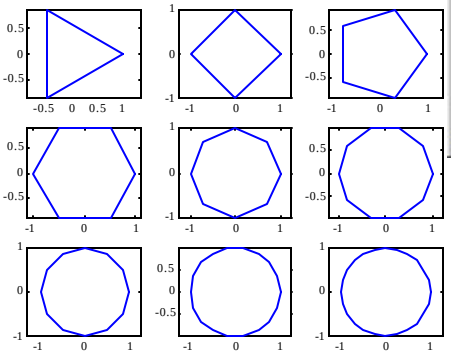
Math.375
III- Doing the math

Vageli Coutsias

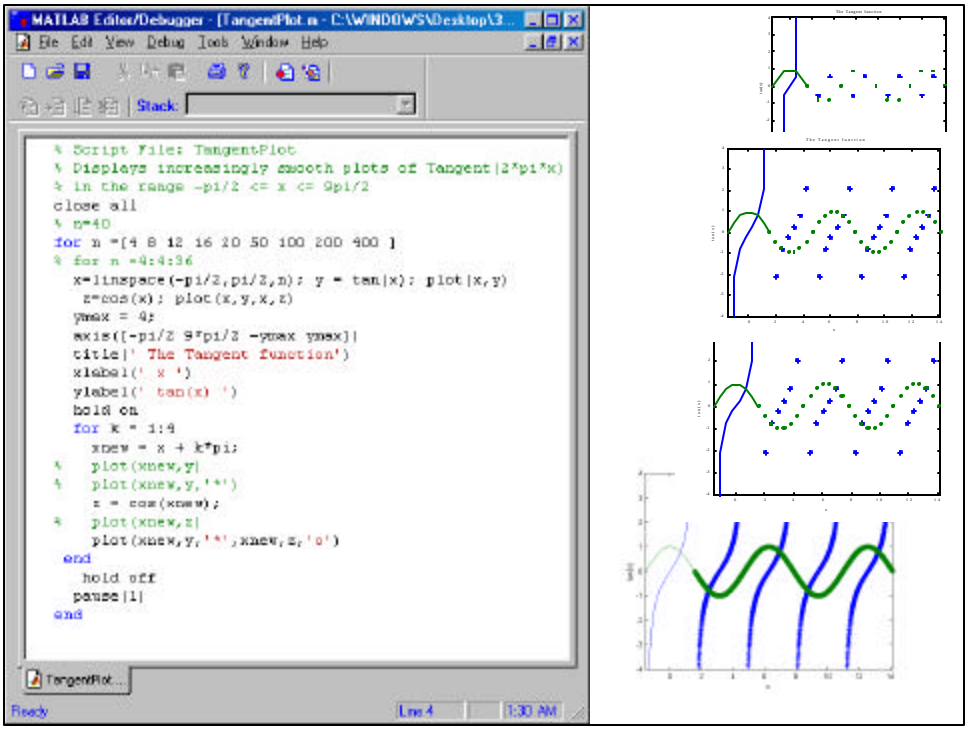
Some formulas, concepts and tricks

- Finding machine constants
- Numerical Differentiation: a taste
- Arrays and cells
- Pade Approximation
- Sorting
- More series

