Standardabweichung mit C, Gnuplot, Zsh, Loops, logischen Verzweigungen und einem Input-File

Ergänzungsfach Informatik, zweites Semester, Thema Datenanalyse

Während des ersten Semesters sind Wahrheitswerte, logische Verzweigungen und die Schleife `for` bereits diskutiert und eingesetzt worden. Dies ist für die hier vorgestellten und war für die ihnen vorausgehenden Lektionen des zweiten Semesters zwar äusserst hilfreich, aber keine zwingende Notwendigkeit. Wie die folgende Grobplanung zeigt, können diese Themen bei Bedarf und unter Anpassung des Zeitplans auch erst im zweiten Semester eingeführt werden.

Vorwissen und Planung

Schuljahr und Altersstufe

Die Lernenden besuchen die 6. Klasse an einem Gymnasium. Entsprechend sind sie ca. 18 Jahre alt.

Vorwissen aus der Mathematik

Es ist für die Durchführung der hier vorgestellten Unterrichtssequenz keine Voraussetzung, dass die Lernenden die Grundlagen der Statistik kennen oder gar verstehen. Die Begriffe Mittelwert und Standardabweichung sowie die damit einhergehenden Konzepte werden in den hier vorgestellten Lektionen eingeführt. Siehe dazu auch der weiter unten folgende Abschnitt zur Grobplanung vom Semester.

Vorwissen aus der Informatik

Die Lernenden haben in diesem Kurs folgende Themen bearbeitet und diesbezügliche Fähigkeiten erlangt:

- Prozedurales Programmieren mit C (inklusive Pointer und Funktionen),
- Kompilation und Ausführung von C-Programmen auf der Kommandozeile sowie
- Arrays in C (inklusive Pointer und Strings).

Grobplanung

Die in diesem Dokument diskutierten Unterrichtseinheiten sind nachstehend aufgeführt unter dem Stichwort ``Thema 2: Standardabweichung mit C''. Einer Lektion entsprechen dabei 45 Minuten.

Thema 1: Mittelwert mit Python
(1.5 Lektionen) Mittelwert, Excel
(3.0 Lektionen) Python, Matplotlib, Loops, logische Verzweigungen, 1 Input-File
(3.0 Lektionen) Python, Matplotlib, Loops, logische Verzweigungen, mehrere Input-Files
(3.0 Lektionen) Dokumentation bzw. Präsentation mit LaTeX/TikZ

Thema 2: Standardabweichung mit C
(1.5 Lektionen) Standardabweichung, Excel
(3.0 Lektionen) C, Gnuplot, Zsh, Loops, logische Verzweigungen, 1 Input-File

Zielsetzung und Motivation

Die drei Lektionen zur Standardabweichung mit C, Gnuplot, Zsh, Loops, logischen Verzweigungen und einem Input-File sollen den Lernenden einen Umgang mit Daten beziehungsweise mit Werkzeugen aus der Informatik zur Bearbeitung von Daten vorstellen und näherbringen. Genauer gesagt soll ein Teil des Prozesses der Datengewinnung, des Datenmanagements, der Datenanalyse – auch im Kontext von Big Data – und der Präsentation der Resultate anhand eines Beispiels vorgestellt und umgesetzt werden. Der inhaltliche Rahmen dafür wird in den hier vorgestellten Lektionen durch die Standardabweichung aus der Statistik definiert. Die notwendigen Daten stammen aus der Finanzwelt.

Am Ende dieser Lektionen soll einerseits ein Teil eines Workflows, also eines Arbeitsflusses für eine Datenanalyse und eine Präsentation existieren; genauer gesagt wird das Folgende vor sich gehen:

1. Ein Zsh-Skript weist einem C-Programm ein Input-File zu.
2. Das C-Programm liest das ihm zugewiesene Input-File und berechnet für die gelesenen Werte einen Mittelwert sowie eine Standardabweichung. Dann füllt das C-Programm mithilfe dieser Werte ein Template eines Gnuplot-Skripts zur Erstellung eines Plots.
3. Zu guter Letzt sorgt das Zsh-Skript dafür, dass Gnuplot das Gnuplot-Skript ausführt und einen Plot der Daten samt Mittelwert und Standardabweichung erzeugt.

