NetSurvival.jl: A glimpse into relative survival analysis.

JuliaCon 2024 @ Eindhoven

Rim Alhajal and Oskar Laverny

July 10, 2024

[1] Aix Marseille Univ, INSERM, IRD, SESSTIM, Sciences Economiques & Sociales de la Santé & Traitement de l'Information Médicale, ISSPAM, Marseille, France.

Table of content

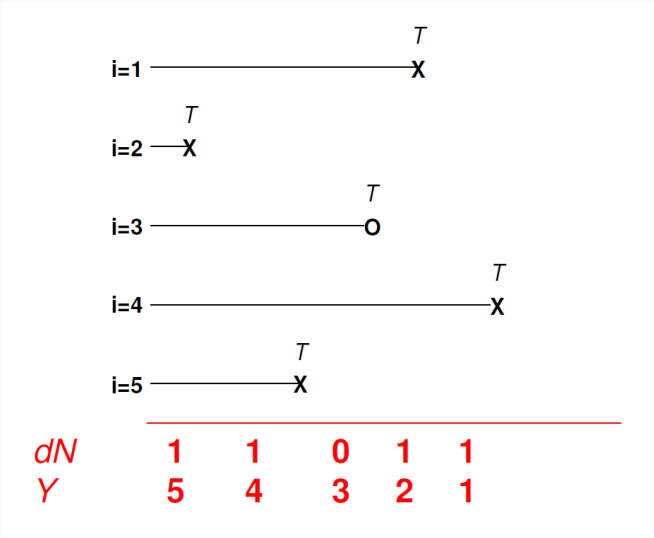
Introduction to relative survival analysis

Survival Analysis is a statistical theory that deals with censored positive random variables:

(i) Events (usually death) can occur or not before censoring (exit of experiment)

(ii) We want to model the time to event (survival)

(iii) Relies on counting processes and local martingales.



Relative survival stochastic settings

Relative Survival Context: In population-based studies, the specific cause of death is often unidentified, unreliable or even unavailable. We then work with the following random variables:

Random Variable	Name	Observed ?
E	"Excess" lifetime	No
P	"Population" lifetime	No, but known distribution.
$O = E \wedge P$	"Overall" lifetime	No
C	"Censoring" time	No
D	Vector of covariates	Yes
$T = O \wedge C$	Event time	Yes
$\Delta = \mathbb{1}\{T \leq C\}$	Event status	Yes
$\mathbb{1}\{E \leq P\}$	Cause of death	No

Notations: We use both $S_X(t) = \mathbb{P}(X > t)$, $\Lambda_X = -\ln S_X$ or $\lambda_X = \partial\Lambda_X$ to characterize a distribution.

Goal: Estimate the distribution of E, say by it's hazard $\partial\Lambda_E(t) = -\partial \ln S_E(t)$.

Remark: Without the cause of death indicatrix, standard competing risks models cannot be used...

Standard independence assumptions

Remark: The joint distribution of (E, P, C, D) characterizes our observations.

Assumptions (Standard independence[1])

(i) C, E and (P, D) are mutually independent

(ii) The distribution of $P|D$ is known from standard life tables (at time 0) for each individual.

(iii) Individuals are i.i.d.

Note: The population hazard for each individual λ_{P_i} is drawn from a reference `RateTable` from `RateTables.jl` and may depend on covariates $D_1, ..., D_p$ such as age, year, sex, country, race, etc...

[1] Maja Pohar Perme, Janez Stare, and Jacques Estève. "On Estimation in Relative Survival". In: *Biometrics* 68.1 (June 2011), pp. 113–120. ISSN: 0006-341X. DOI: 10.1111/j.1541-0420.2011.01640.x.

Standard estimators in the field

The estimation of net survival is usually discussed in terms of the estimation of the cumulative excess hazard $\Lambda_E(t)$ and/or the instantaneous hazard $\lambda_E = \partial\Lambda_E$.

