GEOGEBRA AS A FRONTEND TO GENERATING GRAPHICS FOR LaTeX

James T. Quinlan

Department of Mathematical Sciences, University of New England

Abstract: GeoGebra can be used to create graphics which then can be exported for inclusion in LaTeX documents for presentations, research articles, or classroom materials such as quizzes, exams, and homework. Graphics can be exported in various image formats (e.g., png) or by using the combination *PGF/TikZ* package. This article aims to demonstrates how to export graphics created in GeoGebra and import these graphics into a LaTeX document.

Keywords: LaTeX , PGF/TikZ, Mathematical Graphics, Mathematical communication

1. INTRODUCTION

When doing mathematics we must confront how we are going to communicate our ideas (Meier & Rishel, 1998). LaTeX[1] is a prevalent option in communicating mathematically as it produces high quality documents (Knuth & Bibby, 1986) and is an essential component in a mathematician's toolbox (Grätzer, 2007). Graphics to be included in a LaTeX document are either written or created using software. While there are many software choices to generate graphics (e.g., InkScape), (TeXample, 2013), GeoGebra is one option to generate and export graphics for LaTeX documents.

GeoGebra can be used as a front-end interface to circumvent some of the difficulties with writing (e.g., TiKZ) code syntax of LaTeX markup of even the most common graphical objects, (e.g., triangles, circles, graphs of functions, Venn Diagrams, etc.). Nearly all mathematical content areas including algebra, calculus, linear algebra, abstract algebra, statistics, and graph theory, to name a few, can utilize GeoGebra to create its content specific graphical objects. In particular, GeoGebra can be used to create graphics for use in classroom materials (handouts, quizzes, exams), beamer presentations, and research articles.

The purpose of this paper is to demonstrate how GeoGebra is used to export graphics for LaTeX documents. In this paper we examine exporting GeoGebra graphics to (1) an image file (either raster or vector graphic) and (2) PGF/TikZ (vector code). Examples covering various mathematical topics will be used.

2. GRAPHICS IN LaTeX

A *minimal* LaTeX document will be used as a "starting point" for all examples throughout this article. The (`minimal`) document class can be replaced by any one of several types as needed: `article`, `book`, `report`, `proc`, etc., including the `beamer` presentation class. The `minimal` LaTeX document follows.

```
\documentclass{minimal}
\begin{document}
     image/code from GeoGebra will go here
\end{document}
```

2.1. Include Graphic Images

A standard way to include a graphic in a document is to use an digital image such as a *.png*, *.jpg*, or *.eps*. The basic syntax for including graphics in a LaTeX document is to use the `includegraphics` command.

```
\includegraphics[scale]{file}
```

The file must be located in the same folder as the source. That is, the file is relative to the document. If image file is located in a different location, use path/filename.

```
\includegraphics[scale]{path/file}
```

Using the minimal document template we insert the `\includegraphics[scale]{file}` between the `\begin{document}` and `\end{document}` commands. The general document (including syntax) is contained in the following codebox. For an example using a Venn Diagram, see Example 1.

```
\documentclass{minimal}
\begin{document}
       \includegraphics[scale]{file}
\end{document}
```

[1] For resources, documentation, and downloads visit online TeX Users Group (http://www.tug.org)

2.2. Include TiKZ Code Generated from GeoGebra

Using GeoGebra to generate PGF/TiKZ code is a valuable alternative to exporting the graphics as an image. The PGF/TiKZ code generated by GeoGebra (see Figure 4) can be employed by three methods:

1. **Save As**; open document in LaTeX;
2. Select the **Copy to Clipboard** button, open a new blank LaTeX document, then *Paste*;
3. Select/Highlight (needed code), Copy, and Paste in existing document (e.g., the (`minimal`) document given above).

