State-of-the-art Document Production

An Executive Summary

Michael P. McLaughlin

June, 2021

1 Introduction

This article is about producing documents that have a physical appearance of the highest quality. The discussion applies to *all* documents especially, but not limited to, "technical" documents (lots of figures, tables, equations, etc.), the kind that appear in professional journals rather than magazines. Indeed, *this document* is, to some extent, a good example.

We are not going to review different ways to produce documents because, for many years, the consensus gold standard for document production has been in a league of its own, supreme and uncontested. Therefore, our entire discussion will focus on this world-class technology and describe why it is such, how it is used and examples of its output.

This is not to suggest that everything is automatic. Writing a high-quality document begins with the desire, motivation and expertise to do so. If any of that is lacking, then no software package will compensate.

1.1 The Gold Standard

The gold standard for document production, called LaTeX, is preeminent partly because of its enormous set of capabilities but also because it was purposely designed to generate printed material of the very highest quality. Today, that output usually takes the form of a PDF file but one has to create the source for that PDF and nothing comes close to LaTeX.

Evidence of this is easy to find. One merely has to visit an online repository such as arxiv.org. This archive contains nearly two million documents (largely preprints) in various disciplines. You would be hard-pressed to find even one of them that was not written using LaTeX even though this is not a requirement for inclusion.

These documents look their best when printed but are almost as pleasing when viewed on a computer display in spite of the quantization error arising from pixel-mapping. At the lowest level, LaTeX works in real, floating-point space where everything, text and/or graphics, is "drawn" with a precision of six significant figures. Sometimes, this precision can be reconstituted on a printer but not on a display.

For this and many other reasons, LaTeX is unquestionably the state of the art.

1.2 Historical Highlights

LaTeX is a second-generation production system. The first generation, called TeX, was developed by Donald Knuth to improve the quality of his multi-volume opus, *The Art of Computer Programming*. In Knuth's own words, TeX was "...intended for the creation of beautiful books" [1]. As such, it is a complete typesetting system suitable for creating all kinds of documents, even books (apart from reproduction and binding). In many cases, authors have written entire textbooks using LaTeX exclusively. For one impressive example, "Look inside" this book. It contains nearly 900 pages and everything in it: text, figures, formatting, Table of Contents, index, etc. was all done in LaTeX and completely typeset with just *one keystroke*, probably in less than one minute.

Knuth is a computer scientist and, for him, TEX is a familiar environment. For most people, however, TEX is much too detailed and low-level so, in 1986, Leslie Lamport created LATEX [2], a user-friendly interface layer superimposed on top of TEX. LATEX is a markup language much like HTML. In other words, LATEX is to TEX as C (or FORTRAN) is to assembly language.

1.3 Some Leading References

The best source for all things LATEX, including LATEX itself, is the Comprehensive TeX Archive Network (CTAN). As usual, in the Macintosh™ world, where quality is almost sacred, there is a customized source for a LATEX installation.

In addition to the book written by Lamport, there are other books [3][4] describing LATEX in general. Supplementing these are books devoted to specialized topics [5][6][7]. There is also a "short" online tutorial.

And, of course, there is a huge amount of topical information available on the Internet.

2 Why LATEX?

The information presented in the Introduction is all well and good but clearly there must be more immediate benefits from changing one's writing habits. Otherwise, authors would continue to work in the fashion to which they are accustomed. Benefits are largely in the eye of the beholder and the list below may well be the idiosyncratic opinion of this author. Nevertheless, I suspect that LATEX users would agree with most of it.

So, here are twenty good reasons for using LATEX instead of an alternative.

- **Output Quality**
  Clearly, this has to be #1. If the examples in this document are not persuasive, especially when printed, then the links in Appendix A should be.

- **Public-domain**
  The software is all open-source though tightly controlled. What this means is that, apart from commercial books, *everything* related to LATEX (and TEX) is free—completely without cost or registration (or cookies).

- **Platform-independent**
  Executables are available for Windows™, Mac OS X and GNU/Linux.

- **Stability**
  Unlike some word-processing systems, there is never any incompatibility arising from different versions of LATEX. Any correct source (.tex) file will generate exactly the same output on all platforms regardless of the age of the installation or the source file. Occasionally, there may be exceptions with third-party packages (see below).

- **Supported Worldwide**
  It is used everywhere, in a large number of languages and alphabets, etc. There is, for example, no problem with right-to-left text along with left-to-right, even on the same page. Standard document options include such things as default font size, *twoside*, *twocolumn*, *A4 paper*, etc., all designed for portability throughout the world.

