Clustering time series data with tscR

A R package to cluster time series data or similar longitudinal data, base on slope and Frechet distance

Pérez-Sanz, Fernando. Murcian Institute of biomedical research
Riquelme-Pérez, Miriam. CNRS - CEA, Univ. Paris-Saclay. MIRCen

Contents

Overview

En algunos estudios disponemos de medidas repetidas en el tiempo sobre las mismas muestras, como ocurre en experimentos de expresión génica, con medidas repetidas en el tiempo. En otras ocasiones puede tratarse de medidas sobre las mismas muestras en las que se modifica la intensidad de algún factor (tratamiento, dieta, etc). En tales ciscustancias, tendríamos un caso particular de datos longitudinales, en el que el número de medidas a lo largo del tiempo es relativamente bajo y el número de unidades de medida suele ser único (una sóla variable).

Un ejemplo sería el caso de la expresión génica. Podríamos considerar la expresión de cada gen como un individuo y las variables serían los momentos temporales en los que se ha medido la expresión de dicho gen. El conjunto de mediciones sobre un mismo gen constituiría lo que denominamos trayectoria.

Table 1: 1

	T1	T2	T3
$Traj_1$	140	120	100
$Traj_2$	100	120	140
$Traj_3$	50	30	10
$Traj_4$	10	30	50

En estos casos el investigador puede estar interesado en identificar y agrupar conjuntos de genes con comportamientos similares desde diferentes puntos de vista: (fig.1 B) genes agrupados con niveles de expresión similares (genes con trayectorias cercanas en términos de distancia física), (fig.1 C) genes agrupados con similar evolución independientemente de la "distancia física" a la que se encuentren. Podría tratarse en este

segundo caso de genes que responden de manera similar, pero con diferente intensidad. Además podría ser de interés agrupar genes en función de ambos factores (distancia y tendencia) (fig.1 D).

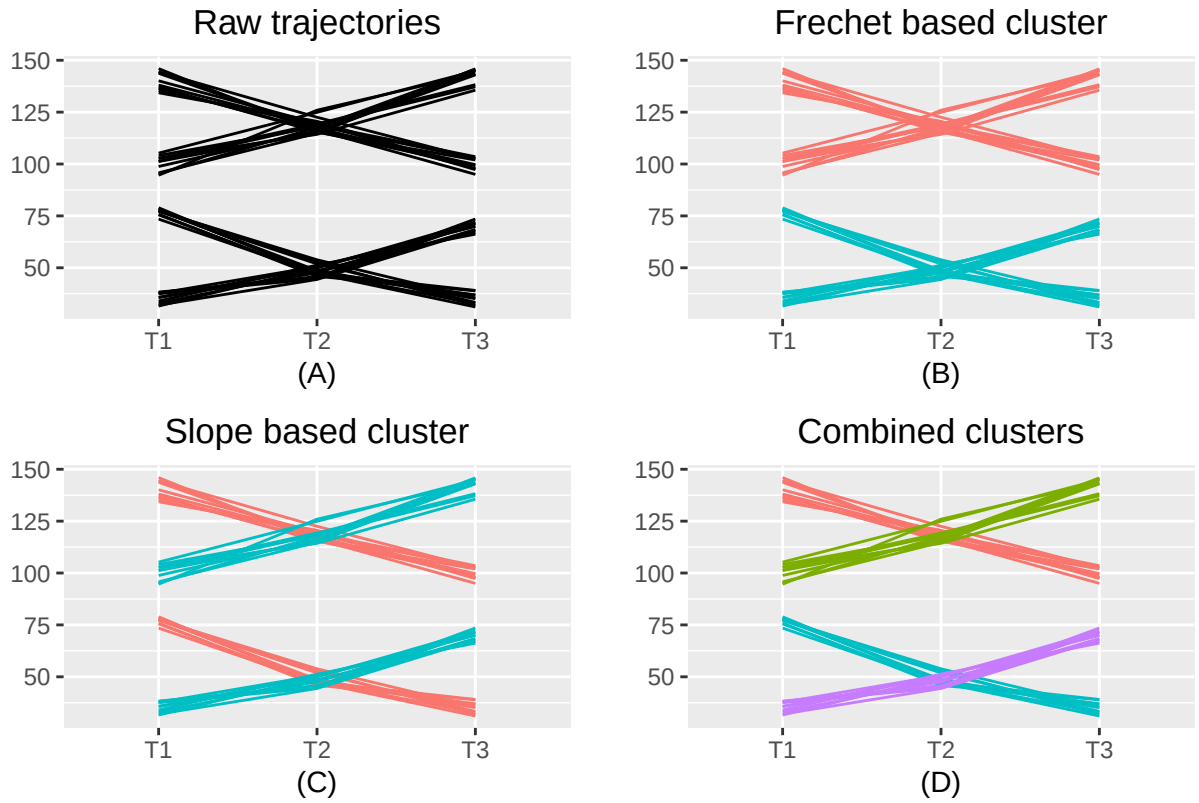


Fig. 1

Existe en la literatura una enorme cantidad de algoritmos y métricas de disimilitud o distancia, y a pesar de ello, no se suele abordar la problemática del segundo caso (trayectorias con similar pendiente independientemente de la intensidad). Normalmente, los individuos con trayectorias próximas terminan clasificados en los mismos grupos.

