Tesis de Licenciatura en Ciencias de la Computación

GENERADORES DE TIEMPO BASADOS EN REDES NEURONALES

Fernando Luis Martínez

Diciembre de 2005

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Directores:
Jean-Philippe Boulanger
Enrique Carlos Segura

Índice general

Resumen

Los Generadores de Tiempo son modelos estocásticos que permiten generar series de índices climáticos que se corresponden con las características del clima actual. En esta tesis se explorará el potencial de las redes neuronales para diseñar una nueva generación de generadores de tiempo.

En primer lugar, se utilizaron Perceptrones Multicapa para construir un generador de precipitaciones diarias en dos etapas que pueda ser aplicado a distintos regimenes climáticos y que corrija los problemas más comunes que presentan estos generadores. Al compararlo con el modelo de Markov para la ocurrencia de días de lluvia, el modelo representó adecuadamente la media y la desviación estándar de la cantidad de días de lluvia por mes y mejoró significativamente la simulación de períodos secos y lluviosos largos. Luego, al compararlo con las tres funciones paramétricas (Gamma, Weibull y Doble Exponencial) utilizadas usualmente para ajustar la distribución de precipitaciones, se obtuvo un mejor ajuste de la distribución y se mejoró la simulación de cantidades extremas y la desviación estándar interanual.

En segundo lugar, se utilizaron Redes Recurrentes Simples para construir un generador de temperatura diarias, basado en el modelo de Richardson, que se adapte a las características de distintos tipos de climas. El modelo logró ajustar adecuadamente la distribución de las series de temperatura y, en general, reprodujo aceptablemente la correlación temporal de los valores observados. Estos resultados fueron obtenidos usando 19 estaciones meteorológicas en Argentina y confirmados para estaciones en EEUU, Europa y los Trópicos.

Abstract

Weather Generators are stochastic models, which allow generating series of climate indices with the same characteristics of past observations. The present thesis explores the potential of neural network architectures to design a new generation of weather generator models.

First, we used Multilayer Perceptrons for building a two-step daily rainfall generator, which can be used with different types of climates and solve the common problems of traditional models. As compared to a rainfall occurrence Markov model, the model represented fairly well the mean and standard deviation of the number of wet days per month, and it significantly improved the simulation of the longest dry and wet periods. Then, we compared the model with the three parametric distribution functions (Gamma, Weibull and Double-Exponential) usually applied to fit rainfall distribution. We found a better fitting of the distribution and a better simulation of the extreme amounts and the interannual standard deviation.

Second, we used Simple Recurrent Networks for building a daily temperature generator, based on the Richardson model, which can be adapted to the caracteristics of different kinds of climate. The model was able to fit relatively well the temperature distribution and, in general, it reproduced well the temporal correlation of observed data. These results were obtained using 19 weather stations in Argentina and were confirmed with stations in the United States, Europe and the Tropics.

Capítulo 1

Introducción

La gestión de riesgos asociados al clima, requiere del análisis de observaciones históricas de distintos índices climáticos a escala diaria. En ciertas ocasiones las series disponibles de datos observados están incompletas o son demasiado cortas. Mediante el uso de modelos estocásticos es posible generar series temporales que replican las características del clima actual y con ellas, realizar análisis de riesgo más precisos. Dichos Modelos, denominados *"Generadores de Tiempo"* o *"Generadores Climáticos"*, son utilizados en una gran cantidad de aplicaciones tales como: modelos de crecimiento de plantas, modelos de procesos hidrológicos y caudal de ríos, predicción del impacto del cambio climático, etc.

Las Redes Neuronales aplicadas al clima nos permiten lidiar con problemas relativos a determinados fenómenos que no se encuentran completamente caracterizados en términos estadísticos o para los cuales no existe una explicación física detallada. De esta manera, podemos hacer que una red aprenda algo que nosotros no pudimos aprender. Muchas veces el problema se desplaza a determinar qué fue realmente lo que la red aprendió. O sea, podemos conseguir una red que resuelve determinado problema y no ser capaces de demostrar formalmente como alcanza dicha solución, sin embargo, la mayoría de las veces esto no importa demasiado.

1.1. Generadores de Precipitaciones

En ([Srikanthan, 2001]) se describen cuatro clases de generadores de precipitaciones: *"two-part model"*, *'transition probability model"*, *"resampling model"* y *"ARMA model"*. El modelo de dos etapas, que será el punto de partida para el desarrollo de nuestro modelo, considera en forma separada la ocurrencia de días de lluvia y la cantidad de precipitaciones.

Los modelos de ocurrencia son principalmente de dos tipos: los modelos basados en cadenas de Markov, en los cuales se especifica el estado de cada día (lluvioso o seco) y se intenta relacionar el estado del día actual con el estado de los días anteriores; los modelos que ven a la serie de la ocurrencia como una secuencia de períodos secos y lluviosos alternados cuya distribución es considerada independiente.

La cantidad de precipitaciones para un día determinado, generalmente es considerada independiente de lo sucedido en días anteriores. Para modelar las precipitaciones han sido usadas distintas funciones paramétricas (Gamma, Weibull, Doble Explonencial) que permiten ajustar la distribución de la serie.

1.2. Generadores de Temperatura

Una característica fundamental que deben preservar los generadores de temperatura es la autocorrelación temporal que generalmente presentan los índices de temperatura mínima y máxima. Si bien existen una gran variedad de modelos, en general se asume que ambos índices tienen una distribución

normal y que su valor, para un día determinado, depende del estado de ese día (seco o lluvioso) y de las temperaturas de los días anteriores.

En este caso, para desarrollar nuestro modelo tomaremos como base el modelo propuesto por [Richardson, 1981]. Este modelo toma las series sin el ciclo estacional y define la temperatura de un día determinado únicamente en función de la temperatura del día anterior.

La bondad de los diferentes modelos propuestos, tanto para precipitaciones como para temperatura, en general depende de las características del clima sobre el cual el modelo se utiliza. La posibilidad de obtener un generador universal, que pueda ser utilizado sobre diferentes regimenes climáticos, constituye la principal motivación para la utilización de Redes Neuronales en el desarrollo de una nueva generación de “Generadores de Tiempo”.

Capítulo 2

Datos

2.1. Argentina

Para verificar los modelos propuestos, se han utilizado las series temporales de 19 estaciones meteorológicas de la Argentina. La mayor parte de la región analizada forma parte de la cuenca del Plata, probablemente la zona agrícola mas importante del país. En el Cuadro 2.1 se muestra en detalle el nombre y la ubicación de cada una de estas estaciones y el período considerado. Los datos fueron obtenidos del Servicio Meteorológico Nacional y han sido pre-procesados en el Departamento de Cs. de la Atmósfera de la FCEyN (UBA). Todas las estaciones cuentan con menos del 10 % de datos faltantes.

2.2. El Mundo

Con el objeto de medir la universalidad de los generadores, se procedió a evaluar su desempeño en diferentes tipos de clima. Fueron seleccionadas para esto, diferentes estaciones de Europa, los EEUU y los trópicos. Los datos europeos fueron obtenidos de la base de datos de "European Climate Assessment & Dataset" (http://eca.knmi.nl/). Los datos de los EEUU y los trópicos, fueron obtenidos de la base de datos del "National Climate Data Center" (http://www.ncdc.noaa.gov). Los Cuadros 2.2, 2.3 y 2.4 muestran en detalle el nombre y la ubicación de cada una de las estaciones y el período considerado. Cabe destacar que no se realizaron pruebas para el generador de temperaturas sobre los EEUU y los trópicos a causa de la falta de disponibilidad de series históricas suficientemente largas y de la calidad necesarias para el correcto funcionamiento de los modelos.

Nombre	Provincia	Latitud (S)	Longitud (O)	Período
Corrientes	Corrientes	27,45	58,76	1963 – 1996
Formosa	Formosa	26,20	58,23	1962 – 1996
Posadas	Misiones	27,36	55,96	1959 – 1996
Paso de los Libres	Corrientes	29,68	57,15	1959 – 1996
Pilar	Córdoba	31,45	64,28	1959 – 1996
Parana	Entre Ríos	31,78	60,48	1959 – 1996
Marcos Juarez	Córdoba	32,70	62,15	1959 – 1996
Pergamino	Buenos Aires	33,56	60,33	1967 – 1996
Rosario	Santa Fe	32,91	60,78	1959 – 1996
Gualeguaychu	Entre Ríos	33,00	58,61	1959 – 1996
Laboulaye	Córdoba	34,13	63,36	1959 – 1996
Pehuajo	Buenos Aires	35,86	61,90	1959 – 1996
Junin	Buenos Aires	34,55	60,95	1959 – 1996
Nueve de Julio	Buenos Aires	35,45	60,88	1959 – 1996
Ezeiza	Buenos Aires	34,81	58,53	1959 – 1996
Santa Rosa	La Pampa	36,56	64,26	1951 – 1996
Pigue	Buenos Aires	37,60	62,38	1959 – 1996
Mar Del Plata	Buenos Aires	37,93	57,58	1959 – 1996
Bahia Blanca	Buenos Aires	38,73	62,16	1959 – 1996

Cuadro 2.1: Estaciones Meteorológicas de la Argentina

Nombre	País	Latitud (N)	Longitud (E)	Período
Dublin	Irlanda	53,21	−06,19	1941 – 2000
Malaga	España	36,40	−4,28	1942 – 2000
Karlstad	Suecia	59,21	13,28	1941 – 2000
Zurich	Suiza	47,23	8,34	1941 – 2000
Lisboa	Portugal	38,43	−9,09	1941 – 2000
Kobenhaven	Dinamarca	55,41	12,32	1941 – 2000
Helsinki	Finlandia	60,10	24,57	1951 – 2000
Paris	Francia	48,49	2,20	1941 – 2000
Berlin	Alemania	52,27	13,18	1941 – 2000
Roma	Italia	41,47	12,35	1951 – 2000
De Bilt	Holanda	52,06	05,11	1941 – 2000
Oxford	Inglaterra	51,46	−1,16	1941 – 2000
Heraklion	Grecia	35,20	25,11	1955 – 2000
Uccle	Bélgica	50,48	4,21	1941 – 2000
Praga	República Checa	50,05	14,25	1941 – 2000
Vilnius	Lituania	54,38	25,17	1941 – 2000
Voru	Estonia	57,50	27,01	1941 – 2000

Cuadro 2.2: Estaciones Meteorológicas de Europa

Nombre	Estado	Latitud (N)	Longitud (O)	Período
Astoria	Oregon	46,15	123,88	$1994 - 2005$
Eugene	Oregon	44,12	123,22	$1994 - 2005$
Portland	Oregon	45,60	122,60	$1994 - 2005$
Salem	Oregon	44,90	123,00	$1994 - 2005$
Reno	Nevada	39,50	119,78	$1994 - 2005$
Minneapolis	Mineapolis	44,88	93,22	$1994 - 2005$
Rochester	Mineapolis	44,00	92,45	$1994 - 2005$
St. Cloud	Mineapolis	45,58	94,18	$1994 - 2005$
Tulsa	Oklahoma	36,20	95,90	$1994 - 2005$
Jackson	Mississippi	32,32	90,08	$1994 - 2005$
Meridian	Mississippi	32,33	88,75	$1994 - 2005$
Atlantic City	New Jersey	39,45	74,58	$1994 - 2005$
Baltimore	Maryland	39,18	76,67	$1994 - 2005$
New York City	New York	40,65	73,78	$1994 - 2005$
Philadelphia	Pennsylvania	39,88	75,23	$1994 - 2005$
Williamsport	Pennsylvania	41,25	76,92	$1994 - 2005$

Cuadro 2.3: Estaciones Meteorológicas de los Estados Unidos

Nombre	País	Latitud (N)	Longitud (O)	Período
Bogota	Colombia	4,70	74,13	$1994 - 2005$
Iquitos	Perú	3,75	73,25	$1994 - 2005$
Puerto Limon	Costa Rica	10,00	83,05	$1994 - 2005$
Asunción	Paraguay	$-25,26$	57,63	$1994 - 2005$
Manila	Filipinas	14,58	$-120,98$	$1994 - 2005$
Abidjan	Ivory Coast	5,25	3,93	$1994 - 2005$
Pretoria	Sud Africa	$-25,73$	$-28,18$	$1994 - 2005$

Cuadro 2.4: Estaciones Meteorológicas de los Trópicos

Capítulo 3

Redes Neuronales Artificiales

Las Redes Neuronales Artificiales (RNA) constituyen un paradigma computacional alternativo denominado *paradigma conexionista*. La forma de resolver un problema utilizando alguno de los paradigmas convencionales, puede resumirse en modelar matemáticamente (con distintos grados de formalismo) el problema en cuestión y luego formular un programa que permita resolverlo. En contraposición, una aproximación basada en RNA parte de un conjunto de datos de entrada suficientemente significativo y su objetivo es conseguir que la red aprenda automáticamente las propiedades deseadas.

Las RNA fueron originalmente una simulación abstracta de los sistemas nerviosos biológicos, formada fundamentalmente por un conjunto de unidades llamadas "neuronas" conectadas unas con otras. Algunas de las características deseables de los sistemas nerviosos biológicos que motivaron el desarrollo de las RNA, son: la robustez y tolerancia a fallas; la flexibilidad para adaptarse a cambios en el entorno a través de un mecanismo de aprendizaje; la capacidad de lidiar con información inconsistente, con ruido o incompleta y un alto grado de concurrencia.

Si bien, desde un punto de vista neurofisiológico, los modelos de redes artificiales son extremadamente más simples que las redes biológicas, tanto en cuanto a la cantidad de neuronas, como en la complejidad de cada una de ellas (en el cerebro humano hay aproximadamente 10^{11} neuronas de distintos tipos, y cada una de ellas está conectada a otras 10^4), la esencia del comportamiento colectivo de una red de células se mantiene. Por otra parte, muchos de los detalles relativos al comportamiento de una neurona real no son relevantes desde un punto de vista computacional.

El primer modelo de red neuronal fue propuesto en 1943 por McCulloch y Pitts. Este es un modelo binario donde cada neurona tiene asociado un umbral de activación. Específicamente, el modelo de neurona, computa una suma ponderada de sus valores de entrada y devuelve un 1 (activa) si el valor resultante es mayor al umbral y 0 (inactiva) si no:

$$v_i = \Theta(\sum w_{ij} v_j - \theta_i) \tag{3.1}$$

$$\Theta(i) = \begin{cases} 1\, si\, x \geq 0 \\ 0\, si\, no \end{cases} \tag{3.2}$$

donde el w_{ij} representa el peso de la conexión entre las neuronas i y j.

Este primer modelo de neurona, puede generalizarse reemplazando la función Θ con una función más general a la que llamaremos función de activación. Así, en una red neuronal típica, cada unidad o neurona, tiene asociado un valor de activación. Este valor se propaga a través de conexiones unidireccionales hacia otras neuronas de la red, influyendo, de esta manera, en su inhibición o activación (la interacción entre 2 neuronas recibe el nombre de *sinapsis*). Cada conexión tiene asociado un peso sináptico (w_{ij}) que determina el efecto de la neurona pre-sináptica j sobre la neurona post-sináptica i

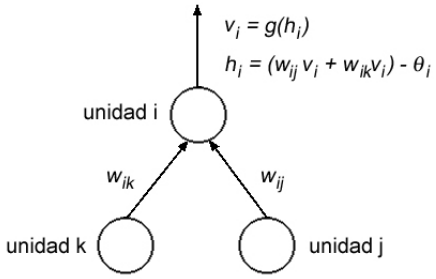

Figura 3.1: Activación de una unidad o neurona post-sináptica conectada a 2 neuronas pre-sinápticas. Siendo, w_{ij} el peso asociado a la conexión entre la unidad j y la unidad i, v_i el valor de salida de la neurona i y g su función de activación.

(ver Figura 3.1). El período de aprendizaje puede resumirse entonces en encontrar los pesos adecuados para que la red efectúe determinada tarea. Durante este período de entrenamiento los pesos de la red son ajustados iterativamente. Este proceso puede realizarse principalmente de dos maneras:

Entrenamiento Supervisado. En este caso, el entrenamiento se basa en comparar las salidas de la red con la respuesta esperada para cada entrada. Para esto es necesario contar con un conjunto de pares (patrones) de entrada-salida que representen significativamente el problema a resolver. Luego, es posible ajustar los pesos para minimizar la diferencia entre las salidas que devuelve la red y las salidas "correctas".

Entrenamiento No Supervisado. Para los modelos de entrenamiento no supervisado, el conjunto de datos de entrenamiento consiste sólo en los patrones de entrada. La idea es que la red extraiga información relevante acerca de la correlación de los datos y de esta manera pueda categorizarlos de alguna manera.

3.1. Perceptrones

Los perceptrones constituyen probablemente el modelo de red neuronal más estudiado. Las unidades de un perceptrón están organizadas en capas, de forma tal que cada unidad de una capa determinada se conecta con todas las unidades de la capa siguiente (dado que todas las conexiones entre neuronas son hacia adelante, estas redes también reciben el nombre de *"feed-forward networks"*). En primer lugar, hay una capa de entrada cuyas unidades tomarán su valor del medio ambiente, luego hay una serie de capas intermedias que por lo general reciben el nombre de capas ocultas y finalmente hay una capa de salida que representa la salida de la red. Además cada capa tiene asociada una unidad "ficticia" llamada bias, que representa los umbrales de las unidades de la capa siguiente. La Figura 3.2 muestra un perceptrón con una única capa oculta.

