Creating mocks that can easily be used for SAS templates

Version 1.0

Date: 25/APR/2022

Author: RPA

rpaul@gcp-service.com

Introduction

The goal of this short introduction is to give you an idea on how to create mock tables and listings that can be read by an R script that creates SAS templates for easier programming. The proposed process is by no means perfect and as of the writing of this document still under development.

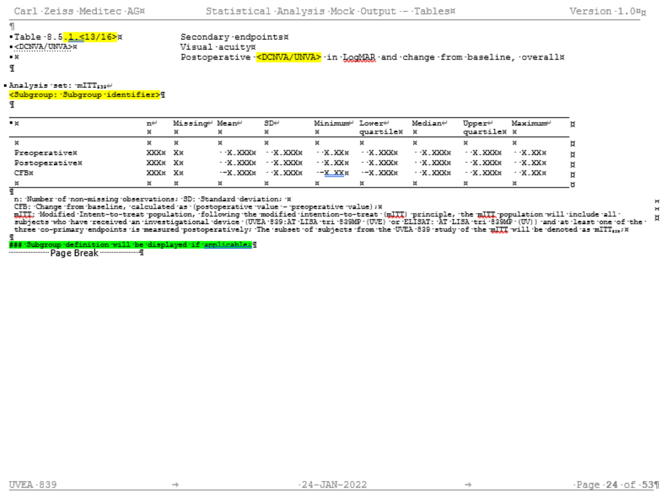
For now, the process encompasses

- Creating a Word file (*.docx) containing the mock output within the required format
- Editing and running the R script (*.R) to create the SAS dummy files
- Editing the SAS files (*.sas) to start working on your analyses

This process is meant to accelerate the start-up phase of the analysis by keeping the overhead of creating file headers and standard code chunks to a minimum. Thus, reducing the risk for copy-paste errors and typos in the documentation parts of the files. As a side effect it adds a standardized structure for mock files.

Creating your Mock File

For the process to work, the file needs to be in *.docx-format and each individual mock table/listing needs to follow a specific pattern.



Carl Zeiss Meditec AG   Statistical Analysis Mock Output – Tables   Version 1.0

Table 8.5.1.<13/16>   Secondary endpoints
<DCNVA/UNVA>   Visual acuity
    Postoperative <DCNVA/UNVA> in LogMAR and change from baseline, overall

Analysis set: mITT$_{839}$
<Subgroup: Subgroup identifier>

	n	Missing	Mean	SD	Minimum	Lower quartile	Median	Upper quartile	Maximum
Preoperative	XXX	X	X.XXX	X.XXX	X.XX	X.XXX	X.XXX	X.XXX	X.XX
Postoperative	XXX	X	X.XXX	X.XXX	X.XX	X.XXX	X.XXX	X.XXX	X.XX
CFB	XXX	X	-X.XXX	X.XXX	-X.XX	-X.XXX	X.XXX	X.XXX	X.XX

n: Number of non-missing observations; SD: Standard deviation;
CFB: Change from baseline, calculated as (postoperative value – preoperative value);
mITT: Modified Intent-to-treat population, following the modified intention-to-treat (mITT) principle, the mITT population will include all subjects who have received an investigational device (UVEA 839:AT LISA tri 839MP (UVE) or ELISAT: AT LISA tri 839MP (UV)) and at least one of the three co-primary endpoints is measured postoperatively; The subset of subjects from the UVEA 839 study of the mITT will be denoted as mITT$_{839}$;

Subgroup definition will be displayed if applicable.

