Julia and Optimization Days 2024

lundi 28 octobre 2024 - mercredi 30 octobre 2024

Toulouse

Recueil des résumés

Contents

7

Wind Energy Harvesting using a Kite

Auteur: Antonin Bavoil[1]

[1] *Université Côte d'Azur, CNRS, Inria, LJAD, McTAO*

Auteur correspondant antonin.bavoil@univ-cotedazur.fr

Using kites to collect wind power and generate energy has been intensively studied in the last decade, see *e.g.* the survey by M. Diehl *et al.* in [1]. In the framework of the KEEP (Kite Electrical Energy Power) funded by CNRS and gathering researchers from ENSTA Bretagne (well acquainted with the topic after previous studies on kites [2], most notably for boats [3]) and Université Côte d'Azur, we are interested in the analysis of a simple device composed of a kite attached to an arm; having the kite running along a well chosen curve will move the arm and generate electric power. We first build a simple point-mass mechanical model where the kite motion is prescribed to a conical surface modelled on an eight curve. The resulting differential equation is a dimension 2 second order ODE, which rapidly converges to a limit cycle. We perform optimisation on the limit cycle to obtain maximum average power and report on the sensitivity of the power generated by this cycle with respect to the design parameters.

Ongoing work consists in devising an unconstrained control model for the arm + kite device, and in verfying that plugging the force extracted from the previous model into the new controlled one indeed allows to initialise the search for a *controlled* periodic cycle.

We extensively use julia capabilities to perform integration (DifferentialEquations.jl, DiffEqCallback.jl), automatic differentiation (ForwardDiff.jl), root finding (NonlinearSolve.jl), and optimisation (ADNLPModels.jl).

References

<span id="diehl-2013a">[1] Ahrens, U.; Diehl, M.; Schmehl, R. (eds.) Airborne Wind Energy, Springer, 2013</span>

<span id="desenclos-2023a">[2] Desenclos, K.; Nême, A.; Leroux, J.-B.; Jochum, C. A novel composite modelling approach for woven fabric structures applied to leading edge inflatable kites. *Mech. Composite Materials* **58** (2023), no. 6, 867—882.</span>

<span id="podeur-2018a">[3] Podeur, V.; Merdrignac, D.; Behrel, M.; Roncin, K.; Fonti, C.; Jochum, C.; Parlier, Y.; Renaud, P. Fuel economy assessment tool for auxiliary kite propulsion of merchant ship. *Houille Blanche* **1** (2018), 5—7.</span>

9

Automatic Differentiation - Julia's most confusing superpower?

Auteur: Guillaume Dalle[1]

[1] *EPFL*

Auteur correspondant guillaume.dalle@epfl.ch

It's 2024, are you still computing gradients by hand? Shame on you.
Thanks to Automatic Differentiation (AD), you can build awesome optimization or machine learning algorithms and let the computer worry about derivatives on its own. Well, mostly.

In Python, life is simple: choose an AD framework first (PyTorch or JAX), then write some code in that framework, and it should work out okay.

In Julia, life is fun: write the code first, then pick an AD package to differentiate it (ForwardDiff, Zygote, Enzyme, etc.). This paradigm is both more flexible and more dangerous, because not all code can be differentiated by every tool.

Julia's diverse AD ecosystem is often daunting for newcomers. In this talk, I will try to bring some clarity by explaining

- How AD works under the hood
- Which tools to use it in your daily work
- What to do if things go sideways

10

Welcome by organisers

11

Writing fast Julia

Auteur correspondant guillaume.dalle@epfl.ch

Writing fast Julia

From the example of a gradient descent for linear regression, we will measure performance, benchmark, enable type inference, reduce memory allocations
and fix the problems.

12

Introduction to Julia

Auteur correspondant francois.pacaud@mines-paristech.fr

13

JuMP, math heuristics, and decomposition algorithms

Auteur: ThomasJacob Riis Stidsen[1]

[1] *Technical University of Denmark*

14

Graph Neural Networks

Auteur correspondant aurora.rossi@inria.fr

Graph Neural Networks have been very successful in solving machine learning tasks such as classification and prediction at the level of nodes, edges, and graphs. This talk will provide a comprehensive overview of GNN architectures, from standard graphs to heterogeneous graphs, which model various relationships among different types of nodes, and temporal graphs, which incorporate dynamic changes in relationships over time. We will present practical applications in various domains demonstrating their versatility and impact.

