

Semantics Mining for Opponent Strategy Estimation

DAVID AL-DABASS, RICHARD CANT, CAROLINE LANGENSIEPEN

School of Computing and Informatics,

Nottingham Trent University

Nottingham NG11 8NS, UK.

E-mail: david.al-dabass@ntu.ac.uk.

Two approaches to winning a game are considered.

1. Opponent's strategic intentions are reduced to a parameter vector which drives a multi agent evolutionary semantic net whose output is the tactics trajectory of game steps.
 - 1.1 Differential analytical models are used in a multi stage semantics mining process to estimate the opponent's strategic intentions from his game steps trajectory.
 - 1.2 A compound 6th order recurrent semantic architecture is used to translate the strategy parameters into tactics in a Kalman like process which emulates the action of 'mirror' neurons in biological intelligence.
2. Strategic intentions are implicit in the solution of the game and Genetic Algorithms are considered for the Sudoku game. GAs are widely regarded as being relatively immune to the problems of local minima that affect many optimization schemes. However there are situations in which the population becomes too uniform in composition and the algorithm gets stuck. We discuss the use of a simulated disease mechanism to overcome this problem. The mechanism acts by reducing the fitness of individuals that are similar to the rest of the population, thereby giving a

competitive advantage to those individuals that display unusual traits. This disease concept is tested using a simple genetic algorithm example.

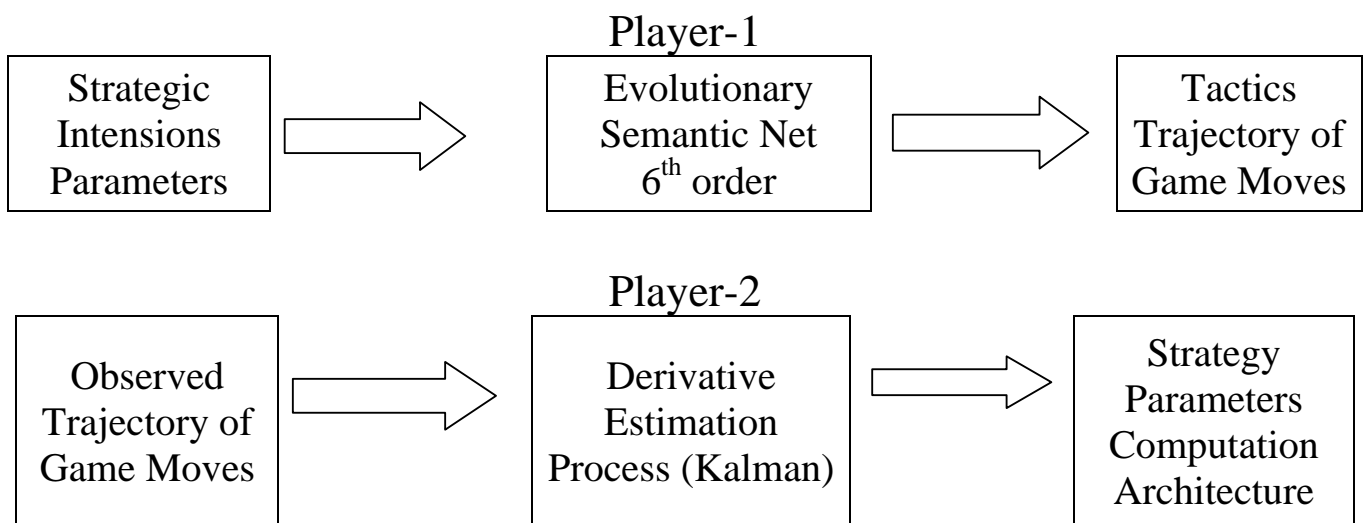
Outline of paper:

- 1. DIFFERENTIAL SEMANTIC MODELS**
- 2. HYBRID RECURRENT NETS**
- 3. STRATEGY ACQUISITION DYNAMICS:**
- 4. SIMULATION STRUCTURE FOR STRATEGY ACQUISITION**
- 5. STRATEGY PARAMETER ACQUISITION EXAMPLES**
- 6. SIMULATED DISEASE TO IMPROVE GENETIC ALGORITHMS**
- 7. TRIAL PROBLEM – SUDOKU**
- 8. THE DISEASE MECHANISM**
- 9. INITIAL RESULTS**
- 10. VARIATIONS ON THE ALGORITHM**
- 11. FURTHER RESULTS**
- 12. CONCLUSIONS**

FIRST APPROACH

Opponent's strategic intensions are reduced to:

- a parameter vector which drives
- a multi agent evolutionary semantic net
- whose output is the tactics trajectory of game steps.
- Differential analytical models are used in a multi stage semantics mining process to estimate the opponent's strategic intensions from his game steps trajectory.
- A compound 6th order recurrent semantic architecture is used to translate the strategy parameters into tactics in
- a Kalman like process which emulates
- the action of 'mirror' neurons in biological intelligence.



