Structural Response Prediction of a Flat Beam with Geometric Nonlinearities with Temporal Convolutional Networks

David Najera and Chin Lee
University of California San Diego
9450 Gilman Drive, La Jolla, CA
dnajera@eng.ucsd.edu, chl083@ucsd.edu

Abstract

Complex structural systems are ubiquitous in the aerospace industry. The structural response characterization of these complex systems and components often involves extensive nondestructive testing. When testing is not an option, analysts rely on numerical simulations to predict the structural response. These simulations are performed by solving the partial differential equations (PDE) of motion. In the presence of nonlinearities, solving the relevant PDEs becomes computationally intractable for real systems. Furthermore, the physical parameters required to perform these simulations may be unknown or hard to obtain. To overcome these challenges, we propose to train a data-driven model that can be used in place of more expensive simulations. The focus of this paper will be the exploration of the Temporal Convolutional Network (TCN). The TCN has been demonstrated to be a powerful architecture in the field of speech recognition and speech-to-text generation but it has not been studied for the application to structural response. The TCN model is enhanced with physical constraints that act as regularizers by leveraging relevant domain-knowledge. The TCN performance is compared to the traditional autoregressive model which is a popular approach in this field. Error metrics are evaluated in the time and frequency domain to assess the performance of the methods considered.

1. Introduction

Complex structural systems are ubiquitous in several industries, such as aerospace, civil, and automotive. The structural response characterization of these complex systems and components often involves extensive nondestructive testing. When testing is not an option, analysts and designers rely on numerical simulations to predict the structural response. These simulations are performed by solving the partial differential equations (PDE) of motion given by the following equation.

$$M\ddot{x} + C\dot{x} + Kx + f_{nl}(x) = f_{ext}$$

Where M, C, and K are the mass, damping, and stiffness matrices, respectively. These system matrices embed the material properties and geometrical features of the structure. f_{nl} is the nonlinear force vector, which is often unknown and it depends upon the state variables, and f_{ext} is the applied external force, usually known.

In the presence of nonlinearities (e.g., $f_{nl}(x)$), solving the relevant PDEs becomes computationally intractable for real systems subjected to long-duration environments due to the computational requirements of nonlinear solvers. However, these analyses are critical for determining the structural response of structural components to prolonged excitation, such as the random vibration loads due to aerodynamic pressure acting on an airplane. Even when these models can be run on high performance clusters, there may be deficiencies in the physical model due to epistemic uncertainty (e.g., an unknown dissipative force acting on the system).

In this paper, we propose the use of the Temporal Convolutional Network (TCN) to predict structural response of a nonlinear component. We adapt the WaveNet architecture [11], originally developed for speech generation, to the problem of structural dynamics. We draw inspiration from linear parametric theory, where autoregressive filters are used to model linear time processes. In doing so, we expand upon the existing TCN architecture to enable an autoregressive component, as well as an external input source. Furthermore, we explore the benefits of adding physical constraints to the network via regularization. Finally, we compare the accuracy of the predictions by computing the mean squared error (MSE) and the Pearson correlation coefficient (CC).

The following sections will describe the traditional autoregressive model and introduce the TCN.

1.1. Linear Parametric Models

The task of formulating a model to reproduce observed data falls under the category of system identification. The most common and well-known method for parametric system identification is the least-squares method. In this algorithm, a model is assumed to be a linear function of external variables and past outputs. The model is formulated by performing linear regression by considering a concatenation of past outputs and external inputs to form the regressor vector. This paradigm forms the basic model structure for the Auto-Regression with eXtra inputs (ARX) [7]. The ARX model assumes that the observed data came from a time-invariant linear system. For this reason, this family of linear models is often inadequate for modeling nonlinear models. In this paper, we will compare our proposed method to the ARX model. It should be noted that nonlinear extensions of the ARX method exist [7] but are not explored here due to the complexities associated with the formulation of these models.