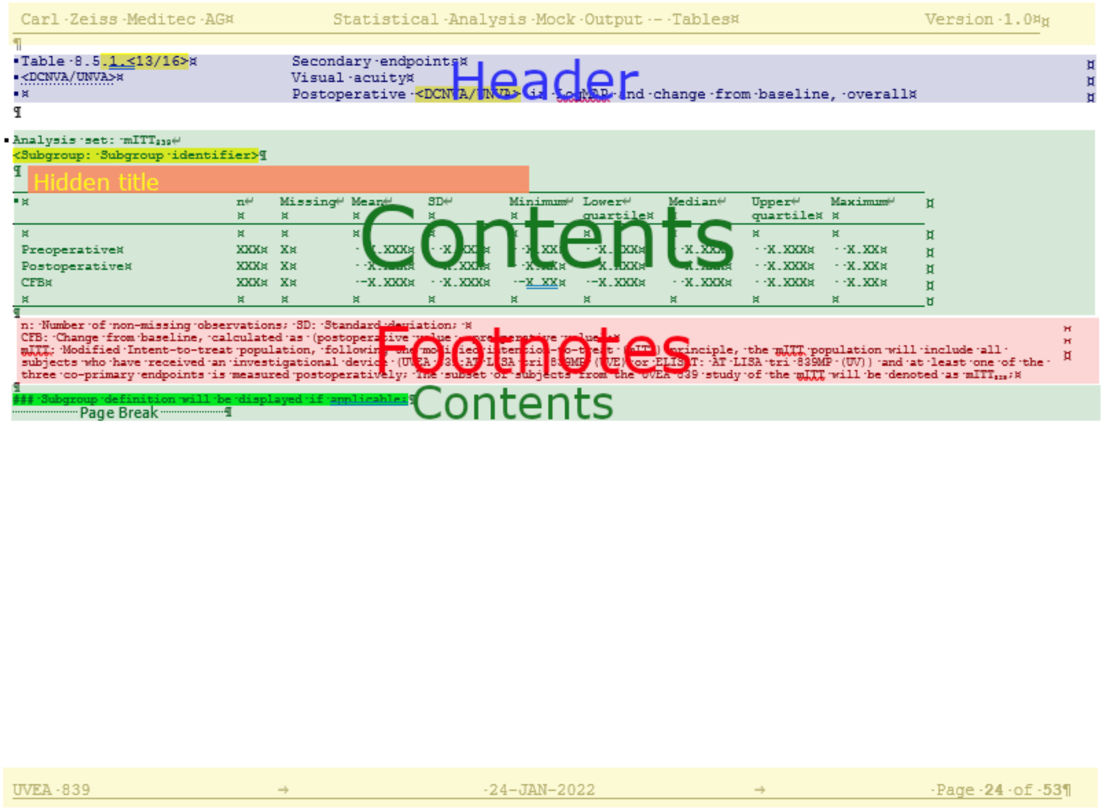
Page Break

UVEA 839   24-JAN-2022   Page 24 of 53

Figure 1 Example mock table taken from UVEA839.

Figure shows an example of a mock table taken from the UVEA839 study. The whole table can be divided into multiple sections.

An example template can be found under SAS Template Creator. It includes all designs and sections needed to start a new mock file directly or to copy them over to your project's mock file.



Carl Zeiss Meditec AG　Statistical Analysis Mock Output – Tables　Version 1.0

Table 8.5.1.<13/16>	Secondary endpoints
<DCNVA/UNVA>	Visual acuity
	Postoperative <DCNVA/UNVA> in logMAR and change from baseline, overall

Analysis set: mITT_{839}
<Subgroup: Subgroup identifier>

	n	Missing	Mean	SD	Minimum	Lower quartile	Median	Upper quartile	Maximum
Preoperative	XXX	X	X.XXX	X.XXX	X.XX	X.XXX	X.XXX	X.XXX	X.XX
Postoperative	XXX	X	X.XXX	X.XXX	X.XX	X.XXX	X.XXX	X.XXX	X.XX
CFB	XXX	X	-X.XXX	X.XXX	-X.XX	-X.XXX	X.XXX	X.XXX	X.XX

n: Number of non-missing observations; SD: Standard deviation;
CFB: Change from baseline, calculated as (postoperative value – preoperative value)
mITT: Modified Intent-to-treat population, following the modified intention-to-treat (mITT) principle, the mITT population will include all subjects who have received an investigational device (UVEA 839: AT LISA tri 839MP (UVE) or ELISAT: AT LISA tri 839MP (UV)) and at least one of the three co-primary endpoints is measured postoperatively; The subset of subjects from the UVEA 839 study of the mITT will be denoted as mITT_{839};

Subgroup definition will be displayed if applicable:

UVEA 839　24-JAN-2022　Page 24 of 53

Yellow: Document Header/Footer, to be done manually

Blue: Mock header; a 2x3 table containing visible content and hidden information for the R script

Orange: Optional hidden title to be referenced by the table of contents.

