AN ABSTRACT OF THE THESIS OF

John David Saugen for the degree of
Master of Science in Electrical and Computer Engineering
presented on December 18, 1987.

Title: AN EXPERT SYSTEM FOR OPTIMAL TUNING
OF ADAPTIVE PID REGULATORS

Abstract approved: ____________________________ Wojciech J. Kolodziej

"Expert systems" is an area in the field of artificial intelligence which attempts to encode an "expert's" heuristic knowledge and reasoning ability into a computer program. The purpose of this study is to investigate the applicability of using an "expert system" in a closed-loop automatic control system.

An "expert system" was developed to automatically tune a proportional-integral-derivative (PID) controller using heuristic reasoning. The encoded "knowledge" stated how changes in the PID gains changed the shape of the system step response. By using abductive reasoning techniques, the "expert system" could change the shape of the system step response, and thus optimize it in terms of normal step response characteristics.

The heuristic knowledge was developed assuming that
the parameter values of the system plant changed in a random manner. It was shown that the expert system could tune the PID for wide variations in the plant parameters and thus the tuning method was shown to be adaptive. An adaptive phase-margin tuning method was also developed for comparison purposes.

The "expert system" tuned the PID controlled system successfully. A discussion is included on how to improve the "expert system" tuning scheme and possible extensions of this technique to more complicated systems.
An Expert System for Optimal Tuning of Adaptive PID Regulators

by

John David Saugen

A THESIS submitted to Oregon State University

in partial fulfillment of the requirements for the degree of Master of Science

Completed December 18, 1987

Commencement June 1988
APPROVED:

Redacted for privacy

Assistant Professor of Electrical and Computer Engineering
in charge of major

Redacted for privacy

Head of Department of Electrical and Computer Engineering

Redacted for privacy

Dean of Graduate School

Date thesis is presented December 18, 1987

Typed by researcher for John David Saugen
ACKNOWLEDGEMENT

I wish to thank my major advisor Professor W. J. Kolodziej for his assistance and refreshing ideas on this subject. I would also like to thank Professor D'Ambrosio for inspiring my interest in this subject through his courses, and Professor Mohler and Professor Jansen for being on my graduate committee.

I cannot give enough thanks to my parents, especially my father, Professor John L. Saugen, for their inspiration and enthusiasm which led me into this profession. And, of course, my deepest thanks to my wife, Jill, for her patience and sacrifices which allowed me to further my education.
# TABLE OF CONTENTS

I. INTRODUCTION ........................................ 1

II. STATEMENT OF PROBLEM .......................... 3
   - Closed-loop Control Systems .................. 3
   - PID Controller .................................. 4
   - System Definition ............................... 5
   - Control Problem Statement .................... 7
   - Expert Systems .................................. 7

III. ADAPTIVE TUNING BACKGROUND ................. 9
    - Frequency Domain Tuning ..................... 9
    - Phase Margin Tuning ......................... 10
    - Adaptive Control Methods .................... 11
    - Artificial Intelligence and Expert Systems 14
    - PID Tuning Methods ............................ 16

IV. PHASE MARGIN TUNING OF PID ................. 18
    - Critical Point Determination ................ 18
    - Critical Point Tuning ....................... 20
    - Phase Margin Adjustment ..................... 21
    - Performance Index of PID Controlled System 23
    - Estimation Method ............................. 25
    - Phase Margin Tuning Procedure .............. 26

V. EXPERT SYSTEMS .................................... 28
    - Adaptive Systems, Learning and Intelligence 28
    - Expert System Definition .................... 29

VI. EXPERT SYSTEM FOR PID TUNING ............... 33
    - Step Response Criteria ...................... 33
    - Development of Heuristics .................... 34
    - System Characterization ....................... 36
    - Causality versus Tuning ....................... 38
    - Goals and Sub-goals ............................ 40
    - Goal Determination .............................. 41

VII. EXPERT SYSTEM STRUCTURE AND IMPLEMENTATION 43
     - Rules and Data Structures ................... 43
     - Inference Engine .............................. 46
     - Prolog Programming Language ................. 52
VIII. RESULTS
Simulation Structure 53
Phase Margin Tuning Simulation 53
Phase Margin Tuning Results 57
Phase Margin Tuning/Estimation Problems and Solutions 60
Expert System Simulation 61
Expert System Results 62
Expert System Problems and Solutions 70

IX. CONCLUSIONS 73
Expert System Tuning versus Phase Margin Tuning 73
Extensions of Expert System 75
Concluding Remarks 78

BIBLIOGRAPHY 79

APPENDIX 81
A.1 Least Square Error Estimation Method 81
A.2 Expert System Tuning Rules 85
<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Textbook PID Control System</td>
<td>5</td>
</tr>
<tr>
<td>2. System configuration</td>
<td>6</td>
</tr>
<tr>
<td>3. Phase margin plot</td>
<td>10</td>
</tr>
<tr>
<td>4. Closed loop system</td>
<td>13</td>
</tr>
<tr>
<td>5. Definition of critical point</td>
<td>18</td>
</tr>
<tr>
<td>6. Relay control of system</td>
<td>19</td>
</tr>
<tr>
<td>7. Phase margin adjustment</td>
<td>22</td>
</tr>
<tr>
<td>8. Phase margin tuning procedure</td>
<td>27</td>
</tr>
<tr>
<td>9. Time response characteristics</td>
<td>34</td>
</tr>
<tr>
<td>10. Examples of PID tuning rules</td>
<td>36</td>
</tr>
<tr>
<td>11. Time response characteristics</td>
<td>37</td>
</tr>
<tr>
<td>12. Damping and stability examples</td>
<td>38</td>
</tr>
<tr>
<td>13. Qualitative categorizations based on parameter value</td>
<td>41</td>
</tr>
<tr>
<td>14. Example goal tree</td>
<td>49</td>
</tr>
<tr>
<td>15. Inference engine analysis flow</td>
<td>51</td>
</tr>
<tr>
<td>16. Simulation flow diagram</td>
<td>54</td>
</tr>
<tr>
<td>17. Example plant parameter sequence</td>
<td>56</td>
</tr>
<tr>
<td>18. ITAE and phase margin (1st time through sequence)</td>
<td>59</td>
</tr>
<tr>
<td>19. ITAE and phase margin (3rd time through sequence)</td>
<td>59</td>
</tr>
<tr>
<td>20. Max1, Min1, Tr and Ts results (Expert system)</td>
<td>64</td>
</tr>
</tbody>
</table>
21. $K_{PID}$, $K_I$, $K_D$, $\zeta_{PID}$ and $\omega_{nPID}$ variation (Expert system)  

22. Max1, Min1, Tr and Ts results  
   (Expert system, third plant parameter set)  

23. $K_{PID}$, $K_I$, $K_D$, $\zeta_{PID}$ and $\omega_{nPID}$ variation  
   (Expert system, third plant parameter set)  

24. ITAE comparison for phase margin tuner  
   and expert system tuner  

25. Multiple PID controllers
AN EXPERT SYSTEM FOR OPTIMAL TUNING
OF ADAPTIVE PID REGULATORS

I. INTRODUCTION

The purpose of this thesis is to investigate methods of tuning proportional-integral-derivative (PID) regulators used to control unknown systems. In particular, an expert system approach will be investigated. The expert system will be composed of a variety of "rules" developed by a person experienced in performing the tuning on the particular systems studied. The resulting tuning scheme will be compared to an automatic parameter estimation method. Lastly, the expert system approach will be extended to a multiple-PID system in which desired sub-system characteristics may change as the different PID's are tuned.

This thesis is organized as follows. Chapter II discusses and defines the system to be studied, and states the goals of the system simulation. Chapter III gives an historical background of several different methods of tuning adaptive regulators. Chapter IV describes a method of tuning based on the closed loop phase margin. Chapter V introduces the concept of "expert systems". Chapter VI discusses the "expert system" used in this study with a
description of its structure and implementation details following in chapter VII. Chapter VIII describes the results obtained through the system simulation. Chapter IX concludes with a comparison of the phase margin tuning method and the "expert system" tuning method, and describes some possible extensions and applications.
II. STATEMENT OF PROBLEM

Chapter II begins with a description of a closed-loop control system and the need for a controller. A PID controller is then defined and its implementation in a closed loop system is given. The chapter concludes with a statement of the control problem examined in this study and describes how an expert system might be utilized.

Closed-loop Control Systems

In general, a system is subjected to varying environmental conditions. These conditions change with time and affect the operation of the system in often unpredictable manners. So-called "open-loop" control consists of applying an input to a system and observing the output. Environmental conditions and/or tolerances in the system make it difficult to predict the system response for a given input. It is for this reason, that so-called "closed-loop" control is used, in which the system response signal is "fed back" and combined with the input-signal in an appropriate manner to form a new system input signal.

This closed-loop control provides a means of comparing the desired system response with the actual system response. However, simple proportional feedback of
the output signal is often inadequate. System gains and dynamics may affect system stability. In addition, even if the system is stable, it might not perform in an acceptable manner. For example, the system might respond too slowly or might never reach the desired output condition. For these reasons, and others, a "controller" is generally incorporated in the loop. A properly designed controller can usually improve the performance of the system in a predictable manner.

**PID Controller**

The particular controller studied in this thesis is the proportional-integral-derivative (PID) controller. Its form is

\[ u(t) = K_{PID} [e(t) + K_D \frac{de(t)}{dt} + K_I \int_{t_0}^{t} e(t) \, dt + u_I(t_0)] \]

(2.1)

where \( e(t) \) is the input signal to the PID, \( u(t) \) is the output signal and \( u_I(t_0) \) represents the initial value of the integral term at time \( t_0 \). Arbitrarily choosing \( t_0 = 0 \) and \( u_I(t_0) = 0 \) and taking Laplace transforms,

\[ U(s) = K_{PID} \left[ 1 + K_D s + \frac{K_I}{s} \right] E(s). \]

(2.2)

The output of the PID is thus the sum of a signal proportional to the input, a signal proportional to the time derivative of the input and a signal proportional to
the time integral of the input. The PID can be used to affect system damping, speed and steady-state errors. It should be noted that many other types of controllers exist. The PID controller is, however, a popular one.

