PharmaSUG 2023 Paper PO-258

Using R to Automate Clinical Trial Data Quality Review

Melanie Hullings, TrialSpark, New York, NY;
Emily Murphy, TrialSpark, San Francisco, CA;
Andrew Burd, TrialSpark, Durham, NC;
Derek Lawrence, TrialSpark, Chapel Hill, NC;
Michelle Cohen, TrialSpark, Houston, TX

ABSTRACT

Over the past 20 years, the detection of the majority of clinical research site quality issues has been the remit of Clinical Research Associates (CRAs) performing on-site monitoring. While there have been advances that enable some remote data review of electronic data capture (EDC) systems, there remains a large amount of manual work by CRAs and Clinical Data Managers (CDMs). Deviations from clinical trial requirements, including critical quality issues such as drug noncompliance and patients who were treated but did not meet all eligibility criteria, are logged in order to track and monitor data integrity and patient safety. To more efficiently facilitate detection, our CDM team created a script in R Studio which can surface potential protocol deviations in the EDC dataset. The script produces a list of findings for CRAs to confirm with sites and enter into the deviation database. Initial beta testing is in progress for a Phase II clinical trial and has already identified additional protocol deviations not detected during manual review. While this process will always require manual effort by CRAs to monitor at the site-level and confirm, programmatic issue detection has the potential to decrease manual review burden on study teams and improve issue identification and clinical trial data quality overall. Future directions include refining, expanding, and standardizing scripts and productizing these features in the EDC.

INTRODUCTION

Current industry practices for clinical data monitoring remain a highly manual process. While there have been recent advances in terms of the number of tasks and volume of data that can be performed and monitored remotely, there is still a large bolus of work that remains for which Clinical Research Associates (CRAs) rely on tried-and-true manual processes to ensure quality.

Traditional methods for detecting clinical trial deviations, including participant eligibility, involve manual review of:

- source at the clinical trial site by the Clinical Research Associates (CRAs)
- data collected in the Electronic Data Capture (EDC) system by the CRAs and Clinical Data Managers (CDM)
- EDC and vendor data system exports by the CDMs, often in Excel.

This is a very expensive and tedious process by which it is determined whether each study visit and assessment was completed within the required timeframe and meets all research protocol requirements such as patient position during collection of vital signs. Often staff turnover of CDMs and CRAs can disrupt timely data review, however once the program is written it is easy for another study team member to easily review the script output and take corrective action. Confirmed deviations are then typically recorded in a separate database used to produce the final deviation dataset for trial analysis.

METHODS

To increase the speed and accuracy of protocol deviation detection, our Clinical Data Management (CDM) team created a script in RStudio to programmatically detect deviations in the EDC datasets. Specifications were authored by the CDM team to define logic for inclusion/exclusion, visit window, and missed assessment checks based on protocol deviation details contained in the protocol. In total, the script checks 52 unique deviations. The output will only contain rows for positively identified protocol deviations. If a participant does not have any protocol deviations, it will not appear in this output. The output is a detailed Excel file with a Subject, Visit, and Deviation granularity which is then distributed to CRAs for final confirmation.

SubjectID	Visit	PD Name	Detail
000001	SCR	Lab test not done	Other
000001	D1	Lab test not done	Patient Refused
000003	D1	Lab test not done	Other
000001	SCR	Visit out of window	Screening is 3 days before the visit window. Visit date is 2023-01-07 the minimum is 2023-01-30 and the maximum is 2023-02-28
000001	Day28 (EOT)	Visit out of window	Day 28 (EOT) is 7 days after the visit window. Visit date is 2023-04-10 the minimum is 2023-03-24 and the maximum is 2023-04-03

Table 1. Example PD Script Output

PARAMETERIZED FUNCTIONS FOR SCRIPT REUSABILITY

38 of the 52 programmed protocol deviations fall into two general logic patterns: prohibited value checks and dates out of permitted window checks. The logic for these two checks were parameterized and turned into functions to increase the reusability of the script. Reusability is key to a high ROI on programming investment. Since protocol deviations can be highly specific for a given protocol, these generalized functions will allow this script to be quickly applied to future studies.

Each function outputs the SUBJECTID, VISIT, PD_NAME, and DETAIL for each protocol deviation checked. Functions only output a positive identification of a protocol deviation. After all functions have been run, the outputs from those functions are row bonded together to create the final PD script output.

Prohibited Value Function

The *prohibited value function* checks if a particular column of interest in a EDC table is equal to a prohibited value. If so, this will trigger a protocol deviation to be displayed in the output. The function will also pull some relevant details to help the CRA investigate the protocol deviation: the visit and the reason for deviation if that exists in the CRF.

