Sequence analysis with TraMineR

Juri Chomé

January 29, 2010

Abstract

TraMineR is an add-on package for R, developed and maintained by the University of Geneva,[2] obtainable from `http://mephisto.unige.ch/traminer/`. The purpose of this paper is to document my use of this package on data describing those medical interventions of working citizens which are paid back by social security. The goal is, to give some examples from which full-fledged calculations might be derived, as well as to do some benchmarking and testing of the current TraMineR package itself. Possible problems with the method itself, depending on the type of data, will be pointed out as well.

1 Goals

2 Introduction

Hard- and software: The computer used was a virtualized x86 DualCore with 2GB RAM running Gentoo Linux. Some but not all operations were able to take advantage of the 2 cores, leading to suspect that a QuadCore would offer little to no gain. An additional 2GB in form of swap proved useless.

The R version used was 2.10.1, TraMineR was at 1.5 at the time of writing. *lapack-atlas* and *blas-atlas* routines were compiled, with the hope of somewhat accelerating calculations. Completion time for the more important parts of the process will be indicated later on.

Hardware, especially RAM, would have to be seriously improved, if any calculations in the realm of 10^5 sequences were to be attempted. According to the TraMineR team, a collection of $4 \cdot 10^4$ sequences is computable on 4GB RAM systems with swapspace, though $7 \cdot 10^4$ sequences already generate a distance matrix taking up around 37GB. In addition to that, a 64bit operating system is mandatory for large calculations, especially since R will be unable to allocate sufficient memory for matrices above 1.2GB. Clustering with 17'000 agents was impossible to achieve.

3 Data specifics

Biofam: In order to gain familiarity with TraMineR, the included *biofam* data set was used as a first playground. The data consists of sequences describing the life course of 2000 individuals during 16 years, starting at the age of 15. The various tags include "living with parents", "having left home", "being married", "having children" and various combinations thereof, for a total of 8 possible states.

Diagnosis data: A database was compiled from 3 years worth of diagnoses of work-leave illness. Starting in 2006 with a resolution of 1 day, this spans a (potential) sequence length of 1096 units. This is a first inconvenience, and its wise to reduce the maximum length to about 100, effectively reducing most calculations by a factor of $^9/_{10}{}^{th}$ according to algorithm size. The various diagnoses were grouped into 20 categories, another slight hurdle for the sequence analysis itself, as this necessitates almost the same number of clusters in order to reveal anything useful.

4 Math

Besides some basic grouping and visualization, the interesting application of sequence analysis comes with clustering methods, requiring thus a feasible metric, the choice of which has to be justified in some way, mathematically or from a practical point of view. Additionally, there should be a way to judge homogeneity of a given group of sequences.

4.1 Metrics

In terms of defining algebraic similarity for sequences, there are roughly four choices to consider: *"Longest Common Prefix" (LCP)*, its reverse *"Longest Common Suffix" (RLCP)*, *"Longest Common Subsequences" (LCS)* and *"Optimal Matching" (OM)*. The following is a quick overview of the different metrics, illustrated in much more detail in "Metric Representations of Categorical Time Series".[1]

Definitions: Let x and y be two string over a given alphabet, whose elements are sometimes called states. The length of x, denoted $|x|$ is its number of non-empty letters. We want to define an *attribute* $A(x, y)$ which is symmetrical and fulfills

$$0 \leq A(x, y) \leq min\{A(x, x), A, y, y)\}.$$

We define the associated *similarity* as follows:

$$s(x,y) := \frac{A(x,y)}{\sqrt{|x| \cdot |y|}}.$$

The (scaled) distance function is then given by $d(x,y) := 1 - s(x,y)$

LCP and RLCP

The *LCP* metric is now given by setting $A_{lcp}(x,y)$ to be the length of the longest common prefix of x and y. *RLCP*, by analogy, has as attribute the length of the longest common suffix, hence the name *"Reverse LCP"*. The concept of the *longest common subsequence* is trickier and requires the additional notions of *substrings* and *subsequences*:

LCS

Definitions: A *substring* of a string x is another string u, such that by concatenation, we have $x = aub$, for some strings a and b. More generally, we say, that u is a *subsequence* of x, if u can be obtained from x by deleting a number of letters in x. The consequence is, that the letters of u appear in the same order inside x, but are not necessarily consecutive in x.

With $S(x,y)$ the set of common subsequences of x and y, we define the attribute to be

$$A_{lcs}(x,y) := \max\{|u| : u \in S(x,y)\}.$$

Similarity is then defined as above, but for the distance function, we set $d_{lcs} := A_{lcs}(x,x) + A_{lcs}(y,y) - 2A_{lcs}(x,y)$. There is a simple recurrence formula, giving the solution in time proportional to mn, m and n being the length of x and y respectively.

