Matlab introductie

Kees Vuik

2014

Delft University of Technology
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft Institute of Applied Mathematics

Voorwoord

Deze korte introductie in het gebruik van Matlab is voornamelijk bedoeld om je zo snel mogelijk op gang te helpen bij het Modelbouw practicum (wi1606).

Het gebruik van de handleiding

Zodra er gevraagd wordt iets letterlijk in te tikken wordt dit weergegeven in een ander lettertype, bijvoorbeeld: `plot(y,t);`
Commando's die binnen Matlab moeten worden uitgevoerd worden bovendien voorafgegaan door het $\gg$–teken, dit teken (de prompt) staat al op het scherm.
Tekst die op het beeldscherm verschijnt wordt als volgt weergegeven:
Enter your password:

Werken met Matlab

Klik na het inloggen op de Matlab button. Hiermee wordt het Matlab programma opgestart. Matlab is een interactief systeem dat wil zeggen dat de ingetikte commando's direct uitgevoerd worden en de uitvoer verschijnt, indien gewenst direct op het scherm. Het is echter ook mogelijk programma's te schrijven met behulp van Matlab. Zo'n programma is in feite niets anders dan een verzameling commando's in een bestand. Het maken van een dergelijk programma komt later aan de orde. Matlab is een numeriek pakket, dat wil zeggen dat er geen symbolische berekeningen uitgevoerd kunnen worden. Matlab kent twee "schermen":

- *tekst* Dit scherm verschijnt zodra je Matlab hebt gestart.
- *grafisch* Op dit scherm verschijnen de plots die je maakt.

Invoeren van opdrachten

Het intikken van opdrachten vraagt wel enige nauwkeurigheid. Het vergeten van een * of een) kan er toe leiden dat de opdracht niet of verkeerd wordt uitgevoerd. Hieronder volgt een lijst met links de in de wiskunde gebruikelijke notatie en rechts de vorm waarin het ingetikt moet worden.

Wiskundige notatie	Intikken
$3xy$	`3*x*y`
$\frac{a}{b}$	`a/b`
a^b	`a^b`
$\sqrt{x}$	`sqrt(x)`

Voor andere standaardfuncties verwijzen we naar de Matlab handleiding.

(i) Hou er rekening mee dat Matlab 'case sensitive' is. Dat wil zeggen dat er verschil is tussen hoofd en kleine letters. Het getal i bij voorbeeld wordt binnen Matlab geschreven als `i`, het invoeren van `I` zal een foutmelding opleveren. Het commando `casesen off` zorgt er voor dat er geen onderscheid meer gemaakt wordt tussen hoofd en kleine letters. Het beste is om commando's met kleine letters in te typen.

(ii) Opdrachten in Matlab worden afgesloten door het indrukken van de Enter of RETURN-toets. Het antwoord verschijnt direct op het scherm. Door het commando af te sluiten met een `;` (puntkomma) wordt de uitvoer onderdrukt.

Onderbreken van een berekening

Het is mogelijk dat je bij het maken van een programma of berekening een fout hebt gemaakt waardoor de berekening niet meer stopt. Door het indrukken van de `Ctrl + C` (Ctrl-toets en C-toets tegelijk indrukken) is dit proces alsnog te stoppen.

Rekenen met getallen

Matlab is natuurlijk ook als 'rekenmachine' te gebruiken. Om met het pakket kennis te maken en te wennen aan het invoeren van commando's zullen we een aantal korte opdrachten uitvoeren.

1. Tik in:

   ≫ `12/3` *(Druk nu de Enter of RETURN–toets in)*

   Op het scherm verschijnt de uitvoer van het commando in de volgende vorm:

   ```
   ans =

         4
   ```

   Het Matlab pakket rekent altijd in 15 cijfers nauwkeurig. Standaard worden de getallen afgedrukt in 5 cijfers met een drijvende komma.