1.2. Deep Neural Network for Sequence Learning

Deep neural networks have demonstrated strong performance improvements over traditional time series models and have increasingly been used in sequence learning [4]. Machine learning practitioners often regard Recurrent architectures as the default architecture for time series forecasting. However, recent research in deep learning shows that convolution architectures can produce state-of-the-art performance and even outperform recurrent models in specific tasks [1]. Seeing success in Natural Language Processing tasks, Transformer-based models also gain popularity in time series forecasting. Unlike sequence-aligned models, transformer models process the entire sequence and learn dependencies through the self-attention mechanism. In this paper, we experimented with WaveNet, a variant of TCN that was originally used to model raw audio waveforms [1]. Our experiments show that TCN outperformed the traditional ARX model and better captures the nonlinearities in our data.

2. Methodology

In previous work, [8] used a Long-Short-Term Memory (LSTM) network to model response of a nonlinear system to random vibration excitation. It was shown that it was important to use a recurrent network to account for the history dependent nonlinear response. In the current work, a TCN was used instead. The main advantage of the TCN is efficiency; it is faster to train and it is less memory intensive. Other advantages cited by [1] include parallelism, flexible receptive field, stable gradients, and variable input lengths. The TCN is based on the work by van den Oord et al. [11] whose work was focused on image completion from occluded data. The original concept of a recurrent neural network with 2D convolutions (PixelRNN) was adapted to machine translation tasks by [12]. Their work provides the basis for this research.

The TCN preserves the main features of the LSTM: the convolutions are causal (i.e., information flows from past to future only), and the architecture maps a sequence of any length to an output sequence of the same length. In contrast, the TCN used does not use any gating mechanisms. The TCN uses a 1D fully-convolutional architecture where all hidden layers are the same length. Sequence length preservation is accomplished by zero-padding subsequent layers after convolutions. The TCN convolves the element at time step t_i with elements $t_{i-k} \dots t_{i-1}$, where k is the length of the reception field (i.e., how far to look back in time).

In past work, sequence modeling has been mainly employed for text generation and music modeling where the sequence is modeled by the mapping function:

$$y_i = F(x_i, ..., x_{i-T})$$

where y is the target sequence and x is the source sequence.

The predicted output only depends on the present and past inputs, and not on any future information. To account for history dependence, the past outputs are added to the sequence following the general structure of the NARMAX model [3].

$$y_i = F(x_i, ..., x_{i-T}, y_{i-1}, ..., y_{i-T-1})$$

In the following equations we denote the input simply by x, noting that x includes both the current input, and the past response.

Following previous work by Bai *et al*. [1], we apply dilated convolutions to enable an exponentially large receptive field. The dilated convolution is defined by

$$F(s) = (x *_d f)(s) = \sum_{i=0}^{k-1} f(i) x_{s-di}$$

where d is the dilation factors and k is the filter size.

The TCN is composed of residual blocks where dilated convolutions occur. A diagram of the residual block architecture used is shown below in Figure 1. The residual block has two outputs: a feature map that gets fed into the next residual block, and a skip connection which is aggregated in the end. The overall architecture is illustrated in Figure 2. The data is first passed through a 1x1 convolutional layer to expand it to Nc number of channels –this transformation is analogous to decomposing the original signal into Nc components, though there are no constraints on the attributes of the decomposition. The data is then passed through residual blocks. The outputs of the residual functions are then aggregated and passed through the ReLU function. Lastly, the output is passed through a linear dense layer to get the data into the right dimensions. The last linear operation is analogous to linear superposition of the transformed time signals.

