Math 375 Fall 2005
3-Functions

Vageli Coutsias

- functions
- plots
- zeros
- derivatives
- integrals

functions and plots

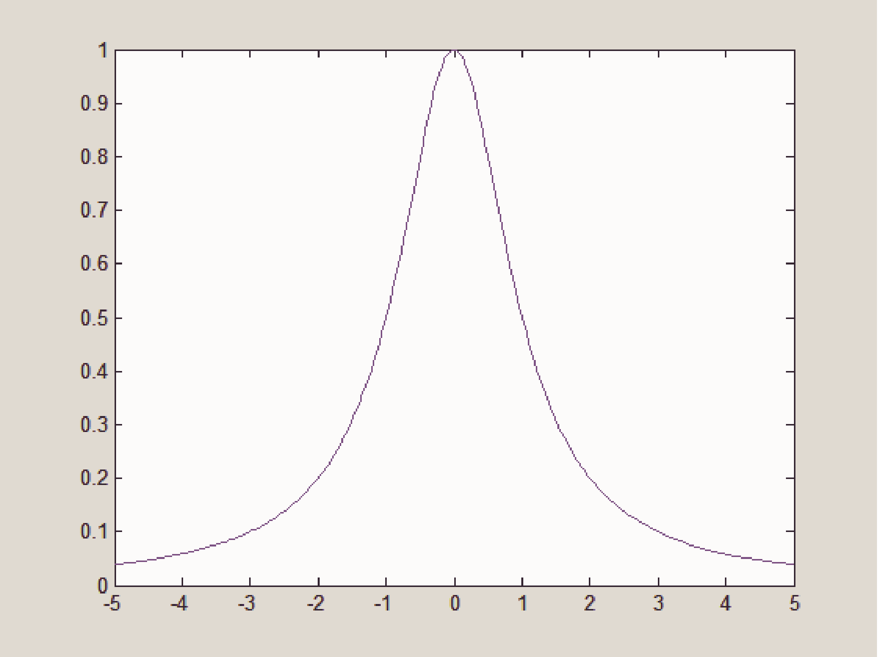
- fplot

f = '1/(1+x^2)';lims=[-5,5];fplot(f,lims);

