redes feedforward para aproximar funciones, cada una de estas funciones  $f(\cdot)$  y  $h(\cdot)$  pueden ser aproximadas por una red neuronal multicapa con las dimensiones de las entradas y salidas apropiadas.

# Identificación de modelos en el Espacio de Estados

Estado del sistema medible

- Si suponemos que el estado del sistema es medible directamente, el modelo de identificación puede ser el siguiente:

$$\begin{aligned}x(k+1) &= NN_f(x(k), u(k)) \\ y(k) &= NN_h(x(k))\end{aligned}\tag{70}$$

donde  $NN_h$  y  $NN_f$  son mapas realizados mediante redes neuronales feedforward y pueden identificarse de forma independiente usando métodos de aprendizaje estáticos (LN93, Lev92).

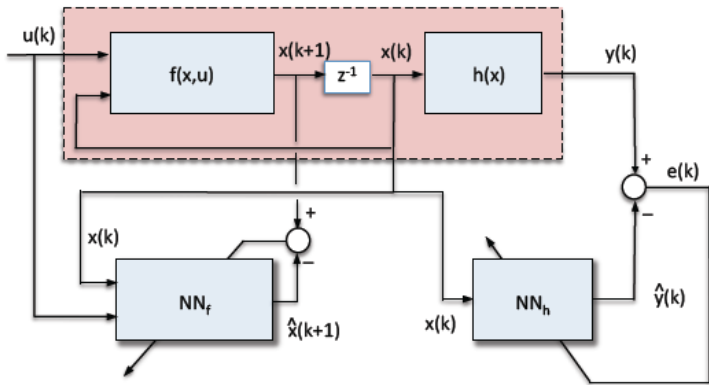
- Una vez entrenadas, los estados del modelo identificado dan una aproximación de los estados del sistema.

# Identificación de modelos en el Espacio de Estados

## Estados

Estado del sistema medible

- Identificación de un modelo de estado mediante redes neuronales.



# Identificación de modelos en el Espacio de Estados

Estado del sistema no medible

- Si el estado del sistema no es medible o accesible el problema se hace mucho más difícil y no es posible obtener un estimador  $\hat{x}(k)$  del estado  $x(k)$ .
- En este caso el modelo de identificación tiene la forma

$$\begin{aligned}x(k+1) &= NN_f(x(k), z(k)) \\ \hat{y}(k) &= NN_h(z(k))\end{aligned}\tag{71}$$

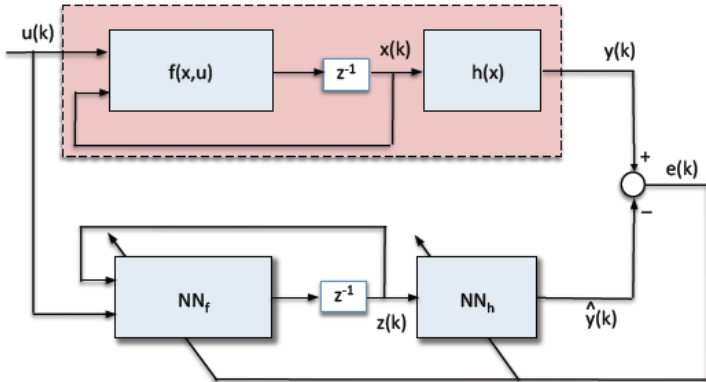
donde  $NN_h$  y  $NN_f$  son mapas realizados mediante redes neuronales feedforward (ver transparencia siguiente).

- Este modelo es una representación equivalente del sistema 69 y su estado  $z(k) = [z_1(k), z_2(k), \dots, z_n(k)]$  está relacionado con el estado del sistema  $x(k)$  por un difeomorfismo.

# Identificación de modelos en el Espacio de Estados

Estado del sistema no medible

- Identificación de un modelo de estado mediante redes neuronales.



## Criterio de optimización

- Un criterio natural de optimización para el ajuste del modelo sería la suma de los cuadrados de los errores entre las salidas del sistema y del modelo:

$$I(k) = \sum_{k=0}^K \|y(k) - \hat{y}(k)\|^2 = \sum_{k=0}^K \|e(k)\|^2 \quad (72)$$

- Dado que  $x(k)$  no es accesible y el error puede medirse solo en la salida, **las redes no pueden ser entrenadas por separado**.
- Dado que el modelo contiene un **lazo de realimentación**, el gradiente del criterio de prestaciones con respecto a los pesos de  $NN_f$  varía con el tiempo, y por tanto debe de usarse una **backpropagation dinámica** .

