Covariate Plots

March 20, 2013

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] "/data/metrumrg/inst/example/project/script"
```

Listing 2:

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

```
  X tool run parameter   moment           value
1 1  nm7   1       ofv  minimum 2641.7825682304
2 2  nm7   1    THETA1 estimate         9.23638
3 3  nm7   1    THETA1     prse            <NA>
4 4  nm7   1    THETA1       se            <NA>
5 5  nm7   1    THETA2 estimate         23.3418
6 6  nm7   1    THETA2     prse            <NA>
```

Listing 3:

```
> 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" "SIGMA2.1" "SIGMA2.2" "cov"
[19] "prob"      "min"       "data"
```

Listing 4:

```
> 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 2.1 2.2  NA  NA
[20]  NA  NA
```

Listing 5:

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

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

Listing 6:

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

  tool run parameter   moment    value
2  nm7   1    THETA1 estimate 9.23638
3  nm7   1    THETA1     prse     <NA>
4  nm7   1    THETA1       se     <NA>
5  nm7   1    THETA2 estimate 23.3418
6  nm7   1    THETA2     prse     <NA>
7  nm7   1    THETA2       se     <NA>
```

Listing 7:

```
> unique(boot$moment)

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

Listing 8:

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

[1] NA
```

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

Listing 9:

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

[1] "nm7"
```

Listing 10:

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

   run parameter     value
2    1    THETA1   9.23638
5    1    THETA2   23.3418
8    1    THETA3 0.0677011
11   1    THETA4   3.82773
14   1    THETA5    114.89
17   1    THETA6  0.981208
```

Listing 11:

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

```
  [1] "0.156227"    "0.0159528"   "-0.0135472"  "0.127894"    "-0.00534085"
  [6] "-0.0287155"  "0.126842"    "0.00397876"  "-0.0144261"  "0.0993418"
 [11] "-0.0131933"  "-0.048178"   "0.127265"    "-0.0130132"  "-0.0390729"
 [16] "0.139407"    "-0.0578728"  "-0.058741"   "0.0721746"   "0.00268187"
 [21] "-0.0253502"  "0.23277"     "0.00380669"  "-0.00121924" "0.159885"
 [26] "0.00848784"  "-0.0243544"  "0.0635549"   "-0.0115515"  "-0.0562696"
 [31] "0.118216"    "-0.0393444"  "-0.0351759"  "0.0901031"   "0.0135461"
 [36] "-0.0219372"  "0.187336"    "-0.0141898"  "-0.0284704"  "0.11439"
 [41] "0.0324234"   "-0.0205696"  "0.0993195"   "0.0143709"   "-0.0283724"
 [46] "0.112899"    "0.0244201"   "-0.00569855" "0.100934"    "-0.0327275"
 [51] "-0.0523527"  "0.121011"    "-0.0184053"  "-0.0492264"  "0.099605"
 [56] "-0.0043363"  "-0.051423"   "0.0899985"   "-0.0649727"  "-0.0499259"
 [61] "0.0544406"   "-0.00898663" "-0.0223262"  "0.1074"      "-0.0361188"
 [66] "-0.0522348"  "0.182506"    "0.00416665"  "-0.0190466"  "0.117682"
 [71] "-0.0385603"  "-0.0334633"  "0.150937"    "-0.0295967"  "-0.048135"
 [76] "0.0987376"   "0.019567"    "-0.00878783" "0.0851163"   "-0.048151"
 [81] "-0.0428449"  "0.126432"    "-0.00991711" "-0.0297188"  "0.0503647"
 [86] "-0.0119514"  "-0.0378401"  "0.128945"    "-0.0410842"  "-0.0595712"
 [91] "0.0523514"   "-0.0306099"  "-0.036361"   "0.128339"    "0.0100295"
 [96] "-0.0311507"  "0.0826849"   "-0.015872"   "-0.0469903"  "0.100465"
[101] "-0.0251948"  "-0.0553825"  "0.102198"    "-0.0162719"  "-0.0447151"
[106] "0.255499"    "0.00846494"  "-0.0221348"  "0.0915992"   "-0.0049864"
[111] "-0.0365348"  "0.0754003"   "-0.0124741"  "-0.0233074"  "0.104294"
[116] "-0.0437689"  "-0.056042"   "0.0774"      "-0.00284751" "-0.0348728"
[121] "0.0537363"   "-0.0441362"  "-0.0549602"  "0.0596223"   "-0.00800716"
[126] "-0.0513857"  "0.142063"    "-0.0331612"  "-0.0469179"  "0.285507"
[131] "0.076442"    "0.0113282"   "0.0877037"   "-0.034745"   "-0.0359714"
[136] "0.150039"    "-0.0510099"  "-0.0319143"  "0.14846"     "-0.0208958"
[141] "-0.0413109"  "0.137549"    "0.024375"    "-0.0114646"  "0.143612"
[146] "0.00225742"  "-0.0254575"  "0.139314"    "0.0253247"   "0.00200758"
[151] "0.125747"    "-0.015348"   "-0.031895"   "0.126632"    "-0.0147088"
[156] "-0.0194588"  "0.0469781"   "-0.0354401"  "-0.035513"   "0.0873566"
[161] "-0.0273512"  "-0.0249502"  "0.123487"    "-0.00356765" "-0.0315459"
[166] "0.171623"    "-0.00509541" "-0.00817674" "0.131233"    "0.00407373"
[171] "-0.0303432"  "0.10031"     "-0.0228062"  "-0.0333228"  "0.0796734"
[176] "0.00183569"  "-0.0297277"  "0.093994"    "-0.000523256" "-0.0366687"
[181] "0.0931635"   "-0.030158"   "-0.055576"   "0.128319"    "-0.0393971"
[186] "-0.0252056"  "0.120193"    "-0.0100479"  "-0.0245258"  "0.146148"
[191] "-0.0191463"  "-0.0350369"  "0.128157"    "0.00698355"  "-0.0199553"
[196] "0.182157"    "0.0158739"   "-0.0117383"  "0.13733"     "0.0237497"
[201] "-0.0219953"  "0.0657874"   "0.025327"    "-0.0106585"  "0.169318"
[206] "-0.0206613"  "-0.0475536"  "0.0997141"   "-0.0135418"  "-0.0535933"
[211] "0.104324"    "-0.00896306" "-0.0334962"  "0.103071"    "0.0167697"
[216] "-0.0155637"  "0.0911511"   "-0.0477166"  "-0.0566334"  "0.103853"
[221] "-0.0039753"  "-0.0335942"  "0.112202"    "-0.0215719"  "-0.0439991"
