Sequence analysis for social scientists

Introduction to sequence analysis

Alexis Gabadinho, Matthias Studer, Gilbert Ritschard, Nicolas S. Müller

Department of Econometrics, University of Geneva
http://mephisto.unige.ch/biomining

Summer School on Advanced Methods for the Analysis of Complex Event History Data, Bristol, 28-29 June 2010

Basics concepts

- State sequences: ordered lists of states defined on a time axis
- In the social sciences, such state sequences are of interest for studying life trajectories such as
  - occupational histories
  - professional carriers
  - cohabitational life courses
- Data collected for example by panel or retrospective surveys
- The states are values of a categorical variable (hence we talk also of categorical sequences)
- Event sequences: ordered lists of time stamped events

State sequences

Definitions

- Alphabet A: finite set of possible states, for example: leaves in the parental home, leaves alone, leaves with partner and no children, leaves with partner and one or more children, leaves with no partner and one or more children
- Sequence of length k: ordered list of k elements taken from A

- State sequences are present in many domains:
  - Text: A = set of letters, words, structure of sociological articles **?**, ...
  - Biology: A = set of nucleotids, proteins, ...
  - Trajectories, biographies: A = set of leaving arrangements, professional status, daily activities (time-use diary data, workdays, workweeks), ...

Example: The mvad data set

- Study from**?** on transition from school to work in Northern Ireland
- Included in the TraMineR library.
- 712 individuals
- Follow-up starting at the end of the compulsory education (July 1993)
- Time series of 70 status variables: September 1993 to June 1999.
- The alphabet is made of the following statuses: EM (employment), FE (Further Education), HE (Higher Education), JL (Joblessness), SC (School), TR (Training).

Aim of the original study

- The aim of the original study was to
  - Use sequence techniques to characterize young peoples' transitions from school to work into types
  - Distinguish between successful and unsuccessful transitions
  - Link transition type with a collection of static individual, family and school characteristics
  - Identify which young people are more at risk to experience unsuccessful transitions into the labour market

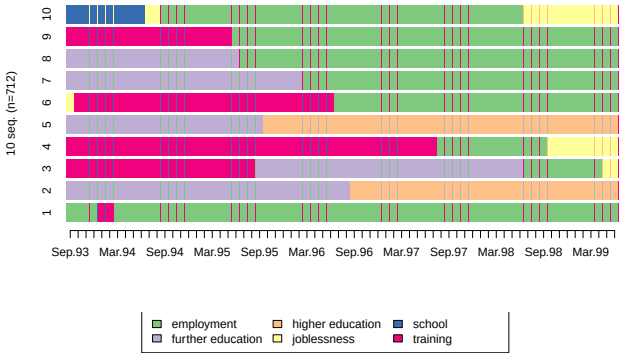
The mvad data set - Variable list

Table: List of Variables in the MVAD data set

id	unique individual identifier
weight	sample weights
male	binary dummy for gender, 1=male
catholic	binary dummy for community, 1=Catholic
Belfast	binary dummies for location of school, one of five Education and Library Board areas in Northern Ireland
N.Eastern	"
Southern	"
S.Eastern	"
Western	"
Grammar	binary dummy indicating type of secondary education, 1=grammar school
funemp	binary dummy indicating father's employment status at time of survey, 1=father unemployed
gcse5eq	binary dummy indicating qualifications gained by the end of compulsory education, 1=5+ GCSEs at grades A-C, or equivalent
fmpr	binary dummy indicating SOC code of father's current or most recent job,1=SOC1 (professional, managerial or related)
livboth	binary dummy indicating living arrangements at time of first sweep of survey (June 1995), 1=living with both parents
jul93	Monthly Activity Variables are coded 1-6, 1=school, 2=FE, 3=employment, 4=training, 5=joblessness, 6=HE
:	"
jun99	"

Graphical representation of state sequences

- Here is a graphical representation of the first 10 state sequences



The holistic approach

- Holistic approach of the life courses versus "event-oriented" approach [**?**]
- Approach promoted by **?**
- Event history or survival analysis focuses on the occurrence of one specific event in the life course
- Sequence analysis allows a more general overview of the life courses
- The holistic approach is based originally on exploratory analysis (data mining techniques)
- Explanatory methods developed more recently

Typical questions

- Some of the typical questions arising in the social sciences:
  - Do life courses obey some social norm? Which are the standard trajectories? What kind of departures do we observe from these standards ? Do we observe a de-standardization of the life trajectories ?
  - Why are some people more at risk to follow a chaotic trajectory?
  - How is the life trajectory related to sex, social origin and other cultural factors?
  - How is a given outcome, such as health status or income, related to a trajectory?

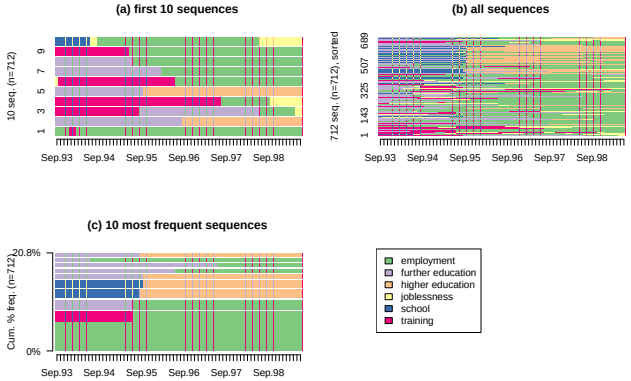
Which tools do we need?

- By considering whole sequences we deal with complex objects
- We need special tools for describing and displaying them
- Questions regarding the exploration and description of sets of sequences such as
  - Which characteristics of sequences are we interested in?
  - What kind of indicators can we compute for a sequence set?
  - Which plots are suited for rendering sequences?
  - How can we measure similarity between sequences?
- From a more explanatory perspective, how can we measure and assess the link between covariates and whole sequences?

Objectives

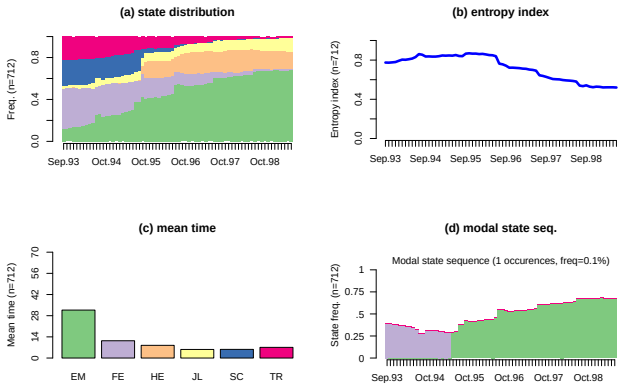
During this course, you will learn to:

- Use the R statistical environnement and the TraMineR package.
- Visualize state sequences with various representations



Objectives - II

- Describe sets of sequences using overall and transversal characteristics.



- Describe individual sequences using longitudinal characteristics.

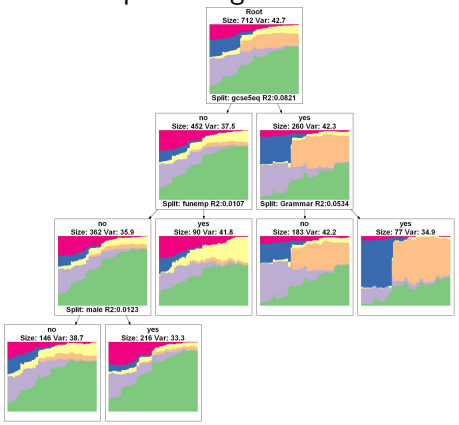
Objectives - III

- Build and visualize a typology of sequences

- Relate this typology with covariates using logistic regression.

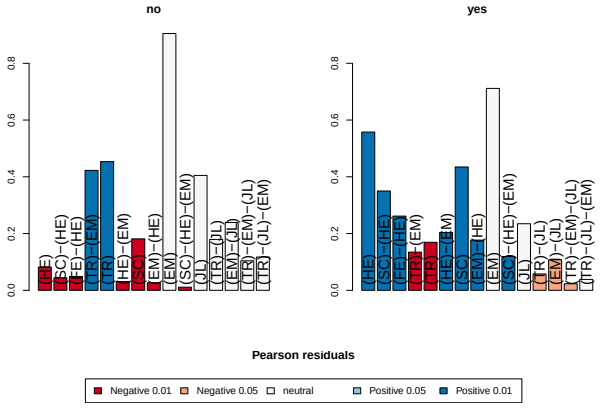
Objectives - IV

- Measure and assess the association between sequences and one or several covariates using sequence discrepancy analysis.
- Create sequence regression trees.



Objectives - V

- Focus on event rather than states by using event sequences analysis.
- Find the most frequent event subsequences.
- Find the most discriminating event subsequences.



References I

Andrew Abbott and Emily Barman. Sequence comparison via alignment and gibbs sampling: A formal analysis of the emergence of the modern sociological article. *Sociological Methodology*, 27:47–87, 1997.

Andrew Abbott and John Forrest. Optimal matching methods for historical sequences. *Journal of Interdisciplinary History*, 16: 471–494, 1986.

Francesco C. Billari. Sequence analysis in demographic research. *Canadian Studies in Population*, 28(2):439–458, 2001. Special Issue on Longitudinal Methodology.

Duncan McVicar and Michael Anyadike-Danes. Predicting successful and unsuccessful transitions from school to work by using sequence methods. *Journal of the Royal Statistical Society. Series A (Statistics in Society)*, 165(2):317–334, 2002. ISSN 09641998. URL `http://www.jstor.org/stable/3559930`.

Sequence analysis for social scientists

Part II - Introduction to R

Alexis Gabadinho, Matthias Studer, Gilbert Ritschard, Nicolas S. Müller

Department of Econometrics, University of Geneva
http://mephisto.unige.ch/biomining

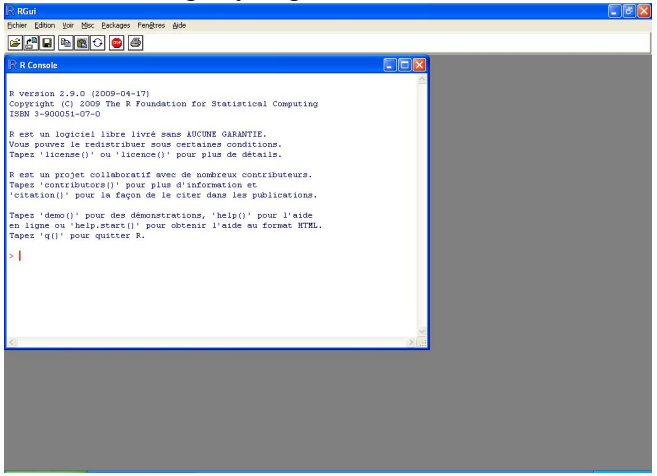
Summer School on Advanced Methods for the Analysis of Complex Event History Data, Bristol, 28-29 June 2010

What is R?

- R is a free software environment for statistical computing and graphics (can be downloaded at `http://www.r-project.org`)
- R is also a language derived from the S language
- R is free and open source
- R is available for Linux, MacOS X, Windows

The R console

- When launching R you get a R console

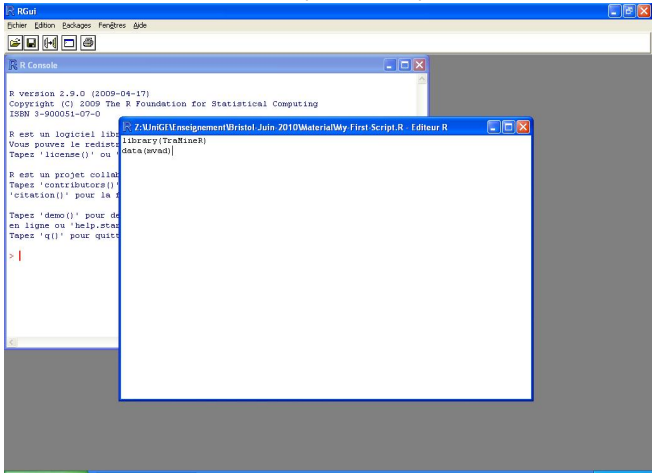


Passing commands to R

- The prompt '>' indicates that R is waiting commands
- There are several ways of passing commands to R
  - By typing them directly in the console
  - By using the script editor (Windows and MacOS X only)
  - By using the R Commander graphical interface (available from the CRAN)
  - By using an external editor
- R scripts are files containing a series of R commands
- To add comments to your code use '#' before
- The usual extension for such files is '.R'
- Using scripts allows to
  - store, re-use or modify later your statistical analyses
  - send to/share with someone else

The script editor

- Here is the script editor (Windows)



R objects

- In R you handle objects that can be of many different types, and have different content types
- An object is created with the "assign" operator: "<-"

```
> a <- 5
> b <- "my object"
```

- The content of an object is displayed by typing its name

```
> a
[1] 5
> b
[1] "my object"
```

- Object names are case sensitive ($a \neq A$) and can be of arbitrary length

```
> My.very.first.R.object <- 13
> My.very.first.R.object
[1] 13
```


Operators and functions

- Operations on objects

```
> a/2
[1] 2.5
> A <- 8
> a == A
[1] FALSE
```

- A function is always called by typing its name followed by brackets.
- Inside the brackets you put zero or more arguments. Arguments can be values or existing objects

```
> b
[1] "my object"
> c <- paste(b, "is beautiful")
> c
[1] "my object is beautiful"
```


Function arguments

- You can usually omit the main argument names, but it is better to give explicitly the names of the other optional arguments you may have to pass to functions

```
> seq(from = 1, to = 10, by = 2)
[1] 1 3 5 7 9
> seq(1, 10, 2)
[1] 1 3 5 7 9
```

- Using argument names allows to pass them in any order

```
> seq(by = 2, to = 10, from = 1)
[1] 1 3 5 7 9
> seq(2, 10, 1)
[1]  2  3  4  5  6  7  8  9 10
```

Functions and libraries

- Many standard statistical functions are available as core functions: descriptive statistics, regression, etc ...
- Additional, specialized functions are available through
  - pre-installed libraries for example foreign for reading data from other statistical packages, survival for survival analysis, etc ...
  - add-on libraries, available from the Comprehensive R Archive Network (CRAN), for example TraMineR for sequence analysis.
- Use install.packages() function to install a library from the CRAN.

