spm: an R-infrastructure package for Stochastic Process Modeling of survival trajectories from longitudinal studies

Ilya Y. Zhbannikov

2015-12-11

Overview

The R-package `spm` (https://github.com/izhbannikov/spm) is developed for modeling trajectories from longitudinal data and it allows (1) data simulation and (2) estimating the process parameters using maximum likelihood estimation by optimizing parameters used in the model. Specifically, developed R-package spm allows (i) one-dimensional SPM; (ii) multiple dimensional SPM; (iii) data simulation for one- and multiple dimensions.

Data description

Data represents a typical longitudinal data in form of two datasets: longitudinal dataset (follow-up studies), in which one record represents a single observation, and vital (survival) statistics, where one record represents all information about the subject. Longitudinal dataset cat contain a subject ID (identification number), status (event(1)/no event(0)), time and measurements across the variables. The `spm` can handle an infinite number of variables but in practice, 5-7 variables is enough.

Below there is an example of clinical data that can be used in `spm` and we will discuss the field later. Longitudinal studies:

```
##   X ID IndicatorDeath Age AgeNext      DBP      BMI
## 1 1  1              0  30      32 80.00000 25.00000
## 2 2  1              0  32      34 80.51659 26.61245
## 3 3  1              0  34      36 77.78412 29.16790
## 4 4  1              0  36      38 77.86665 32.40359
## 5 5  1              0  38      40 96.55673 31.92014
## 6 6  1              0  40      42 94.48616 32.89139
```

Vital statistics:

```
##   X ID IsDead   LSmort
## 1 1  1      1 85.34578
## 2 2  2      1 80.55053
## 3 3  3      1 98.07315
## 4 4  4      1 81.29779
## 5 5  5      1 89.89829
## 6 6  6      1 72.47687
```

Data fields description

Longitude studies

- ID - subject unique identificatin number.
- IndicatorDeath - 0/1, indicates death of a subject.
- Age - current age of subjects.
- AgeNext - next age of subject he will attend to the survey/exam.
- DBP, BMI - covariates, here “DBP” represents a diastolic blood pressure, “BMI” a body-mass index.

Vital statistics

- ID - subject’s unique ID.
- IsDead - death indicator, 0 - alive, 1 - dead.
- LSmort - age at death of stopping observations.

Discrete and Continuous cases

There are two main SPM types in the package: discrete model and continuous model. Discrete model assumes equal intervals between follow-up observations. The example of discrete dataset is given below.

```
library(spm)
data <- simdata_discr_MD(N=10, ystart=c(80), k=1)
head(data)
```

```
##      id xi t1 t2    par1_1    par1_2
## [1,]  1  0 30 31  80.00000  83.21808
## [2,]  1  0 31 32  83.21808  95.29622
## [3,]  1  0 32 33  95.29622  96.13012
## [4,]  1  0 33 34  96.13012  99.94892
## [5,]  1  0 34 35  99.94892 106.62143
## [6,]  1  0 35 36 106.62143 105.35499
```

In this case there are equal intervals between t1 and t2 (Age and Age.next).

The opposite is continuous case, in which intervals between observations are not equal. The example of continuous case dataset is shown below:

```
library(spm)
data <- simdata_cont_MD(N=5,ystart = c(50))
head(data)
```

```
##   id xi       t1       t2       y1  y1.next
## 1  1  0 69.52845 70.81755 50.76055 56.59938
## 2  1  0 70.81755 71.98745 56.59938 54.09433
## 3  1  0 71.98745 72.15929 54.09433 54.14335
## 4  1  0 72.15929 73.54864 54.14335 49.99899
## 5  1  1 73.54864 75.58982 49.99899       NA
## 6  2  0 47.41382 48.23278 48.64780 54.94621
```

Discrete case

In discrete case, we use the following assumptions:

$$\bar{y}(t+1) = \bar{u} + \bar{R} \times \bar{y}(t) + \bar{\epsilon}$$

$$\mu(t) = \mu_0(t) + \bar{b}(t) \times \bar{y}(t) + \bar{Q} \times \bar{y}(t)^2$$

Where:

$$\mu_0(t) = \mu_0 e^{\theta t}$$

$$\bar{b}(t) = \bar{b} e^{\theta t}$$

$$\bar{Q}(t) = \bar{Q} e^{\theta t}$$

Continuous case

$$\mu(u) = \mu_0(u) + (\bar{m}(u) - \bar{f}(u)^* \times \bar{Q}(u) \times (\bar{m}(u) - \bar{f}(u)) + Tr(\bar{Q}(u) \times \bar{\gamma}(u))$$

$$dm(t)/dt = \bar{a}(t) \times (\bar{m}(t) - \bar{f}_1(t)) - 2\bar{\gamma}(t) \times \bar{Q}(t) \times (\bar{m}(t) - \bar{f}(t))$$

$$d\bar{\gamma}(t)/dt = \bar{a}(t) \times \bar{\gamma}(t) + \bar{\gamma}(t) \times \bar{a}(t)^* + \bar{b}(t) \times \bar{b}(t)^* - 2\bar{\gamma}t \times \bar{Q}(t) \times \bar{\gamma}(t)$$

