BayesCog: A freely available course in Bayesian statistics and Hierarchical Bayesian Modeling for Psychological Science

2025-01-01

Summary

We present BayesCog, an openly-available online course for the computational modeling of human behaviour (cognitive modeling) using Bayesian inference. Assuming little to no prior experience, students will be formally grounded in key concepts including Bayesian statistics and reinforcement learning, and practically, will build, assess, and compare models using the R interface to the Stan programming language, Rstan. Starting with binary choice models, students will learn to estimate parameters representing latent components of behaviour by fitting reinforcement learning models, both at the individual and group-level.

The course is generally suitable for those interested in developing models of human cognition at any level of experience. In making the course openly available, we aim for computational modeling under the Bayesian approach to be more strongly represented in the psychological sciences.

Statement of Need

Computational modeling is a general framework for inferring unobserved latent parameters from observed data. Whilst implemented in other disciplines (e.g., physics) for centuries, it's application specifically towards understanding the human mind (i.e., cognitive modeling) [@Farrell2018] is a relatively recent approach, one exponentially increasing in popularity [@Palminteri2017]. By formalising cognitive processes as mathematical operations, cognitive models generate specific, testable hypotheses about observable behaviour, which can be objectively compared and falsified [@Palminteri2017; @Guest2021; @Rocca2021]. Cognitive models present a key framework for understanding how the brain implements cognitive processes such as decision making and (social) learning

[@Eckstein2021; @FeldmanHall2021] and their aberration in mental health disorders [@Huys2016; @Hauser2022].

Complementing this approach is the application of Bayesian methods for parameter estimation [@Annis2017]. Historically restrictive due to their computational burden, these methods are now more accessible though the development of multiple programming languages and software such as JAGS [@Plummer2003] and Stan [@Carpenter2017] which automate the sampling process used for parameter estimation. Bayesian methods confer advantages over frequentist approaches (e.g., Maximum Likelihood Estimation, MLE), by quantifying the uncertainty, and when implemented hierarchically, permit simultaneous estimation of individual and group-level parameters while appropriately pooling information across participants [@Lee2011].

Using Bayesian models of cognition in one's own research requires a conceptual understanding of Bayesian statistics and cognitive modeling, as well as the practical skills to translate these models into code. Textbooks [@Lee2014; @Kruschke2014; @Lambert2018; @McElreath2018] and tutorial papers [@Wilson2019; @Zhang2020; @Lockwood2021; @Baribault2023] have made learning these skills more accessible, but are often not freely available, and challenging for students with little to no prior experience. Additionally, whilst free, online courses for the computational modeling of cognition currently exist, these are few and far between, and exclusively cover non-Bayesian implementations in Python [@Rhoads2022] and MATLAB [OReilly2015], as well as Bayesian approaches in Python [@NivLab2021]. A full course implementing Bayesian models of cognition through the R programming language is therefore a valuable yet currently non-existent resource.

Learning Objectives

In the course, students will:

- Build a foundational knowledge of Bayesian statistics and inference, and how it differs from frequentist definitions of probability
- Understand Bayesian parameter estimation and the conceptual basis of sampling techniques including Markov chain Monte Carlo (MCMC)
- Understand how Bayesian statistics can be applied to uncover latent parameters of cognition through cognitive modeling
- Practically build and code discrete choice and regression models using RStan
- Learn the basic reinforcement learning (RL) concepts
- Build a simple RL model (Rescorla-Wagner) using RStan
- Understand the conceptual basis of hierarchical models and implement them practically in RStan

- Be able to evaluate model performance and perform model comparison using the `loo` package in R
- Ultimately, be able to apply computational modeling in their own experiments

Prior to the first workshop, the course begins by summarizing the broader philosophy in which the techniques and methods are implemented. Specifically, this concerns Marr's influential three levels of analysis [@Marr1982], which describe how algorithmic-level models can help understand behaviour. Doing so shapes the course material within this framework, demonstrating the necessity for building strong theories in psychology [@Press2022]. Assuming no prior experience with programming, the course proper (Workshop 01) begins with a basic introduction to the R programming language and the RStudio interface [@RStudio2020]. This introductory workshop first provides a general introduction to data structures, variables, and packages, after which students will work with simulated data from a reversal learning task, performing basic summary statistics including correlation and regression, and visualizing the results using the `ggplot2` package [@Wickham2016].

