

Seven Required Elements for a zzc Manuscript report.Rmd

Ronald G. Thomas

2026-05-25



Figure 1: An open manuscript on an oak desk, a metal ruler along its numbered margin and a red pen resting across the corner: margins ruled, pages numbered, every element in place.

Seven small additions to a YAML header that together make a manuscript PDF reproducible, annotatable, and unambiguously traceable to its source.

Introduction

The zzzcollab compendium framework provides a reproducible environment for building manuscript PDFs: a Docker container, a locked renv snapshot, and a `stage-render.R` hook that copies every rendered PDF to a shared staging directory with a dated filename. The framework handles a great deal automatically, but only if the `report.Rmd` inside the compendium includes seven specific YAML and structural elements.

After auditing more than fifty `report.Rmd` files across active research projects, the most common finding was not errors in the statistical content. It was missing format elements: a wide right margin

absent, line numbers forgotten, the bibliography landing before the render timestamp rather than after it. Each omission is small and each is easy to miss during initial setup. Together they accumulate into a document that cannot easily be annotated by a reviewer, does not have line numbers for journal submission, and has provenance that is difficult to verify after the fact.

This post describes the seven elements, explains why each is needed, and provides a generic checklist at `docs/zzc-rmd-format-checklist.md` that can be pasted into any new or existing compendium `report.Rmd` as a starting point.

Motivations

- Journal peer review workflows depend on line numbers. Reviewers and co-authors reference lines by number; a manuscript without them forces imprecise ‘paragraph 3 of Section 2’ citations in review correspondence.
- A wide right margin (5 cm on the right versus the standard ~2.5 cm) provides space for handwritten or PDF annotation without the text being obscured. This is particularly useful during advisor review and in pre-submission rounds.
- Without the `knit: hook to stage-render.R`, the dated staging copy is never created. Two collaborators working from separate `report.pdf` files with no date stamp cannot determine which version is current.
- The `{#refs}` div and footer chunk must appear in the correct order. If the footer chunk precedes the references section in the source, the render timestamp lands in the middle of the document, not on the last page.
- Running the same audit across all compendiums in a research program reveals that format debt accumulates project by project. A shared checklist applied at project initialization prevents the debt from forming.

Objectives

1. Define each of the seven required elements and the rationale for including it.
2. Show the exact YAML and chunk code needed for each element in a form that can be copied verbatim into a `report.Rmd`.
3. Provide a generic, project-agnostic checklist that covers all seven elements and can serve as a template for new compendiums.
4. Identify the common failure modes (wrong element ordering, missing path pattern, stale installed package) so they can be diagnosed without a full re-read of the document.



Figure 2: Seven small objects in a precise row: folded card, ruler, stamp, pen, sealed envelope, paperclip, and book, each a discrete element none redundant, none optional.

What Is the zzc Report Format?

A *zzc* report format is a set of YAML fields and R code chunks that every `report.Rmd` in a *zzcollab* manuscript compendium must include to produce a consistently formatted, correctly staged, and fully traceable PDF. Think of it as the stylesheet for the compendium: just as a journal's LaTeX class file ensures consistent typography across submissions, the *zzc* report format ensures consistent presentation, provenance, and staging behavior across all manuscripts in a research program.

The format is not enforced by the framework automatically. It is a convention, documented here and checked by the audit procedure. The generic checklist in `docs/zzc-rmd-format-checklist.md` provides a machine-scannable reference.

Prerequisites

This post assumes:

- **zzcollab** is installed and a compendium has been initialized with `zzc analysis` or equivalent.
- The compendium contains a `tools/stage-render.R` file (present in all compendiums created with *zzcollab* 0.4+).
- The `report.Rmd` uses `bookdown::pdf_document2` or `pdf_document` as its output format.
- **LaTeX packages** `lineno`, `setspace`, and `booktabs` are available in the container's TeX distribution (they are included in TeX Live 2023+, which ships in the *zzcollab* Docker image).
- **R packages:** `rmarkdown`, `bookdown`, and `knitr` are in the compendium's `renv.lock`.

Readers unfamiliar with zzcollab should start with the introductory post in the See Also section before applying these elements.

The Seven Required Elements

Element 1: Wide Right Margin

Add a `geometry:` field as a top-level YAML key:

```
geometry: "left=3cm,right=5cm,top=2cm,bottom=2cm"
```

The right margin is set to 5 cm rather than the LaTeX default of roughly 2.5 cm. This provides space for handwritten annotation and for PDF commenting tools (Preview, Acrobat, Zotero PDF). The asymmetric margin also distinguishes a draft manuscript from a final typeset article, which is useful when multiple PDF versions circulate during review.

