Population and ODE-based models using Stan and Torsten

Charles Margossian, Yi Zhang

StanCon 2019, Cambridge UK
August 2019

Outline

Course information

Introduction and modeling framework
Charles Margossian

Numerical ODE integrators
Yi Zhang

ODE group integrators
Yi Zhang

PMX population solvers
Yi Zhang

Outline

Course information

Introduction and modeling framework

Charles Margossian

Numerical ODE integrators

Yi Zhang

ODE group integrators

Yi Zhang

PMX population solvers

Yi Zhang

Instructors

- Charles Margossian
  - Columbia University, Department of Statistics
- Yi Zhang
  - Metrum Research Group

TA

- Steve Bronder
  - Capital One

Outline

Day 1

- Introduction and modeling framework
- Pharmacometrics models
- Ordinary differential equation(ODE) based models
- Numerical ODE integrators

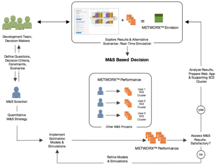
Day 2

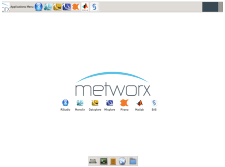
- Population models
- Group/Population ODE integrators and MPI parallelisation
- Group/Population solvers and MPI parallelisation

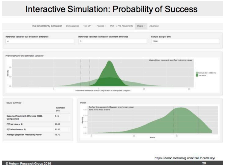
Logistics

METWORX™, cloud-based modeling & simulation platform by Metrum Research Group.

metworx






Interactive Simulation: Probability of Success

Trial Uncertainty Simulator
Demographics
Test CP ▾
Placebo ▾
Ph2 --> Ph3 Adjustments
Output ▾
Advanced

Reference value for true treatment difference: 4

Reference value for estimate of treatment difference: 3

Sample size per arm: 1000

Prior Uncertainty and Estimation Variability

Dashed lines represent specified reference values
density
Treatment difference (U300-Comparator) in Composite Endpoint
Estimate (N = 1000/arm)
True Value

Tabular Summary

	Estimate (%)
Expected Treatment difference (U300-Comparator)	6.15
P(True value > 4)	99.80
P(Trial estimate > 3)	91.30
Average (Bayesian Predictive) Power	78.70

Power

Dashed line represents (Bayesian prior) mean power
Solid line is fixed at 90%
density
Power

https://demo.metrumrg.com/trialUncertainty/

© Metrum Research Group 2016 20

Logistics

Workshop package

- R scripts and Stan files to do the exercises
- These slides
- Outline of the course
- pAdditional documentation

We will be using:

- Torsten v0.87
- RStan v2.19.2
- ggplot, plyr, tidyr, dplyr

Outline

Course information

Introduction and modeling framework
Charles Margossian

Numerical ODE integrators
Yi Zhang

ODE group integrators
Yi Zhang

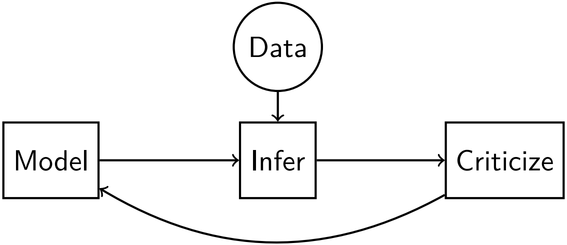
PMX population solvers
Yi Zhang

Preliminary question

- Why Bayesian in a field such as pharmacometrics?
- Example - *Bayesian aggregation of average data: an application in drug development* [**?**]

Modeling framework

Box's loop



Inference

- find the set of parameters consistent with our model and our data
- approximate this set with draws from the posterior distribution

Sampling algorithm

- Use the NUTS to sample $\pi(\theta|y)$
- Requires users the specify $\log \pi(\theta, y) = \log \pi(y|\theta) + \log \pi(\theta)$

The "criticism" step

This step can be broken up in two parts:

1. did we sample from the correct distribution?
2. does our model capture the characteristics of the data we care about?

Diagnosing the inference algorithm

- look at the trace and the density plots
- look at $\hat{R}$ and effective number of samples
- have any warning messages been issued, i.e. divergent transitions ?

Example: fitting a linear model

Likelihood:

$$Y \sim \mathrm{Normal}(x\beta, \sigma^2)$$

Prior:

$$\begin{aligned} \beta \sim& \mathrm{Normal}(2, 1) \\ \sigma^2 \sim& \mathrm{Normal}(1, 1) \end{aligned}$$

Outline

Course information

Introduction and modeling framework
Charles Margossian

Numerical ODE integrators
Yi Zhang

ODE group integrators
Yi Zhang

PMX population solvers
Yi Zhang

Nonlinear ODEs without analytical solution

kinetics of an autocatalytic reaction [?]

