A SURVEY OF METHODS FOR THE SIMULATION OF CONTINUOUS SYSTEMS

Submitted for the degree of Master

of Philosophy at the

University of Aston in Birmingham.

RICHARD BERKELEY ARTHUR

April 1976

SUMMARY

The objectives of the project were to review the range of methods available for the computer simulation of continuous systems and then to apply these methods to a particular system. The range of methods include those using analogue, parallel-logic and digital hardware and the use of each method is discussed with reference to programming time, solution time and ease of method. Full descriptions of the solutions and the results obtained are given

A critical appraisal of each method then follows and comparisons between the simulation solutions and the analytical solution are discussed with reference to any errors occurring during the simulations from which conclusions are formulated as to the most appropriate methods.

A set of appendices describes the computer configurations, details the mathematical solution and tabulates the digital programs and results in full.

CONTENTS

Appendices

1.0 INTRODUCTION

1.1 General Background

Many engineering problems require calculations that call for sophisticated methods of solution due to their increasing complexity as the size and nature of the problems increase. For many years now, it has been possible to use a variety of computing aids for solving these problems. These aids may be divided into two main categories:

a) the analogue computer

and b) the digital computer.

The former is basically a device for solving dynamic time varying problems where the digital computer is essentially to be used for static problems. With the increase in the need for machines to solve both types of problem simultaneously, the development of joint systems has occurred giving rise to hybrid computers where the two types given above have been joined together by a suitable interface.

With so many different methods available, it was decided that a single problem would be solved using as many methods as practical choosing those methods which would most likely be available to engineers without recourse to expensive bureau facilities. Although some methods were relatively easy to implement, it was found that these methods had certain drawbacks which are discussed in the relevant sections of the text.

1.2 The Problem to be Solved

The problem was defined as a typical position control system with a first order lag term in the feedback. The general block diagram for such a system is shown in figure 1.2.1.

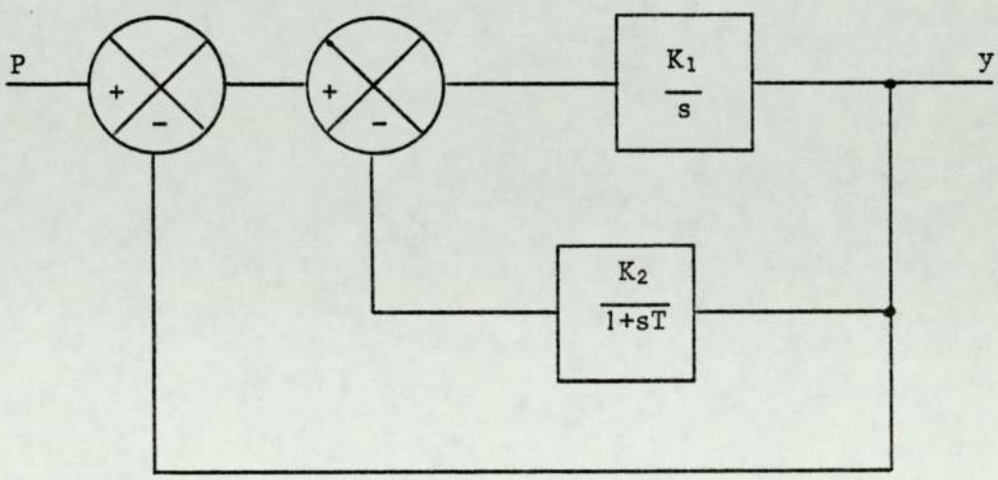

Figure 1.2.1.

The analysis of this block diagram together with the set of differential equations that describe the system are given in chapter 3 where the theoretical solution is given.
In order to make the problem non-linear, it was decided that the input function, P, should be a pulse since this gave a reasonable theoretical solution but was not as trivial as the simple step input.*
Having verified the most suitable method, it would then be possible to replace P by a variety of functions whose theoretical solution would be very time-consuming or impossible to evaluate.

* It should be noted that it is the criteria introduced on page 14 which gives rise to the non-linearity.

The methods used to solve this problem are summarized as being by

a) The Parallel-Logic Computer (incorporating the solution by a pure analogue computer),

b) The Digitally Controlled Hybrid Computer,

c) A simulation language using a large digital computer in batch mode ,

d) An interactive simulation language .

Additionally, the problem has been solved theoretically by two optimisation methods to validate the results of the various simulations and to give a basis for comparisons of the methods.

1.3 Summary of the Historical Background

The present techniques used for computer simulation have their origins in the necessity for predictor-corrector techniques in anti-aircraft problems during World War II. Up to this time, the mechanical differential analyser had been used to solve simple problems but limitations in such equipment as the ball-and-disc integrator became obvious when rapid solutions were required to solve the problems encountered in a war situation.

During this period and the years immediately following, there was intense activity in the hardware field with the introduction of stablized high gain amplifiers which could be used to perform the basic integration operations as typified in an analogue computer. With the use of semi-conductors becoming widespread during the sixties, it became possible to increase the speed of solution to many times real time in spite of the increasing complexity of problems due almost entirely to the parallel nature of the solution. Coupled with this high speed capability was the introduction of logic elements into the analogue computer to form the parallel logic computer which then enabled the user to preprogram changes in parameters and to utilize simple optimisation routines.

A different development was to couple together an analogue and a digital computer to form a full hybrid computer in which each part of the computer was responsible for solving that section of the problem most suited to it. The disadvantage of this type of configuration was that it was necessary to include a complex interface between the two computers to convert the analogue signals (in continuous voltages) into digital variables (in discrete pulses) and vice versa. The slow speed of this data transfer caused the analogue computer to be idle (in the Hold mode) for much of the solution time and made the configuration inefficient. It is only since high speed digital computation with microsecond arithmetic has been available that this method of simulation has been feasible although highly non-linear problems have been solved using hybrid systems for some ten years.

This increase in speed of digital computation has resulted in a more recent development - that of digital simulation languages, in which the complete system is specified and solved without reference to an analogue computer and its complex interface with the digital hardware.

Thus the trend has been away from the pure analogue solution, through the parallel-logic approach and the more complex full hybrid computer to the modern simulation language using a digital computer only.

The development of the "state-of-the art" is discussed more fully in chapter 2.

2.0 THE DEVELOPMENT OF THE STATE-OF-THE ART

2.1 The Mechanical Differential Analyser (1,2)

The idea of a large general purpose automatic calculator was first postulated by Babbage (3) who conceived an "analytical engine" whilst at Cambridge during the 1830's. Although the hardware for such a machine was not yet available, he developed the theoretical model in his book together with a less sophisticated system called a "difference engine".

These ideas were taken a step further by Lord Kelvin and his brother, James Thomson who constructed the first mechanical integrating device (4) but it was left to Bush (5) to design and build the first differential analyser at the Massachusetts Institute of Technology, one hundred years after Babbage's original conception.

The differential analyser consists of a number of units interconnected by means of shafts and gears. Each unit performs one of the operations normally found in an analogue computer and the rotation of each shaft indicates the change in the quantity of each variable (6). Crank in his book (1) describes the various parts of these analysers and discusses their accuracy which in most cases will not exceed one part in 1000. The overall size and cost of such machines was found by Hartree (2) to be virtually prohibitive for larger problems and it was because of the inherent inaccuracies (due to mechanical backlash) coupled with their bulk and cost that eventually forced the mechanical differential analyser to become obsolete.

2.2 The Analogue Computer

Early attempts at producing an electrical system analogous to a set of equations used network circuit theory which implemented the laws of Kirchoff to produce sets of simultaneous equations, the elements in the network being analogous to the co-efficients of the equations.

When it was realised that, by using a.c. circuitry with resistances, capacitances and inductances, the operation of integration could be performed, then elementary differential equations could be solved, similarly to the much more cumbersome differential analyser.

It was during the second world War that these network analysers were developed to great effect in the Predictor-Corrector type of control system needed for anti-aircraft weapons although the hardware still lagged behind the theory.

In 1950, Goldberg (7) at RCA introduced a method of stabilizing the amplifiers used in these early analogues, first developed by Lovell at the Bell Telephone Laboratories and it was from these early investigations that the modern analogue computer has been developed.

The modern analogue computer consists of a series of high stability amplifiers which perform the necessary mathematical operations of integration, summation, inversion, multiplication and division, function generation etc. It is not proposed to give detailed accounts of these units as they are fully described elsewhere (8,9).

Modern analogue computers are designed for highspeed repetitive operation performing many "runs" in a second and hence allowing the user to cover many input possibilities in a short time. Often the type of problem being solved required iterative computation or optimization and for this reason the next type of computer was developed.

2.3 The Parallel-Logic Computer

The parallel-logic computer consists of a modern high speed analogue computer to which has been added a section of logic units which fall into two categories, (a) the control section and (b) the algebraic section.

The control section is used to determine the operating modes of those analogue units which are dependent on timing e.g. integraters, switches, storage units and also to generate logic signals for the algebraic section which consists of gates, bistables, registers etc. which simulate the Boolean algebra expressions representing the combination of the various logic signals.

As in section 2.2 above, it is not proposed to enumerate all the different units since these may be found in the relevant literature (10,11).

The parallel-logic computer has now superceeded the analogue computer as the standard machine for solving these dynamic problems and it is using a parallel-logic computer that one method of solution is investigated in chapter 4.

2.4 The Digital Computer

Apart from the devices postulated by Babbage (3), the first large scale automatic digital computer was developed by IBM and installed at Harvard University (12). This computer was a combination of mechanical counters and electromagnetic clutches controlled by relays and was referred to as the Harvard Mark I Calculator.

Further developments were made at Harvard before an entirely electronic machine was produced at the University of Pennsylvania for ballistics research. This was the ENIAC (13) and was capable of adding two quantities in 0.2 milliseconds. This computer was further developed by IBM who produced the Selective Sequence Electronic Calculator (14) which was the final development in this initial phase.

Following the adaption of an all electronic calculating machine, coupled with the introduction of semi-conductors and transistors, the development of digital computers became very rapid, successive 'generations' of computers being typified by the following list which is by no means complete and is only intended to give the reader an idea as to where the more well-known computers appear in the development programme:

- i) The English Electric DEUCE
- ii) The Elliott 803 and Ferranti Mercury
- iii) The IBM 1620 and Elliott 503
- iv) The IBM 360 and ICL 1900 series
- v) The IBM 370 series

2.5 The Hybrid Computer

The hybrid computer was developed as a result of a need for a complete machine capable of solving large sets of differential equations but also having the back-up facility of a large 'number-cruncher' for function evaluation, analysis of results, automatic check-out of the analogue computer etc.

These hybrid computers required complex interfacing so that the continuous parallel analogue voltages could be converted into discrete sequential digital signals and vice versa. Many hybrid computers have their own language written specifically for the problem-types involved, for example MIDAS (Modified Integration Digital Analog Simulator) by the Martin Company and HYTRAN (a Hybrid Fortran) used by Electronic Associates Inc. in their early hybrid computer (10, chapters 13 and 14).

Initially it was the practice to use a large digital machine in the hybrid computer (for example, an EAI 231-RV analogue and IBM 7090 digital computers) but a recent development has been to incorporate a small high-speed digital computer (or 'mini-computer') as a more efficient and less expensive device (e.g. Solatran HS7-6D analogue interfaced to a Digital Equipment Corporation PDP8L). This system comes under the category of a digital control hybrid and is discussed further in chapter 5. Finally it has become accepted that a hybrid computer produced by a single manufacturer would be more reliable than a joint venture and it is Electronic Associates Inc. who are the main manufacturer of these systems.

2.6 Modern Simulation Techniques

With the development of large digital computers, there has been a swing away from the use of analogue/parallel-logic computer machines and their big brothers, the hybrid computer, towards the use of a digital language written specifically for simulation. About eight years ago, a specification (15) was agreed for the structure of such a language and since then general simulation languages such as CSMP (by IBM) (26) and SLAM (by ICL) (16) have been used for large scale digital simulations. These languages are based on a high level language such as FORTRAN and can thus be used by engineers and scientists who might not have the knowledge to program in a lower level language. Programs written in one of these languages however can become very cumbersome to use and also take a very long time to execute by nature of their complexity. An example of a program written in SLAM is considered in chapter 6.

The most recent development has been that of interactive simulation languages based on the use of a modern high speed mini-computer. These languages include BEDSOCS (17) and ISIS (18) and it has been possible to include a section dealing with the use of ISIS (chapter 7) in this report. The main difference in these two languages is that BEDSOCS is based on BASIC, a language universally applied by engineers and available on many large batch processors by means of terminals or on smaller mini-computers. The other interactive language ISIS is FORTRAN based although many of the facilities are duplicated in each language. This type of solution has all the versatility of a hybrid computer, is extremely quick to implement and is usually easy to learn.

2.7 Summary

It is a long way from the original conceptions by Babbage but the development has been along straightforward lines from mechanical devices, through electromechanical machines to the early all-electronic computers using thermionic components. Following the development of semi-conductors, both analogue and digital computers have become large and complex although with the introduction of relatively inexpensive mini-computers with fast interactive simulation languages, the state-of-the-art has now reached a stage where it is easy for the non-expert to become proficient at simulation in a short space of time.

3.0 THE THEORETICAL SOLUTION

3.1 Introduction

In order to compare the methods of simulation used to solve the problem, an analytical solution was obtained for one of the required criteria. It was found that the results of the interactive simulation (chapter 7) were in close agreement with this one theoretical solution and since only a minor adjustment of the interactive program was required to give the second criterion, it was considered that this justified the set of simulation results without the complication of solving the additional performance integral theoretically.

3.2 The Problem Statement

The basic block diagram given in figure 1.2.1. is repeated here for convenience.

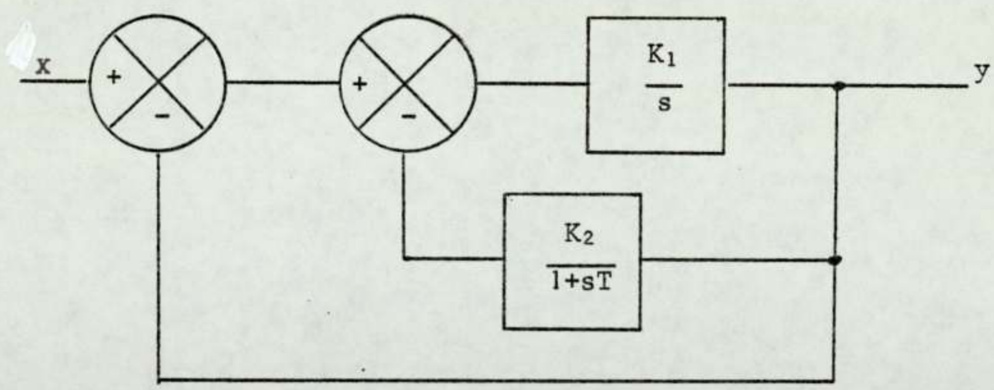

Figure 3.2.1.

From this diagram, we have the relationship

$$y = \frac{K_1}{s}\left[\{x - \frac{K_2}{1+sT}\, y\} - y\right]$$

which reduces to

$$\frac{y}{x} = \frac{K_1(1+sT)}{s^2T + s(1+K_1T) + K_1(1+K_2)} \qquad (3.2.1.)$$

In order to make the problem into a simple non-linear problem, the input function was defined as a pulse of amplitude P and duration time τ as shown in figure 3.2.2.

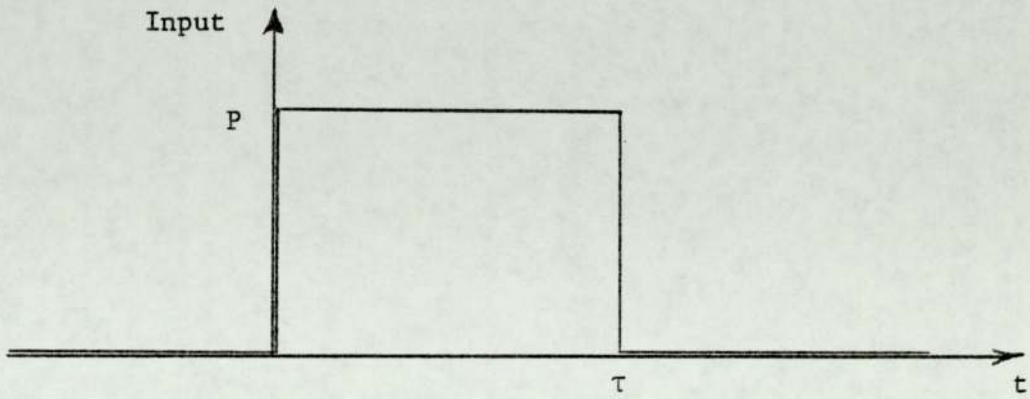

Figure 3.2.2.

It can be shown that by a suitable choice of the constants in equation 3.2.1., the system reduces to the set of equations

$$\frac{dy_2}{dt} = y_1 - P \qquad (3.2.2a)$$

$$\frac{dy_1}{dt} = -\tfrac{1}{2}y_1 - y_2 \qquad (3.2.2b)$$

where P is the input pulse, y_2 is the output function and y_1 is a subsidiary variable.

The criteria to be investigated were to minimise the integrals

$$C_1 = \int_0^\infty y_2^2 \, dt \qquad (3.2.3a)$$

and

$$C_2 = \int_0^\infty t y_2^2 \, dt \qquad (3.2.3b)$$

These criteria were chosen since if the system represents an electrical circuit then y_2^2 is a measure of the energy dissipated in the system and hence it is necessary to find the values of P and τ which minimise these two criteria. The second criterion included a weighting factor, t, to take account of the time taken for this minimum to be obtained.

It should be noted that if the pulse duration is sufficiently large then a quasi-stable solution will be obtained as shown in figure 3.2.3.

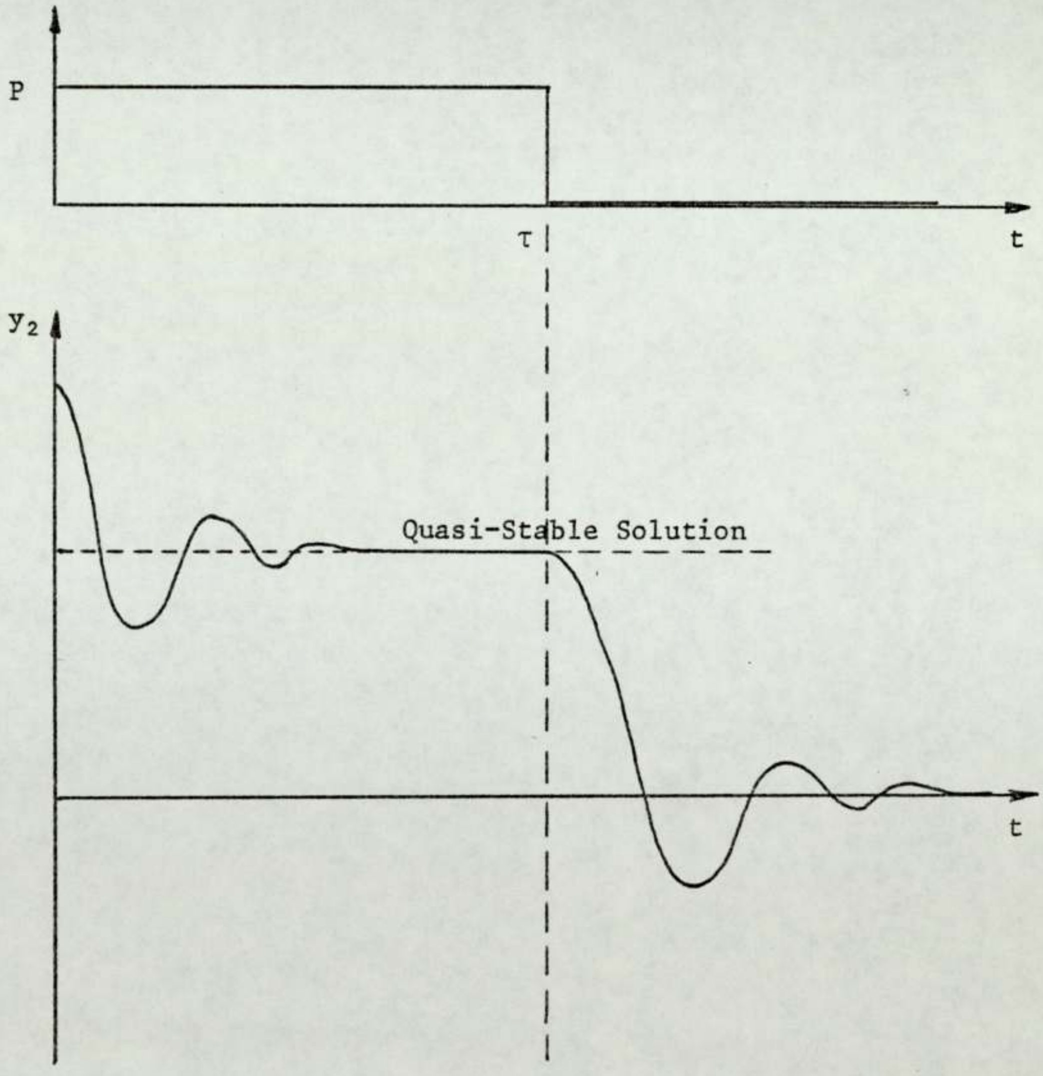

Figure 3.2.3.

Since y_2 is thus steady but non-zero for a considerable time, it follows that any criteria dependent on y_2 will not become steady until the zero steady state occurs in which case these criteria will give very large values.

3.3 The Optimisation Methods

After an initial investigation it was found that, for a constant value of P, the form of the output criterion was dependent on the pulse width τ as shown in figure 3.3.1.

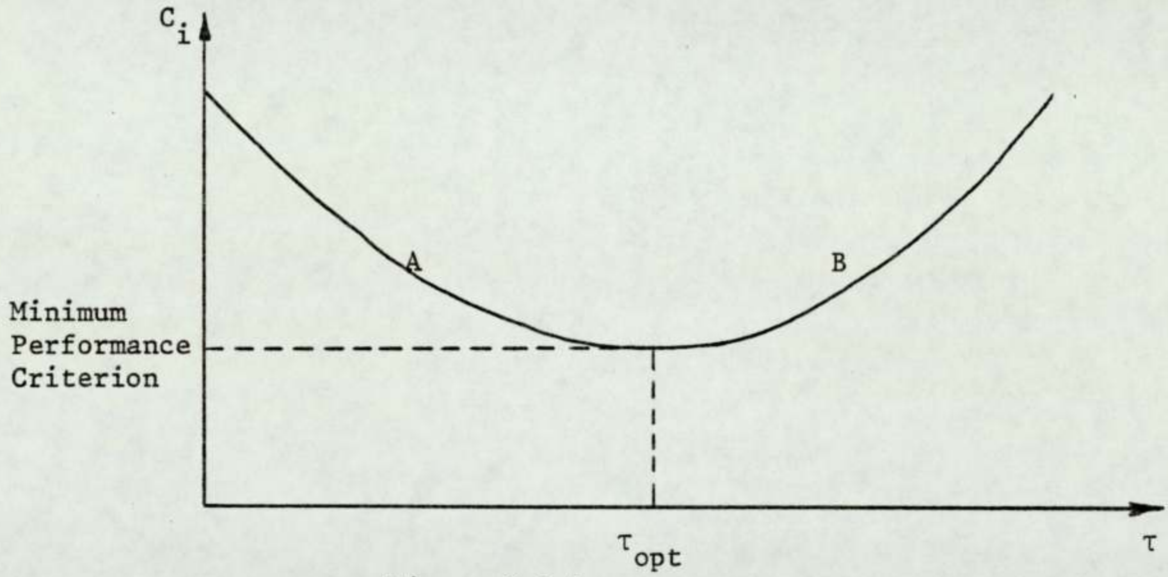

Figure 3.3.1.

Thus it was necessary to determine this minimum value by optimising the value of τ for each pulse amplitude.

Two linear search methods of optimisation were used in order to compare a simple quadratic fit with the more complex golden section method (19). In the first of these methods, an initial guess of τ was used to give a value of C_i and τ was then increased and a decision made as to whether the value of C_i was on section A or B of the results curve (figure 3.3.1). If it was on A, then τ continued to be incremented until C_i started to increase when a simple quadratic fit was made to the three points straddling the minimum. From this quadratic fit, the coordinates of the minimum were determined. This method afforded a quick estimate of the minimum but obviously was dependent on a knowledge of the system so as to correctly estimate the initial value of τ and the increment size.

The golden section method is a more useful method when the solution is unknown since it converges much more quickly than other methods for a unimodal function such as the criteria to be evaluated and also uses a simple algorithm as follows:

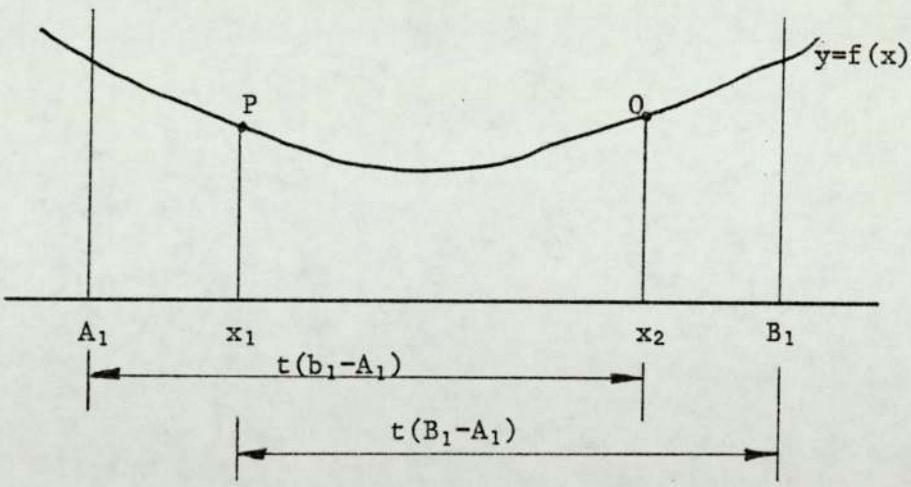

Figure 3.3.2.

