Laboration 1

Johannes Pohjolainen

2021-02-16

```
set.seed(950914) #Global seed för att kunna replikera labbens resultat
```

Sammanfattning

En kaffemaskin har ett av 0-5 tillstånd som representerar hur väl den fungerar. Maskinen repareras ibland och då byter den tillstånd på skalan enligt följande övergångsmatris (en reparation per dag):

```
tm <- matrix(c(
  0.0, 0.0, 0.0, 0.5, 0.0, 0.5,
  0.1, 0.1, 0.0, 0.4, 0.0, 0.4,
  0.0, 0.2, 0.2, 0.3, 0.0, 0.3,
  0.0, 0.0, 0.3, 0.5, 0.0, 0.2,
  0.0, 0.0, 0.0, 0.4, 0.6, 0.0,
  0.0, 0.0, 0.0, 0.0, 0.4, 0.6),
  nrow = 6,
  ncol = 6,
  byrow = TRUE)
```

I denna laboration kommer ovanstående exempel användas för att studera olika aspekter av övergångsmatriser. Detta kommer att göras via fyra uppgifter:

- Uppgift 1 kommer undersöka hur vi beräknar övergångssannolikheter efter olika antal tidssteg.
- Uppgift 2 kommer handla om konvergens och stationära fördelningar.
- Uppgift 3 handlar om tiden som spenderas i de olika tillstånden.
- Uppgift 4 undersöker sannolikheten för gårdagens tillstånd givet dagens tillstånd.

För att behandla dessa områden kommer vi att behöva utföra en del matrisoperationer. Innan vi börjar definierar vi därför några matrisfunktioner.

```r
# Användning:
#   Multiplicerar matris med sig själv n ggr
# Input:
#   Matris A
#   Exponent n
# Output:
#   A^n
mpow <- function(A, n) {
  resultat <- diag(nrow(A))
  potens <- n
  while (potens > 0) {
    resultat <- A %*% resultat
    potens <- potens - 1
  }
  return(resultat)
}

# Användning
#   Kollar att rader i matris är identiska. d är antalet decimaler som beaktas
# Input:
#   Matris A
#   Antal decimaler d
# Output:
#   TRUE eller FALSE beroende på raderna
rows_equal <- function(A, d = 4) {
  A_new <- trunc(A * 10^d) # förstora talet och ta heltalsdelen
  for (k in 2:nrow(A_new)) {
    # Kolla om alla element i rad 1 är lika med motsvarande element i rad k
    if (!all(A_new[1, ] == A_new[k, ])) {
    # Om något element skiljer sig så är raderna ej lika
    return(FALSE)
  }
  }
  #Hamnar vi här så var alla rader lika
  return(TRUE)
}

#Som ovan fast jämför rader från matris A med B istället för alla sina egna
matrices_equal <- function(A, B, d = 4) {
  A_new <- trunc(A * 10^d)
  B_new <- trunc(B * 10^d)
  if (all(A_new == B_new)) {
    return(TRUE)
  } else {
  return(FALSE)
    }
}
```

Uppgift 1

Då vi har ett exempel som dessutom brutits ned till en markovkedja kan vi börja svara på frågor kring tidsteg.

1.1 Maskinen fungerar perfekt idag, hur ser framtiden ut?

Denna fråga kan tolkas om till att lista ut hur övergångsmatrisen ser ut efter n dagar. Tillstånd 5 (rad 6) beskriver det perfekta tillståndets övergångs-sannolikherter till andra tilltånd, så vi fokuserar på den.

Att lista ut detta görs sedan genom att upphöja matrisen i antalet dagar. Nu när vi vet "hur?", fattas "vad?". Dvs vi vill ha något att räkna på, så vi ställer samt besvarar frågor om tid nedan:

Antag att vi startar på en måndag, hur ser raden ut på onsdag, om en vecka, två veckor, tre månader?

```
#Rad efter ovanstående antal dagar:
state5dag3 = mpow(tm, 2)[6,]
state5dag7 = mpow(tm, 7)[6,]
state5dag14 = mpow(tm, 14)[6,]
state5dag90 = mpow(tm, 90)[6,]

#Skapar tabell för att visa resultaten på ett prydligt sätt:
tabell1 <- data.frame(Tid = c("På onsdag", "Om en vecka",
"Om två veckor", "Om tre månader"))
tabell1 <- cbind(tabell1, rbind(state5dag3, state5dag7, state5dag14, state5dag90))
names(tabell1)[-1] <- paste0("Tillstånd ", 0:5)
knitr::kable(tabell1, digits = 4, caption = "Tabell 1: Visar sannolikheten att byta från tillstånd 5 ef
```

Table 1: Tabell 1: Visar sannolikheten att byta från tillstånd 5 efter olika tidsintervall

	Tid	Tillstånd 0	Tillstånd 1	Tillstånd 2	Tillstånd 3	Tillstånd 4	Tillstånd 5
state5dag3	På onsdag	0.0000	0.0000	0.0000	0.1600	0.4800	0.3600
state5dag7	Om en vecka	0.0025	0.0273	0.1243	0.3226	0.2614	0.2619
state5dag14	Om två veckor	0.0026	0.0258	0.1164	0.3108	0.2725	0.2719
state5dag90	Om tre månader	0.0026	0.0259	0.1166	0.3109	0.2720	0.2720

1.2 Byta från tillstånd 3?

Nedan följer ett till exempel enligt samma tänk som ovan. Istället för från tillstånd 5 börjar vi istället i tillstånd 3 testar övergångs-sannolikheterna efter följande tidsperioder:

```
#Rad efter ovanstående antal dagar:
state3dag4 = mpow(tm, 3)[4,]
state3dag7 = mpow(tm, 7)[4,]
state3dag14 = mpow(tm, 14)[4,]
state3dag90 = mpow(tm, 90)[4,]

#Skapar tabell för att visa resultaten på ett prydligt sätt:
tabell2 <- data.frame(Tid = c("På torsdag", "Om en vecka",
"Om två veckor", "Om tre månader"))
tabell2 <- cbind(tabell2, rbind(state3dag4, state3dag7, state3dag14, state3dag90))
names(tabell2)[-1] <- paste0("Tillstånd ", 0:5)
```

```
knitr::kable(tabell2, digits = 4, caption = "Tabell 2: Visar sannolikheten att byta från tillstånd 3 ef
```

Table 2: Tabell 2: Visar sannolikheten att byta från tillstånd 3 efter olika tidsintervall

	Tid	Tillstånd 0	Tillstånd 1	Tillstånd 2	Tillstånd 3	Tillstånd 4	Tillstånd 5
state3dag4	På torsdag	0.0060	0.0480	0.1440	0.2890	0.1720	0.3410
state3dag7	Om en vecka	0.0024	0.0238	0.1105	0.3060	0.2849	0.2723
state3dag14	Om två veckor	0.0026	0.0260	0.1167	0.3108	0.2717	0.2722
state3dag90	Om tre månader	0.0026	0.0259	0.1166	0.3109	0.2720	0.2720

Uppgift 2

I Tabell 1 och 2 ovan ser vi att övergångs-sannolikheterna konvergerade när n, dvs antalet dagar, ökade. Vi såg även att raden med övergångar från tillstånd 5 liknade motsvarande rad för 3. I denna uppgift kommer vi fokusera på denna konvergens.

2.1: Hur många steg tar det innan konvergens skett inom 4 decimaler?

Som texten antyder kommer vi att beräkna hur många steg/dagar det tar innan övergångs-sannolikheterna konvergerat till den nivå att deras 4 första decimaler inte ändras i framtida tidssteg. Då vi tidigare observerat att rader konvergerar mot samma värden kommer en rad att visas upp för att undersöka vilka värden raderna konvergerat mot.

```
getkonv <-function(mtx, decimals) {
  prevMtx = mtx;
  nextMtx = mtx %*% prevMtx;

  mults = 1; #Räknar antalet multiplikationer

  while (!(matrices_equal(prevMtx,nextMtx,decimals) & rows_equal(nextMtx))) {
    mults = mults + 1;


    prevMtx = nextMtx;  #Tn

    nextMtx = mtx %*% nextMtx;    #Tn+1

  }

  result = prevMtx[1,]
  print(paste(mults, "Multiplikationer/dagar krävdes. Det genererade följande rad:"))
  print(result)
}

stat1 = getkonv(tm, 4)
```

```
## [1] "20 Multiplikationer/dagar krävdes. Det genererade följande rad:"
## [1] 0.002590576 0.025907166 0.116583911 0.310887234 0.272016668 0.272014446
```

2.2 Stationär fördelning VS konvergens

I denna del ska vi använda en ekvation för att hitta övergångsmatrisens stationära fördelning.

