European Doctoral School of Demography
Sequence Analysis

Liliana Patricia Calderón Bernal
Gonzalo Daniel Garcia
Ainhoa-Elena Leger

14/4/2021

Load necessary packages.

```
# Call required libraries
library(TraMineR)
library(ggplot2)
library(grDevices)
library(graphics)
library(foreign)
library(cluster)
library(Hmisc)
library(TraMineRextras)
library(WeightedCluster)
library(RColorBrewer)
library(colorspace)
library(tidyverse)
```

Exercise 1

1) Input the Dataset 2

[Sol.]

```
data2 <- read.csv("SFS2018_Data2.csv", na.strings=c(".",".a",".b"))
```

2) Define a sequence object with elements in data columns 2:61 and alphabet 1:6, using the following state names and labels

   1 SNP "Single, childless",
   2 SBP "Single, child b/separat.",
   3 SAP "Single, child a/separat.",
   4 UNP "Union, childless",
   5 UBP "Union, child b/separat.",
   6 UAP "Union, child a/separat."

[Sol.]

```
# Create a vector for the state labels
seqlab <-c("Single, childless", "Single, child b/separat.",
           "Single, child a/separat.", "Union, childless",
           "Union, child b/separat.", "Union, child a/separat.")
# Create a vector of short state names (default would be alphabet labels)
```

```r
sllist <- c("SNP","SBP","SAP","UNP", "UBP", "UAP")
# Define Color palette
color1 <-  sequential_hcl(6, palette = "SunsetDark", rev= TRUE)
# Generate sequence object
seqObj2 <- seqdef(data2,
                  var=2:61,
                  alphabet=c(1:6),
                  cpal=color1,
                  states=sllist,
                  labels=seqlab)
```

3) Display (print) the first 10 sequences in extended and compact form

[Sol.]

Extended form:

```r
# Display the first 10 sequences (STS format - default)
print(seqObj2[1:10, ], format ="STS")
```

Sequence
1 SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-
SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-
SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP
2 SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-
SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-
SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP
3 SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-
UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-
UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-UNP-
UNP-UNP-UNP-UNP-UNP 4 SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-
SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-
SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-
SNP-SNP-SNP-SNP-SNP-SNP 5 SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-
SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-
SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-
SBP-SBP-SBP-SBP-SBP-SBP 6 SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-
SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-
SBP-SBP-SBP
7 SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-
SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-
SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP
8 SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-
SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-
SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP 9
SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-
SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-
SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP-SBP
10 SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-
SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-
SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP-SNP

Compact form:

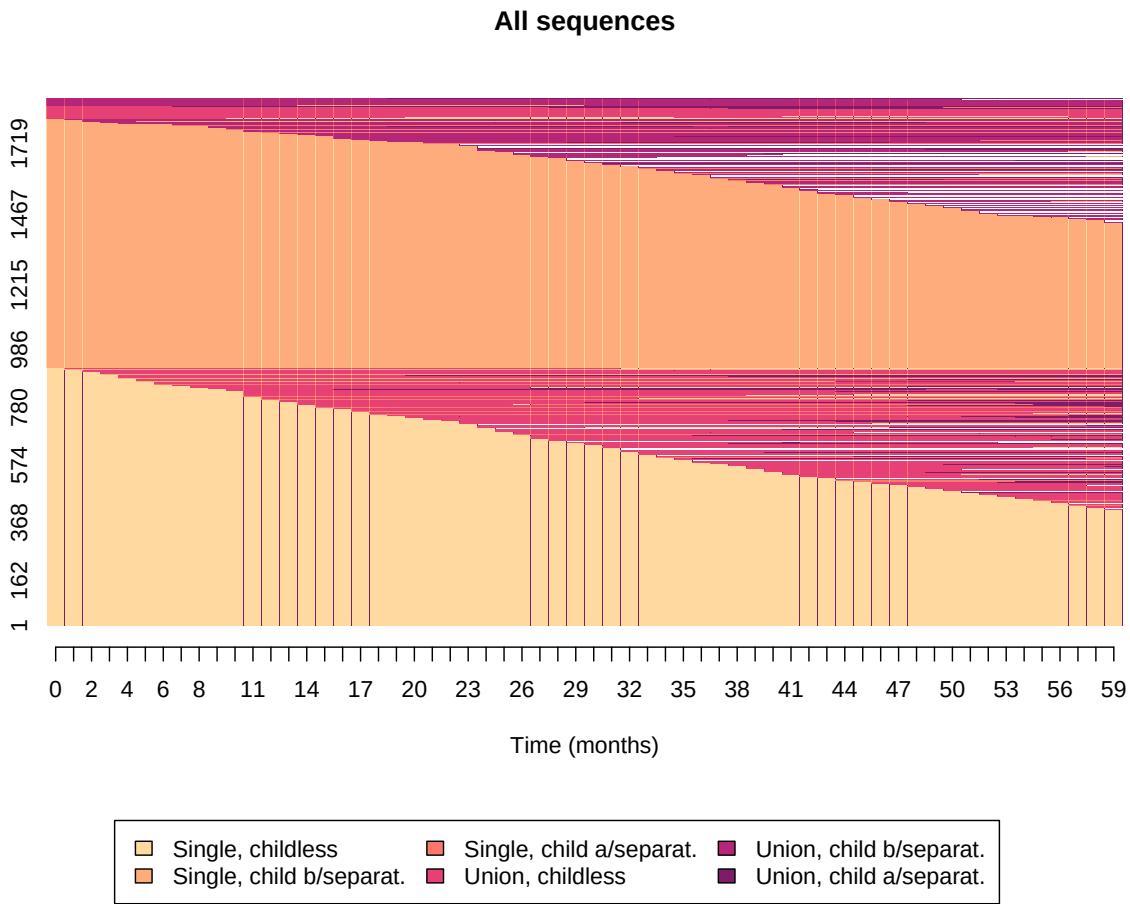
```r
# Display the first 10 sequences (SPS format)
print(seqObj2[1:10, ], format ="SPS")
```

Sequence
1 (SBP,48)
2 (SNP,60)
3 (SNP,11)-(UNP,49) 4 (SNP,60)
5 (SBP,60)
6 (SBP,37)
7 (SBP,59)
8 (SBP,60)
9 (SBP,52)
10 (SNP,60)

4) Plot a full representation of sequences, and order them from the first state

[Sol.]

```
# X-axis for exercise
xtlab=seq(0,60, by=1)
# All sequences -sequence index plot (sorted - first state)
seqIplot(seqObj2, with.legend=TRUE, main= "All sequences",
         xtlab=xtlab, xlab="Time (months)", ylab=NA, yaxis=TRUE,
         border=NA, sortv="from.start")
```



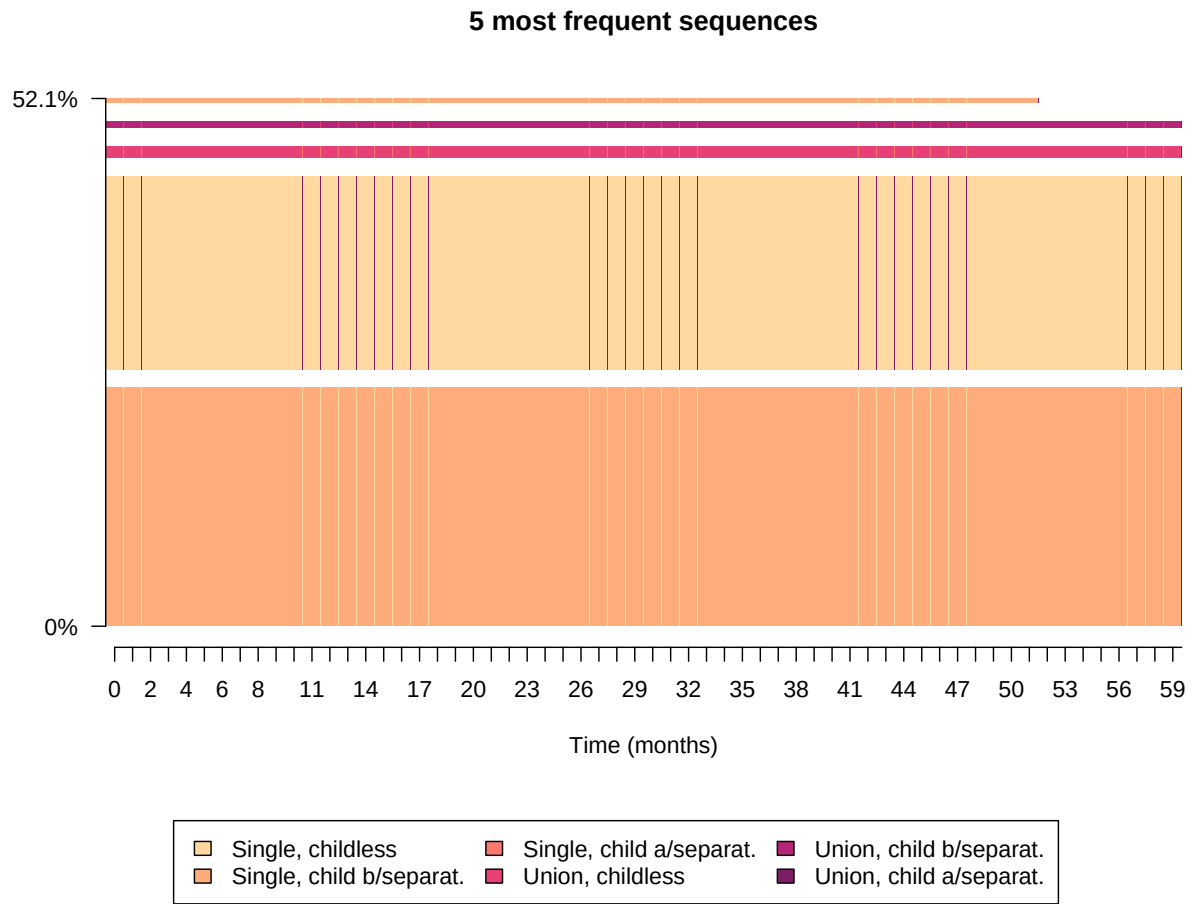
At a first glance, after separation, the majority of the individuals are single, with or without children (lighter colours). With time passing, some individuals experience unions (darker colours) and at the end of the period

about half of the individuals are still single.

5) Plot the 5 most frequent sequences. Comment the plot.

[Sol.]

```
seqfplot(seqObj2, idxs=1:5, main="5 most frequent sequences",
         with.legend=TRUE, border=NA,
         ylab=NA, xlab="Time (months)", xtlab=xtlab)
```



The same information can be obtained as a frequency table with absolute and relative frequencies.

```
seqtab(seqObj2, idxs=1:5)
```