OM

Given two sequences x and y, we can define *substitution* and *insertion-deletion* costs, associated to substituting, deleting or inserting a state in a sequence. The idea is, to fix two matrices, one for indel and the other one for substitution, the question remaining: how? The general consensus for indel operation costs is, to fix them at 1 for any state. The question of the substitution cost matrix is trickier, and subject to much discussion, especially in social sciences, where the cost of an action is difficult to define.

Fortunately, TraMineR is able to compute these costs, by looking at the transition frequency, thus the associated probability in a given data set. The substitution matrix is therefor computed anew, for every data set. A more detailed view on this, is given by Laurent Lesnard.[3] Even though *OM* is

quite prone to criticism, the only important criteria should the homogeneity of the resulting clusters and with it, the usefulness of the resulting grouping. To further the pragmatic goals, the notion of entropy is useful.

4.2 Entropy

To measure the amount of noise in a given group of sequences, the entropy at a given point in tine t can be defined. For every state j, let $p_j(t)$ be the proportion of sequences whose x_t equals j, i.e the proportion of people in state j at the moment t. The entropy of a set of sequences at a point t is given by

$$H_t := -\sum_j p_j(t) \cdot \ln(p_j(t)).$$

Note that we have to leave out a state j if it's not represented in a sequence at t (i.e its proportion being zero).

We now have a tool to measure how heterogeneous a set of sequences is at a given time, with minimum 0 if only one state is represented and maximum $\ln(q)$ if all q states are represented and equally distributed.

5 Examples

The *biofam* data set can be used, in order to gain some elementary understanding about the implications of the different metrics. Calculation times are more or less trivial, the only operations taking more than a few seconds being *seqdef()* and *seqdist()*, functions responsible for generating the sequence from unformatted data and for computing the distance matrix respectively.

Figure 1 shows an overview of the *biofam* dataset, displaying some raw sequences and the overall distribution of the set using *seqiplot()* and *seqdplot()* respectively. Note that TraMineR offers a variety of other plotting functions which will be omitted here.

Once the sequence data is defined, TraMineR is able to compute a distance matrix, using the *seqdist()* function, whose *method* argument specifies the type, *OM* requiring an additional substitution-cost matrix, provided by *seqsubm()*. Using the *cluster* library, agglomeration nesting can then be applied to the distance matrix, the results of which are depicted in figure 2.

In this case, optimal matching generates the more balanced clusters, with *LCS* coming in as a close second. This is indeed reflected when comparing the distributions of the generated clusters for each method (see the *biofam.r*

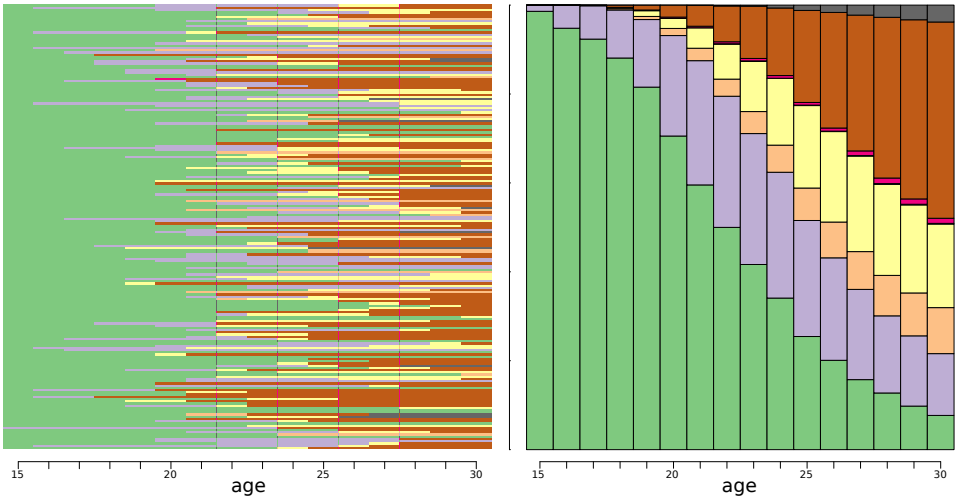

Figure 1: 200 *biofam* sequences and total distribution

script). It can be safely said, that *OM* provides the most homogeneous results, both in cluster size, as well as in terms of entropy. In figure 3, an example of separation into 10 clusters can be seen, with cluster size varying between 50 and 450 members. Interpretation of the data by describing the meaning of the different groupings can be done with relative ease at this point.

