atable: Create Tables for Clinical Trial Reports

by Armin Ströbel

Abstract Examining distributions of variables is the first step in the analysis of a clinical trial before more specific modelling can begin. Reporting these results to stakeholders of the trial is an essential part of a statistician's work. The **atable** package facilitates these steps by offering easy-to-use but still flexible functions.

Introduction

Reporting the results of clinical trials is such a frequent task that guidelines have been established that recommend certain properties of clinical trial reports; see Moher et al. (2010). In particular, Item 17a of CONSORT states that "Trial results are often more clearly displayed in a table rather than in the text". Item 15 of CONSORT suggests "a table showing baseline demographic and clinical characteristics for each group".

The **atable** package facilitates this recurring task of data analysis by providing a short approach from data to publishable tables. The **atable** package satisfies the requirements of CONSORT statements Item 15 and 17a by calculating and displaying the statistics proposed therein, i.e. mean, standard deviation, frequencies, p-values from hypothesis tests, test statistics, effect sizes and confidence intervals thereof. Only minimal post-processing of the table is needed, which supports reproducibility. The **atable** package is intended to be modifiable: it can apply arbitrary descriptive statistics and hypothesis tests to the data. For this purpose, **atable** builds on R's S3-object system.

R already has many functions that perform single steps of the analysis process (and they perform these steps well). Some of these functions are wrapped by **atable** in a single function to narrow the possibilities for end users who are not highly skilled in statistics and programming. Additionally, users who are skilled in programming will appreciate **atable** because they can delegate this repetitive task to a single function and then concentrate their efforts on more specific analyses of the data at hand.

Context

The **atable** package supports the analysis and reporting of randomised parallel group clinical trials. Data from clinical trials can be stored in data frames with rows representing 'patients' and columns representing 'measurements' for these patients or characteristics of the trial design, such as location or time point of measurement. These data frames will generally have hundreds of rows and dozens of columns. The columns have different purposes:

- Group columns contain the treatment that the patient received, e.g. new treatment, control group, or placebo.
- Split columns contain strata of the patient, e.g. demographic data such as age, sex or time point of measurement.
- Target columns are the actual measurements of interest, directly related to the objective of the trial. In the context of ICH E9 ICH E9 (1999), these columns are called 'endpoints'.

The task is to compare the target columns between the groups, separately for every split column. This is often the first step of a clinical trial analysis to obtain an impression of the distribution of data. The **atable** package completes this task by applying descriptive statistics and hypothesis tests and arranges the results in a table that is ready for printing.

Additionally **atable** can produce tables of blank data.frames with arbitrary fill-ins (e.g. X.xx) as placeholders for proposals or report templates.

Usage

To exemplify the usage of `atable`, we use the dataset `arthritis` of **multgee** Touloumis (2015). This dataset contains observations of the self-assessment score of arthritis, an ordered variable with five categories, collected at baseline and three follow-up times during a randomised comparative study of alternative treatments of 302 patients with rheumatoid arthritis.

```
library(atable)
library(multgee)
data(arthritis)
# All columns of arthritis are numeric. Set more appropriate classes:
arthritis = within(arthritis, {
  score = ordered(y)
  baselinescore = ordered(baseline)
  time = paste0("Month ", time)
  sex = factor(sex, levels = c(1,2), labels = c("female", "male"))
  trt = factor(trt, levels = c(1,2), labels = c("placebo", "drug"))})
```

First, create a table that contains demographic and clinical characteristics for each group. The target variables are `sex`, `age` and `baselinescore`; the variable `trt` acts as the grouping variable:

```
the_table <- atable::atable(subset(arthritis, time == "Month 1"),
                            target_cols = c("age", "sex", "baselinescore"),
                            group_col = "trt")
```

Now print the table. Several functions that create a LaTeX-representation Mittelbach et al. (2004) of the table exist: `latex` of **Hmisc** Harrell Jr et al. (2018), `kable` of **knitr** Xie (2018) or `xtable` of **xtable** Dahl et al. (2018). `latex` is used for this document.

