REGULARIZATION OF A PROGRAMMED RECURRENT ARTIFICIAL NEURAL NETWORK

Andrew J. Meade, Jr.
Department of Mechanical Engineering
and Materials Science
Rice University
Houston, Texas, 77251-1892, USA
Phone: (713) 348-5880
E-mail: meade@rice.edu

DRAFT

To be submitted to Journal of Guidance, Controls, and Dynamics.

This work is supported under ONR grant N00014-95-1-0741.

REGULARIZATION OF A PROGRAMMED RECURRENT ARTIFICIAL NEURAL NETWORK

Andrew J. Meade, Jr.
Department of Mechanical Engineering
and Materials Science,
Rice University
Houston, TX 77251-1892, USA

Abstract

A method is developed for manually constructing recurrent artificial neural networks to model the fusion of experimental data and mathematical models of physical systems. The construction requires the use of Generalized Tikhonov Regularization (GTR) and imposing certain constraints on the values of the input, bias, and output weights. The attribution of certain roles to each of these parameters allows for mapping a polynomial approximation into an artificial neural network architecture. GTR provides a rational means of combining theoretical models, computational data, and experimental measurements into a global representation of a domain. Attention is focused on a second-order nonlinear ordinary differential equation, which governs the classic Duffing's oscillator. The nonlinear ordinary differential equation is modelled by the recurrent artificial neural network architecture in conjunction with the popular hyperbolic tangent transfer function. GTR is then used to smoothly merge the response of the RANN and experimental data. Moreover, this approach is shown to be capable of incorporating other smooth neuron transfer functions, as long as they can be described by a Taylor series expansion. A numerical example is presented illustrating the accuracy and utility of the method.

Key Words: Regularization, recurrent artificial neural networks, neural computation, differential equations, chaos, network training

1 INTRODUCTION

Presently, an area of increasing interest is the emulation and control of nonlinear dynamic systems by the recurrent artificial neural network (RANN) architecture. These particular networks use supervised learning algorithms (training algorithms) such as recurrent

∗ Associate Professor, E-mail: meade@rice.edu
This work is supported under ONR grant N00014-95-1-0741.

backpropagation [1]-[3], which allow the network to simulate a dynamic system without knowledge of the governing equations.

The supervised training of a RANN system can be formulated as the procedure of selecting the optimal network parameters so that the response of the neural network is "close" to the available data, as measured by an error criterion. In terms of the connectionist literature, numerical minimization of this error criterion by the steepest descent method constitutes the backpropagation algorithm [1]. The conjugate gradient method, the Levenberg-Marguardt method, and other nonlinear optimization methods have also been used for network training. However, it has been found that even the most advanced optimization algorithms can show unreliable convergence in training and can be highly dependent on nonlinear parameter constraints and their initial values [4]. In modelling physical systems, the initial values for the neural network parameters are commonly selected by random sampling [5]. If a satisfactory solution is not obtained during training, the nonlinear optimization procedure is restarted with new randomly selected values [6]. Clearly this approach can require prohibitive computational resources for some practical problems.

With the many computational mechanics tools available to the typical engineer, it is doubtful that he or she would resort to data driven modelling by a RANN unless an analytical description of acceptable fidelity were unavailable. Add to this the previously mentioned difficulties in training, and it is apparent why artificial neural networks are not used more widely in engineering. However, unlike soft-computing problems, even the most highly complex physical problem possesses a course analytical description that can be derived from semi-empirical relations, the conservation of mass, momentum and energy or even heuristic considerations. Since it is well known that training algorithms either reduce to, or are, nonlinear optimization techniques, it is reasonable to assume that they will benefit from physically meaningful initial values when modelling a physical system.

In this regard, a new connectionist training initialization method is proposed that can be developed from accurate solutions of the time-dependent differential equations, using only

the equations of interest and the initial conditions. This proposed initialization method is based on three assumptions. Firstly, an artificial neural network system can be treated as a general approximation scheme [7] capable of approximating differential equations. Secondly, polynomial bases, which are a traditional method for approximating functions and analyzing experimental data [8], [9], can be mapped into neural network architectures through constraints on the network parameters. Thirdly, a-priori mathematical models of engineering systems, linear and nonlinear, can be utilized to obtain initial values for the network parameters.

Note that this proposed approach would also allow the initialization of a connectionist training algorithm without data; this feature could be especially useful for on-line RANN modelling systems. Also, the method could be useful in obtaining preliminary bounds on the network performance and estimating the accuracy of network approximations. For example, by exploiting the analogy between ANN systems and polynomial approximations, the effect of augmenting networks with additional processing elements could be evaluated.

The framework that is proposed for data-model fusion in the constructed network is based on Generalized Tikhonov Regularization (GTR) [10]. GTR leads to the formulation of a global response function or hypersurface of the parameter space. Engineers and scientists routinely attempt to fit hypersurfaces through empirical information so as to interpolate or determine a trend (prediction) in the information. In doing this, the users make assumptions about the nature of the underlying functional representation or interpolation function. The method proposed in this paper formalizes these assumptions and extends the user's abilities to cases where the measurements and underlying function may exist in different spaces.

The proposed approach to data-model fusion utilizes a common tool used in artificial neural network (ANN) applications, the original theory of Tikhonov Regularization (TR) of nonlinear inverse ill-posed problems [18]. Simply put, the task of selecting the best functional approximation (a hypersurface) to multivariate empirical data is, fundamentally, an ill-posed problem, i.e., the solution is nonunique, or is sensitive to small errors

in the data. This problem is addressed by the addition of constraints in a-process known as regularization. TR involves the addition of smoothing constraints in a variational form resulting in an optimization problem. This regularization functional is typically designed to penalize undesirable characteristics of the approximating hypersurface, such as highly oscillatory behavior between data points. Unfortunately, popular methods, such as ANNs, use regularizing functionals which are strictly smoothness-based [7] and so are void of any physical information about the process being modelled. As a consequence, these models also frequently have poor or unpredictable data generalization properties. What primarily separates GTR from these interpolation methods and the original TR is that the regularization functionals are based on the physics of the problem and can range from heuristics to mathematical models. Thus, GTR seeks to approximate the response of the underlying physical system rather than just interpolate.

The proposed network initialization and regularization methods are critically dependent on demonstrating that a conventional RANN, using nonlinear transfer functions, can accurately approximate a physical model in the form of differential equations, seamlessly merge the embedded mathematical model and experimental, and that the accuracy of the constructed RANN can be controlled by the user. Consequently, the purpose of this paper is to show how a conventional RANN can be constructed to model a nonlinear time-dependent ordinary differential equation without the need for training or resorting to special architectures and that the connection weights can be modified in a systematic manner to include experimental data. The connection weights of this network are initially determined by reducing the governing equations to systems of both linear and nonlinear algebraic equations and approximating the dependent variables by linear combinations of transfer functions. The connection weights are then merged with empirical data through GTR as a postprocessing step. The algorithms used were relatively straightforward and can be run on conventional hardware. The resulting network is architecturally identical to those that use conventional training techniques, yet it possesses bounded errors that can be controlled by the network parameters in a straight-forward manner.

