METRUM
RESEARCH GROUP

A Brief Introduction to Bayesian Modeling Using Stan

Bill Gillespie
Metrum Research Group Staff

Oniine Tutorial
May 2017

A Brief Introduction to Bayesian Modeling Using Stan

Objective

Provide a brief introduction to Bayesian modeling and the practical use of Stan for PKPD applications

Primary intended audience: PKPD scientists

Background assumed

- PKPD modeling
- Some familiarity and experience with nonlinear regression, mixed effects modeling and R (or S-PLUS)
- Basic understanding of Bayesian principles.

A Brief Introduction to Bayesian Modeling Using Stan

- This brief online workshop is essentially a set of excerpts from the Metrum 1 or 2 day workshop entitled Getting Started with Bayesian PKPD Modeling Using Stan.
- I will focus on how to construct Stan models and analyze data with them.
- I will skip material on the fundamentals of Bayesian inference, data analysis and computation. For that I encourage you to read the first 7 chapters and chapters 10–12 of Bayesian Data Analysis [1].

A Brief Introduction to Bayesian Modeling Using Stan

- Introduction to Bayesian statistical principles and methods
  - Bayes Rule
  - Bayesian modeling & inference process
- Computation for Bayesian modeling
  - Key challenge of Bayesian modeling and inference: sampling from high-dimensional probability distributions
  - General computational approach: posterior simulation
  - Brief intro to Markov chain Monte Carlo simulation
- Stan basics
  - What is it?
  - How do I get it?
  - How do I run it?
    - Using rstan
  - Stan demo: Linear regression

A Brief Introduction to Bayesian Modeling Using Stan

- Hands-on example 1: Simple nonlinear regression
- Assessing convergence
- Programming hierarchical models (aka mixed effect or population models)
- Prior distributions
- Truncated distributions
- Hands-on example 2: Nonlinear mixed effects

A Brief Introduction to Bayesian Modeling Using Stan

- User-defined functions
- Programming for pharmacometricians:
  - PK models
  - Dosing and observation event schedules
- Hands-on example 3: Population PK
- Dealing with censored data in Stan, e.g., BQL data

Getting set up for the workshop examples

- Download zip file containing workshop materials from the site where you launched this video.
- Unzip downloaded file.
  - The resulting directory contains a directory tree with the following directories: data, deliv, doc, model and script.
- If necessary install R.
  - Download R from a CRAN site, e.g., `https://cran.r-project.org/`.
  - Install both R Base and Rtools.
- Open the script directory.
- Open the pkgSetup.R file using your favorite R environment.
- Set the working directory to the script directory.
- Run the pkgSetup.R script.
  - This will download and install the R packages used in the examples plus any dependencies. Be patient; this takes a while.

Introduction to Bayesian statistical principles and methods

- Bayesian principles and methods provide a coherent framework for:
  - Quantifying uncertainty,
  - Making inferences in the presence of that uncertainty.
- It is also the basis for formal approaches to incremental model building, parameter estimation and other statistical inference as knowledge and data are accumulated.

The two core notions that distinguish Bayesian analysis are:

- Unknown quantities are viewed as random variables, i.e., they are described in terms of probability distributions.
- Bayes rule provides a formal mechanism for combining prior knowledge and new data.

Bayesian approach to statistical inference

- Model parameters and predictions described in terms of probability distributions representing uncertainty
- Results reflect the combined evidence of data and prior knowledge or belief
- Focuses on estimation and inferences related to probabilities of unknown quantities: parameters, future data, hypotheses.
- Inferences are described directly in terms of probabilities:

Bayesian inference
Bayes Rule

Bayes Rule is the basis for inference about model parameters (θ) given data (y) and prior knowledge about model parameters ($p(\theta)$):

$$\begin{aligned} p(\theta|y) &= \frac{p(\theta)\,p(y|\theta)}{p(y)} = \frac{p(\theta)\,p(y|\theta)}{\int p(\theta)\,p(y|\theta)\,d\theta} \\ &\propto p(\theta)\,p(y|\theta) \end{aligned}$$

The p's are probabilities or probability densities of the specified random variables.

Bayesian modeling/inference process

1. Assess prior distribution $p(\theta)$
   - θ viewed as random variables
   - Subjective
   - Ideally base on all available evidence/knowledge (or belief)
   - Or deliberately select a non-informative prior (e.g., reference, vague or improper prior)
2. Construct a model for the data $p(y|\theta)$, also known as the likelihood function when viewed as a function of θ.
3. Calculate posterior distribution $p(\theta|y)$.
   - Use for inferences regarding parameter values
4. Calculate posterior predictive distribution $p(y_{\text{new}}|y)$.
   - Use for inferences regarding future observations

$$p(y_{new}|y) = \int p(y_{new}|\theta)\, p(\theta|y)\, d\theta$$

So what's the bad news?

- You've a selected suitable distributions for $p(\theta)$ and $p(y|\theta)$.
  - θ is a vector of 6 parameters;
  - y is a vector of 200 observations.
- You would like to estimate the expected value of a function of θ, i.e., $E(f(\theta)|y)$.
- You know that $p(\theta|y) \propto p(y|\theta)\, p(\theta)$ so

$$E(f(\theta)|y) = \int_{\theta_1}\int_{\theta_2}\int_{\theta_3}\int_{\theta_4}\int_{\theta_5}\int_{\theta_6} f(\theta)\, p(\theta|y)\, d\theta = \frac{\int_{\theta_1}\int_{\theta_2}\int_{\theta_3}\int_{\theta_4}\int_{\theta_5}\int_{\theta_6} f(\theta)\, p(y|\theta)\, p(\theta)\, d\theta}{\int_{\theta_1}\int_{\theta_2}\int_{\theta_3}\int_{\theta_4}\int_{\theta_5}\int_{\theta_6} p(y|\theta)\, p(\theta)\, d\theta}$$

- Hmmmm. How do you do that?

So what's the bad news?

- If you're lucky those integrals have known analytic solutions, but that is rarely true for PK/PD modeling applications.
- For integrals in fewer dimensions, a numerical quadrature method might be practical. 6 dimensions is pushing quadrature to its limits.
- Now imagine the computational requirements for hierarchical models, e.g., population PK models, with individual-specific parameters in the hundreds!!

But there's hope: posterior simulation

- What if you could simulate samples of θ from the joint posterior distribution?
- Then you could estimate $E\left(f\left(\theta\right)|y\right)$ by the arithmetic mean:

$$E\left(f\left(\theta\right)|y\right) \approx \frac{1}{n}\sum_{i=1}^{n} f\left(\theta_i\right)$$

- More generally, you could characterize the properties of any marginal posterior distribution of a model parameter or function of model parameters, e.g., moments, quantiles, ...
- But how do you simulate samples from a high dimensional joint posterior distribution?
- Markov chain Monte Carlo (MCMC) simulation via Stan is the approach we will explore today.

Inferences from posterior simulations

- Posterior simulation yields vectors of parameters and/or predictions from a joint posterior distribution
- Marginal distributions
  - To describe the posterior distribution of any scalar function of the parameters apply the function to each simulated vector. The empirical distribution of those values approximates the posterior distribution.
  - The marginal distribution of any single parameter is just a special case of that approach.
- Inferences are usually based on moments, probabilities or percentiles from marginal posterior distributions. They are readily estimated from the corresponding sample statistics for the simulated values.

Tools for Bayesian modeling & inference

- There remains the challenge of simulating from high dimensional posterior distributions.
- Markov chain Monte Carlo (MCMC) simulation has become the primary method.
  - Simulation methods for performing high dimensional simulation required for Bayesian modeling.
  - Simulate random variables from the posterior distributions of interest, e.g., means and variances of the population distributions of PK/PD parameters.
  - Inferences follow directly from the distributions of simulated parameter values
    - Point estimates from mean, median or mode.
    - Posterior intervals from percentiles, e.g., 95% interval from the 2.5 and 97.5 percentiles.

Markov Chain Monte Carlo (MCMC) simulation

- Involves random draws from approximate distributions and then correcting those draws to better approximate the joint posterior.
- The samples are drawn sequentially so that each draw depends on the previous one, thus forming a Markov chain.
- Eventually the Markov chain converges (in distribution) to a stationary distribution that is the joint posterior distribution.
- Algorithms for MCMC include:
  - Metropolis-Hastings algorithm
  - Gibbs sampling
  - Hamiltonian Monte Carlo (HMC) simulation
- MCMC samples are serially correlated:
  - Inferences based on MCMC require more samples than would be required for independent samples
- Practical consequences:
  - Use only samples drawn after convergence is achieved, i.e., discard samples from a warmup phase.
  - Draw more samples than you would for independent random draws.

Gibbs sampling

- In most cases a convergent Markov chain may be constructed by progressively sampling from the univariate full conditional distributions:

$$
\begin{aligned}
\theta_1^i &\sim p\left(\theta_1 | \theta_2^{i-1}, \theta_3^{i-1}, \ldots, \theta_n^{i-1}, y\right) \\
\theta_2^i &\sim p\left(\theta_2 | \theta_1^{i}, \theta_3^{i-1}, \ldots, \theta_n^{i-1}, y\right) \\
&\vdots \\
\theta_n^i &\sim p\left(\theta_n | \theta_1^{i}, \theta_2^{i}, \ldots, \theta_{n-1}^{i}, y\right)
\end{aligned}
$$

- This reduces the multivariate posterior sampling problem to a sequence of more manageable univariate sampling problems.

Metropolis-Hasting algorithm

- The Metropolis-Hastings is a general purpose multivariate MCMC algorithm.
- It requires selection of a conditional proposal density $q(y|x)$ that is easy to sample from.
- To generate $x^{(t+1)} \sim f$ given a previous value $x^{(t)}$:

1: Generate $y_t \sim q\left(y|x^{(t)}\right)$.

2: $x^{(t+1)} = \begin{cases} y_t, & \text{with probability } \rho\left(x^{(t)}, y_t\right) \\ x^{(t)}, & \text{with probability } 1-\rho\left(x^{(t)}, y_t\right) \end{cases}$

where $\rho(x,y) = \min\left\{\frac{f(y)}{f(x)}\frac{q(x|y)}{q(y|x)}, 1\right\}$

Hamiltonian Monte Carlo (HMC) simulation

Physical analogy to motivate HMC

- In classical mechanics the Hamiltonian equations describe the evolution of a system over time.
- The state of the system is described in terms of kinetic energy as a function of momentum (mass $\times$ velocity) and potential energy as a function of position.
- For the analogy equate the model parameters θ to position and equate a set of auxiliary parameters ρ to momentum.
- Now define a Hamiltonian in terms of the joint posterior distribution of θ and ρ:

$$\begin{aligned} H(\theta,\rho) &= -\log\left(p(\theta,\rho|y)\right) = -\log\left(p(\theta|y)\,p(\rho|\theta,y)\right) \\ &= -\log\left(p(\theta|y)\right) - \log\left(p(\rho|\theta,y)\right) \\ &= V(\theta) + T(\rho|\theta) \\ V(\theta) &= -log\left(p(\theta|y)\right) = \text{potential energy} \\ T(\rho|\theta) &= -log\left(p(\rho|\theta,y)\right) = \text{kinetic energy} \end{aligned}$$

Hamiltonian Monte Carlo (HMC) simulation

- θ is what we really care about.
- ρ allows the use of Hamiltonian mechanics to more efficiently move through the relevant parts of the parameter space.
- Usually the distribution of ρ is chosen to be independent of θ, e.g., $p(\rho|\theta) = p(\rho) = N(0, \Sigma)$.
- Suppose we place a frictionless particle on the potential energy surface $(-log(p(\theta|y)))$ at some position θ^{t-1}.
  - We give it a shove that imparts a momentum ρ^{t-1} to that particle at time $t-1$.
  - The particle moves over that surface according to Hamiltonian dynamics.
  - Now stop the particle at time t and measure its position θ^t.
  - Now randomly sample a new momentum from $p(\rho)$ and give the particle another shove, and so on...