Table 1 reports the number of observations per group. The distribution of numeric variable `age` is described by its mean and standard deviation, and the distributions of categorical variable `sex` and ordered variable `baselinescore` are presented as percentages and counts. Additionally, missing values are counted per variable. Descriptive statistics, hypothesis tests and effect sizes are automatically chosen according to the class of the target column; see table 3 for details. Because the data is from a randomised study, hypothesis tests comparing baseline variables between the treatment groups are omitted.

Table 1: Demographics of dataset arthritis.

Group	placebo	drug
Observations		
	149	153
age		
Mean (SD)	51 (11)	50 (11)
valid (missing)	149 (0)	153 (0)
sex		
female	29% (43)	26% (40)
male	71% (106)	74% (113)
missing	0% (0)	0% (0)
baselinescore		
1	7.4% (11)	7.8% (12)
2	23% (35)	25% (38)
3	47% (70)	45% (69)
4	19% (28)	18% (28)
5	3.4% (5)	3.9% (6)
missing	0% (0)	0% (0)

Now, present the trial results with `atable`. The target variable is `score`, variable `trt` acts as the grouping variable, and variable `time` splits the dataset before analysis:

```
the_table <- atable(score ~ trt | time, arthritis)
```

Table 2 reports the number of observations per group and time point. The distribution of ordered variables `score` is presented as counts and percentages. Missing values are also counted per variable and group. The p-value and test statistic of the comparison of the two treatment groups are shown. The statistical tests are designed for two or more independent samples, which arise in parallel group trials. The statistical tests are all non-parametric. Parametric alternatives exist that have greater statistical power if their requirements are met by the data, but non-parametric tests are chosen for their broader range of application. The effect sizes with a 95% confidence interval are calculated; see table 3 for details.

LaTeX is not the only supported output format. All possible formats are:

- LaTeX(as shown in this document), further processed with e.g. `latex` of **Hmisc**, `kable` of **knitr** or `xtable` of **xtable**.
- HTML, further processed with e.g. `knitr::kable` of **knitr**.
- Word, can be further processed with e.g. `flextable` of **flextable** Gohel (2018).
- R's console. Human readable format meant for explorative interactive analysis.

The output format is declared by the argument `format_to` of `atable`, or globally via `atable_options`. The **settings** package van der Loo (2015) allows global declaration of various options of `atable`.

Table 2: Hypothesis tests of dataset arthritis.

Group	placebo	drug	p	stat	Effect Size (CI)
Month 1					
Observations					
	149	153			
score					
1	6% (9)	1.3% (2)	0.08	9.9e+03	-0.12 (-0.24; 0.0017)
2	23% (35)	10% (16)			
3	34% (50)	50% (77)			
4	30% (45)	33% (51)			
5	6% (9)	3.3% (5)			
missing	0.67% (1)	1.3% (2)			
Month 3					
Observations					
	149	153			
score					
1	6% (9)	2% (3)	0.0065	9e+03	-0.2 (-0.32; -0.08)
2	21% (32)	18% (27)			
3	42% (63)	34% (52)			
4	24% (36)	33% (50)			
5	5.4% (8)	10% (16)			
missing	0.67% (1)	3.3% (5)			
Month 5					
Observations					
	149	153			
score					
1	5.4% (8)	1.3% (2)	0.004	8.7e+03	-0.22 (-0.34; -0.1)
2	19% (29)	13% (20)			
3	35% (52)	33% (51)			
4	32% (48)	29% (45)			
5	6.7% (10)	18% (28)			
missing	1.3% (2)	4.6% (7)			

Modifying atable

The current implementation of tests and statistics (see table 3) is not suitable for all possible datasets. For example, the parametric t-test or the robust estimator median may be more adequate for some datasets. Additionally, dates and times are currently not handled by **atable**.

It is intended that some parts of **atable** can be altered by the user. Such modifications are accomplished by replacing the underlying methods or adding new ones while preserving the structures of arguments and results of the old functions. The workflow of **atable** (and the corresponding function in parentheses) is as follows:

1. calculate statistics (`statistics`)

Table 3: R classes, scale of measurement and atable. The table lists the descriptive statistics and hypothesis tests applied by atable to the three R classes factor, ordered and numeric. The table also reports the corresponding scale of measurement. atable treats the classes character and logical as the class factor.

