Weiterbildendes Studium Ingenieurmathematik – WS 98/99

Optimierung mit MATLAB

H.-J. Oberle, G. Opfer, S. Fredenhagen

Arbeitsblatt 4

1. Erstes Beispiel

Das Verfahren der konjugierten Gradienten wurde in der Datei `cg.m` implementiert. Ein einfaches Beispiel der Dimension $n = 500$ ist die Minimierung von

$$f(x) = x_1^2 + 2x_1x_{100} + 2x_{100}^2 + x_{500}^2.$$

Hier hat die Matrix $\boldsymbol{A}$ nur fünf von Null verschiedene Einträge. Da man für das cg-Verfahren nur das Produkt $\boldsymbol{Av}$ für verschiedene Vektoren $\boldsymbol{v}$ benötigt, genügt es dieses zu programmieren. Für diese Funktion findet man es in der Datei `Amalv.m`. Mit

```
>> startcg
```

kann man dieses Beispiel ausprobieren.

2. Matrizen

In Matlab gibt es Befehle, um Vektoren, bzw. Matrizen, einfach zu erzeugen. Einige werden in den Dateien `startcg.m` und `Amalv.m` verwendet. Mit

```
>> e=ones(4,1)
```

erhält man einen Vektor der Dimension vier mit Einsen. Mit

```
>> Z=zeros(4,5)
```

wird die Nullmatrix der Dimension 4×5 erzeugt. Es gibt auch die Möglichkeit auf einzelne Elemente einer Matrix zuzugreifen. Mit

```
>> e(2,1)=7
```

ändert man das zweite Element des Vektors $\boldsymbol{e}$. Auf die gleiche Weise ändert man mit

```
>> Z(3,4)=2
```

bei der Matrix das Element $z_{3,4}$. Um Zeilen oder Spalten einen anderen Wert zuzuweisen, kann man

```
>> Z(1,:)=3
```

oder

```
>> Z(:,2)=e
```

verwenden. Eine größere Matrix kann man mit

```
>> [Z,5*e]
```

zusammensetzen. Die Dimension einer Matrix kann man mit

```
>> size(Z)
```

ausgeben. Mit

```
>> e(length(e))
```

erhält man das letzte Element von $\boldsymbol{e}$. Mit

```
>> max(e)
```

erhält man das größte und mit

```
>> min(e)
```

das kleinste Element von $\boldsymbol{e}$.

3. Aufgabe 4.1: *Modellprobelm*

Bei Beispiel 5.11 wird eine partielle Differentialgleichung, die sogenannte *Potentialgleichung*

$$\Delta u := u_{xx} + u_{yy}, \text{ im Inneren der Platte;} u = u_0 \text{ am Rand,}$$

diskrtisiert. Dieses Beispiel kann man mit

```
>> modell.m
```

aufrufen. Es benutzt die Dateien `prodd.m`, `residuum.m` und `scalar.m`. In der Datei `modell.m` findet man die gewählten Randbedigungen und kann die Feinheit der Diskretisierung mit dem Parameter `n` steuern.

4. Aufgabe 4.2: *Vandermonde-Matrix*

Das Beispiel 5.10 aus dem Skript ist in der Datei `vander.m` implementiert. Mit

```
>> startvan(3)
```

kann man kann man das Beispiel für $n = 3$ aufrufen.

a) Probieren Sie es für verschiedene (nicht zu große) Dimensionen aus.

b) Ändern Sie die Ausgabe in der Datei `startvan.m` ab, und probieren Sie es auch für größere Dimensionen aus.

(i) Geben Sie nur die ersten beiden und das letzte Element des Lösungsvektors aus.

(ii) Geben Sie nur das größte und das kleinste Element des Lösungsvektors aus.

(iii) Geben Sie das Residuum

$$\boldsymbol{r}^{\mathrm{T}}\boldsymbol{r} \quad \text{mit} \quad \boldsymbol{r} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}$$

aus.

5. Aufgabe 4.3: Bedeutung der Eigenwerte

a) In der Datei `ewert.m` ist ein weiteres Beispiel angegeben. Was passiert, wenn man es mit

```
>> startew
```

aufruft? In diesem Fall ist die Matrix $\boldsymbol{A}$ explizit gespeichert, damit man ihre Eigenwerte mit

```
>> eig(A)
```

berechnen kann.

b) Mit

```
>> startew2
```

wird die Datei `ewert2.m` aufgerufen. Hier kann man die Abbruchgenauigkeit mit

```
>> startew2(0.0001)
```

vorgegeben werden. Variieren Sie die Abruchgenauigkeit und beobachten Sie die Zahl der Iterationsschritte und die Lösung.

c) Sehen Sie sich die Dateien `ewert.m` und `ewert2.m` an. Was passiert, wenn man $\boldsymbol{D}$ anders wählt?