Hamiltonian Monte Carlo (HMC) simulation

- Though the initial momentum at each step is random, the subsequent path will favor regions of lower potential energy (higher probability density).
- The set of sampled positions are distributed according to the target posterior density.
- In practice the Hamiltonian equations are solved numerically. As a result some error is introduced in the estimated path.
- A Metropolis step is used to assure that the position samples converge in distribution to the target distribution.

The HMC algorithm

Repeat the following steps:

1. Sample $\rho^{t-1} \sim N(0, \Sigma)$
2. Simultaneously update θ and ρ by numerically solving the Hamiltonian equations using the leapfrog method to generate a proposal θ^* for θ^t.
3. Apply a Metropolis step to decide whether to accept or reject the proposal θ^* as θ^t.

The leapfrog method

Using the starting values θ^{t-1} and ρ^{t-1} the leapfrog algorithm alternates half-step updates of ρ with full step updates of θ:

$$\rho \leftarrow \rho - \frac{\epsilon}{2}\frac{\partial V}{\partial \theta} = \rho + \frac{\epsilon}{2}\frac{d \log\left(p\left(\theta|Y\right)\right)}{d\theta}$$
$$\theta \leftarrow \theta + \epsilon \Sigma \rho$$
$$\rho \leftarrow \rho - \frac{\epsilon}{2}\frac{\partial V}{\partial \theta} = \rho + \frac{\epsilon}{2}\frac{d \log\left(p\left(\theta|Y\right)\right)}{d\theta}$$

For each HMC iteration repeat this L times to yield the proposal values θ^* and ρ^*.

The Metropolis step

- Compute the ratio:

$$r = \exp\left(H\left(\theta^{t-1}, \rho^{t-1}\right) - H\left(\theta^{*}, \rho^{*}\right)\right)$$
$$= \frac{p\left(\theta^{*}|y\right) p\left(\rho^{*}\right)}{p\left(\theta^{t-1}|y\right) p\left(\rho^{t-1}\right)}$$

- Accept/reject step:

$$\theta^{t} = \begin{cases} \theta^{*}, & \text{with probability } \min\left(r, 1\right) \\ \theta^{t-1}, & \text{otherwise} \end{cases}$$

- Since ρ is sampled independently of θ and previous values of ρ, we just discard ρ^{*} and sample a new value for the next HMC iteration.

HMC algorithm parameters

Parameters that must be set: discretization time ϵ, number of leapfrog steps L and mass matrix Σ^{-1}.

Sampling efficiency is very sensitive to those parameters:

- ϵ too large $\rightarrow$ too many proposals rejected
- ϵ too small $\rightarrow$ long simulation times
- L too large $\rightarrow$ too much work for each iteration
- L too small $\rightarrow$ devolves to a random walk
- If Σ^{-1} is poorly tuned to the problem, ϵ needs to be decreased and L increased to maintain precision and efficiency.

Stan automatically optimizes those parameters using the NUTS (no U-turn sampling) algorithm [19].

HMC performance

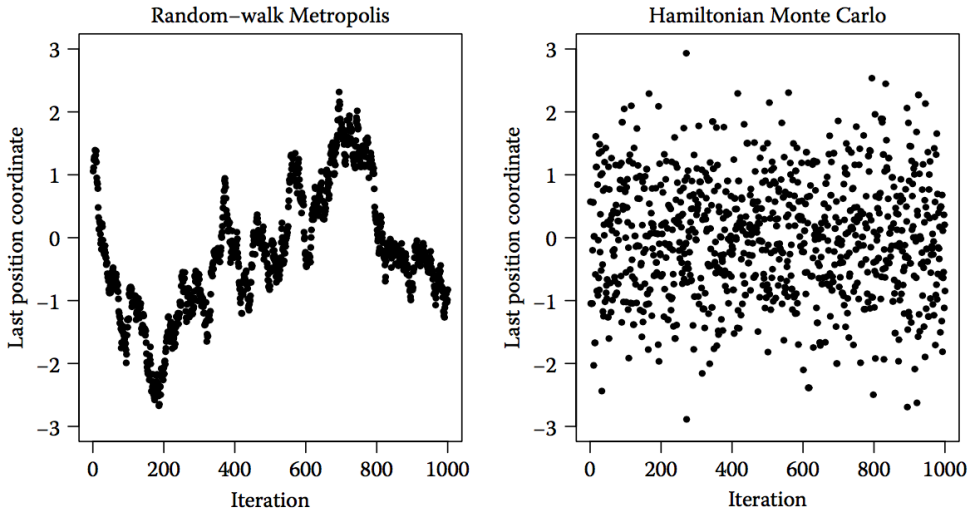


FIGURE 5.6
Values for the variable with largest standard deviation for the 100-dimensional example, from a random-walk Metropolis run and an HMC run with $L = 150$. To match computation time, 150 updates were counted as one iteration for random-walk Metropolis.

from RM Neal. MCMC Using Hamiltonian Dynamics (2011) [20]

HMC performance

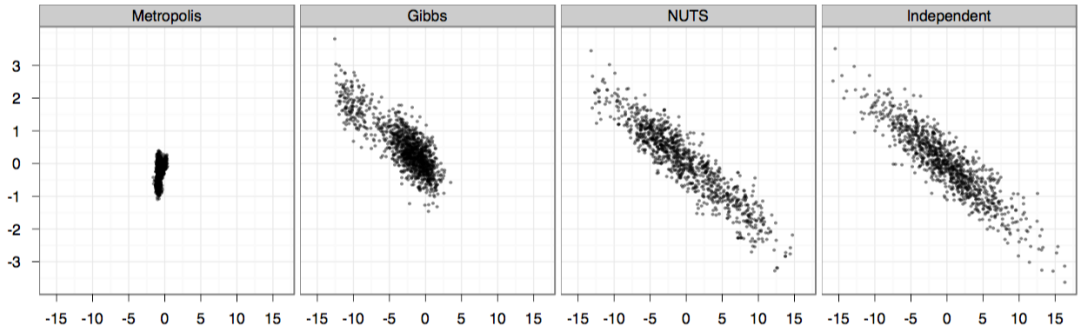


Figure 7: *Samples generated by random-walk Metropolis, Gibbs sampling, and NUTS. The plots compare 1,000 independent draws from a highly correlated 250-dimensional distribution (right) with 1,000,000 samples (thinned to 1,000 samples for display) generated by random-walk Metropolis (left), 1,000,000 samples (thinned to 1,000 samples for display) generated by Gibbs sampling (second from left), and 1,000 samples generated by NUTS (second from right). Only the first two dimensions are shown here.*

from MD Hoffman and A Gelman. The no-U-turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo (2014) [19]

HMC issues/limitations

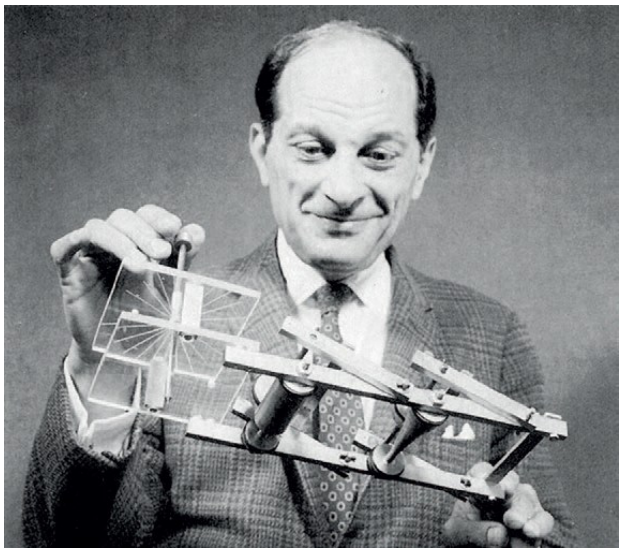
- Requires calculation of the gradient $\frac{d \log(p(\theta|Y))}{d\theta}$
- Suitable for sampling of continuous parameters only
  - Cannot sample discrete parameters
  - Discrete data is OK as long as the likelihood depends only on continuous parameters.
  - Models with discrete parameters, e.g., finite mixture models, can often be implemented by marginalizing out the discrete parameters.

Stan: What is it?

Stan (`http://mc-stan.org/`) is a general purpose Bayesian modeling package [21]

- General model specification language
- Primarily uses a Hamiltonian Monte Carlo (HMC) sampler (standard HMC or NUTS (no U-turn sampler)). Other methods include:
  - Optimization for estimation of posterior modes.
  - Variational inference for approximate Bayesian inference.
- Developed by a team headed by Andrew Gelman of Columbia University
- C++ program available with several interfaces: rstan, PyStan, CmdStan, MatlabStan, Stan.jl, StataStan, ShinyStan

Stan: Why is it called that?



Stanislaw Ulam, co-inventor of Monte Carlo methods, holding an analog computer known as the FERMIAC that performed a mechanical simulation of random diffusion of neutrons (`http://fas.org/sgp/othergov/doe/lanl/pubs/00326866.pdf`).

Stan: How do I get it?

- Most Stan interfaces may be downloaded from the Stan website (`http://mc-stan.org/`).
- rstan [22] is available on CRAN (`https://cran.r-project.org/`)
- Documentation
  - Stan: `https://github.com/stan-dev/stan/releases/download/v2.15.0/stan-reference-2.15.0.pdf`
  - rstan: `https://cran.r-project.org/web/packages/rstan/vignettes/rstan.html`

Stan demo: Linear regression

Linear model with weakly informative prior distributions for slope and intercept:

$$y_i \sim N(\hat{y}_i, \sigma)$$
$$\hat{y}_i = a + bx_i$$

Prior distributions:

$$a \sim N(0, 100)$$
$$b \sim N(0, 100)$$
$$\sigma \sim \text{Cauchy}(0, 2)$$

Data (R format):

- x = c(1,2,3,4,5,6,7,8,9,10)
- y = c(5.19,6.56,9.19,8.09,7.6,7.08,6.74,9.3,8.98,11.5)

Stan model specification language

Very flexible model specification language

- Imperative language: statements executed in the order in which they are written.
- Computational control structures, e.g., if-then-else, for and while loops
- Large collection of:
  - Operators
  - Built-in functions
  - Probability distributions
- User-defined functions

Stan program blocks

The essential program blocks

```
data{
  # Declare data:  variables that remain fixed.
}

parameters{
  # Declare parameters:  random variables.
}

model{
  # Model calculations.
}
```



```
data{
  real x[10];
  real y[10];
}

parameters{
  real a;
  real b;
  real<lower=0> sigma;
}

model{
  # Model calculations.
}
```

Declarations

- All variables must be declared
- Primitive data types
  - real
  - integer (int)
- Compound data types
  - array (real[] or int[])
  - vector
  - row_vector
  - matrix
- Variables may be bounded
  - Required for parameters with bounded priors.

```
model{
  real ymean[10];

  a ~ normal(0, 100);
  b ~ normal(0, 100);
  sigma ~ cauchy(0, 2);

  for(i in 1:10){
    ymean[i] = a + b * x[i];
    y[i] ~ normal(ymean[i],
        sigma);
  }
}
```

Code the model and additional declarations in the model block

- Syntax similar to C, R or BUGS
- = for assignment statements
- ~ for sampling statements
  - Normal parameters are mean and standard deviation.
- Variables declared in the model block may not be monitored.
- Loop indices do not have to be declared.


```
model{
  real ymean[10];

  a ~ normal(0, 100);
  b ~ normal(0, 100);
  sigma ~ cauchy(0, 2);

  for(i in 1:10){
    ymean[i] = a + b * x[i];
    y[i] ~ normal(ymean[i], sigma);
    ypred[i] ~ normal(ymean[i], sigma);
  }
}
```

Additional code to generate predictions for model checking

- Sampling statement used to generate samples from the posterior predictive distributions
- ypred is declared in parameters block.

