Dose-Time Response Modeling of Rat Brain AChE Activity: Propoxur Gavage Dosing

June 24, 2005

1 Preamble

Here is some code to set up the analysis: loading required libraries and datasets, and defining some functions.

First, `CarbamateData` loads the full dataset for this risk assessment, and causes the library `DRUtils` to be loaded.

```
> library(CarbamateData)

Loading required package: DRUtils
Loading required package: nlme
Loading required package: tcltk
```

Set up lattice to use B&W instead of color:

```
> library(lattice)
> ltheme <- canonical.theme(color = FALSE)
> ltheme$strip.background$col <- "transparent"
> lattice.options(default.theme = ltheme)
```

Use package `Hmisc` for some formatting support.

```
> library(Hmisc)

Hmisc library by Frank E Harrell Jr

Type library(help='Hmisc'), ?Overview, or ?Hmisc.Overview')
to see overall documentation.

NOTE:Hmisc no longer redefines [.factor to drop unused levels when
subsetting.  To get the old behavior of Hmisc type dropUnusedLevels().

Attaching package 'Hmisc':


        The following object(s) are masked from package:stats :

         ecdf
```

The rat gavage data for this analysis are in `PadillaData`. The following code prints out documentation for the datasets in use:

```
> printDataDoc(PadillaData)
```

```
----------------------------------------
Data set: PadillaData

Dataset creation date: Wed Jun 22 12:07:28 2005
Script name: /home/setzer/tasks/CarbamateCRA/EmpiricalDoseResponsesFinal/Data/getdata.R
Script last modified: 2005-06-22 12:05:47
sysname: Linux
release: 2.4.24-openmosix2
version: #1 Wed Jul 28 14:33:52 CEST 2004
nodename: node24.cluster2.org
machine: i686
login: unknown
user: setzer
----------------------------------------
```

The following function turns out to be quite useful on subsetted dataframes. It just eliminates unused levels of all factors in the data frame:

```
> CleanUp <- function(x) {
+     for (nm in names(x)) {
+         if (is.factor(x[, nm]))
+             x[, nm] <- factor(x[, nm])
+     }
+     x
+ }
```

To get starting values, we often have to extract values from a previously fit model. The following function simplifies that. The argument `what` is a regular expression:

```
> getParms <- function(what, Par) {
+     Par[grep(what, names(Par))]
+ }
```

This script is for modeling the dose-time response for rat brain. via gavage dosing. It includes an acute study that includes a time course at one dose and a dose-response at around the time of peak effect.

All the data used for this DR model are in `PadillaData`. The methiocarb data need to be extracted, then several variables renamed (cheact = Brain.R, tmpstds == time) added to the dataset.

Set up the analysis dataset.

```
> dta <- subset(PadillaData, chemical %in% "propoxur", select = c("dose",
+     "brain.R", "TMPSTDS"))
> names(dta) <- c("dose", "cheact", "tmpstds")
> dta <- CleanUp(subset(dta, !is.na(cheact)))
```

Summary of this dataset:

```
> summary(dta)

      dose           cheact          tmpstds
 Min.   : 0.00   Min.   : 752.1   Min.   : 0.2500
 1st Qu.: 0.65   1st Qu.:3167.9   1st Qu.: 0.6667
 Median :10.00   Median :4823.1   Median : 0.6667
 Mean   :11.04   Mean   :4405.4   Mean   : 2.9576
 3rd Qu.:20.00   3rd Qu.:5721.9   3rd Qu.: 1.0000
 Max.   :20.00   Max.   :6793.3   Max.   :24.0000

> with(dta, table(dose, tmpstds))
```

```
     tmpstds
dose  0.25 0.5 0.666666666666667 1 4 24
  0   1    1   5                 1 1 1
  0.3 0    0   5                 0 0 0
  1   0    0   5                 0 0 0
  3   0    0   5                 0 0 0
  10  0    0   5                 0 0 0
  20  5    5   5                 5 5 4
```

Padilla's data are from an acute study, with multiple doses at 0.67 hours, and multiple timepoints for 25 mg/kg.

