Table of Contents

G12ISC 2018-2019 Coursework 4

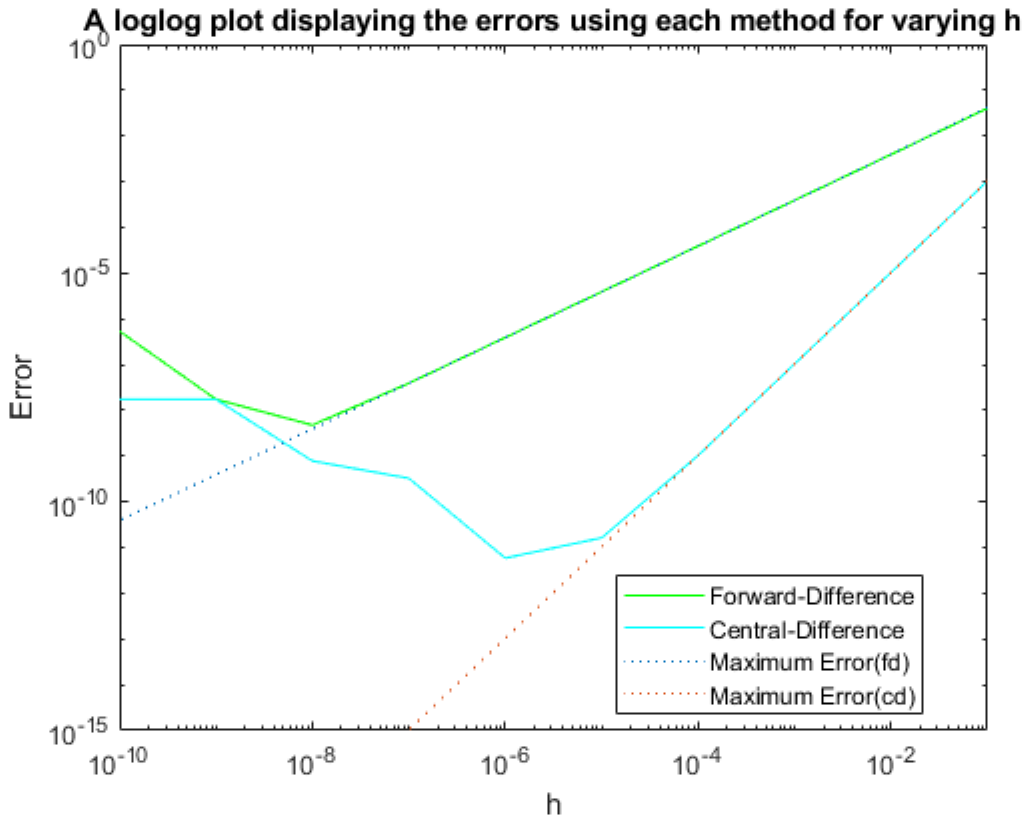
Subject: Numerical Calculus Name: Jake Denton Student ID:14322189

```
clear all
close all
clc
```

Question 2

This code plots the errors in the forward-difference and central-difference approximations against the step length h alongside lines which represent the maximum theoretical error.

```
Almost=zeros(10,2);%initialises matrix for the below for loop
H=zeros(10,1);%initialises vector
f=@(x) sin(x);%defines required function handle
x=0.9;%this is the value at which we want to approximate f'
maxerrorfd=zeros(10,1);%again initialises vector
maxerrorcd=zeros(10,1);
for i=1:10
    h=10^(-i);%defines h (which is smaller by order of 10 for each i)
    maxerrorfd(i)=h*0.5*sin(0.9+h);%this is my formula for the maximum
 error of forward-difference
    maxerrorcd(i)=h^2*(1/6)*cos(0.9);%formula for max error of
 central-difference
    Almost(i,:)=FDCD(f,x,h);%redefines a row of this matrix to the
 approximates for this loops h
    H(i)=h;%redefines element in row vector to use in loglog
end
e=cos(0.9).*ones(10,2);%defines a 10x2 matrix where all entries are
 the value we're approximating
Error=abs(e-Almost);%takes absolute value of the error matrix
loglog(H,Error(:,1),'g');%plots the forward-difference error against H
hold on
loglog(H,Error(:,2),'c');%plots central-difference error against H
loglog(H,maxerrorfd,':');%plots the max. theoretical error for each fd
 against H
loglog(H,maxerrorcd,':');%plots max. theoretical error for each cd
 against H
legend('Forward-Difference','Central-Difference','Maximum
 Error(fd)','Maximum Error(cd)','Location','Best');%rest of this code
 formats the graph
xlim([1e-10,1e-1]);
ylim([1e-15,1e0]);
title('A loglog plot displaying the errors using each method for
 varying h');
xlabel('h');
ylabel('Error');
```



