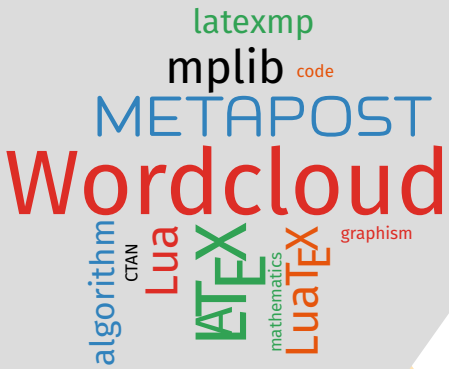
wordcloud

drawing wordclouds
with METAPOST and Lua



Contributor
Maxime CHUPIN
notezik@gmail.com

Version 0.3, 2025, Novembre, 16th
https://github.com/chupinmaxime/wordcloud

Abstract

These METAPOST and LuaLaTeX packages allows drawing wordclouds from a list of words and weights. The algorithm is implemented with METAPOST whereas Lua is used to parse LaTeX commands, to build the list of words and weights from a text file, and to generate METAPOST code interpreted by luamplib.

https://plmlab.math.cnrs.fr/mchupin/wordcloud
https://github.com/chupinmaxime/wordcloud

Contents

This package is in beta version—do not hesitate to report bugs, as well as requests for improvement, or better: to help me to improve it.

1 Installation

wordcloud is on CTAN and can also be installed via the package manager of your distribution.

https://www.ctan.org/pkg/wordcloud

1.1 With TEXlive under Linux or macOS

To install wordcloud with TEXLive, you will have to create the directory `texmf` in your home.

```
user $> mkdir ~/texmf
```

Then, you will have to place the `wordcloud.mp` file in

`~/texmf/metapost/wordcloud/`

You will also have to place the `wordcloud.lua` file in

`~/texmf/scripts/wordcloud/`

And finally, you will have to place the `wordcloud.sty` file in

`~/texmf/tex/latex/wordcloud/`

Once this is done, wordcloud will be loaded with the classic METAPOST input code

```
input wordcloud
```

And for the LuaLATEX side, wordcloud will be loaded with

```
\usepackage{wordcloud}
```

1.2 With MikTEX and Windows

These two systems are unknown to the author of wordcloud, so we refer you to the MikTEXdocumentation concerning the addition of local packages:

`http://docs.miktex.org/manual/localadditions.html`

1.3 Dependencies

wordcloud depends, for the METAPOSTside, of course on METAPOST [6], but also on metapost-colorbrewer [7] and the latexmp package [4]. For the LuaLATEX side [5], wordcloud depends on the luamplib package [2] and the xcolor [3].

2 METAPOST side

2.1 Description of the algorithm

Given a set of words and weights, we first use a *scale function* of the weights to scale the words. In this beta version of wordcloud, we only provide a log-based function[1].

[1]Other scale options could be provided in the next versions.

Then, we compute a spiral line starting at the center[2].

Then the algorithm is quite simple:

Require: set of words $(W_i)_{i\in\{1,\dots,N}$ and corresponding weight $(w_i)_{i\in\{1,\dots,N}$, and a spiral line S

for all $i \in \{1, \dots, N\}$ **do**
  Place W_i at the start of S
  repeat
    Set b_{draw} := true
    for all $j \in 1, \dots, i$ **do**
      if $W_i \cap W_j \neq \emptyset$ **then**
        Set b_{draw} := false
      end if
    end for
    if b_{draw} == true **then**
      Draw W_i
    else
      Move W_i along S
    end if
  until W_i is drawn
end for

The hard part is making it perform efficiently! According to Jonathan Feinberg, Wordle[3] uses a combination of hierarchical bounding boxes and quadtrees to achieve reasonable speeds. Here, with METAPOST, we compute intersections with the bounding box of the word.

Remark

- The words with METAPOST are built with the `textext()` function of latexmp or luamplib. We are trying to use the bounding boxes of the letters when we get an intersection between "global" bounding boxes to allow placing words nearer of each other. Unfortunately, this does not work for the moment. Any help is welcomed.
- We first tried to compute intersections between words by decomposing the letter using their contours and compute intersection of contours (with `intersectiontimes`). Unfortunately, this is much too slow.

