Exploration de données séquentielles avec TraMineR

Gilbert Ritschard

Alexis Gabadinho, Nicolas S. Müller, Matthias Studer

Département d'économétrie et Laboratoire de démographie
Université de Genève
http://mephisto.unige.ch/traminer

Tutoriel SFC 2010

Outline

Objectifs

- Introduction à l'analyse de données séquentielles catégorielles
- Définir et manier des ensembles de séquences
- Identifier les types de séquences
  - avec/sans dimension temporelle
  - séquence d'états, séquences d'événements
- Quelles caractéristiques souhaite-t-on mettre en évidence ?
  - séquences de caractéristiques transversales
  - caractéristique longitudinale de séquences individuelles
  - (dis)similarités deux-à-deux de séquences
  - typologie de séquences
  - ...

Objectifs (suite)

- Formation à la pratique de l'analyse de séquences (TraMineR)
  - Organiser, préparer et manier des séquences.
  - Caractériser et visualiser des ensembles de séquences d'états (vision transversale et longitudinale)
  - Calculer des dissimilarités deux-à-deux entre séquences
  - Classification non-supervisée de séquences d'états
  - Recherche de sous-séquences fréquentes et d'événements et identification de sous-séquences discriminantes

TraMineR

- TraMineR : Trajectory Miner in R

  (Accessoirement inspiré par notre goût pour le Gwürtzraminer)
- Librairie pour l'environnement gratuit et open source R
  - librement disponible sur le CRAN (Comprehensive R Archive Network) `http://cran.r-project.org`
- TraMineR s'intégrant dans R, il peut être simplement et directement combiné avec les autres librairies de R
  - Par exemple, les dissimilarités obtenues avec TraMineR peuvent être utilisées avec les procédures déjà optimisées de clustering, de MDS, les outils de régression linéaire et non-linéaire, ...

Ce que TraMineR permet de faire

- Prise en charge de différents types de données longitudinales et conversion entre formats
- Visualisation d'un ensemble de séquences (index plot, séquences fréquentes, distributions transversales, et plus...)
- Caractéristiques longitudinales de séquences individuelles (complexité, durées de séjour dans chaque état, entropie longitudinale, turbulence, et plus ...)
- Séquence de caractéristiques transversales (distribution des états, entropie transversale, état modal)
- Autres caractéristiques agrégées (taux de transition, durées moyennes de séjour dans chaque état)
- Dissimilarités entre paires de séquences (Optimal matching, Longest Common Subsequence, Hamming, Dynamic Hamming, Multichannel et plus ...)
- Séquences représentatives et mesures de dispersion d'un ensemble de séquences
- Analyse de type ANOVA et arbres d'induction à partir de matrices de dissimilarités
- Extraction de séquences d'événements fréquents
- Identification de séquences d'événements discriminantes
- Règles d'association entre sous-séquences

Autres logiciels pour l'analyse de séquences

- Optimize le logiciel d'Abbott (Abbott, 1997)
  - Calcul des distances d'optimum matching
  - Plus maintenu
- TDA (Rohwer and Pötter, 2002)
  - logiciel statistique gratuit, calcul des distances d'optimum matching
- SQ-ados, macros Stata, (Brzinsky-Fay et al., 2006)
  - gratuit si on a une licence Stata
  - distances optimal matching, visualisation
- CHESA logiciel gratuit de Elzinga (2007)
  - Nombreuses métriques, dont plusieurs non fondées sur l'alignement
  - Turbulence
- MARCH (Berchtold and Berchtold, 2004)
  - Modèles des transitions, chaînes de Markov cachées, ...

Le projet de recherche

TraMineR et le fruit d'un projet projet FNS

- Mining event histories : Towards new insights on personal Swiss life courses

- Project FN 100012-113998 and FN-100015-122230
- Début : 1er février 2007 Fin : 31 janvier 2011

- Gilbert Ritschard, prof. de statistique, requérant principal,
- Eric Widmer, prof. of Sociologie, co-requérant
- Alexis Gabadinho, Demographie
- Nicolas S. Müller, Sociologie, Système d'information
- Matthias Studer, Economie, Sociologie

Séquences en sciences sociales

- TraMineR a été conçu pour répondre à des questions de sciences sociales
- Les séquences (suite d'états ou d'événements) décrivent des trajectoires de vie
  - Les parcours de vie obéissent-ils à une norme sociale ?
    - Quelles sont les types de trajectoires standards ?
    - Quels écarts observe-t-on par rapport à ces normes ?
  - Pourquoi certaines personnes suivent-elles des trajectoires plus chaotiques que d'autres ?
  - Comment les trajectoires de vie sont-elles liées au sexe, à l'origine sociale et à d'autres facteurs
- Les outils proposés s'appliquent également à de nombreux autres domaines
  - fouille de texte, biologie,
  - monitoring de l'activation d'appareils
  - étude de comportements temporels d'acheteurs ou d'utilisateurs,

Le jeu de données mvad

- Etude de McVicar and Anyadike-Danes (2002) sur la transition entre formation et emploi en Irlande du Nord.
- Jeu de données distribué avec la librairie TraMineR.
- Provient d'une enquête auprès 712 jeunes irlandais.
- Les séquences représentent leur suivi pendant les 6 années suivant la fin de la scolarité obligatoire (16 ans) et sont constituées des 70 variables indiquant les états mensuels successifs de chaque individu entre septembre 1993 et juin 1999.
- Les états sont :

EM	en emploi
FE	formation secondaire
HE	formation supérieure
JL	au chômage
SC	école
TR	en stage ou apprentissage.

Séquences d'états - Jeu de données mvad

- Premières séquences du jeu de données

```
  Sequence
1 EM-EM-EM-EM-TR-TR-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
3 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
```

- Représentation plus compacte (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)
```


Aperçu des possibilités de TraMineR

- Charger TraMineR et créer un objet 'séquences d'états'

```
R> library(TraMineR)
R> data(mvad)
R> mvad.seq <- seqdef(mvad, 17:86)
```

- Calcul des dissimilarités OM entre paires de séquences avec un coût d'indel de 1 et des coûts de substitutions déduits des taux de transitions

```
R> mvad.om <- seqdist(mvad.seq, method = "OM", indel = 1, sm = "TRATE")
```

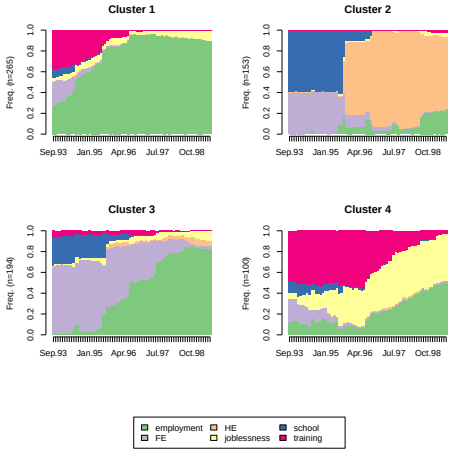
- Classification en 4 groupes par une procédure agglomérative avec critère de Ward

