competitive_analysis

November 04, 2025 at 10:54 AM

Competitive Analysis: zztable1_nextgen vs gt Package Ecosystem

Executive Summary

After conducting a deep dive into both packages, **zztable1_nextgen is positioned as a specialized, medical journal-focused Table 1 generator**, while **gt is a general-purpose table grammar system** similar to ggplot2 for graphics. They serve different but overlapping markets.

Current Status: - **zztable1_nextgen**: ~5,500 LOC, 33 exported functions, specialized for clinical trials - **gt ecosystem**: 2,100+ GitHub stars, 56 contributors, 8,982 commits, general-purpose

Architecture Comparison

Table Construction Philosophy

zztable1_nextgen: - **Formula-based**: `table1(arm ~ age + sex, data=trial)` - familiar to R users - **Lazy evaluation blueprint**: Cell metadata stored until rendering - **Sparse storage**: 60-80% memory reduction via environment-based hash tables - **Purpose-built**: Designed specifically for "Table 1" demographic comparisons

gt Package: - **Grammar-based**: Layered API like ggplot2 (`gt() %>% tab_header() %>% fmt_*()`) - **Immediate evaluation**: Tables built progressively through piping - **Full materialization**: Complete table object with all data - **General-purpose**: Any tabular display, not just clinical tables

gtsummary (gt companion): - **Hybrid approach**: `tbl_summary(by=arm)` - simple interface like zztable1_nextgen - **Builds on gt**: Exports to gt object for further customization - **Specialized**: Designed specifically for medical/clinical tables

Feature Matrix Comparison

Feature Category	zztable1_nextgen	gt	gtsummary
Core Capabilities			

Feature Category	zztable1_nextgen	gt	gtsummary
Formula interface	[CHECKMARK] Native	[X] No	[CHECKMARK] Via `by=`
Piping/chaining API	[X] Limited	[CHECKMARK] Extensive	[CHECKMARK] Decorator functions
Automatic variable detection	[CHECKMARK] Factor/numeric	[X] Manual	[CHECKMARK] All types + dichotomous
Statistical tests	[CHECKMARK] 5 tests	[X] No	[CHECKMARK] Auto-selection
Missing data handling	[CHECKMARK] Counts	[X] No	[CHECKMARK] Automatic
Output Formats			
HTML	[CHECKMARK] Basic	[CHECKMARK] Rich	[CHECKMARK] Via gt
LaTeX	[CHECKMARK] PDF-ready	[CHECKMARK] Advanced	[CHECKMARK] Via gt
RTF	[X] No	[CHECKMARK] Yes	[CHECKMARK] Via gt
Word	[X] No	[CHECKMARK] Via gtsave()	[CHECKMARK] Via gt
PNG/image export	[X] No	[CHECKMARK] Via gtsave()	[CHECKMARK] Via gt
Styling System			
Built-in themes	[CHECKMARK] 4 medical journals	[X] No themes	[CHECKMARK] Journal themes
Custom CSS	[CHECKMARK] generate_theme_css()	[CHECKMARK] Extensive	[CHECKMARK] Via gt
Row striping	[CHECKMARK] NEJM theme	[CHECKMARK] Manual	[CHECKMARK] Manual
Cell formatting	[CHECKMARK] Limited	[CHECKMARK] 20+ fmt_*() functions	[CHECKMARK] Extensive
Conditional formatting	[X] No	[CHECKMARK] tab_style() with location	[CHECKMARK] Bold/italics
Table Structure			
Headers/titles	[X] Limited	[CHECKMARK] tab_header()	[CHECKMARK] Modify functions
Footnotes	[CHECKMARK] Variable-specific	[CHECKMARK] Cell-specific	[CHECKMARK] Multiple types
Source notes	[X] No	[CHECKMARK] tab_source_note()	[CHECKMARK] Via modify_*()
Spanning headers	[X] No	[CHECKMARK] tab_spanner()	[X] Limited