The structure of the reactions is

$$A \xrightarrow{k_1} B, \quad B + B \xrightarrow{k_2} C + B, \quad B + C \xrightarrow{k_3} C + A,$$

where k_1, k_2, k_3 are the rate constants and A, B and C are the chemical species involved. The corresponding ODEs are

$$\begin{aligned} x_1' &= -k_1 x_1 + k_3 x_2 x_3 \\ x_2' &= k_1 x_1 - k_2 x_2^2 - k_3 x_2 x_3 \\ x_3' &= k_2 x_2^2 \end{aligned}$$

Given $k_1 = 0.04, k_2 = 3.0e7, k_3 = 1.0e4$, we make inference regarding the initial condition for $x_1(t = 0)$.

Exercise

Write Stan function for the above ODE's RHS.

Stan function for autocatalytic kinetics

$$
\begin{aligned}
x_1' &= -k_1 x_1 + k_3 x_2 x_3 \\
x_2' &= k_1 x_1 - k_2 x_2^2 - k_3 x_2 x_3 \\
x_3' &= k_2 x_2^2
\end{aligned}
$$

```
functions{
  real[] reaction(real t, real[] x, real[] p, real[] r, int[]
i){
    real dxdt[3];
    real k1 = p[1];
    real k2 = p[2];
    real k3 = p[3];
    dxdt[1] = -k1*x[1] + k3*x[2]*x[3];
    dxdt[2] =  k1*x[1] - k3*x[2]*x[3] - k2*(x[2])^2;
    dxdt[3] =  k2*(x[2])^2;
    return dxdt;
  }
}
```

- What's the initial conditions for x_2 and x_3?

Numerical integrators

- Runge-Kutta 4th/5th (`rk45`)
  - non-stiff equations
  - Most popular, try this if you don't know the nature of the ODE, or what you're doing, or both.
- Backward differentiation formula (`bdf`)
  - stiff equations
  - More expensive to use
- Adams-Moulton
  - non-stiff equations
  - higher-order of accuracy(do you really need it?)
  - scales better with number of steps

Numerical integrators

Integrators	Stan	Torsten
rk45	integrate_ode_rk45	pmx_integrate_ode_rk45
BDF	integrate_ode_bdf	pmx_integrate_ode_bdf
Adams	integrate_ode_adams	pmx_integrate_ode_adams

```
real[ , ] pmx_integrate_ode_rk45(ODE_RHS, real[] y0, real t0,
↪   real[] ts, real[] theta, real[] x_r, int[] x_i, real rtol =
↪   1.e-6, real atol = 1.e-6, int max_step = 1e6);
```

- ODE_RHS: ODE right-hand-side f in $y' = f(y, t, \theta, x_r, x_i)$.
- y0: initial condition at time t0.
- t0: initial time.
- ts: times at which we require a solution.
- theta: parameters to be passed to f.
- x_r: real data to be passed to f.
- x_i: integer data to be passed to f.

Exercise

- In each of 8 experiments performed $x3$ is observed.
- Hierarchical model for $x0[1]$

```
model {
  x0_mu ~ lognormal(log(2.0), 0.5);     /* hyperparam */
  for (i in 1:nsub) {
    x0_1[i] ~ lognormal(x0_mu, 0.5);    /* loop each subject,
     ↪   sample initial condition */
  }
  sigma ~ cauchy(0, 1);                 /* observation
   ↪   uncertainty */
  obs ~ lognormal(log(x3), sigma);
}
```

Exercise

Data available for the inference

```
data {
  int<lower=1> nsub;        /* nb. of subjects */
  int<lower=1> len[nsub];   /* nb. of results-extraction time
   ↪  points for each subject */
  int<lower=1> ntot;        /* total nb. of results-extraction
   ↪  time points */
  real ts[ntot];            /* concatenated array for
   ↪  results-extraction time points */
  real obs[ntot];           /* concatenated array for observed
   ↪  x3 */
}
```

Given above data and model, write the rest of Stan code.

- Hint: see `chem.data.R` and `chem.init.R` .
- Which numerical integrator are you using? Why?

Exercise

How to build & run?