Help on functions and libraries

- To access the functions provided by a library

```
> library(TraMineR)
```

- To get information on a library

```
> library(help = TraMineR)
```

- To get help on a particular function

```
> library(foreign)
> help(read.spss)
```


Higher dimensional objects - Vectors

- Vectors are one-dimensional objects containing numeric or character values. The c() function combines values into a vector

```
> v1 <- c(1, 2, 4, 8)
> v1
[1] 1 2 4 8
> v2 <- c("A", "B", "C", "D")
> v2
[1] "A" "B" "C" "D"
```

- Specific elements of vectors can be retrieved with indexes

```
> v1[3]
[1] 4
> v2[1:3]
[1] "A" "B" "C"
> v2[c(1, 4)]
[1] "A" "D"
```


Indexing with logical expressions

- One can use logical expressions to retrieve vectors elements

```
> v1[v1 >= 4]
[1] 4 8
> v2[v2 %in% c("A", "C")]
[1] "A" "C"
```

- Use which() to get the indexes of the elements satisfying one condition

```
> which(v1 >= 4)
[1] 3 4
> which(v2 %in% c("A", "C"))
[1] 1 3
```

Higher dimensional objects - Matrices

- Matrices are two dimensional objects containing numeric or character values

```
> m1 <- matrix(1:16, 4, 4)
> m1
```

```
     [,1] [,2] [,3] [,4]
[1,]    1    5    9   13
[2,]    2    6   10   14
[3,]    3    7   11   15
[4,]    4    8   12   16
```

- Specific elements of matrices are retrieved with row and column indexes
  - Element in the second row, fourth column

```
> m1[2, 4]
```

```
[1] 14
```

  - Whole fourth column (by omitting row index)

```
> m1[, 4]
```

```
[1] 13 14 15 16
```


Higher dimensional objects - Data frames

- Data frames combine columns (vectors) of any type: factors, numeric, character strings

```
> data(iris)
> iris[1:4, ]
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
```

- You can access variables in data frames using their name preceded by a '$' instead of the column number

```
> iris$Sepal.Width[1:4]
```

```
[1] 3.5 3.0 3.2 3.1
```

```
> iris[1:4, 2]
```

```
[1] 3.5 3.0 3.2 3.1
```


Factors

- Factors are categorical variables, that is variables taking a value that is one of several possible categories
- The Species variable in the iris data frame is a factor

```
> class(iris$Species)
```

```
[1] "factor"
```

- Possible categories of a factor are called levels

```
> levels(iris$Species)
```

```
[1] "setosa"     "versicolor" "virginica"
```

- You can change the labels for the levels

```
> levels(iris$Species) <- c("Species 1", "Species 2", "Species 3")
> head(iris$Species)
```

```
[1] Species 1 Species 1 Species 1 Species 1 Species 1 Species 1
Levels: Species 1 Species 2 Species 3
```


Objects and methods

- There are many other types of objects in R
- For example contingency tables or outputs of regression models are objects of a specific type
- There are usually specific methods for each type of object like print(), plot() or summary() methods
- Indeed when typing the name of an object to display its content the print() method is called automatically

```
> A
```

```
[1] 8
```

```
> print(A)
```

```
[1] 8
```

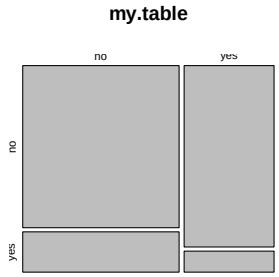
Objects and methods - Example

- We use the table() function to produce a contingency table

```
> my.table <- table(mvad$gcse5eq, mvad$funemp)
> class(my.table)
[1] "table"
```

- The dedicated plot() method produces the following figure

```
> plot(my.table)
```



Importing text files

- Importing from '.csv' (comma separated values) text files

```
> my.data <- read.csv("data/csv-example.csv")
> my.data
  Id Age    Sex
1  1  22   Male
2  2  18 Female
3  3  40   Male
4  4  27 Female
5  5  33 Female
```

- Importing from tab separated text files with read.table()

Importing from other statistical packages

- Importing from SPSS with read.spss()

```
> titanic <- read.spss("data/titanic.sav", to.data.frame = TRUE)
> head(titanic)
  ID CLASS   AGE  SEX LIVING
1  1    c1 adult Male    yes
2  2    c1 adult Male    yes
3  3    c1 adult Male    yes
4  4    c1 adult Male    yes
5  5    c1 adult Male    yes
6  6    c1 adult Male    yes
```

- Importing from Stata, with read.dta()

The mvad data frame

- The *mvad* data set is included in the TraMineR library (with permission of the authors) and can be loaded with

```
> data(mvad)
```

- The *mvad* object is of type '`data frame`'. It contains data from different formats (numeric values, factors)

```
> class(mvad)
[1] "data.frame"
```

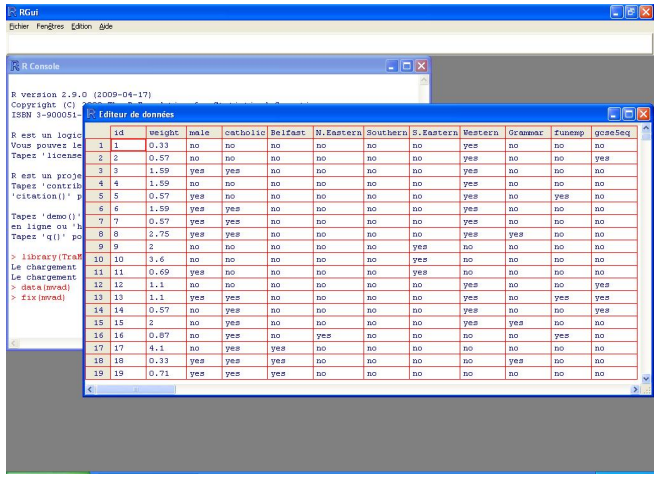
- It contains 712 rows and 86 variables

```
> dim(mvad)
[1] 712  86
> nrow(mvad)
[1] 712
> ncol(mvad)
[1] 86
```

The data editor

- The fix() function opens a simple data editor

```
> fix(mvad)
```



Variable names

- To obtain the list of all variables in the 'data frame'

```
> names(mvad)
```

```
 [1] "id"         "weight"     "male"       "catholic"   "Belfast"    "N.Eastern"
 [7] "Southern"   "S.Eastern"  "Western"    "Grammar"    "funemp"     "gcse5eq"
[13] "fmpr"       "livboth"    "Jul.93"     "Aug.93"     "Sep.93"     "Oct.93"
[19] "Nov.93"     "Dec.93"     "Jan.94"     "Feb.94"     "Mar.94"     "Apr.94"
[25] "May.94"     "Jun.94"     "Jul.94"     "Aug.94"     "Sep.94"     "Oct.94"
[31] "Nov.94"     "Dec.94"     "Jan.95"     "Feb.95"     "Mar.95"     "Apr.95"
[37] "May.95"     "Jun.95"     "Jul.95"     "Aug.95"     "Sep.95"     "Oct.95"
[43] "Nov.95"     "Dec.95"     "Jan.96"     "Feb.96"     "Mar.96"     "Apr.96"
[49] "May.96"     "Jun.96"     "Jul.96"     "Aug.96"     "Sep.96"     "Oct.96"
[55] "Nov.96"     "Dec.96"     "Jan.97"     "Feb.97"     "Mar.97"     "Apr.97"
[61] "May.97"     "Jun.97"     "Jul.97"     "Aug.97"     "Sep.97"     "Oct.97"
[67] "Nov.97"     "Dec.97"     "Jan.98"     "Feb.98"     "Mar.98"     "Apr.98"
[73] "May.98"     "Jun.98"     "Jul.98"     "Aug.98"     "Sep.98"     "Oct.98"
[79] "Nov.98"     "Dec.98"     "Jan.99"     "Feb.99"     "Mar.99"     "Apr.99"
[85] "May.99"     "Jun.99"
```

- A description of the data set and variables is available with

```
> help(mvad)
```


Summary

- We get a summary for the first five variables with

```
> summary(mvad[, 1:5])
```

```
       id            weight        male      catholic  Belfast
 Min.   :  1.0   Min.   :0.1300   no :342   no :368   no :624
 1st Qu.:178.8   1st Qu.:0.4500   yes:370   yes:344   yes: 88
 Median :356.5   Median :0.6900
 Mean   :356.5   Mean   :0.9994
 3rd Qu.:534.2   3rd Qu.:1.0700
 Max.   :712.0   Max.   :4.4600
```

- The weight variable is numeric while catholic is a factor

```
> levels(mvad$catholic)
```

```
[1] "no"  "yes"
```


Frequency tables

- The states we will use to build sequences are in variables Jul.93...Jun.99, that is columns 15 to 86

```
> mvad[1:4, 15:20]
```

```
       Jul.93      Aug.93     Sep.93     Oct.93     Nov.93     Dec.93
1    training    training employment employment employment employment
2 joblessness joblessness         FE         FE         FE         FE
3 joblessness joblessness   training   training   training   training
4    training    training   training   training   training   training
```

- A frequency table of the gcse5eq variable (qualifications gained by the end of compulsory education)

```
> table(mvad$gcse5eq)
```

```
 no yes
452 260
```

Contingency tables

- We want a contingency table for variables funemp (father unemployed) and gcse5eq (qualification gained at the end of compulsory school)
- We change first the labels of the two factors (both are dummy variables whose labels are "yes/no" in the original file)

```
> levels(mvad$funemp) <- c("employed", "unemployed")
> levels(mvad$gcse5eq) <- c("Lower qual.", "Higher qual.")
> t1 <- table(mvad$funemp, mvad$gcse5eq)
> t1
```

```
            Lower qual. Higher qual.
employed            362          233
unemployed           90           27
```


Row and marginal distributions

- Row and column distributions

```
> prop.table(t1, 1)
```

```
            Lower qual. Higher qual.
employed      0.6084034    0.3915966
unemployed    0.7692308    0.2307692
```

```
> prop.table(t1, 2)
```

```
            Lower qual. Higher qual.
employed      0.8008850    0.8961538
unemployed    0.1991150    0.1038462
```

- Margins

```
> margin.table(t1, 1)
```

```
employed unemployed
     595        117
```

```
> margin.table(t1, 2)
```

```
Lower qual. Higher qual.
       452          260
```


Contingency tables

- We perform a chi-squared contingency table test using the chisq.test() function

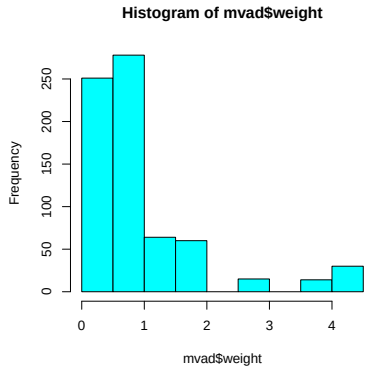
```
> chisq.test(t1)
```

```
Pearson's Chi-squared test with Yates' continuity correction

data:  t1
X-squared = 10.2264, df = 1, p-value = 0.001384
```


Histograms

- Histograms are for numerical variables like weight

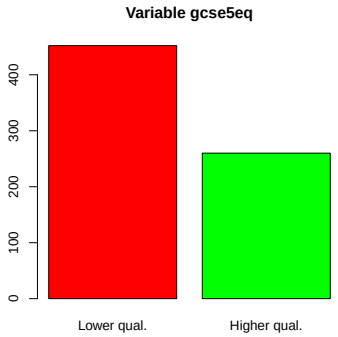
```
> hist(mvad$weight, col = "cyan")
```



Barplots

- Use the plot() method for factors (categorical variables)

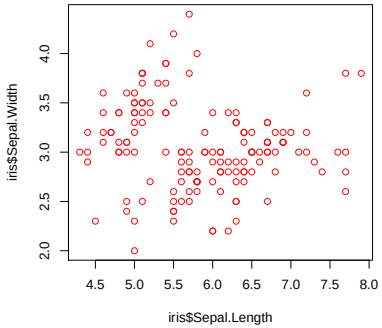
```
> plot(mvad$gcse5eq, col = c("red", "green"), main = "Variable gcse5eq")
```



XY plots

- Use the plot() method for two numerical variables

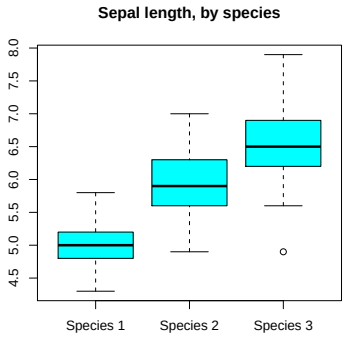
```
> plot(iris$Sepal.Length, iris$Sepal.Width, col = "red")
```



Boxplots

- The boxplot() function accepts a formula as argument to produce one boxplot for each category of a factor

```
> boxplot(iris$Sepal.Length ~ iris$Species, col = "cyan", main = "Sepal length, by specie
```



Saving graphics

- To save graphics in files, depending on the format you can use the pdf(), jpeg() or png() function with the name of the file as argument
- Once you have issued all plotting commands you have to close the file with the dev.off() function

```
> pdf(file = "hist")
> plot(mvad$Sep.93, mvad$Sep.94)
> dev.off()

pdf
  2
```

Some useful functions - A

- We add a covariate to the mvad data frame that is just the weight covariate multiplied by 100

```
> mvad$weight100 <- mvad$weight * 100
```

- head() displays the first parts of an object (elements of a vector, rows of a matrix or data frame)

```
> head(mvad$weight100)
[1]  33  57 159 159  57 159
```

- Some basic statistical functions

```
> mean(mvad$weight100)
[1] 99.9396
> min(mvad$weight100)
[1] 13
> max(mvad$weight100)
[1] 446
```


Some useful functions - B

- To produce a regular sequence use either seq(from,to) or from:to

```
> my.seq <- 1:99
> head(my.seq)
[1] 1 2 3 4 5 6
```

- cut() converts numerical covariates into factors (right=FALSE is for open intervals on the right)

```
> my.groups <- cut(my.seq, c(0, 20, 40, 60, 100), labels = c("0-19",
+       "20-39", "40-59", "60-100"), right = FALSE)
> table(my.groups)
my.groups
  0-19  20-39  40-59 60-100
    19     20     20     40
```


Exercise

Exercise 2.1

1. Load the biofam data set that comes with the TraMineR library
2. Print the variable names
3. Add an age variable, by subtracting the birth year from the year of the survey
4. What is the min., max., median and mean age in the sample?
5. What is the min., max., median and mean age of the woman?
6. Add a cohort factor to the biofam data frame grouping the birth years into the following categories: 1900-1929, 1930-1939, 1940-1949, 1950-1959.
7. Produce a frequency table of the cohort factor

The working environment - A

- The objects you create during your session are stored in the working environment, that is in the memory
- List objects in your R environment

```
> ls()
 [1] "a"                  "A"
 [3] "b"                  "c"
 [5] "iris"               "m1"
 [7] "mvad"               "my.data"
 [9] "my.groups"          "my.seq"
[11] "my.table"           "My.very.first.R.object"
[13] "t1"                 "titanic"
[15] "v1"                 "v2"
```

- When you quit R (either with the menu or with the quit() function, you are asked if you want to save your working environment. If you answer yes, it is saved in a '.RData' file.

