U.S. DEPARTMENT OF COMMERCE
NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION
NATIONAL WEATHER SERVICE
NATIONAL CENTERS FOR ENVIRONMENTAL PREDICTION
4700 Silver Hill Road, Mail Stop 9910
Washington, DC 20233-9910

TECHNICAL NOTE[1]

USE OF NEURAL NETWORKS TO IMPROVE COMPUTATIONAL EFFICIENCY OF ENVIRONMENTAL NUMERICAL MODELS

Vladimir M. Krasnopolsky[2], Dmitry V. Chalikov[3], and Hendrik L. Tolman[3]

Ocean Modeling Branch, Environmental Modeling Center,
National Centers for Environmental Prediction, NWS, NOAA,
5200 Auth Rd., Camp Springs, MD 20746

May 2001

---

[1]OMB Contribution No. 199

[2]SAIC/GSC, corresponding author e-mail address: Vladimir.Krasnopolsky@noaa.gov

[3]UCAR project scientist

Abstract

A new generic approach to improve computational efficiency of certain processes in numerical environmental models is formulated. This approach is based on the neural networks (NN) technique. It can be used to accelerate the calculations and improve the accuracy of the parameterizations of several types of physical processes which generally require computations involving complex mathematical expressions, including differential and integral equations, rules, restrictions and highly nonlinear empirical relations based on physical or statistical models. It is shown that, from a mathematical point of view, such parameterizations can usually be considered as continuous mappings (continuous dependencies between two vectors). It is also shown that NNs are a generic tool for fast and accurate approximation of continuous mappings and, therefore, can be used to replace primary parameterization algorithms. In addition to fast and accurate approximation to the primary parameterization, NN also provides the entire Jacobian for very little computation cost.

Two particular applications of the NN approach are presented here: (1) a NN approximation of the UNESCO equation of state of the sea water (density of the seawater) and an inversion of this equation (salinity of the seawater); and (2) a NN approximation for the nonlinear wave-wave interaction. The first application has been implemented in NCEP oceanic forecasting model, and the second one is being developed for wind wave models.

NNs for the density and salinity of the seawater generate the density and salinity with the accuracy close to that of the original UNESCO equation; however, the NN density equation is from two to four times faster than the UNESCO equation, and the NN salinity equation is orders of magnitude faster than the procedure based on the numerical inversion of the UNESCO equation. A NN based nonlinear wave-wave interaction approximation is also several orders of magnitude faster than a computation of the exact solution. It is twice more accurate than the so-called Discrete Interaction Approximation, which is currently used in wind wave models, and requires only 4-5 times more computational effort (without optimization), while is well within the range of what is considered as an acceptable increase in the computational efforts for a wave forecast model.

The NN approach introduced in this paper can provide numerically efficient solutions to a wide range of problems in environmental numerical models where lengthy, complicated calculations, which describe physical processes, must be repeated frequently. The applications considered here belong to the field of oceanic and wave modeling; however, the method can be applied to efficiently calculate some columnar physical processes in atmospheric models as well. For instance, NNs have been used for fast calculation of atmospheric long-wave radiation by Chavallier et al. (1998, 2000). In environmental numerical models which incorporate chemical and biological components, this method can be applied to efficiently calculate chemical and biological processes as well. Because NN parameterizations can also provide a computationally cheap Jacobian, they will be very beneficial when used in 3-D and especially in 4-D variational data assimilation systems.

1. Introduction

Any atmospheric or oceanic numerical forecast model is based on a set of prognostic and diagnostic differential equations together with additional equations required to obtain a mathematically closed system. Such a system, in principle, can then be solved to predict the evolution of the environment in time if the initial conditions and any required external boundary conditions are prescribed. Even though the forecast problem may now be considered solvable in a theoretical sense, in the real world of running operational forecast models, it is necessary to deal with practical aspects of available computational resources and minimize the computer time taken to produce a forecast by introducing certain simplifications in the system for the following reasons.

The forecast system contains coefficients that appear in the dynamical equations, such as turbulence coefficients representing the unresolvable subgrid scale processes which need to be parameterized in terms of the dependent variables. Also, implicitly contained in the system are processes that deal with model physics such as radiation, convection, etc, which need to be parameterized. Accurate treatments of such parameterizations generally require computations involving complex mathematical expressions which may include differential and integral equations, rules, restrictions, highly nonlinear empirical expressions, etc. that are developed based on physical or statistical models. The complex mathematical formulations of these processes require considerable computational resource.

For example, a spectral atmospheric model with a well developed description of physics and subgrid scale parameterizations may spend up to 70% of calculation time for simulating these processes (Estrade et al-to be published). The long wave radiative code requires > 10% of the computing time in the European Center for Medium-Range Weather Forecast general circulation model (Chevallier 1998) and in the National Centers for Environmental Prediction (NCEP) global model, although the radiative variables in both models are not updated at every time step (in NCEP model they are updated every 3 hours only).

In addition to above types of parameterizations, in ocean models the estimation of the full UNESCO equation of state to compute the sea water density, represented by an empirically derived highly nonlinear equation relating density to pressure, salinity, and temperature, takes a very significant amount (~40%) of the total computational effort. In addition, most forecast models include data assimilation procedures as an integral part of the forecast system to improve the initial conditions of the model. When dealing with ocean models, most often the data assimilation consists of assimilating surface and subsurface temperature observations to correct the model's thermal field. This temperature correction automatically makes it necessary to adjust the salinity field in the ocean model in order to avoid gravitational instabilities in the water column. This requires inverting the complicated oceanic equation of state which makes the computational effort even more time consuming than the forward problem of computing the density itself. Another example where intensive computational is needed in a forecast model is the calculation of the land surface temperature using a set of equations describing the atmospheric boundary layer and physical processes in the soil. Yet another example of intensive computational problem in forecast models is the wind wave forecasting problem in which an exact calculation of the nonlinear wave-wave interactions using the formulation of Hasselmann (1962) takes a prohibitively long time.

In this paper, we use the term **"parameterization***" for convenience to represent in general all the computationally expensive and complex mathematical formulations involved in forecast systems, a few examples of which have been mentioned above.*

In view of the constraints imposed on the available computer resources, calculation time allowed for each parameterization is strictly limited in most operational forecast models. Hence, very often it is found necessary to use simplified forms of these complex representations in carrying out the time integrations in a forecast model, thereby sacrificing accuracy of forecasts to a certain extent. For example, the nonlinear wave interactions in a wave forecast model are replaced by a simplified discreet interaction approximation (DIA) (see Hasselmann et al 1985). Similarly, oversimplified fast parameterizations of physics are used in many parts of atmospheric and oceanic models. In most of these cases, accurate physical models have been developed, but

they cannot be used because they are too expensive computationally. Often simplified parameterizations are obtained, for example, by neglecting higher order terms of perturbation theory, by using empirical approximations, or simply by neglecting the effects which complicate the calculations. It is common in many parameterization schemes that the number of input and output variables is relatively small, whereas the volume of internal calculations is large. A typical example is the parameterization of the radiative fluxes in the atmosphere. Existing physical models usually perform calculations over many narrow spectral bands, making them time consuming. Input for this algorithm includes small number of parameters (vectors) and output - just one vector parameter (temperature change due to radiation absorption). Hence, most often the specific parameterization is a result of a compromise between accuracy and computational efficiency.

It is needless to emphasize the fact that improvements in forecast modeling can be achieved not only by improving the representation of such parameterizations as our understanding of the underlying physical processes increases but also by improving our ability to compute these parameterizations accurately within the constraints imposed by the available computer resources. In this paper we present some of the problems dealing with physical parameterizations and their computations from a different (formal mathematical) point of view, namely that of improving the computational efficiency of available algorithms. We propose a generic approach which is based on developing fast and accurate parameterizations of physics by approximating solutions of exact physical models using neural networks (NNs). From this formal point of view an exact (best known) physical model representing a physical process performs a smooth conversion of an input vector of parameters, $X = \{x_1, x_2, ..., x_n\}$, $X \in U^n$ into an output vector of parameters, $Y = \{y_1, y_2, ..., y_n\}$, $Y \in U^m$. Thus, each output parameter y_i is a continuous function of multiple input variables $x_1, x_2, ..., x_n$ (input vector X). Symbolically this input-output dependence is depicted in Fig. 1.a and can be written as

