



# cachai

## Python Meeting #12

*CACHAI, a customizable visualization toolkit for science*

 [DD-Beltran-F/cachai](https://github.com/DD-Beltran-F/cachai)

 [pypi.org/project/cachai](https://pypi.org/project/cachai)

 [cachai.readthedocs.io](https://cachai.readthedocs.io)



# What is cachai?

---

CACHAI stands for “**C**ustom **A**xes and **CH**arts **A**dvanced **I**nterface”. It is a Python package for **data visualization**, a fully customizable toolkit designed to produce polished, publication-ready plots built on top of *matplotlib*.

matplotlib

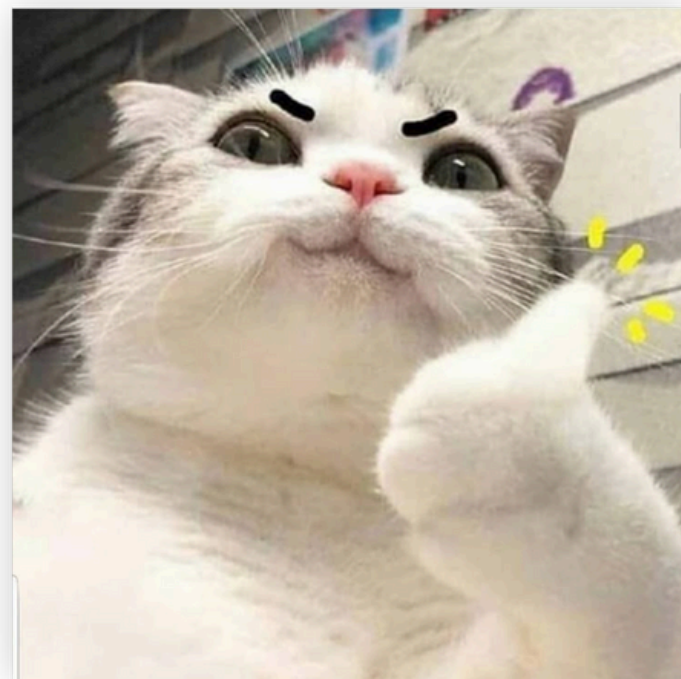




# What is cachai?

---

CACHAI stands for “**C**ustom **A**xes and **CH**arts **A**dvanced **I**nterface”. It is a Python package for data visualization, a fully customizable toolkit designed to produce polished, publication-ready plots built on top of *matplotlib*.



*We want to give you access to creative, unusual (and even funny) plots!*



# Installing cachai

---

All official releases of CACHAI are published on **PyPI**.  
To install, simply run this in your console:

```
user@host:~$ pip install cachai
```



# Installing cachai

---

All official releases of CACHAI are published on [PyPI](#).  
To install, simply run this in your console:

```
user@host:~$ pip install cachai
```

You can also install it with the optional testing dependencies:

```
user@host:~$ pip install cachai[testing]
```



# Modules of cachai

## Plotting

`cachai.chplot`

References all of cachai's charts, providing a matplotlib-like interface for intuitive and easy plotting.

```
chord(...)
```

```
michinoff(...)
```

```
polartext(...)
```

```
...
```

## Datasets

`cachai.data`

Allows you to access pre-created datasets hosted in the *cachai-datasets* [GitHub](#).

```
load_dataset(...)
```

```
get_dataset_repo()
```

```
get_dataset_names()
```

```
get_dataset_metadata(...)
```

```
clear_cache(...)
```

## Utilities

`cachai.utilities`

Contains supporting tools primarily designed for internal use, though some functions may be useful in other contexts.

```
chsave(...)
```

```
map_from_curve(...)
```

```
get_bezier_curve(...)
```

```
mod_color(...)
```

```
...
```

## Gadgets

`cachai.gadgets`

Some additional graphical utilities.

```
PolarText(...)
```

```
...
```

## Tests

Verify the installation of cachai.

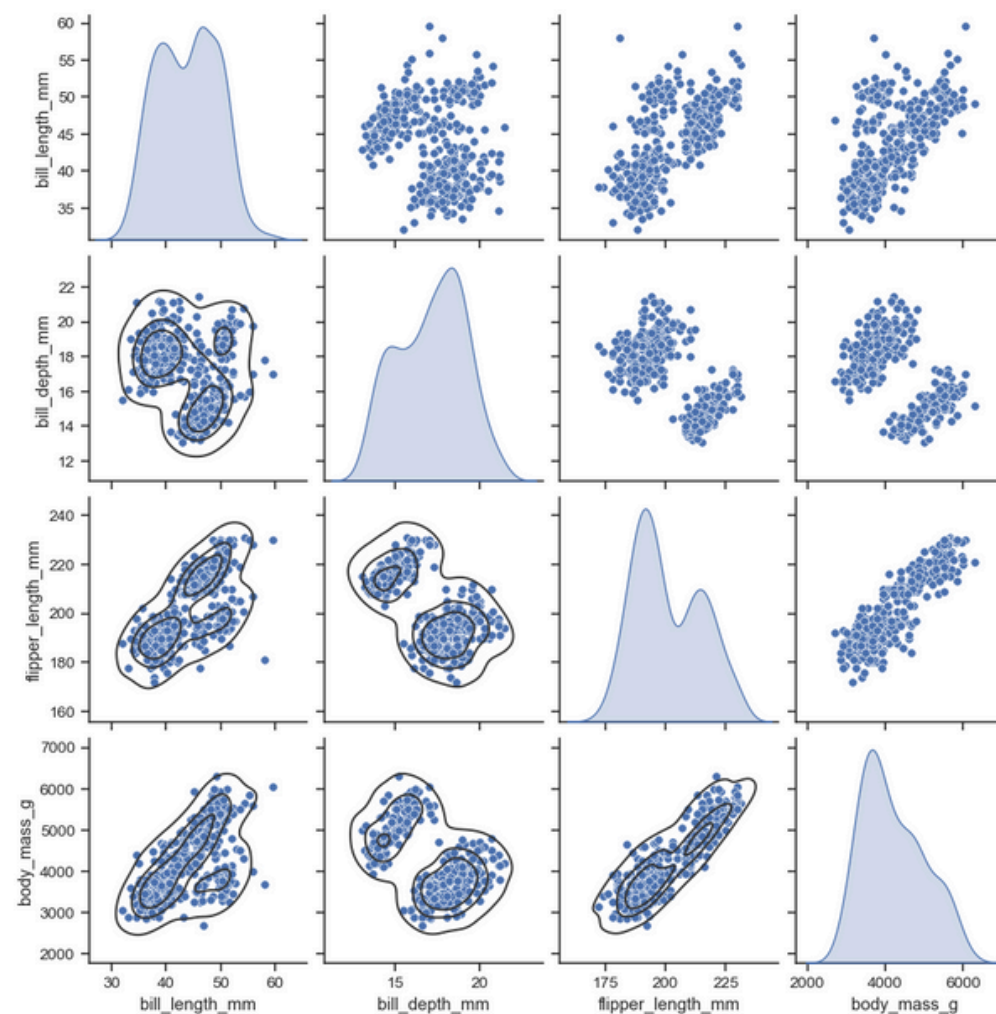
```
get_available_tests()
```

```
run_tests(...)
```

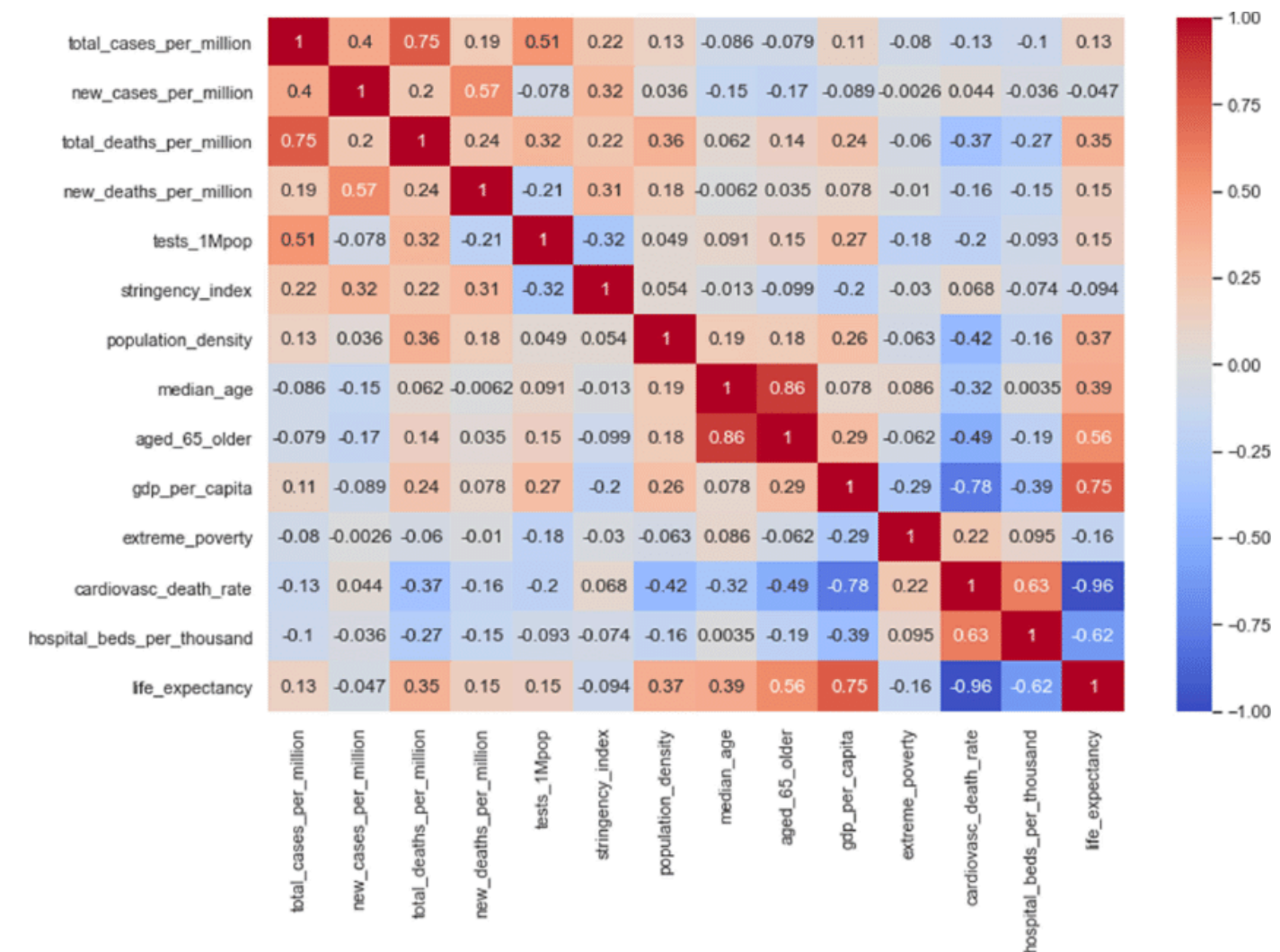


# Chord Diagrams

A Chord Diagram visualize correlations. Common ways to do this include **Pair/Corner Plots** and **Heatmaps**. However, these can quickly become overcrowded (and take up a lot of space in a manuscript!) when dealing with many variables.



seaborn documentation (2025)



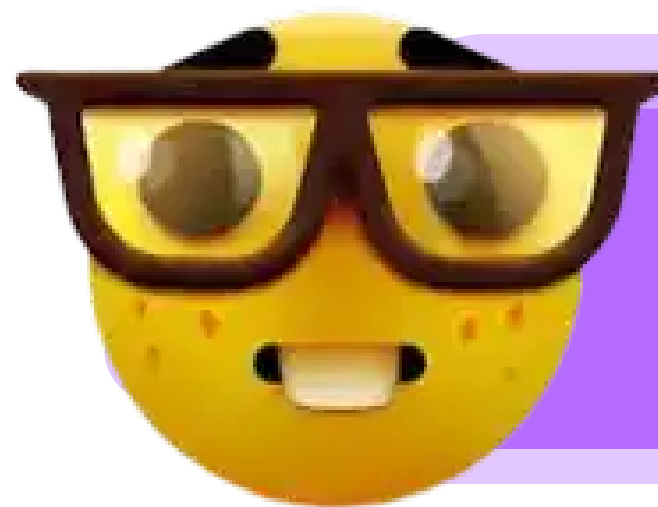
Music et al. (2021)



# Chord Diagrams

---

A Chord Diagram visualize correlations. Common ways to do this include **Pair/Corner Plots** and **Heatmaps**. However, these can quickly become overcrowded (and take up a lot of space in a manuscript!) when dealing with many variables.

