ARC CENTRE OF EXCELLENCE FOR DARK MATTER

THE UNIVERSITY OF SYDNEY

How to make a plot

Ciaran O'Hare, USyd

Outline

1. Motivation and general advice
2. Designing a plot
3. Specific practical tips (matplotlib)
4. Examples of good/bad practice

Every plot in this presentation is reproducible via a Jupyter notebook.
Go to github.com/cajohare/HowToMakeAPlot or download .zip file on confluence

Apologies in advance, many slides are intentionally dense as I want this to be useful as a reference

Motivation: why bother?

- Figures and visualisation are a form of **science communication**, and the success of your science rests upon how well it is communicated.

- Good plots grab attention and convey complex information quickly, helping you explain your science better and to more people. An attention to detail tells others that you ***care*** about being understood.

- Good figures help ***you***. If someone likes your plot, it is more likely they will cite it, use it in their talk and (hopefully) mention your name when doing so.

- Working hard on plots makes your science materially better. A figure you have spent more time working on is also a figure you have spent more time *looking* at. More likely to catch details, errors, room for improvement etc.

Motivating yourself

- Spend *time* on your plots. Do not feel bad about spending upwards of days/weeks refining a plot that is going to convey the main result of a paper.

- An image will speak for itself, whether or not you intend it to. Make your plot in such a way that it is actually saying the thing you want it to say.

- Understand *what* the plot is saying by imagining what someone will think when they see it for the first time. If you can't imagine, just show it to someone and ask them what they think.

- Create a plot that is suitable for the setting in which it is presented. A good plot for a paper is not going to be a good plot for a talk or a poster. **Do not copy/paste plots from your paper into your slides.** (I am aware we all do this, but we shouldn't)

Plots are not scientific results

- Your plots are not scientific results in and of themselves, they are you ***communicating*** your scientific results. They are just as much propaganda as everything else you put in your paper, or say during a talk.

- With that in mind, do not just dump all of your data onto a plot and expect people to draw their own conclusions. Instead, consider what your plots actually are: **a visual description of a quantitative result.** Don't treat them as graphical databases, but rather as an image which is intended to convey a message.

*"Your paper is not what you did, it is what you **say** you did"*

— A clever thing someone said to me once but I forgot who

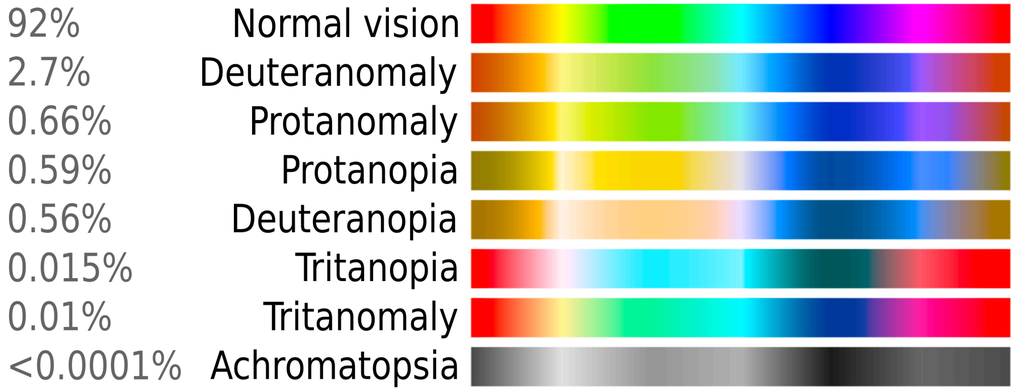
Designing plots

Non-negotiables

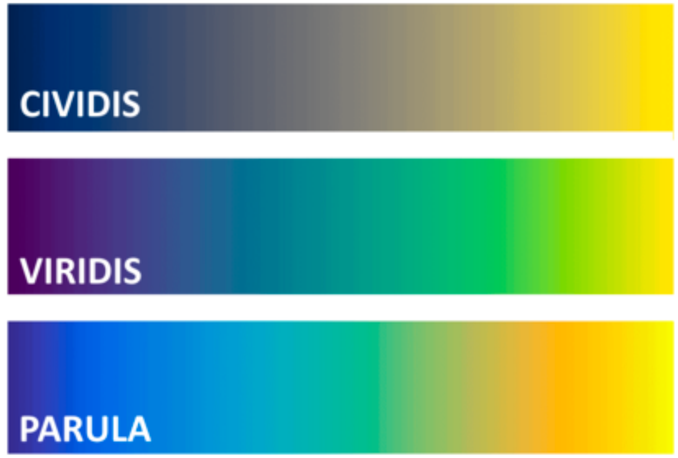
- **Careful with your use of colour.** ~4% of people have some form of colourblindness. There are online tools to apply filters to your plots to check if they're readable to those people. Best cautionary measure is to not have essential messages tied up in interpreting detailed colour information.
(Or just avoid red and green next to each other)

- Be mindful about **file size**. If people can't load the paper pdf on their browser, it doesn't even matter what your plot looks like. Only in extreme cases will a plot be larger than 1 MB.

Types of colourblindness



Colormaps designed with colourblindness in mind



General technical tips

- **Cut your losses.** Python, Mathematica, MATLAB etc. are powerful, but can be very annoying when it comes to certain things. Forcing yourself to finish your entire plot using one tool may not be worth your time. No shame in putting the final touches using something with a visual user interface like powerpoint, keynote, inkscape, illustrator, photoshop etc.

- Always make plots in a **vector graphics** format, unless there really is no other option

- **Watch font sizes.** No font on a plot should appear smaller than the font of the main body of paper/slide. Ideally, it should be bigger, big enough to read when the pdf is zoomed out, or for a person at the back of the room.

Making good plots requires “soft” and “technical” skills

Soft

- An understanding of how humans perceive and process visual information
- An eye for detail
- Intuition for when something is unattractive or “off”

Technical

- Ability to translate an idea into code
- Knowing how to avoid graphical artefacts
- Knowledge of fonts, colour theory, pre-attentive visual attributes etc.

Disclaimer: I am not qualified to lecture you on either, but I can give you tips from experience. You are also welcome to disagree with me on anything that I say.

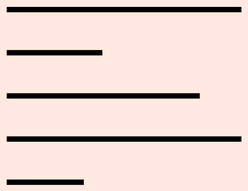
Pre-attentive visual attributes

Attributes your spatial memory subconsciously processes in the first few milliseconds when looking at an image. This is the "*Standard Model*" of information design. You can combine any and all of them to convey both quantitative and qualitative information.

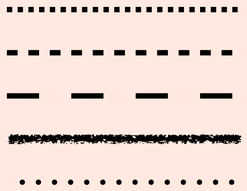
Form

Width

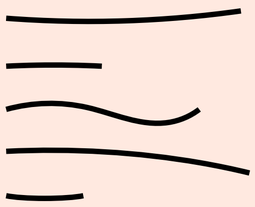
Length



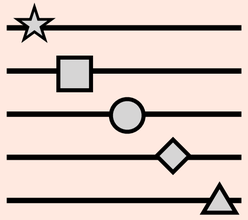
Style



Curvature

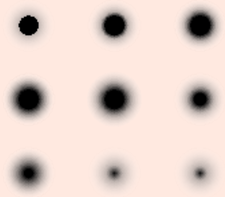


Markers

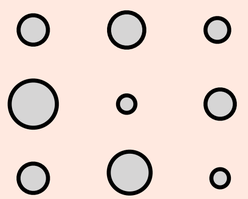


Enclosure

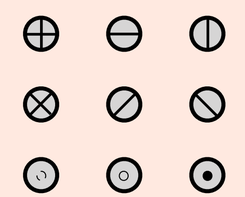
Blur



Size

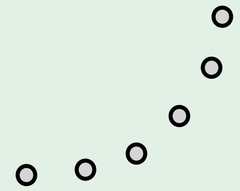


Decoration

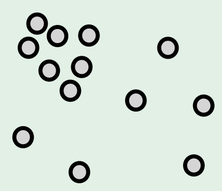


Spatial

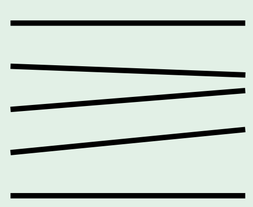
Position



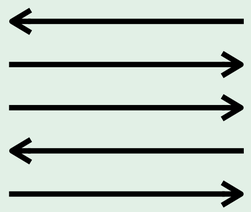
Density



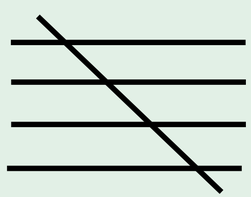
Orientation



Direction



Intersection

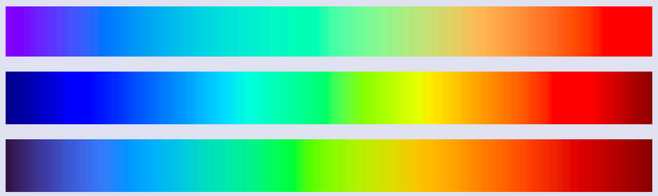


Proximity

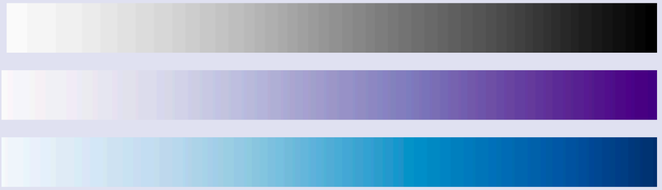
Colour

Hue

Saturation



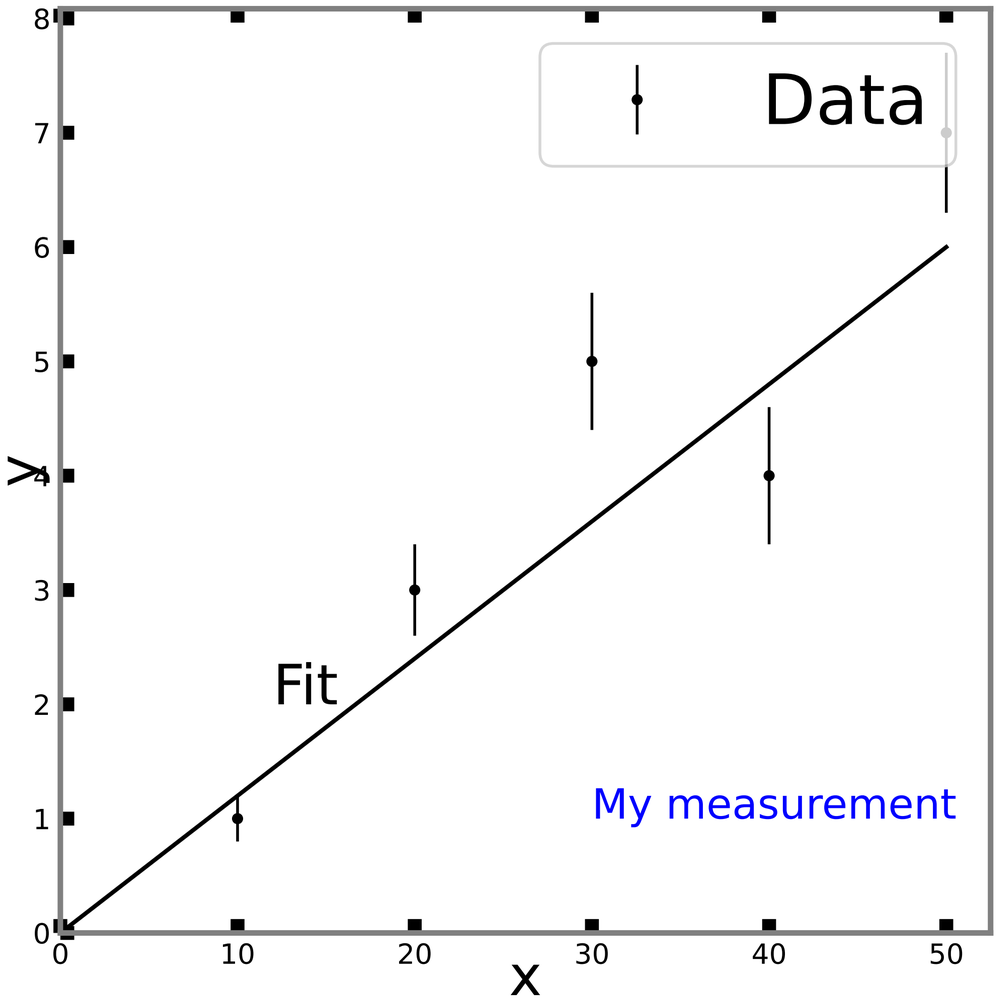
Lightness



Using pre-attentive attributes

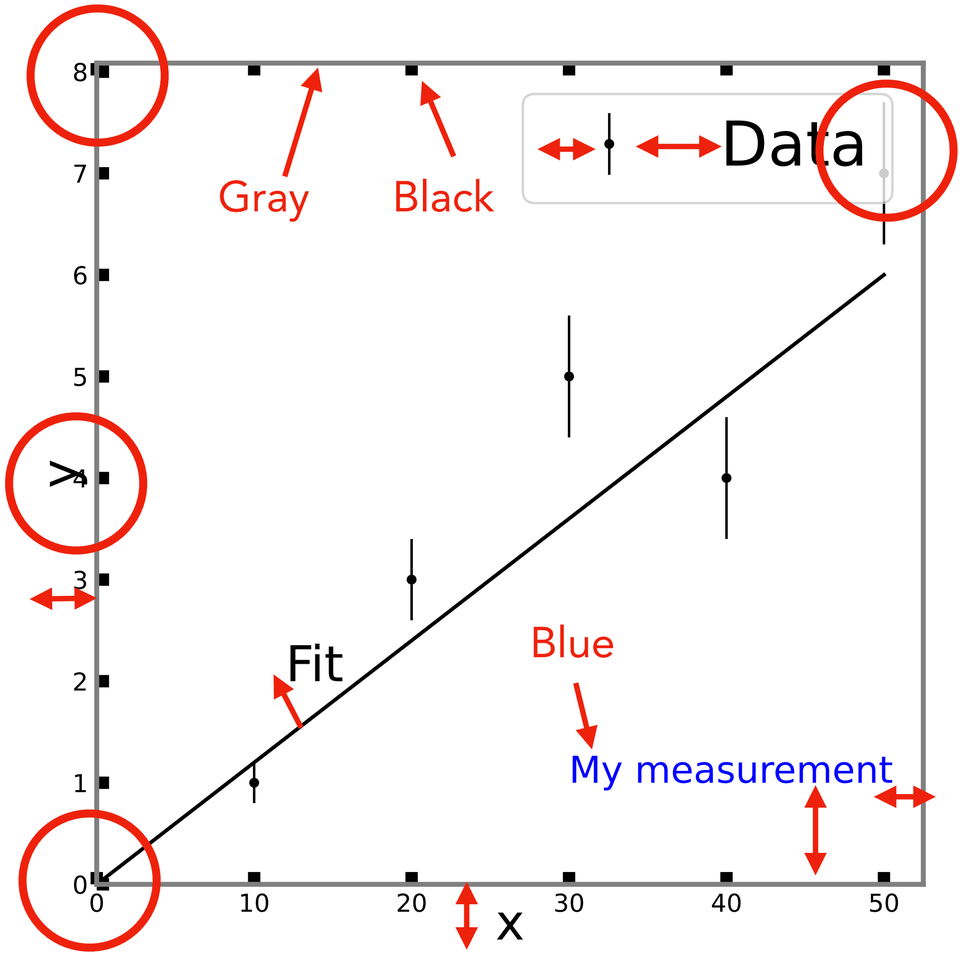
- Every plot is built out of pre-attentive visual attributes automatically. Your task is to **control** which ones you use and to remove ones which appear by accident. Make sure every element is purposeful and supports the main message.
- The reason most bad plots ***look*** bad is because of sloppy use of these attributes, i.e. **visual clutter**. Individually these may seem like small things, but our brains are tuned to spot anything out of place.
- Even seemingly trivial things like asymmetries, misalignment, or overlapping elements can stack up and degrade the intelligibility of a plot.

Does this plot look okay to you?

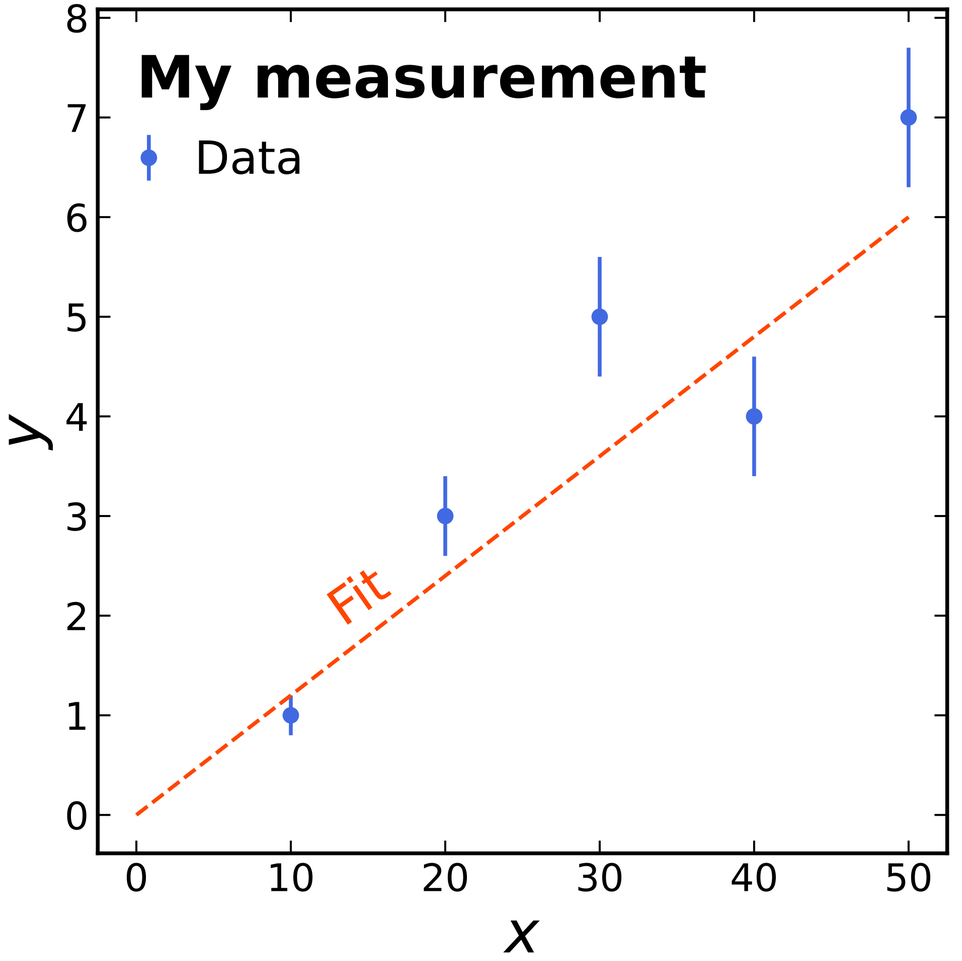


Using pre-attentive attributes

You can instantly improve the visual just by removing visual elements that were in the plot unintentionally

