TANA17 Matematiska beräkningar med MATLAB för M, DPU

Fredrik Berntsson, Linköpings Universitet

Föreläsning 7

- Textsträngar. Formatterade utskrifter.
- Filhantering.
- Seminarieuppgiften.

Text och Strängar

Definition En textrad kallas för en *sträng*. I Matlab lagras strängar som radvektorer där varje element representerar ett tecken.

Exempel Skapa en sträng som innehåller texten *Jag vill vinna!* genom att skriva

```
>> str = 'Jag vill vinna!'
```

Man kan ändra *vill* till *ska* genom

```
>> str(5:8)= 'ska ';
```

Kommentar Tecknet `'` kallas *sträng parentes*. Detta påbörjar och avslutar strängar.

Precis som för matriser kan vi sätta samman strängar.

Exempel Skriver vi

```
>> str1 = 'Jag heter ';  % notera extra mellanslag.
>> str2 = 'Fredrik';
>> str = [ str1 , str2 ];
```

så fås en sträng *Jag heter Fredrik*.

Detta är ofta praktiskt då man vill konstruera strängar som består av både text och siffror.

Funktionen `num2str` omvandlar ett tal till en sträng.

Exempel För att konvertera siffror till strängar finns `num2str` använd denna för att skapa en sträng $\sin(2) = 0.9093$.

Kommentar Det finns även en funktion `str2num` men den är inte alls lika användbar.

Inmatning

Funktionen `input` används för att fråga en användare efter indata.

Exempel Skriver vi

```
>> x = input('Get ett tal: ');
Get ett tal: 17
```

Matlab väntar tills användaren skrivit in ett tal. Detta tilldelas sedan till variabeln *x*.

Exempel Vi kan fråga efter en sträng genom att skriva

```
>> Str = input('Get en kort text: ','s');
Ge en kort text: Hej jag heter Fredrik
>> disp( Str )
Hej jag heter Fredrik
```

Då en *sträng* är en radvektor ger `length` antalet tecken i strängen.

Exempel Skriv ett program `NamnLangd.m` som frågar efter ditt namn och beräknar antalet bokstäver i namnet. Utskrift skall ske enligt följande mönster

```
>> NamnLangd
Vad heter du? Fredrik
Hej, Fredrik. Ditt namn innehåller 7 bokstäver.
```

På filen `NamnLangd.m` skriver vi

```
Namn = input('Vad heter du?','s');
str1=['Hej, ',Namn,'. Ditt namn innehåller '];
str2=[ num2str(length(Namn)),' bokstäver' ];
disp( [str1 str2] )
```

Kommentar Matlab används i allmänhet inte för att behandla text.

Jämförelser mellan strängar utförselementvis. Funktionen `strcmp` testar om två strängar är lika eller inte.

Exempel Vi skriver

```
>> str1='abc'; str2='aBc';
>> str1 == str2
 ans =
     1     0     1
```

där alltså `'b'` och `'B'` är olika tecken. Skriver vi

```
>> strcmp( str1 , str2 )
 ans =
     0
```

dvs vi får falskt då strängarna är olika.

Funktionen `blanks` skapar en sträng bestående av endast blanktecken.

Exempel Skriv en funktion `TaBortBlanka` med en sträng som inparameter och samma sträng, fast utan blanktecken, som utparameter.

Vi skall kunna skriva

```
>> str1 = 'Hej jag heter Fredrik!';
>> str2 = TaBortBlanka( str1 );
>> disp( str2 );
  HejjagheterFredrik!
```

Övning Skriv ett flödesschema och översätt detta till en Matlab funktion.

På filen `TaBortBlanka` skriver vi

```
function [UtStr]=TaBortBlanka( InStr )

  n=length( InStr );
  UtStr=blanks( n );
  j=0;
  for i=1:n
    if InStr(i) ~= ' '
       j=j+1;     % Plats att kopiera nästa tecken!
       UtStr(j)=InStr(i);
    end
  end
  UtStr=UtStr(1:j); % Vi kopierade totalt j tecken.
end
```

Exempel Vi vill undersöka konvergensen hos kvadratrotsiterationen,

$$x_k = \frac{1}{2}\left(x_{k-1} + \frac{a}{x_{k-1}}\right), \quad k = 1, 2, 3, \ldots$$

som konvergerar mot $\sqrt{a}$.

