Management Center Innsbruck

Hebdomon Report and Documentation Standard

v0.1.1 - Angry Avocado

Student Name	D. T. McGuiness	**E-mail**	dtm@mci4me.at
Student Number	21001	**Semester**	5
Module	Robotic & Vision	**Degree**	B.Sc

Current version is v0.1.1 - Angry Avocado [WS2024]

The cover artwork is based from the code written by Nicolas P. Rougier, published in Github sujected to the BSD License.

This work has the LPPL maintenance status maintained.

The Current Maintainer of this work is D. T. McGuiness.

Declaration of Original Work

I affirm that all work in this document is my original work.
Further, I confirm all writing is my own writing and work from others has been cited appropriately.

D. T. McGuiness August 21, 2024

Contents

Chapter 1

The Hebdomon Template

Contents

1.1 Introduction

This document is designed to create an abstraction layer for the LaTeX and TeX commands and MACROs to hide away the programming aspect (i.e., command declarations) and ease into LaTeX programming as a means of introduction.
While some commands are have been changed NO command (or primitive) has been overwritten so feel free to use the original ones if you wish.

> As to the best of the authors knowledge, the original commands should work but the aesthetics may be compromised.

1.2 Title Formatting

To begin a new content, always start the content with a chapter heading. To insert the chapter with an additional `minitoc` use `\Chapter` command, which produces an heading as you see at the beginning of this page.

To have a heading without a `minitoc` use the standard `\chapter`.

The document relies on the user to use the correct title commands to keep the formatting consistent. The commands that starting with a capital letter are the overloaded commands of the standard ones. The following are the current ones in this version:

Types of Headers latex

```latex
\Chapter{...} % Allows entering chapters with mini table of contents
\Section{...} % Allows section titles with blue tint
\Subsection{...} % Allows subsection titles with blue tint
\Subsubsection{...} % Allows subsubsection titles with blue tint
```

> Unless there is a specific reason, it is suggested to use the aforementioned commands. However, original LaTeX command should work as well if you want to use.

1.2.1 Page Geometry

The page geometry is set to the following settings. This is done using the standard package `\usepackage{geometry}` which is defined in the `Hebdomon.cls`.

top 2.5cm,

right 2.0cm,

bottom 2.5cm,

left 3.0cm.

1.3 Image Positioning

The image positioning could be done with the following code snippet:

A Figure Environment latex

```latex
\begin{figure}[ht]
    \centering
    \includegraphics[options]{path.pdf}
    \caption{}\label{fig:...}
\end{figure}
```

Here there are a few options worth mentioning:

`path.pdf` The place where the image is kept. If the image is in the same folder where the `main.tex` file resides, it is as simple as writing the files name. If the file (i.e., `innsbruck.jpg`) is in a folder called `image`, just write `image/innsbruck.jpg`.

Finally if the image is in a higher directory (i.e., image is in `folderA/innsbruck.jpg` and the main tex is in `folderA/document/main.tex` then path becomes `../innsbruck.jpg`

`\caption{..}` Where you write the caption of the image. For consistency make sure every image has a caption. If the image does not need a caption, maybe it should not be present in the document to begin with.

`label{}` This is an identifier for you to use when you need to cite this Figure in a place somewhere. For example if you were to have and image with the following:

A filled Figure environment — latex

```latex
\begin{figure}[ht]
    \centering
    \includegraphics[width=\textwidth]{figures/path.pdf}
    \caption{A photo I found on the web.}\label{fig:innsbruck}
\end{figure}
```

Now this image is referenced as `fig:innsbruck`, which means if we write the following:

Referencing an Image — text

```text
To see the image, have a look at Figure \ref{fig:innsbruck}
```

This line of command will be presented as

> To see the image, have a look at Figure 1.1.

Finally you can see the image here as well.

Figure 1.1: The famous Innsbruck houses near the river Inn. This image is placed with a width value of `width=\linewidth`. This is also a good opportunity to showcase the hanging behaviour of the figure caption.

