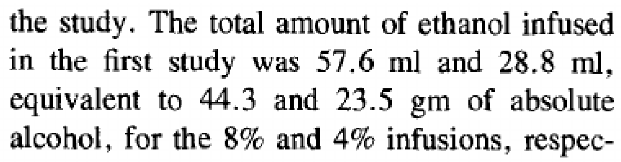
Homework

Lev Mazaev

October 3, 2019

First part

In *Wilkinson et al.* it is mentioned that the absolute amount of intravenously infused alcohol was 44.3 g.



the study. The total amount of ethanol infused
in the first study was 57.6 ml and 28.8 ml,
equivalent to 44.3 and 23.5 gm of absolute
alcohol, for the 8% and 4% infusions, respec-

Then we can calculate the input rate of ethanol the following way:

$$Q_{iv} = \frac{44.3\,\mathrm{g}}{120\,\mathrm{min}} \cdot \frac{1}{46.069\,\mathrm{g\,mol^{-1}}} \approx 0.008\,\mathrm{mol/\,min}$$

Initial constants:

```r
v_body <- 48; v_liver <- 0.61 # l
vmax <- 2.75 # mmol/min
km <- 0.1 # mM
fhv <- 1.5 # l/min
qiv <- 1000 * (44.3 / 120) / 46.069 # mmol/min
```

Grid:

```r
# 10000 steps per minute, 8 hours
step = 10000
steps <- 8 * 60 * step
cb <- numeric(length = steps)
cl <- numeric(length = steps)
dt <- 1 / step
```

Iterating, adding predicted:

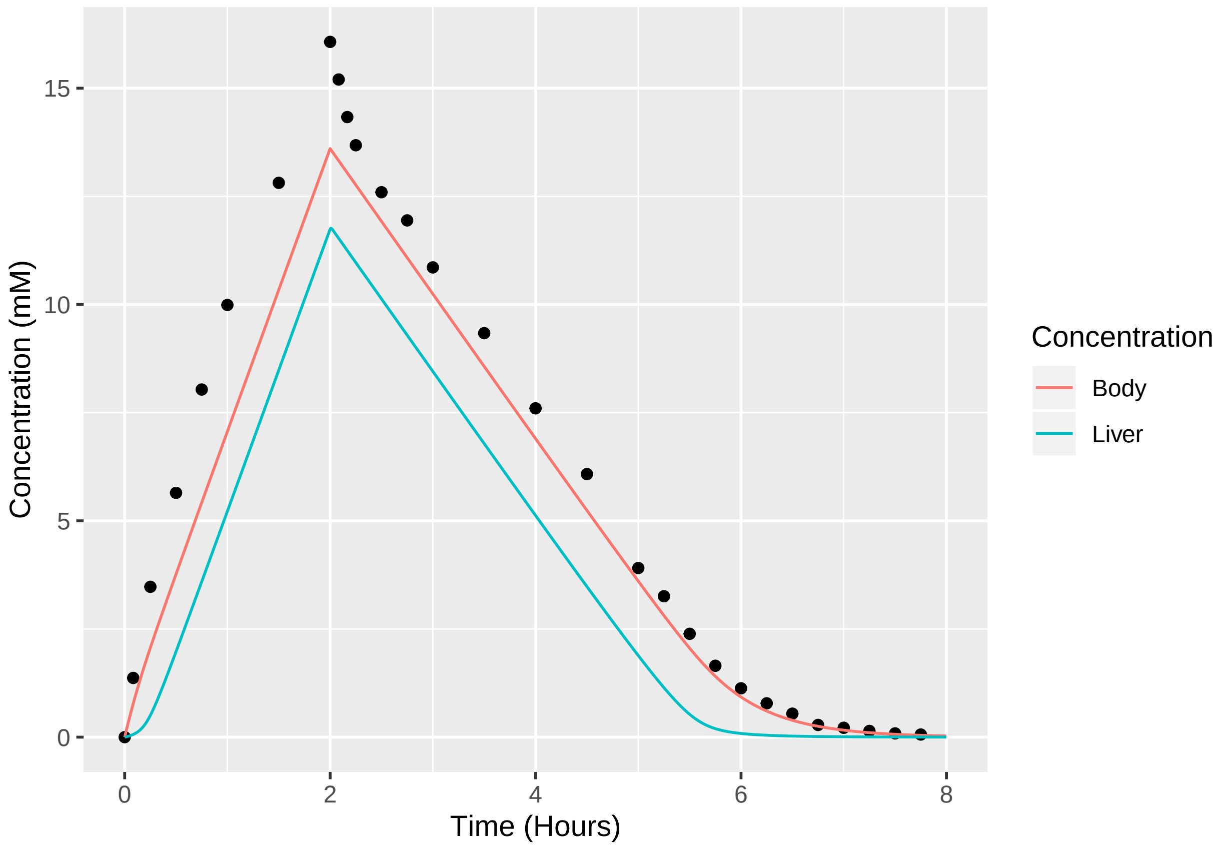
```r
th <- 2 * 60 * step
for (i in 2:steps) {
  cb[i] <- cb[i-1] + dt * (fhv * (cl[i-1] - cb[i-1]) + qiv * (i <= th)) / v_body
  cl[i] <- cl[i-1] + dt * (fhv * (cb[i-1] - cl[i-1]) -
                              (vmax * cl[i-1]) / (km + cl[i-1])) / v_liver
}
t = seq.int(from = 1, to = steps, by = 1000)
predicted <- data.frame(
  t = t,
  'Body' = cb[t],
  'Liver' = cl[t]
)
predicted$t <- predicted$t / step / 60
predicted <- predicted %>% pivot_longer(-t, names_to = 'type', values_to = 'conc')
```

