Journal of VLSI Signal Processing 48, 99–108, 2007
© 2006 Springer Science + Business Media, LLC. Manufactured in The Netherlands.
DOI: 10.1007/s11265-006-0025-6

Sequential Data Fusion via Vector Spaces: Fusion of Heterogeneous Data in the Complex Domain

DANILO P. MANDIC AND SU LEE GOH
Department of Electrical and Electronic Engineering, Imperial College of Science, Technology and Medicine, London, UK

KAZUYUKI AIHARA
Institute of Industrial Science, University of Tokyo, Tokyo, Japan

Received: 1 May 2006; Revised 10 August 2006; Accepted 10 October 2006

Abstract. A sequential data fusion approach via higher dimensional vector spaces is introduced. This is achieved by making use of the representation of directional signals within the field of complex numbers C. The concept of data fusion is next introduced and the place of the proposed approach within that framework is identified. The benefits of such an approach are illustrated and a range of possible applications is shown. The concept introduced is supported by a real world case study which focuses on simultaneous forecasting of wind speed and direction. The architectures and learning algorithms which support this concept are introduced and their distributed sequential fusion nature is highlighted.

Keywords: data fusion, spatio-temporal, complex-valued, vector spaces, complex nonlinearity

1. Introduction

The data fusion framework aims at exploiting synergy between the information fragments, with the aim to make better use of the available data [1]. In simple terms, data fusion aims at obtaining information of greater quality [4], from the constituent single information sources. The exact definition of "greater quality" depends on the application, and may mean a better spatial coverage, better filtering performance, or more efficient modelling of higher order spatio-temporal data dependencies. In practical terms, the data fusion approach combines data from multiple sensors (and associated databases if appropriate) to achieve improved accuracies and more specific inferences that could not be achieved by the use of only a single sensor.[1]

Data fusion principles apply to many domains, and have been (often implicitly) at the core of modern applications in the diverse areas spanning engineering, computing, and biomedicine. The recent interest in the theory and taxonomy of multisensor data fusion has been reflected by a number of special issues of leading international journals and conferences, which have been dedicated to this area [1, 6].

Our aim in this paper is to introduce the concept of *data fusion via vector spaces*. This concept belongs to transform domain fusion, for which no general theory exists as yet. Based on a case study of wind forecasting, we illustrate that a fusion of the heterogeneous wind speed and direction data in the field of complex numbers C allows for the simultaneous modelling and forecasting of these wind field components. Simulations show that the proposed approach comprehensively outperforms the standard univariate approaches.

2. Data Fusion via Vector Spaces

Before we proceed with our proposed fusion model, let us introduce some important notions used later in the paper:

- *Vectorial neural network (NN)* is a network with multiple heterogeneous inputs, operating in R. For more details, please refer to [13]. This is effectively a *multivariate* model with two inputs, which may or may not be correlated. In the case of wind measurements, for instance, these inputs are wind speed and direction. These can be used to produce an estimate of the wind speed in a forecasting configuration. However, no density theorem has as yet been proven for vectorial NNs, which contributes to uncertainty when employing this class of NNs for the function approximation and prediction applications [13];
- "*Complex*" *or Complex-valued* refers to the situation where the inputs, outputs, and weights in the network are quantities in the field of complex numbers C. This interpretation benefits from the well-defined rules of complex algebra, such as the addition, subtraction, multiplication and division operations in C. This also applies to higher-order complex spaces in which hyper-complex NNs (tensor) are defined using the same principle [13, 14];
- *Complex nonlinearity*, refers to a general function $f: C \to C$, whereby we differentiate between two main classes of complex nonlinearities, which are used as activation functions (AFs) within the complex valued neurons. In the case of a *split-complex* AF, the real and imaginary component of the complex-valued signal x are separated and fed independently through a real-valued activation function. The functional expression of the *split-complex* activation function is given by $f(x) = f_R(Re(x)) + jf_I(Im(x))$. A *fully-complex* activation function is a general complex activation function where there is cross-correlation between the real and imaginary channel;
- *Cauchy–Riemann equations*[2] state that the partial derivatives of a function $f(z) = u(x,y) + jv(x,y)$ along the real and imaginary axes should be equal: $f'(z) = \frac{\partial u}{\partial x} + j\frac{\partial v}{\partial x} = \frac{\partial v}{\partial y} - j\frac{\partial u}{\partial y}$ Therefore, the Cauchy–Riemann equations are as: $\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y}, \frac{\partial v}{\partial x} = -\frac{\partial u}{\partial y}$.

2.1. *Taxonomy of Data Fusion*

When approaching a problem from the data fusion viewpoint, we differentiate between the following levels of abstraction [6]:

- *Observation/measurement space* contains vectors of measurement functions which can be univariate, multivariate, and/or multidimensional. It may be possible to build a state-space model, or to assess the data modality [5];
- *Transform domain representations*, which seek features from time and/or frequency models (fast Fourier transform (FFT), (nonlinear) autoregressive (N)ARMA models [8], wavelet), blind processing (independent component analysis (ICA), blind source separation (BSS)), particle/Kalman filter [9], kernels and support vector machines (SVM) [7], kernel ICA [15]);
- *Decision space*, where the classes within the data fusion model (and the corresponding basins of attraction from the measurement space) are mapped into the relevant probabilities of the occurrence of an event.

