

Writing CHI Proceedings Papers With R Markdown*

Using the 2018 ACM L^AT_EX template[†]

Ben Trovato

trovato@corporation.com
Institute for Clarity in Documentation
Dublin, Ohio

G.K.M. Tobin

webmaster@marysville-ohio.com
Institute for Clarity in Documentation
Dublin, Ohio

Lars Thørväld

larst@affiliation.org
The Thørväld Group
Hekla, Iceland

Valerie Béranger

Inria Paris-Rocquencourt
Rocquencourt, France

Aparna Patel

Rajiv Gandhi University
Doimukh, Arunachal Pradesh, India

Huifen Chan

Tsinghua University
Haidian Qu, Beijing Shi, China

Charles Palmer

cpalmer@prl.com
Palmer Research Laboratories
San Antonio, Texas, China

John Smith

jsmith@affiliation.org
The Thørväld Group

Julius P. Kumquat

jpkumquat@consortium.net
The Kumquat Consortium



Figure 1: Write your teaser caption here

ABSTRACT

This is a guided illustration of how to write a full paper for CHI, using the official ACM Master L^AT_EX template.

*Who needs titlenotes anyway?

[†]Certainly can't recall the last time I used subtitlenotes, but it's good to have the option.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
CHI 2019, May 4–9, 2019, Glasgow, Scotland UK

© 2019 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 123-4567-24-567/08/06...\$15.00

https://doi.org/10.475/123_4

CCS CONCEPTS

• **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability.

KEYWORDS

ACM proceedings, L^AT_EX, text tagging

ACM Reference Format:

Ben Trovato, G.K.M. Tobin, Lars Thørväld, Valerie Béranger, Aparna Patel, Huifen Chan, Charles Palmer, John Smith, and Julius P. Kumquat. 1997. Writing CHI Proceedings Papers With R Markdown: Using the 2018 ACM L^AT_EX template. In *CHI Conference on Human Factors in Computing Systems Proceedings (CHI 2019)*, May 4–9, 2019, Glasgow, Scotland UK. ACM, New York, NY, USA, 7 pages. https://doi.org/10.475/123_4

INTRODUCTION

Using a tool like R Markdown to write scientific papers makes your work more transparent and reproducible. It also reduces

the risk of errors, because you can dynamically insert tables, figures, and summary statistics directly from the data they are generated from instead of transferring results manually from statistical software to manuscript.

This example illustrates how to use the ACM Master LaTeX template with R Markdown to write papers for the CHI conference, in the CHI proceedings format. The content in this example is adapted and adjusted from content in the **sample-sigchi.tex** template included with the ACM template, to illustrate how to create the same content through the R Markdown workflow as well as to showcase additional features enabled by R Markdown.

1 PAPER META DATA

Set meta data (copyright, authors, keywords, title, keywords, optional teaser figure, etc.) in the YAML header of the .Rmd file in which you write the manuscript. This is done in the form of key: value pairs, e.g. title: Writing CHI Proceedings Papers With R Markdown. When compiling to a PDF (in RStudio, just click the ‘Knit’ button), the information in the YAML header is plugged into the CHI Extended Abstracts LaTeX template. (If you were to take a look at this template file inside of the rticles package, you would see e.g. \title[$\text{\$short-title}$]{ $\text{\$title}$ } where stuff between dollar signs is interpreted as a variable to be searched for in the YAML header and plugged into the template when generating a PDF).

Note the sole exception for adding paper meta data: The CCS Concepts are messy to insert from the YAML header, so you should manually insert this into the **ccsxml.tex** file from which it will be included into your manuscript.

2 THE BODY OF THE PAPER

Typically, the body of a paper has a hierarchical structure, with numbered or unnumbered headings for sections, subsections, sub-subsections, and paragraphs. Whereas in LaTeX you use the command \section for main sections, in R Markdown you simply use #, as in # The Body of The Paper. For subsections, or sub-subsections, use additional hashes, as in ## This Become a Subsection, and #### This Becomes a Paragraph Heading.¹

If you want some section to be unnumbered in the output, add {-} after the section name, as in # Unnumbered Section{-}.

Indicate the start of a new paragraph with a blank line in your input file; that is why this sentence forms a separate paragraph. This line, however, does not form a separate paragraph.

¹By the way, this is how to insert footnotes.

