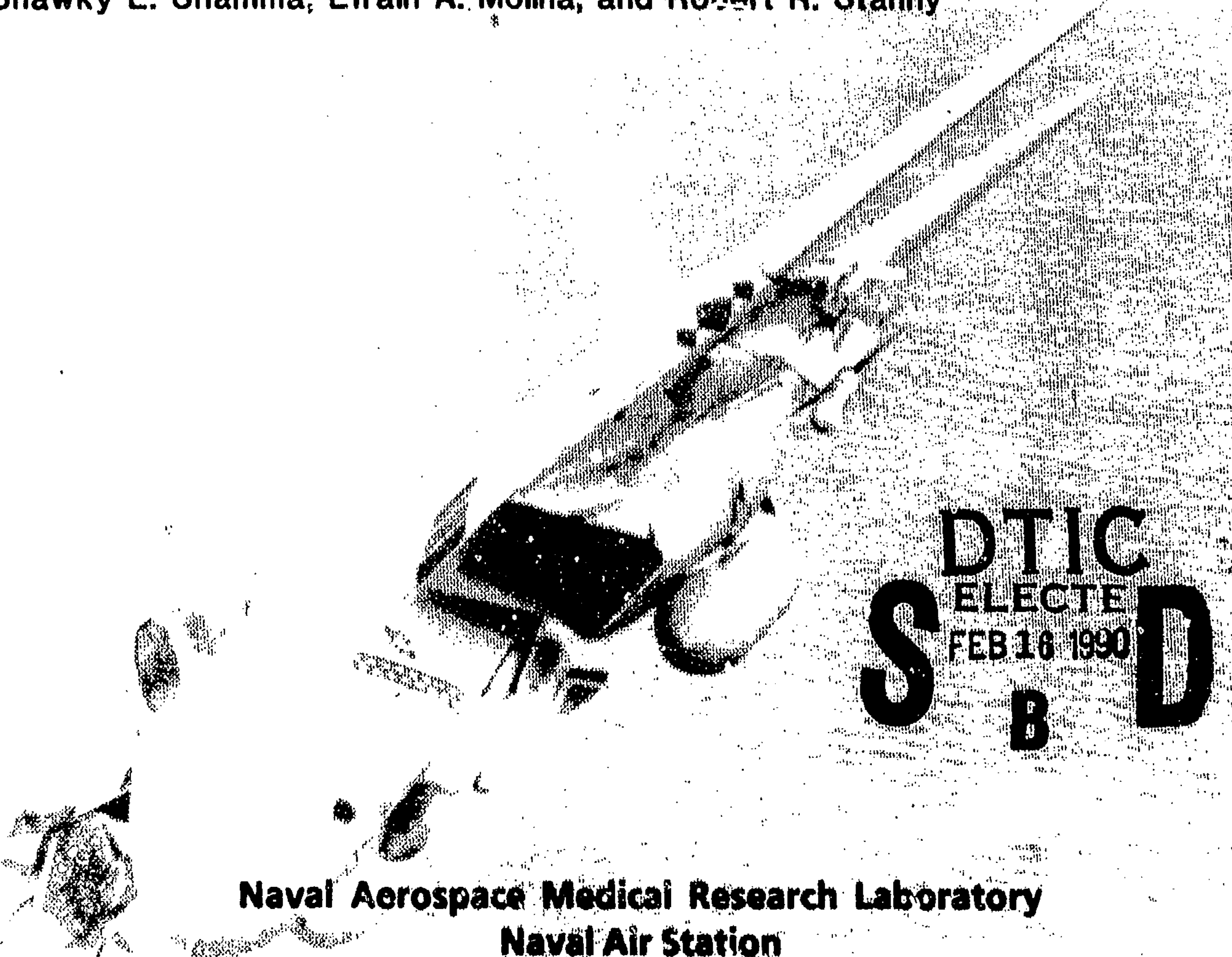
DTIC FILE COPY

(2)

MICRO SAINT PROGRAMS FOR NUMERICAL METHODS OF INTEGRATION AND DIFFERENTIAL EQUATIONS

Shawky E. Shamma, Efrain A. Molina, and Robert R. Stanny



Naval Aerospace Medical Research Laboratory
Naval Air Station
Pensacola, Florida 32508-5700

Approved for public release; distribution unlimited.

Reviewed and approved 9 Sept 1989

J. A. BRADY, CAPT, MSC USN
Commanding Officer

This research was sponsored by the U.S. Army Medical Research and Development Command under work unit 63764A 3M4637648995.AB-088

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited.
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) NAMRL Monograph 39	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Naval Aerospace Medical Research Laboratory 6b. OFFICE SYMBOL (If applicable) 24	7a. NAME OF MONITORING ORGANIZATION Naval Medical Research and Development Command
6c. ADDRESS (City, State, and ZIP Code) Naval Air Station Pensacola, FL 32508-5700	7b. ADDRESS (City, State, and ZIP Code) National Naval Medical Center, Bldg. 1 Bethesda, MD 20814-5044
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code) National Naval Medical Center, Bldg. 1 Bethesda, MD 20814-5044	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. 63764A PROJECT NO. 3M4637 648995 TASK NO. AB-088-7 WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
(U) Micro SAINT Programs for Numerical Methods of Integration and Differential Equations.

12. PERSONAL AUTHOR(S)
Shawky E. Shamma*, Efrain A. Molina, and Robert R. Stanny

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Interim	FROM Jan 89 TO Mar 89	89-09-09	48

16. SUPPLEMENTARY NOTATION
*Institute for Statistical and Mathematical Modeling, University of West Florida, Pensacola, FL 32504

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Micro SAINT; Integration; Differential equations; Numerical

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

We developed Micro SAINT computational networks for numerical integration and solving initial value problems for linear and nonlinear first- and second-order ordinary differential equations, as well as for systems of differential equations. These Micro SAINT computer programs are written with a user friendly approach where the user will be required to supply the input information and the functional form(s) of the function(s) in the "function library" section of Micro SAINT without any changes in the main programs.

These computational modules could be used as subnetworks in modeling psychophysiological and biomedical problems of interest in naval aerospace medical research. For example, Micro SAINT developed models can be used by staff medical officers to predict psychophysiological performance of naval aircrew personnel under sustained operational work schedules.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL J.A. BRADY, CAPT, MSC USN, Commanding Officer	22b. TELEPHONE (Include Area Code) 904/452-4479 22c. OFFICE SYMBOL 00

CONTENTS

This page intentionally left blank.

ABSTRACT

We developed Micro SAINT computational networks for numerical integration and solving initial value problems for linear and nonlinear first- and second-order ordinary differential equations as well as for systems of differential equations. These Micro SAINT computer programs are written with a user friendly approach where the user will be required to supply the input information and the functional form(s) of the function(s) in the "function library" section of Micro SAINT without any changes in the main programs.

These computational modules could be used as subnetworks in modeling psychophysiological and biomedical problems of interest in naval aerospace medical research. For example, Micro SAINT developed models can be used by staff medical officers to predict psychophysiological performance of naval aircrew personnel under sustained operational work schedules.

Acknowledgment

This work was sponsored by the Joint Working Group on Drug Dependent Degradation in Military Performance under U.S. Army Medical Research Development Command work unit 63764A 3M34637648995.AB-088 and conducted at the Naval Aerospace Medical Research Laboratory, Pensacola, Florida. The authors thank Commander R. P. Olafson, MC, USNR, for his support and encouragements for this work, and Mrs. Nell Davis and Kathleen Mayer for their special efforts in the preparation of this manuscript.

This page intentionally left blank.

1. INTRODUCTION

In a 2-day workshop on Micro SAINT analysis given by S. E. Shamma on November 10 and 17, 1988, for staff members of the Naval Aerospace Medical Research Laboratory, Pensacola, Florida, a question was raised about the potential of using Micro SAINT software [1] for numerical solutions of mathematical problems, especially differential equations. In this report, we answer this question affirmatively.

We developed Micro SAINT networks for numerical integration and solving initial-value problems for linear and nonlinear first- and second-order ordinary differential equations as well as for systems of differential equations. These computational modules are expected to be used as subnetworks in modeling problems of interest in naval aerospace medical research.

The following sections constitute a detailed theoretical summary and Micro SAINT programs for:

a. Numerical integration of a function f(x) using Trapezoidal Rule.

b. Numerical integration of a function f(x) using Simpson's Rule.

c. Composite Simpson's Rule for double integrals.

d. Composite Simpson's Rule for triple integrals.

e. Euler method for solving a first-order ordinary differential equation.

f. Modified Euler method for solving a first-order ordinary differential equation.

g. Runga-Kutta method of order four for solving a first-order ordinary differential equation.

h. Runga-Kutta method of order four for solving a first-order system of ordinary differential equations.

i. Runga-Kutta method of order four for solving second-order (linear or nonlinear) ordinary differential equations.

The Micro SAINT computer programs are written with a user friendly approach. The user will supply the input information in specified places in the program, and the output is stored in the snapshots output files. The input information shall be entered in the "function library" section of Micro SAINT. The initial and end conditions shall be entered in a function named "initl" (initial). It consists of:

intilx = initial value of the independent variable x,

endx = end value of the independent variable x,

numincvl = number of subintervals for integration or subdivisions in the case of differential equations; and initial y or initial u_1 and u_2 in the case of solving a system of differential equations.

The integration function or the functional form(s) of the differential equation(s) shall be entered in a straightforward way in the "function library" of Micro SAINT. We will illustrate these procedures by applying the programs on examples from reference [2]. The main computer programs are listed in appendix A. The user may view each main program as a "black box" since the input information is entered separately in the "function library" of Micro SAINT. Some results are presented graphically in appendix B.

