Three perspectives on modelling for ecological risk assessment

A toy example with a simple one-compartment toxicokinetic model

Contributing authors

2023-05-21

Abstract

We provide here a toy example based on the use of a simple one-compartment toxicokinetic model to describe the bioaccumulation of chemical substances within the whole body of living organisms. From a simple ODE model, we will illustrate : (P1) how to simulate both accumulation and depuration phases under constant exposure and to compare model outputs to observed data; (P2) how to fit such a model on data without using any prior information on the model (Frequentist point of view); (3) how to benefit of prior information in combination with knowledge in data to update the calibration results (Bayesian point of view).

Introduction

Perform calculations under the three perspectives as described within the main document

*** to be developed ***

Case study

Raw data, extracted from (Ashauer et al. 2010), concern the freshwater arthropod *Gammarus pulex* exposed to the organo-phosphate insecticide Malathion (CAS number 121-75-5), which acts as an acetyl-cholinesterase inhibitor. Raw data are plotted in Figure @ref(fig:rawdata).

G. pulex organisms were collected in a small headwater stream near Zürich in Switzerland, and acclimatized during about five days under conditions similar to those of the bioaccumulation tests. Organisms were fed *ad libitum* with horse chestnut leaf disks inoculated with *Cladosporium herbarum*. The average wet weight was 32.37 mg ($\pm$16.88 mg SD, $n = 1040$). The sex ratio was assumed equal to 0.5.

Toxicokinetic experiments consisted of a 24-h uptake phase during which the organisms were exposed to the compound and a 6-d elimination phase starting with the transfer of organisms to fresh media without compound. A total of 20 G. pulex organisms were placed in each of the five replicate treated beakers and the solvent control beaker

*** to be developed ***

```
# Load the data set
df <- read.table("data.txt", header = TRUE, sep = "")
```

```
# Summarize the data set
# End of the accumulation phase
tc <- 1
# Number of time points
ntp <- length(unique(df$time))
```

```r
# Plot the raw data
ggplot(data = df, aes(x = time, y = conc)) +
  geom_point() +
  xlab("Time (hours)") +
  ylab("Internal measured concentration (picomol/g wet weight)") +
  geom_vline(xintercept = 1, linetype="dashed") +
  ylim(0, 120)
```

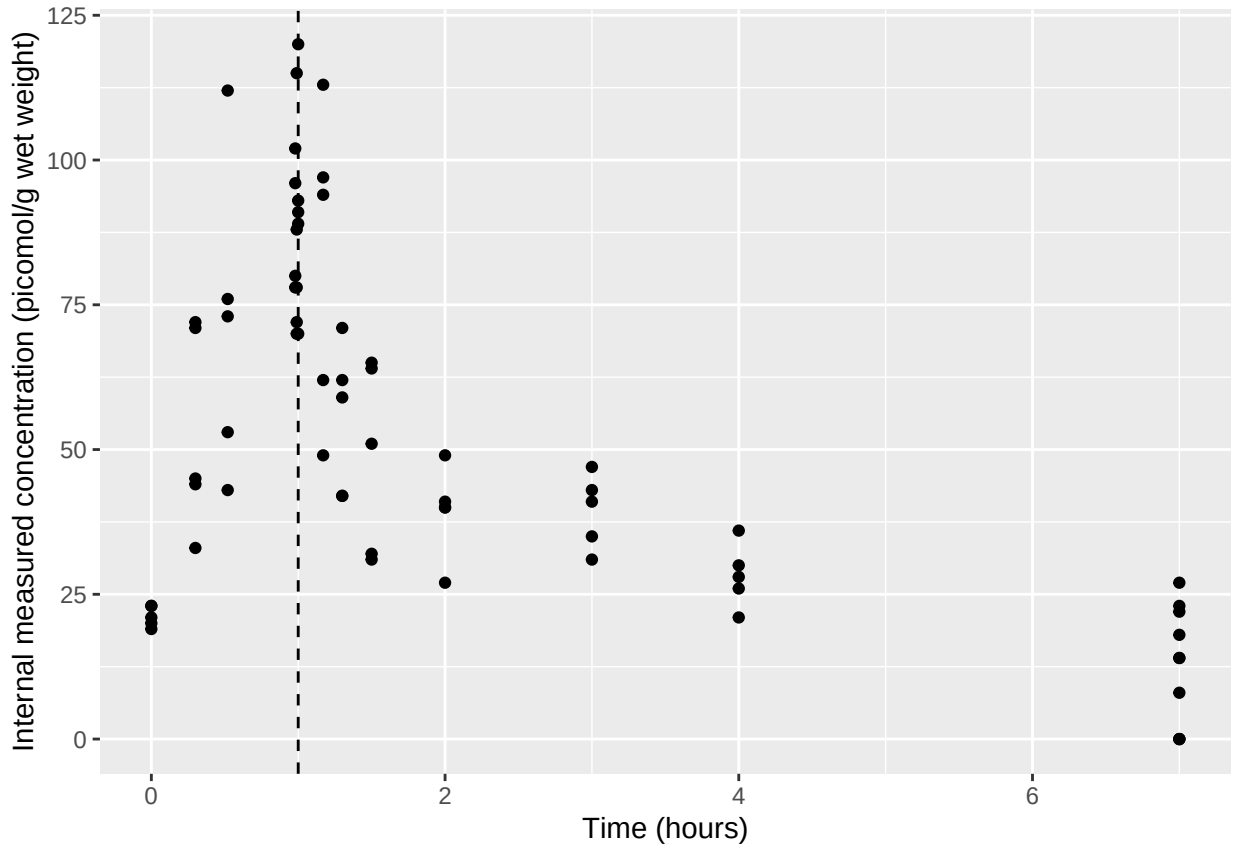


Figure 1: Raw data vizualisation (black dots). The vertical dashed line stands for the end of the accumulation phase ($t_c = 1$ day). Exposure concentration equals 1.485 picomol/ml.

A toxicokinetic (TK) model simply describing bioaccumulation of chemical substances within the whole body of living organisms is based on a set of two differential equations standing for the deterministic part (Charles, Ratier, and Lopes 2021):

$$\begin{cases} \dfrac{dC}{dt}(t) = k_u \times C_w - k_e \times C(t) \quad \forall\, 0 \leqslant t \leqslant t_c & \text{(1a)} \\ \dfrac{dC}{dt}(t) = -k_e \times C(t) \quad \forall\, t > t_c & \text{(1b)} \end{cases}$$

where t_c stands for the duration of the accumulation phase (namely the end of the exposure period, before organisms are transferred into a clean medium). Quantity C_w stands for the exposure concentration in water, while variable $C(t)$ corresponds to the internal concentration within the whole body of organisms over time t. Parameters k_u and k_e are the uptake and the elimination rates, respectively.

Given that state variables are concentrations, an appropriate stochastic part to describe the variability of the data around the mean tendency is the Gaussian distribution, so that:

$$C_{obs}(t_i) \sim \mathcal{N}(C(t_i), \sigma) \tag{2}$$

where $C_{obs}(t_i)$ are the measured internal concentrations at time point t_i, $\forall\, i \in 1, n$, with n the total number of time points. Parameter σ stands for the standard deviation of the normal distribution.

Perspective 1 -

Under perspective 1, the idea is to simulate the toxicokinetic process with model equations (1) and to compare the simulated curve with observed data. To do so, we can use k_u and k_e parameter estimates as provided in Table 1 of (Ashauer et al. 2010). However, we can choose one of the two estimates: the one from the Marquardt fit, or the one from the MCMC fit. Note that these estimates are provided as aggregated values (mean $\pm$ standard deviation), and that the estimation of parameter σ is missing.

Then, two simulation options can be considered: option 1, using only mean values for the simulations; option 2, accounting for the uncertainty. This latter raises the question of the probability distribution to consider. Because estimates are provided as means and standard deviations, this invites us to consider a normal distribution for both k_u and k_e parameters.

