Tentamen, del 2

SF1524

Grundläggande numeriska metoder och programmering

Fredag 13 mars 2020 kl 8.00-11.00

Rättas endast om del 1 är godkänd. Max antal poäng på denna del är 50. Betygsgräns: 10p D, 20p C, 30p B, 40p A. Svar skall motiveras och uträkningar redovisas. Korrekt svar utan motivering eller med felaktig motivering medför poängavdrag.

Inga hjälpmedel är tillåtna (ej heller miniräknare).

P1. Följande datapunkter är givna:

t_i	0	1	2	4
y_i	e	$e^{\frac{3}{2}}$	$e^{\frac{9}{4}}$	$e^{\frac{15}{4}}$

(7p) **a)** Datapunkterna antas passa väl till en kurva på formen $y(t) = Ce^{kt}$. Bestäm C och k med hjälp av minsta kvadratmetoden.

Vi logaritmerar båda sidor av ekvationen för $y(t)$

$$y(t) = Ce^{kt} \iff \log(y(t)) = \log(C) + k\,t$$

för att få ett linjärt ekvationssystem $A\mathbf{c} = \mathbf{b}$ i variablerna $\log(C)$ och k givet datapunkterna i tabellen:

$$\begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \\ 1 & 4 \end{bmatrix} \begin{bmatrix} \log(C) \\ k \end{bmatrix} = \begin{bmatrix} 1 \\ 3/2 \\ 9/4 \\ 15/4 \end{bmatrix}.$$

Normalekvationerna $A^TA\mathbf{c} = A^T\mathbf{b}$ till detta ekvationssystem är

$$\begin{bmatrix}1 & 1 & 1 & 1\\ 0 & 1 & 2 & 4\end{bmatrix}\begin{bmatrix}1 & 0\\ 1 & 1\\ 1 & 2\\ 1 & 4\end{bmatrix}\begin{bmatrix}\log(C)\\ k\end{bmatrix} = \begin{bmatrix}1 & 1 & 1 & 1\\ 0 & 1 & 2 & 4\end{bmatrix}\begin{bmatrix}1\\ 3/2\\ 9/4\\ 15/4\end{bmatrix}$$

$$\iff \begin{bmatrix}4 & 7\\ 7 & 21\end{bmatrix}\begin{bmatrix}\log(C)\\ k\end{bmatrix} = \begin{bmatrix}17/2\\ 21\end{bmatrix} \qquad \{\text{Gausseliminering}\}$$

$$\iff \begin{cases}\log(C) & = 9/10\\ k & = 7/10\end{cases}$$

$$\iff \begin{cases}C & = e^{9/10}\\ k & = 7/10\end{cases}$$

Detta ger, avrundat till närmsta heltal,

$$\begin{cases}C & \approx 2, \qquad (C \approx 3 \text{ är ok att svara})\\ k & \approx 1,\end{cases}$$

och minstakvadratanpassningen $y(t) = 2e^t$.

(12p) **b)** Nu vill vi anpassa en kubisk spline $s(t)$ som interpolerar datapunkterna så att

$$s(t) = \begin{cases}a_0 + a_1 t + a_2 t^2 + a_3 t^3, & 0 \le t \le 1,\\ b_0 + b_1 t + b_2 t^2 + b_3 t^3, & 1 \le t \le 2,\\ c_0 + c_1 t + c_2 t^2 + c_3 t^3, & 2 \le t \le 4.\end{cases}$$

Ställ upp det linjära ekvationssystem som behöver lösas för att bestämma $s(t)$ så att $s(t)$ också har kontinuerlig första- och andraderivata samt att andraderivatan är noll i ändpunkterna $t_0 = 0$ och $t_3 = 4$. Du behöver inte lösa ekvationssystemet.

Splinen ska interpolera datapunkterna, vilket ger de 6 ekvationerna

$$a_0 = e, \tag{1}$$
$$a_0 + a_1 + a_2 + a_3 = e^{\frac{3}{2}}, \tag{2}$$
$$b_0 + b_1 + b_2 + b_3 = e^{\frac{3}{2}}, \tag{3}$$
$$b_0 + 2b_1 + 4b_2 + 8b_3 = e^{\frac{9}{4}}, \tag{4}$$
$$c_0 + 2c_1 + 4c_2 + 8c_3 = e^{\frac{9}{4}}, \tag{5}$$
$$c_0 + 4c_1 + 16c_2 + 64c_3 = e^{\frac{15}{4}}. \tag{6}$$

