PharmaSUG 2017 - Paper SP04

Population PK/PD Analysis – SAS® with R and NONMEM® Make Customization Easy

Sharmeen Reza, Cytel Inc., Cambridge, MA

ABSTRACT

Population pharmacokinetics/pharmacodynamics (pop-PK/PD) modeling and analysis are typically exploratory in nature. Different levels of customization are needed for modeling at various steps, where the core is driven by NONMEM software requiring that a structured file be passed onto it. Development and validation methods of such a NONMEM-ready data file rely heavily on the firmness of PK specs. Multiple tools, including SAS and R, are described in this paper for creating a data set. An interface-based solution reduces programming dependencies but has limitations as its backend logic is constructed on a single version of PK specs. Other parts of the analysis, pre-processing of data file and post-processing of NONMEM software output, are performed in SAS and/or R; the choice of tools is determined by availability and user competency. Since SAS is well-established in the pharmaceutical industry for regulatory submissions (Rickert, 2013), it is prudent to utilize SAS for executing all the pieces – making data investigation easy especially for large data, allowing flexibility with programming, connecting with R and NONMEM software and delivering reports after fine-tuning. SAS can be utilized for a fully customizable solution or to take advantage of a compatible interface that is highly adaptable to the frequently changing requirements of the pop-PK/PD world.

INTRODUCTION

NONMEM software is widely used for population pharmacokinetics/pharmacodynamics (pop-PK/PD) modeling in the pharmaceutical industry. In order to use the software, data need to be in a certain format complying with specifications provided by PK scientists. Typically a NONMEM-ready data set is processed in SAS and converted to a CSV file in preparation for the analysis run.

There are various other modeling software available and a CSV file is typically accepted as input by all. Similarly instead of SAS, other options exist for creating a CSV. In this paper NONMEM is picked due to familiarity and SAS is emphasized because it makes complex data manipulation fairly straightforward.

For pre- and post-processing, the R programming language is a viable option given its flexibility, and statistical and graphical capabilities. For data set formation, the argument goes that SAS stands out especially when handling large data (Fraeman, 2008; Crevar, 2007). The question is whether SAS can be used for all steps: pre-processing, NONMEM-ready data set creation, NONMEM-run, and post-processing. An evaluation of SAS in connecting with other tools to have a single application perform all steps in Pop-PK/PD analyses is explored here, which should help streamline modeling efforts in drug development.

ANALYSIS PIECES

A NONMEM analysis diagram presented in an earlier paper (Reza, 2015) is presented below with minor modifications. It now includes, with dashed arrows, pre-processing and model adjustments by scientists. Pre-processing deals with checking data for consistency and identifying errors. The best way to catch issues is through visual representations such as using a patient concentration-time profile. For model preparation scientists also commonly use graphics, optimizing designs by exploring variables of interest. Finalized models are passed as control streams – shown using connector A in Figure 1 below. Post-processing involves interpreting results with diagnostic plots followed by generating reports in table and graph format which are included in the submission package along with the NONMEM data set.

