Information Technology Journal 3 (1): 6-11, 2004
ISSN 1682-6027

Genetic Design of Neural PID plus Feed Forward Controllers

Naim Ajlouni and Sadeq Al-Hamouz
Faculty of Computer Science and Information, Applied Science University, 11931, Amman-Jordan

Abstract: Evolutionary techniques are proposed in a new and novel paradigm to solve the problem of designing a robust neural PID plus feed forward controller for a plant with prescribed plant parameter uncertainties. The evolutionary scheme used, involves generating two separate populations, one representing the controller and the other the plant. The controller population is then evolved against a fixed population of plants representing the uncertainty space, such that the controller can control all these plants effectively. A cost function involving time-domain performance is then deployed, subject to a frequency domain stability constraint. The resulting paradigm results in a robust controller design with excellent time-domain performance. This evolutionary approach is illustrated by evolving a neural PID plus feed forward controller for a linear plant, which has a set of prescribed uncertainties.

Key words: Genetic algorithms, neural network, feed forward PID controller

INTRODUCTION

The majority of control system designs are based on a model of the plant. However, this model is often only an approximation to the real plant. The difference between the real plant and the model of the plant is referred to as model uncertainty and a control system is said to be robust if it can deal with significant variation between the model and the plant. In this context robustness has to address both stability and performance.

The quest for the design of robust control system has resulted in methodologies for the design of feedback control systems which guarantee stability and performance for a given nominal model of the physical system the majority of these methodologies stem from the influential work of Zames[1].

The technique involves translating the performance and stability requirements into weighting function used to derive a synthesis model. The choice of weighting functions is often a difficult process.

An alternative approach to the robust design of PID controllers has been proposed by Ajlouni[2-4] using the technique of co-evolutionary genetic algorithms. The technique of genetic algorithm was first proposed by Holland[5]. The technique was extended to embrace co-evolution by Hills[6]. The co-evolution is analogous to predator and prey and can be used to evolve robust designs. The technique was devised by Ajlouni to achieve the robust tuning of PID controllers to deal with plant uncertainty. The robustness feature was achieved by using two co-evolving populations in the genetic algorithm. In this study an alternative paradigm is proposed whereby the first population consist of a set of controllers which is subject to the normal genetic operations and a second population which consists of a set of plants which has been chosen to span the entire plant uncertainty space and are not subject to any genetic process. The population of controllers is then evolved against the population of plants in such a manner that a, robust controller emerges. The high performance is obtained by defining the cost function in the time domain and the stability feature is achieved by incorporating a frequency domain vector margin stability constraint. Furthermore, it is also possible to take advantage of the feed forward controller action to further improve performance over the uncertainty space, without changing the stability properties. In this case the feed forward controller has been defined as a three-turn feed forward controller. The feed forward controller is not subject to any constraint other than actuator saturation levels.

The aim of the genetic design is to evolve a neural PID plus feed forward controller which is effective over all the plant uncertainty space, using a minimisation of the maximum technique, this result in a controller that is optimised about the lowest performance point, whilst ensuring the time domain performance everywhere else in operating space is to be better. Moreover, the technique also ensures that the frequency domain stability constraint is met throughout the uncertainty space. The resulting paradigm thus provides an alternative technique for the design of robust, control systems.

PID plus feed forward controllers: In the PID plus feed forward control system, two controllers operate simultaneously in the same loop, in order to achieve a

Corresponding Author: Naim Ajlouni, Faculty of Computer Science and Information, Applied Science University, 11931, Amman-Jordan E-mail: naim_mah@hotmail.com

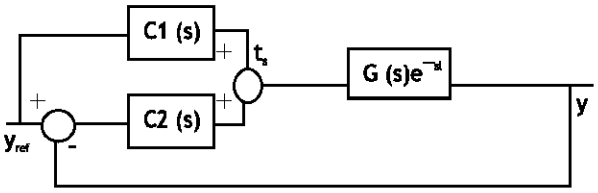

Fig. 1: PID plus feed forward control system

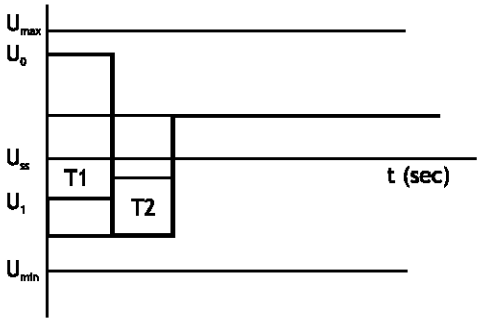

Fig. 2: Three-turn Feed forward controller

higher level of performance than using just one controller (Fig. 1).