- **PDF Environment**
  At its lowest level, the rendering of text, figures, drawings, etc. is all done in real-valued, not bit-mapped, space with about six significant figures. This makes all of the output of higher precision than even the best printers. It also enables book-quality typesetting, with variable spacing, etc. whenever needed.

  Moreover, available editors (see below) typically generate the output as a PDF file, directly from the source, with a single keystroke.

- **Automatic Bibliography, TOC, Index, etc.**
  All of the "front matter" and "back matter" found in documents such as Table of Contents, List of Figures (and Tables), Appendices, Bibliography, Glossary, Index, etc. can be generated automatically. Bibliography, Index and Glossary entries must, of course, be flagged as such.

- **Free Editors, etc.**
  Source files can be created using any text editor but, usually, an editor customized for LaTeX is employed. These are specially designed for many of the operations common to LaTeX source.

  There are other applications available for more specialized use, e.g., database apps that output source files for bibliography creation or apps for writing an equation (or equation array) in LaTeX then exporting it as PDF, PNG, etc. for inclusion in non-LaTeX documents.

- **Better than WYSIWYG**
  Current LaTeX editors typically display two windows: one for the source and one for the PDF output. Output is generated with one keystroke and rendered in the PDF window *exactly* as it would appear in a PDF reader or in printed hardcopy. This is much better than other word processors that require a "Print Preview" operation to show what the output would actually look like when printed.

  Thus, LaTeX shows what you really get more easily and faster than alternatives.

- **Labels**
  LaTeX can, of course, insert figure, table and footnote numbers automatically but it goes *far* beyond that. You can (optionally) create a user-defined label for just about anything: figures, tables, equations, sections, . . . , even individual pages. The element so labelled can then be referred to in the text by its label even if that label has yet to be defined.

There is nothing worse than to be told, by a reviewer, that you need an additional equation in between Equation 2 and Equation 3. Not only must you enter the new equation, you must change all textual references to equations after Equation 2 so that they point to the right one. This would be a lengthy and error-prone process. By using labels, this problem never arises. LaTeX resolves all such references (both backward and forward) automatically.

- **Macros**
  LaTeX is actually a *language* describing, in complete detail, how a document is to be formatted and typeset. Therefore, it is extremely configurable and there is almost nothing that you cannot do. For instance, you can create *macros*, abbreviations or new commands (with or without arguments), even redefine built-in commands.

  If you are ambitious, you can even reach down into the lowest-level of the underlying TeX layer and render PostScript (thus PDF) directly. Later examples show some of the incredible power and flexibility that this allows.

- **Specialized Packages (and Symbols)**
  The ability to do whatever you like has given rise to a very large number of third-party packages. These expand the default LaTeX capabilities with new functionality that is sometime quite extensive and elaborate. At the time of writing, the CTAN archive cited above lists more than 6,000 such packages with a vast repertoire of specialized, and often obscure, functionality.

  Along with these packages, there is a set of (currently) 18,150 symbols in various categories. Examples include all kinds of symbols, from € (euro) to ♃ (astrological sign for Jupiter) and just about everything in between.

- **Existing Templates**
  Along with packages, which focus on within-document functionality, one can also create a brand new *class* of document to augment the standard set. The present document, for instance, is an *article*, not a *report*, *book*, *letter* or *slides*.

  A custom class is typically a template intended for use by a fairly large number of people. Many journals create LaTeX classes for submissions. Likewise, many university departments mandate that all candidates for Masters and PhD degrees write their theses using a LaTeX class customized for that department.

- **Excellent Math Capability**
  As you might expect, given its origins, LaTeX is especially powerful when used to write equations. Not only can one do the ordinary, expected thing such as

$$\sum_{k=1}^{n} k = n(n+1)/2 \tag{1}$$

but one can also insert $\sum_{k=1}^{n} k = n(n+1)/2$ inline without the usual distortion of line spacing so common with other text systems. No special effort is required.[1]

- **Excellent Drawing Capability**
  In LaTeX, everything not imported is "drawn" so making a drawing within a .tex file is just one more capability. Examples will be shown later.

- **Textfile Source**
  A source (.tex) file is just a plain text file such as ASCII or UTF8—an ordinary, human-readable file with no proprietary contents of any kind. Nothing is hidden; source is not encoded in any special way nor is it compressed. Not only does this make it legible and portable, it also means that nothing happens to the source file after you type it so nothing can "go wrong" with it that is not immediately obvious when read with sufficient care.

