

# Anleitung für das beiliegende Dienstplanprogramm

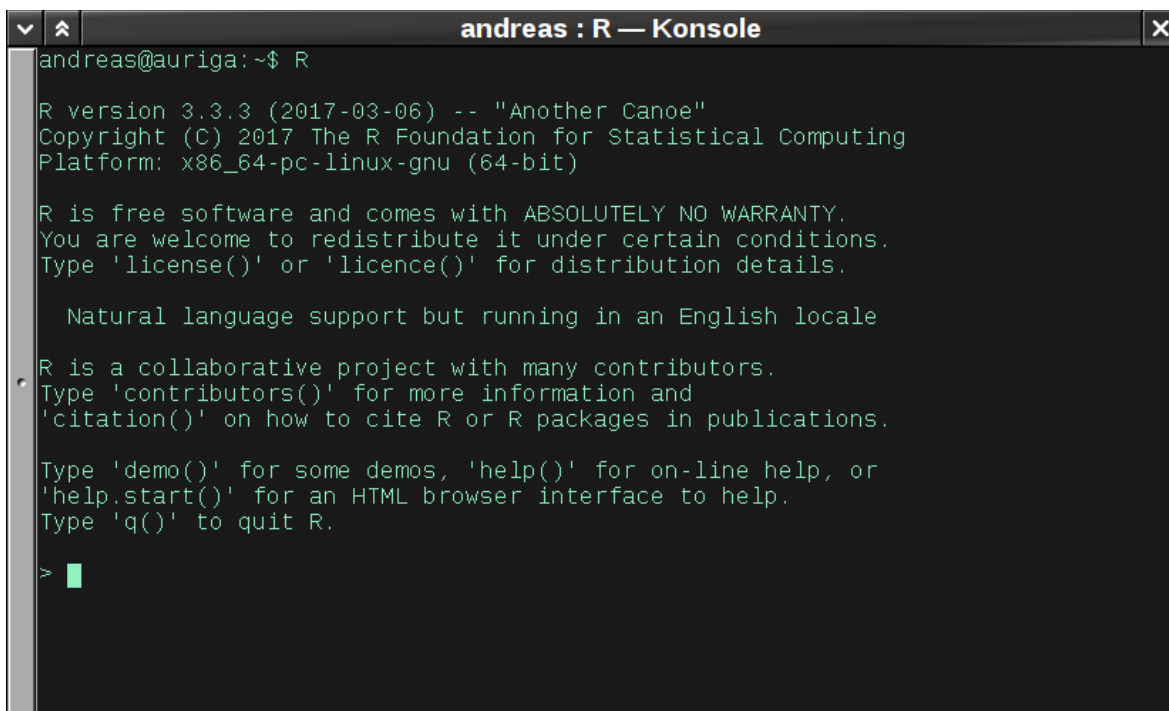
## Vorbereitung

### 1 R installieren

Das Dienstplanprogramm ist in R geschrieben, eine Mathematik/Statistik-Programmiersprache, die muss vorher installiert werden, geht auf allen gängigen (Linux, Windows, OSX) Betriebssystemen. Gehe zu <https://www.r-project.org/> und folge den Anleitungen. (Unter Debian-basierten Systemen einfach mit `apt install r-base` installieren).

### 2 Notwendige Pakete in R installieren

Es gibt zahlreiche Pakete mit Zusatzfunktionen in R, das Dienstplanprogramm benutzt 2 davon, die müssen auch noch installiert werden. Dazu starten wir R. In Linux und OSX einfach einen Terminal öffnen und 'R' eingeben, in Windows bringt R einen eigenen Terminal mit, unter Startmenü/Programme/R/etc. Das sieht dann in etwa so aus:



```
andreas@auriga:~$ R

R version 3.3.3 (2017-03-06) -- "Another Canoe"
Copyright (C) 2017 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> █
```

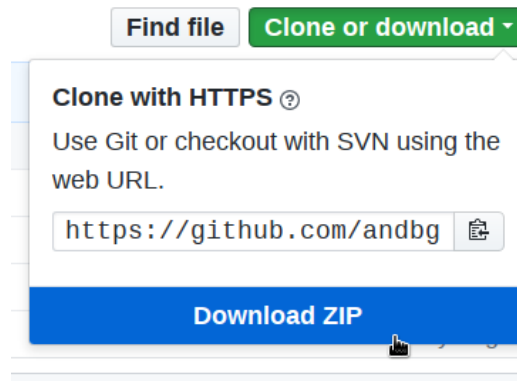
Um die notwendigen Pakete zu installieren, führe aus (d.h. die folgenden Zeilen eingeben und Enter drücken):

```
install.packages("Hmisc")
install.packages("xlsx")
```

Die Pakete sollten jetzt in R installiert sein.

