juliacon

SequentialSamplingModels.jl: Simulating and Evaluating Cognitive Models of Response Times in Julia

Kianté Fernandez[1], Dominique Makowski[2], and Christopher Fisher[3]

[1]University of California, Los Angeles
[2]School of Psychology, University of Sussex, Brighton, UK
[3]Independent Researcher

ABSTRACT

Sequential sampling models (SSMs) are a widely used framework describing decision-making as a stochastic, dynamic process of evidence accumulation. SSMs' popularity across cognitive science has driven the development of various software packages that lower the barrier for simulating, estimating, and comparing existing SSMs. Here, we present a software tool, SequentialSamplingModels.jl (SSM.jl), designed to make SSM simulations more accessible to Julia users, and to integrate with the Julia ecosystem. We demonstrate the basic use of SSM.jl for simulation, plotting, and Bayesian inference.

Keywords

cognitive models,response time, sequential sampling models, evidence accumulation, decision-making

1. Introduction

Sequential sampling models (SSMs) are widely used in cognitive science due to their ability to describe dissociable processes underlying a wide variety of capacities, including memory retrieval, visual perception, and decision making (Forstmann et al. (2016)). These models typically describe decision-making as a stochastic (i.e., random), dynamic evidence accumulation process which evolves until evidence for one option reaches an evidence threshold (Smith (2000); Ratcliff and McKoon; Ratcliff). By doing so, SSMs provide a generative model of joint choice-response time (RT) distributions. Figure 1 illustrates the latent evidence accumulation process for a hypothetical decision between a Margherita pizza and a pineapple pizza. Evidence accumulates stochastically until it reaches either the upper boundary, triggering the selection of the Margherita, or the lower boundary, triggering the selection of the pineapple pizza. While these models have many promising applications, their implementation and use remain a technical challenge.

The widespread interest and continuous use of SSMs across cognitive science have spurred the development of various software packages that lower the barrier for simulating, estimating, and comparing existing SSMs. These software implementations range widely, from those designed to provide maximum flexibility to end-users (Shinn et al. (2020)) to those that streamline the entire analysis of SSMs (Fengler et al.). Toolboxes have also been developed across multiple programming languages commonly used for scientific computing, including MATLAB (Vandekerckhove and Tuerlinckx (2008)), Python (Wiecki et al. (2013); Shinn et al. (2020); Fengler et al.; Murrow and Holmes (2024)), and R (**?**Stevenson et al. (2025); Ahn et al. (2017); Singmann et al.; Hartmann and Klauer (2021)).

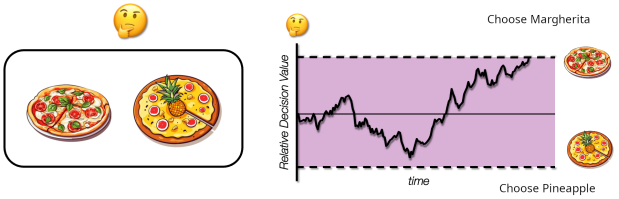

Fig. 1: An example of applying the sequential sampling modeling framework to a choice between two pizzas. The decision-maker samples evidence in favor of both options until reaching a decision boundary.

2. SequentialSamplingModels.jl

While existing methods can already simulate a wide range of SSMs across multiple languages, developments in Julia have either not been established or lack maintenance. SequentialSamplingModels.jl (SSM.jl) provides the first unified interface for simulating validated SSMs, drawn from the literature to represent those most widely used models. The package offers a user-friendly method for simulating a range of SSMs of interest to researchers. It is the first attempt to integrate a package into the Julia ecosystem for simulating and evaluating a popular model of decision-making.

3. Availability, Development, and Documentation

SequentialSamplingModels.jl is available on the Julia package registry. The package is maintained on GitHub at `https://github.com/itsdfish/SequentialSamplingModels.jl`, and the documentation can be found at `https://itsdfish.github.io/SequentialSamplingModels.jl/dev/`. For a more comprehensive exploration of the package functionalities and further details on future developments, the user is invited to consult the package README, documentation, and issues.