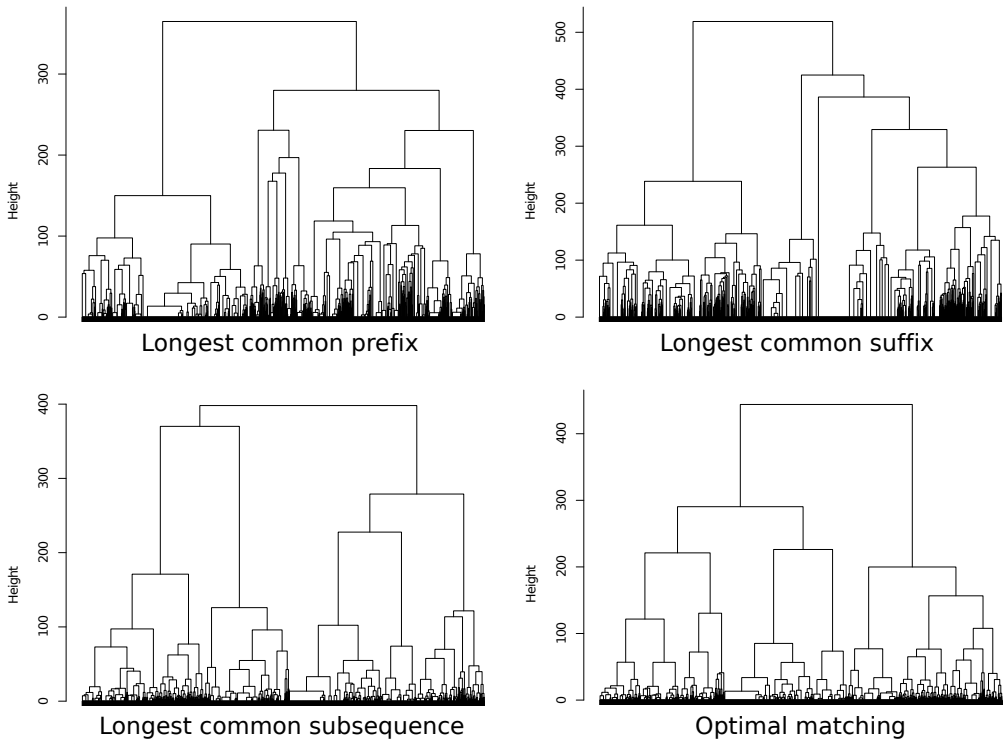

Figure 2: Agnes dendrodrams using different metrics

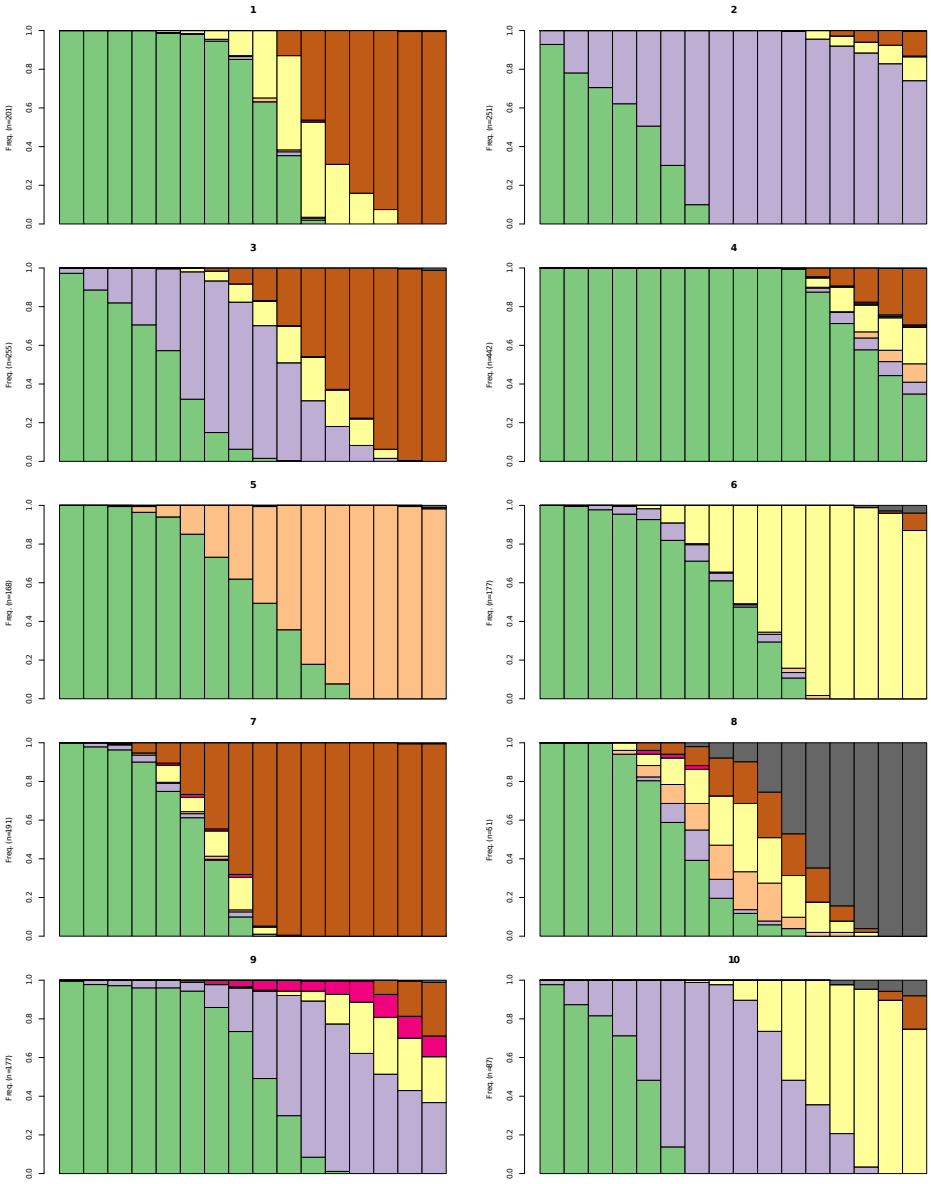

Figure 3: Distribution of 10 groups provided through *OM*

6 The diagnosis database

6.1 First attempts and observations

Since the whole dataset contains 732′244 rows, generating a data frame of 140′760 sequences, each having a length of 1096 days, the information has to be reduced in order to be somewhat manageable. The data was provided in SPELL format, having one entry for each registered intervention by a doctor. The following excerpt gives a general idea (identification column left out):

debut_episode	fin_episode	**diag**_code	natio	eciv	naiss	icd_code	id_ep
2006−08−01	2006−08−13	61	L	1	1947	L	7
2006−01−26	2006−01−27	99	L	5	1947	S	1
2006−02−01	2006−02−05	99	L	5	1947	S	2
2006−02−03	2006−03−12	3	P	2	1947	A	1
2008−09−02	2008−09−02	NA	A	2	1947	NA	1
2008−09−09	2008−09−21	5	A	2	1947	**C**	2
2008−09−03	2008−09−13	29	L	6	1947	E	3
2006−07−03	2006−07−14	10	P	2	1947	**D**	1
2006−07−17	2006−07−27	45	P	2	1947	**I**	2

Commands like *summary(data)* and *table(data$naiss)* can give some more quick information about distribution, min/max of columns and the like. TraMineR can handle SPELL data quite well, though some further manipulations are necessary.

The year of 1954

As a first idea to reduce the data, the year 2008 was singled out, everybody born 1954 was chosen, and converted into sequence data, filling the gaps between the episodes with an additional "work" state. This resulted in 1617 individual sequences with a constant length of 366 (days). Clustering was fairly quick to compute at that point, but turned out to be messy, as work proved to be much too dominant, thereby eclipsing any subtle effects that the other states might have had.

