To appear in *Connection Science - 1996*

Combining Neural Network Forecasts on Wavelet-Transformed Time Series

Alex Aussem† and Fionn Murtagh‡

†Université René Descartes, UFR de Mathématiques et Informatique,
45, rue des Saints-Pères, 75006 Paris - France
Email: alex@math-info.univ-paris5.fr

‡Space Telescope - European Coordinating Facility,
(Affiliated to Astrophysics Div., Space Science Dep., European Space Agency)
European Southern Observatory, Karl-Schwarzschild-Str. 2,
D-85748 Garching bei München - Germany
Email: fmurtagh@eso.org

Abstract

We discuss a simple strategy aimed at improving neural network prediction accuracy, based on the combination of predictions at varying resolution levels of the domain under investigation (here: time series). First, a wavelet transform is used to decompose the time series into varying scales of temporal resolution. The latter provide a sensible decomposition of the data so that the underlying temporal structures of the original time series become more tractable. Then, a Dynamical Recurrent Neural Network (DRNN) is trained on each resolution scale with the temporal-recurrent backpropagation (TRBP) algorithm. By virtue of its internal dynamic, this general class of dynamic connectionist network approximates the underlying law governing each resolution level by a system of nonlinear difference equations. The individual wavelet scale forecasts are afterwards recombined to form the current estimate. The predictive ability of this strategy is assessed with the sunspot series.

Keywords – Dynamical Recurrent Neural Networks, time series prediction, wavelet transform.

1 Introduction

Recently a general framework has been proposed for discrete-time neural networks, endowed with a temporal dimension, through embedding finite impulse response (FIR) filters in a fully recurrent neural structure. These models, called Dynamical Recurrent Neural Networks (DRNN), have an advantage over static recurrent networks in much the same way that feedforward FIR networks [22] have an advantage other static feedforward networks.

The versatility of the DRNN model is not to the detriment of ease of training. Indeed it has been demonstrated that backward error propagation of the unfolded adjoint decays at exponential speed [3]. Therefore it suffices to back-propagate the gradient a limited number of times to approximate the true gradient as closely as desired, thus limiting drastically the computational burden.

In this paper, we show how use of several of these dynamical connectionist models can aid in capturing useful information on the dynamic of complex time series. Our forecasting strategy is based on the subdivision of the prediction task into elementary tasks. For this purpose, we use a particular wavelet transform aimed at laying bear useful information, which is then treated by the neural network. Like the Fourier transform, the wavelet transform has proven to be a versatile tool which gives a better handle on the data. It has been used effectively for image compression, noise removal, object detection, and large-scale structure analysis, among other applications [7, 10, 15, 19, 11, 20, 5].

The "à trous" wavelet transform is a redundant transform (i.e. decimation is not carried out). It decomposes the input signal into detail signals, and a residual. The original signal can be expressed as an additive combination of the wavelet coefficients, at the different resolution levels. The latter provide a sensible decomposition of the signal, or time series, so that faint temporal structures may be revealed. They may also be processed by appropriate statistical models, while they would have remained obscured by the noise and complexity of the original series. Therefore it suffices to run a DRNN model on each resolution level and then recombine the individual predictions to form the final forecast.

This approach is successfully applied to a classical time series for which the many previous analyses provide comparative results.

2 The DRNN Model

A network that is to handle difficult sequences must be able to store and update context information spanning possibly unlimited temporal intervals. And for the network to remember some aspects of the past, one has to build into the network as much knowledge of the past as possible. It should hold some memory of its internal state also. This will be achieved by replacing each synaptic weight by a finite impulse response (FIR) filter. In contrast to feedforward FIR networks, an input excitation of finite duration may generate an output of *infinite* duration. The model is referred to as a dynamic recurrent neural network (DRNN) since the underlying law governing its evolution is a system of difference equations.

Consider a fully connected network of N continuous-valued units. Let the matrix element w_{ij}^d denote the connection strength, or coupling, from the d-th delay link connecting node i to node j, and D the maximum connection delay, i.e. the filter order. The response of a unit is called its activation value (also called unit output). The vector $\mathbf{v}_k$ ($\mathbf{v}_k \in R^{[N\times 1]}$) is the activity of the units at time k. Some of these units feeding input patterns into the rest of the network have their activation clamped to $\mathbf{i}_k$ ($\mathbf{i}_k \in R^{[N\times 1]}$). We do not allow for connections towards input units, i.e. $w_{ij}^d = 0$ if j is an input unit, and for clamped units, we set $i_k(j) = 0$ if unit j is not an input unit. Similarly, some units are termed output units with target values $\mathbf{d}_k$ ($\mathbf{d}_k \in R^{[N\times 1]}$). Units having neither input nor output connections to the outside world are called *hidden* units. In addition, a *bias* input to each unit is achieved by fixing a particular unit activation to 1.0. For notational convenience, we won't distinguish between input, hidden, output and bias units in the following formulae.

