
stemgraphic Documentation

Release 0.5.2

Francois Dion

Feb 25, 2018

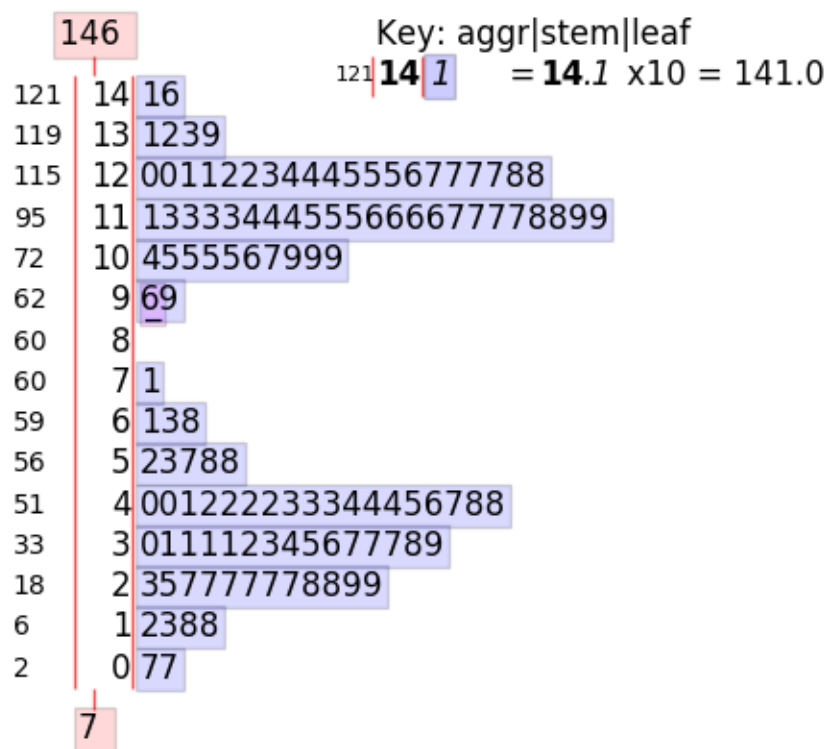
CONTENTS:

1	Introduction	1
2	Installation	3
3	stemgraphic quickstart with numbers	5
4	stemgraphic quickstart with categorical	7
5	stemgraphic quickstart with text	9
6	Included Stop words	11
7	Gallery	13
8	Stemgraphic Modules	17
8.1	stemgraphic	17
8.2	aliases	17
8.3	alpha	18
8.4	graphic	33
8.5	helpers	35
8.6	num	37
8.7	stopwords	37
8.8	text	38
9	Indices and tables	41
	Python Module Index	43
	Index	45

INTRODUCTION

John Tukey's stem-and-leaf plot first appeared in 1970. Although very useful back then, it cannot handle more than 300 data points and is completely text-based. Stemgraphic is a very easy to use python package providing a solution to these limitations.

A typical stem_graphic output:



For an in depth look at the algorithms and the design of stemgraphic, see

<https://github.com/fdion/stemgraphic/raw/master/doc/stemgraphic%20A%20Stem-and-Leaf%20Plot%20for%20the%20Age%20of%20Big%20Data.pdf>

A PDF version of the documentation is available at: <http://stemgraphic.org/doc/stemgraphic.pdf>

The official website of stemgraphic is: <http://stemgraphic.org>

INSTALLATION

Stemgraphic requires docopt, matplotlib, pandas and seaborn. Optionally, having Scipy installed will give you secondary plots, cufflinks for interactive plots and Dask (see requirements_dev.txt for all needed to run all the functional tests) will allow for out of core, big data visualization.

If you use conda, it is recommended you conda install docopt, matplotlib, pandas, seaborn, and scipy before doing a pip install of stemgraphic.

Installation of stemgraphic is simple:

```
pip3 install -U stemgraphic
```

or from a clone of the github repository, in the package root:

```
python3 setup.py install
```


STEMGRAPHIC QUICKSTART WITH NUMBERS

Import `stem_graphic` from `stemgraphic` (shortcut) or explicitly from `stemgraphic.num`.

```
In [1]: %matplotlib inline
import pandas as pd
from stemgraphic import stem_graphic
```

Load a data frame

```
In [2]: df = pd.read_csv('../iris.csv')
```

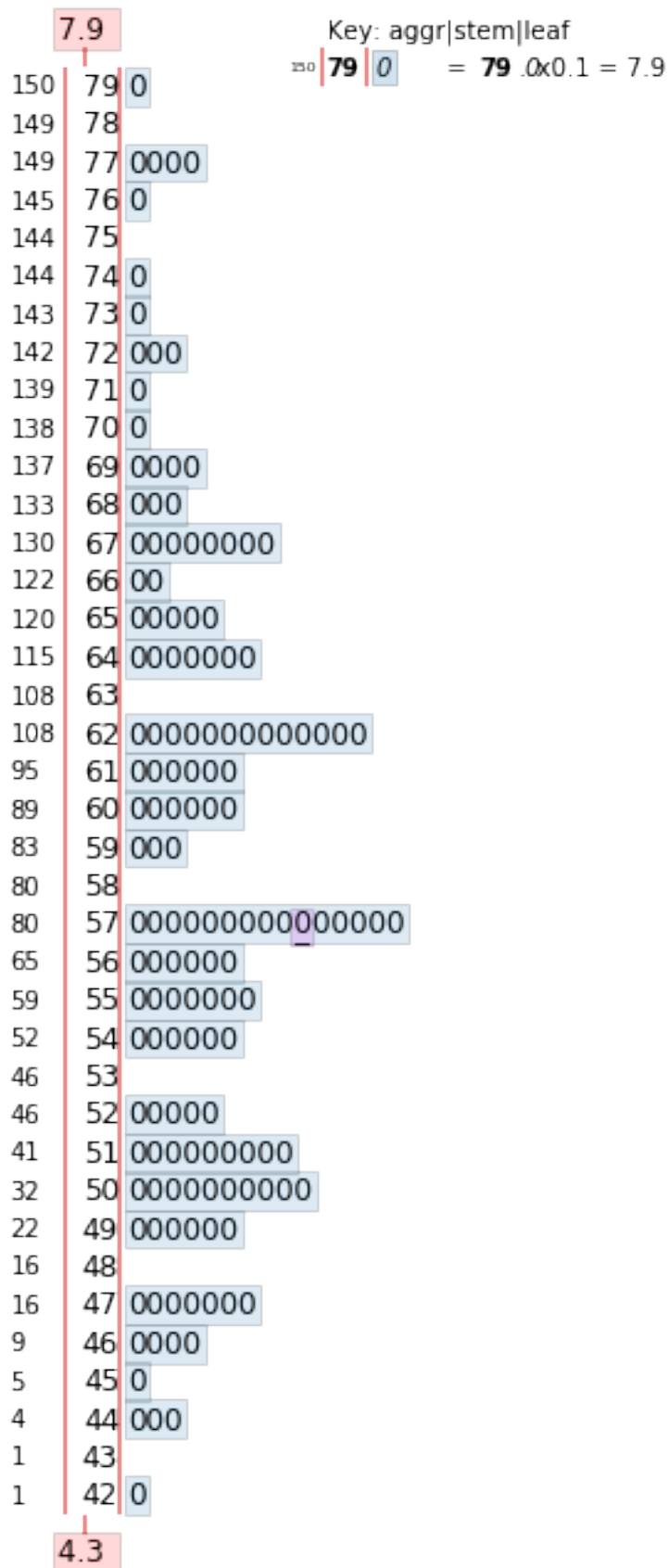
```
In [3]: df.describe()
```

```
Out[3]:
```

	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

Select a column, or pass the whole dataframe if you want `stem_graphic` to select the first numerical column.

```
In [4]: stem_graphic(df['sepal_length']);
```



Import stem_graphic from stemgraphic.alpha

```
In [1]: %matplotlib inline
import pandas as pd
from stemgraphic.alpha import stem_graphic
```

Load a data frame

```
In [2]: df = pd.read_csv('../iris.csv')
```

```
In [3]: df.describe(include='all')
```

```
Out[3]:
```

	sepal_length	sepal_width	petal_length	petal_width	species
count	150.000000	150.000000	150.000000	150.000000	150
unique	NaN	NaN	NaN	NaN	3
top	NaN	NaN	NaN	NaN	versicolor
freq	NaN	NaN	NaN	NaN	50
mean	5.843333	3.054000	3.758667	1.198667	NaN
std	0.828066	0.433594	1.764420	0.763161	NaN
min	4.300000	2.000000	1.000000	0.100000	NaN
25%	5.100000	2.800000	1.600000	0.300000	NaN
50%	5.800000	3.000000	4.350000	1.300000	NaN
75%	6.400000	3.300000	5.100000	1.800000	NaN
max	7.900000	4.400000	6.900000	2.500000	NaN

Select a column with text.

```
In [4]: stem_graphic(list(df['species'].values));
```

```
50 | s| eeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee  
150 | v| eeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeei  
                                     |iiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii
```

From this, we see we have 50 *setosa*, 50 *versicolor* and 50 *virginica*, but you probably already knew that!