Below is a summary table with estimates from (Ashauer et al. 2010) corresponding to the raw data we are dealing with in this document:

```
tab1 <- read.table("table1.txt", header = TRUE)
kable(tab1)
```

Method	ku_mean	ku_sd	ke_mean	ke_sd	BCF_mean	BCF_low	BCF_up	t95
Marquardt	92.58	5.1	0.81	0.083	114.3	NA	NA	3.70
MCMC	93.40	10.2	0.82	0.173	115.3	144.4	144.4	3.64

Simulated model

Because the exposure concentration is here considered as constant, equations (1) can be analytically integrated as written below. See (Charles, Ratier, and Lopes 2021) for details.

$$\begin{cases} C(t) = \dfrac{k_u}{k_e} C_w \left(1 - e^{-k_e t}\right) \quad \forall\, 0 \leq t \leq t_c & \text{(3a)} \\ C(t) = \dfrac{k_u}{k_e} C_w \left(e^{k_e (t_c - t)} - e^{-k_e t}\right) \quad \forall\, t > t_c & \text{(3b)} \end{cases}$$

```r
# Create a function to simulate bioaccumulation
# along both accumulation and depuration phases
bioacc <- function(parameters, expw, tc, tmax){
  tacc <- seq(0, tc, length.out = 100)
  Cacc <- parameters[1]*expw*(1 - exp(-parameters[2]*tacc))/parameters[2]
  tdep <- seq(tc, tmax, length.out = 200)
  Cdep <- parameters[1]*expw*(exp(parameters[2]*(tc - tdep)) - exp(-parameters[2]*tdep))/parameters[2]
  result <- data.frame(time = c(tacc, tdep),
                       conc = c(Cacc, Cdep))
  return(result)
}
```

Persp.1 - option 1

```r
# Assign input parameter values
# Use the mean of Marquardt and MCMC fits
ku.mean <- mean(tab1$ku_mean)
ke.mean <- mean(tab1$ke_mean)
parameters <- c(ku.mean, ke.mean)
expw <- unique(df$expw)
tmax <- max(df$time) # final time of the experiment
```

```r
# Launch simulations
simu.mean <- bioacc(parameters, expw, tc, tmax)
```

```r
# Plot simulations
ymax <- max(unique(df$conc)) # maximal observed concentrations
ggplot(simu.mean, aes(time, conc)) +
  geom_line(col = "orange", size = 1.5) +
  xlab("Time (hours)") +
  ylab("Internal concentrations") +
  geom_vline(xintercept = 1, linetype="dashed") +
  coord_cartesian(xlim = c(0, tmax), ylim = c(0, ymax)) +
  # Add observed data
  geom_point(data = df, aes(time, conc)) +
  theme_bw() +
  ylim(0, 120)
```

Comparison between data and simulations

```r
# Time points of the accumulation phase
tacc <- df %>%
  select(time) %>%
  filter(time <= tc)
# Predicted values of the accumulation phase
tmp <- parameters[1] * expw * (1 - exp(- parameters[2] * tacc)) / parameters[2]
predacc <- tmp %>%
  transmute(pred = time) %>%
  mutate(phase = "accumulation")
# Time points of the depuration phase
tdep <- df %>%
  select(time) %>%
  filter(time > tc)
```

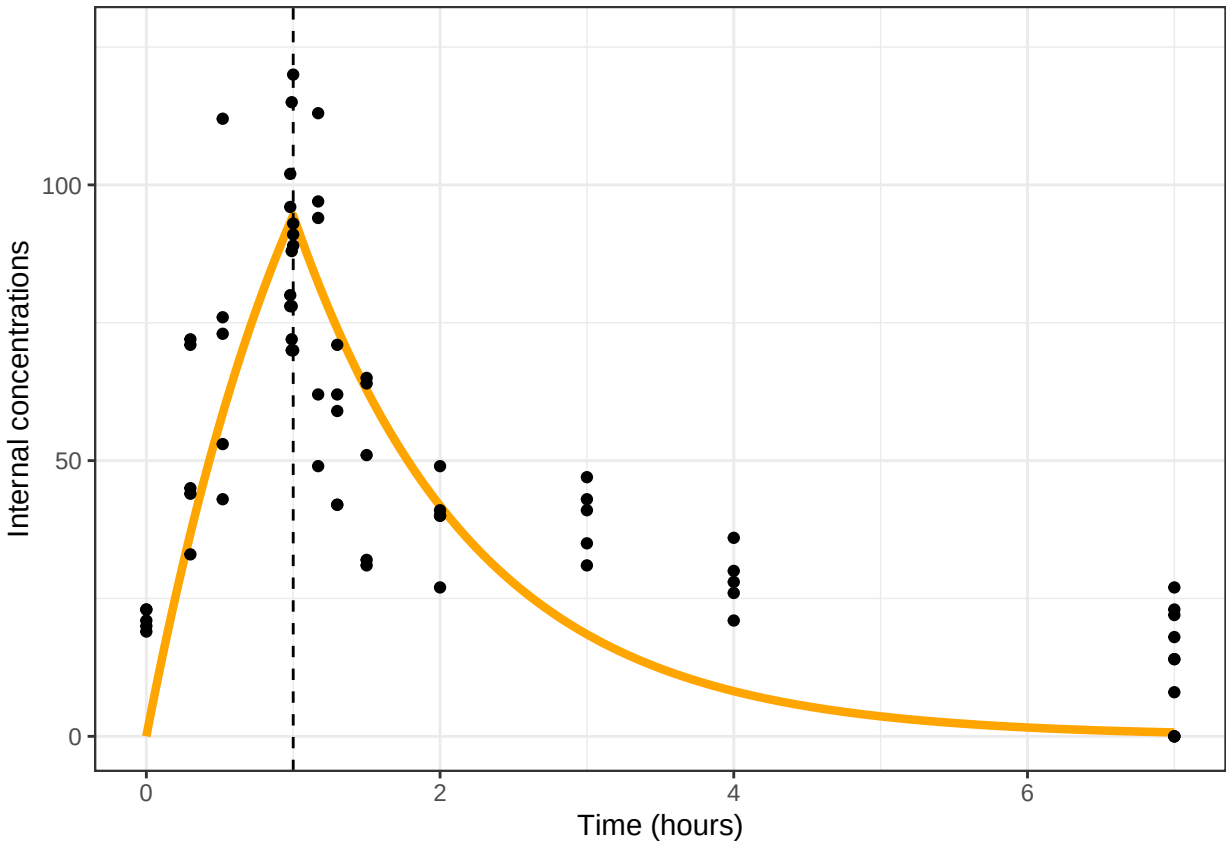


Figure 2: Simulation of a one-comparment toxicokinetic model with parameters values got from medians in Table 1 of (Ashauer et al. 2010) (solid orange line). Black dots are observed data.

```r
# Predicted values of the depuration phase
tmp <- parameters[1] * expw * (exp(parameters[2] * (tc - tdep)) - exp(- parameters[2] * tdep))/parameter
preddep <- tmp %>%
  transmute(pred = time) %>%
  mutate(phase = "depuration")
df <- cbind(df, rbind(predacc, preddep))
```

```r
ggplot(data = df, aes(x = conc, y = pred, colour = phase)) +
  geom_point() +
  lims(x = c(min(df$conc), max(df$conc)),
       y = c(min(df$conc), max(df$conc))) +
  xlab("Measured internal concentrations (picomol/g wet weight)") +
  ylab("Predicted internal concentrations (picomol/g wet weight)") +
  geom_abline(slope = 1, intercept = 0) +
  stat_cor(aes(label = paste(..rr.label.., ..p.label.., sep = "~`;`~")), color = "black", label.x = 0, l
```

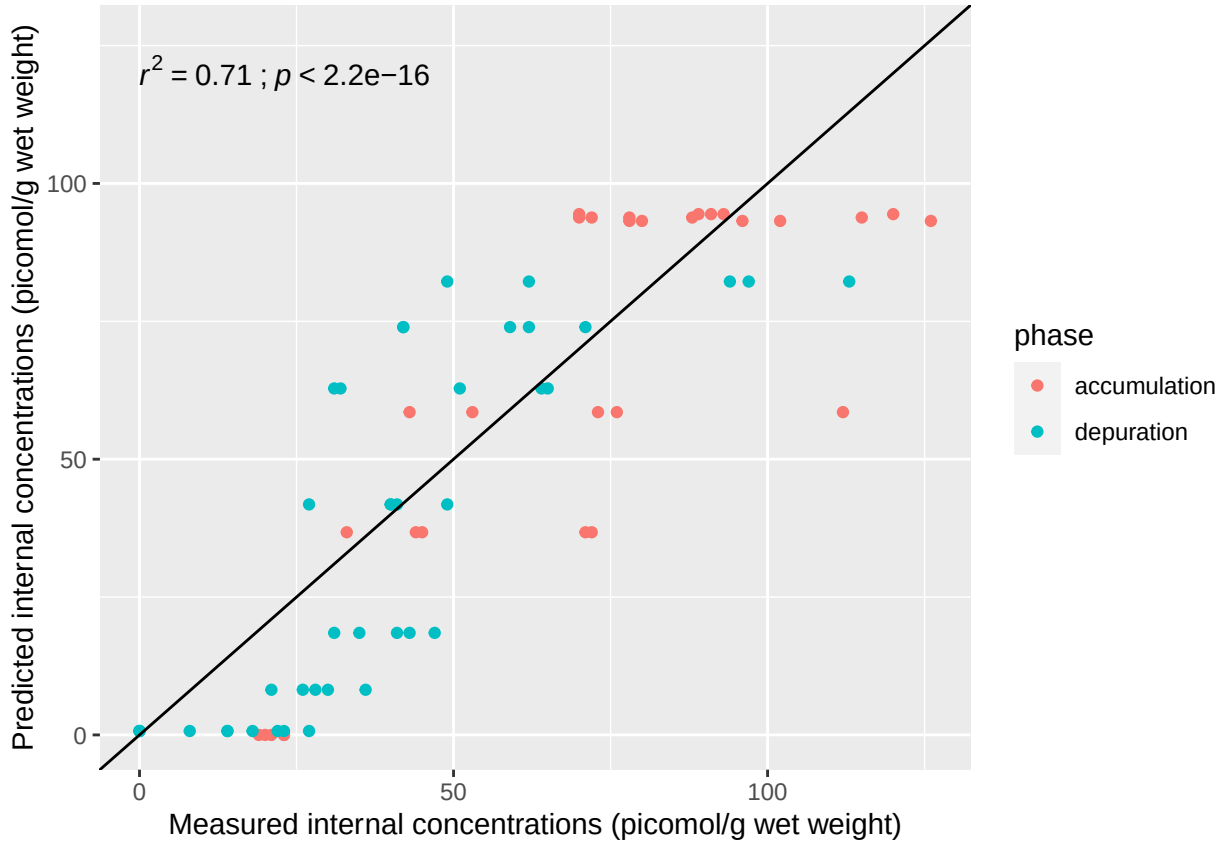


Figure 3: Visual Predictive Check comparing observations (x-axis) to prediction (y-axis) for both the accumulation (in red) and the depuration (in blue) phases.