**System Definition**

The configuration of the system to be studied is the "textbook PID controller," as defined by Astrom and Wittenmark [1] and as shown in figure 1.

![Figure 1. Textbook PID Control System](image)

In this figure, \( u_c(t) \) is the command input and \( y(t) \) is the controlled output. Note that the system consists of a PID controller in series with the plant and unity (negative) feedback.

The actual plant modeled in this study is a slowly-varying linear third-order system. In this instance, slowly-varying means that the plant parameters change slowly compared to the dynamics of the system. It is assumed that the parameter changes are slow enough to enable the system to be tested several times with the same plant parameters. Thus, for any desired series of tests performed, it is assumed that the plant parameters remain
constant\textsuperscript{1}.

The transfer function of the plant is of the following form:

\[
X(s) = \frac{Y(s)}{U(s)} = \frac{K}{(s + a)(s^2 + 2\zeta w_n s + w_n^2)}
\]

where \(K\), \(a\), \(\zeta\) and \(w_n\) are variable parameters. The system configuration is shown again with parameters in figure 2.

Figure 2. System configuration.

For the remainder of this paper, "plant parameters" refers to the parameters \(K\), \(a\), \(\zeta\) and \(w_n\), "PID parameters" refers to the parameters \(K_{PID}', K_D', K_I', \zeta_{PID}'\) and \(\omega_{nPID}'\), and "the system" will refer to the plant/PID controller combination as shown in figure 2.

\textsuperscript{1} This assumption is made to allow the system to be tuned repetitively until the system response is as desired.

\textsuperscript{2} For a description of \(\zeta_{PID}'\) and \(\omega_{nPID}'\), see chapter IV.
Control Problem Statement

For the problem at hand, it is desired to control the system step response. In other words, given a step input to the system, it is desired to control certain characteristics of the output (for example, rise time, overshoot, etc.). Since the environmentally varying parameters $K$, $\alpha$, $\zeta$ and $\omega_n$ are unknown, they must either be estimated (in some manner) or other system characteristics (such as critical gain and frequency) must be found. In both cases, some manner of test on the system must be performed, either during normal system operation or as a separate test. Once these characteristics are known (or estimated), certain techniques can be utilized to tune the PID (that is, adjust $K_{PID}$, $K_D$ and $K_I$). In this manner, the system response can be changed (within limits) to conform to desired specifications.

Expert Systems

It is the goal of this study to investigate how "Expert Systems" from the field of Artificial Intelligence can be utilized to aid in the PID tuning process. This will be done by first developing and analyzing a traditional parameter estimation and performance index minimization technique. Second, the system will be
"operated" (simulated operation) with a human performing the PID tuning. This will lead to the human developing a set of heuristic "rules" which will then be implemented in an expert system. This expert system will then be used to automatically tune the PID in a manner similar to that of a human operator. Lastly, the results will be compared to show the benefits and problems of each method.
III. ADAPTIVE TUNING BACKGROUND

Chapter III begins with a description of frequency domain tuning methods, in particular, a phase margin tuning method. A discussion on classical adaptive techniques follows. The chapter concludes with a presentation of artificial intelligence/expert systems and applications to control and PID tuning problems.

Frequency Domain Tuning

Probably the most widely utilized method for designing controllers is frequency domain analysis. By using this technique, frequency domain characteristics of the system can be predicted assuming that the plant and PID parameters are known. These frequency characteristics can then either be used to predict time response characteristics or be utilized on their own if system specifications are defined as such.

3. For example, system specifications may define system bandwidths and gain and phase margins.
Phase Margin Tuning

A relatively simple frequency domain characteristic to use is to tune the system for a specific phase margin. Given the phase margin frequency, the frequency at which the magnitude of the open-loop system gain is unity, phase margin is simply the amount of phase shift in the system at the phase margin frequency which would just produce instability (D'Azzo and Houpis [2]). This can be easily seen by looking at the Nyquist plot for the system; for example, see figure 3.

![Nyquist plot](image)

Figure 3. Phase margin plot.

As can be seen, the system plotted as $G(j\omega)$ exhibits a phase margin ($\phi$) of about $+45^\circ$, since this is the amount of phase shift which would cause the plot to intersect the $1e^{j(-180^\circ)}$ point and cause instability. It will be shown later that it is possible to arbitrarily change the phase margin (within limits) of a PID controlled system by changing the PID parameters.
Adaptive Control Methods

Adaptive control methods have been studied and implemented for many years. Ziegler and Nichols [3] described a set of "tuning-rules" for tuning PID regulators in 1943. Their method was based on selecting the PID parameters as a function of certain characteristics of the process step response. In this manner, a desired "set-point" could be maintained "...regardless of process load conditions...." This provided an adaptive structure for adjusting the controller based on its operating environment\(^4\).

Much work in adaptive control was done during the 1950's and 60's. During that time, Li, Young and Meiry [4] described 4 types of adaptive control systems:

1) Open loop adaptive control system,
2) Passive adaptive control system,
3) Active adaptive control with precisely defined performance indices,
4) Active adaptive control with inferred performance indices.

\(^4\) This method was, at the time, implemented manually by an operator who performed the PID adjustment. In this case, "self-tuning" implies that the operator is a part of the control process.
An open loop adaptive control system, commonly used in the form of a "gain scheduling" controller, consists of measuring the environmental conditions and using this information to adjust the controller parameters. This method relies on the knowledge of how the environment actually affects the plant parameters. Astrom points out that there is controversy concerning whether this technique should be considered adaptive, since the adjustment is done in open loop. Regardless, gain scheduling is an effective way to adapt a system to a varying environment.

Passive adaptive control refers to a system which is robust with respect to its operating environment and so needs no adjustment of the control parameters. Most closed loop systems possess this ability to some extent. It is an inherent characteristic of the closed loop system that the effects of parameter variations are generally reduced. Consider, for example, the system shown in figure 4. The open loop transfer function is $G_1(s) = 1/(s+a)$. The closed loop transfer function is $G_2(s) = 1/(s+a+1)$. Any variation in the parameter "a" will have more effect (particularly if a is much less than 1) in the open loop system, $G_1(s)$, than the closed loop system, $G_2(s)$. Thus, the closed loop system is adaptive in a passive sense.
Active adaptive control with precise performance indices concerns the minimization of a performance index through proper adjustment of the controller parameters. The technique consists of two parts: process identification and performance index minimization.

Process identification generally refers to the estimation of the parameters of an assumed system dynamical model. There are numerous works describing identification methods (e.g. Astrom [1] and [5], Braun [6]). A typical method is to sample the input-output signals of the process and fit the data to an assumed system model using a least-squared-error criterion. Some other methods include power-spectrum analysis and system perturbation using inserted test-signals. All of these methods provide information which is used to estimate sets of equations describing the dynamic response of the system.

Performance index minimization defines a quantity called the performance index which provides a measure of the quality of the system response to a particular condition. Utilizing the information provided by the
identification process, the performance index can be defined as a function of the system parameters and the controller parameters. This function can then be minimized with respect to the controller parameters resulting in a so-called "optimal" controller (see Bryson and Ho [7]).

Active adaptive control with inferred performance indices refers to what is commonly called "model reference adaptive control." A reference model is developed which has what would be considered desirable properties for the actual system to be controlled. The command signal is applied to both the model and the actual system. The output signals from each are compared resulting in an error signal which is indicative of how well the actual system is performing. This error signal is used to adjust the controller parameters. If the error signal remains close to zero, then the actual system is performing optimally with respect to the reference model.

Artificial Intelligence and Expert Systems

The idea of using a computer with "artificial intelligence" to perform control tasks is not new. Fu [8] described the concept of a learning adaptive control system using statistical decision theory in 1965. Andrease [9] described a learning machine applied to a
vehicle steering problem but goes on to say that

"...there are many problems common to the field of artificial intelligence and automatic control,...[but] the lack of suitable hardware continues to prevent the proper testing and realization of postulated adaptive control schemes."

It is interesting to see that present day hardware has alleviated many of the problems discovered in early artificial intelligence research. Such is the reason for the recent resurgence of research in this area.

Birdwell, Cockett and Gabriel [10] view artificial intelligence as "...the application of symbolic reasoning." A typical application of symbolic reasoning is in the field of expert systems. Astrom and Anton [11] utilize expert system techniques to solve the PID regulator tuning problem by using heuristic rules in conjunction with typical adaptive tuning methods. The reasoning enabled by use of their expert "rules" provides a convenient way of approaching those aspects of the problem less amenable to normal numerical approaches.

D'Ambrosio and others [12] utilize an expert system in a process management role. Their system uses a relatively high level "expert" which supervises and makes judgements on the activities of lower level process controllers, which may themselves use heuristic or analytical reasoning methods.
Most of the recent literature on PID tuning describes the use of many of the adaptive control techniques presented earlier, and the extensions of continuous-time analysis to discrete-time analysis. Ortega and Kelly [13] describe a pole-placement procedure, based on known estimates of the plant parameters. They also introduce an optimization procedure in which a performance criterion is minimized. Kim and Choi [14] present a modification of Ortega and Kelly's procedure which allows a less abrupt response to set-point changes (i.e., step-response performance). Phillips and Parr [15] discuss a digital PID predictor-controller which takes advantage of the prediction properties of the discrete-time system, and utilizes this when designing the PID.

As mentioned before, Astrom and Anton introduce the application of an expert system approach to tune the PID controller. Even though there exist many methods for tuning adaptive PID controllers, Astrom and Anton point out that there are many aspects of an actual working controller that are not readily addressed by the usual numerical techniques (for example, switching between automatic, manual and estimation modes, tuning during changes in plant parameters, operator interface, etc.). These types of problems are typically solved by heuristic
logic imposed by the operator. It is postulated that the use of an expert system is a useful way to deal with these type of issues.
IV. PHASE MARGIN TUNING OF PID

As stated earlier, a phase margin tuning procedure for the PID will be utilized in this study to provide a means of comparison for the expert system tuning procedure. The following discussion will describe a recently published phase margin tuning procedure. A performance index for a PID controlled system will then be described, followed by a method for determining an algebraic estimation for the actual system performance index.

