# Übungen zur Vorlesung Algorithmische Bioinformatik

Freie Universität Berlin, WS 2016/17
Martin Vingron · Annalisa Marsico · Alena van Bömmel · Edgar Steiger · Thimo Wellner

**Blatt 12 · Ausgabe am 16.1.2017**
**Abgabe am 23.1.2017 vor Beginn der Vorlesung**

Name: Matrikelnummer: Übungsgruppe:

**Aufgabe 1** (*40 Punkte Praxis*)**.** Wir möchten die Expressionsdaten mehrerer Samples zweier Leukämieklassen (Golub et al., 1999[1]) analysieren. Berechnen Sie dafür die Hauptkomponenten der Leukämiedaten in R.

1. Es gibt zwei Klassen von Leukämiesamples in diesem Datensatz: AML und ALL. Beschreiben Sie kurz worum es sich bei diesen Leukämieklassen handelt.

2. Lesen Sie die Daten in R ein (z.B. mit *read.table*). Wie viele Gene und Samples gibt es? Wieviele Samples sind in der jeweiligen Leukämieklasse?

3. Log-transformieren Sie die Expressionsdaten und zentrieren Sie diese dann indem Sie die Mittelwerte jeder Zeile subtrahieren. Erstellen Sie einen Boxplot der Daten. Sind die Daten schon normalisiert? Begründen Sie Ihre Antwort.

4. Berechnen Sie die Hauptkomponenten der transponierten Expressionsmatrix. Verwenden Sie dafür die Funktionen `cov, t, eigen, %*%, svd`. Die Verwendung der Funktionen `prcomp`, `princomp` oder ähnlicher Funktionen ist nicht zulässig.

5. Geben Sie die 5 größten Eigenwerte an. Wieviel Varianz erklären die 1. Komponente alleine und die ersten 5 Komponenten zusammen?

6. Erstellen Sie einen Plot der ersten 2 Komponenten. Verwenden Sie die Farbkodierung für die AML Samples `col='red'` und für die ALL Samples `col= 'blue'`. Können Sie eine gute Trennung zwischen den beiden Gruppen erkennen?

**Aufgabe 2** (*30 Punkte Praxis*)**.** In der Vorlesung wurde *DESeq* zum Finden von differentiell exprimierten Genen vorgestellt. Die Methode wurde entwickelt um Count-Daten, wie sie z.B. bei RNA-Seq Experimenten entstehen, zu analysieren. Sie sollen nun DESeq auf einen Testdatensatz von Brooks et al. (2011)[2] anwenden. Es handelt sich dabei um Expressionsdaten aus *Drosophila melanogaster* mit zwei Klassen (treated vs. untreated). Die Gruppe *treated* wurde dabei mit einer RNAi gegen den Splicingfaktor *pasilla* behandelt. Installieren und laden Sie dazu DESeq in R/Bioconductor wie folgt:

```
source("http://bioconductor.org/biocLite.R")
biocLite("DESeq")
library(DESeq)
```

Hilfe zum DESeq-Package finden Sie z.B. auf `http://bioconductor.org/packages/release/bioc/html/DESeq.html`.

[1] Material1: `https://www.molgen.mpg.de/3726808/golub_train.txt`
[2] Material2:`https://www.molgen.mpg.de/3726799/deseq_counttable.txt`

1. Laden Sie die Tabelle der Counts in R. Um mit DESeq arbeiten zu können, müssen Sie Ihre Tabelle in eine *CountDataSet*-Datenstruktur konvertieren. Nutzen Sie dazu die Funktion *newCountDataSet*. Wie viele Samples gibt es? Wie viele Gene?

2. Wahrscheinlich wurden die verschiedenen Samples nicht mit der gleichen Sequenziertiefe sequenziert. Daher müssen wir die Samples vor der Analyse normalisieren, um aussagekräftige Ergebnisse zu erhalten. Schätzen Sie dafür die Size Factors mit *estimateSizeFactors* und inspizieren Sie diese mit der Funktion *sizeFactors*. Gibt es große Unterschiede zwischen den Samples?

3. Wir müssen nun die Varianz jedes Gens abschätzen. Das DESeq-Package schätzt dafür den Dispersionsparameter mit der Funktion *estimateDispersions*.

4. Nun können wir die differentiell exprimierten Gene mit der Funktion *nbinomTest* bestimmen. Extrahieren Sie alle differentiell exprimierten Gene mit einem korrigierten P-Wert kleiner als 0.01. Wie viele Gene erhalten Sie? Wie viele davon sind hoch- bzw. herunter-reguliert?

5. Extrahieren Sie nun die Gen-IDs und führen Sie eine GO-Term-Überrepräsentationsanalyse aus. Nutzen Sie dazu das DAVID-Tool. Beachten Sie, dass es sich bei Ihren Gen-IDs um Flybase-Gene-IDs handelt. Setzen Sie außerdem einen geeigneten Hintergrund für die GO-Analyse. Laden Sie dazu die Liste aller analysierten Gene auf den DAVID-Server. Welche Funktionen sind durch den Knock-down von *pasilla* beeinträchtigt?

**Aufgabe 3** (*30 Punkte; Praxis*)**.** In der Vorlesung haben Sie den $\mathcal{X}^2$-Test als Test auf Unabhängigkeit zweier Zufallsvariablen kennengelernt. Im Folgenden werden unterschiedliche Szenarien für zwei binäre Variablen simuliert und auf Unabhängigkeit getestet.

1. Simulieren Sie in R zwei binäre Zufallsvariablen $V1$ und $V2$ mit $N = 1000$ Beobachtungen und mit folgenden Erfolgswahrscheinlichkeiten: $p_1 = 0.25$; $p_2 = 0.5$. Erstellen Sie dann eine Kontingenztabelle wie hier dargestellt (RS steht fr Randsumme):

| | | V1 | | |
|---|---|---|---|---|
| | | 1 | 0 | RS |
| | 1 | a | b | |
| V2 | 0 | c | d | |
| | RS | | | |

   Testen Sie diese Kontingenztabelle mit dem $\mathcal{X}^2$-Test in R (*chisq.test(tab)*) auf Unabhängigkeit. Würden Sie mit den gegebenen Beobachtungen und dem berechneten $p$-Wert sagen, dass die Variablen unabhängig voneinander sind?

2. Verteilen Sie jetzt die 1000 Beobachtungen auf $a$, $b$, $c$ und $d$ so, dass die Einträge dem Produkt der Randsummen (RS) normiert durch $n$ entsprechen. Erklären Sie kurz, was Sie dadurch modellieren. Testen Sie diese Kontingenztabelle wieder mit dem $\mathcal{X}^2$-Test auf Unabhängigkeit. Würden Sie mit den gegebenen Beobachtungen und dem berechneten $p$-Wert sagen, dass die Variablen unabhängig voneinander sind?

3. Verschieben Sie nun Schritt für Schritt das Gewicht der Beobachtungen auf die Hauptdiagonale der Kontingenztabelle. Verschieben Sie dazu in 1er-Schritten Beobachtungen von der Zelle $c$ auf Zelle $a$ und von Zelle $b$ auf Zelle $d$. Was modellieren Sie dadurch?

4. Berechnen Sie für diese Sequenz die $p$-Werte mit Hilfe des $\mathcal{X}^2$-Tests. Was beobachten Sie? Geben Sie eine Begründung zu Ihrer Beobachtung.