RECCURENT NEURAL NETWORK FOR PREDICTING TIME STEP BISECTIONS IN STRUCTURAL DYNAMICS

Tudor George ALEXANDRU[1], Cristina PUPĂZĂ[2]

The automatic time stepping method is widespread throughout commercial transient dynamics solvers, being a convenient way of adjusting the load increment based on the system state. As part of this algorithm, bisections can be considered an indicator of poor model convergence, demanding most of the times improvement loops. This paper proposes a new methodology for predicting bisection occurrence with the support of Recurrent Neural Networks. A Long-Short Term memory implementation is proposed, training and validation being carried out based on sequential solver output data. The accuracy of the model is proved by means of a highly non-linear structural dynamics simulation.

Keywords: Transient dynamics, Convergence, Bisection, LSTM

1. Introduction

During the past decades, Computer Aided Engineering (CAE) software has emerged as an integrated virtual prototyping approach that allows engineers to verify and optimize design scenarios by recreating their real world behaviour with the support of digital environments [1]. In structural mechanics, the Finite Element Method (FEM) represents one of the most widespread numerical analysis procedures that addresses interdisciplinary problems by dividing a continuous geometric domain into a discrete one [2]. To be more attractive to the market, companies focus more on product innovation by supporting the traditional design processes with emerging disciplines (i.e. nonconventional manufacturing or ergonomics) [3]. Thus, engineering projects rely on the ability of CAE tools to capture the behaviour of non-linear materials or assembly constraints that are subjected to time-varying loads (i.e. 3D printed kinematic structures) [4]. From this perspective, FEM based structural dynamics simulations are the most popular choice for addressing such aspects.

The ongoing improvement of CAE software resulted in lower error troubleshooting demands, less engineering knowledge being required for reaching accurate simulation results [5]. The automatic time stepping method is

[1] PhD. Student, Dept. of Robots and Manufacturing Systems, University POLITEHNICA of Bucharest, Romania, e-mail: alexandru_tudor_imst@yahoo.com
[2] Professor PhD. Eng.., Dept. of Robots and Manufacturing Systems, University POLITEHNICA of Bucharest, Romania, e-mail: cristinapupaza@yahoo.co.uk

implemented in most recent transient dynamics solvers for lowering the model complexity in non-linear problems. This is achieved by estimating the next time step size based on the analysis conditions. An advantage of this approach is its bisection component that has the purpose of adjusting the solver settings based on the convergence residuals state [6]. Even so, the occurrence of bisections also indicates modelling or simulation issues, requiring a careful inspection of the possible error sources. Furthermore, the high computational demands involved in structural dynamics significantly limit the flexibility of engineers to experiment different scenarios.

From this point of view, the present paper proposes a Machine Learning procedure for forecasting convergence errors in transient simulations that deploy the automatic time stepping method. The approach relies on sequential data that is extracted from force, moment and line search convergence criteria that are written at each sub step during the computational process. A Recurrent Neural Network (RNN) is developed using the Long-Short Term Memory (LSTM) implementation. While the problem fits the description of a deep learning classifier, a regression model is deployed instead. The predicted continuous value corresponds to the probability of a bisection to take place at a future time step based on the convergence of the previous ones.

A wide range of structural dynamics simulations that involve the use of automatic time stepping method are discussed throughout the literature. The work disclosed in [7] proposes a simulation methodology that deploys the automatic time stepping method for capturing the transient behaviour of small scale grinding machines. In this case, the step size is calculated based on the natural frequencies extracted from a modal analysis. An approach involving the definition of non-linear contacts is presented in [8] for capturing the tooth interaction behaviour of bevel gears. In this case, due to the complexity of the model, the solution is carried out by using the bisection component in automatic time stepping. Recent guidelines and checklists are published for dealing with such convergence issues [9], each solver being characterized by its own peculiarities. RNNs represent a generalization of feedforward neural networks that have internal memory. The LSTM RNN implementation is widespread in systems that learn and improve from sequential data for solving problems such as image captioning or weather forecasting [10]. On the other hand, different types of Machine Learning algorithms are deployed for supporting optimization processes based on results of CAE simulations [11].