Con este paquete proponemos una metodología que permite agrupar esas trayectorias basándonos en distancias físicas empleando métrica de distancia adaptada a series temporales (distancia de Frechet) y una nueva aproximación basada en similiudes de pendientes de las trayectorias, donde éstas se agrupan en función de dicha similitud independientemente de que se encuentren alejadas (e.d. con valores de expresión diferentes).

Además se da la opción de combinar ambas clasificaciones de manera que el resultado final sería el de trayectorias agrupadas en base a sus valores y a sus evoluciones.

Puesto que en gran cantidad de estudios (especialmente en el campo de la bioinformática), el número de trayectorias puede ser muy grande (miles o decenas de miles), se ha desarrollado una metodología en la que mediante un preagrupamiento, se obtiene una serie de "representantes" de las trayectorias, denominados senators, estos senators son entonces clasificados mediante alguna o ambas de las técnicas anteriormente mencionadas. Finalmente el conjunto de trayectorias es asignado al cluster al que su senator ha sido asignado. De esta manera se reduce el coste computacional y el riesgo de desbordar la memoria.

Getting started

La instalación se lleva a cabo mediante la función `install_github`.

```
devtools::install_github("fpsanz/tscR")
```

Puede verse esta ayuda invocando la viñeta:

```
library(tscR)
```

```
browseVignettes("tscR")
```

Simple clustering

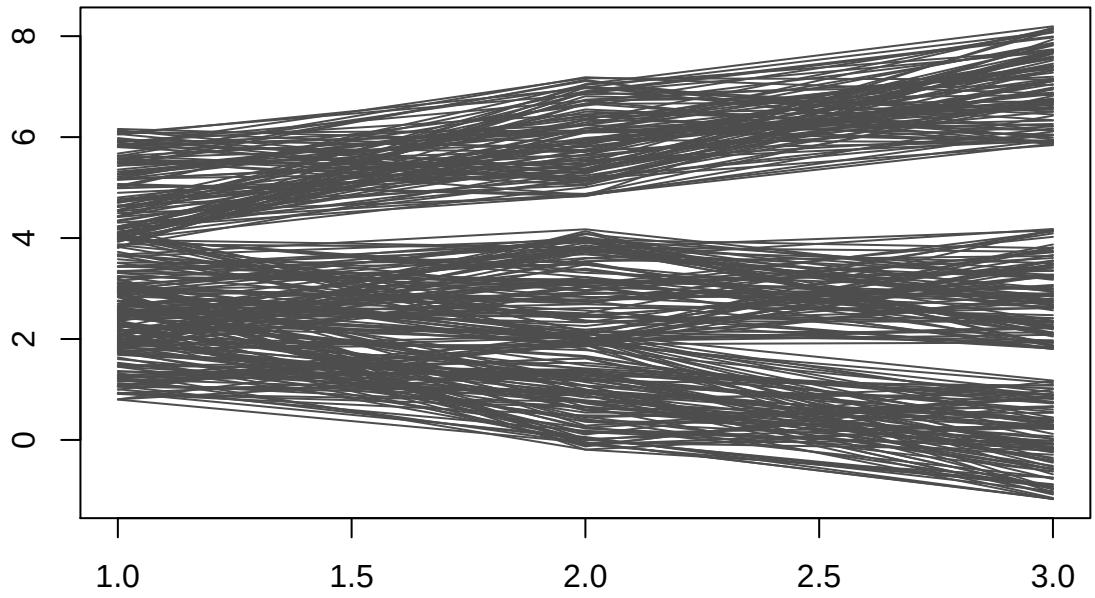
Based on slope distance

Los datos de entrada deben ser un dataframe o matrix donde las filas serán cada una delas trayectorias y las columnas los momentos en el tiempo (Table 1).

En primer lugar vamos a cargar un conjunto de trayectorias de ejemplo incluidas en la librería

```
data(tscR)
df <- tscR
head(df)
#>         T1        T2         T3
#> 1 6.161502 5.9001500  6.6022870
#> 2 3.344366 3.6755408  2.9476736
#> 3 1.058245 0.8805583 -0.5132939
#> 4 4.416688 6.7408719  7.1964777
#> 5 2.409223 3.7986534  2.1898294
#> 6 1.056814 1.8803695 -1.0379369
```

tscR contiene 300 observaciones (trayectorias) con datos tomados en 3 puntos temporales regulares (day 1, day 2, day 3).



A continuación se calcula la matrix de similitudes que será de tamaño n x n, siendo n el número de filas de la matriz de entrada. Esta matrix es un objeto de la clase `dist` y contiene las similitudes entre trayectorias basado en pendientes similares.