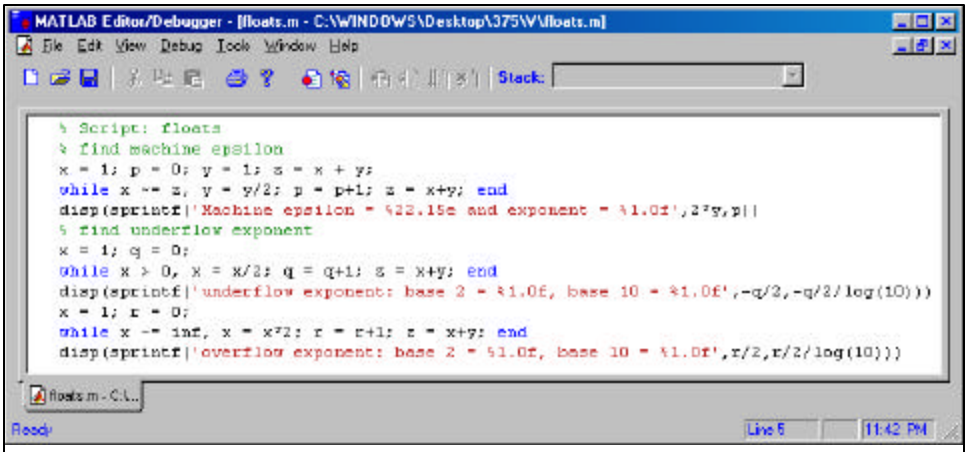
subplots



```
% Script File: Polygons
% Plots selected regular polygons
close all
theta = linspace(0,2*pi,361);
c=cos(theta); s = sin(theta);
k=0;
for sides = [3 4 5 6 8 10 12 18 24]
   stride = 360/sides;
   k = k+1;
   subplot(3,3,k)
   plot(c(1:stride:361),s(1:stride:361))
   axis equal
end
```

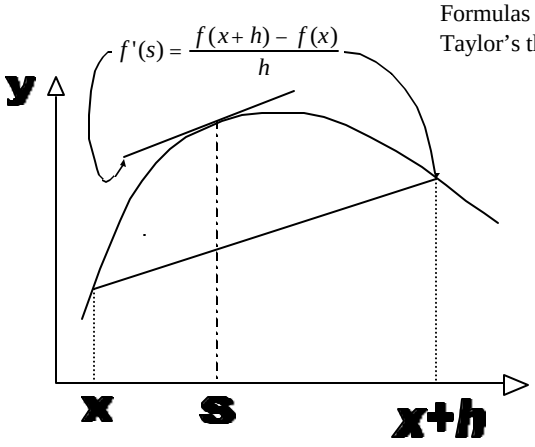


```
% Script File: TangentPlot
% Displays increasingly smooth plots of Tangent(2*pi*x)
% in the range -pi/2 <= x <= 9pi/2
close all
% n=40
for n =[4 8 12 16 20 50 100 200 400 ]
% for n =4:4:36
  x=linspace(-pi/2,pi/2,n); y = tan(x); plot(x,y)
   z=cos(x); plot(x,y,x,z)
  ymax = 4;
  axis([-pi/2 9*pi/2 -ymax ymax])
  title(' The Tangent function')
  xlabel(' x ')
  ylabel(' tan(x) ')
  hold on
  for k = 1:9
    xnew = x + k*pi;
%   plot(xnew,y)
%   plot(xnew,y,'*')
    z = cos(xnew);
%   plot(xnew,z)
    plot(xnew,y,'*',xnew,z,'o')
 end
   hold off
  pause(1)
end
```



» floats
Machine epsilon = 2.220446049250313e-016 and exponent = 53
underflow exponent: base 2 = -538, base 10 = -233
overflow exponent: base 2 = 512, base 10 = 222

The mean value theorem can be used to draw conclusions about the relation between Divided Differences and Derivatives. Similar Formulas for higher derivatives can be found Taylor's theorem.



$$f(x\pm h) = f(x) \pm f'(s_1)h, s_1 \in [x, x\pm h]$$

$$f(x\pm h) = f(x) \pm f'(x)h + \frac{1}{2}f''(s_2)h^2, s_2 \in [x, x\pm h]$$

$$f(x\pm h) = f(x) \pm f'(x)h + \frac{1}{2}f''(x)h^2 \pm \frac{1}{3!}f'''(s_3)h^3, s_3 \in [x, x\pm h]$$