The original contribution of this paper consists in the use of LSTM to support the solving process in structural dynamics simulations, by predicting the probability of a time step bisection to occur based on sequences of solver output data. In this way, a substantial amount of time can be saved in such engineering projects, especially when dealing with large and/or non-linear models. The paper

is divided in four sections. The first part of the work describes the theoretical aspects of transient dynamics simulation with emphasize on the engineering tasks carried out for preparing the model and debugging its convergence errors. The second part of the work illustrates the proposed approach by focusing on the inputs and outputs, along with the LSTM architecture. A case study is presented in the third section, comprising a structural dynamics analysis of a ball bearing assembly using ANSYS software. Due to inappropriate analysis settings and the deliberate suppression of essential geometric elements, several time step bisections occur in the model causing the convergence failure. Finally, conclusions are derived in the fourth section.

2. Theoretical considerations of transient dynamics in CAE

CAE software brings together all necessary tools for preparing, solving and processing the results of transient dynamics simulations based on implicit or explicit integration methods. Examples of engineering problems that can be addressed by such procedures are: the analysis of structures under free and forced vibrations, the simulation of assemblies by considering inertial effects for both rigid and flexible bodies, as well as the approximation of crash behaviour during high velocity impact.

In implicit structural dynamics solvers, the equation of motion can be generalized for multiple degrees of freedom systems:

$$[M]\{\ddot{u}\}+[C]\{\dot{u}\}+[K]\{u\}=\{F(t)\} \tag{1}$$

Where: $[M]\{\ddot{u}\}$ represents the inertial forces, $[C]\{\dot{u}\}$ frictional forces that are proportional to the velocity, $[K]\{u\}$ the elastic forces, $\{F(t)\}$ time varying external forces, $[M]$ the mass matrix or the matrix that describes the inertia of the entire structure and $[C]$ is the damping matrix. The matrices [C], [M] and [K] are assembled from the element matrices [12].

For any given time *t*, the equations derived from (1) can be considered a set of static equilibrium ones that also take into account inertia and damping effects. Various time integration methods (i.e. the Newmark or the improved HHT ones) can be employed for solving them at discrete time points. The time increment between successive iterations is called the integration time step.

The automatic time stepping method is implemented in most recent transient dynamics, electromagnetics or thermal solvers for deciding the optimal increment of time and/or loads in response to the current state of the analysis. The algorithm involves two major components: time step prediction and time step bisection.

Considering a converged solution at time t_n the step size for the next t_{n+1} is determined based on the minimization statement [13]:

$$\Delta t_{n+1} = \min\left(\Delta t_{eq}, \Delta t_1, \Delta t_2, \Delta t_g, \Delta t_c, \Delta t_{p,} \Delta t_m\right) \tag{2}$$

Where: Δt_{eq} is the time increment needed for the previous step to converge (which is limited by the maximum number of equilibrium iterations), Δt_1 and Δt_2 represent the time increments that are influenced by 1st order systems (i.e. transient thermal analysis) as well as 2nd order ones (i.e. transient structural analysis), Δt_g is the time increment that takes into account abrupt changes in the contact status while Δt_c and Δt_p refer to time increments that depend on the allowable creep and plastic strain. On the other hand, Δt_n represents the time increment that is limited by the midstep residual interpolation tolerance defined in the analysis settings.

Time step bisections occur when the number of equilibrium iterations used for one substep exceeds the allowable limit or when all equilibrium iterations are used. In such cases, the current substep solution for Δt_n is removed and the step size is reduced by half.

$$\Delta t_n = \frac{\Delta t_n}{2} \tag{3}$$

Structural dynamics analysis deploys an iterative solving process that is based on the Newton-Raphson method. Theoretically, the convergence in such simulations is achieved when the out-of-balance load vector is equal to zero. However, the existence of nonlinearities in such problems demands an out-of-balance threshold value to be defined.

