Numerieke Modellering en Benadering:
MATLAB Handleiding

Hendrik Speleers
Joris Van Deun
Jan Van lent

Inhoudsopgave

1 Matlab

1.1 Help!

Een heel belangrijk commando in MATLAB is `help`. Enkele voorbeeldjes

```
help
help general
help graphics
help plot
help max
```

Verder kan je ook het commando `lookfor` gebruiken. Als je een grafisch MATLAB venster ter beschikking hebt, kan je ook via de menu's documentatie raadplegen. Op het einde van dit document vind je een aantal verwijzingen naar MATLAB informatie op het Internet.

Opgave 1. Zoek op wat de volgende commando's doen en hoe je ze moet gebruiken: `diff`, `orth`, `fliplr`

Oplossing.

```
help diff
help orth
help fliplr
```

1.2 Matlab verlaten

Je kan MATLAB verlaten met `quit` of `exit`. Je bent dan al je gegevens en berekeningen kwijt, tenzij je ze hebt opgeslagen (zie 1.5).

1.3 Matrices en vectoren

De belangrijkste gegevensstructuur in MATLAB is de matrix. Vectoren zijn matrices met 1 rij of kolom. Getallen kan je beschouwen als 1×1 matrices. MATLAB bezit een heleboel handige operaties voor het manipuleren van matrices. Enkele voorbeeldjes.

1.3.1 Matrices genereren

Met de symbolen `[` en `]` kan je matrices opgeven. De getallen worden gescheiden door spaties of komma's, rijen in een matrix worden gescheiden door newlines of puntkomma's.

```
x = [1 2 4]
y = [1;2;4]
A = [1 2 3
     4 5 6]
```

De functies `zeros(m,n)`, `ones(m,n)`, `rand(m,n)`, `eye(m,n)` genereren $m \times n$ matrices met nullen, enen, willekeurige elementen en enen op de diagonaal respectievelijk. Als je maar 1 argument opgeeft krijg je vierkante matrices. Met de uitdrukkingen `begin:einde` en `begin:stapgrootte:einde` kan je een rijvector van opeenvolgende getallen genereren.

```
a = 1:10
b = -3:4
c = 0:0.1:3
d = 5:-1:1
```

Met de functie `linspace(begin,einde,n)` kan je het aantal getallen opgeven in plaats van de stapgrootte.

1.3.2 Elementen opvragen en wijzigen

```
A = rand(4)
A(2,3)
A(3,2) = 0
```

1.3.3 Submatrices

In plaats van één enkele index kan je ook een vector van indices opgeven. Op deze manier kan je submatrices selecteren of wijzigen. De notatie ':' duidt alle elementen in een rij of kolom aan. Je kan `end` gebruiken als grootst mogelijke index.

```
A = reshape(1:(6*7), 6, 7)
A(:,:)
A(1,:)
A(:,2)
A(1:3, 2:5)
A(6:-1:1, :)
A([1,3,4], [7,2])
A(1:2, 3:4) = A(3:4, 3:4)
A(:, end:-1:1)
```

1.3.4 Matrices samenvoegen

Je kan matrices opgeven als elementen van een matrix.

```
A = rand(2,3)
B = rand(2,3)
C = [A B]
D = [A;B]
```

1.3.5 Matrices manipuleren

```
[m,n] = size(A)  % aantal rijen en kolommen van A
m = size(A, 1)   % aantal rijen van A
n = size(A, 2)   % aantal kolommen van A
l = length(A)    % grootste dimensie van A, vooral nuttig voor vectoren
B = A'           % transpose (hermitisch toegevoegde voor complexe matrices)
L = tril(A)      % onderdriehoeksdeel van A
B = fliplr(A)    % A horizontaal spiegelen (B = A(:, end:-1:1))
```

```
R = real(A)      % re"eel deel van A
I = imag(A)      % imaginair deel van A
I = find(X)      % indices van niet-nul elementen van vector X
[I,J] = find(A)  % indices van niet-nul elementen van matrix A
```

1.4 Operaties en functies

We willen graag berekeningen uitvoeren op matrices. Ook hiervoor is MATLAB heel goed uitgerust.