1.4 Defined Environments

`Excerpt` The template relies on the excellent `tcolorbox` package for formatting the boxes within the document and for that end different styles were created.

Sometimes one needs to quote either a proverb, for this use the `Excerpt` environment with the following notation and effect.

An Excerpt Snippet — latex

```latex
\begin{Excerpt}
    To be, or not to be...
\end{Excerpt}
```

Compiling this code snippet would show as in the document

> To be, or not to be...

`Code` During the preparation of your document, it is useful to showcase your some either as a snippet or in its entirety.

There are two (2) ways of doing this where the first one will be discussed here.

For example to print out a hello world in python, please use the following environment

```
\begin{Code}{python}{Hello, World!}
print("Hello, World!")
\end{Code}
```

Producing the following:

Hello, World! — python

```python
print("Hello, World!")
```

`minted` supports more than 500+ styles currently which you can have a look at their website.

`Example` Sometimes you need to showcase an example or need to highlight a certain idea. For these things the environment Example could be useful.

For example to show as simple example or give a slight attention to a topic you can do the following.

> This is an example. This could be anything which you would like to have a certain amount of attention but not too much as to distract from the flow of the document.

`Highlight` Or sometimes you need to give a clear break to the flow of the document and ask the reader to look at your banner. For that use highlight.

> Hey! Pay attention as this is a highlight box.

`Hgitemize` Similar to the itemize environment with the only modification being an added indentation.

1.5 Defined Commands

The following are the commands used in the document:

`hlight` Highlighting text is very easy, here is an example on how to write one.

Highlighting Text — latex

```latex
Highlighting text is \hlight{very easy}, here is an example:
```

`pcode` Allows you to enter code snippets inline such as `this` example.

Inline Code Snippets — latex

```latex
Allows you to enter code snippets inline such as \pcode{this} example.
```

1.5.1 Writing Equations

One of the strong suits of LaTeX compared to other editors and programs is it simplicity and ease of use methods of writing equations. Consider the following equation:

$$f(x) = x^2 + 2x + 1$$

In code form this would be written as:

Entering an equation without a reference — latex

```latex
\begin{equation*}
    f(x) = x^2 + 2x + 1
\end{equation*}
```

All equations that has their newline and centre staged are mostly written in an environment where it has a `begin` and an `end`. You may have noticed the asterisks sign just after the equation. This implies the environment is not numbered, meaning you won't be able to reference it. This is used to limit the numbering of equations to just the essential parts in the document and not reach 3 digits by the time you are in page 8. For a numbered equation like the following

$$f(x) = x^2 + 2x + 1 \tag{1.1}$$

You only need to do:

Entering an equation with a reference — latex

```latex
\begin{equation}\label{eq:quad}
    f(x) = x^2 + 2x + 1
\end{equation}
```

where `\label{eq:quad}` is the equation reference label. You could also make matrices as well as `amsmath` is preloaded into this template.

1.5.2 Designing a Table

Finally, no template is done without someone telling you how a table should be designed. Below is a standard table And the code used to generate it:

Section	Scientific Method Step
Introduction	states your hypothesis
Methods	details how you tested your hypothesis
Results	provides raw (i.e., uninterpreted) data collected
Discussion	considers whether the data you obtained support the hypothesis

Table 1.1: A Detailed look into the scientific method.

A NiceTabular Table latex

```latex
\begin{table}[!ht]
    \begin{NiceTabular}{rX}[rules/color=[gray]{0.9},rules/width=1pt]
        \CodeBefore
        \rowcolors{1}{black!5}{}
        \rowcolors{3}{blue!5}{}
        \Body
        \toprule
        \textbf{Section}      & \textbf{Scientific Method Step}   \\
        \midrule
        \textbf{Introduction} & states   hypothesis               \\
        \textbf{Methods}      & how you tested hypothesis         \\
        \textbf{Results}      & provides raw  data collected      \\
        \textbf{Discussion}   & whether it support the hypothesis \\
        \bottomrule
    \end{NiceTabular}
    \caption{A Detailed look into the scientific method.}
\end{table}
```

1.6 Cover Styles

Currently the document supports four (4) styles of cover which can be access from the document class with `cover` option.

