5.1 SYSTEM IDENTIFICATION BASED ON GENETIC ALGORITHMS

5.1.1 Introduction

A model is a very useful and comprehensive way to summarise the knowledge about a process. Models can be obtained from prior knowledge, in terms of physical laws, or by experimentation on the process. Models based on physical laws are very difficult and time consuming to obtain. Obtaining the model by experimentation based on observing the input-output relationships is called System Identification (Mendel 1973).

Figure 5.1 shows the block diagram representation of the System Identification problem. The procedure for carrying out the System Identification of a process is as follows:

i Specify and parameterise a class of models that represent the system to be identified.
ii Apply an appropriate chosen test signal to the system and record the input-output data.
iii Perform parameter identification to select the specified class that best fits the statistical data.
iv Perform the validation test to see if the model chosen adequately represents the system.
v If the validation test is passed, the procedure ends. Otherwise steps (ii) through (iv) are performed until a validated model is obtained.
Figure 5.1  Block Diagram Representation of the System Identification Problem
In all estimation techniques, we are seeking to minimise certain optimally defined error criterion as a means to optimally fit the model to the system data. These criteria could be the deviation of the parameter estimates from their true (actual) values known as parameter error, the difference between the output of the system and that of the model in response to the same input known as output error.

Identification of a process can be either off-line or on-line. In off-line identification, the input-output data is recorded and the parameters are then estimated based on the entire data. In on-line identification, the parameters are recursively estimated for every data set so that the new data set is used to correct and update the existing estimates. Recursive Least Squares (RLS) is one of the most popular identification methods of system parameters (Sinha et al 1983).

In this section, parameter estimation of a linear system and three typical non-linear systems is discussed. These are:

1. A Minimum Phase (Linear) System
2. Non-Linear Gain
3. Non-Linear Tank Level System
4. Fed Batch Fermentor

### 5.1.2 Recursive Least Squares Technique (RLS)

RLS is one of the most popular identification methods used for estimation of system parameters.

A multivariable discrete data system can be represented by

\[
X(k+1) = A \cdot X(k) + B \cdot U(k) + v(k) \\
S \cdot Y(k) + v(k) \quad (5.1)
\]
Where

\[
Y(k) = \begin{bmatrix} X(k) \\ U(k) \end{bmatrix}
\]

\[S = [A : B]\]

\(A\) is \(n \times n\) state transition matrix

\(B\) is \(n \times r\) control input matrix

\(X(k)\) is \(n \times 1\) output state matrix

\(U(k)\) is \(r \times 1\) input state matrix

\(v(k)\) is noise vector in measurement

\(k\) is the sampling instant

Let \(N = n + r\), then the order of \(S\) and \(Y(k)\) will become \(n \times N\) and \(N \times 1\) respectively.

The algorithmic steps for identification using RLS is given below:

\[
P(k+1) = \left[ P(k) - \frac{P(k) y(k) y^T(k) P(k)}{1 + y^T(k) P(k) y(k)} \right] \quad (5.2)
\]

\[
\hat{X}(k+1) = S(k) y(k) \quad (5.3)
\]

\[
e(k+1) = X(k+1) - \hat{X}(k+1) \quad (5.4)
\]

\[
S(k+1) = \left[ S(k) + \frac{e(k+1) y^T(k) P(k)}{1 + y^T(k) P(k) y(k)} \right] \quad (5.5)
\]

The initialisation is done as follows:

\[
P(0) = \alpha I_n
\]

\[
S(0) = \frac{1}{\alpha}, \text{ where } \alpha \text{ is a large number, say 1000.}
\]
5.1.3 System Identification Using Genetic Algorithm

Consider a system described by an ARMA model

\[ A(z') y(t) = B(z^d) u(t-\theta) + C(z') v(t) \quad (5.6) \]

Where \( A(z') \), \( B(z') \) and \( C(z') \) are polynomials in the backward shift operator \( z'^1 \) and \( y(t) \), \( u(t) \) and \( v(t) \) are the output, input and noise respectively at time \( t \) and \( \theta \) is the dead time. The noise is a normally distributed random sequence with zero mean and unit variance. The polynomials \( A(z') \) and \( C(z') \) are assumed to be monic. The objective is to estimate \( A(z') \), \( B(z') \) and delay \( d \), given the input \( u(t) \) and output \( y(t) \). A sequence \( \xi(t) \) can be defined, for how well the estimates fit the system, as

\[ \xi(t) = y(t) - \hat{y}(t) \quad (5.7) \]

with

\[ \hat{A}(z') \hat{y}(t) = \hat{B}(z^d) u(t-\hat{\theta}) \quad (5.8) \]

Where \( \hat{y}(t) \) is the output of the estimated system driven by the actual input \( u(t) \), \( \hat{A}(z') \) and \( \hat{B}(z^d) \) are the estimates of \( A(z') \) and \( B(z^d) \) respectively, found by minimising \( E[\xi^2(t)] \). The fitness function to be maximised is chosen as

\[ F(t) = \sum_{i=0}^{w} \frac{1}{[1 + \xi^2(t-i)]]} \quad (5.9) \]

Where \( w \) is the window size or the number of time steps the fitness is accumulated. Selection of window size is very important, because as the window size increases, the parameter estimates became better but the execution becomes slower. The GA does not have to wait for new input-output data before giving new estimates. It can iterate many times for each sample.

The parameters are coded and concatenated to form a string. The input signal has to be properly chosen and should be persistently exiting in the window for better convergence. The GA is then run till a terminating condition is reached.
Here, for reproduction, stochastic remainder selection scheme without replacement, as explained in Chapter 4, is used. Multi-point crossover scheme with the number of crossover points equal to the number of parameters is used and Elitist strategy is used for keeping the best string of the population unaffected by crossover and mutation.

The genetic parameters used are

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population size (N)</td>
<td>100</td>
</tr>
<tr>
<td>Crossover rate p_c</td>
<td>0.6</td>
</tr>
<tr>
<td>Mutation rate p_m</td>
<td>0.01</td>
</tr>
</tbody>
</table>

5.1.4 A Minimum Phase (Linear) System

A minimum phase system (Astrom et al, 1990) chosen for identification is given by

$$A(z^{-1}) y(t) = B(z^{-1}) u(t-\theta) \quad (5.10)$$

Where

$$A(z^{-1}) = 1.0 - 1.5z^{-1} - 0.7z^2$$

$$B(z^{-1}) = 1.0 (1.0 + 0.5z^{-1})$$

$$\theta = 1$$

5.1.5 Non-linear Gain

In this case, Hammerstein model is used to represent the non-linear system. Hammerstein model structure is shown in Figure 5.2. A non-linear gain is approximated by a power polynomial of order ‘n’ (Hsia, 1977)

$$x(k) = a_1 u(k) + a_2 u^2(k) + \ldots + a_n u^n(k) \quad (5.11)$$
The coefficients $a_i$ and order $n$ can be appropriately selected to approximate a given non-linear gain. The linear system is assumed to be stable. The identification problem is to estimate the coefficients $a_i$.