```
R> library(cluster)
R> clusterward <- agnes(mvad.om, diss = TRUE, method = "ward")
R> mvad.cl4 <- cutree(clusterward, k = 4)
R> cl4.lab <- factor(mvad.cl4, labels = paste("Cluster", 1:4))
```

Aperçu des possibilités de TraMineR (suite 1)

- Visualisation des classes : distributions transversales des états

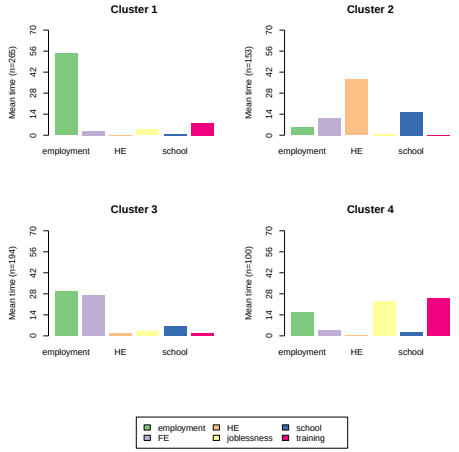
```
R> seqdplot(mvad.seq, group = cl4.lab, border = NA)
```



Aperçu des possibilités de TraMineR (suite 2)

- Temps moyen dans les états par classe

```
R> seqmtplot(mvad.seq, group = cl4.lab, border = NA)
```



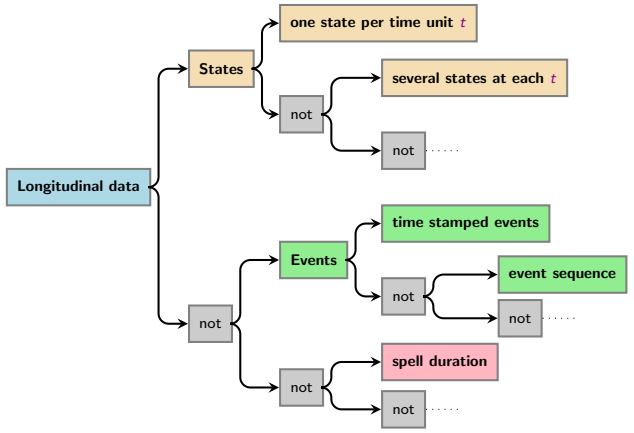
Types de séquences

Nature des séquences

Dépend de

- Information données par la position j dans la séquence
  - Dimension temporelle ? (pas le cas dans les textes ou les séquences de protéines !)
  - La position informe-t-elle sur la date, l'âge, la distance par rapport au début du processus ?
- Nature des éléments de l'alphabet
  - états
    - La position informe sur la date, l'âge, la durée depuis le début du processus.
  - transitions ou événements
    - La position ne donne qu'une information relative, pas de durée.

Ontologie de présentation de données longitudinales
(Aristotelian tree)



Formats de séquences d'états : exemples - I

Code : STS

Id	18	19	20	21	22	23	24	25	26	27
101	S	S	S	M	M	MC	MC	MC	MC	D
102	S	S	S	MC	MC	MC	MC	MC	MC	MC

Code : SPS

Id	1	2	3	4
101	(S,3)	(M,2)	(MC,4)	(D,1)
102	(S,3)	(MC,7)		

Code : SSS*

Id	1	2	3	4
101	(S,18)	(M,21)	(MC,23)	(D,27)
102	(S,18)	(MC,21)		

Code : SRS

Id	$t-9$	$t-8$	$t-7$	$t-6$	$t-5$	$t-4$	$t-3$	$t-2$	$t-1$	t
101	S	S	S	M	M	MC	MC	MC	MC	D
101	.	S	S	S	M	M	MC	MC	MC	MC
101	.	.	S	S	S	M	M	MC	MC	MC
⋮										
101	.	.	.	.	.	.	.	.	S	S
102	S	S	S	MC	MC	MC	MC	MC	MC	MC
102	.	S	S	S	MC	MC	MC	MC	MC	MC
⋮										

Code : DSS

Id	1	2	3	4
101	S	M	MC	D
102	S	MC		

Formats de séquences d'états : exemples - II

Code : SPELL

Id	*Index*	*From*	*To*	*State*
101	1	18	20	Single (S)
101	2	21	22	Married (M)
101	3	23	26	Married w Children (MC)
101	4	27	27	Divorced (D)
102	1	18	20	Single (S)
102	2	21	27	Married w Children (MC)

Code : PPER*

Id	*Index*	*Age*	*State*
101	1	18	Single (S)
101	2	19	Single (S)
101	3	20	Single (S)
101	4	21	Married (M)
⋮	⋮	⋮	
101	10	27	Divorced (D)
102	1	18	Single (S)
⋮	⋮	⋮	

Séquences d'événements : exemples

Code : FCE*

Id	*#marr.*	*marr1*	*marr2*	⋯	*#child.*	*child1*	*child2*	⋯
101	1	21	.	.	2	23	26	.
102	1	21	.	.	1	21	.	.

Code : HTSE*

Id	1	2	3	⋯
101	(marriage, 21)	(childbirth, 23)	(childbirth, 26)	(divorce, 27)
102	(marriage, 21)	(childbirth, 21)		

Code : TSE

Id	*Time*	*Event*
101	21	Marriage
101	23	Childbirth
101	26	Childbirth
101	27	Divorce
102	21	Marriage
102	21	Childbirth

Importation et Gestion de données dans TraMineR

- On utilise les fonctions d'importation de données de R
- A partir d'un `data.frame` on crée
  - soit un objet 'séquences d'états' avec `seqdef()`
  - soit un objet 'séquences d'événements' avec `seqecreate()`
- Les séquences d'états doivent être sous forme STS, SPS ou SPELL
  - Utiliser `seqformat()` pour convertir formats
  - `seqconc()` et `seqdecomp()` pour transformer entre présentations sous forme de textes et vectorielles.
- Les séquences d'événements doivent être en format TSE
  - `seqecreate()` peut dériver des séquences d'événements à partir d'un objet 'séquences d'états'.
- TraMineR permet aussi de contrôler l'affichage de séquences

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

Caractéristiques d'un ensemble de séquences

- Séquences de mesures transversales (état modal, entropie transversale, ...)

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

- Résumé de mesures longitudinales (nombre de transitions, entropie longitudinale, durées moyennes ...)

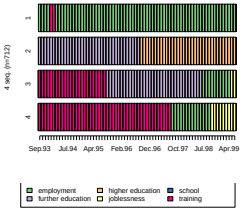
id	t_1	t_2	t_3	$\cdots$
1	B	B	D	$\cdots$
2	A	B	C	$\cdots$
3	B	B	A	$\cdots$

- Caractéristiques globale : séquences représentatives, dispersion de séquences, ...

Visualisation de séquences individuelles

- Une séquence d'états se représente naturellement par un empilement (horizontal) de carrés unitaires coloriés selon l'état représenté.
  - La couleur à chaque position reflète l'état à cette position
  - La longueur des segments d'une même couleur reflète la durée successive dans l'état.