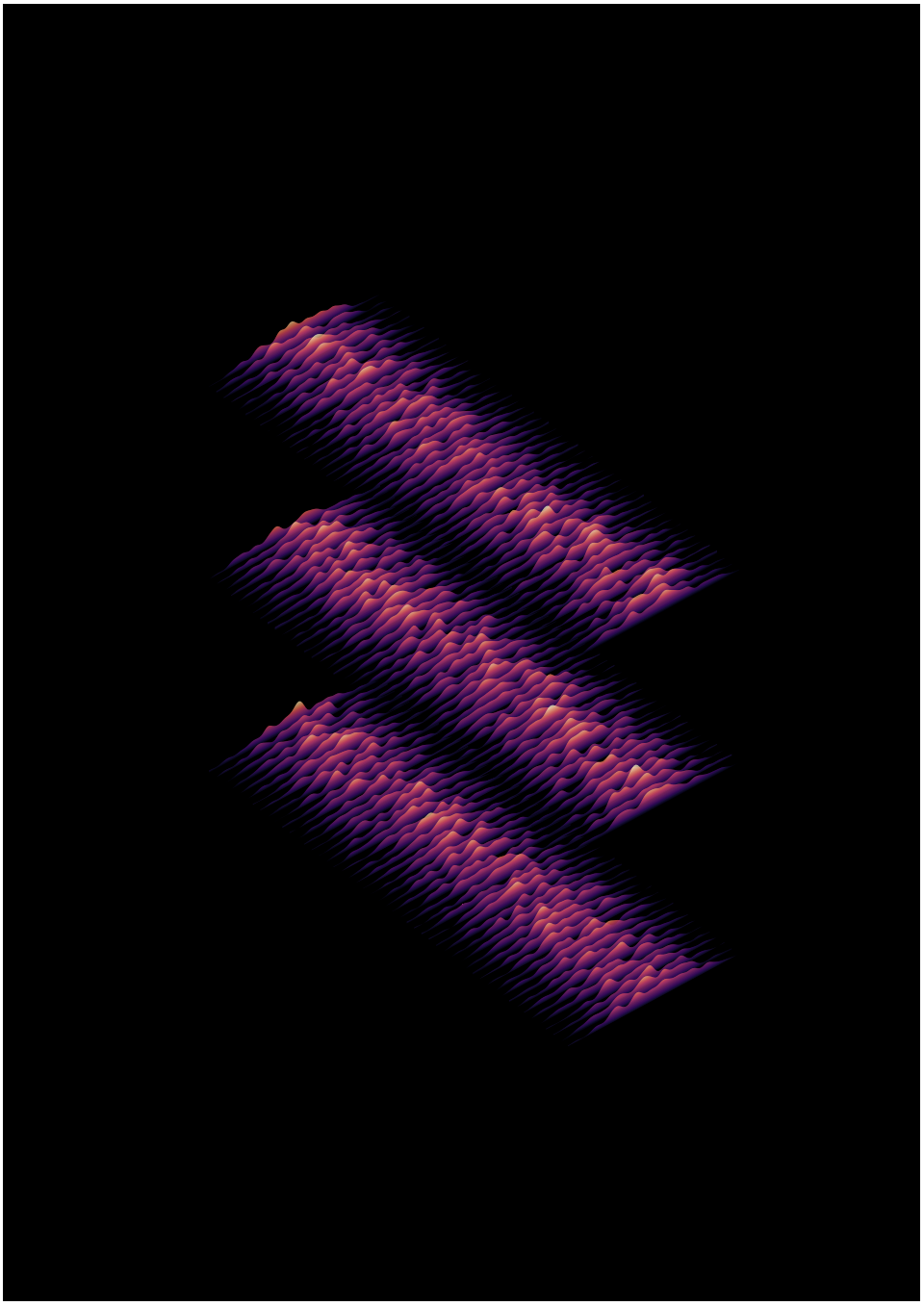

Figure 1.2: Cascading waveforms (options `cover = waves`).