5.1.6 Tank Level System

The system is shown in Figure 5.3, which consists of a tank of uniform cross sectional area ‘$A$’, and valve flow resistance ‘$R$’. We now assume that outflow rate ‘$q_o(t)$’ follows the square root relationship with the level of the tank ‘$h$’.

$$q_o(t) = \alpha \sqrt{h} \quad (5.12)$$

where $\alpha$ is a constant.

For a liquid of inflow rate $q$ and constant density, material balance equation is

$$q(t) - q_o(t) = A \frac{dh}{dt} \quad (5.13)$$

substituting Equation (5.12) in Equation (5.13) gives,

$$q(t) - \alpha \sqrt{h} = A \frac{dh}{dt} \quad (5.14)$$

This equation is non-linear due to the presence of non-linear term $\sqrt{h}$. The identification problem is to estimate $A$ and $\alpha$ from the measured sequence $\{ q(k), h(k) \}$.

5.1.7 Fed Batch Fermentor

Bioprocesses (Bailey et al 1986) are very complex, non-linear and time varying systems. It is difficult to develop kinetic model for such complex bioreactions (McMillan 1987). Bajpai and Reuss have developed a mathematical model of a fed batch fermentor for producing penicillin. They proposed a simple model which is consistent with the observed functional dependencies of the specific rates of growth, glucose uptake and penicillin
Figure 5.2  Hammerstein Model Structure

Figure 5.3  Tank Level System
formation, and then estimated the model parameters to fit the experimental data (Nyttle et al 1993 a & b). The model equations for the fed batch fermentor is given by:

\[
\frac{dX}{dt} = \mu(X,S)X - \frac{(X/(S_f V))U}{V} \tag{5.15}
\]

\[
\frac{dP}{dt} = \rho(S)X - K_d P - \frac{(P/S_f V)}{V} \tag{5.16}
\]

\[
\frac{dS}{dt} = \mu(X,S)X/Y_{xs} - \frac{(S_f)(1-S/V)}{V} \tag{5.17}
\]

\[
\frac{dV}{dt} = U/S_f \tag{5.18}
\]

where \( \mu(X,S) = \mu_m S/(K_x+S) \), \( \rho(S) = \rho_m S/(K_p+S(1+S/K_i)) \) and \( \gamma(S) = m_s/(K_m+S) \). \( X \), \( P \) and \( S \) represent the concentration of biomass, product and the substrate. \( U, S_f \) and \( V \) represent the feed flow rate, feed substrate concentration and reactor volume. Identification problem is to estimate the parameters \( \mu_m, \rho_m, m_s, K_x, K_p, K_m, K_i, K_d, Y_{xs}, Y_{ps} \) and \( S_f \) from the measured values of \( X, P, S \) and \( V \).

5.1.8 Results and Discussion

This section presents the results obtained for the parameter estimation of minimum phase system, non-linear gain, tank level system and fed batch fermentor. For the minimum phase system, non-linear gain and tank level system, parameters are estimated using both RLS and GA. For the fed batch fermentor, parameters are estimated using GA only as RLS cannot be applied for identification of this system.
5.1.8.1 Minimum Phase System

The minimum phase system given in Equation (5.10) can be written, after rearranging as

\[ y(k) = ay(k-1) + by(k-2) + cu(k-\theta) + du(k-\theta-1) \quad (5.19) \]

The true (actual) values of the parameters \(a, b, c, d\) are 1.5, -0.7, 1.0, 0.5 respectively. The delay \(\theta\) is coded as a two bit string to give four choices from 1 to 4. All other parameters are assumed to be in the range \([-2, 2]\). Choosing the resolution to be slightly more than 0.04, requires six bits for each parameter. Hence, the total string length is 26.

A PRBS (pseudo random binary sequence) input with period 127 and bit interval equal to four times the sampling signal is used as the test signal. The GA is run for 600 generations, going through three generations for each input-output data, so that each run is 200 samples. The window size is chosen as 30. Hence, the fitness function is calculated for the current input-output and the previous 30 samples.

First, it is assumed that the delay is known and only the remaining parameters are identified. During the run, at the 300th generation, the parameter \(a\) is changed to 1.3. The results are shown in Figure 5.4. RLS identification for the same system is also shown in Figure 5.4.

The delay parameter \(\theta\) is then estimated with the other parameters using GA and the results are shown in Figure 5.5. The delay parameter cannot be identified directly using RLS, because it can be used only for linear in parameter systems. The input-output data that is used for identification is given in Figure 5.6. The identification results are shown in Table 5.1. It can be seen from the table that there is some bias in the parameter estimates for GA, which is due to the quantization error. This can be reduced by using more number of bits for each parameter, but in that case the convergence will be slower.
Figure 5.4 Identification of Minimum Phase System
Figure 5.5. Identification of the Minimum Phase System including delay parameter by GA

Figure 5.6 Input-Output data for the Minimum Phase System
5.1.8.2 Non-Linear Gain

The following system structure is used to represent the Nonlinear gain:

\[ x(k) = \begin{cases} 
0.6 & \text{when } u(k) > 1 \\
u(k) & \text{when } 0.6 \geq u(k) \geq -0.6 \\
-0.6 & \text{when } u(k) < -0.6 
\end{cases} \quad (5.20) \]

The Nonlinear function is approximated by the fourth-order polynomial.

\[ x(k) = a_1 u(k) + a_2 u^2(k) + a_3 u^3(k) + a_4 u^4(k) \quad (5.21) \]

The parameters \(a_1, a_2, a_3,\) and \(a_4\) are to be identified. The Nonlinear gain response for random input is shown in Figure 5.7. Since Equation (5.21) is linear-in-parameter form, RLS can be used to identify the parameters.

To apply GA to the above system, all the parameters are assumed to lie in the range from -2 to 2 and each parameter is encoded to form a 12 bit string. A window size of 40 is used for identification using GA. The algorithm is run for 3 generations for each input and output pair and the parameter variations are shown in Figure 5.8 and the results are given in Table 5.2.