Feature Category	zztable1_nextgen	gt	gtsummary
Row groups	[X] No	[CHECKMARK] tab_row_group()	[CHECKMARK] Auto from variables
Summary rows	[X] No	[CHECKMARK] summary_rows()	[CHECKMARK] Add N, p-values
Data Transformation			
Column selection	[CHECKMARK] Formula	[CHECKMARK] dplyr-style	[CHECKMARK] include=
Data manipulation	[X] Pre-process only	[CHECKMARK] Within gt	[X] Pre-process only
Column merging	[X] No	[CHECKMARK] cols_merge()	[CHECKMARK] tbl_merge()
Table stacking	[X] No	[X] Limited	[CHECKMARK] tbl_stack()
Specialized Features			
Medical themes (NEJM, Lancet, JAMA)	[CHECKMARK]	[X]	[CHECKMARK]
Regression tables	[X]	[X]	[CHECKMARK] tbl_regression()
Cross-tabulation	[X]	[X]	[CHECKMARK] tbl_cross()
Survey data	[X]	[X]	[CHECKMARK] tbl_svysummary()
Inline reporting	[X]	[X]	[CHECKMARK] inline_text()
Extensibility			
Custom functions	[CHECKMARK] numeric_summary=	[CHECKMARK] Functions everywhere	[CHECKMARK] Custom stats
Themes API	[CHECKMARK] create_custom_theme()	[CHECKMARK] Full CSS control	[CHECKMARK] Theme + gt
Plugin system	[X]	[CHECKMARK] gtExtras package	[CHECKMARK] Multiple extensions

Critical Feature Gaps

High Priority (Must-Have for Competition)

1. **Output Formats:**
   - ☒ **RTF export** - Critical for Word manuscript submission
   - ☒ **Image export** (PNG/PDF) - Essential for presentations
   - ☒ **Word/DOCX** - Direct Word integration needed

2. **Advanced Formatting:**
   - ☒ **Cell-specific styling** - Cannot conditionally format individual cells
   - ☒ **Spanning headers** - Multi-level column headers not supported
   - ☒ **Source notes** - Cannot add data source attributions
3. **Table Composition:**
   - ☒ **Table merging** - Cannot combine multiple tables side-by-side
   - ☒ **Table stacking** - Cannot stack tables vertically
   - ☒ **Summary rows** - No subtotals or group summaries
4. **Formatting Functions:**
   - ☒ **Currency formatting** - No `fmt_currency()`
   - ☒ **Percentage formatting** - No `fmt_percent()` with flexible options
   - ☒ **Date formatting** - No `fmt_date()` helpers
   - ☒ **Scientific notation** - No `fmt_scientific()`
   - ☒ **Number suffixing** - No `fmt_number()` with K/M suffixes

Medium Priority (Important for Differentiation)

5. **Inline Reporting:**
   - ☒ **Extract values for text** - Cannot easily pull statistics into prose
     - Example: gtsummary can do `inline_text(tbl, variable="age")`
6. **Regression Tables:**
   - ☒ **Logistic regression** - No automated odds ratio tables
   - ☒ **Cox proportional hazards** - No hazard ratio tables
   - ☒ **Linear regression** - No coefficient tables
7. **Interactive Features:**
   - ☒ **Sortable columns** - Static tables only
   - ☒ **Filterable** - No interactivity
   - ☒ **DT integration** - No DataTables.js support
8. **Advanced Customization:**
   - ☒ **Cell background colors** - Cannot color-code cells
   - ☒ **Cell borders** - Limited border control
   - ☒ **Custom cell renderers** - Fixed rendering logic

Lower Priority (Nice-to-Have)

9. **Data Manipulation:**
   - ☒ **Column reordering** - Cannot rearrange columns after creation
   - ☒ **Column hiding** - Cannot selectively hide columns
   - ☒ **Column widths** - No width specification
10. **Accessibility:**
    - ☒ **Screen reader support** - No ARIA labels
    - ☒ **Alt text** - No descriptive text for assistive technology