Suppose it is known that the minimum value of function $y = f(x)$ lies in the range (A,B) of x values, initially chosen as (A_1,B_1).

Choose x_1 and x_2 such that

$$x_2 - A_1 = B_1 - x_1 = t(B_1 - A_1)$$

where t is to be determined, such that the minimum number of runs is taken.

The points P and Q (= $f(x_1)$, $f(x_2)$ respectively) are determined and new A and B values such that

a) if $P > Q$, then the minimum lies in the range $[x_1, B_1]$

b) if $P < Q$, then the minimum lies in the range $[A_1, x_2]$

c) if $P = Q$, then the minimum lies in the range $[x_1, x_2]$

The process is now repeated using new (A,B) values of (A_2, B_2) as defined above until $A_i = B_i$ when the optimum has been reached.

It can be shown that the value of t which optimises the process is given by $t = \frac{1}{2}(\sqrt{5} - 1)$ which is the golden mean (Appendix 3)

3.4 Evaluation of the Theoretical Solution

Consider the differential equations defining the system as given in equations 3.2.2.

$$\left.\begin{aligned} \frac{dy_2}{dt} &= y_1 - P \\ \frac{dy_1}{dt} &= -\tfrac{1}{2}y_i - y_2 \end{aligned}\right\} \qquad (3.4.1.)$$

These equations can be written in terms of Laplace transforms as follows where initially $y_2 = 1$ (so giving a normalized solution) and $y_1 = 0$ as follows

$$\left.\begin{aligned} (sY_2 - 1) &= Y_1 - \frac{P}{s}(1 - e^{-s\tau}) \\ sY_1 &= -\tfrac{1}{2}Y_1 - Y_2 \end{aligned}\right\} \qquad (3.4.2.)$$

giving

$$\left.\begin{aligned} s^2Y_2 - sY_1 &= s - P(1 - e^{-s\tau}) \\ Y_2 + (s + \tfrac{1}{2})Y_1 &= 0 \end{aligned}\right\} \qquad (3.4.3.)$$

Eliminating Y_1 from these equations, we have

$$s^2Y_2 - \frac{s}{s+\frac{1}{2}}(-Y_2) = s - P(1-e^{-s\tau}) \qquad (3.4.4.)$$

from which

$$Y_2 = \left[\frac{(s-P)(s+\frac{1}{2})}{s(s^2+\frac{1}{2}s+1)}\right] + \left[\frac{P(s+\frac{1}{2})}{s(s^2+\frac{1}{2}s+1)}\right] e^{-s\tau}$$

$$= Y_{2A} + Y_{2B}\, e^{-s\tau} \qquad (3.4.5.)$$

where Y_{2B} is such that following its inversion back into the time domain, each time parameter is replaced by $(t-\tau)$ by virtue of the $e^{-s\tau}$ term in the Laplace equation (3.4.5.) which indicates a delay function.

The complete analysis of the inversion is given in appendix number 2 and the final result for y_2 is

$$y_2 = y_{2A}(t) + y_{2B}(t-\tau)$$

$$= \left\{e^{-\frac{1}{4}t}\left[(1+\frac{P}{2})\cos\frac{\sqrt{15}}{4}t + \frac{1}{\sqrt{15}}(1-\frac{7P}{2})\sin\frac{\sqrt{15}}{4}t\right] - \frac{P}{2}\right\}H(t)$$

$$+ \frac{P}{2}\left\{1-e^{-\frac{1}{4}(t-\tau)}\left[\cos\frac{\sqrt{15}}{4}(t-\tau) - \frac{7}{\sqrt{15}}\sin\frac{\sqrt{15}}{4}(t-\tau)\right]\right\}H(t-\tau)$$

where $H(t)$ is the unit step function defined by figure 3.4.1.

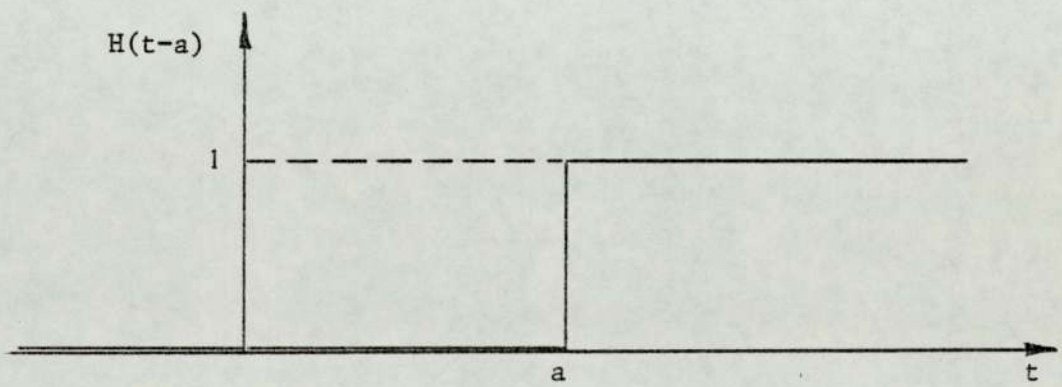

Figure 3.4.1.

In order to overcome difficulties with integrals of the form

$$\int_{o}^{\infty} \frac{P}{2}\, dt$$

we can now write y_2 as

$$y_2 = \{e^{-\frac{1}{4}t}\left[(1+\frac{P}{2})\cos\frac{\sqrt{15}}{4}t + \frac{1}{\sqrt{15}}(1-\frac{7P}{2})\sin\frac{\sqrt{15}}{4}t\right]H(t)\}$$
$$-\{\frac{P}{2}e^{-\frac{1}{4}(t-\tau)}\left[\cos\frac{\sqrt{15}}{4}(t-\tau) - \frac{7}{\sqrt{15}}\sin\frac{\sqrt{15}}{4}(t-\tau)\right]H(t-\tau)\}$$
$$-\{\frac{P}{2}\left[H(t) - H(t-\tau)\right]\} \qquad (3.4.6)$$

$$= f_1(t) - f_2(t-\tau) - f_3(T) \qquad (3.4.7)$$

where $f_1(t)$ is valid for $0 \leqslant t \leqslant \infty$

$f_2(t-\tau)$ is valid for $\tau \leqslant t \leqslant \infty$

and $f_3(T)$ is valid for $0 \leqslant t \leqslant \tau$ only.

To order to determine the values of P and τ for an optimum solution, it is now necessary to evaluate

$$C_1 = \int_{o}^{\infty} y_2^2\, dt$$

The lengthy integration process is detailed in Appendix 2 and the final result was found to be

$$C_1 = \left(\frac{19P^2 - 2P+10}{18}\right) + \tfrac{1}{4}P^2\tau$$
$$+ e^{-\frac{1}{4}\tau}\left[\left(\frac{2P-19P^2}{8}\right)\cos\frac{\sqrt{15}}{4}\tau - \left(\frac{27P^2+78P}{8\sqrt{15}}\right)\sin\frac{\sqrt{15}}{4}\tau\right]$$

FORTRAN programs were written to evaluate this function using both the golden section search method, GOLDOPT, and also the simpler quadratic-fit method, QUADOPT, described in section 3.3 above.

The FORTRAN programs are detailed in Appendices 4.1 (Golden Search) and 4.2 (Simple Method).

The Summary of the results of these programs are tabulated below

	GOLDOPT		QUADOPT	
P	τ	Optimum C_1	τ	Optimum C_1
0.1	1.352	1.07293	1.353	1.07293
0.2	1.249	0.92734	1.249	0.92733
0.3	1.154	0.80757	1.155	0.80758
0.4	1.068	0.70875	1.069	0.70877
0.5	0.990	0.62686	0.991	0.62686
0.6	0.920	0.55860	0.921	0.55858
0.7	0.858	0.50133	0.858	0.50138
0.8	0.802	0.45296	0.802	0.45295
0.9	0.751	0.41179	0.751	0.41187
1.0	0.706	0.37652	0.706	0.37650

It will be seen that there is excellent agreement between the two methods, indicating that whilst the golden section method is preferable particularly if the range of values to be used is unknown, the quadratic fit method is easier to program and yields the same results in a shorter time.

These results thus validate the use of the quadratic fit for the hybrid computer solution and also the interactive simulation method using ISIS.

4.0 OPTIMISATION USING 380 PARALLEL LOGIC COMPUTER

4.1 Introduction

The differential equations and evaluation of the performance criteria were solved using standard analogue scaling techniques with the independent variable, time, having a maximum value of 20 seconds. This value was determined as a compromise between knowledge gained from other methods (giving 30 seconds) and scaling considerations (giving 10 seconds to avoid high integrator gains) The simulation was carried out on an EAL 380 parallel-logic computer and details of the component configuration are given in Appendix A1.

Since the value of C_i $(i = 1,2)$ would be calculated at continuously incremented values of τ, its form under high-speed repetitive operation would be given in figure 4.1.1. where τ_4 is the required

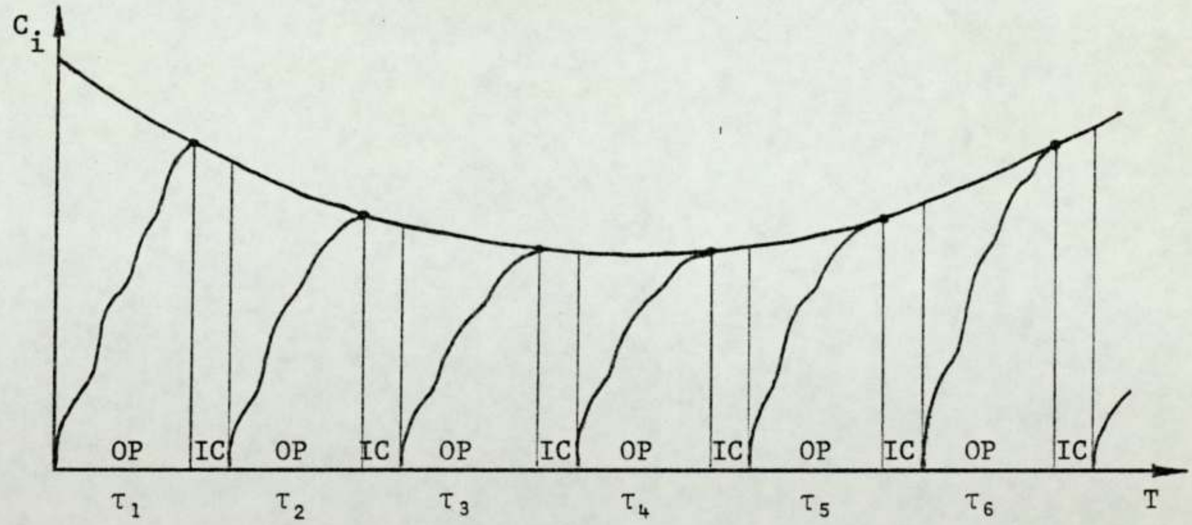

Figure 4.1.1.

value of the duration time for an optimum performance since C_i is a minimum at that value of τ.

Note that the line joining the final values of C_i for each run corresponds to the theoretical curve of C_i versus τ for fixed P. Hence it is necessary to determine the minimum C_i using the final values of each run.

Thus we arrange to compare the final value of the present run $C_{i,f}$ with the previous final value $C_{i,fp}$ and employ the iteration criteria:

If $C_{i,f} > C_{i,fp}$ then $\Delta\tau$ must be negative and if $C_{i,f} < C_{i,fp}$ then $\Delta\tau$ must be positive where $\Delta\tau$ is the increment in τ between runs. When the optimum value is reached, the value of τ will oscillate about this optimum value until $\Delta\tau$ is reduced manually to zero.

4.2 The Analogue Program

Unscaled Equations

$$\frac{dy_2}{dt} = y_1 - P \qquad \text{(U1)}$$

$$\frac{dy_1}{dt} = -0.5y_1 - y_2 \qquad \text{(U2)}$$

$$\frac{dt}{dt} = 1 \qquad \text{(U3)}$$

$$P_1 = y_2 . y_2 \qquad \text{(U4)}$$

$$P_2 = t.P_1 \qquad \text{(U5)}$$

$$\frac{dC_1}{dt} = P_1 (\text{giving } C_1 = \int y_2^2 dt) \qquad \text{(U6)}$$

$$\frac{dC_2}{dt} = P_2 (\text{giving } C_2 = \int t y_2^2 dt) \qquad \text{(U7)}$$

Scaled Variables

$$\left(\frac{y_2}{2}\right), \left(\frac{y_1}{2}\right), \left(\frac{t}{20}\right), \left(\frac{P_1}{4}\right), \left(\frac{P_2}{80}\right), \left(\frac{C_1}{2}\right) \text{ and } \left(\frac{C_2}{2}\right)$$

<u>Scaled Equations</u> (time scale factor, β is taken to be unity)

$$\frac{d}{dt}\left(\frac{y_2}{2}\right) = \left(\frac{y_1}{2}\right) - \left(\frac{P}{2}\right) \qquad (S1)$$

$$\frac{d}{dt}\left(\frac{y_1}{2}\right) = -0.5\left(\frac{y_1}{2}\right) - \left(\frac{y_2}{2}\right) \qquad (S2)$$

$$\frac{d}{dt}\left(\frac{t}{20}\right) = 0.05\ (1) \qquad (S3)$$

$$\left(\frac{P_1}{4}\right) = \left(\frac{y_2}{2}\right)\cdot\left(\frac{y_2}{2}\right) \qquad (S4)$$

$$\left(\frac{P_2}{80}\right) = \left(\frac{P_1}{4}\right)\cdot\left(\frac{t}{20}\right) \qquad (S5)$$

$$\frac{d}{dt}\left(\frac{C_1}{2}\right) = 2\left(\frac{P_1}{4}\right) \qquad (S6)$$

$$\frac{d}{dt}\left(\frac{C_2}{2}\right) = 40\left(\frac{P_2}{80}\right) \qquad (S7)$$

The scaled computer diagram from which the problem was patched is shown in figure 4.2.

Note that C18 ensures that the input is removed after τ secs, C19 terminates the run after 20 secs and C28 detects the minimum value of C_i for a given value of P.

For full details of analogue scaling techniques, the reader is invited to consult the relevant literature (20).

Referring to the patching diagrams, the following list gives the potentiometer descriptions and settings:

Potentiometer	Description	Setting
14	Scaling for maximum t	0.0500
15	Scaling within system equations	0.5000
18	Initial y_2	0.5000
22	τ Increment	Δτ
23	Scaling for C_2	0.4000
30	Input amplitude	P/2

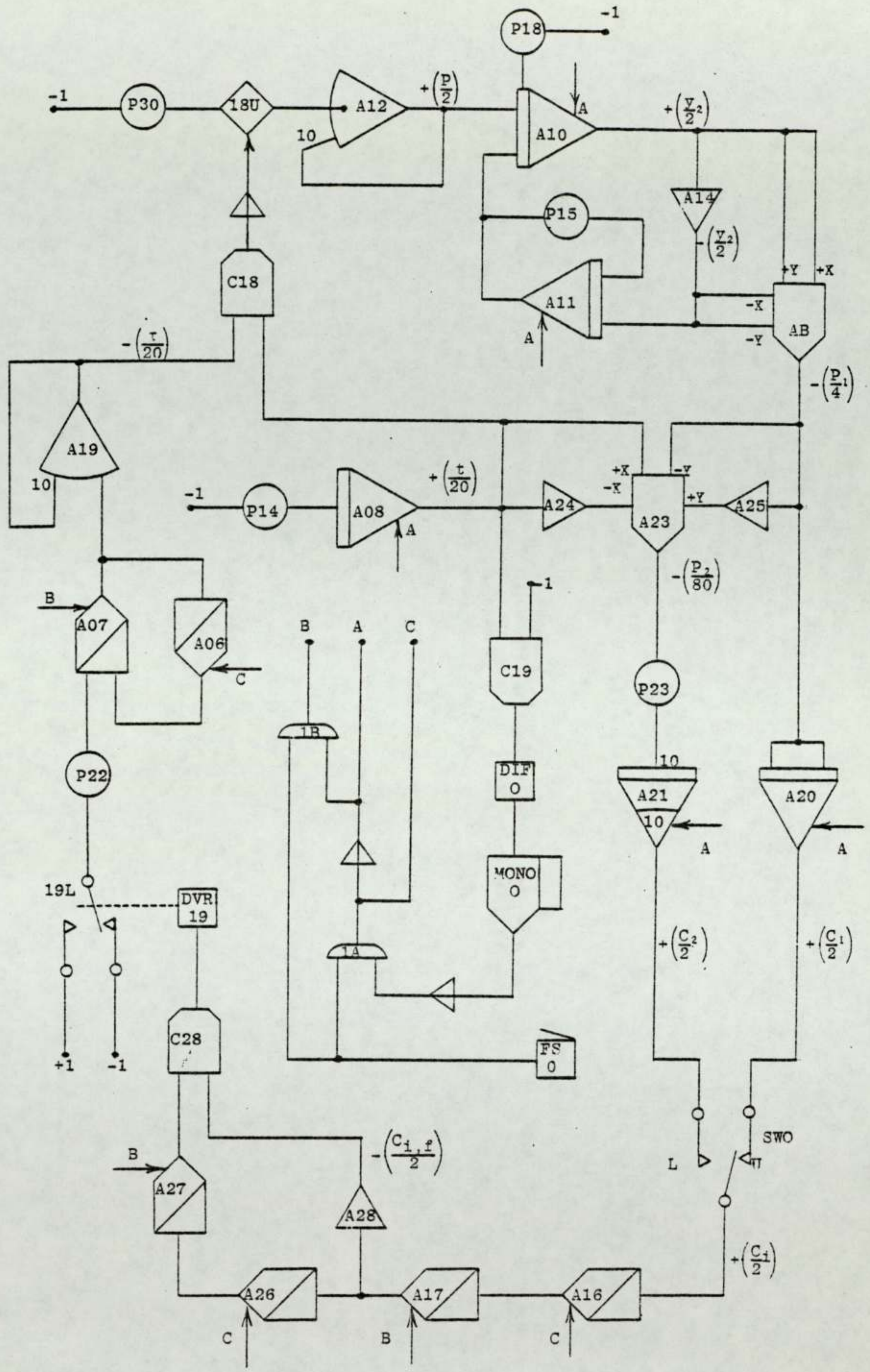

Figure 4.2.1. The 380 Computer Diagram

4.3 Circuit Notes

The runtime for each simulation is 20 secs and the end of the run is detected by C19, which will trigger a logic circuit which ensures correct initialization of the trackstores and gives a fixed IC time as determined by the monostable setting. This method was used in preference to the use of the master timer since it gave more control of the OP/IC cycle.

Comparator C18 determines the cut-off point at $t = \tau$ and, through the D/A switch 18U, ensures that $P = 0$ for $t > \tau$.

The two pairs of trackstores A16/17 and A26/27 are used to generate $C_{i,f}$ and $C_{i,fp}$ for determination of the sign of $\Delta\tau$ via comparator C28 controlling relay driver 19.

4.4 Running the Program

After checking out the patch panel, the program was run twice for each P value, generating C_1 on the first run and C_2 on the second. This was carried out so as to minimise any errors occurring from switching SWO from C_1 to C_2 midway through a set of runs.

Each run was performed using the fast timescale facility with a high initial $\Delta\tau$ reducing to zero when results oscillated about the optimum value so as to converge onto this optimum τ value. Each result was obtained within 30 seconds of starting the simulation.

4.5 Limitations and Inaccuracies

It will be noted from a comparison of the 380 results with those from the theoretical solution that these 380 results do not agree exactly with these theoretical results. This can be ascribed to the following:

(i) Scaling difficulties especially with C_2 *

(ii) Hardware inaccuracies in the track/stores since there seemed to be a certain non-repeatability of the optimum τ value although C_i seemed to give passably repeatable values.

4.6 Table of Results (Unscaled Values)

Several runs were carried out and are averaged in the table below

P	τ	C_1	τ	C_2
0.1	1.461	1.077	1.464	1.764
0.2	1.306	0.930	1.368	1.464
0.3	1.200	0.810	1.184	1.232
0.4	1.109	0.711	1.144	1.032
0.5	1.016	0.630	1.016	0.884
0.6	0.948	0.562	0.980	0.764
0.7	0.892	0.505	0.928	0.676
0.8	0.844	0.458	0.836	0.586
0.9	0.789	0.418	0.798	0.522
1.0	0.720	0.379	0.744	0.466

A full comparison of these results with other methods is given in chapter 8.

* Auto-rescaling techniques can be used to overcome these difficulties.

5.0 SOLUTION BY DIGITAL CONTROL HYBRID

5.1 Introduction

The computer used for this solution was a Solartron HS7-3D analogue computer (21) coupled to a DEC PDP8L digital computer (22). The author's thanks are due to the University of Salford for providing these facilities for his use thus giving access to this further simulation method.

The programming of the analogue equations is similar to that given in section 4, but is repeated here for completeness.

The digital program is fully detailed in appendix A5 and the optimum results are listed at the end of the section.

5.2 The Analogue Section

The following operations are carried out by the HS7-3D analogue and parallel logic section:

(i) the solution of the differential equations

(ii) the evaluation of the chosen performance criterion

(iii) the termination of the analogue run

and (iv) the detection of an overload condition on y_2 if P or τ are too large.

Scaling of the Analogue Computer Section

i) Unscaled Equations

$$\frac{dy_2}{dt} = y_1 - P \qquad \text{(U1)}$$

$$\frac{dy_1}{dt} = -0.5y_1 - y_2 \qquad \text{(U2)}$$

$$\frac{dt}{dt} = 1 \qquad \text{(U3)}$$

$$P_1 = y_2 . y_2 \qquad \text{(U4)}$$

$$P_2 = t.P_1 \qquad \text{(U5)}$$

$$\frac{dC_1}{dt} = P_1 \text{ (giving } C_1 = y_2^2dt) \qquad \text{(U6)}$$

$$\frac{dC_2}{dt} = P_2 \text{ (giving } C_2 = ty_2^2dt) \qquad \text{(U7)}$$

$$PERF = \begin{cases} C_1 \\ C_2 \end{cases} \text{depending on required criterion} \qquad \text{(U8)}$$

ii) <u>Scaled Variables</u>

An initial investigation revealed that the most suitable scaled variables are

$$\left(\frac{y_2}{2}\right), \left(\frac{y_1}{2}\right), \left(\frac{P}{2}\right), \left(\frac{P_1}{4}\right), \left(\frac{t}{10}\right), \left(\frac{P_2}{40}\right), \left(\frac{C_1}{2}\right), \left(\frac{C_2}{2}\right) \text{ and } \left(\frac{PERF}{2}\right)$$

Note that t_{max} = 10 secs since for the range of P and τ under investigation y_2 appeared to have acheived its zero steady state by 10 secs.

iii) <u>Scaled Equations</u>

$$\frac{d}{dt}\left(\frac{y_2}{2}\right) = \left(\frac{y_1}{2}\right) - \left(\frac{P}{2}\right) \qquad \text{(S1)}$$

$$\frac{d}{dt}\left(\frac{y_1}{2}\right) = -0.5\left(\frac{y_1}{2}\right) - \left(\frac{y_2}{2}\right) \qquad \text{(S2)}$$

$$\frac{d}{dt}\left(\frac{t}{10}\right) = 0.1(1) \qquad \text{(S3)}$$

$$\left(\frac{P_1}{4}\right) = \left(\frac{y_2}{2}\right) \cdot \left(\frac{y_2}{2}\right) \qquad \text{(S4)}$$

$$\left(\frac{P_2}{40}\right) = \left(\frac{t}{10}\right) \cdot \left(\frac{P_1}{4}\right) \qquad \text{(S5)}$$

$$\frac{d}{dt}\left(\frac{C_1}{2}\right) = 2\left(\frac{P_1}{4}\right) \tag{S6}$$

$$\frac{d}{dt}\left(\frac{C_2}{2}\right) = 20\left(\frac{P_2}{40}\right) \tag{S7}$$

$$\left(\frac{PERF}{2}\right) = \begin{cases} \left(\frac{C_1}{2}\right) \\ \left(\frac{C_2}{2}\right) \end{cases} \tag{S8}$$

These scaled equations are now used to construct the scaled computer diagram shown in figure 5.2.1.

iv) The Analogue Flow Diagram

The pot-settings are as follows:

*ACD2 = P/2	ACB3	= 0 (used for static check)
*ACD1 = $\tau/10$	ACC3	
ADC4 = $(y_{2,0}/2) = 0.5$	ACB4	
ADB2 = 0.5	ACB5	
ACC4 = 0.1		

*Digital Coefficient Units incorporating an additional sign-reversal (amplifier); remainder are servo-set.

The Comparators are

C1: Detects any negative overload on y_2

C2: Detects end of pulse at $t = \tau$

C3: Detects end of run at $t = 10$.

The Switches are

SW3 Solid State switch to remove the pulse at $t = \tau$ (during OP mode)

SW6 Reed-Relay switch to select required criterion (during and IC or RS operation)

The Lamp Drivers are

LD1: Shows when pulse is on.

