Modelo de red FIR (Respuesta Finita al Impulso)

Neurona estándar : $x_j^{\ell+1} = f\left(\sum_i w_{ij}^{\ell} x_i^{\ell}\right)$

donde ℓ es el índice de la capa.

Neurona FIR : $x_j^{\ell+1}(k) = f\left(\sum_i \vec{w}_{ij}^{\ell} \bullet \vec{x}_i^{\ell}(k)\right)$

donde

$$\vec{x}_i^{\ell}(k) = \left[x_i^{\ell}(k), x_i^{\ell}(k-1), \ldots, x_i^{\ell}(k-T^{\ell})\right]$$
$$\vec{w}_{ij}^{\ell}(k) = \left[w_{ij}^{\ell}(0), w_{ij}^{\ell}(1), \ldots, w_{ij}^{\ell}(T^{\ell})\right]$$

Se reemplazan los pesos sinápticos estáticos por filtros lineales FIR del tipo línea de retardos

$$\sum_{n=0}^{T^{\ell}} w_{ij}^{\ell}(n) x_i^{\ell}(k-n) .$$

En comparación con BP estándar los escalares se reemplazan por vectores y las multiplicaciones por productos puntos.

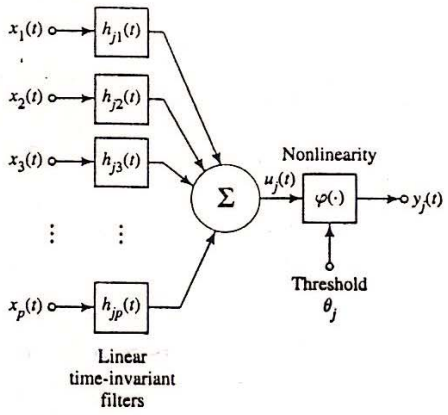
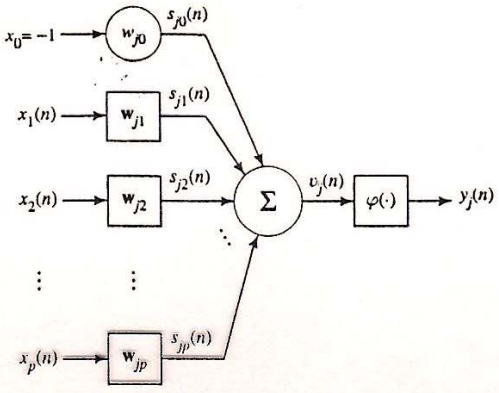
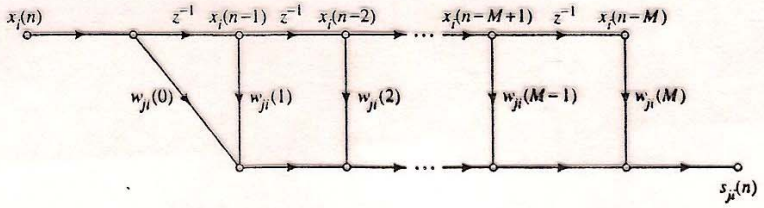


FIGURE 13.3 Dynamic model of a neuron using linear, time-invariant, low-pass filters as a synapses.



(a)



(b)

FIGURE 13.4 (a) Dynamic model of a neuron, incorporating synaptic FIR filters. (b) Signal-flow graph of a synaptic FIR filter.

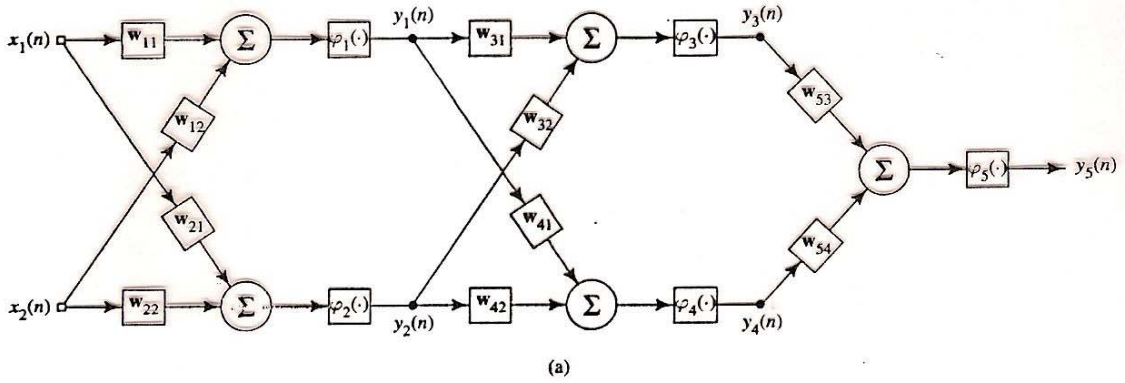
Representaciones alternativas de topología FIR