2. SUMMARY OF NUMERICAL INTEGRATION METHODS AND INPUT INFORMATION TO THE "FUNCTION LIBRARY" OF MICRO SAINT

NUMERICAL INTEGRATION

We considered two methods, Trapezoidal and Simpson's Rules, for computing the integral $\int_a^b f(x)\,dx$ and a composite Simpson's Rule for double integrals.

1. Trapezoidal Rule

If $f \in C^2 [a,b]$ with $h = (b-a)/n$ and $x_j = a + jh$ for each $j = 0,1,2,\ldots,n$, the trapezoidal rule for n subintervals is:

$$\int_a^b f(x)\,dx = \frac{h}{2}\Big[\sum_{j=0}^{n-1} \{f(x_j) + f(x_{j+1})\} \Big] + \text{error},$$

$\text{Error} = O(h^2)$.

The user shall enter the input information in the "function library" of Micro SAINT as follows:

initix = a, endx = b, numintvl = n, and the expression for the function f. The following example illustrates the case where a = 0, b = 1, n = 10, and f(x) = 1 + x.

TA .. Function Library Input for Intgtrpz Program.

FUNCTION LIBRARY Model Name: intgtrpz

Name:	Expression:
f	1+x;
initl	numintvl=10;initlx=0;endx=1;

2. Simpson's Rule

If $f \in C^4[a,b]$, with $h = (b-a)/n$, where $n = 2m$, n must be an even integer, with $x_j = a + jh$ for each $j = 0,1,2,\ldots,2m$, the Simpson's Rule for n subintervals is:

$$\int_a^b f(x)\,dx = \frac{h}{3}\Big\{ \sum_{j=1}^{m} [f(x_{2j-2}) + 4f(x_{2j-1}) + f(x_{2j})]\Big\} + \text{Error},$$

$\text{Error} = O(h^4)$.

The user shall enter the input information in the "function library" of Micro SAINT as illustrated in the trapezoidal case.

3. Composite Simpson's Rule for Double Integrals

The program "dblinteg" approximates the double integral of a function f(x,y) with limits of integration from a to b for x and from c(x) to d(x) for y, using a composite Simpson's Rule. To evaluate the integral

$$\int_{x=a}^{x=b} \int_{y=c(x)}^{y=d(x)} f(x,y)\, dydx,$$

the user needs to supply the functions f(x,y), c(x), and d(x) as well as the parameters initlx = a, endx = b, and the number of divisions, numdivx, and numdivy, along the x and y axes; "numdivx" and "numdivy" must be even numbers. The approximate value of the integral is stored in the output snapshot.

As an illustration, consider the integral

$$\int_{x=0}^{x=1} \int_{y=x^2}^{y=x} (xy)\, dydx.$$

Here initlx = 0, endx = 1, $c(x) = x^2$, d(x) = x, and f(x,y) = xy. The user supplies this information as well as numdivx and numdivy in the "function library" of Micro SAINT as follows:

TABLE 2. Function Library Input for Dblinteg Program.

FUNCTION LIBRARY Model Name: dblinteg

Name:	Expression:
f	x*y;
initl	initlx=0;endx=1;numdivx=10;numdivy=10;
funcdofx	x;
funccofx	x*x;

The exact value of the integral is 1/24, and the computed answer is 0.041650.

4. Composite Simpson's Rule for Triple Integrals

The program "triplint" approximates the triple integral of a function $f(x,y,z)$ with limits from a to b for x, from $c(x)$ to $d(x)$ for y, and from $\alpha(x,y)$ to $\beta(x,y)$ for z. To evaluate the integral

$$\int_{x=a}^{x=b} \int_{y=c(x)}^{y=d(x)} \int_{z=\alpha(x,y)}^{z=\beta(x,y)} f(x,y,z)\, dzdydx,$$

the user needs to supply the functions $f(x,y,z)$, $c(x)$, $d(x)$, $\alpha(x,y)$, $\beta(x,y)$ as well as the parameters initlx = a, endx = b, and the number of divisions, numdivx, numdivy, and numdivz, along x, y, and z axes, respectively; "numdivx," "numdivy," and "numdivz" must be even numbers. The approximate value of the integral is stored in the output snapshot.

As an illustration, consider the integral

$$\int_{x=0}^{x=1} \int_{y=x^2}^{y=x} \int_{z=xy}^{z=2} (xyz)\, dzdydx.$$

Here initlx = 0, endx = 1, $c(x) = x^2$, $d(x) = x$, $\alpha(x,y) = xy$, $\beta(x,y) = 2$, and $f(x,y,z) = xyz$. The user supplies this information as well as numdivx, numdivy, and numdivz in the "function library" of Micro SAINT as follows:

TABLE 3. Function Library Input for Triplint Program.

FUNCTION LIBRARY Model Name: triplint

Name	Expression:
cx	x*x;
dx	x;
betaxy	2;
alphaxy	x*y;
fxyz	x*y*z;
initl	initlx=0;endx=1;numdivx=10;numdivy=10;numdivz=10;

The exact value of the integral is 0.078125, and the computed answer is 0.078578.

3. NUMERICAL SOLUTION OF DIFFERENTIAL EQUATIONS

a. First-order ordinary differential equation: We consider three methods, Euler, modified Euler, and Runga-Kutta of order four for solving the initial value problem.

$$\frac{dy}{dx} = f(x,y), \; y(x_o) = y_o.$$

1. Euler's method:

The difference equation associated with Euler method is:

$y_i = y_{i-1} + hf(x_{i-1},y_{i-1})$ + Error, for $i = 1,2,3,\ldots,n$, where n = number of intervals, $x_i = x_o + ih$, and Error $= 0(h^2)$.

The user shall enter the input information, initlx, endx, numdivx, initly, and the functional form of f(x,y) in the "function library" of Micro SAINT.

The following example illustrates the input information needed for solving

$$\frac{dy}{dx} = f(x,y) = -y + x + 1, \; y(0) = 1, \text{ using numdivx} = 10.$$

TABLE 4. Function Library Input for Diffel Program.

FUNCTION	LIBRARY	Model Name: diffel
Name:	Expression:	
f	1-y+x;	
initl	initlx=0;endx=1;numintvl=10;initly=1;	

2. Modified Euler's method:

The modified Euler's method is a predictor-corrector method. The difference equation associated with the method is:

$$y_i = y_{i-1} + \frac{h}{2}\,[f(x_{i-1},\, y_{i-1}) + f\{x_i,y_{i-1} + hf(x_{i-1},y_{i-1})\}] + \text{Error},$$

Error $= 0(h^2)$.

$i = 1,2,\ldots,n$, where n = number of intervals and $x_i = x_o + ih$.

The user shall enter the input information in the "function library" of Micro SAINT as shown in Euler's method.

3. Runga-Kutta method of order four:

Runga-Kutta method of order four is a high accuracy method, Errors = $O(h^4)$, but it requires more computation per step. The difference equations associated with the method are:

$$k_1 = hf(x_{i-1}, y_{i-1}),$$

$$k_2 = hf(x_{i-1} + h/2, y_{i-1} + k/2),$$

$$k_3 = hf(x_{i-1} + h/2, y_{i-1} + k/2),$$

$$k_4 = hf(x_i, y_{i-1} + k_3),$$

$$y_i = y_{i-1} + (k_1 + 2k_2 + 2k_3 + k_4)/6,$$

$$i = 1,2,\ldots,n.$$

The user shall enter the input information in the "function library" of Micro SAINT as shown in the example in Table 3.

b. First-order system of ordinary differential equations:

We consider two differential equations in two unknowns u_1 and u_2:

$$\frac{du_1}{dx} = f_1(x,u_1,u_2),$$

$$\frac{du_2}{dx} = f_2(x,u_1,u_2),\ u_1(x_o) = \alpha\ u_2(x_o) = \beta\ .$$

The difference equations associated with the extension of Runga-Kutta method to systems of differential equations are:

$$k_{1,i} = hf_i(x_j,u_1,u_2),\ i = 1,2,$$

$$k_{2,i} = hf_i(x_j + h/2,\ u_{1j} + 0.5k_{11},u_{2j} + 0.5k_{12}),\ i = 1,2,$$

$$k_{3,i} = hf_i(x_j + h/2,\ u_{1j} + 0.5k_{21},u_{2j} + 0.5k_{22}),\ i = 1,2,$$

$$k_{4,i} = hf_i(x_j + h,\ u_{1,j} + k_{31},u_{2j} + k_{32}),\ i = 1,2,$$

$$u_{1,j+1} = u_{1,j} + (k_{11} + 2k_{21} + 2k_{31} + k_{41})/6,$$

$$u_{2,j+1} = u_{2,j} + (k_{1,2} + 2k_{22} + 2k_{32} + k_{42})/6.$$

As an illustration of an application, we use an example [2], about the use of Kirchkoff's Law in circuit theory. Assuming that the switch in the circuit shown in Fig. 1 is closed at time t = 0,