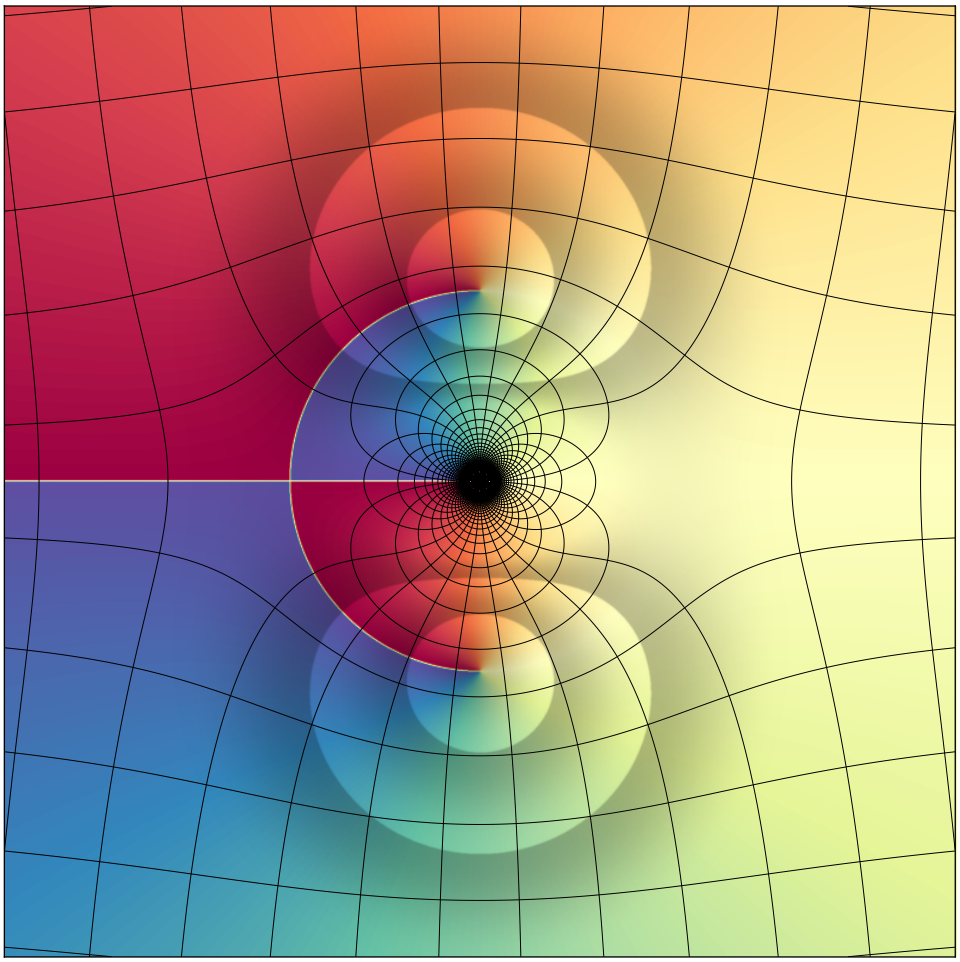

Figure 1.3: Colouring of a complex domain (options `cover = complex`).

