```
# Other needed packages
library(xtable)
library(ggplot2)
library(shiny)
library(shinydashboard)

##
## Attaching package:  'shinydashboard'
## The following object is masked from 'package:graphics':
##
##    box

library(rhandsontable)
```

Adverse medical consequences can arise from both antibiotic underdosing and overdosing of hospitalized patients. In the former case, patients receive an insufficient amount of antibiotic to effectively combat infection. In the latter, an overabundance of antibiotic may increase the risk of toxicity. However, the pharmacokinetics and pharmacodynamics of common antibiotics, such as ciprofloxacin and piperacillin, can exhibit substantial variability among hospitalized patients due to patient characteristics (e.g., body mass) or complications (e.g., organ failure) [**?**, **?**]. As a result, simple algorithmic dosing strategies (e.g., based on creatinine alone) can fail to ensure an appropriate level of antibiotic exposure.

For an individual patient, pharmacokinetic uncertianty can be reduced by measuring drug consentration in the blood over time. These data can be used to estimate current an past drug exposure, predict future drug exposure under alternative dosing strategies, and adjust the dose accordingly. The methods and software presented herein were designed to implement this process using prior information about pharmacokinetic heterogeneity in the target population and, if available for the individual, measurments of drug concentration in the blood over time. In order to impact clinical decision making, these tasks must be accomplished in real-time and account for (often substantial) statistical uncertainty.

We implement Bayesian methods using pharmacokinetic data from a prior study [cite] to provide posterior estimates of individual target attainment, measured as the fraction of the dosing period spent above the necessary blood concentration threshold. The fact that patients tend to vary widely in their response to treatment indicates that measures of uncertainty are critical to understanding response to treatment at both the population and patient-specific level. In order for the implementation of such procedures to be truly beneficial to individuals, it is necessary to have a method of assessing and predicting patient response in real time. This introduces the issue of selecting a statistical approximation technique that is both accurate in its quantification of uncertainty while also being computationally feasible to implement via a web application. Our statistical methodology is paired with a web application that implements these techniques and makes them accessible to, for example, a physician in real time. Assessments of the performance and computational efficiency are presented for various approximation methods.

1 Methods

1.1 Two-compartment model

The two-compartment pharmacokinetic model is expressed as a system of two ordinary differential equations as follows, where m_1 and m_2 are the masses of drug in the central and peripheral compartments, respectively.

$$\begin{aligned}\frac{dm_1}{dt} &= -k_{10}m_1 - k_{12}m_1 + k_{21}m_2 + k_R \\ \frac{dm_2}{dt} &= \qquad\qquad + k_{12}m_1 - k_{21}m_2\end{aligned}$$

Parameter	Units	Description
k_{10}	h^{-1}	Elimination rate from central compartment
k_{12}	h^{-1}	Distribution rate from central to peripheral compartment
k_{21}	h^{-1}	Distribution rate from peripheral to central compartment
k_R	$\text{g}\cdot\text{h}^{-1}$	Infusion rate into central compartment
v_1	L	Volume of central compartment

Table 1: Two-compartment model parameter units (SI) and descriptions.

The concentration of drug in the central compartment is given by $c_1 = m_1/v_1$, where v_1 is the volume of the central compartment. The parameters k_{10}, k_{12}, k_{21}, and k_R are described in Table **??**

1.2 Bayes prediction model

Concentration measurements are modeled using a nonlinear regression method with additive error as follows:

$$c_{ij} = \eta_i(t_{ij}, \theta_i) + \epsilon_{ij}$$

In this expression, c_{ij} is the measured concentration for subject $i = 1 \dots n$ at time t_{ij} for $j = 1 \dots m_i$, $\eta_i(t_{ij}, \theta_i)$ is the two-compartment model solution for subject i at time t_{ij} given parameters $\theta_i = [k_{10i}, k_{12i}, k_{21i}, v_{1i}]$, and ϵ_{ij} represents i.i.d. random error with mean zero. By specifying the random error density function, say $f(\epsilon_{ij}, \sigma)$, the subject-specific likelihood function is

$$L_i(\theta_i, \sigma) = \prod_{j=1}^{m_i} f(c_{ij} - \eta_i(t_{ij}, \theta_i), \sigma).$$

Thus, given a prior distribution $\pi_0(\theta_i)$, the subject-specific posterior is proportional to $\pi_i(\theta_i, \sigma) \propto \pi_0(\theta_i, \sigma) L_i(\theta_i, \sigma)$.