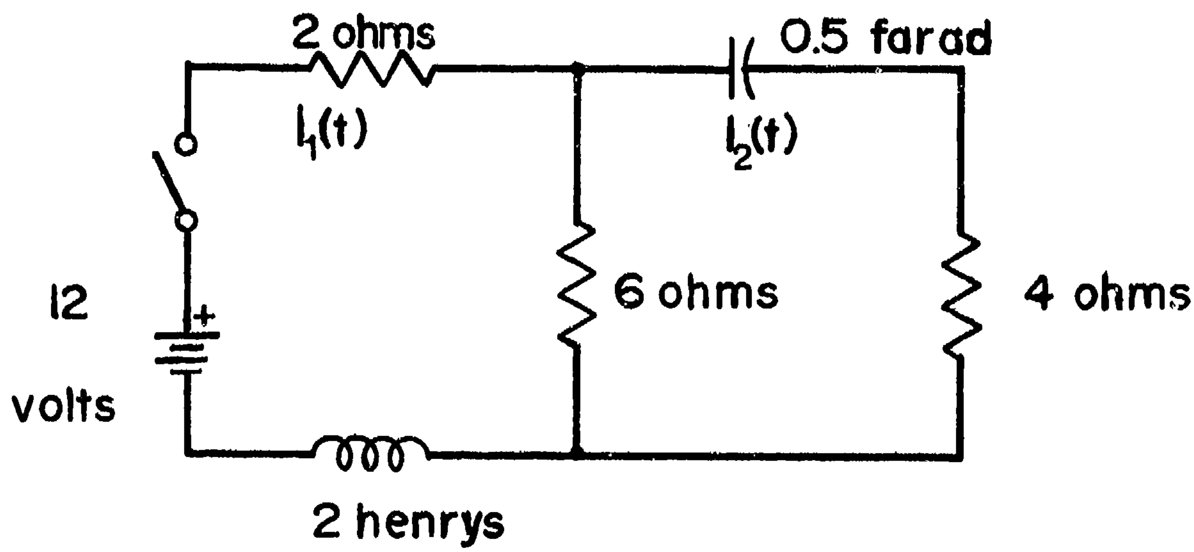


Figure 1. RLC electrical circuit.

the currents $I_1(t)$ and $I_2(t)$ in the left and right loops, respectively, are solutions of the following equations:

$$\frac{dI_1}{dt} = f_1(t,I_1,I_2) = -4I_1 + 3I_2 + 6, I_1(0) = 0,$$

$$\frac{dI_2}{dt} = f_2(t,I_1,I_2) = -2.4I_1 + 1.6I_2 + 3.6, I_2(0) = 0.$$

The user shall enter the input information using x, u_1, and u_2, respectively, for t, I_1, and I_2 in the "function library" of Micro SAINT as shown in Table 5.

TABLE 5. Function Library Input for Diffe4 Program.

FUNCTION LIBRARY Model Name: diffe4

Name:	Expression:
f1	(-4)*u1+3*u2+6;
f2	(-2.4)*u1+1.6*u2+3.6;
initl	initlx=0;endx=1;numintvl=10;initlu1=0;initlu2=0

c. Second-order ordinary differential equation:

To approximate the solution of a general second-order ordinary differential equation

$$\frac{d^2y}{dx^2} = f_2(x,y,\frac{dy}{dx}), \quad a \leq x \leq b,$$

$$y(a) = \alpha, \quad \frac{dy}{dx}(a) = \beta,$$

one needs to transform the equation into a system of first order using the transformation $u_1 = y$, $u_2 = \frac{dy}{dx}$, to get:

$$\frac{du_1}{dx} = u_2,$$

$$\frac{du_2}{dx} = f_2(x,u_1,u_2),$$

$$u_1(a) = \alpha, u_2(a) = \beta.$$

To illustrate the method, we consider the problem

$$\frac{d^2y}{dx^2} = f(x,y,\frac{dy}{dx}) = 2y'/x - 2y/x^2 + x\ln(x),$$

$$y(1) = 1, \qquad y'(1) = 0.$$

The transformed system is:

$$\frac{du_1}{dx} = u_2,$$

$$\frac{du_2}{dx} = 2u_2/x - 2u_1/x^2 + x^2\ln(x).$$

$$u_1(1) = 1, \; u_2(1) = 0.$$

The user needs to enter the initial conditions and the functional form of $f_2(x,u_1,u_2)$ as shown in Table 6, where y is replaced by u_1 and dy/dx is replaced by u_2.

TABLE 6. Function Library Input for Diffe5 Program.

FUNCTION LIBRARY Model Name: diffe5

Name:	Expression:
f2	2*u2/x+(-1)*2*u1/x^2+x*ln(x)
initl	initlx=1;endx=3;numintvl=40;initlu1=1;initlu2=0

4. CONCLUSIONS

We developed Micro SAINT programs for numerical methods of integration and differential equations. These computational modules could be used as subnetworks in modeling problems of interest in naval aerospace medical research. There are many other numerical methods for integration and for solving differential equations. Many of these techniques can be programmed in Micro SAINT[a] despite its minor shortcomings, such as the absence of defined values for the base of natural logarithm, and the capability of reading input data files.

The numerical integration methods presented here are adequate when the function being evaluated is relatively simple, that is, does not require many time-consuming manipulations. The differential equations methods, especially the Runga-Kutta, are adequate for problems where the function is easy to evaluate and the accuracy needed is small (about 10^{-4} for Runga-Kutta methods).

For further details and recommendations on which method to use for solving a given nonstiff initial-value problem (i.e., nonstiff differential equations with initial conditions), we recommend that the papers by Hull et al. [3], and Enright and Hull [4] be consulted. For details on methods for

[a]R.R. Stanny, Naval Aerospace Medical Research Laboratory, Pensacola, FL, personal communication, February 1989.

stiff differential equations, we recommend consulting Gear [5], Lambert [6], Shampine and Gear [7], or Enright et al. [8].

REFERENCES

1. Microsaint User's Guide, Micro Analysis and Design Inc., Boulder, CO, 1987.

2. Burden, R. L. and Faires, L. D., Numerical Analysis, Third Edition, Prindle, Weber, and Schmidt, Boston, MA, 1985.

3. Hull, T. E., Enright, W. H., Fellen, B. M., and Sedwick, A. E., "Comparing Numerical Methods for Ordinary Differential Equations." SIAM Journal on Numerical Analysis, Vol. 9, No. 4, pp. 603-637, 1972.

4. Enright, W. H. and Hull, T. E., "Test Results on Initial-value Methods for Nonstiff Ordinary Numerical Analysis." SIAM Journal on Numerical Analysis, Vol. 13, No. 6, pp. 944-961, 1976.

5. Gear, C. W., Numerical Initial-value Problems in Ordinary Differential Equations, Prentice-Hall, Englewood Cliffs, NJ, 1971.

6. Lambert, J. D., "The Initial Value Problem for Ordinary Differential Equation." In D. Jacobs (Ed.), The State of the Art in Numerical Analysis, Academic Press, New York, NY, 1977, pp. 451-501.

7. Shampine, L. F. and Gear, C. W., "A User's View of Solving Stiff Ordinary Differential Equations." SIAM Review, Vol. 21, No. 1, pp. 1-17, 1979.

8. Enright, W. H., Hull, T. E., and Lindberg, B., "Comparing Numerical Methods for Stiff Systems of O.D.E.'s." BIT Nordisk Tidskrift for Informationshehandling, Vol. 15, pp. 10-48, 1975.

APPENDIX A

Micro SAINT Diagrams and Computer Programs

APPENDIX A

In this appendix, we give actual Micro SAINT diagrams and computer programs for the following algorithms:

a. Numerical integration of a function f(x) using Trapezoidal Rule.

b. Numerical integration of a function f(x) using Simpson's Rule.

c. Composite Simpson's Rule for double integration of a function f(x,y).

d. Composite Simpson's Rule for triple integration of a function f(x,y,z),

e. Euler method for solving a first-order ordinary differential equation.

f. Modified Euler method for solving a first-order ordinary differential equation.

g. Runga-Kutta method of order four for solving a first-order ordinary differential equation.

h. Runga-Kutta method of order four for solving a first-order system of ordinary differential equations.

i. Runga-Kutta method of order four for solving second-order (linear or nonlinear) ordinary differential equations.

Model: intgtrpz Network: 0 intgtrpz

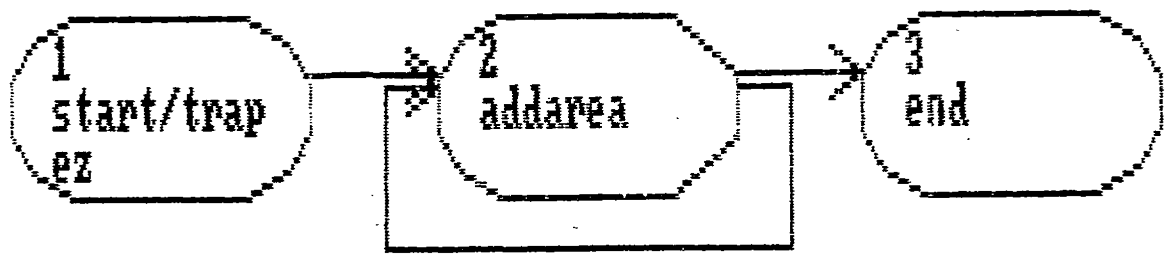


Figure 2. Micro SAINT diagram for Trapezoidal Rule for integrating a function f(x).