Persp. 1 - option 2

Given that k_u and k_e parameters were provided as means and standard deviations, it is tempting to add uncertainties around the previous mean predicted curve. However, no information is available on parameter σ to appropriately characterize the normal distribution of the stochastic part of the model (equation (2)). Without additional information, all parameters will be assumed to be normally distributed. Note also that, in doing as such for simulations, we do not account for any correlation between model parameters.

Below are the probability distributions we simulated for both kinetic parameters k_u and k_e.

```
# Built normal distributions for model parameters
niter <- 1000 # sampling size in parameter distributions
ku.sd <- mean(tab1$ku_sd) # ku standard deviation
ku <- rnorm(niter, ku.mean, ku.sd) # ku distribution
ke.sd <- mean(tab1$ke_sd) # ke standard deviation
ke <- rnorm(niter, ke.mean, ke.sd) # ke distribution
# Plot the simulated normal distributions
# for the model parameters ku and ke
g1 <- ggplot(as.data.frame(ku), aes(ku)) +
  geom_histogram(binwidth = 1)
g2 <- ggplot(as.data.frame(ke), aes(ke)) +
  geom_histogram(binwidth = 0.01)
grid.arrange(g1, g2, ncol = 2)
```

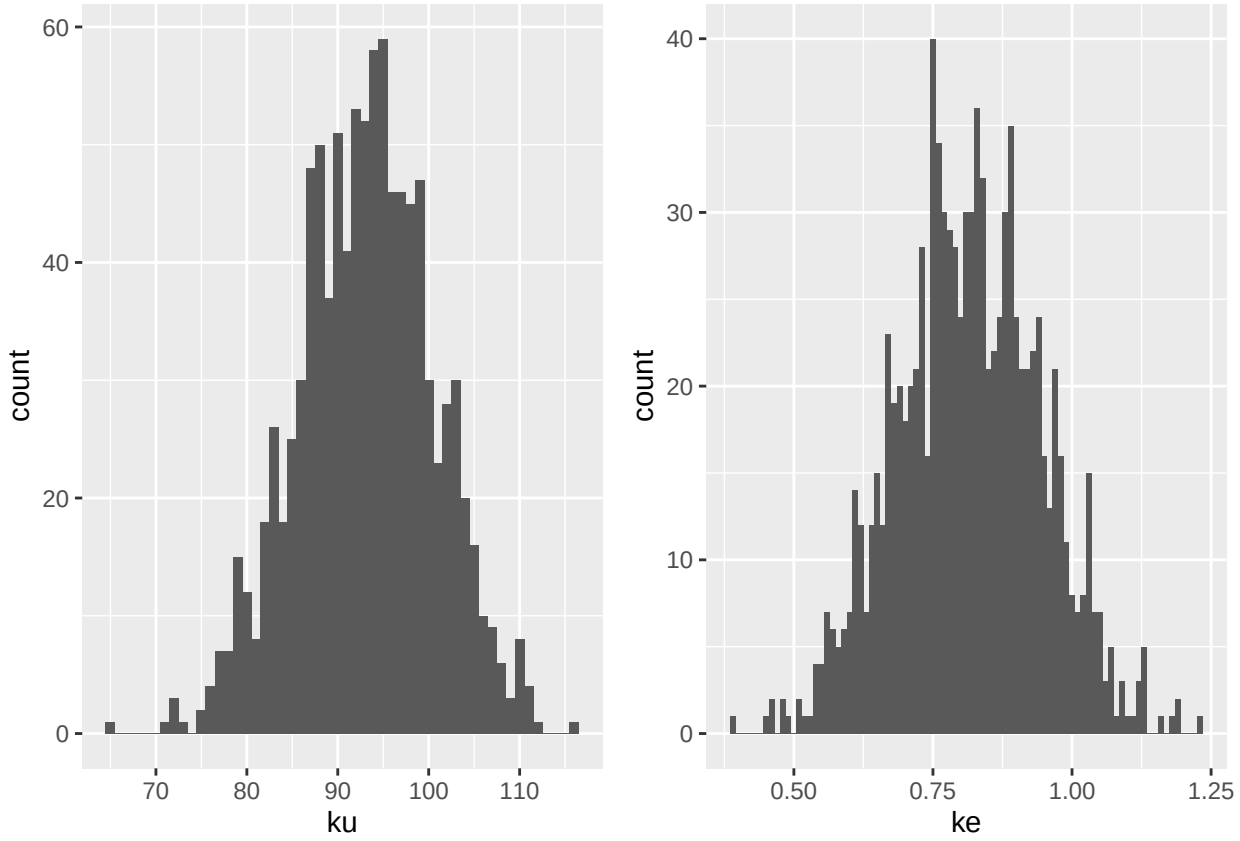


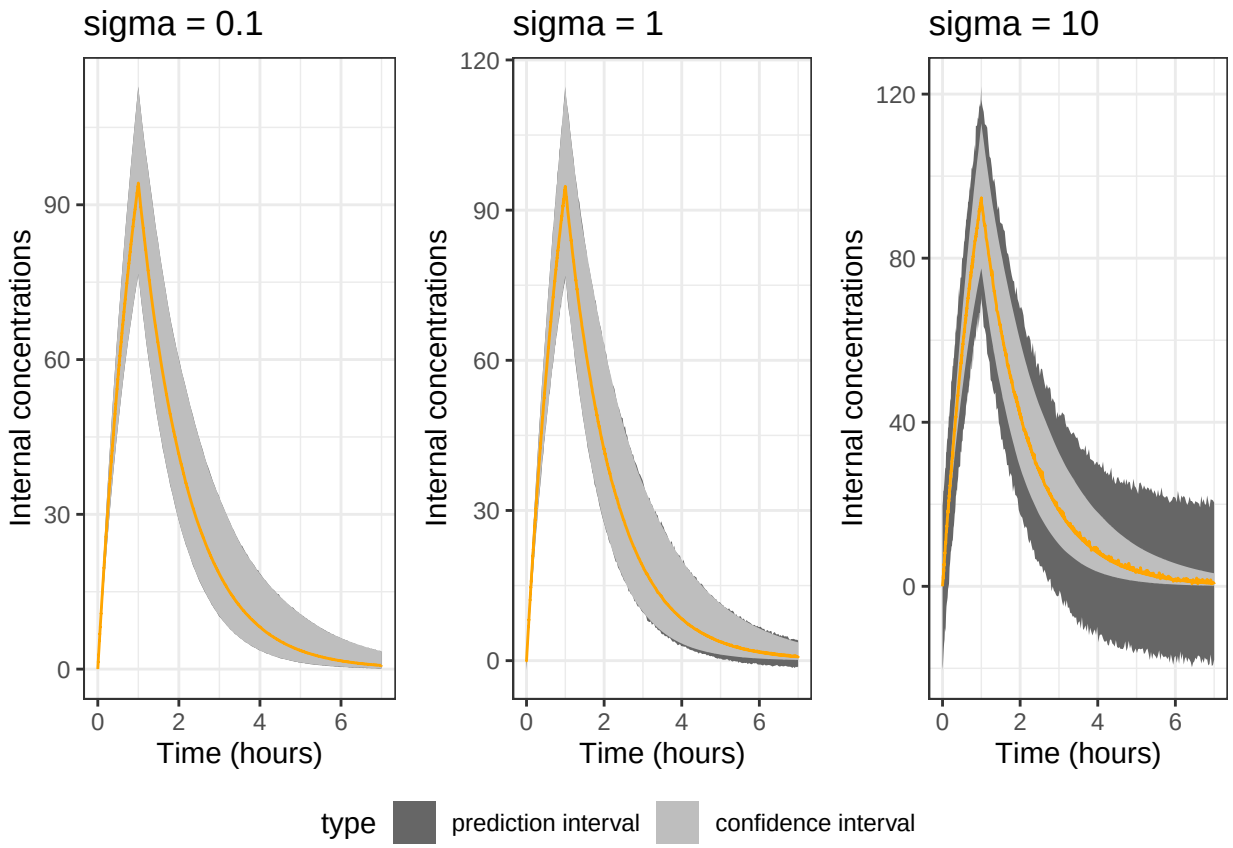
Figure 4: Probability distributions of both kinetic parameters: k_u (left panel) and k_e (right panel).