The convolution operation resembles the Duhamel integral over a finite time segment. Thus, the convolution operation is analogous to transforming the input using the finite impulse response function [7]. However, the TCN also considers the past output as input to the convolution, adding time variance to the network. Moreover, the convolved input is further transformed by additional nonlinear operations. For example, consider the output of the network in the presence of only two residual blocks and Nc channels, ignoring the bias, the diluted convolution, and the normalization terms.

$$\sum_{k=1}^{Nc} \beta_k \{ReLU\{\sum_{\tau} f_k^{(2)}(t-\tau)$$
$$ReLU\{\sum_{\tau} f_k^{(1)}(t-\tau)x(t,\tau) + x(t,\tau)\}$$
$$+ReLU\{\sum_{\tau} f_k^{(1)}(t-\tau)x(t,\tau)\}\}\}$$

Where τ is the time-delay associated with the convolution and β_k is the output layer weight vector. The equation above represents an extension of the linear input-output relation of the system. In this augmented mapping, the convolution kernel can be any arbitrary differentiable nonlinear function. Furthermore, the network is time variant and assumed to have fading memory; the width of the memory window is controlled by the reception field of the network. The coefficients of the kernels and the nonlinear functions are found using the established Adam optimizer [6]. Regularization is handled by using spatial dropout to zero out entire feature channels to promote independence among features [9].

In addition to dropout, we also implement a physical constraint that operates as a regularizer, shown below as $L_{physics}$. This loss term evaluates the error from a multi-step integration scheme, where s represents the number of steps. The k functional is a function of the time derivative of $y(t)$. In this paper, we predict the displacement, velocity, and acceleration response and relate them to each other by evaluating the $L_{physics}$ term. This loss term ensures that the the predictions have physical interpretations. The coefficients b_i are obtained from a Butcher's tableau [2] via the nodepy package [5]. In the equation below, $\hat{y}_i$ is the predicted output and Δt is the time step. α is the weight associated with he $L_{physics}$ term. It was found that $\alpha = 1$ resulted in good performance.

$$L_{MSE} = 1/N \sum_{i}^{N} (\hat{y}_i - y_i)^2$$
$$L_{physics} = y_{t+1} - y_t + \Delta t \sum_{i}^{s} b_i k_i$$
$$Loss = L_{MSE} + \alpha L_{physics}$$

To illustrate the interpretation of the $L_{physics}$ term, consider the case when $s = 1$. In that case $b_1 = 1$, and the equation reduces to:

$$L_{physics} = y_{t+1} - y_t + \Delta t k$$

If we let $k = dy/dt$ then the equation becomes the expression for the residual of the forward Euler scheme. So in its simplest form, the physics loss is the residual of a first-order integration step. However, in this work, we set $s \geq 2$ so the approximation is at least second-order.

3. Experiments and Results

The TCN, physics-constrained TCN, and the ARX model were used to perform dynamic forecasting of the structural response. In dynamic forecasting, it is assumed that only the initial conditions are known, and the signal trajectory is predicted recursively by using the output at each time step as the input for the next one. This approach is consistent with how these techniques would be used in a real application, where future responses are not known a priori. However, this approach is also prone to significant error accumulation as discussed in [8].

We follow the general architecture of the TCN outlined in [1] and use grid search on the number of channels and layers to find a good set of hyperparameters. It was found that 6 layers and 6 channels per layer resulted in the lowest validation error during the hyperparameter grid search. The kernel size of convolution layers was set to 3 based on manual tuning and the intuition that this kernel size would be appropriate for computing something analogous to the time derivative of the signal. Following Bai *et al.* [1], we use an exponential dilation $d = 2^i$ for each layer i in the network. We trained for 15,000 epochs with standard Adam Optimizer and a learning rate of 0.0001. We used a sliding window of length 12 to iterate through the data and construct X, Y pairs. The sliding window size was determined based on experiments with the ARX model, it was found that a delay of 12 time steps resulted in the most stable ARX model. An illustration of a single sliding window is shown in Figure 2.

The training data was generated by simulating the response of a flat beam with geometric nonlinearities subjected to random vibration excitation. This problem is a common benchmark problem in the field of nonlinear dynamics [10]. The simulated data was obtained by solving