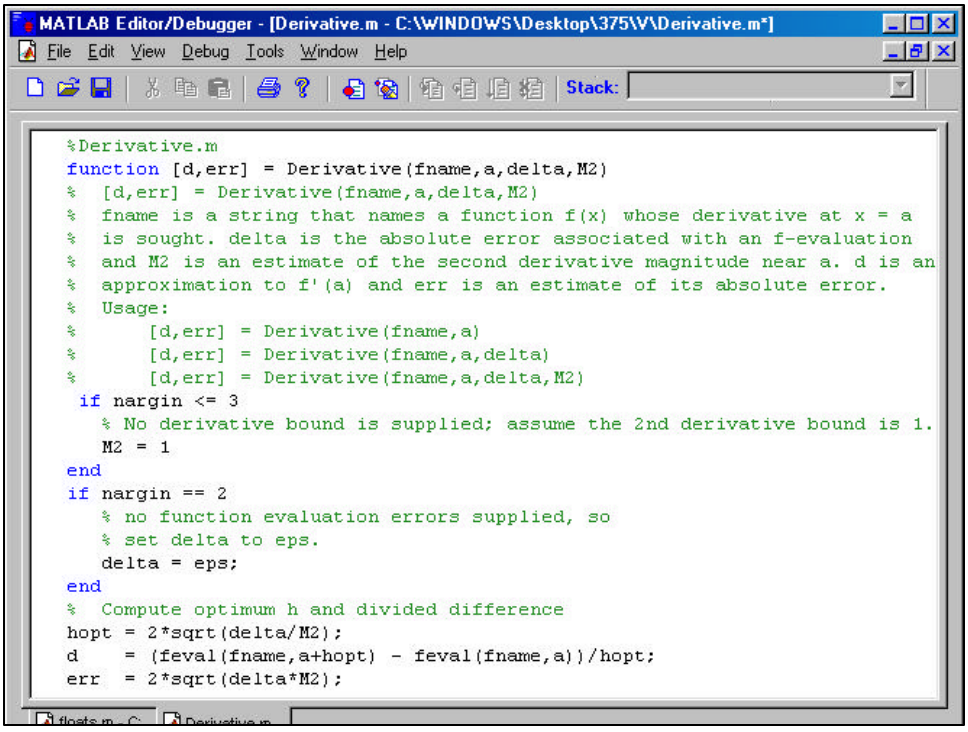
$$\longrightarrow$$

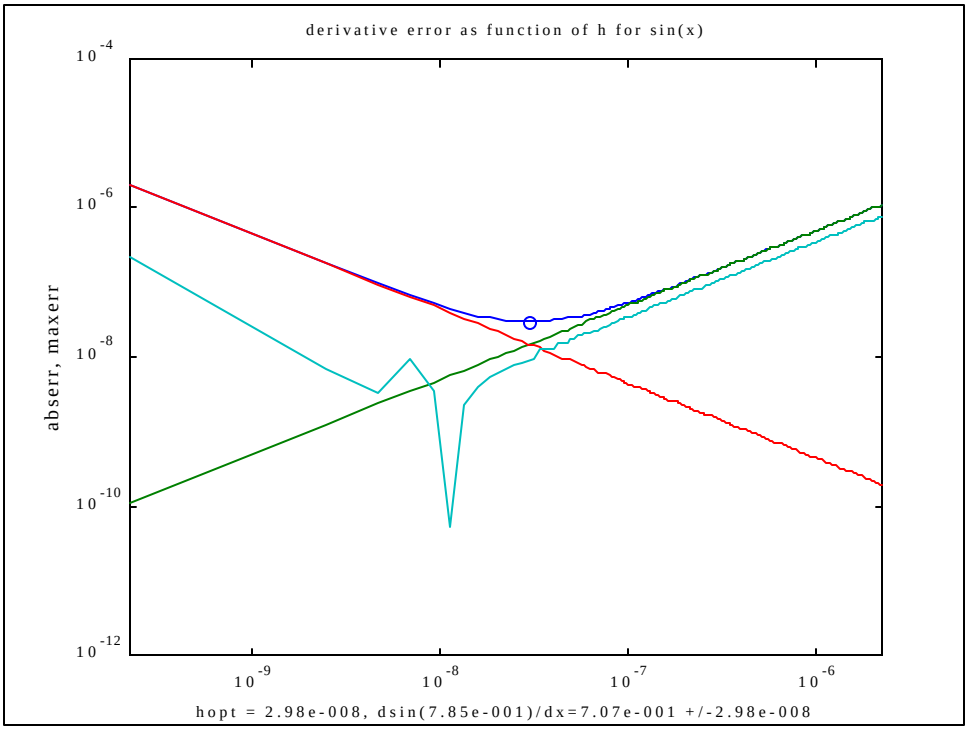
$$D_h = \frac{f(x+h)-f(x)}{h} = f'(s_1) = f'(x) + f''(s_2)\frac{h}{2} = f'(x) + f''(x)\frac{h}{2} + f'''(s_3)\frac{h^2}{3!}$$

