Structure + Style = Communication

Jeffrey Scott Vitter
The University of Kansas
Lawrence, KS 66045-7518

July 21, 2011

As I enter my $(100,000)_2$th year as a faculty member in computer science, I repeatedly feel compelled to give the same basic lessons about how to write — in a variety of settings and to young and old alike. After all, what could be more basic to success than the ability to communicate? In this article, I have distilled the essence of those writing lessons into a list of useful tips and observations to help steer folks of all backgrounds onto the proper path toward literate communication. I also discuss some nuances of technical writing with TEX [2] and LATEX [4].

1. General Writing and Organization

- First things first. When composing a piece of writing with multiple points and facets, such as an essay, technical paper, or business letter, start your writing in a hierarchical, top-down manner. I highly recommend *version-by-version refinement*, which is a simple outlining technique that forces you into proper organization. Start with Version 1 as a rough outline of the main points and topics in the order of treatment. On the side, make a list of all the little detailed ideas and points that come to you in the process, but you don't yet know where they should go. (Don't put them into the outline yet; you're not ready for that stage.)

  Version 2 should be an elaboration of Version 1, in which you identify the section titles and specify the main ideas of each section. Insert the points you want to make into the relevant sections, so that you know where they belong. Version 3 should be a further elaboration of Version 2 with specific subideas and a clear story line, some of which is now in text form rather than outline form. Version 4 should be close to the final version, if not the final version itself.

- Don't start writing any other way. *DON'T START WRITING ANY OTHER WAY!!!* Pardon the capitalization; I get carried away on this point because it is far and away the most important. Even though it seems like you're doing extra or redundant work to go through the version-by-version refinement process, in reality it will save you valuable time, and the result will be organized and coherent. For you computer scientists, the extra percentage of work is bounded by a geometrically decreasing series, so there's only a constant factor of extra overhead. Failure to use version-by-version refinement will cost you much more time because of rewrites, not to mention the likelihood that no one will be happy with the result.

- Go for clarity and simplicity. Review your writing to streamline it. Eliminate unneeded words, phrases, and sentences.

- In papers, state the main contributions clearly in the abstract. The abstract and the introduction of a paper are very important in conference paper submissions; some referees seem to form

their opinions based only upon the first two pages. Explain the problem and background, state the contributions, and explain their importance. The same advice applies to writing letters: tell the reader why you're writing the letter in the first paragraph, if not the first sentence.

- Use active voice in your writing as much as possible. It's good style to write in first person, using the group "we," which helps promote use of active voice. Avoid use of "I" in professional reports and papers, unless you're taking a personal perspective as I am in this article.
- Place the verb close to the subject. Otherwise, in long sentences, confusion will reign. Consider this example: "Any suggestion that the cause of the poor throughput that rendered the application unusable was rooted in slow hardware and not sloppy software design is unreasonable." Here's a better rewrite with the verb close to the subject: "It is unreasonable to assert that the poor throughput that rendered the application unusable was rooted in slow hardware and not sloppy software design." Even better: "The poor throughput that rendered the application unusable was not the sole result of slow hardware; the primary culprit was sloppy software design."
- The two stress points of a sentence are at its beginning and its end. The end generally gets the most emphasis, so try to reserve the end of the sentence for the most important element. Structure each sentence to move from the known concept to the unknown, from the given idea to the new, from the background to the result. Put transitional and background words or phrases (such as "Therefore," "Thus," "However," "In the last section," etc.) at the beginning of the sentence, saving the important content for the end.
- You'll probably store what you write in files on your computer. If so, name your files in a logically meaningful way. Move old files to a special directory and give them a new name that reflects the last date of alteration. Do not keep garbage files in the main directory; get rid of them or put them in a special directory (e.g., called `OldStuff` or `MiscJunk`).

2. Grammar and Style

- Avoid use of "filler" words or phrases that carry no real meaning, such as
  - "actually,"
  - "basically,"
  - "essentially,"
  - "it is important to note that" (a phrase only for exceptional situations),
  - "note that,"
  - "now,"
  - "therefore,"
  - "thus."

