Paper OS16

Harnessing Analysis Results Datasets (ARD) for Clinical Reporting in R: Our First ARD-Based Filing Experience

Daniel D. Sjoberg, Genentech, South San Francisco, California, USA
Nolan Steed, Denali, South San Francisco, California, USA

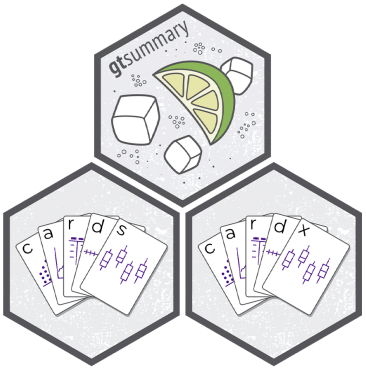
ABSTRACT

We present a pioneering application of the Analysis Results Dataset (ARD) framework for preparing a clinical trial readout. By leveraging open-source R packages, we have demonstrated a streamlined and reproducible approach to generating comprehensive trial results. The newly released {cards} package developed by Roche, GSK, and Novartis builds ARDs, which can be seamlessly and robustly be converted into visually appealing tables utilizing the {gtsummary} package. The recent integration of ARD functionality into {gtsummary}, the most popular package for summary tables in the R ecosystem, marks a significant advancement for pharmaceutical companies. This integration underscores the growing importance of ARD as a standardized format for representing statistical analysis results. Our experience preparing a health authority filing highlights the benefits of using ARDs, including improved automation, reproducibility, reusability, and traceability. We believe that this approach will become increasingly essential for ensuring the quality and efficiency of clinical trial reporting.

INTRODUCTION

The efficient and reproducible reporting of clinical trial results is crucial for timely regulatory submissions and informed decision-making. Traditional methods of presenting these results in static, PDF-based reports often suffer from limitations in navigability, standardization, and hinder automation, traceability, and reusability. To address these challenges, the CDISC Analysis Results Standard (ARS) Model was developed, encompassing the Analysis Results Metadata Technical Specification (ARM-TS) and the Analysis Results Dataset (ARD). While the ARM-TS defines the metadata for describing and organizing analysis results, the ARD provides a standardized format for the analysis results data itself. This structured format offers a predictable and consistent structure, facilitating sharing and reuse across different systems and organizations. As we demonstrate in this paper, leveraging the ARD framework, particularly in conjunction with open-source R packages, offers a powerful solution for streamlining clinical trial reporting. Specifically, we present our pioneering experience applying ARDs, generated via the newly released {cards} package (developed by Roche, GSK, and Novartis), and seamlessly converting them into visually appealing tables using the {gtsummary} package. Figure 1. The recent integration of ARD functionality directly into {gtsummary}, the most popular R package for summary tables, further underscores the growing importance of the ARD as a standardized format for representing statistical analysis results and marks a significant advancement for the pharmaceutical industry. Our experience preparing a health authority filing using this approach highlights the tangible benefits of ARDs, including improved automation, reproducibility, reusability, and traceability, ultimately contributing to the quality and efficiency of clinical trial reporting.

Figure 1. R packages {gtsummary}, {cards}, and {cardx} work together to create summary tables.



ANALYSIS RESULTS DATASETS

There are two primary R packages for creating ARDs: {cards} and {cardx}. The {cards} R package creates ARD from observed data sets, and provides utilities for working with these objects. The {cardx} package builds upon {cards} and exports functions for creating more complex ARDs, typically from statistical analyses.

R PACKAGE {cards}

The simplest way to introduce the {cards} R package is with an example of its most basic functionality. In the example below, we are using the ADSL example data set that is included in the package and we are calculating basic summary statistics for continuous variables `"AGE"` and `"BMIBL"`.

```
library(cards)

ADSL |>
  ard_continuous(by = ARM, variables = c(AGE, BMIBL))

{cards} data frame: 48 x 10

   group1 group1_level variable stat_name stat_label statistic
1     ARM      Placebo      AGE         N          N        86
2     ARM      Placebo      AGE      mean       Mean    75.209
3     ARM      Placebo      AGE        sd         SD      8.59
4     ARM      Placebo      AGE    median     Median        76
5     ARM      Placebo      AGE       p25  25th Per…        69
6     ARM      Placebo      AGE       p75  75th Per…        82
7     ARM      Placebo      AGE       min        Min        52
8     ARM      Placebo      AGE       max        Max        89
9     ARM      Placebo    BMIBL         N          N        86
10    ARM      Placebo    BMIBL      mean       Mean    23.636

i 38 more rows
i Use `print(n = ...)` to see more rows
i 4 more variables: context, statistic_fmt_fn, warning, error
```

