The TeX Family in 2008

Jim Hefferon, Karl Berry

Mathematicians know TeX as a standard tool. Along with associated programs such as LaTeX and BibTeX it allows working professionals to produce documents of journal quality, without first having to become expert at the intricacies of typography.

We will highlight some of the exciting recent developments in the TeX family of programs. We will not aim the presentation towards TeX experts. Rather, as long as you are comfortable producing your own documents in TeX, or perhaps are a beginner interested in learning, you should find here some information that you can use. Our discussion will concentrate on things that are available today or are in a late stage of development.

You may know that TeX has been in some way frozen by its creator. How can there be big changes? Today's executable TeX programs can still give output conforming to the standard, and can still process documents made years ago. The freeze was done in a way that allows developers to adapt TeX to changes in the world of document production. In addition, people in the TeX community have developed other programs of interest, such as graphical front-ends to make writing a document easier.

Note. Plenty of additional information is available on all of the topics mentioned in this article. We've included a few links and references, but for all the pointers please see this article's web page, `http://tug.org/notices`. Having the links online has the advantages that they are clickable and that we can maintain them as they change over time, as links will do.

History

Many *Notices* readers know the story but some may not: TeX began when Donald Knuth of Stanford University received the galleys for an edition of one of the books in his magnum opus *The Art of Computer Programming* and was so discouraged by how they looked that he resolved to write a system to do the job correctly. In 1978, he delivered the Gibbs lecture at the AMS annual meeting outlining the basis of his work on both typography and fonts. This led the Society to adopt his as a standard format, which in turn led to its adoption by the mathematical community (and others), from large journals to individuals.

A major factor in that adoption was that Knuth made the program freely available, including its source code. Soon, an informal community of users arose, porting TeX to many platforms, developing

Theorem 1 (Central Limit Theorem)**.** *Let $\{\mathbf{X}_k\}$ be a sequence of mutually independent random variables with a common distribution. Suppose that $\mu = \mathbf{E}(\mathbf{X}_k)$ and $\sigma^2 = \operatorname{Var}(\mathbf{X}_k)$ exist and $\mathbf{S}_n = \mathbf{X}_1 + \cdots + \mathbf{X}_n$. Then for every fixed β*

$$\mathbf{P}\left\{\frac{\mathbf{S}_n - n\mu}{\sigma\sqrt{n}} < \beta\right\} \rightarrow \mathfrak{N}(\beta).$$

```
\documentclass{article}
\usepackage[paperwidth=2.75in,paperheight=1.5in,
   width=2.75in,height=1.5in,
   lmargin=0in,rmargin=0in]{geometry}
\usepackage{amssymb,amsmath,amsthm}
\DeclareMathOperator{\var}{Var}
\newtheorem{thm}{Theorem}
\newcommand{\rv}[1]{\mathbf{#1}}
\newcommand{\pr}[1]{\mathbf{#1}}
\newcommand{\ev}[1]{\mathbf{#1}}
\begin{document}
\begin{thm}[Central Limit Theorem]
Let $\{\rv{X}_k\}$ be a sequence of mutually
independent random variables with a common
distribution.
Suppose that $\mu=\ev{E}(\rv{X}_k)$ and
$\sigma^2=\var(\rv{X}_k)$ exist and
$\rv{S}_n=\rv{X}_1+\dots+\rv{X}_n$.
Then for every fixed $\beta$
\begin{equation*}
\pr{P}\left\lbrace
  \frac{\rv{S}_n-n\mu}{\sigma\sqrt{n}}<\beta
\right\rbrace \rightarrow \mathfrak{N}(\beta).
\end{equation*}\end{thm}
\end{document}
```

Figure 1. A sample of LaTeX output and the associated input (adapted from [6]**).**

ancillary programs, etc. Following Knuth, this community typically makes its work freely available. A formal infrastructure also appeared, the TeX Users Group. This group continues to be active, with annual meetings, journals, and funds dedicated to further development. Other user groups are now active worldwide, working in many languages in addition to English.

Another factor in TeX's adoption was that Knuth designed the system so that any knowledgeable user could create extensions. Leslie Lamport used this capability to produce the LaTeX format. (A format is the overall set of commands that an author uses to write documents.) LaTeX was a major departure from Knuth's original `plain` format, providing many high-level facilities that authors need: it supports producing articles, reports, and books, including chapters and sections. It allows for floating figures, and automatic generation or maintenance of cross-references, tables of contents, bibliographies, and indexes. It has a simple but powerful syntax for tables, and strong capabilities with graphics and color. It also emphasized a philosophy that authors should describe the logical role of text rather than focus on its appearance. This means that you might start a chapter by typing `\chapter{Introduction}`, rather than directly specifying that the name appear in a larger font surrounded by additional vertical space.