STEMGRAPHIC QUICKSTART WITH TEXT

Import stem_graphic from stemgraphic.alpha

```
In [2]: %matplotlib inline
        from stemgraphic.alpha import stem_graphic
```

Load words from a text file on disk.

```
In [3]: stem_graphic('/usr/share/dict/american-english');
```

/usr/share/dict/american-english

```

1  Y a
2  J i
3  Q u
5  k iu
7  f ae
9  W er
11 O dL
13 K ar
15 N au
17 V ei
20 U knr
24 E bcrv
29 T erruw
34 L aeio
40 R aeooou
46 G aeoeo
52 V eiio
58 J aaoou
65 A cnnttu
72 P aeoeels
79 H Iaaaeou
86 D aeiooy
97 C aaahhooouo
108 w aeehhiiooor
120 u bnnnnnnnnnt
132 S aceehhhiott
144 M TVaaaeiory
157 h aaeeiooouuy
170 B aaehiloouuu
183 n aaeeeeeiiio
197 o abblnpstuuuvv
211 g aaaiillnorrer
228 a bccdfmnnppprssst
248 l aaataaeeeeiiioouu
268 f aaataeeiilllloooorrr
290 e eeLmmnnnnngsstvxxxxx
314 i dflmmnnnnnnnnnnnnnnrrt
340 b aaataaeeeeiiillloorruuuuu
368 t aaeeehhhhiioooorrrrrruuwy
398 r aaataaeeeeieeieeieeieeouu
429 d aaeeieeieeieeieeiiillloooorrruuu
460 m aaataaeataaeieeieeiiillloooouuuu
503 p aaataaeieeieeiiilllloooouooorrrrrrrrruuu
558 c aaataaeataaeieeieeiiilllloooouooouooouooouooorrrrruuu
750 s aaacccceehhhiiooopppppqtttttuuuuuwwy
```


INCLUDED STOP WORDS

```
In [1]: from stemgraphic.stopwords import EN, FR, ES, ALT_EN
```

Very short list of English stop words

```
In [2]: len(ALT_EN)
```

```
Out[2]: 27
```

```
In [3]: print(ALT_EN)
```

```
['a', 'am', 'an', 'and', 'are', 'as', 'at', 'been', 'for', 'from', 'in', 'is', 'of', 'on', 'or', 'out']
```

The French and Spanish stop words are quite similar, but Spanish has several gender specific words (i.e. *quelque* vs. *algun*, *algunos*, *algunas*) so it is larger.

```
In [4]: len(FR)
```

```
Out[4]: 127
```

```
In [5]: len(ES)
```

```
Out[5]: 183
```

The main English stop word list is significantly larger.

```
In [6]: len(EN)
```

```
Out[6]: 316
```


GALLERY

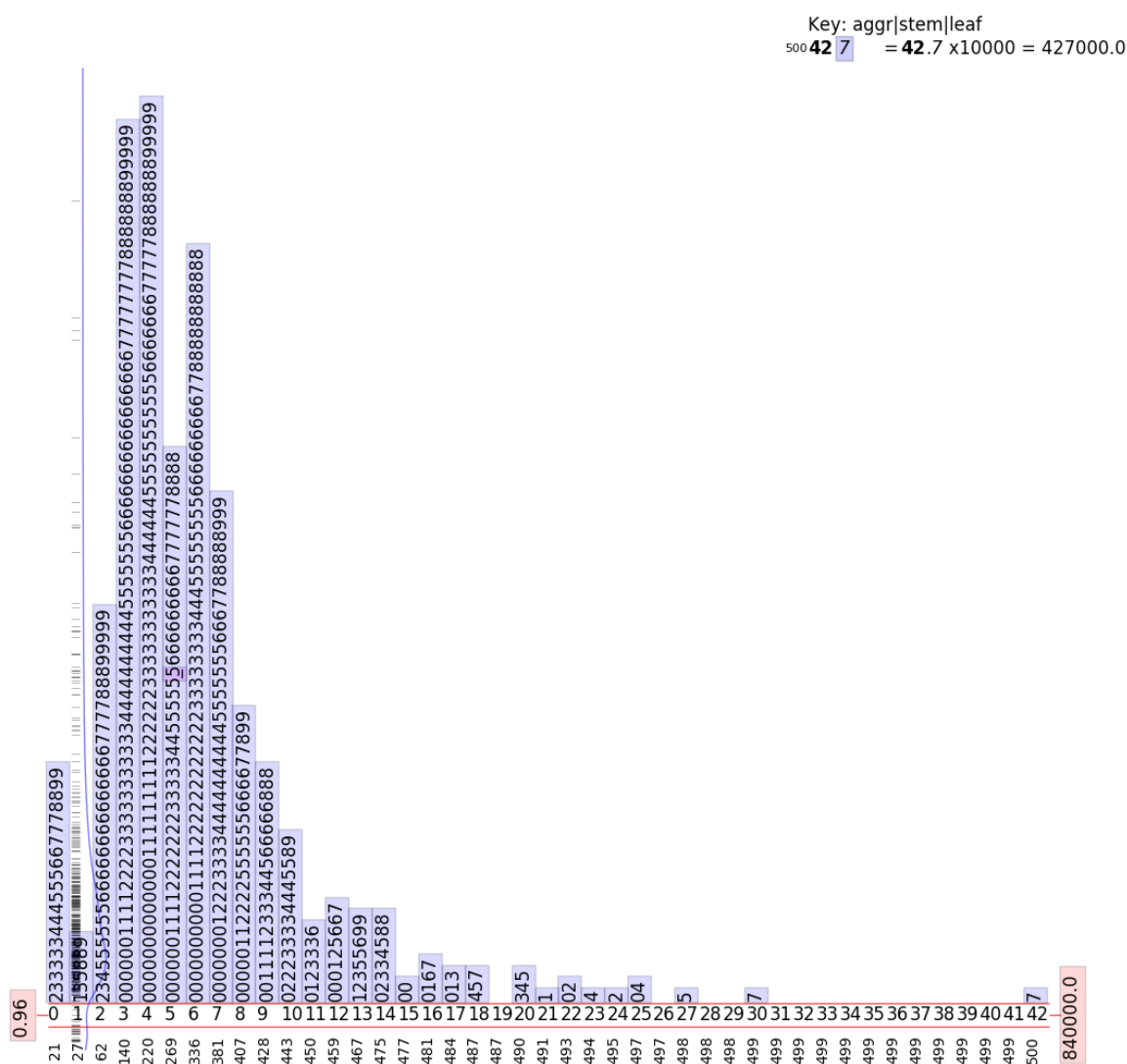


Fig. 7.1: Stem-and-Leaf plot with numbers (salaries)

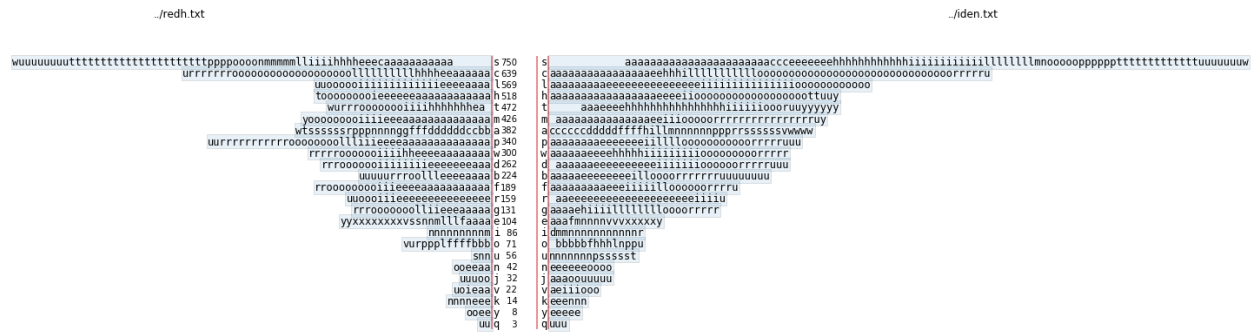


Fig. 7.2: Back-to-back stem-and-leaf plot comparing two text files (Sherlock Holmes stories)