Explanation For Convergence Q2

When h>10^(-8), convergence for the forward-difference approximation is linear as expected (theoretical error(also plotted) has order h). Similarly, for h>10^(-5), convergence for the central-difference is quadratic, since it has error of order h^2. The error in the central-difference is <= to that of the forward-difference for all step lengths h in this interval (as expected). For very small h, both approximations display an increase in error, so that the error is above the theoretical maximum. This is explained by round-off error, which is inversely proportional to h and therefore increases as h decreases.

Question 3

This code computes and displays single strip approximations (using Trapezium and Simpson's rules) to the integral of exp(x) where x is in (2.8,3.4). The absolute errors are then found and compared to the bound for the maximum value of the error for each rule.

```
f=@(x) exp(x);
%Trapezium Rule Single Strip Approximation
h1=0.6;%h=b-a where b=3.4,a=2.8
TR=h1*0.5*(f(2.8)+f(3.4));%calculate the approx. using definition of
 Trap. Rule with one strip
disp(['The approximate value for the integral using the Trapezium Rule
 is ',num2str(TR,10)]);%displays approx. to 10 sig figs
%Simpson Rule Single Strip Approximation
h2=0.3;%h=0.5*(b-a) where a,b as above
SR=h2*(1/3)*(f(2.8)+4*f(3.1)+f(3.4));%calculate the approx. using
 definition of Simpson's rule with one strip
disp(['The approximate value for the integral using Simpsons Rule is
 ',num2str(SR,10)]);%displays approx. to 10 sig figs
%Errors
I=f(3.4)-f(2.8);%calculates the actual value of the integral
trE=abs(I-TR);%takes absolute value of error for trapezium rule
srE=abs(I-SR);%takes absolute value of error for simpson's rule
disp(['Absolute errors for Trapezium and Simpson rules are
 ',num2str(trE,10),' and ',num2str(srE,10),' respectively']);
%Rough bounds
boundTR=(1/12)*h1^3*f(3.4);%calculates error bound for TR using
 (h^3*max(f''))/12
disp(['The theoretical bound on the error of the trapezium rule is
 ',num2str(boundTR,10)]);
disp(['Note that ',num2str(trE,10),'<',num2str(boundTR,10),' as
 required']);
boundSR=(1/90)*h2^5*f(3.4);%calculates error bound for SR using
 (h^5*max(f''''))/90
disp(['The theoretical bound on the error of the Simpson rule is
 ',num2str(boundSR,10)]);
disp(['Note that ',num2str(srE,10),'<',num2str(boundSR,10),' as
 required']);
```

```
The approximate value for the integral using the Trapezium Rule is
 13.92262405
The approximate value for the integral using Simpsons Rule is
 13.52005519
Absolute errors for Trapezium and Simpson rules are 0.4031707692 and
 0.0006019181261 respectively
The theoretical bound on the error of the trapezium rule is
 0.5393538009
Note that 0.4031707692<0.5393538009 as required
The theoretical bound on the error of the Simpson rule is
 0.0008090307013
Note that 0.0006019181261<0.0008090307013 as required
```