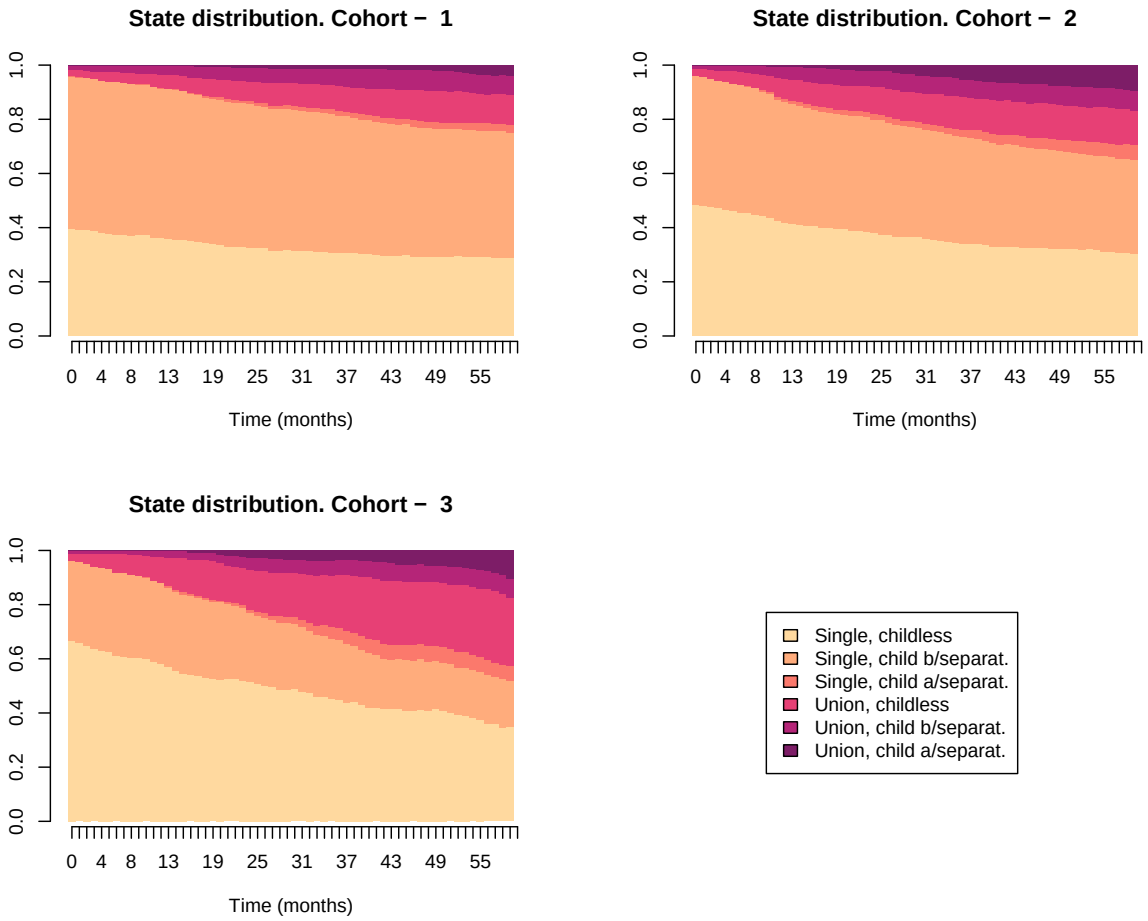
```
       Freq Percent
SBP/60  514   27.34
SNP/60  416   22.13
UNP/60   25    1.33
UBP/60   15    0.80
SBP/52   10    0.53
```

The five most frequent sequences after the separation are "Single, child b/separat." for the whole period (27%), "Single, childless" for the whole period (22%), "Union, childless" for the whole period (1%), "Union, child b/separat." for the whole period (<1%), and "Single, child b/separat." for 52 months (<1%). Overall, the five most frequent sequences account for 52% of all the sequences.

6) Create a state distribution plot for each birth cohort (BIRTHCOH). What are the cross-cohort differences in the distribution of states overtime?

[Sol.]

```
seqdplot(seqObj2, group=data2$BIRTHCOH, with.legend=TRUE,
         main= "State distribution. Cohort", use.layout=FALSE,
         border=NA, xtlab=xtlab, ylab=NA, xlab="Time (months)")
```



From the plot we can observe that Cohort 1 has the bigger proportion of single who are childless or had a child before separation, with the proportion of single who are childless being lower than that of single who had a child before separation. Both proportions tends to remain constant/slightly decreasing over time.

In Cohort 3, single childless' state represents more than half of the whole individuals just after separation, and the second more frequent state is being single with a child before separation. Both proportions decrease consistently over time, in favour of the other states (in particular of being in a union and childless).

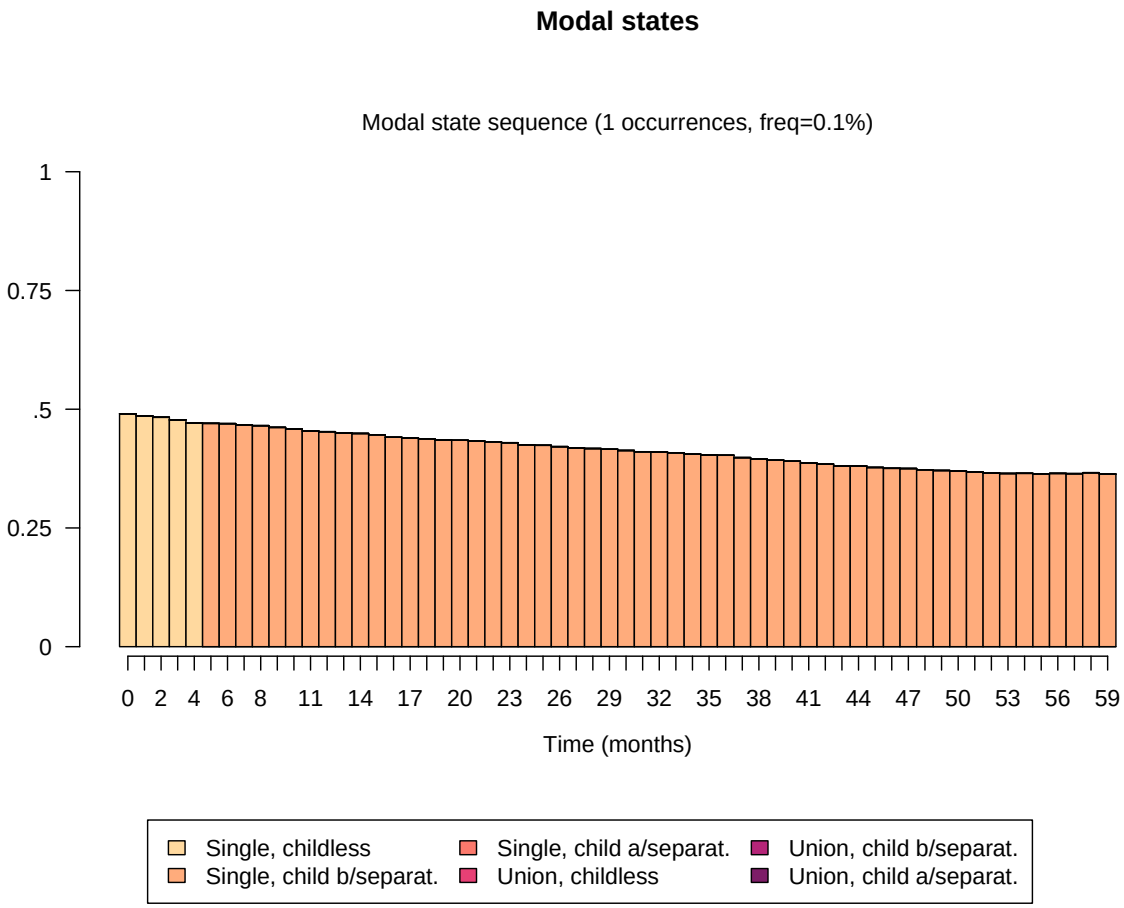
Cohort 2 can be seen as an intermediate situation.

Cohort 1 is the further one in time ordering (1960/69), followed by cohort 2 (1970/79) and cohort 3 (1980/89). Therefore, we can observe that more recent cohorts tends to re-enter in an union after separation, and individuals seems more prone of having children even after a couple dissolution.

7) What are the most frequent states one and five years after break-up? Use a modal state plot for illustration.

[Sol.]

```
par(mfrow=c(1,1))
seqmsplot(seqObj2, with.legend=TRUE, main="Modal states",
          xtlab=xtlab, ylab=NA, xlab="Time (months)")
```



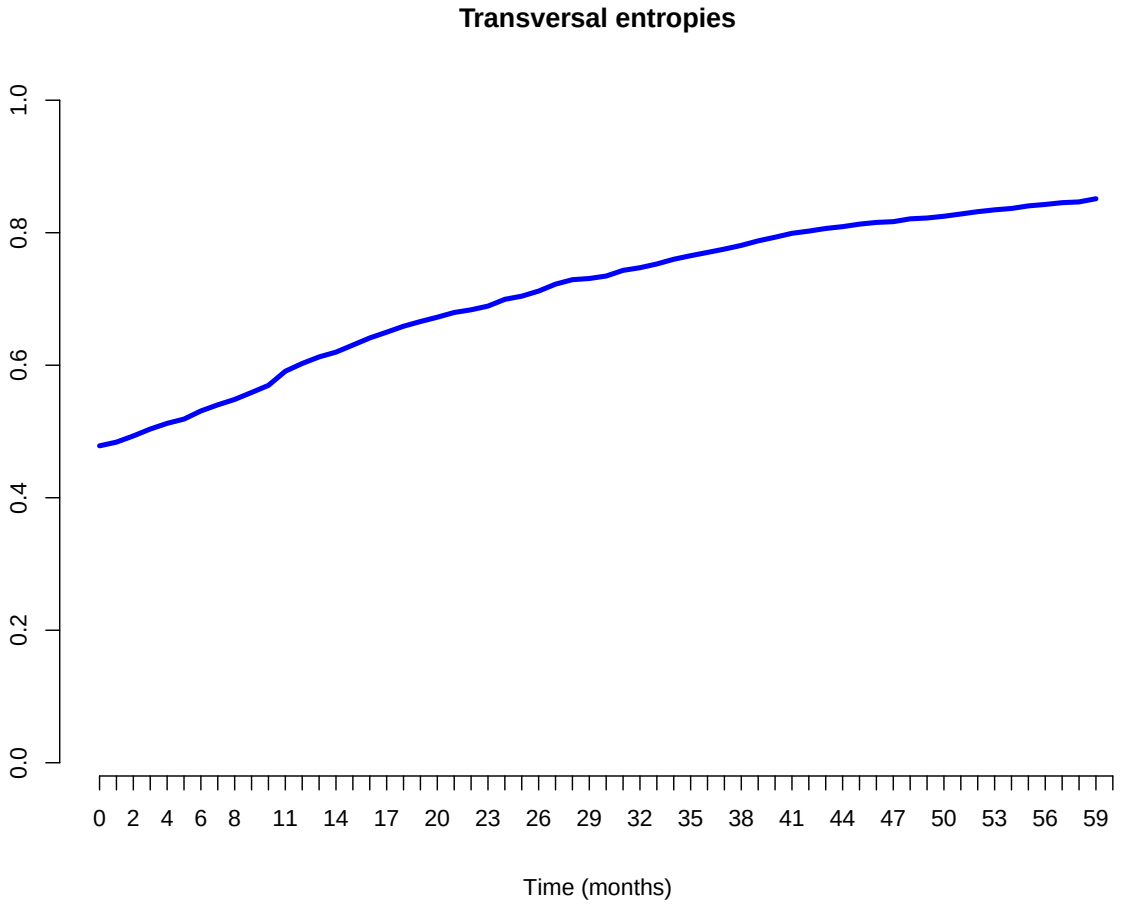
As it can be seen, "Single, childless" is the most frequent state in the first 5 months. After 5 months, "Single, child b/separat." is the most frequent state. Therefore, one and five years after break-up the most frequent state is still "Single, child b/separat.".

This behaviour would have been observable from a distribution plot like the one of the previous question, if considering all the cohorts combined.

8) Assess the cross-sectional state diversity plotting a measure of entropy. At what time after separation is the cross-sectional diversity of the states at its highest?

[Sol.]

```
# Plot the transversal entropies in each position of the sequence
seqHtplot(seqObj2, with.legend=FALSE, main= "Transversal entropies",
          use.layout=FALSE, border=NA,xtlab=xtlab, ylab=NA, xlab="Time (months)")
```



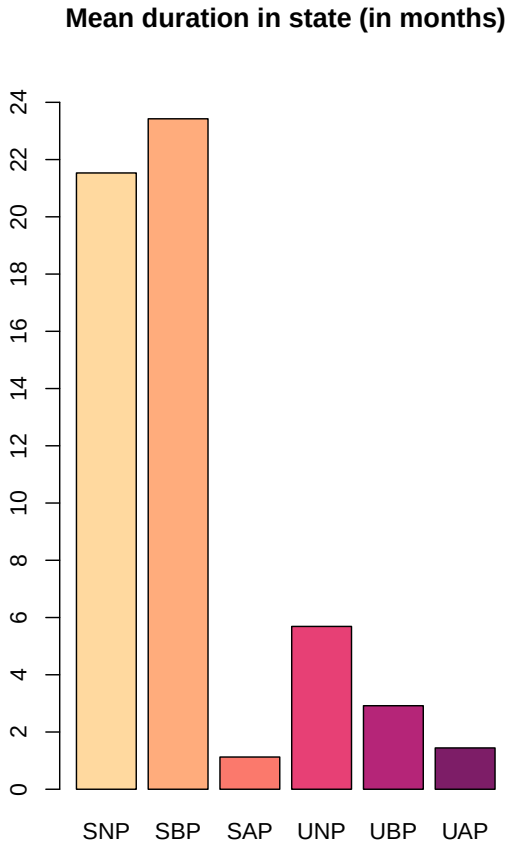
The plot shows that the entropy measure keeps increasing with time, reaching its maximum at month 60 (the end of the observation time).