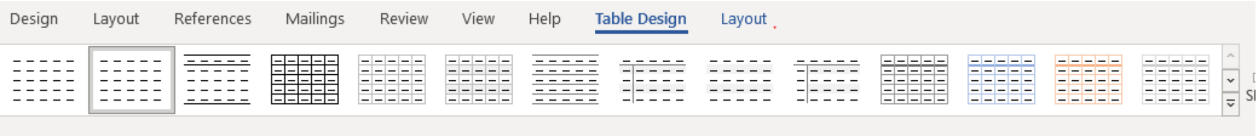
Green: Mock-specific contents. No pre-defined structure

Red: Mock footnotes; a 1xN table. N is the number of footnotes desired to be included. Up to 8 lines of footnote text are possible. The standard footnote "Output created by program …" will be added automatically by the script and doesn't need to be added in the mock file.

The parts that are scanned by the script are the mock header and the mock footnotes, so they will be addressed in detail

Mock header

The header is a mandatory 2x3 Table at the beginning of a mock table. For everything to work, the R script needs a certain table design to look out for. These can be added under "Table Design" in the menu ribbon of Word after clicking somewhere into a table cell.



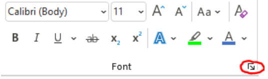
Currently, the script expects the table design to be called "**Mock Header Table**". If the table design has a different name, the script won't function properly!

The overall table cosmetics are up to you. The template under SAS Template Creator. uses a design that mimics the conventional GCPS standard mock TFLs with Courier New as font and white table borders to hide the outlines in the final document.

Currently, 5 of the resulting 6 table cells are used to transport information about the table or the target SAS-program. The contents need to be inserted in the following way:

Table/Listing <Number>	Title level 1
<SAS-Program name>	Title level 2
	Title level 3

The script uses "Table/Listing <Number>" and "Title level 3" to create a table of contents entry for the final TFL output. This is not the table of contents reference used in the Mock file, though!

"SAS-Program name" should be a recognizable short name for the SAS-program that will be created by the R-Script. To avoid confusion at the recipient's end, this information should be converted into hidden text. This way, it will not be displayed in the pdf-version of the file. For this, you need to select the text you want to hide, Home →click the arrow next to "Font"  →Check "Hidden".

The final name for the program will be

- [Study Acronym]_[T/L][Table/Listing <Number>]_[SAS-Program name]_D01_0_0.sas.

The example table in Figure 1 was created as "UVEA839_T8_5_1_13_DCNVA_D01_0_0.sas" (and "UVEA839_T8_5_1_16_UNVA_D01_0_0.sas", but more on how to create multiple programs from one mock later). Hidden text visibility can be toggled for editing by clicking ¶ under Home → Paragraph.

Mock Footnotes

Similar to the header, the footnotes require a specific Design name to be properly detected by the R-Script. Currently, the Design must be called **"Mock Footnote Table"**. Again, cosmetics are up to the user. The template provided uses a design with white borders to be invisible and the standard Courier New font.

Each footnote table row is used by the R-script to detect new footnote lines in the produces SAS-code. For example, the footnote table

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua;
At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet;

Produces the SAS code

```
FOOTNOTE1 J=L HEIGHT=8PT FONT=COURIER " Lorem ipsum dolor sit amet, consetetur sadipscing elitr, 
sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua;
FOOTNOTE2 J=L HEIGHT=8PT FONT=COURIER " At vero eos et accusam et justo duo dolores et ea rebum. 
Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet;";
FOOTNOTE4 J=L HEIGHT=8PT FONT=COURIER "Output generated by program '&this_prog.';";
```

Note, that “Output generated…” is automatically added as the second to next footnote after the last defined footnote in the table.