We here illustrate a sequential data fusion in a complex space, which is supported by an example of wind forecasting. This is an important issue, and has recently received much attention due to the interest in wind as an alternative source of energy, since the output power P of a wind turbine (WT) is a nonlinear function of wind speed v ($P \sim v^3$). Neural network (NN) forecasting methods are attractive for this purpose, since they are perfectly suitable to deal with nonstationarity and nonlinearity within the data and the associated higher order statistics [10]. In particular, temporal recurrent neural networks (RNNs) represent dynamical maps, which not only model adaptively nonlinearity and non-Gaussianity [8], but also the associated nonlinear dynamics.

2.2. *Problem Setting*

The information entering a fusion process should be aligned. This can be applied to both homogeneous (commensurate) and heterogeneous (non- commensurate) information, and may require some sort of conversion or transformation of observations [11]. The problem of building a data fusion model via the complex vector space is illustrated in Fig. 1, where the left diagram represents wind measurements as a vector of its speed $v(k)$ and direction $\theta(k)$ components, in the $N \div E$ coordinate system, whereas the right diagram in Fig. 1 illustrates the distribution of wind speeds over various directions. There is a clear inter-dependence between wind signal components, a fact that is not taken into account in the current approaches to wind forecasting. Although both the speed and direction are integral components of the wind signal, in practical applications, only the speed component is taken into account, hence introducing a systematic error in forecasts [12].

For nonlinear modelling of both wind speed and direction, vectorial temporal NNs seem obvious candidates. This suggests the use of multidimensional networks developed in an associative and division algebra, in which such density theorems do exist. This is exactly the case with complex algebra. Indeed, from Fig. 1, the wind vector $\mathbf{v}(k)$ can be expressed in the complex domain C as

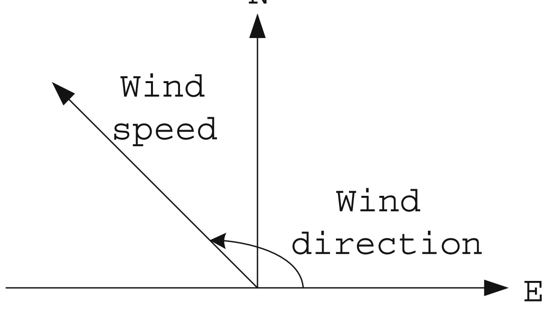
$$\mathbf{v}(k) = |\mathbf{v}(k)|e^{\jmath\theta(k)} = v_E(k) + \jmath v_N(k) \qquad (1)$$

This interpretation now benefits from the properties of complex algebra, and also facilitates *data fusion* via vector spaces.

To clarify this, observe that the two wind components speed v and direction θ (which are of different natures, that is, heterogeneous) are modelled as a single quantity in a complex representation space. If the East and North wind speed components are denoted respectively by v_E and v_N, they are effectively projections of the wind field vector v on the corresponding axes, that is, $v(k) = v_E(k) + \jmath v_N(k)$.

2.3. Data Fusion and Sufficient Information

One of the first proposed data fusion models was the "waterfall model" (Fig. 2), developed for the UK Defence Evaluation Research Agency (DERA). This model reflects the different hierarchical levels in a data fusion setting. We can think of the heterogeneous sensors monitoring a certain process as being "windows" into the phenomenon under observation. Sensors can either have their own window, or the windows "overlap" in space or time. This way, the information obtained can be thought of as "decomposed" or "fragmented" by the sensors, which is sometimes called sensor fission [6], and is related to so-called *sufficient information* (whether



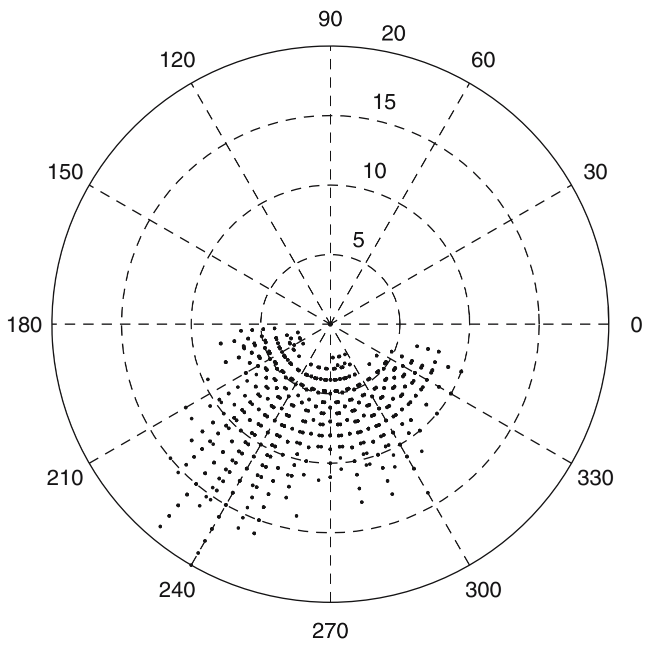


Figure 1. Wind recordings: *Left:* a complex-valued representation, *Right:* wind lattice.