## Ajuste de los parámetros de $NN_f$

- Si  $\theta \in \Theta(NN_f)$  denota un parámetro de  $NN_f$ . El gradiente del criterio a optimizar con respecto a  $\theta$  se obtiene como

$$\frac{dI(k)}{d\theta} = -2 \sum_{k=0}^{\kappa} [y(k) - \hat{y}(k)] \frac{d\hat{y}(k)}{d\theta}, \quad (73)$$

$$\frac{d\hat{y}(k)}{d\theta} = \sum_{j=1}^n \frac{\partial \hat{y}(k)}{\partial z_j(k)} \frac{dz_j(k)}{d\theta}, \quad (74)$$

$$\frac{dz_j(k)}{d\theta} = \sum_{l=1}^n \frac{\partial z_j(k)}{\partial z_l(k-1)} \frac{dz_l(k-1)}{d\theta} + \frac{\partial z_j(k)}{\partial \theta} \quad (75)$$

## Ajuste de los parámetros de $NN_f$

- Por lo tanto el gradiente de la salida con respecto a  $\theta$  viene dado por la salida del sistema lineal

$$\frac{dz(k+1)}{d\theta} = A \frac{dz(k)}{d\theta} + b \frac{\partial z(k)}{\partial \theta}, \quad (76)$$

$$\frac{d\hat{y}(k)}{d\theta} = c^T \frac{dz(k)}{d\theta} \quad (77)$$

- donde:

- $\frac{dz(k)}{d\theta}$  es el vector de estado,
- $\frac{\partial z(k)}{\partial \theta}$  es la entrada,
- y  $A, b, c$  están definidos por  $a_{ij} = \frac{\partial z_i(k+1)}{\partial z_j(k)}$ ,  $b_i = 1$  y  $c_i = \frac{\partial \hat{y}(k)}{\partial z_i(k)}$ .
- Las condiciones iniciales de los estados se ponen a cero.



## Ajuste de los parámetros de $NN_h$

- Dado que la red  $NN_h$  está conectada directamente con la salida, sin lazo de realimentación, el gradiente del error con respecto a sus parámetros es calculado usando **backpropagation estática**. esto puede hacerse on-line o en modo batch en el cual el error sobre un número finito de pasos es sumado antes de actualizar los pesos.

# Identificación de un modelo de estado

## Observaciones

- Por diseño, el modelo no es único, y por lo tanto el estado del modelo (después de que se consiga la identificación) viene dado por  $z = \phi(x)$  y las redes neuronales convergen a una transformación de las funciones del sistema:

$$NN_h(.) \simeq h(.) \circ \phi^{-1}, \quad (78)$$

$$NN_f(.,.) \simeq \phi \circ f(\phi^{-1}(.),.); \quad (79)$$

donde  $\phi : \mathbb{R}^n \rightarrow \mathbb{R}^n$  es continua e invertible.

- Si el sistema puede ser reinicializado a voluntad del diseñador a un estado inicial (que podemos suponer el cero por simplicidad) el entrenamiento se hace más fácilmente tratable y la relación funcional  $\phi$  entre los estados del sistema y el modelo se obtienen de forma más natural.

# Identificación de un modelo de estado

Ejemplo simulado

- Supongamos el modelo

$$\begin{aligned}x_1(k+1) &= x_2(k)[1 + 0,2u(k)], \\x_2(k+1) &= -0,2x_1(k) + 0,5x_2(k) + u(k), \\y(k) &= 0,3[x_1(k) + 2x_2(k)]^2.\end{aligned}\tag{80}$$

- Un posible modelo basado en redes neuronales para identificar este sistema podría ser

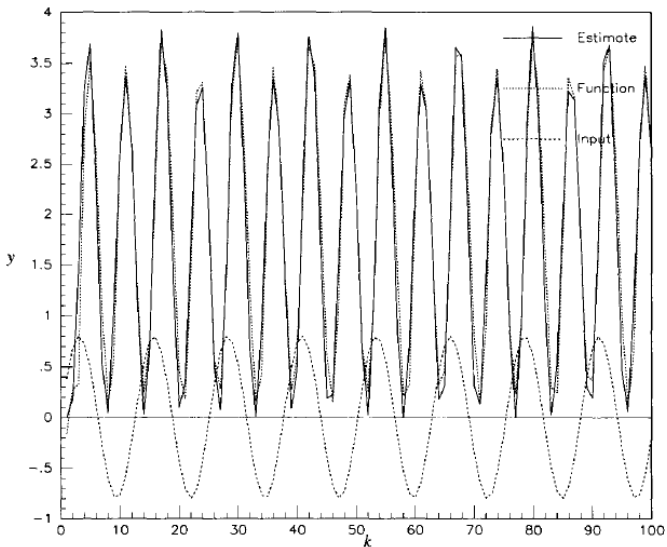
$$\begin{aligned}\hat{x}_1(k+1) &= NN_{f_1}[\hat{x}_1(k), \hat{x}_2(k), u(k)], \\ \hat{x}_2(k+1) &= NN_{f_2}[\hat{x}_1(k), \hat{x}_2(k), u(k)], \\ \hat{y}_1(k) &= NN_h[\hat{x}_1(k), \hat{x}_2(k)].\end{aligned}\tag{81}$$

- Se ha usado una red distinta para estimar cada una de las tres funciones no lineales  $f_1, f_2, h$ .

