Notes on RNMImport

Mango Solutions

August 15, 2014

1 Introduction

This set of notes is a brief overview of the RNMImport package. At the moment (version 4.0-x), these are quite terse, but will be expanded upon in later releases. This is meant to give only a basic idea of how the package works.

```
> require(RNMImport)
```

```
Full path to configuration file:
 C:/Users/jli.MANGO/AppData/Local/Temp/RtmpIN04KF/Rinst1d043fd8510a/RNMImport/configdata/
```

2 Importing runs

The main command for importing a NONMEM run is `importNm`, which works with a control file and a path. List files names are automatically deduced from allowable file extensions (see meta data section below), but can be passed explicitly.

```
> # Import an example run
> runPath <- system.file(package = "RNMImport", "examples/theoph")
> # List file deduced automatically
> run <- importNm(conFile = "theoph.con", path = runPath)
> print(run)
```

```
Control file:
            size mode               mtime               ctime
controlFile  565  666 2014-08-15 12:02:34 2014-08-15 12:02:34
                          atime exe
controlFile 2014-08-15 12:02:34  no

controlFile C:/Users/jli.MANGO/AppData/Local/Temp/RtmpIN04KF/Rinst1d043fd8510a/RNMImport/
Output report file:
           size mode               mtime               ctime
reportFile 6238  666 2014-08-15 12:02:34 2014-08-15 12:02:34
                         atime exe
reportFile 2014-08-15 12:02:34  no

reportFile C:/Users/jli.MANGO/AppData/Local/Temp/RtmpIN04KF/Rinst1d043fd8510a/RNMImport/e
Number of problems: 1
Problems:
*******************
Problem  1

Standard NONMEM problem:
###############
```

```
Problem statement:  Analysis of one compartment model for theophyline data
Data file: ./data.csv
Input table dimensions:
144 7
Input table columns:
ID DOSE WT TIME DV MDV SMOK
PRED:
[1] "KA = THETA(1) + ETA(1) "
[2] "KE = THETA(2) + ETA(2) "
[3] "CL = THETA(3) + ETA(3) "
[4] "F = (DOSE*KE*KA*WT) /(CL * (KA-KE) ) * (EXP(-KE*TIME) - EXP(-KA*TIME) ) "
[5] "Y = F + EPS(1) "
[6] "IPRED = F"
[7] "IRES = (F - DV) "

Parameter estimates:
###############
THETAs:
THETA1 THETA2 THETA3
2.5500 0.0758 2.5600
OMEGAs:
       OMEGA1 OMEGA2 OMEGA3
OMEGA1   4.77  0e+00  0.000
OMEGA2   0.00  4e-05  0.000
OMEGA3   0.00  0e+00  0.203
SIGMAs:
[1] 0.456
Output table files: theoph.out
Output table dimensions:
144 9
Output table columns:
ID TIME DV IPRED DOSE WT IRES ETA1 ETA2
```

```
> print(class(run))
```

```
[1] "NMRun"
attr(,"package")
[1] "RNMImport"
```

When calling importNm, a control file, “list” file and output table files are all required, else an error is generated. Input data tables are optional, but a warning is omitted if it is missing. The returned object is of class NMRun, whose declaration is given below:

```
> print(getClass("NMRun"))
```

```
Class "NMRun" [package "RNMImport"]

Slots:

Name:      controlText      reportText  nmVersionMajor
Class:       character       character       character

Name:   nmVersionMinor controlComments controlFileInfo
Class:         numeric       character      data.frame

Name:   reportFileInfo     numProblems        problems
Class:      data.frame         numeric            list
```