Using rstan

rstan is an R package available from CRAN sites, e.g., `https://cran.r-project.org/`

- Contains several R functions for:
  - Translating Stan models to C++,
  - Compiling the resulting C++ code,
  - Analyzing data using the Stan model,
  - Analyzing, summarizing and plotting the resulting MCMC samples.
- We will focus on only a few of those functions—particularly stan().


```
library(rstan)

data = list(
    x = c(1,2,3,4,5,6,7,8,9,10),
    y = c(5.19,6.56,9.19,8.09,7.6,
        7.08,6.74,9.3,8.98,11.5)
)

init = function() list(
    a = rnorm(1, 6, 3),
    b = rnorm(1, 0.5, 0.5),
    sigma = exp(rnorm(1,log(1.5), 1)))

fit <- stan(file = file.path(dirname(
     getwd()), "model", "linear1.stan
     "),
            data = data,
            pars = c("a","b","sigma"),
            iter = 2000,
            init = init)
```

Minimal R code for the linear model example

- Data format is an R list
  - Named list of scalars, vectors, matrices or arrays
  - Names and data types must be consistent with the declarations in the Stan model data block.
  - Numbers only—no character strings

```
library(rstan)

data = list(
    x = c(1,2,3,4,5,6,7,8,9,10),
    y = c(5.19,6.56,9.19,8.09,7.6,
        7.08,6.74,9.3,8.98,11.5)
)

init = function() list(
    a = rnorm(1, 6, 3),
    b = rnorm(1, 0.5, 0.5),
    sigma = exp(rnorm(1,log(1.5), 1)))

fit <- stan(file = file.path(dirname(
     getwd()), "model", "linear1.stan
     "),
            data = data,
            pars = c("a","b","sigma"),
            iter = 2000,
            init = init)
```

Minimal R code for the linear model example

- Initial estimates
  - Not strictly required, but recommended in nearly all cases
  - Specify as a list or function that returns a list.
  - Use of a function with random number generation results in different initial estimates for each chain—desirable for convergence assessment.
- Runs 4 chains by default.

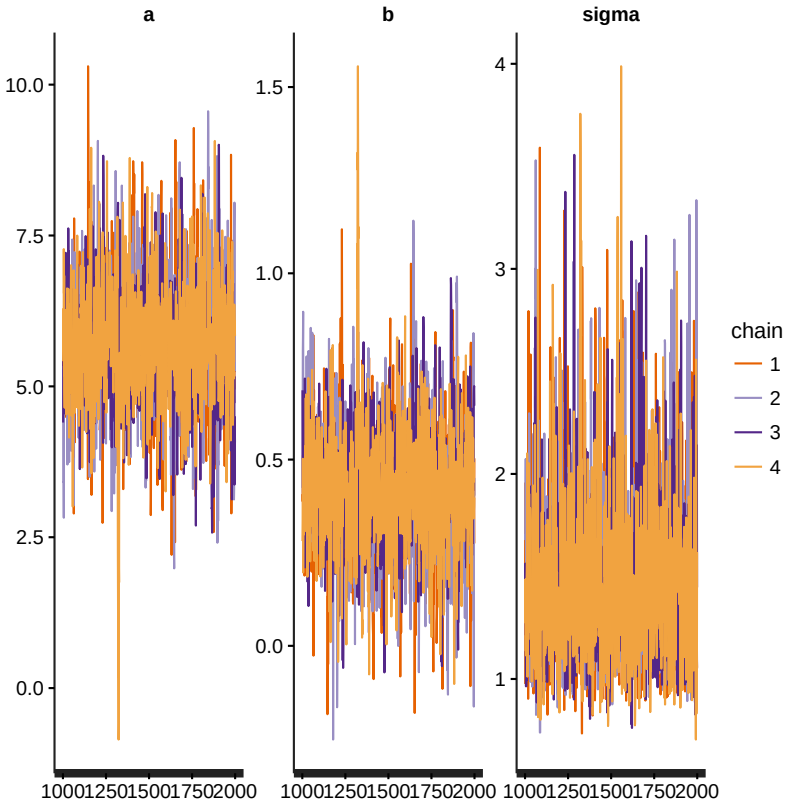
Stan demo

linear1: minimal Stan model and R script for linear regression example

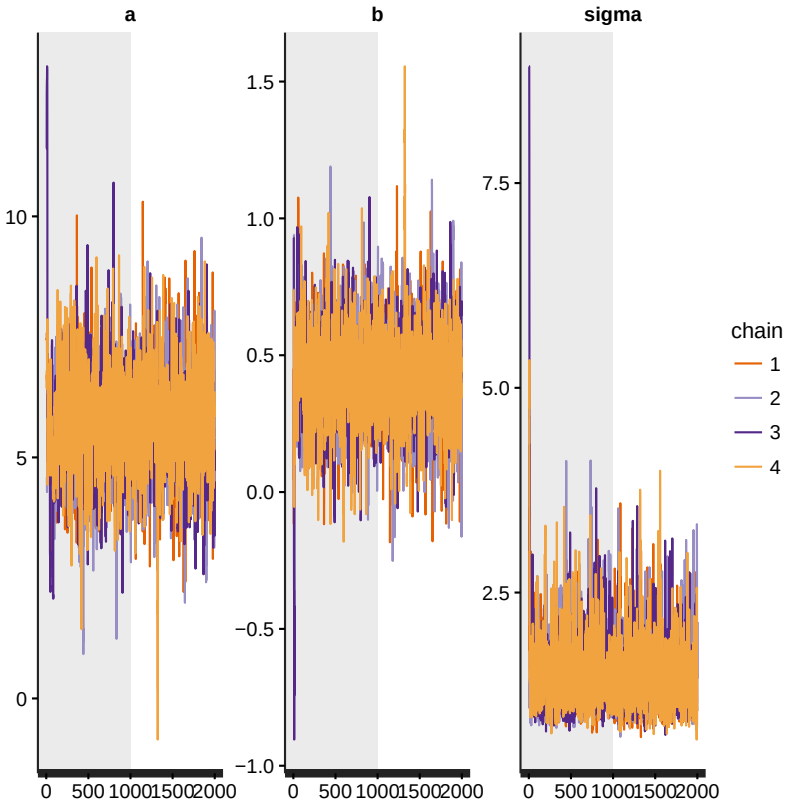
- Stan model: model/linear1.stan
- R script: script/linear1.R
- MCMC results summarized using rstan functions from R command line


```
> fit
Inference for Stan model: linear1.
4 chains, each with iter=2000; warmup=1000; thin=1;
post-warmup draws per chain=1000, total post-warmup draws=4000.

       mean se_mean  sd   2.5%  25%  50%  75% 97.5% n_eff Rhat
a      5.72    0.03 1.01  3.70 5.12 5.74 6.32 7.80  1213    1
b      0.42    0.00 0.16  0.09 0.32 0.42 0.52 0.75  1148    1
sigma  1.48    0.01 0.40  0.93 1.20 1.40 1.67 2.50  1489    1
lp__  -8.47    0.04 1.36 -12.12 -9.03 -8.10 -7.48 -6.98 1051 1

Samples were drawn using NUTS(diag_e) at Fri Oct 7 10:29:45 2016.
For each parameter, n_eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor on split chains (at
convergence, Rhat=1).
```

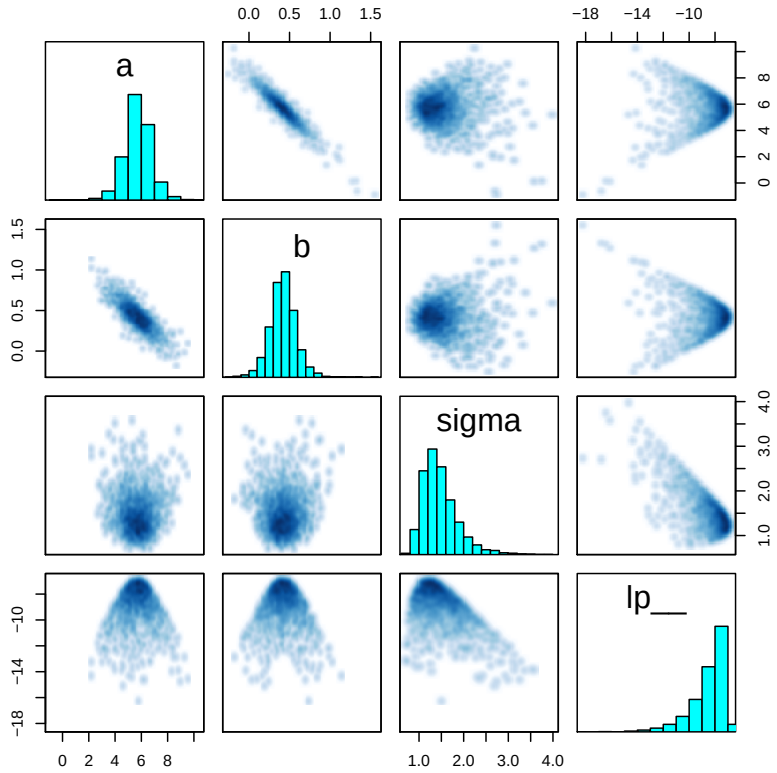
```
> stan_trace(fit)
```




```
> stan_trace(fit,
    inc_warmup = TRUE
    )
```



```
> pairs(fit)
```



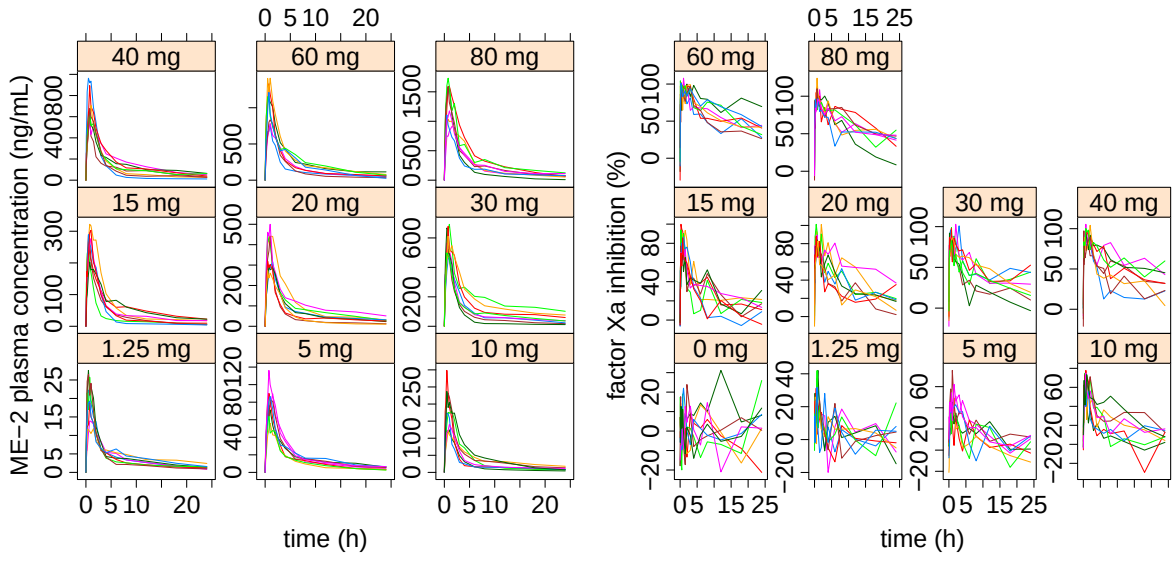
linear2: more complete Stan model and R script

- Stan model: model/linear2.stan
- R script: script/linear2.R
- Posterior predictions generated via Stan model.
- R script now summarizes the MCMC results including simple posterior predictive checks (PPC).
  - Summary table and plots written to files

