Available online at www.sciencedirect.com
ScienceDirect
Neural Networks 20 (2007) 462–478
Neural Networks
www.elsevier.com/locate/neunet

2007 Special Issue

Fast neural network surrogates for very high dimensional physics-based models in computational oceanography

Rudolph van der Merwe[a], Todd K. Leen[a,*], Zhengdong Lu[a], Sergey Frolov[b], Antonio M. Baptista[b]

[a] *Department of Computer Science and Electrical Engineering, OGI School of Science and Engineering, Oregon Health and Science University, Portland, OR 97006, USA*

[b] *Department of Environmental and Biomolecular Systems, OGI School of Science and Engineering, Oregon Health and Science University, Portland, OR 97006, USA*

Abstract

We present neural network surrogates that provide extremely fast and accurate emulation of a large-scale circulation model for the coupled Columbia River, its estuary and near ocean regions. The circulation model has $\mathcal{O}(10^7)$ degrees of freedom, is highly nonlinear and is driven by ocean, atmospheric and river influences at its boundaries. The surrogates provide accurate emulation of the full circulation code and run over 1000 times faster. Such fast dynamic surrogates will enable significant advances in ensemble forecasts in oceanography and weather.

Keywords: Fast neural network dynamic surrogates; Computational oceanography; High-dimensional time series prediction; Physics-based models; Data assimilation; River estuary modelling

1. Introduction

Numerous tasks in computational earth sciences require, or would benefit from, significant acceleration of the simulation of the dynamics of the system (Ferraro, Sato, Brasseur, DeLuca, & Guilyardi, 2003). In computational oceanography, relevant tasks include data assimilation using statistical ensemble approaches, parameter selection using iterative optimization techniques, engineering control approaches to river and estuary resource management and Monte Carlo studies.

State-of-the-art circulation codes such as POM (Blumberg & Mellor, 1987), ADCIRC (Luettich, Westerink, & Scheffner, 1992), QUODDY (Lynch, Ip, Naimie, & Werner, 1996), ROMS (Shchepetkin & McWilliams, 2005) and ELCIRC (Zhang, Baptista, & Myers, 2004) are capable of high-fidelity modelling, simulation and prediction of circulation in river estuaries and plumes. These circulation codes solve the governing partial differential equations on very large ($>10^6$ nodes) three-dimensional grids using numerical approaches such as finite difference or finite element methods. These methods require significant computational resource (CPU cycles, memory and disk space), and are ill-suited to tasks requiring hundreds of model evaluations or more. For example, our ELCIRC models of the Columbia River estuary system run from two to seven times faster than real-time, depending on the size of domain modelled; these execution times prohibit tasks using ensemble predictions or iterative parameter optimization.

We present *emulators* (or equivalently *surrogates*) that provide accurate predictions in a small fraction of the time required by the full circulation codes: in our experiments, the emulators run roughly *one thousand times* faster. This technology does not dispense with the circulation codes — we train emulators to mimic the dynamics of the circulation models. But once trained, the emulators are able to provide accurate predictions over new forcing regimes. This technology enabled the first data assimilation[1] system capable of dealing with the significant nonlinearities in river estuary modelling

* Corresponding address: Department of Computer Science and Engineering, OGI School of Science and Technology, OHSU, 20000 NW Walker Road, Beaverton, OR 97006, USA. Tel.: +1 503 748 1160; fax: +1 503 748 1553.
E-mail address: tleen@csee.ogi.edu (T.K. Leen).

[1] The term "data assimilation" as used in oceanography and meteorology refers to the fusion of data with numerical models to improve prediction (Bennett, 2002). This is also known as state estimation.

0893-6080/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2007.04.023

Nomenclature

$\mathbf{x}(k)$	state at discrete time k
$\mathbf{f}(\cdot)$	state evolution function
$y(k)$	scalar observation at discrete time k
$h(\cdot)$	scalar-valued observation function
τ	sampling period
t_0	origin of time coordinates
$\mathbf{x}_e$	time-delayed embedding of observations
$\tilde{\mathbf{f}}(\cdot)$	state evolution function for $\mathbf{x}_e$
$f_{NN}(\cdot)$	nonlinear map executed by neural network
$\mathbf{u}(k)$	exogenous forcings at discrete time k
$\mathbf{y}(k)$	vector observation at time k
$\mathbf{h}(\cdot)$	vector-valued observation function
n_x	state dimension
n_y	observation dimension
$\mathbf{x}_s(k)$	subspace projection of $\mathbf{x}(k)$
$\mathbf{\Pi}_s$	PCA projection operator
$\boldsymbol{\mu}$	ensemble state mean
$\mathbf{A}, \mathbf{S}, \mathbf{B}^{\mathrm{T}}$	matrices of singular value decomposition
$\mathbf{b}_i$	the ith column vector (eigenvector) of $\mathbf{B}$
s_i	the ith eigenvalue of $\mathbf{S}$
$\tilde{\mathbf{x}}$	time-delayed embedding of $\mathbf{x}_s$
$\tilde{\mathbf{u}}$	time-delayed embedding of exogenous forcings $\mathbf{u}$
$\mathbf{\Pi}_e$	second PCA projection operator
T_x	state history window length
T_u	forcings history window length
$\hat{\mathbf{x}}_s(k)$	neural network prediction at discrete time k
$\mathbf{X}_D$	input data matrix
$\mathbf{Y}_D$	target data matrix
α	weight decay regularization constant

(Frolov, Baptista, Lu, van der Merwe, & Leen, submitted for publication; Frolov, Baptista, Leen, Lu, & van der Merwe, 2006; Frolov, van der Merwe, Lu, Baptista, & Leen, 2006; Lu, Leen, van der Merwe, Frolov, & Baptista, 2007).

Our motivation for this work was the need for a data assimilation system to advance our modelling of the Columbia River estuary. We wanted a system that is portable across problem domains (with minimal requirements for problem-specific infrastructure) and able to deal with the very strong nonlinearities in estuary flow. This suggests an approach based on nonlinear extensions of the Kalman filter that use ensemble methods (van der Merwe, 2004). Such methods require the evaluation of the dynamics over a statistical ensemble of at least $2n+1$ members (with n the dimension of the state-space). For typical circulation codes, this has massive computational implications. For example, evaluating an ensemble of $2n+1$ members[2] in a 10^7 dimensional state space using a full circulation code such as ELCIRC, will take approximately 15×10^9 s (475 years)! Data assimilation clearly requires *fast emulators* of the circulation code.

Fast emulators of cumbersome differential equation solvers have applications beyond data assimilation. We are keenly interested in large-scale probabilistic forecasts for which this technology will enable sample sizes that dwarf current practice (Gneiting & Raftery, 2005) and thereby provide superior estimation.

Effective surrogates must provide fast evaluation with relatively low computational and memory requirements. They need to accurately mimic the dynamic properties of the full circulation codes, for all state variables of interest, with dependence on the initial state of the system, and dependence on the time series of external forcings.[3] They must be trainable from circulation code simulations and allow for extension across different dynamic regimes of the governing physics of the system.

Nonlinear multivariate time series prediction is one of the more promising algorithmic approaches available for the task. However, *direct* time series prediction on a state vector of the order of 10^6–10^7 variables is computationally prohibitive. We require a compact representation of the state that does not impose an extreme loss of information.

Given these requirements and constraints, we propose a surrogate system that uses a nonlinear, time-lag, externally-recurrent *neural network predictor*, with a *dimensionality-reduced* flow state. This surrogate system reduces the time needed to propagate an ensemble of $2n+1$ dimensionality-reduced states one time step (30 min) into the future, to 46 s. This allowed us to implement and successfully apply our ensemble-based data assimilation (DA) system.[4]

Neural network emulation of dynamic systems is well explored. Lapedes and Farber (1988) demonstrated early on that feedforward neural networks could accurately mimic nonlinear dynamic systems, even in chaotic regimes. Since then, neural networks have become the method of choice for nonlinear time series prediction (Bakker, Schouten, Giles, Takens, & van den Bleek, 2000; Casdagli, 1989; Principe, Rathie, & Kuo, 1992; Weigend & Gershenfeld, 1994). In most of these studies, the networks are used as predictors of an observed scalar time series generated by deterministic chaotic systems. These systems are usually autonomous (*unforced*), and the networks are trained from time-lag embeddings of scalar time series. The discrete time state-space model of an autonomous nonlinear dynamic system is[5]

$$\mathbf{x}(k+1) = \mathbf{f}(\mathbf{x}(k)) \tag{1}$$

$$y(k) = h(\mathbf{x}(k)) \tag{2}$$

where $\mathbf{x}(k)$ is the state of the system, $\mathbf{f}$ is a nonlinear vector function modelling the dynamics and $y(k)$ is a scalar observation of the system generated by a possibly

[2] We propagate $2n+1$ state vectors one time step (30 min) into the future.

[3] For the river estuary system, the state variables include the water elevation, temperature, salinity and flow velocities; the external forcings include tides, river flux, wind stress and solar radiation.

[4] A full exposition of our data assimilation system is in preparation (Frolov et al., submitted for publication; Lu et al., 2007). Due to space constraints, we do not include DA application results in this paper, but rather refer the reader to (Frolov, Baptista et al., 2006; Frolov, van der Merwe et al., 2006).

[5] Such discrete-time models are equivalent to the flow map induced by integration of the underlying differential equations in continuous-time dynamic systems. That is the context here.

nonlinear observation function h. In the above equations, $\mathbf{x}(k)$ corresponds to the state of the system at *discrete* time instances $\mathbf{x}(k) \doteq \mathbf{x}(t_0 + k\tau)$ where t_0 is the origin of the time axes and τ is the sampling period.

Through the implications of Takens's Theorem (Takens, 1981), Grassberger, Schreiber, and Schaffrath (1991) showed that the true state of the system, $\mathbf{x}(k)$, can be replaced by a time delayed embedding of the observed (scalar) variable $\mathbf{x}_e(k) = \begin{bmatrix} y(k) & y(k-1) & y(k-2) & \cdots & y(k-N) \end{bmatrix}^{\mathrm{T}}$. This allows for a dynamic state-space model of the form

$$\mathbf{x}_e(k+1) = \tilde{\mathbf{f}}(\mathbf{x}_e(k)) \tag{3}$$

which can be expanded as

$$\begin{bmatrix} y(k+1) \\ y(k) \\ \vdots \\ y(k-N+2) \\ y(k-N+1) \end{bmatrix} = \begin{bmatrix} f_{NN}(\ ) & & & & \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \ddots & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} y(k) \\ y(k-1) \\ \vdots \\ y(k-N+1) \\ y(k-N) \end{bmatrix} \tag{4}$$

where f_{NN} is a nonlinear map

$$\begin{aligned} y(k+1) &= f_{NN}(\mathbf{x}_e(k)) \\ &= f_{NN}(y(k), y(k-1), \ldots, y(k-N)). \end{aligned} \tag{5}$$

Thus, the task is to build a surrogate that can accurately predict the future time evolution of $y(k)$ based on past observations of the same time sequence $\{y(k-N), y(k-N+1), \ldots, y(k-1)\}$.

The thrust of Taken's theorem is that given a sufficiently large embedding space, the orbits of the original system (1) are diffeomorphic to those of the embedded system (3). The dynamics of the embedded system can be implemented as a feed forward neural network such as a multilayer perceptron (MLP) or radial-basis-function (RBF) network.