There is a control animal at each time point, so set up a factor with a level for each time point, to allow the control to vary from time to time. We need to check whether these controls are heterogeneous.

```
> dta$ftime <- factor(dta$tmpstds)
```

2 Dose-Response Modeling

2.1 strategy

Use the model with simple exponential recovery (`tcmfn4()`), first. It looks as if the time to peak effect for all these chemicals is likely to be less than a half-hour (the earliest time point in this dataset), so the exponential recovery model is probably indistinguishable from the one with the more complex time course.

Fitting the model will follow these steps:

1. First, use `GetInitialValues()` to get starting values for the model against these data, and determine if we can estimate `lg` and `tz` of the dose-response parameters.
2. Next, fit `tcmfn4()` using the parameterizations determined in the previous step, and a constant variance model.
3. Try fitting with other variance models the same model for fixed and random effects, using AIC to identify the best variance model.
4. Are the control values heterogeneous?

2.2 Initial Values

Save the initial values so that we do not need to go through all this to re-run the analysis. Also, set the argument `delta` to 0.25, the earliest non-zero time point.

```
> formals(tcmfn4)$delta <- min(dta$tmpstds[dta$tmpstds > 0])
> initfile <- paste("initvals-brain-DR-1.RData", sep = "")
> if (!file.exists(initfile)) {
+     lA.start <- lm(I(log(cheact)) ~ ftime - 1, data = CleanUp(subset(dta,
+         dose %in% 0)))
+     Start <- c(coef(lA.start), log(0.5), -2, log(1), log(1.5))
+     init1 <- GetInitialValues(cheact ~ tcmfn4(dose, tmpstds,
+         lA = lA, tz = tz, lD = lD, lg = lg, lTr = lTr), data = dta,
+         params = list(lA ~ ftime - 1, lD ~ 1, tz ~ 1, lg ~ 1,
+             lTr ~ 1), start = Start, weights = varPower(value = 1))
+     save(init1, file = initfile)
+ } else load(initfile)
> tmp <- t(init1$Redundancy[[1]]$Eigens)
> tmp <- tmp[-grep("^lA", rownames(tmp)), ]
> round(tmp[, 1:5], digits = 2)
```

```
           [,1]  [,2] [,3] [,4] [,5]
CondIndex 52.54 12.53 5.81 2.91 1.99
mu         0.04  0.16 0.34 0.69 1.00
lD         0.71  0.28 0.00 0.00 0.00
tz         0.93  0.07 0.00 0.00 0.00
lg         1.00  0.00 0.00 0.00 0.00
lTr        0.01  0.00 0.95 0.00 0.00
```

The largest condition index is associated with a linear combination of `lD`, `tz`, and `lg`. There is a very good chance that at least one of these will need to be fixed, and not estimated. Preliminary optimization moves `tz` to -15, so set it to -10 and do not try to estimate it.

```
> Start <- init1$start$beta
> Start <- Start[-grep("tz", names(Start))]
> formals(tcmfn4)$tz <- -10
> drmod1 <- gnls(cheact ~ tcmfn4(dose, tmpstds, lA = lA, lD = lD,
+     lg = lg, lTr = lTr), data = dta, params = list(lA ~ ftime -
+     1, lD ~ 1, lg ~ 1, lTr ~ 1), start = Start)
```

Now, plot the residuals, to see if we need to try to add `varPower()` to the `weights`:

```
> print(plot(drmod1))
```