Hidden title (optional)

If you want to wrap up your mock file and create a table of contents, oftentimes it doesn’t turn out to be as desired. Especially, since the table/listing number and the title level that is needed for the table of contents reference are in different cells of the mock header table. A somewhat simple workaround is to include the desired title as white text between the mock header and the contents of the mock. For example, copy the mock numbers and level 3 titles somewhere below the mock header tables, like

- `Table 8.5.1.<1/4>: Postoperative <CDVA/UDVA> in LogMAR and change from baseline, overall`

Then create a style (Home→ Styles→ More (lower arrow down)→ create a style →modify →format →font) and give it white font colour by default. In the template, this style is called “White_Title”. Apply this to all your titles as displayed above to get

For the table of contents to display only the “White_Titles”, got to where you want to insert the table of contents, References→Table of Contents→Custom Table of Contents→Options→remove all numbers from the input fields except for “White_Title”, where it should say “1”. Confirm with “OK” twice, a table of contents should be displayed.

Advanced: How to create two programs from one mock

Sometimes, you want to provide several analyses which are basically the same with only the underlying variable/CRF/scope being different. Let’s take the example in Figure 1. The mock is applicable for distance-corrected near visual acuity (DCNVA) and uncorrected near visual acuity (UNVA). To safe time, it’s possible to combine similar analysis into one mock. The yellow highlighting indicates where the differences in detail are. Nevertheless, we now want to produce two (currently the script only supports up to two output programs) SAS-programs out of it.

As it is displayed in the example, the contents by which the two outputs shall be identified need to be put into a pattern following “<Mock 1/Mock 2>”. This means with the input “Table 8.5.1.<1/4>” the script produces the mock numbering “Table 8.5.1.1” for the first script and “Table 8.5.1.4” for the second.

This can be applied to any title level and footnotes. It must be applied to the hidden text containing the program name, which is necessary to create two separate programs.

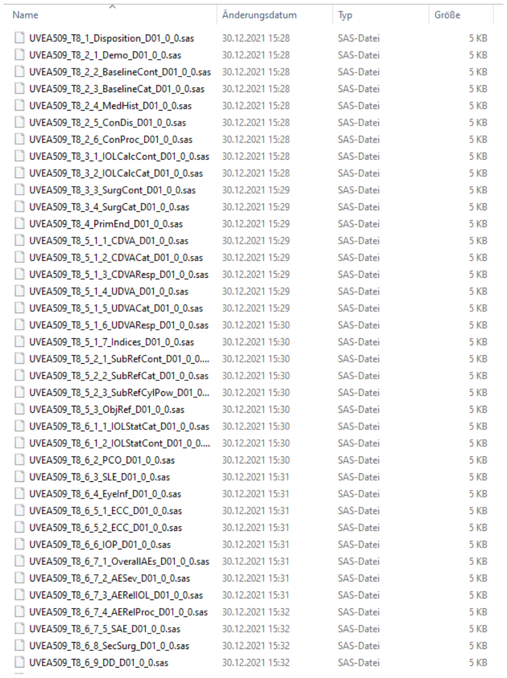
Running the R-script

Open “SAS_Template_Generator_D01_0_0.R” in RStudio. The required user input is marked by comments. The rest of the program doesn’t need any changes.

Make sure you have already installed all necessary packages (“officer”, “tidyverse”, “lubridate”, “stringr”).