Die Kenntnis eines solchen Workflows kann für das Studium genauso wie für eine Arbeitsstelle von grossem Nutzen sein. Je früher die Schritte kennengelernt werden können, desto besser ist es!

Excel ist bei einem Input-File oder/und kleinen Datenmengen nützlich, aber bei vielen Input-Files oder/und grossen Datenmengen nicht zu gebrauchen. Deshalb sollen die hier vorgestellten Unterrichtseinheiten – genauso wie alle anderen in diesem einsemestrigen Kurs zur Datenanalyse – den Lernenden auch die Bearbeitung von Big Data näherbringen.

Genauer gesagt sollen die Lernenden in diesen Lektionen die ``klassisch'' berechneten Werte für den Mittelwert und die Standardabweichung noch für den Fall berechnen, in dem nicht alle Werte von Anfang an vorhanden sind, in dem die Datenmenge wie in einem Data-Warehouse-System[1] im Verlaufe der Zeit grösser wird.

Ein zusätzliches Ziel der Unterrichtseinheiten ist natürlich, dass das C-Programm zum Lesen, Berechnen und Ausgeben im Detail verstanden werden kann. Auch soll Vertrauen in Gnuplot als Werkzeug zum Erstellen von Plots geschaffen werden. Ein Vorteil von Gnuplot ist, dass es nicht nur Skripte ausführen, sondern auch interaktiv bedient werden kann. Das in früheren Lektionen verwendete Python-Modul Matplotlib soll zum Vergleich herangezogen werden.

[1] https://de.wikipedia.org/wiki/Data-Warehouse-System

Installation

Windows

Auf Windows wird Cygwin benötigt. Auf https://www.cygwin.com/install.html kann die Installationssoftware `setup-x86_64.exe` heruntergeladen werden. Anschliessend müssen die folgenden Kategorien und Pakete bei der Installation gewählt und installiert werden (jeweils die neueste Version):

- Kategorie
  - X11
- Pakete
  - gnuplot-x11
  - zsh
  - nano
  - vim
  - gcc-core
  - feh

Nun kann der Ordner `C:\cygwin64\bin` mit den Binaries von Cygwin zur Systemumgebungsvariable `PATH` hinzugefügt werden. Im File `C:\cygwin64\etc\nsswitch.conf` sollte dann die Zeile `# db_shell: /bin/bash` noch durch `# db_shell: /bin/zsh` ersetzt werden, sodass im Cygwin-Terminal wie auf MacOS die Zsh-Shell startet. Schliesslich muss dazu in `C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Cygwin` das Binary noch mit der rechten Maustaste angewählt werden. In den dortigen Einstellungen kann als Target `C:\cygwin64\bin\mintty.exe -i /Cygwin-Terminal.ico /bin/zsh –login` gesetzt werden. Schliesslich kann für X11-Sessions zum File `/home/<Username>/.bashrc` noch die Zeile `exec /bin/zsh` hinzugefügt werden. Dadurch startet ein Terminal auch in einer X11-Session (und nicht nur unter Cygwin) mit der Zsh-Shell.

MacOS

Unter MacOS gibt es bereits ein Unix-Terminal. Zsh ist von Grund auf die Shell auf MacOS und Nano sowie Vim sind ebenfalls von Grund auf verfügbar. Wir brauchen hier noch GCC von https://macappstore.org/gcc/ und Gnuplot von https://macappstore.org/gnuplot/. Die Installationsanleitungen werden auf den soeben zitierten Webseiten gezeigt.

Standardabweichung mit C

C, Gnuplot, Zsh, Loops, logische Verzweigungen und ein Input-File

Vorbereitung

Für den Unterricht sollten vorgängig gewisse Programme installiert und Einstellungen vorgenommen werden (siehe dazu das Dokument mit der Installationsanleitung). Die gesamte Arbeit kann dann auf Windows in einer X11-Session erledigt werden. Als Alternative dazu kann auf Windows der VirtualBox-Manager samt einer Virtual-Machine für ein Linux-Betriebssystem installiert werden. Für MacOS sind keine speziellen Vorbereitungen notwendig.

