METHODOLOGY

The software in the paper consists of three parts:

1. A library that solves for the first- and second-order symbolic derivatives to the model equations.
2. A library that provides differentiable simulations and likelihoods, given the state-space models.
3. A Bayesian estimator for model parameters.

In translating the program from Julia to Python, I propose replacing the first and second portions with model-specific Tensorflow auto-differentiators, for two reasons. First, automatic differentiation is ubiquitous in machine learning, and is naturally well-developed in Tensorflow. Second, the goal of the exercise is only to recreate the results in the paper (which has only three models), while the goal of the paper was more general and developed a method to work with models of arbitrary complexity.

In terms of implementation, the paper lists its own setup:

> *"All timed numerical experiments in this section are conducted on an AWS t3.xlarge instance with four vCPUs and 16 GiB memory. Untimed experiments and the empirical application are run on an m5.8xlarge instance with 32 vCPUs and 128 GiB memory"*

I am prepared to use a similar AWS setup for speed if practically necessary, but some initial testing must be done on a smaller system.