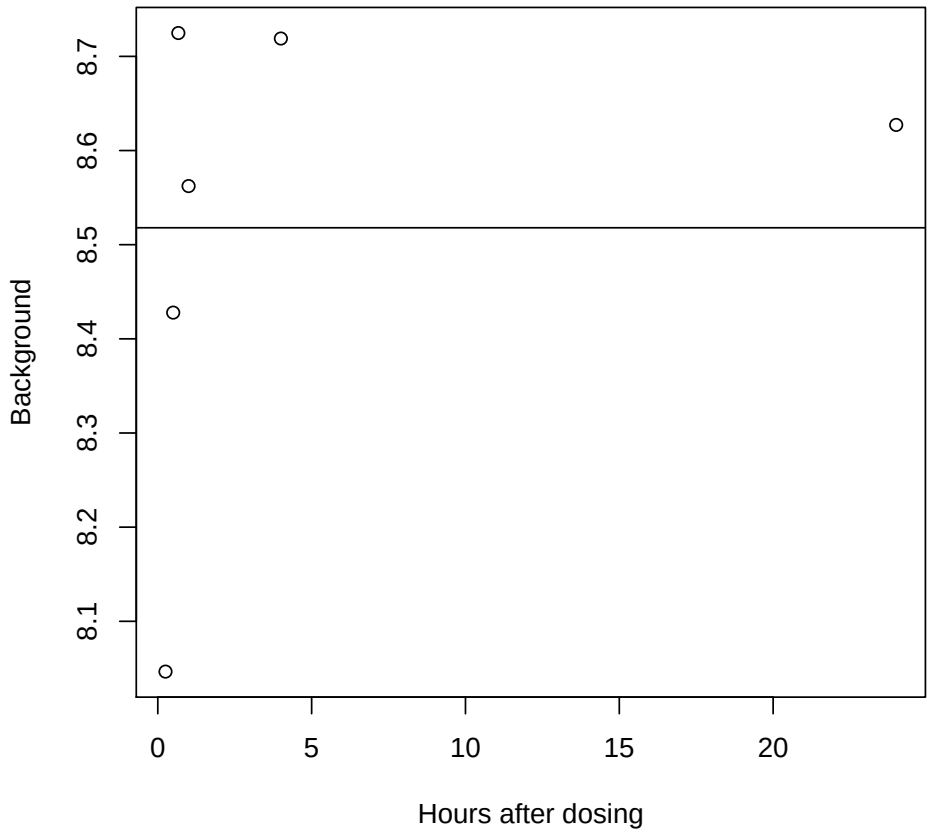
This looks pretty good as is.

Are the control values heterogeneous?