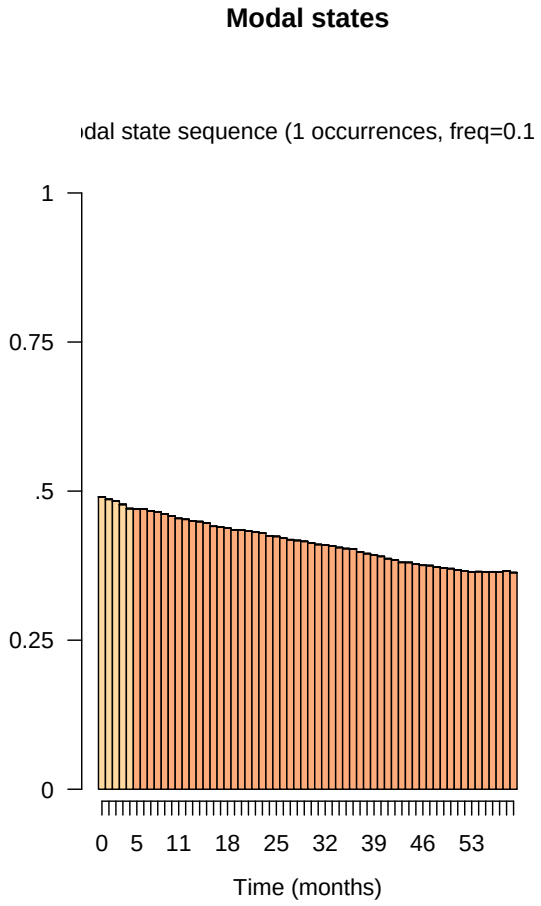
This could be a result of having different ages in the individual sequence: even though we are analyzing the data from a life-course perspective, that 60-months could represent many different ages (and stages) in a woman's life, which could create a bias in the plot.

9) Display side by side in a same plot area the mean times spent in each of the states and the sequence of modal states.

[Sol.]

```
par(mfrow = c(1, 2))
# Plot the mean time spent in eache state
seqmtplot(seqObj2, with.legend=FALSE,
          main= "Mean duration in state (in months)",
          ylab=NA, ylim=c(0,25), yaxis=F)
axis(2, at=seq(from=0, to=25, by=2))
# Plot modal states in each position of the sequence
seqmsplot(seqObj2, with.legend=FALSE, main="Modal states", xtlab=xtlab,
          ylab=NA, xlab="Time (months)")
```





The exact information on the mean times can be obtained in a table.

```
mean.month.seq <- seqistatd(seqObj2)
apply(mean.month.seq,2,mean)
```

```
      SNP       SBP       SAP       UNP       UBP       UAP
21.532447 23.424468  1.125532  5.688830  2.918085  1.443617
```

The mean time of individuals in state "Single, child b/separat." is 23.4 months, in "Single, childless" it is 21.5 months. In all the other states, individuals stay on average less than 6 months.

```
modal.month.seq <- seqdef(as_tibble(seqmodst(seqObj2)))
print(modal.month.seq, format = "SPS")
```

```
  Sequence
1 (SNP,5)-(SBP,55)
```

The most frequent state is "Single, childless" for the first 5 months and "Single, child b/separat." for the next 55 months.

10) Compute the (overall) transition rate matrix. What is the largest transition rate between two different states?

[Sol.]

```
seqtrate(seqObj2)
```

```
            [-> SNP]     [-> SBP]     [-> SAP]    [-> UNP]    [-> UBP]
[SNP ->] 0.989975189 0.0003508684 0.0011779153 0.008445904 0.000000000
[SBP ->] 0.000000000 0.9959117681 0.0004388498 0.000000000 0.003603187
[SAP ->] 0.000000000 0.0000000000 0.9956033219 0.000000000 0.000000000
[UNP ->] 0.005545993 0.0000000000 0.0000000000 0.985848155 0.000000000
[UBP ->] 0.000000000 0.0041083100 0.0001867414 0.000000000 0.990476190
[UAP ->] 0.000000000 0.0000000000 0.0042405551 0.000000000 0.000000000
             [-> UAP]
[SNP ->] 5.012406e-05
[SBP ->] 4.619471e-05
[SAP ->] 4.396678e-03
[UNP ->] 8.605852e-03
[UBP ->] 5.228758e-03
[UAP ->] 9.957594e-01
```

The largest transition rate between two different states is the one that goes from SNP to UNP: from Single with no children to Union with no children. The second largest is the one that reverses that states from UNP to SNP. This means that going in and out from Unions without children is more probable than moving to other states, which relates to the fact that states where children are present are less frequent through cohorts.

11) Compute the sequence length, the number of transitions, the number of sub-sequences and the longitudinal entropy

[Sol.]

```
# Sequence length - number of elements with valid cases
length <-seqlength(seqObj2)
# Number of transitions between state episodes in each sequence
transn <-seqtransn(seqObj2)
# Number of sub-sequences contained in a sequence
subseq <- seqsubsn(seqObj2)
# Longitudinal or within-sequence entropy
entropy <- seqient(seqObj2)
```

12) Using summary(), look at the min, max, mean, median and quartiles of the distribution of each of the computed longitudinal characteristics.

[Sol.]

Summary of Sequence length:

```
summary(length)
```

```
     Length
 Min.   :23.00
 1st Qu.:60.00
 Median :60.00
 Mean   :56.13
 3rd Qu.:60.00
 Max.   :60.00
```

Summary of Number of transitions between state episodes:

```
summary(transn)
```

```
     Trans.
 Min.   :0.0000
 1st Qu.:0.0000
 Median :0.0000
```

```
 Mean   :0.4234
 3rd Qu.:1.0000
 Max.   :5.0000
```

Summary of Number of sub-sequences contained in a sequence:

```
summary(subseq)
```

```
    Subseq.
 Min.   : 2.000
 1st Qu.: 2.000
 Median : 2.000
 Mean   : 3.093
 3rd Qu.: 4.000
 Max.   :48.000
```

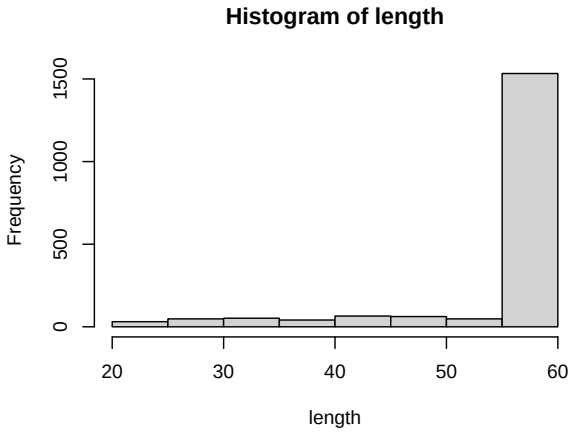
Summary of Entropy:

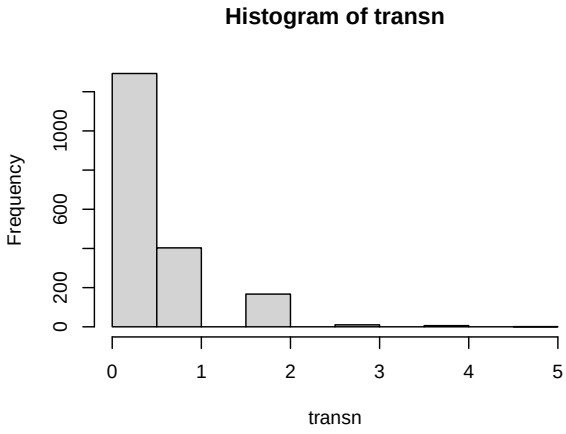
```
summary(entropy)
```

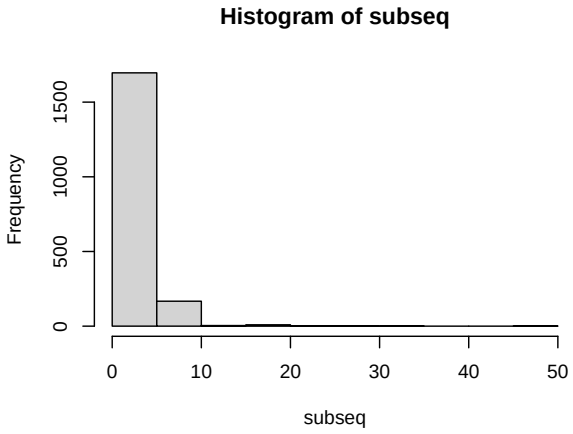
```
    Entropy
 Min.   :0.0000
 1st Qu.:0.0000
 Median :0.0000
 Mean   :0.1045
 3rd Qu.:0.2359
 Max.   :0.7323
```

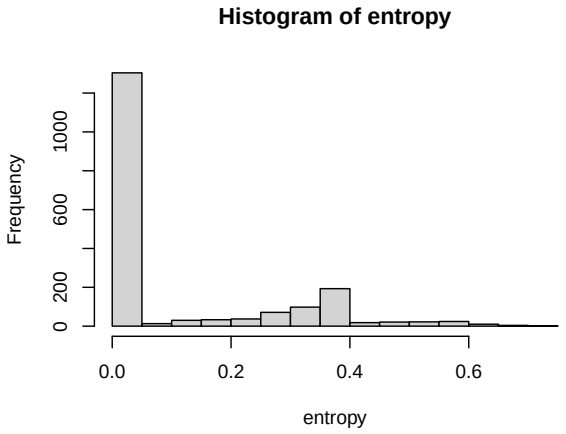
By looking at the histogram of these quantities, we can identify that most of the sequences are concentrated on higher length, low number of transitions and sub-sequences, and low entropies.

```
par(mfrow=c(2,2))
hist(length)
hist(transn)
hist(subseq)
hist(entropy)
```


Histogram of length
Frequency
0
500
1000
1500
20
30
40
50
60
length


Histogram of transn
Frequency
0
200
600
1000
0
1
2
3
4
5
transn


Histogram of subseq
Frequency
0
500
1000
1500
0
10
20
30
40
50
subseq


Histogram of entropy
Frequency
0
200
600
1000
0.0
0.2
0.4
0.6
entropy

Exercise 2

1) Input the Dataset 2

[Sol.]

```
data2 <- read.csv("SFS2018_Data2.csv", na.strings=c(".",".a",".b"))
```

2) Define a sequence object with elements in data columns 2:61 and alphabet 1:6, using the following state names and labels

   1 SNP "Single, childless",
   2 SBP "Single, child b/separat.",
   3 SAP "Single, child a/separat.",
   4 UNP "Union, childless",
   5 UBP "Union, child b/separat.",
   6 UAP "Union, child a/separat."

[Sol.]

```
# Create a vector for the state labels
seqlab <-c("Single, childless",
           "Single, child b/separat.",
           "Single, child a/separat.",
           "Union, childless",
           "Union, child b/separat.",
           "Union, child a/separat.")
# Create a vector of short state names (default would be alphabet labels)
sllist <- c("SNP","SBP","SAP","UNP", "UBP", "UAP")
# Generate sequence object
seqObj2 <- seqdef(data2,
                  var=2:61,
                  alphabet=c(1:6),
                  cpal=color1,
                  states=sllist,
                  labels=seqlab)
```

3) Compute the matrix of pairwise distances - OM with constant costs - between all sequences and display the results for the first 5 sequences.

[Sol.]