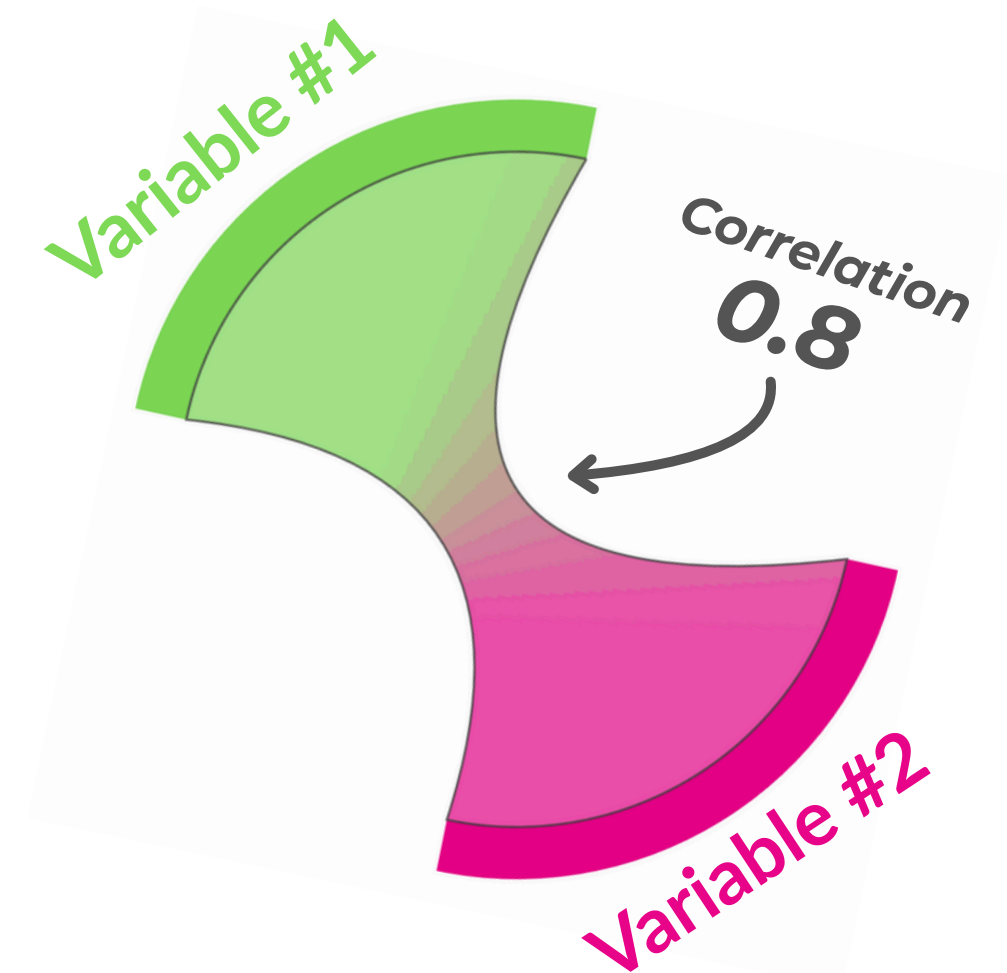
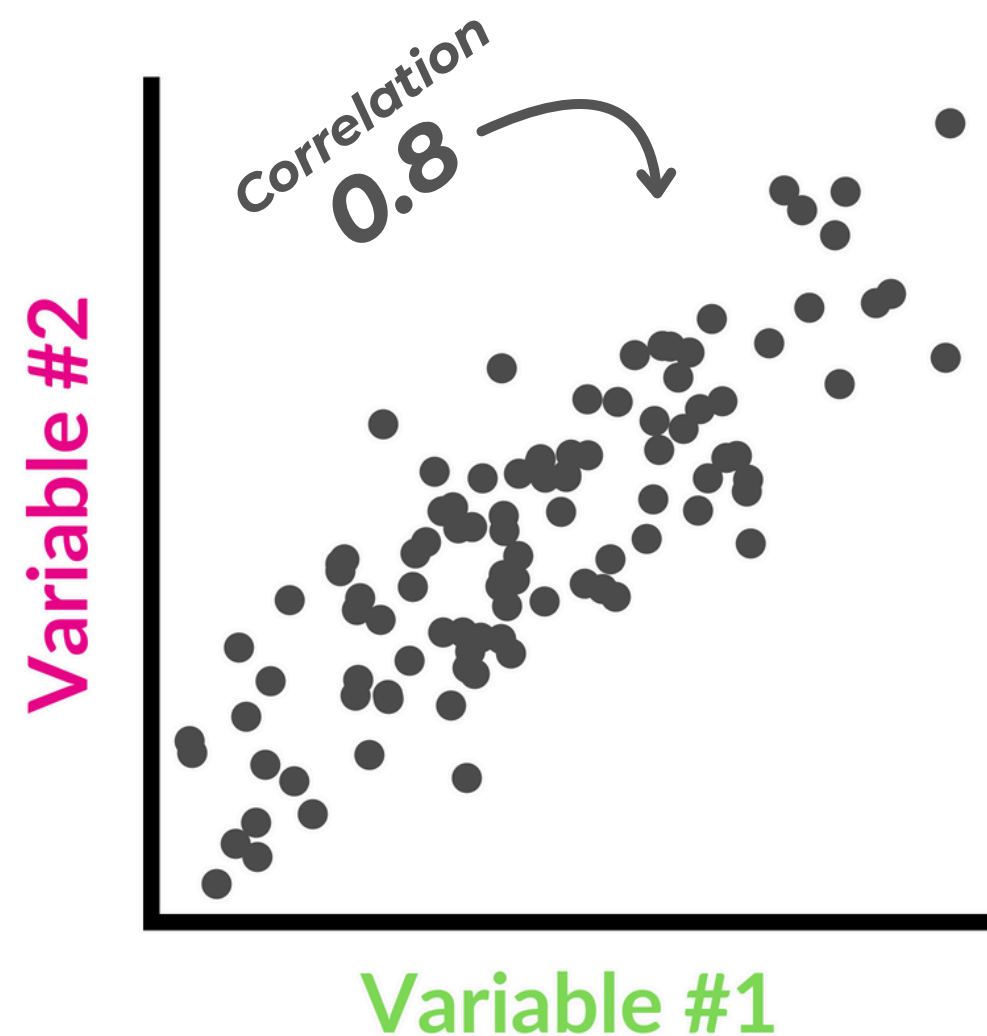


**Chord diagrams** solve this problem by providing a **compact** and **intuitive** view of all relationships at once.



# Chord Diagrams

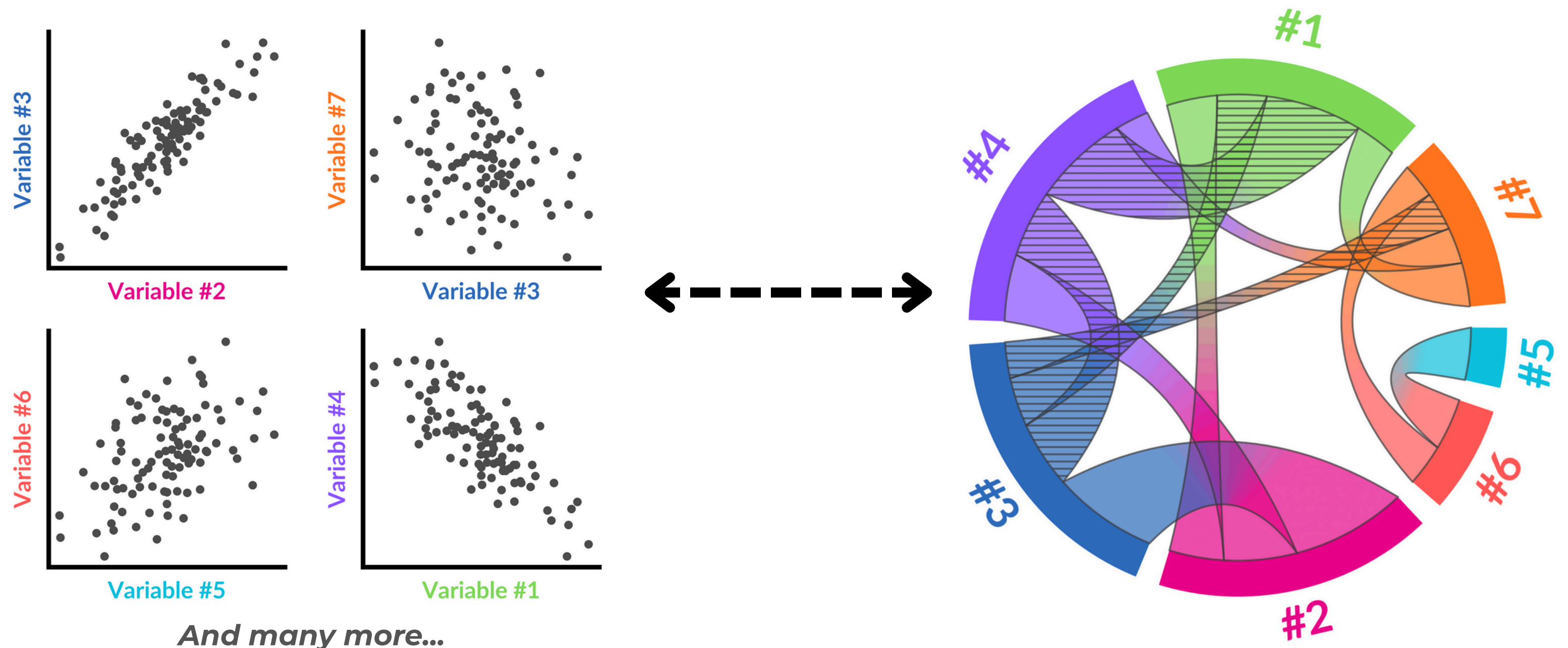
A **Chord Diagram** visualize correlations between multiple variables by representing them as nodes arranged around a circle, and links (chords) connecting them.





