Torsten: Stan functions for pharmacometric applications
New functionality including within chain parallel computation
Yi Zhang, William R. Gillespie
Metrum Research Group

Introduction
Stan is a widely used, open-source, probabilistic programming language and Bayesian inference engine [3, 4]. It provides a general model specification language and uses HMC simulation for fully Bayesian data analysis. Torsten is a library of Stan functions that simplifies implementation of pharmacometric (PMX) models and extends the range of models that may be implemented [2]. The objective of the work presented here is to improve and extend Torsten. This includes the addition of new functions and enhancements to existing ones.
Currently Torsten provides following functions for single-subject PMX modeling.

```
// Functions for single-subject pharmacometric analysis

matrix pmx_integrate_ode_rk45(...) { ...

... 

pmx_integrate_ode_bdf(...) { ...

... 

pmx_integrate_ode_adams(...) { ...

... 

pmx_solve_onecpt(...) { ...

... 

pmx_solve_onecpt_bdf(...) { ...

... 

pmx_solve_rk45(...) { ...

... 

pmx_solve_bdf(...) { ...

... 

pmx_solve_adams(...) { ...

... 

pmx_solve_rk45(...) { ...

... 

pmx_solve_bdf(...) { ...

... 

pmx_solve_adams(...) { ...

... 
```

Torsten's ODE integrators
To facilitate design for within-chain parallelization, Torsten v0.86 added its own implementation of ODE integrators based on the same backend libraries (odeint & CVODES [5]) as Stan's, and they share the same signature. Additionally, Torsten's ODE integrators allow solution times ts to be parameters. Table 1-3 show performance comparison of the integrators based on two ODE systems (Torsten v0.86, Stan v2.19.1).

```
integrator run 1 run 2 run 3
integrate_ode_bdf 1.41 1.42 1.43
```

Table 1: wall time(s) of solving a chemical kinetics ODE using BDF integrators

```
integrator run 1 run 2 run 3
integrate_ode_adams 2.09 2.11 2.07
```

Table 2: wall time(s) of solving a chemical kinetics ODE using Adams integrators

```
integrator run 1 run 2 run 3
integrate_ode_adams 8.12 8.75 8.22
```

Table 3: wall time(s) of solving Lorenz system ODE using Adams integrators

Torsten's within-chain parallel integrators and solvers
Torsten now includes versions of Stan's ordinary differential equation (ODE) solvers that have been revised to improve performance and provide MPI-based parallel computing. We have implemented new population functions that calculate the model states for a group of individuals in a single function call. Those functions also employ MPI to distribute those calculations over multiple processors, thus providing efficient within chain parallel computation.
Torsten provides an alternative to Stan's map_rect framework that avoids use of boost mpi and serialization, simplifies user programming tasks by providing a simplified function signature and automating computation load distribution. Unlike map_rect, Torsten's parallel computing capability is available through functions for specific heavy-lifting tasks. Currently we focus on tasks involving numerical solution of ODEs.

ODE group integrators
The ODE group integrators solve a set of systems governed by a single ODE but with different parameters/data. Function arguments y0, theta, x_r, and x_i are 2D arrays with 1st dimension indicating subject within the group. ts is a ragged array for each subject's solution time, and len consists of length of each subject's record in ts.

```
matrix pmx_integrate_ode_group_adams(...) { ...

... 

pmx_integrate_ode_group_rk45(...) { ...

... 
```

We demonstrate the MPI performance of the integrators using two tests.

• Test 1: Neutropenia PKPD ODE group of size 1000 solved using BDF group integrator. Each ODE is of size 8 and has 9 parameters.

• Test 2: predator-prey model parameter inference for a set of 16 Lotka-Volterra equations, with the ODE group solved using BDF4 integrator.

```
```
```
```

Pharmacometric population solvers
Similar to single-subject solvers, the group/population PMX solvers use a model specification and data format based on NONMEM®/NMTRAN/PREDPP conventions. Function arguments specify dosing events for a population using ragged arrays, with each subject's data located through array len that contains the subject record length.
We demonstrate the MPI performance of the population PMX solvers using two tests.

• Test 3: Two-compartment model with first-order absorption among a population of 8. Stan model parameters are the PK model parameters for each subject. The population PK model is solved numerically using pmx_solve_group_bdf.

• Test 4: Parametric time-to-event model for the time to the first grade 2+ peripheral neuropathy event in 60 patients treated with an antibody-drug conjugate delivering monomethyl auristatin E. The hazard ODE system is of size 3 and has 5 parameters.

Interface to external PDE libraries
We have also implemented interface to external PDE libraries for applications that involve PDE parameter inference. The methodology has been tested using multiple PDE libraries. Example on the right shows Darcy's flow velocity in an incompressible model for porosity I of in irregular flow region, using Torsten. The PDE interface feature will be available in a future Torsten release.

Conclusions and future work
Recent developments in Stan and Torsten significantly improve computational efficiency and extend the range of models that may be implemented. The addition of within chain parallel computation to Stan/Torsten makes fully Bayesian analysis with Stan an increasingly practical option for PMX applications. Some near-future work planned for Torsten:

• Further improve MPI performance.
• Multi-level and hybrid parallel computing based on MPI and multithreading.

References
[1] M2R: Modular Data-driven methods library. m2r.org.