Mechanical Substrates for Analog Computing via Inverse-Designed Topological Optimization

P. Peralta-Braz[a], E. Atroshchenko[b], S.P.A. Bordas[a]

[a]*Department of Engineering, University of Luxembourg, Luxembourg*
[b]*School of Civil and Environmental Engineering, University of New South Wales, Sydney, Australia*

Abstract

The increasing demand for real-time, low-power data processing has renewed interest in analog computing paradigms that operate directly in the physical domain. In this work, we explore the use of mechanical substrates as computational units, designed through inverse topology optimization. By defining input gates as force applications and output gates as displacement measurements, computational tasks are encoded into the elastic response of the material. Our framework couples a numerical model with an optimization scheme, enabling the discovery of geometries that perform target operations. We illustrate the versatility of this approach through study cases including classification, function regression, and matrix–vector multiplication, demonstrating how stress propagation and tailored structural topologies can perform such tasks without digitization. These results highlight the potential of mechanical metamaterials as physical computing substrates, paving the way for ultra-low-power and embedded analog processors.

Keywords: Analog Computing, Topological Optimization, Metamaterials.

1. Introduction

In line with Moore's Law, processors have become faster, smaller, and more powerful over recent decades [1]. However, the recent growth of data-intensive applications, such as artificial intelligence (AI), has introduced new challenges in computational architecture. One of the key issues lies in the traditional von Neumann architecture, which physically separates the central processing unit (CPU) from memory. This separation requires constant data transfer between the two units. Nevertheless, memory access speeds have not kept pace with processor improvements, leading to a growing performance gap. This phenomenon, known as the memory wall, refers to the point at which delays and energy costs associated with data movement begin to dominate overall system performance [2]. On the other hand, the widespread adoption of real-time data, driven by applications in AI, Internet of Things (IoT), and autonomous systems, introduces significant challenges related to the digitization process itself. The continuous conversion of analog signals into digital form imposes constraints in terms of latency, energy consumption, and limited parallelization [3]. These issues become especially critical at the edge, where devices must operate under strict power and computational constraints [4]. As real-time data streams increase in volume and frequency, the inefficiencies of conventional digitization pipelines risk undermining the responsiveness and scalability of next-generation intelligent systems.

These issues have motivated the search for new computing paradigms that go beyond traditional solutions. In this context, analog computing (a concept once considered obsolete) has re-emerged as a promising alternative [5]. Today's revival of analog computing is being driven by advances in

metamaterials, which serve as catalysts for a new generation of physical information processing [3, 6]. Metamaterials (and their quasi-2D counterparts, metasurfaces) are artificially engineered materials possess extraordinary capabilities in ways that natural materials cannot. This field has seen explosive growth in both theoretical understanding and practical implementation driven by advances in materials science and fabrication technologies, and by the development of powerful numerical models that enable their precise design and optimization [7]. Their ability to tailor their response have garnered interest for computing applications, providing a natural substrate for analog, parallel, and in-material information computing. In particular, metamaterial-based analog computers unlock their full potential when placed directly at the source of information, processing signals in their raw analog form without the need of intensive digitization [3, 6].

This work proposes an integrated framework for analog computing that exploits mechanical substrates as physical processors, designed via topology optimization. By encoding inputs as applied forces and outputs as measured displacements, computational tasks are directly embedded in the elastic response of the structure. This approach bypasses the digitization bottleneck, reducing both latency and energy consumption while enabling in-material computation. The study evaluates optimized mechanical substrates using a numerical model coupled with an inverse-design procedure, and demonstrates their ability to perform fundamental operations such as classification, function approximation, and matrix–vector multiplication. The use of canonical tasks is motivated by the objective of establishing a benchmark scenario widely recognized in the research community, ensuring comparability with prior studies and robust validation of the results. The proposed framework is conceived as a pathway toward ultra-low-power, embedded computing at the edge.

