Labb 3 - Förgreningsprocesser

Sebastijan Babic & Esbjörn Runesson

2025-03-10

Contents

0.1 Uppgift 1

Reflektera över systemet som definierar födelse-dödsprocessen. Skriv ned vad fördelningen för tiden till nästa tillståndsbyte är och vad sannolikheten att processen går upp eller ned ett steg är, beroende på processens rådande tillstånd. När ni gjort detta för generella och μ, gör det specifikt för förslag 1 också.

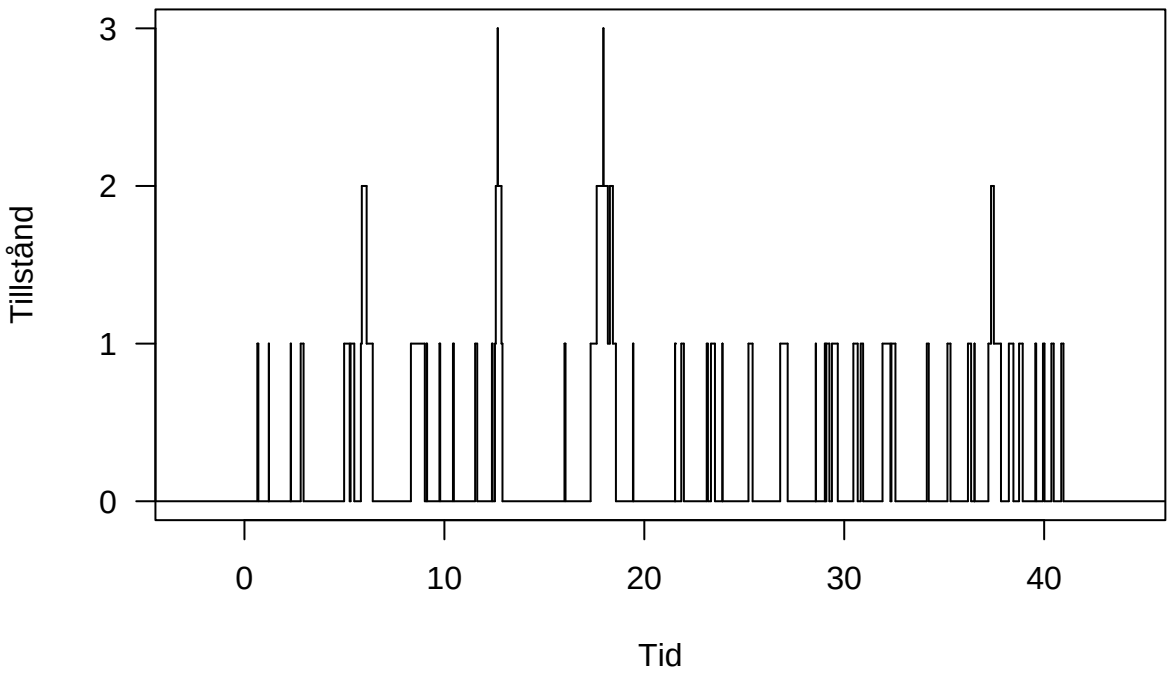
Fördelningen för tiden till nästa tillståndsbyte är $Exp(\lambda + \mu)$ och sannolikheterna är för tillstånd 0: $\lambda_0 = 1, \mu_0 = 0$, 1: $\lambda_1 = \frac{\lambda}{\lambda+\mu}, \mu_1 = \frac{\mu}{\lambda+\mu}$, 2: $\lambda_2 = \frac{\lambda}{\lambda+\mu}, \mu_2 = \frac{\mu}{\lambda+\mu}$ och för 3: $\lambda_3 = 0, \mu_3 = 1$ För förslag 1 har vi att tiden till nästa tillståndsbyte är $Exp(6)$ och sannolikheterna för tillstånd 0: $\lambda_0 = 1, \mu_0 = 0$, 1: $\lambda_1 = \frac{1}{6}, \mu_1 = \frac{5}{6}$, 2: $\lambda_2 = \frac{1}{6}, \mu_2 = \frac{5}{6}$ och för 3: $\lambda_3 = 0, \mu_3 = 1$