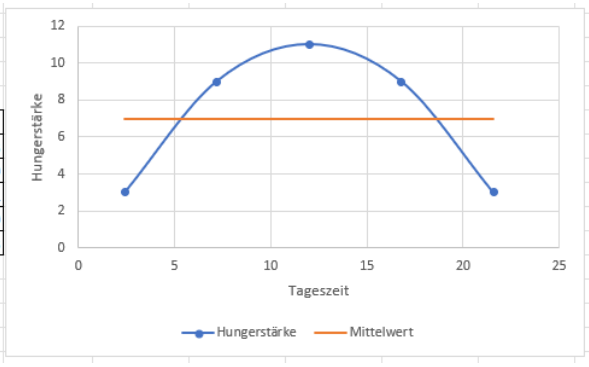
Datenquelle

Die Lernenden können von https://finance.yahoo.com/currencies eine zeitliche Folge von Werten eines Wechselkurses als CSV-File herunterladen. Mithilfe von zum Beispiel Excel kann anschliessend der ``Close''-Wert (das heisst der Schlusskurs) für einen Zeitraum von zum Beispiel einem Quartal extrahiert und in einer neuen Excel-Datei gespeichert werden.

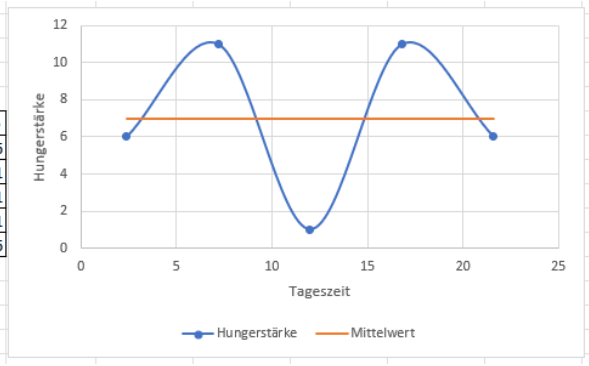
Standardabweichung

Der Mittelwert einer Verteilung von Daten kann zu ihrer Beschreibung verwendet werden, aber eine Verteilung von Daten wird dadurch nicht eindeutig festgelegt, das heisst, die in den folgenden beiden Abbildungen gezeigten Verteilungen der Hungerstärke von zwei Personen während eines Tages besitzen dieselbe mittlere Hungerstärke, und zwar 7, obwohl sie offensichtlich unterschiedlich sind. Bei der Interpretation dieses Wertes beziehungsweise beim Rückschliessen von diesem Wert auf die Verteilung der Daten gilt es also, sich dieser Tatsache, wann immer es möglich ist, bewusst zu sein.

Zeit (h)	Hungerstärke
2.4	3
7.2	9
12	11
16.8	9
21.6	3



Zeit (h)	Hungerstärke
2.4	6
7.2	11
12	1
16.8	11
21.6	6



Als weiteres Mass zur Beschreibung einer Verteilung von Daten kann die Standardabweichung verwendet werden. Sie beschreibt, wie gross die Streuung der Werte um den Mittelwert ist. Für normalverteilte Werte bedeutet die Standardabweichung sogar, dass sich 68 % von ihnen innerhalb einer, 95 % innerhalb von zwei und 99.7 % innerhalb von drei Standardabweichungen vom Mittelwert entfernt befinden.

Die Standardabweichung lässt sich für Werte x_k, $k \in \{1, 2, \dots, n\}$ durch

$$\sigma = \sqrt{\frac{1}{n}\sum_{k=1}^{n}(x_k - \mu)^2}$$

berechnen, wobei der Mittelwert der Verteilung durch

$$\mu = \frac{1}{n}\sum_{k=1}^{n} x_k$$

definiert wird.