$$Y = F(X); \qquad X \in U^n, Y \in U^m \tag{1.a}$$

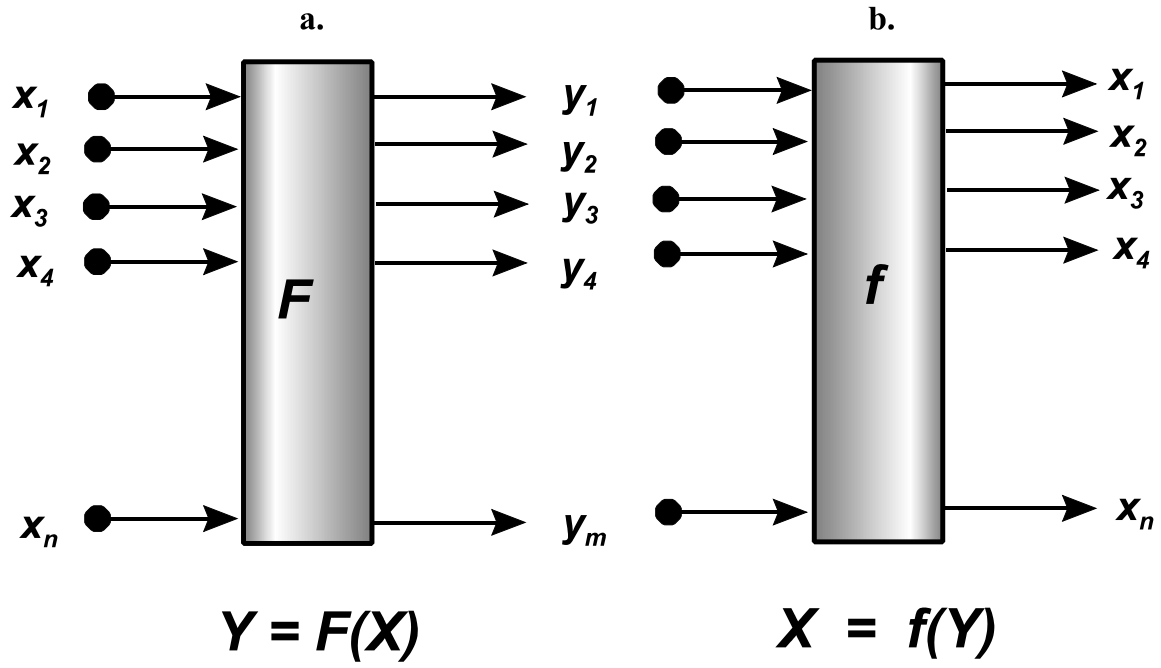

Figure 1. Graphical representation of forward (a) and inverse (b) parameterizations.

If X and Y are related through a cause and effect principle, the forward parameterization, eq. (1.a), can be derived from first principles. If the inverse dependence

$$X = f(Y); \qquad X \in U^n,\ Y \in U^m \tag{1.b}$$

is required (see Fig. 1.b) in a numerical model, the inverse problem should be solved, which implies that eq. (1.a) should be inverted. A solution of the inverse problem (1.b) or an inverse parameterization provides each output parameter x_i as a continuous function of multiple input variables $y_1, y_2, ..., y_n$ (vector Y is an input vector now). Both forward, eq. (1.a), and inverse, eq. (1.b), parameterizations represent the same mathematical object - a continuous mapping which is a continuous relationship between two vectors. Usually these input/output relationships are highly complex, but smooth, for physical processes taken into account in atmospheric, oceanic, and wave models. Hence, if exact solutions to these complex relationships are calculated, however expensive the computational efforts may be, these solutions can be used by the generic mathematical tool - that is, the neural networks - to produce fast and accurate approximations for

continuous mappings. In this approach the costly exact calculation of the physics needs to be performed only once and "off line" to enable the development of the fast and accurate approximation. After that only this fast and accurate approximation will be used to calculate the physics (coefficients of differential equations) "on line" in a numerical model.

In their pioneering works Chevallier et al. (1998, 2000) considered a particular case of the generic problem formulated above – the long-wave atmospheric radiation – and applied neural network technique to solve this specific problem. In this paper we generalize their approach and apply it to solve several problems in oceanic and wave numerical models. In Section 2 of this work we demonstrate that NNs are a generic, fast, and accurate tool for approximating any continuous mapping and, therefore, can be used for the fast and accurate calculation of the parameterizations of physics used in numerical models. In Section 3 we present three NN parameterizations (for oceanic and wind wave models) developed using NNs, and, in Section 4, formulate conclusions.

2. NN - a generic tool for continuous mapping.

The above considerations show that both forward and inverse parameterizations (eqs. 1.a, and 1.b) can be considered as continuous mappings which map a vector of input parameters, $X \in U^n$, to a vector of output parameters, $Y \in U^m$ or vice versa (for the inverse parameterization). These mappings are defined on finite discrete sets of pairs of input/output vectors X and Y, $\{X_i, Y_i\}_{i=1,\ldots,N}$. The development data sets needed to produce accurate NN tool are generally obtained by integrating the exact physical models separately ("off line") even though such a computation may be expensive in terms of computer resources, since this needs to be done only ones.

We assume that the mappings (1.a) and (1.b) are continuous. However, these dependencies can be complicated and nonlinear. Different components y_i of the output vector Y may demonstrate different types of nonlinear dependencies on the input vector X. Moreover, for the same component y_i, the type of nonlinear behavior may be different in different parts of the X domain. When we approximate (1.a) or (1.b), we also usually do not know in advance what kind of nonlinearity to expect; therefore, we need a flexible, self-adjustable approach that can accommodate various types of nonlinear behavior and represent a broad class of nonlinear mappings.

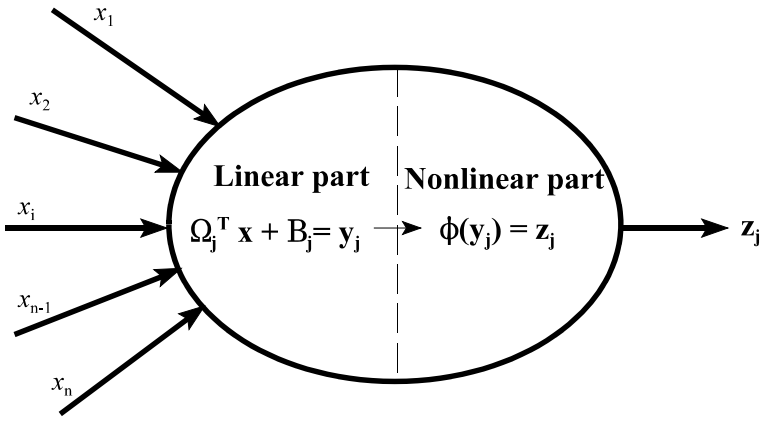

Figure 2. Typical processing element (neuron).

Neural networks (NNs) are well suited for a very broad class of such nonlinear approximations and mappings (Funahashi, 1989). A neural network is a complicated combination of uniform processing elements, nodes, units, or neurons. A typical processing element is shown in Fig.2. Each processing element has usually several inputs (components of vector X) and one output, z_j . The neuron usually consists of two parts, a linear part and a nonlinear part. The linear part calculates the inner product of the input vector X and a weight vector $\mathbf{S}_j$ (which is a column of the weight matrix $\mathbf{S}_{ji}$), and adds a bias, $\%_j$ The result of this linear transformation of the input vector X goes into the nonlinear part of the neuron as the argument of an activation function N. The neuron output, z_j , can be written as,

$$z_j = \phi\left(\sum_{i=1}^{n} \Omega^T_{ji} x_i + \mathrm{B}_j\right) \tag{2}$$

$$\boldsymbol{\Omega} \in \Re^{n\times m};\ \mathbf{B} \in \Re^{m}$$

For the activation (squashing, transition) function N, it is sufficient to be a Tauber-Wiener (nonpolynomial, continuous, bounded) function (Chen and Chen, 1995). The three mostly popular activation functions are sigmoid, hyperbolic tangent, and step function. The sigmoid function can be expressed as:

$$\phi(x) = \frac{1}{1+\exp(-x)};\quad x\in(-\infty,\infty),\ \phi\in(0,1) \tag{3}$$

The neuron is a nonlinear element because its output z_j is a nonlinear function of its inputs X. Neurons can be connected in many different ways into networks with complicated architectures (or topologies).

The most common topology is the multilayer perceptron which is shown in Fig. 3. In a multilayer perceptron, neurons are situated into layers. A multilayer perceptron always has one input layer which receives inputs and distributes them to the neurons in the hidden layer. The

