Proceedings of the 14th The ASME 2020 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE

DETC2020-XXXXX

Single MEMS reservoir computer

Mohammad H Hasan
Mechanical Engineering Department
University of Nebraska at Lincoln
mohammadhhasan@huskers.unl.edu
Lincoln, NE, USA

Fadi Alsaleem
School of Architectural Engineering
University of Nebraska at Lincoln
falsaleem2@unl.edu
Omaha, NE, USA

ABSTRACT

In this work, we show the computational potential of MEMS devices by predicting the dynamics of a 10th order nonlinear auto-regressive moving average (NARMA10) dynamical system. Modeling this system is considered complex due to its high nonlinearity and dependency on its previous values. To model the NARMA10 system, we used a reservoir computing scheme by utilizing one MEMS device as a reservoir, produced by the interaction of 100 virtual nodes. The virtual nodes are attained by sampling the input of the MEMS device and modulating this input using a random modulation mask. The interaction between virtual nodes within the system was produced through delayed feedback and temporal dependence. Using this approach, the MEMS device was capable of adequately capturing the NARMA10 response with a normalized root mean square error (NRMSE) = 6.18% and 6.43% for the training and testing sets, respectively. In practice, the MEMS device would be superior to simulated reservoirs due to its ability to perform this complex computing task in real time.

INTRODUCTION

In the age of the internet of things (IoT), processors are expected to perform complex computational processes on large amounts of data in real time. These processes, such as classification, clustering and image processing were typically performed through artificial neural networks (ANN), simulated using digital computers. While successful in some of these tasks, ANNs reacted to each of their inputs separately without considering the inputs past values. Recurrent neural networks (RNNs) [1] were introduced as means to incorporate memory in ANNs by altering ANNs from functions to dynamical systems, where the current state is influenced by the network inputs and the network past states.

While RNNs offer the benefit of implementing colocalized memory and computation in the system, they are much harder to train than simple ANNs. This complexity arises from the bifurcations in the system due to parameter changes in training [2] and the ineffectiveness of the gradient descent method in systems with long-term dependencies [3]. Reservoir computing (RC) [4] was introduced as a means to utilize RNNs while bypassing their training dilemma.

A classical reservoir computing scheme is composed of a network of interconnected nonlinear neurons (like RNNs), coupled through random coupling weights, as shown in Fig.1. The RNN network is named a reservoir. Using this reservoir, inputs are nonlinearly transformed into higher dimensional space to produce the reservoir output signal in form of a vector quantity. This vector is multiplied with a weight matrix to produce the system outputs. The nonlinear transformation within the reservoir offers great nonlinear richness in the system and can be used for computational problems, such as classification [5]. Because of this dynamical richness, RCs are much easier to train than classical RNNs as the only weights in need of training in the system are located in the external weight matrix, which can be trained via simply linear regression.

Classical RCs found great success as approximators of chaotic systems [6] and as biologically-inspired controllers in insect inspired robots [7,8]. However, despite their great potential, reservoir computers are computationally expensive to simulate, due to the large number of nonlinear components in the reservoir. Recent works have shown the potential to solve this problem by introducing a reservoir of virtual nodes using a single dynamical device [9] such as a MEMS resonator [10]. In this work, we utilize an electrostatic MEMS device to simulate the response of a 10th order nonlinear auto-regressive moving average (NARMA10) to investigate the potential of MEMS reservoirs as dynamical system approximators in a prediction task rather than a classification task.

The organization of this paper is as follows: first, we introduce the concept of a reservoir of virtual nodes. Next, we introduce the NARMA10 benchmark followed by the mathematical model of the MEMS reservoir and its parameters. Afterwards, we discuss the system training and its results. Finally, we discuss the results and offer our concluding remarks.

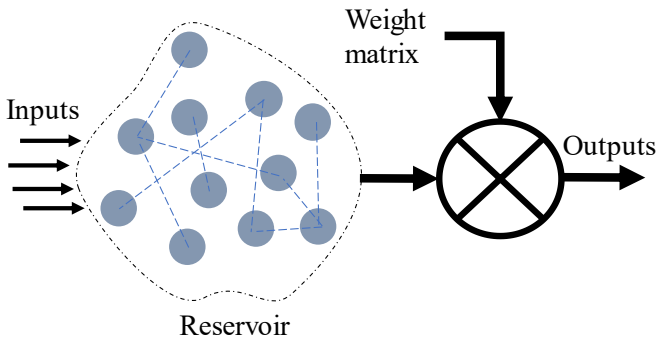

Figure 1. Classical reservoir computing scheme.