As a numerical example a RANN using a single input and output layer, with additive neurons utilizing the hyperbolic tangent activation function, has been constructed to approximate the solution to the inhomogeneous Duffing's equation. The Duffing's equation describes the forced oscillations of a particle in a two-well potential [11]. This is considered to be an interesting and practical example since the equation exhibits both chaotic and nonchaotic behavior for specific parameter combinations. The network output, starting with the chaotic regime, is altered through GTR and synthetic data to accurately reconstruct the system response in the nonchaotic regime. Results in the form of phase plane trajectories compare well with the results of previous computational investigations for both chaotic and nonchaotic conditions.

2 APPROACH: RANN Construction

Since the RANN construction method is to ultimately serve as a supervised learning initialization technique, the development of the construction method is given in the framework of supervised training.

2.1 RANN Architecture

A recurrent artificial neural network (RANN) can consist of several layers of processing elements as illustrated in Fig. 1. It is understood that the network output is fed back to the input layer, though the connections are not shown in the figure. The variables $u_i(t)$, $i = 1, ..., M$ denote the input states of the RANN at the time t while the variables $v_j(t+1)$, $j = 1, ..., N$ denote the output states of the network at the time $t + 1$ which in turn are fed back as some of the input variables used to determine the RANN response at the next time period. The first P components of the input state vector $\overline{u}(t)$ are selected as the input variables of the physical system, $P < M$. Similarly, the first L components of the vector $\overline{v}(t + 1)$ specify the response of the corresponding physical system, $L \leq N$. The output at time $t + 1$ of a RANN, consisting of k layers of nonlinear processing elements, can be

determined by the equation

$$v_j(t+1,\overline{\beta}) \;=\; \beta_{k,j}^{N+1}\Upsilon_k\left(\sum_l \beta_{(k-1),l}^{j}\Upsilon_{(k-1)}\left(...\Upsilon_2\left(\sum_i \beta_{1,i}^{p}\Upsilon_1\left(u_i(t)\right)\right)\right)\right), \qquad (1)$$

where $\overline{\beta}$ represents the set of network parameters, and Υ_j is the nonlinear transfer function of neurons associated with the j-th intermediate layer of the network.

Several alternative functions have been used as the processing elements in RANNs, but the more popular training procedures, such as backpropagation, require that the functions be continuously differentiable and bounded. In this regard, sigmoidal neurons are often selected for network applications. The corresponding functions are continuous and are characterized by the following limits

$$\lim_{\xi\to\infty}\Upsilon(\xi)=1 \quad , \quad \lim_{\xi\to-\infty}\Upsilon(\xi)=-1 \quad , \qquad (2)$$

The function

$$\Upsilon(\xi)=\tanh(\xi) \qquad (3)$$

is a paradigmatic example of such sigmoidal functions.

Figure 1 presented the general architecture of a RANN. However, in an actual application the user must specify the number of neuron layers as well as the number of neurons in each layer and their connections. These user specifications may be based on intrinsic considerations. In this study a simple RANN with two layers of nonlinear processing elements is used since this type of RANN is arguably the most common in dynamic network applications [12]. The transfer functions of these layers are taken as the hyperbolic tangent function of Eq.(3). The response of this type of RANN can be written as

$$v_j(t+1) \;=\; \beta_{2,j}^{N+1}\tanh\left(\sum_i \beta_{1,i}^{j}\tanh\left(u_i(t)\right)\right) \quad . \qquad (4)$$

2.2 RANN Parameter Initialization Using Mathematical Models

2.2.1 Linear versus Non-Linear Training

The training of the RANN system is performed by adjusting the linear and nonlinear network parameters in $\overline{\beta}$ by nonlinear optimization procedures. As previously mentioned,

the success in training depends to a great extent on an appropriate selection of the initial values of the network parameters. Note that the network response from Eq.(4) depends linearly only on the parameters $\beta_{2,j}^{N+1}$, other parameters affect the network response in a nonlinear manner. Consequently, initial values of the network parameters that would be adequate for successful network training can be determined efficiently by establishing a set of constraints on the nonlinear parameters. The response of these types of neural networks can then be rewritten as a linear combination of basis functions $\Phi_k\left(\overline{u}(t)\right)$. Specifically,

$$v_j(t+1) \cong \sum_k c_k^j \Phi_k\left(\overline{u}(t)\right) \quad . \tag{5}$$

In this paper global bases will be formed from linear combinations of transfer functions. The linear coefficients (connection weights) associated with the bases will be determined in the process of approximating the dependent variables.

2.2.2 Sigmoidal Networks and Polynomial Bases Approximation

The idea of mapping a conventional approximation scheme, with well established approximation properties, into neural network architecture has been pursued in the literature [13]-[17] for theoretical studies regarding the density and approximation properties of ANNs and the dependence of the network approximation error on the number of neurons. In this paper, the mapping approach is used with a new and practical intent. Specifically, the well known method of polynomial approximation, commonly used in data analysis and computational mechanics applications, will be mapped into a network architecture for the emulation of mechanical systems. Attention is drawn to the following formula

$$\psi_{\wp}\left(\overline{w},\overline{x}\right) \;=\; \frac{\partial^{|\wp|}}{\partial w_1^{p_1}...\partial w_s^{p_s}}\psi\left(\overline{w}^T\overline{x}+\theta\right) = \overline{x}^{\wp}\psi^{(|\wp|)}\left(\overline{w}^T\overline{x}+\theta\right) \quad . \tag{6}$$

where $(\cdot)^T$ denotes a vector transpose, $\overline{x}^{\wp} = {x_1}^{p_1}...\ {x_s}^{p_s}$, $\wp$ is a multi-index such that $|\wp| = p_1 + ... + p_s$, ψ is a nonlinear and non-polynomial function and superscript $(\cdot)$ represents the order of the ordinary derivative of ψ. Equation (6) shows that a polynomial $\overline{x}^{\wp}$ can be readily expressed as

$$\overline{x}^{\wp} \;=\; \left(\psi^{(|\wp|)}(\theta)\right)^{-1}\psi_{\wp}\left(0,\overline{x}\right) \quad . \tag{7}$$

where the bias θ is selected by the user. Therefore, by replacing the partial derivative $\psi_{\wp}(0,\overline{x})$ by an adequate finite difference approximation, one can accurately approximate the polynomial $\overline{x}^{\wp}$ by a finite linear combination of functions in the form $\psi\left(\overline{w}^{T}\overline{x}+\theta\right)$. The following finite difference scheme is given as an example

$$\psi_{\wp}(0,\overline{x}) \;\cong\; \Psi(\overline{x}) = (2\Delta w)^{-|\wp|}\sum_{0\leq\zeta\leq\wp}(-1)^{|\zeta|}\left(\begin{array}{c}\wp\\ \zeta\end{array}\right)\psi\left((2\zeta-\wp)^{T}\Delta w\overline{x}+\theta\right) \quad , \tag{8}$$

where Δw is a small positive constant and the multi-integer binomial is defined as

$$\left(\begin{array}{c}\wp\\ \zeta\end{array}\right) \;=\; \prod_{j=1}^{s}\left(\begin{array}{c}p_j\\ r_j\end{array}\right) \quad . \tag{9}$$