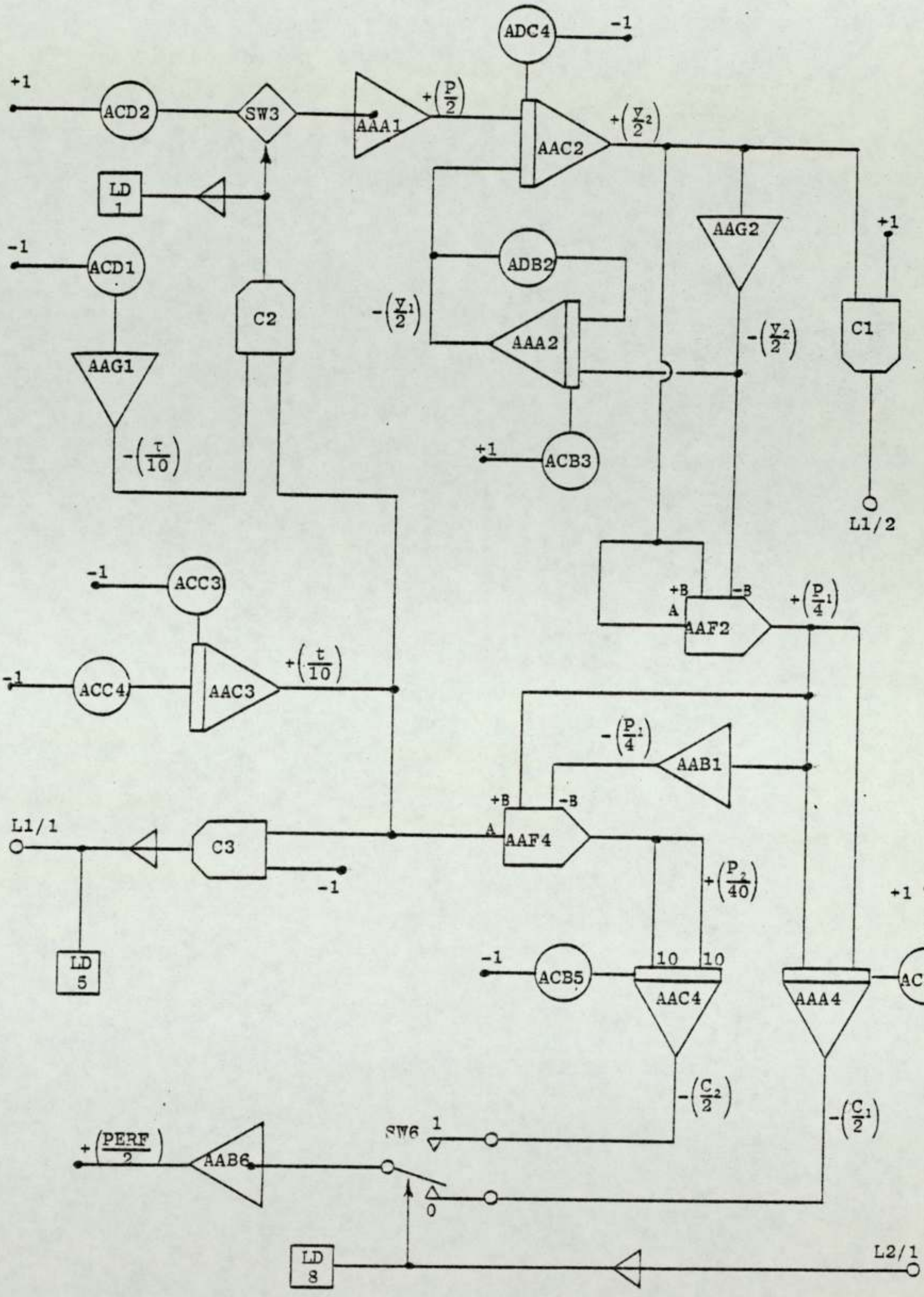

Figure 5.2.1. The HS7-3D Computer Diagram

LD5: Shows when compute mode is on

LD8: Shows which criterion is selected

The Logic Monitor lines are

L1/1: Changes when y_2 becomes negatively overloaded

L1/2: Monitors the end of the compute period

The Logic Sense line is

L2/1: Determines the performance criterion required

5.3 The Digital Computer Section

The flow diagram for the digital computer program and the actual instructions, using Hybrid Fortran, are given in appendix A5.

The digital section performs the following parts of the problem:

(i) Communication with the user via a teletype, accepting data and printing results and requests

(ii) Setting pots and selecting the performance criterion

(iii) Controlling the modes of the analogue computer

(iv) Detection of an overload condition and the end-of-run signal

(v) To read the problem answer from the analogue computer at the end of a run.

(vi) To determine the next run conditions

(vii) To optimise the results at a suitable point in the calculation.

It should be noted that the Hybrid Fortran language used is based on normal FORTRAN instructions with the addition of those instructions needed for setting potentiometers, reading out values etc. (23).

5.4 Analysis of the Results

It will be seen that all the results conform to the following format:

Request for which criterion is to be used
Rate Setting (P)
Initial Duration Time (τ)
Table of Results
Optimum Values
Request for Restart or Exit

Table 5.4.1.: Format of Results

Results Table 5.4.2. indicates a result where the initial τ value is in section B, figure 3.3.1 i.e. greater than the optimum value of τ.

Results Table 5.4.3. indicate two results where several increments are needed before optimisation occurs (τ initial value on Section A, figure 3.3.1.) i.e. less than the optimum value of τ.

On both these tables, the underlined numbers are those which have to be typed in by the operator.

The full set of results is given by Appendix A5.

```
TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 0

RATE SETTING = 0.01

DURATION TIME DT = 1.2

DT = +1.2000      PERFORMANCE = +1.1468
DT = +1.3200      PERFORMANCE = +1.1476
SECOND RUN INDICATES TRY SMALLER DT

DURATION TIME DT = 1.1

DT = +1.1000      PERFORMANCE = +1.1440
DT = +1.2100      PERFORMANCE = +1.1452
SECOND RUN INDICATES TRY SMALLER DT

DURATION TIME DT = 0.8

DT = +0.8000      PERFORMANCE = +1.1374
DT = +0.8800      PERFORMANCE = +1.1386
SECOND RUN INDICATES TRY SMALLER DT

DURATION TIME DT = 0.1

DT = +0.1000      PERFORMANCE = +1.1128
DT = +0.1100      PERFORMANCE = +1.1158
SECOND RUN INDICATES TRY SMALLER DT

DURATION TIME DT = 0.01

DT = +0.0100      PERFORMANCE = +1.1404
DT = +0.0110      PERFORMANCE = +1.1358
DT = +0.0120      PERFORMANCE = +1.1324
DT = +0.0130      PERFORMANCE = +1.1280
DT = +0.0140      PERFORMANCE = +1.1226
DT = +0.0150      PERFORMANCE = +1.1256

OPTIMUM VALUES
DT = +0.0141      PERFORMANCE = +1.1225

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1
```

Table 5.4.2.

```
TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 0

RATE SETTING = 1.1

DURATION TIME DT = 0.4

DT = +0.4000      PERFORMANCE = +0.4224
DT = +0.4400      PERFORMANCE = +0.3914
DT = +0.4800      PERFORMANCE = +0.3658
DT = +0.5200      PERFORMANCE = +0.3468
DT = +0.5600      PERFORMANCE = +0.3332
DT = +0.6000      PERFORMANCE = +0.3260
DT = +0.6400      PERFORMANCE = +0.3256
DT = +0.6800      PERFORMANCE = +0.3310

OPTIMUM VALUES
DT = +0.6228      PERFORMANCE = +0.3251

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1


TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 1

RATE SETTING = 1.1

DURATION TIME DT = 0.5

DT = +0.5000      PERFORMANCE = +0.3068
DT = +0.5500      PERFORMANCE = +0.2688
DT = +0.6000      PERFORMANCE = +0.2490
DT = +0.6500      PERFORMANCE = +0.2490

OPTIMUM VALUES
DT = +0.6250      PERFORMANCE = +0.2465

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = -1
```

Table 5.4.3.

Table of Optimum Results

	$C_1 = \int_0^\infty y_2^2 dt$		$C_2 = \int_0^\infty t y_2^2 dt$	
Pulse Amplitude, P	Pulse Duration, τ	Performance	Pulse Duration, τ	Performance
0.1	1.2125	0.9962	1.2912	1.3633
0.2	1.1786	0.8577	1.2123	1.6789
0.3	1.1005	0.7471	1.1482	0.8680
0.4	1.0224	0.6599	1.0633	0.7202
0.5	0.9467	0.5867	0.9730	0.6048
0.6	0.8771	0.5249	0.8991	0.5126
0.7	0.8164	0.4721	0.8271	0.4371
0.8	0.7563	0.4268	0.7661	0.3758
0.9	0.7083	0.3879	0.7143	0.3246
1.0	0.6637	0.3541	0.6692	0.2813

Table 5.4.4.

6.0 SOLUTION USING SLAM

6.1 Introduction

SLAM, Simulation Language for Analogue Modelling, was developed by ICL and Cranfield Institute of Technology for modelling continuous systems on the ICL range of computers. It is designed to conform to the specifications laid down in the CSSL report (15). SLAM (16) is written to incorporate many standand FORTRAN statements and the SLAM instructions are translated into FORTRAN prior to compilation and execution. For this reason it can be understood by many scientific high-level language users as well as having a large library of subroutines available to back up the language. Because of the nature of the differential equations to be solved being formed in loops, the SLAM translator automatically sorts the instructions into the correct solution order although certain rules must be observed regarding sortable instructions, unsortable instructions being placed in blocks which specifically obviate sorting.

6.2 The Program Structure

A SLAM program will consist of at least one segment which must include a master segment. This master segment may be used to control the whole program in the case of a multisegment program. Within each segment the program will be divided into regions which will be explicitly defined as INITIAL, DYNAMIC and TERMINAL regions. In the case of short single-segment programs, an implicit mode may be defined in which case, the translator automatically sorts the instructions into the correct regions.

6.2a The Structure of a Segment

Any segment may have up to three regions, defined above, which have the following functions:

a) The INITIAL region which is executed only once on entry to the segment and contains all the initializing steps for the segment.

b) The DYNAMIC region which may contain both a DERIVATIVE section for defining the equations to be integrated and a PARALLEL section for non-integrable instructions. It will also contain termination tests, output statements and, within the DERIVATIVE section, the integration algorithm will be defined by means of an INTINF block. This region is executed repeatedly until the termination condition is satisfied.

c) The TERMINAL region in which final outputs are printed, and any calculations required for an optimisation routine performed so as to be available for re-entry into the program for the next set of executions.

6.2b Automatic Sorting

All equations describing differential equations have to be solved simultaneously since time (the independent variable) must be the same for each equation. Since the digital computer only works sequentially this time variable must be kept stationary while all the variables are discretely evaluated. Thus it is necessary for the statements to be sorted such that all the values on their right hand sides are available before an attempt is made to evaluate the left hand side of the expression. These sortable statements are:

i) assignment statements

ii) input and output statements

iii) PERFORM statements (for calling subroutines or other segments).

Inevitably there will be statements which cannot be sorted since they form loops within themselves or with other statements. These must be included in NOSORT blocks and are statements of the form.

```
  i)  A = Q + A      : attempt to calculate own input

 ii)  A = X + Y
      A = Q + 2Y     : one variable with two values

iii)  A = B
      B = C          : cyclically dependent
      C = A

 iv)  2 X = A + B
        :            : labelled loops
        GO TO 2
```

6.2c The Integration Algorithm

In any digital simulation language, it is necessary to be able to integrate step-by-step in an efficient manner. The integration parameters are specified in the INTINF block which besides defining the communication interval, number of steps per interval, the independent variable etc., also allows the use of a variety of numerical integration algorithms.

The algorithms available in the SLAM language are the trapezoidal rule (TRPZ), Simpson's Rule (SIMP), Runge-Kutta Fixed Step (RKFS), Runge-Kutta Variable Step (RKVS) and the Adams Moulton method (ADMN). For variable input functions, it would appear most practical to use the RKVS algorithm although it was found that the presence of a step function caused delays to obtaining the solution while trying to find a suitable step size. For this reason, RKFS is used and gives results comparable with the theoretical values although provision is made for further studies using all the available methods.

It should be noted that Martens (24) concludes that of all integration methods, the RKVS or Kutta-Merson method is most suited to simulation problems.

6.3 The SLAM Program

Detailed listing of the SLAM program is given in appendix A 6 but the following points should be noted concerning the choice of communication interval.

The integration is performed in three phases during which the maximum communication interval (within which the number of steps is a minimum) is used giving speed of computation without loss of accuracy. These three phases are as follows:

(i) Since the pulse terminates at different values of DT (equivalent to the τ parameter of the theoretical solution) it is arranged that the largest possible interval is used during the 'pulse-on' phase i.e. CINT = DT with 20 steps in this interval.

(ii) Immediately after t = DT, the next communication interval is from DT up to the next whole number of seconds, specified by WDIGIT in the program also with 20 steps in the interval.

(iii) Thereafter so that a constant value is taken until the steady state occurs, the communications interval is taken as 1.0 seconds with 10 steps per interval.

These three phases are shown diagramatically in figure 6.3.1. although it should be noted that the use of the RKVS algorithm should obviate the necessity for this partitioning of the timescale.

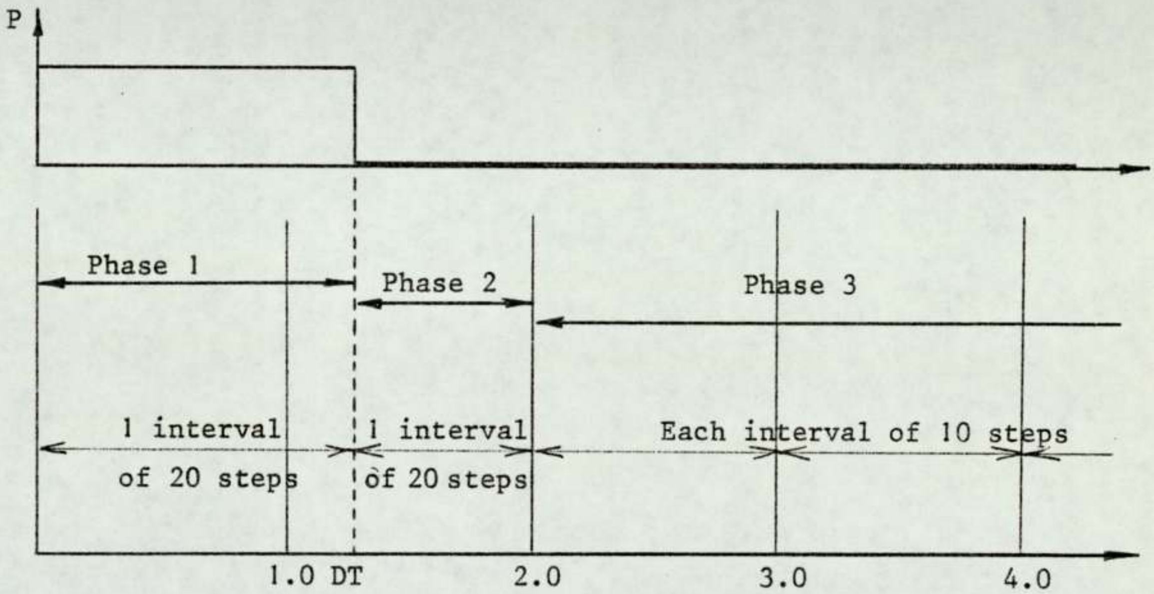

Figure 6.3.1.

6.4 The Results

The full results of the simulation are given in Appendix A6 and comparisons with other methods are given in chapter 8. The table below gives the optimum results for the program run using SLAM.

	For C_1		For C_2	
Pulse P	Optimum τ	Minimum Perf	Optimum τ	Minimum Perf
0.1	1.364	1.07292	1.412	1.66517
0.2	1.260	0.92733	1.302	1.35344
0.3	1.164	0.80755	1.202	1.10892
0.4	1.077	0.70873	1.111	0.91659
0.5	0.999	0.62684	1.029	0.76459
0.6	0.928	0.55858	0.955	0.64367
0.7	0.865	0.50132	0.889	0.54674
0.8	0.808	0.45294	0.830	0.46841
0.9	0.758	0.41178	0.776	0.40457
1.0	0.712	0.37651	0.729	0.35210

Table 6.4.1.

7.0 The Solution Using An Interactive Simulation Language

7.1 Introduction

As mentioned in chapter 2, a recent development has been the introduction of interactive simulation languages for use at a terminal rather than the method of a batch process language such as SLAM. The language used was ISIS (Interactive Simulation by an Interpretative System) which has been developed by Dr. J. Hay and Dr. J.G. Pearce at the Unviersity of Salford for use on their PDP8F system. [For details of programs based on the BEDSOCS language, the reader is referred to Ord-Smith and Stephenson (25).]

The language is based on FORTRAN and uses many of the facilities available in that language. An ISIS program consists of two regions: a control region and a dynamic region which is called by the control region using the command SIM.

The control region sets initial values, constant parameters and controls the updating of these parameters while the dynamic region, entered by the #DYNAMIC instruction solves the differential equations and tabulates or plots output values.

Full editing facilities are available and these are listed in the ISIS manual (18).

All instructions, editing, results, tabulations, listings etc. are outputted onto a VDU but hardcopy may be obtained by transferring control to a DECWRITER using a standard command.

7.2 The ISIS language

Since the language is FORTRAN based, the names of variables and values of constants are similar to FORTRAN except that all numbers are regarded as decimal and variables may be up to five characters in length with certain reserved names for the communication interval, error values and the independent variable T.

In order to indicate derivatives, a series of dashes (') is used to denote the order of the derivative of any variable provided the total number of characters does not exceed six.

Thus $\frac{d^3x}{dt^3}$ would be specified by X''' but the third derivative of the variable CØST would not be allowed since CØST''' uses seven characters. Thus variable names with high derivatives must have a small number of characters in the name.

Arithmetic statements follow the normal rules for a high level language and expressions are evaluated in standard order e.g. parentheses, exponentiation, multiply/divide and add/subtract. Standard functions are also available as in FORTRAN.

Differential equations are written in the usual "decomposed" form, the variables being assigned as shown above.

Thus $\frac{d^2x}{dt^2} = -w^2x$

becomes X'' = -W*W*X

This statement generates the variable X'' as well as the intermediate derivative X'.

The values of X and X' are assumed to be zero on entry to the dynamic region unless specifically set up in the control region.

The integration process is controlled by certain restricted variables which can be set in the control region or are given default values when the SIM instruction is encountered. Multiple runs can be achieved using DO loops but care must be taken to initialize each entry into the dynamic region correctly. All variables take their final value when returning to the control region.

Exit from the dynamic region may be acheived by use of a conditional statement e.g. IF(Y<0)STOP or by reaching the final value of the independent variable as specified by the number of communication intervals or fixed time step (defined by the reserved variables CINT and NCOM).

Input and output is obtained by a simple READ, 'list' or PRINT, 'list' instruction. The latter may be put anywhere in the program but if it is in the dynamic region, it will be executed every communication interval. The program used for solving the optimisation problem only prints at the end of a dynamic run i.e. a final value print although intermediate printing was used during the development of the program.

A further useful output statement is the PLOT statement used in conjunction with a Tektronix 4010 storage oscilloscope. It is possible to specify the maximum and minimum values of the variables so that the axes are scaled correctly. These values may be omitted in which case the PLOT instruction is automatically scaled prior to execution at the end of the run.

The package allows for error messages to be printed during development to assist in the diagnostic procedure. Full details of the facilities offered are contained in the ISIS manual (18).

7.3 The Optimisation Routine

The flow diagram for the optimisation problem is very similar to that used for the hybrid solution except that the IC/OP cycle of that solution is replaced by the dynamic region of the ISIS program. The optimisation algorithm is also that used for the hybrid solution, incorporating a simple quadratic method of finding the optimum solution. The reason for this was the ease with which the instructions in the hybrid program could be read into the PDP8F by typing in each instruction from a listing rather than use of the actual hybrid paper tape input.

Although the program was shown to be working effectively, it should be noted that the package was still in its development phase, the facilities being minimal for efficient operation in particular with reference to the output format connected with the PRINT statement. It is expected that later versions will employ a more flexible layout.

7.4 The Results

The full results are listed in appendix A7 and were obtained using the DECWRITER associated with the PDP8F. The development of the program was carried out entirely using the visual display so that no record was kept during this phase which incorporated additional printing to act as a check on the simulation. Only the fully developed program was listed although the program itself is available on magnetic tape storage should it be required for further development.

The optimum results are listed on table 7.4.1. and comparisons made with the other methods in chapter 8.

	For C_1		For C_2	
Pulse Amplitude	Optimum Time	Minimum Performance	Optimum Time	Minimum Performance
0.1	1.3525	1.0729	1.3993	1.6645
0.2	1.2493	0.9273	1.2908	1.3529
0.3	1.1546	0.8076	1.1913	1.1085
0.4	1.0687	0.7088	1.1009	0.9163
0.5	0.9912	0.6268	1.0195	0.7644
0.6	0.9212	0.5586	0.9467	0.6436
0.7	0.8583	0.5014	0.8808	0.5465
0.8	0.8024	0.4529	0.8221	0.4684
0.9	0.7514	0.4119	0.7696	0.4043
1.0	0.7065	0.3765	0.7222	0.3521

Table 7.4.1.

8.0 CONCLUSIONS

8.1 Programming Comparisons

It was found that the use of SLAM posed the most problems during the actual development phase of each method. This was due to the inherent internal structure of the language with regard to the different regions, the necessity to identify sortable and non-sortable blocks and the fact that it was not possible in the version used to return to any point in the program except to the start of the initial region within the segment. There was also no facility to jump out of a segment before reaching the END statement. In addition to these internal difficulties with the program structure, most of the language had to be understood before programming could begin and this in itself could be a handicap to any engineer wishing to avail himself of the method.

With regard to the ISIS interactive simulation language, it will be noted that although an elementary optimisation algorithm was used, it would be a simple matter to modify the program to use the more sophisticated golden-section method without recourse to the complicated structure of SLAM. ISIS had all the ease of programming associated with the widely used high level language FORTRAN (or BASIC in the case of BEDSOCS) and as such took very little time to learn. The great advantage of ISIS was that it was virtually possible to take a FORTRAN program and insert into it the dynamic region activated by the command SIM. As such, ISIS has much to recommend it from the programming viewpoint. It is understood that BEDSOCS, based on the BASIC language is also similarly easy to learn and use.

It will have been seen that the digital part of the program written for the Hybrid solution is very similar to the ISIS program except that the dynamic region in ISIS is replaced by the RESET, COMPUTE and HOLD instructions together with the programming of the analogue and logic sections. Since the digital section was merely a FORTRAN program with the insertion of necessary instructions for potentiometer setting, amplifier read-out etc., it was relatively easy to program although the difficulties encountered in execution are detailed in section 8.2 below. Since there was very little parallel logic in this program, the overall analogue programming will be discussed next.

The main difficulty in the analogue program was found to be the scaling, especially for the multipliers whose scaling depended on the inputs. Although the scaling was elementary in itself, it had disastrous effects on the results (see below). With this particular program, the parallel logic was of a reasonably simple nature although experience of other problems has shown that it is the logic part of this method that causes most problems, mainly in deciding the algorithms necessary to acheive a specified iteration routine. Even though the ISIS program was based on an earlier FORTRAN listing, there can be no doubt that the knowledge required to program ISIS with confidence is easily gained by a programmer starting from only a knowledge of fundamental FORTRAN.

8.2 Execution Comparisons

Confidence in the execution of a program depends largely on the reliability of the equipment rather than the programmer/operators expertise in programming. It is also necessary to take into account the time taken for execution of the program.

By far the longest method appeared to be the SLAM solution since when started this program proceeded to carry out the complete set of results at one run of the computer. The difficulty occurred in that, being run under batch processing conditions it was necessary to utilize the computer for a long period of continuous operation (or, if run under a multi-programming system, taking even longer in operation while it took its turn). The actual execution presented no problems in itself apart from the time and hence the expense factor.

The Digital-controlled Hybrid computer method took a long time to "get-off-the-ground" due mainly to small but significant hardware difficulties, such as faulty amplifiers, difficulties with translation etc. Because the PDP8L had only minimal storage, it was necessary to convert the program instructions into binary code before actually translating it, necessitating the input of several tapes before the system was ready. Difficulties were also encountered with the interface lines and although when the system finally worked it produced all the results very quickly, it is felt that this method did not have any significant merit in view of the introduction of interactive simulation.

The solution using the parallel-logic computer was relatively simple in its execution but it was the only method used where the operator had to physically write down the values obtained and also to change the various parameters between sets of runs, although the parameter sweep feature was used within the program. In any analogue type method, there is always a danger of loss of contact between the patch panel and the equipment which does not occur in 'hard-wired' digital equipment. Scaling restrictions made multiplication difficult especially in relation to the ty_2^2 term since t started small and increased whilst y_2^2 was large initially (in machine unit terms) and dropped to near zero after the end of the pulse. This meant that the overall product was always small although it had to be scaled dependent on t and y_2^2.

For ease of production of results through a simple set of execution commands, the ISIS program again appeared to be the best of the methods. There were no overload problems as with the analogue/hybrid methods and the interactive nature of the solution gave the operator the feeling of controlling the situation rather than the more remote methods used for the batch processing of a SLAM program.

The execution of the program by a digital method was found to be superior to the analogue-based methods but the programming effort required for a SLAM program coupled with the long impersonal running time made the use of the interactive simulation language the most useful and easily understood of the methods.

8.3 Comparison of Results

Results from individual programs are given in the various appendices but for convenience the optimum values obtained by each method are summarized together below.

The optimum results are presented by considering each criterion and pulse amplitude separately together with the values obtained by each method.