R class	factor	ordered	numeric
scale of measurement	nominal	ordinal	interval
statistic	counts occurrences of every level	as factor	Mean and standard deviation
two-sample test	χ^2 test	Wilcoxon rank sum test	Kolmogorov-Smirnov test
effect size	two levels: odds ratio, else Cramér's ϕ	Cliff's Δ	Cohen's d
multi-sample test	χ^2 test	Kruskal-Wallis test	Kruskal-Wallis test

2. apply hypothesis tests (`two_sample_htest` or `multi_sample_htest`)
3. format statistics results (`format_statistics`)
4. format hypothesis test results (`format_tests`).

These five functions may be altered by the user by replacing existing or adding new methods to already existing S3-generics. Two examples are as follows:

Replace existing methods

The **atable** package offers three possibilities to replace existing methods:

- pass a function to `atable_options`. This affects all following calls of `atable`.
- pass a function to `atable`. This affects only a single call of `atable` and takes precedence over `atable_options`.
- replace a function in **atable**'s namespace. This is the most general possibility, as it is applicable to all R packages, but it also needs more code than the other two and is not as easily reverted.

We now define three new functions to exemplify these three possibilities.

First, define a modification of `two_sample_htest.numeric`, which applies `t.test` and `ks.test` simultaneously. See the documentation of `two_sample_htest`: the function has two arguments called `value` and `group` and returns a named list.

```
new_two_sample_htest_numeric <- function(value, group, ...){

  d <- data.frame(value = value, group = group)

  group_levels <- levels(group)
  x <- subset(d, group %in% group_levels[1], select = "value", drop = TRUE)
  y <- subset(d, group %in% group_levels[2], select = "value", drop = TRUE)

  ks_test_out <- stats::ks.test(x, y)
  t_test_out <- stats::t.test(x, y)

  out <- list(p_ks = ks_test_out$p.value,
              p_t = t_test_out$p.value )

  return(out)
}
```

Secondly define a modification of `statistics.numeric`, that calculates the median, MAD, mean and SD. See the documentation of `statistics`: the function has one argument called `x` and the ellipsis `...`. The function must return a named list.

```
new_statistics_numeric <- function(x, ...){

  statistics_out <- list(Median = median(x, na.rm = TRUE),
                         MAD = mad(x, na.rm = TRUE),
                         Mean = mean(x, na.rm = TRUE),
                         SD = sd(x, na.rm = TRUE))

  class(statistics_out) <- c("statistics_numeric", class(statistics_out))
  # We will need this new class later to specify the format
```

Third, define a modification of `format_statistics`: the median and MAD should be next to each other, separated by a semicolon; the mean and SD should go below them. See the documentation of `format_statistics`: the function has one argument called `x` and the ellipsis `...`. The function must return a data.frame with names `tag` and `value` with class factor and character, respectively. Setting a new format is optional because there exists a default method for `format_statistics` that performs the rounding and arranges the statistics below each other.

```
new_format_statistics_numeric <- function(x, ...){

  Median_MAD <- paste(round(c(x$Median, x$MAD), digits = 1), collapse = "; ")
  Mean_SD <- paste(round(c(x$Mean, x$SD), digits = 1), collapse = "; ")

  out <- data.frame(
    tag = factor(c("Median; MAD", "Mean; SD"), levels = c("Median; MAD", "Mean; SD")),
    # the factor needs levels for the non-alphabetical order
    value = c(Median_MAD, Mean_SD),
    stringsAsFactors = FALSE)
  return(out)
}
```

Now apply the three kinds of modification to `atable`: We start with **atable**'s namespace:

```
utils::assignInNamespace(x = "two_sample_htest.numeric",
                         value = new_two_sample_htest_numeric,
                         ns = "atable")
```

Here is why altering `two_sample_htest.numeric` in **atable**'s namespace works: R's lexical scoping rules state that when `atable` is called, R first searches in the enclosing environment of `atable` to find `two_sample_htest.numeric`. The enclosing environment of `atable` is the environment where it was defined, namely, **atable**'s namespace. For more details about scoping rules and environments, see e.g. Wickham (2014), section 'Environments'.