# Chord Diagrams

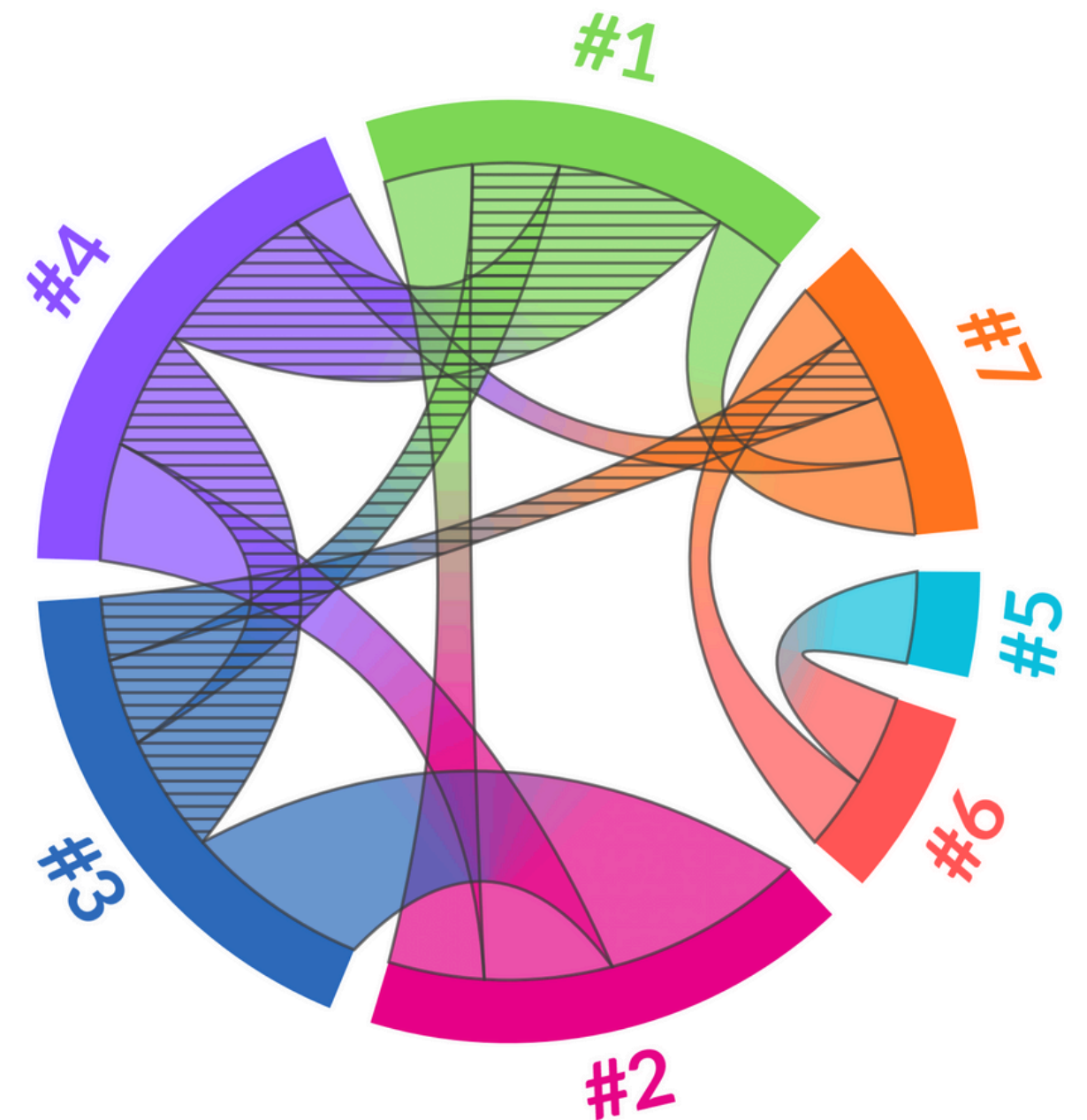
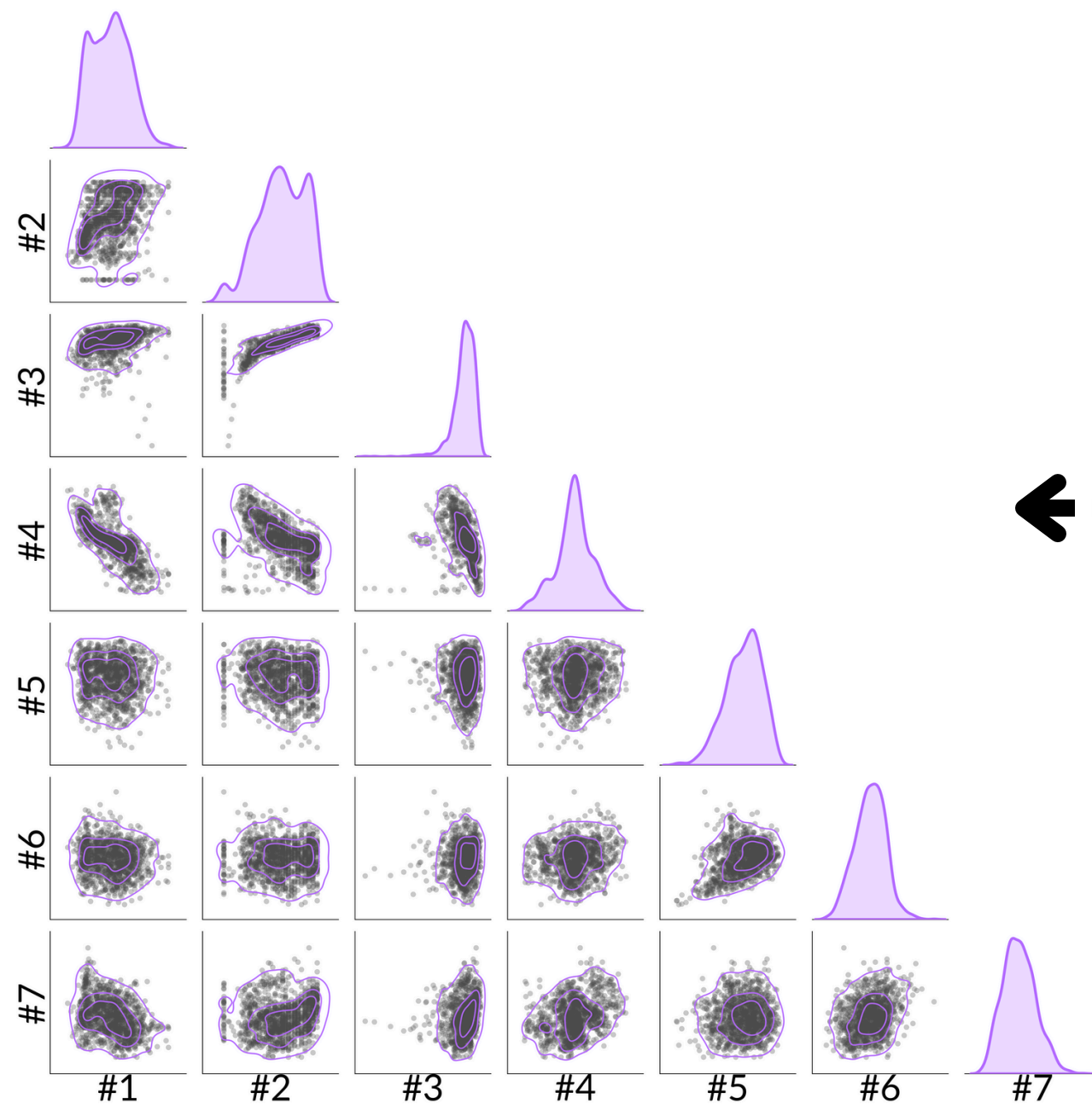
Each chord's **thickness** and **pattern** (hatch) indicate the strength and sign of the relationship (e.g. correlation or anti-correlation).





# Chord Diagrams

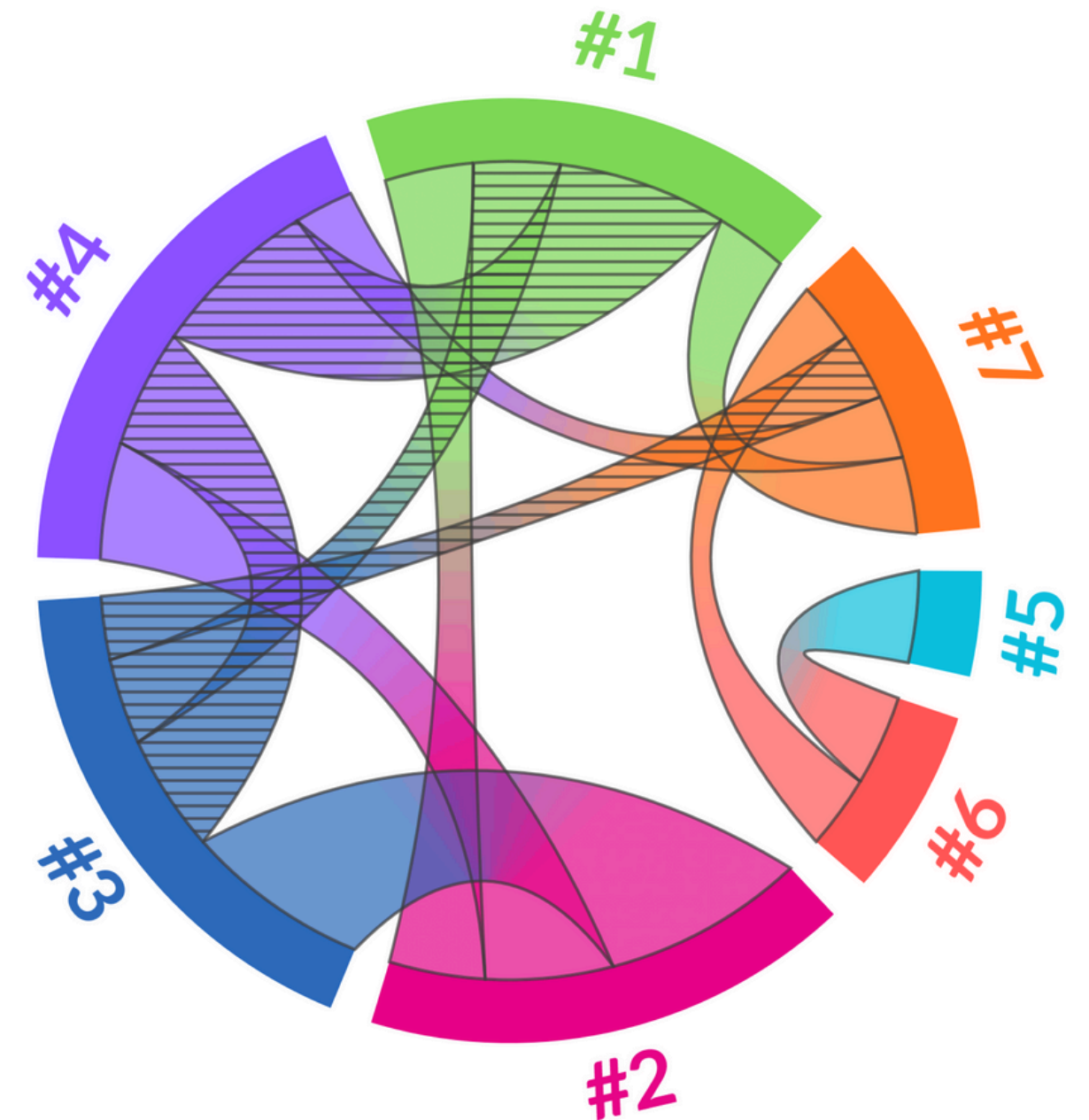
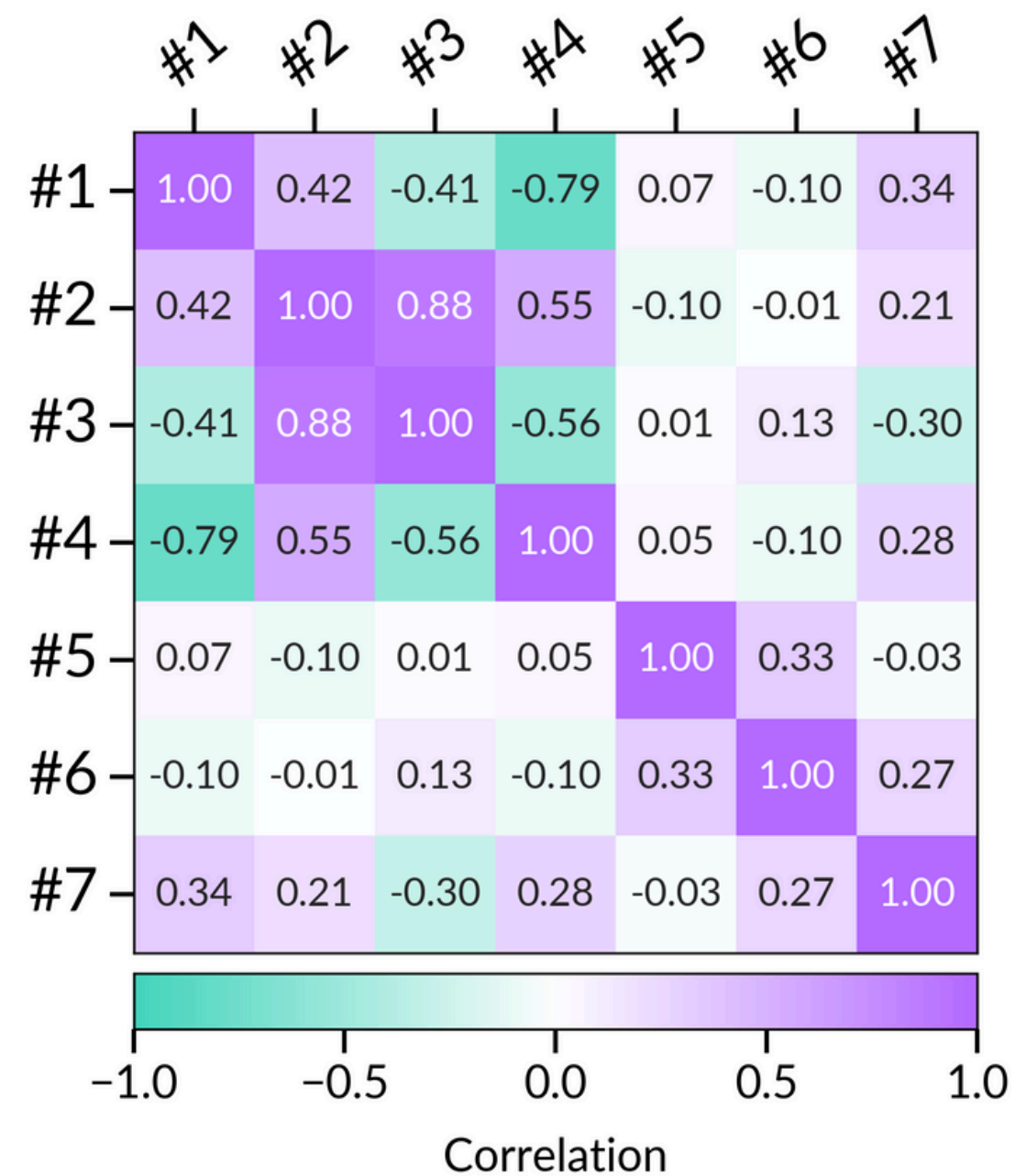
Each chord's **thickness** and **pattern** (hatch) indicate the strength and sign of the relationship (e.g. correlation or anti-correlation).





# Chord Diagrams

Each chord's **thickness** and **pattern** (hatch) indicate the strength and sign of the relationship (e.g. correlation or anti-correlation).