The information of primary interest is in `problems`. This list has one element corresponding to each `$PROB` statement in the control file, although at the moment, ONLY ONE problem statement can be handled. An individual problem can be extracted with the `getProblem` function.

```
> prob <- getProblem(run)
> print(prob)
```

```
Standard NONMEM problem:
###############
Problem statement:  Analysis of one compartment model for theophyline data
Data file: ./data.csv
Input table dimensions:
144 7
Input table columns:
ID DOSE WT TIME DV MDV SMOK
PRED:
[1] "KA = THETA(1) + ETA(1) "
[2] "KE = THETA(2) + ETA(2) "
[3] "CL = THETA(3) + ETA(3) "
[4] "F = (DOSE*KE*KA*WT) /(CL * (KA-KE) ) * (EXP(-KE*TIME) - EXP(-KA*TIME) ) "
[5] "Y = F + EPS(1) "
[6] "IPRED = F"
[7] "IRES = (F - DV) "

Parameter estimates:
###############
THETAs:
THETA1 THETA2 THETA3
2.5500 0.0758 2.5600
OMEGAs:
       OMEGA1 OMEGA2 OMEGA3
OMEGA1   4.77  0e+00  0.000
OMEGA2   0.00  4e-05  0.000
OMEGA3   0.00  0e+00  0.203
SIGMAs:
```

```
[1] 0.456
Output table files: theoph.out
Output table dimensions:
144 9
Output table columns:
ID TIME DV IPRED DOSE WT IRES ETA1 ETA2
```

Individual problems can be of class `NMBasicModel`, `NMSimDataGen` or `NMSimModel`.

- `NMBasicModel` Is a standard NONMEM model fit, without simulation step
- `NMSimDataGen` Is a NONMEM problem with simulation step, but no model fitting
- `NMSimModel` Is a NONMEM propblem with simulation and model fitting on each simulation

3 Extracting data from a problem

3.1 Parameter estimates

For retrieving parameter estimates, one uses the functions `getThetas`, `getOmegas` and `getSigmas`. These take an additional parameter `stdError`, which controls whether or not standard errors should be returned if they are available.

```
> print(getThetas(prob))
```

```
THETA1 THETA2 THETA3
2.5500 0.0758 2.5600
```

```
> print(getOmegas(prob))
```

```
       OMEGA1 OMEGA2 OMEGA3
OMEGA1   4.77  0e+00  0.000
OMEGA2   0.00  4e-05  0.000
OMEGA3   0.00  0e+00  0.203
```

Additional extraction functions include `getObjective`, `getEstimateCov` (extract estimator covariance and correlation matrices), `getControlStatements` (extract an object holding the parsed statements of an object's control file), and others. See the online help for full details. Note that these can be used with `NMRun` objects as long as the problem numnber is specified (it is 1 by default), for instance:

```
> print(getThetas(run, problemNum = 1))
```

```
THETA1 THETA2 THETA3
2.5500 0.0758 2.5600
```

```
> print(getOmegas(prob, problemNum = 1))
```

```
       OMEGA1 OMEGA2 OMEGA3
OMEGA1   4.77  0e+00  0.000
OMEGA2   0.00  4e-05  0.000
OMEGA3   0.00  0e+00  0.203
```

```
>
```

3.2 Input and output data

nmData is a generic function for extracting a NONMEM run's input and output data tables, as described by the control file $DATA and $TABLE statements. The data is allowed to be missing when a run is loaded, in which case obviously it will not be retrievable. For a basic model, nmData has the following arguments:

- obj - NMBasicProblem class object
- dataTypes - character vector with strings input and/or output, determines which type of data is to be retrieved.
- returnMode - Whether to return data as a list of input and outputs, or a single data frame

```
> probOutData <- nmData(prob, dataTypes = "output")
> print(head(probOutData))
```

```
  ID TIME    DV   IPRED DOSE   WT     IRES     ETA1       ETA2
1  1 0.00  0.74  0.0000 4.02 79.6 -0.74000 -0.98198 -0.0049971
2  1 0.00  0.74  0.0000 4.02 79.6 -0.74000 -0.98198 -0.0049971
3  1 0.25  2.84  3.7650 4.02 79.6  0.92501 -0.98198 -0.0049971
4  1 0.57  6.57  6.7669 4.02 79.6  0.19690 -0.98198 -0.0049971
5  1 1.12 10.50  9.2182 4.02 79.6 -1.28180 -0.98198 -0.0049971
6  1 2.02  9.66 10.1210 4.02 79.6  0.46098 -0.98198 -0.0049971
```

```
> probData <- nmData(prob)
> print(head(probData))
```

```
  ID TIME    DV   IPRED DOSE   WT     IRES     ETA1       ETA2 MDV
1  1 0.00  0.74  0.0000 4.02 79.6 -0.74000 -0.98198 -0.0049971   1
2  1 0.00  0.74  0.0000 4.02 79.6 -0.74000 -0.98198 -0.0049971   0
3  1 0.25  2.84  3.7650 4.02 79.6  0.92501 -0.98198 -0.0049971   0
4  1 0.57  6.57  6.7669 4.02 79.6  0.19690 -0.98198 -0.0049971   0
5  1 1.12 10.50  9.2182 4.02 79.6 -1.28180 -0.98198 -0.0049971   0
6  1 2.02  9.66 10.1210 4.02 79.6  0.46098 -0.98198 -0.0049971   0
  SMOK ID.INPUT DOSE.INPUT WT.INPUT TIME.INPUT DV.INPUT
1    1        1       4.02     79.6       0.00     0.74
2    1        1       4.02     79.6       0.00     0.74
3    1        1       4.02     79.6       0.25     2.84
4    1        1       4.02     79.6       0.57     6.57
5    1        1       4.02     79.6       1.12    10.50
6    1        1       4.02     79.6       2.02     9.66
```