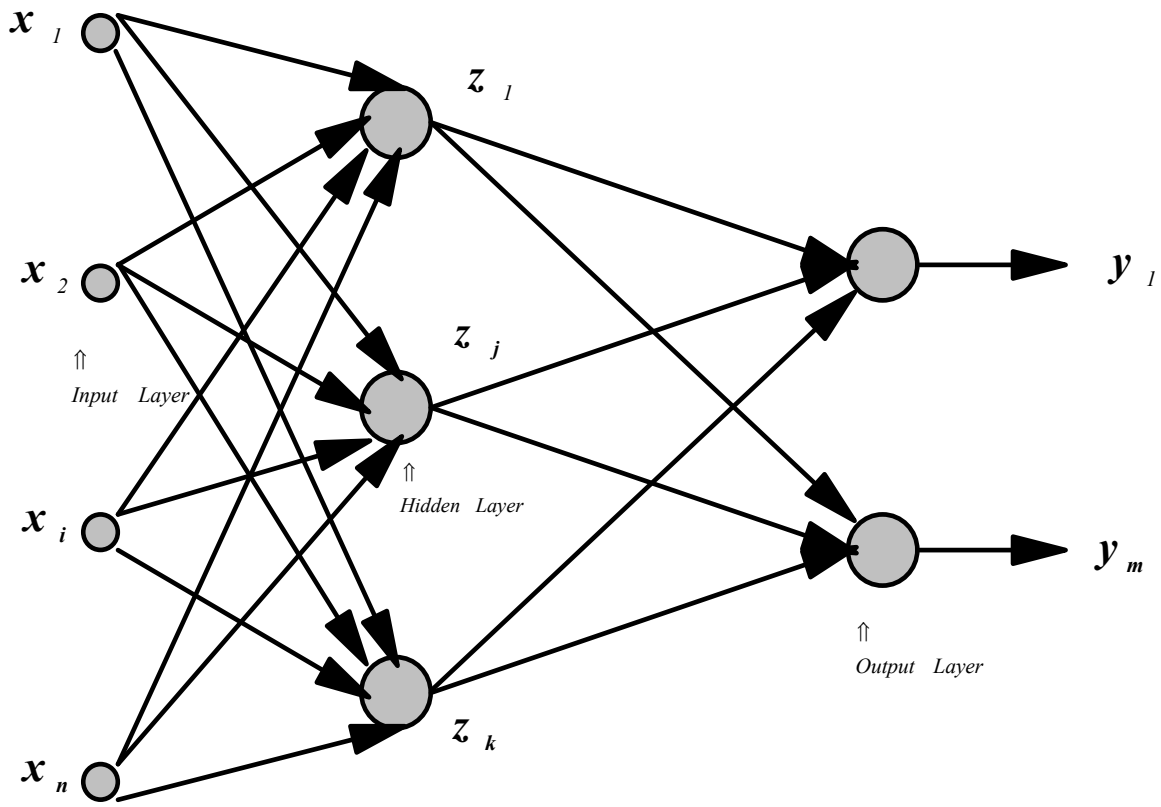

Figure 3. Multilayer perceptron - feed forward, fully connected topology.

neurons in the input layer are linear; they are simple distributors of inputs. The number of input neurons in the input layer is equal to the number of inputs (dimension of input vector X). A multilayer perceptron always has one output layer. The neurons in the output layer may be linear and/or nonlinear, depending on the problem to be solved. The number of output neurons in the output layer is equal to the number of outputs (dimension of output vector Y). A multilayer perceptron always has at least one hidden layer. The neurons in the hidden layer(s) are usually nonlinear. The number of hidden layers, the number of neurons in each hidden layer, and the type of connections between neurons and layers depend on the complexity of the problem to be solved. The topology of the multilayer perceptron shown in Fig. 3 is called feed-forward (there are no feedbacks; the data flow moves only forward) and fully-connected (each neuron in a previous layer is connected to each neuron in the following one). We will consider here only this topology because it is sufficient for solving any continuous mapping problems.

From the discussion above it is clear that the NN generally performs a nonlinear mapping of an input vector $X \in U^n$ (n is dimension of the input vector or the number of inputs) onto an output vector $Y \in U^m$ (m is dimension of the output vector or the number of outputs). Symbolically this mapping can be written as,

$$\boldsymbol{Y} = \boldsymbol{F}_{NN}(\boldsymbol{X}) \tag{4}$$

where $\boldsymbol{F}_{NN}$ denotes this neural network mapping. If we assume for the NN the topology shown in Fig.3, then, using (2), for the NN with k neurons in one hidden layer and activation function, $\Phi(x) = \tanh(x)$, the symbolic expression (4) can be written down explicitly as,

$$\begin{aligned} y_q &= b_q + a_q \phi\left(\sum_{j=1}^{k} \omega_{qj} z_j + \beta_q\right) \\ &= b_q + a_q \phi\left\{\sum_{j=1}^{k} \omega_{qj} \left[\phi\left(\sum_{i=1}^{n} \Omega_{ji} x_i + B_j\right)\right] + \beta_q\right\} \\ &= b_q + a_q \tanh\left\{\sum_{j=1}^{k} \omega_{qj} \left[\tanh\left(\sum_{i=1}^{n} \Omega_{ji} x_i + B_j\right)\right] + \beta_q\right\} \end{aligned} \tag{5}$$

where the matrix Ω_{ji} and the vector θ_j represent weights and biases in the neurons of the hidden layer; ω_{qj} and the β_q represent weights and biases in the neurons of the output layer; and a_q and b_q are scaling parameters. For some applications (e.g., see Section 3.1) we need to know the Jacobian matrix, whose elements are partial derivatives $\partial y_i / \partial x_j$. From (5) these derivatives can be calculated analytically,

$$\frac{\partial y_q}{\partial x_p} = \frac{1}{a_q}\left(a_q^2 + (y_q - b_q)^2\right)\sum_{j=1}^{k} (1 - z_j^2)\Omega_{pj}\omega_{jq} \tag{6}$$

It has been shown by many authors (e.g., Chen and Chen, 1995; Hornik, 1991; Funahashi, 1989; Gybenko, 1989) that a NN with one hidden layer, like NN (5), can approximate any continuous mapping defined on compact sets in U^n. It means that any problem, which can be

mathematically reduced to a nonlinear mapping like (1.a) or (1.b), can be solved using the NN represented by (5). What is the difference between NN solutions given by (5) for different problems? These NNs can have different number of inputs, *n*, and outputs, *m*. They can have different numbers of neurons, *k*, in the hidden layer. They will also have different weights and biases in the hidden and output layers. The next and crucial problem is how to determine all these parameters.

For each particular problem, *n* and *m* are determined by the dimensions of the input and output vectors *X* and *Y*. The number of hidden neurons, *k*, in each particular case should be determined taking into account the complexity of the problem. The more complicated the mapping, the more hidden neurons are required (Attali and Pagès, 1997). Unfortunately, there is no universal recommendation to be given here. Usually *k* is determined by experience and experiment. After these topological parameters are defined, the weights and biases can be found, using a procedure which is called NN training. To explain the training procedure, let us assume that we have a training database which consists of pairs of vectors $\boldsymbol{C_T} = \{\boldsymbol{X_p}, \mathbf{Q}_p\}_{p=1,..N}$, where

$$\boldsymbol{X_p} = \{\boldsymbol{x}_{p1}, \boldsymbol{x}_{p2}, \ldots, \boldsymbol{x}_{pn}\} \in \Re^n, \textbf{\textit{and}} \quad \boldsymbol{\Psi_p} = \{\psi_{p1}, \psi_{p2}, \ldots, \psi_{pm}\} \in \Re^m$$

and an independent matchup data set $\boldsymbol{C_V} = \{X_p, Q_p\}_{p=1,\ldots,M}$ for validating (or testing) the NN after the training is completed. We also assume that vectors X_p, Q_p are related by an unknown continuous mapping $\boldsymbol{F}$,

$$\mathbf{Q}_p = \boldsymbol{F}(\boldsymbol{X_p}), \quad p = 1,..., N \tag{7}$$

and we want to find a NN

$$\boldsymbol{Y_p} = \boldsymbol{F}_{NN}(\boldsymbol{X_p}), \tag{8}$$

$$\boldsymbol{Y_p} = \{\boldsymbol{y}_{p1}, \boldsymbol{y}_{p2}, \ldots, \boldsymbol{y}_{pm}\} \in \Re^m$$

which gives the best (in the sense of some criterion or metric) approximation for mapping $\boldsymbol{F}$.

This criterion may be defined as the minimum (with respect to weights, **S** and ***T***, and biases, **%** and ***$***) of an error or cost function ***E***, which characterizes a difference between mappings (7) and (8),

$$E = \left\| F - F_{NN} \right\|$$

when Euclidian metric is used the error function, ***E***, can be expressed as,

$$E(\Omega, \omega, \mathrm{B}, \beta) = \sum_{p=1}^{N} \left| Y_p - \Psi_p \right|^2 = \sum_{p=1}^{N} \sum_{q=1}^{m} (y_{pq} - \psi_{pq})^2; \qquad (9)$$
$$Y_p = f_{NN}(X_p); (X_p, \Psi_p) \in C_T$$

Thus, optimal values for weights and biases can be obtained by minimizing the error function (9). Therefore, the training of the NN (8) can be reduced to a minimization problem; this problem, however, is a nonlinear minimization problem, which is not an easy problem to solve. A number of methods have been developed for solving this problem (e.g.,Beale and Jackson, 1990). Here we consider a simplified version of the steepest (or gradient) descent method known as the back-propagation training algorithm.

The back-propagation training algorithm is based on the simple idea that searching for a minimum of the error function (9) can be performed step by step, and that on each step we should increment or decrement weights and biases in such a way as to decrease the error function. This can be done using, for example, a simple steepest descent rule as follows:

$$\Delta W = -\eta \frac{\partial E}{\partial W} \qquad (10)$$

where ***0*** is a so-called learning constant and ***W*** is one of the weights, **S** and ***T***, or biases, **%** and ***$***. Using (9) and (5), the derivative in (10) can be expressed through the derivative of the activation function **N**. For example, if ***W*** is a weight $T_{q'j'}$ in the output layer:

$$\frac{\partial E}{\partial \omega_{q'j'}} = 2\sum_{p=1}^{N} (y_{pq'} - \psi_{pq'}) \phi' z_{pj'} \qquad (11)$$

For activation function $N(x) = \tanh(x)$, we have $N' = (1 - N^2)$ and from the first line of (5),

$$N = (y_{pq''} - b_{q''}) / a_{q''}$$

so, finally, the adjustment for a weight $T_{q''j''}$ can be written as:

$$\Delta\omega_{q'j'} = -2\eta\sum_{p=1}^{N}(y_{pq'} - \psi_{pq'})\left(1 - \left(\frac{y_{pq'} - b_{q'}}{a_{q'}}\right)^2\right)z_{pj'} \qquad (12)$$

Adjustments for other weights and biases can be calculated similarly, following the same procedure.

