Technische Universität Dortmund
Fakultät Statistik
Lehrstuhl Computergestützte Statistik

Dr. Uwe Ligges
M.Sc. Marieke Stolte
Luca Sauer
Rudi Zulauf

Allgemeine Information zur Vorlesung
Computergestützte Statistik
Wintersemester 2022/2023

In der Übung zur Computergestützten Statistik werden Sie viel Programmieren müssen. Um uns und auch Ihnen selbst die Arbeit zu erleichtern, möchten wir Sie bitten diese Hinweise zu beachten.

Verpflichtende Hinweise:

Wir bitten Sie sich auf jeden Fall an diese Punkte zu halten. Wir behalten es uns vor wiederholte Verstöße mit Punktabzug zu bestrafen.

Die Datei

- Quellcode muss im dafür vorgesehenen Dateiformat, also als .R Datei, abgegeben werden. Quellcode, der nicht im entsprechenden Format, also z.B. stattdessen als Word oder OpenOffice Dokument eingesendet wird, kann nicht korrigiert werden. Weitere Ergebnisse bündeln Sie bitte in einer einzelnen PDF-Datei.
- Um die Korrektur zu vereinfachen, benutzen Sie bitte folgendes Schema für Ihre Dateinamen: <nachname>-<uebungsblattnr>. Zum Beispiel: ligges-01.R oder ligges-01.pdf.
- Beachten Sie dabei bitte, dass der Gebrauch von Umlauten in Datei-, Variablen- und Funktionsnamen häufig zu Problemen führt. Vermeiden Sie diese also!
- Quellcode muss eigenständig lauffähig sein. Testen Sie also Ihre Dateien vor Abgabe. Starten Sie dazu eine neue R-Session und lesen Sie die gesamte R-Datei ein. Dabei darf kein Fehler auftreten.
- Das Ausführen Ihrer Abgabe sollte weniger als 1 Minute dauern. Sie werden im Laufe des Semesters jedoch auch Programme schreiben, die eine längere Laufzeit haben. In diesem Fall bitten wir Sie, die entsprechenden Aufrufe in Ihrem Programmcode auszukommentieren und die Ergebnisse (Graphiken, Tabellen, ...) gesondert (als PDF oder als Kommentar in der R-Datei) abzugeben.
- Das Ausführen Ihrer Datei sollte keine neuen Dateien etc. erstellen. Sollten Sie zum Beispiel Ihre Graphiken durch den Befehl pdf erstellen, kommentieren Sie diesen in Ihrer Abgabe bitte aus.
- Jede Zeile in Ihrer R-Datei darf maximal 80 Zeichen beinhalten, da ansonsten beim Ausdrucken ein unschöner Zeilenumbruch erstellt. Auch lässt die Lesbarkeit bei Zeilen mit mehreren 100 Zeichen irgendwann nach.

Handschriftliche Abgaben

- Bitte geben Sie jede Aufgabe auf eigenen Zettel ab. Heften Sie bitte alle Zettel, die zu einer Aufgabe gehören, zusammen. Zettel, die zu unterschiedliche Aufgaben gehören, bitte nicht zusammenheften.
- Auf jeder Heftung von Zetteln muss Ihr Name stehen.
- Schreiben Sie leserlich, unlesbare Passagen werden im Zweifel zu Ihrem Nachteil ausgelegt.
- Sie dürfen Ihre handschriftlichen Abgaben gerne einscannen und als PDF einsenden, abfotografieren ist jedoch nicht gestattet.

Programmierstil Achten Sie auf einen einheitlichen Programmierstil und behalten Sie diesen konsequent bei. Viele freiwillige Hinweise finden Sie weiter unten in diesem Dokument. Verpflichtend ist:

- Programmcode muss ordentlich eingerückt werden.
- Variablen müssen sinnvoll benannt werden. a und b sind meistens keine sinnvollen Variablennamen. Halten Sie sich bei Ihren Variablennamen an die Notation der Vorlesung und die Vorgaben des Übungszettels.
- Die Aufrufe source, load und library gehören an den Anfang Ihrer Datei, damit wir wissen welche Pakete und übrigen Dateien wir zum Ausführen benötigen.
- Verwenden Sie relative Pfade zum Einlesen von Dateien. Setzen Sie dazu mit setwd das working directory Ihrer R-Session auf den Quellpfad Ihrer Datei und lesen Sie Dateien mit source(name.R) bzw. load(name.RData) und ähnlichem ein.
- Bündeln Sie Ihrem Programmcode möglichst immer in Funktionen und rufen Sie anschließend diese Funktionen auf. Falls Sie zum Beispiel eine Graphik erstellen müssen, schreiben Sie eine Funktion plotIt zum Erstellen der Graphik und rufen Sie anschließend plotIt() auf.