16

Julia for simulation using a production planning model for helping SMEs in Decision Making

Auteurs: Caroline THIERRY None; François GALASSO None; Jérémy TRAVERSAC None; Pascale ZARATE None

Auteur correspondant jeremy.traversac@iut-rodez.fr

Decision-makers in Small and Medium Enterprises (SMEs) rely on available data and their environment knowledge to plan activities. In doing so, they may have contradictory criteria or insights according to different decision-makers. While Group Decision Support Systems exist in literature and industrial practice as well, their use can be complex and time-consuming for decision-makers.
In order to support decision makers, a framework is proposed through a Multi-Period Mixed Integer Linear Programming (MILP) production planning model that simulates the behaviour of a real company. Based on a real industrial problem, this model combines dynamic parameters (production capacity, forecasted demand, etc.) with static parameters (Bills of Materials (BOMs), deliveries, supplier capacity, etc.).
The model is implemented in Julia, using JuMP and the Cbc solver. First, decision-makers fill in the relevant data and parameters, which are then processed for being used by the solver. Decision variables such as production, deliveries, purchases, inventory, and back-orders are defined to simulate the company's behaviour at different levels. Inventory equilibrium and Bills of Materials (BOMs) are incorporated into the constraints in order to generate the production and supply plans over different time periods. These variables and constraints affect the company's expected benefit, which is incorporated into the model objective function.
The model serves as a simulation tool that generates production data, which can be used to complete Key Performance Indicators (KPIs) for each decision-maker. Both data and KPIs provides the necessary data for efficient group decision-making. The proposed simulation tool also enables decision-makers to test different scenarios, allowing them to better align outcomes with their risk tolerance and behavior.

17

Interpretable stochastic weather generator, application to a crop model, and climate change analysis

Auteur: David Métivier[1]

[1] *MISTEA - INRAE Montpellier*

Auteur correspondant david.metivier@inrae.fr

The challenges of climate change force industrials to carefully analyze the resilience of their assets to anticipate future weather conditions. In particular, the estimation of future extreme hydrometeorological events, like the frequency of long-lasting dry spells, is critical for hydropower/nuclear generation or agriculture. Stochastic Weather Generators (SWG) are essential tools to determine these future risks, as they can quickly sample climate statistics from models. In this work, the SWG described and validated with French historical data is based on a spatial Hidden Markov Model (HMM) optimized with the maximum likelihood estimator.
It generates (correlated) multisite precipitation, with a special focus on the correct reproduction of the distribution of dry and wet spells. The hidden states are viewed as global weather regimes, e.g.,

dry all over France, rainy in the north, etc. The resulting model is fully interpretable; it can even approximately recover large-scale structures such as North Atlantic Oscillations. We will show two applications to a crop model and climate change scenarios.
This talk will be the occasion to describe the associated Julia package StochasticWeatherGenerators.jl and a few associated personal Julia anecdotes and questioning.

18

Optimizing the Future: From Energy Optimization to Open-Source Solver Opportunities

Auteurs: Harley Mackenzie[1]; Rory Yarr[2]

[1] *Managing Director HARD software*

[2] *Intern*

Auteurs correspondants: harleymackenzie@gmail.com, roryyarr314@gmail.com

This presentation introduces the use of Julia and JuMP in the development of OptiGen, a next-generation optimizer tailored for renewable energy generation and battery management within the Australian National Electricity Market (NEM). OptiGen is designed to optimize the performance of utility-scale batteries and DC-coupled wind and solar plants, catering to both scheduled and non-scheduled generation.

Additionally, this talk will introduce juLinear.jl, an open-source linear programming solver project supported by HARD software. While not a direct consequence of OptiGen's development, juLinear.jl was created to address the need for a deeper understanding of how linear programming solvers function and how they can be improved. Julia's easily understandable code and mathematical syntax make it an ideal platform for implementing and experimenting with optimization algorithms. The project provides a foundation for researchers, developers, and the Julia/JuMP community to contribute to the development of solver techniques and innovations, enabling further research and exploration in the field.

By connecting the real-world applications of OptiGen with the foundational goals of juLinear.jl, this presentation highlights the growing importance of open-source tools in the evolution of energy optimization, benefiting both the Australian NEM and broader international applications and free access for all.