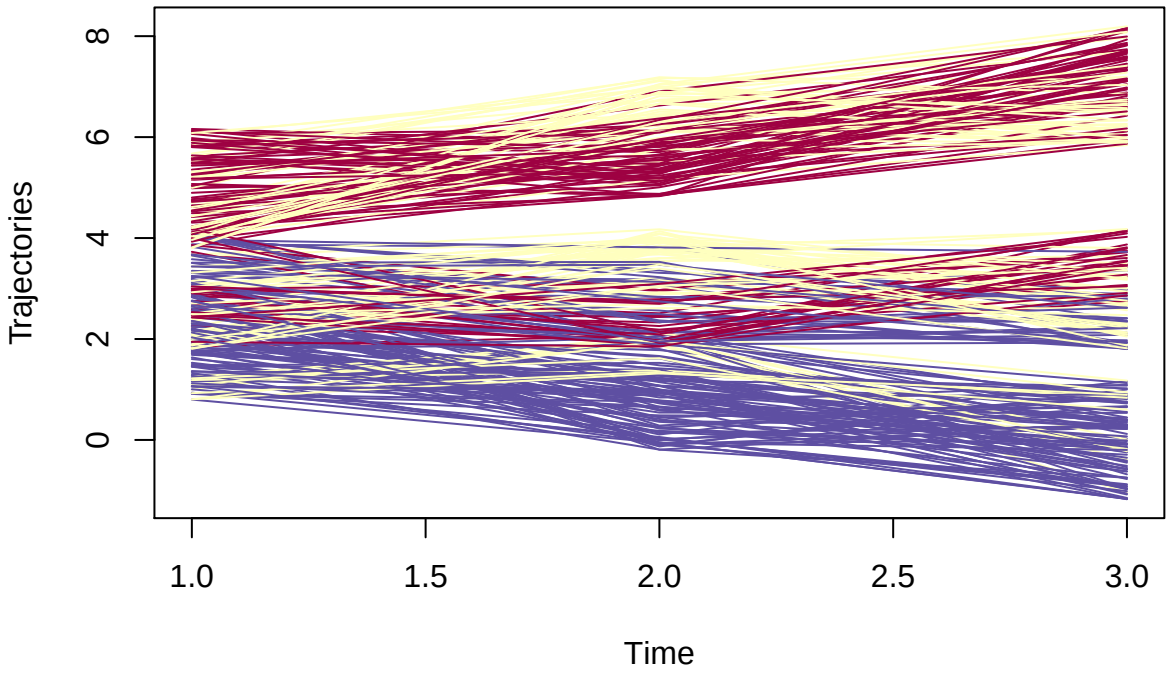
```
time <- c(1,2,3)
sDist <- slopeDist(df, time)
```

El siguiente paso sería agrupar las trayectorias basándonos en la similitud de sus pendientes independientemente de la distancia a la que se encuentren (meter referencia al paper).

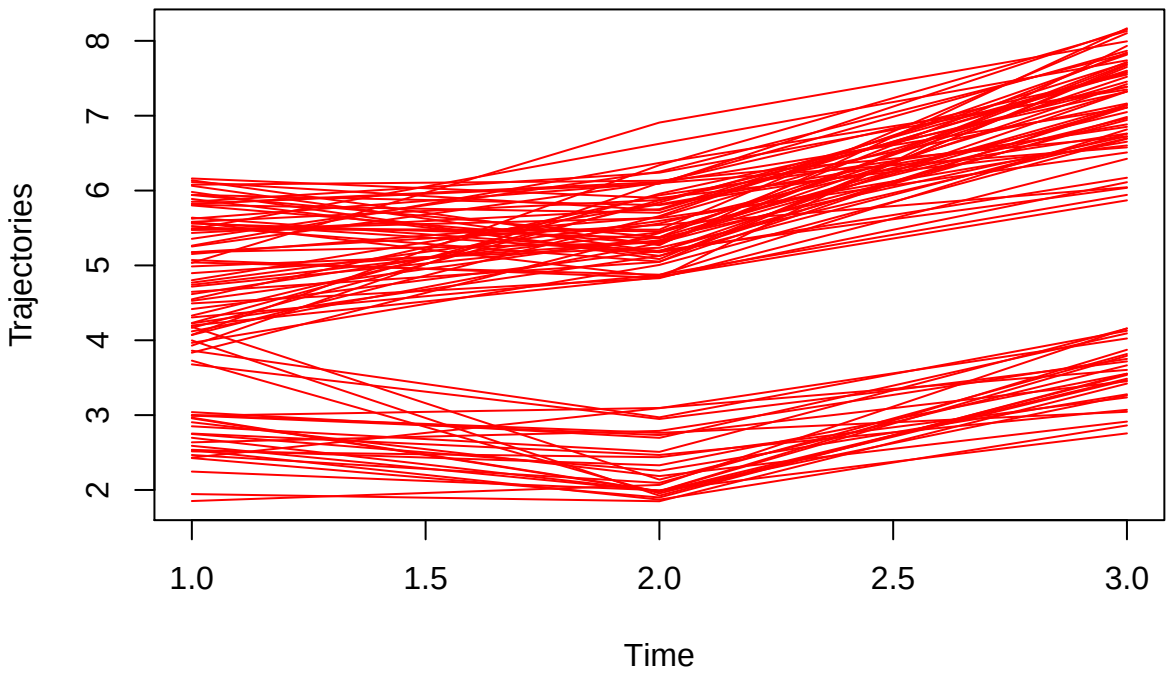
```
sclust <- getClusters(sDist, k = 3)
```