```
   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)
```



Création de l'objet 'séquences d'états'

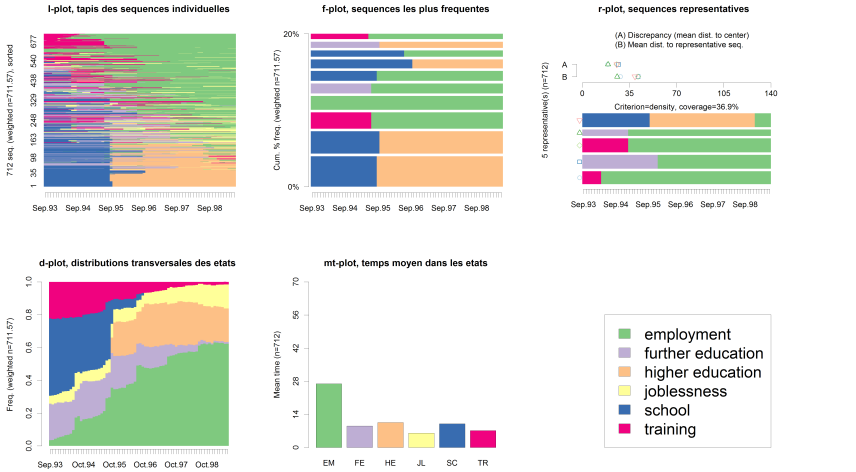
- L'objet séquences d'états comprend
  - les séquences
  - et leurs attributs (alphabet, étiquettes, couleurs, poids, ...)
- La première opération à effectuer est donc la création de l'objet 'state sequence'

```
R> data(mvad)
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, weights = mvad$weight)
R> mds.mvad <- cmdscale(mvad.om, k = 2)
```


Objet 'séquences d'états'

- L'objet séquences d'états est au cœur des commandes TraMineR pour séquences d'états.
- En plus de l'ensemble des séquences, il inclut en particulier
  - l'alphabet
  - les noms courts des états pour les sorties textes (`states`)
  - les étiquettes longues des états pour les légendes des couleurs dans les graphiques (`labels`)
  - les étiquettes des positions (`cnames`)
  - la palette des couleurs pour représenter les états (`cpal`)
  - les poids (`weights`)
  - des spécifications sur les valeurs manquantes (`left, gaps, right`)
  - ...
- Pour assurer l'homogénéité des sorties textes et des figures, toutes ces caractéristiques sont définies une seule fois avec `seqdef()` ... et récupérées quand nécessaire par les autres fonctions TraMineR.

Présentations graphiques : Exemples



Principes des fonctions graphiques

- Les fonctions graphiques de TraMineR (alias de `seqplot()`)

  `seqiplot()` `seqdplot()`
  `seqIplot()` `seqHtplot()`
  `seqfplot()` `seqmsplot()`
  `seqrplot()` `seqmtplot()`
  `seqlegend()`

- argument obligatoire : objet séquences d'états (cf `seqdef()`)
- les fonctions partagent notamment les arguments

`group=NULL`	variable de partitionnement (plots par groupe)
`withlegend="auto"`	autre valeurs : `FALSE`, `"right"`
`title=NULL`	titre (chaîne de caractère)
`ylab=NULL`	libellé de l'axe *y*, utiliser `NA` pour supprimer
`xaxes=NULL`	axes des *x*, utiliser `NA` pour supprimer

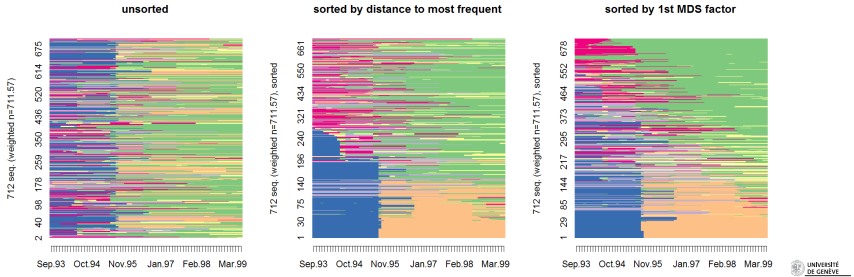
- Voir aide en ligne (`?seqplot`) pour plus de détails

Les fonctions `seqiplot()` et `seqIplot()`

- `seqiplot()` visualise une sélection de séquences
  - sélection contrôlée avec `tlim`
  - exemples : `tlim=1:10` (défaut), `tlim=c(24,2,56)`
  - les indices se réfèrent à l'ordre retenu des séquences
  - utiliser
    - `border=NA` pour supprimer les cadres autour de chaque état et
    - `space=0` pour supprimer l'espace entre séquences
- `seqIplot()` visualise toutes les séquences avec (`border=NA`, `space=0`)
- `sortv=` permet de donner l'ordre des séquences.

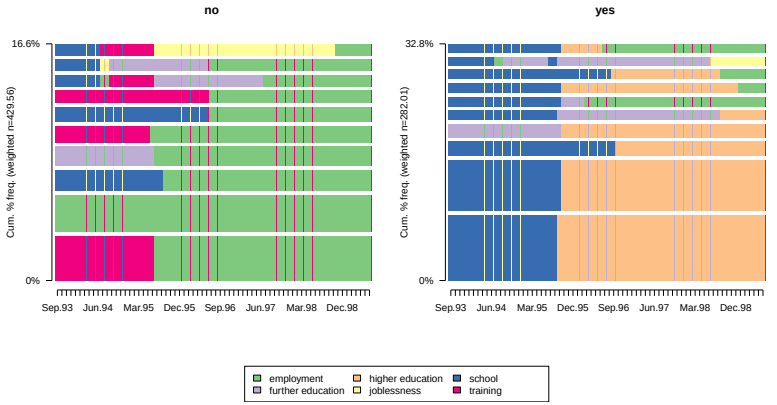
`seqiplot` et ordre des séquences