For convenience, the methods used are abbreviated in the tables according to the following key:

T(GS) : Theoretical Result using the Golden Section Method
T(QF) : Theoretical Result using the Quadratic Fit Method
PLC : Result using the Parallel-Logic Computer
DCH : Result using the Digital Control Hybrid Computer
SLAM : Result using SLAM Simulation Language
ISIS : Result using ISIS Interactive Simulation Language

Comparison Tables for $C_1 = \int_0^\infty y_2^2 dt$

(a) P = 0.1

Method	Optimum τ	Minimum C_1
T(GS)	1.352	1.0729
T(QF)	1.353	1.0729
PLC	1.461	1.077
DCH	1.213	0.996
SLAM	1.364	1.0729
ISIS	1.352	1.0729

(b) P = 0.2

Method	Optimum τ	Minimum C_1
T(GS)	1.249	0.9273
T(QF)	1.249	0.9273
PLC	1.306	0.930
DCH	1.179	0.897
SLAM	1.259	0.9273
ISIS	1.249	0.9273

(c) P = 0.3

Method	Optimum τ	Minimum C_1
T(GS)	1.154	0.8076
T(QF)	1.155	0.8076
PLC	1.200	0.810
DCH	1.101	0.747
SLAM	1.164	0.8076
ISIS	1.155	0.8076

(d) P = 0.4

Method	Optimum τ	Minimum C_1
T(GS)	1.068	0.7087
T(QF)	1.069	0.7087
PLC	1.109	0.711
DCH	1.022	0.659
SLAM	1.077	0.7087
ISIS	1.068	0.7088

(e) P = 0.5

Method	Optimum τ	Minimum C_1
T(GS)	0.990	0.6268
T(QF)	0.991	0.6267
PLC	1.016	0.630
DCH	0.947	0.586
SLAM	0.999	0.6268
ISIS	0.991	0.6268

(f) P = 0.6

Method	Optimum τ	Minimum C_1
T(QS)	0.920	0.5586
T(QF)	0.921	0.5586
PLC	0.948	0.562
DCH	0.877	0.525
SLAM	0.928	0.5586
ISIS	0.921	0.5586

(g) P = 0.7

Method	Optimum τ	Minimum C_1
T(QS)	0.858	0.5013
T(QF)	0.858	0.5014
PLC	0.892	0.505
DCH	0.816	0.472
SLAM	0.865	0.5013
ISIS	0.858	0.5014

(h) P = 0.8

Method	Optimum τ	Minimum C_1
T(QS)	0.802	0.4529
T(QF)	0.802	0.4529
PLC	0.844	0.458
DCH	0.756	0.426
SLAM	0.808	0.4529
ISIS	0.802	0.4529

(j) P = 0.9

Method	Optimum τ	Minimum C_1
T(QS)	0.751	0.4118
T(QF)	0.751	0.4119
PLC	0.789	0.418
DCH	0.708	0.388
SLAM	0.758	0.4118
ISIS	0.751	0.4119

(k) P = 1.0

Method	Optimum τ	Minimum C_1
T(QS)	0.706	0.3765
T(QF)	0.706	0.3765
PLC	0.720	0.379
DCH	0.664	0.354
SLAM	0.712	0.3765
ISIS	0.706	0.3765

Before considering the results for C_2 (where no theoretical results are available), it is necessary to conclude which method gives the best results for C_1.

It should first be noted that there is a range of values of τ over which the value of the optimum value of C_1 changes very little giving a shallow minimum for C_1. Dependent on the method used, there will thus be variations in τ to be expected. This is borne out by the results and although there are minor discrepancies it would appear that the results from ISIS are the best fit to those of the theoretical solution. It should also be noticed that the SLAM results give a good fit with both theoretical results and ISIS which would imply that the digital methods are preferable to analogue based methods.

Using the ISIS results as a reference we can now tabulate the results for C_2

Comparison Tables for $C_2 = \int_0^\infty t y_2^2 dt$

(a) P = 0.1

Method	Optimum τ	Minimum C_2
ISIS	1.399	1.664
SLAM	1.411	1.665
DCH	1.291	1.363
PLC	1.464	1.764

(b) P = 0.2

Method	Optimum τ	Minimum C_2
ISIS	1.291	1.353
SLAM	1.302	1.353
DCH	1.212	1.079
PLC	1.368	1.464

(c) P = 0.3

Method	Optimum τ	Minimum C_2
ISIS	1.191	1.108
SLAM	1.201	1.108
DCH	1.148	0.868
PLC	1.184	1.232

(d) P = 0.4

Method	Optimum τ	Minimum C_2
ISIS	1.101	0.916
SLAM	1.111	0.917
DCH	1.063	0.720
PLC	1.144	1.032

(e) P = 0.5

Method	Optimum τ	Minimum C_2
ISIS	1.019	0.764
SLAM	1.028	0.765
DCH	0.973	0.605
PLC	1.016	0.884

(f) P = 0.6

Method	Optimum τ	Minimum C_2
ISIS	0.947	0.644
SLAM	0.955	0.644
DCH	0.899	0.513
PLC	0.980	0.764

(g) P = 0.7

Method	Optimum τ	Minimum C_2
ISIS	0.880	0.546
SLAM	0.889	0.547
DCH	0.827	0.437
PLC	0.928	0.676

(h) P = 0.8

Method	Optimum τ	Minimum C_2
ISIS	0.822	0.468
SLAM	0.830	0.468
DCH	0.766	0.376
PLC	0.836	0.586

(j) P = 0.9

Method	Optimum τ	Minimum C_2
ISIS	0.769	0.404
SLAM	0.777	0.405
DCH	0.714	0.325
PLC	0.798	0.522

(k) P = 1.0

Method	Optimum τ	Minimum C_2
ISIS	0.722	0.352
SLAM	0.729	0.352
DCH	0.669	0.281
PLC	0.744	0.466

The large discrepancies in the DCH and PLC results are due to inaccuracies as a result of scaling and also in the DCH case, the fact that time was scaled to a maximum value of 20 hence all these results would be expected to be on the low side, which they are.

8.4 Conclusions

The above three subsections indicate that an interactive simulation language (such as ISIS) is the best method of tackling these types of problem.

The reasons for this conclusion can be summarized as follows:

(a) excellent comparison with theoretical results,

(b) ease of understanding the language structure and of programming the problem,

(c) the time involved which was very short (only six working hours from starting with the manual for the first time to the production of the results), and

(d) the interactive nature of the method which gave confidence in being able to control the system.

8.5 Further Reading

For additional material, the reader is invited to consult Stojak (27) who compares the use of a full-hybrid computer (the EAL 231 R-V analogue computer linked to an ICL/Elliott 4130 digital computer) with results obtained from an IBM 370 system for a large distillation simulation and also Gay and Payne (28) who discuss simulation techniques using both BASIC and FORTRAN on a small digital computer (the Honeywell 316).

9.0 REFERENCES AND ACKNOWLEDGEMENTS

9.1 References

1. Crank, J., "The Differential Analyser" (Longmans, 1947).
2. Hartree, D.R., "Calculating Instruments and Machines" (Cambridge, 1950).
3. Babbage, C., "Passages from the Life of a Philosopher" (Longmans, 1864).
4. Thomson, J., *Proc. Roy. Soc.* 24 (1876) 262. Description of disc, ball and cylinder Integrator.
5. Bush, V., *J.Frank. Inst.* 22 (1931) 447. Description of first differential analyser.
6. Hartree, D.R., *Math. Gazette*, 22 (1938) 242 (et al.) Descriptions of the Manchester differential analyser.
7. Goldberg, E.A., *R.C.A. Rev.,* 11 (1950) 296, Stabilization of Wide-Band D-C Amplifiers for zero drift and gain.
8. Korn, G.A. and Korn, T.M., "Electronic Analog Computers" (McGraw Hill, 1964).
9. Karplus, W.J. and Saroka, W.W., "Analog Methods" (McGraw Hill, 1959).
10. Smith, R. (et al.) "Advanced Techniques Manual" (Electronic Associates Ltd., 1966).
11. Hausner, A., "Analog and Analog/Hybrid Computer Programming" (Prentice Hall, 1971)
12. Harvard Computation Laboratory Annals, Vol.1 (1946).
13. Hartree, D.R., *Nature* 158 (1946) 500.
14. IBM Corporation Booklet on the IBM Selective Sequence Electronic Calculator (New York, 1948).

15. Augustin, D.C., (et al.), *Simulation,* 9(1967) 281. The SCi Continuous System Simulation Language, CSSL.

16. International Computer Limited, "SLAM programming language" (ICL Technical Publication 4317, 1972).

17. Brown, G., "BEDSOCS, An Interactive Continuous System Simulation Language" (University of Bradford 1973).

18. Hay, J.L., "ISIS, An Interactive Simulation Language" (University of Salford, 1975).

19. Walsh, G.R., "Methods of Optimisation", (Wiley, 1975).

20. Charlesworth, A.S. and Fletcher, J.R., "Systematic Analogue Computer Programming" (Pitman, 1967).

21. Solartron Electronic Group "The HS7 series programming manual".

22. Digital Equipment Corporation, "Introduction to Programming, the PDP.8 Family Computers" (DEC, 1970).

23. R.Chaplin (et al.), Course notes on programming a digital Control Hybrid System. (University of Salford, 1972).

24. Martens, H.R., *Simulation,* 12 (1969) 87, A Comparative Study of Digital Integration Methods.

25. Ord-Smith, R.J. and Stephenson, J. "Computer Simulation of Continuous Systems" (Cambridge University, 1975).

26. Brennan, R.D. and Silberberg, M.Y., *IBM Syst. J.* 6 (1967) 242, Two Continuous System Modelling Programs.

27. Stojak, P.F., *Comp. J.* 16 (1973) 368, Hybrid and Digital Computation Results in Multicomponent Distillation Simulation.

28. Gay, B. and Payne, S.G., *Comp. J.* 16 (1973) 118, Interactive Digital Simulation on a Small Computer.

9.2 Acknowledgments

The author acknowledges with thanks the permission given by the Polytechnic, Wolverhampton to register for a research degree at the University of Aston and for their assistance in paying the various expenses incurred.

Thanks are due to the project supervisor, Mr. K.J. Bowcock, for his encouragement and advice throughout the project; to the various members of the University of Salford for their assistance and guidance and also for the use of their computing facilities; to Mrs. Hilary Rodwell for typing the manuscript from my original handwritten scrawl and bearing with my continual amendments, and finally to my family for their forebearance with my preoccupation with the project.

APPENDICES

APPENDIX A1 THE COMPUTER CONFIGURATIONS

A1.1 The EAL 380 Parallel Logic Computer

The analogue section consists of

30 amplifiers as 10 summer/integrators

6 summer/track-stores

8 summers

6 inverters

32 hand-set potentiometers

3 bipolar multipliers

1 sine/cosine diode function generator

1 20 segment variable diode function generator

4 comparators (with logic output)

4 manual function switches

8 logic controlled analogue switches

4 logic controlled double pole reed-relays

The logic section consists of

1 Master Timer for OP/IC cycle

1 Control section for integrators, trackstores, relays and switches

15 AND gates

2 4-Bit Registers

2 Differentiators

2 Monostables

2 Counters

A1.2 The Digital Control Hybrid Computer

The HS7-3D Analogue computer complement is

66 amplifiers of which
- 16 off 6-input summer/integrators
- 8 off 6-input summers
- 6 off track stores
- 5 off Q.S. multipliers (2 amps each)
- 6 off 3-input summers
- 6 off 3-input integrators
- 2 off diode function generators (2 amps each)
- 10 off inverters

24 servo-set pots

12 hand-set pots

12 digital coefficient units (reed relay switches)

5 comparators

2 dual limiters

5 digital/analog switches

5 buffered reed relay switches

Logic expansion (gates, flipflops, counters, timers, etc.)

8 channel A/D converter

The PDP8-L digital computer consists of

8K memory

High-speed paper tape reader/punch

ASR-33 teletype

A1.3 The PDP8-F Digital Computer (used for ISIS)

32K memory

1 FPP-12 floating point processor

1 Dual TD8E Magnetic tape units

1 RK8E disc unit

1 LA30 Decwriter (30 characters/second)

1 Lynwood VDU

1 Tektronix 4010 Storage Tube Display units

A1.4 The ICL 1903A Digital Computer

This system was used for GOLDOPT, QUADOPT and SLAMOPT and consists of a 96K memory and central processor with the following peripherals :

1 off Card Reader (Model 2101)

1 off Card Punch (1920)

1 off Paper Tape Reader/Punch (2602)

4 off Magnetic Tape Units (1971/2)

3 off EDS8 Disc Transports (2802/3)

2 off Disc Controllers (2802/0)

1 off Line Printer (2402)

1 off 31" Graph Plotter (1934/6)

1 off Universal Scanner (7930)

plus terminal peripherals

11 off Westrex teletypes

2 off Termiprinters

1 off Visual Display unit

APPENDIX A2 The Mathematical Solution

A2.1 Inversion of the Laplace Transforms for Y_2

$$Y_{2A} = \frac{(s-P)(s+\frac{1}{2})}{s(s^2+\frac{1}{2}s+1)}$$

Taking partial fractions, we obtain

$$Y_{2A} = \frac{-\frac{1}{2}P}{s} + \frac{(1+\frac{P}{2})s + \frac{1}{4}(2-3P)}{(s^2+\frac{1}{2}s+1)}$$

$$= \frac{-\frac{1}{2}P}{s} + \frac{(1+\frac{P}{2})s + \frac{1}{4}(2-3P)}{(s+\frac{1}{4})^2 + \omega^2} \qquad \text{where } \omega = \frac{\sqrt{15}}{4}$$

$$= \frac{-\frac{1}{2}P}{s} + (1+\frac{P}{2})\left[\frac{(s+\frac{1}{4})}{(s+\frac{1}{4})^2 + \omega^2}\right] + \frac{1}{\sqrt{15}}(1-\frac{7P}{2})\left[\frac{\omega}{(s+\frac{1}{4})^2 + \omega^2}\right]$$

In this form, inverse transforms may be taken to obtain

$$y_{2A} = -\frac{P}{2} + \left[(1+\frac{P}{2})\cos\omega t + \frac{1}{\sqrt{15}}(1-\frac{7P}{2})\sin\omega t\right]e^{-\frac{1}{4}t}$$

Since the function is valid for all t>0, the function may be written in terms of the unit step function, H(t), defined in figure 3.4.1 as

$$y_{2A}(t) = \left\{\left[(1+\frac{P}{2})\cos\frac{\sqrt{15}}{4}t + \frac{1}{\sqrt{15}}(1-\frac{7P}{2})\sin\frac{\sqrt{15}}{4}t\right]e^{-\frac{1}{4}t} - \frac{P}{2}\right\}H(t)$$

Also $$Y_{2B} = \frac{P(s+\frac{1}{2})}{s(s^2+\frac{1}{2}s+1)}$$

Again, taking partial fractions, we obtain

$$Y_{2B} = \frac{\frac{1}{2}P}{s} + \frac{(\frac{3}{4} - \frac{1}{2}s)P}{s^2 + \frac{1}{2}s + 1}$$

which reduces in a similar manner to Y_{2A} to give

$$Y_{2B} = \frac{\frac{1}{2}P}{s} - \frac{1}{2}P\left[\frac{s+\frac{1}{4}}{(s+\frac{1}{4})^2 + \omega^2}\right] + \frac{7P}{2\sqrt{15}}\left[\frac{\omega}{(s+\frac{1}{4})^2 + \omega^2}\right]$$

and hence taking inverse transforms

$$y_{2B} = \tfrac{1}{2}P\left\{1 - e^{-\frac{1}{4}t}\left[\cos\omega t - \frac{7P}{\sqrt{15}}\sin\omega t\right]\right\}$$

Referring to equation (3.4.5), it will be seen that this function is delayed by time τ and hence we have the final result

$$y_{2B}(t-\tau) = \tfrac{1}{2}P\left\{1 - e^{-\frac{1}{4}(t-\tau)}\left[\cos\omega(t-\tau) - \frac{7P}{\sqrt{15}}\sin\omega(t-\tau)\right]\right\}H(t-\tau)$$

A2.2 Evaluation of $\int_0^\infty y_2^2 dt$

We have already seen that

$$y_2 = \left\{e^{-\frac{1}{4}t}\left[\left(1+\frac{P}{2}\right)\cos\frac{\sqrt{15}}{4}t + \frac{1}{\sqrt{15}}\left(1-\frac{7P}{2}\right)\sin\frac{\sqrt{15}}{4}t\right] - \frac{P}{2}\right\}H(t)$$

$$+ \frac{P}{2}\left\{1 - e^{-\frac{1}{4}(t-\tau)}\left[\cos\frac{\sqrt{15}}{4}(t-\tau) - \frac{7}{\sqrt{15}}\sin\frac{\sqrt{15}}{4}(t-\tau)\right]\right\}H(t-\tau)$$

$$= y_{2A}(t) + y_{2B}(t-\tau)$$

and hence

$$\int_0^\infty y_2^2 dt = \int_0^\infty \left[y_{2A}(t) + y_{2B}(t-\tau)\right]^2 dt$$

$$= \int_0^\infty y_{2A}{}^2(t)dt + 2\int_\tau^\infty y_{2A}(t).y_{2B}(t-\tau)dt$$

$$+ \int_\tau^\infty y_{2B}{}^2(t-\tau)dt$$

i.e.

$$\int_o^\infty y_2^2 dt$$

$$= \int_o^\infty \left\{ e^{-\frac{1}{4}t} \left[\left(1+\frac{P}{2}\right) \cos \frac{\sqrt{15}}{4}t + \frac{1}{\sqrt{15}}\left(1-\frac{7P}{2}\right) \sin\frac{\sqrt{15}}{4}t \right] - \frac{P}{2} \right\}^2 dt$$

$$+ 2\int_\tau^\infty \left\{ e^{-\frac{1}{4}t} \left[\left(1+\frac{P}{2}\right) \cos \frac{\sqrt{15}}{4}t + \frac{1}{\sqrt{15}}\left(1-\frac{7P}{2}\right) \sin\frac{\sqrt{15}t}{4} \right] - \frac{P}{2} \right\} \bullet$$

$$\left\{ e^{-\frac{1}{4}(t-\tau)} \left[\frac{7P}{2\sqrt{15}} \sin\frac{\sqrt{15}}{4}(t-\tau) - \frac{P}{2}\cos\frac{\sqrt{15}}{4}(t-\tau) \right] + \frac{P}{2} \right\} dt$$

$$+ \int_\tau^\infty \left\{ \frac{P}{2} - e^{-\frac{1}{4}(t-\tau)} \left[\frac{P}{2}\cos\frac{\sqrt{15}}{4}(t-\tau) - \frac{7P}{2\sqrt{15}} \sin\frac{\sqrt{15}}{4}(t-\tau) \right] \right\}^2 dt$$

After expansion of the squared and product terms, and combining these using compound angle formulae, the integral reduces to

$$\int_o^\infty y_2^2 dt$$

$$= \int_o^\infty e^{-\frac{1}{2}t} \left\{ \frac{8+4P+8P^2}{15} + \left(\frac{28+44P-17P^2}{60}\right) \cos\frac{\sqrt{15}}{4}t + \left(\frac{4-12P+7P^2}{4\sqrt{15}}\right) \sin\frac{\sqrt{15}}{4}t \right. dt$$

$$- \int_o^\tau e^{-\frac{1}{4}t} \left\{ P\left(1+\frac{P}{2}\right) \cos\frac{\sqrt{15}}{4}t + \frac{P}{\sqrt{15}}\left(1-\frac{7P}{2}\right) \sin\frac{\sqrt{15}}{4}t \right\} dt + \frac{1}{4}P^2 \int_o^\tau dt$$

$$+ \int_\tau^\infty e^{-\frac{1}{2}(t-\tau)} \left\{ \left(\frac{8P^2}{15} - \frac{17P^2}{60}\right) \cos\frac{\sqrt{15}}{4}(t-\tau) - \left(\frac{7P^2}{4\sqrt{15}}\right) \sin\frac{\sqrt{15}}{4}(t-\tau) \right\} dt$$

$$+ \int_\tau^\infty e^{-\frac{1}{4}(2t-\tau)} \left\{ \left(\frac{6P+7P^2}{2\sqrt{15}}\right) \sin\frac{\sqrt{15}}{4}(2t-\tau) + \left(\frac{17P^2-22P}{30}\right) \cos\frac{\sqrt{15}}{4}(2t-\tau) \right.$$

$$\left. - \left(\frac{4P}{\sqrt{15}}\right) \sin\frac{\sqrt{15}}{4}\tau - \left(\frac{4P+16P^2}{15}\right) \cos\frac{\sqrt{15}}{4}\tau \right\} dt$$

Each one of these integrals is now evaluated during which process several terms will cancel out to give the following result:

$$\int_0^\infty y_2^2 dt$$

$$= \left\{ \tfrac{1}{4}P^2\tau + e^{-\frac{1}{4}\tau}\left[\left(\frac{2P-3P^2}{4}\right)\cos\frac{\sqrt{15}}{4} - \left(\frac{14P+11P^2}{4\sqrt{15}}\right)\sin\frac{\sqrt{15}}{4}\tau\right] - \left(\frac{2P-3P^2}{4}\right)\right\}$$

$$+ \left(\frac{20+4P+13P^2}{16}\right) - \left\{ e^{-\frac{1}{4}\tau}\left[\left(\frac{5P^2+50P}{8\sqrt{15}}\right)\sin\frac{\sqrt{15}}{4}\tau + \left(\frac{2P+13P^2}{8}\right)\cos\frac{\sqrt{15}}{4}\tau\right]\right\}$$

$$+ \left(\frac{13P^2}{16}\right)$$

After collection of the common terms, this expression reduces to

$$C_1 = \int_0^\infty y_2^2 dt$$

$$= \frac{19P^2-2P+10}{8} + \tfrac{1}{4}P^2\tau + e^{-\frac{1}{4}\tau}\left\{\left(\frac{2P-19P^2}{8}\right)\cos\frac{\sqrt{15}}{4}\tau - \left(\frac{27P^2+78P}{8\sqrt{15}}\right)\sin\frac{\sqrt{15}}{4}\tau\right\}$$

which is the function calculated by the subroutine CALC in the analytical solutions QUADOPT and GOLDOPT.

APPENDIX A3 The Optimisation Algorithms

A3.1 The Quadratic Fit Method

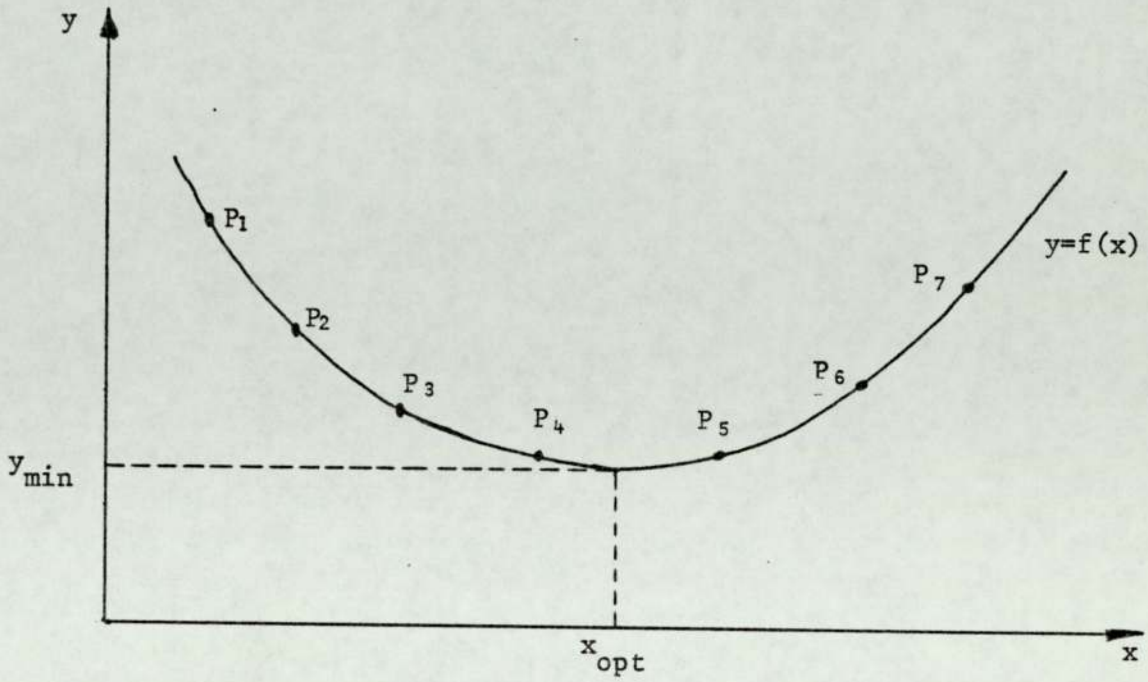

Figure A3.1.1.

Given the above function y = f(x), then y is evaluated at successive values x_1, x_2, x_3,... etc. giving y_i at P_i until such time as the value at P_K is greater than the previous value at P_{K-1}.

The program is arranged to store the last three values of x and y giving the three pairs of coordinates (for the above function)

$$(x_3,\ y_3), \quad (x_4,\ y_4) \quad \text{and} \quad (x_5,\ y_5)$$

where $y_5 > y_4$.

It is then assumed that a quadratic equation of the form $y = Ax^2 + Bx + C$ can be fitted through the three points giving

$$x_1^2A + x_1B + C = y_1$$

$$x_2^2A + x_2B + C = y_2$$

$$x_3^2A + x_3B + C = y_3$$

from which we can obtain the coefficients as follows

$$A = \frac{\dfrac{y_1 - y_2}{x_1 - x_2} - \dfrac{y_1 - y_3}{x_1 - x_3}}{x_2 - x_3}$$

$$B = \frac{y_1 - y_2}{x_1 - x_2} - (x_1 + x_2)A$$

and hence

$$C = y_1 - Bx_1 - Ax_1^2$$

Having obtained A, B and C, simple differentiation can be used to give the minimum point whose coordinates are

$$x_{opt} = -\frac{B}{2A}$$

$$y_{min} = A.x_{opt}^2 + B.x_{opt} + C$$

A3.2 The Golden Section Method

Consider a function f(x) which is uni-model in the range x = A to x = B within which it is required to determine the minimum value, $\bar{x}$ of f(x), figure A3.2.1.

