

Mlfuns Sample Script

Phase I Modeling

June 8, 2011

Tim Bergsma

1 Purpose

This script runs NONMEM models and diagnostics for sample phase1 data.

2 Model Development

2.1 Set up for NONMEM run.

Listing 1:

```
> getwd()

[1] "/Users/timb/code/inst/sample/script"
```

Listing 2:

```
> library(MIfuns, lib.loc="~/Rlibs")

MIfuns 4.3.4
```

Listing 3:

```
> command <- '/common/NONMEM/nm7_osx1/test/nm7_osx1.pl'
> cat.cov='SEX'
> cont.cov=c('HEIGHT', 'WEIGHT', 'AGE')
> par.list=c('CL', 'Q', 'KA', 'V', 'V2', 'V3')
> eta.list=paste('ETA', 1:10, sep='')
```

2.2 Run NONMEM.

Listing 4:

```
> NONR (  
+   run=1005,  
+   command=command,  
+   project='../nonmem',  
+   grid=FALSE,  
+   nice=TRUE,  
+   checkrunno=FALSE,  
+   cont.cov=cont.cov,  
+   cat.cov=cat.cov,  
+   par.list=par.list,  
+   eta.list=eta.list,  
+   plotfile='../nonmem/*/diagnostics.pdf',  
+   streams='../nonmem/ctl',  
+   checksum=FALSE  
+ )  
> getwd()
```

```
[1] "/Users/timb/code/inst/sample/script"
```

Covariance succeeded on model 1005.

3 Predictive Check

3.1 Create a simulation control stream.

Convert control stream to R object.

Listing 5:

```
> ctl <- read.nmctl('../nonmem/ctl/1005.ctl')
```

Strip comments and view.

Listing 6:

```
> ctl[] <- lapply(ctl,function(rec)sub(' *;.*',' ',rec))
> ctl

[1] "$PROB 1005 phase1 2 CMT like 1004 but diff. initial on V3"
[2] "$INPUT C ID TIME SEQ=DROP EVID AMT DV SUBJ HOUR TAFD TAD LDOS MDV HEIGHT WT SEX AGE DOSE FED"
[3] "$DATA ../../data/derived/phase1.csv IGNORE=C"
[4] "$SUBROUTINE ADVAN4 TRANS4"
[5] "$PK"
[6] " CL=THETA(1)*EXP(ETA(1)) * THETA(6)**SEX * (WT/70)**THETA(7) "
[7] " V2 =THETA(2)*EXP(ETA(2)) "
[8] " KA=THETA(3)*EXP(ETA(3)) "
[9] " Q =THETA(4) "
[10] " V3=THETA(5) "
[11] " S2=V2 "
[12] " "
[13] "$ERROR"
[14] " Y=F*EXP(ERR(1)) "
[15] " IPRE=F"
[16] ""
[17] "$THETA"
[18] "(0,10,50) "
[19] "(0,10,100) "
[20] "(0,0.2, 5) "
[21] "(0,10,50) "
[22] "(0,100,1000) "
[23] "(0,1,2) "
[24] "(0,0.75,3) "
[25] ""
[26] "$OMEGA 0.09 0.09 0.09"
[27] ""
[28] ""
[29] ""
[30] ""
```

```
[31] ""
[32] "$SIGMA 0.09"
[33] ""
[34] ""
[35] ""
[36] "$ESTIMATION MAXEVAL=9999 PRINT=5 NOABORT METHOD=1 INTER MSFO=./1005.msfc"
[37] "$COV PRINT=E"
[38] "$TABLE NOPRINT FILE=./1005.tab ONEHEADER ID AMT TIME EVID PRED IPRE CWRES"
[39] "$TABLE NOPRINT FILE=./1005par.tab ONEHEADER ID TIME CL Q V2 V3 KA ETA1 ETA2 ETA3"
[40] ""
[41] ""
[42] ""
[43] ""
[44] ""
[45] ""
[46] ""
[47] ""
[48] ""
[49] ""
[50] ""
[51] ""
[52] ""
```

Fix records of interest.

