Why zztab2fig? A Practical Comparison with pander

Ronald G. Thomas

Introduction

R provides multiple packages for creating tables from data frames, each with distinct design philosophies. This vignette compares **pander** and **zztab2fig** through practical examples to help you choose the right tool.

pander is designed for quick, inline table generation in R Markdown documents. It converts R objects to Markdown format, which Pandoc then renders to HTML, PDF, or Word. This approach is fast and integrates seamlessly with the R Markdown ecosystem. **zztab2fig** takes a different approach: it generates standalone PDF tables via LaTeX compilation, then crops them for inclusion in documents. This produces publication-quality tables with features that Markdown cannot express, such as footnotes, spanning headers, and decimal alignment.

The choice depends on your needs:

- **Quick reports and HTML output**: pander is simpler and faster
- **Journal submissions and formal publications**: zztab2fig provides the control and formatting features required

```
suppressPackageStartupMessages(library(zztab2fig))
```

Part 1: Basic Tabling

These examples show that both packages handle simple tables well. The differences emerge in formatting control and output quality.

Challenge 1: Simple Data Frame

Task: Display a basic data frame.

pander Approach

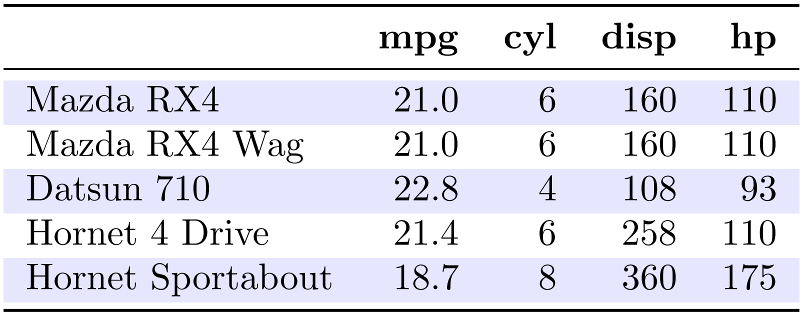
```
df <- mtcars[1:5, 1:4]
pander::pander(df)
```

pander-style output (rendered via Markdown):

	mpg	cyl	disp	hp
Mazda RX4	21.0	6	160	110
Mazda RX4 Wag	21.0	6	160	110
Datsun 710	22.8	4	108	93
Hornet 4 Drive	21.4	6	258	110
Hornet Sportabout	18.7	8	360	175

zztab2fig Approach

```
df <- mtcars[1:5, 1:4]
t2f(df, filename = "basic_table", sub_dir = output_dir)
```



	mpg	cyl	disp	hp
Mazda RX4	21.0	6	160	110
Mazda RX4 Wag	21.0	6	160	110
Datsun 710	22.8	4	108	93
Hornet 4 Drive	21.4	6	258	110
Hornet Sportabout	18.7	8	360	175

Key difference: zztab2fig produces a PDF with professional typesetting (booktabs rules, proper spacing). pander produces Markdown for inline display.

Challenge 2: Column Alignment

Task: Left-align text columns, right-align numeric columns.

pander

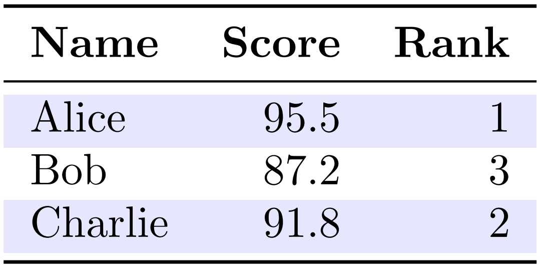
```
df <- data.frame(
  Name = c("Alice", "Bob", "Charlie"),
  Score = c(95.5, 87.2, 91.8),
  Rank = c(1, 3, 2)
)
pander::pander(df, justify = c("left", "right", "right"))
```

pander-style output:

Name	Score	Rank
Alice	95.5	1
Bob	87.2	3
Charlie	91.8	2

zztab2fig

```
df <- data.frame(
  Name = c("Alice", "Bob", "Charlie"),
  Score = c(95.5, 87.2, 91.8),
  Rank = c(1, 3, 2)
)
t2f(df, filename = "aligned_table", sub_dir = output_dir,
    align = c("l", "r", "r"))
```



Name	Score	Rank
Alice	95.5	1
Bob	87.2	3
Charlie	91.8	2

Key difference: Both handle L/C/R alignment. zztab2fig also supports decimal alignment via siunitx (shown later).