LaTeX itself can also be extended. Many people have contributed packages of materials that adjust LaTeX's default behavior. One example is the AMS's class *amsart* for articles. Another is the AMS's *amsmath* style, which adds many options for presenting equations, mathematical symbols, arrays, etc. There are many smaller examples, such as the *fancyhdr* style that easily adjusts page headings and footings. (A LaTeX package is a class if it controls whole documents by setting margins, headers, type size, etc. If it is focused on more local aspects, such as adding facilities for including computer code listings, then it is a style.)

Formats other than LaTeX exist. These include Eplain, which adds some basic authoring features to `plain`, and ConTeXt, which is a comprehensive, modern system. However, LaTeX remains the most popular by far, in part because it has such a large library of useful additions. Below we will focus on solutions based on LaTeX, although much of our discussion holds for any TeX document.

What TeX can do

The steps for producing a TeX document are well-known to most *Notices* readers. If you are not familiar: the author creates a source file containing a mixture of text and commands, somewhat similar to a document in the web page language HTML. The author then runs a program or sequence of programs to convert the source to typeset output. For instance, Figure 1 gives a sample of moderately complex output along with its complete input source. (*Aside:* This article is not about persuading you to adopt TeX but we cannot resist one suggestion: this input was done by a user who is not a wizard but is just reasonably competent. To judge the value added by TeX, we invite you to ask a reasonably-competent user of a word processing system to produce a version of this text, and then compare the two on input effort as well as output comprehensibility and appearance.)

Increased capabilities in the engines: pdfTEX, XƎTEX, and LuaTEX

The main executable TEX program that runs on your system is called the engine. The user community has ported the engine and other parts of the TEX suite to every computing platform that you are likely ever to use. In addition to porting it, developers have improved it. For instance, a modern TEX does not have the memory constraints of earlier systems. So the TEX engine that you are running is a descendant of Knuth's original.

The engine has changed in other ways. One has to do with output devices such as printers and computer screens. Rather than try to include in the engine the ability to work with every printer or screen on the market, Knuth designed the system to produce output in a format called DVI that is easily read by computers. Separate programs convert these DVI files for use on particular output devices. This approach is flexible but has the disadvantage that to share documents with colleagues, an author must distribute either the TEX source or the DVI file, and in either case the recipient must do further processing.

A step forward came when many printer vendors adopted the PostScript language for describing pages. Now an author could distribute work as a PostScript file and expect that colleagues could print or view it without further processing. Developers wrote a number of programs that convert DVI to PostScript; the most popular is `dvips` by Tomas Rokicki. It is still in widespread use today as part of the document production system of many individuals and publishers, including the production system at the AMS.

In recent years a descendant of the PostScript language, PDF, has become very popular. This language is designed for use on the Internet and web browsers easily display these files. So now authors have a practical way to distribute their work online while retaining their fonts and formatting, something that is especially important with mathematical documents (the web language HTML does not easily offer such control). In response to the emergence of PDF, which has become an open Internet standard, Hàn Thế Thành and others developed an extension of TEX's engine called pdfTEX. This directly outputs PDF so that the intermediate DVI step is no longer necessary. Many documents written with the original engine in mind can be output in PDF simply by processing it with the command `pdflatex myfile` instead of `latex myfile`.

Development of the engine continues. Another extended engine included in today's distributions is XƎTEX. This allows you to move away from TEX-specific fonts, to easily use any font installed in your system. XƎTEX also improves TEX's non-English language handling because it fully supports input in Unicode. (Unicode is a standard to provide a representation for every character in every human language. The AMS's Barbara Beeton, among many others, has been working to ensure that Unicode supports all mathematical symbols.)

On the horizon is LuaTEX, which connects the computer language Lua with the TEX engine. Since writing in TEX's standard extension language can be quite hard, while Lua is a scripting language, meaning that it is well-suited to this kind of work, this makes programming with TEX much easier. LuaTEX is being actively developed as of this writing.

Graphics

Knuth designed TEX to be able to use or import any sort of graphics.