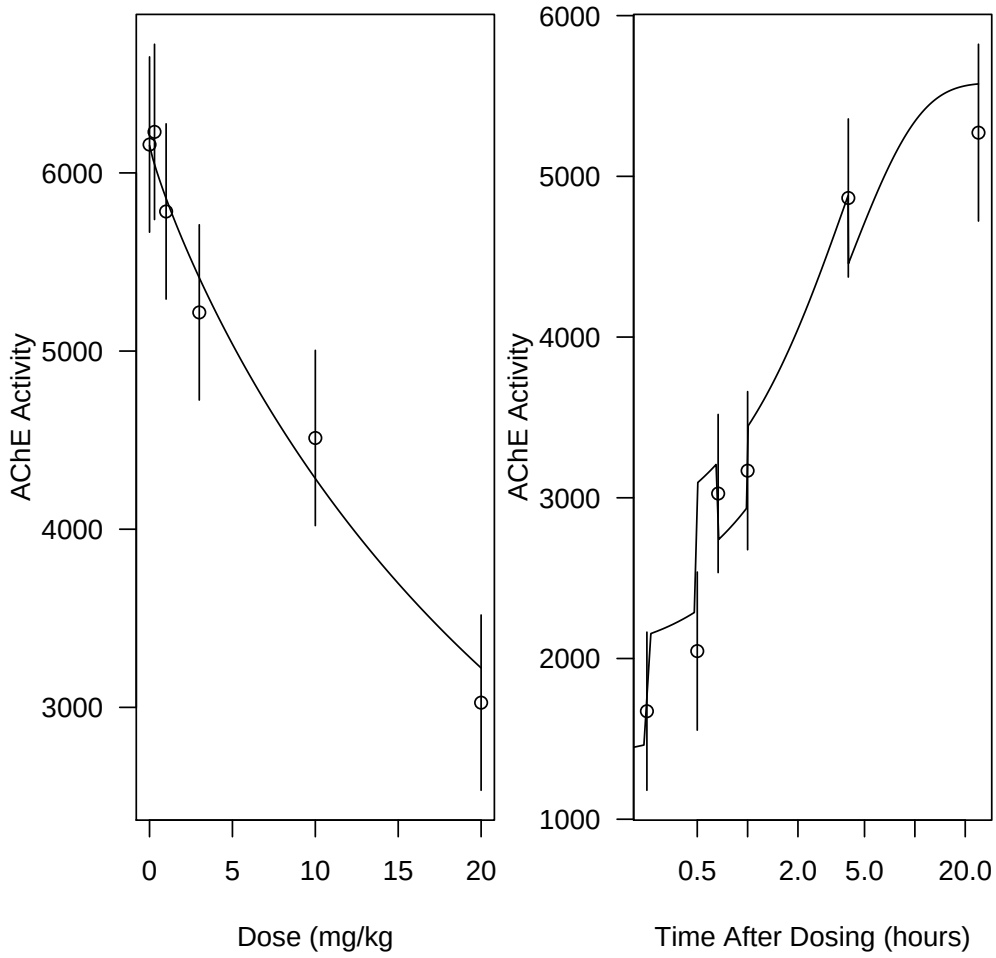
```
> nms <- names(getParms("^lA", coef(drmod1)))
> L <- diag(length(nms)) - matrix(1, nrow = length(nms), ncol = length(nms))/length(nms)
> L <- L[-nrow(L), ]
> colnames(L) <- nms
> anova(drmod1, L = L)

Denom. DF: 50
 F-test for linear combination(s)
  lA.ftime0.25 lA.ftime0.5 lA.ftime0.666666666666667  lA.ftime1  lA.ftime4
1    0.8333333  -0.1666667                -0.1666667 -0.1666667 -0.1666667
2   -0.1666667   0.8333333                -0.1666667 -0.1666667 -0.1666667