4. Example one: simulating choices and response times

SSMs.jl is an extension of the established functionalities within the Distributions.jl package Lin et al. (2024), designed for generating multivariate choice-RT distributions. To achieve this, a new abstract

type was defined to support both continuous and discrete values. Then, a suite of constructors were designed to accommodate multivariate distributions with mixed support, which are necessary for generating discrete choices and continuous response times. Across the defined sub-types, distributions utilizing SSMs.jl will output a NamedTuple consisting of a vector of choices and RTs. With samplers designed to integrate with Distributions.jl in mind, the constructors can leverage many of the related functions for probabilistic distributions, including sampling and likelihood evaluation. In Code Block 1, using the Diffusion Decision Model (Ratcliff et al. (2016); DDM), we provide an example of these features for simulating choices and response times from a pre-defined set of parameters, as well as computing the log-likelihood of the data. The DDM consists of four parameters:

—ν: the drift rate which represents the rate of evidence accumulation towards the upper boundary

—α: the decision threshold which represents the amount of evidence required to make a decision

—z: the starting point bias towards the upper decision boundary

—τ: non-decision time which represents the sum of encoding and motor response times.

Code 1: A code example illustrating how to create a Diffusion Decision Model object, generate simulated data, and evaluate the log likelihood of the simulated data.

```
using SequentialSamplingModels
# Create DDM distribution
dist = DDM(; ν = 1.00, α = 0.80, τ = 0.30, z = 0.50)
# Sample 10,000 simulated data from the DDM
sim_data = rand(dist, 10_000)
# compute log likelihood of simulated data
logpdf(dist, sim_data)
```

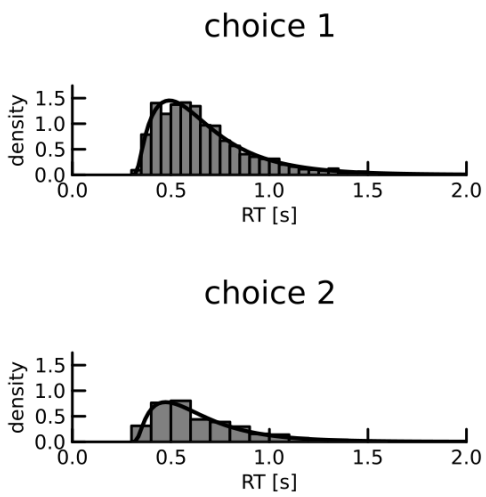

Fig. 2: RT distributions of the racing diffusion model based on Code Block 2

5. Example two: plotting RT-choice distributions and simulated traces

SSM.jl also enables plotting functionality for SSMs when `Plots.jl` (Christ et al. (2023)) is active in your Julia session. In Code Block 2, using the racing diffusion model (Tillman et al. (2020); RDM), we demonstrate how to generate two plots commonly used with SSMs. The RDM consists of the following parameters:

—ν: a vector of drift rates representing the evidence accumulation rate for each response option

—k: a positive constant added to A such that $A + k$ is the decision threshold, which represents the amount of evidence required to make a decision

—A: the maximum starting point of the diffusion process

—τ: non-decision time which represents the sum of encoding and motor response times.

The first plot, shown in Figure 2, is a histogram of joint choice-RT reaction distributions paneled by choice with a probability density overlay. The second plot illustrates five simulated trajectories of the latent evidence accumulation process of the RDM, which is shown in Figure 3. In addition, users can animate the evidence accumulation trace with the function `animate` to illustrate the model's dynamics.

Code 2: SSM.jl works with Plots.jl. The first example is a histogram of joint choice-RTs (see Figure 2), and the second example is a trace of the latent evidence accumulation dynamics (see Figure 3).

```
using SequentialSamplingModels
using Plots
# Define parameters
ν = [2,1]
k = 0.50
A = 1.0
τ = 0.30
# Create RDM distribution
dist = RDM(;ν, k, A, τ)
# Plot distribution
histogram(dist; xlims=(0, 2))
plot!(dist; t_range=range(.301, 2.5, length=100))
# Plot model simulations traces
plot_model(dist, n_sim=5,
t_range=range(.20, 1.5, length=100), xlims=(0, 1.5))
```

6. Example three: Bayesian inference

In cases where closed-form likelihoods are available, SSM.jl can also facilitate parameter estimation routines available with Turing.jl, a general-purpose probabilistic programming library (Ge et al. (2018)). Below, we provide an example of writing a Turing.jl model using a SSM constructor and conducting a simple parameter estimation exercise using the linear ballistic accumulator (Brown and Heathcote (2008); LBA). The LBA has the same parameters as the RDM, but makes different assumptions about the sources of stochasticity in the evidence accumulation process. In code example 3, we use a Markov Chain Monte Carlo (MCMC) technique called the No U-Turn Sampler (Hoffman et al., 2014) to estimate the posterior distribution of the LBA parameters from simulated data.

