UNIVERSITETET I OSLO

Det matematisk-naturvitenskapelige fakultet

Eksamen i	MoD 190 — Numeriske beregninger
Eksamensdag:	31. Mai 2002
Tid for eksamen:	9.00 – 13.00
Oppgavesettet er på 8 sider.	
Vedlegg:	1
Tillatte hjelpemidler:	Ingen

Kontroller at oppgavesettet er komplett før
du begynner å besvare spørsmålene.

Alle 8 delspørsmål vektlegges likt.

Oppgaver med Løsningsforslag

Oppgave 1 Diverse Matlab-oppgaver

1a

Skriv opp vektoren z etter at følgende Matlab-kommandoer er utført:

```
A = [3 2 1 ; 2 3 2 ; 1 2 3]
x = A(:,1).*A(:,3)
y = A(2,:)
B = x*y
z = [B(1,:) B(2,:) B(3,:)]
z(1:2:9) = ones(1,5)
```

Vis resultatet etter hver beregning.

Løsning:

```
A=[3 2 1; 2 3 2; 1 2 3]

A =

     3     2     1
     2     3     2
     1     2     3
```

(Fortsettes på side 2.)

```
x=A(:,1).*A(:,3)

x =

     3
     4
     3

y=A(2,:)

y =

     2     3     2

B=x*y

B =

     6     9     6
     8    12     8
     6     9     6

z=[B(1,:) B(2,:) B(3,:)]

z =

     6     9     6     8    12     8     6     9     6

z(1:2:9)=ones(1,5)

z =

     1     9     1     8     1     8     1     9     1
```

1b

Skriv en Matlab funksjon `Pellipse(P,A,theta)` *som plotter en skjev ellipse gitt ved*

$$\begin{aligned} x(t) &= \cos(\theta)\left[\frac{P-A}{2}+\frac{P+A}{2}\cos(t)\right]-\sin(\theta)\left[\sqrt{A\cdot P}\sin(t)\right] \\ y(t) &= \sin(\theta)\left[\frac{P-A}{2}+\frac{P+A}{2}\cos(t)\right]+\cos(\theta)\left[\sqrt{A\cdot P}\sin(t)\right] \end{aligned}$$

for $0 \le t \le 2\pi$. *Din implementasjon skal ikke ha noen løkker.*

Løsning:

(Fortsettes på side 3.)

```
function [x,y]=Pellipse(P,A,theta)
t=linspace(0,2*pi);
ct=(P-A)/2+(P+A)/2*cos(t);
st=sqrt(A*P)*sin(t);
x=cos(theta)*ct-sin(theta)*st;
y=sin(theta)*ct+cos(theta)*st;
plot(x,y);
```

1c

Skriv et Matlab program som lager en tabell over verdien av integralene

$$\int_0^{k/10} \sqrt{x}e^{-x}dx, \quad for\ k = 1, 2, \ldots, 50.$$

Bruk Matlabrutinen `quad` *med toleranse* 10^{-6} *(se vedlegg). Hvorfor kan man forvente at* `quad` *vil trenge mange evalueringer av integranden når* x *er nær 0? Prøv å organisere beregningene slik at programmet ikke foretar unødvendig mange beregninger av integranden.*

Løsning:

Feilleddet i en kvadraturformel inneholder en høy derivert av integranden i et punkt i integrasjonsintervallet. Hvis dette leddet er stort trenger vi mange funskjonsberegninger for å kompensere for dette leddet. I vårt tilfelle vil alle deriverte av integranden være store nær 0 og vi trenger derfor mange funksjonsberegninger i alle de 50 integralene. Vi kan unngå dette ved å addere opp integralene over delintervaller. Med

$$I_k = \int_{x_{k-1}}^{x_k} \sqrt{x}e^{-x}dx, \quad k = 1, \ldots, 50$$

hvor $x_k = k/10$ har vi

$$\int_0^{x_k} \sqrt{x}e^{-x}dx = \int_0^{x_{k-1}} \sqrt{x}e^{-x}dx + I_k$$

Her er et program:

```
% Script fil: integraltabell
%lager en tabell over integralet av \sqrt(x)e^(-x)
clc
n=50;
x=linspace(0.1,5,50);
disp(' ')
disp('  b     integral(0,b) ');
disp('-----------------------');
int=quad('funk',0,0.1);k=1;
disp(sprintf(' %3.1f    %10.6f  ',x(k),int));
for k=2:50
```