Continuing with notation, the weights for the d-th delay connections heading to unit j at iteration k are specified by the vector $\mathbf{w}_k^d(j) = [w_k^d(1,j), \ldots, w_k^d(N,j)]^T$. Similarly, the $N \times N$ matrix $\mathbf{W}_k^d = [\mathbf{w}_k^d(1), \mathbf{w}_k^d(2), \ldots, \mathbf{w}_k^d(N)]$ contains all adaptive weights of delay $d \le D$ at iteration k. The initial vector $\mathbf{W}_0^d$ is usually chosen randomly. We determine the relevant system parameters at each step k by the relaxation of the following system of coupled differential equations:

$$
\begin{aligned}
\mathbf{T}\frac{d\mathbf{v}_k(t)}{dt} &= -\mathbf{v}_k(t) + g(\mathbf{h}_k(t)) + \mathbf{i}_k \\
\mathbf{h}_k(t) &= \sum_{d=1}^{D} {\mathbf{W}_k^d}^T \mathbf{v}_{k-d} + {\mathbf{W}_k^0}^T \mathbf{v}_k(t)
\end{aligned}
\tag{1}
$$

$\mathbf{T}$ is a $N \times N$ diagonal matrix with conveniently chosen positive time constants, $\mathbf{h}_k$ is the unit internal input vector at iteration k, and

$$\begin{aligned} g(\mathbf{h}_k) &= [g(h_k^1), \ldots, g(h_k^n)]^T \\ g(h) &= \frac{1}{1+e^{-\beta h}} \end{aligned} \tag{2}$$

The composition of an affine mapping followed by a sigmoidal function is a common choice for the nonlinear activation function. The parameter β controls the steepness of the logistic function and is arbitrarily set to $1/2$.

It is not guaranteed that Eqn. 1 always settles down to a stable equilibrium solution; existence of at least one stable fixed-point attractor is assumed. Therefore, if such a solution exists for system (1), it is readily seen that it verifies

$$\mathbf{v}_k = g(\sum_{d=0}^{D} \mathbf{W}_k^{d^T} \mathbf{v}_{k-d}) + \mathbf{i}_k \tag{3}$$

By virtue of the *time-* and *space*-recurrency, the DRNN behavior is described by a system of nonlinear difference equations, whose internal variables are the hidden unit activations. The main attraction of these networks is that they are endowed with a trainable internal memory and hence can directly implement dynamical systems. Furthermore the use of a time-lagged vector is not required anymore.

The DRNN may be represented by two NARMA (nonlinear autoregressive moving average) sub-networks $\mathcal{N}_1$ and $\mathcal{N}_2$ in series

$$\begin{aligned} \mathbf{v}_k &= \mathcal{N}_1(\mathbf{v}_{k-1}, \ldots, \mathbf{v}_{k-D}; x_k, \ldots, x_{k-D}) \\ \hat{x}_{k+1} &= \mathcal{N}_2(\mathbf{v}_k) \end{aligned} \tag{4}$$

where $\mathcal{N}_2$ is fed by the output of $\mathcal{N}_1$, which in turn can be decomposed into two sub-networks, and so on. Now define the ND-dimensional vector made up of all the activations at times $t, t-1, \ldots, t-D+1$,

$$\mathbf{z}_k = [\mathbf{v}_k, \ldots, \mathbf{v}_{k-D+1}; x_k, \ldots, x_{k-D+1}] \tag{5}$$

Eqn. (4) turns out to have a canonical form

$$\begin{aligned} \mathbf{z}_k &= \mathcal{N}_1(\mathbf{z}_{k-1}, x_k) \\ \tilde{\hat{x}}_{k+1} &= \mathcal{N}_2(\mathbf{z}_k) \end{aligned} \tag{6}$$

where higher order delays have been removed. In view of (6), such a model should have an advantage over linear Kalman estimators in much the same way that NARMA models have an advantage over ARMA models [6].

3 DRNN Training

The introduction of time delays leads to a somewhat more complicated training procedure because the current state of a unit may depend on several events occuring in the neighboring neurons at different times. Consequently, the gradients with respect to past and present parameters have to be determined for proper parameter adaptation.