DIFFERENTIAL SEMANTIC MODELS

An agent (game player) may show a temporal behaviour even when the input parameters to the strategy semantic model are constant, figure 1.

The causal parameters themselves may be the output of other nodes, which may either be recurrent nodes or static nodes,- the latter may be logical or arithmetic.

To model this oscillatory behaviour a second order integral hybrid model is proposed, figure 2.

This model is based on the well known second order dynamical system which has the following form:

$$\omega^{-2} x'' + 2. \zeta. \omega^{-1}. x' + x = u$$

Where x is the trajectory of game steps (tactics),

ω , ζ and u are the natural frequency, damping ratio and input respectively of the strategy semantic model and represent the 3 causal parameters that form the input.

To configure this differential model as a recurrent network, a twin integral elements are used to form a hybrid integral-recurrent net as shown in Figure 2-a.

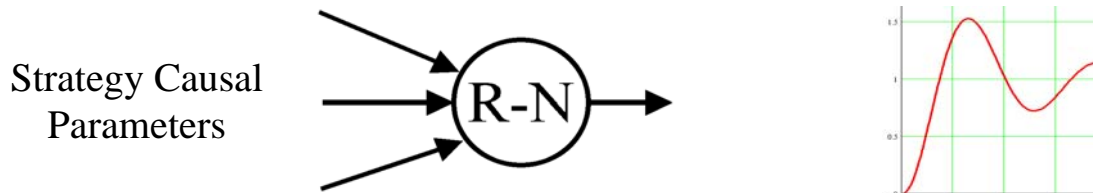


Figure 1. A Recurrent strategy semantic node (SSN) exhibits a temporal behaviour at the output despite having constant causal parameters.

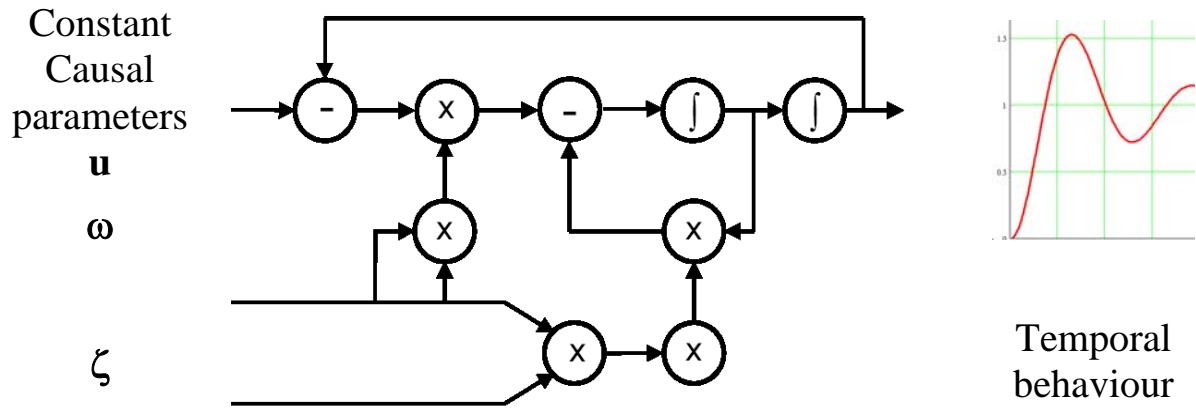


Figure 2-a. Hybrid integral recurrent net to model the temporal behaviour of strategy semantic node in Fig. 1