(Fortsettes på side 4.)

```
    int =int+quad('funk',x(k-1),x(k));
    disp(sprintf(' %3.1f    %10.6f  ',x(k),int));
end
```

`funk.m`:

```
function fx=funk(x)
fx=sqrt(x).*exp(-x);
```

1d

Et polynom på formen

$$p(x) = a_1 + a_2x^2 + a_3x^4 + \cdots + a_nx^{2n-2}$$

sies å være like, *mens et polynom på formen*

$$p(x) = a_1x + a_2x^3 + a_3x^5 + \cdots + a_nx^{2n-1}$$

sies å være odde. *Generaliser* `HornerV(a,z)` *(se vedlegg) slik at funksjonen har et valgfritt tredje argument* `type` *som indikerer om det underliggende polynomet er like eller odde. Ved et kall på* `HornerV(a,z,'like')` *antas det at a_k er koeffisienten til x^{2k-2}, mens det ved et kall på* `HornerV(a,z,'odde')` *antas at a_k er koeffisienten til x^{2k-1}. Test på antall argumenter ved å bruke verdien av den innebygde Matlabvariablen* `nargin`.

Løsning:

Hvis $q(x) = a_1 + a_2x + \cdots + a_nx^{n-1}$ har vi $p(x) = q(x^2)$ hvis p er like, $p(x) = x * q(x^2)$ hvis p er odde og $p = q$ i det generelle tilfellet. Vi får da følgende kode:

```
 function pVal = HornerV(a,z,type)
% pVal = HornerV(a,z,type)
% evaluates a polynomial on z where
% a is an n-vector and z is an m-vector. type is an optional
%third argument which indicates if the underlying polynomial is even or odd.
%
% pVal is a vector the same size as z. If type is absent then it is assumed that
%
%          p(x) = a(1) + a(2)x + .. +a(n)x^(n-1)
%
% A call of the form a=HornerV(a,z,'like') assumes that
%
%         p(x) = a(1) + a(2)x^2 + .. +a(n)x^(2n-2)
%
% while a call of the form a=HornerV(a,z,'odde') assumes that
%
```

(Fortsettes på side 5.)

```
%         p(x) = a(1)x + a(2)x^3 + .. +a(n)x^(2n-1)
%
% pVal is a vector with the same length as z with the property that
%
%          pVal(i) = p(z(i))   ,  i=1:m.

n = length(a);
m = length(z);
pVal = a(n)*ones(size(z));
if nargin==2
    y=z;
else
    y=z.*z;
end;
for k=n-1:-1:1
    pVal = y.*pVal + a(k);
end
if ((nargin==3 ) & (type=='odde'))
    pVal=z.*pVal;
end
```

Oppgave 2 En ikke-lineær ligning

2a

La $t > 1$ være gitt. Forklar hvorfor ligningen

$$f(x) = e^x + t(x-1) = 0 \tag{1}$$

har en entydig løsning $x = x(t)$ med $0 < x(t) < 1$.

Løsning:

Vi har $f(0) = 1 - t < 0$ og $f(1) = e > 0$. Det betyr at f skifter fortegn på intervallet $(0, 1)$. Siden f er kontinuerlig finnes $0 < x = x(t) < 1$ slik at $f(x) = 0$. Siden $f'(x) = e^x + t > 0$ for $t > 0$ er f strengt monotont voksende og løsningen må være entydig.

2b

Anta at $t > 1$ og at vi benytter Newton's metode med $x_0 = 1$ til å løse ligningen (1). La $\{x_k\}$ være følgen generert ved Newton's metode og sett $e_k = x_k - x$ for alle k, hvor $x = x(t)$. Vis for $k = 0, 1, 2, \ldots$ at

$$e_{k+1} = e_k - \frac{f(x_k)}{f'(x_k)} \tag{2}$$

$$0 = f(x_k) - e_k f'(x_k) + \frac{1}{2} e_k^2 f''(\xi_k) \tag{3}$$

$$e_{k+1} = \frac{1}{2} e_k^2 \frac{f''(\xi_k)}{f'(x_k)} \tag{4}$$

(Fortsettes på side 6.)

for en ξ_k mellom x og x_k.