Challenge 3: Table with Caption

Task: Add a descriptive caption.

pander

```
df <- iris[1:5, ]
pander::pander(df, caption = "First Five Rows of Iris Dataset")
```

pander-style output:

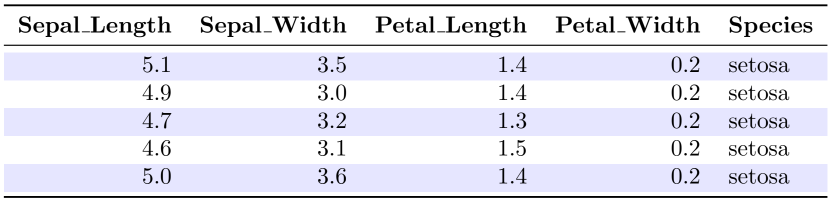
Table 3: First Five Rows of Iris Dataset

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa

zztab2fig

```
df <- iris[1:5, ]
t2f(df,
    filename = "iris_sample",
    sub_dir = output_dir,
    caption = "First Five Rows of Iris Dataset",
    label = "tab:iris")
```

Table 1: First Five Rows of Iris Dataset



Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa

Key difference: zztab2fig adds LaTeX cross-reference labels for use with `\ref{tab:iris}` in documents. pander has no equivalent.

Challenge 4: Regression Model Output

Task: Display coefficients from a linear model.

pander

```r
model <- lm(mpg ~ cyl + hp + wt, data = mtcars)
pander::pander(model)
```

pander-style output:

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	38.7518	1.7869	21.6870	0.0000
cyl	-0.9416	0.5509	-1.7092	0.0985
hp	-0.0180	0.0119	-1.5188	0.1400
wt	-3.1670	0.7406	-4.2764	0.0002

zztab2fig

```r
model <- lm(mpg ~ cyl + hp + wt, data = mtcars)
t2f(model,
    filename = "regression",
    sub_dir = output_dir,
    include = c("estimate", "std.error", "p.value"),
    caption = "Linear Regression Results")
```

Table 1: Linear Regression Results

Term	**Estimate**	**Std__Error**	**p_value**
(Intercept)	38.752	1.787	<0.001
cyl	-0.942	0.551	0.098
hp	-0.018	0.012	0.140
wt	-3.167	0.741	<0.001

Key difference: Both handle `lm` objects. zztab2fig provides S3 methods for selecting which statistics to display (`include` parameter) and supports additional object types (glm, anova, htest).

Part 2: Advanced Features

These challenges demonstrate capabilities unique to zztab2fig. pander cannot replicate these without manual LaTeX coding.

Challenge 5: Table Footnotes

Task: Add footnotes explaining abbreviations and significance levels.

pander

```r
df <- data.frame(
  Variable = c("BMI*", "SBP", "DBP"),
  Mean = c(27.5, 142.3, 88.2),
  SD = c(4.2, 18.5, 11.3)
)
pander::pander(df, caption = "Clinical Measurements")
# Footnotes must be added as separate text below the table
```

pander-style output (no footnote support):

Table 5: Clinical Measurements

Variable	Mean	SD
BMI*	27.5	4.2
SBP	142.3	18.5
DBP	88.2	11.3

Note: With pander, footnotes must be added as separate text below the table.

zztab2fig

```r
df <- data.frame(
  Variable = c("BMI", "SBP", "DBP"),
  Mean = c(27.5, 142.3, 88.2),
  SD = c(4.2, 18.5, 11.3)
)

df$Variable[1] <- t2f_mark("BMI", 1, "symbol")

fn <- t2f_footnote(
  general = "SBP = Systolic Blood Pressure; DBP = Diastolic Blood Pressure.",
  symbol = "Body Mass Index (kg/m2).",
  threeparttable = TRUE
)

t2f(df,
    filename = "clinical_footnotes",
    sub_dir = output_dir,
    caption = "Clinical Measurements",
    footnote = fn)
```

Table 1: Clinical Measurements

Variable	**Mean**	**SD**
BMI[*]	27.5	4.2
SBP	142.3	18.5
DBP	88.2	11.3

Note:
SBP = Systolic Blood Pressure; DBP = Diastolic Blood Pressure.
[*] Body Mass Index (kg/m2).