Listing 7:

```
> ctl$prob
```

```
[1] "1005 phase1 2 CMT like 1004 but diff. initial on V3"
```

Listing 8:

```
> ctl$prob <- sub('1005','1105',ctl$prob)
> names(ctl)
```

```
[1] "prob"      "input"      "data"      "subroutine" "pk"  
[6] "error"     "theta"     "omega"     "sigma"      "estimation"  
[11] "cov"       "table"     "table"
```

Listing 9:

```
> names(ctl)[names(ctl)=='theta'] <- 'msfi'  
> ctl$msfi <- '../1005/1005.msf'  
> ctl$omega <- NULL  
> ctl$sigma <- NULL  
> names(ctl)[names(ctl)=='estimation'] <- 'simulation'  
> ctl$simulation <- 'ONLYSIM (1968) SUBPROBLEMS=500'  
> ctl$cov <- NULL  
> ctl$table <- NULL  
> ctl$table <- NULL  
> ctl$table <- 'DV NOHEADER NOPRINT FILE=../1105.tab FORWARD NOAPPEND'  
> write.nmctl(ctl, '../nonmem/ctl/1105.ctl')
```

3.2 Run the simulation.

This run makes the predictions (simulations).

Listing 10:

```
> NONR(  
+   run=1105,  
+   command=command,  
+   project='../nonmem',  
+   grid=FALSE,  
+   nice=TRUE,  
+   diag=FALSE,  
+   streams='../nonmem/ctl',  
+   checksum=FALSE
```

```
+ )
> getwd()

[1] "/Users/timb/code/inst/sample/script"
```

3.3 Recover and format the original dataset.

Now we fetch the results and integrate them with the other data.

Listing 11:

```
> phase1 <- read.csv('../data/derived/phase1.csv', na.strings='.')
> head(phase1)
```

	C	ID	TIME	SEQ	EVID	AMT	DV	SUBJ	HOUR	TAFD	TAD	LDOS	MDV	HEIGHT	WEIGHT
1	C	1	0.00	0	0	NA	0.000	1	0.00	0.00	NA	NA	0	174	74.2
2	<NA>	1	0.00	1	1	1000	NA	1	0.00	0.00	0.00	1000	1	174	74.2
3	<NA>	1	0.25	0	0	NA	0.363	1	0.25	0.25	0.25	1000	0	174	74.2
4	<NA>	1	0.50	0	0	NA	0.914	1	0.50	0.50	0.50	1000	0	174	74.2
5	<NA>	1	1.00	0	0	NA	1.120	1	1.00	1.00	1.00	1000	0	174	74.2
6	<NA>	1	2.00	0	0	NA	2.280	1	2.00	2.00	2.00	1000	0	174	74.2

```
SEX AGE DOSE FED SMK DS CRCN predose zerodv
1 0 29.1 1000 1 0 0 83.5 1 1
2 0 29.1 1000 1 0 0 83.5 0 0
3 0 29.1 1000 1 0 0 83.5 0 0
4 0 29.1 1000 1 0 0 83.5 0 0
5 0 29.1 1000 1 0 0 83.5 0 0
6 0 29.1 1000 1 0 0 83.5 0 0
```

Listing 12:

```
> phase1 <- phase1[is.na(phase1$C), c('SUBJ', 'TIME', 'DV')]
> records <- nrow(phase1)
> records
```

```
[1] 550
```

Listing 13:

```
> phase1 <- phase1[rep(1:records,500),]  
> nrow(phase1)
```

```
[1] 275000
```

Listing 14:

```
> phase1$SIM <- rep(1:500,each=records)  
> #head(phase1,300)  
> with(phase1,DV[SIM==1 & SUBJ==12])
```

```
[1]      NA  2.260  2.830  8.730 19.300 15.200 16.200  8.830 12.900 12.700  
[11]  7.140  5.740  1.980  0.791
```

Listing 15:

```
> with(phase1,DV[SIM==2 & SUBJ==12])
```

```
[1]      NA  2.260  2.830  8.730 19.300 15.200 16.200  8.830 12.900 12.700  
[11]  7.140  5.740  1.980  0.791
```

3.4 Recover and format the simulation results.

Listing 16:

```
> pred <- scan('../nonmem/1105/1105.tab')  
> nrow(phase1)
```

```
[1] 275000
```