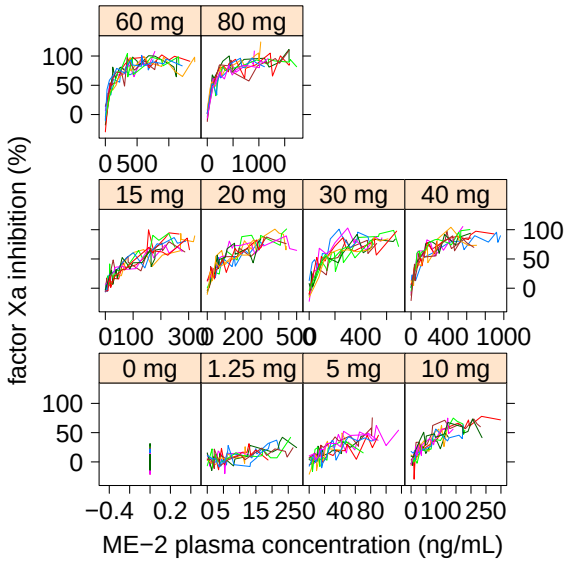
Hands-on session 1
PK-PD modeling of time-averaged biomarker and PK data

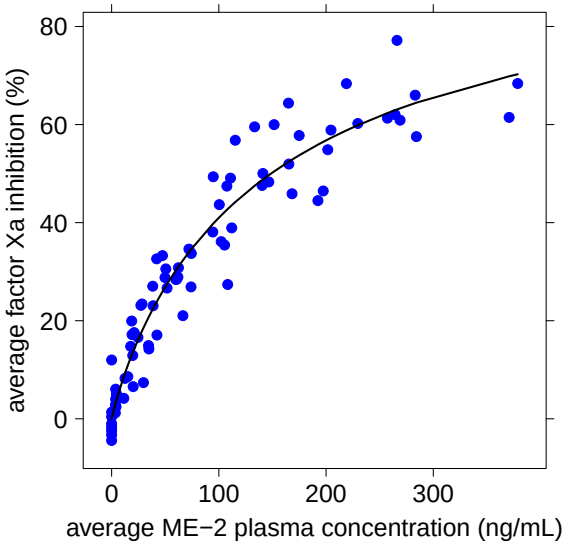
- Phase 1 single dose study in healthy volunteers
  - Parallel dose-escalation design
  - 8 subjects per dose arm
  - Single doses of ME-2
    - Placebo, 1.25, 5, 10, 15, 20, 30, 40, 60 and 80 mg
  - PK: plasma concentrations of parent drug
  - Biomarker: ex vivo inhibition of factor Xa activity in plasma
    - PK and biomarker measured at 0, 0.083, 0.167, 0.25, 0.5, 0.75, 1, 1.5, 2, 3, 4, 6, 8, 12, 18 and 24 hours after dose.
- Hands-on exercise:
  - Model relationship between time-averaged factor Xa inhibition and time-averaged ME-2 plasma concentrations

Hands-on session 1
EDA: PK and biomarker data



Hands-on session 1
EDA: Relationship between biomarker and PK data





Hands-on session 1
Proposed model

- Sigmoid Emax model relating time-averaged % inhibition of factor Xa activity to time-averaged ME-2 plasma concentration in the i^{th} subject:

$$\begin{aligned} \overline{E}_{24,i} &\sim N\left(\hat{E}_{24,i}, \sigma\right) \\ \hat{E}_{24,i} &= \frac{E_{max}\overline{c}^{\gamma}_{24,i}}{EC^{\gamma}_{50} + \overline{c}^{\gamma}_{24,i}} \end{aligned}$$

- Some possible weakly informative prior distributions:

$$\begin{aligned} E_{max} &\sim U(0, 100) \\ EC_{50} &\sim \text{half-}N(0, 250) \\ \gamma &\sim \text{half-}N(0, 5) \\ \sigma &\sim \text{half-Cauchy}(0, 10) \end{aligned}$$

See [23] for half-Cauchy rationale.

Hands-on session 1
Files

- Data: data/derived/fxa.data.avg.csv
- Version 1
  - Uses linear2 as a template
  - Stan model: model/fxaInhibitAvg1.stan
  - R script: script/fxaInhibitAvg1.R
- Version 2
  - Introduces additional Stan features
    - Vectors and vectorized calculations
    - Additional program blocks: “transformed parameters” and “generated quantities”
  - Stan model: model/fxaInhibitAvg2.stan
  - R script: script/fxaInhibitAvg2.R

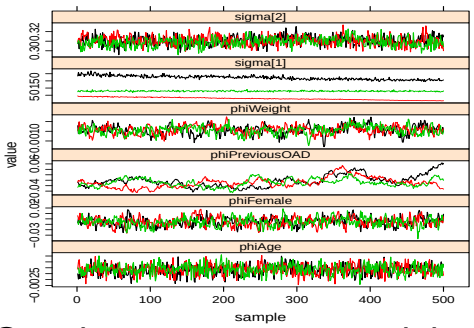
Assessing convergence & adequacy of sample sizes

- Early samples may be unrepresentative of the target distribution
- MCMC samples within a chain are autocorrelated
  - Inferences based on MCMC samples are less precise than those from the same number of independent samples
  - Autocorrelation also influences the rate of convergence

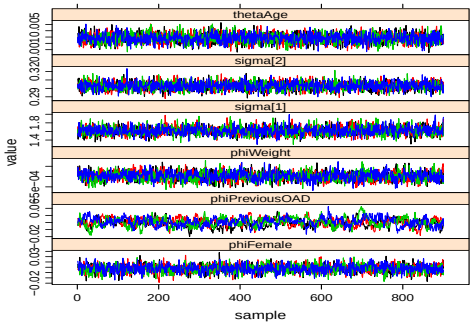
Assessing convergence & adequacy of sample sizes

- Use a warmup phase, i.e., discard early iterations
- Monitor convergence via multiple chains with different starting points
  - Look for chains to converge to a common distribution
  - You want chain history plots to look more like straight horizontal fuzzy caterpillars than wiggly snakes
  - Monitor Gelman-Rubin diagnostics (Rhat) and/or Gelman-Rubin-Brooks plots
    - Essentially ratios of total variance to within chain variance.
    - Should approach 1 for all parameters of interest on convergence

Poor convergence & mixing



Good convergence & mixing

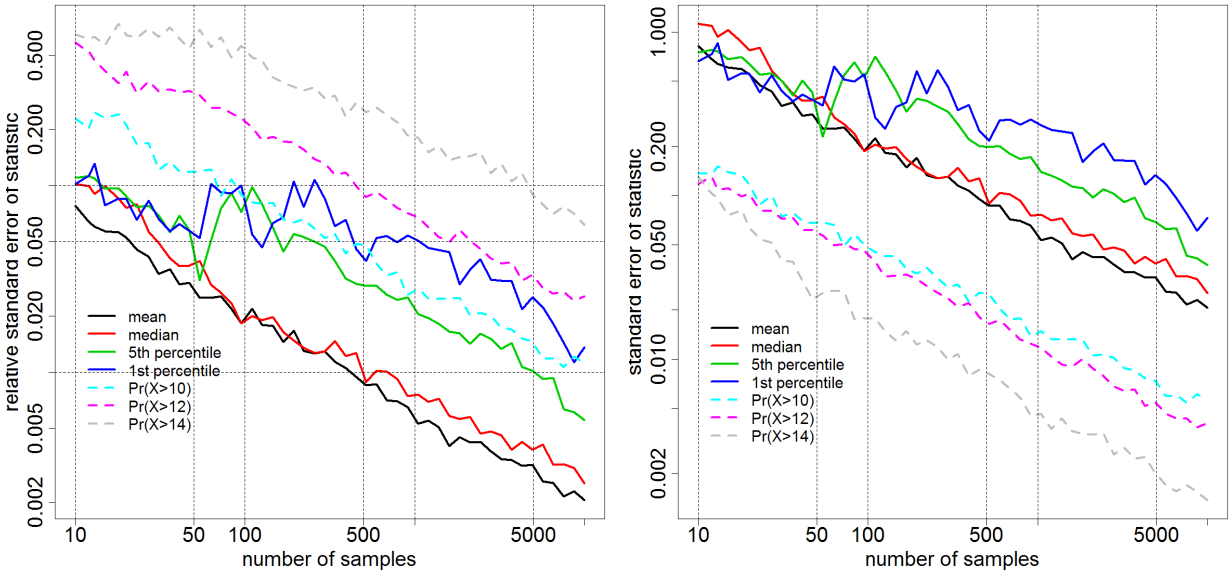


How many samples?

- Number of samples depends on the inference(s) of interest and the desired precision
- For independent samples:
  - A posterior mean of a parameter θ is estimated with an error of $\sim \frac{s_\theta}{\sqrt{n}}$ where s_θ is the standard deviation of the simulated values of θ and n is the number of samples.
  - A probability p is estimated with an error of $\sim \sqrt{\frac{p(1-p)}{n}}$.
- In general more samples are required for estimating tail quantiles and probabilities than for central tendencies and probabilities near 0.5

How many samples?

- Suppose the posterior distribution of θ is $N(\mu = 10, \sigma = 2)$.
- What if we use simulation to estimate various features of that distribution?
- The following are bootstrap estimates of error due to simulation:



How many MCMC samples?

- Given a set of MCMC samples it is possible to adjust for autocorrelation to estimate:
  - The equivalent number of independent samples, aka effective sample size (n_eff)
  - The standard error in the estimated posterior mean
  - The rstan print and summary functions provide estimates of each.
- Guidance based on independent samples may then be applied

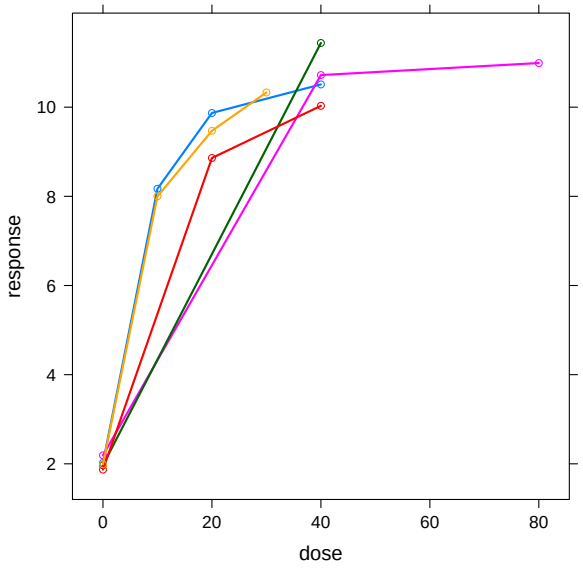
For more rigorous and comprehensive treatment see Robert and Casella 2004 and 2010 [8, 7].

Programming hierarchical models (e.g. population models)

- Consider the usual case where you want to model 2 levels of variability, e.g., inter-individual and residual
- The bulk of the data set consists of a set of equal length vectors
  - Data value(s) for each observation
  - Individual identifier: Use a consecutive sequence of integers
  - Covariates (dose, time, etc.)
- May include constants such as numbers of observations and patients, parameters of prior distributions, etc.