Coefficient conversion between continuous and discrete cases

$$Q = Q$$

$$\bar{a} = \bar{R} - diag(k)$$

$$\bar{b} = \bar{\epsilon}$$

$$\bar{f}1 = -1 \times \bar{u} \times a^{\bar{-}1}$$

$$\bar{f} = -0.5 \times \bar{b} \times Q^{\bar{-}1}$$

$$mu_0 = mu_0 - \bar{f} \times \bar{Q} \times t(\bar{f})$$

$$\theta = \theta$$

Case with time-dependent coefficients

In two previous cases, we assumed that coefficients is sort of time-dependant: we multiplied them on to

$$e^{\theta t}$$

. In general, this may not be the case. We extend this to a general case, i.e. (we consider one-dimensional case):

$$a\bar{(t)} = par_1 t + par_2$$

- linear function.

The corresponding equations will be equivalent to one-dimensional continuous case described above.

Simulation

We added one- and multi- dimensional simulation to be able to generate test data for hyphotesis testing. Data, which can be simulated can be discrete (equal intervals between observations) and continuous (with arbitrary intervals).

Discrete

For discrete case:

```r
library(spm)
data <- simdata_discr_MD(N=100, ystart=c(75, 94), k=2)
head(data)
```

```
##      id xi t1 t2   par1_1   par1_2   par2_1   par2_2
## [1,]  1  0 30 31 75.00000 66.73556 94.00000 99.20051
## [2,]  1  0 31 32 66.73556 74.91567 99.20051 99.99805
## [3,]  1  0 32 33 74.91567 70.14932 99.99805 95.69201
## [4,]  1  0 33 34 70.14932 57.48320 95.69201 91.20037
## [5,]  1  0 34 35 57.48320 46.02214 91.20037 91.65936
## [6,]  1  0 35 36 46.02214 37.53984 91.65936 80.16952
```

Continuous

For continuous case:

```r
library(spm)
data <- simdata_cont_MD(N=100)
head(data)
```

```
##   id xi       t1       t2       y1  y1.next
## 1  1  0 44.55783 46.16160 71.91485 72.17498
## 2  1  0 46.16160 47.79085 72.17498 71.99416
## 3  1  0 47.79085 48.56471 71.99416 72.28497
## 4  1  0 48.56471 48.99581 72.28497 72.56095
## 5  1  0 48.99581 50.97423 72.56095 74.74800
## 6  1  0 50.97423 51.81338 74.74800 65.76733
```

Simulation strategies

R-package `spm` currently offers continuous- and discrete time simulations.

Continuous-time simulation

Step 1

We come with a new subject and at first, we think that subject under study is alive at the starting observation time t1:

```
event(t1) = False
new_subject(t1) = True
```

Here `event` represents the particular event happened to the subject (for example, death or a failure). Then we start observing the subject over time.

Step 2

Computing t1 and t2:

```
if event = False and new_subject = True:
  t1 = runif(1, tstart, tend)
  t2 = t1 + 2*runif(1, 0, 1)
else if event = False and new_subject = False:
  t1 = t2
  t2 = t1 + 2*runif(1, 0, 1)
```

If we come with a "fresh" subject, we assume that `t1` as a random value, uniformly distributed from start time (tstart) to end (tend). Here `runif()` a random number generator which returns uniformly distributed value (from tstart to tend).

Step 3

Computing y1:

```
if event = False and new_subject = True:
  y1 = rnorm(1, ystart, sd0)
} else {
  y1 = y2
}
```

Here `rnorm()` a random number generator which returns normally distributed values, with mean=0 and sd=1.

Step 4

Compute Survival Fuction `S` based on the equations above.

Step 5

In this case we compare the `S` to the random number, uniformly distributed. If it is larger, than we assume that the event is happened (death of subject or system failure). Otherwise we proceed to the next iteration.

```
if S > runif(1, 0, 1) :
    y2 = rnorm(1, m, sqrt(gamma))
    event = True
    new_subject = True
else if event = False:
  y2 = rnorm(1, m, sqrt(gamma))
  event = False
  new_record = True
```

Discrete-time simulation

In this case we use equal intervals `dt` between observations and survival function `S` is computed directly from μ:

$S = e^{-1\mu(t_1)}$

The rest of the discrete simulation routine is the same as in continuous-time simulation case.