The following input is expected:

Study_Acronym	The short name of the study
Sponsor	The full name of the Sponsor
Author	Your initials
project_path	Usually: "Z:/STUDIES/<studyfolder>/WORKSPACE/"
mock_file	The **name** (not the complete path!) of your *.docx-file containing the mocks with the defined format from the section before, located under “Z:/STUDIES/<studyfolder>/WORKSPACE/07_STATISTIC/08_Mock_TFL”
output_path	The path (relative to project path, i.e. starting in project_path), where you want the SAS-programs to be created



Name	Änderungsdatum	Typ	Größe
UVEA509_T8_1_Disposition_D01_0_0.sas	30.12.2021 15:28	SAS-Datei	5 KB
UVEA509_T8_2_1_Demo_D01_0_0.sas	30.12.2021 15:28	SAS-Datei	5 KB
UVEA509_T8_2_2_BaselineCont_D01_0_0.sas	30.12.2021 15:28	SAS-Datei	5 KB
UVEA509_T8_2_3_BaselineCat_D01_0_0.sas	30.12.2021 15:28	SAS-Datei	5 KB
UVEA509_T8_2_4_MedHist_D01_0_0.sas	30.12.2021 15:28	SAS-Datei	5 KB
UVEA509_T8_2_5_ConDis_D01_0_0.sas	30.12.2021 15:28	SAS-Datei	5 KB
UVEA509_T8_2_6_ConProc_D01_0_0.sas	30.12.2021 15:28	SAS-Datei	5 KB
UVEA509_T8_3_1_IOLCalcCont_D01_0_0.sas	30.12.2021 15:28	SAS-Datei	5 KB
UVEA509_T8_3_2_IOLCalcCat_D01_0_0.sas	30.12.2021 15:28	SAS-Datei	5 KB
UVEA509_T8_3_3_SurgCont_D01_0_0.sas	30.12.2021 15:29	SAS-Datei	5 KB
UVEA509_T8_3_4_SurgCat_D01_0_0.sas	30.12.2021 15:29	SAS-Datei	5 KB
UVEA509_T8_4_PrimEnd_D01_0_0.sas	30.12.2021 15:29	SAS-Datei	5 KB
UVEA509_T8_5_1_1_CDVA_D01_0_0.sas	30.12.2021 15:29	SAS-Datei	5 KB
UVEA509_T8_5_1_2_CDVACat_D01_0_0.sas	30.12.2021 15:29	SAS-Datei	5 KB
UVEA509_T8_5_1_3_CDVAResp_D01_0_0.sas	30.12.2021 15:29	SAS-Datei	5 KB
UVEA509_T8_5_1_4_UDVA_D01_0_0.sas	30.12.2021 15:29	SAS-Datei	5 KB
UVEA509_T8_5_1_5_UDVACat_D01_0_0.sas	30.12.2021 15:29	SAS-Datei	5 KB
UVEA509_T8_5_1_6_UDVAResp_D01_0_0.sas	30.12.2021 15:30	SAS-Datei	5 KB
UVEA509_T8_5_1_7_Indices_D01_0_0.sas	30.12.2021 15:30	SAS-Datei	5 KB
UVEA509_T8_5_2_1_SubRefCont_D01_0_0....	30.12.2021 15:30	SAS-Datei	5 KB
UVEA509_T8_5_2_2_SubRefCat_D01_0_0.sas	30.12.2021 15:30	SAS-Datei	5 KB
UVEA509_T8_5_2_3_SubRefCylPow_D01_0...	30.12.2021 15:30	SAS-Datei	5 KB
UVEA509_T8_5_3_ObjRef_D01_0_0.sas	30.12.2021 15:30	SAS-Datei	5 KB
UVEA509_T8_6_1_1_IOLStatCat_D01_0_0.sas	30.12.2021 15:30	SAS-Datei	5 KB
UVEA509_T8_6_1_2_IOLStatCont_D01_0_0....	30.12.2021 15:30	SAS-Datei	5 KB
UVEA509_T8_6_2_PCO_D01_0_0.sas	30.12.2021 15:30	SAS-Datei	5 KB
UVEA509_T8_6_3_SLE_D01_0_0.sas	30.12.2021 15:31	SAS-Datei	5 KB
UVEA509_T8_6_4_EyeInf_D01_0_0.sas	30.12.2021 15:31	SAS-Datei	5 KB
UVEA509_T8_6_5_1_ECC_D01_0_0.sas	30.12.2021 15:31	SAS-Datei	5 KB
UVEA509_T8_6_5_2_ECC_D01_0_0.sas	30.12.2021 15:31	SAS-Datei	5 KB
UVEA509_T8_6_6_IOP_D01_0_0.sas	30.12.2021 15:31	SAS-Datei	5 KB
UVEA509_T8_6_7_1_OverallAEs_D01_0_0.sas	30.12.2021 15:31	SAS-Datei	5 KB
UVEA509_T8_6_7_2_AESev_D01_0_0.sas	30.12.2021 15:31	SAS-Datei	5 KB
UVEA509_T8_6_7_3_AERelIOL_D01_0_0.sas	30.12.2021 15:31	SAS-Datei	5 KB
UVEA509_T8_6_7_4_AERelProc_D01_0_0.sas	30.12.2021 15:32	SAS-Datei	5 KB
UVEA509_T8_6_7_5_SAE_D01_0_0.sas	30.12.2021 15:32	SAS-Datei	5 KB
UVEA509_T8_6_8_SecSurg_D01_0_0.sas	30.12.2021 15:32	SAS-Datei	5 KB
UVEA509_T8_6_9_DD_D01_0_0.sas	30.12.2021 15:32	SAS-Datei	5 KB

