BMC Bioinformatics

SOFTWARE Open Access

vivaGen – a survival data set generator for software testing

Matthias Gietzelt[1,2], Christian Karmen[1], Petra Knaup-Gregori[1] and Matthias Ganzinger[1*]

*Correspondence: matthias.ganzinger@med.uni-heidelberg.de
[1]Heidelberg University, Institute of Medical Biometry and Informatics, Im Neuenheimer Feld 130.3, 69120 Heidelberg, Germany
Full list of author information is available at the end of the article

Abstract

Background: Software testing is an essential part of the software development process, but real-world data may not be suited or available for testing purposes. In the medical context, it can be especially hard to get the necessary test data for various reasons such as privacy concerns. To overcome these obstacles and provide data for the necessary thorough tests of software, the generation of simulated data sets can be a solution. In this paper, we focus on the challenging task of generating such survival data sets containing known effects. So far, no user-friendly software exists for the simulation of survival data, as they are typically derived from clinical trials with follow-ups.

Results: To overcome these shortcomings, we developed an easy to use software package called *vivaGen*. In our Java software, parameters of survival time distributions are replaced by comprehensive measures that can be configured more intuitive by practitioners. *vivaGen* is equipped with a graphical frontend that allows users to adjust parameters and visualize the results in survival plots of the simulated cohorts.

Conclusions: *vivaGen* is freely available and published as open source. It provides a novel way to generate test data sets based on probability distributions in a comprehensive and user-friendly way.

Keywords: Data set generator, Survival data, Biomarker, Java

Background

Software testing is an essential part of the software development process [1]. The testing process is usually divided into different consecutive test levels: function test (on function/method level), module test, integration test, and system test. The higher the test level, the higher is the need for more complex test data. Especially, medical and clinical systems require a systematic and exceptionally intensive testing process, before such systems can be considered to assist physicians in diagnosis or treatment of patients. Although these systems could and should be tested using real-world data, there are sometimes disadvantages for this approach: real-world data can possibly be influenced or interfered by effects such as confounders [2] or mediators [3]. These effects are non-random and in many cases unknown. As a consequence, they are difficult to control and

© The Author(s). 2020 **Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/. The Creative Commons Public Domain Dedication waiver (http://creativecommons.org/publicdomain/zero/1.0/) applies to the data made available in this article, unless otherwise stated in a credit line to the data.

must be taken into account when dealing with real-world data sets for testing purposes. In addition, real-world data might contain gaps, making some tests hard to perform.

The use of simulation data is often a very helpful way to test complex software systems at certain test stages. Simulated data can be designed in a way that all effects are well-defined and all relationships between attributes are controllable. Thus, for testing purposes it seems to be a sensible step to include test data from an environment that is fully controllable, but which is as close as possible to the intended setting into the test process.

Simulation data in survival analysis

In a medical context, survival data typically emerge from longitudinal trials or registers, which include follow-up data. The main characteristics of survival data are the two related endpoints: the time period and the so-called event. The event can be for example "death" in case of overall survival observations, or "progression" in case of investigating progression-free survival periods. The time period is either the time, until the event happens, or otherwise, in case of no event, the duration of the observation period. If no event has occurred during the observation period, the observation is called right-censored [4].

Testing of systems that are processing survival data demands for well-defined data sets. It can be a challenging task to generate such survival data containing known effects. These effects should be found during the test by the system.

Related work

Currently, there are a number of software packages available to simulate survival data in the statistical language *R* [5]: The *survsim* package [6] is able to generate single or multiple event data for different survival time distributions and it allows for a flexible handling of its parameters. However, it is not easy to define the characteristics of the attributes. For example, the number of subjects affected by an event is defined via corresponding distributions. This configuration might be no problem for experts, but is challenging for users with only basic experience.

The *genSurv* package [7] allows for the generation of data from a progressive illness-death model [8] using a variety of models: a Cox Markov model, a time-dependent Cox model, and a Cox Proportional Hazard model. Again, the configuration of effects of the covariates requires special skills.

There are other *R* packages, too, which provide survival data simulation functions. But the main focus of these packages is the provision of survival estimation functions rather than the simulation aspect itself. Thus, the simulation functions are provided as a test environment to test the core functions. For example, the package *splinesurv* [9] is able to simulate clustered survival data with a specified number of clusters in the sample. The simulated data is used to provide sample data of the correct format to test the main survival estimation functions of the package.

To the authors' best knowledge, there is no package available that simulates data of a randomized controlled trial and supports a simple and comprehensive way for setting up survival time distribution parameters.

Objective

The aim of this research was to design and to implement a data set generator for survival data that is comprehensively configurable and is also suitable for users with only basic experience in setting up survival time distribution parameters. The data set should simulate survival data of a typical randomized controlled trial with follow-up data and should be designed to feature defined differences between the randomized groups.