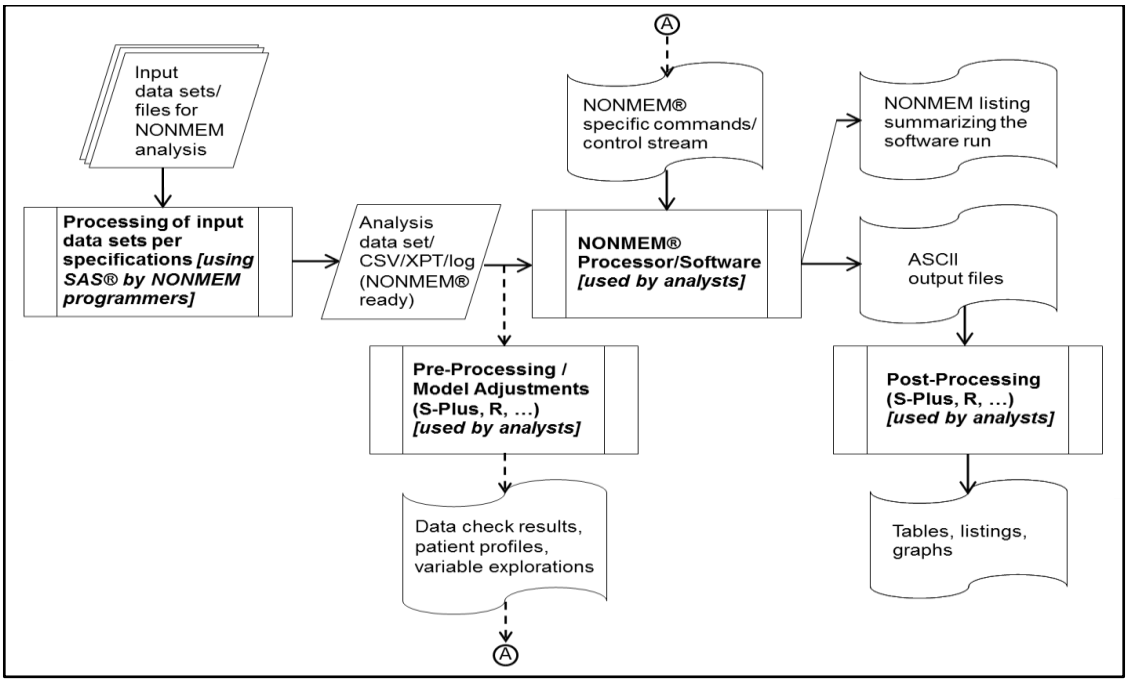

Figure 1. Pop-PK/PD Analysis with Pre-processing and Model Adjustments

Programmers generally use SAS for creating a NONMEM-ready data set and an associated CSV file that is fed into the NONMEM software – illustrated in Figure 1. Often a non-SAS option (examples: S-Plus, R) is considered for performing pre- and post-processing, where NONMEM software handles the analyses portion. Choice of environment, language, or package is generally dependent on tools availability, cost, ease of use, and expertise.

NONMEM DATA REQUIREMENTS, STRUCTURE, AND FORMAT

Programmers develop NONMEM-ready data sets using PK specs that come from scientists. This document lists variables, their definitions, and instructions for stacking observations from various input SDTM/ADaM domains. The primary ones are dosing (EX) and PK concentration samples (PC), whereas, other domain datasets are added given analysis and/or study-specific needs.

PK specs include missing value interpretation and data imputation guidelines, sorting order, subject or record exclusion criteria, etc. With all these details they can become complex, and get more convoluted when several studies are included for a meta-analysis, for instance. NONMEM software expects the input CSV file in a specific structure such as one row per POP, ID, DATE, TIME, EVID, and CMT (Reza, 2015). Variable order and format (examples: text, numeric) are also critical and follow from the PK specs, designed by scientists according to NONMEM software directives.

DEVELOPMENT AND VALIDATION OF A NONMEM-READY DATA SET

To streamline data set creation, programmers first generate a NONMEM DDT (Data Definition Table) using definitions in PK specs. Two main options of data build are:

PROGRAMMATICALLY

It is important to note that PK specs go through numerous changes during a clinical study life cycle and so does the DDT, requiring code changes in both development and validation. Standardization of specs and code specific to Therapeutic Areas (TAs) has been suggested (Reza, 2016) in an earlier paper for making the repetitive process less time consuming. Nonetheless, coding a NONMEM-ready data set is a

major effort, especially in taking care of all data scenarios; also the length of a program can be large by itself.

Since data sets are the groundwork for all submission reports, validation is essential for NONMEM-ready data set. Independent programming is performed in parallel, by Source Programmer and QCer.

