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

The opinions and interpretations contained herein are those of the authors and do not necessarily represent the views, policies, or endorsement of the Department of the Navy or any other government agency.

Trade names of materials and/or products of commercial or nongovernment organizations are cited as needed for precision. These citations do not constitute official endorsement or approval of the use of such commercial materials and/or products.

Reproduction in whole or part is permitted for any purpose of the United States Government.
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 psychophysical 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 psychophysical performance of naval aircrew personnel under sustained operational work schedules.
CONTENTS

ABSTRACT ......................................................................................................................... v

1. INTRODUCTION ............................................................................................................. 1

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

   Trapezoidal Rule ............................................................................................................. 3
   Simpson's Rule ................................................................................................................ 3
   Composite Simpson's Rule for Double Integrals ............................................................. 4
   Composite Simpson's Rule for Triple Integrals .............................................................. 5

3. NUMERICAL SOLUTION OF DIFFERENTIAL EQUATIONS ............................................. 6

   First-order Ordinary Differential Equation ................................................................ 6
   Euler's Method ................................................................................................................ 6
   Modified Euler's Method .............................................................................................. 6
   Runge-Kutta Method of Order Four ............................................................................. 7

   First-order System of Ordinary Differential Equations ............................................... 7

   Second-order Ordinary Differential Equation ............................................................. 9

4. CONCLUSION ................................................................................................................ 10

REFERENCES .................................................................................................................... 11

APPENDIX A. Micro SAINT Diagrams and Computer Programs ....................................... A-1

APPENDIX B. Micro SAINT Graphical Representation for Solutions of
   Differential Equations .................................................................................................. B-1
This page intentionally left blank.
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. Olifson, 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.
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. Runge-Kutta method of order four for solving a first-order ordinary differential equation.

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

i. Runge-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:

- \texttt{inilx} = initial value of the independent variable \(x\),
- \texttt{endx} = end value of the independent variable \(x\),
- \texttt{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 \( j = 0,1,2,...,n \), the trapezoidal rule for \( n \) subintervals is:

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

\[
\text{Error} = 0(h^2).
\]

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

- \text{initx} = a, \text{endx} = b, \text{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 Intgtprz Program.**

<table>
<thead>
<tr>
<th>FUNCTION LIBRARY</th>
<th>Model Name: intgtprz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name:</td>
<td>Expression:</td>
</tr>
<tr>
<td>( f )</td>
<td>( 1+x );</td>
</tr>
<tr>
<td>\text{initl}</td>
<td>\text{numintvl}=10; \text{initl}=0; \text{endx}=1;</td>
</tr>
</tbody>
</table>

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 \( j = 0,1,2,...,2m \), the Simpson's Rule for \( n \) subintervals is:

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

\[
\text{Error} = 0(h^4).
\]

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

$$\int_a^b \int_{c(x)}^{d(x)} f(x,y) \, dy \, dx,$$

the user needs to supply the functions $f(x,y)$, $c(x)$, and $d(x)$ as well as the parameters $\text{initlx} = a$, $\text{endx} = b$, 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

$$\int_0^1 \int_{x^2}^{x} (xy) \, dy \, dx.$$

Here $\text{initlx} = 0$, $\text{endx} = 1$, $c(x) = x^2$, $d(x) = x$, 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.**

<table>
<thead>
<tr>
<th>FUNCTION</th>
<th>LIBRARY</th>
<th>Model Name: dblinteg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name: f</td>
<td>Expression:</td>
<td>$x*y$;</td>
</tr>
<tr>
<td>initlx</td>
<td>initlx=0;endx=1;numdivx=10;numdivy=10;</td>
<td></td>
</tr>
<tr>
<td>funcdofx</td>
<td>x;</td>
<td></td>
</tr>
<tr>
<td>funccofx</td>
<td>$x^2$x;</td>
<td></td>
</tr>
</tbody>
</table>

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

\[
\int_{x=a}^{x=b} \int_{y=c(x)}^{y=d(x)} \int_{z=\alpha(x,y)}^{z=\beta(x,y)} f(x,y,z) \, dz \, dy \, dx,
\]

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 \( \text{init}x = a \), \( \text{end}x = b \), and the number of divisions, \( \text{num}divx \), \( \text{num}divy \), and \( \text{num}divz \), along \( x \), \( y \), and \( z \) axes, respectively; "\text{num}divx," "\text{num}divy," and "\text{num}divz" must be even numbers. The approximate value of the integral is stored in the output snapshot.