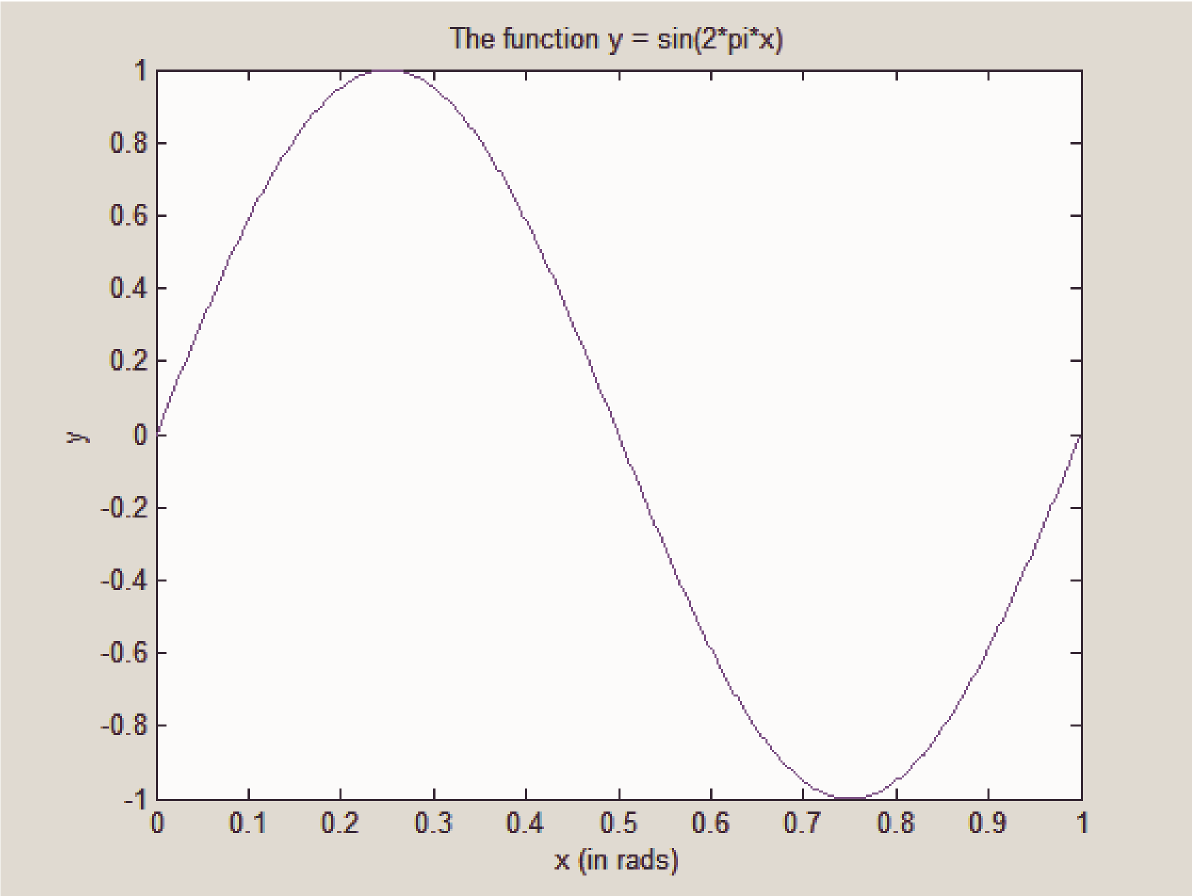
- eval

>> x=1.1;y=eval(f)

y = 0.4525



```
% Script File: SinePlot
% Displays increasingly smooth plots of sin(2*pi*x)
close all
for n =[4 8 12 16 20 50 100 200 400]
  x=linspace(0,1,n); y = sin(2*pi*x);
  plot(x,y)
  title(' The function y = sin(2*pi*x) ')
  xlabel(' x (in rads) ')
  ylabel(' y ')
  pause(1)
end
```



```
% some useful commands
%
% rand is a random number generator
% (uniform in [0.0,1.0])
% the following command resets its starting value
RAND('state',sum(100*clock))
%
% x = fzero(@fun,x0,options,a1,a2,...)
% finds the zero of the function
% fun(x,a1,a2,...) nearest x0
fun = 'x^2-x-1';
fzero(fun,1)
fzero(fun,-1)
```

```
% function schedule
% schedule(537.24,67912.5,4.98454,15)
function balance = schedule(pmt,principal,ann_rate, yrs)
    I = ann_rate/(100*12);
    months = yrs*12;
    bal(1) = principal;
    int = 0;
    disp(' ')
    disp(        '  month         balance    interest ')
    disp(        '-----------------------------------------')
    for i = 1:months
        int = int + bal(i)*I;
        bal(i+1) = bal(i)*(1+I)-pmt;
        disp(sprintf('   %3.0f         %8.2f     %6.2f',i,bal(i),int))
    end
```

```
% function mortgage
% finds payment, given yearly rate, principal and years
% (assumes interest compounded monthly!)
% mortgage(67912.5, 4.98454, 15)
function payment = mortgage(principal, rate, years)
    I = rate/(100*12);
    months = years*12;
    payment0 = principal/months;

payment = fzero(@iter,payment0,[],principal,I,months);

function balance = iter(payment,principal, I, months)
    balance = principal;
    for i = 1:months
        balance = balance*(1+I)-payment;
    end
```

```
% Script File: SinTaylor
% Plots, as a function of n, the relative error in the Taylor
% approximation x -  x^2/2! +...+ (-1)^n x^(2*n+1)/(2*n+1)! to sin(x)
nTerms = 50;
err = zeros(50);
for x=[10 5 1 -1 -5 -10]
   figure
   term = x;
   s = x;
   f = sin(x);
   for k=2:50
      term = -x^2*term/((2*k-2)*(2*k-1));
      s = s+term;
      err(k) = abs(f - s+10^(-100));
   end
   relerr = err/abs(sin(x));
   semilogy(1:nTerms,relerr)
   ylabel('Relative Error in Partial Sum.')
   xlabel('Order of Partial Sum.')
   title(sprintf('x = %5.2f',x))
end
```

Taylor series for exp(x)

$$T_n(x) = \sum_{k=0}^{n} \frac{x^n}{k!}$$

```
function y = expF(x,n)
term = 1; y = 1;
for k = 1:n
  x = x*term/k;
  y = y + term;
end
```

MATLAB Editor/Debugger - [ExpTaylor.m - C:\WINDOWS\Desktop\375\IV\ExpTaylor.m*]

