Bayesian Workflow for Hierarchical and ODE-based Models using Stan

Charles C. Margossian

September 2023

mc-stan.org

Summer School on Advanced Bayesian Methods
Lueven, Belgium

Instructor. Charles Margossian
cmargossian@flatironinstitute.org
https://charlesm93.github.io./

Teaching Assistant. Maxime Fajgenblat
maxime.fajgenblat@kuleuven.be

Overview. In this course, we will discuss the tenants of the Bayesian workflow and how to execute them using state-of-the-art software. The Bayesian workflow is the iterative process through which we build, fit, and criticize models, with the latter step often motivating useful revisions to our model. We will take advantage of `Stan`, a Bayesian inference software which boasts a flexible language to specify models, and supports scalable inference algorithms, notably an adaptive Hamiltonian Monte Carlo (HMC) sampler — currently one of the most successful Markov chain Monte Carlo (MCMC) methods. This will be a hands-on workshop: students will be expected to code and attempt several exercises. Throughout, ongoing research as well as open questions on the subject of Bayesian modeling will also be highlighted.

Goals.

- Learn the `Stan` language.
- Develop a (deeper than usual) understanding of Bayesian inference using MCMC.
- Learn the fundamentals of the Bayesian workflow.
- Apply these principles to hierarchical models and ODE-based models.
- The third day focuses on applications in pharmacometrics using the add-on `Torsten`.

Helpful Prerequisites.

- Familiarity with Bayesian modeling and probability distributions.
- Familiarity with a coding language, e.g `R` or `Python`. We will use R scripts for the exercises, but the focus will be on coding in `Stan`. Students are not expected to know `Stan`.

Installation. For this course, we will use `R` as a scripting language. The code in `R` is already written and we will write `Stan` code together. The `R` code will be available at

`https://github.com/charlesm93/stanTutorial`

From the GitHub page you can open notebooks and open them in Colab. If you do so, you will be able to run the R script on Colab's cloud server. The first cells of the notebook install all the requisite R packages and Stan (and when relevant Torsten).

You also have the option to run R on your local machine, in which case you must install the following packages:

- cmstanr, ggplot2, rjson, posterior, bayesplot, loo

If you're not able to install the packages before the tutorial, that's ok! You still have the option to use Colab.

- For instructions on installing R, see https://www.r-project.org/. I also recommend downloading RStudio (https://posit.co/products/open-source/rstudio/) — the free version is all I've ever used.
- cmdstanr is a light-weight wrapper around Stan and the most up-to-date R interface for Stan.[1] For instructions on how to install cmdstanr, see https://mc-stan.org/cmdstanr/. In our experience, installing cmdstanr is straightforward with Mac and Linux. For windows machine, you need to make sure you have a suitable C++ toolchain.
- Most other R packages are available on CRAN and can be installed directly from R using the install function, e.g. `install.packages("ggplot2")`. Some of them require special (though still straightforward) installation.
- Instructions for installing the version of Stan with the Torsten add-on can be found at https://metrumresearchgroup.github.io/Torsten/installation/.

[1] A similar package exists in Python called cmdstanPy.

Outline

1. Review of Bayesian analysis
    - Bayesian model
    - Bayesian inference
    - Beyond inference: the Bayesian workflow
2. Markov chain Monte Carlo
    - Probing the posterior distribution
    - Monte Carlo estimators
    - Markov chain and stationary distribution
    - Examples
        - Metropolis algorithm
        - Langevin diffusion
    - Warmup and sampling phases
3. Basics of Stan
    - Language blocks in Stan
    - Example: Bayesian linear regression
    - Checking the inference
        - Trace and density plots
        - $\widehat{R}$ and nonstationary variance
        - ESS for the bulk and tail of the posterior
    - Criticizing the model with posterior predictive checks
    - Exercise: improve the Bayesian linear regression model
4. ODE-based models
    - Susceptible-Infected-Recovered (SIR) model
    - Measurement model for count data
    - Solving an ODE in Stan
    - Exercise: build, fit, and criticize the SIR model(s).
    - Discussion: which choice of likelihood should we use?
5. Model comparison
    - Predictive scores on validation set
    - Leave-one-out cross validation
    - Pareto-smoothed importance sampling (PSIS)

- Exercise: implement the PSIS estimator to compare the two candidate SIR models.

6. Hamiltonian Monte Carlo
    - Geometric structure in the posterior and limitation of random walk
    - The mechanics of HMC
    - Learning the tuning parameters of HMC
    - Automatic differentiation
    - Relative computational cost of each block in Stan
7. Tuning ODE solvers in a Bayesian context
    - Control parameters for ODE solvers
    - Exercise: fit the the SIR model with varying choices of tolerances.
    - Importance sampling approach to check the tuning of ODE solvers
    - Exercise: Implement diagnostics for ODE solvers in the SIR model
    - ODE solvers supported by Stan
    - Open questions regarding solving ODEs in a Bayesian context
8. Hierarchical models
    - Regularization and partial pooling
    - Exercise: fit the 8 schools model (first attempt).
    - Divergent transitions and where they come from
    - Exercise: fit the 8 schools model, using the following strategies to handle the pathological geometry of the posterior:
        - (i) increase precision of the Hamiltonian simulations
        - (ii) use a non-centered parameterization
        - (iii) marginalize out the latent variables
    - Exercise: build and fit the disease map of Finland.
    - Strategies to handle hierarchical models: integrated Laplace approximation, Riemannian HMC, and delayed rejection HMC.
9. Torsten
    - Example: two compartment pharmacokinetic model
    - Clinical event schedule and basics of Torsten
    - Exercise: build, fit and criticize a two compartment model, and find the least strict tolerance for the ODE solver that still returns accurate inference.
    - ODE solvers supported by Torsten: analytical, semi-analytical, numerical and mixed.

- Exercise: fit the one and two compartment models using analytical solutions. Compare the two models.

10. Population models
    - Exercise: build, fit, and criticize a population pharmacokinetic model
    - Within-chain parallelization
    - Exercise: implement the population pharmacokinetic model using within-chain parallelization.
11. Concluding remarks