A holistic overview of the stages required to troubleshoot a structural dynamics analysis are depicted in Figure 1 and described in the paragraph below.

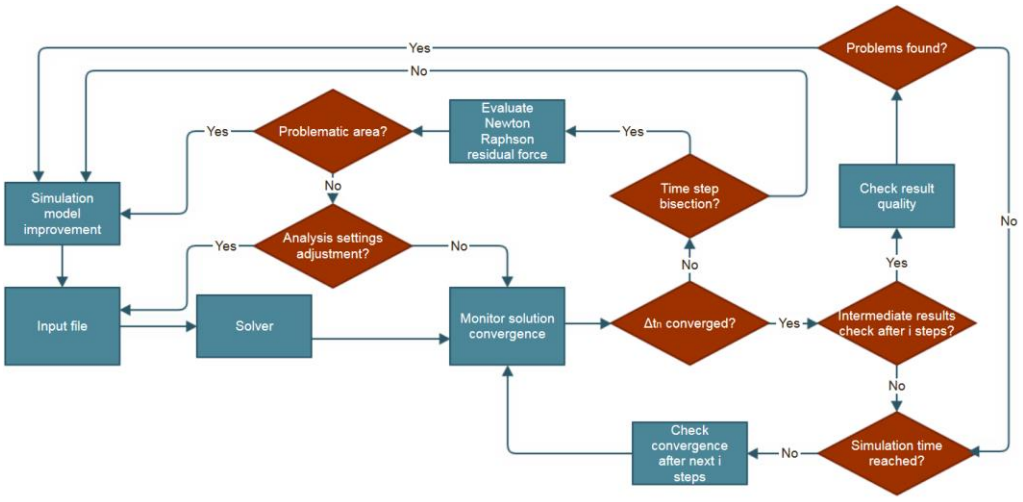

Fig. 1. A holistic overview of the structural dynamics simulation project

The input file represents a standardized data structure that transposes the digital model defined using CAE Pre-processors in a list of commands that are further interpreted by the solver. This allows the equilibrium equations to be assembled and solved for each substep. Output files are generated during this iterative procedure, providing valuable insights regarding the state of the simulation. Based on force, moment, displacement and/or line search criteria, the convergence at each Δt_n is evaluated. Figure 2 depicts the force convergence graph derived from a structural dynamics analysis of a spindle subjected to a constant rotational velocity that was previously completed using ANSYS Workbench Transient Structural module.

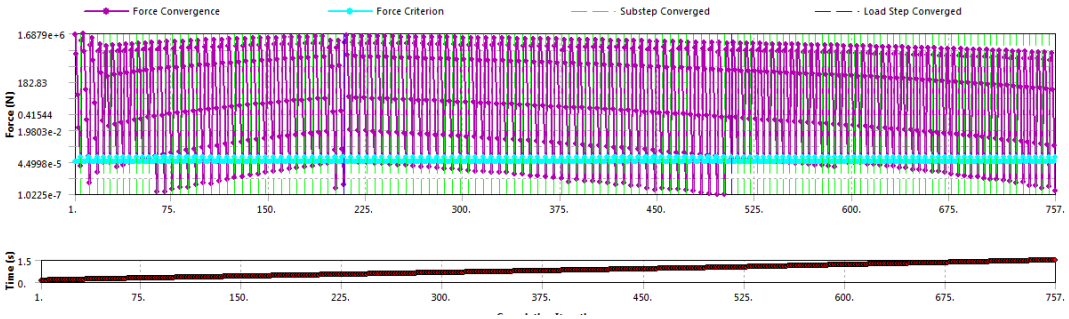

Fig. 2. Force convergence in a structural dynamics simulation

In this case, the calculated reaction forces match the applied loads for each substep and subsequent load steps. Furthermore, the difference between the internal force and the applied one remains at all-steps below the convergence criteria proving that no unbalances occur.

As a good practice, intermediate results are checked after each i substeps to identify any modelling problems, such as stress concentrators or unrealistic behaviour due to inertial effects. In case of convergence errors, the occurrence of time step bisections is verified. Figure 3 represents another attempt of solving the same rotating spindle example by using excessively large time step settings.