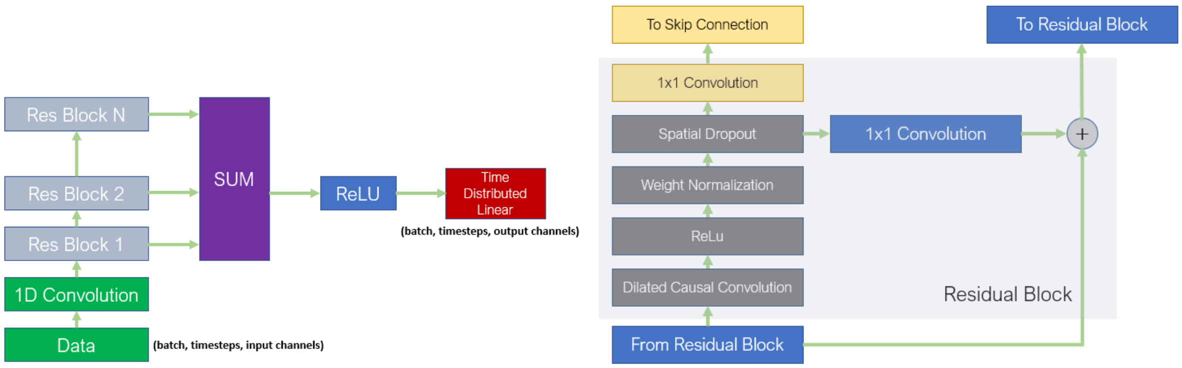

Figure 1. The overall architecture of the TCN (left) and the residual block (right).

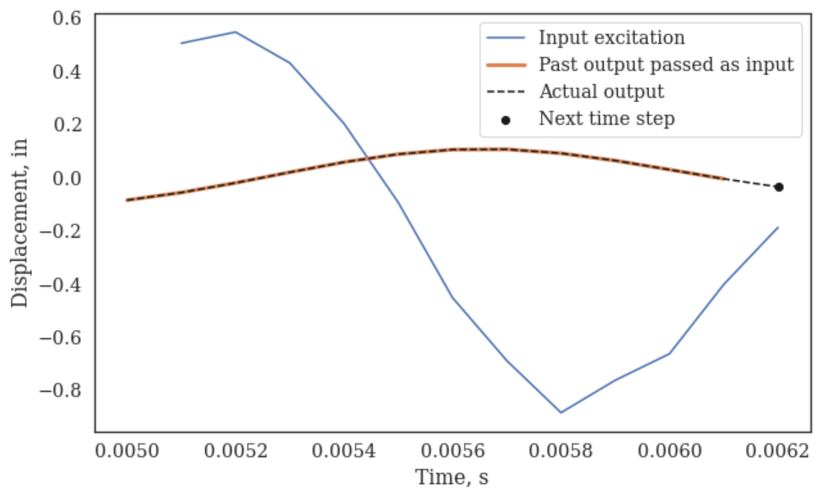

Figure 2. Example of a single sliding window. The current input (blue) and past output (orange) are passed to the TCN to predict the next time step (black).

the nonlinear problem using OSFern, an open-source finite element package. For this paper, we focus on the response at the center of the beam, and ignore the response at other locations. The generated data is shown in Figure 3, where it is shown that 80% of the data was used for training, 10% for validation, and 10% for testing.

The predicted response for all three techniques is shown in Figure 4. The mean squared error (MSE) and the Pearson correlation coefficient (CC) are reported in Table 1. For the Pearson CC, a value of 1.0 would indicate a perfect prediction. As is evident in the error metrics reported, both versions of the TCN are superior to the linear ARX model predictions. Furthermore, a visual inspection of Figure 4 shows that the TCN predictions capture some of the more intricate features of the time signal, such as abrupt changes in amplitude and frequency. The quantitative metrics fail to capture some of these intricacies, but the bottom image in 4 shows how the physics-constrained TCN is able to follow the actual response closely even in the presence of accumulated error.

Another way of looking at the response is to look at the frequency domain response. Figure 5 shows a spectrogram of the response, which is an expression of the energy content of the signal as a function of time and frequency. A notable feature of this image is how the ARX response is fairly narrow-banded, whereas the TCN predictions are able to capture time and frequency variations with higher resolution. The squared error is plotted in Figure 6, also showing how error bands form in the ARX prediction. The MSE values in the frequency and time domain are $2.70e-5$, $2.56e-5$, and $3.00e-5$, for the baseline TCN, physics-constrained TCN, and ARX, respectively.