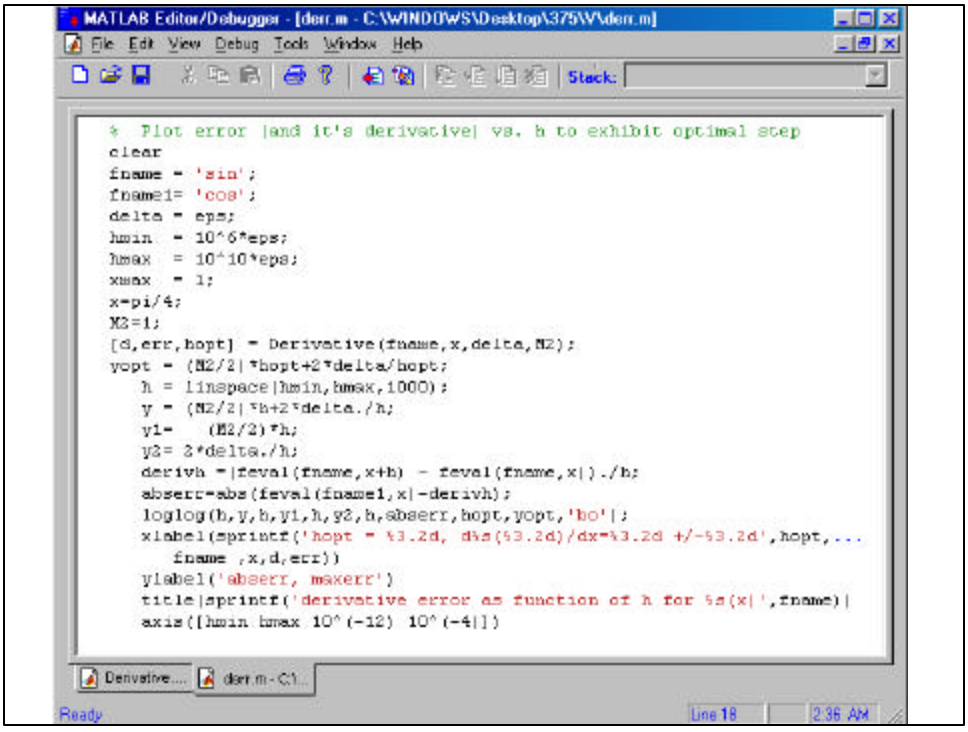
But, due to roundoff:

$$|\hat{D}_h - f'(x)| = \left|\frac{f(x+h) + eps_1 - (f(x) + eps_2)}{h} - f'(x)\right| = \left|D_h - f'(x) + \frac{eps_1 - eps_2}{h}\right|$$

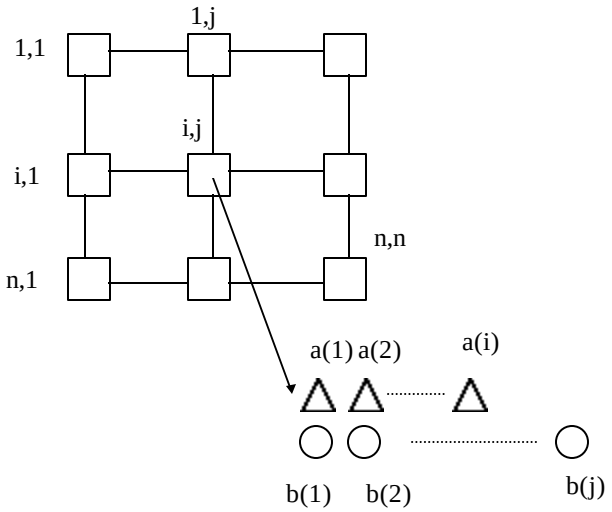
$$\le |D_h - f'(x)| + \frac{2*eps}{h} = \frac{h}{2}|f''(s_2)| + \frac{2*eps}{h}$$




derivative error as function of h for sin(x)
abserr, maxerr
10^-4
10^-6
10^-8
10^-10
10^-12
10^-9
10^-8
10^-7
10^-6
hopt = 2.98e-008, dsin(7.85e-001)/dx=7.07e-001 +/-2.98e-008