As σ is expressed in the same unit as the observed concentrations, in first intention, we could assume that σ follows a normal distribution of mean and standard deviation equal to the mean and standard deviation of the observations. Nevertheless, an alternative that is less computing demanding, is to fix σ at any other arbitrary value.

Below are simulations accounting for either only the uncertainty on parameter estimates k_u and k_e, or also the stochastic part of the model in addition (namely, using the full model with parameter σ). The first simulations provide credible intervals around point theoretical values, while the second simulations provide prediction intervals around point theoretical values. We performed simulations with three different fixed value for σ: 0.1, 1 and 10.

```r
# Simulation of internal concentrations
# considering the above probability distributions
# and a fixed sigma value
source("simu-sigma.R")
Qpred.0.1 <- simu(sigma = 0.1) # 50% of the observed sd
Qpred.1 <- simu(sigma = 1)
Qpred.10 <- simu(sigma = 10)
```

```r
g1 <- ggplot(data = Qpred.0.1, aes(x = time, y = median)) +
  geom_ribbon(aes(ymin = lower, ymax = upper, fill = type),
              linetype = "dashed") +
  geom_line(col = "orange") +
  theme_bw() +
  labs(title = "sigma = 0.1") +
  xlab("Time (hours)") +
  ylab("Internal concentrations") +
  # ylim(0, 120) +
  scale_fill_manual(values = c("gray40", "gray"))
g2 <- ggplot(data = Qpred.1, aes(x = time, y = median)) +
  geom_ribbon(aes(ymin = lower, ymax = upper, fill = type),
              linetype = "dashed") +
  geom_line(col = "orange") +
  theme_bw() +
  labs(title = "sigma = 1") +
  xlab("Time (hours)") +
  ylab("Internal concentrations") +
  # ylim(0, 120) +
  scale_fill_manual(values = c("gray40", "gray"))
g3 <- ggplot(data = Qpred.10, aes(x = time, y = median)) +
  geom_ribbon(aes(ymin = lower, ymax = upper, fill = type),
              linetype = "dashed") +
  geom_line(col = "orange") +
  theme_bw() +
  labs(title = "sigma = 10") +
  xlab("Time (hours)") +
  ylab("Internal concentrations") +
  # ylim(0, 120) +
  scale_fill_manual(values = c("gray40", "gray"))
ggarrange(g1, g2, g3, nrow = 1, ncol = 3, common.legend = TRUE, legend="bottom")
```



Perspective 2

In this section, we fit the one compartment model (equations (2)) under a frequentist framework based on the R function `nls()` of the R software, and the R-package `nlstools` that provides a collection of tools to evaluate the goodness-of-fit of Gaussian non linear models (Baty et al. 2015).

Fit the one compartment model

```
# Define the model to fit
model <- as.formula(conc ~ (time <= 1) * (ku/ke) * 1.485 * ( 1 - exp(- ke * time)) + (time > 1) * (ku/k
```

```
# Fit the model
fit <- nls(model, data = df, start = list(ku = ku.mean, ke = ke.mean))
```

Summary of parameter estimates

Below is a summary of parameter estimates with confidence intervals, the residual standard error and the sum of squares, as well as the correlation matrix of the estimates.

```
overview(fit)
```

```
------
Formula: conc ~ (time <= 1) * (ku/ke) * 1.485 * (1 - exp(-ke * time)) +
    (time > 1) * (ku/ke) * 1.485 * (exp(ke * (1 - time)) - exp(-ke *
        time))
```

```
Parameters:
   Estimate Std. Error t value Pr(>|t|)
ku  84.6970     6.7005  12.640  < 2e-16 ***
ke   0.6929     0.1133   6.114 5.36e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 20.02 on 68 degrees of freedom

Number of iterations to convergence: 17
Achieved convergence tolerance: 8.1e-06

------
Residual sum of squares: 27200

------
t-based confidence interval:
         2.5%      97.5%
ku 71.3264169 98.0675376
ke  0.4667698  0.9190471

------
Correlation matrix:
          ku        ke
ku 1.0000000 0.8539935
ke 0.8539935 1.0000000
```

Fitting vizualization

The Figure below shows the visualization of the fitted curve superimposed to the observations.

```
par(mar=c(4, 4, 0.2, 0.2))
plotfit(fit, las = 1, pch.obs = 19, ylim = c(0, 120),
        smooth = TRUE, col.fit = "orange", lwd = 3,
        xlab = "Time (hours)", ylab = "Internal concentrations")
grid()
```

Check the residuals

```
fit.res <- nlsResiduals(fit)
```

```
par(mar=c(4, 4, 3, 0.2))
plot(fit.res)
```

Figure 5 does not underline any problem with the model assumptions: residuals seem to be approximately normally distributed (a clear alignment along the diagonal in the QQ plot), without evidence of autocorrelation or heteroscedastic variance.

```
test.nlsResiduals(fit.res)
```

```
------
    Shapiro-Wilk normality test

data:  stdres
```

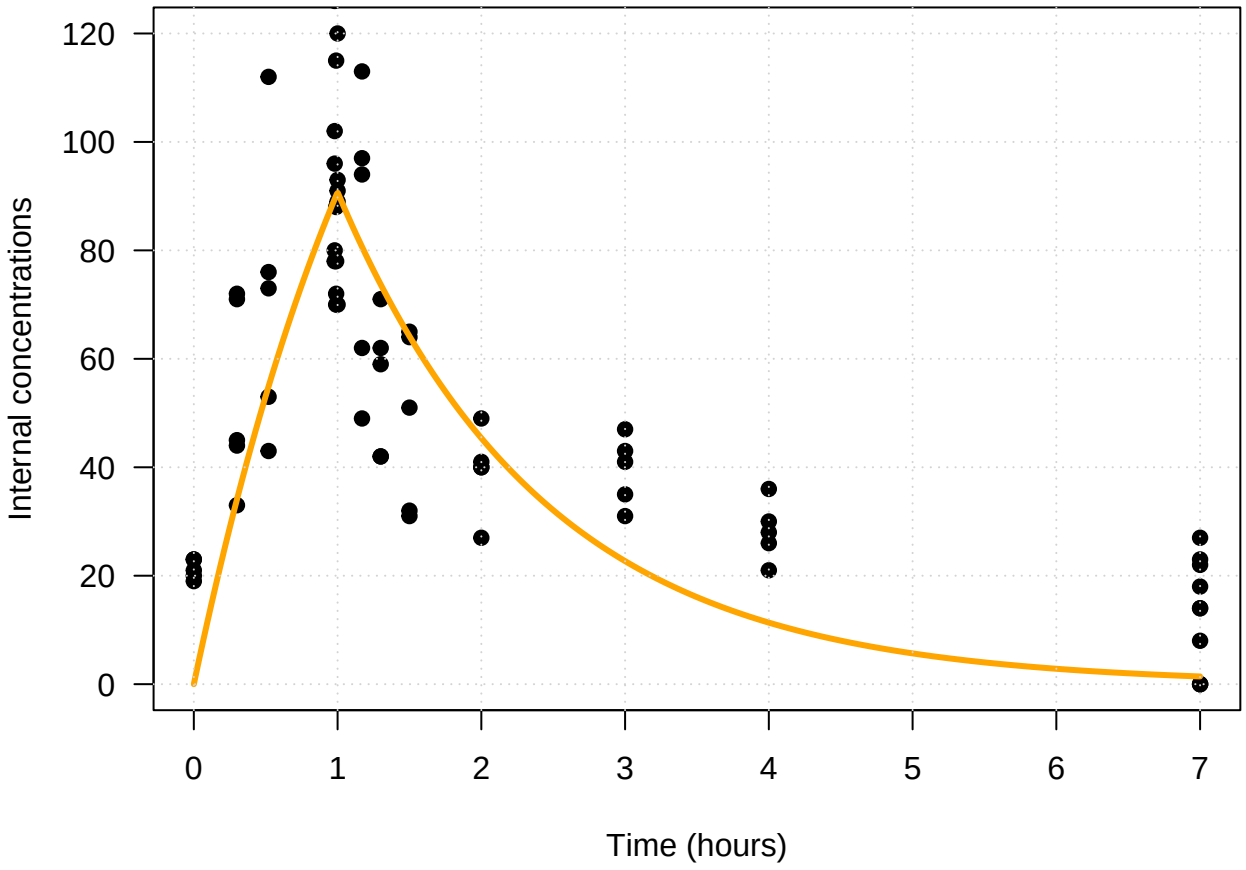


Figure 5: Plot of the data (dependent vs. independent variable) with the fitted model superimposed.