3   -0.1666667  -0.1666667                 0.8333333 -0.1666667 -0.1666667
4   -0.1666667  -0.1666667                -0.1666667  0.8333333 -0.1666667
5   -0.1666667  -0.1666667                -0.1666667 -0.1666667  0.8333333
  lA.ftime24
1 -0.1666667
2 -0.1666667
3 -0.1666667
4 -0.1666667
5 -0.1666667
  numDF  F-value p-value
1     5 5.938846   2e-04
```

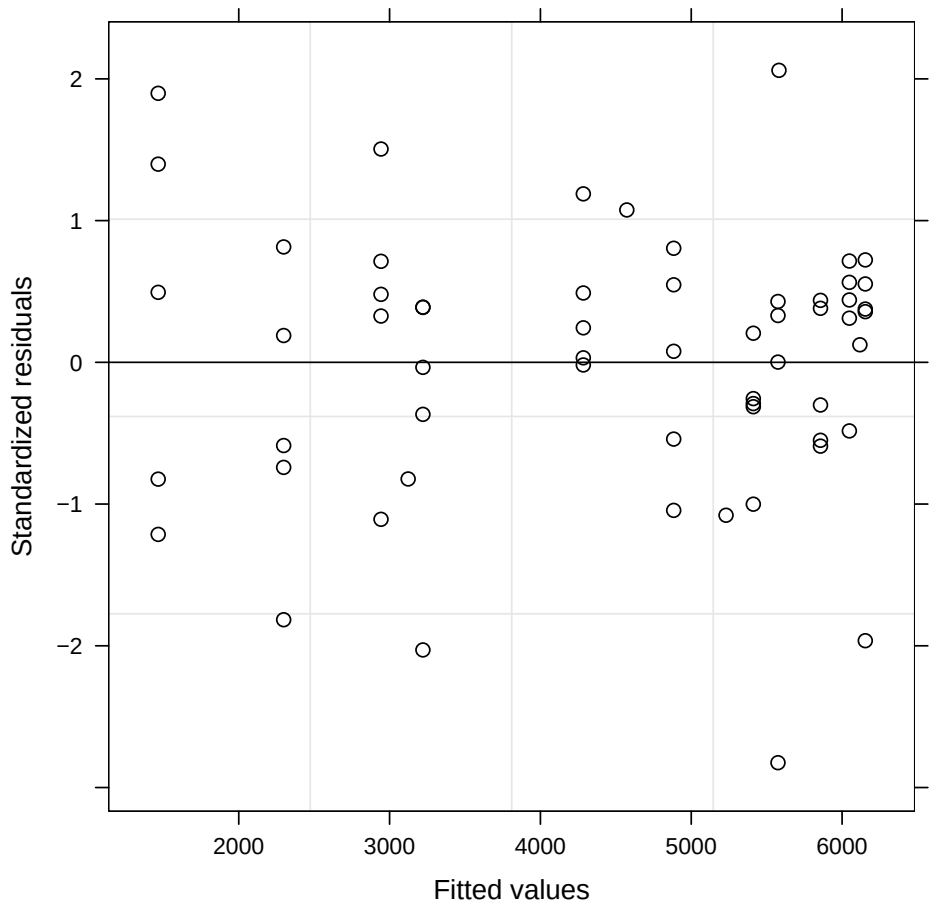
Apparently so. Calculate the deviations from their mean, and plot versus time:

