ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

# Memristive-Friendly Hadamard Reservoirs

Andrea Ceni[1], Gianluca Milano[2], Carlo Ricciardi[3], and Claudio Gallicchio[1] *

1- Department of Computer Science, University of Pisa, Italy
2- Istituto nazionale di ricerca metrologica (INRiM), Italy
3- Department of Applied Science and Technology, Politecnico di Torino, Italy

**Abstract**. Reservoir Computing (RC) processes temporal data using a fixed recurrent system and a trained linear readout, making it appealing for hardware-limited neuromorphic settings. Memristive-friendly reservoirs refine this idea by adopting neuron dynamics inspired by resistive devices, but they still rely on dense recurrent matrices that are costly to implement physically. We introduce a Hadamard-based alternative in which the recurrence is replaced by an orthogonal, multiplier-free operator with $O(N \log N)$ complexity and $O(N)$ parameters. Experiments on time-series tasks show that the proposed approach matches the performance of dense RC baselines while improving hardware compatibility.

## 1 Introduction

Reservoir computing (RC) [1] offers an attractive framework for neuromorphic and in-memory processing with recurrent neural models, because training is restricted to a linear readout, while the recurrent core remains fixed. This property makes RC conceptually compatible with memristive and nanowire substrates [2], where implementing large numbers of adaptive synapses is impractical. However, the classical construction of reservoirs relies on dense random recurrent matrices, which are difficult to realize in hardware due to their $O(N^2)$ memory footprint, the need for high-precision multipliers, and limited opportunities for calibration or structural simplification. Recent work has proposed memristive-friendly reservoir models in which the neuron update is derived from the dynamics of resistive devices [3]. This leads to discrete-time reservoirs with controlled stability and physical interpretability, but it still requires a dense recurrent matrix to achieve sufficient state mixing. In parallel, structured orthogonal transforms, in particular, the Walsh-Hadamard transform, have emerged as efficient substitutes for dense random projections [4]. These transforms are exactly orthogonal, multiplier-free, and can be applied in $O(N \log N)$ time, while retaining the statistical mixing properties needed for high-dimensional computation.

Motivated by these developments, this paper introduces a Hadamard-based variant of the memristive reservoir, in which the recurrent matrix is replaced by a structured orthogonal operator composed of a Walsh-Hadamard transform, random diagonal sign matrices, and a permutation. The objective of this paper is to present the mathematical formulation of this Hadamard-based memristive-friendly reservoir, and to provide a first software demonstration showing that

*Work supported by NEURONE, a project funded by the European Union - Next Generation EU, M4C1 CUP I53D23003600006, under program PRIN 2022 (prj code 20229JRTZA).

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

it retains the predictive performance of its dense counterpart while significantly reducing computational complexity.

## 2 Background

Reservoir Computing (RC) provides an efficient framework for processing temporal data by driving a fixed (i.e., untrained) recurrent dynamical system with an external input and training only a linear readout. In what follows, we refer to the notation of the well-known Echo State Network (ESN) model [5].

Let $\mathbf{h}_t \in \mathbb{R}^N$ and $\mathbf{x}_t \in \mathbb{R}^d$ denote, respectively, the reservoir state and the external input at time $t$. Reservoir dynamics are described in terms of an input-driven non-linear iterated map, as follows:

$$\mathbf{h}_t = (1-\alpha)\mathbf{h}_{t-1} + \alpha \tanh\big(\mathbf{W}_h\mathbf{h}_{t-1} + \mathbf{W}_x\mathbf{x}_t + \mathbf{b}_h\big), \tag{1}$$

where $\mathbf{W}_h \in \mathbb{R}^{N\times N}$ and $\mathbf{W}_x \in \mathbb{R}^{N\times d}$ are fixed random matrices, $\mathbf{b}_h \in \mathbb{R}^N$ is a bias vector, and $\alpha \in (0,1]$ is the leaking rate. Since the recurrent weights are not trained, the dynamical system must satisfy the Echo State Property (ESP), ensuring that its state is determined by the input history rather than by initial conditions. In standard ESN practice, the dynamical regime is controlled through the spectral radius of $\mathbf{W}_h$ (denoted as $\rho$), which is set by rescaling the initialized matrix to a desired value. The values in $\mathbf{W}_x$ and $\mathbf{b}_h$ are typically drawn from a uniform distribution in $(-1,1)$ and then rescaled, respectively, by an input scaling $\omega_x$ and a bias scaling $\omega_b$. The readout is learned by linear regression on the reservoir states, while the recurrent dynamics remain fixed throughout training. As the internal weights remain fixed, the choice of the recurrent operator plays a central role in determining stability, memory, and expressivity. Classical reservoirs rely on dense random matrices, which offer good empirical performance but are expensive to store and compute, scaling as $O(N^2)$ in both memory and operations.