Krav på kontinuerlig första- och andraderivata i noderna $t = 1$ och $t = 2$ ger

$$s'(t) = \begin{cases} a_1 + 2a_2t + 3a_3t^2, & 0 \le t \le 1, \\ b_1 + 2b_2t + 3b_3t^2, & 1 \le t \le 2, \\ c_1 + 2c_2t + 3c_3t^2, & 2 \le t \le 4, \end{cases}$$

och

$$s''(t) = \begin{cases} 2a_2 + 6a_3t, & 0 \le t \le 1, \\ 2b_2 + 6b_3t, & 1 \le t \le 2, \\ 2c_2 + 6c_3t, & 2 \le t \le 4, \end{cases}$$

samt de 4 ekvationerna

$$a_1 + 2a_2 + 3a_3 = b_1 + 2b_2 + 3b_3, \tag{7}$$
$$b_1 + 4b_2 + 12b_3 = c_1 + 4c_2 + 12c_3, \tag{8}$$

och

$$2a_2 + 6a_3 = 2b_2 + 6b_3, \tag{9}$$
$$2b_2 + 12b_3 = 2c_2 + 12c_3. \tag{10}$$

Ekvationssystemet sluts av de 2 ekvationer som ges av kraven $s''(0) = 0$ och $s''(4) = 0$, det vill säga

$$a_2 = 0 \tag{11}$$
$$c_2 + 12c_3 = 0. \tag{12}$$

Sammanfattningsvis definieras konstanterna i splinen $s(x)$ av det linjära ekvationssystemet

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 4 & 8 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 4 & 8 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 16 & 64 \\ 0 & 1 & 2 & 3 & 0 & -1 & -2 & -3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 4 & 12 & 0 & -1 & -4 & -12 \\ 0 & 0 & 1 & 3 & 0 & 0 & -1 & -3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 6 & 0 & 0 & -1 & -6 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 12 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ b_0 \\ b_1 \\ b_2 \\ b_3 \\ c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} e \\ e^{\frac{3}{2}} \\ e^{\frac{3}{2}} \\ e^{\frac{9}{4}} \\ e^{\frac{9}{4}} \\ e^{\frac{15}{4}} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

P2. Trapetsmetoden kan användas för att approximera integralen

$$I = \int_a^b f(x)dx.$$

(2p) **a)** Följande funktionshuvud ska användas i en tillämpning för att approximera värdet av en integral med trapetsmetoden:

```
function Th = trapets(f, int, h)
% Beräknar en approximation av integralen I av f(x) fr{\aa}n
% a till b med Trapetsmetoden och steglängd
%
% f     - Funktionshandtag till integranden f(x),
%         t.ex. f = @(x) cos(x)
% int   - [a, b], t.ex. int = [0, 1]
% h     - Steglängd i trapetsmetoden, t.ex. h = 0.1
% Th    - Numerisk approximation till integralen av f(x)

Th = ?

end
```

Komplettera funktionen med Matlab-kod som beräknar `Th`.

```
function Th = trapets(f, int, h)
% Beräknar en approximation av integralen I av f(x) fr{\aa}n a till b med
% Trapetsmetoden och steglängd h
%
% f     - Funktionshandtag till integranden f(x), t.ex. f = @(x) cos(x)
% int   - [a, b], t.ex. int = [0, 1]
% h     - Steglängd i trapetsmetoden, t.ex. h = 0.1
% Th    - Numerisk approximation till integralen av f(x)

a = int(1);
b = int(2);
x = a:h:b;
y = f(x);

Th = h*sum(y) - h/2*(y(1) + y(end));

end
```

(8p) **b)** Skriv ett program som använder funktionen `trapets(f, int, h)` för att verifiera att noggrannhetsordningen för trapetsmetoden är 2 för integralen

$$\int_0^1 e^{\cos(2x)} dx.$$

Använd en lämplig uppskattning av felet. (Du kan anta att du har funktionen `trapets(f, int, h)` även om du inte har gjort uppgift a))

```
f = @(x) exp(cos(2*x));
int = [0, 1];

h0 = 1;
hvec = [h0/2];
Th = trapets(f, int, hvec(1));
T2h = trapets(f, int, h0);
eh = abs(Th - T2h);
errvec = [eh];

tol = 1e-8;
while errvec(end) > tol
    hvec = [hvec; hvec(end)/2];
    T2h = Th;
    Th = trapets(f, int, hvec(end));
    eh = abs(Th - T2h);
    errvec = [errvec; eh];
end

figure
loglog(hvec, errvec, '*-', hvec, hvec.^2)
legend('Trapetsmetoden', 'h^2')
title('Konvergensordning för trapetsmetoden med integrand exp(cos(2x))')
xlabel('h')
ylabel('e_h')
```