Critical Point Determination

In order to implement a phase margin tuner, it is necessary that certain characteristics of the plant frequency response be known. A particularly simple characteristic to use is the critical point on the Nyquist plot. The critical point is that point where the plant has a phase shift of -180°. See figure 5.

![Figure 5. Definition of critical point.](image)
The critical point of the plant is thus defined by $K_C$, the critical gain$^5$, and $\omega_C$, the critical frequency. A unique method for determining the critical point experimentally has been developed by Astrom and Hagglund [16].

Astrom and Hagglund's method for determining the critical point consists of applying relay control to the closed loop system. See figure 6.

![Figure 6. Relay control of system.](image)

It is easy to show using sinusoidal-input-describing-function analysis that under relay control, the system

---

$^5$ Note that the critical gain $K_C$ is also the gain margin of the system.
in figure 6 will oscillate with frequency $\omega_C$ and amplitude $a = 4d/\pi K_C$, where $d =$ relay amplitude. (see Shinners [17]). From this, $K_C$ can be determined. Determining the critical gain and frequency using relay control is a relatively simple method. However, it does require an interruption of normal system operation since the PID must be removed and the relay inserted into the loop. This might not be acceptable or desirable for certain systems.

**Critical Point Tuning**

By adjusting the PID parameters, it is possible to move the critical point on the Nyquist plot to an arbitrary position (within limits). To see this, define

6. Note that $\omega_C$ is actually the frequency of the first harmonic component of the system oscillation at the output $y(t)$. If the system is of a "low pass" nature, then high frequency components of oscillation will be attenuated. If the system is not of a "low pass" nature, then high frequency components will remain, making the determination of $\omega_C$ more difficult. Note also that introducing a relay into the system does not guarantee that the system will oscillate.
the critical point of the plant $G(j\omega_c)$ and the PID as
\[ G(j\omega_c) = \frac{1}{K_C}e^{j(-180^\circ)}, \quad (4.1) \]
\[ G_{\text{PID}}(j\omega_c) = Ae^{j\theta}, \quad (4.2) \]

where
\[ A = K_{\text{PID}} \left[ 1 + \left( K_D \omega_C - K_I / \omega_C \right)^2 \right]^{1/2} \quad (4.3) \]
and \[ \theta = \tan^{-1} \left( K_D \omega_C - K_I / \omega_C \right), \quad -90^\circ < \theta < 90^\circ. \quad (4.4) \]
The complex gain of the system at $\omega_C$ is then
\[ G_{\text{TOTAL}}(j\omega_c) = \left( A / K_C \right) e^{j(\theta-180^\circ)}. \quad (4.5) \]

Thus, by adjusting $K_D$ and $K_I$, it is possible to arbitrarily adjust $\theta$ (within ±90°). It is then possible to arbitrarily adjust $A$, for a given $K_D$ and $K_I$, simply by adjusting $K_{\text{PID}}$. In this manner, the magnitude and phase of the system at $\omega_C$ may be changed to any desired values (if $K_{\text{PID}}$ is positive, the Nyquist plot at $\omega_C$ may be moved to any point in the left-half plane).

**Phase Margin Adjustment**

As was shown in figure 3, the phase margin of a plant is that point where the Nyquist plot crosses the unit circle. By adjusting the PID parameters, the complex gain of the system as in equation 4.5 can be changed so that the Nyquist plot crosses the unit circle precisely at the point where the phase margin would be as desired. See figure 7.
Since the phase of $G_{TOTAL}$ at $\omega_C$ is $0-180^\circ$, it is obvious that the phase margin is also equal to $\theta$. From equation 4.4,

$$K_D\omega_C - K_I/\omega_C = \tan \theta.$$  \hspace{1cm} (4.6)

Letting

$$K_I = \alpha/K_D,$$  \hspace{1cm} (4.7)

where $\alpha$ is an arbitrary design parameter related to the damping coefficient and natural frequency of the PID, then

$$K_D = \frac{\tan \theta + (\tan^2 \theta + 4\alpha)^{1/2}}{2\omega_C}.$$  \hspace{1cm} (4.8)

7. Rewriting $G_{PID}(s)$,

$$G_{PID}(s) = (K_{PID}K_D/s) [s^2 + (1/K_D)s + K_I/K_D].$$

Then, $\omega_{nPID} = (K_I/K_D)^{1/2} = \alpha^{1/2}/K_D$ and $\zeta_{PID} = 1/(2\omega_{nPID}K_D) = 1/2\alpha^{1/2}$.

It is thus possible to specify the PID parameters in terms of $\omega_{nPID}$ or $\zeta_{PID}$. The other parameter and $K_{PID}$ then become functions of the phase margin $\theta$ as follows in the discussion.
For the Nyquist plot to intersect the unit circle at $\omega_C$ with a phase margin of $\theta$, $A/K_C$ must equal unity (see equation 4.5). Substituting equation 4.6 into 4.3, it follows that
\[ A = \frac{K_{PID}}{\cos \theta}. \] (4.9)
Thus,
\[ \frac{A}{K_C} = \frac{K_{PID}}{(K_C \cos \theta)} = 1, \] (4.10)
and
\[ K_{PID} = K_C \cos \theta. \] (4.11)

Using equations 4.7, 4.8 and 4.11, a system with a plant having a critical gain $1/K_C$ and a critical frequency $\omega_C$ can be adjusted to have a desired phase margin $\theta$.

**Performance Index of PID Controlled System**

The phase margin tuner defined in the previous section has as variables $\theta$, the phase margin, and $\alpha$, a parameter related to the damping coefficient or natural frequency of the PID. What remains to be found are suitable values for $\theta$ and $\alpha$. In order to optimize the PID controlled system, a performance index needs to be developed and then minimized with respect to $\theta$ and $\alpha$. A performance index shown by Shinners [17] is chosen. The performance index chosen is the integral of time multiplied by the absolute value of error (ITAE).
The ITAE has the following form:

\[
J = \text{ITAE} = \int_{t_0}^{t} |c(t) - r(t)| \, dt \tag{4.12}
\]

where \( c(t) - r(t) = \text{error}(t) \). The ITAE thus penalizes long-term errors in the system output, in effect, requiring the system to have a short settling time\(^8\).

To minimize the performance index \( J \), it is desirable to find a set of equations relating \( c(t) \) and \( r(t) \) to \( J \). However, since the system being developed is one which is adaptive, i.e. the plant parameters are changing in an unpredictable way, it is very difficult to find such a relation. Even if the plant parameters were known, it would still be necessary to find a solution for \( c(t) \) and then minimize \( J \). Rather than do this, estimation techniques can be used to find an algebraic function relating \( J \) to \( \omega_c, K_c, \theta \) and \( \sigma \) using actual system response data.

8. It was found during the system simulation that a performance index of this form unduly penalized responses which "looked good" but had small errors after a length of time. The actual ITAE performance index used was modified to eliminate any penalty for long-term errors not exceeding 5% of the input signal (i.e., within the tolerances of the settling-time definition).
Estimation Method

Suppose that $J = \text{ITAE} = f(\omega_C, K_C, \theta, \alpha)$ with $\alpha$ constant. Given a number of system "trials", this function $f$ can be estimated by performing a linear-regression analysis on the data $J$, $\omega_C$, $K_C$ and $\theta$. It is advantageous to choose the function $f$ in the form of a quadratic (second-order). This choice is one of the simplest which results in a well-defined minimum.

Let $Y$ be an estimate for $J$, such that

$$Y = A + B[\omega_C K_C \theta]^T + [\omega_C K_C \theta]C[\omega_C K_C \theta]^T$$

(4.13)

where $B = [b_1 \ b_2 \ b_3]$ (4.14)

and $C = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ 0 & c_{22} & c_{23} \\ 0 & 0 & c_{33} \end{bmatrix}$ (4.15)

Then in order to minimize $J$ with respect to $\theta$, $dY/d\theta$ must be set equal to zero:

$$dY/d\theta = d/d\theta [A + b_1 \omega_C + b_2 K_C + b_3 \theta +$$

$$c_{11} \omega_C^2 + c_{12} \omega_C K_C + c_{13} \omega_C \theta +$$

$$c_{22} K_C^2 + c_{23} K_C \theta +$$

$$c_{33} \theta^2] = b_3 + c_{13} \omega_C + c_{23} K_C + 2c_{33} \theta = 0.$$ (4.16)

Then,

$$\theta = -(b_3 + c_{13} \omega_C + c_{23} K_C) / 2c_{33}$$ (4.17)

minimizes $J$ with respect to $\theta$. 
Thus given values for $A$, $B$ and $C$, it is possible to predict the value of $J = ITAE$ for future values of $\omega_C$, $K_C$ and $\theta$ and also minimize $J$ with respect to $\theta$. A procedure for finding $A$, $B$ and $C$, given a set of measurements of $ITAE$, $\omega_C$, $K_C$ and $\theta$ is presented in appendix A.1.

**Phase Margin Tuning Procedure**

Tuning the PID for a desired phase margin thus consists of the following tasks:

1. critical point determination,
2. estimation of performance index as $f(\omega_C, K_C, \theta, \alpha)$,
3. minimization of performance index with respect to $\theta$,
4. critical point adjustment for phase margin $= \theta$,
5. actual PID adjustment.

A flow diagram of the tuning procedure is shown in figure 8.
Find critical point (e.g. using relay-control)

Update performance index estimation

Find phase margin which minimizes performance index

Calculate PID parameters which give desired phase margin

Adjust PID parameters and continue system operation

Figure 8. Phase margin tuning procedure.
V. EXPERT SYSTEMS

This chapter describes how the concepts of learning and intelligence are applicable to adaptive systems. A definition of an expert system and its ability to reason follows.

Adaptive Systems, Learning and Intelligence

A common misconception about artificial intelligence is the idea of learning. It is an attractive but not very useful notion to put together a system which learns everything that it needs starting from scratch. As Charniak and McDermott [18] point out, one cannot "...build a baby and let it just learn these things." Instead, it is necessary to build an initial body of knowledge into the original system. Only then can the system "learn" efficiently.

