Computergestützte Mathematik zur Linearen Algebra

Skripte und Funktionen

Achim Schädle

Übungsleiter: Lennart Jansen

Tutoren: Marina Fischer, Kerstin Ignatzy, Narin Konar
Pascal Kuhn, Nils Sänger, Tran Dinh

13. November 2014

Skripte

- Ein Matlabskript ist eine Text Datei mit der Endung `.m`, ein sogenanntes m-file. *`skriptname`*`.m`, die Matlabbefehle enthält.
- Die Eingabe des Names dieser Datein führt dazu, dasss die in der Datei aufgeführten Befehle nacheinander ausgeführt werden.

MATLAB Skripte: Beispiel

Inhalt der Datei `plotsinusxdurchx.m`

```
% Plotten der Funktion sin(x)/x in [-5pi,5pi]
% sinc

x = linspace(-5*pi,5*pi,1000);
y = 5*sin(x)./x;

figure(7)
plot(x,y)
axis equal
axis([-5*pi 5*pi -1 1])
```

Aufruf im Command Window durch `plotsinusxdurchx`

Funktionen I

- Einfache Funktionen mit nur einem Rückgabewert @
- Beispiele:
  $q = @(x, N)\ x\hat{}N + x - 1;$
  $v = @(x1, x2)\ [x1,\ -x2];$

Funktionen II

Aufbau einer Funktion:

1. Definition der Funktion (abgespeichert in *`funname`*`.m`)

   ```
   function [out_1,out_2,...,out_k] = ...
            funname(in_1,in_2,...,in_m)
   ```

   Die Listen der Ein- oder Ausgabevariablen können auch leer sein.
2. Kommentarzeilen (was tut die Funktion, welche Ein- und Ausgabevariablen muss der Benutzer verwenden).
   - Der erste zusammenhängende Block von Kommentarzeilen wird bei `help funname` angezeigt.
   - Die erste Zeile der Kommentarzeilen wird bei `lookfor` durchsucht.

Funktionen in **MATLAB**

Wir kennen schon diverse **MATLAB**-Funktionen:

- `sin, cos, diag, size, ...`

diese wurden meist durch `y = funktion(x)` aufgerufen, wobei `x` eine Eingabevariable und `y` eine Ausgabevariable war.

- was in der Funktion passiert, ist für uns unsichtbar (black box)
- der Funktionsaufruf hinterlässt im Unterschied zu Skripten keine Spuren (z.B. Hilfsvariablen) im Arbeitsspeicher.

MATLAB kann nicht nur durch Skripte sondern auch durch Funktionen ergänzt werden!

MATLAB-eigene Funktionen

In **MATLAB** gibt es so genannte built-in Funktionen, die besonders schnell laufen und nicht in **MATLAB** programmiert sind.

Beispiel: `type diag`

Es gibt aber auch viele in **MATLAB** programmierte Funktionen:

Beispiel: `type flipud`

Funktionen sind wie Skripte Textdateien, deren Name mit `.m` endet.

MATLAB Funktionen: Beispiel I

endliche geometrische Reihe als Funktion, gespeichert in `geomr.m`

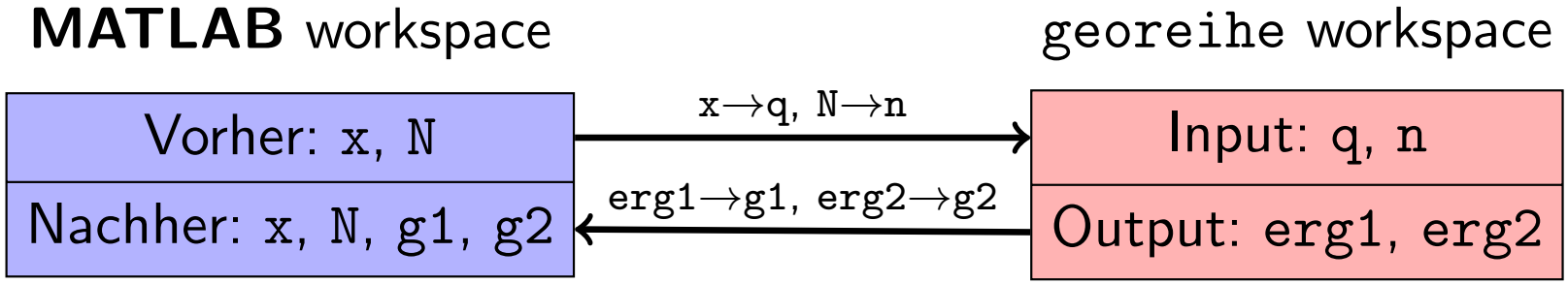
```
function [erg1, erg2] = geomr(q,n);
% function [erg1, erg2] = geomr(q,n);
% Berechnung der Summe q^0+q^1+ ... q^n
% auf zwei Arten

Nvec = [0:n];
erg1 = sum(q.^Nvec);
erg2 = (1-q^(n+1))/(1-q);
```

Aufruf: `x=1.2; N=5; [g1,g2]=geomr(x,N); whos`

MATLAB Funktionen: Work Spaces

Aufruf `[g1,g2] = geosum(x,N)`



- **MATLAB** legt "lokale" Kopien der Eingabedaten an
- **MATLAB** "vergisst" bei Verlassen der Funktion alle lokalen Variablen
- Nur Variablen, die als Rückgabewert gekennzeichnet sind, werden "nach draußen" weitergegeben

Der Workspace kann mit `whos` angezeigt werden

MATLAB Funktionen: Beispiel II

Inhalt der Datei PlotFunktion.m

```
% Plotten einer Funktion f im Intervall [a,b]
function [] = PlotFunktion(f,a,b)

x = linspace(a,b,1000);
y = feval(f,x);

plot(x,y)
axis tight
```

- Drei Input- und kein Outputargument
- Aufruf im Command Window durch
  - `PlotFunktion(@sin,-2*pi,2*pi)` oder
  - `PlotFunktion('sin',-2*pi,2*pi)` oder
  - `PlotFunktion(f,-2,2)` (mit `f` = `@(x)x.^2`)

Regeln für Funktionen I

- Für Funktionsnamen gelten dieselben Regeln wie für Variablen
- Eine Funktion endet nachdem die letzte Zeile ausgeführt wurde oder durch ein `return`-Kommando
- Eine Funktion kann durch `warning('eine Warnung')` eine Warnung geben. Im Unterschied zu `disp('eine Warnung')` können durch `warning on` bzw. `warning off` Warnungen global an- bzw. ausgeschaltet werden
- **MATLAB**-Funktionen können Skripte oder andere **MATLAB**-Funktionen aufrufen.

Regeln für Funktionen II

- In einem m-File können mehrere Funktionen abgespeichert werden. Solche Unterfunktionen werden an die erste Funktion (die Hauptfunktion) angehängt. Sie beginnen mit einem Standard `function`-Kommando und genügen denselben Regeln wie vorher.
- Unterfunktionen können durch die Hauptfunktion oder durch andere Unterfunktionen aus demselben m-File aufgerufen werden, nicht aber aus anderen Funktionen oder Skripten.
- Unterfunktionen haben ihren eigenen lokalen Arbeitsspeicher.
- Die Reihenfolge von Unterfunktionen spielt keine Rolle.