Se puede demostrar que las redes FIR y TDNN son funcionalmente equivalentes. La red TDNN se describe como una red en capas en que las salidas de una capa se almacenan por varios pasos de tiempo y luego se alimentan a la capa siguiente.

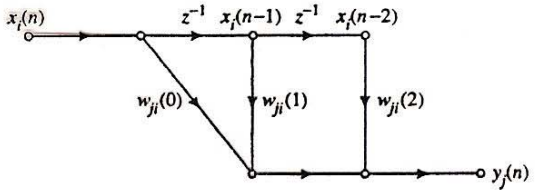
La red FIR se puede "desenrrollar" en el tiempo. Las líneas de retardos se interpretan como neuronas virtuales cuyas entradas se retrasan apropiadamente. Luego se eliminan los retardos replicando las capas previas de la red. Este método produce una estructura estática equivalente donde las dependencias del tiempo se hacen externas.

La red FIR es una representación compacta de una red estática más grande con simetrías impuestas (redundancias en los pesos). Estas simetrías fuerzan la red a subdividir el espacio de entradas en regiones locales que se traslapan.

Se puede aplicar BP a la estructura estática equivalente que resulta del desenrrollado en el tiempo. Pero ya que la red contiene pesos duplicados los términos del gradiente deben ser cuidadosamente recombinados para encontrar el gradiente total de un peso dado.



(a)



(b)

FIGURE 13.6 (a) Architectural flow graph of 2-2-2-1 FIR multilayer perceptron. (b) FIR model of a synaptic filter.