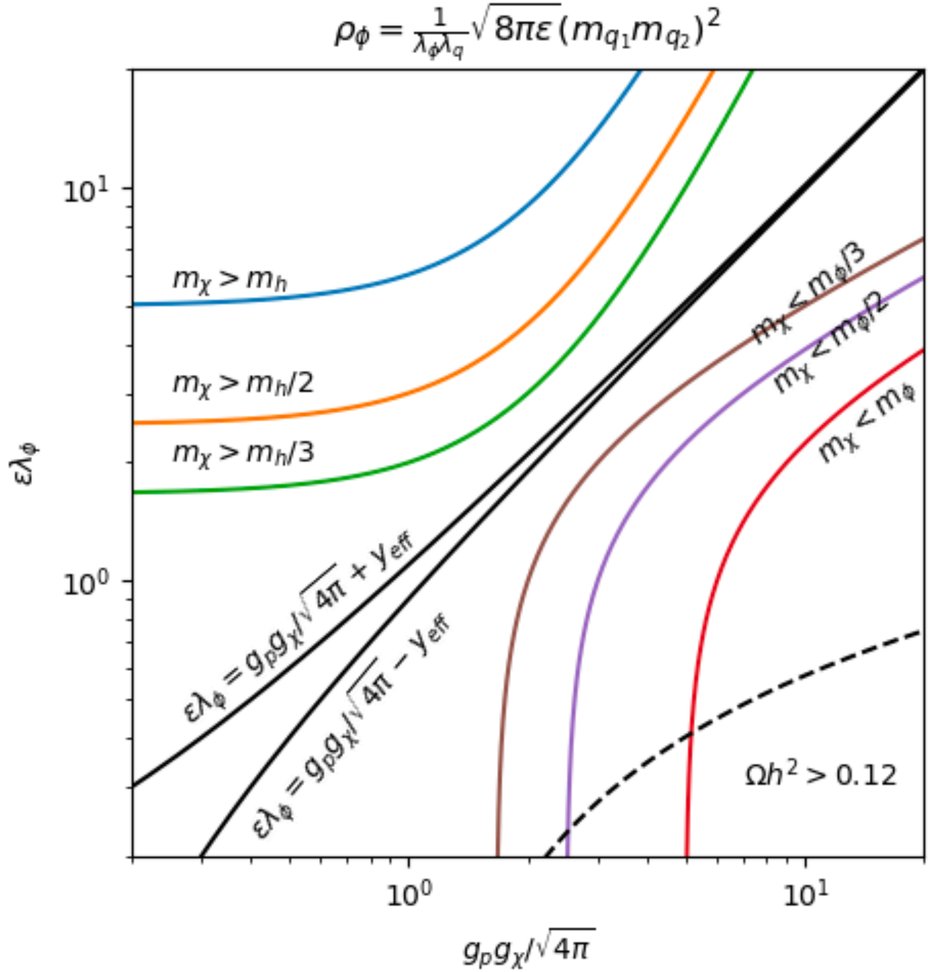


Clutter.ipynb

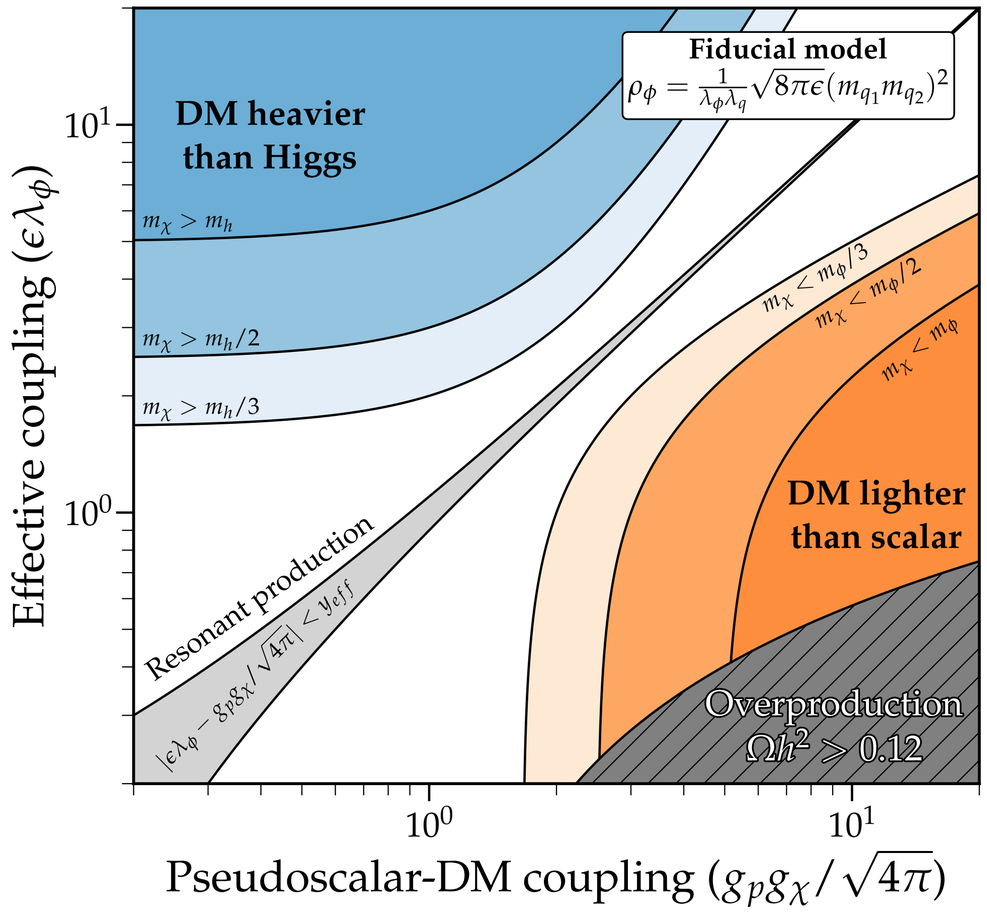


Label your plots properly

Not labelling your plot using English is the best way to make your reader's life as difficult as possible. Don't make them hunt down the definition of every variable name, just spell out what the plot is saying. This is especially important for plots used in talks.



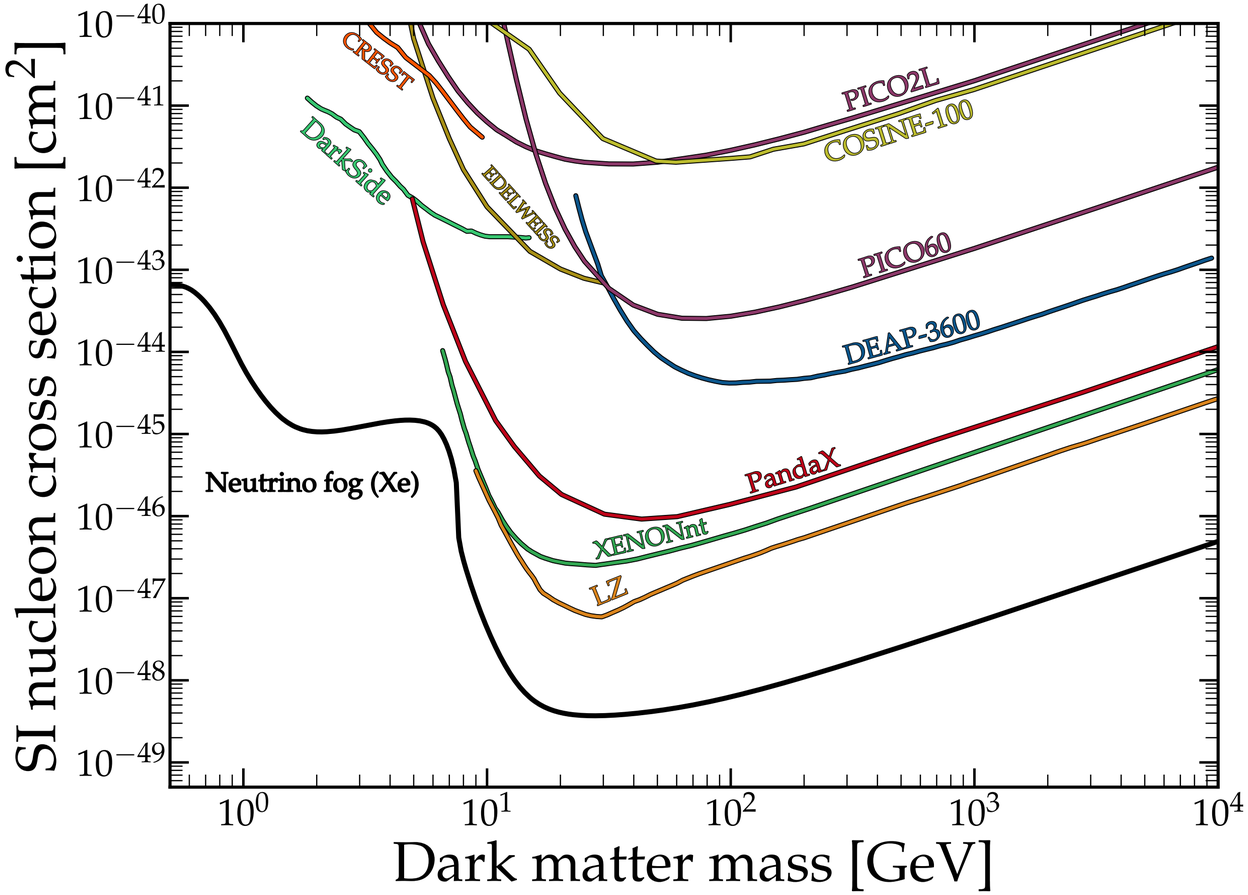
MathematicalPlot.ipynb

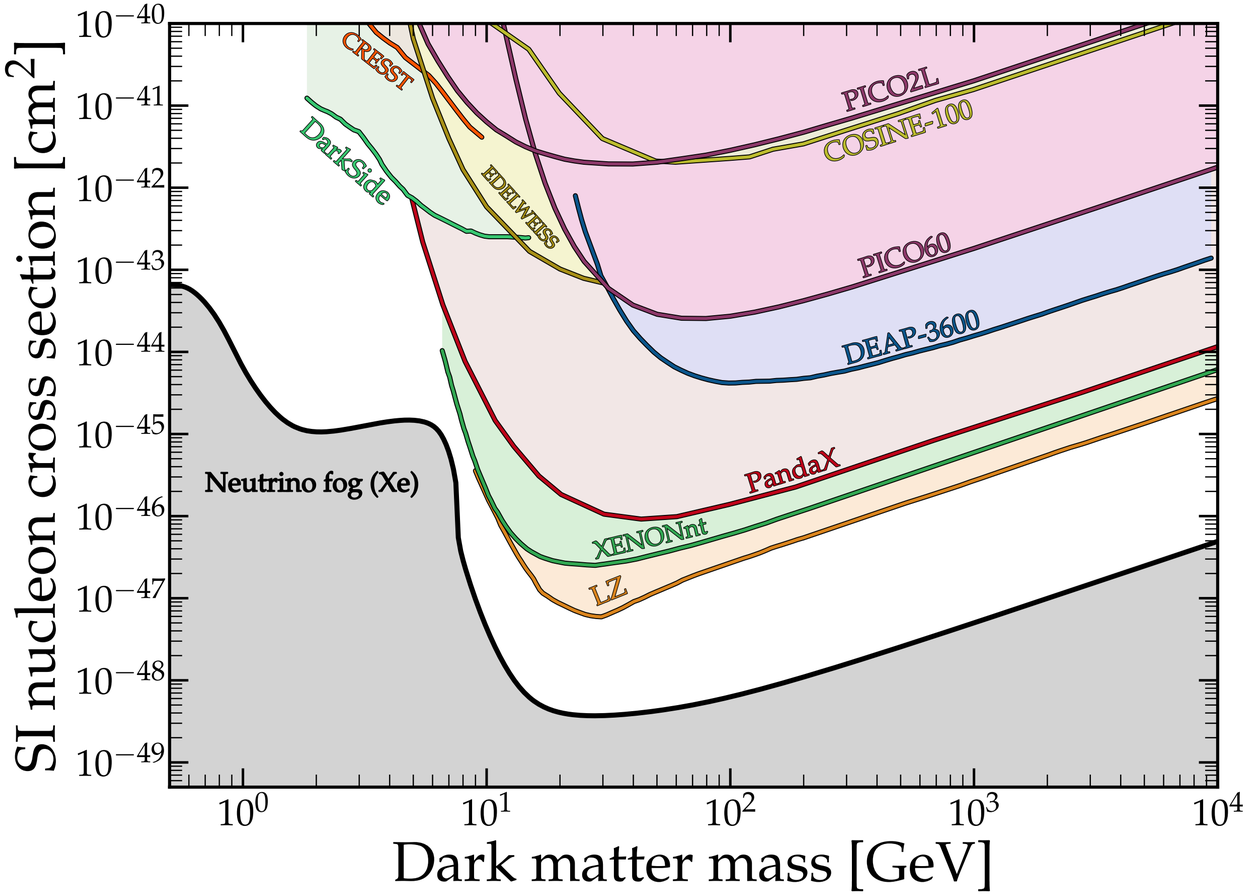


Inequalities and exclusion limits

One of my pet peeves is when people use naked lines as opposed to filled regions to display inequalities, e.g. exclusion limits. How is an uninitiated reader supposed to know which part is excluded?

ExclusionLimits.ipynb



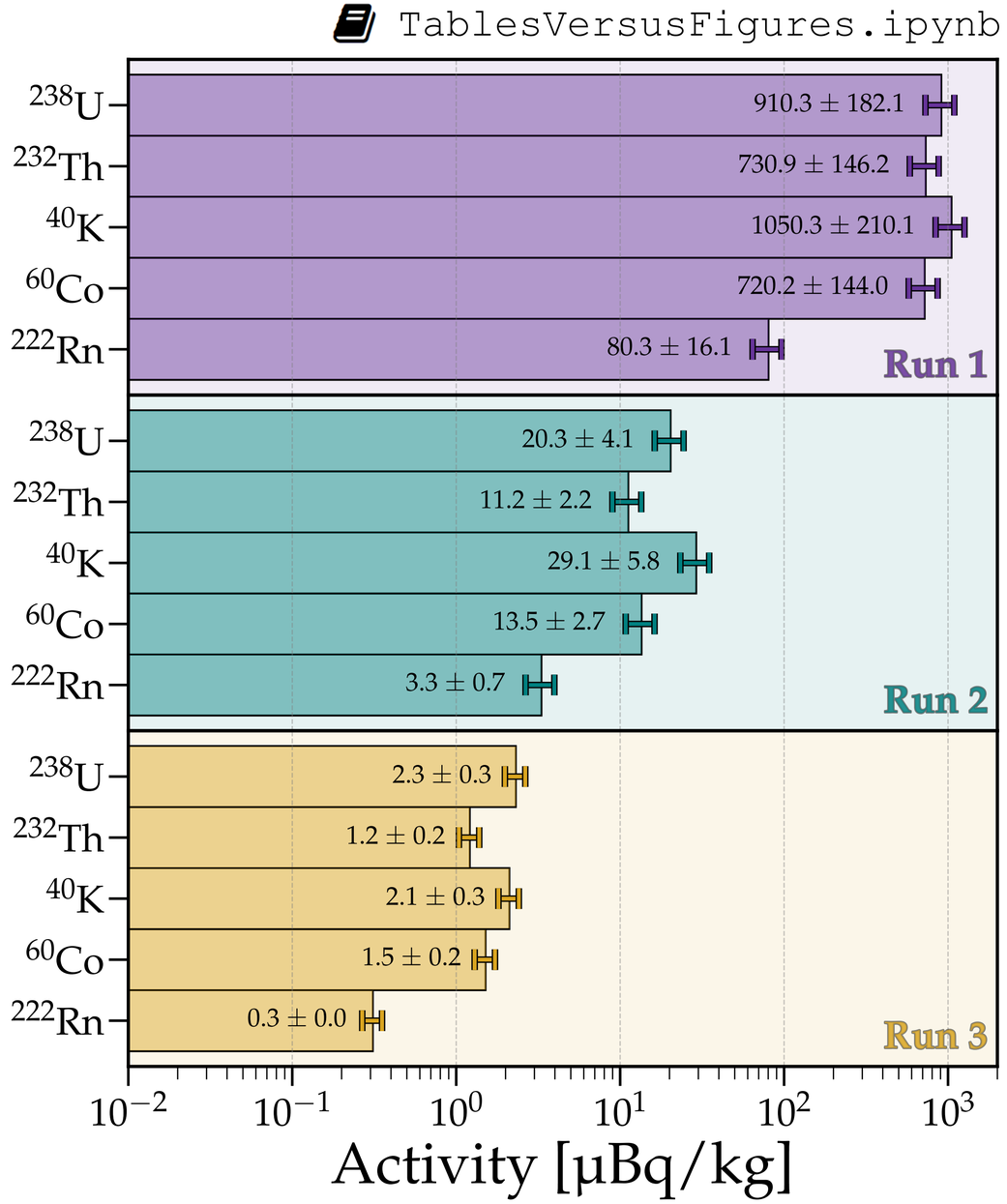


Tables versus figures

Tables are sometimes essential, e.g. if you need to record lots of numbers and the relationships between them. However, if the takeaway message of your paper is conveyed solely via referring to numbers in a table, consider turning that into figure

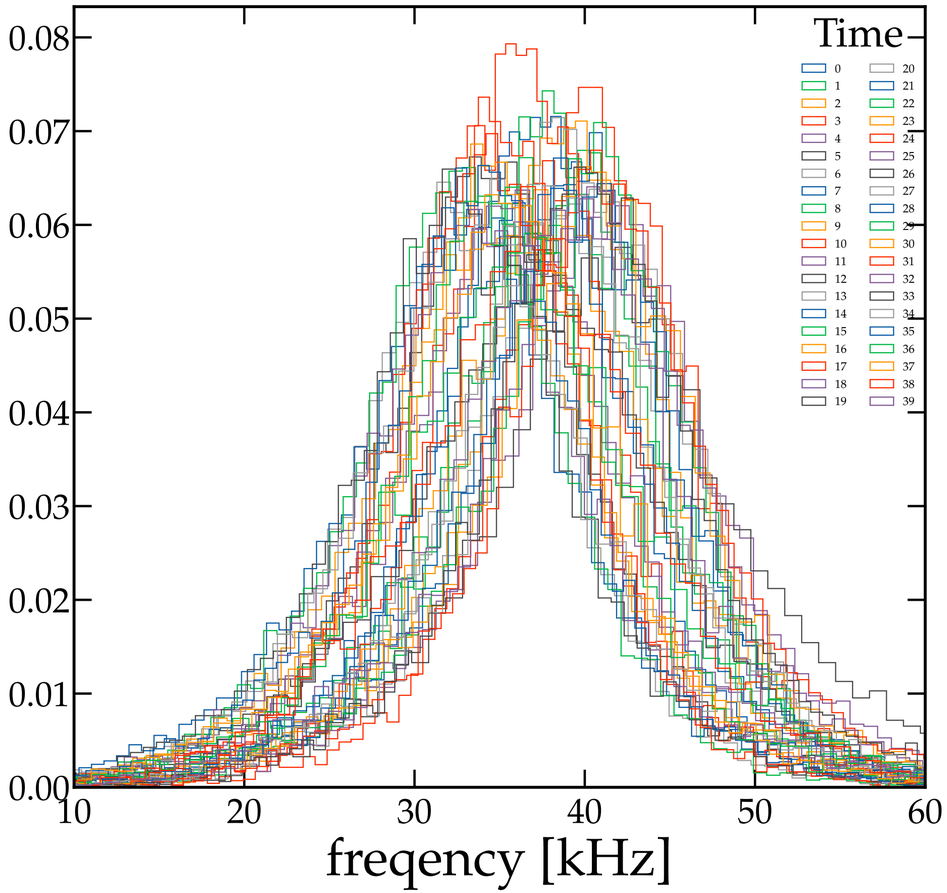
	^{238}U (μBq/kg)	^{232}Th (μBq/kg)	^{40}K (μBq/kg)	^{60}Co (μBq/kg)	^{222}Rn (μBq/kg)
Run 1	910.3 ± 182.1	730.9 ±146.2	1050.3 ± 210.1	720.2 ± 144.0	80.3 ± 16.1
Run 2	20.3 ± 4.1	11.2 ±2.2	29.1 ± 5.8	13.5 ± 2.7	3.3 ± 0.7
Run 3	2.3 ± 0.3	1.2 ± 0.2	2.1 ± 0.3	1.5 ± 0.2	0.3 ± 0.0

(And it goes without saying that tables are terrible things to show during talks)