Stan demo: Hierarchical model for dose-response

- Data: sample mean response from 5 clinical trials
- Illustration of meta-analysis where inter-trial differences are modeled as random



Study	Dose	n	Mean Response
1	0	40	2.02
1	10	40	8.17
1	20	40	9.87
1	40	40	10.51
2	0	40	2.19
2	40	40	10.72
2	80	40	10.99
3	0	200	1.97
3	40	200	11.44
4	0	50	1.87
4	20	50	8.86
4	40	50	10.03
5	0	20	1.99
5	10	20	8
5	20	20	9.47
5	30	20	10.33

Hierarchical model for dose-response

Emax model with inter-trial variation in E_0 & E_{max}:

$$\begin{aligned} \log(E_{ij}) &\sim N\left(\log\left(\widehat{E}_{ij}\right), \frac{\sigma}{\sqrt{n_{ij}}}\right) \\ \widehat{E}_{ij} &= E_{0,j} + \frac{E_{max,j} D_{ij}}{ED_{50} + D_{ij}} \\ \log(E_{0,j}) &\sim N\left(\log\left(\widehat{E_0}\right), \omega_{E_0}\right) \quad \log(E_{max,j}) \sim N\left(\log\left(\widehat{E_{max}}\right), \omega_{E_{max}}\right) \end{aligned}$$

Weakly informative prior distributions

$$\begin{aligned} \widehat{E_0} &\sim \text{half-}N(0,10) \quad \widehat{E_{max}} \sim \text{half-}N(0,50) \quad ED_{50} \sim \text{half-}N(0,50) \\ \sigma &\sim \text{half-Cauchy}(0,2) \quad \omega_{E_0} \sim \text{half-Cauchy}(0,2) \quad \omega_{E_{max}} \sim \text{half-Cauchy}(0,2) \end{aligned}$$

Stan demo: Hierarchical model for dose-response
Files

- Data: data/derived/doseResponse.csv
- Stan model: model/doseResponseMBMA1.stan
- R script: script/doseResponseMBMA1.R

```
 parameters{
      ⋮
  vector<lower = 0>[nStudies] e0;
  vector<lower = 0>[nStudies] emax;
}

transformed parameters{
      ⋮
  for(i in 1:nArms){
    responseHat[i] = e0[study[i]] +
        emax[study[i]] * dose[i] / (ed50
         + dose[i]);
  }
}
```

Stan program excerpts

- Nested indexing of study-specific parameters
- Sampling statements specify distributions for study-specific parameters.
  - Explicit loops not required

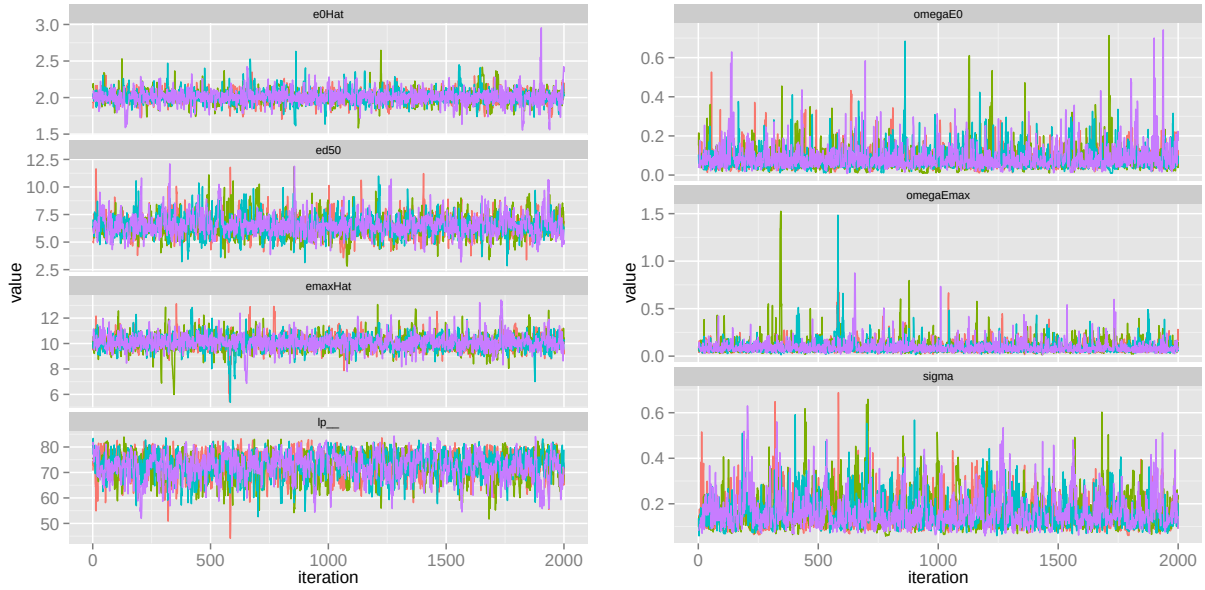

```
      ⋮
model{
      ⋮
  e0 ~ lognormal(log(e0Hat), omegaE0);
  emax ~ lognormal(log(emaxHat),
      omegaEmax);
  logResponse ~ normal(log(responseHat)
      , sigma ./ exp(log(n) / 2));
}
```

Stan program excerpts

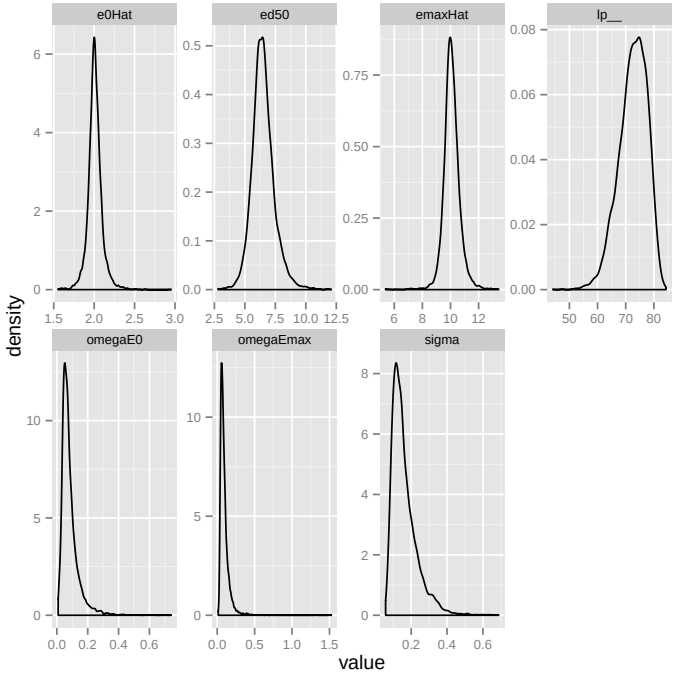
- Nested indexing of study-specific parameters
- Sampling statements specify distributions for study-specific parameters.
  - Explicit loops not required

Data: data/derived/doseResponse.csv
Stan program: model/doseResponseMBMA1.stan
R script: script/doseResponseMBMA1.R

Stan demo
MCMC history plots for model parameters



Stan demo
Posterior marginal densities of model parameters



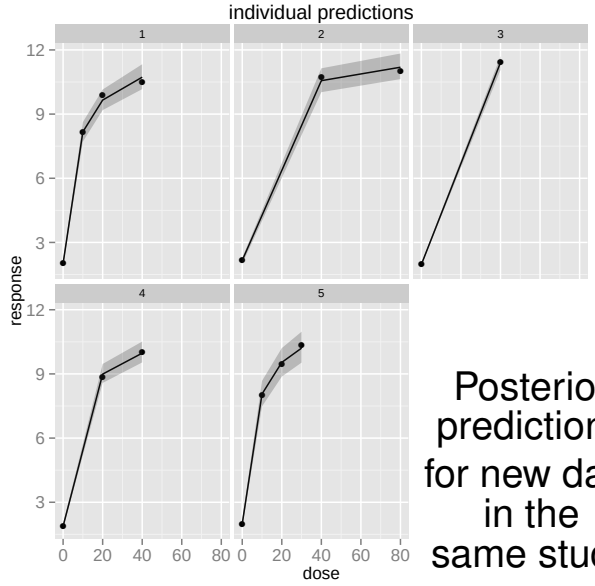
Stan demo

Summary statistics for posterior marginal distributions of model parameters

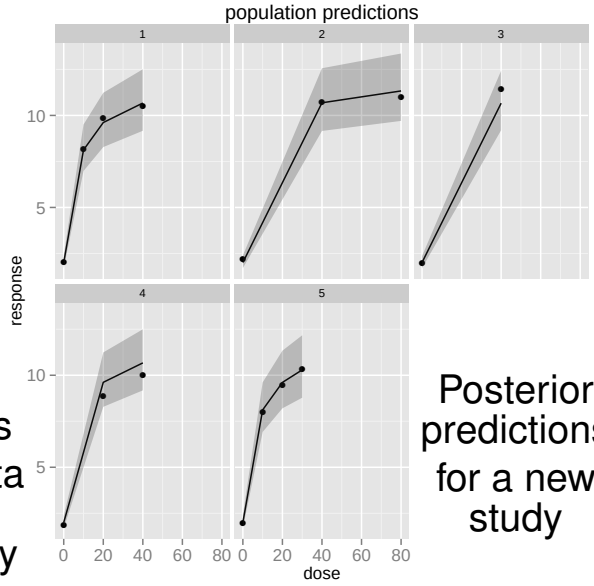
parameter	mean	se_mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
lp__	72.6	0.175	5.03	61.7	69.4	73	76.3	80.7	825	1.01
e0Hat	2.01	0.00212	0.0918	1.83	1.96	2.01	2.05	2.21	1870	1
emaxHat	10.1	0.0141	0.598	9.01	9.75	10	10.4	11.3	1800	1
ed50	6.49	0.0233	0.93	4.82	5.92	6.42	6.98	8.55	1590	1
sigma	0.162	0.00247	0.0719	0.0769	0.112	0.143	0.192	0.346	850	1.01
omegaE0	0.0832	0.00144	0.058	0.0216	0.0486	0.0683	0.0989	0.239	1630	1
omegaEmax	0.0954	0.00184	0.0743	0.0345	0.0572	0.0778	0.11	0.252	1630	1

Stan demo

Model predictions (median + 90% prediction intervals) compared to observed data

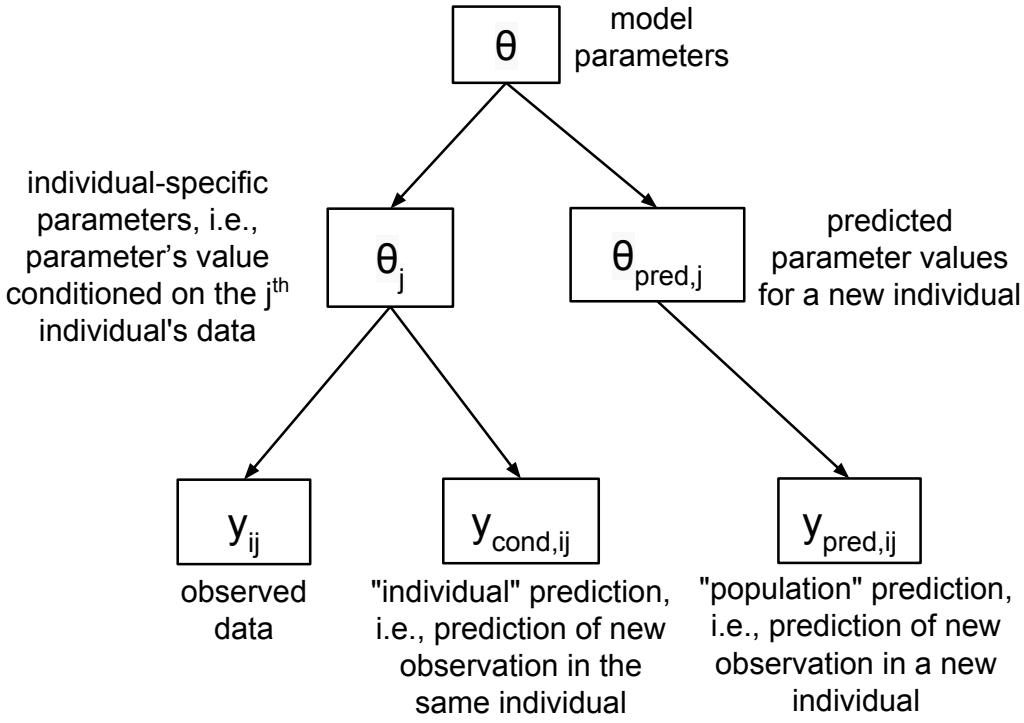


Posterior predictions for new data in the same study



Posterior predictions for a new study

”Individual” and ”population” predictions



Brief discussion on prior distributions

- Think of prior distributions as part of the model.
- Priors should be chosen and subjected to scrutiny much like other model components.
- Model checking should ideally include sensitivity analysis of the priors.
- Choice of priors is most critical with sparse or limited data.