- **Comments**
  A source file contains whatever you like, including (optional) comments, perhaps reminders of why you did something or where to find information for what follows, etc. Typically, editors display comments in a different color.

- **Translatable**
  Another consequence of the source file being plain text is that it can be translated to other formats, e.g., HTML. Utilities are available for such purposes.

- **Automatic Source Generation**
  Yet another benefit of a text source file is that it is not difficult to have it written by a computer program or script. Often, much of a .tex file is boilerplate. That can be defined as a set of constant strings and the variable contents written on the fly. To a programmer, this is all fairly trivial.

- **User-enabling**
  Finally, the ease with which even elaborate formatting can be accomplished with LaTeX is a powerful enabler that should not be underestimated. Authors will find themselves willing to create much more ambitious documents once they know that they can do so with a moderate amount of effort.

Is there any downside to using LaTeX?

Not really unless you have co-authors who cannot cope. There is a learning curve, of course, but it is fairly shallow and there are many online tutorials available. Moreover, you need not learn any more than necessary. Expertise will be gained incrementally in any case and none of it is difficult. Even if you forget things, all you have to do is find a relevant source file that you created before you forgot. Then, just cut-and-paste.

[1] One could construct equations where an inline representation is not as nice but, typically, it is.

3 LaTeX in Action

On the next page, rotated for clarity, we show a typical editor environment, in this case TeXShop. The left window is the editor window; the right window is the resulting PDF rendering. Note: Figure 1 is a PNG screenshot, bit-mapped[2] then shrunk to fit, so its quality is seriously degraded compared to a PDF graphic.

More interesting, perhaps, is what LaTeX can do in practice, shown below.

3.1 Ordinary Stuff

Here, we provide examples of things that one would expect of any document system.

Table 1 is fairly typical. Decimal tabs (col. 6) is not so impressive but footnotes of its own is definitely unusual.[3]

Table 1: A Chain of World Chess Champions

		Winner's Results				
Winner	**Opponent**	**Win**	**Loss**	**Draw**	**Score**	**# Games**
Lasker	Steinitz	10	5	4	12.0	19
Lasker	Steinitz	10	2	5	12.5	17
Capablanca	Lasker	4	0	10	9.0	14
Alekhine	Capablanca	6	3	25	18.5	34
Euwe	Alekhine	9	8	13	15.5	30
Alekhine	Euwe	10	4	11	15.5	25
Smyslov	Euwe	4	1	0	4.0	5[a]
Botvinnik	Smyslov	7	7	10	12.0	24[b]
Botvinnik	Smyslov	7	5	11	12.5	23
Smyslov	Botvinnik	6	3	13	12.5	22
Botvinnik	Tal	10	5	6	13.0	21
Tal	Botvinnik	6	2	13	12.5	21
Petrosian	Botvinnik	5	2	15	12.5	22
Petrosian	Spassky	4	3	17	12.5	24
Spassky	Petrosian	6	4	13	12.5	23
Fischer	Spassky	7	2	11	12.5	20[c]
Fischer	Spassky	10	5	15	17.5	30[d]

[a] within a 5-player round robin
[b] match tied
[c] one (forfeited) game discounted
[d] rematch (1992)

[2] Screenshots are necessarily bit-mapped.
[3] uses packages *tabularx* and *dcolumn*

