Themes: What is left?

Next lecture is devoted to review of the whole course by going over some old exam paper. On the ovning, we do only problems from part 2 of exams. If something left from övn 5, do that; Otherwise, as follows (in some order you choose).

Problem 3 from 2011-12-17

3. En variant av Van der Pol-generatorn beskrivs av den ordinära differentialekvationen

$$x'' - (1 - x^2)x' + \sin x = 0$$

(a) Skriv en detaljerad algoritm (gärna i MATLAB) som beräknar lösningen x vid t = 5 för begynnelsedata $x(0) = 1$ och $x'(0) = 0$. Algoritmen ska vara baserad på Framåt Euler-metoden med steglängden h = 0.01. (10 p)

(b) Modifiera din algoritm så att den istället använder den implicita Bakåt Euler-metoden,

$$\mathbf{u}_{n+1} = \mathbf{u}_n + h\boldsymbol{f}(t_{n+1}, \mathbf{u}_{n+1}).$$

Nya element kan behöva introduceras i algoritmen. (6 p)

OK, just to remind students of the phase plane etc. The classical vdp oscillator with + x instead of + sin(x) has a limit cycle solution : for any initial data, the solution approaches a periodic orbit. Same here for small x because x & sin x are close. But the sinx equation has many critical points $x = k\pi$, k = +-1,+-2,… which are asymptotically stable whereas 0 is unstable

```
% vdpsin ex
y = [3.0;0];
[tout,yout]=ode45('vdpsin',[0 30],y);
plot(yout(:,1),yout(:,2))

function dydt = vdpsin(t,y)
dydt = [y(2);(1-y(1)^2)*y(2)-sin(y(1))];
```

The explicit Euler is simple; we do only the implicit scheme. (this was a hard question, note the exam was for the engrg physics students only)

```
% euler backward
clear all
close all
y = [1;0];
h = 0.1;
t = 0; tend = 10;
n = round(tend/h);
tol = 1e-5
ytab = y;
for k= 1:n
    z = y;
    corr = 2*tol;
    t  = t+h;
    %it = 0;
    while norm(corr)> tol
        % it = it + 1;
        % solve z-h*f(z) = y;
        res = z-h*vdpsin(t+h,z)-y;
        J = [0                          , 1      ;...
            -2*z(1)*z(2)-cos(z(1)), 1-z(1)^2];
        corr = (eye(2)-h*J)\res;
        z    = z-corr;
    end
    % disp([t,norm(corr),it,z'])
    y = z;
    ytab = [ytab,y];
end
plot(ytab(1,:),ytab(2,:))
```

you can uncomment to see how many iterations are needed, etc.

Problem 2 from 2011-12-17

2. Givet funktionen $y(x)$.
(a) Bestäm ett tredjegradspolynom som interpolerar $y(x)$ i punkterna $x = 0, 1, 2, 3$. Redovisa det linjära ekvationssystem som erhålles och specificera hur polynomet är relaterat till systemets lösning. (Inga beräkningar behöver genomföras.) (5 p)
(b) Bestäm en (styckvis polynom-) funktion $S(x)$ med egenskapen
i. $S(x)$ interpolerar $y(x)$ i punkterna $x = 0, 1, 2, 3$,
ii. $S(x)$ är ett förstagradspolynom i intervallet [0, 1],
iii. $S(x)$ är ett tredjegradspolynom i intervallet [1, 2],
iv. $S(x)$ är ett förstagradspolynom i intervallet [2, 3],
v. $S'(x)$ kontinuerlig på hela intervallet [0, 3].
Redovisa det linjära ekvationssystem som erhålles och specificera hur $S(x)$ är relaterat till systemets lösning. (Inga beräkningar behöver genomföras.) (5 p)

a) Several solutions possible to represent the polynomial:
The standard monomial basis,
”centered version”,
Newton’s way