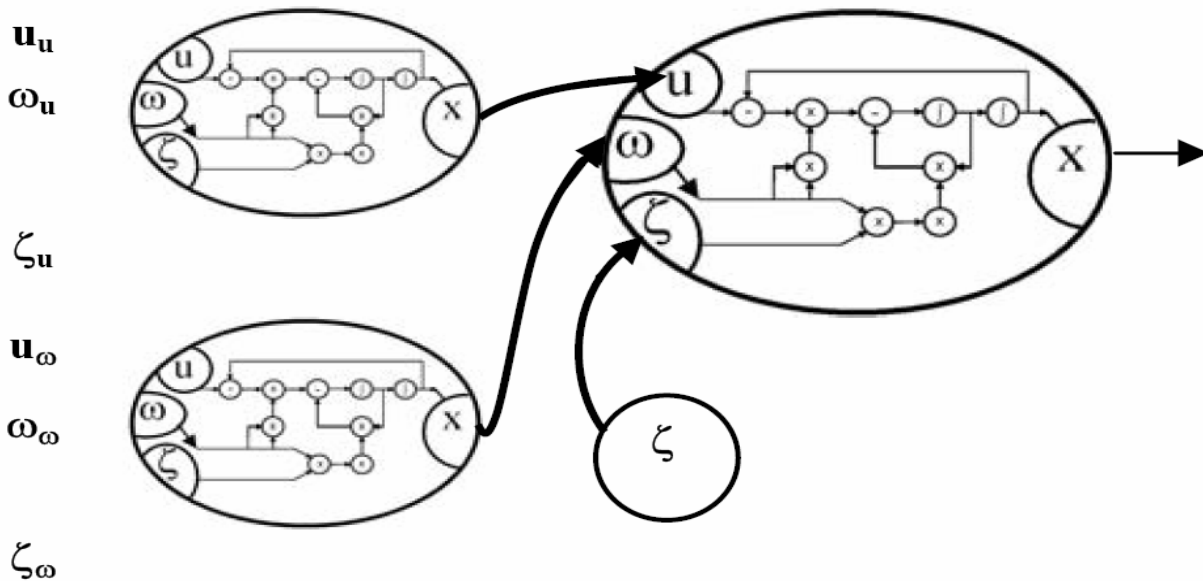
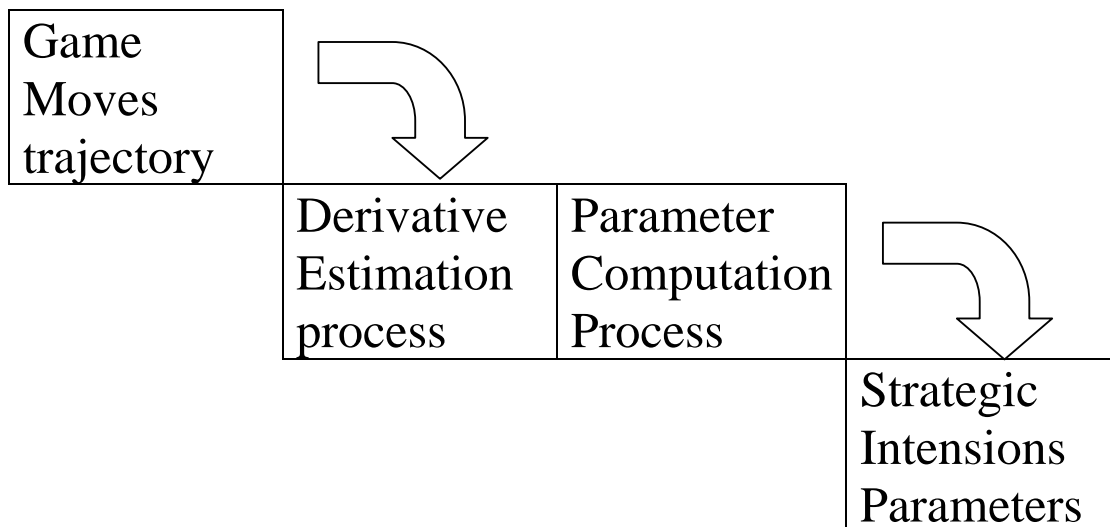


Figure 2-b. Two of the strategy parameters of semantic layer i have 2nd order dynamics.

Strategy Acquisition Dynamics

- Work here extends these ideas to recurrent models where some or all the strategy parameters are time varying.
- The effect is now a time dependent game-move pattern, which forms the input to a differential estimation process to determine the strategy semantics in terms of time varying causal parameters.
- These causal parameters will themselves embody knowledge (meta semantics) which may be obtained through a second level estimation process to yield 2nd level causal parameters.
- These processes consist of a differential part to estimate the higher time derivative of the game moves, followed by a non-linear algebraic part to compute the causal parameters.