El resultado puede ser visualizado con la función `plotCluster`. Esta función toma como parámetros los datos originales (data), el objeto resultante de `getClusters` (cluster) y los cluster que se desean visualizar: "all" para visulizarlos todos en un gráfico único, un entero para visulizar las trayectorias de ese cluster o un vector definiendo que clusters se desean visualizar (uno por subplot)

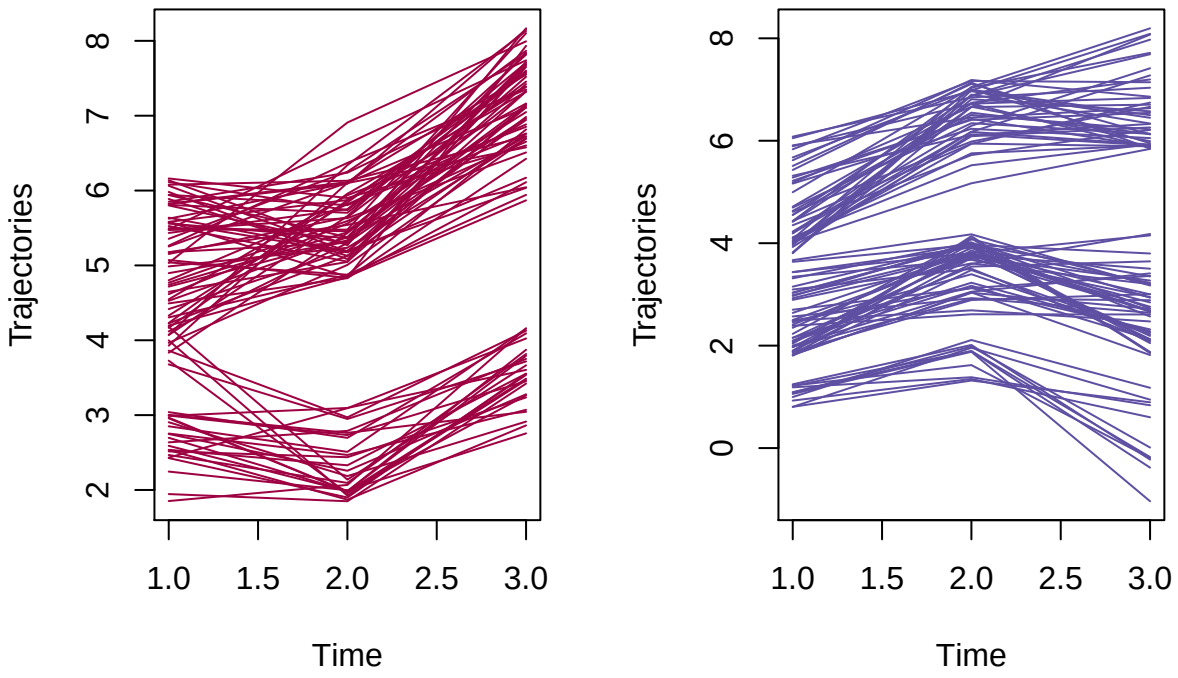
```
plotCluster(data = df, clust = sclust, ncluster = "all")
```




```
plotCluster(df, sclust, 1)
```




```
plotCluster(df, sclust, c(1,2))
```



Como puede observarse en este último gráfico, las trayectorias con evolución descedente-ascendente se han agrupado por un lado, independientemente de su distancia (plot izquierdo) y las de trayectoria ascendente-descentente por otro (plot derecho).