0.2 Uppgift 2

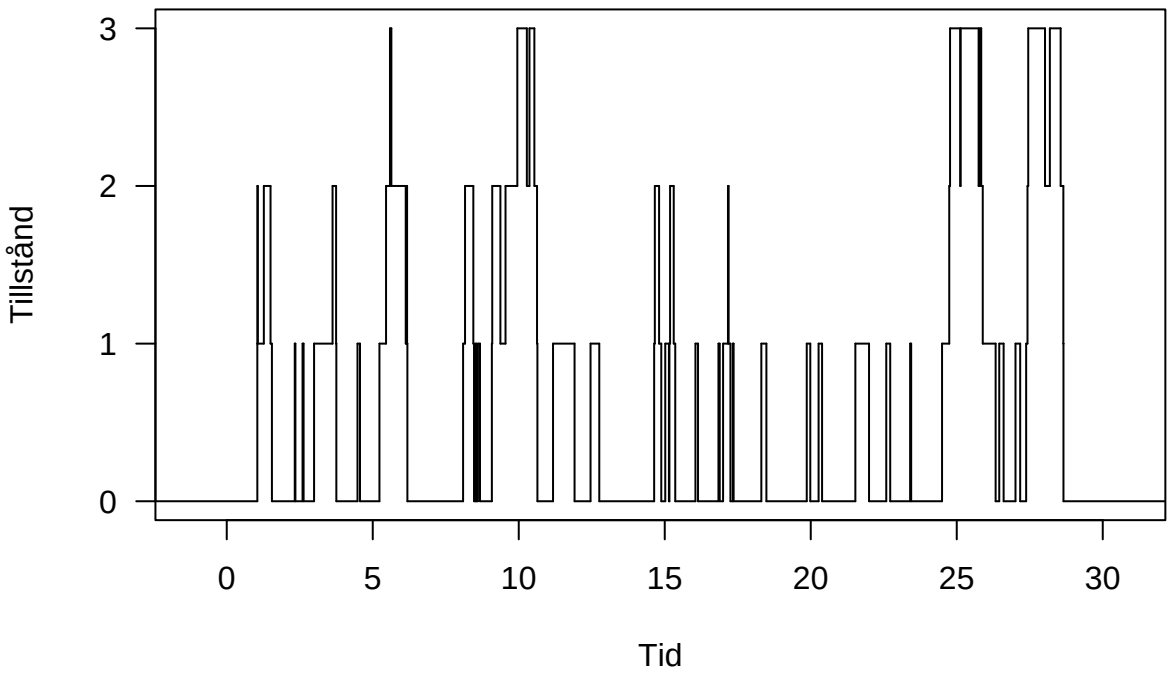
Rita upp X(t) som funktion av t, d.v.s. en kurva som rör sig språngvis mellan de fyra värdena 0, 1, 2 och 3. För att göra detta ska ni skapa en vektor time, som innehåller tidpunkterna för sprången, och en vektor state, där motsvarande element anger vilket tillstånd språnget leder till. Vi låter vektorn state börja med en nolla, eftersom det är initialtillståndet, och vektorn time har en nolla som första element eftersom vi börjar tidsräkningen därifrån

```
set.seed(980608)
bd_process <- function(lambda, mu, initial_state = 0, steps = 100) {
  time_now <- 0
  state_now <- initial_state

  # Dessa vektorer ska byggas på i loopen nedan
  time <- 0
  state <- initial_state

  for (i in 1:steps) {
    # Bestäm lambda och mu baserat på aktuellt tillstånd
    if (state_now == 3) {
      lambda_now <- 0  # Inga fler ankomster när systemet är fullt
    } else {
      lambda_now <- lambda
    }
```