Then modify via `atable_options`:

```
atable_options('statistics.numeric' = new_statistics_numeric)
```

Then modify via passing `new_format_statistics_numeric` as an argument to `atable`, together with actual analysis. See table 4 for the results.

```
the_table <- atable(age ~ trt, arthritis,
                    format_statistics.statistics_numeric = new_format_statistics_numeric)
```

The modifications in `atable_options` are reverted by calling `atable_options_reset()`, changes in the namespace are reverted by calling `utils::assignInNamespace` with suitable arguments.

Replacing methods allows us to create arbitrary tables, even tables independent of the supplied data. We will create a table of a blank data.frame with arbitrary fill-ins (here X.xx) as placeholders. This is usefull for proposals or report templates:

```
# create empty data.frame with non-empty column names
E <- atable::test_data[FALSE, ]

stats_placeholder <- function(x, ...){

  return(list(Mean = "X.xx",
              SD = "X.xx"))
}

the_table <- atable::atable(E, target_cols = c("Numeric", "Factor"),
                            statistics.numeric = stats_placeholder)
```

Table 4: Modified atable now calculates the median, MAD, t-test and KS-test for numeric variables. The median is greater than the mean in both the drug and placebo group, indicating a skewed distribution of age. Additionally the KS-test is significant at the 5% level, while the t-test is not.

Group	placebo	drug	p_ks	p_t
Observations				
	447	459		
age				
Median; MAD	55; 10.4	53; 10.4	0.043	0.38
Mean; SD	50.7; 11.2	50.1; 11		

See table 5 for the results. This table also shows that atable accepts empty data frames without errors.

Table 5: atable applied to an empty data frame with placeholder statistics for numeric variables. The placeholder-function is applied to the numeric variable, printing X.xx in the table. The empty factor variable is summarized in the same way as non-empty factors: by returning percentages and counts; in this case yielding 0/0 = NaN percent and counts of 0 in every category, as expected. Note, that the empty data frame still needs non-empty column names.

Group	value
Observations	
	0
Numeric	
Mean	X.xx
SD	X.xx
Factor	
G3	NaN% (0)
G2	NaN% (0)
G1	NaN% (0)
G0	NaN% (0)
missing	NaN% (0)

Add new methods

In the current implementation of **atable**, the generics have no method for class `Surv` of **survival** Therneau (2015). We define two new methods: the distribution of survival times is described by its mean survival time and corresponding standard error; the Mantel-Haenszel test compares two survival curves.

```
statistics.Surv <- function(x, ...){

  survfit_object <- survival::survfit(x ~ 1)

  # copy from survival:::print.survfit:
  out <- survival:::survmean(survfit_object, rmean = "common")

  return(list(mean_survival_time = out$matrix["*rmean"],
              SE = out$matrix["*se(rmean)"]))
}

two_sample_htest.Surv <- function(value, group, ...){

  survdiff_result <- survival::survdiff(value~group, rho=0)

  # copy from survival:::print.survdiff:
```

```
  etmp <- survdiff_result$exp
  df <- (sum(1 * (etmp > 0))) - 1
  p <- 1 - stats::pchisq(survdiff_result$chisq, df)

  return(list(p = p,
              stat = survdiff_result$chisq))
}
```

These two functions are defined in the user's workspace, the global environment. It is sufficient to define them there, as R's scoping rules will eventually find them after going through the search path, see Wickham (2014).

Now, we need data with class `Surv` to apply the methods. The dataset `ovarian` of **survival** contains the survival times of a randomised trial comparing two treatments for ovarian cancer. Variable `futime` is the survival time, `fustat` is the censoring status, and variable `rx` is the treatment group.

```
library(survival)
# set classes
ovarian <- within(survival::ovarian, {time_to_event = survival::Surv(futime, fustat)})
```

Then, call `atable` to apply the statistics and hypothesis tests. See tables 6 for the results.

```
atable(ovarian, target_cols = c("time_to_event"), group_col = "rx")
```

Table 6: Hypothesis tests of the dataset ovarian.

Group	1	2	p	stat
Observations				
	13	13		
time_to_event				
mean_survival_time	650	889	0.3	1.1
SE	120	115		

Discussion

A single function call does the job, and in conjunction with report-generating packages such as **knitr**, accelerates the analysis and reporting of clinical trials.