The working environment - B

- You can also save your working environment with the save.image() function or specific objects with the save() function.

```
> save.image(file = "myenv.RData")
> save(my.table, file = "data/mytable.RData")
```

- You get the current working directory with the getwd() function
- You can change the working directory with the setwd() function

References I

Carlos Alzola and Frank Harrell. *An Introduction to S and The Hmisc and Design Libraries*, 2006.

Julian J. Faraway. *Practical Regression and Anova using R*, July 2002.

Petra Kuhnert and Bill Venables. *An Introduction to R: Software for Statistical Modelling & Computing*. CSIRO Mathematical and Information Sciences, Cleveland, Australia, 2005. URL `http://www.csiro.au/resources/Rcoursenotes.html`.

Emmanuel Paradis. *R pour les débutants*. Institut des Sciences de l' Evolution Université Montpellier II, F-34095 Montpellier, septembre 2005.

W. N. Venables, D. M. Smith, and the R Development Core Team. *An Introduction to R*, 2009.

Sequence analysis for social scientists

Part III - Describing and visualizing sequence data sets

Alexis Gabadinho, Matthias Studer, Gilbert Ritschard, Nicolas S. Müller

Department of Econometrics, University of Geneva
http://mephisto.unige.ch/biomining

Summer School on Advanced Methods for the Analysis of Complex Event History Data, Bristol, 28-29 June 2010

Formats

- State sequences can be represented in many different ways, depending on the data source and on how the information is organized
  - States observed at the successive time points (panel surveys)
  - Spells, i.e. of distinct states stamped with their beginning/ending time (retrospective surveys)
- It is then essential for the user to identify the nature and organization of her/his data and to possibly transform it in an input form accepted by the software.
- Data organization and conversion between formats is discussed in details in **?**

The STS representation

- State sequence of length ℓ : ordered list of ℓ elements successively chosen from a finite set A (the *alphabet*) of size $a = |A|$.
- A natural way of representing a sequence x is $x = (x_1, x_2, \ldots, x_\ell)$, with $x_j \in A$. This is the STS representation.
- Such *state sequences* have two important properties
  1. They are formed by elements that are states, i.e. something that can last as opposed for instance to events that occur at given time points.
  2. The position of each element conveys meaningful information in terms of age, date or more generally elapsed time or distance from the beginning of the sequence.

Sequence representations

- More compact representations by giving only one of several same successive states (the Distinct Successive State (DSS) sequence) :
  A-A-A-B-B-C-C-C-C-D => A-B-C-D
- To keep time and alignment information successive distinct states are stamped with their duration (State Permanence Sequence (SPS) representation) ...
  A-B-C-D => (A,3)-(B,2)-(C,3)-(D,1)
- ... or their starting and ending positions (SPELL representation)

Start	*End*	*State*
1	3	A
4	5	B
6	9	C
10	10	D

Sequence representations - Example

- Two sequences describing family formation histories of two individuals
- Alphabet : S=single, M=married, MC=married with children, D=divorced

Format	Example
STS	*Id* *a18* *a19* *a20* *a21* *a22* *a23* *a24* *a25* *a26* *a27* 101 S S S M M MC MC MC MC D 102 S S S MC MC MC MC MC MC MC
DSS	*Id* *s1* *s2* *s3* *s4* 101 S M MC D 102 S MC
SPS	*Id* *s1* *s2* *s3* *s4* 101 (S,3) (M,2) (MC,4) (D,1) 102 (S,3) (MC,7)
SPELL	*Id* *Index* *From* *To* *State* 101 1 18 20 S 101 2 21 22 M 101 3 23 26 MC 101 4 27 27 D 102 1 18 20 S 102 2 21 27 MC

Converting between formats

- In this example the sequence s1 is in the State Permanence (SPS) representation and we get its STS representation with the seqformat() function

```
> s1 <- "(A,2)-(B,10)-(C,3)-(D,2)-(A,3)"
> s1
[1] "(A,2)-(B,10)-(C,3)-(D,2)-(A,3)"
> seqformat(s1, from = "SPS", to = "STS", compressed = TRUE)
    Sequence
[1] "A-A-B-B-B-B-B-B-B-B-B-B-C-C-C-D-D-A-A-A"
```


State sequence objects

- State sequence objects are required as input by the functions for state sequence analysis
- These objects store the state sequences (as a matrix) together with important attributes
  - The alphabet (the list of possible states)
  - The (long) labels associated with each state of the alphabet that will be used in the graphics
  - The colors for state legends
  - Information about missing states in the sequences
  - Possibly case weights

The mvad data frame - Variables list

- We create a state sequence object with the *mvad* data set

Table: List of Variables in the MVAD data set

id	unique individual identifier
weight	sample weights
male	binary dummy for gender, 1=male
catholic	binary dummy for community, 1=Catholic
Belfast	binary dummies for location of school, one of five Education and Library Board areas in Northern Ireland
N.Eastern	"
Southern	"
S.Eastern	"
Western	"
Grammar	binary dummy indicating type of secondary education, 1=grammar school
funemp	binary dummy indicating father's employment status at time of survey, 1=father unemployed
gcse5eq	binary dummy indicating qualifications gained by the end of compulsory education, 1=5+ GCSEs at grades A-C, or equivalent
fmpr	binary dummy indicating SOC code of father's current or most recent job,1=SOC1 (professional, managerial or related)
livboth	binary dummy indicating living arrangements at time of first sweep of survey (June 1995), 1=living with both parents
jul93	Monthly Activity Variables are coded 1-6, 1=school, 2=FE, 3=employment, 4=training, 5=joblessness, 6=HE
:	"
jun99	"

Locating the sequence data

- We locate the sequence data in the data frame

```
> names(mvad)
```

```
 [1] "id"        "weight"    "male"      "catholic"  "Belfast"
 [6] "N.Eastern" "Southern"  "S.Eastern" "Western"   "Grammar"
[11] "funemp"    "gcse5eq"   "fmpr"      "livboth"   "Jul.93"
[16] "Aug.93"    "Sep.93"    "Oct.93"    "Nov.93"    "Dec.93"
[21] "Jan.94"    "Feb.94"    "Mar.94"    "Apr.94"    "May.94"
[26] "Jun.94"    "Jul.94"    "Aug.94"    "Sep.94"    "Oct.94"
[31] "Nov.94"    "Dec.94"    "Jan.95"    "Feb.95"    "Mar.95"
[36] "Apr.95"    "May.95"    "Jun.95"    "Jul.95"    "Aug.95"
[41] "Sep.95"    "Oct.95"    "Nov.95"    "Dec.95"    "Jan.96"
[46] "Feb.96"    "Mar.96"    "Apr.96"    "May.96"    "Jun.96"
[51] "Jul.96"    "Aug.96"    "Sep.96"    "Oct.96"    "Nov.96"
[56] "Dec.96"    "Jan.97"    "Feb.97"    "Mar.97"    "Apr.97"
[61] "May.97"    "Jun.97"    "Jul.97"    "Aug.97"    "Sep.97"
[66] "Oct.97"    "Nov.97"    "Dec.97"    "Jan.98"    "Feb.98"
[71] "Mar.98"    "Apr.98"    "May.98"    "Jun.98"    "Jul.98"
[76] "Aug.98"    "Sep.98"    "Oct.98"    "Nov.98"    "Dec.98"
[81] "Jan.99"    "Feb.99"    "Mar.99"    "Apr.99"    "May.99"
[86] "Jun.99"    "region"    "sex"       "religion"
```


Overview of the sequences

- The monthly statuses are in variables `Jul.93` to `Jun.99`, that is columns 15 to 86

```
> mvad[1:4, 15:18]
```

```
       Jul.93      Aug.93     Sep.93     Oct.93
1    training    training employment employment
2 joblessness joblessness         FE         FE
3 joblessness joblessness   training   training
4    training    training   training   training
```

- In the mvad data frame, the sequence data are in the STS format, which is the default for the seqdef() function
- If your sequence data is not in the STS format, use the informat argument (or the seqformat() function beforehand to format your data)

Creating the state sequence object

- The list of the states found in the data is returned by the seqstatl() function

```
> seqstatl(mvad[, 15:86])
```

```
[1] "employment"  "FE"          "HE"          "joblessness" "school"
[6] "training"
```

- By default, these labels are used for both the state names and their long labels
- To override these default settings we create a vector of state names ...

```
> mvad.shortlab <- c("EM", "FE", "HE", "JL", "SC", "TR")
```

- ... and a vector for the state labels

```
> mvad.lab <- c("Employment", "Further education", "Higher education",
+       "Joblessness", "School", "Training")
```


Sequence object

- Now we create the mvad.seq sequence object using the seqdef() function

```
> mvad.seq <- seqdef(mvad, 17:86, states = mvad.shortlab, labels = mvad.lab)
```

- We can display the sequences in the 'STS' format (default) ...

```
> mvad.seq[1:4, 1:20]
```

```
  Sequence
1 EM-EM-EM-EM-TR-TR-EM-EM-EM-EM-EM-EM-EM-EM-EM-EM-EM-EM-EM-EM
2 FE-FE-FE-FE-FE-FE-FE-FE-FE-FE-FE-FE-FE-FE-FE-FE-FE-FE-FE-FE
3 TR-TR-TR-TR-TR-TR-TR-TR-TR-TR-TR-TR-TR-TR-TR-TR-TR-TR-TR-TR
4 TR-TR-TR-TR-TR-TR-TR-TR-TR-TR-TR-TR-TR-TR-TR-TR-TR-TR-TR-TR
```

- .. or in the shorter 'SPS' format

```
> print(mvad.seq[1:4, ], format = "SPS")
```

```
    Sequence
[1] (EM,4)-(TR,2)-(EM,64)
[2] (FE,36)-(HE,34)
[3] (TR,24)-(FE,34)-(EM,10)-(JL,2)
[4] (TR,47)-(EM,14)-(JL,9)
```

Summary

- The summary() function returns usefull information about a state sequence object

```
> summary(mvad.seq)
```

```
 [>] sequence object created with TraMineR version 1.6
 [>] 712 sequences in the data set, 490 unique
 [>] min/max sequence length: 70/70
 [>] alphabet (state labels):
     1=EM (Employment)
     2=FE (Further education)
     3=HE (Higher education)
     4=JL (Joblessness)
     5=SC (School)
     6=TR (Training)
 [>] dimensionality of the sequence space: 350
 [>] colors: 1=#7FC97F 2=#BEAED4 3=#FDC086 4=#FFFF99 5=#386CB0 6=#F0027F
```


Retrieving and setting attributes

- One can also retrieve ...

```
> alphabet(mvad.seq)
```

```
[1] "EM" "FE" "HE" "JL" "SC" "TR"
```

```
> stlab(mvad.seq)
```

```
[1] "Employment"        "Further education" "Higher education"
[4] "Joblessness"       "School"            "Training"
```

- ... or set some attributes with dedicated functions

```
> stlab(mvad.seq) <- c("(1)=EMPLOYED", "(2)=FURTHER EDUC.",
+       "(3)=HIGHER EDUC.", "(4)=JOBLESSNESS", "(5)=SCHOOL",
+       "(6)=TRAINING")
> cpal(mvad.seq) <- c("green", "blue", "red", "yellow",
+       "cyan", "grey")
```


Subscripts and attribute inheritance

- Subsets of sequence objects with indexes like matrices or data frames

```
> seq1 <- mvad.seq[1, ]
> print(seq1, format = "SPS")
```

```
   Sequence
[1] (EM,4)-(TR,2)-(EM,64)
```

- Subsets inherits attributes

```
> alphabet(seq1)
```

```
[1] "EM" "FE" "HE" "JL" "SC" "TR"
```

```
> stlab(seq1)
```

```
[1] "Employment"        "Further education" "Higher education"
[4] "Joblessness"       "School"            "Training"
```


Missing values - A

- Missing values in STS sequences occur for example when
  - Sequences do not start at a same date and a calendar time axis is used ;
  - The follow up time is shorter for some individuals than for others yielding sequences that do not end up at the same position ;
  - The observation at some positions is missing due to non-response, yielding internal gaps in the sequences.
- We illustrate this with the ex1 data frame

```
> data(ex1)
> ex1[, 1:13]
```

```
   [P1] [P2] [P3] [P4] [P5] [P6] [P7] [P8] [P9] [P10] [P11] [P12] [P13]
s1 <NA> <NA> <NA>    A    A    A    A    A    A     A     A     A     A
s2    D    D    D    B    B    B    B    B    B     B  <NA>  <NA>  <NA>
s3 <NA>    D    D    D    D    D    D    D    D     D     D  <NA>  <NA>
s4    A    A <NA> <NA>    B    B    B    B    D     D  <NA>  <NA>  <NA>
s5    A <NA>    A    A    A    A <NA>    A    A     A  <NA>  <NA>  <NA>
s6 <NA> <NA> <NA>    C    C    C    C    C    C     C  <NA>  <NA>  <NA>
```

Types of missing values

- Three types of missing values depending on where they appear in the sequence :
  - left-missing-values, that is missing values appearing before the first valid entry in the sequence ;
  - in-between-missing-values or gaps, that is missing values appearing somewhere between the first and last valid entries ;
  - right-missing values, that is missing values appearing after the last valid entry.
- The seqdef() function provides special options for dealing with each type of missing values (left, gaps and right) with possible value
  - NA : each missing value is left as an explicit missing element
  - DEL : missing elements are deleted
  - a state, belonging to the alphabet or not

Missing values - C

- Default values are left=NA, gaps=NA and right="DEL" : the sequence is considered as ending after the last valid state (all missing values encountered after the last (rightmost) valid state are deleted)
- In the sequence object remaining missing values are coded with '*'

```
> seqdef(ex1, 1:13)
```

```
   Sequence
s1 *-*-*-A-A-A-A-A-A-A-A-A
s2 D-D-D-B-B-B-B-B-B-B
s3 *-D-D-D-D-D-D-D-D-D
s4 A-A-*-*-B-B-B-B-D-D
s5 A-*-A-A-A-A-*-A-A-A
s6 *-*-*-C-C-C-C-C-C-C
```


Missing values - D

- By changing to left="DEL", sequences are left aligned

```
> seqdef(ex1, 1:13, left = "DEL")
```

```
   Sequence
s1 A-A-A-A-A-A-A-A-A-A-A-A-A
s2 D-D-D-B-B-B-B-B-B-B
s3 D-D-D-D-D-D-D-D-D
s4 A-A-*-*-B-B-B-B-D-D
s5 A-*-A-A-A-A-*-A-A-A
s6 C-C-C-C-C-C-C
```


Exercise - State sequence objects

Exercise 3.1

1. Load the biofam data set that comes with the TraMineR library (look at the online help to get more information)
2. Create a state sequence object named biofam.seq with variables a15 to a30, using the following state names and labels