Im Vergleich zur mittleren absoluten Abweichung

$$\sigma_{\text{Betrag}} = \frac{1}{n}\sum_{k=1}^{n}|x_k - \mu|$$

besitzt die Standardabweichung σ Vorteile. Betrachten wir dazu nochmals die Verteilungen der vorherigen beiden Abbildungen. Für die Verteilung in der oberen Abbildung ergeben sich die Werte $\sigma \approx 3.35$ und $\sigma_{\text{Betrag}} = 3.2$, wohingegen für die Verteilung in der unteren Abbildung $\sigma \approx 3.74$ und $\sigma_{\text{Betrag}} = 3.2$ gilt.[1] Die Standardabweichung σ ergibt für dieses Beispiel also nicht nur unterschiedliche Werte, sondern auch grössere im Vergleich zu den Werten von σ_{Betrag}. Dies hat damit zu tun, dass durch das Quadrieren grössere Abweichungen vom Mittelwert stärker gewichtet werden als bei der mittleren absoluten Abweichung σ_{Betrag}.

Eine nennenswerte Anwendung findet die Standardabweichung unter anderem in der Finanzwelt, der Industrie wie auch in der Meteorologie.

Big Data

Der Mittelwert und die Standardabweichung sollen auch für einen Stream von Daten wie im Kontext von Big Data berechnet werden.[2] Dabei sind nicht von Anfang an alle Werte vorhanden und sowohl der Mittelwert als auch die Standardabweichung müssen für jeden neuen Wert aktualisiert werden. Auf keinen Fall sollen sie in einer solchen Situation von Grund auf neu berechnet werden.

Die Berechnung des Mittelwertes ist dabei wie folgt umzusetzen:

1. Die Anzahl Werte n wird laufend aktualisiert. Für jeden neuen Wert wird n um 1 erhöht.
2. Die Summe $\sigma_n = \sum_{k=1}^{n} x_k$ wird zugleich laufend aktualisiert. Das heisst, sie wird nicht explizit, sondern über $\sigma_n = \sigma_{n-1} + x_n$ rekursiv berechnet. Das wiederum bedeutet, dass jeder neue Wert x_n zu σ_{n-1} addiert wird, um σ_n zu erhalten, wobei $\sigma_1 = x_1$ als Startwert festgelegt wird. Dadurch muss die Summe nicht für jeden neuen Wert komplett neu berechnet werden. Für Big Data ist dies von grösster Bedeutung.

[1] Die Lernenden sollen diese Werte ebenfalls berechnen, sodass sie kontrollieren können, ob sie die Formeln für den Mittelwert und die Standardabweichung richtig verstanden haben.
[2] https://en.wikipedia.org/wiki/Stream_(computing)

Die Berechnung der Standardabweichung ist etwas involvierter. Zunächst beobachten wir, dass

$$\sigma^2 = \frac{1}{n}\sum_{k=1}^{n}(x_k - \mu)^2 = \frac{1}{n}\sum_{k=1}^{n}(x_k^2 - 2x_k\mu + \mu^2) = \left(\frac{1}{n}\sum_{k=1}^{n}x_k^2\right) - \mu^2$$

gilt. Daraus folgt, dass wir zur Berechnung der Standardabweichung σ im Kontext von Big Data einzig die Summe $\sum_{k=1}^{n} x_k$ für den Mittelwert μ und die Summe $\sum_{k=1}^{n} x_k^2$ laufend aktualisieren müssen.

C

Der komplette Quellcode des C-Programms zur Berechnung des Mittelwertes und der Standardabweichung nach klassischer Statistik für die Werte in einem Datenfile befindet sich in den beigelegten Materialien. Auch der Quellcode des C-Programms für die Big Data-Berechnung liegt in fertiger Form vor. Die Idee wäre aber, dass die Lernenden dem Auftrag im hier mitgelieferten, entsprechenden Dokument folgen, um die gesamten Programme eigenständig zu schreiben.