19

Tests of uniformity

Auteur: Claire Brécheteau[1]

[1] *LMJL*

Auteur correspondant claire.brecheteau@ec-nantes.fr

I will present a Julia package. This package proposes two statistical tests aiming at testing uniformity of samples of data points on homogeneous compact Polish spaces. Such tests are based on the computation of nearest neighbours distances.
I will show numerical results on the flat torus, the circle, the sphere, and a subset of the Poincaré disk.

20

Model and solve optimal control problems with ODE's in Julia

Auteurs: Jean-Baptiste Caillau[1]; Joseph Gergaud[2]; Olivier Cots[3]; Pierre Martinon[4]

[1] *Université Côte d'Azur, CNRS, Inria, LJAD*

[2] *Université de Toulouse, INP-ENSEEIHT-IRIT*

[3] *Toulouse Université*

[4] *CAGE team, Inria Paris*

Auteurs correspondants: olivier.cots@toulouse-inp.fr, joseph.gergaud@toulouse-inp.fr, pierre.martinon@inria.fr, jean-baptiste.caillau@univ-cotedazur.fr

We first introduce in this tutorial the indirect shooting method and the direct method in numerical optimal control on a basic example. We focus on the use of Julia packages to implement such methods: JuMP.jl \[1\], OptimalControl.jl \[2, 3, 4\], NonlinearSolve.jl \[5\], MINPACK.jl \[6\]. Then, we present how to combine direct and indirect methods to solve an optimal control problem with a Bang-Singular-Bang optimal structure. The idea is to use the direct method to determine the optimal structure and find an initial guess, respectively to define the shooting function and solve the associated shooting equations. Automatic differentiation is crucial at any step of the resolution as we will see.

\[1\]: https://github.com/jump-dev/JuMP.jl
\[2\]: https://github.com/control-toolbox
\[3\]: JuliaCon 2023
\[4\]: Caillau, J.-B., Cots, O., Gergaud, J., Martinon, P., & Sed, S. OptimalControl.jl: a Julia package to model and solve optimal control problems with ODE's. https://doi.org/10.5281/zenodo.13336563
\[5\]: https://github.com/SciML/NonlinearSolve.jl
\[6\]: https://github.com/sglyon/MINPACK.jl

21

Ring Star Problems Solver

Auteur: Julien KHAMPHOUSONE[1]

Co-auteurs: André ROSSI [2]; Fabian CASTAÑO [3]; Sonia TOUBALINE [2]

[1] *Orange INNOV*

[2] *CNRS LAMSADE UMR 7243, Université Paris Dauphine–PSL, Paris, France*

[3] *Research and Development Team, Frubana, Bogota, Colombia*

Auteurs correspondants: julien.khamphousone@orange.com, sonia.toubaline@dauphine.psl.eu, fancagi@gmail.com, andre.rossi@lamsade.dauphine.fr

The Ring Star Problem consists in minimizing the cost of a spanning network made of a cycle called the ring to link a subset of nodes called hubs (to be determined), and to connect each terminal, i.e., non-hub, to a single hub at minimum cost. The package RingStarProblems.jl solves a resilient variant of the Ring Star Problem named 1-R-RSP in which the failure of a node in a given subset of so-called uncertain nodes triggers two corrective operations to repair a Ring Star network. The first operation repairs the ring and the second operation reconnects the non-hubs that were originally connected to the failing hub. For a given real positive input parameter F which represents the total duration of breakdowns during the considered time horizon, the objective is to minimize the Ring Star Problem cost plus the costs incurred by the corrective operations when a hub fails in the worst case during F unit of time. We model this problem as an Integer Linear Program (ILP), that is also addressed with a Branch-and-Benders-cut decomposition. Several enhancements to both the ILP and the Branch-and-Benders-cut (B&BC) algorithm are also implemented. The registered package RingStarProblems.jl is

coded in the Julia Mathematical Programming Language, it solves 1-R-RSP to optimality using either ILP or B&BC. The package uses callbacks through Julia Mathematical Programming (JuMP) and is solver-agnostic, which means it can be used with Gurobi, GLPK, CPLEX or any JuMP solver that can support Lazy Constraints Callbacks. The package RingStarProblems.jl is highly parametrized, can plot the obtained solutions to PDF files, and other features will be added in the future.