Løsning:

Newton's metode har formen

$$x_{k+1} = x_k - \frac{f(x_k}{f'(x_k)}.$$

Trekker vi x fra på begge sider følger (2).

Ved Taylorutvikling

$$0 = f(x) = f(x_k) + (x - x_k)f'(x_k) + \frac{1}{2}(x - x_k)^2 f''(\xi_k)$$

for en ξ_k mellom x og x_k. Siden $e_k = x_k - x$ følger (3).

Fra (3) har vi $f(x_k) = e_k f'(x_k) - \frac{1}{2}e_k^2 f''(\xi_k)$. Setter vi dette inn i (2) finner vi

$$e_{k+1} = e_k - \frac{e_k f'(x_k) - \frac{1}{2}e_k^2 f''(\xi_k)}{f'(x_k)} = \frac{1}{2}e_k^2 \frac{f''(\xi_k)}{f'(x_k)}$$

som viser (4).

2c

Vis at følgen $\{x_k\}$ generert ved Newton's metode tilfredsstiller

$$0 < x(t) < x_{k+1} < x_k \leq 1, \quad \textit{for} \;\; k = 0, 1, 2, \ldots$$

Løsning:

Siden $t > 1$ følger fra 2a at $x = x(t) > 0$. Fra (4) ser vi at $e_{k+1} \geq 0$ for $k \geq 0$ siden $f'(x) = e^x + t$ og $f''(x) = e^x$ begge er positive for alle x. Fra 2a følger at $e_0 = 1 - x > 0$ og hvis $e_k > 0$ for en $k \geq 0$ er $e_{k+1} = \frac{1}{2}e_k^2 \frac{f''(\xi_k)}{f'(x_k)} > 0$. Ved induksjon på k følger at $e_k = x_k - x > 0$ for alle $k \geq 0$. Vi har også $x_{k+1} - x_k = -\frac{f(x_k)}{f'(x_k)} < 0$. For siden $x_k > x$ og f er strengt monotont voksende er $f(x_k) > 0$ og vi har allerede observert at $f'(x_k) > 0$.

2d

Anta vi blir bedt om å lage en tabell over numeriske tilnærmelser til $x(t_n)$ for $t_n = 1 + nh$ for $n = 0, 1, \ldots, 1000$. Det blir foreslått at tilnærminger y_n til $x(t_n)$ skal beregnes ved hjelp av algoritmen

$y_0 = 0$

for $n = 0 : 999$

$y_{n+1} = y_n + h\frac{1 - y_n}{t_n + e^{y_n}}$

Hva er begrunnelsen for denne algoritmen? Hint: $x(t)$ er løsning av et initialverdiproblem.

(Fortsettes på side 7.)

Løsning:

Vi har $e^{x(t)} + t(x(t) - 1) = 0$ for alle $t > 1$. Vi deriverer denne ligningen med hensyn på t og finner

$$x'(t)e^{x(t)} + x(t) - 1 + tx'(t) = 0, \quad t \geq 1.$$

Løser vi denne ligningen med hensyn på $x'(t)$ og observerer at $x(1) = 0$ får vi at $x(t)$ er løsning av initialverdiproblemet

$$y'(t) = \frac{1 - y(t)}{t + e^{y(t)}}, \quad y(1) = 0.$$

Algoritmen er Euler's metode anvendt på dette problemet.

(Fortsettes på side 8.)

Matlab-vedlegg

```
  function pVal = HornerV(a,z)
% pVal = HornerV(a,z)
% evaluates the Vandermonde interpolant on z where
% a is an n-vector and z is an m-vector.
%
% pVal is a vector the same size as z with the property that if
%
%          p(x) = a(1) + .. +a(n)x^(n-1)
% then
%          pVal(i) = p(z(i))  ,  i=1:m.

n = length(a);
m = length(z);
pVal = a(n)*ones(size(z));
for k=n-1:-1:1
   pVal = z.*pVal + a(k);
end
```

```
QUAD   Numerically evaluate integral, adaptive Simpson quadrature.
    Q = QUAD(FUN,A,B) tries to approximate the integral of function
    FUN from A to B to within an error of 1.e-6 using recursive
    adaptive Simpson quadrature.  The function Y = FUN(X) should
    accept a vector argument X and return a vector result Y, the
    integrand evaluated at each element of X.
```