# Identificación de un modelo de estado

Ejemplo simulado

- Resultado



# Identificación de modelos de estado

## Inconvenientes

- En la identificación de modelos en el espacio de estados es necesario:
  - El uso de backpropagation dinámica, que es **computacionalmente muy costosa**.
  - El uso de ganancias pequeñas para ajustar los parámetros lo que da lugar a **tiempos muy largos** hasta alcanzar la convergencia.
- Una posible alternativa consiste en tratar de utilizar modelos de Entrada-Salida, donde se trata de determinar la salida futura del sistema como función de las observaciones de entradas y salidas en instantes anteriores.

## Identificación de modelos de Entrada-Salida

- Es decir si existe un número  $l$  y una función continua  $\tilde{h} : \mathcal{Y}_l \times \mathcal{U}_l \rightarrow \mathcal{Y}$  tal que el modelo recursivo

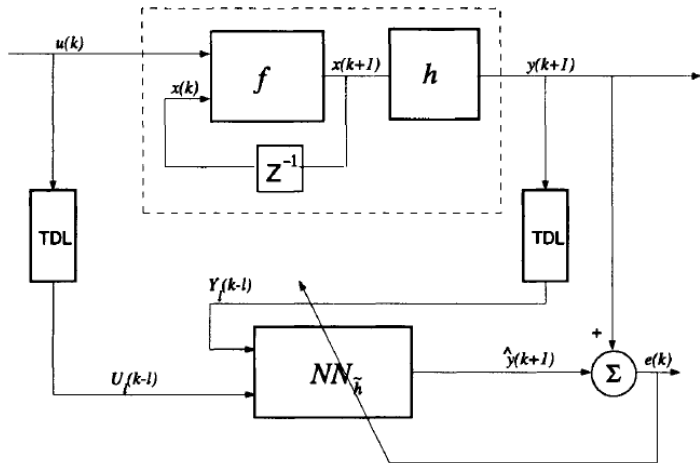
$$y(k+1) = \tilde{h}[Y_l(k-l+1), U_l(k-l+1)] \quad (82)$$

tiene la misma entrada-salida que el sistema original, entonces el modelo de identificación puede realizarse mediante una red neuronal con  $2l$  entradas y una salida (ver figura de la transparencia siguiente).

- Dado que tanto la entrada como la salida a la red son observables directamente en cada instante de tiempo, puede usarse **backpropagation estática** para entrenar la red.

# Identificación de modelos de Entrada-Salida

- Modelo de Entrada-Salida para identificación.



TDL: Tapped delay line (línea de retardo).

## Identificación de modelos de Entrada-Salida

- Para el caso de sistemas lineales siempre existe el modelo de Entrada-Salida.
- Para estos modelos la relación entre entrada y salida puede modelarse recursivamente en la forma

$$y(k) = \sum_{i=1}^n a_i y(k-i) + \sum_{i=1}^n b_i u(k-i) \quad (83)$$

- Aunque estos modelos de entrada-salida han sido propuestos para la identificación de sistemas dinámicos no lineales en la literatura [Jor86, NP90], no resulta obvio que estos modelos existan para sistemas generales de la forma

$$\begin{aligned} x(k+1) &= f(x(k), u(k)) \\ y(k) &= h(x(k)) \end{aligned} \quad (84)$$



## Modelos de Entrada-Salida locales

- Consideremos en primer lugar el problema de establecer un *modelo de entrada-salida local alrededor de un punto o estado de equilibrio* del sistema (que denominaremos origen).
- El problema se podría formular del siguiente modo:  
*Supuesto que el origen es un punto de equilibrio, existe una región  $\Omega_x$  alrededor del origen tal que sí  $x(k) \in \Omega_x$ , la salida del del sistema en el instante  $k$  está determinada de forma única por una función de un número finito de las observaciones de entrada y salida previas?*
- Se demostrará que esto se cumple si el sistema es **localmente fuertemente observable** (locally strongly observable) en la región  $\Omega_x$ .

