Supplementary material II:
Manual for %SimulateJointFrailty-SAS-Macro

(Associated article: Computational issues in fitting joint frailty models for recurrent events and an associated terminal event, *Computer Methods and Programs in Biomedicine*)

Contents

1 Description

The `%SimulateJointFrailty`-SAS-Macro simulates datasets from a joint frailty model for recurrent events with an associated terminal event. The joint frailty model is given by

$$\lambda_1(t|X,Z) = Z\lambda_{10}(t)\exp(\beta_1 X)$$

$$\lambda_2(t|X,Z) = Z^{\gamma}\lambda_{20}(t)\exp(\beta_1 X)$$

where Z is a gamma or lognormal distributed frailty with $E(Z) = 1$ and $Var(Z) = \theta$. To imitate the scenario of a two-arm randomized controlled trial (treatment vs. control), only a single $Bin(1, 0.5)$-distributed covariate X is considered. The baseline hazards originate from Weibull-distributions, i.e. the baseline-hazard for the j-th endpoint (1 = recurrent, 2 = terminal) is given by

$$\lambda_{j0}(t) = \lambda_j \nu_j t^{\nu_j - 1},$$

with λ_j being the scale-parameter and ν_j being the shape-parameter. The latter determines, if the hazard is decreasing ($\nu_j < 1$), constant ($\nu_j = 1$) or increasing ($\nu_j > 1$) over time.

2 Arguments

```
%macro SimulateJointFrailty(output,nrep,n,seed,frailtydist,theta,gamma,
                            scalerec,shaperec,scaleterm,shapeterm,
                            FU,censprob,beta1,beta2,path);
   ...
%mend SimulateJointFrailty;
```

`output`

Name of the large output-dataset containing the `nrep` stacked single sub-datasets (see Output-section).

`nrep`

Number of sub-datasets that are to be simulated.

`n`

Number of subjects per sub-dataset.

`seed`

Seed for reproducible random number generation.

`frailtydist`

Frailty-distribution: You have to specify either `frailtydist = lognormal` or `frailtydist = gamma`.

`theta`

Variance of the frailty-variable Z (mean is 1).

`gamma`

Exponent-parameter for the terminal event frailty.

`scalerec`

Scale-parameter for the recurrent event baseline hazard, which originates from a Weibull distribution.

`shaperec`

Shape-parameter for the recurrent event baseline hazard, which originates from a Weibull distribution.

`scaleterm`

Scale-parameter for the terminal event baseline hazard, which originates from a Weibull distribution.

`shapeterm`

Shape-parameter for the terminal event baseline hazard, which originates from a Weibull distribution.

`FU`

Time of administrative censoring, i.e. no subject's follow-up duration can be longer than that time.

`censprob`

Cumulative probability of random censoring within the time interval $[0, \texttt{FU})$. Random censoring is modeled as being uniform on $[0, \texttt{FU})$.

`beta1`

Treatment-effect of the binary covariate X on the recurrent event rate.

`beta2`

Treatment-effect of the binary covariate X on the terminal event rate.

`path`

This argument specifies the path where to store the output-datasets in csv-format (example: `path = C:\documents\results`). If no output-datasets should be stored in csv-format, please specify `path = none`.

3 Output

The macro produces 3 datasets in your SAS-library:

- A large dataset that contains the simulated stacked sub-datasets. Its name is determined by the `output`-argument.
- A first summary dataset that summarizes the event-number distributions within the `nrep` sub-datasets (suffix `_summary1`).
- A second summary dataset that summarizes the subject-number distributions for various recurrent event numbers within the `nrep` sub-datasets (suffix `_summary2`).

These three output-datasets may also be stored in csv-format by using the argument `path`. The argument `output` determines the name of the large dataset that contains the simulated stacked sub-datasets. At the same time, it determines the prefix of the names for the two summary-datasets. Let's illustrate that by an example: We use the macro to simulate 10 datasets from a joint frailty model, each with 500 subjects, and specify `output = Test`. The output-datasets are given by:

`Test`

sampleid	subjectid	frailty	x	timestart	timestop	eventindicator
1	1	0.174	1	0	1.362	1
1	1	0.174	1	1.362	2	0
1	2	1.863	0	0	0.435	0
1	3	0.962	1	0	0.653	1
1	3	0.962	1	0.653	1.162	1
1	3	0.962	1	1.162	1.872	2
⋮	⋮	⋮	⋮	⋮	⋮	⋮
1	500	2.653	1	0	2	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
10	1	0.763	1	0	0.652	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
10	500	1.972	0	0	0.543	1
10	500	1.972	0	0.543	2	2

The output-dataset `Test` contains the simulated stacked sub-datasets in long format structure (i.e. with multiple rows per subject if recurrent events occur during the follow-up) with the following variables:

- `sampleid` is the sub-dataset-number. In our example, `nrep = 10` sub-datasets are contained in the output-dataset `Test`, each with `n = 500` subjects.
- `subjectid` is the subject-specific identification number within a sub-dataset.

