Writing CHI Proceedings Papers With R Markdown*

Using the 2018 ACM LaTeX template†

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

G.K.M. Tobin
Institute for Clarity in Documentation
Dublin, Ohio
webmaster@marysville-ohio.com

Lars Thørväld
The Thørväld Group
Hekla, Iceland
larst@affiliation.org

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

John Smith
The Thørväld Group
jsmith@affiliation.org

Julius P. Kumquat
The Kumquat Consortium
jpkumquat@consortium.net

Figure 1: Write your teaser caption here

ABSTRACT

This is a guided illustration of how to write a full paper for CHI, using the latest official ACM LaTeX template Version 1.55, September 11, 2018.

*Who needs titlenotes anyway?

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

WOODSTOCK'97, July 1997, El Paso, Texas USA

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, LaTeX, 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 LaTeX template. In *Proceedings of ACM Woodstock conference (WOODSTOCK'97).* ACM, New York, NY, USA, 6 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 insted of transferring results manually from statistical software to manuscript.

This example illustrates how to use the latest ACM LaTeX templates (Version 1.55, from September 11, 2018) 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 made possible 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 Proceedings LaTeX template (specified in the YAML header with `template: sample-sigchi.tex`). If you take a look at this template file, you will see e.g. `\title[$short-title$]{$title$}` - things between dollar signs are 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 either the template `sample-sigchi.tex` or the **.tex** file generated for your manuscript when you knit to PDF.

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.

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

Type Changes and *Special* Characters

Make words or phrases *italicized* by surrounding them with a single *; **embolden** 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 in the `.bib` file for your article, in which your references are listed in BibTex format.

For example, to cite the article "Deciding equivalances among conjunctive aggregate queries" from our `.bib` file, write `[@Cohen07]`[4]. 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:

```
```{ coding_language chunk_label, chunk_options}
# your code goes here
```
```

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

Setup chunks

The first chunk in an R Markdown document is usually used to load packages and set default chunk options, for example like so:

```
```{ r setup, include=FALSE}
library(tidyverse)
```

[2]Another footnote here. Let's make this a rather long one to see how it looks.
```

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
Carmelo Anthony	3

````
knitr::opts_chunk$set(echo = FALSE,
  message = FALSE, warning = FALSE)
# by default, don't include code output,
# messages, or warnings in manuscript
```
````

You might read in a made-up data set of goals scored by basketball players like so (the chunk option `include=FALSE` means we don't want this chunk to have any output in the manuscript):

````
```{ r, include=FALSE}
data <- read_csv("data/fakeBasketData.csv")
```
````

Inline results

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".

Tables

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

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

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.

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.

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 `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.

At the moment, if you need to style text in the caption, or include references in the caption, you need to use LaTeX rather than markdown syntax. As the figure caption is a string, you must escape the LaTeX syntax's `\` with another `\`, as in Figure 3.
````

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

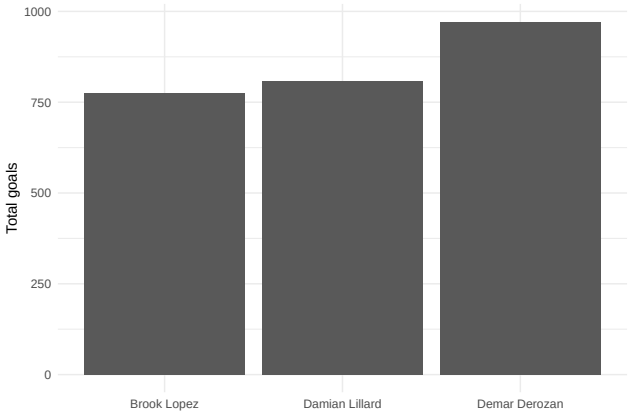

Figure 4: Total number of goals by the top 3 players in made-up basketball season

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 (note that we restrict the figure size to the column width).

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*'`. If your image is very large, restrict its width to the text width with the chunk option `out.width='0.98\\textwidth'`.

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. 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: $\lim_{n\to\infty} x = 0$. 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` environment[3]. So here's that nice equation from above:

$$\lim_{n\to\infty} x = 0 \tag{1}$$

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

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

Theorem-like Constructs

See bookdown.org for guidance if you want to do it the R Markdown way. Here's the usual way, using LaTeX syntax:

Theorem 3.1. *Let f be continuous on $[a, b]$. If G is an antiderivative for f on $[a, b]$, then*

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

Here is a definition:

Definition 3.2. 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**.