In a sense, an adaptive system is a system which learns. Most classical adaptive systems incorporate some method of system identification. This identification can be viewed as learning, that is, the acquisition of knowledge about the system or the environment. Classical estimation techniques would not generally be classified as having "intelligence"; rather they consist of strict algorithms which estimate system parameters. These
algorithms perform a large number of computations, and thus provide the system with information which is not readily apparent. This knowledge is, however, not based upon reasoning, and the ability to reason is, by definition, a criterion for intelligence.

An adaptive system which reasons, however, could be considered to have intelligence. Reasoning in this sense means to use an initial set of facts and rules to infer other facts. A system which truly learns would also have the ability to infer new rules and apply them to the initial facts as well as to new facts. The first step, however, is to form a system which can make inferences from the initial set of facts. Such a system is called an expert system.

Expert System Definition

An expert system is a system which applies a set of rules developed by a person who is an "expert" at the task at hand. Charniak and McDermott portray the process of developing these rules as an iterative process. In their description, an expert is asked how tasks are performed. The expert then replies with general, somewhat vague descriptions of how things are done. This process is continued indefinitely, general rules become supplemented with and conditioned by more and more specific rules. The
key issue which prevents the process from becoming one of simply writing a traditional computer program is the difficulty or impossibility of defining the task simply in terms of normal algorithms. Yin and Solomon [19] and others call the process of developing these rules knowledge engineering. Rules, in this context, are of the form

\[
\text{if } \langle \text{set of facts} \rangle \\
\text{then } \langle \text{perform action or make conclusion} \rangle.
\]

Expert systems are often used to perform abductive reasoning. Abductive reasoning is the determination of possible explanations for an event or condition. Taking an example from Charniak and McDermott, given the rule

\[
\text{if (a person is drunk)} \\
\text{then (the person will not walk straight)}
\]

and the fact

Jack does not walk straight,

abductive reasoning says that Jack could be drunk. This is not to say that Jack is drunk, rather drunkenness is a possible explanation for his not walking straight. Suppose that there is another rule

\[
\text{if (a person has a broken leg)} \\
\text{then (the person will not walk straight)}.
\]

If Jack does not walk straight, it can be abduced that Jack could be drunk or Jack could have a broken leg. As the rules are stated, there is nothing to determine which
is the correct explanation for Jack's behavior. It must be clarified that these are not the only possible explanations. It is possible that Jack just fell out of a chair and so was dizzy. But this explanation is not stated as one of the rules. The abductive reasoning only applies to the particular set of rules given.

There are several advantages in using this abductive technique. First, an expert can generally explain what will happen in a given situation, given a particular set of facts, but may have trouble determining the facts which led to a particular event which might have occurred. Second, the level of abduction can be increased indefinitely. Continuing the above example, suppose that there is another rule

\[
\text{if (a person drinks too much)}
\]
\[
\text{then (the person will become drunk).}
\]

The abduction process will continue to postulate that Jack might have had too much to drink. A third advantage is that this process allows one to draw a large number of conclusions from a relatively small number of rules. For example, given the following rules:

\[
\text{if a then b, if b then c,}
\]
\[
\text{if c then d, if d then e,}
\]

then the following statements are also true:

\[
\text{if a then c, if a then d, if a then e,}
\]
\[
\text{if b then d, if b then e, if c then e.}
\]
A strictly deductive scheme which must specify all possible rule combinations would require 10 rules instead of 4. If there are a large number of rules, formally stating all of the possible combinations could be time consuming, if not impossible. The final and perhaps most important advantage of using this type of rule structure is that rules developed in this manner are easy to modify. It is easy to add conditional statements to each rule and change requirements.
VI. EXPERT SYSTEM FOR PID TUNING

This chapter begins by defining step response criteria and follows with a description of how the expert system rules, or heuristics, were developed for this study. It continues with a discussion of how these rules are used to infer tuning decisions. The chapter concludes with a description of the method used in determining actual system tuning goals.

Step Response Criteria

System specifications, as in this study, are sometimes specified in terms of time response characteristics (instead of or in addition to frequency domain characteristics). In this case, the analysis of the system must either be done in the time domain, or the time domain results must be translated from the frequency domain analysis. The predicted results can then be compared with the system time response specifications.

Typical time-response specifications are:

rise-time (from 10% to 90% of command value, \( t_r \)),
maximum overshoot (\( \text{max1} \)),
minimum value of second-overshoot (\( \text{min1} \))
and settling-time (to within ±5% of command value, \( t_s \)).

These characteristics are in response to a step input to
the system. A typical response is shown in figure 9.

Figure 9. Time response characteristics.

Development of Heuristics

During the development of the expert system for this study, the development of heuristic rules posed several problems. Initial work focussed on finding analytic solutions to the control problem. This was a natural and classical approach. However, it was soon discovered that the computations involved were either extremely simple (for example, using the phase margin tuner in section 4 with constant values for $\theta$ and $\alpha$) or extremely complex (for example, using classical optimization techniques). The problem with either of these approaches, notwithstanding the difficult computations in the optimization approach, is that neither gives a very good intuitive feeling for what is being accomplished during the tuning process. Furthermore, an expert system is most
amenable to a situation where an algorithmic approach is difficult.

Several different approaches can be used when developing these heuristic rules. Analysis of the data from sample responses can provide a very well defined set of rules, but this gives very little intuitive feel for how the PID parameters effects the system response. To provide a better medium for portraying the system response, a graphical display of the actual time response was used. In this manner, the "expert operator" could develop an intuitive insight into how different aspects of the system response related to the PID parameters and to other response characteristics.

Experiments were performed in which the PID parameters were varied and the resulting change in the system response was noted. By performing these tests in a number of different, random "environments" in which the plant parameters varied in an unknown random manner, rules relating the change in system performance to the change in PID parameters, regardless of the plant parameters, were developed. These rules thus stated nothing about what actual values to set the PID parameters to. Instead, the rules stated general notions about how changing a

9. Here, one can consider the tuning method to be robust, as opposed to the plant itself being robust.
particular PID parameter changes the system response, and how different characteristics of the system response affected other response characteristics. Examples of the rules developed are shown in figure 6.2 (a complete listing of the rules is shown in appendix A.2).

\[
\text{if } (\text{and } (max_1 > 1.02) \ (min_1 < 0.98) \\
\quad \quad (max_2 < max_1)) \\
\text{then } (\text{and } (\text{system underdamped}) \ (\text{system stable}))
\]

\[
\text{if } (K_{PID} \text{ increases}) \text{ then } (max_1 \text{ increases})
\]

\[
\text{if } (\text{and } (K_D \text{ increases}) \ (\text{system underdamped})) \\
\text{then } (t_{\text{max}} \text{ decreases})
\]

\[
\text{if } (\text{and } (K_D \text{ increases}) \ (\text{system underdamped})) \\
\text{then } (t_{\text{min}} \text{ decreases})
\]

\[
\text{if } (\text{and } (K_D \text{ increases}) \ (\text{system underdamped})) \\
\text{then } (max_2 \text{ decreases})
\]

\[
\text{if } (\text{and } (\text{second-max decreases}) \\
\quad \quad (t_{\text{min}} \text{ decreases})) \\
\text{then } (\text{settling-time decreases})
\]

Figure 10. Examples of PID tuning rules.

**System Characterization**

As mentioned earlier, the time-response specification parameters were assumed to be:

- rise-time (from 10% to 90% of command value, tr),
- maximum overshoot (max1),
- minimum value of second-overshoot (min1)

and settling-time (to within ±5% of command value, ts).

In addition to the time-response specifications, it
was found that certain other characteristics of the time-
response were apparent and could furnish additional
heuristics to aid in the tuning process. The other
characteristics used were:

\[ t_{\text{max}}, t_{\text{min}}, \text{max}^2 \]

and ITAE (integral of time multiplied by the absolute
value of error).

All of the characteristics used were those which give
the expert an intuitive feel for their effects on the
system performance. Additionally, all of the
characteristics (except for ITAE) were relatively easy to
see from the step response display (though the system
itself found the values analytically). See figure 11.

![Time response characteristics](image)

Figure 11. Time response characteristics.

In addition to the above, the system response was
characterized by the damping and stability of the system.
These characterizations were determined from the other
system characteristics. Examples are shown in figure 12.
Causality versus Tuning

In a typical expert system, rules such as those developed above are used to find explanations for the occurrence of events. From figure 10, for example, if max1 increases, then a possible explanation is that $K_{PID}$ increased. This is an example of using the expert system to determine the causality behind an event, namely the fact that max1 increased. This is a fundamental aspect of a "traditional" adaptive system in which an attempt is made to determine the characteristics of the system and then algorithmic techniques are used to perform system tuning.

The expert system developed for this paper, however, uses an opposite approach. In this approach, the
characteristics of the system are determined directly from the system response. The expert system then uses the rules to find a set of possible tuning actions to initiate a desired change in the system response. For example, if it is determined from the system response that max1 needs to be increased, then the expert system can determine that increasing $K_{PID}$ would be a valid way to accomplish the necessary increase in max1$^{10}$.

To be more precise in the terminology, a rule written in the form, for example, (if (k decreases) then (max1 decreases)), is not an abductive rule. If, however, the rule is rewritten in the form, (if (k too-large) then (max1 too-large)), then this rule would be considered abductive. This rule would imply that a possible reason for max1 being too large is that k is too large (and so, k should be decreased). To find the consequences of a change in a PID parameter, however, it is more convenient to retain the rules in the first form. The information contained in the rules is identical, but for convenience, the rules were written only in the first form, to simplify their use in both the deductive (finding the consequences) and abductive (finding what is the cause of the problem) modes.

10. Note that with this scheme, there may exist several valid ways to tune the PID.
Goals and Sub-goals

It can be seen from the sample of rules above that a particular "goal" may consist of several "levels" of sub-goals. For example, suppose that it is desired to decrease the settling-time. One way is to decrease the value of max2 and decrease tmin. A way to decrease the value of max2, if the system is underdamped, is to increase $K_D$. A way to decrease tmin, if the system is underdamped, is also to increase $K_D$. Thus, if the system is underdamped, then an increase in $K_D$ should decrease the settling-time, as well as decrease the value of max2 and tmin.