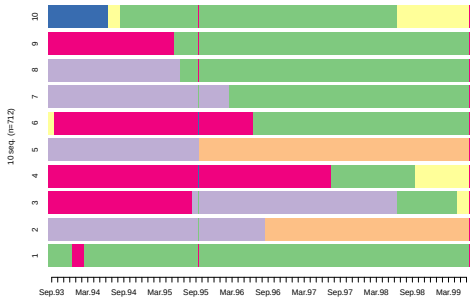
State	*Name*	*Label*
0	P	Parent
1	L	Left
2	M	Married
3	LM	Left/Married
4	C	Child
5	LC	Left/Child
6	LMC	Left/Married/Child
7	D	Divorced

3. Print the first sequences in biofam.seq, in the STS format and then in the SPS format

Sequence index plots (A)

- Sequence index plots [??] allow to visualize individual sequences as horizontally stacked boxes coloured according to the state at the successive positions.
- The seqiplot() function produces such plots (by default the first 10 sequences, use tlim to override)

```
> seqiplot(mvad.seq, border = NA)
```



Sequence index plots (B)

- When the number of displayed sequences is large these plots are often hard to interpret.
- For better results we can sort the sequences according to the values of a covariate. Good choices are for instance the distance to the most frequent sequence or the scores of a multidimensional scaling analysis.
- We plot all the sequences with seqIplot() grouped by qualifications gained of compulsory education (gcse5eq covariate) and sorted by distance to the most frequent sequence.

```
> dist.mostfreq <- seqdist(mvad.seq, method = "LCS", refseq = 0)
> seqIplot(mvad.seq, border = NA, group = mvad$gcse5eq, sortv = dist.mostfreq)
```


Sequence index plots by group

Frequency table

- A first simple overview of a set of state sequences is the sequence frequency table
- It is returned by the seqtab() function (by default the 10 most frequent sequences are displayed, use tlim to override)

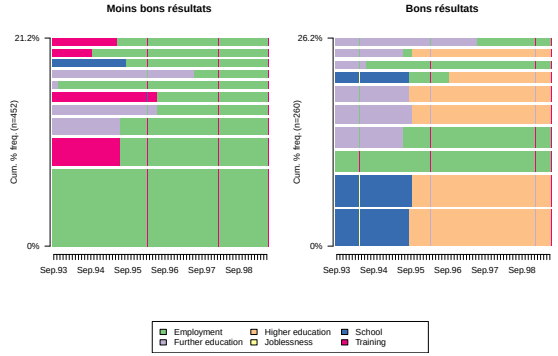
```
> seqtab(mvad.seq)
```

	Freq	Percent
EM/70	50	7.02
TR/22-EM/48	18	2.53
FE/22-EM/48	17	2.39
SC/24-HE/46	16	2.25
SC/25-HE/45	13	1.83
FE/25-HE/45	8	1.12
FE/34-EM/36	7	0.98
FE/46-EM/24	7	0.98
FE/10-EM/60	6	0.84
FE/24-HE/46	6	0.84

Visualizing sequence frequency table

- A graphical view of the sequence frequency table where bar widths are proportional to the frequencies is obtained with the seqfplot() function.

```
> seqfplot(mvad.seq, group = mvad$gcse5eq, border = NA)
```



Exercise - Visualizing sequence data sets

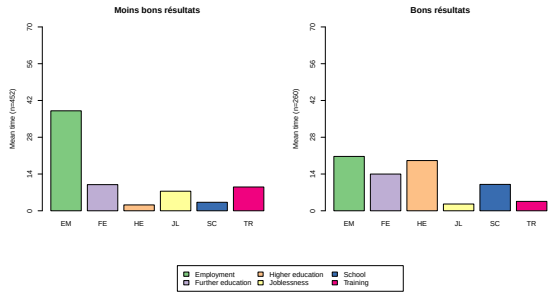
Exercise 3.2

1. Create a full sequence index plot grouped by values of the cohort variable created in exercise 2.1 and sorted by sex
2. Create a sequence frequency plot grouped by values of the cohort variable and save it as a 'jpeg' file

Mean time spent in each state

- A first synthetic information is given by the mean –non necessarily consecutive– time spent in the different states

```
> seqmtplot(mvad.seq, group = mvad$gcse5eq)
```



Transition rates

- The transition rate between each couple of states (s_i, s_j) is the probability to switch to state s_j when we are in state s_i.
- The seqtrate() function returns the transition rates matrix.

```
> mvad.trate <- seqtrate(mvad.seq)
> round(mvad.trate, 2)
```

	[-> EM]	[-> FE]	[-> HE]	[-> JL]	[-> SC]	[-> TR]
[EM ->]	0.99	0.00	0.00	0.01	0.00	0.00
[FE ->]	0.03	0.95	0.01	0.01	0.00	0.00
[HE ->]	0.01	0.00	0.99	0.00	0.00	0.00
[JL ->]	0.04	0.01	0.00	0.94	0.00	0.01
[SC ->]	0.01	0.01	0.02	0.01	0.95	0.00
[TR ->]	0.04	0.00	0.00	0.01	0.00	0.94

- Transition rates provide information about the most frequent state changes observed in the data together with an assessment of the stability of each state.

Transversal statistics

- We can summarize a set of sequences with a series (sequence) of transversal indicators

id	t_1	t_2	t_3	$\cdots$
1	B	B	D	$\cdots$
2	A	B	C	$\cdots$
3	B	B	A	$\cdots$

- These indicators are based on the state distribution at each position/time point in the sequences

Sequence of state distributions

- The seqstatd() function returns the series of state distributions together with other information

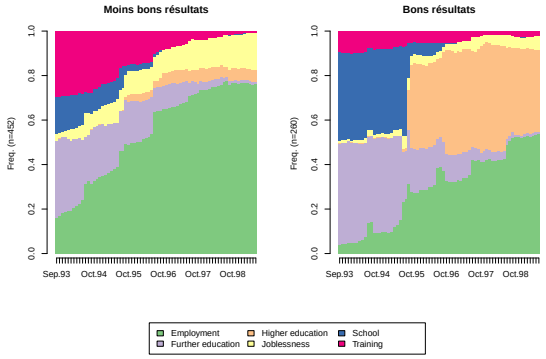
```
> seqstatd(mvad.seq[, 1:8])
```

```
    [State frequencies]
   Sep.93 Oct.93 Nov.93 Dec.93 Jan.94 Feb.94 Mar.94 Apr.94
EM  0.117  0.124  0.133  0.138  0.140  0.140  0.149  0.157
FE  0.386  0.388  0.382  0.381  0.369  0.364  0.361  0.353
HE  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
JL  0.024  0.021  0.020  0.021  0.028  0.038  0.034  0.035
SC  0.251  0.246  0.244  0.242  0.240  0.242  0.240  0.240
TR  0.222  0.222  0.221  0.219  0.222  0.216  0.216  0.215

    [Valid states]
   Sep.93 Oct.93 Nov.93 Dec.93 Jan.94 Feb.94 Mar.94 Apr.94
N     712    712    712    712    712    712    712    712

    [Entropy index]
   Sep.93 Oct.93 Nov.93 Dec.93 Jan.94 Feb.94 Mar.94 Apr.94
H    0.77   0.77   0.78   0.78    0.8    0.8    0.8   0.81
```


State distribution plots

- The series of transversal state distributions is plotted with the seqdplot() function
- We plot the series for the two groups defined by the values of the gcse5eq covariate

```
> seqdplot(mvad.seq, group = mvad$gcse5eq, border = NA)
```



Entropy index

- The Shannon entropy of the state distribution at each position is

$$h(p_1, \ldots, p_s) = -\sum_{i=1}^{s} p_i \log_2(p_i)$$

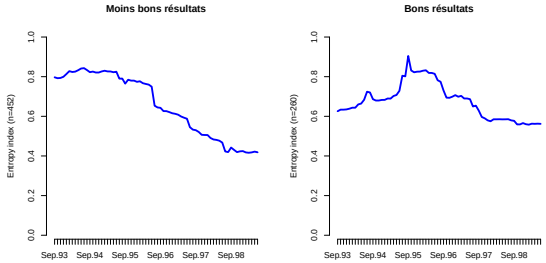
where p_i is the frequency of the ith state and s is the size of the alphabet

- This indicator is called the *entropy index* Billari [2001]
  - It equals 0 when all cases are in the same state (it is thus easy to predict in which state an individual is)
  - It is maximum when the cases are equally distributed between the states of the alphabet (it is thus hard to predict in which state an individual is)

Plotting the series of transversal entropies

- The series of of transversal entropies can be plotted with the seqHtplot() function

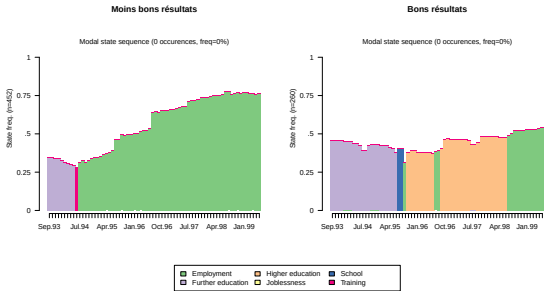
```
> seqHtplot(mvad.seq, group = mvad$gcse5eq)
```



Sequence of modal states

- By taking the most frequent (modal) state at each position we get the modal state sequence
- The modal state sequence is returned by the seqmodst() function and can be plotted with the seqmsplot() function

```
> seqmsplot(mvad.seq, group = mvad$gcse5eq, border = NA)
```



Exercise - Overall and transversal statistics

Exercise 3.3

1. Compute the transition rate matrix for the biofam data set
2. What is the transition rate between states "Left/Maried" and "Left/Maried/Child"?
3. Display the mean times spent in each of the states for each cohort
4. Display the sequence of transversal state distributions for each cohort
5. Display the sequence of modal states for each cohort
6. Within each cohort, at what age is the diversity of the transversal state distribution at its highest ?

Longitudinal features of individual sequences

- We focus now on the characterization and summarization of longitudinal features of individual sequences

id	t_1	t_2	t_3	$\cdots$
1	B	B	D	$\cdots$
2	A	B	C	$\cdots$
3	B	B	A	$\cdots$

- The aim is to define measures that inform on how each sequence is constituted, i.e. on whether it takes a simple or more complex form.
- TraMineR allows to calculate several indicators of sequence complexity
  - "One-dimensional" indicators : number of transitions, number of subsequences, longitudinal entropy
  - Composite complexity measures : Turbulence, Complexity index

Distinct states and durations

- In SPS form a state sequence is represented as an ordered list of distinct successive states (DSS) with their associated durations, i.e. as a sequence of couples (x_j, t_j) where x_j is a state and t_j its duration

```
> print(mvad.seq[1:3, ], format = "SPS")
```

```
    Sequence
[1] (EM,4)-(TR,2)-(EM,64)
[2] (FE,36)-(HE,34)
[3] (TR,24)-(FE,34)-(EM,10)-(JL,2)
```

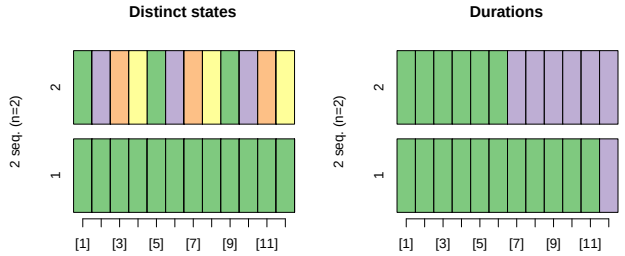
```
> seqdss(mvad.seq[1:3, ])
```

```
  Sequence
1 EM-TR-EM
2 FE-HE
3 TR-FE-EM-JL
```

- This suggests to distinguish characteristics of the state sequencing—the distinct successive states (DSS)—from those of the durations.

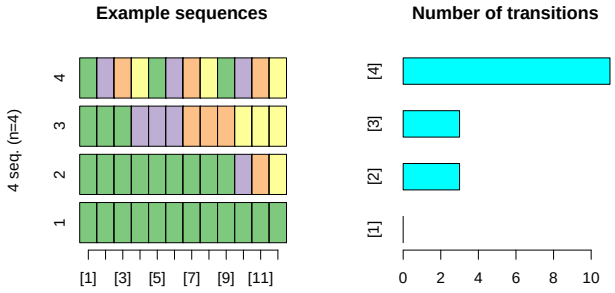
Distinct states and durations

- Complexity measures take into account two main characteristics
  - Distinct state appearing in the sequence : number of transitions, number of distinct subsequences
  - Durations in each of the state : variance of the state durations, entropy of the state distribution
- In the two examples below, sequence 2 is more complex than sequence 1



Number of transitions

- A transition in a state sequence is defined as a change of status between positions (time) t and $t+1$.
- The seqtransn() function returns the number of transitions contained in each sequence of a state sequence object.
- The number of transitions in a sequence x is $\ell_d(x) - 1$ where $\ell_d(x)$ is the length of its DSS sequence



Longitudinal entropy (A)

- Total time spent in each state, that is, when there are multiple spells in a same state the sum of the lengths of these spells.
- In sequence 1 of the *mvad.seq* sequence object, (EM,4)-(TR,2)-(EM,64), there are two spells in state EM with respective durations 4 and 64. Hence, the time spent in EM is 68 months.
- The seqistatd() function returns for each sequence the total time spent in each state of the alphabet (the longitudinal state distribution).

```
> seqistatd(mvad.seq[1:4, ])
```

```
   EM FE HE JL SC TR
1 68  0  0  0  0  2
2  0 36 34  0  0  0
3 10 34  0  2  0 24
4 14  0  0  9  0 47
```

Longitudinal entropy (B)

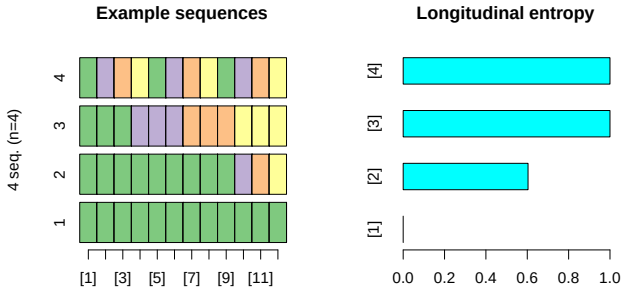
- The entropy of the state distribution within a sequence is a measure of the diversity of its states.
- The seqient() function returns the vector of the longitudinal Shannon entropies of the sequences, i.e. for each sequence the entropy

$$h(\pi_1, \ldots, \pi_a) = -\sum_{i=1}^{a} \pi_i \log \pi_i$$

where a is the size of the alphabet and π_i the proportion of occurrences of the ith state in the considered sequence.

Longitudinal entropy (C)

- The entropy can be interpreted as the 'uncertainty' in predicting the states in a given sequence :
  - when the state remains the same during the whole sequence, the entropy equals 0
  - maximum entropy is reached when the time spent in each element of the alphabet is the same.



