To appear in *Proceedings of the 2002 Genetic and Evolutionary Computation Conference (GECCO-2002), 2002.*

Adaptive Control utilising Neural Swarming

Alex v.E. Conradie
Department of Chemical Engineering
University of Stellenbosch
South Africa
aconradi@ing.sun.ac.za

Risto Miikkulainen
Department of Computer Sciences
University of Texas at Austin
USA
risto@cs.utexas.edu

Christiaan Aldrich
Department of Chemical Engineering
University of Stellenbosch
South Africa
ca1@ing.sun.ac.za

Abstract

Process changes, such as flow disturbances and sensor noise, are common in the chemical and metallurgical industries. To maintain optimal performance, the controlling system has to adapt continuously to these changes. This is a difficult problem because the controller also has to perform well while it is adapting. The Adaptive Neural Swarming (ANS) method introduced in this paper satisfies these goals. Using an existing neural network controller as a starting point, ANS modifies the network weights through Particle Swarm Optimisation. The ANS method was tested in a real-world task of controlling a simulated non-linear bioreactor. ANS was able to adapt to process changes while simultaneously avoiding hard operating constraints. This way, ANS balances the need to adapt with the need to preserve generalisation, and constitutes a general tool for adapting neural network controllers on-line.

1. INTRODUCTION

The chemical and metallurgical industries face constant demands for greater economic return that requires increased production and greater product purity. Also, environmental concerns call for the use of minimal resources. By addressing these issues, intelligent control techniques add economic value to process plants.

A process' operating point (i.e., the process state) determines the product purity and production rate. The operating point thus has an intrinsic economic value. Control engineers select fixed operating points (i.e., set points) based on their economic value. Process changes, due to process disturbances and drifting dynamics, cause deviations from the set points, requiring corrective action. Optimal set points and effective corrective actions yield greater economic return. Typically, linear controllers (e.g., PID controllers) maintain the set points and provide corrective action to process changes (Seborg et al., 1989).

For example, a chemical reactor has an optimal operating temperature. This temperature determines the production rate that directly impacts on the economic return from the reactor. The control engineer selects this optimal temperature as a set point. A PID controller responds to process disturbances that affect the reactor temperature, by increasing or decreasing the cooling water flow rate, thereby maintaining the set point

PID controllers typically utilise both set points and fixed controller parameters. The PID controller parameters govern the corrective action (i.e., the control response) to process changes. There are three PID controller parameters: gain, integral and derivative. PID control's linear control structure is the industry standard, though not suited to non-linear processes.

Non-linear processes are common in the process industries. In such cases, PID controller parameters are optimal only over a limited operating region. Process changes may cause the operating point to stray far from the set point, whereupon PID controllers may implement sub-optimal corrective actions. Sub-optimal performance may be avoided only by adapting the controller parameters. As the set points largely determine the economic return, the set points must also adapt in response to process changes. Tracking the economic optimum therefore requires adapting both the controller parameters and the set points (Hrycej, 1997).

Effective generalisation and adaptability during process changes are essential to tracking a process' economic optimum. Generalisation tools, such as neural networks, are invaluable in creating non-linear controllers for non-linear processes. Non-linear controllers are near optimal over wider operating regions than possible with PID control (Conradie, 2000). Near optimal performance may be further improved by on-line adaptation of the neural network weights in response to process changes. Robust search techniques are required for effective on-line adaptation of neurocontroller weights.

This paper introduces an adaptive neurocontrol strategy, Adaptive Neural Swarming (ANS). A highly non-linear bioreactor benchmark is used in the control simulation. The bioreactor's dynamic behaviour is

changed continuously, which shifts the operating point with maximum economic return. ANS adapts an existing neurocontroller's weights to reap greater economic return from the changing bioreactor process. ANS emerges as an effective tool for adapting existing neural network strategies, resulting in enhanced performance.

Section 2 outlines basic notions in conventional adaptive control, which remain relevant to an advanced scheme such as ANS. Section 3 describes Adaptive Neural Swarming (ANS). Section 4 outlines the bioreactor case study. The paper concludes with an explanation of ANS' mechanism.

2. ADAPTIVE CONTROL METHODS

Control design requires a dynamic process model. Optimal control design is possible only if the process model is accurate. However, the model and the actual process are invariably mismatched. Also, exact knowledge of possible process changes is seldom available for control design purposes. Despite these shortcomings, robust control remains a control requirement. Generalisation is the ability of a controller to deliver near optimal performance, despite limited process knowledge during its design.