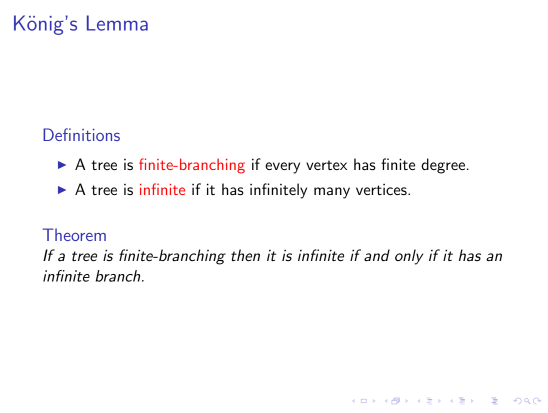
There are good ways to create graphics within a TEX source file, notably with the packages *PSTricks* and *TikZ*. This approach describes the figure with a graphics language. Another example of a graphics language that is good at making technical figures is MetaPost; with this language, the code is written outside of the TEX source.

Alternatively, you may prefer to produce graphics using a program such as Gnuplot, Excel, Matlab, or Illustrator. In addition, you may also want to use graphics that don't come from a program, such as photographs.

The typesetting engines in current distributions can incorporate all of the above. For instance, pdfTEX can directly import JPEG, PNG, and PDF format graphics. You can convert graphics in other formats to one of these three. An example is that you may have graphics in the EPS format (closely related to PostScript), and there are programs to convert EPS graphics to PDF; one that runs on all platforms is `epspdf`.

To insert and manipulate external graphics, LATEX provides the package *graphicx* that is both powerful and straightforward. As an example, the command `\includegraphics[width=16pc]{lions.jpg}` appears in the source of this article to include the graphic on the first page. That command causes the *graphicx* package to examine the graphic file for its natural width and height. The package then puts the graphic in the document and scales it to a width that works with the column size used by the *Notices* (16 `pc` is about 2.66 inches). This

package can also do simple manipulations: in this paragraph, the picture from the first page


König's Lemma

Definitions

- A tree is finite-branching if every vertex has finite degree.
- A tree is infinite if it has infinitely many vertices.

Theorem

If a tree is finite-branching then it is infinite if and only if it has an infinite branch.

```
\documentclass{beamer}
\usepackage{amsthm}
\theoremstyle{plain}
  \newtheorem{thm}{Theorem}
\theoremstyle{definition}
  \newtheorem{defs}{Definitions}

\begin{document}
\begin{frame}\frametitle{K\"onig's Lemma}
\begin{defs}
\begin{itemize}
  \item A tree is \alert{finite-branching} if
     every vertex has finite degree.
  \item A tree is \alert{infinite} if it has
     infinitely many vertices.
\end{itemize}
\end{defs}
\begin{thm}
  If a tree is finite-branching then it is infinite
  if and only if it has an infinite branch.
\end{thm}
\end{frame}
\end{document}
```

Figure 2. Using the beamer presentation package. The slide has been shrunk to fit this page (which is why the navigation elements in the lower right look tiny).

has been clipped and shrunken by giving the `\includegraphics` command the optional arguments `width=5pc`, `viewport=120 300 360 590`, and `clip`. (The viewport unit is 1/72 inch. Also, we have wrapped the text around the image by using the LaTeX add-on package *wrapfig*.) We recommend an article [2] by Klaus Höppner for background and tips on working with graphics.

Hypertext

Web links, technically called URL's, didn't exist when TeX was first developed. They are a typesetting challenge because they tend to be long and hard to break across lines. They are also a challenge because readers expect document links to be clickable and so the text in the source must be associated with the right target. The *url* package does a good job of handling the typesetting requirements. In particular, it handles the special characters that can appear in URL's, such as percent signs, that otherwise have a special meaning for TeX.

The LaTeX document style *hyperref* is even more ambitious: it tries to turn a paper-based document into a hyperlinked document without requiring any intervention by the author. For instance, suppose that you have an existing LaTeX document in which the input `\ref{th:LamesThm}` produces the output "Theorem 3." Adding `\usepackage{hyperref}` at the start of the document will produce that same text, but now it is also a link to the referenced spot. Similarly, entries in the table of contents become links to the chapters and sections, literal URL's become clickable, and so on.

Presentations

Another application that became widespread since TeX was developed is presentation software, of which Microsoft PowerPoint is an example.