In the classical manner, our goal is to find a procedure to adjust the weight components w_{ij}^d so that a given fixed initial set of conditions $\mathbf{v}_0$ and a set of input vectors $\{\mathbf{i}_k\}$ taken from the training set result in a set of fixed-points, whose components along the output units have a desired set of values $\{\mathbf{d}_k\}$. This will be accomplished by our minimizing a quadratic function

$$E = \frac{1}{2} \sum_{k=0}^{k_f} \mathbf{e}_k^T \mathbf{e}_k \tag{7}$$

which measures the errors $\mathbf{e}_k = \mathbf{d}_k - \mathbf{v}_k$ between the desired and the actual fixed-point values along a trajectory or epoch starting at iteration 0 up to k_f. Classically, a new weight is generated at the end of the trajectory by

$$\Delta w = -\eta \frac{\partial E}{\partial w} \tag{8}$$

where η is the learning rate. Our primary goal is to find an expression for the term $\partial E / \partial w$.
In order to clarify our notation, let $\mathbf{G}_k$ denote the $N \times N$ diagonal matrix constructed from $g(u_k(j))$ for $j = 1, \ldots, N$ and $\mathbf{G'}_k = 2\beta \mathbf{G}_k (1 - \mathbf{G}_k)$. (Recall that $g'(h) = 2\beta g(h)(1 - g(h))$.)

Proposition 1 *(On learning fixed-points). If the DRNN model is convergent over the trajectory $0 \leq k \leq k_f$, then the gradient of cost can be computed at time step k_f by*

$$\frac{\partial E}{\partial \mathbf{W}^d} = \sum_{k=d}^{k_f} \boldsymbol{\delta}_k \mathbf{v}_{k-d}^T \quad \forall d \leq D \tag{9}$$

where the vector $\boldsymbol{\delta}_k$ is given by $\boldsymbol{\delta}_k = \mathbf{G'}_k \mathbf{y}_k$ and $\mathbf{y}_k$ is the solution of following backward difference equation

$$\mathbf{y}_k = [\mathbf{I} - \mathbf{W}_k^0 \mathbf{G'}_k]^{-1} [\mathbf{e}_k + \sum_{j=1}^{D} \mathbf{W}_{k+j}^j \mathbf{G'}_{k+j} \mathbf{y}_{k+j}] \tag{10}$$

with the boundary conditions $\mathbf{y}_{k_f} = [\mathbf{I} - \mathbf{G'}_k \mathbf{W}_k^0]^{-1} \mathbf{e}_{k_f}$ and $\mathbf{y}_{k+j} = \mathbf{0}$ for $j > 0$.

The demonstration of this proposition has been given elsewhere [1, 2] and relies – in the classical manner – on the repeated application of a particular chain rule expansion of partial derivatives in a set of ordered equations [13, 24].
This generalized ∂ rule is essentially a *backsweep* or *backward* recurrent propagation procedure in that it puts recurrent-backpropagation [14] and backpropagation-through-time [17] into a single framework. Its implementation makes necessary the artifact of unfolding the adjoint network in a reverse way in time. This is achieved by replicating the units and the connections backward through time until all delays have been removed. The method produces a *time*-feedforward and *space*-recurrent constrained static structure where the time dependencies have been made external to the network itself.
At the end of the trajectory, the terms $\mathbf{y}_k, k < k_f$ are backpropagated through the adjoint network up to time 0. Each gradient term is recombined afterwards to store the overall weight update due to the error $\mathbf{e}_n$.
For the unfolded version to be strictly equivalent to the original network, the manyfold duplication of the adjoint model when $k_f \rightarrow \infty$ should be repeated *ad infinitum*. Because the storage and memory requirements grow proportionally with the pattern index, the procedure becomes quickly inhibitory from an implementation standpoint when long trajectories are to be considered.

In order to prevent the systematic expansion of the error-propagation network and to ensure the stability of the parameter adaptation procedure, the fundamental result is used.

Theorem 1 *(Gradient Exponentail Decay). Let the instantaneous scalar cost be* $E_n = \frac{1}{2}\mathbf{e}_n^T\mathbf{e}_n$ *and consider the error vector* ${\mathbf{y}_k^{k+n}}^T = \frac{\partial E_{k+n}}{\partial \mathbf{v}_k}[\mathbf{I} - \mathbf{G'}_k {\mathbf{W}_k^0}^T]^{-1}$ *and let* $k_f > 0$ *denote the trajectory length of the system.*
If *the following conditions hold* $\forall d = 1, \ldots, D$, $\forall k = 0, \ldots, k_f$

$$\|\mathbf{W}_k^0\| < 1 \quad \text{and} \quad \frac{\|\mathbf{W}_k^d\|}{1 - \|\mathbf{W}_k^0\|} < \frac{1}{D} \tag{11}$$

then $\exists 0 < \rho < 1$ *and* $\exists M > 0$ *such that the inequality*

$$\|\mathbf{y}_{k_f-n}^{k_f}\| < M\rho^n, \tag{12}$$

holds for $n = 0, 1, \ldots, k_f$.