Note that the `.INPUT` postfix is used to handle data that is repeated in the output and input tables. Precedence is given to output data, which has no postfix. For simulation problems, one can select a vector of subproblems from which to extract the data.

Data may also be extracted by type via the `nmDatabyType` function. This extracts columns according to the type of data they hold, and type mappings are defined in the metadata. See the next section for details.

```
> x <- nmDatabyVarType(run, varTypes = "Parameter,Covariate", problemNum = 1 )
> print(head(x))
```

```
  DOSE SMOK Eta.ETA1    Eta.ETA2
1 4.02    1 -0.98198 -0.0049971
2 4.02    1 -0.98198 -0.0049971
3 4.02    1 -0.98198 -0.0049971
4 4.02    1 -0.98198 -0.0049971
5 4.02    1 -0.98198 -0.0049971
6 4.02    1 -0.98198 -0.0049971
```

Additional variables may be created by certain functions, including `addDerivedCategorical`, which derives a categorical variable from an existing data column. These added columns may then be extracted with `addedData`.

```
> prob <- getProblem(run)
> prob <- addDerivedCategorical(prob, "IRES", "IRES.CUT",
+     breaks = 3, labels = c("low", "medium", "high"))
> print(head(addedData(prob)))
```

```
  IRES.CUT
1   medium
2   medium
```

```
3     high
4   medium
5      low
6   medium
```