1.4.1 Elementaire bewerkingen

De bewerkingen `+`, `-`, `*`, `/` werken zoals je het zou verwachten, zowel op getallen als op matrices of combinaties van de twee. Voor vierkante matrices is de bewerking `A/B` wiskundig equivalent met `A*inv(B)`. De bewerking `A\B` is wiskundig equivalent met `inv(A)*B`. De ingebouwde bewerkingen zijn echter numeriek gezien beter dan het expliciet berekenen van een inverse. De operatoren `/` en `\` zijn ook gedefinieerd voor niet-vierkante matrices. In dat geval berekenen ze een kleinste kwadraten oplossing. Machtsverheffing '`^`' kan toegepast worden op vierkante matrices.

De elementaire bewerkingen kunnen ook element per element uitgevoerd worden door ze door een punt te laten voorafgaan.

```
A = rand(4,4)
B = tril(ones(4))
A*B
A.*B
A^(-1)
A.^(-1)
(1:10).^2
2.^(1:10)
```

1.4.2 Relationele bewerkingen

Vergelijkingen zoals `==`, `<`, `<=`, `~=` werken ook voor matrices; ze worden element per element uitgevoerd. Je kan dit bijvoorbeeld gebruiken om bepaalde elementen uit een matrix te selecteren.

```
a = linspace(0, 1, 10)
b = linspace(-1, 2, 10)
a < b
c = (a < b) .* a + (a >= b) .* b
d = a(find(a < b))
```

Opgave 2. Ga na wat de laatste twee bevelen uit het voorgaande voorbeeld doen (voor algemene `a` en `b`).

Oplossing. `c` *is de vector met de kleinste waarden van de overeenkomstige elementen uit* `a` *en* `b` *(*`c = min(a, b)`*). De vector* `d` *bevat enkel die waarden uit* `a` *die kleiner zijn dan de overeenkomstige waarden in* `b`*.*

1.4.3 Logische bewerkingen

```
a & b     % en
a | b     % of
~a        % negatie
xor(a, b) % exclusieve of
```

Opgave 3. Zoek uit wat de volgende uitdrukkingen doen (voor algemene `a`, `b`, `c` en `d`).

```
all((a == b) | (c > 1))
any((a < b) & (c < d))
```

Oplossing. *Er geldt dat* `all((a == b) | (c > 1)) == 1` *als voor alle* `i` *geldt dat* `(a(i) == b(i)) | (c(i) > 1)`. *Er geldt dat* `any((a < b) & (c < d)) == 1` *als er een* `i` *bestaat waarvoor* `(a(i) < b(i)) & (c(i) < d(i)) == 1`.

1.4.4 Wiskundige functies

Enkele functies:

```
sin, cos, tan, asin, atan
exp, log, log10, sqrt
abs, floor, ceil, mod, rem, sign
```

1.4.5 Opmerkingen

- De ingebouwde operaties zijn vaak compact en efficiënt; maak er dan ook gebruik van.
- Zie ook `help elmat`, `help ops`, `help relop`, `help elfun`.

1.5 In- en uitvoer

Als we een berekening uitvoeren in MATLAB krijgen we onmiddellijk het resultaat te zien. Je kan dit vermijden door je bevel te beëindigen met ';'.

```
x = rand(100,1);
y = rand(1,100);
A = x * y;
size(A)
```

Je kan gegevens naar een bestand schrijven met het commando `save` en ze weer opvragen met het commando `load`. Met de optie `-ascii` worden de getallen leesbaar weggeschreven.

```
A = rand(10, 3)
B = rand(40, 3)
save matrixa A
save gegevens A, B
save matrixa.txt A -ascii
save matrixad.txt A -ascii -double
load matrixd.txt -ascii
```

Voor meer controle over de indeling van de uitvoer zie `help fprintf`.

1.6 Tijd meten

Voor het bepalen van de tijd die een berekening vraagt kan je gebruik maken van de commando's `tic` en `toc`. `tic` zet de timer op 0 en `toc` geeft de tijd verlopen sinds de vorige oproep van `tic`. Bijvoorbeeld:

```
tic; A = rand(10,100)*(rand(10,100)'); toc
```

Zie ook het commando `cputime`.