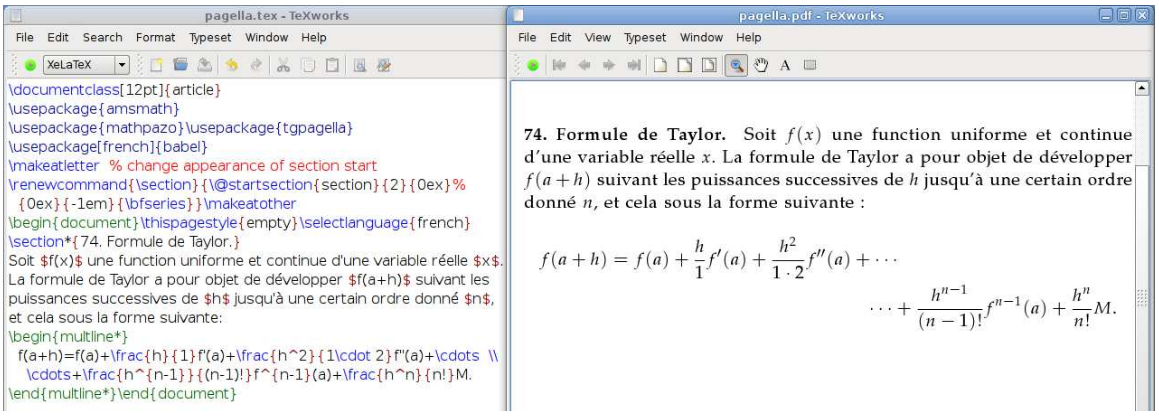
Several packages bring TeX's ability with mathematical text to making presentations. These change the page size and orientation, produce navigation elements such as buttons for changing pages, and perhaps have a screen area showing an outline of the talk. They also allow the slides appear in steps such as one bullet point at a time.

The *beamer* package is one of the most popular and well-documented; a sample is shown in Figure 2. The tutorial article by Andrew Mertz and William Slough [3] introduces it with a graduated sequence of examples.

Editing and running TeX: TeXworks

Many users compose their TeX documents inside of a graphical user interface. Such a front end typically incorporates an editor that is specialized for writing TeX, along with a way to invoke commands; for instance, it may have a single button to compile the document and displays the result. Everyone has a favorite. On Windows, both TeXnicCenter and WinEdt are among the most popular. In a Unix environment, including GNU/Linux, many people use the programming editor Emacs, perhaps with the AUCTeX add-on.

One front end that has made a big splash in recent years is TeXShop by Richard Koch and others, for the Macintosh. It is a clean working environment directed at an average user. There are many useful features, and an excellent help system (including



pagella.tex - TeXworks
File Edit Search Format Typeset Window Help
XeLaTeX
pagella.pdf - TeXworks
File Edit View Typeset Window Help

```
\documentclass[12pt]{article}
\usepackage{amsmath}
\usepackage{mathpazo}\usepackage{tgpagella}
\usepackage[french]{babel}
\makeatletter  % change appearance of section start
\renewcommand{\section}{\@startsection{section}{2}{0ex}%
  {0ex}{-1em}{\bfseries}}\makeatother
\begin{document}\thispagestyle{empty}\selectlanguage{french}
\section*{74. Formule de Taylor.}
Soit $f(x)$ une function uniforme et continue d'une variable réelle $x$.
La formule de Taylor a pour objet de développer $f(a+h)$ suivant les
puissances successives de $h$ jusqu'à une certain ordre donné $n$,
et cela sous la forme suivante:
\begin{multline*}
  f(a+h)=f(a)+\frac{h}{1}f'(a)+\frac{h^2}{1\cdot 2}f''(a)+\cdots  \\
   \cdots+\frac{h^{n-1}}{(n-1)!}f^{n-1}(a)+\frac{h^n}{n!}M.
\end{multline*}\end{document}
```

74. Formule de Taylor. Soit $f(x)$ une function uniforme et continue d'une variable réelle x. La formule de Taylor a pour objet de développer $f(a+h)$ suivant les puissances successives de h jusqu'à une certain ordre donné n, et cela sous la forme suivante :

$$f(a+h) = f(a) + \frac{h}{1}f'(a) + \frac{h^2}{1\cdot 2}f''(a) + \cdots$$
$$\cdots + \frac{h^{n-1}}{(n-1)!}f^{n-1}(a) + \frac{h^n}{n!}M.$$

Figure 3. TEXworks screenshot with the edit and previewer windows (text from [5]).

videos). But there is one feature that users report makes the biggest difference in their productivity: source/preview synchronization. (Jêrome Laurens wrote the foundation work for this feature.) It allows you to use a mouse shortcut to switch back and forth between the TEX source and the output preview. That is, clicking in the output PDF sends you to the corresponding spot in the TEX input and, conversely, clicking in the input sends you to the corresponding spot in the output. This works even with multifile LATEX documents joined by `\include`.

