Numerik

6. Dezember 2022

Bereits im *Aufgabenblatt 3* haben Sie sich mit der Bestimmung von Extremwerte in mathematischen Gleichungen beschäftigt. Sie mussten dazu die 1. bzw. 2. Ableitung bestimmen. Das haben Sie für den damaligen Anwendungsfall noch mit Stift und Papier erledigt. Diese Vorgehensweise ist allerdings nicht unbedingt zielführend und widerspricht dem Wunsch, ein Programm zu entwickeln, welches keine zusätzlichen, schriftlichen Nebenrechnungen benötigt. Symbolisches Umformen der Gleichungen und Bestimmen der Ableitungen ist in der Programmiersprache `C` allerdings nicht so ohne Weiteres möglich. Zur Lösung solcher Anwendungsfälle bietet die Numerik eine Vielzahl an Methoden, mit denen Sie bestimmte Punkte einer Gleichung wie Extremwerte oder Nullstellen bestimmen können. In diesem Aufgabenblatt sollen Sie einmal selbst so einen numerischen Algorithmus programmieren. Verwenden Sie dafür Funktionen!

Lösung zu Aufgabe 7.0

Schwerpunkt liegt auf dem Umgang mit Funktionen und Implementierung eines vorgeschriebenen Algorithmus.

Aufgabe	Bewertung
Abgabe: Hauptprogamm + Ausgabe im Terminal	1
Abgabe: Funktionen für Ableitungen	1
Abgabe: Funktionen Bisektionsverfahren	2
Abgabe: Minimum oder Maximum	1
Vertiefung: Polynom + Ableitung; Abfrage zum Startintervall	1
Vertiefung: Funktionen mit Zeigern	1
Summe	7

Vorbereitung

Aktualisieren Sie Ihr Template-Verzeichnis. Wie in den vorherigen Aufgabenblättern finden Sie ein neues Template im öffentlichen Repository, welches Sie mittels des bekannten Befehls in Ihr lokales Verzeichnis übertragen können:

```
$>  git pull upstream main --no-rebase
```

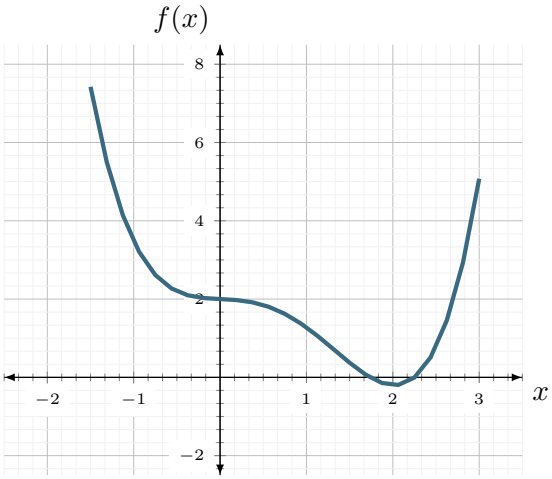
Bei korrekter Ausführung sollten Sie den Ordner *sheet7_temps* in Ihrem lokalen Verzeichnis finden. Beachten Sie, dass Sie das gegebene *Makefile* nach Belieben an Ihr Programm anpassen können.

Aufgabe 7.1: Abgabe

Die Bearbeitung und Lösung dieser Aufgabe ist zum Erwerb des Klausurbonus erforderlich. Die Lösung wird dem Tutor vorgestellt und bildet die Bewertungsgrundlage für die Testate. Übertragen Sie Ihre Ergebnisse 2 Tage vor dem Meeting mit Ihrem Tutor in Ihr Gruppen-Repository. Achten Sie darauf, dass Sie keine Binär-Dateien hochladen.

Minimum mit Bisektionsverfahren

Numerische Methoden sind oft hilfreich beim Bestimmen von z.B. Nullstellen oder Extremwerten. Im *Aufgabenblatt 3* haben Sie bereits einen Extremwert mit Hilfe einer analytischen Nebenrechnung gefunden. Die Schritte, die Sie vor wenigen Wochen mit Stift und Papier erledigt haben, sollen Sie nun einmal in einem Programm implementieren. Nutzen Sie dafür die Kenntnisse aus den Vorlesungen und verwenden Sie Funktionen. Es ist ebenfalls gestattet die bereits bekannten Funktionen aus der Vorlesung zu modifizieren.