Turbulence

- The *Turbulence* $T(x)$ of a sequence x is a composite measure proposed by Elzinga [Elzinga and Liefbroer, 2007] that accounts for
  1. the number $\phi(x)$ of distinct subsequences in the DSS sequence
  2. the variance $s_t^2(x)$ of the consecutive times t_j spent in the distinct states
- The formula is

$$T(x) = \log_2(\phi(x)\,\frac{s^2_{t,max}(x)+1}{s^2_t(x)+1})$$

where
  - $s_t^2(x)$ is the variance of the state-durations t_j $j = 1, \ldots, \ell_d(x)$ for sequence x
  - $s^2_{t,max}(x)$ is the maximum value that this variance can take given the total duration $\ell(x) = \sum_j t_j$ of that sequence.

Turbulence - B

- The turbulence is returned by the seqST() function

```
> mvad.turb <- seqST(mvad.seq)
> hist(mvad.turb)
```

- This measure is not standardized

```
> min(mvad.turb)
[1] 1
> max(mvad.turb)
[1] 12.95858
```

Complexity index

- The Complexity Index is another composite measure proposed by Gabadinho et al. [2010] that combines
  1. the number of transitions in the sequence
  2. the longitudinal entropy.
- The Complexity index of a sequence x is defined as

$$C(x) = \sqrt{\frac{\ell_d(x)}{\ell(x)} \frac{h(x)}{h_{max}}}$$

where

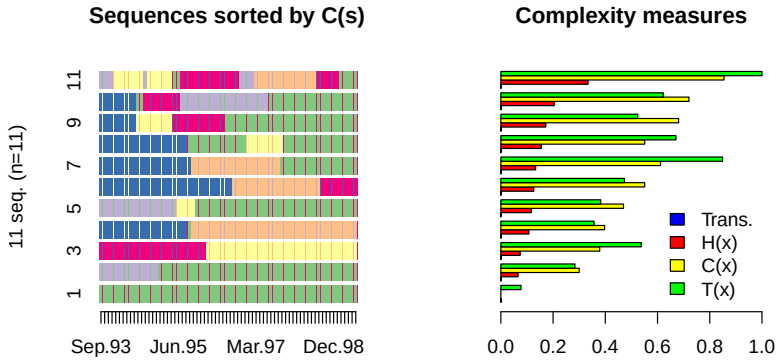
- h_{max} is the theoretical maximum of the entropy, that is $h_{max} = \log a$.
- $\ell_d(x)$ is the length of the Distinct Successive State sequence

Complexity index - Values

- Minimum value of $C(x)$ is 0 and can only be reached by a sequence made of a single distinct state, which contains thus 0 transitions and has an entropy of 0.
- Maximum, 1, of $C(x)$ is reached when the two following conditions are fulfilled :
  1. Each of the state in the alphabet is present in the sequence and the same time ℓ/a is spent in each of them ;
  2. The number of transitions in the sequence is equal to $\ell - 1$, that is, the length ℓ_d of the DSS is equal to the length of the sequence ℓ.

Complexity measures - Summary

- We compare complexity measures for sequences extracted at regular interval in the distribution of $C(s)$



Exercise - Individual sequence characteristics

Exercise 3.4

1. Compute the complexity index and turbulence of the sequences in the biofam data set
2. Display a XY plot of the two indicators
3. Display boxplots of turbulence by cohort. Do the same for the complexity index.
4. What can you say about the trend in the complexity of family life histories ?

References I

Francesco C. Billari. The analysis of early life courses : complex descriptions of the transition to adulthood. *Journal of Population Research*, 18(2) : 119(24)–, November 2001. ISSN 1443-2447.

Cees Elzinga and Aart Liefbroer. De-standardization of family-life trajectories of young adults : A cross-national comparison using sequence analysis. *European Journal of Population/Revue européenne de Démographie*, 23(3) : 225–250, October 2007. URL `http://dx.doi.org/10.1007/s10680-007-9133-7`.

Alexis Gabadinho, Gilbert Ritschard, Matthias Studer, and Nicolas S. Müller. Indice de complexité pour le tri et la comparaison de séquences catégorielles. *Revue des nouvelles technologies de l'information RNTI*, E-19 :61–66, 2010.

Sequence analysis for social scientists

Part IV - Analyzing sequences using dissimilarities

Matthias Studer, Alexis Gabadinho, Gilbert Ritschard, Nicolas S. Müller

Department of Econometrics, University of Geneva
http://mephisto.unige.ch/biomining

Summer School on Advanced Methods for the Analysis of Complex Event History Data, Bristol, 28-29 June 2010

Example data set

- Data from McVicar and Anyadike-Danes [2002]
- 712 individuals
- 14 variables + 72 monthly activity state variables
  1. 1 = school
  2. 2 = Further Education (FE)
  3. 3 = employment
  4. 4 = training
  5. 5 = joblessness
  6. 6 = Higher Education (HE)

Creating the state sequence object

- Loading TraMineR and the `mvad` data set

```
R> library(TraMineR)
R> data(mvad)
```

- We consider only 70 states from Sept 93, i.e. we skip July and August 93 (i.e. summer holiday).

```
R> mvad.lab <- c("employment", "further education", "higher education",
+       "joblessness", "school", "training")
R> mvad.shortlab <- c("EM", "FE", "HE", "JL", "SC", "TR")
R> mvad.seq <- seqdef(mvad, 17:86, states = mvad.shortlab, labels = mvad.lab)
```


Dissimilarities between pairs of sequences

- Distance between sequences
  - Different metrics (LCP, LCS, OM, HAM, DHD, ...)
- With pairwise dissimilarity matrix, we can
  - determine a central sequence (centro-type)
  - measure the discrepancy between sequences
  - build a typology of the sequences using Cluster Analysis
  - represent the sequences using a scatter plot using MDS
  - test and measure the link between one or several covariates and the sequences using discrepancy analysis.

Dissimilarity: Concepts

- A dissimilarity is a quantification of how far two objects are.
- For instance, consider two incomes x and y:
  - $d(x, y) = |x - y|$
  - $d(x, y) = \log(1 + |x - y|)$
  - $d(x, y) = (x - y)^2$
- How to do it with categorical sequences?
- Depending on the issue, we want our dissimilarity measure to account for:
  - Order of the states and transitions in each sequence.
  - Temporality of the transitions.
  - Duration of stay in each state.

Optimal matching (optimal alignment)

Optimal matching

- Based on Levenshtein [1966]'s distance
- Inspired from alignment used in biology (ADN or protein sequences)
- Introduced in social sciences by Abbott and Forrest [1986]
- Also known as Edit distance.

Optimal matching (OM): principle

- OM distance is the minimal cost needed to transform one sequence into the other one.
- Want to transform one sequence into the other one.
- Using two types of operations
  - Insertion or deletion of an element
  - Substitution of an element
- Each operation has a cost.

indel and substitution costs

indel and substitution costs

- indel (insertion-deletion) costs:
  - Same cost for each 'insertion' or 'deletion'.
  - indel cost is a single constant.
- Substitution costs:
  - Each substitution may receive a different cost.
  - Matrix of substitution costs.
  - However: symmetrical cost $c_{i,j} = c_{j,i}$

OM : insertion deletion example.

Consider the two sequences :

1	SC	SC	SC	EM	EM	EM	JL
2	SC	SC	SC	EM	EM	JL	JL

Using indel cost of 1 and constant substitution cost of 2.
Insertion of element 'EM', cost 1.

1	SC	SC	SC	EM	EM	EM	JL	
2	SC	SC	SC	EM	EM	EM	JL	JL

Deletion of element 'JL', cost 1.

1	SC	SC	SC	EM	EM	EM	JL
2	SC	SC	SC	EM	EM	EM	JL

The two sequences are now identical, total cost: 2.

OM: substitution example

Consider the 2 sequences

1	SC	SC	SC	EM	EM	EM	JL
2	SC	SC	SC	EM	EM	JL	JL

Substitution of 'JL' by element 'EM', cost 2.

1	SC	SC	SC	EM	EM	EM	JL
2	SC	SC	SC	EM	EM	EM	JL

Defining substitution costs

- Unique cost $c_{ij} = c$ (user defined).
  - Using $c = 2$ and $indel = 1$ OM is equivalent to "LCS" metric.
  - Let LLCS be the length of the longest common subsequence and $|x|$ the length of the sequence x
  - $d_{LCS}(x, y) = |x| + |y| - 2 \cdot LLCS(x, y)$
- Based on transition rates (no additional input required)
  - $c_{i,j} = c_{j,i} = 2 - p(i_t \mid j_{t-1}) - p(j_t \mid i_{t-1})$
- Custom costs

OM: complete exemple

Consider the distance between these two sequences:

I	N	D	U	S	T	R	Y
I	N	T	E	R	E	S	T

With a constant substitution cost of 2 and indel of 1, the optimal matching distance is 8 with the following operations.

1	I	N	D	-	-	U	S	T	R	Y
2	I	N	T	E	R	E	S	T	-	-
cost	0	0	2	3	4	6	6	6	7	8

The longest common subsequence is `INST` and its length is 4.

Using Optimal Matching in TraMineR

- Create the state sequence object with `seqdef()`
- Compute matrix of OM distances with `seqdist(..., method="OM", indel=..., sm=...)`
- Specify the substitution costs using
  - `sm="CONSTANT"` for constant substitution costs.
  - `sm="TRATE"` for substitution costs based on transition rates.
  - or a custom substitution cost matrix using the same `sm` argument.

Computing the distances

- Using the constant substitution cost, we compute distances

```
R> mvad.dist <- seqdist(mvad.seq, method = "OM", indel = 1, sm = "CONSTANT")
R> mvad.dist[1:10, 1:10]
```

```
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]    0  140  116  108  140   64   60   44   38    48
 [2,]  140    0   72  140   22  140   80   96  140   140
 [3,]  116   72    0   68   90   72   60   76   78   116
 [4,]  108  140   68    0  140   46  112  112   70    94
 [5,]  140   22   90  140    0  140   90   96  140   140
 [6,]   64  140   72   46  140    0   68   68   26    66
 [7,]   60   80   60  112   90   68    0   16   60    60
 [8,]   44   96   76  112   96   68   16    0   44    48
 [9,]   38  140   78   70  140   26   60   44    0    48
[10,]   48  140  116   94  140   66   60   48   48     0
```


Cost Matrix: Custom Costs

```
R> subm.custom <- matrix(c(0, 1, 1, 2, 1, 1, 1, 0, 1, 2,
+       1, 2, 1, 1, 0, 3, 1, 2, 2, 2, 3, 0, 3, 1, 1, 1, 1,
+       3, 0, 2, 1, 2, 2, 1, 2, 0), nrow = 6, ncol = 6, byrow = TRUE,
+       dimnames = list(mvad.shortlab, mvad.shortlab))
R> subm.custom
R> mvad.dist.custom <- seqdist(mvad.seq, method = "OM",
+       indel = 1.5, sm = subm.custom)
```


Dissimilarity measures

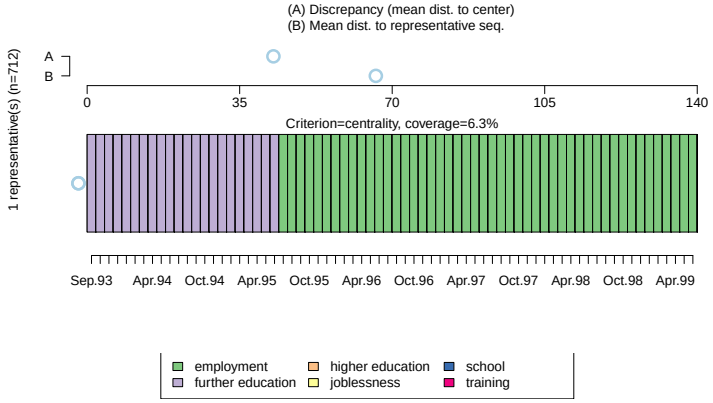
Other dissimilarity measures provided by TraMineR:

- Longest Common Prefix (LCP)
- Longest Common Suffix (RLCP)
- Longest Common Subsequence (LCS)
- Hamming distance (HAM)
- Dynamic Hamming Distance (DHD) [Lesnard, 2010]

Centrotype (medoid)

- Sequence with minimal sum of distances to other sequences.

```
R> seqrplot(mvad.seq, dist.matrix = mvad.dist, criterion = "dist",
+       nrep = 1)
```



Set of representative sequences

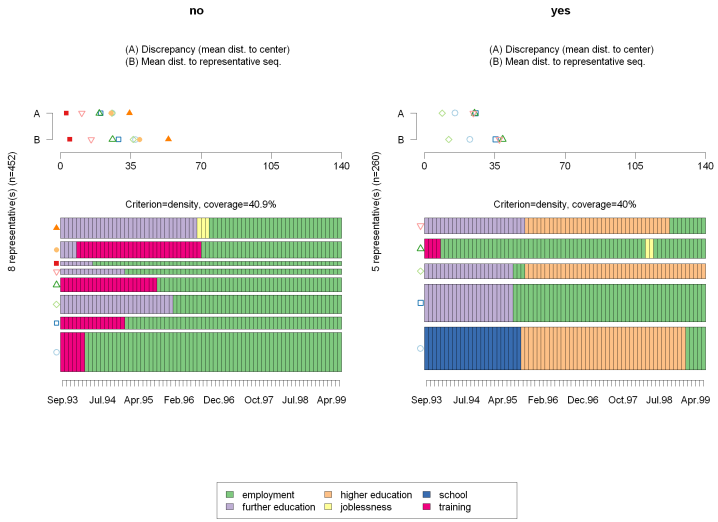
- Aim: Find a minimal set of representative sequences such that
  - they are non redundant (none in the neighborhood of another representative)
  - Coverage: minimum percentage of all sequences that are in the neighborhood of the representative sequences
- Representatives are found with (`tsim`: neighborhood diameter, `trep`: coverage)

```
R> seqrep(mvad.seq, dist.matrix = mvad.dist, criterion = "density",
+       trep = 0.4, tsim = 0.1)
```

- Representatives are plotted with

```
R> seqrplot(mvad.seq, dist.matrix = mvad.dist, group = mvad$gcse5eq,
+       cex.plot = 1, trep = 0.4)
```


Plot of representative sequences



Cluster analysis

- Cluster analysis automatically classify different objects in a reduced number of categories.
- It simplifies the large number of distinct sequences in a few different types of trajectories.
- It is used to build a typology of the trajectories.
- It offers a descriptive approach to analyze the sequences.