Since the parameters have no exact value to compare with, a plot of estimated Nonlinear gain is made in Fig. 5.9 for comparison with the ideal saturation function. It is seen that until 20 instants, GA estimation gives maximum error of 0.3 (when compared with ideal output of 0.6), while RLS estimation gives maximum error of 0.1. At the linear region, GA estimation is better than RLS estimation and after 35th instant, RLS estimation gives larger errors compared with GA. If a larger range of approximation of Nonlinear gain is desired, then we need to use a higher order polynomial of \(u(k)\).
Figure 5.7  Response of Non-linear Gain to Random Input

Figure 5.8  Identification of Parameters for the Non-linear Gain
### Table 5.1 Results for Minimum Phase System

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Actual Value</th>
<th>Estimated Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1.5</td>
<td>1.483871</td>
</tr>
<tr>
<td></td>
<td>(1.3)</td>
<td>(1.301587)</td>
</tr>
<tr>
<td>b</td>
<td>-0.7</td>
<td>-0.709677</td>
</tr>
<tr>
<td>c</td>
<td>1.0</td>
<td>1.096774</td>
</tr>
<tr>
<td>e</td>
<td>0.5</td>
<td>0.471613</td>
</tr>
<tr>
<td>θ</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

### Table 5.2 Results for Nonlinear Gain

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Estimated Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>0.809524</td>
</tr>
<tr>
<td>b</td>
<td>0.015873</td>
</tr>
<tr>
<td>c</td>
<td>-0.142857</td>
</tr>
<tr>
<td>d</td>
<td>-0.15873</td>
</tr>
</tbody>
</table>
5.1.8.3 Tank Level System

Model equation of tank level system is given in Equation (5.14). The dynamics of the tank is assumed as follows:

- Cross section area of the tank (A) = 2.0 m\(^2\)
- In flow rate (q) = 0.5 m\(^3\)/s
- Constant \(\alpha\) = 0.8

For simulation, sampling time (T) of 1 sec is considered and random noise of zero mean and variance 0.1 is added to the Inflow-rate (q). Now, discretizing Equation (5.14), we get

\[
h(k+1) = h(k) + \left(\frac{T}{A}\right) q(k) - \left(\frac{T \alpha}{A}\right) V_h(k)
\]

(5.22)

Since, the Equation (5.22) is in linear in parameter form, RLS can be used to identify the parameters, \(T/A\) and \(T \alpha/ A\) from which A and \(\alpha\) can be calculated. The RLS algorithm is run for 200 sampling instants. At the 100th instant tank area (A) is changed from 2.0 m\(^2\) to 2.5 m\(^2\) and \(\alpha\) is changed from 0.8 to 1.0. The parameter variations are shown in Figure 5.10 and the results are given in Table 5.3. The algorithm estimates the parameters correctly within 2 instants initially and when the parameters are changed at the 100th instant, it senses the parameter changes at the same instant and is capable of tracking the parameter variations within 50 instants.

The same parameters A and \(\alpha\) are identified using GA. For each input and output pair, the algorithm is run for 3 generations and the parameter variations are shown in Figure 5.10 and the results are given in Table 5.3 and it is seen that GA estimates the parameters within 9 generations (3 instants) with small variations. When the parameters are varied, it senses the parameter changes within 6 generations (2 instants) and tracks the parameter variations within 100 generations (33 instants). To apply GA, the parameter A is
Figure 5.9  Comparison of Response with the Ideal Saturation Function

Figure 5.10  Identification of Tank Level System
assumed to lie in the range from 1 to 5, and $\alpha$ is assumed to lie in the range from 0 to 2.

Each parameter is encoded to form a 12-bit string and window size of 40 was chosen.

5.1.8.4 Fed Batch Fermentor

The following parameter values and initial conditions are used for simulation (Nyttle et al, 1993):

The values of parameters are:

- $\mu_m = 0.11 \text{ h}^{-1}$
- $\rho_m = 0.0055 \text{ g h}^{-1}$
- $K_x = 0.006 \text{ g h}^{-1}$
- $K_p = 0.0001 \text{ g l}^{-1}$
- $K_i = 0.1 \text{ g l}^{-1}$
- $K_d = 0.01 \text{ h}^{-1}$
- $m_c = 0.029 \text{ g g}^{-1} \text{ h}^{-1}$
- $Y_{xx} = 0.47 \text{ g g}^{-1}$
- $Y_{ps} = 1.2 \text{ g g}^{-1}$
- $S_f = 500 \text{ g l}^{-1}$

The initial conditions used are:

- $X(0) = 1.5 \text{ g l}^{-1}$
- $P(0) = 0.0 \text{ g l}^{-1}$
- $S(0) = 0.0 \text{ g l}^{-1}$
- $V(0) = 71.0 \text{ l}$

$U$ is the Feed flow rate with an initial value of 20g/h, the noise term is given by,

$$U(t) = 20.0 + 0.9(\sin 2\pi t/25 + \cos 2\pi t/25) \quad (5.23)$$

Sampling time of 0.01h is used.

RLS cannot be used to estimate these parameters, since the model Equations (5.15) to (5.18) cannot be written in linear-in-parameter forms. But GA can be used to estimate these parameters.

While applying GA, the model equations are taken in the form they appear and estimated values of $dX/dt$, $dP/dt$, $dS/dt$, $dV/dt$ are calculated at each generation and the optimisation problem is to minimise the error between actual values of these derivatives and their estimated values.

Each parameter is encoded to form a 12 bit binary string and the algorithm is run for 5 generations for each input and output pair. At the 150th instant (750th generation), the parameters are changed as follows:
\[ \mu_m = 0.11 \text{ to } 0.14 \; \text{h}^{-1}, \; K_x = 0.006 \text{ to } 0.0075 \; \text{g.g}^{-1}\text{h}^{-1}; \; Y_{xs} = 0.47 \text{ to } 0.77 \; \text{g.g}^{-1}, \; \rho_m = 0.0055 \text{ to } 0.0075 \; \text{g.g}^{-1}\text{h}^{-1} \]

The algorithm is capable of tracking the parameter variations within 100
generations and parameter variations are given in Figure 5.11 a & b and the results are
given in Table 5.4.

It is seen that GA takes around 200 generations (40 instants) to estimate the
parameters and when the parameters are varied, it takes 30 generations (6 instants) to sense
the change and is capable of tracking the parameter variations within 150 generations (30
instants).