Note that the accuracy of the finite difference scheme of Eq.(8), within a bounded domain of $\overline{x}$, is given by

$$\max\left|\Psi(\overline{x})-\psi_{\wp}(0,\overline{x})\right| \leq C(\Delta w)^{2} \quad . \tag{10}$$

By combining Eqs. (8) and (9) and replacing the function $\psi(\overline{x})$ by the neuron transfer function $\Upsilon(\overline{x})$ it can be shown that the polynomial $\overline{x}^{\wp}$ can be approximated by a network layer of $\prod_{j=1}^{s}(p_j+1)$ neurons. The error of this approximation can be made arbitrarily small by the selection of Δw and θ. For example, the linear function x can be approximated by x_a, where

$$x_a = \left(2\Delta w \;\; \Upsilon^{(1)}(\theta)\right)^{-1}\left[\Upsilon(\Delta w x+\theta)-\Upsilon(-\Delta w x+\theta)\right] \quad . \tag{11}$$

This approach is also valid for constructing products of dependent variables,

$$\begin{aligned} x_a\, y_a = \left(4\Delta w_x\;\Delta w_y\;\Upsilon^{(2)}(\theta)\right)^{-1}&\left[\Upsilon(\Delta w_x x+\Delta w_y y+\theta)+\Upsilon(-\Delta w_x x-\Delta w_y y+\theta)+\right.\\ &\left.-\Upsilon(-\Delta w_x x+\Delta w_y y+\theta)-\Upsilon(\Delta w_x x-\Delta w_y y+\theta)\right] \end{aligned} \tag{12}$$

and multilayered approximations

$$x_{a2} = \left(2\Delta w \;\; \Upsilon^{(1)}(\theta)\right)^{-1}\left[\Upsilon(\Delta w x_{a1}+\theta)-\Upsilon(-\Delta w x_{a1}+\theta)\right] \quad , \tag{13}$$

where x_{a1} is the approximation from Eq.(11) and x_{a2} is the result from the second layer of neurons.

In section 4.1.1 this approach will be combined with the mathematical model of a physical system to obtain a reliable and accurate scheme for RANN construction for training initialization.

3 APPROACH: GTR

3.1 Background

As previously mentioned, GTR was specifically developed to solve ill-posed problems. The concept of ill-posed problems can be illustrated with a simple example. Assume some unknown or underlying physical process, u, produces an observable output h, i.e.,

$$\mathcal{F}(u) = h \ , \tag{14}$$

where for a linear operator $\mathcal{F}$, h may be an approximation of u. On the other hand, $\mathcal{F}$ may be nonlinear and h and u may exist in entirely different function spaces, in which case h is defined as the"proxy" output.

The measured outputs h_e are the experimental data defined as

$$h_e(x_i) \ = \ h(x_i) + \mu_i \tag{15}$$

where μ_i denotes the random noise of measurements at the coordinate x_i. If h is a proxy output then $h_e(x_i)$ is referred to as the proxy data. The inverse problem is to construct an approximating hypersurface

$$u_{\text{gtr}}(x) \ = \ \sum_k \Phi_k(x) c_k \ \ , \tag{16}$$

where c_k are coefficients corresponding to the basis functions Φ_k. The function $u_{\text{gtr}}(x)$ is designed to pass approximately through the experimental data points $h_e(x_i)$. The construction of a response surface from the minimization of the standard mean squared error,

$$\sum_{i=1}^{N} \left(h_e(x_i) - \mathcal{F}(u_{\text{gtr}}(x_i))\right)^2 \tag{17}$$

constitutes an ill-posed problem in the sense of Hadamard [19] because the reconstructed hypersurface $u_{\text{gtr}}(x)$ is non-unique and is sensitive to noise in the data.

The single central idea in GTR is the minimization of an error term ϵ where

$$\epsilon = \mathcal{A} + \lambda \mathcal{B} \tag{18}$$

where $\mathcal{A}$ is a measure of the agreement between the hypersurface approximation $u_{\text{gtr}}(x)$ and the data and can usually be written as

$$\mathcal{A} = \left(\sum_{i=1}^{N} |h_e(x_i) - \mathcal{F}(u_{\text{gtr}}(x_i))| - \delta \right)^2 \tag{19}$$

where

$$\sum_{i=1}^{N} |h_e(x_i) - \mathcal{F}(u_{\text{gtr}}(x_i))| \leq \delta \quad . \tag{20}$$

In Bayesian terms $\mathcal{A}$ is related to a posteriori knowledge. $\mathcal{B}$ is a stabilizing functional, also known as the regularizing operator, which can usually be written as

$$\mathcal{B} = (\Lambda(u_{\text{gtr}}) - \eta)^2 \tag{21}$$

where

$$\Lambda(u_{\text{gtr}}) \leq \eta \quad , \tag{22}$$

and is related to a priori information. In practice $\mathcal{B}$ is classified as either quantitative or qualitative and can range from systems of time-dependent nonlinear partial differential equations, to statistical correlations, to heuristics. The regularizing parameter λ controls the relevance of the a posteriori and a priori information to the hypersurface u_a. Finding an approximate solution to Eq.(18) reduces to: (a) finding regularizing operators $\mathcal{B}$, and (b) determining the regularization parameter λ from supplementary information pertaining to the problem, for example, pertaining to the noise level in h_e. It is apparent then that the form of $\mathcal{B}$ is not unique and depends on the type of physical process and the data type. In addition, the direct determination of λ is an area of active research.

It is believed that the framework offered by GTR is ideally suited for the intelligent analysis of experimental data and the seamless merging of data and computational mechanics models in ANN architecture. However, finding an approximate solution to Eq.(14)

reduces to finding regularizing operators Λ, and determining the regularization parameter λ from supplementary information pertaining to the problem, for example, pertaining to the size of the error in h_e.

It is apparent then that the form of $\mathcal{B}$ is not unique and depends on the type of physical process and the type of data h_e. In practice $\mathcal{B}$ are classified as either quantitative or qualitative and can range from systems of time-dependent nonlinear partial differential equations, to statistical correlations, to heuristics. In the utilization of GTR in this paper, the physical model in nonvariational form [20] is put in place of Λ and the minimization of ϵ is replaced with a conventional computational mechanics form using the method of weighted residuals (MWR) [21].