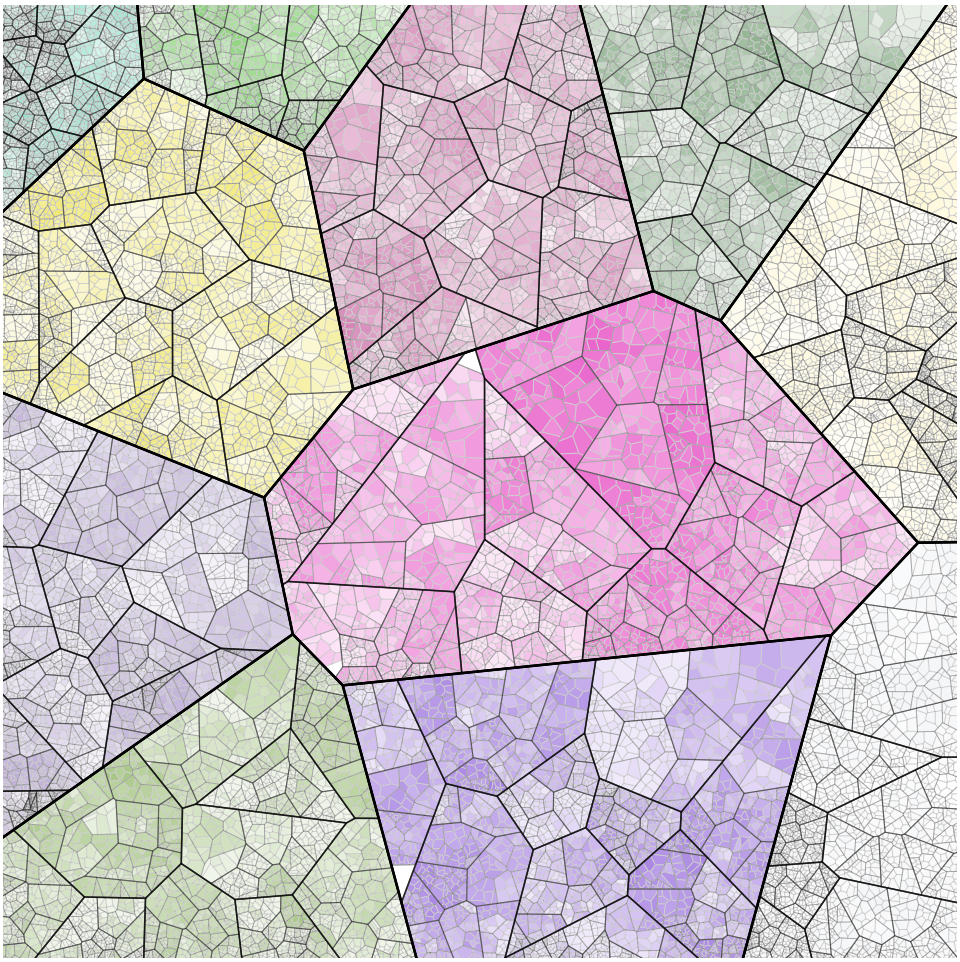

Figure 1.4: Colouring of a Voronoi Plot (options `cover = voronoi`).