INTERFACE-BASED

Automation is being investigated for NONMEM-ready data set creation. There is interface-based software available in the market which utilizes drag-and-drop technology through a Graphical User Interface (GUI). It allows field selection from input sources and generates output data set that follows the DDT specification.

If the same automated system is to be used for validation though, it can lead to similar results using the same derivations, violating Good Clinical Practices (GCP) and regulatory guidelines. Thus, programming is still likely the answer ensuring independent settings for QC.

COMPARISON OF SAS, R, AND OTHER OPTIONS IN NONMEM ANALYSIS

In modeling, scientists usually come with a background in R (especially academics as R is freely available as open source software), and tend to continue with R due to its strengths in statistics and graphics, which helps in modeling work. On the other hand, programmers stick to SAS because of its fast learning curve while focusing on submission requirements and the intricacies of NONMEM data set creation. The section below describes the pros and cons of using various options for processing different pieces in NONMEM analysis.

SAS

SAS syntax is easy to learn mainly because it is not a full-fledged computing language, rather an analytical system for data processing, analyses, and visualization.

For making a NONMEM-ready data set, sophisticated programming skills are not really required, but mastering data structures is critical (Reza, 2015). SAS is an attractive choice especially when dealing with large data, numerous domains, and multiple studies in a single data set.

SAS is equally effective for other parts in an analysis; its statistical techniques are useful for preparing and testing models, while procedures like REPORT, TEMPLATE, and FORMAT are available particularly for rendering outputs.

For greater flexibility, SAS 9.22 and higher allow embedding R code and running it from its environment (SAS Blog, 2013).

In SAS version 9.3 and beyond, Base SAS includes SG (Statistical Graphics) procedures and ODS Graphics by default, giving different ways of producing graphs; it allows data display quick and adjustable through GTL (Graph Template Language) which runs in the background. Choices of graph production method can be: SG procedures, ODS Graphics Designer, and ODS output from any SAS analytical procedure where output is editable in the ODS Graphics Editor (Zong, 2013).

Debugging is comfortably done in different SAS platforms: PC/Windows SAS or SAS Enterprise Guide. One can effortlessly see inputs and outputs at each step checking their validity.

Big corporations and sponsor companies usually have SAS installed on their system; it is an established tool used for drug submission (Rickert, 2013). SAS readily meets validation requirements providing built-in documentation (log, lst files) – with traceability and reproducibility capabilities meeting FDA 21 CFR part 11 (Sporon-Fiedler et al., 2002).

DDT and code generation may take many iterations because of frequent changes in PK specs and data incorporation at various snapshots throughout a trial. Yet fast turnaround time is required for data set delivery to scientists. With SAS, customization is conceivable providing the flexibility needed for PK analyses.

Also SAS has a backward compatibility feature, which is suitable in integrating updates in programs and useful in many legacy cases; an example is when a regulatory agency comes back with queries on a data set submitted years ago and written in an older version.

For audit trails tracking changes is critical. Versioning a SAS project coupled with any third-party version control software is simple in computing platform like Windows and UNIX.

The main disadvantage of SAS is the hefty license cost and for a smaller organization can be prohibitive. Also installing SAS is generally time consuming, though this depends on the number of components in the installation process.

R

R has gained popularity in recent decades because of its statistical strength and open source availability. Thus, no license cost is associated with R or setting up its environment such as RStudio®.

Powerful graphics in R go hand in hand with PK analyses where visualization of data is important.

For informal peer review R can be handy, however, code in R becomes tricky when handling missing data imputation for example. Additionally the amount of code required for for simple data manipulation like sorting becomes lengthy. For full NONMEM-ready data set programming, the length of a program written in R can be very long and difficult to maintain.

R is typically not suited for rendering tables, though; it is possible to load additional packages requiring more programming knowledge and training.

Debugging in R is difficult as its error messages are quite ambiguous to a novice users; it takes years of experience having a good grasp of online help.