Two classes of methods can be used to solve the regularization problem: implicit methods, and explicit methods. In implicit methods, the regularizers are formed from a priori mathematical models. Until recently, such regularizers had only been developed when the optimization problem of Eq.(18) could be cast in variational form. Explicit methods, by contrast, treat the mathematical model as a numerical (or synthetic) data stream to be fused with the experimental data. In this explicit approach, the regularization step is applied as a post-processing step. The equations of the mathematical model are replaced with a smoothing functional applied to the difference between and the numerical data. It has been shown [20] that both implicit and explicit approaches to regularization can be interpreted as an integration of experimental and computational data by which the theoretical response of a physical system is "fine-tuned" through available experimental data and the adjustment of λ. It has also been shown [20] that the developed method can be used efficiently to process and filter noisy measurements. The response of the mathematical model characterizes target features of the physical system, which are then used to separate the serviceable part of the data from its noisy component.

Say the mathematical model of the mechanics problem can be described by the equations

$$L[u(x)] - g(x) = R(x) = 0 \ , \ x \in \Omega \quad \text{and} \quad B[u(x)] = 0 \ , \ x \in \partial\Omega \quad , \tag{23}$$

where $L[\cdot]$ is a general differential operator, $B[\cdot]$ is the boundary operator and $R(x)$ is the equation residual. The solution to Eq. (23) can also be determined by minimization of the objective function,

$$\epsilon(u_{\rm gtr}) = \sum_{i=1}^{N} \left(|h_e(x_i) - h_{\rm gtr}(x_i)| - \delta\right)^2 + \lambda \left(\langle R_{\rm gtr}(x), R_{\rm gtr}(x)\rangle + \left(B[u_{\rm gtr}]\right)^2 - \eta\right) \quad . \tag{24}$$

where $\langle\cdot\rangle$ represents the inner product and $R_{\rm gtr}(x)$ represents the equation residual for $u_{\rm gtr}$. Preselecting the distribution and localization properties of the basis functions, Eq. (24) is minimized with respect to the remaining unknowns, c_k, resulting in

$$\left(\lambda \mathbf{G} + \mathbf{Q}\right) \bar{c} = \lambda \mathbf{b} + \mathbf{e} \ . \tag{25}$$

where

$$\mathrm{G}_{kj} = \langle \psi_k(x), L[\Phi_j]\rangle \ , \quad \mathrm{Q}_{kj} = \sum_{i=1}^{N} \mathcal{F}[\Phi_k(x_i)]\mathcal{F}[\Phi_j(x_i)] \ , \quad \mathrm{b}_k = \langle \psi_k(x), g\rangle \ ,$$

$$\text{and} \quad \mathrm{e}_k = \sum_{i=1}^{N} \mathcal{F}[\Phi_k(x_i)] \left(h_e(x_i) - \delta\right) \quad .$$

The symbol $\mathbf{G}$ represents the conventional discretization matrix from computational mechanics, $\mathbf{Q}$ is a matrix containing the independent variable information of the measurement points, the vector $\mathbf{b}$ is the forcing term of the a priori mathematical model, $\mathbf{e}$ represents the observational data, and $\psi(x)$ is the weighting function. Without regularization, Eq.(25) is ill-posed. The addition of the model-based $\mathbf{G}$ and $\mathbf{b}$ terms effectively desingularizes the problem. One advantage of this method is that it applies to all of the conventional numerical techniques (such as finite elements, finite volume, finite difference, and spectral methods) with the appropriate choices of method parameters. These method parameters involve the choice of basis functions, weighting functions, and integral quadrature.

Use of Eq.(25) requires a-priori knowledge regarding the intensity of the noise in measurements (δ). This is usually available in engineering applications. Also, Eq.(25) suggests that the regularization technique will work equally well by equating the equation residual of the a-priori mathematical model to the function residual of the $h_e - h_{\rm gtr}$. Therefore, the value of the regularization parameter λ can be determined such that the trace of $\lambda\mathbf{G} + \mathbf{Q}$ is close to zero.

4 RESULTS

4.1 Developing A RANN Model of Duffing's Equation

The a-priori mathematical model of the the inhomogeneous Duffing's equation can be described by the following second-order ordinary differential equation

$$\frac{\mathrm{d}^2\mathrm{x}}{\mathrm{dt}^2} + 2\mu\frac{\mathrm{dx}}{\mathrm{dt}} - \frac{1}{2}\left(x - x^3\right) = F_0\ \cos\left(\omega t\right) \tag{26}$$

where t, x, μ, F_0, and ω represent the time, displacement, damping coefficient, force amplitude, and frequency of excitation, respectively. Equation (26) describes the forced motion of a particle between two equilibrium states. The importance of this equation is that its chaotic and nonchaotic behavior has been extensively examined by theoretical [22], experimental [23], and numerical methods [11], [24] - [26].

To make the Duffing's equation more easily approachable by the RANN construction method, and to allow the user to obtain particle displacement and velocity, Eq.(26) is reduced to a system of two first-order equations by the following change of variables

$$s = \frac{x}{\kappa_1} \quad , \quad y = \frac{1}{\kappa_2}\frac{\mathrm{dx}}{\mathrm{dt}} = \frac{1}{\kappa_2}\dot{x} \quad , \tag{27}$$

where κ_1 and κ_2 are constants used to scale the new variables. Therefore, Eq.(26) becomes

$$\frac{\mathrm{ds}}{\mathrm{dt}} = \kappa_3 y \tag{28}$$

$$\text{where} \quad \kappa_3 = \frac{\kappa_2}{\kappa_1} \quad \text{and}$$

$$\frac{\mathrm{dy}}{\mathrm{dt}} = -2\mu y + \frac{1}{2\kappa_3}\left(s - {\kappa_1}^2 s^3\right) + \frac{F_0}{\kappa_2}\ \cos\left(\omega t\right) \quad . \tag{29}$$

4.1.1 RANN Model of the Duffing's Equation

For this paper integration of the initial value problem of Eqs. (28) and (29) will be made, for arbitrary coefficients, by the explicit MacCormack method [27]. The MacCormack method is a finite-difference, predictor-corrector scheme commonly used in the solution of time-dependent fluid dynamics equations. It should be noted that the RANN programming approach can use any time marching technique that can be put into explicit form (e.g.

Runge-Kutta method). The application of the MacCormack technique to a general first-order ordinary differential equation

$$\frac{\mathrm{du}}{\mathrm{dt}} - f(u,t) = 0 \ ,$$

where f is some arbitrary function of the dependent variable u and independent variable t, results in

$$\text{Predictor:} \qquad u^* = u^n + \Delta t f(u^n, t^n)$$

$$\text{Corrector:} \qquad u^{n+1} = \frac{1}{2}\left(u^n + u^* + \Delta t f(u^*, t^*)\right) = u^n + \frac{\Delta t}{2}\left(f(u^n, t^n) + f(u^*, t^*)\right)$$

where the superscripts denote the time level and $t^* = t^n + \Delta t$.