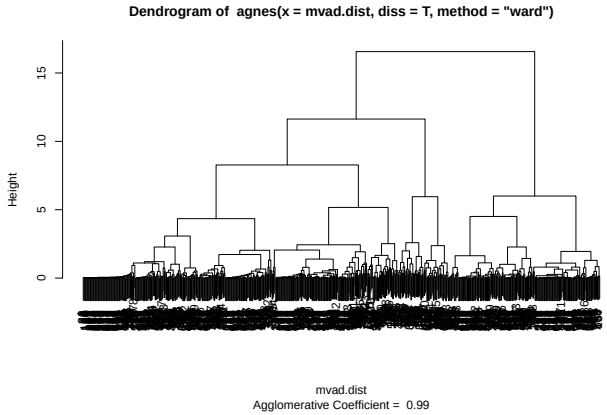
Cluster analysis

- Clustering may be done using a dissimilarity matrix.
- There are several possibilities in R, for instance with the `cluster` library
  - `agnes()`: agglomerative nesting, i.e. hierarchical clustering (average, ward, ...).
  - `diana()`: divisive analysis.
  - `pam()`: partitioning around medoids.

Hierarchical clustering (Ward)

- Ward is a hierarchical clustering algorithm.
- At each step, it joins together the two less distant groups.
- Ward aims at minimizing the within cluster discrepancy.

```
R> library(cluster)
R> mvad.clusterward <- agnes(mvad.dist, diss = T, method = "ward")
R> plot(mvad.clusterward, ask = F, which.plots = 2)
```



Retrieving cluster membership

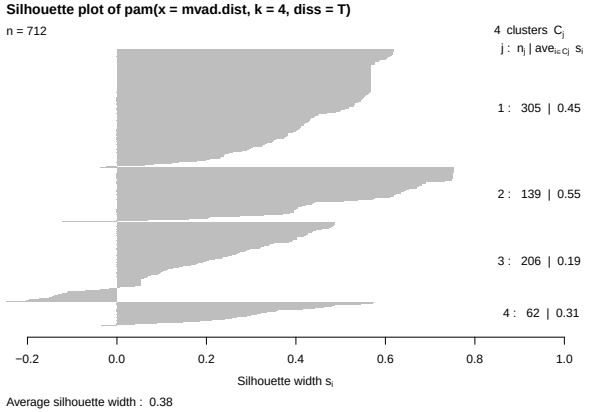
- Select the number of clusters,
- Cut the tree at chosen level, and store cluster membership into a vector.

```
R> mvad.cl4 <- cutree(mvad.clusterward, k = 4)
R> mvad.cl4[1:10]
 [1] 1 2 3 3 2 3 1 1 3 4
R> clust.labels <- c("Employment", "Higher Education", "Training",
+       "Joblessness")
R> mvad.cl4.factor <- factor(mvad.cl4, levels = 1:4, labels = clust.labels)
```


PAM clustering

- Partitioning Around Medoids.
- Non hierarchical, number of cluster must be set a priori.
- Faster, but results may depend on the starting point.

```
R> mvad.pam4 <- pam(mvad.dist, k = 4, diss = T)
R> plot(mvad.pam4)
```



Retrieving cluster membership

- With PAM, the cluster membership is stored on the `clustering` element.
- We may retrieve it using:

```
R> mvad.pam4$clustering[1:10]
 [1] 1 2 3 4 2 1 3 3 1 1
```


Warning!!!

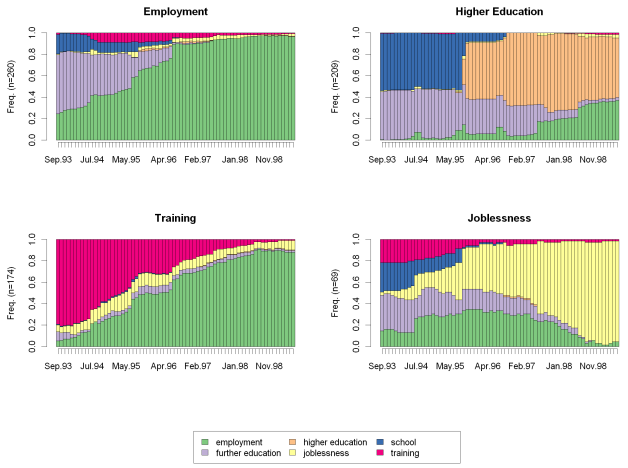
- Do not forget to specify the `diss = T` option.
- Otherwise (i.e. by default) functions `agnes()`, `diana()`, `pam()`, ... compute the Euclidean distance matrix between rows of the dissimilarity matrix.

Exploring clusters graphically

- Three types of graphics
  1. Transversal distribution with `seqdplot()`
  2. Frequency plots with `seqfplot()`
  3. Individual index-plots `seqiplot()`
  4. Representative sequences `seqrplot()`
- Required argument: state sequence object.
- Use `group = cluster.membership.factor` to get plots by cluster.

Transversal Distributions

```
R> seqdplot(mvad.seq, group = mvad.cl4.factor)
```



Logistic Regression

- It allow to estimate the influence of independent covariates on the probability of belonging to a given cluster (i.e. type of trajectory) rather than another.
- We fit a logistic regression model using the `glm` command.

```
R> jobless <- mvad.cl4 == 4

R> jobless.reglog <- glm(jobless ~ male + funemp + gcse5eq, family = binomial(link = logi
+       data = mvad)
```



```
R> summary(jobless.reglog)

Call:
glm(formula = jobless ~ male + funemp + gcse5eq, family = binomial(link = logit),
    data = mvad)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.8974  -0.5581  -0.3852  -0.2897   2.8116

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -1.7806     0.2096  -8.497  < 2e-16 ***
maleyes      -0.7830     0.2694  -2.906 0.003659 **
funempyes     1.0790     0.2817   3.830 0.000128 ***
gcse5eqyes   -1.3695     0.3605  -3.799 0.000146 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 453.17  on 711  degrees of freedom
Residual deviance: 412.16  on 708  degrees of freedom
AIC: 420.16

Number of Fisher Scoring iterations: 6
```


Exercises

Exercise 4.1

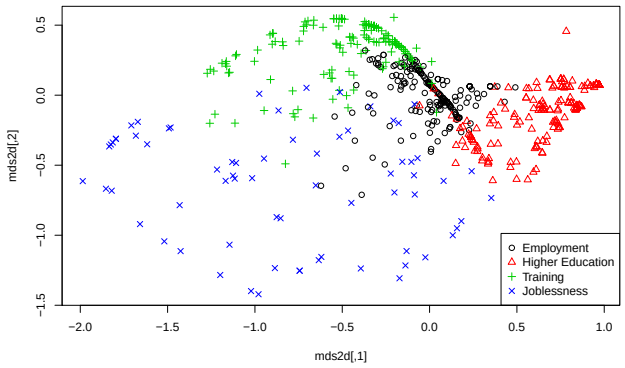
1. Use the `biofam` data set.
2. Compute the OM distance matrix with substitution costs set according to transition rates.
3. Cluster the sequences in 3 groups (using either "Ward" or "PAM").
4. Explore your clustering solution graphically using representative sequences.
5. Name and interpret your clusters.
6. Fit a logistic regression model for one of your cluster using the cohort, language (plingu02) and sex covariates.

Scatterplot (MDS)

- Multidimensional Scaling (MDS) seeks numerical factors such that the Euclidean distance in the factor space reproduces at best the dissimilarity matrix.
- Through MDS, we get a scatter plot representation of sequences.

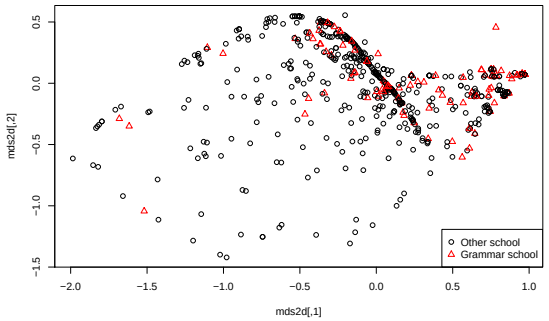
```
R> mds2d <- cmdscale(mvad.dist, k = 2)

R> plot(mds2d, pch = mvad.cl4, col = mvad.cl4)
R> legend("bottomright", col = 1:4, pch = 1:4, legend = clust.labels)
```

Sequence scatterplot colored by cluster



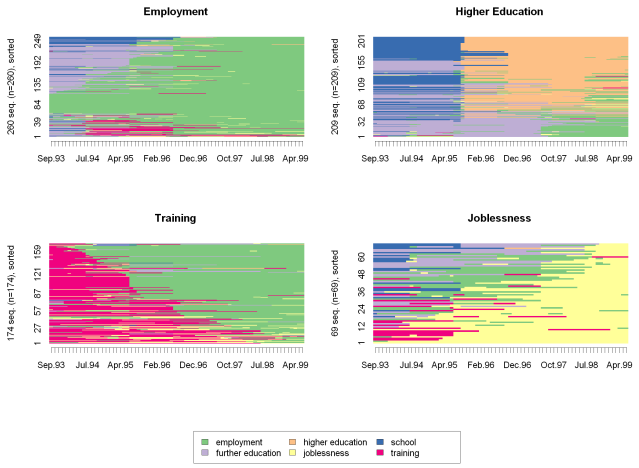
Code for scatterplot colored by type of school

```
R> plot(mds2d, pch = as.integer(mvad$Grammar), col = mvad$Grammar)
R> legend("bottomright", col = 1:2, pch = 1:2, legend = c("Other school",
+       "Grammar school"))
```



Sort: First factor of MDS analysis

```
R> seqIplot(mvad.seq, group = mvad.cl4.factor, sortv = mds2d[,
+       1])
```



Sequence discrepancy analysis

- How to measure the link between explanatory factors and sequences?
- Using cluster and logistic regression is problematic:
  - Clusters are not homogenous.
  - Clustering choice are made on a statistical ground.
  - These choices may artificially hide or create an association.
- General principle of sequence discrepancy analysis.
  - Definition of a measure of discrepancy of a set of sequence using any dissimilarity measure.
  - Compute the share of this discrepancy accounted by a given explanatory factor.
  - The method is based on a generalization of ANOVA [Studer et al., 2010].
  - It may be extended to the multi-factor case and regression trees.

Discrepancy of the set of sequences: Definition

- In the Euclidean case, the sum of squares SS can be expressed in terms of distances between pairs

$$SS = \sum_{i=1}^{n}(y_i - \bar{y})^2 = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=i+1}^{n}(y_i - y_j)^2$$
$$= \frac{1}{n}\sum_{i=1}^{n}\sum_{j=i+1}^{n} d_{ij}$$

- Setting d_{ij} equal to OM, LCP, LCS ... distance, we get a measure of dispersion.

Discrepancy of the set of sequences: Interpretation

- Interpretation of the discrepancy.
  - The discrepancy measures the between individual variability of the trajectories.
  - It may reflect a form of precariousness
  - r a multiplicity of choices faced by the individuals.
  - It is different from the within individual longitudinal state diversity measures

Computing the sequence discrepancy

Computing the sequence discrepancy

```
R> dissvar(mvad.dist)
```

```
[1] 42.74502
```


Analysis of sequence discrepancy

- ANOVA like analysis based on pairwise dissimilarities
- We decompose the SS (Sum of squares equivalent)

$$SS_T = SS_B + SS_W$$

- Here, with the formula shown earlier

$$SS_T = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=i+1}^{n} d_{ij}$$
$$SS_W = \sum_{g}\left(\frac{1}{n_g}\sum_{i=1}^{n_g}\sum_{j=i+1}^{n_g} d_{ij,g}\right)$$
$$SS_B = SS_T - SS_W$$

Pseudo R-square and ANOVA Table

- ANOVA table for m groups

	Discrepancy	df	Mean Discr.	F
Between	SS_B	$df_B = m - 1$	$\frac{SS_B}{df_B}$	$\frac{SS_B}{SS_W}\frac{df_W}{df_B}$
Within	SS_W	$df_W = \sum_g n_g - m$	$\frac{SS_W}{df_W}$	
Total	SS_T	$df_T = n - 1$		

- Pseudo R^2

$$R^2 = \frac{SS_B}{SS_T}$$

- Share of sequence discrepancy explained by the grouping variable.

Significance

- Pseudo F

$$F = \frac{SS_B/(m-1)}{SS_W/(n-m)}$$

- Normality is not defendable in this setting.
- F cannot be compared with an F distribution.
- The significance is assessed through a permutation test

Permutation test

- Estimate the empirical distribution of F under independence.
- Compute F_{perm} the F value associated with a random permutation of the profil.
- Permutation: randomly reassign each covariate profile to one of the observed sequence
- Repeat this step R times.
- The p-value associated with the test is $p(F_{obs} > F_{perm})$.
- The confidence interval of the p-value is given by $p \pm 1.96\sqrt{p(1-p)/R}$
- It is generally admitted that 5000 permutations should be used to assess a significance threshold of 1% and 1000 for a threshold of 5% [Manly, 2007].

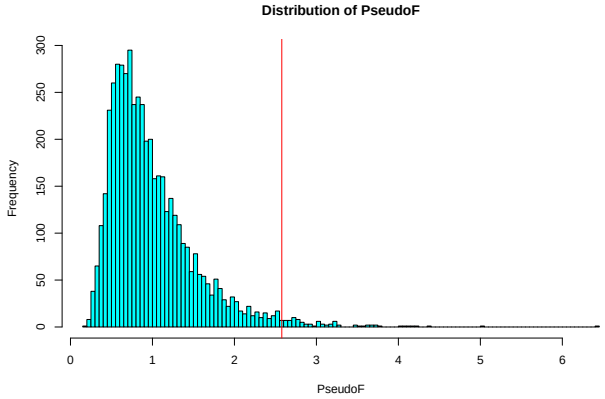
Analysis of sequence discrepancy

- Running an ANOVA like analysis for `livboth`

```
R> da <- dissassoc(mvad.dist, group = mvad$livboth, R = 5000)

R> print(da)

Pseudo ANOVA table:
              SS  df       MSE
Exp     110.0993   1 110.09927
Res   30324.3558 710  42.71036
Total 30434.4551 711  42.80514

Test values  (p-values based on 4999 permutations):
  PseudoF    PseudoR2 PseudoF_Pval  PseudoT PseudoT_Pval
 2.577812 0.003617586       0.0176 0.333323        0.023

Variance per level:
        n variance
no    261 40.88325
yes   451 43.57833
Total 712 42.74502
```

Empirical distribution of F under independence

```
R> hist(da, col = "cyan")
```



Homogeneity of the discrepancy

Homogeneity of the discrepancy

- Test of the difference of within group discrepancy.
- Based on a generalization of the Bartlett Test.
- Significance assessed through permutation tests.