In the current context, f is taken as the normal density function with mean zero and standard deviation σ, and the prior distribution is generated to satisfy the following:

$$[\log \theta_i] \sim N_4(\mu_0, \Sigma_0) \tag{1}$$

$$[\log \sigma] \sim N_1(m_0, s_0) \tag{2}$$

where $N_4(\mu_0, \Sigma_0)$ represents the 4-variate normal distribution with mean μ_0 and covariance matrix Σ_0, and $N_1(m_0, s_0)$ represents the univariate normal distribution with mean m_0 and variance s_0. The prior distribution for the PK parameters represents our prior knowledge about the PK heterogeneity for a particular drug in a target population. Thus, the values of the prior hyperparameters should be carefully selected for the task at hand. In the current context, and by default, the hyperparameters are specified to correspond with estimates that arose from a study of piperacillin pharmacokinetics in a hospitalized, critically ill population. The particular values are listed in an appendix.

Due to the nonlinearity of the two-compartment model, the posterior distribution does not take a familiar form, and posterior summaries must be approximated. In particular, we sought to compute 95% credible bands for subject-specific concentration-time curves, and for a specific summary of the curve: the length of time in which the concentration exceeds a specified value. Monte Carlo techniques are often used to approximate these quantities. However, because these posterior summaries are presented in a web application in real time, we sought alternatives that were less computationally intensive and deterministic. We consider several methods as described in the following sections. [REWORD THIS LAST SENTENCE - all one method, just looking at different types of intervals]

1.2.1 Laplace approximation

We first considered a method that makes use of two approximations. The first is a Laplace approximation to the subject-specific posterior density, and the second is a first order Taylor approximation of the target summary (i.e., the 'delta method'). The Laplace approximation is given as follows

$$\pi_i(\log \theta_i, \log \sigma) \approx N([\log \hat{\theta}_i, \log \hat{\sigma}], [-H]^{-1})$$

where $[\log\hat{\theta}_i, \log\hat{\sigma}]$ is the posterior mode and $H_{\hat{\theta}_i}$ is the posterior Hessian with respect to $[\log\theta_i, \log\sigma]$ evaluated at the posterior mode. The second approximation makes use of the delta method, such that a posterior functional $h(\log\theta_i)$ has an approximate normal distribution. In the present context, $h(\log\theta_i)$ represents the logit of the fraction of the dosing period in which the concentration of drug exceeds a spedified value. The first-order Taylor expansion of $h(\log\theta_i)$ is

$$h(\log\theta_i) \approx h(\log\hat{\theta}_i) + G^T(\log\theta_i - \log\hat{\theta}_i),$$

where G is the gradient of $h(\log\theta_i)$ with respect to $[\log\theta_i, \log\sigma]$ evaluated at the posterior mode. Thus, given that $[\log\theta_i, \log\sigma]$ has an approximate normal distribution, the same is true for $h(\log\theta_i)$:

$$h(\log\theta_i) \sim N(h(\log\hat{\theta}_i), G^T[-H]^{-1}G).$$

An approximate $(1-\alpha)\cdot 100\%$ credible interval for $h(\log\theta_i)$ is thus given by the $\alpha/2$ and $(1-\alpha)/2$ quantiles of the approximate posterior distribution for $h(\log\theta)$. This method is computationally elegant, since the posterior Hessian and posterior mode can be computed simultaneously by most optimization software routines (e.g., the R function 'optim'). Indeed, this method usually has the smallest computational burden (i.e., number of likelihood evaluations) among the three methods considered here.

1.3 Evaluation of approximation method

We compared the performance of three different interval types: logit-transformed, probit-transformed, and untransformed (further referred to as the linear interval). We obtain 1,000 draws ("patients") from the prior distribution of pharmacokinetic parameters. For each patient, we simulated 95% confidence intervals for each of the three types using the Metropolis MCMC algorithm.