The simulated data set should contain both: random attributes, which are unrelated to the survival outcome, and attributes with defined and controllable effects. We define such non-random attributes as "biomarkers", which are able to indicate, if a patient would be able to profit from a certain therapy in terms of an improved survival outcome. Such generated data sets can be used for testing in order to find the defined differences.

Organization of the paper

The paper is organized as follows. Chapter "Implementation" contains information about the requirements of the survival data generation process, highlights the data organization and important points during the software development process, and describes how the requirements were met. "Results" contains information about the current status of the development. Finally, we describe the "Lessons learned" and our "Future plans" in the chapter "Conclusion".

Implementation

Long-term survivors vs. short-term survivors

In a typical longitudinal survival study, one may observe a bimodal distribution with two peaks in the survival time when represented in a histogram. Figure 1 shows a survival time histogram from the "colon" data set as an example for longitudinal survival data with two fitted Weibull distributions [10]. This data set shows a bimodal distribution. Indeed, this is not a result of an isolated study – a bimodal distribution of survival time is a common phenomenon [11]. As a consequence for modelling survival data, one has to consider two typical main groups of survivors: the so-called long-term survivors (LTS, group with a higher chance of a long-term survival) and the early decliners resp. short-term survivors (STS, group with a lower chance of a long-term survival). Therefore, the simulated cohort should represent such a LTS and a STS group.

General cohort organization

One requirement for this data generator is to simulate survival data of typical randomized controlled trials on therapeutic comparisons with follow-up data. This means that the study cohort is organized in two balanced study arms: Arm B should be seen as the new therapy (e. g. a new medication), which is to be compared to the standard treatment of arm A (competing therapy/placebo arm). As a requirement, it should be possible to specify that one of both arms has a longer (or at least equal) survival compared to the other arm. This enables the specific manipulation of the data with an expected outcome in terms of an improved survival for a certain point in time t in one arm compared to the other arm. In the default settings, we chose arm B to be superior (or at least non-inferior) in comparison to arm A, i. e. $S_B(t) \geq S_A(t)$ for this scenario. But the user can also define $S_B(t) < S_A(t)$, which means that arm A is superior compared to arm B.

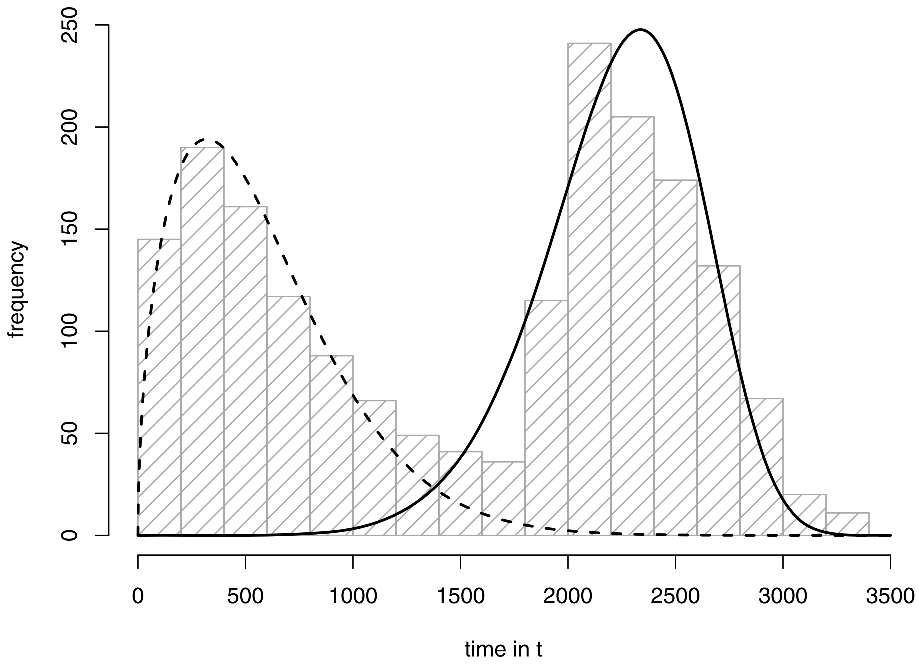


Fig. 1 Survival time histogram.The figure shows a typical bimodal distribution of survival time in a longitudinal data set with fitted Weibull distributions at each peak. The fitted Weibull distributions were computed using a maximum likelihood estimation method. The dashed function characterizes the group of short-term survivors, the full line function the long-term survivors. The data used to plot the histogram originates from the "colon" data set, which was originally described in [10] and is provided as part of the *R* package *survival* [18]

After having derived the necessity for distinguishing between the LTS and the STS group, a generalized model for generating the cohort data can be drawn. As shown by the example of Fig. 2, the cohort will be organized into two arms, both containing LTS and STS. Thereby, our approach specifies survivors and non-survivors in order to model defined effects into the data.