The agglomeration dendrogram in figure 4 gives a good indication of the problem: There are a few very dominant branches, but the splitting on the far right happens much too far down, with too much noise coming from the left part. The only relevant groups thus singled out, were those almost never sick, the constantly ill and a group of chronic ill. Bigger data might have allowed to take a closer look at the ill groups, but it proved more efficient, to just forget about work periods. Distribution graphs with an attempt at clustering may be found in the *naiss1954-ann2008* folder.

Figure 4: Dendrogram for the birthyear of *1954*

The year of 1977

Leaving out the "work" state results in, what TraMineR knows as *process* based sequence, meaning that the sequences themselves are not necessarily situated in the same time frame, but are aligned by their first event. Even though all work periods will be left out here, there is also the possibility to remove only the periods in the beginning, the middle or at the end.

On a technical level, TraMineR requires an additional "birth" data frame, providing each identifier with a start date. The sequence are then left-aligned, starting at the same point of reference. This time, the whole three years were selected, giving rise to a total of 4275 individuals, with min/max length being 1/98. Results were encouraging enough, that an attempt on the real dataset was immediately started. Calculations and pictures of the 1977 subset can be found in the *1977* sub-folder in *test_itt*.

Problems with ignoring work periods

The first problem with this is, that we might introduce similarity where none is desired. Two sequences, each containing two distinct diagnoses each would be considered very similar, even if one person had them in quick succession, and the other one had the episodes spread over all three years. In this case, the differences have to be considered insignificant out of computational necessity. The fact, that the time-window is restricted somewhat helps here. Either way, this fact should be kept in mind, when offering conclusions based on the resulting data.