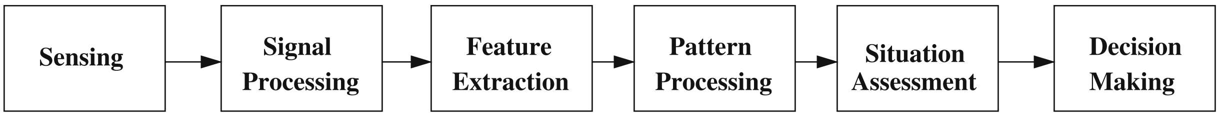


Figure 2. The waterfall model.

the character and number of sensors can indeed describe the phenomenon sufficiently well). This is analogous to the concept of *embeddology*, where we wish to model the nonlinear dynamics of a multidimensional process based on its time delay representation [5]. The information fragments coming from sensors are exposed to spectral shaping, saturation, and noise; *data fusion* aims at retrieving the "interesting" characteristics of the phenomenon.

3. Architectures and Performance Aspects

Combining multi-sensor data in the data fusion framework has the potential of faster and cheaper processing and allows for new interfaces, together with reducing overall uncertainty (increase in reliability). Such data can be combined in various ways, for instance by: (1) linear combiner, (2) combination of posteriors (weights, model significance), (3) product of posteriors (independent information). Based on the different ways of combining information and different semantic levels, we differentiate between the following data fusion architectures, shown in Fig. 3:

- *Centralised:* characterised by fairly simple algorithms, but rather inflexible to sensor malfunction and failures;
- *Hierarchical:* suitable for so-called collaborative processing and allows for two way communication;
- *Decentralised:* designed to be robust to sensor malfunction and failures, but requires mathematically rather demanding algorithms.

3.1. Practical Considerations

We have seen that we can use complex-valued algebra in order to perform fusion of the heterogeneous wind speed and direction data, however, it is natural to ask whether this will help to obtain information of "greater quality" from the data, as required in the data fusion framework. The answer to this question depends on the relationship between the components of a multivariate measurement process. Some tests for the variable dependence within a signal model (bivariate vs. complex-valued) can be found in [16].

We next illustrate the importance of testing for component dependencies within the proposed framework, by comparing the one-step ahead forecasting performances using three NN based approaches:

(1) The standard *univariate* approach, where the wind signal components (speed and direction) are predicted separately and then put back together to form a complex vector;
(2) The *split*-complex approach, where the wind speed and direction are modelled in the complex algebra, with the split-complex nonlinear activation of neurons;

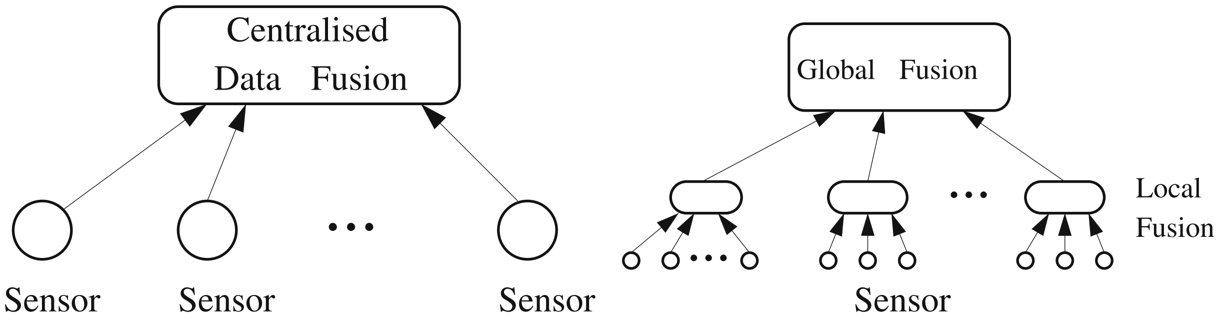


Figure 3. Centralised and hierarchical data fusion.

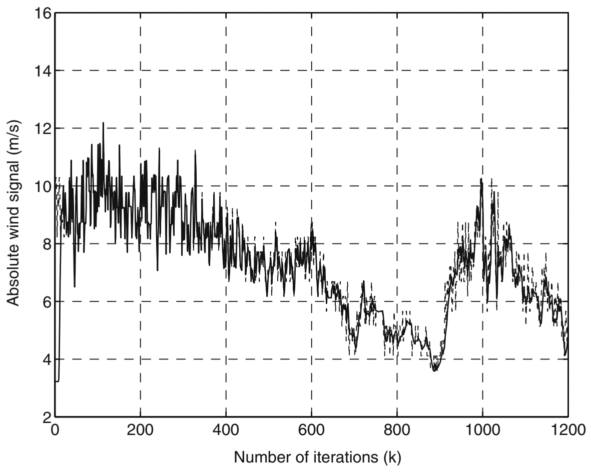


Figure 4. *Fully*-complex approach.

(3) The *fully*-complex representation in the complex algebra, with a general complex-valued nonlinear activation function of neurons.

For the univariate approach, an RNN architecture trained by the Real Time Recurrent Learning (RTRL) [2] algorithm was used, whereas for the split- and fully-complex approach we used a complex RNN trained by the complex-valued RTRL (CRTRL) [17]. The simulation results on the prediction of complex-valued real-world wind measurements, for the *univariate*, *split*- and *fully*-complex case are shown respectively in Figs. 4, 5 and 6. Observe that the *fully* CRTRL algorithm exhibited better and more consistent performance than the split-complex and univariate approaches as indicated by the solid line (predicted signal) being much more in accordance with the dotted line (actual signal).