The posterior coverage of the MIC statistic 95% confidence interval was estimated as follows:

One thousand samples were drawn from the prior distribution of PK parameters. For each set of prior parameters, 6 observations were simulated using the concentration predicted by the first 4 PK parameters and the standard error term corresponding to the 5th prior parameter. All observations were simulated during the final (5th) dosing period at 32, 32.5, 33, 34, 36, and 38 hours.

For each of the 50 sets of prior parameters and simulated data:

- A Laplace approximation to the posterior density was calculated and the delta method used to estimate the standard errors of the MIC statistic. A 95% asymptotic normal confidence interval was used with the delta method standard errors to estimate upper and lower bounds.
- 3,000 samples from the posterior distribution of PK parameters were taken using a Metropolis MCMC algorithm with 2,000 warm-up iterations. The proportion of the 3,000 posterior samples falling within the upper and lower bounds was used to estimate the proportion of the posterior distribution of the MIC statistic falling within the confidence interval.

2 Results

2.1 Evaluation of approximation method

```
load("/Users/hannahweeks/Desktop/all_cov.rdta")

# Median (IQR) for each
miqr <- purrr::partial(quantile, probs = c(.25, .5, .75), na.rm = T)

miqr_logit <- miqr(coverage_res$coverage_logit)
miqr_probit <- miqr(coverage_res$coverage_probit)
miqr_frac <- miqr(coverage_res$coverage_frac)

spf <- purrr::partial(sprintf, fmt = "%.1f")
```

replace On 1,000 samples we calculated a median posterior coverage of ‘r spf(miqr_frac[2]*100)‘% with an IQR ‘r spf(miqr_frac[1]*100)‘% to ‘r spf(miqr_frac[3]*100)‘%.

Confidence intervals for the MIC statstic were also considered on the logit and probit scales in order to restrict the bounds to the interval [0,1]; however, as these transformations resulted in slightly lower and more variable posterior coverage, the original scale was used. For the logit-transformed interval, the Bayesian coverage had a median (IQR) of ‘r spf(miqr_logit[2]*100)‘% (‘r spf(miqr_logit[1]*100)‘%, ‘r spf(miqr_logit[3]*100)‘%) and the probit-transformed interval had ‘r spf(miqr_probit[2]*100)‘% (‘r spf(miqr_probit[1]*100)‘%, ‘r spf(miqr_probit[3]*100)‘%)

```
ncov_logit$type <- "logit"
ncov_probit$type <- "probit"
ncov_frac$type <- "linear"

ncov_res <- rbind(ncov_logit, ncov_probit, ncov_frac)
# make sure column names are correct and that class == data.frame

ncov_long <- reshape(ncov_res, varying = c("lower", "upper"), v.names = "ncov",
                     timevar = "side", times = c("lower", "upper"),
                     direction = "long")
rownames(ncov_long) <- NULL
ncov_long <- subset(ncov_long, select = c("ncov", "side", "type"), drop = TRUE)

cbColors <- c("#009E73", "#0072B2", "#D55E00")
```

- used MCMC as gold-standard
- quantified posterior probability contained within approximate intervals for $logit^{-1}(h(log\theta_i))$
- quantified number of posterior evaluations
- quantified computing time
- note: make sure to uncomment line below and set up the bibtex doc

