Paper SM11

How to handle huge amounts of output tables
A systematic approach

Nico Schreiber, ClinStat GmbH, Huerth, Germany

ABSTRACT

In the context of clinical programming, a key objective is the consolidation of all tables, listings and figures (TLF) into one or more comprehensive documents. Managing potentially hundreds or even thousands of output tables, particularly in the context of Health Technology Assessment (HTA) analysis, poses a considerable organizational challenge. To address this issue effectively, this paper presents a systematic approach to handle this huge amount of outputs in SAS® 9.4.

This methodology utilizes a set of macros designed to manage table metadata, facilitating the seamless and automatic creation of the final TLF documents. Although it was originally designed for a large number of outputs, it can also be easily adapted to a smaller number of outputs.

Alongside the substantial time savings in the process, this systematic approach additionally reduces the risk of errors associated with manual execution, thereby amplifying the efficiency and reliability of clinical programming workflows.

INTRODUCTION

There are various methods to evaluate clinical trials, all of which result in a large number of tables and figures that must be included in a final report. This is particularly true for phase III and IV studies, where the volume of outputs can grow exponentially with the evaluation of subgroups. Without accurate documentation, organizational challenges are inevitable. Issues such as determining the correct sorting order and appropriately placing section titles require manual intervention which leads to an increasing risk of errors.

When a single program is used to generate all outputs and consolidate them into a report, it can become overly long and complex, as it must handle the creation of numerous tables and figures. Alternatively, when multiple programs produce various outputs (e.g., RTFs, PNGs), these need to be merged into one document in the correct sequence. Regardless of the method employed, manual compilation of the final report becomes increasingly difficult, time-consuming, and error-prone as the number of outputs increases.

This paper presents a systematic approach to efficiently manage and organize a large number of outputs, culminating in the creation of a final TLF (Tables, Listings, and Figures) document. While this approach is demonstrated using the SAS programming language, the concepts can be adapted to other programming languages as well.

A SYSTEMATIC APPROACH

There are various approaches to designing a study evaluation workflow, but this paper focuses on two common methods. The first method involves using multiple programs to generate separate output files in formats such as RTF or PNG, which are later consolidated into a single final report document.
The second method involves running multiple programs through a main program that immediately incorporates all outputs into a single report, eliminating the need to create interim output files.

Regardless of the method used, the systematic approach outlined in this paper can be applied to any evaluation process that involves using multiple programs to produce a final report document.

STUDY EVALUATION PROCESS

To illustrate the concept of this automation process, two different study processes are described here. On the one hand there could be several programs generating output tables and figures, which are then consolidated into a single final TLF document. On the other hand, there could be a main program which executes all TLF programs in the correct order to create a final report immediately.

