Zürcher Hochschule
für Angewandte Wissenschaften

School of Engineering
IDP Institut für Datenanalyse
und Prozessdesign

FS 2016

Stochastische Prozesse

Woche 5

Aufgabe 1 PageRank-Algorithmus von Google

Das Herz der Google-Suchmaschine ist ein Algorithmus, der alle Dokumente des WWW nach ihrer Wichtigkeit anordnet. Die Auflistung der Dokumente, die unter einem bestimmten Suchbegriff gefunden werden, legt diese Rangfolge, den sogenannten *PageRank*, zugrunde.[1]

Grundidee ist es, das Surf-Verhalten eines repräsentativen Internet-Surfers als geeignete Markov-Kette zu modellieren, wobei jede Site einem Zustand entspricht. Die stationäre Verteilung gibt dann an, wie häufig dieser Surfer (wenn er sehr lange im Netz bleibt) an den verschiedenen Sites vorbeikommt. Die Seite mit der grössten asymptotischen Wahrscheinlichkeit wird als die wichtigste interpretiert und erhält den höchsten PageRank.

Die Übergangsmatrix für ein Universum aus N Websites wird nach folgenden Regeln aufgebaut:

- Die Übergangsmatrix P ist ergibt sich aus zwei Matrizen U und S durch Addition:

  $$P = \alpha \cdot S + (1-\alpha) \cdot U \quad 0 \leq \alpha \leq 1$$

  P heisst auch *Google-Matrix*. Die Erfinder von PageRank, Sergey Brin und Larry Page, verwendeten $\alpha = 0.85$. Die Matrix S entspricht dem Surf-Verhalten aufgrund der Links auf einer Seite:

  - Für eine Seite i, die l_i Links auf andere Seiten enthält, setzen wir die Übergangswahrscheinlichkeit

    $$S_{ij} = \frac{1}{l_i},$$

    falls ein Link nach j existiert. Wenn kein Link existiert setzen wir $S_{ij} = 0$. Das heisst, wir nehmen an, dass ein Surfer zufällig einem der Links folgt.
  - Für eine Seite, die keine Links hat, setzen wir $S_{ij} = 1/N$ für alle $j = 1, \ldots, N$. Das heisst, wir nehmen an, dass der Surfer zufällig irgendeine andere WebSite aufsucht.

- Die Matrix U entspricht einem rein zufälligen Surf-Verhalten ohne Links: $U_{ij} = 1/N$ für alle $i, j = 1 \ldots, N$.

Wenn wir das Surf-Verhalten nur aufgrund der Links einer Seite modellieren wollten, würden wir $\alpha = 1$ verwenden. Dies entspricht der Topologie des WWW. Die Wahl $\alpha = 0$ entspricht einem absolut zufälligen Surfen. Setzen wir $\alpha = 0.85$, ist das Verhalten dominiert von den weiterführenden Links, enthält aber auch andere Anteile, die wir als zufällig beschreiben.

[1] Literatur: David Austin, How Google finds Your Needle in the Web's Haystack. http://www.ams.org/featurecolumn/archive/pagerank.html

a) Zeigen Sie allgemein, dass P für beliebiges α eine zulässige Übergangsmatrix ist, d.h. dass $0 \leq P_{ij} \leq 1$ ist, und dass die Zeilensummen gleich 1 sind.

Es gilt

$$P_{ij} = \alpha \cdot S_{ij} + (1-\alpha) \cdot U_{ij}$$

und damit wegen $\alpha \in [0,1]$

$$P_{ij} = \alpha \cdot S_{ij} + (1-\alpha) \cdot U_{ij} \geq \alpha \cdot 0 + (1-\alpha) \cdot 0 = 0$$

und

$$P_{ij} = \alpha \cdot S_{ij} + (1-\alpha) \cdot U_{ij} \leq \alpha \cdot 1 + (1-\alpha) \cdot 1 = 1$$

und für alle i

$$\begin{aligned}\sum_{j=1}^{n} P_{ij} &= \sum_{j=1}^{n} (\alpha \cdot S_{ij} + (1-\alpha) \cdot U_{ij}) \\ &= \alpha \cdot \sum_{j=1}^{n} S_{ij} + (1-\alpha) \cdot \sum_{j=1}^{n} U_{ij} \\ &= \alpha \cdot 1 + (1-\alpha) \cdot 1 \\ &= 1\end{aligned}$$

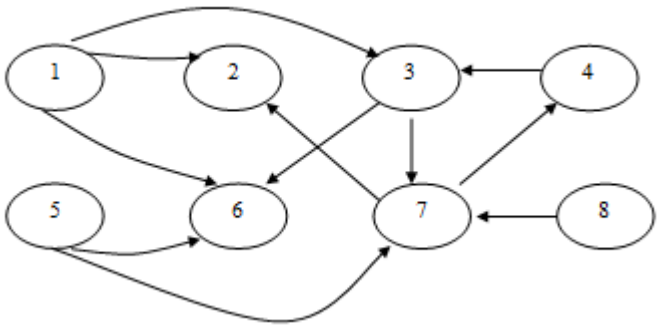
b) Zeigen Sie, dass P für $\alpha < 1$ irreduzibel und aperiodisch ist. Das heisst, dass es genau eine asymptotische Verteilung gibt.