RESERVOIR OF VIRTUAL NODES

RC reservoirs may be formed using a single nonlinear dynamical systems by creating temporal virtual nodes rather than physical nodes. To achieve this, the network requires two additions:

1. Modulating the input signal to actuate the temporal nodes.
2. Utilizing a delayed feedback to couple the temporal nodes.

Modulating the input signal, *u(t)*, is achieved through the following process (Fig.2): first, the signal is sampled and held with a sampling time *T*. This produces the discretized input signal *I(t)*. Next, at each *T*, the input is modulated using a randomly initialized weight mask *m(t)*. The weight mask contains *N* weights, each related to a virtual node within the reservoir. The temporal separation between the random mask inputs is $\theta = T/N$. It is essential that the system remains transient to maintain coupling between adjacent modes. Therefore, θ is chosen such that it is close to be lower than the system's time constant (or characteristic time), τ. The discretized input signal is finally multiplied by the random weight mask to produce the modulated input, *J(t)*, which represents the input to the individual nodes.

The signal transformations of Fig.2 are shown in Fig.3, demonstrating the original signal, *I(t)*, in Fig.2(a), the sampled input, *u(t)*, in Fig.2(b), the modulation mask, *m(t)*, in Fig.2(c) and the modulated input, *J(t)* in Fig.2(d). The modulated input *J(t)* is used as a part of the MEMS driving signal [9, 10] within the computational stage of the RC scheme (Fig.4). The MEMS deflection, *x(t)*, represents the response of the reservoir. Because *J(t)* is modulated using a mask with a temporal separation of θ, the MEMS deflection at intervals multiple of θ represents the response of a temporal node with a total of *N* nodes distributed within a time span of *T*. The virtual nodes are coupled together using a delayed feedback in the system. Here, the MEMS deflection is delayed by *T* and multiplied by a feedback gain, α. The feedback offers interactions between the different MEMS nodes and offers additional memory for the system.

As MEMS device may be driven quasi-statically, at or around the primary resonance or at secondary resonances. The operational regime is controlled by using an AC voltage at the chosen resonance. The signal modulator in Fig.4 is used to facilitate the use of the MEMS within the operational range and to maintain transience. Thus, the overall effective DC voltage V_{DC} acting on the MEMS device is given by (1):

$$V_{DC} = V_b + \alpha x(t - T) + J(t) \quad (1)$$

where V_b is some bias voltage signal and α is the delayed feedback gain.

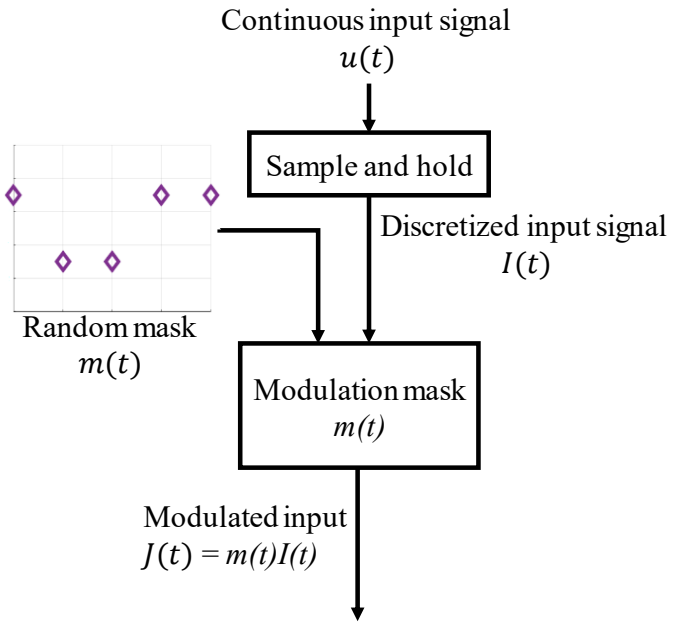

Figure 2. Input stage of the reservoir computing setup.