Question 4

This code creates a table displaying approximations of the given integral using the composite trapezium rule alongside their errors when different numbers of strips are used.

```
format long
vm=zeros(10,1);%initialises the approximation vector
M=zeros(10,1);%initialises the vector which contains number of strips
v=2543.189629746544*ones(10,1);%creates vector with the actual value
 of the integral
f=@(t) 35*(1-exp(-t*(1/256)+(t*(1/128))^2-(t*(1/64))^3));%defines the
 function handle f(t) as given
for i=0:9
    m=2^(i);%for each i defines the number of strips as required
    vm(i+1)=CoTrapRule(0,128,f,m);%calls the composite trapezium rule
 function to provide approximation for integral
    M(i+1)=m;%assigns (i+1)st element of M to the appropriate number
 of strips used in approx. above
end
Error=abs(v-vm);%calculates the vector of absolute error for the
 approximations
T=table(M,vm,Error);%creates a table with the appropriate columns
disp(T);%displays table
```

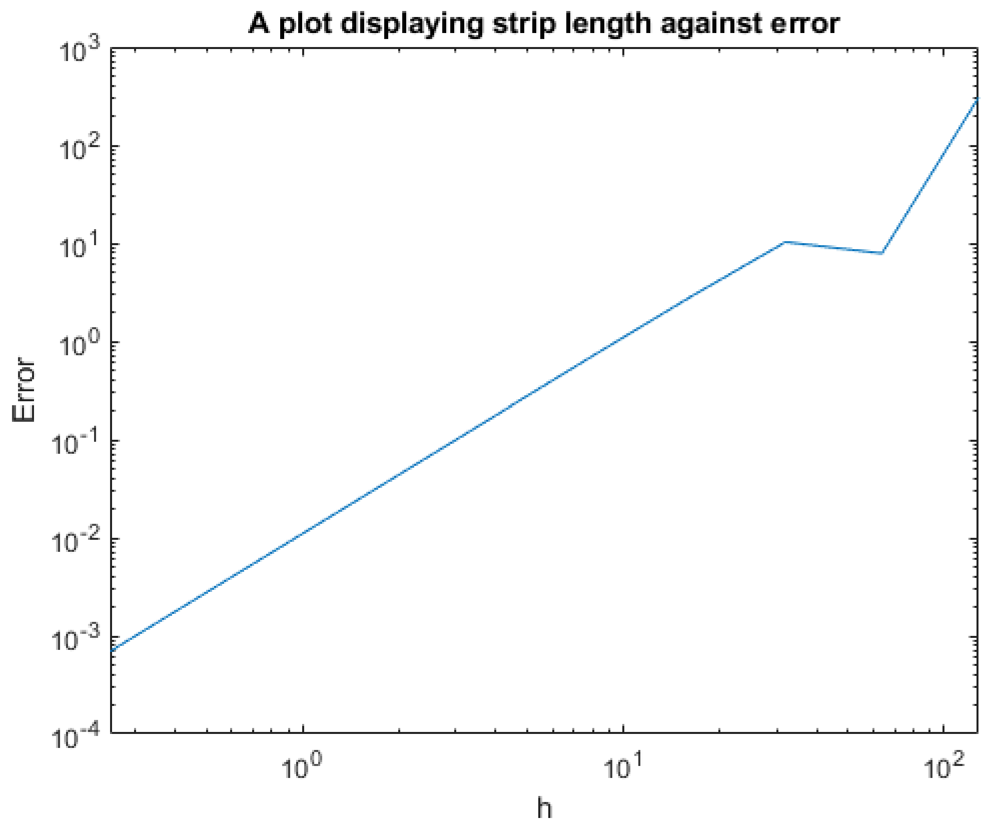
```
     M              vm                     Error
    ___    ________________    ____________________

      1    2238.76109101087        304.428538735675
      2     2535.3305972814         7.8590324651409
      4    2532.92476536656        10.2648643799844
      8    2540.41608108436         2.7735486621873
     16      2542.483025007       0.706604739539671
     32    2543.01215108231       0.177478664234968
     64    2543.14520833575      0.0444214107960761
    128    2543.17852115934      0.0111085871994874
    256    2543.18685239758     0.00277734896690163
    512    2543.18893539666    0.000694349881996459
```