Differences over time

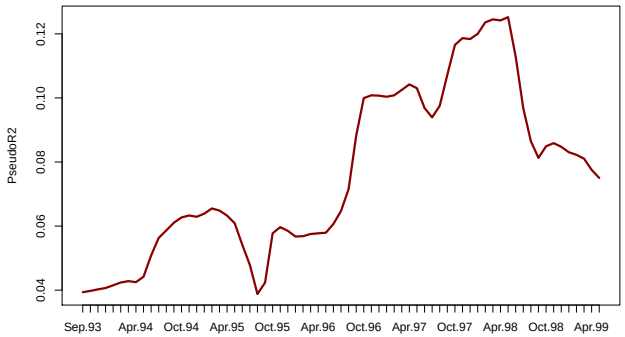
- How do differences between groups vary over time?
- At which age do trajectories most differ across birth cohorts?
- Compute R^2 for short sliding windows (length 6)
- We get thus a sequence of R^2, which can be plotted
- Similarly, we can plot series of
  - total within (residual) discrepancy (SS_W)
  - within discrepancy of each group (SS_G)

Differences over time

```
R> mvad.diff <- seqdiff(mvad.seq, group = mvad$gcse5eq)

R> mvad.diff$stat[1:4, ]

        PseudoF   PseudoR2  PseudoT
Sep.93 29.09196 0.03936176 2.313692
Oct.93 29.39664 0.03975760 2.223468
Nov.93 29.76849 0.04024027 2.265784
Dec.93 30.09793 0.04066750 2.304112

R> mvad.diff$variance[1:4, ]

             no       yes     Total
Sep.93 0.3688107 0.3113979 0.3620982
Oct.93 0.3691362 0.3127219 0.3629661
Nov.93 0.3704210 0.3133136 0.3642237
Dec.93 0.3725771 0.3146893 0.3663363
```

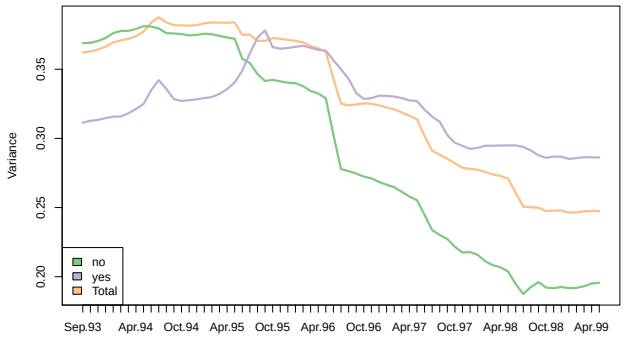
Plotting R-squares over time

```
R> plot(mvad.diff, lwd = 3, col = "darkred")
```



Plotting within discrepancies over time

```
R> plot(mvad.diff, lwd = 3, stat = "Variance", legendposition = "bottomleft")
```



Multiple factor analysis

- Generalize previous approach for multiple covariates.
- Here, we consider Type III effects
- Measure the additional contribution of each covariate v when we accounted for all other covariates.
- The F statistics reads

$$F_v = \frac{(SS_{B_c} - SS_{B_v})/p}{SS_{W_c}/(n-m-1)}$$

where the SS_{B_c} and SS_{W_c} are the explained and residual sums of squares of the full model, SS_{B_v} the explained sum of squares of the model after removing variable v, and p the number of indicators or contrasts used to encode the covariate v.

- Significance is assessed again through permutation tests.

Running a Multiple factor analysis

```
R> da.mfac <- dissmfac(mvad.dist ~ male + Grammar + funemp +
+       gcse5eq + fmpr + livboth, data = mvad, R = 1000)

R> print(da.mfac)

  Variable   PseudoF    PseudoR2 p_value
1     male  3.274802 0.003840223   0.018
2  Grammar 21.124081 0.024771330   0.000
3   funemp  4.483016 0.005257046   0.004
4 gcse5eq 75.725976 0.088800698   0.000
5     fmpr  2.715988 0.003184926   0.048
6 livboth  2.314571 0.002714201   0.058
7    Total 24.829102 0.174448528   0.000
```

Tree structured discrepancy analysis

- Objective: Find the most important predictors and their interactions.
- Iteratively segment the cases using values of covariates (predictors)
- Such that groups be as homogenous as possible.

- At each step, we select the covariate and split with highest R^2.
- Significance of split is assessed through a permutation F test.
- Growing stops when the selected split is not significant.

Growing the tree

```
R> dt <- disstree(mvad.dist ~ male + Grammar + funemp + gcse5eq +
+       fmpr + livboth, data = mvad, R = 5000)

R> print(dt)

Dissimilarity tree
Global R2: 0.113
       |-- Root [ 712 ] var: 42.7
         |-> gcse5eq  R2: 0.0821
              |-- no [ 452 ] var: 37.5
                |-> funemp  R2: 0.0107
                     |-- no [ 362 ] var: 35.9
                       |-> male  R2: 0.0123
                            |-- no [ 146 ] var: 38.7
                            |-- yes [ 216 ] var: 33.3
                     |-- yes [ 90 ] var: 41.8
              |-- yes [ 260 ] var: 42.3
                |-> Grammar  R2: 0.0534
                     |-- no [ 183 ] var: 42.2
                     |-- yes [ 77 ] var: 34.9
```


Creating a Graphviz plot of the tree

- Using simplified interface to generate a .dot file for GraphViz

```
R> seqtree2dot(dt, "fg_mvadseqtree", seqdata = mvad.seq,
+       type = "d", border = NA, withlegend = FALSE,
+       axes = FALSE, ylab = "", yaxis = FALSE)
```

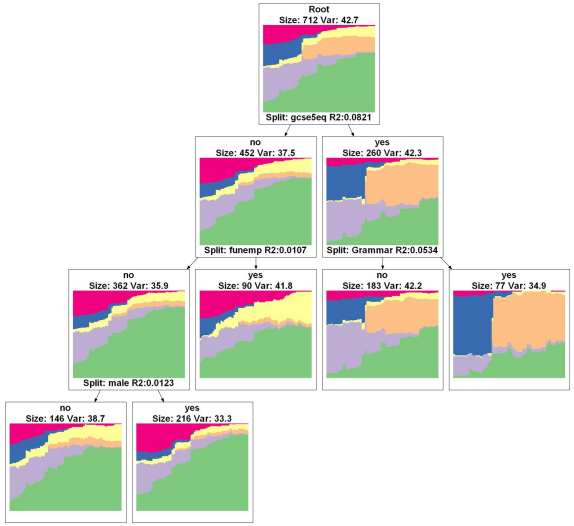
- The file may be converted to a .svg graphical file. using

```
R> shell("dot -Tsvg -ofg_mvadseqtree.svg fg_mvadseqtree.dot")
```

- The file may be converted to a .jpg image using ImageMagick

```
R> shell("convert fg_mvadseqtree.svg fg_mvadseqtree.jpg")
```


Graphical Tree



Exercises

Exercise 4.2

1. Use the previous distance matrix.
2. Compute the association with the `cohort` covariate using `dissassoc`.
3. Interpret the differences graphically using Index-plot with all sequences sorted according to the first dimension of an MDS.
4. Explore the evolution of the association using seqdiff.
5. Fit a regression tree and plot the results.

References I

Andrew Abbott and John Forrest. Optimal matching methods for historical sequences. *Journal of Interdisciplinary History*, 16:471–494, 1986.

Laurent Lesnard. Setting cost in optimal matching to uncover contemporaneous socio-temporal patterns. *Sociological Methods Research*, 38:389–419, 2010.

V. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10:707–710, 1966.

Bryan F. J. Manly. *Randomization, Bootstrap and Monte Carlo Methods in Biology*. Chapman & Hall, third edition edition, 2007.

D. McVicar and M. Anyadike-Danes. Predicting successful and unsuccessful transitions from school to work using sequence methods. *Journal of the Royal Statistical Society A*, 165(2):317–334, 2002.

Matthias Studer, Gilbert Ritschard, Alexis Gabadinho, and Nicolas S. Müller. Discrepancy analysis of complex objects using dissimilarities. In Fabrice Guillet, Gilbert Ritschard, Henri Briand, and Djamel A. Zighed, editors, *Advances in Knowledge Discovery and Management*, Studies in Computational Intelligence. Springer, Berlin, 2010. (forthcoming).

Sequence analysis for social scientists

Part V - Analysis of event sequences

Matthias Studer, Alexis Gabadinho, Gilbert Ritschard, Nicolas S. Müller

Department of Econometrics, University of Geneva
http://mephisto.unige.ch/biomining

Summer School on Advanced Methods for the Analysis of Complex Event History Data, Bristol, 28-29 June 2010

Example data set

- Data from McVicar and Anyadike-Danes [2002]
- 712 individuals
- 14 variables + 72 monthly activity state variables
  1. 1 = school
  2. 2 = Further Education (FE)
  3. 3 = employment
  4. 4 = training
  5. 5 = joblessness
  6. 6 = Higher Education (HE)

Creating the state sequence object

- Loading TraMineR and the `mvad` data set

```
R> library(TraMineR)
R> data(mvad)
```

- We consider only 70 states from Sept 93, i.e. we skip July and August 93 (i.e. summer holiday).

```
R> mvad.lab <- c("employment", "further education", "higher education",
+       "joblessness", "school", "training")
R> mvad.shortlab <- c("EM", "FE", "HE", "JL", "SC", "TR")
R> mvad.seq <- seqdef(mvad, 17:86, states = mvad.shortlab, labels = mvad.lab)
```


Analysis of event sequences
Objective

- Focus on events, rather than states.
- Interest in the patterns of events.
  - Pattern of events: events that occur systematically together and in the same order
- Are there typical "patterns" of events?
- Relationship with covariates
  - Which patterns best discriminate specific groups?
  - Typical differences in event sequences between men and women.
- Event sequences analysis may be used for searching typical state sequencing.
- Association rules between event subsequences:
  - Sequence *Leaving home* → *Childbirth* generally followed by *Marriage* → *Second Childbirth*

Events and transitions

- Event sequence: time ordered transitions.
- Transition: set of non-ordered events.

Example

(LHome, Union) → (Marriage) → (Childbirth)

- (LHome, Union) and (Marriage) are transitions.
- "LHome", "Union" and "Marriage" are events.

subsequence

- A subsequence B of a sequence A is an event sequence such that
  - each event of B is an event of A.
  - the events occur in B in the same order as in A.

Example

A (LHome, Union) → (Marriage) → (Childbirth).

B (LHome, Marriage) → (Childbirth).

C (LHome) → (Childbirth).

- C is a subsequence of A and B, since order of events is respected.
- B is not a subsequence of A, since we do not know if "LHome" occurs before "Marriage".

Frequent and discriminant subsequences

- Support of a subsequence: number of sequences that contain the subsequence.
  - A frequent subsequence is a sequence with support greater than a minimal support.
  - A subsequence is discriminant between groups if its support varies significantly across groups.

Data Format

- For doing event sequence analysis in TraMineR, we need an event sequence object
- We create it with `seqecreate()`
- to which we provide data in either of the following form:
  - Time Stamped Event (TSE), which permits to directly specify the events.
  - A state sequence object together with a choice for automatic conversion
    - **transition** A distinct event for the transition between each pair of states.
    - **state** An distinct event for the start of a spell in a given state.
    - **period** An event for the start and an other for the end of a spell in a given state.

"Time Stamped Event" (TSE)

- **id** Individual identifier.
- **timestamp** Time stamp (real valued) of the event.
- **event** The code (string) of the event.
- One line per event.

```
R> data(actcal.tse)
R> head(actcal.tse)
```

```
  id time      event
1  1    0   PartTime
2  2    0 NoActivity
3  2    4      Start
4  2    4   FullTime
5  2   11       Stop
6  3    0   PartTime
```


Creating an event sequence object

From a state sequence object

- Function `seqecreate()`.
- Argument `tevent` sets the choice for automatic conversion.
- Here, we assign an event to the start of each spell spent in a given "state".
- Look for state patterns.
- Other solutions are available.

```
R> data(mvad)
R> mvad.shortlab <- c("EM", "FE", "HE", "JL", "SC", "TR")
R> mvad.seq <- seqdef(mvad[, 17:86], labels = mvad.shortlab)
R> mvad.seqe <- seqecreate(mvad.seq, tevent = "state")
```


Creating an event sequence object

Using the TSE format

- Function `seqecreate()`.
- With arguments `id`, `timestamp` and `event` we provide the data in the TSE format.

```
R> actcal.seqe <- seqecreate(id = actcal.tse$id,
+       timestamp = actcal.tse$time, event = actcal.tse$event)
```


Event sequence representation

- Each sequence is displayed in the following form `(e1,e2,...)-time-(e2,...)-time`
- where `(e1,e2,...)` is the transition defined by the simultaneous occurrences of events `e1,e2,...`.
- `time` is the time (numerical value) between two transitions (or to the end of the observation time)

```
R> print(mvad.seqe[2])
```

```
[1] (FE)-36.00-(HE)-34.00
```

Finding most frequent subsequences

Function `seqefsub()`, to which we must provide

- The event sequences (an event sequence object)
- The minimal support (with argument `pMinSupport`).

```
R> mvad.fsubseq <- seqefsub(mvad.seqe, pMinSupport = 0.05)
R> mvad.fsubseq[1:5]
```

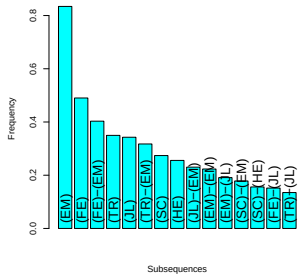
```
  Subsequence   Support Count
1        (EM) 0.8342697   594
2        (FE) 0.4901685   349
3   (FE)-(EM) 0.4030899   287
4        (TR) 0.3497191   249
5        (JL) 0.3426966   244

Computed on 712 event sequences
  Constraint           Value
 countMethod One by sequence
```


Graphical display of most frequent subsequences

We can just apply `plot()` on the object returned by `seqefsub()`

- Use indexes (`[1:15]`) for selecting the subsequences to include (subsequences are sorted by decreasing frequencies).
- Other arguments are passed to the function `barplot()`

```
R> plot(mvad.fsubseq[1:15], col = "cyan", ylab = "Frequency",
+       xlab = "Subsequences", cex = 1.5)
```



Finding most discriminant subsequences

- Aim is to identify the frequent sequences that are most strongly related with a given factor.
- Discriminant power is evaluated with *p*-value of a Chi-square independence test.

- Function `seqecmpgroup()`
- To which we provide the frequent subsequence object and a group factor (gcse5eq).
- A Bonferroni correction is applied when passing argument `method="bonferroni"`.