Placement: Any top-level YAML field position is correct, but placing it near `header-includes:` keeps layout-related settings together.

Element 2 and 3: Line Numbers

Add two entries to `header-includes::`

```
header-includes:
- \usepackage{lineno}
- \linenumbers
```

`\usepackage{lineno}` loads the LaTeX `lineno` package. `\linenumbers` activates continuous line numbering from the first line of the document body. Line numbers are required by most biomedical journals for peer review submission and are expected by reviewers in pre-submission rounds. They also make it unambiguous when a collaborator says ‘the sentence on line 47.’

Note: `lineno` can conflict with some `display-math` environments. If a LaTeX error mentioning `lineno` appears during render, wrap the offending equation in `\begin{linenomath*}` ... `\end{linenomath*}`.

Element 4: Line Spacing

Add two more entries to `header-includes::`

```
header-includes:
- \usepackage{setspace}
- \setstretch{1.1}
```

`\setstretch{1.1}` applies a line-spacing multiplier of 1.1, slightly wider than single-spacing and narrower than 1.5. This is a common journal-review convention: tight enough to reduce page count relative to double-spacing, loose enough to accommodate handwritten comments between lines. The value 1.1 is the program standard; adjust per journal requirements.

Element 5: Author Affiliation and ORCID

The `author:` field must include a `\thanks{}` command with the institutional affiliation and ORCID identifier:

```
author: "Author Name, Ph.D.\thanks{Institution name;  
ORCID: 0000-0000-0000-0000}"
```

The `\thanks{}` command produces a footnote on the title page. Most journals require the affiliation and ORCID to appear on the first page of a submitted manuscript. Including them in the `author:` field rather than in the body text ensures they appear in the standard title-block position regardless of the output format.

Note: Use double quotes around the entire author string (single quotes cannot contain `\thanks{}`). The backslash before `thanks` must be doubled (`\\thanks`) because YAML processes one level of escaping before LaTeX sees the string.

Exception: Multi-author or team-attributed manuscripts (such as simulation studies with collective authorship) may use a plain author string such as `author: 'project team'` when no single `\thanks{}` affiliation applies uniformly.

Element 6: Knit Hook to stage-render.R

The `knit:` field replaces the default knit function with a call to `stage-render.R`:

```
knit: (function(input, encoding)  
  source(normalizePath(file.path(  
    dirname(input), '..', '..', '..', 'tools',  
    'stage-render.R')))$value(input, encoding))
```

When the Knit button is pressed in RStudio (or when the document is rendered via the custom function), `stage-render.R`:

1. Calls `rmarkdown::render()` to produce `report.pdf` alongside the source `report.Rmd`.
2. Creates `analysis/report/share/` if it does not exist.
3. Copies the PDF to `analysis/report/share/<slug>-<YYYY-MM-DD>-dN.pdf`, where `N` is a per-day draft counter starting at 1.
4. Appends a row to `analysis/report/share/MANIFEST.md` recording the filename, source path, and render timestamp.

The path depth in `file.path(dirname(input), '..', '..', '..', 'tools')` assumes the `report.Rmd` is three directory levels below the project root (`analysis/report/NN-slug/report.Rmd`). For reports at two levels deep (`analysis/report/report.Rmd`), use `'..', '..', 'tools'` instead.



Figure 3: A stack of dated pages in an open manila folder beside a rubber date stamp: provenance recorded without manual effort.

Element 7a: Bibliography Placement with

Place an explicit div immediately after the `# References` heading:

```
# References

::: {#refs}
:::
```

Without the `{#refs}` div, pandoc places the bibliography at the location where it first has room, which in a bookdown document may not be immediately after the `# References` heading. The div pins the bibliography to that exact location in the output, ensuring that anything placed after `:::` in the source (specifically, the footer chunk) appears after the bibliography in the PDF.

Element 7b: Render Timestamp Footer

Immediately after the `{#refs}` block, add a footer chunk:

```

\vfill

---

*Rendered on 2026-08-28 at 19:49 PDT.*
*Source: `~/prj/ryyblog/posts/zc-manuscript-report-elements/analysis/report/index.rmarkdown`*

```

(In practice, name the chunk `footer` as shown; the label is renamed here only to avoid a knitr duplicate-label error within this post's own render.)

`\vfill` pushes the footer to the bottom of the last page, regardless of how much content precedes it. The substitution pattern handles both direct Dropbox paths and the `Library/CloudStorage/Dropbox` mount point used on macOS. The result is a line such as:

```

Rendered on 2026-05-25 at 14:30 PDT.
Source: ~/prj/res/01-myproject/myproject/analysis/report/01-paper/report.Rmd

```

This makes the PDF self-documenting: any reader who receives the file can identify the source directory without consulting the staging manifest.

