

## Mlfuns Sample Script

### Covariate Plots

June 10, 2011

Tim Bergsma

## 1 Purpose

This script picks up after model.Rnw to process bootstrap results and make covariate plots.

### 1.1 Summarize bootstrap models.

Listing 1:

```
> #wait for bootstraps to finish
> getwd()
```

```
[1] "/Users/timb/project/metrum/inst/sample/script"
```

Listing 2:

```
> require(Mifuns)
```

```
Mifuns 4.3.5
```

Listing 3:

```
> boot <- read.csv('../nonmem/1005.boot/log.csv', as.is=TRUE)
> head(boot)
```

	X	tool	run	parameter	moment	value
1	1	nm7	1	ofv	minimum	2668.50793105071
2	2	nm7	1	THETA1	estimate	8.9009
3	3	nm7	1	THETA1	prse	<NA>
4	4	nm7	1	THETA1	se	<NA>
5	5	nm7	1	THETA2	estimate	20.9039
6	6	nm7	1	THETA2	prse	<NA>

Listing 4:

```
> unique (boot$parameter)

[1] "ofv"      "THETA1"    "THETA2"    "THETA3"    "THETA4"    "THETA5"
[7] "THETA6"    "THETA7"    "OMEGA1.1"  "OMEGA2.1"  "OMEGA2.2"  "OMEGA3.1"
[13] "OMEGA3.2"  "OMEGA3.3"  "SIGMA1.1"  "cov"       "prob"      "min"
[19] "data"
```

Listing 5:

```
> text2decimal (unique (boot$parameter))

[1] NA 1.0 2.0 3.0 4.0 5.0 6.0 7.0 1.1 2.1 2.2 3.1 3.2 3.3 1.1 NA NA NA NA
```

Listing 6:

```
> boot$X <- NULL
```

It looks like we have 14 estimated parameters. We will map them to the original control stream.

Listing 7:

```
> boot <- boot[!is.na(text2decimal(boot$parameter)),]
> head(boot)
```

	tool	run	parameter	moment	value
2	nm7	1	THETA1	estimate	8.9009
3	nm7	1	THETA1	prse	<NA>
4	nm7	1	THETA1	se	<NA>
5	nm7	1	THETA2	estimate	20.9039
6	nm7	1	THETA2	prse	<NA>
7	nm7	1	THETA2	se	<NA>

Listing 8:

```
> unique (boot$moment)
```

```
[1] "estimate" "prse"      "se"
```

Listing 9:

```
> unique(boot$value[boot$moment=='prse'])
```

```
[1] NA
```

prse, and therefore moment, is noninformative for these bootstraps.

Listing 10:

```
> boot <- boot[boot$moment=='estimate',]  
> boot$moment <- NULL  
> unique(boot$tool)
```

```
[1] "nm7"
```

Listing 11:

```
> boot$tool <- NULL  
> head(boot)
```

	run	parameter	value
2	1	THETA1	8.9009
5	1	THETA2	20.9039
8	1	THETA3	0.0706347
11	1	THETA4	3.36588
14	1	THETA5	116.254
17	1	THETA6	1.04097

Listing 12:

```
> unique(boot$value[boot$parameter %in% c('OMEGA2.1','OMEGA3.1','OMEGA3.2')])
```

```
[1] "0"
```

Listing 13:

```
> unique (boot$parameter[boot$value=='0'])  
  
[1] "OMEGA2.1" "OMEGA3.1" "OMEGA3.2"
```

Off-diagonals (and only off-diagonals) are noninformative.

Listing 14:

```
> boot <- boot[!boot$value=='0',]  
> any(is.na(as.numeric(boot$value)))  
  
[1] FALSE
```

Listing 15:

```
> boot$value <- as.numeric(boot$value)  
> head(boot)
```

	run	parameter	value
2	1	THETA1	8.9009000
5	1	THETA2	20.9039000
8	1	THETA3	0.0706347
11	1	THETA4	3.3658800
14	1	THETA5	116.2540000
17	1	THETA6	1.0409700

## 1.2 Restrict data to 95 percentiles.

We did 300 runs. Min and max are strongly dependent on number of runs, since with an unbounded distribution, (almost) any value is possible with enough sampling. We clip to the 95 percentiles, to give distributions that are somewhat more scale independent.