Question 5

This code plots the error against strip length on double log axes.

```
close all hidden
axes
vm=zeros(10,1);%first two lines initialise vectors for the for loop
H=zeros(10,1);
v=2543.189629746544*ones(10,1);%this is actual value of integral
f=@(t) 35*(1-exp(-t*(1/256)+(t*(1/128))^2-(t*(1/64))^3));%defines
 function handle as required
for i=0:9
    m=2^(i);%for each i gives required number of strips
    vm(i+1)=CoTrapRule(0,128,f,m);%evaluates the approximation and
 assigns to element of vm
    H(i+1)=128/m;%calculates strip length and assigns to element of H
end
Error=abs(v-vm);%calculates absolute error vector
loglog(H,Error);%plots error against strip length with double log axes
title('A plot displaying strip length against error');%rest of this
 code formats the graph
xlabel('h');
ylabel('Error');
xlim([0.25,128]);
```



Rate Of Convergence Q5

The observed rate of convergence in the plot is explained by considering the order of the error for the composite trapezium rule, which is h^2. Therefore we see an error that increases steeply with h.

Question 6

This code finds the second derivative of f(t) and uses it to obtain the theoretical error bounds for varying numbers of strips m, displaying this information in a table then highlighting the value for m=32 as required.

```
syms t
f2=diff(35*(1-exp(-t*(1/256)+(t*(1/128))^2-(t*(1/64))^3)),t,2);%this
 obtains the second derivative of f(t) wrt t
```

```
t=0:640:128;%defines vector to evaluate the second derivative at
f2t=subs(f2);%substitute the vector above into the second derivative
maxf2=abs(max(f2t));%finds the absolute maximum of the second
 derivative in the interval
vpamax=vpa([maxf2]);%creates variable precision number
disp('Rough error bound is (32/3)*h^2*');
disp(vpamax);
maxerror=zeros(10,1);%next two lines initialise vectors for loop
M=zeros(10,1);
for i=0:9
    m=2^i;%defines number of strips in each iteration
    h=128/m;%defines length of strip
    M(i+1)=m;%assigns element of M with no. of strips
    maxerror(i+1)=(32/3)*h^2*vpamax;%assigns element of maxerror with
 the maximum theoretical error
end
format long
T=table(M,maxerror);
disp(T);%displays table with no. of strips/error bounds, done as extra
 to check error develops as expected
disp('Thus when m=32, the error bound is ')
disp(maxerror(6));
```

```
Rough error bound is (32/3)*h^2*
0.0048065185546875

     M          maxerror
    ___     _________________

      1                   840
      2                   210
      4                  52.5
      8                13.125
     16               3.28125
     32             0.8203125
     64           0.205078125
    128         0.05126953125
    256       0.0128173828125
    512     0.003204345703125

Thus when m=32, the error bound is
   0.820312500000000
```