Figure 2 Automatically created dummy programs for UVEA509. All programs were defined by the information transported by the mock file

Check your input and then hit “Run” (Ctrl+Enter). If you like, you can open your output_path folder and watch it being filled with SAS-programs.

Figure 2 shows the result of the script running the mocks of UVEA509. All 40 table programs were initialized in approx. 5 minutes, containing (almost) complete headers, program definitions, footnotes, titles, output destinations, basic PROC REPORT structures and workspace clean-up.

Contents of an automatically creates SAS-file

Below you find the output SAS-Program created by the mock for UVEA839 table 8.5.1.13 (edited to fit a the page).

```
/*************************************************************************
 Programme name:               UVEA839_T8_5_1_13_DCNVA_D01_0_0.sas
 Programme language:           SAS 9.4
 Initial date:                 14/DEC/2021
 Sponsor | study:              Carl Zeiss Meditec | UVEA839
 Author(s):                    RPA
************************************************************************
  Short description:    Table 8.5.1.13 -  Secondary endpoints
                                          Visual acuity
                                          Postoperative DCNVA

 Requirements:                 ---
 Risk assessment:              Low
************************************************************************
 Input:                 <TBD> dataset
 Output:                Table in RTF document
 Required programmes:   current version of master programme
                        'UVEA839_MasterProg_TFLs'
 ************************************************************************
 Document history :
 Version    Date        Author            Purpose
 D01_0_0    14/DEC/2021 RPA               First initiation
 ************************************************************************/

 ****************************************************************************;
 **********************        General settings      ************************;
 ****************************************************************************;

 *** Change the output destination to SAS and empty the log as well as the
output window;
 ODS LISTING;
 DM LOG 'CLEAR';
 DM OUTPUT 'CLEAR';
 ODS EXCLUDE ALL;

 *** Clear old titles and footnotes;
 TITLE;
```

Note that the header is automatically prepopulated with the correct contents.

This is somewhat of a preview of the output title

<TBD> needs to be replaced by the respective datasets that need to be used in the program.

The date of file creation is used in the document history

Standard LOG-cleaning and ODS options

```
 FOOTNOTE;

 *** Name of this program;
 %LET THIS_PROG= UVEA839_T8_5_1_13_DCNVA_D01_0_0;
 %LET THIS_TOC_LABEL = Table 8.5.1.13;
 %LET DS_NAMES = UVEA839_T8_5_1_13;
 %LET CAPTION_LVL1 = Effectiveness endpoints;
 %LET CAPTION_LVL2 = Visual acuity;
 %LET CAPTION_LVL3 = Postoperative DCNVA in LogMAR;
```

Here, some MACRO variables are defined that are used below in PROC REPORT. After validation, “D01” in THIS_PROG needs to be changed to “V01”.

```
*****************************************************************************;
********************        Programme start          ************************;
*****************************************************************************;
```