Type Changes and Special Characters

Make words or phrases *italicized* by surrounding them with a single `*`; **bolden** them by surrounding them with **`**two**`**. Typewriter-style (for instance, for computer code) you create by surrounding text with ``backticks``.²

Citations

Citations to articles [1, 2, 4], conference proceedings [3] or maybe books [5, 6] listed in the Bibliography section of your article will occur throughout the text of your article. To insert a reference in the R Markdown syntax, type @ followed by the citation key. The key is a short reference uniquely identifying each entry in the .bib file for your article, in which your references are listed in BibTeX format.

For example, to cite the article “Deciding equivalences among conjunctive aggregate queries” from our .bib file, write [Cohen07]. If you drop the []’s, you get author names, as well as the citation: Cohen et al. [4]. See this short guide for more.

3 DYNAMIC REPORTING

One of the most important benefits of writing in R Markdown (aside from being able to compile to other formats than PDF, such as HTML or even Microsoft Word), is the ability to insert results dynamically into your manuscript using code chunks or inline code. This means that you can do analyses **directly** in your manuscript or, probably better, read file(s) with data, summaries, or results directly into your manuscript and refer to them dynamically.

This is important for two (related) reasons: 1. You avoid initial manual transfer of results from statistical software to manuscript, which reduces the risk of error. 2. If at a later stage you update the analysis files, the results reported in your manuscript are automatically also updated - this again reduces the risk of mistakes, because you don’t need to manually update figures and tables.

In R Markdown syntax, **code chunks** have the following form (cf. *R Markdown: The Definitive Guide*):

```
```{coding_language chunk-label, chunk_options}
your code goes here
```
```

Inline code has the form ``coding_language #code here``.

Setup chunks and figure descriptions

The first chunk in an R Markdown document is usually used to load packages and set default chunk options, for example like so (we normally add the chunk option `include=FALSE` to not include output from this chunk in the manuscript;

²Another footnote here. Let’s make this a rather long one to see how it looks.

here we just add `message=FALSE` to suppress the message that the tidyverse package has been loaded):

```
library(tidyverse)
knitr::opts_chunk$set(echo = FALSE,
  message = FALSE, warning = FALSE)
# these options will exclude code output,
# messages, or warnings in knitted manuscript
```

In addition, version 1.56 of the ACM Master template adds the ability to provide descriptions of figures via the latex command `\Description{my description}`. To be able to add these descriptions easily, as well as an option to position chunks vertically, include this code in your initial setup chunk:

```
# create additional chunk options
hook_chunk = knit_hooks$get('chunk')
knit_hooks$set(chunk = function(x, options) {
  txt = hook_chunk(x, options)
  # add chunk option 'vspaceout' which positions
  # chunks vertically with \vspace
  if (!is.null(options$vspaceout)) {
    latex_vspace <- paste0("\\1\\vspace\\{",
      options$vspaceout, "\\}")
    txt <- sub('(\\begin[\\*]+)',
      latex_vspace, txt)
  }
  # add chunk option 'description' which adds
  # \Description{...} to figures
  if (!is.null(options$description)) {
    latex_include <- paste0("\\1\\Description\\{",
      options$description, "\\}")
    gsub('(\\includegraphics[\\*]+)',
      latex_include, txt)
  } else {
    return(txt) # pass to default hook
  }
})
```

You can then add descriptions to your figures by setting `description="my description"` as a chunk option to images and plots as you will see below.

Inline results

You might read in a made-up data set of goals scored by basketball players like so:

```
data <- read_csv("data/fakeBasketData.csv")
```

We can use inline code to dynamically report properties of this data set. For example, “there are a total of 270 observations of goals scored. The mean number of goals made by any player in a given game is: 17.2555556”.

Table 1: Frequency of Special Characters

| Non-English or Math | Frequency | Comments |
|-----------------------------|-------------|-------------------|
| Ø | 1 in 1,000 | For Swedish names |
| π | 1 in 5 | Common in math |
| \$ | 4 in 5 | Used in business |
| Ψ ₁ ² | 1 in 40,000 | Unexplained usage |

Table 2: The first 5 rows of some made-up basket data.

| Player | goals |
|-----------------|-------|
| Carmelo Anthony | 4 |
| Carmelo Anthony | 2 |
| Carmelo Anthony | 10 |
| Carmelo Anthony | 3 |
| Carmelo Anthony | 1 |

Table 3: Summary statistics of goals scored by top players in made-up basketball season.

| Player | Total goals scored |
|-----------------|--------------------|
| Blake Griffin | 406 |
| Brook Lopez | 776 |
| Carmelo Anthony | 166 |
| Damian Lillard | 808 |
| David Lee | 362 |
| David West | 492 |
| Demar Derozan | 972 |
| Deron Williams | 365 |
| Dwyane Wade | 312 |

Tables

For tables, you could use LaTeX syntax directly. This might be useful if your table itself contains LaTeX syntax, as in Table 1.

