NADA, KTH

2D1240 Numeriska Metoder gk 2
Måndagen den 18/12 2006 kl 14-17

Skrivtid 3 tim. **Maximal poäng** 35 + bonuspäng från labbarna. Maxpoäng för uppgifterna anges inom parantes bredvid uppgiftsnumret.

Nedre gräns för betyg **G**: 20 poäng.

Del 1: 10 poäng. **Skrivtid högst 1 timme. Inga hjälpmedel.**

När du lämnar in del 1 får du del 2, och får ta fram rosa formelsamlingen.

(2) T1. Ange naiva ansatsen och Newtons ansats för ett interpolerande polynom som går genom $(-3, -27), (5, -1), (10, 0), (17, -20)$.

(2) T2. Beskriv trapetsregeln för integralberäkning med formel och figur.

VÄND!

(2) T3. Bestäm minstakvadratlösningen till

$$c_1 + c_2 = 10, \quad c_1 = 16, \quad c_1 - c_2 = 5$$

(2) T4. Visa i en figur hur Newton-Raphsons metod fungerar. Inför därefter lämpliga beteckningar och formulera Newton-Raphssons metod.

(1) T5a. Runge-Kuttas metod (RK4) är bättre än Eulers metod, fjärde noggrannhetsordningen jämfört med första, säger man. Vad menar man med det?

(1) T5b. Inför lämpliga beteckningar och formulera Eulers metod.

NADA, KTH

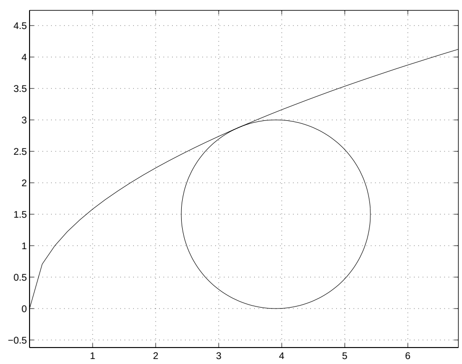
2D1240 Numeriska metoder gk 2, 4 p
Måndagen den 18/12 2006 kl 14-17

Hjälpmedel är rosa formelsamlingen som får tas fram när del 1 har lämnats in.
Två användbara **Matlabrutiner** finns på sista sidan.
Del 2: 25 poäng.

(6) P1. Följande icke-linjära ekvationssystem bestämmer centrumpunkten $(x_c, y_c = R)$ och tangeringspunkten (x_p, y_p) för en cirkel med radien $R = 2$, som tangerar både x-axeln, och parabeln $x = 0.4y^2$.

$$(x_p - x_c)^2 + (y_p - R)^2 = R^2$$
$$x_p = 0.4y_p^2$$
$$2(x_p - x_c) + 2(y_p - R)\frac{1}{0.8y_p} = 0$$

Formulera en algoritm i Matlab för problemet. Skriv programmet så R lätt kan ändras. Bifogad matlabrutin får gärna användas. Då cirkeln bestämts skall den ritas upp tillsammans överdelen av parabeln, se figuren (för annat R-värde).



(5) P2. Derivatan av funktionen $f(x)$ i punkten $x = 1.2$ har beräknats approximativt med en differensmetod med olika steglängder h. Följande resultat erhölls för ett testproblem där $f'(1.2) = 2.0042$.

h	approx.
5.00e-02	2.0717
2.50e-02	2.0209
1.25e-02	2.0084
6.25e-03	2.0053

Bestäm, från tabellen, noggrannhetsordningen för den differensmetod som använts. Motivering krävs för full poäng.

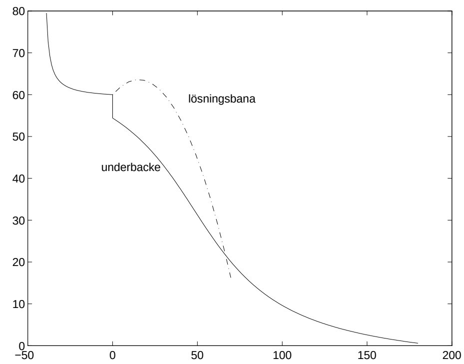
VÄND!

(6) **P3.** Givet randvärdesproblemet

$$\frac{d^2y}{dx^2} = \frac{\sin(0.4\pi x)}{1+y^2}, \qquad y(2) = 1; \quad y(4) = 3;$$

Diskretisera problemet med differensmetod. Inför lämpliga beteckningar och dela in området 4 lika delintervall, så vi får 3 inre punkter. Formulera ett ekvationssystem för den approximativa lösningen. Ni skall inte lösa ekvationssystemet, och ni skall inte programmera.

P4. Vi skall studera backhoppning och använder följande modell. Backhopparens bana $(x(t), y(t))$ efter uthoppet från överbacken ges av differentialekvationerna



$$m\frac{d^2x}{dt^2} = -C\frac{dx}{dt}w$$

$$m\frac{d^2y}{dt^2} = -mg - C\frac{dy}{dt}w$$

med $w = \sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2}$

Uthoppsvinkeln ϕ och uthoppshastigheten w_0 kan översättas till värden för hastigheterna dx/dt och dy/dt i uthoppsögonblicket ($t = 0$).

(5) **P4a.** Skriv ett Matlabprogram som beräknar och ritar upp backhopparens bana enligt bifogade figur med följande parametervärden och startvärden: $m = 80$, $C = 0.08$, $g = 9.81$, $x = 0$, $y = 60$, $\phi = 25°$, $w_0 = 20$. Beräkningarna skall avbrytas då y blir noll. Nedslagspunkten behöver inte bestämmas noga. Programmet skall använda Runge-Kuttas metod av ordningen 4. Bifogad matlabrutin får gärna användas.

(3) **P4b.** Underbackens höjd ges av

$$U(x) = 32.5 - 25\arctan(x/40 - 1.2)$$

Modifiera ditt program så att beräkningarna avbryts då hopparen slår ner i underbacken i stället. Nedslagspunkten behöver inte bestämmas noga. Backhopparens bana och underbacken skall nu ritas upp.

Matlabrutiner:

```
function [xut,yut]=RKsteg(F,x,y,h)
% beräknar ett steg med steglängden h från (x,y) till (xut,yut)
% med Runge-Kuttas metod av  ordning 4 för
% differentialekvationssystemet  y'=F(x,y)
 k1=h*feval(F,x,y);
 k2=h*feval(F,x+h/2,y+k1/2);
 k3=h*feval(F,x+h/2,y+k2/2);
 k4=h*feval(F,x+h,y+k3);
yut=y+(k1+2*k2+2*k3+k4)/6;
xut=x+h;
```

```
function jac=numjac(Fcn,z);
%beräknar en numerisk approximation jac till jakobianmatrisen till
%funktionen Fcn i punkten z
  NR=length(z); F=feval(Fcn,z); jac=[]; stegtol=1.E-8;
  for i=1:NR,
   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
```