The systems we are modelling are not *autonomous* but rather strongly forced by tides, wind shear and river flux. A general formulation of such systems is

$$\mathbf{x}(k+1) = \mathbf{f}(\mathbf{x}(k), \mathbf{u}(k)) \tag{6}$$

$$\mathbf{y}(k) = \mathbf{h}(\mathbf{x}(k)), \tag{7}$$

where $\mathbf{x}(k)$ is the vector state of the system, $\mathbf{u}(k)$ are the exogenous forcings, and $\mathbf{y}(k)$ are the (possibly sparse) vector observation of the system's state.

For our circulation codes, $\mathbf{x}(k) \in \mathbb{R}^{n_x}$ and $\mathbf{y}(k) \in \mathbb{R}^{n_y}$, typically with $n_x > 10^6$ and $n_y \approx 100$ for stationary direct observation sites[6] of physical variables such as salinity, temperature, pressure and velocity. It is not feasible to build neural networks for direct prediction in such large-dimensional state spaces. Instead, we assume that the actual orbits of the system occupy a significantly lower-dimensional manifold and will use an appropriate *reduced-dimensionality* representation.

There is previous work using neural networks to approximate the physics of *forced* nonlinear systems: Su, McAvoy, and Werbos (1992) used a recurrent neural network to predict the time evolution of the state of two different chemical systems. They had relatively few degrees of freedom, and the networks were trained on measurements of the full state. Bishop, Haynes, Smith, Todd, and Trotman (1995) used a neural network to predict parameters describing the shape of the boundary of a Tokamak plasma from sparse measurements of the magnetic field inside the reactor. They trained the neural network on simulated data generated by numerical solution of the governing physics model using a free-boundary equilibrium code. They do not use the network to predict the entire state of the plasma, only geometrical parameters that specify the boundary. The *NeuroAnimator* introduced by Grzeszczuk, Terzopoulos, and Hinton (1998) uses a neural network to learn from simulation the dynamic physics-based models of the motion of vehicles, humans and animals used for graphics. The trained network is used as a surrogate for the physics-based models, allowing fast simulation for computer animation. Their approach is similar to ours, but does not scale up to high-dimensional state spaces. Chevallier, Cheruy, Scott, and Chedin (1998) developed a fast neural network approach for modelling long-wave radiative transfer for atmospheric models. Krasnopolsky, Chalikov, and Tolman (2002) and Krasnopolsky and Chevallier (2003) used neural network modules to replace computational blocks within standard high-dimensional numerical environmental models. Their networks are not used to emulate the dynamics of the full system state directly, but rather to learn complex parameterizations of several types of physical processes (nonlinear mappings from exogenous inputs and certain state variables to other state variables) that are needed for the time-evolution of the system state. This allows for considerable speed up of the numerical models. Tang and Hsieh (2003) and Li, Hsieh, and Wu (2005) developed hybrid coupled models of the tropical Pacific that model the atmosphere response with a neural network and the ocean by a dynamical model. Tolman, Krasnopolsky, and Chalikov (2005) proposed to extend Krasnopolsky et al. (2002)'s neural network based approach (mentioned above) by making use of an empirical orthogonal function (EOF), or principal component (PC), decomposition to reduce the size of the modelling problem in the context of oceanic applications.

1.1. Description of circulation codes

Although the approach presented in this paper is not limited to a specific computational code, we use the ELCIRC circulation code (Zhang et al., 2004), developed and used extensively in our group.[7] ELCIRC is an unstructured-grid model designed for the effective simulation of 3D

[6] For remote sensing, such as satellite-based observations of surface characteristics, n_y can be of a much larger dimension.

[7] Centre for Coastal and Land-Margin Research, OHSU, http://www.ccalmr.ogi.edu.

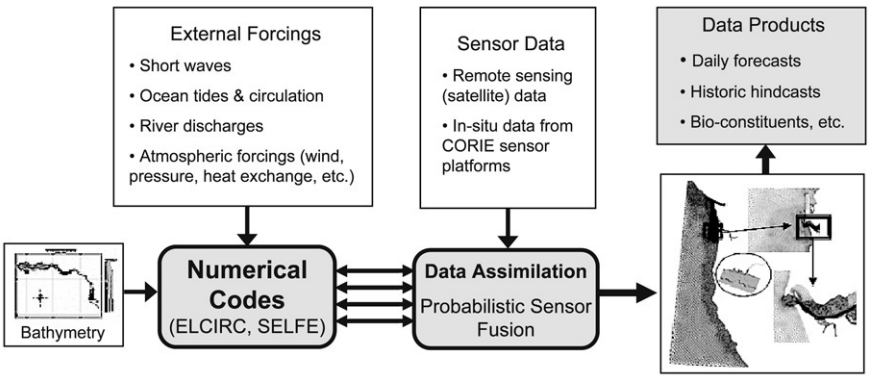

Fig. 1. CORIE is a pilot environmental observation and forecasting system (EOFS) for the Columbia River. It integrates a real-time sensor network, a data management system, advanced numerical models such as ELCIRC or SELFE and an advanced data-assimilation framework.

baroclinic circulation across river-to-ocean scales. It uses a finite-volume–finite-difference Eulerian–Lagrangian semi-implicit algorithm to solve the shallow water equations. It realistically addresses a wide range of physical processes and includes atmospheric, ocean and river forcings, and naturally incorporates wetting and drying of tidal flats (Baptista et al., 2005). ELCIRC uses an unstructured grid in the horizontal direction and z-coordinates in the vertical direction. The numerical algorithm is low-order, volume conservative, stable and computationally efficient. The state variables are elevation, salinity, temperature and horizontal velocity specified on the unstructured grid. The grid size depends on the system simulated. ELCIRC typically runs 2–7 times faster than real-time on a single CPU, depending on the system size, grid resolution and CPU speed. This is prohibitively expensive for ensemble-based inference tasks, such as data assimilation, that require the evaluation and propagation of large (>100 member) ensembles. While ELCIRC was originally developed to meet specific modelling challenges for the Columbia River estuary modelling and forecasting system (CORIE[8]) (Baptista et al., 1999), it has been extensively tested against standard ocean–coastal benchmarks and is starting to be applied to estuaries and continental shelves around the world. The CORIE modelling system (see Fig. 1) runs on a parallel cluster of 20 Intel dual CPU nodes (2.4–3.6 GHz, 4 GB memory) with 48 TB of primary storage. The computational engine is ELCIRC (Zhang et al., 2004) and data products are daily forecasts and long-term hindcast databases.[9] For the whole CORIE domain (river, estuary, plume and boundary ocean), the grid used by ELCIRC contains roughly 10^7 vertices.

2. Approach

The need for fast, accurate surrogates that can be trained from simulations using the full circulation code, and provide good generalization to forcing time series not seen in the training data is a formidable task. Here we present the system architecture that evolved from these requirements. It includes a dimensionality reduction of the full circulation state space, and a neural network that predicts the future state from a history of states and forcings. The network is regularized using standard weight decay and operates in an iterative prediction mode, that is, with external recurrence from the output back to the time-delayed state inputs.

2.1. Subspace projection of simulation data

Since an ultra-fast direct prediction on the full 10^7 dimensional ELCIRC state is not possible, we use a singular-value decomposition (SVD) (Golub & van Loan, 1996) implementation of principle component analysis (PCA) (Jollife, 1986) to reduce the very high dimensional state space to a manageable size

$$\mathbf{x}(k) \in \mathbb{R}^N \xrightarrow{\text{PCA}} \mathbf{x}_s(k) \in \mathbb{R}^r, \tag{8}$$

with $N \approx 10^7$ and $r \approx 30$. The projection is implemented by

$$\mathbf{x}_s(k) = \mathbf{\Pi}_s \left(\mathbf{x}(k) - \boldsymbol{\mu}\right), \tag{9}$$

where $\mathbf{x}(k)$ is the state variable in the full space at time k, $\boldsymbol{\mu}$ is the ensemble mean of $\mathbf{x}$, and $\mathbf{\Pi}_s$ is the PCA projection onto the subspace of the leading r eigenvectors of the covariance of original state. The T-by-10^7dimensional data-matrix of ELCIRC simulation states for time $k = 1 \ldots T$ is denoted by

$$\mathbf{X} \doteq [\,\mathbf{x}(1) - \boldsymbol{\mu} \quad \mathbf{x}(2) - \boldsymbol{\mu} \quad \cdots \quad \mathbf{x}(T) - \boldsymbol{\mu}\,]^{\mathrm{T}}. \tag{10}$$

The SVD decomposition of $\mathbf{X}$ is given by

$$\mathbf{X} = \mathbf{ASB}^{\mathrm{T}}, \tag{11}$$

where $\mathbf{A}$ $(T \times N)$ is column-orthogonal, $\mathbf{B}$ $(N \times N)$ is orthogonal and $\mathbf{S}$ is a diagonal matrix of the singular values $s_i, i = 1, \ldots, N$. The columns of $\mathbf{B} = [\,\mathbf{b}_1 \quad \mathbf{b}_2 \quad \cdots \quad \mathbf{b}_N\,]$ are the eigenvectors of the covariance of the sample vectors. The corresponding eigenvalues are s_i^2/T. Any vector $\mathbf{x}$ in the state space can be written as a linear combination of the eigenvectors

$$\mathbf{x} = \sum_{i=1}^{m} \alpha_i \mathbf{b}_i + \boldsymbol{\mu}. \tag{12}$$

[8] http://www.ccalmr.ogi.edu.

[9] *Hindcast* refers to simulations over past periods for which full boundary conditions are available. In *forecasting* some of the boundary conditions and forcings, such as water release at the dams, are uncertain.

The PCA projection operator $\mathbf{\Pi}_s$ is formed by retaining the leading r basis vectors (corresponding to the *largest* singular values) of $\mathbf{B}$,

$$\mathbf{\Pi}_s \doteq [\,\mathbf{b}_1 \quad \mathbf{b}_2 \quad \cdots \quad \mathbf{b}_r\,]^{\mathrm{T}}, \tag{13}$$

with $r < N$.

We are assuming that the original 10^7-dimensional state has many redundant degrees of freedom, and that we can reduce this redundancy by the PCA projection without impairing our ability to construct accurate surrogates. For our pilot system, we retain 30 degrees of freedom ($r = 30$ in (13)). For a one month time series, the 30-dimensional PCA captures approximately 95% of the variance exhibited in the full numerical model (ELCIRC).