The contributions of this paper can be summarised as follows:

- Propose a novel analog computing framework based on mechanical metamaterials, reducing reliance on digitization and enabling direct, energy-efficient signal processing in the physical domain.
- Demonstrate that stress propagation and tailored topologies can reliably execute computational primitives such as classification and matrix–vector multiplication.

The remainder of the paper is organised as follows. Section 2 presents the topology-optimization framework for the design of mechanical substrates. Section 3 illustrates implementations of the framework, including classification, function approximation, and matrix–vector multiplication. Finally, Section 4 provides concluding remarks.

2. Background

In this work, we focus on the development of mechanical analog computers, where information is processed through the physical response of a substrate engineered via topology optimization within a design space defined as $\Omega = L \times W$. Specifically, n_{in} input gates are defined as locations where external forces are applied to encode computational inputs for a given task, and n_{out} output gates, where the resulting displacements are measured and interpreted as computational outputs. For each computational task, the number and placement of input and output gates are customized according to the specific requirements of the inputs and outputs. As a representative example, it is considered a plate clamped at its two upper corners, as illustrated in Figure 1. The design of such analog computing substrates involves solving a learning problem, where the substrate topology is optimized so that, under prescribed inputs at the input gates, the measured outputs at the output gates closely match the desired computational operations.

$$\begin{aligned} &\min_{\mathbf{\Phi}} \mathcal{L}\{\mathbf{u}_0(\mathbf{x})\} \\ &\text{subject to } \mathbf{K}\mathbf{u} = \mathbf{F} \\ &\qquad V(x) = V_0 \end{aligned} \tag{1}$$

where $\mathcal{L}$ is the loss function, $\mathbf{u}$ is the displacement field in the structure, $\mathbf{u}_0$ denotes the displacements at the output gates, $\mathbf{K}$ is the global stiffness matrix, $\mathbf{F}$ is the force vector applied at the input gates, $V(x)$ is the volume, V_0 is a prescribed volume and $\mathbf{x}$ is the vector of design variables $\mathbf{x}$, that represents the material distribution within the design domain, where each component $x_e \in [0, 1]$ corresponds to the relative density assigned to the e-th finite element.

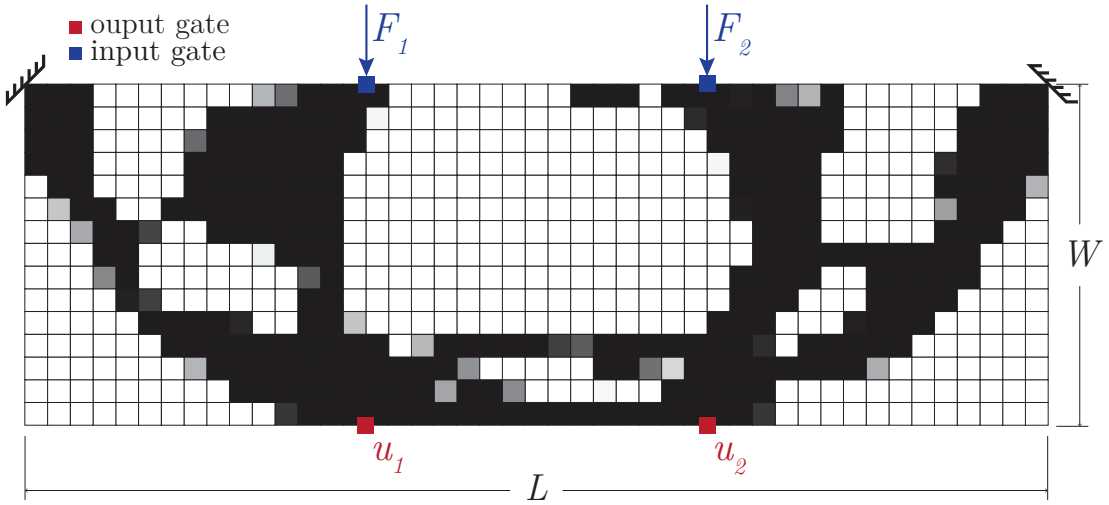


Figure 1: Schematic of the mechanical analog computing substrate, which is clamped at its two upper corners. The design domain Ω has dimensions $W \times L$. Forces are applied at the designated input gates (blue squares), encoding the computational inputs for the task. The resulting displacements are measured at the output gates (red squares), corresponding to the computational outputs.