Advantage: zztab2fig produces proper LaTeX footnotes using threeparttable, with support for general notes, numbered notes, and symbol notes.

Challenge 6: Spanning Column Headers

Task: Group related columns under a common header (e.g., "Treatment" and "Control" groups).

pander

```r
# pander cannot create spanning headers
# Best approximation: rename columns to indicate grouping
names(df) <- c("Outcome", "Treatment Mean", "Treatment SD",
               "Control Mean", "Control SD")
pander::pander(df)
```

pander-style output (workaround with renamed columns):

Outcome	Treatment Mean	Treatment SD	Control Mean	Control SD
Score A	45.2	8.3	42.1	7.9
Score B	38.7	6.2	37.9	5.8

Limitation: pander cannot create multi-column headers. Column names must encode the grouping, resulting in longer, less readable headers.

zztab2fig

```r
df <- data.frame(
  Outcome = c("Score A", "Score B"),
  T_Mean = c(45.2, 38.7),
  T_SD = c(8.3, 6.2),
  C_Mean = c(42.1, 37.9),
  C_SD = c(7.9, 5.8)
)
names(df) <- c("Outcome", "Mean", "SD", "Mean", "SD")

hdr <- t2f_header_above(
  " " = 1,
  "Treatment" = 2,
  "Control" = 2
)

t2f(df,
    filename = "spanning_header",
    sub_dir = output_dir,
    caption = "Outcomes by Group",
    header_above = hdr)
```

Table 1: Outcomes by Group

	Treatment		**Control**	
Outcome	**Mean**	**SD**	**Mean**	**SD**
Score A	45.2	8.3	42.1	7.9
Score B	38.7	6.2	37.9	5.8

Advantage: zztab2fig creates proper LaTeX multicolumn headers with cmidrule separators, essential for clinical trial and research tables.

Challenge 7: Multi-Row Cells (Hierarchical Data)

Task: Merge repeated values vertically to show data hierarchy.

pander

```
df <- data.frame(
  Category = c("Treatment", "Treatment", "Control", "Control"),
  Subgroup = c("Male", "Female", "Male", "Female"),
  N = c(45, 52, 48, 49),
  Response = c("72%", "68%", "45%", "42%")
)
# pander displays repeated values without merging
pander::pander(df)
```

pander-style output (no cell merging):

Category	Subgroup	N	Response
Treatment	Male	45	72%
Treatment	Female	52	68%
Control	Male	48	45%

Category	Subgroup	N	Response
Control	Female	49	42%

Limitation: pander cannot merge cells vertically. Repeated values clutter the table and obscure the hierarchical structure.

zztab2fig

```
df <- data.frame(
  Category = c("Treatment", "Treatment", "Control", "Control"),
  Subgroup = c("Male", "Female", "Male", "Female"),
  N = c(45, 52, 48, 49),
  Response = c("72%", "68%", "45%", "42%")
)

t2f(df,
    filename = "multirow",
    sub_dir = output_dir,
    caption = "Response by Group and Sex",
    collapse_rows = t2f_collapse_rows(
      columns = 1,
      valign = "middle",
      latex_hline = "major"
    ))
```

Table 1: Response by Group and Sex

Category	**Subgroup**	**N**	**Response**
Treatment	Male	45	72%
Treatment	Female	52	68%
Control	Male	48	45%
Control	Female	49	42%

Advantage: zztab2fig automatically collapses repeated values using LaTeX multirow, with configurable vertical alignment and horizontal rules.

Challenge 8: Decimal Point Alignment

Task: Align numbers on the decimal point for easy visual comparison.

pander

```
df <- data.frame(
  Item = c("A", "B", "C"),
  Value1 = c(1.5, 123.45, 12.345),
  Value2 = c(0.001, 10.1, 1000.01)
)
```

```r
# pander uses right-alignment (no decimal alignment)
pander::pander(df, justify = c("left", "right", "right"))
```

pander-style output (right-aligned, not decimal-aligned):