- `frailty` is the subject-specific realization of the frailty variable.
- `x` is the $Bin(1, 0.5)$-distributed, subject-specific covariate.
- `timestart` is the start-time of a new at-risk-interval.
- `timestop` is the stop-time of an at-risk-interval, i.e. a time point where anything happened in the subject's follow-up (recurrent event, terminal event, censoring).
- `eventindicator` specifies the type of event that happened at `timestop`. The following coding is applied: 0 for censoring, 1 for recurrent event, 2 for terminal event. In the example above, subject 1 from sample 1 has a recurrent event at time 1.362 and is censored at time 2. Subject 2 is censored at time 0.435 without having a recurrent event before. Subject 3 has two recurrent events at times 0.653 and 1.162 before having its terminal event at time 1.872. Importantly, the last line of each subject always contains either 0 or 2 as `eventindicator`, because the follow-up may only end due to censoring or due to the terminal event.

`Test_summary1`

eventindicator	x	min	mean	median	max
0	0	186	200.0	199.5	218
0	1	195	213.7	215.0	228
1	0	94	121.1	124.0	134
1	1	81	101.3	97.0	136
2	0	41	53.0	53.0	65
2	1	27	33.3	32.5	42

The output-dataset `Test_summary1` shows how the event numbers in different strata, defined by the `eventindicator` and the binary covariate `x`, are distributed in the sample of the `nrep = 10` sub-datasets. As an example, in our simulated `Test`-data,

- each sub-dataset has at least 81 recurrent events (`eventindicator = 1`) in the treatment group (`x = 1`).
- on average, each sub-dataset has 101.3 recurrent events (`eventindicator = 1`) in the treatment group (`x = 1`).
- in median, each sub-dataset has 97 recurrent events (`eventindicator = 1`) in the treatment group (`x = 1`).
- each sub-dataset has maximum 136 recurrent events (`eventindicator = 1`) in the treatment group (`x = 1`).

```
Test_summary2
```

Recevents	x	min	mean	median	max
0	0	163	176.0	176.0	189
0	1	164	176.6	176.5	198
1	0	39	48.5	48	59
1	1	41	49.4	50.5	57
2	0	11	18.1	18.5	22
2	1	8	13.9	12.5	22
3	0	4	7	7	10
3	1	2	5.1	4.5	10
4	0	1	2.2	2	3
4	1	1	1.6	1.5	3
>=5	0	1	1.6	2	2
>=5	1	1	1.4	1	3

The output-dataset `Test_summary2` shows how the subject numbers in different strata, defined by the recurrent event number `Recevents` and the binary covariate `x`, are distributed in the sample of the `nrep = 10` sub-datasets. As an example, in our simulated `Test`-data,

- in each sub-dataset there are at least 2 subjects that are in treatment group `x = 1` and have exactly `Recevents = 3` recurrent events during follow-up.
- on average, each sub-dataset has 5.1 subjects in treatment group `x = 1` that have exactly `Recevents = 3` recurrent events during follow-up.
- in median, each sub-dataset has 4.5 subjects in treatment group `x = 1` that have exactly `Recevents = 3` recurrent events during follow-up.
- in each sub-dataset there are maximum 10 subjects that are in treatment group `x = 1` and have exactly `Recevents = 3` recurrent events during follow-up.

4 Examples

Example 1

Simulate 1000 stacked sub-datasets, each with 2000 subjects, from a joint frailty model with the following parameter specifications:

- Baseline hazards: $\lambda_1 = 2$, $\nu_1 = 1$, $\lambda_2 = 0.3$, $\nu_2 = 1$
- Treatment effects: $\beta_1 = -0.4$, $\beta_2 = -0.2$
- Gamma-distributed frailty with variance $\theta = 3$ and exponent-parameter $\gamma = 0.8$
- Administrative censoring at time point 4 and cumulative censoring probability 0.15 within the interval $[0, 4)$

The output-datasets should have the prefix `SimJF` and be stored in csv-format in the folder `C:\documents\results`.

```
%SimulateJointFrailty(output = SimJF,
                      nrep = 1000,
                      n = 2000,
                      seed = 4535,
                      frailtydist = gamma,
                      theta = 3,
                      gamma = 0.8,
                      scalerec = 2,
                      shaperec = 1,
                      scaleterm = 0.3,
                      shapeterm = 1,
                      FU = 4,
                      censprob = 0.15,
                      beta1 = -0.4,
                      beta2 = -0.2,
                      path = C:\documents\results);
```

Example 2

Simulate 50 stacked sub-datasets, each with 1000 subjects, from a joint frailty model with the following parameter specifications:

- Baseline hazards: $\lambda_1 = 0.7$, $\nu_1 = 1$, $\lambda_2 = 0.3$, $\nu_2 = 1$
- Treatment effects: $\beta_1 = -0.4$, $\beta_2 = -0.2$
- Lognormal-distributed frailty with variance $\theta = 1.45$ and exponent-parameter $\gamma = 1.25$
- Administrative censoring at time point 2 and no additional random censoring within the interval [0,2]

The output-datasets should have the prefix `SimJFnew` and not be stored in csv-format.

```
%SimulateJointFrailty(output = SimJFnew,
                      nrep = 50,
                      n = 1000,
                      seed = 625373,
                      frailtydist = lognormal,
                      theta = 1.45,
                      gamma = 1.25,
                      scalerec = 0.7,
                      shaperec = 1,
                      scaleterm = 0.3,
                      shapeterm = 1,
                      FU = 2,
                      censprob = 0,
                      beta1 = -0.4,
                      beta2 = -0.2,
                      path = none);
```