Gegeben sind Messdaten, die mit einem Polynom approximiert werden können.

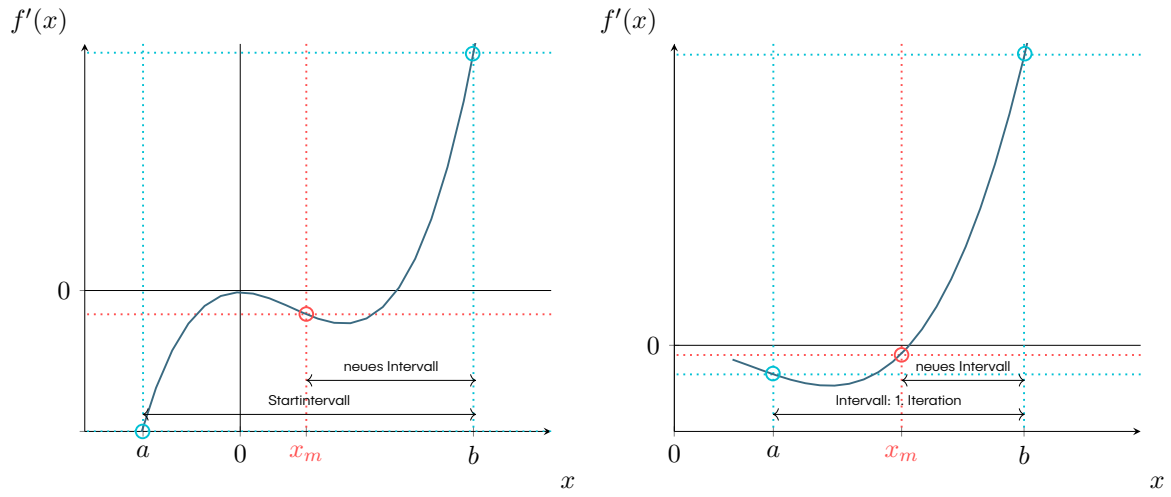
$$y = f(x) = \frac{3}{8}x^4 - x^3 - \frac{1}{10}x + 2$$
$$\text{mit } -1.5 \leq x \leq 3$$

Da das gegebene Polynom differenzierbar ist, sind Sie in der Lage den Extremwert mit den aus der Kurvendiskussion bekannten Methodiken zu bestimmen. Das soll allerdings nicht mit Stift und Papier geschehen.

Grundlegend ist es für die Ermittlung des Extremwertes nötig, die Nullstellen der ersten Ableitung $f'(x) \stackrel{!}{=} 0$ zu bestimmen. Da symbolisches Rechnen in C nicht so einfach möglich ist, bietet sich auch hier eine numerische Lösung an.

Eine gängige Methode für das Bestimmen von Nullstellen, hier das Bestimmen von Nullstellen einer Ableitung, ist das **Bisektionsverfahren**. Die Grundidee besteht darin, eine Nullstelle innerhalb eines Intervalls durch wiederholtes Halbieren einzugrenzen. Das geschieht solange bis eine hinreichend kleine Invervallgröße erreicht wurde. Sie können sich dabei die Eigenschaft zunutze machen, dass die Funktionswerte der einschließenden Intervallgrenzen ein unterschiedliches Vorzeichen besitzen, sofern die Nullstelle von dem resultierenden Intervall eingeschlossen wird. Zur Implementierung des Algorithmus sind folgende Schritte nötig, siehe ergänzend dazu Abb. 1:

1. Initialisierung eines Startintervalls $[a, b]$ in einem sinnvollen Definitionsbereich. Die Funktionswerte $f(a)$ und $f(b)$ an den Intervallgrenzen a und b müssen jeweils unterschiedliche Vorzeichen besitzen, ansonsten ist das Intervall falsch gewählt.
2. Halbieren des Intervalls, sodass sich zwei Teilintervalle $[a, x_m]$ und $[x_m, b]$ ergeben.
3. Überprüfen der Funktionswerte an den Grenzen der beiden Teilintervalle auf ein unterschiedliches Vorzeichen.
4. Wahl desjenigen Teilintervalls, dessen Funktionswerte an den Grenzen unterschiedliche



(a) Start (b) 1. Iteration

Abbildung 1: Grundprinzip Bisektionsverfahren. Beachten Sie, dass die Ableitung $f'(x)$ dargestellt ist.