[226] "0.141921"    "0.0221883"   "-0.00842366" "0.112078"    "-0.0293556"
[231] "-0.0468418"  "0.171907"    "0.015393"    "-0.0177328"  "0.136378"
[236] "0.00779403"  "-0.0218202"  "0.150747"    "-0.00345454" "-0.0245817"
[241] "0.183948"    "0.0161263"   "-0.0243461"  "0.118834"    "0.00125101"
[246] "-0.0353765"  "0.115061"    "0.00340433"  "-0.0333913"  "0.136096"
```

```
[251] "-0.0170945"  "-0.030838"   "0.109808"    "-0.00709553" "-0.0335206"
[256] "0.160059"    "0.0248198"   "-0.0109128"  "0.112517"    "0.0159173"
[261] "-0.00938595" "0.108432"    "-0.0265611"  "-0.0416907"  "0.11617"
[266] "-0.0121557"  "-0.0331705"  "0.137285"    "-0.0474463"  "-0.0449732"
[271] "0.145432"    "0.0268214"   "-0.0298667"  "0.0813123"   "0.0114738"
[276] "-0.0159414"  "0.119721"    "-0.0135702"  "-0.0343614"  "0.104604"
[281] "-0.0263675"  "-0.0488223"  "0.141042"    "-0.0243538"  "-0.0402116"
[286] "0.12291"     "0.00758508"  "-0.0339353"  "0.0656227"   "-0.0333673"
[291] "-0.0395299"  "0.124603"    "0.0263632"   "-0.002676"   "0.083981"
[296] "-0.00102412" "-0.0266232"  "0.123294"    "0.00381394"  "-0.0244724"
[301] "0.17292"     "-0.00970248" "-0.0127962"  "0.181886"    "-0.017694"
[306] "-0.0286785"  "0.129665"    "0.0264432"   "-0.0309526"  "0.151022"
[311] "-0.0078379"  "-0.0324558"  "0.146179"    "-0.0207242"  "-0.0279331"
[316] "0.133603"    "-0.00171313" "-0.0386058"  "0.0915628"   "-0.0110001"
[321] "-0.0477608"  "0.147787"    "-0.00746187" "-0.0447148"  "0.0857886"
[326] "-0.0303487"  "-0.0479556"  "0.104093"    "-0.00825775" "-0.0320089"
[331] "0.121367"    "0.0139687"   "-0.0353464"  "0.102999"    "-0.00424476"
[336] "-0.0363616"  "0.0834914"   "-0.022373"   "-0.0540236"  "0.107415"
[341] "-0.0054095"  "-0.0475559"  "0.139837"    "-0.0335157"  "-0.0518725"
[346] "0.182958"    "0.0199211"   "-0.0143953"  "0.103835"    "-0.0375877"
[351] "-0.0382998"  "0.125518"    "-0.0142602"  "-0.0335761"  "0.0931846"
[356] "0.0197082"   "-0.0298824"  "0.0740316"   "-0.0221008"  "-0.0413043"
[361] "0.113781"    "-0.026497"   "-0.0507665"  "0.0942481"   "-0.0606044"
[366] "-0.0412218"  "0.278396"    "-0.00706858" "-0.0181813"  "0.0902824"
[371] "-0.0168196"  "-0.0278898"  "0.0657135"   "-0.00117424" "-0.0516082"
[376] "0.147592"    "-0.00400659" "-0.0187836"  "0.127401"    "-0.0239393"
[381] "-0.0362752"  "0.193438"    "0.0288493"   "-0.0254224"  "0.170627"
[386] "-0.0189935"  "-0.0382035"  "0.098659"    "-0.00946011" "-0.03748"
[391] "0.0887404"   "-0.0239842"  "-0.047071"   "0.102068"    "-0.00680018"
[396] "-0.0475187"  "0.121779"    "-0.0169515"  "-0.0411449"  "0.0904357"
[401] "0.00887722"  "-0.0219956"  "0.0977255"   "-0.0414795"  "-0.05333"
[406] "0.0898638"   "-0.0668332"  "-0.0493432"  "0.10406"     "-0.00633542"
[411] "-0.0501093"  "0.144721"    "-0.0146678"  "-0.0438669"  "0.0937251"
[416] "-0.022682"   "-0.0274813"  "0.119161"    "-0.009861"   "-0.0323643"
[421] "0.0963911"   "-0.0138369"  "-0.0418917"  "0.0879488"   "0.0135003"
[426] "-0.0188883"  "0.0662287"   "-0.00166997" "-0.0297563"  "0.218618"
[431] "-0.0249036"  "-0.0311848"  "0.195222"    "-0.00277758" "-0.0222522"
[436] "0.10353"     "0.0143966"   "-0.0318285"  "0.0677653"   "-0.0285179"
[441] "-0.044494"   "0.0728236"   "-0.00964799" "-0.039288"   "0.117708"
[446] "0.00409308"  "-0.0315527"  "0.206349"    "-0.0185911"  "-0.0470862"
[451] "0.121647"    "0.0019113"   "-0.0255574"  "0.104044"    "-0.00463571"