A few items to note from this result:

- The default statistics returned are N, Mean, Standard Deviation, Median, 25th and 75th percentiles, and the minimum and maximum: these can be modified to use any univariate statistic whether that function is user-defined or from another package.
- These results are calculated by the treatment arm. There is no requirement to include the ard_continuous(by) argument.
- Any unobserved levels of the by= column(s), whether that is unobserved combinations of the by= variables or unobserved factor levels, will appear in the results ARD table. The results are returned a structured data frame common among all ard_*() functions.

There exists similar functionality for categorical data.

```
ard_categorical(ADSL, by = ARM, variables = AGEGR1)
```

```
{cards} data frame: 27 x 11
```

```
   group1 group1_level variable variable_level stat_name stat_label statistic
1     ARM      Placebo   AGEGR1            <65         n          n        14
2     ARM      Placebo   AGEGR1            <65         N          N        86
3     ARM      Placebo   AGEGR1            <65         p          %     0.163
4     ARM    Xanomeli…   AGEGR1            <65         n          n        11
5     ARM    Xanomeli…   AGEGR1            <65         N          N        84
6     ARM    Xanomeli…   AGEGR1            <65         p          %     0.131
7     ARM    Xanomeli…   AGEGR1            <65         n          n         8
8     ARM    Xanomeli…   AGEGR1            <65         N          N        84
9     ARM    Xanomeli…   AGEGR1            <65         p          %     0.095
10    ARM      Placebo   AGEGR1            >80         n          n        30
```

```
i 17 more rows
i Use `print(n = ...)` to see more rows
i 4 more variables: context, statistic_fmt_fn, warning, error
```

As a result of the common structure between this result and the continuous variable summary above, these two data frames can be combined with `bind_ard()`.

It is common that errors or warnings be returned by functions performing these calculations. The {cards} package will continue to return a data frame of the expected structure. Where a statistic that could not be calculated would have appeared, we will now see a `NULL` value, and the error will be captured and returned in the `"error"` column.

```
mean_with_error <- function(x) {
  stop("There was an error calculating the mean.")
  mean(x)
}

ard_with_error <-
  ard_continuous(
    ADSL,
    variables = AGE,
    statistics = ~list(mean = mean_with_error)
  )
ard_with_error
```

```
{cards} data frame: 1 x 8
```

```
  variable   context stat_name stat_label statistic     error
1      AGE continuo…      mean       Mean      NULL There wa…
```

```
i 2 more variables: statistic_fmt_fn, warning
```

The {cards} package exports many utilities for working with ARDs. The example below is a utility to print any errors or warnings that may have occurred while calculating the statistics.

```
print_ard_conditions(ard_with_error)
```

```
The following errors were returned while calculating statistics:
✖ For variable `AGE` and "mean" statistic: There was an error calculating
the
  mean.
```

EXTRA ARDs WITH {cardx}

As indicated above, the {cards} package exports many utilities for working with ARD objects. Additionally, {cards} exports utilities for creating new `ard_*()` functions. The {cardx} package takes advantage of this infrastructure, and exports many other functions for creating more complex ARD objects.

Utilizing these utilities from {cards}, we can easily create a function to prepare the results from a t-test.

```
ADSL |>
  # keep two treatment arms for the t-test calculation
  dplyr::filter(ARM %in% c("Placebo", "Xanomeline High Dose")) |>
  cardx::ard_ttest(by = ARM, variable = AGE)
```

```
{cards} data frame: 14 x 9

   group1 variable context   stat_name stat_label statistic
1     ARM      AGE   ttest    estimate  Mean Dif…     0.828
2     ARM      AGE   ttest   estimate1  Group 1 …    75.209
3     ARM      AGE   ttest   estimate2  Group 2 …    74.381
4     ARM      AGE   ttest   statistic  t Statis…     0.655
5     ARM      AGE   ttest     p.value    p-value     0.513
6     ARM      AGE   ttest   parameter  Degrees …   167.362
7     ARM      AGE   ttest    conf.low  CI Lower…    -1.668
8     ARM      AGE   ttest   conf.high  CI Upper…     3.324
9     ARM      AGE   ttest      method     method Welch Tw…
10    ARM      AGE   ttest alternative  alternat… two.sided
11    ARM      AGE   ttest          mu    H0 Mean         0
12    ARM      AGE   ttest      paired  Paired t…     FALSE
13    ARM      AGE   ttest   var.equal  Equal Va…     FALSE
14    ARM      AGE   ttest  conf.level  CI Confi…      0.95

i 3 more variables: statistic_fmt_fn, warning, error
```

The utilities allow us to return, not only the results of the t-test, but rows for each of the arguments. This allows us to both report the results, and also in a re-use case, know exactly how the results were calculated, e.g. assuming equal variances, the level of the confidence interval, etc.