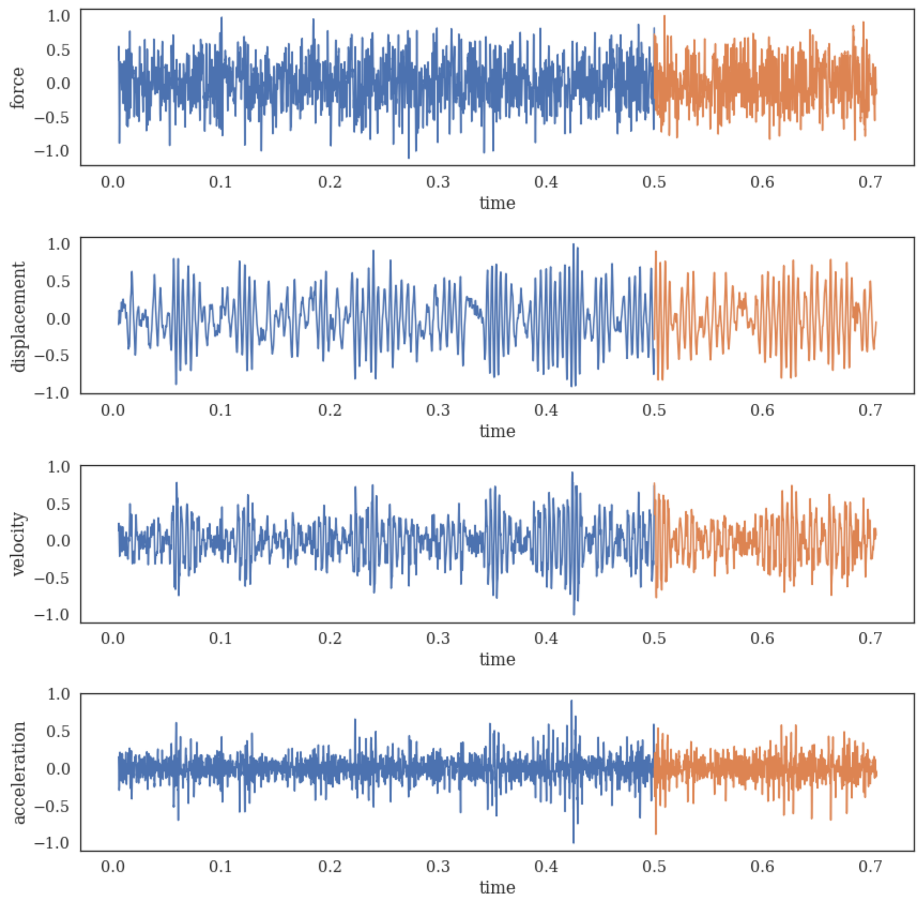

Figure 3. Synthetic structural response data generated from finite element simulation. Training portion shown in blue and test portion shown in orange.

Table 1. Performance metrics used to compare the three techniques.

	TCN Baseline	TCN Physics	ARX
MSE	0.080	0.080	0.096
Pearson CC	0.640	0.630	0.510

It is noted that the authors also compared the performance of the TCN to other popular techniques, namely the LSTM and Transformers. Neither of these techniques were successful at improving upon the TCN. The LSTM best validation error was 630 times larger than the TCN error. Similarly, the best Transformer validation error was 600 times larger. Furthermore, the LSTM training took on the order of several days while the TCN training only takes a few hours. For this reason, these two methods were not pursued further.