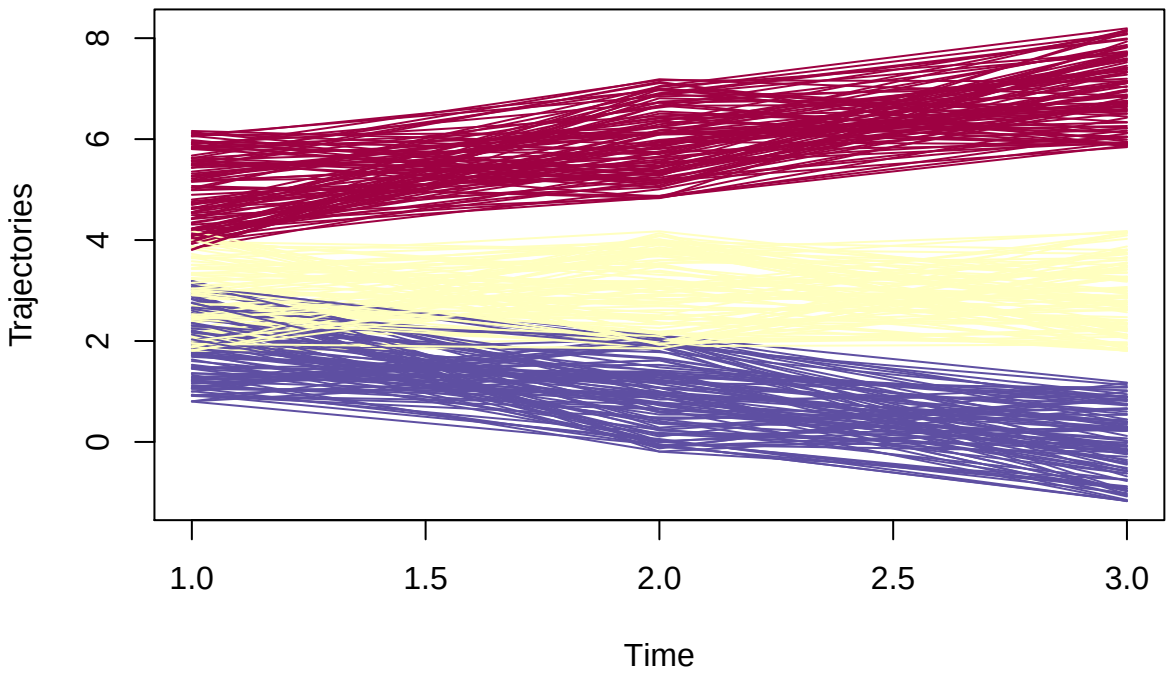
Based on Frechet distance

Es una medida de similitud entre curvas que tiene en cuenta la localización y orden de los puntos a lo largo de la curva (TODO: meter alguna referencia)

El procedimiento sería similar al caso anterior:

- Calcular la matriz de distancia.
- Obtener los clusters.
- Visualizar los resultados.

```
fdist <- frechetDistC(df, time)
fclust <- getClusters(fdist, 3)
plotCluster(df, fclust, "all")
```



Se observa que el agrupamiento tiene más que ver con la distancia (en términos euclídeos) entre trayectorias que con las pendientes en si mismas tal como se ilustra en el overview. En ciertas circustancias puede ser esta la clasificación que interese.

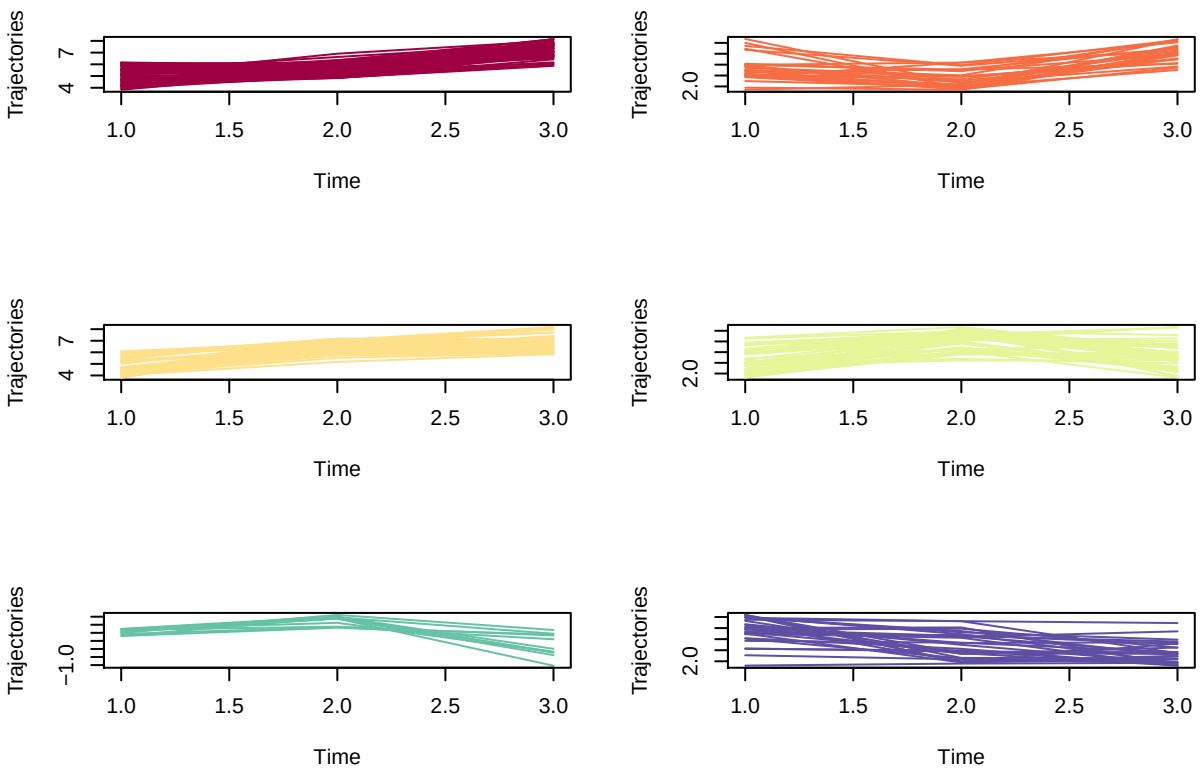
Combined clusters

Una tercera opción es combinar los resultados de ambos agrupamientos de manera que se obtendran conjuntos de trayectorias agrupados en función tanto de su distancia como de la similitud de sus pendientes.