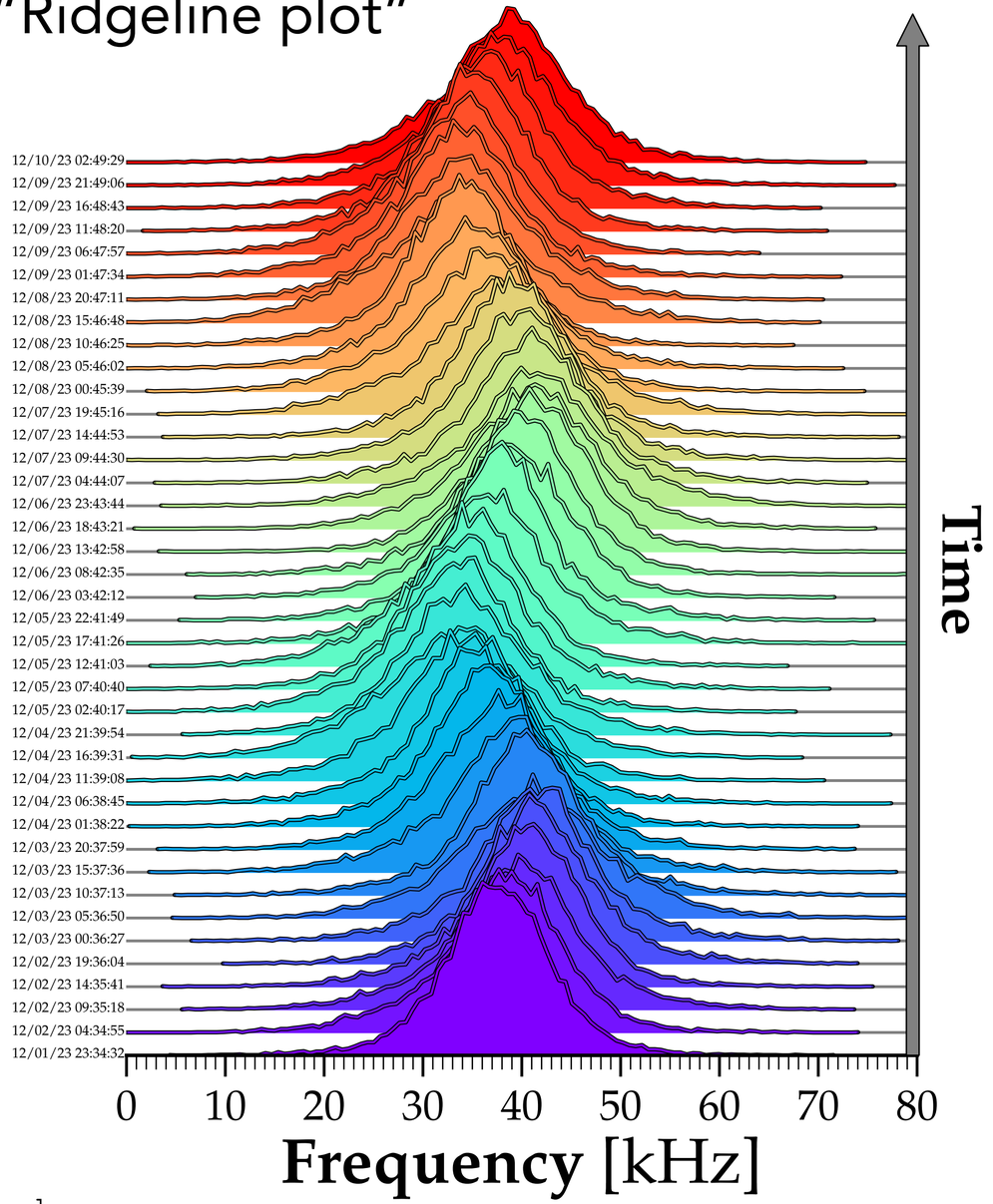


Histograms

Show up often, and can end up very confusing due to cluttered visual information.

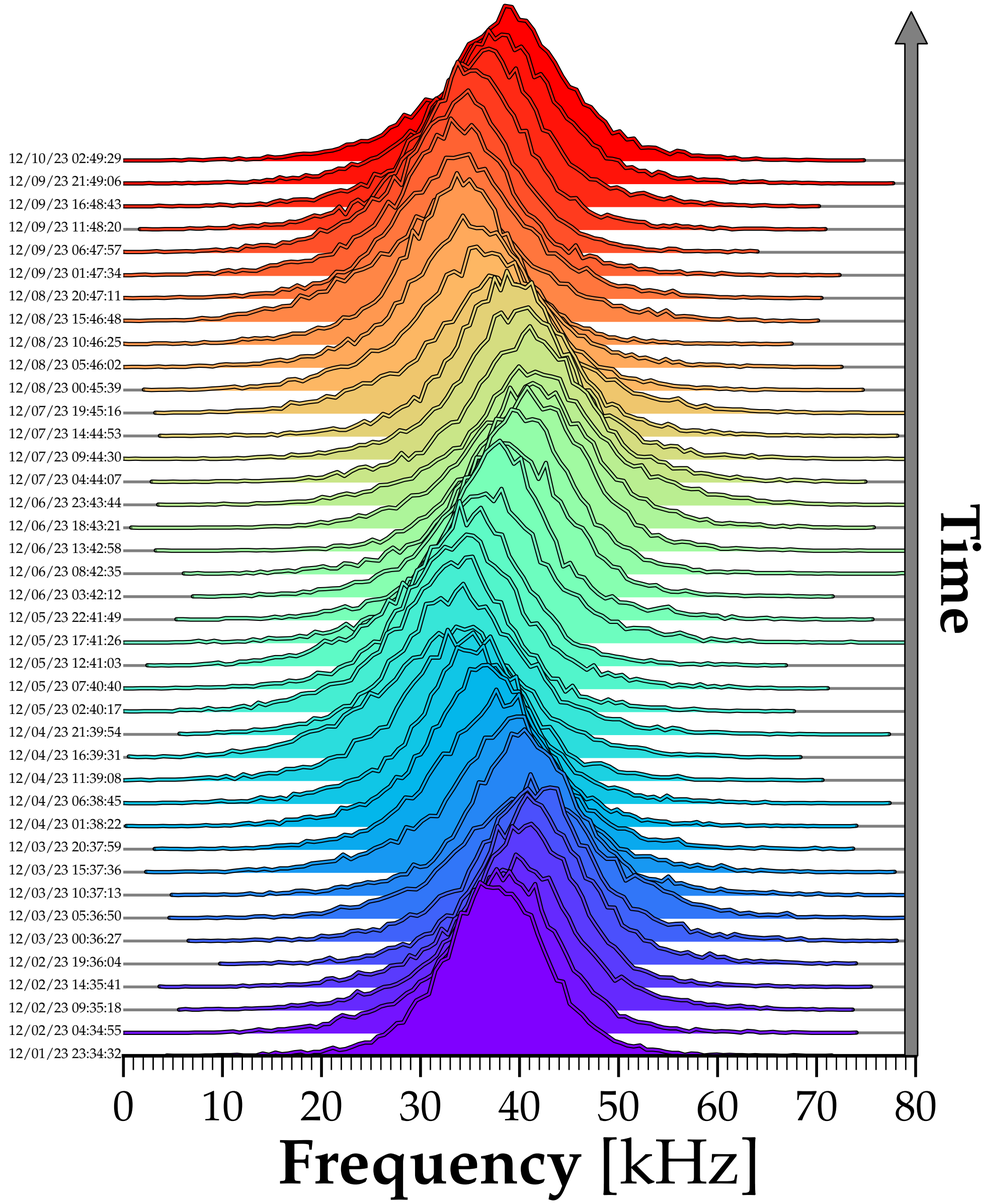


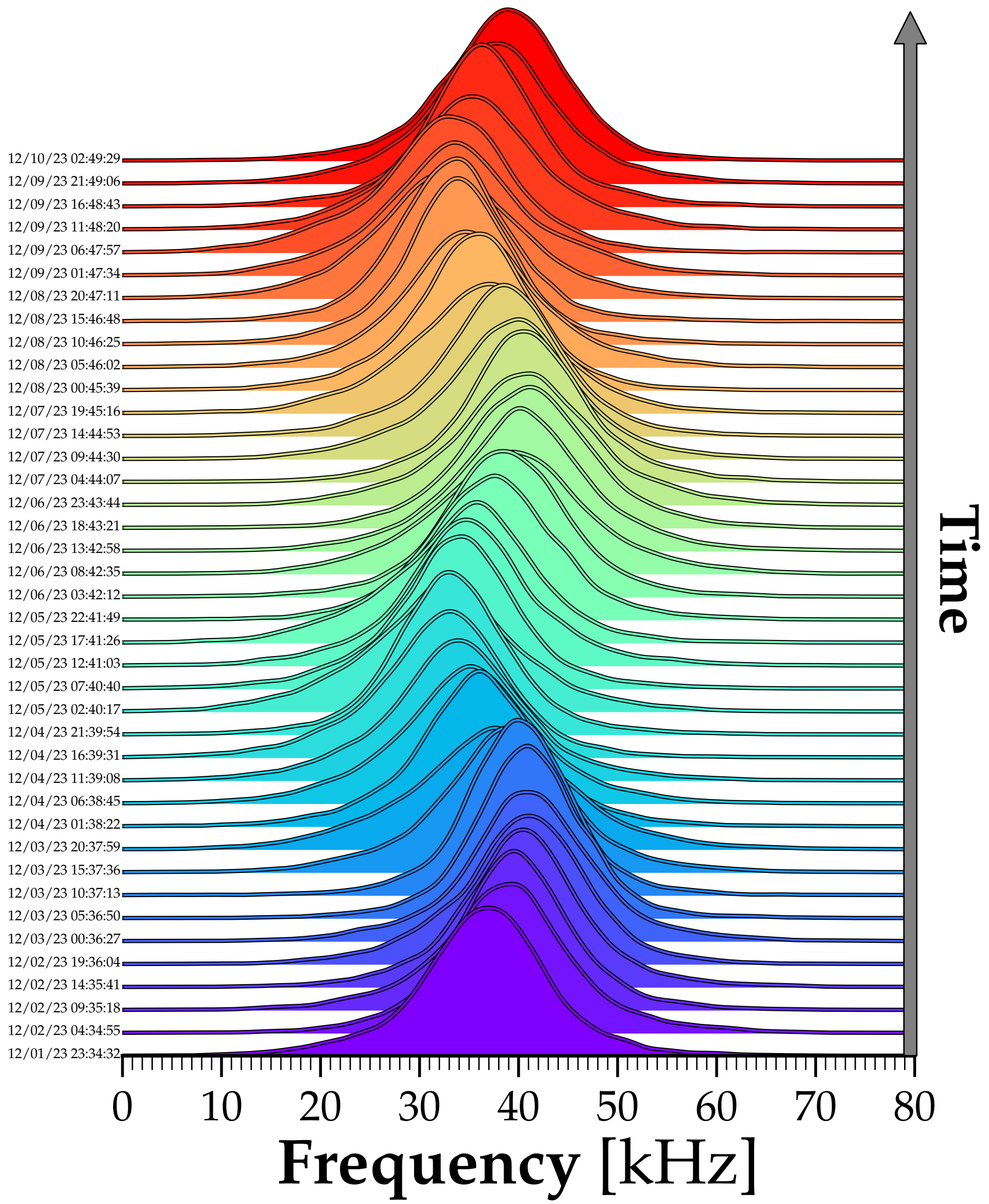
"Ridgeline plot"



Histograms.ipynb

You can improve clarity further by doing a kernel density estimation (kde) of the histogram



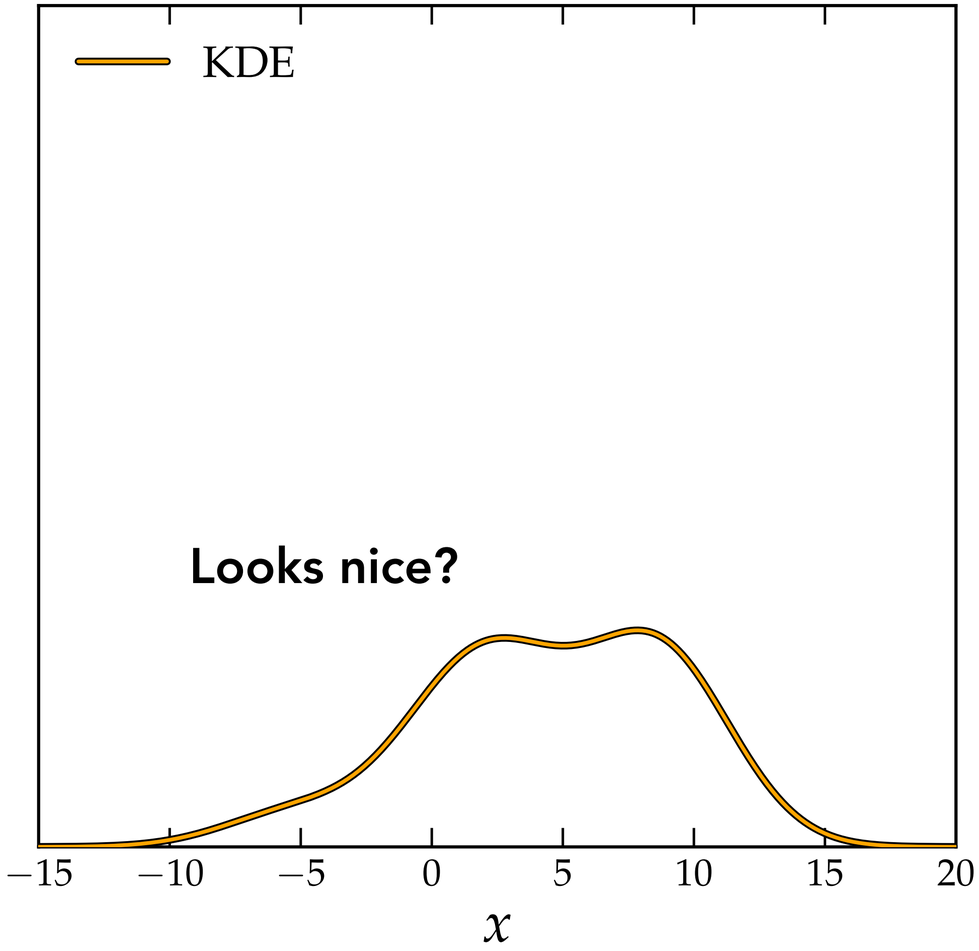


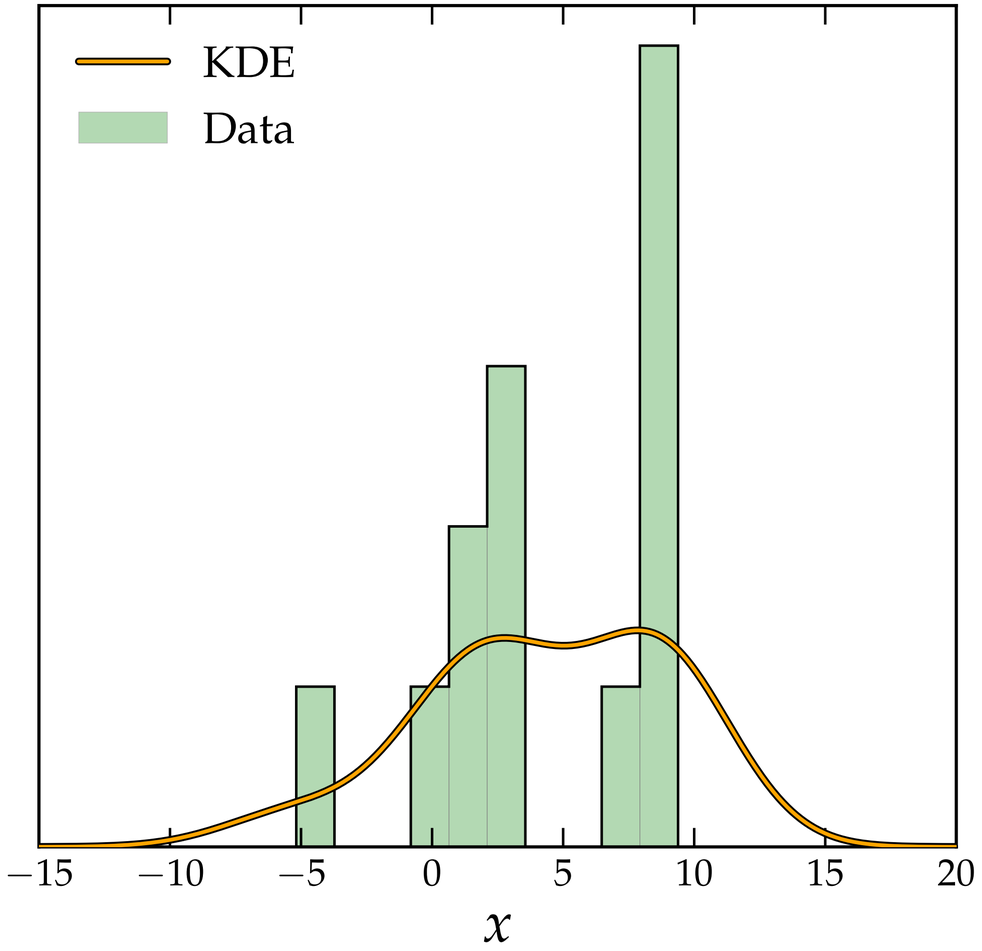
Histograms.ipynb

Lying by omission

Tempting to use kde to smooth out messy or sparse data. But good practice is to always show the underlying data when doing smoothing like this.