  Filler words can serve a useful purpose; an example is the use of "thus" or "therefore" to signal a conclusion. But they're easy to overuse, so minimize their use.
- Certain words are awkward in formal papers, and there are better substitutes:
  - Replace "a lot of" with "many" or "much."
  - Spell out contractions. (OK, I'm breaking that rule in this article too to keep the tone informal.)
  - Symbols and abbreviations should be reserved for mathematical formulas. Avoid their use within text. Instead, spell out the corresponding words. For example, replace "$\exists$" by "there exists," "$\forall$" by "for all," "$\Rightarrow$" by "implies," and "$\ni$" and "s.t." by "such that."

- Use a colon to introduce a list of items or a mathematical formula only if the preceding sentence is complete. A colon is inappropriate when the list or formula is a required part of the natural flow of a sentence. In the preceding two rules, look at how the two lists of bulleted items are introduced. In particular, in the former, there is no colon after

  "Avoid use of 'filler' words or phrases that carry no real meaning, such as"

  because what follows acts as the object of "such as." It would be OK to replace "such as" by the phrase (with colon) "such as the following:" or better yet to use the alternate sentence

  "Avoid the following 'filler' words or phrases that carry no real meaning:"

- A colon can also be used to restate or elaborate upon a full sentence: in this example, I am not capitalizing what follows the colon, since my elaboration does not extend to multiple sentences, or otherwise I would capitalize "In" (the word following the colon).

- In a list of three or more items, phrases, or clauses (like this one), put a comma between each pair. The last comma in such a list is referred to as the "serial comma" or "Oxford comma." The general decline of Western civilization has led to journalistic conventions that omit the serial comma, but I urge you to maintain standards of common decency and resist those influences.

- On the western side of the Atlantic, commas and periods are generally placed inside the closing quotation mark, even when the logical grouping would suggest otherwise, as in the list of bulleted items at the beginning of this section. We computer scientists often have trouble with this rule! For reasons of clarity, I often violate the rule when describing edits I want someone to make to a paper, such as replace "this" with "that". (Did it again!) Fortunately the rule doesn't apply to other forms of punctuation, such as colons, semicolons, question marks, and punctuation marks.

- Unless the clauses are short, put a comma immediately before a coordinating conjunction (such as "and") that connects two independent clauses, and if the second clause starts with an introductory phrase (as this one does), don't put a comma right after the conjunction.

- Follow this example of not putting a comma before "and" when the sentence (or clause) has one subject and uses two different verbs (known as "compound verbs").

- An exception to the previous rule occurs when the sentence is long and a comma would improve readability, but in such cases you'd be well advised to break the sentence into two separate sentences.

- Another logical inconsistency: it's always fine to put a comma before "but" even when there are compound verbs. For example, writing is most effective when original and innovative, but still requires attention to basic mechanics.

- Use "that" (with no comma before it) to introduce a dependent clause that specifies what you're talking about. If instead the purpose of the clause is only to give complementary information, which strictly speaking is not needed for basic understanding, then use "which" and separate the clause by commas from the rest of the sentence. Reread this rule for examples.

- Try not to use "this" or "that" as nouns. Use them instead as adjectives by specifying this *what* or that *what*.

- The words "their" and "they" should always refer to more than one person. Avoid the common mistake of using "their" and "they" to refer to a single person whose gender is not specified, as in "Each person raised their hand when they wanted to ask a question." A correct version is "Each person raised his or her hand to ask a question." It may seem awkward to write "his or her," but it's more awkward to be regarded as illiterate! Better yet: "The people raised their hands to ask questions."

- Use a hyphen between words that combine to form a single adjective. Example: The $O(n)$-I/O algorithms execute $O(n)$ I/Os. (If we forget the hyphen, then we're incorrectly talking about a set of I/O algorithms that are $O(n)$ in number.) Another example: I/O-efficient algorithms are the main goal of the study of I/O efficiency. Some word combinations and adjective phrases that have (happily) become heavily used no longer need hyphens: Several years ago, I used to write the multi-word expression "external-memory algorithms," but now the multiword format is simply "external memory algorithms."