Adding observed data:

```
observed <- data.frame(
  t = c(
    0.0, 0.083, 0.25, 0.5, 0.75, 1.0, 1.5, 2.0, 2.083, 2.167, 2.25, 2.5,
    2.75, 3.0, 3.5, 4.0, 4.5, 5.0, 5.25, 5.5, 5.75, 6.0, 6.25, 6.5,
    6.75, 7.0, 7.25, 7.5, 7.75
  ),
  ec = c(
    0.0, 0.063, 0.16, 0.26, 0.37, 0.46, 0.59, 0.74, 0.7, 0.66, 0.63, 0.58,
    0.55, 0.5, 0.43, 0.35, 0.28, 0.18, 0.15, 0.11, 0.076, 0.052, 0.036, 0.025,
    0.013, 0.010, 0.0066, 0.0040, 0.0028
  )
)
observed$ec <- 1000 * observed$ec / 46.049 # mM
```

Plotting:

```
ggplot() +
  geom_point(data = observed,
             mapping = aes(x = t, y = ec)) +
  xlab('Time (Hours)') +
  ylab('Concentration (mM)') +
  geom_line(data = predicted,
            mapping = aes(x = t, y = conc, color = type)) +
  labs(color = 'Concentration')
```

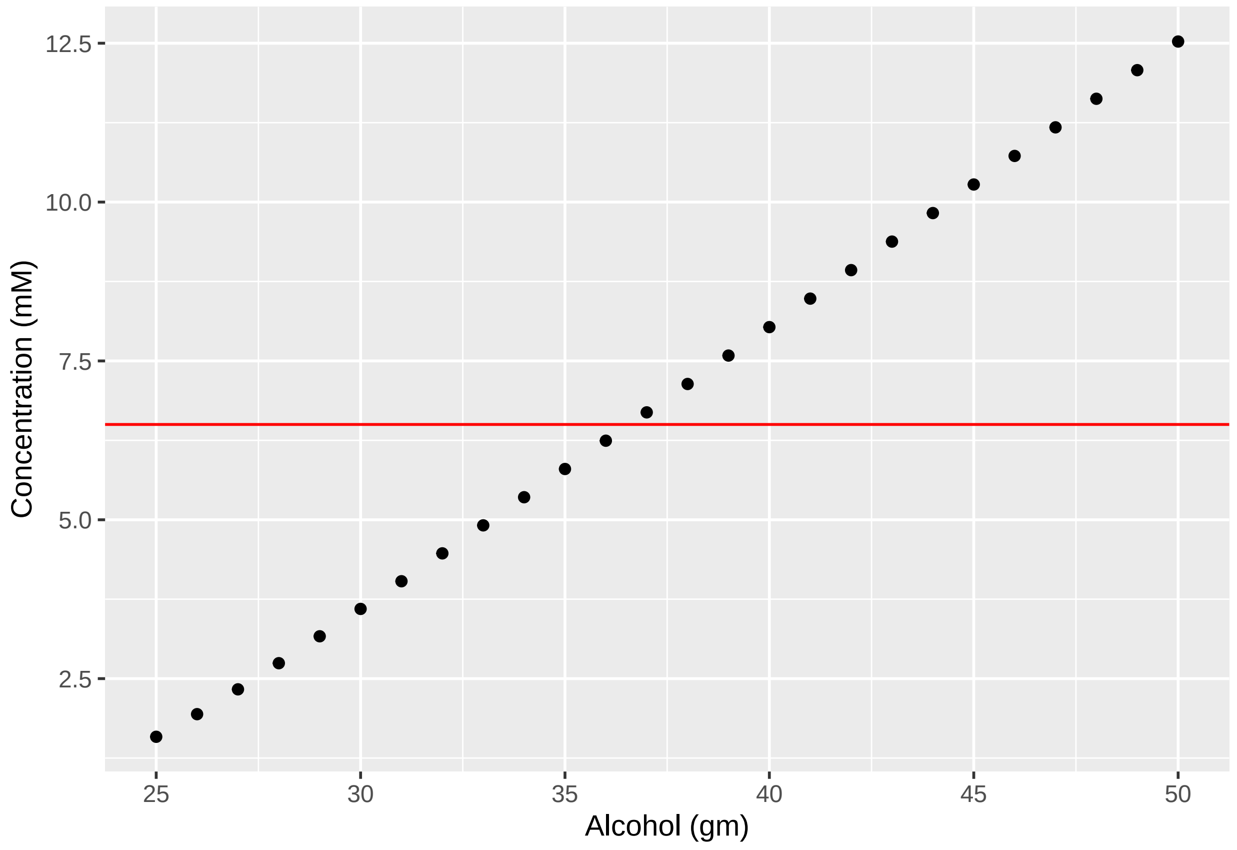


Black dots correspond to points from the paper (observed data).