$$\begin{bmatrix} N_{i+1} \\ x_{i+1}^1 \\ x_{i+1}^2 \\ Ex_{i+1} \\ Exd_{i+1} \\ Exdd_{i+1} \\ Extd_{i+1} \\ Exq_{i+1} \end{bmatrix} := \begin{bmatrix} N_i + 1000 \left[\left(\text{rnd}(n) - \frac{n}{2} \right) - N_i \right] \cdot d \\ x_1^1 + x_2^1 \cdot d \\ x_2^2 + \left(u - 2 \cdot \zeta \cdot \omega^{-1} \cdot x_2^1 - x_1^1 \right) \cdot \omega^2 \cdot d \\ Ex_i + G \left(x_1^1 + N_i - Ex_i \right) \cdot d \\ Exd_i + G1 \left[G \left(x_1^1 + N_i - Ex_i \right) - Exd_i \right] \cdot d \\ Exdd_i + G2 \left[G1 \left[G \left(x_1^1 + N_i - Ex_i \right) - Exd_i \right] - Exdd_i \right] \cdot d \\ Extd_i + G3 \left[G2 \left[G1 \left[G \left(x_1^1 + N_i - Ex_i \right) - Exd_i \right] - Exdd_i \right] - Extd_i \right] \cdot d \\ Exq_{i+1} + G4 \left[G3 \left[G2 \left[G1 \left[G \left(x_1^1 + N_i - Ex_i \right) - Exd_i \right] - Exdd_i \right] - Extd_i \right] - Exq_{i+1} \right] \cdot d \end{bmatrix}$$

Figure.3: Iterative Euler integrator for the noise source, behaviour generator and higher time derivative estimator.

- The top term represents the random number generator feeding an integrator whose output N forms the noise source.
- The 2nd and 3rd terms represent the state space terms of the 2nd order system generating the observed trajectory x_1 and its derivative x_2 .
- The 4th term shows a 1st order filter whose input is the noise corrupted trajectory x_1+N with a gain G and integration step d .
- The 5th to 8th terms show the successive first order filters that generate the time derivatives of x , i.e. x' , x'' , x''' and x'''' .

Derivative Estimation and Noise

- Deliberate misleading random moves added to the trajectory undergoes successive filtering as it passes through the stages of the higher time derivatives.
- Low values of G correspond to low cut-off frequencies in the filter, which result in smoother derivative estimates.
- Figure 4 below shows 2 such cases for $G=20$ and 30 respectively, the heavier filtering effect of $G=20$ on the derivative trajectories is quite noticeable. The noise level is 10% of the nominal value of x at 1.0.

-

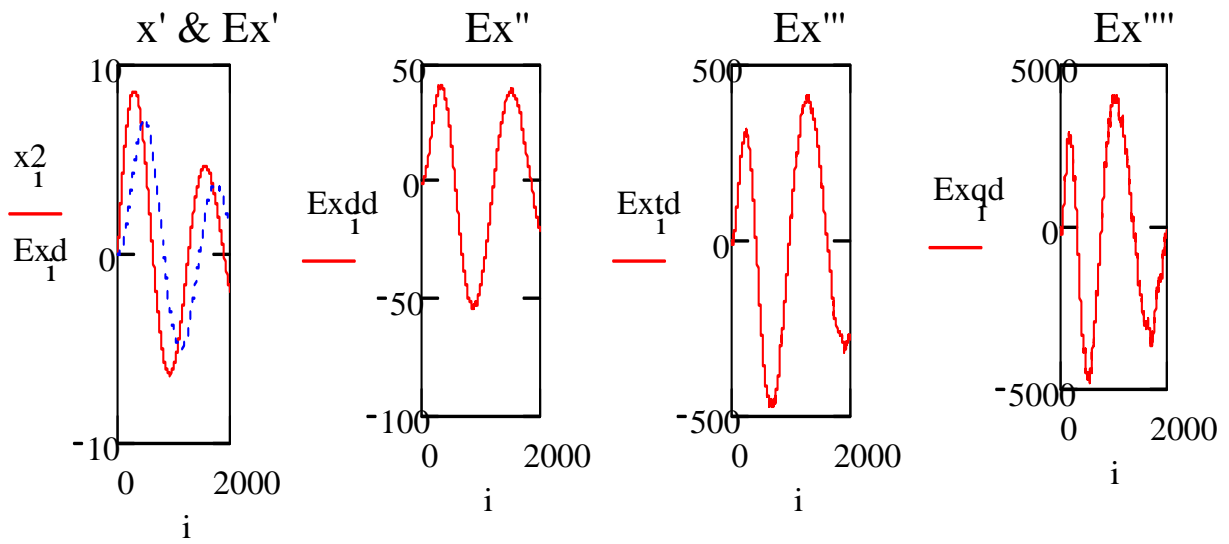


Figure 4-a: Higher derivative estimation: $n=0.1$, $G=20$.