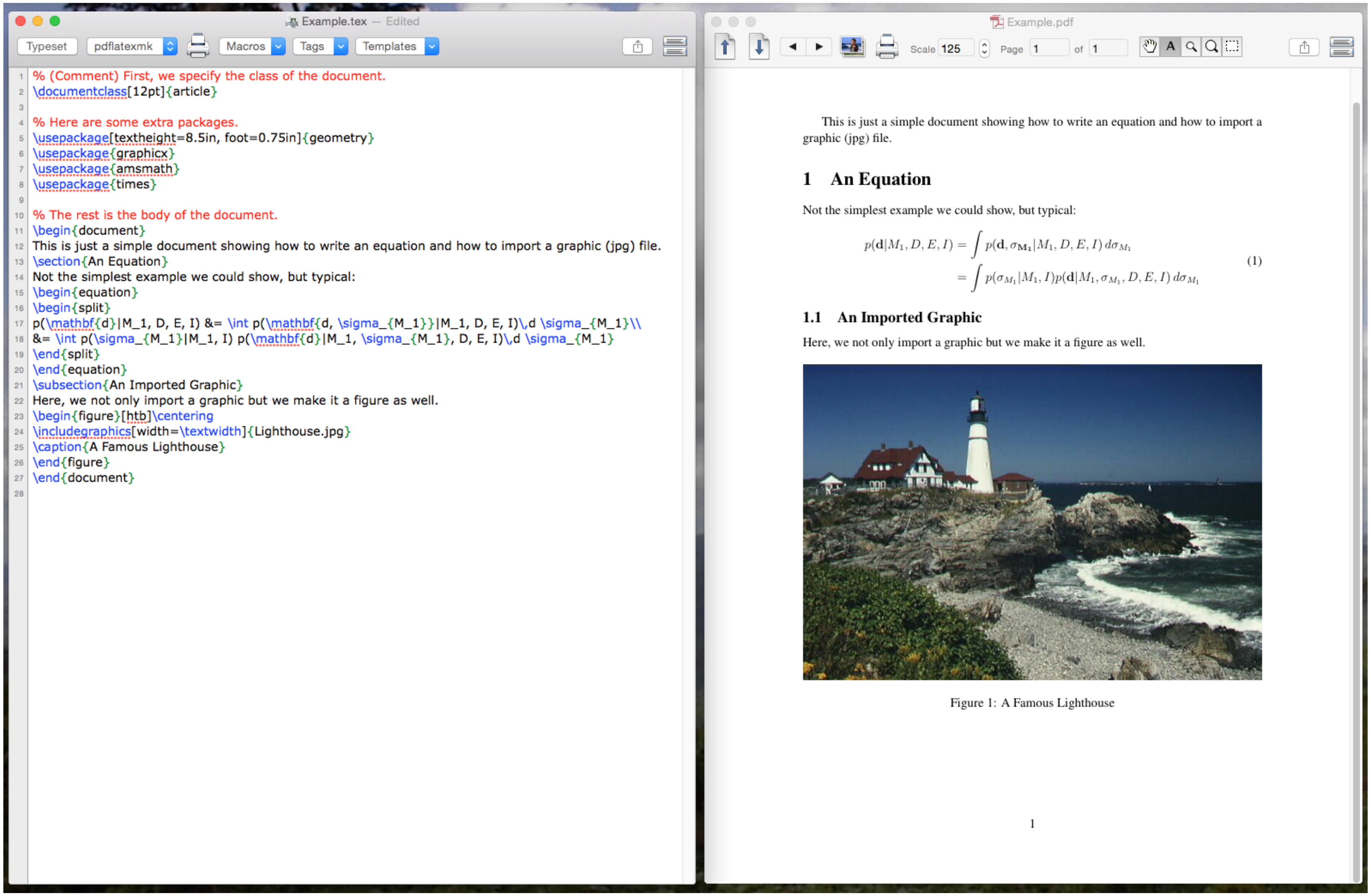


Figure 1: *TeXShop* Editor Environment

Graphics are basic elements. Here is a graphic (optionally) embedded in a figure.

Figure 2: Portland Headlight

Here is the LaTeX source that imported this graphic and created Figure 2. This figure was (optionally) given a label so that it could be referenced in the text without knowing its figure number.

```
\begin{figure}[h]\centering
\label{fig:lighthouse}
\includegraphics[width=\textwidth]{Lighthouse.jpg}
\caption{Portland Headlight}
\end{figure}
```

In technical documents, there are often some equations. Equation 2 is fairly typical although it is split into two lines.

$$\begin{aligned} p(\mathbf{d}|M_1, D, E, I) &= \int p(\mathbf{d}, \sigma_{\mathbf{M_1}}|M_1, D, E, I)\, d\sigma_{M_1} \\ &= \int p(\sigma_{M_1}|M_1, I) p(\mathbf{d}|M_1, \sigma_{M_1}, D, E, I)\, d\sigma_{M_1} \end{aligned} \tag{2}$$

Here is the LaTeX source for this equation.

```
\begin{equation}
\label{eq:splitEq}
\begin{split}
p(\mathbf{d}|M_1, D, E, I) &=
\int p(\mathbf{d, \sigma_{M_1}}|M_1, D, E, I)\,d \sigma_{M_1}\\
&= \int p(\sigma_{M_1}|M_1, I) p(\mathbf{d}|M_1,
\sigma_{M_1}, D, E, I)\,d \sigma_{M_1}
\end{split}
\end{equation}
```

It is probably evident that one can write equations that are quite elaborate because the source is not writing them but telling the computer *how* to write them. LaTeX is a *language*.

3.2 Not So Ordinary Stuff

LaTeX is intended for document preparation in general so much of its capability is devoted to doing things that are well beyond the simple tasks described above. We present a few in this section with the caveat that they are meant to be illustrative, not comprehensive. As noted earlier, non-standard packages offer some very specialized capabilities. We have presented examples of the latter in Appendix A.