We assign the substitution cost to 2 and the indel cost to 1.

```
# OM with CONSTANT substitution costs (OM with indel=1, subs=2)
Matrix.OM.Const <- seqdist(seqObj2, method="OM", indel=1, sm="CONSTANT")
# Display matrix
print(Matrix.OM.Const[1:5,1:5])
```

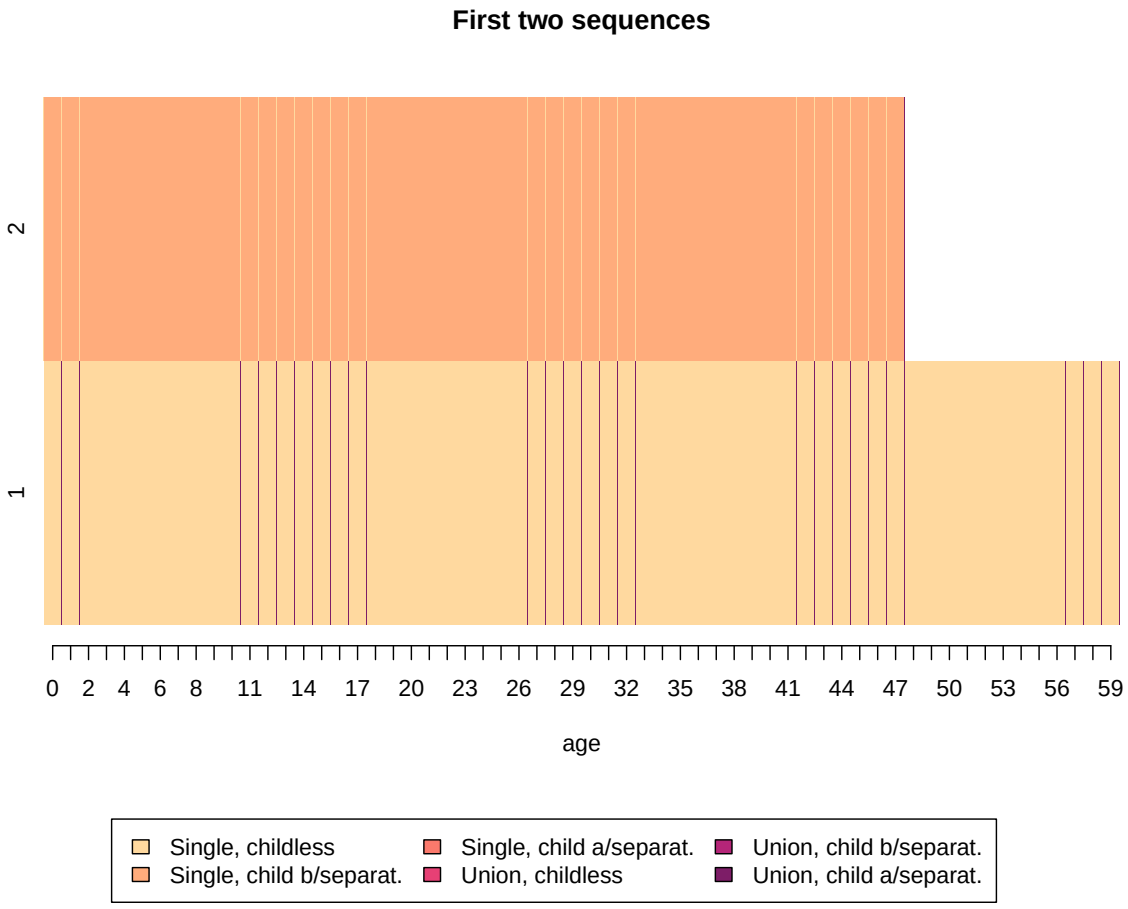
```
     [,1] [,2] [,3] [,4] [,5]
[1,]    0  108  108  108   12
[2,]  108    0   98    0  120
[3,]  108   98    0   98  120
[4,]  108    0   98    0  120
[5,]   12  120  120  120    0
```

The diagonal is 0 because the cost of keeping the sequence as it is 0. The elements outside the diagonal represent the cost to transform a sequence into another.

4) Plot the first 2 sequences and check that the OM distance is the number of non matching positions between them.

[Sol.]

```
# Plot the first 2 sequences
xtlab=seq(0,60, by=1)
seqIplot(seqObj2[1:2, ], with.legend=TRUE, main= "First two sequences",
         xtlab=xtlab, xlab="age", ylab=NA, yaxis=TRUE, sortv="from.start")
```




```
# Display the first 2 sequences (SPS format)
print(seqObj2[1:2, ], format ="SPS")
```

```
  Sequence
1 (SBP,48)
2 (SNP,60)
```

The OM distance between two sequences reflects the cost to transform one sequence into another. The first two sequences have different states for the first 48 months, then the first sequence stop and the second one continues until the end of the period (60th month). As we assigned the substitution cost to 2 and the indel cost to 1, the cost of transforming the first sequence to the second is 2 for the first 48 months and 1 for the last 12 months. The final cost is $48 * 2 + 12 = 108$, as it can also be seen from the previous matrix (elements [1,2] and [2,1] are 108).

5) Check data that the LCS distance provides the same (non-normalized) distances as OM with constant costs.

[Sol.]

```
# Longest common sub-sequence
Matrix.LCS <- seqdist(seqObj2, method="LCS")
# Display matrix
print(Matrix.LCS[1:5,1:5])
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]    0  108  108  108   12
[2,]  108    0   98    0  120
[3,]  108   98    0   98  120
[4,]  108    0   98    0  120
[5,]   12  120  120  120    0
```

```
# Compare
print(Matrix.OM.Const[1:5,1:5])
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]    0  108  108  108   12
[2,]  108    0   98    0  120
[3,]  108   98    0   98  120
[4,]  108    0   98    0  120
[5,]   12  120  120  120    0
```

6) Define a substitution cost matrix reflecting what (according to your prior knowledge) are the distances between two states (i.e. customize state-dependent substitution costs).

[Sol.]

We start by calculating the transition matrix, rounded to the 3rd decimal:

```
round(seqtrate(seqObj2),3)
```

```
         [-> SNP] [-> SBP] [-> SAP] [-> UNP] [-> UBP] [-> UAP]
[SNP ->]    0.990    0.000    0.001    0.008    0.000    0.000
[SBP ->]    0.000    0.996    0.000    0.000    0.004    0.000
[SAP ->]    0.000    0.000    0.996    0.000    0.000    0.004
[UNP ->]    0.006    0.000    0.000    0.986    0.000    0.009
[UBP ->]    0.000    0.004    0.000    0.000    0.990    0.005
[UAP ->]    0.000    0.000    0.004    0.000    0.000    0.996
```

We will try to reflect the ordinal character of this matrix by:

- setting the cost of remaining in the same state as 0.
- the highest expected cost will be 10 (since dividing 1 by the transition probability would give us an `Inf`).
- the higher the probability, the lower the cost.
- for all other costs, we subtract the 3rd decimal of the transition probability to the maximum cost.

We get the following state-dependent subcost matrix:

```
# OM with customized state-dependent subcosts
submatrix <- matrix(c( 0,10,9,2,10,10,
                       10,0,10,10,6,10,
                       10,10,0,10,10,6,
                       4,10,10,0,10,1,
```

```
                        10,6,10,10,0,5,
                        10,10,6,10,10,0), nrow = 6, ncol = 6, byrow = TRUE)
```

7) Compute the OM dissimilarity matrix using the previously derived substitution. Set the indel cost as half the maximum substitution cost.

[Sol.]

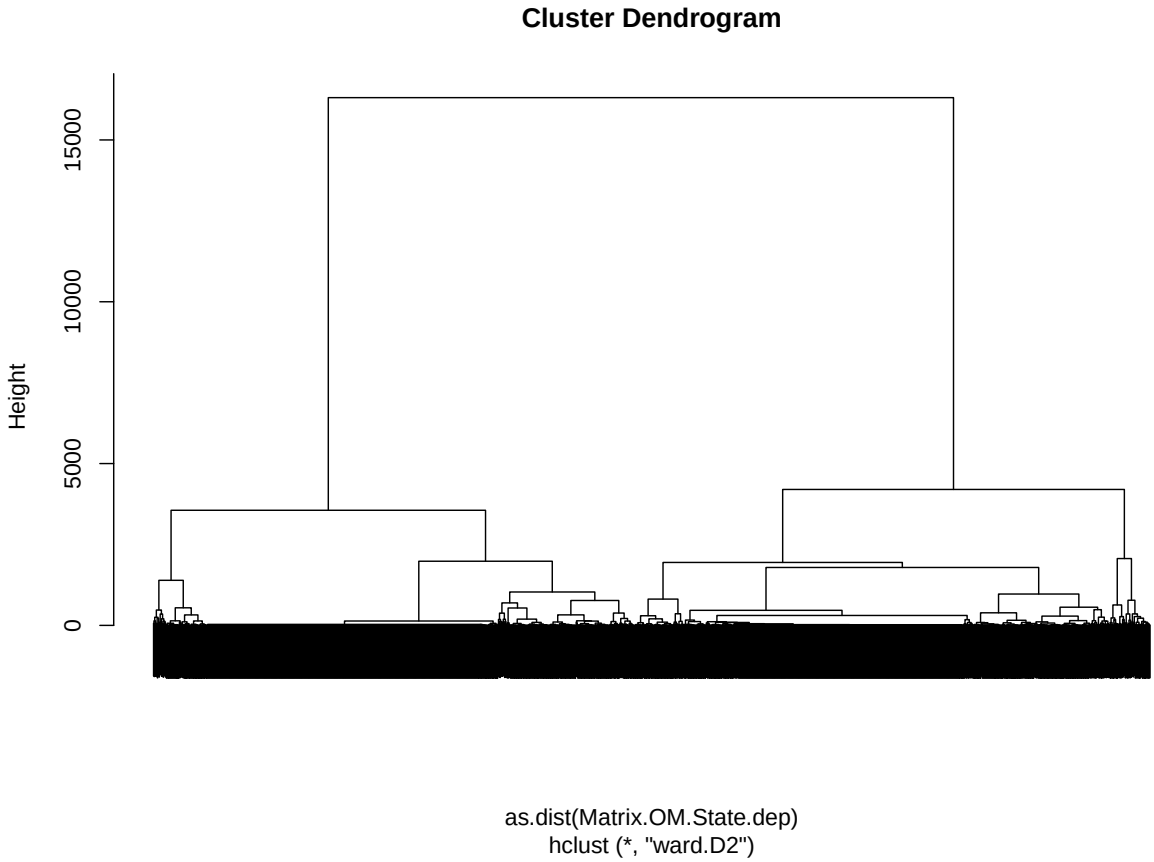
```
# OM dissimilarity matrix
Matrix.OM.State.dep <- seqdist(seqObj2, method="OM", indel=5, sm=submatrix)
# Display matrix
print(Matrix.OM.State.dep[1:5,1:5])
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]    0  540  540  540   60
[2,]  540    0   98    0  600
[3,]  540   98    0   98  600
[4,]  540    0   98    0  600
[5,]   60  600  600  600    0
```

8) From the previously computed OM dissimilarity matrix, create a hierarchical cluster tree object with Ward method. Display the hierarchical tree.

[Sol.]

```
# Cluster sequences using the OM distances with state-dependent costs and Ward method
ward.OM <- hclust(as.dist(Matrix.OM.State.dep), method = "ward.D2")
# Dendogram
plot(ward.OM, labels=FALSE)
```



9) Calculate appropriate cluster cut-off criteria. Assess what is an empirically optimal cluster solution.

[Sol.]