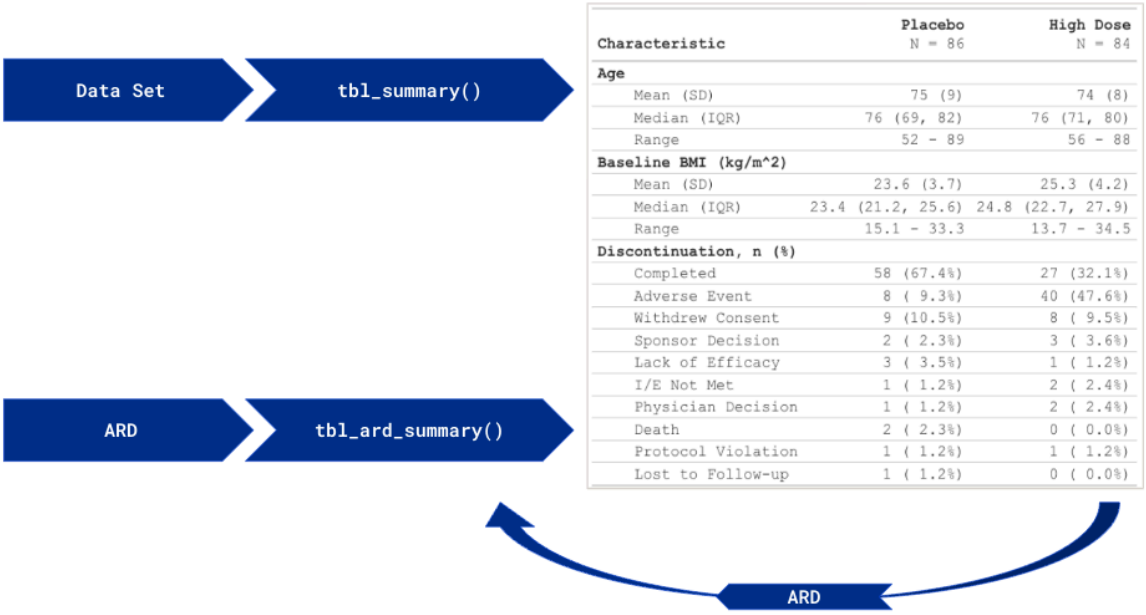
TABLES WITH {gtsummary}

The {cards} package does not present results and this is where the {gtsummary} package shines. The {gtsummary} package offers a modular framework to construct summary tables. The {gtsummary} package is the most widely used package for summary tables in the R ecosystem, won the American Statistical Association's 2021 award for Innovation in Statistical Programming and Analytics, and the submission using {gtsummary} to re-create an entire Clinical Study Report (CSR) won the pharmaceutical track in Posit, PBC's 2024 Table Contest..

The {gtsummary} R package runs on ARDs, meaning that every calculation performed, every number that appears in a {gtsummary} table is associated with an ARD. As a result, a structured ARD can be extracted from any {gtsummary} table.

Additionally, one may first create an ARD and with that ARD create a {gtsummary} table. This workflow is particularly useful for more bespoke tables that are not covered by the suite of `tbl_*()` functions exported by {gtsummary}. Functions with the `tbl_ard_*()` prefix all accept an ARD as its primary input and create summary tables. (Figure 2)

Figure 2. {gtsummary} works with either a data frame-first approach or an ARD-first approach. When a data frame-first approach is utilized, the ARD can be extracted from the table object.



