Neurocomputing 692 (2026) 133849

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Cross-scale reservoir computing for large spatio-temporal forecasting and modeling

Nicola Alboré [a, b, c, 1], Gabriele Di Antonio [a, b, d, f, 1, *], Fabrizio Coccetti [a], Andrea Gabrielli [a, e]

[a] Centro Ricerche Enrico Fermi (CREF), Rome, Italy
[b] National Center for Radiation Protection and Computational Physics, Istituto Superiore di Sanità, Rome, Italy
[c] Department of Physics, University of Rome Tor Vergata, Rome, Italy
[d] IRCCS Santa Lucia Foundation, Rome, Italy
[e] Department of Civil, Computer Science and Aeronautical Technologies Engineering, Roma Tre University, Rome, Italy
[f] Universitat Pompeu Fabra, Barcelona, Spain

ARTICLE INFO

Communicated by M. Bianchini

Keywords:
Reservoir computing
Multiscale modeling
Spatiotemporal forecasting
Climate dynamics

ABSTRACT

We propose a new reservoir computing method for forecasting high-resolution spatiotemporal datasets. While many machine learning methods suffer from limited interpretability and computational intractability in high-dimensional settings, reservoir computing offers inherent advantages through its lightweight linear training. Our hierarchical architecture combines multi-resolution inputs from coarser to finer layers, enabling the capture of both local dynamics and long-range dependencies. Applied to Sea Surface Temperature data, it outperforms standard parallel reservoir models in long-term forecasting, demonstrating the effectiveness of cross-layer coupling in improving predictive accuracy. We find that the optimal network dynamics in each layer become increasingly linear, revealing the slow modes propagated to subsequent layers. Finally, we evaluate the method on a chaotic system, where a strongly coupled two-layer configuration achieves accurate forecasts and offers favorable accuracy-cost tradeoffs compared with established reservoir baselines.

Complex spatio-temporal systems govern phenomena across multiple scientific domains, from atmospheric dynamics to plasma physics. These systems universally exhibit behavior across diverse timescales, posing fundamental challenges for modeling and forecasting. Although conventional approaches rely on detailed knowledge of the governing equations [4,9,14], recent advances in data-driven modeling offer promising alternatives for systems where such knowledge remains incomplete or computationally intractable [26,27]. Yet these methods often suffer from elusive underlying mechanisms and incur prohibitively high training costs, particularly when dealing with high-dimensional datasets.

Reservoir computing (RC) has emerged as a powerful paradigm for modeling nonlinear dynamical systems, demonstrating remarkable efficacy in forecasting chaotic behavior with minimal computational overhead [13,20]. Standard implementations, however, typically struggle to simultaneously capture dynamics across disparate timescales, particularly when predicting systems with significant separation between fast and slow processes. In recent years, new reservoir architectures have been proposed to address the limitations of the classical formulation [11,12,16,21,22,29]. In this paper, we turn our attention to the analysis of large-scale spatiotemporal datasets, where the approach of [22] remains the key reference. In that work, and in related studies, the domain is partitioned into grids of locally interacting reservoirs [2,22]. However, these designs do not account for the long-range dependencies essential for long-term forecasts. Inspired by physics-based treatments of long-range interactions in large N-body systems [3], our architecture uses lower resolution layers to inform higher resolution reservoirs about long distances (Fig. 1).

Application on Sea Surface Temperature (SST) dynamics demonstrates improved long-term reproduction compared to single-layer models (Fig. 2). This result comes mainly from the cross-scale underlying structure. Low-resolution levels filter out fast modes and emphasize slow, coherent dynamics, helping the higher detailed layers to better dissect such information.

We demonstrate that the optimal RNN dynamics in each layer become increasingly linear as the resolution of the corresponding layer decreases. This results in an interpretable decomposition of the dynamical

* Corresponding author at: Universitat Pompeu Fabra, Barcelona, Spain.
Email addresses: nicola.albore@cref.it (N. Alboré), gabriele.diantonio@upf.edu (G. Di Antonio).

[1] These authors contributed equally to this work.

https://doi.org/10.1016/j.neucom.2026.133849
Received 24 October 2025; Received in revised form 22 April 2026; Accepted 2 May 2026
Available online 2 May 2026

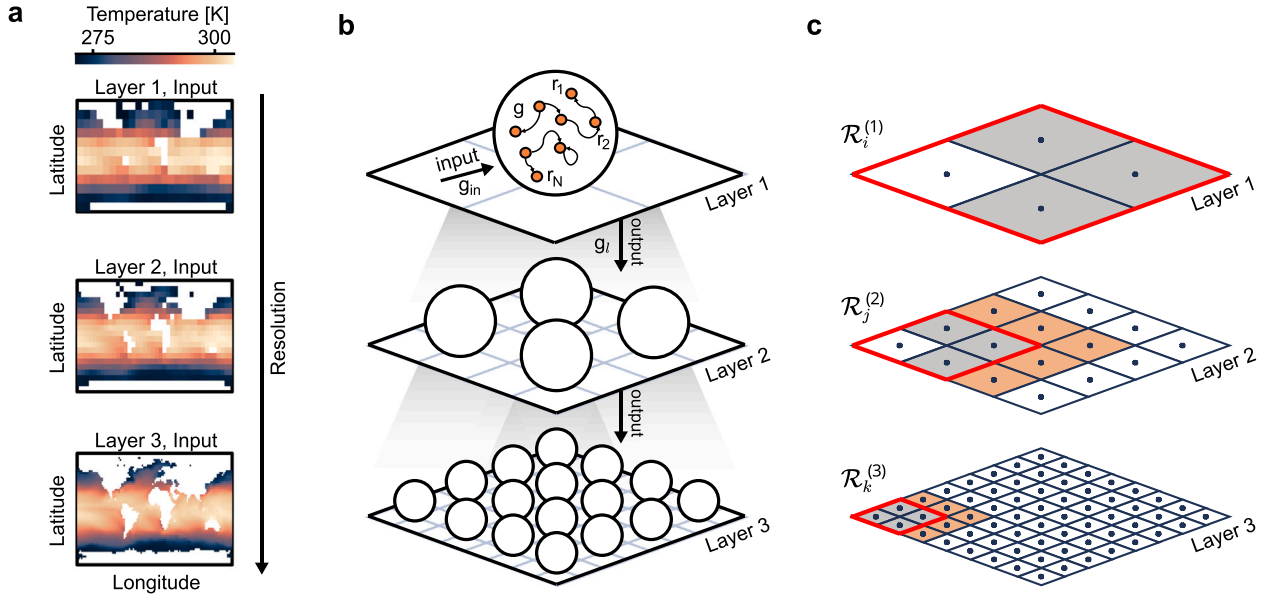

Fig. 1. a, A dataset is organized into different resolution levels, forming a hierarchical architecture. **b,** Sketch of a three-layer cross-scale reservoir computing. Each resolution level comprises a two-dimensional spatial domain represented as a grid of coordinates, each corresponding to an independent time series of a dynamical variable. The domain is partitioned into local subregions each serving as the learning target for a distinct reservoir network. Information from distant regions is passed through the previous layer at a lower resolution. In particular, looking at panel **c**, in each layer, the highlighted reservoir $\mathcal{R}$ (superscript indicates layer hierarchy) receives additional input from the output of the corresponding reservoir in the layer above (red contour), excluding contributions from the region that overlaps directly with its own coordinates (white area). This ensures that only non-local coarse-grained dynamics are passed downward, promoting hierarchical context without redundancy. The orange area highlights the neighboring coordinates which are included as additional input features to incorporate local spatial context as in [22].

system into linearly evolving modes, revealing the slow modes propagated to subsequent layers.