2.2. Time embedding & PCA projection of input vectors

Given a fixed time window (length T_x) of current and past states of the system in the PCA subspace (a delay vector of time-embedded states),

$$\tilde{\mathbf{x}}(k, T_x, \tau) = [\,\mathbf{x}_s^{\mathrm{T}}(k) \quad \mathbf{x}_s^{\mathrm{T}}(k-1) \quad \cdots \quad \mathbf{x}_s^{\mathrm{T}}(k-N+1)]^{\mathrm{T}},$$
$$N = \frac{T_x}{\tau}, \tag{14}$$

and the time series history of forcings (window length T_u),

$$\tilde{\mathbf{u}}(k, T_u, \tau) = [\,\mathbf{u}^{\mathrm{T}}(k) \quad \mathbf{u}^{\mathrm{T}}(k-1) \quad \cdots \quad \mathbf{u}^{\mathrm{T}}(k-M+1)]^{\mathrm{T}},$$
$$M = \frac{T_u}{\tau}, \tag{15}$$

a neural network is used to predict the next state (in the PCA subspace) of the system,

$$\mathbf{x}_s(k+1) = f_{NN}\left(\mathbf{z}(k, \mathbf{\Pi}_e, T_x, T_u, \tau)\right), \tag{16}$$

where

$$\mathbf{z}(k, \mathbf{\Pi}_e, T_x, T_u, \tau) = \mathbf{\Pi}_e \mathbf{w}(k, T_x, T_u, \tau). \tag{17}$$

In the above set of equations, $\mathbf{w}(k, T_x, T_u, \tau)$ is the concatenated input delay vector

$$\mathbf{w}(k, T_x, T_u, \tau) = [\,\tilde{\mathbf{x}}^{\mathrm{T}}(k, T_x, \tau) \quad \tilde{\mathbf{u}}^{\mathrm{T}}(k, T_u, \tau)\,]^{\mathrm{T}}, \tag{18}$$

τ is the sampling period (equal to the embedding time step).

The operator $\mathbf{\Pi}_e$ is a *second* linear PCA projection operator. Although the coordinate components of the subspace states $\mathbf{x}_s$ are uncorrelated by construction, there is considerable correlation between the time-delayed input vectors in (14), and presumably between the time-delayed forcing vectors in (15). This second PCA projection removes those correlations and reduces the size of the input layer of the network, which improves generalization and optimization.

This input preprocessing step allows us to *independently* specify the input window lengths (dictated by the low frequency characteristics of the system, such as tidal periods) and the sampling time (dictated by the Nyquist sampling criteria), and still retain a compact network representation. We chose to retain about 99% of the variance of the input to the PCA, which for our specific problem domain, typically results in a factor 5–10 reduction in the dimension of $\mathbf{w}(k, T_x, T_u, \tau)$ to $\mathbf{z}(k, \mathbf{\Pi}_e, T_x, T_u, \tau)$.

We thus have three parameters available for adjusting the size of the state representation input to the neural network: (1) the number of dimensions retained in the PCA subspace projection $\mathbf{x}_s(k)$ of the full ($\approx 10^7$-dimensional) numerical state vector, (2) the number of past subspace projections incorporated into the delay vector of time-embedded states $\tilde{\mathbf{x}}(k, T_x, \tau)$, and (3) the number of dimensions retained in the PCA projection of the vector $\mathbf{w}(k, T_x, T_u, \tau)$ of concatenated state and forcing histories. The first two operations provide alternative controls on the amount of state information available to the network. One can capture more state information by increasing the number of dimensions retained in the PCA subspace projection, or by increasing the number of time-lags in the embedded state $\tilde{\mathbf{x}}$. We have elected, somewhat arbitrarily, to retain a significant amount of the variance (95%) in the PCA projection and tune the amount of state information available by adjusting the number of lagged states used by cross-validation on the prediction error.[10] One could tune both the PCA dimension and the number of lagged states, at the cost of adding significant complexity to the model selection process.

For our surrogate implementation we arrived at input vectors $\tilde{\mathbf{x}}(k, T_x, \tau)$ and $\tilde{\mathbf{u}}(k, T_u, \tau)$ consisting of the current (reduced dimensionality) state and history running back 30 h in 30 min increments ($N = 60$ in Eq. (14) with $\tau = 0.5$ h). A 30 h history of tides, wind, river flux and air pressure, also at 30 min intervals, capture the model forcings ($M = 60$ in Eq. (15)). The optimal values of N and M were determined through empirical cross-validation around a nominal window length of 24 h. The window length was suggested by a harmonic analysis of the main tidal forcing frequencies which dominate the system's response. The eight main diurnal and semi-diurnal tidal components caused by the gravitational influence of the sun and moon are: M2 (12.42 h period), S2 (12.00 h period), N2 (12.66 h period), K2 (11.97 h period), K1 (23.93 h period), O1 (25.82 h period), P1 (24.07 h period) and Q1 (26.87 h period) (Dietrich, Kalle, Krauss, & Siedler, 1980). The optimal value of $N = M = 30$ h suggests that this is the minimum window length needed to accurately resolve and capture the frequency content of the periodic inputs to the neural network. Fig. 2 shows a power spectrum of the tidal forcings signal that was used as input to the experiment we discuss in Section 3.

2.3. Neural network structure

Our neural-network surrogates are externally recurrent (iterative prediction) nonlinear feed-forward multi-layer perceptrons (Bishop, 1995; Werbos, 1990). Such networks are very

[10] In the extreme case, one could imagine reducing the number of dimensions retained in the first PCA project to unity, and embed a very long time series. In fact, Taken's theorem is usually invoked to develop phase-space portraits from such *scalar* time-series. However, significant nonstationarity in the observed river estuary dynamics suggests using shorter time series and embedding *vector* samples as we have done here.

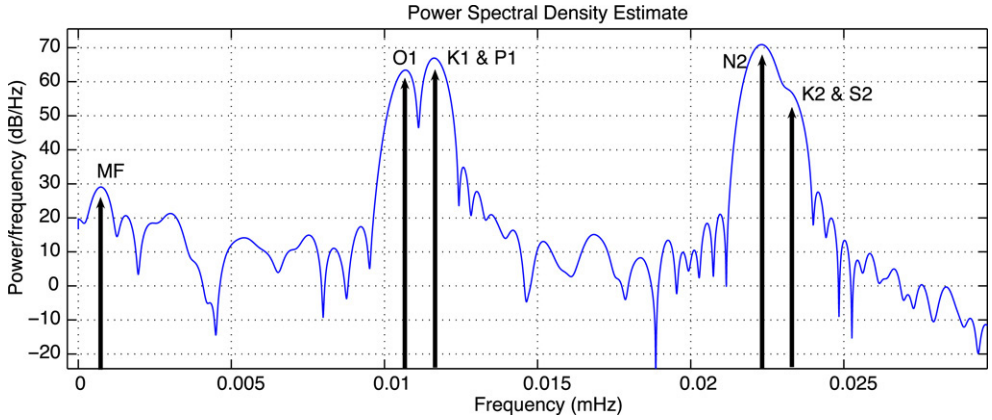

Fig. 2. Frequency analysis of tidal forcings.

well-equipped for modelling nonlinear relations among high-dimensional inputs and outputs where large datasets are available for model fitting (training). Where there is significant nonlinearity, their performance far exceeds that of traditional linear models such as ARX, ARMA, ARMAX and GLMs (Bishop et al., 1995).

In fact, for our prediction problem we found that linear predictors (in this case a model of the form $\mathbf{x}(k+1) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(k) + \mathbf{d}$), fitted using standard robust least-squares regression techniques were *inherently unstable* with poles lying outside the unit circle. This causes the prediction response to grow exponentially and become unbounded, which precludes their use in circulation code surrogate applications such as data assimilation.

Fig. 3 shows a schematic diagram of the general neural network structure used for our surrogates. The delay lines of state and forcing inputs feed into the PCA preprocessing front end that reduces the dimensionality of the neural-network input vector. The neural network is a multilayer perceptron (MLP) using a single hidden layer with hyperbolic tangent activation functions, and a linear output layer. This architecture is often used for general nonlinear regression and time series prediction problems. The size of the network's input and output layers are dictated by the dimension and embedding length of the subspace state variables and forcings, as well as the number of degrees of freedom retained by the secondary PCA, $\mathbf{\Pi}_e$. The *size* of the hidden layer can be set in order to control total model complexity (number of free parameters). Typically, this hyperparameter can be set using some form of cross-validation. We use a generous hidden layer and control model complexity by weight-decay regularization (Bishop, 1995).

The output of the PCA front-end that feeds into the neural network input layer is *whitened* (each component is mean-zeroed and normalized by the square root of the long-term variance as calculated on the training dataset). This improves the condition number of the Hessian, and hence convergence during training.

The neural network output prediction of the next subspace state of the system, $\hat{\mathbf{x}}_s(k+1)$, is fed back through a unit delay operator to the input of the state delay-line (see Fig. 3). In this fashion the feed-forward network becomes an externally recurrent network (Su et al., 1992). Even though the state

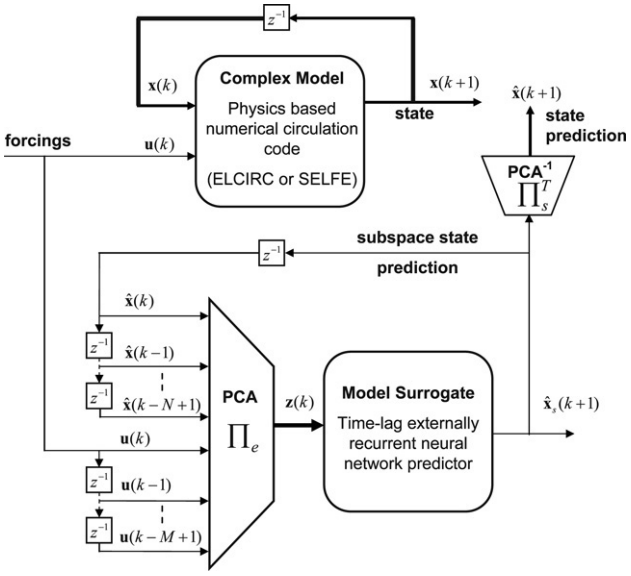

Fig. 3. Schematic diagram of circulation code surrogate operational framework.

of the neural network surrogate is recursively propagated in the subspace, one needs to *re-embed* this state into the full numerical model space in order to use it in the same manner as the full circulation code state predictions. This re-embedding operation is given by

$$\hat{\mathbf{x}}(k) = \mathbf{\Pi}_s^{\mathrm{T}} \hat{\mathbf{x}}_s(k) + \boldsymbol{\mu}. \tag{19}$$

2.4. Neural network training

We investigated two different neural network training strategies typically used for time series prediction. Both approaches are supervised training methods (Bishop, 1995). The first approach, which we call *SUR*1, trains the neural network to predict the next state of the system in the subspace. The second approach, *SUR*2,[11] trains the network to recursively predict the next P steps into the future. *SUR*1 can be interpreted

[11] From now on we will refer to the network trained with the *SUR*1 strategy as *SUR*1, and likewise the *SUR*2 trained network will simply be called *SUR*2.

as a special case of *SUR*2 with $P = 1$. Although both *SUR*1 and *SUR*2 are used in the same externally recurrent feedback manner (see Fig. 3), only *SUR*2 is trained in a recurrent fashion. This training (and the cost function it minimizes) more accurately reflects the manner in which the networks will be used. The training targets for *SUR*2 are the P consecutive future subspace states. The corresponding future forcings also need to be provided. The optimal value of P is determined by cross-validation. Su et al. (1992) showed that for certain time series prediction problems, the recurrent training used for *SUR*2 leads to better long-term iterative feedback prediction performance, compared to the one-step ahead training of *SUR*1.