Some explanations can be found here:

`https://www.jasondavies.com/wordcloud/about/`

2.2 Main command

The main command is

`draw_wordcloud(⟨words⟩, ⟨weights⟩,⟨rotation⟩,⟨size⟩)`

[2]There is variants of the algorithm that use different line: squared spiral, etc.
[3]One of the first web application to build wordcloud.

⟨***words***⟩**:** array of strings ;

⟨***weights***⟩**:** array of numerics ;

⟨***rotation***⟩**:** angle for wordcloud drawing;

⟨***size***⟩**:** number of elements in arrays.

Exemple METAPOST **1**

```
input wordcloud
beginfig(0);
string words[];
numeric weights[];
words[1]:="\LaTeX";
words[2]:="\hologo{METAPOST}";
words[3]:="Document";
words[4]:="Lua";
words[5]:="\TeX";
weights[1]:=5;
weights[2]:=4;
weights[3]:=3.5;
weights[4]:=3;
weights[5]:=3;
draw_wordcloud(words,weights,0,5);
endfig;
```

Remark

The “unity” of weights is not important because internally, wordcloud compute new weights to work with the internal scaling function.

2.3 Parameters

There are few parameters.

2.3.1 Colors

You can use set of colors to draw the wordcloud. For that, you have to use the following command:

`wordcloud_use_color(`⟨*bool*⟩`)`

⟨***bool***⟩: boolean `true` or `false` (default `false`).

wordcloud provides a set of five colors using the METAPOST package metapost-colorbrewer [7]. wordcloud defines an array of `color`s and a `numeric` to set the colors to use.

```
wordcloud_colors[1]:=Reds[3][3];
wordcloud_colors[2]:=Greens[3][3];
wordcloud_colors[3]:=Blues[3][3];
wordcloud_colors[4]:=Oranges[3][3];
wordcloud_colors[5]:=black;
wordcloud_colors_number:=5;
```

Feel free to modify that variables to customize the colors.

2.3.2 Scaling

You can globally scale the picture using the following command:

`set_wordcloud_scale(`⟨*scale*⟩`)`

⟨***scale***⟩: `numeric`.

2.3.3 Margins

You can adjust the margins of the global bounding boxes of words using the following command:

`set_box_margin(`⟨*dim*⟩`)`

⟨***dim***⟩: a dimension with units (default 0.3pt).

3 LuaLaTeX side

wordcloud provides a LuaLaTeX package. It uses the package luamplib to interpret the METAPOST code produced by Lua.

3.1 Main commands

The first LaTeX command provided by wordcloud is:

`\wordcloud[`⟨*options*⟩`]{`⟨*list of words and weights*⟩`}`

where

⟨***list of words and weights***⟩: is a list of couples of the form `(word1,weight1);(word2,weight2);(word3,weight3);...`

The second LaTeX command allows to read a text file, to build the list of words and weights and draw the wordcloud up to a certain number of words.

`\wordcloudFile[⟨options⟩]{⟨text file⟩}{⟨number of words⟩}`

where:

⟨***text file***⟩**:** is a text file to analyze and from which the wordcloud is build ;

⟨***number of words***⟩**:** is the number of words composing the wordcloud.

3.1.1 Options

Both of these functions (`\wordcloud` and `\wordcloudFile`) have the same options:

`scale`=⟨***value***⟩**:** to scale the picture[4] ;

`margin`=⟨***value with units***⟩**:** to adjust the margins (default `0.3pt`) ;

`rotate`=⟨***angle***⟩**:** to rotate (degrees) the words with ±⟨*angle*⟩ alternatively (default 0) ;

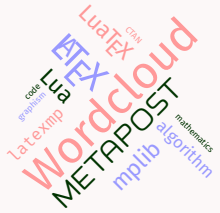
`usecolor`: to use color for word drawing (boolean, default `false`) as described in section 2.3.1 ;

`colors`=⟨***list of colors***⟩**:** to define a new set of colors as described in section 2.3.1[5].

Here an example:

Exemple LaTeX 1

```
\wordcloud[scale=1,rotate=45,margin=0.5pt,usecolor,colors={red
    !40,blue!40,green!20!black}]{(Wordcloud,10);(\hologo{
    METAPOST},6);(\LaTeX,7);(Lua,4);(algorithm,3);(code,2);(
    mathematics,2);(CTAN,2);(mplib,4);(\hologo{LuaTeX},4);(\
    texttt{latexmp},3);(graphism,2)}
```



[4]Beware that scaling increases the computation time and the values manipulated by METAPOST.