Setting up survival time distribution parameters

Another requirement is to have a convenient way to set up abstract survival time distribution parameters, which can be difficult to adjust. Since practitioners have an intuition about the estimation of the survival rate $S(t)$ for a certain point in time t, we decided to use this measure for setting up the survival time distribution. Thus, the user will be able to configure the survival function in a way that it touches a specific point in a survival plot.

In contrast to other approaches, this data generation approach should allow for the survival time distribution parametrization of a specific survival rate at a certain point in time. In the following paragraphs we introduce our concept for a simple parametrizations for a Weibull- and a log-logistic-distributed random variable for modelling survival time.

Let $Y_{\mathrm{Wei}} \sim \mathrm{Wei}\,(\lambda, k)$ be a Weibull-distributed random variable with λ as a scale parameter and k as a shape parameter and $\lambda, k \in \mathbb{R}_{>0}$. Given a point in time $t \in \mathbb{R}_{>0}$ with a fixed corresponding survival probability $S(t) \in (0, 1)$, the survival function S of the Weibull distribution can be solved for the unknown parameter λ:

$$S(t) = e^{-\left(\frac{t}{\lambda}\right)^k}. \tag{1}$$

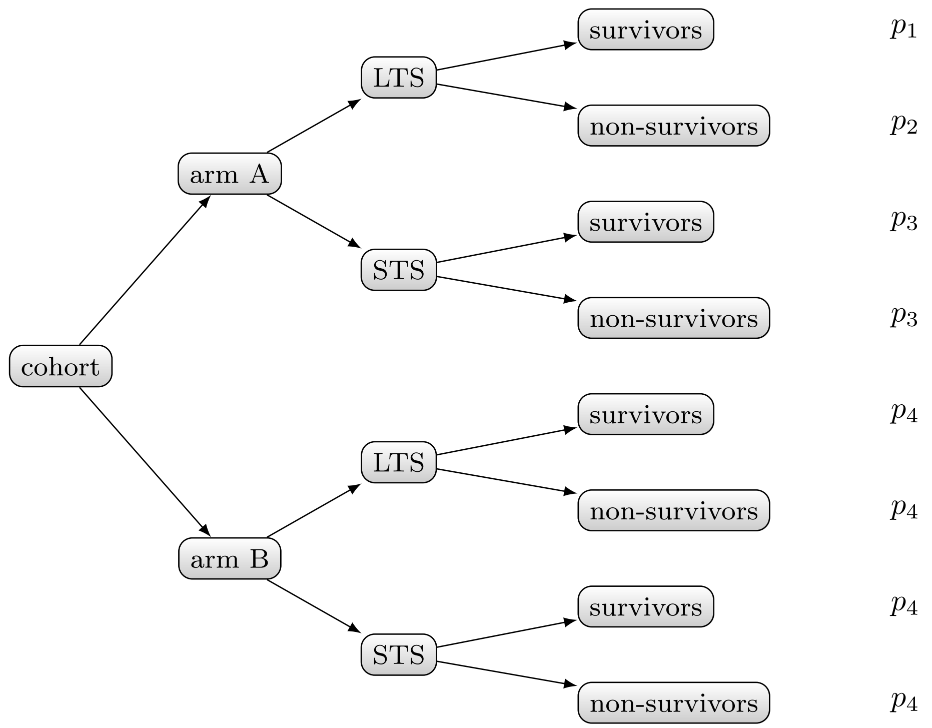


Fig. 2 Organization of the cohort and study arms. The leafs of this tree have assignments of probabilities $p_1, \ldots, p_4$ to indicate the prevalence of biomarkers in specific subgroups

It emerges for the parameter λ, which depends on t, $S(t)$ and k:

$$\lambda = 10^{\log(t) - \log(-\ln(S(t)))\cdot k^{-1}}. \tag{2}$$

With this result we are able to estimate parameter λ, but not k, which models the shape of the survival function. It seems that there is no useful possibility to estimate this parameter in a convenient way, so that the user still has to choose the shape parameter explicitly.

The existence of λ is shown in the following term by analyzing the exponent of the right hand side of Eq. 2:

$$\log \underbrace{(t)}_{>0} - \log \underbrace{\left(- \overbrace{\ln \underbrace{(S(t))}_{>0}}^{<0 \Rightarrow S(t)<1} \right)}_{>0} \cdot k^{-1}. \tag{3}$$

It emerges that

1. $t > 0$
2. $k \neq 0$, but the Weibull distribution already demands for $k > 0$
3. $0 < S(t) < 1$
4. $\lambda > 0$ (as follows from Eq. 2).