MATLAB Editor/Debugger - [derr.m - C:\WINDOWS\Desktop\375\V\derr.m]
File Edit View Debug Tools Window Help
Stack:
% Plot error |and it's derivative| vs. h to exhibit optimal step
clear
fname = 'sin';
fname1= 'cos';
delta = eps;
hmin = 10^6*eps;
hmax = 10^10*eps;
xmax = 1;
x=pi/4;
N2=1;
[d,err,hopt] = Derivative(fname,x,delta,N2);
yopt = (N2/2|*hopt+2*delta/hopt;
h = linspace|hmin,hmax,1000);
y = (N2/2|*h+2*delta./h;
y1= (N2/2)*h;
y2= 2*delta./h;
derivh =|feval(fname,x+h) - feval(fname,x|)./h;
abserr=abs(feval(fname1,x|-derivh);
loglog(h,y,h,y1,h,y2,h,abserr,hopt,yopt,'ho'|;
xlabel(sprintf('hopt = %3.2d, d%s(%3.2d)/dx=%3.2d +/-%3.2d',hopt,...
fname ,x,d,err))
ylabel('abserr, maxerr')
title|sprintf('derivative error as function of h for %s(x|',fname)|
axis([hmin hmax 10^(-12) 10^(-4|])
Derivative....
derr.m - C:\...
Ready
Line 18
2:36 AM

THE PADE TABLE



A CELL array: it's elements are STRUCTURES
(here sets of arrays)

Evaluation of the fraction

$$R_{pq}(z) = \frac{\sum_{k=0}^{p} \frac{(p+q-k)!\,p!}{(p+q)!k!(p-k)!} z^k}{\sum_{k=0}^{q} \frac{(p+q-k)!q!}{(p+q)!k!(q-k)!} (-z)^k}$$

$$p, q \to \infty, R_{pq}(z) \to e^z$$

$$e^z \approx \frac{1 + \frac{1}{2}z}{1 - \frac{1}{2}z}$$

Possible project: Pade approximant for the exponential series

$$\frac{dx}{dt} = Ax \Rightarrow x = e^{At}x_0$$

$$e^{At} = I + tA + \frac{t^2}{2}A^2 + \cdots \approx \frac{Q_p(A)}{R_q(A)} = \frac{a_0 + a_1 tA + .. + a_p t^p A^p}{b_0 + b_1 tA + .. + b_q t^q A^q}$$

Can use for highly accurate approximations to the solution of a system of ODEs with constant coefficients; "division" here means multiplication by the inverse of the denominator (notation is unambiguous since numerator and denominator COMMUTE!

$$a_1 = a_0 \frac{p}{(p+q)} \qquad a_k = a_{k-1} \frac{p-k+1}{k(p+q-k+1)}$$

$$a_k = a_0 \frac{p(p-1)\cdots(p-k+1)}{(p+q)(p+q-1)\cdots(p+q-k+1)1\cdot 2\cdots k} = a_0 \frac{(p-k)_k}{k!(p+q-k)_k}$$

$$b_1 = -b_0 \frac{q}{(p+q)} \qquad b_k = -b_{k-1} \frac{q-k+1}{k(p+q-k+1)}$$

$$b_k = (-1)^k b_0 \frac{q(q-1)\cdots(q-k+1)}{(p+q)(p+q-1)\cdots(p+q-k+1)1\cdot 2\cdots k} = (-1)^k b_0 \frac{(q-k)_k}{k!(p+q-k)_k}$$

$$R_{i,j} = \left\{a^{(i)}, b^{(j)}\right\} = \left\{\left[a_0^i, \ldots, a_i^i\right], \left[b_0^j, \ldots, b_j^j\right]\right\}$$

