Themes:

Boundary and Initial value problems for ODE. If something left from ODE övning, do that.
Lab 2 has a boundary value problem, difference method, with Dirichlet boundary conditions so we do one with Robin conditions
Use numerical differencing to compute Jacobian matrix

a) linear problem
b) non-linear so use Newton

Task

write the discretized differential equation and boundary conditions as
r(**u**) = 0 (residual = 0)
If problem linear, **r** = **Au –b**,
so **b** = -**r**(0)
A computed by differencing **r** (= Jacobian matrix)
then **u = A \ b**;

Problem to solve:

$$\frac{d^2w}{dx^2} = b$$

$$w(0) = 0, \frac{dw}{dx}(1) = -cw(1)$$

a model for a catenary (overhead wire for electric train), supported at x = 0 and by a spring (c) at x = 1 and acted on by gravity (b)

w1 … wN , h = 1/(N-1), w1 at x = 0
Equations:
x = 0: w1 - 0 = 0
x = h, 2h, …,
(wi+1 – 2wi + wi-1)/h2 -1 = 0, i = 2,…,N-1
x = 1: (wN-wN-1)/h+c wN = 0

this probably gives first order error: in approx.. to w' at x = 1: slow convergence

N		
11	0.2250	-0.2250
21	0.2375	-0.2375
41	0.2438	-0.2438
81	0.2469	-0.2469
161	0.2484	-0.2484

Check the order of accuracy!

Then try second order formula for w' in boundary cond., x = 1: (3/2wN -2wN-1 + 0.5wN-2)/h+c wN

```
   N      w'
   11    0.2500   -0.2500
   21    0.2500   -0.2500
   41    0.2500   -0.2500
   81    0.2500   -0.2500
  161    0.2500   -0.2500
```

`what is going on here? ans. exact answer!`

```
% run catenary
global B C
B = 1; C = 1;
for N =[11,21,41,81,161];
% linear:
    w = zeros(N,1);
    b = -caten(w);
    A = jacobian('caten',w,1e-4);
    w = A\b;
    plot(w,'.')
    F = (w(end)-w(end-1))*(N-1);
    F = (1.5*w(N)-2*w(N-1)+0.5*w(N-2))*(N-1);
    disp([N F C*w(N)])
end

function res = caten(w)
global B C
N = length(w); h = 1/(N-1); res = 0*w;
res(1)    = w(1)-0;
res(2:N-1)= diff(diff(w))/(h*h)-B;
res(N)    = (w(N)-w(N-1))/h+C*w(N);
res(N)    = (1.5*w(N)-2*w(N-1)+0.5*w(N-2))/h+C*w(N);

function J = jacobian(F,x,steg)
n = length(x);
J = zeros(n,n);
f = feval(F,x);
for j=1:length(x)
   xplus = x;
   if x(j)==0
      xplus(j) = steg;
   else
      xplus(j) = (1+steg)*x(j);
   end
   J(:,j)=(feval(F,xplus)-f)/(xplus(j)-x(j));
end
```

Problem 2, exam 24/10 2009

2. *Släpptest för pumpor*
Halloweenpumpor genomgår fallprov. Vid tiden $t = 0$ släpps de från en meters höjd, $y = 1$, mot en elastisk testmatta. Efter exakt 3.2 sekunder ska pumpan ha studsat

tillbaka till $y = 0$. Pumpans rörelse bestäms av differentialekvationen
$y'' = 4\cos 2\pi t - 3y - ty'$
med $y(0) = 1$, $y(3.2) = 0$.
Figuren visar lösningen $y(t)$ då finitadifferens-metoden utnyttjas med dels åtta dels sexton delintervall.
(9p) a) Härled för fallet åtta delintervall det ekvationssystem som uppstår för beräkning av y-värdena. Hur många ekvationer erhålls?
(7p) b) Skriv ett matlabprogram som beräknar och skriver ut figurens fallprovskurvor.

so now: a non.linear problem

$$\frac{1}{(1+w'^2)^{1/2}} w'' = b$$
$$w'(0) = 0, w(1) = 1,$$

the shape of a chain, hanging from two pegs at the same height. Only half due to symmetry.

```
function res = chain(w)
global B
N = length(w); h = 1/(N-1); res = 0*w;
res(1)    = w(2)-w(1);
wp        = (w(3:end)-w(1:end-2))/(2*h);
res(2:N-1)= diff(diff(w))/(h*h)-B*(1+wp.^2).^(1/2);
res(N)    = w(N)-1;

% run chain, non-linear:
clear all, close all
figure(1), clf
global B
N = 41;
for B = linspace(0.5,5,20)
    w = linspace(0,1,N)';
    w = NewRaph('chain',w,1e-5);
    plot(w,'.')
    hold on
    disp([B w(1)])
    pause
end

function x = NewRaph(F,x,tol)
h  = 2*tol*norm(x);
steg = 1e-5;
iter = 0; itmax = 50;
while norm(h) > tol*norm(x)
   f = feval(F,x);
   J = jacobian(F,x,steg);
   h = J\f;
   x = x-h;
   disp([x(1:min(3,end))' norm(h)])
   iter = iter + 1;
   if iter > itmax
       break
   end
```

```
end
if iter > itmax
    x = NaN;
end
```

problem 4 a & b, 23/10 2010

(7)P4.a Problemet nedan diskretiseras med steglängden h = 0.25 och centraldifferens-approximationer till derivatorna.

$$(1+z'^2)z'' = z$$
$$z(0) = 1, z(2) = 1$$

Diskretisera intervallet (hur många inre punkter blir det?) och inför lämpliga beteckningar samt formulera det icke-linjära ekvationssystem **f**(**z**) = 0 som erhålls.

(8)P4.b Skriv ett Matlabprogram som för steglängd $h = 2/(N + 1)$, med valfritt N, formulerar och lös er det ickelinjära ekvationssystemet med Newtons metod. Bifogad rutin `minjac` får användas. Skriv ut mellanresultat så iterationernas konvergens kan studeras och diskutera förväntade utskrifter. Beskriv hur du gör för att bedöma diskretiserings-felet i lösningen?

Not: `minjac` och ovan använda `jacobian` är ekvivalenta

Användbara Matlabrutiner:

```
-----------------------------------------------------------------
function jac=minjac(Fcn,z);
%beräknar en numerisk approximation jac till
%jakobianmatrisen till funktionen Fcn i punkten z
N=length(z); F=feval(Fcn,z); jac=[]; stegtol=1.E-8;
for i=1:N,
   z0=z;
   st=z0(i)*stegtol;
   if st==0,
      st=1.E-10;
   end
   z0(i)=z0(i)+st;
   jac=[jac ( feval(Fcn,z0)-F )/st];
end
-----------------------------------------------------------------
```