TASK NETWORK FOR INTGTRPZ PROGRAM

```
Network Number:  0
(1)  Name:  intgtrpz                                   (2)  Type:  Network
(3)  Upper Network:
(4)  Release Condition:  1;
(5)  First sub-job:  1  start/trapez
(6)  Sub-jobs (each can be task or network):
Number:          Name:               Type:
1                start/trapez        Task
2                addarea             Task
3                end                 Task

Task Number:  1
(1)  Name:  start/trapez                               (2)  Type:  Task
(3)  Upper Network:  0  intgtrpz
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:
initl;j=1;sum=0;delx=(endx-initlx)/numintvl;x=initlx;
f;prevf=f;
(10) Decision Type:  Single choice
     Following Task/Network:      Probability Of Taking
      Number:        Name:          This Path:
(11)  2              addare   (12)  1;
(13)                          (14)
(15)                          (16)
(17)                          (18)
(19)                          (20)
(21)                          (22)
(23)                          (24)

Task Number:  2
(1)  Name:  addarea                                    (2)  Type:  Task
(3)  Upper Network:  0  intgtrpz
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  x=initlx+j*delx;f;
sum=sum+(prevf+f);prevf=f;j=j+1;
(10) Decision Type:  Tactical
     Following Task/Network:      Tactical Expression:
      Number:        Name:
(11)  2              addare   (12)  j <= numintvl;
(13)  3              end      (14)  j > numintvl;
(15)                          (16)
(17)                          (18)
(19)                          (20)
(21)                          (22)
(23)                          (24)

Task Number:  3
(1)  Name:  end                                        (2)  Type:  Task
(3)  Upper Network:  0  intgtrpz
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  sum=sum*delx/2;
(10) Decision Type:  Last task
     Following Task/Network:      Probability Of Taking
      Number:        Name:          This Path:
(11)                          (12)
(13)                          (14)
(15)                          (16)
(17)                          (18)
(19)                          (20)
(21)                          (22)
(23)                          (24)
```

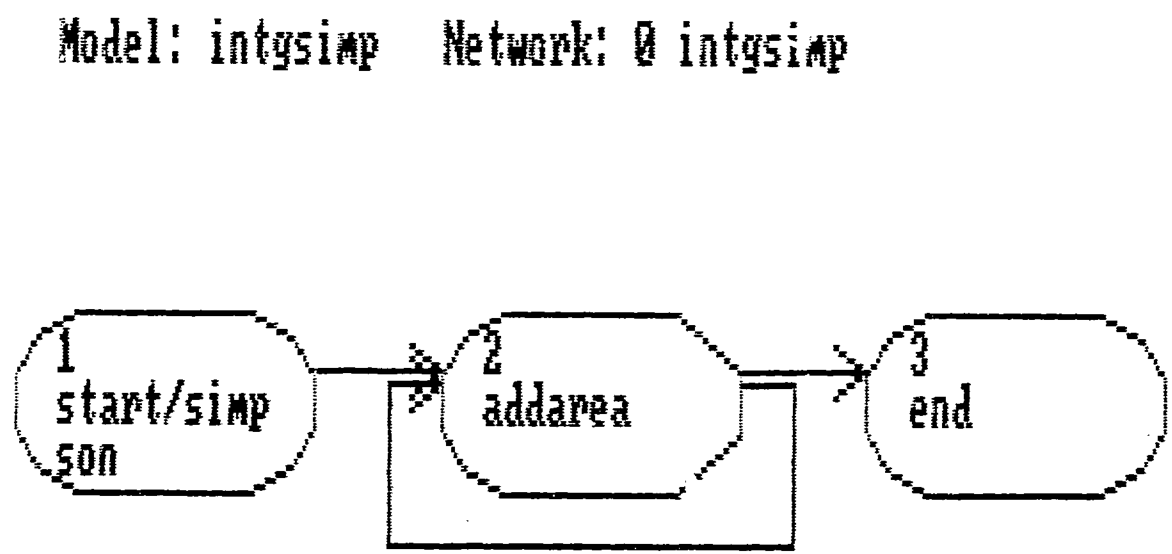


Figure 3. Micro SAINT diagram for Simpson's Rule for integrating a function f(x).

TASK NETWORK FOR INTGSIMP

```
Network Number:  0
(1)  Name:  intgsimp                                    (2)  Type:  Network
(3)  Upper Network:
(4)  Release Condition:  1;
(5)  First sub-job:  1  start/simpson
(6)  Sub-jobs (each can be task or network):
Number:         Name:               Type:
1               start/simpson       Task
2               addarea             Task
3               end                 Task

Task Number:  1
(1)  Name:  start/simpson                               (2)  Type:  Task
(3)  Upper Network:  0  intgsimp
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  initl;
delx=(endx-initlx)/numintvl;m=numintvl/2;
j=1;sum=0;xin=initlx;x=xin;f;tempf=f;
(10) Decision Type:  Single choice
     Following Task/Network:        Probability Of Taking
      Number:         Name:           This Path:
(11)  2               addare   (12)  1;
(13)                           (14)
(15)                           (16)
(17)                           (18)
(19)                           (20)
(21)                           (22)
(23)                           (24)

Task Number:  2
(1)  Name:  addarea                                     (2)  Type:  Task
(3)  Upper Network:  0  intgsimp
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  x=xin+delx;f;f1=f;x=x+delx;f;f2=f;
sum=sum+tempf+4*f1+f2;
xin=x;
xin=x;tempf=f2;j=j+1;
(10) Decision Type:  Tactical
     Following Task/Network:        Tactical Expression:
      Number:         Name:
(11)  2               addare   (12)  j < m+1;
(13)  3               end      (14)  j >= m+1;
(15)                           (16)
(17)                           (18)
(19)                           (20)
(21)                           (22)
(23)                           (24)

Task Number:  3
(1)  Name:  end                                         (2)  Type:  Task
(3)  Upper Network:  0  intgsimp
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  sum=sum*delx/3;
(10) Decision Type:  Last task
     Following Task/Network:        Probability Of Taking
      Number:         Name:           This Path:
(11)                           (12)
```

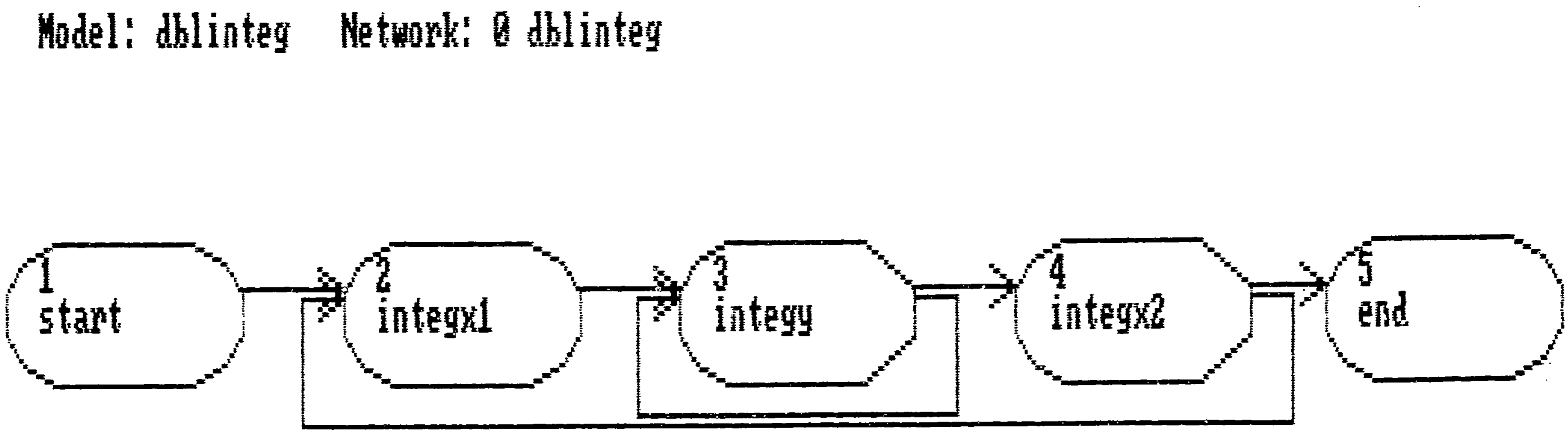


Figure 4. Micro SAINT diagram for a method of double integration of a function f(x,y).

TASK NETWORK FOR DBLINTEG PROGRAM

```
Network Number:  0
(1)  Name:  dblinteg                                  (2)  Type:  Network
(3)  Upper Network:
(4)  Release Condition:  1;
(5)  First sub-job:  1  start
(6)  Sub-jobs (each can be task or network):
Number:          Name:                 Type:
1                start                 Task
2                integx1               Task
3                integy                Task
4                integx2               Task
5                end                   Task

Task Number:  1
(1)  Name:  start                                     (2)  Type:  Task
(3)  Upper Network:  0  dblinteg
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  initl;delx=(endx-initlx)/numdivx;
sum1=0;sum2=0;sum3=0;i=0;
(10) Decision Type:  Single choice
     Following Task/Network:        Probability Of Taking
      Number:          Name:          This Path:
(11)  2                integx   (12)  1;
(13)                            (14)
(15)                            (16)
(17)                            (18)
(19)                            (20)
(21)                            (22)
(23)                            (24)

Task Number:  2
(1)  Name:  integx1                                   (2)  Type:  Task
(3)  Upper Network:  0  dblinteg
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  x=initlx+i*delx;funcdofx;funccofx;
dofx=funcdofx;cofx=funccofx;HX=(dofx-cofx)/numdivy;
y=dofx;f;f1=f;y=cofx;f;f2=f;tempi=2*int(i/2);
 k1=f1+f2;k2=0;k3=0;
j=1;
(10) Decision Type:  Single choice
     Following Task/Network:        Probability Of Taking
      Number:          Name:          This Path:
(11)  3                integy   (12)  1;
(13)                            (14)
(15)                            (16)
(17)                            (18)
(19)                            (20)
(21)                            (22)
(23)                            (24)

Task Number:  3
(1)  Name:  integy                                    (2)  Type:  Task
(3)  Upper Network:  0  dblinteg
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  y=cofx+j*HX;f;z=f;
temp=2*int(j/2);
if temp == j then k2=k2+z else k3=k3+z;
L=(k1+2*k2+4*k3)*HX/3;
j=j+1 ;
(10) Decision Type:  Tactical
     Following Task/Network:        Tactical Expression:
      Number:          Name:
(11)  3                integy   (12)  j<=numdivy-1;
(13)  4                integx   (14)  j > numdivy-1;
```

```
Task Number:  4
(1)  Name:  integx2                                          (2)  Type:  Task
(3)  Upper Network:  0  dblinteg
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:
if i == 0 ! i == numdivx then sum1=sum1+L else if tempi==i then
sum2=sum2+L else sum3=sum3+L;
if i == numdivx then endinteg=1;i=i+1;
(10) Decision Type:  Tactical
     Following Task/Network:        Tactical Expression:
      Number:        Name:
(11)  2              integx   (12)  endinteg==0;
(13)  5              end      (14)  endinteg==1;
(15)                          (16)
(17)                          (18)
(19)                          (20)
(21)                          (22)
(23)                          (24)

Task Number:  5
(1)  Name:  end                                              (2)  Type:  Task
(3)  Upper Network:  0  dblinteg
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviatinn:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  sum=(sum1+2*sum2+4*sum3)*delx/3;
(10) Decision Type:  Last task
     Following Task/Network:        Probability Of Taking
      Number:        Name:           This Path:
(11)                          (12)
(13)                          (14)
(15)                          (16)
(17)                          (18)
(19)                          (20)
(21)                          (22)
(23)                          (24)
```