Given that it is open source, any user can add R packages which could introduce inconsistencies in syntax. Also R documentation is diverse and not integrated in any way, making it difficult to pick up the language quickly. Furthermore, it is a true programming language with a steep learning curve, requiring a long training phase (Analytics Training, 2011).

Validation in R is tedious and not suitable for frequent iterations, such as in the PK/PD scenario and beyond, and R has not historically been used in submissions.

For R projects, one favorable aspect of R is that version control is integrated in RStudio and comes as open source.

GUI

It is desirable to have a system automated to improve efficiency in NONMEM-ready data set production. The main attraction in interface-based approach is that users do not need to be involved in programming. The caveat is, in the background, standard prototypes must be in place. Basically code behind the GUI would need to be based on a single version of PK specs.

Additionally, there are always possibilities of study-specific nuances that one encounters leading to spec changes. Thus, the automation interface must be flexible enough to accommodate such adjustments. Customizing a data set might not be possible right away until a future version of the GUI software provides additional features.

Some software do allow customization but at the cost of programming, since backend logic is not able to handle study-specific needs, say for including more derived variables or domains.

Also, for pre- and post-processing, the GUI system design could be enhanced with supplemental components (Jayaraman et al., 2010).

Of course developing a GUI, user-friendly software comes with a price. In case of making an interface for NONMEM analysis, multiple functions and resources (software developers, clinical programmers, scientists, data management, CDISC standards group, etc.) are involved boosting development and license cost.

As always there is a learning curve associated with any software.

COMPARISON TABLE

For the purpose of pop-PK/PD analyses, exclusively in drug industry, and from the above discussions the following table recaps the comparison of three options:

Capabilities \ Options	SAS	R	GUI
Backward Compatibility	Yes	Mostly	Design-dependent
Code Complexity	Simple	Complex	Design-dependent
Customizable	Yes	Yes	No, version-specific
Data Manipulation	Easy	Difficult	Design-dependent
Data Visualization & Graphics	Advanced	Advanced	Design-dependent
Debugging	Simple	Complex	Design-dependent
Established Tool in Submission	Yes	No	No
Flexibility	Yes	Yes	Design-dependent
Installation Process	Moderate	Easy	Design-dependent
Learning Curve	Short	Steep	Design-dependent
License Cost	Applicable	None	Applicable
Object-oriented Programming	No	Yes	Design-dependent
Programming Skills	Basic to medium	Advanced	None
Rendering Table Outputs	Easy	Takes extra steps	Design-dependent
Software Documentation	Targeted	Diverse	Company based
Software Support	Strong/On-demand	Open-ended/Online	Company based
Statistical Features, if needed	Advanced	Advanced	Design-dependent
Syntax	Simple	Inconsistent	Design-dependent
Traceable & Reproducible	Yes	Yes	Design-dependent
Training Phase	Short	Long	Design-dependent
User Intervention	Yes	Yes	No
Validation for Regulatory	Somewhat built-in	Takes extra steps	Design-dependent
Version Control Compatibility	Yes	Yes, with Rstudio	Design-dependent

Table 1. Comparison Table

A CUSTOMIZABLE SOLUTION

SAS certainly is advantageous in NONMEM-ready data set creation. If SAS is already deployed at an organization, it is worthwhile to make an effort to have a dedicated solution for pop-PK/PD analyses.

When an interface for NONMEM-ready data set production is built in SAS, integrating with other parts which are also executable in SAS environment would be seamless. Updating data set logic for study-specific needs is feasible given a GUI-based code-generator; it might spit out code which can be tweaked when prototypes are modified. Figure 2 shows the addition of GUI on the previous diagram by a shadowed processor.