Listing 16:

```
> boot <- inner(
```

```
+ boot,  
+ preserve='run',  
+ id.var='parameter',  
+ measure.var='value'  
+ )  
> head(boot)
```

	run	parameter	value
1	1	THETA1	8.9009000
2	1	THETA2	20.9039000
3	1	THETA3	0.0706347
4	1	THETA4	3.3658800
5	1	THETA5	116.2540000
6	1	THETA6	1.0409700

Listing 17:

```
> any(is.na(boot$value))
```

```
[1] TRUE
```

Listing 18:

```
> boot <- boot[!is.na(boot$value),]
```

### 1.3 Recover parameter metadata from a specially-marked control stream.

We want meaningful names for our parameters. Harvest these from a reviewed control stream.

Listing 19:

```
> wiki <- wikipar(1005, '../nonmem')  
> wiki
```

parameter	description			
1 THETA1	apparent oral clearance			
2 THETA2	central volume of distribution			
3 THETA3	absorption rate constant			
4 THETA4	intercompartmental clearance			
5 THETA5	peripheral volume of distribution			
6 THETA6	male effect on clearance			
7 THETA7	weight effect on clearance			
8 OMEGA1.1	interindividual variability of clearance			
9 OMEGA2.2	interindividual variability of central volume			
10 OMEGA3.3	interindividual variability of Ka			
11 SIGMA1.1	proportional error			

	model	tool	run
1 CL/F (L/h) ~ theta_1 * theta_6 ^MALE * (WT/70)^theta_7 * e^eta_1	nm7	1005	
2 V_c /F (L) ~ theta_2 * (WT/70)^1 * e^eta_2	nm7	1005	
3 K_a (h^-1 ) ~ theta_3 * e^eta_3	nm7	1005	
4 Q/F (L/h) ~ theta_4	nm7	1005	
5 V_p /F (L) ~ theta_5	nm7	1005	
6 MALE_CL/F ~ theta_6	nm7	1005	
7 WT_CL/F ~ theta_7	nm7	1005	
8 IIV_CL/F ~ Omega_1.1	nm7	1005	
9 IIV_V_c /F ~ Omega_2.2	nm7	1005	
10 IIV_K_a ~ Omega_3.3	nm7	1005	
11 err_prop ~ Sigma_1.1	nm7	1005	

	estimate	prse	se
1	8.57997	9.51	0.815572
2	21.6409	9.33	2.02017
3	0.0684281	8.04	0.005504
4	3.78411	13.5	0.510932
5	107.375	15.7	16.8257
6	0.998986	13.7	0.1364
7	1.67117	21.9	0.366424
8	0.195776	23.1	0.0451412
9	0.128574	30.4	0.0391464

```
10 0.106528 25.2 0.0268981
11 0.067111 11.4 0.0076591
```

Listing 20:

```
> wiki$name <- wiki2label(wiki$model)
> wiki$estimate <- as.numeric(wiki$estimate)
> unique(wiki$parameter)

[1] "THETA1" "THETA2" "THETA3" "THETA4" "THETA5" "THETA6"
[7] "THETA7" "OMEGA1.1" "OMEGA2.2" "OMEGA3.3" "SIGMA1.1"
```

Listing 21:

```
> unique(boot$parameter)

[1] "THETA1" "THETA2" "THETA3" "THETA4" "THETA5" "THETA6"
[7] "THETA7" "OMEGA1.1" "OMEGA2.2" "OMEGA3.3" "SIGMA1.1"
```

Listing 22:

```
> boot <- stableMerge(boot, wiki[,c('parameter', 'name')])
> head(boot)
```

	run	parameter	value	name
1	1	THETA1	8.9009000	CL/F
2	1	THETA2	20.9039000	V <sub>c</sub> /F
3	1	THETA3	0.0706347	K <sub>a</sub>
4	1	THETA4	3.3658800	Q/F
5	1	THETA5	116.2540000	V <sub>p</sub> /F
6	1	THETA6	1.0409700	MALE <sub>CL</sub> /F



## 1.4 Create covariate plot.

Now we make a covariate plot for clearance. We will normalize clearance by its median (we also could have used the model estimate). We need to take cuts of weight, since we can only really show categorically-constrained distributions. Male effect is already categorical. I.e, the reference individual has median clearance, is female, and has median weight.

