Computergestützte Mathematik zur Linearen Algebra

Programmsteuerung I

Achim Schädle

Übungsleiter: Lennart Jansen

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

13. November 2014

Steuerung von Programmabläufen

MATLAB-Skripte und Funktionen enthalten Abfolgen von Befehlen, die Zeile für Zeile ausgewertet werden.

Frage:

- Ist es möglich sich wiederholende Befehle zusammenzufassen?

Schleifen

Bisher haben wir **MATLAB**-Befehle nacheinander im Command Window eingegeben oder in einer `m`-Datei gespeichert

- **MATLAB** arbeitet die Befehle nacheinander ab
- Mehrfach auszuführende Befehle, müssen mehrfach eingegeben werden

Schleifen (Zusammenfassung sich wiederholender Befehle)

- Grundlegende Unterscheidung in zwei Arten

for-Schleifen

```
for var = mat
   Matlab-Befehle
end
```

while-Schleifen ($\rightarrow$ nächste Woche)

```
while Bedingung
   Matlab-Befehle
end
```

for-Schleifen I

Allgemeine Form

```
for var = mat
    Matlab-Befehle
end
```

- Im *i*-ten Durchlauf der Schleife gilt `var = mat(:,i)`, d.h. `var` enthält die *i*-te Spalte der Matrix `mat`
- Die Schleife wird wiederholt, bis alle Spalten von `mat` durchlaufen wurden (d.h., `size(mat,2)`-mal)

Häufige Varianten

- `mat = 1:n` (`var = i` im *i*-ten Durchlauf)
- `mat = anfang:inkrenent:ende`
- `mat = linspace(a,b,N)`

for-Schleifen II: Beispiel

Aufgabe: Berechne zu gegebenem $q \in (0,1)$ und $n \in \mathbb{N}$

$$\sum_{k=0}^{n} q^k$$

```
function sum = geosum(q,n)

sum = 0;
for k = 0:n
    sum = sum + q^k;
end
```

Anwendungen von `for`-Schleifen

```
for var = mat
   Matlab-Befehle
end
```

- Es muss bekannt sein, wieoft die Befehle wiederholt werden und u. U. welche Werte `var` in jedem Durchlauf enthalten soll

- for-Schleifen können in **MATLAB** häufig durch vektorisierte Ausdrücke ersetzt werden
- Programmierstil: die Schleifenvariable sollte nicht in der Schleife verändert werden (obwohl dies erlaubt ist)
- verwenden Sie nicht i oder j als Schleifenvariablen

Anwendungen von `for`-Schleifen

```
for var = mat
   Matlab-Befehle
end
```

- Es muss bekannt sein, wieoft die Befehle wiederholt werden und u. U. welche Werte `var` in jedem Durchlauf enthalten soll

- for-Schleifen können in **MATLAB** häufig durch vektorisierte Ausdrücke ersetzt werden
- Programmierstil: die Schleifenvariable sollte nicht in der Schleife verändert werden (obwohl dies erlaubt ist)
- verwenden Sie nicht i oder j als Schleifenvariablen

Vektorisierung von `for`-Schleifen

Beispiel: $y(n) = \sin(\frac{\pi n}{10}), n = 1, \ldots, 10$

```
for n = 1:10
   y(n) = sin(n*pi/10);
end
```

Diese `for`-Schleife kann ersetzt werden durch

```
n = 1:10
y = sin(n*pi/10);
```

```
y = sin([1:10]*pi/10);
```

Vorteile

- Kürzerer Code, oftmals schnellere Ausführung
- Einfacher lesbar, weniger Schreibarbeit

Bemerkungen zu Dimensionen

```
for n=1:10
    y(n) = sin(n*pi/10);
end
```

Beobachtung: Dimension von y wächst bei jedem Schleifendurchlauf, **MATLAB** muss daher die Dimension in jedem Durchlauf anpassen

$\longrightarrow$ Teuer bei vielen Durchläufen

Besser: Initialisierung vor der Schleife:

```
y = zeros(1,n);
for n=1:10
    y(n) = sin(n*pi/10);
end
```

Bemerkungen zu Dimensionen

```
for n=1:10
    y(n) = sin(n*pi/10);
end
```

Beobachtung: Dimension von `y` wächst bei jedem Schleifendurchlauf, **MATLAB** muss daher die Dimension in jedem Durchlauf anpassen

$\longrightarrow$ Teuer bei vielen Durchläufen

Besser: Initialisierung vor der Schleife:

```
y = zeros(1,n);
for n=1:10
    y(n) = sin(n*pi/10);
end
```

Bemerkungen zu Dimensionen

```
for n=1:10
    y(n) = sin(n*pi/10);
end
```

Beobachtung: Dimension von `y` wächst bei jedem Schleifendurchlauf, **MATLAB** muss daher die Dimension in jedem Durchlauf anpassen

$\longrightarrow$ Teuer bei vielen Durchläufen

Besser: Initialisierung vor der Schleife:

```
y = zeros(1,n);
for n=1:10
    y(n) = sin(n*pi/10);
end
```