The demonstration of this technical result can be found in [3]. The theorem says, in essence, that the terms $\mathbf{y}_{k_f-n}^{k_f}$ are bounded by a series decaying at exponential rate with respect to n whenever conditions (11) are satisfied. In practice, however, much weaker bounds on the weights seem to exist as indicated by empirical studies of the dynamics of networks with random weights.

Proposition 2 *Consider the samplewise backward recurrent propagation algorithm. If the conditions* (11) *are fulfilled, then for the weight changes computed at each iteration to follow as closely as desired the true gradient, it is sufficient to backpropagate a finite number of time steps through the adjoint.*

The proof of this proposition is a direct consequence of the theorem. Interestingly, the cost due to the manyfold duplication of the hardware can be maintained within acceptable limits, thanks to the exponential decay of the gradient through the adjoint network.
We propose an implementation-friendly algorithm that preserves the symmetry between the forward propagation of unit outputs and the backward propagation of error terms at each time increment k:

- Relax the DRNN once until a fixed-point solution is obtained, using the current input vector $\mathbf{i}_k$ and all past variables up to $k - D$ required by

$$\mathbf{v}_k = f(\mathbf{i}_k, \ldots, \mathbf{i}_{k-D}, \mathbf{v}_k, \mathbf{v}_{k-1}, \ldots \mathbf{v}_{k-D}, \boldsymbol{\mathcal{W}}_k) \tag{13}$$

- Calculate an approximation for $\mathbf{y}_{k-n}^k$, denoted by $\hat{\mathbf{y}}_{k-n}^k$, for $n > 0$ by unfolding the adjoint network backward in time and relaxing

$$\mathbf{T}\frac{d\hat{\mathbf{y}}_{k-n}^k(t)}{dt} = [\mathbf{W}_k^0\mathbf{G'}_{k-n} - \mathbf{I}]\hat{\mathbf{y}}_{k-n}^k(t) + \sum_{n=1}^{D}\mathbf{W}_k^n\mathbf{G'}_{k-n+j}\hat{\mathbf{y}}_{k-n+j}^k \tag{14}$$

  The boundary conditions of (14) take the form

$$\hat{\mathbf{y}}_k^k = \mathbf{y}_k^k = [\mathbf{I} - \mathbf{W}_k^0\mathbf{G'}_k]^{-1}\mathbf{e}_k \tag{15}$$

and

$$\hat{\mathbf{y}}_{k+1}^{k}, \ldots, \hat{\mathbf{y}}_{k+D}^{k} = \mathbf{0} \tag{16}$$

Notice that $\mathbf{y}_{k-n}^{k}$ cannot be calculated precisely because of the weight changes at each iteration and the weight matrix $\mathbf{W}_{k-n+j}^{j}$ is approximated by $\mathbf{W}_{k}^{j}$, which is the current weight matrix. Obviously, the smaller the difference between the weights at iteration k and those at iteration $k-n+j$ the more accurate this approximation will be and thus the more accurate the calculation of the terms $\hat{\mathbf{y}}_{k-n}$.

Locally in the adjoint, the connections are reversed, that is, the coupling w_{ij}^{d} from i to j is replaced by $g'(h_j)w_{ji}^{d}$ from j to i. The network errors are now taken as inputs and the cells now perform a linear transformation of their input.

- Stop the backward recursion of (14) at iteration $k-s$ satisfying

$$\frac{||\mathbf{y}_{k-s+j}^{k}||}{||\mathbf{y}_{k}^{k}||} \leq \epsilon \quad \text{for} \quad j = 1, \ldots, D \tag{17}$$

  The terms $\mathbf{y}_{k}^{k-j}$ are stored at each backward iteration until the iterative process terminates.

- The overall weight variation – defined as the sum of individual updates – is then obtained by applying the generalized delta-rule

$$\Delta \mathbf{W}^{d} = \eta \sum_{n=d}^{k} \boldsymbol{\delta}_{n}^{k} \mathbf{v}_{n-d}{}^{T} \tag{18}$$

- Store for the next gradient propagation the terms $g(\mathbf{h}_{k-j})$ and $\mathbf{v}_{k-j}$ for $j = 0, 1, \ldots, L$ where L is taken large enough to satisfy $s < L$ at any time. In practice, L is first set to a multiple of D and incremented each time $s > L$ until a correct value is found.

This leads to an efficient parallel implementation requiring a limited number of relaxations, say s, of the transposed error network. Since s is of order D (see [3]), storage of order DN is needed to keep track of the previous network states. The time required at each backward pass to compute the elementary weight update scales as $O(DN^2)$ so that $O(D^2N^2)$ operations are needed at each time step to adjust the network parameters.