Here is another table. Tables with merged columns are very common. LaTeX, as you might expect, allows you to merge rows as well (Table 2, col. 1). A lot more about table formatting is described in reference [7].

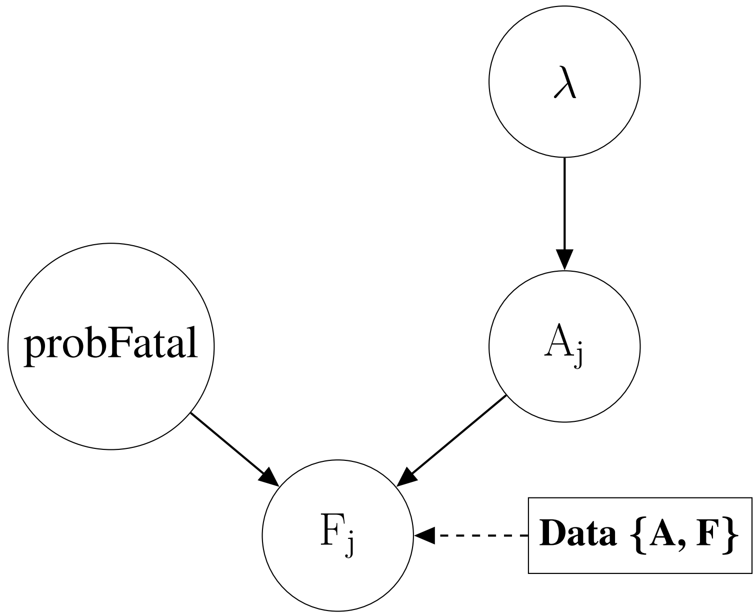
Table 2: Parameter Comparisons

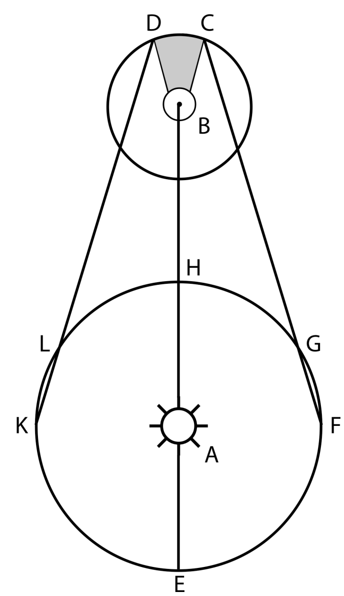
Model	Criterion	μ	σ	λ	p
Quantization	ML	2.6972	8.1236	9.0103	0.7738
No quantization	MAP	2.7094	8.1598	9.3303	0.8020
	Mean	2.7094	8.2826	10.0887	0.7867

The LaTeX language is flexible enough to allow you to draw pictures with it, that is, within the source file, not imported. Such pictures are vector graphics and therefore have very high resolution (six sig. fig.).

A very simple example, a directed graph, is shown at the top of the next page. It took just the 12 lines of LaTeX source (comments discounted) shown in Figure 3.

With a *lot* more effort, you can draw really fancy pictures such as the one shown in Figure 4. We shall *not* overwhelm you with the source for this one!





Sometimes, one wishes to insert "floats" (tables and/or figures) but not use the full page width to do so because the float is simply not that big. This requires changing the global layout in some fashion. In word processors, this could well be a major undertaking fraught with much trial and error—assuming that it were possible at all.

The diagram at the right was drawn in LaTeX and then imported into the text as a graphic. How does it look?

Personally, I would consider it of publishable quality. Moreover, this document was created as a PDF so all of it (except Figure 1, a bitmap) can be magnified on the display without loss of precision. Therefore, the small labels in this diagram can be made bigger if necessary.

This diagram utilized a third-partly package, *floatfig*, to achieve this non-standard behavior. It took 33 lines of source to draw this graphic and three more to import it.