Für Lernenden mit geringer Programmiererfahrung besteht die Möglichkeit, die Programme durch Ausfüllen von lückenhaften Programmtexten zu entwickeln. Dazu sollen sie zunächst die Logik der lückenhaften Programme zu verstehen und erst danach die fehlenden Zeilen Quellcode mithilfe des Internets zu vervollständigen versuchen. Da alle Lernenden Funktionen, Pointer, Arrays und vieles mehr der Programmiersprache C bereits kennen sollten, kann man davon ausgehen, dass sie ihre Fähigkeiten – mit Unterstützung durch die Lehrperson oder unter Zuhilfenahme des Internets – fortentwickeln werden.

Im Quellcode werden sowohl logische Verzweigungen als auch `for` Schleifen auftauchen. Diese sollten vorgängig oder/und in der abschliessenden Diskussion im Detail diskutiert, erklärt und/oder repetiert werden.

Die Quelldatei für den Teil zu Gnuplot kann anfangs ignoriert werden. Die Zeilen Code dazu können auskommentiert werden. Erst wenn die Pipeline ansonsten einmal steht, soll am Ende Gnuplot diskutiert und die relevanten Zeilen Quellcode erarbeitet werden. Dazu lohnt es sich, Gnuplot interaktiv mit dem Kommando `gnuplot` zu starten. Dort kann eine Grafik zuerst handgeschrieben erstellt werden. Das daraus generierte Wissen soll anschliessend in die Quelldatei für Gnuplot übertragen werden.

Für die Big Data-Berechnung soll durch ein C-Programm einzig ein Machbarkeitsnachweis geliefert werden. Das heisst, die klassische Berechnung soll in einem C-Programm mithilfe der Big Data-Berechnung reproduziert werden. Dazu soll angenommen werden, dass die Werte im Input-File wie in einem Data-Stream verfügbar werden. Insbesondere soll Gnuplot bei der Big Data-Berechnung keine Rolle spielen.

Zsh

Vom Terminal aus sollen die Quellcodes kompiliert werden. Dazu müssen alle Quelldateien verlinkt werden. Der dafür notwendige Befehl kann den Lernenden zur Verfügung gestellt werden. Alternativ können die Lernenden dazu aufgefordert werden, ihn selbst mithilfe des Internets zu finden. Auf jeden Fall lautet eine Version des Befehles für die klassische Berechnung

```
gcc *.c -Werror -lm -o main
```,

wobei die Option `-lm` für die C-Bibliothek `math.h` notwendig ist. Falls Gnuplot ausgeschlossen werden soll, kann `gnuplotp_schreiben.c` beim Kompilieren entsprechend ignoriert werden. Auf jeden Fall sollte

der obige Befehl ein ausführbares Programm `main` erzeugen. Dieses kann dann im Terminal wie folgt gestartet werden:

```
./main.exe eur_usd_q1_2022
```

Dabei wird der Euro-zu-USD-Wechselkurs für das erste Quartal im Jahre 2022 analysiert (falls das entsprechende CSV diese Daten tatsächlich enthält). Als Resultat davon sollte am Ende ein Gnuplot-File mit Endung `gp` erzeugt werden. Mithilfe von Gnuplot, sprich mithilfe des Befehls

`gnuplot eur_usd_q1_2022.gp`,

kann dafür dann eine Grafik (in diesem Fall eine PNG-Datei) generiert werden.

Bei der Big Data-Berechnung soll zunächst ein kleines Programm `konzept` geschrieben werden, an das über die Kommandozeile laufend neue Werte übergeben werden können. Dabei soll über ein Kommandozeilenargument vorweg die maximale Anzahl zu übergebender Werte festgelegt werden. Ein Aufruf für 10 Werte würde also wie folgt aussehen:

```
./konzept.exe 10
```

Weiterführende Aufgaben

Die Lernenden sind motiviert, alle Programme selbständig zu schreiben – bei Bedarf sollte die Lehrperson um Hilfe geben werden. Dazu gehört das Entwickeln und Erzeugen des Gnuplot-Skripts (siehe dazu das mitgelieferte Dokument zum Auftrag).