Definition (Pohar Perme 2012[2])

The Pohar Perme estimator of the excess hazard is given, using $w_i(s) = S_{P_i}(s)^{-1}$, by:

$$\partial\hat{\Lambda}_E(s) = \frac{\sum_{i=1}^{n} w_i(s)\{\partial N_i(s) - Y_i(s)\partial\Lambda_{P_i}(s)\}}{\sum_{i=1}^{n} w_i(s)Y_i(s)}$$

and its variance can be estimated by:

$$\partial\hat{\sigma}^2_E(s) = \frac{\sum_{i=1}^{n} w_i(s)^2\partial N_i(s)}{\left(\sum_{i=1}^{n} w_i(s)^2 Y_i(s)\right)^2}$$

Good properties: This estimator is biased but convergent and asymptotically unbiased.

[2] Maja Pohar Perme, Janez Stare, and Jacques Estève. "On Estimation in Relative Survival". In: *Biometrics* 68.1 (June 2011), pp. 113–120. ISSN: 0006-341X. DOI: 10.1111/j.1541-0420.2011.01640.x.

Sidenote: Unicode is beautiful.

One big advantage with Julia is the how easy to read and comprehensible the syntax is.

```
const PoharPerme = NPNSEstimator{PoharPermeMethod}

function Λ!(::Type{PoharPermeMethod}, ∂Nₑ, Yₑ, ∂Nₚ, Yₚ, ∂V, T, Δ, age, date, rate_preds, ratetable, grid, ∂t)
    for i in eachindex(age)
        Tᵢ = searchsortedlast(grid, T[i])
        Λₚ, wₚ, rtᵢ = 0.0, 1.0, ratetable[rate_preds[i,:]...]
        for j in 1:Tᵢ
            λₚ       = daily_hazard(rtᵢ, age[i] + grid[j], date[i] + grid[j])
            ∂Λₚ      = λₚ * ∂t[j]
            Λₚ      += ∂Λₚ
            wₚ       = exp(Λₚ)
            ∂Nₚ[j] += ∂Λₚ * wₚ
            Yₑ[j] += wₚ
        end
        ∂Nₑ[Tᵢ] += wₚ * Δ[i]
        ∂V[Tᵢ]  += wₚ^2 * Δ[i]
    end
    Yₚ .= Yₑ
end
```

Figure 1: The Pohar Perme function in the `NetSurvival.jl` package looks like the formula itself.

Note: The `daily_hazard` function is thoroughly optimized. 60% of runtime in exp !

Gra□éo's log-rank-type test

The Gra□éo log-rank test[3] is designed to compare net survival functions across multiple groups. There is a stratified version of this test as well.

Definition (Gra□éo log-rank test)

The null H_0 hypothesis tests the following assumption:

$$\forall t \in [0, T], \quad \Lambda_{E,g_1}(t) = \Lambda_{E,g_2}(t) = ... = \Lambda_{E,g_k}(t)$$

where $G = \{g_1, ..., g_k\}$ is a partition of $1, ..., n$ consisting of disjoint groups of individuals that we wish to compare to each other.

[3] Nathalie Gra□éo, Fabienne Castell, Aurélien Belot, and Roch Giorgi. "A Log-Rank-Type Test to Compare Net Survival Distributions". In: *Biometrics* 72.3 (Jan. 2016), pp. 760–769. ISSN: 0006-341X. DOI: 10.1111/biom.12477.

Definition (Gra□éo log-rank test)

For all groups $g \in G$, let's denote the numerator and denominator of the Pohar Perme (partial) excess hazard estimators, restricted to individuals in the group, by:

$$\partial N_{E,g}(s) = \sum_{i \in g} \frac{\partial N_i(s)}{S_{P_i}(s)} - \frac{Y_i(s)}{S_{P_i}(s)} \partial \Lambda_{P_i}(s)$$

$$Y_{E,g}(s) = \sum_{i \in g} \frac{Y_i(s)}{S_{P_i}(s)}$$

$$R_g(s) = \frac{Y_{E,g}(s)}{\sum_{g \in G} Y_{E,g}(s)}$$

Then, define the vector $Z = (Z_{g_r} : \ r \in 1, ..., k-1)$ with entries:

$$Z_g(T) = N_{E,g}(s) - \int_0^T Y_{E,g}(s) \partial \hat{\Lambda}_E(s)$$

Definition (Gra□éo log-rank test)

The test statistic is then given by:

$$U(T) = Z(T)' \hat{\Sigma}_Z^{-1} Z(T)$$

where the entries of the $\hat{\Sigma}_Z$ matrix are given by:

$$\hat{\sigma}_{g,h}(T) = \int_0^T \sum_{\ell \in G} (\delta_{g,\ell} - R_g(t)) (\delta_{h,\ell} - R_h(t)) \left(\sum_{i \in \ell} \frac{\partial N_i(s)}{S_{P_i}^2} \right)$$

Under H_0, the statistic $U(T)$ is asymptotically $\chi^2(k-1)$-distributed. We reject the H_0 hypothesis when the p-value obtained is under a certain value depending on the error rate chosen, thus admitting the notable di□erence between the groups.