# Chord Diagrams: Basic usage

notebook.ipynb ×

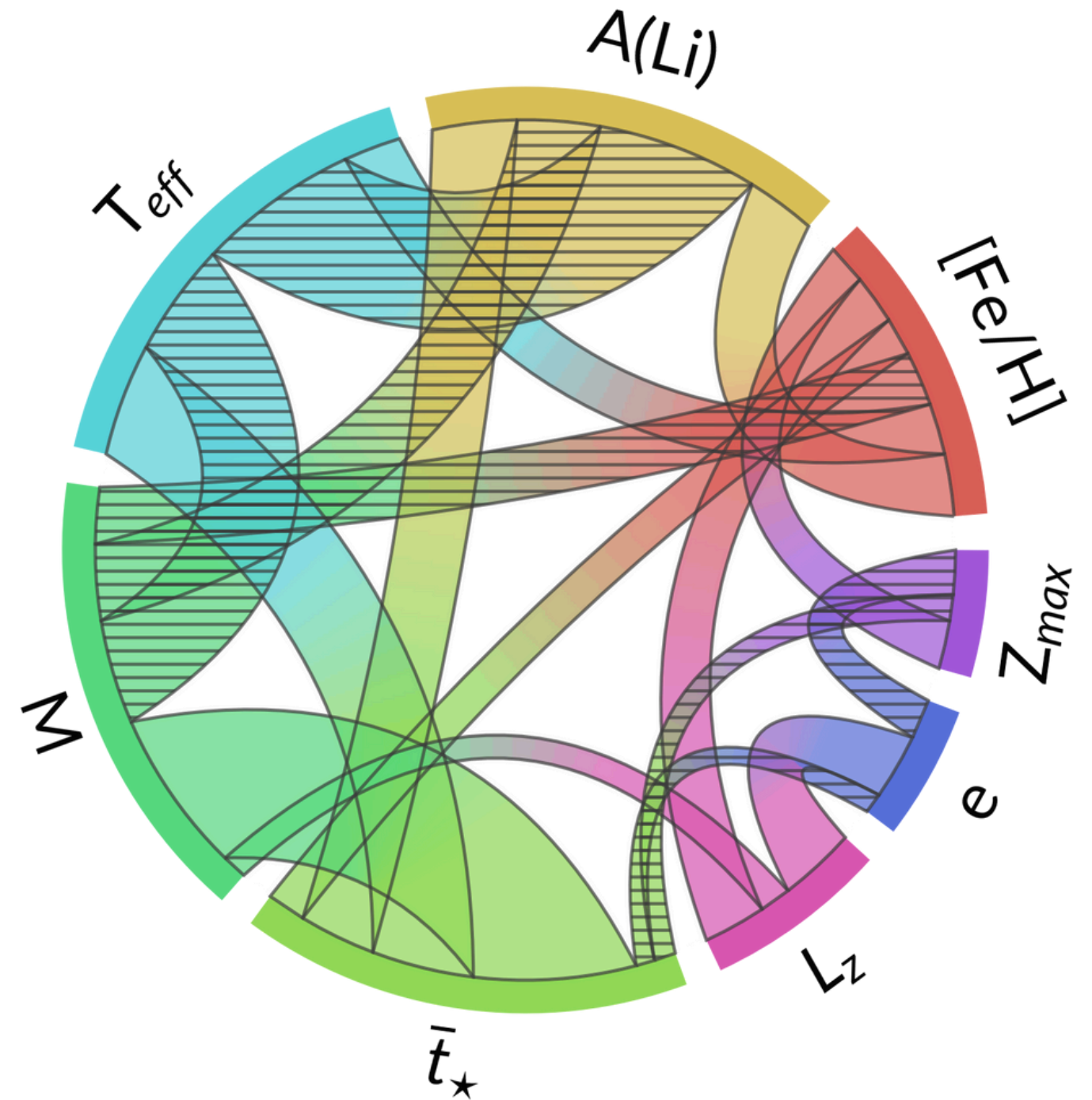
```
# Some imports
import cachai.data as chd
import cachai.chplot as chp

# Let's use a cachai dataset
data = chd.load_dataset('lithium')

# Now we calculate the correlation
# matrix
corr_matrix = data.corr('spearman')

# And that's it! Now we plot it
chp.chord(corr_matrix)
```

Output



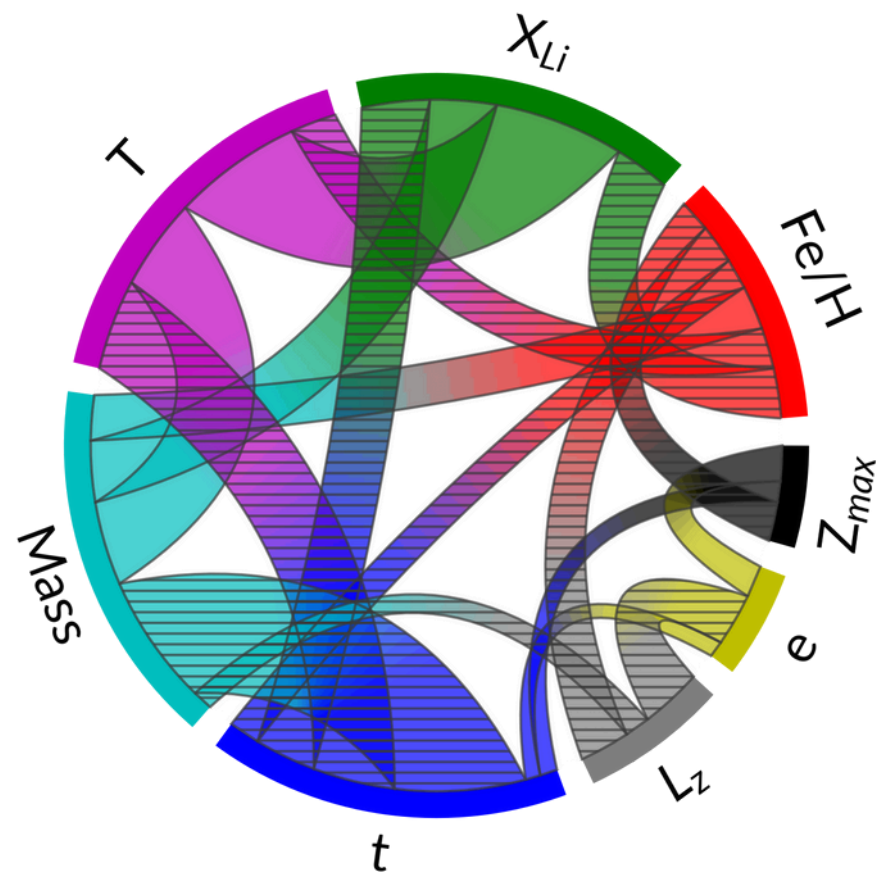


# Chord Diagrams: Customization

notebook.ipynb ×

```
# Basic customization includes variable names and colors
names = ['Fe/H', r' $X_{Li}$ ', 't', 'Mass', 'T', 'e', r' $Z_{max}$ ', r' $L_z$ ']
colors = ['r', 'g', 'b', 'c', 'm', 'y', 'k', 'gray']
chp.chord(corr_matrix, names, colors)
```

Output



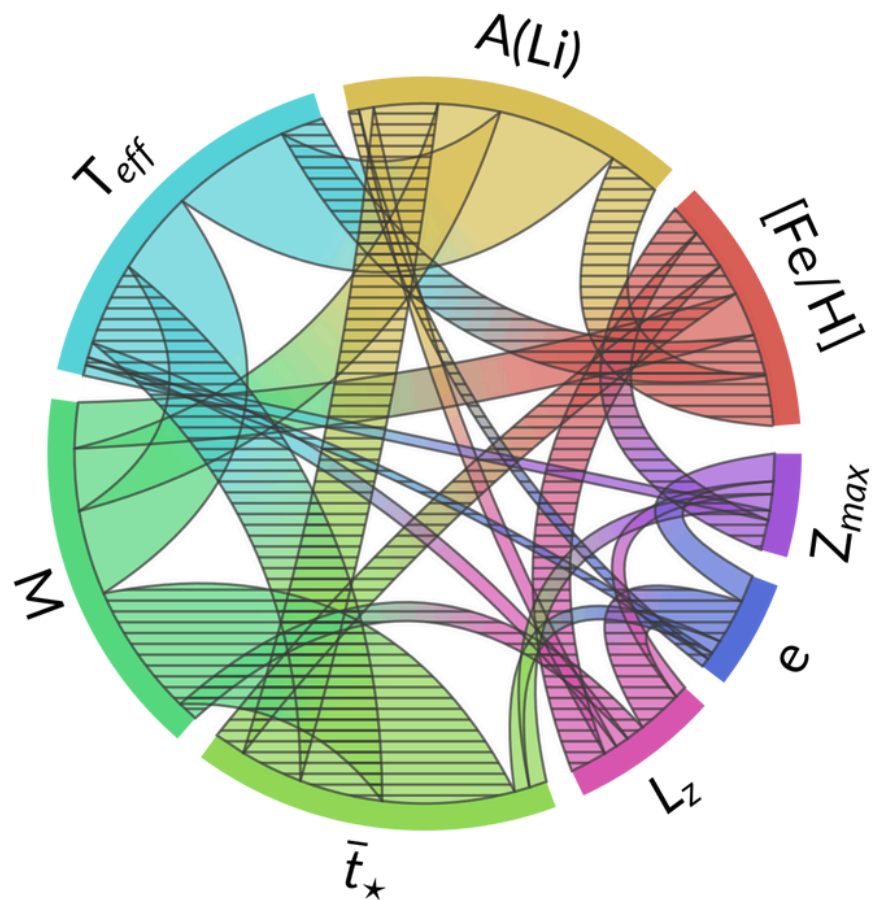


# Chord Diagrams: Customization

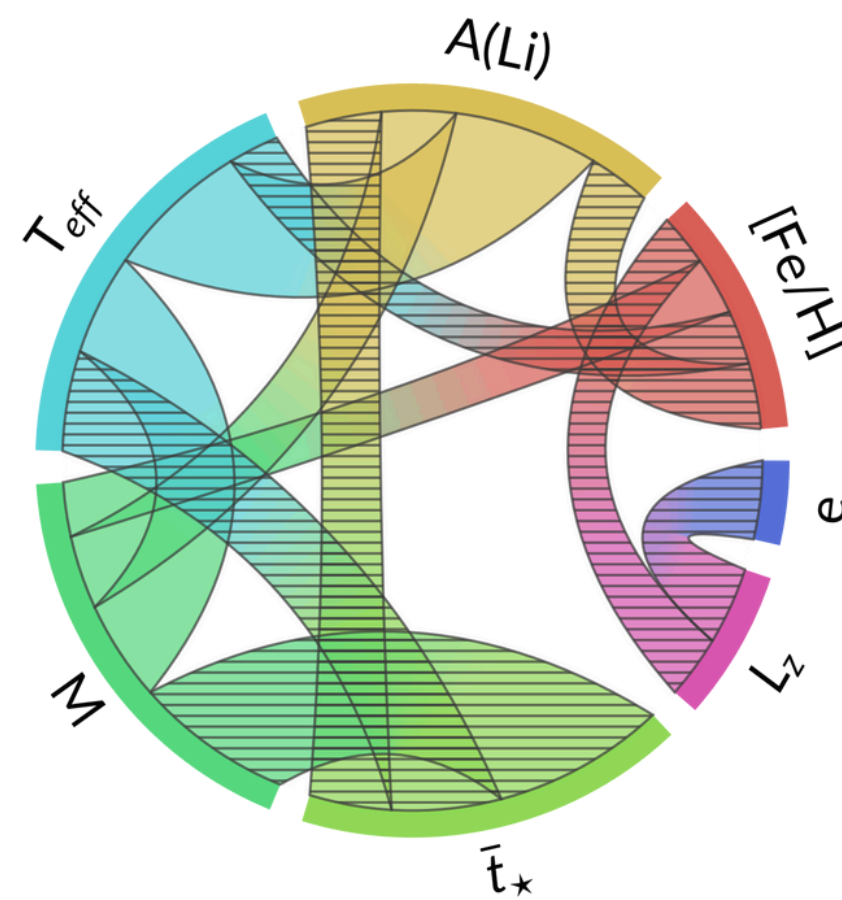
notebook.ipynb ×

```
# threshold / th : Minimum correlation value to display  
chp.chord(corr_matrix, threshold=your_threshold)
```