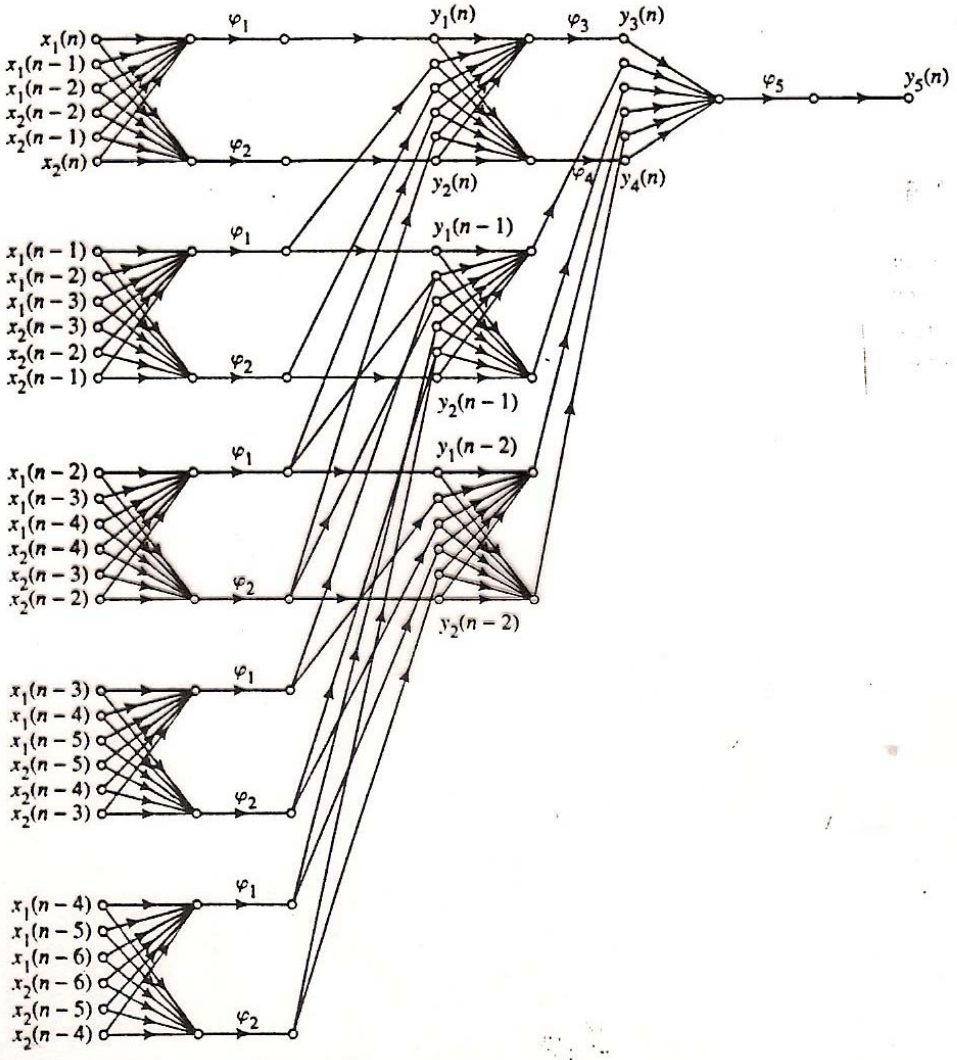


FIGURE 13.7 Signal-flow graph of 2-2-2-1 FIR multilayer perceptron that has been unfolded in time, starting from its input end.

Backpropagation Temporal

Se desea asociar una secuencia de entradas x(k) a una secuencias de salidas deseadas d(k), k=1,...,K. Sea y(k) la salida del modelo de red neuronal, $y(k) = \Im(W, x(k))$.

El error instantáneo es $e^2(k) = \left\| d(k) - \Im(W, x(k)) \right\|^2$

y el error total sobre todos los elementos de la secuencia es

$$E = \sum_{k=1}^{K} e^2(k)$$

El gradiente se calcula como

$$\frac{\partial E}{\partial \vec{w}_{ij}^{\ell}} = \sum_k \frac{\partial E}{\partial s_j^{\ell+1}(k)} \frac{\partial s_j^{\ell+1}(k)}{\partial \vec{w}_{ij}^{\ell}} \qquad (1)$$

donde

$$s_j^{\ell+1}(k) = \sum_i \vec{w}_{ij}^{\ell} \bullet \vec{x}_i^{\ell}(k)$$

es la salida lineal de la neurona j-ésima.

Notar que (1) difiere del enfoque tradicional en que se expresa el gradiente total como la suma de los gradientes instantáneos :

$$\frac{\partial E}{\partial s_j^{\ell+1}(k)} \frac{\partial s_j^{\ell+1}(k)}{\partial \vec{w}_{ij}^{\ell}} \neq \frac{\partial e^2(k)}{\partial \vec{w}_{ij}^{\ell}}$$

Sólo la suma sobre todo k es equivalente.

La primera derivada en el lado derecho de (1) son los "deltas", i.e.

$$\delta_j^{\ell+1}(k) = \frac{\partial E}{\partial s_j^{\ell+1}(k)} \quad (2)$$

La segunda derivada en el lado derecho de (1) toma la expresión

$$\frac{\partial s_j^{\ell+1}(k)}{\partial \vec{w}_{ij}^{\ell}} = \vec{x}_i^{\ell}(k) \qquad (3)$$

Reemplazando (2) y (3) en (1), y usando el algoritmo estocástico incremental se tiene :

$$\vec{w}_{ij}^{\ell}(k+1) = \vec{w}_{ij}^{\ell}(k) - \mu\delta_j^{\ell+1}(k)\vec{x}_i^{\ell}(k)$$

Falta encontrar una fórmula explícita para el cálculo de los deltas. Para la capa de salida se tiene :

$$\delta_j^L(k) = \frac{\partial E}{\partial s_j^L(k)} = \frac{\partial e^2(k)}{\partial s_j^L(k)} = -2e_j(k)sgm'(s_j^L(k))$$

donde e_j es el error en la j-ésima neurona de salida.