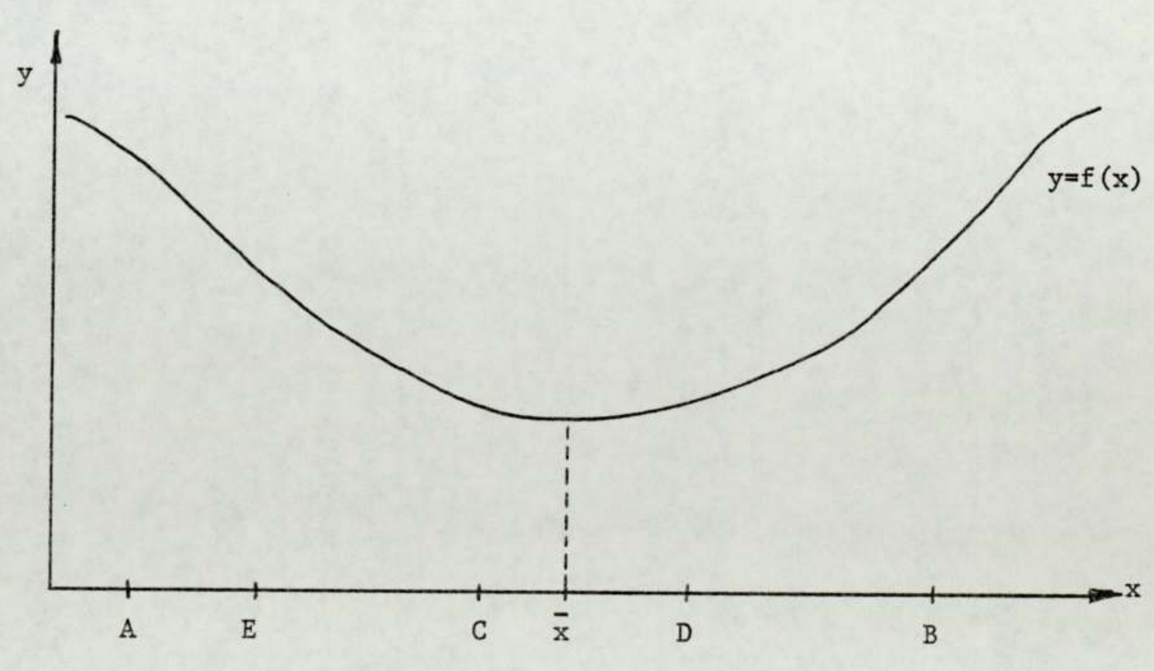

Figure A3.2.1.

Suppose we evaluate f(x) at x = C and x = D such that AD = CB.
Then if

$$f(x_C) \geqslant f(x_D) \qquad \text{(A3.2.1.)}$$

then the required minimum value, $\bar{x}$, will lie in the range AD and hence with two evaluations, it is possible to determine whether $\bar{x}$ lies in AD or CB which it would seem reasonable to choose as being equal in length.

Suppose equation (A3.2.1.) holds so that $\bar{x}$ lies in AD. Another evaluation is now made at E so that AC = ED as before.

If these ratios were made equal each time a subdivision of the interval is required then there would be a simple invariant procedure for dividing the known range.

$$\text{i.e.} \quad \frac{AC}{AD} = \frac{AD}{AB}$$

Let $t = \frac{AD}{AB}$ and note that BC = AD.

Thus

$$\frac{AC}{AD} = \frac{AB - BC}{AD} = \frac{AD}{AB}$$

i.e. $$\frac{1}{t} - 1 = t$$

or $$t^2 + t - 1 = 0$$

and since t is a positive ratio, then solving this quadratic equation gives

$$t = \tfrac{1}{2}(\sqrt{5} - 1)$$

Thus after the initial stage, the intervals are divided successively in the ratios t:1 and 1:t until there is no significant error between the latest two values for $\bar{x}$.

Footnote: It should be noted that

$$t = \tfrac{1}{2}(\sqrt{5} - 1) = \lim_{n\to\infty}\left(\frac{F_n}{F_{n+1}}\right)$$

where F_n and F_{n+1} are successive terms of the Fibonacci sequence given by 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, ...

APPENDIX A4 The FORTRAN Programs and Results

A4.1 GOLDOPT

A4.2 QUADOPT

A4.1. GOLDOPT

This program solves the theoretical result using the Golden-Section Optimisation algorithm.

```
      MASTER GOLDOPT
      DIMENSION P(50),DT(50),C1(50)
      COMMON PF
      K=0
      PF=0.1
      T=0.5*((SQRT(5.0))-1.0)
    1 TOP=2.0
      BOTTOM=0.0
      I=1
      WRITE(2,100)
      WRITE(2,101)PF
      WRITE(2,102)
      WRITE(2,103)
    2 DT1=BOTTOM+(1.0-T)*(TOP-BOTTOM)
      DT2=BOTTOM+T*(TOP-BOTTOM)
      CALL CALC(PERF1,DT1)
      CALL CALC(PERF2,DT2)
      WRITE(2,104)I,BOTTOM,TOP,DT1,PERF1
      WRITE(2,105)DT2,PERF2
      I=I+1
      IF(PERF1-PERF2)3,4,5
    3 TOP=DT2
      GO TO 6
    4 TOP=DT2
      BOTTOM=DT1
      GO TO 6
    5 BOTTOM=DT1
    6 CONTINUE
      IF(ABS(DT2-DT1)-1.0E-4)7,2,2
    7 K=K+1
      P(K)=PF
      DT(K)=DT1
      C1(K)=PERF1
      WRITE(2,106)DT1,PERF1
      PF=PF+0.1
      IF(PF-1.05)1,10,10
   10 WRITE(2,107)
      WRITE(2,108)
      DO 11 I=1,K
   11 WRITE(2,109)P(I),DT(I),C1(I)
      STOP
  100 FORMAT(1H1,5X,37HANALYTICAL SOLUTION.   GOLDEN SECTION,
     18H METHOD.,//)
  101 FORMAT(1H0,25X,18HPULSE AMPLITUDE = ,F5.3,//)
  102 FORMAT(1H0,5X,9HITERATION,10X,2HDT,12X,8HDURATION,
     23X,11HPERFORMANCE)
  103 FORMAT(1H ,6X,6HNUMBER,9X,8HINTERVAL,11X,4H(DT),9X,2H I)
  104 FORMAT(1H ,6X,I2,8X,F6.3,2H -,2(F6.3,6X),F8.5)
  105 FORMAT(1H ,36X,F6.3,6X,F8.5)
  106 FORMAT(1H0,/,15X,16HOPTIMUM RESULTS:,6X,F6.3,6X,F8.5)
  107 FORMAT(1H1,//,17X,24HTABLE OF OPTIMUM RESULTS)
  108 FORMAT(1H0,5X,12HAMPLITUDE, P,4X,12HDURATION, DT,
     34X,15HPERFORMANCE, C1,/)
  109 FORMAT(1H0,8X,F5.2,11X,F6.3,9X,F8.5)
      END
```

Program A4.1.   GOLDOPT

```
      SUBROUTINE CALC(C,D)
      COMMON PF
      F=0.25*D
      Q=PF*PF
      R=SQRT(15.0)
      X=(19.0*Q-2.0*PF+10.0)/8.0
      Y=F*Q
      E=EXP(-F)
      S=((2.0*PF-19.0*Q)/8.0)*COS(R*F)
      Z=((27.0*Q+78.0*PF)/(8.0*R))*SIN(R*F)
      C=X+Y*E*(S-Z)
      RETURN
      END
```

Program A4.1 GOLDOPT (continued)

ANALYTICAL SOLUTION. GOLDEN SECTION METHOD.

PULSE AMPLITUDE = 0.100

ITERATION NUMBER	DT INTERVAL	DURATION (DT)	PERFORMANCE I
1	0.000 - 2,000	0.764	1.10638
		1.236	1.07417
2	0.764 - 2,000	1.236	1.07417
		1.528	1.07562
3	0.764 - 1,528	1,056	1,08121
		1.236	1.07417
4	1,056 - 1,528	1,236	1.07417
		1.348	1.07293
5	1.236 - 1,528	1,348	1,07293
		1.416	1.07329
6	1.236 - 1,416	1,305	1,07313
		1.348	1.07293
7	1.305 - 1.416	1,348	1,07293
		1.374	1.07297
8	1,305 - 1,374	1,331	1,07297
		1.348	1.07293
9	1,331 - 1,374	1,348	1,07293
		1.358	1.07293
10	1.331 - 1.358	1,341	1,07294
		1.348	1.07293
11	1.341 - 1,358	1,348	1,07293
		1.351	1.07293
12	1,348 - 1,358	1,351	1,07293
		1.354	1.07293
13	1,348 - 1,354	1,350	1.07293
		1.351	1.07293
14	1,350 - 1,354	1,351	1,07293
		1.352	1.07293
15	1,351 - 1,354	1,352	1,07293
		1.353	1.07293
16	1,351 - 1,353	1,352	1,07293
		1.352	1.07293
17	1,352 - 1,353	1,352	1,07293
		1.352	1.07293
18	1,352 - 1,353	1,352	1,07293
		1.353	1.07293
19	1,352 - 1,353	1,352	1,07293
		1.352	1.07293

OPTIMUM RESULTS: 1,352 1,07293

ANALYTICAL SOLUTION, GOLDEN SECTION METHOD.

PULSE AMPLITUDE = 0,200

ITERATION NUMBER	DT INTERVAL	DURATION (DT)	PERFORMANCE I
1	0,000 - 2,000	0,764	0.97539
		1,236	0,92737
2	0,764 - 2,000	1,236	0,92737
		1,528	0,94158
3	0,764 - 1,528	1,056	0,93474
		1,236	0,92737
4	1,056 - 1,528	1,236	0,92737
		1,348	0,92918
5	1,056 - 1,348	1,167	0,92865
		1,236	0,92737
6	1,167 - 1,348	1,236	0,92737
		1,279	0,92751
7	1,167 - 1,279	1,210	0,92764
		1,236	0,92737
8	1,210 - 1,279	1,236	0,92737
		1,252	0,92734
9	1,236 - 1,279	1,252	0,92734
		1,262	0,92738
10	1,236 - 1,262	1,246	0,92734
		1,252	0,92734
11	1,236 - 1,252	1,242	0,92735
		1,246	0,92734
12	1,242 - 1,252	1,246	0,92734
		1,248	0,92734
13	1,246 - 1,252	1,248	0,92734
		1,250	0,92734
14	1,246 - 1,250	1,248	0,92734
		1,248	0,92734
15	1,248 - 1,250	1,248	0,92734
		1,249	0,92734
16	1,248 - 1,250	1,249	0,92734
		1,249	0,92734
17	1,248 - 1,249	1,249	0,92734
		1,249	0,92734
18	1,249 - 1,249	1,249	0,92734
		1,249	0,92734
19	1,249 - 1,249	1,249	0,92734
		1,249	0,92734

OPTIMUM RESULTS: 1,249 0,92734

ANALYTICAL SOLUTION. GOLDEN SECTION METHOD.

PULSE AMPLITUDE = 0.300

ITERATION NUMBER	DT INTERVAL	DURATION (DT)	PERFORMANCE I
1	0.000 - 2.000	0.764	0.85703
		1.236	0.80961
2	0.764 - 2.000	1.236	0.80961
		1.528	0.84787
3	0.764 - 1.528	1.056	0.81061
		1.236	0.80961
4	1.056 - 1.528	1.236	0.80961
		1.348	0.81874
5	1.056 - 1.348	1.167	0.80762
		1.236	0.80961
6	1.056 - 1.236	1.125	0.80784
		1.167	0.80762
7	1.125 - 1.236	1.167	0.80762
		1.193	0.80804
8	1.125 - 1.193	1.151	0.80757
		1.167	0.80762
9	1.125 - 1.167	1.141	0.80762
		1.151	0.80757
10	1.141 - 1.167	1.151	0.80757
		1.157	0.80757
11	1.151 - 1.167	1.157	0.80757
		1.161	0.80758
12	1.151 - 1.161	1.155	0.80757
		1.157	0.80757
13	1.151 - 1.157	1.153	0.80757
		1.155	0.80757
14	1.153 - 1.157	1.155	0.80757
		1.156	0.80757
15	1.153 - 1.156	1.154	0.80757
		1.155	0.80757
16	1.153 - 1.155	1.154	0.80757
		1.154	0.80757
17	1.154 - 1.155	1.154	0.80757
		1.154	0.80757
18	1.154 - 1.155	1.154	0.80757
		1.155	0.80757
19	1.154 - 1.155	1.154	0.80757
		1.154	0.80757

OPTIMUM RESULTS: 1.154 0.80757

ANALYTICAL SOLUTION. GOLDEN SECTION METHOD.

PULSE AMPLITUDE = 0.400

ITERATION NUMBER	DT INTERVAL	DURATION (DT)	PERFORMANCE I
1	0,000 - 2,000	0.764	0.75130
		1.236	0.72087
2	0,764 - 2,000	1.236	0.72087
		1.528	0.79449
3	0,764 - 1,528	1.056	0.70882
		1.236	0.72087
4	0,764 - 1,236	0.944	0.71566
		1.056	0.70882
5	0,944 - 1,236	1.056	0.70882
		1.125	0.71014
6	0,944 - 1,125	1.013	0.71010
		1.056	0.70882
7	1,013 - 1,125	1.056	0.70882
		1.082	0.70883
8	1,013 - 1,082	1.039	0.70912
		1.056	0.70882
9	1,039 - 1,082	1.056	0.70882
		1.066	0.70875
10	1,056 - 1,082	1.066	0.70875
		1.072	0.70876
11	1,056 - 1,072	1.062	0.70877
		1.066	0.70875
12	1,062 - 1,072	1.066	0.70875
		1.068	0.70875
13	1,066 - 1,072	1.068	0.70875
		1.070	0.70875
14	1,066 - 1,070	1.067	0.70875
		1.068	0.70875
15	1,067 - 1,070	1.068	0.70875
		1.069	0.70875
16	1,067 - 1,069	1.068	0.70875
		1.068	0.70875
17	1,068 - 1,069	1.068	0.70875
		1.068	0.70875
18	1,068 - 1,068	1.068	0.70875
		1.068	0.70875
19	1,068 - 1,068	1.068	0.70875
		1.068	0.70875

OPTIMUM RESULTS: 1.068 0.70875

ANALYTICAL SOLUTION. GOLDEN SECTION METHOD.

PULSE AMPLITUDE = 0,500

ITERATION NUMBER	DT INTERVAL	DURATION (DT)	PERFORMANCE I
1	0,000 - 2,000	0,764	0.65819
		1,236	0.66116
2	0,000 - 1,236	0,472	0.79529
		0,764	0.65819
3	0,472 - 1,236	0,764	0,65819
		0,944	0.62813
4	0,764 - 1,236	0,944	0.62813
		1,056	0.62935
5	0,764 - 1,056	0,875	0.63483
		0,944	0.62813
6	0,875 - 1,056	0,944	0,62813
		0,987	0.62686
7	0,944 - 1,056	0,987	0.62686
		1,013	0.62716
8	0,944 - 1,013	0,971	0.62709
		0,987	0.62686
9	0,971 - 1,013	0,987	0.62686
		0,997	0.62688
10	0,971 - 0,997	0,981	0.62691
		0,987	0.62686
11	0,981 - 0,997	0,987	0.62686
		0,991	0.62686
12	0,987 - 0,997	0,991	0.62686
		0,993	0.62686
13	0,987 - 0,993	0,989	0.62686
		0,991	0.62686
14	0,989 - 0,993	0,991	0.62686
		0,992	0.62686
15	0,989 - 0,992	0,990	0,62686
		0,991	0.62686
16	0,990 - 0,992	0,991	0.62686
		0,991	0.62686
17	0,990 - 0,991	0,990	0.62686
		0,991	0.62686
18	0,990 - 0,991	0,990	0.62686
		0,990	0.62686
19	0,990 - 0,991	0,990	0.62686
		0,991	0.62686
	OPTIMUM RESULTS:	0,990	0.62686

ANALYTICAL SOLUTION, GOLDEN SECTION METHOD.

PULSE AMPLITUDE = 0,600

ITERATION NUMBER	DT INTERVAL	DURATION (DT)	PERFORMANCE 1
1	0,000 - 2,000	0,764	0.57770
		1,236	0.63048
2	0,000 - 1,236	0,472	0.71979
		0,764	0.57770
3	0,472 - 1,236	0,764	0.57770
		0,944	0.55902
4	0,764 - 1,236	0,944	0.55902
		1,056	0.57222
5	0,764 - 1,056	0,875	0.56017
		0,944	0.55902
6	0,875 - 1,056	0,944	0.55902
		0,987	0.56191
7	0,875 - 0,987	0,918	0.55860
		0,944	0.55902
8	0,875 - 0,944	0,902	0.55887
		0,918	0.55860
9	0,902 - 0,944	0,918	0.55860
		0,928	0.55864
10	0,902 - 0,928	0,912	0.55866
		0,918	0.55860
11	0,912 - 0,928	0,918	0.55860
		0,922	0.55860
12	0,918 - 0,928	0,922	0.55860
		0,924	0.55861
13	0,918 - 0,924	0,920	0.55860
		0,922	0.55860
14	0,918 - 0,922	0,919	0.55860
		0,920	0.55860
15	0,919 - 0,922	0,920	0.55860
		0,921	0.55860
16	0,919 - 0,921	0,920	0.55860
		0,920	0.55860
17	0,920 - 0,921	0,920	0.55860
		0,921	0.55860
18	0,920 - 0,921	0,921	0.55860
		0,921	0.55860
19	0,920 - 0,921	0,920	0.55860
		0,921	0.55860

OPTIMUM RESULTS: 0.920 0.55860

ANALYTICAL SOLUTION. GOLDEN SECTION METHOD.

PULSE AMPLITUDE = 0,700

ITERATION NUMBER	DT INTERVAL	DURATION (DT)	PERFORMANCE I
1	0,000 - 2,000	0,764	0.50985
		1,236	0,62882
2	0,000 - 1,236	0,472	0,64944
		0,764	0,50985
3	0,472 - 1,236	0,764	0,50985
		0,944	0,50834
4	0,764 - 1,236	0,944	0,50834
		1,056	0,53743
5	0,764 - 1.056	0,875	0,50162
		0,944	0,50834
6	0,764 - 0,944	0,833	0,50194
		0,875	0,50162
7	0,833 - 0,944	0,875	0,50162
		0,902	0,50315
8	0,833 - 0,902	0,859	0,50134
		0,875	0,50162
9	0,833 - 0,875	0,849	0,50141
		0,859	0,50134
10	0,849 - 0,875	0,859	0,50134
		0,865	0,50139
11	0,849 - 0,865	0,855	0,50134
		0,859	0,50134
12	0,855 - 0,865	0,859	0,50134
		0,861	0,50135
13	0,855 - 0,861	0,858	0,50133
		0,859	0,50134
14	0,855 - 0,859	0,857	0,50134
		0,858	0,50133
15	0,857 - 0,859	0,858	0,50133
		0,858	0,50133
16	0,857 - 0,858	0,857	0,50133
		0,858	0,50133
17	0,857 - 0,858	0,858	0,50133
		0,858	0,50133
18	0,858 - 0,858	0,858	0,50133
		0,858	0,50133
19	0,858 - 0,858	0,858	0,50133
		0,858	0,50133

OPTIMUM RESULTS: 0,858 0,50133

ANALYTICAL SOLUTION. GOLDEN SECTION METHOD.

PULSE AMPLITUDE = 0,800

ITERATION NUMBER	DT INTERVAL	DURATION (DT)	PERFORMANCE I
1	0,000 - 2,000	0,764	0.45462
		1,236	0.65620
2	0,000 - 1,236	0,472	0.58423
		0,764	0.45462
3	0,472 - 1,236	0,764	0.45462
		0,944	0.47608
4	0,472 - 0,944	0,652	0.47935
		0,764	0.45462
5	0,652 - 0,944	0,764	0.45462
		0,833	0.45408
6	0,764 - 0,944	0,833	0.45408
		0,875	0.45921
7	0,764 - 0,875	0,807	0.45298
		0,833	0.45408
8	0,764 - 0,833	0,790	0.45311
		0,807	0.45298
9	0,790 - 0,833	0,807	0.45298
		0,817	0.45321
10	0,790 - 0,817	0,800	0.45296
		0,807	0.45298
11	0,790 - 0,807	0,796	0.45299
		0,800	0.45296
12	0,796 - 0,807	0,800	0.45296
		0,803	0.45296
13	0,800 - 0,807	0,803	0.45296
		0,804	0.45296
14	0,800 - 0,804	0,802	0.45296
		0,803	0.45296
15	0,800 - 0,803	0,801	0.45296
		0,802	0.45296
16	0,801 - 0,803	0,802	0.45296
		0,802	0.45296
17	0,801 - 0,802	0,802	0.45296
		0,802	0.45296
18	0,802 - 0,802	0,802	0.45296
		0,802	0.45296
19	0,802 - 0,802	0,802	0.45296
		0,802	0.45296

	OPTIMUM RESULTS:	0,802	0.45296

ANALYTICAL SOLUTION. GOLDEN SECTION METHOD.

PULSE AMPLITUDE = 0,900

ITERATION NUMBER	DT INTERVAL	DURATION (DT)	PERFORMANCE 1
1	0,000 - 2,000	0,764	0.41202
		1,236	0.71261
2	0,000 - 1,236	0,472	0.52417
		0,764	0.41202
3	0,472 - 1,236	0,764	0.41202
		0,944	0.46225
4	0,472 - 0,944	0,652	0.42557
		0,764	0.41202
5	0,652 - 0,944	0,764	0.41202
		0,833	0.42097
6	0,652 - 0,833	0,721	0.41304
		0,764	0.41202
7	0,721 - 0,833	0,764	0.41202
		0,790	0.41391
8	0,721 - 0,790	0,748	0.41181
		0,764	0.41202
9	0,721 - 0,764	0,738	0.41205
		0,748	0.41181
10	0,738 - 0,764	0,748	0.41181
		0,754	0.41180
11	0,748 - 0,764	0,754	0.41180
		0,758	0.41185
12	0,748 - 0,758	0,752	0.41179
		0,754	0.41180
13	0,748 - 0,754	0,750	0.41179
		0,752	0.41179
14	0,750 - 0,754	0,752	0.41179
		0,752	0.41179
15	0,750 - 0,752	0,751	0.41179
		0,752	0.41179
16	0,750 - 0,752	0,751	0.41179
		0,751	0.41179
17	0,751 - 0,752	0,751	0.41179
		0,751	0.41179
18	0,751 - 0,752	0,751	0.41179
		0,751	0.41179
19	0,751 - 0,751	0,751	0.41179
		0,751	0.41179
	OPTIMUM RESULTS:	0,751	0.41179

ANALYTICAL SOLUTION. GOLDEN SECTION METHOD.