Application of the MacCormack method to Eqs. (28) and (29) results in

$$\begin{aligned} s^* &= s^n + \Delta t f_1(s^n, y^n, t^n) \\ s^{n+1} &= s^n + \frac{\Delta t}{2}\left(f_1(s^n, y^n, t^n) + f_1(s^*, y^*, t^*)\right) \qquad (30) \\ y^* &= y^n + \Delta t f_2(s^n, y^n, t^n) \\ y^{n+1} &= y^n + \frac{\Delta t}{2}\left(f_2(s^n, y^n, t^n) + f_2(s^*, y^*, t^*)\right) \qquad (31) \end{aligned}$$

where

$$f_1(s,y,t) = \kappa_3 y \quad \text{and} \quad f_2(s,y,t) = -2\mu y + \frac{1}{2\kappa_3}\left(s - {\kappa_1}^2 s^3\right) + \frac{F_0}{\kappa_2}\cos(\omega t)$$

Substitution of the expressions for f_1, f_2, s^*, and y^* into the equations for s^{n+1} and y^{n+1} result in the following algebraic system,

$$z^n = (s^n + \kappa_3 \Delta t \ y^n) \qquad (32)$$

$$s^{n+1} = H_1 s^n + H_2 y^n + H_3 (s^n)^3 + H_4 \cos(n\omega\Delta t) \qquad (33)$$

$$y^{n+1} = H_5 s^n + H_6 y^n + H_7 (s^n)^3 + H_8 (z^n)^3 + H_9 \cos(n\omega\Delta t) - H_{10} \sin(n\omega\Delta t) \qquad (34)$$

$$\sin((n+1)\omega\Delta t) = H_{11} \sin(n\omega\Delta t) + H_{12} \cos(n\omega\Delta t) \qquad (35)$$

$$\cos((n+1)\omega\Delta t) = H_{11} \cos(n\omega\Delta t) - H_{12} \sin(n\omega\Delta t) \qquad (36)$$

The coefficients H_1 through H_{12} are

$$H_1 = 1 + \frac{\Delta t^2}{4} \quad , \quad H_2 = \kappa_3 \Delta t \, (1 - \mu \Delta t) \quad , \quad H_3 = -\frac{\kappa_1^2 \Delta t^2}{4}$$

$$H_4 = \frac{F_0 \Delta t^2}{2\kappa_1} \quad , \quad H_5 = \frac{\Delta t}{2\kappa_3}(1 - \mu \Delta t) \quad , \quad H_6 = 1 - 2\mu\Delta t + 2\mu^2 \Delta t^2 + \frac{\Delta t^2}{4}$$

$$H_7 = -\frac{{\kappa_1}^3}{\kappa_2}\frac{\Delta t}{4}(1 - 2\mu\Delta t) \quad , \quad H_8 = -\frac{{\kappa_1}^3}{\kappa_2 {\kappa_4}^3}\frac{\Delta t}{4} \quad , \quad H_9 = \frac{F_0}{\kappa_2}\frac{\Delta t}{2}(1 - 2\mu\Delta t + \ \cos(\omega\Delta t))$$

$$H_{10} = -\frac{F_0}{\kappa_2}\frac{\Delta t}{2} \ \sin(\omega\Delta t) \quad , \quad H_{11} = \cos(\omega\Delta t) \quad , \quad H_{12} = \sin(\omega\Delta t) \tag{37}$$

If it is required that Δt be constant then the time dependent coefficients of Eq.(37) become constants for specific values of μ, F_0, κ_1 and κ_2. Equations (32) through (36) are the linear and nonlinear algebraic equations that approximate Duffing's equation and that must be modelled by the RANN.

For the sake of convenience the magnitude of all dependent variables are scaled to lie in the closed interval $[-1, 1]$. To satisfy this constraint on the magnitude of the dependent variables s^n, y^n and z^n it is required that

$$\kappa_1 > |x| \quad , \quad \kappa_2 > \left|\frac{\mathrm{dx}}{\mathrm{dt}}\right| \quad \text{and} \quad \kappa_4 \geq 1 + \Delta t \kappa_3 \quad , \quad \text{respectively.} \tag{38}$$

Using the results of section 2.2.2 the time dependent unknowns of the nonlinear set of algebraic equations can be approximated by the linear combination of transfer functions Υ with a user specified order of accuracy. Assuming ξ^n represents the dependent variables s^n, y^n, z^n, $\sin(n\omega\Delta t)$ and $\cos(n\omega\Delta t)$, an eighth-order accurate scheme was chosen for the approximation $({\xi_a}^n)$ of the the linear and cubic terms, with $\theta = 0$ and assuming that $\Upsilon(-\xi) = -\Upsilon(\xi)$, $\Upsilon^{(1)}(0) \neq 0$ and $\Upsilon^{(3)}(0) \neq 0$.

$$\begin{aligned} {\xi_a}^n = \left(5\Delta w \ \Upsilon^{(1)}(0)\right)^{-1} \Big[& 8\Upsilon(\Delta w \xi^n) - 2\Upsilon(2\Delta w \xi^n) + (21)^{-1}(8)\Upsilon(3\Delta w \ \xi^n) \\ & -(28)^{-1}\Upsilon(4\Delta w \xi^n)\Big] + O\left(\Delta w^8\right) \end{aligned} \tag{39}$$

$$\begin{aligned} ({\xi_a}^n)^3 = \left(2520\Delta w^3 \ \Upsilon^{(3)}(0)\right)^{-1} [& -11683\Upsilon(\Delta w \xi^n) + 8738\Upsilon(2\Delta w \xi^n) \\ & -(2)^{-1}(4869)\Upsilon(3\Delta w \ \xi^n) + (3)^{-1}(1261)\Upsilon(4\Delta w \xi^n) \\ & -(12)^{-1}(410)\Upsilon(5\Delta w \xi^n)\Big] + O\left(\Delta w^8\right) \end{aligned} \tag{40}$$

Since the updated dependent variables must each exit a single node in the second layer, the following approximation was used:

$$m\xi_a{}^{n+1} \;=\; m\left(\Delta w\ \Upsilon^{(1)}(0)\right)^{-1}\Upsilon(\Delta w\ \xi^{n+1}) + O\left(\Delta w^2\right) \quad . \tag{41}$$

where m is an integer constant.