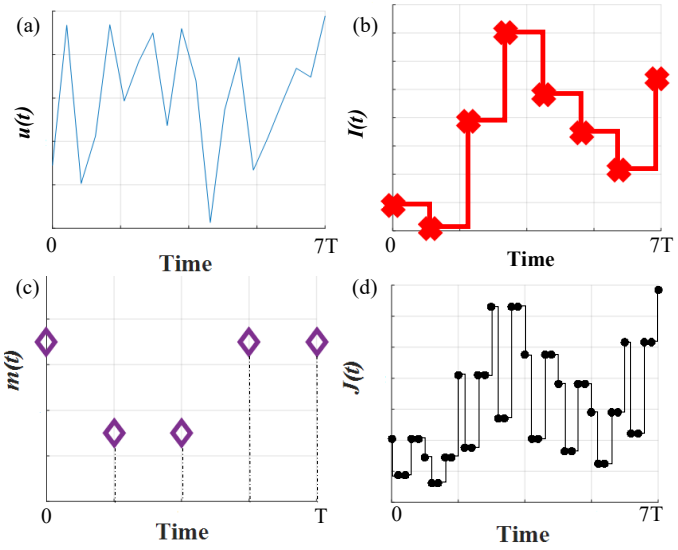

Figure 3. Signal transformation within the input stage of the RC scheme. (a) Analog input u(t). (b) Discretized input I(t). (c) Modulation mask m(t). (d) Modulated input J(t).

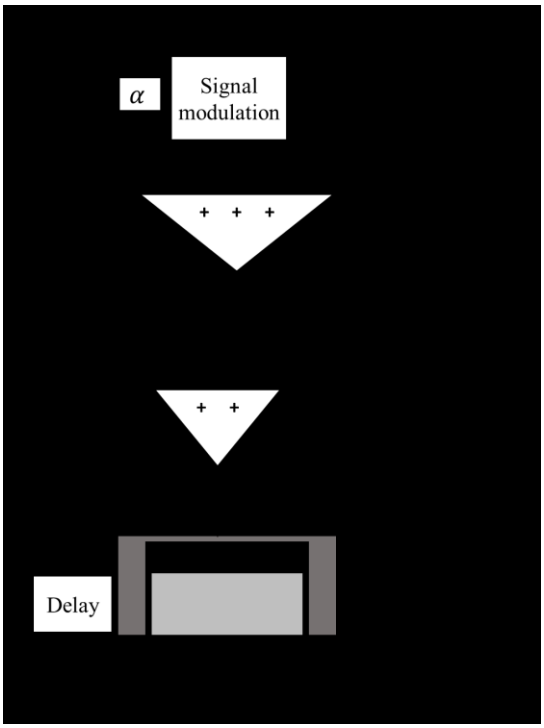

Figure 4. Computational stage of the MEMS reservoir computing setup.

Finally, after all inputs are processed, the deflections matrix *X* is formed by sampling the MEMS response, *x(t)*, a rate of one sample per θ seconds in the output stage of the RC scheme (Fig.5). This matrix has a size *M*×*N* where *N* is the number of virtual nodes and *M* is the number of time steps.

The RC output matrix *S (M×R)* is produced by multiplying the deflection matrix and the weights matrix *W*, as shown in (2):

$$S = X\,W \qquad (2)$$

Here, *W* is an *N*×*R* matrix and *R* is the number of outputs in the system. We note here that, for an appropriately large reservoir, *W* is the only matrix that requires training in the system. Moreover, *W* can simply be trained using linear regression, following (3) [11]:

$$W = (X^T X + k\,I)^{-1} * (X^{-1} Y) \qquad (3)$$

Where $k \cong 0$ is a constant used for regularization, $I = \delta_{ij}$, $i,j = 1,2, \ldots N$ is the identity matrix, *Y* is the expected output matrix with a size *M*×*R* and the $(.)^T$ operator is the transpose operator.

TEST BENCH MARK: NARMA10

As a case study for the MEMS reservoir computer, we study the potential of predicting the response of a tenth order discrete time NARMA (nonlinear auto-regressive moving average) dynamical system [12] using a simple MEMS reservoir. A NARMA10 dynamical system is modelled using (4):

$$y_{k+1} = 0.3y_k + 0.05y_k[\sum_{i=0}^{9} y_{k-i}] + 1.5u_k u_{k-9} + 0.1 \qquad (4)$$

Where y_k is the k^{th} NARMA state and u_k is the input at the k^{th} time step. Following [9], *u* is chosen to be a random number such that $u_k \in [0,0.2]$. y_k is complicated to fit due to the influence of past values on future responses, which makes this problem a compelling benchmark for nonlinear approximators.