3.2. *Distributed Data Fusion via Vector Spaces*

After having established the concept of data fusion via vector spaces, we next address the issue of data locality, and identify some architectures for distributed signal processing within this framework. Block diagrams of the two main topologies in distributed data/sensor processing are shown in Fig. 7, these configurations are:

- *Parallel:* Sensors (blocks, modules) do not communicate with each other and there is no feedback from the fusion centre to any sensor;
- *Serial:* The $(m-1)$th sensor passes its information to the mth sensor. The first sensor in the network uses only its observation to make a local decision. The last sensor makes a final decision.

Notice that the topology depicted in Fig. 7a represents a "mixture of experts" and can, for instance, be implemented as an RBF, whereas the topology in Fig. 7b has its direct implementation as the Pipelined Recurrent Neural Network (PRNN) [18]. In this paper, we only consider the latter case.

4. RNN Topologies and Learning Algorithms

We shall now introduce the architectures and algorithms for the training of complex RNNs and the architecture of the proposed sequential data fusion model realised as a complex PRNN.

4.1. *The Basic Algorithm*

Figure 8 shows a Fully Connected Recurrent Neural Network (FCRNN), which consist of N neurons with p external inputs. The network has two distinct layers consisting of the external input-feedback layer and a layer of processing elements. Let $y_l(k)$ denote the complex-valued output of each neuron, $l = 1, \ldots, N$ at time index k and $\mathbf{s}(k)$ the $(1 \times p)$ external complex-valued input wind signal vector. The overall input to the network $\mathbf{Z}(k)$ represents a

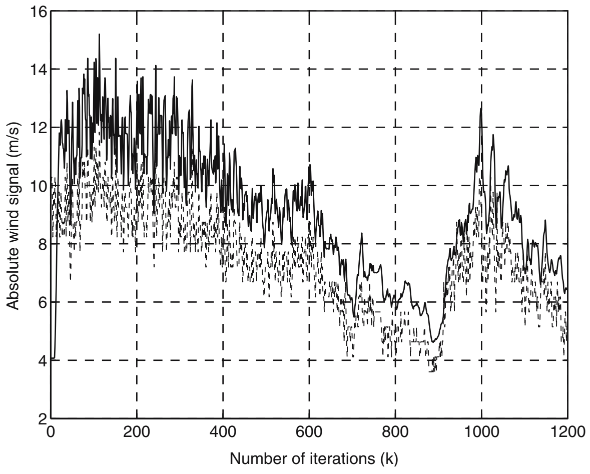


Figure 5. Dual *univariate* approach.

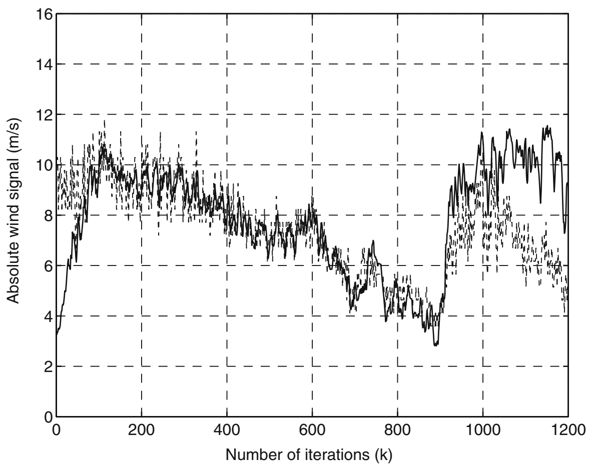


Figure 6. *Split*-complex approach.

concatenation of vectors $\mathbf{y}(k)$, $\mathbf{s}(k)$ and the bias input $(1+j)$, and is given by

$$\begin{aligned}\mathbf{Z}(k) &= [s(k-1),\dots,s(k-p),1+j,\\ &\qquad y_1(k-1),\dots,y_N(k-1)]^T\\ &= Z_n^r(k)+jZ_n^i(k),\\ n &= 1,\dots,p+N+1\end{aligned} \tag{2}$$

where $j=\sqrt{-1}$ and the superscripts $(\cdot)^r$ and $(\cdot)^i$ denote respectively the real and imaginary part of a complex number. A complex-valued weight matrix for the network is denoted by $\mathbf{W}$, where for the nth neuron, its weights form a $(p+N+1)\times 1$ dimensional weight vector $\mathbf{w}_l^T = \left[w_{l,1},\dots,w_{l,p+N+1}\right]$. The output of each neuron can be expressed as $y_l(k)=\Phi(net_l(k)),\quad l=1,\dots,N$, where

$$net_l(k) = \sum_{n=1}^{p+N+1} w_{l,n}(k)Z_n(k) \tag{3}$$

is the net input to lth node at time index k. For simplicity, we state that

$$\begin{aligned}y_l(k) &= \Phi^r(net_l(k))+j\Phi^i(net_l(k))\\ &= u_l(k)+jv_l(k)\end{aligned} \tag{4}$$

where Φ is a complex nonlinear activation function of a neuron.