1.7 For, while, if

MATLAB is een volwaardige programmeertaal en heeft dus ook constructies voor lussen en voorwaardelijke uitvoering. Zie `help for`, `help while` en `help if` voor de syntax van deze constructies. Vermijd het gebruik van deze constructies als het kan. De ingebouwde operaties en functies zijn meestal sneller en overzichtelijker. Gebruik dus niet

```
s = 0; for i = 1:length(x), s = s + x(i)*x(i); end
```

maar

```
s = sum(x.^2);
```

1.8 M-files

Je kan in MATLAB al heel wat bereiken door gewoon commando's in te typen. Vaak is het echter interessant om een aantal bevelen te groeperen in een bestand. Als je zo'n bestand een naam geeft die eindigt op `.m` en het in een map plaatst die gelezen wordt door MATLAB (bijvoorbeeld de huidige map, zie ook `help pwd`, `help cd` en `help path`), dan kan je de bevelen uitvoeren door de naam van het bestand in te tikken. Zulke bestanden worden scripts genoemd. Ze zijn handig om bijvoorbeeld een reeks bevelen een aantal keer uit te voeren met lichte wijzigingen. Je zorgt er zo ook voor dat je resultaten reproduceerbaar zijn. Het volgende bestand geeft een sinus weer. Je kan makkelijk de periode en het interval wijzigen.

```
% plotgolfje.m

a = 0.0;
b = 1.0;
T = 2;
n = 20;
t = linspace(a, b, n);
y = sin(2*pi*t/T);
plot(t, y)
d = 1 + 1 + 1 + 1 + 1 + 1 + 1 ...
    + 1 + 1 + 1 + 1 + 1 + 1 ...
    + 1 + 1 + 1 + 1 + 1;
```

Het %-teken zorgt ervoor dat alles tot het einde van de lijn genegeerd wordt. Je kan het gebruiken om commentaar in je programma's te zetten. Lange lijnen kan je opsplitsen met '...'.

Je kan ook zelf functies schrijven. De m-file begint dan met het specifiëren van naam, argumenten en resultaten van de functie. Je kan dan de naam van het bestand gebruiken alsof het een ingebouwde functie was. De naam van het bestand en de naam gebruikt op de eerste lijn van het bestand zijn bij voorkeur identiek. Een voorbeeld:

```
function [t, y] = golfje(n, T, a, b)

% function [t, y] = golfje(n, T, a, b)
%
% n abscissae (t) en functiewaarden (y) voor een sinus
% met periode T in het interval [a, b]

t = linspace(a, b, n);
y = sin(2*pi*t/T);
```

Het eerste blok commentaar kan opgevraagd worden met het commando `help <bestandsnaam>`.

Opgave 4. Schrijf een functie die 2 matrices met elkaar vermenigvuldigt. Gebruik enkel scalaire optellingen en vermenigvuldigingen (je hebt dus 3 expliciete lussen nodig). Vergelijk efficiëntie van deze functie met de ingebouwde operatie.

Oplossing. *Het bestand* `matmul.m`*:*

```
function C = matmul(A, B)

% function C = matmul(A, B)
%
% Vermenigvuldig matrices A en B
% gebruikt lussen ipv de ingebouwde operaties

[m,k1] = size(A);
[k2,n] = size(B);
if k1 ~= k2
  error('size(arg1,2) ~= size(arg2,1)');
end

C = zeros(m,n);
for i = 1:m
  for j = 1:n
    for k = 1:k1
      C(i,j) = C(i,j) + A(i,k)*B(k,j);
    end
  end
end
```

Oplossing (vervolg). *Een test:*

```
m = 40;
n = 50;
k = 100;
A = rand(m,k);
B = rand(k,n);
tic ; C1 = A*B; ; toc
tic ; C2 = matmul(A, B) ; toc
norm(C1 - C2)
```

1.9 Visualisatie

Een van de grote troeven van MATLAB is het gemak waarmee je aantrekkelijke figuren kan produceren. Een greep uit de mogelijkheden.