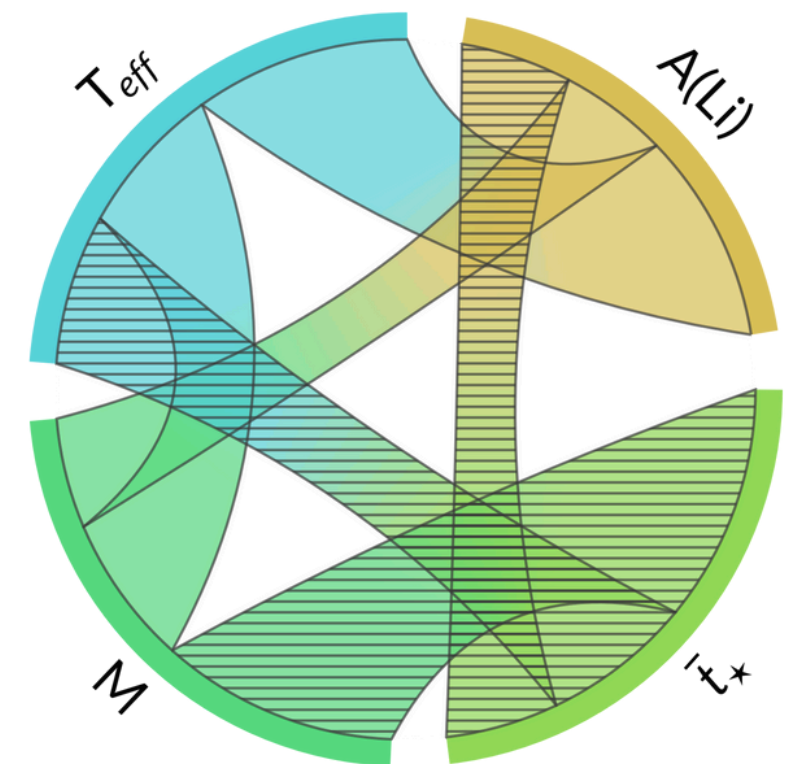
threshold=0.05



threshold=0.25



threshold=0.4

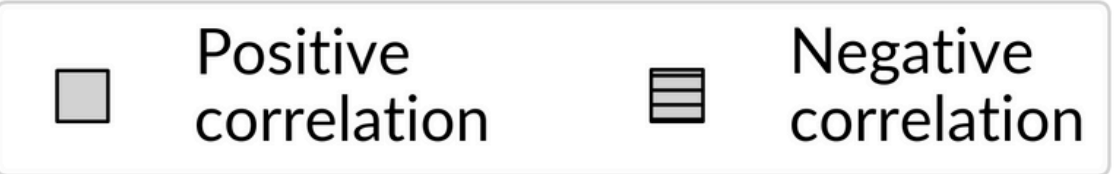
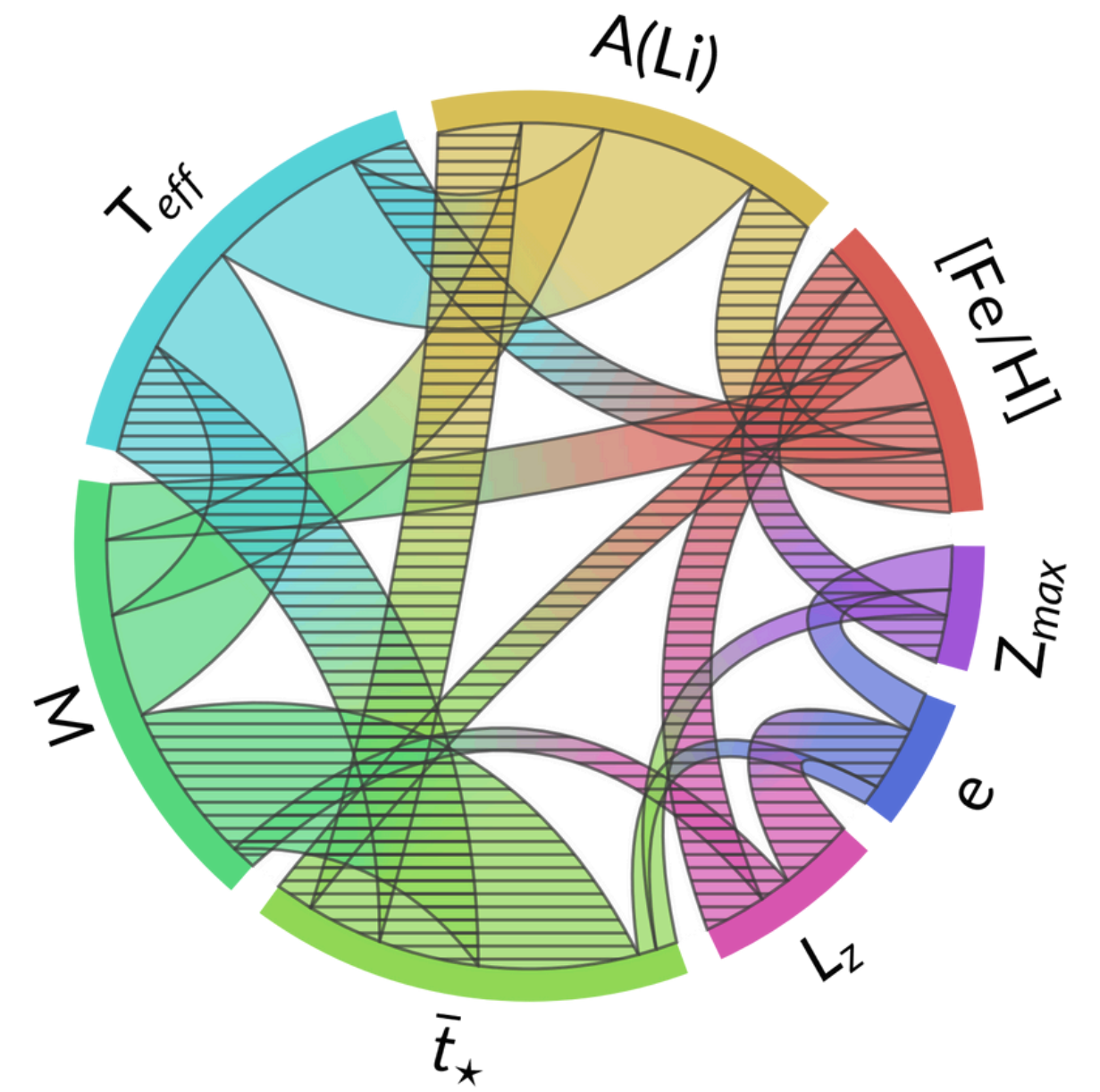




# Chord Diagrams: Customization

```
notebook.ipynb ×  
  
# legend : Adds default positive and  
# negative labels in the legend.  
  
chp.chord(corr_matrix, legend=True)  
  
plt.legend(  
    loc='center',  
    bbox_to_anchor=[0.5,0.0],  
    ncol=2,  
)
```

Output



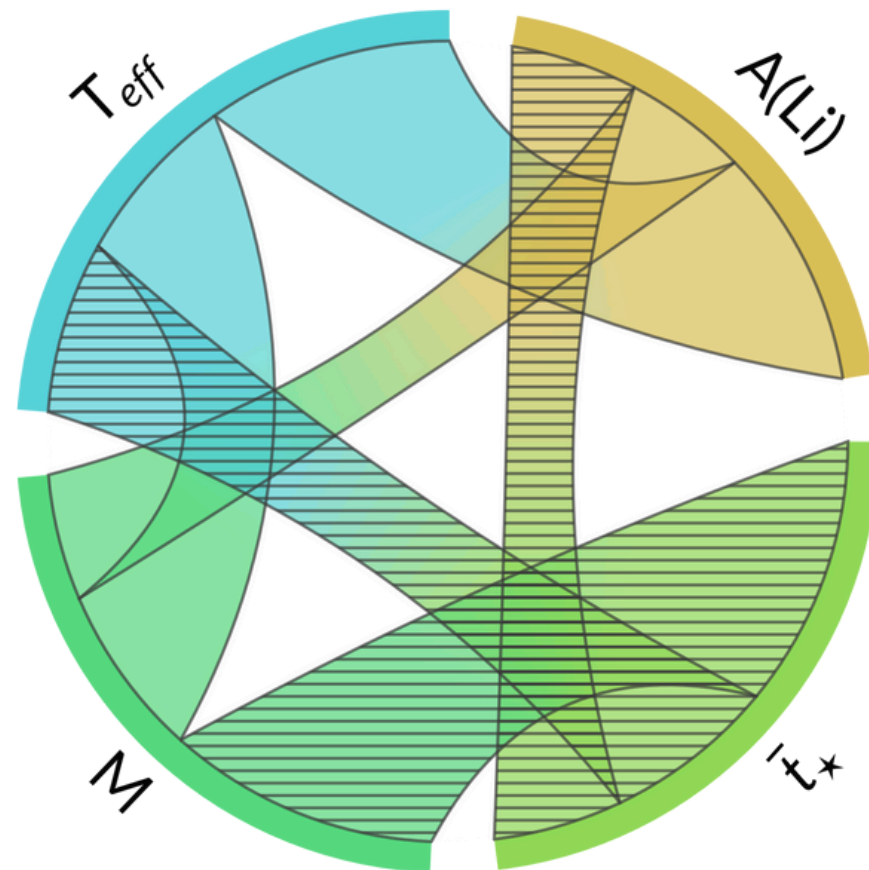


# Chord Diagrams: Customization

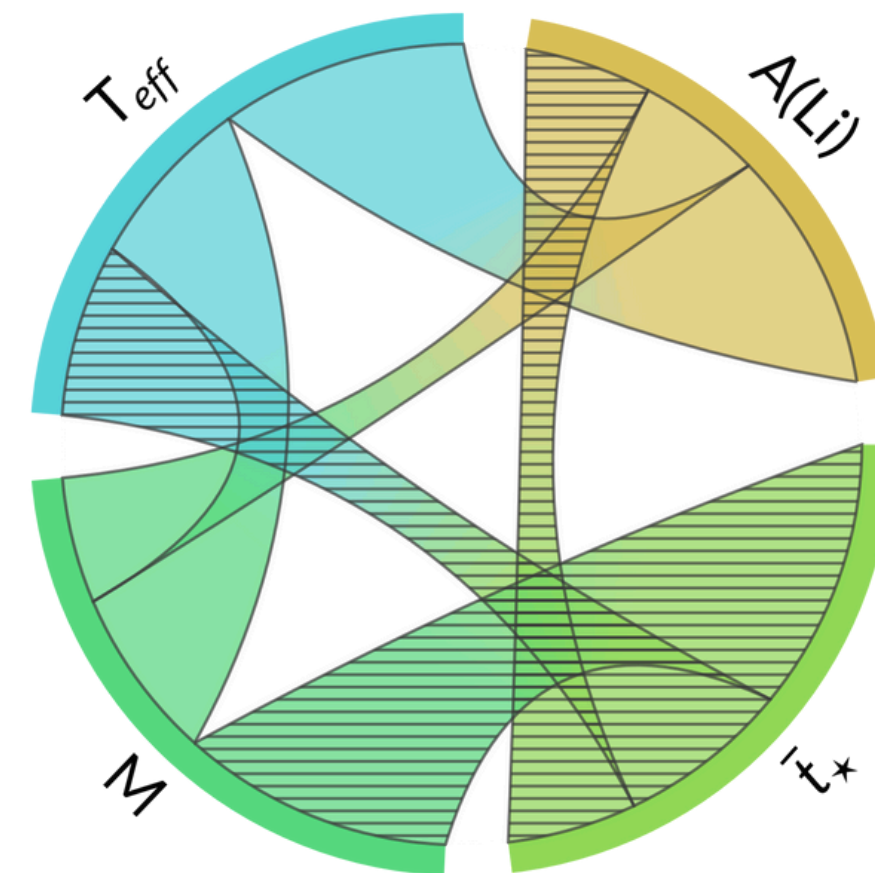
notebook.ipynb ×

```
# scale : Scale use to set chord's thickness, whether "linear" or "log"  
chp.chord(corr_matrix, scale=your_scale)
```

scale='linear'



scale='log'