Another construct is **proof**, for example,

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

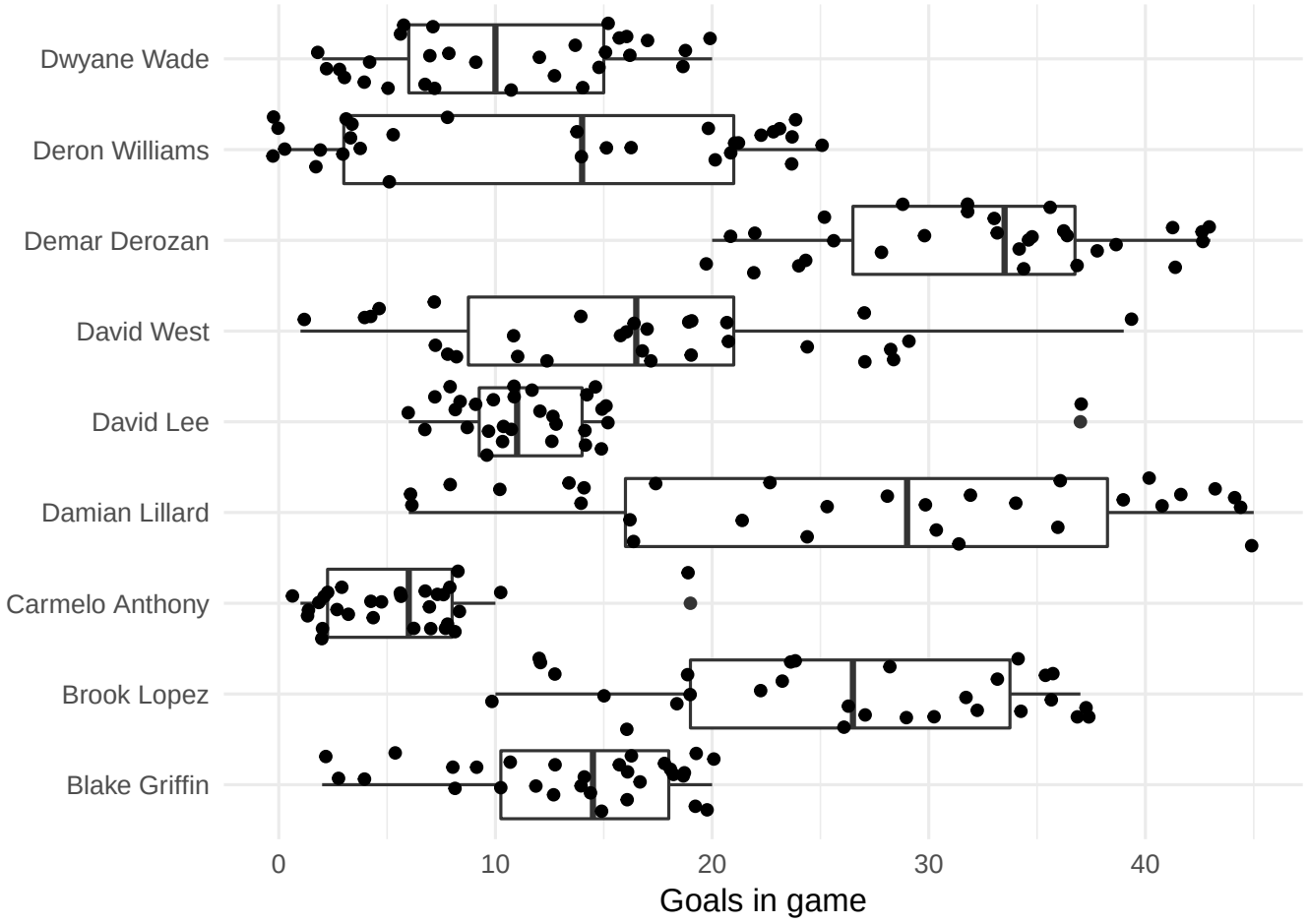

Figure 5: Distribution of goals scored by game for players in made-up basketball season

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

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

Then

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

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

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 being 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:

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:

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 Sagic. 2007. Deciding equivalances 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. *LaTeX: 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.