It is unlikely that a complete document will contain a single graphic, therefore Item #3 is the most practical in the sense that typically you will need to include a graphic in an existing document and will use GeoGebra to create the graphic.

```
\documentclass{minimal}

% Add these packages
     \usepackage{pgf,tikz}
     \usetikzlibrary{arrows}

\begin{document}
     code from GeoGebra will go here
\end{document}
```

3. EXPORTING GRAPHICS FROM GEOGEBRA

Results from the GeoGebra graphics windows can be exported and used in LaTeX documents including most graphical objects, control objects (e.g., checkboxes, buttons, sliders), text, and formula, method depended. In this section we will consider two methods of exporting graphics that can be subsequently used in LaTeX documents: (1) export to an image file and (2) export to the PGF/TikZ code.

3.1. Export to Image

There are two types of graphics: raster and vector. Raster images include JPG *.jpg*, GIF *.gif*, PNG (*.png*), and BMP *.bmp*. A common mistake when using rasterized images is an attempt to scale the image larger. For example, a image that is 300px × 400px cannot be upsized to 600px × 800px while preserving image quality, the image will look “pixelated". Another common mistake when working with graphics is non-proportional resizing. For example, resizing a 300px × 400px image to a 200px × 300px image. Although downsizing will not cause the image to be fuzzy in the same manner upsizing causes, the image’s height to width ratio will be out of proportion, thus look “odd".

One solution to upsizing is to use vector graphic formats (*.eps*, *.pdf*, & *.svg*). See Appendix for definitions of file formats. LaTeX does not support including *.svg* files directly. Include only PostScript images (*.eps*) if your goal is a PostScript document. Use *.pdf*, *.png*, *.jpg* and *.gif* images if your goal is a PDF document using pdflatex, TeXShop, or other PDF-oriented compiler.

The compiler pdflatex (Unix) and TeXShop (Macintosh) convert LaTeX source directly to *.pdf*, and do not accept PostScript images. Instead, they take *.pdf* images, as well as bitmap pictures in *.png* or *.jpg* or *.gif* format. So to use *pdflatex*, you must convert any PostScript images to one of these other forms. Conversion from *.eps* to *.pdf* can be accomplished during compilation of *pdflatex* by using the directive, `\usepackage{epstopdf}` in the preamble.

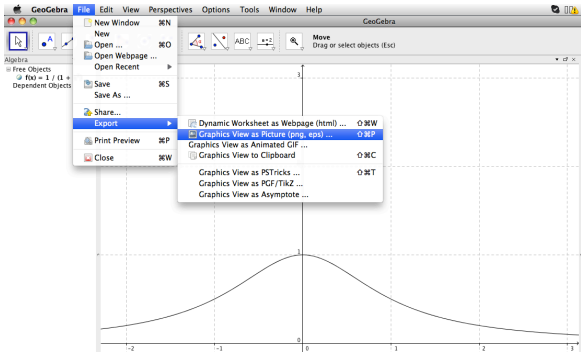

Fig 1: Export to Image Menu Option

Export to Picture dialog box contains options including file type, scale, and resolution (dpi); these options depend on the file type.

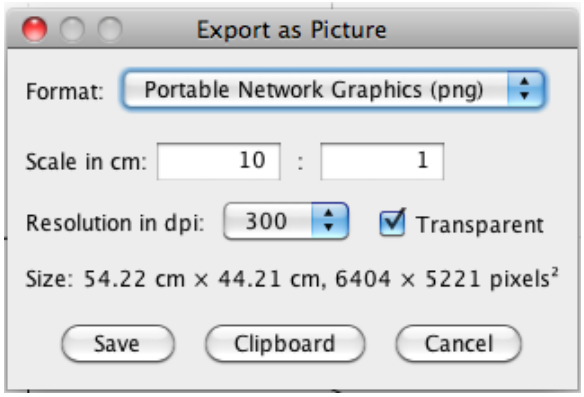

Fig 2: Export as Picture Dialog box

Steps to Export to Picture

1. Produce graphics in GeoGebra, including text labels;
2. Export to Picture:
   File → Graphics View a Picture (png, eps)...;
3. Select settings in dialog box (e.g., height and width)
4. Click `Save` button. Save to directory with LaTeX source.

3.2. Export to PGF/TikZ

The Portable Graphics Format (or PGF) is a package of TeX commands for producing inline vector graphics

while TikZ is a frontend to PGF that contain "higher" level macros. Unfortunately even with the TikZ frontend that makes using PGF easier, it is still difficult to use and has a steep learning curve. GeoGebra solves this problem and serves as a kind of WYSIWYG to creating vector graphics using PGF/TikZ.