Code 3: Use Turing.jl to conduct paramter estimation with a SequentialSamplingModels.jl constructor

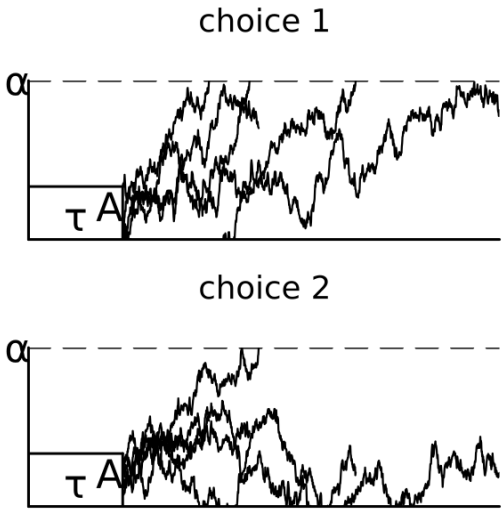

Fig. 3: Five traces of the racing diffusion model based on Code Block 2. On each simulations, the accumulators race independently until the evidence of the fastest accumulator reaches its threshold (horizontal, dashed line)

```julia
using LinearAlgebra
using Random
using SequentialSamplingModels
using StatsPlots
using Turing

Random.seed!(104)
# Simulate data
dist = LBA(ν=[3.0, 2.0], A=.8, k=.2, τ=.3)
data = rand(dist, 100)
# Define model using Turing.jl macro
@model function model_lba(data)
    ν ~ MvNormal(zeros(2), I(2))
    A ~ truncated(Normal(.8, .4), 0.0, Inf)
    k ~ truncated(Normal(.2, .2), 0.0, Inf)
    τ ~ Uniform(0.0, minimum(data.rt))
    data ~ LBA(;ν, A, k, τ)
end
# Fit model
chains = sample(model_lba(data), NUTS(1000, .85),
MCMCThreads(), 1000, 4)
# Examine convergence quality
summarystats(chains)
# Plot Markov Chain Monte Carlo (MCMC)
# trace and posterior distributions
plot(chains)
```

In addition, SSM.jl integrates with Pigeons.jl ((Surjanovic et al.)) and ParetoSmooth.jl (<cdp49@cam.ac.uk> (2021)), via Turing.jl, allowing the user to estimate the marginal likelihood and approximate leave-one-out cross-validation for Bayesian model comparison (Vehtari et al. (2017)), respectively.

7. Future Applications

A major application of SSMs is to fit models to data via parameter estimation routines. However, not all SSMs have closed-form likelihoods that can be computed efficiently, and thus simulation-based inference has recently been proposed as a solution to accommodate parameter estimation for SSMs.

In the future, we aim to incorporate approximate solutions for likelihoods with no known closed form using approximate Bayesian computation (Palestro et al. (2018)) and deep learning methods

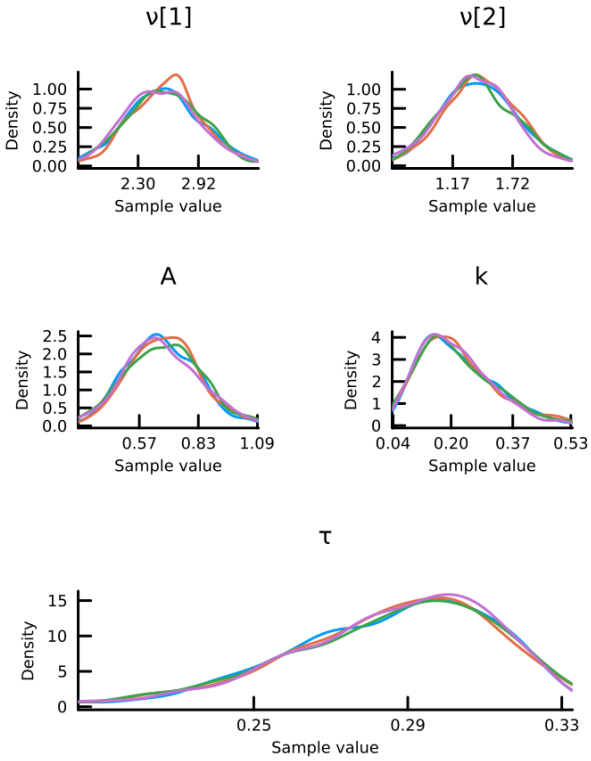

Fig. 4: Posterior distributions for the parameters of the Linear Ballistic Accumulator model based on Block 3. MCMC trace plots were removed due to space limitations.