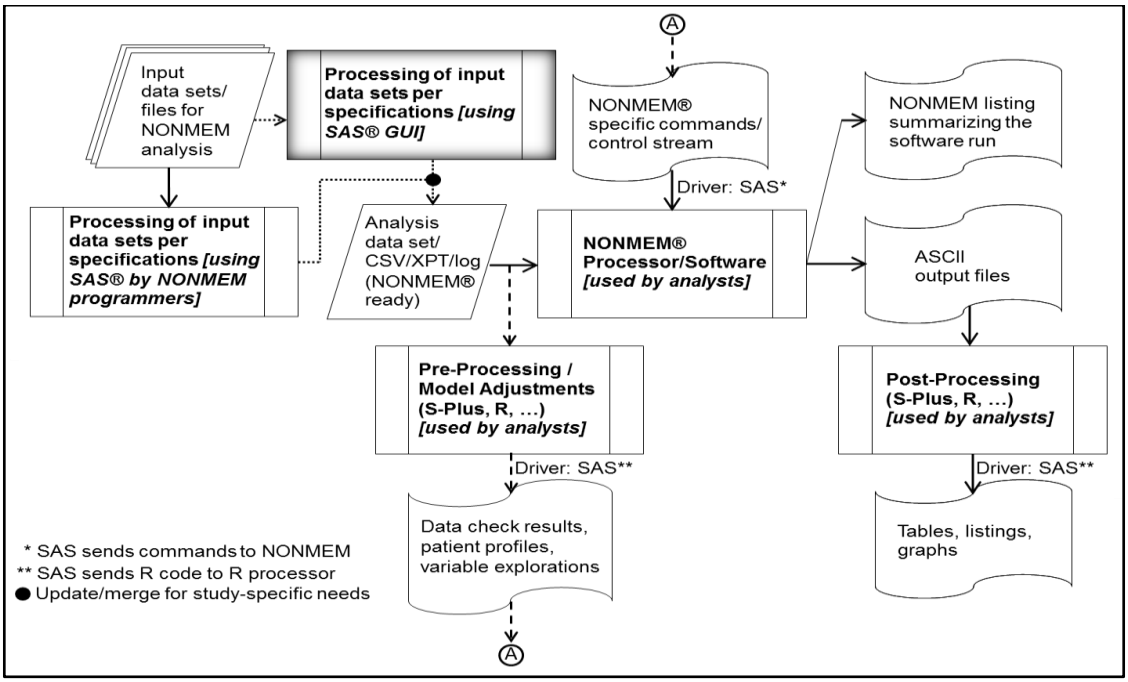

Figure 2. Pop-PK/PD Analysis with Pre-processing and Model Adjustment, and GUI

Also, an interface-based solution should have the ability to add a new prototype promptly given business requests. One example would be an addition of PD domain which is not uncommon in exploratory pop-PK/PD. If this requires a total software upgrade and becomes resource intensive, SAS can still be used for writing the PD code from scratch and merging it with the output data set produced by the GUI. Likewise for a whole new TA, on top of primary EX and PC, more SDTM/ADaM domains can be included by coding manually until any interface upgrade happens. Any update or merge on GUI output is shown in Figure 2 by a solid dot.

Pre-processing, investigating data, displaying patient profiles, checking errors, preparing models can be easily done using code snippets in SAS. If scientists prefer to use existing R programs, SAS has wrapper procedure IML that encapsulates R code and runs it from within its environment (SAS Communities, 2015) – see double asterisks in Figure 2. SAS works here as a 'driver' sending necessary instructions to the R processor and retrieving results.

Similarly for post-processing if scientists have table and graph layouts established, populating them in SAS is very much manageable, as in regulated Clinical Study Reporting (CSR). The same is true for creating evaluation/diagnostics plots.

Since the IML component of SAS poses an additional licensing cost, instead of PROC IML this paper gives a few examples of R-3.3.2 and SAS 9.4 code for commonly used plots in pop-PK/PD analysis, achieving equivalent outcomes: 1) Concentration vs Time, 2) Covariate Matrix, 3) Goodness of Fit – displayed in Figure 3 below.