4.2. *Complex-Valued Real Time Recurrent Learning (CRTRL) Algorithm*

The system configuration in Fig. 8 is employed for the task of adaptive prediction. The output error which consists of its real and imaginary parts can be expressed as

$$e(k) = d(k)-y_1(k) = e^r(k)+je^i(k) \tag{5}$$

$$e^r(k) = d^r(k)-u_1(k), \quad e^i(k) = d^i(k)-v_1(k) \tag{6}$$

where $d(k)$ is the teaching signal. For real-time applications and gradient descent algorithms the cost function of the network is given by $E(k)=\frac{1}{2}\times|e(k)|^2=\frac{1}{2}e(k)e^*(k)=\frac{1}{2}\left[(e^r)^2+(e^i)^2\right]$ [6], where $(\cdot)^*$ denotes the complex conjugate. The CRTRL aims at minimising the error by recursively altering the weight coefficients based on the gradient search technique, given by

$$w_{l,n}(k+1) = w_{l,n}(k)-\eta\nabla_{w_{l,n}}E(k)\big|_{w_{l,n}=w_{l,n}(k)} \tag{7}$$

Notice that $E(k)$ is a real-valued function and to calculate the gradient, the partial derivates of $E(k)$ with respect to the real and imaginary part of the weight coefficients separately have to be derived, as

$$\nabla_{w_{l,n}}E(k) = \frac{\partial E(k)}{\partial w_{l,n}^r}+j\frac{\partial E(k)}{\partial w_{l,n}^i} \tag{8}$$

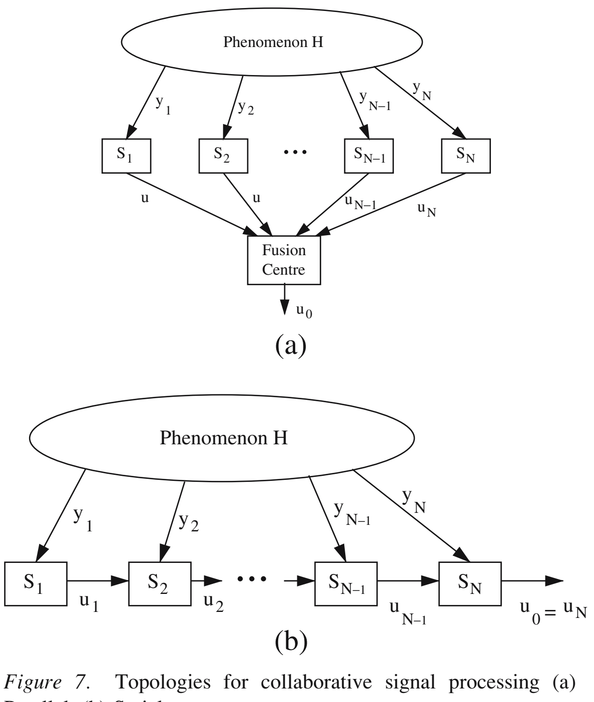


Figure 7. Topologies for collaborative signal processing (a) Parallel, (b) Serial.

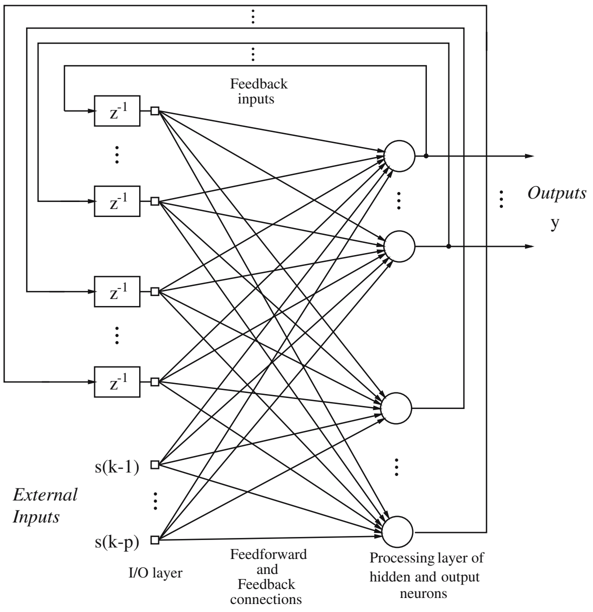

Feedback inputs
z-1
Outputs
y
s(k-1)
s(k-p)
External Inputs
I/O layer
Feedforward and Feedback connections
Processing layer of hidden and output neurons

Figure 8. A fully connected recurrent neural network for prediction.

Calculating the gradient of the cost function with respect to the real part of the complex weight gives[3]

$$\frac{\partial E(k)}{\partial w^r_{l,n}(k)} = \frac{\partial E}{\partial u_1}\left(\frac{\partial u_1(k)}{\partial w^r_{l,n}(k)}\right) + \frac{\partial E}{\partial v_1}\left(\frac{\partial v_1(k)}{\partial w^r_{l,n}(k)}\right) \tag{9}$$

Similarly, the derivative of the cost function with respect to the imaginary part of the complex weight yields

$$\frac{\partial E(k)}{\partial w^i_{l,n}(k)} = \frac{\partial E}{\partial u_1}\left(\frac{\partial u_1(k)}{\partial w^i_{l,n}(k)}\right) + \frac{\partial E}{\partial v_1}\left(\frac{\partial v_1(k)}{\partial w^i_{l,n}(k)}\right), \tag{10}$$