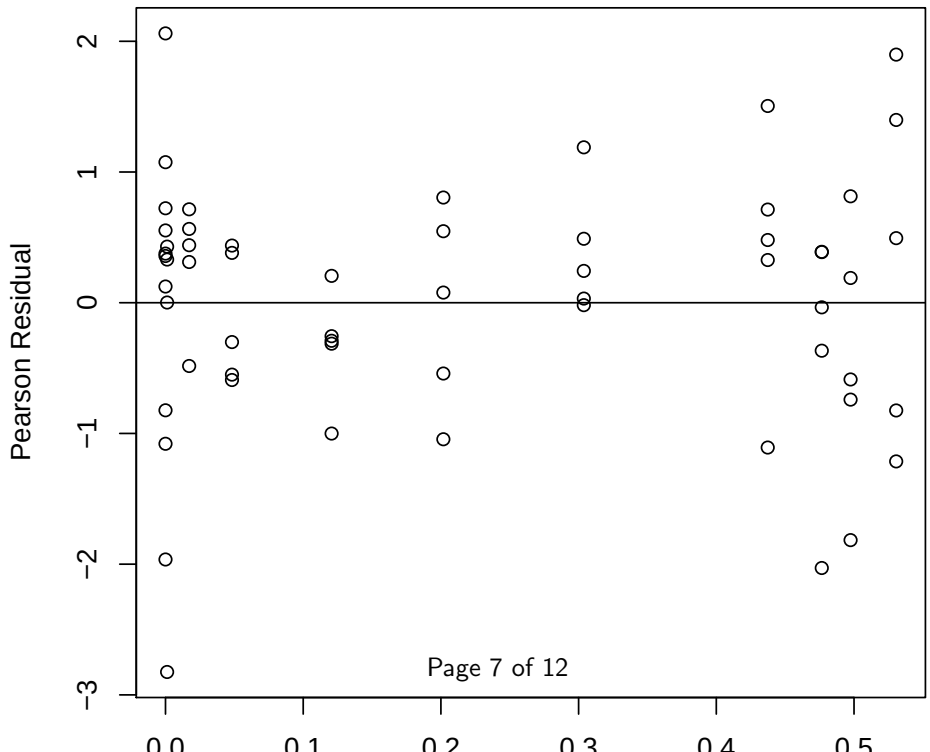


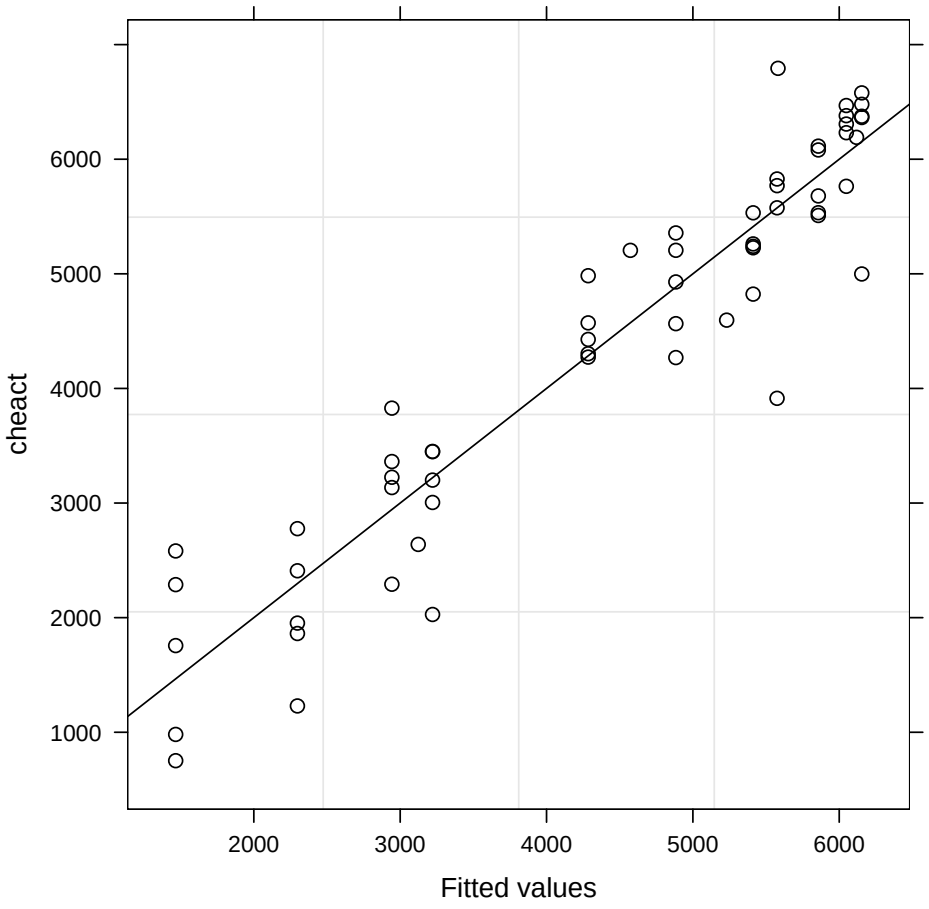
Diagnostic graphics for the model contained in `drmod1`: First, the dose-response and recovery curves with means and 95% confidence intervals from the data.