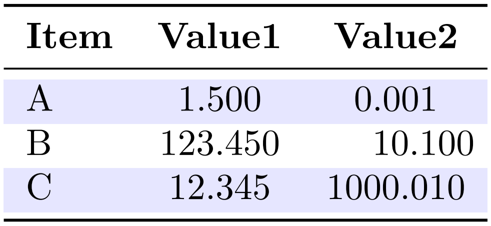
Item	Value1	Value2
A	1.500	0.001
B	123.450	10.100
C	12.345	1000.010

Limitation: pander only supports left/center/right alignment, not decimal alignment. Values with different decimal places are harder to compare visually.

zztab2fig

```r
df <- data.frame(
  Item = c("A", "B", "C"),
  Value1 = c(1.5, 123.45, 12.345),
  Value2 = c(0.001, 10.1, 1000.01)
)

t2f(df,
    filename = "decimal_aligned",
    sub_dir = output_dir,
    align = list(
      "l",
      t2f_siunitx(table_format = "3.3"),
      t2f_siunitx(table_format = "4.2")
    ))
```



Item	Value1	Value2
A	1.500	0.001
B	123.450	10.100
C	12.345	1000.010

Advantage: zztab2fig uses siunitx for true decimal alignment, making numeric comparisons significantly easier to read.

Challenge 9: Journal-Specific Themes

Task: Format a table for NEJM (New England Journal of Medicine) submission.

pander

```r
# pander has no journal-specific themes
# Manual formatting required for each style guide
pander::panderOptions("table.style", "rmarkdown")
pander::pander(df, caption = "Baseline Characteristics")
```

pander-style output (default styling, no NEJM formatting):

Table 9: Baseline Characteristics

Characteristic	Treatment	Placebo
Age, years	65.2 (8.4)	64.8 (8.1)
Male sex, n (%)	142 (58%)	138 (56%)
BMI, kg/m2	27.3 (4.1)	27.1 (3.9)

Limitation: pander provides no built-in journal themes. Meeting specific journal requirements requires manual formatting.

zztab2fig

```r
df <- data.frame(
  Characteristic = c("Age, years", "Male sex, n (%)", "BMI, kg/m2"),
  Treatment = c("65.2 (8.4)", "142 (58%)", "27.3 (4.1)"),
  Placebo = c("64.8 (8.1)", "138 (56%)", "27.1 (3.9)")
)

t2f(df,
    filename = "nejm_table",
    sub_dir = output_dir,
    caption = "Baseline Characteristics",
    theme = "nejm")
```

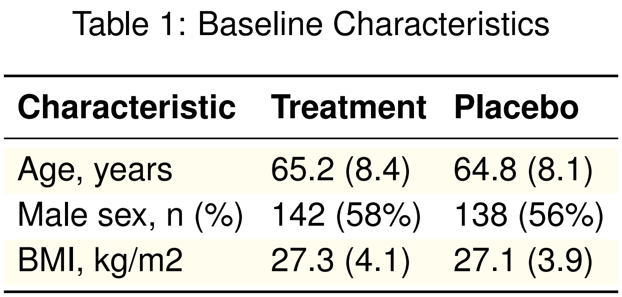


Table 1: Baseline Characteristics

Characteristic	Treatment	Placebo
Age, years	65.2 (8.4)	64.8 (8.1)
Male sex, n (%)	142 (58%)	138 (56%)
BMI, kg/m2	27.3 (4.1)	27.1 (3.9)

Advantage: zztab2fig includes built-in themes for NEJM, APA, Nature, and a minimal style. Each theme sets appropriate fonts, spacing, and rules.

Challenge 10: Model Comparison Table

Task: Display multiple regression models side-by-side with significance stars.

pander

```r
m1 <- lm(mpg ~ cyl, data = mtcars)
m2 <- lm(mpg ~ cyl + hp, data = mtcars)
m3 <- lm(mpg ~ cyl + hp + wt, data = mtcars)

# pander displays one model at a time
pander::pander(m1)
```

```
pander::pander(m2)
pander::pander(m3)
# Side-by-side comparison requires manual table construction
```

pander-style output (separate tables, no side-by-side comparison):

Model 1:

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	37.8846	2.0738	18.2678	0
cyl	-2.8758	0.3224	-8.9197	0

Model 2:

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	36.9083	2.1908	16.8470	0.0000
cyl	-2.2647	0.5759	-3.9325	0.0005
hp	-0.0191	0.0150	-1.2747	0.2125