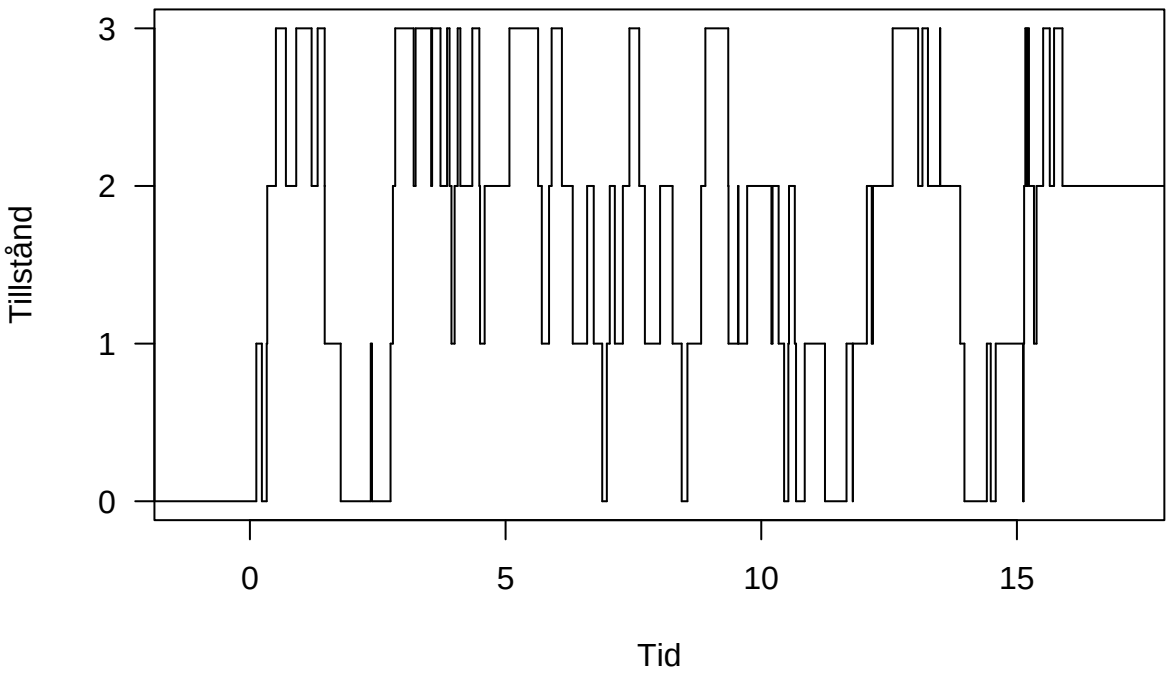
```r
    if (state_now == 0) {
      mu_now <- 0  # Inga avslut när systemet är tomt
    } else {
      mu_now <- mu
    }

    # Tiden till nästa tillståndsbyte är exponentialfördelad med parameter lambda_now + mu_now
    time_to_transition <- rexp(1, rate = lambda_now + mu_now)

    # Bestäm om nästa tillstånd är upp eller ner
    if (lambda_now == 0) {
      # Om lambda_now är 0, måste vi gå ner
      state_now <- state_now - 1
    } else if (mu_now == 0) {
      # Om mu_now är 0, måste vi gå upp
      state_now <- state_now + 1
    } else {
      # Annars, slumpa om vi går upp eller ner
      if (runif(1) < lambda_now / (lambda_now + mu_now)) {
        state_now <- state_now + 1
      } else {
        state_now <- state_now - 1
      }
    }

    # Uppdatera tiden och lägg till i vektorerna
    time_now <- time_now + time_to_transition
    time <- c(time, time_now)
    state <- c(state, state_now)
  }

  # Returnera en lista med de två vektorerna tid och state
  list(time = time, state = state)
}

# Exempel på användning för förslag 1 (lambda = 1, mu = 5)

forslag1 <- bd_process(lambda = 1, mu = 5)
forslag2 <- bd_process(lambda = 2, mu = 5)
forslag3 <- bd_process(lambda = 5, mu = 5)
time1 <- forslag1$time
state1 <- forslag1$state
time2 <- forslag2$time
state2 <- forslag2$state
time3 <- forslag3$time
state3 <- forslag3$state
plot(stepfun(time1[-1], state1),
do.points = FALSE,
xlab = "Tid",
ylab = "Tillstånd",
main = "",
yaxt = "n")
axis(2, at = c(0, 1, 2, 3), las = 2)
```




```
plot(stepfun(time2[-1], state2),
do.points = FALSE,
xlab = "Tid",
ylab = "Tillstånd",
main = "",
yaxt = "n")
axis(2, at = c(0, 1, 2, 3), las = 2)
```




```
plot(stepfun(time3[-1], state3),
do.points = FALSE,
xlab = "Tid",
ylab = "Tillstånd",
main = "",
yaxt = "n")
axis(2, at = c(0, 1, 2, 3), las = 2)
```