Show the standard & Newton & point out that polyval (if you want to use that) requires the coefficients in descending order

b) The coefficients to determine are, for the three intervals
a1,b1; a2,b2,c2,d2; a3,b3 - 8 unknowns
P(x) = a + bx + cx^2 + dx^3
so (8 equations!)

```
x = 0: a1                          = y(0)
x = 1: a1 + b1                     = y(1),
       a2 + b2 + c2 + d2           = y(1)
       b1 – (b2 + 2*c2 + 3*d2)     = 0 derivative cont.
x = 2: a3 + b3*2                   = y(2)
       a2 + b2*2 + c2*4 + d2*8     = y(2)
       b3 - (b2 + 2*c2*2 + 3*d2*4) = 0 derivative continuity
x = 3: a3 + b3*3                   = y(3)
```

and we can be sure that the solution is unique because
a1, b1 are determined from eqn 1 and 2, a3 and b3 from 5 and 8,
and the rest is a Hermite interpolation problem which we know
have a unique solution.

EXS 4.28

A minimization problem. Solve by Newton on grad $f = 0$, $f(\mathbf{x}) = \sum_{j=1}^{n} \left\| \mathbf{x} - \mathbf{X}_j \right\|$.

Show that there is a problem when all the points are collinear (Jacobian singular, non-unique solution) Then change problem to

$$\min_{\mathbf{x}} \phi(\mathbf{x}), \phi(x) = \sum_{j=1}^{n} \left\| \mathbf{x} - \mathbf{X}_j \right\|^2$$

which is a least squares problem. Compute the gradient and Hessian of ϕ. Show that the same (linear) problem appears as normal equations for the overdetermined system

$$\mathbf{x} = \mathbf{X}_j,\ j = 1,2,\ldots,N$$

and that the solution is that **x** be the center of gravity of the set of points {**X**j}

$$f(\mathbf{x}) = \sum_{j=1}^{n} \left\| \mathbf{x} - \mathbf{X}_j \right\| = \sum_{j=1}^{n} r_j,, r_k = \sqrt{(x - X_k)^2 + (y - Y_k)^2}$$

$$\frac{dr_k}{dx} = \frac{x - X_k}{r_k}, \frac{dr_k}{dy} = \frac{y - X_k}{r_k}$$

$$\frac{d^2 r_k}{dx^2} = \frac{1}{r_k} - \frac{x - X_k}{r_k^2} \cdot \frac{x - X_k}{r_k} = \frac{(y - X_k)^2}{r_k^3}$$

$$\frac{d^2 r_k}{dy^2} = \frac{(x - X_k)^2}{r_k^3}, \frac{d^2 r_k}{dxdy} = -\frac{(x - X_k)(y - Y_k)}{r_k^3}$$

so we get

$$grad = \sum_{j=1}^{n} \frac{1}{r_j} \mathbf{dx}_j, \mathbf{dx}_j = \begin{pmatrix} x - X_j \\ y - Y_j \end{pmatrix}$$

$$Hessian = \sum_{j=1}^{n} \frac{1}{r_j^3} \mathbf{H}_j, \mathbf{H}_j = \begin{pmatrix} (y - X_k)^2 & -(x - X_k)(y - Y_k) \\ -(x - X_k)(y - Y_k) & (x - X_k)^2 \end{pmatrix} = \mathbf{z}_j \cdot \mathbf{z}_j{}^T, \mathbf{z}_j = \begin{pmatrix} x - X_j \\ -(y - Y_j) \end{pmatrix}$$

So we see that the Hessian is a sum of rank-1 matrices. Now if all Xj are on a line and so is x, they are all multiples of the same rank-1 matrix hence their sum is singular. But if not on a line, no problem for any **x**. For matlab, use the trusted NewRaph which uses the jacobian() function to compute Jacobian matrices (in this case the Hessian)

Let students compute gradient & Hessian of the least squares problem. Note that the Hessians become just unit matrices and the gradients are just the **dx of the above.**

```
% the point problem
clear all
close all
figure(1)
global X Y
n = 1000;
X = rand(1,n)*20;
Y = rand(1,n)*20;
x = [mean(X);  mean(Y)];
x = NewRaph('pointprob',x,1e-10);
plot(X,Y,'ok');
```

```
hold on
plot(x(1),x(2),'*r','markersize',20)

function f = pointprob(z)
global X Y
x = z(1); y = z(2);
dx = x-X; dy = y-Y;
r = sqrt(dx.^2+dy.^2);
f = [sum(dx./r);sum(dy./r)];
```

And finally, a few problems on sensitivity analysis or error estimation. Show both estimates obtained by differentiation and by perturbation computation.