Listing 17:

```
> length(pred)
```

```
[1] 275000
```

3.5 Combine the original data and the simulation data.

Listing 18:

```
> phase1$PRED <- pred  
> head(phase1)
```

	SUBJ	TIME	DV	SIM	PRED
2	1	0.00	NA	1	0.00000
3	1	0.25	0.363	1	0.17932
4	1	0.50	0.914	1	0.53642
5	1	1.00	1.120	1	0.78983
6	1	2.00	2.280	1	1.84990
7	1	3.00	1.630	1	1.96530

Listing 19:

```
> phase1 <- phase1[!is.na(phase1$DV),]  
> head(phase1)
```

	SUBJ	TIME	DV	SIM	PRED
3	1	0.25	0.363	1	0.17932
4	1	0.50	0.914	1	0.53642
5	1	1.00	1.120	1	0.78983
6	1	2.00	2.280	1	1.84990
7	1	3.00	1.630	1	1.96530
8	1	4.00	2.040	1	2.01810

3.6 Plot predictive checks.

3.6.1 Aggregate data within subject.

Since subjects may contribute differing numbers of observations, it may be useful to look at predictions from a subject-centric perspective. Therefore, we wish to calculate summary statistics for each subject, (observed and predicted) and then make obspred comparisons therewith.

Listing 20:

```
> head(phase1)

  SUBJ TIME   DV SIM   PRED
3     1  0.25 0.363   1 0.17932
4     1  0.50 0.914   1 0.53642
5     1  1.00 1.120   1 0.78983
6     1  2.00 2.280   1 1.84990
7     1  3.00 1.630   1 1.96530
8     1  4.00 2.040   1 2.01810
```

Listing 21:

```
> subject <- melt(phase1,measure.var=c('DV','PRED'))
> head(subject)

  SUBJ TIME SIM variable value
1     1  0.25  1      DV 0.363
2     1  0.50  1      DV 0.914
3     1  1.00  1      DV 1.120
4     1  2.00  1      DV 2.280
5     1  3.00  1      DV 1.630
6     1  4.00  1      DV 2.040
```

We are going to aggregate each subject's DV and PRED values using `cast()`. `cast()` likes an aggregation function that returns a list. We write one that grabs min med max for each subject, sim, and variable.

Listing 22:

```
> metrics <- function(x) list(min=min(x), med=median(x), max=max(x))
```

Now we cast, ignoring time.

Listing 23:

```
> subject <- data.frame(cast(subject, SUBJ + SIM + variable ~ ., fun=metrics))  
> head(subject)
```

	SUBJ	SIM	variable	min	med	max
1	1	1	DV	0.363000	1.6100	3.0900
2	1	1	PRED	0.179320	1.9653	5.0314
3	1	2	DV	0.363000	1.6100	3.0900
4	1	2	PRED	0.096462	3.0448	7.4728
5	1	3	DV	0.363000	1.6100	3.0900
6	1	3	PRED	0.450430	5.5284	8.7665

Note that regardless of SIM, DV (observed) is constant.

Now we melt the metrics.

Listing 24:

```
> metr <- melt(subject, measure.var=c('min', 'med', 'max'), variable_name='metric')  
> head(metr)
```

	SUBJ	SIM	variable	metric	value
1	1	1	DV	min	0.363000
2	1	1	PRED	min	0.179320
3	1	2	DV	min	0.363000
4	1	2	PRED	min	0.096462
5	1	3	DV	min	0.363000
6	1	3	PRED	min	0.450430

Listing 25:

```
> metr$value <- reapply(
+   metr$value,
+   INDEX=metr[,c('SIM', 'variable', 'metric')],
+   FUN=sort,
+   na.last=FALSE
+ )
> metr <- data.frame(cast(metr))
> head(metr)
```

	SUBJ	SIM	metric	DV	PRED
1	1	1	min	0.139	0.064213
2	1	1	med	1.025	1.943600
3	1	1	max	2.530	3.945400
4	1	2	min	0.139	0.016162
5	1	2	med	1.025	1.476300
6	1	2	max	2.530	3.463200

Listing 26:

```
> nrow(metr)
```

```
[1] 60000
```

Listing 27:

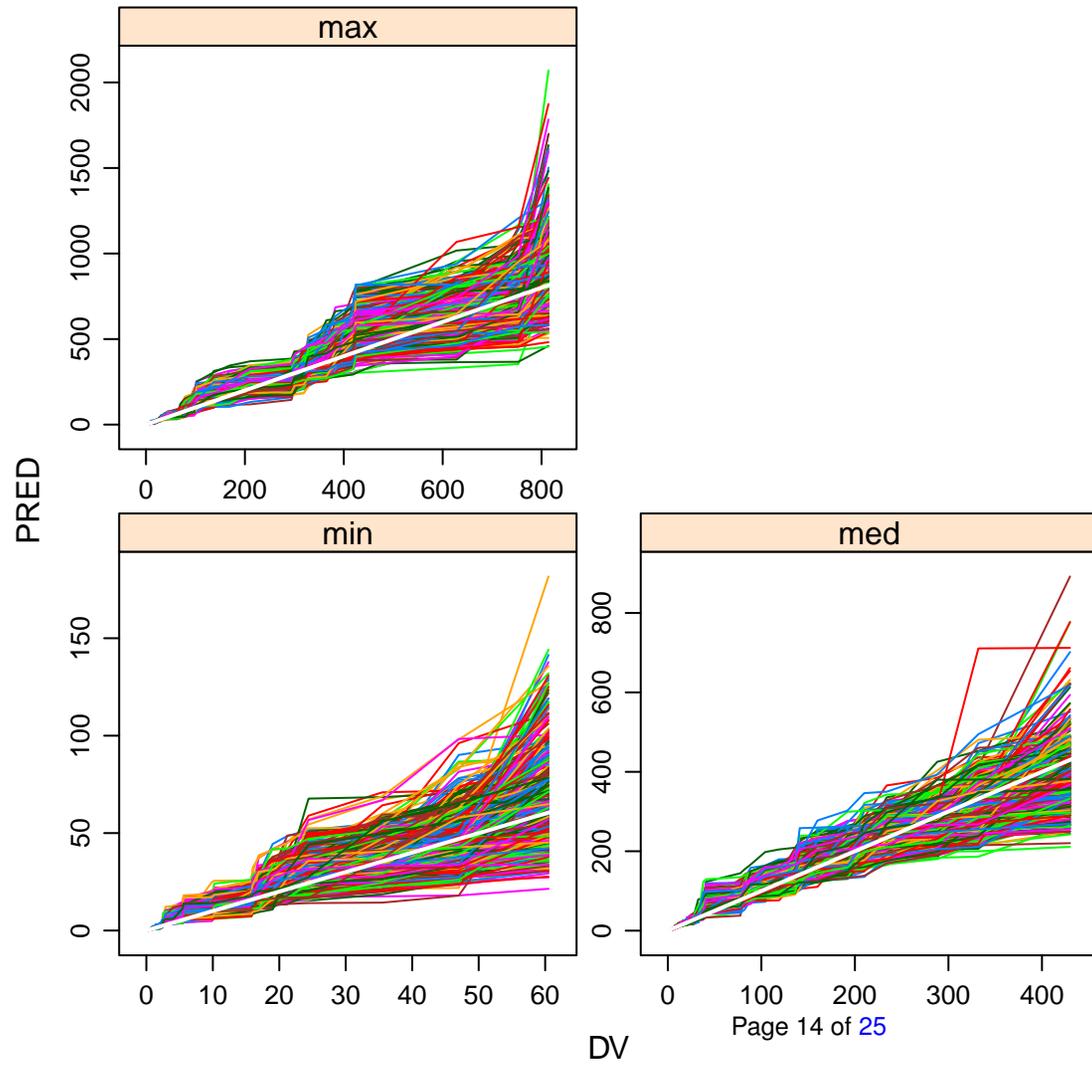
```
> metr <- metr[!is.na(metr$DV),]#maybe no NA
> nrow(metr)
```

```
[1] 60000
```

We plot using lattice.