Vi vet att irreducibla och icke-periodiska markovkedjors stationära fördelning kan hittas via följande ekvation: $\pi T = \pi$ där π är den stationära fördelningen och T är övergångsmatrisen.

Denna typ av ekvation löses enklast via egenvektorsberäkning, då dess utseende är ett specialfall där egenvektorn har egenvärde 1. Vi (och r) är vana vid att beräkningarna sker med variablerna på höger sida av matrisen, så vi transponerar vi innan r tar över med dess egenvärdesfunktion: $(\pi T)^t = T^t\pi^t = \pi^t$

```
#Hittar stationära fördelningen för matris via egenvektor om möjligt
stationary <- function(TM) {

  #Hitta egenvärden och egenvektorer
  info <- eigen(t(TM))

  eigenvals = info$values

  #Letar efter det egenvärde som uppfyller lambda = 1 enligt ekvation. Skriver sedan ut den tillhörade
  for (i in 1:length(eigenvals)) {

    if (round(eigenvals[i],10) == 1) {
      return(Re(info$vectors[,i]/sum(info$vectors)))
    }
  }
  print("Cannot find stationary distrib m8")
}

stat2 = stationary(tm)
```

Vi får 0.0025907, 0.0259067, 0.1165803, 0.3108808, 0.2720207, 0.2720207 som vår stationära fördelning. Detta är likt resultaten ur 2.1 vilket antyder på att konvergens sker mot den stationära fördelningen i detta fall.

Uppgift 3

Över tid kommer kaffemaskinens skick att variera enligt tillstånden i övergångsmatrisen. I denna uppgift är vi intresserade av hur många gånger vi hamnar i varje tillstånd för 1000 tidssteg. Fördelningen borde vara snarlik den stationära fördelningen och vi kan testa detta genom att simulera 1000 övergångar med följande kod:

```
#Givet starttillstånd och övergångsmatris slumpar denna funktion fram nästa tillstånd
#Input:
#Starttillstånd x
#Övergångsmatris P
#Output:
#Nästa tillstånd, next
gen_next_state <- function(x, P) {
  #Väljer ut sannolikheter för övergång från x ()
  picker <- runif(1)
  #Skapar sannolikhet, markerar nästa tillstånd
  next_state <- 0
  #Väljer ut sannolikheter för övergång från x. Vill loopa igenom kolonnerna
  state_interval <- P[x + 1, 1]
  #Delar upp [0,1] enligt sannolikheterna i P. Loopar sedan igenom för att hitta vilket tillstånd vi ha
  while (picker >= state_interval) {
    next_state <- next_state + 1
    state_interval <- state_interval + P[x + 1, next_state + 1]
  }
  next_state
}
```

```r
# Simulera n tillstånd i en diskret Markovkedja som definieras av
# övergångsmatrisen P, där initialtillståndet är x.
simulera_kedja <- function(x, n, P) {
  results <- numeric(n + 1) # vektor av nollor, till en början
  results[1] <- x # första elementet är initialtillståndet
  for (i in 2:n) {
   results[i] <- gen_next_state(results[i - 1], P) # simulera övriga element
  }
  # ta bort initialtillståndet och returnera vektorn med simulerade tillstånd
  results[-1]
}

simed = simulera_kedja(5,1000,tm)
```