The free parameters (weights) of both surrogate networks are optimized (trained) using the scaled conjugate gradient (SCG) algorithm (Møller, 1996) with weight-decay regularization to control model complexity (Bishop, 1995). The optimal value of the weight-decay regularization constant for each surrogate was determined using N-fold cross-validation on a validation set of data. For *SUR*1, the network gradients (which are needed by the SCG optimizer) are calculated using standard *backpropagation* (Bishop, 1995). For the recurrently trained *SUR*2 network, the gradients are calculated using the *backpropagation-through-time* (BPTT) algorithm. This can be interpreted as doing normal backpropagation on a recurrent network structure that has been *unfolded in time* (Werbos, 1990). The computational burden for the SCG/BPTT recurrent training is significantly more than for one-step- ahead training (it scales with the prediction horizon length P).

We split the available training data into three sets: a *training set* (used for weight optimization), a *validation set* (used to adjust network regularization), and a *test set* (used for independent performance evaluation). The training and validation sets (collectively called the development set) are derived from the first time-contiguous block of data in the SVD subspace. We construct a data matrix of input vectors and corresponding targets,

$$\mathbf{X}_D = [\,\mathbf{w}(0, T_x, T_u, \tau) \quad \mathbf{w}(1, T_x, T_u, \tau) \quad \cdots \quad \mathbf{w}(K_D - 1, T_x, T_u, \tau)\,] \tag{20}$$

$$\mathbf{Y}_D = [\,\mathbf{x}_s(1) \quad \mathbf{x}_s(2) \quad \cdots \quad \mathbf{x}_s(K_D)], \tag{21}$$

where K_D is the number of datapoints in the development set, and then split it 2:1 into a training set and a validation set. The remaining time-contiguous block of data ($k > K_D$) forms the independent test set that is never used during neural network training. The test set is used for independent performance evaluation after regularized training is concluded.

Determining which part of the *development set* to use for training and which part for validation is nontrivial. We want two independent sets with no overlap, but with similar statistical and dynamical properties. This imposes strong constraints on the stationarity of the data and forcings during the period in question. We want the same forcing regimes covered by data in both the training and validation set, without having overlap of the exact same data points. This is hard to realize with finite length datasets constructed by time embedding of forcings which themselves are highly nonstationary. We investigated the use of two different methods for splitting the development set into training and validation subsets.

The first method uses a *random shuffling* of the $\{\mathbf{X}_D(i), \mathbf{Y}_D(i)\}$ input–output pairs. For N-fold cross-validation, we randomly generate N different such development set shuffles. The advantage of this method is that we get good coverage in time (of the different forcing regimes) in both the training and the validation set. However, due to the *time-embedding* there is actually a significant amount of *overlap in time* between some of the components of the input-vectors in the training and validation sets, which might lead to under-regularization. This might adversely affect the generalization performance of the trained surrogates.

The second method uses a *sliding window* approach to select 2/3 of the development set for the training and the remaining 1/3 for validation. Here the individual input–output pairs of the subsets are contiguous in time (no shuffling). For each of the N development set splits, the contiguous window is moved ahead by the corresponding number of input–output pairs (in a circular buffer fashion) to generate the next training/validation split. This second approach results in much less overlap in time between the training and validation subsets. However, if there is significant non-stationarity in the forcings, then there might be very little signal range coverage similarity between the training and validation sets. These larger differences between the two subsets, can lead to network over-regularization and subsequent suboptimal generalization performance.

3. Experimental results

We tested the proposed neural-network surrogate approach on a one month long ELCIRC simulation of the Columbia River estuary (CORIE) system. The simulation covers June 30 through August 28, 2002. The reference simulation is a part of the multi-year hindcast simulation database generated for the CORIE domain (Baptista et al., 2005). The modeling domain extends from Bonneville dam (the last dam on the Columbia River, 240 km upstream) to the continental shelf of Oregon, Washington, California and British Colombia (See Fig. 4). The computational grid, totalling ≈34000 nodes[12] in the horizontal grid and 62 vertical layers, expends most of its resource in the river, estuary and plume. The simulation is forced by tides, winds, air pressure, river discharge and radiative flux. Outputs from global and regional ocean and atmospheric models provide ocean boundary conditions and wind forcing.

Prominent dynamic features in the simulations are the intrusion of salt into the estuary (driven by density gradients, tidal forces, and river discharge) and the dynamics of the freshwater plume (driven by wind forces, river discharge and Coriolis). Comparisons with data suggest that the reference simulation can realistically represent aspects of the salt intrusion and the freshwater plume (Baptista et al., 2005).

We chose to use the specific month long sequence of data mentioned above for the following two reasons: (1) We

[12] Each computational element is a triangular prism with six vertices and five faces.

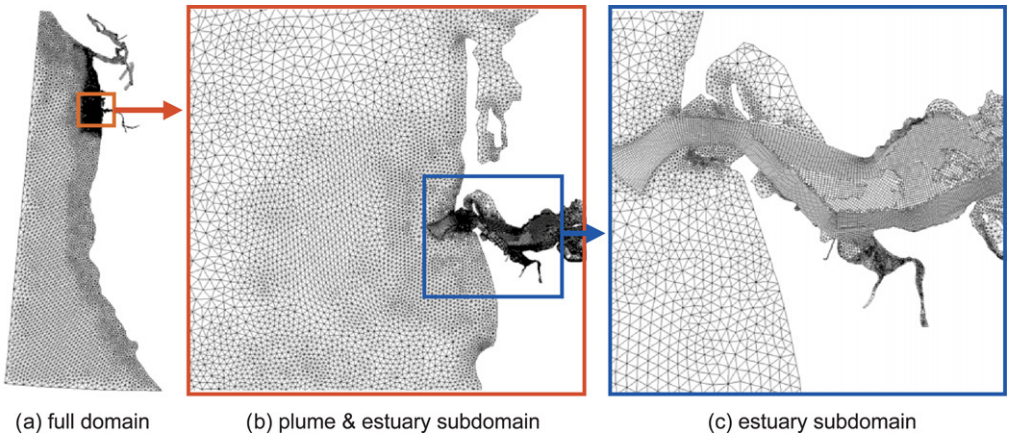

Fig. 4. The CORIE domain.

had to pick a simulation period that had good availability (in time and space) of observational data (from the CORIE distributed sensor network). This requirement was dictated by the need to incorporate the trained surrogates into our data assimilation system. (2) We chose a time span that limited the nonstationarity of the simulated data. One of the contributing causes for nonstationarity in the simulated data sets are the inherent nonstationarities in the input forcings which drive the simulations (Jay & Flinchem, 1997). These nonstationarities in the forcings can persist over multiple seasons and annual cycles. This increases the nonstationarity of longer-simulated time series and compound the emulation problem. We expect that emulation over long periods of time that include substantial nonstationarity will require more complex *mixture-of-expert models* (Jacobs, Jordan, Nowlan, & Hinton, 1991) or similar techniques specifically crafted to deal with nonstationarity.

Fig. 4 shows the plume and estuary parts of the simulation domain and how it relates to the full simulation space. The figure also indicates the variations in the density of the simulation grid. Note that the grid is denser in the plume and estuary than in the extended ocean. This is required to accurately capture the short-length scale physical processes that dominate these regions. A side effect of the concentration of grid points in the river and estuary is that those regions will be more strongly represented in the SVD front-end than the plume and ocean regions.

The SVD analysis of the CORIE simulations led us to a 30-dimensional subspace (30 leading EOFs retained as a basis for subspace projection), which captured about 95% of the variance. The samples included in the SVD were drawn at random from the total four-week long simulation set. The total number of 10^7 dimensional vectors used for the SVD calculation was constrained by the memory requirements of the SVD implementation executed on a 2 GB, 32 bit machine. This restricted us to 130 state vectors for the subspace calculation. The SVD training and test sets show roughly equal error[13] indicating that these sampling constraints did not severely hinder us. Longer simulation periods could require more samples, for which we will turn to more memory efficient SVD implementations such as Lanczos-(or Krylov-) based methods (Golub & van Loan, 1996).

We trained the neural network surrogates to emulate the ELCIRC predictions projected onto the SVD subspace for the same four-week period. The first three weeks of time-consecutive data were used for development (2/3 for training set and 1/3 for validation set) and the fourth week was used for the independent test set.

All of our subsequent performance reporting will show the surrogates' predictive performance for the *entire* four-week period (the three weeks spanned by the development set and the one week test set). Since we use the surrogates in an *iterative prediction mode*,[14] the input state data seen by the surrogates during testing are not identical to that used during training. For this reason it is informative to see how the surrogates perform on both the development and test sets in a single four-week long predictive run.

3.1. Experiment 1

For this experiment, we train both the *SUR*1 and *SUR*2 surrogates using the *random shuffle* N-fold cross-validation methodology with $N = 3$. For *SUR*1 (trained for one-step ahead prediction 30 min into the future), the resulting optimal weight decay regularization constant is $\alpha = 277$. The development set has 632 input–output training pairs and 316 input–output validation pairs. Each input vector has 2940 components (30 dimensional SVD subspace states embedded at 30 min intervals for 30 h + 30 h embedding of multivariate forcings at 30 min intervals). The secondary PCA ($\mathbf{\Pi}_e$) further reduces this to 298 components, retaining 99% of the variance. The resulting network topology is 298 input, 328 hidden, and 30 output units resulting in 107,942 free parameters (weights). Convergence (based on one-step ahead prediction on the validation set) occurred at around 1000 iterations through the dataset.

[13] Signal variance *not* captured by the principal directions ≈ 5%–7%.

[14] The predicted state at time t is used to provide one of the inputs for the prediction of the state at time $t + 1$. See Fig. 3.

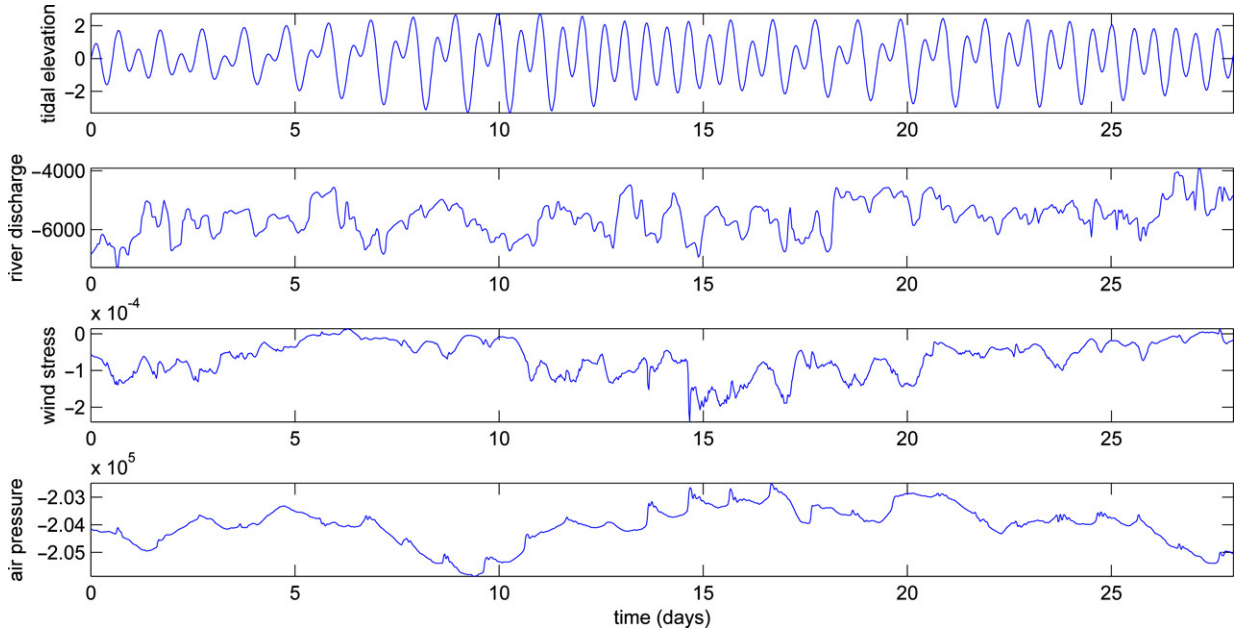

Fig. 5. Time series plots of the first principal component of each of the four classes of forcings used as exogenous input to the surrogates.