However, the power of writing in R Markdown is that you can read in data and automatically create corresponding LaTeX tables. The easiest way is probably to use the `kable` function. For example, Table 2 shows the first 5 rows in our basket data set.

You can reference Table 2 with `@ref(tab:basket-data)`.

You can also do arbitrary transformations and analyses of the data before creating a table, as in Table 3.

To set a wider table, which takes up the whole width of the page’s live area, put it in a `\table*` environment by adding the parameter `table.env = 'table*'` to the `kable` function, like in Table 4.

Table 4: Bigger display of more summary statistics of goals scored by top players in made-up basketball season.

| Player | Total goals scored | Goals per game |
|-----------------|--------------------|----------------|
| Blake Griffin | 406 | 13.533333 |
| Brook Lopez | 776 | 25.866667 |
| Carmelo Anthony | 166 | 5.533333 |
| Damian Lillard | 808 | 26.933333 |
| David Lee | 362 | 12.066667 |
| David West | 492 | 16.400000 |
| Demar Derozan | 972 | 32.400000 |
| Deron Williams | 365 | 12.166667 |
| Dwyane Wade | 312 | 10.400000 |



Figure 2: Here’s a little pretty fly.



Figure 3: A sample black and white graphic that has been resized with the `out.height` and `out.width` chunk options.

Figures

Static figures. Figures are similarly included via code chunks. You can include arbitrary image files, as in Figure 2.

If you don’t give it a caption in the chunk options (with something like `fig.cap="My caption"`), the figure does not float:



You can resize the figures with the chunk options `out.height` and `out.width`, as in Figure 3. If you only care about LaTeX output, you can resize e.g. in inches or relative to the column width (`out.height = '1in'` or `out.height = '0.50\columnwidth'`), but if you want to get maximum value out of R Markdown and be able to output also to html formats, set it with a percentage (`out.height = '50%'` - when outputting to PDF via LaTeX, this will be translated into `out.height = '.5\linewidth'`, see the bookdown reference).

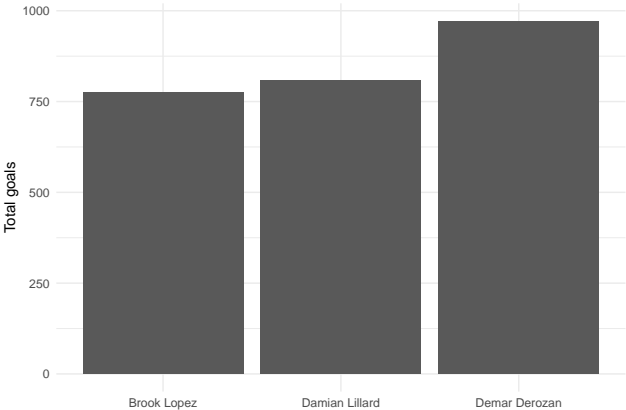


Figure 4: Total number of goals by the top 3 players in made-up basketball season

If you need to style text in a caption, or include references in the caption, you have two options (see bookdown on ‘text references’):

1. set the caption with the chunk option `fig.cap` and use LaTeX rather than markdown syntax. As the figure caption is a string, you must escape the LaTeX syntax’s `\` with another `\`. The caption for Figure 3 would then have been written like this: `fig.cap="A sample black and white graphic that has been resized with the \texttt{out.height} and \texttt{out.width} chunk options."`
2. write the caption in the body text with the syntax `(ref:chunk_label)` My caption here. and then refer to it in the chunk options with `fig.cap='(ref:chunk_label)` as we did for the resized fly caption.

Dynamic figures. Again, the power of R Markdown is that you can include e.g. plots that are dynamically generated from the underlying data. For example, Figure 4 is a simple visualisation of the basket data.

As with tables, you may want a figure to span two columns. To do this, set the environment to `figure*` with the chunk option `fig.env = 'figure*'`. You can fiddle around with the size and aspect ratio of the generated plot with the chunk options `fig.height` and `fig.width`. If your image is very large, you may want to restrict its width with `out.width`.