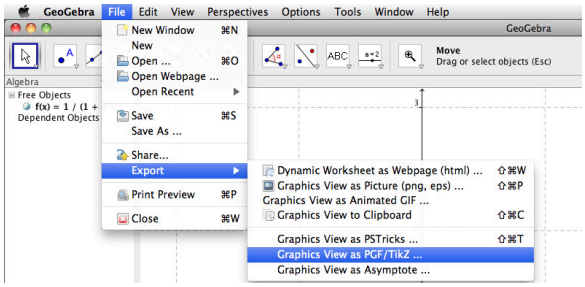

Fig 3: Export to PGF / TiKz: *File* → *Export* → *Graphics View as PGF/TiKz*

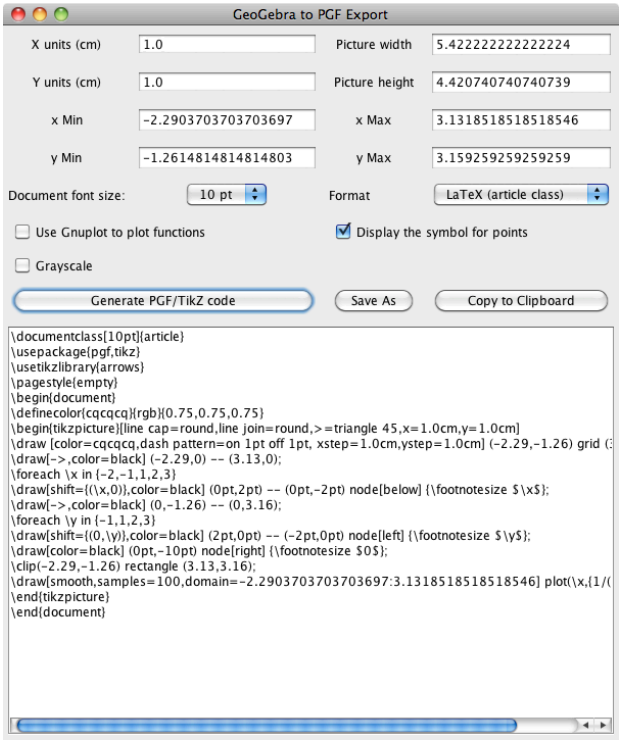

Fig 4: GeoGebra to PGF Export dialog options box

Steps to Export PGF/TiKZ

1. Produce graphics in GeoGebra, including text labels;
2. Export to PGF/TikZ:
   File → *Graphics View a PGF/TikZ*;
3. Select settings in dialog box (e.g., height and width)
4. Click `Generate Code` button
5. *Copy to Clipboard*, open a new LaTeX document and paste. However, in most cases, you will only copy and paste the code needed beginning with `\begin{tikzpicture}` and ending with `\end{tikzpicture}`. You will also need to copy and paste any any libraries (e.g., *arrows*) and defined colors (e.g., `{rgb}{0.7,0.7,0.7}`) to include in the document. This applies only when these lines of code have not previously been selected. In many cases, if this is any graphic to be included after the first graphic, often these libraries have already been included in previous export.

4. EXAMPLES

This section provides an example of an image created in GeoGebra, exported to a PDF format then including it in LaTeX and three examples of graphics created in GeoGebra then exported to PGF/TiKZ. These examples come from a diverse set of courses: set theory, calculus/statistics, linear algebra, and graph theory.

4.1. Example using an Image

Perhaps the simplest and most direct way to obtain mathematical graphics (e.g., Venn Diagram) for inclusion in LaTeX documents is to generate the graphic using GeoGebra, then export to a standard image format (e.g., *.png*, *.pdf*).

The basic steps to incorporate a Venn diagram in a LaTeX document are to (1) create the diagram, (2) export to an image for inclusion using the menus *File*, *Export*, etc. (see Example 1); a dialog box appears with a few options (such as image format type, resolution, and size); (3) save file in desired directory; and (4) include graphic in LaTeX document using the `includegraphics` command.