It is also worth remarking that the relaxation phase boils down to a single iteration of the network forward in time when no *space*-recurrent links are used. This is the case for synchronous dynamics (i.e. $w_{ij}^{d} \neq 0$ for $d > 0$ only) including those proposed in [22, 25].

4 Wavelet Decomposition

A wavelet transform provides a decomposition in terms of time and frequency, or of scale and position. A wavelet transform for discrete data is provided by the particular version known as the *à trous* (with holes) algorithm [9][18]. This is a "stationary" [12] or redundant transform, i.e. decimation is not carried out.

Take $c_0(t) = x(t)$, the input data. We compute the wavelet coefficients using the following convolution with the discrete low-pass filter h,

$$c_{i+1}(k) = \sum_{l=-\infty}^{+\infty} h(l) c_i(k + 2^i l) \tag{19}$$

The increase in distances between the sampled points (i.e. $2^i l$) explains why the name "à trous" has been applied to this method.
After this sequence of smoothings, the discrete wavelet transform is computed by taking the signal difference w_i between two consecutive resolutions:

$$w_i(k) = c_{i-1}(k) - c_i(k) \tag{20}$$

The set $\mathcal{W} = \{w_0, w_1, \ldots, w_p, c_p\}$ represents the wavelet tranform of the data up to resolution level p. The term c_p is the residual.
Handling of boundaries may be carried out in different ways. We took the intuitive approach by taking (for a time series of size N) $c(N + k) = c(N - k)$. Clearly, in prediction studies, very careful attention must be given to the (upper, time step t) boundary, even if this can be partially mitigated by a long-memory prediction method. We hypothesize that handling the boundaries by mirroring, as here, emphasizes the particularity of the extreme sampled values available to us for prediction, and therefore is an appropriate way to handle the boundaries.
The wavelet expansion of the time series, in terms of wavelet coefficients, is given by

$$x(t) = c_p(t) + \sum_{i=1}^{p} w_i(t) \tag{21}$$

This equation provides a reconstruction formula for the original time series. It is additive, which leads us to fuse predictions also in an additive manner. We will see shortly, however, that we can adopt a hybrid strategy in regard to exactly what is combined to yield an overall prediction. I.e. we can test a number of short-memory and long-memory predictions at each resolution level, and retain the method which performs best.
The low-pass filter, h, which we used was a B_3 spline, defined as $(\frac{1}{16}, \frac{1}{4}, \frac{3}{8}, \frac{1}{4}, \frac{1}{16})$. This is of compact support (necessary for a wavelet transform), and is point-symmetric. The latter does not allow for the fact that time is a fundamentally asymmetric variable. However, in the absence of a clear choice for what sort of asymmetry would be most appropriate, we feel that the symmetric smoothing function used was a good choice.
For a time series of size N, the wavelet decomposition used here can be determined in $O(N)$ time. It therefore has more favorable computational cost than the FFT.
Armed with (i) a powerful modeling and forecasting method, and (ii) a versatile and tractable data decomposition scheme, we now turn attention to the combination of both. We consider the following scheme. At time t, we have the observations $x(t), x(t-1), \ldots, x(1)$ and we are seeking an accurate estimate of $x(t+1)$. The idea is to carry out a wavelet transform on $x(t), x(t-1), \ldots, x(1)$, with a user-specified number of levels, say 5, plus the residual. Hence we obtain an array of wavelet coefficients $\{W(i,j)\}$ for $i = t, t-1, \ldots, 1$ and $j = 1, 2, \ldots, 6$. This is an additive decomposition of the input signal, so we have $x(i) = \sum_j W(i,j) \quad \forall i = 1, 2, \ldots, t$. We carry out predictions independently on each $W(i,j) \ \ \forall j$. If the resolution scales are physically interpretable (as they may well be) then the independent predictions are similarly open to interpretation. Training of the prediction engines is carried out in the classical manner, i.e. using a training subset of the data.

5 Application

We are concerned with assessing the predictive power of this wavelet-connectionist strategy in a concrete scenario. The sunspot series was the first time series studied with autoregressive models [16, 21, 26], and thus has served as a benchmark in the forecasting literature. The sunspots are dark blotches on the sun that can be related to other solar activities such as the magnetic field cycles, which in turn influence, by indirect and intangible means, the meteorological conditions on earth. The solar cycle with which sunspots occur varies over a 7 to 15 year period. Although the data exhibit strong regularities, attempts to understand the underlying features of the series have failed because the amount of available data was insufficient. Thus the sunspots provide an interesting data-set to test our wavelet decomposition method.