PULSE AMPLITUDE = 1.000

ITERATION NUMBER	DT INTERVAL	DURATION (DT)	PERFORMANCE I
1	0.000 - 2.000	0.764	0.38204
		1.236	0.79805
2	0.000 - 1.236	0.472	0.46926
		0.764	0.38204
3	0.472 - 1.236	0.764	0.38204
		0.944	0.46684
4	0.472 - 0.944	0.652	0.38125
		0.764	0.38204
5	0.472 - 0.764	0.584	0.40157
		0.652	0.38125
6	0.584 - 0.764	0.652	0.38125
		0.695	0.37671
7	0.652 - 0.764	0.695	0.37671
		0.721	0.37692
8	0.652 - 0.721	0.679	0.37773
		0.695	0.37671
9	0.679 - 0.721	0.695	0.37671
		0.705	0.37652
10	0.695 - 0.721	0.705	0.37652
		0.711	0.37657
11	0.695 - 0.711	0.701	0.37655
		0.705	0.37652
12	0.701 - 0.711	0.705	0.37652
		0.707	0.37652
13	0.701 - 0.707	0.704	0.37653
		0.705	0.37652
14	0.704 - 0.707	0.705	0.37652
		0.706	0.37652
15	0.705 - 0.707	0.706	0.37652
		0.707	0.37652
16	0.705 - 0.707	0.706	0.37652
		0.706	0.37652
17	0.705 - 0.706	0.705	0.37652
		0.706	0.37652
18	0.705 - 0.706	0.706	0.37652
		0.706	0.37652
19	0.706 - 0.706	0.706	0.37652
		0.706	0.37652

OPTIMUM RESULTS: 0.706 0.37652

TABLE OF OPTIMUM RESULTS

AMPLITUDE, P	DURATION, DT	PERFORMANCE, C1
0.10	1.552	1.07293
0.20	1.249	0.92734
0.30	1.154	0.80757
0.40	1.068	0.70875
0.50	0.990	0.62686
0.60	0.920	0.55860
0.70	0.858	0.50133
0.80	0.802	0.45296
0.90	0.751	0.41170
1.00	0.706	0.37652

A.4.2. QUADOPT

This program solves the theoretical result using the simple Quadratic Fit optimisation algorithm.

```
      MASTER QUADOPT
      DIMENSION T(3),P(3),PO(10),TO(10),FO(10)
      COMMON PF
      J=0
    1 READ(1,110)PF,DT
      IF(PF.LE.0.0)GO TO 6
      J=J+1
      PO(J)=PF
      WRITE(2,100)
      WRITE(2,101)PF
      I=0
    2 I=I+1
      CALL CALC(PERF,DT)
      T(I)=DT
      P(I)=PERF
      IF(I.EQ.1)GO TO 3
      IF(I.GT.2)GO TO 4
    3 WRITE(2,102)T(I),P(I)
      DT=DT+0.1
      GO TO 2
    4 IF(P(3).GE.P(2))GO TO 5
      P(I-2)=P(I-1)
      P(I-1)=P(I)
      T(I-2)=T(I-1)
      T(I-1)=T(I)
      I=I-1
      GO TO 3
    5 WRITE(2,102)T(I),P(I)
      A1=(P(1)-P(2))/(T(1)-T(2))
      A2=(P(1)-P(3))/(T(1)-T(3))
      A=(A1-A2)/(T(2)-T(3))
      B=A1-(T(1)+T(2))*A
      C=P(1)-B*T(1)-A*T(1)*T(1)
      TM=-B/(2.0*A)
      PM=A*TM*TM+B*TM+C
      WRITE(2,103)
      WRITE(2,104)TM,PM
      TO(J)=TM
      FO(J)=PM
      GO TO 1
    6 WRITE(2,107)
      WRITE(2,108)
      DO 7 I=1,J
    7 WRITE(2,109)PO(I),TO(I),FO(I)
      STOP
  100 FORMAT(1H1,///,12X,24H ANALYTICAL SOLUTION.    ,
     123HSIMPLE QUADRATIC METHOD)
  101 FORMAT(1H0,//,22X,18HPULSE AMPLITUDE = ,F5.3,//)
  102 FORMAT(1H0,15X,5HDT = ,F4.1,23H GIVES A VALUE OF C1 = ,
     2F7.4)
  103 FORMAT(1H0,//,18X,21H OPTIMUM PERFORMANCE:)
  104 FORMAT(1H0,12X,5HDT = ,F7.4,17H GIVES AN OPTIMUM,
     315H VALUE OF C1 = ,F8.5)
  107 FORMAT(1H1,//,17X,24HTABLE OF OPTIMUM RESULTS)
  108 FORMAT(1H0,5X,12HAMPLITUDE, P,4X,12HDURATION, DT,
     44X,15HPERFORMANCE, C1,/)
  109 FORMAT(1H0,3X,F5.2,11X,F6.3,9X,F8.5)
  110 FORMAT(F5.2,3X,F4.1)
      END
```

Program A4.2 QUADOPT

```
SUBROUTINE CALC(C,D)
COMMON PF
F=0.25*D
Q=PF*PF
R=SQRT(15.0)
X=(19.0*Q-2.0*PF+10.0)/8.0
Y=F*Q
E=EXP(-F)
S=((2.0*PF-19.0*Q)/8.0)*COS(R*F)
Z=((27.0*Q+78.0*PF)/(8.0*R))*SIN(R*F)
C=X+Y+E*(S-Z)
RETURN
END
```

Program A4.2 QUADOPT (continued)

ANALYTICAL SOLUTION.   SIMPLE QUADRATIC METHOD

PULSE AMPLITUDE = 0.100

DT = 1.0 GIVES A VALUE OF C1 = 1.0847

DT = 1.1 GIVES A VALUE OF C1 = 1.0789

DT = 1.2 GIVES A VALUE OF C1 = 1.0751

DT = 1.3 GIVES A VALUE OF C1 = 1.0732

DT = 1.4 GIVES A VALUE OF C1 = 1.0731

DT = 1.5 GIVES A VALUE OF C1 = 1.0748

OPTIMUM PERFORMANCE:

DT = 1.3527 GIVES AN OPTIMUM VALUE OF C1 = 1.07293

ANALYTICAL SOLUTION.   SIMPLE QUADRATIC METHOD

PULSE AMPLITUDE = 0.200

DT = 0.9 GIVES A VALUE OF C1 = 0.9519

DT = 1.0 GIVES A VALUE OF C1 = 0.9397

DT = 1.1 GIVES A VALUE OF C1 = 0.9317

DT = 1.2 GIVES A VALUE OF C1 = 0.9278

DT = 1.3 GIVES A VALUE OF C1 = 0.9273

OPTIMUM PERFORMANCE:

DT = 1.2495 GIVES AN OPTIMUM VALUE OF C1 = 0.92735

ANALYTICAL SOLUTION. SIMPLE QUADRATIC METHOD

PULSE AMPLITUDE = 0.300

DT = 0.8 GIVES A VALUE OF C1 = 0.8482

DT = 0.9 GIVES A VALUE OF C1 = 0.8283

DT = 1.0 GIVES A VALUE OF C1 = 0.8151

DT = 1.1 GIVES A VALUE OF C1 = 0.8085

DT = 1.2 GIVES A VALUE OF C1 = 0.8082

DT = 1.3 GIVES A VALUE OF C1 = 0.8140

OPTIMUM PERFORMANCE:

DT = 1.1546 GIVES AN OPTIMUM VALUE OF C1 = 0.80758

ANALYTICAL SOLUTION. SIMPLE QUADRATIC METHOD

PULSE AMPLITUDE = 0.400

DT = 0.7 GIVES A VALUE OF C1 = 0.7715

DT = 0.8 GIVES A VALUE OF C1 = 0.7417

DT = 0.9 GIVES A VALUE OF C1 = 0.7215

DT = 1.0 GIVES A VALUE OF C1 = 0.7103

DT = 1.1 GIVES A VALUE OF C1 = 0.7092

DT = 1.2 GIVES A VALUE OF C1 = 0.7163

OPTIMUM PERFORMANCE:

DT = 1.0687 GIVES AN OPTIMUM VALUE OF C1 = 0.70877

ANALYTICAL SOLUTION. SIMPLE QUADRATIC METHOD

PULSE AMPLITUDE = 0.500

DT = 0.6 GIVES A VALUE OF C1 = 0.7215

DT = 0.7 GIVES A VALUE OF C1 = 0.6787

DT = 0.8 GIVES A VALUE OF C1 = 0.6439

DT = 0.9 GIVES A VALUE OF C1 = 0.6318

DT = 1.0 GIVES A VALUE OF C1 = 0.6269

DT = 1.1 GIVES A VALUE OF C1 = 0.6338

OPTIMUM PERFORMANCE:

DT = 0.9912 GIVES AN OPTIMUM VALUE OF C1 = 0.62686

ANALYTICAL SOLUTION. SIMPLE QUADRATIC METHOD

PULSE AMPLITUDE = 0.600

DT = 0.6 GIVES A VALUE OF C1 = 0.6401

DT = 0.7 GIVES A VALUE OF C1 = 0.5963

DT = 0.8 GIVES A VALUE OF C1 = 0.5699

DT = 0.9 GIVES A VALUE OF C1 = 0.5580

DT = 1.0 GIVES A VALUE OF C1 = 0.5633

OPTIMUM PERFORMANCE:

DT = 0.9212 GIVES AN OPTIMUM VALUE OF C1 = 0.55855

ANALYTICAL SOLUTION. SIMPLE QUADRATIC METHOD

PULSE AMPLITUDE = 0.700

DT = 0.5 GIVES A VALUE OF C1 = 0.6285

DT = 0.6 GIVES A VALUE OF C1 = 0.5667

DT = 0.7 GIVES A VALUE OF C1 = 0.5256

DT = 0.8 GIVES A VALUE OF C1 = 0.5046

DT = 0.9 GIVES A VALUE OF C1 = 0.5030

DT = 1.0 GIVES A VALUE OF C1 = 0.5201

OPTIMUM PERFORMANCE:

DT = 0.8585 GIVES AN OPTIMUM VALUE OF C1 = 0.50138

ANALYTICAL SOLUTION. SIMPLE QUADRATIC METHOD

PULSE AMPLITUDE = 0.800

DT = 0.5 GIVES A VALUE OF C1 = 0.5627

DT = 0.6 GIVES A VALUE OF C1 = 0.5015

DT = 0.7 GIVES A VALUE OF C1 = 0.4651

DT = 0.8 GIVES A VALUE OF C1 = 0.4530

DT = 0.9 GIVES A VALUE OF C1 = 0.4640

OPTIMUM PERFORMANCE:

DT = 0.8024 GIVES AN OPTIMUM VALUE OF C1 = 0.45295

```
ANALYTICAL SOLUTION.   SIMPLE QUADRATIC METHOD

             PULSE AMPLITUDE = 0.900


      DT =  0.4 GIVES A VALUE OF C1 =  0.5910
      DT =  0.5 GIVES A VALUE OF C1 =  0.5026
      DT =  0.6 GIVES A VALUE OF C1 =  0.4443
      DT =  0.7 GIVES A VALUE OF C1 =  0.4155
      DT =  0.8 GIVES A VALUE OF C1 =  0.4151
      DT =  0.9 GIVES A VALUE OF C1 =  0.4420

          OPTIMUM PERFORMANCE:
DT =  0.7514 GIVES AN OPTIMUM VALUE OF C1 =  0.41137
```

```
ANALYTICAL SOLUTION.   SIMPLE QUADRATIC METHOD

             PULSE AMPLITUDE = 1.000


      DT =  0.4 GIVES A VALUE OF C1 =  0.5365
      DT =  0.5 GIVES A VALUE OF C1 =  0.4482
      DT =  0.6 GIVES A VALUE OF C1 =  0.3953
      DT =  0.7 GIVES A VALUE OF C1 =  0.3766
      DT =  0.8 GIVES A VALUE OF C1 =  0.3909

          OPTIMUM PERFORMANCE:
DT =  0.7065 GIVES AN OPTIMUM VALUE OF C1 =  0.37650
```

TABLE OF OPTIMUM RESULTS

AMPLITUDE, P	DURATION, DT	PERFORMANCE, C1
0.10	1.653	1.07293
0.20	1.269	0.92753
0.30	1.155	0.80758
0.40	1.069	0.70877
0.50	0.991	0.62686
0.60	0.921	0.55853
0.70	0.858	0.50133
0.80	0.802	0.48295
0.90	0.751	0.41187
1.00	0.707	0.37650

APPENDIX A5 The Digital Control Hybrid Program and Results

This appendix contains

(i) The Hybrid FORTRAN Program

(ii) The Flow Chart

and (iii) The Results

```
C;      HYBRID PROBLEM
        DIMENSION P(3),T(3)
        I=0
        E=0.0
        POT ACB3,E
        POT ACC3,E
        POT ACB4,E
        POT ACB5,E
        F=0.5
        POT ADB2,F
        POT ADC4,F
        F=0.2*F
        POT ACC4,F
100;    RESET
        I=0
        TYPE 23
        TYPE 25
        TYPE 24
        ACCEPT 26,K
        IF(K)99,98,97
98;     LOGOUT 1,0
        GO TO 96
97;     LOGOUT 1,1
96;     TYPE 27
        ACCEPT 28,RATE
        RATE=0.5*RATE
        DCU ACD2,RATE
101;    TYPE 29
        ACCEPT 28,DT
        IF(DT)102,102,103
102;    TYPE 30
        TYPE 34
        ACCEPT 26,J1
        IF(J1)99,99,100
103;    TPOT=0.1*DT
        DCU ACD1,TPOT
        I=I+1
        ICOND
        COMPUTE
104;    LOGIN 1,L
        IF(L)104,104,105
105;    HOLD
        LOGIN 2,L
        IF(L)107,107,106
106;    TYPE 31
        I=0
        GO TO 101
107;    T(I)=DT
```

Program A5 The PDP8-L Program

```
      DVM AAB6,OUTPT
      PERF=2.0*OUTPT
      P(I)=PERF
      IF(I-1)99,108,109
108;  TYPE 32,DT,PERF
      DTIN=DT/10.0
      DT=DT+DTIN
      GO TO 103
109;  IF(I-2)99,110,113
110;  TYPE 32,DT,PERF
      IF(P(2)-P(1))112,112,111
111;  TYPE 33
      I=0
      GO TO 101
112;  DT=DT+DTIN
      GO TO 103
113;  IF(P(3)-P(2))114,115,115
114;  P(1)=P(2)
      P(2)=P(3)
      T(1)=T(2)
      T(2)=T(3)
      I=I-1
      TYPE 32,DT,PERF
      GO TO 112
115;  TYPE 32,DT,PERF
      A=(((P(1)-P(2))/(T(1)-T(2)))-((P(1)-P(3))/(T(1)-T(3))))
      A=A/(T(2)-T(3))
      B=((P(1)-P(2))/(T(1)-T(2)))-((T(1)+T(2))*A)
      C=P(1)-(B*T(1))-(A*(T(1)*T(1)))
      TM=-B/(A*2.0)
      PM=(A*(TM*TM))+(B*TM)+C
      TYPE 37,TM,PM
      GO TO 102
23;   FORMAT(/,/,/,"TYPE 0 FOR SQUARE CRITERION")
25;   FORMAT(/,"TYPE 1 FOR TIME-SQUARE CRITERION",/)
26;   FORMAT(I)
24;   FORMAT("TYPE -1 FOR STOP",/,"INPUT = ")
27;   FORMAT(/,"RATE SETTING = ")
28;   FORMAT(E)
29;   FORMAT(/,"DURATION TIME DT = ")
30;   FORMAT(/,/,"TYPE -1 FOR EXIT, +1 FOR RESTART",/)
34;   FORMAT("INPUT = ")
31;   FORMAT(/,"OVERLOAD - RESET DT",/)
32;   FORMAT(/,"DT = ",M,"PERFORMANCE = ",M)
33;   FORMAT(/,"SECOND RUN INDICATES TRY SMALLER DT",/)
37;   FORMAT(/,/,"OPTIMUM VALUES",/,"DT = ",M,'
      "PERFORMANCE = ",M)
99;   STOP
      END
```

Program A5 The PDP8-L Program (continued)

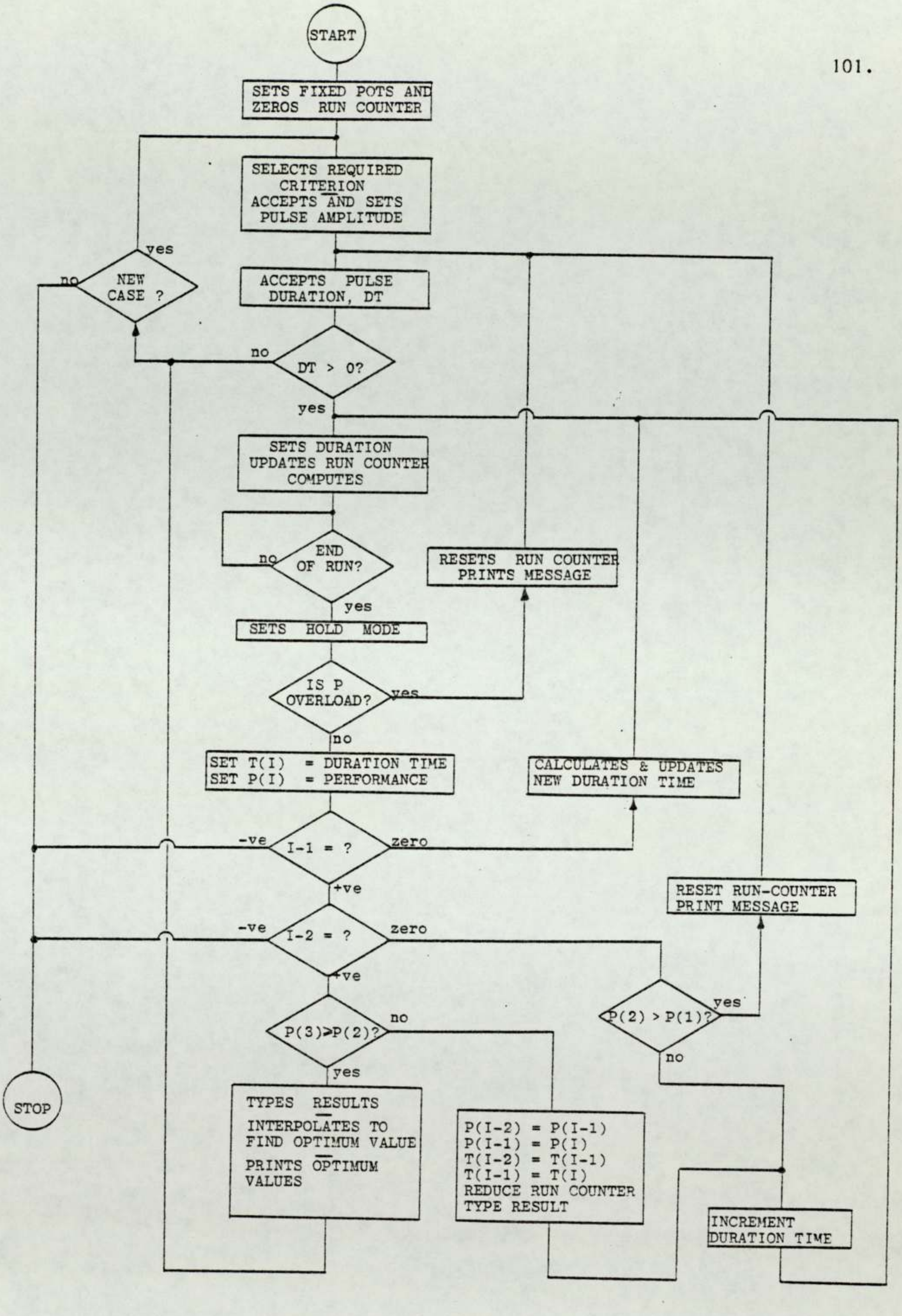

Figure A5.2.1. The PDP8-L Flow Chart

```
TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 0

RATE SETTING = 0.1

DURATION TIME DT = 1.0

DT = +1.0000      PERFORMANCE = +0.9994
DT = +1.1000      PERFORMANCE = +0.9972
DT = +1.2000      PERFORMANCE = +0.9962
DT = +1.3000      PERFORMANCE = +0.9968

OPTIMUM VALUES
DT = +1.2125      PERFORMANCE = +0.9962

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1


TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 0

RATE SETTING = 0.2

DURATION TIME DT = 1.0

DT = +1.0000      PERFORMANCE = +0.8624
DT = +1.1000      PERFORMANCE = +0.8590
DT = +1.2000      PERFORMANCE = +0.8578
DT = +1.3000      PERFORMANCE = +0.8608

OPTIMUM VALUES
DT = +1.1786      PERFORMANCE = +0.8577

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1
```

```
TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 0

RATE SETTING = 0.3

DURATION TIME DT = 0.9

DT = +0.9000      PERFORMANCE = +0.7572
DT = +0.9900      PERFORMANCE = +0.7504
DT = +1.0800      PERFORMANCE = +0.7472
DT = +1.1700      PERFORMANCE = +0.7484

OPTIMUM VALUES
DT = +1.1005      PERFORMANCE = +0.7471

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1


TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 0

RATE SETTING = 0.4

DURATION TIME DT = 0.8

DT = +0.8000      PERFORMANCE = +0.6776
DT = +0.8800      PERFORMANCE = +0.6672
DT = +0.9600      PERFORMANCE = +0.6614
DT = +1.0400      PERFORMANCE = +0.6600
DT = +1.1200      PERFORMANCE = +0.6636

OPTIMUM VALUES
DT = +1.0224      PERFORMANCE = +0.6599

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1
```

```
TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 0

RATE SETTING = 0.5

DURATION TIME DT = 0.8

DT = +0.8000      PERFORMANCE = +0.5980
DT = +0.8800      PERFORMANCE = +0.5890
DT = +0.9600      PERFORMANCE = +0.5868
DT = +1.0400      PERFORMANCE = +0.5912

OPTIMUM VALUES
DT = +0.9467      PERFORMANCE = +0.5867

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1


TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 0

RATE SETTING = 0.6

DURATION TIME DT = 0.7

DT = +0.7000      PERFORMANCE = +0.5452
DT = +0.7700      PERFORMANCE = +0.5322
DT = +0.8400      PERFORMANCE = +0.5258
DT = +0.9100      PERFORMANCE = +0.5256
DT = +0.9800      PERFORMANCE = +0.5320

OPTIMUM VALUES
DT = +0.8771      PERFORMANCE = +0.5249

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1
```

```
TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 0

RATE SETTING = 0.7

DURATION TIME DT = 0.7

DT = +0.7000      PERFORMANCE = +0.4836
DT = +0.7700      PERFORMANCE = +0.4740
DT = +0.8400      PERFORMANCE = +0.4726
DT = +0.9100      PERFORMANCE = +0.4798

OPTIMUM VALUES
DT = +0.8164      PERFORMANCE = +0.4721

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1


TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 0

RATE SETTING = 0.8

DURATION TIME DT = 0.6

DT = +0.6000      PERFORMANCE = +0.4548
DT = +0.6600      PERFORMANCE = +0.4376
DT = +0.7200      PERFORMANCE = +0.4282
DT = +0.7800      PERFORMANCE = +0.4274
DT = +0.8400      PERFORMANCE = +0.4342

OPTIMUM VALUES
DT = +0.7563      PERFORMANCE = +0.4268

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1
```

```
TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 0

RATE SETTING = 0.9

DURATION TIME DT = 0.5

DT = +0.5000      PERFORMANCE = +0.4452
DT = +0.5500      PERFORMANCE = +0.4218
DT = +0.6000      PERFORMANCE = +0.4038
DT = +0.6500      PERFORMANCE = +0.3924
DT = +0.7000      PERFORMANCE = +0.3880
DT = +0.7500      PERFORMANCE = +0.3902

OPTIMUM VALUES
DT = +0.7083      PERFORMANCE = +0.3879

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1


TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 0

RATE SETTING = 1.0

DURATION TIME DT = 0.5

DT = +0.5000      PERFORMANCE = +0.3960
DT = +0.5500      PERFORMANCE = +0.3746
DT = +0.6000      PERFORMANCE = +0.3606
DT = +0.6500      PERFORMANCE = +0.3544
DT = +0.7000      PERFORMANCE = +0.3562

OPTIMUM VALUES
DT = +0.6637      PERFORMANCE = +0.3541

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1
```

```
TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 1

RATE SETTING = 0.1

DURATION TIME DT = 1.1

DT = +1.1000      PERFORMANCE = +1.3704
DT = +1.2100      PERFORMANCE = +1.3656
DT = +1.3200      PERFORMANCE = +1.3636
DT = +1.4300      PERFORMANCE = +1.3700

OPTIMUM VALUES
DT = +1.2912      PERFORMANCE = +1.3633

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1


TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 1

RATE SETTING = 0.2

DURATION TIME DT = 1.0

DT = +1.0000      PERFORMANCE = +1.1016
DT = +1.1000      PERFORMANCE = +1.0856
DT = +1.2000      PERFORMANCE = +1.0790
DT = +1.3000      PERFORMANCE = +1.0830

OPTIMUM VALUES
DT = +1.2123      PERFORMANCE = +1.0789

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1
```

```
TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 1

RATE SETTING = 0.3

DURATION TIME DT = 0.9

DT = +0.9000       PERFORMANCE = +0.9078
DT = +0.9900       PERFORMANCE = +0.8848
DT = +1.0800       PERFORMANCE = +0.8716
DT = +1.1700       PERFORMANCE = +0.8684
DT = +1.2600       PERFORMANCE = +0.8776

OPTIMUM VALUES
DT = +1.1482       PERFORMANCE = +0.8680

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1


TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 1

RATE SETTING = 0.4

DURATION TIME DT = 0.9

DT = +0.9000       PERFORMANCE = +0.7408
DT = +0.9900       PERFORMANCE = +0.7248
DT = +1.0800       PERFORMANCE = +0.7204
DP = +1.1700       PERFORMANCE = +0.7300

OPTIMUM VALUES
DT = +1.0633       PERFORMANCE = +0.7202

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1
```

```
TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 1

RATE SETTING = 0.5

DURATION TIME DT = 0.8

DT = +0.8000       PERFORMANCE = +0.6378
DT = +0.8800       PERFORMANCE = +0.6148
DT = +0.9600       PERFORMANCE = +0.6050
DT = +1.0400       PERFORMANCE = +0.6100

OPTIMUM VALUES
DT = +0.9730       PERFORMANCE = +0.6048

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1


TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 1

RATE SETTING = 0.6

DURATION TIME DT = 0.75

DT = +0.7500       PERFORMANCE = +0.5424
DT = +0.8250       PERFORMANCE = +0.5212
DT = +0.9000       PERFORMANCE = +0.5126
DT = +0.9750       PERFORMANCE = +0.5216

OPTIMUM VALUES
DT = +0.8991       PERFORMANCE = +0.5126

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1
```

```
TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 1

RATE SETTING = 0.7

DURATION TIME DT = 0.7

DT = +0.7000      PERFORMANCE = +0.4672
DT = +0.7700      PERFORMANCE = +0.4436
DT = +0.8400      PERFORMANCE = +0.4374
DT = +0.9100      PERFORMANCE = +0.4508

OPTIMUM VALUES
DT = +0.8271      PERFORMANCE = +0.4371

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1


TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 1

RATE SETTING = 0.8

DURATION TIME DT = 0.6

DT = +0.6000      PERFORMANCE = +0.4388
DT = +0.6600      PERFORMANCE = +0.4012
DT = +0.7200      PERFORMANCE = +0.3806
DT = +0.7800      PERFORMANCE = +0.3762
DT = +0.8400      PERFORMANCE = +0.3882

OPTIMUM VALUES
DT = +0.7661      PERFORMANCE = +0.3758

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1
```

```
TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 1

RATE SETTING = 0.9

DURATION TIME DT = 0.5

DT = +0.5000      PERFORMANCE = +0.4496
DT = +0.5500      PERFORMANCE = +0.3992
DT = +0.6000      PERFORMANCE = +0.3608
DT = +0.6500      PERFORMANCE = +0.3362
DT = +0.7000      PERFORMANCE = +0.3252
DT = +0.7500      PERFORMANCE = +0.3282

OPTIMUM VALUES
DT = +0.7143      PERFORMANCE = +0.3246

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = 1


TYPE 0 FOR SQUARE CRITERION
TYPE 1 FOR TIME-SQUARE CRITERION
TYPE -1 FOR STOP
INPUT = 1

RATE SETTING = 1.0

DURATION TIME DT = 0.5

DT = +0.5000      PERFORMANCE = +0.3714
DT = +0.5500      PERFORMANCE = +0.3266
DT = +0.6000      PERFORMANCE = +0.2978
DT = +0.6500      PERFORMANCE = +0.2826
DT = +0.7000      PERFORMANCE = +0.2846

OPTIMUM VALUES
DT = +0.6692      PERFORMANCE = +0.2813

TYPE -1 FOR EXIT, +1 FOR RESTART
INPUT = -1
```

APPENDIX A6 The SLAM Results

The appendix contains the program listing and sets of results for the SLAMOPT program.