4. Discussion of Results

The results obtained show a definite benefit of using the TCN model over the standard ARX model. Both versions of the TCN implemented resulted in superior performance when compared to ARX. The quantitative metrics showed that both version of the TCN had similar performance. However, there are qualitative differences between these two versions, as shown by the distinct trajectories in Figure 4. In particular, the bottom image shows how the physics constraints permit the TCN to follow the trajectory closer, while the baseline TCN diverges from the response when the response amplitude is lower. These low amplitude events are likely due to complex interactions in the modes of vibration of the structure. The implication is that these events are not directly driven by the input but by complex interactions of the structural response. In this case, the physics-based TCN appears to be more capable of capturing these interactions. Another interesting feature of the TCN predictions is how the energy is spread over

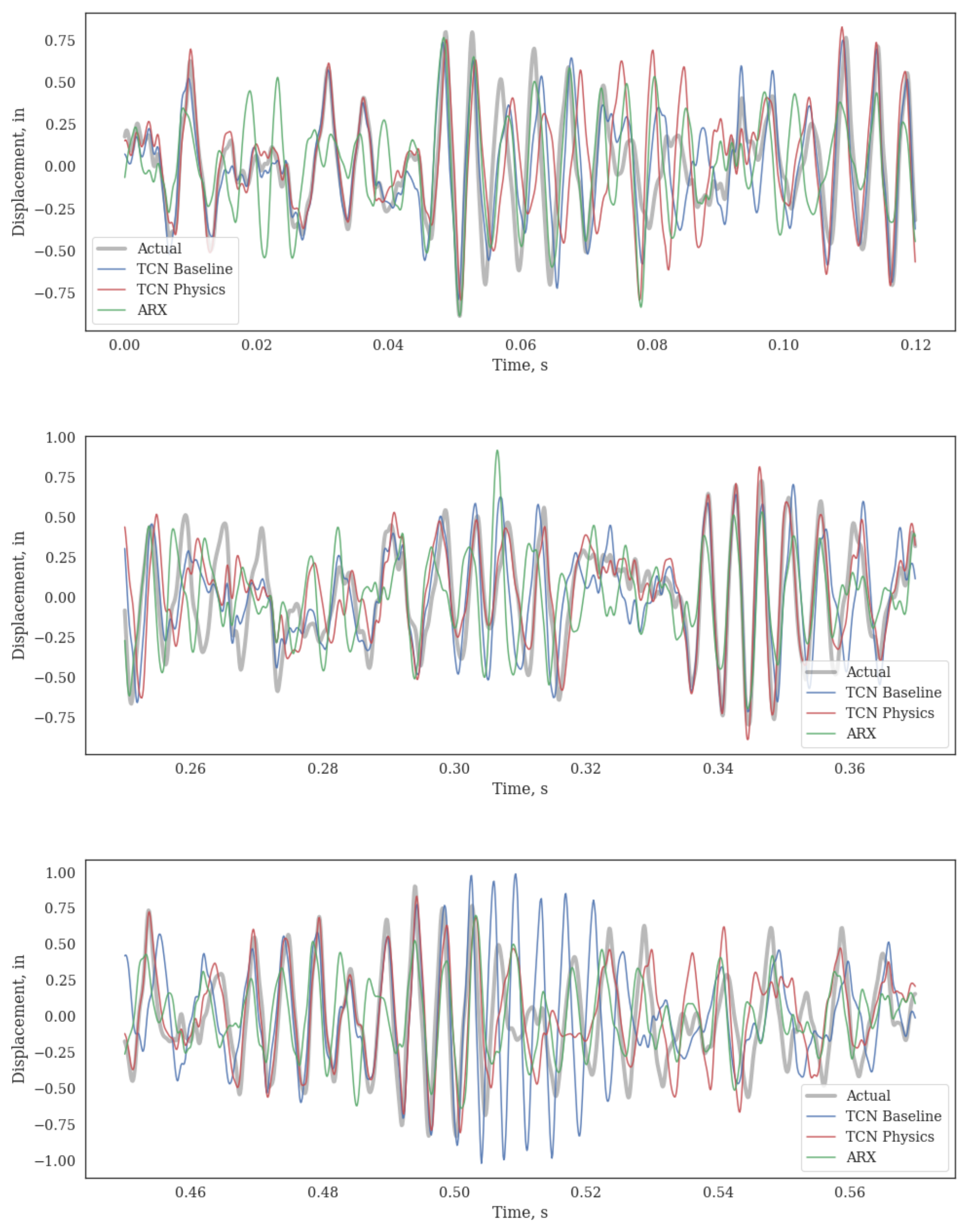

Figure 4. Response prediction comparison between all three techniques towards the start of the simulation (top), the middle (middle), and the end (bottom).

wider frequency bands when compared to ARX. In particular, the ARX model struggles to capture higher frequency content. All methods deal with the lower frequency content well, suggesting that the size of the sliding window was not a limitation in this regard.

