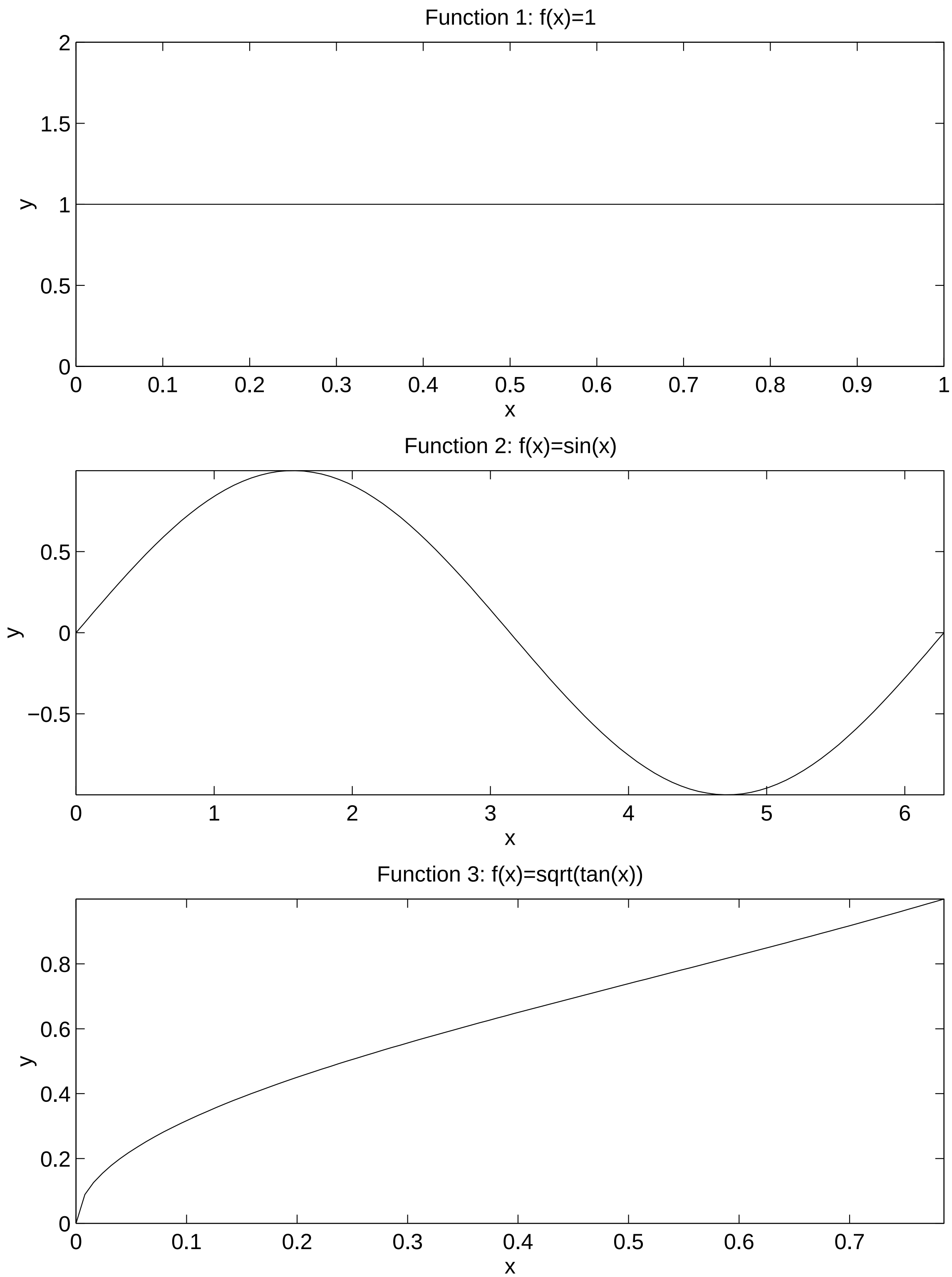
CS100b Program 6 Sample Solution

Fall 1999

1 Matlab Plot



2 M-File that plots function and performs the Trapezoidal Rule

```
%
%   Assignment #6  CS100B
%
%   Numerical Integration using MatLab
%
%   Author:                CS100B Student
%   CUID:                  123456
%   Section:               99, Sunday, 10:00pm, CS100B TA
%
%   Signature:             My Signature


%   a:  lower Integral limit
%   b:  upper Integral limit
%
%   f1: y=1
%   f2: y=sin(x)
%   f3: y=sqrt(tan(x))

%
%    INPUT
%
clear
format compact

fprintf(' Please enter the function to integrate:\n')
fprintf(' f1 for y=1\n f2 for y=sin(x)\n')
fprintf(' f3 for y=sqrt(tan(x)): ')
f_name=input(' ','s');

a=input(' Enter lower integral bound (a):');
b=input(' Enter upper integral bound (b):');

%
%    PLOT
%
clc
orient tall
x=linspace(a,b,100); y=feval(f_name,x);
plot(x,y)
if (f_name=='f1')
   axis([a b 0 2]);
else
   axis([a b -inf inf]);
end
title(['Function: ',f_name]); xlabel('x'); ylabel('y');
```

```
%
%    CHECK FOR ROOTS USING LHS-RHS METHOD
%
%    tol : tolerance, stopping criteria for lhs/rhs method
tol=sqrt(eps);
root=lhs_rhs(f_name,a,b,tol);

%
%    DETERMINE THE INTEGRAL WITH WRITTEN TRAPEZOIDAL FUNCTION
%                       AND WITH TRAPZ FUNCTION
fprintf('\n *** Trapezoidal rule for Function %2s: ***\n',f_name)
fprintf('     n    mytrapez        trapz\n')

for i=0:3
   n=10^i;
   % Use written trapezoidal rule
   if (root==a)
     I1 = abs(trapez(f_name,a,b,n));
   else
     I1a =abs(trapez(f_name,a,root,n));
     I1b =abs(trapez(f_name,root,b,n));
     I1=I1a+I1b;
   end

   % Use Matlab's trapz function
   x=linspace(a,b,n+1);y=feval(f_name,x);
   I2 = trapz(x,y);

   % Output
   fprintf(' %5.0f  %10.6f   %10.6f\n',n,I1,I2)
end

%
%    DETERMINE THE INTEGRAL WITH INT FUNCTION
%
if (f_name=='f1')
  F=int('1','x',a,b);
elseif (f_name=='f2')
  F=int('sin(x)','x',a,b);
elseif (f_name=='f3')
  F=int('sqrt(tan(x))','x',a,b);
end

I=eval(F);
fprintf('\n Solution using MatLab''s int-function: %7.6f\n',I)
```

3 Sample Program Listing for function 1, 2, and 3

```
% Function f1 to be integrated
function y=f1(x)
   y=ones(1,length(x));

% Function f2 to be integrated
function y=f2(x)