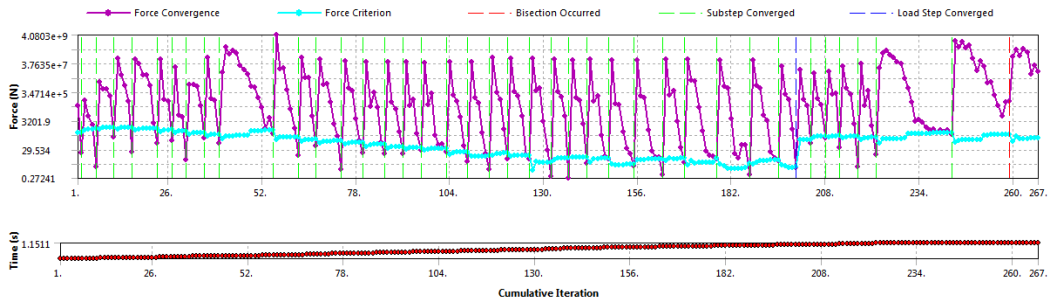

Fig. 3. Time step bisection due to inappropriate solver settings

The analysis converges for the first load step but bisections occur in the second one. In such cases, the Newton-Raphson residual forces are assessed to indicate any problematic areas in the model. Subsequently, the shape of the convergence curves is analysed in the proximity of the bisection. When the residuals represent a steep increase from the previous Δt_{n-1} substep, the automatic time step settings require adjustment. In other scenarios, bisections can occur without a clear indicator of model or solver setting issues. Therefore, the convergence of the model is verified for the next Δt_{n+1} substep. The workflow is completed when the simulation time is reached.

The workflow for solving structural dynamics convergence represents a limiting aspect of the simulation environments considering the iterative solving process deployed and the coexistence of several error factors. Furthermore, the high computational demands of non-linear models limit the amount of time available for performing and validating adjustment loops. From this perspective, an approach that can predict convergence errors based on the information that is available during the solving process can significantly lower the complexity of such workflows.

3. Implementing RNNs for supporting convergence troubleshooting

RNNs represent a generalization of feed forward Deep Learning that remembers previous inputs in an internal memory. Most recent Machine Learning libraries include improved versions of RNNs (i.e. Gated recurrent unit or LSTM) that solve the well-known exploding or vanishing gradients problems. This allows regression or classification models to be developed based on sequential data by using high level programming interfaces [14].

In the proposed approach, a LSTM implementation is deployed to predict the probability of a time step bisection to occur based on the convergence state of the previous substeps. For this purpose, the raw data that is generated by structural dynamics solvers is converted into a dataset that is further used for training and validating the neural model. The procedure is divided in three main stages: conversion of raw solver output data; encoding the LSTM inputs and model training and validation (see Fig. 4).

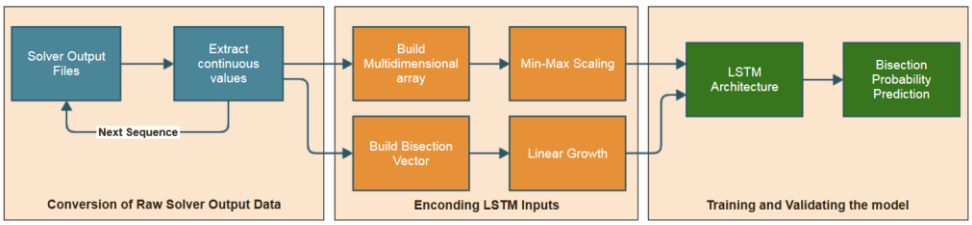

Fig. 4. A schematic representation of the proposed approach

In the first stage, the output files that provide relevant convergence information are chosen for extracting continuous features in a dataset. Each structural dynamics solver has its own standard for writing such logs (i.e. ANSI encoding text format). From this perspective, a line by line read procedure can be deployed in accordance with the occurrence of certain strings (i.e. "Solution not converged"). Figure 5 represents an excerpt of the worksheet output of the solver. The highlighted values consist of the max degree of freedom increment (Max. DOF Incr.), line search parameter (Line Search Param) and force and displacement convergence criteria (F CRIT, F L2, U CRIT, U INF) derived during the extraction process.

