FH AACHEN
UNIVERSITY OF APPLIED SCIENCES

Fachbereich Elektrotechnik
und Informationstechnik

Prof. Georg Hoever

SS 2022

1. Praktikum zur Höhere Mathematik 2 für (Wirtschafts-)Informatik

Ziel dieses Praktikums ist eine Implementierung des Gradientenverfahrens mit Schrittweitensteuerung.

1. Aufgabe

Um bequem mit Vektoren $\vec{x} \in \mathbb{R}^n$ arbeiten zu können, soll eine Klasse `CMyVektor` implementiert werden:

- Überlegen Sie sich, durch welche(n) Datentyp(en) Sie die Informationen speichern, z.B. die Dimension als Integer und die Werte in einem `double`-Array; Sie können auch die Standardklasse `vector<double>` der C++-Standard-Template-Library nutzen.

  Implementieren Sie die Informationen als private Attribute.

- Implementieren Sie (public-)Methoden, um
  - einen Vektor einer bestimmten Dimension anzulegen,
  - die Dimension eines Vektors zurückzugeben,
  - eine bestimmte Komponente des Vektors zu setzen,
  - eine bestimme Komponente des Vektors zurückzugeben.

  Tipp: Sie können beispielsweise elegant den Indexoperator `[]` oder Klammeroperator `()` überladen.

- Implementieren Sie eine (public-)Methode, die die Länge des Vektors zurückgibt.

Implementieren Sie ferner zwei überladene Operator-Funktionen

```
CMyVektor operator+(CMyVektor a, CMyVektor b)
CMyVektor operator*(double lambda, CMyVektor a),
```

die eine Vektor-Addition und eine skalare Multiplikation realisieren.

2. Aufgabe

Zu einer Funktion $f : \mathbb{R}^n \to \mathbb{R}$ soll der Gradient an einer Stelle $\vec{x} \in \mathbb{R}^n$ berechnet werden:

- Implementieren Sie eine Funktion

  `CMyVektor gradient(CMyVektor x, double (*funktion)(CMyVektor x))`,

  der man im ersten Parameter die Stelle $\vec{x}$ und im zweiten Parameter die Funktion f als Funktionspointer übergibt, und die den Gradienten $\vec{g} = \operatorname{grad} f(\vec{x})$ numerisch durch

  $$g_i = \frac{f(x_1, \ldots, x_{i-1}, x_i + h, x_{i+1} \ldots, x_n) - f(x_1, \ldots, x_n)}{h}$$

  zu festem $h = 10^{-8}$ berechnet.

3. Aufgabe

Zu einer Funktion $f : \mathbb{R}^n \to \mathbb{R}$ soll ausgehend von einer Stelle $\vec{x} \in \mathbb{R}^n$ das Gradientenverfahren zur Maximierung von f mit der folgenden Schrittweitensteuerung durchgeführt werden:

1. Man geht testweise einen Schritt in Richtung des Gradienten mit der aktuellen Schrittweite.
2. Ist der Funktionswert nach dem Testschritt von 1. kleiner oder gleich dem aktuellen Funktionswert, so halbiert man die Schrittweite.

   Dies führt man solange durch, bis man eine Stelle mit größerem Funktionswert gefunden hat.
3. Ist der Funktionswert nach dem Testschritt von 1. größer als der aktuelle Funktionswert, so testet man – um gegebenenfalls schneller voran zu kommen – eine doppelt so große Schrittweite und arbeitet mit der Schrittweite weiter, die den größeren Funktionswert liefert.

Dieses Verfahren soll solange durchgführt werden, bis $\|\operatorname{grad} f(\vec{x})\| < 10^{-5}$ ist, oder bis 25 Schritte gemacht wurden.

- Implementieren Sie das entsprechende Verfahren.

  Nutzen Sie wieder einen Funktions-Pointer zur Angabe der zu maximierenden Funktion. Neben der Startstelle $\vec{x}$ soll die Start-Schrittweite λ optionales Argument mit default-Wert `1.0` sein.

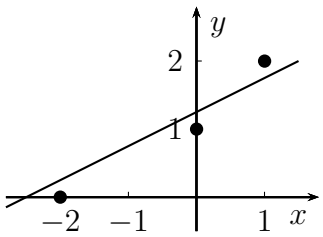
- Testen Sie das Verfahren an den folgenden Beispielen:
  - $f : \mathbb{R}^2 \to \mathbb{R}$, $f(x, y) = \sin(xy) + \sin x + \cos y$, Startstelle $\vec{x} = (0.2, -2.1)^T$, default-Start-Schrittweite,
  - $g : \mathbb{R}^3 \to \mathbb{R}$, $g(x_1, x_2, x_3) = -(2x_1^2 - 2x_1x_2 + x_2^2 + x_3^2 - 2x_1 - 4x_3)$, Startstelle $\vec{x} = (0, 0, 0)^T$, Start-Schrittweite $\lambda = 0.1$.

Für Interessierte

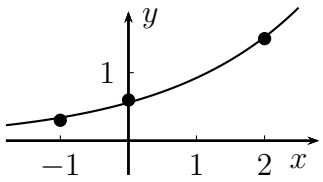
Häufig sucht man Parameter zu einer Funktion f, die „möglichst gut“ durch gemessene Daten (x_k, y_k), $k = 1, \ldots, N$, führen soll.