Using these approximations with $\Upsilon(\xi) = \tanh(\xi)$, Eqs. (32) through (36) become

$$m_1 \Delta w s_a{}^{n+1} \;=\; m_1 \tanh\left[\Delta w\ H_1 s_a{}^n + ... + \Delta w\ H_4\ \cos_a(n\omega\Delta t)\right] \tag{42}$$

for $m_1 = 1, ..., 5$,

$$m_2 \Delta w y_a{}^{n+1} \;=\; m_2 \tanh\left[\Delta w\ H_5 s_a{}^n + ... - \Delta w\ H_{10}\ \sin_a(n\omega\Delta t)\right] \tag{43}$$

for $m_2 = 1, ..., 4$,

$$m_3 \Delta w z_a{}^{n+1} \;=\; m_3 \tanh\left[\Delta w\ s_a{}^n + \kappa_3 \Delta t\ y_a{}^n\right] \quad \text{for } m_3 = 1, ..., 5 \ , \tag{44}$$

$$m_4 \Delta w\ \sin_a((n+1)\omega\Delta t) \;=\; m_4 \tanh\left[\Delta w\ H_{11}\ \sin_a(n\omega\Delta t) + \Delta w\ H_{12}\ \cos_a(n\omega\Delta t)\right] \tag{45}$$

for $m_4 = 1, ..., 4$ and

$$m_5 \Delta w\ \cos_a((n+1)\omega\Delta t) \;=\; m_5 \tanh\left[\Delta w\ H_{11}\ \cos_a(n\omega\Delta t) - \Delta w\ H_{12}\ \sin_a(n\omega\Delta t)\right] \tag{46}$$

for $m_5 = 1, ..., 4$.

Figure 2 is a sketch of the fully connected two-layer, twenty two node, RANN given by Eqs. (42) through (46), which are third-order accurate. An initial time $t^0 = 0$ is assumed.

4.2 Modification of an ANN Generated Chaotic Time Series

The explicit approach outlined in section 3.1 was chosen for the regularization of the RANN. In this approach merit function is written as

$$\epsilon = \left(\sum_{i=1}^{N}\left|s_e(t^i) - s_{\rm gtr}(t^i)\right| - \delta_1\right)^2 + \left(\sum_{i=1}^{N}\left|y_e(t^i) - y_{\rm gtr}(t^i)\right| - \delta_2\right)^2 + \lambda_1\left(\langle R1_{\rm gtr}(x), R1_{\rm gtr}(x)\rangle - \eta_1\right) + \lambda_1\left(\langle R2_{\rm gtr}(x), R2_{\rm gtr}(x)\rangle - \eta_2\right) \quad , \tag{47}$$

where

$$R1_{\rm gtr}(t) = \frac{{\rm d}\left(s_{\rm gtr} - s_a\right)}{{\rm d}t} + \left(s_{\rm gtr} - s_a\right)$$
$$R2_{\rm gtr}(t) = \frac{{\rm d}\left(y_{\rm gtr} - y_a\right)}{{\rm d}t} + \left(y_{\rm gtr} - y_a\right) \quad ,$$

If both the regularized and the pre-existing numerical approximations of s and y can be represented by a weighted sum of piecewise linear basis functions, we can write their respective time series up to time level n as

$$s_{\rm gtr}(t) \;\; = \;\; \sum_{j=1}^{n} \left(\Phi_j(t) s_{\rm gtr}^{j} + \varphi_{j-1}(t) s_{\rm gtr}^{j-1} \right) \quad ,$$

where

$$\Phi_j(t) \equiv \begin{cases} \left(t - t^{j-1}\right)/\Delta t^{j-1} & t^{j-1} \le t \le t^{j} \\ \\ 0 & \text{otherwise} \end{cases}$$

$$\varphi_j(t) \equiv \begin{cases} \left(t^{j+1} - t\right)/\Delta t^{j} & t^{j} \le t \le t^{j+1} \\ \\ 0 & \text{otherwise} \end{cases}$$

where $\Delta t^{j-1} = t^j - t^{j-1} = \Delta t^j = t^{j+1} - t^j = \Delta t$ for a constant time step.

Using s as an example, Eq.(47) is minimized with respect to the time level n, resulting in

$$\lambda_1 \sum_{j=1}^{n} \left(\langle \psi_n, L\left[\Phi_j\right] \rangle s_{\rm gtr}^{j} + \langle \psi_n, L\left[\varphi_{j-1}\right] \rangle s_{\rm gtr}^{j-1} \right) + \sum_{j=1}^{n} \sum_{i=1}^{N} \left(\Phi_n(t^i) \Phi_j(t^i) s_{\rm gtr}^{j} + \Phi_n(t^i)\; \varphi_{j-1}(t^i) s_{\rm gtr}^{j-1} \right) =$$
$$\lambda_1 \sum_{j=1}^{n} \left(\langle \psi_n, L\left[\Phi_j\right] \rangle s_{a}^{j} + \langle \psi_n, L\left[\varphi_{j-1}\right] \rangle s_{a}^{j-1} \right) + \sum_{i=1}^{N} \Phi_n(t^i) \left(y_e(t^i) - \delta_1 \right) \quad .$$

where

$$\psi_n = \frac{{\rm d}\Phi_n}{{\rm d}t} + \Phi_n \quad , \quad L\left[\Phi_j\right] = \frac{{\rm d}\Phi_j}{{\rm d}t} + \Phi_j \quad , \text{ and } \quad L\left[\varphi_j\right] = \frac{{\rm d}\varphi_j}{{\rm d}t} + \varphi_j \quad .$$

Restricting our attention to the interval between time levels t^{n-1} and t^n, we can finally write the linear algebraic equation that computes the regularized system response as,

$$\left(\lambda_1 \left(2\Delta t + 3\right) + 12 \sum_{i=1}^{N} \Phi_n(t^i) \Phi_n(t^i) \right) s_{\rm gtr}^{n} = - \left(\lambda_1 \left(\Delta t - 3\right) + +12 \sum_{i=1}^{N} \Phi_n(t^i)\; \varphi_{n-1}(t^i) \right) s_{\rm gtr}^{n-1} +$$
$$\lambda_1 \left(2\Delta t + 3\right) s_a^n + \lambda_1 \left(\Delta t - 3\right) s_a^{n-1} + 12 \sum_{i=1}^{N} \Phi_n(t^i) \left(s_e(t^i) - \delta_1 \right) \quad . \tag{48}$$

The algebraic equation for $y^n_{\rm gtr}$ is nearly identical,

$$\left(\lambda_2\left(2\Delta t+3\right)+12\sum_{i=1}^{N}\Phi_n(t^i)\Phi_n(t^i)\right)y^n_{\rm gtr}=-\left(\lambda_2\left(\Delta t-3\right)++12\sum_{i=1}^{N}\Phi_n(t^i)\;\varphi_{n-1}(t^i)\right)y^{n-1}_{\rm gtr}+$$
$$\lambda_2\left(2\Delta t+3\right)y^n_a+\lambda_2\left(\Delta t-3\right)y^{n-1}_a+12\sum_{i=1}^{N}\Phi_n(t^i)\left(y_e(t^i)-\delta_2\right)\quad . \tag{49}$$

The RANN of Fig. 2 was constructed with the damping value and excitation amplitude of $\mu = 0.084$ and $F_0 = 0.178$, respectively, and run for the same forcing frequency range ($\omega = 0.633$ to $\omega = 1.02$) illustrated in Fig. 3. Previous results of the two well potential system [11] indicate a maximum displacement between $\pm$ 2.0 and maximum velocity between $\pm$ 1.0. As a result $\kappa_1 = 4.0$ and $\kappa_2 = 2.0$. Initial conditions were set for $x(0) = 1.0$ and $\dot{x}(0) = 0.0$. To ensure that steady-state solutions were displayed, approximations were run for 900 periods, T, based on the forcing frequency ω (T $= 2\pi/\omega$). Throughout the simulation the time step ($\Delta t =$ T/200) used in the RANN was half that used in the Masoud and Asfar study. All computations were done in single precision with $\Delta w = 1 \times 10^{-4}$ and all test cases were run in less than 40 seconds on a Sparc 20 workstation.