# Modelos de Entrada-Salida Locales

## Condiciones para la observabilidad local fuerte

- Las condiciones para la observabilidad local fuerte de un sistema  $\Sigma$  alrededor del origen pueden derivarse de las propiedades de observabilidad de su linealización en el origen  $\Sigma_L$ :

$$\begin{aligned}\delta x(k+1) &= f_x|_{0,0}\delta x(k) + f_u|_{0,0}\delta u(k) \\ &= A\delta x(k) + b\delta u(k),\end{aligned}\tag{85}$$

$$\begin{aligned}\delta y(k) &= h_x|_0\delta x(k) \\ &= c^T\delta x(k),\end{aligned}\tag{86}$$

donde  $A = f_x|_{0,0}$ ,  $b = f_u|_{0,0}$  y  $c^T = h_x|_0$ .

# Modelos de Entrada-Salida Locales

Condiciones para la observabilidad local fuerte

## Teorema

Sea  $\Sigma$  el sistema no lineal

$$\begin{aligned}x(k+1) &= f(x(k), u(k)) \\ y(k) &= h(x(k))\end{aligned}\tag{87}$$

y  $\Sigma_L$  su linealización alrededor de un punto de equilibrio. Si  $\Sigma_L$  es observable, entonces  $\Sigma$  es localmente fuertemente observable. Además, localmente,  $\Sigma$  puede ser realizado o aproximado por un modelo de entrada-salida.

El teorema nos dice que la existencia de un modelo de entrada-salida para un sistema no lineal puede ser determinada por la simple observación de las propiedades de observabilidad del sistema linealizado subyacente.

## Modelos de Entrada-Salida Locales

Ejemplo

- Dado el sistema:

$$\begin{aligned}x(k+1) &= x(k) + u(k) \\ y(k) &= x(k) + x^2(k)\end{aligned}\tag{88}$$

(89)

con

$$\frac{\partial y(k)}{\partial x(k)} = 2x + 1, \quad \frac{\partial y(k)}{\partial x(k)} \Big|_{(0,0)} = 1\tag{90}$$

Por lo tanto el sistema linealizado en el origen ( $x = 0, u = 0$ ) es observable y en un entorno alrededor del origen hay una representación de entrada-salida para el sistema anterior dada por

$$\begin{aligned}y(k) &= x(k+1) + x^2(k+1) \\ &= x(k) + u(k) + x^2(k) + u^2(k) + 2x(k)u(k) \\ &= y(k) + u^2(k) + 2u(k)\sqrt{1 + 4y(k)}\end{aligned}\tag{91}$$

## Realización mediante Redes Neuronales

- Si se sabe (o se asume) que se verifican las condiciones de observabilidad fuerte para el sistema en una cierta región de funcionamiento, entonces el procedimiento de identificación usando una red neuronal feedforward es casi inmediato.
- En cada instante de tiempo, las entradas de la red (ojo que no son las entradas del sistema) consisten en los  $n$  valores de las entradas pasadas al sistema y los  $n$  valores pasados de las salidas del sistema consisting of the system's (en total  $2n$ ), se introducen como entradas de la red neuronal (el orden del sistema se supone conocido  $n$  o se usa una estimación del mismo).
- Las salidas de la red se comparan con la siguiente observación de la salida del sistema y se obtiene el error

$$e(k+1) = y(k+1) - NN[Y_n(k-n+1), U_n(k-n+1)]. \quad (92)$$

- Los pesos de la red se ajustan usando **backpropagation estática** para minimizar la suma cuadrática de los errores.

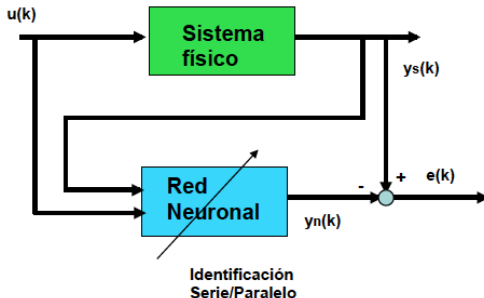
# Realización mediante Redes Neuronales

Una vez que se realiza la identificación, dos formas de operación son posibles:

- **Modo serie-paralelo:**
- **Modo paralelo:**

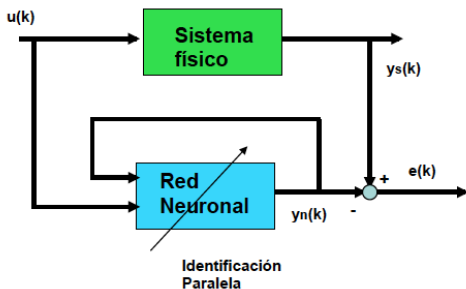
## Modo Serie-Paralelo

- Este esquema utiliza las entradas y salidas de la planta y deja que la red deduzca el error existente.
- También se llama teacher forcing y puede usarse con cualquier algoritmo de aprendizaje.