Other R packages exist to accomplish this task:

- **furniture** Barrett et al. (2018)
- **tableone** Yoshida and Bohn. (2018)
- **stargazer** Hlavac (2018): focus is more on reporting regression models; no grouping variables, so no two-sample hypothesis tests included; and descriptive statistics are comparable to **atable**
- **DescTools** Signorell (2018): comparable functions are `Desc` (only describes data.frames, no hypothesis tests) and `PercTable` (contingency tables only).

furniture and **tableone** have high overlap with `atable`, and thus we compare their advantages relative to `atable` in greater detail:

Advantages of `furniture::table1` are:

- interacts well with **margrittr**'s pipe `%>%` Bache and Wickham (2014), as mentioned in the examples of `?table1`. This facilitates reading the code.
- handles objects defined by **dplyr**'s `group_by` to define grouping variables Wickham et al. (2019). `atable` has no methods defined for these objects.
- uses non-standard evaluation, which allows the user to create and modify variables from within the function itself, e.g.:

  `table1(df, x2 = ifelse(x > 0, 1, 0))`.

This is not possible with `atable`.

Advantages of `tableone::CreateTableOne` are:

- allows arbitrary column names and prints these names in the resulting table unaltered. This is useful for generating human-readable reports. Blanks and parentheses are allowed for reporting e.g. 'Sex (Male) x%'. Also, non-ASCII characters are allowed. This facilitates reporting in languages that have little or no overlap with ASCII. `atable` demands syntactically valid names defined by `make.names`.
- counting missing values is easily switched on and off by an argument of `tableone::CreateTableOne`. In `atable` a redefinition of a function is needed.
- allows pairwise comparisons tests when data is grouped into more than two classes. `atable` allows only multivariate tests.

Advantages of `atable` are:

- options may be changed locally via arguments of `atable` and globally via `atable_options`,
- easy expansion via S3 methods,
- formula syntax,
- distinction between `split_cols` and `group_col`,
- accepts empty data.frames. This is useful when looping over a list of possibly empty data frames in subgroup analysis, see table 5,
- allows to create tables with a blank data.frame with arbitrary fill-ins (e.g. X.xx) as placeholders for proposals or report templates, also see table 5.

Changing options is exemplified in section 2.4: passing options to `atable` allows the user to modify a single `atable`-call; changing `atable_options` will affect all subsequent calls and thus spares the user passing these options to every single call.

Descriptive statistics, hypothesis tests and effect sizes are automatically chosen according to the class of the target column. R's S3-object system allows a straightforward implementation and extension of this feature, see section 2.4.

`atable` supports the following concise and self-explanatory formula syntax:

```
atable(target_cols ~ group_col | split_cols, ...)
```

R users are used to working with formulas, such as via the `lm` function for linear models. When fitting a linear model to randomised clinical trial data, one can use

```
lm(target_cols ~ group_col, ...)
```

to estimate the influence of the interventions `group_col` on the endpoint `target_cols`. `atable` mimics this syntax:

```
atable(target_cols ~ group_col, ...)
```

performs a hypothesis test, whether there is an influence of the interventions `group_col` on the endpoint `target_cols`.
Also, statisticians know the notion of conditional probability:

```
P(target_cols | split_cols).
```

This denotes the distribution of `target_cols` given `split_cols`. `atable` borrows the pipe `|` from conditional probability:

```
atable(target_cols ~ group_col | split_cols)
```

shows the distribution of the endpoint `target_cols` within the interventions `group_col` given the strata defined by `split_cols`.

`atable` distinguishes between `split_cols` and `group_col`: `group_col` denotes the randomised intervention of the trial. We want to test whether it has an influence on the `target_cols`; `split_cols` are variables that may have an influence on `target_cols`, but we are not interested in that influence in the first place. Such variables, for example, sex, age group, and time point of measurement, arise often in clinical trials. See table 2: the variable `time` is such a supplementary stratification variable: it has an effect on the arthritis score, but that is not the effect of interest; we are interested in the effect of the intervention on the arthritis score.

The package can be used in other research contexts as a preliminary unspecific analysis. Displaying the distributions of variables is a task that arises in every research discipline that collects quantitative data.