Generalisation may provide robust control, but optimal control is rarely ensured during the control design process. The designed controller frequently requires on-line refinements to the controller parameters and set points. Improved generalisation is difficult to impart on-line, as it involves reconciling past (i.e., design) and current process information into a single control strategy. For example, catalyst decay may cause the optimal temperature of a reactor to change over time. In contrast, adaptation changes controller parameters giving precedence to on-line process information. However, degraded performance may result should past process conditions return. A balance must thus be maintained between retaining generalisation imparted during design, while allowing adaptation to exploit changes in the process conditions (Hrycej, 1997).

On-line process information contains inaccuracies due to sensor noise and short-lived disturbances. Adapting controller parameters based on imperfect process information involves operational risk. The process may become unstable. On-line adaptation to control parameters faces numerous challenges: (1) Balancing the use of past and present process information, (2) Supervising process stability, (3) Implementing emergency procedures should the process become unsafe, due to on-line adaptation (Hrycej, 1997).

The following two sub-sections illustrate the aims of conventional methods for adapting controller parameters (section 2.1) and process set points (section 2.2). ANS has the same aims, though its methodology is dissimilar.

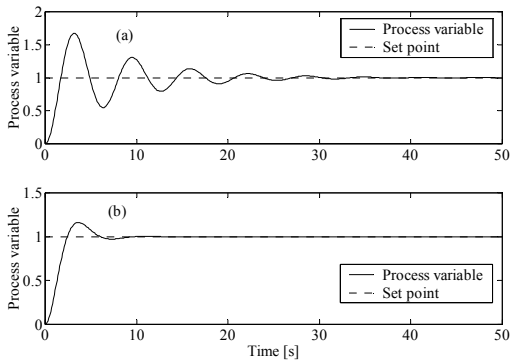

Figure 1: Objective of linear adaptive control. An oscillatory control response around the set point (a) is changed to a specified control response (b). The specified response settles sooner on the set point.

2.1 CONVENTIONAL ADAPTIVE CONTROL

An adaptive linear controller maintains a specified control response (i.e., corrective action) around a set point during process changes. For non-linear processes, a set of PID controller parameters can only maintain the specified control response for a limited range of process conditions. Process changes in non-linear processes may cause the control response to become oscillatory around the set point, as illustrated in figure 1a. Adaptive linear control tunes the PID controller parameters, which corrects the oscillatory response in figure 1a to the specified response in figure 1b. Conventional adaptive control relies on on-line process modelling (i.e., Model Reference Adaptive Control) and heuristic methods (i.e., Ziegler-Nichols) for adapting controller parameters (Ghanadan, 1990). ANS must also ensure that a specified control response is maintained.

2.2 EVOLUTIONARY OPERATION

Adaptive control does not change the set points that largely determine the economic return. Set points are selected during design based on an optimisation of dynamic model equations. The optimisation considers both economic return and controllability. However, process changes during operation may make the current set points economically sub-optimal.

Evolutionary operation (EVOP) challenges the use of constant set points in a continuously changing process. EVOP monitors the process and improves operation by changing the set points towards the economic optimum. EVOP makes a number of small set point changes that do not disrupt production. However, the set point changes need to be sufficiently large to discover potential improvements in the operating point. EVOP uses an experimental design to determine the number of set point change experiments. Pattern search methods use the experimental results to determine whether and in which direction the set points should be changed (Walters, 1991).

Consider figure 2, which graphs the economic return of

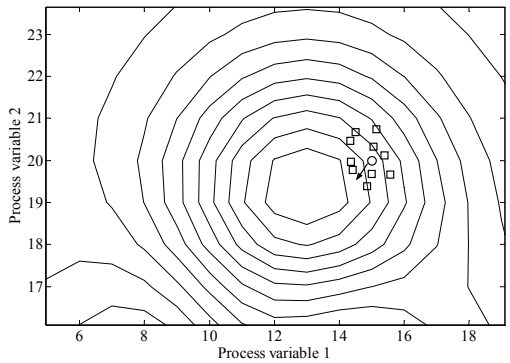

Figure 2: EVOP for a process with two process variables. The current set point (circular marker) is moved along the arrow's trajectory based on the economic return of each set point experiment (square markers). The process operation is thus improved.

a process that has two process variables. The contour lines represent operating points with similar economic returns. The circular marker represents the current set point, which is economically sub-optimal. The set points for both process variables should be reduced for optimal economic return. EVOP conducts a number of set point change experiments (represented by square markers) in the neighbourhood of the current set point. The economic return for each set point experiment is determined. In figure 2, three experiments have greater economic return than the current set point. EVOP adjusts the current set point in the direction of greater economic return. The process is repeated until optimal set points are found (Walters, 1991).