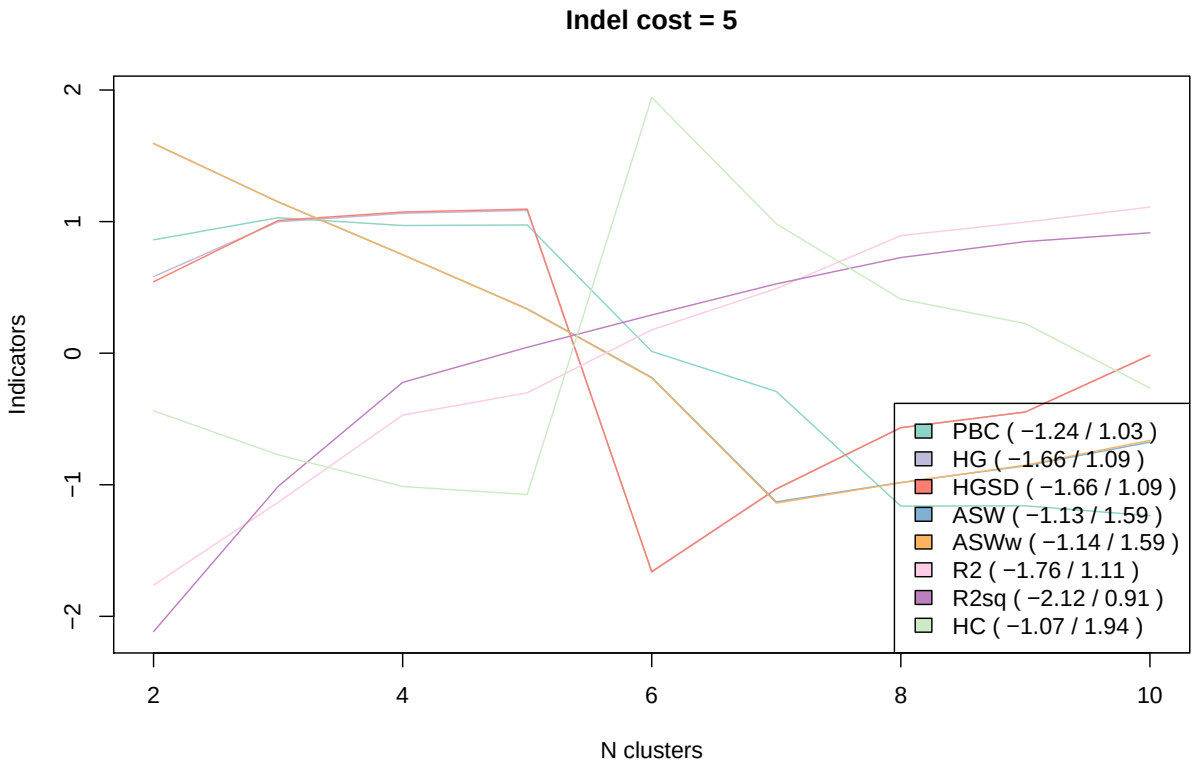
We computed several clustering quality measures for a range of numbers of groups (2 until 10 clusters) and retrieved the three best solutions according to each quality measure.

```
# Generate an object with 1-10 cluster solutions
wardrange.OM <- as.clustrange(ward.OM, diss=Matrix.OM.State.dep, ncluster=10)
# Show cluster cut-off measure values - indicate three optimal cluster solutions
summary(wardrange.OM, max.rank=3)
```

```
     1. N groups     1.  stat 2. N groups     2.  stat 3. N groups     3.  stat
PBC            3 9.243997e-01           5 9.179705e-01           4 9.175173e-01
HG             5 9.912195e-01           4 9.907705e-01           3 9.894552e-01
HGSD           5 9.911783e-01           4 9.907285e-01           3 9.894158e-01
ASW            2 7.822016e-01           3 7.575884e-01           4 7.353381e-01
ASWw           2 7.824296e-01           3 7.581802e-01           4 7.362024e-01
CH             2 3.512789e+03           3 2.224554e+03           4 1.955590e+03
R2            10 8.875757e-01           9 8.782356e-01           8 8.697589e-01
CHsq           2 8.773176e+03          10 7.727431e+03           9 7.699131e+03
R2sq          10 9.738157e-01           9 9.705187e-01           8 9.645109e-01
HC             5 6.903321e-03           4 7.399632e-03           3 9.362936e-03
```

According to most of the cluster quality criteria, 2 or 5 cluster solutions are first best empirical fits. The same information can be plotted and used to identify the best number of groups.

```
plot(wardrange.OM, norm="zscore", main="Indel cost = 5")
```

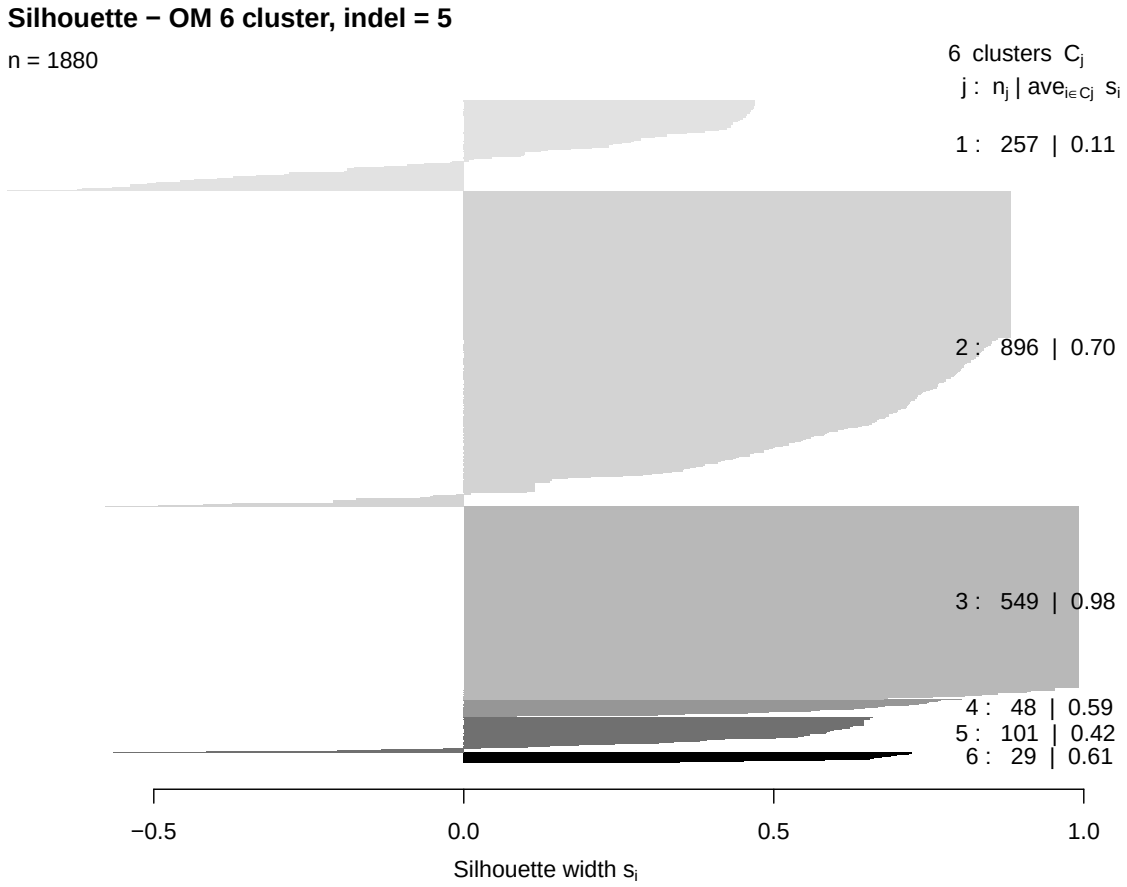


10) Select the six-cluster solution from the Ward analysis, check cluster consistency, and label the clusters by looking at the full sequence index plots (or the relative frequency version) by cluster.

[Sol.]

Let's first check cluster consistency.

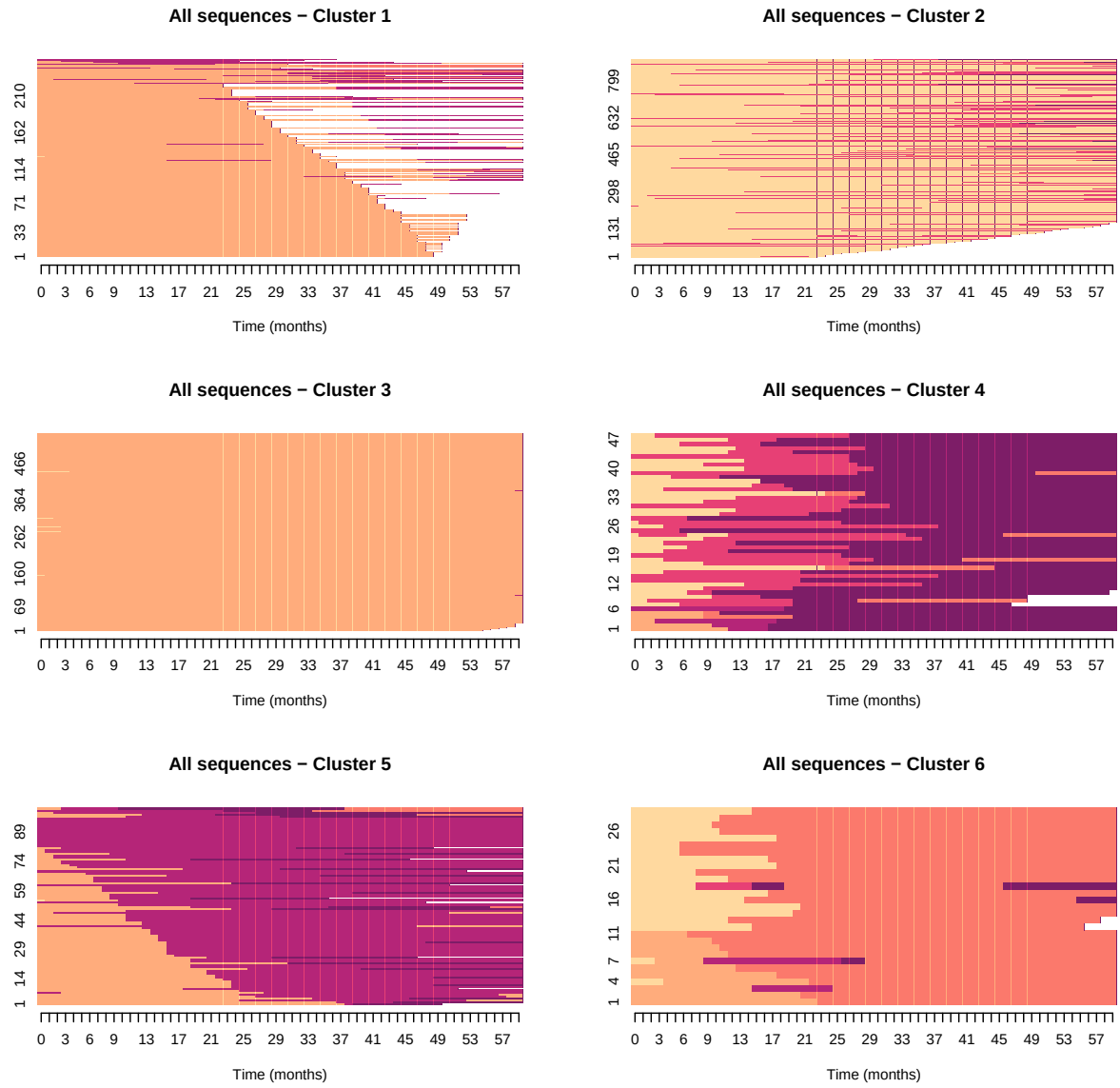
```
# Store 6 cluster solutions
wardrange.OM.6 <- cutree(ward.OM , k=6)
# Cluster consistency (plot silhouette widths)
silh.OM.6 <- silhouette(wardrange.OM.6, dmatrix = Matrix.OM.State.dep)
#summary(silh.OM.6)
plot(silh.OM.6, main= "Silhouette - OM 6 cluster, indel = 5", border=NA,
     col=c("#E2E2E2", "#D3D3D3", "#B8B8B8", "#969696", "#707070", "#000000"))
```



According to the silhouettes, the individuals are well matched to their own cluster. Only in the first cluster the individuals seems to be not coherently classified. The averaged silhouette width indicates an overall coherence of the clusters (ASW=0.68).

We then labelled the clusters by looking at the full sequence index plot by cluster. To avoid over-plotting, we sorted the sequences according to their distance to the most representative sequence, defined by the neighbourhood density.

```
# Generate a variable that contains the OM distance of each sequence to repseq1
repseq1 <- seqrep(seqObj2, diss=Matrix.OM.State.dep, criterion="density", nrep=1)
OMdistRepseq <- seqdist(seqObj2, refseq = repseq1, method = "OM",
                        indel=5, sm=submatrix)
# Define the cluster variable
cl.6fac <- factor(wardrange.OM.6, labels = paste("Cluster", 1:6))
# Plot the full sequence index plot
seqIplot(seqObj2, group=cl.6fac, with.legend=TRUE,
         main= "All sequences", xtlab=xtlab, xlab="Time (months)",
         ylab=NA, yaxis=TRUE, border=NA, sortv=OMdistRepseq)
```



We can identify the following patterns:

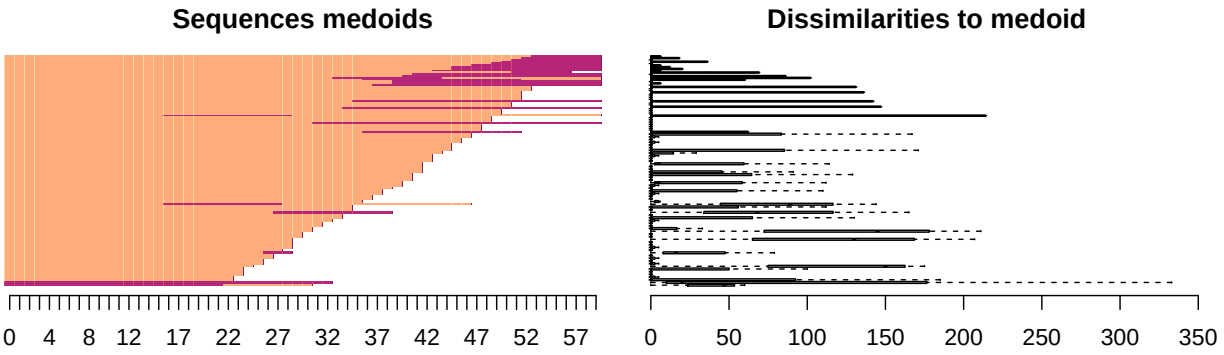
- Cluster 1 - Single parents (shorter sequences): individuals who had a child b/ separation, remain single for 2-3 years after separation and then are not followed any more.
- Cluster 2 - Childless single entering in a union: individuals who are single and childless entering in an union and remaining childless.
- Cluster 3 - Parents ever single: individuals who had a child before separation and remain single over the whole period.
- Cluster 4: Childless single entering in a union and having a child after separation.