4 Configuration / metadata

RNMImport has tools for modifying the package configuration. For instance, paths can be stored under “names”. These names can be referenced by using round brackets in numerous functions.

```
> print(runPath)
```

```
[1] "C:/Users/jli.MANGO/AppData/Local/Temp/RtmpIN04KF/Rinst1d043fd8510a/RNMImport/example
```

```
> setNmPath("runPath", runPath)
> # note the use of round brackets
> controlContents <- importNmMod("theoph.con", path = "(runPath)" )
> print(head(controlContents))
```

```
$Raw
 [1] "$PROB Analysis of one compartment model for theophyline data"
 [2] "$INPUT ID DOSE WT TIME DV MDV SMOK"
 [3] "$DATA ./data.csv IGNORE=@"
 [4] "$PRED"
 [5] "KA = THETA(1) + ETA(1)"
 [6] "KE = THETA(2) + ETA(2)"
 [7] "CL = THETA(3) + ETA(3)"
 [8] "F = (DOSE*KE*KA*WT) /(CL * (KA-KE)) * (EXP(-KE*TIME) - EXP(-KA*TIME))"
 [9] "Y = F + EPS(1)"
[10] "IPRED = F"
[11] "IRES = (F - DV)"
[12] "$THETA"
[13] "(0.0,1.491825,50.0)"
[14] "(0.0,1.0,50.0)"
[15] "(0.0,2.773195,50.0)"
[16] "$OMEGA 0.4 0.4 0.4"
[17] "$SIGMA 0.4"
[18] "$EST MET = 0 POSTHOC MAXEVAL=6000 PRINT=5"
[19] "$TABLE"
[20] "ID TIME DV IPRED DOSE WT IRES ETA1 ETA2"
[21] "NOPRINT NOAPPEND ONEHEADER FILE=theoph.out"

$Comments
```

```
NULL

$controlFile
[1] "C:/Users/jli.MANGO/AppData/Local/Temp/RtmpIN04KF/Rinst1d043fd8510a/RNMImport/example

$problemContents
$problemContents[[1]]
$problemContents[[1]]$Theta
       Lower      Est Upper
THETA1     0 1.491825    50
THETA2     0 1.000000    50
THETA3     0 2.773195    50

$problemContents[[1]]$Omega
       OMEGA1 OMEGA2 OMEGA3
OMEGA1    0.4    0.0    0.0
OMEGA2    0.0    0.4    0.0
OMEGA3    0.0    0.0    0.4

$problemContents[[1]]$Sigma
       SIGMA1
SIGMA1    0.4

$problemContents[[1]]$Problem
[1] "Analysis of one compartment model for theophyline data"

$problemContents[[1]]$Tables
        File                                         Columns NoHeader
1 theoph.out ID, TIME, DV, IPRED, DOSE, WT, IRES, ETA1, ETA2    FALSE
  firstOnly append
1     FALSE  FALSE

$problemContents[[1]]$Input
  nmName Label
1 "ID"   "ID"
2 "DOSE" "DOSE"
3 "WT"   "WT"
4 "TIME" "TIME"
5 "DV"   "DV"
6 "MDV"  "MDV"
7 "SMOK" "SMOK"

$problemContents[[1]]$Data
     File         IG  ACCEPT REWIND  RECORDS TRANSLATE NULL
[1,] "./data.csv" "@" ""     "FALSE" ""      ""        ""

$problemContents[[1]]$PRED
[1] "KA = THETA(1) + ETA(1) "
[2] "KE = THETA(2) + ETA(2) "
[3] "CL = THETA(3) + ETA(3) "
```

```
[4] "F = (DOSE*KE*KA*WT) /(CL * (KA-KE) ) * (EXP(-KE*TIME) - EXP(-KA*TIME) ) "
[5] "Y = F + EPS(1) "
[6] "IPRED = F"
[7] "IRES = (F - DV) "

$problemContents[[1]]$Estimates
[1] "MET=0 POSTHOC MAXEVAL=6000 PRINT=5"
```

```
> removeNmPath("runPath")
```

One can also configure categorical variable “formats”, which define how levels of the category should be interpreted, as well as what the variables mean. These format descriptions are comma seperated lists. Below we show the existing formats (defaults are defined in a file included with the package) for SEX and SMOK, and then change SMOK. The function imposeCategoryFormat then forces variables to take a particular format.

```
> print(getVarDescription(c("SEX", "SMOK")))
```

```
   Variable   Label           Format   VarType
63      SEX  Gender 0=male, 1=female Covariate
64     SMOK Smoking      0=no, 1=yes Covariate
```

```
> setVarDescription("SMOK", "Smokes", varFormat = "0=NO,
+      1 = YES", varType = "Covariate")
> dat <- nmData(prob)
> dat <- imposeCategoryFormat(dat, varSubset = "SMOK")
> print(head(dat))
```

```
  ID TIME    DV   IPRED DOSE   WT     IRES     ETA1       ETA2 MDV
1  1 0.00  0.74  0.0000 4.02 79.6 -0.74000 -0.98198 -0.0049971   1
2  1 0.00  0.74  0.0000 4.02 79.6 -0.74000 -0.98198 -0.0049971   0
3  1 0.25  2.84  3.7650 4.02 79.6  0.92501 -0.98198 -0.0049971   0
4  1 0.57  6.57  6.7669 4.02 79.6  0.19690 -0.98198 -0.0049971   0
5  1 1.12 10.50  9.2182 4.02 79.6 -1.28180 -0.98198 -0.0049971   0
6  1 2.02  9.66 10.1210 4.02 79.6  0.46098 -0.98198 -0.0049971   0
  SMOK ID.INPUT DOSE.INPUT WT.INPUT TIME.INPUT DV.INPUT
1  YES        1       4.02     79.6       0.00     0.74
2  YES        1       4.02     79.6       0.00     0.74
3  YES        1       4.02     79.6       0.25     2.84
4  YES        1       4.02     79.6       0.57     6.57
5  YES        1       4.02     79.6       1.12    10.50
6  YES        1       4.02     79.6       2.02     9.66
```