Using `traj` Package to Identify Clusters of Longitudinal Trajectories

Marie-Pierre Sylvestre*& Dan Vatnik†

November 25, 2014

Abstract

The `traj` package implements the 3-step procedure proposed by Leffondre et al. (2004) to identify clusters of longitudinal trajectories. The first step calculates 24 summary measures that describes features of the trajectories. The second step performs a factor analysis on these 24 measures to select measures that best describenthe main features of the trajectories. The third step classifies the trajectories into clusters based on the previously selected factors. The `traj` package also offers a wide variety of plotting function used to visualize the results.

This vignette illustrates the use of the `traj` package using simulated data. A more detailed description of the methods can be found in Sylvestre et al. (2006) or Leffondre et al. (2004).

1 Data

Data consist in two dataframes. The first dataframe, `example.data$data`, contains the values for each individual trajectory. Each row correspond to a trajectory.

```
library(traj)
head(example.data$data)
```

```
  ID     X1     X2     X3     X4     X5     X6
1  1  5.659  9.340  3.770 17.361  8.824  9.281
2  2 23.593 11.752  7.684 12.830 13.002  9.665
3  3 15.469  8.756  6.493 11.261 10.420 17.405
4  4  7.312 11.688 12.476  8.890  6.522  7.701
5  5 12.844 11.088  7.650 10.269 12.453 11.557
6  6  3.522 15.285  7.860  7.114 17.954  4.168
```

The second dataframe, `example.data$time`, contains the time points at which the corresponding values for each individual trajectory were measured. Both dataframes must be of the same dimension.

```
head(example.data$time)
```

```
  ID time.1 time.2 time.3 time.4 time.5 time.6
1  1      1      2      3      4      5      6
2  2      1      2      3      4      5      6
3  3      1      2      3      4      5      6
4  4      1      2      3      4      5      6
5  5      1      2      3      4      5      6
6  6      1      2      3      4      5      6
```

*Department Social and Preventive Medicine, Universit? de Montr?al, CHUM Research Centre
†Statistical Programming, CHUM Research Centre

2 Analysis

The first step in the analysis consists of the computing 24 measures of each trajectory.

The 24 measures are:

1. Range
2. Mean-over-time
3. Standard deviation (SD)
4. Coefficient of variation (CV)
5. Change
6. Mean change per unit time
7. Change relative to the first score
8. Change relative to the mean over time
9. Slope of the linear model
10. R^2: Proportion of variance explained by the linear model
11. Maximum of the first differences
12. SD of the first differences
13. SD of the first differences per time unit
14. Mean of the absolute first differences
15. Maximum of the absolute first differences
16. Ratio of the maximum absolute difference to the mean-over-time
17. Ratio of the maximum absolute first difference to the slope
18. Ratio of the SD of the first differences to the slope
19. Mean of the second differences
20. Mean of the absolute second differences
21. Maximum of the absolute second differences
22. Ration of the maximum absolute second difference to the mean-over-time
23. Ratio of the maximum absolute second difference to mean absolute first difference
24. Ratio of the mean absolute second difference to the mean absolute first difference