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

File Edit View Debug Tools Window Help

Stack:

```
  function P = PadeArray(m,n)
% P = PadeArray(m,n)
% m and n are nonnegative integers.
% P is an (m+1)-by-(n+1) cell array.
% P{i,j} represents the (i-1,j-1) Pade approximation N(x)/D(x) to exp(x)
P = cell(m+1,n+1);
for i=1:m+1
   for j=1:n+1
      P{i,j} = PadeCoeff(i-1,j-1);
   end
end
  function R = PadeCoeff(p,q)
% R = PadeCoeff(p,q)
% p and q are nonnegative integers and R is a representation of the
% (p,q)-Pade approximation N(x)/D(x) to exp(x):
% R.num  is a row (p+1)-vector whose entries are the coefficients of the
%           p-degree numerator polynomial N(x).
% R.den  is a row (q+1)-vector whose entries are the coefficients of the
%           q-degree denominator polynomial D(x).
% Thus,
%                  R.num(1) + R.num(2)x + R.num(3)x^2
%                -------------------------------------
%                        R.den(1) + R.den(2)x
%  is the (2,1) Pade approximation.
 a(1) = 1; for k=1:p, a(k+1) =  a(k)*(p-k+1)/(k*(p+q-k+1)); end
 b(1) = 1; for k=1:q, b(k+1) = -b(k)*(q-k+1)/(k*(p+q-k+1)); end
 R = struct('num',a,'den',b);
```

Testing the Pade approximant: compute the error

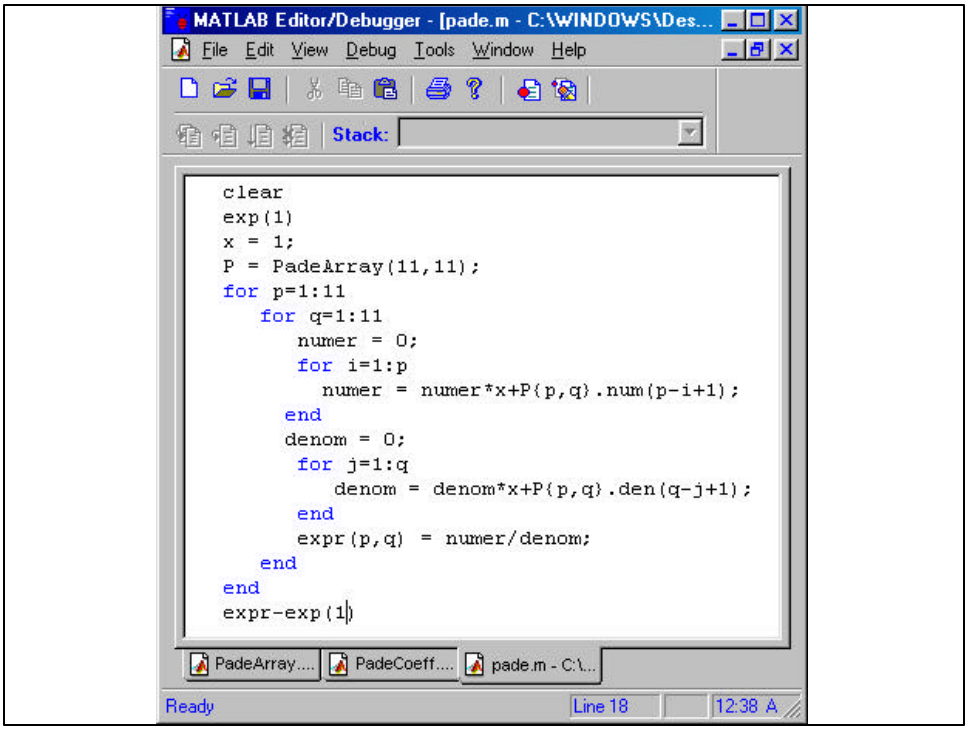
$$\varepsilon(t) = |\, e^t - P\{p,q\}(x) \,| = |\, e^t - \frac{a_0^p + a_1^p t + \cdots + a_p^p t^p}{b_0^q + b_1^q t + \cdots + b_q^q t^q} \,|$$