Question 8

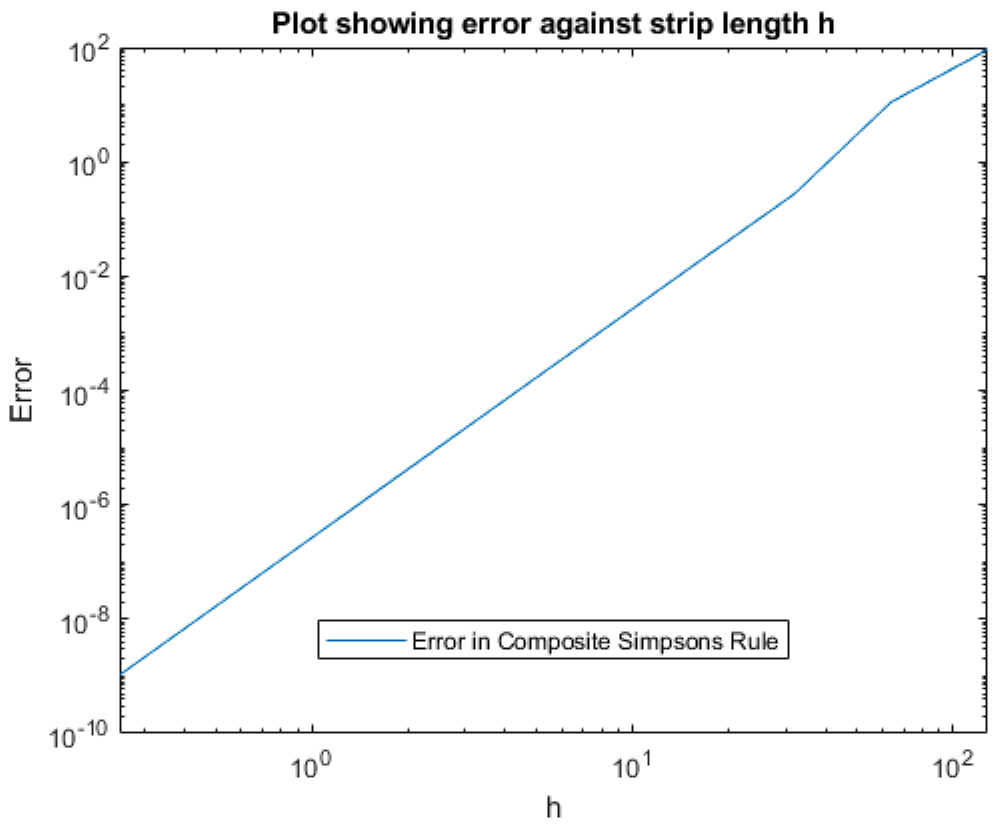
This code obtains approximations of the integral given using the Composite Simpson's Rule with varying numbers of strips, then finds the absolute error. It then displays a table with the approximations and their associated errors and produces a double log plot displaying how the error changes with different lengths of strip.

```
close all hidden
axes
```

```
format long
M=zeros(10,1);%next few lines initialise vectors for loops
vmcsr=zeros(10,1);
H=zeros(10,1);
v=2543.189629746544*ones(10,1);%define actual value of integral
f=@(t) 35*(1-exp(-t*(1/256)+(t*(1/128))^2-(t*(1/64))^3));%define
 function handle f(t)
for i=0:9
    m=2^i;%define no. of strips for this iteration
    vmcsr(i+1)=CoSimRule(0,128,f,m);%find composite simpson's rule
 approx. for this m
    M(i+1)=m;%assigns element in M to number of strips (for the table)
    H(i+1)=128/m;%assigns element in H to length of strip (for the
 curve)
end
SRError=abs(v-vmcsr);%calculates the absolute error in the
 approximation
T=table(M,vmcsr,SRError);%next two lines produce/output the required
 table
disp(T);
loglog(H,SRError);%this plots the error against strip length on double
 log axes
title('Plot showing error against strip length h');%rest of this code
 formats the graph
xlabel('h');
ylabel('Error');
xlim([0.25,128]);
legend('Error in Composite Simpsons Rule','Location','Best');
```

```
     M          vmcsr                 SRError
    ___    ________________    ____________________

      1    2634.18709937158        90.9974696250374
      2    2532.12282139495        11.0668083515989
      4    2542.91318632362       0.276443422922512
      8    2543.17200631455      0.0176234319906143
     16    2543.18852644074     0.00110330580037044
     32    2543.18956075356    6.89929843247228e-05
     64    2543.18962543388    4.31266789746587e-06
    128    2543.18962947699    2.69555130216759e-07
    256    2543.18962972969    1.68524820765015e-08
    512    2543.18962974549    1.05637809610926e-09
```