Model 3:

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	38.7518	1.7869	21.6870	0.0000
cyl	-0.9416	0.5509	-1.7092	0.0985
hp	-0.0180	0.0119	-1.5188	0.1400
wt	-3.1670	0.7406	-4.2764	0.0002

Limitation: pander cannot create comparative model tables automatically. Each model must be displayed separately, making comparison difficult.

zztab2fig

```
m1 <- lm(mpg ~ cyl, data = mtcars)
m2 <- lm(mpg ~ cyl + hp, data = mtcars)
m3 <- lm(mpg ~ cyl + hp + wt, data = mtcars)

t2f_regression(
  Model1 = m1,
  Model2 = m2,
  Model3 = m3,
  stars = TRUE,
  filename = "model_comparison",
  sub_dir = output_dir
)
```

Term	Model1	Model2	Model3
(Intercept)	37.885* (2.074)	36.908* (2.191)	38.752* (1.787)
cyl	-2.876* (0.322)	-2.265* (0.576)	-0.942 (0.551)
hp		-0.019 (0.015)	-0.018 (0.012)
wt			-3.167* (0.741)
N	32	32	32
R-squared	0.726	0.741	0.843
Adj. R-squared	0.717	0.723	0.826

Advantage: `t2f_regression()` automatically aligns terms across models, adds significance stars, and includes model statistics (N, R-squared).

Challenge 11: Complex Combined Features

Task: Create a publication-quality table with footnotes, spanning headers, multi-row cells, and NEJM styling.

pander

```r
# This combination cannot be achieved in pander
# Requires manual LaTeX coding
```

zztab2fig

```r
df <- data.frame(
  Endpoint = c("Primary", "Primary", "Secondary", "Secondary"),
  Timepoint = c("Week 26", "Week 52", "Week 26", "Week 52"),
  N = c(245, 232, 245, 232),
  Difference = c(-0.42, -0.58, -0.28, -0.35),
  P = c("0.008", "0.002", "0.045", "0.018")
)

df$Difference <- sapply(seq_len(nrow(df)), function(i) {
  p <- as.numeric(df$P[i])
  if (p < 0.01) t2f_mark(as.character(df$Difference[i]), 2, "symbol")
  else if (p < 0.05) t2f_mark(as.character(df$Difference[i]), 1, "symbol")
  else as.character(df$Difference[i])
})

hdr <- t2f_header_above(" " = 2, "Results" = 3)

fn <- t2f_footnote(
  general = "Negative values favor treatment.",
  symbol = c("p < 0.05", "p < 0.01")
)

t2f(df,
    filename = "complex_table",
    sub_dir = output_dir,
    caption = "Efficacy Results by Endpoint and Timepoint",
    caption_short = "Efficacy Results",
    header_above = hdr,
    collapse_rows = t2f_collapse_rows(1, valign = "top"),
    footnote = fn,
    theme = "nejm")
```

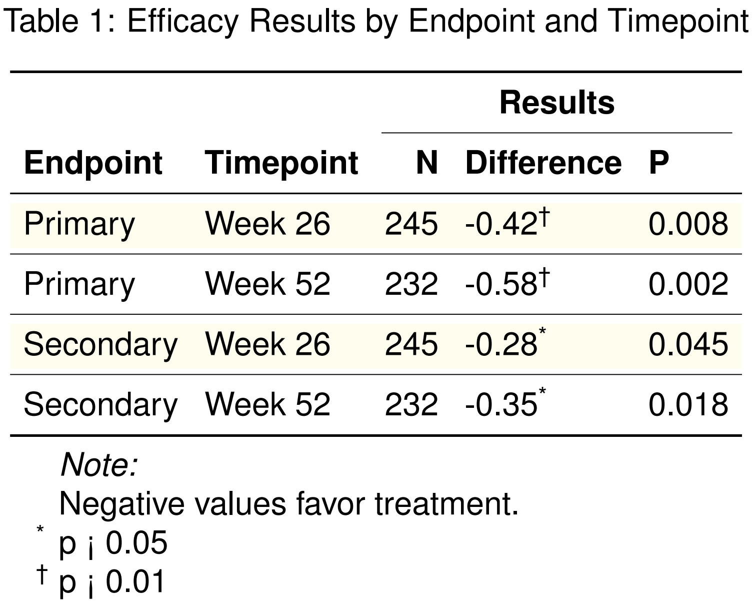


Table 1: Efficacy Results by Endpoint and Timepoint

		Results		
Endpoint	**Timepoint**	**N**	**Difference**	**P**
Primary	Week 26	245	-0.42†	0.008
Primary	Week 52	232	-0.58†	0.002
Secondary	Week 26	245	-0.28*	0.045
Secondary	Week 52	232	-0.35*	0.018