All values on the right-hand side of (12) can be calculated using (5) and the values for weights and biases can be taken from the previous step of the training. Therefore, after r iterations, the simplest rule for calculating new weights and biases is

$$W^{r+1} = W^{r} + \Delta W^{r} \qquad (13)$$

Here we returned to notations used in (10), and $)W^r$ is given by (12) or a similar expression. When $r = 0$, the initialization problem familiar to people who use various kinds of iteration schemas arises: how can we calculate the right-hand side in (13) and (12) at the first step when we do not have weights and biases from a previous step of training. Many publications have been devoted to this problem (e.g., Nguyen and Widrow, 1990; Wessels and Bernard, 1992). A random initialization is usually used.

The simple version of the back-propagation training algorithm described here may be modified and improved in many different ways (Beale and Jackson, 1990; Chen, 1996); however, the above discussion introduces the main ideas of this method. Usually the training process is terminated after some maximum number of adjustment steps, which is a given parameter of the training procedure, or after the error function becomes less than some value, which is also a given parameter of the training procedure.

Here, in the conclusion of this section, several main properties of NNs are presented which make them a very suitable generic tool for nonlinear mapping (and, therefore, for fast parameterization of physics). Some of these properties have been illustrated above, others are described in the literature.

- NNs are able to accurately approximate complicated nonlinear input/output relationships (any continuous nonlinear mapping).
- While training the NN is often time consuming, its application is not. After the training is finished (it is usually performed only once), each application of the trained NN is an estimation of (5) with known weights and biases which is practically instantaneous (several tens of floating point additions and multiplications).
- NNs are analytically differentiable.
- NNs are well-suited for parallel processing (Cheng, 1996) (all neurons in the same layer are completely independent and can be evaluated simultaneously).

3. Oceanic and Wave Applications

3.1 Application of neural networks for efficient calculation of sea water density or salinity from the UNESCO equation of state

Here we apply a NN technique to two related problems of fast calculation of physics in oceanic modeling and data assimilation. (i) In most ocean models, the UNESCO International Equation of State for Seawater (e.g., UNESCO, 1981) (UES) is used for the calculation of the density at each point of a 3-D grid using a relatively small time step. The frequency of updating the density depends on specifics of the model. For example, if the model explicitly describes external waves, the time step for recalculating the density should not be more than several times larger than the global time step. Hence, for high-resolution models, the solution of this equation consumes a significant part of the overall computation time. (ii) In the data assimilation process, assimilation of temperature only in ocean models which employ the full equation of state, without making corresponding adjustments to salinity can lead to problems of gravitational instabilities (Woodgate 1998, Chalikov et al. 1998). To adjust the salinity, we need to calculate the salinity from UES as a function of temperature, density and depth (or pressure), i.e. solve an inverse problem in many points. Numerical inversion of the UES is an iterative procedure which can consume several orders of magnitude more time than solving of the UES itself.

The UES for sea water gives the following expression for the density anomaly σ_D (kg/m^3) as described by Foffonoff and Millard (1983),

$$\delta_\rho(T,S,P)=\rho(T,S,P)-1000$$
$$\rho(T,S,P)=\frac{\rho(T,S,0)}{1-\frac{P}{K(T,S,P)}} \qquad (14)$$

where D is the density of seawater in kg/m^3, T is the temperature in °C, S is the salinity in psu, P is the pressure, and $K(T,S,P)$ is a bulk modulus.

The UES (14) is empirically based and given over a three-dimensional domain for D = { -2 < T < 40°C, 0 < S < 40 psu, and 0 < P < 10000 decibars}. This domain represents all possible

combinations of *T, S,* and *P* which are encountered globally. Mathematically, the functions $\rho(T,S,0)$ and $K(T,S,P)$ are represented by multidimensional polynomials and, as a result, the density (14) is a ratio of two, three-dimensional polynomials which contain more than 40 parameters.

The UES has several drawbacks when it is applied in the context of ocean modelling. First is its cumbersome form. In most ocean models, the UES, if it is used for calculating the density, is estimated at each point of a three-dimensional grid for each time step. For high-resolution models, the solution of this equation consumes a significant part (up to 40%) of the overall computation time. Second, in spite of the fact that this equation is accurate and smooth enough for density itself, its first derivative is locally non-monotonic because eq. (14) is constructed using high order polynomials and because it has poles (zeros of the polynomial in denominator). Third, it is not a simple matter using the UES, to obtain solutions for salinity since this solution represents an inverse dependence.

The UES determines the density field from observed temperature, salinity, and pressure to within a standard error of approximately 0.009 kg m^{-3}; however, due to variations in the composition of dissolved salts, the uncertainty in the density of natural seawater is of the order of 0.05 kg m^{-3} (Apel, 1987). The UES is usually applied in numerical models in combination with an approximate hydrostatic pressure - depth relationship of the following form:

$$P = \rho(T,S,0)\, g\, Z \qquad (15)$$

where *g* is the acceleration of gravity. This linear approximation neglects the dependence of *g* on latitude and the nonlinear terms in the dependence of *Z* on *P* (Foffonoff and Millard, 1983) which introduce additional uncertainties in the calculation of $\rho(T,S,Z)$ of about 0.05 - 0.1 kg m^{-3}.

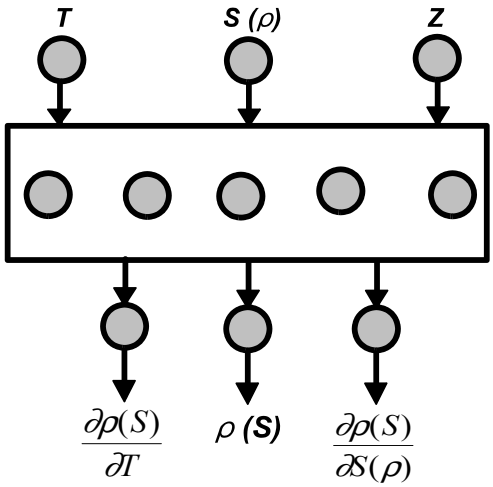

Figure 4. Schematic representation of NN parameterizations for the density and salinity of sea water.

Taking these uncertainties into account, the accuracy of the NN parameterization for density is expected to be # 0.1 kg m^{-3}, and the accuracy of the NN parameterization for salinity expressed in terms of density to be # 0.1 kg m^{-3}.

The UES defines two relationships (second relationship for salinity through inversion),

$$D = D(T,S,Z) \tag{16.a}$$

$$S = S(T,D,Z) \tag{16.b}$$

which are continuous mappings (degenerated mappings because one dimensional vectors are on the left). As we learned from the previous sections the NN technique can be applied to approximate these mappings (see also Krasnopolsky et al 2000a). To create a training set for these NN parameterizations, in the three-dimensional domain D (see above), 4,000 points (T_i,S_i,Z_i) were generated. The UES was used to estimate the density of seawater, D_i, for each point. This simulated data set $\{D_i;T_i, S_i, Z_i\}$ was used as a proxy for experimental data in order to train the NNs to extract density and salinity. Two expressions were obtained (see also Fig. 4):

$$D = D_{NN}(T,S,Z) \tag{17.a}$$

$$S = S_{NN}(T,D,Z) \qquad (17.b)$$

where both D_{NN} and S_{NN} are expressed by eq. (5).

To evaluate the accuracy of the NN approximation (17), 16,000 points were generated within the domain D. The expression (14) for the UES, and the NN which was trained to extract density, $D_{NN}(T,S,Z)$ (17.a) were used to obtain separate estimates for the density of sea water, and then the results obtained from each were compared. These comparisons are presented in Tables 1 and 2 below. Table 1 shows statistics for the density anomaly, $\sigma = D(T,S,Z) - 1000$, for seawater, generated by the UES, and by the NN (17.a). Table 2 shows several statistical measures of the differences between the UES and the NN estimates for density. In terms of the bias and the RMS differences, the NN results for density clearly satisfy the criterion mentioned above; both the bias and the RMS values do not exceed the uncertainties indicated there, and are less than 0.1 kg m^{-3}.

Table 1. Statistics for the density anomaly $\sigma = D(T,S,Z) - 1000$. Minimum, maximum, and mean values together with the standard deviations F_σ are shown (kg m^{-3}).

Expression	Min σ	Max σ	Mean σ	F_σ
UES (14)	-2.26	56.58	27.16	11.02
NN (17.a)	-1.80	55.95	27.16	11.02

Table 2. Minimum, maximum, and mean differences (i.e., the biases), $\gamma = D_{UES} - D_{NN}$, and the RMS differences, all expressed in kg m^{-3}.