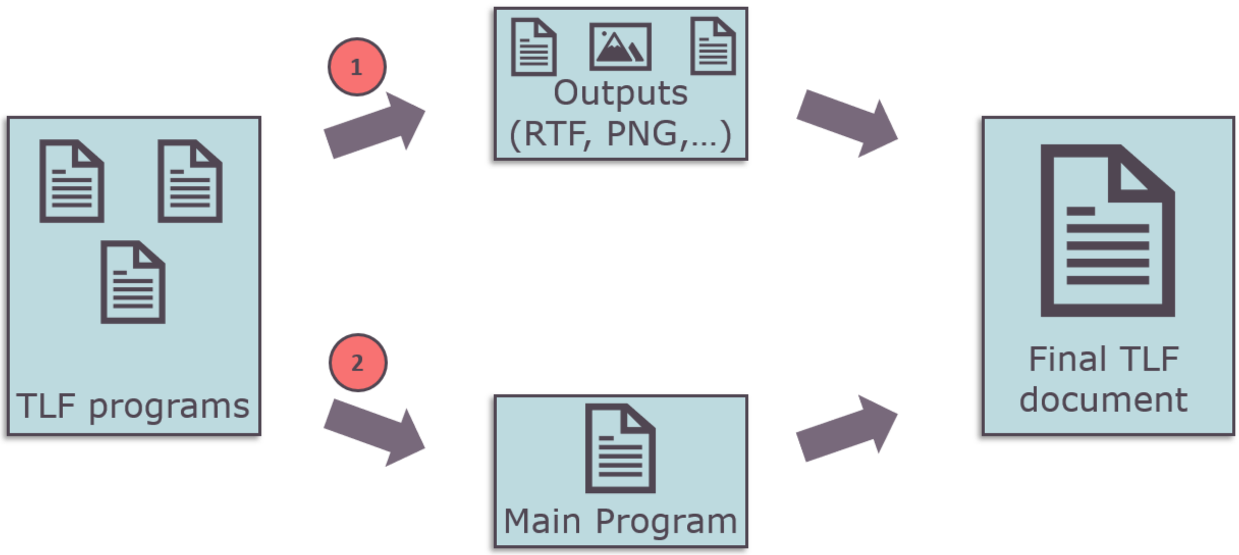

Figure 1: Study evaluation process. The first work flow describes the creation of a TLF document while creating interim output documents. The second work flow describes the creation via a main program which executes all TLF programs in the correct order.

As described in Figure 1, both approaches will result in a final report document. The interim files could be either output documents (1) or a main program which executes all TLF programs (2). These are just two examples of study processes. A combination of both approaches is also possible which could be optimized and automatized too.

PROBLEM DEFINITION

The process outlined in the previous section is generally straightforward to implement. However, several challenges may arise during its execution. First, it's common to generate not just a few, but potentially hundreds of table and figure outputs. Depending on the naming conventions used, program and output file names may not include table or figure numbers, making it difficult to ensure the correct order in the final TLF document. Additionally, different file types, such as RTF or PNG, may require distinct handling during the report compilation process.

These issues create significant organizational challenges, particularly in the context of Health Technology Assessment (HTA) analyses, where large volumes of outputs are produced. To address this, automating the management of these outputs can be a more efficient approach, helping to save time and minimize errors.

IDEA

A key challenge in output creation is the definition of titles, table numbers, file names, and other metadata across multiple locations. For example, the output filename is defined within a program and then referenced again in another file to ensure it appears in the correct position in the report. Similarly, table and figure numbers are used to maintain the correct order. To address this, the proposed approach will centralize this metadata, allowing it to be defined once and reused across multiple processes.

There are two approaches to managing the metadata for tables and figures. One approach is to manually create an overview file, such as an Excel spreadsheet, at the start of the study evaluation. This file would contain all the necessary metadata such as filenames, file types, titles, footnotes, and table numbers for each program and its outputs. However, this method can be error-prone, as each new program and output must be manually added to the file. Additionally, if multiple people are working on the same file, it could create challenges in versioning.

The second approach, and the one advocated in this paper, involves generating the overview file after the program execution. In this method, all metadata related to the outputs - such as titles and footnotes - are stored directly within the individual table programs, ensuring that this information is only defined in one place.

Both approaches ultimately result in the creation of an output overview file. Once this file contains all the necessary information, it serves multiple purposes. First, it provides a clear overview for programmers, statisticians, and other team members, making it easy to track missing files and verify that all titles and footnotes are spelled correctly. Second, it can be used to determine the order of program execution and the placement of outputs in the final report. This second benefit is the central focus of this paper and will be explored in the following sections, starting with an explanation of how the output overview is created.

PROCESS FLOW

Manually creating an output overview at the start of a study evaluation can be both time-consuming and error-prone. As a result, this paper focuses on automating the process. In this approach, programs that generate tables and figures include title and footnote statements to define the metadata for each output. Following these statements, the SASHELP dataset VTITLE can be used to extract all titles and footnotes for a given table into a single dataset. This dataset is then exported into a permanent SAS dataset, where the metadata from all table and figure programs are stored. To ensure that each output is properly linked to its corresponding title and footnotes, the output file- and program name must also be passed into the permanent dataset.

```
title “Table 1-1.1: This is the first table”;

data permlib.output_overview;
   set permlib.output_overview sashelp.vtitle(in=NEW);
   if NEW then do;
      filename = "t_first.rtf";
      progname = "t_first.sas";
   end;
run;

<output creation>

title;
footnote;
```

It is important to ensure that titles and footnotes are reset after each output creation to avoid any mix-up with subsequent outputs. Additionally, the output overview dataset must be appended before a new output is generated. To streamline this process, it would be efficient to create a macro with FILENAME and PROGNAME as its only parameters. The parameter FILENAME can also be retrieved automatically by extracting it out of the SAS global macro variable `&execpath`. To enhance the robustness of the macro and safeguard against potential changes, additional features and validation checks can be implemented. For example, it should be checked if the same output name already exists and the old observation should be deleted to avoid duplicates. Additionally, the dataset should be locked before adapting such that other users cannot update it at the same time.