Competitive Advantages of zztable1_nextgen

What You Do Better

1. **Formula Interface** □□□□□
   - More intuitive for statisticians: `arm ~ age + sex + bmi`
   - Familiar to R users (similar to `lm()`, `anova()`)
   - Less verbose than gt's piping chains
2. **Memory Efficiency** □□□□
   - Sparse storage (60-80% reduction)
   - Lazy evaluation architecture
   - Scales better for large datasets
3. **Medical Journal Themes** □□□□□
   - Authentic NEJM, Lancet, JAMA formatting
   - Based on actual journal papers
   - Row striping, precise spacing, typography
   - **gt/gtsummary don't have these built-in**
4. **Automatic Statistical Tests** □□□□
   - Configurable: t-test, ANOVA, Welch, Kruskal-Wallis
   - Fisher's exact, Chi-square
   - **gt has NO statistical testing**
   - gtsummary has tests but less flexible
5. **Stratified Analysis** □□□□
   - Native support: `strata="center"`
   - Multi-center trial analysis
   - **gt doesn't support this pattern natively**
6. **Missing Data Display** □□□
   - Automatic detection and display
   - Counts per variable
   - **gt requires manual handling**

What Makes You Different (Not Better/Worse, Just Different)

7. **Specialized Focus:**
   - **You:** Purpose-built for Table 1 in clinical trials
   - **Them:** General-purpose table system
8. **Design Philosophy:**
   - **You:** Immediate results, sensible defaults
   - **Them:** Incremental building, maximal flexibility

Recommendations for Competitive Positioning

Short-Term (Essential for Viability)

1. **Add RTF Export** (Critical)

```
export_rtf <- function(blueprint, file) {
  # RTF is essential for Word manuscripts
}
```

2. **Add Image Export** (High Priority)

```
save_table <- function(blueprint, file, format=c("png", "pdf", "html")) {
  # Use webshot2 or similar for PNG
}
```

3. **Enhance Footnote System** (High)
   - Cell-specific footnotes
   - Multiple footnote types (regular, abbreviations, statistical notes)
4. **Add Spanning Headers** (High)

```
table1(...) %>%
  add_spanner("Treatment Groups", columns = c("Control", "Drug A", "Drug B"))
```

5. **Basic Cell Formatting Functions** (High)

```
fmt_percent(blueprint, columns = "response_rate")
fmt_pvalue(blueprint, columns = "p_value", threshold = 0.001)
```

Medium-Term (Competitive Differentiation)

6. **Regression Table Support**

```
table_regression(glm_model, theme = "nejm", exponentiate = TRUE)
```

7. **Table Merging**

```
merge_tables(table1_demographics, table1_outcomes, direction = "horizontal")
```

8. **Inline Reporting**

```
extract_stat(blueprint, variable = "age", group = "Treatment", stat = "mean")
```

9. **Enhanced Styling API**

```
blueprint %>%
  style_cells(rows = p_value < 0.05, bold = TRUE) %>%
  style_cells(rows = age > 65, background = "#ffffcc")
```

10. **Interactive Output Option**

```
table1(..., interactive = TRUE) # Uses DT package
```

Long-Term (Market Expansion)

11. **Visual Elements**
    - Sparklines in cells
    - Bar charts for comparison
    - Color scales for continuous variables
12. **Additional Table Types**

- Adverse events tables (AE tables)
- Time-to-event summaries
- Longitudinal data tables

13. **Reporting Integration**
    - Quarto extension
    - Shiny widgets
    - Officer package integration for PowerPoint
14. **Collaboration Features**
    - Export table specification for reproducibility
    - Import from RedCap/EHR formats
    - API for clinical trial management systems

Strategic Positioning

Don't Try to Beat gt at Everything

gt is a **general-purpose table grammar** (like ggplot2 for graphics). You cannot and should not compete on: - □ Breadth of formatting functions - □ Flexibility for non-clinical tables - □ Visual customization depth

Instead, Dominate Your Niche

Position as **"The Clinical Trials Table 1 Specialist":**

1. **Better defaults for medical research:**
    - "Works out of the box for RCTs"
    - "Authentic journal formatting"
    - "Statistical tests included"
2. **Faster for your use case:**
    - "One line vs 10 lines for Table 1"
    - Formula: `table1(arm ~ age + sex, trial, theme="nejm")`
    - vs gt+gtsummary requiring multiple pipes
3. **Domain expertise:**
    - Medical journal themes
    - Stratified analysis
    - Missing data handling
    - Regulatory compliance considerations

Complementary Positioning

You are NOT a gt competitor. You are a gt COMPLEMENT.