This density variable governs the stiffness properties of the element through an interpolation scheme, typically relating the Young's modulus E_e of element e to the design variable [8] as,

$$E_e(x_e) = E_{\min} + x_e^p \left(E_0 - E_{\min}\right), \tag{2}$$

here E_0 is the Young's modulus of the solid material, $E_{\min}$ is a small stiffness value assigned to void regions to ensure numerical stability, and $p \geq 1$ is the penalization parameter to promote discrete (solid–void) designs.

For the design update, we rely on the Method of Moving Asymptotes (MMA) [9], a widely used gradient-based algorithm in topology optimization. At each iteration, MMA constructs a strictly convex and separable approximation of the objective and constraint functions using moving asymptotes, which ensures robustness and stability even in large-scale problems with thousands of design variables. The update scheme requires the objective and constraint gradients ($\partial\mathcal{L}/\partial x_e$ and $\partial V/\partial x_e$) with respect to the design variables, which in our case correspond to the sensitivities of the loss function.

The objective gradients $\partial\mathcal{L}/\partial x_e$ with respect to the element densities x_e can be conveniently calculated as,

$$\frac{\partial \mathcal{L}}{\partial x_e} = \frac{\partial \mathcal{L}}{\partial \mathbf{u}} \frac{\partial \mathbf{u}}{\partial x_e} = \mathbf{u}_{adj}^T \frac{\partial \mathbf{K}}{\partial x_e} \mathbf{u} \tag{3}$$

where $\mathbf{u}_{adj}$ is the adjoint displacement, which can be understated as "response" of the system under the adjoint force $-\left(\frac{\partial \mathcal{L}}{\partial \mathbf{u}}\right)^T$ [10], i.e.,

$$\mathbf{K}\mathbf{u}_{adj} = \mathbf{F} \tag{4}$$

While the adjoint formulation provides the classical route to derive sensitivities, in this work we compute all gradients using automatic differentiation (AD). Specifically, we employ JAX [11], a high-performance Python library that enables end-to-end differentiation of user-defined numerical programs. Following the framework proposed by [12], the derivatives of the loss function with respect to the design variables are automatically obtained by tracing the computational graph of the finite element solver. This approach eliminates the need for manual derivation of adjoint equations, reduces implementation complexity, and provides flexibility for incorporating non-standard material models, filters, or constraints.

3. Study Cases

To demonstrate the versatility of the proposed framework, several canonical tasks are considered as representative study cases. Matrix–vector multiplication, classification, and regression are selected because they constitute fundamental building blocks in machine learning and scientific computing, and also provide clear benchmarks for evaluating the effectiveness of the analog computing paradigm. Each case is formulated in terms of input–output mappings that are encoded in the mechanical substrate through the application of forces at input gates and the measurement of displacements at output gates. The following subsections present the implementation details and results for each case, highlighting the ability of optimized topologies to accurately reproduce the desired computational operations.

3.1. Matrix-vector multiplication

Matrix-vector multiplication (MVM) is a fundamental operation underpinning a vast array of modern algorithms in signal processing, machine learning, and scientific computing [13]. In this first study case, we demonstrate how our mechanical substrate can naturally perform matrix-vector multiplication by exploiting the intrinsic properties of stress propagation and engineered material responses. This approach realizes MVM directly in the physical domain, offering the benefit of energy efficiency by avoiding unnecessary data digitization and transmission. This study not only aims to highlight the computational versatility of the physical medium but also suggests that MVM can be the core building block for more advanced mechanical analog computing architectures.