Para las capas ocultas se usa la regla de la cadena, expandiendo sobre todo el tiempo y sobre todas las $N_{\ell+1}$ entradas $s^{\ell+1}(k)$ en la próxima capa :

$$\delta_j^{\ell}(k) = \frac{\partial E}{\partial s_j^{\ell}(k)} = \sum_{m=1}^{N_{\ell+1}} \sum_t \frac{\partial E}{\partial s_m^{\ell+1}(t)} \frac{\partial s_m^{\ell+1}(t)}{\partial s_j^{\ell}(k)}$$

$$= \sum_{m=1}^{N_{\ell+1}} \sum_t \delta_m^{\ell+1}(t) \frac{\partial s_m^{\ell+1}(t)}{\partial s_j^{\ell}(k)} = sgm'(s_j^{\ell}(k)) \sum_{m=1}^{N_{\ell+1}} \sum_t \delta_m^{\ell+1}(t) \frac{\partial s_{jm}^{\ell+1}(t)}{\partial x_j^{\ell}(k)} \qquad (4)$$

Recordando que $$s_{jm}^{\ell+1}(t) = \sum_{k'=0}^{T^{\ell}} w_{jm}^{\ell}(k') x_j^{\ell}(t-k')$$

se obtiene

$$\frac{\partial s_{jm}^{\ell+1}(t)}{\partial x_j^{\ell}(k)} = \begin{cases} w_{jm}^{\ell}(t-k) & \text{para } 0 \le \text{t - k} \le \text{T}^{\ell} \\ 0 & \text{otro caso} \end{cases} \qquad (5)$$

Reemplazando (5) en (4) se obtiene

$$\delta_j^{\ell}(k) = sgm'(s_j^{\ell}(k)) \sum_{m=1}^{N_{\ell+1}} \sum_{t=k}^{T^{\ell}+k} \delta_m^{\ell+1}(t) w_{jm}^{\ell}(t-k)$$

$$= sgm'(s_j^{\ell}(k)) \sum_{m=1}^{N_{\ell+1}} \sum_{t=0}^{T^{\ell}} \delta_m^{\ell+1}(k+n) w_{jm}^{\ell}(n)$$

$$= sgm'(s_j^{\ell}(k)) \sum_{m=1}^{N_{\ell+1}} \vec{\delta}_m^{\ell+1}(k) \bullet \vec{w}_{jm}^{\ell}$$

En resumen el algoritmo BP temporal es el siguiente :

$$\vec{w}_{ij}^{\ell}(k+1) = \vec{w}_{ij}^{\ell}(k) - \mu\delta_j^{\ell+1}(k)\vec{x}_i^{\ell}(k)$$

$$\delta_j^{\ell}(k) = \begin{cases} -2e_j(k)sgm'(s_j^L(k)) & \ell = L \\ sgm'(s_j^{\ell}(k))\sum_{m=1}^{N_{\ell+1}}\vec{\delta}_m^{\ell+1}(k) \bullet \vec{w}_{jm}^{\ell} & 1 \le \ell \le L-1 \end{cases}$$

donde

$$\vec{\delta}_m^{\ell}(k) = \left[\delta_m^{\ell}(k), \delta_m^{\ell}(k+1), \ldots, \delta_m^{\ell}(k+T^{\ell-1})\right].$$

Es la generalización vectorial del algoritmo BP tradicional. Si se reemplazan los vectores **x**, **w** y δ por escalares el algoritmo se reduce a BP estándar para redes estáticas. Para calcular el δ para una neurona dada deben filtrarse los deltas de la capa inmediatamente superior hacia atrás a través de las sinapsis FIR.

Cada coeficiente se calcula sólo una vez en contraste con el redundante uso de términos al aplicar BP estándar a la red desenrrollada.

Condición de Causalidad

Los deltas son no-causales. De su definición $\delta_j^{\ell}(k) = \dfrac{\partial E}{\partial s_j^{\ell}(k)}$,

pero ya que toma tiempo para que la salida de cualquier neurona interna se propage a través de la red, el cambio en el error total debido a un cambio en el estado interno es una función de valores futuros dentro de la red. Para una red FIR basta considerar un número finito de valores futuros, y una simple reindexación permite escribir el algoritmo en forma causal.

$$\vec{w}_{ij}^{\,L-1-n}(k+1) = \vec{w}_{ij}^{\,L-1-n}(k) - \mu\delta_j^{\,L-n}(k-nT)\vec{x}_i^{\,L-1-n}(k-nT)$$

$$\delta_j^{\,L-n}(k-nT) = \begin{cases} -2e_j(k)sgm'(s_j^L(k)) & n=0 \\ sgm'(s_j^{L-n}(k-nT))\sum_{m=1}^{N_{\ell+1}} \vec{\delta}_m^{\,L+1-n}(k-nT) \bullet \vec{w}_{jm}^{\,L-n} & 1 \le n \le L-1 \end{cases}$$

El efecto neto de este cambio es retrasar el ajuste del gradiente unos pocos pasos, cambiando ligeramente la convergencia.