EXS 8.7

Geometry:

$$h = \frac{l}{\cot\alpha - \cot\beta}, dh = \frac{dl}{\cot\alpha - \cot\beta} + \frac{l}{(\cot\alpha - \cot\beta)^2}\cdot\frac{d\alpha}{\sin^2\alpha} - \frac{l}{(\cot\alpha - \cot\beta)^2}\cdot\frac{d\beta}{\sin^2\beta} =$$

$$= h\frac{dl}{l} + \frac{l}{(\cot\alpha - \cot\beta)^2}\cdot(\frac{d\alpha}{\sin^2\alpha} - \frac{d\beta}{\sin^2\beta}) = h\frac{dl}{l} + \frac{h^2}{l}\cdot(\frac{d\alpha}{\sin^2\alpha} - \frac{d\beta}{\sin^2\beta})$$

$$\left|\frac{dh}{h}\right| \le \left|\frac{dl}{l}\right| + \frac{h}{l}(\frac{|d\alpha|}{\sin^2\alpha} + \frac{|d\beta|}{\sin^2\beta})$$

so h = 60.11, sinα = 0.81, sinβ = 0.54,

dh/60 < 0.2/50+60/50 (0.3 π/180/0.66 + 0.3 π /180 /0.29)

= 0.2/50+60/50 (1/0.66 + 1 /0.29)/180 = 0.004 +4.96/150 = 0.004+0.033=0.037

dh < 2.3. Angle β contributes most to error.

NOTE angles must be in radians so we know how to differentiate! Ask students what is dsind(x)/dx (sind is matlab function for sine with angle in degrees)!

Show matlab code from EXS answers. The systematic variation +/- is done by a triple loop. This seems hard-coded and how would we do 10 variables?

Matlab coding exercise:

For n variables, how to perturb each variable can be coded as an n-bit binary number (0 = -, 1 = +) so how to generate an n x 2^x table with +1 and -1 .

```
function t = pmgen(n)
t = [-1 1];
for k = 2:n
    b = ones(size(t));
    t = [t,t;-b,b];
end
>>pmgen(3)
ans =
    -1     1    -1     1    -1     1    -1     1
    -1    -1     1     1    -1    -1     1     1
    -1    -1    -1    -1     1     1     1     1
    -1    -1    -1    -1     1     1     1     1
```

so now can code the perturbation as a single loop

```
t = pmgen(n);
var0 = [ xx,yy,zz,…,ww]';
fmax = -1e6; fmin = +1e6;
for k = 1:2^n
    var=var0+delta* t(:,k);
% compute for variable values var
    f = fun(var)
% record max & min
    fmax = max(fmax,f); fmin = min(fmin,f);
end
```

EXS 8.9

This is about cancellation and how NOT to compute exp(-x) for positive x. 1/exp(x) is so much better. Here is the loop

```
close all
clear all
x   = -20;
tol = 1e-6*exp(x);
S = 1; term = x; k = 1;
while 1
    S = S + term(end);
    term = [term,x*term(end)/(k+1)];
    k = k+1;
    if abs(term(end))< tol
        break
    end
end
error = abs(exp(x)-S);
plot(log10(abs(term)),'.k');
title([num2str(x),' error ',num2str(error)])
ylabel('10log|term|'); xlabel('term number')
```

Point out how big the largest term is; show that we can see that the largest term must have number |x|, look at how the next term is formed from the previous.