Let's try to answer the question about vodka amount by searching initial conditions (integers only). Assume that ethanol concentration in body water is equal to that of blood.

```r
conc_after_3_hrs <- function(alc) {
  alc <- 1000 * (alc / 46.049) / v_body # grams to mmol/l
  step = 1000
  steps <- 3 * 60 * step
  cb <- numeric(length = steps)
  cl <- numeric(length = steps)
  cb[1] <- alc
  cl[1] <- alc
  dt <- 1 / step
  for (i in 2:steps) {
  cb[i] <- cb[i-1] + dt * (fhv * (cl[i-1] - cb[i-1])) / v_body
  cl[i] <- cl[i-1] + dt * (fhv * (cb[i-1] - cl[i-1]) -
                              (vmax * cl[i-1]) / (km + cl[i-1])) / v_liver
  }
  return(cb[steps])
}

data <- data.frame(alcohol = 25:50)
data$concentration <- sapply(data$alcohol, conc_after_3_hrs)
ggplot(data = data, mapping = aes(x = alcohol, y = concentration)) +
  geom_point() +
  geom_hline(yintercept = 6.5, color = 'red') +
  xlab('Alcohol (gm)') +
  ylab('Concentration (mM)')
```



The max amount of alcohol is 36 grams. That corresponds to ≈ 114 ml of vodka.