```
R> dref <- seqdist(mvad.seq, ref = 0, method = "LCS")
R> mvad.lcs <- seqdist(mvad.seq, method = "LCS")
R> mds <- cmdscale(mvad.lcs, k = 1)
R> par(mfrow = c(1, 3))
R> seqIplot(mvad.seq, title = "unsorted", withlegend = FALSE)
R> seqIplot(mvad.seq, title = "sorted by distance to most frequent",
+       withlegend = FALSE, sortv = dref)
R> seqIplot(mvad.seq, title = "sorted by 1st MDS factor", withlegend = FALSE,
+       sortv = mds)
```



Argument group= : Graphique par groupes

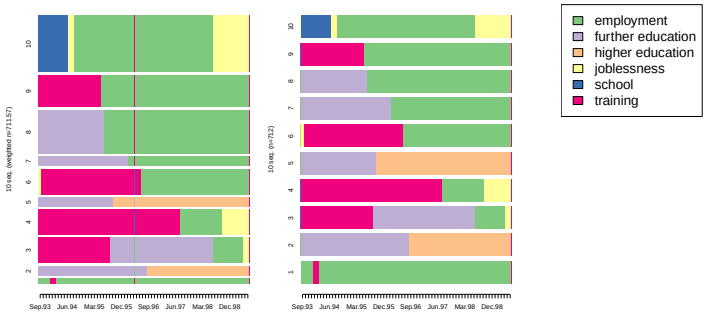
```
R> seqfplot(mvad.seq, group = mvad$gcse5eq, border = NA)
```



A noter : production automatique de la légende

Prise en compte des poids

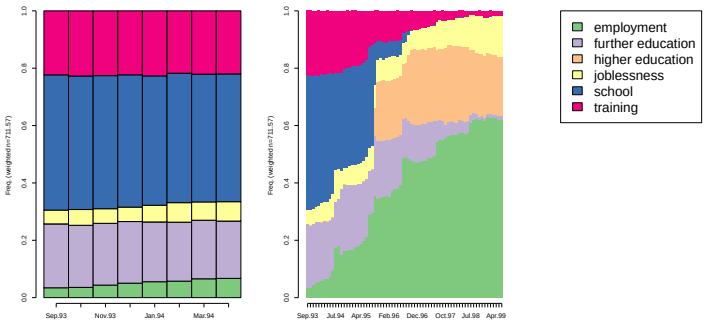
```
R> par(mfrow = c(1, 3))
R> seqiplot(mvad.seq, border = NA, withlegend = FALSE)
R> seqiplot(mvad.seq, border = NA, withlegend = FALSE, weighted = FALSE)
R> seqlegend(mvad.seq, fontsize = 2)
```



! Légende automatique utilise `layout` pas compatible avec `par(mfrow=...)`

Distributions transversales des états
graphique

```
R> par(mfrow = c(1, 3))
R> seqdplot(mvad.seq[, 1:8], withlegend = FALSE)
R> seqdplot(mvad.seq, border = NA, withlegend = FALSE)
R> seqlegend(mvad.seq, fontsize = 2)
```

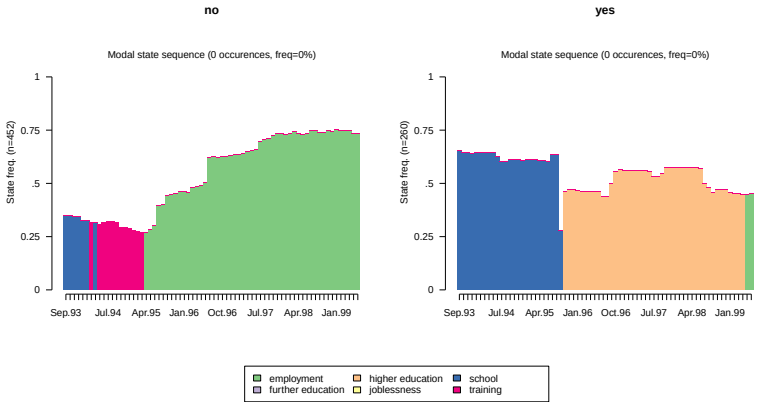


Distribution transversales des états
tableau

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

     [State frequencies]
   Sep.93 Oct.93 Nov.93 Dec.93 Jan.94 Feb.94 Mar.94 Apr.94
EM  0.034  0.036  0.044  0.051  0.055  0.058  0.066  0.067
FE  0.223  0.217  0.215  0.215  0.209  0.206  0.204  0.200
HE  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
JL  0.048  0.055  0.051  0.050  0.058  0.068  0.064  0.067
SC  0.472  0.466  0.464  0.461  0.451  0.451  0.446  0.446
TR  0.223  0.227  0.226  0.223  0.227  0.218  0.221  0.220

     [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    0.72   0.73   0.73   0.74   0.75   0.76   0.77   0.77
```

Séquence des états modaux

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

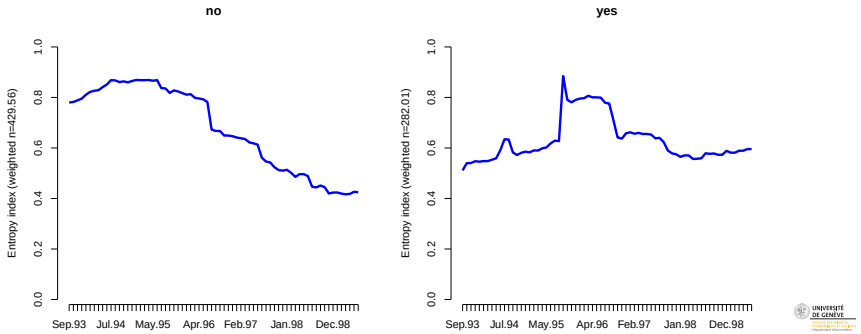


Séquence des entropies transversales

Diversité des états

- Entropie : $H(p_1, \ldots, p_a) = -\sum_{i=1}^{a} p_i \log(p_i)$
- 0 si un p_i vaut 1 et tous les autres sont nuls
- maximum si tous les p_i égaux (uniforme)

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



- Caractéristiques des séquences individuelles

`seqistatd()`	temps dans chaque état (distribution longitudinale)
`seqlength()`	longueur de la séquence
`seqtransn()`	nombre de transitions
`seqdss()`	liste des états successifs distincts (DSS)
`seqdur()`	liste des durées dans les états de la DSS
`seqsubsn()`	nombre de sous-séquence (`DSS=FALSE`)
`seqient()`	entropie longitudinale
`seqST()`	Turbulence (Elzinga and Liefbroer, 2007)
`seqici()`	indice de complexité (Gabadinho et al., 2010)

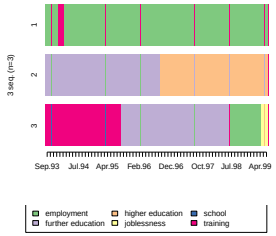
Etats successifs distincts et leur durées

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

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

```
R> seqdur(mvad.seq)[1:3, 1:5]
```

```
  DUR1 DUR2 DUR3 DUR4 DUR5
1    4    2   64   NA   NA
2   36   34   NA   NA   NA
3   24   34   10    2   NA
```




```
R> tab <- data.frame(seqlength(mvad.seq), seqtransn(mvad.seq),
+       seqsubsn(mvad.seq, DSS = TRUE))
R> tab[1:3, ]
```

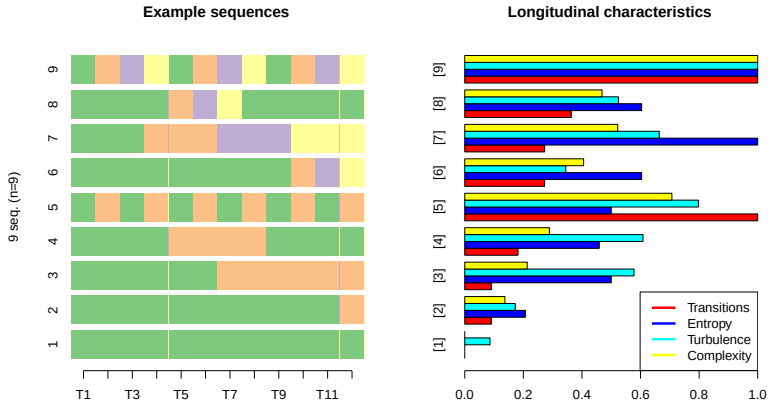
```
  Length Trans. Subseq.
1     70      2       7
2     70      1       4
3     70      3      16
```