EVOP does not adapt the PID controller parameters for each of the set point experiments. As discussed in section 2.1, using the same controller parameters for all the set point experiments may give oscillatory responses. Poor control responses impact negatively the accurate determination of economic returns.

Adaptive control and EVOP may be combined in a two-step methodology to track a changing economic optimum. EVOP selects a number of set point experiments. An adaptive control method establishes a specified control response for each set point experiment. The economic evaluations for each experiment will consequently be comparable, whereupon EVOP adjusts the current set point. This cumbersome two-step process is repeated until the optimal set point is found. Ideally, a single on-line experiment (evaluation) should provide information on both the economic return and the control response.

3. ADAPTIVE NEURAL SWARMING

This section describes Adaptive Neural Swarming (ANS), which combines adaptive control and EVOP into a single comprehensive step. In ANS, both the economic return and the control response are combined into a single feedback signal. A local PSO uses this sparse reinforcement information to adapt the weights of existing neural network controllers towards greater economic return in response to a changing process.

3.1 NEURAL NETWORK STRUCTURES

Neurocontrollers may originate from various sources. Neural networks may be trained to mimic the control actions of existing PID controllers, thereby distributing the PID functionality over several neurons. Existing fuzzy logic systems may be converted to equivalent neural network architectures (Jong & Sun, 1993). Neurocontrollers are also developed utilising evolutionary reinforcement learning techniques (Conradie et al., 2000). Neural networks possess characteristics that are beneficial to an adaptive scheme, such as generalisation and graceful degradation.

Once a PID controller is adapted, the small number of control parameters prohibits effective generalisation to past process conditions. Neural network controllers are collections of neurons, with each neuron specifying the weights from the input layer (process states) to output layer (control actions). Neurocontroller parameters are the neural network weights. A neurocontroller that is equivalent to a PID controller, has additional degrees of freedom, owing to a larger number of controller parameters. During adaptation, a neural network's distributed functionality preserves greater generalisation to past process conditions. The need for effective generalisation justifies the use of neural networks.

Neural networks also exhibit graceful degradation. Graceful degradation allows small changes to the weights, without causing catastrophic control performance loss (S'euim & Clay, 1990). Process stability is preserved during adaptation.

These neural network characteristics are relied upon in a reinforcement learning framework, described below, to provide process stability and continued generalisation.

3.2 REINFORCEMENT LEARNING

Reinforcement learning (RL) automates the acquisition of on-line performance (i.e., feedback) information and the adaptation process. RL uses on-line performance evaluations to guide adaptation. RL improves controller performance without a need to specify how the control objectives should be reached (Kaelbling et al., 1996).

ANS maintains a population of possible neurocontroller solutions that serve as RL evaluations, similar to EVOP experiments. Each neurocontroller is evaluated individually over a number of sensor sample periods while interacting with a dynamic process as in figure 3. Initially, the process may be at an arbitrary operating point (state, s_t). The neurocontroller observes the current process operating point at sample, t, and selects a control action, a_t. The control action changes

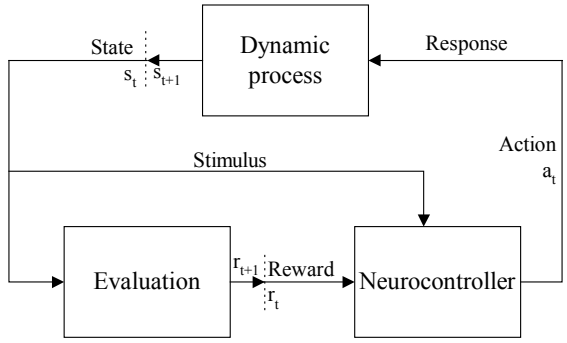

Figure 3: Reinforcement learning framework. A neurocontroller interacts with a dynamic process to learn an optimal control policy from cause-effect relationships.

the operating point to s_{t+1}. A reward, r_t, is assigned based on the economic value of this new operating point. The objective is to maximise the total reward over a series of control actions, while maintaining a specified control response. An optimisation algorithm adapts the neural network weights based the reward feedback from each evaluations.

ANS treats the population of neurocontrollers as a swarm, using a local particle swarm optimisation for adapting the weights of each neurocontroller.