Skriv därför ett Matlab program som beräknar talföljden x_k, för $k = 1, 2, \ldots, 20$.

Skriv ut k, x_k, och $|x_k - a|$ i varje steg.

Programmet blir

```
xk=1;a=3;
for k=1:20
   xk=(xk+a/xk)/2;
   str=['k=',num2str(k),' ger xk=',num2str(xk)];
   str=[str,' och felet ',num2str(abs(xk-sqrt(a)))];
   disp(str)
end;
```

Kör vi programmet fås en utskrift

```
k=1 ger xk=2 och felet 0.26795
k=2 ger xk=1.75 och felet 0.017949
k=3 ger xk=1.7321 och felet 9.205e-05
k=4 ger xk=1.7321 och felet 2.4459e-09
k=5 ger xk=1.7321 och felet 0
```

Vi behöver ett sätt att styra exakt hur siffror skrivs ut!

Formatterade utskrifter

För formatterade utskrifter används funktionen `fprintf`. Kommandot har formen

```
>> fprintf( <dest>,'sträng',<variabler> )
```

där `'sträng'` består av text och speciella styrtecken. Vill man skriva ut till skärmen gäller att `<dest>` är 1.

Exempel Skriv ut en textsträng *sin(2.3)=0.7457* genom

```
>> x=2.3; y=sin(x);
>> fprintf( 1 , 'sin(%3.1f)=%6.4f\n',x,y);
sin(2.3)=0.7457
```

Vid utskrift med `fprintf` gäller

Styrtecken är `\n` för nyradstecken, `\t` för tab, och `''` för apostrof.

Siffror kan skrivas ut på

- `%e` eller `%E` för exponentform.
- `%f` för vanlig decimalform.
- `%u` eller `%i` för heltal.

Hjälptexten är hyfsad (`help fprintf`).

Exempel Skriv ut texten

```
Det gäller att exp(-7.600)=5.00e-04.
```

Utskrift på exponentform är bra då små eller stora tal skall skrivas ut.

Exempel Kvadratrots programmet ändras till

```
xk=1;a=3;
for k=1:5
 xk=(xk+a/xk)/2;
 ek=abs(xk-sqrt(a));
 fprintf(1,'x(%i)=%6.4f och e(%i)=%6.1e.\n',k,xk,k,ek);
end;
```

Kör vi programmet fås en utskrift

```
  x(1)=2.0000 och e(1)=2.7e-01.
  x(2)=1.7500 och e(2)=1.8e-02.
  x(3)=1.7321 och e(3)=9.2e-05.
  x(4)=1.7321 och e(4)=2.4e-09.
  x(5)=1.7321 och e(5)=0.0e+00.
```

Utskrifter till en textfil

Innan vi kan läsa/skriva till en fil så måste den *öppnas*. Detta görs med kommandot

```
>> fid = fopen('filnamn.tex','w');
```

där `fid` är en *filidentifierare*, och `'w'` betyder att vi skall skriva till filen.

Alternativ är `'r'` för läsning och `'a'` för att skriva till slutet av en existerande fil.

Efter att vi skrivit färdigt måste filen *stängas*. Det görs med

```
>> fclose( fid );
```

Efter att vi öppnat en fil för skrivning kan vi skriva till den med `fprintf`. Skriv i Matlab

```
>> utfil = fopen('textfil.txt','w');
>> fprintf( utfil , 'Hej\n' );
```

så hamnar texten *Hej* på filen `textfil.txt`.

Exempel Vi behöver en tabell över funktionen $f(x) = \sin(x^2)$ för Nst jämnt utspridda värden mellan 0 och 2π.

Tabellen skall skrivas till en fil `sinus_tabell.txt` och ha följande utseende

```
************************
*    x     *   sin(x)  *
************************
* 0.23124 *  0.229185 *
************************
```

Exempel Programmet AMPL löser ett optimeringsproblem av typen

$$\min c^T x, \text{ då } Ax \leq b.$$

där vektorerna c, b och matrisen A är indata till programmet. Lösningen blir vektorn x.