The 24 measures can be computed using the step1measures function.

```
s1 = step1measures(example.data$data, example.data$time, ID = TRUE)
```

```
[1] "Correlation of m5 and m6 : 1"
[1] "Correlation of m12 and m13 : 1"
[1] "Correlation of m17 and m18 : 0.999"
```

```
head(s1$measurments)
```

```
  ID     m1     m2    m3    m4       m5       m6       m7       m8       m9
1  1 13.590  9.039 4.661 51.57   3.6225  0.60376  0.64015  0.40076  0.861616
2  2 15.909 13.088 5.534 42.28 -13.9279 -2.32131 -0.59035 -1.06421 -1.735574
3  3 10.912 11.634 4.107 35.30   1.9365  0.32275  0.12519  0.16645  0.555447
4  4  5.955  9.098 2.447 26.90   0.3893  0.06488  0.05324  0.04279 -0.489632
5  5  5.194 10.977 1.875 17.08  -1.2863 -0.21438 -0.10015 -0.11718  0.008112
6  6 14.432  9.317 5.955 63.91   0.6457  0.10761  0.18333  0.06930  0.299663
        m10    m11    m12    m13   m14    m15    m16     m17     m18    m19
1 1.196e-01 13.590  8.656  8.656 6.367 13.590 1.5035  15.773  10.047 -0.806
2 3.442e-01  5.146  6.237  6.237 4.913 11.841 0.9047  -6.822  -3.594  2.126
3 6.402e-02  6.985  5.515  5.515 4.314  6.985 0.6004  12.576   9.929  3.425
4 1.401e-01  4.376  3.146  3.146 2.460  4.376 0.4809  -8.936  -6.426 -0.799
5 6.549e-05  2.619  2.598  2.598 2.179  3.438 0.3132 423.805 320.306  0.215
6 8.864e-03 11.763 11.197 11.197 8.912 13.786 1.4797  46.006  37.367 -6.387
     m20    m21    m22   m23   m24
1 14.883 22.127 2.4479 3.475 2.338
2  6.367  9.214 0.7040 1.876 1.296
3  6.229  7.826 0.6727 1.814 1.444
4  3.182  4.374 0.4808 1.778 1.294
5  2.813  6.057 0.5518 2.780 1.291
6 15.520 24.626 2.6431 2.763 1.741
```

Each row in the dataframe returned by step1measures corresponds to the trajectory on the same row in the input data (`example.data$data`). For each trajectory, the 24 measures have been calculated and correspond to columns m1 to m24.

In the second step of the analysis, a factor analysis is performed to select a subset of measures that describes the main features of the trajectories. The function step2factors is used to perform the factor analysis.

```
s2 = step2factors(s1)
```

```
[1] "m6 is removed because it is perfectly correlated with m5"
[2] "m13 is removed because it is perfectly correlated with m12"
[1] "Computing reduced correlation e-values..."
```

```
head(s2$factors)
```

```
  ID    m4       m5    m21   m24
1  1 51.57   3.6225 22.127 2.338
2  2 42.28 -13.9279  9.214 1.296
3  3 35.30   1.9365  7.826 1.444
4  4 26.90   0.3893  4.374 1.294
5  5 17.08  -1.2863  6.057 1.291
6  6 63.91   0.6457 24.626 1.741
```

In this example, the step2factors has identified measures 4, 5, 21 and 24 as the main factors of this set of trajectories. Measures 6, 13 and 18 were not considered because they were too correlated with other measures (measures with a correlation higher than 0.95 are omitted from the factor analysis).

Once this step is done, the third step of the procedure consists in clustering the trajectories based on the measures identified in the factor analysis. This step is implemented in the step3clusters function. Two options are available to select the number of clusters. First, the user can a priori decide on the number of clusters, such as in the following example in which the number of clusters is set to 4.

```
s3 = step3clusters(s2, nclusters = 4)
```

Alternatively, the number of clusters can be left blank in which case the step3clusters function will rely on the `NbClust` function from the `NbClust` package to determine the optimal number of clusters based on one of the criteria available in `NbClust`. Please see `NbClust` documentation for more details.