## Modo paralelo

- En este esquema la red neuronal no tiene información directa de las salidas de la planta, solo de las entradas y del error de estimación entre la salida de la red y la de la planta.



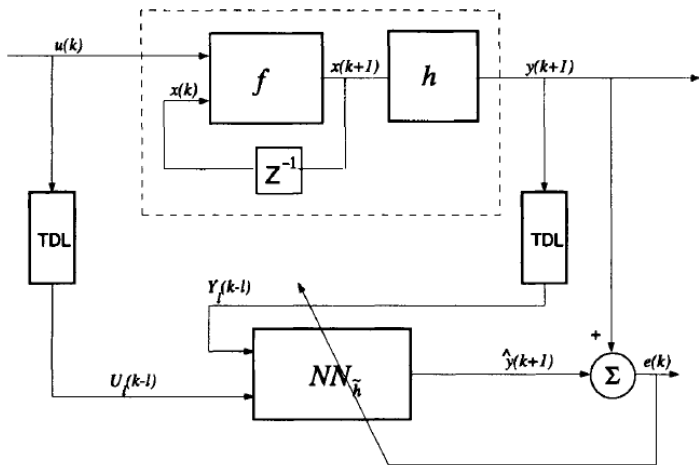


## Operación en Modo Serie-Paralelo

- En este modo, las salidas reales del sistema se usan como entradas en el modelo.
- Este esquema puede ser usado solo en conjunción con el sistema, y solo puede generar la predicción para el próximo ciclo por adelantado (one-step-ahead prediction).
- La arquitectura es idéntica a la usada para la identificación y vista anteriormente.

## Operación en Modo Serie-Paralelo

- Identificación en Modo Serie-Paralelo



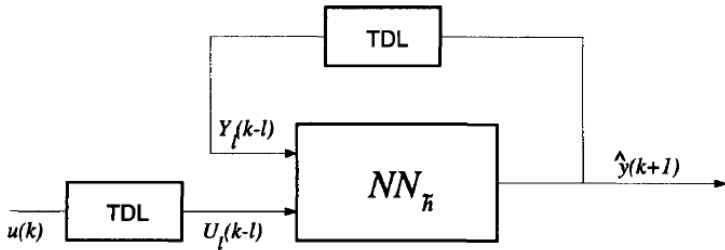
TDL: Tapped delay line (línea de retardo).

## Operación en Modo Paralelo

- Cuando es necesaria una predicción de más instantes de tiempo, es necesario usar el modo independiente o paralelo.
- En este esquema, la salida de la red es realimentada a la entrada de la red (ver transparencia siguiente), es decir, las salidas de la propia red son usadas para generar predicciones futuras.
- Como el modelo no es perfecto, este modo de operación genera una forma viable de hacer predicciones a corto plazo ( $> 1$  ciclos).
- En muchos casos el objetivo no es hacer predicciones específicas concernientes al sistema sino entrenar la red para generar trayectorias temporales complejas. En este caso si la identificación es precisa el modelo exhibirá el mismo tipo de conducta (en un sentido topológico) que el sistema original.

## Operación en Modo Paralelo

- Identificación en Modo Paralelo



TDL: Tapped delay line (línea de retardo).

## Identificación local con modelo entrada-salida Ejemplo

- Sea el sistema

$$\begin{aligned}x_1(k+1) &= 0,25x_2(k) + 0,2x_1(k)x_2(k) \\x_2(k+1) &= -0,3x_1(k) + 0,8x_2(k) + u(k) \\y(k) &= x_1(k) + [x_2(k)]^2\end{aligned}\tag{93}$$

(94)

- El sistema linealizado en el equilibrio es

$$\begin{aligned}\delta x_1(k) &= 0,5\delta x_2(k) \\ \delta x_2(k) &= -0,3\delta x_1(k) + 0,8\delta x_2(k) + \delta u(k) \\ \delta y(k) &= \delta x_1(k)\end{aligned}\tag{95}$$

su matriz de observabilidad es:

$$M_O = [ c | cA ] = \begin{bmatrix} 1 & 0 \\ 0 & 0,5 \end{bmatrix}$$

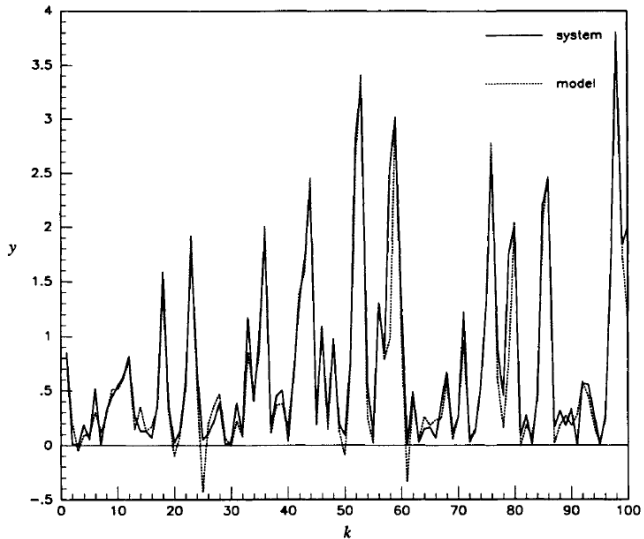
# Identificación local con modelo entrada-salida