The implementation of feed forward control used in this study, introduces the case where the feed forward controller is a simple three-turn controller (Fig. 2).

The three-turn feed forward digital controller is governed by the following equations:

$$\begin{cases} u(k) = u_0 < u_{max} & k = 0 \ldots . nT \\ u(k) = u_1 > u_{min} & k = (n+1)T \ldots . (n+m)T \\ u(k) = u_{ss} & k \geq (n+m+1)T \end{cases} \quad (1)$$

where $T \in R^+$ is the sampling period and u_{max} and u_{min} represent operational constraints of the actuator. In this case the feed forward controller parameters u_0, u_1, u_{ss}, nT and mT. are chosen so as to be most effective for set-point changes.

The digital PID controller is defined in the incremental form and is governed on the discrete time set {0,T,2T,...,kT,...} by control law equations of the form:

$$\Delta u(k)T = K_p(\Delta e(k)T + K_i e(k)T + k_d \Delta^2 e(k)) \quad (2)$$

where $T \in R^+$ is the sampling period,

$$e(k)T = v - y(k)T \quad (3)$$

is the error, the set-point and

$$\Delta e(k)T = e(k)T - e(k-1)T \quad (4)$$

$$\Delta^2 e(k)T = e(k)T - 2e(k-1)T + e(k-2)T \quad (5)$$

Neural PID controller gain function mapping: In order to use a neural network to map the non-linear gain functions for the non-linear incremental PID controller, the number of layers and the number of neurons in the layers has to be defined. In this work it is proposed to use 1 layer with 4 neurons, plus 1 input and 1 output neuron. From the above analysis it can be seen that the output function for the proposed network is given by:

$$y_i^{(1)} = f_i^{(0)}(e_k^{(0)}) \quad (6)$$

this is the output of the input layer, thus, the input to the first layer, is given by:

$$y_j^{(1)} = \sum_{i=1}^{p} w_{ji}^{(1)} y_i^{(0)} \quad (7)$$

and the output of the first layer is given by

$$y_j^{(1)} = f_j^{(1)}(e_j^{(1)}) \quad (8)$$

Finally in the output layer

$$y_k^{(2)} = \sum_{k=1}^{M} w_{kj}^{(2)} y_i^{(1)} \quad (9)$$

and the output of the output layer is given by

$$y_k^{(2)} = f_k^{(2)}(y_k^{(2)}) \quad (10)$$

From the above it can be seen that the neural network can be used to map the non-linear PID gain functions required by the non-linear PID controllers.

The above gain function is the integral gain function since the input to the network is e_k the tracking error, for the other two gain functions the input to the network is Δe_k and $\Delta^2 e_k$, for the proportional and derivative gain function, respectively. Fig. 3 shows the neural network representation of the three gain functions as they are mapped into the system.

The non-linear gain function in Fig. 3 is for an integral gain function. Also as it can be seen, the neural networks dose not have a bias input. This is because the gain functions are required to go through the origin. By omitting the bias and ensuring the sigmoidal function