Example 1.
Create a Venn Diagram in GeoGebra (see Fig. 5). From the **File** menu, select **Export** → **Graphics View as Picture (png, eps)...** (see Fig. 2). The *Export as Picture* dialog box appears and defaults to png (see Fig. 1). Use the drop down menu and select the file format best for your compiler (e.g., pdflatex) and your needs. PDF and EPS are vector graphics and can be easily scaled, however EPS files need to be converted during compilation to PDF using `epstopdf` package depending on TeX compiler. Note the use of LaTeXin GeoGebra during creation of the image (i.e., $A \cap B$).

Save image as `venndiagram` in the same directory as the LaTeX document. Use the `\includegraphics` command to insert image in document. See Figure 5.

```
\documentclass{minimal}
\begin{document}
      \includegraphics[scale=0.4]{venndiagram}
\end{document}
```

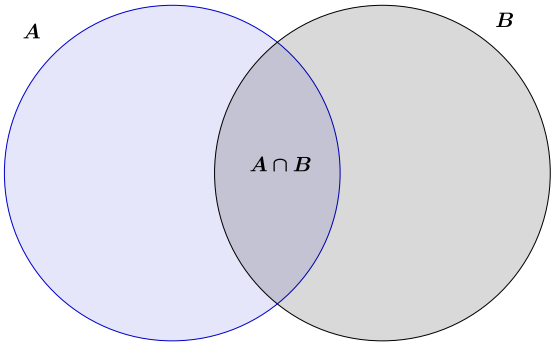

Fig 5: Venn Diagram created in GeoGebra exported to PDF

4.2. Examples using TiKZ

The next examples use GeoGebra to export to PGF/TiKZ. Although the topics to use as examples are many as any graphics created in GeoGebra can be exported to support nearly all content areas (e.g., triangles/geometry, functions/calculus, etc.), we give three examples: (1) a standard graph of a function found in calculus or statistics, (2) least squares approximation from linear algebra, and (3) a complete graph on 5 vertices found in an introductory graph theory course.

Example 2.
Let $f(x) = e^{-x^2}$. Create graph in GeoGebra as usual by entering the following command into the input bar.

```
f(x) = exp(-x^2)
```

The resulting graph is seen in Figure 6 and the GeoGebra generated TiKZ code is given below.

```
\documentclass[10pt]{article}
 \usepackage{pgf,tikz}
 \usetikzlibrary{arrows}
 \pagestyle{empty}
 \begin{document}
   \definecolor{cqcqcq}{rgb}{0.75,0.75,0.75}
   \begin{tikzpicture}
[line cap=round,line join=round,>=triangle 45,
        x=2.0cm,y=2.0cm]
\draw[->,color=black] (-2.13,0) -- (2.15,0);
\foreach \x in {-2,-1,1,2}
\draw
[shift={(\x,0)},color=black] (0pt,2pt)--(0pt,-2pt)
        node[below]
         {\footnotesize $\x$};
\draw[->,color=black] (0,-0.57) -- (0,1.28);
\foreach \y in {,1}
\draw
[shift={(0,\y)},color=black] (2pt,0pt)--(-2pt,0pt)
        node[left]
        {\footnotesize $\y$};
\draw
[color=black] (0pt,-10pt)
        node[right]
        {\footnotesize $0$};
\clip(-2.13,-0.57) rectangle (2.15,1.28);
\draw
[smooth,samples=100,domain=-2.12513:2.15497]
        plot(\x,{exp(0-(\x)^2)});
\end{tikzpicture}
\end{document}
```

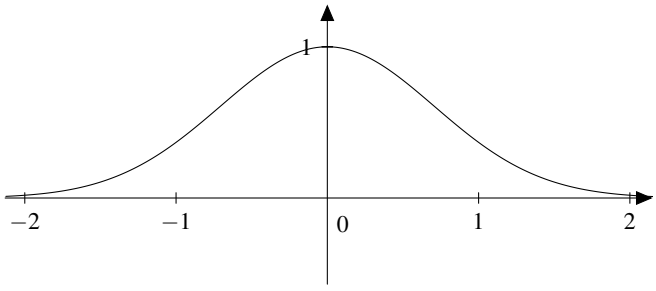

Fig 6: e^{-x^2} generated using GeoGebra (PGF/TiKz)