As an illustration, consider the integral

\[
\int_{x=0}^{x=1} \int_{y=x}^{y=2} \int_{z=xy}^{z=2} (xyz) \, dz \, dy \, dx.
\]

Here \( \text{init}x = 0 \), \( \text{end}x = 1 \), \( c(x) = x \), \( d(x') = x \), \( \alpha(x,y) = xy \), \( \beta(x,y) = 2 \), and \( f(x,y,z) = xyz \). The user supplies this information as well as \( \text{num}divx \), \( \text{num}divy \), and \( \text{num}divz \) in the "function library" of Micro SAINT as follows:

**TABLE 3. Function Library Input for Triplint Program.**

<table>
<thead>
<tr>
<th>Name</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>( cx )</td>
<td>( x^x )</td>
</tr>
<tr>
<td>( dx )</td>
<td>( x )</td>
</tr>
<tr>
<td>( betaxy )</td>
<td>( 2 )</td>
</tr>
<tr>
<td>( alphaxy )</td>
<td>( x^y )</td>
</tr>
<tr>
<td>( fxyz )</td>
<td>( x^y^z )</td>
</tr>
<tr>
<td>( \text{init}l )</td>
<td>( \text{init}lx=0;\text{end}x=1;\text{num}divx=10;\text{num}divy=10;\text{num}divz=10; )</td>
</tr>
</tbody>
</table>

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 Runge-Kutta of order four for solving the initial value problem.

\[ \frac{dy}{dx} = f(x,y), \quad y(x_0) = y_0. \]

1. Euler's method:

The difference equation associated with Euler method is:

\[ y_i = y_{i-1} + hf(x_{i-1},y_{i-1}) + \text{Error}, \quad \text{for } i = 1, 2, 3, \ldots, n, \quad \text{where } n = \text{number of intervals}, \quad x_i = x_0 + ih, \quad \text{and } \text{Error} = O(h^2). \]

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

\[ \frac{dy}{dx} = -y + x + 1, \quad y(0) = 1, \quad \text{using numdivx} = 10. \]

TABLE 4. Function Library Input for Diffeq Program.

<table>
<thead>
<tr>
<th>FUNCTION LIBRARY</th>
<th>Model Name: diffel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name:</td>
<td>Expression:</td>
</tr>
<tr>
<td>f</td>
<td>( f = y + x; )</td>
</tr>
<tr>
<td>initlx</td>
<td>initlx=0;endx=1;numintvl=10;initly=1;</td>
</tr>
</tbody>
</table>

2. Modified Euler's method:

The modified Euler's method is a predictor-corrector method.

The difference equation associated with the method is:

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

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

\( i = 1, 2, \ldots, n, \) \text{where } n = \text{number of intervals and } x_i = x_0 + 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:

$$
\begin{align*}
  k_1 &= h f(x_{i-1}, y_{i-1}), \\
  k_2 &= h f(x_{i-1} + h/2, y_{i-1} + k/2), \\
  k_3 &= h f(x_{i-1} + h/2, y_{i-1} + k/2), \\
  k_4 &= h f(x_i, y_{i-1} + k_3), \\
  y_i &= y_{i-1} + (k_1 + 2k_2 + 2k_3 + k_4)/6, \\
  i &= 1, 2, \ldots, n.
\end{align*}
$$

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 $u_2$:

$$
\begin{align*}
  \frac{du_1}{dx} &= f_1(x, u_1, u_2), \\
  \frac{du_2}{dx} &= f_2(x, u_1, u_2), \\
  u_1(x_0) &= \alpha, \\
  u_2(x_0) &= \beta.
\end{align*}
$$