A deficiency of the proposed study that we must acknowledge is the fact that we studied the response of a single location in the structure. In reality, the response at different points in the structure are coupled to each other. Future work could focus on studying the spatial correlations in the response that were left out of this study. Another opportunity for future research is the formulation of data-driven models with uncertainty. This consideration is important when the structural system undergoes state changes or when the training data is noisy.

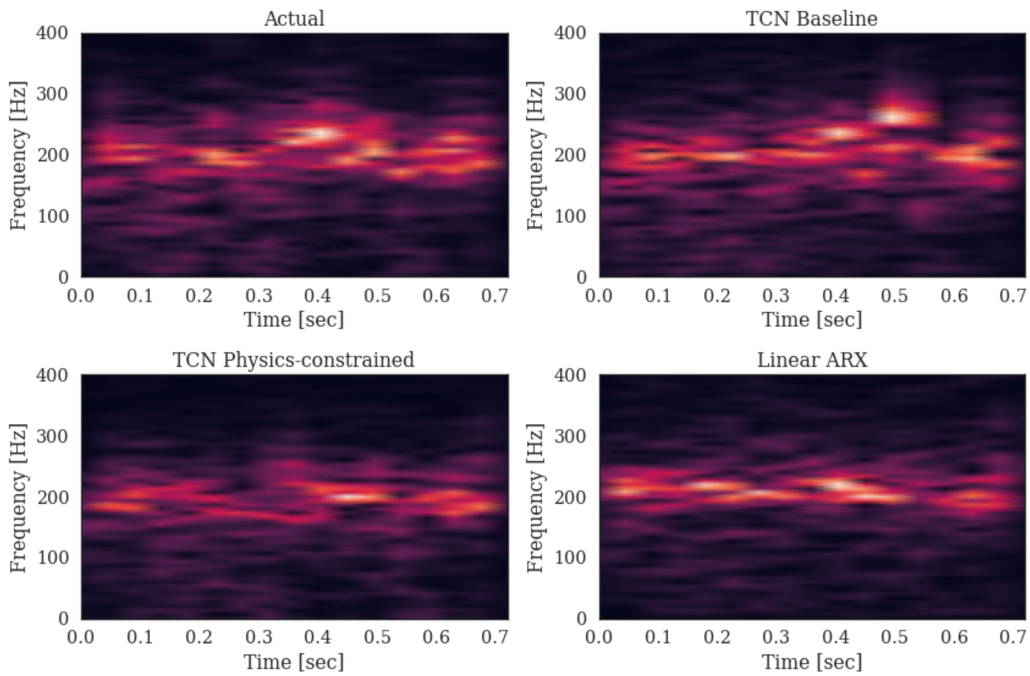

Figure 5. Response prediction comparison in the frequency and time domain between all three techniques.