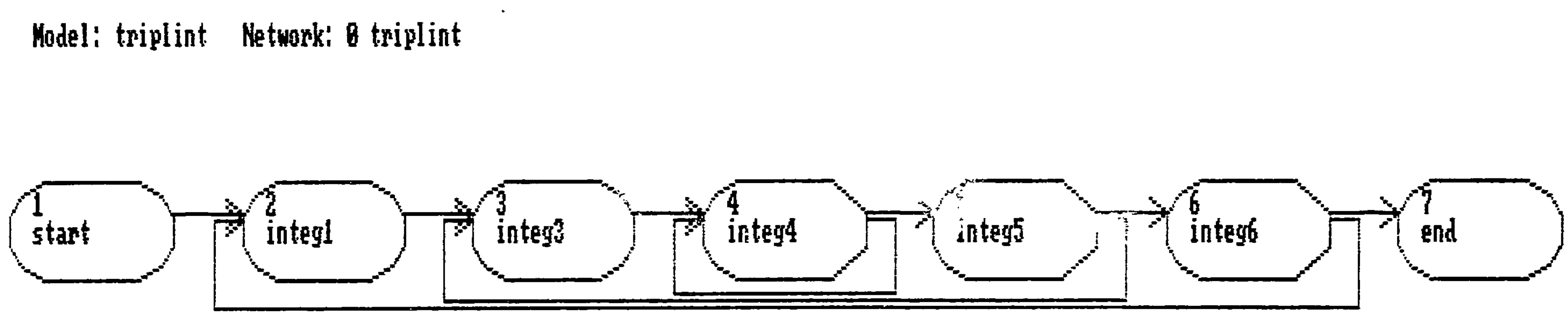


Figure 5. Micro SAINT diagram for a method of triple integration of a function f(x,y,z).

TEST NETWORK FOR TRIPLINT PROGRAM

```
Network Number:  0
(1)  Name:  triplint                                        (2)  Type:  Network
(3)  Upper Network:
(4)  Release Condition:  1;
(5)  First sub-job:  1  start
(6)  Sub-jobs (each can be task or network):
Number:          Name:                  Type:
1                start                  Task
2                integ1                 Task
3                integ3                 Task
4                integ4                 Task
5                integ5                 Task
6                integ6                 Task
7                end                    Task

Task Number:  1
(1)  Name:  start                                           (2)  Type:  Task
(3)  Upper Network:  0  triplint
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:
initl;n=numdivx/2;m=numdivy/2;p=numdivz/2;h=(endx-initlx)/(2*n);
j1=0;j2=0;j3=0;i=0;j=0;
(10) Decision Type:  Single choice
     Following Task/Network:           Probability Of Taking
      Number:          Name:             This Path:
(11)  2                integ1     (12)   1;
(13)                              (14)
(15)                              (16)
(17)                              (18)
(19)                              (20)
(21)                              (22)
(23)                              (24)

Task Number:  2
(1)  Name:  integ1                                          (2)  Type:  Task
(3)  Upper Network:  0  triplint
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  x=initlx+i*h;dx;cx;hx=(dx-cx)/(2*m);
k1=0;k2=0;k3=0;j=0;
(10) Decision Type:  Single choice
     Following Task/Network:           Probability Of Taking
      Number:          Name:             This Path:
(11)  3                integ3     (12)   1;
(13)                              (14)
(15)                              (16)
(17)                              (18)
(19)                              (20)
(21)                              (22)
(23)                              (24)

Task Number:  3
(1)  Name:  integ3                                          (2)  Type:  Task
(3)  Upper Network:  0  triplint
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:
y=cx+j*hx;betaxy;z=betaxy;fxyz;f1=fxyz;alphaxy;
hy=(betaxy-alphaxy)/(2*p);z=alphaxy;fxyz;f2=fxyz;L1=f1+f2;
L2=0;L3=0;k=1;
(10) Decision Type:  Single choice
     Following Task/Network:           Probability Of Taking
      Number:          Name:             This Path:
(11)  4                integ4     (12)   1;
(13)                              (14)
(15)                              (16)
```

```
Task Number:  4
(1)  Name:  integ4                                        (2)  Type:  Task
(3)  Upper Network:  0  triplint
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  z=alphaxy+k*hy;fxyz;qq=fxyz;
if int(k/2)==k/2 then L2=L2+qq else L3=L3+qq;
k=k+1;
(10) Decision Type:  Tactical
     Following Task/Network:          Tactical Expression:
      Number:           Name:
(11)  5                 integ5   (12)  k ==2*p;
(13)  4                 integ4   (14)  k < 2*p;
(15)                             (16)
(17)                             (18)
(19)                             (20)
(21)                             (22)
(23)                             (24)

Task Number:  5
(1)  Name:  integ5                                        (2)  Type:  Task
(3)  Upper Network:  0  triplint
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  L=(L1+2*L2+4*L3)*hy/3;
if j==0 |j==2 then k1=k1+L else if int(j/2)==j/2 then k2=k2+L else
k3=k3+L;
j=j+1;
(10) Decision Type:  Tactical
     Following Task/Network:          Tactical Expression:
      Number:           Name:
(11)  6                 integ6   (12)  j==2*m+1;
(13)  3                 integ3   (14)  j <2*m+1;
(15)                             (16)
(17)                             (18)
(19)                             (20)
(21)                             (22)
(23)                             (24)

Task Number:  6
(1)  Name:  integ6                                        (2)  Type:  Task
(3)  Upper Network:  0  triplint
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  ksum=(k1+2*k2+4*k3)*hx/3;
if i==0 | i==2*n then j1=j1+ksum else if int(i/2)==i/2 then j2=j2+ksum
else j3=j3+ksum;
i=i+1;
(10) Decision Type:  Tactical
     Following Task/Network:          Tactical Expression:
      Number:           Name:
(11)  7                 end      (12)  i==2*n+1;
(13)  2                 integ1   (14)  i <2*n+1;
(15)                             (16)
(17)                             (18)
(19)                             (20)
(21)                             (22)
(23)                             (24)

Task Number:  7
(1)  Name:  end                                           (2)  Type:  Task
(3)  Upper Network:  0  triplint
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  jsum=(j1+j2*2+4*j3)*h/3;
(10) Decision Type:  Last task
     Following Task/Network:          Probability Of Taking
      Number:           Name:           This Path:
(11)                             (12)
(13)                             (14)
(15)                             (16)
```

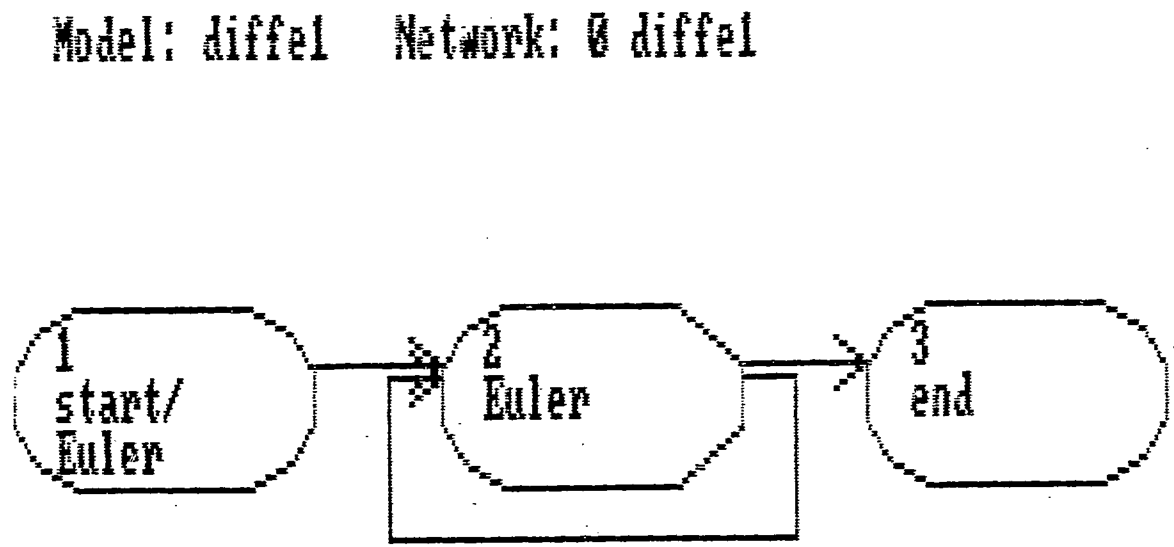


Figure 6. Micro SAINT diagram for Euler method for solving a differential equation.