- Hyphens are also used in some compound words, like "mother-in-law," but avoid them whenever possible (the hyphens, that is), unless the compound word is used as an adjective.

- Use an en-dash "–" (obtained in TEX [2] and LATEX [4] by a double hyphen `--`) to indicate a range as in pages 1–4 and months June–July or to replace the word "to" as in the Lawrence–Topeka toll road. Never put a space before or after an en-dash. A more obscure use of an en-dash is to replace a hyphen in a multiword adjective phrase when one of the nouns being connected has multiple words or a hyphen; that's really being pro–en-dash!

- Em-dashes "—" (obtained in TEX and LATEX by a triple hyphen `---`) provide emphasis, usually to separate one part of a sentence from another. I used to not put blank spaces before and after an em-dash, but I've been softened over the years by *The Chicago Manual of Style* [1] and now include spaces before and after.

  For you Mac and PC users, Microsoft Word can be configured in its AutoCorrect settings to automatically convert two consecutive hyphens into a single dash. A pet peeve of mine is that Word makes it hard to use dashes properly: If you put a space before the two hyphens, the dash you end up with in Word is an en-dash, but then you have to go back and manually delete the spaces because en-dashes should never have spaces before or after. If instead you leave out the space before the two hyphens, Word converts them into an em-dash, but then in order to follow the *Chicago* style convention, you have to go back and manually add the missing spaces. Word has it backwards! Oh well . . . , maybe Microsoft will fix it.

  To dash out on a happy note, here's an easy way to remember which dash is which: An old font convention, no longer strictly followed, was that an em-dash is a dash that has the width of a capital M, while the thinner en-dash has the width of a capital N. In typography, the units "em" and "en" are used to describe those traditional lengths. So just remember that "em" refers to M and "en" refers to N, and you'll never again wonder how to tell the difference!

- Spell out all positive numbers less than 10 when used as an adjective. Use the numeral when used as a noun. Example: Rule 1 for writing a paper is to always do at least three or four versions. You should never need 10 versions.

- Use a spell checker periodically to get rid of obvious errors. Be sure you spell-check before distributing a version of the paper.

3. TEX and LATEX Basics

- LATEX documents are stored in a file whose name ends in the extension `.tex`. They begin with a `documentclass` line such as

  ```
  \documentclass[11pt]{article}
  ```

  which specifies that the default font size is 11 points and that the document style is article format. What follows is the preamble, consisting of a series of declarations using `\usepackage` that specify which style packages to use.

- On my web page [8], you can find the files `template.tex` and `template_full.tex` that can serve as templates for a longer article or paper. Several useful macros (which are shorthand definitions of more complicated sets of commands) appear in the style file `jsv1.sty`. You can use any of those macros in your LATEX document by including the line

  ```
  \usepackage{jsv1}
  ```

  in the preamble of the file. Simply make sure that the style file `jsv1.sty` resides in the same directory as your LATEX document.

- Be sure to use `\label` and `\ref` for referring to equations and sections, since their numbering may change when new material is added. It's convenient to have label names that indicate the type and location of the label; for example, just after the `\section` macro that started this section, I included the LATEX code `\label{sec:TeXbasics}`, so that I could produce the reference "Section 3" elsewhere in this report by writing `Section~\ref{sec:TeXbasics}`.

- Use `\cite` for references. BIBTEX is highly recommended. Several large BIBTEX bibliographies in computer science are on the Web, such as at `http://liinwww.ira.uka.de/bibliography/`.

- The LATEX macro `\emph{ ... }` allows you to emphasize important terms when they are first introduced, as well as words in the text that deserve *special* emphasis. For example, the last sentence ends with "... `deserve \emph{special} emphasis`." (An older way of getting italics is `{\em special\/}`, but the simpler `\emph` syntax is preferred. The italic correction `\/` is called for with `\em` in this particular case but is never needed with `\emph`.)