[5]This needs xcolor because the colors are converted to rgb coding and then transferred to METAPOST.

Since version 0.2, you could use any LaTeX code to define words for the wordcloud. In the definition of the LaTeX command, wordcloud uses `\luaescapestring` to deals with commands. For example, as far as we know, this wordcloud package is the only wordcloud tool that could build a wordcloud of mathematical formulas.

Remark

- Because the METAPOST side uses `textext` command to build words, `\mplibtextextlabel{enable}` is used to enable string labels typeset via `textext()` instead of *infont* operator.
- The `\wordcloud` command build an `mplibcode` environment with METAPOST code. Each of the `mplibcode` environment use the `luamplib` [2] mechanism of *instance* named `wordcloud`.

3.2 Add ignored words

The Lua function that builds words and weights from a text file ignores some words (and characters). For the moment, wordcloud only includes word lists to ignore for English and French.

However, you can add a list of words to ignore with the following command:

`\wordcloudIgnoreWords{`⟨*word list*⟩`}`

⟨***word list***⟩**:** the list of words, separated with commas, to ignore `word1`, `word2`, `word3`, `etc.`

3.3 PDF tagging

Thanks to the demand of David Carlisle, the LaTeX commands generate now, if the tagging in on with LuaLaTeX, alternative text to describe the wordcloud produced. It uses the `luamplib` mechanism tot deal with PDF tagging.

4 With `pdftotext`

Thanks to wordcloud and the program `pdftotext`[6] one can easily produce the wordcloud of the current PDF.

For that, you can produce the text file of the PDF:

```
user $> pdftotext wordcloud-doc-en.pdf
```

and then, you can use the following code:

[6] It should be possible to parse a PDF with LuaTeX, though. See https://tex.stackexchange.com/questions/692930/recovering-the-textual-content-of-a-pdf-file-with-luatex.

```
\wordcloudFile[usecolor]{wordcloud-doc-en.txt}{50}
```

This produce the following wordcloud[7]

5 To do

Some things to do:

- Improve intersection of words by using the letters bounding boxes.
- Work on speed of the algorithm.
- Add supported languages (ignored words).
- Improve text file analysis with Lua to build the set of words and weights.
- Build wordcloud inside a shape.

[7]Note that, because the wordcloud production is slow, we used a separate file to only produce the PDF of the wordcloud, without any scaling, and we inserted the result scaling it with graphicx [1].

- Add options for rotation of words.

6 History

v0.3 (11/2025): Add PDF tagging for LaTeX figures.

v0.2 (09/2023): Deal with `\luaescapestring` for list with `\wordcloud`. Typos in doc.

v0.1 (08/2023): First publication.

References

[1] David Carlisle and The LaTeX Project Team. *The `graphicx` package. Enhanced support for graphics*. Version 1.2d. Nov. 12, 2021. URL: https://ctan.org/pkg/graphicx.

[2] Hans Hagen et al. *The `luamplib` package. Use LuaTeX's built-in MetaPost interpreter*. Version 2.23.0. Jan. 12, 2022. URL: https://ctan.org/pkg/luamplib.

[3] Uwe Kern and The LaTeX Project Team. *The `xcolor` package. Driver-independent color extensions for LaTeX and pdfLaTeX*. Version 2.14. June 12, 2022. URL: https://ctan.org/pkg/xcolor.

[4] Jens-Uwe Morawski. *The `latexMP` package. Interface for LaTeX-based typesetting in MetaPost*. Version 1.2.1. June 21, 2020. URL: https://ctan.org/pkg/latexmp.

[5] Manuel Pégourié-Gonnard. *The `lualatex-doc` package. A guide to use of LaTeX with LuaTeX*. June 24, 2016. URL: https://ctan.org/pkg/lualatex-doc.

[6] The MetaPost Team and John Hobby. *The `metapost` package. A development of Metafont for creating graphics*. Aug. 26, 2021. URL: https://ctan.org/pkg/metapost.

[7] Toby Thurston. *The `metapost-colorbrewer` package. An implementation of the colorbrewer2.org colours for MetaPost*. Sept. 25, 2018. URL: https://ctan.org/pkg/metapost-colorbrewer.

Command Index