The factors $\frac{\partial u_1(k)}{\partial w^r_{l,n}(k)}$, $\frac{\partial v_1(k)}{\partial w^r_{l,n}(k)}$, $\frac{\partial u_1(k)}{\partial w^i_{l,n}(k)}$ and $\frac{\partial v_1(k)}{\partial w^i_{l,n}(k)}$ are measures of sensitivity of the output of the lth neuron at time k to a small variation in the value of $w_{l,n}(k)$. For convenience, denote the corresponding sensitivities by $\pi^{rr}(k) = \frac{\partial u_1(k)}{\partial w^r_{l,n}(k)}$, $\pi^{ir}(k) = \frac{\partial v_1(k)}{\partial w^r_{l,n}(k)}$, $\pi^{ri}(k) = \frac{\partial u_1(k)}{\partial w^i_{l,n}(k)}$ and $\pi^{ii}(k) = \frac{\partial v_1(k)}{\partial w^i_{l,n}(k)}$. For a complex function to be analytic at a point in C, it needs to satisfy the Cauchy–Riemann equations [5]. To arrive at the Cauchy–Riemann equations, the partial derivatives (sensitivities) along the real and imaginary axes should be equal, that is $\pi(k) = \pi^{rr}(k) + j\pi^{ir}(k) = \pi^{ii}(k) - j\pi^{ri}(k)$. Equating the real and imaginary parts of the sensitivities leads to

$$\pi^{rr}(k) = \pi^{ii}(k), \pi^{ri}(k) = -\pi^{ir}(k) \tag{11}$$

By using the Cauchy–Riemann equations, a more compact representation of $\nabla_{\mathbf{w}} E(k)$ is obtained as

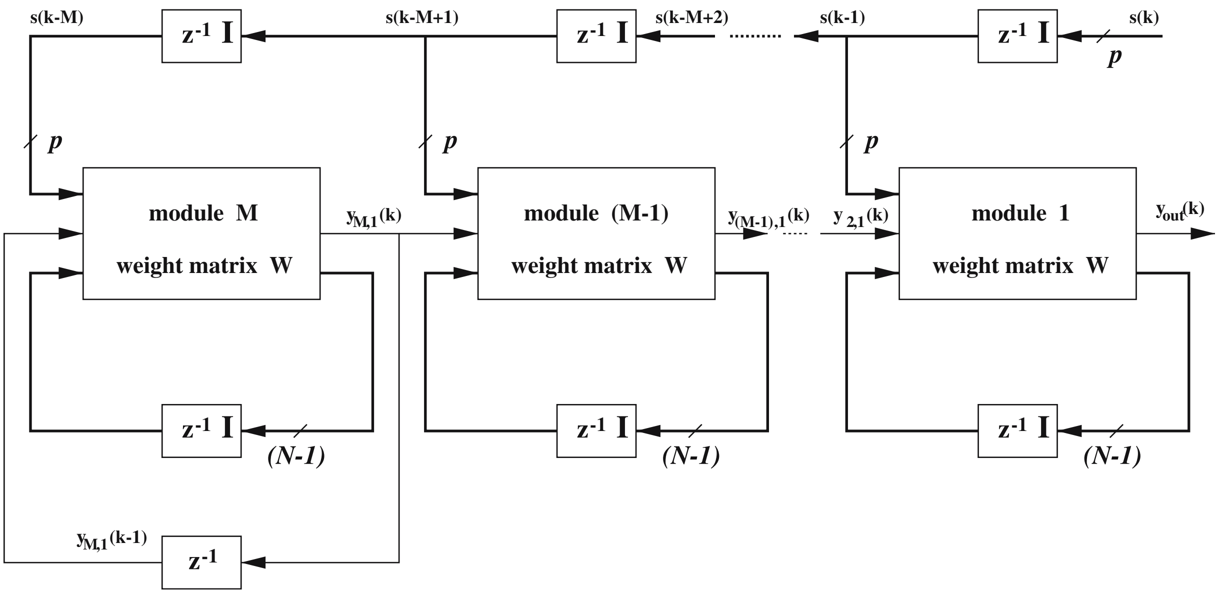


Figure 9. The pipelined recurrent neural network.

Table 1. Statistical properties of the wind data sets.

Set	W1 (1 m)	W2 (17 m)
Cumulative samples	199,744	187,648
Minimum speed [m/s]	0	0.0182
Maximum speed [m/s]	3.0746	21.6548
Mean speed [m/s]	0.0917	5.5228
Standard deviation [m/s]	0.7306	3.1786

$\nabla_{\mathbf{W}} E(k) \equiv e(k)\pi^*(k)$. The total weight matrix update is then

$$\mathbf{W}(k+1) = \mathbf{W}(k) + \eta e(k)\pi^*(k) \qquad (12)$$

with the initial condition $\pi(0) = \mathbf{0}$. Following the approach from [7], the update for the sensitivities $\pi^*(k) = \pi^{rr}(k) + j\pi^{ri}(k)$ can be derived as

$$\pi^*(k) = \{\Phi^*(k)\}' \left[\delta_{ln} Z_n^*(k) + \mathbf{w}_1^*(k)\pi^*(k-1)\right] \qquad (13)$$

where

$$\delta_{ln} = \begin{cases} 1, & l = n \\ 0, & l \neq n \end{cases} \qquad (14)$$

is the Kronecker delta.