These results show that no additional limitations for the parameters must be met for the Weibull distribution.

Let $Y_{\mathrm{LL}} \sim \mathrm{LL}\,(\alpha, \beta)$ be a log-logistic-distributed random variable with $\alpha, \beta \in \mathbb{R}_{>0}$. Given a point in time $t \in \mathbb{R}_{>0}$ with a fixed corresponding survival probability $S(t) \in (0, 1)$, the survival function S of the log-logistic distribution

$$S\,(t) = \frac{1}{1 + \left(\frac{t}{\alpha}\right)^{\beta}} \tag{4}$$

can be solved for the unknown parameter α:

$$\alpha = t \cdot 10^{-\log\left(S^{-1}(t)-1\right)\cdot\beta^{-1}}. \tag{5}$$

The existence of α is shown in the following term by investigating the exponent of the right hand side of Eq. 5:

$$\log\left(\underbrace{S^{-1}(t) - 1}_{>0 \Rightarrow 0 < S(t) < 1}\right) \cdot \beta^{-1}. \tag{6}$$

It emerges that

1. $t > 0$
2. $\beta \neq 0$, but the log-logistic distribution already demands for $\beta > 0$
3. $0 < S(t) < 1$
4. $\alpha > 0$ (as follows from Eq. 5) for $t > 0$.

These results show that no additional limitations for the parameters must be met for the log-logistic distribution. β has a similar influence as compared to k of the Weibull distribution.

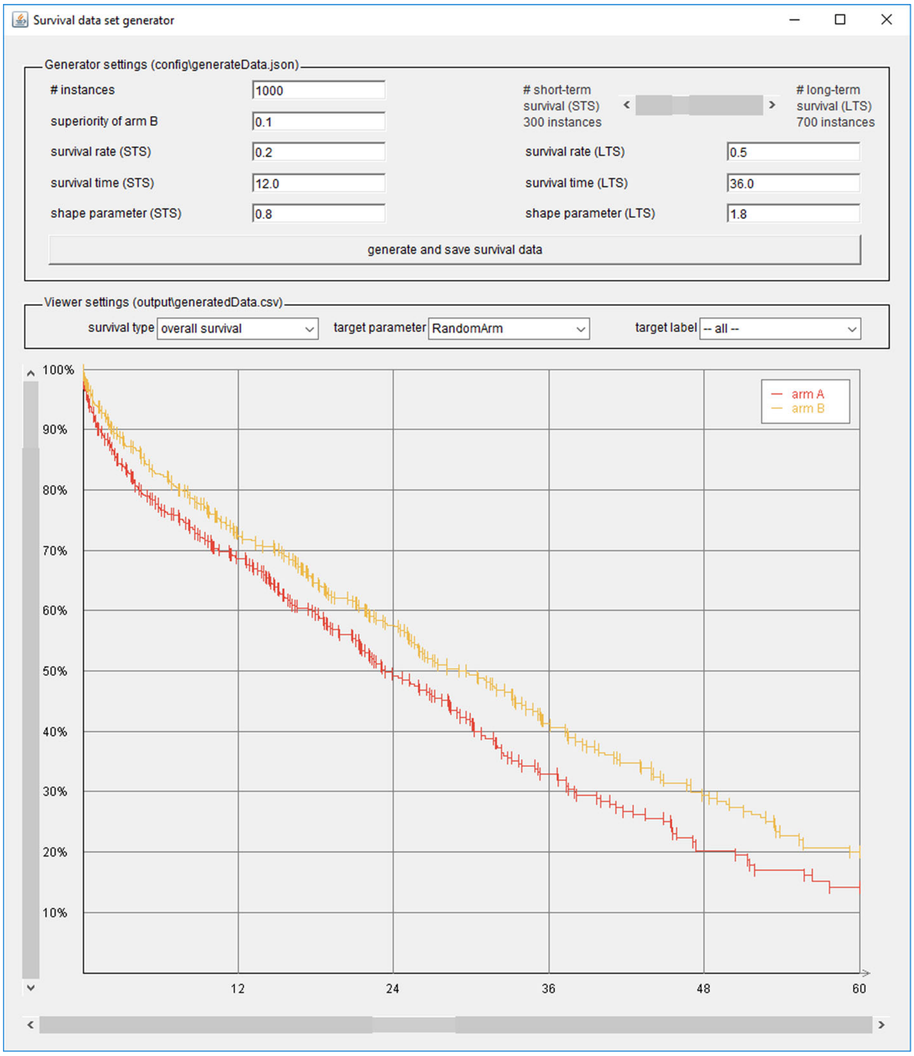
With the help of Eq. 2 (and Eq. 5, resp.), it is possible to easily set up survival time distribution parameters.