Consistent with previous appraisals [21, 23], we use range-normalized yearly averages of the sunspot data tabulated from 1720 to 1979. Since quantitative performance measures are expressed in terms of one-step ahead errors in the literature, the latter is used as a performance criterion. The single-step prediction error is monitored on the 59 withheld sunspot values ranging from 1921 to 1979, while the remaining data is used for training. The performance measure is expressed in terms of one-step ahead errors.

5.1 DRNN Architecture Used

A natural manner to assess the performance of this subdivision strategy is to take a given network architecture and compare the mean squared error obtained by the direct prediction with the error obtained by the decomposition method.

With this in mind, it is wise to select an efficient architecture which balances pure performance against complexity, given some constraints on the topology, the number of neurons and synaptic connections. For the sake of simplicity, we restrict ourselves to a network having 1 input unit, 2 hidden units fully connected with FIR filters of order 2, and one output unit. Both input and output units are connected with static links so that the memory of the system is purposely confined to the core of the model. In particular, a single sunspot number is presented to the input and the network is trained to output the next value, thus forcing the DRNN to encode some past information on the sequence. A time lagged vector is not required anymore thanks to the internal dynamics of the model.

The network was trained up to complete convergence. This tiny architecture yielded quite acceptable performance despite its 19 adjustable parameters. The resulting log-normalized mean squared error for single step predictions over the prediction set is 0.154. Of course, better performance is achieved for larger networks but the aim of this experiment was not achieve pure performance but instead a rigorous comparison between two methodologies.

5.2 Combining Network Outputs

	w_1	w_2	w_3	w_4	w_5	res
DRNN	-0.55	-1.04	-1.46	-0.65	-1.03	-1.24
MLP	-0.38	-1.00	-0.95	-0.38	-0.66	-1.12
Carbon copy	+0.06	-0.13	-0.21	-0.49	-1.12	-1.66

Table 1: Log-normalized single-step prediction errors for the five wavelet resolution coefficients and the residual. The more negative the outcome, the more favorable.

The same DRNN configuration was also used to forecast each wavelet resolution. Each network was fed with the prior series value for the series $W(t,j)$, at some time t, to provide an estimate of the next value for $W(t+1,j)$. So the prediction engine was called into operation 6 times, here, entirely independently. Once the univariate predictions were obtained for the series $W(t,j)$ for j between 1 and 6, the error was afterwards compared to other indexes of performance, namely the MLP (multilayer perceptron, more on which below) and a very basic index which we term the "carbon-copy" error, i.e. the error based on the most simple autoregressive forecast. This carbon-copy error was more favorable for the last coefficient and the residual (w_5 and res in Table 5.2. Note that these are the more smoothly-varying series resulting from the wavelet decomposition. This led us to benefit from improved performance of this simple autoregressive forecaster: we retained predictions by it, for $w5$ and res; and we retained predictions by the very much more sophisticated and long-memory DRNN for w_1, w_2, w_3 and w_4.

To illustrate the predictive ability of such a small DRNN model, an MLP of size $(6 \times 5 \times 1)$ having 41 weights including bias and fed with the delay vector $[w_i(t-1), w_i(t-2), \ldots, w_i(t-6)]$ for j between 1 and 5, was trained to provide comparative predictions. Selection of these dimensions was based mostly on trial and error. We remark that the MLP has twice as many parameters as the DRNN although better predictions are obtained for all wavelet scales with the DRNN as can be seen in Table 5.2.

The ability of the network to capture dynamical behavior over higer resolution levels deteriorates quite fast. This is intuitively clear in view of the plots. The higher the order of the resolution scale, the smoother the curve, and thus the less information the network can retrieve. This is also the reason why we stopped the wavelet decomposition process at order 6. From lag 5 onwards, the network outputs were replaced by the carbon copy estimates in these cases. The individual resolution coefficient predictions were afterwards summed together to form an estimate of the sunspot value, $x(t+1)$. The resulting log-normalized mean squared error for single step predictions over the prediction set is 0.123 with the hybrid DRNN and carbon copy method, instead of 0.209 with the hybrid MLP and carbon copy. Finally, this compares with 0.154 with use of the direct DRNN approach on the original time series (not wavelet-decomposed). Consequently, the combination of the DRNN on the wavelet decomposition provides improvement of the prediction accuracy.

Attempts to decompose other highly irregular meteorological data into wavelet resolutions has not yet proven to be successful. Further substantiation through more experiments and analysis are needed to ascertain the type of data best suited for a wavelet decomposition, as described in this article. Experiments being conducted at present will be reported on in due course.