The function step3clusters assigns each trajectory to one and only one cluster and returns a dataframe that identifies cluster membership.

```
head(s3$clusters)
```

```
  ID cluster
1  1       2
2  2       2
3  3       2
4  4       2
5  5       2
6  6       2
```

```
s3$clust.distr
```

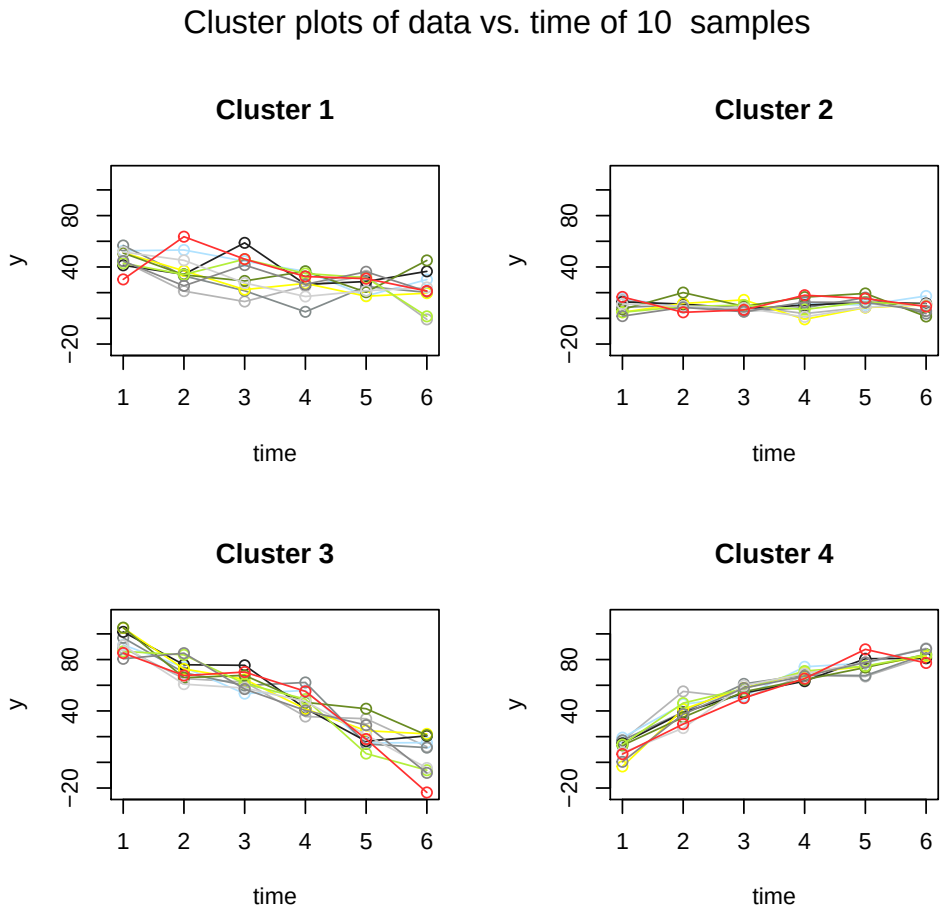
```
 1  2  3  4
25 55 40 10
```

The `traj` object returned by the function step3clusters can be plotted by an array of plotting functions, as described in the next section.