This is where the actual programming goes. Currently, this is where you actually have to du manual work. Sorry.

```
*****************************************************************************;
********************        Report                   ************************;
*****************************************************************************;


 %IF &OUTPUT. EQ FALSE %THEN %DO;
      ODS TAGSETS.RTF FILE="Z:/STUDIES/ZEISS/UVEA839/WORKSPACE/
07_STATISTIC/04_Statistical_Analysis/01_Final/02_Output/01_Tables/&THIS_PROG..
rtf"
      STYLE=gcpservice_style;
%END;

ODS ESCAPECHAR="^";
OPTIONS NODATE NONUMBER NOCENTER; ODS EXCLUDE NONE;
```

The &OUTPUT.-variable needs to be set in the master program, it is used as a switch for the complete TFL-output (OUTPUT = TRUE) or single execution of SAS-Programs (OUTPUT = FALSE). This way, the ODS output stream doesn’t need to be commented out after validation.

```
ods proclabel="&THIS_TOC_LABEL: &CAPTION_LVL3.";
```

ODS PROCLABEL defines the entry in the table of contents

```
TITLE1 j=l height=10pt font=courier "&THIS_TOC_LABEL:^R/RTF'\tab'
&CAPTION_LVL1.";
 TITLE2 j=l height=10pt font=courier "^R/RTF'\tab\tab\tab' &CAPTION_LVL2.";
 TITLE3 j=l height=10pt font=courier "^R/RTF'\tab\tab\tab' &CAPTION_LVL3.";
```

In the final output, check if the number of \tab is enough/to high for decent output

```
footnote1 j=l height=8pt font=courier "n: Number of non-missing observations;
SD: Standard deviation; DCNVA: Distance Corrected Near Visual Acuity; ";
footnote2 j=l height=8pt font=courier "Modified intent-to-treat population
(mITT): Following the modified intention-to-treat (mITT) principle, the
modified intent-to-treat population (mITT) will include all subjects from the
SAF who have at least one measurement for the primary effectiveness
endpoint;";
 footnote4 j=l height=8pt font=courier "Output generated by program
'&this_prog.';";
```

All footnotes are displayed here. Every row of the footnote table has its own FOOTNOTE statement

```
 ods listing close;
 PROC REPORT DATA=UVEA509_T8_1_1_1_D99 HEADLINE HEADSKIP NOWD SPLIT='|'
MISSING CONTENTS='';
       COLUMNS (DUMMY (
            EMPTY_COLUMN
       ));
       DEFINE DUMMY       /         ORDER        NOPRINT;
       define empty_column/ ""                   style(column)=[cellwidth=0.2
      cm];


       compute before;
             line ' ';
       endcomp;

       compute after dummy;
            line ' ';
       endcomp;

       break before dummy / contents="" page;
 run;

 footnote; title;
```

A basic PROC REPORT structure is created. The dataset in DATA needs to be adapted, since it is not possible to foresee the actual name of the dataset used for output. Also as of now, there is no way to get the number of columns and their headers from the mocks. They need to be added manually as well.

```
 %IF &OUTPUT. EQ FALSE %THEN %DO;
 ods tagsets.rtf close;
 ods _all_ close;

 %END;
```

Again, the output stream can be toggled on/of by OUTPUT

```
*** Cleaning up;
PROC DATASETS NOLIST;
      DELETE UVEA839_T8_5_1_13_D:;
QUIT;

****************************************************************************;
**********************        Programme End           **********************;
****************************************************************************;
```

This deletes all working datasets specific to this program.

Conclusion

While being developed as an easy method to create template SAS programs for multiple similar studies, it turned out that even the early prototype of this process reduced the amount of work per program that needed to be invested in setting up the header and the basic structure to a minimum. At the same time, transfer errors, typos, oversights were largely avoided.

While the script and the process behind it is still under development and may still have its shortcomings, its potential is already visible: in the future it may be possible to produce programs simple descriptive statistic analyses directly from their respective mocks – ready to be run with minor adjustments necessary.