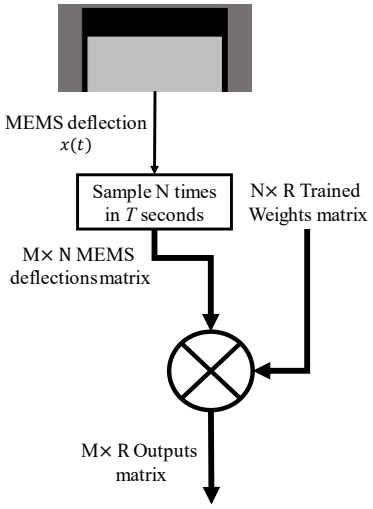

Figure 5. output stage of the MEMS reservoir computing setup.

MEMS RESERVOIR COMPUTER FOR NARMA10 APPROXIMATION

In this work, a commercial double-cantilever electrostatic MEMS device was used to create the system reservoir. The parameters of this MEMS device are found in Table 1 and a schematic of this MEMS is found in Fig.6. The MEMS in-plane dimensions are quite large for a MEMS device. However, because of its relatively small electrode separation gap (*d*), the MEMS device retains the same characteristics as smaller MEMS devices, as was shown in a previous work [13].

The deflection of the MEMS device due to electrostatic forcing is given by (5):

$$m\ddot{x}(t) + c(x)\,\dot{x}(t) + k\,x(t) = \frac{\varepsilon L b V_{MEMS}^2}{2(d - x(t))^2} \qquad (5)$$

where *x(t)* is the MEMS deflection as a function of time (*t*), *c(x)* is the nonlinear squeeze film damping coefficient and V_{MEMS} is the voltage across the MEMS device, given by (6):

$$V_{MEMS} = V_{AC} Cos(2\pi f t) + V_{DC} \qquad (6)$$

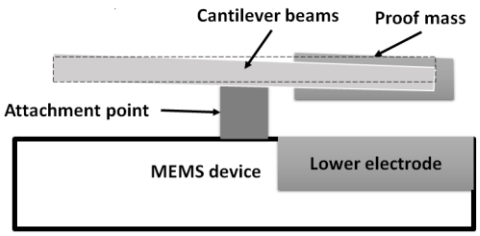

Figure 6. MEMS side-view schematic.

where V_{AC} is the amplitude of the AC voltage *f* is the AC driving frequency and V_{DC} is the DC voltage amplitude, given by (1) in

our setup. The nonlinear squeeze film damping can be calculated using (7)-(11) by adapting the Blech model [14, 15]:

$$\lambda_a = \lambda_0 P_0 / P_a \quad (7)$$

$$Kn = \lambda_a / d \quad (8)$$

$$\eta_{eff} = \eta / (1 + 9.638 Kn^{1.159}) \quad (9)$$

$$\sigma(x) = \frac{12 Lb\omega \eta_{eff}}{P_a (d-x)^2} \quad (10)$$

$$c(x) = \frac{64\sigma(x) P_a Lb}{\pi^6 \omega (d-x)} \frac{1+\beta^2}{(1+\beta^2)^2 + \frac{\sigma^2}{\pi^4}} \quad (11)$$

Where λ_a and λ_0 are the mean free path of gas modelcules at the ambient pressure P_a and atmospheric pressure P_o, respectively, K_n is the Knudsen number, η and η_{eff} are the nominal and effective viscosities of air, respectively, $\omega = 2\pi f$ is the MEMS vibrational angular rate, $\beta = b/L$ is the microbeam aspect ratio, set to unity following [13] and $\sigma(x)$ is the squeeze number.

Table 1: MEMS parameters

Parameter	Value	Definition
L	9 mm	Microbeam length
b	4.4 mm	Microbeam width
m	143 mg	MEMS effective mass
k	215 N/m	Microbeam linear stiffness
d	42 μm	Unactuated gap seperation
ε	8.85×10^{-12} F/m	Electrical permittivity

In this work, the MEMS device is operated at a pressure of 20 Pa using V_b = 30 V with no AC voltage. The delay value, T was chosen to be 0.9s and the feedback gain α = 0.1 V/μm was used. The modulation mask was chosen to be composed of a random sequence of $\pm$0.5 and 0. To ensure sensitivity to inputs, *J(t)* was linearly scaled linearly 5 times. The reservoir in this work was composed of *N = 100* virtual nodes with θ = 0.002 ms.

The NARMA10 simulation in this work were carried out using a sequence of 6000 random inputs (*M= length(u) = 6000*). To ensure good linear fitting, the number of time steps, *M,* must be chosen such that *M > N*.