3.3 PARTICLE SWARM OPTIMISATION

PSO is loosely based on the social behaviour of flocks of birds. A population of individuals is updated based on feedback evaluations, gathered from the collective experience of the swarm individuals (Shi & Eberhart, 1999). Equations 1 and 2 determine the velocity and position of the swarm in the solution space:

$$v_{id} := v_{id} + c_1 \cdot rand() \cdot (p_{id} - x_{id}) + c_2 \cdot rand() \cdot (p_{gd} - x_{id}) \quad (1)$$

$$x_{id} := x_{id} + v_{id} \quad (2)$$

where each particle, i, moves through the solution space with dimension, d. Each particles velocity vector, v_{id}, is dynamically adjusted according to the particle's own best experience, p_{id}, and that of the current best particle, p_{gd}, in the swarm. These two knowledge components are blended with each particle's current velocity vector to determine the next position of the particle as per equation 2 (Shi and Eberhart, 1999).

The best swarm particle is a beacon to a region of the solution space that may contain better optimisation solutions. Each particle searches the solution space along its unique trajectory for better solutions. Should a better solution be found, the new best swarm particle moves the swarm in a new direction. The momentum in each particle's current velocity provides some protection against convergence to a local optimum (Shi and Eberhart, 1999).

PSO has been utilised in tracking changing optima in function optimisation problems (Carlisle and Dozier, 2001; Angeline, 1997). PSO's success in these artificial domains motivates its use in complex real-world problems.

3.4 ON-LINE OPTIMISATION

ANS uses a local PSO search as the optimisation algorithm within a reinforcement learning framework. ANS thereby tracks the shifting economic optimum resulting from a changing process. Practical considerations for on-line use relate to the selection of swarm size, swarm initialisation, appropriate PSO parameters and duration of an RL evaluation.

Each on-line experiment is time and resource intensive, since no control improvements are possible during the evaluation phase. The number of reinforcement learning evaluations per PSO adaptation must therefore be minimal. However, the dimensionality of the control task constrains the minimum number of evaluations. More process information (i.e., more evaluations) is required during the evaluation phase, as the dimensionality of the control task increases. Otherwise, effective adaptation based on on-line feedback is not possible. Each neuron in a neurocontroller represents a partial solution to the control task. The number of neuron weights reflects the dimensionality of such partial solutions. For example, to effectively adapt neurons with 12 weights, an absolute minimum of 12 evaluations is required. The number of swarm neurocontrollers *(n)* is thereby selected based on the dimensionality of the control task, as reflected by the number of neuron weights.

In ANS, each swarm particle is an altered version of an existing neurocontroller. The initial swarm consists of the original (i.e., existing) neurocontroller and *(n-1)* altered neurocontrollers. Each altered neurocontroller is initialised with a small gaussian deviation from the existing neurocontroller weights. The maximum weight deviation is 3% from the each original weight, thereby altering the control policy only marginally. A neurocontroller swarm is thus initialised in a local region of the network weight solution space. This slight weight alteration determines the direction in which the swarm should move, without negatively effecting production and inducing process instability. On-line evaluation (experimentation) is thus limited to neighbouring solutions of an existing solution.

Each swarm neurocontroller is evaluated on-line for a limited number of sensor samples. A process' time constant is defined as the process response time to a step change in a control action. The process' time constant determines the number of sensor samples used in each evaluation. Equation 3 is the fitness evaluation that serves as feedback of each swarm neurocontroller's economic return:

$$Fitness = \int_{t_1}^{t_2} t \cdot P(t) \cdot dt - Penalty \quad (3)$$

where the evaluation is conducted for the number of

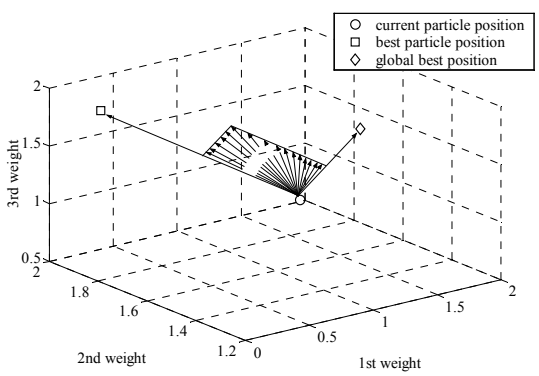

Figure 4: Possible adaptation trajectories of a weight vector based on the swarm's experience. The possible final position after adaptation lies in the plane formed by the arrow lines. The limited trajectories make the search exploitative.

samples between t_1 and t_2 and $P(t)$ is the instantaneous profit at time t.