Edit/Add cmdstan/make/local

```
TORSTEN_MPI = 1  # flag on torsten's MPI solvers
CXXFLAGS += -isystem /usr/local/include    # path to MPI
 ↪   library's headers
```

Build in cmdstan

```
make ../example-models/examples/chemical_reactions/chem
```

Run

```
./chem sample adapt delta=0.95 random seed=1104508041 data
 ↪   file=chem.data.R init=chem.init.R
```

Outline

Course information

Introduction and modeling framework
Charles Margossian

Numerical ODE integrators
Yi Zhang

ODE group integrators
Yi Zhang

PMX population solvers
Yi Zhang

ODE group integrators

Single ODE system	ODE group
pmx_integrate_ode_rk45	pmx_integrate_ode_group_rk45
pmx_integrate_ode_bdf	pmx_integrate_ode_group_bdf
pmx_integrate_ode_adams	pmx_integrate_ode_group_adams

Single ODE system

```
real[,]
pmx_integrate_ode_xxx(
      f,
      real[] y0, real t0,
      real[] ts,
      real[] theta,
      real[] x_r, int[] x_i,
      ...);
```

ODE group

```
matrix
pmx_integrate_ode_group_xxx(
     f,
     real[ , ] y0, real t0,
     int[] len, real[] ts,
     real[ , ] theta,
     real[ , ] x_r, int[ , ] x_i,
     ...);
```

ODE group integrators

Single ODE system

```
real[ , ]
pmx_integrate_ode_xxx(
      f,
      real[] y0, real t0,
      real[] ts,
      real[] theta,
      real[] x_r, int[] x_i,
      ...);
```

ODE group

```
matrix
pmx_integrate_ode_group_xxx(
     f,
     real[ , ] y0, real t0,
     int[] len, real[] ts,
     real[ , ] theta,
     real[ , ] x_r, int[ , ] x_i,
     ...);
```

- `len` specifies the length of data for each subject within the above ragged arrays, and the size of `len` is the size of the population.
- The group integrators return a single matrix ragged column-wise. The number of rows equals to the size of ODE system.

Exercise

autocatalytic reaction model: ODE group version

- Change the loop with the numerical integrator to use group integrator.
- Edit/Add `cmdstan/make/local`

```
TORSTEN_MPI = 1  # flag on torsten's MPI solvers
CXXFLAGS += -isystem /usr/local/include    # path to MPI
↪  library's headers
```

- Build in `cmdstan`

```
make ../example-models/chemical_reactions/chem_group
```

- Run

```
mpiexec -n 2 -l ./chem_group sample adapt delta=0.95 random
↪  seed=1104508041 data file=chem.data.R init=chem.init.R
```

Exercise

- What does output say?
- How many cores can you use until performance saturates? Why?
- Can you do it using Stan's `map_rect`? Is there a difference in style, output, and performance?

Outline

Course information

Introduction and modeling framework
Charles Margossian

Numerical ODE integrators
Yi Zhang

ODE group integrators
Yi Zhang

PMX population solvers
Yi Zhang

PMX population solvers

Single ODE system	ODE group
pmx_solve_rk45	pmx_solve_group_rk45
pmx_solve_bdf	pmx_solve_group_bdf
pmx_solve_adams	pmx_solve_group_adams

Individual solvers

```
matrix
pmx_solve_bdf(f, int nCmt,
  real[] time, real[] amt,
  real[] rate, real[] ii,
  int[] evid, int[] cmt,
  real[] addl, int[] ss,
  real[] theta, real[]
   ↪   biovar,
  real[] tlag, real rel_tol,
  real abs_tol, int
   ↪   max_step);
```

Population solvers

```
matrix
pmx_solve_group_bdf(f, int nCmt,
  int[] len, real[] time,
  real[] amt, real[] rate,
  real[] ii, int[] evid,
  int[] cmt, real[] addl,
  int[] ss, real[ , ] theta,
  real[ , ] biovar, real[ , ] tlag,
  real rel_tol, real abs_tol,
  int max_step);
```

PMX population solvers

```
matrix
pmx_solve_group_bdf(f, int nCmt, int[] len, real[] time, real[]
↪   amt, real[] rate, real[] ii, int[] evid, int[] cmt, real[]
↪   addl, int[] ss, real[,] theta, real[,] biovar, real[,]
↪   tlag, real rel_tol, real abs_tol, int max_step);
```

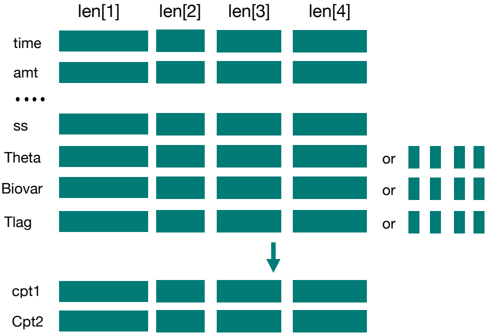


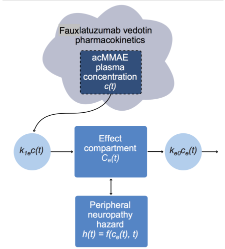
Figure: arguments and output of `pmx_solve_group_xxx`

Exercise

We analyze the time to the first grade 2+ peripheral neuropathy (PN) event in patients treated with an antibody-drug conjugate (ADC) delivering monomethyl auristatin E (MMAE). We will simulate and analyze data using a simplified version of the model reported in [**?**].