- Cluster 5 - Single parents entering a union: single individuals who had a child b/ separation and entered a union before two years after separation.
- Cluster 6 - Single becoming parents: childless single or single who already had a child before separation having a child approximately one year after separation.

We can also look at the relative frequency plot.

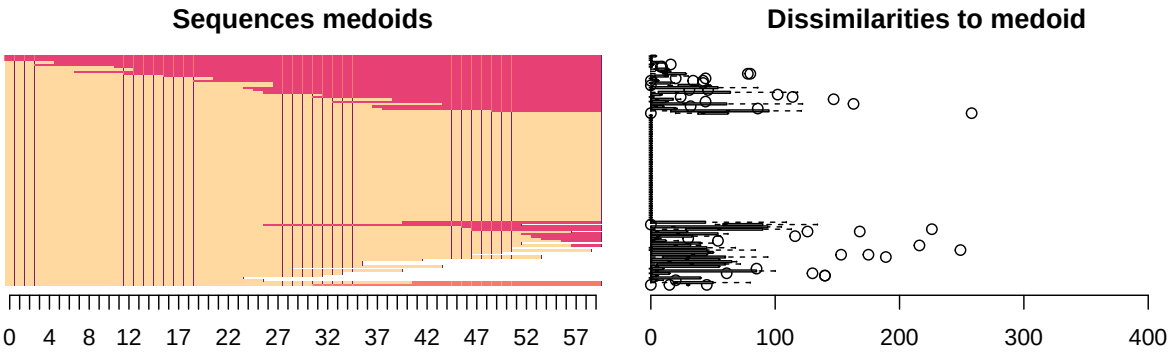
```
# Compute cluster specific matrix distances (OM 6-cluster solution)
Matrix.OM.1 <- seqdist(seqObj2[wardrange.OM.6 == 1,], method="OM", sm=submatrix)
Matrix.OM.2 <- seqdist(seqObj2[wardrange.OM.6 == 2,], method="OM", sm=submatrix)
Matrix.OM.3 <- seqdist(seqObj2[wardrange.OM.6 == 3,], method="OM", sm=submatrix)
Matrix.OM.4 <- seqdist(seqObj2[wardrange.OM.6 == 4,], method="OM", sm=submatrix)
Matrix.OM.5 <- seqdist(seqObj2[wardrange.OM.6 == 5,], method="OM", sm=submatrix)
Matrix.OM.6 <- seqdist(seqObj2[wardrange.OM.6 == 6,], method="OM", sm=submatrix)

# Compute the cluster specific relative frequency sequence index plots (OM 6-cluster solution)
par(mfrow=c(5,5))
seqplot.rf(seqObj2[wardrange.OM.6 == 1,], diss=Matrix.OM.1, k=100, xtlab=xtlab, yaxis=FALSE)
```



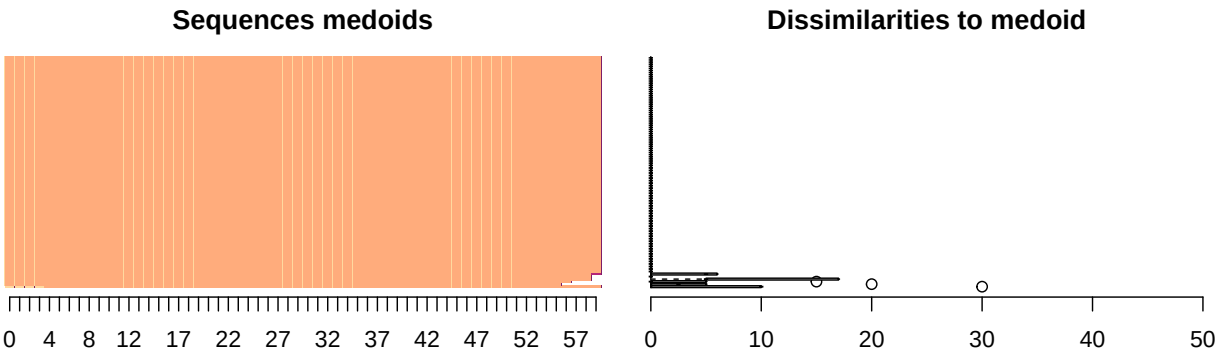
Representation quality: R2=0.56 and F=2.00

```
seqplot.rf(seqObj2[wardrange.OM.6 == 2,], diss=Matrix.OM.2, k=100, xtlab=xtlab, yaxis=FALSE)
```



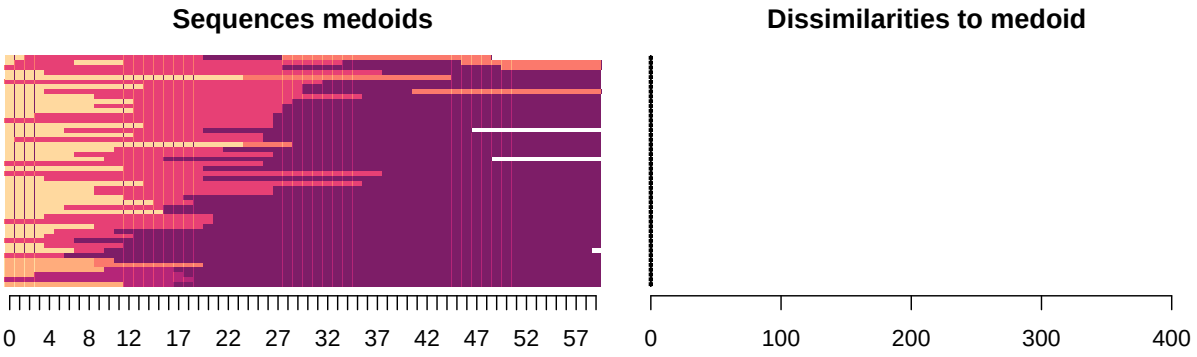
Representation quality: R2=0.82 and F=37.03

```
seqplot.rf(seqObj2[wardrange.OM.6 == 3,], diss=Matrix.OM.3, k=100, xtlab=xtlab, yaxis=FALSE)
```



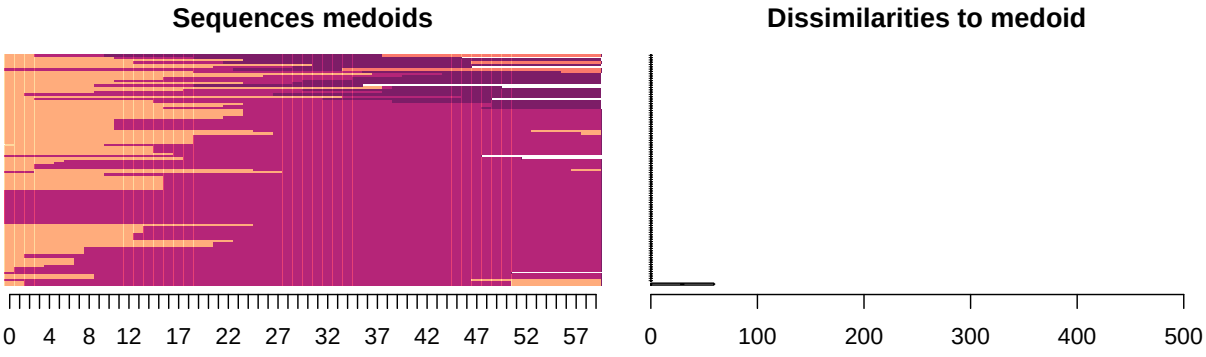
Representation quality: R2=0.81 and F=19.27

```
seqplot.rf(seqObj2[wardrange.OM.6 == 4,], diss=Matrix.OM.4, k=100, xtlab=xtlab, yaxis=FALSE)
```



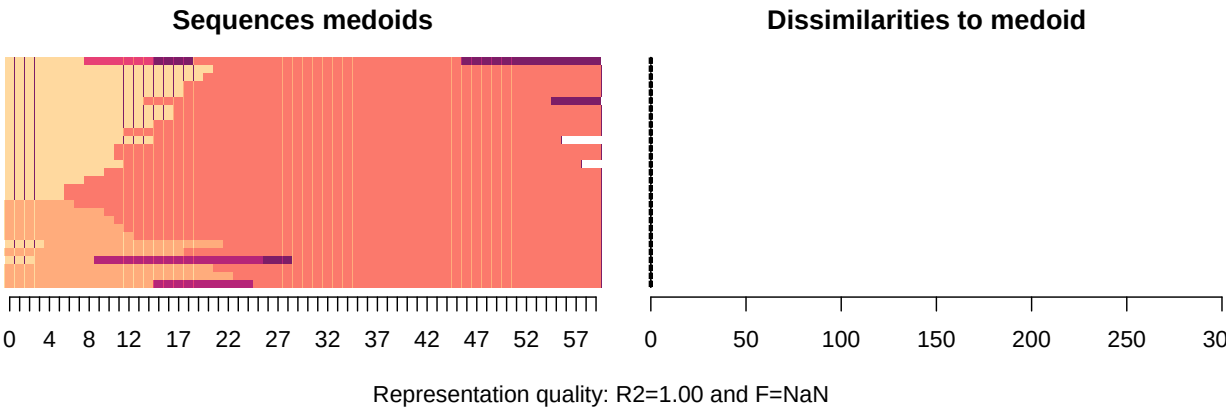
Representation quality: R2=1.00 and F=NaN

```
seqplot.rf(seqObj2[wardrange.OM.6 == 5,], diss=Matrix.OM.5, k=100, xtlab=xtlab, yaxis=FALSE)
```



Representation quality: R2=1.00 and F=4.07

```
seqplot.rf(seqObj2[wardrange.OM.6 == 6,], diss=Matrix.OM.6, k=100, xtlab=xtlab, yaxis=FALSE)
```

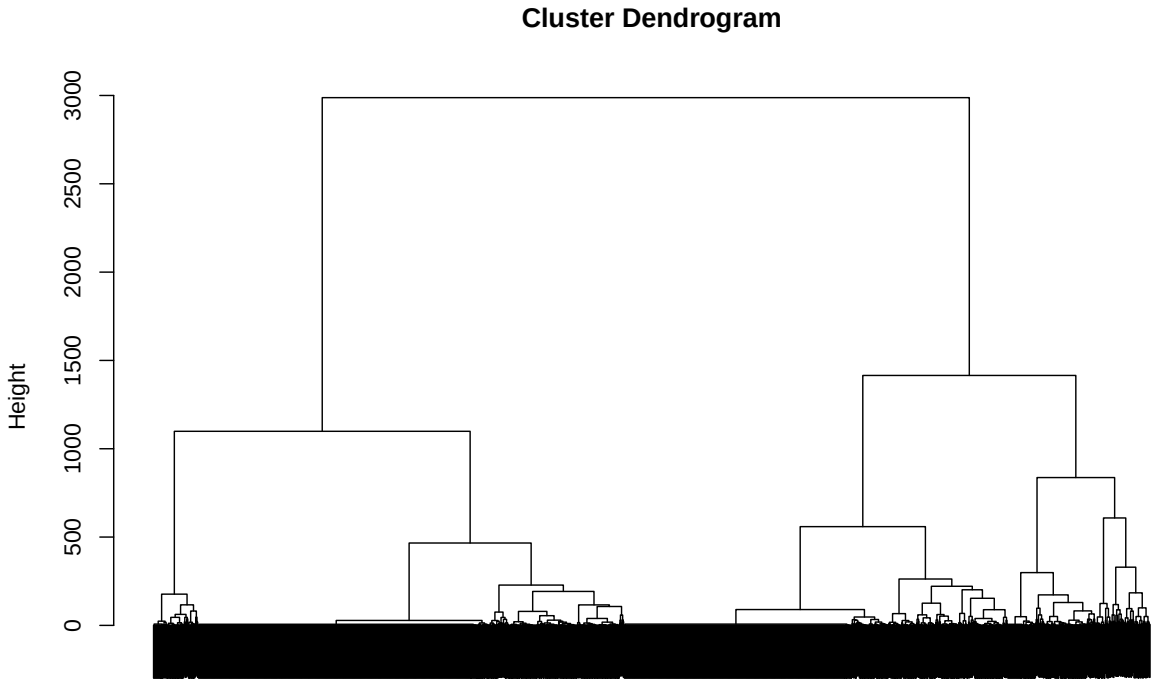


11) Repeat steps 8-10 using an OM with transition rates as substitution costs, and 1 as indel costs.

