Uppsala universitet
Institutionen för informationsteknologi

Lösningsförslag Tentamen
Beräkningsvetenskap I och KF 5.0 hp, 2020-01-13

Del A

1. (a) Simpsons metod med $h = 0.25$:

$$S(h = 0.25) = \frac{0.25}{3} \left(f(1) + 4 \cdot f(1.25) + 2 \cdot f(1.5) + 4 \cdot f(1.75) + f(2)\right) \approx 0.393468.$$

För feluppskattning behövs även beräkning med dubbel steglängd:

$$S(h = 0.5) = \frac{0.5}{3}(f(1) + 4 \cdot f(1.5) + f(2)) \approx 0.39250646.$$

Diskretiseringsfelet kan då uppskattas med

$$E = \frac{S(h = 0.25 - S(h = 0.5)}{15} = 0.000064.$$

Integralens värde blir alltså 0.393468 ± 0.000064

(b) Diskretiseringsfelet för Simpsons metod för en steglängd h är $\mathcal{O}(h^4)$. Minskning av felet med en faktor 16 ger $\frac{1}{16}\mathcal{O}(h^4) = \mathcal{O}((\frac{h}{2})^4) = \mathcal{O}(h_1^4)$, dvs vår nya steglängd h_1 blir $h_1 = h/2 = 0.125$ (åtta intervall istället för fyra).
Obs. För godkänt resultat krävs någon typ av analys (som ovan). Korrekt svar, men utan analys ger underkänd uppgift.

2. (a) Det fel som dominerar då (i) $h > 10^{-3}$ är diskretiseringsfelet (eller trunkeringsfelet) och då (ii) $h < 10^{-3}$ är avrundningsfelet (även funktionsfelet är korrekt här).

(b) Noggrannhetsordning anger på vilket sätt diskretiseringsfelet avtar med minskande steglängd h. I figuren så kan man se noggrannhetsordningen som lutningen på grafen då $h > 10^{-3}$.

3. Ansatsen $t = a + b \cdot (y - \hat{y})$, med $\hat{y} = 1985$, och datavärdena ger

$$\begin{cases} a + b \cdot (1960 - 1985) = 5.3 \\ a + b \cdot (1970 - 1985) = 4.1 \\ a + b \cdot (1980 - 1985) = 4.6 \\ a + b \cdot (1990 - 1985) = 7.0 \\ a + b \cdot (2000 - 1985) = 7.3 \\ a + b \cdot (2010 - 1985) = 4.8 \end{cases} \Rightarrow \begin{pmatrix} 1 & -25 \\ 1 & -15 \\ 1 & -5 \\ 1 & 5 \\ 1 & 15 \\ 1 & 25 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 5.3 \\ 4.1 \\ 4.6 \\ 7.0 \\ 7.3 \\ 4.8 \end{pmatrix}$$

Bilda normalekvationerna $A^T Ax = A^T b$ ger

$$\begin{pmatrix} 6 & 0 \\ 0 & 1750 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 33.1 \\ 47.5 \end{pmatrix}$$

Ur detta fås $a = 33.1/6 = 5.5167$ och $b = 47.5/1750 = 0.02714$. Polynomet blir alltså $t = 5.5167 + 0.02714 \cdot (y - 1985)$.

4. För Newton-Raphsons metod gäller kvadratisk konvergens (i närheten av lösningen), vilket medför att felet i iteration $k \approx$ kvadraten på felet i iteration $k - 1$. Felen kan uppskattas med $e_k = |x_k - x_{k-1}|$.
För den vänstra metoden fås felen
$e_3 = |x_3 - x_2| = |1.9818160 - 2.2971761| = 0.3153601$
$e_4 = |x_4 - x_3| = |1.9359357 - 1.9818160| = 0.0458803$
$e_5 = |x_5 - x_4| = |1.9337545 - 1.9359357| = 0.0021812$
$e_6 = |x_6 - x_5| = |1.9337544 - 1.9337545| = 0.0000001$
I varje steg fördubblas antalet korrekta decimaler, dvs felet är ungefär kvadraten på felet i föregående steg (i sista fallet är det t o m bättre än kvadratiskt), vilket medför att detta bör vara Newton-Raphson.

Om man gör samma sak med högra kolumnen får man
$e_3 = 0.3153601$
$e_4 = 0.0022003$
$e_5 = 0.0008091$
$e_6 = 0.0002969$
vilket inte är kvadratisk konvergens.
Alternativt kan man titta på talsekvensen och se att antalet korrekta decimaler ungefär fördubblas vid varje iteration i den vänstra sekvensen. Detta innebär kvadratisk konvergens.

Obs. För godkänt resultat krävs någon typ av analys (som ovan). Korrekt svar, men utan analys ger underkänd uppgift.