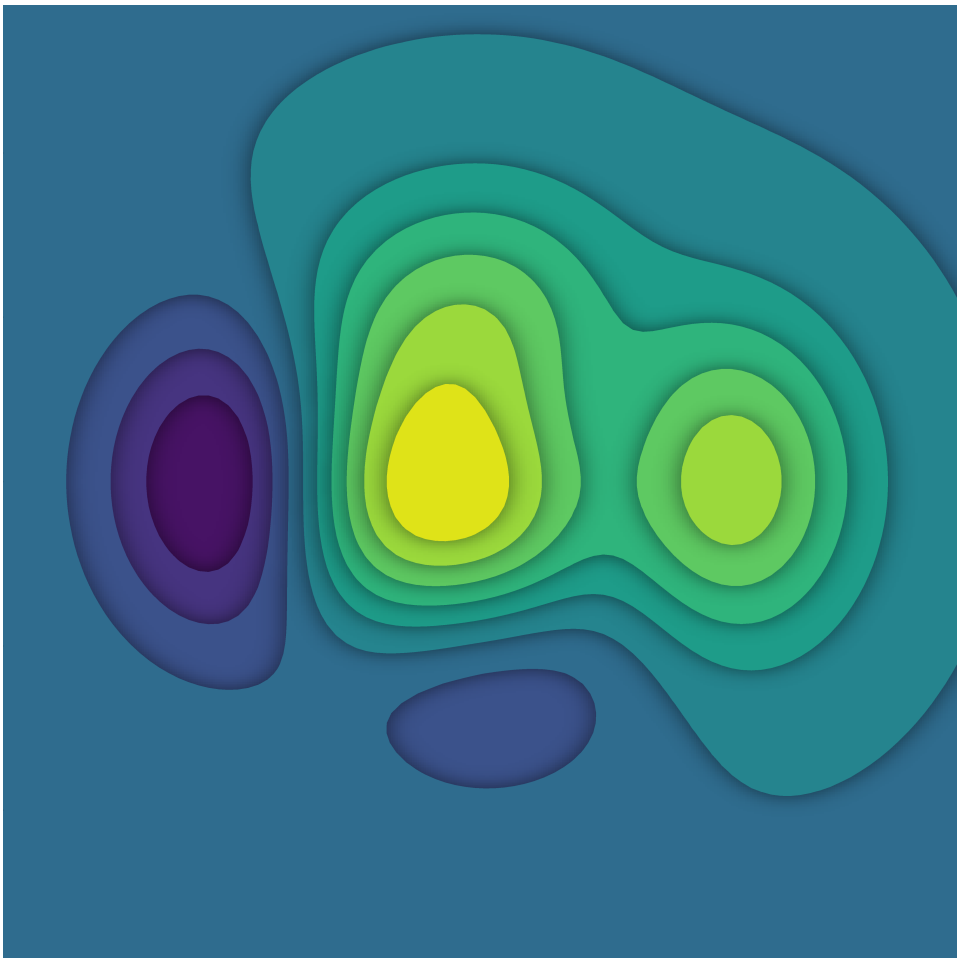

Figure 1.5: Colouring of a Voronoi Plot (options `cover = contour`).

Chapter 2

Plotting your data using PGF/TikZ

Contents

2.1 Introduction

PGFplots and Tikz are powerful scripting languages allowing you to draw high-quality diagrams using only a programming language. PGFplots are generally used for plotting data from a wide variety of representations from simple 2D plots to complex 3D geometries.
But wikipedia description put it best:

> PGF/TikZ is a pair of languages for producing vector graphics (e.g., technical illustrations and drawings) from a geometric/algebraic description, with standard features including the drawing of points, lines, arrows, paths, circles, ellipses and polygons. PGF is a lower-level language, while TikZ is a set of higher-level macros that use PGF. The top-level PGF and TikZ commands are invoked as TeX macros, but in contrast with PSTricks, the PGF/TikZ graphics themselves are described in a language that resembles MetaPost.

For more info please look at the documentation here. It is of course up to the user to select which graphical software to produce the necessary visual components but unless it requires complex functions/processing, it would be be easier to have it in PGF/TikZ format for easy editing/maintenance.
For this manual we will be looking at the two (2) plot types you may encounter in your studies.

2.1.1 A Simple 2D Plot

2D plots are simple yet powerful to show the relation of a single parameters and its related function. Below is an example of a simple comparison of two (2) functions. The image above

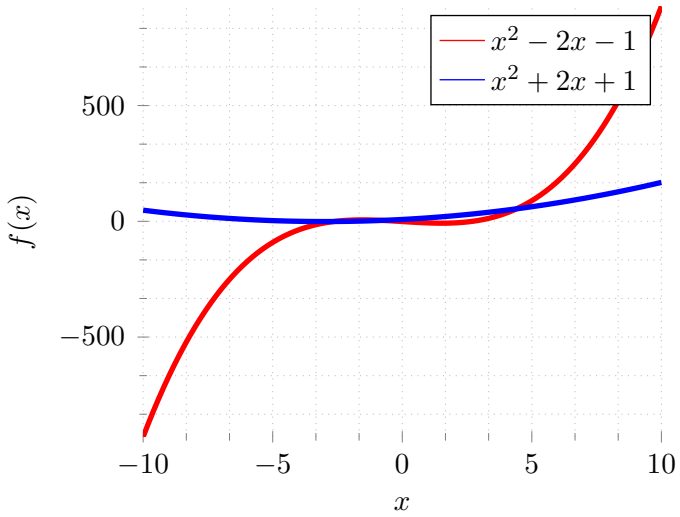

Figure 2.1: This is an example of a 2D PGF plot comparing two functions where these functions are calculated using PGF itself rather than entering/reading from data.