Finally, we assess computational efficiency on a canonical spatiotemporally chaotic system (a forced Kuramoto–Sivashinsky equation): in a strong cross-scale coupling regime, performance is largely controlled by the coarse layer, enabling accurate predictions up to 8 Lyapunov times with reduced computational cost relative to standard reservoir baselines (Fig. 4).

1. A multi-scale reservoir computing structure

1.1. Reservoir computing framework

We begin by describing the basic formulation of reservoir computing as used in this study. The framework consists of a high-dimensional, nonlinear dynamical system, namely the reservoir $\mathcal{R}$ which is driven by a lower-dimensional input signal. Let $\mathbf{u}(t) \in \mathbb{R}^{D_{\text{in}}}$ denote the input vector at time t, and let $\mathbf{r}(t) \in \mathbb{R}^{D_r}$ be the internal state of the reservoir. The evolution of the reservoir state is governed by an ODE, in which the force field depends nonlinearly on the current state $\mathbf{r}(t)$ and the input $\mathbf{u}(t)$ via

$$\tau \frac{d\mathbf{r}(t)}{dt} = \tanh\left[\mathbf{W}\mathbf{r}(t) + \mathbf{W}_{\text{in}}\mathbf{u}(t)\right] - \mathbf{r}(t), \tag{1}$$

where $\tau > 0$ is a time constant and $\tanh(\cdot)$ is applied element-wise. The input $\mathbf{u}(t) \in \mathbb{R}^{D_{\text{in}}}$ is mapped into the reservoir via $\mathbf{W}_{\text{in}} \in \mathbb{R}^{D_r \times D_{\text{in}}}$, which is typically drawn at random once and then kept fixed during both training and prediction. The recurrent connectivity $\mathbf{W} \in \mathbb{R}^{D_r \times D_r}$ is commonly initialized as a sparse random matrix with sparsity p, with nonzero entries sampled from a zero-mean Gaussian distribution and then rescaled to attain a prescribed spectral radius g.

If the reservoir dynamics settle into a subspace where responder states $\mathbf{r}(t)$ are constrained by the driver $\mathbf{u}(t)$, a synchronization map links the two. Extra dimensions in $\mathbf{r}(t)$ may encode redundant or delayed information, preserving an echo of past information [13]. This embedding is used to generate an output $\mathbf{v}(t) \in \mathbb{R}^{D_{\text{out}}}$ via a linear readout

$$\mathbf{v}(t) = \mathbf{W}_{\text{out}}\,\boldsymbol{\psi}(\mathbf{r}(t)), \tag{2}$$

where $\mathbf{W}_{\text{out}} \in \mathbb{R}^{D_{\text{out}} \times D_\psi}$ is the matrix of trainable output weights and $\boldsymbol{\psi}(\mathbf{r})$ is a fixed set of features, often simply linear $\boldsymbol{\psi}(\mathbf{r}) = \mathbf{r}$, but quadratic or higher-order extensions are commonly used. Training records reservoir states $\mathbf{r}(t)$ for $t \in [-T, 0]$ under inputs $\mathbf{u}(t)$ and targets $\hat{\mathbf{v}}(t)$. The weights $\mathbf{W}_{\text{out}}$ are then chosen to minimize the mean-squared error between $\mathbf{v}(t)$ and $\hat{\mathbf{v}}(t)$ across the training interval, typically using ridge regression. After training, the reservoir can be used for prediction. If the target is the driver itself, the dynamics describes an autonomous system. The input $\mathbf{u}(t)$ in Eq. (1) is replaced by the output $\mathbf{v}(t)$ of Eq. (2), giving

$$\tau \frac{d\mathbf{r}(t)}{dt} = \tanh\left[\mathbf{W}\mathbf{r}(t) + \mathbf{W}_{\text{in}}\mathbf{W}_{\text{out}}\,\boldsymbol{\psi}(\mathbf{r}(t))\right] - \mathbf{r}(t), \tag{3}$$

which then generates predictions at each timestep $t > 0$ via the mapping in Eq. (2).

1.2. Parallel reservoir computing

When applying reservoir computing to systems characterized by a large spatial domain, directly feeding the full input vector $\mathbf{u}(t) \in \mathbb{R}^{D_{\text{in}}}$ into a single reservoir becomes impractical. The number of connections required to maintain expressive internal dynamics scales unfavorably with input dimensionality, leading to increased computational cost and degraded performance due to overloading the reservoir. To address this limitation, Pathak et al. [22] suggested employing a parallel architecture in which the input space is partitioned into n spatially localized subdomains, each assigned to an independent reservoir.

Formally, let the spatial domain be covered by pairwise disjoint central supports $\{S_i\}_{i=1}^n$ with $S_i \cap S_j = \emptyset$ for $i \neq j$ and $\bigcup_{i=1}^n S_i$ covering the domain. For each i, define an enlarged overlap neighborhood $\Omega_i \supseteq S_i$ (e.g., S_i augmented by a symmetric halo of fixed width). Reservoir $\mathcal{R}^{(i)}$ is responsible for predicting the dynamics on S_i, but it is driven by the restriction of the full input field to Ω_i.

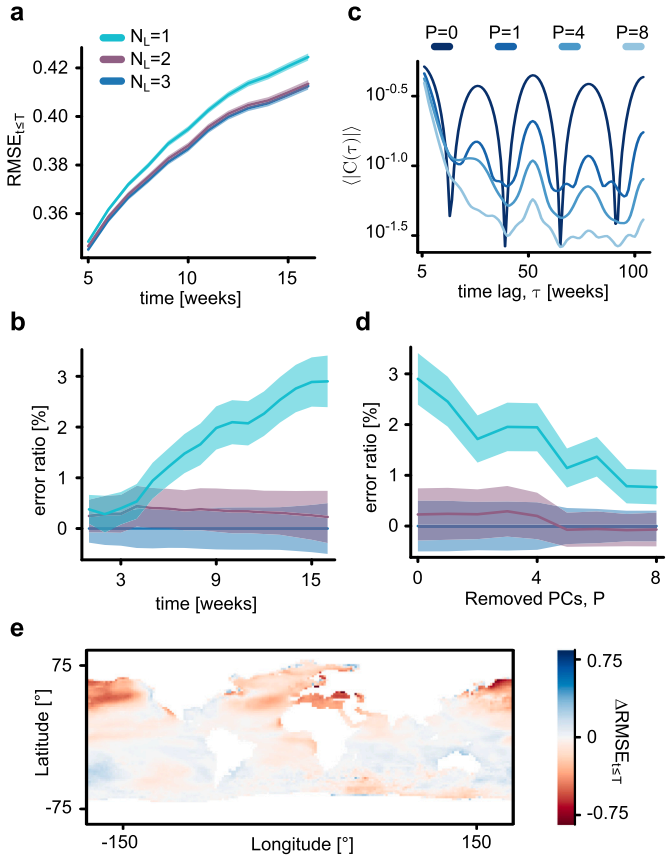

Fig. 2. a, Average total error ($\mathrm{RMSE}_{t\le T}$) as a function of time T for different numbers of layers, N_L. **b,** Ratio of the $\mathrm{RMSE}_{t\le T}$ for various N_L to the reference case with $N_L = 3$, shown as a function of time. **c,** Absolute autocorrelation of the dataset versus time lag, averaged over all spatial locations, for different numbers of removed principal components, P. **d,** Ratio of the $\mathrm{RMSE}_{t\le T}$ at a fixed time of 10 weeks, plotted as a function of P. **e,** Difference in $\mathrm{RMSE}_{t\le T}$ between $N_L = 3$ and $N_L = 1$ for individual geographical locations at $T = 50$ weeks, averaged over 25 runs. In all panels, solid lines denote the mean and shaded areas indicate the SEM, computed over 30 independent runs (unless otherwise stated).