0.3 Uppgift 3

Räkna ut hur lång tid det tog innan systemet hade ändrat tillstånd 500 gånger, för alla tre förslag. För att göra detta behöver du kalla funktionen på nytt för de tre förslagen, med ett nytt värde på steps (vilket?). Jämför tiderna mellan de tre förslagen och reflektera över varför de skiljer sig åt (eller inte).

```
set.seed(980608)
calculate_time_for_500_steps <- function(lambda, mu) {
  result <- bd_process(lambda, mu, steps = 500)
  total_time <- result$time[length(result$time)]  # Sista värdet i time-vektorn
  return(total_time)
}

# Beräkna tiderna för de tre förslagen
time_proposal_1 <- calculate_time_for_500_steps(lambda = 1, mu = 5)
time_proposal_2 <- calculate_time_for_500_steps(lambda = 2, mu = 5)
time_proposal_3 <- calculate_time_for_500_steps(lambda = 5, mu = 5)

cat("Tid för 500 tillståndsbyten (förslag 1):", time_proposal_1, "timmar\n")
```

```
## Tid för 500 tillståndsbyten (förslag 1): 229.6018 timmar
```

```
cat("Tid för 500 tillståndsbyten (förslag 2):", time_proposal_2, "timmar\n")
```

```
## Tid för 500 tillståndsbyten (förslag 2): 125.5418 timmar
```

```
cat("Tid för 500 tillståndsbyten (förslag 3):", time_proposal_3, "timmar\n")
```

```
## Tid för 500 tillståndsbyten (förslag 3): 65.70487 timmar
```

Vi kan här se att det tog ca 229.6 timmar att ändra tillstånd 500 gånger för förslag 1, ca 125.5 timmar för förslag 2 och ca 65.7 timmar för förslag 3. Anledningen till att de skiljer sig åt, att tiden minskar då vi ökar värdet på λ, är att då värdet på λ stiger ökar antalet jobb som ankommer per timme. Systemet slutför i genomsnitt 5 jobb på 0.2 timmar och desto fler jobb som ankommer per timme desto snabbare ändrar systemet tillstånd vilket leder till att det tar kortare tid att ändra tillstånd 500 gånger.

0.4 Uppgift 4

1) Skriv en funktion proportion_in_state som tar ett tillstånd s (0, 1, 2 eller 3) och en simulerad födelse-dödsprocess bdp (som t.ex. variabeln forslag1 ovan) och returnerar andelen tid som processen spenderade i det givna tillståndet. Alltså, det är fördelningen av tiden (t.ex. i procent) som efterfrågas, inte hur många tidsenheter.

```r
# set.seed(980608)
proportion_in_state <- function(s, bdp) {
  # Extrahera tider och tillstånd från bdp
  time <- bdp$time
  state <- bdp$state

  # Beräkna tiden som spenderats i varje tillstånd
  time_in_state <- 0
  for (i in 1:(length(time) - 1)) {
    if (state[i] == s) {
      # Lägg till tiden mellan time[i] och time[i + 1]
      time_in_state <- time_in_state + (time[i + 1] - time[i])
    }
  }

  # Beräkna total tid
  total_time <- max(time)

  # Beräkna andelen tid i tillstånd s
  proportion <- time_in_state / total_time
  return(proportion)
}

# Beräkna andelen tid i tillstånd 0
proportion_state_0 <- proportion_in_state(0, forslag1)
cat("Andel tid i tillstånd 0:", proportion_state_0*100, "%\n")
```

```
## Andel tid i tillstånd 0: 80.14705 %
```

```r
# Beräkna andelen tid i tillstånd 1
proportion_state_1 <- proportion_in_state(1, forslag1)
cat("Andel tid i tillstånd 1:", proportion_state_1*100, "%\n")
```

```
## Andel tid i tillstånd 1: 16.49643 %
```

```r
# Beräkna andelen tid i tillstånd 2
proportion_state_2 <- proportion_in_state(2, forslag1)
cat("Andel tid i tillstånd 2:", proportion_state_2*100, "%\n")
```

```
## Andel tid i tillstånd 2: 3.307559 %
```

```r
# Beräkna andelen tid i tillstånd 3
proportion_state_3 <- proportion_in_state(3, forslag1)
cat("Andel tid i tillstånd 3:", proportion_state_3*100, "%\n")
```

```
## Andel tid i tillstånd 3: 0.0489676 %
```