TRAINING AND RESULTS

We use the sequence of 6000 random inputs u to construct the NARMA10 response Y, which represents the target response for the system. The random input is also fed to the MEMS device after modulation to drive the system. The MEMS response to a random input array is shown in Fig.7. The response loses its periodicity due to the input modulation and delayed feedback, which is desirable to perform calculations. The response of the MEMS device is sampled at a period of $\theta = 2$ ms and stored in a matrix X. This matrix is split into a training matrix and testing matrix as follows: the first 2000 rows of X are discarded to eliminate the effect of initial conditions, the next 2000 rows of X are chosen as the training set and the final 2000 rows of X are chosen as the testing set. The training set is used to train the Weight Matrix, W, through linear regression. Equation 3 was used in the training process while setting $k = 1\times10^{-21}$.

The performance is evaluated by calculating the normalized root mean square error (NRMSE) as shown in (12):

$$NRMSE = \sqrt{\left(\frac{1}{M}\frac{\sum_{i=1}^{M}(s_i - y_i)^2}{(\bar{y})^2}\right)} \quad (12)$$

where s_i and y_i are the ith element of the concatenated RC output matrix S and expected output matrix Y, respectively, and $\bar{y}$ is the mean of the vector Y.

Using linear regression (3) to train the weights of the MEMS reservoir using the training set yields NRMSE = 6.18%. The fitting results are shown in Fig.8 by comparing the results of NARMA10 to the results of the MEMS reservoir using the training set again as a test set. Next, the trained weights matrix was tested using the test set (Fig.9). The result of the test set is NRMSE = 6.43% which is predictably higher than NRMSE from the training set. However, it remains within an acceptable range.

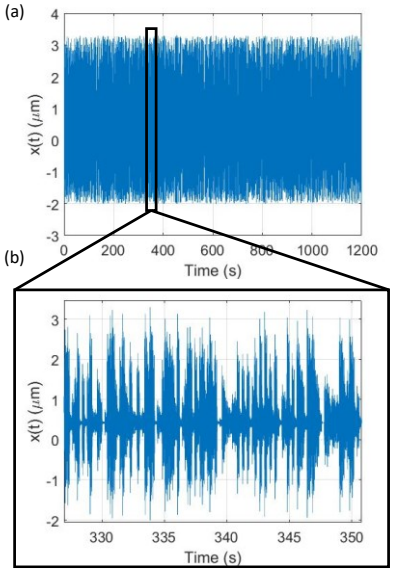

Figure 7. (a) sample MEMS response to a random input, u(t). (b) zoomed view of response.

DISCUSSION AND CONCLUSION

We show the use of a single MEMS as a reservoir to approximate the response of a NARMA10 system. NARMA10 is considered a standard benchmark for nonlinear approximators due to its complexity and high dependence on its time history information. Therefore, the nonlinear approximator requires memory. Dynamical systems retain memory as their future inputs rely on their past state. Here, we utilize a dynamical system (A MEMS

device) to model another complex dynamical system (NARMA10).

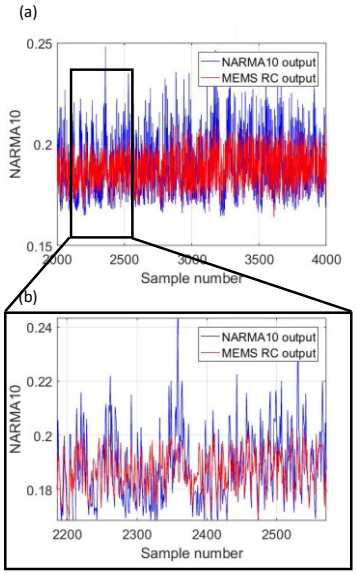

Figure 8. NARMA10 approximation (training set). (a) Full view. (b) Zoomed view.

The MEMS device operates as a reservoir of *N* nodes by creating temporally separated virtual nodes. This is achieved using the modulation mask *m(t)*. The interaction between nodes occurs due to the delayed feedback used in the reservoir circuit, which also allows past states to visibly influence the MEMS response. Another means of interaction between adjacent nodes occurs automatically through the reliance of each node on the information of previous nodes by virtue of the time-dependence of dynamical systems. We note here that MEMS devices reach a stable limit cycle when actuated, using moderate AC and DC voltages excitation, after passing through a brief transient state. If the MEMS device is allowed to reach the stable periodic region, the system loses its time dependence, which decouples adjacent modes. To avoid this issue, the separation time between nodes (θ) is chosen such that it is smaller than the characteristic time (time constant) of the MEMS ($\theta < \tau$).