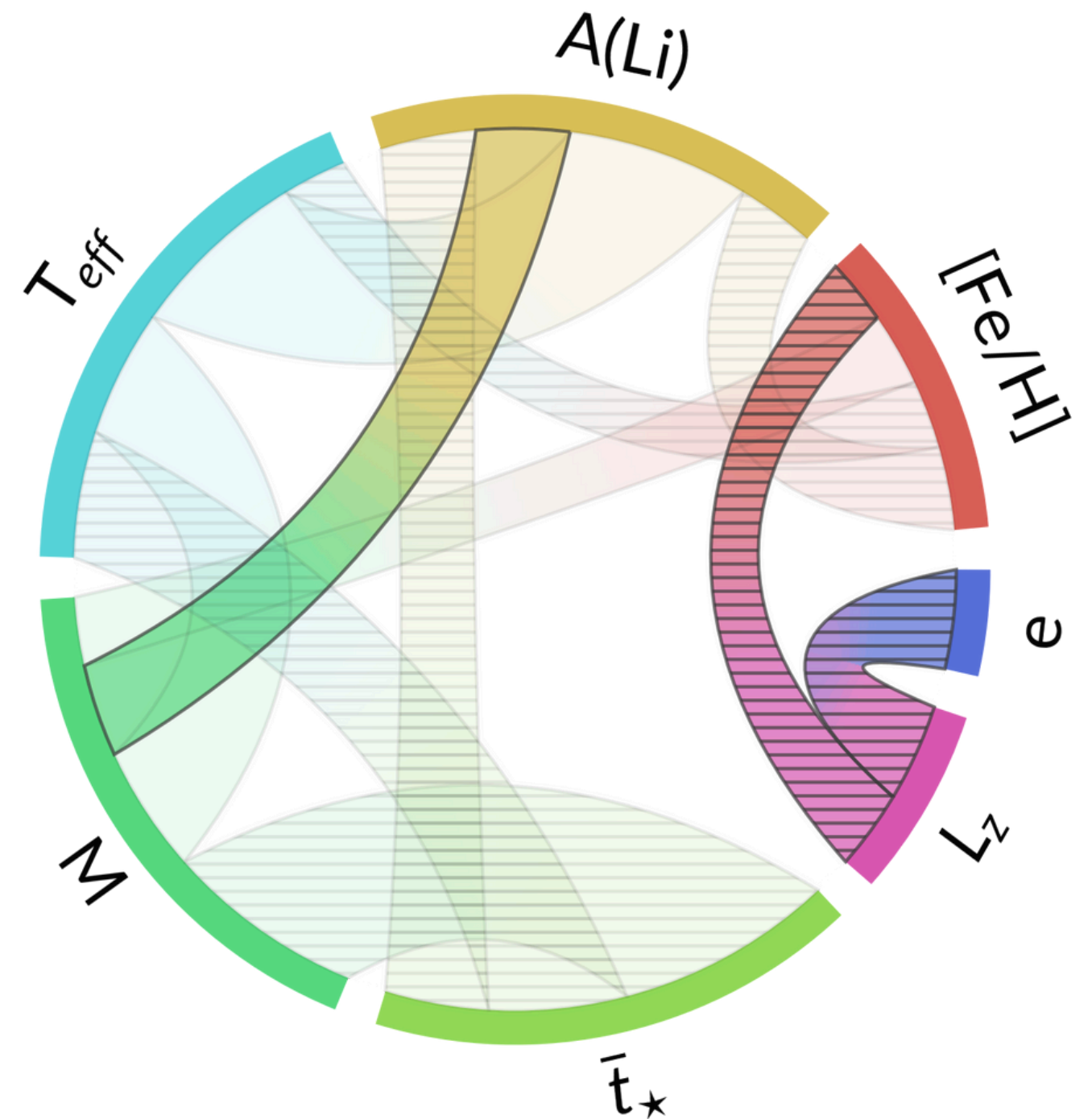


# Chord Diagrams: Customization

notebook.ipynb ×

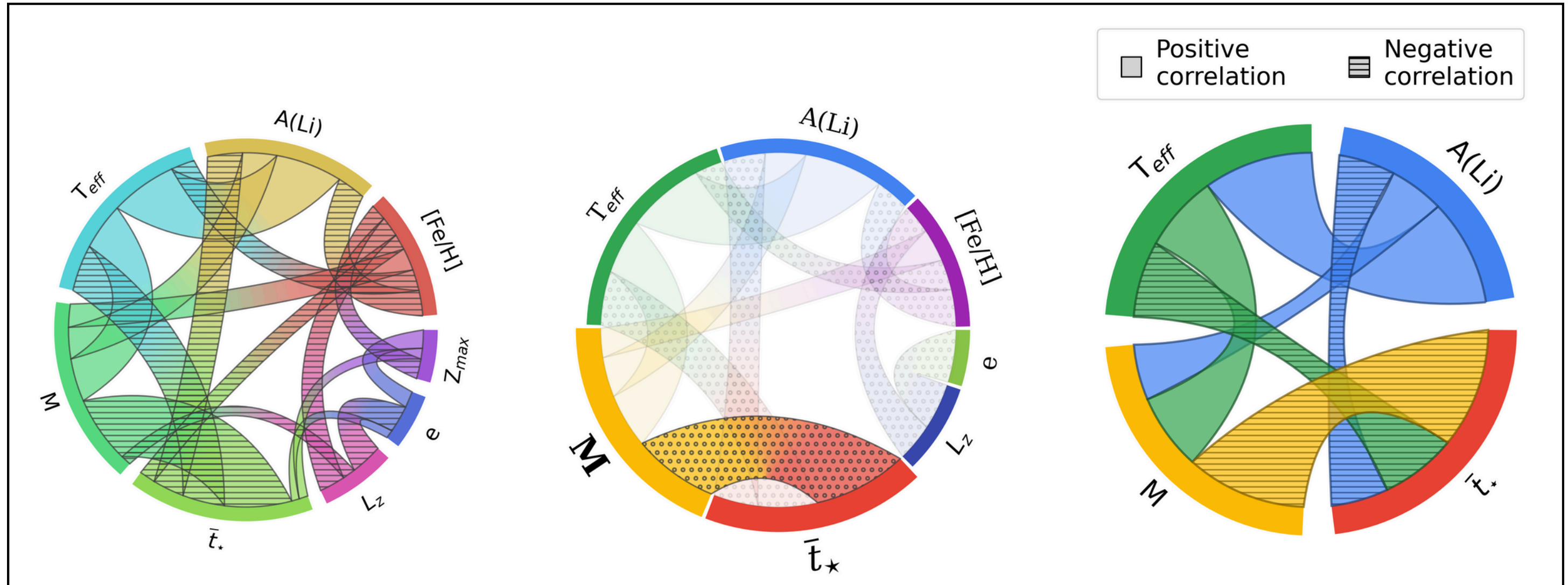
```
# We also have a highlighting method  
  
# We need to return the chord object  
my_chord = chp.chord(corr_matrix)  
  
# Let's highlight the 6th node  
# (node=5)  
my_chord.highlight_node(5)  
  
# And also highlight the 3rd chord  
# (chord=2) of the 2nd node (node=1)  
my_chord.highlight_chord(1,2)
```

Output





# Chord Diagrams: More examples



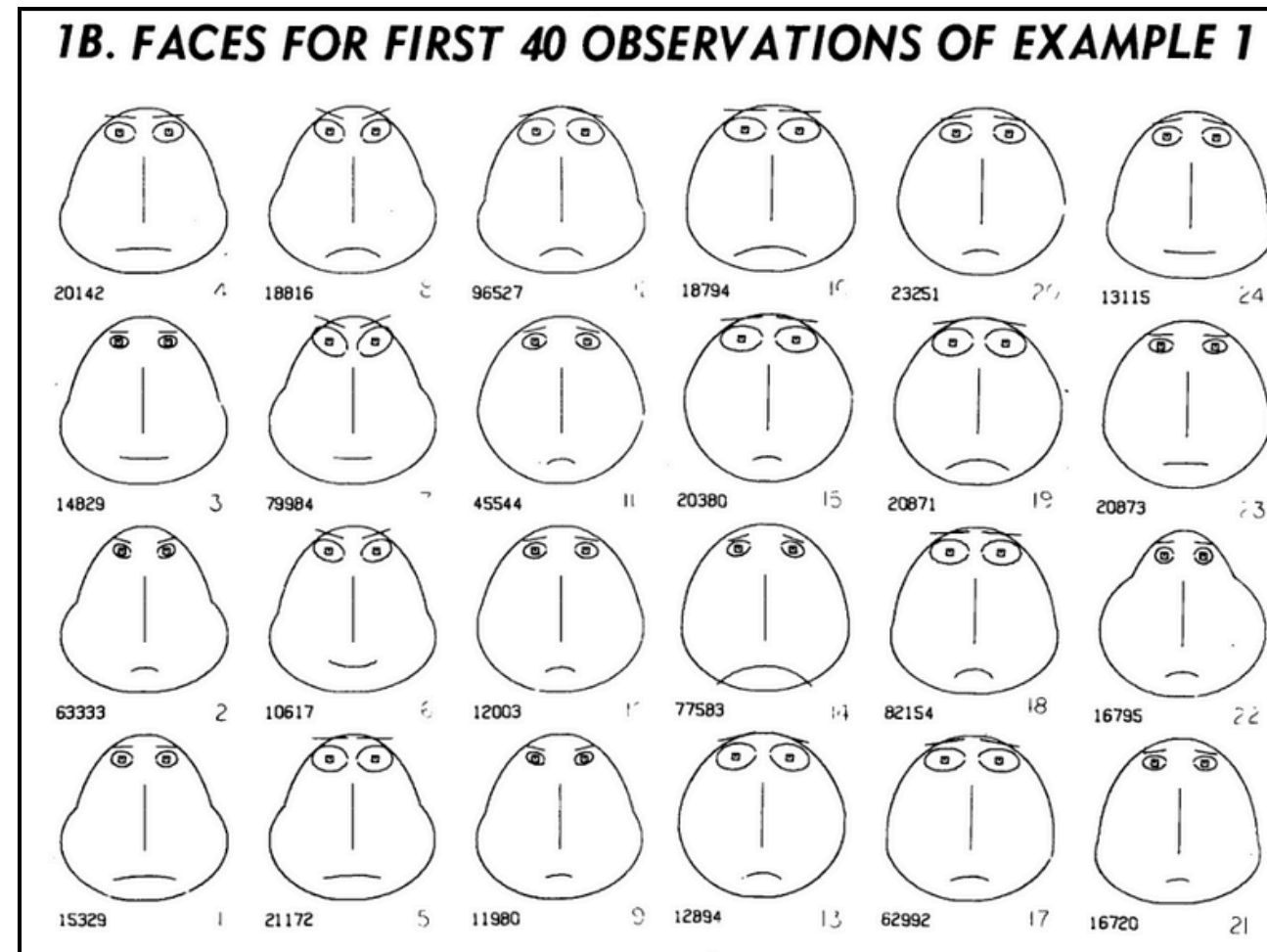
D. Beltrán & M. L. L. Dantas (2025)

Check out the [examples section](#) in the documentation!



# Michinoff Faces

**Chernoff faces** visualize multivariate data by mapping each variable to a **facial feature**. Because humans are highly sensitive to facial differences, this method makes it easy to **spot patterns** and **compare observations**.

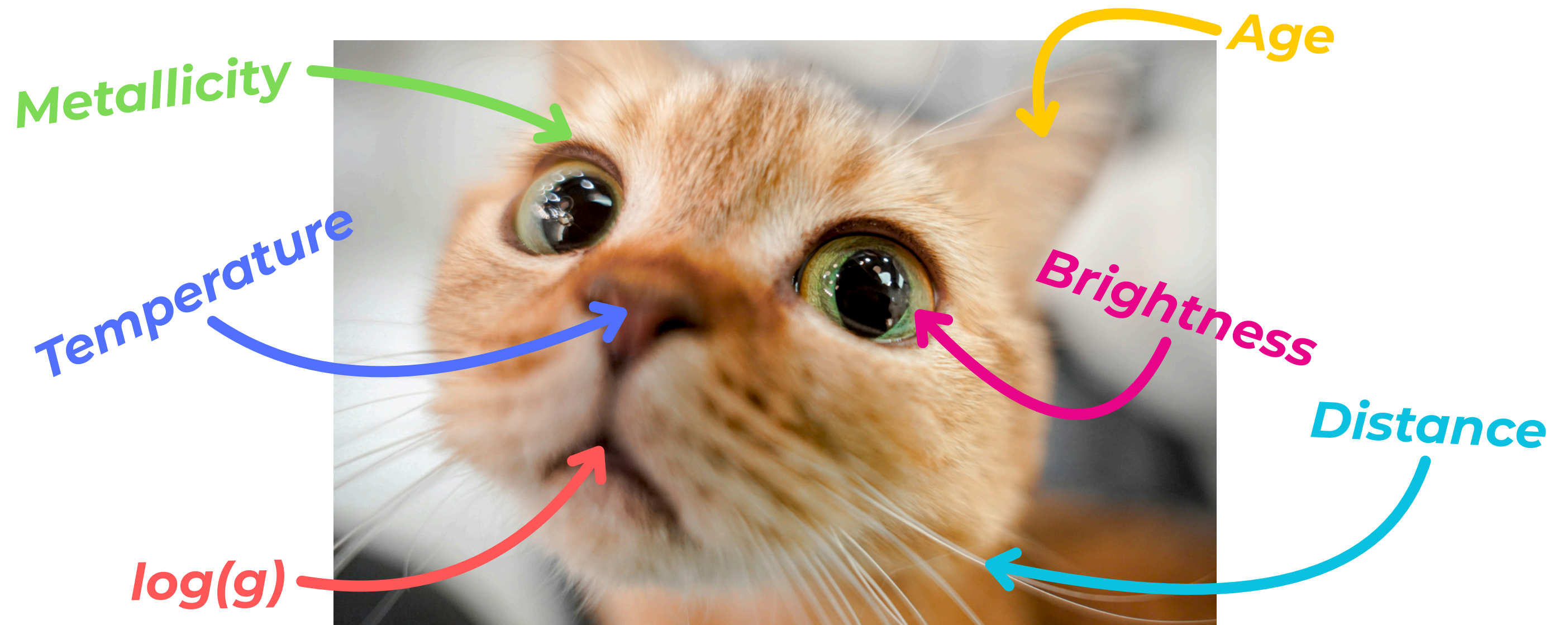


Chernoff, H. (1973)