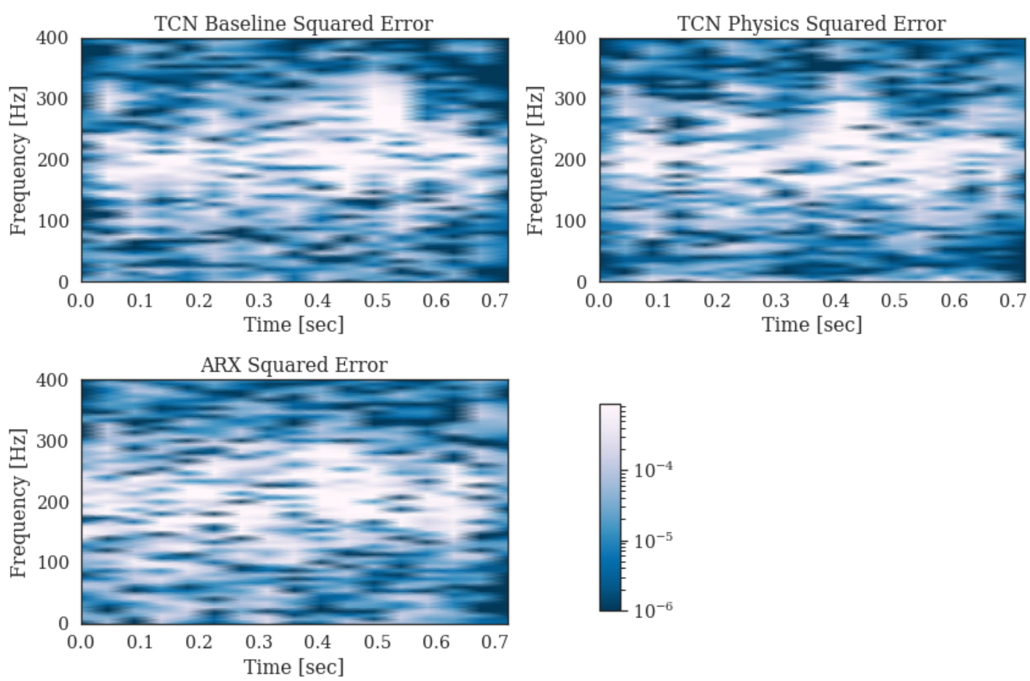

Figure 6. Squared error comparison in the frequency and time domain between all three techniques.

5. Conclusion

We presented an exploration of the TCN, a neural network architecture previously used for speech processing, applied to a nonlinear structural dynamics problem to predict structural response. It was shown that the TCN approach provides significant benefits when it comes to modeling nonlinear structures compared to the more traditional ARX approach, evidenced by smaller errors and higher correlations coefficients, in both the time and frequency domain. We also showed how a physics-constraint added to the loss term can have benefits in the structural response prediction, as evidenced by visual inspection of the time signal predictions. The error level attained with both versions of the TCN were really similar, suggesting that, on average, they can attain similar performance. This work is useful as a foundation for data-driven modeling of nonlinear structures, potentially accelerating the assessment of the structural response of a deployed structural component by providing real-time forecasting of its performance, enabling structural health monitoring. Future work could focus on enhancing the network to provide probabilistic predictions via Bayesian layers, which would be useful if the training data is noisy (e.g., from noisy sensors).

References

[1] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling, 2018.

[2] J. C. Butcher. Implicit runge-kutta processes. *Mathematics of Computation*, 18(85):50–64, 1964.

[3] S. Chen and S. A. Billings. Representations of non-linear systems: the narmax model. *International Journal of Control*, 49(3):1013–1032, 1989.

[4] Zhiyong He, Zanbo Wang, Wei Wei, Shanshan Feng, Xianling Mao, and Sheng Jiang. A survey on recent advances in sequence labeling from deep learning models, 2020.

[5] D. I Ketcheson. Nodepy software version 1.0.1. `http://github.com/ketch/nodepy/`, 2020.

[6] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[7] Lennart Ljung. *System Identification (2nd Ed.): Theory for the User*. Prentice Hall PTR, USA, 1999.

[8] David A. Najera-Flores and Adam R. Brink. Efficient random vibration analysis of nonlinear systems with long short-term memory networks for uncertainty quantification. In *Proceedings of International Conference on Noise and Vibration Engineering and International Conference on Uncertainty in Structural Dynamics*, 2018.

[9] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christopher Bregler. Efficient object localization using convolutional networks, 2015.

[10] C. Van Damme and M. Allen. Nonlinear modal models for sonic fatigue response prediction: A comparison of methods. In *Proceedings of the 38th International Modal Analysis Conference*, 2020.

[11] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio, 2016.

[12] Aäron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. *CoRR*, abs/1601.06759, 2016.