Example 3.
GeoGebra can be used to create a graphic to demonstrate the concept of least squares approximation. The graphic and code follow.

```
\documentclass{minimal}
      \usepackage{pgf,tikz}
      \usetikzlibrary{arrows}
\begin{document}
 \begin{center}
 \definecolor{zzzzzz}{rgb}{0.6,0.6,0.6}
 \definecolor{zzttqq}{rgb}{0.6,0.2,0}
 \begin{tikzpicture}[line cap=round,
  line join=round,>=triangle 45,x=0.7cm,y=0.7cm]
 \clip(7,1.28) rectangle (19.44,6.64);
 \fill[color=zzttqq,fill=zzttqq,fill opacity=0.1]
(7.9,3.0)--(15.9,2.0)--(18.5,3.3)--(13.1,4.0)--cycle;
```

```
 \draw [->] (8,3.02)--(15.68,3.1);
 \draw (15.84,3.58)
  node[anchor=north west] {$\mathbf{ y^*}$};
 \draw [->] (8,3.02)--(15.66,5.88);
 \draw (15.64,6.5)
  node[anchor=north west] {$\mathbf{ x }$};
 \draw (17.46,4.04)
  node[anchor=north west] {$\mathbf{ M }$};
 \draw [dotted] (15.66,5.88)--(15.68,3.1);
 \draw (7.58,3.46)
  node[anchor=north west] {$\mathbf{ 0 }$};
 \draw [->,color=zzzzzz]
  (8,3.02)--(13.26,2.36);
 \draw [->,color=zzzzzz]
  (8,3.02)--(11.26,3.66);
 \draw [line width=0.4pt,dotted]
  (11.26,3.66)--(15.68,3.12);
 \draw [line width=0.4pt,dotted]
  (15.68,3.12)--(13.26,2.36);
 \draw (11.04,4.38)
 node[anchor=north west]{$\langle x,e_2\rangle e_2$};
 \draw (12.76,2.44)
 node[anchor=north west]
 {$ \langle x, e_1 \rangle e_1 $};
 \begin{scriptsize}
 \fill [color=black] (8,3.02) circle (1.5pt);
 \end{scriptsize}
 \end{tikzpicture}
 \end{center}
\end{document}
```

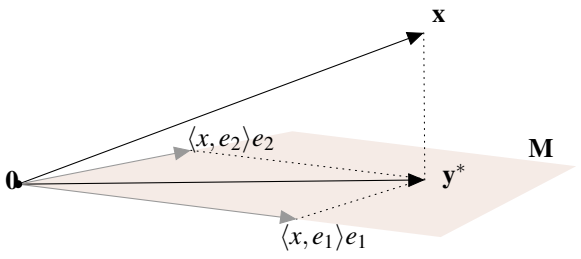

Fig 7: Least Squares approximation y^* of x.

Example 4.
GeoGebra used to generate a complete graph on 5 vertices (i.e., K_5).

```
\documentclass{minimal}
 % Add these packages
      \usepackage{pgf,tikz}
      \usetikzlibrary{arrows}
  \begin{document}
      \begin{center}
\begin{tikzpicture}[line
cap=round,line join=round,>=triangle
45,x=2.0cm,y=2.0cm]
\clip(4.84,2.76) rectangle (9.25,6.33);
\draw (6,3)--(8,3);
\draw (8,3)--(8.62,4.9);
\draw (8.62,4.9)--(7,6.08);
\draw (7,6.08)--(5.38,4.9);
\draw (5.38,4.9)--(6,3);
\draw (7,6.08)--(6,3);
\draw (7,6.08)--(8,3);
\draw (5.38,4.9)--(8.62,4.9);
\draw (5.38,4.9)--(8,3);
\draw (8.62,4.9)--(6,3);
\begin{scriptsize}
\fill [color=black] (6,3) circle (2.0pt);
\draw[color=black] (5.93,2.95) node {d};
\fill [color=black] (8,3) circle (2.0pt);
\draw[color=black] (8.08,2.98) node {c};
\fill [color=black] (8.62,4.9) circle (2.0pt);
\draw[color=black] (8.71,4.95) node {b};
\fill [color=black] (7,6.08) circle (2.0pt);
\draw[color=black] (7,6.25) node {a};
\fill [color=black] (5.38,4.9) circle (2.0pt);
\draw[color=black] (5.28,4.94) node {e};
\end{scriptsize}
\end{tikzpicture}
\end{center}
 \end{document}
```