**Memristive-friendly Reservoir Computing.** Recently, a memristive-friendly formulation of the ESN operation has been introduced, where the state update is derived from the effective kinetics of resistive devices. A representative formulation is provided through a potentiation-depression mechanism governed by rates $K_p(z) = \kappa_{p0}e^{\eta_p z}$ and $K_d(z) = \kappa_{d0}e^{-\eta_d z}$, leading to the following reservoir dynamics of a Memristive-friendly ESN (MF-ESN) [3]:

$$\mathbf{z}_t = \mathrm{Rescale}(\mathbf{W}_h\mathbf{h}_{t-1} + \mathbf{W}_x\mathbf{x}_t + \mathbf{b}_h), \tag{2}$$

$$\mathbf{h}_t = \gamma\mathbf{h}_{t-1} + \varepsilon\big[K_p(\mathbf{z}_t) - \mathrm{diag}(K_p(\mathbf{z}_t) + K_d(\mathbf{z}_t))\mathbf{h}_{t-1}\big], \tag{3}$$

where $\mathrm{Rescale}(z) = (b-a)/(1+e^{-z\,s}) + a$ is a fixed rescaling operation, $a = 0.35$ and $b = 1.15$ are fixed parameters as in [3], and $s$ is a hyperparameter that tunes the nonlinearity of the rescaling. Here, $\kappa_{p0}$, $\kappa_{d0}$, $\eta_p$, and $\eta_d$ are physics-related hyper-parameters that determine the strength and asymmetry of potentiation and depression, while $\varepsilon$ and $\gamma \in (0,1]$ control, respectively, the discretization and leakage. Under bounded inputs, stability conditions can be expressed directly

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

in terms of these hyper-parameters and the spectral properties of the recurrent matrix $\mathbf{W}_h$ (see [3]). This formulation provides a physically interpretable and tunable neuron model, but the recurrent term $\mathbf{W}_h\mathbf{h}_{t-1}$ in eq. 2 remains the dominant computational and memory bottleneck due to its quadratic cost.

**Hadamard-based recurrent mixing.** The Walsh-Hadamard transform is an efficient replacement for dense random matrices in high-dimensional models. For state dimension $N = 2^K$, the Hadamard matrix $\mathbf{H}_N \in \{\pm 1\}^{N\times N}$ is defined recursively by: $\mathbf{H}_1 = [1], \qquad \mathbf{H}_{2n} = \begin{bmatrix} \mathbf{H}_n & \mathbf{H}_n \\ \mathbf{H}_n & -\mathbf{H}_n \end{bmatrix}$, and the normalized transform $\widehat{\mathbf{H}}_N = \mathbf{H}_N/\sqrt{N}$ is orthogonal. Crucially, the product $\widehat{\mathbf{H}}_N\mathbf{h}$ can be computed using the fast Walsh-Hadamard transform (FWHT) in $O(N \log N)$ additions and subtractions by decomposing the matrix into a sequence of butterfly stages of the form $(a, b) \mapsto (a + b, a - b)/\sqrt{2}$. This reduces computational cost by a factor of $N/\log N$ comparative to dense multiplication and eliminates all multipliers. In RC, replacing the dense recurrent matrix $\mathbf{W}_h$ by a structured operator composed of a Hadamard transform and simple randomization primitives (sign flips and permutations) preserves the norm of the linear mixing and achieves the statistical spreading needed for effective state diversification.

## 3 Model formulation

Here we introduce a different formulation of the memristive-friendly reservoir operation, in which the recurrent mixing uses a structured orthogonal operator. Instead of using a dense random matrix, the recurrence is replaced by a structured orthogonal operator built from a normalized Hadamard transform, random sign matrices, and a permutation. For a reservoir state dimension $N = 2^K$, the recurrent pre-activation in eq. 2 becomes:

$$\mathbf{z}_t = \text{Rescale}(\rho_M\, \mathbf{M}\mathbf{h}_{t-1} + \mathbf{W}_x\mathbf{x}_t + \mathbf{b}_h), \tag{4}$$