4.3. Complex Pipelined Recurrent Neural Network (PRNN)

The PRNN is shown in Fig. 9 and it consists of M modules, each of which represents an RNN with N neurons. The first module in the PRNN is an RNN, whereas all other modules receive an input from the preceding module. Therefore, the whole concept mimics precisely the serial topology in distributed data/sensor fusion. The $(p \times 1)$ dimensional external complex-valued signal vector $\mathbf{s}^T(k) = [s(k-1), \ldots, s(k-p)]$ is delayed by m time steps $(z^{-m}\mathbf{I})$ before feeding the module m, where z^{-m}, $m = 1, \ldots, M$ denotes the m-step time delay operator, and $\mathbf{I}$ is a $(p \times p)$ dimensional identity matrix. The complex-valued weight vectors $\mathbf{w}_l$ are embodied in an $(p+N+1) \times N$ dimensional weight matrix $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_N]$. All the modules operate using the same weight matrix $\mathbf{W}$. The following equations provide a mathematical description of the CPRNN,

$$y_{t,l}(k) = \Phi(net_{t,l}(k)), \quad t = 1, 2, \ldots, M \qquad (15)$$

$$net_{t,l}(k) = \sum_{n=1}^{p+N+1} w_{l,n}(k) P_{t,n}(k), \quad \begin{array}{r} l = 1, \ldots, N \\ n = 1, \ldots, p+N+1 \end{array} \qquad (16)$$

$$\begin{aligned} \mathbf{P}_t^T(k) = [&s(k-t), \ldots, s(k-t-p+1), 1+j, \\ &\times y_{t+1,1}(k-1), y_{t,2}(k-1), \ldots, y_{t,N}(k-1)] \end{aligned}$$
$$\text{for} \quad 1 \leq t \leq M-1 \qquad (17)$$

$$\begin{aligned} \mathbf{P}_M^T(k) = [&s(k-M), \ldots, s(k-M-p+1), 1+j, \\ &\times y_{M,1}(k-1), y_{M,2}(k-1), \ldots, y_{M,N}(k-1)] \end{aligned}$$
$$\text{for} \quad t = M \qquad (18)$$

For simplicity we state that

$$\begin{aligned} y_{t,l}(k) = \Phi(net_{t,l}(k)) &= \Phi^r(net_{t,l}(k)) \\ &+ j\Phi^i(net_{t,l}(k)) \\ &= u_{t,l}(k) + jv_{t,l}(k) \end{aligned} \qquad (19)$$

The overall output of the CPRNN is $y_{1,1}(k)$, that is the output of the first neuron of the first module, and Φ is a nonlinear complex-valued activation function of a neuron.

5. Simulations

To illustrate the benefits of data fusion via vector spaces, together with those associated with the use of serial topology in distributed signal processing, we perform simultaneous forecasting of wind signal components using a complex-valued PRNN. The wind measurements used in simulations comes from an AM ultrasonic anemometer and were sampled at 50 Hz for an hour long interval. The measurements

Table 2. Parameter selection details.

Wind measurements	M	N	p
W1	8	2	10
W2	15	2	18

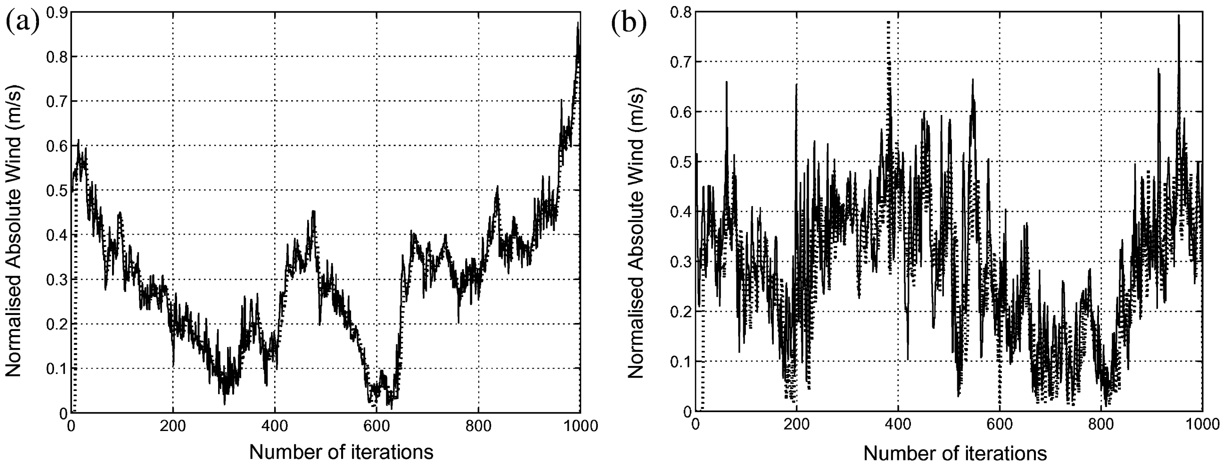


Figure 10. Prediction of wind using CPRNN. *Solid curve*: actual series. *Dashed curve*: predicted series.

were recorded at one meter and 17 m high above ground level. The difference in height during measurements gave different wind dynamics. Table 1 shows the statistical properties of the wind data sets considered. The initial weight values of the network were chosen randomly. A temporal modular complex-valued PRNN predictor was trained using a version of complex CRTRL [19] with 1,000 data points of the complex wind measurements. For the experiments, the nonlinearity at the neuron was chosen to be the complex *tanh* function, given by

$$\Phi(x) = \frac{e^{\beta x} - e^{-\beta x}}{e^{\beta x} + e^{-\beta x}} \tag{20}$$

where $x \in C$. The value of the slope of $\Phi(x)$ was $\beta = 1$. The value of the learning rate for the CPRNN architecture was $\eta = 0.01$. The forgetting factor for the CPRNN architecture was $\lambda = 0.995$. Table 2[4] shows the complex PRNN parameters selected for each wind measurements. The measurement used to assess the performance was the prediction gain [18]

$$R_p \equiv 10\log_{10}\left(\frac{\sigma_{\mathbf{s}}^2}{\sigma_e^2}\right)[dB] \tag{21}$$

Table 3. Prediction gain R_p [dB].