In this work, using a MEMS device to form a reservoir with 100 nodes resulted in a NARMA10 fitting with NRMSE = 6.18% using the training set and 6.43% using the testing set.

We note here that simulating a virtual reservoir is extremely computationally expensive due to the complexity of the delayed differential equation of the MEMS device. However, using a real MEMS device would theoretically allow us to perform this computation in real time, which is not possible using digital computing currently.

MEMS devices can offer further attractive properties that allow them to serve as excellent reservoirs such as the existence of multiple states that can be accessed simultaneously within the MEMS device by relying on each of the MEMS modeshapes as a state rather than simply studying the deflection of the MEMS device. MEMS devices are also capable of sensing external inputs. This may enable MEMS devices to perform as dedicated sensing-and-computing units simultaneous. We aim to explore these concepts in a future work.

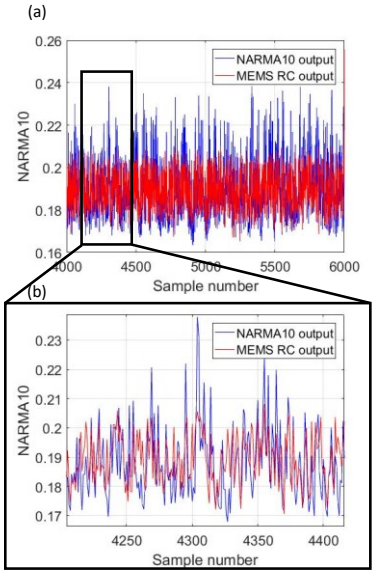

Figure 9. NARMA10 approximation (test set). (a) Full view. (b) Zoomed view.

REFERENCES

1. Medsker, L., & Jain, L. C. (1999). Recurrent neural networks: design and applications. CRC press.
2. Doya, K. (1992, May). Bifurcations in the learning of recurrent neural networks. In [Proceedings] 1992 IEEE International Symposium on Circuits and Systems (Vol. 6, pp. 2777-2780). IEEE.
3. Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. IEEE transactions on neural networks, 5(2), 157-166.
4. Lukoševičius, M., & Jaeger, H. (2009). Reservoir computing approaches to recurrent neural network training. Computer Science Review, 3(3), 127-149.
5. Cover, T. M. (1965). Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. IEEE transactions on electronic computers, (3), 326-334.
6. Jaeger, H., & Haas, H. (2004). Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. science, 304(5667), 78-80.
7. Dasgupta, S., Goldschmidt, D., Wörgötter, F., & Manoonpong, P. (2015). Distributed recurrent neural forward models with synaptic adaptation and CPG-based control for complex behaviors of walking robots. Frontiers in neurorobotics, 9, 10.
8. Arena, E., Arena, P., Strauss, R., & Patané, L. (2017). Motor-skill learning in an insect inspired neuro-

computational control system. Frontiers in neurorobotics, 11, 12.
9. Appeltant, L., Soriano, M. C., Van der Sande, G., Danckaert, J., Massar, S., Dambre, J., ... & Fischer, I. (2011). Information processing using a single dynamical node as complex system. Nature communications, 2, 468.
10. Dion, G., Mejaouri, S., & Sylvestre, J. (2018). Reservoir computing with a single delay-coupled non-linear mechanical oscillator. Journal of Applied Physics, 124(15), 152132.
11. Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. Technometrics, 12(1), 55-67.
12. Atiya, A. F., & Parlos, A. G. (2000). New results on recurrent network training: unifying the algorithms and accelerating convergence. IEEE transactions on neural networks, 11(3), 697-709.
13. Alsaleem, F. M., Younis, M. I., & Ouakad, H. M. (2009). On the nonlinear resonances and dynamic pull-in of electrostatically actuated resonators. Journal of Micromechanics and Microengineering, 19(4), 045013.
14. Blech, J. J. (1983). On isothermal squeeze films. Journal of lubrication technology, 105(4), 615-620.
15. Veijola, T., Kuisma, H., Lahdenperä, J., & Ryhänen, T. (1995). Equivalent-circuit model of the squeezed gas film in a silicon accelerometer. Sensors and Actuators A: Physical, 48(3), 239-248.