Characteristic	Placebo N = 86	High Dose N = 84
Age		
Mean (SD)	75 (9)	74 (8)
Median (IQR)	76 (69, 82)	76 (71, 80)
Range	52 - 89	56 - 88
Baseline BMI (kg/m^2)		
Mean (SD)	23.6 (3.7)	25.3 (4.2)
Median (IQR)	23.4 (21.2, 25.6)	24.8 (22.7, 27.9)
Range	15.1 - 33.3	13.7 - 34.5
Discontinuation, n (%)		
Completed	58 (67.4%)	27 (32.1%)
Adverse Event	8 (9.3%)	40 (47.6%)
Withdrew Consent	9 (10.5%)	8 (9.5%)
Sponsor Decision	2 (2.3%)	3 (3.6%)
Lack of Efficacy	3 (3.5%)	1 (1.2%)
I/E Not Met	1 (1.2%)	2 (2.4%)
Physician Decision	1 (1.2%)	2 (2.4%)
Death	2 (2.3%)	0 (0.0%)
Protocol Violation	1 (1.2%)	1 (1.2%)
Lost to Follow-up	1 (1.2%)	0 (0.0%)

The {gtsummary} package makes it simple to create complex tables, by allowing users to simply focus on individual sections of a larger, complicated table. Once each simpler section of the table has been created, they can be cobbled together using functions like `tbl_stack()` and `tbl_merge()`. The general structure of a {gtsummary} table also allows any user to construct their own bespoke table-building functions.

Lastly, for the most complex tables that do not follow any common structures the {gtsummary} package exports the `as_gtsummary()` function. This workflow allows users to construct a data frame with the complex summaries, and convert that data frame into a {gtsummary} object. Once converted, the resulting table can be integrated with any other table or used standalone.

HEALTH AUTHORITY FILING

Our initial experience leveraging Analysis Results Datasets (ARDs) in conjunction with the {gtsummary} R package for our first health authority filing proved highly successful, despite a demanding timeline and the absence of formal onboarding. We were able to deliver the complete Clinical Study Report (CSR) on time, meeting all submission deadlines. A key factor in this success was the ability to generate all required tables, even those with non-standard formats, directly from our ARDs using {gtsummary}. This capability streamlined the table creation process significantly, eliminating the need for manual table construction and reducing the risk of errors. Furthermore, the consistent and well-organized structure of the ARDs greatly simplified the Quality Control (QC) process. Notably, a team member without a background in statistical programming was able to independently and efficiently execute the QC steps, demonstrating the accessibility and usability of the ARD + {gtsummary} workflow. Finally, {gtsummary} provided the flexibility to easily export the generated tables into a variety of formats, including HTML, Word, and PDF. This versatility supported our diverse reporting requirements and facilitated seamless integration with various document preparation and submission platforms. The combination of ARDs and {gtsummary} proved to be a powerful and efficient solution for our regulatory reporting needs.

CONCLUSION

In conclusion, the {cards} R package provides a robust and versatile tool for generating CDISC Analysis Results Datasets (ARDs), facilitating automation, reproducibility, and reusability of clinical trial results. Its ability to create ARDs from various statistical analyses, combined with utilities for handling errors and seamless integration with packages like {gtsummary}, positions it as a key component in modernizing clinical trial reporting. This was clearly

demonstrated during our first health authority filing using ARDs and {gtsummary}. Despite a compressed timeline and the absence of formal onboarding, we successfully delivered the complete Clinical Study Report (CSR) on time, meeting all submission deadlines. This success was largely attributed to the streamlined table creation process afforded by {gtsummary}'s direct integration with our ARDs. Generating all required tables, including those with non-standard formats, directly from the ARDs eliminated manual table construction, significantly reducing the risk of errors. Furthermore, the structured nature of the ARDs simplified our Quality Control (QC) process. The flexibility of {gtsummary} to export tables in various formats, including HTML, Word, and PDF, further supported our diverse reporting needs and ensured seamless integration with our submission platform. This experience underscores the practical benefits and significant potential of ARDs, particularly when combined with the capabilities of {gtsummary} and the ease of use provided by {cards}, in streamlining regulatory submissions, enhancing data integrity, and ultimately improving the efficiency and quality of clinical trial reporting.

RECOMMENDED READING

Learn more about constructing ARDs on the {cards} and {cardx} package websites:
https://insightsengineering.github.io/cards/
https://insightsengineering.github.io/cardx/

The {gtsummary} package website includes details on constructing ARD-based tables:
https://www.danieldsjoberg.com/gtsummary/

The slides from the talk given in-person at PHUSE US Connect 2025 can be found here:
https://www.danieldsjoberg.com/ARD-PHUSE-talk-2025/

CONTACT INFORMATION

Daniel D. Sjoberg
Genentech
1 DNA Way, South San Francisco, CA 94080
sjobergd@gene.com
https://www.danieldsjoberg.com/

Nolan Steed
Denali Therapeutics
161 Oyster Point Blvd Suite 200, South San Francisco, CA 94080
steed@dnli.com
www.linkedin.com/in/nolan-steed9