For *SUR*2 (recurrently trained using backprop-through-time), the resulting optimal weight decay regularization constant is $\alpha = 178$. The network is optimized to recursively predict the next three future states, 1.5 h into the future. The total number of weights are constrained to be the same as that for *SUR*1. The resulting network topology is 298 input, 278 hidden and 90 output units. *SUR*2 takes much longer to train than *SUR*1 due to the increased complexity of computing the recursive derivatives in the backpropagation-through-time algorithm. Convergence occurred at around 1500 iterations through the dataset.

We make use of weight decay regularization and N-fold cross- validation to adjust the "*effective number of parameters*" of the networks (Moody, 1992; Murata, Yoshizawa, & Amari, 1994). This addresses overfitting and generalization concerns in cases such as this where we have to fit a large number of parameters using a limited data set.[15]

For our discussion of the prediction results and generalization ability, it is helpful to examine the exogenous neural network input forcings throughout the four-week period. Fig. 5 shows the four main groups of forcings: tidal elevation at the deep ocean boundary, river flux at the upriver interface, wind stress on the surface interface and air pressure.[16]

Fig. 6 shows the time-series traces of the predicted output (in the SVD subspace) as generated by *SUR*1 (dashed line) and *SUR*2 (thin solid line) compared to that of the ELCIRC simulation projected onto the SVD subspace (thick black line). The plot displays three of the thirty SVD EOF coefficient time-series predictions.

The lower order EOFs explain more of the variance in the signal, so errors in these components dominate the cost-function being minimized by the neural network training. Clearly, EOF-1 captures the main tidal constituents and both surrogates are able to predict this component very well, both in the development set period (up to day 21) and in the test set period (days 21–27). In the higher frequency component (EOF-30) the surrogate predictions have more pronounced errors starting at day 21 (the beginning of the test set). This is most likely due to generalization difficulties the surrogates are experiencing due to under-regularization during training. This in turn is possibly caused by the larger than desired overlap-in-time between the training and validation sets (a disadvantage of the *random shuffle* cross-validation methodology when used with long time-embeddings of the input vectors). Another cause for the larger errors in the test set is the nonstationarity of the forcings between the development and testing parts of the data set. This can be seen in Fig. 5: Note the high value and positive trend of the river discharge during the final week (test set) which is coupled with a high trending wind stress, a temporal input pattern which is *not* represented in the development set (days 1–21).

Fig. 7 shows a total (sum over all variables) error analysis for different parts of the CORIE domain. We first re-embed the surrogate predictions (of the subspace state progression) in the full space, and then calculate the variance normalized mean-square error (MSE) between the surrogate predictions and that of ELCIRC. The error between the original full ELCIRC simulation and the ELCIRC projection onto the SVD subspace (the SVD reconstruction error) is calculated in a similar fashion.

[15] These regularized neural network surrogates have proven to be consistently stable in time and space as a key component of our data assimilation system (Frolov, Baptista et al., 2006; Frolov, van der Merwe et al., 2006; Lu et al., 2007). This further corroborates the widely held view that large regularized networks generalize better than smaller underfitted (biased) networks (Caruana, Lawrence, & Giles, 2000).

[16] Only the first principal component of each of the multi-variate time series signals is shown.

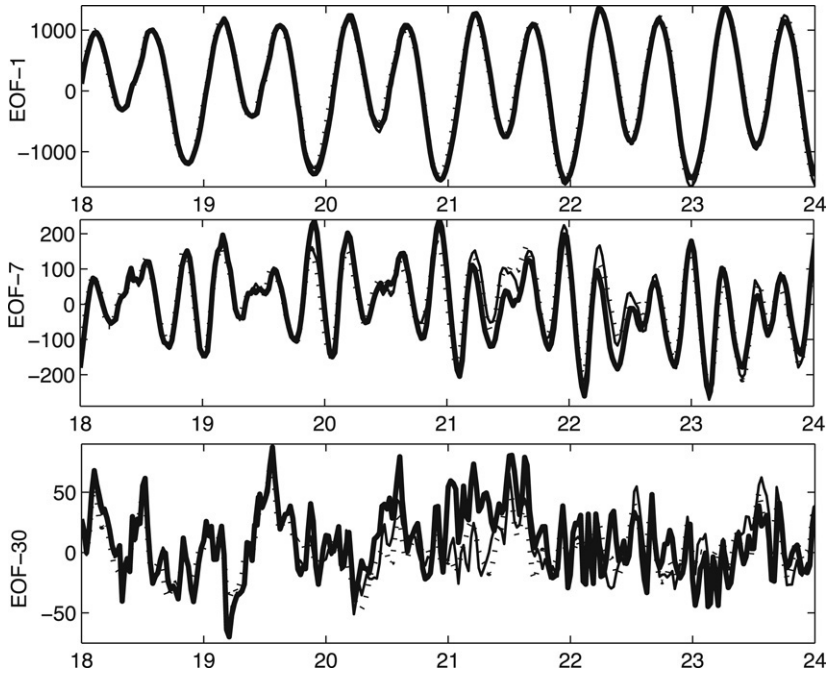

Fig. 6. Experiment 1: Surrogate feedback iterative time-series prediction of three of the 30 SVD subspace components. (bold line) ELCIRC in SVD subspace, $(\cdots)$ *SUR*1, $(-)$ *SUR*2. The x-axis is time in days and the y-axis is EOF coefficient amplitude.

The left column of plots shows error traces for the surrogates vs. ELCIRC in the subspace (dashed and thin solid lines) as well as the SVD reconstruction error (thick solid line). This error concurs with the design specification of the SVD analysis that aimed for a 5%–7% error. The right column of plots shows the total error in the full space, that is, the surrogate prediction error combined with the SVD reconstruction error. Each row of plots indicate the error in different parts of the CORIE domain. The top row shows the error in the estuary, the middle row the error in the plume region and the bottom row indicates the error for the whole domain (estuary + plume + rest). The errors in the plume region, particularly the SVD reconstruction error, are larger on average than those in the estuary. This is most likely due to the much larger variability and nonstationarity of the plume dynamics compared to the more periodic, stationary and advection-dominated estuary dynamics.

The temporal behaviour of the prediction errors shows the surrogates performing better during days 0–21 (covered by development set) than during days 21–27 (test set period). This implies possible (but not conclusive) under-regularization of the network which adversely effects generalization performance. This problem is most likely caused by the high level of nonstationarity of the forcings when moving from the development set to the test set. There seem to be very little difference between the performance of the two different surrogates: Even though *SUR*2 outperforms *SUR*1 in the development set period, the difference is much less significant in the test set period.

The actual physical state variables being modelled by ELCIRC are elevation, salinity, temperature and velocity. For further insight into the predictive fidelity of the surrogates we will next look at full space visualizations of a subset of one of these variables, the *surface salinity field*.[17] This is an interesting variable to visualize since it will indicate how well the surrogates are capable of modelling the highly variable behaviour of the fresh water plume (low salinity) as it discharges from the river estuary into the coastal ocean. We look at the surrogate predicted surface salinity field and compare it to the ELCIRC simulation, both in the full space and projected onto the SVD subspace. For comparative visualization purposes, we re-embed all subspace fields in the full space using Eq. (19).

Fig. 8 shows the salinity field for the plume region of the domain for day 14 (top four panels) and day 27 (bottom four panels) of the simulation. On day 14 the plume is separated from the coast and moving in a south westerly direction. The surrogates are clearly able to accurately capture the gross structural characteristics of the plume. The largest source of error on day 14 seems to be due to the SVD subspace projection; the differences (error) between the top-left and top-right fields are more pronounced than those between the bottom two fields and the top-right field. The results for day 27, the last day in the test set, is shown in the bottom four panels of the figure. Even though the surrogates still capture the general shape of the plume, there are more pronounced discrepancies compared to the original ELCIRC simulation. This corroborates the total error trace of Fig. 7. The surrogates seem to predict a lower salinity to the north of the estuary mouth than ELCIRC and a slightly higher salinity to the south. At this point in the

[17] Although the static results presented here captures most of the salient points relating to the potential fidelity of the neural network surrogates, it is very insightful to view video animations of their dynamic performance at the following URL: http://purl.oclc.org/NET/nnsvid.

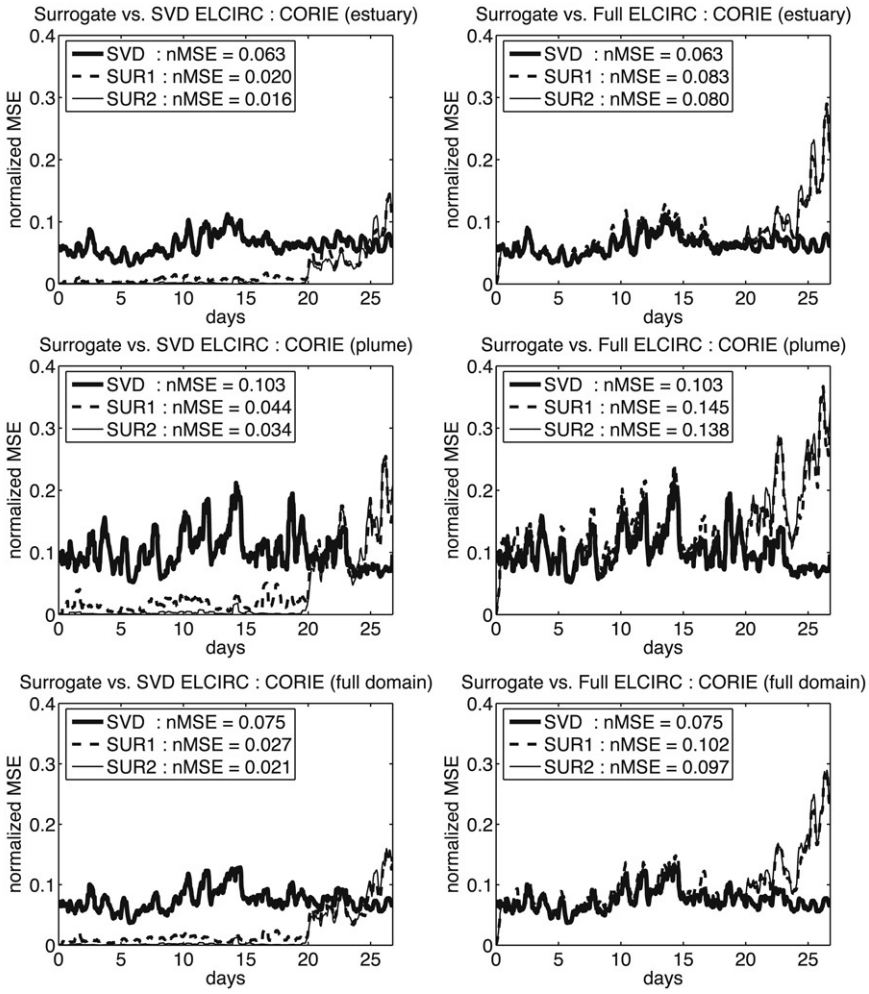

Fig. 7. Experiment 1: Normalized errors — *SVD* is the SVD reconstruction error, *SUR*1 is the feedforward network based surrogate error, and *SUR*2 is the recurrent neural network-based surrogate error. [left column] Surrogates vs. ELCIRC in SVD subspace: (top) estuary subdomain, (middle) plume subdomain, (bottom) full domain. [right column] Surrogates vs. ELCIRC in full space: (top) estuary, (middle) plume, (bottom) full domain.

simulation the total error seems to be dominated by the within subspace surrogate prediction error.