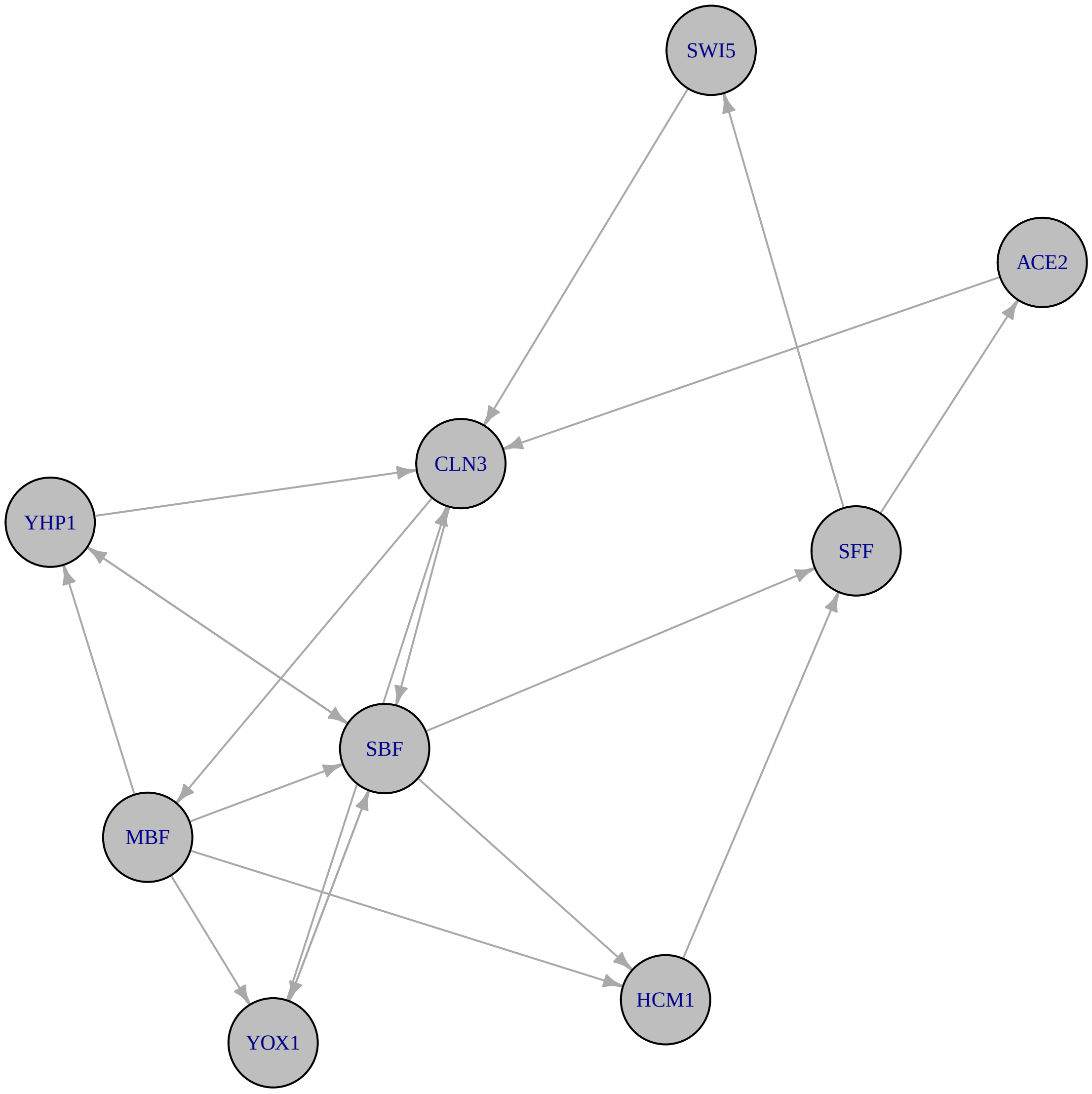
Second part

1. Assembling a network from natural-language statements

```
geneNet <- loadNetwork('mynetwork.txt')
geneNet
```

```
## Boolean network with 9 genes
##
## Involved genes:
## MBF SBF YOX1 HCM1 YHP1 SFF ACE2 SWI5 CLN3
##
## Transition functions:
## MBF = CLN3
## SBF = (CLN3 | MBF) & (! YOX1 | ! YHP1)
## YOX1 = (MBF & SBF)
## HCM1 = (MBF & SBF)
## YHP1 = (MBF | SBF)
## SFF = (SBF & HCM1)
## ACE2 = SFF
## SWI5 = SFF
## CLN3 = (SWI5 & ACE2) & (! YOX1 | ! YHP1)
```

```
plotNetworkWiring(geneNet)
```