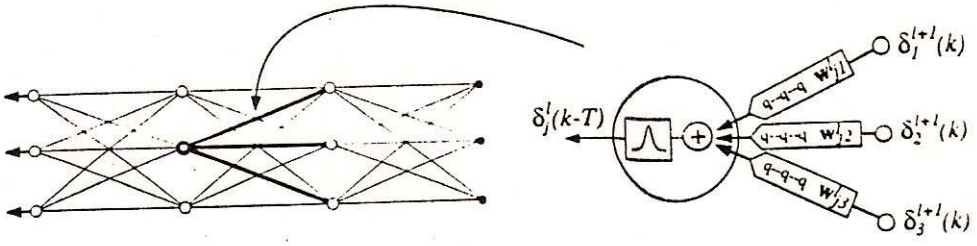


FIGURE 6 In temporal backpropagation, delta terms are *filtered* through synaptic connections to form the deltas for the previous layer. The process is applied layer by layer, working backward through the network.

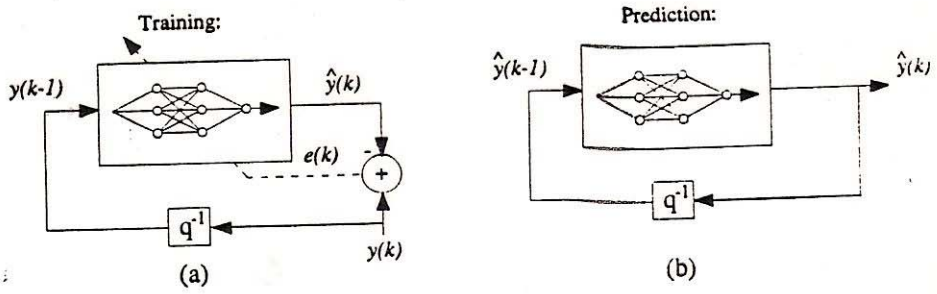


FIGURE 7 Network Prediction Configuration: The single-step prediction $\hat{y}(k)$ is taken as the output of the network driven by previous sample of the sequence $y(k)$. During training the single-step squared prediction error, $e^2(k) = (y(k) - \hat{y}(k))^2$, is minimized using temporal backpropagation. Feeding the estimate $\hat{y}(k)$ back forms a closed loop process used for long-term iterated predictions.

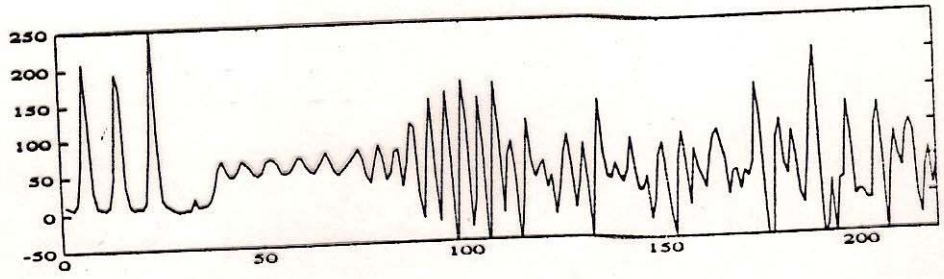


FIGURE 11 Extended iterated prediction.

Resultados de la competencia del Instituto Santa Fe (1992).

Se desea predecir las fluctuaciones caóticas en la intensidad de un láser infrarojo NH_3. El sistema se puede describir por 3 ecuaciones diferenciales ordinarias no lineales acopladas. Para la competencia se dieron a conocer 1000 muestras y la tarea fue predecir las próximas 100 muestras, incluyendo un colapso de la señal (las que no se dieron a conocer).

La red FIR ganó la competencia, obteniendo una predicción notablemente exacta con una pequeña degradación de fase.