I Matlab har vi beräknat vektorerna c och b och vill skriva dem till en indata fil som kan läsas av AMPL. Om

$$c = (1\,,\ -2\,,\ -3)^T \text{ och } b = (2\,,\ 1\,,\ 5)^T,$$

skall datafilen innehålla raderna

```
param c:= 1 1.0 2 -2.0 3 -3.0;
param b:= 1 2.0 2 1.0 3 5.0;
```

Skriv en lämpliga Matlab funktioner som skriver ner dessa rader till en fil.

Lösning Vi behöver en funktion som skriver ut en vektor till filen. Alltså

```
function []=printVector( fid,vectorName,data )
```

som kan anropas exempelvis

```
>> printVector( 1 , 'c' , [ 1.0 -2.0 -3.0] );
param c:= 1 1.0 2 -2.0 3 -3.0;
```

där utskriften sker till skärmen.

Vi behöver dessutom en huvudprogram som öppnar en fil och skriver ner de båda vektorerna till den.

```
function []=printDataFile( fileName,c,b );
```

Vektorerna *c* och *b* skall kunna vara av godtycklig dimension.

På filen `printVector.m` skriver vi

```
function []=printVector( fid,vectorName,data )
  fprintf( fid,'param %s:=',vectorName);
  n=length(data);
  for i=1:n
    fprintf( fid,' %i %7.3f',i,data(i) );
  end
  fprintf( fid,';\n'); % Avsluta med ; och nyrad.
end
```

I Matlab terminalen skriver vi

```
>> printVector(1,'c',[1 -2 3 2] )
param c:= 1  1.0000 2 -2.0000 3  3.0000 4  2.0000;
```

På filen `printDataFile.m` skriver vi

```
function []=printDataFile( fileName,c,b )
  fid=fopen( fileName , 'w' );
  printVector( fid , 'c' , c );
  printVector( fid , 'b' , b );
  fclose( fid );
end
```

I Matlab skriver vi

```
>> printDataFile( 'data.txt',[1 -2 -3],[2 1 5] );
>> type data.txt
param c:= 1  1.0000 2 -2.0000 3 -3.0000;
param b:= 1  2.0000 2  1.0000 3  5.0000;
```

Filen `data.txt` har skapats med rätt innehåll!

Seminarieuppgifterna

Mål Seminarieuppgiftena syftar till att ge

- träning i att skriva lite mer omfattande program med flera deluppgifter som kan lösas separat som olika funktioner. Delarna skall sedan kopplas samman för att lösa hela uppgiften.
- träning i skriftlig presentation. Uppgiften, Lösningen, och resultatet skall beskrivas i en kort rapport.
- erfarenhet av att använda Matlab för att lösa ett någorlunda realistiskt problem från en tillämpning.

Genomförande

- Varje grupp tilldelas en uppgift. Denna annonseras på kurshemsidan i början av vecka 50.
- Två handledningstillfällen i datorsal finns. Det är viktigt att ni är förberedda så att ni kan utnyttja läraren effektivt.
- Senast dagen innan seminariet skall den grupp som skall opponera ha fått rapporten. På seminariet ges varje grupp kommentarer då rapporterna och resultaten diskuteras i gruppvis. Lärare finns tillgänglig för frågor.
- Senast två dagar efter seminariet skall opponenternas kritik åtgärdas och rapporten lämnas in till respektive lärare för godkännande.

Krav I rapporten skall ingå

- en kort bakgrund till uppgiften.
- en beskrivning av de olika funktioner som utvecklats. Den metod som implementeras i funktionen skall beskrivas.
- ett exempel där funktionerna används för att lösa en uppgift. Här ingår grafer.
- Matlabprogram skall inte ingå i rapporten utan dessa skickas med som bilaga.

Reservtillfälle Deltagande i seminariet är obligatoriskt. Ett ett extra tillfälle kommer att organiseras i januari precis innan period VT1 börjar.

Vet du att du inte kan vara med på ordinarie seminarietillfälle så kontakta din lektionsledare. Missar du ordinarie tillfälle på grund av sjukdom så ta kontakt med mig snarast efter.