Es gilt für alle ij wegen $(1-\alpha) > 0$

$$P_{ij} = \alpha S_{ij} + (1-\alpha) U_{ij} \geq (1-\alpha) U_{ij} = (1-\alpha) \frac{1}{N} > 0,$$

damit ist klar, dass $i \leftrightarrow j$ ist (man kann sogar in einem Schritt von jedem Zustadn in jeden anderen kommen), d.h. die Markov-Kette ist irreduzibel. Wegen $P_{ii} > 0$ für alle i ist die Kette auch aperiodisch.

c) Berechnen Sie die PageRanks für alle Seiten des folgenden Systems (Pfeile bedeuten Links) für $\alpha = 1$, $\alpha = 0.85$, und $\alpha = 0$. Sortieren Sie die Seiten nach ihrem Rang. Ändert sich die Reihenfolge der Seiten, wenn Sie von $\alpha = 1$ auf $\alpha = 0.85$ gehen?



```
dummy <- c(0, 1/3, 1/3, 0, 0, 1/3, 0, 0,
    1/8, 1/8, 1/8, 1/8, 1/8, 1/8, 1/8, 1/8,
    0, 0, 0, 0, 0, 1/2, 1/2, 0,
    0, 0, 1, 0, 0,   0,   0, 0,
    0, 0, 0, 0, 0, 1/2, 1/2, 0,
    1/8, 1/8, 1/8, 1/8, 1/8, 1/8, 1/8, 1/8,
    0, 0.5, 0.0, 0.5, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 1, 0)
  S <- matrix(dummy, ncol=8, byrow=T)
  U <- matrix(rep(1/8, 64), ncol=8)

  getPi <- function(alpha, S, U) {
    P <- alpha * S + (1 - alpha) * U
    rowSums(P)
    es <- eigen(t(P));es
    stopifnot(abs(as.double(es$values[1]) - 1.0) < 1e-6)
    d <- as.double(es$vectors[,1])
    pi <- d / sum(d)
    pi
  }
  rk100 <- getPi(1.0, S, U)
  rk100
```

```
## [1] 0.04147465 0.15668203 0.19815668 0.14285714 0.04147465 0.17511521
## [7] 0.20276498 0.04147465
```

```
  order(rk100)
```

```
## [1] 1 5 8 4 2 6 3 7
```

```
  rk085 <- getPi(0.85, S, U)
  order(rk085)
```

```
## [1] 1 5 8 4 2 6 3 7
```

```
  rk085
```

```
## [1] 0.0528533 0.1522600 0.1845205 0.1372849 0.0528533 0.1687123 0.1986625
## [8] 0.0528533
```

```
  rk000 <- getPi(0, S, U)
  order(rk000)
```

```
## [1] 1 3 4 5 6 7 8 2
```

```
  rk000
```

```
## [1] 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125
```

Die Ergebnisse sind für $\alpha = 1$ und $\alpha = 0.85$ die selben, Site 7 wird am häufigsten besucht, am seltensten Sites 1, 5 und 8, die die den selben PageRank erhalten. Für $\alpha = 0$ werden alle Sites gleich häufig besucht.

d) Nehmen Sie an, Sie sind Eigentümer der Site 5. Sie wollen Ihr Ranking verbessern, indem Sie einen Link von einer der anderen Sites auf Ihre Site erhalten. Von welcher Site aus wäre der Link am wirkungsvollsten, d.h. bei welchem zusätzlichen Link $i \to 5$ würden Sie Ihr Ranking am meisten verbessern? Wie wäre Ihr neuer Platz im Ranking, wenn Sie diesen Link erhalten würden? (Verwenden Sie hier $\alpha = 0.85$).

```
addLink <- function(i, j, S) {
  S2 <- S>0 # Zählen, wie viele Links es von i aus gibt
  if (sum(S2[i,])==dim(S2)[2]) S2[i,] <- 0  # Falls bisher keine Links mit 0 auffüllen
    S2[i,j] <- 1
  sweep(S2, 1, rowSums(S2), "/")
}

newRank <- function(i) getPi(0.85, addLink(i, 5, S), U)[5]

neuePR <- sapply(c(1:4,6:8), newRank)
names(neuePR) <- c(1:4,6:8)
neuePR
```

```
##         1         2         3         4         6         7
## 0.06332020 0.14929211 0.10092310 0.11119936 0.18325967 0.11074674
##         8
## 0.07562406
```