[456] "-0.0539604"  "0.122352"    "-0.0189161"  "-0.0380843"  "0.0905268"
[461] "-0.0464468"  "-0.0424179"  "0.0922275"   "-0.013442"   "-0.0288039"
[466] "0.101624"    "-0.0223923"  "-0.026549"   "0.160719"    "-0.0280859"
[471] "-0.0481335"  "0.142071"    "0.00578165"  "-0.0261225"  "0.177004"
[476] "-0.00864672" "-0.0107352"  "0.123245"    "0.0262577"   "-0.0134449"
[481] "0.122679"    "-0.0132769"  "-0.0250786"  "0.0798768"   "-0.0109221"
[486] "-0.0425768"  "0.119248"    "-0.0220389"  "-0.0472546"  "0.0878249"
[491] "-0.0246585"  "-0.00750851" "0.12965"     "-0.0197424"  "-0.0318449"
[496] "0.107137"    "-0.0106163"  "-0.029723"   "0.0160289"   "-0.036996"
```

```
[501] "-0.0287133"   "0.0680665"    "-0.0103254"   "-0.0384736"   "0.083917"
[506] "-0.031936"    "-0.0476524"   "0.095128"     "-0.0427206"   "-0.0342096"
[511] "0.114886"     "-0.0244739"   "-0.0306805"   "0.0818132"    "-0.00800649"
[516] "-0.0428066"   "0.139296"     "0.00631299"   "-0.0208287"   "0.153918"
[521] "0.0041499"    "-0.0213978"   "0.100663"     "0.00193413"   "-0.0308643"
[526] "0.120771"     "0.00137212"   "0.00130525"   "0.150564"     "-0.0229715"
[531] "-0.0340155"   "0.129644"     "-0.00692566"  "-0.0387418"   "0.077383"
[536] "-0.0286357"   "-0.0324173"   "0.091335"     "-0.0134751"   "-0.027486"
[541] "0.0655578"    "-0.0191627"   "-0.0428257"   "0.167989"     "-0.00659166"
[546] "-0.0331723"   "0.124987"     "-0.0176043"   "-0.0390198"   "0.117923"
[551] "-0.0374096"   "-0.0307362"   "0.106677"     "-0.00683247"  "-0.0411212"
[556] "0.322303"     "0.0346376"    "-0.00832182"  "0.0810913"    "0.0125724"
[561] "-0.0318828"   "0.0949605"    "-0.0582629"   "-0.0510204"   "0.169111"
[566] "-0.0257637"   "-0.0328185"   "0.143579"     "0.0100669"    "-0.0144292"
[571] "0.0921006"    "-0.0193888"   "-0.0395264"   "0.0964377"    "0.0233946"
[576] "-0.0126189"   "0.0943814"    "0.0228365"    "-0.0351794"   "0.0935161"
[581] "-0.0335367"   "-0.0361671"   "0.131486"     "0.0131206"    "-0.0340262"
[586] "0.108542"     "-0.0522091"   "-0.0373783"   "0.0896632"    "-0.0335599"
[591] "-0.0268513"   "0.0841944"    "-0.00105725"  "-0.0394727"   "0.09836"
[596] "-0.038718"    "-0.0441725"   "0.102277"     "-0.0232714"   "-0.0428383"
[601] "0.0884494"    "0.00384483"   "-0.0372837"   "0.0811651"    "-0.0100908"
[606] "-0.0497528"   "0.159658"     "-0.0504886"   "-0.0505731"   "0.0838126"
[611] "-0.00801171"  "-0.0267642"   "0.0645074"    "-0.00171628"  "-0.0393291"
[616] "0.167553"     "0.013042"     "-0.0181085"   "0.0753708"    "-0.0191694"
[621] "-0.0477448"   "0.161165"     "0.0108868"    "-0.0147942"   "0.164829"
[626] "-0.00518677"  "-0.0291827"   "0.124685"     "-0.0340624"   "-0.0437537"
[631] "0.130915"     "-0.0414497"   "-0.0351812"   "0.122726"     "-0.0141264"
[636] "-0.0222556"   "0.174769"     "0.0225853"    "-0.00508478"  "0.162744"
[641] "-0.0214249"   "-0.026872"    "0.145307"     "0.0029906"    "-0.0312918"
[646] "0.141633"     "-0.0158909"   "-0.0278366"   "0.0914485"    "0.012461"
[651] "-0.0296438"   "0.0724357"    "-0.0338341"   "-0.0518615"   "0.148088"
[656] "0.0161538"    "0.00113638"   "0.0774178"    "-0.0380824"   "-0.0494172"
[661] "0.104704"     "-0.0119937"   "-0.0478963"   "0.0813489"    "-0.0400834"
[666] "-0.054908"    "0.110295"     "-0.0015311"   "-0.0391485"   "0.0908491"
[671] "0.000419918"  "-0.013846"    "0.168641"     "-0.020853"    "-0.0407037"