22

Co-Design of Microgrids with Renewable Integration: A Modular Julia-based Tool and Applications

Auteur: Corentin Boennec[1]

Co-auteurs: Bruno Sareni [1]; Camille Bergougnoux [1]; Evelise de Godoy Antunes [1]; Sandra Ulrich Ngueveu [2]

[1] *LAPLACE UMR CNRS-INPT-UPS, Universite de Toulouse, Toulouse, France*

[2] *LAAS-CNRS, Université de Toulouse, CNRS, INP, Toulouse, France*

Auteurs correspondants: antunes@laplace.univ-tlse.fr, bergougnoux@laplace.univ-tlse.fr, sareni@laplace.univ-tlse.fr, ngueveu@laas.fr, boennec@laplace.univ-tlse.fr

The integration of renewable energy sources in our energy systems is a global objective to mitigate global warming by reducing CO2 emissions, especially when replacing fossil fuels. International agencies consistently highlight the importance of increasing renewable energy penetration to reduce temperature increases over the coming decades. In the context of smart microgrids, Distributed Energy Systems (DES) allow the local integration of renewable energy sources, enabling both on-site generation and consumption. However, co-designing their sizing (types and quantity of units) and operation (optimal use of production, conversion, and storage units over time) is challenging for several reasons. The problem is inherently bi-level, with long time horizons (decades), fine temporal granularity (hourly or less to capture renewable dynamics), and significant uncertainties (stemming from environmental and modeling aspects).

Despite these complexities, this problem has been studied for a long time, evolving with the introduction of new elements, such as non-dispatchable energy sources and advanced storage technologies. However it has been and still is common to perform multiple simplifications in order to build an Energy System Optimization Model (ESOM).

To address these challenges, we introduce our modular julia-based tool devoted to the co-design of microgrids. It allows flexible customization of key elements, such as microgrid architecture, input data, component models, and optimization techniques.

This poster provides an overview of the general problem, highlights the features of our julia-based tool, and presents case studies using it. These include a multi-energy network for Toulouse Blagnac airport, a study on the impact of modeling simplifications on decision-making criteria, an in depth study on modeling and integration into microgrids of second-life batteries, and the co-design of an electric vehicle charging station.

23

Solving optimal control problems with loss control regions using OptimalControl.jl

Auteurs: Anas Bouali[1]; Loïc Bourdin[2]; Olivier Cots[3]; Térence Bayen[4]

[1] *INRAE*

[2] *Institut de recherche XLIM*

[3] *Toulouse Université*

[4] *Laboratoire de Mathématiques d'Avignon*

Auteurs correspondants: olivier.cots@toulouse-inp.fr, terence.bayen@univ-avignon.fr, loic.bourdin@unilim.fr, anas.bouali@inrae.fr

In this presentation, we address optimal control problems involving *loss control regions*. In this context, the state space is partitioned into disjoint sets referred to as regions, which are classified into two types: *control regions* and *loss control regions*. When the state belongs to a control region, the control is permanent (i.e., the control value can be modified at any time). On the other hand, when the state belongs to a loss control region, the control must remain constant, equal to the last assigned value before the state enters the loss control region, and this value is kept until the state exits this region [1].

The objective of this presentation is twofold:

(i) First, we introduce a direct method that is based on a regularization technique and the consideration of additional states and controls. This allows to transform the initial problem with discontinuous dynamics to a (classical) optimal control problem with smooth dynamics. Furthermore, thanks to OptimalControl.jl [4], we are able to obtain the optimal solution of the regularized problem and also extract the adjoint vectors.

(ii) Second, based on first-order necessary optimality conditions in a loss control framework (see [1] and [3]), we develop a shooting method that includes new conditions such as the jumps of adjoint vectors at each crossing times and the averaged Hamiltonian gradient condition. This approach relies on the direct method that provides a good initialization of adjoint vectors, crossing times, and adjoint vector jumps, which allows to solve the initial problem.

Finally, we demonstrate the effectiveness of this numerical approach through various illustrative examples (for detailed documentation, we refer to LossControl.jl).

References

[1]: T. Bayen, A. Bouali, L. Bourdin & O. Cots, "Loss control regions in optimal control problems," Journal of Differential Equations, 2024, vol. **405**, p. 359-397.

[2]: Bayen, T., Bouali, A., & Bourdin, L. "The Hybrid Maximum Principle for Optimal Control Problems with Spatially Heterogeneous Dynamics is a Consequence of a Pontryagin Maximum Principle for L^1_{square}local Solutions," SIAM Journal on Control and Optimization, 2024, vol. **62**, no 4, p. 2412-2432.

[3]: T. Bayen, A. Bouali, & L. Bourdin, "Optimal control problems with non-control regions: necessary optimality conditions," IFAC-PapersOnLine, vol. 55, no. 16, pp. 68-73, 2022.