where $\mathbf{M} = \mathbf{D}_2\,\widehat{\mathbf{H}}_N\,\Pi\,\mathbf{D}_1$, and $\rho_M$ is a recurrent scaling hyper-parameter that plays the role of a spectral-radius control for the structured operator. Here, $\widehat{\mathbf{H}}_N = \mathbf{H}_N/\sqrt{N}$ is the normalized Walsh-Hadamard matrix, $\mathbf{D}_1$ and $\mathbf{D}_2$ are diagonal matrices with random independent $\pm 1$ entries, and $\Pi$ is a random permutation matrix. These elements introduce controlled randomness and heterogeneity while remaining orthogonal, so the transformation $\mathbf{M}\mathbf{h}_{t-1}$ preserves the Euclidean norm and produces a global mixing of the state. We dub this model as the **Memristive-friendly Hadamard ESN** (MF-H-ESN). A further simplified variant, denoted MF-H-ESN$_{si}$, adopts a *simplified* hardware oriented structure also for the *input* connections. In this case, the input matrix $\mathbf{W}_x$ and bias $\mathbf{b}_h$ take the form of dense Rademacher matrices and vectors, whose elements are random $\pm 1$ values, re-scaled by $\omega_x$ and $\omega_b$, that act as input and bias scaling hyper-parameters as in conventional ESN/MF-ESN. This eliminates the need for floating-point parameters also in the input drive, making the recurrence and input contributions both multiplier-free.

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

**Cost analysis.** The proposed approach brings a substantial reduction in the required costs involved by the reservoir implementation. First, notice that the structured operator $\mathbf{M}$ is applied without storing the full matrix. The diagonal matrices introduce sign changes, the permutation reorders coordinates, and the Hadamard transform is computed with the fast Walsh-Hadamard transform, which performs the butterfly decomposition of $\widehat{\mathbf{H}}_N$ in $O(N \log N)$ additions and subtractions. This reduces the cost of the recurrent step from $O(N^2)$ to $O(N \log N)$, and reduces storage to $O(N)$ for the sign patterns, the permutation, and the scaling constants. A corresponding reduction appears in the MF-H-ESN$si$ model, where the input matrix and bias no longer require storing floating-point entries. Since $\mathbf{W}_x$ and $\mathbf{b}_h$ are defined by deterministic scalings of dense Rademacher structures, the cost of applying the input transformation collapses to sign-controlled injections followed by constant multiplications shared across all units. This removes the need for $N \times d$ individually programmed analog conductances in the input path, replacing them with a single global gain $\omega_x$ and a fixed binary pattern that can be generated procedurally. The memory required for the input parameters therefore scales linearly with $N$, matching the cost of storing the masks used in the recurrent operator, and the computational overhead becomes negligible compared to the Hadamard mixing.

**Hardware considerations.** The decomposition of the recurrence into sign flips, routing, and pairwise additive interactions substantially lowers the complexity of a physical implementation compared to a dense analog matrix. In contrast to a conventional dense recurrent weight matrix that requires $O(N^2)$ distinct conductances and analog multiplications, the operator $\mathbf{M}$ can be realized by structured interconnects and polarity-controlled summations, which are more naturally compatible with memristive substrates [6]. The simplified input structure in MF-H-ESN$_{si}$ further reduces the number of analog parameters and removes precision-sensitive components. In this case, both the recurrent and input operations rely entirely on polarity manipulations and simple scalings, aligning naturally with the constraints of resistive hardware and keeps compatibility with the global mixing ensured by the structured recurrence.

## 4 Experimental evaluation

We evaluated the proposed architectures on a set of time-series benchmarks covering both classification and regression tasks. [1] For time-series classification, the original datasets were split into training and test using a stratified 66%-33% division, with a nested stratified 66%-33% split of the training portion to obtain training and validation sets. Input sequences were linearly rescaled to the interval (0,1) using statistics computed solely on the training data. For time-series regression, we used the predefined train-test splits provided by the repository and further extracted an 80%-20% training-validation split from the training portion to allow model selection.

[1] Classification datasets from `https://timeseriesclassification.com`; regression datasets from `http://tseregression.org`.