Ejemplo

- Como el rango de la  $M_O$  es 2 el sistema puede ser aproximado por un modelo de entrada-salida de orden 2.
- Se entrena una red  $NN_{\tilde{h}} \in \mathcal{N}_{4,12,6,1}^3$  para implementar el modelo de entrada-salida.
- El sistema se excita con una entrada aleatoria  $u(k) \in [-1, 1]$ .
- Las entradas a la red en cada instante de tiempo consisten en  $y(k), y(k-1), u(k), u(k-1)$ , y la salida de la red es  $\hat{y}(k+1)$  es comparada con la salida del sistema  $y(k+1)$ .
- El error  $e(k+1) = y(k+1) - \hat{y}(k+1)$  se ha usado para ajustar la red mediante backpropagation estática a lo largo
- El resultado después de 20000 pasos de entrenamiento se muestra en la transparencia siguiente. (para una secuencia de entrada aleatoria).

# Ejemplo

- Identificación con modelo de entrada-salida

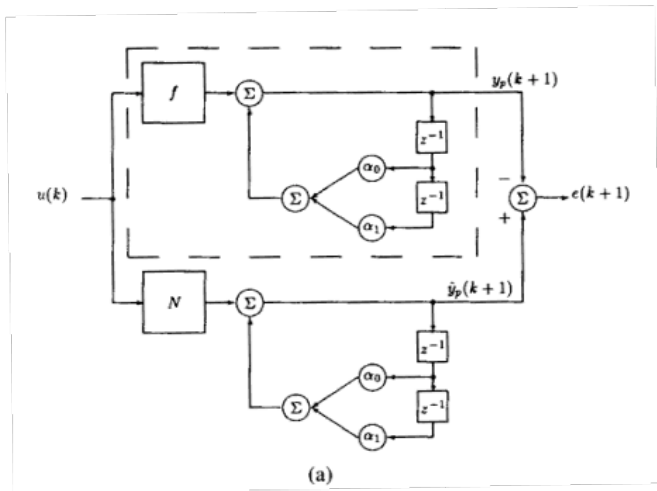


# Identificación en modelos híbridos

Identif en paralelo: modelo tipo I

- Para un modelo SISO tipo 1 con  $n = 2$  y  $m = 1$

$$\hat{y}_p(k+1) = \hat{\alpha}_0 \hat{y}_p(k) + \hat{\alpha}_1 \hat{y}_p(k-1) + N[u(k)] \quad (96)$$





# Identificación en modelos híbridos

Identif en paralelo: modelo tipo I

- Para el modelo SISO tipo 1 exterior

$$\hat{y}_p(k+1) = \hat{\alpha}_0 \hat{y}_p(k) + \hat{\alpha}_1 \hat{y}_p(k-1) + N[u(k)] \quad (97)$$

- Es necesario estimar:
  - ①  $\hat{\alpha}_i$
  - ② Los pesos de la red neuronal N, en base al error  $e(k)$  entre la salida del modelo  $\hat{y}_p(k)$  y la salida del sistema real  $y_p(k)$ .
  - ③ Los errores se retropropagan mediante backpropagation.

# Identificación en modelos híbridos

Identif en paralelo: modelo tipo I

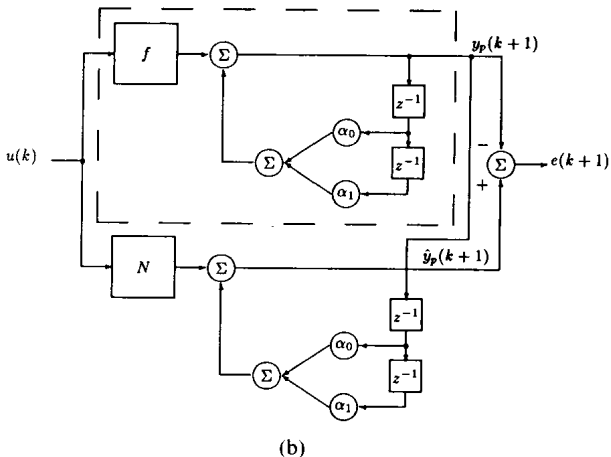
- No hay garantía de que los parámetros converjan
- Ni de que el error tienda a cero
- Las condiciones necesarias para garantizar que un sistema lineal converja no están aún bien establecidas
- No es por lo tanto un modelo conveniente, se suele preferir el modelo serie-paralelo para las plantas tipo I-IV