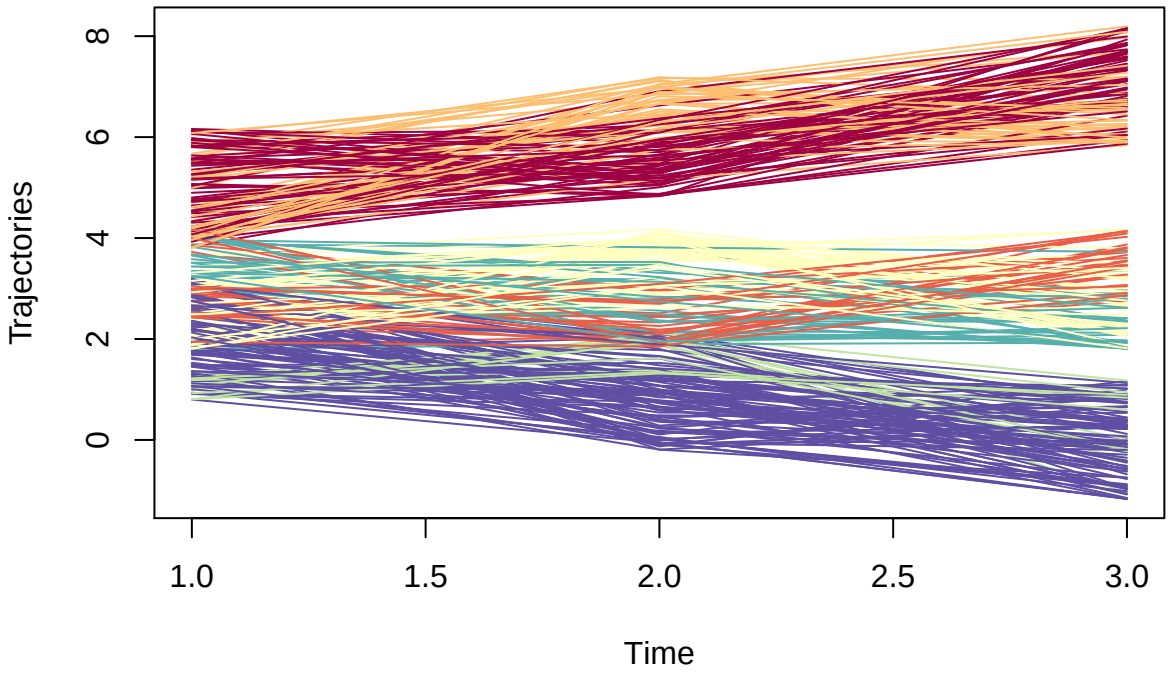
Para ello, la función `combineCluster` toma como entrada los objetos generados por getClusters generando una clasificación combinada.

De esta manera trayectorias muy cercanas pero con pendientes diferentes serán clasificadas en grupos diferentes y trayectorias con pendientes similares pero alejadas también se claficicarán en grupos diferentes. Así se obtiene una clasificación más fina (precisa) de las trayectorias.

```
ccluster <- combineCluster(sclust, fclust)
plotCluster(df, ccluster, c(1:6))
```




```
plotCluster(df, ccluster, "all")
```



Clustering large data

En los casos en los que se dispone de una cantidad de trayectorias elevadas (>1000), el coste computacional y de memoria puede ser muy elevado, principalmente cuando se trata de calcular las matrices de distancia. Nosotros proponemos hacer un clustering previo con la función `clara`, implementada dentro de `imputeSenators`, con un número de clusters elevado (p.e 100), esto generará 100 centroides, asumiendo que esos 100 centroides pueden representar al conjunto de formas de todas las trayectorias (de ahí el nombre de senators). Una vez obtenidos estos senators, se puede proceder con ellos como en el apartado anterior, agrupando en base distancia de Frechet, pendientes o ambas y generando el número de cluster deseados. Una vez obtenidos los clusters, como se ha guardado a que senator pertenecia cada trayectoria original, mediante `imputeSenatorToData` se asigna cada trayectoria a los clusters finales. Esta metodología permite clasificar un número muy elevado de trayectorias a un bajo coste computacional y de memoria.

El procedimiento, sería el siguiente:

En primer lugar obtener los senators

```
data( "tscR" )
bigDF <- tscR
senators <- imputeSenators( bigDF, k = 100 )
#> Setting k to 30 . 10% total of data
```