The Duffing equation with chaotic response ($\omega = 0.860$) shown in Fig. 4, was used as the a-priori mathematical model in approximating the non-chaotic solution from $\omega = 1.02$ as shown in Fig. 5. Two hundred data points ($N = 200$) for s_e and y_e were used to simulate experimental information, with $\delta_1 = \delta_2 = 0$ and $\lambda = 0.15$.

Figure 6(a) shows the transient response of the RANN when the GTR acts to modify the forcing function, using the explicit formulation of Eqs. (48) and (49). When in the region of the data, the response of the RANN is pulled from the a-priori trajectory, oscillates and settles into the target response. Figure 6(b) illustrates the satisfactory response of regularized RANN after the transients have settled out. One can use functional ANOVA [28] to permanently modify the forcing function or use parameter identification to modify the model equation.

5 CONCLUSIONS

A method was developed for manually constructing recurrent artificial neural networks to model the fusion of experimental data and mathematical models of physical systems. The construction required the use of GTR and imposing certain constraints on the values of the input, bias, and output weights. The attribution of certain roles to each of these parameters allowed for mapping a polynomial approximation into an artificial neural network architecture. GTR provided a rational means of combining theoretical models, computational data, and experimental measurements into a global representation of a domain. Particular attention was focused on the inhomogeneous Duffing's oscillator. The nonlinear ordinary differential equation was modelled by the recurrent artificial neural network architecture in conjunction with the popular hyperbolic tangent transfer function. GTR was then used to smoothly merge the response of the RANN and experimental data. A numerical example was presented illustrating the accuracy and utility of the method.

The author chose to map the algebraic equations, rather than the explicit differential equations, because it was thought that this will give greater utility to the engineer. The more popular numerical techniques, such as finite difference and finite elements, reduce differential equation to algebraic form for use on digital computers. As a result, the engineer can use these familiar and well understood techniques in the process of programming and regularizing the RANN.

6 ACKNOWLEDGEMENTS

This work was supported under ONR grant N00014-95-1-0741.

REFERENCES

[1] Rumelhart, D. E. and McClelland, J. L. (1986) *Parallel Distributed Processing: Explorations in the Microstructure of Cognition,* (MIT Press, Cambridge MA) pp. 318-362.

[2] Williams, R. J. and Zipser, D. (1989) A Learning Algorithm for Continually Running Fully Recurrent Neural Networks, *Neural Computation*, **1**, pp. 270-280.

[3] Pineda, F. J. (1989) Recurrent Backpropagation and the Dynamical Approach to Adaptive Neural Computation, *Neural Computation*, **1**, pp. 161-172.

[4] Saarinen, S., Bramley, R. and Cybenko, G. (1993) Ill-Conditioning in Neural Network Training Problems, *SIAM J. Sci. Comput.*, **14**, 3, pp. 693-714.

[5] Igelnik, B. and Pao, Y.-H. (1995) Stochastic Choice of Basis Function in Adaptive Function Approximation and the Function-Link Net, *IEEE Transactions on Neural Networks*, **6**, 6, pp. 1320-1329.

[6] Schidt, W.F., Raudys, S., Kraaijveld, M.A., Skurikhina, M. and Duin, R.P.W. (1993) Initializations, Back-Propagation and Generalization of Feed-Forward Classifiers, Proceedings of IEEE International Conference on Neural Networks, ICNN-93, Vol. 1, pp. 598-604.

[7] Poggio, T. and Girosi, F. (1990) Networks for Approximation and Learning, Proceedings of the IEEE, **78**, 9, pp. 1481-1497.

[8] Canto, C., Hussaini, M.Y., Quarteroni, A. and Zang, T.A. (1988) *Spectral Methods in Fluid Dynamics*, (Springer-Verlag, Berlin Heidelberg), pp. 31-75.

[9] Box, G.E.P. and Draper, N.R. (1987) *Empirical Model-Building and Response Surface* (Wiley, New York).

[10] Ulbrich, M. (1998) *A Generalized Tikhonov Regularization for Nonlinear Inverse Ill-Posed Problems,* TUM-M9810, Fak. F. Math, Techn. Univ. München, München, Germany.

[11] Masoud, K. K. and Asfar, K. R. (1993) Period Doublings, Bifurcations and Strange Attractors in a Two-Well Potential Oscillator, *European Journal of Mechanics, A/Solids*, **12**, 3, pp. 417-428.

[12] Hecht-Nielsen, R. (1990) *Neurocomputing* (Addison-Wesley, Inc., New York), pp. 183-191.

[13] Hornik, K., Stinchcombe, M. and White, H. (1989) Multilayer Feedforward Networks are Universal Approximators, *Neural Networks*, **2**, pp. 359-366.

[14] Hornik, K., Stinchcombe, M. and White, H. (1990) Universal Approximation of an Unknown Mapping and its Derivatives Using Multilayer Feedforward Networks, *Neural Networks*, **3**, pp. 551-560.

[15] Ito, Y. (1991) Representation of Functions by Superpositions of a Step or Sigmoidal Function and Their Applications to Neural Network Theory, *Neural Networks*, **4**, pp. 385-394.

[16] Leshno, M., Lin, V.Y., Pinkus, A. and Schocken, S. (1993) Multilayer Feedforward Networks with a Nonpolynomial Activation Function Can Approximate Any Function, *Neural Networks*, **6**, pp. 861-867.

[17] Mhaskar, H.N. (1995) Neural Networks for Optimal Approximation of Smooth and Analytic Functions, *Neural Computation*, **8**, pp. 164-177.

[18] Tikhonov, A. N. and Arsenin, V. Y. (1977) *Solution of Ill-Posed Problems,* V. H. Winston & Sons, Washington, D.C.

[19] Hadamard, J., *Lecture on Cauchy Problem in Linear Partial Differential Equations*, Yale University Press, New Haven, CT, 1923.

[20] Zeldin, B. A. and Meade, A. J. (1997) Integrating Experimental Data and Mathematical Models in Simulation of Physical Systems, *AIAA J.*, **35**, 11, pp. 1787-1790.

[21] Fletcher, C. A. J. (1984) *Computational Galerkin Methods*, Springer, New York, N.Y., 1984.