2. Markov chain simulation

```
x <- markovSimulation(geneNet, cutoff = 0.000000001)
x$reachedStates
```

```
##   MBF SBF YOX1 HCM1 YHP1 SFF ACE2 SWI5 CLN3 Probability
## 1   0   0    0    0    0   0    0    0    0   0.1972656
## 2   1   1    0    0    0   0    0    0    0   0.1406250
## 3   0   1    1    1    1   0    0    0    0   0.1113281
## 4   0   0    0    0    1   1    0    0    0   0.1992188
## 5   0   0    0    0    0   0    1    1    0   0.0703125
## 6   0   0    0    0    0   0    0    0    1   0.2812500
```

Only 6 states have non-zero probability.

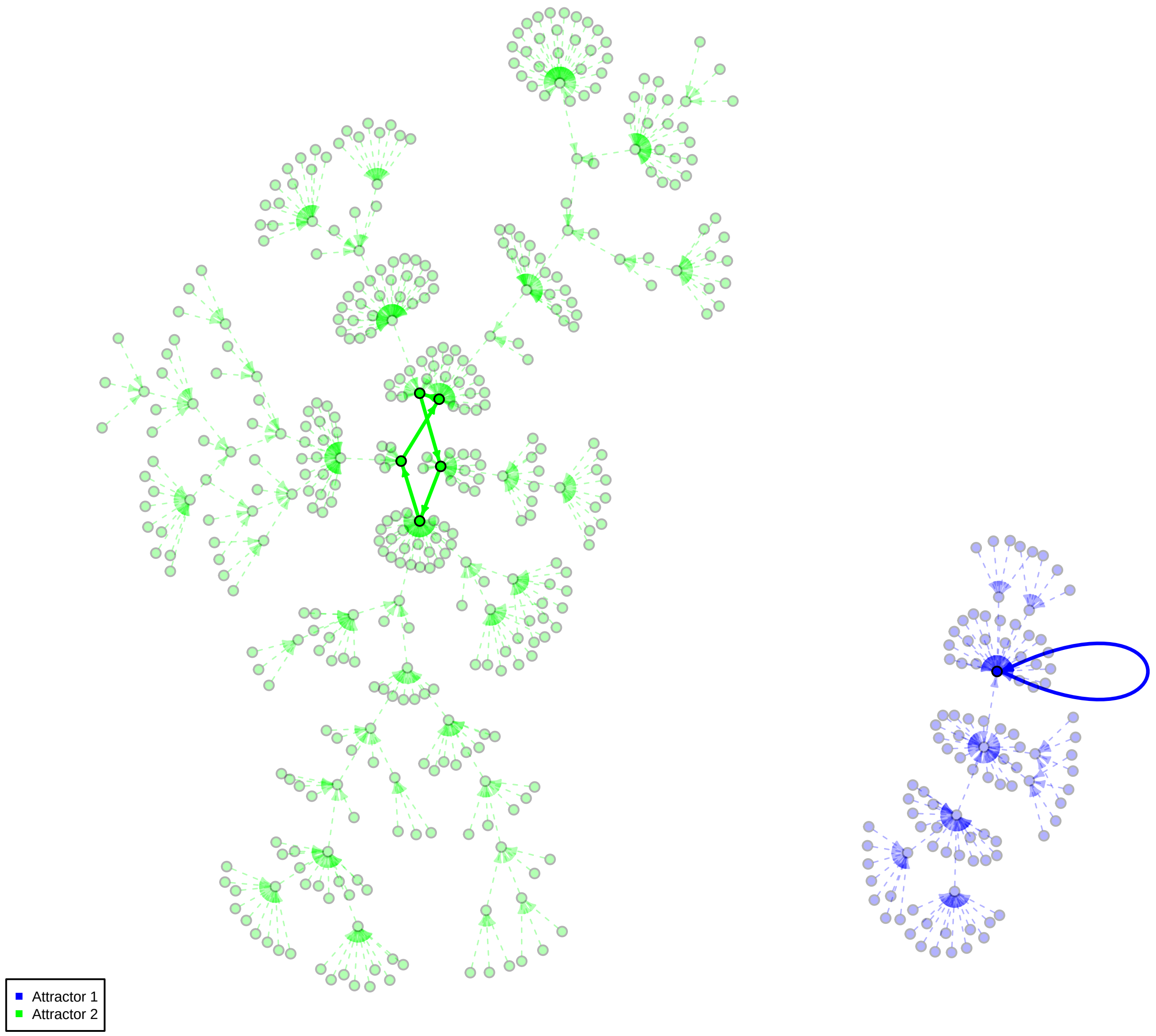
3. Attractors

```r
x <- getAttractors(geneNet, type = 'synchronous', method = 'exhaustive')
x
```