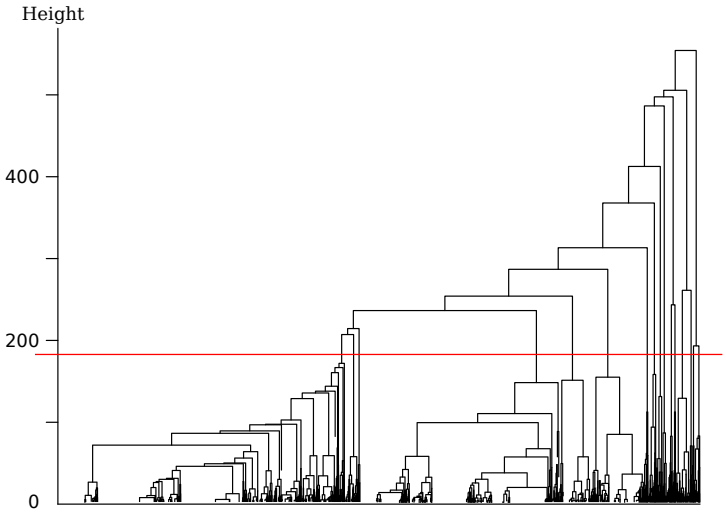

Figure 5: A good height to cut the dendrogram of the *1977* set

The more general problem revolves around the alignment of the sequences. How can we even compare everybody at the same time? Here, only people from one birth year are chosen, which circumvents this issue on the big scale, but the fact remains, that we align the trajectories based on the completely random variable of the first diagnosis registered in 2006. What if everybody had their interventions in the exact same order, but shifted by some months, some earlier, some later? This especially kind of excludes *LCP* and *RLCP* right from the beginning. The good news is, that *OM* (and *LCS* for that matter) are quite stable in that regard, especially since *indel* costs are low in comparison to substitution costs.

In addition to that, the large sample size helps, and the results shown in the next section indicate, that the groupings provided seem to be quite agnostic to noise. Many groups in figure 11 shows exactly this kind of small, random conditions in the beginning, slowly fading away over time. The overall picture remains thus untainted by the random illnesses at the starting point.

6.2 The entire set

The *seqdef()* procedure took 23 hours to complete. The resulting frame contained 140′760 sequences (137′273 after cleaning out the empty ones), with min/max length conveniently being 1/100. It was clear from the start, that clustering couldn't be performed on this set. As a result, the difference between long-term and short-term total intervention period was made, singling out every sequence longer than 20 days. This left 17′765 people, 7′777 of

which had distinct trajectories, min/max length being naturally at 21/100.

From this point, one can plot the state distributions to gain an overview of the data, already revealing some interesting features. For instance, there is a noticeable jump between day 21 and 22 due to stage 18 ("cures thermales") diminishing almost instantly in proportion.

More importantly, state distribution seems to alter significantly after the split into two disjoint groups. If figure 7 is compared to figures 8 and 9, there is already a huge difference in initial distribution. Observe, that despite our concern with noise, the main middle part of the distribution is remarkably consistent.

Another unsurprising observation is, that short-term illnesses like infections are in general dominant at the start, but quickly fade away, as long-term conditions like traumata and the degeneration of joints and the skeletal system. So if we hadn't known before, we now suspect that depressions are a major long-term cause of concern which we might want to take a closer look at.

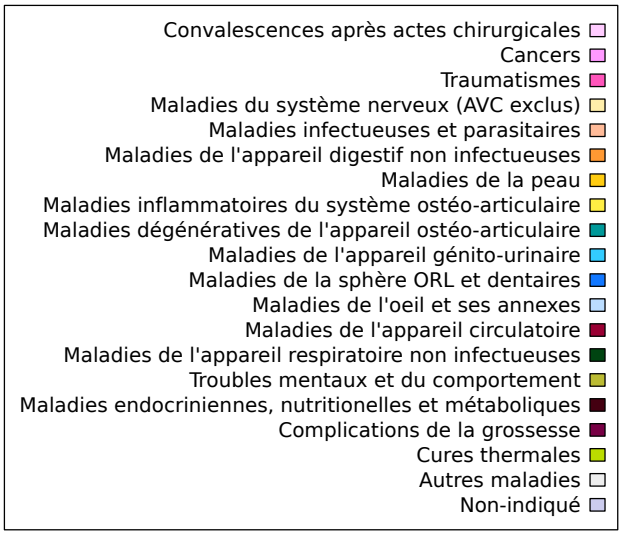

Figure 6: This might be relevant

After getting allocation errors in R, the realization was made, that a 64bit operating system should have been installed. In order to attempt clustering anyway, the data was further reduced in the most unscientific way of simply selecting the first 5′000.