Once all programs have been executed, the final output overview can be generated. However, before doing so, additional metadata must be extracted from the output titles. To ensure accurate and efficient extraction, the titles should consistently follow the same format. SAS functions such as `SCAN`, `SUBSTR`, or `COMPRESS` can be particularly useful for extracting the required information. In Table 1, two programs *t_first.sas* and *t_second.sas* have been included which created in total three output documents *t_first.rtf, t_first_png* and *t_second_rtf*. The output- and file types have been extracted from the file name, while the table number is received from the title. The section value is received by applying a format on the first digits in front of the first special character (1 → Section 1). The output overview dataset can then be sorted by the table number and output type, using a linguistic sorting sequence. Depending on the study analysis plan, the output type should be sorted either in ascending or descending order.

Table 1: Output overview dataset, filled with two programs that produced three output documents.

Title	File name	Program name	Output type	File type	Table number	Section
Table 1-1.1: First table	t_first.rtf	t_first.sas	table	rtf	1-1.1	Section 1
Figure 1-1.1: First figure	f_first.png	t_first.sas	figure	png	1-1.1	Section 1
Table 2-1.1: Second table	t_second.rtf	t_second.sas	table	rtf	2-1.1	Section 2

Finally, the output overview can be exported as an Excel file using `PROC EXPORT`.
Once the output overview Excel file is created, it can be reviewed by all study team members to identify any issues with titles, footnotes, or table numbers. With all outputs listed in one place, it becomes much easier to detect errors in the metadata compared to reviewing each output individually.

An additional feature that can be incorporated into the output overview is a new SUBSET column, which can be manually or automatically updated using another parameter in the output overview creation. This column allows the selection and extraction of only a subset of outputs that are relevant for the current delivery.

AUTOMATIZATION OF OUTPUT GENERATION

Having an overview of all outputs is valuable not only for monitoring the studyprogramming process but also for automating the creation of the final report. As shown in Figure 1, there are two approaches that can be applied in study evaluations, each with its own advantages and disadvantages. When the process is automated, the primary factor in choosing between these approaches is the tool used to generate the final report. This could be a SAS macro or a tool from another programming language, such as Microsoft VBA or Java. Regardless of the tool used, these two approaches encompass most reporting processes.

1) Creation of single output documents and compilation afterwards.

The assumption is made that all table and figure programs have already been executed to generate individual output documents, which are stored in a separate folder. During execution, the output overview dataset was filled with the required metadata. Afterwards the output overview Excel file was created as outlined in the “Process Flow” section. Depending on the tool used for report creation, this information can be extracted and applied as needed. This step must be tailored to each specific tool, but once implemented, it can be reused across all clinical trials where that report tool is applied.

In the following example, the report creation tool used is called Reindeer® (Glass, 2021), a VBA macro developed for creating TLF documents for clinical study submissions. This macro requires a text file where all individual output documents are listed in the correct order.

Table 2: Part of the output overview in Table 1.

Title	File name	Table number	Section
Table 1-1.1: First table	t_first.rtf	1-1.1	Section 1
Figure 1-1.1: First figure	f_first.png	1-1.1	Section 1
Table 2-1.1: Second table	t_second.rtf	2-1.1	Section 2

```
Header1;Section 1
RTF;t_first.rtf
FigureTitle;Figure 1-1.1: First figure
Figure;f_first.png
Header2;Section2
RTF;t_second.rtf
```

Once this text file is created, it can be used by the Reindeer macro to compile all individual outputs and section headers in the correct order within the final report document.

2) Creation of the final report within one main program

For this approach to work, all table and figure programs must be executed at least once by the developer to populate the output overview with the necessary metadata, as described in the "Process Flow" section. If any metadata such as the output file name, program name, or title is modified during development, the program must be re-executed after these changes to ensure the output overview remains up to date. However, if the changes only affect the layout or results within the table and not the metadata, re-execution is not required. It is important to note that generated outputs within one program must appear consecutively in the final report, as they cannot be reordered later.

Once all programs have been executed at least once and the output overview is complete, the creation of the main program can begin. The key technique here is leveraging SAS to generate additional SAS code by writing all required lines into a dataset, which is then exported as a SAS program file. First, the program overview must be converted into correct and meaningful SAS code that can execute all programs in the appropriate order. Note that each program only needs to be run once, regardless of how many outputs it generates.