Un *perceptrón simple* es un caso particular de un perceptrón sin capas ocultas. La salida de un perceptrón simple para una entrada dada que llamaremos ξ^μ es igual a:

$$O_i = g\left(\sum_{k=0}^{N} w_{ik}\xi_k\right) = g\left(\sum_{k=1}^{N} w_{ik}\xi_k - \theta_i\right) \tag{3.3}$$

donde O_i es el valor de la neurona i de la capa de salida; θ_i es el umbral asociado a dicha neurona y ξ_k es el valor de la entrada k. Notar que si fijamos $\xi_0 = -1$ entonces w_{i0} corresponde al umbral de la neurona i.

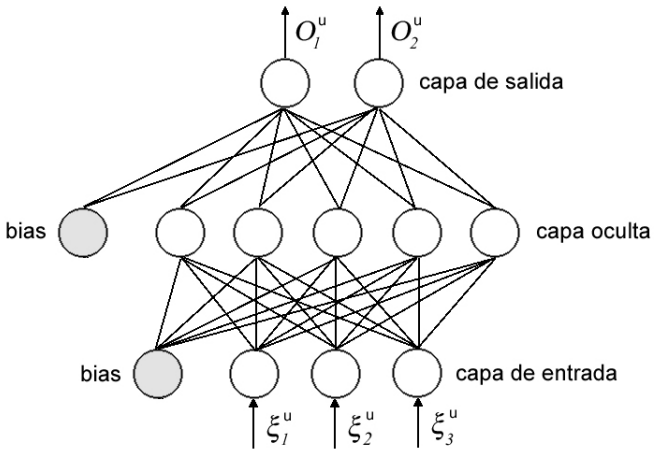

Figura 3.2: Perceptrón con una capa oculta. ξ^u es la entrada u de la red y O^u su correspondiente salida. Las unidades llamadas bias, son unidades sin conexiones de entrada cuyo valor es siempre -1 y representan los umbrales de la capa siguiente.

Distintas funciones de activación pueden definirse para las neuronas de salida, dependiendo de las características del problema en cuestión. Para problemas de regresión, por ejemplo, se utiliza comúnmente una función lineal, mientras que para problemas de clasificación puede utilizarse una función sigmoidea.

La idea de la etapa de entrenamiento es ajustar los pesos con el objetivo de minimizar alguna función que represente el error de la red con respecto a las salidas "correctas" para un conjunto de patrones de entrada. Este error puede escribirse de la siguiente manera:

$$E[w] = \frac{1}{2}\sum_{i\mu}(\zeta_i^\mu - O_i^\mu)^2 = \frac{1}{2}\left[\sum_{i\mu}\zeta_i^\mu - g\left(\sum_k w_{ik}\xi_k^\mu\right)\right]^2 \tag{3.4}$$

donde O_i^μ es la salida de la red y ζ_i^μ la salida esperada para la entrada ξ_k^μ. A esta función se la llama usualmente función de costo. Para minimizar esta función, puede utilizarse el método del *descenso por el gradiente*, ajustando cada peso w_{ij} en dirección contraria al gradiente en ese punto:

$$w_{ik} = w_{ik} + \Delta w_{ik} \tag{3.5}$$

$$\Delta w_{ik} = -\eta\frac{\partial E}{\partial w_{ik}} = \eta\sum_\mu(\zeta_i^\mu - O_i^\mu)\xi_k^\mu \tag{3.6}$$

donde η representa la velocidad del aprendizaje.

Utilizando la ecuación anterior es posible actualizar el valor de los pesos luego de computar la salida de la red para todo el conjunto de entrenamiento (entrenamiento batch). Otra posibilidad es ajustar los pesos luego de computar el valor de la red para cada uno de los patrones de entrenamiento (entrenamiento online). En este caso la ecuación (3.6) quedaría así:

$$\Delta w_{ik} = -\eta\frac{\partial E^\mu}{\partial w_{ik}} = \eta(\zeta_i^u - O_i^u)\xi_k^\mu. \tag{3.7}$$

Para esto, los patrones deben ser tomados en orden aleatorio. Luego de "mostrarle" a la red todos los patrones del conjunto de entrenamiento diremos que pasó una época. El proceso de entrenamiento

continuará iterativamente época tras época, hasta que se cumpla algún criterio de parada (típicamente que el error de la red sea menor a un ϵ dado).

Para redes que poseen capas ocultas, el método de entrenamiento más usado es el algoritmo de *"back propagation"*. Este algoritmo da una manera de encontrar las derivadas parciales con respecto a cada peso, propagando el error de la red hacia atrás desde la capa de salida a través de toda la red.

En primer lugar, el error de las neuronas de la capa de salida para un patrón de entrada-salida μ, puede definirse como:

$$\delta_k = g'(h_k)(\zeta_k^\mu - O_k^\mu) \tag{3.8}$$

donde g es la función de activación asociada a las neuronas de la capa de salida, h_k es el valor de entrada de la unidad k, O_k^μ es la salida de la red para la neurona k y ζ_k^μ es la salida esperada.

Luego, es posible actualizar los pesos que conectan a la última capa oculta con la capa de salida:

$$w_{kj} = w_{kj} + \eta \delta_k v_j \tag{3.9}$$

donde w_{kj} es el peso que conecta a la neurona j de la capa oculta con la neurona k de la capa de salida y v_j el valor de la unidad j y $\delta_k v_j = -\partial E^\mu / \partial w_{kj}$.

Después de actualizar los valores de los pesos que conectan a la última capa oculta con la capa de salida, es posible calcular el error que aporta cada neurona de la capa oculta:

$$\delta_j = g'(h_j) \sum_k w_{kj} \delta_k. \tag{3.10}$$

Nuevamente, utilizando los valores δ_j es posible actualizar los pesos de la capa anterior con las mismas ecuaciones, y continuar así hasta llegar a la capa de entrada.

Una alternativa al algoritmo original de *"back propagation"* es sumar el valor de $\partial E^\mu / \partial w_{kj}$ para cada patrón del conjunto de entrenamiento y utilizar el resultado de esta sumatoria ($\partial E / \partial w_{kj}$) para actualizar los pesos al final de cada época. Distintos algoritmos proponen diferentes alternativas para realizar esta actualización. Algunos ejemplos son: el descenso por gradiente y el método del gradiente conjugado.

Como función de activación para las neuronas de la capa oculta, en general se utiliza la función logística o la tangente hiperbólica (las cuales se comportan de forma similar a una función escalón, con la ventaja de que son continuas y derivables). Otra ventaja de estas funciones es que su derivada ($f'(x)$) puede escribirse en función de $f(x)$, con esto no hace falta calcular $f'(x)$ (ver Figura 3.3).

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{3.11}$$

$$logistic(x) = \frac{1}{1 + e^{-x}}. \tag{3.12}$$

Por último, cabe destacar que, dada una cantidad suficientemente grande de neuronas en la capa oculta y eligiendo adecuadamente los valores de los pesos, un perceptrón con una capa oculta puede aproximar cualquier función continua de una región compacta del espacio de entrada, con una precisión arbitraria ([Funahashi, 1989]).

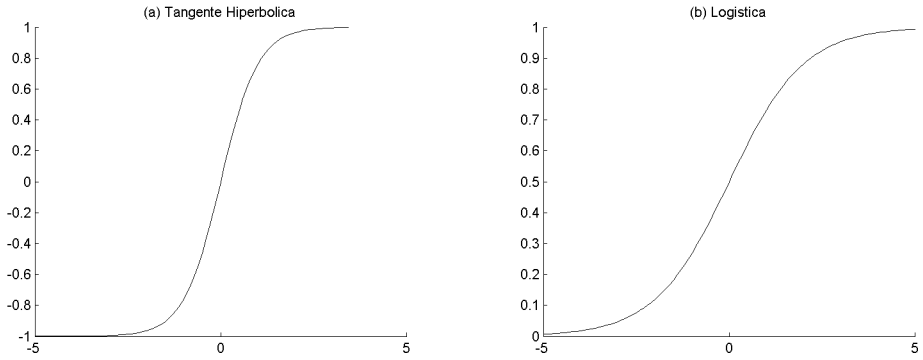

Figura 3.3: Funciones sigmoideas utilizadas como funciones de activación.

3.2. Redes Recurrentes

La forma más sencilla de tener en cuenta la información temporal presente en un conjunto determinado de datos, es utilizar un perceptrón y representar explícita y espacialmente el tiempo, tomando una ventana de tamaño fijo w en la capa de entrada. De esta manera la red tendrá en cuenta para el aprendizaje la correlación temporal entre los patrones de entrada que le serán mostrados de a w patrones por vez. A este tipo de redes se las llama comúnmente "tapped delay line feedforward networks" (ver Figura 3.4). Algunos problemas en este tipo de redes son: la necesidad de definir un tamaño fijo "a priori" para la ventana temporal, el uso de pesos independientes para conectar entradas que representan lo mismo pero a distintos instantes de tiempo y sobre todo, la dimensión de la capa de entrada hace necesario un conjunto de entrenamiento más grande ([Bodén, 2001]).

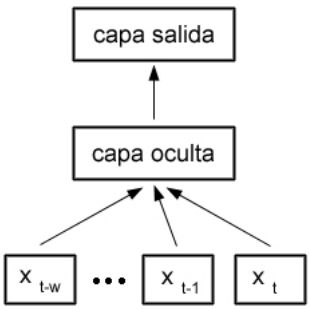

Figura 3.4: "Tapped Delay Line Feed Forward Network" con una ventana temporal de longitud w.

Las redes neuronales recurrentes surgen de la necesidad de incorporar la noción de tiempo en una red neuronal. Las conexiones recurrentes permiten mantener un estado interno que representa la memoria de la red. De esta manera, la salida de la red en un momento determinado no dependerá únicamente de la entrada, sino también de las entradas anteriores que contribuyeron a constituir su memoria. Dependiendo de la forma en la que es considerado el tiempo, podemos dividir a las redes recurrentes en dos grandes grupos: redes recurrentes de tiempo continuo y redes recurrentes de tiempo discreto.

Las *redes recurrentes simples* fueron propuestas por Elman en 1990 y constituyen una de las arquitecturas de tiempo discreto más usadas. Básicamente, una red recurrente simple es un perceptrón con una única capa escondida, al que se le agrega una capa de contexto o estado (ver Figura 3.5). Cada neurona de la capa oculta está conectada a una neurona de la capa de estado a través de una conexión con un peso fijo de 1.0 (estos pesos no son actualizados durante el entrenamiento de la red). Al computar la salida de la red para una entrada dada, los valores de las neuronas de la capa oculta son automáticamente copiados a la capa de estado. De esta forma, en un instante de tiempo t, la capa de estado contiene los valores de

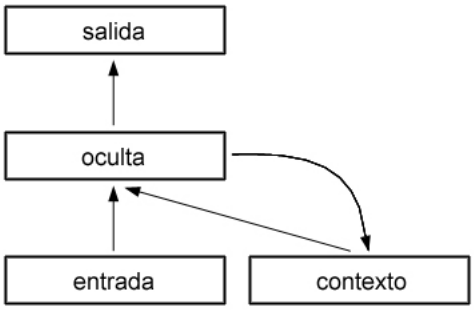

Figura 3.5: Una red recurrente simple en la cual los valores de activación de las neuronas de la capa oculta son copiados uno a uno a la capa de contexto o estado.

la capa oculta en el instante de tiempo t-1. Además, cada neurona de la capa de estado está conectada con todas las neuronas de la capa oculta, así, los valores de la capa oculta para el instante de tiempo t-1, influyen en la salida de la red en el instante t.

La forma más sencilla de entrenar una RRS es utilizar el algoritmo de *"back probagation"* (ver Sección 3.1) en su versión on-line. De esta manera los pesos de la red son actualizados con cada patrón del conjunto de entrenamiento. La única diferencia con el algoritmo original radica en que las conexiones recurrentes no son tenidas en cuenta en la etapa "backward". A esta versión del algoritmo se la conoce normalmente con el nombre de *"truncated back propagation"*.

3.3. Métodos Bayesianos

3.3.1. Arquitectura Óptima

Cuando optimizamos un modelo para representar determinado conjunto de datos, usualmente se considera al modelo como una función $y = f(x, w) + \epsilon$ donde y son las observaciones, x el input, w los parámetros a optimizar (en nuestro caso los pesos de la red) y ϵ el error. Modelos más complejos usualmente realizan un mejor "fit" de los datos (el error es menor), pero al especializarse demasiado en dichos datos ("overfitting") tienen una performance más pobre para datos no vistos (no incluidos en la etapa de entrenamiento). Una manera de evitar este problema es optimizar el modelo, considerando el error asociado a la incertidumbre de los parámetros. El uso de métodos bayesianos, es de gran ayuda a este respecto ([Nabney, 2002])

En la estrategia Bayesiana, la incertidumbre acerca de la estimación de los parámetros a partir de los datos es representada por una distribución de probabilidad. Dada una distribución inicial $p(w)$ la cual expresa nuestro conocimiento sobre los parámetros antes de observar los datos, este conocimiento puede ser actualizado (luego de observar los datos) a través del teorema Bayesiano; obteniendo de esta manera la distribución posterior $p(w|D)$ que representa la densidad de los parámetros para un conjunto de datos D dado.

Teorema Bayesiano:

$$P(w|D) = \frac{p(D|w)p(w)}{p(D)} \tag{3.13}$$

donde $p(D)$ es un factor de normalización (en nuestro caso, w son los pesos)

Siguiendo este teorema y considerando únicamente los términos que dependen de los pesos, el error bayesiano para un modelo determinado, esta dado por:

$$E = -log\, p(w|D) = -log\, p(D|w) - log\, p(w). \tag{3.14}$$

La probabilidad $p(D|w)$ representa el error del "fit" , el cual puede ser modelado (para un problema de regresión) por una función Gauseana de la siguiente manera:

$$p(D|w) = \left(\frac{\beta}{2\pi}\right)^{N/2} exp\left(-\frac{\beta}{2}\sum\{y(x^n;w) - t^n\}^2\right) \tag{3.15}$$

donde β representa la inversa de la varianza del error del "fit".

El requerimiento de pesos pequeños (para evitar el "overfitting") sugiere una distribución Gauseana para los pesos de la forma:

$$p(w) = \left(\frac{\alpha}{2\pi}\right)^{W/2} exp\left(-\frac{\alpha}{2}\sum w_i^2\right) \tag{3.16}$$

donde α representa la inversa de la varianza de la distribución de los pesos.

Dado que α y β son parámetros sobre parámetros, se los denomina hiperparámetros.

Finalmente la función del error puede reescribirse como:

$$E = \beta E_D + \alpha E_W$$

donde E_D es el aporte de los datos y E_W el error inicial de los pesos.

3.3.2. Evidence Procedure

Durante el aprendizaje, se ajustarán los pesos de la red, minimizando el error bayesiano. El "evidence procedure" es un algoritmo iterativo que se utiliza para aproximar la distribución posterior.

Ahora bien, la aproximación realizada por el algoritmo, es la siguiente:

$$p(w|D) \cong p(w|\alpha^{mp}, \beta^{mp}, D) \tag{3.17}$$

(mp = "most probable"), entonces podemos encontrar los valores para los hiperparámetros que optimizan $p(w|D)$ y utilizar la distribución, con estos valores fijos. Encontrar dichos valores implica calcular $p(\alpha, \beta|D)$.

$$p(\alpha, \beta|D) = \frac{p(D|\alpha, \beta)p(\alpha, \beta)}{p(D)} \tag{3.18}$$

donde $p(\alpha, \beta)$ puede pensarse como una distribución constante, luego, dado que $p(D)$ no depende de los pesos, todo se resume a maximizar $p(D|\alpha, \beta)$, este es el "evidence" de los hiperparámetros.

La idea es utilizar el "evidence procedure", durante el período de entrenamiento, para actualizar el valor de los hiperparámetros cada vez que se llega a un mínimo local de la función E.

Dados dos modelos distintos (en nuestro caso dos arquitecturas de red distintas) $M1$ y $M2$ es posible compararlos obteniendo $P(M1|D)$ y $P(M2|D)$, que es una medida de la probabilidad de reproducir D con cada modelo (penalizando a los modelos mas complejos que realizan un "overfit" de la función).

Entonces, si $P(M1|D) > P(M2|D)$, esto implica necesariamente que $p(D|\alpha_1, \beta_1) > p(D|\alpha_2, \beta_2)$. Así, podemos utilizar el logaritmo del "evidence" de los hiperparámetros ($logev = log(p(D|\alpha, \beta))$) para comparar arquitecturas.

Capítulo 4

Generadores de Tiempo

En este capítulo se describirán en detalle los modelos para la generación de los índices de precipitaciones y temperatura. Se expondrán también las características fundamentales de los modelos que se tomarán como referencia para medir el desempeño de los generadores, destacando diferencias y similitudes. El generador de precipitaciones basado en redes neuronales recibirá el nombre de *NNGEN-P* y el de temperatura *NNGEN-T*.

4.1. Ocurrencia de Días de Lluvia

Existen principalmente dos tipos de modelos para la ocurrencia de días de lluvia. *Cadenas de Markov* y *"Alternating renewal processes"* El modelo de Markov relaciona el estado del día actual con el estado de los N días anteriores; este N define el orden del modelo. Usualmente se utiliza un modelo de 1^{er} orden ([Richardson, 1981], [Wilks, 1989], [Wilks, 1992]), sin embargo una crítica que generalmente se le hace a este modelo es el hecho de que muchas veces es incapaz de reproducir períodos secos muy largos ([Racsko, 1991]).

El modelo de Markov de primer orden puede definirse en base a dos probabilidades:

P_{01} = probabilidad de tener un día de lluvia habiendo tenido un día seco el día anterior.