Note:
Negative values favor treatment.
* p ¡ 0.05
† p ¡ 0.01

Part 3: R Markdown Workflow

Challenge 12: Inline Tables Without Floats

Task: Insert a coefficient table exactly where the code chunk appears, with proper caption and cross-reference but without LaTeX float positioning.

pander

```r
model <- lm(mpg ~ cyl + hp, data = mtcars)
pander::pander(model)
```

pander tables appear inline, but captions use Markdown syntax that may not produce proper table numbering or cross-references in PDF output.

zztab2fig

The `t2f_inline()` function is designed for exactly this use case:

```r
model <- lm(mpg ~ cyl + hp, data = mtcars)
t2f_inline(model,
           width = "3in",
           align = "left",
           caption = "Linear model coefficients",
           label = "tab:model",
           caption_position = "above")
```

Key features:

- Table appears exactly at code chunk location (no float)
- Uses `\captionof{table}{...}` for proper "Table 1:" numbering
- Cross-references work with `\ref{tab:model}`
- `caption_position`: "above" (default, standard for tables) or "below"
- Auto-detects PDF vs HTML output format

Convenience function for models:

```
t2f_coef(model, caption = "Model results", label = "tab:coef")
```

This is equivalent to `t2f_inline()` with sensible defaults for coefficient tables (width = "3in", align = "left").

Note: When using captions with `t2f_inline()`, add to your R Markdown YAML:

```
header-includes:
  - \usepackage{caption}
```

Summary

Feature Comparison

Feature	pander	zztab2fig
Basic data frames	Yes	Yes
Column alignment (L/C/R)	Yes	Yes
Decimal alignment	No	Yes (siunitx)
Captions	Yes	Yes + short captions
Cross-reference labels	No	Yes
Non-float captions	No	Yes (captionof)
Caption position control	No	Yes (above/below)
Table footnotes	No	Yes (4 notation types)
Spanning headers	No	Yes
Multi-row cells	No	Yes
Multi-page tables	No	Yes (longtable)
Journal themes	No	Yes (NEJM, APA, Nature)
Model comparison	No	Yes
Statistical object S3 methods	Partial	Yes (20+ types via broom)
Output formats	Markdown	PDF, PNG, SVG, TEX
Inline R Markdown workflow	Yes	Yes (t2f_inline)

When to Use Each Package

Choose pander when:

- Creating R Markdown documents for HTML or Word output
- Need quick, inline display of many R object types
- Document will be processed through Pandoc
- Simple tables with basic formatting are sufficient
- Rapid iteration is more important than typographic quality

Choose zztab2fig when:

- Creating tables for journal submission
- Need footnotes, spanning headers, or multi-row cells
- Working with LaTeX documents directly
- Require decimal alignment for numeric comparison

- Want consistent journal-specific styling (NEJM, APA, Nature)
- Need cropped PDF tables for inclusion in external documents
- Building model comparison tables
- Want proper table numbering without float positioning

v0.2.0 Features Used in This Vignette

This vignette demonstrates several features introduced in zztab2fig v0.2.0: - **t2f_inline()**: Inline tables for R Markdown without floats - **t2f_coef()**: Quick coefficient tables with sensible defaults - **caption_position**: Control caption placement (above/below) - **S3 methods**: Support for 20+ object types via broom integration - **Theme system**: Built-in NEJM theme (`theme = "nejm"`) - **t2f_regression()**: Side-by-side model comparison - **t2f_footnote()**: Structured footnotes with multiple notation types - **t2f_header_above()**: Spanning column headers - **t2f_collapse_rows()**: Multi-row cell merging - **t2f_siunitx()**: Decimal alignment - **t2f_mark()**: Footnote markers in cells - **caption_short**: Short captions for List of Tables

For complete documentation of these features, see the "Advanced Features" and "Object Types and Themes" vignettes.