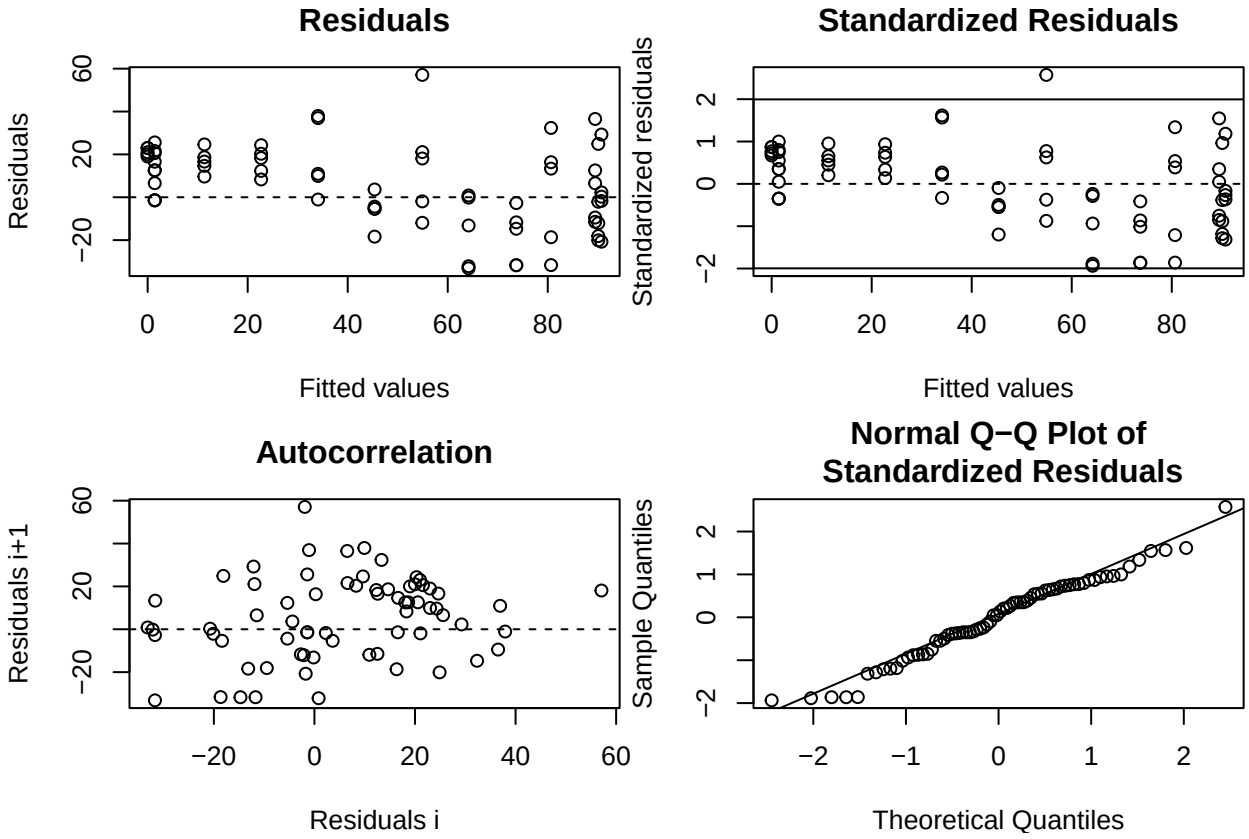


Figure 6: Plot of residuals: (top left panel) Raw residuals vs. fitted values; (top right panel) Standardized residuals (with mean = 0 and standard deviation = 1) vs. fitted values; (bottom left panel) Autocorrelation plot; (bottom right panel) QQ plot of the standardized residuals.

```
W = 0.97836, p-value = 0.2665


------


    Runs Test

data:  as.factor(run)
Standard Normal = -3.5964, p-value = 0.0003227
alternative hypothesis: two.sided
```

With this example, the null hypothesis of a normal distribution for the residuals cannot be rejected (Shapiro-Wilk test: $p = 0.27$) but there is an indication of autocorrelation (runs test: $p = 0.00032$).

Parameter correlations

```
source("nlsConfRegions.R")
fit.conf <- nlsConfRegions(fit, exp = 3, length = 2000)
```

```
plot(fit.conf)
```

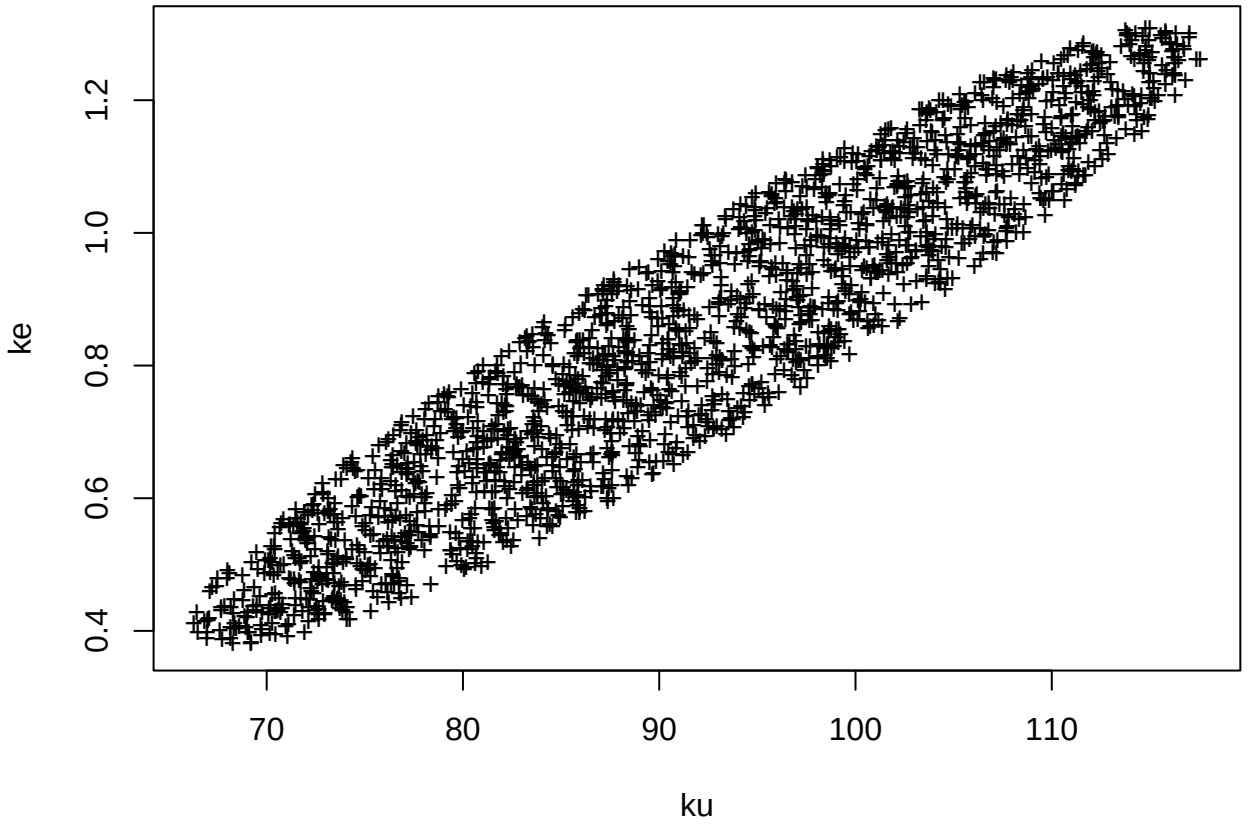


Figure 7: Projections of the confidence region according to the Beale's criterion.

The Beale's confidence region between k_u and k_e is perfectly elliptical with a global minimum at the centre (Beale 1960). This indicates a good non linear regression model fit. Moreover, the narrow elliptic shape of the Beale's confidence region between both kinetic parameters reflects a relatively high correlation (correlation coefficient equals to 0.85).