[4]: J.-B. Caillau, O. Cots, J. Gergaud, P. Martinon, & S. Sed, "OptimalControl.jl: a Julia package to model and solve optimal control problems with ODE's," DOI: 10.5281/zenodo.13336563

24

Nonlinear optimization with Artelys Knitro in Julia

Auteurs: Florian Fontan[None]; Marvin Stanczak[1]; Maxime Dufour[None]; Renaud Saltet[None]

[1] *Artelys*

Auteurs correspondants: maxime.dufour@artelys.com, florian.fontan@artelys.com, renaud.saltet@artelys.com, marvin.stanczak@artelys.com

Introduction

Artelys Knitro is a mathematical programming solver for nonlinear and mixed-integer nonlinear problems. As input, it accepts linear structures, quadratic structures and black-box functions, with if possible, their first and second-order derivatives. Knitro relies on derivative-based algorithms to find locally optimal solutions. Knitro finds the global optimum for convex problems. For non-convex problems, Knitro converges to a first order stationary point (e.g. local optimum) for continuous models and is a heuristic for mixed-integer problems.

Recent developments in Artelys Knitro

For nonlinear continuous problems, Artelys Knitro includes two interior point algorithms, a sequential linear quadratic programming algorithm and a sequential quadratic programming (SQP) algorithm. For mixed-integer nonlinear problems, Artelys Knitro includes a nonlinear branch-and-bound algorithm, a Quesada-Grossman branch-and-bound algorithm, and a mixed-integer sequential quadratic programming algorithm. The new version Artelys Knitro 14.1 provides specific improvements on quadratic models (QP) and quadratically constrained quadratic models (QCQP), both convex and non-convex. On non-convex continuous models, users will notice similar solutions returned in shorter time thanks to the automatic termination of multistart when the probability of finding new local solutions is small. Furthermore, high quality solutions can be obtained on non-convex MINLP models thanks to a new behavior when activating multistart. Finally, Knitro Julia interface now supports newer versions of Julia 1.10.2+.

Tutorial in Julia

In this talk, we offer to present a tutorial of Artelys Knitro in Julia presenting both the modeling directly from the Knitro Julia interface (on a geometric application) or through JuMP (for hydropower plant optimization). The audience will be able to connect to a small server setup for the event with a Jupyter notebook without having to install anything on their laptop.

Bibliography

1. P. Bonami (2011). Lift-and-Project Cuts for Mixed Integer Convex Programs. In: O. Günlük, GJ. Woeginger (eds), Integer Programming and Combinatorial Optimization. [link]

2. A. Gleixner, L. Gottwald, A. Hoen (2023). PaPILO: A Parallel Presolving Library for Integer and Linear Optimization with Multiprecision Support. INFORMS Journal on Computing. [link]

3. Luteberget, B., Sartor, G. Feasibility Jump: an LP-free Lagrangian MIP heuristic. Math. Prog. Comp. 15, 365–388 (2023). [link]

4. R. H. Byrd, J. Nocedal, and R.A. Waltz (2006). KNITRO: An integrated package for nonlinear optimization, In G. di Pillo and M. Roma, editors, Large-Scale Nonlinear Optimization, pages 35–59. Springer. [link]

5. Github Artelys Knitro tutorials - https://github.com/Artelys/knitro-modeling-examples

6. Artelys website - https://www.artelys.com/news/artelys-knitro-14-1-delivers-very-quick-solutions-on-non-convex-models/

25

Approximation of the Shapley value with sampling

Auteurs: Francois Lamothe[1]; Sandra Ulrich Ngueveu[2]

[1] *Laas-cnrs*

[2] *Laplace*

Auteurs correspondants: ffrancois.lamothe@gmail.com, ngueveu@laas.fr

Distributing the total cost of a service-providing system among its users is a challenging topic in game theory. Shapley proposed in 1953 a cost-sharing mechanism nowadays known as the Shapley value. It is usually interpreted as assigning to each player the cost this player induces on the system. This cost assignment has many desirable properties but it is very time-consuming to compute exactly when the number of users is large. Thus, a large effort has been directed toward its approximation. Many Shapley value algorithms use sampling to compute a good approximation. In this package, we re-implement in Julia many algorithms from the literature which rely on sampling. We also implement several algorithms which were shown to improve the state of the art in the scientific paper corresponding to this package. Finally, We implement a framework to automatically generate systems which enable to test and compare Shapley approximation algorithms.