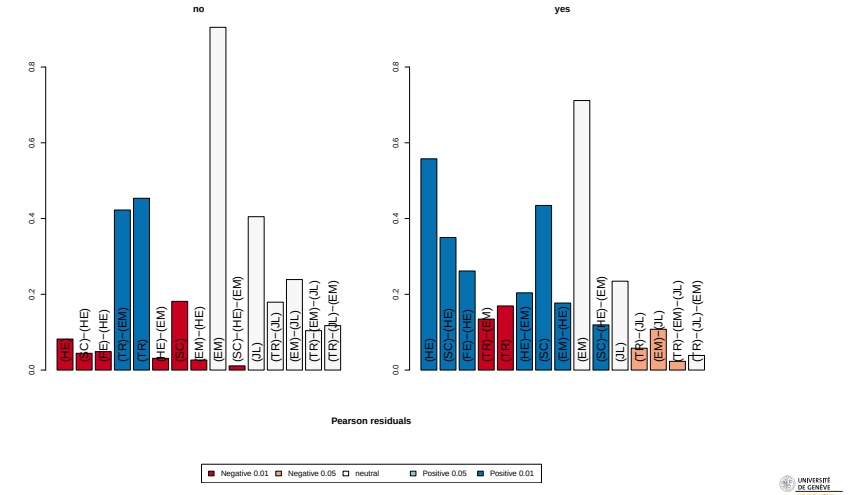
Seeking discriminant subsequences

```
R> mvad.discr <- seqecmpgroup(mvad.fsubseq, group = mvad$gcse5eq,
+       method = "bonferroni")
R> mvad.discr[1:5]
```

```
  Subsequence   Support      p.value statistic index    Freq.no  Freq.yes
1        (HE) 0.2556180 0.000000e+00 193.91346     8 0.08185841 0.5576923
2   (SC)-(HE) 0.1558989 0.000000e+00 114.94365    13 0.04424779 0.3500000
3   (FE)-(HE) 0.1264045 1.865175e-14  65.81448    16 0.04867257 0.2615385
4   (TR)-(EM) 0.3174157 1.538769e-13  61.84409     6 0.42256637 0.1346154
5        (TR) 0.3497191 1.478151e-12  57.42334     4 0.45353982 0.1692308
   Resid.no Resid.yes
1 -7.306716  9.633959
2 -6.011885  7.926715
3 -4.648228  6.128722
4  3.967957 -5.231780
5  3.732446 -4.921258

Computed on 712 event sequences
  Constraint           Value
 countMethod One by sequence
```

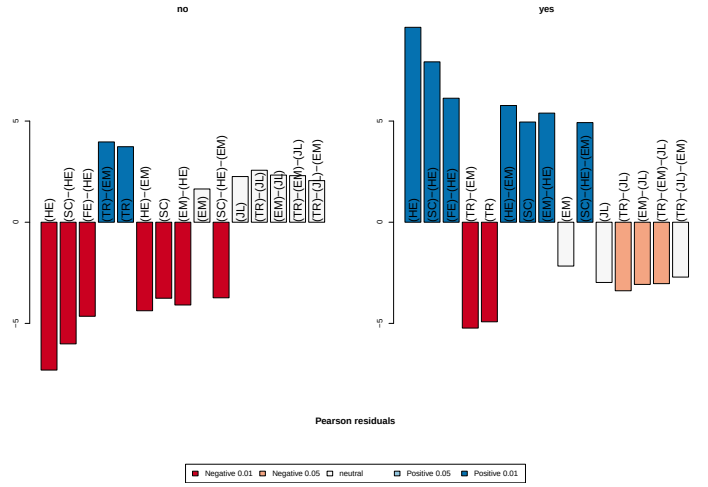
Graphical display, frequencies

```
R> plot(mvad.discr[1:15], cex = 1.5)
```



Graphical display, residuals

```
R> plot(mvad.discr[1:15], ptype = "resid", cex = 1.5)
```



Sequential association rules

Sequential association rule

A rule $\text{subseq}_1 \rightarrow \text{subseq}_2$ such that

1. Has a minimal support
2. When subseq_1 occurs, it is most often followed by subseq_2

- Extracted from frequent sequences.
- Extraction criteria:
  - Confidence: $p(\text{subseq}_2 \mid \text{subseq}_1)$
  - Lift: $\frac{p(\text{subseq}_2 \mid \text{subseq}_1)}{p(\text{subseq}_2)}$)
  - ...

Extracting association rules

- From the mined frequent subsequences, we can extract association rules :

```
R> rules <- TraMineR:::seqerules(mvad.fsubseq)
R> rules[order(rules$Lift, decreasing = TRUE)[1:10], 1:4]
```

	Rules	Support	Conf	Lift
7	(SC) => (HE)	111	0.5692308	2.226881
50	(SC) => (HE)-(EM)	36	0.1846154	1.961883
23	(JL) => (EM)-(JL)	56	0.2295082	1.201543
9	(TR) => (JL)	96	0.3855422	1.125025
28	(TR) => (EM)-(JL)	53	0.2128514	1.114340
19	(TR) => (JL)-(EM)	63	0.2530120	1.098443
2	(TR) => (EM)	226	0.9076305	1.087934
32	(FE)-(TR) => (EM)	50	0.8771930	1.051450
36	(EM)-(TR) => (EM)	46	0.8679245	1.040341
10	(FE) => (HE)	90	0.2578797	1.008848

Exercises

Exercise 5.1

1. Use the `biofam` data set.
2. Create an event sequence from the state sequence object.
3. Find the most frequent subsequences.
4. Find the subsequences that discriminate the most the cohort covariable.
5. Plot the results.
6. Extract all association rules with a "Lift" greater than 1.

References I

R. Agrawal and R. Srikant. Mining sequential patterns. In Philip S. Yu and Arbee L. P. Chen, editors, *Proceedings of the International Conference on Data Engeneering (ICDE), Taipei, Taiwan*, pages 487–499. IEEE Computer Society, 1995.

D. McVicar and M. Anyadike-Danes. Predicting successful and unsuccessful transitions from school to work using sequence methods. *Journal of the Royal Statistical Society A*, 165(2):317–334, 2002.

Nicolas S. Müller, Matthias Studer, Gilbert Ritschard, and Alexis Gabadinho. Extraction de règles d'association séquentielle à l'aide de modèles semi-paramétriques à risques proportionnels. *Revue des nouvelles technologies de l'information RNTI*, E-19:25–36, 2010.

Matthias Studer, Nicolas S. Müller, Gilbert Ritschard, and Alexis Gabadinho. Classer, discriminer et visualiser des séquences d'événements. *Revue des nouvelles technologies de l'information RNTI*, E-19:37–48, 2010.

Sequence analysis for social scientists

Part VI - Overview of sequence analysis with Stata

Alexis Gabadinho, Matthias Studer, Gilbert Ritschard, Nicolas S. Müller

Department of Econometrics, University of Geneva
http://mephisto.unige.ch/biomining

Summer School on Advanced Methods for the Analysis of Complex Event History Data, Bristol, 28-29 June 2010

Exporting from R

- We begin by exporting the mvad data frame
- We do some data preparation in R because it is easier
- We rename some variables because original names contain a '.' that is not supported by Stata

```
> names(mvad)[6] <- "N_Eastern"
> names(mvad)[8] <- "S_Eastern"
> names(mvad)[15:86] <- paste("month", 1:72, sep = "")
```

- We export the mvad data frame in the Stata format

```
> library(foreign)
> write.foreign(mvad, datafile = "mvad.dta", codefile = "import-mvad.do",
+       package = "Stata")
```


Importing in Stata and preparing the data

- We import the mvad data set by running the '.do' script produced by the write.foreign() function

```
. do "Z:\...\import-mvad.do"
```

- We need to reshape the sequence data (data have to be in a kind of "SPELL" format)

```
. reshape long month, i(id) j(order)
```

- Now we declare the sequence data (in the same way as we do with seqdef() in R)

```
. sqset month id order
```


Sequence frequency tables and index plots

- The sqtab command displays the sequence frequency table

```
. sqtab, ranks(1/10)
```

- The sqindexplot produces sequence index plots

```
. sqindexplot, by(gcse5eq)
```

Optimal matching - I

- Constant substitution cost matrix (default value is 1 for indels and 2*indel cost for substitutions)

```
. sqom
```

- Distances are standardized by dividing each distance by the length of the longest sequence in the data set
- By default, sqom performs Optimal Matching between each sequence and the most frequent sequence in the data
- The distances are stored in the _SQdist variable

Optimal matching - II

- Defining a substitution cost matrix has to be done manually (we use the output of seqsubm(mvad.seq, method="TRATE") to define substitution costs based on observed transition rates)

```
. matrix sub = (
0.000000,1.967601,1.987270,1.951231,1.984684,1.959993\
1.967601,0.000000,1.993341,1.963360,1.987531,1.992045\
1.987270,1.993341,0.000000,1.996033,1.982969,1.999488\
1.951231,1.963360,1.996033,0.000000,1.985649,1.972029\
1.984684,1.987531,1.982969,1.985649,0.000000,1.994867\
1.959993,1.992045,1.999488,1.972029,1.994867,0.000000)
```

- Now we compute the pairwise distances by adding the full option (the distance matrix is stored as _SQdist)

```
. sqom, subcost(sub) full
```


Clustering

- First we have to issue sqclusterdat, which constructs a dataset built from the last sqom command

```
. sqclusterdat
```

- Now we perform a hierarchical clustering and select the number of clusters

```
. clustermat wardslinkage SQdist, name(wards) add
. cluster tree wards, cutnumber(4)
```

- Finally we merge the cluster results with the original sequence data

```
. sqclusterdat, return
```


References I

Christian Brzinsky-Fay, Ulrich Kohler, and Magdalena Luniak. Sequence analysis with Stata. *The Stata Journal*, 6:435 460, 2006. URL http://www.wz-berlin.de/~brzinsky-fay/publications.de.htm.

Sequence analysis for social scientists

Part VII - Conclusion

Matthias Studer, Alexis Gabadinho, Gilbert Ritschard, Nicolas S. Müller

Department of Econometrics, University of Geneva
http://mephisto.unige.ch/biomining

Summer School on Advanced Methods for the Analysis of Complex Event History Data, Bristol, 28-29 June 2010

Sequence of analysis (state sequences) - I

- Exploring sequence distribution (seqdplot, seqfplot, seqiplot)
- Characteristics of the set of sequences
  - Sequence of transversal characteristics (entropies, modal states, ...)
  - Distribution of longitudinal characteristics (entropy, turbulence, time spent in each state, ...)
    - Association between longitudinal characteristics of parallel sequences (family-profession, ego-partner, ...)
- Preceding analyses by groups (sex, birth cohorts, ...), comparisons

Sequence of analysis (state sequences) - II

- Defining a dissimilarity measure between individual sequences to:
  - find representative sequence (most frequent, centroid, ...)
  - compute the dispersion of the sequences
  - build a typology (cluster analysis)
    - study the relationships between clusters and covariates (sex, cohort, ...), logistic models ...
  - plot sequences in a plan using Multi-dimensional scaling
  - assess the association with a covariable using ANOVA (Analysis of discrepancy: Part of discrepancy explained by one or several factors).
  - segment the sequences in homogenous groups through tree structured approach.

Sequence of analysis (event sequences)

- Seeking frequent subsequences
- Studying the relationship between frequent event subsequences and covariates (sex, cohort, ...), logistic models.
- Studying the effect of experimenting the subsequence on entropy, or other response variable ...
- Finding most discriminant subsequences for a given categorical variable (sex, cluster, ...) Extracting sequential association rules (next release of TraMineR)

The TraMineR website

TraMineR website:

- `http://mephisto.unige.ch/traminer/`

There, you may:

- subscribe to the TraMineR user mailing list.
- submit a bug report or a feature request.
- find documentation and articles about TraMineR.
- find a list of articles using TraMineR (maybe yours?)

References I

Andrew Abbott and John Forrest. Optimal matching methods for historical sequences. *Journal of Interdisciplinary History*, 16:471–494, 1986.

Andrew Abbott and Alexandra Hrycak. Measuring resemblance in sequence data: An optimal matching analaysis of musician's carrers. *American Journal of Sociolgy*, 96(1):144–185, 1990.

Alexis Gabadinho, Gilbert Ritschard, Matthias Studer, and Nicolas S. Müller. Summarizing sets of categorical sequences. In *International Conference on Knowledge Discovery and Information Retrieval, Madeira, 6-8 October, 2009*, pages 62–69. INSTICC, 2009a. (Received the Best Paper Award).

Alexis Gabadinho, Gilbert Ritschard, Matthias Studer, and Nicolas S. Müller. Mining sequence data in R with the TraMineR package: A user's guide. Technical report, Department of Econometrics and Laboratory of Demography, University of Geneva, Geneva, 2009b. URL `http://mephisto.unige.ch/traminer/`.

Alexis Gabadinho, Gilbert Ritschard, Matthias Studer, and Nicolas S. Müller. Indice de complexité pour le tri et la comparaison de séquences catégorielles. *Revue des nouvelles technologies de l'information RNTI*, E-19:61–66, 2010.

References II

D. McVicar and M. Anyadike-Danes. Predicting successful and unsuccessful transitions from school to work using sequence methods. *Journal of the Royal Statistical Society A*, 165(2):317–334, 2002.

Nicolas S. Müller, Alexis Gabadinho, Gilbert Ritschard, and Matthias Studer. Extracting knowledge from life courses: Clustering and visualization. In Il-Yeol Song, Johann Eder, and Tho Manh Nguyen, editors, *Data Warehousing and Knowledge Discovery, 10th International Conference, DAWAK 2008, Turin, Italy, September 2-5*, volume LNCS 5182 of *Lectures Notes in Computer Science*, pages 176–185. Springer, Berlin Heidelberg, 2008a. doi: 10.1007/978-3-540-85836-2_17.

Nicolas S. Müller, Sylvain Lespinats, Gilbert Ritschard, Matthias Studer, and Alexis Gabadinho. Visualisation et classification des parcours de vie. *Revue des nouvelles technologies de l'information RNTI*, E-11, II:499–510, 2008b.

Nicolas S. Müller, Matthias Studer, Gilbert Ritschard, and Alexis Gabadinho. Extraction de règles d'association séquentielle à l'aide de modèles semi-paramétriques à risques proportionnels. *Revue des nouvelles technologies de l'information RNTI*, E-19:25–36, 2010.

References III

Matthias Studer, Gilbert Ritschard, Alexis Gabadinho, and Nicolas S. Müller. Analyse de dissimilarités par arbre d'induction. *Revue des nouvelles technologies de l'information RNTI*, E-15:7–18, 2009.

Matthias Studer, Nicolas S. Müller, Gilbert Ritschard, and Alexis Gabadinho. Classer, discriminer et visualiser des séquences d'événements. *Revue des nouvelles technologies de l'information RNTI*, E-19:37–48, 2010a.

Matthias Studer, Gilbert Ritschard, Alexis Gabadinho, and Nicolas S. Müller. Discrepancy analysis of complex objects using dissimilarities. In Fabrice Guillet, Gilbert Ritschard, Henri Briand, and Djamel A. Zighed, editors, *Advances in Knowledge Discovery and Management*, Studies in Computational Intelligence. Springer, Berlin, 2010b. (forthcoming).