Min γ	Max γ	Bias	RMS
-0.62	0.63	0.00	0.06

To further evaluate the accuracy of the NN-derived densities, the partial derivatives of density with respect to temperature, salinity, and depth were calculated, recognizing the fact that any

errors in the NN approximation will be amplified in the process of taking first differences. To estimate the deviations of the NN-obtained estimates from those obtained using the UES, the partial derivatives of density were calculated using UES (14) and the NN (17.a) representation (see eq. (6) in the Section 2) for the same 16,000 points. They were then compared as before. Table 3 summarizes the results which have been normalized by dividing the biases and RMS differences by the absolute mean values of the corresponding derivatives. The results show that the NN representation (17.a) produces derivatives of density which are similar to those obtained from the UES (14). The largest differences occur with respect to temperature but even in this case they do not exceed 5%.

Table 3. Differences for the normalized biases and RMS values for the derivatives of seawater density with respect to temperature, salinity, and depth.

Derivatives	Normalized Bias	Normalized RMS Differences
$\partial D/\partial T$	0.008	0.048
$\partial D/\partial S$	0.001	0.016
$\partial D/\partial Z$	0.000	0.001

To evaluate the errors in using the NN approach to estimate the salinity, we used the same 16,000 points (D_i, T_i, S_i, Z_i) which were used for estimating the density. Initially, the NN for S_{NN} (17.b) was applied to calculate a new salinity, s_i, using the corresponding values (T_i, D_i, Z_i). Then the differences (S_i - s_i) were utilized to estimate the accuracy of the NN-derived salinities (first line in Table 4). To further evaluate the quality of the NN-derived salinities, the UES was applied again, this time to the triad (T_i, s_i, Z_i) to recalculate the density of seawater, $D\%_i$. If the NN-obtained values for salinity were perfect, then the density, $D\%_i$, would be equal to D_i. The differences between these two values, (D_i - $D\%_i$), were then used to further estimate the accuracy of the salinity-trained NN in terms of the density (second line in Table 4).

Table 4. Accuracies of the salinities estimated by the NN in terms of salinity and density. Minimum, maximum, and mean errors together with the RMS errors are presented.

Units	Min error	Max error	Mean error	RMS error
psu	-0.33	0.85	0.00	0.10
Kg m^{-3}	-0.27	0.71	0.00	0.08

Table 4 shows that the NN estimates of salinity (17.b) have an RMS error of 0.1 psu. In terms of the related error in density, this accuracy corresponds to an RMS error of 0.08 kg m^{-3}, which again does not exceed the uncertainties discussed above.

A substantial additional acceleration of calculations may be achieved by use of differential increments of density, temperature, and salinity. Hence, we extend our approach to estimate these quantities also. Additionally, an efficient way of substantially reducing the computational burden is to replace the calculations of density *per se* by calculations of its total differential

$$\Delta\rho = \frac{\partial\rho}{\partial \boldsymbol{T}}\Delta \boldsymbol{T} + \frac{\partial\rho}{\partial \boldsymbol{S}}\Delta \boldsymbol{S} \tag{18}$$

where ^{a}T and ^{a}S are increments of T and S, MD/MT, and MD /MS are functions of T, S, and z, where z is usually constant. Because the increments ^{a}T and ^{a}S are usually small, accuracy requirements in this case are not stringent. As a result, these terms can be represented at fixed depths by low-order NN approximation. As a precaution, however, it may be advisable to recalculate the densities and its derivatives periodically, using the exact form or NN approximation of the UES, to update the estimated values obtained using (18).

As we already pointed out, the derivatives MD/MT and MD /MS can be accurately calculated from the NN D_{NN}, eq. (17.a), using eq. (6) (see also Fig. 4). Equation (18) can be reduced to

$$\Delta\rho = \frac{\partial\rho_{NN}}{\partial T}\Delta T + \frac{\partial\rho_{NN}}{\partial S}\Delta S \tag{19}$$

This use of NN and its derivatives has been shown to accelerate model performance significantly. Statistics which compare this NN with the derivatives calculated using the UES are presented in Table 4. Table 5 estimates an absolute accuracy in calculating ^{a}D from equation (19), as compared with ^{a}D calculated using UES with the same ^{a}T and ^{a}S. In this case bias is negligible and the RMSE < 0.5%. Fig. 5 shows that using eq. (19) the calculations may be accelerated

Table 5. Mean, standard deviation (F), and mean differences (i.e., biases) and the standard deviations (SD) for the differences for ^{a}D calculated using eq. (19) with $^{a}T = 0.1$ and $^{a}S = 0.1$

	Mean	**F**	**Bias**	**SD**
^{a}D	0.052	0.012	-0.00001	0.0002

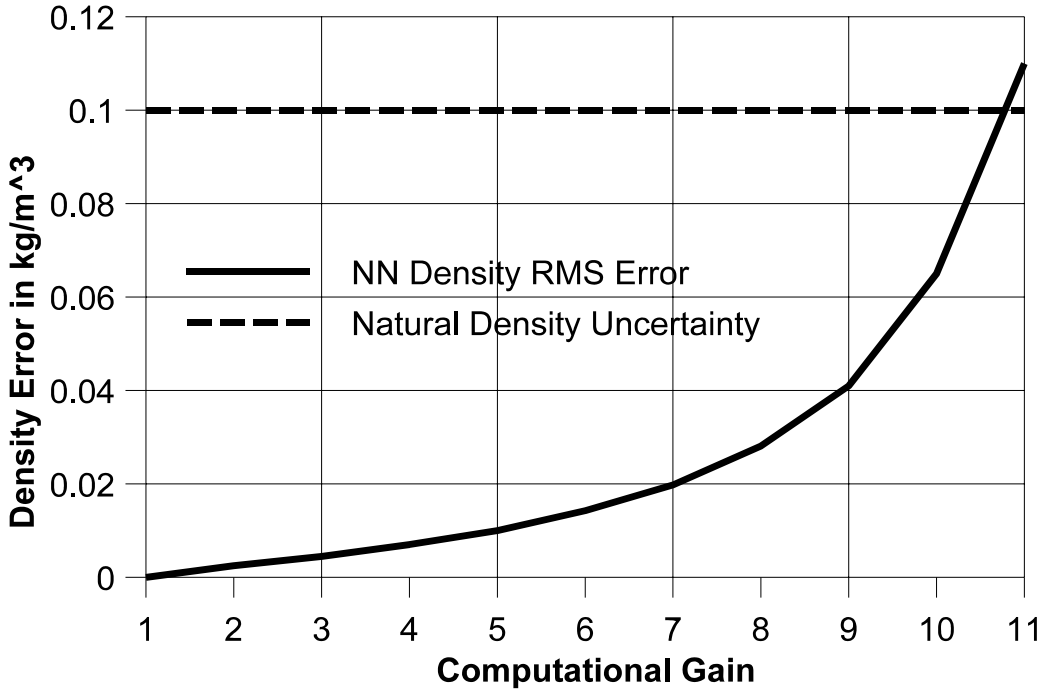

Figure 5. Error in the density as a function of computational gain when eq. (19) is used for density calculations.

about 10 times with the error in the density calculations not exceeding the natural uncertainty 0.1 kg/m^3.

Finally, we accepted the following scheme for calculating the density. At the beginning of each run of the model, the density and its derivatives are calculated using the NN approximation. Then for each time step assigned for updating the density, the density increment values are calculated using Eq. (19), with the initial density and its derivatives kept unchanged. The frequency of updating (recalculating) the density and its derivatives depends on specific properties of the model and region to which the model is applied. For the global version of the NCEP ocean model with the horizontal resolution of about 80 km the density and its derivatives are updated once per 90 time integration steps of the external mode (i.e., every 180 min). This updating period may probably be enlarged, but the current 90 time steps period already decreases expenses for density calculations about 10 times.

3.2 A Neural Network Approximation for Nonlinear Interactions in Wind Wave Models.

Ocean wind wave modeling for hindcast and forecast purposes has been at the center of interest for many decades. Numerical prediction models are generally based on a form of the spectral energy or action balance equation

$$\frac{DF}{Dt} = S_{in} + S_{nl} + S_{ds} + \cdots, \tag{20}$$

where F is the spectrum, S_{in} is the input source term, S_{nl} is the nonlinear interaction source term, S_{ds} is the dissipation or 'whitecapping' source term, and **…** represents additional (shallow water) source terms. The JONSWAP study (Hasselman et al 1973) identified the active role of the nonlinear interactions in wave growth. The SWAMP study (SWAMP Group 1985) then identified the need for explicit modeling of S_{nl} in wave models. State-of-the-art or so-called third generation wave models therefore explicitly model this source term.