is generated using the following code:

A 2D PGF Plot — latex

```latex
\begin{figure}[!ht]
    \centering
    \begin{tikzpicture}
        \begin{axis}[hebdomon, xlabel = \(x\), ylabel = {\(f(x)\)}]
            %
            \addplot [domain=-10:10, samples=100,red]{x^3 - 7*x - 1};
            \addlegendentry{\(x^2 - 2x - 1\)}
            %
            \addplot [domain=-10:10, samples=100, blue]{x^2 + 6*x + 8};
            %
            \addlegendentry{\(x^2 + 2x + 1\)}
            %
        \end{axis}
    \end{tikzpicture}
    \caption{This is an example of a 2D PGF plot comparing
        two functions where these functions are calculated using
        PGF itself rather than entering/reading from data.}
\end{figure}
```

As can be seen it is relatively standard to create plots. Some aspect which need mentioning.

`\addplot` You invoke this command when you want to create a plot. In the square brackets (i.e., []) you insert your configuration of your plot. The most important ones are

`domain` the range in which the function will be calculated

`sample` the number of calculations will be done within the defined domain.

As seen the code uses a pre-defined style for plotting which in detail can be seen below:

The hebdomon PGF style — latex

```latex
\pgfplotsset{
    hebdomon/.style={
            minor grid style={dotted, gray!50},
            major grid style={dotted, gray!50},
            %
            grid = both,
            minor tick num=2,
            ytick align=outside,
            xtick align=outside,
            axis line style={draw=none},
            axis lines = left,
            %
            line width=2pt,
            %
            legend style = {
                    line width=0.5pt
                },
            %
            every non boxed x axis/.append style={x axis line style=-},
            every non boxed y axis/.append style={y axis line style=-},
            %
        },
}
```

2.1.2 Plotting 3D plots

Plotting data with PGFplots is also quite possible and will generate great plot (as long as it is not massively complicated). For more information on the precautions on designing 3D plots, please have a look at here.
Below is the prototypical plot to showcase the 3D capabilities of PGF:

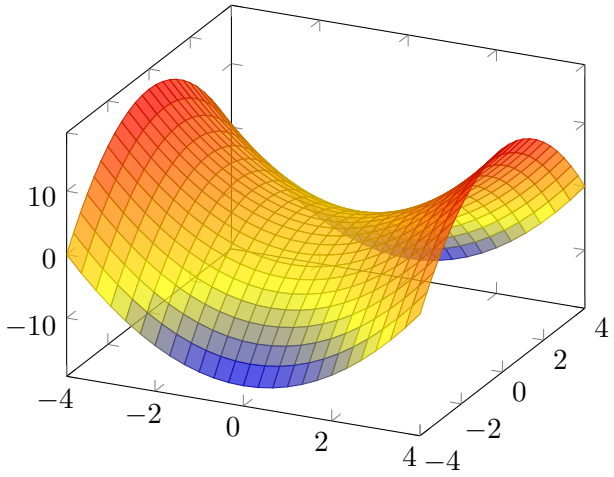

Figure 2.2: An example 3D plot done wit PGFplots.

And, of course the code for generating the plot is given as follows:

A 3D PGF plot — latex

```latex
\begin{figure}[!ht]
    \centering
    \begin{tikzpicture}
        \begin{axis}[view={25}{30},mark layer=like plot]
            \addplot3 [
                surf,
                shader=faceted,
                fill opacity=0.75,
                samples=25,
                domain=-4:4,
                y domain=-4:4,
                on layer=main,
            ] {x^2-y^2};
        \end{axis}
    \end{tikzpicture}
    \caption{An example 3D plot done wit PGFplots.}
\end{figure}
```

Some options worth mentioning are as follows:

`surf` Generates a surface based on the 2D data it was given (in this case these are x and y.

`shader` Describes, basically how each segment should be filled.

`samples` Similar to 2D plots, tells how many data points will be measured. However, make a note that 3D is significantly more taxing on the TeX memory than 2D and making this sampling high may result in exceeding the memory limit.