We encode this restriction using a binary selection matrix $\mathbf{P}_i \in \{0,1\}^{d_i \times D_{\text{in}}}$ with $d_i := |\Omega_i|$, and define

$$\mathbf{u}_{\Omega_i}(t) := \mathbf{P}_i\, \mathbf{u}(t) \in \mathbb{R}^{d_i}.$$

Spatial continuity is promoted by overlap: for neighboring subdomains, $\Omega_i \cap \Omega_j \neq \emptyset$, so that $\mathbf{u}_{\Omega_i}(t)$ and $\mathbf{u}_{\Omega_j}(t)$ share components of $\mathbf{u}(t)$. Given the local input, reservoir i evolves an internal state $\mathbf{r}_i(t) \in \mathbb{R}^{D_r^{(i)}}$ according to Eq. (1) (with $\mathbf{u}(t)$ replaced by $\mathbf{u}_{\Omega_i}(t)$). All reservoirs run in parallel with fixed internal weights; only the readouts are trained (typically independently for each i).

Reservoir i produces a local prediction on its central support, e.g.,

$$\hat{\mathbf{v}}_{S_i}(t) = \mathbf{W}_{\text{out}}^{(i)}\, \boldsymbol{\psi}\big(\mathbf{r}_i(t)\big).$$

Finally, the global prediction is obtained by assembling the local readouts on the (disjoint) central supports:

$$\hat{\mathbf{v}}(t) = \sum_{j=1}^{n} \mathbf{E}_j\, \hat{\mathbf{v}}_{S_j}(t),$$

where $\mathbf{E}_j \in \{0,1\}^{D_{\text{in}} \times |S_j|}$ injects the j-th local prediction into the full field. In closed-loop mode each reservoir is driven by this reconstructed field, hence

$$\tau\, \frac{d\mathbf{r}_i(t)}{dt} = \tanh\Big[\mathbf{W}^{(i)}\mathbf{r}_i(t) + \sum_{j=1}^{n} \mathbf{W}^{(ij)}\, \boldsymbol{\psi}\big(\mathbf{r}_j(t)\big)\Big] - \mathbf{r}_i(t), \tag{4}$$

where $\mathbf{W}^{(ij)} := \mathbf{W}_{\text{in}}^{(i)}\, \mathbf{P}_i\, \mathbf{E}_j\, \mathbf{W}_{\text{out}}^{(j)}$ describes the interactions between the local reservoirs induced by overlapping spatial tiles.

1.3. Cross-scale reservoir computing

To incorporate information across disparate spatial scales, we introduce a hierarchical multi-layer architecture, named Cross-Scale Reservoir Computing (CS-RC) that represents the same dynamical system on a family of grids with increasing resolution. We index resolution levels by $\ell \in \{1, \dots, L\}$, from the coarsest $\ell = 1$ to the finest $\ell = L$, and denote by $\mathbf{u}^{(\ell)}(t) \in \mathbb{R}^{D_{\text{in}}^{(\ell)}}$ the input field regridded to level ℓ (Fig. 1a and b). In our implementation, regridding is performed by spatial averaging over increasingly large cells, so that upper (coarser) levels retain only dominant large-scale structures while finer levels preserve small-scale variability.

Within-layer parallel reservoirs. At each fixed level ℓ, we apply the parallel reservoir construction of Section 1.2. Concretely, the level-ℓ spatial domain is partitioned into pairwise disjoint central supports $\{S_i^{(\ell)}\}_{i=1}^{n^{(\ell)}}$ and associated overlap neighborhoods $\Omega_i^{(\ell)} \supseteq S_i^{(\ell)}$. Reservoir $\mathcal{R}_i^{(\ell)}$ is driven by the restriction of the level-ℓ input to $\Omega_i^{(\ell)}$, encoded via a binary selection matrix

$$\mathbf{u}_{\Omega_i}^{(\ell)}(t) = \mathbf{P}_i^{(\ell)}\mathbf{u}^{(\ell)}(t), \qquad \mathbf{P}_i^{(\ell)} \in \{0,1\}^{|\Omega_i^{(\ell)}| \times D_{\text{in}}^{(\ell)}}.$$

As in Section 1.2, overlap between neighboring $\Omega_i^{(\ell)}$ promotes spatial continuity within the same resolution level.

Cross-scale (coarse-to-fine) coupling. Parallel reservoirs at a single resolution have limited access to long-range dependencies because their inputs remain spatially local. To propagate non-local information, we couple adjacent resolutions by augmenting each reservoir input at level $\ell \ge 2$ with a coarse signal extracted from level $\ell - 1$.

For each reservoir $\mathcal{R}_i^{(\ell)}$, let $\Pi_i^{(\ell)}$ denote the set of indices on the level-$(\ell-1)$ grid corresponding to the unique coarse cell that contains the central support $S_i^{(\ell)}$ under the regridding map. To avoid feeding back information that is already directly available at level ℓ (through $u_{\Omega_i}^{(\ell)}$), we exclude the child central support from the parent signal defining the non-overlapping region

$$\Gamma_i^{(\ell)} := \Pi_i^{(\ell)} \setminus S_i^{(\ell)},$$

as depicted by the gray cells of the parent layer in Fig. 1(c). We extract the corresponding coarse signals using a binary selection matrix $\mathbf{B}_i^{(\ell-1\to\ell)} \in \{0,1\}^{|\Gamma_i^{(\ell)}| \times D_{\text{in}}^{(\ell-1)}}$ and define the non-overlapping input

$$\mathbf{u}_{\Gamma_i}^{(\ell-1\to\ell)}(t) := \mathbf{B}_i^{(\ell-1\to\ell)}\, \mathbf{u}^{(\ell-1)}(t) \in \mathbb{R}^{|\Gamma_i^{(\ell)}|}.$$

where $\mathbf{u}^{(\ell-1)}(t)$ denotes the coarse signal made available by level $\ell - 1$ (externally driven during training, and replaced by the level-$(\ell-1)$ prediction during autonomous rollout).