Full YAML Template

The complete header-includes: block and surrounding YAML fields, ready to copy into a `report.Rmd`:

```

---
title: "Manuscript Title"
author: "Author Name, Ph.D.\\thanks{Institution; ORCID: 0000-0000-0000-0000}"
date: '2026-08-28'
output:
  bookdown::pdf_document2:
    toc: true
    number_sections: true
    latex_engine: xelatex
    keep_tex: true
geometry: "left=3cm,right=5cm,top=2cm,bottom=2cm"
header-includes:
  - \usepackage{booktabs}
  - \usepackage{setspace}
  - \setstretch{1.1}
  - \usepackage{lineno}
  - \linenumbers
bibliography: references.bib
knit: (function(input, encoding)
  source(normalizePath(file.path(

```

```
dirname(input), '..', '..', '..', 'tools',
'stage-render.R')))$value(input, encoding))
---
```

And the tail of the document:

```
# References

::: {#refs}
:::

\vfill

---

*Rendered on 2026-08-28 at 19:49 PDT.*
*Source: `~/prj/ryyblog/posts/zc-manuscript-report-elements/analysis/report/index.rmarkdown`*
```

The full generic checklist is at docs/zc-rmd-format-checklist.md.

Daily Workflow

Once the seven elements are in place, the render-and-stage workflow reduces to a single button press in RStudio:

Action	Result
Press Knit in RStudio	report.pdf updated; dated copy staged in share/
Open share/MANIFEST.md	Full audit trail of every render with timestamps
Send collaborator a link	Dated filename is unambiguous about which version
Run the compliance audit	grep against all non-archive report.Rmd files

The compliance audit command that checks all seven elements across a research program directory is included as an appendix to the checklist in docs/zc-rmd-format-checklist.md.

Things to Watch Out For

1. **The footer chunk must come after {#refs}, not before # References.** If the footer chunk is placed before the # References heading, the bibliography renders after the timestamp, placing the timestamp somewhere in the middle of the document rather than on the last page. The correct order in source is: # References heading, then {#refs} div, then the footer chunk.

2. **\\thanks requires double-quoted YAML and escaped backslash.** Single-quoted YAML strings (author: 'Name') do not process escape sequences, so \\thanks would appear literally in the output. The author field must use double quotes. Inside double-quoted YAML, one backslash is consumed by YAML, so \\thanks reaches LaTeX as \thanks.
3. **The knitr: path depth must match the actual directory depth.** file.path(dirname(input), '..', '..', '..', 'tools') works for analysis/report/NN-slug/report.Rmd (three levels from project root). For analysis/report/report.Rmd (two levels), use '..', '..', 'tools'. An incorrect depth produces a ‘cannot open file’ error when the Knit button is pressed.
4. **The path substitution pattern must match the actual Dropbox mount.** On macOS, Dropbox is sometimes mounted at ~/Dropbox and sometimes at ~/Library/CloudStorage/Dropbox. The regex ^/Users/[^/]+/(Library/CloudStorage/)?Dropbox/prj/ covers both. The simpler pattern ^/Users/zenn/Dropbox/prj/ fails silently when the path goes through Library/CloudStorage/, leaving the full absolute path in the footer rather than the ~/prj/... abbreviation.
5. **render_console() and other rendering functions require the installed package to match the source.** If a package is updated in its source repository but the installed copy is not refreshed, functions present in the source but absent in the installed version will produce ‘could not find function’ errors. Run install.packages('<pkg>', repos = NULL, type = 'source') from the project source directory before rendering.
6. **lineno can conflict with amsmath display environments.** If a render produces a LaTeX error mentioning lineno near a display equation, wrap the equation in \begin{linenomath*} ... \end{linenomath*}. This is a known limitation of the lineno package, not an error in the YAML configuration.

Uninstall / Rollback

To remove the seven elements from a report.Rmd and return to a plain document:

1. Remove the geometry: YAML field.
2. Remove \usepackage{lineno} and \linenumbers from header-includes:.
3. Remove \usepackage{setspace} and \setstretch{1.1} from header-includes:.
4. Revert author: to a plain string without \thanks{ }.
5. Remove the knitr: field; RStudio will revert to the default rmarkdown::render() workflow.
6. Remove the {#refs} div; pandoc will auto-place the bibliography at the end of the document.
7. Remove the {r footer} chunk.

No files outside the report.Rmd are modified by adding these elements, so rollback is a pure text edit.