Listing 28:

```
> print(  
+   xyplot(  
+     PRED ~ DV|metric,  
+     metr,  
+     groups=SIM,  
+     scales=list(relation='free'),  
+     type='l',  
+     panel=function(...){  
+       panel.superpose(...)  
+       panel.abline(0,1,col='white',lwd=2)  
+     }  
+   )  
+ )
```



For detail, we show one endpoint, tossing the outer 5 percent of values, and indicating quartiles.

Listing 29:

```
> med <- metr[metr$metric=='med',]  
> med$metric <- NULL  
> head(med)
```

	SUBJ	SIM	DV	PRED
2	1	1	1.025	1.943600
5	1	2	1.025	1.476300
8	1	3	1.025	1.466300
11	1	4	1.025	1.342400
14	1	5	1.025	1.362350
17	1	6	1.025	0.625815

Listing 30:

```
> trim <- inner(med, id.var=c('SIM'),measure.var=c('PRED','DV'))  
> head(trim)
```

	SIM	DV	PRED
1	1	NA	NA
2	2	NA	NA
3	3	NA	NA
4	4	NA	NA
5	5	NA	NA
6	6	NA	NA

Listing 31:

```
> nrow(trim)
```

```
[1] 20000
```

Listing 32:

```
> trim <- trim[!is.na(trim$DV),]  
> nrow(trim)
```

```
[1] 19000
```

Listing 33:

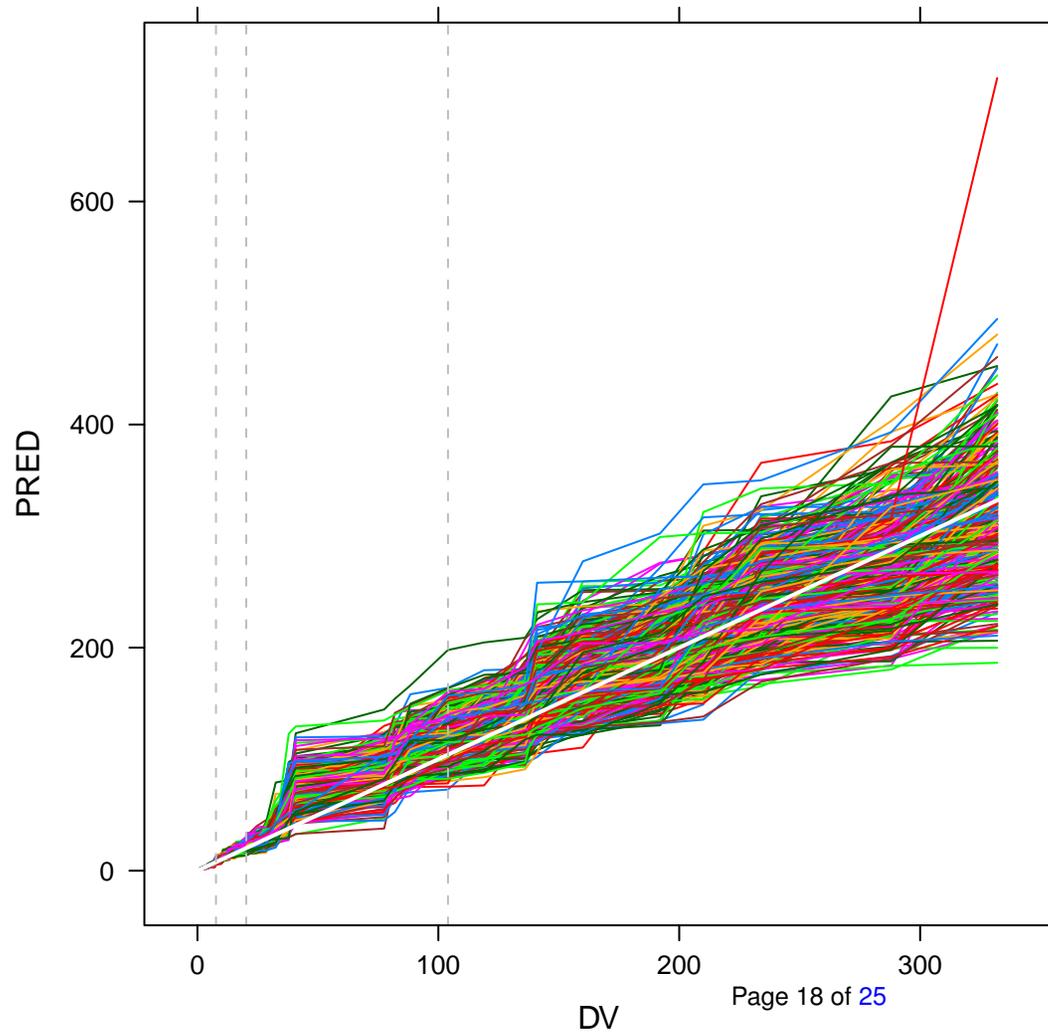
```
> head(trim)
```

```
      SIM  DV  PRED  
501    1 1.13 1.9653  
502    2 1.13 1.5989  
503    3 1.13 1.4754  
504    4 1.13 1.4074  
505    5 1.13 1.3787  
506    6 1.13 1.4753
```