Vorzeichen besitzen, wie in Abb. 1 dargestellt.

5. Wiederholung der Schritte 2. bis 4. solange bis das Intervall eine vorher definierte, hinreichend kleine Größe erreicht hat.

Schreiben Sie ein Programm, welches die Extremwerte des gegebenen Polynoms mit Hilfe des beschriebenen Bisektionsverfahrens berechnet. Nehmen Sie an, dass das Polynom nur einen Extremwert besitzt – die Koeffizienten wurden dementsprechend gewählt. Verwenden Sie zur Bearbeitung dieser Aufgabe Funktionen.

- Implementieren Sie die Funktionen `dpoly1()` und `dpoly2()`, mit denen Sie die jeweils 1. bzw. 2. Ableitung des Polynoms bestimmen können.
  Nutzen Sie dabei folgende Hinweise:
  - Ableitungen von Polynomen lassen sich mit folgenden mathematischen Gleichungen beschreiben:

$$f'(x) = \sum_{i=1}^{n} i \cdot a_i \cdot x^{i-1}$$
$$f''(x) = \sum_{i=2}^{n} i \cdot (i-1) \cdot a_i \cdot x^{i-2}$$

  - Orientieren Sie sich an dem in der Vorlesung gegebenen Beispiel zur Berechnung eines Polynoms.
- Berechnen Sie den Extremwert der Funktion. Bestimmen Sie dazu die Nullstelle der 1. Ableitung des gegebenen Polynoms mit Hilfe des Bisektionsverfahrens. Implementieren Sie das Verfahren in einer Funktion `bisection()`. Wählen Sie als Abbruchkriterium das Unterschreiten einer Intervallgröße von $\epsilon = 5 \cdot 10^{-3}$. Nehmen Sie für die gefundene Nullstelle die Mitte des resultierenden Intervalls an.
- Überprüfen Sie mit Hilfe der 2. Ableitung, ob es sich bei dem gefundenen Punkt um ein

Minimum oder Maximum handelt. Dazu müssen Sie das vorher ermittelte Argument am Extremwert Ihrer Funktion `dpoly2()` übergeben. Im Anschluss überprüfen Sie, ob der berechnete Funktionswert (der 2. Ableitung) größer oder kleiner als 0 ist und treffen dann eine Entscheidung.

- Lassen Sie als Ergebnisse den Extremwert sowie die Art des Extremums (Minimum oder Maximum) in gut lesbarer Form im Terminal ausgeben.

Sie können Ihr Ergebnis selbst überprüfen, indem Sie den Extremwert mit Stift und Papier ausrechnen und mit Ihrem Ergebnis im Terminal vergleichen.

Lösung zu Aufgabe 7.1

- Programm und Funktionen erklären lassen
- 2 Möglichkeiten Funktionen zu definieren: direkt in *extremwert.c* über `main ()` oder als extra Dateien (dafür müssen Header geschrieben und eingebunden werden)
- mögliche Fragen zu Funktionen: (siehe dazu Vorlesungsfolien)
  1. Allgemeine Syntax einer Funktion. Was bedeuten die einzelnen Einträge?
  2. Unterschied Call by Value und Call by Reference?
  3. Wie werden die Parameter in einer Funktion verarbeitet? (Lokal? Global? Rückgabewert?)
  4. Was ist bei der Reihenfolge von Funktionen zu beachten? (Funktionen müssen definiert werden, bevor sie benutzt werden können.)