The initial challenge in this study case is the appropriate definition of the loss function that will guide the optimization of design of the subtract for the compute of the MVM. In particular, it is considered a 2×2 matrix multiplication as illustrative example. Therefore, there are define two input gates, which encode the entries of the vector $\mathbf{P}$, and two output gates, where the resulting deformations correspond to the vector $\mathbf{d} = \mathbf{A}\mathbf{P}$ that we aim to realize analogically in the substrate. To fully characterize the matrix, it is used the standard basis vectors $\mathbf{P}_1 = [1, 0]^{\mathrm{T}}$ and $\mathbf{P}_2 = [0, 1]^{\mathrm{T}}$ as inputs, performing two separate simulations. By exploiting the linearity of the system, any linear combination of these two basis cases can reconstruct the result for an arbitrary input vector. Therefore, for each simulation i (corresponding to input vector $\mathbf{P}_i$), the displacement measured at output gate j should match the matrix element A_{ji} of the target matrix $\mathbf{A}$. This procedure, illustrated schematically

in Figure 2, follows the approach originally proposed in [14] and can be generalized to any size of matrices.

$$\mathcal{L}(\mathbf{u}_0(\mathbf{x})) = \sum_{i=1}^{2}\sum_{j=1}^{2}\left|u_{y,j}^{(i)}(\mathbf{x}) - A_{ji}\right|^2 \tag{5}$$

The outcome of this optimization is shown in Figure 3, which presents the resulting substrate topology tailored to perform the 2×2 matrix–vector multiplication. In this example, the objective was to replicate the target matrix

$$\mathbf{A} = \begin{bmatrix} 63 & 13 \\ 17 & 75 \end{bmatrix} \tag{6}$$

so that the analog displacements at the output gates reproduce the product $\mathbf{d} = \mathbf{AP}$. The optimized topology successfully channels stress propagation to achieve this mapping, with deviations below 0.5% from the desired outputs. These results validate the feasibility of embedding linear algebraic operations directly into the mechanical response of the structure, positioning optimized substrates as reliable building blocks for more complex analog computing tasks.

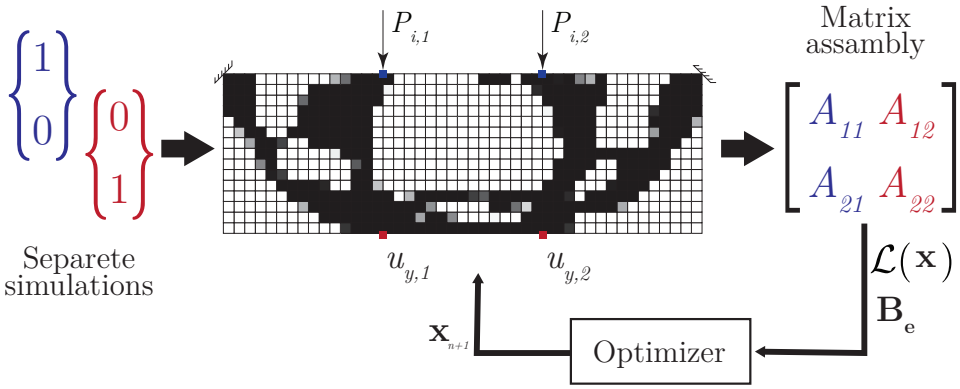
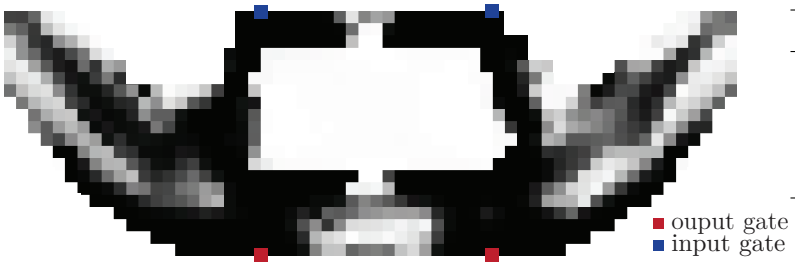