A higher $P(t)$ for each sample reflects a higher economic return, which increases the fitness value. ANS thus searches for improved economic return. Equation 3 also dictates the specified control response. An ITAE (integral-time-absolute-error) control response has minimal oscillation, which is suited to numerous process control applications. Maximising the integral results in an ITAE control response. The fitness evaluation thus contains information regarding both the economic return and the control response. Also, should hard operating constraints exist for the process, a penalty is assigned should such operating constraints be approached during adaptation. This penalty reduces the fitness and solutions are therefore pursued only within the search boundaries.

An exploitative search preserves generalisation and reduces the risk of inducing process instability. A local (i.e., exploitative) PSO search is implemented by selecting a small inertia weight (ϖ = 0.4) and the parameters c_1 and c_2 equal to 0.5 (conventionally 2.0) in equation 1. Each neurocontroller, i, adapts each weight, x_{id}, at position d in accordance with equation 2.

A neurocontroller may move only in a limited number of trajectories based on the swarm's experience. Consider a neurocontroller comprised of one neuron with 3 weights with no initial velocity. In figure 4, the circular marker represents the current weight vector. The dashed arrow lines illustrate the possible adaptation trajectories. These trajectories are determined by the global best neurocontroller (square marker) and the neurocontroller's own best experience (diamond marker). These limited trajectories make the search exploitative and are relevant to the optimisation objectives, since the directions are determined by the swarm's collective experience (Shi and Eberhart, 1999).

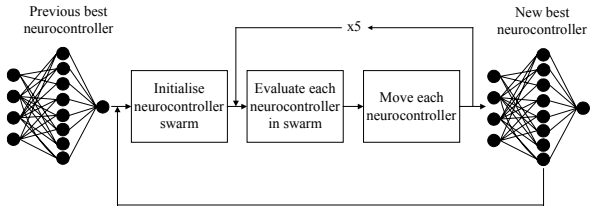

Figure 5: Adaptive neural swarming flow diagram. An effective neurocontroller is initialised into a swarm and adapted based on the evaluation of the swarm.

The local PSO search is run for five iterations, as illustrated in figure 5. The swarm is then re-initialised around the new best neurocontroller. Re-initialisation starts a new search in the neighbouring solution space of the new best neurocontroller. The search thus continues outside the space of the prior initialisations.

ANS was tested in a real-world bioreactor case study. The case study illustrates ANS' ability to adapt the neurocontroller weights towards greater economic return.

4. BIOREACTOR CASE STUDY

4.1 BIOREACTOR CONTROL PROBLEM

A bioreactor is a continuous stirred tank fermenter. It contains a biomass of micro-organisms that grow by consuming a nutrient substrate. The liquid substrate is fed continuously into the reactor, which also determines the reactor's liquid level (i.e., hold-up). The biomass is sold as the product. The bioreactor's dynamic behaviour is highly complex, non-linear and varies unpredictably. Also, the bioreactor process is difficult to quantify, due to unreliable biosensors and long sampling periods (Brengel and Seider, 1992).

Furthermore, the maximum bioreactor liquid level is a hard operating constraint. Should operation exceed the maximum level, the bioreactor is shut down and must then be restarted at great operational cost. However, the maximum instantaneous profit increases as operation approaches the hard level constraint. A trade-off between safety and the maximum economic return is required (Brengel and Seider, 1992).

The operating objective is to maximise the venture profit of the process on-line in response to process changes. This entails tracking the operating point with the maximum venture profit and ensuring acceptable control responses. The bioreactor may be simulated accurately and as such constitutes a benchmark for testing new adaptive methodologies without risking unsafe operation.

4.2 EXPERIMENTAL SET-UP

Typical process changes were simulated to mimic real-world bioreactor operation. The bioreactor's model was changed significantly by reducing the cell mass growth K (figure 6a) and increasing the substrate feed concentration S_F. The increased (i.e., off-set) S_F is also

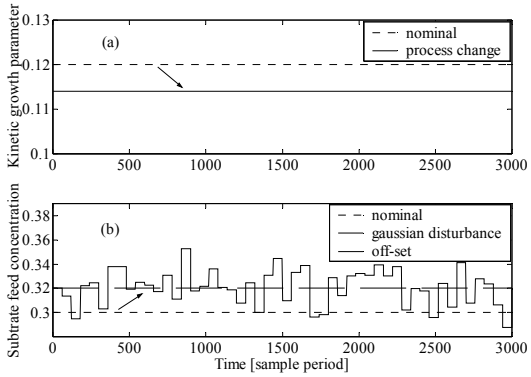

Figure 6: Process changes to the bioreactor. The arrows show the changes from the nominal process conditions (dashed line) for the growth parameter (a) and the substrate feed (b). The process is changed significantly.

disturbed with a gaussian distribution (figure 6b). In addition, the biosensors were inaccurate with a gaussian distribution around the correct reading.