1.9.1 X-Y grafieken

`plot(X,Y)` tekent een gebroken lijn door de punten `(X(1),Y(1))`, `(X(2), Y(2))`, enz. Als je het eerste argument weglaat worden de getallen 1, 2, enz. als x-waarden gebruikt. Met extra argumenten kan je de lijnstijl en symbolen wijzigen.

```
plot(X, Y, '.')   % enkel punten
plot(X, Y, '+-')  % kruisjes en lijnen
plot(X, Y, '-.')  % punt-streep lijn
plot(X, Y, 'r--') % rode onderbroken lijn
```

Je kan ook meerdere grafieken op één figuur bekomen.

```
t = linspace(0, 2*pi, 20);
X = sin(t); Y = cos(t);
plot(X, Y, '+', t, X, 'bx:')
```

Als `X` of `Y` een matrix is (of allebei) dan worden de kolommen (met verschillende kleuren) afgebeeld.

Je kan de commando's `title`, `xlabel`, `ylabel` en `text` gebruiken om tekst toe te voegen aan een figuur. Het commando `axis` laat toe de weergave van de assen te controleren. Met het commando `hold on` zorg je ervoor dat de figuur niet gewist wordt bij het volgende grafische commando. Dit kan je weer afzetten met `hold off`. Je kan een figuur expliciet wissen met het commando `clf`.

1.9.2 Logaritmische grafieken

De commando's `semilogx`, `semilogy` en `loglog` hebben dezelfde syntax als `plot`, maar gebruiken logaritmische assen (x, y of allebei).

1.9.3 3D grafieken

Het commando `surf(X,Y,Z)` geeft het oppervlak gedefinieerd door de punten `X(i,j)`, `Y(i,j)`, `Z(i,j)` weer (zie ook `mesh`). Je kan de contourlijnen van de functie $z(x, y)$ bekijken met `contour(X,Y,Z)`. Voor regelmatige rechthoekige roosters kunnen de matrices `X` en `Y` vervangen worden door de vectoren met x en y coördinaten. De functie `[X, Y] = meshgrid(x, y)` genereert de matrices `X` en `Y` die overeenkomen met de vectoren van coördinaten `x` en `y`.

```
x = linspace(0, 1, 20);
y = linspace(0, 1, 15);
[X, Y] = meshgrid(x, y);
Z = sin(4*pi*X).*(1-Y).*Y;
surf(x,y,Z)
```

1.9.4 Meerdere grafieken combineren

Het commando `figure` opent een nieuw venster, met `figure(n)` zorg je ervoor dat de volgende grafiek in venster `n` terechtkomt. Met het commando `subplot` kan je meerdere grafieken op een figuur combineren. `subplot(m, n, p)` creëert een subfiguur op positie `p` in een rooster van grafieken met `m` rijen en `n` kolommen. De posities worden geteld van links naar rechts en van boven naar beneden. Als `p` een vector is, overlapt de nieuwe grafiek al deze posities.

```
figure(1); clf
fplot('sin(x)',[0,2*pi])
figure(2); clf
subplot(3,2,1); fplot('cos(x)', [0,2*pi])
subplot(3,2,2); fplot('cos(2*x)', [0,2*pi])
subplot(3,2,[3,6]); fplot('sin(x)', [0,2*pi])
```

Zie ook `help graphics`, `help graph2d`.

2 Verwijzingen

De website van de makers van MATLAB: `www.mathworks.com`

Op het internet zijn er een massa MATLAB handleidingen en referenties te vinden. Gebruik hiervoor je favoriete zoekmachine. Enkele voorbeelden:

- `http://www.mathworks.com/help/techdoc/learn_matlab/bqr_2pl.html`
- `http://www.mathtools.net/MATLAB/Tutorials/index.html`
- `http://www-h.eng.cam.ac.uk/help/tpl/programs/Matlab/tricks.html`
- `http://www.cyclismo.org/tutorial/matlab/`
- `http://www.math.ufl.edu/help/matlab-tutorial/`