Fig. 7.3: Stem-and-leaf heatmap



Fig. 7.4: Stem-and-Leaf plot from a list of words

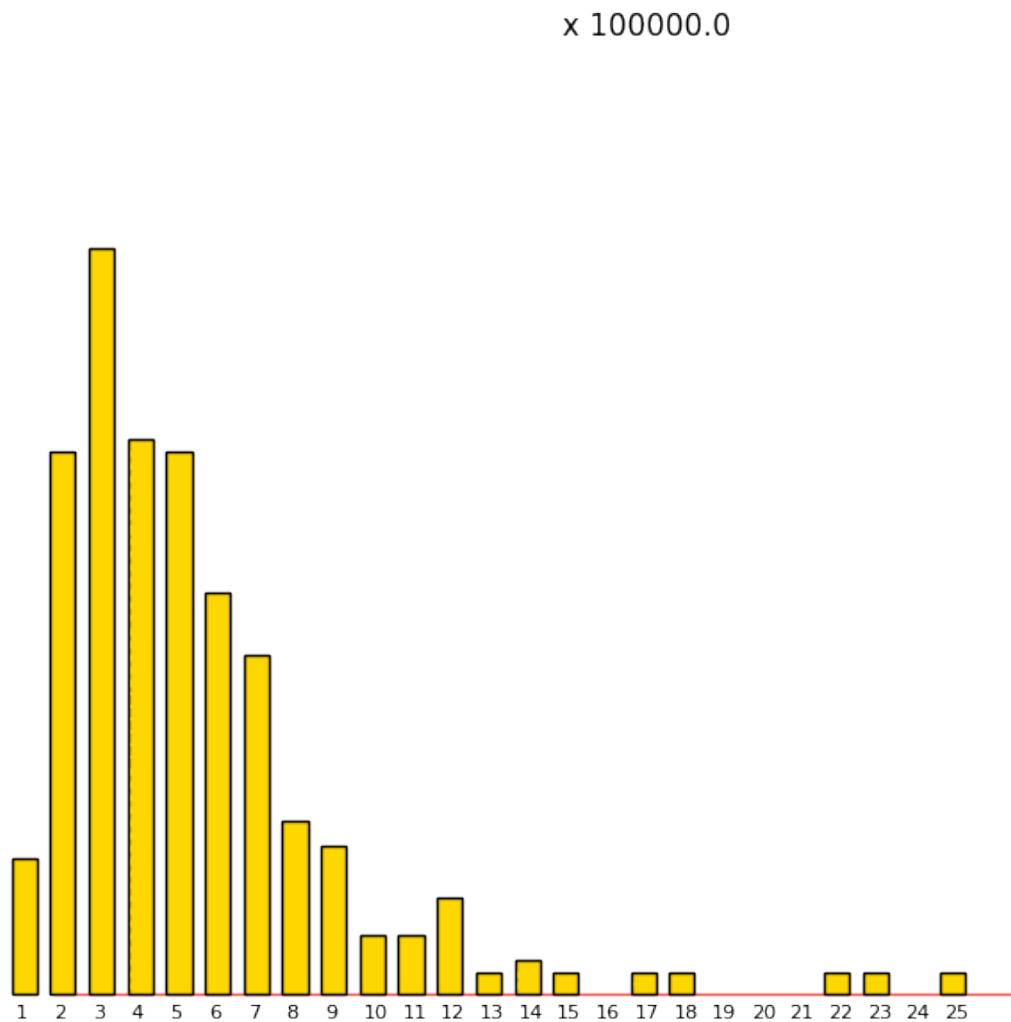


Fig. 7.5: Stem-and-Leaf plot styled as a histogram (with stem binning)

STEMGRAPHIC MODULES

8.1 stemgraphic

stem_graphic

Package implementing a complete toolkit for text and a graphical stem-and-leaf plots and other visualizations adapted to stem-and-leaf pair values, such as heatmaps and sunburst charts.

It also handles very large data sets through scaling, sampling, trimming and other techniques.

See research paper (<http://artchiv.es/pydata2016/stemgraphic>) for more technical details.

A command line utility was installed along with the package, allowing to process excel or csv files. See: `stem -h`

8.2 aliases

Handy aliases for stem_graphic options.

```
stemgraphic.aliases.stem_hist(x, aggregation=False, alpha=1, asc=True, column=None,
                               color='b', delimiter_color='r', display=300, flip_axes=True, leg-
                               end_pos='short', outliers=False, trim=False)
```

stem_hist builds a histogram matching the stem-and-leaf plot, with the numbers hidden, as shown on the cover of the companion brochure.

Parameters

- **legend_pos** –
- **x** – list, numpy array, time series, pandas or dask dataframe
- **aggregation** – Boolean for sum, else specify function
- **alpha** – opacity of the bars, median and outliers, defaults to 15%
- **asc** – stem sorted in ascending order, defaults to True
- **column** – specify which column (string or number) of the dataframe to use, else the first numerical is selected
- **color** – the bar facecolor
- **delimiter_color** – color of the line between aggregate and stem and stem and leaf
- **display** – maximum number of data points to display, forces sampling if smaller than `len(df)`
- **flip_axes** – X becomes Y and Y becomes X

- **outliers** – this is NOP, for compatibility
- **trim** – this is NOP, for compatibility

Returns matplotlib figure and axes instance

`stemgraphic.aliases.stem_kde(x, **kw_args)`

`stem_kde` builds a stem-and-leaf plot and adds an overlaid kde as secondary plot.

Parameters

- **x** – list, numpy array, time series, pandas or dask dataframe
- **kw_args** –

Returns matplotlib figure and axes instance

`stemgraphic.aliases.stem_line(x, aggregation=False, alpha=0, asc=True, column=None, color='k', delimiter_color='r', display=300, flip_axes=True, outliers=False, secondary_plot=None, trim=False)`

`stem_line` builds a stem-and-leaf plot with lines instead of bars.

Parameters

- **x** – list, numpy array, time series, pandas or dask dataframe
- **aggregation** – Boolean for sum, else specify function
- **alpha** – opacity of the bars, median and outliers, defaults to 15%
- **asc** – stem sorted in ascending order, defaults to True
- **column** – specify which column (string or number) of the dataframe to use, else the first numerical is selected
- **color** – the color of the line
- **delimiter_color** – color of the line between aggregate and stem and stem and leaf
- **display** – maximum number of data points to display, forces sampling if smaller than `len(df)`
- **flip_axes** – X becomes Y and Y becomes X
- **outliers** –
- **secondary_plot** – One or more of 'dot', 'kde', 'margin_kde', 'rug' in a comma delimited string or None
- **trim** – this is NOP, for compatibility

Returns matplotlib figure and axes instance

8.3 alpha

`stemgraphic.alpha`.

BRAND NEW in V.0.5.0!

Stemgraphic provides a complete set of functions to handle everything related to stem-and-leaf plots. `alpha` is a module of the `stemgraphic` package to add support for categorical and text variables.

The module also adds functionality to handle whole words, beside stem-and-leaf bigrams and n-grams.

For example, for the word “alabaster”:

With **word_** functions, we can look at the word frequency in a text, or compare it through a distance function (default to Levenshtein) to other words in a corpus

With **stem_** functions, we can look at the fundamental stem-and-leaf, stem would be ‘a’ and leaf would be ‘l’, for a bigram ‘al’. With a `stem_order` of 1 and a `leaf_order` of 2, we would have ‘a’ and ‘la’, for a trigram ‘ala’, so on and so forth.

```
stemgraphic.alpha.add_missing_letters(mat, stem_order, leaf_order, letters=None)
```

Add missing stems based on LETTERS. defaults to a-z alphabet.

Parameters

- **mat** – matrix to modify
- **stem_order** – how many stem characters per data point to display, defaults to 1
- **leaf_order** – how many leaf characters per data point to display, defaults to 1
- **letters** – letters that must be present as stems

Returns the modified matrix

```
stemgraphic.alpha.heatmap(src, alpha_only=False, annotate=False, asFigure=False, ax=None,
                           caps=False, compact=True, display=None, interactive=True,
                           leaf_order=1, random_state=None, stem_order=1, stop_words=None)
```

The heatmap displays the same underlying data as the stem-and-leaf plot, but instead of stacking the leaves, they are left in their respective columns. Row ‘a’ and Column ‘b’ would have the count of words starting with ‘ab’. The heatmap is useful to look at patterns. For distribution, `stem_graphic` is better suited.

Parameters

- **src** – string, filename, url, list, numpy array, time series, pandas or dask dataframe
- **alpha_only** – only use stems from a-z alphabet
- **annotate** – display annotations (Z) on heatmap
- **asFigure** – return plot as plotly figure (for web applications)
- **ax** – matplotlib axes instance, usually from a figure or other plot
- **caps** – bool, True to be case sensitive
- **compact** – remove empty stems
- **display** – maximum number of data points to display, forces sampling if smaller than `len(df)`
- **interactive** – if cufflinks is loaded, renders as interactive plot in notebook
- **leaf_order** – how many leaf characters per data point to display, defaults to 1
- **random_state** – initial random seed for the sampling process, for reproducible research
- **stem_order** – how many stem characters per data point to display, defaults to 1

:param stop_words: stop words to remove. None (default), list or builtin EN (English), ES (Spanish) or FR (French)
:return:

```
stemgraphic.alpha.heatmap_grid(src1, src2, src3=None, src4=None, alpha_only=True,
                               annot=False, caps=False, center=0, cmap=None, display=1000,
                               leaf_order=1, random_state=None, robust=False, stem_order=1,
                               stop_words=None, threshold=0)
```

heatmap_grid.

With `stem_graphic`, it is possible to directly compare two different sources. In the case of a heatmap, two different data sets cannot be visualized directly on a single heatmap. For this task, we designed `heatmap_grid` to adapt to the number of sources to build a layout. It can take from 2 to 4 different source.

With 2 sources, a square grid will be generated, allowing for horizontal and vertical comparisons, with an extra heatmap showing the difference between the two matrices. It also computes a norm for that difference matrix. The smaller the value, the closer the two heatmaps are.

With 3 sources, it builds a triangular grid, with each source heatmap in a corner and the difference between each pair in between.

Finally, with 4 sources, a 3 x 3 grid is built, each source in a corner and the difference between each pair in between, with the center expressing the difference between top left and bottom right diagonal.