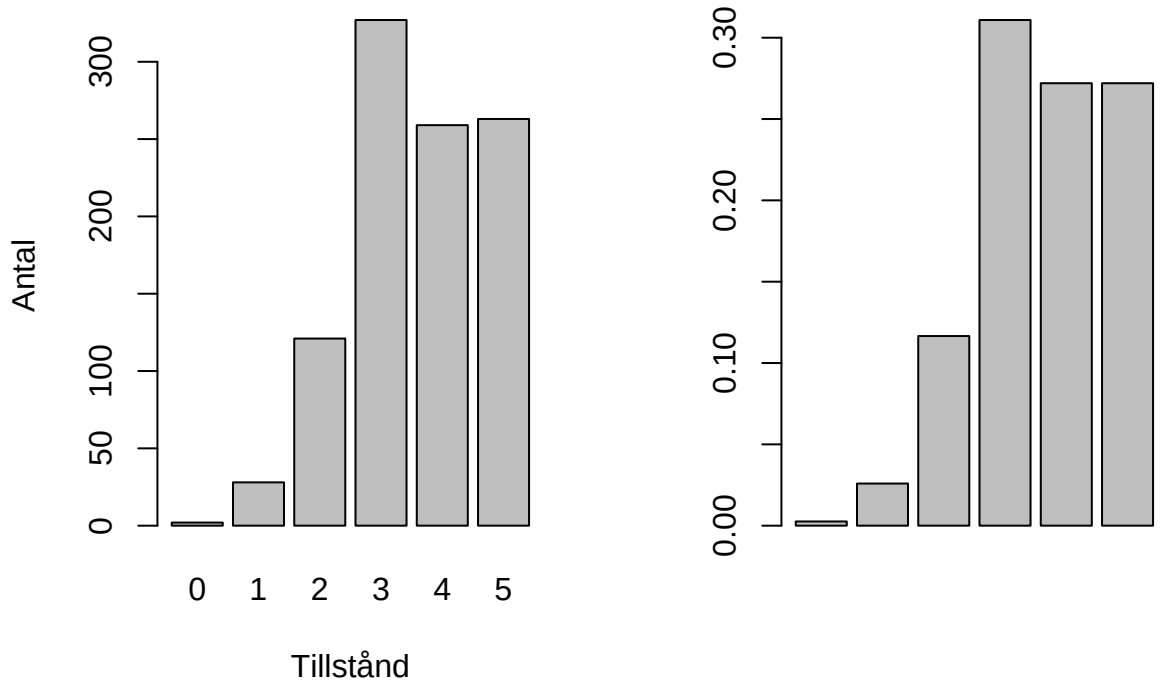

Figure 1: Figur 1: Jämförelse mellan antalet tidssteg i tillstånden (vänster) och stationära fördelningen (höger)

I Figur 1 ser vi att antalet gånger vi hamnade i varje tillstånd var i stil med den stationära fördelningen.

Uppgift 4

Vad är sannolikheten att vi var i tillstånd 1 igår om vi är i tillstånd 5 idag. Precis det, dvs $P(X_n = 1|X_{n+1} = 5)$, är vad vi kommer att beräkna i denna uppgift. Vi kommer att beräkna detta på två sätt i denna uppgift.

4.1 Övergångssannolikhet från simulering

Då vi simulerat 1000 övergångar kan vi beräkna sannolikheten på ett mindre teoretiskt plan genom att räkna antalet gånger vi gått från 1 till 5 och dela med antalet övergångar. Vi börjar med att räkna antalet övergångar:

```
went <- function(prev, current, Trans) {
  fromOne = 0
  toFive = 0
  for (i in 2:length(Trans)) {
    if (Trans[i]==current) {
      toFive = toFive + 1
      if(Trans[i-1] == prev) {
        fromOne = fromOne + 1
      }
    }
  }
  fromOne/toFive
}

went(1, 5, simed)
```

```
## [1] 0.01908397
```

4.2 Övergångssannolikhet teoretiskt

Vi kan även beräkna ovan teoretiskt via $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$ som i vårt fall blir $\frac{P(X_{n+1}=5|X_n=1)P(X_n=1)}{P(X_{n+1}=5)}$. $P(X_n = 5)$ och $P(X_n = 1)$ får vi från den stationära fördelningen medans $P(X_{n+1} = 5|X_n = 1))$ fås från övergångsmatrisen.

```
res = tm[2,6]*stat2[2]/stat2[6]
```

Vi får därav svaret $P(X_n = 1|X_{n+1} = 5)$=0.0380952