3 Plotting the traj object

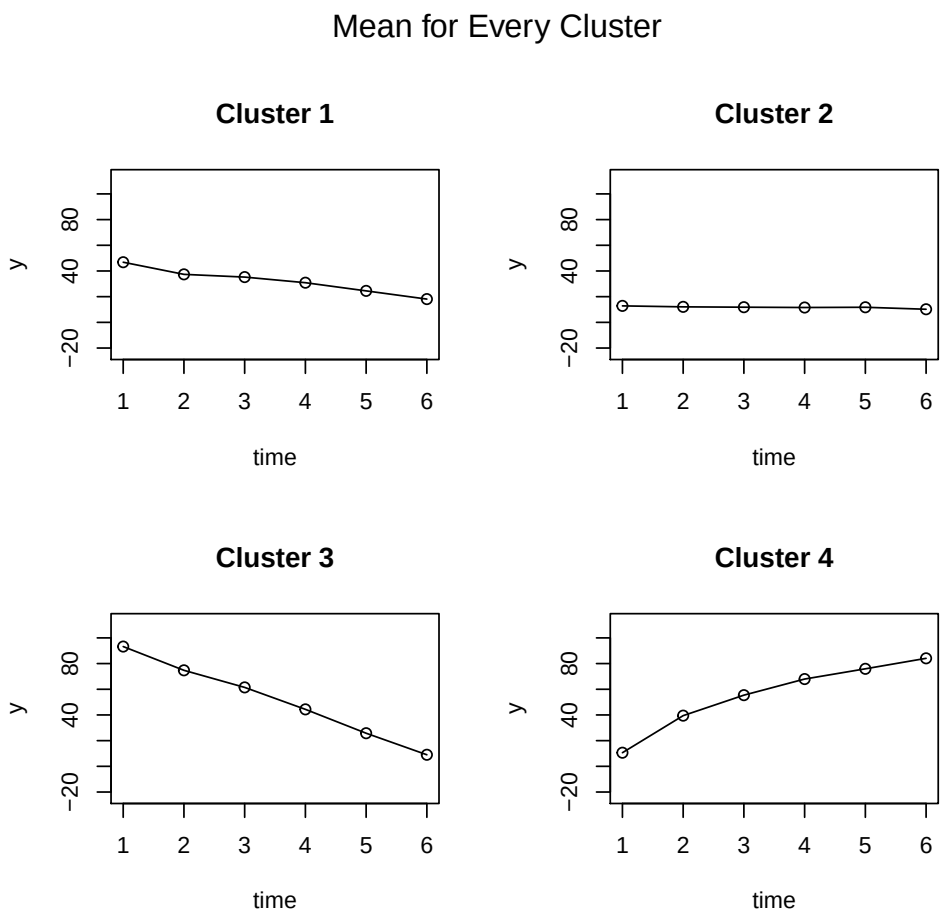
The `traj` object created by `step3clusters` can be plotted by an array of plotting functions.

```
plot(s3)
```



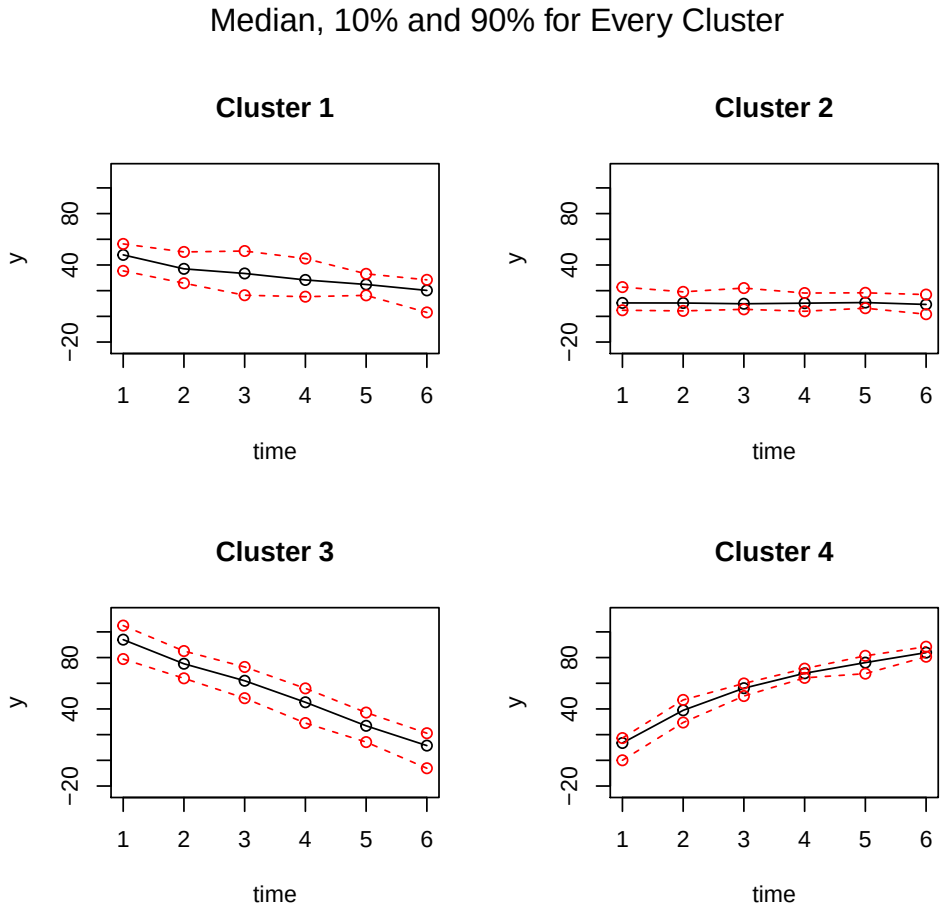
This function selects 10 random trajectories from each cluster and plots them using randomly selected colours. The user can specify the number of trajectories to plot, the colours or any other generic plotting parameter. The user can request that trajectories from only one cluster be plotted.