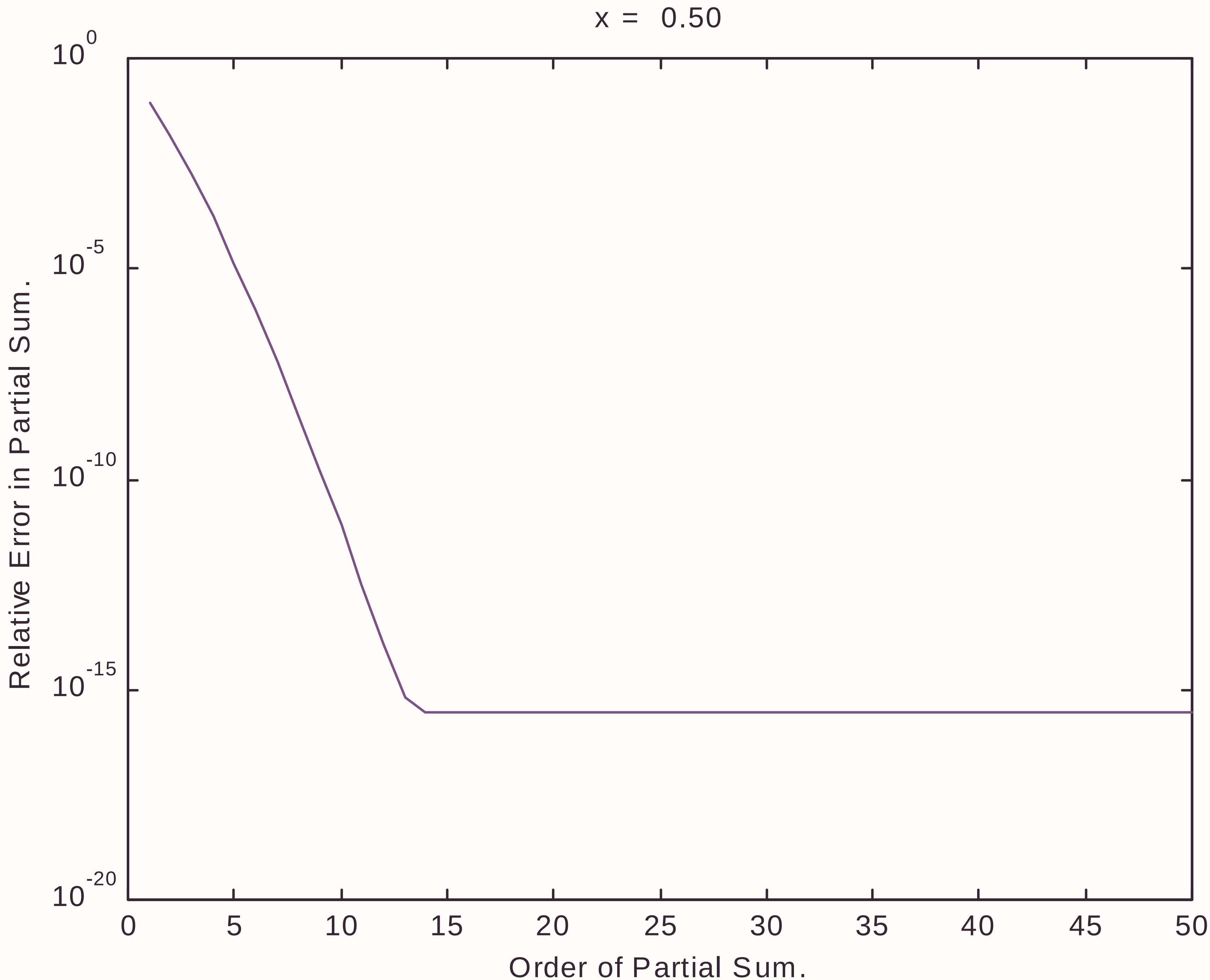
File Edit View Debug Tools Window Help

Stack:

```
% Script File: ExpTaylor
% Plots, as a function of n, the relative error in the Taylor
% approximation 1 + x + x^2/2! +...+ x^n/n! to exp(x)
close all
nTerms = 50;
%for x=[10 5 1 -1 -5 -10]
x = input('Enter x: ');
   figure
   term = 1; s = 1; f = exp(x)*ones(nTerms,1);
   for k=1:50,
      term = x*term/k;
      s = s+term;
      err(k) = abs(f(k) - s);
   end
   relerr = err/exp(x);
   semilogy(1:nTerms,relerr)
%  semilogy(1:nTerms,err)
   ylabel('Relative Error in Partial Sum.')
   xlabel('Order of Partial Sum.')
   title(sprintf('x = %5.2f',x))
   figure
   semilogy(1:nTerms,err)
%end
```

monster1.m... | monster2.m... | monster4.m... | Stirling.m - ... | experr.m - ... | ExpTaylor.m ...

Ready Line 2 12:55 AM


x = 0.50
Relative Error in Partial Sum.
Order of Partial Sum.
10^{0}
10^{-5}
10^{-10}
10^{-15}
10^{-20}
0
5
10
15
20
25
30
35
40
45
50