Independent random and non-random attributes

The generated independent attributes should represent both: random and non-random attributes. The non-random attributes are called "biomarker" in this paper. Biomarkers should be able to indicate, whether a patient will be responsive with regard to a certain treatment, or not. The requirement for the generation process was to have at least one nominal and one numeric biomarker. Biomarkers should have a configurable intensity. For the generation process it is mandatory that the biomarkers are overall equally distributed in study arms A and B, but show an accumulation in specific subgroups.

Results

The name of the software presented in this paper is *vivaGen*. Currently, in *vivaGen* it is possible to configure the above mentioned features in a graphical user interface (GUI). The GUI also provides an output visualization in terms of a survival plot. This plot instantly shows the changes made. Figure 3 shows a screenshot of the GUI. A more extensive configuration can be made using a configuration file in JavaScript Object Notation (JSON) format [12]. All parameters related to data generation presented in this section can be set in this file, too.



Survival data set generator

Generator settings (config\generateData.json)

# instances	1000	# short-term survival (STS) 300 instances	# long-term survival (LTS) 700 instances
superiority of arm B	0.1		
survival rate (STS)	0.2	survival rate (LTS)	0.5
survival time (STS)	12.0	survival time (LTS)	36.0
shape parameter (STS)	0.8	shape parameter (LTS)	1.8

Viewer settings (output\generatedData.csv)

survival type	overall survival	target parameter	RandomArm	target label	-- all --

Fig. 3 Screenshot of the GUI of *vivaGen*

Independent random and non-random attributes

Random attributes are being generated using the transformations given in the *Background* section. Currently, the following distributions are supported:

- Uniform distribution (nominal and numeric attributes are generated)
- Binomial distribution
- (Standard-)normal distribution
- Exponential distribution
- Weibull distribution
- Log-logistic distribution

vivaGen also allows for the definition of two kinds of biomarkers for each therapy arm independently: one nominal and one numerical biomarker. In the current state of development, the nominal biomarker is binomially distributed and the numerical biomarker follows a normal distribution. The specification of the biomarkers in this section will reflect the definition of those of arm A as an example, but the biomarkers of arm B are specified respectively.

For the biomarker generation process we make use of the cohort organization described in the *Implementation* section. The nominal biomarker is actually a composite attribute of four binomial distributions $\text{Bin}\,(n_i, p_i)$, which have four different probability values p_i in order to emphasize the ability to survive for particular subgroups. Binomial distributions provide results in terms of "success" and "failure", i. e. "true" and "false". Figure 2 shows the assignments of the probability values $p_1, \ldots, p_4$ as annotations of the leafs.

The assignment of probability values leads to a selective emphasizing of particular subgroups. The distinction within the LTS group of arm A was done to better emphasize the difference between both subgroups. Tests confirmed that a differentiation of the LTS group is necessary for the nominal attribute in order to get a distinct and visible result – especially for the arm that is inferior to the other. Because of the inferiority of the results of this arm as a whole, improvements might be only very slight, when modelling the LTS survivor group alone.

For configuring the biomarker's intensity, the user has to choose the probability values of

$$p_1 = p(\text{biomarker}|\text{arm A} \cap \text{LTS} \cap \text{survivors}), \tag{7}$$

$$p_2 = p(\text{biomarker}|\text{arm A} \cap \text{LTS} \cap \text{non-survivors}) \text{ and} \tag{8}$$

$$p_4 = p(\text{biomarker}|\text{arm B}) = p(\text{biomarker}) = p_{\text{prevalence}}. \tag{9}$$

The probability p_4 is equal to the overall prevalence of the biomarker. During the development process, it seemed to be reasonable that practitioners have an intuition about it. The missing probability $p_3 = p(\text{biomarker}|\text{arm A} \cap \text{STS})$ can be computed based on the other probabilities in order to meet the requirement that the biomarker is overall equally distributed in study arm A and B. Overall equally distributed in this sense means that the number of values for "true" is the same in arm A and arm B.

The numerical biomarker is specified as a composition of two normally distributed variables. The next equations define the identifiers of the distribution parameters:

$$\text{B}_{\text{biomarker, num}} \sim \mathcal{N}\left(\mu_{\text{biomarker}}, \sigma^2_{\text{biomarker}}\right) \text{ and} \tag{10}$$

$$\text{B}_{\text{non-biomarker num}} \sim \mathcal{N}\left(\mu_{\text{non-biomarker}}, \sigma^2_{\text{non-biomarker}}\right), \tag{11}$$

whereat the term "biomarker" is used for indicating that the biomarker might be present and "non-biomarker" that the biomarker is not necessarily present, because it is very likely that both distributions do overlap. The grade of overlapping can be interpreted as one measure of the configurable intensity of this biomarker.