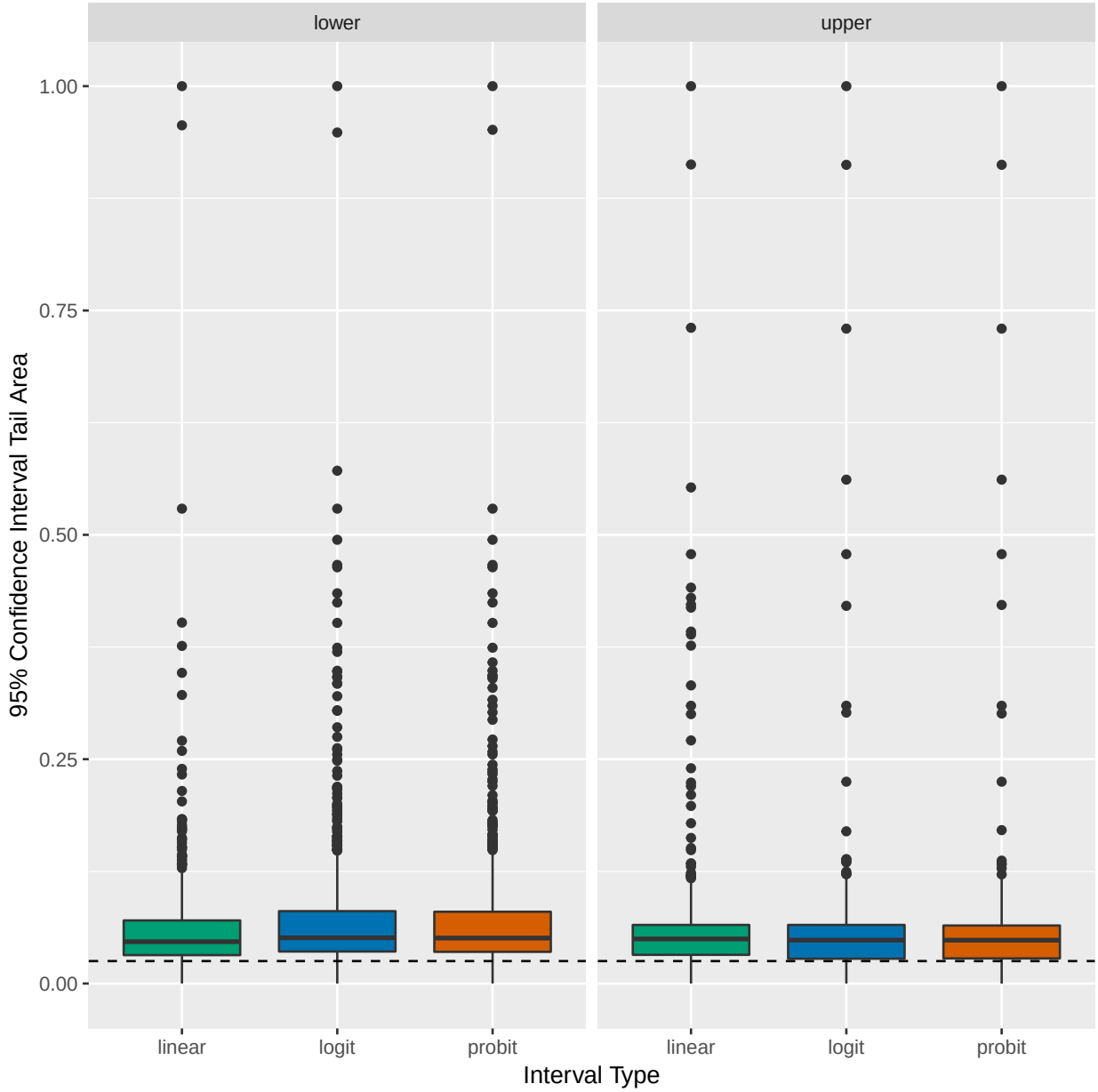
3 Example

- Generate ivt data frame
- obtain raw mic estimate
- fit pkm model
- plot estimates
- update the model and re-plot estimates
- shiny app demo

```
# devtools::install_github("hlweeks/pkpredict")
library(pkpredict)
```

note: for key functions, list the parameters that can be manipulated and what they do, default values, etc

Figure 1: Comparison of confidence intervals for estimate of target attainment



Plotted values represent the "non-coverage" in each tail region outside the 95% confidence interval.

3.1 Sample infusion schedule and concentation data

package functions: `ivt_toList`

We have a vector of start times for five doses in hours since first infusion, and the duration of each dose. In this case, the duration is the same for each dose: half an hour. The rate of infusion is 6 g/h for all administered doses.

```
# Time in hours since first infusion
start_times <- c(0, 8, 16, 24, 32)
# Duration of each infusion in hours
duration <- 0.5
# Rate in g/h
rate_of_infusion <- rep(6, 5)

ivt_d <- ivt_toList(begin = start_times, dur = duration, rate = rate_of_infusion)
```

Table 2: Sample Infusion Schedule

Begin (h)	End (h)	Infusion Rate (g/h)
0.0	0.5	6.0
8.0	8.5	6.0
16.0	16.5	6.0
24.0	24.5	6.0
32.0	32.5	6.0