5.1.9 Conclusion

In this section GA is applied for identification of non-linear systems. It is
demonstrated that GA is able to estimate the parameters of the non-linear systems like fed
batch fermentor for which RLS cannot be used. Simulation results are shown which are
found to be satisfactory.
Figure 5.11 Identification of Fed-Batch Fermentor
### Table 5.3 Results for Tank Level System

<table>
<thead>
<tr>
<th>Actual Values</th>
<th>Converged Estimated Values (RLS)</th>
<th>Converged Estimated Values (GA)</th>
<th>Actual Values</th>
<th>Converged Estimated Values (RLS)</th>
<th>Converged Estimated Values (GA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area (A) in m²</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.0</td>
<td>2.000296</td>
<td>1.988235</td>
<td>2.5</td>
<td>2.490232</td>
<td>2.517647</td>
</tr>
<tr>
<td>Constant α</td>
<td>0.8</td>
<td>0.799785</td>
<td>1.0</td>
<td>1.059569</td>
<td>1.003922</td>
</tr>
</tbody>
</table>

### Table 5.4 Results for Fed Batch Fermentor

<table>
<thead>
<tr>
<th>Actual Values</th>
<th>Converged Estimated Values (GA)</th>
<th>Actual Values</th>
<th>Converged Estimated Values (GA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>α = 0.11 h⁻¹</td>
<td>0.110293</td>
<td>μₙ = 0.14 h⁻¹</td>
<td>0.138828</td>
</tr>
<tr>
<td>Kₙ = 0.006g.g⁻¹ h⁻¹</td>
<td>0.006022</td>
<td>Kₙ = 0.0075g.g⁻¹ h⁻¹</td>
<td>0.007389</td>
</tr>
<tr>
<td>Yₙ = 0.47 g.g⁻¹</td>
<td>0.466348</td>
<td>Yₙ = 0.77 g.g⁻¹</td>
<td>0.759177</td>
</tr>
<tr>
<td>mₙ = 0.029g.g⁻¹ h⁻¹</td>
<td>0.027790</td>
<td>mₙ = 0.029g.g⁻¹ h⁻¹</td>
<td>0.027070</td>
</tr>
<tr>
<td>kₙ = 0.1 g. l⁻¹</td>
<td>0.100986</td>
<td>kₙ = 0.1 g. l⁻¹</td>
<td>0.103592</td>
</tr>
<tr>
<td>kₙ = 0.01 h⁻¹</td>
<td>0.009731</td>
<td>kₙ = 0.01 h⁻¹</td>
<td>0.009806</td>
</tr>
<tr>
<td>Yₙ = 1.2 g.g⁻¹</td>
<td>1.180592</td>
<td>Yₙ = 1.2 g.g⁻¹</td>
<td>1.128006</td>
</tr>
<tr>
<td>ρₙ = 0.0055g.g⁻¹ h⁻¹</td>
<td>0.005499</td>
<td>ρₙ = 0.0075g.g⁻¹ h⁻¹</td>
<td>0.007467</td>
</tr>
</tbody>
</table>
5.2 GENETIC ALGORITHMS IN CONTROLLER DESIGN & TUNING

5.2.1 Introduction

When controllers are to be designed without having an accurate mathematical model of the system to be controlled, two problems arise: first, how to establish the structure of the controller, and second, how to set numerical values of the controller parameters. In solving the first problem, many techniques have proved useful, as for example, manual design of expert controllers, qualitative model-based approach and machine learning. In contrast, the second problem is often solved ad hoc. GAs can learn to control a dynamic system without any prior knowledge about the system. In this section the application of GAs to both learning controller structure from scratch and tuning controller parameters is explained.

However, when GAs are used to learn the rules, some serious problems such as incomprehensibility and reliability of learned rules, robustness, and cost of learning exists. Thus the following framework is proposed, and shown in Figure 5.12 (Varsek et al, 1993).

i. Learning control without prior knowledge
ii. Tuning the control parameters of suboptimal control rule
iii. Reducing the cost of learning by exploiting domain knowledge.

As a case study, the problem of inverted pendulum control is considered. Besides being a standard benchmark problem for classical and alternative control approaches, it also has much in common with tasks of greater practical importance, such as two-legged walking and satellite attitude control.

The control of inverted pendulum system is done by three methods:

i. Makarovic's control rule, a manual design of expert control for inverted pendulum
Figure 5.12 Genetic Algorithm in Controller Design and Tuning: An Overview
ii. Genetic controller, where the rules and numerical parameters for the control of inverted pendulum is designed from scratch, considering the system as a black box.

iii. Genetic tuned controller where the rules are extracted from Bratko’s control rule, again a manual design of expert control for inverted pendulum, in which the numerical parameters are tuned by GAs.

The performances of the above three methods are compared on a simulated system based on the time for the system to reach the setpoint, and the error. These two criteria put together are given in a fitness function.

5.2.2 Inverted Pendulum Control

5.2.2.1 Inverted Pendulum Description

The inverted pendulum system consists of a cart and a rigid pole pivoted to the cart (Figure 5.13). A dc motor drives the cart, so that it can move left and right on a one-dimensional bounded track. The state of the system is defined by values of four system variables \( x, \dot{x}, \phi \) and \( \dot{\phi} \) representing the cart position, cart velocity, pole inclination and angular velocity of the pole, respectively. Control force is applied to the system to prevent the pole from falling while keeping the cart within the specified limits.

5.2.2.2 Inverted Pendulum Dynamics

From the dynamic equations of the pendulum the wagon and the motor, we get the torque equation and transfer function as

\[
K_t(U_m-K_b\dot{\theta})/R = J_m\ddot{\theta} + B_m\dot{\theta} + F_r
\]  

\[
G_{ps} = X(s)/\phi(s) = (M_p g L - J_p \dot{\phi}^2)/M_p L s^2
\]
Figure 5.13  Inverted Pendulum System
Where the values of constants are

\[ J_p = 2.584 \times 10^{-2} \text{ Kg m}^2 \]
\[ g = 9.81 \text{ m/s}^2 \]
\[ r = 5.84 \times 10^{-3} \text{ m} \]
\[ L = 0.26 \text{ m} \]
\[ M_p = 0.38 \text{ Kg} \]
\[ J_m = 6.1 \times 10^{-6} \text{ Kg m}^2 \]
\[ K_T = 9 \times 10^{-2} \text{ Nm/A} \]
\[ R = 11 \Omega \]

From the above equations and constants, the differential equation describing the system dynamics, used to simulate the system are given below (Renganathan, 1992)

\[ x' = -4.12x - 2.42\phi + 0.825U_m \]  
(5.26)

\[ \phi' = 12.37x + 36.69\phi - 2.47U_m \]  
(5.27)

Where

\( x \) is the Displacement of the cart in m, \( \phi \) is the Pole inclination with vertical in rad and \( U_m \) is the Armature voltage to DC motor (in volts)

5.2.2.3 Inverted Pendulum Control

Inverted pendulum system is an unstable, non-linear and more than one system variable is to be controlled by a single manipulated variable. Classical methods can be applied to this problem when several assumptions hold, for example, dynamics of variables can be approximated by linear equations (i.e. pole inclination is small enough), and sufficiently accurate numerical values of system variables are available. Here, control rules
can be obtained in two essentially different ways: by learning from experience, and by qualitative reasoning.