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

$$
\begin{align*}
  k_{1,i} &= h f_1(x_j, u_1, u_2), \\
  k_{2,i} &= h f_1(x_j + h/2, u_{1j} + 0.5k_{11} + 0.5k_{12}), \\
  k_{3,i} &= h f_1(x_j + h/2, u_{1j} + 0.5k_{21} + 0.5k_{22}), \\
  k_{4,i} &= h f_1(x_j + h, u_{1j} + k_{31} + k_{32}), \\
  u_{1,j+1} &= u_{1,j} + (k_{11} + 2k_{21} + 2k_{31} + k_{41})/6, \\
  u_{2,j+1} &= u_{2,j} + (k_{12} + 2k_{22} + 2k_{32} + k_{42})/6.
\end{align*}
$$
As an illustration of an application, we use an example [2], about the use of Kirchhoff's Law in circuit theory. Assuming that the switch in the circuit shown in Fig. 1 is closed at time \( t = 0 \),

\[
\begin{align*}
\frac{dI_1}{dt} &= f_1(t, I_1, I_2) = -4I_1 + 3I_2 + 6, I_1(0) = 0, \\
\frac{dI_2}{dt} &= f_2(t, I_1, I_2) = -2.4I_1 + 1.6I_2 + 3.6, I_2(0) = 0.
\end{align*}
\]

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.**

<table>
<thead>
<tr>
<th>Function Library Model Name: diffe4</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>( f_1 )</td>
</tr>
<tr>
<td>( f_2 )</td>
</tr>
<tr>
<td>initl</td>
</tr>
</tbody>
</table>
c. Second-order ordinary differential equation:

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

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

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

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

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

\[ y(1) = 1, \quad y'(1) = 0. \]

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 \( \frac{dy}{dx} \) is replaced by \( u_2 \).
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 Runge-Kutta, are adequate for problems where the function is easy to evaluate and the accuracy needed is small (about 10\(^{-4}\) for Runge-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

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:

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).
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:
initt;:=i;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:

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:
sum=sum+(prevf+f);prevf=f; j=j+1;
(10) Decision Type: Tactical
 Following Task/Network: Tactical Expression:
Number: Name:

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:

A-3
Model: intgsimp     Network: Ø intgsimp

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
(3) Upper Network:  
(4) Release Condition: 1;  
(5) First sub-job: 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  
(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;  
\[ \text{delx} = (\text{endx} - \text{initx}) / \text{numintvl} = \text{numintvl}/2; \]  
\[ j = \text{if} (\text{xin} - \text{initx} < \text{xin}; \text{tempf} = \text{f}); \]  
(10) Decision Type: Single choice  
Following Task/Network: Probability Of Taking

```
Number: Name: This Path:
12 addarea (12) 1;
13 (14)
15 (16)
17 (18)
19 (20)
21 (22)
23 (24)
```

Task Number: 2  
(1) Name: addarea  
(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:  
\[ \text{xin} = \text{xin} + \text{delx}; \]  
\[ \text{f1} = \text{f}; \]  
\[ \text{f2} = \text{f}; \]  
\[ \text{sum} = \text{sum} + \text{tempf} + 4* \text{f1} + \text{f2}; \]  
\[ \text{xin} = \text{xin}; \]  
\[ \text{tempf} = \text{f2}; \]  
\[ j = j + 1; \]  
(10) Decision Type: Tactical  
Following Task/Network: Tactical Expression:  

```
Number: Name: This Path:
12 addarea (12) j < m+1;  
13 end (14) j >= m+1;  
15 (16)
17 (18)
19 (20)
21 (22)
23 (24)
```

Task Number: 3  
(1) Name: end  
(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:  
\[ \text{sum} = \text{sum} \times \text{delx}/3; \]  
(10) Decision Type: Last task  
Following Task/Network: Probability Of Taking

```
Number: Name: This Path:
12 (12)
```
Model: dblinteg  Network: Ø dblinteg

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

Network 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: x=initlx+i*delx;funcdofx;funccofx;
dofx=funcdofx;cofx=funccofx;HX=(dofx-cofx)/numdivy;
y=dofx;f1=f;f2=f;f3=f;tempi=2*int(i/2);
k1=f1+f2;k2=0;k3=0;
if temp == j then k2=k2+z else k3=k3+z;
L=(f1+2*k2+4*k3)*HX/3;
j=j+1;
(10) Decision Type: Tactical
Following Task/Network: Tactical Expression:
Number: 3
(1) Name: integx
(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=(f1+2*k2+4*k3)*HX/3;
j=j+1;
(10) Decision Type: Tactical
Following Task/Network: Tactical Expression:
Number: 4
Task Number: 4
(1) Name: integx2
(2) Type: Task
(3) Upper Network: 0 dblinteg
(4) Release Condition: i;
(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 temp[i==i then
  sum2=sum2+L else sum3=sum3+L;
if i == numdivx then endinteg=i;i=i+1;
(10) Decision Type: Tactical
  Following Task/Network: Tactical Expression:
  Number: 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: i;
(5) Time Distribution Type: Normal
(6) Mean Time: 0;
(7) Standard deviation: 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)
Model: triplint  Network: 0 triplint