In Workshop 02, students will be introduced to Bayesian statistics, learning the differences between Bayesian and frequentist definitions of probability. This can be conceptually difficult for those with little statistical or mathematical experience, the material therefore slowly builds to the concept of Bayes' theorem from simpler probability concepts including joint, conditional and marginal probability, and probability types – discrete and continuous. These concepts are put into practical use in Workshop 03, where the Bayesian approach is transformed from a purely mathematical concept to a system where one can determine the values of unknown parameters from data. The goal of Bayesian inference - computing the probability distribution of model parameters given the observed data – is also introduced, firstly using a simple instance, grid approximation, adapted from [@McElreath2018]. Students will subsequently learn that for more complex models, sampling procedures which approximate the posterior distribution are used. To this end, a popular approach – Markov chain Monte Carlo (MCMC) – is conceptually described.

At this point, students will have the conceptual knowledge of Bayesian statistics and probability to practically implement models. Workshop 04 introduces students to the Stan programming language [@Carpenter2017] and it's R interface RStan [@StanTeam2024]. Following an overview of Stan syntax, students will construct a binomial model estimating the same parameter calculated by grid approximation in the previous workshop. In doing so, students will become familiar with the relationship between Stan and R, with models being coded in a .stan file, and sourced to the respective R script. Students will also examine basic outputs of the sampler, including the summary statistics, and by plotting the posterior density distribution. Workshop 5 builds upon this introductory workshop by introducing some specific properties of Stan, including vectorization and variable declaration, and introduces two more models: the Bernoulli model and linear regression.

In Workshop 6, we turn to the specific application of these methods to infer latent cognitive processes. A basic overview to the principles of cognitive modeling is followed by an introduction to reinforcement learning, a popular theory of human behaviour. To this end, a simple reinforcement learning algorithm consisting of the Rescorla-Wagner model [@Rescorla1972], and softmax choice rule is introduced. Students will then practically implement this 'Simple RL' model in Stan, for simulated choice data for a single subject, before fitting multiple subjects. For the latter exercise, students will do so in two ways, by firstly looping across subjects and fitting each subject individually. In doing so, they will learn about statistical concerns with parameter estimation in groups. Workshop 7 directly builds upon this topic by introducing hierarchical Bayesian models [@Lee2012] for parameter estimation. Introducing the concepts of 'hyperparameters' and 'hyperpriors', students will then run a hierarchical implementation of the Simple RL model. Given that Bayesian cognitive models often require troubleshooting for parameter estimation [Baribault2023] optimization strategies are also introduced in this workshop. Specifically, this covers Stan's sampling parameters and reparameterization; the latter being particularly relevant for hierarchical models.

Model comparison is introduced in Workshop 8, with the basis behind model fitting, predictive accuracy and information criterion firstly described. Taking Widely Applicable Information Criterion (WAIC) as an example, students will subsequently compare two RL models: the Simple RL and 'Fictitious RL' using the `loo` package [@Behtari2024] in R, and plot posterior predictive checks as a measure of model validation. The final workshop (Workshop 9) describes key strategies for code debugging in Stan, using a purposely error-laden delay-discounting model [@Lee2014) for students to interactively troubleshoot problematic code. To conclude, a published study [@Crawley2020] where computational models were implemented to uncover learning differences is described, providing real-case examples of model and parameter recovery.

The course begins from the shared assumption of no-to-little prior experience among students with programming and statistics, however specific workshops may be skipped for certain audiences. For example, if students are comfortable with the RStudio environment, teachers could begin with Workshop 2. Similarly, if looking to introduce experienced Stan users to the specific topic of modeling cognitive processes through RL, Workshop 6 provides an ideal starting point. Any tutorial can be adopted, transformed, or built upon for other educational purposes (e.g., courses, single class sessions, workshops) under a Creative Commons Attribution-ShareAlike 4.0 International License.