Figure 2: Schematic of the analog matrix-vector multiplication implementation in an elastic substrate. Separate simulations are performed for each basis vector. The resulting displacements at the output gates are used to define the loss function $\mathcal{L}(\mathbf{x})$. The optimizer updates the substrate design $\mathbf{x}$ by minimizing the loss function.



input vector		estimated output	target output	accuracy vector
1	$u_{1,y}$	62.94	63	99.91%
0	$u_{2,y}$	16.93	17	99.56%
0	$u_{1,y}$	12.98	13	99.85%
1	$u_{2,y}$	74.99	75	99.99%

Figure 3: Resulting optimized topology for the implementation of a 2×2 matrix–vector multiplication. Input gates (blue squares) encode the entries of the input vector, while output gates (red squares) provide the analog displacement response. The table compares estimated and target outputs for the basis vectors, with accuracies above 99.5%.

3.2. Classification task

One of the core tasks in machine learning is classification. It is a fundamental predictive modeling process in which the goal is to assign the correct label to given input data, essential for making

decision [15]. In the context of mechanical analog computing, classification is realized by encoding the input data as forces applied at designated input gates and interpreting the resulting displacements at output gates as decision variables. Through optimization, the substrate topology is tailored so that distinct input patterns produce clearly separated displacement responses corresponding to different classes, thereby embedding a physical decision boundary in the structure.

To demonstrate the classification capability of the mechanical substrate, the well-known Iris flower dataset [16] is employed, as it is widely recognized in machine learning and commonly adopted as a benchmark [17, 10, 18]. The dataset is characterized by four distinct features (sepal length, sepal width, petal length, and petal width), which are used to distinguish between three classes of Iris flowers: setosa, versicolor, and virginica. In the mechanical implementation, these four features are encoded as forces applied at four input gates, while three output gates are defined to represent the possible flower classes. The structure is optimized so that, for a given input vector, the output gate exhibiting the largest displacement is interpreted as the predicted flower type, as illustrated in Figure 4.

To complete the formulation, a loss function is introduced to guide the optimization process. The global loss function J is defined as the average cross-entropy [19] between the predicted probability distribution at the output gates and the one-hot encoded target labels. Accordingly, the loss function is given by,

$$\mathcal{L}(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} J^{(i)}(\mathbf{x}) \tag{7}$$

where the cross-entropy loss for the i-th training sample $\mathcal{L}^{(i)}(\mathbf{x})$ penalizes cases where the true class is assigned a low probability, thereby encouraging the design to produce clearly separable displacement responses. It is defined as,

$$J^{(i)}(\mathbf{x}) = -\sum_{k=1}^{n_{out}} y_k^{(i)} \log(p_k^{(i)}(\mathbf{x})) \tag{8}$$

here, $y_k^{(i)}$ denotes the one-hot encoded target label, represented as a binary vector of length $n_{out} = 3$, where all entries are zero except for the index corresponding to the correct class, which is set to one. Meanwhile, the class probabilities $p_k^{(i)}(\mathbf{x})$ are obtained from the displacements at the output gates through the softmax function as,

$$p_k^{(i)}(\mathbf{x}) = \frac{\exp(u_k^{(i)}(\mathbf{x}))}{\sum_{j=1}^{n_{out}} \exp(u_j^{(i)}(\mathbf{x}))} \tag{9}$$

The classification process was carried out by training the substrate with 70% of the dataset and testing it with the remaining 30%. The accuracy, defined as the percentage of correctly classified samples with respect to the total, reached 96.19% for the training set and 95.56% for the test set. These results demonstrate the ability of the optimized structure to generalize beyond the training data. The resulting topology obtained through this optimization is presented in Figure 4.