- Fauxlatuzumab vedotin 1.2 mg/kg IV boluses q3w × 6 does.
- 19 patients with 6 right-censored (simulated data).

Model scheme


Fauxlatuzumab vedotin pharmacokinetics
acMMAE plasma concentration $c(t)$
$k_{1e}c(t)$
Effect compartment $C_e(t)$
$k_{e0}c_e(t)$
Peripheral neuropathy hazard $h(t) = f(c_e(t), t)$

Note

- To keep things simpler, we use the simulated individual CL and V values, and only model PD part of the problem.
- PN hazard is substantially delayed relative to PK exposure.
- Hazard increases over time to an extent not completely described by PK.

Exercise

Likelihood for time to first PN ≥ 2 event in the i^{th} patient:

$$
\begin{aligned}
&L\left(\theta|t_{\mathrm{PN},i}, \mathrm{censor}_i, X_i\right) \\
&= \begin{cases} h_i\left(t_{\mathrm{PN},i}|\theta, X_i\right) e^{-\int_0^{t_{\mathrm{PN},i}} h_i(u|\theta,X_i)du}, & \mathrm{censor}_i = 0 \\ e^{-\int_0^{t_{\mathrm{PN},i}} h_i(u|\theta,X_i)du}, & \mathrm{censor}_i = 1 \end{cases}
\end{aligned}
$$

where

$$
\begin{aligned}
t_{\mathrm{PN}} &\equiv \text{time to first PN} \geq 2 \text{ or right censoring event} \\
\theta &\equiv \text{model parameters} \\
X &\equiv \text{independent variables / covariates} \\
\text{censor} &\equiv \begin{cases} 1, & \text{PN} \geq 2 \text{ event is right censored} \\ 0, & \text{PN} \geq 2 \text{ event is observed} \end{cases}
\end{aligned}
$$

One can see the expression

$$
e^{-\int_0^{t_{\mathrm{PN},i}} h_i(u|\theta,X_i)du}
$$

as the survival function at time t.

Exercise

Hazard of PN grade 2+ based on the Weibull distribution, with drug effect proportional to effect site concentration of MMAE:

$$\begin{aligned} h_j(t) &= \beta E_{\text{drug}j}(t)^{\beta} t^{(\beta-1)} \\ E_{\text{drug}j}(t) &= \alpha c_{ej}(t) \\ c'_{ej}(t) &= k_{e0}\left(c_j(t) - c_{ej}(t)\right). \end{aligned}$$

Overall ODE system including integration of the hazard function:

$$x'_1 = -\frac{CL}{V} x_1 \tag{1}$$

$$x'_2 = k_{e0}\left(\frac{x_1}{V} - x_2\right) \tag{2}$$

$$x'_3 = h(t) \tag{3}$$

where $x_2(t) = c_e(t)$ and $x_3(t) = \int_0^t h(u)du$ aka cumulative hazard.

Exercise

"just walk in a minute ago, literally" mode

Apply `pmx_solve_group_rk45` function

Intermediate mode

Code `pmx_solve_group_rk45` function and its args. Use input data file `ttp2n.data2.R` as hint.

hard mode

Code ODE, `pmx_solve_group_rk45` function and its args, and the likelihood for harzard and censor event. Use input data file `ttp2n.data2.R` and `model` block as hint.

"why bother" mode

Exercise

Edit/Add cmdstan/make/local

```
TORSTEN_MPI = 1          # flag on torsten's MPI solvers
CXXFLAGS += -isystem /usr/local/include # path to MPI library's
 ↪   headers
```

Build in cmdstan

```
make ../example-models/ttpn2/ttpn2_group
```

Run

```
mpiexec -n 4 -l ttpn2_group sample num_warmup=500
 ↪   num_samples=500 data file=ttpn2.data2.R init=ttpn2.init.R
```

Exercise

- The parallel performance is not optimal, why?
- Can you do it using Stan's `map_rect`?

Reference

Reference