[676] "0.15126"      "-0.00923592"  "-0.0485535"   "0.0810875"    "-0.0430208"
[681] "-0.0578183"   "0.147601"     "0.0140929"    "0.00317507"   "0.140425"
[686] "-0.00298776"  "-0.0350169"   "0.146948"     "0.00117569"   "-0.0294128"
[691] "0.163811"     "0.0441442"    "0.000395307"  "0.0894596"    "-0.0218916"
[696] "-0.0354665"   "0.0990998"    "-0.0197941"   "-0.0258214"   "0.0767375"
[701] "-0.0376434"   "-0.0371933"   "0.127169"     "-0.0203791"   "-0.0359885"
[706] "0.155938"     "-0.0141085"   "-0.0369582"   "0.0902598"    "-0.0148727"
[711] "-0.033949"    "0.100909"     "0.012865"     "-0.0366748"   "0.0479753"
[716] "0.00455662"   "-0.0410609"   "0.0728055"    "-0.0307688"   "-0.0406978"
[721] "0.147163"     "-0.0173894"   "-0.0401154"   "0.121854"     "0.0152048"
[726] "-0.0207313"   "0.140714"     "-0.0209404"   "-0.0369639"   "0.167167"
[731] "0.00685101"   "-0.0217506"   "0.0750764"    "-0.00255085"  "-0.0436457"
[736] "0.107334"     "-0.00373994"  "-0.0280744"   "0.0902622"    "-0.00762635"
[741] "-0.0342191"   "0.137109"     "0.0265652"    "-0.00316519"  "0.158305"
[746] "-0.0134835"   "-0.0352053"   "0.112675"     "-0.00772547"  "-0.0375772"
```

```
[751] "0.14672"     "-0.0264658"  "-0.0414208"  "0.175412"    "0.00889336"
[756] "-0.00502695" "0.114916"    "0.00941087"  "-0.03346"    "0.0532543"
[761] "-0.0149894"  "-0.0397522"  "0.125328"    "-0.0156381"  "-0.0229857"
[766] "0.137535"    "-0.0147341"  "-0.0384469"  "0.1772"      "0.0018514"
[771] "-0.00759116" "0.0742238"   "-0.00393504" "-0.0355385"  "0.136119"
[776] "0.0253505"   "-0.00123372" "0.0891006"   "-0.0290592"  "-0.0393014"
[781] "0.107136"    "-0.0275636"  "-0.0289668"  "0.134234"    "-0.0282785"
[786] "-0.0457407"  "0.132621"    "0.00465362"  "-0.0260594"  "0.154333"
[791] "-0.0263149"  "-0.0423259"  "0.140013"    "-0.0177322"  "-0.0286442"
[796] "0.137841"    "-0.0188566"  "-0.024523"   "0.0952979"   "0.00963383"
[801] "-0.0239677"  "0.124585"    "-0.000733301" "-0.0370763" "0.0637665"
[806] "-0.0110663"  "-0.0396288"  "0.158731"    "-3.65028e-05" "-0.042261"
[811] "0.110566"    "0.014848"    "-0.027753"   "0.128956"    "-0.0245695"
[816] "-0.0575748"  "0.20035"     "0.0322329"   "0.00474023"  "0.133611"
[821] "-0.0343994"  "-0.0409556"  "0.131331"    "0.00159212"  "-0.0263606"
[826] "0.105567"    "0.0300378"   "-0.0267307"  "0.0676963"   "-0.0137024"
[831] "-0.0164749"  "0.104278"    "-0.0723554"  "-0.0322967"  "0.03446"
[836] "-0.0254784"  "-0.0577979"  "0.125957"    "0.00959378"  "-0.0259816"
[841] "0.0696536"   "-0.0244617"  "-0.0477368"  "0.118004"    "-0.00436055"
[846] "-0.0379923"  "0.130931"    "-0.00321759" "-0.0351325"  "0.145661"
[851] "-0.0479728"  "-0.0427793"  "0.112976"    "-0.0100286"  "-0.0255242"
[856] "0.0872527"   "-0.00713972" "-0.0251535"  "0.0871989"   "-0.0165869"
[861] "-0.0434855"  "0.128209"    "-0.00971458" "-0.0353069"  "0.0970421"
[866] "0.0186014"   "-0.0274255"  "0.169576"    "-0.0236375"  "-0.0318608"
[871] "0.221648"    "-0.0163224"  "-0.0339322"  "0.141422"    "0.00924223"
[876] "-0.0327104"  "0.0822966"   "-0.00331199" "-0.0499057"  "0.149429"
[881] "0.00551382"  "-0.0263345"  "0.144499"    "-0.0165897"  "-0.0155797"
[886] "0.119238"    "-0.0248579"  "-0.0271283"  "0.165497"    "0.0148976"
[891] "-0.0316042"  "0.107062"    "-0.0333402"  "-0.036574"   "0.101135"
[896] "0.00445035"  "-0.0566516"  "0.182794"    "0.0148809"   "-0.012067"
```