This manner of abduction can be continued to several levels of goals. To satisfy one goal, it may be necessary to satisfy another goal, which, in turn, may require other goals to be satisfied, etc. Note, however, that it is possible for sub-goals to be in contradiction. For example, if it were desired to decrease the settling-time and increase tmin, then the above rules would not lead to a valid result.
Goal Determination

In order for the expert system to operate, a set of initial goals is necessary. In this case, the initial goals are simply to tune the system until all of the time-response specifications are met. It was found, however, that simple goals such as "decrease max1 until within specifications" were not as adequate as hoped. Simple goals such as this relied on true or false statements about the system such as within-specifications or not-within-specifications. What was needed was a more quantitative description of the system, without reverting to the algorithmic approach.

This quantitative approach was handled by defining a range of acceptable values and a range of desired values for each time-response parameter. This gave rise to 5 different qualitative description possibilities for each parameter as shown in figure 13.

<table>
<thead>
<tr>
<th>I</th>
<th>II</th>
<th>III</th>
<th>IV</th>
<th>V</th>
</tr>
</thead>
<tbody>
<tr>
<td>min</td>
<td>min</td>
<td>max</td>
<td>max</td>
<td>Value</td>
</tr>
<tr>
<td>acceptable</td>
<td>desired</td>
<td>desired</td>
<td>acceptable</td>
<td></td>
</tr>
</tbody>
</table>

Figure 13. Qualitative categorizations based on parameter value

From this, each parameter results in a goals depending on its categorization:
<table>
<thead>
<tr>
<th>Category</th>
<th>Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>parameter must increase</td>
</tr>
<tr>
<td>II</td>
<td>parameter may increase or stay the same, but not decrease</td>
</tr>
<tr>
<td>III</td>
<td>parameter may increase, decrease or stay the same</td>
</tr>
<tr>
<td>IV</td>
<td>parameter may decrease or stay the same but not increase</td>
</tr>
<tr>
<td>V</td>
<td>parameter must decrease.</td>
</tr>
</tbody>
</table>

Thus, the initial goal will be one of necessity or allowance for each parameter to increase, decrease or stay the same. This categorization technique allows for the possibility of "rough" tuning, before "fine" tuning\(^{11}\).

---

\(^{11}\) This also prevented oscillations in the tuning recommendations by effectively preventing small changes in performance from resulting in drastic changes in the tuning goals. If a parameter was in category II or IV, the system allowed it to change appropriately or remain the same. This allowed for the more efficient tuning of the parameters which were out of specification, i.e. in category I or V.
VII. EXPERT SYSTEM STRUCTURE AND IMPLEMENTATION

This chapter begins with a discussion of rules and data structures which were used in the expert system, and follows with a description of how the expert system inference engine was developed. The chapter concludes with a discussion on Prolog, the programming language used to implement the expert system.

Rules and Data Structures

In the initial development of the expert system for this study, many different rule and data structures were investigated. In subsequent thought and trials, with the emphasis on simplicity and flexibility, it became apparent that certain structures gave better results than others.

Initially, formal, rigid types of structures were utilized. Example of these structures follow:

\[(\text{parameter}, \text{effect of changing } K_{\text{PID}}), \ldots, \text{effect of changing } w_{\text{nPID}})\]

\[(\text{parameter}, \text{effect of changing first-max}), \ldots, \text{effect of changing settling-time})\].

These have the advantage that they are easy to manipulate. Relationships between different parameters are easy to find due to the strict ordering and simplicity of the structure. However, this type of structure proves to be
rather inflexible. Strict ordering implies that all attributes and parameters have to be included in each rule, whether they are needed or not. Thus, null values must be defined. In addition, a rigid structure makes it difficult to add rules using new parameters.

To alleviate the problems in the preceding paragraph, a list type of structure was investigated. For example:

\[(<\text{parameter}>, [[K_{\text{PID}}\text{, effect}], \ldots, [\omega_{n\text{PID}}\text{, effect}], \ldots])\].

By using lists, infinitely long characterizations can be implemented. The difficulty with lists is that some form of referencing scheme is necessary (for example, including $K_{\text{PID}}$ with its effect in the above example). This is not that difficult, but adds unnecessary complexity to the problem.

Another possible structure is to combine the two just mentioned. This is an idea used by Yin and Solomon. Certain characteristics which are known to be used in every instance are put into a rigid structure, followed by a list of optional elements. For example:

\[(<\text{parameter}>, <\text{system characterization}>, \ldots, <\text{system stability}>,

[[K_{\text{PID}} \text{ effect}], \ldots, [\omega_{n\text{PID}} \text{ effect}], \ldots]).\]

All of the above structures are what Yin and Solomon call "logic-based" expert systems. More popular, and more amenable to the problem at hand, however, are "rule-based" expert systems. The information contained is identical to
that in the logic-based structure, but its if-then type of structure necessitates a different method of obtaining inferences.

The rule-based structure has the form:

if <set of conditions> then <set of results>.

For example:

if (\(K_{\text{PID}}\) increases) then (\(\text{max1}\) increases).

One major advantage of writing rules in this format is the ease in which they can be written and understood. In effect, the rules are written in a form similar to English grammar. Furthermore, the set of conditions and/or set of results can include any number of standard Boolean operators. Thus, there is a "built in" list structure in the sense that any number of influences can be included.

Another distinct advantage of the rule-based structure is that each rule is actually a clause. As a clause, the rule can be interpreted as both a fact and as an operator to discover other facts. Thus, several rules can be utilized in finding an inference to a single rule (this was shown earlier in the "Jack drinking too much" example).

Since each rule is a clause, the system inference engine (to be discussed in the next section) is simplified. The rules are more or less self contained in that no structure needs to be defined to interpret their
meaning\textsuperscript{12}. Thus, human-interpretation and machine-interpretation are both simplified.

Since simplicity and flexibility are the key issues in the initial development phase for this expert system, the rule-based structure is the natural choice. As the system becomes better defined, it may be found that another structure could be more beneficial in terms of speed and efficiency. For the present system, however, this aspect of the problem will be secondary to that of ease of development.

**Inference Engine**

The *expert system* inference engine is that part of the *expert system* which processes the rules in a manner to arrive at valid inferences. The inference engine is thus a computer program which can apply the appropriate rules to given situations and properly identify the correct inferences from the rules.

The inference engine for the PID tuning *expert system*

\textsuperscript{12} Compare the rule-based "if (K\textsubscript{PID} increases) then (max1 increases)" with the logic-based "(max1, increases, K\textsubscript{PID}', increases,K\textsubscript{I}',null,...,\omega\textsubscript{nPID}',null)." The first structure can be interpreted directly while some explanation needs to be made for the second.
must handle 3 types of inferences: system characterization, goal determination and goal resolution. Each is handled separately.

System characterization, as discussed earlier, uses the raw data from the simulated step response and finds the necessary system descriptions. This process is relatively simple, primarily consisting of deductive types of rules. These rules are taken directly from the definitions shown earlier.

Goal determination consists of two steps. The first step is determining which tuning category, as defined previously in figure 13, a parameter fits into. Again this is a simple deductive procedure. The second step is interwoven with the goal resolution inference scheme.

Since the categorization scheme presented in figure 13 allows for the possibility of a non-changing parameter, the actual set of goals after a particular event can be viewed as consisting of 2 constraints. For example, the set of goals

\[(\text{max1 must increase}), \ (\text{max2 may increase}),\]

can be interpreted as

\[(\text{max1 must increase}), \ (\text{not (max2 must decrease)}).\]

In this case the first goal is one constraint and the second goal leads to a constraint which is a sort of "negative" of the original goal. The actual way that this is implemented in the expert system is to define necessary
goals and acceptable goals. For example, the goals above would give rise to the following 5 requirements:

\[(\text{max1 must increase}), \quad (\text{max1 may increase}), \quad (\text{max2 must "null"}), \quad (\text{max2 may increase}), \quad (\text{max2 may stay-the-same})\].

In this case, there is one "must" requirement and 3 "may" requirements. Note that "null" in this case means that there are no "must" requirements for max2. The condition for a goal succeeding is that all of the "must" goals must be satisfied, and any resulting consequences must be valid under the "may" restrictions. For example, with the above goals, suppose that increasing $K_{\text{PID}}$ increases max1 and increases max2. Then increasing $K_{\text{PID}}$ is a valid result since all of the "must" goals are satisfied and any other consequences, namely that max2 also increases, are valid under the "may" restrictions.

The actual goal resolution scheme can be performed in two different manners. The first method assumes a selected parameter change, finds the consequences of that change, checks for the satisfaction of "must" goals and checks for the validity of the consequences with respect to the "may" requirements. This is performed 10 different times, twice (increase/decrease) for each of the five variable PID parameters ($K_{\text{PID}}$, $K_I$, $K_D$, $\omega_{\text{PID}}$, $\xi_{\text{PID}}$). This method thus gives all of the possible ways to tune the PID and satisfy the goals. For example, suppose the
system had the goal tree shown in figure 14.

```
        tune PID
         /         /
        /           /
        /             /  
 increase | increase     
          /           /  
        /             /        /  
 increase | increase     | increase    
           /           /             /  
          /             /               /  
 max1 increases | max2 increases | max1 increases    
                 |             | damping critical
                 |             | max2 increases
```

Figure 14. Example goal tree.

If the system goal was "increase max1" and "do anything to max2", then the inference engine would first try increasing $K_{PID}$ and find that this increases max1 and max2 and thus is a valid parameter change. Then, increasing $K_I$ would be attempted and found to also be valid (if the system damping was critical).

The second method begins with an actual goal, and finds any parameter change whose set of consequences satisfies all of the requirements. Using the same goal tree and goals as above, this method would first try increasing $K_{PID}$ (as before) and find that max1 increases. Backtracking would then take place to the "increase $K_{PID}$" point. The system would then find all of the consequences of increasing $K_{PID}$ and find that max1 and max2 increase and so increasing $K_{PID}$ is again a valid parameter change. This method returns only one possible way to tune the PID.