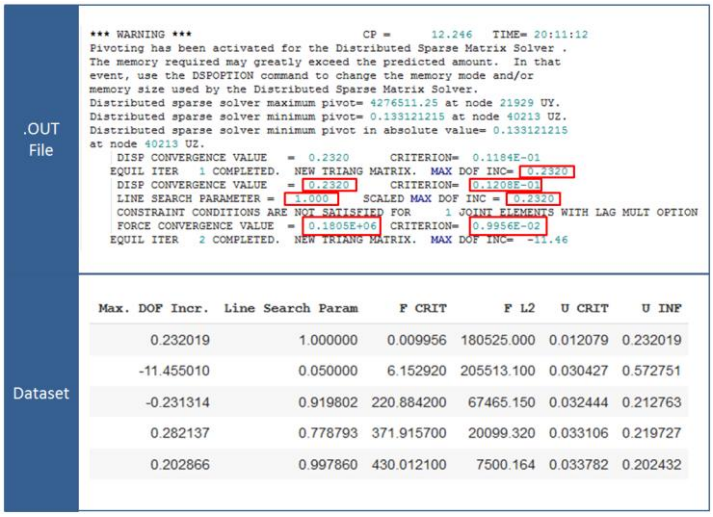

Fig. 5. Conversion of raw solver output data

The resulting labelled data is converted into a multidimensional array, the number of lines and columns corresponding to the 2D shape of the dataset while the 3rd dimension is defined based on a training constant that describes the size of each sequence that is passed to the model. This parameter depends on the number of substeps and provides the previous state of the simulation for which the future bisection probability is predicted. Feature normalization (denoted x' in equation 4) is applied for each entry (x) in the resulting array by using the formula:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{4}$$

In the next stage, the time steps for which bisections occur are identified from the convergence summary output files. A vector is defined, having a shape that is equal to the number of iterations performed in the simulation. Each row in

this vector is filled with zeros when the corresponding iteration achieves convergence or with non-zero values in case of time step bisections.

To this end, training and validation of the neural model can be accomplished by using the features defined in the multidimensional array and the labels derived in this stage. From this perspective, a LSTM classifier can be developed to predict the probability of an iteration to belong to the zero (convergence) or non-zero (bisection) classes based on previous inputs. Even so, in most structural dynamics simulations the existence of time step bisections accounts only for a small fraction of labels than the converged substeps.

In this regard, the ability of the model to classify bisection occurrences is limited, considering the lack of consistent training data. To overcome this issue, a LSTM regression model is developed instead. For this purpose, the labels are converted from categorical to numerical ones, having values ranging between zero and one. This allows the occurrence probability of a time step bisection to be predicted as a continuous value. Therefore, a linear growth algorithm is depicted in Table 1 using as input the bisection vector (vector) and its 1D shape (shape).

Table 1

Melting points and elemental analyses

Algorithm for Linear Growth

```
procedure linear_growth (vector, shape)
    read vector
    row_start = 0
    row_end = 0
    counter = 1
    while row_start <= shape do
        while vector (row_start) = 0
            increment row_start, counter
        end while
        int_quantum = 1 / counter
        quantum = int_quantum
        for i =1 to counter do
            vector2 (row_end) = quantum
            increment row_end
            quantum = quantum * (i+1)
        end for
        increment row_start
        counter = 1
    end while
end procedure
```

The procedure reads each row of the vector and identifies the position of zero values. Two counters are incremented when this condition is satisfied (row_start and counter). If the value of the row is different than zero, a variable (int_quantum) is calculated as the multiplicative inverse of that row number. A second vector (vector2) is filled (row_end) by linearly growing the int_quantum variable (quantum) between subsequent loops of i and counter. The procedure ends when the shapes of the two vectors are equal.