P_{11} = probabilidad de tener un día de lluvia habiendo tenido un día lluvioso el día anterior.

Notar que no es necesario calcular P_{00} y P_{10} ya que $P_{00} = 1 - P_{01}$ y $P_{10} = 1 - P_{11}$.

Luego, es posible reconstruir una serie completa de valores para días secos o lluviosos, tomando un número al azar cada vez y comparándolo con la probabilidad correspondiente. Supongamos que empezamos la serie arbitrariamente con un día seco; acto seguido, tomamos un número al azar entre 0 y 1; si el número es menor a P_{01} entonces el día siguiente es lluvioso, sino es seco. Ahora, si nos encontramos en la primer situación, el próximo número al azar deberíamos compararlo contra P_{11}. Repitiendo este proceso podemos generar una serie tan larga como sea necesario.

Los modelos del tipo *"Alternating renewal processes"* ven a la serie de valores observados como una secuencia de períodos lluviosos o secos alternados. Así, se busca asignar una probabilidad a la ocurrencia de períodos secos o lluviosos de distinta longitud. Cual es la probabilidad de tener 2 días secos seguidos, cual la de tener 3 y así siguiendo. Lo mismo para días de lluvia. Diferentes modelos han sido propuestos para ajustar la distribución de los períodos secos y lluviosos.

Nuestro modelo para la ocurrencia de días de lluvia, es de este último tipo. A partir de las observaciones se establecieron 12 distribuciones de períodos secos (1 por cada mes) y 12 de períodos lluviosos para cada una de las estaciones a estudiar. Claramente, la separación en meses es arbitraria, la idea es capturar el hecho de que la distribución de las lluvias varía a lo largo del año. Luego se utilizó un perceptrón multicapa con una capa escondida, una neurona en la capa de entrada y una neurona en la capa

de salida, para ajustar cada una de estas distribuciones. Los patrones de entrenamiento para cada red fueron definidos de la siguiente manera: como entrada la probabilidad observada y como salida esperada la longitud del período (en cantidad de días) que corresponde a esa probabilidad.

La ventaja de utilizar un perceptrón multicapa es que, debido a su capacidad para representar cualquier función continua (ver Sección 3.1), este mismo modelo puede utilizarse para estaciones meteorológicas con regímenes de lluvias muy distintos, sin la necesidad de asumir una distribución a priori para la ocurrencia de días de lluvia. La capacidad de aproximador universal de los perceptrones multicapa está sujeta a la elección de la arquitectura (en particular la cantidad de neuronas en la capa escondida) que permiten realizar con éxito esta tarea. En este caso, la arquitectura de cada una de las redes fue determinada utilizando estadística bayesiana. Concretamente, para cada distribución, se entrenaron distintas redes con un número creciente de neuronas en su capa escondida (empezando por 3 neuronas) y se las comparó según el logaritmo del "evidence" de sus hiperparámetros (ver Sección 3.3). Luego de realizar varios experimentos sobre distintas estaciones de la Argentina, se determinó un número máximo de 14 neuronas.

Para generar una serie completa de ocurrencias de días de lluvia, podemos comenzar nuevamente suponiendo que el primer día de la serie es un día seco, luego tomamos un número al azar y evaluamos la red correspondiente (en este caso la red para períodos secos del mes de enero) en este valor. La red, devolverá el valor asociado a la probabilidad ingresada (tomada al azar) y de esta forma, conseguimos la longitud del primer período seco. Ahora, deberíamos encontrar la longitud del período lluvioso que sigue al primer período seco en la serie; nuevamente tomamos un número al azar y evaluamos la red correspondiente en este punto (en cada paso se debe verificar el mes en el que comienza el período, según la cantidad de días simulados hasta el momento). De esta manera, alternando períodos secos y lluviosos es posible construir toda la serie.

Para verificar el desempeño del modelo, se tomará como modelo de referencia el proceso de Markov de primer orden.

4.2. Cantidad de Precipitaciones

Generalmente la cantidad de precipitaciones diaria es modelada como un fenómeno independiente. Si bien, la autocorrelación de la cantidad de lluvias puede ser estadísticamente significativa, en la práctica, la capacidad de modelar o no esto, tiene consecuencias despreciables ([Wilks, 1999b]). Tradicionalmente, para modelar la cantidad de precipitaciones se utilizan funciones paramétricas que permiten ajustar la función de densidad acumulativa (CDF) de la cantidad de precipitaciones durante los días de lluvia.

En este caso se tomarán como modelos de referencia tres distribuciones: la distribución Gama ([Richardson, 1981], [Wilks, 1989], [Wilks, 1992]), Weibull ([Geng, S., 1986], [Selker, 1990]) y Doble Exponencial ([Wilks, 1998], [Wilks, 1999b]).

- Gama:

$$f(x) = \frac{(\alpha/\beta)^{\alpha-1} exp(-x/\beta)}{\beta\Gamma(\alpha)} \quad con\, x, \alpha, \beta > 0 \tag{4.1}$$

- Weibull:

$$f(x) = ba^{-b}x^{b-1} exp\left[-\left(\frac{x}{a}\right)^{b}\right] \quad con\, x, a, b > 0 \tag{4.2}$$

- Doble Exponencial:

$$f(x) = \left(\frac{\alpha}{\beta_1}\right) exp\left[-\frac{x}{\beta_1}\right] + \frac{1-\alpha}{\beta_2} exp\left[-\frac{x}{\beta_2}\right] \qquad con\, x, \alpha, \beta_1, \beta_2 > 0. \tag{4.3}$$

Nuevamente, se tomaron las distribuciones para cada mes en forma separada. Para obtener la serie de precipitaciones, en primer lugar es necesario determinar los parámetros que permiten ajustar las CDFs observadas (1 para cada mes) con cada una de las funciones paramétricas, utilizando algún método de regresión lineal. Luego, tomamos la serie de ocurrencia de días de lluvia y para cada día de lluvia tomamos un número al azar en forma independiente. Por último, evaluamos la función correspondiente en este valor (que representa una probabilidad acumulativa), de acuerdo al mes; el valor obtenido representa la cantidad de lluvia caída ese día.

En el *NNGEN-P*, para ajustar cada una de las CDFs observadas, se utilizó un perceptrón multicapa con una capa escondida, una neurona de entrada que representa una probabilidad dada y una neurona de salida que representa la cantidad de lluvia asociada a dicha probabilidad. Al igual que en el caso de la ocurrencia, la arquitectura de las redes fue determinada utilizando estadística bayesiana. Para obtener la serie de precipitaciones, igual que antes, se debe tomar un número al azar para cada día de lluvia y evaluando la red correspondiente según el mes en este punto es posible obtener la cantidad de lluvia que corresponde a ese día.

Cabe destacar que cada una de las distribuciones tomadas como modelos de referencia han demostrado un desempeño aceptable en determinadas regiones. Nuevamente, la ventaja de utilizar una red neuronal para representar las distribuciones de los datos, nos permite desarrollar un modelo que potencialmente podrá utilizarse sobre cualquier régimen climático sin la necesidad de asumir a priori la forma en la que deberían distribuirse las observaciones.

Dado la importancia que tiene la varianza mensual y anual en muchas de las aplicaciones de los generadores de precipitaciones, la capacidad de nuestro modelo en reproducir estos valores será una de las principales medidas para evaluar su desempeño y compararlo con los modelos de referencia.

4.3. Temperatura Máxima y Mínima

Si bien, para modelar los índices de temperatura existen una gran cantidad de modelos, la mayoría de estos fallan en reproducir la correlación temporal que generalmente presentan los índices de temperatura ([Caballero, 2001]). El procedimiento descripto por [Richardson, 1981] y [Matalas, 1967], asume que la temperatura diaria (tanto la mínima como la máxima) es un proceso continuo, multivariado y débilmente estacionario ([Mitchell, 2000]).

Siguiendo el algoritmo propuesto por Richardson, en primer lugar, debemos quitarle el ciclo estacional a la serie. Para esto, se calcularon, para cada día del año, la media y la desviación estándar, tomando los valores para días secos y días lluviosos en forma separada. Luego, se utilizó un perceptrón multicapa con una capa escondida para ajustar las dos distribuciones, consiguiendo de esta manera una función suave que descarta el ruido de las series (en el algoritmo original, se utilizan con el mismo fin series de Fourier). A partir de ahora, se utilizarán, como media y desviación estándar diaria, los valores generados por la red.

Los valores que resultan de sacarle el ciclo estacional a las series de temperatura, son denominados "residuos":

$$R^t = (T^t - M)/S \tag{4.4}$$

donde T es la temperatura del día t (mínima o máxima), M es la media diaria (ya sea para días secos o días de lluvia según corresponda con el estado del día t) y S es la desviación estándar. Se asume que dichos residuos tienen una distribución normal y pueden describirse siguiendo un proceso de autorregresión lineal de primer orden:

$$X^{t+1} = AX^t + B\varepsilon^{t+1} \tag{4.5}$$

donde X^t es un vector columna que contiene el residuo de la temperatura máxima y el residuo de la temperatura mínima. ε es un error gausseano con media 0 y varianza 1. A y B están definidas tal que la serie producida mantenga la correlación entre los dos índices:

$$A = M_1 M_0^{-1} \tag{4.6}$$

$$BB^T = M_0 - M_1 M_0^{-1} M_1^T. \tag{4.7}$$

M_0, denominada matriz de "cross-correlation", contiene los coeficientes de correlación entre la temperatura mínima y la temperatura máxima sin "laguear". M_1, denominada matriz de "serial-correlation", contiene los coeficientes de correlación "lagueados" un día:

$$M_0 = \begin{pmatrix} 1 & P_0(tmin, tmax) \\ P_0(tmin, tmax) & 1 \end{pmatrix}$$

$$M_1 = \begin{pmatrix} P_1(tmin, tmin) & P_1(tmin, tmax) \\ P_1(tmax, tmin) & P_1(tmax, tmax) \end{pmatrix}$$

donde $P_0(tmin, tmax)$ representa la correlación entre las series de temperaturas mínima y máxima, sin "laguear" y $P_1(tmin, tmax)$ representa la correlación entre las series con un "lag" de un día.

De esta manera el modelo permite generar estocásticamente una serie de residuos, empezando con valores arbitrarios para t=0 y utilizando la ecuación 4.5. Notar que la aleatoriedad del proceso radica en el término que representa al error. Para obtener los valores de las series de temperatura máxima y temperatura mínima se le suma a los residuos simulados la media diaria y el resultado se multiplica por la desviación estándar del día correspondiente.

Basándonos en los criterios utilizados para el desarrollo de otros modelos, nuestra hipótesis será que la temperatura de un día determinado depende del estado del día (seco o lluvioso) y de la temperatura de los días anteriores. Dado que utilizaremos una red recurrente simple para aprender las series de residuos correspondientes a la temperatura máxima y a la temperatura mínima, no es necesario fijar a priori la cantidad de días hacia atrás que deben mirarse para generar los valores de un día determinado. Más aún, la correlación temporal de las series puede variar a lo largo del año. Esta es la principal diferencia con el modelo de Richardson que siempre mira sólo un día hacia atrás. La ventaja principal sobre los otros modelos es la posibilidad de contar con un modelo que pueda aplicarse sobre distintos regímenes de temperatura que presentan diferencias significativas en los valores de correlación temporal. En cuanto al estado del día (seco o lluvioso), otros modelos, en lugar de mirar el estado del día a predecir, tienen en cuenta el estado del día anterior o los dos en conjunto ([Srikanthan, 2001]).

Dado que *NNGEN-T* está basado en el modelo de Richardson, similarmente, la serie es modelada de la siguiente manera:

$$X^{t+1} = Srn(X^t ... X^{t-N}) + A\varepsilon^{t+1} \tag{4.8}$$

donde Srn es una red recurrente simple y A es una matriz que permite correlacionar los errores correspondientes a los dos índices. Una diferencia importante con respecto al modelo de Richardson es que en este caso, los residuos serán producidos sin diferenciar el ciclo estacional para días secos y días de lluvia. Esta información estará contenida en la red.

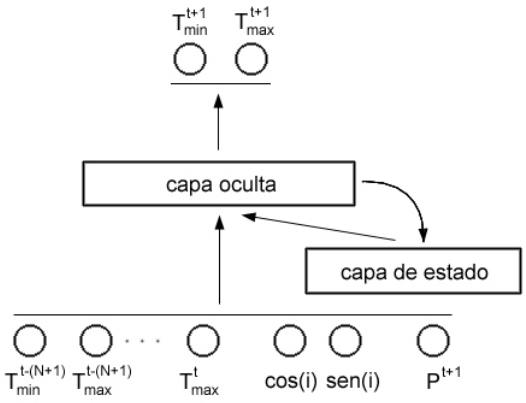

Figura 4.1: Esquema de la red del generador de temperaturas. T^t_{min} representa la temperatura mínima para el día t, T^t_{max} representa la temperatura máxima para el día t, $i = (2 * \pi)/365 * d$, d es la posición del día en el año (1 a 365) y P^t es el estado del día t (0=seco, 1=lluvioso).

La red tiene como entrada, los residuos de la temperatura mínima y máxima de los últimos N días (a este N lo llamaremos “delay”) y como salida los residuos de la temperatura mínima y máxima del día que se desea predecir. Además la capa de entrada tiene una neurona con el estado de este día (seco o lluvioso) y dos neuronas que representan la posición del día respecto de la época del año (ver Figura 4.1).

Para entender como se representa la posición de un día en el año, pensemos en la representación del 31 de Diciembre (día número 365):

$$cos((2 * \pi)/365 * 365) = 1$$
$$sen((2 * \pi)/365 * 365) = 0.$$

Ahora, si tomamos la representación del 1 de Enero del año siguiente, nos queda:

$$cos((2 * \pi)/365 * 1) = 0{,}9999$$
$$sen((2 * \pi)/365 * 1) = 0{,}0172.$$

Como puede verse, los valores para los dos días son muy parecidos, esto es consistente con el hecho de que son días consecutivos. De esta forma, esta representación nos permite conservar la idea de un ciclo anual.

Como el lector habrá notado, en este caso se utilizará una única red para todo el año. Esto podría verse como una dificultad, dado que probablemente la distribución de los índices de temperatura, varía a lo largo de un año y la red deberá aprender las diferentes distribuciones subyacentes. Sin embargo, si tomáramos una red por cada mes, perderíamos información relativa a la correlación temporal. Por ejemplo, si consideramos una red que es entrenada para reproducir las temperaturas del mes de enero, el conjunto de entrenamiento estará constituido por los valores de todos los eneros, el problema aquí es que se perdería la correlación temporal entre los primeros días de enero y los últimos días del mes de diciembre.

Volviendo a la arquitectura de la red, la idea de implementar un “delay” en la capa de entrada es representar con este una memoria a corto plazo, y dejar la capa de estado o contexto para guardar información relacionada con una memoria a más largo plazo. Luego de una serie de pruebas sobre diferentes estaciones de la Argentina, se estableció un “delay” de 5 días para todas las estaciones, debido a que para valores mayores a 5 no se observaron mejoras significativas. Cabe destacar que para las estaciones cuyas

series presentan una memoria a mas largo plazo, una red sin “delay” no logra reproducir adecuadamente la autocorrelación.

Como criterio para seleccionar la mejor arquitectura de red se determinó la capacidad de la red en reproducir series que conserven la correlación temporal de las observaciones. Luego de realizar mediciones para distinta cantidad de neuronas, se estableció un número fijo de 15 neuronas para la capa escondida de todas las redes. Concretamente, para cada estación se entrenan 10 redes distintas con la misma arquitectura y se selecciona la que mejor reproduce la correlación temporal de los datos. Como método de entrenamiento se utilizó el algoritmo de “truncated back propagation” (ver Sección 3.2)

La utilización de una red recurrente permite aumentar la sensibilidad de la red a la memoria que presenta la serie, con sólo aumentar la cantidad de neuronas en la capa oculta o bien aumentar el “delay”. En el primero de estos dos casos, un incremento en la dimensión de la capa oculta (y por consiguiente de la capa de estado), permite almacenar mayor información de contexto y con ello la red puede considerar una mayor cantidad de información hacia atrás. En el segundo caso, aumentando el “delay”, forzamos explícitamente a la red a observar N valores hacia atrás dejando el contexto para mantener información relevante a más largo plazo. El problema con esto es que, al aumentar la dimensión de la capa de entrada, es posible que necesitemos una mayor cantidad de patrones en el conjunto de entrenamiento. La elección de una arquitectura de red que permita resolver el problema, y que además lo haga en un tiempo razonable, deberá tener en cuenta estos aspectos.

Capítulo 5

Resultados

5.1. Ocurrencia de Días de Lluvia

Para las 19 estaciones de la Argentina se calcularon las probabilidades de transición del método de Markov de primer orden para cada mes. Luego, fueron computadas las probabilidades empíricas para secuencias de días secos y lluviosos comenzando en un mes determinado y con un largo de N días. A partir de estas probabilidades se determinó la función de densidad acumulativa (CDF) de períodos secos y lluviosos correspondiente a cada mes. Como se mencionó en la Sección 4.1 distintos perceptrones simples fueron entrenados para reproducir dichas CDFs.

En primer lugar se comparó la capacidad de *NNGEN-P* y de Markov para realizar un "fit" de las distribuciones. Para esto se utilizó un gráfico denominado PP-Plot (Probability Probability Plot) que compara, la probabilidad observada con la probabilidad asignada por el modelo para un mismo valor (en este caso, dichos valores representan la longitud de un período seco o lluvioso, dependiendo del gráfico). De esta manera, si trazamos una recta con origen en el (0, 0) y pendiente 1, los puntos que caen por arriba de esta recta deben interpretarse como una subestimación del modelo y los valores que caen por debajo de la recta como una sobreestimación. Así, puntos más cercanos a la recta significan que el modelo ajusta mejor la CDF de la serie observada. La Figura 5.1 muestra una leve mejora del *NNGEN-P* con respecto a Markov.