The resulting augmented input to reservoir $\mathcal{R}_i^{(\ell)}$ is the concatenation

$$\tilde{\mathbf{u}}_i^{(\ell)}(t) := \big[\mathbf{u}_{\Omega_i}^{(\ell)}(t);\ \mathbf{u}_{\Gamma_i}^{(\ell-1\to\ell)}(t)\big],$$

which yields the (input-driven) reservoir dynamics

$$\tau_\ell\, \frac{d\mathbf{r}_i^{(\ell)}(t)}{dt} = \tanh\Big(\mathbf{W}_i^{(\ell)}\, \mathbf{r}_i^{(\ell)}(t) + \mathbf{W}_{\text{in},i}^{(\ell)}\, \tilde{\mathbf{u}}_i^{(\ell)}(t)\Big) - \mathbf{r}_i^{(\ell)}(t). \tag{5}$$

In this way, each fine-scale reservoir is driven by both (i) local within-layer information and (ii) non-local coarse input from its parent region, while explicitly excluding self-information that would otherwise be duplicated.

Training and inference. The architecture is trained top-down: we first train the coarsest layer $\ell = 1$, then proceed sequentially to finer levels while keeping all previously trained layers fixed. At level $\ell \geq 2$, training uses the augmented inputs $\widetilde{\mathbf{u}}_i^{(\ell)}(t)$, where the coarse contextual term $\mathbf{u}_{\Gamma_i}^{(\ell)}(t)$ is generated by the already-trained level $\ell - 1$ predictor. During prediction, the model is evaluated coarse-to-fine at each time step: we compute $\mathbf{v}^{(1)}(t)$, then recursively obtain $\mathbf{v}^{(\ell)}(t)$ using both within-layer mixed inputs and the coarser signal passed down from level $\ell-1$, filtered by the binary selection matrices.

Computational complexity and storage requirements. To clarify the scalability of CS-RC, we characterize both its computational complexity, i.e., the number of required operations, and its storage requirements, i.e., the number of coefficients that must be stored.

Consider a reservoir $\mathcal{R}_i^{(\ell)}$ at layer ℓ, with state dimension $D_r^{(\ell)}$, local input dimension $|\Omega_i^{(\ell)}|$, cross-scale input dimension $|\Gamma_i^{(\ell)}|$, and readout feature dimension $D_\psi^{(\ell)} := \dim[\boldsymbol{\psi}(\mathbf{r}_i^{(\ell)})]$. We assume that all reservoirs within the same layer have the same number of recurrent nodes.

Let $s^{(\ell)} := \mathrm{nnz}(\mathbf{W}^{(\ell)})$ denote the number of nonzero entries of the recurrent matrix at layer ℓ. Since the reservoir matrices are sparse, one evaluation of the force field in Eq. (5) has a cost

$$C_{\mathrm{step},i}^{(\ell)} = O\Big(s^{(\ell)} + D_r^{(\ell)}\big(|\Omega_i^{(\ell)}| + |\Gamma_i^{(\ell)}|\big)\Big),$$

where the first term is the recurrent sparse matrix-vector multiplication and the second is the input contribution. Under a fixed average degree, it holds $s^{(\ell)} = O(D_r^{(\ell)})$; therefore, up to the input term, the state-update cost is linear in the reservoir size.

Summing over all reservoirs and layers gives the total single step cost

$$C_{\mathrm{step,train}} = O\left(\sum_{\ell=1}^{L}\sum_{i=1}^{n^{(\ell)}}\Big[s^{(\ell)} + D_r^{(\ell)}\big(|\Omega_i^{(\ell)}| + |\Gamma_i^{(\ell)}|\big)\Big]\right),$$

so it scales approximately linearly with the number of reservoirs and the number of layers used. With explicit Euler integration, the total cost of simulating the dynamics over T training steps is $T\, C_{\mathrm{step,train}}$.

During training, the inputs are externally prescribed. In autonomous operation, by contrast, one must additionally evaluate the local output $\mathbf{W}_{\mathrm{out},i}^{(\ell)}\boldsymbol{\psi}(\mathbf{r}_i^{(\ell)})$. If $m_i^{(\ell)}$ denotes the dimension of the local prediction over the central support, then the readout evaluation costs $O(m_i^{(\ell)} D_\psi^{(\ell)})$. The total per-step test cost is therefore

$$C_{\mathrm{step,test}} = C_{\mathrm{step,train}} + O\left(\sum_{\ell=1}^{L}\sum_{i=1}^{n^{(\ell)}} m_i^{(\ell)} D_\psi^{(\ell)}\right).$$

As for weight learning, reservoir computing only requires fitting the linear readout map. For each reservoir, given T collected samples, ridge regression solved through the normal equations has cost dominated by the construction of the feature covariance matrix and its inversion, namely

$$O\Big(T(D_\psi^{(\ell)})^2 + (D_\psi^{(\ell)})^3 + T\, m_i^{(\ell)} D_\psi^{(\ell)}\Big). \tag{6}$$

Hence, using many local reservoirs with separate readouts keeps both the dynamical updates and the linear solves moderate in size, whereas a single global reservoir would require a much larger state dimension and, because of the higher-order dependencies, a substantially more expensive readout fit.

We next evaluate the model memory footprint by counting the stored coefficients. The recurrent and input matrices remain fixed after initialization, leaving only the readout matrices to be trained. Following the parallel reservoir formulation of [22], we assume a homogeneous spatial grid, so that $|\Omega_i^{(\ell)}|$ and $|\Gamma_i^{(\ell)}|$ are independent of the reservoir index i. Under this assumption, reservoirs in the same layer can share identical fixed matrices while retaining independent internal states and reservoir-specific readout weights. Consequently, the number of fixed parameters does not scale with the number $n^{(\ell)}$ of reservoirs in that layer, and is given by

$$N_{\mathrm{fixed}} = \sum_{\ell=1}^{L}\Big[s^{(\ell)} + D_r^{(\ell)}\big(|\Omega^{(\ell)}| + |\Gamma^{(\ell)}|\big)\Big].$$

By contrast, the readout weights are reservoir-specific, because each reservoir predicts over its own central support. Therefore, the total number of trainable coefficients is

$$N_{\mathrm{train}} = \sum_{\ell=1}^{L}\sum_{i=1}^{n^{(\ell)}} m_i^{(\ell)} D_\psi^{(\ell)}.$$

Although this quantity seems to scale with the number of reservoirs, increasing $n^{(\ell)}$ reduces the size of the local output $m_i^{(\ell)}$ and often allows for smaller $D_\psi^{(\ell)}$ as well. This separation between layer-shared reservoir dynamics and many small linear readouts is what allows CS-RC to remain scalable while still resolving high-dimensional spatio-temporal outputs.

2. Performances, comparison and functioning

2.1. Parallel vs cross-scale RC

We trained and evaluated our cross-scale reservoir computing model using the Copernicus satellite-derived global daily sea surface temperature (SST) [7] spanning from 1981 to 2016.

We fixed the readout nonlinearity $\psi(r) = r$, so that predictive capacity stems linearly from the internal reservoir dynamics. Inputs are coarse-grained via spatial averaging into three-layer resolutions $18°\times18°$ (layer 1), $6°\times6°$ (layer 2), and $2°\times2°$ (layer 3), yielding progressively smoothed fields that emphasize large-scale variability at upper layers and residual fine-scale variability at lower layers. Each layer is partitioned into overlapping local subregions served by independent parallel reservoirs. For all the layers the overlap (orange areas in Fig. 1c) is fixed to 2 tiles.

The baseline model consists of a single-layer of parallel reservoirs at fixed resolution, comprising 162 networks. In contrast, the proposed cross-scale architecture employs a multilayered design with 2, 18, and 162 reservoirs in layers one, two, and three, respectively. Each reservoir contains 1250 neurons.