Listing 12:

```
> unique(boot$parameter[boot$value=='0'])
```

```
[1] "SIGMA2.1"
```

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

Listing 13:

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

```
[1] FALSE
```

Listing 14:

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

```
   run parameter       value
2    1    THETA1   9.2363800
5    1    THETA2  23.3418000
8    1    THETA3   0.0677011
11   1    THETA4   3.8277300
14   1    THETA5 114.8900000
17   1    THETA6   0.9812080
```

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 15:

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

```
  run parameter       value
1   1    THETA1   9.2363800
2   1    THETA2  23.3418000
3   1    THETA3   0.0677011
4   1    THETA4   3.8277300
5   1    THETA5 114.8900000
6   1    THETA6   0.9812080
```

Listing 16:

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

```
[1] TRUE
```

Listing 17:

```
> 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 18:

```
> wiki <- wikitab(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	OMEGA2.1	interindividual clearance-volume covariance
10	OMEGA2.2	interindividual variability of central volume
11	OMEGA3.1	interindividual clearance-Ka covariance
12	OMEGA3.2	interindividual volume-Ka covariance
13	OMEGA3.3	interindividual variability of Ka
14	SIGMA1.1	proportional error
15	SIGMA2.2	additive 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	cov_CL,V ~ Omega_2.1	nm7	1005
10	IIV_V_c /F ~ Omega_2.2	nm7	1005
11	cov_CL,Ka ~ Omega_3.1	nm7	1005
12	cov_V,Ka ~ Omega_3.2	nm7	1005
13	IIV_K_a ~ Omega_3.3	nm7	1005
14	err_prop ~ Sigma_1.1	nm7	1005
15	err_add ~ Sigma_2.2	nm7	1005

	estimate	prse	se
1	16	15	13
2	14	14	15
3	6	13	1
4	15	3	12
5	12	5	16
6	11	2	10
7	13	8	11
8	10	6	8
9	8	7	7
10	7	9	6
11	2	4	5
12	3	12	3
13	4	11	4
14	5	1	2
15	9	10	9