Wind measurements (in dB)	RNN	CPRNN
W1	12.9512	19.5635
W2	7.8491	14.2488

where $\sigma_{\mathbf{s}}^2$ denotes the variance of the input signal $\mathbf{s}(k)$, and σ_e^2 denotes the estimated variance of the forward prediction error $e(k)$. The simulation results for complex wind data are shown in Fig. 10. The prediction performance of the CPRNN applied to the real-world (velocity and angle components) wind signal showed the ability of the CPRNN to adapt to the changes in the dynamics of the complex wind input. The prediction gain obtained for each measurement using a single module RNN and the CPRNN is shown in Table 3. In both measurements, there was a significant improvement in the prediction gain when the CPRNN architecture was employed over the performance of a single module RNN. This also justifies the use of serial PRNN topology, which is capable of modelling efficiently higher order dependencies between the data.

6. Conclusions

A vector space based approach for the fusion of heterogeneous data has been presented. This has been achieved in the field of complex numbers C, and can be straightforwardly extended to the case of hypercomplex spaces. Next, we have addressed the, recurrent neural network (RNN) based data fusion in such spaces, and identified the pipelined recurrent neural network (PRNN) as a convenient collaborative signal processing topology for this application. This combination of modular neural networks and data fusion via vector spaces has shown excellent performance on forecasting of real world wind measurements.

Notes

1. This concept is hardly new: living organisms have the capability to use multiple senses to learn about the environment. The brain then *fuses* all this available information to perform a decision task.
2. The Cauchy–Riemann equations helped to relax the requirements on the desired properties of nonlinerities within fully complex neurons which are to be analytic and bounded almost everywhere in C.
3. We derive the CRTRL for prediction applications (only one output y_1), however the derivation is general enough to be straightforwardly extended to a RNN with more than one output.
4. Please refer to [3] for these choices of parameters.

References

1. D. L. Hall and J. Llinas, "An Introduction to Multisensor Data Fusion," *Proc. IEEE*, vol. 5, 1997, pp. 6–23.
2. R. J. Williams and D. A. Zipser, "A Learning Algorithm for Continually Running Fully Recurrent Neural Networks," *Neural Comput.*, vol. 1, no. 2, 1989, pp. 270–280.
3. S. L. Goh and D. P. Mandic, "Nonlinear Adaptive Prediction of Complex-valued Signals by Complex-valued PRNN," *IEEE Trans. Signal Process.*, vol. 53, no. 5, 2005, pp. 1827–1836.
4. L. Wald, "Some Terms of Reference in Data Fusion," *IEEE Trans. Geosci. Remote Sens.*, vol. 37, 1999, pp. 1190–1193.
5. H. Kantz and T. Schreiber, *Nonlinear Time Series Analysis*, 2nd edn. Cambridge University Press, 2004.
6. D. P. Mandic et al., "Data Fusion for Modern Engineering Applications: An Overview," in *Proceedings of ICANN'05*, 2005, pp. 715–721.
7. C. Zhu and A. Kuh, "Sensor Network Loc. Using Pat. Rec.," in *Proc. of HISC*, 2005.
8. D. P. Mandic and J. A. Chambers, *Recurrent Neural Networks for Prediction: Architectures, Learning Algorithms and Stability*, Wiley, 2001.
9. G. Deco and D. Obradovic, *An Information–Theoretic Approach to Neural Computing*, Springer, 1997.
10. S. Haykin, "Neural Networks Expand SP's Horizons," *IEEE Signal Process. Mag.*, vol. 13, 1996, pp. 24–49.
11. (http://www.data-fusion.org/article.php?sid=70).
12. M. C. Alexiadis, P. S. Dokopoulos, H. S. Sahsamanoglou, and I. M. Manousadiris, "Short-term Forecasting of Wind Speed and Related Electrical Power," *Sol. Energy*, vol. 63, 1998, pp. 61–68.
13. P. Arena, L. Fortuna, G. Muscato, and M. G. Xibilia, *Neural Networks in Multidimensional Domains*, Springer, 1998.
14. T. Kim and T. Adali, "Approximation by Fully Complex Multilayer Perceptrons," *Neural Comput.*, vol. 15, 2003, pp. 1641–1666.
15. C. Fyfe and P. L. Lai, "ICA Using Kernel Canonical Correlation Analysis," in *Proceedings of the International Workshop On Independent Component Analysis and Blind Signal Separation*, 2002, pp. 279–284.
16. T. Gautama, D. P. Mandic, and M. V. Hulle, "A Nonparametric Test for Detecting the Complex-valued Nature of Time Series," *Int. J. Knowl. Base Intell. Eng. Syst.,* vol. 8, 2004, pp. 99–106.
17. S. L. Goh and D. P. Mandic, "A General Complex Valued RTRL Algorithm for Nonlinear Adaptive Filters," *Neural Comput.*, vol. 16, 2004, pp. 2699–2731.
18. S. Haykin and L. Li, "Nonlinear Adaptive Prediction of Nonstationary Signals," *IEEE Trans. Signal Process.*, vol. 43, 1995, pp. 526–535.
19. S. L. Goh and D. P. Mandic, "Nonlinear Adaptive Prediction of Complex-valued Signals by Complex-valued PRNN," *IEEE Trans. Signal Process.*, vol. 53, 2005, 1827–1836.