Definition

The crude mortality rate is the global mortality rate for a population. It can be separated by cause of death, such as:

$$M_E(t) = \int_0^t S_O(u-)\partial\Lambda_E(u)$$

Note: We denote $M_P(t) = \int_0^t S_O(u-)\partial\Lambda_P(u)du$ and $F_O(t) = M_E(t) + M_P(t)$.

Lemma (The Cronin-Feuer estimator[4])

To estimate the quantity above, we first introduce the Cronin-Feuer estimator given by:

$$\hat{M}_E(t) = \int_0^t \hat{S}_O(u-)\partial\hat{\Lambda}_E(u)$$

[4] Kathleen A Cronin and Eric J Feuer. "Cumulative cause-specific mortality for cancer patients in the presence of other causes: a crude analogue of relative survival". In: *Statistics in medicine* 19.13 (2000), pp. 1729–1740.

Estimated sample size and lifespan

The `nessie` function allows us to calculate the estimated sample size by yearly intervals as well as the average lifespan left.

Definition (Estimated Sample Size or ESS)

The estimated sample size is given by: $ESS(t) = \sum_i^N S_{P_i}(t)$

Definition (expected lifespan)

Due to the constance of the hazard rates on each cell of the lifetable, the life expectation can be computed through the following formula[5]:

$$\mathrm{E}(P) = \int_0^\infty S_P(t)\partial t = \sum_{j=0}^{\infty} \frac{S_P(t_j)}{\lambda_P(t_j)} \left(1 - e^{-\lambda_P(t_j)(t_{j+1}-t_j)}\right)^{-1}$$

Note: This feature is implemented in the `RateTables.jl` package and depends on the `Distributions.expectation` function.

[5] Per Kragh Andersen. "Life years lost among patients with a given disease". In: *Statistics in medicine* 36.22 (2017), pp. 3573–3582.

Comparison with the R/C++ standard

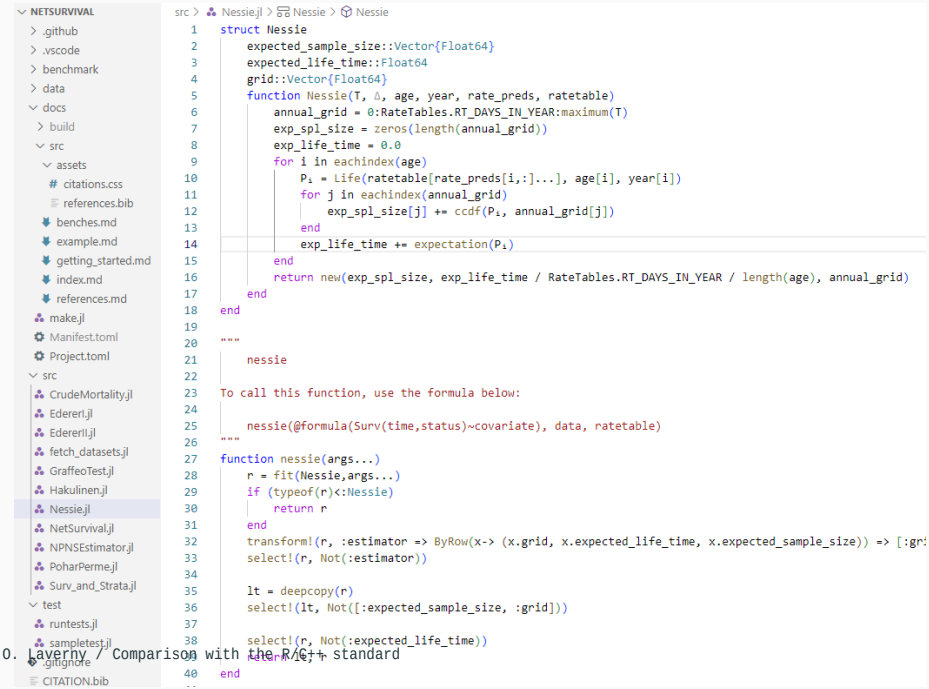
With Julia, the code is concise and easy to read, half of it is docs:

NetSurvival.jl files

Lines of code: `R` VS `Julia`

With `Julia`, the code is concise and easy to read, half of it is docs:

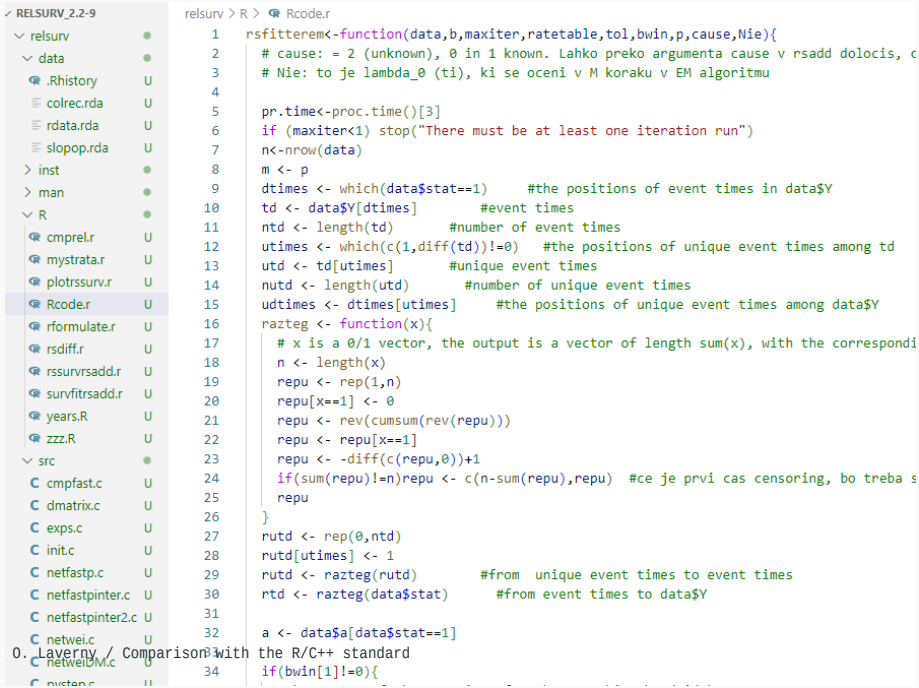
NetSurvival.jl files



Lines of code: R VS Julia

On the other hand, most of the R and C++ files from R::relsurv span over thousands of lines of code:

Rcode.r file



Lines of code: R VS Julia

On the other hand, most of the R and C++ files from R::relsurv span over thousands of lines of code:

Rcode.r file

Lines of code: R VS Julia

On the other hand, most of the R and C++ files from R::relsurv span over thousands of lines of code:

Rcode.r file

Lines of code: R VS Julia

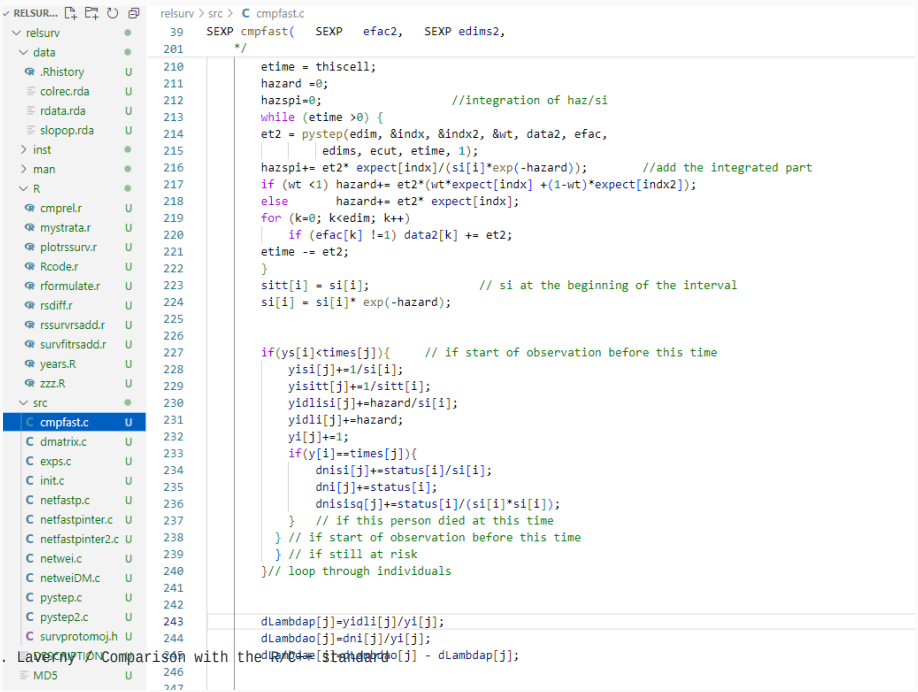
On the other hand, most of the R and C++ files from R::relsurv span over thousands of lines of code:

Rcode.r file

Lines of code: R VS Julia

On the other hand, most of the R and C++ files from R::relsurv span over thousands of lines of code:

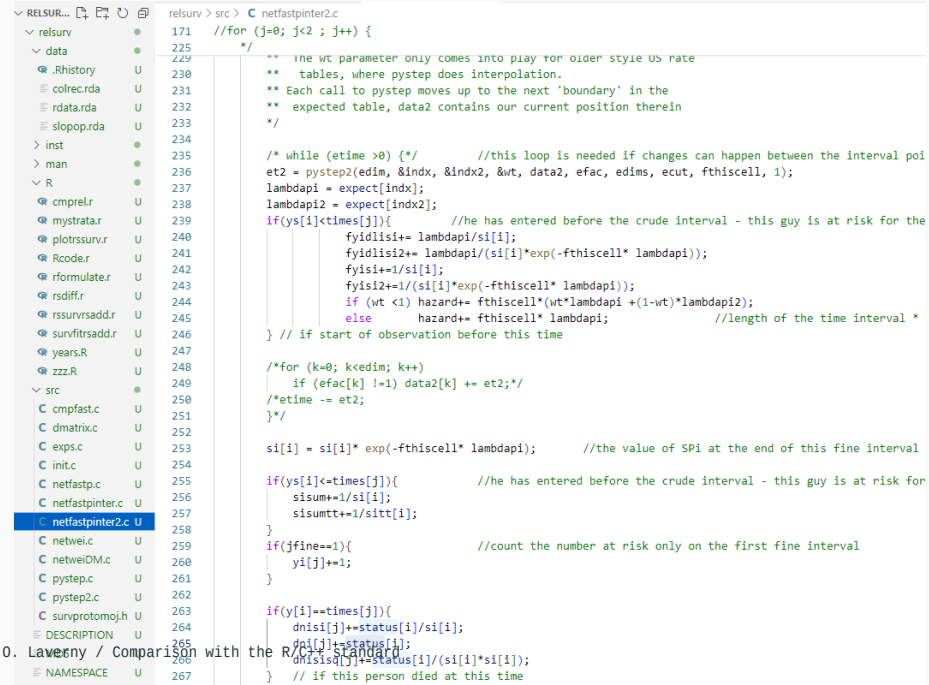
C code



Lines of code: R VS Julia

On the other hand, most of the R and C++ files from R::relsurv span over thousands of lines of code:

C code



Count Lines of Code

The `cloc` software yields the following numbers for each repository:

Language	files	blank	comment	code
R	40	883	1793	6096
C	11	284	526	1222
C/C++ Header	1	11	6	25
Total	52	1178	2325	7343

Table 1: Summary of the count of lines of code for different languages in `relsurv`

Language	files	blank	comment	code
Julia	16	151	143	547

Table 2: Summary of the count of lines of code in `NetSurvival.jl`

A glimpse in our benchmarks for a few standard algorithms:

	Unstratified	Stratified
Pohar Perme ($\partial\Lambda_E$)	20.8431	20.1461
Ederer I ($\partial\Lambda_E$)	7.216	4.1363
Ederer II ($\partial\Lambda_E$)	29.2397	29.0399
Gra□éo (log-rank-type test)	13.1556	18.156

Table 3: Runtime multipliers comparing `NetSurvival.jl` to `R::relsurv`, computed on a i9-13900 processor. The data used is the `colrec` dataset and the `slopop` mortality table.