Für weiterführende Unterrichtseinheiten – insbesondere auch für schnelle Lernende – gilt, dass die folgenden Aufträge, Themen, Fragestellungen, Projekte, etc. zusätzlich noch bearbeitet werden können:

- Der Befehl zur Kompilation des C-Programms kann eigenständig entwickelt und gefunden werden.
- Das Zsh-Skript, welches den Befehl zur Ausführung des C-Programms enthält, kann selbständig geschrieben werden.
- Die Input-Daten können anstatt handschriftlich mit Excel auch mithilfe der GNU-Tools `cut`, `awk`, etc. oder des C-Programms selbst automatisch gefiltert und vorbereitet werden.
- In Vorbereitung auf weitere Lektionen können Daten für mehrere Zeitintervalle (zum Beispiel Quartale) vorbereitet und untersucht werden. Dabei kann ein laufender Mittelwert und eine entsprechende, laufende Standardabweichung berechnet und graphisch dargestellt werden. Die Darstellung kann für jedes Zeitintervall separat oder für alle Zeitintervalle zusammen erstellt werden.
- Die Daten der weiteren Zeitintervalle können über Big Data-Berechnungen in die Daten-Gesamtheit aufgenommen werden. Der laufende Mittelwert und die entsprechende laufende Standardabweichung resultieren somit aus Berechnungen über alle Werte und nicht nur über die Werte eines einzigen Zeitintervalls.

Auftrag

Datenvorbereitung

1. Die Werte eines Wechselkurses sollen von https://finance.yahoo.com/currencies in Form einer CSV-Datei heruntergeladen werden.
2. Mithilfe von Excel sollen die Werte des Wechselkurses auf ein Quartal beschränkt und in einer separaten CSV-Datei gespeichert werden.

Standardabweichung

1. Für ein paar Werte des obigen Wechselkurses soll der Mittelwert handschriftlich mithilfe der im Skript gegebenen Formel berechnet werden. Anschliessend soll der Mittelwert für das gesamte File mithilfe von Excel berechnet werden.
2. Für ein paar Werte des obigen Wechselkurses soll die Standardabweichung handschriftlich mithilfe der im Skript gegebenen Formel berechnet werden. Anschliessend soll die Standardabweichung für das gesamte File mithilfe von Excel berechnet werden.

C-Programm und Gnuplot

1. Es soll ein Plan notiert werden, demzufolge der Mittelwert und die Standardabweichung für die Daten mithilfe eines Programms bestimmt werden können. Das heisst, es soll notiert werden, in welcher Reihenfolge das Programm welche Schritte ausführen soll. Die vom Programm berechneten Werte sollen zur Kontrolle zunächst auf der Kommandozeile ausgegeben werden. Anschliessend sollen sie zum Schreiben des Gnuplot-Skripts verwendet werden.
2. Das Programm soll nun vollständig und eigenständig geschrieben werden. Es soll Schritt für Schritt mit der vorgeschlagenen Lösung in Übereinstimmung gebracht werden. Dabei soll wie folgt oder dem im vorherigen Schritt geschriebenen Plan entsprechend vorgegangen werden:
    a. Es soll ein Input-File gelesen und die gelesenen Zeilen zur Kontrolle ausgegeben werden. Im Internet soll nach der C-Bibliothek `stdlib.h` gesucht werden, um Hinweise auf die dafür relevanten Zeilen Quellcode zu erhalten.
    b. Die Daten vom Input-File sollen in ein Array geschrieben werden.
    c. Beim Füllen des Arrays soll die Anzahl Werte bestimmt werden, sodass die Länge des Arrays bekannt wird.
    d. Zur Kontrolle sollen alle Werte im Array ausgegeben und mit den Werten in der Excel-Datei verglichen werden.
    e. Nun soll die Summe aller Werte im Array bestimmt und das Resultat zur Kontrolle mit demjenigen aus der Excel-Berechnung verglichen werden.
    f. Die Summe soll nun zur Berechnung des Mittelwertes verwendet und das Resultat davon zur Kontrolle wiederum mit dem Excel-Resultat verglichen werden.
    g. Im nächsten Schritt soll die Standardabweichung berechnet und das Resultat mit Excel überprüft werden.
    h. Gnuplot soll nun interaktiv ausgeführt werden, um schrittweise den oder einen Graphen zu erzeugen. Auf der Gnuplot-Homepage http://www.gnuplot.info/ gibt es Tipps, wie Gnuplot verwendet werden kann.
    i. Nun soll vom Programm ein Gnuplot-Skript geschrieben und ausgegeben werden, wobei die berechneten Werte für den Mittelwert und die Standardabweichung