Process search limits ensured that the process operation did not exceed the operation constraints. An adaptation scheme should never induce process shutdown by searching for operating points that are unsafe. The reactor level must remain below a high level alarm, which is a safety margin before bioreactor shutdown is initiated. The high level alarm was set at 5.95 [m] and bioreactor shutdown at 6.2 [m].

An optimal neurocontroller, with 8 neurons comprised of 7 weights each, was developed for the nominal process conditions using methods developed in prior work (Conradie et al, 2000). As discussed in section 3.4, ANS utilised this original neurocontroller to initialise a swarm of 10 neurocontrollers and each swarm neurocontroller was evaluated on-line over 20 sample periods. The inaccurate sensors and randomly changing process conditions make obtaining accurate feedback (i.e., evaluations) for ANS difficult. The ten evaluations, though not based on precise information, determined the direction and velocity of the neurocontroller swarm.

4.3 RESULTS

4.3.1 Adaptation efficiency of ANS

Figure 7 presents the instantaneous profit (IP) for the original neurocontroller and the ANS neurocontroller over a hundred day operating period. Figure 7 illustrates the effect of the process changes on the IP. The average instantaneous profit for the original neurocontroller was 55 [$/min]. As shown in table 1, this is well below the optimal profit of 96 [$/min] expected during design for the nominal process conditions. The original neurocontroller's IP is reduced due to sub-optimal generalisation to the process changes, though it was able to keep the process stable.

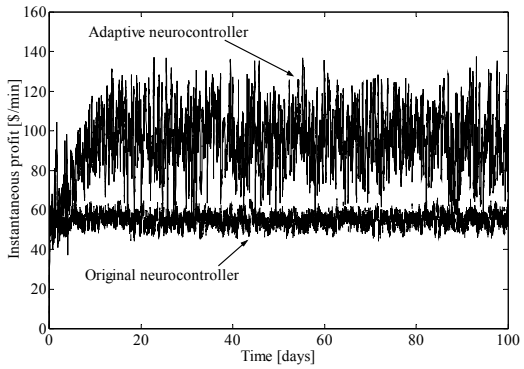

Figure 7: Instantaneous profit for the original and ANS neurocontrollers over 100 days of on-line operation. The adaptive neurocontroller garners greater economic return from the changing process than the original neurocontroller.

Table 1: Maximum IP for changing process conditions

Process condition	*Maximum profit[$/min]*
Nominal process conditions	96
K reduced, Off-set S_F	106
K reduced, Minimum S_F deviation	69
K reduced, Maximum S_F deviation	130

The original neurocontroller incurs an economic opportunity cost. Improved performance over 55 [$/min] is attainable with ANS. The average increase in S_F (i.e. off-set) presents an opportunity for greater venture profit. ANS achieves a substantially increased average profit of 94 [$/min] (figure 7), which is only slightly below the attainable 106 [$/min] possible for the increased S_F (table 1).

As seen in figure 7, the ANS neurocontroller has a larger IP standard deviation than the original neurocontroller. ANS tracks the optimal IP that is due to the gaussian disturbance in S_F. For high S_F values over extended periods (figure 6b, between samples 2000-2250), an IP of 120 [$/min] was attained, though a maximum of 130 [$/min] is attainable (table 1). For unusually low S_F values over extended periods, the swarm attained a minimal profit of 60 [$/min]. The optimal profit for this unfavourable process condition is 69 [$/min] (table 1). ANS thus approximates the changing optimal IP. A small difference remains, because S_F changes substantially over time periods that are too short for the swarm to adapt completely. The swarm is thus essentially tracking the moving average of S_F. Nevertheless, the IP for ANS control exceeds the highest IP for the original neurocontroller at all times (figure 7). ANS offers considerable benefits over the generalisation offered by the original neurocontroller.

4.3.2 Avoiding Hard Process Constraints

Figure 8 illustrates the swarm's ability to avoid the process search limits. Recall that the IP increases as the bioreactor level increases. The swarm neurocontrollers thus searched for control policies that increased the

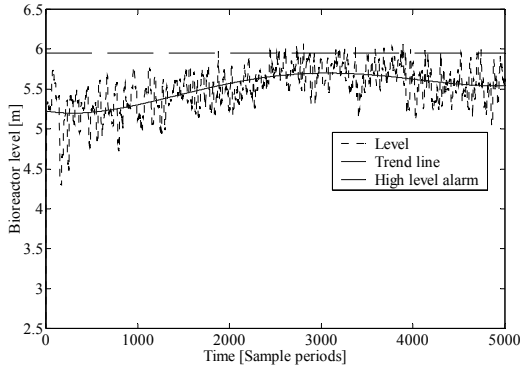

Figure 8: Avoiding the hard level constraint. The trend line (solid) illustrates the swarm moving away from high level alarm set at 5.95 [m]. Process shutdown is thereby avoided.