We ran a grid search, sweeping g and noise level linearly, and g_{in}, g_l, and τ logarithmically across layers (see "Simulation settings"). Better hyperparameter combinations may exist, but the comparison is consistent, exploring the same set of parameters for the highest spatial resolution in the 1-, 2-, and 3-layers models. For each configuration and prediction horizon T, we computed the total root-mean-square error $\mathrm{RMSE}_{t\leq T}$ defined as

$$\mathrm{RMSE}_{t\leq T} = \sqrt{\frac{1}{C}\sum_{i=1}^{C}\Big\langle\big(\mathrm{v}_i(t) - \hat{\mathrm{v}}_i(t)\big)^2\Big\rangle_t}, \tag{7}$$

where the temporal average is over all the samples up to time T, C is the set of observed coordinates (e.g., grid points), $\hat{\mathrm{v}}_i(t)$ is the true value, and $\mathrm{v}_i(t)$ is the model forecast for coordinate i at time step t. We defined the best set of parameters as the one that corresponds with the minimal error for $T = 1, 5, 10, 50$ weeks.

Both two- and three-layer models progressively reduce $\mathrm{RMSE}_{t\leq T}$ over time compared to the single-layer baseline, with the gain widening at longer time horizons where error accumulation is most severe Fig. 2(a). Error ratios relative to the three-layer model demonstrate that improvements largely saturate between two and three layers, indicating that once dominant slow modes are captured by a layer, further depth does not yield significant returns for this task (Fig. 2b).

To pinpoint the source of this improvement, we conducted a principal component analysis (PCA). We removed the first P principal components

(PCs) of the target, which carry slowly varying and quasi-periodic variability; the absolute autocorrelation of the data decays faster and with less periodicity as P increases, confirming the removal of some long-memory content (Fig. 2c). Consistently, the performance gap between multi-layer and single-layer models shrinks as P grows, with the ratio at a fixed horizon of 10 weeks showing clear degradation of the multi-layer advantage when slow PCs are removed (Fig. 2d), demonstrating that inter-layers coupling primarily benefits forecasting by capturing and propagating slow coherent modes.

Finally, to characterize where this multi-layer structure yields the largest forecasting improvements, Fig. 2(e) maps the total error reduction at 50 weeks when comparing three-layer versus single-layer models ($\Delta\mathrm{RMSE}_{t\le T} = \mathrm{RMSE}^{N_L=3}_{t\le T} - \mathrm{RMSE}^{N_L=1}_{t\le T}$). Red colors indicate regions where the multi-layer approach performs better, while blue colors show regions where it performs worse. The improvement is most pronounced over mid- and high-latitude oceans of the Northern Hemisphere, while the Southern Hemisphere shows only scattered improvements. This hemispheric difference likely reflects fundamental contrasts in climate dynamics between the two hemispheres. The Northern Hemisphere contains large landmasses that create strong temperature contrasts between land and ocean, promoting hemispheric asymmetry and intensified slowly-evolving climate patterns that persist for months to decades [17,28] – exactly the type of data components that benefit from the approach used here.

2.2. Emerging linearity

Spatial averaging in the first layers contracts the dynamic range of the reservoir input. Consequently, we expect simpler dynamical regimes for optimally trained networks at lower resolutions. Empirically, we find that the maximum network activity within the optimized reservoirs decreases monotonically from the finest to the coarsest layer (Fig. 3a), indicating operation closer to the linear region of the gain function (Eq. (3)) at larger spatial scales. The mechanics of these regimes are fully tractable [8], enabling the identification of slow dynamical modes that are inherited from the low-resolution input layers. This linearization yields the approximated autonomous linear dynamics:

$$\tau\dot{\mathbf{r}} \approx (\mathbf{W} + \mathbf{W}_{\mathrm{in}}\mathbf{W}_{\mathrm{out}})\mathbf{r} - \mathbf{r}\,. \tag{8}$$

Consequently, the dynamical repertoire of the network and its predictive output are determined by the eigenspectrum of the effective connectivity matrix, $\tilde{\mathbf{W}} = \mathbf{W} + \mathbf{W}_{\mathrm{in}}\mathbf{W}_{\mathrm{out}}$.

Diagonalizing $\tilde{\mathbf{W}} = \mathbf{Z}\mathbf{\Lambda}\mathbf{Z}^{-1}$ with $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \dots, \lambda_{D_r})$, and introducing coordinates $\mathbf{z}(t) = \mathbf{Z}^{-1}\mathbf{r}(t)$, we obtain independent modal evolutions: $z_k(t) = z_k(0)\exp[(\lambda_k - 1)t/\tau]$. This provides a linear decomposition for the network output signal:

$$\mathrm{v}_i(t) \approx \sum_{k=1}^{D_r} [\mathbf{W}_{\mathrm{out}}\mathbf{Z}]_{ik}\, z_k(0) e^{(\lambda_k - 1)t/\tau}\,. \tag{9}$$

At the layer level, the ensemble of trained reservoirs can be represented as a single, larger effective reservoir, whose state vector concatenates the states of individual networks and whose structured connectivity matrix reflects interactions induced by overlapping spatial tiles, as shown in Eq. (4). Fig. 3(b) shows an example of the eigenvalue spectrum of the effective reservoir for the first layer. The eigenvalues are color-coded by their corresponding mode weight ($w_k = \sum_i [\mathrm{W}_{\mathrm{out}}\mathrm{Z}]_{ik}\, z_k(0)$), which quantifies the contribution of each mode to the output. The spectrum exhibits a highly skewed contribution profile: a small subset of modes carries decomposition weights orders of magnitude larger than the bulk, demonstrating that the output is dominated by a few slow oscillatory components (Fig. 3c).

The top pair forms a complex conjugate with large weight, generating the leading oscillatory signal that aligns with the annual seasonal cycle in SST, consistent with the slow coherent structure emphasized by coarse-graining (Fig. 3d).

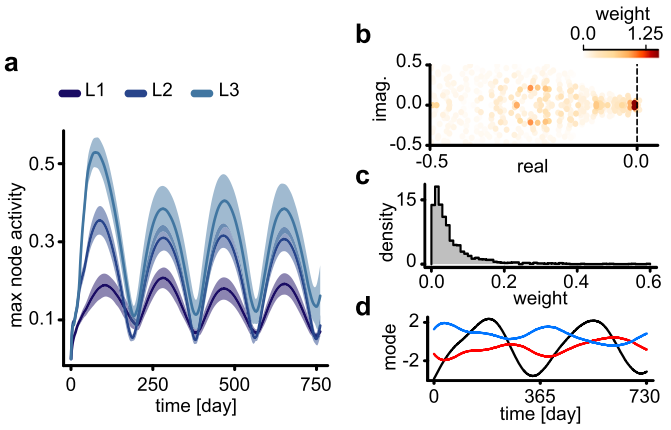

Fig. 3. a, Maximum network activity across all nodes for three different layers (L1, L2, L3). The solid line represents the mean and the shaded area represents the standard deviation across 25 independent system realizations. **b,** The eigenvalue spectrum of the effective reservoir that reproduces the dynamics of layer 1 (L1). The color of each point indicates its decomposition weight. The data shown are from a single system realization. **c,** Decomposition weight distribution for the effective reservoir shown in (b). **d,** The real (black) and imaginary (red, blue) components of the complex conjugate linear mode pair associated with the highest decomposition weight.