Getting even more non-standard, you can do math by writing LaTeX and having the latter do both the computation as well as show the result. For instance, the following non-numbered equation was produced with nothing but the LaTeX source shown underneath it.

$$\mathrm{e}^2\cos \pi/3 = 3.69453$$

```
\EXP{2}{\exptwo}
\COS{\numberTHIRDPI}{\costhirdpi}
\MULTIPLY{\exptwo}{\costhirdpi}{\sol}
\[ \mathrm{e}^2\cos \pi/3 = \sol \]
```

```
\begin{tikzpicture}[node distance=2cm, auto]
% nodes
\draw (0,0) node[circle,minimum size=2cm,draw] (lik) {\Large{$\mathrm{F_j}$}};
\draw (-3,2.5) node[circle,minimum size=2cm,draw] (p) {\Large{probFatal}};
\draw (3,2.5) node[circle,minimum size=2cm,draw] (A) {\Large{$\mathrm{A_j}$}};
\draw (3,6) node[circle,minimum size=2cm,draw] (lam) {\Large{$\lambda$}};
\draw (4,0) node[rectangle,minimum width=2cm,minimum height=1cm,draw] (d)
{\large{\textbf{Data \{A, F\}}}};
% arrows
\draw[-triangle 45,thick] (p) -- (lik);
\draw[-triangle 45,thick] (A) -- (lik);
\draw[-triangle 45,thick] (lam) -- (A);
\draw[-triangle 45,thick,dashed] (d) -- (lik);
\end{tikzpicture}
```