Math Equations

You may want to display math equations in three distinct styles: inline, numbered or non-numbered display. Each of the three are discussed in the next sections. You can use usual LaTeX syntax directly, or R Markdown.

Inline (In-text) Equations. A formula that appears in the running text is called an inline or in-text formula. In LaTeX it is produced by the **math** environment, which can be invoked by surrounding text with dollar signs: $\$$. You can use any of the symbols and structures, from α to ω , available in LaTeX. For example, here's a nice equation inline: $\lim_{n \rightarrow \infty} x = 0$. If you're writing in RStudio, you can even hover over it to see the rendered output displayed!

Display Equations. A numbered display equation—one set off by vertical space from the text and centered horizontally—is produced by using LaTeX syntax directly to put the content in an equation environment³. So here's that nice equation from above:

$$\lim_{n \rightarrow \infty} x = 0 \quad (1)$$

They can be assigned labels with the syntax `(\#eq:label)`. Refer to the equation with `\@ref(eq:display-equation)`, e.g. see Equation (1).

To make an unnumbered display equation, surround the expression with two dollar signs:

$$\lim_{n \rightarrow \infty} x = 0$$

Theorem-like Constructs

To create theorems, use this syntax (`echo=TRUE` is only necessary if you've set `echo=FALSE` as the default):

```
```{theorem, echo=TRUE}
Here is my theorem.
```
```

For example:

THEOREM 3.1. *Let f be continuous on $[a, b]$. If G is an anti-derivative for f on $[a, b]$, then*

$$\int_a^b f(t) dt = G(b) - G(a).$$

³In fact, you can use any arbitrary LaTeX syntax directly in your .Rmd document.

Similarly for definitions, use the syntax

```
```{definition}
Here is my theorem.
```
```

For example:

Definition 3.1. If z is irrational, then by e^z we mean the unique number that has logarithm z :

$$\log e^z = z.$$

In the ACM LaTeX template, pre-defined theorem-like constructs are **theorem**, **conjecture**, **proposition**, **lemma** and **corollary**. The pre-defined definition-like constructs are **example** and **definition**.

Unsurprisingly, for proofs use

```
```{proof}
Here is my theorem.
```
```

For example:

PROOF. Suppose on the contrary there exists a real number L such that

$$\lim_{x \rightarrow \infty} \frac{f(x)}{g(x)} = L.$$

Then

$$l = \lim_{x \rightarrow c} f(x) = \lim_{x \rightarrow c} \left[g(x) \cdot \frac{f(x)}{g(x)} \right] = \lim_{x \rightarrow c} g(x) \cdot \lim_{x \rightarrow c} \frac{f(x)}{g(x)} = 0 \cdot L = 0,$$

which contradicts our assumption that $l \neq 0$. \square

4 CONCLUSIONS

This paragraph ends the body of this sample document. Remember that you might still have Acknowledgments or Appendices; brief samples of these follow. There is still the Bibliography to deal with; and we will make a disclaimer about that here: with the exception of the reference to the LaTeX book, the citations in this paper are to articles which have nothing to do with the present subject and are used as examples only.

A HEADINGS IN APPENDICES

The rules about hierarchical headings discussed above for the body of the article are different in the appendices. You begin the **appendix** section with the special header `# (APPENDIX) Appendix {-}`. Then, any subsequent top level headers (`#`) indicates the start of each Appendix, with alphabetic order designation (i.e., the first is A, the second B, etc.). So, if you need hierarchical structure *within* an Appendix, start with **subsection** (`##`) as the highest level. Here is an outline of the body of this document in Appendix-appropriate form:

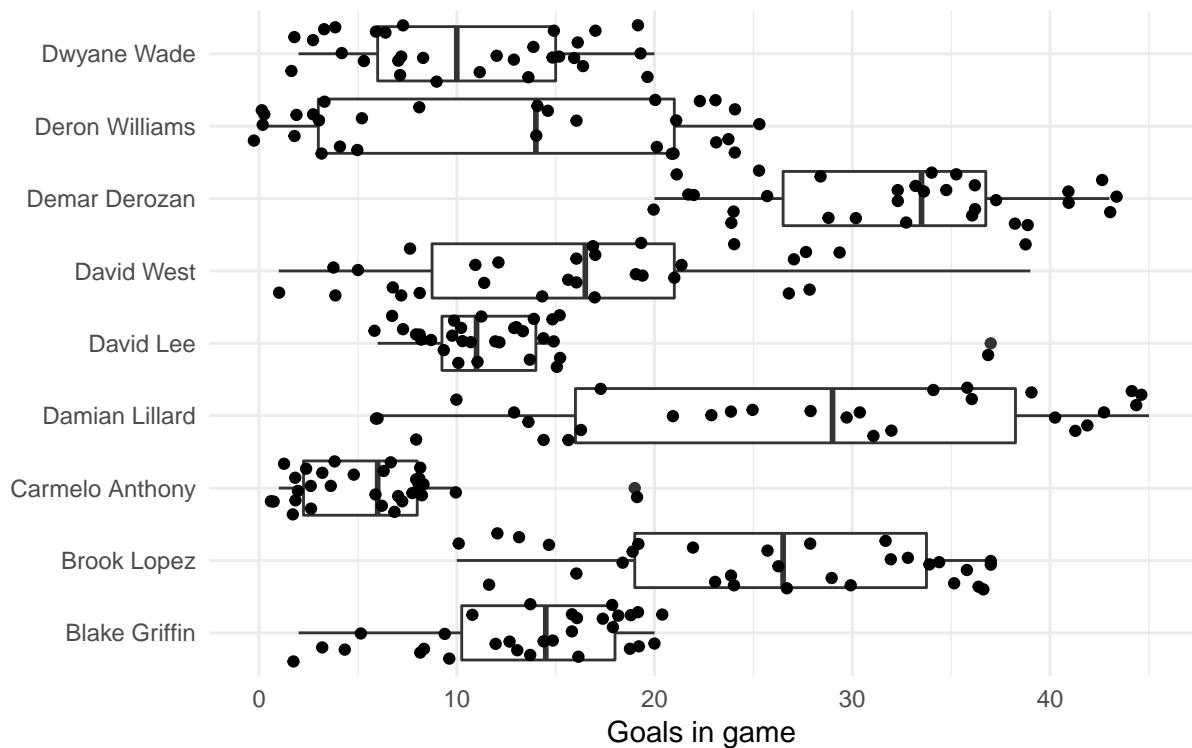


Figure 5: Distribution of goals scored by game for players in made-up basketball season

Introduction

Paper meta data

The Body of the Paper

Type Changes and Special Characters.

Citations.

Dynamic reporting

Inline results.

Tables.

Figures.

Math Equations.

Inline (In-text) Equations.

Display Equations.

Theorem-like Constructs.

Conclusions

References

B MORE HELP FOR THE HARDY

For acknowledgements, you may want to use the LaTeX syntax for this from the ACM template example, in which case

you'll put acknowledgement text in between `\begin{acks}` and `\end{acks}`. Alternatively, just start an unnumbered heading `# Acknowledgements{-}` and write your text, like this:

ACKNOWLEDGEMENTS

The authors would like to thank Dr. Yuhua Li for providing the MATLAB code of the BEPS method.

The authors would also like to thank the anonymous referees for their valuable comments and helpful suggestions. The work is supported by the National Natural Science Foundation of China under Grant No. 61273304 and Young Scientists' Support Program (<http://www.nnsf.cn/youngscientists>).

REFERENCES

[1] Mic Bowman, Saumya K. Debray, and Larry L. Peterson. 1993. Reasoning About Naming Systems. *ACM Trans. Program. Lang. Syst.* 15, 5 (November 1993), 795–825. <https://doi.org/10.1145/161468.161471>

[2] Johannes Braams. 1991. Babel, a Multilingual Style-Option System for Use with LaTeX's Standard Document Styles. *TUGboat* 12, 2 (June 1991), 291–301.

[3] Malcolm Clark. 1991. Post Congress Tristesse. In *TeX90 Conference Proceedings*. TeX Users Group, 84–89.

[4] Sarah Cohen, Werner Nutt, and Yehoshua Sagie. 2007. Deciding equivalences among conjunctive aggregate queries. *J. ACM* 54, 2, Article 5 (April 2007), 50 pages. <https://doi.org/10.1145/1219092.1219093>

[5] Leslie Lamport. 1986. *TeX: A Document Preparation System*. Addison-Wesley, Reading, MA.

[6] S.L. Salas and Einar Hille. 1978. *Calculus: One and Several Variable*. John Wiley and Sons, New York.