Dokumentation

- Zu jeder Funktion, die Sie schreiben, gehört auch Dokumentation. Diese sollte als Kommentarblock vor der eigentlichen Funktionsdefinition im Quellcode stehen.
- Dokumentieren Sie die Eingabe jeder Funktion. Dabei muss für jeden Eingabeparameter klar ersichtlich sein 1) sein Datei-Typ ist und 2) seine Bedeutung
- Gleiches gilt für die Ausgabe Ihrer Funktion.
- Eine Kurzbeschreibung (1 Satz) *was tut die Funktion* gehört ebenfalls vor den Funktionskopf
- Übertreiben Sie es nicht mit der Dokumentation innerhalb der Funktionen. Nicht jede Zeile muss mit einem Kommentar versehen werden. Guter Code ist auch zu einem Teil „selbsterklärend“. Die Kunst liegt darin, die richtige Balance zu finden - wie bei fast allem im Leben...

Weitere Hinweise:

Jetzt kommen noch einige weitere Hinweise. Wir werde die Missachtung dieser Punkte nicht bestrafen, würden Ihnen dennoch empfehlen sich an zumindest einige der Punkte zu halten. Diese tragen zur besseren Lesbarkeit Ihres Codes bei und führen dazu, dass sowohl wir als auch Sie Fehler schneller findne.

- Verschiedene Beispiele für einen einheitlichen Programmierstil finden Sie darüber hinaus auf den folgenden Internetseiten:

  `https://github.com/tudo-r/PackagesInfo/wiki/R-Style-Guide`

  `https://google-styleguide.googlecode.com/svn/trunk/Rguide.xml`

  `http://r-pkgs.had.co.nz/style.html`

  Wir würden Ihnen empfehlen sich einen dieser Programmierstile anzueignen und diesen konsequent zu verwenden. Falls Sie dies nicht wünschen, würden wir Ihnen folgende Punkte nahe legen:

- Leerzeichen: Vor und hinter (fast) jeden binären Operator gehört ein Leerzeichen. Es heißt z.B. `1 + 2` oder `2 * 3` und nicht `1+2` oder `2*3`. Vor allem bei langen, arithmetischen Ausdrücken wird Programmcode ohne Leerzeichen schnell unübersichtlich. Wenige Ausnahmen sind z.B. `1:2` und `2^3`. Hinter jedes (!) Komma gehört ein Leerzeichen. Leerzeichen vor und hinter runden und eckigen Klammern sind nicht erwünscht. D.h. `sin(2)` ist gut, `sin ( 2 )` hingegen nicht.

- Leerzeilen: Oft hilft es der Struktur von Programmcode, nach einigen sinngemäß zusammengehörigen Zeilen eine Leerzeile einzufügen. Häufig ist dies auch die richtige Stelle, um einen kurzen Kommentar in den Code einzufügen.

- Variablennamen: Variablen beginnen stets mit einem kleinen Buchstaben. Funktionen werden im Camel-Case Stil geschrieben, d.h. jedes neue Wort innerhalb des Variablennamens beginnt mit einem großen Buchstaben. Beispiel: doIt, plotIt, getMinimum, ..., bei allen weiteren R-Objekten erfolgt die Trennung durch einen Punkt. Beispiel: x.min, y.dach, data.transponsed. Ausnahmen können zum Beispiel Matrizen sein, deren Name üblicherweise ein einzelner, große Buchstabe ist (konsistent zur Notation der Vorlesung). Variablennamen sollten nicht länger als 10-15 Zeichen sein, ansonsten leidet die Lesbarkeit.

- Schreiben Sie TRUE und FALSE immer aus. T und F werden auch gerne mal als Variablennamen verwendet. Das führt zu Fehlern, die fast unmöglich zu finden sind. Vermeiden Sie es daher am besten auch, T und F als Variablennamen zu verwenden. Gleiches gilt auch für z.B. c, da c() eine viel benutzt Funktion ist.

- Viele kleine Angewohnheiten im Code sind überflüssig. Z.B. Den Vergleich if (ok == TRUE) müssen Sie nie machen, if (ok) hat die gleiche Bedeutung, oder aber auch bei c(1) oder c(1:10) ist der Aufruf von c() überflüssig. Versuchen Sie diese zu vermeiden, wir meckern solche Stellen im Code meistens an.

- Vermeiden Sie Copy-Paste in Ihren Abgaben. Schreiben Sie stattdessen Funktionen und rufen Sie diese mehrfach auf. Copy-Paste ist eine schlimme Fehlerquellen und erzeugt sehr unübersichtlichen Programmcode.

- Achten Sie auf Effizienz, übertreiben Sie es aber nicht. Ein Ziel dieser Veranstaltung ist es, Sie dafür zu sensibilisieren, wie „teuer“ manche Rechenoperationen sind. Versuchen Sie daher stets in Ihrer Implementierung möglichst effizient mit den Rechnerressourcen umzugehen. Andererseits sollte Ihre Implementierung nicht zugunsten der Lesbarkeit optimiert werden. Auf diesen Aspekt wird in der Übung immer wieder eingegangen.

- Und der vielleicht wichtigste Punkt: Lesen Sie unsere Korrekturen und versuchen Sie unsere Ratschläge zu befolgen. Nehmen Sie es uns nicht böse, wenn wir am Anfang viel meckern. Es ist noch kein Meister vom Himmel gefallen und Sie sind hier um das alles zu lernen. Vieles mag für Sie bereits selbstverständlich sein, der Rest ist es hoffentlich nach diesem Semester.