for efficient simulation-based inference, such as likelihood approximation networks (Fengler et al. (2021)) specified in Flux.jl (Innes (2018)). This approach facilitates sampling from the posterior distribution of models without a closed-form likelihood function while maintaining a consistent syntax for end-users.

References

Woo-Young Ahn, Nathaniel Haines, and Lei Zhang. Revealing neurocomputational mechanisms of reinforcement learning and decision-making with the hBayesDM package. *Computational Psychiatry*, 1:24–57, 2017. doi: 10.1162/CPSY_a_00002.

Scott D Brown and Andrew Heathcote. The simplest complete model of choice response time: Linear ballistic accumulation. *Cognitive Psychology*, 57(3):153–178, 2008. doi: https://doi.org/10.1016/j.cogpsych.2007.12.002.

Carlos Parada <cdp49@cam.ac.uk>. Paretosmooth.jl, 6 2021. URL `https://github.com/TuringLang/ParetoSmooth.jl`.

Simon Christ, Daniel Schwabeneder, Christopher Rackauckas, Michael Krabbe Borregaard, and Thomas Breloff. Plots.jl – a user extendable plotting api for the julia programming language. 2023. doi: https://doi.org/10.5334/jors.431. URL `https://openresearchsoftware.metajnl.com/articles/10.5334/jors.431/`.

Alexander Fengler, Krishn Bera, Mads L. Pedersen, and Michael J. Frank. Beyond drift diffusion models: Fitting a broad class of decision and reinforcement learning models with HDDM. *Journal of Cognitive Neuroscience*, 34(10):1780–1805. ISSN 0898-

929X. doi: 10.1162/jocn_a_01902. URL https://doi.org/10.1162/jocn_a_01902.

Alexander Fengler, Lakshmi N Govindarajan, Tony Chen, and Michael J Frank. Likelihood approximation networks (lans) for fast inference of simulation models in cognitive neuroscience. *eLife*, 10:e65074, apr 2021. ISSN 2050-084X. doi: 10.7554/eLife.65074. URL https://doi.org/10.7554/eLife.65074.

Birte U Forstmann, Roger Ratcliff, and E-J Wagenmakers. Sequential sampling models in cognitive neuroscience: Advantages, applications, and extensions. *Annual review of psychology*, 67(1):641–666, 2016. doi: https://doi.org/10.1146/annurev-psych-122414-033645.

Hong Ge, Kai Xu, and Zoubin Ghahramani. Turing: a language for flexible probabilistic inference. In *International Conference on Artificial Intelligence and Statistics, AISTATS 2018, 9-11 April 2018, Playa Blanca, Lanzarote, Canary Islands, Spain*, pages 1682–1690, 2018. URL http://proceedings.mlr.press/v84/ge18b.html.

Raphael Hartmann and Karl Christoph Klauer. Partial derivatives for the first-passage time distribution in wiener diffusion models. *Journal of Mathematical Psychology*, 103:102550, 2021. doi: https://doi.org/10.1016/j.jmp.2021.102550.

Matthew D Hoffman, Andrew Gelman, et al. The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. *J. Mach. Learn. Res.*, 15(1):1593–1623, 2014.

Mike Innes. Flux: Elegant machine learning with julia. *Journal of Open Source Software*, 2018. doi: 10.21105/joss.00602.

Dahua Lin, David Widmann, Simon Byrne, John Myles White, Andreas Noack, Mathieu Besançon, Douglas Bates, John Pearson, John Zito, Alex Arslan, Moritz Schauer, Kevin Squire, David Anthoff, Theodore Papamarkou, Jan Drugowitsch, Benjamin Deonovic, Avik Sengupta, Seth Axen, Viral B. Shah, Brian J Smith, Glenn Moynihan, Giuseppe Ragusa, Christoph Dann, Mike J Innes, Mohamed Tarek, Michael, Martin O'Leary, and Gustavo Lacerda. Juliastats/distributions.jl: v0.25.112, September 2024. URL https://doi.org/10.5281/zenodo.13838344.