R-3.3.2 Code

```
library(lattice)
# 1
pdf(file="C:\\test1_R.pdf")
xyplot(conc ~ time |
         paste("ID =", id),
  data=mydata,
  main="R, 1. Conc vs Time",
  xlab="Time", ylab="Conc (ng/mL)",
  type='b', groups=trt,
  par.settings=list(par.xlab.text=
                     list(cex=.9),
                    par.ylab.text=
                     list(cex=.9)),
  auto.key=list(title='Treatment',
                space='top',
                columns=2,
                border=TRUE,
                cex=.8),
  index.cond=list(c(2,3,6,1,4,5))
); dev.off()

# 2
pdf(file="C:\\test2_R.pdf")
panel.hist <- function(x, ...) {
  usr <- par("usr")
  on.exit(par(usr))
  par(usr=c(usr[1:2], 0, 1.5))
  h <- hist(x, plot=FALSE)
  breaks <- h$breaks
  nB <- length(breaks)
  y <- h$counts; y <- y/max(y)
  rect(breaks[-nB], 0,
       breaks[-1], y,
       col="cyan", ...)
}
pairs(mydata[c(8,5,4)], pch=21,
      diag.panel=panel.hist,
      main="R, 2. Covariate Matrix"
); dev.off()

# 3
pdf(file="C:\\test3_R.pdf")
lm.r <- lm(mydata$conc ~
           mydata$time)
plot(fitted(lm.r), mydata$conc,
  main="R, 3. Goodness of Fit",
  xlab="Predicted Conc",
  ylab="Observed Conc",
  xlim=c(0, 500), ylim=c(0, 500)
)
lm.r_n <- lm(mydata$conc ~
             fitted(lm.r))
abline(lm.r_n); dev.off()
```

SAS 9.4 Code

```
* 1;
ods listing close;
ods pdf style=word;
ods pdf file="C:\test1_SAS.pdf";
title "SAS, 1. Conc vs Time";
proc sgpanel data=mydata;
  panelby trt id/columns=3;
  scatter x=time y=conc/group=trt;
  series x=time y=conc/group=trt
     markers lineattrs=(pattern=1);
run;
ods pdf close;
ods listing;

* 2, note: Can create same plot;
*      in ODS Graphics Designer;
ods listing close;
ods pdf style=word;
ods pdf file="C:\test2_SAS.pdf";
title "SAS, 2. Covariate Matrix";
proc sgscatter data=mydata;
  matrix age wt sex/group=trt
  diagonal=(histogram);
run;
ods pdf close;
ods listing;

* 3, note: Can update sge file;
*     (ex: title/labels);
*     in ODS Graphics Editor;
ods listing close;
ods pdf style=journal;
ods pdf file="C:\test3_SAS.pdf";
ods graphics on;
ods listing sge=on;
title "SAS, 3. Reg - Obs vs Pred";
proc reg data=mydata
  plots(only)=ObservedByPredicted;
    model conc=time;
run;
quit;
ods listing sge=off;
ods graphics off;
ods pdf close;
ods listing;
```