Learning approaches treat the system to be controlled as a black box, and a program learns to control it through trials. A trial starts with the system positioned at an initial state chosen within some specified limits and lasts until failure occurs or successful control is performed for a prescribed maximal period of time. Failure indicates that the cart position or pole inclination exceeds the given boundaries. The time duration of a trial is called survival time. Learning is carried out by performing trials repeatedly until a certain success criterion is met. Typically, this criterion requires successful control within a trial to exceed a prescribed period of time.

Most learning systems consider the "bang-bang" control regime, where only two control actions can be applied, namely positive or negative force of a fixed magnitude. Initial control decisions (rules) are usually random, resulting in a very poor control performance. However, on basis of experimental evidence and experience, control decisions are evaluated and possibly changed, thus improving control quality.

Taking into account the domain knowledge, comprehensible solutions were obtained manually, by qualitative reasoning. Here the following two reference methods are reviewed.

i. A solution, distinguished by its simplicity, was derived by Makarovic. Rules are derived as a tree structure as shown in Figure 5.14. The control rules are devised for bang-bang control regime.

ii. Another solution was inferred by Bratko, the control law in its simple form is given as

\[ F = \text{sign}(p_1 x + p_2 x' + p_3 \phi + p_4) \]  \hspace{1cm} (5.28)

Where \( p_1, p_2, p_3 \), and \( p_4 \) are numerical parameters.
Figure 5.14 Makarovic Controller Rule
These control rules are also devised for bang-bang control regime, where $F$ is the fixed magnitude whose sign is given by sign of Equation (5.28).

### 5.2.3 Genetic Controller Design and Tuning

The inverted pendulum system is simulated using differential Equations (5.26) & (5.27). The values of the states of the system $x$, $x'$, $\phi$ and $\phi'$ are computed for each time instant, and the control force is determined and applied. The boundaries of legal cart and pole position values, and initial state boundaries are given by

**Legal positions**

$-x_{\text{max}} \leq x \leq x_{\text{max}}$ \quad $x_{\text{max}} = 1.0$ m

$-\phi_{\text{max}} \leq \phi \leq \phi_{\text{max}}$ \quad $\phi_{\text{max}} = 12^\circ$

**Initial state boundaries**

$0.5$ m $\leq x \leq 0.5$m

$-2^\circ \leq \phi \leq 2^\circ$

$-0.05$ m/s $\leq x' \leq 0.05$ m/s

$-1^\circ$/s $\leq \phi' \leq$ s

### 5.2.3.1 Fitness Equations

The aim of this problem is to maximise the survival time, and simultaneously, to minimise the discrepancy between the desired and actual behaviour. This criterion is embodied in a cost function, called fitness equations, used to evaluate candidate control rules during the learning process. These equations are also used as a performance index for the behaviour of the system. The fitness Equation is given by (Varsek et al 1993):
\[ f = \hat{S} (1 - \hat{\text{Err}}) \quad (5.29) \]
\[ \hat{S} = 1/N \sum_k S_k/S_{\text{max}} \quad (5.30) \]
\[ \hat{\text{Err}} = 1/N \sum_k \hat{\text{Err}_k}/S_k \quad (5.31) \]
\[ \text{Err}_k = \frac{1}{2} \sum_i \left( \frac{|x_i|}{x_{\text{max}}} + \frac{\phi_i}{\phi_{\text{max}}} \right) \quad (5.32) \]

Where

\[ \hat{S} \] is the normalised survival time
\[ \hat{\text{Err}} \] is the normalised error
\[ N \] is the number of trials performed
\[ S_k \] is the survival time in the kth trial
\[ S_{\text{max}} \] is the maximal duration of a trial and
\[ \text{Err}_k \] is the cumulative error of the kth trial

5.2.3.2 GAs Approach to Inverted Pendulum Control

The learning process is considered to be an instance of a combinatorial optimisation problem based on maximisation of survival time of a candidate solution and on minimisation of the difference between the actual and the desired behaviour. We have, therefore, a pair \((f, c)\), where \(f\) is a finite or countably infinite set of feasible solutions and \(c: f \to \mathbb{R}\) is a cost function. The task is to find an \(x \in f\), such that \(\forall y \in f: c(x) \leq c(y)\).

The fitness value is calculated in two steps:

i. Raw fitness is calculated from Equations (5.29) to (5.32)

ii. Raw fitness is scaled to prevent the early domination extraordinary individuals, and to encourage competition among near equals during the search.
Scaled fitness values are used to evaluate the population, i.e., for the parental selection process. Some initial experiments are needed to tune the GA parameters. Due to robustness of GAs with respect to parameter values, reasonable parameter settings ensuring good results and convergence are found soon. However, finding optimal GA parameters is not the objective of this thesis, and that different settings are chosen for each particular task, reflecting different nature and hardness of the problem.

5.2.3.3 Control Without Prior Knowledge

Learning inverted pendulum control rules with GA consists of three phases. First, a GA is applied to learn control rules in the form of decision tables or strings of binary digits, associating the system states with control actions. In the second phase these rules are made comprehensible. Finally, these rules are fine-tuned by tuning the numerical parameters appearing in the rules. The control voltage applied during the learning procedure is of magnitude ±10V.

5.2.3.4 Learning Control Rules “From Scratch”

During this phase, BOXES-like decision rules are considered. For each of the pole-cart variable x, x', φ and φ' the domain is partitioned into three labelled intervals in the following manner. Let y be a real variable and let thresholds be two real numbers. Then three symbolic values are introduced.

- \( y_{neg} \) all values less than \( y_{ref} \)
- \( y_{pos} \) all values greater than \( y_{ref} \)
- \( y_{zero} \) all values between \( y_{ref} \) and \( y_{ref} \)
Coding

Each decision rule is represented as a four-dimensional array. Since there are four system variables, each assuming one out of three symbolic values, there are $3^4 = 81$ array entries. Each entry represents a control action namely positive or negative force, and is coded with a single binary digit. In addition, two thresholds are required for each system variable, resulting in a total of eight numerical values appearing as rule parameters. These values are also coded as binary digits. Candidate solutions are thus represented as binary strings, each consisting of a decision table part and a numerical parameter part. This type of coding is called multi-parameter coding, since more than one parameter is coded in a single chromosome.

Due to the symmetry of the problem, i.e., all the decisions taken for one half, is suited for its other half, by complementing the decisions. Thus only 41 bits are actually used to encode a decision table and only four thresholds are needed, since for each variable $y$ we have, $y_{ref} = -y_{ref}^-$. As six bits are used to encode a threshold value, the chromosome length amounts to 65 bits.

Selection

Initially the chromosomes of a population are randomly chosen. They are selected for the next generation based on their fitness. First the raw fitness of each individual string is found by using Equations (5.29) to (5.32), by carrying out 25 trials with the maximal duration of a trial $S_{max}$ set to 5000 time steps.