S L A M   T R A N S L A T I O N   B Y   # X 2 S T   M A R K   1 . 4

SOURCE INPUT LISTING

```
              OPTIONS FLIST,DEBUG 1,EXPLICIT
              FORTRAN(ED,SLAM OUTPUT,DSHM)
              PROGDESC
              LIBRARY(SUBGROUPSLAM)
              LIST
         ..   PROGRAM(DSHM)
              INPUT1 = CR0
              OUTPUT2 = LP0
              COMPRESS INTEGER AND LOGICAL
              COMPACT
              END
              MASTER SLAMOPT
              REAL INC,LOWER,LOWC
              DATA INC/0.25/,FIVE/5.0/,TERM/30.0/
              INITIAL
              TT=(SQRT(FIVE)-1.0)/2.0
              INPUT AMP,IP,LOWER,UPPER
              IT=1
              NOSORT
              IF(AMP)13,12,12
           12 CONTINUE
              WRITE(2,110)
              WRITE(2,111)AMP
              IF(UPPER-LOWER-1.E-4)21,21,26
           21 CONTINUE
              LOWER=0.0
           15 UPPER=LOWER+INC
              END
              PERFORM SEG1(LOWC=LOWER,AMP,IP,TERM,IT)
              PERFORM SEG1(HIGHC=UPPER,AMP,IP,TERM,IT)
              NOSORT
              IF(UPPER-3.0)11,11,13
           11 IF(HIGHC-LOWC-1.E-6)27,27,28
           27 CONTINUE
              LOWER=UPPER
              IT=IT+1
              GO TO 15
           28 IF(IT.EQ.1)GO TO 26
              LOWER=UPPER-2.0*INC
              IT=1
           26 CONTINUE
              END
              DT1=LOWER+(1.0-TT)*(UPPER-LOWER)
              DT2=LOWER+TT*(UPPER-LOWER)
              PERFORM SEG1(PERFY=DT1,AMP,IP,TERM,IT)
              PERFORM SEG1(PERFZ=DT2,AMP,IP,TERM,IT)
              NOSORT
              WRITE(2,94)IP
              PERFA=PERFY
              PERFB=PERFZ
            8 CONTINUE
```

Program A6 SLAMOPT

```
S L A M   T R A N S L A T I O N   B Y   # X 2 S T   M A R K   1 . 4
------------------------------------------------------------------------

          WRITE(2,97)IT,LOWER,UPPER,DT1,PERFA
          WRITE(2,98)DT2,PERFB
          IT=IT+1
          IF(ABS(DT2-DT1)-1.E-4)9,10,10
       10 CONTINUE
          IF(PERFA-PERFB)6,5,7
        5 LOWER=DT1
          UPPER=DT2
          GO TO 26
        6 UPPER=DT2
          PERFB=PERFA
          DT2=DT1
          DT1=LOWER*(1,0-TT)*(UPPER-LOWER)
          END
          PERFORM SEG1(PERFA=DT1,AMP,IP,TERM,IT)
          NOSORT
          GO TO 8
        7 LOWER=DT1
          PERFA=PERFB
          DT1=DT2
          DT2=LOWER*TT*(UPPER-LOWER)
          END
          PERFORM SEG1(PERFB=DT2,AMP,IP,TERM,IT)
          END
          GO TO 8
        9 WRITE(2,99)DT1,IP,PERFA
          TERMINAL
          REPEAT
          END
       13 CONTINUE
          WRITE(2,112)
       94 FORMAT(1H ,5X,28HITERATION  DURATION-INTERVAL,4X,
         123HDURATION   PERFORMANCE,I1,/,9X,2HNO,6X,5HLOWER,
         25X,5HUPPER,7X,4H(DT),9X,5HF(DT))
       97 FORMAT(1H ,8X,I2,5X,F7.4,3X,F7.4,5X,F7.4,6X,F7.5)
       98 FORMAT(1H ,37X,F7.4,6X,F7.5)
       99 FORMAT(//,5X,16HOPTIMUM RESULTS:,2X,6HPULSE ,
         310HDURATION =,F7.4,/,24X,12HPERFORMANCE ,I1,
         42H =,F8.5)
      110 FORMAT(1H1,///,21X,22HOPTIMISATION WITH SLAM,/)
      111 FORMAT(1H ,5X,17HPULSE AMPLITUDE =,F7.4)
      112 FORMAT(1H1,16H  END OF RESULTS)
          END
          SEGMENT SEG1(PERF=DT,PL2,IP,TERM,IT)
          INITIAL
          PERF=0.0
          CINT=DT
          PDT0=0.0
          PL0=0.0
          MINS=20
          IALG=1
          END
          JJ=DT
```

Program A6   SLAMOPT   (continued)

S L A M   T R A N S L A T I O N   B Y   # X 2 S T   M A R K   1 . 4

```
          WDIGIT=JJ+1
          JSW=0
          DYNAMIC
          DERIVATIVE
          TIME=INTGRL(1.0,0.0)
          P=SPULSE(TIME,PDTO,DT,PLO,PL2)
          PNEG=PROD(P,-1.0)
          YDOT2=SUM(PNEG,Y1)
          Y2=INTGRL(YDOT2,1.0)
          Y2NEG=PROD(Y2,-1.0)
          HY1=PROD(Y1,-0.5)
          YDOT1=SUM(Y2NEG,HY1)
          Y1=INTGRL(YDOT1,0.0)
          CADOT=PROD(Y2,Y2)
          CBDOT=PROD(CADOT,TIME)
          CA=INTGRL(CADOT,0.0)
          CB=INTGRL(CBDOT,0.0)
          INTINF
          STEPS:MINS
          CI:CINT
          ALG:IALG=RKFS,TRPZ,SIMP,ADMN,RKVS
          INDVAR:TIME
          END
          END
          PARALLEL
          NOSORT
          IF(TIME-DT)32,34,34
       34 IF(JSW)31,31,30
       31 CINT=WDIGIT-DT
          JSW=1
          GO TO 32
       30 CINT=1.0
          MINS=10
       32 CONTINUE
          END
          END
          TERMINATE(TIME.GT.TERM)
          END
          TERMINAL
          NOSORT
          IF(IP.EQ.1)PERF=CA
          IF(IP.EQ.2)PERF=CB
          END
          END
          END
          FINISH

     ****  -  END OF INPUT
```

Program A6 SLAMOPT (concluded)

OPTIMISATION WITH SLAM

PULSE AMPLITUDE = 0,1000

ITERATION NO	DURATION-INTERVAL LOWER	DURATION-INTERVAL UPPER	DURATION (DT)	PERFORMANCE 1 F(DT)
1	0,0000	2,0000	0,7639	1,10712
			1,2361	1,07440
2	0,7639	2,0000	1,2361	1,07440
			1,5279	1,07524
3	0,7639	1,5279	1,0557	1,08171
			1,2361	1,07440
4	1,0557	1,5279	1,2361	1,07440
			1,3475	1,07294
5	1,2361	1,5279	1,3475	1,07294
			1,4164	1,07316
6	1,2361	1,4164	1,3050	1,07323
			1,3475	1,07294
7	1,3050	1,4164	1,3475	1,07294
			1,3738	1,07293
8	1,3475	1,4164	1,3738	1,07293
			1,3901	1,07298
9	1,3475	1,3901	1,3638	1,07292
			1,3738	1,07293
10	1,3475	1,3738	1,3576	1,07292
			1,3638	1,07292
11	1,3576	1,3738	1,3638	1,07292
			1,3676	1,07292
12	1,3576	1,3676	1,3614	1,07292
			1,3638	1,07292
13	1,3614	1,3676	1,3638	1,07292
			1,3653	1,07292
14	1,3614	1,3653	1,3629	1,07292
			1,3638	1,07292
15	1,3629	1,3653	1,3638	1,07292
			1,3643	1,07292
16	1,3629	1,3643	1,3634	1,07292
			1,3638	1,07292
17	1,3634	1,3643	1,3638	1,07292
			1,3640	1,07292
18	1,3634	1,3640	1,3637	1,07292
			1,3638	1,07292
19	1,3637	1,3640	1,3638	1,07292
			1,3639	1,07292

OPTIMUM RESULTS: PULSE DURATION = 1,3638
PERFORMANCE 1 = 1,07292

OPTIMISATION WITH SLAM

PULSE AMPLITUDE = 0,2000

ITERATION NO	DURATION-INTERVAL LOWER	UPPER	DURATION (DT)	PERFORMANCE 1 F(DT)
1	0,0000	2,0000	0,7639	0,97668
			1,2361	0,92743
2	0,7639	2,0000	1,2361	0,92743
			1,5279	0,94032
3	0,7639	1,5279	1,0557	0,93543
			1,2361	0,92743
4	1,0557	1,5279	1,2361	0,92743
			1,3475	0,92877
5	1,0557	1,3475	1,1672	0,92897
			1,2361	0,92743
6	1,1672	1,3475	1,2361	0,92743
			1,2786	0,92739
7	1,2361	1,3475	1,2786	0,92739
			1,3050	0,92771
8	1,2361	1,3050	1,2624	0,92733
			1,2786	0,92739
9	1,2361	1,2786	1,2523	0,92734
			1,2624	0,92733
10	1,2523	1,2786	1,2624	0,92733
			1,2686	0,92734
11	1,2523	1,2686	1,2585	0,92733
			1,2624	0,92733
12	1,2523	1,2624	1,2562	0,92733
			1,2585	0,92733
13	1,2562	1,2624	1,2585	0,92733
			1,2600	0,92733
14	1,2585	1,2624	1,2600	0,92733
			1,2609	0,92733
15	1,2585	1,2609	1,2594	0,92733
			1,2600	0,92733
16	1,2585	1,2600	1,2591	0,92733
			1,2594	0,92733
17	1,2591	1,2600	1,2594	0,92733
			1,2597	0,92733
18	1,2594	1,2600	1,2597	0,92733
			1,2598	0,92733
19	1,2594	1,2598	1,2596	0,92733
			1,2597	0,92733

OPTIMUM RESULTS: PULSE DURATION = 1,2596
PERFORMANCE 1 = 0,92733

OPTIMISATION WITH SLAM

PULSE AMPLITUDE = 0.3000

ITERATION NO	DURATION-INTERVAL LOWER	UPPER	DURATION (DT)	PERFORMANCE 1 F(DT)
1	0.0000	2.0000	0.7639	0.85868
			1.2361	0.80911
2	0.7639	2.0000	1.2361	0.80911
			1.5279	0.84525
3	0.7639	1.5279	1.0557	0.81117
			1.2361	0.80911
4	1.0557	1.5279	1.2361	0.80911
			1.3475	0.81748
5	1.0557	1.3475	1.1672	0.80755
			1.2361	0.80911
6	1.0557	1.2361	1.1246	0.80802
			1.1672	0.80755
7	1.1246	1.2361	1.1672	0.80755
			1.1935	0.80781
8	1.1246	1.1935	1.1509	0.80760
			1.1672	0.80755
9	1.1509	1.1935	1.1672	0.80755
			1.1772	0.80760
10	1.1509	1.1772	1.1610	0.80755
			1.1672	0.80755
11	1.1509	1.1672	1.1571	0.80756
			1.1610	0.80755
12	1.1571	1.1672	1.1610	0.80755
			1.1633	0.80755
13	1.1610	1.1672	1.1633	0.80755
			1.1648	0.80755
14	1.1610	1.1648	1.1624	0.80755
			1.1633	0.80755
15	1.1624	1.1648	1.1633	0.80755
			1.1639	0.80755
16	1.1633	1.1648	1.1639	0.80755
			1.1643	0.80755
17	1.1633	1.1643	1.1637	0.80755
			1.1639	0.80755
18	1.1637	1.1643	1.1639	0.80755
			1.1640	0.80755
19	1.1639	1.1643	1.1640	0.80755
			1.1641	0.80755

OPTIMUM RESULTS: PULSE DURATION = 1.1640
PERFORMANCE 1 = 0.80755

OPTIMISATION WITH SLAM

PULSE AMPLITUDE = 0.4000

ITERATION NO	DURATION-INTERVAL LOWER	UPPER	DURATION (DT)	PERFORMANCE 1 F(DT)
1	0.0000	2.0000	0.7639	0.75311
			1.2361	0.71942
2	0.7639	2.0000	1.2361	0.71942
			1.5279	0.79001
3	0.7639	1.5279	1.0557	0.70893
			1.2361	0.71942
4	0.7639	1.2361	0.9443	0.71655
			1.0557	0.70893
5	0.9443	1.2361	1.0557	0.70893
			1.1246	0.70970
6	0.9443	1.1246	1.0132	0.71053
			1.0557	0.70893
7	1.0132	1.1246	1.0557	0.70893
			1.0820	0.70874
8	1.0557	1.1246	1.0820	0.70874
			1.0983	0.70892
9	1.0557	1.0983	1.0720	0.70874
			1.0820	0.70874
10	1.0720	1.0983	1.0820	0.70874
			1.0883	0.70878
11	1.0720	1.0883	1.0782	0.70873
			1.0820	0.70874
12	1.0720	1.0820	1.0758	0.70873
			1.0782	0.70873
13	1.0758	1.0820	1.0782	0.70873
			1.0797	0.70873
14	1.0758	1.0797	1.0773	0.70873
			1.0782	0.70873
15	1.0758	1.0782	1.0767	0.70873
			1.0773	0.70873
16	1.0767	1.0782	1.0773	0.70873
			1.0776	0.70873
17	1.0767	1.0776	1.0771	0.70873
			1.0773	0.70873
18	1.0771	1.0776	1.0773	0.70873
			1.0774	0.70873
19	1.0771	1.0774	1.0772	0.70873
			1.0773	0.70873

OPTIMUM RESULTS: PULSE DURATION = 1.0772
PERFORMANCE 1 = 0.70873

OPTIMISATION WITH SLAM

PULSE AMPLITUDE = 0.5000

ITERATION NO	DURATION-INTERVAL LOWER	UPPER	DURATION (DT)	PERFORMANCE 1 F(DT)
1	0.0000	2.0000	0.7639	0.65998
			1.2361	0.65838
2	0.7639	2.0000	1.2361	0.65838
			1.5279	0.77462
3	0.7639	1.5279	1.0557	0.62871
			1.2361	0.65838
4	0.7639	1.2361	0.9443	0.62858
			1.0557	0.62871
5	0.7639	1.0557	0.8754	0.63587
			0.9443	0.62858
6	0.8754	1.0557	0.9443	0.62858
			0.9868	0.62692
7	0.9443	1.0557	0.9868	0.62692
			1.0132	0.62696
8	0.9443	1.0132	0.9706	0.62730
			0.9868	0.62692
9	0.9706	1.0132	0.9868	0.62692
			0.9969	0.62684
10	0.9868	1.0132	0.9969	0.62684
			1.0031	0.62685
11	0.9868	1.0031	0.9931	0.62686
			0.9969	0.62684
12	0.9931	1.0031	0.9969	0.62684
			0.9993	0.62684
13	0.9969	1.0031	0.9993	0.62684
			1.0007	0.62684
14	0.9969	1.0007	0.9984	0.62684
			0.9993	0.62684
15	0.9969	0.9993	0.9978	0.62684
			0.9984	0.62684
16	0.9978	0.9993	0.9984	0.62684
			0.9987	0.62684
17	0.9984	0.9993	0.9987	0.62684
			0.9989	0.62684
18	0.9984	0.9989	0.9986	0.62684
			0.9987	0.62684
19	0.9986	0.9989	0.9987	0.62684
			0.9988	0.62684

OPTIMUM RESULTS: PULSE DURATION = 0.9987
PERFORMANCE 1 = 0.62684

OPTIMISATION WITH SLAM

PULSE AMPLITUDE = 0.6000

ITERATION NO	DURATION-INTERVAL LOWER	UPPER	DURATION (DT)	PERFORMANCE 1 F(DT)
1	0.0000	2.0000	0.7639	0.57929
			1.2361	0.62597
2	0.0000	1.2361	0.4721	0.72267
			0.7639	0.57929
3	0.4721	1.2361	0.7639	0.57929
			0.9443	0.55877
4	0.7639	1.2361	0.9443	0.55877
			1.0557	0.57051
5	0.7639	1.0557	0.8754	0.56070
			0.9443	0.55877
6	0.8754	1.0557	0.9443	0.55877
			0.9868	0.56112
7	0.8754	0.9868	0.9180	0.55866
			0.9443	0.55877
8	0.8754	0.9443	0.9017	0.55911
			0.9180	0.55866
9	0.9017	0.9443	0.9180	0.55866
			0.9280	0.55858
10	0.9180	0.9443	0.9280	0.55858
			0.9342	0.55861
11	0.9180	0.9342	0.9242	0.55859
			0.9280	0.55858
12	0.9242	0.9342	0.9280	0.55858
			0.9304	0.55858
13	0.9242	0.9304	0.9265	0.55858
			0.9280	0.55858
14	0.9265	0.9304	0.9280	0.55858
			0.9289	0.55858
15	0.9265	0.9289	0.9275	0.55858
			0.9280	0.55858
16	0.9275	0.9289	0.9280	0.55858
			0.9284	0.55858
17	0.9280	0.9289	0.9284	0.55858
			0.9286	0.55858
18	0.9280	0.9286	0.9282	0.55858
			0.9284	0.55858
19	0.9282	0.9286	0.9284	0.55858
			0.9284	0.55858

OPTIMUM RESULTS: PULSE DURATION = 0.9284
PERFORMANCE 1 = 0.55858

OPTIMISATION WITH SLAM

PULSE AMPLITUDE = 0.7000

ITERATION NO	DURATION-INTERVAL LOWER	DURATION-INTERVAL UPPER	DURATION (DT)	PERFORMANCE 1 F(DT)
1	0.0000	2.0000	0.7639	0.51103
			1.2361	0.62221
2	0.0000	1.2361	0.4721	0.65252
			0.7639	0.51103
3	0.4721	1.2361	0.7639	0.51103
			0.9443	0.50711
4	0.7639	1.2361	0.9443	0.50711
			1.0557	0.53432
5	0.7639	1.0557	0.8754	0.50142
			0.9443	0.50711
6	0.7639	0.9443	0.8328	0.50230
			0.8754	0.50142
7	0.8328	0.9443	0.8754	0.50142
			0.9017	0.50256
8	0.8328	0.9017	0.8591	0.50135
			0.8754	0.50142
9	0.8328	0.8754	0.8491	0.50156
			0.8591	0.50135
10	0.8491	0.8754	0.8591	0.50135
			0.8653	0.50132
11	0.8591	0.8754	0.8653	0.50132
			0.8692	0.50133
12	0.8591	0.8692	0.8630	0.50132
			0.8653	0.50132
13	0.8630	0.8692	0.8653	0.50132
			0.8668	0.50132
14	0.8630	0.8668	0.8644	0.50132
			0.8653	0.50132
15	0.8644	0.8668	0.8653	0.50132
			0.8659	0.50132
16	0.8644	0.8659	0.8650	0.50132
			0.8653	0.50132
17	0.8644	0.8653	0.8648	0.50132
			0.8650	0.50132
18	0.8648	0.8653	0.8650	0.50132
			0.8651	0.50132
19	0.8650	0.8653	0.8651	0.50132
			0.8652	0.50132

OPTIMUM RESULTS: PULSE DURATION = 0.8651
PERFORMANCE 1 = 0.50132

OPTIMISATION WITH SLAM

PULSE AMPLITUDE = 0.8000

ITERATION NO	DURATION-INTERVAL LOWER	UPPER	DURATION (DT)	PERFORMANCE 1 F(DT)
1	0.0000	2.0000	0.7639	0.45522
			1.2361	0.64708
2	0.0000	1.2361	0.4721	0.58743
			0.7639	0.45522
3	0.4721	1.2361	0.7639	0.45522
			0.9443	0.47361
4	0.4721	0.9443	0.6525	0.48131
			0.7639	0.45522
5	0.6525	0.9443	0.7639	0.45522
			0.8328	0.45362
6	0.7639	0.9443	0.8328	0.45362
			0.8754	0.45802
7	0.7639	0.8754	0.8065	0.45294
			0.8328	0.45362
8	0.7639	0.8328	0.7902	0.45332
			0.8065	0.45294
9	0.7902	0.8328	0.8065	0.45294
			0.8166	0.45302
10	0.7902	0.8166	0.8003	0.45301
			0.8065	0.45294
11	0.8003	0.8166	0.8065	0.45294
			0.8103	0.45294
12	0.8003	0.8103	0.8041	0.45296
			0.8065	0.45294
13	0.8041	0.8103	0.8065	0.45294
			0.8080	0.45294
14	0.8065	0.8103	0.8080	0.45294
			0.8089	0.45294
15	0.8065	0.8089	0.8074	0.45294
			0.8080	0.45294
16	0.8074	0.8089	0.8080	0.45294
			0.8083	0.45294
17	0.8080	0.8089	0.8083	0.45294
			0.8085	0.45294
18	0.8080	0.8085	0.8082	0.45294
			0.8083	0.45294
19	0.8082	0.8085	0.8083	0.45294
			0.8084	0.45294

OPTIMUM RESULTS: PULSE DURATION = 0.8083
PERFURMANCE 1 = 0.45294

OPTIMISATION WITH SLAM

PULSE AMPLITUDE = 0.9000

ITERATION NO	DURATION-INTERVAL LOWER	UPPER	DURATION (DT)	PERFORMANCE 1 F(DT)
1	0.0000	2.0000	0.7639	0.41183
			1.2361	0.70060
2	0.0000	1.2361	0.4721	0.52740
			0.7639	0.41183
3	0.4721	1.2361	0.7639	0.41183
			0.9443	0.45827
4	0.4721	0.9443	0.6525	0.42713
			0.7639	0.41183
5	0.6525	0.9443	0.7639	0.41183
			0.8328	0.41947
6	0.6525	0.8328	0.7214	0.41358
			0.7639	0.41183
7	0.7214	0.8328	0.7639	0.41183
			0.7902	0.41324
8	0.7214	0.7902	0.7477	0.41191
			0.7639	0.41183
9	0.7477	0.7902	0.7639	0.41183
			0.7740	0.41215
10	0.7477	0.7740	0.7577	0.41178
			0.7639	0.41183
11	0.7477	0.7639	0.7539	0.41180
			0.7577	0.41178
12	0.7539	0.7639	0.7577	0.41178
			0.7601	0.41179
13	0.7539	0.7601	0.7563	0.41178
			0.7577	0.41178
14	0.7563	0.7601	0.7577	0.41178
			0.7586	0.41178
15	0.7563	0.7586	0.7572	0.41178
			0.7577	0.41178
16	0.7572	0.7586	0.7577	0.41178
			0.7581	0.41178
17	0.7572	0.7581	0.7575	0.41178
			0.7577	0.41178
18	0.7572	0.7577	0.7574	0.41178
			0.7575	0.41178
19	0.7574	0.7577	0.7575	0.41178
			0.7576	0.41178

OPTIMUM RESULTS: PULSE DURATION = 0.7575
PERFORMANCE 1 = 0.41178

OPTIMISATION WITH SLAM

PULSE AMPLITUDE = 1.0000

ITERATION NO	DURATION-INTERVAL LOWER	UPPER	DURATION (DT)	PERFORMANCE 1 F(DT)
1	0.0000	2.0000	0.7639	0.38089
			1.2361	0.78275
2	0.0000	1.2361	0.4721	0.47244
			0.7639	0.38089
3	0.4721	1.2361	0.7639	0.38089
			0.9443	0.46108
4	0.4721	0.9443	0.6525	0.38225
			0.7639	0.38089
5	0.6525	0.9443	0.7639	0.38089
			0.8328	0.39986
6	0.6525	0.8328	0.7214	0.37665
			0.7639	0.38089
7	0.6525	0.7639	0.6950	0.37696
			0.7214	0.37665
8	0.6950	0.7639	0.7214	0.37665
			0.7376	0.37759
9	0.6950	0.7376	0.7113	0.37651
			0.7214	0.37665
10	0.6950	0.7214	0.7051	0.37658
			0.7113	0.37651
11	0.7051	0.7214	0.7113	0.37651
			0.7151	0.37652
12	0.7051	0.7151	0.7089	0.37652
			0.7113	0.37651
13	0.7089	0.7151	0.7113	0.37651
			0.7128	0.37651
14	0.7089	0.7128	0.7104	0.37651
			0.7113	0.37651
15	0.7104	0.7128	0.7113	0.37651
			0.7119	0.37651
16	0.7113	0.7128	0.7119	0.37651
			0.7122	0.37651
17	0.7113	0.7122	0.7117	0.37651
			0.7119	0.37651
18	0.7113	0.7119	0.7115	0.37651
			0.7117	0.37651
19	0.7115	0.7119	0.7117	0.37651
			0.7117	0.37651

OPTIMUM RESULTS: PULSE DURATION = 0.7117
PERFORMANCE 1 = 0.37651

OPTIMISATION WITH SLAM

PULSE AMPLITUDE = 0,1000

ITERATION NO	DURATION-INTERVAL LOWER	DURATION-INTERVAL UPPER	DURATION (DT)	PERFORMANCE 2 F(DT)
1	0,0000	2,0000	0,7639	1,76962
			1,2361	1,67338
2	0,7639	2,0000	1,2361	1,67338
			1,5279	1,66884
3	1,2361	2,0000	1,5279	1,66884
			1,7082	1,68912
4	1,2361	1,7082	1,4164	1,66517
			1,5279	1,66884
5	1,2361	1,5279	1,3475	1,66627
			1,4164	1,66517
6	1,3475	1,5279	1,4164	1,66517
			1,4590	1,66577
7	1,3475	1,4590	1,3901	1,66529
			1,4164	1,66517
8	1,3901	1,4590	1,4164	1,66517
			1,4327	1,66529
9	1,3901	1,4327	1,4064	1,66517
			1,4164	1,66517
10	1,4064	1,4327	1,4164	1,66517
			1,4226	1,66520
11	1,4064	1,4226	1,4126	1,66517
			1,4164	1,66517
12	1,4064	1,4164	1,4102	1,66517
			1,4126	1,66517
13	1,4102	1,4164	1,4126	1,66517
			1,4140	1,66517
14	1,4102	1,4140	1,4117	1,66517
			1,4126	1,66517
15	1,4102	1,4126	1,4111	1,66517
			1,4117	1,66517
16	1,4111	1,4126	1,4117	1,66517
			1,4120	1,66517
17	1,4111	1,4120	1,4114	1,66517
			1,4117	1,66517
18	1,4114	1,4120	1,4117	1,66517
			1,4118	1,66517
19	1,4114	1,4118	1,4116	1,66517
			1,4117	1,66517