Figure 3: LaTeX Source for Directed Graph

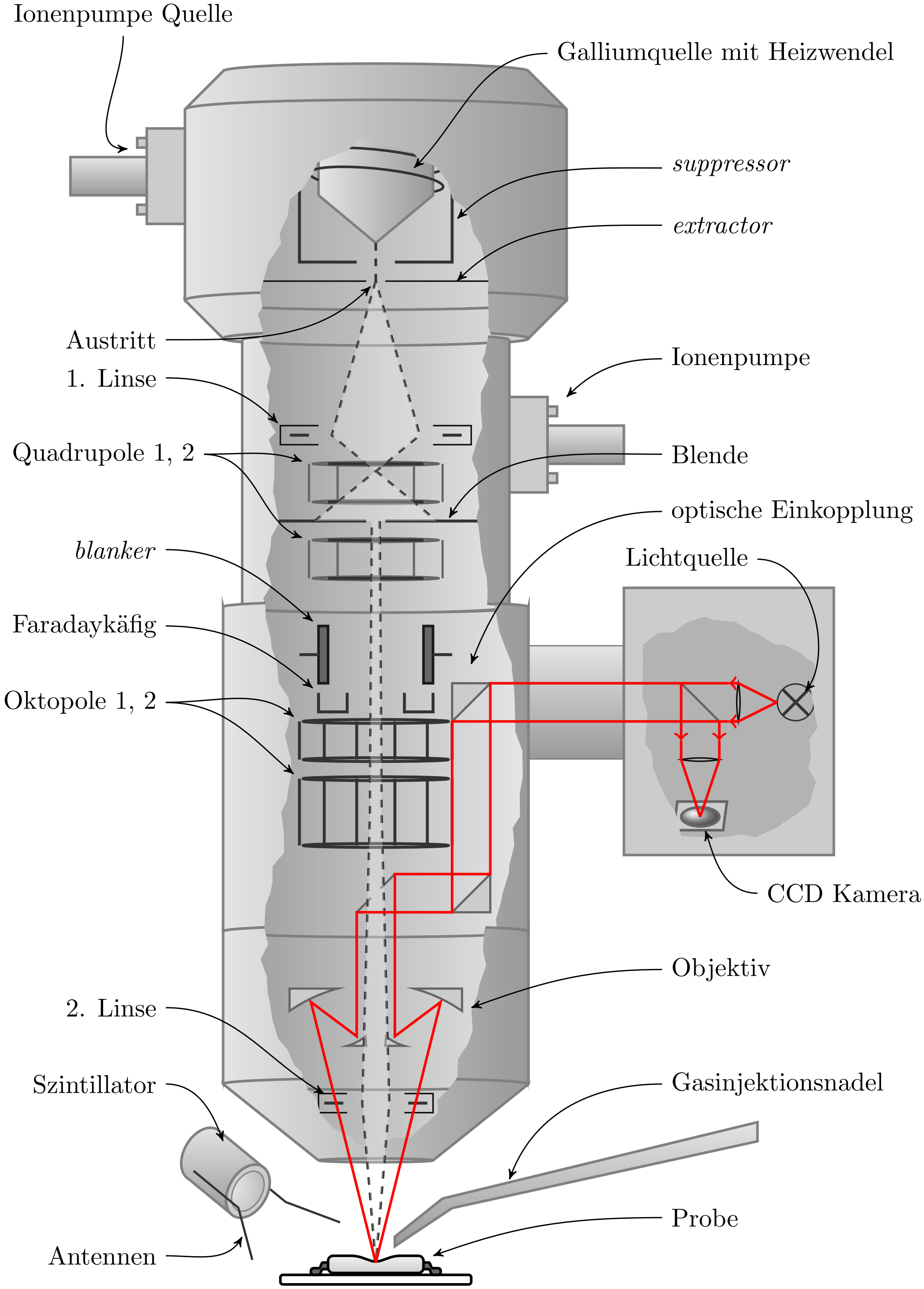


Figure 4: Another LaTeX Drawing

Up to now, the text, apart from links, has been in black-and-white but LaTeX can not only put things in color, you can even define your own colors. In fact, all of the link colors shown in this document were customized since defaults like red, green, etc. are too garish for this author. I chose to define all of my colors in standard RGB format. The source is as follows:

```
\usepackage{color}
\definecolor{Red90}{rgb}{0.9, 0.0, 0.0}
\definecolor{Green75}{rgb}{0.0, 0.75, 0.0}
\definecolor{Blue90}{rgb}{0.0, 0.0, 0.9}
\definecolor{Brown}{rgb}{0.63, 0.32, 0.18}
```

You can easily see the difference since this is in red while this is in my custom Red90 which is toned down by ten percent.

There is no point in going on and on regarding what you can do in LaTeX when it comes to document production. Suffice it to say that you can do whatever you want. Much of it was designed for technical documents but it is just as easy to generate a letter, a large poster or a set of slides, etc. Moreover, the "technical" content need not be mathematical. You might want to write a chemical reaction such as

$$^{1}_{0}\mathrm{n} + ^{14}_{7}\mathrm{N} \longrightarrow ^{14}_{6}\mathrm{C} + ^{1}_{1}\mathrm{H}$$

or simply note that, in June, 2021, £1 = \$1.41 = ¥155.54. These may be unusual but they are still familiar. More offbeat examples have been relegated to Appendix A.

4 Summary

In this document, we have tried to show why LaTeX is considered the state of the art when it comes to producing documents, especially technical documents. The primary reason, apart from the obvious quality, is that LaTeX is a language with capabilities limited only by the expertise of the writer. That said, it does not take much expertise since most of it has been deliberately designed to be easy to understand and flexible enough to enhance on those rare occasions when availability functionality does not suffice.

The fact that LaTeX is completely free is an obvious bonus. So, whether decisions regarding document preparation tools are made on the basis of functionality or cost, it would seem that there is no reason not to use the best. LaTeX is the best.

A Further Examples

The examples in this Appendix push the boundaries of what can readily be done in LaTeX. Some are extremely easy to do; others take more effort. All of them were done completely in LaTeX—no other software required.

We begin with an easy one-liner showing a stereo (3-D) representation of a cholesterol molecule.

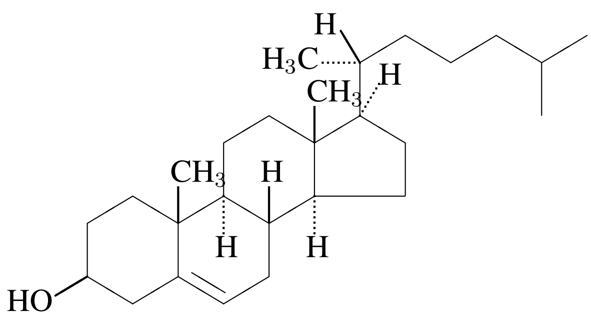


Figure 5: Structure of Cholesterol, $C_{27}H_{46}O$

While we are in a scientific mood, let's give physics equal time.

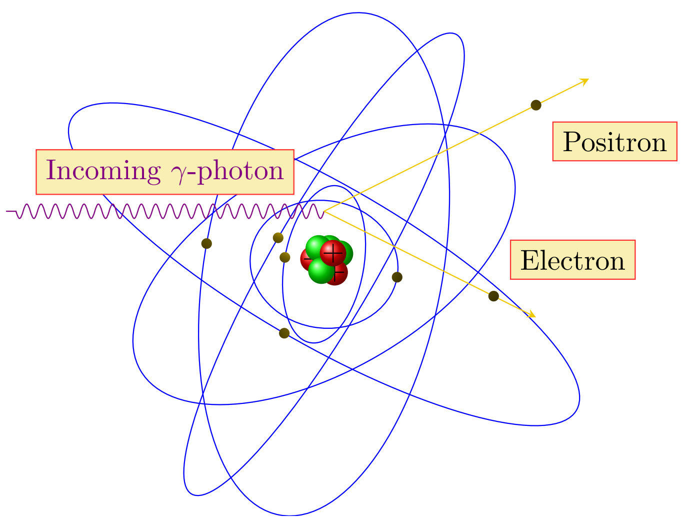


Figure 6: Pair Creation ($e^-$and e^+)

One of the hardest tasks, even with drawing software, is to produce a figure *exactly* to scale. With real numbers to six signifiant figures, LaTeX can do this easily (see Fig. 7).