### 1.4.1 Recover original covariates for guidance.

Listing 23:

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

	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 24:

```
> with(covariates, constant(WEIGHT, within=ID))
```

```
[1] TRUE
```

Listing 25:

```
> covariates <- unique(covariates[,c('ID', 'WEIGHT')])  
> head(covariates)
```

```
   ID WEIGHT  
1    1   74.2  
16   2   80.3  
31   3   94.2  
46   4   85.2  
61   5   82.8  
76   6   63.9
```

Listing 26:

```
> covariates$WT <- as.numeric(covariates$WEIGHT)  
> wt <- median(covariates$WT)  
> wt
```

```
[1] 81
```

Listing 27:

```
> range(covariates$WT)
```

```
[1] 61 117
```

#### 1.4.2 Reproduce the control stream submodel for selective cuts of a continuous covariate.

In the model we normalized by 70 kg, so that cut will have null effect. Let's try 65, 75, and 85 kg. We have to make a separate column for each cut, which is a bit of work. Basically, we make two more copies of our weight effect columns, and raise our normalized cuts to those powers, effectively reproducing the submodel from the control stream.

Listing 28:

```
> head(boot)
```

```

run parameter      value      name
1  1      THETA1    8.9009000    CL/F
2  1      THETA2   20.9039000    V_c/F
3  1      THETA3    0.0706347     K_a
4  1      THETA4    3.3658800     Q/F
5  1      THETA5  116.2540000    V_p/F
6  1      THETA6    1.0409700 MALE_CL/F

```

Listing 29:

```
> unique(boot$name)
```

```

[1] "CL/F"      "V_c/F"      "K_a"        "Q/F"        "V_p/F"      "MALE_CL/F"
[7] "WT_CL/F"   "IIV_CL/F"   "IIV_V_c/F"  "IIV_K_a"    "err_prop"

```

Listing 30:

```

> clearance <- boot[boot$name %in% c('CL/F','WT_CL/F','MALE_CL/F'),]
> head(clearance)

```

```

run parameter      value      name
1  1      THETA1    8.90090    CL/F
6  1      THETA6    1.04097 MALE_CL/F
7  1      THETA7    1.04042   WT_CL/F
12 2      THETA1   10.04230    CL/F
17 2      THETA6    1.03979 MALE_CL/F
18 2      THETA7    1.27276   WT_CL/F

```

Listing 31:

```

> frozen <- data.frame(cast(clearance, run ~ name), check.names=FALSE)
> head(frozen)

```

```

run      CL/F MALE_CL/F WT_CL/F
1  1  8.90090  1.040970 1.04042

```

```
2  2 10.04230  1.039790 1.27276
3  4  9.01211  1.303000      NA
4  5  8.88781  0.900391 1.42585
5  6  7.68465  1.071680 2.05066
6  7  8.15527  0.988161 2.33554
```

Listing 32:

```
> frozen$`WT_CL/F:65` <- (65/70)**frozen$`WT_CL/F`
> frozen$`WT_CL/F:75` <- (75/70)**frozen$`WT_CL/F`
> frozen$`WT_CL/F:85` <- (85/70)**frozen$`WT_CL/F`
```

### 1.4.3 Normalize key parameter

Listing 33:

```
> #cl <- median(boot$value[boot$name=='CL/F'])
> cl <- with(wiki, estimate[name=='CL/F'])
> cl
```

```
[1] 8.57997
```

Listing 34:

```
> head(frozen)

  run      CL/F MALE_CL/F WT_CL/F WT_CL/F:65 WT_CL/F:75 WT_CL/F:85
1   1  8.90090  1.040970 1.04042  0.9257941  1.074421  1.223853
2   2 10.04230  1.039790 1.27276  0.9099900  1.091782  1.280325
3   4  9.01211  1.303000      NA      NA      NA      NA
4   5  8.88781  0.900391 1.42585  0.8997243  1.103375  1.318952
5   6  7.68465  1.071680 2.05066  0.8590138  1.151979  1.489064
6   7  8.15527  0.988161 2.33554  0.8410685  1.174844  1.573747
```

Listing 35:

```
> frozen[['CL/F']] <- frozen[['CL/F']]/c1
> head(frozen)
```

	run	CL/F	MALE_CL/F	WT_CL/F	WT_CL/F:65	WT_CL/F:75	WT_CL/F:85
1	1	1.037405	1.040970	1.04042	0.9257941	1.074421	1.223853
2	2	1.170435	1.039790	1.27276	0.9099900	1.091782	1.280325
3	4	1.050366	1.303000	NA	NA	NA	NA
4	5	1.035879	0.900391	1.42585	0.8997243	1.103375	1.318952
5	6	0.895650	1.071680	2.05066	0.8590138	1.151979	1.489064
6	7	0.950501	0.988161	2.33554	0.8410685	1.174844	1.573747

Listing 36:

```
> frozen$`WT_CL/F` <- NULL
> molten <- melt(frozen,id.var='run',na.rm=TRUE)
> head(molten)
```

	run	variable	value
1	1	CL/F	1.037405
2	2	CL/F	1.170435
3	4	CL/F	1.050366
4	5	CL/F	1.035879
5	6	CL/F	0.895650
6	7	CL/F	0.950501

#### 1.4.4 Plot.

Now we plot. We reverse the variable factor to give us top-down layout of strips.