La reproducción de la varianza interanual es una característica deseable en una gran cantidad de aplicaciones. Sin embargo, la mayoría de los modelos subestiman la varianza a escala mensual (este fenómeno es denominado "overdispersion") ([Wilks, 1999a]). Con el objetivo de medir la capacidad del modelo para simular la varianza interanual del total de precipitaciones para cada mes, es posible descomponer esta varianza en dos componentes:

$$V(P) = E(N)\sigma^2 + V(N)\mu^2 \tag{5.1}$$

donde P es el total de precipitaciones caídas en el mes, N es la cantidad de días de lluvia, $E()$ es el operador "esperanza", $V()$ es el operador "varianza", y μ y σ son la media y la desviación estándar de las precipitaciones diarias del mes.

Considerando que, tanto Markov como *NNGEN-P* son capaces de reproducir adecuadamente la media mensual de ocurrencias ($E(N)$), en esta sección nos enfocaremos en la capacidad de los modelos en reproducir la varianza de la cantidad de días de lluvia ($V(N)$) . La Figura 5.2 no muestra diferencias significativas entre los dos modelos.

$$Variance\,Overdispersion = \left[\frac{varianza\,observada}{varianza\,simulada} - 1\right] \times 100\,\% \tag{5.2}$$

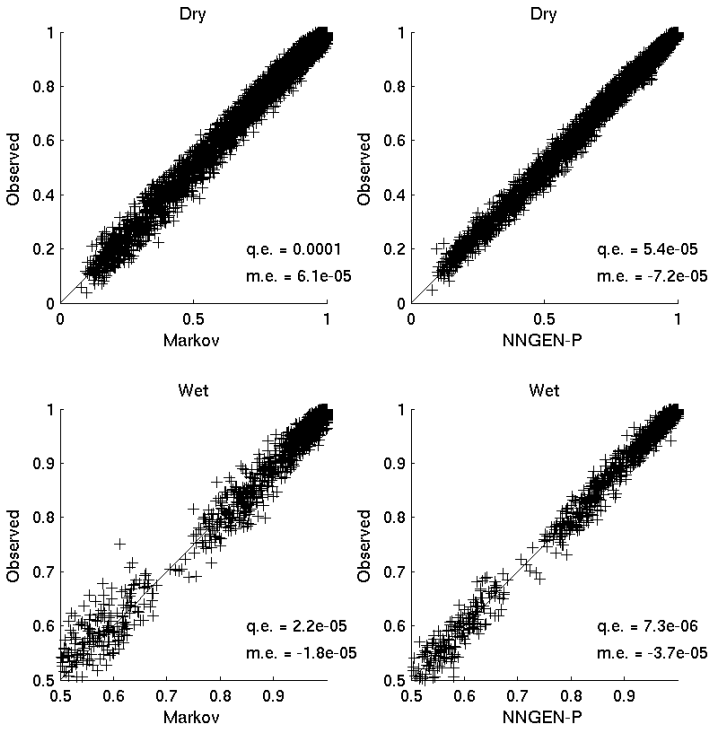

Figura 5.1: PP-Plot de las 19 estaciones de la Argentina para períodos secos y lluviosos donde q.e. es el error cuadrático y m.e. el error medio. Cada mes fue considerado en forma separada.

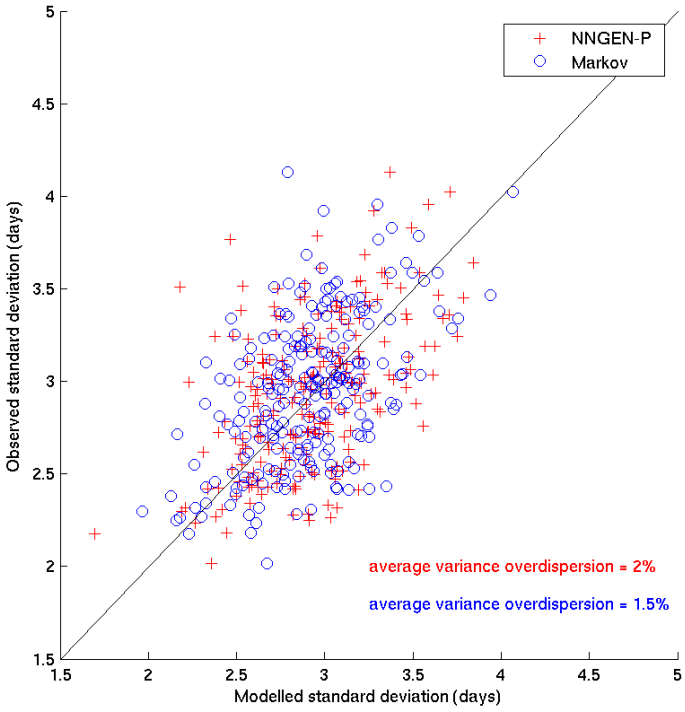

Figura 5.2: Comparación entre la desviación estándar interanual para la cantidad de días de lluvia observada en cada mes vs. la simulada. Un punto por cada mes por cada estación para las 19 estaciones de la Argentina.

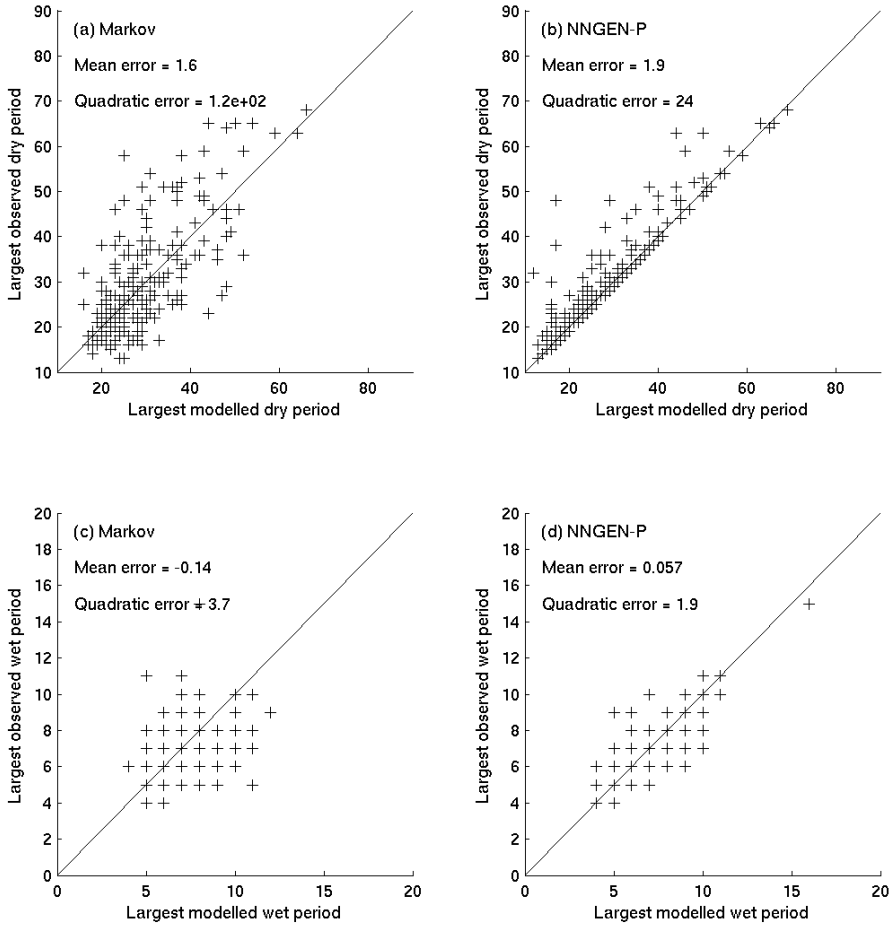

Figura 5.3: Comparación entre los valores observados y los valores simulados para las 19 estaciones de la Argentina. Cada punto representa la longitud (en cantidad de días) del período más largo (seco o lluvioso) de una estación en un mes determinado.

La capacidad de los modelos en reproducir períodos secos y lluviosos largos es de gran importancia para los modelos de crecimiento de plantas ([Racsko, 1991]), Fundamentalmente, la existencia de largos períodos de sequía puede afectar seriamente la producción agrícola. Si comparamos la capacidad de los modelos en simular los períodos secos y lluviosos más largos de cada mes (Figura 5.3), podemos observar que el modelo de Markov tiende al subestimar los períodos secos más largos y a sobreestimar los períodos lluviosos. En cuanto a *NNGEN-P*, este reproduce mejor que Markov tanto los períodos secos como los lluviosos. Esto es consistente con el hecho de que Markov, debido a su distribución exponencial, tiende a sobreestimar los períodos cortos y a subestimar los períodos largos. *NNGEN-P*, dado que no es un modelo paramétrico, tiene la capacidad de reproducir distribuciones de diferente naturaleza.

Cabe destacar que *NNGEN-P* nunca sobreestima la longitud de los períodos secos. Este hecho probablemente se deba a su naturaleza estocástica. Dada una simulación de 100 años, la probabilidad de que el *NNGEN-P* llegue a simular períodos tan largos como los extremos observados, puede ser relativamente baja. Si tomáramos simulaciones más largas, esta probabilidad aumentaría y de esta manera observaríamos como este fenómeno desaparece.

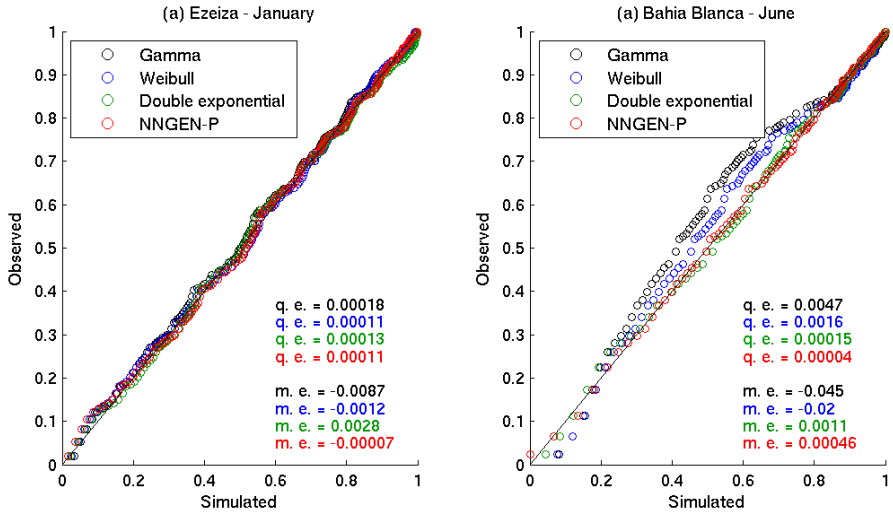

Figura 5.4: (a) PP-Plot para la cantidad de precipitaciones del mes de enero de la estacion Ezeiza (87576). (b) PP-Plot para la cantidad de precipitaciones del mes de junio de la estación de Bahia Blanca (87750).

5.2. Cantidad de Precipitaciones

La capacidad de las funciones paramétricas de ajustar la función de densidad acumulativa (CDF) de precipitaciones varía de acuerdo a la estación y al mes considerado. En ciertos casos encontramos que tanto las funciones paramétricas como la red realizan un "fit" casi perfecto (Figura 5.4 (a)) de los datos. En otros en cambio, la diferencia resulta considerable (Figura 5.4 (b)).

Al contrario de lo que pasa con las funciones paramétricas, la habilidad de *NNGEN-P* para ajustar las distribuciones observadas, no depende de la estación ni del mes a tener en cuenta (en particular no depende de la forma de la distribución). Esto puede verse en la Figura 5.5, donde el error que muestra *NNGEN-P* es mucho menor al de las funciones paramétricas. En particular, las distribuciones Gamma y Weibull sobreestiman las cantidades más pequeñas (entre 0 y 0.2) y subestiman las cantidades más grandes (entre 0.4 y 0.8). La Doble Exponencial muestra un error mucho menor al de las otras dos funciones paramétricas aunque en general sigue siendo un poco peor que nuestro modelo, salvo para las probabilidades mas pequeñas (entre 0 y 0.1) donde *NNGEN-P* muestra una mayor dispersión.

Estas diferencias, en cuanto a la capacidad de los modelos en reproducir adecuadamente la distribución de las observaciones, resaltan aún más utilizando un gráfico QQ-Plot (Quantity Quantity Plot) que compara la cantidad observada con la simulada para cada valor de probabilidad acumulativa. (Figura 5.6). Aquí puede verse como, en general, el error aumenta a medida que aumenta la cantidad de precipitaciones y puede observarse como *NNGEN-P* se queda mucho más pegado a la recta.

Considerando que, todos los modelos son capaces de reproducir la media mensual de cantidad de precipitaciones (μ en la ecuación 5.1) bastante bien, en esta sección nos enfocaremos en la capacidad de los modelos en reproducir la desviación estándar de la cantidad de precipitaciones (σ en la ecuación 5.1) para cada mes. Las Figuras 5.7 y 5.8 no muestran diferencias significativas entre el uso de Markov y el uso de *NNGEN-P* para la etapa de ocurrencia, esto es consistente con lo observado en la Sección 5.1, ya que nuestro modelo sólo se diferencia significativamente del modelo de Markov en su capacidad para reproducir períodos largos. En este caso, las distribuciones Gamma y la Doble Exponencial tienden a subestimar la desviación estándar interanual mientras que Weibull subestima los valores más pequeños y sobreestima los valores más grandes. *NNGEN-P*, presenta un pequeño bias pero en general muestra un comportamiento mucho mejor.

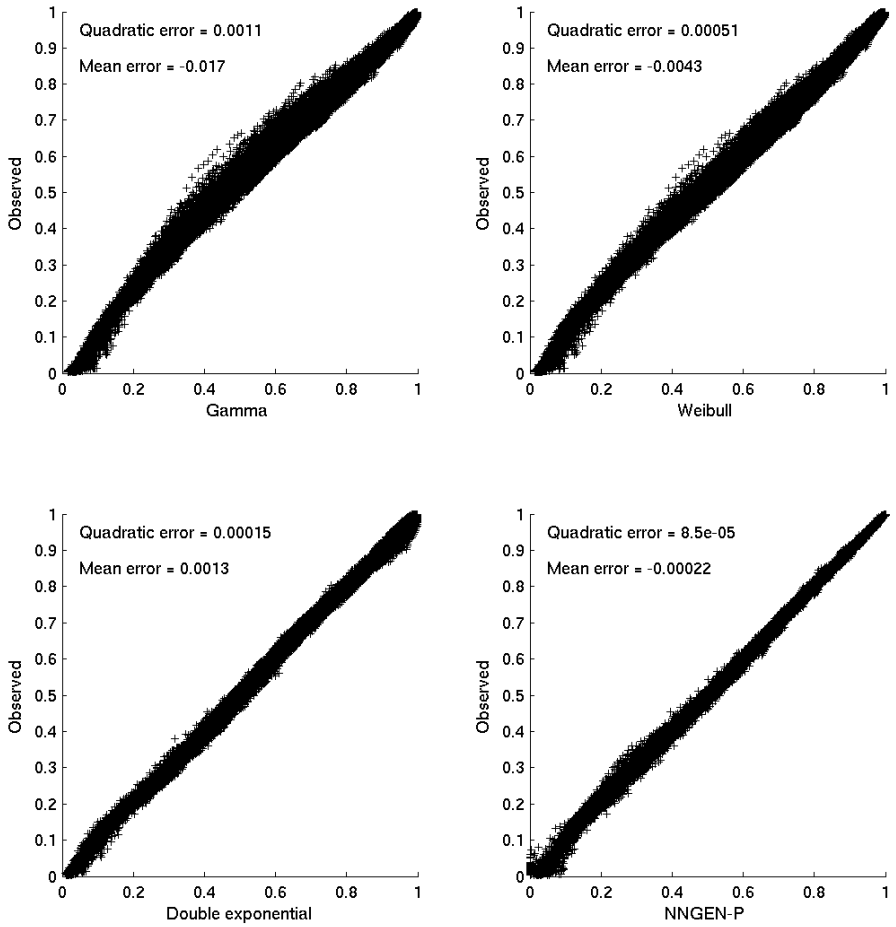

Figura 5.5: PP-Plot para la cantidad de precipitaciones de las 19 estaciones de la Argentina. Cada punto representa la probabilidad observada vs la probabilidad simulada para una cantidad determinada en una estación en un mes en particular.