bioreactor level. Consequently, the swarm moved towards the high level alarm during on-line operation. The high level alarm of 5.95 [m] should never be exceeded; preserving the safety margin before bioreactor shutdown. A neurocontroller's fitness was penalised severely for exceeding the high level alarm. Such a penalised fitness was always lower than the fitness of a neurocontrol policy that remained within the search boundaries. Neurocontrollers, with a penalised fitness, no longer guided the swarm and the swarm moved away from the high level alarm. In Figure 8 at 3000 sample periods, the trend line indicates a move away from the high level alarm. Shutdown at a reactor level of 6.2 [m] was thus safely avoided in ANS' on-line search.

4.3.3 Neuron Weight Adaptations

Each neuron in a neurocontroller has a particular functionality that is a partial solution to the control task. A neuron's weight vector determines its functionality. The changes to a neuron's weight vector during adaptation, provides insight into how its functionality changed in response to the changing process conditions. Principal component analysis allows visualisation of neuron weight vectors and therefore neuron functionality.

Figure 9 is a principal component plot of the weight vector of each neuron in the swarm's current best neurocontroller. After each adaptation, all the neuron weight vectors for the best swarm neurocontroller were plotted in figure 9 as circular markers. The markers thus represent the history of adapted neuron functionalities.

In figure 9, the clusters indicate the different neuron functionalities that solve the control task. A cluster that is distributed over a larger region of the neuron weight space, had undergone a greater degree of on-line adaptation to its functionality. The extent of each neuron's adaptation is determined by the reigning process changes.

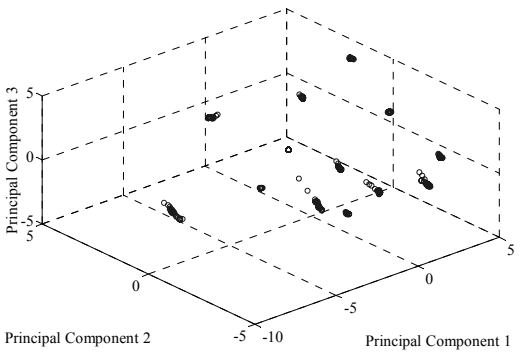

Figure 9: Principle component analysis of neuron functionality (85% variance explained). Circular markers represent neuron weight vectors. Each cluster represents the change in neuron functionality due to adaptation. The extent of each neuron's adaptation is determined by the reigning process changes.

5. DISCUSSION AND FUTURE WORK

ANS' exploitative search preserves the existing neurocontroller's generalisation. For the bioreactor, adaptation failure (i.e., shutdown) never occurs during extensive implementation. Also, instability is never induced in the control response. The bounded nature of each neuron cluster in figure 9 provides insight into how ANS preserves generalisation. Each neurocontroller retains memory of its best position (eq. 1) during the five iterations between initialisations. As the fitness landscape changes, the fitness value of a neurocontroller's best position is no longer valid. A neurocontroller's best position rather serves as an example of where previous good solutions have been found. Memory of past neurocontroller positions biases the search in the direction of good past solutions. This memory function preserves generalisation by considering both past and current process information in the search. Re-initialisation, which clears the swarm's memory, limits prolonged bias to past solutions. Without limiting memory of past solutions, a drifting optimum would be difficult to track.

ANS' search for optimal control policies in a changing process works as follows. Process changes affect each neuron's functionality differently. Some neurons consequently no longer contribute to optimal economic return. The functionality of such a neuron needs to be updated, while retaining information in its weight structure that is still valid.