The raw fitness obtained, are scaled and the selection is done based on this scaled fitness. The selection scheme used here is Roulette wheel method. Elitism of retaining a best string of the previous generation to the next generation is also done. Thus the best string of the last generation is retained.
Crossover

Then crossover is done on the selected strings by pairing them randomly depending on crossover probability. Here three-point crossover is done with crossover probability 0.05.

Mutation

After crossover, mutation is done on the strings, by complementing any bit in the population chosen at random. Usually, in GAs, the mutation probability will be very small. The mutation probability used here is 0.05.

Termination criterion

The termination criterion used here is a maximum of 50 generations. It has been noted that the population converges to an optimal solution within these generations for well-chosen GA parameters.

GA parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of generations</td>
<td>50</td>
</tr>
<tr>
<td>Population size</td>
<td>30</td>
</tr>
<tr>
<td>Chromosome length</td>
<td>65</td>
</tr>
<tr>
<td>Elitism</td>
<td>1</td>
</tr>
<tr>
<td>Crossover sites</td>
<td>3</td>
</tr>
<tr>
<td>Crossover probability</td>
<td>1</td>
</tr>
<tr>
<td>Mutation probability</td>
<td>0.05</td>
</tr>
</tbody>
</table>

Thus, it requires 37500 trials to get a sub-optimal control decision that can control the system within some accepted initial conditions. To get a control over a wide range of initial conditions, by trying with more trials, population size and generations, the cost of learning goes very high.
5.2.3.5 Obtaining Understandable Rule Structure

The decision rule structure obtained is in the form of binary strings. Although operational, these rules are unsuitable for human interpretation. Therefore, these rules are studied and converted manually to some readable form such as in the form of tree structure or in the form of if-then rules. The tree structure of a candidate control decision in bang-bang regime is given in Figure 5.15.

5.2.3.6 Tuning Rule Parameters

Here, the interpretation of symbolic values, i.e., interval labels, appearing in the decision table, found during the learning process form scratch is adjusted to maximise the control quality. For this purpose, a GA is employed again. This time, each chromosome represents four binary coded thresholds, using 10 bits per thresholds to allow higher precision. The rule structure is set to the same, obtained from the manual refinement of decision rules, throughout the optimisation process.

The coding, selection, crossover and mutation methods are same as in learning process. To calculate a fitness value for each individual, the fitness Equations (5.29) to (5.32) are used, by carrying out 15 trials with the maximal time $S_{\text{max}}$ set to 2000 steps. This is possible due to reduced task complexity, as the rule structure has already been determined.

*GA parameters*

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>No of generations</td>
<td>50</td>
</tr>
<tr>
<td>Population size</td>
<td>30</td>
</tr>
<tr>
<td>Chromosome length</td>
<td>40</td>
</tr>
<tr>
<td>Elitism</td>
<td>1</td>
</tr>
<tr>
<td>Crossover sites</td>
<td>3</td>
</tr>
<tr>
<td>Crossover probability</td>
<td>1</td>
</tr>
<tr>
<td>Mutation probability</td>
<td>0.05</td>
</tr>
</tbody>
</table>
Figure 5.15 Control Decision Tree
5.2.3.7 Exploiting Domain Knowledge

In this section it is demonstrated how the domain knowledge can be exploited to bypass the costly process of learning control rules from scratch. Instead of searching for both the structure of rule and the values of numerical parameters required by the rule, it has been started with a known rule structure and a GA is employed to tune the parameter values.

5.2.3.8 Bratko's control rule

To bypass the costly method of learning control rules from scratch, the rule structure has to be known. The inverted pendulum control rule derived by Bratko is considered as given in Equation (5.28) and diagrammatically shown in Figure 5.16. The rule includes four parameters $p_1$, $p_2$, $p_3$ and $p_4$.

A qualitative analysis of this control rule reveals that all the parameters have to be positive (Geva et al., 1993). The analysis also helps to understand the control strategy embodied in the control law. Consider first the situation where the cart is stationary at the centre of the tract, and the pole leaning at a positive angle with no angular velocity. The control action is then determined by:

$$ F = \text{sign} \left( p_3 \theta \right) $$

Clearly in this case only a positive force will erect the pole, and hence $p_3$ has angle, but with positive angular velocity only a positive $p_4$ will produce a force that reduces the angular velocity.
Figure 5.16  Bratko's Control Rule
The parameters $p_1$ and $p_2$ also have to be positive is not so obvious. Suppose that the cart is stationary somewhere on the right hand side of the track with the pole perfectly balanced. In this case the control force is determined by $p_1$. A positive value will cause the force to accelerate the cart away from the centre off the track. This action will initially move the cart further away from the centre, and may appear to be incorrect.

However, as a result of this action the pole will start falling to the left, making $\phi$ and $\phi'$ negative contribution to the force, attempting to erect the pole, will overcome the positive contribution from $x$. The net effect, over time, of the spoiling effect of $p_1$ and the correcting effect of $p_3$ and $p_4$ is to accelerate the cart towards the centre. To see how this net effect comes about consider the pole in a stationary upright position. A sequence of $n$ control actions in one direction will cause it to fall in the opposite direction, helped by gravity. More than $n$ control actions in the reverse direction are required to restore balance. This is to compensate for work done by gravity during recovery. Hence, the cart-pole receives a net acceleration in direction opposite to the initial direction of force application.

Being accelerated towards the centre, the cart eventually overshoots the centre and the opposite sequence of control action will tend to bring back towards the centre again. This mechanism alone will cause the cart to oscillate about the centre. Without friction the oscillations will not dampen required that will take the role of friction. This is provided by the positive value of the parameter $p_2$. Suppose that the cart is at the centre of the track, with pole balanced, but moving with constant velocity to the right. The braking is induced by positive value of $p_2$, through the angular contributions, in the same way as centring is induced by a positive $p_1$. The velocity contribution to the force causes the cart to accelerate to the right, thus increasing the cart’s velocity. However, again the side effect of this action is to cause the pole to fall to the left. The control actions that follow to erect the pole produce the desired net result of slowing down the cart.

Thus the strategy embodied in the control law reveals that the parameters $p_1$ and $p_4$ work towards maintaining the pole in an upright position (Geva et al, 1993). The parameter $p_1$ indirectly causes the cart to accelerate towards the centre of the track, by
causing the pole to lean in that direction. The parameter $p_2$ indirectly slows down the cart, by causing the pole to lean in a direction opposite to the direction of movement.

Thus the objective is to find their values not only ensuring the inverted pendulum system to survive virtually infinite number of time steps, but also yielding a high-quality behaviour of the system. High quality in the sense, the system should reach the desired condition in a minimal time and to stay in that condition with less deviation.

5.2.3.9 GA Tuned Controller

Thus by knowing the rule structure from Bratko's model, only the four parameters $p_1$, $p_2$, $p_3$, and $p_4$ were to be found out using GA. The control force used during the tuning process is of magnitude $\pm 10V$.