In its full form (e.g., Hasselmann and Hasselmann 1985), the calculation of the interactions S_{nl} requires the integration of a six-dimensional Bolzmann integral:

$$\begin{aligned} &S_{nl}(\vec{k}_4) = T \otimes F(\vec{k}) = \\ &= \omega_4 \int G(\vec{k}_1, \vec{k}_2, \vec{k}_3, \vec{k}_4) \cdot \delta(\vec{k}_1 + \vec{k}_2 - \vec{k}_3 - \vec{k}_4) \cdot \delta(\omega_1 + \omega_2 - \omega_3 - \omega_4) \\ &\times [n_1 \cdot n_3 \cdot (n_4 - n_2) + n_2 \cdot n_4 \cdot (n_3 - n_1)] d\vec{k}_1 \, d\vec{k}_2 \, d\vec{k}_3 \\ &n(\vec{k}) = \frac{F(\vec{k})}{\omega}; \qquad \omega^2 = g \cdot k \cdot \tanh(kh) \end{aligned} \tag{21}$$

with the complicated coupling coefficient G which contains moving singularities (K. Hasselmann 1963). This integration requires roughly 10^3 to 10^4 times more computational effort than all other aspects of the wave model combined. Present operational constraints require that the computational effort for the estimation of S_{nl} should be of the same order of magnitude as the remainder of the wave model. This requirement was met with the development of the Discrete Interaction Approximation (DIA, Hasselmann et al 1985). The development of the DIA allowed

for the successful development of the first third-generation wave model WAM (WAMDI Group 1988, Komen et al. 1994). More than a decade of experience with the WAM model and its derivatives have identified shortcomings of the DIA. The DIA tends to unrealistically increase the directional width of spectra, has a systematic spurious impact on the shape of the spectrum near the spectral peak frequency, and has a much too strong signature at high frequencies. In present third generation wave models, these deficiencies can be countered at least in part by the dissipation source term S_{ds}, which is generally used for tuning energy balance in the equation (20). Although this approach gives good results, it is counterproductive, because it prohibits development of dissipation source terms based on solid physical considerations. With our increased understanding in the physics of wave generation and dissipation, this becomes an even bigger obstacle for the further development of third-generation wave models.

Considering the above, it is of crucial importance for the development of third generation wave models to develop *an economical yet accurate approximation* for S_{nl}. Here, we explore a Neural Network Interaction Approximation (NNIA) to achieve this goal (see also Krasnopolsky et al 2000b). NNs can be applied here because the nonlinear interaction (21) is essentially a nonlinear mapping (symbolically represented in eq. (21) by T) which relates two vectors (2-D fields in this case). Thus, the nonlinear interaction source term can be considered as a nonlinear mapping between a continuous source term S_{nl} and a continuous spectrum F

$$S_{nl} = T(F) \ , \tag{22}$$

where T is the exact nonlinear operator given by the full Bolzmann interaction integral (21) (Hasselmann and Hasselmann 1985, Resio and Perrie 1991). Discretization of S and F (as is necessary in any numerical approach) reduces (22) to continuous mapping of two vectors of finite dimensions. Modern high resolution wind wave models use descretization on a two dimensional grid which leads to dimensions of S and F vectors of order of $N > 600$ ((Tolman 1999). It seems unreasonable to develop a NN approximation of such a high dimensionality (more than 600 inputs and outputs). Moreover, such a NN will be grid dependent.

In order to reduce dimensionality of the NN and convert the mapping (22) to a continuous mapping of two finite vectors independent on the actual spectral discretization, the spectrum F and source function S_{nl} are expanded using systems of two-dimensional functions each of which (M_i and Q_q) creates a complete and orthogonal two-dimensional basis

$$F \approx \sum_{i=1}^{n} x_i \Phi_i, \quad S_{nl} \approx \sum_{q=1}^{m} y_q \Psi_q, \tag{23}$$

where for x_i and y_q we have

$$x_i = \iint F \Phi_i, \quad y_q = \iint S_{nl} \Psi_q \ , \tag{24}$$

where the double integral identifies integration over the spectral space. Because both sets of basis functions $\{M_i\}_{i=1,\ldots,n}$ and $\{Q_q\}_{q=1,\ldots,m}$ are complete, increasing n and m in (23) improves the accuracy of approximation, and any spectrum F and source function S_{nl} can be approximated by (23) with a required accuracy. Substituting (23) into Eq. (22) we can get

$$\boldsymbol{Y} = T(\boldsymbol{X}) \ , \tag{25}$$

which represents a continuous mapping of the finite vectors $X \in U^n$ and $Y \in U^m$, and where T still represents the full nonlinear interaction operator. As described in the previous section, this operator can be approximated with a NN with n inputs and m outputs and k neurons in the hidden layer

$$\boldsymbol{Y} = T_{NN}(\boldsymbol{X}) \ . \tag{26}$$

The accuracy of this approximation (T_{NN}) is determined by k, and can generally be improved by increasing k (see Section 2 above).

To train the NN approximation T_{NN} of T, a training set has to be created which consists of pairs of vectors $\boldsymbol{X}$ and $\boldsymbol{Y}$. To create this training set, a representative set of spectra F_p has to be generated with corresponding (exact) interactions $S_{nl,p}$ using eq. (21). For each pair $(F, S_{nl})_p$, the

corresponding vectors $(\boldsymbol{X},\boldsymbol{Y})_p$ are determined using eq. (24). These pairs of vectors are then used to train the NN to obtain T_{NN}.

After T_{NN} has been obtained by training, the resulting NN Interaction Approximation (NNIA) algorithm consists of three steps :

- Decompose the input spectrum, *F,* by applying Eq. (24) to calculate $\boldsymbol{X}$.
- Estimate $\boldsymbol{Y}$ from $\boldsymbol{X}$ using Eq. (26).
- Compose the output source function, S_{nl}, from $\boldsymbol{Y}$ using Eq. (23).

A graphical representation of the NNIA algorithm is shown in Fig. 6.

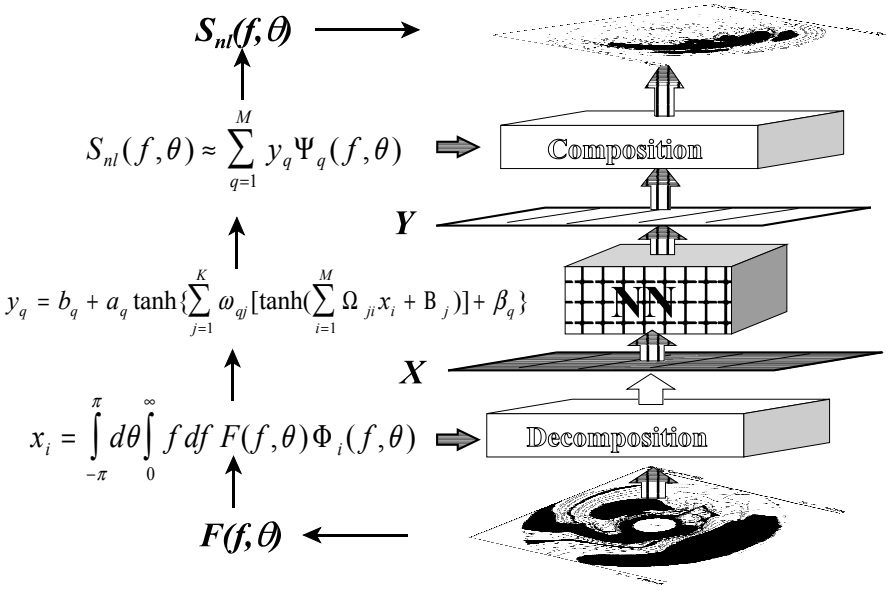

Figure 6. Graphical representation of the NNIA algorithm.

The above describes the general procedure for developing an NNIA. Development of an actual NNIA requires the following steps:

< Select basis functions M_i and Q_q and the number of each *(n,m)*.

< Design a NN topology (number of neurons *k*).

< Construct a representative training set.

< Select training strategies.

The first three points all have a significant impact on both accuracy and economy of a NNIA. Unfortunately, there is no pre-defined way to tackle these issues as mentioned in section 2. It is therefore unavoidable that the development of a NNIA involves many iterations. The first requirement of an NNIA to be potentially useful in operational wave modeling, is that the exact interactions S_{nl} are closely reproduced for computational costs comparable to that of the DIA. The following shows the potential of this approach with the design of a simple ad-hoc NNIA.

To address the basic feasibility of a NNIA, we have considered an NNIA to estimate the nonlinear interactions $S_{nl}(f,2)$ as a function of frequency f and direction 2 from the corresponding spectrum $F(f,2)$. We first also consider deep water only. To train and test this NNIA, we used a set of about 20,000 simulated realistic spectra for $F(f,2)$, and the corresponding exact estimates of $S_{nl}(f,2)$ (Van Vledder et al 2000). Simulation has been performed using a generator that calculated a spectral function as a composition of several Pierson-Moskowitz (1964) spectra for different peak frequencies each peak oriented randomly in [0,2B] interval. Comparison of simulated spectra with spectra simulated by the WAVEWATCH model (Tolman 1999, Tolman and Chalikov 1996) shows that this approach allowed us to simulate reasonably realistic and complicated spectra describing a broad range of wave systems. Spectra with four peaks were used in calculations below. Separate data sets have been generated for training and validation.