TEXShop is Macintosh-specific. So the TEX user groups have sponsored the development of a new front end called TEXworks (by Jonathan Kew, also the author of XETEX). This has the same "keep it simple" attitude as TEXShop and is available for all of today's major systems: Windows, GNU/Linux and other X11-based systems, and Mac OS X. The editor works in Unicode so non-English text is not an issue. It will color TEX commands differently from regular text for easier reading. The default output is PDF so that you can share your files online or in email with recipients who do not have TEX. Finally, it includes an integrated PDF viewer that supports source/preview synchronization, so it brings that advantage of TEXShop to the other major computing platforms.

Figure 3 shows an example of TEXworks in action on a GNU/Linux system (it will look different on other platforms because it uses the window drawing system of the platform). It continues to develop but is very usable now. The TEXworks home page is `http://tug.org/texworks`.

Fonts: Latin Modern, TEX Gyre, and STIX

Computer fonts come in two types: a bitmap font has each character specified by an array of pixels, while a vector font has each character given by a set of parametric curves. Bitmap fonts are simpler but vector fonts scale smoothly to different sizes. Because of this advantage, vector fonts have come to dominate practical typesetting, originally in the form of PostScript Type 1 and TrueType fonts.

The recently developed OpenType font format standard was created and is supported by the major font vendors. It combines and extends the capabilities of TrueType and Type 1. The description earlier of the XETEX engine hints at its capability — it allows you to easily use any OpenType font. This means, for instance, that you can use a font that came with your system, or download one, without having to prepare it for use in TEX.

However, if you want to use the font in a document that contains mathematics then you still must supply a large number of parameters (for example, the spacing needed to position superscripts and subscripts). So fonts that have been prepared for mathematical text remain unusual and of special interest to the TEX community. A number of alternatives to Knuth's original font (Computer Modern) are currently available [1]. Here we will mention three recently been developed vector font families that are well-suited for use with TEX documents. All three are freely available.

Latin Modern, by Bogusław Jackowski, Janusz M. Nowacki, and Marcin Wolínski of the Polish TEX users group, is based on Knuth's original Computer Modern fonts, but it is supplemented with a rich collection of diacritical marks such as accents or umlauts that can be added to letters, as well as letters that are precomposed with them. These supplements make for a better appearance and also improve hyphenation, since TEX's original mechanism for diacritical marks interferes with its hyphenation algorithm.

The TEX Gyre collection of fonts from the same group is based on the fonts commonly available in PostScript printers. Gyre adds to these fonts its own collection of diacritical marks for use in European languages. Figure 3 shows one of these, Gyre Pagella (comparable to the Palatino font; its use with mathematics is not finalized so this example uses the package *mathpazo* to bring in a closely related font for math).

Finally, the STIX fonts are being developed by several leading scientific and technical publishers, including the AMS, and will provide a comprehensive set of mathematical symbols. When released, the fonts will have the look of the familiar Times Roman.

Obtaining TEX

A downloadable working TEX installation is called a distribution. Today there are two maintained distributions that are Free Software — both are free of cost and contain only materials with license conditions that allow for free redistribution. (Proprietary TEX distributions are also available.) These two, MiKTEX and TEX Live, are discussed below. Both support installation over the Internet or from a DVD, and provide an install wizard to walk users through the steps. Both are large collections, with the programs, fonts, LATEX style files, etc., that users have come to expect. (To obtain these distributions, online or on DVD, see the `http://tug.org/notices` web page.)

MiKTEX is the most popular distribution for the Windows platform (it is also being ported to GNU/Linux systems). It is best known for its package management and update programs, which are mature and very capable. For example, it can detect when a document being generated requires a package that is not installed, and then download and install that package.

The other major distribution, TEX Live, also works on Windows but is most popular on Unix-like systems such as GNU/Linux. It has an active community of developers who have recently released a new package management and installation system.

Apple's Mac OS X is a Unix-like system, but for this platform we recommend the MacTEX distribution, which installs the complete TEX Live distribution along with a few extra Mac-specific features and programs.