Listing 34:

```
> print(  
+   xyplot(  
+     PRED ~ DV,  
+     trim,  
+     groups=SIM,  
+     type='l',  
+     panel=function(x,y,...){  
+       panel.xyplot(x=x,y=y,...)  
+       panel.abline(0,1,col='white',lwd=2)  
+       panel.abline(  
+         v=quantile(x,probs=c(0.25,0.5,0.75)),  
+         col='grey',  
+         lty=2  
+       )  
+     )  
)
```

```
+      }  
+    )  
+ )
```



We also show densityplots of predictions at those quartiles.

Listing 35:

```
> head(trim)

      SIM  DV  PRED
501    1 1.13 1.9653
502    2 1.13 1.5989
503    3 1.13 1.4754
504    4 1.13 1.4074
505    5 1.13 1.3787
506    6 1.13 1.4753
```

Listing 36:

```
> quantile(trim$DV)

      0%    25%    50%    75%   100%
1.13    7.69   20.25  104.00  332.00
```

Listing 37:

```
> molt <- melt(trim, id.var='SIM')
> head(molt)

  SIM variable value
1   1         DV  1.13
2   2         DV  1.13
3   3         DV  1.13
4   4         DV  1.13
5   5         DV  1.13
6   6         DV  1.13
```

Listing 38:

```
> quart <- data.frame(cast(molt, SIM+variable ~ ., fun=quantile, probs=c(0.25, 0.5, 0.75)))  
> head(quart)
```

	SIM	variable	X25.	X50.	X75.
1	1	DV	7.950000	20.2500	100.10000
2	1	PRED	10.329750	22.8675	91.61825
3	2	DV	7.950000	20.2500	100.10000
4	2	PRED	10.241500	23.4225	97.26175
5	3	DV	7.950000	20.2500	100.10000
6	3	PRED	8.081437	20.0330	106.59750

Listing 39:

```
> molt <- melt(quart, id.var='variable', measure.var=c('X25.', 'X50.', 'X75.'), variable_name='quartile')  
> head(molt)
```

	variable	quartile	value
1	DV	X25.	7.950000
2	PRED	X25.	10.329750
3	DV	X25.	7.950000
4	PRED	X25.	10.241500
5	DV	X25.	7.950000
6	PRED	X25.	8.081437

Listing 40:

```
> levels(molt$quartile)
```

```
[1] "X25." "X50." "X75."
```

Listing 41:

```
> levels(molt$quartile) <- c('first quartile', 'second quartile', 'third quartile')  
> head(molt)
```

```
variable    quartile    value
1      DV first quartile  7.950000
2      PRED first quartile 10.329750
3      DV first quartile  7.950000
4      PRED first quartile 10.241500
5      DV first quartile  7.950000
6      PRED first quartile  8.081437
```