I thank the anonymous reviewer for his/her helpful and constructive comments.

Bibliography

S. M. Bache and H. Wickham. *Magrittr: A Forward-Pipe Operator for R*, 2014. URL https://CRAN.R-project.org/package=magrittr. R package version 1.5. [p7]

T. S. Barrett, E. Brignone, and D. J. Laxman. *Furniture: Tables for Quantitative Scientists*, 2018. URL https://CRAN.R-project.org/package=furniture. R package version 1.7.9. [p7]

D. B. Dahl, D. Scott, C. Roosen, A. Magnusson, and J. Swinton. *Xtable: Export Tables to LaTeX or HTML*, 2018. URL https://CRAN.R-project.org/package=xtable. R package version 1.8-3. [p2]

D. Gohel. *Flextable: Functions for Tabular Reporting*, 2018. URL https://CRAN.R-project.org/package=flextable. R package version 0.4.4. [p3]

F. E. Harrell Jr, with contributions from Charles Dupont, and many others. *Hmisc: Harrell Miscellaneous*, 2018. URL https://CRAN.R-project.org/package=Hmisc. R package version 4.1-1. [p2]

M. Hlavac. *Stargazer: Well-Formatted Regression and Summary Statistics Tables*. Central European Labour Studies Institute (CELSI), Bratislava, Slovakia, 2018. URL https://CRAN.R-project.org/package=stargazer. R package version 5.2.2. [p7]

ICH E9. ICH Harmonised Tripartite Guideline. Statistical Principles for Clinical Trials. International Conference on Harmonisation E9 Expert Working Group. *Statistics in medicine*, 18:1905–1942, 1999. ISSN 0277-6715. [p1]

F. Mittelbach, M. Goossens, J. Braams, D. Carlisle, and C. Rowley. *The LaTeX Companion (Tools and Techniques for Computer Typesetting)*. Addison-Wesley Professional, 2004. ISBN 0-201-36299-6. URL https://www.amazon.com/LaTeX-Companion-Techniques-Computer-Typesetting/dp/0201362996?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimbori05-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=0201362996. [p2]

D. Moher, S. Hopewell, K. F. Schulz, V. Montori, P. C. Gøtzsche, P. J. Devereaux, D. Elbourne, M. Egger, and D. G. Altman. Consort 2010 explanation and elaboration: Updated guidelines for reporting parallel group randomised trials. *BMJ*, 340, 2010. ISSN 0959-8138. URL https://doi.org/10.1136/bmj.c869. [p1]

A. Signorell. *DescTools: Tools for Descriptive Statistics*, 2018. URL https://cran.r-project.org/package=DescTools. R package version 0.99.24. [p7]

T. M. Therneau. *A Package for Survival Analysis in S*, 2015. URL https://CRAN.R-project.org/package=survival. version 2.38. [p6]

A. Touloumis. R package multgee: A generalized estimating equations solver for multinomial responses. *Journal of Statistical Software*, 64(8):1–14, 2015. URL http://www.jstatsoft.org/v64/i08/. [p1]

M. van der Loo. *Settings: Software Option Settings Manager for R*, 2015. URL https://CRAN.R-project.org/package=settings. R package version 0.2.4. [p3]

H. Wickham. *Advanced R*. Taylor & Francis Inc, 2014. ISBN 1466586966. URL https://dx.doi.org/10.1201/b17487. [p5, 7]

H. Wickham, R. François, L. Henry, and K. Müller. *Dplyr: A Grammar of Data Manipulation*, 2019. URL https://CRAN.R-project.org/package=dplyr. R package version 0.8.0.1. [p7]

Y. Xie. *Knitr: A General-Purpose Package for Dynamic Report Generation in R*, 2018. URL https://yihui.name/knitr/. R package version 1.20. [p2]

K. Yoshida and J. Bohn. *Tableone: Create 'Table 1' to Describe Baseline Characteristics*, 2018. URL https://CRAN.R-project.org/package=tableone. R package version 0.9.3. [p7]

Armin Ströbel
German Center for Neurodegenerative Diseases (DZNE)
Witten
Armin-Michael.Stroebel@dzne.de