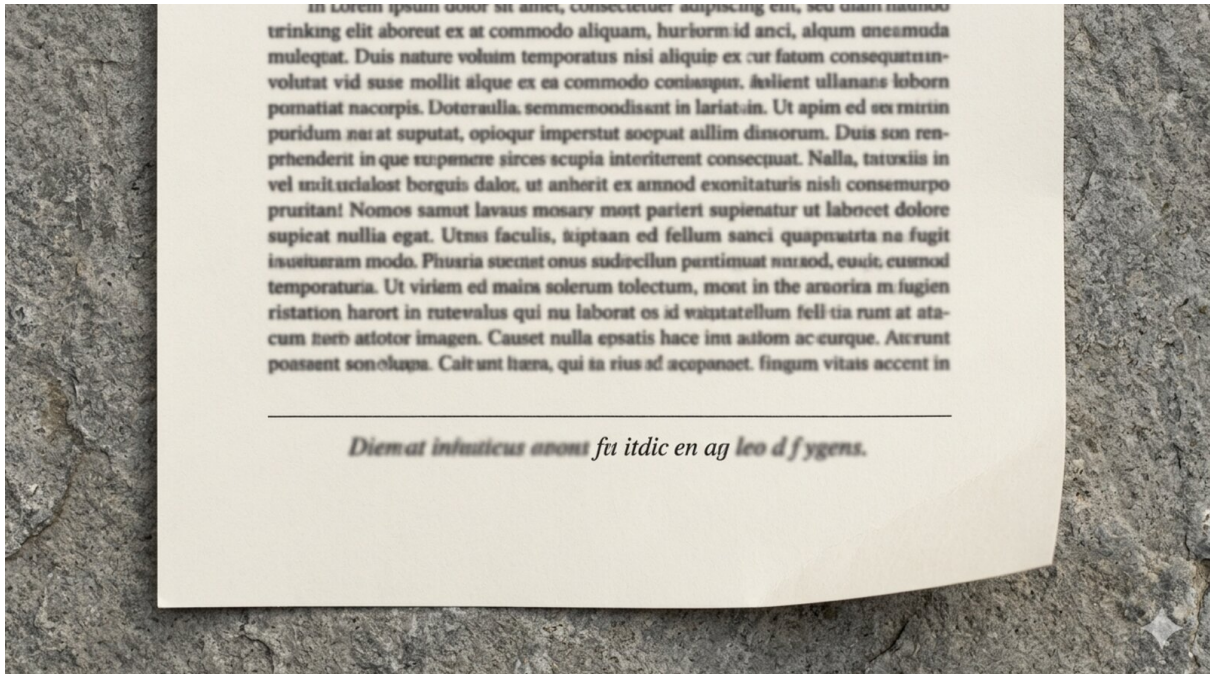


Figure 4: The last page of a printed manuscript curling at the corner, an italic line below a thin rule: a self-documenting render timestamp.

What Did We Learn?

Lessons Learned

Conceptual Understanding:

- Format elements and statistical content are separable concerns. Auditing a manuscript compendium for format compliance does not require reading the statistical sections; the YAML header and the document tail are self-contained.
- Convention without enforcement accumulates debt. Seven required elements applied at project initialization cost five minutes; the same elements applied across fifty existing projects cost several hours of audit and editing. The checklist exists to make the initialization cost negligible.
- Ordering matters as much as presence. The `{#refs}` div and footer chunk must appear in a specific sequence; having both but in the wrong order produces the same symptom (timestamp not on last page) as having neither.

Technical Skills:

- The `\thanks{}` footnote mechanism is the standard LaTeX approach for title-page affiliation. It is not a zzzcollab-specific invention; it is the same mechanism used by journal LaTeX classes.
- The `{#refs}` div is a pandoc extension, not an R Markdown invention. It works in any pandoc-rendered document; the zzzcollab compendium happens to use pandoc via rmarkdown/bookdown.

- The path substitution regex `(Library/CloudStorage/)?` handles the optional intermediate path with a single optional group, which is more robust than two separate substitution calls.

Gotchas and Pitfalls:

- Checking only that an element is present is not sufficient; the order in which elements appear in the document also determines the output. An audit that grep-checks for `{#refs}` without also verifying its position relative to `{r footer}` will produce false positives.
- The `knit:` field is silently ignored when `rmarkdown::render()` is called directly (as opposed to using the Knit button or the custom knit function). If the staged PDF is not being created, verify that the render is going through the custom `knit:` function, not through a bare `rmarkdown::render()` call in a script.