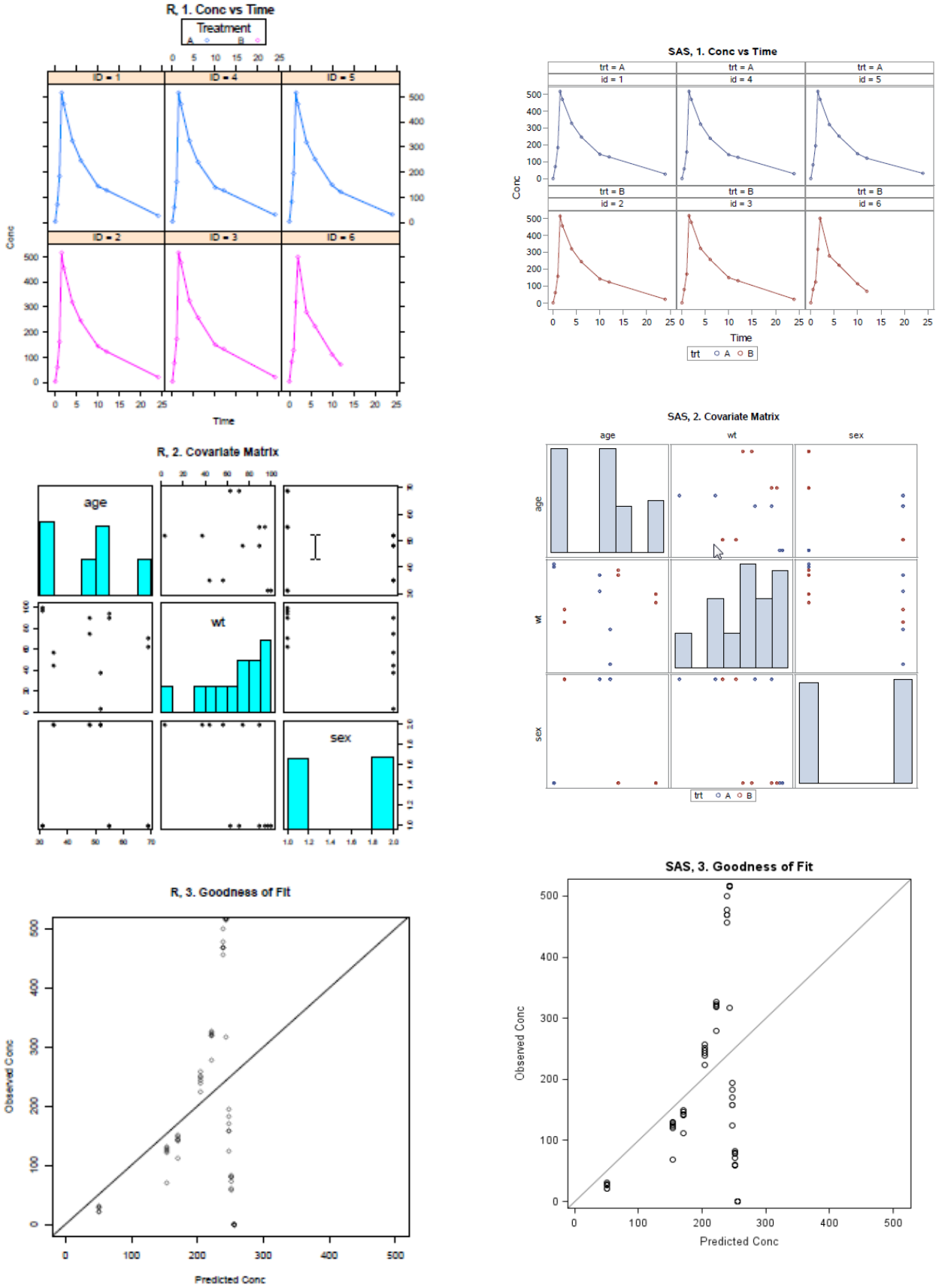

Figure 3. Graphs in R-3.3.2 and SAS 9.4, Bottom-Right: Updated Title/Labels in ODS Graphics

The core part of analyses, running the NONMEM software, might be driven by scripts written in R or another 4th generation language (ex: Perl) where the script is embedded and executed in SAS shown with single asterisk in Figure 2. Since the focus of this paper is to utilize SAS for efficiency, the details of such scripts are beyond the scope here; the relevant R package, the Perl command and steps for NONMEM run from SAS are described in the literatures (Knebel et al., 2013; Jayaraman et al., 2010).

By default all footprints of SAS jobs go into its log. Redirecting – R log and command-prompt-output (from running NONMEM software) with the PIPE command into the SAS log – make the whole analysis easier to track, maintain, and debug.

Furthermore, tables and graphs produced in SAS (directly or via R code) can be enhanced and customized by tweaking different parameters of the outputs. Also a final model might be selected on the fly, from SAS, by evaluating a set of test models (Ford, 2015; Bonate et al., 2012).