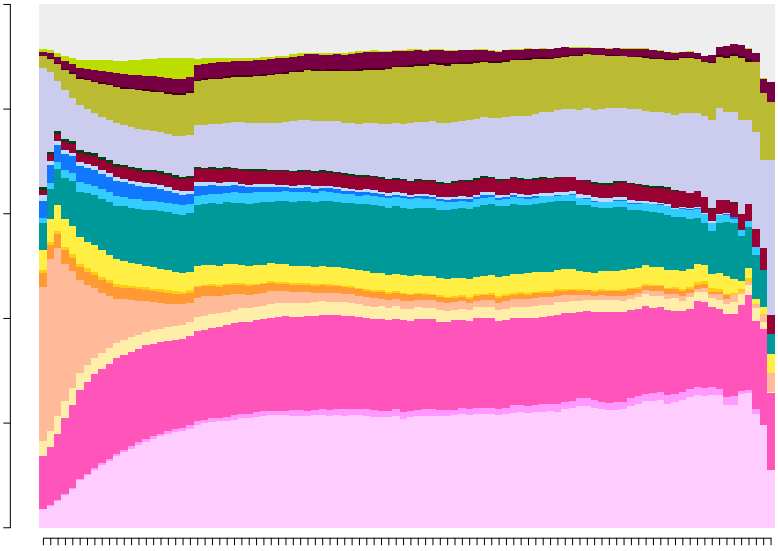

Figure 7: State distribution plot for all 137'273 sequences

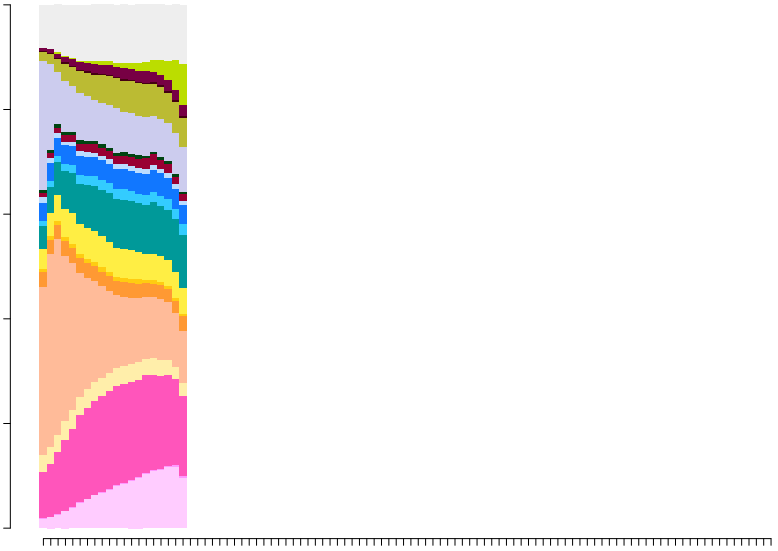

Figure 8: Distributions for sequences shorter than 21 days

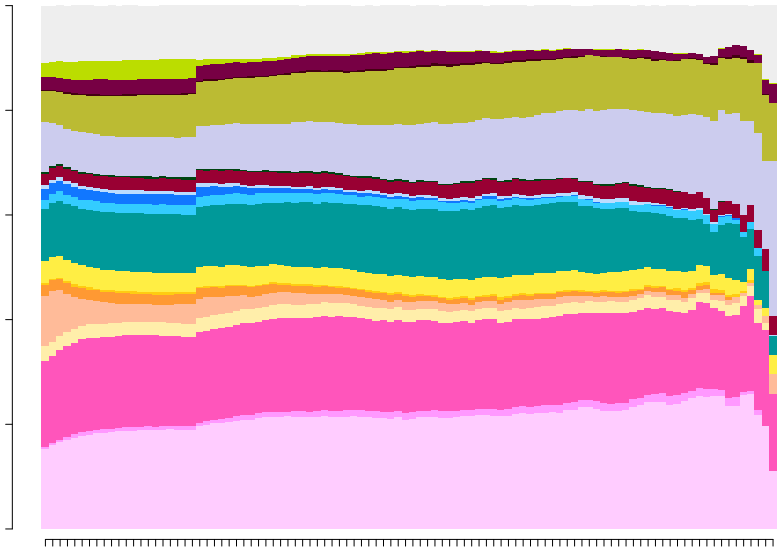

Figure 9: Distribution for sequences longer than 21 days

Although both *longest common prefix* and *optimal matching* were tried, results of *LCP* will be omitted here, due to *OM* significantly outperforming it. Figure 10 shows its cluster-graph, which was cut to obtain 17 clusters. Cluster 11 was merged into number 4 to produce a more plot-friendly number of groups. Looking at the distribution plot, a fairly uniform picture is

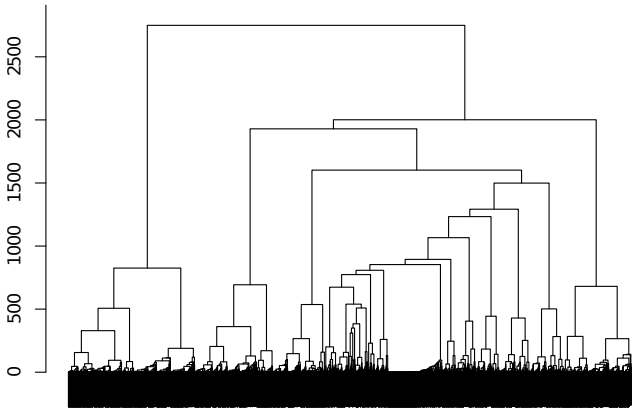

Figure 10: Agnes with *OM* on the reduced set

revealed. Most conditions are grouped into their own cluster, with very little noise. However, in order to judge this adequately, the number of sequences at each point in time is absolutely necessary. Figure 12 provides this information. The following table indicates, what the member count is for each of the 16 groups:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
580	553	465	166	215	209	542	119	304	267	277	174	479	391	139	120

Again, several interesting observations offer themselves based on figures 12 and 12. For instance, according to group 13, state 18 seems to indeed come to a sudden stop after day 21. The associated "degenerative" condition in the distribution plot might be relevant, however it has to be noted that the proportion if people concerned at that point is fairly small.