Fig. 9 shows the salinity field for the estuary region of the CORIE domain on days 14 (in development set) and 27 (end of test set). Clearly the accuracy of the SVD subspace projection and surrogate predictions in this part of the domain is much better than in the plume. This further corroborates the total error plots of Fig. 7, for which the error in the estuary is lower than the plume. The surrogate prediction errors on day 27 are much lower in the estuary than in the plume.

Fig. 10 show vertical transects of the salinity field starting outside the mouth of the Columbia and extending 25 km into the estuary and river system. The fidelity of the surrogates in capturing the salient features of the salt wedge entering the estuary is very good. On day 14 there are almost no surrogate prediction errors in the subspace and only some low level SVD reconstruction error. On day 27 the surrogate errors are again higher than during the first three weeks, but the general shape of the salt wedge is still quite well preserved. Most of the errors are being made on the interface boundary between the salt wedge and the surrounding fresher river water.

3.2. Experiment 2

For this experiment, both the *SUR*1 and *SUR*2 surrogates were trained using the *sliding window* N-fold cross-validation methodology with $N = 10$. Except for the difference in cross-validation and resulting regularization constants, all of the other experimental parameters where similar to Experiment 1. For this experiment we expect the regularization constants to be larger due to significantly less overlap-in-time for this cross-validation method.

For *SUR*1, the resulting optimal weight decay regularization constant is $\alpha = 4642$. Convergence occurred at around 50 iterations through the dataset. For *SUR*2, the resulting optimal weight decay regularization constant is $\alpha = 8333$. The network was again optimized to recursively predict the next three states, 1.5 h into the future. Convergence of the *SUR*2 network occurred at around 400 iterations through the dataset.

Fig. 11 shows the time-series traces of the predicted output (in the SVD subspace) as generated by *SUR*1 (dashed line) and *SUR*2 (thin solid line) compared to that of the ELCIRC simulation projected onto the SVD subspace (thick black line).

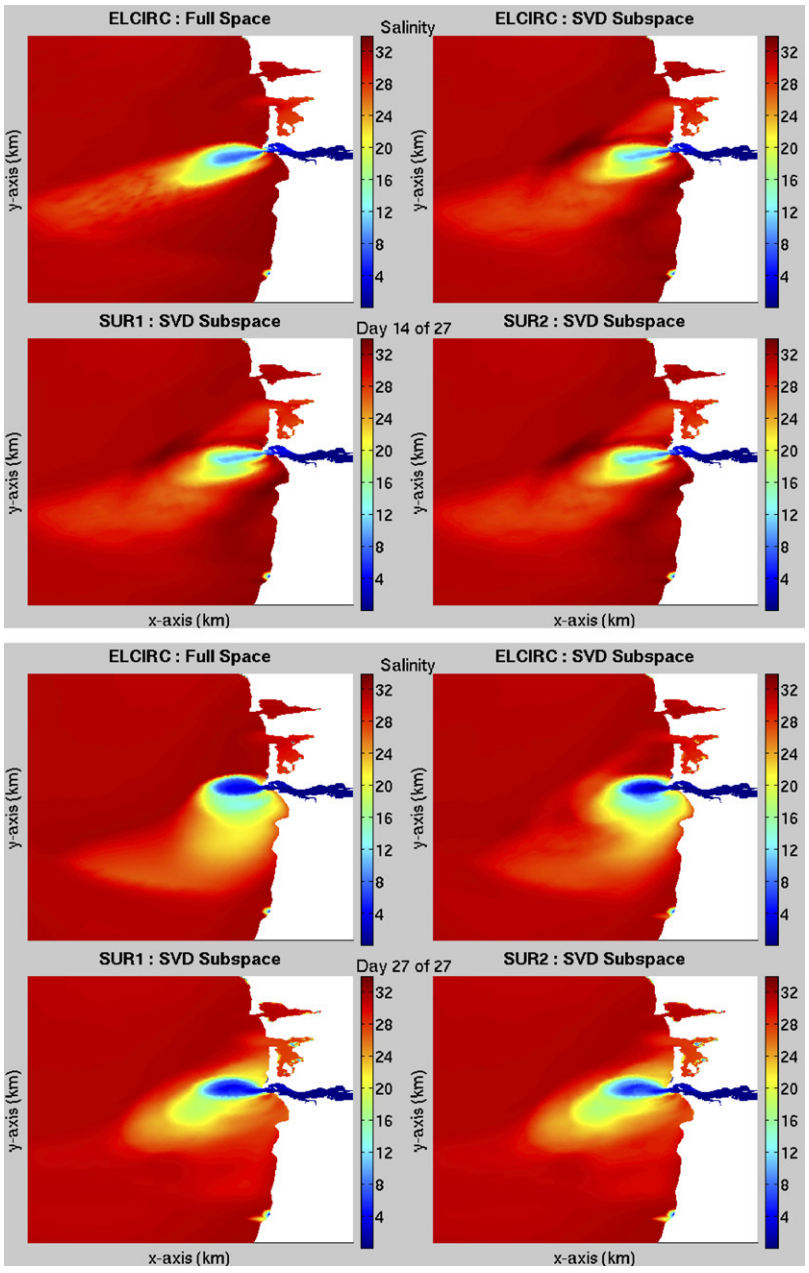

Fig. 8. Surface salinity field in the plume region of CORIE for day 14 (top four panels) and day 27 (bottom four panels) of the simulation.

Comparing this plot to the time-series plot for Experiment 1 (Fig. 6) one can clearly see that the higher frequency components are over regularized; they seem to simply predict the mean value of those components. Furthermore, the same diverging error behaviour within the test set region (days 21–27) seems to be present.

The total error plots for Experiment 2 (Fig. 12) differ substantially from those of Experiment 1. The errors in the development set period are higher, which is expected due to a larger regularization constant used during training. Unfortunately, the performance on the test set behaviour is only slightly better than was the case for Experiment 1, and we still see a slowly growing error (divergence) starting at day 21. For some regions in the development set (around days 5–15) there also seems to be quite large errors. This seems to indicate that overfitting (due to under-regularization) in Experiment 1 was not the only reason for the growing errors in the test set. The nonstationarity of the forcings seems to be playing a significant role in causing this divergent behaviour.

What is evident from these plots is that *SUR*2 outperforms *SUR*1 during days 1–21, with a slightly lower improvement during the test set period. The difference between the two differently-trained surrogates seems to be more pronounced for the cross-validation method used in Experiment 2 than was the case for Experiment 1. The recurrently trained neural networks exhibit more fidelity here. We might expect a performance improvement with longer

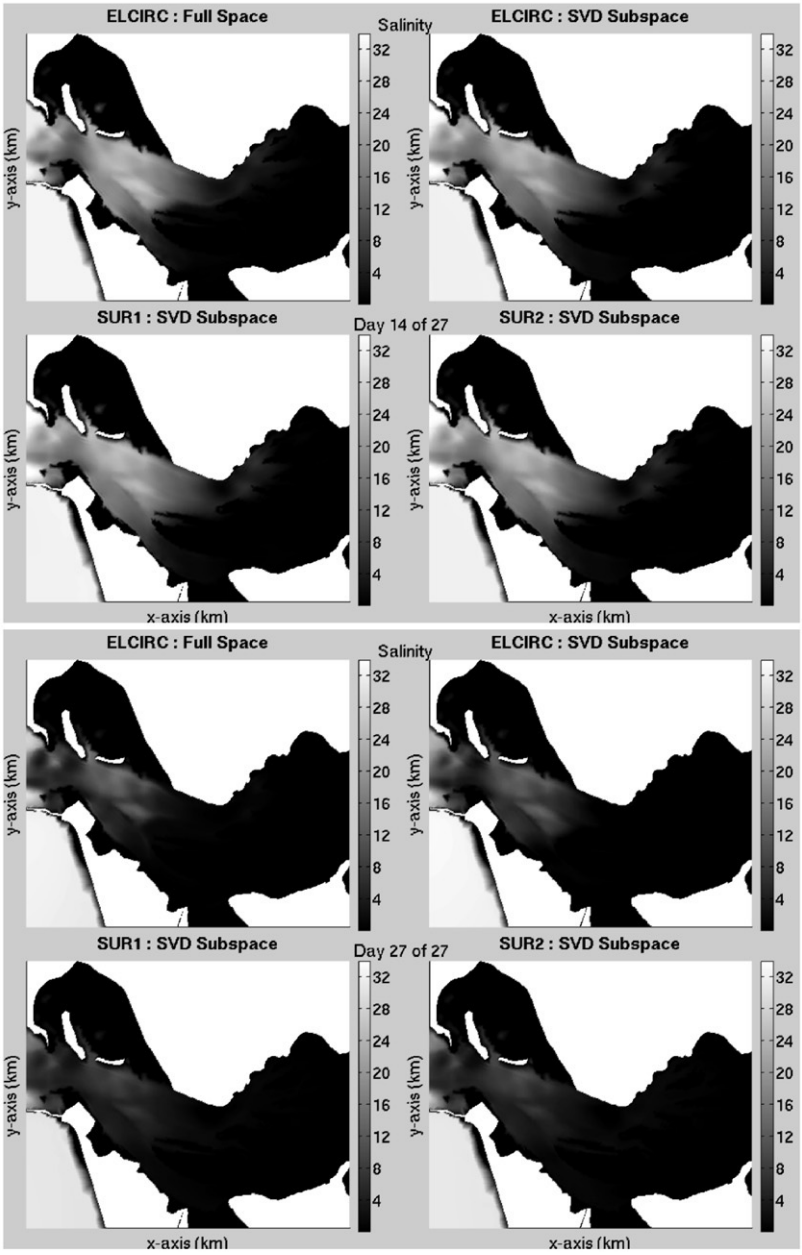

Fig. 9. Salinity field in the estuary region of CORIE for day 14 (top four panels) and day 27 (bottom four panels) of the simulation.

prediction horizons (at a significant cost in training time). However, our preliminary investigation shows that the test set performance does not improve significantly for longer prediction horizons.