Columns 1 through 4

-1.71828182845905	Inf	0.71828182845905	0.28171817154095
-0.71828182845905	0.28171817154095	-0.05161516179238	0.00899089881368
-0.21828182845905	0.03171817154095	-0.00399611417333	0.00046817154095
-0.05161516179238	0.00394039376318	-0.00033311051033	0.00002803069588
-0.00994849512571	0.00046817154095	-0.00002786020508	0.00000175363051
-0.00161516179238	0.00005150487429	-0.00000225856657	0.00000010986695
-0.00022627290349	0.00000520857799	-0.00000017471227	0.00000000674695
-0.00002786020508	0.00000048446612	-0.00000001280881	0.00000000040153

Columns 5 through 8

-0.05161516179238	0.00899089881368	-0.00130069638357	0.00016477348270
-0.00130069638357	0.00016477348270	-0.00001849669878	0.00000186543440
-0.00004978426015	0.00000482722464	-0.00000042890571	0.00000003511782
-0.00000225856657	0.00000017210609	-0.00000001235433	0.00000000083512
-0.00000011017733	0.00000000674695	-0.00000000039834	0.00000000002256
-0.00000000551510	0.00000000027665	-0.00000000001364	0.00000000000066
-0.00000000027623	0.00000000001154	-0.00000000000048	0.00000000000002
-0.00000000001364	0.00000000000048	-0.00000000000002	0.00000000000000




```
----------------------------------------
I.
A=[1 2 3;4 5 6;7 8 9];
y=x'; %transpose
x=zeros(1,n);
length(x);
x=linspace(a,b,n);
a=x(5:-1:2);
%SineTable  (Help SineTable)
disp(sprintf('%2.0f  %3.0f ',k, a(k)));

----------------------------------------
```

```
---------------------------------------------------------------
II.
VECTOR OPERATIONS:

(1) vector:  scale, add, subtract (scalar*vector)
    2*[10 20 30] = [20 40 60]
(2) POINTWISE vector: multiply, divide, exponentiate (pointwise vector * vector)
    [2 3 4].*[10 20 30] = [20 60 120]
(for pointwise operations, dimensions must be identical!)
(3) Setting ranges for axes,
    Superimposing plots:
plot(x,y,x,exp(x),'o')
  axis([-40 0 0 0.1])
     using "hold on/off"
     extended parameter lists
(4) Subplots: subplot(m,n,k)
(5) Vector linear combos: matrix*vector
(6) Tensor Products (x is array, A is array of arrays of same shape as x)
       A=[f1(x) f2(x) ... fn(x)]
    A(k,:) k-th row; A(:,i) i-th column
(7) loops: loop until ~condition or for fixed no. of times
    while any(abs(term) > eps*abs(y))
    for term = 1:n
    end
(8) size(A) gives array dimensions (array, can be used to index other arrays)
    length(x) gives vector dimension
---------------------------------------------------------------
```

```
-----------------------------------------------------------------
II' (Miscellaneous topics)
rem(x,n): remainder
  if x == 1
(< <= == >= > ~=)
(& and; | or; ~ not; xor exclusive or)
  else
  endif
s3 = [s1 s2]; concatenation of strings, treated like arrays
[xmax, imax] = max(x); (max val, index)
rat = max(x)/x(1)
x(1) = input('Enter initial positive integer:')
density = sum(x<=x(1))/x(1);
```

Summary

- Finding machine constants
- Multiple plots
- Error estimates and differentiation
- Review of Matlab commands
- Using Cell arrays
- Examples of mathematical calculations:
  The Pade approximant to the exponential
(application to numerical solution of ODEs)

References

- find m-files at
  aix: ~coutsias/375/7
- C.F.van Loan,
  Intro to Sci.Comp. (1.4, 1.6)