Consider a neuron weight that is optimal once adapted to a fixed value, despite continued process changes. Such fixed weights correspond to process conditions that remain constant (e.g., fixed growth parameter). As described in section 3.4, the possible directions for adaptation are limited to the positional experiences of all the swarm neurocontrollers. In ANS, the swarm neurocontrollers align along such a fixed weight, preventing (as per eq. 1) the swarm from moving along that particular weight dimension. After several ANS

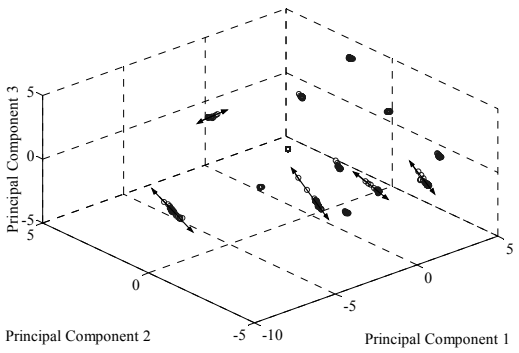

Figure 10: Arrow lines indicate the trajectories of neuron functionalities in response to common (re-occurring) process changes such as S_F. ANS implicitly takes advantage of common process changes, which facilitates effective adaptation.

iterations, only weights still relevant to improving the IP are implicitly changed. Re-occurring process changes (e.g., S_F) govern which specific neuron weights are continuously changed to track the economic return. The dimensionality of the search is thus somewhat reduced. Figure 10 is a copy of figure 9, except that adaptation trajectories are emphasised by drawing arrow lines through the clusters. Each neuron functionality (cluster) moves along a fixed trajectory in response to re-occurring process changes. ANS establishes these trajectories implicitly and exploits this swarm knowledge for greater economic return.

Future work will explicitly identify neuron functionalities that require adaptation. Such explicit knowledge may be used to further speed adaptation using fewer on-line evaluations. As ANS is a robust means for adapting neurocontrollers, it will be tested in other complex domains such as robotics and gaming.

6. CONCLUSIONS

Although neurocontrollers generalise their control actions in a changing process, such generalisation (though robust) may be economically sub-optimal. Adaptive Neural Swarming augments neurocontroller weights on-line, thereby garnering greater economic return from the changing process. ANS balances the need to adapt with the need to preserve generalisation. ANS also effectively avoids hard operating constraints during its on-line search. ANS implicitly identifies re-occurring process changes and uses this knowledge to speed adaptation. ANS is therefore a robust general tool for adapting of neural network controllers on-line. The greater economic return for the bioreactor case study suggests that the process industries would benefit significantly by implementing Adaptive Neural Swarming.

Acknowledgements

This work was supported in part by the South African Foundation for Research and Development, the Harry-Crossley Scholarship Fund, the National Science Foundation under grant IIS-0083776 and by the Texas Higher Education Coordinating Board under grant ARP-003658-476-2001.

References

Angeline, P.J., (1997). *Tracking Extrema in Dynamic Evironments*. Proceedings of the 6th Int. Conference on Evolutionary Programming, Vol. 1213: 335-345.

Brengel, D.D., and Seider, W.D., (1992). Coordinated design and control optimization of nonlinear processes. *Computers and Chemical Engineering* 16(9): 861-886.

Carlisle, A., and Dozier, G., (2000*). Adapting Particle Swarm Optimization to Dynamic Environments.* Proceedings ICAI 2000, Las Vegas, Vol. I: 429:434.

Conradie, A.v.E, (2000), *Neurocontroller development for nonlinear processes utilising evolutionary reinforcement learning*. M.S.c. thesis, Univeristy of Stellenbosch, South Africa.

Conradie, A., Nieuwoudt, I., and Aldrich, C., (2000). *Nonlinear neurocontroller development with evolutionary reinforcement learning*. 9th National Meeting of SAIChe, Secunda, South Africa.

Ghanadan, R., (1990). *Adaptive PID control of nonlinear systems*, M.Sc. thesis, University of Maryland, USA.

Hrycej, T., (1997). *Neurocontrol: Towards an industrial control methodology*. John Wiley & Sons (New York): 223-242.

Jang, J.S.R., and Sun, C.T., (1993). *Functional equivalence between radial basis function networks and fuzzy inference systems*. IEEE Transactions on Neural Networks, 4(1): 156-159.

Kaelbling, L.P., Littman, M.L., and Moore, A.W., (1996). *Reinforcement Learning: A Survey*. Journal of Artificial Intelligence Research, 4: 237-285.

Seborg, D.E., Edgar, T.F., and Mellicamp, D.A., (1989). *Process Dynamics and Cont*rol. John Wiley & Sons (New York).

S'equim, C.H., and Clay, R.D., (1990). *Fault tolerance in artificial neural networks*. Int'l Joint Conference Neural Networks (San Diego), vol. 1: 703-708.

Shi, Y., and Eberhart, R.C., (1999). *Empirical study of particle swarm optimization*. Proceedings of the 1999 Congress on Evolutionary Computation. IEEE Service Center (Piscataway, NJ): 1945-1950 .

Walters, F.H., (1991). *Sequential simplex optimization*. CRC Press (Florida): 55-60.