Detect influential observations

Both jackknife and bootstrap procedures are implemented in the R-package `nlstools`. By using a leave-one-out procedure, the jackknife procedure provides jackknife parameter estimates together with confidence intervals. It also shows the influence of each observation on each parameter estimate (Figure).

```
fit.jack <- nlsJack(fit)
summary(fit.jack)
```

```
------
Jackknife statistics
    Estimates       Bias
ku 79.5406454 5.15633186
ke  0.5955157 0.09739271

------
Jackknife confidence intervals
           Low         Up
ku 51.91624518 107.165046
ke  0.08010752   1.110924

------
Influential values
* Observation 7 is influential on ku
* Observation 9 is influential on ku
* Observation 12 is influential on ku
* Observation 14 is influential on ku
* Observation 15 is influential on ku
* Observation 19 is influential on ku
* Observation 21 is influential on ku
* Observation 22 is influential on ku
* Observation 23 is influential on ku
* Observation 26 is influential on ku
* Observation 29 is influential on ku
* Observation 41 is influential on ku
* Observation 43 is influential on ku
* Observation 46 is influential on ku
* Observation 51 is influential on ku
* Observation 52 is influential on ku
* Observation 54 is influential on ku
* Observation 55 is influential on ku
* Observation 57 is influential on ku
* Observation 58 is influential on ku
* Observation 59 is influential on ku
* Observation 60 is influential on ku
* Observation 7 is influential on ke
* Observation 9 is influential on ke
* Observation 12 is influential on ke
* Observation 14 is influential on ke
* Observation 15 is influential on ke
* Observation 19 is influential on ke
* Observation 22 is influential on ke
* Observation 26 is influential on ke
* Observation 41 is influential on ke
```

```
* Observation 43 is influential on ke
* Observation 46 is influential on ke
* Observation 51 is influential on ke
* Observation 52 is influential on ke
* Observation 53 is influential on ke
* Observation 54 is influential on ke
* Observation 55 is influential on ke
* Observation 57 is influential on ke
* Observation 58 is influential on ke
* Observation 59 is influential on ke
* Observation 60 is influential on ke
```

```
plot(fit.jack)
```

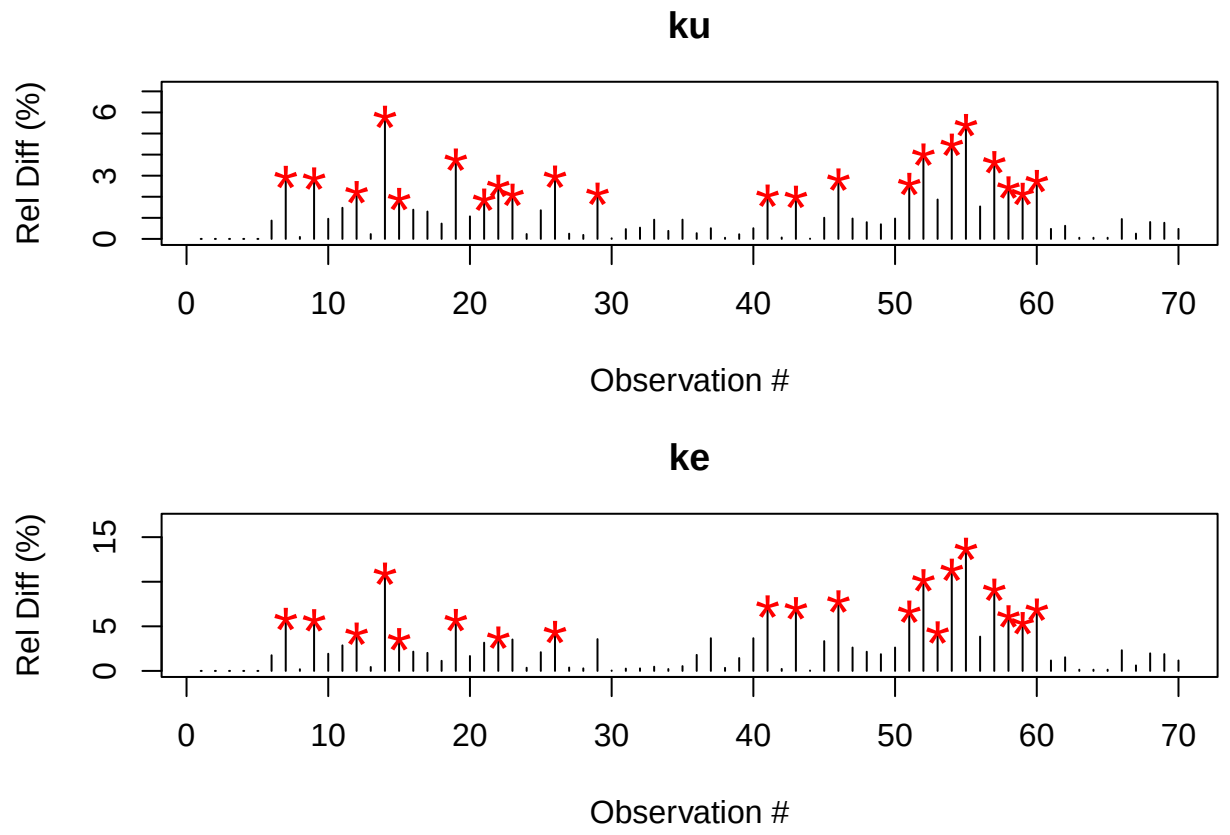


Figure 8: Resampling Jackknife procedures showing the influence of each observation on each parameter estimate.

Perspective 3

In this section, we fit the one compartment model (equations (1)) under a Bayesian framework with the R-package `rbioacc` (Ratier et al. 2022). The same calculation can be easily reproduced on-line with the MOSAIC web platform and its `bioacc` module: https://mosaic.univ-lyon1.fr/bioacc.

Model fitting

```
# Prepare the data to be use in the `rbioacc` package
mdf <- modelData(df, time_accumulation = 1, )
```

```
# fit the TK model built by default from the data
fit <- fitTK(mdf, refresh = 0)
```

Model equations

```
# Below is the code line allowing to get
# the used model equations
equations(fit, df)
```

Fitting results

```
# Get parameter estimates
# medians and 95% credible intervals
quantile_table(fit)
```

```
               2.5%        50%      97.5% parameter
ku        53.5243660 65.8793723 84.4982393        ku
kee        0.3757962  0.5628269  0.8804306       kee
sigmaConc 15.4280178 18.1512066 21.6314885 sigmaConc
```

```
# Fitting plot
plot(fit)
```

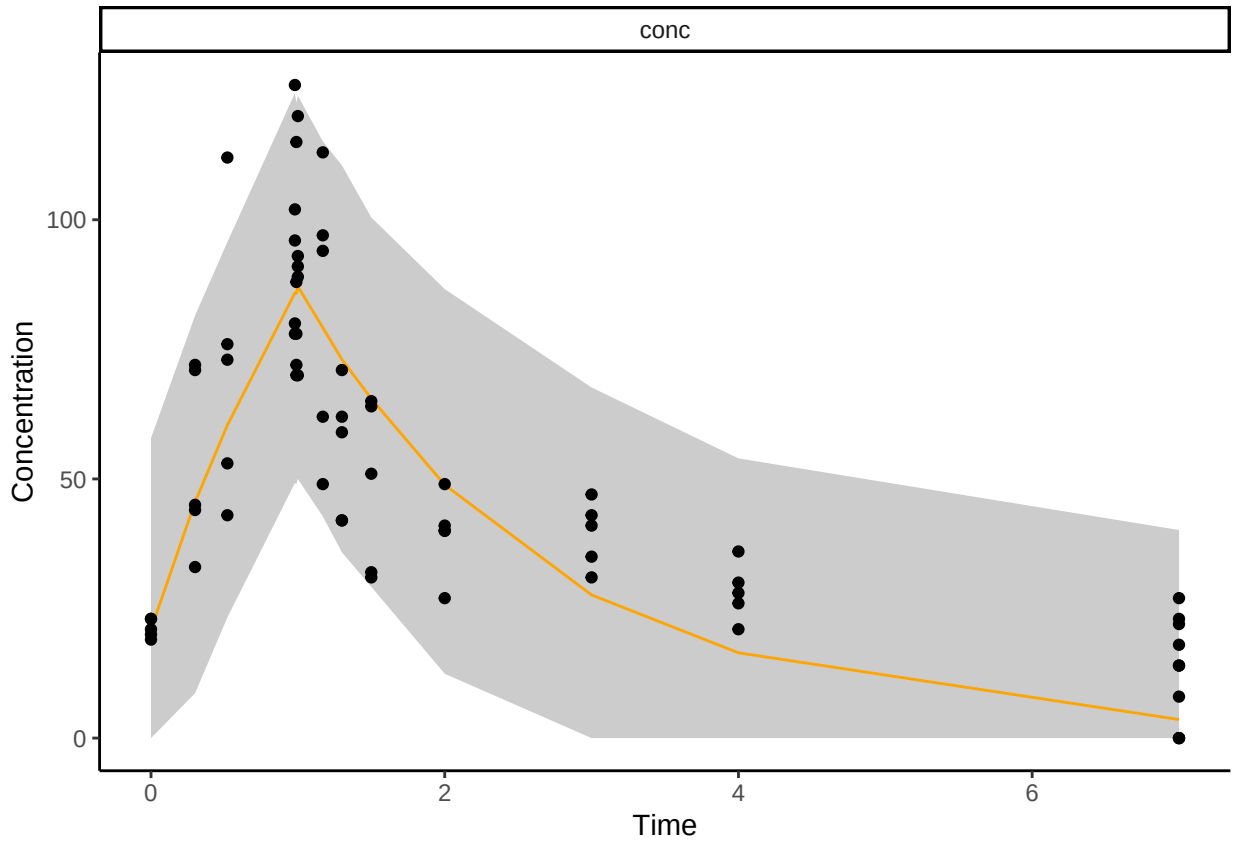


Figure 9: Fitting plot with black dots representing the observed data, the solid orange line the median predictive model and the grey area the 95% uncertainty band including the uncertainty on the model parameter estimates as well as the stochastic part of the model.