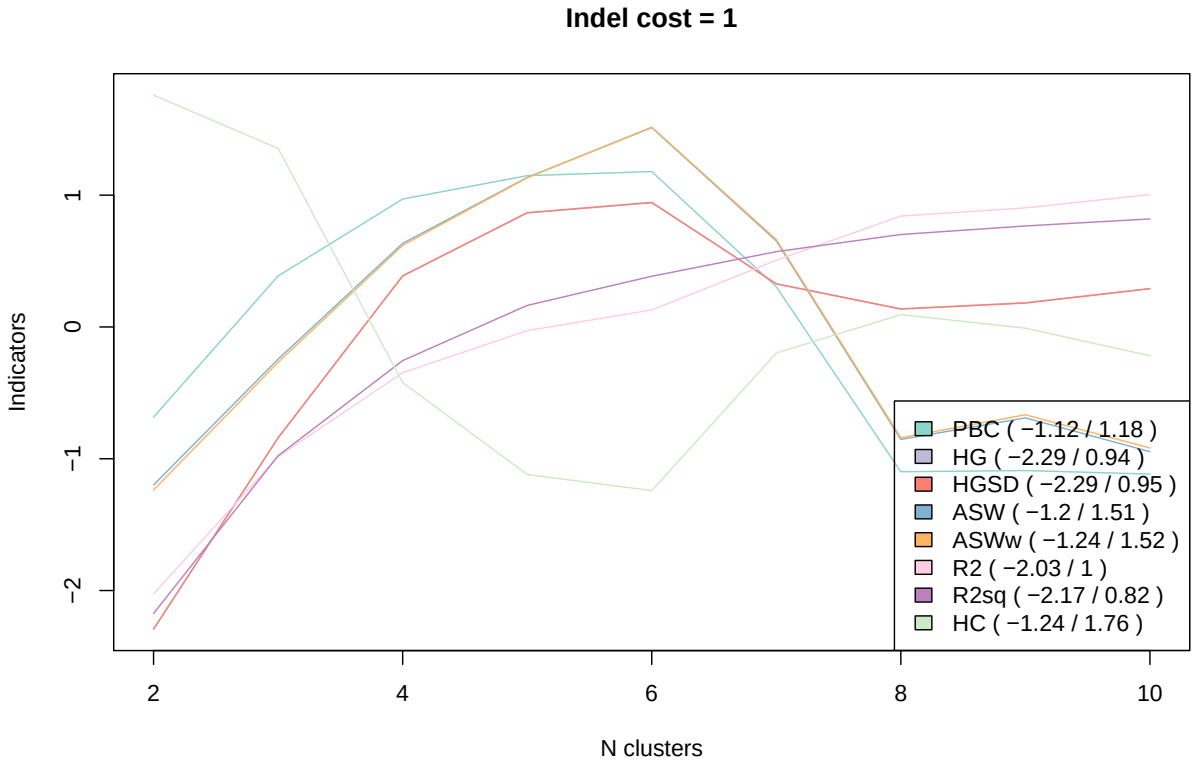
[Sol.]

Since we have already used the transition rates matrix as a proxy of substitution costs, now we lower the indel cost from 5 to 1 and compare results.

```
# OM dissimilarity matrix
Matrix.OM.TRATE <- seqdist(seqObj2, method="OM", indel=1, sm="TRATE")
# Cluster sequences using the OM distances with state-dependent costs and Ward method
ward.OM.TRATE <- hclust(as.dist(Matrix.OM.TRATE), method = "ward.D2")
# Dendogram
plot(ward.OM.TRATE, labels=FALSE)
```

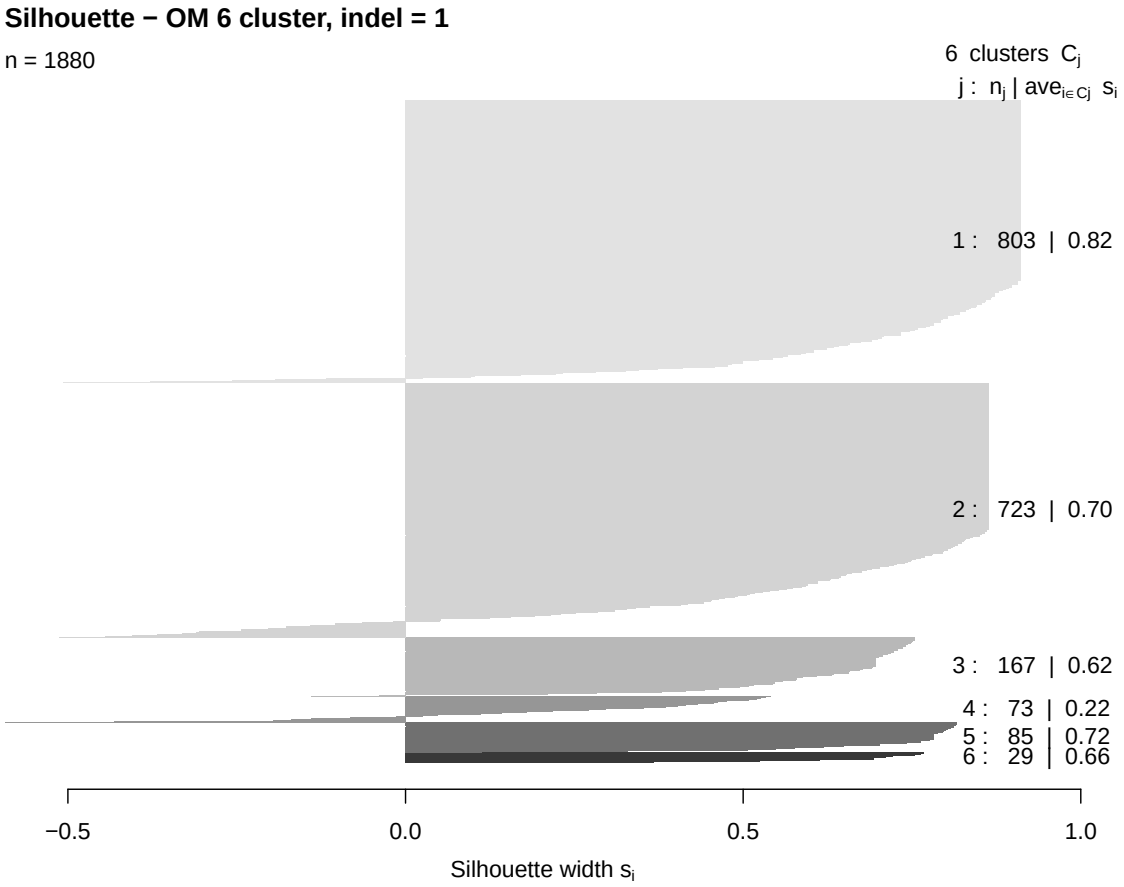



```
# Generate an object with 1-10 cluster solutions
wardrange.OM.T <- as.clustrange(ward.OM.TRATE, diss=Matrix.OM.TRATE, ncluster=10)
plot(wardrange.OM.T, norm="zscore", main="Indel cost = 1")
```



By lowering the indel costs, we have a more clear indication that the best fit is obtained with 6 clusters.

```
# Store cluster 6 cluster solution
wardrange.OM.T6 <- cutree(ward.OM.TRATE, k=6)
# Cluster consistency (plot silhouette widths)
silh.OM.T6 <- silhouette(wardrange.OM.T6, dmatrix = Matrix.OM.TRATE)
# Summary(silh.OM.T6)
plot(silh.OM.T6, main= "Silhouette - OM 6 cluster, indel = 1", border=NA,
     col=c("#E2E2E2", "#D3D3D3", "#B8B8B8", "#969696", "#707070", "#383838"))
```



By choosing again the solution with 6 clusters, also the average silhouette improve (0.72).

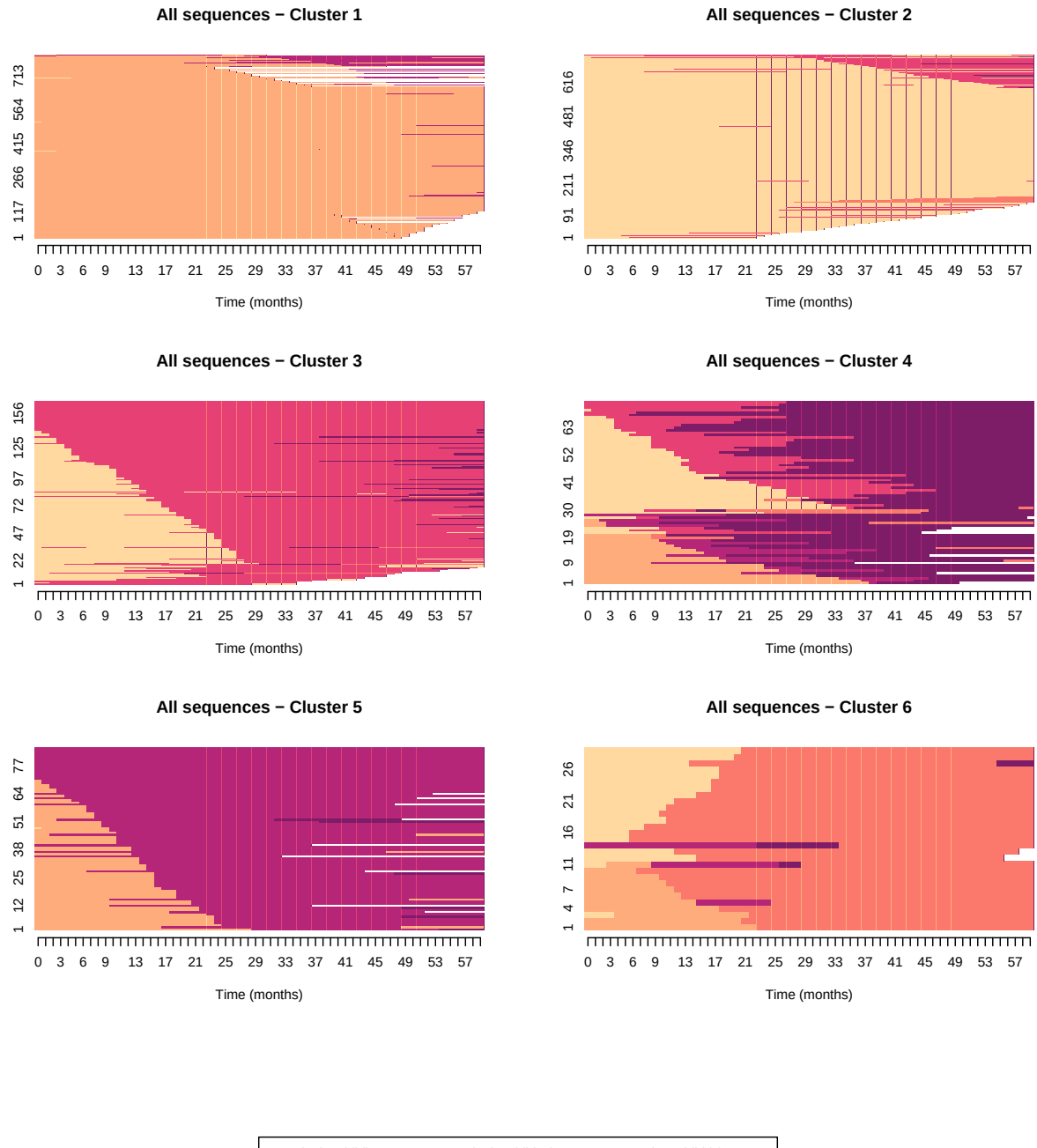
We understand that if we use the transition rates matrix as a substitution matrix cost, we need lower indel costs than usual to have better fitting for clustering.