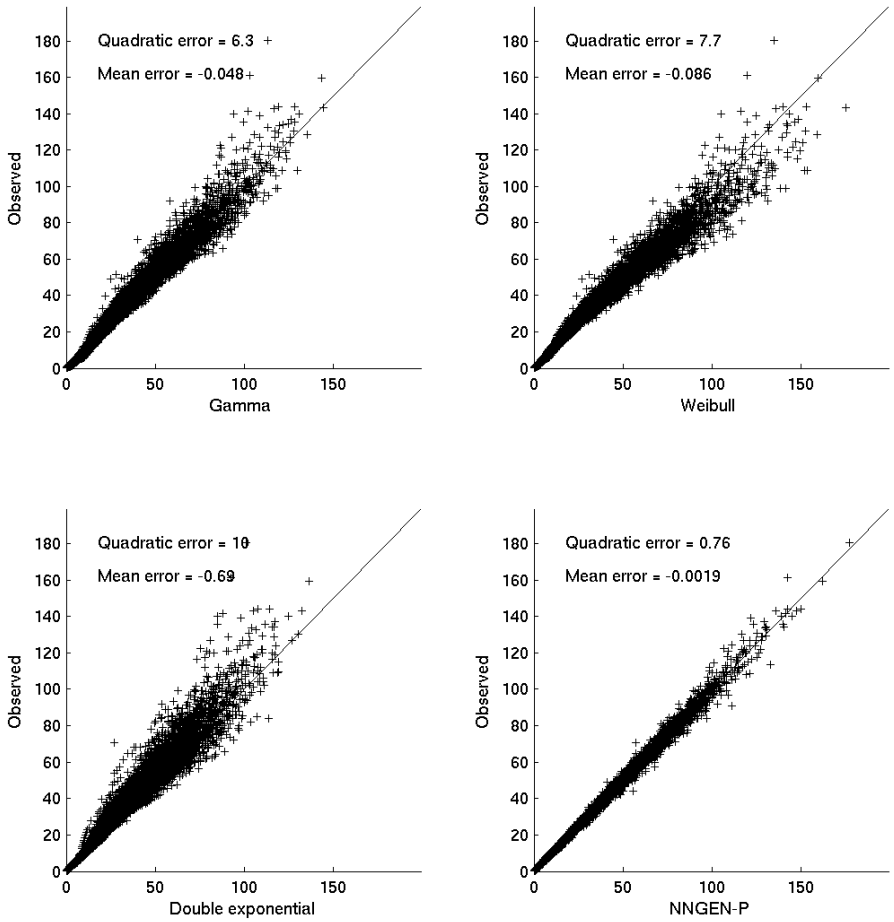

Figura 5.6: QQ-Plot para la cantidad de precipitaciones de las 19 estaciones de la Argentina. Cada punto representa la cantidad observada vs. la cantidad simulada para una probabilidad acumulativa determinada en una estación en un mes en particular.

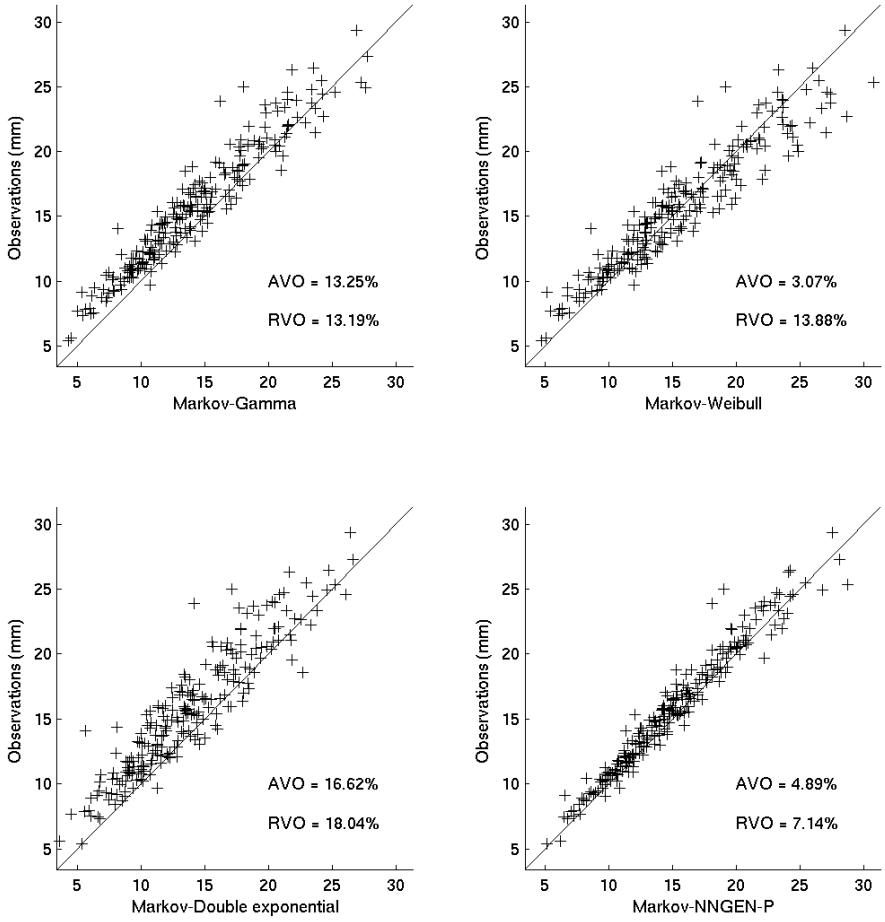

Figura 5.7: Comparación entre la desviación estándar interanual a escala mensual de la cantidad de precipitaciones observada vs. la simulada. Utilizando el modelo de Markov para realizar la simulación de la ocurrencia de días de lluvia. AVO = “Average Variance Overdispersion” y RVO = “Standard Deviation Variance Overdispersion”

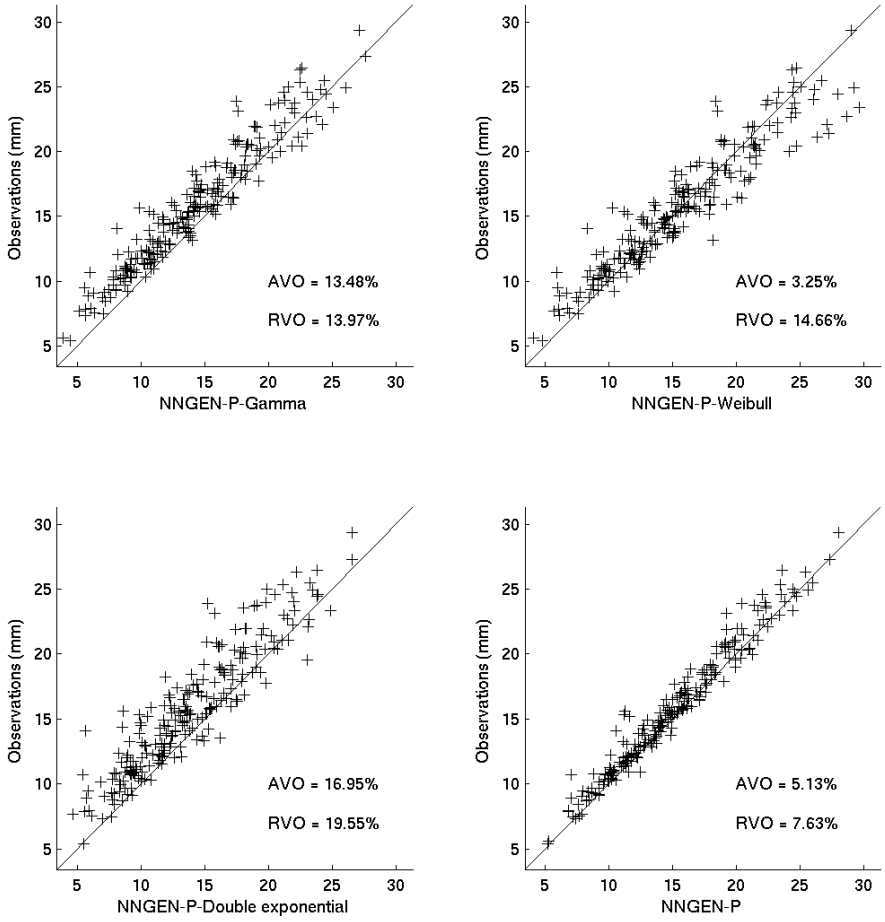

Figura 5.8: Comparación entre la desviación estándar interanual a escala mensual de la cantidad de precipitaciones observada y la simulación. Utilizando *NNGEN-P* para realizar la simulación de la ocurrencia de días de lluvia. AVO = “Average Variance Overdispersion” y RVO = “Standard Deviation Variance Overdispersion”

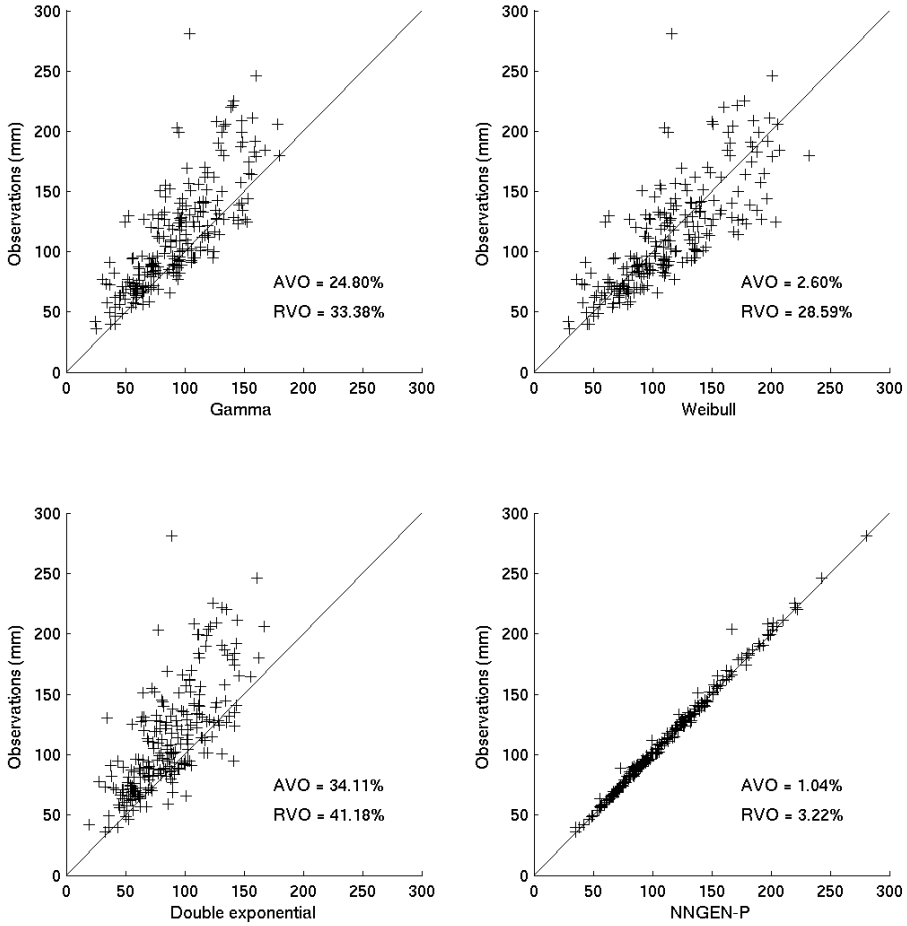

Figura 5.9: Comparación entre las precipitaciones extremas simuladas vs. las observadas para cada estación en cada uno de los meses del año para las 19 estaciones de la Argentina.

Para determinar la capacidad de los modelos para reproducir cantidades extremas (Figura 5.9), se compararon los valores de mayor probabilidad acumulativa entre los modelos y las observaciones para cada mes y para cada estación. De esta forma la comparación resulta independiente de la simulación. Dado que para una probabilidad acumulativa igual a 1, las funciones paramétricas tienden a infinito, se tomaron los valores asociados a la probabilidad $\frac{(N-1/3)}{(N+1/3)}$, con N el número de observaciones ([Wilks, 1999b]). En la figura 5.9 puede verse una diferencia importante entre *NNGEN-P* y las distintas funciones paramétricas. En particular, se observa que las distribuciones Gamma y Doble Exponencial tienden a subestimar los valores extremos. Estos resultados son consistentes con lo que habíamos observado en la Figura 5.6.

Finalmente, para verificar la capacidad del modelo para reproducir el régimen de lluvias de distintos tipos de climas, se utilizaron estaciones de Europa, Estados Unidos y los Trópicos. En las figuras 5.10, 5.11 y 5.12 puede observarse que *NNGEN-P* ajusta mucho mejor las funciones de probabilidad acumulativa.

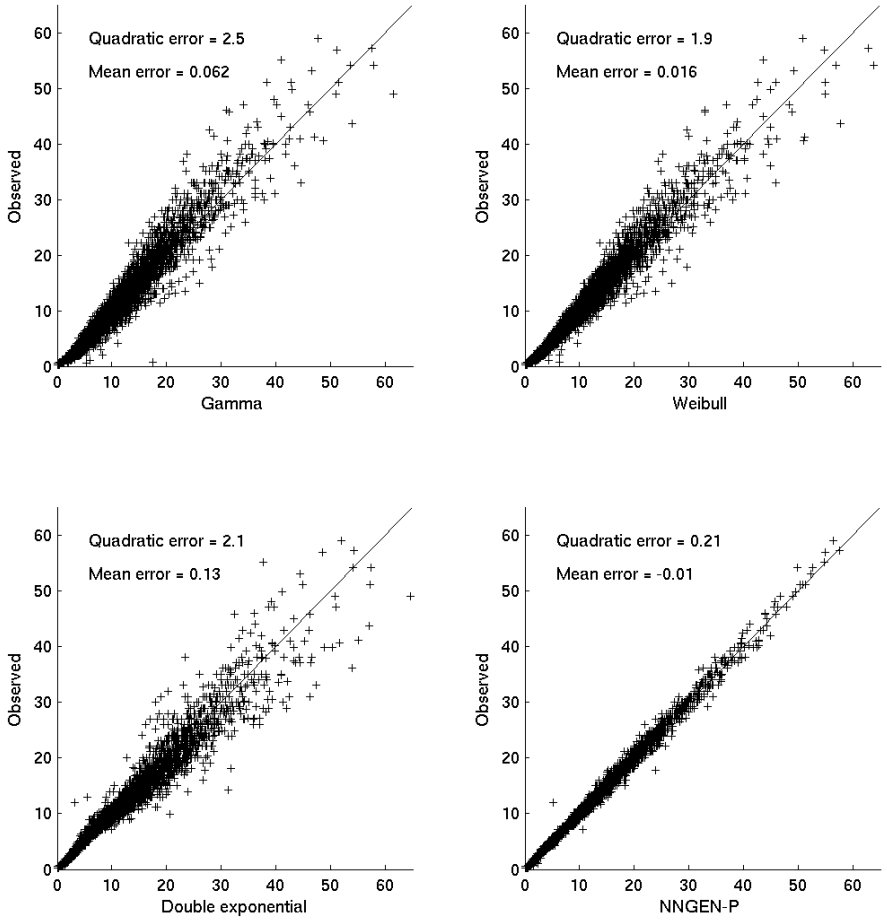

Figura 5.10: QQ-Plot para la cantidad de precipitaciones de las estaciones de Europa. Cada punto representa la cantidad observada vs. la cantidad simulada para una probabilidad acumulativa determinada en una estación en un mes en particular.

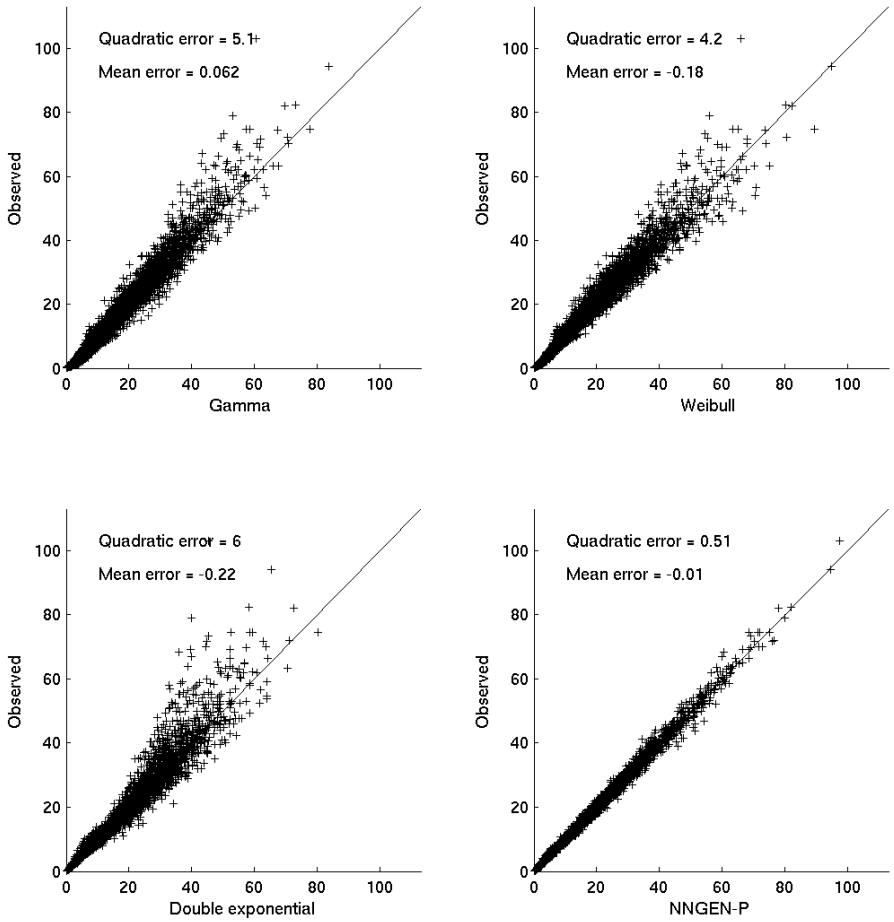

Figura 5.11: QQ-Plot para la cantidad de precipitaciones de las estaciones de Estados Unidos. Cada punto representa la cantidad observada vs. la cantidad simulada para una probabilidad acumulativa determinada en una estación en un mes en particular.