Listing 19:

```
> 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" "OMEGA2.1" "OMEGA2.2" "OMEGA3.1" "OMEGA3.2"
[13] "OMEGA3.3" "SIGMA1.1" "SIGMA2.2"
```

Listing 20:

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

```
 [1] "THETA1"    "THETA2"    "THETA3"    "THETA4"    "THETA5"    "THETA6"
 [7] "THETA7"    "OMEGA1.1" "OMEGA2.1" "OMEGA2.2" "OMEGA3.1" "OMEGA3.2"
[13] "OMEGA3.3" "SIGMA1.1" "SIGMA2.2"
```

Listing 21:

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

```
  run parameter       value      name
1   1    THETA1   9.2363800      CL/F
2   1    THETA2  23.3418000     V_c/F
3   1    THETA3   0.0677011       K_a
4   1    THETA4   3.8277300       Q/F
5   1    THETA5 114.8900000     V_p/F
6   1    THETA6   0.9812080 MALE_CL/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 22:

```
> covariates <- read.csv('../data/derived/phase1.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      0
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 23:

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

```
[1] TRUE
```

Listing 24:

```
> 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 25:

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

```
[1] 81
```

Listing 26:

```
> 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 27:

```
> head(boot)
```

```
  run parameter       value      name
1   1    THETA1   9.2363800      CL/F
2   1    THETA2  23.3418000     V_c/F
3   1    THETA3   0.0677011       K_a
4   1    THETA4   3.8277300       Q/F
5   1    THETA5 114.8900000     V_p/F
6   1    THETA6   0.9812080 MALE_CL/F
```

Listing 28:

```
> 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"   "cov_CL,V"   "IIV_V_c/F"  "cov_CL,Ka"  "cov_V,Ka"
[13] "IIV_K_a"    "err_prop"   "err_add"
```

Listing 29:

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

```
   run parameter    value      name
1    1    THETA1 9.236380      CL/F
6    1    THETA6 0.981208 MALE_CL/F
7    1    THETA7 1.597710   WT_CL/F
16   2    THETA1 8.721030      CL/F
21   2    THETA6 0.941112 MALE_CL/F
22   2    THETA7 1.376750   WT_CL/F
```

Listing 30:

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

```
  run     CL/F MALE_CL/F  WT_CL/F
1   1  9.23638  0.981208 1.597710
2   2  8.72103  0.941112 1.376750
3   3  9.65402  1.011370 1.408970
4   4 10.39540  0.918476 0.715433
5   5 10.04480  0.863337 1.386840
6   6 10.13660  1.020360 0.580109
```

Listing 31:

```
> 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 32:

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

```
[1] 16
```

Listing 33:

```
> 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  9.23638  0.981208 1.597710  0.8883379   1.116536   1.363705
2   2  8.72103  0.941112 1.376750  0.9030041   1.099643   1.306438
3   3  9.65402  1.011370 1.408970  0.9008505   1.102091   1.314636
4   4 10.39540  0.918476 0.715433  0.9483617   1.050598   1.149016
5   5 10.04480  0.863337 1.386840  0.9023292   1.100409   1.309000
6   6 10.13660  1.020360 0.580109  0.9579203   1.040835   1.119220
```

Listing 34:

```
> frozen[['CL/F']] <- frozen[['CL/F']]/cl
> 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 0.5772738  0.981208 1.597710  0.8883379   1.116536   1.363705
2   2 0.5450644  0.941112 1.376750  0.9030041   1.099643   1.306438
3   3 0.6033762  1.011370 1.408970  0.9008505   1.102091   1.314636
4   4 0.6497125  0.918476 0.715433  0.9483617   1.050598   1.149016
5   5 0.6278000  0.863337 1.386840  0.9023292   1.100409   1.309000
6   6 0.6335375  1.020360 0.580109  0.9579203   1.040835   1.119220
```

Listing 35:

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