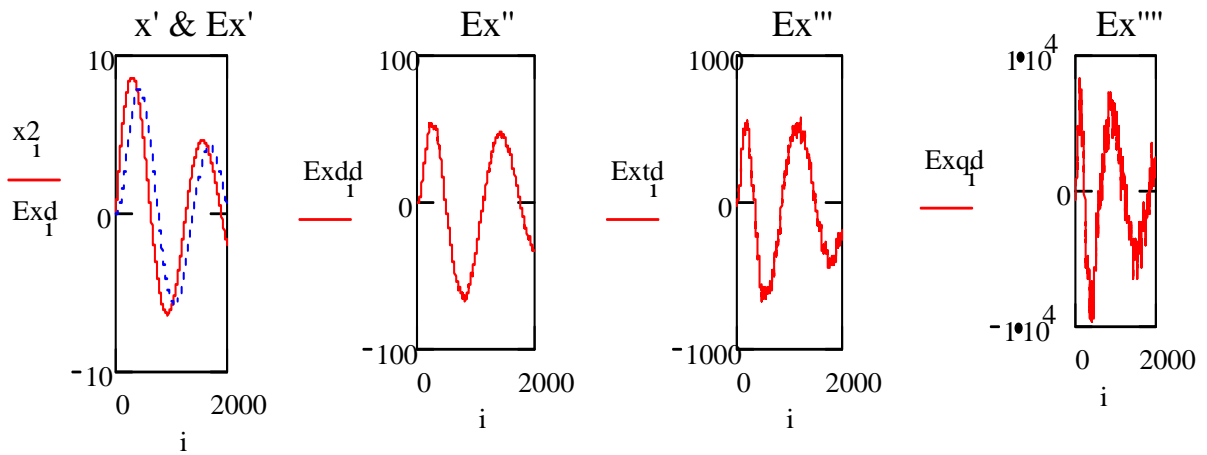


Figure 4-b: Higher derivative estimation: $n=0.1$, $G=30$

Strategy Parameter Acquisition

In Architecture-1, 3 points on the trajectory of x , x_1 and x'' were used to estimate ω followed by ζ and u using the following expressions derived in earlier work.

$$\begin{aligned}\Delta 12 &= (x_1 - x_2), & \Delta' 12 &= (x_1' - x_2'), & \Delta'' 12 &= (x_1'' - x_2'') \\ \Delta 13 &= (x_1 - x_3), & \Delta' 13 &= (x_1' - x_3'), & \Delta'' 13 &= (x_1'' - x_3'') \\ E\omega^2 &= [\Delta'' 13 \cdot \Delta' 12 - \Delta'' 12 \cdot \Delta' 13] / [\Delta 12 \cdot \Delta' 13 - \Delta 13 \cdot \Delta' 12] \\ E\zeta &= [-E\omega^{-2} \cdot \Delta'' 12 - \Delta 12] / [2 \cdot E\omega^{-1} \cdot \Delta' 12] \\ Eu &= E\omega^{-2} \cdot x_1'' + 2 \cdot E\zeta \cdot E\omega^{-1} \cdot x_1' + x_1\end{aligned}$$

- The highest time derivative used in this architecture is x'' and thus unaffected by errors in estimating x''' and x'''' .
- The gain was set to 20, and a low noise level of 0.1% was injected into the measured trajectory.
- The 3 estimated parameter trajectories are shown in Figure 5-a. After transient period, all 3 parameters converged to give good estimation accuracy with $\omega=8.9$, $\zeta=0.097$ and $u=1.004$. The lag index shift, L , clearly had a direct effect on the accuracy of estimation where it was found that best results were obtained when $L=95$.

Architectures 2 and 3 used higher time derivatives at a single point, where ω , ζ and u are given by:

$$\begin{aligned}E\omega^2 &= [x'' \cdot x'''' - x'''^2] / [x' \cdot x''' - x''^2] \\ E\zeta &= -[E\omega^{-2} x''' + x'] / [2 \cdot E\omega^{-1} \cdot x''] \\ Eu &= E\omega^{-2} \cdot x'' + 2 \cdot E\zeta \cdot E\omega^{-1} \cdot x' + x\end{aligned}$$

Although these 2 architectures use higher derivatives x''' and x'''' , they performed equally well using the heavy smoothing of $G=20$ and suitable lag compensation as shown Figure 5-b and 5-c.