Parameters

- **src1** – string, filename, url, list, numpy array, time series, pandas or dask dataframe (required)
- **src2** – string, filename, url, list, numpy array, time series, pandas or dask dataframe (required)
- **src3** – string, filename, url, list, numpy array, time series, pandas or dask dataframe (optional)
- **src4** – string, filename, url, list, numpy array, time series, pandas or dask dataframe (optional)
- **alpha_only** – only use stems from a-z alphabet
- **annot** – display annotations (Z) on heatmap
- **caps** – bool, True to be case sensitive, defaults to False, recommended for comparisons.
- **center** – the center of the divergent color map for the difference heatmaps
- **cmap** – color map for difference heatmap or None (default) to use the builtin red / blue divergent map
- **display** – maximum number of data points to display, forces sampling if smaller than `len(df)`
- **leaf_order** – how many leaf characters per data point to display, defaults to 1
- **robust** – reduce effect of outliers on difference heatmap
- **random_state** – initial random seed for the sampling process, for reproducible research
- **stem_order** – how many stem characters per data point to display, defaults to 1
- **stop_words** – stop words to remove. None (default), list or builtin EN (English), ES (Spanish) or FR (French)
- **threshold** – absolute value minimum count difference for a difference heatmap element to be visible

Returns

`stemgraphic.alpha.matrix_difference(mat1, mat2, thresh=0)`

Parameters

- **mat1** – first heatmap dataframe
- **mat2** – second heatmap dataframe

- **thresh** – : absolute value minimum count difference for a difference heatmap element to be visible

Returns difference matrix, norm and ratio of the sum of the first matrix over the second

```
stemgraphic.alpha.ngram_data(df, alpha_only=False, ascending=True, binary=False,
                             break_on=None, caps=True, char_filter=None, column=None,
                             compact=False, display=750, leaf_order=1, leaf_skip=0, per-
                             sistence=None, random_state=None, remove_accents=False,
                             reverse=False, rows_only=True, sort_by='len', stem_order=1,
                             stem_skip=0, stop_words=None)
```

This is the main text ingestion function for stemgraphic.alpha. It is used by most of the visualizations. It can also be used directly, to feed a pipeline, for example.

If selected (`rows_only=False`), the returned dataframe includes in each row a single word, the stem, the leaf and the ngram (stem + leaf) - the index is the 'token' position in the original source:

12 salut s a sa 13 chéri c h ch

Parameters

- **df** – list, numpy array, series, pandas or dask dataframe
- **alpha_only** – only use stems from a-z alphabet (NA on dataframe)
- **ascending** – bool if the sort is ascending
- **binary** – bool if True forces counts to 1 for anything greater than 0
- **break_on** – letter on which to break a row, or None (default)
- **caps** – bool, True to be case sensitive, defaults to False, recommended for comparisons.(NA on dataframe)
- **char_filter** – list of characters to ignore. If None (default) CHAR_FILTER list will be used
- **column** – specify which column (string or number) of the dataframe to use, or group of columns (stems) else the frame is assumed to only have one column with words.
- **compact** – remove empty stems
- **display** – maximum number of data points to display, forces sampling if smaller than `len(df)`
- **leaf_order** – how many leaf characters per data point to display, defaults to 1
- **leaf_skip** – how many leaf characters to skip, defaults to 0 - useful w/shared bigrams: 'wol', 'wor', 'woo'
- **persistence** – will save the sampled datafrae to filename (with csv or pkl extension) or None
- **random_state** – initial random seed for the sampling process, for reproducible research
- **remove_accents** – bool if True strips accents (NA on dataframe)
- **rows_only** – bool by default returns only the stem and leaf rows. If false, also the matrix and dataframe
- **sort_by** – default to 'len', can also be 'alpha'
- **stem_order** – how many stem characters per data point to display, defaults to 1

- **stem_skip** – how many stem characters to skip, defaults to 0 - useful to zoom in on a single root letter
- **stop_words** – stop words to remove. None (default), list or builtin EN (English), ES (Spanish) or FR (French)

Returns ordered rows if rows_only, else also returns the matrix and dataframe

`stemgraphic.alpha.plot_sunburst_level` (*normalized, ax, label=True, level=0, offset=0, ngram=False, plot=True, stem=None, vis=0*)

utility function for sunburst function.

Parameters

- **normalized** –
- **ax** –
- **label** –
- **level** –
- **ngram** –
- **offset** –
- **plot** –
- **stem** –
- **vis** –

Returns

`stemgraphic.alpha.polar_word_plot` (*ax, word, words, label, min_dist, max_dist, metric, offset, step*)

Utility function for radar plot.

Parameters

- **ax** – matplotlib ax
- **word** – string, the reference word that will be placed in the middle
- **words** – list of words to compare
- **label** – bool if True display words centered at coordinate
- **min_dist** – minimum distance based on metric to include a word for display
- **max_dist** – maximum distance for a given section
- **metric** – any metric function accepting two values and returning that metric in a range from 0 to x
- **offset** – where to start plotting in degrees
- **step** – how many degrees to step between plots

Returns

`stemgraphic.alpha.radar` (*word, comparisons, ascending=True, display=100, label=True, metric=None, min_distance=1, max_distance=None, random_state=None, sort_by='alpha'*)

The radar plot compares a reference word with a corpus. By default, it calculates the levenshtein distance between the reference word and each words in the corpus. An alternate distance or metric function can be provided. Each word is then plotted around the center based on 3 criteria.

1. If the word length is longer, it is plotted on the left side, else on the right side.
2. Distance from center is based on the distance function.
3. the words are equidistant, and their order defined alphabetically or by count (only applicable if the corpus is a text and not a list of unique words, such as a password dictionary).

Stem-and-leaf support is upcoming.

Parameters

- **word** – string, the reference word that will be placed in the middle
- **comparisons** – external file, list or string or dataframe of words
- **ascending** – bool if the sort is ascending
- **display** – maximum number of data points to display, forces sampling if smaller than `len(df)`
- **label** – bool if True display words centered at coordinate
- **metric** – Levenshtein (default), or any metric function accepting two values and returning that metric
- **min_distance** – minimum distance based on metric to include a word for display
- **max_distance** – maximum distance based on metric to include a word for display
- **random_state** – initial random seed for the sampling process, for reproducible research
- **sort_by** – default to 'alpha', can also be 'len'

Returns

`stemgraphic.alpha.scatter`(*src1*, *src2*, *alpha_only=True*, *ascending=False*, *asFigure=False*, *ax=None*, *caps=False*, *compact=True*, *display=None*, *interactive=True*, *jitter=False*, *label=False*, *leaf_order=1*, *leaf_skip=0*, *log_scale=True*, *random_state=None*, *sort_by='alpha'*, *stem_order=1*, *stem_skip=0*, *stop_words=None*)

Scatter compares the word frequency of two sources, on each axis. Each data point Z value is the word or stem-and-leaf value, while the X axis reflects that word/ngram count in one source and the Y axis reflect the same word/ngram count in the other source, in two different colors. If one word/ngram is more common on the first source it will be displayed in one color, and if it is more common in the second source, it will be displayed in a different color. The values that are the same for both sources will be displayed in a third color (default colors are blue, black and pink. In interactive mode, hovering the data point will give the precise counts on each axis along with the word itself, and filtering by category is done by clicking on the category in the legend.

Parameters

- **src1** – string, filename, url, list, numpy array, time series, pandas or dask dataframe
- **src2** – string, filename, url, list, numpy array, time series, pandas or dask dataframe
- **alpha_only** – only use stems from a-z alphabet (NA on dataframe)
- **ascending** – stem sorted in ascending order, defaults to True
- **asFigure** – return plot as plotly figure (for web applications)
- **ax** – matplotlib axes instance, usually from a figure or other plot
- **caps** – bool, True to be case sensitive, defaults to False, recommended for comparisons.(NA on dataframe)
- **compact** – do not display empty stem rows (with no leaves), defaults to False

- **display** – maximum number of data points to display, forces sampling if smaller than `len(df)`
- **interactive** – if cufflinks is loaded, renders as interactive plot in notebook
- **jitter** – random noise added to help see multiple data points sharing the same coordinate
- **label** – bool if True display words centered at coordinate
- **leaf_order** – how many leaf digits per data point to display, defaults to 1
- **leaf_skip** – how many leaf characters to skip, defaults to 0 - useful w/shared bigrams: 'wol', 'wor', 'woo'
- **log_scale** – bool if True (default) uses log scale axes
- **random_state** – initial random seed for the sampling process, for reproducible research
- **sort_by** – sort by 'alpha' (default) or 'count'
- **stem_order** – how many stem characters per data point to display, defaults to 1
- **stem_skip** – how many stem characters to skip, defaults to 0 - useful to zoom in on a single root letter
- **stop_words** – stop words to remove. None (default), list or builtin EN (English), ES (Spanish) or FR (French)

Returns matplotlib ax, dataframe with categories

```
stemgraphic.alpha.stem_freq_plot(df, alpha_only=False, asFigure=False, column=None,
                                compact=True, caps=False, display=2600, interac-
                                tive=True, kind='barh', leaf_order=1, leaf_skip=0,
                                random_state=None, stem_order=1, stem_skip=0,
                                stop_words=None)
```

Word frequency plot is the most common visualization in NLP. In this version it supports stem-and-leaf / n-grams.

Each row is the stem, and similar leaves are grouped together and each different group is stacked in bar charts.

Default is horizontal bar chart, but vertical, histograms, area charts and even pie charts are supported by this one visualization.