Complexité des séquences

- Pour évaluer la complexité de la composition de la séquence on peut considérer
- L'entropie longitudinale
  - mesure la diversité des états qui composent la séquences
  - fondé sur le temps passé dans les divers états
  - ne tient pas compte du séquencement des états

  La Turbulence (Elzinga and Liefbroer, 2007)
  - mesure composite fondée sur
    - le nombre de sous-séquences de la séquence DSS
    - la variance des durées dans les états successifs
  - sensible au séquencement
- L'indice de complexité (Gabadinho et al., 2010)
  - mesure composite fondée sur
    - le nombre de transitions
    - l'entropie longitudinale
  - sensible au séquencement

Comportement de mesures longitudinales



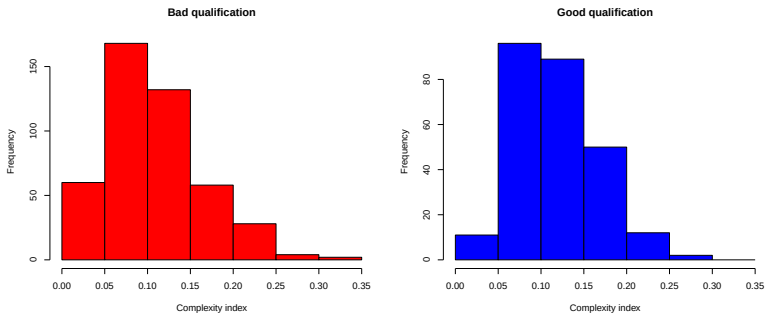
Exemple d'analyse de la complexité

```
R> mvad$ici <- seqici(mvad.seq)
R> lm.ici <- lm(ici ~ male + funemp + gcse5eq, mvad)
```

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	0.11	0.00	28.01	0.00
male	-0.01	0.00	-3.04	0.00
father unemployed	0.01	0.01	1.24	0.22
good ECS grade	0.01	0.00	2.20	0.03

Exemple d'analyse de la complexité (suite)

```
R> par(mfrow = c(1, 2))
R> h.ici <- hist(mvad$ici[mvad$gcse5eq == "no", ], main = "Bad qualification",
+       col = "red", xlab = "Complexity index")
R> hist(mvad$ici[mvad$gcse5eq == "yes", ], main = "Good qualification",
+       col = "blue", xlab = "Complexity index", breaks = h.ici$breaks)
```



Dissimilarités entre séquences

- Dissimilarités entre séquences
  - Différentes mesures (LCP, RLCP, LCS, OM, HAM, DHD, ...)
- Avec une matrice des dissimilarités, on peut
  - Déterminer les séquences représentatives (Gabadinho et al., 2009b)
  - Utiliser une procédure de regroupement automatique pour construire une typologie
  - Représenter les séquences dans un plan (MDS),
  - Analyser la dispersion des séquences (ANOVA) (Studer et al., 2010)
  - Construire un arbre de régression sur les séquences

Les mesures de dissimilarités

- Mesures fondées sur une mesure de proximité $A(x, y)$

$$d(x, y) = A(x, x) + A(y, y) - 2A(x, y)$$

  - LCP $A(x, y)$ = longueur du plus long préfixe commun
  - RLCP $A(x, y)$ = longueur du plus long suffixe commun
  - LCS $A(x, y)$ = longueur plus longue sous-séquence commune
  - HAM simple $A(x, y)$ = moitié du nombre d'états identiques
- Coût minimal de transformation de x en y
  - OM Appariement optimal (Levenshtein, 1966)
    - coût de l'indel (insertion/suppression)
    - coûts de substitution entre paires d'états
  - HAM, Hamming = OM sans indel
  - DHD, Dynamic Hamming Distance, Coûts de substitution variant dans le temps (position) (Lesnard, 2006)

La fonction `seqdist()`

- Le calcul des dissimilarité se fait avec `seqdist()`
- Exemple : matrice des distances LCS

```
R> mvad.lcs <- seqdist(mvad.seq, method = "LCS")
R> mvad.lcs[1:4, 1:8]
```

```
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
[1,]    0  140  116  108  140   64   60   44
[2,]  140    0   72  140   22  140   80   96
[3,]  116   72    0   68   90   72   60   76
[4,]  108  140   68    0  140   46  112  112
```

- Distance à la séquence la plus fréquente

```
R> seqdist(mvad.seq, method = "LCS", refseq = 0)[1:8]
```

```
[1] 140  72 140 140  50 140 140 140
```


Matrice des coûts de substitutions

- Pour OM (appariement optimal) on doit fournir les coûts
- Matrice des coûts de substitution ($a \times a$)
- Exemple : coûts basés sur les taux de transition entre états

```
R> mvad.sm <- seqsubm(mvad.seq, method = "TRATE")
R> round(mvad.sm, digits = 3)
```

```
      EM->  FE->  HE->  JL->  SC->  TR->
EM-> 0.000 1.971 1.987 1.957 1.988 1.961
FE-> 1.971 0.000 1.993 1.977 1.991 1.993
HE-> 1.987 1.993 0.000 1.997 1.981 1.999
JL-> 1.957 1.977 1.997 0.000 1.992 1.976
SC-> 1.988 1.991 1.981 1.992 0.000 1.995
TR-> 1.961 1.993 1.999 1.976 1.995 0.000
```

- Utiliser l'option `time.varying=TRUE` pour des coûts variables dans le temps (matrice à 3 dimensions)

La fonction `seqdist(...,method="OM")`

- coût de l'indel : 1 par défaut
- matrice des coûts de substitution
  - donner la matrice

    R> mvad.om <- seqdist(mvad.seq, method = "OM", sm = mvad.sm)
  - donner une méthode de calcul de la matrice

    R> mvad.om <- seqdist(mvad.seq, method = "OM", sm = "TRATE")
  - autre possibilité `sm="CONSTANT"` (coût=2)

R> round(mvad.om[1:4, 1:8], digits = 3)

```
       [,1]    [,2]    [,3]    [,4]    [,5]    [,6]    [,7]    [,8]
[1,]   0.000 138.544 114.054 105.840 138.727  62.737  59.168  43.402
[2,] 138.544   0.000  71.834 139.291  21.921 138.941  79.396  95.162
[3,] 114.054  71.834   0.000  67.722  89.803  71.149  59.796  75.563
[4,] 105.840 139.291  67.722   0.000 139.368  45.074 110.730 110.472
```


Distances normalisées

- Distance $d(x, y) = 10$ entre deux séquences de longueur 6 est plus importante que distance de 10 entre séquences de longueur 100.
- Pour rendre plus comparable, on peut normaliser
- Type de normalisations
  - `maxlength` (Abbott), $d_{\text{norm}} = \frac{d(x,y)}{\max(|x|,|y|)}$
  - `gmean` (Elzinga), $d_{\text{norm}} = 1 - \frac{A(x,y)}{\sqrt{\left(A(x,x)A(y,y)\right)}}$
  - `maxdist`, $d_{\text{norm}} = \frac{d(x,y)}{\max_{theorique} d(x,y)}$
  - `YujianBo` (Yujian & Bo) Une normalisation qui transforme tout $d(x, y)$ respectant l'inégalité triangulaire en un $d_{\text{norm}}(x, y)$ vérifiant également cette inégalité.