2. Voer in:

   ≫ `12/9` Als antwoord verschijnt op het scherm:

```
ans =

      1.3333e+000
```

Met behulp van het format-commando kan je opgeven hoe het resultaat van een berekening verschijnt. In onderstaande tabel staan de verschillende opties:

Commando	**Resultaat**	**Voorbeeld**
format short	5 cijfers, vaste komma	3.1416
format long	15 cijfers, vaste komma	3.14159265358979
format short e	5 cijfers, drijvende komma	$3.1416e+000$
format long e	16 cijfers, drijvende komma	$3.141592653589793e+000$
format compact	onderdrukt lege regels in uitvoer	
format loose	toevoegen lege regels in uitvoer	

3. Tik in:

```
≫ format long
≫ 12/9
```

Het antwoord verschijnt nu in 15 cijfers.

De help-faciliteit

Matlab beschikt over een uitgebreide help-functie. Met behulp van het commando `help` *onderwerp* verschijnt er een toelichting bij *onderwerp*. Bijvoorbeeld:
`≫ help help` geeft informatie over de help-faciliteit. Het commando `help` zonder *onderwerp* geeft een lijst van mogelijke *onderwerpen*.

4. Tik in:

```
≫ help format
```

Hierna verschijnt alle informatie over het `format` commando.

Toekenningen

Binnen Matlab kunnen ook variabelen gebruikt worden.

5. Tik in:

```
≫ format short e
≫ a = 1
≫ b = 2
≫ a + b
≫ a - b
```

De variabelen blijven hun waarde behouden totdat het Matlab pakket verlaten wordt, of nadat het `clear` commando gegeven is.

Het gebruik van een herhaal lus

Zoals in het begin al gezegd kan er ook geprogrammeerd worden met behulp van Matlab. Je gaat nu een eenvoudig (deel) programmaatje maken dat een aantal waarden van functie $y = \sqrt{x}$ zal bepalen. Je gebruikt daarvoor een herhaal-lus.
De syntax voor een herhaal-lus in Matlab is:

```
for variabele = expressie
      opdracht
      ...
      opdracht
end
```

De `expressie` heeft in dit geval de volgende vorm: `1:1:200`. Dat wil zeggen dat we de functiewaarden van $y = \sqrt{x}$ gaan berekenen met beginwaarde 1, stapgrootte 1 en eindwaarde 200.
De berekende functiewaarden worden opgeslagen in een array. Een voordeel van Matlab is dat het niet nodig is van te voren aan te geven hoeveel elementen een array zal gaan bevatten. Je hoeft je dus niet af te vragen wat de maximale grootte van het array zou kunnen worden. De array begint altijd met de index 1.

6. Tik in (de puntkomma (;) achter de tweede regel zorgt er voor dat de uitvoer niet op het scherm verschijnt):

```
≫ for x = 1:1:200
y(x) = sqrt(x);
end;
```

7. Een mogelijkheid om de functiewaarden op het scherm te tonen is het weg laten van de puntkomma achter de regel `y(x) = sqrt(x)`. Het probleem hierbij is dat alle antwoorden niet op één scherm passen. Er ontstaat een zeer onrustig beeld dat weinig informatie verschaft. Bovendien kost het hele proces meer tijd.

De functiewaarden worden opgeslagen in een array. De antwoorden kunnen nu worden afgedrukt door het array op het scherm te tonen.

8. Tik in:

   ≫ `y`

   Ook nu scrollen de antwoorden over het scherm, echter zeer snel en op een onoverzichtelijke manier. Matlab drukt het array af als een rij-vector. Door:

9. ≫ `y'`

   in te tikken wordt het array afgedrukt als een kolomvector.

10. Het 133^e element uit de array krijg je door het volgende in te tikken.

    ≫ `y(133)`
    Het antwoord is 1.1533e+001. Als je geïnteresseerd bent in het resultaat van elke tiende stap dan kan je de volgende opdrachten gebruiken. De verschillende commando's worden na de opdracht toegelicht.

11. Tik het volgende in:

```
≫ for n = 1:1:200
      if rem(n,10) == 0
         y(n)
      end;
end;
```

    De eerste regel is de definitie van een herhaal-lus. In de regel `if rem(n,10) == 0` wordt gecontroleerd of een getal `n` een 10-voud is. De $==$ betekent: is gelijk aan. Zo heb je ook: $<, >, <=, >=$ en $\tilde{}=$ die respectievelijk kleiner dan, groter dan, kleiner of gelijk, groter of gelijk en ongelijk betekenen. Het `rem`–statement bepaalt de restwaarde van de deling van het getal `n` door 10. In algemene termen: `rem(g,d)` geeft als resultaat de rest van de deling $\frac{g}{d}$. Het commando `y(n)` drukt het n^e element uit array y af.