Since the numerical biomarker should also have the same frequency in both arms, we applied a two-step procedure for assigning the two normally distributed variables. First, we used to same strategy as for the nominal biomarker for indicating the same number of subjects in arm A and arm B as "true" for having the biomarker. Furthermore, the intensity of the biomarker can be adjusted with similar, but independent adjustment parameters as for the nominal attribute. After that, values of $\text{B}_{\text{biomarker, num}}$ are assigned to those subjects marked as "true" and $\text{B}_{\text{non-biomarker num}}$ to those marked as "false".

Long-term survivors vs. short-term survivors

In both groups there are subjects, who have survived and others who have not survived. A simple histogram of survival time is only able to represent the observation period of the subjects. There is no indication about the subjects' survival status. In other words, the histogram does not reflect whether the subjects

- have refused to participate in the study or
- have been rejected from study observation (e. g. because of non-adherence) with status "alive" or
- died during the observation period.

Survival plots are diagrams, which are able to visualize outcomes for survival time and survival status. One example for such a survival plot is the Kaplan-Meier survival curve. Figure 4 illustrates such a survival plot. Using the results of the section *setting up survival time distribution parameters*, the diagram shows an example configuration of a resulting LTS and STS curve. Both curves are touching specified points. If there is no need for distinguishing between the groups LTS and STS, the size of one of the groups can be set to 0.

Nominalization of attributes

In order to visualize numeric attributes (random as well as non-random attributes) in a survival plot, these attributes have to be nominalized. Therefore, a nominalization process will be applied that uses a cutoff value c for dichotomizing attribute A into two survival

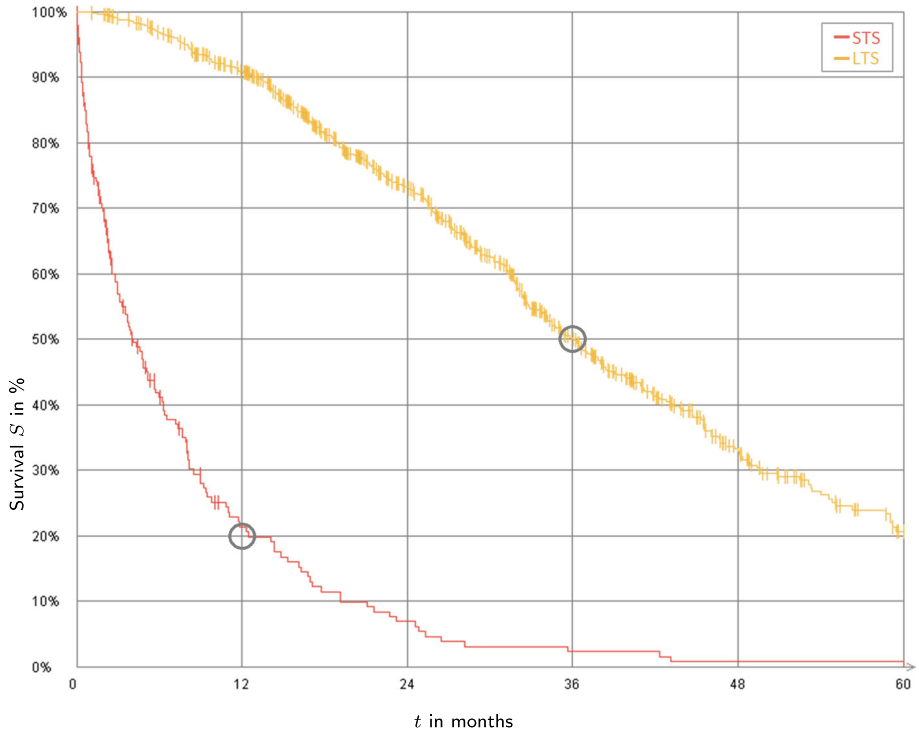


Fig. 4 LTS vs. STS. Each survival curve touches a specified point: The black curve (in color reproduction: red) of STS touches (t, $S(t)$) = (12 months, 20 %) and the light gray curve (in color reproduction: yellow) of LTS touches (t, $S(t)$) = (36 months, 50 %)

functions: $S(t|A \leq c)$ and $S(t|A > c)$. The cutoff value c will be computed in order to maximize the area between the emerging survival functions in a survival plot to get the best discriminating value. The time complexity for the nominalization process is in $\mathcal{O}\left(N^2\right)$, whereas N denotes the number of subjects.

We evaluated the nominalization process using the following hardware and software equipment: Intel CPU i5-4570@3.20 GHz with 16 GByte RAM, Java SE Runtime Environment (build 1.8.0_131-b11) on a Microsoft Windows 7 Professional 64-bit version 6.1.7601 Service Pack 1 Build 7601. Table 1 shows the computing times for nominalizing a single numeric attribute for different cohort sizes.