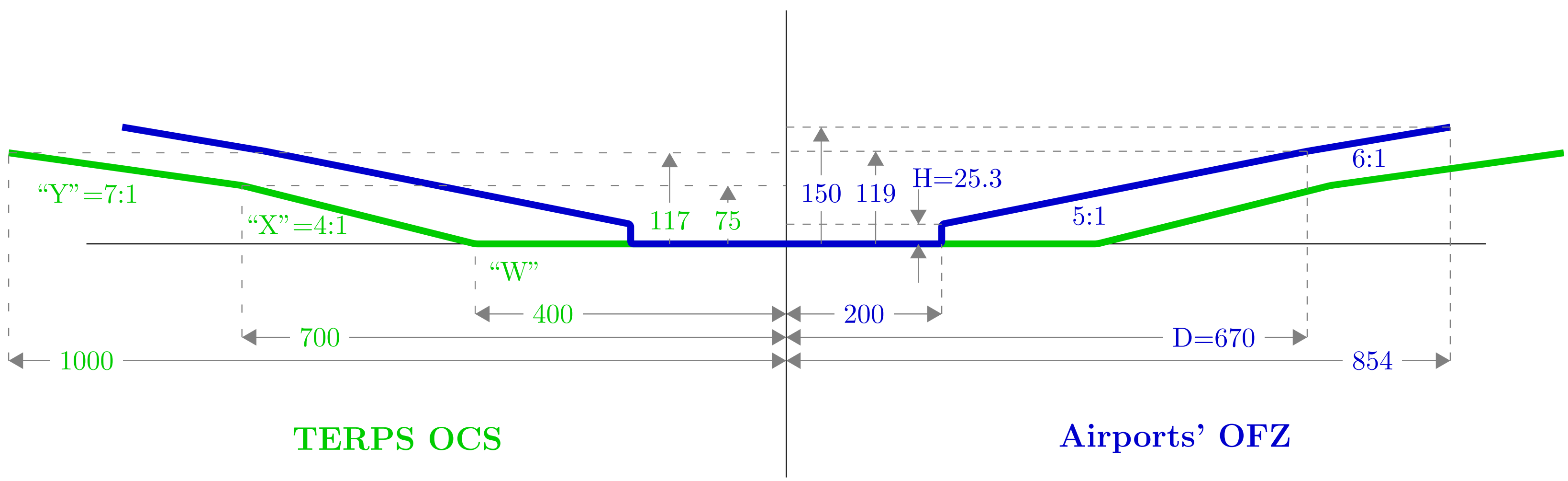


Figure 7: Airport Protection Surfaces

There are many other constructs besides drawings that might be needed. For instance, here is an algorithm.

Algorithm 1 Euclid's algorithm

```
procedure EUCLID(a, b)                          ▷ The g.c.d. of a and b
    r ← a mod b
    while r ≠ 0 do                              ▷ We have the answer if r is 0
        a ← b
        b ← r
        r ← a mod b
    end while
    return b                                    ▷ The gcd is b
end procedure
```

LaTeX knows all about PostScript™ so you can use some

PSTricks

if you want to.

We end with some external links, mostly just for fun. One of my favorites is this one.

In case you are not impressed yet, here is a link to some math notes and, for good measure, some other notes.

Finally, here is the link to the TeX Showcase where you can find a lot more examples. Is it any wonder that people have switched to LaTeX to write their documents?

References

[1] D. Knuth, *The TEXbook*. Addison-Wesley, 1994.

[2] L. Lamport, *LATEX – A Document Preparation System*. Addison-Wesley, 1994.

[3] H. Kopka and P. W. Daly, *Guide to LATEX*, 4th ed. Addison-Wesley, 2003.

[4] G. A. Grätzer, *More Math into LATEX*, 5th ed. Springer, 2016.

[5] F. Mittlebach *et al.*, *The LATEX Companion*, 2nd ed. Addison-Wesley Professional, 2004.

[6] M. Gossens *et al.*, *The LATEX Graphics Companion*, 2nd ed. Addison-Wesley Professional, 2007.

[7] H. Voss, *Typesetting Tables with LATEX*. UIT Cambridge Ltd., 2010.