benutzt werden sollen. Im Internet soll wiederum nach der C-Bibliothek `stdlib.h` gesucht werden, um Hinweise auf die für das Schreiben einer Datei zu verwendenden Zeilen Quellcode zu erhalten.

j. Für die verschiedenen (zuvor aufgeführten) Schritte von oben sollen Funktionen geschrieben werden, sodass die Hauptfunktion schlanker wird. Die Resultate des auf diese Weise entstehenden neuen Programms sollen mit den Resultaten des alten Programms verglichen werden.

k. [*Optional*] Die Funktionsdeklarationen und -definitionen sollen in Header- und Source-Dateien verlagert werden, um die Lesbarkeit des Programms noch weiter zu verbessern. Der Output des neuen Programms soll wiederum mit demjenigen des alten Programms verglichen werden.

3. Für besonders schwache Lernende besteht die Möglichkeit, anstatt von Grund auf ein lauffähiges Programm zu schreiben, in den beigefügten Dateien schrittweise die unvollständigen Zeilen Quellcode zu vervollständigen. Dabei sollen die Lernenden mit dem Wesentlichen, also mit den wichtigsten Funktionen beginnen und darauf aufbauend schrittweise das finale Programm schreiben (incremental Programming).

Zsh-Skript

1. Die Zeile zur Ausführung des Programms soll separat im Terminal ausgeführt werden.
2. Die Zeile zum Generieren des Plots mithilfe von Gnuplot soll separat im Terminal ausgeführt werden.
3. Die obigen Befehle sollen in ein Shell-Skript verlegt werden. Das Skript kann mithilfe von `chmod +x skript.sh` ausführbar gemacht und getestet werden.
4. Das Skript soll so modifiziert werden, dass es ein Kommandozeilenargument für das Input-File versteht.

Big Data-Berechnung

Damit die Big Data-Berechnung eigenständig umgesetzt werden kann, sollen die relevanten Formeln vorgängig motiviert und weitmöglichst hergeleitet werden. Dies hängt auch vom Vorwissen im Fachbereich Mathematik ab.

Anschliessend soll möglichst wie zuvor vorgegangen werden. Dabei kann der Gnuplot-Teil vollständig ignoriert werden. Zunächst soll das Folgende erreicht werden:

1. [*Machbarkeitsnachweis*] In einem Programm soll gezeigt werden, dass der Mittelwert und die Standardabweichung von über die Kommandozeile eingegebenen Werten laufend mithilfe der Big Data-Methode berechnet werden können. Die Anzahl Werte soll so klein sein, dass handschriftlich gezeigt werden kann, dass die Resultate korrekt sind.
2. [*Finanzen*] Das Programm für die klassische Berechnung soll so angepasst werden, dass der Mittelwert und die Standardabweichung nicht nur nach der klassischen, sondern auch nach der Big Data-Methode berechnet werden. Am Ende sollen die Werte verglichen werden. Es soll sichergestellt werden, dass die Werte bis auf Rundungsfehler identisch sind. Der Gnuplot-Teil soll entfernt werden.