The function takes 4 variables as input: the column of interest from the EDC table, the name of the EDC table, the value of the column that constitutes a protocol deviation, and the name of the column that contains the reason for protocol deviation if applicable. If there is a protocol deviation for a particular SUBJECTID-VISIT, the relevant information will be outputted. If there is no protocol deviation, then there will be nothing flagged in the output for that SUBJECTID-VISIT.

Example

To test is a particular lab was completed, the programmer will pass the following information to the *prohibited value function*:

```
PD_Example_1 <- prohibited_value(table = LAB,
                                 variable_name = LABYN,
                                 PD_value = "NO",
                                 reason = LBREASND)
```

LAB Input			
SUBJECTID	**VISIT**	**LABYN**	**LBREASND**
000001	SCR	NO	Other

LAB Input			
000001	D1	NO	Patient Refused
000002	SCR	YES	
000003	D1	NO	Other

Table 2. Example Input Lab Dataset Passed to the *Prohibited Value Function*

PD Script Output			
SUBJECTID	**VISIT**	**PD_NAME**	**DETAIL**
000001	SCR	Lab test not done	Other
000001	D1	Lab test not done	Patient Refused
000003	D1	Lab test not done	Other

Table 3. *Prohibited Value Function* Example Output

Dates Out of Permitted Window

The *dates out of the permitted window function* checks that a given visit date does not take place before or after the permitted window of that visit as per the protocol. In order to perform this logic, permissible visit windows defined in the protocol are applied to the patient's actual Day 1 visit. These window calculations are only possible if the patient has had a Day 1 visit. The window table is then joined to EDC data on SUBJECTID-VISIT to pull in the actual visit date. This is then fed to the function to identify out of window visits as well as the number of days before or after the window the visit falls.

Example

To test if a particular visit is out of window, the programmer will first construct the window table, and then pass the following information to the *dates out of permitted window function*:

```
PD_Example_2 <- dates_out_of_permitted_window(table = WINDOW,
                                             VISIT_NAME = “Day 1”)
```

Visit	Window
Screening	-30 to -1
Day 1	0 days
Day 14	+/- 2 days
Day 21	+/- 2 days
Day 28 (EOT)	+/- 5 days

Table 4. Example Windowing Conditions in the Protocol

Visit	Visit.Date	Visit.Expected	Visit.Min	Visit.Max
Screening	Visit date from CRF	NA	Day1 - 30	Day1 - 1
Day 1		NA	NA	NA

Day 14		Day1 + 14	Visit.Expected - 2	Visit.Expected + 2
Day 21		Day1 + 21	Visit.Expected - 2	Visit.Expected + 2
Day 28 (EOT)		Day1 + 28	Visit.Expected - 5	Visit.Expected + 5

Table 5. Specifications for the WINDOW Table

WINDOW Input					
SUBJECTID	**VISIT**	**ACTUAL_VISIT_DATE**	**EXP_VISIT_DATE**	**VISIT_MIN**	**VISIT_MAX**
000001	SCR	2023-01-27	NA	2023-01-30	2023-02-28
000001	Day1	2023-03-01	NA	NA	NA
000001	Day14	2023-03-16	2023-03-15	2023-03-13	2023-03-17
000001	Day21	2023-03-22	2023-03-22	2023-03-20	2023-02-24
000001	Day28 (EOT)	2023-04-10	2023-03-29	2023-03-24	2023-04-03

Table 6. Example Input Window Table Passed to the *Dates out of Permitted Window Function*

PD Script Output			
SUBJECTID	**VISIT**	**PD_NAME**	**DETAIL**
000001	SCR	Visit out of window	Screening is 3 days before the screening window. Visit date is 2023-01-07 the minimum is 2023-01-30 and the maximum is 2023-02-28
000001	Day28 (EOT)	Visit out of window	Day 28 (EOT) is 7 days after the visit window. Visit date is 2023-04-10 the minimum is 2023-03-24 and the maximum is 2023-04-03

Table 7. Example PD Script Output for Checking Day1 Visits for All Participants

RESULTS AND DISCUSSION

SENSITIVITY AND UTILITY OF SCRIPTED DEVIATION DETECTION

The script identified 33 net new deviations over 2 months that were not previously identified by CRAs monitoring the data. This demonstrates that R scripts can be used effectively to automatically detect issues in clinical trial datasets including protocol deviations and participant eligibility. However, 63% of potential issues were not confirmed to be protocol deviations after manual review. This was expected as the script was designed to have an intentionally low sensitivity and optimize for detecting all true positives. While this low positive predictive value does result in a larger number of potential issues to review, it ensures that as many deviations as possible are identified by the script. The script can continue to be refined to be more precise using additional data elements and logic, but there must always be a balance between deviation identification and CRA workload. For example, the script currently identifies any prohibited concomitant medications regardless of those allowed with a specific washout period to have a wider net and ensure any potentially prohibited medication is reviewed manually. It is critical that eligibility deviations like this that can impact the clinical outcomes are identified, so cases like this drive the need for low sensitivity.