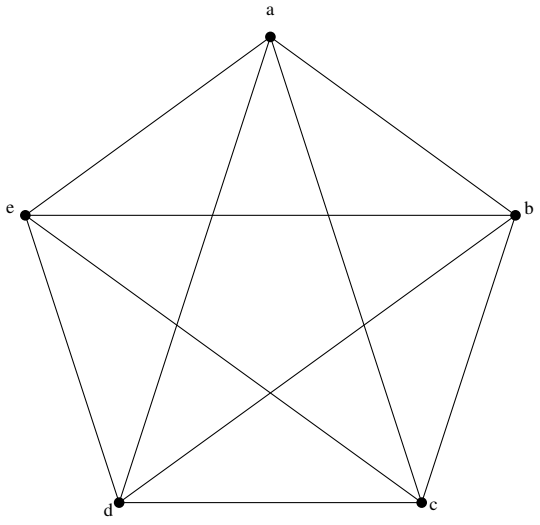

Fig 8: K_5 generated using GeoGebra (PGF/TiKz)

5. CONCLUSIONS

In addition to utilizing LaTeX for making a high quality interactive demonstration, GeoGebra can be used as an front-end interface for creating graphics for use in LaTeX documents. In this paper two techniques of using GeoGebra to support graphics for use in LaTeX documents were discussed. These two methods, (1) export the graphics to either a particular image format (e.g., .jpg, .eps, .pdf) or (2) using the PGF/TikZ package, are straight foward and simple. Because GeoGebra can be used to create graphical representations that can be used for algebra, geometry, and calculus, as well as illustrating several concepts of linear algebra, set theory, and graph theory, it is a robust tool for mathematics teaching and learning as well as creating representations and visualizations for mathematical research

papers.

Any graphical representation created in GeoGebra can be exported as an image and then used in various documents including LaTeX, MS Word, and HTML. Exporting graphics to a vector format (e.g., *.eps*, *.pdf*) is the preferred image format to preserve resolution and combat scalability issues.

PGF/TikZ packages have greater capabilities than what GeoGebra can produce, but GeoGebra provides a user-friendly means to create many graphics to embed in LaTeX documents. (In other words, GeoGebra can not be used to do everything that can be done with PGF/TikZ, however, it provides a user-friendly means to create many graphics and can be a real time saver.

When using GeoGebra to produce PGF/TikZ code, there are a few technically issues that can arise. For example, (1) in some cases, LaTeX in GeoGebra sometimes will be exported with double dollar signs ($$). In particular, $\langle x, e_2\rangle e_2$ in the least squares approximation (See Example 3), was exported as `$$ \langle x, e_2\rangle e_2$$`. It is helpful to know LaTeX and PGF/TikZ in order to correct these minor issues.

APPENDIX – ACRONYMS DEFINITIONS

Table 1: Graphic file type; acronym

File	Name	Type
PNG	Portable Network Graphic	Raster
JPG	Joint Photographers Group	Raster
GIF	Graphic Interchange Format	Raster
SVG	Scalable Vector Graphic	Vector
PDF	Portable Document Format	Vector
ESP	Encapsulated Postscript	Vector

REFERENCES

Grätzer, G. (2007). *More math into latex*. Springer.

Knuth, D., & Bibby, D. (1986). *The texbook* (Vol. 1993). Addison-Wesley.

Meier, J., & Rishel, T. (1998). *Writing in the teaching and learning of mathematics* (No. 48). Mathematical Association of America.

TeXample. (2013). *Tikz and pgf resources: A growing collection of links to various tikz and pgf resources.* Retrieved from `http://www.texample.net/tikz/resources/`