    Er staat twee maal een `end` commando. De eerste `end` beeindigt het if-statement en de tweede het for-statement. Het nadeel van deze methode is dat bij het afdrukken van de resultaten ook het woord `ans` wordt afgedrukt. Dit geeft een erg onrustig beeld en bovendien past nu nog niet alles op een scherm. Om de resultaten netjes af te drukken kan je gebruik maken van het commando `disp`. `disp` toont de waarde van een variabele op het scherm zonder de naam van de variabele weer te geven.

12. Geef eerst het commando

    ≫ format compact

    Gebruik dan de toets met het ↑–teken om het eerste commando van onderdeel (11) achter de ≫–prompt te krijgen. Druk op ENTER. Gebruik de ↑–toets ook om het tweede commando te plaatsen. Druk weer op ENTER.
    Vervolgens tik je in:

```
      disp([n y(n)])
   end;
end;
```

    Het resultaat is dat de waarden n en $\sqrt{n}$ op één regel afgedrukt worden.

Het maken van grafieken

Matlab kan, naast gebruik als rekenmachine, ook heel goed gebruikt worden als grafisch-hulpmiddel. Matlab biedt de mogelijkheid op een eenvoudige manier 2-D of 3-D grafieken te maken. Het commando `plot(y)`, waarin `y` een vector is, geeft de grafiek van de elementen van `y` uitgezet tegen de indices van de vector. Het commando met twee parameters `plot(x,y)` geeft als resultaat de grafiek waarin y tegen x wordt uitgezet.

13. Geef het commando `plot(y)`

    In het vervolg zullen we van de functie $sin(x) + \frac{1}{2}$ de grafiek tekenen en de nulpunten berekenen. We zullen het programma voor deze berekeningen stap voor stap op gaan bouwen.

14. Let op: het Figure Window kan zich achter het MATLAB Command Window bevinden. Geef de volgende commando's

```
≫for n = 0:1:100
   x(n+1) = n * ( pi /100 );
   y(n+1) = sin( x(n+1) );
end;
≫ plot(y)
```

    Merk op dat het resultaat niet overeenkomt met je verwachtingen. De reden is dat de array $y(n)$ voor $n \in [102, 200]$ nog de vorige waarden bevat. De afmeting van de array staat nog steeds op 200. Om dit te voorkomen is het goed om het `clear` commando te gebruiken. Hiermee wordt de inhoud van een variabele of array verwijderd.

15. Tik in:

```
≫ clear y
≫ y = sin(x) + 0.5;
```

Hiermee wordt de vektor y gevuld zonder herhaal lus te gebruiken.

```
≫ plot(y)
```

Merk op dat nu langs de x-as de index staat uitgezet. Om y tegen x uit te zetten geven we

```
≫ plot(x,y)
```

Met behulp van de commando's `title`, `xlabel` en `ylabel` zijn grafieken van labels te voorzien. Om een functie te plotten kan ook gebruik gemaakt worden van het volgende commando

```
≫ fplot('sin(x)+0.5',[0 2*pi],'r')
```

16. Tik de volgende commando's in:

```
≫title('De sinus+0.5-functie')
≫xlabel('0 <= x <= 2*pi')
≫ylabel('y = sin(x)+0.5')
```

Het is ook mogelijk meer dan één grafiek in één plaatje te tekenen.

17. Tik in:

```
≫ z = cos(x);
≫ plot(x,y,'-r',x,z,':g')
```

Hierbij hebben de symbolen tussen de quotes de volgende betekenis: - geeft een doorgetrokken lijn en : een stippellijn, r (g) geeft een rode (groene) afbeelding. Andere mogelijkheden zijn te achterhalen met het `help plot` commando. De figuur kan zwart wit afgedrukt worden door het commando `print` te geven.

Het maken van een programma

Een Matlab-programma wordt gemaakt met behulp van een editor. Het bestand waarin het programma wordt opgeslagen **moet** als extensie **m** krijgen, bijvoor-

beeld `oefen.m`. Het programma wordt vervolgens gestart door, achter de Matlab–prompt, `oefen` in te tikken.