2) Kalla nu på funktionen bd_process på nytt med steps = 1000 för vart och ett av förslagen, och räkna för vart och ett av dem ut hur stor del av tiden som har tillbringats i tillstånd 0, 1, 2 respektive 3. Avgör vilket eller vilka av förslagen 1–3 som är acceptabla för institutionsstyrelsen.

```r
# set.seed(980608)
forslag1000.1 <- bd_process(1,5,0,1000)
forslag1000.2 <- bd_process(2,5,0,1000)
forslag1000.3 <- bd_process(5,5,0,1000)

# Beräkna andelen tid i tillstånd 0
proportion_state_0 <- proportion_in_state(0, forslag1000.1)
cat("Andel tid i tillstånd 0 för förslag 1:", proportion_state_0*100, "%\n")
```

```
## Andel tid i tillstånd 0 för förslag 1: 79.76651 %
```

```r
# Beräkna andelen tid i tillstånd 1
proportion_state_1 <- proportion_in_state(1, forslag1000.1)
cat("Andel tid i tillstånd 1 för förslag 1:", proportion_state_1*100, "%\n")
```

```
## Andel tid i tillstånd 1 för förslag 1: 15.53486 %
```

```r
# Beräkna andelen tid i tillstånd 2
proportion_state_2 <- proportion_in_state(2, forslag1000.1)
cat("Andel tid i tillstånd 2 för förslag 1:", proportion_state_2*100, "%\n")
```

```
## Andel tid i tillstånd 2 för förslag 1: 3.892017 %
```

```r
# Beräkna andelen tid i tillstånd 3
proportion_state_3 <- proportion_in_state(3, forslag1000.1)
cat("Andel tid i tillstånd 3 för förslag 1:", proportion_state_3*100, "%\n")
```

```
## Andel tid i tillstånd 3 för förslag 1: 0.8066132 %
```

```r
# Beräkna andelen tid i tillstånd 0
proportion_state_0 <- proportion_in_state(0, forslag1000.2)
cat("Andel tid i tillstånd 0 för förslag 2:", proportion_state_0*100, "%\n")
```

```
## Andel tid i tillstånd 0 för förslag 2: 63.80987 %
```

```r
# Beräkna andelen tid i tillstånd 1
proportion_state_1 <- proportion_in_state(1, forslag1000.2)
cat("Andel tid i tillstånd 1 för förslag 2:", proportion_state_1*100, "%\n")
```

```
## Andel tid i tillstånd 1 för förslag 2: 25.47639 %
```

```r
# Beräkna andelen tid i tillstånd 2
proportion_state_2 <- proportion_in_state(2, forslag1000.2)
cat("Andel tid i tillstånd 2 för förslag 2:", proportion_state_2*100, "%\n")
```

```
## Andel tid i tillstånd 2 för förslag 2: 8.170303 %
```

```r
# Beräkna andelen tid i tillstånd 3
proportion_state_3 <- proportion_in_state(3, forslag1000.2)
cat("Andel tid i tillstånd 3 för förslag 2:", proportion_state_3*100, "%\n")
```

```
## Andel tid i tillstånd 3 för förslag 2: 2.543442 %
```

```
# Beräkna andelen tid i tillstånd 0
proportion_state_0 <- proportion_in_state(0, forslag1000.3)
cat("Andel tid i tillstånd 0 för förslag 3:", proportion_state_0*100, "%\n")
```

```
## Andel tid i tillstånd 0 för förslag 3: 22.81363 %
```

```
# Beräkna andelen tid i tillstånd 1
proportion_state_1 <- proportion_in_state(1, forslag1000.3)
cat("Andel tid i tillstånd 1 för förslag 3:", proportion_state_1*100, "%\n")
```

```
## Andel tid i tillstånd 1 för förslag 3: 22.40378 %
```

```
# Beräkna andelen tid i tillstånd 2
proportion_state_2 <- proportion_in_state(2, forslag1000.3)
cat("Andel tid i tillstånd 2 för förslag 3:", proportion_state_2*100, "%\n")
```

```
## Andel tid i tillstånd 2 för förslag 3: 25.33671 %
```

```
# Beräkna andelen tid i tillstånd 3
proportion_state_3 <- proportion_in_state(3, forslag1000.3)
cat("Andel tid i tillstånd 3 för förslag 3:", proportion_state_3*100, "%\n")
```

```
## Andel tid i tillstånd 3 för förslag 3: 29.44588 %
```