Distances normalisées, exemples

- Avec `norm=TRUE`,
  - Abbott pour OM, HAM et DHD
  - Elzinga pour LCP, RLCP, LCS
- On peut aussi choisir la normalisation

R> mvad.lcs.norm <- seqdist(mvad.seq, method = "LCS", norm = "maxdist")

R> round(mvad.lcs.norm[1:4, 1:8], digits = 3)

```
      [,1]  [,2]  [,3]  [,4]  [,5]  [,6]  [,7]  [,8]
[1,] 0.000 1.000 0.829 0.771 1.000 0.457 0.429 0.314
[2,] 1.000 0.000 0.514 1.000 0.157 1.000 0.571 0.686
[3,] 0.829 0.514 0.000 0.486 0.643 0.514 0.429 0.543
[4,] 0.771 1.000 0.486 0.000 1.000 0.329 0.800 0.800
```


Construire une typologie de séquences

- On peut utiliser toute procédure acceptant une matrice de dissimilarités en entrée
- Dans R, librairie `cluster` (Kaufman and Rousseeuw, 2005; Maechler et al., 2005)
  - `agnes()` : regroupement agglomératif (average, ward, ...).
  - `diana()` : analyse divisive.
  - `pam()` : partitionnement autour de medoïdes (non hiérarchique, plus rapide, mais nombre de groupes doit être fixé à priori).

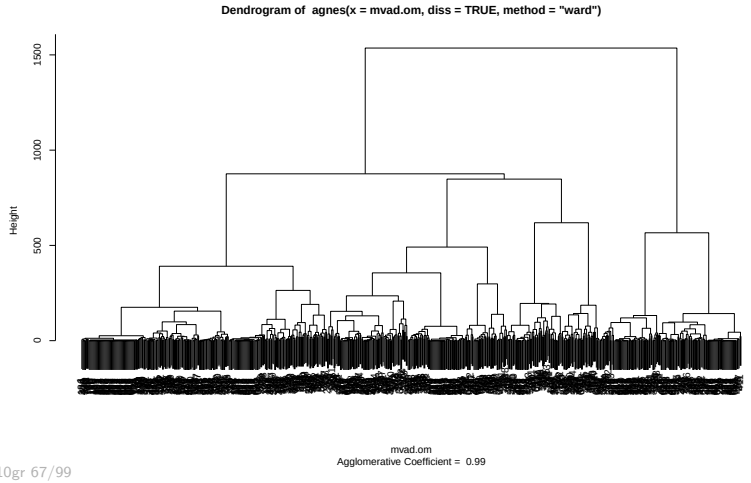
Clustering : exemple

- Exemple d'une classification hiérarchique ascendante avec Ward

```
R> library(cluster)
R> clusterward <- agnes(mvad.om, diss = TRUE, method = "ward")
R> mvad.cl4 <- cutree(clusterward, k = 4)
R> cl4.lab <- factor(mvad.cl4, labels = paste("Cluster", 1:4))
```

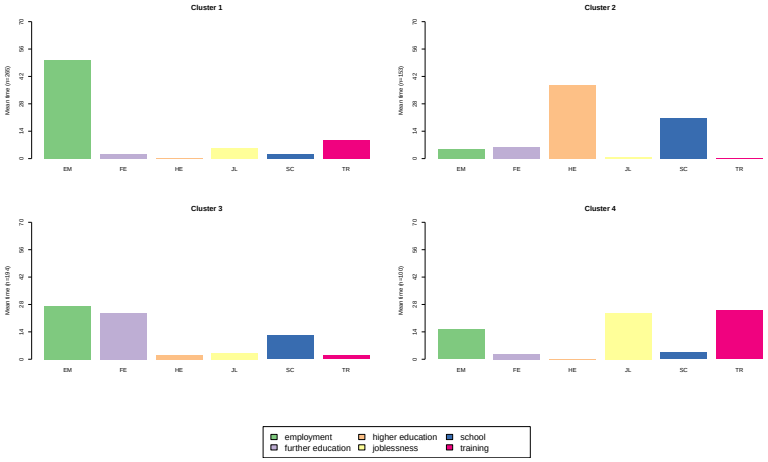

Dendrogramme

```
R> plot(clusterward, which = 2)
```



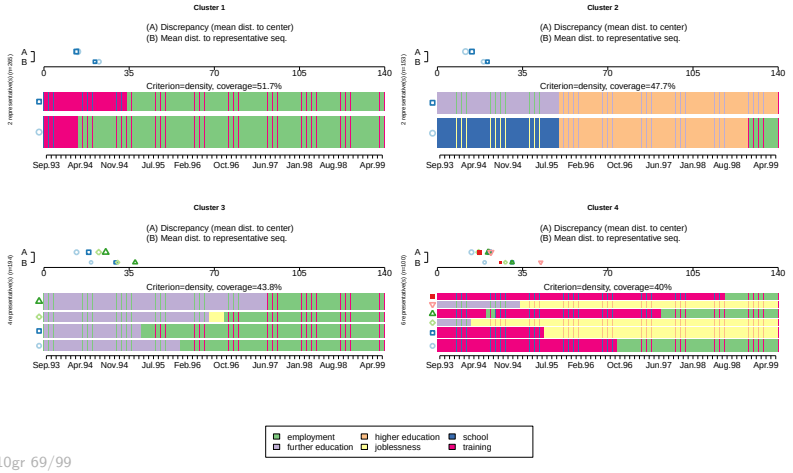
Clusters : représentation graphique

```
R> seqmtplot(mvad.seq, group = cl4.lab, cex.legend = 1.5, border = NA)
```



Clusters : séquences représentatives
(Gabadinho et al., 2009b)

```
R> seqrplot(mvad.seq, group = cl4.lab, cex.legend = 1.5, cex.plot = 1.5,
+       dist.matrix = mvad.om, trep = 0.4, border = NA)
```



Exploration de séquences d'événements
Objectifs

- Accent sur événements, plutôt qu'états.
- Intérêt dans la configuration (*pattern*) d'événements.
  - Configuration type d'événements : événements survenant systématiquement ensemble et dans le même ordre
- Y a-t-il des configurations types (fréquentes) d'événements ?
- Relation avec covariables
  - Quelle configuration caractérise le mieux un groupe par rapport aux autres ?
  - Différences typiques hommes-femmes dans l'ordonnancement des événements.

Evénements et transitions

- Séquence d'événements : suite ordonnée de transitions.
- Transition : ensemble non-ordonné d'événements.

Exemple

(LHome, Union) → (Mariage) → (Enfant)

- (LHome, Union) et (Mariage) sont des transitions.
- "LHome", "Union" et "Mariage" sont des événements.

Sous-séquence

- Une sous-séquence B d'une séquence A est une séquence d'événements telle que
  - chaque événement de B est un événement de A,
  - événements de B dans même ordre que dans A.