(6p) **c)** Anta att integranden $f(x)$ är en glatt funktion (alla derivator är kontinuerliga) då har trapetsmetoden $T(h)$ noggrannhetsordning 2, dvs trunkeringsfelet är $E = |T(h) - I| \leq Ch^2$. För att visa detta kan felet skrivas som en summa av trunkeringsfel E_k från varje delintervall $[x_k,\, x_{k+1}] \subset [a,\, b]$, så kallade lokala trunkeringsfel:

$$E_k = \left| \frac{h}{2}(f(x_k) + f(x_{k+1})) - \int_{x_k}^{x_{k+1}} f(x)dx \right|$$

Visa att $E_k \leq Ch^3$, där C är en konstant oberoende av $h = x_{k+1} - x_k$.

Låt

$$I = \int_a^b f(x)dx = \sum_{k=0}^{n-1} \int_{x_k}^{x_{k+1}} f(x)dx,$$

där $a = x_0 < x_1 < \ldots < x_n = b$ är en diskretisering av intervallet $[a,\, b]$ i n delintervall $[x_k,\, x_{k+1}]$ med $x_{k+1} - x_k = h$, $k = 0,\, 1, \ldots,\, n$.

Det lokala trunkeringsfelet på delintervall k är

$$\begin{aligned}
E_k &= \left| \frac{h}{2}(f(x_k) + f(x_{k+1})) - \int_{x_k}^{x_{k+1}} f(x)dx \right| = \{\text{Taylorutveckling av } f(x) \text{ kring } x_k\} \\
&= \left| \frac{h}{2}(f(x_k) + f(x_{k+1})) - \int_{x_k}^{x_{k+1}} [f(x_k) + f'(x_k)(x - x_k) + \frac{1}{2}f''(x_k)(x - x_k)^2 \right. \\
&\quad \left. + \frac{1}{6}f^{(3)}(\xi)(x - x_k)^3]dx \right| = \{\text{Termvis integration}\} \\
&= \left| \frac{h}{2}(f(x_{k+1}) - f(x_k)) - \frac{h^2}{2}f'(x_k) - \frac{h^3}{6}f''(x_k) - \frac{h^4}{18}f^{(3)}(\xi) \right| \\
&= \left\{ \text{Taylors formel: } f(x_{k+1}) = f(x_k) + hf'(x_k) + \frac{h^2}{2}f''(x_k) + \frac{h^3}{6}f^{(3)}(\eta) \right\} \\
&= \left| \frac{h^3}{12}f''(x_k) + \frac{h^4}{36}\left(2f^{(3)}(\xi) + 3f^{(3)}(\eta)\right) \right|,
\end{aligned}$$

för några $\xi, \eta \in (x_k,\, x_{k+1})$.

Det följer av triangelolikheten att

$$\begin{aligned}
E_k \leq & \frac{h^3}{12}\left|f''(x_k)\right| + \frac{h^4}{36}\left|2f^{(3)}(\xi) + 3f^{(3)}(\eta)\right| \leq \{\text{Triangelolikheten igen}\} \\
\leq & \frac{h^3}{12}\left|f''(x_k)\right| + \frac{h^4}{36}\left(2|f^{(3)}(\xi)| + 3|f^{(3)}(\eta)|\right) \leq \{\text{Antag } h < 1\} \\
\leq & \frac{h^3}{36}(3|f''(x_k)| + 5\max_{x\in[x_k,\,x_{k+1}]}|f^{(3)}(x)|) = Ch^3,
\end{aligned}$$

ty f är en glatt funktion och $(x_k,\, x_{k+1})$ är ett begränsat intervall. Därmed är $f^{(k)}(x)$ begränsad för $x \in (x_k,\, x_{k+1})$ och alla k.