Example: The Pohar Perme function on `Julia` takes ≈ 0.11 seconds to run whereas `R` takes ≈ 2.27.
Remark: One key advantage the `NetSurvival.jl` package has on `R::relsurv` is the function that fetches the daily hazard rates from life tables and matches them to the individuals from the dataset, hosted in the `JuliaSurv/RateTables.jl`. *Our implementation is blazing fast w.r.t. the original C++ one.*

Real data showcase

Data: `colrec` and `slopop`

Cohort: `colrec`

(i) 5971 patients

(ii) Colon or rectal cancer diagnosis between 1994 and 2000

(iii) 7 variables: age, year, sex, status, follow-up time, cancer stage, and cancer site.

(iv) Sourced from the Slovenian cancer registry

Rate table: `slopop`

(i) Slovenian mortality table

(ii) Includes information on age, year, and sex.

(iii) Extracted from o□cial census mortality rates.

Let's take a closer look at those variables:

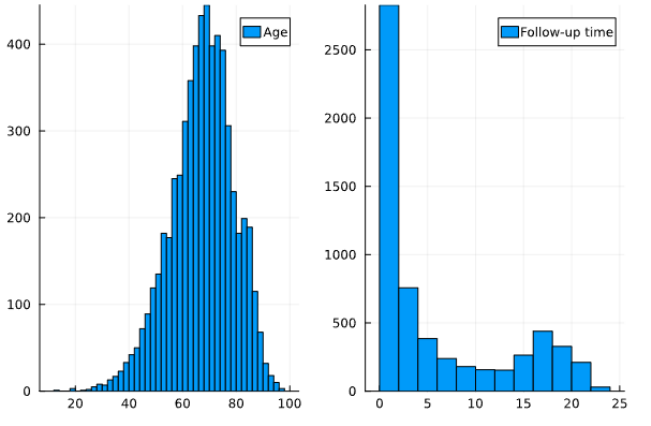


Figure 4: Histograms showing the distributions of age and the follow-up time (in years).

Applying the Pohar Perme estimator

Given the cohort `colrec` and the related mortality table `slopop`, we will apply the Pohar Perme estimator using:

Julia code

```
using NetSurvival, DataFrames, RateTables
pp_estimator = fit(PoharPerme, @formula(Surv(time,status)~1),colrec,slopop)
```

By compiling the code above, we get a table with the net survival probability in daily intervals. Let's plot the results for better visualization.

Output of the Pohar Perme estimator

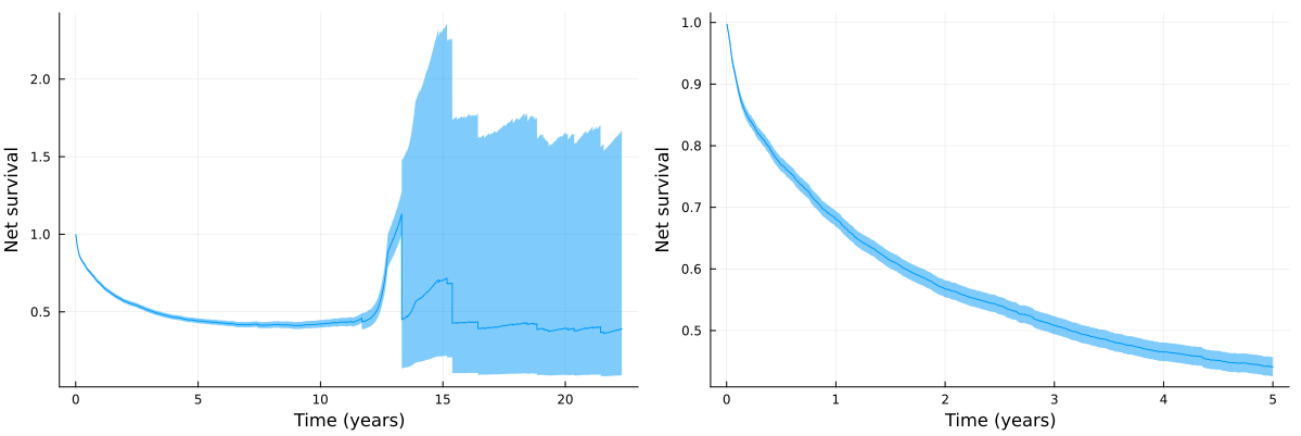


Figure 5: Pohar Perme net survival estimator. Right: only the first 5 years.