Generation of single events

For our use case, it was adequate that the endpoint of interest is a single event such as overall survival or progression-free survival. Thus, in the current state of the system, we only consider single event outcomes.

Conclusion

In this paper we introduced the software *vivaGen* for flexibly and comprehensively generating simulated survival data sets of a typical randomized controlled trial with follow-up data. *vivaGen* is designed to feature defined differences between the randomized groups. The data sets include random and non-random attributes and can be used for testing software for predicting or modelling survival outcomes. The non-random attributes are of configurable intensity regarding survival outcomes. The GUI of *vivaGen* allows for a targeted specification of survival parameters and attributes, which can be visualized and controlled using survival plots. The generated survival data can have a bimodal distribution of survival time in order to simulate long-term and short-term survivors.

Lessons learned

The main question during the implementation process of *vivaGen* was how to achieve the equal distribution in arm A and arm B of the biomarkers. We tried many different approaches until we got the final solution presented in this paper. First implementations showed a low intensity, especially of the nominal biomarker of the arm that is inferior compared to the other.

The requirement of modelling one arm as being superior compared to the other was another challenge. We handled the problem with a simple solution and corrected the survival rate of each subgroup (see Fig. 2). The solution was to add half of the correction factor to the survival rate of arm B and subtract the other half from the survival rate of arm A. Thereby, one has to consider the sizes of the single subgroups.

Table 1 Computing times for nominalization

Number of subjects N	Computing time for nominalization in s
100	0.025
500	0.392
1000	1.353
5000	31.90
10000	142.6

The computing times for nominalization (in seconds) are given per attribute to nominalize

In recent studies, also concurrent endpoints are observed [13]. These endpoints can be, for example, progression of disease or death. In the current state of development of *vivaGen*, a combination of such endpoints has not been implemented yet. But the authors consider the generation of multiple events, which can be interpreted as either different concurrent events or recurrent events of the same type.

We used data sets generated by *vivaGen* to test a research decision support system (DSS) developed by our group. Aim of the DSS is to give a recommendation for a certain therapy for a new patient given a case base of patients and known survival outcomes. The defined relations of attributes and survival outcome in the generated data sets were successfully used to train and test the DSS on system level.

Future plans

First, the survival data set generator currently generates independently distributed random attributes. One of the next steps is to generate these attributes with configurable correlations between them. This could be achieved by implementing an attribute representing a certain "degree of dependence" between other attributes and the outcome attributes (survival time and survival status). Bender et al. describe techniques and models to generate simulation data regarding outcomes of the Cox proportional hazards model [14].

Second, currently *vivaGen* only supports the generation of binomial and normal distributions for the biomarkers. Other distributions could be added to gain more flexibility for the data generator.

Third, the parametrization of the generated biomarkers could be made more user-friendly. The idea is to provide one value of intensity for each biomarker, for example in the interval $[0, 1]$ such that 1 stands for a maximum possible intensity and 0 for no intensity.

Fourth, the GUI could be extended to configure all attributes in a more user-friendly environment. This also includes the visualization of histograms for configuring the random and non-random attributes.

Fifth, multiple events could be supported by the generator, such as recurrent events to support the test of systems demanding for such complex outcome data.

These future plans are intended for intensifying the tests of our DSS and should also encourage a broader spectrum of researchers as well as software developers to generate survival simulation data using *vivaGen* for testing their software.

Availability and requirements

Project name: *vivaGen*

Project home page: https://sourceforge.net/projects/vivagen/

Operating system(s): Platform independent

Programming language: Java

Other requirements: Java SE Runtime Environment 8 or higher, WEKA package [15], and a JSON package [12] with a suitable and compatible license, such as the JSON Software Bundle provided by Oracle [16].

License: GPL v2 [17]

Any restrictions to use by non-academics: no

Abbreviations

DSS: Decision support system; GUI: Graphical user interface; JSON: JavaScript object notation; LTS: Long-term survivor; STS: Short-term survivor

Acknowledgments

Not applicable.

Authors' contributions

MG2 – development of software and requirement analysis, wrote main parts of the manuscript. CK – assisted in developing the software and requirement analysis, and critically revised the manuscript. PK – supervised IT development part of the project and critically revised the manuscript. MG1 – lead IT development part of the project, supervised developing the software and requirement analysis, and critically revised the manuscript. All authors have read and approved the manuscript.

Funding

This research was done within the systems medicine project "Clinically Applicable, Omics-based Assessment of Survival, Side Effects, and Targets in Multiple Myeloma" (CLIOMMICS). The project was funded by the German Federal Ministry of Education and Research (BMBF, grant id: 01ZX1609A) as part of the e:Med initiative. Publication fees were covered by the Post-Grant-Fund for projects funded by BMBF (grant id: 16PGF0182). The funding body did not play any role in the design and development of the *vivaGen* software or in writing the manuscript.