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

We ran experiments with MF-H-ESN and MF-H-ESN$_{si}$ using N = 256 reservoir units. For comparison, we performed the same evaluations with the MF-ESN model and with the conventional ESN defined in eq. 1. To isolate the contribution of the structured recurrence, we further included Hadamard-based counterparts of the standard ESN, denoted H-ESN and H-ESN$_{si}$, obtained by replacing the dense recurrent matrix with the same orthogonal Hadamard operator used in the memristive-friendly variants. Hyper-parameters were explored through a random search over 500 configurations sampled from the ranges: $\rho$ and $\rho_M$ in $\{0.7, 0.8, ...1.2\}$, $\omega_x$ and $\omega_b$ in $\{0.01, 0.1, 1.0, 2.0, 5.0\}$, $\alpha$ in $\{1, 0.1, 0.01, 0.001\}$, and, for MF-ESN-based models, $\varepsilon$ in $\{0.1, 0.01, 0.001\}$, $\gamma$ in $\{0.5, 0.8, 0.95, 1.0\}$, and $s$ in $\{1, 5\}$. Model selection was carried out on the validation split, after which the chosen configuration was retrained on the entire training set and evaluated on the test set. The values of the physics-related hyper-parameters used in our experiments are $\kappa_{p_0} = 0.0001$, $\kappa_{d_0} = 0.5$, $\eta_p = 10$, and $\eta_d = 1$, following [7]. Reported results are averages and standard deviations over 3 runs with different random initializations. In all experiments, the readout was trained on the reservoir state at the final time step of each sequence using ridge regression, with the regularization parameter selected from $\{10^{-5}, 10^{-4}, \ldots, 10^{2}\}$ via leave-one-out on the training set. Performance is reported as classification accuracy for time-series classification tasks and mean squared error for regression tasks, with mean and standard deviation computed over the repeated runs.

Tables 1 and 2 report the results for the classification and regression benchmarks, respectively. Across the classification tasks, the Hadamard-based variants show a clear impact on the memristive-friendly models. MF-H-ESN often matches or slightly improves the performance of MF-ESN, while MF-H-ESN$_{si}$ displays more variability due to its fully binary input structure but remains competitive on several datasets. Similar trends are observed in the regression benchmarks, where the structured variants preserve the overall behaviour of the dense models. For the standard ESN models, the introduction of the Hadamard-based recurrence generally preserves performance and occasionally yields mild improvements. This suggests that the orthogonal structured operator is a viable substitute for dense recurrent matrices also in conventional reservoirs. Overall, the results show that the Hadamard-based mixing preserves the effectiveness of both ESN and MF-ESN architectures while substantially reducing computational and parametric complexity. The memristive-friendly variants exhibit comparable performance under the structured recurrence, confirming the potential of the proposed MF-H-ESN and MF-H-ESN$_{si}$ models as efficient hardware-compatible reservoir architectures.

## 5 Conclusions

We introduced Hadamard-based structured mixing into the memristive-friendly reservoir framework, defining the MF-H-ESN and MF-H-ESN$_{si}$ models. The approach replaces the dense recurrent matrix with an orthogonal operator built from Hadamard transforms, random sign matrices, and a permutation, and op-

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

| **Dataset** | **ESN** | **H-ESN** | **H-ESN$_{si}$** | **MF-ESN** | **MF-H-ESN** | **MF-H-ESN$_{si}$** |
|---|---|---|---|---|---|---|
| Adiac | $0.27_{\pm 0.03}$ | $0.48_{\pm 0.04}$ | $0.53_{\pm 0.05}$ | $0.56_{\pm 0.04}$ | $0.54_{\pm 0.02}$ | $0.39_{\pm 0.03}$ |
| BasicMotions | $1.00_{\pm 0.00}$ | $1.00_{\pm 0.00}$ | $0.91_{\pm 0.07}$ | $1.00_{\pm 0.00}$ | $0.95_{\pm 0.03}$ | $0.86_{\pm 0.05}$ |
| Earthquakes | $0.82_{\pm 0.01}$ | $0.81_{\pm 0.01}$ | $0.81_{\pm 0.00}$ | $0.82_{\pm 0.01}$ | $0.81_{\pm 0.00}$ | $0.79_{\pm 0.01}$ |
| ECG5000 | $0.95_{\pm 0.00}$ | $0.95_{\pm 0.00}$ | $0.94_{\pm 0.00}$ | $0.94_{\pm 0.00}$ | $0.95_{\pm 0.00}$ | $0.95_{\pm 0.00}$ |
| Epilepsy | $0.96_{\pm 0.00}$ | $0.95_{\pm 0.01}$ | $0.96_{\pm 0.01}$ | $0.95_{\pm 0.01}$ | $0.95_{\pm 0.01}$ | $0.96_{\pm 0.01}$ |
| FordA | $0.68_{\pm 0.01}$ | $0.72_{\pm 0.01}$ | $0.70_{\pm 0.01}$ | $0.68_{\pm 0.01}$ | $0.66_{\pm 0.02}$ | $0.66_{\pm 0.00}$ |
| FordB | $0.74_{\pm 0.00}$ | $0.72_{\pm 0.01}$ | $0.73_{\pm 0.01}$ | $0.71_{\pm 0.00}$ | $0.74_{\pm 0.01}$ | $0.70_{\pm 0.01}$ |
| MindReading | $0.60_{\pm 0.01}$ | $0.57_{\pm 0.02}$ | $0.63_{\pm 0.01}$ | $0.56_{\pm 0.01}$ | $0.57_{\pm 0.00}$ | $0.61_{\pm 0.02}$ |
| PowerCons | $0.99_{\pm 0.00}$ | $0.98_{\pm 0.01}$ | $0.98_{\pm 0.00}$ | $0.97_{\pm 0.00}$ | $1.00_{\pm 0.00}$ | $1.00_{\pm 0.00}$ |
| RacketSports | $0.89_{\pm 0.02}$ | $0.88_{\pm 0.00}$ | $0.88_{\pm 0.01}$ | $0.88_{\pm 0.01}$ | $0.85_{\pm 0.01}$ | $0.87_{\pm 0.01}$ |