2.3. An efficient way to predict chaos

To further evaluate the proposed architecture, we assessed its efficiency in forecasting chaotic dynamics. Following [22], we consider the Kuramoto–Sivashinsky (KS) equation augmented with a spatially inhomogeneous forcing term,

$$u_t = -u\,u_x - u_{xx} - u_{xxxx} + \mu\cos\left(\frac{2\pi x}{\lambda}\right). \tag{10}$$

where the parameter μ determines the amplitude of the inhomogeneous forcing.

We set $L_{\mathrm{KS}} = 44$, $\lambda = 100$, $\mu = 0.01$, and we simulate the system on $Q = 128$ uniformly spaced grid points, yielding time series with 128 state variables. As reported in the supplementary material of [22], small μ has negligible impact on the maximum Lyapunov exponent; accordingly, we assumed $\Lambda_{\max} \approx 0.08$ for our set of parameters [10].

The simulated KS dynamics exhibit structured spatiotemporal correlations (Fig. 4a): at short lags the correlation remains appreciable over almost all spatial offsets Δx and shows oscillatory patterns. This implies that, for finite lead times, incorporating the recent history of spatially separated locations may provide additional predictive information, even though the governing equation is local.

To test this hypothesis, we employ a two-layer Cross-Scale Reservoir (CS-RC) with strong cross-scale coupling. The coarse layer operates on a regridded resolution $Q^{(1)} = Q/4$, and model hyperparameters are chosen from a broad sweep (see "Simulation settings"). The first layer comprises 4 reservoirs, while the second layer comprises 16 reservoirs. A representative spatiotemporal error field is shown in Fig. 4(b), demonstrating accurate forecasts up to 8 Lyapunov times.

In this strong-coupling regime, performance is primarily determined by the coarse (layer-1) dynamics. Increasing the layer-1 reservoir size $D_r^{(1)}$ yields a marked improvement, whereas increasing the layer-2 size $D_r^{(2)}$ produces comparatively modest gains; in particular, performance is already strong for $D_r^{(2)} = 200$ (Fig. 4c). This places CS-RC in a high-efficiency operating regime in which most computation is delegated to the lower-resolution (and lighter) layer.

To quantify efficiency, we compare accuracy and computational cost as $D_r^{(1)}$ increases against established reservoir approaches: a standard Echo State Network (ESN, [13]), Next-Generation Reservoir Computing (NG-RC, [12]), Deep Reservoir Computing (Deep-RC, [11]), and Parallel

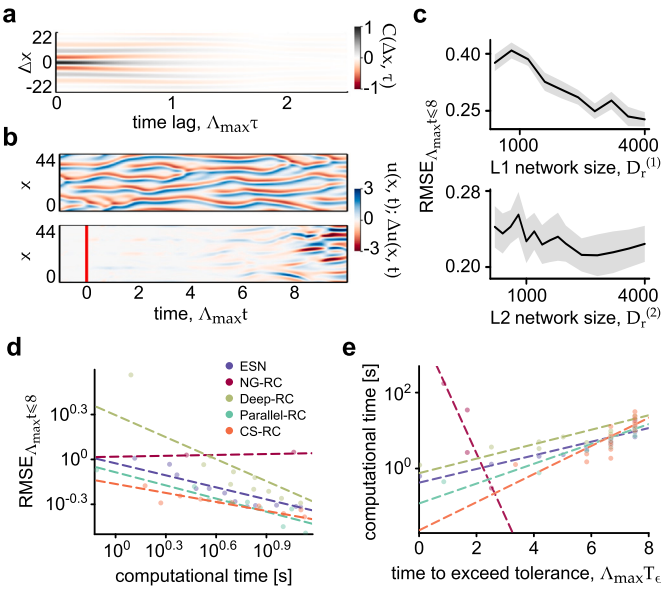

Fig. 4. a, Spatiotemporal correlation in the Kuramoto–Sivashinsky (KS) system with $L_{KS} = 44$, $Q = 128$, $\lambda = 100$, $\mu = 0.01$. **b,** Representative KS activity $u(x,t)$ (top) and CS-RC prediction error $\Delta u(x,t)$ (bottom) in the regime dominated by cross-layers coupling. **c,** Mean total prediction error at 8 Lyapunov times versus reservoir size in layer 1 (top; layer-2 size fixed at 400) and layer 2 (bottom; layer-1 size fixed at 4000); shaded regions indicate the SEM. **d,** Accuracy–cost comparison across reservoir architectures: mean total error at 8 Lyapunov times versus total computational time (varied by increasing node count; for NG-RC, by increasing memory length). Dashed lines show power-law fits, linear regression in $\log_{10} y$–$\log_{10} x$ space. **e,** Computational time versus predictability horizon T_ϵ, defined as the largest T such that $\mathrm{RMSE}_{t\le T} < \epsilon$. Dashed lines show exponential fits, linear regression in $\log_{10} y$–x space. Results are averaged over 20 runs.

Reservoir Computing (Parallel-RC, [22]). For each method, we tune hyperparameters across multiple configurations (see "Simulation settings") and report wall-clock time for the full run, including reservoir-state evolution during training, test and readout fitting. We vary node count for reservoir-based methods and vary the memory length (number of past steps) for NG-RC.

With the prediction horizon set to $T = 8/\Lambda_{max}$ (eight Lyapunov times), CS-RC achieves lower error than all baselines in the low-compute regime (Fig. 4d). Moreover, for an error tolerance $\epsilon = 0.3$, CS-RC attains longer under-tolerance predictability horizons T_ϵ, defined as the largest T such that $\mathrm{RMSE}_{t\le T} < \epsilon$, at substantially lower computational cost (Fig. 4e). The advantage diminishes at very large $D_r^{(1)}$, where single-reservoir readout fitting becomes expensive, consistent with Eq. (6). In this case the efficiency should be recovered by increasing the number of reservoirs per layer while reducing the number of nodes per reservoir.

The same Eq. (6) also helps clarify why CS-RC can outperform the other reservoir computing models. In monolithic approaches such as ESN, Deep-RC, and NG-RC, ridge regression is performed on a single large feature block, so the superlinear growth of the readout cost rapidly becomes dominant as the feature dimension increases. By contrast, Parallel-RC and CS-RC distribute the readout computation across many smaller local regressions. CS-RC gains an additional advantage because the fine layer can be kept small, while most of the predictive burden is delegated to the coarse layer, which involves fewer spatial partitions.

3. Conclusions, limitations and future directions

We have introduced a cross-scale reservoir computing framework for efficient forecasting of large, multi-resolution spatiotemporal data. By hierarchically coupling coarse and fine representations, the architecture better captures slow dynamical modes and distributes predictive load across layers. Our results demonstrate consistent improvements over single-layer baselines across forecasting horizons, particularly due to the model capacity to retain and propagate quasi-periodic principal components. The benefits diminish when these slow modes are removed, underscoring their critical role. Additionally, the emergence of linear dynamics in coarser layers allows for interpretable modal decompositions of the learned dynamics.

