Bridging Statistics and Dynamic Modeling with Vensim, Python, and Stan

Abstract
Stanify is a new library for translating Vensim models into probabilistic programs defined with the Stan language. It is composed of a source-to-source code translator for model conversion and a miniature modeling language for specifying Bayesian models on top of the Vensim model. Stanify brings together the powerful, intuitive interface for designing dynamic models of Vensim and the robust inference performance of Stan. Stanify allows users to specify priors for Vensim variables, declare observational models on top of their existing Vensim model without having to write any Stan code. Stan provides access to different algorithms, including Hamiltonian Monte Carlo, and diverse diagnostics.

The workflow includes two novelties: symmetry between generator and estimator can be leveraged to graphically validate procedures from end to end, including the model, priors, and methods; samples from the estimation can be reprojected to Vensim sensitivity simulations to support decision making under uncertainty.

We introduce these methods with an example, exploring variants of the classic Lotka-Volterra predator-prey model, using Bayesian hierarchy and simulation-based calibration (SBC).

BayesSD, Verification, Validation

Workflow and Tools

Results
From predator-prey model, birth fractional adjustment for prey and predator (in red) were estimated 500 times and compared with 500 ground truth parameter values sampled from prior distribution.

Fig.2 Testing workflow. For detailed guidelines for the workflow, especially on test statistics choice (step 4), refer to simulation-based calibration paper by Modrák et al. (2023).

Fig.3 Architecture of Vensim + Stan on Python. Stanify exploits open-source tools, including PySD, to translate the Vensim model into a Stan-readable format. Stan output is then directed into an ArviZ inferenceData object for visualization.

Next Steps
1. Answering what benefit Bayesian inference can provide over the conventional inferential methods for ODE models
   - in line with sequential and structured decision making, easier to test
   - uncertainty modeling and control: possible to model uncertainty in any input (scalar: fractional adjustment, process and measurement noise variance, vector: driving data, function: ode equation)

2. Answering implementation difficulties of Bayesian inference for typical ODE models in SD domain and how to overcome?
   - hierarchical Bayesian epidemic and economy + Dream MCMC (Haschik Rahmandad, Evan Feldman, Angie Moon)
   - hierarchical Bayesian good job strategy + Dream MCMC (Hazhir Rahmandad, Jason Friedman)
   - hierarchical Bayesian chronic waste disease + Dream MCMC (Tom Fiddaman)
   - epidemic model + HMC (Lin Amin, Damir Duggan)
   - planetary movement + HMC (Andrew Sansone et al)
   - disease transmission + HMC (Léo Grinsztajn et al)
   - delegated improvement efforts on inter-level trust and future success + (tbd) (Cathy Samsó, Huy Vu)

3. Collecting case studies introducing Bayesian analysis for ODE models
   - hierarchical Bayesian epidemic and economy + Dream MCMC
   - hierarchical Bayesian good job strategy + Dream MCMC
   - hierarchical Bayesian chronic waste disease + Dream MCMC
   - epidemic model + HMC
   - planetary movement + HMC
   - disease transmission + HMC
   - delegated improvement efforts on inter-level trust and future success + (tbd)

4. Identifying common failure modes in Fig.2

Selected References

Tom Fiddaman, CTO, Ventana systems
tom@ventanasystems.com
Angie H. Moon, Ph.D. student, MIT
amoon@mit.edu