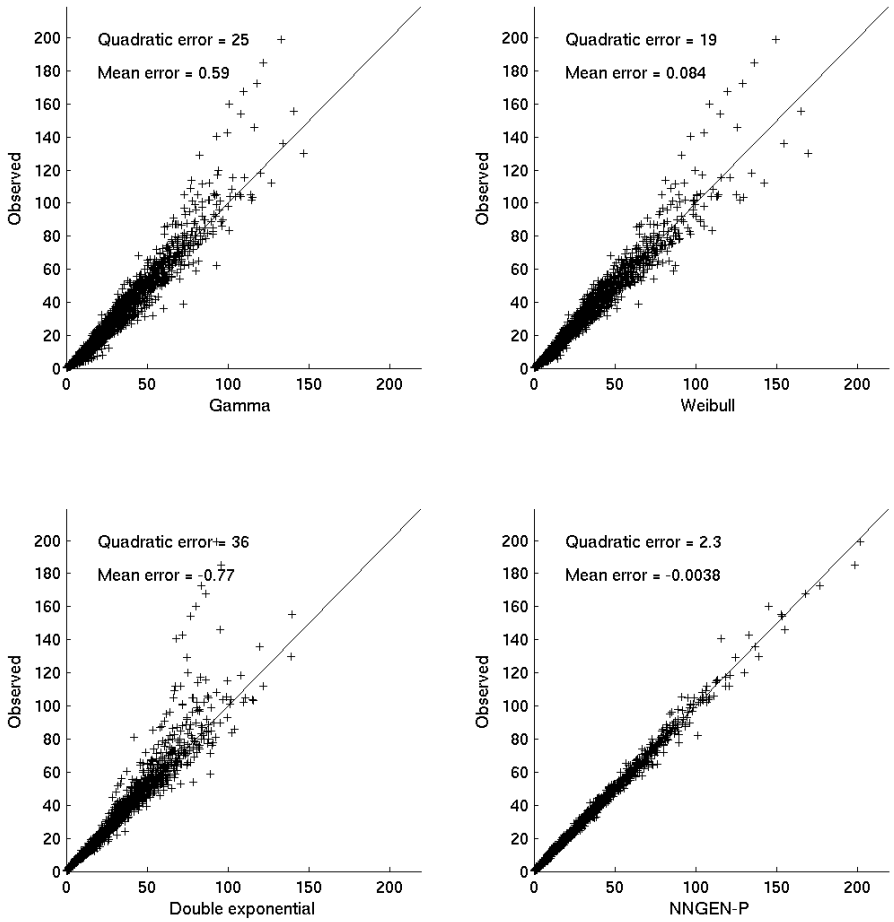

Figura 5.12: QQ-Plot para la cantidad de precipitaciones de las estaciones de los Trópicos. Cada punto representa la cantidad observada vs. la cantidad simulada para una probabilidad acumulativa determinada en una estación en un mes en particular.

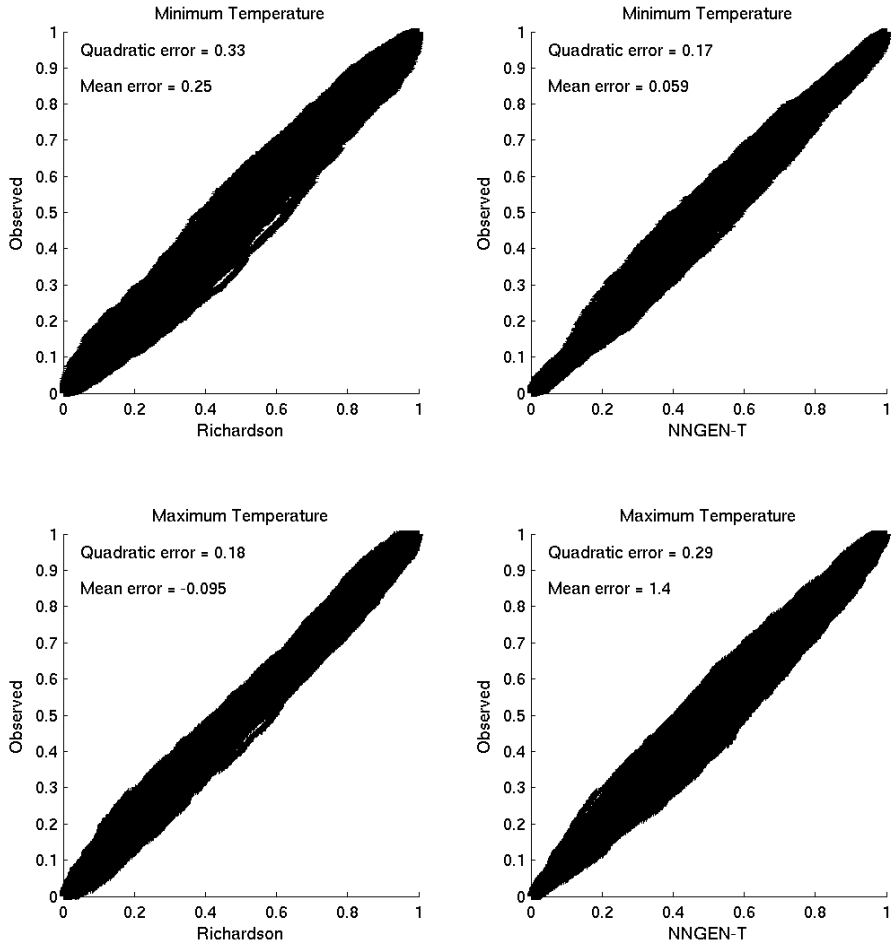

Figura 5.13: PP-Plot para la temperatura máxima y mínima de las 19 estaciones de la Argentina. Cada punto representa la probabilidad asignada por el modelo respecto de la probabilidad acumulativa observada para un valor determinado, en una estacíon, en un mes en particular.

5.3. Temperatura

En primer lugar, al igual que con el modelo de precipitaciones, se analizó la capacidad de cada modelo para ajustar las distribuciones empíricas de las series de temperatura mínima y máxima. Para esto se realizaron, para las series de residuos (series a las que se les saco el ciclo estacional), un gráfico PP-Plot (Figura 5.13) y un QQ-Plot (Figura 5.14). En este caso no se observaron diferencias significativas entre los dos modelos.

La capacidad de los modelos en reproducir los valores extremos de temperatura es especialmente importante en modelos de desarrollo de plantas ([Racsko, 1991]). Dado que los modelos de temperatura, a diferencia de los de precipitaciones, no se basan en realizar un “fit” de la distribución de las series, para compararlos, se tomarán los valores máximos y mínimos simulados. (En el caso de precipitaciones, se consideraron extremos, los valores asociados a la probabilidad $(N - 1/3)/(N + 1/3)$ en la CDF dada por el modelo). En la Figura 5.15, puede verse que si bien las diferencias entre los modelos no son significativas, el error de *NNGEN-T* es levemente inferior al error del modelo de Richardson. Sin embargo, ambos modelos tienden a sobreestimar las temperaturas máximas y subestimar las mínimas.

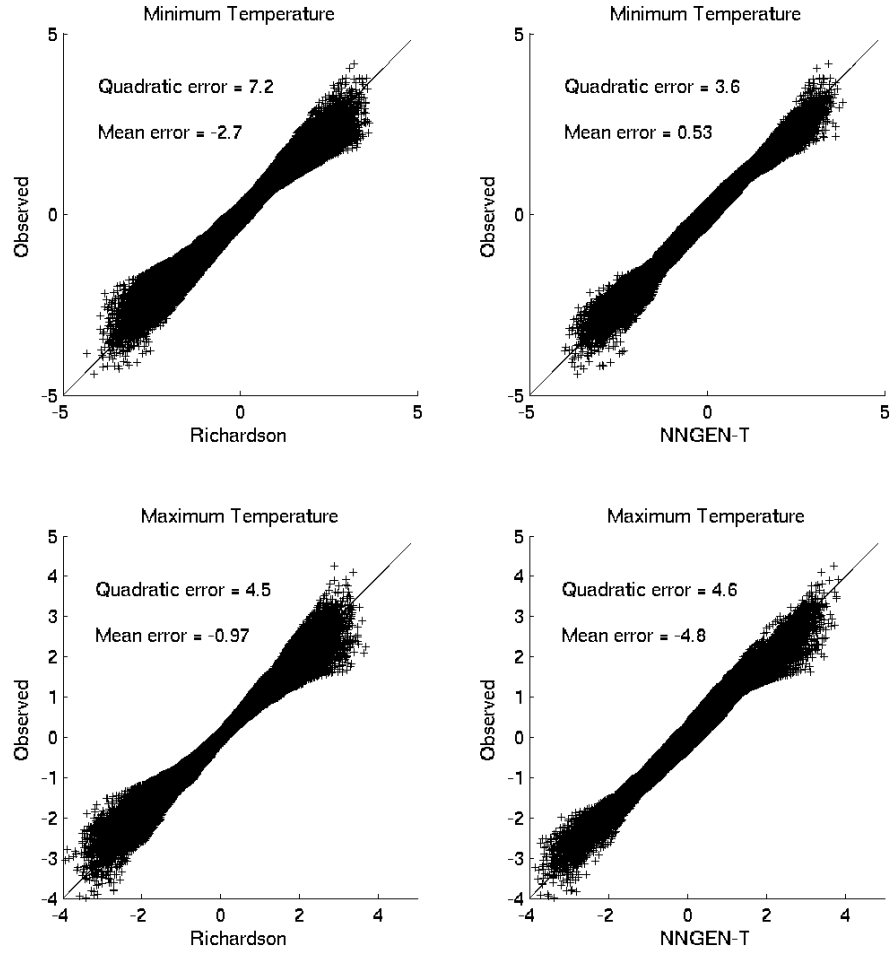

Figura 5.14: QQ-Plot para la temperatura máxima y mínima de las 19 estaciones de la Argentina. Cada punto representa la cantidad asociada a una probabilidad determinada, por el modelo respecto de las observaciones, en una estación, en un mes en particular.

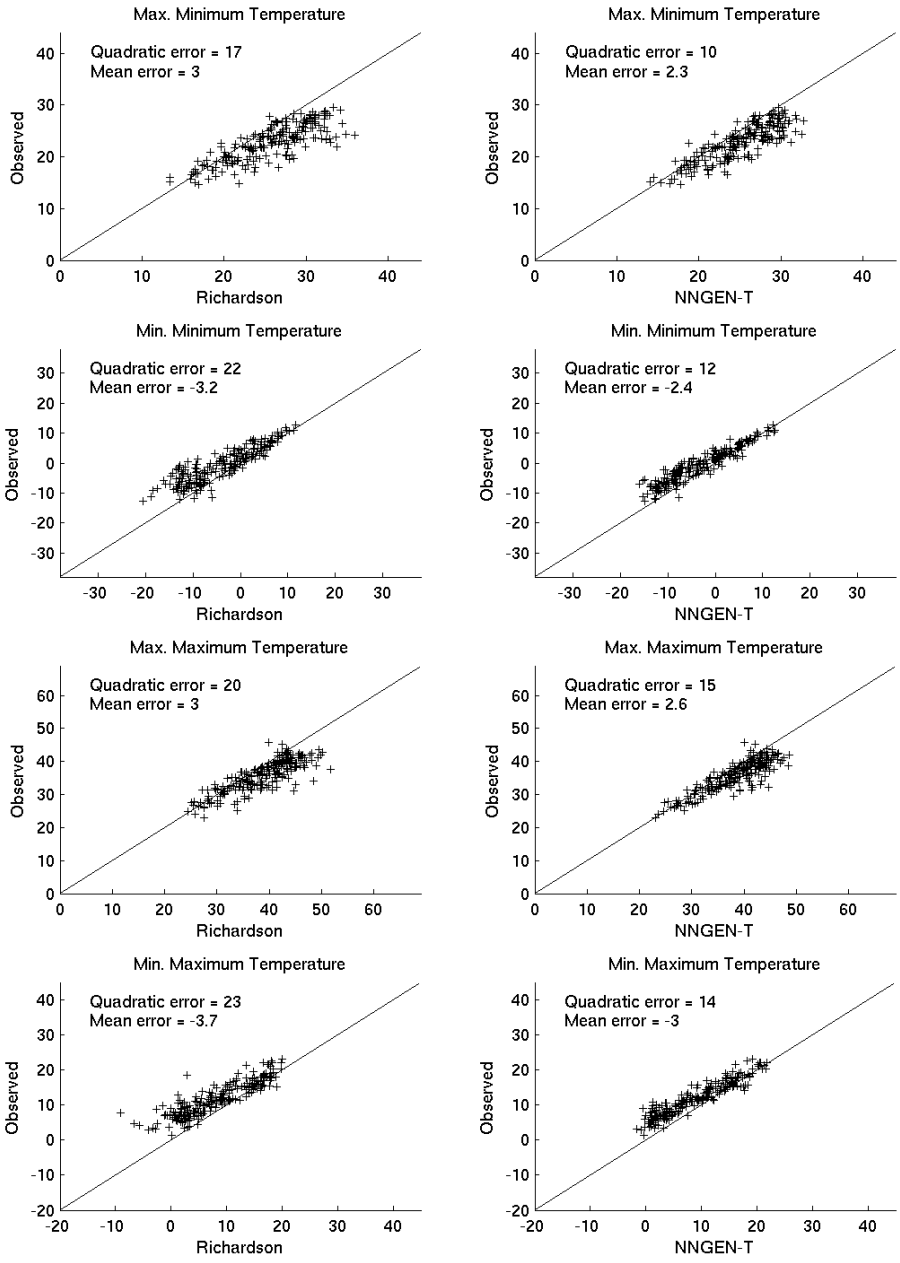

Figura 5.15: Comparación entre los valores mínimos y máximos modelados vs. los observados, para cada mes, para las temperaturas mínimas y máximas, para las 19 estaciones de la Argentina.

Con el objetivo de analizar la capacidad de los modelos para reproducir adecuadamente la correlación de los índices de temperatura, se seleccionaron 8 estaciones representativas tomadas de diferentes zonas del territorio argentino. Para esto se buscaron estaciones que se diferencien significativamente según la distribución de sus observaciones. En este caso, para realizar el mencionado análisis, se utilizaron las series de residuos (datos a los que se les quitó el ciclo estacional).

En primer lugar, se verificó la presencia de dependencias a largo plazo en las series de temperatura media de cada estación. Para esto se utilizaron dos test diferentes: "Aggregated Variance" y "Differenced Variance" ([Caballero, 2001]). Una propiedad de las series que presentan memoria a largo plazo es que, si dividimos la serie en períodos de longitud m, la varianza de los valores medios de cada período, decrece lentamente a medida que aumentamos el m. Para verificar esta propiedad se dividieron las series en N/m bloques de longitud m y se computó la media para cada bloque:

$$x_k(m) = \frac{1}{m} \sum_{i=(k-1)m+1}^{km} X_i \quad k = 1, ..., N/m \tag{5.3}$$

y la varianza de las series obtenidas:

$$s^2(m) = \frac{1}{N/m - 1} \sum_{k=1}^{N/m} [x_k(m) - \overline{x}]^2 \tag{5.4}$$

donde, $\overline{x}$ es la media total.

Luego, si graficamos el logaritmo de $s^2(m)$ contra el logaritmo de m (a este tipo de gráfico usualmente se lo llama "log-log plot") podemos identificar una línea recta de pendiente $2d - 1$ con $d \in (0, 1/2)$. Este coeficiente d mide la "intensidad" de la memoria a largo plazo. Este método es conocido con el nombre de "Aggregated Variance" . En ciertas ocasiones, la heterogeneidad de los datos, puede producir valores positivos para d en series que en realidad no presentan una memoria a largo plazo. El método de "Differenced Variance", corrige este problema, concentrándose solamente en la diferencia entre la varianza de los bloques de longitud m y los de longitud $m + 1$ para cada valor de m.

$$\Delta s^2(m) = s^2(m) - s^2(m + 1). \tag{5.5}$$

Nuevamente, realizando un "log-log plot" de $\Delta s^2(m)$ contra m, podemos identificar una recta de pendiente $2d - 1$.

Para cada una de las 8 estaciones seleccionadas, se realizaron los 2 gráficos descriptos, para 50 valores de m logarítmicamente equidistantes con $30 < m < 5000$ (ver Figuras 5.16 y 5.17). Si observamos los gráficos correspondientes al primero de los tests ("Aggregated Variance"), en general pueden observarse valores medianamente bajos para d en todas las estaciones, salvo en Mar del Plata. En el segundo de los tests ("Differenced Variance"), las diferencias no son tan importantes, pero sigue siendo Mar del Plata la estación con mayor d. Esto indicaría, exceptuando a esta última estación, una falta de dependencias a largo plazo en las series.

Para verificar la capacidad de los modelos en reproducir la correlación entre la temperatura mínima y máxima (a este valor se lo denominará "cross-correlation"), se graficó el valor de este coeficiente para cada uno de los meses del año (ver Figura 5.18). Para esto se utilizaron simulaciones de 100 años. La simulación realizada por el modelo de Richardson, presenta una correlación entre la temperatura mínima y máxima mucho mayor a la de las observaciones. Esta es probablemente la diferencia más significativa entre los dos modelos.