### 3 Dienstplanprogramm herunterladen und entpacken

Gehe zu [https://github.com/andbgr/doctor\\_scheduling](https://github.com/andbgr/doctor_scheduling) und lade das Dienstplanprogramm herunter:



Danach den ZIP Ordner irgendwo entpacken.

## Dienstplanprogramm verwenden

### 1 Neuer Ordner für neuen Monat

Geh in den Ordner `doctor_scheduling`. Es empfiehlt sich, für jeden Monat einen neuen Unterordner anzulegen. Der Ordner `example` dient als Vorlage, den kopieren wir, der neue Ordner heisst z.B. `2019-03`, weil wir März machen (und wenn später die Ordner mehr werden ist das besser zu sortieren als “März”).

### 2 Liste der Ärzte und allgemeine Präferenzen: `doctors.xlsx`

Öffne die Datei `doctors.xlsx` und gib die Ärzte mit ihren allgemeinen Präferenzen ein:

<code>ward</code>	Station
<code>weekhours</code>	Wochenstunden
<code>weekhours_max</code>	Wochenstunden maximal im Plan
<code>split_shifts</code>	‘yes’ oder ‘no’, macht der Arzt geteilte (12.5h) Dienste?
<code>friday_sunday</code>	‘yes’ oder ‘no’, macht der Arzt Freitag-Sonntag-Kombinationen
<code>number_of_shifts_factor</code>	Nur falls ein Arzt mehr oder weniger Dienste als die Anderen macht. Z.B. neuer Kollege, hat die ersten 2 Wochen Schonfrist, im input (siehe später) steht in den ersten 2 Wochen

	‘!N’, dann könnte man hier 0.5 hinschreiben, damit er nicht 4 Dienste am Ende des Monats hat.
long_day_hours	6,7, oder 8: Wenn ein Arzt mit Nachtdiensten + 5h-Tagen noch nicht 40 Wochenstunden erreicht hat, welche längere Dienstform ist bevorzugt? (mehrere 6h-Tage oder weniger 7h-Tage, etc)
fill_all_days	Wenn die Sollpräsenz der Station UND die 40 Wochenstunden des Arztes erreicht sind, sollen dann trotzdem noch mehr Tagdienste vergeben werden? (Wenn nicht: Stricherltage)
shifts_carryover	Wenn im letzten Monat ‘zu viele’ oder ‘zu wenige’ Dienste (gemessen an einem rechnerischen Soll, mit Kommastellen) gemacht wurden, kann man die aus der Statistik des Vormonats hierher kopieren, dann wird das berücksichtigt.
weekends_carryover	Wie oben, nur für Wochenenden

### 3 Dienstplanprogramm ausführen

In unserem Ordner liegt ein Helferskript, `example.R`, von dem aus die Funktionen aus dem eigentlichen Dienstplanprogramm aufgerufen werden.

1. Öffne einen Terminal. Geh in den neuen Ordner, den wir vorhin angelegt haben. In diesem Beispiel ist der Ordner in meinem home Verzeichnis `/home/andreas`, dann wäre der Befehl  
`cd /home/andreas/doctor_scheduling/2019-03/`  
Dann starte wieder R mit  
R  
in Windows kann man im R-Fenster auch das Menü verwenden, mit Datei > Verzeichnis wechseln
2. Öffne `example.R` im Texteditor deiner Wahl (in Windows bringt R auch einen Texteditor mit, einfach im R Hauptfenster: Datei > Skript öffnen)
3. Editiere das Skript: passe `start_date` und `end_date` an
4. lösche die Datei `input.xlsx`, wir schreiben jetzt eine neue, für den neuen Monat
5. Führe Alles aus `example.R` bis inklusive `write.template(...)` aus. Das heisst, kopiere die Zeilen ins Terminalfenster, oder (im Windows-R) markiere die Zeilen und Rechtsklick > Auswahl ausführen. Die Funktion `write.template(...)` schreibt eine neue `input.xlsx` Datei, mit den vorher eingegebenen Ärzte und dem vorher angegebenen Zeitraum. Info: Wenn man die Funktion erneut aufruft, weigert sie sich, die Datei zu überschreiben, weil möglicherweise schon Wertvolles drinsteht.
6. **Befülle `input.xlsx` mit den Einschränkungen/Wünschen der Ärzte und (im zweiten Tabellenblatt) den Sollpräsenzen der Stationen. Anleitung zu validen Eingaben sind in der**