6 Conclusion

In this paper, we have shown that the combination of several dynamical recurrent neural networks, aiming at capturing the dynamics of several multiresolution versions of a data signal, can aid in improving the prediction accuracy. This innovative approach combines data analysis and prediction is an integrated fashion. It leads to combination of connectionist engines, with outputs combined in a natural way. Therefore problem decomposition and information fusion are both handled in a closely coordinated, and indeed in an integrated, way. Future work will aim at specific guidelines, for varying classes of problem, as to the best way to combine network outputs in order to complement and improve on the forecasting abilities of the individual connectionist and other methods.

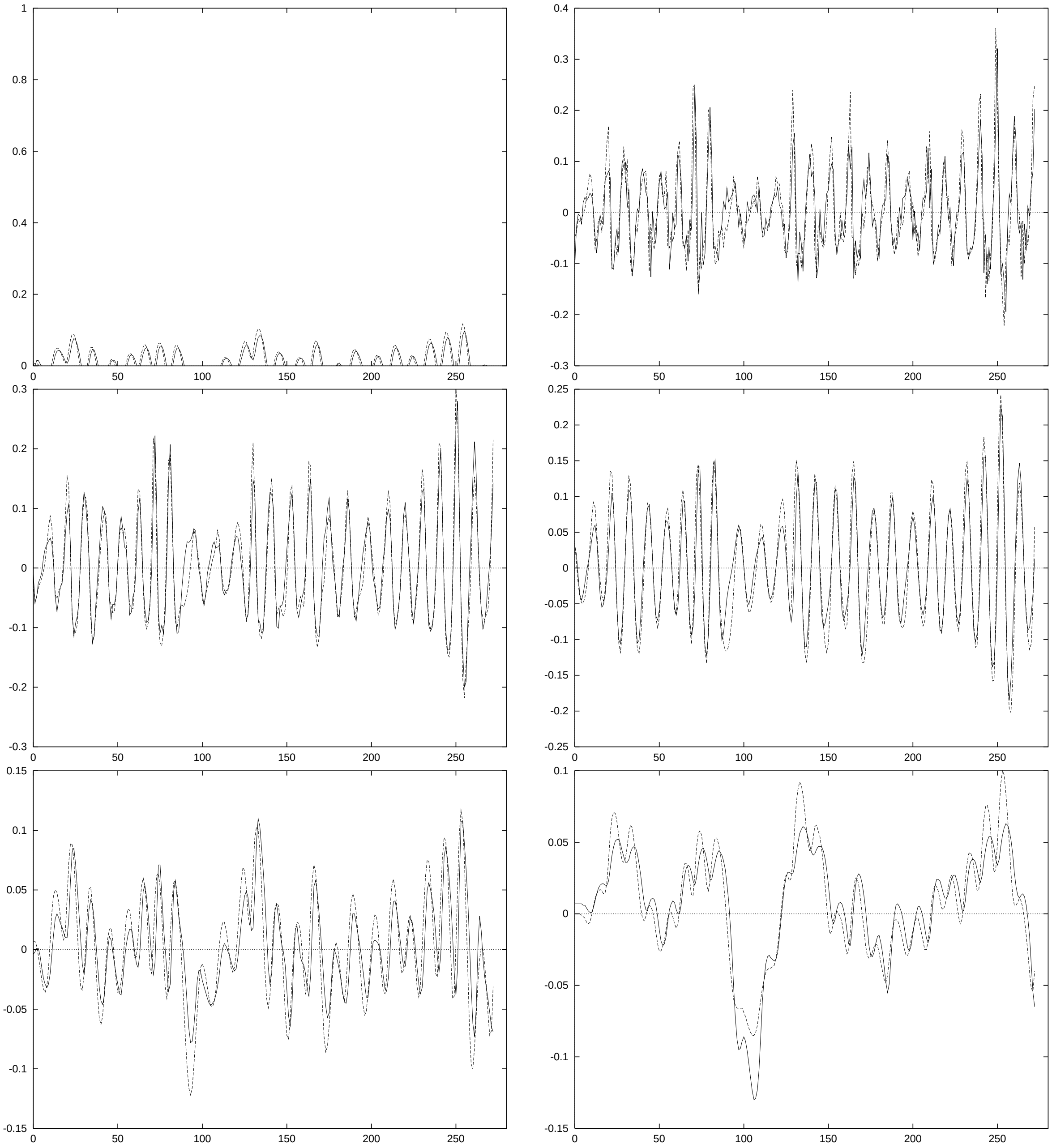

Figure 1: Single step prediction with several DRNNs. Top left: original and predicted sunspot series overplotted; the prediction was obtained from the combination of the individual wavelet resolution scale forecasts w_i for $i = 1, \ldots, 5$ and the residual. The w_i are shown from the top right to the bottom. The dashed curve corresponds to the actual series and the plain curve is the prediction.