TASK NETWORK FOR DIFFE1 PROGRAM

```
Network Number:  0
(1)   Name:  diffe1                                          (2)   Type:   Network
(3)   Upper Network:
(4)   Release Condition:  1;
(5)   First sub-job:  1  start/  Euler
(6)   Sub-jobs (each can be task or network):
Number:           Name:                 Type:
1                 start/  Euler         Task
2                 Euler                 Task
3                 end                   Task

Task Number:  1
(1)   Name:  start/  Euler                                   (2)   Type:   Task
(3)   Upper Network:  0  diffe1
(4)   Release Condition:  1;
(5)   Time Distribution Type:  Normal
(6)   Mean Time:  0;
(7)   Standard deviation:  0;
(8)   Task's beginning effect:
(9)   Task's ending effect:  initl;delx=(endx-initlx)/numintvl;
xin=initlx;yout=initly;
k=1;
(10)  Decision Type:  Single choice
      Following Task/Network:        Probability Of Taking
        Number:          Name:           This Path:
(11)    2                Euler      (12)  1;
(13)                                (14)
(15)                                (16)
(17)                                (18)
(19)                                (20)
(21)                                (22)
(23)                                (24)

Task Number:  2
(1)   Name:  Euler                                           (2)   Type:   Task
(3)   Upper Network:  0  diffe1
(4)   Release Condition:  1;
(5)   Time Distribution Type:  Normal
(6)   Mean Time:  0;
(7)   Standard deviation:  0;
(8)   Task's beginning effect:
(9)   Task's ending effect:
x=xin;y=yout;f;yout=yout+delx*f;k=k+1;xin=xin+delx;
(10)  Decision Type:  Tactical
      Following Task/Network:        Tactical Expression:
        Number:          Name:
(11)    2                Euler      (12)  k<=numintvl;
(13)    3                end        (14)  k>numintvl;
(15)                                (16)
(17)                                (18)
(19)                                (20)
(21)                                (22)
(23)                                (24)

Task Number:  3
(1)   Name:  end                                             (2)   Type:   Task
(3)   Upper Network:  0  diffe1
(4)   Release Condition:  1;
(5)   Time Distribution Type:  Normal
(6)   Mean Time:  1;
(7)   Standard deviation:  0;
(8)   Task's beginning effect:
(9)   Task's ending effect:
(10)  Decision Type:  Last task
      Following Task/Network:        Probability Of Taking
        Number:          Name:           This Path:
(11)                                (12)
```

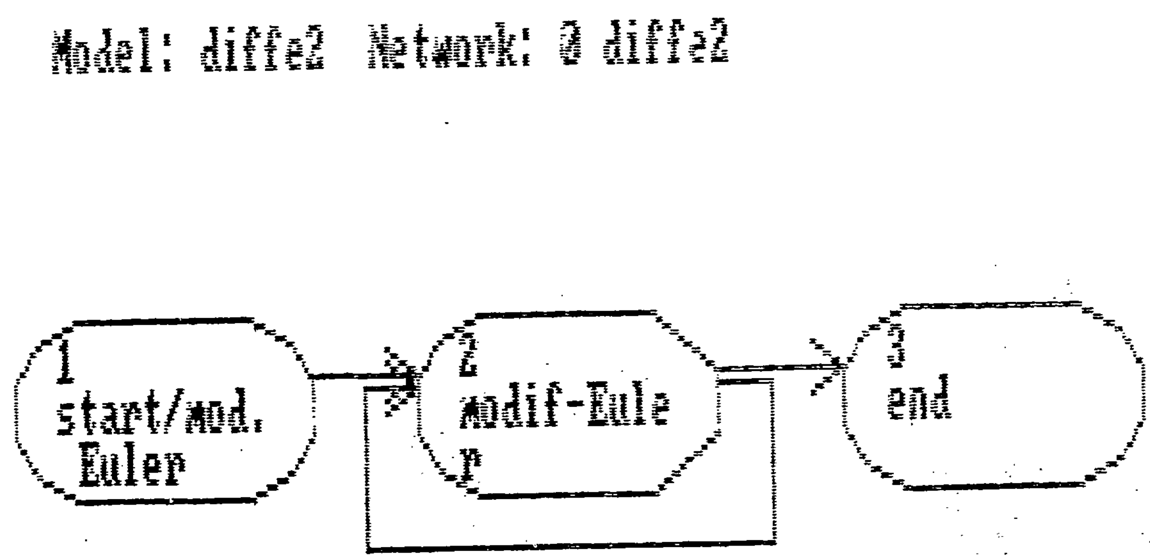


Figure 7. Micro SAINT diagram for modified Euler method for solving a differential equation.

TASK NETWORK FOR DIFFE2 PROGRAM

```
Network Number:  0
(1)  Name:  diffe2                                (2)  Type:  Network
(3)  Upper Network:
(4)  Release Condition:  1;
(5)  First sub-job:  1  start/mod. Euler
(6)  Sub-jobs (each can be task or network):
Number:        Name:                 Type:
1              start/mod. Euler      Task
2              modif-Euler           Task
3              end                   Task

Task Number:  1
(1)  Name:  start/mod. Euler                      (2)  Type:  Task
(3)  Upper Network:  0  diffe2
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  initl;delx=(endx-initlx)/numintvl;
xin=initlx;yout=initly;
k=1;
(10) Decision Type:  Single choice
     Following Task/Network:        Probability Of Taking
      Number:         Name:          This Path:
(11)  2               modif-   (12)  1;
(13)                           (14)
(15)                           (16)
(17)                           (18)
(19)                           (20)
(21)                           (22)
(23)                           (24)

Task Number:  2
(1)  Name:  modif-Euler                           (2)  Type:  Task
(3)  Upper Network:  0  diffe2
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  x=xin;y=yout;f;y=yout+delx*f;f;cf=f;
yout=yout+delx*(f+cf)/2;k=k+1;
xin=xin+delx;
(10) Decision Type:  Tactical
     Following Task/Network:        Tactical Expression:
      Number:         Name:
(11)  2               modif-   (12)  k<=numintvl;
(13)  3               end      (14)  k>numintvl;
(15)                           (16)
(17)                           (18)
(19)                           (20)
(21)                           (22)
(23)                           (24)

Task Number:  3
(1)  Name:  end                                   (2)  Type:  Task
(3)  Upper Network:  0  diffe2
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  1;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:
(10) Decision Type:  Last task
     Following Task/Network:        Probability Of Taking
      Number:         Name:          This Path:
(11)                           (12)
```

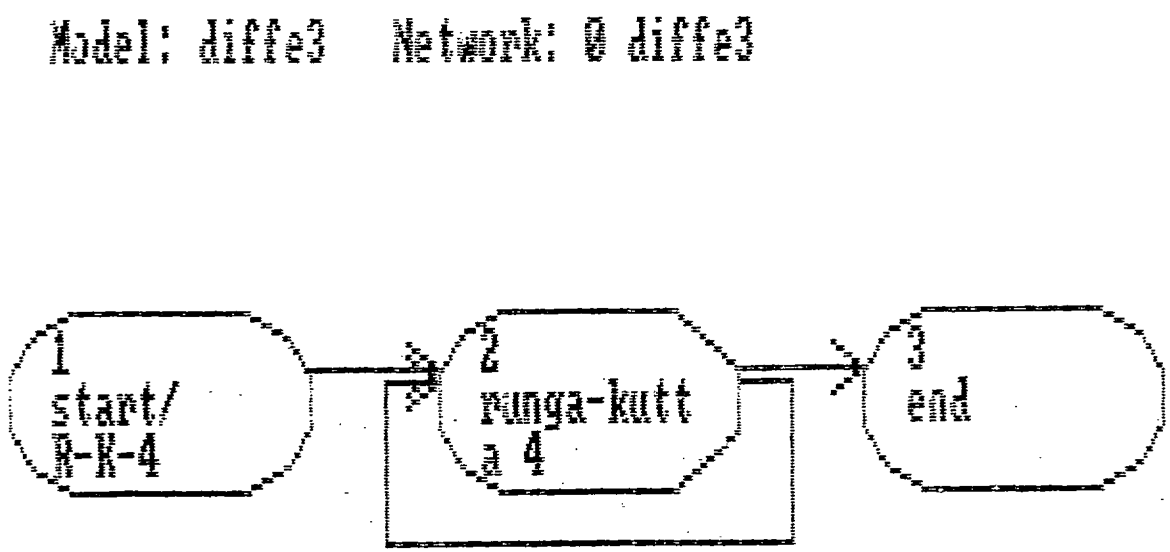


Figure 8. Micro SAINT diagram for Runga Kutta method for solving a differential equation.