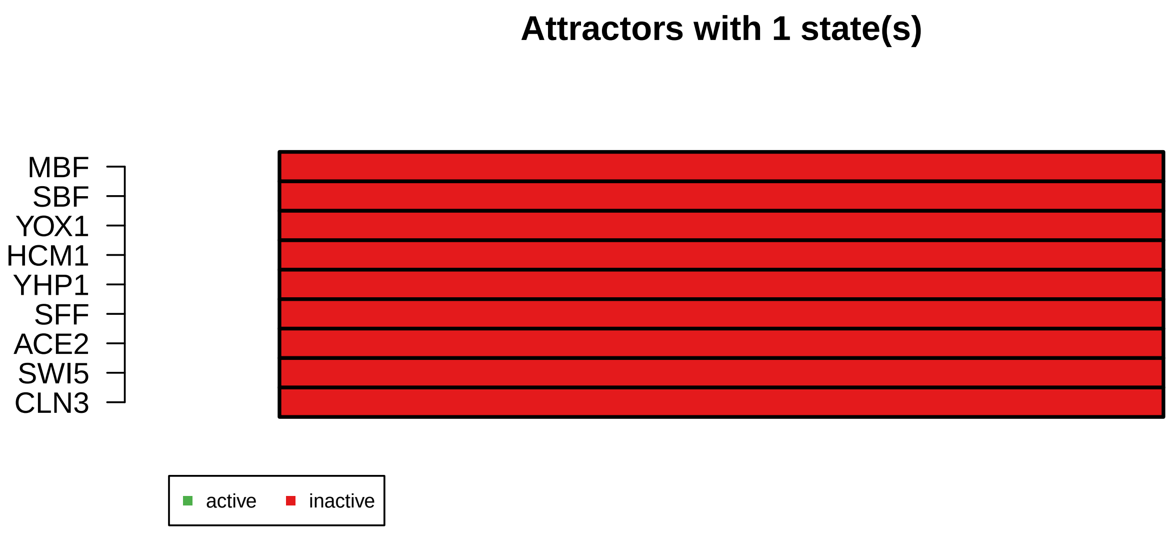
```
## Attractor 1 is a simple attractor consisting of 1 state(s) and has a basin of 101 state(s):
##
##  |--<--------|
##  V           |
##  000000000   |
##  V           |
##  |-->--------|
##
##
## Genes are encoded in the following order: MBF SBF YOX1 HCM1 YHP1 SFF ACE2 SWI5 CLN3
##
## Attractor 2 is a simple attractor consisting of 5 state(s) and has a basin of 411 state(s):
##
##  |--<--------|
##  V           |
##  110000000   |
##  011110000   |
##  000011000   |
##  000000110   |
##  000000001   |
##  V           |
##  |-->--------|
##
##
## Genes are encoded in the following order: MBF SBF YOX1 HCM1 YHP1 SFF ACE2 SWI5 CLN3
```