Those clusters, corresponding to the top 5 states of the distribution, seem to be fairly distinct, consistent in themselves and low in noise. Most of them don't lead from one dominant state into another one, with clusters 7 and 14 being a notable exception. Remember that cluster 4 has to be disregarded in that aspect, as it's an artificial mixture of two smaller groups.

In order to inquire about the relevance of the later mutations of cluster 7 and 11, its important to look at the sequence distribution provided by figure 12. This reveals, while later states are introduces, that were not so

strongly present at the beginning, the sequence length decreases especially fast, leading to the conclusion, that the number of people concerned is almost negligible.

Interestingly, cluster 1 and 7 describe the same dominant state, with the only important difference being, that the overall sequence length is completely different. It's thus a distinction made, between long-term and short-term surgical complications. This property seems to be relevant enough, to show through the clustering. If The number of groups were to be increased, one might observe the same behaviour with other states.

The remaining observations are left as an exercise to the reader.

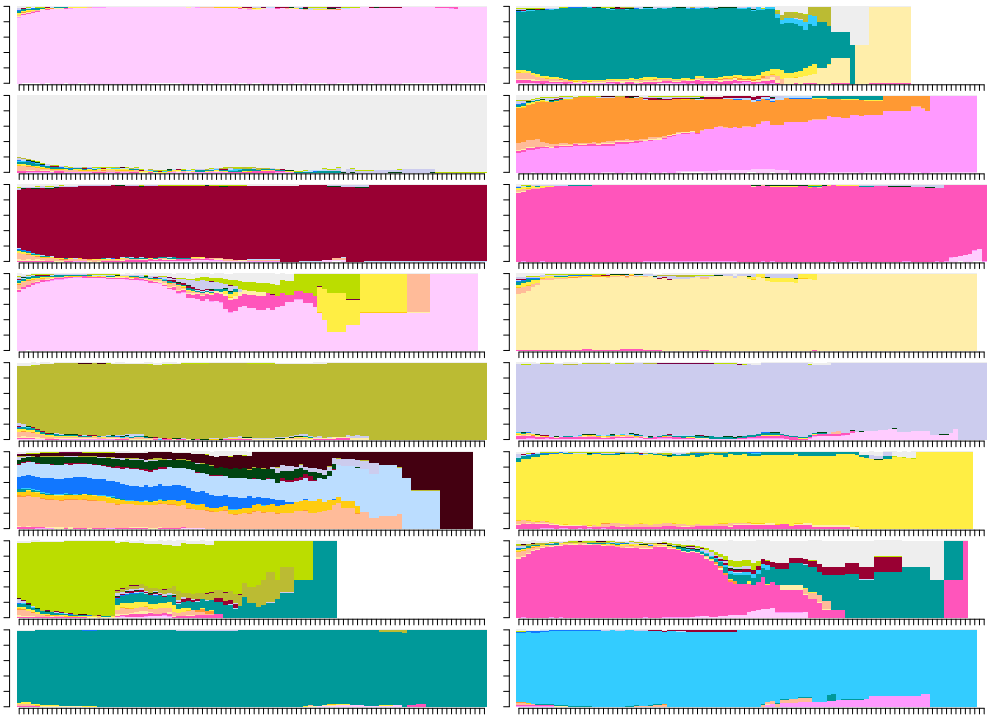

Figure 11: Distribution per cluster

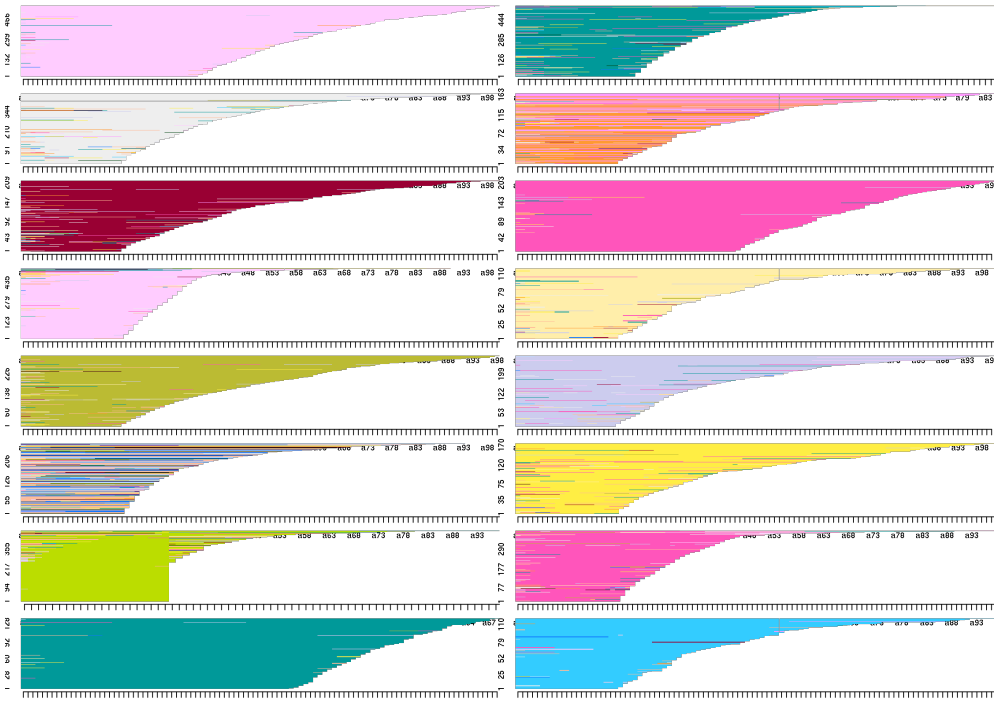

Figure 12: All grouped sequences, sorted by length

The associated Entropy plot, shows, that most of the 16 groups seem to be fairly uniform, with some exceptions. Comparing the high-entropy graphs to their respective distribution plots leads to expect, that the concerned groups would have split into smaller ones if the number of clusters would have been chosen to be bigger. Entropy confirms some intuitions one might have when just presented with the distribution plots.

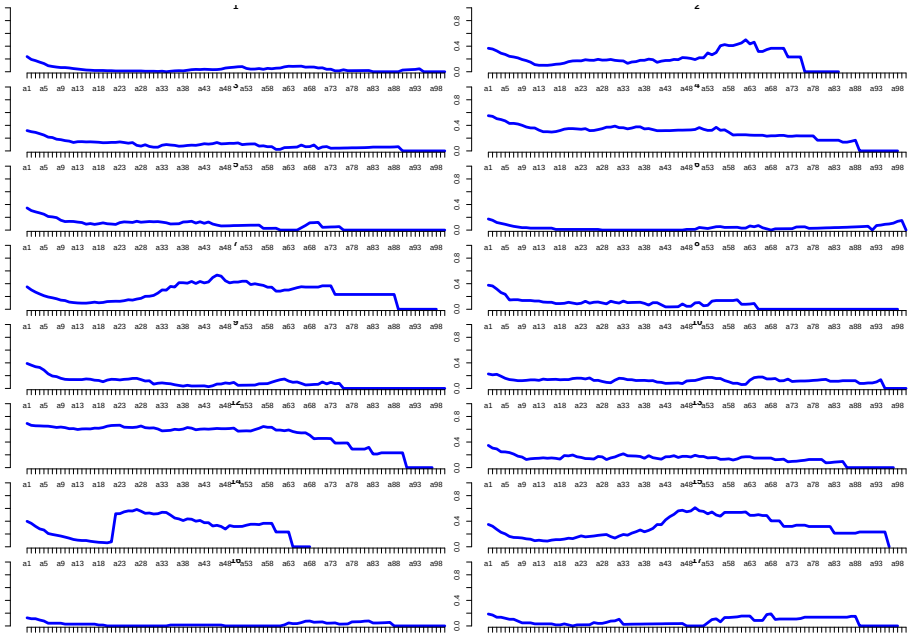

Figure 13: Shannons entropy in each group

7 Conclusions and further steps