TASK NETWORK FOR DIFFE3 PROGRAM

```
Network Number:  0
(1)  Name:  diffe3                                        (2)  Type:  Network
(3)  Upper Network:
(4)  Release Condition:  1;
(5)  First sub-job:  1  start/   R-K-4
(6)  Sub-jobs (each can be task or network):
Number:         Name:                 Type:
1               start/   R-K-4        Task
2               runga-kutta 4         Task
3               end                   Task

Task Number:  1
(1)  Name:  start/   R-K-4                                (2)  Type:  Task
(3)  Upper Network:  0  diffe3
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  initl;delx=(endx-initlx)/numintvl;
xin=initlx;yout=initly;k=1;
(10) Decision Type:  Single choice
     Following Task/Network:       Probability Of Taking
       Number:         Name:          This Path:
(11)   2               runga-   (12)  1;
(13)                            (14)
(15)                            (16)
(17)                            (18)
(19)                            (20)
(21)                            (22)
(23)                            (24)

Task Number:  2
(1)  Name:  runga-kutta 4                                 (2)  Type:  Task
(3)  Upper Network:  0  diffe3
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:
x=xin;y=yout;f;k1=delx*f;x=xin+delx/2;y=yout+k1/2;
f;k2=delx*f;y=yout+k2/2;f;k3=delx*f;x=xin+delx;
y=yout+k3;f;k4=delx*f;
yout=yout+(k1+2*(k2+k3)+k4)/6;
xin=x;k=k+1;
(10) Decision Type:  Tactical
     Following Task/Network:       Tactical Expression:
       Number:         Name:
(11)   2               runga-   (12)  k<=numintvl;
(13)   3               end      (14)  k>numintvl;
(15)                            (16)
(17)                            (18)
(19)                            (20)
(21)                            (22)
(23)                            (24)

Task Number:  3
(1)  Name:  end                                           (2)  Type:  Task
(3)  Upper Network:  0  diffe3
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  1;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:
(10) Decision Type:  Last task
     Following Task/Network:       Probability Of Taking
       Number:         Name:          This Path:
(11)                            (12)
```

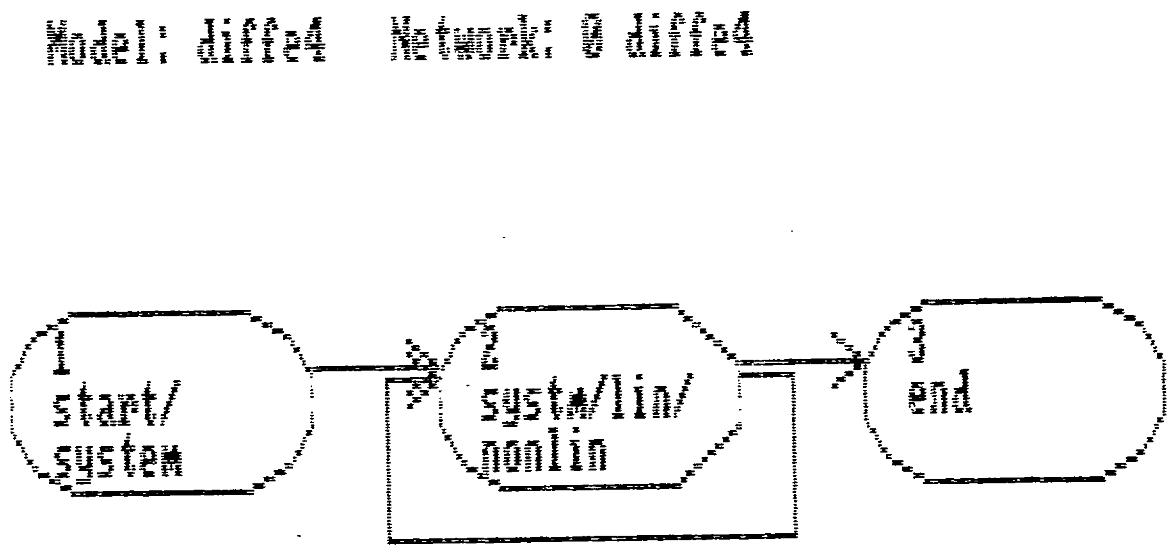


Figure 9. Micro SAINT diagram for solving a system of differential equations.

TASK NETWORK FOR DIFFE4 PROGRAM

```
Network Number:  0
(1)  Name:  diffe4                                      (2)  Type:  Network
(3)  Upper Network:
(4)  Release Condition:  1;
(5)  First sub-job:  1  start/  system
(6)  Sub-jobs (each can be task or network):
Number:        Name:                Type:
1              start/  system       Task
2              systm/lin/nonlin     Task
3              end                  Task

Task Number:  1
(1)  Name:  start/  system                              (2)  Type:  Task
(3)  Upper Network:  0  diffe4
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  initl;delx=(endx-initlx)/numintvl;
xin=initlx;u1out=initlu1 ;u2out=initlu2;
k=1;
(10) Decision Type:  Single choice
     Following Task/Network:        Probability Of Taking
      Number:        Name:           This Path:
(11)  2              systm/   (12)  1;
(13)                          (14)
(15)                          (16)
(17)                          (18)
(19)                          (20)
(21)                          (22)
(23)                          (24)

Task Number:  2
(1)  Name:  systm/lin/nonlin                            (2)  Type:  Task
(3)  Upper Network:  0  diffe4
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  x=xin;u1=u1out;u2=u2out;
f1;f2;k11=delx*f1;k12=delx*f2;x=xin+delx/2;
u1=u1out+k11/2;u2=u2out+k12/2;f1;f2;k21=delx*f1;
k22=delx*f2;u1=u1out+k21/2;u2=u2out+k22/2;f1;f2;
k31=delx*f1;k32=delx*f2;u1=u1out+k31;u2=u2out+k32;
x=xin+delx;f1;f2;k41=delx*f1;k42=delx*f2;
u1out=u1out+(k11+2*(k21+k31)+k41)/6;
u2out=u2out+(k12+2*(k22+k32)+k42)/6;
xin=xin+delx;k=k+1;
(10) Decision Type:  Tactical
     Following Task/Network:        Tactical Expression:
      Number:        Name:
(11)  2              systm/   (12)  k<=numintvl;
(13)  3              end      (14)  k>numintvl;
(15)                          (16)
(17)                          (18)
(19)                          (20)
(21)                          (22)
(23)                          (24)

Task Number:  3
(1)  Name:  end                                         (2)  Type:  Task
(3)  Upper Network:  0  diffe4
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  1;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:
(10) Decision Type:  Last task
     Following Task/Network:        Probability Of Taking
      Number:        Name:           This Path:
(11)                          (12)
```

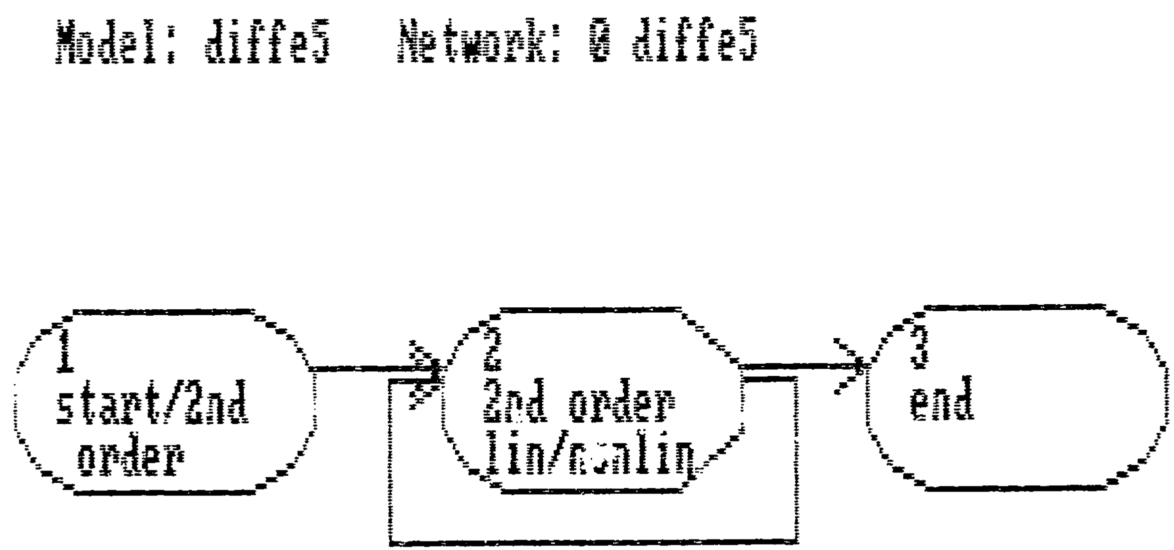


Figure 10. Micro-SAINT diagram for solving a second order for differential equation.

TASK NETWORK FOR DIFFE5 PROGRAM

```
Network Number:  0
(1)  Name:  diffe5                                    (2)  Type:  Network
(3)  Upper Network:
(4)  Release Condition:  1;
(5)  First sub-job:  1  start/2nd  order
(6)  Sub-jobs (each can be task or network):
Number:         Name:                Type:
1               start/2nd  order     Task
2               2nd order lin/nonlin  Task
3               end                  Task

Task Number:  1
(1)  Name:  start/2nd  order                          (2)  Type:  Task
(3)  Upper Network:  0  diffe5
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  initl;delx=(endx-initlx)/numintvl;
xin=initlx;u1out=initlu1 ;u2out=initlu2;
k=1;error=0;
(10) Decision Type:  Single choice
     Following Task/Network:        Probability Of Taking
      Number:          Name:          This Path:
(11)  2               2nd or   (12)  1;
(13)                           (14)
(15)                           (16)
(17)                           (18)
(19)                           (20)
(21)                           (22)
(23)                           (24)

Task Number:  2
(1)  Name:  2nd order lin/nonlin                      (2)  Type:  Task
(3)  Upper Network:  0  diffe5
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:  x=xin;u1=u1out;u2=u2out;f1=u2;f2;
k11=delx*f1;k12=delx*f2;x=xin+delx/2;
u1=u1out+k11/2;u2=u2out+k12/2;f1=u2;f2;
k21=delx*f1;k22=delx*f2;u1=u1out+k21/2;
u2=u2out+k22/2;f1=u2;f2;k31=delx*f1;k32=delx*f2;
u1=u1out+k31;u2=u2out+k32;x=xin+delx;f1=u2;f2;
k41=delx*f1;k42=delx*f2;
u1out=u1out+(k11+2*(k21+k31)+k41)/6;
u2out=u2out+(k12+2*(k22+k32)+k42)/6;
xin=xin+delx;k=k+1;
(10) Decision Type:  Tactical
     Following Task/Network:        Tactical Expression:
      Number:          Name:
(11)  2               2nd or   (12)  k<=numintvl;
(13)  3               end      (14)  k>numintvl;
(15)                           (16)
(17)                           (18)
(19)                           (20)
(21)                           (22)
(23)                           (24)

Task Number:  3
(1)  Name:  end                                       (2)  Type:  Task
(3)  Upper Network:  0  diffe5
(4)  Release Condition:  1;
(5)  Time Distribution Type:  Normal
(6)  Mean Time:  0;
(7)  Standard deviation:  0;
(8)  Task's beginning effect:
(9)  Task's ending effect:
(10) Decision Type:  Last task
     Following Task/Network:        Probability Of Taking
      Number:          Name:          This Path:
(11)                           (12)
```

APPENDIX B

Graphical Representation for Solutions of Differential Equations.