Many users will use both: - **zztable1_nextgen** for quick Table 1 generation - **gt** for customized results tables, adverse events, etc.

Consider: **`as_gt()` function** to export zztable1_nextgen to gt for further customization:

```
blueprint <- table1(arm ~ age + sex, data = trial, theme = "nejm")
gt_table <- as_gt(blueprint) # Export to gt for more customization
```

```
gt_table %>% tab_style(...) %>% gtsave("table1.rtf")
```

This makes you **interoperable** rather than **competitive**.

Priority Implementation Roadmap

Phase 1: Core Gaps (3-6 months)

1. RTF export via gt::gtsave() or similar
2. PNG export via webshot2/chromote
3. Spanning column headers
4. Enhanced footnote system
5. fmt_percent(), fmt_pvalue() formatters

Estimated effort: 120-180 hours

Phase 2: Differentiation (6-12 months)

6. Regression table support (tbl_regression equivalent)
7. as_gt() export function for interoperability
8. Inline statistics extraction
9. Table merging (horizontal/vertical)
10. Conditional cell styling

Estimated effort: 200-300 hours

Phase 3: Expansion (12-18 months)

11. Interactive tables option
12. Adverse event tables
13. Quarto/Shiny integration
14. Additional medical journal themes (BMJ, Annals, JAMA subspecialties)
15. Visual enhancements (sparklines, etc.)

Estimated effort: 300-400 hours

Market Analysis

Target Users

Primary (You Excel): - Clinical trial statisticians - Medical researchers writing manuscripts - Regulatory submission specialists - Epidemiologists

Secondary (Shared with gt): - Academic researchers - Data scientists in healthcare - Medical journal editors

Tertiary (gt Dominates): - Business analysts - Marketing researchers - General R users

Estimated Market Share Opportunity

- **Total medical/clinical R users**: ~50,000-100,000

- **Clinical trial statisticians**: ~5,000-10,000
- **Your potential users**: 10-30% = **1,000-3,000 primary users**
- **gt's broader market**: 500,000+ R users

Conclusion: You can succeed with a smaller but more loyal user base by being THE BEST at clinical Table 1 generation.

Code Architecture Recommendations

Current Strengths to Preserve

1. **Sparse storage with environments** - Keep this, it's brilliant
2. **Blueprint/lazy evaluation** - Unique approach, maintain
3. **Theme system** - Well-designed, extend it
4. **Formula interface** - Core differentiator, protect

Areas Needing Enhancement

1. **Rendering system** - Add pluggable renderers for RTF, DOCX

   ```
   # R/renderers/rtf_renderer.R
   # R/renderers/docx_renderer.R
   # R/renderers/png_renderer.R
   ```

2. **Formatting functions** - Create fmt_* family

   ```
   # R/formatting/fmt_percent.R
   # R/formatting/fmt_pvalue.R
   # R/formatting/fmt_number.R
   ```

3. **Interoperability layer** - Bridge to gt/flextable

   ```
   # R/export/as_gt.R
   # R/export/as_flextable.R
   ```

4. **Table composition** - Merge/stack operations

   ```
   # R/composition/merge_tables.R
   # R/composition/stack_tables.R
   ```

Technical Implementation Priorities

1. RTF Export (Critical - Week 1-2)

Use existing packages:

```
# Option 1: Via gt (easiest)
as_gt.table1_blueprint <- function(blueprint, ...) {
  # Convert blueprint to gt object
  gt_obj <- gt::gt(as.data.frame(blueprint))
  # Apply theme
  # Return gt object for gtsave("file.rtf")
```

```
}

# Option 2: Direct RTF via officer/flextable
export_rtf <- function(blueprint, file) {
  ft <- flextable::flextable(as.data.frame(blueprint))
  # Apply formatting
  flextable::save_as_rtf(ft, path = file)
}
```