The experiments on the forced Kuramoto–Sivashinsky system further clarify the computational role of the architecture. In the strong cross-scale coupling regime, the predictive burden is concentrated in the coarse layer, while the fine layer mainly refines the forecast locally. This separation yields a favorable accuracy-cost trade-off in the low-compute regime over ESN, NG-RC, Deep-RC, and Parallel-RC baselines (Fig. 4).

More broadly, these results support the view that cross-scale organization is useful not only because many physical systems are multiresolutional, but because scale separation can be exploited as an architectural prior that reallocates computation toward the variables carrying the longest predictive memory.

Several limitations should, however, be made explicit. First, the present construction depends on design choices that are fixed a priori, including the number of layers, the regridding operator, the partition into local tiles and the overlap width. These ingredients were chosen in a principled but ultimately heuristic way, and different datasets may favor different coarse-graining maps or coupling topologies, based on the heterogeneity of the underlying system. A related issue is that the analysis here presented assumes fairly regular spatial organization and shared reservoir matrices within a layer, assumptions that help scalability but may become restrictive.

A second limitation concerns model selection. Although the CS-RC training and simulation are cheap, selecting a good hyperparameter configuration for a multi-layer architecture is substantially more demanding than for simpler models because the search space grows combinatorially with the number of layers and with the number of scales introduced at each layer. In the present study this burden was controlled by a sequential top-down strategy, in which layer 1 is tuned first and subsequent layers are optimized conditionally on the already selected coarser predictors. This procedure is computationally practical and consistent with the causal structure of the architecture, but it is not equivalent to joint optimization. It can miss compensatory interactions between layers and bias the selected solution toward early design choices. This motivates future work on more efficient tuning strategies.

A promising way to ease part of this computational burden is physical reservoir computing, where some of the nonlinear processing and memory are provided directly by the device itself, potentially simplifying hyperparameter tuning [25,29]. This makes it a natural candidate for low-power, real-time forecasting close to the sensor or instrument, and recent hardware implementations have already shown high-accuracy predictions [18,19].

The framework is also readily extensible at the algorithmic level. Replacing classical ESN units with more expressive reservoir variants, including NG-RC-inspired feature constructions [12], hybrid knowledge-based models [6,23], or nearly linear recurrent networks [5], may improve either predictive performance or interpretability depending on the application. Likewise, incorporating spatially adaptive normalization or resolution-dependent weighting could help in strongly nonstationary domains where variability is localized and where Euclidean distance alone is a poor proxy for dynamical relevance. As recent high-resolution studies show, climatological variability is often highly localized, and predictive architectures should account for both intensity scaling and resolution distance between sensors or grid points [24]. Incorporating such spatial sensitivity and, where available physical priors, could further improve performance on irregular and nonstationary regions.

Finally, the cross-scale design is well suited to other domains with naturally hierarchical measurements, including neuroscience. Coupling slow signals (e.g., fMRI) with faster signals (e.g., MEG, EEG) and

fine-grained electrophysiology (e.g., LFP, MUA) [30] may both improve predictive capacity and provide interpretable links between dynamics across spatial scales.

Simulation settings

All experiments used identical training and evaluation protocols across datasets and reservoir architectures to ensure a fair comparison.

Datasets and preprocessing

Kuramoto–sivashinsky (KS)

For the Kuramoto–Sivashinsky (KS) system, we used precomputed trajectories obtained by solving the PDE with a Fourier pseudo-spectral solver and exponential time-differencing Runge–Kutta of order 4 (ETDRK4) [15]. Data were sampled with a nominal time step $\Delta t = 0.25$ and normalized independently at each spatial location to have zero mean and unit variance. Spatial coarse-graining was implemented via block averaging, reducing the effective spatial dimension according to the chosen resolution divisor.

Sea surface temperature (SST)

Sea Surface Temperature (SST) experiments followed the same preprocessing pipeline, using multi-resolution fields normalized per grid point over time.

Reservoir construction and parameterization

Reservoir recurrent matrices were generated as sparse random matrices with entries drawn from a standard normal distribution. Sparsity was controlled via the average network degree (here set to 10), yielding a connection probability $p = \text{degree}/N$. Matrices were rescaled to match a prescribed spectral radius g using

$$\mathbf{W} \leftarrow \mathbf{W}\,\frac{g}{\rho(W)},$$

where $\rho(\mathbf{W})$ denotes the maximum eigenvalue modulus.

Input weights $\mathbf{W}_{\text{in}}$ were sampled i.i.d. from a uniform distribution on $[-g_{\text{in}}, g_{\text{in}}]$.

Following the parallel reservoir formulation of [22], reservoirs belonging to the same layer share identical recurrent connectivity matrices, while maintaining independent internal states and readout weights.

Input mapping and reservoir dynamics

Input weights were generated via structured block assignments mapping contiguous neuron groups to spatial inputs and neighboring regions. Reservoir dynamics were integrated with the forward Euler method using a time step Δt. We used $\Delta t = 0.25\,\text{day}$ for the SST experiment and $\Delta t = 0.25$ for the KS experiment.

Training protocol and evaluation

KS training and prediction

Training was performed after a washout period of 1000 time steps, followed by 50000 training time steps. Autonomous predictions were evaluated over 1000 time steps after an additional warmup interval of 1000 steps using teacher forcing.

SST training and prediction

Training sequences consisted of 30000 samples after a washout of 1000 steps, with prediction horizons of 1000 steps.

Readout training and feature design

Output weights were trained using ridge regression,

$$W_{\text{out}} = Y X^{\top}\left(X X^{\top} + \alpha I\right)^{-1},$$

with the regularization coefficient α.

In the KS experiment, readouts used a quadratic feature augmentation obtained by squaring the even-indexed reservoir state components prior to regression as in [22].

Tuning

All models in the manuscript were tuned via a grid search. Each layer of a CS-RC is optimized sequentially in a top-down fashion.

For reservoir-based algorithms, the spectral radius g was swept linearly from 0.1 to 1.0 in steps of 0.1. All other input and recurrent weight scales were explored on a $\log_{10}$ grid, and normalized by the square root of the corresponding number of elements. Similarly the time scale τ was swept for SST, whereas for KS it was fixed to $\tau = 0.25$.

For NG-RC, the polynomial degree was fixed to 2. The memory length was swept linearly from 1 to 6. Both the ridge regularization coefficient α and the input-noise standard deviation (used as an additional regularizer by perturbing the input data) were explored on a $\log_{10}$ grid. Across all models, α was selected over the range 10^{-6} to 10^{-1}.

Hardware

The wall-clock time analysis in Fig. 4 results from averaging across 20 single Julia processes with 16 Julia threads on a dual-socket AMD EPYC 9654 CPU node (2×96 physical cores; 192 cores total; SMT disabled), running AlmaLinux 9.7.

CRediT authorship contribution statement

Nicola Alboré: Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Gabriele Di Antonio:** Writing – original draft, Visualization, Validation, Supervision, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Fabrizio Coccetti:** Writing – review & editing, Supervision, Resources, Project administration, Conceptualization. **Andrea Gabrielli:** Writing – review & editing, Supervision, Resources, Project administration, Conceptualization.

Code availability

A public repository for the proposed algorithm has been published at [1].

Declaration of generative AI and AI-assisted technologies in the writing process

Large language models were used solely as writing aids to enhance grammar, phrasing, and readability of the manuscript.