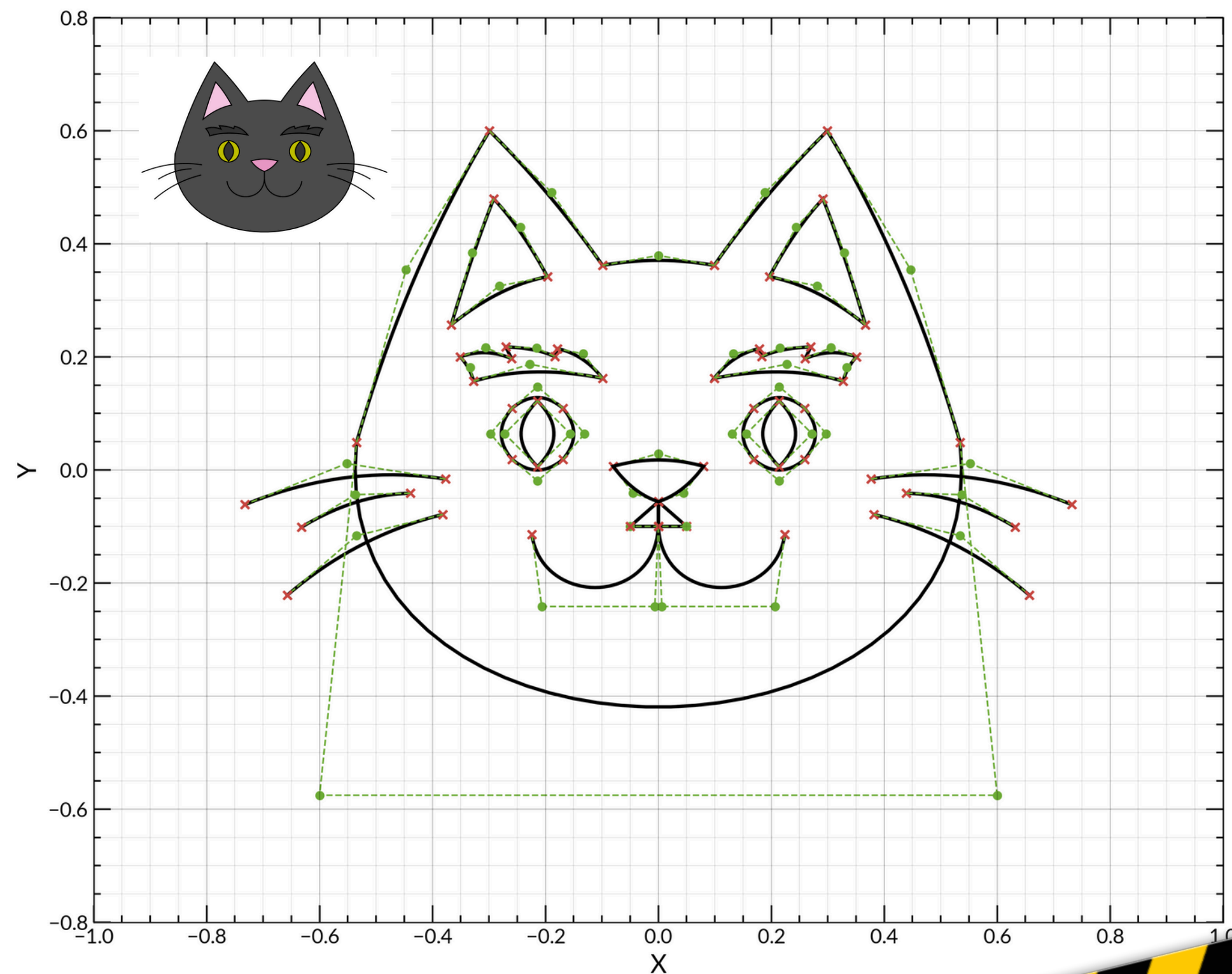
# Michinoff Faces

**Michinoff faces** follow the same idea, but replace the human face with a **cat face**. The name comes from *michi*, a colloquial word used in Latin America (including Chile) to mean “cat”.



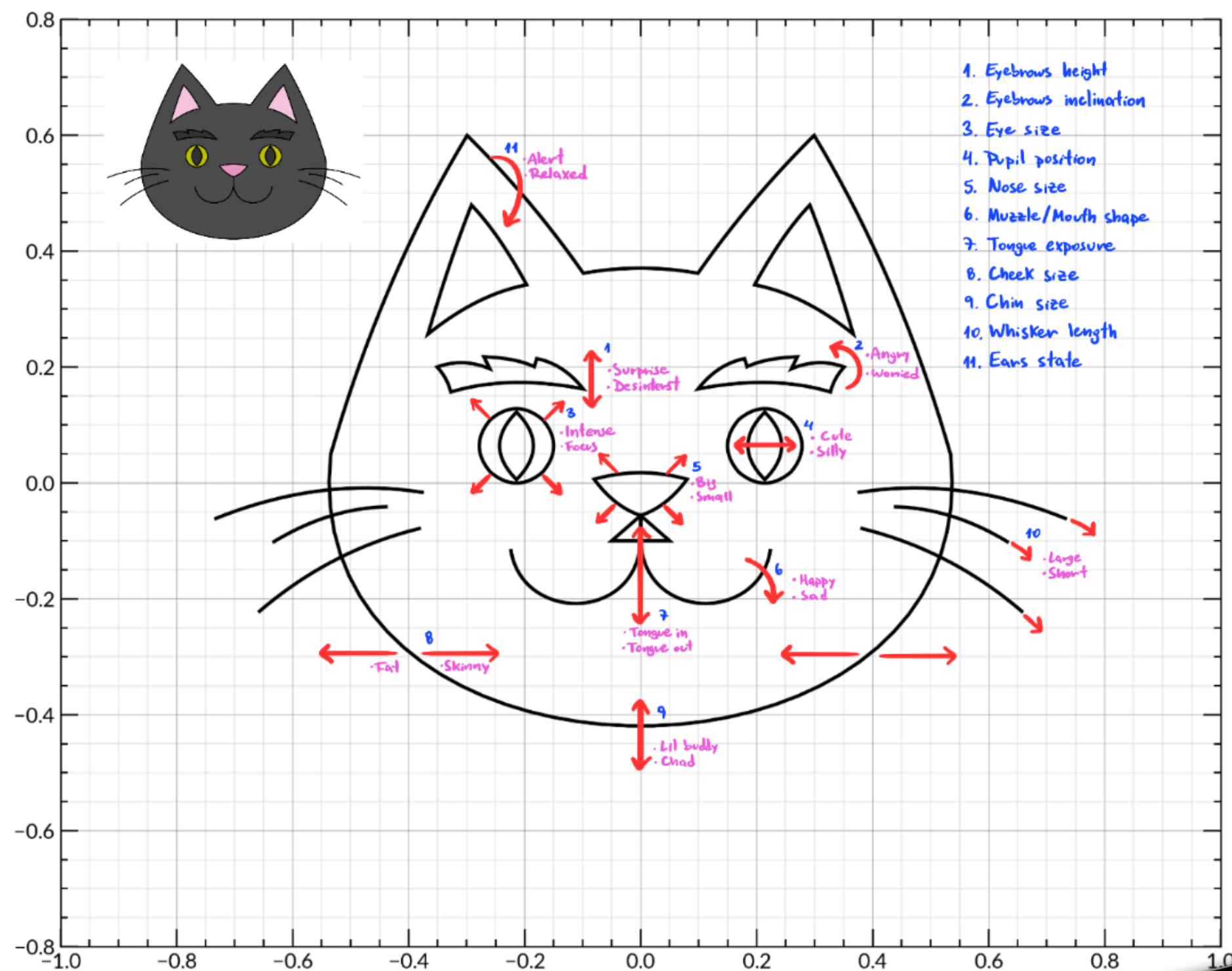


# Michinoff Faces: Construction



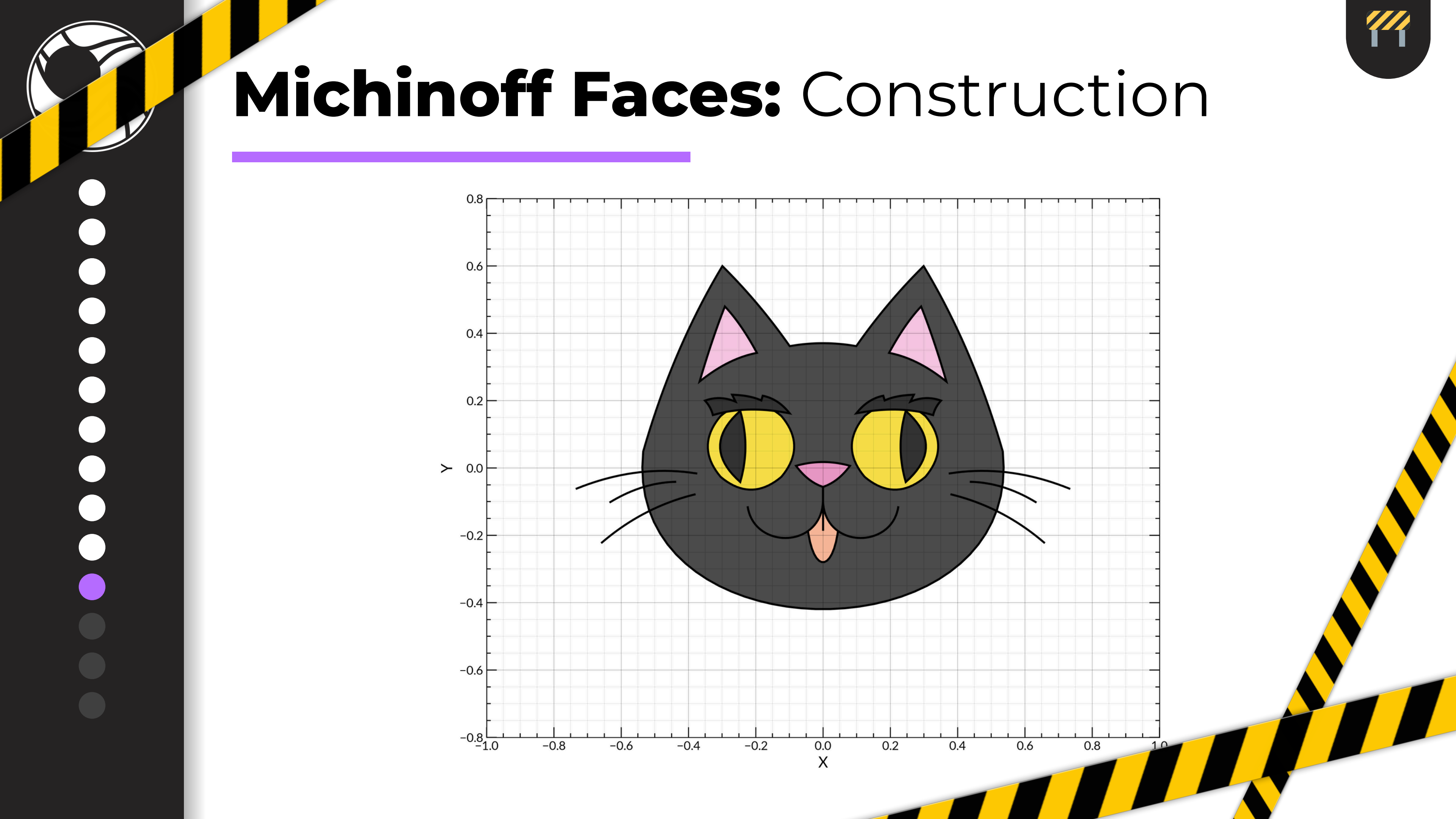
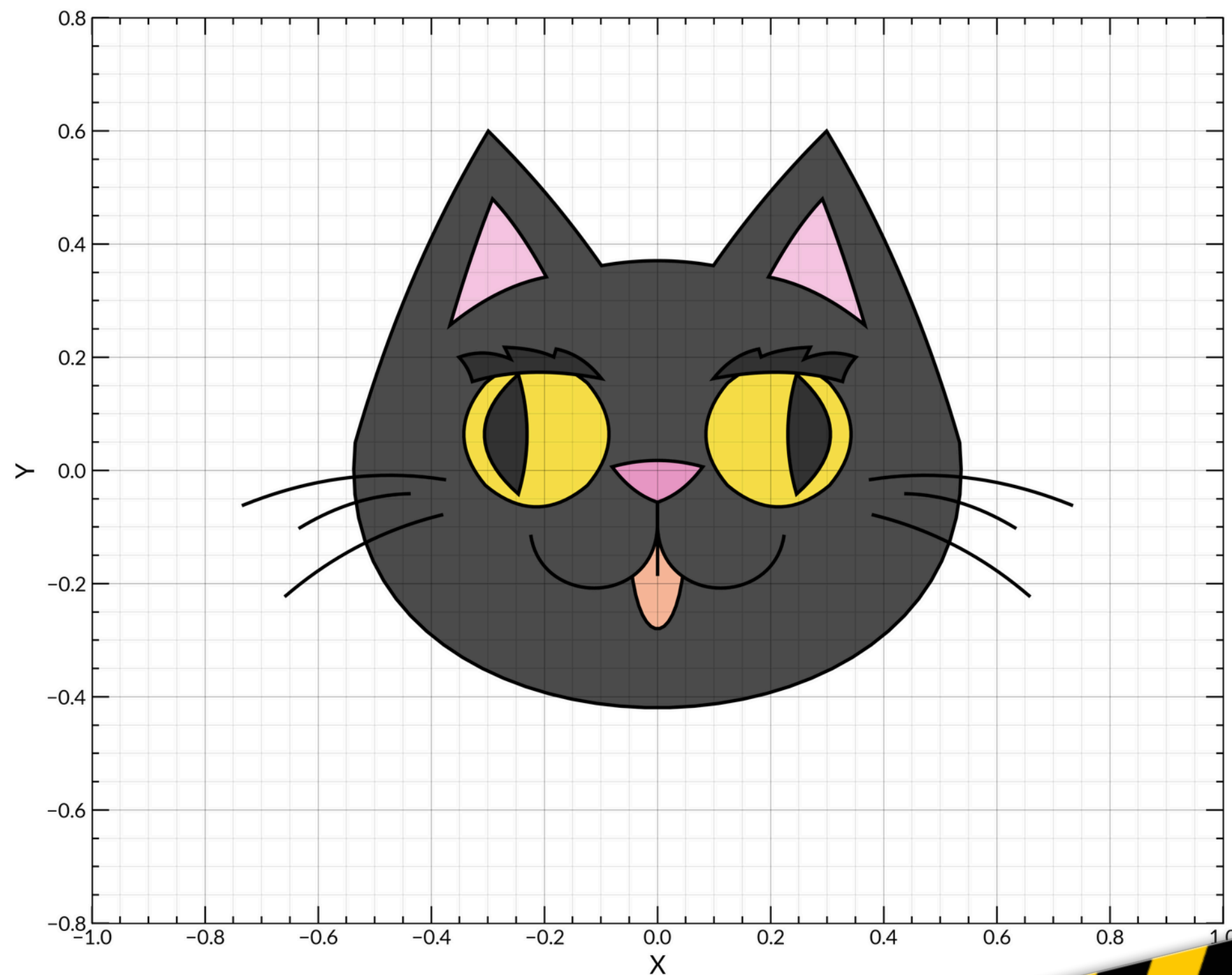