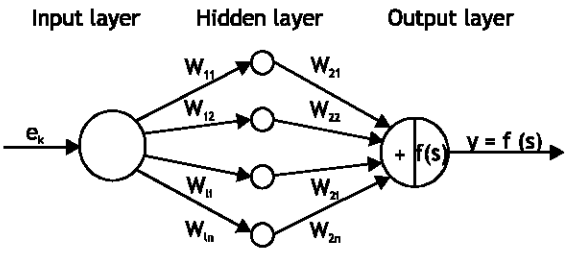

Fig. 3: Neural network representation of a gain function

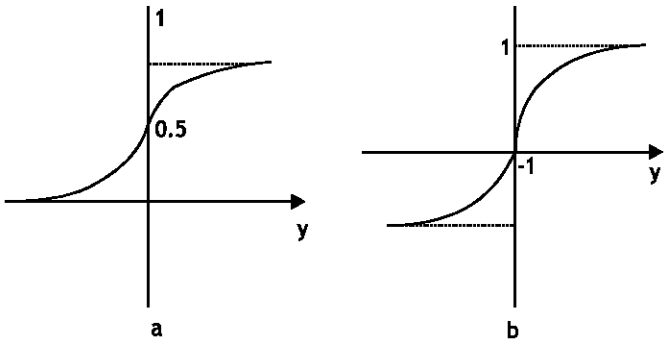

Fig. 4a: Typical sigmoidal function b: Unbiased sigmoidal function

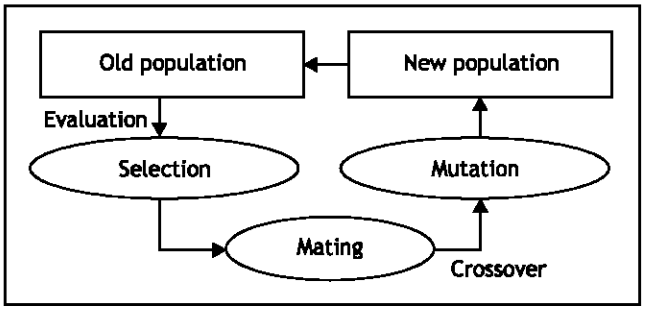

Fig. 5: The basic genetic algorithm cycle

goes through the origin, there are less parameters in the neural network and it is easier to search for the remaining parameters. The diagram in Fig. 4a shows a typical sigmoidal function resulting from a neural networks with a bias and Fig. 4b shows the sigmoidal function used in this research.

Genetic tuning of PID plus feed forward controllers: The genetic algorithm (GA), is a technique to search for the best answers to through problems and was first suggested by Holland[5]. Over the last 20 years, GAs have been used to solve a wide range of search, optimisation and machine learning problems[6]. As their name indicates, GAs attempt to solve problems in a fashion similar to the way in which biological optimisation processes seem to operate. The basic idea is to maintain a population of knowledge structures that represents candidate solutions for the current problem. The population evolves over time through competition (survival of the fittest) and controlled variation (recombination and mutation). In this way the best elements of the current population are used to form the new population. If this is done correctly then the new population will, on average, be "better" than the old population. Three processes are used to make the transition (reproduction) from one population to the next selection, mating and mutation. The basic GA cycle based on this processes is shown in Fig. 5.

In the PID plus feed forward control system, two controllers operate simultaneous in the same loop, to achieve a higher level of performance than using just one controller.

In order to use the GA to design the neural PID plus feed forward controller, the objective is to change the weights in the neural networks representing the PID controller gains, this is achieved by encoding the weights of the neural networks parameters in each set ({Wn1}, {Wn2}, {Wn3}) involved in the design equation (2) in accordance with a system of concatenated, multi-parameter, fixed point coding, plus the five parameters: $\{u_0, u_1, u_{ss}, T_1=nT$ and $T_2=mT\}$ associated with the feed forward controller.
Equation (3) becomes

$$\Delta u(k)\,T = Wn1\,(\Delta e(k)\,T + Wn2e(k)T + Wn3\Delta^2 e(k)$$

Thus each set of parameters: $\{u_0, u_1, u_{ss}, n, m,$ {Wn1}, {Wn2}, {Wn3}} are encoded as a binary string in accordance with a system of concatenated, multiparameter, mapped, fixed-point coding described in Goldberg[7] as a binary string.

Genetic tuning of robust PID plus feed forward controllers: In the case of the robust tuning of neural PID plus feed forward controllers to deal with the plant