See `https://github.com/stan-dev/stan/wiki/Prior-Choice-Recommendations`

What is the function of a prior distribution

- Represent prior knowledge
- Regularization to facilitate computation
  - Typically weakly-moderately informative
  - E.g., Cauchy with most of its mass in a plausible range, but heavy tails allow for diagnosis of prior-posterior discrepancies.

What does it mean to be informative, uninformative or weakly informative?

Not well defined, but here's an attempt at some loose definitions:

- Weakly informative prior: A prior that rules out unreasonable parameter values but is not so strong as to rule out values that might make sense
- Informative prior: A prior that purposely represents information intended to influence the posterior distribution
  - To capture prior knowledge
  - To challenge the analysis with competing points of view, e.g., use of pessimistic or optimistic priors.
- Uninformative prior: Ostensibly a prior that represents no information and therefore "let's the data tell the story."
  - E.g., a constant over the entire real line—an improper prior

Beware: That "uninformative" prior might not be!

- Suppose you use an improper prior for a standard deviation—a constant over the positive real line.
- That means all positive values are equally likely. Sounds like a reasonable definition of uninformative doesn't it?
- But that means that the prior assigns infinitely more probability to the set of values greater than any fixed value you care to choose.
- This will tend to bias the posterior to high values.
- Bottom line: A uniform distribution does not automatically confer uninformativeness.

Brief note on truncated distributions

- Truncated priors
  - Parameter bounds are sufficient
- Truncated distributions for derived parameters or data
  - T(,) distribution modifier, e.g.,

```
real<lower = L, upper = U> y;

y ~ normal(mu, sigma) T(L, U);
```

    for a truncated normal distribution with lower bound L and upper bound U.
  - Univariate distributions only
  - Not vectorized
- Not the same as censoring!

Hands-on session 2

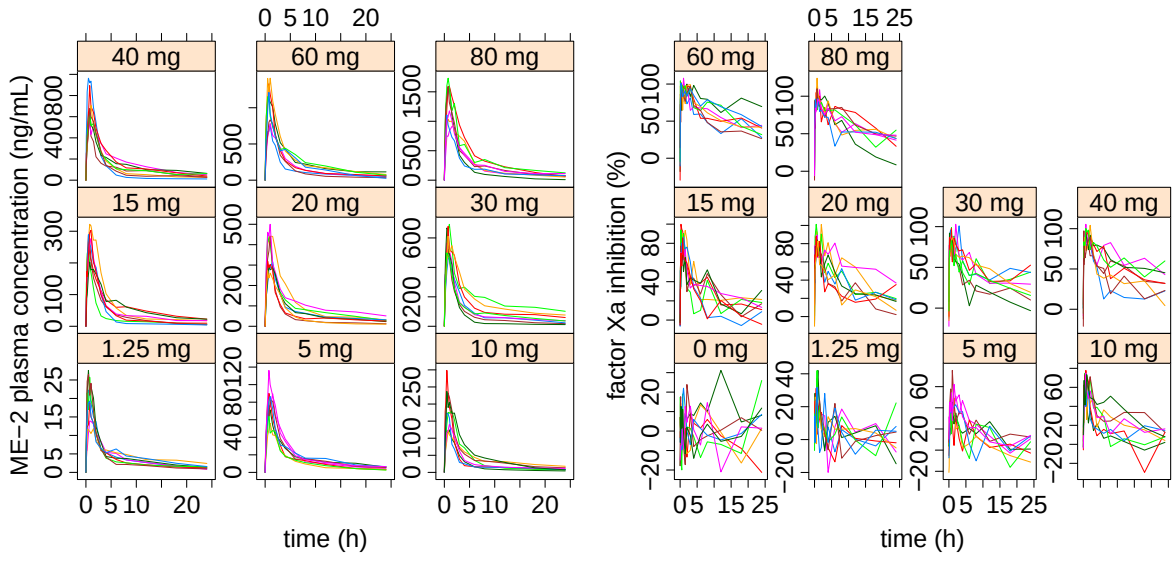
Population PK-PD modeling of time-matched biomarker and PK data

Now we analyze the time course data from the same Phase I study we analyzed in hands-on session 1:

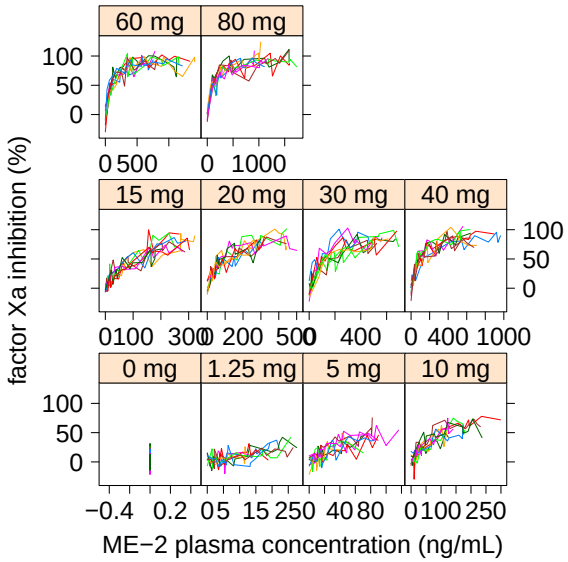
- Phase 1 single dose study in healthy volunteers
  - Parallel dose-escalation design
  - 8 subjects per dose arm
  - Single doses of ME-2
    - Placebo, 1.25, 5, 10, 15, 20, 30, 40, 60 and 80 mg
  - PK: plasma concentrations of parent drug
  - Biomarker: ex vivo inhibition of factor Xa activity in plasma
    - PK and biomarker measured at 0, 0.083, 0.167, 0.25, 0.5, 0.75, 1, 1.5, 2, 3, 4, 6, 8, 12, 18 and 24 hours after dose.
- Hands-on exercise:
  - Apply a direct action PK/PD model to the time-matched factor Xa inhibition and ME-2 plasma concentrations.

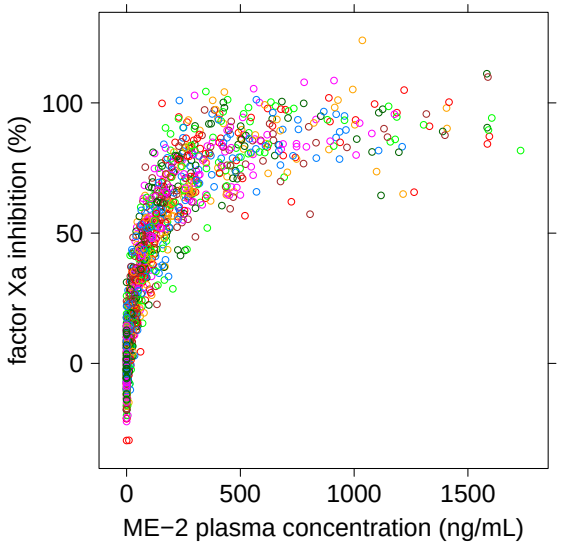
Hands-on session 2

EDA: PK and biomarker data



Hands-on session 2
EDA: Relationship between biomarker and PK data





Hands-on session 2
Proposed model

- Sigmoid Emax model relating % inhibition of factor Xa activity to ME-2 plasma concentration on the i^{th} occasion in the j^{th} subject:

$$\begin{aligned} E_{ij} &\sim N\left(\hat{E}_{ij}, \sigma\right) \\ \hat{E}_{ij} &= \frac{E_{max} c_{ij}^{\gamma}}{EC_{50,j}^{\gamma} + c_{ij}^{\gamma}} \\ \log\left(EC_{50,j}\right) &\sim N\left(\log\left(\widehat{EC_{50}}\right), \omega_{EC_{50}}\right) \end{aligned}$$

- Some possible weakly informative prior distributions:

$$\begin{aligned} E_{max} &\sim U(0, 100) \quad \widehat{EC_{50}} \sim \text{half-}N(0, 250) \\ \gamma &\sim \text{half-}N(0, 5) \\ \omega_{EC_{50}} &\sim \text{half-Cauchy}(0, 1) \quad \sigma \sim \text{half-Cauchy}(0, 10) \end{aligned}$$

Hands-on session 2
Files

- Data: data/derived/fxa.data.csv
- Stan model: model/fxaInhibit1.stan
- R script: script/fxaInhibit1.R
- Non-centered parametrization to improve sampling efficiency and prevent divergent transitions
  - Stan model: model/fxaInhibit1Ncp.stan
  - R script: script/fxaInhibit1Ncp.R

User-defined functions

New Stan language functions may be defined in a program block called functions.

```
functions{
  real oneCpt(real time, real dose, real CL, real V){
    real k;
    k = CL / V;
    return dose * exp(-k * time) / V;
  }
}
```

- Such functions may be used just like built-in functions.
- It is also possible to define new probability distributions and random number generating functions.

User-defined function syntax

- The functions block must precede all other blocks.
- Function arguments and return types may be any Stan data type, e.g., int, int[ ], real, real[ ], vector, row_vector, matrix.
- Function arguments and return types are not specified with sizes or constraints, e.g.,
  - Two dimensional real array: y[,];
  - Vector: vector y;
- Void functions are permitted, i.e., functions with no return value.

Programming for pharmacometricians

Let's focus on 2 programming tasks:

- Pharmacokinetic compartmental models
- Dosing and observation event schedules

Stan demo: ME-2 single dose PK

- Two compartment model with first order absorption describing ME-2 plasma concentration on the i^{th} occasion in the j^{th} subject as a function of time, dose and body weight:

$$\begin{aligned}
\log\left(c_{ij}\right) &\sim N\left(\log\left(\hat{c}_{ij}\right), \sigma\right) \\
\hat{c}_{ij} &= f_{2cpt}\left(t_{ij}, D_j, CL_j, Q_j, V_{1j}, V_{2j}, k_{aj}\right) \\
\log\left(CL_j, Q_j, V_{1j}, V_{2j}, k_{aj}\right) & \\
&\sim N\left(\log\left(\widehat{CL}\left(\frac{bw_j}{70}\right)^{0.75}, \widehat{Q}\left(\frac{bw_j}{70}\right)^{0.75}, \widehat{V}_1\left(\frac{bw_j}{70}\right), \widehat{V}_2\left(\frac{bw_j}{70}\right), \widehat{k}_a\right), \Omega\right)
\end{aligned}$$

- Some possible weakly informative prior distributions:

$$\begin{aligned}
\widehat{CL} &\sim \text{half-}N\left(0, 25\right) \quad \widehat{Q} \sim \text{half-}N\left(0, 50\right) \quad \widehat{V}_1 \sim \text{half-}N\left(0, 100\right) \\
\widehat{V}_2 &\sim \text{half-}N\left(0, 200\right) \quad \widehat{k}_a \sim \text{half-}N\left(0, 5\right) \quad \sigma \sim \text{half-Cauchy}\left(0, 1\right) \\
\Omega &= \text{diag}\left(\omega\right) P \, \text{diag}\left(\omega\right) \\
\omega_i &\sim \text{half-Cauchy}\left(0, 1\right), i \in \{1, 2, 3, 4, 5\} \quad P \sim \text{LKJCorr}\left(1\right)
\end{aligned}$$

Stan demo: ME-2 single dose PK
Files

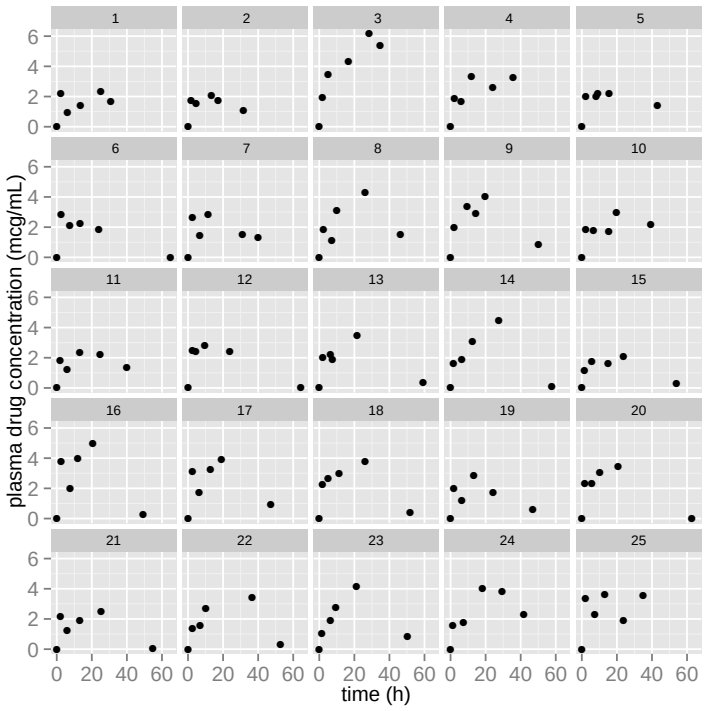
- Data: data/derived/fxa.data.csv
- Stan model: model/singleDosePK1.stan
- R script: script/singleDosePK1.R
- Version with vectorized function
  - Stan model: model/singleDosePK2.stan
  - R script: script/singleDosePK2.R

Recursive approach

Dealing with dosing and observation event schedules

- Data format: Time-ordered event records for each individual ala NONMEM
- For each event time calculate the amount in each compartment given the compartment amounts plus doses at the previous event time.
- This also allows for time-varying (piece-wise constant) parameter values.
- Same approach works for compartmental models described in terms of analytic or numerical solutions.