0.1% Noise level

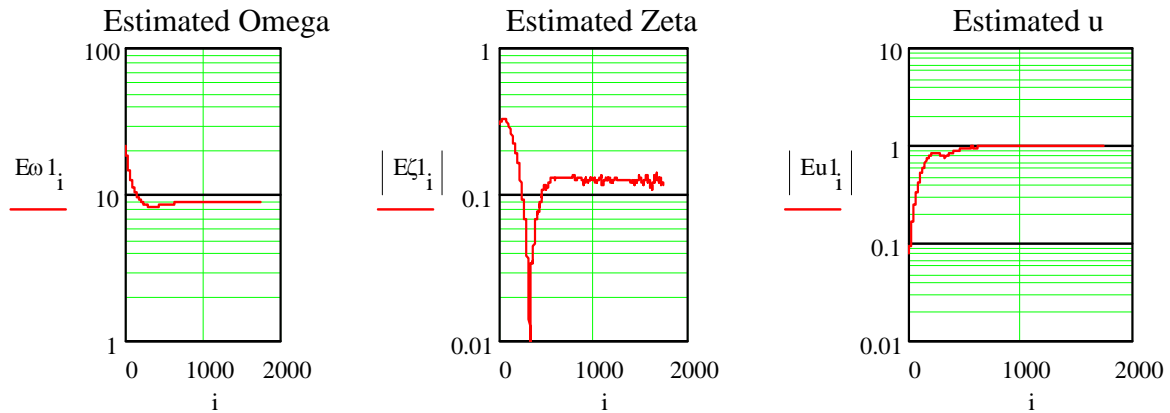


Figure 5-a: Parameter estimation of ω , ζ and u using Architecture 1, $n=0.001$, $G=20$, $L=95$.

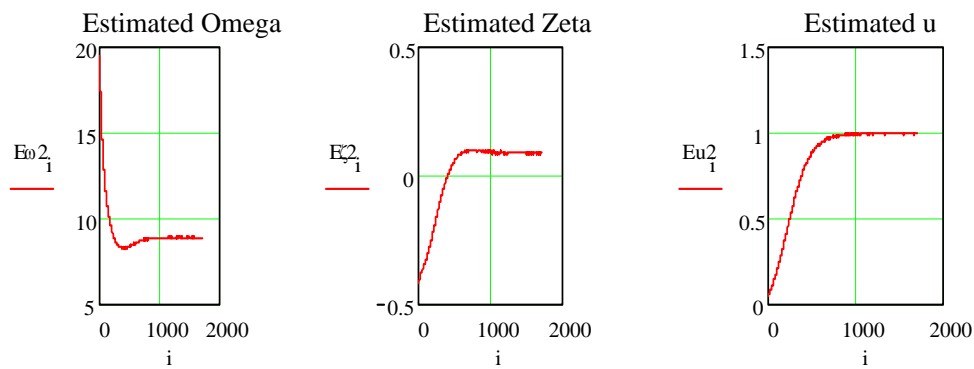


Figure 5-b: Estimated ω , ζ , u using Architecture-2, $n=0.001$, $G=20$, $L=95$.

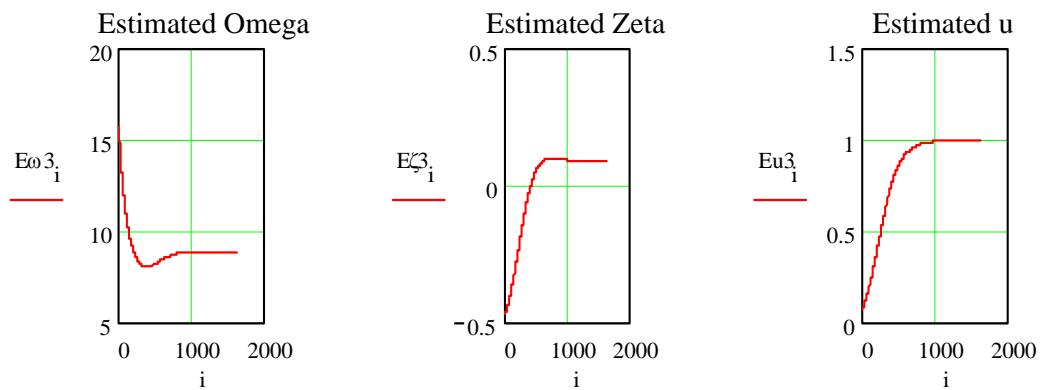


Figure 5-c: Estimated ω , ζ , u using Architecture-3, $n=0.001$, $G=20$, $L=95$.

Sensitivity to Misleading Moves

- The noise level was increased to 1% and 10% and the performance of all 3 architectures was tested as shown in Figures 6-a, 6-b, 6-c, 10, 11 and Figures 7-a, 7-b and 7-c respectively.
- It is clear that the erratic noise in the parameter trajectories reduces from Architecture-1 to 2 and from 2 to 3, i.e. architecture 3 provides the smoothest parameter estimates.