OPTIMUM RESULTS: PULSE DURATION = 1,4116
PERFORMANCE 2 = 1,66517

OPTIMISATION WITH SLAM

PULSE AMPLITUDE = 0,2000

ITERATION NO	DURATION-INTERVAL LOWER	UPPER	DURATION (DT)	PERFORMANCE 2 F(DT)
1	0,0000	2,0000	0,7639	1,50393
			1,2361	1,35586
2	0,7639	2,0000	1,2361	1,35586
			1,5279	1,38212
3	0,7639	1,5279	1,0557	1,38647
			1,2361	1,35586
4	1,0557	1,5279	1,2361	1,35586
			1,3475	1,35459
5	1,2361	1,5279	1,3475	1,35459
			1,4164	1,36076
6	1,2361	1,4164	1,3050	1,35345
			1,3475	1,35459
7	1,2361	1,3475	1,2786	1,35375
			1,3050	1,35345
8	1,2786	1,3475	1,3050	1,35345
			1,3212	1,35364
9	1,2786	1,3212	1,2949	1,35347
			1,3050	1,35345
10	1,2949	1,3212	1,3050	1,35345
			1,3112	1,35349
11	1,2949	1,3112	1,3011	1,35344
			1,3050	1,35345
12	1,2949	1,3050	1,2987	1,35345
			1,3011	1,35344
13	1,2987	1,3050	1,3011	1,35344
			1,3026	1,35344
14	1,3011	1,3050	1,3026	1,35344
			1,3035	1,35344
15	1,3011	1,3035	1,3020	1,35344
			1,3026	1,35344
16	1,3011	1,3026	1,3017	1,35344
			1,3020	1,35344
17	1,3017	1,3026	1,3020	1,35344
			1,3022	1,35344
18	1,3020	1,3026	1,3022	1,35344
			1,3024	1,35344
19	1,3020	1,3024	1,3022	1,35344
			1,3022	1,35344

OPTIMUM RESULTS: PULSE DURATION = 1,3022
PERFORMANCE 2 = 1,35344

OPTIMISATION WITH SLAM

PULSE AMPLITUDE = 0,3000

ITERATION NO	DURATION-INTERVAL LOWER	UPPER	DURATION (DT)	PERFORMANCE 2 F(DT)
1	0,0000	2,0000	0,7639	1,26541
			1,2361	1,10993
2	0,7639	2,0000	1,2361	1,10993
			1,5279	1,20233
3	0,7639	1,5279	1,0557	1,12712
			1,2361	1,10993
4	1,0557	1,5279	1,2361	1,10993
			1,3475	1,12746
5	1,0557	1,3475	1,1672	1,10995
			1,2361	1,10993
6	1,1672	1,3475	1,2361	1,10993
			1,2786	1,11405
7	1,1672	1,2786	1,2098	1,10897
			1,2361	1,10993
8	1,1672	1,2361	1,1935	1,10898
			1,2098	1,10897
9	1,1935	1,2361	1,2098	1,10897
			1,2198	1,10920
10	1,1935	1,2198	1,2035	1,10892
			1,2098	1,10897
11	1,1935	1,2098	1,1997	1,10892
			1,2035	1,10892
12	1,1997	1,2098	1,2035	1,10892
			1,2059	1,10893
13	1,1997	1,2059	1,2021	1,10892
			1,2035	1,10892
14	1,1997	1,2035	1,2012	1,10892
			1,2021	1,10892
15	1,2012	1,2035	1,2021	1,10892
			1,2026	1,10892
16	1,2012	1,2026	1,2017	1,10892
			1,2021	1,10892
17	1,2012	1,2021	1,2015	1,10892
			1,2017	1,10892
18	1,2015	1,2021	1,2017	1,10892
			1,2019	1,10892
19	1,2017	1,2021	1,2019	1,10892
			1,2019	1,10892

OPTIMUM RESULTS: PULSE DURATION = 1,2019
PERFORMANCE 2 = 1,10892

OPTIMISATION WITH SLAM

PULSE AMPLITUDE = 0,4000

ITERATION NO	DURATION-INTERVAL LOWER	UPPER	DURATION (DT)	PERFORMANCE 2 F(DT)
1	0,0000	2,0000	0,7639	1,05408
			1,2361	0,93559
2	0,7639	2,0000	1,2361	0,93559
			1,5279	1,12949
3	0,7639	1,5279	1,0557	0,92020
			1,2361	0,93559
4	0,7639	1,2361	0,9443	0,94918
			1,0557	0,92020
5	0,9443	1,2361	1,0557	0,92020
			1,1246	0,91682
6	1,0557	1,2361	1,1246	0,91682
			1,1672	0,92043
7	1,0557	1,1672	1,0983	0,91678
			1,1246	0,91682
8	1,0557	1,1246	1,0820	0,91758
			1,0983	0,91678
9	1,0820	1,1246	1,0983	0,91678
			1,1084	0,91660
10	1,0983	1,1246	1,1084	0,91660
			1,1146	0,91661
11	1,0983	1,1146	1,1045	0,91664
			1,1084	0,91660
12	1,1045	1,1146	1,1084	0,91660
			1,1107	0,91659
13	1,1084	1,1146	1,1107	0,91659
			1,1122	0,91660
14	1,1084	1,1122	1,1098	0,91660
			1,1107	0,91659
15	1,1098	1,1122	1,1107	0,91659
			1,1113	0,91659
16	1,1098	1,1113	1,1104	0,91659
			1,1107	0,91659
17	1,1104	1,1113	1,1107	0,91659
			1,1109	0,91659
18	1,1104	1,1109	1,1106	0,91659
			1,1107	0,91659
19	1,1106	1,1109	1,1107	0,91659
			1,1108	0,91659

OPTIMUM RESULTS: PULSE DURATION = 1,1107
PERFORMANCE 2 = 0,91659

OPTIMISATION WITH SLAM

PULSE AMPLITUDE = 0,5000

ITERATION NO	DURATION-INTERVAL LOWER	DURATION-INTERVAL UPPER	DURATION (DT)	PERFORMANCE 2 F(DT)
1	0,0000	2,0000	0,7639	0,86992
			1,2361	0,83285
2	0,7639	2,0000	1,2361	0,83285
			1,5279	1,16358
3	0,7639	1,5279	1,0557	0,76573
			1,2361	0,83285
4	0,7639	1,2361	0,9443	0,77558
			1,0557	0,76573
5	0,9443	1,2361	1,0557	0,76573
			1,1246	0,77908
6	0,9443	1,1246	1,0132	0,76496
			1,0557	0,76573
7	0,9443	1,0557	0,9868	0,76730
			1,0132	0,76496
8	0,9868	1,0557	1,0132	0,76496
			1,0294	0,76459
9	1,0132	1,0557	1,0294	0,76459
			1,0395	0,76477
10	1,0132	1,0395	1,0232	0,76464
			1,0294	0,76459
11	1,0232	1,0395	1,0294	0,76459
			1,0333	0,76462
12	1,0232	1,0333	1,0270	0,76459
			1,0294	0,76459
13	1,0270	1,0333	1,0294	0,76459
			1,0309	0,76460
14	1,0270	1,0309	1,0285	0,76459
			1,0294	0,76459
15	1,0270	1,0294	1,0280	0,76459
			1,0285	0,76459
16	1,0280	1,0294	1,0285	0,76459
			1,0289	0,76459
17	1,0280	1,0289	1,0283	0,76459
			1,0285	0,76459
18	1,0283	1,0289	1,0285	0,76459
			1,0286	0,76459
19	1,0285	1,0289	1,0286	0,76459
			1,0287	0,76459

OPTIMUM RESULTS: PULSE DURATION = 1,0286
PERFORMANCE 2 = 0,76459

OPTIMISATION WITH SLAM

PULSE AMPLITUDE = 0,6000

ITERATION NO	DURATION-INTERVAL LOWER	DURATION-INTERVAL UPPER	DURATION (DT)	PERFORMANCE 2 F(DT)
1	0,0000	2,0000	0,7639	0,71295
			1,2361	0,80169
2	0,0000	1,2361	0,4721	1,06132
			0,7639	0,71295
3	0,4721	1,2361	0,7639	0,71295
			0,9443	0,64389
4	0,7639	1,2361	0,9443	0,64389
			1,0557	0,66370
5	0,7639	1,0557	0,8754	0,65588
			0,9443	0,64389
6	0,8754	1,0557	0,9443	0,64389
			0,9868	0,64566
7	0,8754	0,9868	0,9180	0,64632
			0,9443	0,64389
8	0,9180	0,9868	0,9443	0,64389
			0,9605	0,64373
9	0,9443	0,9868	0,9605	0,64373
			0,9706	0,64414
10	0,9443	0,9706	0,9543	0,64367
			0,9605	0,64373
11	0,9443	0,9605	0,9505	0,64370
			0,9543	0,64367
12	0,9505	0,9605	0,9543	0,64367
			0,9567	0,64367
13	0,9505	0,9567	0,9529	0,64367
			0,9543	0,64367
14	0,9529	0,9567	0,9543	0,64367
			0,9552	0,64367
15	0,9543	0,9567	0,9552	0,64367
			0,9558	0,64367
16	0,9543	0,9558	0,9549	0,64367
			0,9552	0,64367
17	0,9543	0,9552	0,9547	0,64367
			0,9549	0,64367
18	0,9547	0,9552	0,9549	0,64367
			0,9550	0,64367
19	0,9547	0,9550	0,9548	0,64367
			0,9549	0,64367

OPTIMUM RESULTS: PULSE DURATION = 0,9548
PERFORMANCE 2 = 0,64367

OPTIMISATION WITH SLAM

PULSE AMPLITUDE = 0,7000

ITERATION NO	DURATION-INTERVAL LOWER	UPPER	DURATION (DT)	PERFORMANCE 2 F(DT)
1	0,0000	2,0000	0,7639	0,58315
			1,2361	0,84213
2	0,0000	1,2361	0,4721	0,92963
			0,7639	0,58315
3	0,4721	1,2361	0,7639	0,58315
			0,9443	0,55408
4	0,7639	1,2361	0,9443	0,55408
			1,0557	0,61411
5	0,7639	1,0557	0,8754	0,54717
			0,9443	0,55408
6	0,7639	0,9443	0,8328	0,55414
			0,8754	0,54717
7	0,8328	0,9443	0,8754	0,54717
			0,9017	0,54713
8	0,8754	0,9443	0,9017	0,54713
			0,9180	0,54876
9	0,8754	0,9180	0,8916	0,54676
			0,9017	0,54713
10	0,8754	0,9017	0,8854	0,54677
			0,8916	0,54676
11	0,8854	0,9017	0,8916	0,54676
			0,8955	0,54684
12	0,8854	0,8955	0,8893	0,54674
			0,8916	0,54676
13	0,8854	0,8916	0,8878	0,54674
			0,8893	0,54674
14	0,8878	0,8916	0,8893	0,54674
			0,8902	0,54674
15	0,8878	0,8902	0,8887	0,54674
			0,8893	0,54674
16	0,8878	0,8893	0,8884	0,54674
			0,8887	0,54674
17	0,8884	0,8893	0,8887	0,54674
			0,8889	0,54674
18	0,8887	0,8893	0,8889	0,54674
			0,8891	0,54674
19	0,8887	0,8891	0,8888	0,54674
			0,8889	0,54674

OPTIMUM RESULTS: PULSE DURATION = 0,8888
PERFORMANCE 2 = 0,54674

OPTIMISATION WITH SLAM

PULSE AMPLITUDE = 0,8000

ITERATION NO	DURATION-INTERVAL LOWER	UPPER	DURATION (DT)	PERFORMANCE 2 F(DT)
1	0,0000	2,0000	0,7639	0,48054
			1,2361	0,95416
2	0,0000	1,2361	0,4721	0,80799
			0,7639	0,48054
3	0,4721	1,2361	0,7639	0,48054
			0,9443	0,50618
4	0,4721	0,9443	0,6525	0,55493
			0,7639	0,48054
5	0,6525	0,9443	0,7639	0,48054
			0,8328	0,46844
6	0,7639	0,9443	0,8328	0,46844
			0,8754	0,47437
7	0,7639	0,8754	0,8065	0,46992
			0,8328	0,46844
8	0,8065	0,8754	0,8328	0,46844
			0,8491	0,46948
9	0,8065	0,8491	0,8228	0,46854
			0,8328	0,46844
10	0,8228	0,8491	0,8328	0,46844
			0,8390	0,46866
11	0,8228	0,8390	0,8290	0,46841
			0,8328	0,46844
12	0,8228	0,8328	0,8266	0,46843
			0,8290	0,46841
13	0,8266	0,8328	0,8290	0,46841
			0,8304	0,46841
14	0,8266	0,8304	0,8281	0,46841
			0,8290	0,46841
15	0,8281	0,8304	0,8290	0,46841
			0,8295	0,46841
16	0,8290	0,8304	0,8295	0,46841
			0,8299	0,46841
17	0,8290	0,8299	0,8293	0,46841
			0,8295	0,46841
18	0,8293	0,8299	0,8295	0,46841
			0,8297	0,46841
19	0,8295	0,8299	0,8297	0,46841
			0,8298	0,46841

OPTIMUM RESULTS: PULSE DURATION = 0,8297
PERFORMANCE 2 = 0,46841

OPTIMISATION WITH SLAM

PULSE AMPLITUDE = 0,9000

ITERATION NO	DURATION-INTERVAL LOWER	UPPER	DURATION (DT)	PERFORMANCE 2 F(DT)
1	0,0000	2,0000	0,7639	0,40510
			1,2361	1,13778
2	0,0000	1,2361	0,4721	0,69640
			0,7639	0,40510
3	0,4721	1,2361	0,7639	0,40510
			0,9443	0,50016
4	0,4721	0,9443	0,6525	0,45485
			0,7639	0,40510
5	0,6525	0,9443	0,7639	0,40510
			0,8328	0,41517
6	0,6525	0,8328	0,7214	0,41465
			0,7639	0,40510
7	0,7214	0,8328	0,7639	0,40510
			0,7902	0,40519
8	0,7214	0,7902	0,7477	0,40734
			0,7639	0,40510
9	0,7477	0,7902	0,7639	0,40510
			0,7740	0,40459
10	0,7639	0,7902	0,7740	0,40459
			0,7802	0,40461
11	0,7639	0,7802	0,7701	0,40471
			0,7740	0,40459
12	0,7701	0,7802	0,7740	0,40459
			0,7764	0,40457
13	0,7740	0,7802	0,7764	0,40457
			0,7778	0,40457
14	0,7740	0,7778	0,7754	0,40457
			0,7764	0,40457
15	0,7754	0,7778	0,7764	0,40457
			0,7769	0,40457
16	0,7754	0,7769	0,7760	0,40457
			0,7764	0,40457
17	0,7760	0,7769	0,7764	0,40457
			0,7766	0,40457
18	0,7764	0,7769	0,7766	0,40457
			0,7767	0,40457
19	0,7764	0,7767	0,7765	0,40457
			0,7766	0,40457

OPTIMUM RESULTS: PULSE DURATION = 0,7765
PERFORMANCE 2 = 0,40457

OPTIMISATION WITH SLAM

PULSE AMPLITUDE = 1,0000

ITERATION NO	DURATION-INTERVAL LOWER	UPPER	DURATION (DT)	PERFORMANCE 2 F(DT)
1	0,0000	2,0000	0,7639	0,35685
			1,2361	1,39299
2	0,0000	1,2361	0,4721	0,59486
			0,7639	0,35685
3	0,4721	1,2361	0,7639	0,35685
			0,9443	0,53605
4	0,4721	0,9443	0,6525	0,37441
			0,7639	0,35685
5	0,6525	0,9443	0,7639	0,35685
			0,8328	0,39435
6	0,6525	0,8328	0,7214	0,35233
			0,7639	0,35685
7	0,6525	0,7639	0,6950	0,35652
			0,7214	0,35233
8	0,6950	0,7639	0,7214	0,35233
			0,7376	0,35239
9	0,6950	0,7376	0,7113	0,35330
			0,7214	0,35233
10	0,7113	0,7376	0,7214	0,35233
			0,7276	0,35211
11	0,7214	0,7376	0,7276	0,35211
			0,7314	0,35213
12	0,7214	0,7314	0,7252	0,35216
			0,7276	0,35211
13	0,7252	0,7314	0,7276	0,35211
			0,7290	0,35210
14	0,7276	0,7314	0,7290	0,35210
			0,7299	0,35211
15	0,7276	0,7299	0,7285	0,35210
			0,7290	0,35210
16	0,7285	0,7299	0,7290	0,35210
			0,7294	0,35210
17	0,7285	0,7294	0,7288	0,35210
			0,7290	0,35210
18	0,7288	0,7294	0,7290	0,35210
			0,7292	0,35210
19	0,7288	0,7292	0,7290	0,35210
			0,7290	0,35210

OPTIMUM RESULTS: PULSE DURATION = 0,7290
PERFORMANCE 2 = 0,35210

APPENDIX A7 The ISIS Program and Results

LINE TEXT

```
DIMENSION D(3),P(3)
Y=1
CINT=0.1;NCOM=200
1 I=0
READ K,AMP
INTAMP=AMP
2 READ DT
IF(DT<0)GOTO 11
3 I=I+1
AMP=INTAMP
RESET
SIM
D(I)=DT
P(I)=PF
IF(I=1)GOTO 4
IF(I=2)GOTO 6
IF(P(2)<P(3))GOTO 9
P(1)=P(2)
P(2)=P(3)
D(1)=D(2)
D(2)=D(3)
I=I-1
4 PRINT DT,PF
DT=DT+0.1
GOTO 3
6 IF(P(2)>P(1))GOTO 8
GOTO 4
8 START =0
PRINT START
I=0
GOTO 2
9 PRINT DT,PF
X1=D(1);X2=D(2);X3=D(3)
Y1=P(1);Y2=P(2);Y3=P(3)
A=((Y1-Y2)/(X1-X2))-((Y1-Y3)/(X1-X3))
A=A/(X2-X3)
B=((Y1-Y2)/(X1-X2))-((X1+X2)*A)
C=Y1-(B*X1)-(A*X1*X1)
DTOPT=-B/(2*A)
PFOPT=A*DTOPT*DTOPT+B*DTOPT+C
PRINT DTOPT,PFOPT
11 READ J
IF(J>0)GOTO 1
STOP
#DYNAMIC
Y'=X-AMP
X'=-0.5*X-Y
IF(K=2)GOTO 20
PF'=Y*Y
GOTO 21
20 PF'=T*Y*Y
21 IF(T>DT)AMP=0
```

Program A7 The ISIS Program

```
K      =?   1.0000
AMP    =?   1.0000
DT     =?  0.60000
DT     =   0.60000        PF     =   0.39525
DT     =   0.70000        PF     =   0.37657
DT     =   0.80000        PF     =   0.39094
DTOPT  =   0.70652        PFOPT  =   0.37650
J      =?   1.0000
K      =?   1.0000
AMP    =?  0.90000
DT     =?  0.60000
DT     =   0.60000        PF     =   0.44431
DT     =   0.70000        PF     =   0.41547
DT     =   0.80000        PF     =   0.41509
DT     =   0.90000        PF     =   0.44198
DTOPT  =   0.75140        PFOPT  =   0.41186
J      =?   1.0000
K      =?   1.0000
AMP    =?  0.80000
DT     =?  0.70000
DT     =   0.70000        PF     =   0.46513
DT     =   0.80000        PF     =   0.45295
DT     =   0.90000        PF     =   0.46402
DTOPT  =   0.80238        PFOPT  =   0.45294
J      =?   1.0000
K      =?   1.0000
AMP    =?  0.70000
DT     =?  0.70000
DT     =   0.70000        PF     =   0.52556
DT     =   0.80000        PF     =   0.50454
DT     =   0.90000        PF     =   0.50300
DT     =    1.0000        PF     =   0.52013
DTOPT  =   0.85825        PFOPT  =   0.50138
J      =?   1.0000
K      =?   1.0000
AMP    =?  0.60000
DT     =?  0.80000
DT     =   0.80000        PF     =   0.56986
DT     =   0.90000        PF     =   0.55891
DT     =    1.0000        PF     =   0.56334
DTOPT  =   0.92121        PFOPT  =   0.55857
```

```
J     =?   1.0000
K     =?   1.0000
AMP   =?  0.50000
DT    =?  0.80000
DT    =   0.80000     PF    =  0.64890
DT    =   0.90000     PF    =  0.63176
DT    =    1.0000     PF    =  0.62690
DT    =    1.1000     PF    =  0.63383
DTOPT =   0.99120     PFOPT =  0.62685
J     =?   1.0000
K     =?   1.0000
AMP   =?  0.40000
DT    =?  0.90000
DT    =   0.90000     PF    =  0.72153
DT    =    1.0000     PF    =  0.71081
DT    =    1.1000     PF    =  0.70918
DT    =    1.2000     PF    =  0.71625
DTOPT =    1.0687     PFOPT =  0.70875
J     =?   1.0000
K     =?   1.0000
AMP   =?  0.30000
DT    =?   1.0000
DT    =    1.0000     PF    =  0.81507
DT    =    1.1000     PF    =  0.80846
DT    =    1.2000     PF    =  0.80819
DT    =    1.3000     PF    =  0.81395
DTOPT =    1.1546     PFOPT =  0.80756
J     =?   1.0000
K     =?   1.0000
AMP   =?  0.20000
DT    =?   1.1000
DT    =    1.1000     PF    =  0.93169
DT    =    1.2000     PF    =  0.92778
DT    =    1.3000     PF    =  0.92781
DTOPT =    1.2493     PFOPT =  0.92730
J     =?   1.0000
K     =?   1.0000
AMP   =?  0.10000
DT    =?   1.2000
DT    =    1.2000     PF    =   1.0750
DT    =    1.3000     PF    =   1.0731
DT    =    1.4000     PF    =   1.0731
DT    =    1.5000     PF    =   1.0748
DTOPT =    1.3525     PFOPT =   1.0729
```

```
J      =?   1.0000
K      =?   2.0000
AMP    =?   1.0000
DT     =?  0.60000
DT     =  0.60000        PF     =  0.41002
DT     =  0.70000        PF     =  0.35406
DT     =  0.80000        PF     =  0.37562
DTOPT  =  0.72219        PFOPT  =  0.35215
J      =?   1.0000
K      =?   2.0000
AMP    =?  0.90000
DT     =?  0.60000
DT     =  0.60000        PF     =  0.49962
DT     =  0.70000        PF     =  0.42092
DT     =  0.80000        PF     =  0.40750
DT     =  0.90000        PF     =  0.46253
DTOPT  =  0.76960        PFOPT  =  0.40434
J      =?   1.0000
K      =?   2.0000
AMP    =?  0.80000
DT     =?  0.70000
DT     =  0.70000        PF     =  0.51086
DT     =  0.80000        PF     =  0.46977
DT     =  0.90000        PF     =  0.48566
DTOPT  =  0.82211        PFOPT  =  0.46838
J      =?   1.0000
K      =?   2.0000
AMP    =?  0.70000
DT     =?  0.80000
DT     =  0.80000        PF     =  0.56241
DT     =  0.90000        PF     =  0.54743
DT     =   1.0000        PF     =  0.58102
DTOPT  =  0.88085        PFOPT  =  0.54654
J      =?   1.0000
K      =?   2.0000
AMP    =?  0.60000
DT     =?  0.80000
DT     =  0.80000        PF     =  0.68543
DT     =  0.90000        PF     =  0.64782
DT     =   1.0000        PF     =  0.64910
DTOPT  =  0.94671        PFOPT  =  0.64358
```

```
J      =?   1.0000
K      =?   2.0000
AMP    =?  0.50000
DT     =?  0.90000
DT     =   0.90000      PF     =  0.78685
DT     =   1.0000       PF     =  0.76500
DT     =   1.1000       PF     =  0.77458
DTOPT  =   1.0195       PFOPT  =  0.76440
J      =?   1.0000
K      =?   2.0000
AMP    =?  0.40000
DT     =?   1.0000
DT     =   1.0000       PF     =  0.92869
DT     =   1.1000       PF     =  0.91630
DT     =   1.2000       PF     =  0.92823
DTOPT  =   1.1009       PFOPT  =  0.91630
J      =?   1.0000
K      =?   2.0000
AMP    =?  0.30000
DT     =?   1.1000
DT     =   1.1000       PF     =   1.1159
DT     =   1.2000       PF     =   1.1086
DT     =   1.3000       PF     =   1.1189
DTOPT  =   1.1913       PFOPT  =   1.1085
J      =?   1.0000
K      =?   2.0000
AMP    =?  0.20000
DT     =?   1.1000
DT     =   1.1000       PF     =   1.3732
DT     =   1.2000       PF     =   1.3576
DT     =   1.3000       PF     =   1.3530
DT     =   1.4000       PF     =   1.3597
DTOPT  =   1.2908       PFOPT  =   1.3529
J      =?   1.0000
K      =?   2.0000
AMP    =?  0.10000
DT     =?   1.2000
DT     =   1.2000       PF     =   1.6752
DT     =   1.3000       PF     =   1.6672
DT     =   1.4000       PF     =   1.6645
DT     =   1.5000       PF     =   1.6672
DTOPT  =   1.3993       PFOPT  =   1.6645
J      =?  -1.0000
```