Parameters

- **df** – string, filename, url, list, numpy array, time series, pandas or dask dataframe
- **alpha_only** – only use stems from a-z alphabet (NA on dataframe)
- **asFigure** – return plot as plotly figure (for web applications)
- **column** – specify which column (string or number) of the dataframe to use, or group of columns (stems) else the frame is assumed to only have one column with words.
- **compact** – do not display empty stem rows (with no leaves), defaults to False
- **caps** – bool, True to be case sensitive, defaults to False, recommended for comparisons.(NA on dataframe)
- **display** – maximum number of data points to display, forces sampling if smaller than `len(df)`
- **interactive** – if cufflinks is loaded, renders as interactive plot in nebook
- **kind** – defaults to 'barh'. One of 'bar','barh','area','hist'. Non-interactive also supports 'pie'

- **leaf_order** – how many leaf digits per data point to display, defaults to 1
- **leaf_skip** – how many leaf characters to skip, defaults to 0 - useful w/shared bigrams: 'wol', 'wor', 'woo'
- **random_state** – initial random seed for the sampling process, for reproducible research
- **stem_order** – how many stem characters per data point to display, defaults to 1
- **stem_skip** – how many stem characters to skip, defaults to 0 - useful to zoom in on a single root letter
- **stop_words** – stop words to remove. None (default), list or builtin EN (English), ES (Spanish) or FR (French)

Returns

`stemgraphic.alpha.stem_graphic(df, df2=None, aggregation=True, alpha=0.1, alpha_only=True, ascending=False, ax=None, bar_color='C0', bar_outline=None, break_on=None, caps=True, column=None, combined=None, compact=False, delimiter_color='C3', display=750, figure_only=True, flip_axes=False, font_kw=None, leaf_color='k', leaf_order=1, leaf_skip=0, legend_pos='best', median_color='magenta', mirror=False, persistence=None, primary_kw=None, random_state=None, remove_accents=False, reverse=False, secondary=False, show_stem=True, sort_by='len', stop_words=None, stem_order=1, stem_skip=0, title=None, trim_blank=False, underline_color=None)`

The principal visualization of stemgraphic.alpha is `stem_graphic`. It offers all the options of `stem_text` (3.1) and adds automatic title, mirroring, flipping of axes, export (to pdf, svg, png, through `fig.savefig`) and many more options to change the visual appearance of the plot (font size, color, background color, underlining and more).

By providing a secondary text source, the plot will enable comparison through a back-to-back display

Parameters

- **df** – string, filename, url, list, numpy array, time series, pandas or dask dataframe
- **df2** – string, filename, url, list, numpy array, time series, pandas or dask dataframe (optional). for back 2 back stem-and-leaf plots
- **aggregation** – Boolean for sum, else specify function
- **alpha** – opacity of the bars, median and outliers, defaults to 10%
- **alpha_only** – only use stems from a-z alphabet (NA on dataframe)
- **ascending** – stem sorted in ascending order, defaults to True
- **ax** – matplotlib axes instance, usually from a figure or other plot
- **bar_color** – the fill color of the bar representing the leaves
- **bar_outline** – the outline color of the bar representing the leaves
- **break_on** – force a break of the leaves at that letter, the rest of the leaves will appear on the next line
- **caps** – bool, True to be case sensitive, defaults to False, recommended for comparisons.(NA on dataframe)
- **column** – specify which column (string or number) of the dataframe to use, or group of columns (stems) else the frame is assumed to only have one column with words.

- **combined** – list (specific subset to automatically include, say, for comparisons), or None
- **compact** – do not display empty stem rows (with no leaves), defaults to False
- **delimiter_color** – color of the line between aggregate and stem and stem and leaf
- **display** – maximum number of data points to display, forces sampling if smaller than `len(df)`
- **figure_only** – bool if True (default) returns matplotlib (fig,ax), False returns (fig,ax,df)
- **flip_axes** – X becomes Y and Y becomes X
- **font_kw** – keyword dictionary, font parameters
- **leaf_color** – font color of the leaves
- **leaf_order** – how many leaf digits per data point to display, defaults to 1
- **leaf_skip** – how many leaf characters to skip, defaults to 0 - useful w/shared bigrams: 'wol', 'wor', 'woo'
- **legend_pos** – One of 'top', 'bottom', 'best' or None, defaults to 'best'.
- **median_color** – color of the box representing the median
- **mirror** – mirror the plot in the axis of the delimiters
- **persistence** – filename. save sampled data to disk, either as pickle (.pkl) or csv (any other extension)
- **primary_kw** – stem-and-leaf plot additional arguments
- **random_state** – initial random seed for the sampling process, for reproducible research
- **remove_accents** – bool if True strips accents (NA on dataframe)
- **reverse** – bool if True look at words from right to left
- **secondary** – bool if True, this is a secondary plot - mostly used for back-to-back plots
- **show_stem** – bool if True (default) displays the stems
- **sort_by** – default to 'len', can also be 'alpha'
- **stem_order** – how many stem characters per data point to display, defaults to 1
- **stem_skip** – how many stem characters to skip, defaults to 0 - useful to zoom in on a single root letter
- **stop_words** – stop words to remove. None (default), list or builtin EN (English), ES (Spanish) or FR (French)
- **title** – string, or None. When None and source is a file, filename will be used.
- **trim_blank** – remove the blank between the delimiter and the first leaf, defaults to True
- **underline_color** – color of the horizontal line under the leaves, None for no display

Returns matplotlib figure and axes instance, and dataframe if figure_only is False

```
stemgraphic.alpha.stem_scatter(src1, src2, alpha_only=True, ascending=False, asFigure=False,
                               ax=None, caps=False, compact=True, display=None, interac-
                               tive=True, jitter=False, label=False, leaf_order=1, leaf_skip=0,
                               log_scale=True, random_state=None, sort_by='alpha',
                               stem_order=1, stem_skip=0, stop_words=None)
```

stem_scatter compares the word frequency of two sources, on each axis. Each data point Z value is the word or stem-and-leaf value, while the X axis reflects that word/ngram count in one source and the Y axis reflect the

same word/ngram count in the other source, in two different colors. If one word/ngram is more common on the first source it will be displayed in one color, and if it is more common in the second source, it will be displayed in a different color. The values that are the same for both sources will be displayed in a third color (default colors are blue, black and pink. In interactive mode, hovering the data point will give the precise counts on each axis along with the word itself, and filtering by category is done by clicking on the category in the legend.

Parameters

- **src1** – string, filename, url, list, numpy array, time series, pandas or dask dataframe
- **src2** – string, filename, url, list, numpy array, time series, pandas or dask dataframe
- **alpha_only** – only use stems from a-z alphabet (NA on dataframe)
- **ascending** – stem sorted in ascending order, defaults to True
- **asFigure** – return plot as plotly figure (for web applications)
- **ax** – matplotlib axes instance, usually from a figure or other plot
- **caps** – bool, True to be case sensitive, defaults to False, recommended for comparisons.(NA on dataframe)
- **compact** – do not display empty stem rows (with no leaves), defaults to False
- **display** – maximum number of data points to display, forces sampling if smaller than len(df)
- **interactive** – if cufflinks is loaded, renders as interactive plot in notebook
- **jitter** – random noise added to help see multiple data points sharing the same coordinate
- **label** – bool if True display words centered at coordinate
- **leaf_order** – how many leaf digits per data point to display, defaults to 1
- **leaf_skip** – how many leaf characters to skip, defaults to 0 - useful w/shared bigrams: 'wol', 'wor', 'woo'
- **log_scale** – bool if True uses log scale axes
- **random_state** – initial random seed for the sampling process, for reproducible research
- **sort_by** – sort by 'alpha' (default) or 'count'
- **stem_order** – how many stem characters per data point to display, defaults to 1
- **stem_skip** – how many stem characters to skip, defaults to 0 - useful to zoom in on a single root letter
- **stop_words** – stop words to remove. None (default), list or builtin EN (English), ES (Spanish) or FR (French)

Returns matplotlib polar ax, dataframe

```
stemgraphic.alpha.stem_sunburst(words, alpha_only=True, ascending=False, caps=False,
                                compact=True, display=None, hole=True, label=True,
                                leaf_order=1, leaf_skip=0, median=True, ngram=False,
                                random_state=None, sort_by='alpha', statistics=True,
                                stem_order=1, stem_skip=0, stop_words=None, top=0)
```

Stem-and-leaf based sunburst. See sunburst for details

Parameters

- **words** – string, filename, url, list, numpy array, time series, pandas or dask dataframe
- **alpha_only** – only use stems from a-z alphabet (NA on dataframe)

- **ascending** – stem sorted in ascending order, defaults to True
- **caps** – bool, True to be case sensitive, defaults to False, recommended for comparisons.(NA on dataframe)
- **compact** – do not display empty stem rows (with no leaves), defaults to False
- **display** – maximum number of data points to display, forces sampling if smaller than len(df)
- **hole** – bool if True (default) leave space in middle for statistics
- **label** – bool if True display words centered at coordinate
- **leaf_order** – how many leaf digits per data point to display, defaults to 1
- **leaf_skip** – how many leaf characters to skip, defaults to 0 - useful w/shared bigrams: 'wol', 'wor', 'woo'
- **median** – bool if True (default) display an origin and a median mark
- **ngram** – bool if True display full n-gram as leaf label
- **random_state** – initial random seed for the sampling process, for reproducible research
- **sort_by** – sort by 'alpha' (default) or 'count'
- **statistics** – bool if True (default) displays statistics in center - hole has to be True
- **stem_order** – how many stem characters per data point to display, defaults to 1
- **stem_skip** – how many stem characters to skip, defaults to 0 - useful to zoom in on a single root letter
- **stop_words** – stop words to remove. None (default), list or builtin EN (English), ES (Spanish) or FR (French)
- **top** – how many different words to count by order frequency. If negative, this will be the least frequent

Returns

`stemgraphic.alpha.stem_text(df, aggr=False, alpha_only=True, ascending=True, binary=False, break_on=None, caps=True, column=None, compact=False, display=750, legend_pos='top', leaf_order=1, leaf_skip=0, persistence=None, remove_accents=False, reverse=False, rows_only=False, sort_by='len', stem_order=1, stem_skip=0, stop_words=None, random_state=None)`

Tukey's original stem-and-leaf plot was text, with a vertical delimiter to separate stem from leaves. Just as stemgraphic implements a text version of the plot for numbers, stemgraphic.alpha implements a text version for words. This type of plot serves a similar purpose as a stacked bar chart with each data point annotated.