## Datei.

- Führe den Rest des Skripts aus. (Oder das ganze Skript erneut – bestehender Input wird nicht überschrieben). Dies ruft schliesslich die Funktion `optimal.schedule()` auf, die den Dienstplan erstellt. Beim Aufruf der Funktion kann man einige Parameter einstellen, die Anzahl der Iterationen, und die Gewichtung der Qualitätsparameter. Probiere verschiedene aus. Auf meinem Computer gehen sich etwa 14000 Iterationen in 1h aus. Während das Programm läuft, sieht das in etwa so aus:

```
Optimizing - trying 7000 variations...
n.unres n.rq_den n.srq_granted r.shifts r.weekends r.nights n.split day_presence
0 0 2/7 1.306 0.8444 3 9/16 24.4 (28.9)
0 0 2/7 1.806 0.6556 1 8/16 24.6 (26.4)
0 0 5/7 1.667 0.8222 1 6/16 22.6 (22.3)
0 0 6/7 1.194 0.7222 1 6/16 23.6 (25.0)
0 0 6/7 1.028 0.8222 1 7/16 24.8 (26.1)
0 0 4/7 1 0.8222 1 8/16 21.8 (21.0)
0 0 6/7 1.25 0.4556 1 8/16 23.4 (24.8)
0 0 6/7 1.028 0.4556 1 8/16 25.8 (29.1)
0 0 5/7 0.7778 0.4778 1 9/16 23.0 (24.4)
```

Dabei repräsentiert jede Zeile 1 neuen Dienstplan mit seinen Qualitätsparametern. Es wird aber nur eine neue Zeile ausgegeben, falls der neue Dienstplan besser als der vorige ist. Somit geht es am Anfang schnell, dann kommen immer seltener neue Zeilen hinzu. Am Ende wird der ‘beste’ Dienstplan dann in die Datei `schedule.xlsx` geschrieben. Die Differenz zur Sollpräsenz je Station steht unter dem Dienstplan, ausführliche Statistik im Tabellenblatt ‘doctors.stats’. Bitte auch im Tabellenblatt ‘warnings’ nachsehen, ob es Probleme gibt (z.B. ein FT konnte nicht vergeben werden o.Ä.). Hier noch eine Erklärung der Qualitätsparameter:

n.unres	Anzahl der ungelösten Tage (kein Nachtdienst) – sollte 0 sein (ausser es geht sich nicht aus)
n.rq_den	Anzahl der Verstösse gegen Einschränkungen – sollte 0 sein (ausser bei unauflösbaren Eingaben, z.B. fixe Eingabe von N bei 2 Ärzten am gleichen Tag – da wird dann einer rausgeworfen)
n.srq_granted	Anzahl der “soft requests” die erfüllt wurden. Das sind alle Eingaben mit einem ‘?’
r.shifts	Range, also Spanne der Anzahl der Nachtdienste. Ein Beispiel: Das rechnerische Soll an Nachtdiensten in einem Monat ist 3.6. Jetzt haben die Ärzte zwischen 3 und 4 Diensten, dann wäre r.shifts 1, weil maximale Abweichung nach unten (0.6) plus maximale Abweichung nach oben (0.4) = 1.
r.weekends	Wie oben, nur für Wochenenden. Aktuell zählt hier ein Samstag als 1, Freitag 0.4, Sonntag 0.6 (im Sinne von ‘betroffene Wochenenden’, als Dienste werden sie natürlich separat gezählt.
r.nights	Maximaler Unterschied Anzahl N1 vs N2 bei einem Arzt – sollte 0 oder 1 sein
n.split	Wie viele der prinzipiell teilbaren Dienste (Alle Mo-Do) wurden geteilt?
day_presence	Anzahl der fehlenden Ärzte-Tage gesamt. Die Zahl in (Klammer) ist das gleiche, aber quadriert bevor es addiert wurde. Ein Beispiel:

	<p>Es fehlen je 1 Arzt an 2 Tagen: <math>1+1=2(1^2+1^2=2)</math></p> <p>Es fehlen 2 Ärzte an 1 Tag: <math>2=2(2^2=4)</math></p> <p>Mit anderen Worten: wenn die Zahl in Klammer deutlich höher ist, dann fehlen irgendwo mehrere am gleichen Tag.</p> <p>Der Wert in Klammer wird bei der Optimierung verwendet, es wird also stärker bestraft, wenn an weniger Tagen mehrere fehlen als umgekehrt</p>
--	--