In general, there are not significant qualitative differences between the large-scale structure of the predicted fields generated by the surrogates trained using the cross-validation method of Experiment 1 (*random shuffle*) versus that of Experiment 2 (*sliding window*). This is surprising since the subspace coefficient time-series predictions (Figs. 6 and 11) differ substantially. The high frequency components, which the over-regularized surrogates predict poorly, are not playing a significant role in determining the large-scale structure of the state. Presumably they influence detail on smaller spatial scales, though we have not investigated that in detail.

3.3. Computational acceleration

Simulating four weeks of the full CORIE domain at 30 min intervals using ELCIRC on a single processor Pentium-4 2.4 GHz PC with 2 Gb of memory takes 17,196 min. Computing the same simulation using our surrogates, *and* re-embedding the predicted subspace state into the full domain takes 17 min on the same computer. The surrogates provide an acceleration factor of 1000.

The situation is actually better for applications (such as our data assimilation system) where most of the computation is

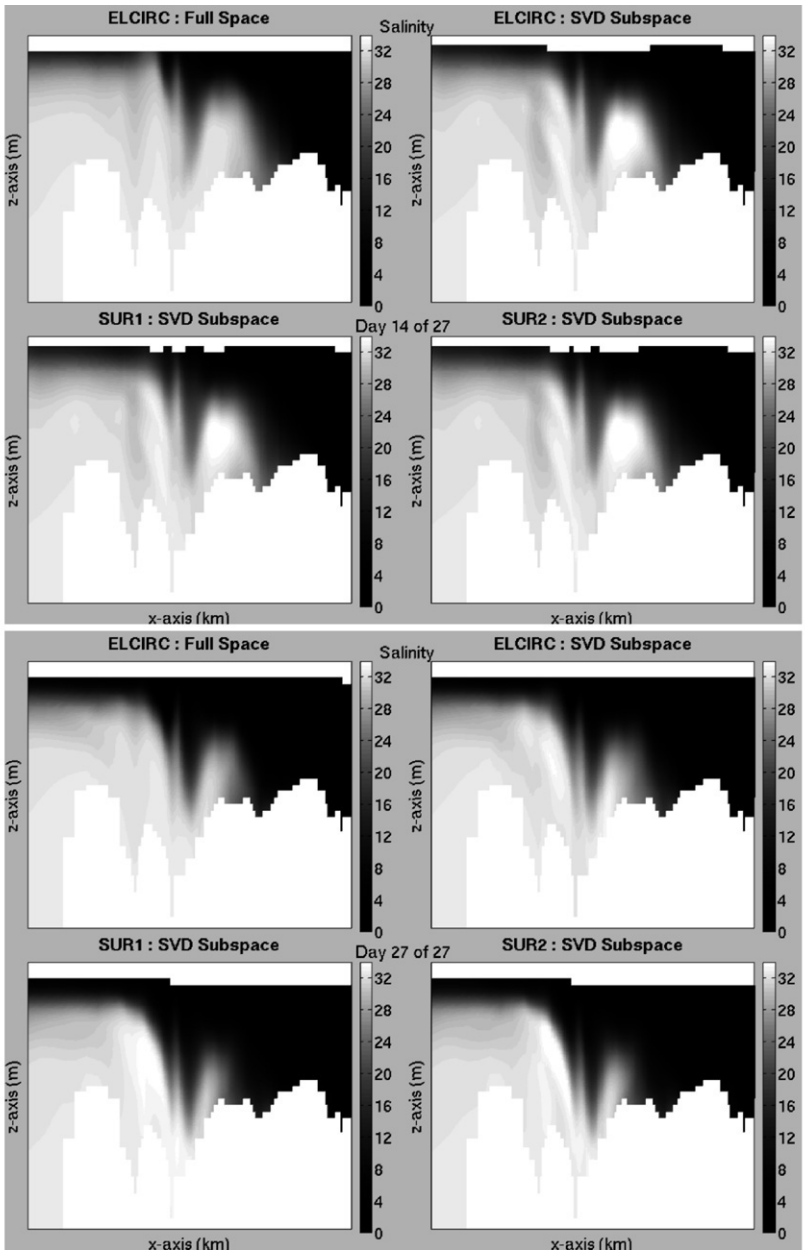

Fig. 10. Salinity field transect on day 14 (top four panels) and day 27 (bottom four panels).

done in the subspace. Of the 17 min used by the surrogate system, only 0.7% of the time is spent calculating the next state of the system in the subspace using the neural network, the remaining 99.3% of the time is spent performing the state re-embedding into the full space. This last operation is given by Eq. (19).

4. Discussion

Our experiments demonstrate that neural network (vector) time-series predictors are viable surrogates for extremely large-scale, highly nonlinear numerical circulation codes with tens of millions of dynamic degrees of freedom, and highly variable forcings. A simple architecture combining a dimensionality-reduction front-end with networks trained by standard back-propagation, or back-propagation-through-time provides a surrogate for the circulation codes that is relatively straightforward to train. The complete system that predicts the dynamics in the dimension-reduced subspace, *and* embeds those dynamics back into the full space yields an acceleration of roughly 1000 relative to the circulation code. Prediction in the subspace, *without* the re-embedding executes roughly 12,000 times faster than the full circulation code.

The application motivating this work was the construction of a data-assimilation (state-estimation) system for the Columbia River estuary. The flow in this domain *requires* the use of a nonlinear forced dynamic model, since linear models

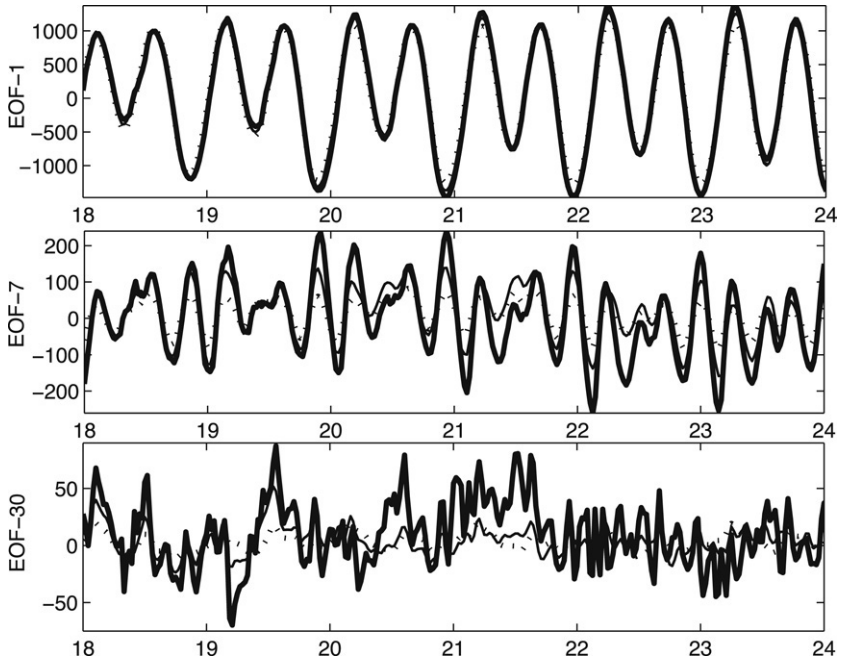

Fig. 11. Experiment 2: Surrogate time-series predictions of three of the 30 SVD subspace components. (bold line) ELCIRC in SVD subspace, $(\cdots)$ *SUR*1 feedback iterative prediction, $(-)$ *SUR*2 feedback iterative prediction.

trained to mimic the circulation code simulations are unstable. Neural networks have long been used to emulate nonlinear dynamic systems; and our application to estuary circulation codes extends their domain of practice to extremely large-scale, forced systems.

The computational acceleration ($\approx$1000 in this example) provided by this technology enables new applications of large-scale, nonlinear circulation codes. Coupling of our surrogates to sigma-point Kalman filters (van der Merwe, 2004) has provided the *first* data assimilation system capable of improving numerical model predictions in a river estuary system (Frolov et al., submitted for publication; Frolov, Baptista et al., 2006; Frolov, van der Merwe et al., 2006; Lu et al., 2007). The ability to execute *thousands* of simulations instead of a handful (Gneiting & Raftery, 2005) to build probabilistic forecasts provides the potential for significant change in the extent and frequency of use, as well as the accuracy of ensemble predictions.

The river, estuary, near-ocean system is a highly dynamic environment with significant nonlinearity and strong nonstationarity in forcings. Some of forcing variability has been discussed in conjunction with Fig. 5. More generally, the river flux from water released at Bonneville dam varies widely over hours and days. The spring melt increases the river flux dramatically. There are large seasonal temperature changes in the fresh water components. Winds force strong upwelling and downwelling in the near-ocean region.

The nonstationarity challenges surrogate construction. Nonstationarity in the forcings limits the data available for cross-validation. What is required are circulation code examples that cover a range of forcings similar to that encountered in the end application. For linear systems, one ensures such coverage by matching the power spectra of forcings between the training and application (test) contexts. For nonlinear systems, specifying forcing coverage is an open issue without substantial theoretical grounding. In the absence of strong general theoretical guidance, we are left to reason about and construct ensembles of forcings based on detailed domain knowledge. Strong nonstationarity may require different surrogate models in different regimes. It should be straightforward to adopt mixture-of-expert approaches, (Jacobs et al., 1991, for example,) for this domain.

Finally, our dimensionality reduction is a simple global SVD, and the results in Section 3 suggest that this is not adequate for accurate reproduction of the plume dynamics. More recent experiments associated with the data assimilation application (Frolov, Baptista et al., 2006; Frolov, van der Merwe et al., 2006) show that separate SVD on the estuary and plume regions improve the representation of the plume shape and dynamics. It is commonplace for *nonlinear* dimensionality reduction to provide significantly more compact representations than the principal component analysis (DeMers & Cottrell, 1993; Kambhatla & Leen, 1997; Kramer, 1991), and we expect that ultimately such techniques will improve the computational efficacy of this technology.

Acknowledgments

This work was supported by NSF grant OCI-0121475. The authors thank Joseph Zhang and Eric Wan for helpful discussions.