  Nested uses of `\emph` undo one another. So if you use `\emph` inside a `theorem` environment (in which the text is normally italicized), the emphasized text will be upright, not italicized.

- LATEX also defines macros `\textrm`, `\textit`, `\textsl`, `\textbf`, `\textsf`, `\textsc`, and `\texttt`, which format text enclosed in braces into roman, italics, slant, boldface, sans serif, small caps, and typewriter font.

- The `\verb` macro formats its argument verbatim into typewriter font so that even the special characters like `\` and `&` are treated like ordinary characters. The argument to `\verb` must be enclosed in a pair of matching characters (other than a space, letter, or asterisk) that do not appear elsewhere in the argument; for example, the previous sentence starts with "`The \verb#\verb# macro` ...." The `verbatim` environment can be used to simulate multiline typewriter text including the carriage returns.

- Names of identifiers (or variables or program modules) having more than one letter should be formatted so that they appear in italics with the proper spacing. Feel free to use my macro `\id` for that purpose. Single-letter identifiers are formatted correctly in math mode without need for any macros. My style file `jsv1.sty` has macros called `\idrm`, `\idit`, `\idbf`, `\idsf`,

`\idtt`, and `\idcal` for formatting math text into roman, italics, boldface, sans serif, typewriter, and calligraphic font. The macro `\id` is shorthand for `\idit`. The argument to `\idcal` must contain only uppercase letters and spaces. A space must be preceded by `\` to be recognized, and underscores are denoted by `\_`.

The macros work correctly in both text and math modes, and they automatically use the right size in subscripts and superscripts. Example: The LaTeX code

```
We can insert key~\idcal{K} into the hash table \id{table\_ptr} by means
of the function call $\id{hash\_insert}(\idcal{K}, \id{table\_ptr}, M)$,
where $M \leq M_\idrm{max}$ is the size of the table.
```

produces the following:

> "We can insert key $\mathcal{K}$ into the hash table *table_ptr* by means of the function call *hash_insert*($\mathcal{K}$, *table_ptr*, M), where $M \leq M_{\max}$ is the size of the table."

- LaTeX's list-making environments `itemize`, `enumerate`, etc. often leave excessive vertical space between items. A simple fix is to insert my macro `\cramped` before the first `\item`, as I did in the two examples near the beginning of Section 2.
- LaTeX puts extra space before the start of an environment if there is a blank line in the input before the `\begin{ ... }` macro that starts the environment. The extra space is fine for environments like `theorem` and `definition` that logically start a new paragraph, but not usually for list-making environments. Delete the blank lines in the latter case.
- Use `~` and `\␣` and `\@` appropriately to prevent bad line breaks and for proper spacing in TeX. (The ␣ character in the preceding sentence denotes a space.) Example: The LaTeX code

```
Prof.~Vitter wants all Comp.\ Sci.\ students to write English well,
not just those from the U.S\@.   The \LaTeX\ code~\verb#U.S\@.#
instructs \LaTeX\ to treat the previous ``U.S.'' not only as initials
but also as the end of a sentence, so that more spacing is added.
```

produces the following:

> "Prof. Vitter wants all Comp. Sci. students to write English well, not just those from the U.S. The LaTeX code `U.S\@.` instructs LaTeX to treat the previous "U.S." not only as initials but also as the end of a sentence, so that more spacing is added."

Another example:

```
Let~$I$ denote the subset of $n$ points in the interior of circle~$C$.
(Refer to Figures 2, 3, and~4.)
```

produces the following:

> "Let I denote the subset of n points in the interior of circle C. (Refer to Figures 2, 3, and 4.)"

The use of `~` in this example is to prevent an isolated symbol or number from appearing at the beginning of a line of text. There is no `~` before `n points`, because `n` is linked to `points` and thus it would not be isolated if it appeared at the start of a line.