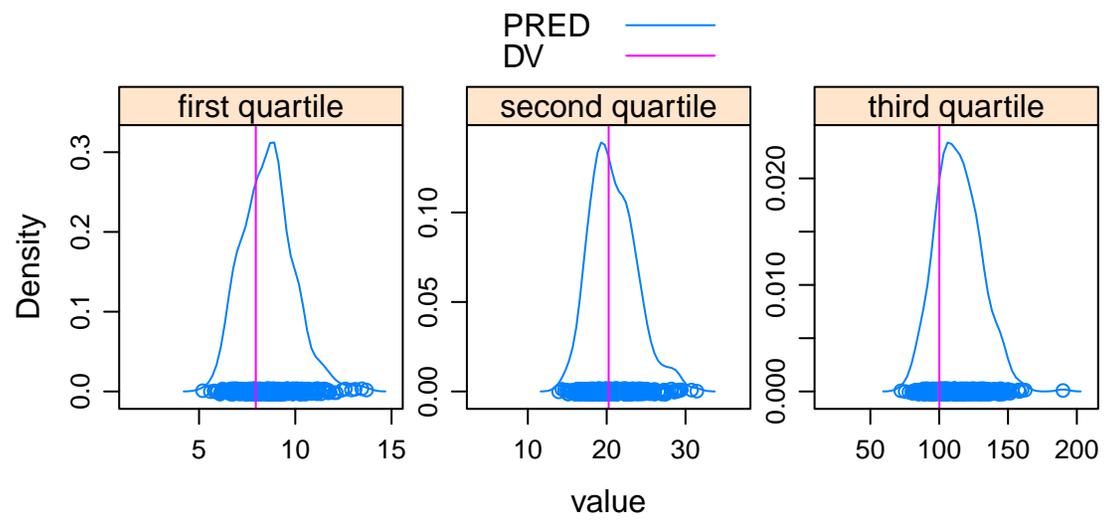
Listing 42:

```
> levels(molt$variable)
```

```
[1] "DV"  "PRED"
```

Listing 43:

```
> molt$variable <- factor(molt$variable, levels=c('PRED', 'DV'))
> print(
+   densityplot(
+     ~ value|quartile,
+     molt,
+     groups=variable,
+     layout=c(3,1),
+     scales=list(relation='free'),
+     aspect=1,
+     panel=panel.superpose,
+     panel.groups=function(x,...,group.number){
+       if(group.number==1) panel.densityplot(x,...)
+       if(group.number==2) panel.abline(v=unique(x),...)
+     },
+     auto.key=TRUE
+   )
+ )
```



4 Bootstrap Estimates of Parameter Uncertainty

4.1 Create directories.

Listing 44:

```
> getwd()

[1] "/Users/timb/code/inst/sample/script"
```

Listing 45:

```
> dir.create('../nonmem/1005.boot')
> dir.create('../nonmem/1005.boot/data')
> dir.create('../nonmem/1005.boot/ctl')
```

4.2 Create replicate control streams.

Listing 46:

```
> t <- metaSub(
+   clear(readLines('../nonmem/ctl/1005.ctl'), ';.+ ', fixed=FALSE),
+   names=1:300,
+   pattern=c(
+     '1005',
+     '.././data/derived/phase1.csv',
+     '$COV',
+     '$TABLE'
+   ),
+   replacement=c(
+     '*',
+     '../data/*.csv',
+     '$COV',
+     '$TABLE'
+   )
+ )
```

```
+ ),  
+ fixed=TRUE,  
+ out='../nonmem/1005.boot/ctl',  
+ suffix='.ctl'  
+ )
```

4.3 Create replicate data sets by resampling original.

Listing 47:

```
> bootset <- read.csv('../data/derived/phase1.csv')  
> r <- resample(  
+   bootset,  
+   names=1:300,  
+   key='ID',  
+   rekey=TRUE,  
+   out='../nonmem/1005.boot/data',  
+   stratify='SEX'  
+ )
```

4.4 Run bootstrap models.

Listing 48:

```
> NONR(  
+   run=1:300,  
+   command=command,  
+   project='../nonmem/1005.boot/',  
+   boot=TRUE,  
+   nice=TRUE,  
+   grid=FALSE,  
+   concurrent=FALSE,  
+   streams='../nonmem/1005.boot/ctl',  
+   checksum=FALSE
```

```
+ )  
> boot <- rlog(  
+   run=1:300,  
+   project='../nonmem/1005.boot',  
+   boot=TRUE,  
+   append=FALSE,  
+   tool='nm7'  
+ )  
> write.csv(boot, '../nonmem/1005.boot/log.csv')
```