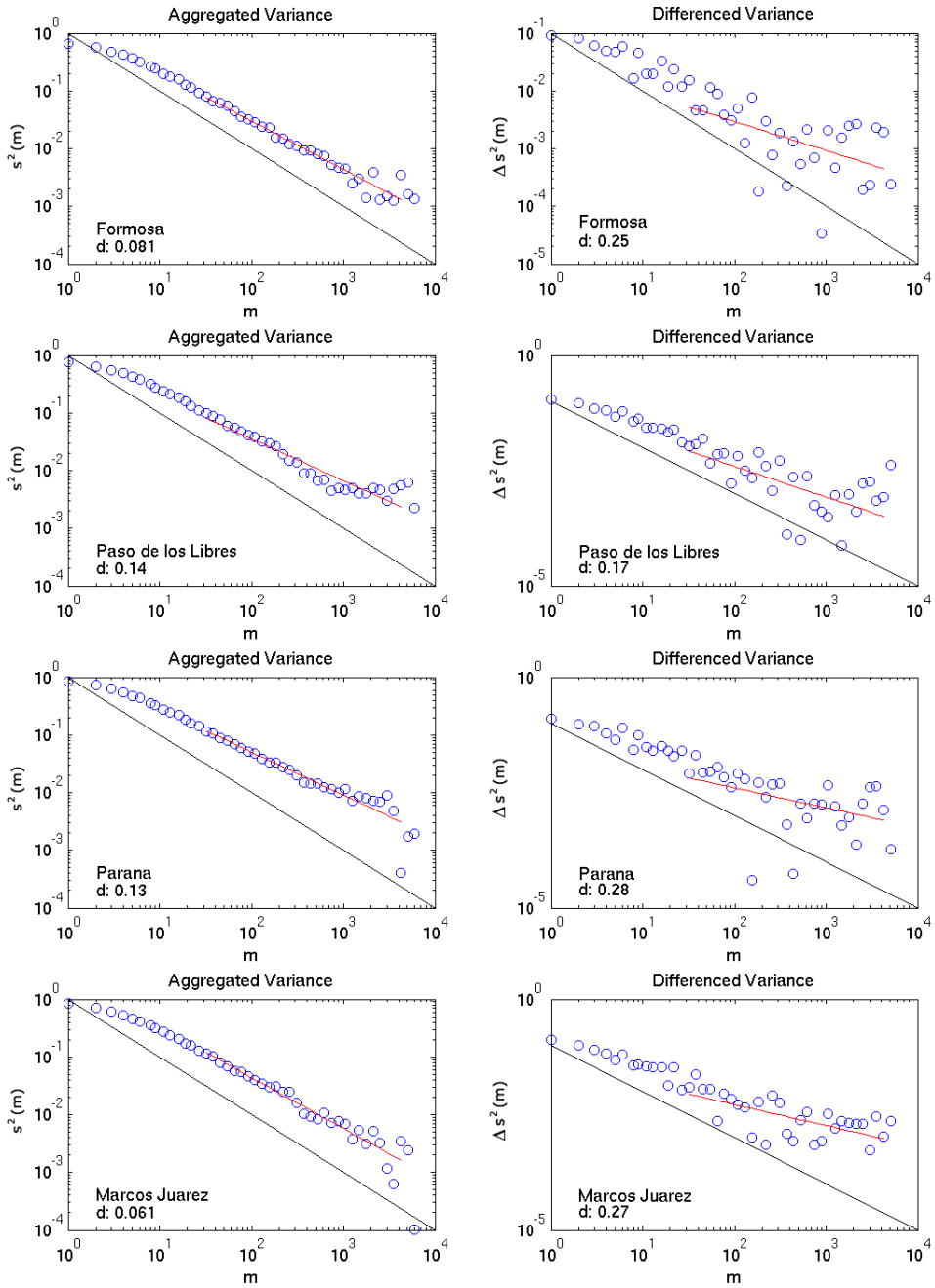

Figura 5.16: “Aggregated Variance” y “Differenced Variance” para 4 estaciones de Argentina. Para realizar este gráfico se tomaron las series de residuos (ver Sección 4.3).

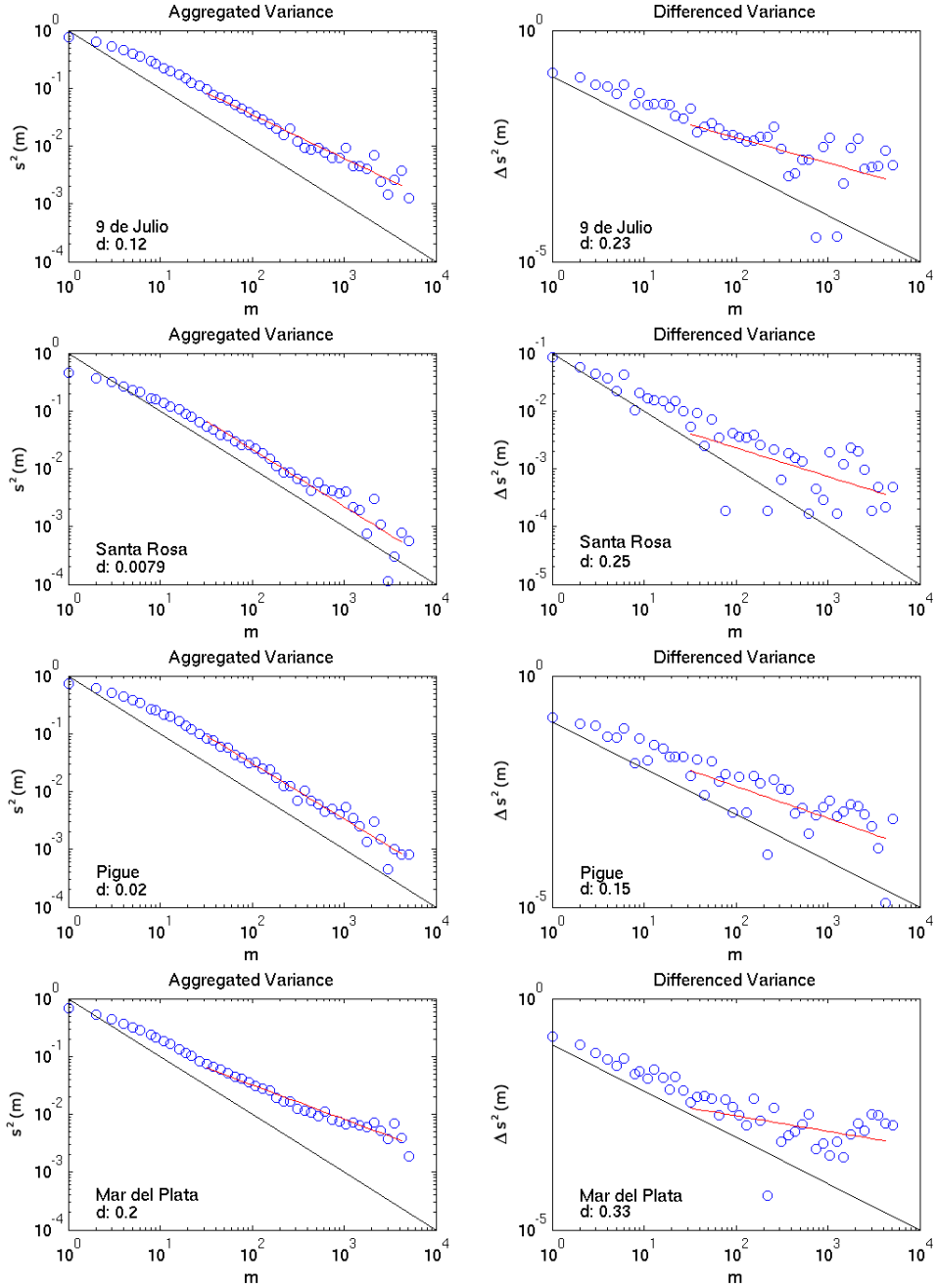

Figura 5.17: “Aggregated Variance” y “Differenced Variance” para 4 estaciones de Argentina. Para realizar este gráfico se tomaron las series de residuos (ver Sección 4.3).

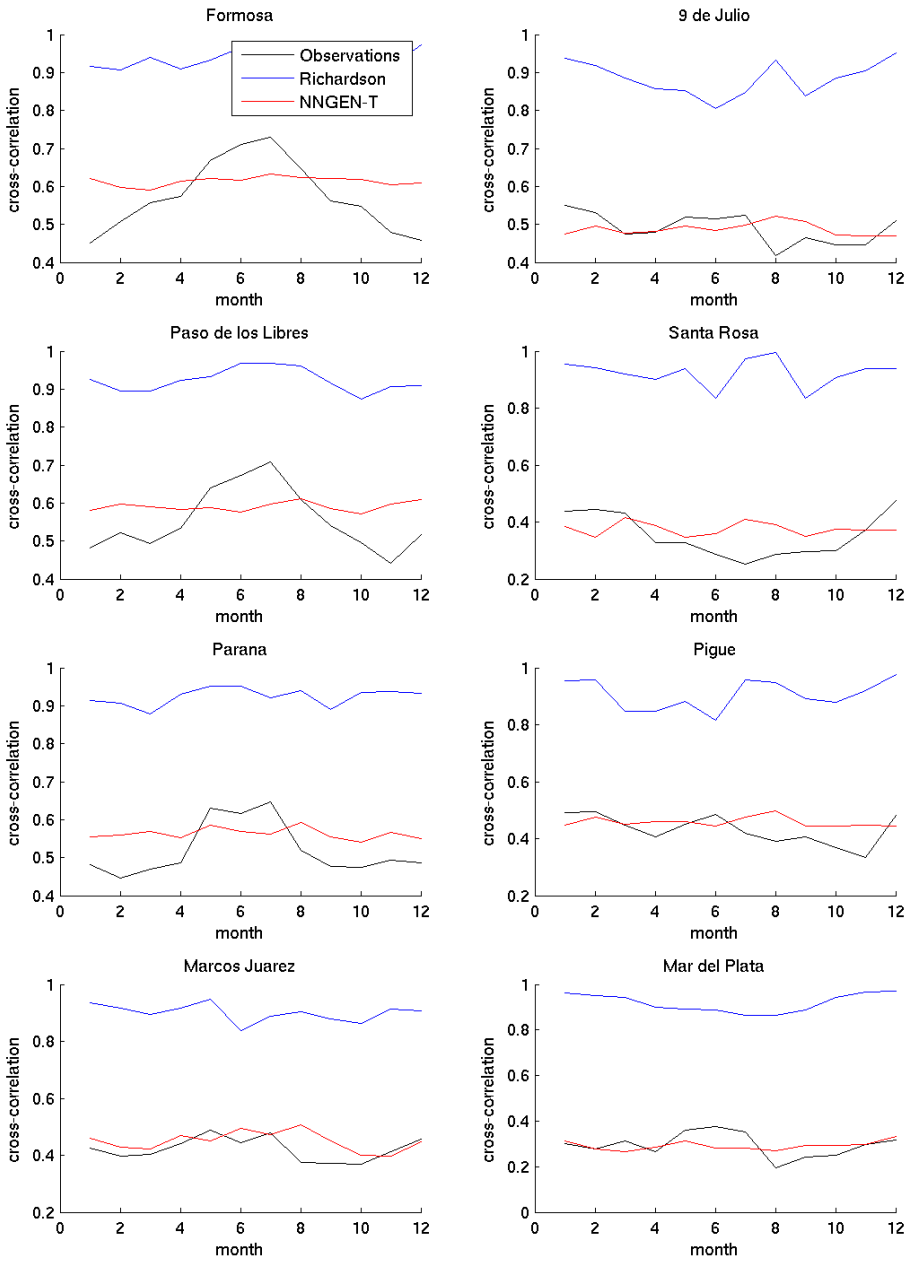

Figura 5.18: Correlación entre la temperatura máxima y mínima para cada mes, para 8 estaciones de Argentina.

Como se mencionó en la Sección 4.3 los modelos de temperatura generalmente fallan en reproducir la correlación temporal de los datos. Las Figuras 5.19 y 5.20 muestran la correlación observada para la temperatura mínima y la temperatura máxima en comparación con la correlación que presenta una simulación de 100 años realizada con cada uno de los modelos. Aquí puede verse como *NNGEN-T*, en general reproduce mejor la correlación temporal de las series para valores significativos.

Al analizar los resultados para algunas estaciones europeas que presentan una dependencia temporal mucho mayor en sus series (en la Figura 5.21 puede verse un ejemplo de 4 estaciones europeas con una fuerte dependencia a largo plazo en su temperatura media), en comparación con las estaciones de la Argentina, se observó que con la arquitectura propuesta para la Argentina el modelo no tenía la capacidad de reproducir la autocorrelación de los índices adecuadamente. Sin embargo, aumentando la dimensión de la capa escondida de la red a 30 neuronas y el “delay” a 7 (ver Sección 3.1), se mejoraron considerablemente los resultados del modelo de Richardson (ver Figura 5.22). De esta manera se pudo confirmar la flexibilidad del modelo para realizar simulaciones basadas en datos de diferentes regímenes climáticos.

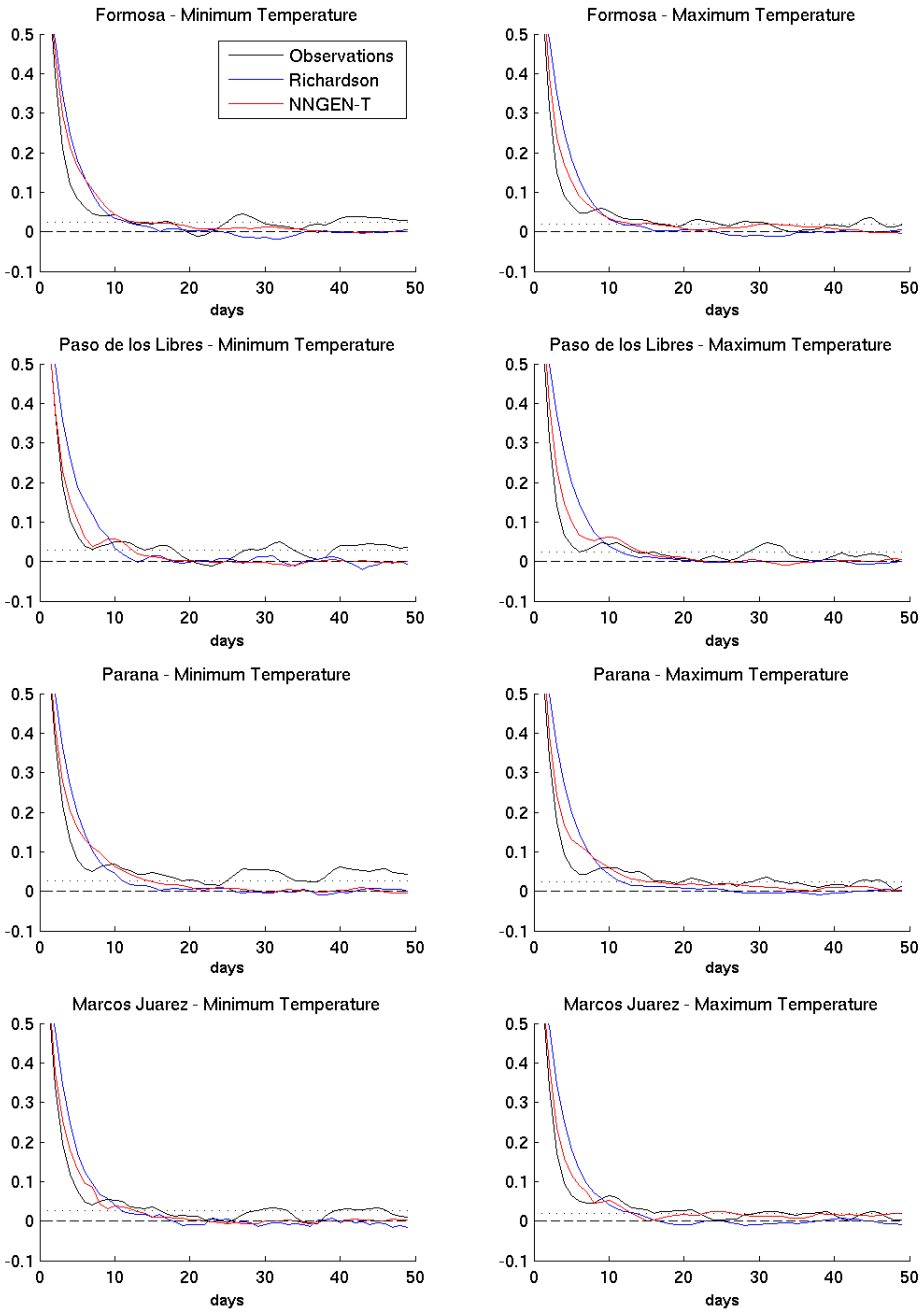

Figura 5.19: Correlación temporal de los índices de temperatura. La línea punteada indica el límite debajo del cual la correlación ya no es significativa.