plotMeanTraj(s3)



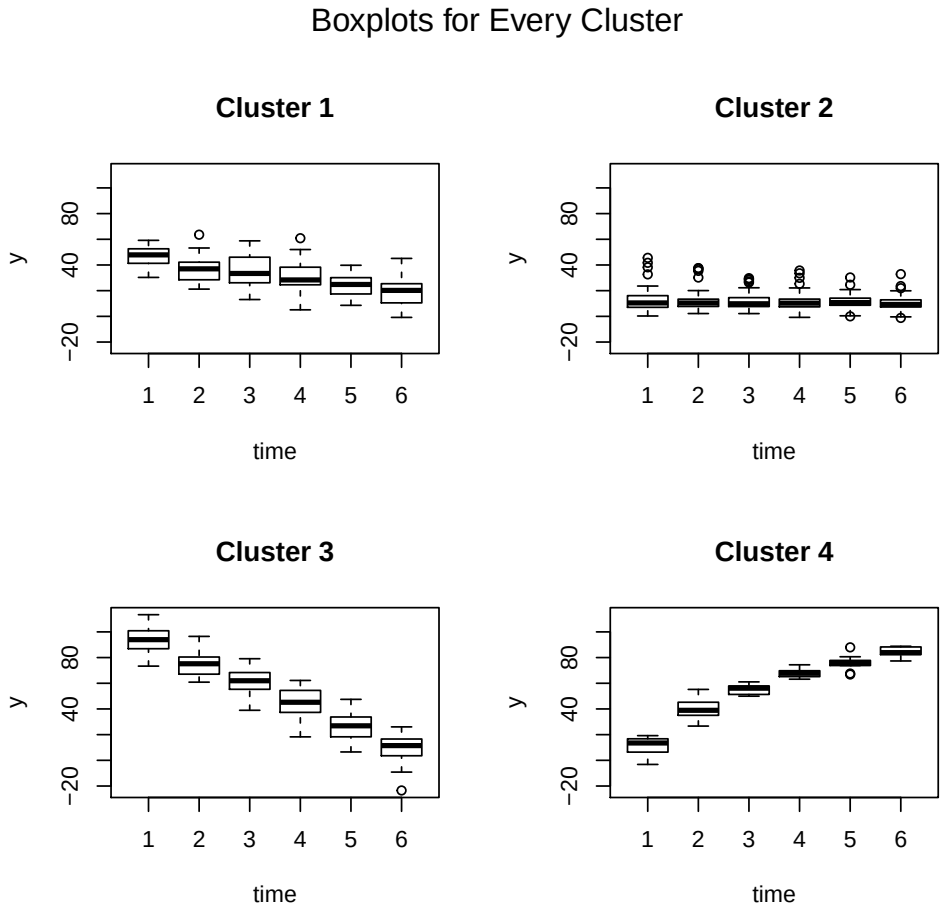
The `plotMeanTraj` function plots the mean trajectory of every cluster. The user can request that trajectories from only one cluster be plotted.

```
plotMedTraj(s3)
```