Having both features and labels defined in accordance with the regression problem, a LSTM architecture can be developed by using in the first stage a fraction of the dataset for training the RNN and another one for validating its predictions (see Fig. 6)

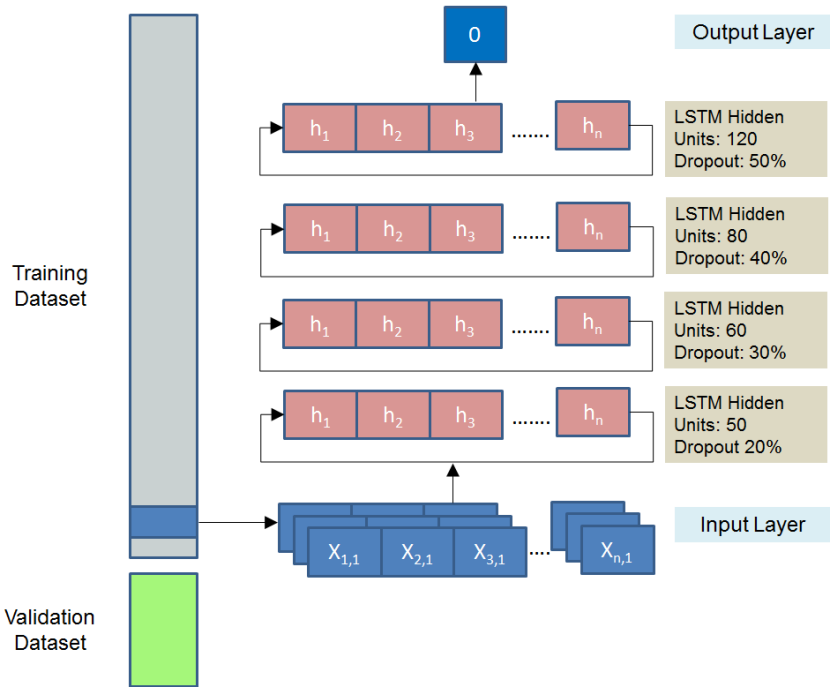

Fig. 6. Graphical representation of the LSTM architecture

In the proposed model, the input layer consists of a multidimensional array comprising the normalized solver output data. A stack of four LSTM layers is added to capture the abstract concepts in the sequences. The Rectified Linear Unit activation function is used between their outputs. Each hidden layer comprises hidden cells that are characterized by multiple hidden units. On the other hand, each hidden unit addresses the problem of long-term dependencies by adjusting the flow of information with the support of input, output and forget gates. To improve the generalization ability of the model, the dropout regularization method is deployed to randomly exclude LSTM units during each training step. The

accuracy of the model depends on the sequential data that is fed into it. Even so, the choice of hidden units and drop out rates that achieve best settings can be summarized as: 50, 60, 80 and 120 hidden units having dropout rates of 20% to 50%. In the output layer, a continuous value is predicted that corresponds to the bisection occurrence probability. The Adam optimizer is deployed to perform an iterative update of the network's weights based on stochastic gradient descent method. Typical to a regression problem, the Mean Squared Error is used as convergence metric.

4. Verification of the given concepts

A structural dynamics analysis is completed using the Transient Structural module from the ANSYS Workbench interface to verify the given concepts. The aim of this simulation is to emphasize the computational aspects that lead to convergence issues.

The geometry in discussion consists of a rigid spindle that is supported by a deep groove ball bearing. Kinematic joints are defined to materialize the motion of the bodies, non-linear frictional contacts being included to capture the interaction between the rolling elements and the inner and outer rings. Only the rotation of the spindle around its axis is considered in the simulation. Topology operations are conducted to ensure that a Hex-Dominant mesh can be generated (see Fig.7).