1% Noise level

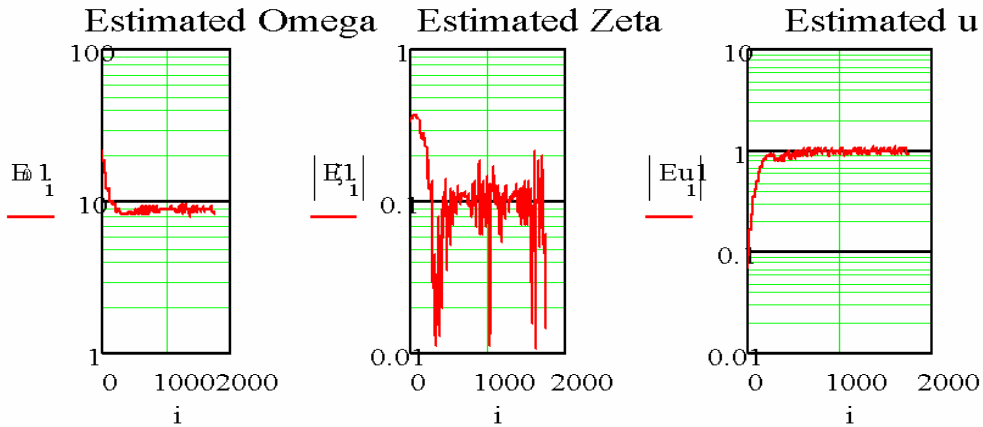


Figure 6-a: Architecture-1: Estimated ω , ζ , u using $n=0.01$, $G=20$, $L=95$

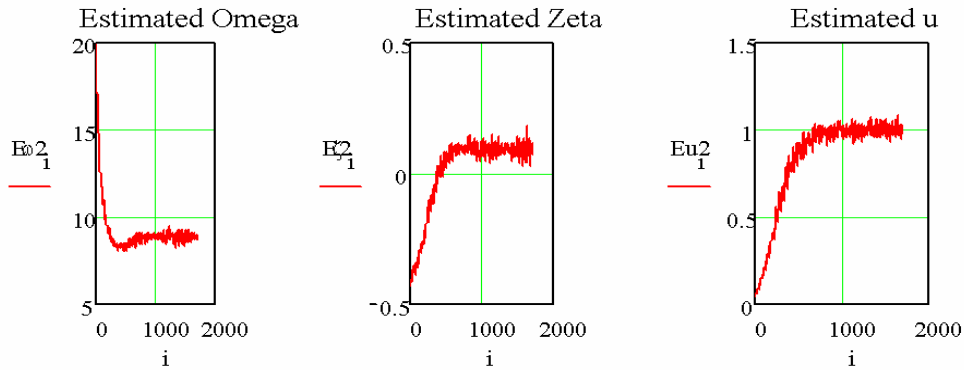


Figure 6-b: Architecture-2: Estimated ω , ζ , u using $n=0.01$, $G=20$, $L=95$.

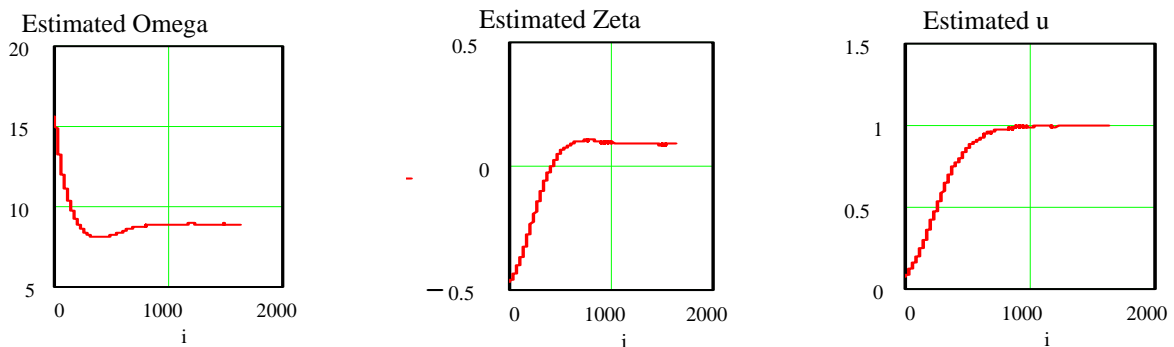


Figure 6-c: Architecture-3: Parameter estimation of ω , ζ and u using $n=0.01$, $G=20$ and $L=95$