References

[1] A. Aussem, F. Murtagh and M. Sarazin, "Dynamical recurrent neural networks - towards environmental time series prediction," *International Journal on Neural Systems*, 6, 145–170, 1995.

[2] A. Aussem, F. Murtagh and M. Sarazin, "Fuzzy astronomical seeing nowcasts with a dynamical and recurrent connectionist network," *Neurocomputing*, 1995, in press.

[3] A. Aussem, *Theory and applications of dynamical and recurrent neural networks towards prediction, modeling and adaptive control of dynamical processes*, Ph.D. Thesis, Université René Descartes, June, 1995.

[4] P. Baldi, "Gradient descent learning algorithm overview: a general dynamical systems perspective," *IEEE Trans. on Neural Networks*, 6, 182–195, 1995.

[5] A. Bruce and H.-Y. Goa, *S+Wavelets User's Manual*, Version 1.0, StatSci Division, MathSoft Inc., Seattle, WA, 1994.

[6] J.T. Connor, R.D. Martin and L.E. Atlas, "Recurrent neural networks and robust time series prediction," *IEEE Trans. on Neural Networks*, 5, 240–254, 1994.

[7] I. Daubechies, *Ten Lectures on Wavelets*, Phildelphia: Society for Industrial and Applied Mathematics (SIAM), 1992.

[8] J. Hertz, A. Krogh and R. Palmer, *An Introduction to the Theory of Neural Computation*, Redwood City, CA: Addison-Wesley, 1991.

[9] M. Holschneider and P. Tchamitchian, in *Les ondelettes en 1989*, P.G. Lemarié, ed., Berlin: Springer-Verlag, 102, 1990.

[10] Y. Meyer, *Wavelets: Algorithms and Applications*, Phildelphia: Society for Industrial and Applied Mathematics (SIAM) 1993.

[11] F. Murtagh, J.-L. Starck and A. Bijaoui, "Multiresolution in astronomical image processing: a general framework", *International Journal of Imaging Systems and Technology*, 6, 1995, in press.

[12] G.P. Nason and B.W. Silverman, "The stationary wavelet transform and some statistical applications", preprint, University of Bath, 1995.

[13] S.W. Piche, "Steepest descent algorithms for neural network controllers and filters," *IEEE Trans. on Neural Networks*, 5, 198–212, 1994.

[14] F.J. Pineda, "Generalization of back-propagation to recurrent neural networks," *Physical Review Letters* 59, 2229–2232, 1987.

[15] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery, *Numerical Recipes*, 2nd ed., Chapter 13, New York: Cambridge University Press, 1992.

[16] M.B. Priestley, *Spectral Analysis and Time Series*, New York: Academic Press, 1981.

[17] D.E. Rumelhart, G.E. Hinton and R.J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing Explorations in the Microstructure of Cognition, Vol. 1*, D.E. Rumelhart and J.L. McClelland, Eds., Cambridge, MA: MIT Press, Bradfords Books, 318–362, 1986.

[18] M.J. Shensa, "Discrete wavelet transforms: wedding the à trous and Mallat algorithms", *IEEE Transactions on Signal Processing*, 40, 2464–2482, 1992.

[19] J.-L. Starck and A. Bijaoui, "Filtering and deconvolution by the wavelet transform", *Signal Processing*, 35, 195–211, 1994.

[20] J.L. Starck, F. Murtagh, A. Bijaoui, "Multiresolution support applied to image filtering and deconvolution", *Graphical Models and Image Processing*, 57, 420–431, 1995.

[21] H. Tong, *Non Linear Time Series.* Oxford: Clarendon Press, 1990.

[22] E.A. Wan, *Finite Impusle Response Neural Networks with Applications in Time Series Prediction.* Ph.D. Thesis, Stanford University, November 1993.

[23] A.S. Weigend, D.E. Rumelhart and B.A. Huberman, "Predicting the future: a connectionist approach," *International Journal of Neural Systems*, 1, 195–220, 1990.

[24] P. Werbos, *Beyond regression: new tools for prediction and analysis in behavioral sciences*, Ph.D. thesis, Harvard University, Cambridge, MA, August 1974.

[25] R.J. Williams and D. Zipser, "Experimental analysis of the real-time recurrent learning algorithm," *Connection Science*, 1, 87–111, 1989.

[26] G.U. Yule, "On a method of investigating periodicities in disturbed series with special reference to Wolfer's sunspot numbers," *Philos. Trans. Roy. Soc. London Ser. A*, 226, 267, 1927.