4. Figures and Tables

- It's easy to include pictures or displays in a LaTeX document. The UNIX program `latex` can process files that embed pictures or displays in encapsulated postscript (eps) format. Simply include `\usepackage{graphicx}` in the preamble near the top of the file (after the `\documentclass` line) and use the macro `\includegraphics` to insert the picture. For example, the code

```
\begin{figure}[h]
\begin{center}
\includegraphics[width=2.9in]{picture1}
\caption{Here's a sample figure of something near and dear to my heart.}
\label{fig:figexample}
\end{center}
\end{figure}
```

  reads in the picture of a disk platter from the eps file `picture1.eps` and produces Figure 1.

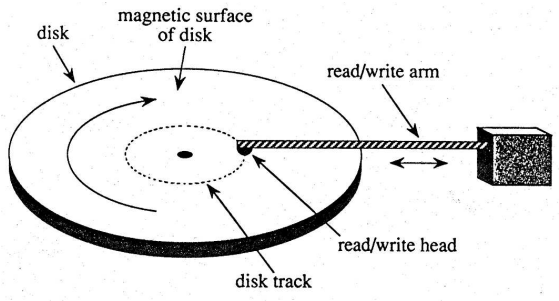

Figure 1: Here's a sample figure of something near and dear to my heart.

The `h` argument to `\begin{figure}` inserts the figure immediately after where it occurs in the LaTeX file. A `b` argument would cause the figure to be "floated" to the bottom of the current page (or if it doesn't fit on the current page, it goes to the next page on which it fits). The argument `t` would instead float the figure to the top of the page. The width argument in the `\includegraphics` macro sets the actual width for the picture, and the picture in `picture1.eps` gets scaled appropriately. You can also scale to a certain height, or even to a certain height and width simultaneously.

- Some newer software packages use the `pdflatex` program rather than the `latex` program. The `pdflatex` program can process files with figures in formats such as pdf, png, jpeg, and tiff, but not eps. When using `pdflatex`, figures in eps format should first be converted into pdf format using a UNIX command such as `epstopdf`.
- Use `\begin{table}` instead of `\begin{figure}` if you want a table instead of a figure. Tables can also be floated to the top or bottom of a page.
- Tables are somewhat cumbersome in LaTeX using the tabular environment, so use some good templates to design them. Several good examples appear in [4]. In the example below, note how `\begin{center}` is used to center the table. The macro `\0` provides horizontal spacing equal to that of a digit, so that the numbers line up correctly.

```
\begin{center}
\def\0{\phantom{0}}
\begin{tabular}{|c|c|c|c|} \hline
\textit{Error Norm}& \textit{Exact} & \textit{Prob.\ Counting} &
\textit{Static} \\
\hline
$\|e^{\mathrm{abs}}\|_1 $  & 52.13 & 52.42 & 52.13 \\
$\|e^{\mathrm{abs}}\|_2 $  & 88.70 & 88.80 & 88.70 \\
$\| e^{\mathrm{rel}} \|_1$ &\00.17 &\00.17 &\00.17 \\
$\|e^{\mathrm{comb}}\|_1$  & 16.71 & 16.72 & 16.71 \\
\hline
\end{tabular}
\end{center}
```

Here's what's produced:

Error Norm	*Exact*	*Prob. Counting*	*Static*
$\|e^{\mathrm{abs}}\|_1$	52.13	52.42	52.13
$\|e^{\mathrm{abs}}\|_2$	88.70	88.80	88.70
$\|e^{\mathrm{rel}}\|_1$	0.17	0.17	0.17
$\|e^{\mathrm{comb}}\|_1$	16.71	16.72	16.71

- It's sometimes useful in TEX and LATEX to insert a zero-width `\strut` into text in order to produce the proper vertical spacing when the text is enclosed in a box or abutted to other items in a vertical list. LATEX often needs extra help. For example, I've found it useful to define macros `\tallstrut` and `\tallerstrut` in my style file `jsv1.sty` to add extra spacing above the first line of boxed tables made with LATEX's tabular environment. Here's an example where I slightly modify the last example by adding `\tallstrut` immediately after the first two instances of `\hline`:

Error Norm	*Exact*	*Prob. Counting*	*Static*
$\|e^{\mathrm{abs}}\|_1$	52.13	52.42	52.13
$\|e^{\mathrm{abs}}\|_2$	88.70	88.80	88.70
$\|e^{\mathrm{rel}}\|_1$	0.17	0.17	0.17
$\|e^{\mathrm{comb}}\|_1$	16.71	16.72	16.71

5. Formatting Mathematics

- Avoid using `\frac` for fractions that appear in text style or when the resulting characters will be too small, such as in exponents, subscripts, and superscripts. For example, use $(N/B)\exp(z/t)$ or $(N/B)e^{z/t}$ in text, but not $\frac{N}{B}e^{\frac{z}{t}}$. It's marginally OK to use $\frac{N}{B}e^{z/t}$, but one of the previous two choices is preferred. This distinction would more obvious if the $e^{z/t}$ term were replaced by something like $\log N$. In that case, $(N/B)\log N$ is definitely better than $\frac{N}{B}\log N$; otherwise, the term that is mathematically the most significant (namely, $\frac{N}{B}$) is dwarfed in type size by the less significant term ($\log N$). In display format, use

$$\frac{N}{B}e^{z/t} \quad \text{and} \quad \frac{N}{B}\log N. \tag{1}$$

- Be sure to include proper punctuation in sentences involving math formulas like $E = mc^2$. Periods always go outside of math mode when the formula is in text, like in the last sentence, but inside math mode when in math display format, as in (1).
- Long sets and sequences should be specified differently when they appear in text than when they are displayed, so as to allow a line break in the text if needed. Example: The LaTeX code

```
The set $\{ x_1$, $x_2$, \dots,~$x_n\}$ is in text style,
but in display style, it should be $$\{ x_1, x_2, \ldots, x_n \}.$$
```

produces the following:

"The set $\{x_1, x_2, \ldots, x_n\}$ is in text style, but in display style, it should be

$$\{x_1, x_2, \ldots, x_n\}.$$"

Always use `\dots` for an ellipsis in text mode. In math mode, `\dots` and `\ldots` are interchangeable, and `\cdots` results in centered dots.

Another example: The LaTeX code

```
The sequence $x_1$, $x_2$,~\dots\ is in text style,
but in display style, it should be $$x_1, x_2, \ldots. $$
```

produces the following:

"The sequence $x_1, x_2, \ldots$ is in text style, but in display style, it should be

$$x_1, x_2, \ldots.$$"

- Use `\langle` and `\rangle`, not `<` and `>`, to get the math angle brackets "$\langle$" and "$\rangle$."

6. Becoming a TeXpert

TeX is a finely tuned machine for producing beautifully formatted documents. We've already seen some basic guidelines for using TeX in Sections 3 and 5. Below are some fine points about how get the most out of TeX and LaTeX.

- Use `\bigl` and `\bigr` in math expressions that appears in text, but `\left` and `\right` in math displays. (There's also `\bigm` for middle delimiters as well as plain ol' `\big`.) In some LaTeX environments, `\bigl`, `\bigr`, etc. don't have any effect in text style, but you can fix that problem by including `\usepackage{amsmath}` in the preamble. Example: The LaTeX code

```
Write $O\bigl( (N^2 (\log N) + \sqrt m \,) / 2 \bigr)$ in text style,
not $O( (N^2 (\log N) + \sqrt m \,) / 2 )$.  In displays, write
$$O\left( \frac{1}{2} \left(N^2 (\log N) + \sqrt m \,\right)\right)
\quad\mbox{or}\quad O\left( \frac{N^2 (\log N) + \sqrt m\,}{2}\right).$$
```

produces the following:

"Write $O\big((N^2(\log N) + \sqrt{m}\,)/2\big)$ in text style, not $O((N^2(\log N) + \sqrt{m}\,)/2)$. In displays, write

$$O\left(\frac{1}{2}\left(N^2(\log N) + \sqrt{m}\,\right)\right) \quad \text{or} \quad O\left(\frac{N^2(\log N) + \sqrt{m}}{2}\right).$$"