In general, a rule may contain conditional statements in addition to the cause-effect relationship between the parameter change and the change in system response. For
example, from the goal tree above, there would be a corresponding rule:

\[
\text{if (and (} K_I \text{ increases) (damping critical))}
\]
\[
\text{then (max2 increases).}
\]

The conditional statement in this rule is "damping critical". Since the rules are stated in an if-then form, verification of this statement simply involves finding the corresponding damping rule and checking its conditionals.

For example, the corresponding rule might be

\[
\text{if (and (max1 < 1.02) (min1 > 0.98) (max2 max1))}
\]
\[
\text{then (damping critical).}
\]

Thus, 2nd-level and deeper inferences are automatically performed due to the if-then rule structure.

The entire inference engine scheme is shown in figure 15.
Figure 15. Inference engine analysis flow.
The primary symbolic reasoning languages in use today are Lisp and Prolog. Both languages have list processing and symbolic manipulation capabilities. Prolog was chosen to implement the expert system in this study. The version of Prolog used was Turbo Prolog™ by Borland International, Inc.

Turbo Prolog has several features which were utilized in the development of the system simulation. The programs are compiled, resulting in fast execution speeds. Turbo Prolog's unification scheme is imbedded in the software, transparent to the user. The software featured windowing and graphics capabilities. Database manipulation and disk access routines allowed for the efficient handling of the program modules (subroutines).
VIII. RESULTS

This chapter begins with a description of the structure of the system simulation. It follows with a discussion of the phase margin tuning results, and its associated problems. The chapter concludes with a discussion of the expert system results, and the problems which arose during the expert system simulation.

Simulation Structure

The phase-margin tuner and expert system were programmed and implemented on a Sperry\textsuperscript{TM} Model 5 Personal Computer (IBM\textsuperscript{TM} compatible). The simulation flow diagram is shown in figure 16.
In order to compare the phase-margin tuner with the expert system tuner, a sequence of plant parameter values was generated. This sequence consisted of random variations in the following parameters:

\[ K, \xi, \text{ and } \omega_n, \]

where the plant transfer function is:

\[ X(s) = \frac{K}{(s + \alpha)(s^2 + 2\xi\omega_n s + \omega_n^2)}. \quad (8.1) \]

The random variations in the plant parameters were accomplished by generating a random number for each parameter (within a selected range) and adding this to the
"old" value to get the "new" parameter value. Limits were placed on each parameter to prevent overly-large and overly-small parameter values. This method thus simulates a situation in which the plant parameters change in a random, continuous manner. The resulting sequences were applied to both the phase-margin tuner and the expert system tuner\textsuperscript{13} (see figure 17).

\textsuperscript{13} The range of system parameter values allowed was:

\[0.5 \leq K \leq 3.0, \ 0.4 \leq \xi \leq 2.0, \ 0.5 \leq \omega_n \leq 2.0, \ \alpha=1.0.\]

The amount of change which could occur (randomly) from one set of parameter values to the next was:

\[-0.2 \leq \delta K \leq +0.2, \ -0.2 \leq \delta \xi \leq +0.2, \ -0.2 \leq \delta \omega_n \leq +0.2.\]
Figure 17. Example plant parameter sequence.

The actual time response was calculated using a fourth-order Runge-Kutta method with a step size approximately equal to 1/100 of the settling time of a typical system step response.

After the time response is completed, the tuner (i.e. phase-margin tuner or expert system tuner) uses the data to tune the PID appropriately. If the simulation is performing phase-margin tuning, then the estimation parameters are updated and a single time response is performed with the new PID. If the simulation is using
the expert system to perform the PID tuning, the time response is repeated, each time using a newly tuned PID, until the response meets the time response specifications. The simulation then retrieves the next set of plant parameters and repeats the entire process.

**Phase Margin Tuning Simulation**

The phase margin tuning simulation consists of 4 steps: time response calculation, critical gain and frequency calculation, ITAE estimation process and PID tuning. The time response calculation is performed as described previously. The critical gain and frequency are calculated directly from the actual plant parameter values\(^{14}\). The estimation of the A, B and C coefficients (equations 4.13-4.15) are performed using previous simulation results. Once the coefficients are determined, the phase margin which minimizes the estimated ITAE for the particular critical gain and frequency is used to select the next set of PID parameters (equations 4.9-

\(^{14}\) It is assumed that a method such as Astrom and Hagglund's relay-control or any other appropriate method could be used in an actual system.
4.11)\textsuperscript{15}. The simulation then repeats the process for the next set of plant parameters.

**Phase Margin Tuning Results**

Several different sequences of plant parameter values were used during testing of the phase margin tuning method. A representative set of results, using the plant parameter sequence shown in figure 17, will be described in the following paragraphs.

The ITAE values calculated for each system response were found to vary widely as the phase margin was varied. Referring to figure 18, it is seen that the estimation and ITAE prediction process resulted in large changes in the phase margin and a resulting large change in the ITAE values. As more information was acquired, after more system tests, the phase margin is seen to vary less. As a result, the ITAE values exhibit less drastic changes, and in general, become smaller (i.e. are minimized).

\textsuperscript{15} The actual phase margin values used were constrained to be between $+30^\circ$ and $+60^\circ$. 
Subsequent tests of the same sequence of plant parameter values, while continuing to update the estimation process, show that the phase margin values become quite steady, and the ITAE values are much improved. Figure 19 shows the results of the third time through the sequence.
Phase Margin Tuning/Estimation Problems and Solutions

The phase margin tuning/estimation procedure worked rather well, producing stable results while minimizing the ITAE performance index. It is worth noting, however, that the phase margin tuning method by itself (e.g., using a constant value for the phase margin) gives stable system responses, but does not minimize the ITAE values. The purpose of the estimation procedure is to additionally minimize the ITAE value. Thus, if it is desired to only provide stable system responses, the estimation procedure need not be performed. In that instance, the phase margin must be chosen a priori, and the resulting ITAE is predetermined.

If it is also desired to minimize the ITAE, an estimation/minimization scheme must be used. The method used in this study performed reasonably well, but could be improved. Higher order estimation models could be used, as could a "forgetting-factor" (Astrom and Anton [11]). In this study, however, the phase margin tuning/estimation scheme is used to provide a comparison for the expert system approach results, and so further improvements in its implementation will not be addressed.
Tuning the PID using the expert system amounts to finding the step response to a given system and finding an appropriate way to tune the PID. The initial values for the PID parameters were chosen somewhat arbitrarily from results of past experience tuning the PID manually. The initial setting is important primarily in the sense of speeding up the convergence to a tuned condition. Since the simulation relies on the notion that the plant parameters change "slowly", an initially poorly tuned PID unnecessarily slows the tuning for the first set of plant parameters. If the PID is tuned too poorly initially, the tuning does not always converge (see "Expert System Problems and Solutions").

It should be noted that the expert system defined throughout this paper made no mention of how much to change the PID parameters. The expert system only gave possible "directions" to tune the PID. In the simulation, the amount of change for the PID parameters was made proportional to the sum of the squares of the differences between the nominal desired response characteristics and the actual response characteristics. This was found to be
quite acceptable for simulation purposes\textsuperscript{16}.

\textbf{Expert System Results}

The system simulation with expert system tuning began with the observation of the step response of an untuned system. From the results, the expert system determined whether the system performed within specifications (i.e. Max1, Min1, Tr and Ts within specification allowances). If the response was within specifications, the simulation would proceed to the next set of random plant parameters. If not within specifications, the expert system would change the PID parameters appropriately until the response was within specifications, then the simulation would proceed to the next set of plant parameters and repeat the process.

Since it is unknown how well tuned the PID will be when a new set of plant parameters is tested, it is apparent that the tuning process for each set may require

\textsuperscript{16} Note that this method does not guarantee convergence. More elaborate schemes which would help speed the convergence could be employed. These schemes would have the flavor of many root-finding algorithms. The aforementioned method, however, is relatively simple and performed rather well for the problem at hand.
many iterations to converge to a tuned solution. This was seen to be the case. Some parameter sets required no tuning while others required as many as 25 iterations to converge.

Refer to figure 20 in which the values of \textit{Max1}, \textit{Min1}, \textit{Tr} and \textit{Ts} (defined in figure 9) are plotted sequentially as the simulation progresses (again using the plant parameter sequence of figure 17). The vertical dotted lines represent that point when tests on a new set of plant parameters were begun. The horizontal dotted lines represent the specification limits (range) for each plotted value.
Figure 20. Max1, Min1, Tr and Ts results (Expert system)

Figure 21 shows how the PID parameters $K_{PID}$, $K_I$, $K_D$, $\zeta_{PID}$ and $\omega_{nPID}$ varied during the same sequence. Comparing figures 20 and 21, it can be seen that the system response values varied as the PID parameters were changed, until that point when all of the response values were within specifications, at which point a new set of plant parameters was tested.
It is interesting to note that the values of Max1 and Min1 varied less than did Tr and Ts. This might lead one to conclude that these values are more robust with respect
to the tuning process. But this is not the case. It must be remembered that the tuning process is based on the specification limits for each parameter. Max1 and Min1 have the tightest tolerances so the expert system allows these values to vary the least. Conversely, Tr and Ts have rather loose tolerances so they are allowed to vary in a greater amount.

Several of the plant parameter sets required little or no tuning. In these cases, the system response parameters were within or close to the specifications at the beginning of their tuning sequence. Several sets, however, required a rather extensive amount of tuning, i.e., several iterations. One such case is the third set of plant parameters. The results are shown in more detail in figures 22 and 23.
Figure 22. Max1, Min1, Tr and Ts results.
(Expert system, third plant parameter set)
Figure 23. $K_p$, $K_i$, $K_d$, $\zeta_{\text{PID}}$, and $\omega_{\text{PID}}$ variation. (Expert system, third plant parameter set)

At the start of this parameter set, $Max_1$ and $Ts$ are too large. The expert system determines that increasing $K_d$ (which automatically changes both $\zeta_{\text{PID}}$ and $\omega_{\text{PID}}$)
while holding \(K_{\text{pid}}\) and \(K_{\text{i}}\) constant would improve the system response. The resulting response did, indeed, decrease \(M_{\text{x}}\), but also increased \(T_{\text{s}}\). \(M_{\text{n}}\) and \(T_{\text{r}}\) were found to also increase. In the next iteration, \(M_{\text{x}}\), \(T_{\text{r}}\) and \(T_{\text{s}}\) are all too large, so \(K_{\text{i}}\) is increased. The process continues, increasing \(K_{\text{d}}\), decreasing \(W_{\text{n}}\) and increasing \(Zeta_{\text{pid}}\). At this point, all of the values are out of specification, but a large increase in \(K_{\text{pid}}\) brings them all to within specifications except \(M_{\text{n}}\), which is too small. From this point on, \(K_{\text{pid}}\) is decreased slowly several times until \(M_{\text{x}}\) becomes close to its minimum value. An increase in \(W_{\text{n}}\) then brings all of the values to within specifications.