2. Image Export (Critical - Week 3-4)

```
save_table <- function(blueprint, file, format = "png",
                       width = 8, height = 6, dpi = 300) {
  # Render to HTML first
  html_file <- tempfile(fileext = ".html")
  html <- render_html(blueprint)
  writeLines(html, html_file)

  # Convert to image
  if (format %in% c("png", "pdf")) {
    webshot2::webshot(html_file, file = file,
                      vwidth = width * dpi,
                      vheight = height * dpi)
  }
}
```

3. Spanning Headers (High Priority - Week 5-6)

```
add_spanner <- function(blueprint, label, columns) {
  # Modify blueprint$metadata$spanners
  blueprint$metadata$spanners <- c(
    blueprint$metadata$spanners,
    list(list(label = label, columns = columns))
  )
  blueprint
}

# Update rendering to handle spanners
render_html.table1_blueprint <- function(blueprint, ...) {
  # Check for spanners in metadata
  # Render <thead> with multiple rows
}
```

4. Formatting Functions (High Priority - Week 7-8)

```r
fmt_percent <- function(blueprint, columns, decimals = 1,
                        scale = 100, pattern = "{x}%") {
  # Modify cells in specified columns
  for (col in columns) {
    # Apply formatting to blueprint cells
  }
  blueprint
}

fmt_pvalue <- function(blueprint, columns,
                       threshold = 0.001,
                       pattern = "{x}") {
  # Format p-values: <0.001, 0.023, etc.
  blueprint
}
```

5. as_gt() Interoperability (Medium Priority - Week 9-10)

```r
as_gt.table1_blueprint <- function(blueprint,
                                   include_footnotes = TRUE,
                                   include_theme = TRUE, ...) {
  # 1. Evaluate all cells
  df <- as.data.frame(blueprint)

  # 2. Create gt object
  gt_obj <- gt::gt(df, rowname_col = names(df)[1])

  # 3. Apply theme styling
  if (include_theme) {
    theme <- blueprint$metadata$theme
    # Map theme to gt tab_style() calls
  }

  # 4. Add footnotes
  if (include_footnotes) {
    # Add via gt::tab_footnote()
  }

  gt_obj
}
```

Final Recommendations

Must-Do (Survival)

1. [CHECKMARK] Add RTF export
2. [CHECKMARK] Add PNG/PDF export
3. [CHECKMARK] Build `as_gt()` interoperability
4. [CHECKMARK] Add spanning headers
5. [CHECKMARK] Enhance footnotes

Should-Do (Competitive)

6. [CHECKMARK] Regression tables
7. [CHECKMARK] Table merging
8. [CHECKMARK] Inline statistics
9. [CHECKMARK] Basic conditional formatting
10. [CHECKMARK] Word export

Could-Do (Nice-to-Have)

11. Interactive tables
12. Sparklines/visual elements
13. Additional journal themes
14. Shiny/Quarto deep integration
15. REDCap integration

Don't-Do (Wrong Direction)

- ☒ General business intelligence tables
- ☒ Financial statement formatting
- ☒ Web dashboard tables
- ☒ Pivot table functionality
- ☒ Trying to match gt's breadth

Your competitive moat is DEPTH in clinical trials, not BREADTH in table types.

Conclusion

zztable1_nextgen has a clear path to success by:

1. **Dominating the clinical trials niche** (1,000-3,000 loyal users)
2. **Interoperating with gt** (not competing)
3. **Focusing on depth** (better Table 1) not breadth (all tables)
4. **Filling critical gaps** (RTF, PNG, spanning headers)
5. **Maintaining unique strengths** (formula interface, medical themes, sparse storage)

The package is 70% feature-complete for its niche. The remaining 30% (output formats, advanced formatting) are achievable in 6-12 months with focused development.

Strategic positioning: "The fastest way to create publication-ready Table 1 for clinical trials, with authentic medical journal formatting and one-line statistical testing."

Tagline: "Table 1, done right. First time."