10% Noise level

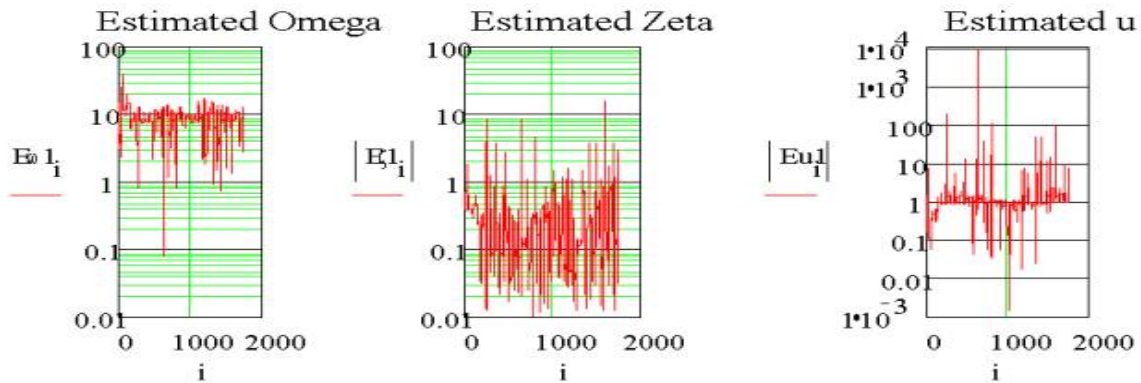


Figure 7-a: Architecture-1: Estimated ω , ζ , u using Algorithm-1, $n=0.1$, $G=20$, $L=95$.

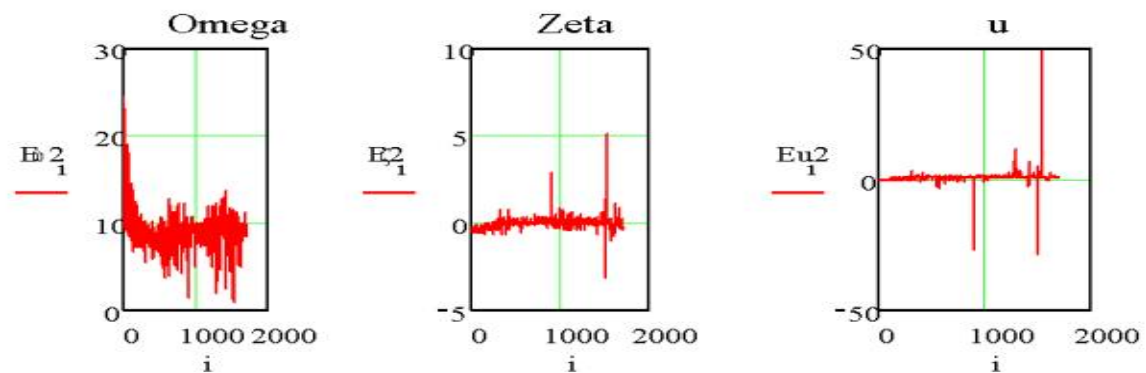


Figure 7-b: Architecture-2: Estimated ω , ζ , u using Algorithm-2, $n=0.1$, $G=20$, $L=95$.

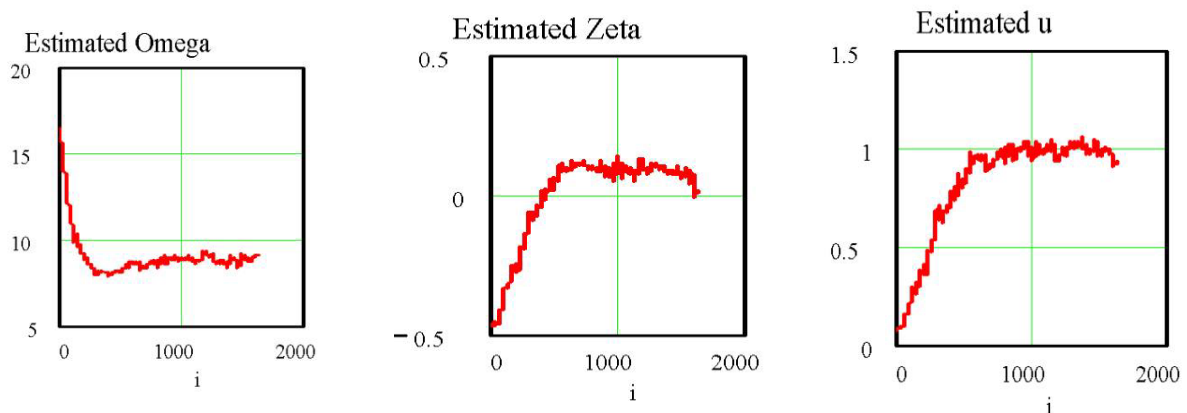


Figure 7-c: Architecture-3: Parameter estimation of ω , ζ and u using Architecture 3, $n=0.1$, $G=20$ and $L=95$

**SIMULATED DISEASE TO
IMPROVE GENETIC
ALGORITHMS**

**GO TO POWERPOINT
PRESENTATION**