We also have sample data from three blood draws, with information on the concentration of piperacillin in the blood in μg/mL at the time at which the blood was drawn, again in hours since first infusion.

```
# Time is in hours since first infusion
# Concentration is in mcg/ml (or equivalently, mg/L)
dat_d <- data.frame("time" = c(1, 4, 40),
                    "con" = c(82.7, 80.4, 60),
                    check.names = FALSE)
```

Table 3: Sample Data (observed from blood draws)

Time (h)	Concentration (mcg/mL)
1.0	82.7
4.0	80.4
40.0	60.0

3.2 MIC estimates

package functions: `mic_stat`

We first obtain a prior estimate of target attainment (**make sure this is defined above**) based on the infusion schedule outlined above. By default, the `mic_stat` function calculates the $fT > 4 \times MIC$ (notation from paper Matt sent - make sure this notation is introduced above).

```
# Threshold is in mcg/mL (same scale as observed data)
# By default, considering time through 12 hours after end of the final dose

# Prior estimate
```

```
prior_ft.mic <- mic_stat(ivt = ivt_d, th = 64)
prior_ft.mic

## $ftmic
## [1] 0.4782617
##
## $conf.int
## [1] 0.09186105 0.89255457
##
## attr(,"class")
## [1] "mic"  "list"
```

This prior estimate of target attainment represents how we would expect a typical patient to respond to treatment over the course of this infusion schedule, in lieu of any blood concentration measurements. We can incorporate the observed blood draws and obtain an individualized estimate of target attainment based on the patient's data.

```
# Posterior estimate after observed measurements
post_ft.mic <- mic_stat(ivt = ivt_d, dat = dat_d, th = 64)
post_ft.mic

## $ftmic
## [1] 0.7352296
##
## $conf.int
## [1] 0.4755578 0.8947764
##
## attr(,"class")
## [1] "mic"  "list"
```

3.3 PK Model

package functions: `pkm`

Fitting the model here

```
pk.fit <- pkm(formula = con ~ time, data = dat_d,
              ivt = ivt_d, thres = 64)
pk.fit$ftmic

## $ftmic
## [1] 0.7352296
##
## $conf.int
## [1] 0.4755578 0.8947764
##
## attr(,"class")
## [1] "mic"  "list"
```

The model MIC statistic is exactly that produced by the `mic_stat` function. Below we extract the predicted values, either directly from the model object (returns values calculated at infusion and blood draw times), or at user-specified times with the `predict.pkm` method.

```
pred.df <- pk.fit$fitted.values
# all.equal(pred.df£conc == predict(pk.fit)) # need to fix predict.pkm
```

Table 4: Predicted Values

Time Since First Infusion (h)	Concentration (mcg/mL)	Standard Error
0.00	0.20	0.11
0.50	97.34	0.11
1.00	91.58	0.10
4.00	63.51	0.08
8.00	38.99	0.10
8.50	134.03	0.09
16.00	53.68	0.13
16.50	147.85	0.08
24.00	59.22	0.14
24.50	153.07	0.08
32.00	61.32	0.15
32.50	155.03	0.08
40.00	62.11	0.16

3.4 Model Methods

package functions: `plot.pkm`, `predict.pkm`, `update.pkm`, `confint.pkm`

3.5 Shiny GUI

An interactive implementation of the methods presented here can be launched within the package by calling `shiny_pkm`. This function procudes a web application, created using the *shiny* package in R. In the application, users can enter information about a patient's infusion schedule and and concentration data collected from blood draws. By default, the application uses the infusion schedule in the example above.