Aufgabe 7.2 : Übung und Vertiefung

Der Inhalt dieser Aufgabe ist relevant für die Testatfragen. Die Aufgaben sollen bearbeitet werden, müssen aber nicht mit abgegeben werden. Die Fragen des Tutors können thematisch Bezug zu den Aufgaben dieses Abschnittes nehmen.

Funktionen clever gestalten

Im vorherigen Aufgabenteil haben Sie mehrere Funktionen implementiert, mit deren Hilfe Sie die Extremwerte eines Polynoms berechnen können. Verbessern Sie Ihre Implementierung, indem Sie folgene Punkte berücksichtigen:

1. Implementieren Sie eine Funktion `dpoly()`, die in der Lage ist, jede beliebige Ableitung von Polynomen beliebigen Grades zu berechenen. Testen Sie Ihr Hauptprogramm mit der neu implementierten Funktion. Diese sollte nun `poly()`, `dpoly1()` und `dpoly2()` ersetzen können.
2. Der Algorithmus für das Bisektionsverfahren setzt voraus, dass der Extremwert sich innerhalb des initialisierten Startintervalls befindet. In der Aufgabe wurde das Startintervall vorgegeben. Für gewöhnlich müssen Sie dieses selbst bestimmen. Es kann vorkommen, dass die gewählten Grenzen den Extremwert nicht einschließen (oder sogar mehrere Extremwerte enthalten sind). Als Fehlervermeidungsstrategie erweitern Sie Ihre Funktion `bisection()` mit einer Überprüfung, ob das Intervall einen möglichen Extremwert beinhaltet. Die Funktion soll den Wert `true` zurückgeben, wenn der Extremwert garantiert eingeschlossen wird, oder `false`, wenn der Extremwert nicht eingeschlossen ist. Gehen Sie hierbei davon aus, dass Ihr Polynom nur einen Extremwert besitzt.
3. Modifizieren Sie `bisection()` derart, dass Sie das Startintervall $[a, b]$ als Array übergeben können. Ihre Funktion sollte das ermittelte Intervall, welches die Nullstelle und damit den Extremwert einschließt, der aufrufenden Funktion zur Verfügung stellen.
4. Fügen Sie Ihrer Funktion einen weiteren Parameter hinzu, über den die aufrufende Funktion feststellen kann, ob es sich um ein Minimum oder Maximum handelt, beispielweise:
   - Maximum, wenn der Wert `-1` ist
   - Minimum, wenn der Wert `1` ist
   - keine Entscheidung, wenn der Wert `0` ist

Lösung zu Aufgabe 7.2

- Grundlegendes Vorgehen erklären
- muss nicht zwingend im Vorfeld implementiert und abgegeben sein
- Für 1. und 2. siehe Musterlösung im Programm
- bei 3. und 4. Verwendung von Pointern, um die Werte an die Funktion zu übergeben

Aufgabe 7.3 : Optionale Aufgabe

Diese Aufgabe ist nicht relevant für das Erlangen von Bonuspunkten und wird auch nicht im Rahmen der wöchentlichen Meetings abgefragt.

Weihnachtsmarkt

Alle Jahre wieder treibt es die Menschen zum Weihnachtsmarkt. Auch in diesem Jahr nutzen Sie wieder die Gelegenheit und schlängeln sich mit Ihren Kommilitonen zwischen gebrannten Mandeln, Bratäpfeln und mit Schokolade überzogene Bananen hindurch. Doch bei all den Verlockungen haben Sie immer das große Weihnachtsessen bei Ihren Liebsten im Kopf, für das Sie doch extra etwas weniger Naschereien im Vorfeld zu sich nehmen wollten. In diesem Jahr steht der Entschluss: Weihnachtsmarkt auf jeden Fall, aber dieses mal nur die weniger kalorienreichen Stände.

Da Sie ja ein Programmier-Fuchs sind, bereiten Sie sich auf den Weihnachtsmarktbesuch vor, indem Sie sich ein Programm schreiben, welches Ihnen den besten Weg – den Weg, mit den kalorienarmsten Ständen – berechnet.