**Coding**

The four parameters values are represented as binary strings. Each string comprised of four 10-bit sections, corresponding to the parameters $p_1$, $p_2$, $p_3$, and $p_4$, respectively. The lower bound of considered parameter values is 0, while the discretization step is 0.05, giving the upper bound of 51.15.

**Selection**

The strings are evaluated from Equations. To get the fitness value of each individual, 25 trials are carried out with the maximal time Max set to 2000 steps, corresponding to 40s of simulated time. Elitism of a best string is done. Thus the best string of the last generation is retained.
### Crossover

After selection, crossover is done. Here three-point crossover with crossover probability of -0.6 is used. Since multi-parameter coding was done crossover sites are chosen so that at least one site is present between each 13 bits.

### Mutation and Termination Criterion

After crossover, mutation is done by flipping any bit, with a mutation probability of 0.05. The termination criterion was set to a maximum of 50 generations, within which the parameter values converges to an optimal solution.

### GA parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of generation</td>
<td>50</td>
</tr>
<tr>
<td>Population</td>
<td>30</td>
</tr>
<tr>
<td>Chromosome length</td>
<td>40</td>
</tr>
<tr>
<td>Elitism</td>
<td>1</td>
</tr>
<tr>
<td>Crossover</td>
<td>3</td>
</tr>
<tr>
<td>Crossover probability</td>
<td>0.6</td>
</tr>
<tr>
<td>Mutation</td>
<td>0.05</td>
</tr>
</tbody>
</table>
5.2.4 Results and Discussions

The control of the inverted pendulum on a simulated system is obtained by three methods.

i. By using Makarovic control rule without GA

ii. By learning the rules and finding the numerical values using GA to control the system

iii. By applying GA to tune the numerical parameters of Bratko control rule for inverted pendulum.

For each method, the system performances are obtained by plotting cart position and pole inclination against time instants. The system is simulated for 10000 time instants, so that a clear picture of the performance of the controller can be obtained. One time instant is equivalent to 20ms of the simulation time. The performance plots are obtained for different initial conditions, taken over the initial condition boundaries.

The quality of the controllers is inferred, by considering the time to attain the desired condition, and the degree to which the system stays in the desired condition. To make a comparative study, the performances for the following six sets of initial conditions are observed.

1. $x = 0.5$, $x' = -0.05$, $\phi = 2$, $\phi' = 1$
2. $x = -0.5$, $x' = -0.035$, $\phi = 2$, $\phi' = 1$
3. $x = -0.4$, $x' = 0.04$, $\phi = 1.5$, $\phi' = 0.8$
4. $x = 0.3$, $x' = -0.03$, $\phi = -2$, $\phi' = 0.6$
5. $x = -0.2$, $x' = 0.04$, $\phi = 1.5$, $\phi' = 0.7$
6. $x = -0.8$, $x' = -0.03$, $\phi = 2$, $\phi' = 0.6$
5.2.4.1 Makarovic Controller Performance

The controller performance plots for the above initial conditions are given in Figures 5.17 (a) to (f). It can be noted that the centring is perfectly done in most of the conditions. The time taken for centring depends on the initial conditions. It takes more time for centring, when the initial state is farther away. While considering the pole, it is balanced and oscillates about the zero angle with small magnitude. It can also be noted that for the 6th initial condition (Figure 5.17(f)), which is outside the legal initial condition boundary, the system goes unstable.

5.2.4.2 Genetic Controller Performance

The performance plots for genetic controller are given in Figures 5.18 (a) to (f). It can be seen that, though stable within the legal boundaries, the system performance is not good. This is because the rules are learned from scratch, considering the system as a black box, are sub-optimal rules. It can be noted that the cart tends to the centre, but centring is not perfect. Also, it can be seen that the pole oscillates abruptly, with large magnitudes about the zero angle. The system is unstable for the 6th initial condition (Figure 5.18(f)).

The rule decision tree is given in Figure 5.15 and the numerical parameter values for each state is given below:

\[
\begin{align*}
  x_{ref}^+ &= 0.54 & x_{ref}^- &= -0.54 \\
  x'_{ref}^+ &= 0.02 & x'_{ref}^- &= -0.02 \\
  \phi_{ref}^+ &= 0.60 & \phi_{ref}^- &= -0.60 \\
  \phi'_{ref}^+ &= 9.40 & \phi'_{ref}^- &= -9.40
\end{align*}
\]

5.2.4.3 Genetic Tuned Controller Performance

The performance plots for genetic tuned controllers are given in Figures 5.19 (a) to (f). It can be observed that the cart is perfectly centred in all cases. The time taken for the cart to centre for all initial conditions is the least for this method compared to the previous two methods. It can be seen that the cart centring is done within 500 time instants in all
Figure 5.17 (a) 
CART POSITION [initial condns $x=5$, $x'=-0.05$, $q=2$, $q'=1$]

Figure 5.17 (b) 
POLE INCLINATION [initial condns $x=5$, $x'=-0.05$, $q=2$, $q'=-1$]

Figure 5.17 (c) 
CART POSITION [initial condns $x=-5$, $x'=0.35$, $q=2$, $q'=-1$]

Figure 5.17 (d) 
POLE INCLINATION [initial condns $x=-5$, $x'=-0.35$, $q=2$, $q'=-1$]

Figure 5.17 Makarovic Controller Performance (Cont.)
Figure 5.17 (c)

CART POSITION
(initial conds \(x = -0.4, x' = 0.04, q = -1.5, q' = 0.8\))

Figure 5.17 (d)

POLE INCLINATION
(initial conds \(x = -0.4, x' = -0.04, q = -1.5, q' = 0.8\))

Figure 5.17 Makarovic Controller Performance (Cont.)
Figure 5.17 (e)  
CART POSITION \[ \text{initial conditions} \ x = -0.2 \ x' = 0.04 \ q = 1.5 \ q' = -7 \]  

Figure 5.17 (f)  
POLE INCLINATION \[ \text{initial conditions} \ x = -0.2 \ x' = 0.04 \ q = 1.5 \ q' = -7 \]  

Figure 5.17  Makarovic Controller Performance
Figure 5.18 (a)

Figure 5.18 (b)

Figure 5.18 Genetic Controller Performance (Cont.)
Figure 5.18 (c) CART POSITION [initial conditions $x = -0.4$, $x' = 0.04$, $q = -1.5$, $q' = 8$]

Figure 5.18 (d) POLE INCLINATION [initial conditions $x = -0.4$, $x' = 0.04$, $q = -1.5$, $q' = 8$]

Figure 5.18 (Cont.) CART POSITION [initial conditions $x = -0.3$, $x' = -0.03$, $q = -2$, $q' = -6$]

Figure 5.18 (d) POLE INCLINATION [initial conditions $x = -0.3$, $x' = -0.03$, $q = -2$, $q' = -6$]
Figure 5.18(e)  Genetic Controller Performance
Figure 5.19 (a)  CART POSITION (initial condns $x=.5 \ x'=-.05 \ q=2 \ q'=1$)

Figure 5.19 (b)  CART POSITION (initial condns $x=-.5 \ x'=-.035 \ q=2 \ q'=-1$)

Figure 5.19  POLE INCLINATION (initial condns $x=.5 \ x'=-.05 \ q=2 \ q'=1$)

Figure 5.19  POLE INCLINATION (initial condns $x=-.5 \ x'=-.035 \ q=2 \ q'=-1$)

Figure 5.19 (Cont.)  Genetic Tuned Controller Performance (Cont.)
**Figure 5.19**  Genetic Tuned Controller Performance (Cont.)