Medida de Desempeño

Se usó el error cuadrático normalizado

$$NMSE = \frac{\sum_{k=1}^{N}\left(y(k)-\hat{y}(k)\right)^2}{\sum_{k=1}^{N}\left(y(k)-\bar{y}(k)\right)^2} = \frac{1}{\sigma^2 N}\sum_{k=1}^{N}\left(y(k)-\hat{y}(k)\right)^2$$

donde y es el valor deseado e $\hat{y}$ el valor estimado. Por otra parte $\bar{y}$ es el valor medio y σ^2 es la varianza de la secuencia verdadera de k=1 a N. Un valor de NMSE=1 corresponde a predecir la media incondicional de la data, i.e. el predictor trivial. Lo deseable es predecir la media condicional.

Dimensión de la red

Se usó un red de 3 capas con estructura 1-12-12-1 y 25 :5 :5 rezagos por capa. La selección de estas dimensiones se hizo básicamente por prueba y error. La selección del orden del filtro en la primera capa se hizo a partir de las técnicas lineales. El error residual no mejora para predictores lineales AR de orden mayor a 15 y la autocorrelación muestra una correlación importante hasta rezagos de orden 60.

Entrenamiento y Validación Cruzada

Los datos fueron escalados para tener media cero y varianza uno. Los primeros 900 puntos de la serie se usaron para entrenar y los restantes 100 para validar. Se usó el error de predicción iterada a partir de los 550 puntos para evaluar las distintas redes y ver cuan bien la red predecía el colapso en el punto 600.

Ajuste de parámetros y conducta de largo plazo

La red usada para entrenar tenía 1105 parámetros y fue ajustada con 1000 muestras. Parece que este tamaño de la red es necesario para modelar el colapso de la señal, de la que hay sólo dos ejemplos en el conjunto de entrenamiento.

Sin embargo, el gran número de parámetros afecta la conducta de largo plazo, donde la señal estimada despliega una conducta ruidosa.

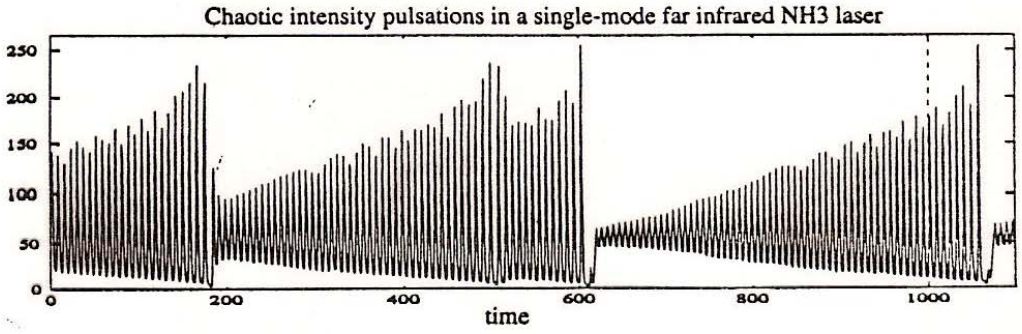


FIGURE 8 1100 time points of chaotic laser data.