Stan demo: Multiple dose PK with sparse sampling



Sparsely sampled plasma drug concentrations resulting from administration of 200 mg of drug every 8 hours for 5 doses.

Stan demo: Multiple dose PK with sparse sampling

- One compartment model with first order absorption describing plasma drug concentration on the i^{th} occasion in the j^{th} subject as a function of time and dose:

$$\begin{aligned} \log\left(c_{ij}\right) &\sim N\left(\log\left(\hat{c}_{ij}\right), \sigma^2\right) \\ \hat{c}_{ij} &= f_{1cpt}\left(t_{ij}, D_j, \tau_j, CL_j, V_j, k_{aj}\right) \\ \log\left(CL_j, V_j, k_{aj}\right) &\sim N\left(\log\left(\widehat{CL}, \widehat{V}, \widehat{k}_a\right), \Omega\right) \end{aligned}$$

- Prior distributions: strongly informative for k_a; weakly informative for the remaining parameters

$$\begin{aligned} \widehat{CL} &\sim \text{half-}N\left(0, 20^2\right) \quad \widehat{V} \sim \text{half-}N\left(0, 100^2\right) \\ \widehat{k}_a &\sim \text{lognormal}\left(\log\left(0.45\right), 0.2^2\right) \quad \sigma \sim \text{half-Cauchy}\left(0, 2\right) \\ \Omega &= \text{diag}\left(\omega\right) P \,\text{diag}\left(\omega\right) \\ \omega_i &\sim \text{half-Cauchy}\left(0, 2\right), i \in \{1, 2\} \quad \omega_3 \sim \text{lognormal}\left(\log\left(0.25\right), 0.3^2\right) \\ P &\sim \text{LKJCorr}\left(1\right) \end{aligned}$$

Stan demo: Multiple dose PK with sparse sampling
Files

- Data: data/derived/prob_2fixed2.csv
- Stan model: model/multiDosePK1.stan
- R script: script/multiDosePK1.R

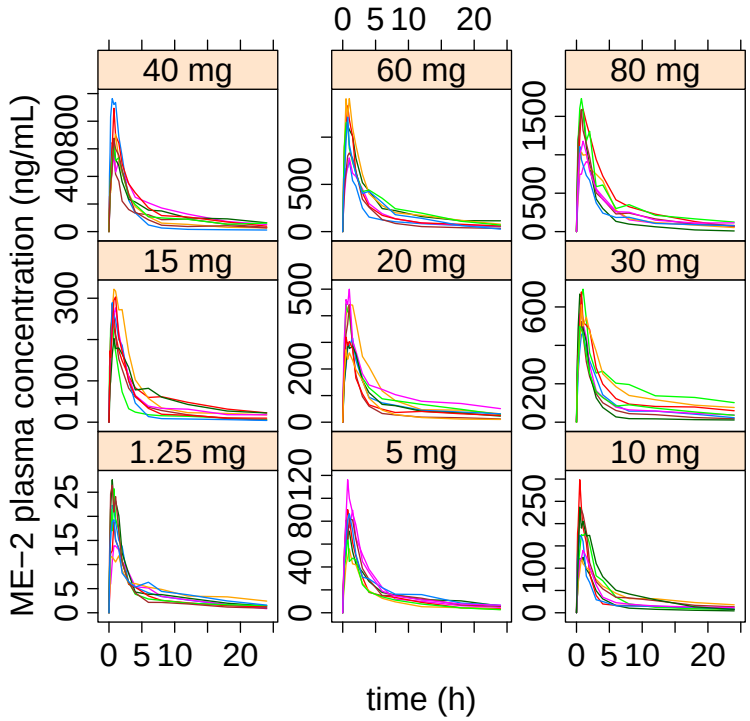
Hands-on session 3
Population PK of ME-2

- A phase IIa PoC trial of ME-2 for prevention of post-op VTEs has just been completed.
- One of your tasks is to do a pop PK analysis based on the accumulated ME-2 PK data (Phase I SD, Phase I MD & Phase IIa)
- Phase 1 single dose study in healthy volunteers described during hands-on sessions1 & 2
- Phase 1 multiple dose study in healthy volunteers
  - Parallel dose-escalation design
  - 8 subjects per dose arm
  - Placebo or ME-2 5, 10, 20, 40 or 80 mg bid (q12h) x 7 days
  - PK: plasma concentrations of parent drug
    - PK measured at 0, 0.083, 0.167, 0.25, 0.5, 0.75, 1, 1.5, 2, 3, 4, 6, 8, 12, 12.1, 12.2, 12.2, 12.5, 12.8, 13, 13.5, 14, 15, 16, 18, 20, 24, 36, 48, 60, 72, 84, 96, 108, 120, 132, 144, 156, 168, 168, 168, 168, 168, 169, 169, 170, 170, 171, 172, 174, 176, 180, 186 and 192 hours after the first dose.

Hands-on session 3
Population PK of ME-2

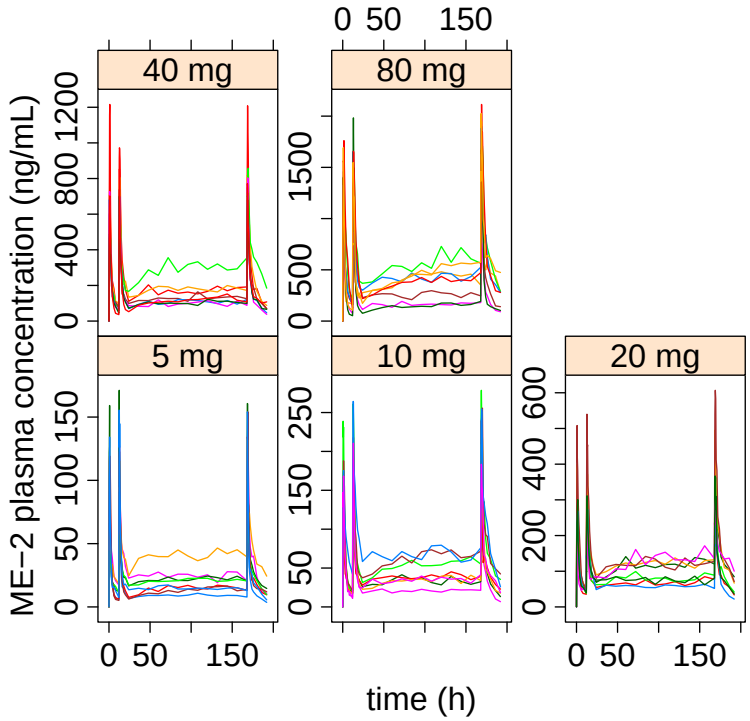
- A phase IIa PoC trial of ME-2 for prevention of post-op VTEs has just been completed.
- One of your tasks is to do a pop PK analysis based on the accumulated ME-2 PK data (Phase I SD, Phase I MD & Phase IIa)
- Phase IIa trial design:
  - Treatments
    - ME-2 20 mg bid (q12h) x 7 days
    - Enoxaparin 30 mg bid (q12h) x 7 days
  - 100 patients per treatment arm
  - Sparse ME-2 PK data (3-6 samples/patient)
    - LOQ = 10 ng/mL
- Available patient-specific covariates: weight, age, gender