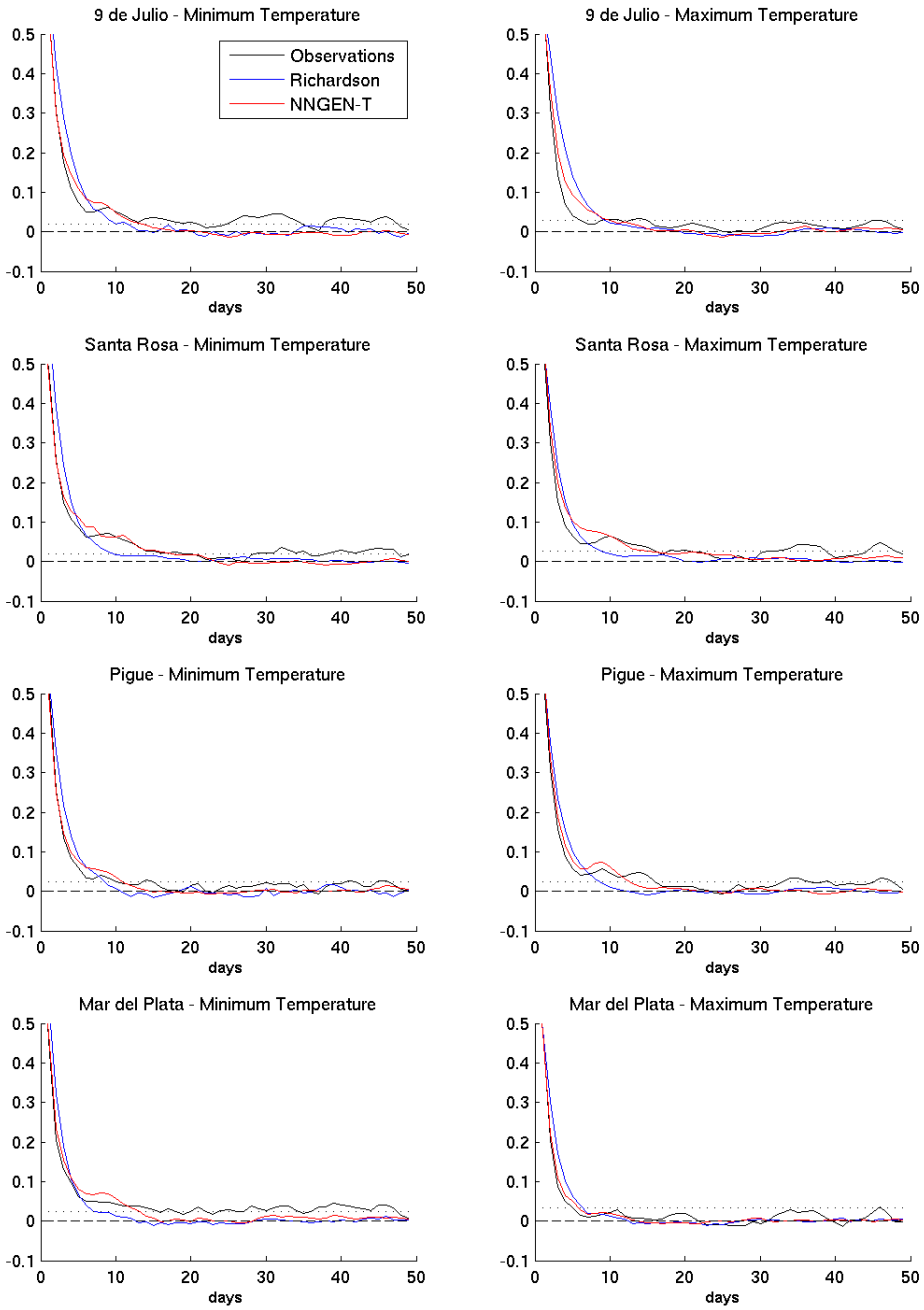

Figura 5.20: Correlación temporal de los índices de temperatura. La línea punteada indica el límite debajo del cual la correlación ya no es significativa.

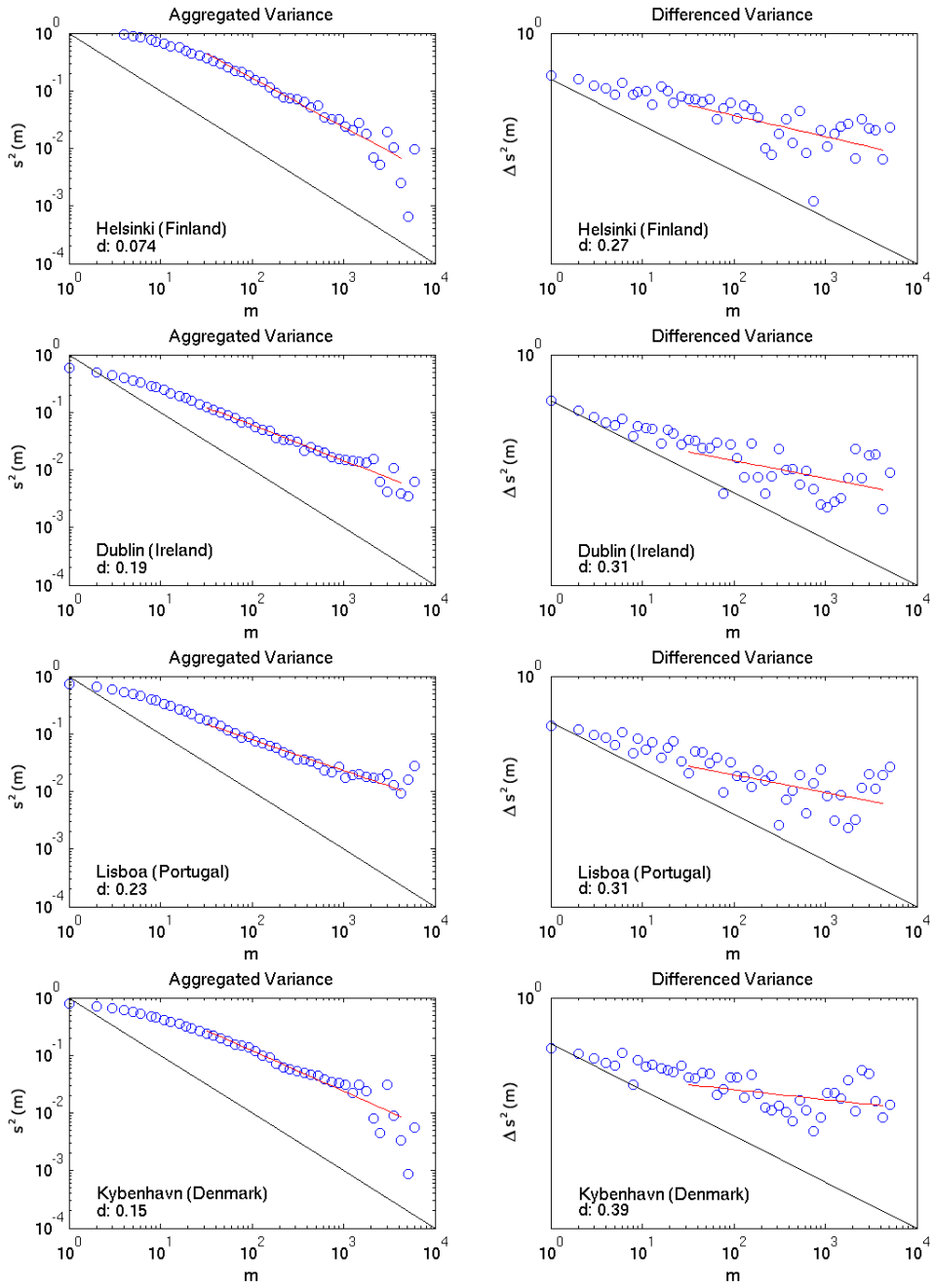

Figura 5.21: “Aggregated Variance” y “Differenced Variance” para 4 estaciones de Europa.

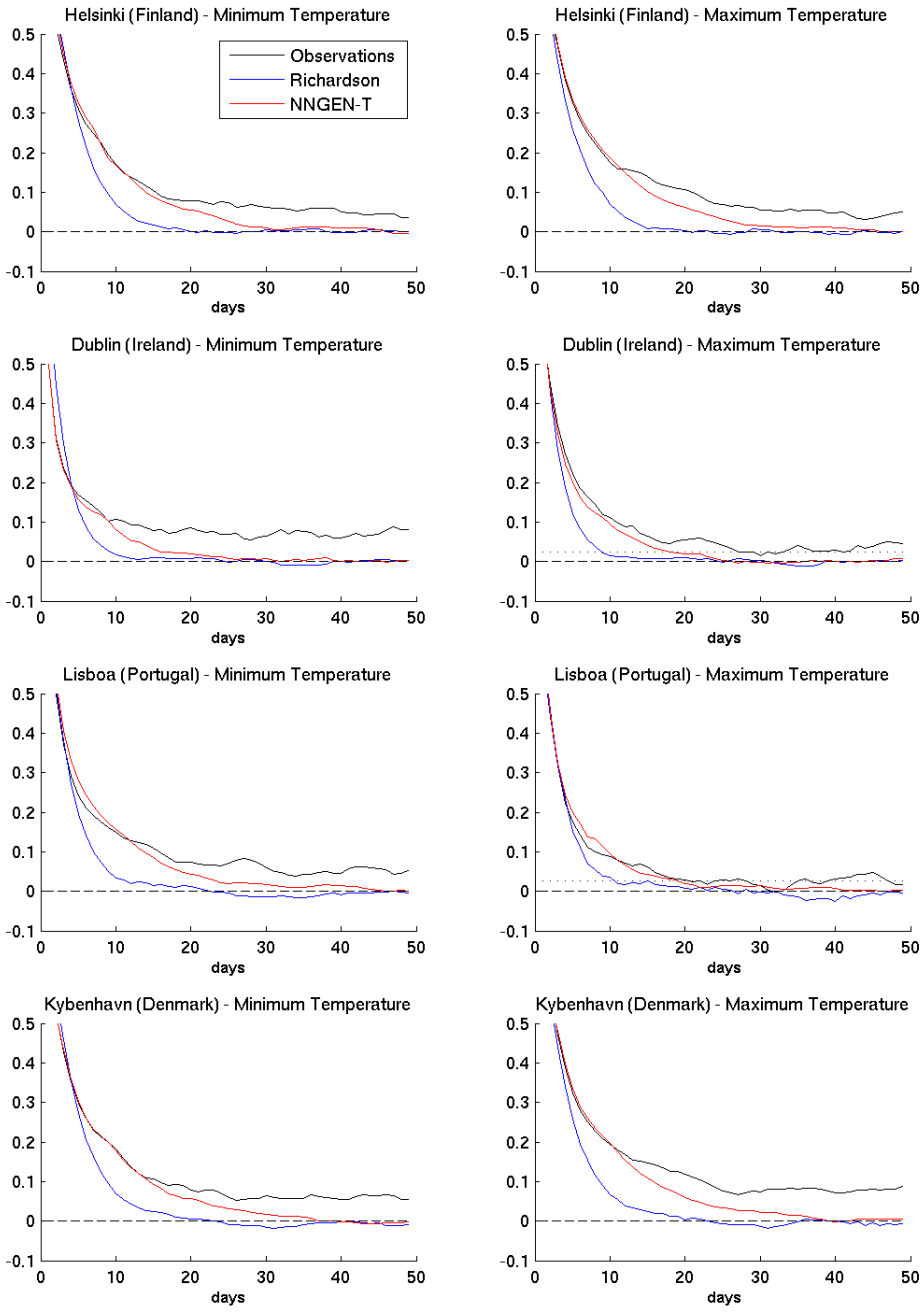

Figura 5.22: Correlación temporal de los índices de temperatura. La línea punteada indica el límite debajo del cual la correlación ya no es significativa.

Capítulo 6

Conclusiones

En este trabajo presentamos una nueva clase de generadores de tiempo basados en redes neuronales. En primer lugar, se desarrolló un generador de precipitaciones (al que llamamos *NNGEN-P*) basado en perceptrones multicapa. *NNGEN-P*, al igual que los modelos tradicionales de dos etapas, considera a la ocurrencia de días de lluvia y a la cantidad de precipitaciones para los días lluviosos en forma separada. Este modelo fue aplicado sobre distintas estaciones de la Argentina, Europa, EEUU y los Trópicos, mostrando una diferencia significativa con respecto a los modelos tradicionales, fundamentalmente en su capacidad para reproducir períodos largos de sequía y cantidades de precipitaciones extremas. Estas habilidades pueden ser de gran importancia cuando se utiliza a los generadores para forzar modelos de producción agrícola. Luego, se desarrolló un generador de temperaturas (al que llamamos *NNGEN-T*) basado en redes recurrentes simples. *NNGEN-T*, al igual que los modelos tradicionales, considera a la temperatura diaria como un proceso continuo, multivariado y débilmente estacionario. Este modelo fue aplicado sobre distintas estaciones de Argentina y Europa, mostrando su mayor habilidad en la capacidad de reproducir la correlación entre la temperatura mínima y la temperatura máxima. En cuanto a la reproducción de la correlación temporal de las series, el *NNGEN-T* se comportó mucho mejor que el modelo de Richardson para estaciones que presentan una correlación temporal importante en los datos históricos. Dada la flexibilidad del modelo, una selección más cuidadosa de la arquitectura, posiblemente permita lograr aún mejores resultados.

Luego de comprar al *NNGEN-P* y *NNGEN-T* con diferentes modelos, considerando que la bondad de cada modelo es relativa a las características de los datos observados, resulta evidente la importancia de tener un generador universal que pueda ser aplicado a diferentes tipos de clima. El papel que juegan las redes neuronales en el desarrollo de este tipo de generadores de tiempo es fundamental. En relación al generador de precipitaciones, los perceptrones multicapa nos permiten ajustar distribuciones sin la necesidad de determinar a priori la forma de la distribución, esto es importante debido a que distintos regímenes climáticos, usualmente presentan distintos tipos de distribuciones. En cuanto a la temperatura, donde la reproducción de la correlación temporal de la serie resulta ser uno de los aspectos más deseables, las redes recurrentes permiten simular estas series sin la necesidad de fijar la cantidad de días hacia atrás que se tendrán en cuenta o la dependencia con respecto a los períodos secos y lluviosos. De esta forma, utilizando redes neuronales, no es necesario caracterizar los procesos detalladamente en términos estadísticos, lo cual va en contra de la esencia misma de un generador universal, dado que los datos de distintos regímenes climáticos eventualmente mostrarán distintas características.

Unos de los principales problemas que encontramos en la utilización de redes neuronales fue el tiempo de cómputo necesario para obtener un generador, para una estación dada. Este tiempo fue particularmente grande para los generadores de temperatura. Sin embargo, somos optimistas, ya que existe una gran variedad de tipos de redes recurrentes que podrían ser utilizadas con los mismos fines. Además, siempre es posible mejorar la implementación o bien explorar distintas formas de entrenar a las redes.

Bibliografía

Bodén M. (2001) A guide to recurrent neural networks and backpropagation.

Boulanger J., Martínez F. L. and Segura E. C. Projection of future climate change conditions using IPCC simulations, neural networks and bayesian statistics. Part 1: Temperature mean state and seasonal cycle in South America. (2005) *Climate Dynamics.* Aceptado.

Boulanger J., Martínez F. L. and Segura E. C. Projection of future climate change conditions using IPCC simulations, neural networks and bayesian statistics. Part 2: Precipitation mean state and seasonal cycle in South America. (2005) *Climate Dynamics.* Aceptado.

Boulanger J., Martínez F. L., Penalba O. and Segura E. C. The Neural Network Weather Generator (NNGEN). Improving stochastic simulation of daily precipitation with NNGEN-P. (2005) *Climate Dynamics.* Aceptado.

Buishand, T. A. (1978) Some remarks on the use of daily rainfall models. *J. Hydrology.* 36, 295-308.

Caballero, R., S. Jewson and A. Brix (2001) Long mempry in surface air temperature: Detection, modeling and application to weather derivative calculation. *Climate Research.*

Castellví F, Mormeneo I. and Perez P. J. (2004) Generation of daily amounts of precipitation from standard climatic data: a case study for Argentina. *J. Hydrology.* 289, 286-302.

Jeffrey L. Elman (1990) Finding Structure in Time. *Cognitive Science.* 14, 179-211.

Funahashi K. (1989) On the approximate realization of continuous mapping by neural networks. *Neural Networks* 2, 183-192.

Geng, S., Penning De Vries F.W.T. and Supit, I. (1986) A simple method for generating daily rainfall data. *Agric. Forest Meteorol.* 36, 363-376.

Hayhoe H. N. (2000) Improvements of stochastic weather data generators for diverse climates. *Climate Research.* 14, 75-87.

Hertz J., Krogh A. and Palmer R. (1991) Introduction to the Theory of Neural Computation. *Addison-Wesley.*

Katz, R. W. (1977) Precipitation as a chain-dependent process *J. Applied Meteor.* 16, 671-676.

Matalas, N. S. (1967) Mathematical assessment of synthetic hydrology. *Water Resourc. Res.* 3(4), 937-45.

Mitchell P. D. (2000) Generation of Simulated Daily Precipitation and Air and Soil Temperatures. *Working Paper 00-WP 234*

Nabney, I. T. (2002) NETLAB, Algorithms for Pattern Recognition. *Springer Verlag, pp. 420.*

Racsko, P., Szeidl, L. and Semenov, M. (1991) A serial approach to local statistics weather models. *Ecol. Modeling.* 57, 27-41.

Richardson, C. W. (1981) Stochastic simulation of daily precipitation, temperature, and solar radiation. *Water Resources Res.* 17, 182-190.

Selker J. S. and Haith D. A. (1990) Development and testing of single-parameter precipitation distributions. *Water Resour. Res.*

Srikanthan, R. and McMahon, T.A. (2001) Stochastic generation of annual, monthly and daily climate data: A review. *Hydrology and Earth System Sciences.* 5, 653-670.

Wilks, D. S. (1989) Wilks, D. S., Conditioning stochastic daily precipitation models on total monthly precipitation. *Water Resources Res.* 25, 1429-1439.

Wilks, D. S. (1992) Adapting stochastic weather generation algorithms for climate change studies. *Climatic Change* 22, 67-84.

Wilks, D. S. (1998) Multisite generalization of a daily stochastic precipitation generation model. *J. Hydrology.* 210, 178-191.

Wilks, D. S. (1999) Simultaneous stochastic simulation of daily precipitation, temperature and solar radiation at multiple sites in complex terrain. *Agricultural and Forest Meteorology.* 96, 85-101.

Wilks, D. S. (1999) Interannual ariability and extreme-value characteristics of several stochastic daily precipitations models. *Agricultural and Forest Meteorology.* 93, 153-169.