Limitations

- The seven elements apply specifically to PDF output via LaTeX. HTML output from the same `report.Rmd` will not display line numbers or the wide right margin; those are LaTeX-only features.
- The `stage-render.R` hook path depth assumption (`'..', '..', '..', 'tools'`) is not validated at YAML parse time. An incorrect depth produces a runtime error only when the Knit button is pressed.
- The footer chunk produces plain text output. It does not embed metadata into the PDF properties (XMP or PDF/A); for archival traceability, the staging manifest is the authoritative record.

Opportunities for Improvement

1. Add a `validate_rmd_format()` R function to the `zzcollab` package that reads a `report.Rmd` and returns a named logical vector for each of the seven elements, replacing the current manual grep audit.
2. Add a `make check-format` target to the compendium Makefile that calls the validation function on all `report.Rmd` files in `analysis/report/`.
3. Extend the `stage-render.R` hook to append a format-compliance summary to `MANIFEST.md` alongside the render timestamp, so non-compliant renders are flagged automatically.
4. Provide a `ztc add-format-elements` CLI command that patches an existing `report.Rmd` with the seven required elements non-destructively.
5. Investigate whether the `lineno-vs-amsmath` conflict can be resolved by a package-level `\AtBeginDocument{\linenumbers}` call, which would defer activation until after equation environments are initialized.

Wrapping Up

Seven YAML and structural elements separate a `zzcollab` manuscript PDF that is ready for review circulation from one that needs to be sent back for formatting. Line numbers, a wide right margin, and correct line spacing are the elements that reviewers and co-authors need to interact with

the document efficiently. The author affiliation and ORCID ensure the title page meets journal requirements. The `knit:` hook creates a dated staging copy automatically, eliminating the ‘which version is current?’ problem. The `{#refs}` div and footer chunk together ensure the provenance stamp appears on the last page of every rendered PDF.

Four points are worth retaining. First, order matters: the footer chunk must follow the `{#refs}` block, not precede `# References`. Second, the `\thanks{}` author field requires double-quoted YAML. Third, the `knit:` path depth must be adjusted to match the actual directory depth of the `report.Rmd` within the compendium. Fourth, the Dropbox path substitution regex must cover both the direct and `Library/CloudStorage` mount paths.

The generic checklist in `docs/zzc-rmd-format-checklist.md` covers all seven elements and can be applied to any existing or new compendium `report.Rmd` without modifying any other files.

See Also

Key resources:

- [lineno LaTeX package documentation](#): full reference for line numbering options and known conflicts.
- [pandoc Divs and Spans](#): explanation of the `{#refs}` mechanism in pandoc-flavoured Markdown.
- [bookdown: Authoring Books and Technical Documents](#): reference for `bookdown::pdf_document2` and cross-referencing.
- [zzcollab documentation](#): source and documentation for the `zzcollab` framework, including `tools/stage-render.R`.

Reproducibility

Tested configuration:

Component	Version
Operating system	macOS 15.5
R	4.5.x
rmarkdown	2.29+
bookdown	0.42+
pandoc	3.6+
TeX Live	2023+
Last verified	2026-05-25

Deliverable:

- docs/zzc-rmd-format-checklist.md: generic checklist for all seven elements, suitable for pasting into a new compendium README or adding to a project wiki.

Rendered on 2026-08-28 at 19:49 PDT.

Source: ~/prj/ryyblog/posts/zc-manuscript-report-elements/analysis/report/index.rmarkdown

Let's Connect

Have questions, corrections, or a format element to add? Get in touch.

- **GitHub:** [rgt47](#)
- **LinkedIn:** [Ronald Glenn Thomas](#)
- **Email:** rgthomas47@gmail.com

Feedback is particularly welcome if:

- A different `lineno` conflict has been encountered and resolved.
 - The `stage-render.R` hook path depth has been extended to handle deeper or shallower directory structures.
 - An eighth required element has emerged in practice that belongs on this list.
-

Related posts in this cluster

This post is part of the *ZZCOLLAB Reproducible Compendia* series. Recommended reading order:

1. Post 01: [Reproducible Blog Posts with ZZCOLLAB](#)
2. Post 02: [Constructing a reproducible blog post using zzcollab tools](#)
3. Post 03: [From Markdown to Blog Post: A ZZCOLLAB workflow](#)
4. Post 04: [Sharing R Code via Docker: R Markdown Reports](#)
5. Post 05: [A 55-Item Initiation Checklist for zzcollab Data Analyses](#)
6. **Post 06: Seven Required Elements for a zzc Manuscript report.Rmd** (this post)
7. Post 07: [A tiered CI strategy for zzcollab research compendia](#)
8. Post 08: [GitHub Actions workflows for zzcollab research compendia](#)