Der Weihnachtsmarkt lässt sich in einem Raster aufteilen. Sie haben jeden Stand mit einer Gewichtung vermerkt, die abhängig von den kalorienreichen Weihnachtsleckerein ist. Alle Gewichte sind nicht-negativ.

Nehmen Sie an, dass Sie an jedem Stand, den Sie passieren, anhalten und einkaufen. Die so gesammelten Kalorien (Gewichte) summieren sich auf. Die Stände am Ein- und Ausgang werden immer besucht.

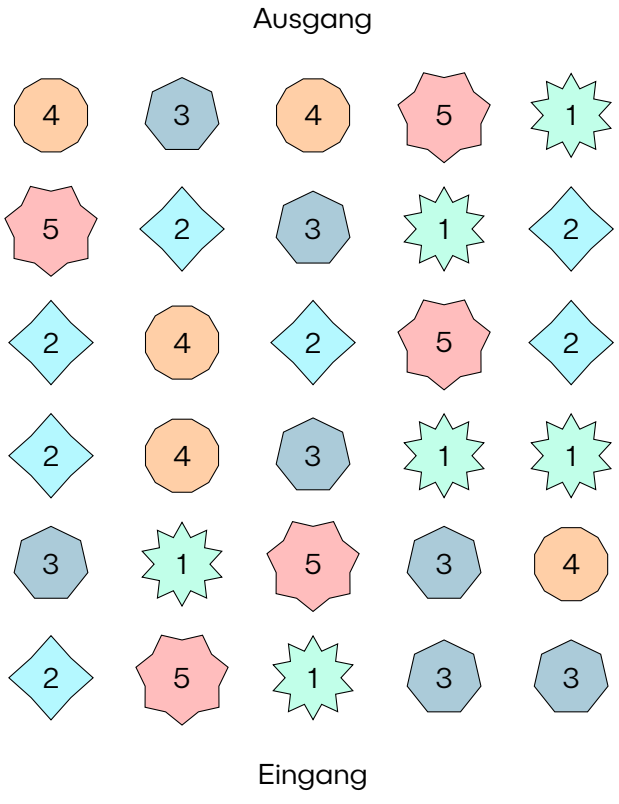


Abbildung 2: Lageplan Weihnachtsmarkt

Eingabe: Der Weihnachtsmarkt wird in einer einzelnen Datei wie folgt beschrieben. Die erste Zeile enthält die Anzahl der Zeilen ($0 < Y \leq 100$) und Spalten ($0 < X \leq 100$). Die zweite Zeile enthält die vier Werte y_e, x_e, y_a und x_a ($0 \leq x_e, x_a < X$, $0 \leq y_e, y_a < Y$). Diese bezeichnen den Eingang (y_e, x_e) und den Ausgang (y_a, x_a) des Marktes. Es folgen Y Zeilen mit jeweils X Einträgen, welche die Gewichte der einzelnen Stände darstellen. Alle Gewichte sind nicht nicht-negativ und kleiner als 100. Der erste Eintrag hat die Koordinate $(0, 0)$.

Beispiel:

Input (*case-0.in*)

```
6 5
0 2 5 2
4 3 4 5 1
5 2 3 1 2
2 4 2 5 2
2 4 3 1 1
3 1 5 3 4
2 5 1 3 3
```

Output (*case-0.out*)

```
18
```

Ausgabe: Eine einzelne Zahl, der kalorienärmste Wege (Summe der Gewichte) über den Weihnachtsmarkt.

Hinweise:

- Sie können Ihr Programm mit den gegebenen Eingabedateien *case-x.in* testen. Lassen Sie sich die Ausgabe im Terminal anzeigen.
- Testen Sie Ihr Programm mit unterschiedlichen Startpunkten. Lassen Sie sich als Ergebnis die Summe der Gewichte der abgelaufen Marktstände im Terminal ausgeben.
- Die Aufgabe lässt sich grundsätzlich mittels Rekursion lösen.
- Überlegen Sie, ob ein rekursiver Ansatz an dieser Stelle hilfreich bzw. effizient ist. Versuchen Sie Ihr Programm mit alternativen Algorithmen zu implementieren.