Hands-on session 3



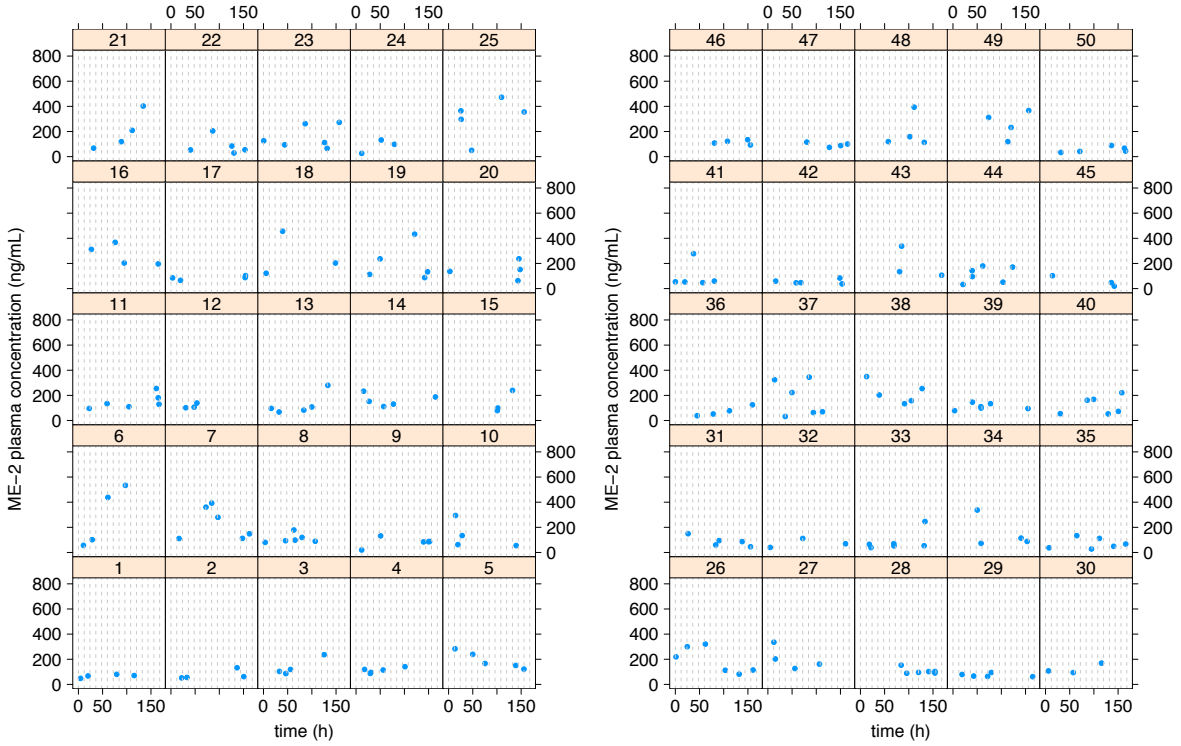
ME-2 PK data from Phase I SD trial

Hands-on session 3

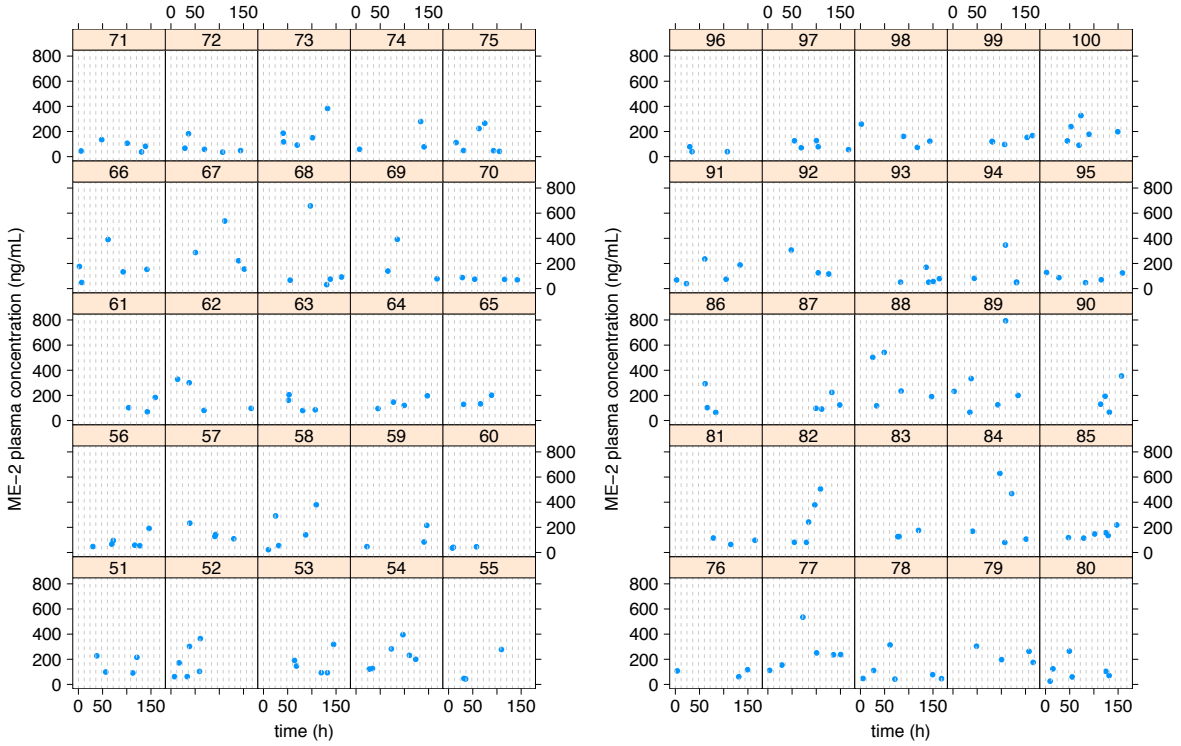


ME-2 PK data from Phase I MD trial

Hands-on session 3
ME-2 PK data from Phase IIa trial



Hands-on session 3
ME-2 PK data from Phase IIa trial



Hands-on session 3
Proposed base model

- Two compartment model with first order absorption describing ME-2 plasma concentration on the i^{th} occasion in the j^{th} subject as a function of time, dose and body weight:

$$\begin{aligned}
\log\left(c_{ij}\right) &\sim N\left(\log\left(\hat{c}_{ij}\right), \sigma\right) \\
\hat{c}_{ij} &= f_{2cpt}\left(t_{ij}, D_j, \tau_j, CL_j, Q_j, V_{1j}, V_{2j}, k_{aj}\right) \\
\log\left(CL_j, Q_j, V_{1j}, V_{2j}, k_{aj}\right) & \\
&\sim N\left(\log\left(\widehat{CL}\left(\frac{bw_j}{70}\right)^{0.75}, \widehat{Q}\left(\frac{bw_j}{70}\right)^{0.75}, \widehat{V}_1\left(\frac{bw_j}{70}\right), \widehat{V}_2\left(\frac{bw_j}{70}\right), \widehat{k}_a\right), \Omega\right)
\end{aligned}$$

- Some possible weakly informative prior distributions:

$$\begin{aligned}
\widehat{CL} &\sim \text{half-}N\left(0, 25\right) \quad \widehat{Q} \sim \text{half-}N\left(0, 50\right) \quad \widehat{V}_1 \sim \text{half-}N\left(0, 100\right) \\
\widehat{V}_2 &\sim \text{half-}N\left(0, 200\right) \quad \widehat{k}_a \sim \text{half-}N\left(0, 5\right) \quad \sigma \sim \text{half-Cauchy}\left(0, 1\right) \\
\Omega &= \text{diag}\left(\omega\right) P \,\text{diag}\left(\omega\right) \\
\omega_i &\sim \text{half-Cauchy}\left(0, 1\right), i \in \{1, 2, 3, 4, 5\} \quad P \sim \text{LKJCorr}\left(1\right)
\end{aligned}$$

Hands-on session 3
Files

- Data
  - ME-2 plasma concentration data (including all covariates) in NONMEM format: data/derived/fxaNONMEMData.csv
- Stan model: model/multiDoseME2PK1.stan
- R script: script/multiDoseME2PK1.R
- Non-centered parametrization
  - Stan model: model/multiDoseME2PK1Ncp.stan
  - R script: script/multiDoseME2PK1Ncp.R

Dealing with censored data in Stan

- The Stan language strongly separates the roles of data and parameters (random variables).
- This has consequences for handling of censored data.
  - Observed data are known values and thus true data.
  - Censored data are unknown values and thus random variables.
- The user must explicitly separate and declare observed and censored data:
  - Observed data in the data block.
  - Censored data in the parameters block.
    - Only need to declare censored data if you want to simulate predicted values.
    - Information such as censoring bounds and whether the data is censored is data that should be declared in the data block.
- And also specify different likelihoods for observed and censored data.

A little aside on Stan machinery

A Stan language model is essentially a set of instructions for Stan to calculate the log probability given the data and a set of parameter values.

- For (conditionally) independent data the total log probability is the sum of the individual log probabilities.
- In Stan

  ```
  y ~ normal(mu, sigma);
  ```

  is equivalent to

  ```
  target += normal_lpdf(y | mu, sigma);
  ```

- More generally you can use “target +=” to specify a user-defined log probability when none of the built-in distributions are appropriate.

Likelihoods for censored normally distributed data

Let's use "target +=" to specify the log probability function for censored data.

- Right censored data
  - Observation whose value exceeds some upper bound U.
  - The appropriate likelihood (probability) function is $\Pr(y > U)$ or 1 minus the normal CDF evaluated at U in the normal case.

```
target += normal_lccdf(U | mu, sigma);
```

- Left censored data
  - Observation whose value is less than some lower bound L, e.g., BQL data.
  - The appropriate likelihood (probability) function is $\Pr(y \leq L)$ or the normal CDF evaluated at L in the normal case.

```
target += normal_lcdf(L | mu, sigma);
```


Stan demo: PK modeling with BQL data

During our hands-on session 3 we excluded the BQL data.

Let's return to that example and appropriately incorporate the BQL data in the analysis.

- Data: data/derived/fxaNONMEMData.csv
- Stan model: model/multiDoseME2PK2.stan
- R script: script/multiDoseME2PK2.R

Some of what didn't we cover?

- Fundamentals of Bayesian inference and data analysis
- Model evaluation and comparison
- Use of informative prior distributions in clinical pharmacology applications
- Why Stan? Comparison with other general purpose modeling tools with Bayesian analysis capabilities
- Numerical solution of ODEs
- Torsten: Prototype library of PKPD functions for Stan
- Categorical, count and time-to-event data
- Population and trial simulations based on MCMC results
- User-defined probability distributions and likelihoods
- Diagnosing and remedying sampling problems encountered with Stan
- Optimizing Stan code
- ...

References I

[1] Andrew Gelman, John B Carlin, Hal S Stern, David B Dunson, Aki Vehtari, and Donald B Rubin.
Bayesian Data Analysis.
CRC Press, Boca Raton, FL, third edition, 2014.

[2] Christian P. Robert.
The Bayesian Choice: From Decision-Theoretic Foundations to Computational Implementation.
Springer, second edition, 2007.

[3] B. P. Carlin and T. A. Louis.
Bayesian Methods for Data Analysis.
Chapman & Hall/CRC, third edition, 2008.

[4] J.O. Berger.
Statistical Decision Theory and Bayesian Analysis.
Springer, second edition, 1993.

[5] D.J. Lunn, A. Thomas, N. Best, and D. Spiegelhalter.
Winbugs – a bayesian modelling framework: concepts, structure, and extensibility.
Statistics and Computing, 10:325–337, 2000.

References II

[6] D. J. Spiegelhalter, K. R. Abrams, and J. P. Myles.
Bayesian Approaches to Clinical Trials and Health-Care Evaluation.
Wiley, 2004.

[7] Christian P. Robert and George Casella.
Introducing Monte Carlo Methods with R.
Springer, 2010.

[8] Christian P. Robert and George Casella.
Monte Carlo Statistical Methods.
Springer, second edition, 2004.

[9] C. Dansirikul, R. G. Morris, S. E. Tett, and S. B. Duffull.
A bayesian approach for population pharmacokinetic modelling of sirolimus.
Br.J Clin.Pharmacol., 62(4):420–434, 2006.

[10] M. G. Dodds and P. Vicini.
Assessing convergence of markov chain monte carlo simulations in hierarchical bayesian models for population pharmacokinetics.
Ann.Biomed.Eng, 32(9):1300–1313, 2004.

References III

[11] A. Dokoumetzidis and L. Aarons.
Propagation of population pharmacokinetic information using a bayesian approach: comparison with meta-analysis.
J Pharmacokinet.Pharmacodyn., 32(3-4):401–418, 2005.

[12] L. E. Friberg, G. K. Isbister, L. P. Hackett, and S. B. Duffull.
The population pharmacokinetics of citalopram after deliberate self-poisoning: a bayesian approach.
J Pharmacokinet Pharmacodyn, 32(3-4):571–605, 2005.

[13] I. Gueorguieva, L. Aarons, and M. Rowland.
Diazepam pharamacokinetics from preclinical to phase i using a bayesian population physiologically based pharmacokinetic model with informative prior distributions in winbugs.
J Pharmacokinet.Pharmacodyn., 33(5):571–594, 2006.

[14] Steven J Kathman, Daphne H Williams, Jeffrey P Hodge, and Mohammed Dar.
A bayesian population pk-pd model for ispinesib/docetaxel combination-induced myelosuppression.
Cancer Chemother Pharmacol, 63(3):469–476, 2009 Feb.

[15] D. J. Lunn, N. Best, A. Thomas, J. Wakefield, and D. Spiegelhalter.
Bayesian analysis of population pk/pd models: general concepts and software.
Journal of Pharmacokinetics and Pharmacodynamics, 29(3):271–307, 2002.

References IV

[16] S. Mu and T. M. Ludden.
Estimation of population pharmacokinetic parameters in the presence of non-compliance.
Journal of Pharmacokinetics and Pharmacodynamics, 30(1):53–81, 2003.

[17] M. K. Smith, I. Jones, M. F. Morris, A. P. Grieve, and K. Tan.
Implementation of a bayesian adaptive design in a proof of concept study.
Pharm Stat, 5(1):39–50, 2006.

[18] M. K. Smith and S. Marshall.
A bayesian design and analysis for dose-response using informative prior information.
J Biopharm Stat, 16(5):695–709, 2006.

[19] Matthew D Hoffman and Andrew Gelman.
The no-U-turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo.
The Journal of Machine Learning Research, 15(1):1593–1623, 2014.

[20] Radford M. Neal.
MCMC using hamiltonian dynamics.
In Steve Brooks, Andrew Gelman, Galin L. Jones, and Xiao-Li Meng, editors, *Handbook of Markov Chain Monte Carlo*, chapter 5, pages 113–162. Chapman & Hall/CRC, Boca Raton, FL, 2011.

References V

[21] Stan Modeling Language Users Guide and Reference Manual, Version 2.8.0, 2015.
`http://mc-stan.org/`.

[22] RStan: the R interface to Stan, Version 2.7.0, 2015.
`http://mc-stan.org/rstan.html`.

[23] Andrew Gelman.
Prior distributions for variance parameters in hierarchical models.
Bayesian Analysis, 1:515–533, 2006.