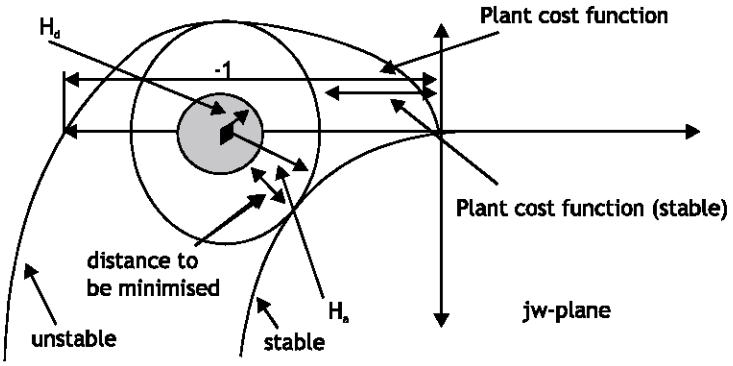

Fig. 6: Frequency domain cost function definition

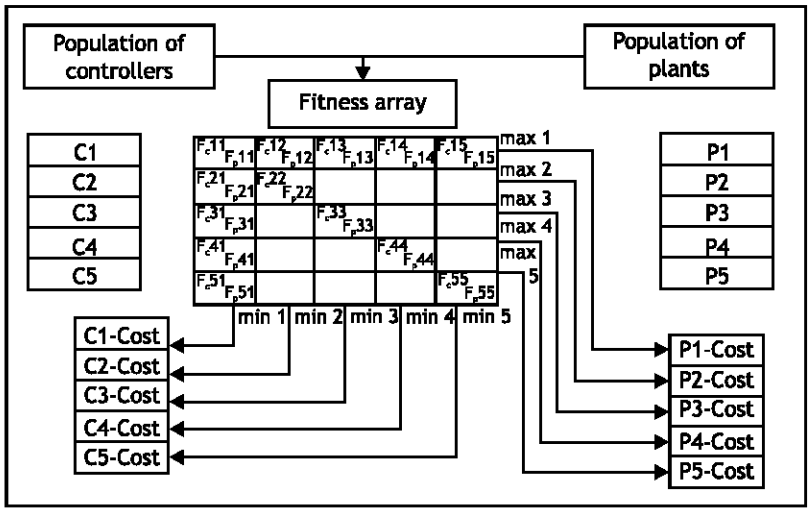

Fig. 7: Evolution fitness test procedure

uncertainty, the first task is to generate a population of plants which spans the uncertainty space to describe the fitness landscape. In order to use the evolutionary methods to design robust time-domain control systems, a population of controllers are evolved against the fixed population of plants within the uncertainty space.

To achieve this, the controller is used to control each of the plants in turn. Each controller-plant pair will then have an ISE and a vector margin (α). The robust stability property is obtained by setting a minimum vector margin (α_c), as indicated in Fig. 6, for the controller when considered against all the plants. If the controller can control all the plants without violating the stability constraint it is viewed to be a robust stable solution. However, if any controller plant pair violates the frequency domain constraint the controller is deemed unsuitable and is effectively eliminated from the population

The problem space for the controller thus consists of a population of the plants representing the plant uncertainty. The testing process is illustrated in Fig. 7. Each individual controller Ci from the controller population is tested against all the plants Pj. In this way the fitness associated with a given controller-plant pair is stored in the appropriate row and column of a bi-directional fitness array. The cost for each controller Ci is then chosen as the maximum of all the costs in the ith row, provided that each controller-plant pair satisfies the robustness constraint. The maximum cost is chosen because by using the GA to minimise the maximum cost, a controller emerges which can deal with all the other plants with a lower cost, whilst ensuring the robustness constraint is satisfied. This procedure effectively searches the plant population for the lowest performance plant and then tunes the controller up for this plant, whilst simultaneously checking that this tuning up procedure has not forced some of the other plants to either yield a worst controller-plant pair performance than the currently worse controller-plant pair, or violate the stability constraint. The evolutionary scheme can reach one of two equilibrium. The first equilibrium occurs when a low performance controller-plant pair is tuned up. It will

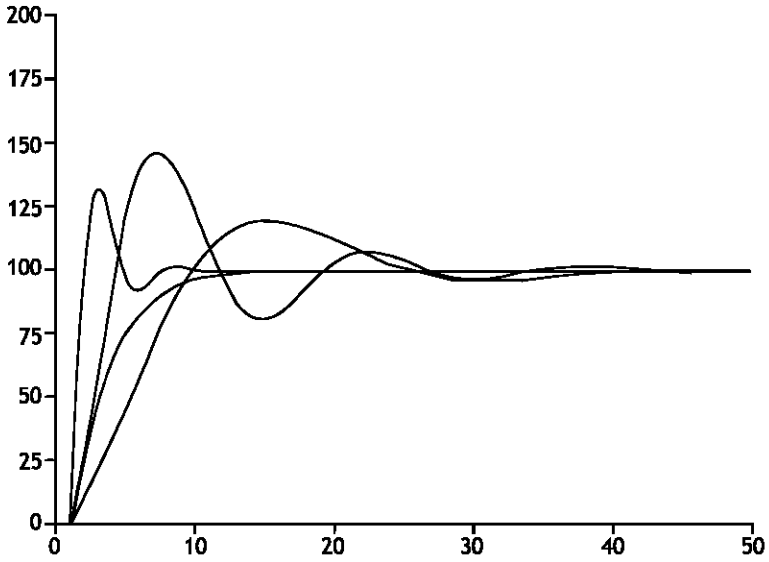

Fig. 8: Time domain robust frequency response performance for best controller and worst four plants

eventually force one of the other plants on to the stability constraint. The constraint then provides evolutionary pressure to de-tune the controller. The second equilibrium occurs when two low performance controller-plant pairs compete with one another for worst case position. This is seen when improving the ISE of one controller-plant pair degrades the ISE of another controller-plant pair and vice versa. The existence of one of these two equilibrium is the key to the robustness characteristic of the evolutionary algorithm, because this process forces the controller to continuously improve through an evolutionary process against the population of plants. Once the difficult plants are found in the search space the controller population will self-adapt to these plants in such a way as to force an emergent robust solution to the PID plus feed forward design problem.