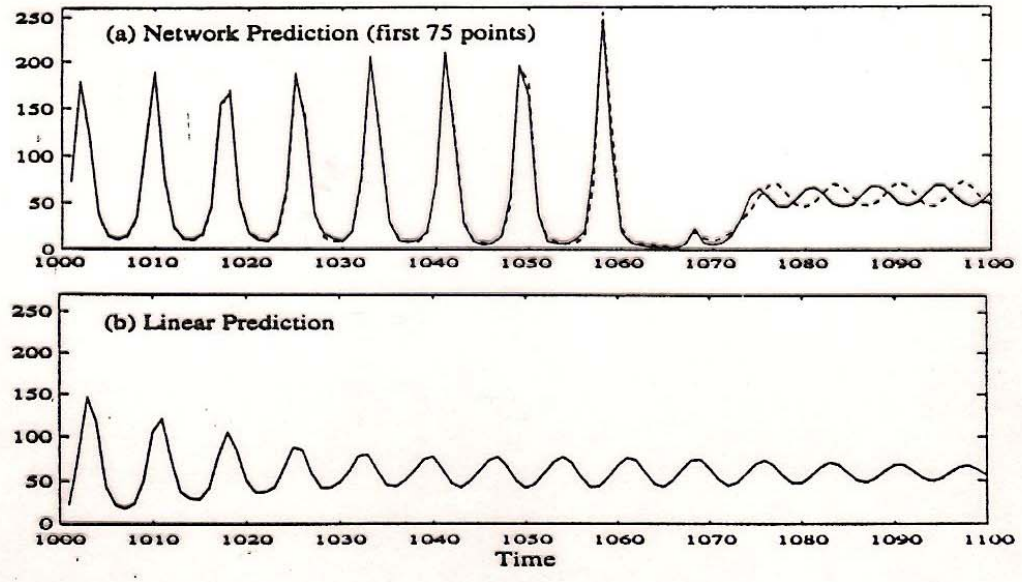


FIGURE 9 Time series predictions: (a) Iterated neural network prediction (first 75 points). The remaining 25 points were selected by adjoining a similar sequence taken from the training set. The prediction is based only on the supplied 1,000 points. Dashed line corresponds to actual series continuation. (b) 100-point iterated prediction based on a 25th-order linear autoregression. Regression coefficients were solved using a standard least squares method.

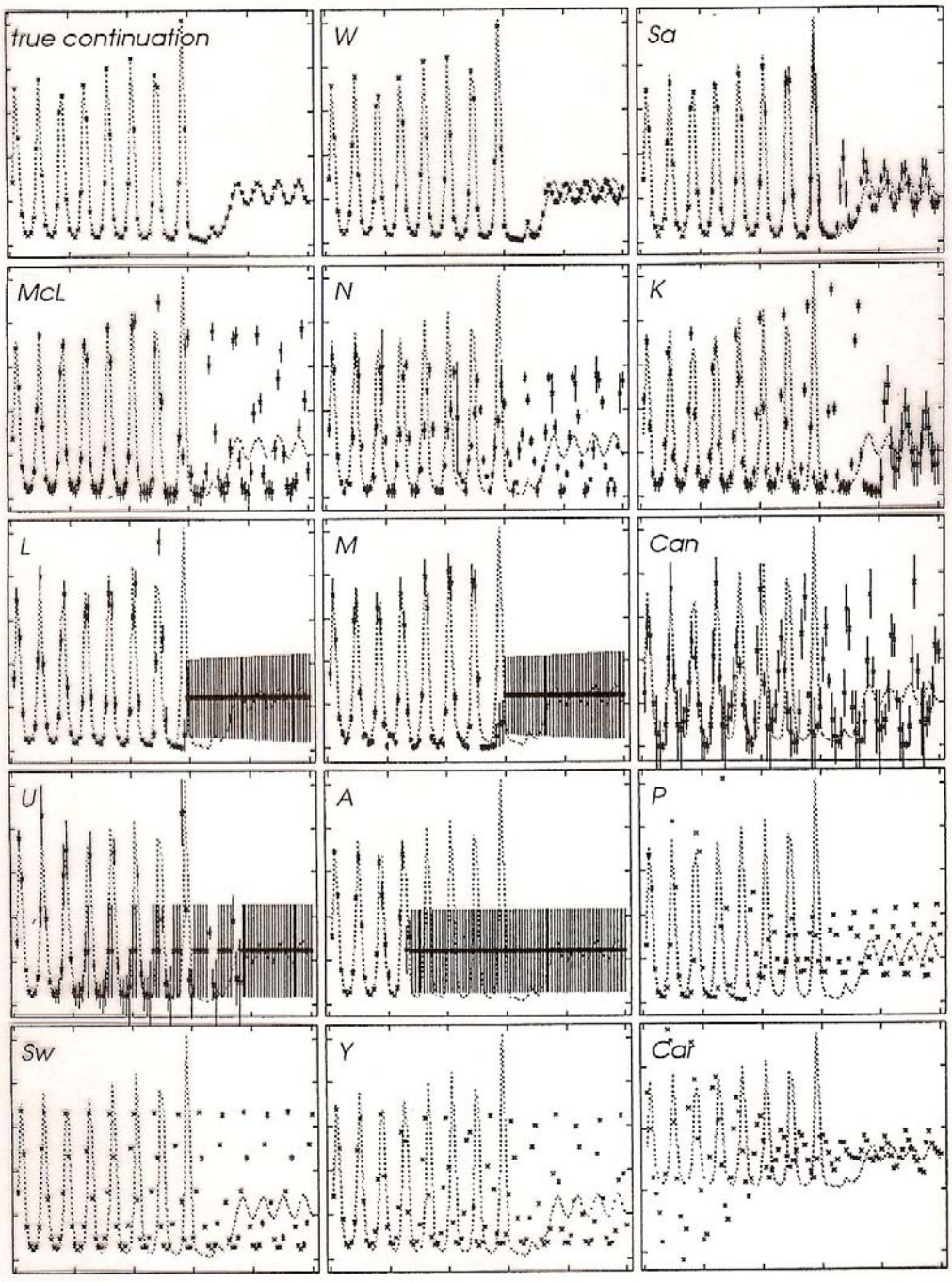


FIGURE 10 Continuations of Data Set A (laser). The letters correspond to the code of the entrant. Grey lines indicate the true continuation, × the predicted values, and vertical bars the predicted uncertainty.