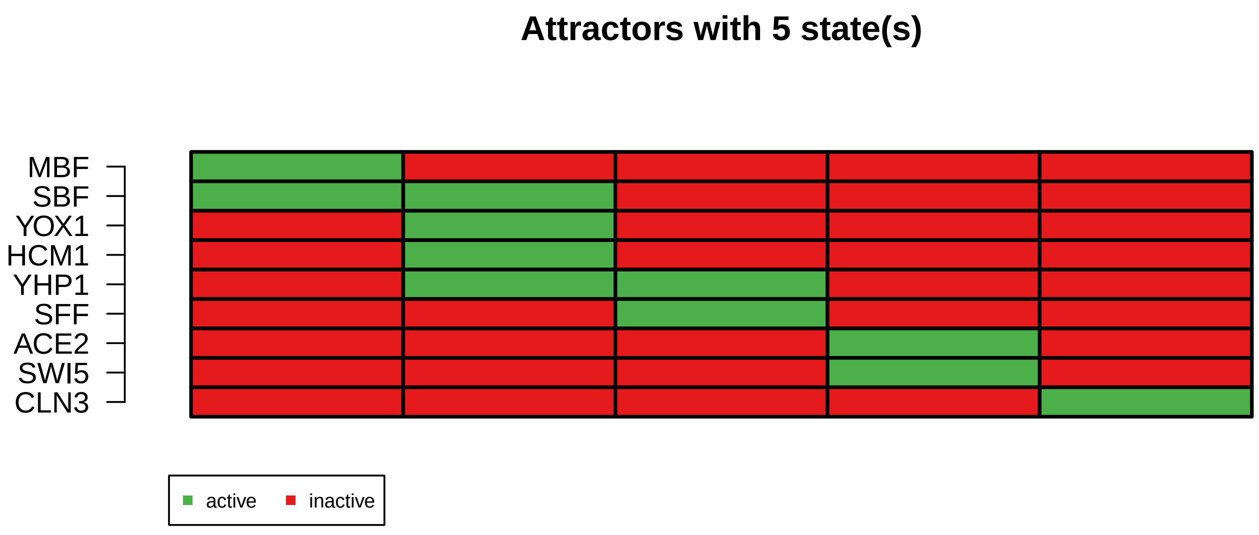
2 attractors are found: the first is steady state, the second is multi-state cycles. All states present in these attractors are the states resulted from markov chain sumulation.

```r
plotStateGraph(x)
```



```
par(mfrow=c(2, 1))
plotAttractors(x)
```