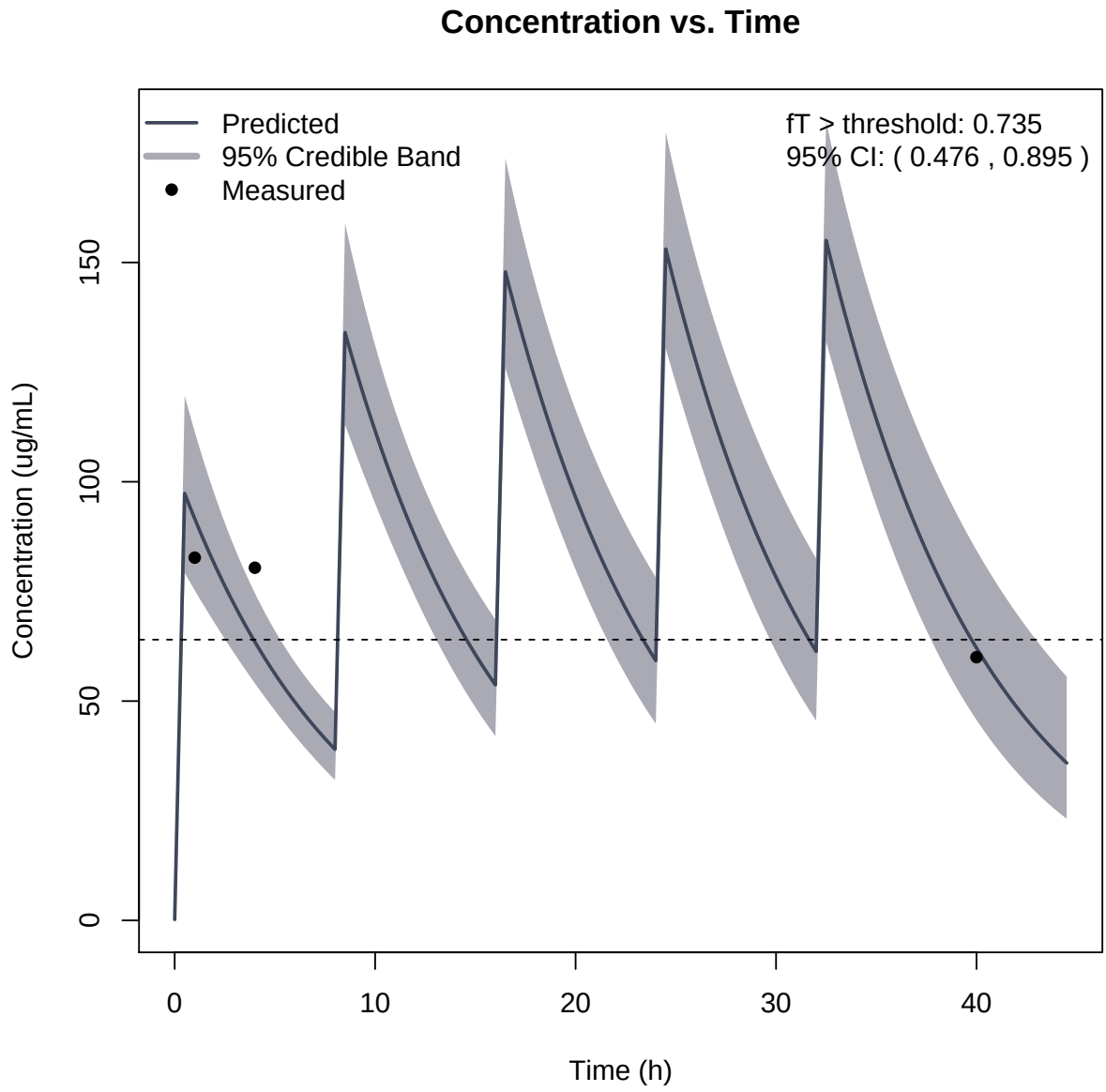
```
shiny_pkm()
```

4 Discussion

- Recommendation about how to use this software: tinkering to get an idea of how patient is responding. After tinkering should do final round of MCMC sampling for most accurate estimates
- How it might be used in clinical settings
- Future directions
- How to modify the functions to adapt to other drugs
- extension to 3 compartment model or nonlinear kinetics (?)
- clincal complications - maybe get Bill's input on how to address the fact that we are dealing with AKI patients and piperacillin is partially renally excreted (caution of using this methodology)

5 Appendix: Prior hyperparameters for piperacillin among critically ill

```
plot(pk.fit)
```



Parameter	Value
μ_0	
Σ_0	
α_0	
β_0	

Table 5: Prior hyperparameters for piperacillin in a population of hospitalized, critically ill patients.