26

LinA.jl : A julia package to compute piecewise linear approximations

Auteurs: Julien Codsi[None]; Sandra Ulrich Ngueveu[1]; Bernard Gendron[None]

[1] *Toulouse INP-ENSEEIHT / LAAS-CNRS*

Auteurs correspondants: julien.codsi@umontreal.ca, sandra.ulrich.ngueveu@laas.fr

We present new algorithms that can solve the problem of approximating and over/under-estimating univariate functions with piecewise linear (PWL) functions with the minimum number of linear segments given a bound on the pointwise approximation error allowed. These new algorithms can solve the problem in quasi-logarithmic time on a very broad class of errors types. Such algorithms find many applications, for example related to solving certain classes of (mixed-integer) nonlinear and nonconvex programming (MINLP) problems by mixed-integer linear programming (MILP) techniques. An efficient implementation of our algorithms is available as a Julia package that will be presented.

27

MadNLP.jl : A condensed-space interior-point method for nonlinear programming on GPUs

Auteurs: Alexis Montoison[1]; François Pacaud[2]; Sungho Shin[3]

[1] *Argonne National Lab*

[2] *Mines Paris - PSL*

[3] *MIT*

Auteurs correspondants: sushin@mit.edu, amontoison@anl.gov, francois.pacaud@mines-paristech.fr

We present a novel interior-point method to solve nonlinear programs on graphical processing units (GPUs). The classical interior-point method solves a sequence of symmetric indefinite linear systems, or Karush-Kuhn-Tucker (KKT) systems, that are increasingly ill-conditioned as we approach the solution. Solving a KKT system with traditional sparse factorization methods involves numerical pivoting, making parallelization difficult. A remedy is to condense the KKT system into a symmetric positive-definite matrix and solve it with a Cholesky factorization, stable without pivoting. We have implemented the condensed-space interior-point method on the GPU using MadNLP.jl, an optimization solver interfaced with the NVIDIA sparse linear solver cuDSS and with the GPU-accelerated modeler ExaModels.jl. Our experiments on large-scale OPF and optimal control instances reveal that GPUs can attain up to a tenfold speed increase compared to CPUs.

28

Demystifying metaprogramming and package precompilation

Auteur correspondant cedric.bel@hotmail.fr

Metaprogramming in Julia is often presented as a powerful language feature, implemented via macros and types representing Julia code. Indeed, it allows the expression of complex constructs in a simple way, such as domain-specific languages (DSL) expressing mathematical problems. Yet, the term metaprogramming is somewhat overloaded and unclear as to what it really means under the hood.
We will have a look at most of the relevant concepts in an interactive session in an effort to make it more intuitive. This should help you leverage the benefits of metaprogramming as a developer, and better understand what's going on as a user.

In the second part of this workshop, we will cover the topic of package precompilation. In recent years, there has been a large focus on the issue of TTFP - time to first plot - where Julia users were often annoyed by the large delay caused by the just-in-time compilation model of the language. Great improvements were made in recent years to grant users and developers the tools to significantly reduce this delay, to the point where it is almost a solved issue today.
There again, we will try to present the topic intuitively and present tools that developers can apply to make packages more convenient to use. We will also present a way for users to shave off some latency for workflows involving packages in your global environment.

29

NetSurvival.jl: A glimpse into relative survival analysis

Auteur correspondant rim.hajal@gmail.com

Data arising from large cancer studies often lack reliable information about the cause of death (supposed binary: death from the studied cancer or from other causes) for each individual. Relative survival analysis is a subfield of survival analysis specifically targeted at this particular type of datasets. The field's goal is to extract survival curves that only takes into account the deaths from cancer, the so-called net survival curves, for comparison purposes between different groups or directly for diagnostic purposes. For that, a few standard estimators were established in the last 50 years, backed by a wide literature.

Standard tools nowadays are composed of R packages, with underlying C and C++ routines, that are hard to read, maintain, and globally use. This package is an attempt to provide a fast, reliable, but most importantly easily maintainable package to implement standard estimators and routines from the field onto the StatsModels.jl API. The hope is that the junction with classical modeling API in Julia will allow later extensions of the existing modeling methods, with a simple interface for the practitioners.

In this talk, I present the current state of the implementation: the few tools and methods that were implemented, their integration into the Julian ecosystem, and showcase the functionalities and performance of the implemented methods. Note that in particular the use of native Julia allows for concision and readability of the code.