Vi har visat att $E_k \leq Ch^3$, vilket ger

$$\begin{aligned}
E = & |T(h) - I| = \left|\sum_{k=0}^{n-1}\left(\frac{h}{2}(f(x_k) + f(x_{k+1})) - \int_{x_k}^{x_{k+1}} f(x)dx\right)\right| \\
\leq & \sum_{k=0}^{n-1}\left|\frac{h}{2}(f(x_k) + f(x_{k+1})) - \int_{x_k}^{x_{k+1}} f(x)dx\right| = \sum_{k=0}^{n-1} E_k \leq nCh^3 \\
= & C\frac{b-a}{h}h^3 = C'h^2.
\end{aligned}$$

Alltså gäller $E \leq C'h^2$, och vi har visat att trapetsmetoden har noggranhetsordning 2.

P3. Vi önskar lösa begynnelsevärdesproblemet

$$u'(t) = -\sin(\alpha u(t) + \beta), \quad u(0) = 2, \tag{13}$$

där $\alpha = 10$ och $\beta = 1/8$.

(3 p) **a)** Formulera metoden Euler Bakåt (Implicit Euler) för ekvation (13).

Euler bakåt för det generella problemet

$$\begin{cases} u'(t) &= f(t,u), \\ u(0) &= u_0, \end{cases}$$

lyder

$$\begin{cases} u_{k+1} &= u_k + hf(t_{k+1}, u_{k+1}), \quad k = 0, 1, 2, \ldots, \\ u_0 &= u(0), \end{cases}$$

så vi får

$$\begin{cases} u_{k+1} = u_k - h\sin(\alpha u_{k+1} + \beta), \quad k = 0, 1, 2, \ldots, \\ u_0 = 2, \end{cases}$$

där $\alpha = 10$ och $\beta = 1/8$.

(6 p) **b)** I varje tidssteg med Bakåt Euler behöver en ickelinjär ekvation lösas. Formulera Newtons metod för denna ekvation. Specificera en lämplig startpunkt för Newtoniterationerna i varje tidssteg.

Låt $y = u_{k+1}$, där u_k är en approximation av $u(t_k)$ med Euler bakåt. Den ickelinjära ekvationen som måste lösas i varje tidssteg med bakåt Euler är

$$g(y) = 0,$$

där

$$\begin{aligned} g(y) &= y + h\sin(\alpha y + \beta) - u_k, \\ g'(y) &= 1 + \alpha h\cos(\alpha y + \beta). \end{aligned}$$

Newtons metod för den ickelinjära ekvationen lyder

$$\begin{cases} y_{i+1} &= y_i - \dfrac{g(y_i)}{g'(y_i)} \\ &= y_i - \dfrac{y_i + h\sin(\alpha y_i + \beta) - u_k}{1 + \alpha h\cos(\alpha y_i + \beta)}, \quad i = 0, 1, 2\ldots, \\ y_0 &= u_k, \end{cases}$$

där $y_0 = u_k$ är en lämplig startgissning, α, β är konstanter och h är tidssteget i Eulers metod.

(6 p) **c)** Skriv ett Matlab-program som beräknar en approximation av $u(3)$, dvs. lösningen till ekvation (13) vid tiden $t = 3$. Matlabprogrammet ska använda Bakåt Euler för tidsdiskretiseringen och Newton i varje tidssteg, och ska skriva ut approximationen av $u(3)$ på skärmen. För full poäng krävs att Newtons

metod är implementerad med kontroll av felet i varje iteration så att absolutbeloppet av felet är mindre än en given tolerans, tex 10^{-3}. Det får antas känt att Newtoniterationerna i varje tidssteg konvergerar.

```
% Definition av ODE:n som ska lösas
alpha = 10;
beta = 1/8;
f = @(t, u) -sin(alpha*u + beta);
u0 = 2;
T = 3;

h = 0.1;        % Steglängd i Eulers metod
tvec = 0:h:T;   % Diskretisering av intervallet [0, T] med steglängd h
uk = u0;        % Första u-värdet i Eulers metod

% Funktioner som används av Newtons metod i varje steg i Euler bakåt
g = @(y, ui) y + h*sin(alpha*y + beta) - ui;
dgdy = @(y) 1 + alpha*h*cos(alpha*y + beta);

for k = 1:length(tvec)

    % Iterera enligt Newtons metod
    yold = uk;
    ynew = yold - g(yold, uk)/dgdy(yold);

    while abs(ynew - yold) > 1e-3
        % iterera tills absolutbeloppet av felet för varje
        % steg i Euler bakåt är mindre än toleransen, t.ex 1e-3
        yold = ynew;
        ynew = yold - g(yold, uk)/dgdy(yold);
    end

    uk = ynew;
end

disp(strcat('u(3) är ungefär ', num2str(uk)))
```