   y=sin(x);

% Function f3 to be integrated
function y=f3(x)
   y=sqrt(tan(x));
```

4 Sample Program Listing for LHS-RHS Method

```
function root=lhs_rhs(f_name,a,b,tol)
% determines the root of a function
%
% Input:  f_name: function name
%         a:      lower integral bound
%         b:      upper integral bound
%         tol:    tolerance for stopping criteria
%
% Output: root:   root of function or a if there is no root
%                 in interval [a,b]

step=0.01;
x=a;
while (feval(f_name,x+step)>tol & x<b)
  x=x+step;
end

if (abs(b-x)<=step)
  root=b;
else
  root=x;
end
```

5 Output

5.1 Function: $y = 1$

```
>> trapRR
 Please enter the function to integrate:
 f1 for y=1
 f2 for y=sin(x)
 f3 for y=sqrt(tan(x)):
 f1

 Enter lower integral bound (a):0

 Enter upper integral bound (b):1

 *** Trapezoidal rule for Function f1: ***
     n    mytrapez        trapz
     1    1.000000     1.000000
    10    1.000000     1.000000
   100    1.000000     1.000000
  1000    1.000000     1.000000

 Solution using MatLab's int-function: 1.000000
>>
```

5.2 Function: $y = sin(x)$

```
>> trapnew
 Please enter the function to integrate:
 f1 for y=1
 f2 for y=sin(x)
 f3 for y=sqrt(tan(x)):
 f2

 Enter lower integral bound (a):0

 Enter upper integral bound (b):2*pi

 *** Trapezoidal rule for Function f2: ***
     n    mytrapez        trapz
     1    0.001862    -0.000000
    10    3.967047    -0.000000
   100    3.999671     0.000000
  1000    3.999996    -0.000000

 Solution using MatLab's int-function: 0.000000
>>
```

5.3 Function: $y = tan^{1/2}(x)$

```
>> trapnew
 Please enter the function to integrate:
 f1 for y=1
 f2 for y=sin(x)
 f3 for y=sqrt(tan(x)):
 f3

 Enter lower integral bound (a):0

 Enter upper integral bound (b):pi/4

 *** Trapezoidal rule for Function f3: ***
     n    mytrapez        trapz
     1    0.392699     0.392699
    10    0.483434     0.483434
   100    0.487356     0.487356
  1000    0.487491     0.487491

 Solution using MatLab's int-function: 0.487495
>>
```

6 Questions

Increasing the values of **n** improves the accuracy. You could also say that the truncation error decreases. (However, at a very high **n** the accuracy starts getting worse, or, the rounding error increases.)