It also displays some basic statistics on the whole text (or subset if using column).

Parameters

- **df** – list, numpy array, time series, pandas or dask dataframe
- **aggr** – bool if True display the aggregated count of leaves by row
- **alpha_only** – only use stems from a-z alphabet (NA on dataframe)
- **ascending** – bool if the sort is ascending
- **binary** – bool if True forces counts to 1 for anything greater than 0

- **break_on** – force a break of the leaves at that letter, the rest of the leaves will appear on the next line
- **caps** – bool, True to be case sensitive, defaults to False, recommended for comparisons.(NA on dataframe)
- **column** – specify which column (string or number) of the dataframe to use, or group of columns (stems) else the frame is assumed to only have one column with words.
- **compact** – do not display empty stem rows (with no leaves), defaults to False
- **display** – maximum number of data points to display, forces sampling if smaller than len(df)
- **leaf_order** – how many leaf characters per data point to display, defaults to 1
- **leaf_skip** – how many leaf characters to skip, defaults to 0 - useful w/shared bigrams: 'wol', 'wor', 'woo'
- **legend_pos** – where to put the legend: 'top' (default), 'bottom' or None
- **persistence** – will save the sampled datafrae to filename (with csv or pkl extension) or None
- **random_state** – initial random seed for the sampling process, for reproducible research
- **remove_accents** – bool if True strips accents (NA on dataframe)
- **reverse** – bool if True look at words from right to left
- **rows_only** – by default returns only the stem and leaf rows. If false, also return the matrix and dataframe
- **sort_by** – default to 'len', can also be 'alpha'
- **stem_order** – how many stem characters per data point to display, defaults to 1
- **stem_skip** – how many stem characters to skip, defaults to 0 - useful to zoom in on a single root letter
- **stop_words** – stop words to remove. None (default), list or builtin EN (English), ES (Spanish) or FR (French)

`stemgraphic.alpha.sunburst` (*words*, *alpha_only=True*, *ascending=False*, *caps=False*, *compact=True*, *display=None*, *hole=True*, *label=True*, *leaf_order=1*, *leaf_skip=0*, *median=True*, *ngram=True*, *random_state=None*, *sort_by='alpha'*, *statistics=True*, *stem_order=1*, *stem_skip=0*, *stop_words=None*, *top=40*)

Word sunburst charts are similar to pie or donut charts, but add some statistics in the middle of the chart, including the percentage of total words targeted for a given

number of unique words (ie. top 50 words, 48% coverage).

With stem-and-leaf, the first level of the sunburst represents the stem and the second level subdivides each stem by leaves.

Parameters

- **words** – string, filename, url, list, numpy array, time series, pandas or dask dataframe
- **alpha_only** – only use stems from a-z alphabet (NA on dataframe)
- **ascending** – stem sorted in ascending order, defaults to True
- **caps** – bool, True to be case sensitive, defaults to False, recommended for comparisons.(NA on dataframe)

- **compact** – do not display empty stem rows (with no leaves), defaults to False
- **display** – maximum number of data points to display, forces sampling if smaller than `len(df)`
- **hole** – bool if True (default) leave space in middle for statistics
- **label** – bool if True display words centered at coordinate
- **leaf_order** – how many leaf digits per data point to display, defaults to 1
- **leaf_skip** – how many leaf characters to skip, defaults to 0 - useful w/shared bigrams: 'wol', 'wor', 'woo'
- **median** – bool if True (default) display an origin and a median mark
- **ngram** – bool if True (default) display full n-gram as leaf label
- **random_state** – initial random seed for the sampling process, for reproducible research
- **statistics** – bool if True (default) displays statistics in center - hole has to be True
- **sort_by** – sort by 'alpha' (default) or 'count'
- **stem_order** – how many stem characters per data point to display, defaults to 1
- **stem_skip** – how many stem characters to skip, defaults to 0 - useful to zoom in on a single root letter
- **stop_words** – stop words to remove. None (default), list or builtin EN (English), ES (Spanish) or FR (French)
- **top** – how many different words to count by order frequency. If negative, this will be the least frequent

Returns matplotlib polar ax, dataframe

```
stemgraphic.alpha.word_freq_plot(src, ascending=False, alpha_only=False, asFigure=False,  
                                caps=False, display=None, interactive=True, kind='barh',  
                                random_state=None, sort_by='count', stop_words=None,  
                                top=100)
```

word frequency bar chart.

This function creates a classical word frequency bar chart.

Parameters

- **src** – Either a filename including path, a url or a ready to process text in a dataframe or a tokenized format.
- **alpha_only** – words only if True, words and numbers if False
- **ascending** – stem sorted in ascending order, defaults to True
- **asFigure** – if interactive, the function will return a plotly figure instead of a matplotlib ax
- **caps** – keep capitalization (True, False)
- **display** – if specified, sample that quantity of words
- **interactive** – interactive graphic (True, False)
- **kind** – horizontal bar chart (barh) - also 'bar', 'area', 'hist' and non interactive 'kde' and 'pie'
- **random_state** – initial random seed for the sampling process, for reproducible research
- **sort_by** – default to 'count', can also be 'alpha'

- **stop_words** – a list of words to ignore
- **top** – how many different words to count by order frequency. If negative, this will be the least frequent

Returns text as dataframe and plotly figure or matplotlib ax

```
stemgraphic.alpha.word_radar(word, comparisons, ascending=True, display=100, label=True,
                             metric=None, min_distance=1, max_distance=None, random_state=None, sort_by='alpha')
```

Radar plot based on words. Currently, the only type of radar plot supported. See ‘[radar](#)’ for more detail.

Parameters

- **word** – string, the reference word that will be placed in the middle
- **comparisons** – external file, list or string or dataframe of words
- **ascending** – bool if the sort is ascending
- **display** – maximum number of data points to display, forces sampling if smaller than `len(df)`
- **label** – bool if True display words centered at coordinate
- **metric** – any metric function accepting two values and returning that metric in a range from 0 to x
- **min_distance** – minimum distance based on metric to include a word for display
- **max_distance** – maximum distance based on metric to include a word for display
- **random_state** – initial random seed for the sampling process, for reproducible research
- **sort_by** – default to ‘alpha’, can also be ‘len’

Returns

```
stemgraphic.alpha.word_scatter(src1, src2, alpha_only=True, ascending=False, asFigure=False, ax=None, caps=False, compact=True, display=None, interactive=True, jitter=False, label=False, leaf_order=None, leaf_skip=0, log_scale=True, random_state=None, sort_by='count', stem_order=None, stem_skip=0, stop_words=None)
```

Scatter compares the word frequency of two sources, on each axis. Each data point Z value is the word or stem-and-leaf value, while the X axis reflects that word count in one source and the Y axis reflect the same word count in the other source, in two different colors. If one word is more common on the first source it will be displayed in one color, and if it is more common in the second source, it will be displayed in a different color. The values that are the same for both sources will be displayed in a third color (default colors are blue, black and pink). In interactive mode, hovering the data point will give the precise counts on each axis along with the word itself, and filtering by category is done by clicking on the category in the legend.