Table 1: Test set accuracy (the higher the better) on the time-series classification benchmarks.

| **Dataset** | **ESN** | **H-ESN** | **H-ESN$_{si}$** | **MF-ESN** | **MF-H-ESN** | **MF-H-ESN$_{si}$** |
|---|---|---|---|---|---|---|
| AppliancesEnergy | $8.15_{\pm 0.75}$ | $7.65_{\pm 0.93}$ | $7.96_{\pm 1.83}$ | $8.60_{\pm 1.13}$ | $10.05_{\pm 0.36}$ | $9.89_{\pm 2.32}$ |
| Covid3M $_{(\times 10^{-3})}$ | $1.94_{\pm 0.03}$ | $2.01_{\pm 0.09}$ | $2.02_{\pm 0.01}$ | $1.84_{\pm 0.03}$ | $1.86_{\pm 0.03}$ | $2.08_{\pm 0.01}$ |
| FloodM1 $_{(\times 10^{-5})}$ | $6.64_{\pm 0.54}$ | $7.03_{\pm 1.87}$ | $8.71_{\pm 0.01}$ | $2.46_{\pm 0.30}$ | $2.69_{\pm 0.08}$ | $4.29_{\pm 0.99}$ |
| FloodM2 $_{(\times 10^{-5})}$ | $2.71_{\pm 0.17}$ | $2.49_{\pm 0.22}$ | $1.89_{\pm 0.02}$ | $2.75_{\pm 0.06}$ | $2.57_{\pm 0.26}$ | $4.85_{\pm 1.37}$ |
| FloodM3 $_{(\times 10^{-5})}$ | $6.58_{\pm 0.11}$ | $6.71_{\pm 0.10}$ | $7.79_{\pm 0.03}$ | $5.28_{\pm 0.80}$ | $3.86_{\pm 0.74}$ | $10.04_{\pm 0.73}$ |

Table 2: Mean squared error (the lower the better) on the time-series regression benchmarks Covid3Months is abbreviated Covid3M. FloodModeling is abbreviated FloodM.

tionally applies the same structure to the input pathway. The resulting architectures require only $O(N \log N)$ operations and $O(N)$ parameters, offering a hardware-oriented alternative to dense reservoirs. Experiments show that the structured variants maintain performance levels comparable to ESN and MF-ESN models on both classification and regression tasks.

Future work will address physical implementations on resistive substrates and explore extensions of the Hadamard-based mixing to modular reservoir architectures and structured state-space models.

## References

[1] K. Nakajima and I. Fischer. *Reservoir computing*. Springer, 2021.

[2] G. Milano, G. Pedretti, K. Montano, S. Ricci, S. Hashemkhani, L. Boarino, D. Ielmini, and C. Ricciardi. In materia reservoir computing with a fully memristive architecture based on self-organizing nanowire networks. *Nature materials*, 21(2):195–202, 2022.

[3] V. Pistolesi, A. Ceni, G. Milano, C. Ricciardi, and C. Gallicchio. A memristive computational neural network model for time-series processing. *APL Machine Learning*, 3(1), 2025.

[4] J. Dong, R. Ohana, M. Rafayelyan, and F. Krzakala. Reservoir computing meets recurrent kernels and structured transforms. *Advances in Neural Information Processing Systems*, 33:16785–16796, 2020.

[5] H. Jaeger, M. Lukoševičius, D. Popovici, and U. Siewert. Optimization and applications of echo state networks with leaky-integrator neurons. *Neural networks*, 20(3):335–352, 2007.

[6] A. Sebastian, M. Le Gallo, R. Khaddam-Aljameh, and E. Eleftheriou. Memory devices and applications for in-memory computing. *Nature nanotechnology*, 15(7):529–544, 2020.

[7] G. Milano, E. Miranda, and C. Ricciardi. Connectome of memristive nanowire networks through graph theory. *Neural Networks*, 150:137–148, 2022.