# Identificación en modelos híbridos

Identif serie-paralelo: modelo tipo I

- Para el modelo SISO tipo 1 anteriormente visto con  $n = 2$  y  $m = 1$

$$\hat{y}_p(k+1) = \hat{\alpha}_0 y_p(k) + \hat{\alpha}_1 y_p(k-1) + N[u(k)] \quad (98)$$



# Identif serie-paralelo: modelo tipo I

## Ejemplo

- Supongamos que el modelo de la planta tiene las siguientes ecuaciones

$$y_p(k+1) = 0,3y_p(k) + 0,6y_p(k-1) + f[u(k)] \quad (99)$$

$$f[u] = 0,6 \sin(\pi u) + 0,3 \sin(3\pi u) + 0,1 \sin(5\pi u) \quad (100)$$

- El modelo que se usará para identificarlo es de la forma

$$y_p(k+1) = 0,3y_p(k) + 0,6y_p(k-1) + NN_f[u(k)] \quad (101)$$

$$NN_f[u] \sim N_{1,20,10,1}^3 \quad (102)$$

- La entrada a la planta y al modelo es:

$$u(k) = \sin(2\pi k/250) \quad (103)$$

## Identif serie-paralelo: modelo tipo II

Ejemplo

- Supongamos ahora que el modelo de la planta tiene las siguientes ecuaciones

$$y_p(k+1) = f[y_p(k), y_p(k-1)] + u(k) \quad (104)$$

$$f[y_p(k), y_p(k-1)] = \frac{y_p(k)y_p(k-1)[y_p(k) + 2,5]}{1 + y_p^2(k) + y_p^2(k-1)} \quad (105)$$

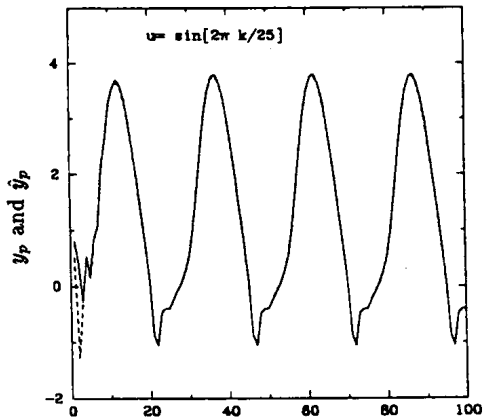
- El modelo que se usará para identificarlo es de la forma

$$y_p(k+1) = NN_f[y_p(k), y_p(k-1)] + u(k) \quad (106)$$

$$NN_f[u] \sim N_{2,20,10,1}^3 \quad (107)$$

## Identif serie-paralelo: modelo tipo II

Ejemplo



Example 2: Outputs of the plant and the identification model.

## Identif serie-paralelo: modelo tipo III

Ejemplo

- Supongamos que el modelo de la planta tiene las siguientes ecuaciones

$$y_p(k+1) = \frac{y_p(k)}{1 + y_p^2(k-1)} + u^3(k) \quad (108)$$

$$f[y_p(k)] = \frac{y_p(k)}{1 + y_p^2(k-1)} \quad (109)$$

$$g[u(k)] = u^3(k) \quad (110)$$

- El modelo que se usará para identificarlo es de la forma

$$y_p(k+1) = NN_f[y_p(k)] + NN_g[u(k)] \quad (111)$$

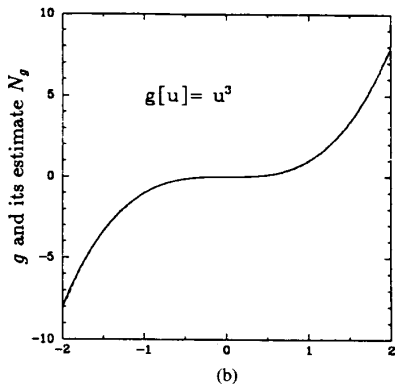
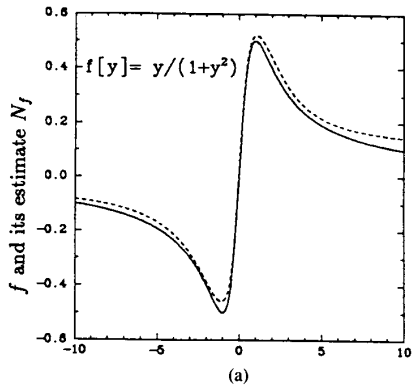
$$NN_f[u] \sim N_{1,20,10,1}^3 \quad (112)$$