Declaration of competing interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

We thank Roberto Benzi (University of Rome Tor Vergata), Carlo Buontempo (Director of C3S), Giulio Iannelli (Centro Ricerche Enrico Fermi) and Marcello Petitta (Roma Tre University) for feedback on an earlier version of the manuscript, and Carmelo Pellegrino (INFN-CNAF) for support with computational resources. This work was partially supported by the European Union's Horizon Europe programme under the Marie Skłodowska-Curie grant agreement No 101126667 for the Ramon Llull-AIRA Postdoctoral Programme.

Data availability

The dataset is publicly available on the Copernicus website. Custom code is shared on GitHub at https://github.com/nicoalbo0/cross-scale-reservoir-computing.

References

[1] N. Alboré, G. Di Antonio, CrossScaleRC (v1.0.1) [software], Zenodo, 2026, https://doi.org/10.5281/zenodo.18846959

[2] T. Arcomano, I. Szunyogh, J. Pathak, A. Wikner, B.R. Hunt, E. Ott, A machine learning-based global atmospheric forecast model, Geophys. Res. Lett. 47 (2020) e2020GL087776.

[3] J. Barnes, P. Hut, A hierarchical O (N log N) force-calculation algorithm, Nature 324 (1986) 446–449.

[4] P. Bauer, A. Thorpe, G. Brunet, The quiet revolution of numerical weather prediction, Nature 525 (2015) 47–55.

[5] M. Brenner, C.J. Hemmer, Z. Monfared, D. Durstewitz, Almost-linear RNNs yield highly interpretable symbolic codes in dynamical systems reconstruction, Adv. Neural Inf. Process. Syst. 37 (2024) 36829–36868.

[6] R. Chepuri, D. Amzalag, T. Antonsen, M. Girvan, Hybridizing traditional and next-generation reservoir computing to accurately and efficiently forecast dynamical systems, Chaos Interdiscip. J. Nonlinear Sci. 34 (2024).

[7] Copernicus Climate Change Service, Climate Data Store, Sea surface temperature daily gridded data from 1981 to 2016 derived from a multi-product satellite-based ensemble, 2019, https://doi.org/10.24381/cds.ab204534 (Accessed on 1 September 2024).

[8] G. Di Antonio, T. Gili, A. Gabrielli, M. Mattia, Linearizing and forecasting: a reservoir computing route to digital twins of the brain, Adv. Sci. (2026) e17234.

[9] K. Duraisamy, G. Iaccarino, H. Xiao, Turbulence modeling in the age of data, Annu. Rev. Fluid Mech. 51 (2019) 357–377.

[10] R.A. Edson, J.E. Bunder, T.W. Mattner, A.J. Roberts, Lyapunov exponents of the kuramoto–sivashinsky PDE, The ANZIAM Journal 61 (2019) 270–285.

[11] C. Gallicchio, A. Micheli, L. Pedrelli, Deep reservoir computing: a critical experimental analysis, Neurocomputing 268 (2017) 87–99.

[12] D.J. Gauthier, E. Bollt, A. Griffith, W.A. Barbosa, Next generation reservoir computing, Nat. Commun. 12 (2021) 1–8.

[13] H. Jaeger, H. Haas, Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication, Science 304 (2004) 78–80, https://doi.org/10.1126/science.1091277

[14] E. Kalnay, Atmospheric Modeling, Data Assimilation and Predictability, Cambridge University Press, 2003.

[15] A.K. Kassam, L.N. Trefethen, Fourth-order time-stepping for stiff PDEs, SIAM J. Sci. Comput. 26 (2005) 1214–1233.

[16] J.Z. Kim, D.S. Bassett, A neural machine code and programming framework for the reservoir computer, Nat. Mach. Intell. 5 (2023) 622–630.

[17] F. Liu, F. Song, Y. Luo, Human-induced intensified seasonal cycle of sea surface temperature, Nat. Commun. 15 (2024) 3948.

[18] J. Liu, G. Feng, W. Li, S. Hao, S. Han, Q. Zhu, B. Tian, C. Duan, J. Chu, Physical reservoir computing for edge AI applications, The Innovation Materials 3 (2025, 100127–1.

[19] X. Liu, B. Liu, G. Guan, High-accuracy prediction of chaotic time series using multi-stage coupled Chua's circuit-based reservoir computing, Neurocomputing (2026) 132867.

[20] M. Lukoševičius, H. Jaeger, Reservoir computing approaches to recurrent neural network training, Comput. Sci. Rev. 3 (2009) 127–149.

[21] L. Manneschi, M.O. Ellis, G. Gigante, A.C. Lin, P. Del Giudice, E. Vasilaki, Exploiting multiple timescales in hierarchical echo state networks, Front. Appl. Math. Stat. 6 (2021) 616658.

[22] J. Pathak, B. Hunt, M. Girvan, Z. Lu, E. Ott, Model-free prediction of large spatiotemporally chaotic systems from data: a reservoir computing approach, Phys. Rev. Lett. 120 (2018) 024102.

[23] J. Pathak, A. Wikner, R. Fussell, S. Chandra, B.R. Hunt, M. Girvan, E. Ott, Hybrid forecasting of chaotic processes: using machine learning in conjunction with a knowledge-based model, Chaos Interdiscip. J. Nonlinear Sci. 28 (2018).

[24] V. Pecorino, T.D. Matteo, M. Milazzo, L. Pasotti, A. Pluchino, A. Rapisarda, Empirical analysis of hourly rainfall data in Sicily from 2002 to 2023, Eur. Phys. J. B 97 (2024) 154.

[25] E. Picco, L. Jaurigue, K. Lüdge, S. Massar, Efficient optimisation of physical reservoir computers using only a delayed input, Commun. Eng. 4 (2025) 3.

[26] I. Price, A. Sanchez-Gonzalez, F. Alet, T.R. Andersson, A. El-Kadi, D. Masters, T. Ewalds, J. Stott, S. Mohamed, P. Battaglia, et al., Probabilistic weather forecasting with machine learning, Nature 637 (2025) 84–90.

[27] M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707.

[28] R. Ruela, M. Sousa, M. DeCastro, J. Dias, Global and regional evolution of sea surface temperature under climate change, Global and Planetary Change 190 (2020) 103190.

[29] M. Yan, C. Huang, P. Bienstman, P. Tino, W. Lin, J. Sun, Emerging opportunities and challenges for the future of reservoir computing, Nat. Commun. 15 (2024) 2056.

[30] C. Yen, C.L. Lin, M.C. Chiang, Exploring the frontiers of neuroimaging: a review of recent advances in understanding brain functioning and disorders, Life 13 (2023) 1472.

Author biography

Nicola Alborè is a Ph.D. student in Physics at the University of Rome Tor Vergata, supported by the Enrico Fermi Research Center (Centro Ricerche Enrico Fermi, CREF), Rome, Italy.

Gabriele Di Antonio is a postdoctoral researcher at Universitat Pompeu Fabra, Barcelona, Spain, with an MSCA Cofund Ramon Llull-AIRA fellowship.

Fabrizio Coccetti is the Director of Technology at the Enrico Fermi Research Center (Centro Ricerche Enrico Fermi, CREF), Rome, Italy.

Andrea Gabrielli is an Associate Professor of Physics at Roma Tre University, Rome, Italy.