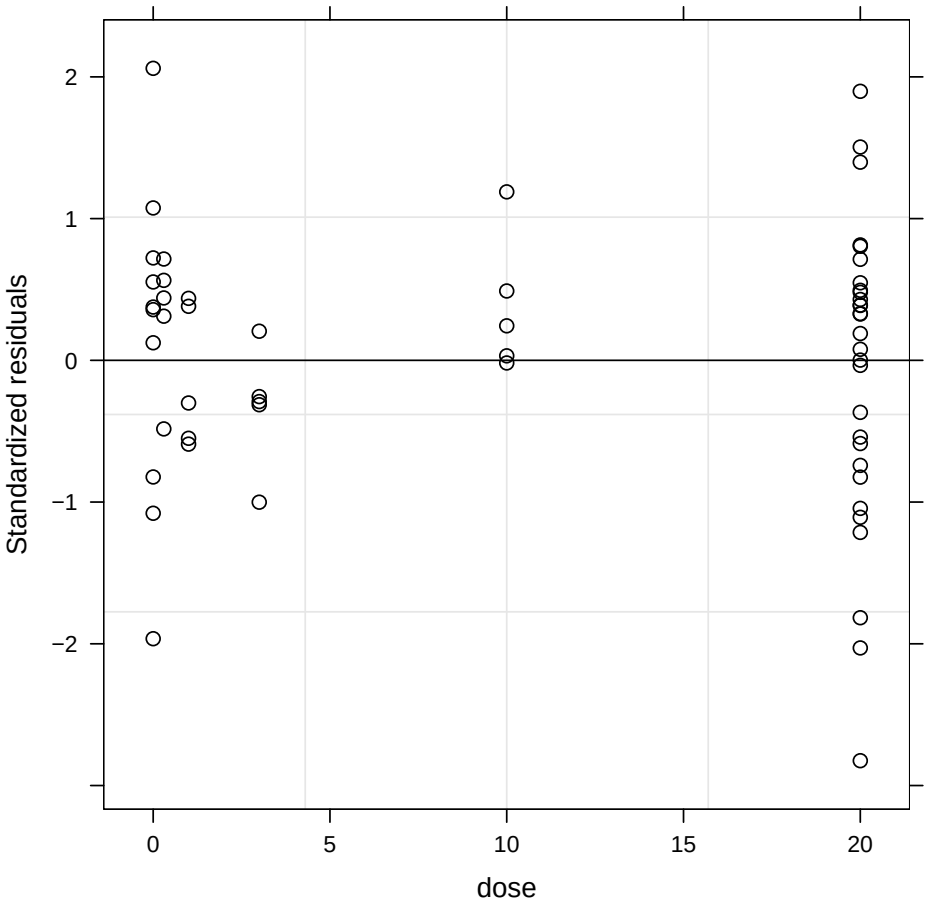


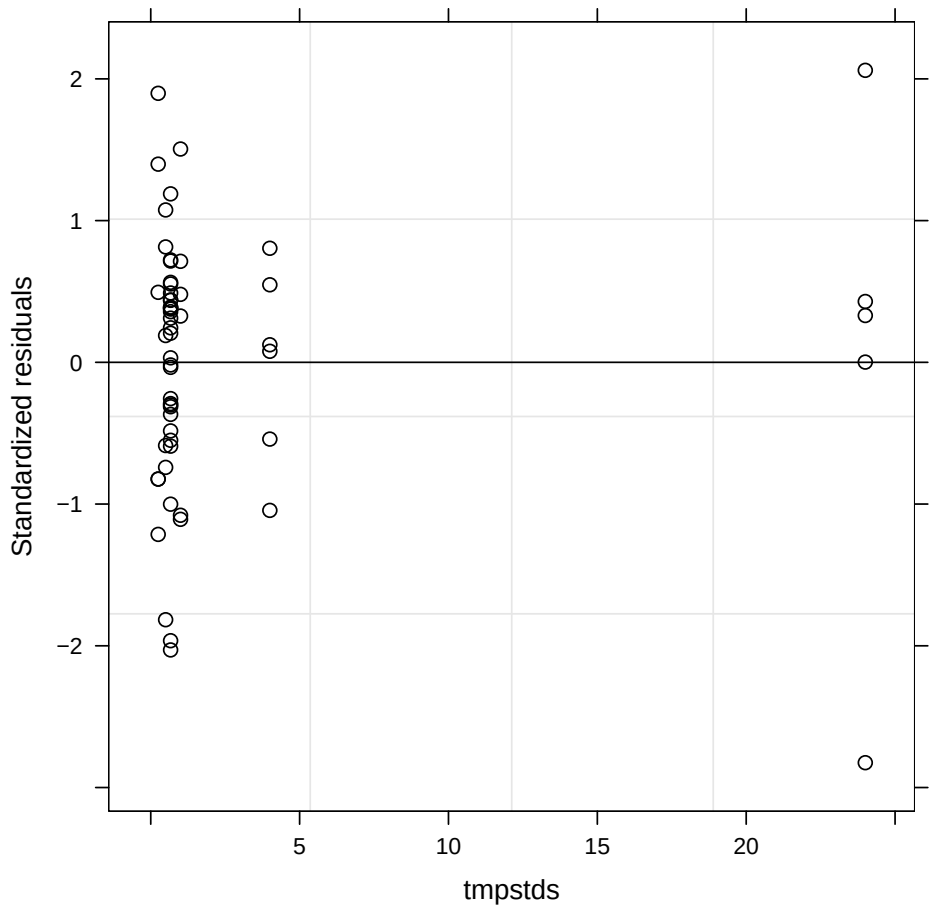
(The segmentation of the recovery curve is due to the fact that the background level varies across the recovery times).