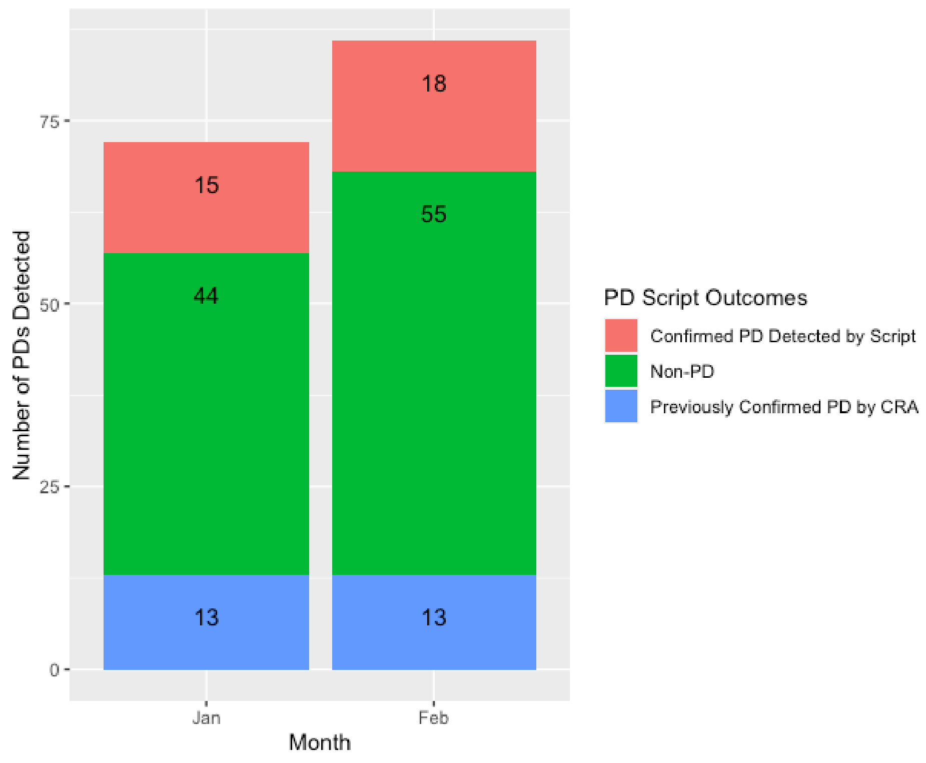

Figure 1. PDs by Outcome Category for January and February

Month	PPV
Jan	0.39
Feb	0.36

Table 8. Positive Predictive Value of the PD Script by Month

Process Improvement of Trial Operations Deviation Management

Programmatic issue detection has the potential to decrease manual review burden of study teams and improve issue identification and clinical trial data quality overall. Though there is an opportunity cost of identifying potential issues that are not deviations, scripted detection still reduces the manual work of CRAs reviewing all source and EDC data. There is also a risk that deviations are missed during manual review as CRAs are also reviewing for data integrity and not primarily for deviation identification. There is also a limit to how many deviations can be detected using structured data, but it still reduces the workload and the true value is in detecting deviations that would otherwise be missed by manual review. This has not yet been explored, but there is also the potential that scripts can identify other non-PD issues that require followup action that aids in monitoring and quality. Overall, programmatic detection will always be less work than relying on CRAs to manually identify and confirm all protocol deviations.

Conclusion

R scripts can be used to automatically detect issues in clinical trial datasets including protocol deviations and participant eligibility. Programmatic issue detection has the potential to decrease manual review burden of study teams, as well as cost, and improve issue identification and clinical trial data quality overall. Future directions include refining scripts to make issue identification more accurate and expanding the number of data issues that scripts are detecting.

Contact Information

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

Melanie Hullings
TrialSpark
mhullings@trialspark.com

Emily Murphy
TrialSpark
emurphy@trialspark.com