TABLE 2 Entries received before the deadline for the prediction of Data Set A (laser). We give the normalized mean squared error (NMSE), and the negative logarithm of the likelihood of the data given the predicted values and predicted errors. Both scores are averaged over the prediction set of 100 points.

code	method	type	computer	time	NMSE(100)	-log(lik.)
W	conn	1-12-12-1; lag 25,5,5	SPARC 2	12 hrs	0.028	3.5
Sa	loc lin	low-pass embd, 8 dim, 4nn	DEC 3100	20 min	0.080	4.8
McL	conn	feedforward, 200-100-1	CRAY Y-MP	3 hrs	0.77	5.5
N	conn	feedforward, 50-20-1	SPARC 1	3 weeks	1.0	6.1
K	visual	look for similar stretches	SG Iris	10 sec	1.5	6.2
L	visual	look for similar stretches			0.45	6.2
M	conn	feedforward, 50-350-50-50	386 PC	5 days	0.38	6.4
Can	conn	recurrent, 4-4c-1	VAX 8530	1 hr	1.4	7.2
U	tree	k-d tree; AIC	VAX 6420	20 min	0.62	7.3
A	loc lin	21 dim, 30 nn	SPARC 2	1 min	0.71	10.
P	loc lin	3 dim time delay	Sun	10 min	1.3	–
Sw	conn	feedforward	SPARC 2	20 hrs	1.5	–
Y	conn	feedforward, weight-decay	SPARC 1	30 min	1.5	–
Car	linear	Wiener filter, width 100	MIPS 3230	30 min	1.9	–

Predicción de Series de Tiempo

Modelo NAR (auto-regresión no lineal) :

$$y(k) = \hat{y}(k) + e(k) = N\left[y(k-1), y(k-2), \ldots, y(k-T)\right] + e(k)$$

Teorema de Takens (1981)
Para una amplia clase de sistemas determinísticos existe un mapeo diferencial uno a uno entre una ventana finita de la serie de tiempo $\left[y(k-1), y(k-2), \ldots, y(k-T)\right]$ y el estado subyacente del sistema dinámico. Esto implica que existe en teoría una autoregresión nolineal de la forma

$$y(k) = g\left[y(k-1), y(k-2), \ldots, y(k-T)\right]$$

que modela la serie en forma exacta (suponiendo que no hay ruido).

Una red neuronal prealimentada puede aproximar la función ideal g. Modelo FIR permite aproximar NAR. Modelo de enseñanza forzada (Teacher forcing) : Durante el entrenamiento se realiza una adaptación de lazo abierto, i.e. la salida de la red no se realimenta sino que se usa la salida deseada.

En un ambiente estocástico estacionario, minimizar la suma de los errores cuadráticos instantáneos corresponde a minimizar el valor esperado del error cuadrático :

$$E\left[e^2(k)\right] = E\left[y(k) - \hat{y}(k)\right]^2 = E\left[y(k) - N_c\left[\vec{y}_1^T(k)\right]\right]^2$$
$$= E\left[y(k) - E\left[y(k) / \vec{y}_1^T(k)\right]\right]^2 + E\left[N_c - E\left[y(k) / \vec{y}_1^T(k)\right]\right]^2$$

donde

$$\vec{y}_1^T = \left[y(k-1), y(k-2), \ldots, y(k-T)\right]$$

Se deduce de aquí que el modelo óptimo es la media condicional

$$N_c^* = E\left[y(k) / \vec{y}_1^T(k)\right]$$

Esto motiva el uso de entrenamiento forzado.

Una vez que se ha entrenado la red, se puede predecir en forma iterada realimentando el estimador como entrada a la red :

$$\hat{y}(k) = N_q\left[\hat{y}(k-1)\right]$$