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):
7) Name: Type:
   1) start Task
   2) integ1 Task
   3) integ2 Task
   4) integ3 Task
   5) integ4 Task
   6) integ5 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:
10) Decision Type: Single choice

Following Task/Network:
Number: Name: This Path:
   2) integ1 (12) 1;

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:
10) Decision Type: Single choice

Following Task/Network:
Number: Name: This Path:
   3) integ3 (12) 1;

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:
10) Decision Type: Single choice

Following Task/Network:
Number: Name: This Path:
   4) integ4 (12) 1;


A-10
Task Number: 4
Name: integ4
Upper Network: 0 triplint
Release Condition: 1:
Time Distribution Type: Normal
Mean Time: 0:
Standard deviation: 0:
Task's beginning effect: $z = \alpha x y + k \times h y : f x y z : q q = f x y z$
if \( \text{int}(k/2) = k/2 \) then $L2 = l2 + q q$ else $L3 = L3 + q q$
Task Type: Tactical
Decision Type: Tactical
Following Task/Network Type: Tactical
Expression:
Number: Name:
4 integ5 (12) $k = 2^*p$
15 integ4 (16) $k < 2^*p$
17
19
23

Task Number: 5
Name: integ5
Upper Network: 0 triplint
Release Condition: 1:
Time Distribution Type: Normal
Mean Time: 0:
Standard deviation: 0:
Task's beginning effect: $L = (L1 + 2 \times L2 + 4 \times L3) \times h y / 3$
if $i = 0$ then $k1 = k1 + L$ else if $\text{int}(j/2) = j/2$ then $k2 = k2 + L$ else
$k3 = k3 + L$
Task Type: Tactical
Decision Type: Tactical
Following Task/Network Type: Tactical
Expression:
Number: Name:
6 integ6 (12) $j = 2^*m + 1$
3 integ3 (10) $j < 2^*m + 1$
19
20
23

Task Number: 6
Name: integ6
Upper Network: 0 triplint
Release Condition: 1:
Time Distribution Type: Normal
Mean Time: 0:
Standard deviation: 0:
Task's beginning effect: $ksum = (k1 + 2 \times k2 + 4 \times k3) \times h x / 3$
if $i = 0$ then $j1 = j1 + ksum$ else if $\text{int}(i/2) = i/2$ then $j2 = j2 + ksum$
else $j3 = j3 + ksum$
Task Type: Tactical
Decision Type: Tactical
Following Task/Network Type: Tactical
Expression:
Number: Name:
7 end (11) $i = 2^*n + 1$
2 integ1 (16) $i < 2^*n + 1$
17
19
23

Task Number: 7
Name: end
Upper Network: 0 triplint
Release Condition: 1:
Time Distribution Type: Normal
Mean Time: 0:
Standard deviation: 0:
Task's beginning effect: $jsum = (j1 + j2 + 2 \times j3) \times h y / 3$
Task Type: Last task
Expression Type: Last task
Following Task/Network Type: Probability Of Taking
Expression:
Number: Name:
13 This Path:
14
15

A-11
Model: diffel  Network: Ø diffel

Figure 6. Micro SAINT diagram for Euler method for solving a differential equation.
TASK NETWORK FOR DIFFEL PROGRAM

Network Number: 0
(1) Name: diffel
(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
(3) Upper Network: 0 diffel
(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;

Task Number: 2
(1) Name: Euler
(3) Upper Network: 0 diffel
(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; wyout+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;

Task Number: 3
(1) Name: end
(3) Upper Network: 0 diffel
(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)
Model: diffe2  Network: 0 diffe2

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: diffe2
(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: \( \text{init}_1 \); \( \text{del}_x = \frac{(\text{end}_x - \text{init}_x)}{\text{numintvl}} \);
\( \text{x}_i = \text{init}_i \); \( \text{y}_i = \text{init}_i \);
\( k = 1; \)

(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: \( x_i = x_{init} + \text{del}_x \); \( y_i = y_{init} + \text{del}_y \); \( f = f \);
\( y_{out} = y_{out} + \text{del}_x \times \frac{(f+cf)}{2}; k = k+1; \); \( x_{init} = x_{init} + \text{del}_x \);