```
  run variable     value
1   1     CL/F 0.5772738
2   2     CL/F 0.5450644
3   3     CL/F 0.6033762
4   4     CL/F 0.6497125
5   5     CL/F 0.6278000
6   6     CL/F 0.6335375
```

1.4.4 Plot.

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

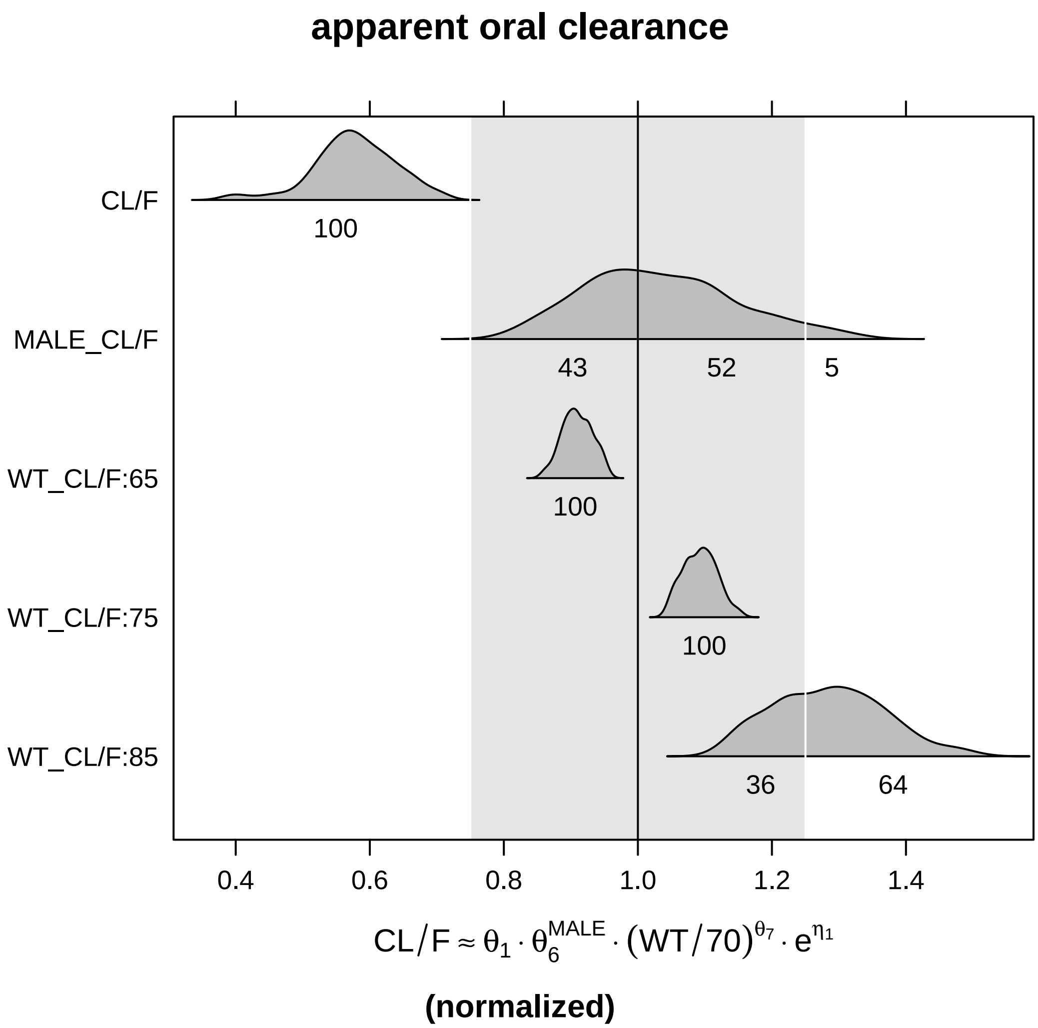
Listing 36:

```
> levels(molten$variable)

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

Listing 37:

```
> 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')
+   )
+ )
```



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.