El objeto generado (`senators`) es una lista de 3 elementos

- $data: dataframe con los datos originales

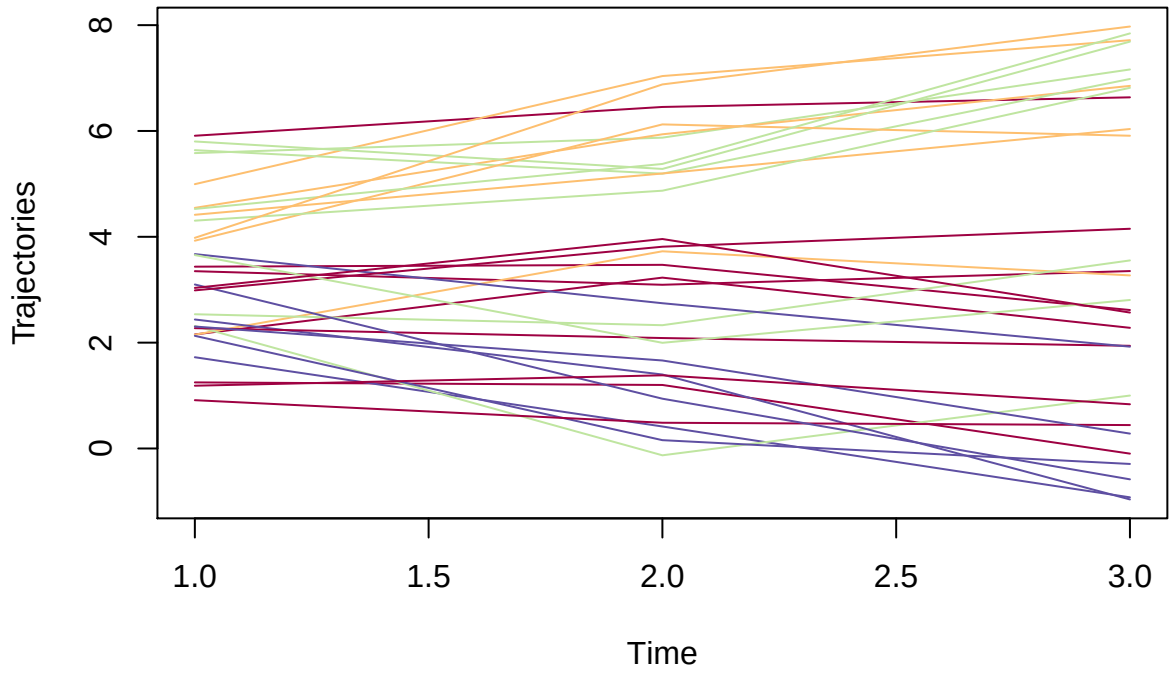
- $senatorData: es una matrix con las trajectorias de los senators
- $senatorCluster: vector con los clusters de los senators

Como ejemplo clasificaremos en función de la distancia basada en pendientes, aunque como se ha mencionado anteriormente se podría hacer en base cualquiera de las distancias e incluso combinarlas.

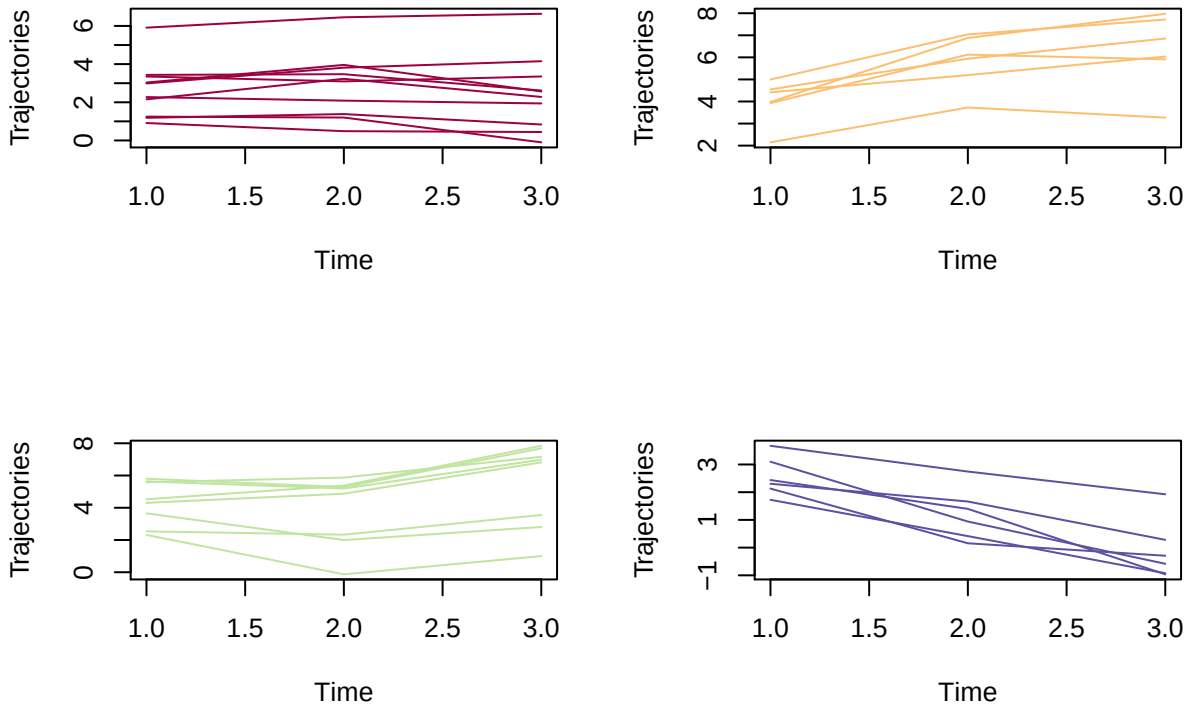
```
sdistSen <- slopeDist( senators$senatorData, time = c( 1, 2, 3 ) )
cSenators <- getClusters( sdistSen, k = 4 )
```

Se podría visualiza la clasificación realizada a los senators

```
plotCluster(senators$senatorData, cSenators, "all")
```




```
plotCluster(senators$senatorData, cSenators, c(1,2,3,4))
```