The question arises as to why the expert system changed the parameters in this particular order. The reasoning is that the expert system categorizes the system response as underdamped, overdamped or critically damped. From this categorization, the tuning goals and the other system response parameters (\(T_{\text{max}}, T_{\text{min}}, M_{\text{x2}}\)), the tuning method is determined. For example, consider the first iteration. \(M_{\text{x}}\) and \(T_{\text{s}}\) are too large. The expert system determines that the system is underdamped, so an increase in \(K_{\text{pid}}\) would be inappropriate. However, an increase in \(K_{\text{d}}\) should decrease \(M_{\text{x}}\), increase \(M_{\text{n}}\) and decrease \(M_{\text{x2}}\) (see tuning rules in appendix). The increase in \(M_{\text{n}}\) and decrease in \(M_{\text{x2}}\) should in turn decrease \(T_{\text{s}}\). Thus,
increasing Kd is chosen as a possible solution. Succeeding iterations repeat the process in a similar manner.

**Expert System Problems and Solutions**

It was seen in the previous example that the expert system tuning method can sometimes be slow to converge. Part of the reason is that the expert system used does not give any indication of the amount by which the system will change for a given PID parameter change. A more involved structure which defines magnitudes as well as directions would improve the rate of convergence.

It was also seen that occasionally the system response will not change in the direction which was predicted. This is due to limitations in the present tuning rules. The rule set can, however, be expanded to include extra conditionals in rules which are known to not apply to all situations. This is done to a certain degree with the damping categorization. Any number of other categorization schemes could be developed and incorporated into the rules.

Occasionally, a set of tuning conditions would lead to a situation in which there was no solution. In this case, the expert system proceeded to consider only the worst parameter, that is, the response parameter which was
the farthest from nominal, and tune appropriately. This sometimes resulted in the other response parameters being changed in the wrong directions. An alternative method would be to correctly change as many response parameters as possible. This would, however, add considerable complexity to the expert system.

The expert system developed changes one PID parameter at a time (neglecting the relationship between Ki/Kd and ZetaPid/WnPid). A more involved expert system might consider changing more than one PID parameter at a time. This would necessitate a scheme which could predict future values of the response parameters (instead of trends) and thus predict future response characteristics (such as damping). This process could then be repeated from the new predicted condition. It is possible that the resulting scheme could tune the PID in one iteration, but this would require a very extensive increase in the capacities of the present expert system, as well as the inclusion of an estimation process (such as that used in the phase margin tuner).

A final problem to be noted deals with large changes in system parameters. It was observed that overly large changes in plant parameters resulted in a decrease in tuning performance. This is due to the fact that the tuning rules were developed assuming small changes in plant parameters. Thus, there exists a region of
convergence around a nominal solution. If the system is too far out of tune, the tuning method becomes one of "guessing" instead of predicting. In these cases, the tuning process either oscillated or diverged.
IX. CONCLUSIONS

This chapter begins with a comparison of the results of the phase margin and expert system tuning methods. It follows with a description of possible extensions to the expert system developed in this study. The chapter ends with some concluding remarks.

**Expert System Tuning versus Phase Margin Tuning**

It has been shown that both the phase margin tuning and expert system tuning methods perform reasonably well in adapting the system to environmental or plant parameter changes. The phase margin tuner provides system stability and minimizes the ITAE performance index. The expert system tuner provides a means for meeting specific system response criteria and, in doing so, minimizes the ITAE performance index to some degree. A plot comparing the ITAE results for both methods is shown in figure 24.
The expert system tuning method thus produces results comparable to that of the more traditional phase margin tuner. An advantage of using the expert system approach, however, is the use of response criteria which are more intuitive than the idea of the phase margin. Thus, the system designer can have a good "feel" for what the system response will be.

It is apparent from the large number of rules shown in the appendix that a good deal of effort must be expended in developing an expert system. For this reason, the use of an expert system is probably not worthwhile for a system as simple as that studied herein. However, for the reasons described in chapter V, as a system becomes
more complicated in terms of its control logic, the expert system approach can greatly reduce the total number of logic statements needed. Furthermore, the development of these statements (rules) is simplified since they are more intuitive to the system designer.

Extensions of Expert System

It follows from the discussion that an expert system may be placed at many levels in an overall system, from a general level to a very specific level. The example presented in this paper is very specific; the expert system performed a basic function, i.e., PID tuning. In a more complicated system composed of, for example, several PID's, it may be desired to include an expert system at a "supervisory" level. This level might determine the actual specification requirements for the individual PID's based on overall system criteria. The individual PID's may be tuned by the expert system or perhaps a more conventional method.

For example, consider an aircraft's roll axis. During cruise conditions, it might be desired to have an overdamped, slow roll response to alleviate unnecessary aircraft accelerations. In this case, the inner control loop, perhaps composed of a PID controller, could be adjusted accordingly. If the aircraft is flying an
automatic approach to land, however, the roll response needs to be much faster to allow the aircraft to perform the necessary maneuver. An expert system at a high level might determine what the overall system requirements are for the particular conditions and direct lower level processes to perform the needed tuning operations.

Consider another case, that of a system consisting of multiple PID controllers (see figure 25).

![Diagram of Multiple PID Controllers](image)

Figure 25. Multiple PID controllers.

Suppose that certain requirements are placed on the responses for $Y_1$ and $Y_2$. It is obvious that these responses are not independent. For example, if the response at $Y_1$ is slow, it may be difficult to enforce a fast response at $Y_2$. An expert system overseeing the system might detect this conflict and adjust the requirements appropriately.

As mentioned earlier, tuning of a complicated system involves much more than selection of the PID parameters. An extended expert system might address such issues as switching between manual tuning, automatic tuning and estimation processes, transient protection, non-linear effects, etc.

An expert system with "intelligence" might also
incorporate the ability to learn. In this context, learning means the ability to verify the validity of its present set of rules, and add, delete or modify rules as necessary. To accomplish this, any rules which produce incorrect results could be deleted or extra conditions could be incorporated in the rules. Charniak and McDermott discuss methods of incorporating measures of probability in the heuristic rules. As the system "learns", these probabilities could be updated appropriately.
Concluding Remarks

It has been shown how an expert system approach can be applied to a simple adaptive regulator tuning problem. It was shown how heuristic reasoning, similar to the way which a human expert reasons, can be used to adaptively tune a simple PID regulator in an unknown, changing environment.

For more complicated systems, the expert system developed in this study could be extended to incorporate additional heuristics to handle those aspects of the control problem which are difficult to model analytically. The extended reasoning ability could be used to aid in the operation of complicated control systems by providing consistent tuning methods and, in the process, providing a clear description of the reasoning behind the tuning decisions.
BIBLIOGRAPHY


APPENDIX
A.1 Least Square Error Estimation Method

Consider a scalar function \( Y = A + BX + X^T CX \), where

\[
A = a, \quad \text{(A1.1)}
\]
\[
B = [b_1 \ b_2 \ \ldots \ b_n], \quad \text{(A1.2)}
\]
\[
X = [x_1 \ x_2 \ \ldots \ x_n]^T \quad \text{(A1.3)}
\]

and

\[
C = \begin{bmatrix}
    c_{11} & c_{12} & c_{13} & \ldots & c_{1n} \\
    0 & c_{22} & c_{23} & \ldots & c_{2n} \\
    \vdots & \vdots & \ddots & \vdots \\
    0 & 0 & \ldots & 0 & c_{nn}
\end{bmatrix} \quad \text{(A1.4)}
\]

Given \( m \) sets of data points: \( x_1(1), x_1(2), \ldots, x_1(m), x_2(1), x_2(2), \ldots, x_2(m), \ldots, x_n(1), x_n(2), \ldots, x_n(m) \), \( y(1), y(2), \ldots, y(m) \), it is desired to find \( A, B \) and \( C \) such that the sum of the squared errors is a minimum. In other words, minimize

\[
S = \sum_{i=1}^{m} e(i)^2 = \sum_{i=1}^{m} [y(i) - Y(i)]^2, \quad \text{(A1.5)}
\]

where \( Y(i) = A + BX(i) + X^T(i)CX(i) \) \( \text{(A1.6)} \)

and \( X(i) = [x_1(i) \ x_2(i) \ \ldots \ x_n(i)]^T. \) \( \text{(A1.7)} \)

To minimize \( S \) with respect to each coefficient of \( A, B \) and \( C \), the derivative of \( S \) with respect to each coefficient can be taken and set equal to zero. Since each coefficient occurs once and of the first order in \( e(i) \), the derivatives are as follows:
\( \frac{dS}{da} = \Sigma 2 \, e(i) \, \frac{de(i)}{da} = 0, \quad (A1.8) \)

\( \frac{dS}{db_j} = \Sigma 2 \, e(i) \, \frac{de(i)}{db_j} = 0, \quad j=1,2,\ldots,n \quad (A1.9) \)

\( \frac{dS}{dc_{jk}} = \Sigma 2 \, e(i) \, \frac{de(i)}{dc_{jk}} = 0, \quad j=1,2,\ldots,n \quad k=j,j+1,\ldots,n \quad (A1.10) \)

where each summation is from \( i=1 \) to \( m \). Since

\[
    e(i) = y(i) - [a + b_1x_1(i) + b_2x_2(i) + \ldots + b_nx_n(i) +
\]
\[
    + c_{11}x_1(i)x_1(i) + c_{12}x_1(i)x_2(i) + \ldots + c_{1n}x_1(i)x_n(i) +
\]
\[
    + c_{22}x_2(i)x_2(i) + \ldots + c_{2n}x_2(i)x_n(i) +
\]
\[
    + \ldots + c_{nn}x_n(i)x_n(i) ],
\]

then

\( \frac{de(i)}{da} = -1, \quad (A1.12) \)