APPENDIX B

In this appendix we present actual Micro SAINT graphical outputs for solutions of the following cases:

a. Runga-Kutta method for

$y' = 1 - y + x, \; y(0) = 1.$

b. Runga-Kutta method for the system:

$\frac{dI_1}{dt} = f_1(t, I_1, I_2) = -4I_1 + 3I_2 + 6, I_1(0) = 0,$

$\frac{dI_2}{dt} = f_2(t, I_1, I_2) = -2.4I_1 + 1.6I_2 + 3.6, I_2(0) = 0.$

c. Runga-Kutta method for second-order differential equation:

$x^2y'' - 2xy' + 2y = x^3 Ln(x), y(1) = 1, y'(1) = 0.$

figure 10. solution of 1st order ODE-RK4

y'=1-y+x,y(0)=1.

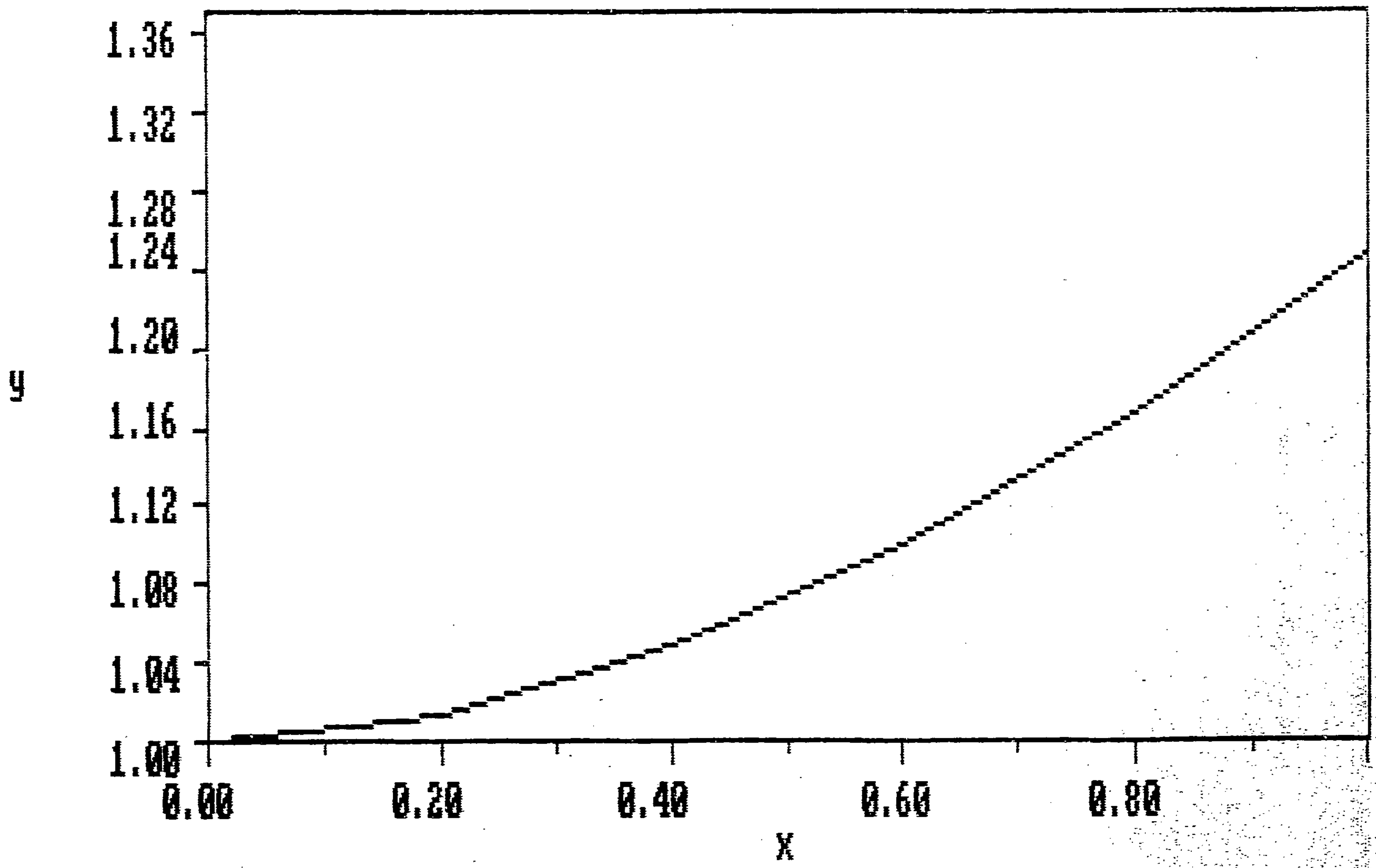


figure 11. solution of 1st order system of

ODE for the electric circuit problem.

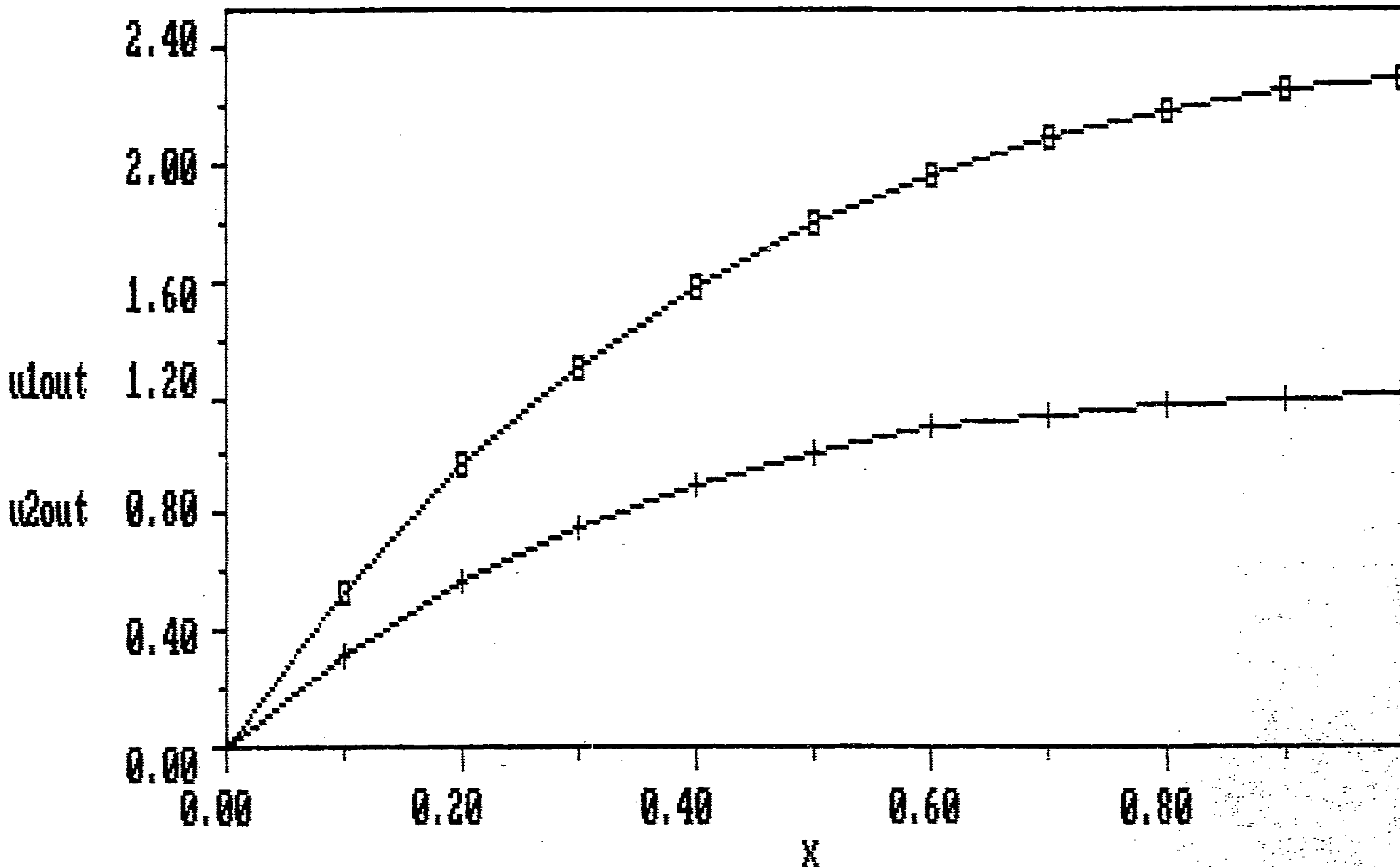


figure 12. solution of 2nd order ODE.

x^2*y''-2*x*y'+2*y=x^3*ln(x),y(1)=1,y'(1)=0.