In this example, the final report will be an RTF file, with all outputs correctly ordered. This is accomplished using the `ODS RTF` statement at the beginning of the program and `ODS RTF CLOSE` at the end. The TLF programs are included in the correct order and if necessary, an additional page including the section number is included before each section. One advantage of this approach is that no output file names or titles are required as described in Table 3.

Table 3: Part of the output overview in Table 1.

Program name	Table number	Section
t_first.sas	1-1.1	Section 1
t_first.sas	1-1.1	Section 1
t_second.sas	2-1.1	Section 2

```
/*Start RTF creation*/
ods tagsets.rtf
    file="C:\Users\nico.schreiber\final_report.rtf"
    startpage  = yes
;

ods escapechar="^";

/*Execute programs*/
ods text="^S={fontsize=24pt}Section 1";
ods startpage=now;

%include "t_first.sas";

ods text="^S={fontsize=24pt}Section 2";
ods startpage=now;

%include "t_second.sas";

/*Finish RTF creation*/
ods rtf close;
```

This code block serves as a minimal example of how the final report can be generated. It illustrates the foundational idea behind automated code creation, which can be adapted to most program layouts.

Once the RTF creation code is complete, it can be executed to produce the final report. If needed, the generated code can be manually adjusted. However, this should only be done in special situations, as any subsequent execution of the code creation macro will overwrite these changes.

DISCUSSION

The use of output metadata, as outlined in this paper, can significantly enhance automation and reduce errors. However, the development, validation, and implementation of this approach can be time-consuming and may not yield immediate visible benefits. Therefore, it is essential to assess whether such an approach is practical and worthwhile for a given company.

The output overview approach can be highly beneficial in certain scenarios, while in others, it may be unnecessary or overly complex. The effectiveness of this approach depends on the nature of the study evaluations and the company's strategic priorities. For instance, if there are only a few study evaluations with a limited number of output tables and figures spread over an extended period, implementing the output overview approach might be too time-consuming. In such cases, it may be more efficient to manually create the output overview or develop the main program to run all individual programs.

Conversely, there are two key scenarios where this approach proves invaluable. First, when multiple study evaluations are conducted within a short timeframe, this approach can significantly reduce the time required to create the output overview and the main program automatically. Second, in clinical trials that require the generation of hundreds of output tables and figures, such as HTA analyses, this approach greatly reduces the risk of errors, minimizing the likelihood of missing outputs in the final report. Therefore, it is essential to critically assess the demand for this approach based on specific project requirements.

As discussed in previous sections, the output overview approach offers many advantages in various contexts and can be integrated into different evaluation processes. The following lists outline the advantages and disadvantages of this approach.

Advantages

- Fast creation of the final report document
- Significantly less error-prone
- Adaptable to various study processes
- Suitable for clinical trials of any complexity
- Ensures accurate sorting of tables and figures
- Initial macro implementation is straightforward
- Flexible naming conventions for program files

Disadvantages

- Requires development and validation of processing macros
- Implementing flexibility within macros can be challenging

CONCLUSION AND OUTLOOK

The approach outlined in this paper aims to streamline the study evaluation process by reducing processing time, enhancing efficiency, and minimizing errors. However, it is essential to assess whether this approach is necessary or beneficial for a specific study evaluation process. In most cases, the advantages will likely outweigh the disadvantages. While the initial implementation of this approach can be completed relatively quickly, achieving greater flexibility and making further improvements may present challenges.

At ClinStat GmbH, this approach has been in use for two years, significantly simplifying and easing the report creation process. The required macros are straightforward and user-friendly, enabling even beginners to effectively use this report creation method without difficulty.

The concepts presented here represent just a fraction of the potential applications. The output overview can be valuable in various other scenarios, such as interim study evaluations, focusing on a subset of outputs, or serving as an additional information sheet for sponsors, detailing existing outputs along with their titles and footnotes. Therefore, the enhancement of the output overview will continue, aiming to achieve more accurate and error-resistant results in clinical study evaluations.

REFERENCES

- Glass, K. (2021). Katja Glass Consulting, https://github.com/KatjaGlassConsulting/reindeer

ACKNOWLEDGMENTS

The concept and implementation of this approach were developed collaboratively by Kevin Nguyen and myself. I am deeply grateful to Kevin for his invaluable contributions and the many insightful discussions we had during the macro development and refinement process. I would also like to extend my sincere thanks to Katrin Grabowski and Laura Marquis for their support in optimizing this paper.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

nico.schreiber@clinstat.eu