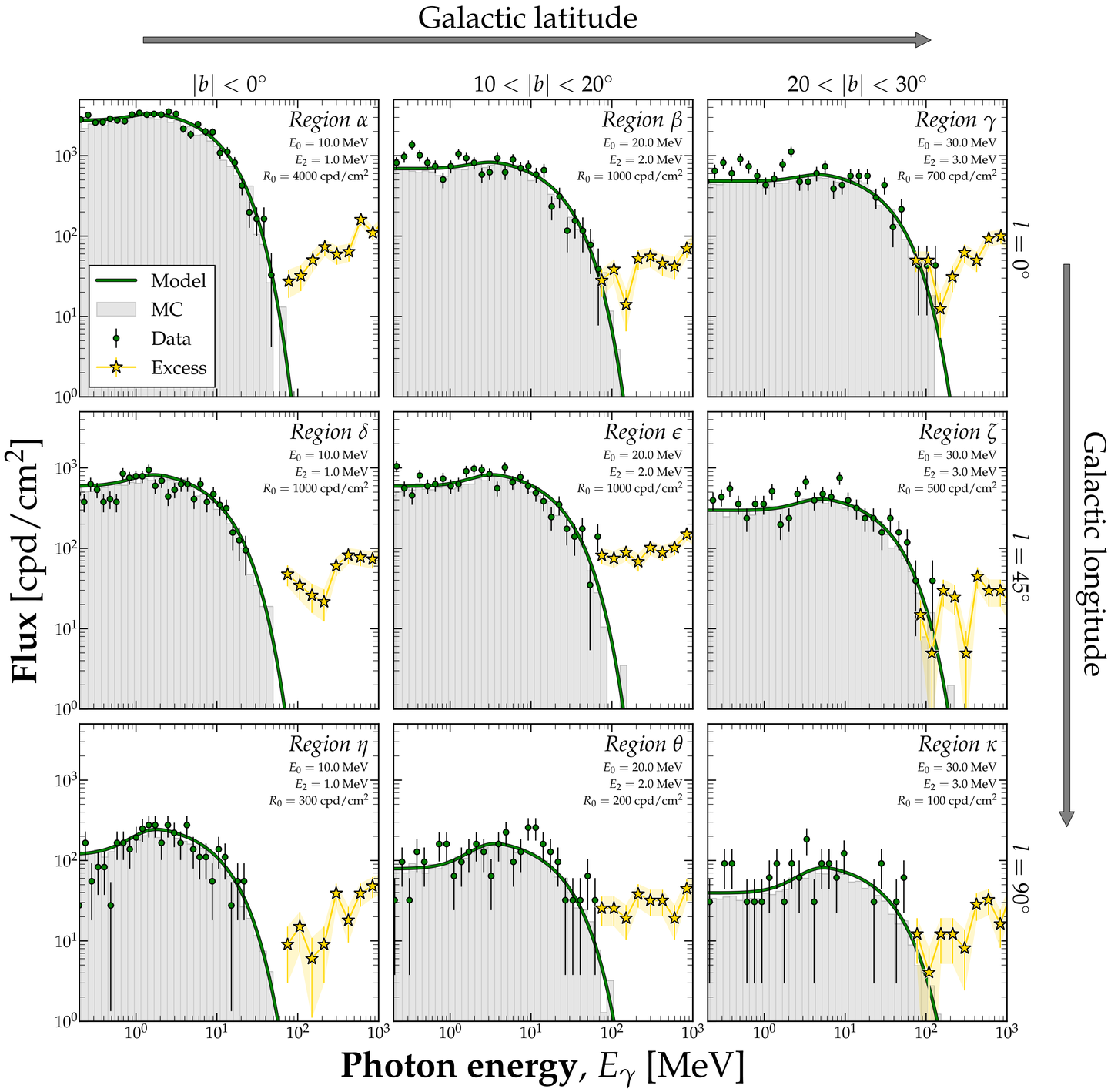
Histograms.ipynb





Complicated plots

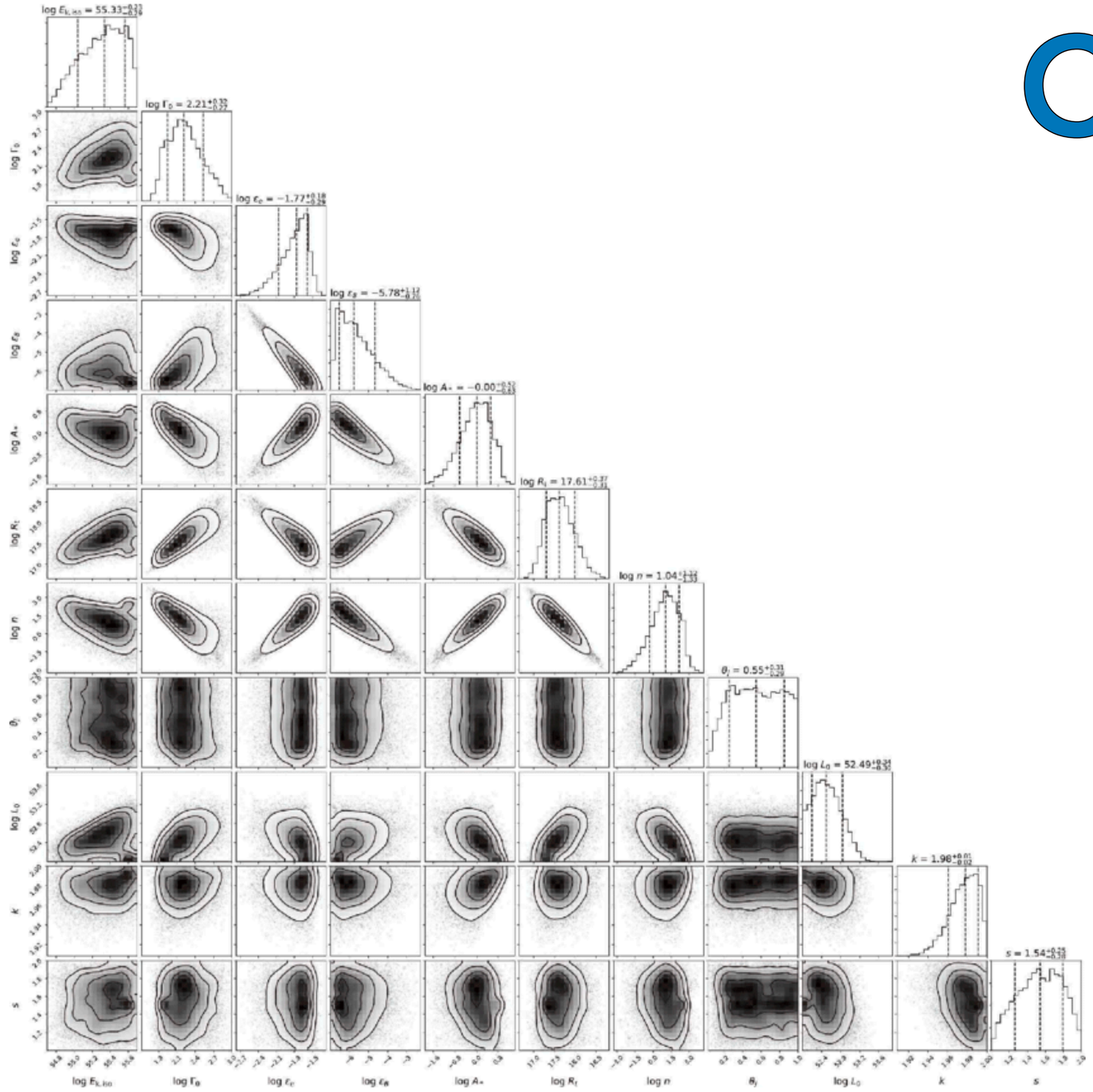
- Step 1: Avoid making complicated plots. Why does your plot need to be so complicated? Are you trying to impress someone? Because you won't - you will just confuse or annoy them. Simplify. Draw out the message you are trying to say and just show that.
- That said, sometimes plots in papers do get complicated. In those cases, you still want to maximise the time people spend thinking about your message rather than just figuring out your plot's internal logic.
- Use pre-attentive attributes to express multiple layers of information that the reader can appreciate in stages.
- Use labels liberally. Do not force the reader to play tennis between the figure and the caption



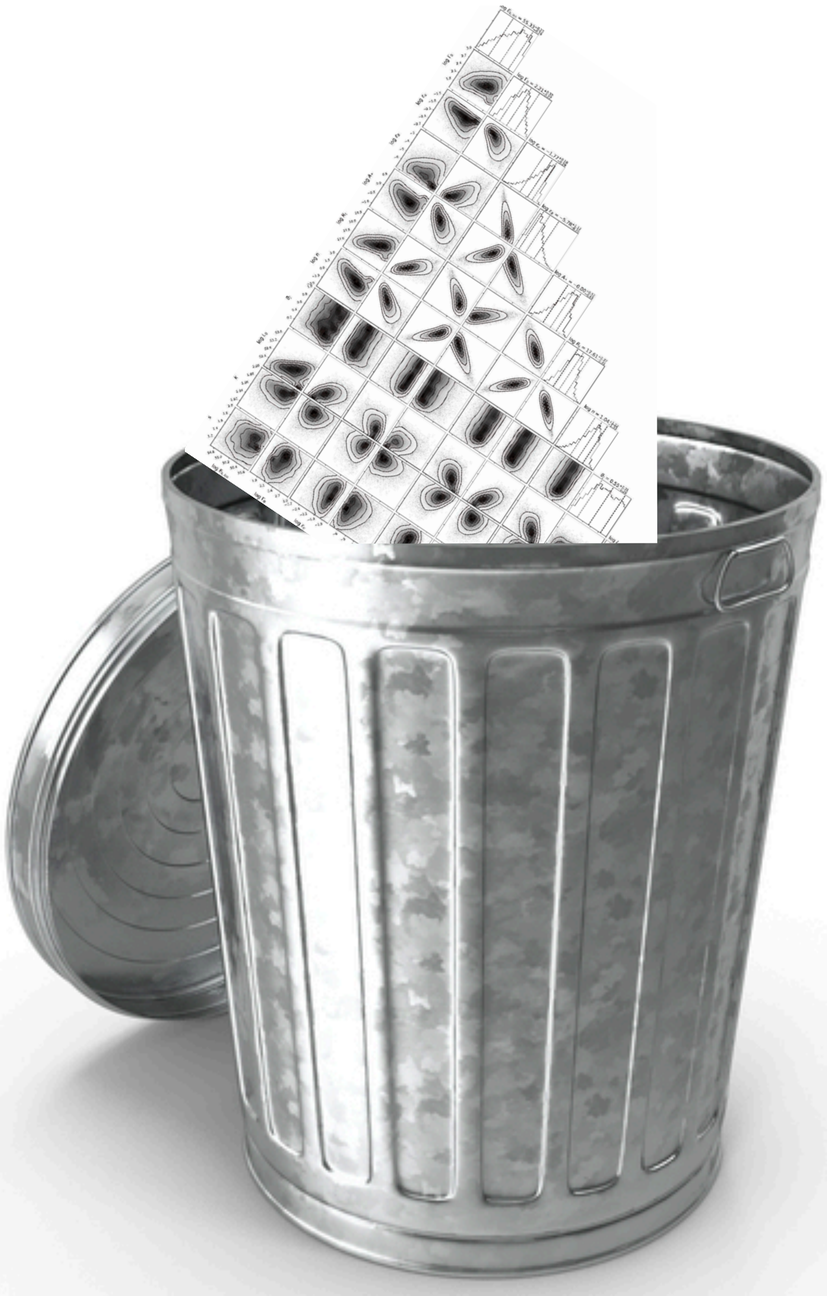
ComplicatedPlots.ipynb

Corner plots

...

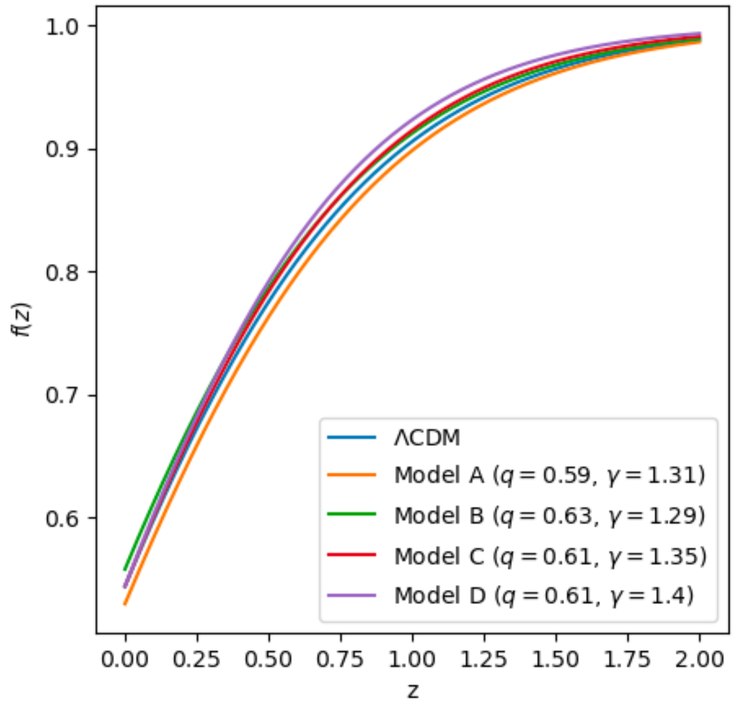


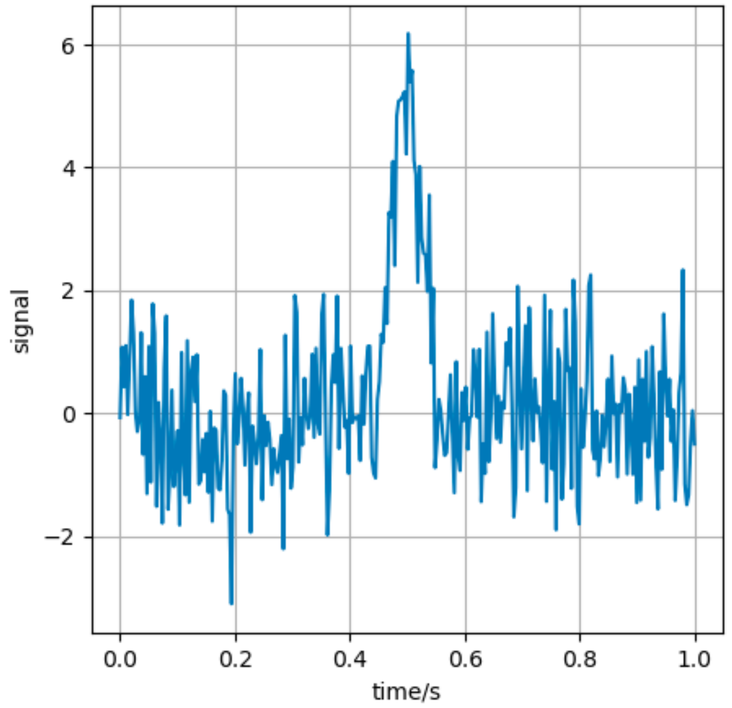
Corner plots

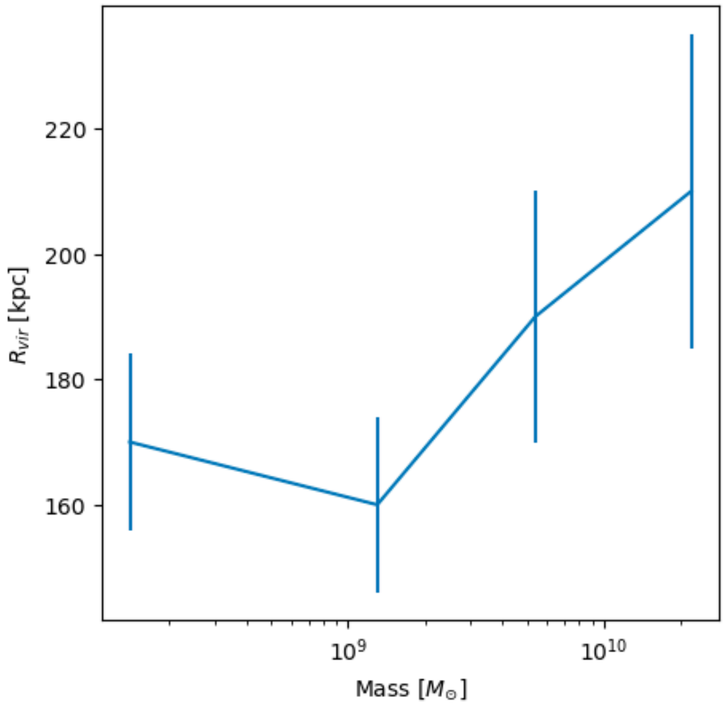


Sometimes your data is just ugly or uninteresting

It’s hard to get excited by an ugly plot, even if the point it is making is exciting.





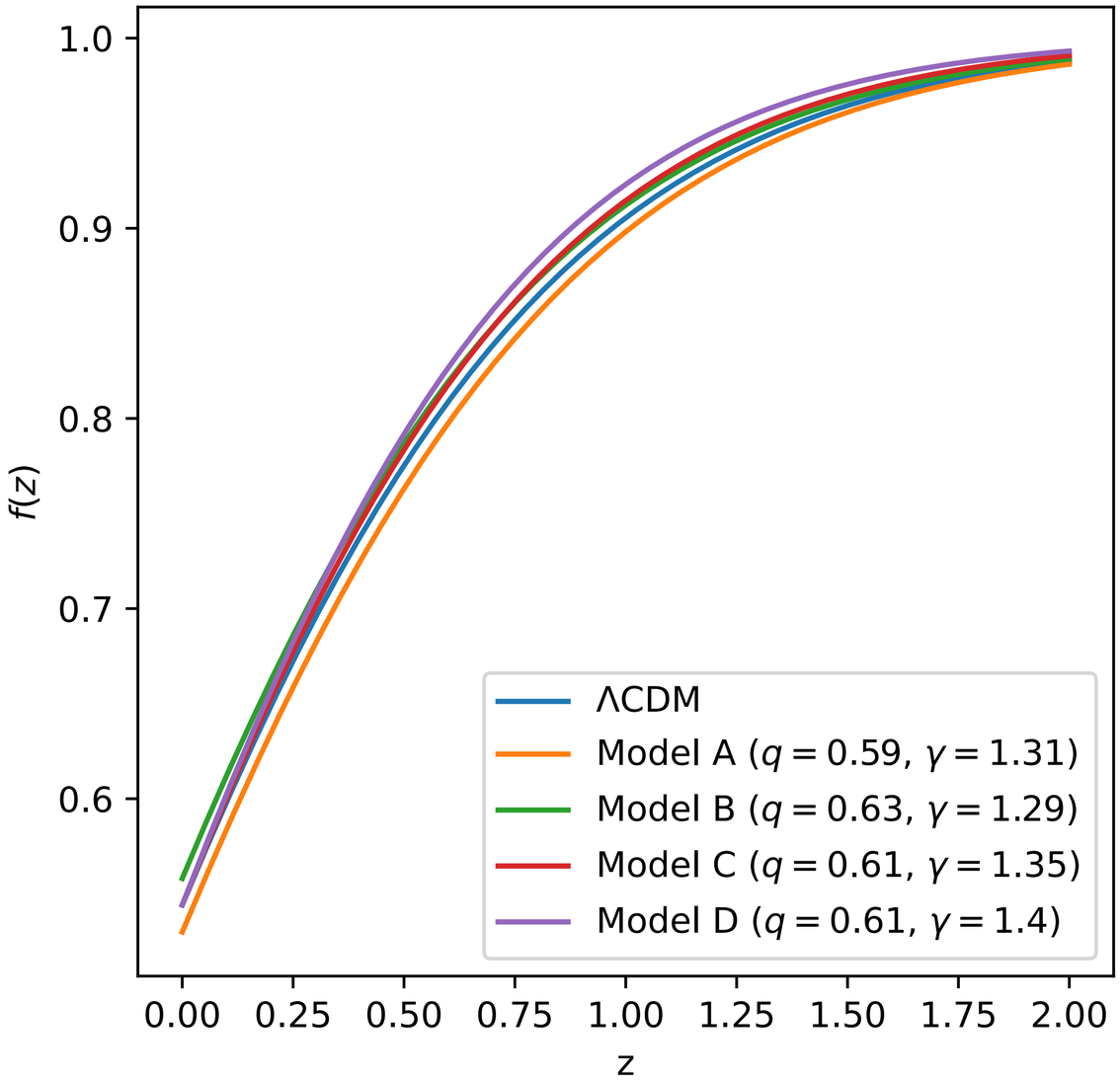


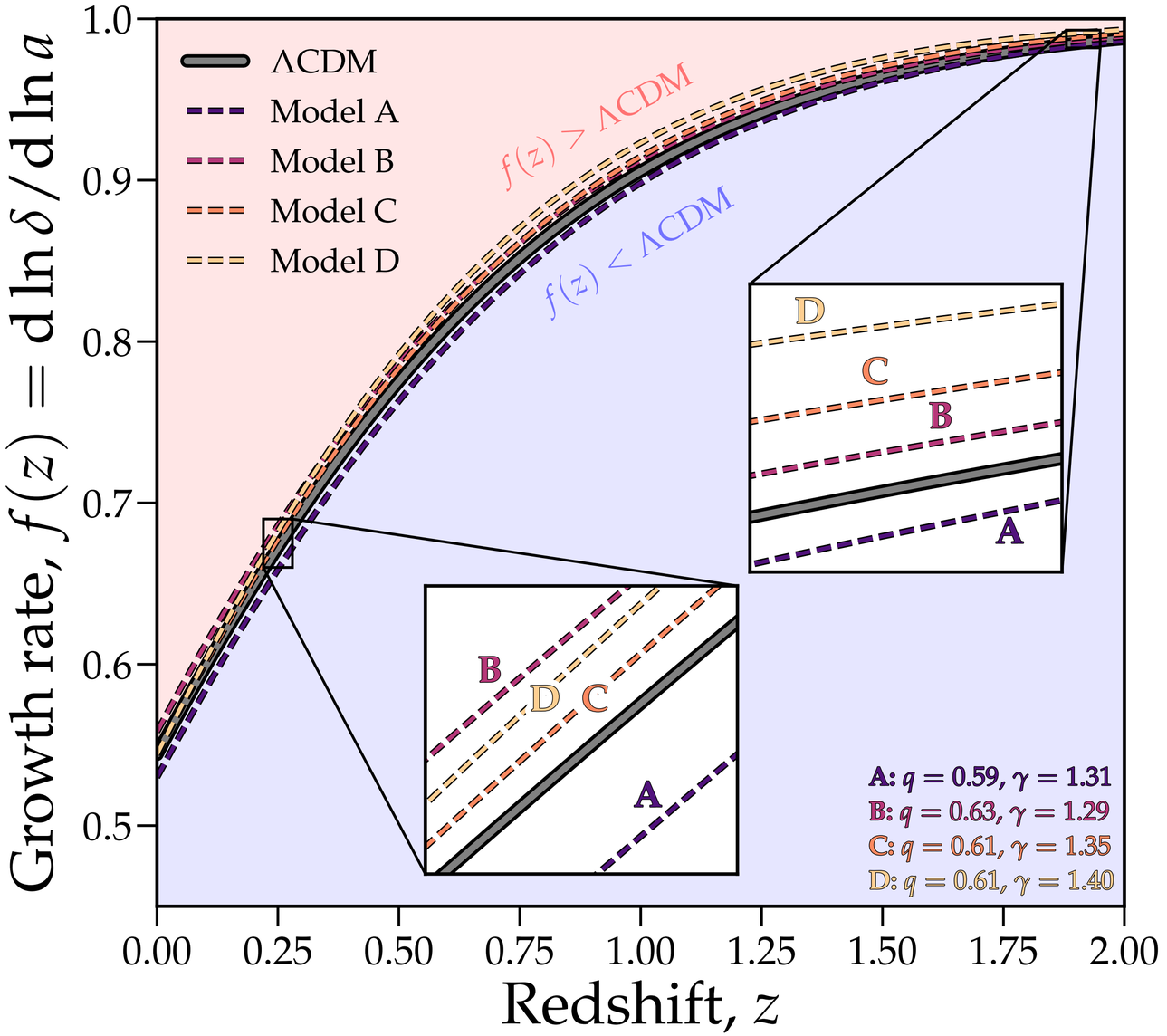
Remedy these cases by making the plot as pretty as possible and using the fact that the data is uninteresting to make the message even clearer. Some of the best plots are actually the simplest