Am wertvollsten wäre nicht ein Link von der bisher höchstgerankten Site 7, sondern von Site 6. Von Site 7 aus gehen bislang schon 2 Links aus, von Site 6 gar keiner, weshalb das setzen eines eizigen Links auf 5 den PageRank von Seite 5 stark erhöht.

```
pr  <- getPi(0.85, addLink(6, 5, S), U)
pr
```

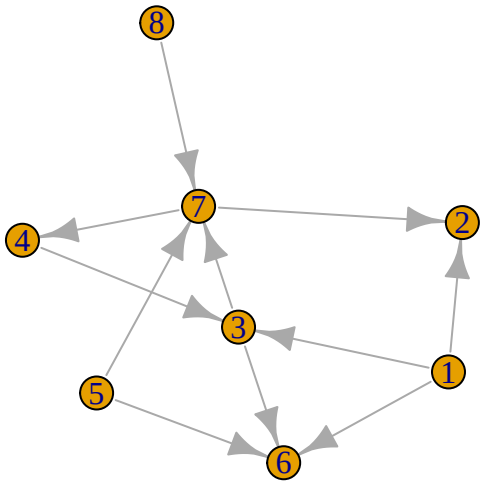
```
## [1] 0.03196391 0.12436620 0.13903365 0.11530976 0.18325967 0.17799501
## [7] 0.19610789 0.03196391
```

```
rank(pr)[5]
```

```
## [1] 7
```

Mit den zusätzlichen Link von Site 6 hat Site 5 nun den zweithöchsten PageRank.

```
  # Lösung mittels igraph
  library("igraph")
  es <- matrix(c(1,2, 1,3,  1,6,3,6,3,7,4,3,5,6,5,7, 7,2,7,4, 8,7), ncol=2, byrow=T)
  g <- graph.edgelist(es, directed=T)
  plot(g)
```




```
res <- rep(NA, 8)
for (i in c(1:4,6:8)) {
  g1 <- add.edges(g, c(i, 5))
  res[i] <- page.rank(g1)$vector[5];
}
res
```

```
## [1] 0.06332020 0.14929211 0.10092310 0.11119936         NA 0.18325967
## [7] 0.11074674 0.07562406
```

Aufgabe 2 Spezielle Zustände

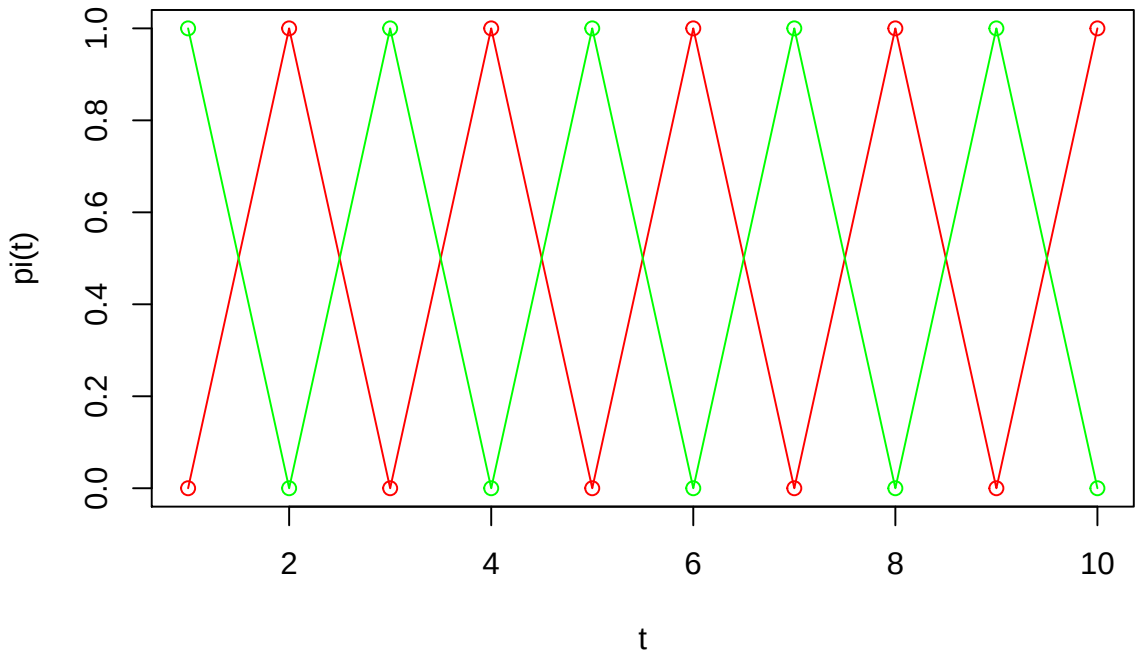
Eine Markov-Kette habe die Übergangsmatrix:

$$P = \begin{pmatrix} 0.00 & 1.0 \\ 1.00 & 0.0 \end{pmatrix}$$

a) Zeichnen Sie das Übergangsdiagramm.

b) Plotten Sie $\vec{\pi}(t)$ jeweils für die Anfangsverteilung $\vec{\pi}(0) = (1, 0)$ und $\vec{\pi}(0) = (0, 1)$ gegen die Zeit t.

```
library("expm")
P <- matrix(c(0,1,1,0), ncol=2, byrow=T)
t <- 1:10
d <- sapply(t, FUN=function(t){return ((P %^% t)[1,1])})
plot(t,d, t='o', col='red', ylab="pi(t)")
d <- sapply(t, FUN=function(t){return ((P %^% t)[1,2])})
lines(t,d, t='o', col='green')
```

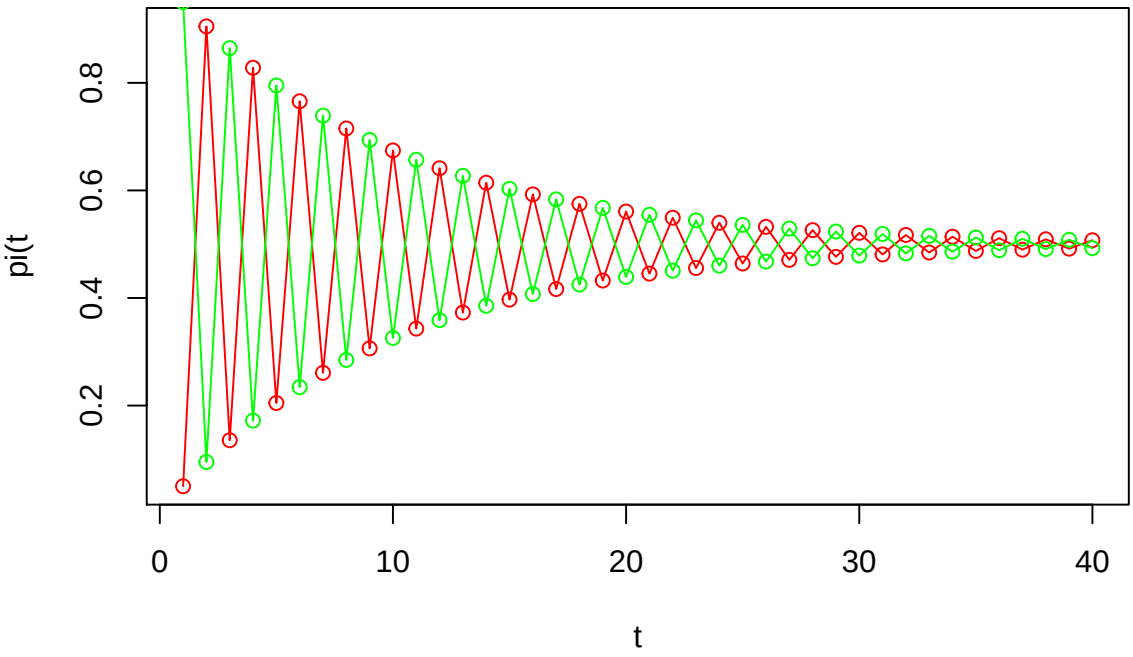


c) Ersetzen Sie nun die Übergangsmatrix durch

$$P = \begin{pmatrix} 0.05 & 0.95 \\ 0.95 & 0.05 \end{pmatrix}$$

und erstellen Sie den gleichen Plot wie in b).

```
P <- matrix(c(0.05,0.95,0.95,0.05), ncol=2, byrow=T)
t <- 1:40
d <- sapply(t, FUN=function(t){return ((P %^% t)[1,1])})
plot(t,d, t='o', col='red', ylab="pi(t")
d <- sapply(t, FUN=function(t){return ((P %^% t)[1,2])})
lines(t,d, t='o', col='green')
```


pi(t
0.2
0.4
0.6
0.8
0
10
20
30
40
t