As is common in parametric spectral descriptions, we choose separable basis functions where frequency and angular dependence are separated. For M_i this implies:

$$\Phi_i(f,\theta) \Rightarrow \Phi_{ij} = \phi_{f,i}(f)\phi_{\theta,j}(\theta) \qquad (27)$$

A similar separation is used for Q_q. Considering the strongly suppressed behavior of F and S_{nl} for f 6 0, and the exponentially decreasing asymptotic for f 6 4, generalized Laguerre's

polynomials (Abramowitz and Stegun 1964) are used to define N_f and R_f. Considering that no directional preferences exist in F and S_{nl}, a Fourier decomposition is used for N_2 and R_2. The number of base functions is chosen to be $n = 51$ and $m = 64$ to keep the accuracy of approximation for F on average better than 2% and for S_{nl} - better than 5-6%. The number of hidden neurons was taken $k = 30$ which allows a satisfactory approximation (26) for the mapping (25).

Table 6. RMSE statistics for 10,000 S_{nl}

	Mean RMSE	F_{RMSE}	Max RMSE
DIA	0.0133	0.0111	0.104
NNIA	0.0068	0.0063	0.065

Table 6 compares three important statistics for the source function RMS errors (with respect to exact solution) calculated using DIA and NNIA for 10,000 spectra (independent validation set). The NNIA improves the accuracy about twice as compared with DIA.

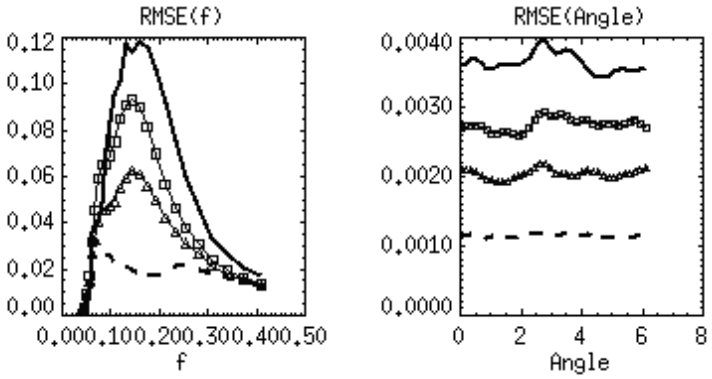

Figure 7. RMSE as functions of frequency f and angle (averaged over entire test set). Dashed line – error of approximation (lower bound for all other errors). Solid line – DIA, line with squares – NNIA (51:20:64), and line with triangles – NNIA (51:30:64)

Figure 7 shows mean RMSE as function of the frequency f (left) and the angle 2 (right). It also illustrates the improvement of the NNIA accuracy with the increasing of the number of neurons,

k, in the hidden layer from 20 to 30. Numbers in Table 6 correspond to a NNIA with 30 neurons in the hidden layer (51:30:64). Fig. 8 shows 3 pairs (one row in the figure corresponds to one pair) of one dimensional, integrated over *2,* source functions S_{nl} *(f)* (left column) and one dimensional, integrated over *f*, source functions S_{nl}(*2*) (right column) from the validation data set. Thick solid curves correspond to the exact S_{nl}. Dashed curves correspond to DAI of S_{nl}. Curves with triangles correspond to the NNIA estimate of S_{nl}. Numbers inside the panels show DIA and NNIA errors in percents with respect to exact solution.

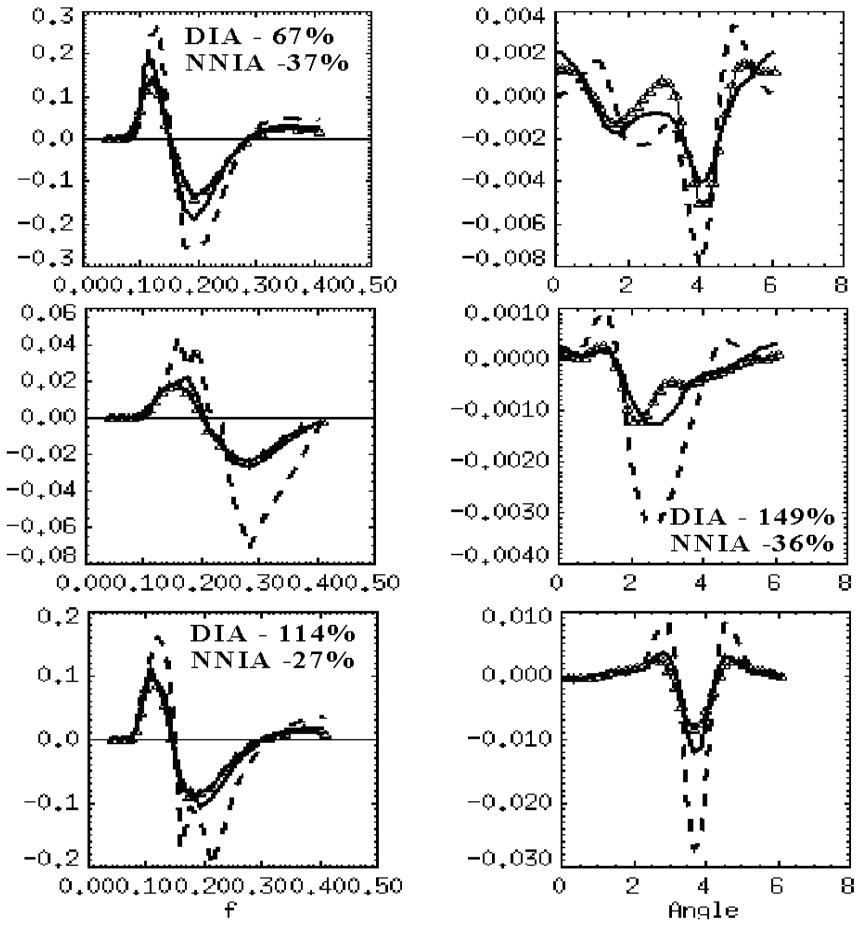

Figure 8. See explanations in the text.

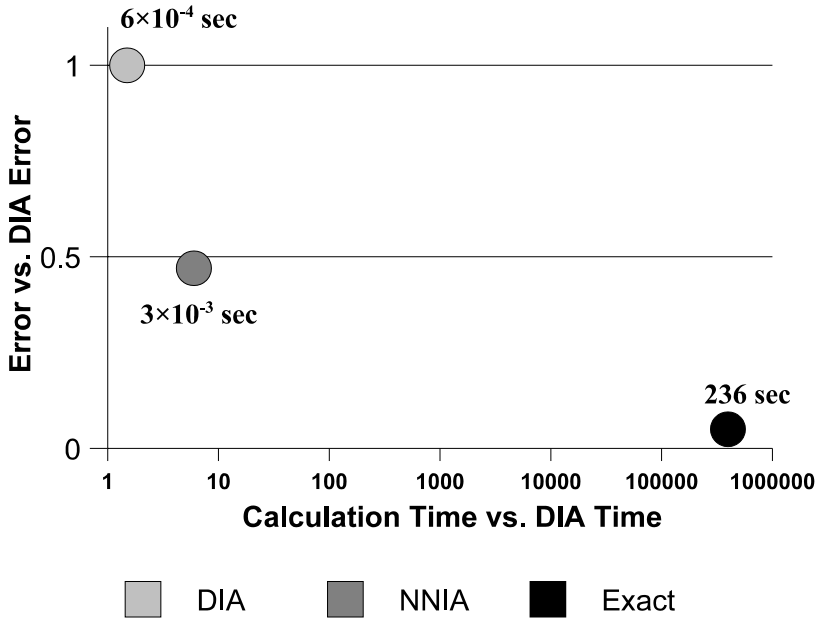

Figure 9. Comparison of the accuracy and computational efficiency of the DIA, NNIA , and exact algorithms. The horizontal time scale is logarithmic.

The results in Fig. 8 are fairly representative for the validation data set. In general, the NNIA reproduces the exact S_{nl} accurately. Even if though spurious oscillations are present in the DIA spectrum (e.g., dashed lines in middle and lower panel on the left), the NNIA shows no such behavior, and gives reasonable results. In general, many DIA source functions exhibit complicated behavior and spurious oscillations. Major peaks in these functions coexist with more or less random small-scale fluctuations. These fluctuations are probably an artifact produced by a simplified nature of DIA. Exact interactions are the result of averaging over much lager number of resonant sets of wave numbers, and are therefore much smoother than the results of the DIA.

And finally, Fig. 9 compares the DIA, NNIA, and exact algorithms in terms of the accuracy and computational efficiency. Computational time (in sec) corresponds to a control calculation performed on the same computer. The current preliminary version of the NNIA algorithm is twice more accurate and only about 5 times slower than the DIA algorithm. In current version of

the wind wave models an algorithm which is up to 20 times slower than DIA can be accommodated; therefore, we still have enough room for further improvement of the NNIA accuracy. Considering that no optimization has yet been applied in the development of the NNIA, in the composition and decomposition procedures, it appears reasonable to expect a final NNIA algorithm with similar computation requirements as the DIA and significantly better accuracy.

4. Conclusions

In this paper we formulated a new approach for simplifying and accelerating time consuming calculations in environmental numerical models using neural networks technique. Parameterization of physical, chemical, biological, etc. processes which occur at different scales constitute an important class of such calculations . It is shown that, from mathematical point of view, descriptions of such processes can usually be considered as continuous mappings (continuous dependencies between two vectors). It is also shown that neural networks are a generic tool for approximation of such mappings and, therefore, can be used for fast and accurate approximation of parameterizations of such processes. In addition to fast and accurate approximation to the primary parameterization, NN also provides the entire Jacobian for very little computation cost. Because NN Jacobian is computationally cheap , this approach is expected to be very beneficial when used in 3-D and especially in 4-D variational data assimilation systems. We also illustrated this approach applying it to three specific problems. These applications belong to the field of oceanic and wave modeling; however, such an approach has been be applied to parameterization of physical processes (e.g., long-wave radiation, Chevallier et al., 1998, 2000) in atmospheric models as well.