Parameters

- **src1** – string, filename, url, list, numpy array, time series, pandas or dask dataframe
- **src2** – string, filename, url, list, numpy array, time series, pandas or dask dataframe
- **alpha_only** – only use stems from a-z alphabet (NA on dataframe)
- **ascending** – stem sorted in ascending order, defaults to True
- **asFigure** – return plot as plotly figure (for web applications)
- **ax** – matplotlib axes instance, usually from a figure or other plot

- **caps** – bool, True to be case sensitive, defaults to False, recommended for comparisons.(NA on dataframe)
- **compact** – do not display empty stem rows (with no leaves), defaults to False
- **display** – maximum number of data points to display, forces sampling if smaller than len(df)
- **interactive** – if cufflinks is loaded, renders as interactive plot in notebook
- **jitter** – random noise added to help see multiple data points sharing the same coordinate
- **label** – bool if True display words centered at coordinate
- **leaf_order** – how many leaf digits per data point to display, defaults to 1
- **leaf_skip** – how many leaf characters to skip, defaults to 0 - useful w/shared bigrams: 'wol', 'wor', 'woo'
- **log_scale** – bool if True (default) uses log scale axes
- **random_state** – initial random seed for the sampling process, for reproducible research
- **sort_by** – sort by 'alpha' or 'count' (default)
- **stem_order** – how many stem characters per data point to display, defaults to 1
- **stem_skip** – how many stem characters to skip, defaults to 0 - useful to zoom in on a single root letter
- **stop_words** – stop words to remove. None (default), list or builtin EN (English), ES (Spanish) or FR (French)

Returns matplotlib polar ax, dataframe

`stemgraphic.alpha.word_sunburst(words, alpha_only=True, ascending=False, caps=False, compact=True, display=None, hole=True, label=True, leaf_order=None, leaf_skip=0, median=True, ngram=True, random_state=None, sort_by='alpha', statistics=True, stem_order=None, stem_skip=0, stop_words=None, top=40)`

Word based sunburst. See sunburst for details

Parameters

- **words** – string, filename, url, list, numpy array, time series, pandas or dask dataframe
- **alpha_only** – only use stems from a-z alphabet (NA on dataframe)
- **ascending** – stem sorted in ascending order, defaults to True
- **caps** – bool, True to be case sensitive, defaults to False, recommended for comparisons.(NA on dataframe)
- **compact** – do not display empty stem rows (with no leaves), defaults to False
- **display** – maximum number of data points to display, forces sampling if smaller than len(df)
- **hole** – bool if True (default) leave space in middle for statistics
- **label** – bool if True display words centered at coordinate
- **leaf_order** – how many leaf digits per data point to display, defaults to 1
- **leaf_skip** – how many leaf characters to skip, defaults to 0 - useful w/shared bigrams: 'wol', 'wor', 'woo'
- **median** – bool if True (default) display an origin and a median mark

- **ngram** – bool if True (default) display full n-gram as leaf label
- **random_state** – initial random seed for the sampling process, for reproducible research
- **statistics** – bool if True (default) displays statistics in center - hole has to be True
- **sort_by** – sort by ‘alpha’ (default) or ‘count’
- **stem_order** – how many stem characters per data point to display, defaults to 1
- **stem_skip** – how many stem characters to skip, defaults to 0 - useful to zoom in on a single root letter
- **stop_words** – stop words to remove. None (default), list or builtin EN (English), ES (Spanish) or FR (French)
- **top** – how many different words to count by order frequency. If negative, this will be the least frequent

Returns

8.4 graphic

Stemgraphic.graphic

Stemgraphic provides a complete set of functions to handle everything related to stem-and-leaf plots. Stemgraphic.graphic is a module implementing a graphical stem-and-leaf plot function and a stem-and-leaf heatmap plot function for numerical data.

```
stemgraphic.graphic.heatmap(df, annotate=False, asFigure=False, ax=None, column=None,
                             compact=False, display=900, interactive=True, persistence=None,
                             random_state=None, scale=None, trim=False, trim_blank=True,
                             unit="", zoom=None)
```

The heatmap displays the same underlying data as the stem-and-leaf plot, but instead of stacking the leaves, they are left in their respective columns. Row ‘42’ and Column ‘7’ would have the count of numbers starting with ‘427’ of the given scale.

The heatmap is useful to look at patterns. For distribution, stem_graphic is better suited.

Parameters

- **df** – list, numpy array, time series, pandas or dask dataframe
- **annotate** – display annotations (Z) on heatmap
- **asFigure** – return plot as plotly figure (for web applications)
- **ax** – matplotlib axes instance, usually from a figure or other plot
- **column** – specify which column (string or number) of the dataframe to use, else the first numerical is selected
- **compact** – do not display empty stem rows (with no leaves), defaults to False
- **display** – maximum number of data points to display, forces sampling if smaller than len(df)
- **interactive** – if cufflinks is loaded, renders as interactive plot in notebook
- **persistence** – filename. save sampled data to disk, either as pickle (.pkl) or csv (any other extension)

- **random_state** – initial random seed for the sampling process, for reproducible research
- **scale** – force a specific scale for building the plot. Defaults to None (automatic).
- **trim** – ranges from 0 to 0.5 (50%) to remove from each end of the data set, defaults to None
- **trim_blank** – remove the blank between the delimiter and the first leaf, defaults to True
- **unit** – specify a string for the unit ('\$', 'Kg'...). Used for outliers and for legend, defaults to ""
- **zoom** – zoom level, on top of calculated scale (+1, -1 etc)

Returns count matrix, scale and matplotlib ax or figure if interactive and asFigure are True

```
stemgraphic.graphic.stem_graphic(df, alpha=0.15, aggregation=True, asc=True, ax=None,
                                  bar_color='C0', bar_outline=None, break_on=None,
                                  column=None, compact=False, delimiter_color='C3', display=900,
                                  flip_axes=False, font_kw=None, leaf_color='k',
                                  leaf_order=1, legend_pos='best', median_color='magenta',
                                  mirror=False, outliers=None, outliers_color='r', persistence=None,
                                  primary_kw=None, random_state=None, scale=None,
                                  secondary_kw=None, secondary_plot=None, trim=False,
                                  trim_blank=True, underline_color=None, unit="", zoom=None)
```

A graphical stem and leaf plot. stem_graphic provides horizontal, vertical or mirrored layouts, sorted in ascending or descending order, with sane default settings for the visuals, legend, median and outliers.

Parameters

- **df** – list, numpy array, time series, pandas or dask dataframe
- **aggregation** – Boolean for sum, else specify function
- **alpha** – opacity of the bars, median and outliers, defaults to 15%
- **asc** – stem sorted in ascending order, defaults to True
- **ax** – matplotlib axes instance, usually from a figure or other plot
- **bar_color** – the fill color of the bar representing the leaves
- **bar_outline** – the outline color of the bar representing the leaves
- **break_on** – force a break of the leaves at x in (5, 10), defaults to 10
- **column** – specify which column (string or number) of the dataframe to use, else the first numerical is selected
- **compact** – do not display empty stem rows (with no leaves), defaults to False
- **delimiter_color** – color of the line between aggregate and stem and stem and leaf
- **display** – maximum number of data points to display, forces sampling if smaller than len(df)
- **flip_axes** – X becomes Y and Y becomes X
- **font_kw** – keyword dictionary, font parameters
- **leaf_color** – font color of the leaves
- **leaf_order** – how many leaf digits per data point to display, defaults to 1
- **legend_pos** – One of 'top', 'bottom', 'best' or None, defaults to 'best'.

- **median_color** – color of the box representing the median
- **mirror** – mirror the plot in the axis of the delimiters
- **outliers** – display outliers - these are from the full data set, not the sample. Defaults to Auto
- **outliers_color** – background color for the outlier boxes
- **persistence** – filename. save sampled data to disk, either as pickle (.pkl) or csv (any other extension)
- **primary_kw** – stem-and-leaf plot additional arguments
- **random_state** – initial random seed for the sampling process, for reproducible research
- **scale** – force a specific scale for building the plot. Defaults to None (automatic).
- **secondary_kw** – any matplotlib keyword supported by .plot(), for the secondary plot
- **secondary_plot** – One or more of ‘dot’, ‘kde’, ‘margin_kde’, ‘rug’ in a comma delimited string or None
- **trim** – ranges from 0 to 0.5 (50%) to remove from each end of the data set, defaults to None
- **trim_blank** – remove the blank between the delimiter and the first leaf, defaults to True
- **underline_color** – color of the horizontal line under the leaves, None for no display
- **unit** – specify a string for the unit (‘\$’, ‘Kg’...). Used for outliers and for legend, defaults to “
- **zoom** – zoom level, on top of calculated scale (+1, -1 etc)

Returns matplotlib figure and axes instance

8.5 helpers

helpers.py

Helper functions for stemgraphic.

```
stemgraphic.helpers.APOSTROPHE = '''
    Typographical apostrophe - ex: I'm, l'arbre
```

```
stemgraphic.helpers.CHAR_FILTER = ['\t', '\n', '\\', '/', '`', '*', '_', '{', '}', '[', ']'
    Characters to filter. Does a relatively good job on a majority of texts ‘- ‘ and ‘-’ is to skip quotes in many plays
    and dialogues in books, especially French.
```

```
stemgraphic.helpers.DOUBLE_QUOTE = '''
    Double straight quote mark
```

```
stemgraphic.helpers.LETTERS = 'abcdefghijklmnopqrstuvwxyz'
    Default definition of standard letters remove_accent has to be called explicitly for any of these letters to match
    their accented counterparts
```

```
stemgraphic.helpers.NON_ALPHA = ['-', '+', '/', '[', ']', '_', '£', '1', '2', '3', '4', '5'
    List of non alpha characters. Temporary - I want to balance flexibility with convenience, but still looking at
    options.
```

`stemgraphic.helpers.NO_PERIOD_FILTER = ['\t', '\n', '\\', '/', '`', '*', '_', '{', '}', '[`
Similar purpose to `CHAR_FILTER`, it keeps the period. The last word of each sentence will end with a `'.`
Useful for manipulating the dataframe returned by the various visualizations and `ngram_data`, to break down frequencies by sentence instead of the full text or list.