Standardized residuals
Fitted values
2
1
0
−1
−2
2000
3000
4000
5000
6000


Pearson Residual
2
1
0
−1
−2
−3
0.0
0.1
0.2
0.3
0.4
0.5


cheact
1000
2000
3000
4000
5000
6000
2000
3000
4000
5000
6000
Fitted values


Standardized residuals
2
1
0
−1
−2
0
5
10
15
20
dose



Details of this final model fit:

```
> tTab <- summary(drmod1)$tTable
> Ints <- intervals(drmod1, which = "coef")$coef
> Ints90 <- intervals(drmod1, which = "coef", level = 0.9)$coef
> summary(drmod1)
```

```
Generalized nonlinear least squares fit
  Model: cheact ~ tcmfn4(dose, tmpstds, lA = lA, lD = lD, lg = lg, lTr = lTr)
  Data: dta
       AIC      BIC    logLik
  930.1383 950.9136 -455.0691

Coefficients:
                              Value Std.Error   t-value p-value
lA.ftime0.25               8.046596 0.1400503  57.45506  0.0000
lA.ftime0.5                8.427883 0.0942649  89.40641  0.0000
lA.ftime0.666666666666667  8.724843 0.0326988 266.82469  0.0000
lA.ftime1                  8.562292 0.0785171 109.04999  0.0000
lA.ftime4                  8.719004 0.0919609  94.81207  0.0000
lA.ftime24                 8.627232 0.0472381 182.63303  0.0000
lD                         0.739158 0.5538710   1.33453  0.1881
lg                        -0.135227 0.2492508  -0.54253  0.5899
```

```
lTr                        0.988863 0.4796483   2.06164  0.0445

 Correlation:
                          lA.0.2 lA.0.5 lA.0.6 lA.ft1 lA.ft4 lA.f24 lD
lA.ftime0.5                0.149
lA.ftime0.666666666666667  0.060  0.081
lA.ftime1                  0.062  0.123  0.080
lA.ftime4                 -0.170 -0.082  0.019  0.233
lA.ftime24                -0.014 -0.009  0.000  0.012  0.048
lD                        -0.030 -0.024 -0.757  0.011  0.077  0.005
lg                         0.071  0.084 -0.702  0.055 -0.046 -0.005  0.963
lTr                       -0.247 -0.160 -0.011  0.209  0.848  0.057  0.095
                          lg
lA.ftime0.5
lA.ftime0.666666666666667
lA.ftime1
lA.ftime4
lA.ftime24
lD
lg
lTr                       -0.084

Standardized residuals:
       Min         Q1        Med         Q3        Max
-2.8246862 -0.5456248  0.2055985  0.4917340  2.0605022

Residual standard error: 588.0752
Degrees of freedom: 59 total; 50 residual
```

3 Summary

The critical estimates from this analysis are listed below. They are printed with greater than usual precision, in case they are to be used in further computation. For reporting, round to two or three significant digits. BMD has units mg/kg, and times are in hours.

species RAT

mrid Padilla

`lD` **(standard error)** 0.739158018885824 (0.553871022464743)

BMD (95% CI) 2.09417151910424 (0.688441680047657, 6.37026269403646)

BMDL, the one-sided lower 95% CL 0.827722594975338

`lTr (standard error)` 0.988863323871846 (0.479648325315642)

Recovery Half-life dose: [0, 10 (95% CI)] 2.68817714829209 (1.02578821719772, 7.04462798406946)

Save everything:

```
> save.image(file = "RatBrainDR.RData")
```

Save the results for incorporating into a database:

```
> propoxur.oral.brain <- list(mrid = "Padilla", species = "RAT",
+     BMDs = list(male = list(lD = tTab["lD", 1], lD.se = tTab["lD",
+         2], BMD = exp(Ints["lD", 2]), BMD.CI = exp(Ints["lD",
+         c(1, 3)]), BMDL = exp(Ints90["lD", 1]))), HalfLives = list(male = list(lTr = tTab["lTr",
+         "Value"], lTr.se = tTab["lTr", "Std.Error"], Tr = exp(Ints["lTr",
+         "est."]), Tr.CI = exp(Ints["lTr", c("lower", "upper")]))))
> save(propoxur.oral.brain, file = file.path("..", "..", "01Summaries",
+     "propoxur.oral.brain.RData"))
```