# Michinoff Faces: Construction





# Michinoff Faces: Construction





# Michinoff Faces: *First tests*





# Michinoff Faces: *First tests*

---

A dark gray rectangular area containing a YouTube error message. On the left is a white exclamation mark inside a white circle. To its right, the text reads: Watch video on YouTube, Error 153, and Video player configuration error. In the bottom right corner of this area is a small white play button icon.

**Link:** <https://youtu.be/ULfOqun6qyM>





# Michinoff Faces: Example

Let's see what the three interstellar comets "look like" using the **Michinoff faces**.

Here is some data:

Name	$v_{\infty}$ (km/s)	$e$	$q$ (AU)	$B - V$ (mag)	$r_n$ (km)	$i$ (deg)	$dM/dt$ (kg/s)
<b>I1/'Oumuamua</b>	26	1.2	0.25	0.70	0.06-0.11	122	$< 2 \times 10^{-3}$
<b>I2/Borisov</b>	32	3.4	2.00	0.82	0.20-0.50	44	160-250
<b>I3/ATLAS</b>	58	6.1	1.36	0.98	$< 2.80$	175	88-880



# Michinoff Faces: Example

Let's see what the three interstellar comets "look like" using the **Michinoff faces**.

Here is some data:

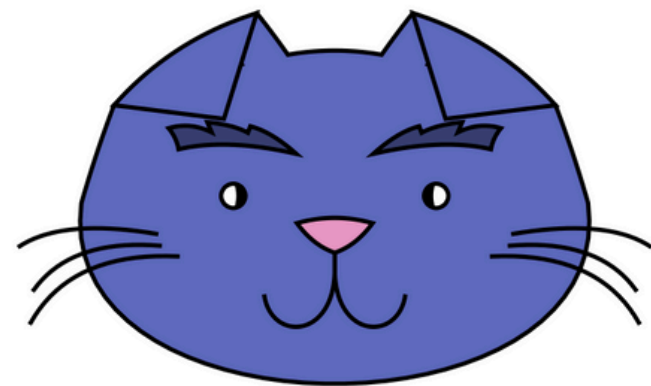
Name	<i>Eye size</i> $v_{\infty}$ (km/s)	<i>Pupil</i> $e$	<i>Tongue</i> $q$ (AU)	<i>Face color</i> $B - V$ (mag)	<i>Chin</i> $r_n$ (km)	<i>Eyebrow inclination</i> $i$ (deg)	<i>Ears</i> $dM/dt$ (kg/s)
I1/'Oumuamua	26	1.2	0.25	0.70	0.06-0.11	122	$< 2 \times 10^{-3}$
I2/Borisov	32	3.4	2.00	0.82	0.20-0.50	44	160-250
I3/ATLAS	58	6.1	1.36	0.98	$< 2.80$	175	88-880



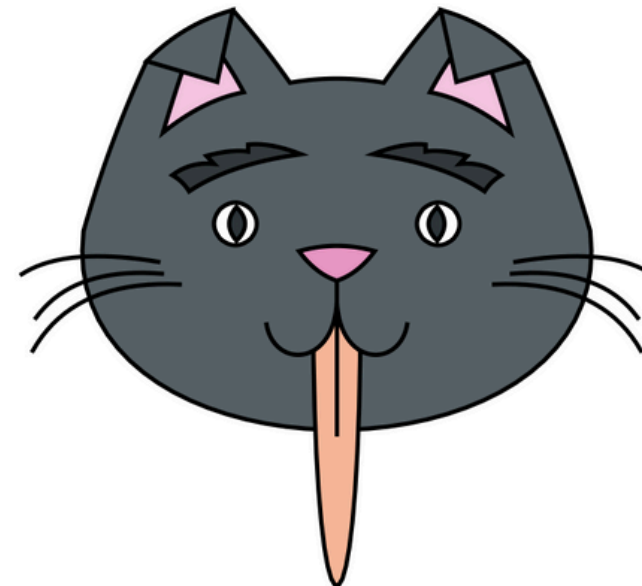
# Michinoff Faces: Example

Let's see what the three interstellar comets "look like" using the **Michinoff faces**.

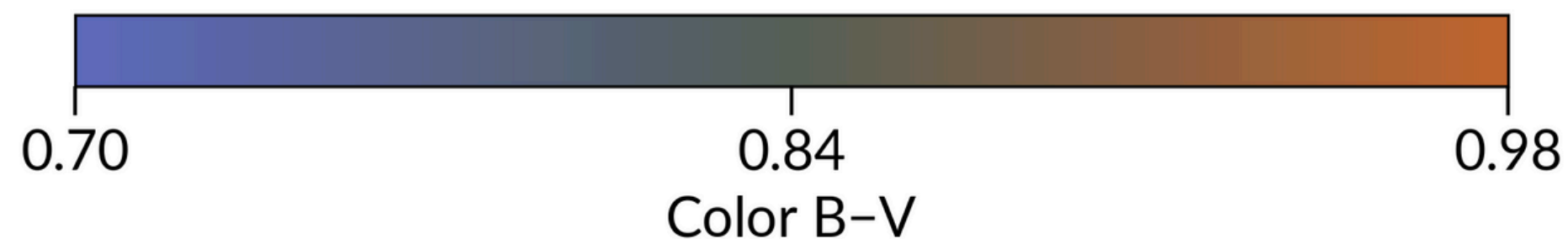
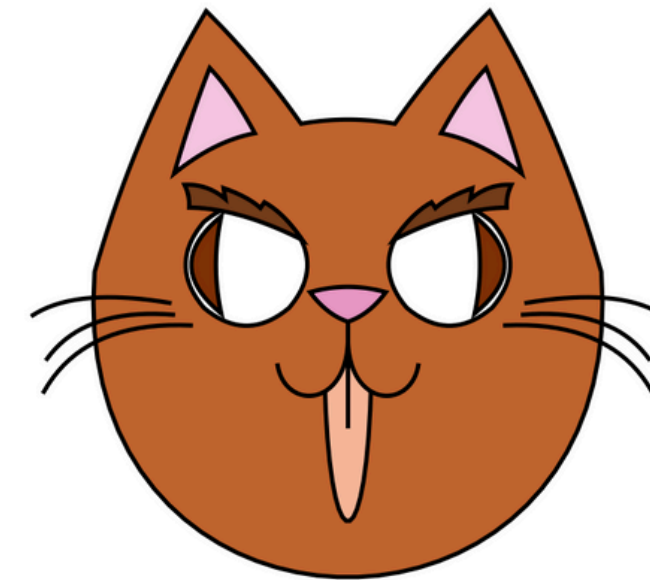
1I/'Oumuamua



2I/Borisov



3I/ATLAS





# Project links

**GitHub:** [github.com/DD-Beltran-F/cachai](https://github.com/DD-Beltran-F/cachai)

The screenshot shows the GitHub repository page for 'cachai' by user 'dbeltran'. The repository is public and has 5 stars, 0 forks, and 0 watches. It features a 'main' branch, 1 branch, and 11 tags. The repository contains several files and folders, including '.github/workflows', 'cachai', 'docs', '.gitignore', '.readthedocs.yaml', 'LICENSE', 'README-pypi.md', 'README.md', and 'pyproject.toml'. The most recent commit is 'Notebook update' by 'dbeltran' 2 hours ago, with 79 commits in total. The repository also has 9 releases, with the latest being '0.1.4' on Oct 14.

File/Folder	Commit Message	Commit Hash	Time Ago
dbeltran	Notebook update	bef6cba	2 hours ago
.github/workflows	Fix PyPI readme		2 months ago
cachai	v0.1.4		last month
docs	Notebook update		2 hours ago
.gitignore	v0.1.0		2 months ago
.readthedocs.yaml	Fix versions of docs dependencies		2 months ago
LICENSE	LICENSE		5 months ago
README-pypi.md	v0.1.3		last month
README.md	v0.1.3		last month
pyproject.toml	v0.1.4		last month



# Project links

**PyPI:** [pypi.org/project/cachai](https://pypi.org/project/cachai)

The screenshot shows the PyPI project page for 'cachai 0.1.4'. The header is blue and contains the version number 'cachai 0.1.4' on the left, a green checkmark and 'Latest version' button on the right, and the release date 'Released: Oct 14, 2025' below the button. A dark blue box contains the command 'pip install cachai' with a copy icon. Below the header is a light grey bar with the description 'Customizable visualization toolkit for science'. The main content area is split into two columns. The left column, titled 'Navigation', has a blue button for 'Project description' and links for 'Release history' and 'Download files'. The right column, titled 'Project description', features the 'cachai' logo, which consists of a circular icon with purple and teal swirls and the word 'cachai' in teal text. At the bottom left, there is a 'Verified details' badge with a green checkmark and the text 'These details have been verified by PyPI'.



# Project links

Read *the Docs*: [cachai.readthedocs.io](https://cachai.readthedocs.io)


The screenshot shows the documentation page for the 'cachai' project. The page has a dark purple header with the 'cachai' logo and name. Below the header is a search bar and a 'latest' version selector. The left sidebar contains a 'CONTENTS' section with links to 'Installation Guide', 'Getting Started', 'Examples', 'Documentation', and 'Citing', and a 'LINKS' section with links to 'Source code', 'Report an issue', 'Changelog', and 'PyPI'. The main content area features a large circular logo with the word 'cachai' in a light blue font. Below the logo, there is a welcome message: 'Welcome! cachai (Custom Axes and CHarts Advanced Interface) is a fully customizable Python visualization toolkit designed to deliver polished, publication-ready plots built on top of Matplotlib. Currently, the package includes the `ChordDiagram` module as its primary feature. For details on the toolkit's capabilities, motivations and future projections, refer to [this paper](#). To contribute or report bugs, please visit the [Issues page](#).' A 'Fun fact' box contains the text: 'Cachai' (/ka:'tʃai/) is a slang word from Chilean informal speech, similar to saying "ya know?" or "get it?" in English. Don't know how to pronounce it? Think of "kah-CHAI" (like "cut" + "chai" tea, with stress on "CHAI").' At the bottom, there is a 'Contents' section with a link to the 'Installation Guide'.



It would mean a lot to me if **cachai** is useful to anyone :)  
If you decide to use it, please consider citing this article:

**CACHAI's First Module: A Fully Customizable Chord Diagram for  
Astronomy and Beyond - IOPscience**

CACHAI's First Module: A Fully Customizable Chord Diagram for Astronomy and  
Beyond, Beltrán, D., Dantas, M. L. L.

 [iopscience.iop.org](https://iopscience.iop.org) / Aug 11

**DOI: 10.3847/2515-5172/adf8d**



**Thank you very much for your time and attention!**