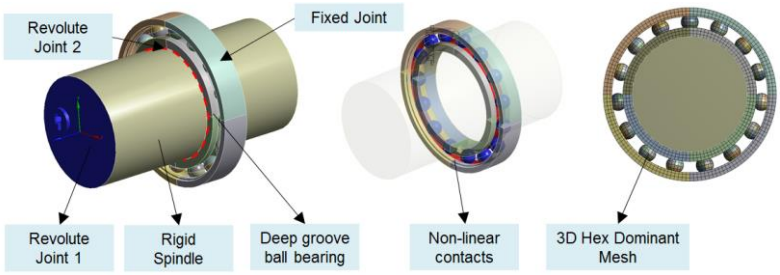

Fig. 7. Overview of the simulation model

Excessively large automatic time step settings are adopted (initial time step: 0.1 seconds; minimum time step: 0.0001 seconds and maximum time step 0.5 seconds) to enforce convergence issues. Furthermore, the cage of the rolling elements was removed from the geometry, causing an unconstrained motion of the balls that rub against each other.

During the solving process, several time step bisections occur due to the unbalance of energies in the system. A total of 895 iterations are completed prior to aborting the solution. Fig. 8 depicts the force convergence residuals, emphasizing the location of time step bisections.

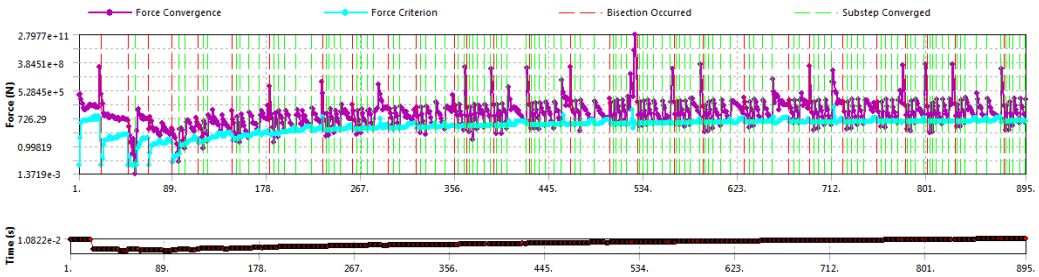

Fig. 8. Force convergence residuals and the location of time step bisections

The dataset is assembled as a 3D array having 895 rows (corresponding to the number of substeps), 6 columns (corresponding to the maximum degree of freedom Increment, line search parameter, force convergence, force criteria as well as displacement convergence and displacement criteria) and sets of 7 sequences. A selection of 700 entries is employed for training the neural model while the remaining samples are used for validating its predictions.

TensorFlow 2.0 with Keras Machine Learning library is used for defining the LSTM architecture A total number of 10 epochs is chosen as training parameter, meaning that the dataset is backward and forward passed for 10 times. During each epoch, the mean square error loss function is monitored to evaluate the prediction accuracy of the model. At epoch 1, the loss function has a value of 0.1913 while at epoch 10 the value is minimized to 0.0514.

The ability of the model to predict time step bisections is determined by using the features from the validation set. For this purpose, the probability peaks that exceed 50% threshold between successive sequences are considered bisection locations. A maximum error of 1.27% is achieved, the accuracy being around +/- 11 substeps. Fig. 9 depicts the predicted vs. real bisection occurrences.