Even with the present subsection of the data, the whole potential of interpretation has yet to be tapped. Some obvious starting points would be, to associate age distribution to each of the clusters, yielding some expected and hopefully some unexpected results. The same could be done with gender and all other kinds of data associated to the people represented in the sequences. Once the clustering is done at a good resolution, the question of correlation should be sufficiently resolvable.

The next logical step would be to consider adding a maximum of data into the set, possibly spanning much more than the three years represented here. If the big number-crunching hardware is not available, the number of days

could be condensed, depending on how much precision can be sacrificed. It is however possible, that adding a gargantuan number of states will not give significant change in usefulness of the obtained results, being mainly associated to the fact, that calculating with a huge amount of information is a painful procedure to both man and machine.

Having done that, provided that all the calculations were feasible, the number of clusters can be increased, to reveal even finer detail.There is of course a limit to that, being that for clusters too small in membership size, the noise starts to show through, and the distinctions become less and less relevant. Since distribution of sequence length seems to be more or less follow a power law, it probably makes sense to consider manually splitting up sequences of different length like it was done here, since it considerably reduces the sample size while focusing on the relevant aspects one might be looking for. In our case, it is natural to consider long sequences of special interest, following through with more persistent conditions.

References

[1] Cees H. Elzinga. *Sequence Analysis: Metric Representations of Categorical Time Series*. Vrije Universiteit Amsterdam, 2007.

[2] Alexis Gabadinho, Gilbert Ritschard, Matthias Studer, and Nicolas S. Müller. *Mining Sequence Data in R with TraMineR: A User's Guide*. Geneva, 2009.

[3] Laurent Lesnard. Cost setting in optimal matching to uncover contemporaneous socio-temporal patterns. 2009.