Exemple

A (LHome, Union) → (Mariage) → (Enfant).
B (LHome, Mariage) → (Enfant).
C (LHome) → (Enfant).

- C est sous-séquence de A et B, car ordre des événements respecté.
- B n'est pas sous-séquence de A, car on ne sait pas dans B si "LHome" survient avant "Mariage".

Sous-séquences fréquentes et discrimiantes

- Support d'une sous-séquence : nombre de séquences contenant la sous-séquence.
  - Sous-séquence fréquente : séquence avec support supérieur à un support minimal.
  - Une sous-séquence est discriminante entre groupes si son support varie significativement entre groupes.

Data Format

- L'analyse de séquences d'événements dans TraMineR requiert un objet séquences d'événements
- On le crée avec `seqecreate()`
- à qui on fournit les données sous l'une des formes suivantes :
  - Evénements datés (TSE, Time Stamped Event), spécifie directement les événements.
  - Un objet séquences d'états avec une des méthodes de conversion suivantes
    - **transition** Un événement distinct pour chaque transition entre états distincts.
    - **state** Un événement distinct pour chaque début d'épisode dans un état distinct.
    - **period** Un événement pour le début et un autre pour la fin de chaque épisode dans un état distinct.

Format "Time Stamped Event" (TSE)

- **id** Identificateur individuel.
- **timestamp** date (valeur réelle) de l'événement.
- **event** l'événement (code string).
- Une ligne par événement.

```
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
```


Création d'un objet séquences d'événements
Avec le format TSE

- Fonction `seqecreate()`.
- Avec les arguments `id`, `timestamp` et `event` on passe les colonnes du TSE

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

- Alternativement, on peut utiliser l'argument `data`

```
R> actcal.seqe <- seqecreate(data = actcal.tse)
```


Création d'un objet séquences d'événements
A partir d'un objet séquences d'états

- Fonction `seqecreate()`.
- Avec l'argument `tevent` on choisi la méthode de conversion.
- Exemple : pour un événement par transition

```
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 = "transition")
```

Définition d'une matrice de transition

- Conversion selon une matrice de transition (qui peut être personnalisée).
- On peut en générer une (`seqecreate()` le fait pour vous)

```
R> seqetm(mvad.seq, method = "transition")
```

```
             employment FE      HE      joblessness school  training
employment   "EM"       "EM>FE" "EM>HE" "EM>JL"     "EM>SC" "EM>TR"
FE           "FE>EM"    "FE"    "FE>HE" "FE>JL"     "FE>SC" "FE>TR"
HE           "HE>EM"    "HE>FE" "HE"    "HE>JL"     "HE>SC" "HE>TR"
joblessness "JL>EM"    "JL>FE" "JL>HE" "JL"        "JL>SC" "JL>TR"
school       "SC>EM"    "SC>FE" "SC>HE" "SC>JL"     "SC"    "SC>TR"
training     "TR>EM"    "TR>FE" "TR>HE" "TR>JL"     "TR>SC" "TR"
```


Représentation des séquences d'événements

- Chaque séquences est affichée dans la forme `(e1,e2,...)-time-(e2,...)-time`
- où `(e1,e2,...)` est la transition définie par l'occurrence simultanée des événements `e1,e2,...`.
- `time` est la durée (valeur numérique) entre deux transitions (ou la fin du temps d'observation)

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

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


Extraire les sous-séquences fréquentes

Fonction `seqefsub()`, à laquelle nous passons

- Les séquences d'événements (un objet séquences d'événements)
- Le support minimal (avec l'argument `pMinSupport`).

```
R> mvad.fsubseq <- seqefsub(mvad.seqe, pMinSupport = 0.01)
R> mvad.fsubseq[1:5]
```

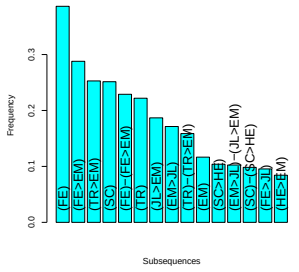
```
   Subsequence   Support Count
1         (FE) 0.3862360   275
2      (FE>EM) 0.2879213   205
3      (TR>EM) 0.2528090   180
4         (SC) 0.2514045   179
5 (FE)-(FE>EM) 0.2289326   163

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


Graphique des sous-séquences les plus fréquentes

Appliquer `plot()` sur l'objet retourné par `seqefsub()`

- Préciser les indexes (`[1 :15]`) pour sélectionner les sous-séquences à inclure (les sous-séquences ordonnées par ordre décroissant des fréquences).
- Les autres arguments sont passés à la fonction `barplot()`

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



Trouver les sous-séquences les plus discriminantes

- But : identifier, parmi les sous-séquences fréquentes, celles qui sont le plus fortement liées à un facteur donné.
- Pouvoir discriminant évalué avec *p*-value du Khi-2 du test d'indépendance.

- Fonction `seqecmpgroup()`
- à laquelle on donne l'objet sous-séquences et une covariable (`group=gcse5eq`).
- Utiliser `method="bonferroni"` pour *p*-values avec correction de Bonferroni.

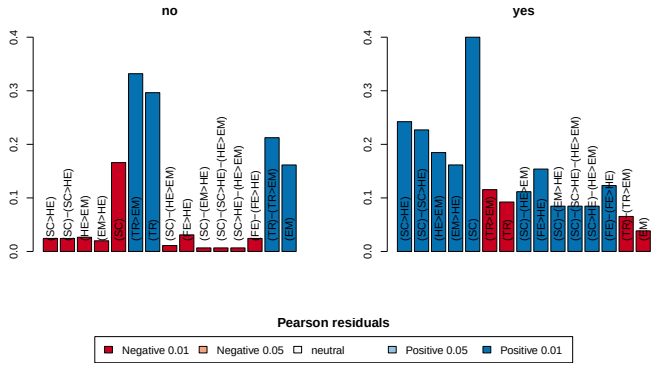
Recherche des sous-séquences discriminantes

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

```
   Subsequence    Support      p.value statistic index    Freq.no  Freq.yes
1      (SC>HE) 0.10393258 1.445408e-19  81.88088    11 0.02433628 0.2423077
2 (SC)-(SC>HE) 0.09831461 7.250286e-18  74.14723    13 0.02433628 0.2269231
3      (HE>EM) 0.08426966 7.487216e-13  51.41219    15 0.02654867 0.1846154
4      (EM>HE) 0.07162921 5.019013e-12  47.67954    21 0.01991150 0.1615385
5         (SC) 0.25140449 7.798571e-12  46.81571     4 0.16592920 0.4000000
   Resid.no Resid.yes
1 -5.249117  6.920999
2 -5.016083  6.613742
3 -4.227342  5.573781
4 -4.108312  5.416839
5 -3.624293  4.778657