The graph above shows that, with time, the estimator loses a lot of its accuracy. Let's find out why.

nessie function output: ess object

Looking at the previous histograms, we can see that the age variable is important. The nessie function gives an estimated of expected sample size w.r.t. cancer only:

Julia code

```
elt, ess = nessie(@formula(Surv(time,status)~age65), colrec, slopop)
```

Output:

Year	Young	Old
1	2352.0	3619.0
2	2323.53	3389.0
...	...	...
23	1419.49	316.588

Table 4: Estimated sample size by year for patients above and under 65 years old.

Again, for a better understanding of these values, we present them in the graph below:

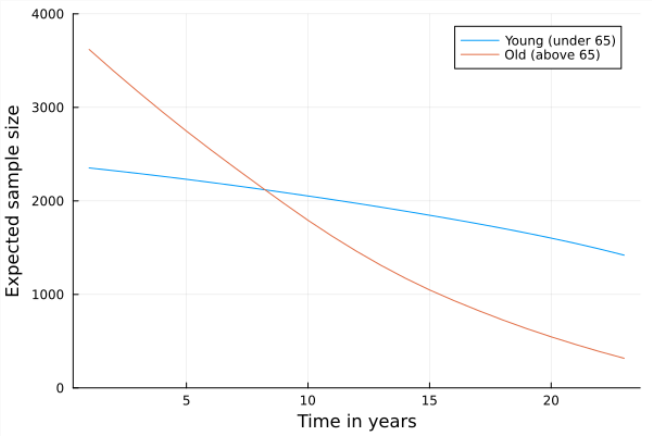


Figure 6: Graph representation of the estimated sample size for younger and older patients in yearly intervals.

Conclusion: We choose to censor the data after 5 years.

As for the expected years left for these two groups, we get the output below:

Age category	Expected life time
young	24.7882
old	10.2949

Table 5: Expected life time for patients grouped by age.

Around 5 years into the study, the crude mortality is calculated:

Julia code

```
CrudeMortality(pp_estimator5)
```

Output at time = 5 years:

$\hat{F}_O(5)$	$\hat{M}_E(5)$	$\hat{M}_P(5)$
0.645	0.531	0.114

Table 6: Crude mortality at year 5.

This shows that of the 64% patients that have died, 53% is due to colorectal cancer while 11% is due to other causes.

When applying the Gra□éo test, we find that only two variables carry real significance in the study.

Julia code

```
test_age = fit(GraffeoTest, @formula(Surv(time5,status5)~age65), colrec, slopop)
test_stage = fit(GraffeoTest, @formula(Surv(time5,status5)~stage), colrec, slopop)
```

They are unsurprisingly `age65` and `stage`, with p-values of $1.85 * 10^{-18}$ and $7.18 * 10^{-237}$ respectively for the first 5 years of the study.

Conclusion on the example

Conclusion: In the `colrec` dataset, and with reference to the `slopop` mortality table, age and cancer stages are, unsurprisingly, the more important variables in a net survival context. The older and the more advanced the cancer is, the lower the risk of survival.

Conclusion

A note on JuliaSurv

The JuliaSurv organisation on GitHub was created to house all the packages related to survival analysis in one place in an attempt to grow the Julia general registry within that context, and to keep track of older packages.

So far, it comprises:

(i) NetSurvival.jl

(ii) RateTables.jl

(iii) SurvivalBase.jl

(iv) SurvivalDistributions.jl.

More to come in the near future!

Conclusion

To summarize:

(i) Survival analysis deals with (*censored*) data where the variable of interest is the time until a certain event occurs (e.g., death).

(ii) In some specific cases, in particular for cancer registries, the exact cause of death of each individual is unavailable and/or unreliable.

(iii) Relative Survival is a theory built to handle this issue.

(iv) Our `Julia` implementation is easier to read, more concise, and faster than `R`'s, thus, making it future-proof.

(v) The `JuliaSurv` organisation has a bright future ahead!

Contributions to `NetSurvival.jl` as well as to `JuliaSurv` are more than welcome!

`JuliaSurv/NetSurvival.jl`

Star it on Github :)