Matthew Murrow and William R Holmes. Pybeam: A bayesian approach to parameter inference for a wide class of binary evidence accumulation models. *Behavior Research Methods*, 56(3):2636–2656, 2024. doi: https://doi.org/10.3758/s13428-023-02162-w.

James J Palestro, Per B Sederberg, Adam F Osth, Trisha Van Zandt, and Brandon M Turner. *Likelihood-free methods for cognitive science*. Springer, 2018. doi: https://doi.org/10.1007/978-3-319-72425-6.

R. Ratcliff. A theory of memory retrieval. *Psychological Review*, 85:59–108. doi: 10.1037/0033-295X.85.2.59.

Roger Ratcliff and Gail McKoon. The diffusion decision model: Theory and data for two-choice decision tasks. *Neural Computation*, 20(4):873–922. ISSN 0899-7667. doi: 10.1162/neco.2008.12-06-420. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2474742/.

Roger Ratcliff, Philip L. Smith, Scott D. Brown, and Gail McKoon. Diffusion decision model: Current issues and history. *Trends in Cognitive Sciences*, 20(4):260–281, 2016. ISSN 1364-6613. doi: 10.1016/j.tics.2016.01.007. URL https://www.sciencedirect.com/science/article/pii/S1364661316000255.

Maxwell Shinn, Norman H Lam, and John D. Murray. A flexible framework for simulating and fitting generalized drift-diffusion models. *eLife*, 9, 2020. doi: https://doi.org/10.7554/eLife.56938.

Henrik Singmann, S. Brown, M. Gretton, A. Heathcote, A. Voss, J. Voss, and A. Terry. rtdists: Response time distributions. *R package version 0.4-9. URL http://CRAN. R-project. org/package= rtdists*. URL https://www.vps.fmvz.usp.br/CRAN/web/packages/rtdists/rtdists.pdf.

Philip L Smith. Stochastic dynamic models of response time and accuracy: A foundational primer. *Journal of Mathematical Psychology*, 44(3):408–463, 2000. doi: https://doi.org/10.1006/jmps.1999.1260.

Niek Stevenson, Michelle C Donzallaz, Reilly J Innes, Birte Forstmann, Dora Matzke, and Andrew Heathcote, PhD. EMC2: An r package for cognitive models of choice. 2025. doi: 10.31234/osf.io/2e4dq_v3. URL https://osf.io/preprints/psyarxiv/2e4dq_v3/.

Nikola Surjanovic, Miguel Biron-Lattes, Paul Tiede, Saifuddin Syed, Trevor Campbell, and Alexandre Bouchard-C{\textbackslash}. Pigeons.jl: Distributed sampling from intractable distributions. *Proceedings of the JuliaCon Conferences*, 7(69):139. ISSN 2642-4029. doi: 10.21105/jcon.00139. URL https://proceedings.juliacon.org/papers/10.21105/jcon.00139.

Gabriel Tillman, Trish Van Zandt, and Gordon D Logan. Sequential sampling models without random between-trial variability: The racing diffusion model of speeded decision making. *Psychonomic Bulletin & Review*, 27(5):911–936, 2020. doi: https://doi.org/10.3758/s13423-020-01719-6.

Joachim Vandekerckhove and Francis Tuerlinckx. Diffusion model analysis with matlab: A dmat primer. *Behavior Research Methods*, 40(1):61–72, 2008. doi: https://doi.org/10.3758/BRM.40.1.61.

Aki Vehtari, Andrew Gelman, and Jonah Gabry. Practical bayesian model evaluation using leave-one-out cross-validation and waic. *Statistics and Computing*, 27(5):1413–1432, Sep 2017. ISSN 1573-1375. doi: 10.1007/s11222-016-9696-4. URL https://doi.org/10.1007/s11222-016-9696-4.

Thomas V Wiecki, Imri Sofer, and Michael J Frank. Hddm: Hierarchical bayesian estimation of the drift-diffusion model in python. *Frontiers in Neuroinformatics*, 7:55610, 2013. doi: https://doi.org/10.3389/fninf.2013.00014.