- La entrada es

$$u(k) = \sin(2\pi k/25) + \sin(2\pi k/10)$$

# Identif serie-paralelo: modelo tipo III

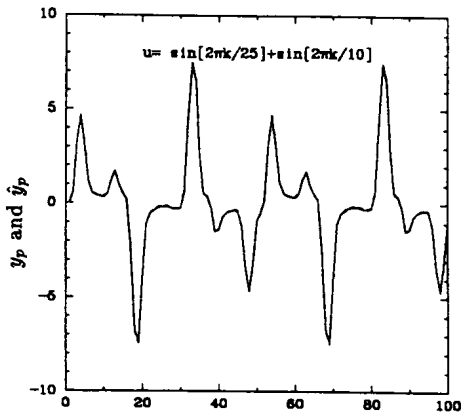
Ejemplo





# Identif serie-paralelo: modelo tipo III

Ejemplo

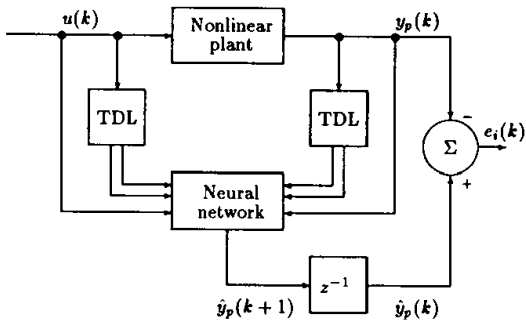


(c)

# Identificación en modelos híbridos

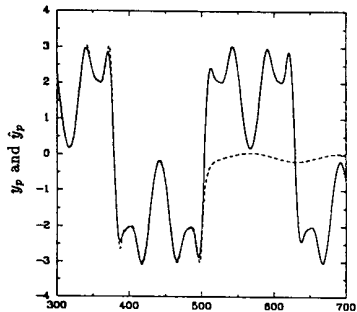
Identif serie-paralelo: modelo tipo IV

- La planta es completamente no-lineal

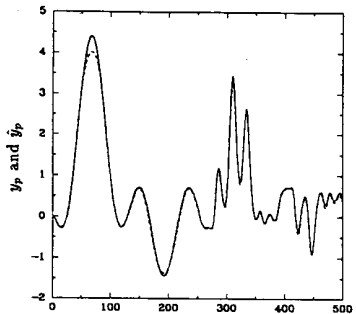


# Identificación en modelos híbridos

Identif serie-paralelo: modelo tipo IV



(a)



(b)

Fig. 12. Example 1: (a) Outputs of the plant and identification model when adaptation stops at  $k = 500$ . (b) Response of plant and identification model after identification using a random input.

- No funciona bien ya que se excita el sistema con una sola frecuencia

# Identificación en modelos híbridos

Identif serie-paralelo: modelo tipo IV

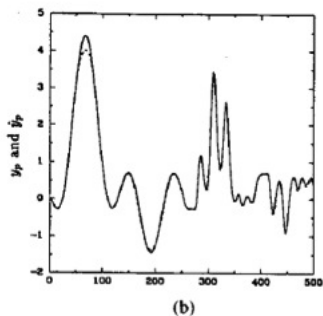
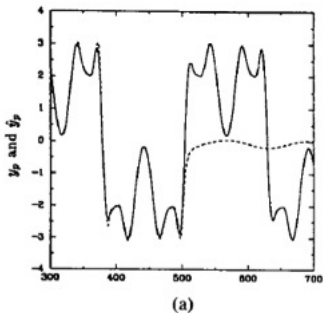








Fig. 12. Example 1: (a) Outputs of the plant and identification model when adaptation stops at  $k = 500$ . (b) Response of plant and identification model after identification using a random input.

- Al excitar con una señal aleatoria se excita correctamente al sistema y la identificación es correcta.

# Bibliografía

-  Ljung, L. (1999). System identification. Wiley Encyclopedia of Electrical and Electronics Engineering.
-  Ljung, L. (1995). System identification toolbox: User's guide. Natick, MA: MathWorks Incorporated.
-  Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. Mathematics of control, signals and systems, 303-314.
-  Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. Neural networks, 359-366.
-  Sánchez Camperos, E. N., Alanís García, A. Y. (2006). Redes Neuronales: Conceptos fundamentales y aplicaciones a control automático. Edgar Nelson Sánchez Camperos y Alma Yolanda Alanís García.
-  Narendra, K. S., & Mukhopadhyay, S. (1992). Intelligent control using neural networks. IEEE Control systems magazine, 11-18.

- Fin L9