3.3. Regression tasks

Another fundamental tasks in machine learning is regression. It is a predictive modeling process in which the goal is to estimate a continuous output variable from given input data, providing a quantitative basis for decision-making and inference [15]. In the context of mechanical analog computing, regression is realized by encoding the input data as forces applied at designated input gates

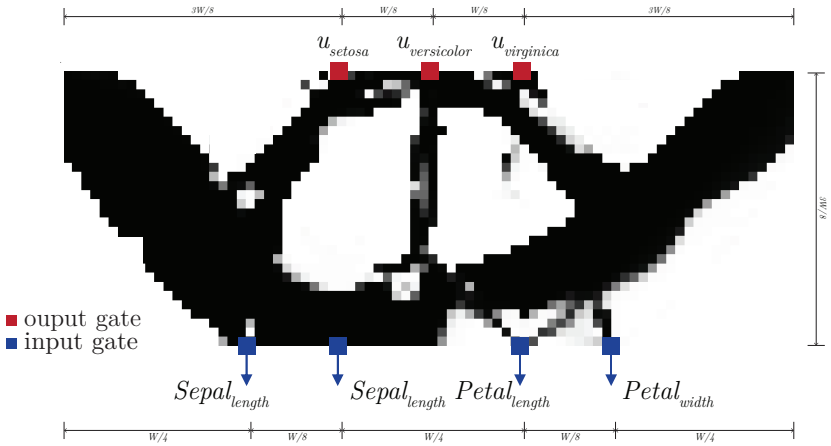


Figure 4: Resulting optimized topology for the classification task using the Iris flower dataset. The four input gates (blue squares) encode the features: sepal length, sepal width, petal length, and petal width, while the three output gates (red squares) correspond to the classes *setosa*, *versicolor*, and *virginica*. The predicted class is determined by the output gate exhibiting the largest displacement.

and interpreting the resulting displacements at output gates as continuous predictions. Through optimization, the substrate topology is tailored so that the analog responses closely approximate the target functional relationship, thereby embedding a physical regression model in the structure.

To illustrate the regression capability of the proposed framework, a representative case study is presented. The task consists of reproducing the solution of the one-dimensional Poisson equation with a constant load [20]. This problem is chosen because it provides a useful physical context and its solution can be conveniently adapted to different load magnitudes by applying proportionally scaled forces at the input gates. Although the system is linear, strategies can still be proposed to approximate nonlinear functions within the same framework. The governing equation is written as,

$$-u''(x) = f, \quad x \in (0,1), \tag{10}$$

subject to homogeneous Dirichlet boundary conditions

$$u(0) = 0, \qquad u(1) = 0 \tag{11}$$

For a constant load f, the analytical solution $u_{ana}(x)$ is given by

$$u_{ana}(x) = \frac{f}{2}\, x(1-x) \tag{12}$$

This functional form provides a convenient target for optimization, as it can be directly scaled to represent different load magnitudes.

In the mechanical analog implementation, the input gate is defined as the entire upper boundary of the substrate, where a concentrated force is applied at a position $x \in (0,1)$ proportional to the total width of the device. This spatial coordinate corresponds directly to the variable x in the one-dimensional Poisson equation. The regression output is taken as the vertical displacement at the midpoint of the lower boundary u_{output}. As shown in Figure 5, by varying the load position the displacement response traces the functional form of the analytical solution, effectively embedding a nonlinear mapping within the structure. In this way, the position of the applied force becomes an independent variable of the analog system, enabling the reproduction of the characteristic parabolic response of the Poisson equation.