Beispiele:

1. Gesucht sind die Parameter m und a einer linearen Funktion $f(x) = mx + a$, die möglichst gut zu den Punkten $(-2, 0)$, $(0, 1)$ und $(1, 2)$ passen soll.



2. Gesucht sind die Parameter c und λ einer Exponentialfunktion $f(x) = c \cdot \mathrm{e}^{\lambda x}$, die möglichst gut zu den Punkten $(-1, 0.3)$, $(0, 0.6)$ und $(2, 1.5)$ passen soll (oder zu den RKI-Angaben der täglichen Corona-Neuinfektionen).



Als Maß für „möglichst gut“ nimmt man üblicherweie die *least-squares*-Anpassung, d.h., man minimiert die Summe s der quadratischen Abstände:

$$s \;=\; \sum_{k=1}^{N} (f(x_k) - y_k)^2. \qquad (*)$$

Die Summe s kann man als Funktion der Parameter auffassen und mittels des Gradientenverfahrens minimieren (bzw. die Funktion $-s$ maximieren).

- Implementieren Sie zu den Beispielen 1 und 2 entsprechende Funktionen $s(a, m)$ bzw. $s(c, \lambda)$ und bestimmen Sie die optimalen Parameter.

  Als Startwerte können Sie beispielsweise $a = 1$ und $m = 0.1$ bzw. $c = 1$ und $\lambda = 0.1$ wählen.

- Vergleichen Sie Ihre Ergebnisse zum ersten Beispiel mit dem Resultat der entsprechenden Aufgabe auf dem Übungsblatt.

- Vergleichen Sie Ihre Ergebnisse mit einer linearen bzw. exponentiellen Anpassung, die z.B. Excel automatisch berechnet.

  Sie sollten beobachten, dass die Werte bei der linearen Anpassung mit Ihren berechneten Werten ziemlich übereinstimmen, es aber bei den exponentiellen Anpassungen deutliche Differenzen gibt; zu den Hintergründen s. unten.

Weitere Hintergründe:

Die Optimalstellenbestimmung bei einer linearen Ausgleichsfunktion kann man auch analytisch berechnen, vgl. die entsprechende Übungsaufgabe. Bei einer exponentiellen Ausgleichsfunktion ist eine analytische Bestimmung für die Minimierung von $(*)$ nicht möglich, da man analytisch das entsprechende Gleichungssystem $\operatorname{grad} s = 0$ nicht auflösen kann.

Bei einer exponentiellen Ausgleichsfunktion führt man allerdings üblicherweise keine *least-squares*-Anpassung der Werte wie bei $(*)$ durch, sondern eine *least-squares*-Anpassung der logarithmierten Werte, also von

$$s \;=\; \sum_{k=1}^{N} (\ln(f(x_k)) - \ln(y_k))^2.$$

Bei $f(x) = c \cdot \mathrm{e}^{\lambda x}$ ist

$$g(x) \;=\; \ln(f(x)) \;=\; \ln(c \cdot \mathrm{e}^{\lambda x}) \;=\; \ln(c) + \ln(\mathrm{e}^{\lambda x}) \;=\; \ln c + \lambda x,$$

was wieder auf ein lineares Ausgleichsproblem führt.

Dies erklärt, warum Ihre exponentielle Anpassung oben (Beispiel 2) andere Werte liefert als z.B. Excel. Sie können nachrechnen, dass Ihre Paramterwerte bei der Summe s von $(*)$ tatsächlich einen geringeren Wert liefert als bei den z.B. von Excel berechneten Werten.

Wenn Sie die lineare Anpassung entsprechend Beispiel 1 auf die logarithmierten Werte von Beispiel 2 anwenden(also die Punkte $(-1, \ln(0.3))$, $(0, \ln(0.6))$ und $(2, \ln(1.5))$, dann erhalten Sie optimale Parameter a und m, wobei für die zu einer exponentiellen Anpassung z.B. von Excel berechneten Werte c und λ gilt: $\ln(c) \approx a$, also $c \approx \mathrm{e}^a$, und $m \approx \lambda$.

Bei den bisherigen Beispielen könnte man den Gradienten von s bzgl. der Parameter analytisch berechnen bzw. von Hand implementieren. Mit der gleichen Technik wie oben kann man aber nun auch Kurven anpassen, die man numerisch berechnet, z.B. als Lösung einer Differenzialgleichung (Differenzialgleichungen werden in Mathe 2 im Mai behandelt). Hier ist dann keine analytische Berechnung des Gradienten mehr möglich, so dass die numerische Bestimmung des Gradienten wie in diesem Praktikum dargestellt, notwendig ist.

Ein Beispiel hierzu ist das sogenannte SIR-Modell, mit dem man den Verlauf einer Epidemie beschreiben kann; dabei gibt es Parameter wie Startwerte, die Dauer, in der ein Infizierter weitere Personen anstecken kann, und die Wahrscheinlichkeit, dass pro Zeiteinheit tatsächlich eine Infektion geschieht (diese kann man durch Kontaktbeschränkungen beeinflussen). Die aktuellen Parameterwerte kann man versuchen, durch entsprechende Anpassung der Lösungen der Differenzialgleichung an die RKI-Daten zu ermitteln.