\( \frac{de(i)}{db_j} = -x_j(i), \quad (A1.13) \)

\( \frac{de(i)}{dc_{jk}} = -x_j(i)x_k(i), \quad k \geq j. \quad (A1.14) \)

Thus a set of equations are formed, one for each derivative. Combining A1.8, A1.9, A1.10, A1.12, A1.13 and A1.14, then

\( \frac{dS}{da} = -2\Sigma e(i) = 0, \quad (A1.15) \)

\( \frac{dS}{db_j} = -2\Sigma e(i)x_j(i) = 0, \quad (A1.16) \)

\( \frac{dS}{dc_{jk}} = -2\Sigma e(i)x_j(i)x_k(i) = 0, \quad k \geq j, \quad (A1.17) \)

where each summation is from \( i=1 \) to \( m \).
Using equation A1.11,

\[
\frac{dS}{da} = 0 \implies \Sigma e(i) = 0 = \Sigma [y(i) - Y(i)]
\]

\[
\implies \Sigma y(i) = \Sigma Y(i)
\]

\[
= \Sigma [a + b_1 x_1(i) + b_2 x_2(i) + \ldots + b_n x_n(i)
\]

\[
+ c_{11} x_1(i)x_1(i) + c_{12} x_1(i)x_2(i) + \ldots + c_{1n} x_1(i)x_n(i)
\]

\[
+ c_{22} x_2(i)x_2(i) + \ldots + c_{2n} x_2(i)x_n(i)
\]

\[
+ \ldots
\]

\[
+ c_{nn} x_n(i)x_n(i)],
\]

\[\text{(A1.18)}\]

\[
\frac{dS}{db_j} = 0 \implies \Sigma e(i)x_j(i) = 0
\]

\[
\implies \Sigma y(i)x_j(i) = \Sigma x_j[a + b_1 x_1(i) + b_2 x_2(i) + \ldots + b_n x_n(i)
\]

\[
+ c_{11} x_1(i)x_1(i) + c_{12} x_1(i)x_2(i) + \ldots + c_{1n} x_1(i)x_n(i)
\]

\[
+ c_{22} x_2(i)x_2(i) + \ldots + c_{2n} x_2(i)x_n(i)
\]

\[
+ \ldots
\]

\[
+ c_{nn} x_n(i)x_n(i)],
\]

\[j = 1, 2, \ldots, n\]

\[\text{(A1.19)}\]

\[
\frac{dS}{dc_{jk}} = 0 \implies \Sigma e(i)x_j(i)x_k(i) = 0
\]

\[
\implies \Sigma y(i)x_j(i)x_k(i) = \Sigma x_j[a + b_1 x_1(i) + b_2 x_2(i) + \ldots + b_n x_n(i)
\]

\[
+ c_{11} x_1(i)x_1(i) + c_{12} x_1(i)x_2(i) + \ldots + c_{1n} x_1(i)x_n(i)
\]

\[
+ c_{22} x_2(i)x_2(i) + \ldots + c_{2n} x_2(i)x_n(i)
\]

\[
+ \ldots
\]

\[
+ c_{nn} x_n(i)x_n(i)],
\]

\[j = 1, 2, \ldots, n\]

\[k = j, j+1, \ldots, n\]

\[\text{(A1.20)}\]

where each summation is from i = 1 to m.
Defining the following,

\[ A_i = \Sigma x_i(a), \]  
\[ B_{ij} = \Sigma x_i(a)x_j(a), \]  
\[ C_{ijk} = \Sigma x_i(a)x_j(a)x_k(a), \]  
\[ D_{ijkn} = \Sigma x_i(a)x_j(a)x_k(a)x_n(a), \]  
\[ E = \Sigma y(a), \]  
\[ F_i = \Sigma x_i(a)y(a), \]  
\[ G_{ij} = \Sigma x_i(a)x_j(a)y(a), \]  

where each summation is from \( \alpha = 1 \) to \( m \), it follows that the set of equations has the following form:

\[
\begin{bmatrix}
A_1 & A_2 & \cdots & A_n \\
B_{11} & B_{12} & \cdots & B_{1n} \\
B_{12} & B_{22} & \cdots & B_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
B_{1n} & B_{2n} & \cdots & B_{nn} \\
C_{11} & C_{12} & \cdots & C_{1n} \\
C_{12} & C_{22} & \cdots & C_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
C_{1n} & C_{2n} & \cdots & C_{nn} \\
D_{111} & D_{112} & \cdots & D_{11n} \\
D_{112} & D_{122} & \cdots & D_{12n} \\
\vdots & \vdots & \ddots & \vdots \\
D_{11n} & D_{12n} & \cdots & D_{nnn}
\end{bmatrix}
\begin{bmatrix}
b_1 \\
b_2 \\
\vdots \\
b_n
\end{bmatrix}
= 
\begin{bmatrix}
F_1 \\
F_2 \\
\vdots \\
F_n
\end{bmatrix}
\]

Noting that the matrix in equation A1.28 is symmetrical, a Gaussian elimination method with partial pivoting can be used to solve for the \( A \), \( B \) and \( C \) coefficients. A variation of "Subroutine ELIM" in Gerald [20] was used in the actual simulation to solve for \( A \), \( B \) and \( C \).
A.2 Expert System Tuning Rules

Max1 Rules:
(if (or
(kpid increases) (ki increases)
(kd decreases)
(zeta decreases) (wn increases))
then (max1 increases)).

(if (or
(kpid decreases) (ki decreases)
(kd increases)
(zeta increases) (wn decreases))
then (max1 decreases)).

Tmax Rules:
(if (or
(kpid decreases)
(and (ki increases)(critically-damped))
(and (ki increases)(under-damped))
(and (ki decreases)(over-damped))
(and (kd decreases)(critically-damped))
(and (kd decreases)(under-damped))
(and (ζ increases)(critically-damped))
(ωn increases))
then (tmax increases)).

(if (or
(kpid increases)
(and (ki decreases)(critically-damped))
(and (ki decreases)(under-damped))
(and (ki increases)(over-damped))
(and (kd increases)(critically-damped))
(and (kd increases)(under-damped))
(and (ζ decreases)(critically-damped))
(ωn decreases))
then (tmax decreases)).
Mini Rules:

(if (or
  (and (kpid decreases) (critically-damped))
  (and (kpid decreases) (under-damped))
  (and (ki decreases) (under-damped) (unstable))
  (and (ki increases) (under-damped) (stable))
  (and (kd decreases) (critically-damped))
  (and (kd increases) (under-damped))
  (and (ω increases)
     (and (kpid increases) (over-damped))
     (and (ki increases) (over-damped)))
then (mini increases).

(if (or
  (and (kpid increases) (critically-damped))
  (and (kpid increases) (under-damped))
  (and (ki increases) (under-damped) (unstable))
  (and (ki decreases) (under-damped) (stable))
  (and (kd increases) (critically-damped))
  (and (kd decreases) (under-damped))
  (and (ω decreases)
     (and (kpid decreases) (over-damped))
     (and (ki decreases) (over-damped)))
then (mini decreases).

Tmin Rules:

(if (or
  (and (kpid decreases) (critically-damped))
  (and (kpid decreases) (under-damped))
  (ki increases)
  (and (kd decreases) (critically-damped))
  (and (kd decreases) (under-damped))
  (and (ω increases)
     (and (kpid increases) (critically-damped))
     (and (ki increases) (over-damped)))
then (tmin increases).

(if (or
  (and (kpid increases) (critically-damped))
  (and (kpid increases) (under-damped))
  (ki decreases)
  (and (kd increases) (critically-damped))
  (and (kd increases) (under-damped))
  (and (ω decreases)
     (and (kpid decreases) (critically-damped))
     (and (ki decreases) (over-damped)))
then (tmin decreases)).
Max2 Rules:
(if (or (kpid increases)
    (and (ki increases) (critically-damped))
    (and (ki decreases) (under-damped))
    (and (kd decreases) (critically-damped))
    (and (kd decreases) (under-damped))
    (and (\(\zeta\) decreases) (critically-damped))
    (\(\omega_n\) increases))
then (max2 increases)).

(if (or (kpid decreases)
    (and (ki decreases) (critically-damped))
    (and (ki increases) (under-damped))
    (and (kd increases) (critically-damped))
    (and (kd increases) (under-damped))
    (and (\(\zeta\) increases) (critically-damped))
    (\(\omega_n\) decreases))
then (max2 decreases)).

Tr Rules:
(if (or (kpid decreases)
    (and (ki decreases) (critically-damped))
    (and (ki decreases) (over-damped))
    (and (kd decreases) (under-damped))
then (tr increases)).

(if (or (kpid increases)
    (and (ki increases) (critically-damped))
    (and (ki increases) (over-damped))
    (and (kd increases) (under-damped))
then (tr decreases)).

Ts Rules:
(if (and (max2 increases) (tmin increases))
then (ts increases)).

(if (or (and (max2 decreases) (tmin decreases))
    (and (max2 decreases) (min1 increases)))
then (ts decreases)).
Damping Rules:
(if (and (max1 < 1.02) (min1 > 0.98) (max2 ≤ Max1) (tmax < 30))
then (critically-damped)).

(if (and (max1 ≥ 1.02) (min1 ≤ 0.98))
then (under-damped)).

(if (and (max1 < 0.98) (tmax > 30) (max2 < 1.05))
then (over-damped)).

(if (and (not (over-damped)) (not (under-damped))
(not (critically-damped))
then (damping-undefined)).

Stability Rules:
(if (max2 ≤ max1)
then (stable)).

(if (max2 > max1)
then (unstable)).

(if (over-damped)
then (stable)).

Response Tolerances:
(if (and (and (max1 ≥ 1.0)(max1 ≤ 1.1))
(and (min1 ≥ 0.90)(min1 ≤ 1.0))
(and (tr ≥ 5)(tr ≤ 15))
(and (ts ≥ 10)(ts ≤ 60))
then (PID tuned)).