Listing 37:

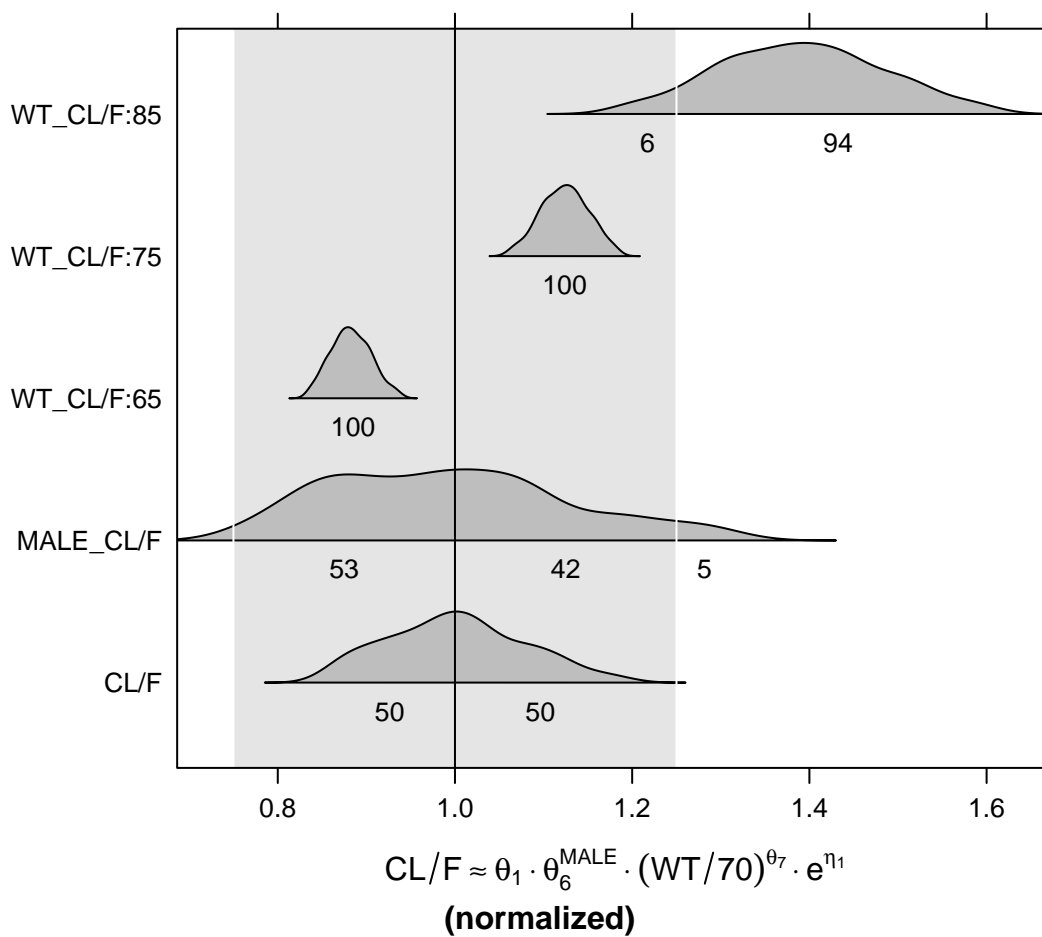
```
> levels(molten$variable)
```

```
[1] "CL/F"      "MALE_CL/F" "WT_CL/F:65" "WT_CL/F:75" "WT_CL/F:85"
```

**Listing 38:**

```
> molten$variable <- factor(molten$variable, levels=rev(levels(molten$variable)))
> print(
+   stripplot(
+     variable ~ value,
+     data=molten,
+     panel=panel.covplot,
+     xlab=parse(text=with(wiki, wiki2plotmath(noUnits(model[name=='CL/F'])))),
+     main=with(wiki, description[name=='CL/F']),
+     sub=(' (normalized) \n\n\n')
+   )
+ )
```

### apparent oral clearance



#### **1.4.5 Summarize**

We see that clearance is estimated with good precision. Ignoring outliers, there is not much effect on clearance of being male, relative to female. Increasing weight is associated with increasing clearance. There is a 93 percent probability that an 85 kg person will have at least 25 percent greater clearance than a 70 kg person.