Hence, the aim of the genetic optimisation is to minimise the ISE of the lowest performance plant subject to the frequency domain stability constraint being satisfied for all the plants.

Illustrative example: Many industrial processes can be modelled using a linear simple first order model with a transfer function of the form:

$$Gp(s)=\frac{K}{(1+sT)}e^{-sL}$$

where K is the dc gain, T is the time constant and L is the dead time. In this case the dead-time has been fixed at 1 second and the prescribed model plant uncertainties have been defined to lie in the following range: 0.3<K<1.0 and 5<T<1.25. The aim is to design a neural robust PID plus feed forward controller to cope with this plant variations, in such genetic neural control systems, the objective is to change the weights in the neural networks representing the PID controller gains such that the fitness function is defined as having an ISE subject to a frequency domain robust constraint vector margin VM>0.5.

In solving this problem two populations are used. The first population represents the controller parameters {U_0, U_1, U_{ss}, n, m, {Wn1}, {Wn2}, {Wn3}}. These parameters are encoded in accordance with a system of concatenated, multiparameter mapped, fixed-point coding[7]. Both the sets of neural PID gains and feed forward parameters, are then represented as strings of binary digits. Then, following random initial choice, entire generations of such strings can be processed in accordance with the basic genetic cycle described in Fig. 5. The selection procedure ensures that the successive generations of digital PID plus feed forward controllers produced by the genetic algorithm exhibit progressively improving behaviour in respect to the prescribed robustness fitness measure. In this way assuring that at the end of each generation the best PID plus feed forward controller is evolving against a set of fixed plants which span the uncertainty space of the plant.

The results of solving this problem by means of GAs, with population size of n=100, a crossover probability pc=0.65 and a mutation probability of pm=0.005 (Fig. 8). The resulting time domain robust frequency response performance for best controller and worst four plants, corresponding to K = 1 and T = 5, together with three other cases corresponding to K = 1 and T = 1.25; K = 0.3 and T = 1.25; and finally K = 0.3 and T = 5.

When the evolutionary process converges, an optimised controller emerges for the lowest performance plant, such that if this controller is used anywhere else in the operating envelope the time domain performance will be better, whilst simultaneously ensuring that at all operating points the stability constraint is satisfied.

Evolutionary techniques have been proposed in a new and novel paradigm to solve the problem of designing a robust neural PID plus feed forward controller in the time domain for a plant with prescribed plant parameter uncertainties. The evolutionary scheme used, involves generating two separate populations, one representing the controller and the other the plant. These two populations were then evolved such that the population of fixed plants, with the prescribed uncertainties, contains the set of difficult plants to control and a population of controllers evolve, which can control all these plants effectively. This co-evolutionary approach was illustrated through a co-evolving PID controller for a linear plant which has a set of prescribed uncertainties.

REFERENCES

1. Zames, G., 1981. Feedback and Optimal Sensitivity: Model reference transformation, multiplicative seminorms control, Ac-26, pp: 301-302.
2. Ajlouni, N.M., 1995. Genetic Algorithms for Control System Design, Ph.D. Thesis, Salford University, U.K.
3. Ajlouni, N.M., 2000a. Genetic Design of Robust PID Controller to Deal with Prescribed Plant Uncertainties Through a Process of Competitive Co-Evolution, Jordan J. Applied Sci., Applied Sci. Univ.
4. Ajlouni, N.M., 2000b. Genetic Design of Interpolated Non-Linear Controller for linear Plants. Al-Darast Jordan University.
5. Holland, J.H., 1975, Adaptation in Natural and Artificial Systems, The University of Michigan Press, Ann Arbor. G. Zames, 1981. Feedback and Optimal Sensitivity: Model Reference Transformations, Multiplictive Seminorms and Approximate Inverses, IEEE Trans. Aut. Control, AC-26, pp: 301-320.
6. Hills, H.D., 1990. Co-evolving Parasites Improve Simulated Evolution as an Optimization Procedure, Emeregent Computation: Self-Organizing Collective and Co-operative Computing Networks MIT press.
7. Goldberg, D.E. 1989, Genetic Algorithms in Search, Optimization and Machine Learning, Adison Wesley P.C.