5. (a) `v = [1; -1; 0; 2; -3]` $=> v = \begin{pmatrix} 1 \\ -1 \\ 0 \\ 2 \\ -3 \end{pmatrix}$, `a = 1`

```
equal = 0, smaller = 0, greater= 0, k = 1
len_vec = 5
k <= len_vec dvs 1 <= 5 är sant => går in i while-loopen
  if-sats:
    vec(1) = 1 > a är falskt
    vec(1) = 1 < a är falskt
    går in i else-grenen (alla andra fall) av if
    equal = 0 + 1 = 1
  slut på if
 k = 1 + 1 = 2

k <= len_vec, dvs 2 <= 5 är sant => går in while-loop
  if-sats:
     vec(2) = -1 > a är falskt
     vec(2) = -1 < a är sant
       smaller = 0 + 1= 1
   slut på if
 k = 2 + 1 = 3

k <= len_vec dvs 3 <= 5 är sant => går in i while-loopen
  if-sats:
    vec(3) = 0 > a är falskt
    vec(3) = 0 < a är sant
     smaller = 1 + 1 = 2
  slut på if
 k = 3 + 1 = 4

k <= len_vec dvs 4 <= 5 är sant => går in i while-loopen
  if-sats:
    vec(4) = 2 > a är sant
     greater = 0 + 1 = 1
  slut på if
 k = 4 + 1 = 5

k <= len_vec dvs 5 <= 5 är sant => går in i while-loopen
```

```
 if-sats:
   vec(5) = -3 > a är falskt
   vec(5) = -3 < a är sant
     smaller = 2 + 1 = 3
 slut på if
k = 5 + 1 = 6

k <= len_vec dvs 6 <= 5  är falskt => while-loop avslutas
```

Slutresultatet blir `s = 3`, `g = 1`, `e = 1`.

(b) Använd ”anonym” funktion:

```
f = @(x) 1./(1 + 0.5*x.^3);
I = integral(@(x) f(x),1, 2); alternativt I = integral(@f, 1, 2);
```

eller skriv en Matlabfunktion i en egen m-fil:

```
function fx = f(x)
fx = 1./(1 + 0.5*x.^3);
end
```

och sedan anropa `integral` på samma sätt som ovan.
OBS, det är inte kritiskt att Matlabsyntaxen är korrekt i alla detaljer, t ex punkter och semikolon.

Del B

6. Formulera om $y = \sqrt[3]{a} \Rightarrow y^3 = a \Rightarrow y^3 - a = 0$. Detta är ett icke-linjärt problem som kan lösas med Newton-Raphsons (lämpligen). Bisektionsmetoden är mindre lämplig eftersom den konvergerar så långsamt. Formulera N-R: $f(y) = y^3 - a$ och $f'(y) = 3y^2$. Obs att y^3 kan lösas som $y \cdot y \cdot y$, dvs genom att använda vanlig multiplikation (motsvarande för y^2).
Test för $a = 3$:
N-R: $y_{k+1} = y_k - \frac{y_k^3 - 3}{3y_k^2}$, startgissning t ex $y_0 = a/2 = 1.5$.

$y_1 = y_0 - \frac{y_0^3-3}{3y_0^2} = 1.5 - \frac{1.5^3-3}{3 \cdot 1.5^2} = 1.44444$
Fel: $e_1 = |y_1 - y_0| = 0.0555 > 0.5 \cdot 10^{-2}$
$y_2 = y_1 - \frac{y_1^3-3}{3y_1^2} = 1.4422529$
Fel: $e_2 = |y_2 - y_1| = 0.0021 < 0.5 \cdot 10^{-2}$

Toleransen uppfylld, $y \approx 1.44225 \pm 0.0021$.

7.
```
%=====================================================
% Lösning Uppgift 7, tenta 2020-01-13
% Numeriska fel:
% - Fel i varje y-värde i N-R, begränsas av tol
% - Felfortplantning i dydx och i sqrt(1+dydx^2)
% - Funktionsfel i integralen pga ovan
% - Diskretiseringsfel i Simpson, beror av h^4
% - Representationsfel i datorn (försumbart)
%=====================================================

% Parameterar
clear;
a=0; b=1; tol=0.5e-6;
n=100; h=(b-a)/n;
x=a:h:b;
y=zeros(size(x));
fplot(@(y)f(y,a),[0 3]); grid on;
xlabel('y'); ylabel('f(y,x), x=0');
disp('Tryck return för att fortsätta!'); pause

% Hitta y-värden
y0=1.5; % startgissning från plot
for i=1:length(x)
    y(i)=NewtonRaphson(@(y)f(y,x(i)),@(y)fprim(y,x(i)),y0,tol);
    y0=y(i); % Nästa startgissning
end

% Beräkna integralen
fvec=sqrt(1+dydx(x,y).^2);
I=simpson(fvec,h);
disp(['Båglängden blir ' num2str(I)]);
plot(x,y,'o-');
xlabel('x'); ylabel('y');
```

```
%==================================================
% Hjälpfunktioner
%==================================================
function z=f(y,x)
z=y*sin(x)-x^3-cos(y);
end

function z=fprim(y,x)
z=sin(x)+sin(y);
end

function yout=NewtonRaphson(f,fprim,y0,tol)
err=tol+1;
yny=y0;
while err>tol
    y=yny;
    yny=y-f(y)/fprim(y);
    err=abs(yny-y);
end
yout=yny;
end

function yout=dydx(x,y)
% Implicit derivering (alt numerisk differens)
yout=(3*x.^2-y.*cos(x))./(sin(x)+sin(y));
end

function I=simpson(f,h)
I=h/3*(f(1)+4*sum(f(2:2:end-1))+2*sum(f(3:2:end-2))+f(end));
end
```