(10) Decision Type: Tactical
Following Task/Network: Tactical Expression:
Number: Name:
(11) 2 modif- (12) \( k = \text{numintvl}; \)
(13) (14) \( k > \text{numintvl}; \)
(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)
Model: diffe3  Network: 0 diffe3

Figure 8. Micro SAINT diagram for Runge-Kutta method for solving a differential equation.
Network Number: 0
(1) Name: diffe3
(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):

<table>
<thead>
<tr>
<th>Number</th>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>start/ R-K-4</td>
<td>Task</td>
</tr>
<tr>
<td>2</td>
<td>runga-kutta 4</td>
<td>Task</td>
</tr>
<tr>
<td>3</td>
<td>end</td>
<td>Task</td>
</tr>
</tbody>
</table>

Task Number: 1
(1) Name: start/ R-K-4
(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: \( x_{in} = x_{init}, y_{in} = y_{in} + \frac{\Delta t}{N} \)
(9) Task's ending effect: \( x_{out} = x_{init}, y_{out} = y_{out} + \frac{\Delta t}{N} \)
(10) Decision Type: Single choice

Following Task/Network: Probability Of Taking

<table>
<thead>
<tr>
<th>Number</th>
<th>Name</th>
<th>This Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>runga-</td>
<td>1</td>
</tr>
</tbody>
</table>

Task Number: 2
(1) Name: runga-kutta 4
(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: \( x_{in} = x_{init}, y_{in} = y_{init} + k_{init} \)
(9) Task's ending effect: \( x_{out} = x_{init}, y_{out} = y_{init} + k_{init} \)
(10) Decision Type: Tactical

Following Task/Network: Tactical Expression:

<table>
<thead>
<tr>
<th>Number</th>
<th>Name</th>
<th>This Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>runga-</td>
<td>k &lt;= numintvl;</td>
</tr>
<tr>
<td>3</td>
<td>end</td>
<td>k &gt; numintvl;</td>
</tr>
</tbody>
</table>

Task Number: 3
(1) Name: end
(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: \( x_{in} = x_{init}, y_{in} = y_{init} + k_{init} \)
(9) Task's ending effect: \( x_{out} = x_{init}, y_{out} = y_{init} + k_{init} \)
(10) Decision Type: Last task

Following Task/Network: Probability Of Taking

<table>
<thead>
<tr>
<th>Number</th>
<th>Name</th>
<th>This Path</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
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: I start/ system
(6) Sub-jobs (each can be task or network):
Number: Name: Type:
1 start/ system Task
2 system/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: initlx=initlx+k12/1/2;
xin=initlx;
u1=initlv;
u2=initlu2;
k1=;
(10) Decision Type: Single choice
Following Task/Network:
Number: Name: This Path:
(11) 2 system/ (12) 1;
(13) 3 (14)
(15) (16)
(17) (18)
(19) (20)
(21) (22)
(23) (24)

Task Number: 2
(1) Name: system/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: m=xinlu1=u1out; u2=u2out;
fl; f2;
k12=delx*fl; k13=delx*f2; u1=ulout+k11/2; u2=ulout+k12/2; k12=
(10) Decision Type: Tactical
Following Task/Network:
Number: Name: Tactical Expression:
(11) 2 system/ (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:
Number: Name: This Path:
(11) (12)
Model: diffe5  Network: @ diffe5

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;ulout=initlu1;u2out=initlu2;
(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+ul1=ulout+u2=u2out;fl=f2f2;
k11=delx*fl;k12=delx*f2;\ldots;
u1=ulout+k11/2;u2=u2out+k12/2;
k21=delx*fl;k22=delx*f2;\ldots;
u1=ulout+k21/2;u2=ulout+k22/2;
(10) Decision Type: Tactical
Following Task/Network: Tactical Expression:
Number: Name: This Path:
(11) 2 2nd or (12) k<=numintvl;
(13) (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. Runge-Kutta method for
   \[ y' = 1-y + x, \ y(0) = 1. \]

b. Runge-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. Runge-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
Figure 11. Solution of 1st order system of ODE for the electric circuit problem.
figure 12. solution of 2nd order ODE.

\[ x^2y'' - 2xy' + 2y = x^3 \ln(x), y(1) = 1, y'(1) = 0. \]