[22] Holmes, P. (1979) A Nonlinear Oscillator with a Strange Attractor, *Phil. Trans. The Roy. Soc. London*, **292**, pp. 419-448.

[23] Moon, F. C. (1980) Experiments on Chaotic Motions of a Forced Nonlinear Oscillator: Strange Attractors, *ASME Journal of Applied Mechanics*, **47**, pp. 638-644.

[24] Ueda, Y. (1979) Randomly Transitional Phenomena in the System Governed by Duffing's Equation, *Journal of Statistical Physics*, **20**, 2, pp. 181-196.

[25] Dowell, E. H. and Pezeshki, C. (1986) On the Understanding of Chaos in Duffings Equation Including a Comparison With Experiment, *Journal of Applied Mechanics*, **53**, pp. 5-9.

[26] Pezeshki, C. and Dowell, E. H. (1988) On Chaos and Fractal Behaviour in a Generalised Duffing's System, *Physica D*, **32**, pp. 194-209.

[27] Anderson, D. A., Tannehill, J. C. and Pletcher, R. H. (1984) *Computational Fluid Mechanics and Heat Transfer* (Hemisphere Publishing Corporation, New York), pp. 163-164.

[28] Lin, X., Wahba, G., Xiang, D. Gao, F., Klein, R. and Klein, B. (1998) *Smoothing Spline ANOVA Models for Large Data Sets With Bernoulli Observations and Randomized GACV*, TR No. 998, Department of Statistics, University of Wisconsin, Madison, WI.

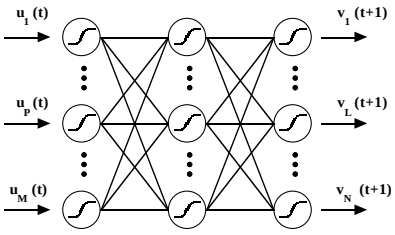

Figure 1: General RANN architecture.

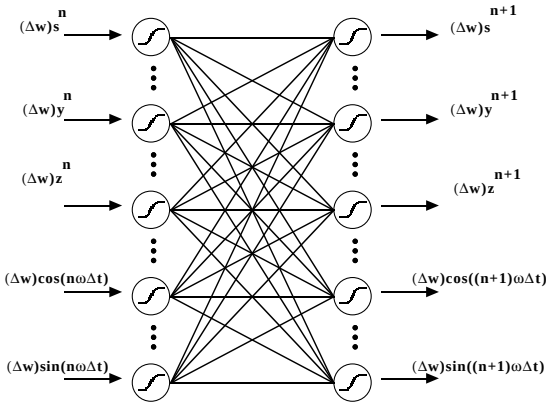

Figure 2: Complete two-layer twenty two node RANN assembly from the network construction scheme applied to Duffing's equation.

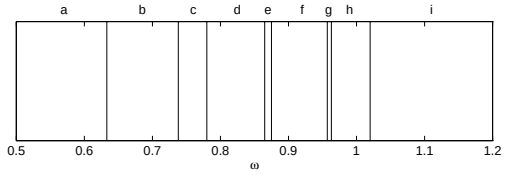

Figure 3: Classification of frequency response for the inhomogenous Duffing's equation: (a) and (i) period-1 motion; (b) second chaotic region; (c) evolution of period-3 attractor into chaos; (d) first chaotic region; (e) period doubling bifurcations in the period-5 attractor; (f) bifurcations from the symmetric period-5 attractor to asymmetric one; (g) coexistence of the period-5 attractor; (h) period doubling bifurcations.

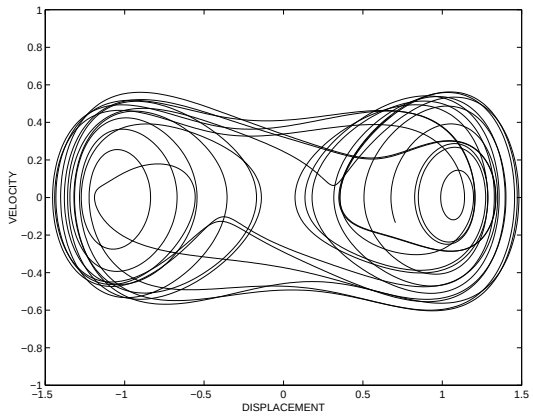

Figure 4: Chaotic phase space trajectory of the a-priori model ($\omega = 0.860$).

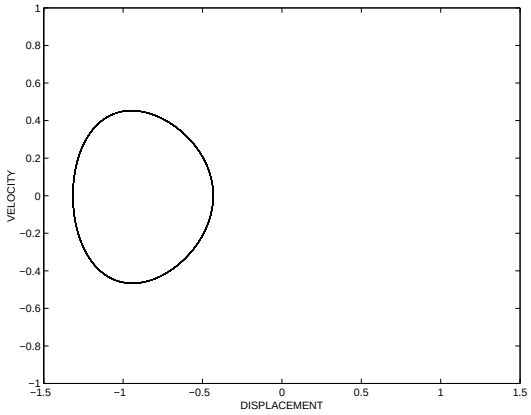

Figure 5: Desired nonchaotic trajectory ($\omega = 1.02$).

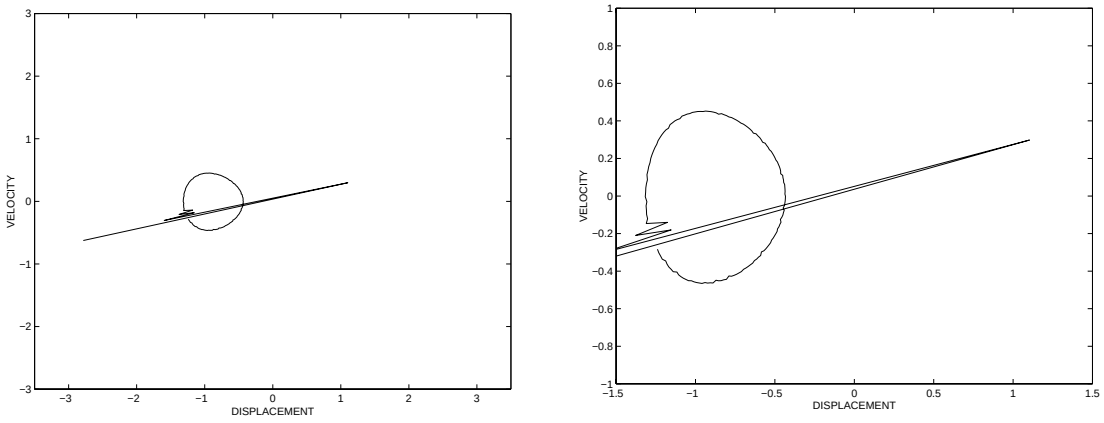

Figure 6: (a) Transients of the regularized RANN using GTR. (b) Response of the RANN using GTR using $\lambda = 0.15$.