Another example: The LaTeX code

```
Write $\bigl\{ \bigl( x(t), y(t)\bigr) \bigm| 0 \leq t \leq n \bigr\}$
instead of $\{ (x(t), y(t)) \mid 0 \leq t \leq n \}$ since $x(t)$ already
has a parenthesis and the larger delimiters make it easier to read.
```

produces the following:

> "Write $\bigl\{ \bigl(x(t), y(t)\bigr) \bigm| 0 \leq t \leq n \bigr\}$ instead of $\{ (x(t), y(t)) \mid 0 \leq t \leq n \}$ since $x(t)$ already has a parenthesis and the larger delimiters make it easier to read."

It would have been even worse in the latter case if `|` were used instead of `\mid`, since

```
$\{ (x(t), y(t)) | 0 \leq t \leq n \}$
```

gives the wrong spacing around the | relation:

> "$\{(x(t), y(t))|0 \leq t \leq n\}$"

- In text style, enclose a tall or deep math expression or subexpression within `\smash{ ... }` to make TeX ignore its height. Otherwise, TeX may add extra space between adjacent lines in the text, which can look awkward, for example, when there is a really deep expression like $x_{h(y_{f_1})}$ on one line and a really tall expression like $2^{2^{2^n}}$ on the next. Often the lines won't interfere with one other if the extra space is eliminated by use of `\smash{ ... }`, and the resulting look (e.g., $2^{2^{2^n}}$) will be much improved. Very tall or deep expressions are best put in displays.

7. What's More

For more extensive guidance on writing, there are several sources worth consulting: My PhD adviser Don Knuth has been the biggest influence on my writing — as he has been for many others. I highly recommend the course notes he coauthored on mathematical writing [3]; they give a useful list of do's and don't's worthy of memorization. And of course, be sure to read Strunk and White's classic text on the elements of style [6]. Williams and Colomb pick up where Strunk and White left off and provide a clear, systematic process for improved writing [9]. On the humorous side, Safire's self-contradicting "Fumblerules" entertain while illustrating basic principles of grammar [5]. A new must-read entry (although from a British point of view!) is Truss's lighthearted look at the importance of punctuation [7].

The writing notes I've included in this article are kept in up-to-date form on my web page [8], along with the source code, the style file, and some template files. My plan is to update the article on a semiregular basis. I've included just a few rules so far; no doubt more will be coming. I welcome any suggestions, corrections, or comments you might have; you can send me email at ⟨jsv@vitter.org⟩. Every suggestion of yours that I implement earns you 2.56 KRW (with payments rounded off to the nearest dollar).

References

[1] *The Chicago Manual of Style*. University of Chicago Press, 16th edition, 2010.

[2] Donald E. Knuth, *Computers & Typesetting*. Volume A: *The TeXbook*. Addison-Wesley, Reading, Massachusetts, 1984.

[3] Donald E. Knuth, Tracy L. Larrabee, and Paul M. Roberts. *Mathematical Writing*. Mathematical Association of America, Washington, D.C., 1989.

[4] Leslie Lamport. *LaTeX: A Document Preparation System*. Addison-Wesley, 1994.

[5] William Safire. *How Not to Write: The Essential Misrules of Grammar*. W. W. Norton & Company, New York, 2005.

[6] William Strunk, Jr. and E. B. White. *The Elements of Style*. Allyn and Bacon, Boston, 5th edition, 2009.

[7] Lynn Truss. *Eats, Shoots & Leaves: The Zero Tolerance Approach to Punctuation*. Gotham Books, New York, 2004.

[8] Jeffrey S. Vitter. Web page `http://www.vitter.org/jsv/`. The updated version of this article, its source code, some LaTeX template files, and the style file `jsv1.sty` can be found by following links to the web page `http://www.provost.ku.edu/jsv/jsvteachinglinks.html`.

[9] Joseph M. Williams and Gregory G. Colomb. *Style: The Basics of Clarity and Grace*. Longman, New York, 4th edition, 2011.