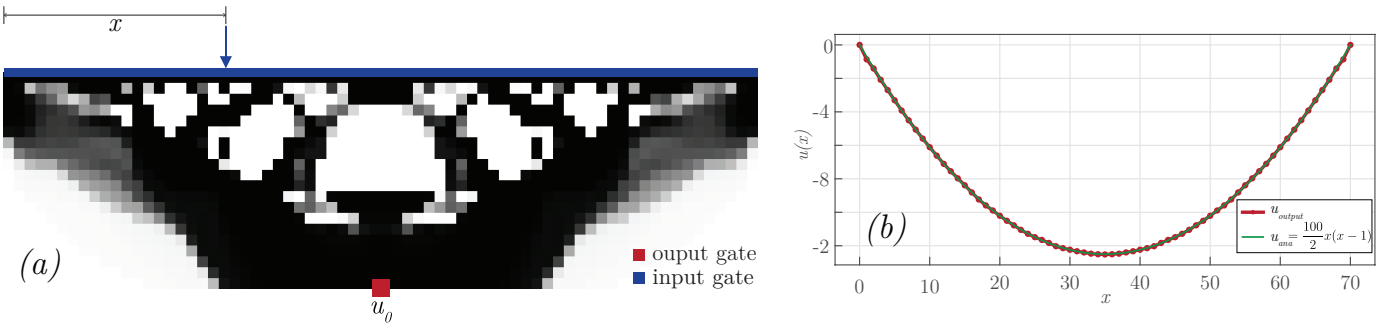


Figure 5: (a) Optimized topology for the regression task, where the input gate corresponds to the upper boundary and the output gate is located at the midpoint of the lower boundary. A concentrated force is applied along the upper edge at position x, proportional to the device width. (b) Comparison between the predicted displacement response of the optimized substrate and the analytical solution of the one-dimensional Poisson equation with constant load, with a total loss value of 0.07 over the independent simulations.

To train the mechanical substrate for this regression task, a loss function is defined based on the discrepancy between the predicted and analytical responses of the Poisson equation. Specifically, the system is evaluated through $n_{in} = 70$ independent simulations, in which the force is applied at equally spaced positions along the upper boundary. For each case, the predicted displacement at the output gate is compared to the analytical solution (Eq. 12). In addition, a symmetry constraint is imposed on the topology to ensure that the optimized structure remains symmetric with respect to the vertical axis. The loss function is then defined as the sum of squared errors across all simulations, providing a quantitative measure of how closely the substrate reproduces the target functional behavior:

$$\mathcal{L}(\mathbf{x}) \;=\; \sum_{i=1}^{n_{in}} \left(u^{(i)}_{\text{output}}(\mathbf{x}) - u^{(i)}_{\text{ana}} \right)^2 \tag{13}$$

The resulting optimized topology for the regression task is shown in Figure 5a, where the input gate corresponds to the upper boundary and the output displacement is measured at the center of the lower boundary. The optimization achieved a loss value of 0.07, which indicates a close agreement between the predicted and target responses, with an average error of only about 3% across the independent simulations. Figure 5b presents the comparison between the analog prediction and the analytical solution of the Poisson equation, showing an good agreement of the curves and confirming the ability of the optimized substrate to accurately reproduce the desired parabolic response.

4. Conclusions

This work introduced a framework for mechanical analog computing based on inverse-designed substrates optimized through topology optimization. By defining computational tasks in terms of applied forces at input gates and measured displacements at output gates, a direct mapping between physical responses and target operations was established.

Through three representative study cases—matrix–vector multiplication, classification, and regression, it was demonstrated that optimized mechanical substrates can reliably reproduce algorithmic primitives. The results showed that stress propagation and tailored geometries enable accurate analog execution of both discrete and continuous tasks, with classification accuracies above 95% and regression errors below 3%. These findings confirm the feasibility of embedding computation into the elastic response of a structure without reliance on digitization, offering a pathway toward ultra-low-power and in-material computing.