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

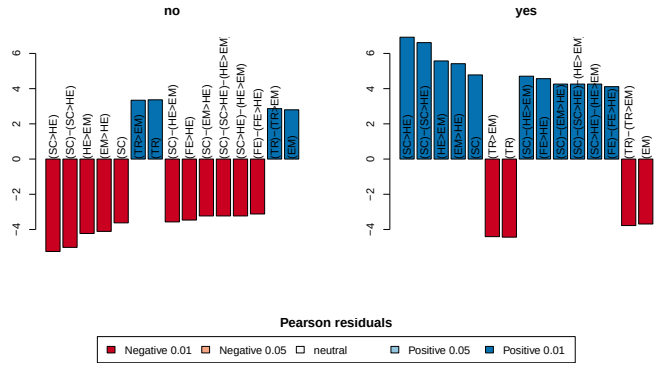

Graphique : fréquences des séquences discriminantes

```
R> plot(mvad.discr[1:15])
```



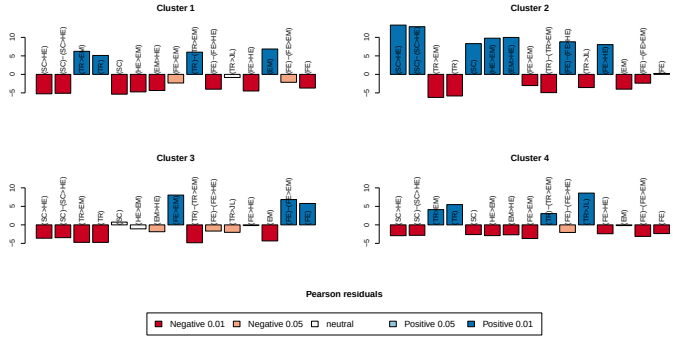
Graphique des ss-séquences discriminantes, résidus

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



Sous-séquences discriminant les clusters

```
R> mvad.discr.cl4 <- seqecmpgroup(mvad.fsubseq, group = cl4.lab)
R> plot(mvad.discr.cl4[1:15], ptype = "resid", legend.cex = 1.3)
```



Conclusion 1 : Etendre l'analyse

- Comme TraMineR est une librairie R, ses sorties peuvent être facilement combinée dans un même script avec d'autres procédures R
- Nous avons vu l'analyse en cluster, ...
- Analyse en coordonnées principales (MDS)
- Dans Widmer and Ritschard (2009), nous avons étudié
  - La relation entre les trajectoires occupationnelles and cohabitationnelles en régressant l'entropie longitudinales de chacune d'elles sur les clusters occupationnels et cohabitationnels tout en contrôlant pour la cohorte de naissance et le sexe.
  - L'appartenance à un cluster avec des régressions logistiques ou des arbres de classification.

Conclusion 2 : A propos TraMineR

- TraMineR est un outil unique et puissant pour séquences discrètes
- Peut faire bien plus que ce qui a été vu, par exemple
  - gestion de séquences, conversion entre événements et d'états
  - données manquantes
  - métrique multi-canal pour séquences parallèles
  - dissimilarités entre séquences d'événements (Studer et al., 2010)
  - extraction de règles d'association entre sous-séquences d'événements (Müller et al., 2010)
  - analyse de la dispersion des séquences (ANOVA et arbre de régression de séquences) (Studer et al., 2010) ...

Où trouve-t-on TraMineR ?

- ... et, comme R, disponible gratuitement sur CRAN
  http://cran.r-project.org
- Voir également la page web de TraMineR
  http://mephisto.unige.ch/traminer

Merci !

References I

Abbott, A. (1997). Optimize. http://home.uchicago.edu/ aabbott/om.html.

Abbott, A. (2001). *Time Matters. On Theory and Methods*. Chicago: Chicago Press.

Berchtold, A. and A. Berchtold (2004). MARCH 2.02: Markovian model computation and analysis. User's guide.

Brzinsky-Fay, C., U. Kohler, and M. Luniak (2006). Sequence analysis with Stata. *The Stata Journal 6*(4), 435–460.

Elzinga, C. H. (2007). CHESA 2.1 User manual. User guide, Dept of Social Science Research Methods, Vrije Universiteit, Amsterdam.

Elzinga, C. H. and A. C. Liefbroer (2007). De-standardization of family-life trajectories of young adults: A cross-national comparison using sequence analysis. *European Journal of Population 23*, 225–250.

Gabadinho, A., G. Ritschard, M. Studer, and N. S. Müller (2009a). 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.

References II

Gabadinho, A., G. Ritschard, M. Studer, and N. S. Müller (2009b). Summarizing sets of categorical sequences. In *International Conference on Knowledge Discovery and Information Retrieval, Madeira, 6-8 October, 2009*, pp. 62–69. INSTICC. (Received the Best Paper Award).

Gabadinho, A., G. Ritschard, M. Studer, et N. S. Müller (2010). Indice de complexité pour le tri et la comparaison de séquences catégorielles. *Revue des nouvelles technologies de l'information RNTI E-19*, 61–66.

Kaufman, L. et P. J. Rousseeuw (2005). *Finding Groups in Data*. Hoboken : Wiley.

Lesnard, L. (2006). Optimal matching and social sciences. Série des Documents de Travail du CREST 2006-01, Institut National de la Statistique et des Etudes Economiques, Paris.

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

Maechler, M., P. Rousseeuw, A. Struyf, and M. Hubert (2005). Package 'cluster': Cluster analysis basics and extensions. Refence manual, R-project, CRAN.

References III

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

Müller, N. S., A. Gabadinho, G. Ritschard, and M. Studer (2008). Extracting knowledge from life courses: Clustering and visualization. In I.-Y. Song, J. Eder, and T. M. Nguyen (Eds.), *Data Warehousing and Knowledge Discovery, 10th International Conference, DAWAK 2008, Turin, Italy, September 2-5*, Volume LNCS 5182 of *Lectures Notes in Computer Science*, pp. 176–185. Berlin Heidelberg: Springer.

Müller, N. S., M. Studer, G. Ritschard, et A. Gabadinho (2010). 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.

Ritschard, G., A. Gabadinho, N. S. Müller, and M. Studer (2008). Mining event histories: A social science perspective. *International Journal of Data Mining, Modelling and Management 1*(1), 68–90.

References IV

Rohwer, G. and U. Pötter (2002). TDA user's manual. Software, Ruhr-Universität Bochum, Fakultät für Sozialwissenschaften, Bochum.

Studer, M., N. S. Müller, G. Ritschard, et A. Gabadinho (2010). Classer, discriminer et visualiser des séquences d'événements. *Revue des nouvelles technologies de l'information RNTI E-19*, 37–48.

Studer, M., G. Ritschard, A. Gabadinho, et N. S. Müller (2009). Analyse de dissimilarités par arbre d'induction. *Revue des nouvelles technologies de l'information RNTI E-15*, 7–18.

Studer, M., G. Ritschard, A. Gabadinho, et N. S. Müller (2010). Discrepancy analysis of complex objects using dissimilarities. In F. Guillet, G. Ritschard, H. Briand, et D. A. Zighed (Eds.), *Advances in Knowledge Discovery and Management*, Studies in Computational Intelligence. Berlin : Springer. (forthcoming).

Widmer, E. and G. Ritschard (2009). The de-standardization of the life course: Are men and women equal? *Advances in Life course Research 14*(1-2), 28–39.