### (a) Initial Conditions: $x = -0.4$, $x' = -0.04$, $q = -1.5$, $q' = 8$

- **Graph:** CART POSITION
- **Axes:** TIME STEPS (1 unit = 0.02 sec)
- **Displacement (m):**
  - Minimum: $-1$
  - Maximum: $1$

### (b) Initial Conditions: $x = -0.4$, $x' = -0.04$, $q = -1.5$, $q' = 8$

- **Graph:** POLE INCLINATION
- **Axes:** TIME STEPS (1 unit = 0.02 sec)
- **Angle (degrees):**
  - Minimum: $-20$
  - Maximum: $20$

### (c) Initial Conditions: $x = 0.3$, $x' = -0.03$, $q = -2$, $q' = 6$

- **Graph:** CART POSITION
- **Axes:** TIME STEPS (1 unit = 0.02 sec)
- **Displacement (m):**
  - Minimum: $-1$
  - Maximum: $0.5$

### (d) Initial Conditions: $x = 0.3$, $x' = -0.03$, $q = -2$, $q' = 6$

- **Graph:** POLE INCLINATION
- **Axes:** TIME STEPS (1 unit = 0.02 sec)
- **Angle (degrees):**
  - Minimum: $-10$
  - Maximum: $10$
Figure 5.19 (e) Genetic Tuned Controller Performance

Figure 5.19 (f) Genetic Tuned Controller Performance
cases. While considering the pole, it oscillates about the zero angle, with less magnitude, which is lesser than that of the Makarovic’s controller and Genetic controller. It can also be seen that for the 6th initial condition this controller performs well, where as the other two controllers fail.

Genetic tuned controllers are also capable of tracking any set point. It can be seen from Figure 5.20 (a) to (f), that the cart tracks the setpoint initially and also when changed in the middle, i.e., at 2000th time instant. The performance is good, and the cart reaches the desired location from the previous location within 500 time instants in all the cases. While considering the pole, it oscillates about zero with small magnitude when the cart is at setpoint but there is a sudden overshoot at the instant when the setpoint is changes.

The numerical parameter values in Bratko control rule, which is found by GA, are given below

\[ p_1 = 5.95, \ p_2 = 14.05, \ p_3 = 44.05, \ p_4 = 6.45 \]

Thus the genetic tuned controller using the above parameters are robust. The robustness of the controller can be tested by changing the sampling time and giving asymmetrical voltages of +10 V and -15 V and +15 V and -10 V, in Bang-Bang Control Regime. The performances for the above conditions, for the 6th initial condition are given in Figure 5.21 (a) to (c). It can be seen that when sampling instant is changed to 40 ms (Figure 5.21 (a)) the controller tracks the setpoint and balances the pole, but with large oscillations. When the asymmetrical voltages are applied to the system a small offset occurs while tracking the setpoint (Figures 5.21 (b) and (c)) and overshoot occurs in some cases (Figure 5.21 (b)). The pole is balanced with small oscillations.

The performances of the above controllers are quantitatively evaluated from the fitness Equations for all the initial conditions. A comparison of controller performance, in terms of fitness, is given in Table 5.5.
Figure 5.20 Genetic Tuned Controller for Setpoint Tracking (Cont.)
Figure 5.20 (c) Genetic Tuned Controller for Setpoint Tracking (Cont.)
Figure 5.20 (e)  
POLE INCLINATION  
(initial conditions x=-.2 x'=-.04 q=1.5 q'= -7; setpoints: [-6 & -5])

Figure 5.20 (f)  
CART POSITION  
(initial conditions x=-.8 x'=-.03 q=-2 q'= -6; setpoints: [-6 & -5])
Figure 5.21 Genetic Tuned Controller Checked for Robustness
Figure 5.21 (c)  Genetic Tuned Controller Checked for Robustness
5.2.5 Conclusion

In this section, a three-stage framework, based on GAs, was proposed for learning control. First, operational control rules, represented as binary strings, were obtained without the prior knowledge about the system to be controlled. Then these rules, in the form of binary strings were manually converted to if-then sub-optimal rules. Thus the obtained control knowledge was made operational by fine-tuning numerical parameters that are part of this knowledge.

The above method of learning rules from scratch is expensive, also the rules are sub-optimal, which gives only a less satisfactory performance. Thus GA is used for tuning the numerical parameter values of a prior known knowledge of the system, given by Bratko’s control rule in bang-bang regime. The performances of the Genetic tuned controller was far better than the Genetic controller learned from scratch.

This framework was evaluated on the problem of the inverted pendulum control. It has been found out that the genetic tuned controller outperformed the genetic controller learned from scratch and the Makarovic controller.
### Table 5.5 Comparison of Fitness

<table>
<thead>
<tr>
<th>Initial Conditions</th>
<th>Makarovic Controller</th>
<th>Genetic Controller</th>
<th>Genetic Tuned Controller</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>$X'$</td>
<td>$\phi'$</td>
<td></td>
</tr>
<tr>
<td>0.5</td>
<td>-0.05</td>
<td>2.0</td>
<td>1.0</td>
</tr>
<tr>
<td>-0.5</td>
<td>-0.035</td>
<td>2.0</td>
<td>-1.0</td>
</tr>
<tr>
<td>-0.4</td>
<td>0.04</td>
<td>-1.5</td>
<td>0.8</td>
</tr>
<tr>
<td>0.3</td>
<td>-0.03</td>
<td>2.0</td>
<td>-0.6</td>
</tr>
<tr>
<td>-0.2</td>
<td>0.04</td>
<td>1.5</td>
<td>-0.7</td>
</tr>
<tr>
<td>-0.8</td>
<td>-0.03</td>
<td>-2.0</td>
<td>-0.6</td>
</tr>
</tbody>
</table>

Average Fitness Taken over the Whole Initial Condition Limits 0.881486 0.623189 0.986720