Bioaccumulation metric

```
# Calculation of the posterior probability distribution
# of the kinetic bioconcentration factor
bm <- bioacc_metric(fit)
# Display median and 95% credible interval of the BCF_k
quantile(bm$BCFk, probs = c(0.025, 0.5, 0.975))
```

```
     2.5%       50%     97.5%
 92.35396 116.67281 147.62552
```

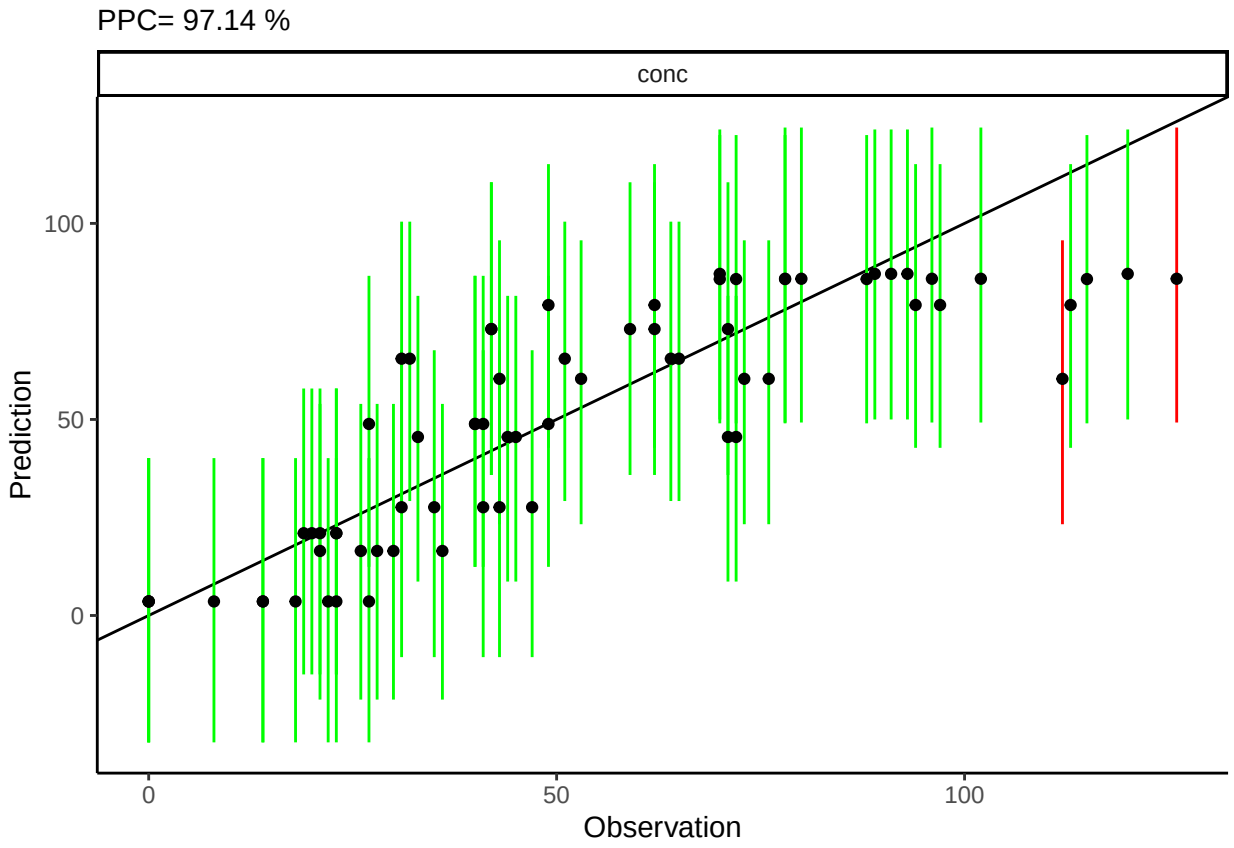
The 95% elimination time can also be easily calulated.

```
signif(t95(fit), digits = 3)
```