Schliesslich soll noch folgende Aufgabe bearbeitet werden:

Bei der Big Data-Methode können bei sehr langen Streams die Summen $\sum_{k=1}^{n} x_k$ und $\sum_{k=1}^{n} x_k^2$ so gross werden, dass ein sogenannter Overflow entsteht.[1] Zur Behebung dieses Problems für die Berechnung des Mittelwertes sollen die Lernenden die folgende Fragestellung bearbeiten:

1. Wie kann das Overflow-Problem durch Speichern von einzig der Anzahl Werte n sowie des Mittelwertes μ umgangen werden? Dabei soll beachtet werden, dass die Summe $\sigma_n = \sum_{k=1}^{n} x_k$ zwar berechnet werden kann, aber nicht gespeichert werden muss, wodurch das Overflow-Problem gelöst wird. Der folgende Tipp soll gegeben werden: Es gilt $\sigma_n = n\mu$, wovon die rechte Seite zur Berechnung des neuen Mittelwertes, welcher x_{n+1} in σ_{n+1} berücksichtigt, derart verwendet werden kann, dass das Overflow-Problem umgangen wird.
2. Der resultierende Ansatz für das bzw. die Lösung zum vorherigen Problem soll als lauffähiges Programm umgesetzt werden. Dazu soll das bereits geschriebene Programm für den Machbarkeitsnachweis umgeschrieben werden.

Zusatzaufgaben

Siehe dazu das Dokument `04_Skript.pdf`.

[1] https://en.wikipedia.org/wiki/Floating-point_arithmetic#Dealing_with_exceptional_cases

Lösungen

Die Lösung zur Big-Data-Aufgabe für einen langen Stream lautet wie folgt:

1. Es sei $\mu_n = \frac{1}{n}\sigma_n$ für $\sigma_n = \sum_{k=1}^{n} x_k$ der ``alte'' Mittelwert. Somit gilt für den ``neuen'' Mittelwert $\mu_{n+1} = \frac{1}{n+1}\sigma_{n+1} = \frac{1}{n+1}(\sigma_n + x_{n+1}) = \frac{1}{n+1}(n\mu_n + x_{n+1})$. Daraus wird ersichtlich, dass der ``neue'' Mittelwert $\mu_{n+1}$ aus dem ``alten'' Mittelwert μ_n berechnet werden kann.
2. Damit kein Overflow entsteht, sollte
$$\mu_{n+1} = \frac{n}{n+1}\mu_n + \frac{x_{n+1}}{n+1}$$
anstatt
$$\mu_{n+1} = \frac{1}{n+1}(n\mu_n + x_{n+1})$$
berechnet werden. Dadurch werden grosse Zahlen gänzlich vermieden (der Term $n\mu_n = \sigma_n$ muss nicht mehr berechnet werden). Im Programm wird dazu

```
mittelwert = ((float) anzahl_werte)/(++anzahl_werte)*mittelwert+x/anzahl_werte
```

berechnet, wobei `x` dem neuen Wert entspricht.

Die Lösungen zu allen anderen Aufgaben befinden sich in den beigelegten Dateien.

Fazit

Diese Unterrichtssequenz vermittelt den Lernenden einen Eindruck, wie mit Big Data im Studium und der Arbeitswelt gearbeitet wird, der für sie von äusserster Wichtigkeit ist, da Daten unsere Gegenwart und Zukunft bestimmen.

Die verwendeten Tools sind aus den folgenden Gründen relevant:

- Die Programmiersprache C wird in vielen Arbeitsbereichen noch immer verwendet. Sie ist unter anderem auch die Grundlage der Programmiersprachen C++, Java und Python.
- Gnuplot ist zwar scheinbar etwas aus der Mode gekommen, relevant bleibt das Tool aber trotzdem. Insbesondere liegt dies daran, dass es einerseits interaktiv verwendet werden kann. Andererseits kann es aber auch über Skripts verwendet werden, was gerade im Zusammenhang mit Big Data von grosser Bedeutung ist.
- Die Zsh-Shell hat die Shell Bash ersetzt oder ist dabei, die Shell Bash zu ersetzen. Zum Beispiel ist sie auf MacOS bereits die von Grund auf gegebene Shell. Auch die Linux Distribution Kali benutzt Zsh von Grund auf.
- Die Arbeit auf der Kommandozeile ist immer von Bedeutung, wenn IT eine Rolle spielt. So früh wie möglich damit vertraut zu werden, ist von äusserster Relevanz!