Availability of data and materials

All data and software used and developed for this research are available at https://sourceforge.net/projects/vivagen/.

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Author details

[1]Heidelberg University, Institute of Medical Biometry and Informatics, Im Neuenheimer Feld 130.3, 69120 Heidelberg, Germany. [2]Peter L. Reichertz Institute for Medical Informatics of TU Braunschweig and Hannover Medical School, Carl-Neuberg-Str. 1, 30625 Hannover, Germany.

Received: 19 June 2019 Accepted: 31 March 2020
Published online: 29 April 2020

References

1. Ammann P, Offutt J. Introduction to Software Testing, 2nd edn. New York: Cambridge University Press; 2016, p. 364. https://doi.org/10.1017/9781316771273.
2. VanderWeele TJ, Shpitser I. On the definition of a confounder. Ann Stat. 2013;41(1):196–220. https://doi.org/10.1214/12-aos1058.
3. MacKinnon DP. Introduction to Statistical Mediation Analysis. Mahwah: Erlbaum; 2008, p. 488. https://doi.org/10.4324/9780203809556.
4. Singer JD. Applied Longitudinal Data Analysis: Modeling Change and Event Occurrence. New York: Oxford University Press; 2003, p. 644. https://doi.org/10.4324/9780203809556.
5. *R* Core Team. *R*: A Language and Environment for Statistical Computing. Vienna: *R* Foundation for Statistical Computing; 2017. https://www.R-project.org/.
6. Moriña D, Navarro A. The *R* package *survsim* for the simulation of simple and complex survival data. J Stat Softw. 2014;59(2):1–20. https://doi.org/10.18637/jss.v059.i02.
7. Araújo A, Meira-Machado L, Faria S. *genSurv*: Generating Multi-State Survival Data; 2015. *R* package version 1.0.3. https://CRAN.R-project.org/package=genSurv.
8. Meira-Machado L, de Uña-Álvarez J, Cadarso-Suárez C, Andersen PK. Multi-state models for the analysis of time to event data. Stat Methods Med Res. 2009;18:195–222. https://doi.org/10.1177/0962280208092301.
9. Sharef E, Strawderman RL, Ruppert D. *splinesurv*: Nonparametric Bayesian Survival Analysis; 2010. *R* package version 0.6-14. https://cran.r-project.org/src/contrib/Archive/splinesurv/.
10. Moertel CG, Fleming TR, MacDonald JS, Haller DG, Laurie JA, Goodman PJ, et al. Levamisole and fluorouracil for adjuvant therapy of resected colon carcinoma. N Engl J Med. 1990;322(6):352–8. https://doi.org/10.1056/NEJM199007193230312.
11. Karakoca A, Erisoglu U, Erisoglu M. A comparison of the parameter estimation methods for bimodal mixture weibull distribution with complete data. J Appl Stat. 2015;42:1472–89. https://doi.org/10.1080/02664763.2014.1000275.
12. The JavaScript Object Notation (JSON) Data Interchange Format. 2017. https://www.rfc-editor.org/rfc/rfc8259.
13. Emura T, Matsui S, Rondeau V. Survival Analysis with Correlated Endpoints: Joint Frailty-Copula Models. Singapore: Springer; 2019, p. 118. https://doi.org/10.1007/978-981-13-3516-7.
14. Bender R, Augustin T, Blettner M. Generating survival times to simulate cox proportional hazards models. Stat Med. 2005;24(11):1713–23. https://doi.org/10.1002/sim.2059.

15. Witten IH, Frank E, Hall MA, Pal CJ. Data Mining: Practical Machine Learning Tools and Technique, 4th edn. San Francisco: Morgan Kaufmann; 2016, p. 620.
16. Oracle Corp. *JSON* Software Bundle, Version 1.0.4; 2013. https://repo1.maven.org/maven2/org/glassfish/javax.json/1.0.4/javax.json-1.0.4.jar.
17. Free Software Foundation, General Public License GNU Version 2; 1991. https://www.gnu.org/licenses/old-licenses/gpl-2.0.
18. Therneau TM, Lumley T. *survival*: Survival Analysis; 2017. *R* package version 2.41-3. http://CRAN.R-project.org/package=survival.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Ready to submit your research? Choose BMC and benefit from:

- fast, convenient online submission
- thorough peer review by experienced researchers in your field
- rapid publication on acceptance
- support for research data, including large and complex data types
- gold Open Access which fosters wider collaboration and increased citations
- maximum visibility for your research: over 100M website views per year

At BMC, research is always in progress.

Learn more biomedcentral.com/submissions

BMC