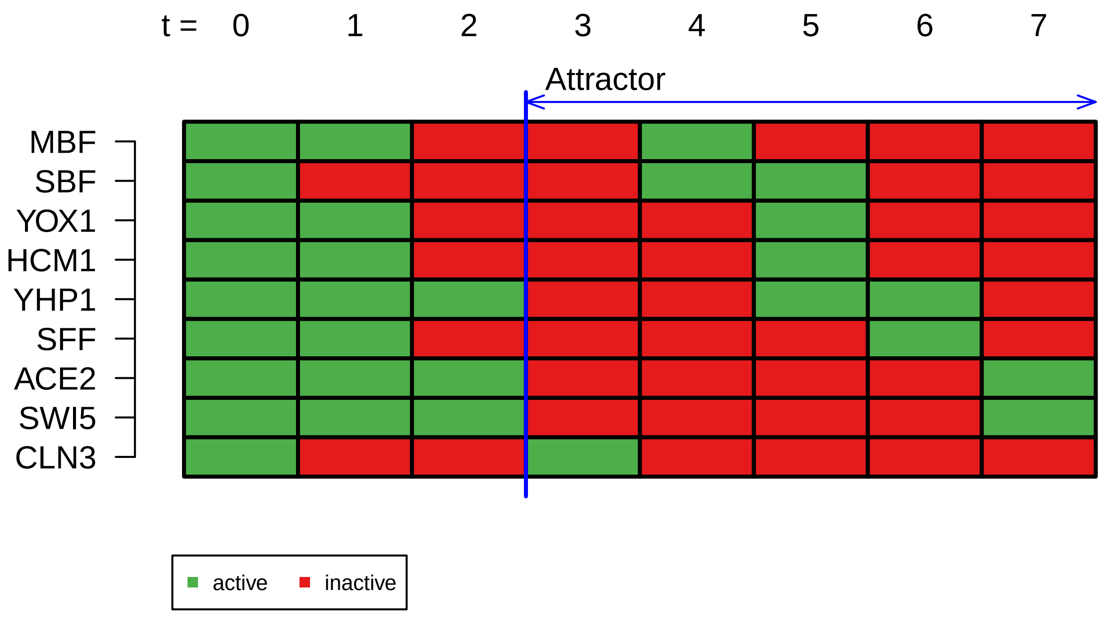
```
## $`1`
##      Attr1.1
## MBF        0
## SBF        0
## YOX1       0
## HCM1       0
## YHP1       0
## SFF        0
## ACE2       0
## SWI5       0
## CLN3       0
##
## $`5`
##      Attr2.1 Attr2.2 Attr2.3 Attr2.4 Attr2.5
## MBF        1       0       0       0       0
## SBF        1       1       0       0       0
## YOX1       0       1       0       0       0
## HCM1       0       1       0       0       0
## YHP1       0       1       1       0       0
## SFF        0       0       1       0       0
## ACE2       0       0       0       1       0
## SWI5       0       0       0       1       0
## CLN3       0       0       0       0       1
```

```
getAttractors(geneNet, type = 'synchronous', method = 'exhaustive',
              startStates = list(rep(1, 9)))
```

```
## Attractor 1 is a simple attractor consisting of 1 state(s) and has a basin of 101 state(s):
##
##  |--<--------|
##  V           |
##  000000000   |
##  V           |
##  |-->--------|
##
##
## Genes are encoded in the following order: MBF SBF YOX1 HCM1 YHP1 SFF ACE2 SWI5 CLN3
##
## Attractor 2 is a simple attractor consisting of 5 state(s) and has a basin of 411 state(s):
##
##  |--<--------|
##  V           |
##  110000000   |
##  011110000   |
##  000011000   |
##  000000110   |
##  000000001   |
##  V           |
##  |-->--------|
##
##
## Genes are encoded in the following order: MBF SBF YOX1 HCM1 YHP1 SFF ACE2 SWI5 CLN3
```

Both attractors are reached from the state with all genes are expressed.

```
plotSequence(geneNet, startState = rep(1, 9))
```



```
##      1 2 3 4 5 6 7 8
## CLN3 1 0 0 1 0 0 0 0
## SWI5 1 1 1 0 0 0 0 1
## ACE2 1 1 1 0 0 0 0 1
## SFF  1 1 0 0 0 0 1 0
## YHP1 1 1 1 0 0 1 1 0
## HCM1 1 1 0 0 0 1 0 0
## YOX1 1 1 0 0 0 1 0 0
## SBF  1 0 0 0 1 1 0 0
## MBF  1 1 0 0 1 0 0 0
```

```
getAttractors(geneNet, type = 'asynchronous')
```

```
## Attractor 1 is a simple attractor consisting of 1 state(s):
##
##  |--<--------|
##  V           |
##  000000000   |
##  V           |
##  |-->--------|
##
##
## Genes are encoded in the following order: MBF SBF YOX1 HCM1 YHP1 SFF ACE2 SWI5 CLN3
```

Only one attractor with all 0’s state is found when searching for asynchronous ones.

4. Gene knock-out and overexpression

```
geneNet2 <- fixGenes(geneNet, fixIndices = c('MBF', 'CLN3'), values = c(1, 0))
getAttractors(geneNet2, type = 'synchronous', method = 'exhaustive')
```

```
## Attractor 1 is a simple attractor consisting of 4 state(s) and has a basin of 128 state(s):
##
##  |--<--------|
##  V           |
##  110010000   |
##  111110000   |
##  101111000   |
##  100010110   |
##  V           |
##  |-->--------|
##
##
## Genes are encoded in the following order: MBF SBF YOX1 HCM1 YHP1 SFF ACE2 SWI5 CLN3
```

In this case we have only one attractor consisting of 4 states different from those we've seen previously.