Example 1

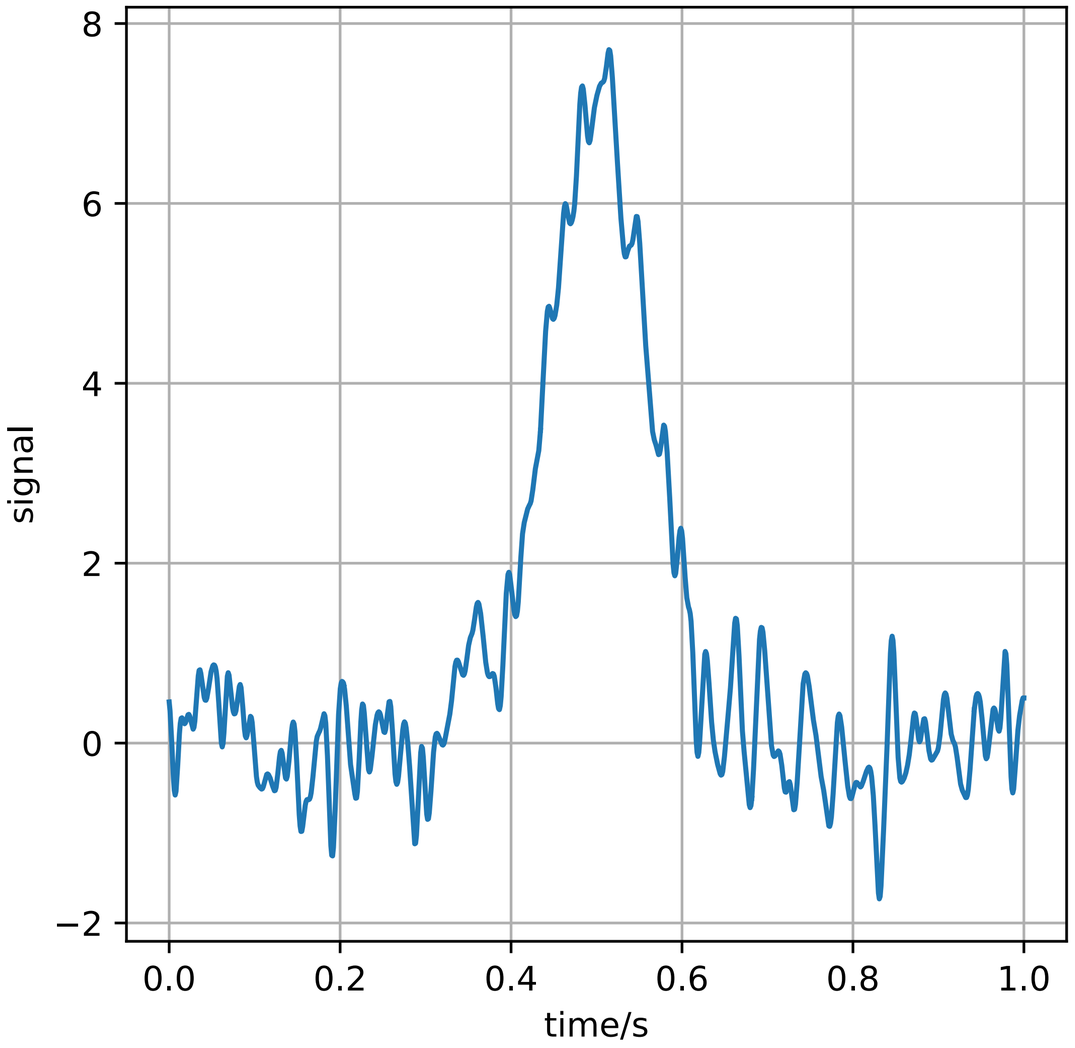
Tip: Adding inset axes to plots in matplotlib is much easier than it seems
See: https://matplotlib.org/stable/gallery/axes_grid1/inset_locator_demo.html

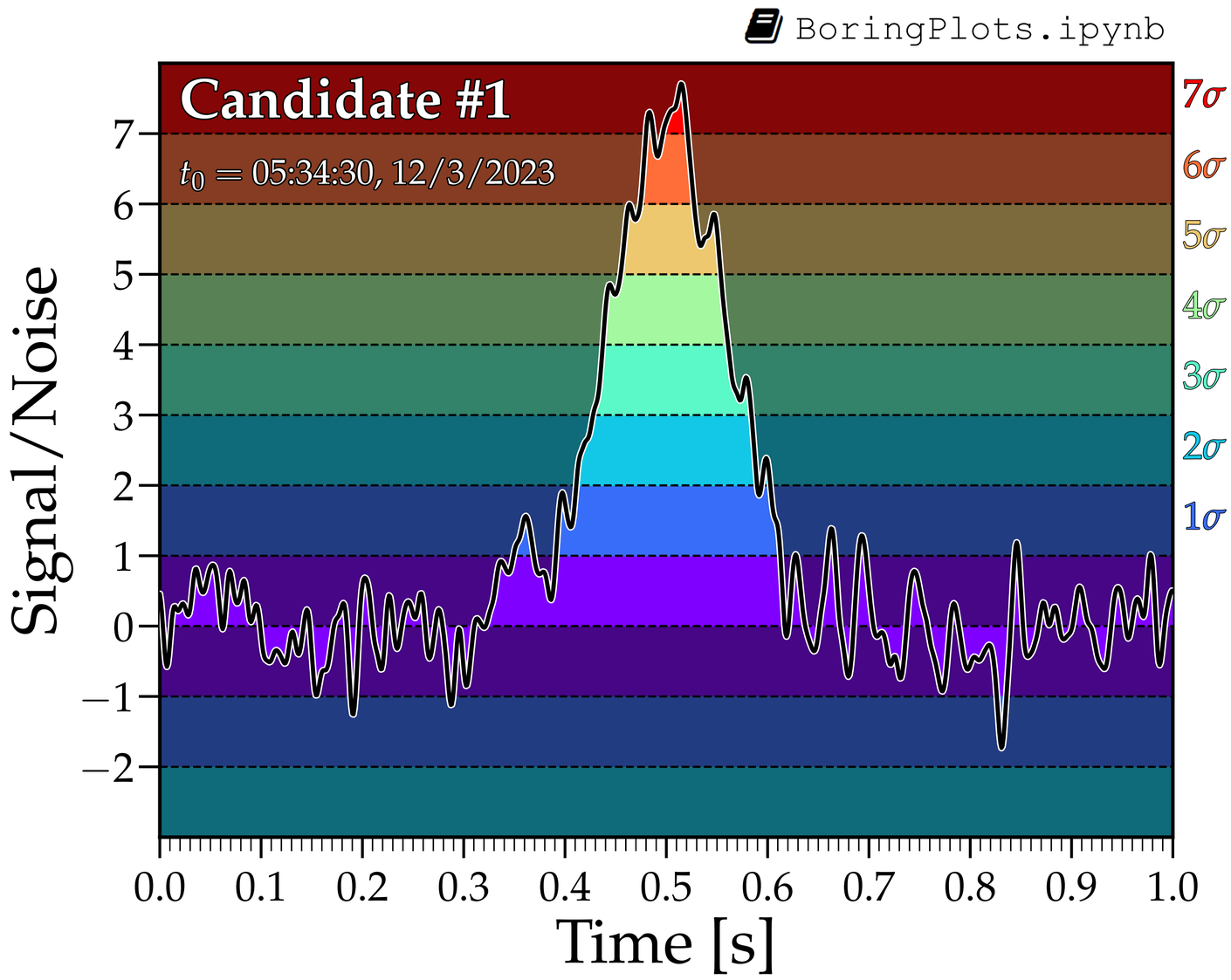
BoringPlots.ipynb



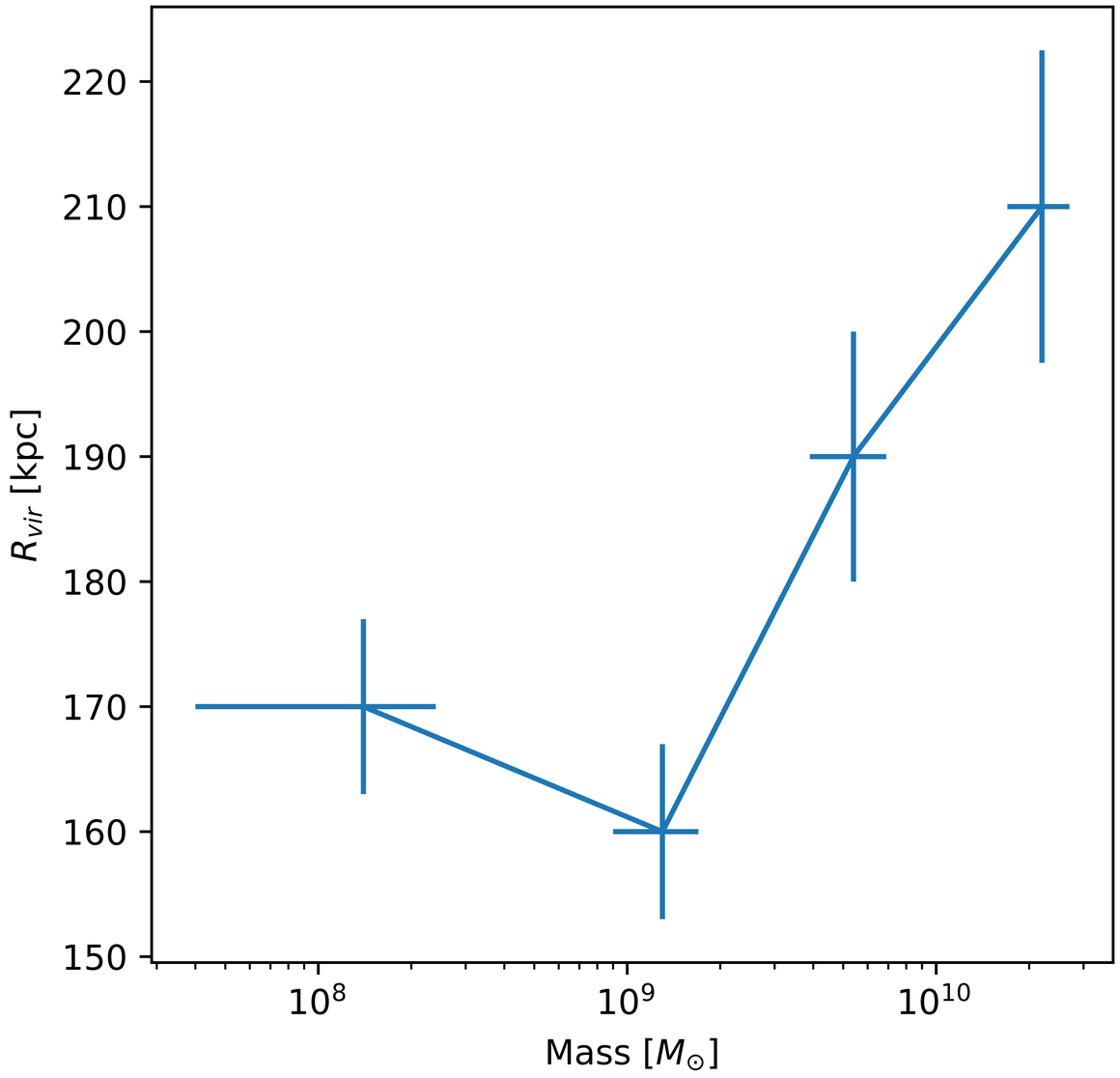


Example 2

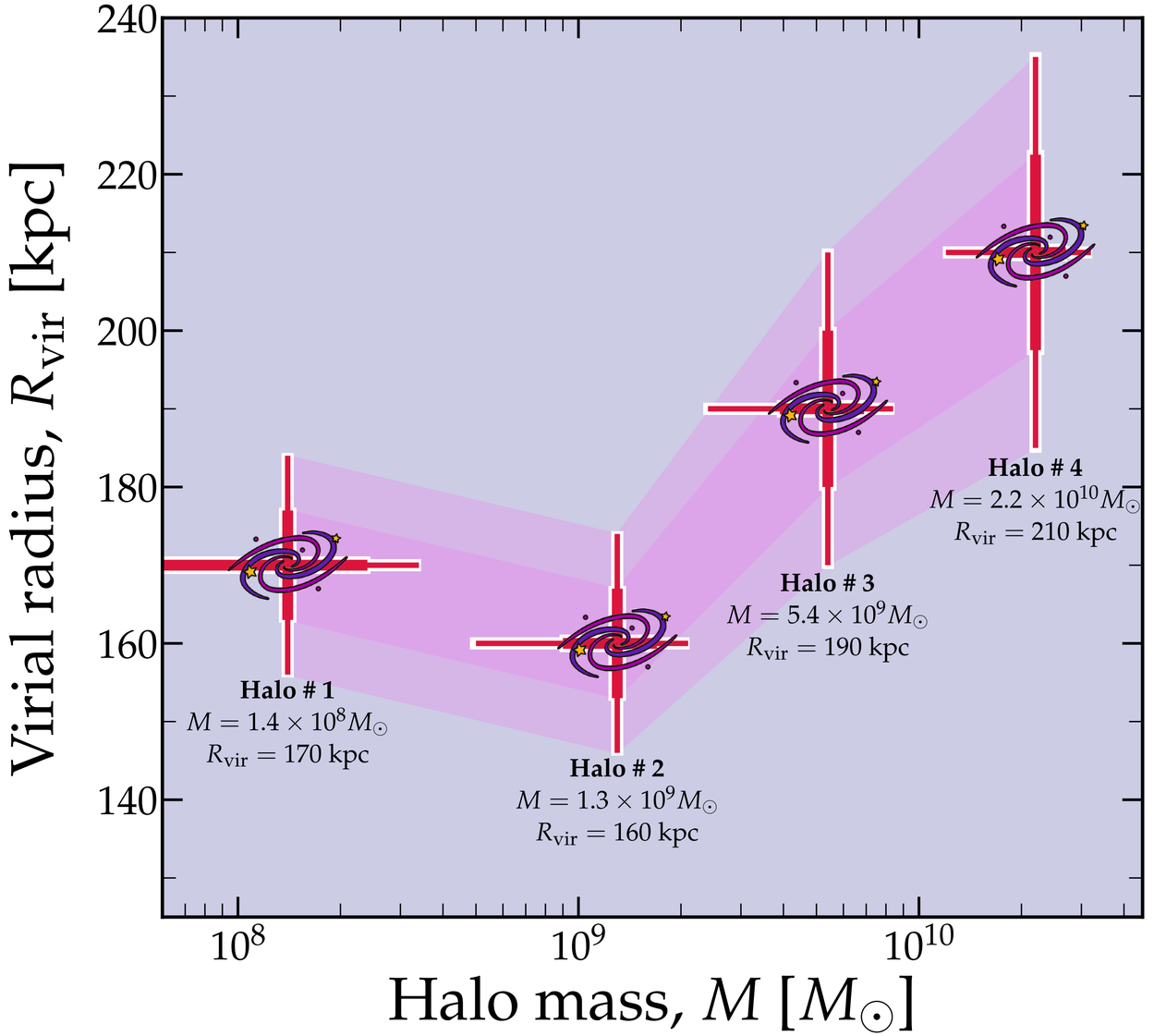

signal
time/s
8
6
4
2
0
−2
0.0
0.2
0.4
0.6
0.8
1.0


BoringPlots.ipynb
Candidate #1
t_0 = 05:34:30, 12/3/2023
Signal/Noise
Time [s]
7σ
6σ
5σ
4σ
3σ
2σ
1σ
7
6
5
4
3
2
1
0
−1
−2
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0

Example 3



BoringPlots.ipynb



Practical/stylistic tips

- Defaults
- Style sheets
- Colours
- Fonts

Disclaimers

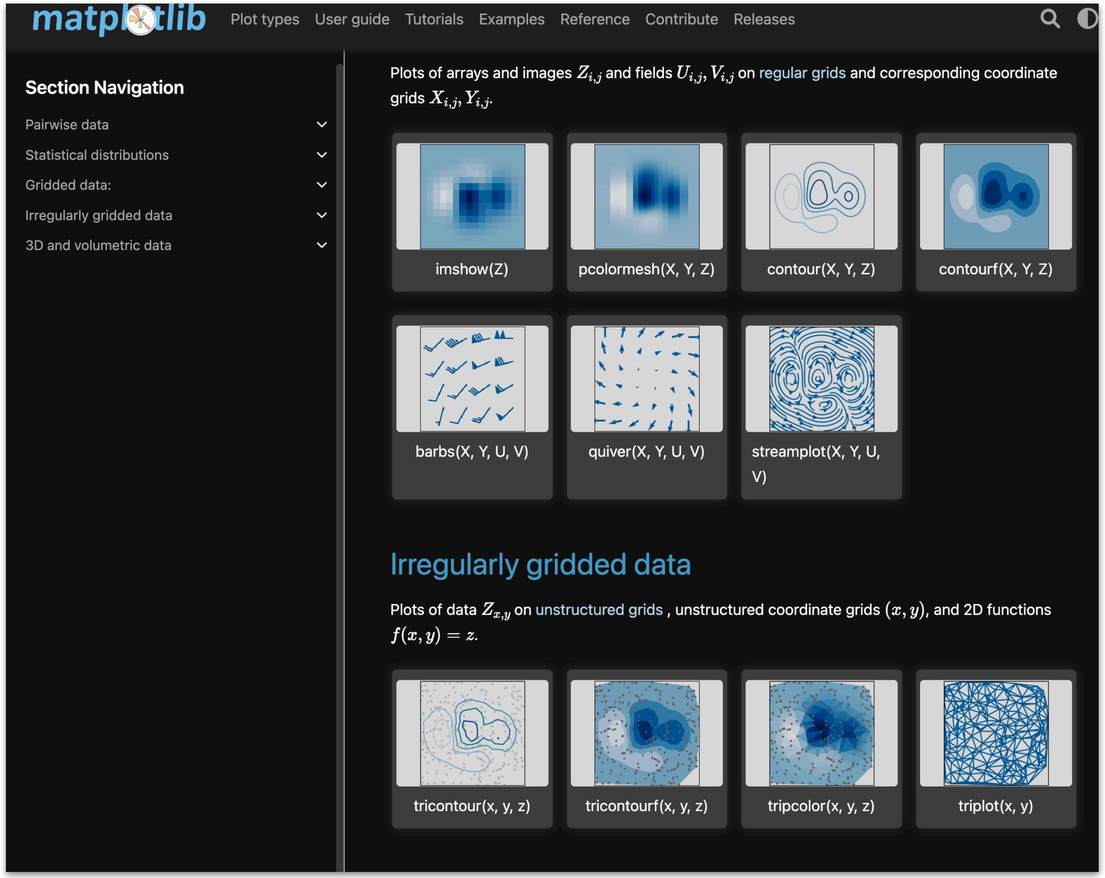
- I will use the example of python/matplotlib for concreteness, though basically all of this advice can be translated to other software like R, Tikz, Mathematica, MATLAB, ROOT, etc.