18. De editor kan als volgt aangeroepen worden: klik op de File button van het MATLAB Command Window, selecteer New, en dan M-file. Je komt dan in de editor. Open een nieuwe file.

    Tik het onderstaande programma in:

```
% Dit is een programma dat gebruikt wordt bij de Matlab
% introductie van het vak wi1149. Een % teken aan het begin
% van de regel, geeft aan dat de rest van de regel commentaar is.

clear x y

% vullen van x en y

for n = 0:1:100
   x(n+1) = n * (2*pi / 100);
end

y = sin(x)+0.5;

% het plotten van de functie

figure(1)
plot(x,y)
title(' De sinus+0.5 functie')
xlabel(' 0 <= x <= 2*pi ')
ylabel(' y = sin(x) + 0.5')
```

    Save de file als `oefen.m` en verlaat de editor.

19. Tik in het Matlab window in:

```
≫ help oefen
```

    Dit geeft de eerste commentaar regels van de file oefen.m

```
≫ oefen
```

    Hiermee wordt het programma uitgevoerd. Een bestaande file kan geopend worden via de File button van het MATLAB Command Window.

Functies

In Matlab is het ook mogelijk functies te gebruiken. Functies worden los van het hoofdprogramma gemaakt. Iedere functie komt in een aparte file. Waarom functies?

- handig als je ingewikkelde formules meerdere malen wilt uitrekenen,
- een functie kan in meerdere programma's gebruikt worden,
- het hoofdprogramma wordt leesbaarder als op de plaats van een ingewikkelde formule een eenvoudige functie met een geschikt gekozen naam staat,
- functies worden gebruikt bij integratie (nulpunt etc.) routines van Matlab.

We gaan nu een functie maken die van een gegeven vector x de vector $sin(x) + 0.5$ berekent.

20. Maak met behulp van de editor een file aan met de naam `sinus.m` die er als volgt uit ziet:

```
function y = sinus(x)
   y = sin(x)+0.5;
```

De functie `sinus` hangt af van de variabele x. Deze variabele en ook y zijn alleen locaal (dat wil zeggen in de functie zelf) bekend en dus niet in het hoofdprogramma.

21. Vervang in het hoofdprogramma (oefen.m) de regel `y = sin(x)+0.5;` door `y = sinus(x);` en voer het programma opnieuw uit.

Bepalen van een nulpunt

Voor een aantal wiskundige operaties zijn Matlab commando's beschikbaar. Voorbeelden hiervan zijn: het vinden van nulpunten, integratie, oplossen van differentiaalvergelijkingen etc. Bij deze operaties worden vaak functies gebruikt. We zullen dit illustreren aan de hand van het zoeken van een nulpunt van de functie $sin(x) + \frac{1}{2}$.

Het bepalen van een nulpunt gaat met behulp van het commando `fzero('sinus',3)`. Het eerst argument bevat de naam van de functie (let op de quotes) en het tweede argument bevat de startwaarde.

22. Voeg aan het programma oefen.m de volgende regels toe:

```
hold on
nulpunt = fzero('sinus',3);
disp([nulpunt sinus(nulpunt)]);
plot(nulpunt, sinus(nulpunt), '*r')
hold off
```

en voer het programma uit. Met commando `hold on` blijft de eerste grafiek bewaard. Merk op dat het nulpunt gelijk is aan $1\frac{1}{6}\pi$

23. Tik in:

    ```
    ≫ (1+1/6)*pi
    ```

    Het is soms gewenst om in een programma invoer van het scherm te ontvangen. Dit kan met behulp van het `input`-commando.

24. Vervang in oefen.m de regel `nulpunt = fzero('sinus',3);` door:

    ```
    start = input('Geef de startwaarde voor het nulpunt: ');
    nulpunt = fzero('sinus',start);
    ```

    en voer het programma uit. Op het scherm verschijnt dan:

    Geef de startwaarde voor het nulpunt:

    Tik in 3 en bekijk het resultaat. Doe dit nog eens maar dan voor de startwaarde 5.

    Hiermee zijn we aan het einde gekomen van de introductie. Reacties, op- of aanmerkingen zijn welkom.

25. Beeindig de practicumsessie via *start* | *logout* en klik daarna op *shutdown* | *reboot*.