12) Compare the results between the OM and the OM with transition rates as substitution costs, and 1 as indel costs approaches.

[Sol.]

Let's look at the full sequence index plot by cluster.

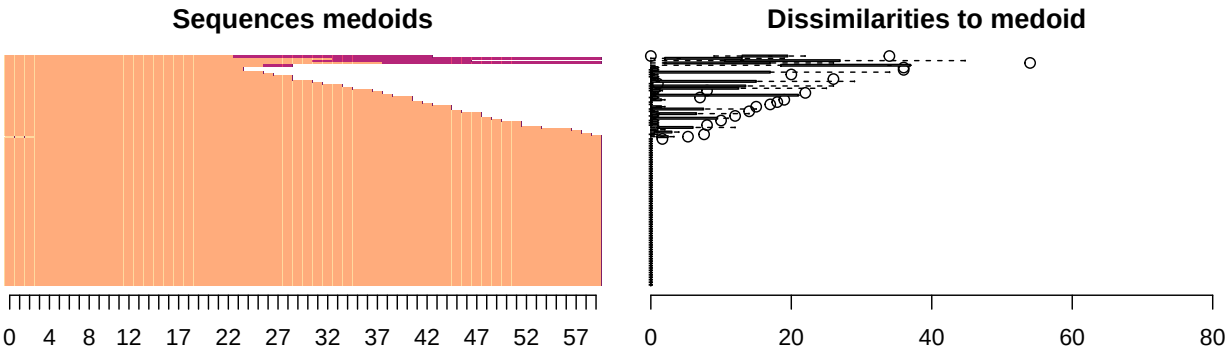
```
# Generate a variable that contains the OM distance of each sequence to repseq1
repseq2 <- seqrep(seqObj2, diss=Matrix.OM.TRATE, criterion="density", nrep=1)
OM.T.distRepseq <- seqdist(seqObj2, refseq = repseq2, method = "OM",
                        indel=1, sm="TRATE")
# Define the cluster variable
cl.6fac2 <- factor(wardrange.OM.T6, labels = paste("Cluster", 1:6))
# Plot the full sequence index plot
seqIplot(seqObj2, group=cl.6fac2, with.legend=TRUE,
         main= "All sequences", xtlab=xtlab, xlab="Time (months)",
         ylab=NA, yaxis=TRUE, border=NA, sortv=OM.T.distRepseq)
```



We can also look at the relative frequency plot.

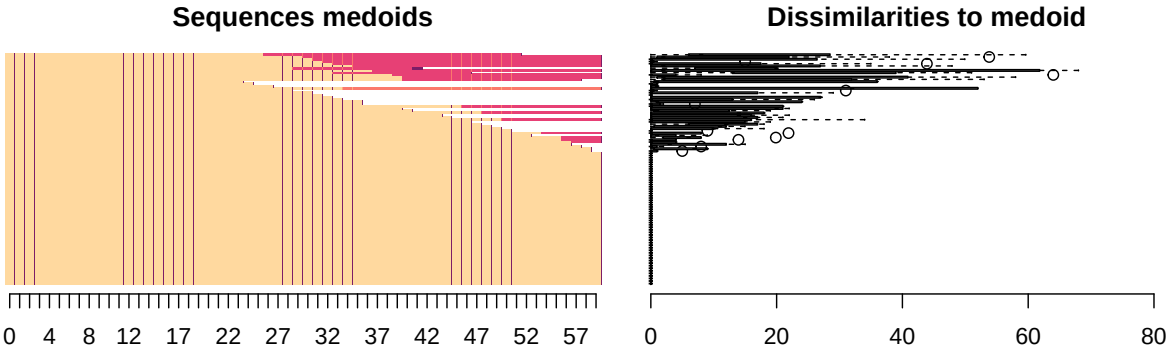
```
# Compute cluster specific matrix distances (OM 6-cluster solution)
Matrix.OM.TRATE.1 <- seqdist(seqObj2[wardrange.OM.T6 == 1,], method="OM", sm="TRATE")
Matrix.OM.TRATE.2 <- seqdist(seqObj2[wardrange.OM.T6 == 2,], method="OM", sm="TRATE")
Matrix.OM.TRATE.3 <- seqdist(seqObj2[wardrange.OM.T6 == 3,], method="OM", sm="TRATE")
Matrix.OM.TRATE.4 <- seqdist(seqObj2[wardrange.OM.T6 == 4,], method="OM", sm="TRATE")
Matrix.OM.TRATE.5 <- seqdist(seqObj2[wardrange.OM.T6 == 5,], method="OM", sm="TRATE")
Matrix.OM.TRATE.6 <- seqdist(seqObj2[wardrange.OM.T6 == 6,], method="OM", sm="TRATE")
```

```
# Compute the cluster specific relative frequency sequence index plots (OM 6-cluster solution)
par(mfrow=c(5,5))
seqplot.rf(seqObj2[wardrange.OM.T6 == 1,], diss=Matrix.OM.TRATE.1, k=100, xtlab=xtlab, yaxis=FALSE)
```



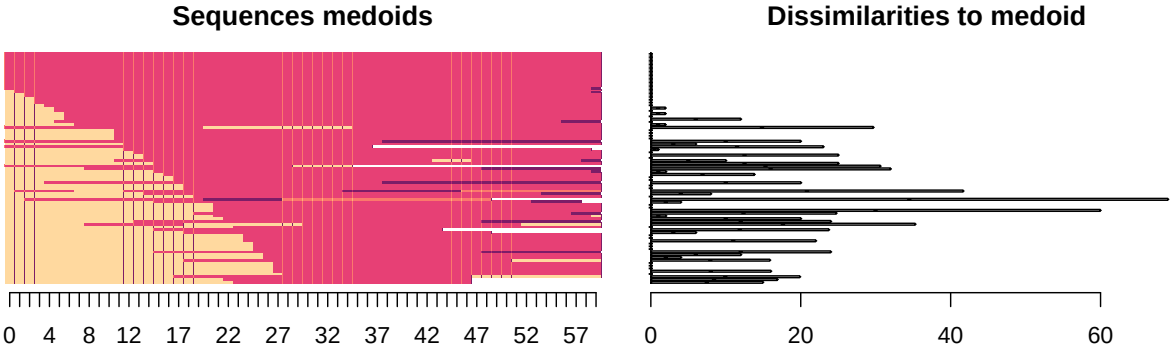
Representation quality: R2=0.85 and F=40.92

```
seqplot.rf(seqObj2[wardrange.OM.T6 == 2,], diss=Matrix.OM.TRATE.2, k=100, xtlab=xtlab, yaxis=FALSE)
```



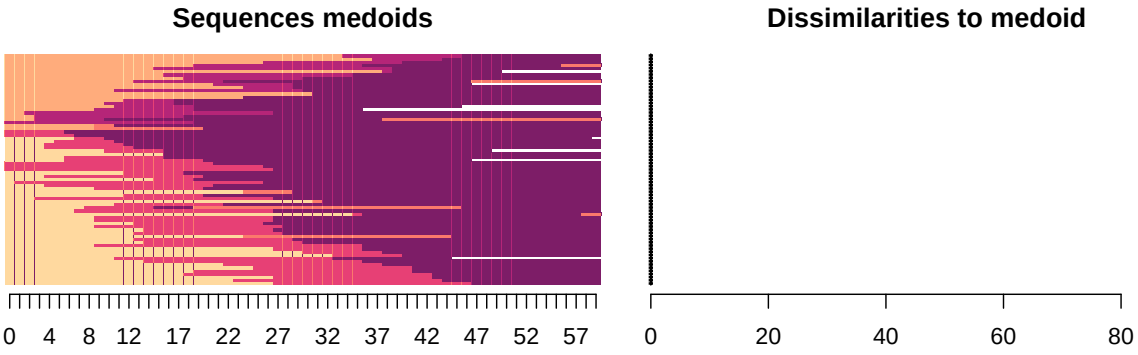
Representation quality: R2=0.74 and F=18.32

```
seqplot.rf(seqObj2[wardrange.OM.T6 == 3,], diss=Matrix.OM.TRATE.3, k=100, xtlab=xtlab, yaxis=FALSE)
```



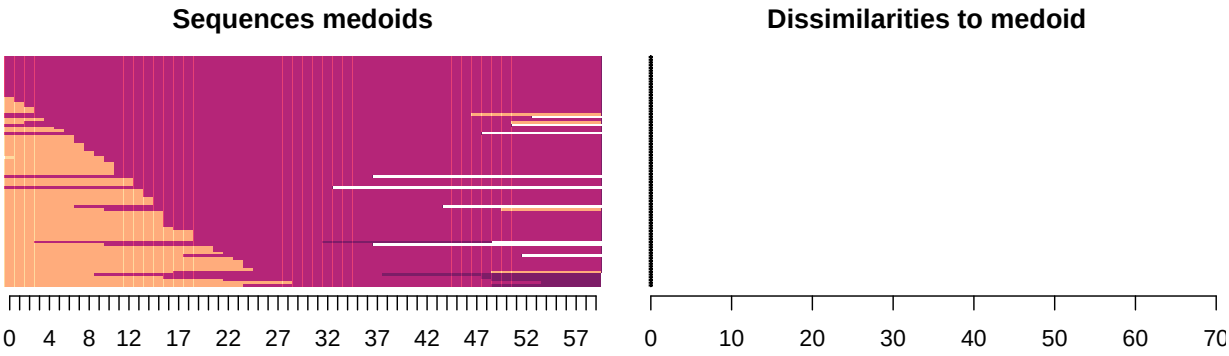
Representation quality: R2=0.74 and F=1.93

```
seqplot.rf(seqObj2[wardrange.OM.T6 == 4,], diss=Matrix.OM.TRATE.4, k=100, xtlab=xtlab, yaxis=FALSE)
```



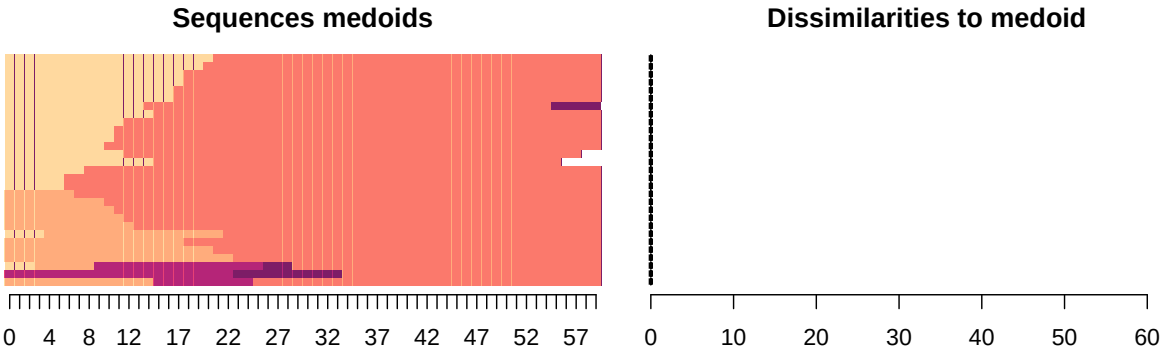
Representation quality: R2=1.00 and F=NaN

```
seqplot.rf(seqObj2[wardrange.OM.T6 == 5,], diss=Matrix.OM.TRATE.5, k=100, xtlab=xtlab, yaxis=FALSE)
```



Representation quality: R2=1.00 and F=NaN

```
seqplot.rf(seqObj2[wardrange.OM.T6 == 6,], diss=Matrix.OM.TRATE.6, k=100, xtlab=xtlab, yaxis=FALSE)
```



Representation quality: R2=1.00 and F=NaN

References