All course workshops are accompanied by example data and scripts, the latter provided in both uncompleted and completed versions where appropriate. All software required for the course are open-source and easy to install; instructions are provided in the 'Course overview' page. Whilst specific R packages (`rstan`, `loo`, `ggplot2`) are required, the course uses `renv` [Ushey2025] to simplify the

Figure 1.

Figure 1: Figure 1.

installation process. Users simply run a single command which installs the required packages for all successive R sessions. However, `renv` only provides a simplified solution to reproducibility and can be mired by system dependencies and versions of RStudio. Therefore, users can alternatively, pull a pre-made Docker image, building a container locally to host the RStudio environment. This does not necessitate users to have R or RStudio installed, maintaining a more consistent and reproducible development environment across different systems and platforms. In either case, generating the working environment requires minimal effort (Fig 1.)

Course Overview

Folder	Task	Model
01.R_basics	NA	NA
02.binomial_globe	Globe toss	Binomial Model
03.bernoulli_coin	Coin flip	Bernoulli Model
04.regression_height	Observed weight and height	Linear regression model
05.regression_height_poly	Observed weight and height	Polynomial regression model
06.reinforcement_learning	2-armed bandit task	Simple reinforcement learning (RL)
07.optm_rl	2-armed bandit task	Simple reinforcement learning (RL)
08.compare_models	Probabilistic reversal learning task	Simple and Fictitious RL models
09.debugging	Memory Retention	Exponential decay model
10.model_based	NA	NA

Future Directions

The BayesCog course provides a general introduction to Bayesian statistics and computational modeling within the context of psychological research. Subsequently, the materials and topics could be developed further. This includes expanding the family of models beyond reinforcement learning, introducing the application of computational methods with neuroimaging data [@Gläscher2010;

@deHollander2016] and discussing theory-based modeling of psychiatric disorders [@Maia2017]. Furthermore, the Stan programming language remains technically challenging, leading to the development of user-friendly packages for computational modeling [@Ahn2017]. Tutorials on how to implement these packages could further broaden the use of computational methods within the psychological sciences.

To this end, we openly receive feedback and suggestions from the wider community. Broader comments can be communicated on the GitHub repository by either reporting an issue or requesting an enhancement. On-the-other-hand, we recommend major changes to be made communicated beforehand and – if appropriate - made directly by forking the repository and pushing changes to the main branch. A member of the contributing team will then review the changes. Accepted contributions will be credited and following the community guidelines outlined on the CONTIBUTORS page.

Acknowledgments

L.Z. is grateful to . . .

A.S. is grateful to Lei Zhang for developing the initial version of these materials, including slides, code, and GitHub repositories. He would also like to thank the creators of existing open-source materials, including Shawn Rhoads' Computational Models of Human Social Behavior and Neuroscience [@Rhoads2022], Luke Chang's DartBrains: An online open access resource for learning functional neuroimaging analysis methods in Python [@Chang2020] and Magdalena Chechlacz's MRI on BEAR.

Author Contributions

L.Z. created, designed and taught the original course materials by creating the slides, writing the Stan and R code, creating and interpreting the datasets and recording the YouTube videos.

L.Z. developed the course using prior experience . . .

L.Z. was supported by . . .

A.S. created the website, added the content by converting, editing and expanding the source material. A.S. adapted the material using prior experience as a Teaching Assistant, adapting course materials for Magdalena Chechlacz's Magnetic Resonance Imaging in Cognitive Neuroscience (MRICN) module at the University of Birmingham. Both authors revised the course materials and wrote the manuscript. The course materials were created to make the content more

generally accessible and engaging, both generally and for students taught by L.Z. and A.S.

A.S. is supported by an MRC AIM iCASE DTP Studentship (Ref: MR/W007002/1).