`stemgraphic.helpers.QUOTE = ""`
Straight quote mark - ex: `'INCONCEIVABLE'`

`stemgraphic.helpers.key_calc(stem, leaf, scale)`
Calculates a value from a stem, a leaf and a scale.

Parameters

- `stem` –
- `leaf` –
- `scale` –

Returns calculated values

`stemgraphic.helpers.legend(ax, x, y, asc, flip_axes, mirror, stem, leaf, scale, delimiter_color, aggregation=True, cur_font=None, display=10, pos='best', unit=")`
Builds a graphical legend for numerical stem-and-leaf plots.

Parameters

- `display` –
- `cur_font` –
- `ax` –
- `x` –
- `y` –
- `pos` –
- `asc` –
- `flip_axes` –
- `mirror` –
- `stem` –
- `leaf` –
- `scale` –
- `delimiter_color` –
- `unit` –
- `aggregation` –

`stemgraphic.helpers.min_max_count(x, column=0)`
Handles min, max and count. This works on numpy, lists, pandas and dask dataframes.

Parameters

- `column` –
- `x` – list, numpy array, series, pandas or dask dataframe

Returns min, max and count

`stemgraphic.helpers.percentile(data, alpha)`

Parameters

- **data** – list, numpy array, time series or pandas dataframe
- **alpha** – between 0 and 0.5 proportion to select on each side of the distribution

Returns the actual value at that percentile

`stemgraphic.helpers.stack_columns(row)`

stack multiple columns into a single stacked value :param row: a row of letters :return: stacked string

8.6 num

`stemgraphic.num.`

BRAND NEW in V.0.5.0!

Stemgraphic provides a complete set of functions to handle everything related to stem-and-leaf plots. `num` is a module of the stemgraphic package to handle numerical variables.

This module structure is new as of v.0.5.0 to match the addition of `stemgraphic.alpha`.

The shorthand from previous versions of stemgraphic is still available and defaults to the numerical functions:

```
from stemgraphic import stem_graphic, stem_text, heatmap
```

8.7 stopwords

`stopwords`

This module includes 4 lists of stop words: EN (main English list), ALT_EN (alternate English list), FR (French) and SP (Spanish).

A PT (Portuguese) list is in the works.

```
stemgraphic.stopwords.ALT_EN = ['a', 'am', 'an', 'and', 'are', 'as', 'at', 'been', 'for',
    ALT_ENglish stopwords
```

```
stemgraphic.stopwords.EN = ['a', 'about', 'above', 'across', 'after', 'afterwards', 'again',
    English stop words
```

```
stemgraphic.stopwords.ES = ['a', 'alguna', 'algunas', 'alguno', 'algunos', 'algún', 'ambas',
    Spanish (ESpanol) stop words
```

```
stemgraphic.stopwords.FR = ['a', 'alors', 'au', 'aucuns', 'aussi', 'autre', 'autres', 'aux',
    French (FRancais) stop words
```

```
stemgraphic.stopwords.VOCALES = ['a', 'á', 'e', 'é', 'i', 'í', 'o', 'ó', 'u', 'ú', 'ü']
    Spanish vowels
```

```
stemgraphic.stopwords.VOWELS = ['a', 'e', 'i', 'o', 'u']
    English vowels
```

```
stemgraphic.stopwords.VOYELLES = ['a', 'â', 'ä', 'à', 'æ', 'e', 'ê', 'ë', 'é', 'è', 'i', 'ï', 'j', 'k', 'l', 'm', 'n', 'o', 'ô', 'ö', 'ò', 'p', 'q', 'r', 's', 't', 'u', 'ü', 'v', 'w', 'x', 'y', 'z']
    French vowels
```

8.8 text

`stemgraphic.text.stem_data` (*x*, *column=None*, *display=300*, *leaf_order=1*, *break_on=None*, *scale=None*, *compact=False*, *omin=None*, *omax=None*, *outliers=False*, *persistence=None*, *random_state=None*, *total_rows=None*, *trim=False*, *zoom=None*)

Returns scale factor, key label and list of rows.

Parameters

- **x** – list, numpy array, time series, pandas or dask dataframe
- **column** – specify which column (string or number) of the dataframe to use, else the first numerical is selected
- **display** –
- **leaf_order** –
- **break_on** –
- **scale** –
- **compact** –
- **outliers** –
- **omin** –
- **omax** –
- **persistence** – persist sampled dataframe
- **random_state** –
- **total_rows** –
- **trim** –
- **zoom** –

`stemgraphic.text.stem_dot` (*df*, *asc=True*, *break_on=None*, *column=None*, *compact=False*, *display=300*, *legend_pos='best'*, *outliers=True*, *random_state=None*, *scale=None*, *trim=False*, *unit=""*, *zoom=None*)

Parameters

- **df** – list, numpy array, time series, pandas or dask dataframe
- **asc** –
- **break_on** –
- **column** – specify which column (string or number) of the dataframe to use, else the first numerical is selected
- **compact** –
- **display** –
- **legend_pos** –
- **outliers** –
- **random_state** –
- **scale** –

- **trim** –
- **unit** –
- **zoom** –

`stemgraphic.text.stem_text(df, asc=True, break_on=None, column=None, compact=False, display=300, legend_pos='best', outliers=True, persistence=None, random_state=None, scale=None, trim=False, unit="", zoom=None)`

Parameters

- **df** – list, numpy array, time series, pandas or dask dataframe
- **asc** –
- **break_on** –
- **column** – specify which column (string or number) of the dataframe to use, else the first numerical is selected
- **compact** –
- **display** –
- **legend_pos** –
- **outliers** –
- **persistence** –
- **random_state** –
- **scale** –
- **trim** –
- **unit** –
- **zoom** –

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`

PYTHON MODULE INDEX

S

- `stemgraphic.__init__`, [17](#)
- `stemgraphic.aliases`, [17](#)
- `stemgraphic.alpha`, [18](#)
- `stemgraphic.graphic`, [33](#)
- `stemgraphic.helpers`, [35](#)
- `stemgraphic.num`, [37](#)
- `stemgraphic.stopwords`, [37](#)
- `stemgraphic.text`, [38](#)

A

add_missing_letters() (in module stemgraphic.alpha), 19
 ALT_EN (in module stemgraphic.stopwords), 37
 APOSTROPHE (in module stemgraphic.helpers), 35

C

CHAR_FILTER (in module stemgraphic.helpers), 35

D

DOUBLE_QUOTE (in module stemgraphic.helpers), 35

E

EN (in module stemgraphic.stopwords), 37
 ES (in module stemgraphic.stopwords), 37

F

FR (in module stemgraphic.stopwords), 37

H

heatmap() (in module stemgraphic.alpha), 19
 heatmap() (in module stemgraphic.graphic), 33
 heatmap_grid() (in module stemgraphic.alpha), 19

K

key_calc() (in module stemgraphic.helpers), 36

L

legend() (in module stemgraphic.helpers), 36
 LETTERS (in module stemgraphic.helpers), 35

M

matrix_difference() (in module stemgraphic.alpha), 20
 min_max_count() (in module stemgraphic.helpers), 36

N

ngram_data() (in module stemgraphic.alpha), 21
 NO_PERIOD_FILTER (in module stemgraphic.helpers), 35
 NON_ALPHA (in module stemgraphic.helpers), 35

P

percentile() (in module stemgraphic.helpers), 36
 plot_sunburst_level() (in module stemgraphic.alpha), 22
 polar_word_plot() (in module stemgraphic.alpha), 22

Q

QUOTE (in module stemgraphic.helpers), 36

R

radar() (in module stemgraphic.alpha), 22

S

scatter() (in module stemgraphic.alpha), 23
 stack_columns() (in module stemgraphic.helpers), 37
 stem_data() (in module stemgraphic.text), 38
 stem_dot() (in module stemgraphic.text), 38
 stem_freq_plot() (in module stemgraphic.alpha), 24
 stem_graphic() (in module stemgraphic.alpha), 25
 stem_graphic() (in module stemgraphic.graphic), 34
 stem_hist() (in module stemgraphic.aliaes), 17
 stem_kde() (in module stemgraphic.aliaes), 18
 stem_line() (in module stemgraphic.aliaes), 18
 stem_scatter() (in module stemgraphic.alpha), 26
 stem_sunburst() (in module stemgraphic.alpha), 27
 stem_text() (in module stemgraphic.alpha), 28
 stem_text() (in module stemgraphic.text), 39
 stemgraphic.__init__ (module), 17
 stemgraphic.aliaes (module), 17
 stemgraphic.alpha (module), 18
 stemgraphic.graphic (module), 33
 stemgraphic.helpers (module), 35
 stemgraphic.num (module), 37
 stemgraphic.stopwords (module), 37
 stemgraphic.text (module), 38
 sunburst() (in module stemgraphic.alpha), 29

V

VOCALES (in module stemgraphic.stopwords), 37
 VOWELS (in module stemgraphic.stopwords), 37
 VOYELLES (in module stemgraphic.stopwords), 37

W

`word_freq_plot()` (in module `stemgraphic.alpha`), [30](#)

`word_radar()` (in module `stemgraphic.alpha`), [31](#)

`word_scatter()` (in module `stemgraphic.alpha`), [31](#)

`word_sunburst()` (in module `stemgraphic.alpha`), [32](#)