Enligt dessa simuleringar kan vi därför dra slutsatsen att att förslag 1 och 2 är acceptabla för institutionssyrelsen.

0.5 Uppgift 5

Ange en formel för den stationära fördelningen som funktion av kvoten $\rho = \lambda/$. Bestäm den stationära fördelningen för vart och ett av de tre förslagen. För vilka förslag "borde" systemet få godkänt?

Den stationära fördelningen $\pi = (\pi_0, \pi_1, \pi_2, \pi_3)$ ges av $\pi_i = \pi_0 \cdot \rho^i$ där $\rho = \frac{\lambda}{\mu}$. Vi beräknar $\pi_0 = \frac{1}{1+\rho+\rho^2+\rho^3}$ och sedan $\pi_1 = \pi_0 \cdot \rho$, $\pi_2 = \pi_0 \cdot \rho^2$, $\pi_3 = \pi_0 \cdot \rho^3$.

För förslag 1 har vi $\rho = 0.2$, $\pi_0 = \frac{1}{1+\rho+\rho^2+\rho^3} \approx 0.801$, $\pi_1 = \pi_0 \cdot \rho \approx 0.16$, $\pi_2 = \pi_0 \cdot \rho^2 \approx 0.032$ och $\pi_3 = \pi_0 \cdot \rho^3 \approx 0.006$ vilket ger oss den stationära fördelningen $\pi = (\pi_0, \pi_1, \pi_2, \pi_3) = (0.801, 0.16, 0.032, 0.006)$.

För förslag 2 har vi $\rho = 0.4$, $\pi_0 = \frac{1}{1+\rho+\rho^2+\rho^3} \approx 0.616$, $\pi_1 = \pi_0 \cdot \rho \approx 0.246$, $\pi_2 = \pi_0 \cdot \rho^2 \approx 0.098$ och $\pi_3 = \pi_0 \cdot \rho^3 \approx 0.039$ vilket ger oss den stationära fördelningen $\pi = (\pi_0, \pi_1, \pi_2, \pi_3) = (0.616, 0.246, 0.098, 0.039)$.

För förslag 3 har vi $\rho = 1$, $\pi_0 = \frac{1}{1+\rho+\rho^2+\rho^3} = 0.25$, $\pi_1 = \pi_0 \cdot \rho = 0.25$, $\pi_2 = \pi_0 \cdot \rho^2 = 0.25$ och $\pi_3 = \pi_0 \cdot \rho^3 = 0.25$ vilket ger oss den stationära fördelningen $\pi = (\pi_0, \pi_1, \pi_2, \pi_3) = (0.25, 0.25, 0.25, 0.25)$.

Eftersom $\pi_3 < 0.05$ för förslag 1 och 2 genom beräkning av den stationära fördelningen så kan vi dra slutsatsen att dessa förslag är acceptabla för institutionsstyrelsen.