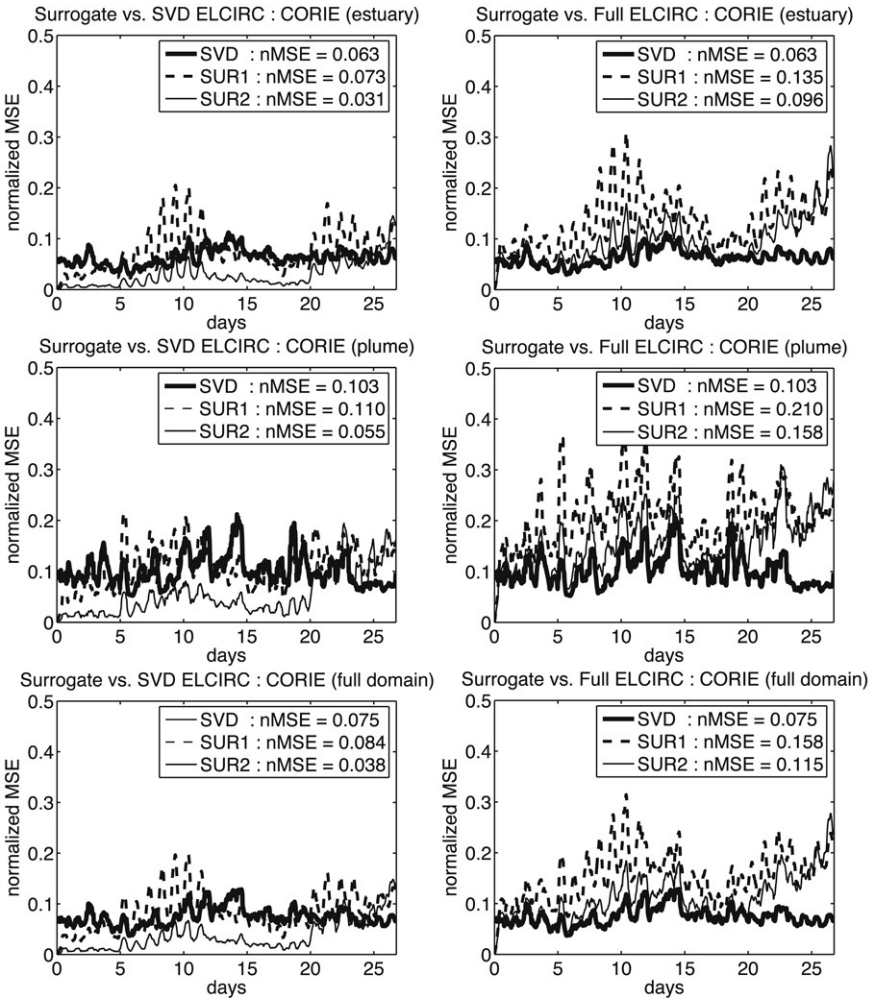

Fig. 12. Experiment 2: Normalized errors — *SVD* is the SVD reconstruction error, *SUR*1 is the feedforward network based surrogate error and *SUR*2 is the recurrent neural network- based surrogate error. [left column] Surrogates vs. ELCIRC in SVD subspace: (top) estuary subdomain, (middle) plume subdomain, (bottom) full domain. [right column] Surrogates vs. ELCIRC in full space: (top) estuary, (middle) plume, (bottom) full domain.

References

Bakker, R., Schouten, J. C., Giles, C. L., Takens, F., & van den Bleek, C. M. (2000). Learning chaotic attractors by neural networks. *Neural Computation*, *12*(10), 2355–2383.

Baptista, A. M., Wilkin, M., Pearson, P., Turner, P., McCandlish, C., & Barrett, P. (1999). Coastal and estuarine forecast systems: A multi-purpose infrastructure for the Columbia river. *Earth System Monitor*, *9*(3).

Baptista, A. M., Zhang, Y. L., Chawla, A., Zulauf, M., Seaton, C., Myers, E. P., et al. (2005). A cross-scale model for 3d baroclinic circulation in estuary-plume-shelf systems: II. Application to the Columbia river. *Continental Shelf Research*, *25*, 935–972.

Bennett, A. (2002). *Inverse modeling of the ocean and atmosphere*. Cambridge University Press.

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford University Press.

Bishop, C. M., Haynes, P. S., Smith, M. E., Todd, T. N., & Trotman, D. L. (1995). Real-time control of a tokamak plasma using neural networks. *Neural Computation*, *7*(1), 206–217.

Blumberg, A. F., & Mellor, G. L. (1987). A description of a three-dimensional coastal ocean circulation model. In N. S. Heaps (Ed.), *Three-dimensional ccoastal ocean models*: *Vol. 4* (pp. 1–16). Washington, DC: American Geophysical Union.

Caruana, R., Lawrence, S., & Giles, C. L. (2000). Overfitting in neural networks: Backpropagation, conjugate gradient, and early stopping. In *Advances in neural information processing systems: Vol. 13*. Denver, Colorado.

Casdagli, M. (1989). Nonlinear prediction of chaotic time series. *Physica D*, *35*, 335–355.

Chevallier, F., Cheruy, F., Scott, N. A., & Chedin, A. (1998). A neural network approach for a fast and accurate computation of longwave radiative budget. *Journal of Applied Meteorology*, *37*(11), 1385–1397.

DeMers, D., & Cottrell, G. (1993). Non-linear dimensionality reduction. In *Advances in neural information processing systems: Vol. 5*. San Mateo, CA.

Dietrich, G., Kalle, K., Krauss, W., & Siedler, G. (1980). *General oceanography: An introduction* (2nd ed.). New York: Wiley-Interscience.

Ferraro, R., Sato, T., Brasseur, G., DeLuca, C., & Guilyardi, E. (2003). Modeling the earth system. In *Proceedings of the international geoscience and remote sensing symposium*. Toulouse, France: IEEE.

Frolov, S., Baptista, A. M., Lu, Z., van der Merwe, R., & Leen, T. K. (2007). Fast data assimilation using a nonlinear Kalman filter and a model surrogate: An application to the Columbia river estuary. *Ocean Modeling* (submitted for publication).

Frolov, S., Baptista, A. M., Leen, T., Lu, Z., & van der Merwe, R. (2006). Assimilating in-situ measurements into a reduced-dimensionality model of an estuary-plume system. *Eos Transactions: AGU*, *87*(52), Fall Meeting Supplement, Abstract A31A-0846. http://purl.oclc.org/net/agùa31a-0846.

Frolov, S., van der Merwe, R., Lu, Z., Baptista, A. M., & Leen, T. (2006). Fast and model-independent data assimilation of estuarine circulation, using neural networks. *Eos Transactions: AGU*, *87*(36), Ocean Sciences Meeting Supplement, Abstract OS26O-06. http://purl.oclc.org/net/agu_os26o-06.

Gneiting, T., & Raftery, A. E. (2005). Weather forecasting with ensemble methods. *Science*, *310*, 248–249.

Golub, G. H., & van Loan, C. F. (1996). *Matrix computations* (3rd ed.). Baltimore, MD: Johns Hopkins University Press.

Grassberger, P., Schreiber, T., & Schaffrath, C. (1991). Nonlinear time sequence analysis. *International Journal on Bifurcation and Chaos*, *1*, 521–547.

Grzeszczuk, R., Terzopoulos, D., & Hinton, G. E. (1998). Fast neural network emulation of dynamical systems for computer animation. In *Advances in neural information processing systems 11* (pp. 882–888).

Jacobs, R., Jordan, M., Nowlan, S., & Hinton, G. (1991). Adaptive mixture of local experts. *Neural Computation*, *3*, 79–87.

Jay, D. A., & Flinchem, E. P. (1997). Interaction of fluctuating river flow with a barotropic tide: A demonstration of wavelet tidal analysis methods. *Journal of Geophysical Research*, *102*, 5705–5720.

Jollife, I. T. (1986). *Principal component analysis*. New York: Springer-Verlag.

Kambhatla, N., & Leen, T. K. (1997). Optimal dimension reduction by local PCA. *Neural Computation*, *9*, 1493–1516.

Kramer, M. A. (1991). Nonlinear prinipal component analysis using autoassociative neural networks. *AIChE Journal*, *37*(2), 233–243.

Krasnopolsky, V. M., Chalikov, D. V., & Tolman, H. L. (2002). A neural network technique to improve computational efficiency of numerical oceanic models. *Ocean Modelling*, *4*, 363–383.

Krasnopolsky, V. M., & Chevallier, F. (2003). Some neural network applications in environmental sciences. Part II: Advancing computational efficiency of environmental numerical models. *Neural Networks*, *16*, 335–348.

Lapedes, A., & Farber, R. (1988). How Neural Nets work. In D. Z. Anderson (Ed.), *Neural information processing systems* (pp. 442–456). New York: American Institute of Physics.

Li, S., Hsieh, W. W., & Wu, A. (2005). Hybrid coupled modeling of the tropical pacific using neural networks. *Journal of Geophysical Research*, *110*. doi:10.1029/2004JC002595.

Lu, Z., Leen, T. K., van der Merwe, R., Frolov, S., & Baptista, A. M. (2007). Sequential data assimilation with sigma-point Kalman filter on low-dimensional manifold. Technical report TR-07-001, NSF-STC for Coastal Margin Observation & Prediction. http://purl.oclc.org/NET/CMOP-TR-07-001.

Luettich, R. A., Westerink, J. J., & Scheffner, N. W. (1992). ADCIRC: An advanced three-dimensional circulation model for shelves coasts and estuaries, Report 1: Theory and methodology of ADCIRC-2DDI and ADCIRC-3DL. *Technical report DRP-92-6*. U.S. Army Engineers Waterways Experiment Station.

Lynch, D. R., Ip, J. T. C., Naimie, C. E., & Werner, F. E. (1996). Comprehensive coastal circulation model with application to the gulf of maine. *Continental Shelf Research*, *16*(7), 875–906.

Møller, M. (1996). A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks*, *6*(4), 525–533.

Moody, J. (1992). The effective number of parameters: An analysis of generalization and regularization in nonlinear learning systems. In *Advances in neural information processing systems*: *Vol. 4* (pp. 847–854). CA: Palo Alto.

Murata, N., Yoshizawa, S., & Amari, S. (1994). Network information criterion - determining the number of hidden units for artificial neural network models. *IEEE Transactions on Neural Networks*, *5*, 865–872.

Principe, J. C., Rathie, A., & Kuo, J. M. (1992). Prediction of chaotic time series with neural networks and the issue of dynamic modeling. *International Journal of Bifurcation and Chaos*, *2*(4), 989–996.

Shchepetkin, A., & McWilliams, J. (2005). The regional ocean modeling system: A split-explicit, free-surface, topography following coordinates ocean model. *Ocean Modelling*, *9*, 347–404.

Su, H. -T., McAvoy, T. J., & Werbos, P. (1992). Long-term predictions of chemical processes using recurrent neural networks: A parallel training approach. *Industrial & Engineering Chemistry Research*, *31*, 1338–1352.

Takens, F. (1981). Detecting strange attractors in turbulence. In D. A. Rand, & L. -S. Young (Eds.), *Lecture Notes in Mathematics*: *Vol. 898*. *Dynamical systems and turbulence* (pp. 366–381). Berlin: Springer-Verlag.

Tang, Y., & Hsieh, W. W. (2003). ENSO simulation and prediction in a hybrid coupled model with data assimilation. *Journal of Meteorological Society of Japan*, *81*(1), 1–19.

Tolman, H. L., Krasnopolsky, V. M., & Chalikov, D. V. (2005). Neural network approximation for nonlinear interactions in wind wave spectra: Direct mapping for wind seas in deep water. *Ocean Modelling*, *8*(3), 253–278.

van der Merwe, R. (2004). Sigma-point kalman filters for probabilistic inference in dynamic state-space models. *Ph.D. thesis*. OGI School of Science & Engineering, Oregon Health & Science University, Portland, OR.

Weigend, A. S., & Gershenfeld, N. A. (Eds.). (1994). *Time series prediction: Forecasting the future and understanding the past*. Reading, MA: Addison-Wesley.

Werbos, P. J. (1990). Backpropagation through time: What it does and how to do it. *Proceedings of the IEEE*, *78*(10), 1550–1560.

Zhang, Y. L., Baptista, A. M., & Myers, E. P. (2004). A cross-scale model for 3d baroclinic circulation in estuary-plume-shelf systems: I. Formulation and skill assessment. *Continental Shelf Research*, *24*, 2187–2214.