- For the sake of your own sanity, I recommend using notebooks (e.g. jupyter) to make plots, at least when you are still in the creation phase

  (NB: I do not condone the use notebooks for all your work)

General tips for matplotlib

- Matplotlib is kind of annoying. This isn't a tip, but if you're struggling don't worry, it's probably not your fault.
- Documentation is quite arcane, but they have tons of examples. You'll get much better mileage by extending pre-existing code that approximates what you want.
- Google images is a good way to find those examples if you don't know exactly the right words to describe what you want.
- ChatGPT is pretty good for the initial setup of plots and to get over any syntax barriers



https://matplotlib.org/stable/plot_types/index.html

Matplotlib style sheets

- Style sheets (`.mplstyle` files) are a way to change any of the default settings for making plots and have them hold over to all subsequent plots.

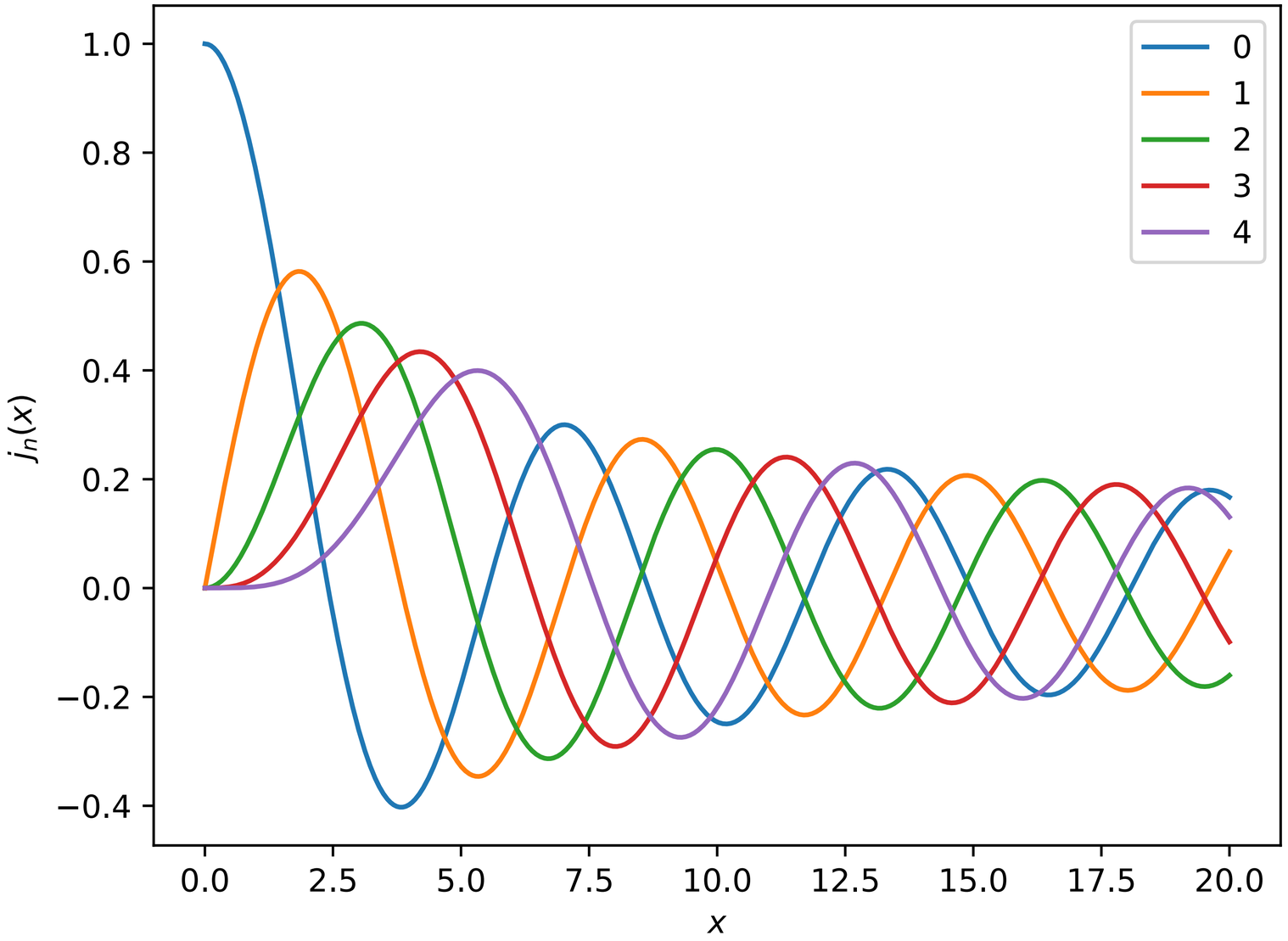
- See here for more details on how to construct them. Just about every default setting can be changed, including tick, axis and figure properties, default line colours, fonts etc.

```
# Set default figure size
figure.figsize : 13, 12

# Set x axis
xtick.major.size : 15
xtick.major.width : 2
xtick.minor.size : 10
xtick.minor.width : 1
xtick.direction : in
xtick.top : True

# Set y axis
ytick.major.size : 15
```

Matplotlib default style

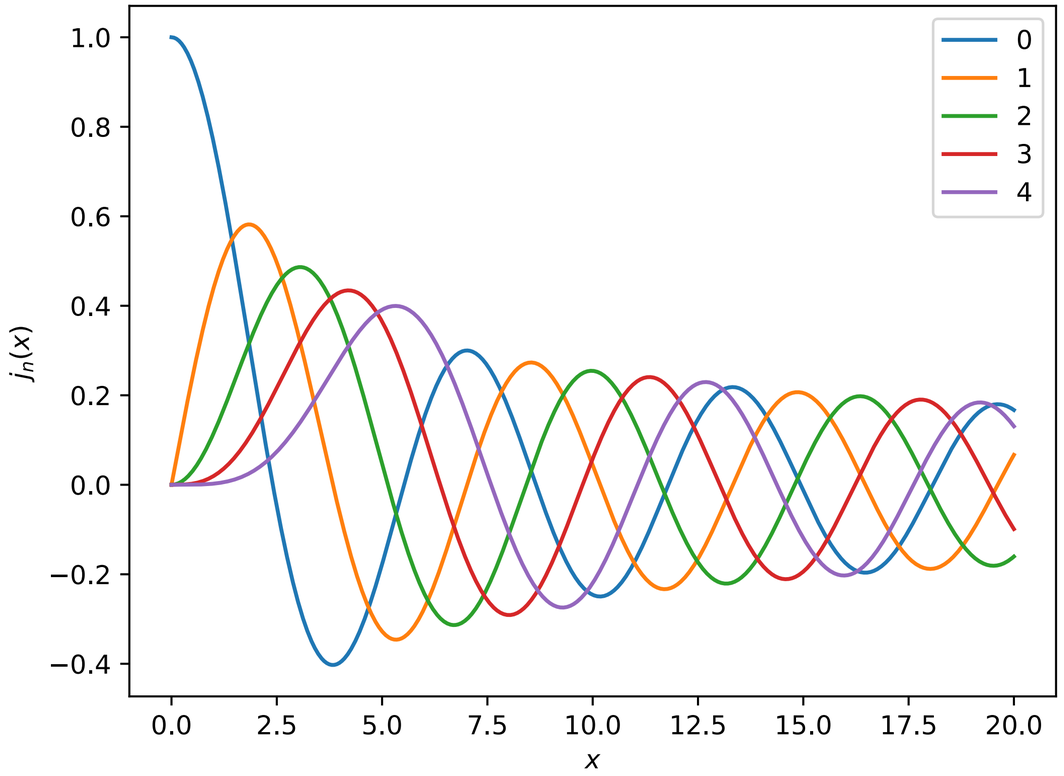


There is nothing really *wrong* with this plot, but there is also nothing special about it either. Its greatest crime is just that it is relying on the default Matplotlib settings, which is immediately obvious to anyone who uses python

Example of a style sheet

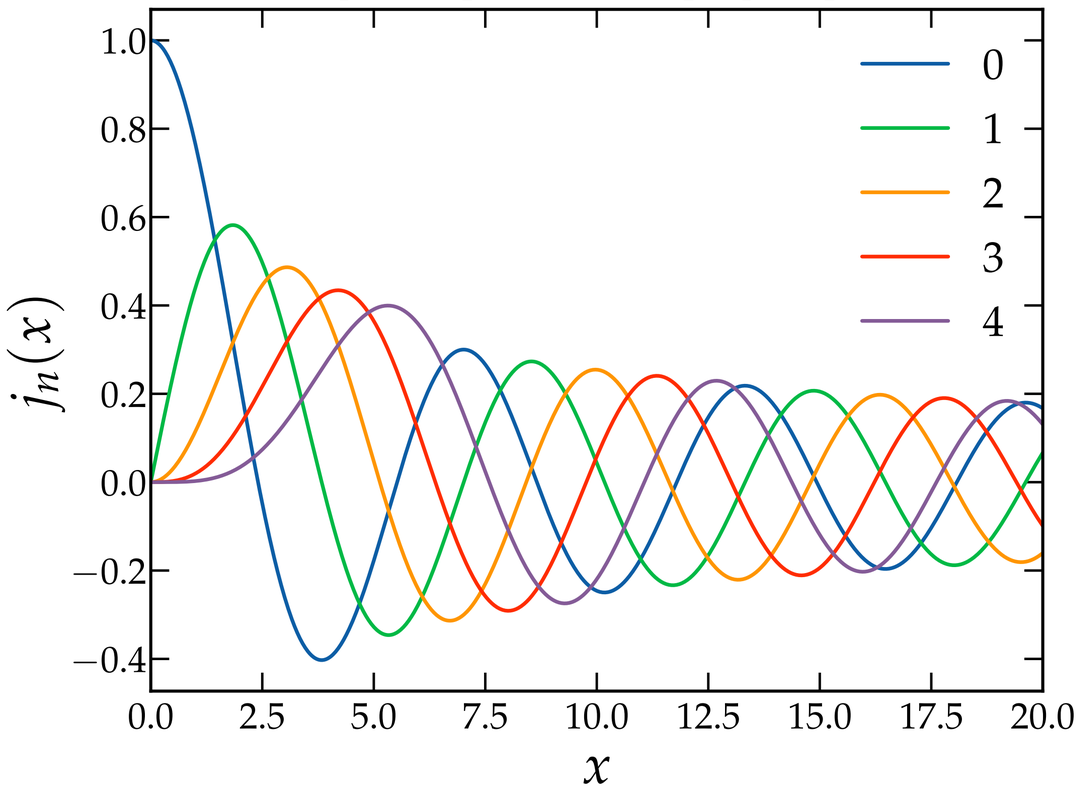
Using my style sheet ('sty.mplstyle' provided in the Confluence materials), you can improve the look of your plot with a single line of code.

Default



Style sheet

→ `plt.style.use('sty.mplstyle')`

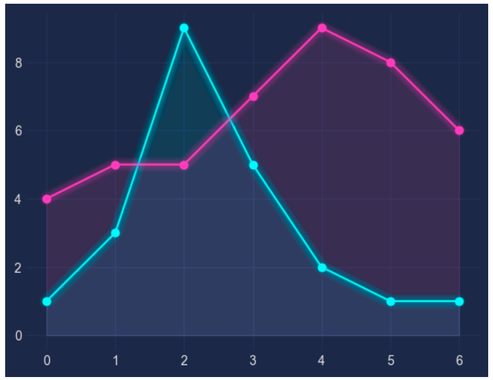


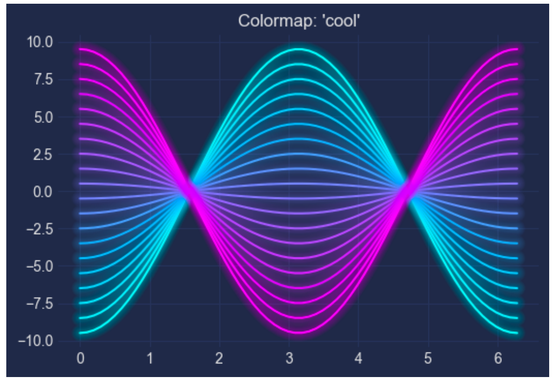
`Defaults.ipynb`

Alternative styles

Cyperpunk

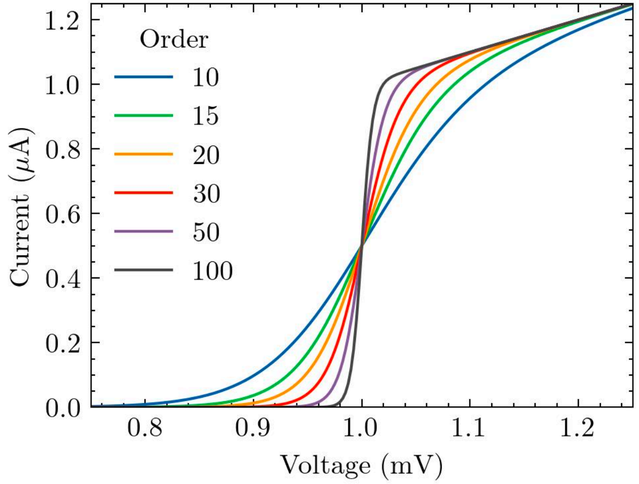
https://github.com/dhaitz/mplcyberpunk

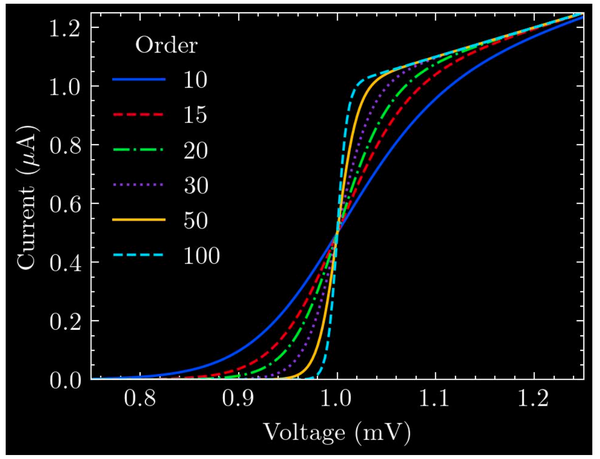




Science plots

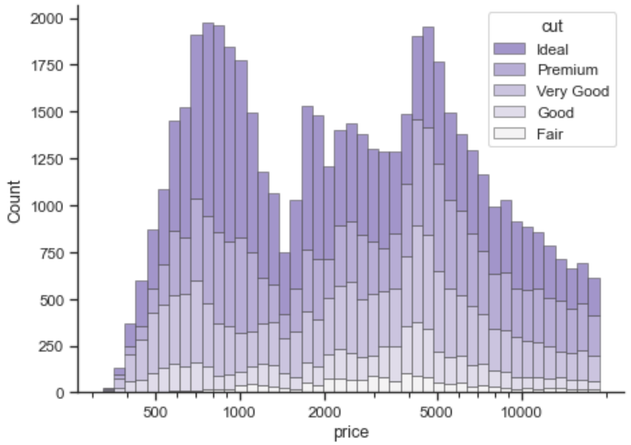
https://github.com/garrettj403/SciencePlots

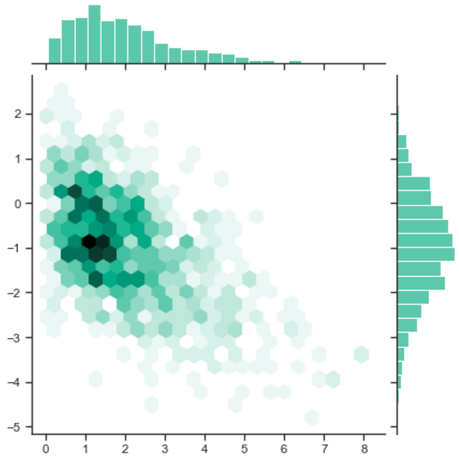




Seaborn

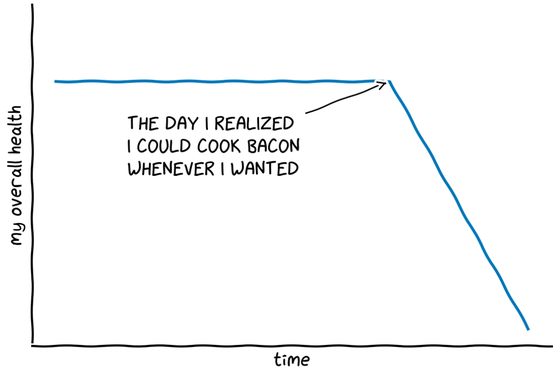
https://seaborn.pydata.org/



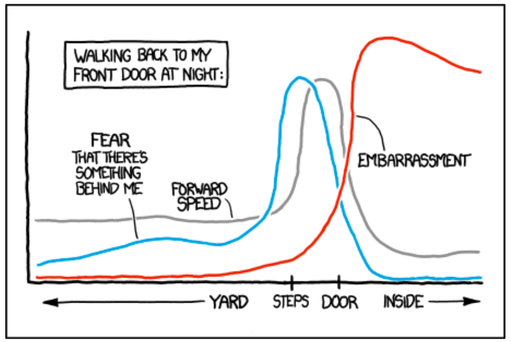


xkcd

https://matplotlib.org/stable/gallery/showcase/xkcd.html



"Stove Ownership" from xkcd by Randall Munroe



Picking colours

Matplotlib has a long list of confusingly named colours:

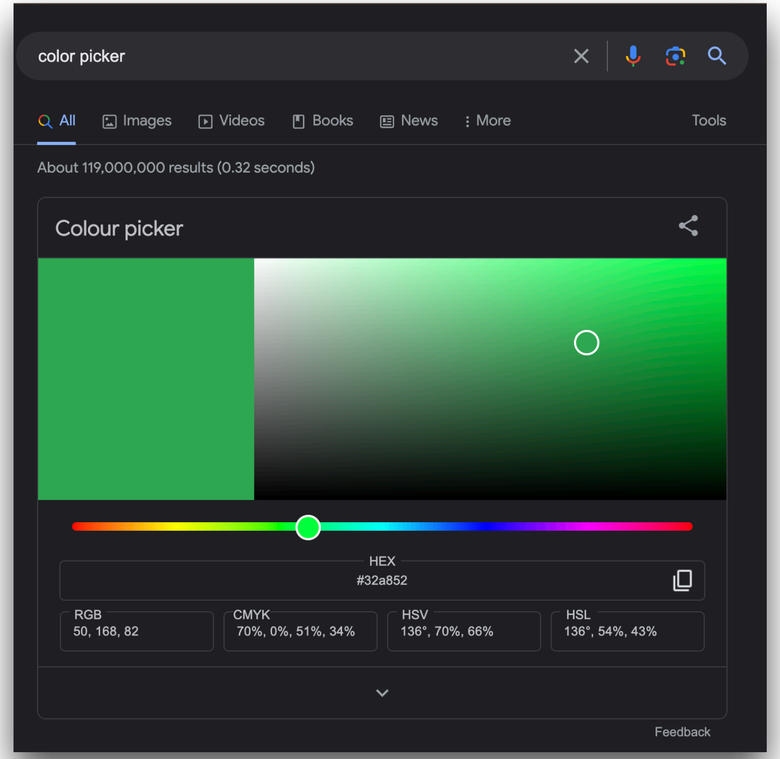
black
gray
silver
whitesmoke
rosybrown
firebrick
red
darksalmon
sienna
sandybrown
bisque
tan
moccasin
floralwhite
gold
darkkhaki
lightgoldenrodyellow
olivedrab
chartreuse
palegreen
darkgreen
seagreen
mediumspringgreen
lightseagreen
paleturquoise
darkcyan
darkturquoise
deepskyblue
aliceblue
slategray
royalblue
navy
blue
mediumpurple
darkorchid
plum
m
mediumvioletred
palevioletred

k
grey
lightgray
w
lightcoral
maroon
mistyrose
coral
seashell
peachpuff
darkorange
navajowhite
orange
darkgoldenrod
lemonchiffon
ivory
olive
yellowgreen
lawngreen
lightgreen
g
mediumseagreen
mediumaquamarine
mediumturquoise
darkslategray
c
cadetblue
skyblue
dodgerblue
slategrey
ghostwhite
darkblue
slateblue
rebeccapurple
darkviolet
violet
fuchsia
deeppink
crimson

dimgray
darkgray
lightgrey
white
indianred
darkred
salmon
orangered
chocolate
peru
burlywood
blanchedalmond
wheat
goldenrod
khaki
beige
y
darkolivegreen
honeydew
forestgreen
green
springgreen
aquamarine
azure
darkslategrey
aqua
powderblue
lightskyblue
lightslategray
lightsteelblue
lavender
mediumblue
darkslateblue
blueviolet
mediumorchid
purple
magenta
hotpink
pink

dimgrey
darkgrey
gainsboro
snow
brown
r
tomato
lightsalmon
saddlebrown
linen
antiquewhite
papayawhip
oldlace
cornsilk
palegoldenrod
lightyellow
yellow
greenyellow
darkseagreen
limegreen
lime
mintcream
turquoise
lightcyan
teal
cyan
lightblue
steelblue
lightslategrey
cornflowerblue
midnightblue
b
mediumslateblue
indigo
thistle
darkmagenta
orchid
lavenderblush
lightpink

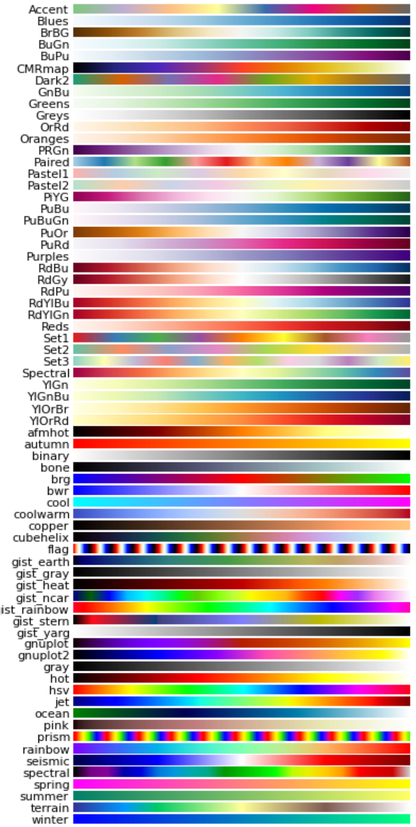
Better way: google "*color picker*" and you can select the HEX code of any colour you want



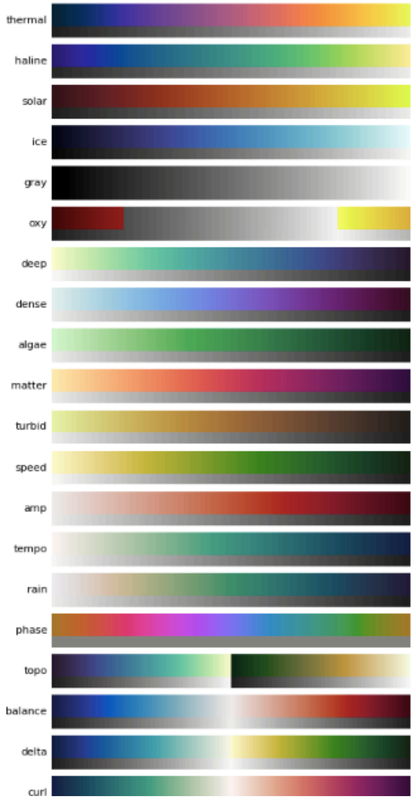
Colormaps

Many options available, don’t have to limit yourself to the ones provided by Matplotlib by default (although the default ones are default for a reason)

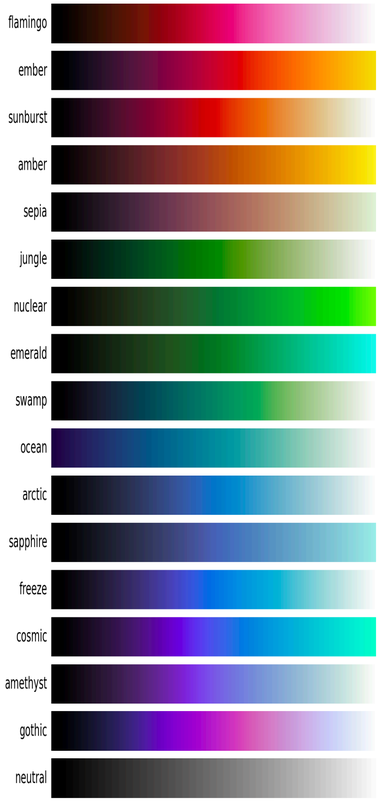
Matplotlib



cmocean



cmasher



Custom (e.g. Img2cmap)

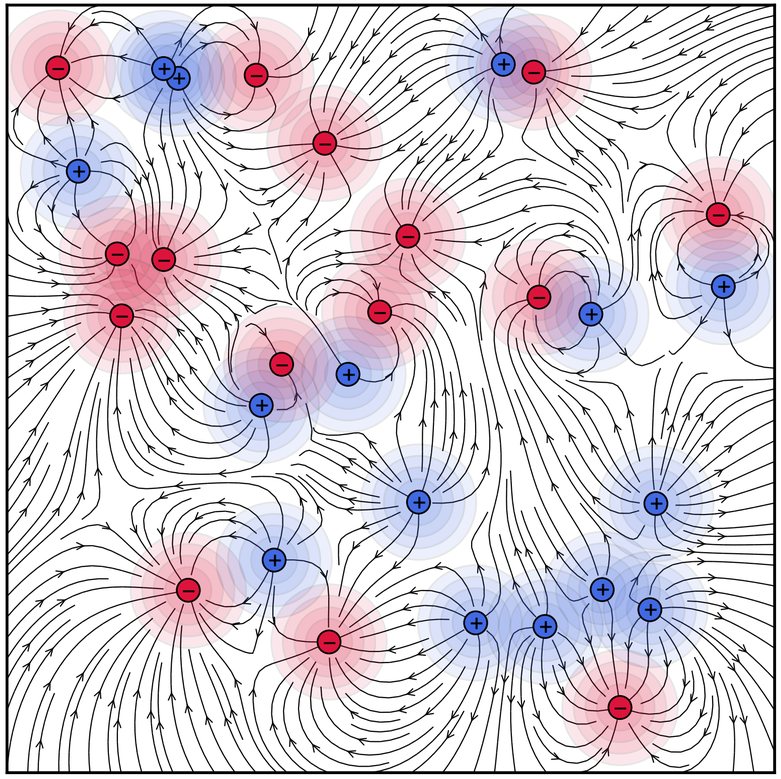
https://github.com/arvkevi/img2cmap

See how to make a colormap out of any list of colors: https://matplotlib.org/stable/gallery/color/custom_cmap.html

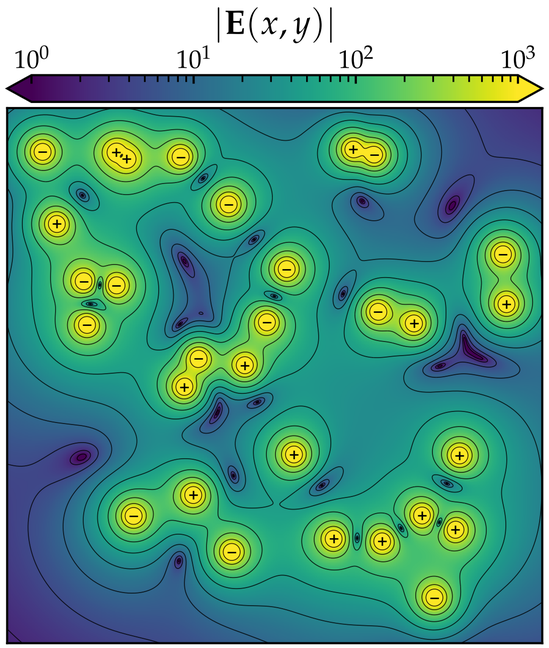
Colormaps

Tip: watch the file size!
If you use, `scatter`, `imshow`, `pcolormesh`, etc. to display large datasets the file size can get pretty huge. To fix that set "`rasterized=True`" inside the function, e.g. `plt.pcolormesh(… rasterized=True)`

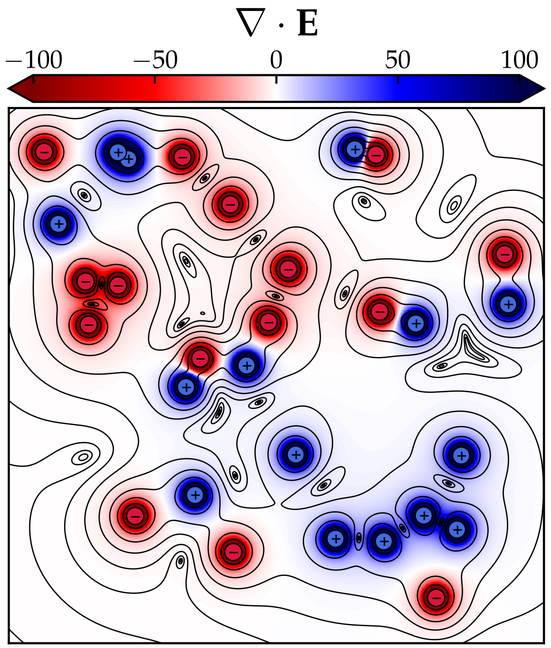
You cannot solely opt for aesthetics, you must choose the correct type for your data



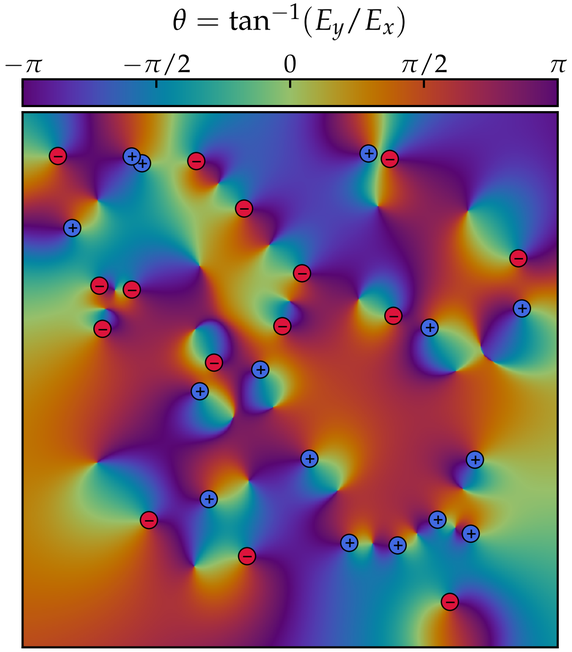
Charges.ipynb



Sequential (cm.viridis)



Diverging (cm.seismic)

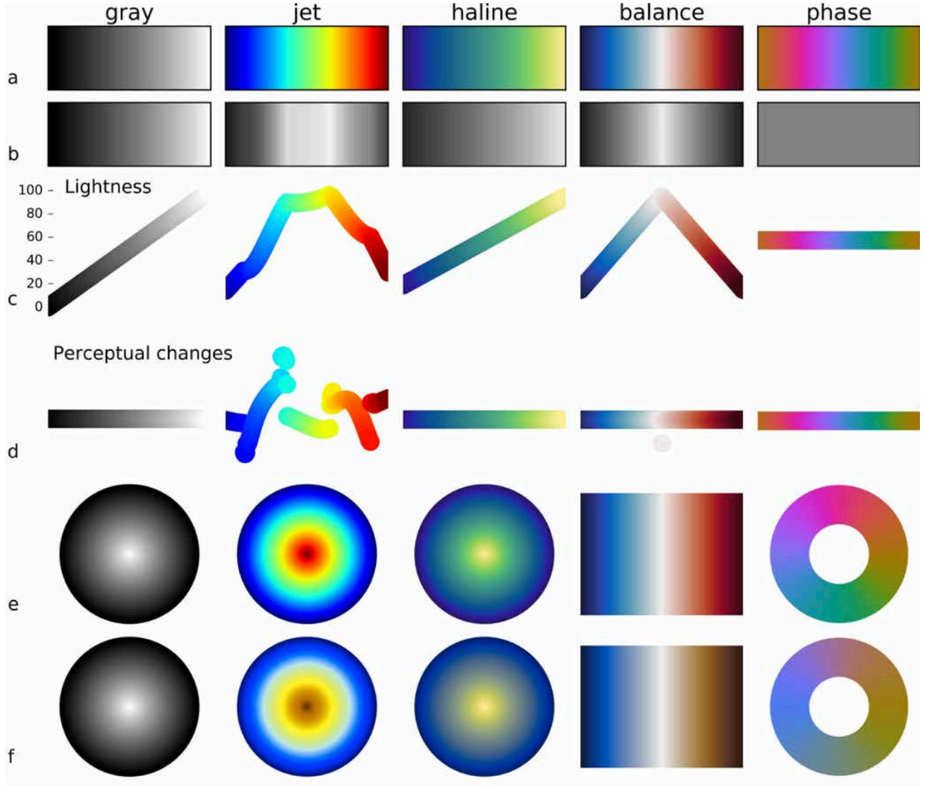


Cyclic (cmasher.infinity)

Visually uniform colormaps

https://colorcet.holoviz.org/

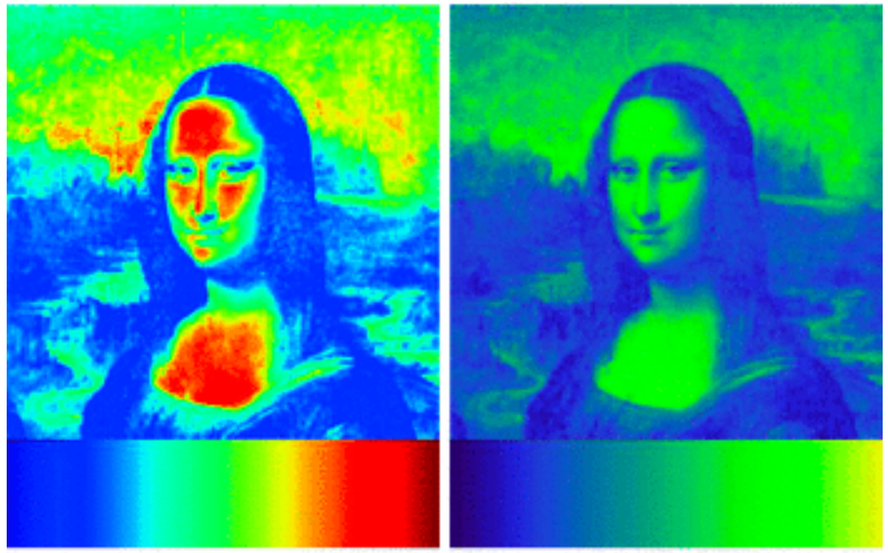
For sequential colormaps you want to have a uniform perceptual change as a function of distance through the colormap. Not all colormaps have this (the default *viridis* does)



https://tos.org/oceanography/article/true-colors-of-oceanography-guidelines-for-effective-and-accurate-colormap

Some colormaps induce artificial levels in the data due to perceptual discontinuities

→ **Bad! Don't use them!**



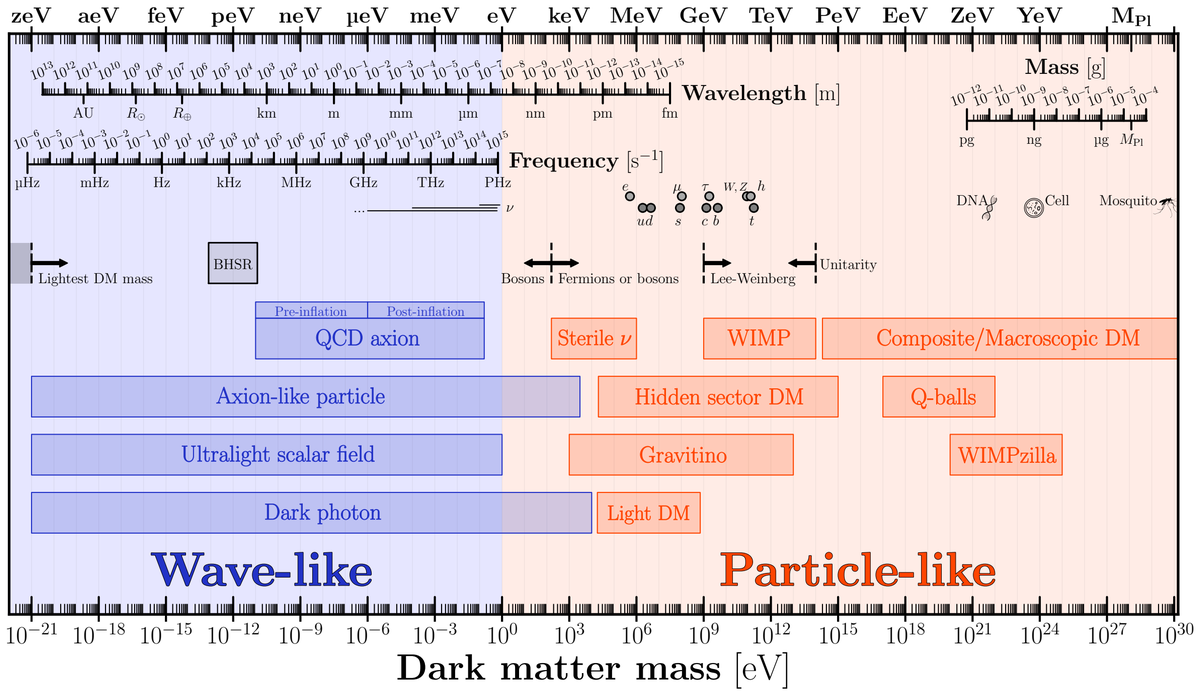
(a) Jet colormap. (b) Viridis colormap.