In the TEX community

After successfully installing, what's next?

Most TEX users choose LATEX as the core format for their documents. If you are new to TEX, we recommend it. One free manual that is enough to get you well underway in writing LATEX is *The Not So Short Introduction to LATEX2e* [4], available in many translations. Many other excellent documents and books have been written over the years; we have links on this article's web page.

An advantage of using LATEX, and TEX in general, is that you are joining a community that has been active for a long time. Whatever issues you come across have very likely been tackled before. You can save yourself many struggles by looking for solutions that others have shared.

A great place to look for answers and to ask questions is the online discussion group `comp.text.tex`, which is one of the most active TEX forums. You can search through two decades of past questions. If you are still stuck then post one yourself.

The TEX community's largest resource for packages, programs, documentation, and more is the Comprehensive TEX Archive Network at `http://www.ctan.org`. This is a multi-gigabyte collection that contains far more than any TEX distribution. The solution to your TEX problem may have been already developed and made available here. Conversely, if you develop a package then please consider uploading it to CTAN. This not only gives your work greater visibility but it also gives it a stable web address for years to come.

Another way to contribute to the community is to join your local TEX user group — this helps support TEX development in all its facets. We provide a link on our web page.

Best practices with publishers

For mathematicians, one area of special interest is working with professional publishers. Here are five points that publishers have passed on to us.

The biggest one is communication: you should communicate with the publisher's technical staff early on, particularly if you will need special packages or fonts.

The second point is to use LATEX; every serious mathematics publisher can use it. Some can use other formats, but don't count on it.

The third is to make sure that your source files are portable. The production staff at your publisher will greatly appreciate this. In addition, you may want to resubmit your work to a second publisher after a rejection, or otherwise reuse your mathematical work in different settings: books, class notes, grant applications, and so on. Having a portable source means that it will adapt to new situations with less effort.

How can you make your documents portable? The most important way is to take advantage of the structure inherent in LATEX. Use `\chapter` and `\section` and so on, rather than defining your own structure. In addition to portability, this makes your source file more organized. Be aware that if you define special theorem environments, or other special formatting commands, then your publisher may also define those and yours may be overridden. (Never redefine basic elements of TEX or LATEX such as `\par`.) In general, a light hand at doing your own formatting is best; accept the defaults.

In contrast with specifying your own commands for detailed spacing and fonts, a good portability practice is to create shorthands for frequent but cumbersome bits of code. For instance, in Figure 1 the random variables are not entered as `\mathbf{X}` but rather as `\rv{X}`, where `\rv` has been defined appropriately.

The fourth point is to be careful with graphics. Use standard programs and packages to create and include them. Find out what formats your publisher prefers for including graphics. (The AMS usually wants EPS files.) Be cautious about rescaling the size of the graphic via LaTeX commands since this can blur differences in line thicknesses.

And, the fifth point, if you are writing a book then decide at the beginning whether you will have an index. If so, then use LaTeX's indexing commands right from the start. Indexing is a painstaking job. If you do it as you compose the work then it is less tedious and the result will probably be more comprehensive.

Finally, you may find useful the AMS's list of frequently asked questions for authors: `http://www.ams.org/authors/author-faq.html`.

Closing

Despite its age — ancient in computer years — but because of its capabilities, TeX remains a standard. This includes publication platforms that didn't exist when TeX was written, such as the online preprint archive `arXiv`.

In recent years TeX has evolved rapidly, driven by the emergence of clear standards and by the effort of a development community that consciously keeps users in mind. The worldwide TeX user groups provide a framework and sponsorship for the activities.

We hope that you will find that taking advantage of these innovations helps you to be more productive.

References

[1] S. Hartke, A survey of free math fonts for TeX and LaTeX, `http://ctan.org/tex-archive/info/Free_Math_Font_Survey`.
[2] K. Höppner, Strategies for including graphics in LaTeX documents, *TUGboat* 26:1, `http://tug.org/TUGboat/Articles/tb26-1/hoeppner.pdf`.
[3] A. Mertz and W. Slough, Beamer by example, *TUGboat* 26:1, `http://tug.org/TUGboat/Articles/tb26-1/mertz.pdf`.
[4] T. Oetiker, et al., The not so short introduction to LaTeX2e, `http://ctan.org/tex-archive/info/lshort`.
[5] C. de la Vallée Poussin, Cours d'analyse, *Dover*, 1938.
[6] W. Feller, Probability, *Wiley*, 1950.