Comparing Rates Of Convergence Q8

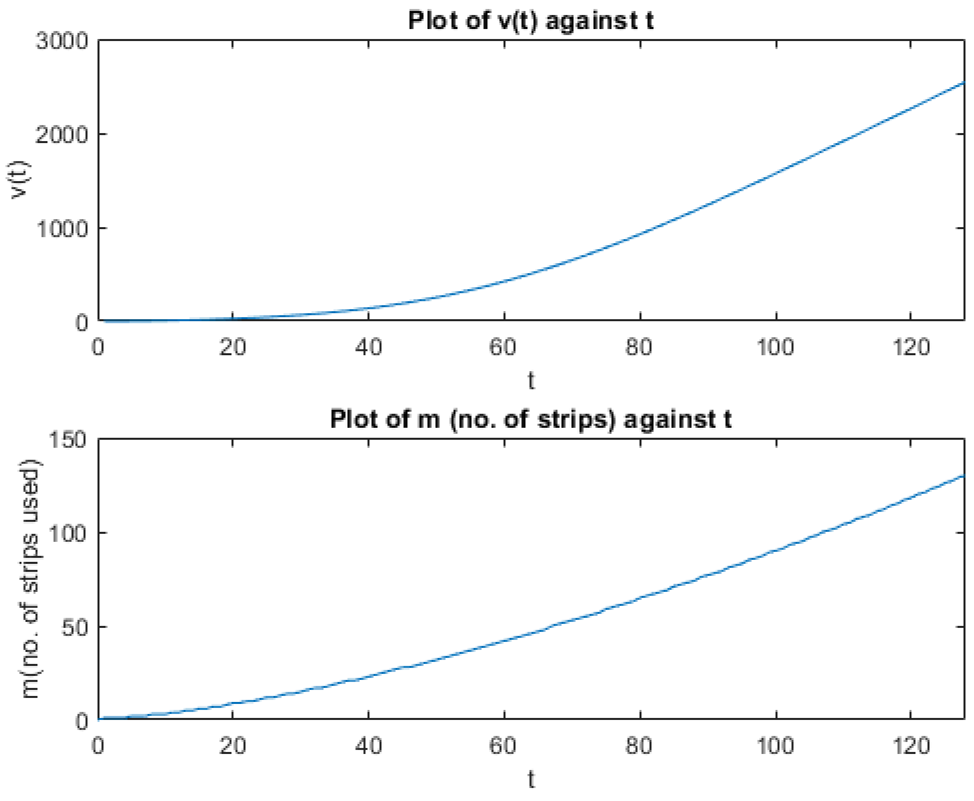
The observed rate of convergence when the Composite Simpson's Rule is used is quicker than that of the Composite Trapezium Rule as expected due to the fact that the error term for the CSR has order h^4.

Question 10

This code computes the approximations v(t) for t in [0,128] which have error less than 0.05 along with the number of strips used to obtain such an error. It then produces a figure with 2 subplots, one being v(t) against t, the second being m (number of strips) against t.

```
format long g
f=@(t) 35*(1-exp(-t*(1/256)+(t*(1/128))^2-(t*(1/64))^3));%defines
 function handle as given
d=315/65536;%this is the maximum of the second derivative of f as
 computed in Q6
T=zeros(129,1);%next two rows initialise vectors for the for loop
vtandm=zeros(129,2);
for i=0:128
    T(i+1)=i;%assigns element of T to i (for use in each plot)
    vtandm(i+1,:)=VelTrapRule(i,0.05,f,d);%reassigns each row of this
 matrix to [vt m] for this value of i
end
Velocity=vtandm(:,1);%extracts the approximations v(t) from the matrix
subplot(2,1,1);
```

```
plot(T,Velocity);%plots v(t) against t
title('Plot of v(t) against t');%next few lines format first figure
xlabel('t');
ylabel('v(t)');
xlim([0,128]);
M=vtandm(:,2);%extracts the number of strips used, m, from the matrix
subplot(2,1,2);
plot(T,M);%plots m against t
title('Plot of m (no. of strips) against t');%next few lines format
 second figure
xlabel('t');
ylabel('m(no. of strips used)');
xlim([0,128]);
```



Published with MATLAB® R2019a