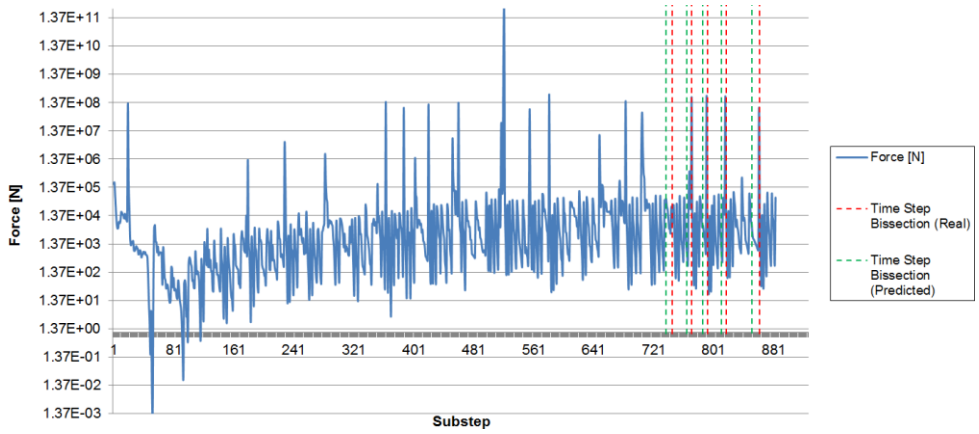

Fig. 9. The occurrence of time step bisections in real vs. predicted cases

6. Conclusions

The present paper addresses the limiting aspects of troubleshooting convergence issues in structural dynamics analysis. A RNN approach is proposed for predicting time step bisections by using the LSTM implementation. Training and testing of the neural model is accomplished with the support of solver output data. Methods for encoding the LSTM inputs are depicted throughout the work. A general deep learning architecture is proposed together with best hyperparameters choices. The results achieved prove the ability of the approach to be implemented in large scale structural dynamics projects.

R E F E R E N C E S

[1]. *S. F. Qin and K. Cheng*," Future digital design and manufacturing: embracing industry 4.0 and beyond" in Chinese Journal of Mechanical Engineering, **vol. 30**, no. 5, 2017, pp. 1047-1049

[2]. *I.G.Ghionea*," Optimization approach to conception of a mechanical part using CAD/FEM techniques", in UPB Scientific Bulletin, Series D., **vol. 71**, no. 4, 2009, pp. 43-52

[3]. *A. Hatchuel, P. Le Masson, Y. Reich and E.* Subrahmanian "Design theory: a foundation of a new paradigm for design science and engineering", in Research in Engineering Design., **vol. 29**, no. 1, 2018, pp. 5-21

[4]. *Q. Wang, G.Senatore, K.Jansen, A.Habraken and P.Teuffel*," Design and characterization of variable stiffness structural joints", in Materials & Design, **vol. 187**, 2020, pp. 1-17

[5]. *C. Mullen,*" A Review of Finite Element Analysis with respect to Experimental Results", in Proceedings of the 16th LACCEI International Multi-Conference for Engineering, Education, and Technology., **vol. 1**, no. 1, July. 2018, pp. 1-3

[6]. *D. F.Rossi, Mauthe, W.G. Ferreira, W.J.Mansur and A.F.G.Calenzani*,"A review of automatic time-stepping strategies on numerical time integration for structural dynamics analysis", in Engineering structures, **vol. 80**, 2014, pp. 118-136

[7]. *J. Byiringiro*," Dynamic analysis of structure of a small-scale ball mill using ANSYS and EDEM", in Journal of Sustainable Research in Engineering., **vol. 4**, no. 3, 2018, pp. 99-110

[8]. *H. Shengy and, Y.Xiao,* "Tooth Surface Contact Analysis for Logarithmic Spiral Bevel Gear based on ANSYS", in International Journal of Civil Engineering and Machinery Manufacture, **vol. 4**, no. 3, 2019, pp. 27-34

[9]. *R.J. Boulbes,* Troubleshooting Finite-Element Modeling with Abaqus: With Application in Structural Engineering Analysis, Springer Nature, Cham, 2019.

[10]. *A. Sherstinsky,* "Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network", in Physica D: Nonlinear Phenomena., **vol. 404**, no. 3, Sept. 1996, pp.1-28

[11]. *S. M.Yuan, D.Hutchison, G.Coulson and S.Namuye*," Optimum Design of Machine Tool Structures Based On BP Neural Network and Genetic Algorithm", in Advanced Materials Research, **vol. 655-657**, 2013, pp. 1291-1295

[12]. *E. Madenci, I. Guyen,* The finite element method and applications in engineering using ANSYS®, Springer, New York, 2015.

[13]. *** ANSYS – Finite Element Analysis, Release 19.0 User Guide, 2018.

[14]. *J. Brownlee,* Long Short-term Memory Networks with Python: Develop Sequence Prediction Models with Deep Learning, Machine Learning Mastery, 2017.