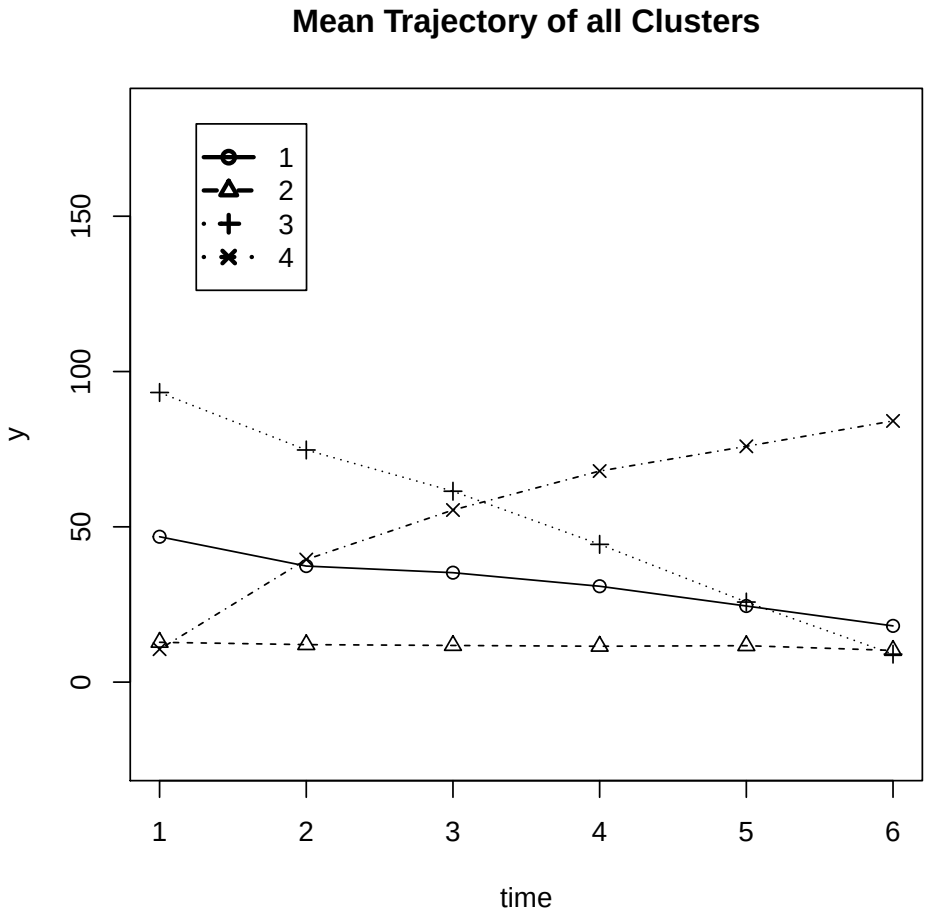
The `plotMedTraj` function plots the median trajectory of every cluster with 10th and 90th percentiles. The user can request that trajectories from only one cluster be plotted.

plotBoxplotTraj(s3)



The `plotBoxplotTraj` function will plot the box-plot distribution of every time point in each cluster. The user can request that trajectories from only one cluster be plotted.

plotCombTraj(s3)



The `plotCombTraj` function will plot the mean or median of all the clusters on one single graph. Different colours can be selected as well as different line styles.

References

(a) Sylvestre MP; et al. (2006). Classification of patterns of delirium severity scores over time in an elderly population. International Psychogeriatrics; 18(4); 667-680. doi:10.1017/S1041610206003334.

(b) Leffondree; K. et al. (2004). Statistical measures were proposed for identifying longitudinal patterns of change in quantitative health indicators. Journal of Clinical Epidemiology; 57; 1049-1062. doi : 10.1016/j.jclinepi.2004.02.012.