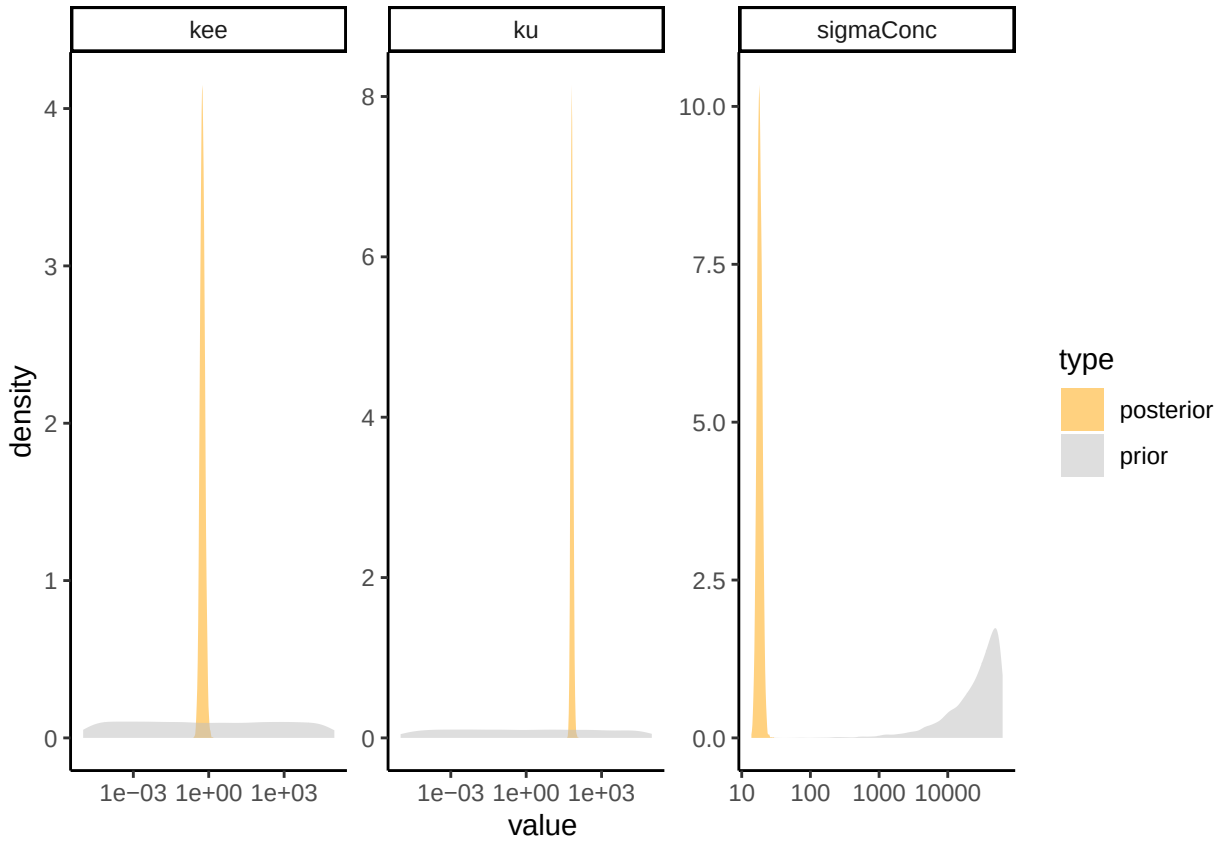
```
[1] 5.32
```

Goodness-of-fit criteria

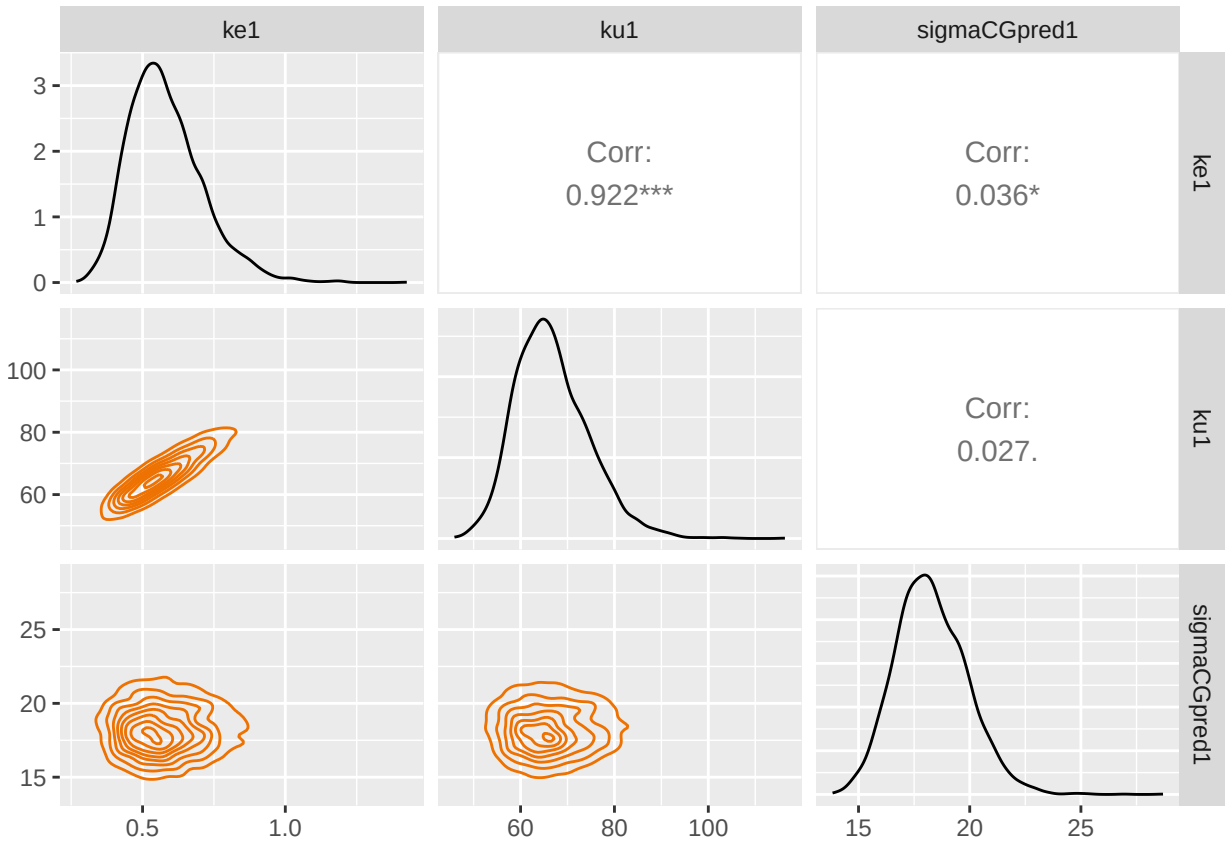
```
# Posterior Predictive Check (PPC)
# The expectation is to get ~95% of data
# within their prediction interval
ppc(fit)
```




```
# Compare priors and posteriors
plot_PriorPost(fit)
```



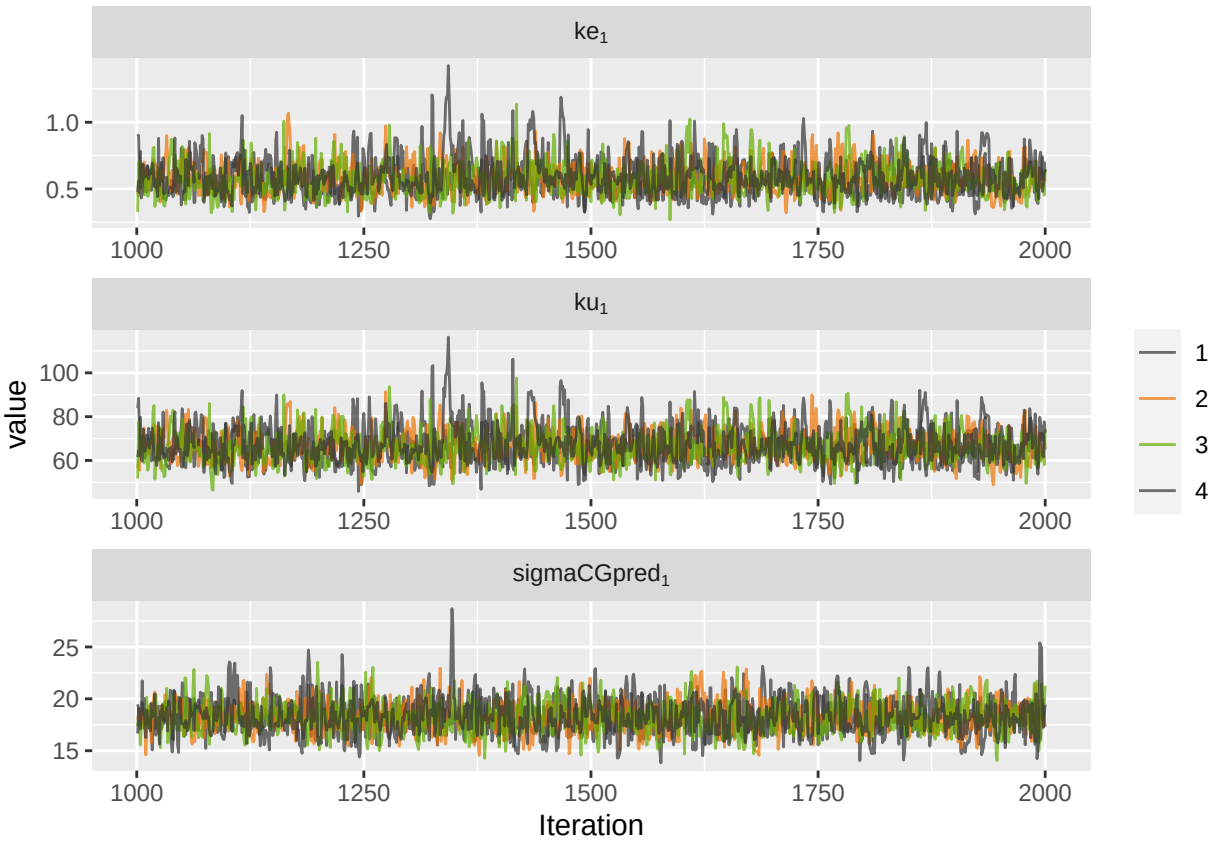

```
# Check for correlations between parameters
corrPlot(fit)
```




```
# Check  for non-signigicantly different traces
# of the four MCMC chains run in paralell
psrf(fit)
```

```
           PSRF parameter
ku            1        ku
kee           1       kee
sigmaConc     1 sigmaConc
```

```
# Look at the traces of the 4 MCMC chains
mcmcTraces(fit)
```



References

Ashauer, Roman, Ivo Caravatti, Anita Hintermeister, and Beate Escher. 2010. "Bioaccumulation kinetics of organic xenobiotic pollutants in the freshwater invertebrate Gammarus pulex modeled with prediction intervals." *Environmental Toxicology and Chemistry* 29 (7): 1625–36. https://doi.org/10.1002/etc.175.

Baty, Florent, Christian Ritz, Sandrine Charles, Martin Brutsche, Jean-Pierre P Flandrois, and Marie Laure Delignette-Muller. 2015. "A Toolbox for Nonlinear Regression in R: The Package nlstools." *Journal of Statistical Software* 66 (5): 1–21.

Beale, E. M. L. 1960. "Confidence Regions in Non-Linear Estimation." *Journal of the Royal Statistical Society. Series B (Methodological)* 22 (1): 41–88.

Charles, Sandrine, Aude Ratier, and Christelle Lopes. 2021. "Generic Solving of One-compartment Toxicokinetic Models." *Journal of Exploratory Research in Pharmacology* 6 (4): 158–67. https://doi.org/10.14218/jerp.2021.00024.

Ratier, Aude, Virgile Baudrot, Miléna Kaag, Aurélie Siberchicot, Christelle Lopes, and Sandrine Charles. 2022. "rbioacc: an R-package to analyse toxicokinetic data." *Ecotoxicology and Environmental Safety* 242 (July): 113875. https://doi.org/10.1016/j.ecoenv.2022.113875.

APPENDIX

Table of raw data

```
df <- read.table("data.txt", header = TRUE, sep = "")
kable(df[1:25,], format="latex")
```

time	conc	replicate	expw
0.00	23	1	1.485
0.00	19	2	1.485
0.00	20	3	1.485
0.00	21	4	1.485
0.00	23	5	1.485
0.30	44	1	1.485
0.30	72	2	1.485
0.30	33	3	1.485
0.30	71	4	1.485
0.30	45	5	1.485
0.52	43	1	1.485
0.52	76	2	1.485
0.52	53	3	1.485
0.52	112	4	1.485
0.52	73	5	1.485
0.98	102	1	1.485
0.98	78	2	1.485
0.98	96	3	1.485
0.98	126	4	1.485
0.98	80	5	1.485
0.99	72	1	1.485
0.99	115	2	1.485
0.99	70	3	1.485
0.99	88	4	1.485
0.99	78	5	1.485

One more thing

This will be Appendix B.