CONCLUSION

Irrespective of roles, there should be definitive lines separating the pieces involved in pop-PK/PD analysis and doing them with the appropriate analytics tools. It is probably compelling to think of R as one. For a comprehensive resolution though R does not quite fit the bill. R could be convenient for partial coding or peer review, but not for bulk data assembly and rendering outputs.

The analysis work calls for an environment that lets data processing be efficient, data visualization easy, memory management smooth, debugging straightforward, and validation formal. SAS comes with these features and is well-recognized in drug industry where inquiries and audit trails are common. Its ability to couple with other software provides flexibility and quality in submission. SAS allows quick customization for exploratory work. It can be utilized for a complete pop-PK/PD solution, including as an interface builder, combining all the analysis pieces.

REFERENCES

- Analytics Training, 2011. R vs SAS (Comparison and Opinion). http://www.learnanalytics.in/blog/?p=9
- Bonate, P. L., Strougo, A., Desai, A., Roy, M., Yassen, A., van der Walt, J. S., Kaibara, A., Tannenbaum, S., 2012. Guidelines for the Quality Control of Population Pharmacokinetic–Pharmacodynamic Analyses: an Industry Perspective. AAPS Journal, 14(4): 749-758.
- Crevar, M., 2007. How to Maintain Happy SAS® Users.
- Ford, N., 2015. Principles of Covariate Modelling. http://holford.fmhs.auckland.ac.nz/docs/principles-of-covariate-modelling.pdf
- Fraeman, K.H., 2008. Common Sense Tips and Clever Tricks for Programming with Extremely Large SAS® Data Sets.
- Jayaraman, B., Dombrowsky, E., Marsteller, D., Winkler, J., Barrett, J.S., 2010. A SAS-based Solution for NONMEM run management and post-processing. PAGE 19 (2010) Abstr 1781. http://www.page-meeting.org/?abstract=1781
- Knebel, B., Gibianski, L., Bergsma, T., 2013. Package 'MIfuns'.
- Reza, S., 2015. Growing Needs in Drug Industry for NONMEM Programmers Using SAS®. PharmaSUG, Paper SP07.
- Reza, S., 2016. Scrambled Data – A Population PK/PD Programming Solution. PharmaSUG, Paper SP04.
- Rickert, J., 2013. R, drug development and the FDA. http://blog.revolutionanalytics.com/2013/08/r-drug-development-and-the-fda.html
- SAS Blog, 2013. What versions of R are supported by SAS? http://blogs.sas.com/content/iml/2013/09/16/what-versions-of-r-are-supported-by-sas.html
- SAS Communities, 2015. Run R code inside SAS easily. https://communities.sas.com/t5/General-SAS-Programming/Run-R-code-inside-SAS-easily/td-p/210116
- Sporon-Fiedler, G., Lassen, M., A/S, H.L., 2002. SAS® Coexistence with FDA 21 CFR Part 11, How Far Can We Get? PharmaSUG Proceeding.
- Zong, A., 2013. SAS® 9.3: Better graphs, Easier lives for SAS programmers, PK scientists and pharmacometricians. PharmaSUG, Paper SP09.

ACKNOWLEDGMENTS

The author would like to acknowledge Cytel Inc. for providing the opportunity to work on this paper. The author thanks the following managers and colleagues for thoughtful comments.

Baker, Jim (Senior Vice President of Clinical Research Services, Cytel)
Collins, Art (Associate Director, Statistical Programming at Biogen)
Eschenberg, Michael (Director Biostatistics, Amgen)
Jemiai, Yannis (Senior Vice President, Strategic Consulting, Software Solutions & Marketing at Cytel)
Vanderstay, Mark (Director, Technical Development at SoulPad)

CONTACT INFORMATION

Sharmeen Reza
Cytel Inc.
Work Phone: 269-743-7221
E-mail: Sharmeen.Reza@cytel.com