Finalmente, hay que asignar los datos originales a los nuevos cluster

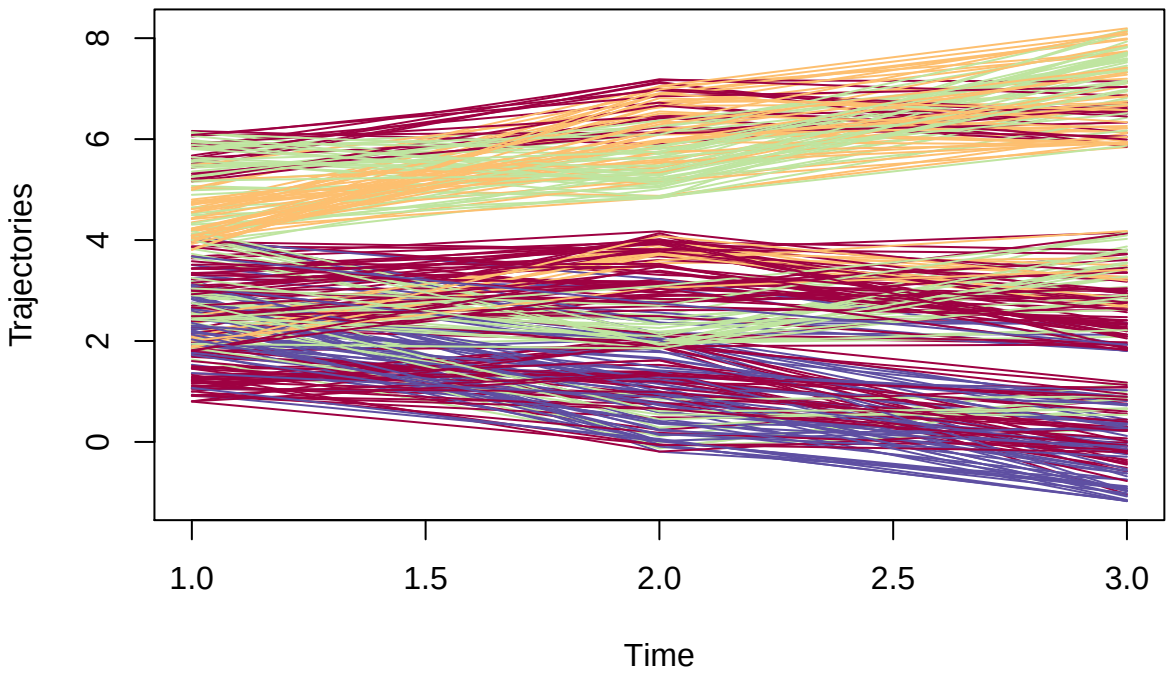
```
endCluster <- imputeSenatorToData(senators, cSenators)
```

Esto crea un objeto de la clase imputeSenator con 3 slots

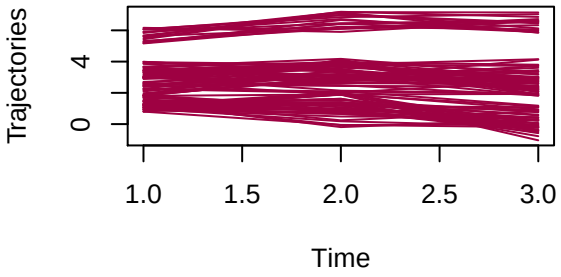
- @data: contiene el data frame con los datos originales
- @senators: identifica cada dato de data a que senator pertenece
- @endcluster: contiene los clusters finales a los que han sido asignados los datos

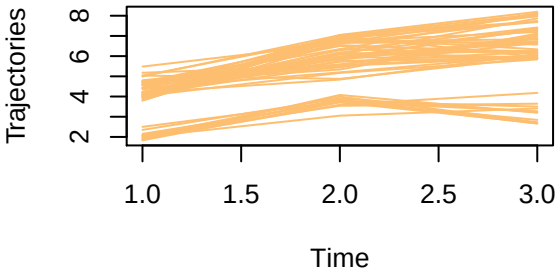
Y la visualización

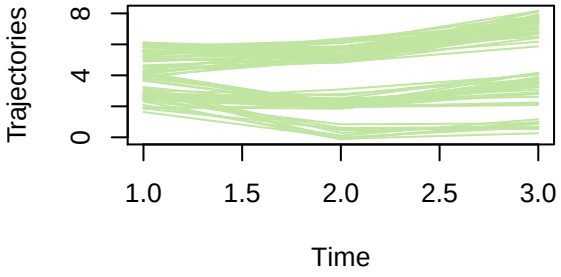
```
plotClusterSenator(endCluster, "all")
```

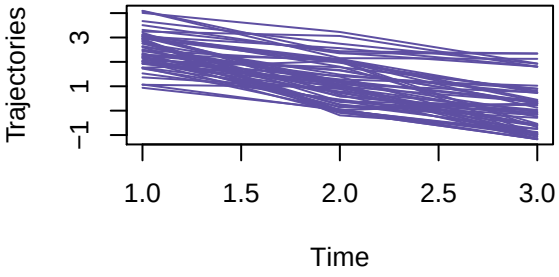



```
plotClusterSenator(endCluster, c(1,2,3,4))
```


Trajectories
0
4
1.0
1.5
2.0
2.5
3.0
Time


Trajectories
2
4
6
8
1.0
1.5
2.0
2.5
3.0
Time


Trajectories
0
4
8
1.0
1.5
2.0
2.5
3.0
Time


Trajectories
−1
1
3
1.0
1.5
2.0
2.5
3.0
Time