Beyond these examples, the proposed approach highlights the potential of mechanical metamaterials to serve as physical processors for more advanced operations. Future extensions may include the design of substrates capable of handling nonlinear tasks, scalable architectures combining multiple computational blocks, and experimental implementations to validate the numerical predictions. Collectively, this study positions optimized mechanical substrates as promising candidates for next-generation edge computing technologies.

References

[1] K. S. Kim, J. Kwon, H. Ryu, C. Kim, H. Kim, E.-K. Lee, D. Lee, S. Seo, N. M. Han, J. M. Suh, et al., The future of two-dimensional semiconductors beyond moore's law, Nature nanotechnology 19 (2024) 895–906.

[2] A. Gholami, Z. Yao, S. Kim, C. Hooper, M. W. Mahoney, K. Keutzer, Ai and memory wall, IEEE Micro 44 (2024) 33–39.

[3] F. Zangeneh-Nejad, D. L. Sounas, A. Alù, R. Fleury, Analogue computing with metamaterials, Nature Reviews Materials 6 (2021) 207–225.

[4] M. Shirvanimoghaddam, K. Shirvanimoghaddam, M. M. Abolhasani, M. Farhangi, V. Z. Barsari, H. Liu, M. Dohler, M. Naebe, Towards a green and self-powered internet of things using piezoelectric energy harvesting, Ieee Access 7 (2019) 94533–94556.

[5] B. Ulmann, Analog and hybrid computer programming, Walter de Gruyter GmbH & Co KG, 2023.

[6] A. Alù, A. F. Arrieta, E. Del Dottore, M. Dickey, S. Ferracin, R. Harne, H. Hauser, Q. He, J. B. Hopkins, L. P. Hyatt, et al., Roadmap on embodying mechano-intelligence and computing in functional materials and structures, Smart Materials and Structures 34 (2025) 063501.

[7] P. Jiao, J. Mueller, J. R. Raney, X. Zheng, A. H. Alavi, Mechanical metamaterials and beyond, Nature communications 14 (2023) 6004.

[8] E. Andreassen, A. Clausen, M. Schevenels, B. S. Lazarov, O. Sigmund, Efficient topology optimization in matlab using 88 lines of code, Structural and Multidisciplinary Optimization 43 (2011) 1–16.

[9] K. Svanberg, The method of moving asymptotes—a new method for structural optimization, International journal for numerical methods in engineering 24 (1987) 359–373.

[10] S. Li, X. Mao, Training all-mechanical neural networks for task learning through in situ backpropagation, Nature communications 15 (2024) 10528.

[11] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, Q. Zhang, JAX: composable transformations of Python+NumPy programs, 2018.

[12] A. Chandrasekhar, S. Sridhara, K. Suresh, Auto: a framework for automatic differentiation in topology optimization, Structural and Multidisciplinary Optimization 64 (2021) 4355–4365.

[13] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, nature 521 (2015) 436–444.

[14] C. Silva, G. Romano, Analog computing with heat: Matrix-vector multiplication with inverse-designed metastructures, arXiv preprint arXiv:2503.22603 (2025).

[15] A. Géron, Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow, " O'Reilly Media, Inc.", 2022.

[16] R. A. Fisher, Iris, UCI Machine Learning Repository, 1936. DOI: https://doi.org/10.24432/C56C76.

[17] M. Moghaddaszadeh, M. Mousa, A. Aref, M. Nouh, Mechanical intelligence via fully reconfigurable elastic neuromorphic metasurfaces, APL Materials 12 (2024).

[18] T. Fu, Y. Zang, Y. Huang, Z. Du, H. Huang, C. Hu, M. Chen, S. Yang, H. Chen, Photonic machine learning with on-chip diffractive optics, Nature Communications 14 (2023) 70.

[19] L. Li, M. Doroslovački, M. H. Loew, Approximating the gradient of cross-entropy loss function, IEEE access 8 (2020) 111626–111635.

[20] L. C. Evans, Partial differential equations, volume 19, American mathematical society, 2022.