The first application considered in the paper deals with the oceanic equation of state, which is used for estimating the density and salinity of sea water in ocean circulation models. Separate neural networks for density and salinity were developed using the UES as a basis. Although the estimation of density represents a forward problem, estimating salinity from the UES represents a complicated inverse problem which has been very efficiently solved using the NN approach. The accuracy of the neural network-generated densities and salinities were of the same order as those obtained directly from the UES itself. However, the time required to perform the calculations of density using the neural network is several times less than that for UES. The time required for calculating salinity using the neural network is several hundred times less than that required for the numerical inversion of the USE. Consequently, this approach has direct application to numerical ocean models where the equation of state must be estimated repeatedly. At NCEP, a NN equation for sea water density is currently used in the NCEP oceanic model.

The second application deals with the nonlinear wave-wave interactions in wind wave models. A prototype of the NN approximation for this interaction is presented in this work. The NNIA calculations improve the accuracy of DIA S_{nl} calculations by a factor of two and require roughly 4-5 times more computational effort than the DIA calculations with less than 5% of this time spent in the actual NN part of the algorithm [i.e., Eq. (7)]. Decomposition of the input spectra F and composing the source function S_{nl} from the NN output accounts for the rest. Having established that a NNIA has the potential of being both accurate and efficient, we intend to take the following steps towards developing a NNIA for application in operational wave models: (1) Use more realistic spectra and S_{nl} calculated from them for training; (2) Optimize the NNIA by successive integration of physical properties in the base functions, normalizing F and S_{nl}, optimizing the number of base functions and network topology, and optimizing numerical aspects of the decomposition / compositions algorithms; (3) Expand the NNIA to arbitrary water depths, either by expanding the underlying NN or by scaling as in the DIA.

Acknowledgments

We thank D.B. Rao for his thorough and critical review of this paper and useful comments. We thank Gerbrant Ph. Van Vledder for providing us with a code for calculating the exact nonlinear wave-wave interaction. We also thank ONR for supporting NNIA research project.

References

Abramowitz, M. and I. A. Stegun, Editors, 1964: *Handbook of Mathematical Functions with Formulas, Graphs and Mathematical Tables*. National Bureau of Standards

Apel, J.R., "Principles of Ocean Physics", Academic Press, New York, 1987, p. 145

Attali J-G. and G. Pagès, 1997: Approximations of Functions by a Multilayer Perceptron: A New Approach., *Neural Networks*, Vol. 6, pp. 1069-81

Beale, R. and T. Jackson, 1990: *Neural Computing: An Introduction,* Adam Hilger, Bristol, Philadelphia and New York, 240 pp.

Chalikov D., L. C. Breaker, V. Krasnopolsky, and D. B. Rao. "Revisiting the Question of Assimilating Temperature alone into a Full Equation of State Ocean Model", *Ocean Modeling*, Issue 116, p. 13-14, 1998

Chen, C.H. (Editor in Chief), 1996: *Fuzzy Logic and Neural Network Handbook*, McGraw-Hill, New York

Chen, T., and H. Chen, 1995a: Approximation Capability to Functions of Several Variables, Nonlinear Functionals and Operators by Radial Basis Function Neural Networks, *Neural Networks*, **6**, pp. 904-910,

-----, and -----, 1995b: Universal Approximation to Nonlinear Operators by Neural Networks with Arbitrary Activation Function and Its Application to Dynamical Systems*, Neural Networks*, **6**, pp. 911-917

Chevallier,F., F. Chéruy, N.A. Scott, and A. Chédin, 1998: "A neural network approach for a fast and accurate computation of longwave radiative budget.", *J. Appl. Meteor*., **37**, 1385-1397

Chevallier, F., J.-J. Morcrette, F. Chéruy, and N.A. Scott, 2000: "Use of a neural-network-based longwave radiative transfer scheme in the EMCWF atmospheric model.", *Quat. J. Roy. Meteor.*, **126**, 761-776

Estrade, J.-F., Y. Trémolet, and J. Sela, 2001: Experiments with NCEP Spectral Model, to be published

Fofonoff, N.P., and R.C. Millard Jr., "Algorithms for computation of fundamental properties of seawater", UNESCO technical paper in marine science 44, UNESCO, 1983

Funahashi, K., 1989: On the Approximate Realization of Continuous Mappings by Neural Networks, *Neural Networks*, **2**, pp. 183-192

Gybenko, G., 1989: Approximation by Superposition of Sigmoidal Functions, in *Mathematics of Control, Signals and Systems*, **2**, No. 4, pp. 303-314

Hasselmann, K., 1963: On the non-linear energy transfer in a gravity-wave spectrum. Part 3: Evaluation of the energy flux and swell-sea interaction for a Neumann spectrum. *Journal of Fluid Mechanics*, **15**, pp. 385-399

Hasselmann, K., T.P. Barnett, E. Bouws, H. Carlson, D.E. Cartwright, K. Enke, J.A. Ewing, H. Gienapp, D.E. Hasselmann, P. Kruseman, A. Meerburg, P. Müller, D.J. Olbers, K. Richter, W. Sell and H. Walden, 1973: Measurements of wind-wave growth and swell decay during the Joint North Sea Wave Project (JONSWAP). *Ergänzungshelf zur Deutschen Hydrographischen Zeitschrift,* Reihe A (8) Nr. 12, 95 pp.

Hasselmann, S, and K. Hasselmann, 1985: Computations and parametrizations of the nonlinear energy transfer in a gravity wave spectrum. Part I: a new method for efficient computations of the exact nonlinear transfer integral. *J. Phys. Oceanogr.*, **15**, 1369-77

-----, -----, J.A. Allender, and T.P. Barnet, 1985: Computations and parametrizations of the nonlinear energy transfer in a gravity wave spectrum. Part II: parametrization of the nonlinear transfer for application in wave models. *J. Phys. Oceanogr*., **15**, 1378-91

Hornik, K., 1991: Approximation Capabilities of Multilayer Feedforward Network, *Neural Networks,* Vol. 4, pp. 251-257

Komen, G.J., et al., 1994: *Dynamics and Modelling of Ocean Waves*, Cambridge University press, Cambridge, 532 pp.

Krasnopolsky V.M., D. Chalikov, L. C. Breaker, and D. Rao, 2000a: Application of neural networks for efficient calculation of sea water density or salinity from the UNESCO equation of state. Proceedings, Second Conference on Artificial Intelligence, AMS, Long Beach, CA, 9-14 January, 2000, pp. 27-30

Krasnopolsky, V., D. Chalikov, and H. Tolman, 2000b: A neural network approach to parametrizing nonlinear interactions in wind wave models, The Fourth Int. Symp. on Ocean Wave Measurement and Analysis, September 3-5, 2001, San Francisco, California, (in press)

Levitus,S., 1982: Climatological atlas of the world ocean. NOAA Prof. Pap., 13, 173pp.

Mamayev, O.I., 1975: Temperature - Salinity Analysis of the World Ocean Waters. Elsevier Scientific, Amsterdam.

Nguyen, D. and B. Widrow, Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights, International Joint Conference of Neural Networks, July 1990, **3**, 21-26, 1990

Resio, D.T. and W. Perrie, 1991: A numerical studyof nonlinear energy fluxes due to wave-wave interactions, 1, methodology and basic results. *J. Fluid Mech*., vol. 223, 603-629

SWAMP Group, 1985: *Ocean wave Modeling*, Plenum Press, 256 pp.

Tolman, H. L., 1999: User manual and system documentation of WAVEWATCH-III version 1.18. NOAA / NWS / NCEP /OMB technical note 166, 110 pp.

Tolman, H.L., and D.V. Chalikov, 1996: Source terms in a third-generation wind wave model. *J. Phys. Oceanogr*., **26**, 2497-2518.

UNESCO, 1981: The practical salinity scale 1978 and the International Equation of State for Seawater 1980. Tenth Report of the Joint Panel on Oceanographic Tables and Standards, UNESCO Technical Papers in Marine Science No. 36, UNESCO, Paris, France.

Van Veldder, G.Ph., T.H.C. Herbers, B. Jensen, D.T. Resio, and B. Tracy: Modelling of nonlinear quadruplet wave-wave interactions in operational coastal wave models. Abstract, accepted for presentation at ICCE 2000, Sydney

WAMDI Group, 1988: The WAM model - a third generation ocean wave prediction model. *J. Phys. Oceanogr*., **18**, 1775-1810.

Wessels L. F. A. and E. Bernard, Avoiding false local minima by proper initialization of connections, *IEEE Trans. Neural Networks , 3,* 899-905, 1992

Woodgate, R.A., 1998: Can we assimilate temperature data alone into a full equation of state model? *OCEAN modelling*, No.114, 4 - 5.