Text and fonts

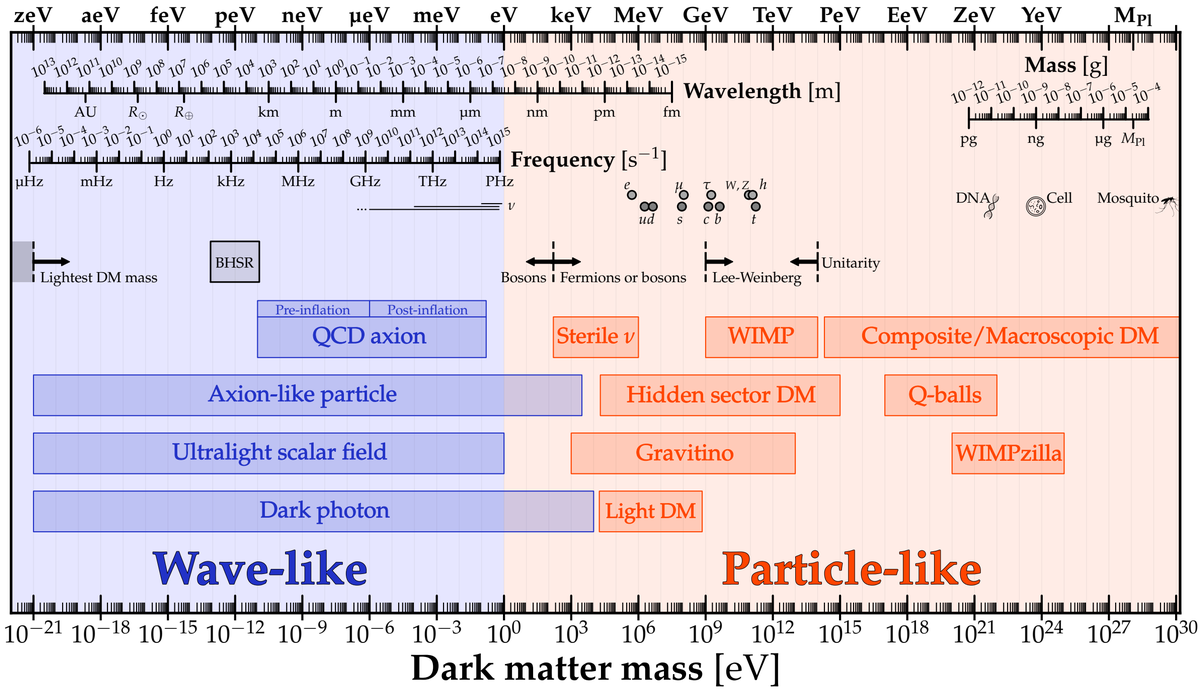
Tip: Placing text labels
→ `plt.text(x,y,'label')` will add a text label at the point (x,y) defined by whatever your axis coordinates are.
→`plt.gcf().text(x,y,'label')`will add a label to the figure itself where (x,y) are defined with respect to the bottom left corner, e.g. (0.5,0.5) is the middle of the figure.

For papers your plot will look best if all text (including labels, tick marks, etc.) is rendered as TeX. I am a big advocate for the "Palatino" font.

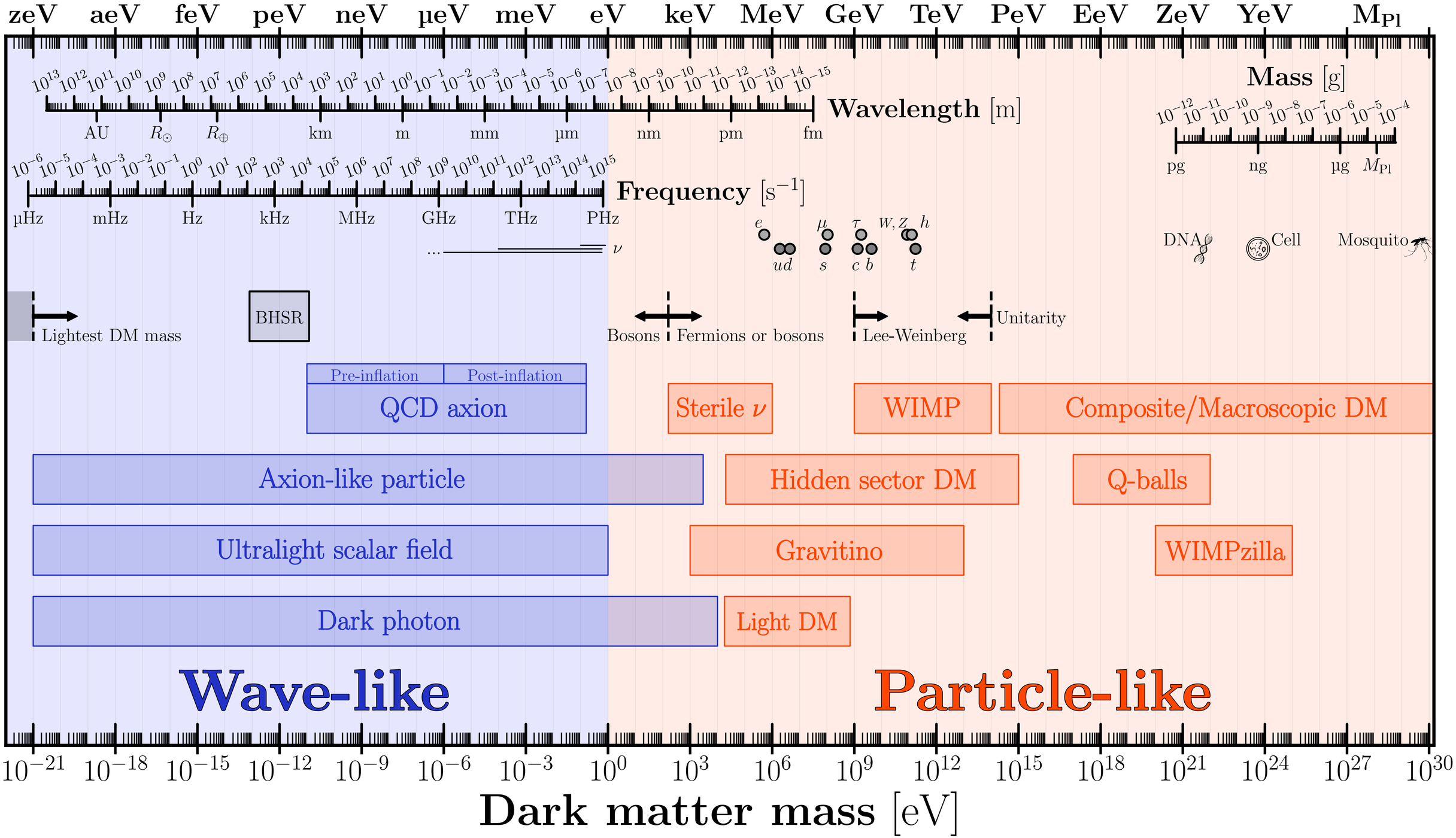
CM

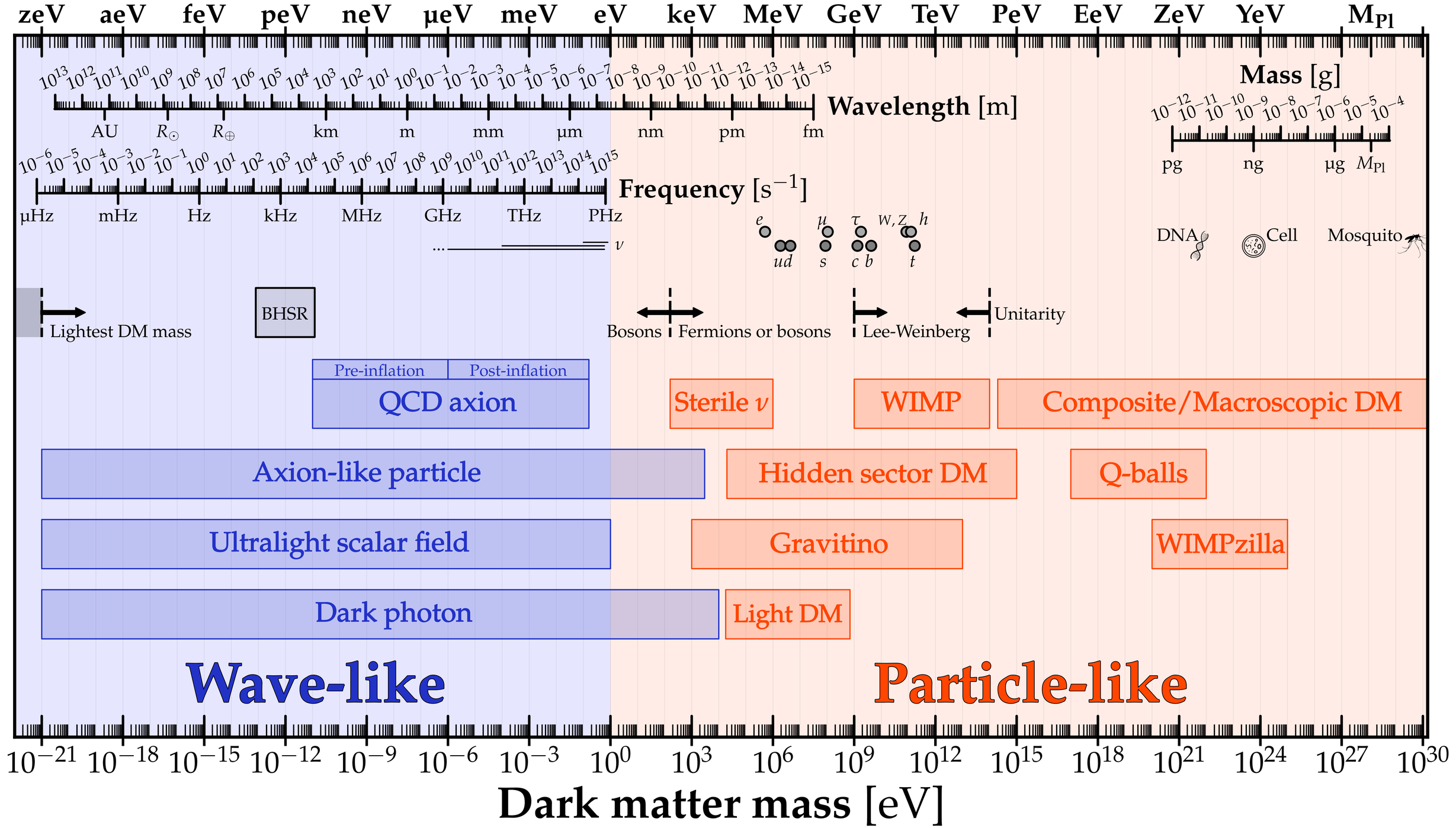


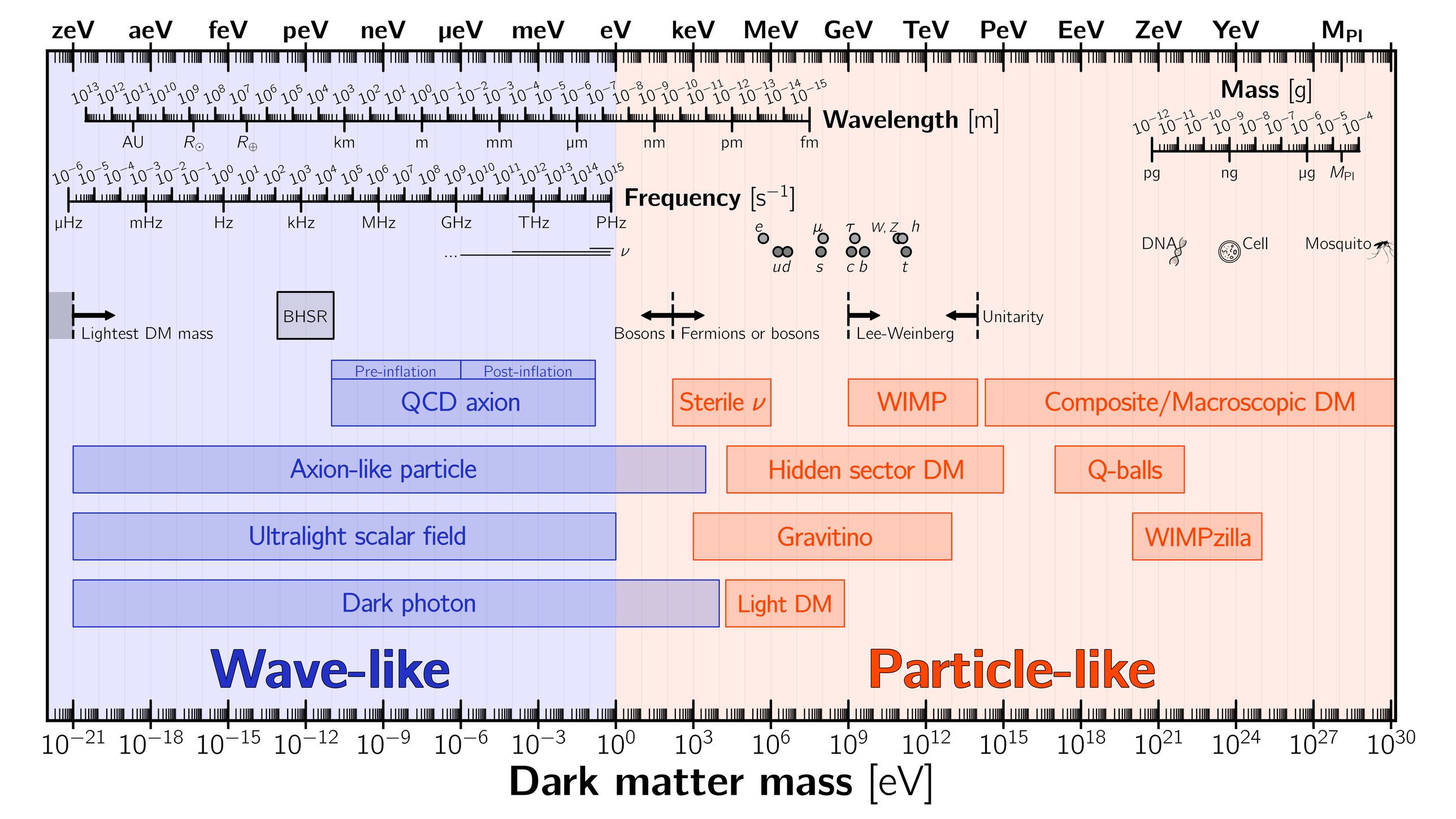
Palatino



DarkMatterLandscape.ipynb


zeV aeV feV peV neV μeV meV eV keV MeV GeV TeV PeV EeV ZeV YeV M_{Pl}
Wavelength [m]
AU $R_\odot$ $R_\oplus$ km m mm μm nm pm fm
Mass [g]
pg ng μg M_{Pl}
Frequency [s^{-1}]
μHz mHz Hz kHz MHz GHz THz PHz
ν
e ud μ s τ c b W, Z h t
DNA Cell Mosquito
Lightest DM mass
BHSR
Bosons
Fermions or bosons
Lee-Weinberg
Unitarity
Pre-inflation
Post-inflation
QCD axion
Sterile ν
WIMP
Composite/Macroscopic DM
Axion-like particle
Hidden sector DM
Q-balls
Ultralight scalar field
Gravitino
WIMPzilla
Dark photon
Light DM
Wave-like
Particle-like
10^{-21} 10^{-18} 10^{-15} 10^{-12} 10^{-9} 10^{-6} 10^{-3} 10^{0} 10^{3} 10^{6} 10^{9} 10^{12} 10^{15} 10^{18} 10^{21} 10^{24} 10^{27} 10^{30}
Dark matter mass [eV]


zeV aeV feV peV neV μeV meV eV keV MeV GeV TeV PeV EeV ZeV YeV M_{Pl}
Wavelength [m]
AU $R_\odot$ $R_\oplus$ km m mm μm nm pm fm
Mass [g]
pg ng μg M_{Pl}
Frequency [s^{-1}]
μHz mHz Hz kHz MHz GHz THz PHz
e μ τ W, Z h
ud s c b t
ν
DNA Cell Mosquito
Lightest DM mass
BHSR
Bosons
Fermions or bosons
Lee-Weinberg
Unitarity
Pre-inflation
Post-inflation
QCD axion
Sterile ν
WIMP
Composite/Macroscopic DM
Axion-like particle
Hidden sector DM
Q-balls
Ultralight scalar field
Gravitino
WIMPzilla
Dark photon
Light DM
Wave-like
Particle-like
10^{-21} 10^{-18} 10^{-15} 10^{-12} 10^{-9} 10^{-6} 10^{-3} 10^{0} 10^{3} 10^{6} 10^{9} 10^{12} 10^{15} 10^{18} 10^{21} 10^{24} 10^{27} 10^{30}
Dark matter mass [eV]

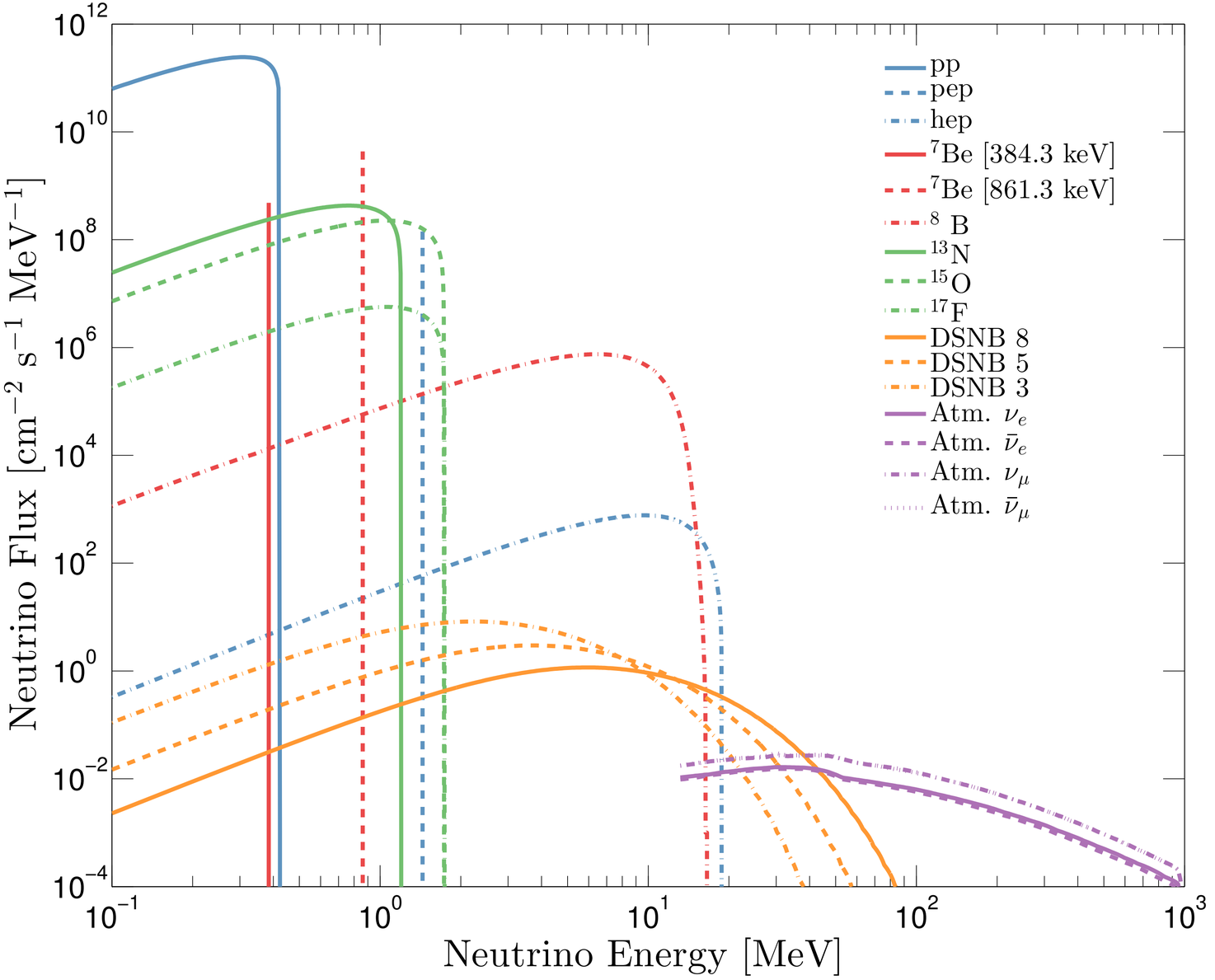

zeV aeV feV peV neV μeV meV eV keV MeV GeV TeV PeV EeV ZeV YeV M_{Pl}
Wavelength [m]
AU $R_\odot$ $R_\oplus$ km m mm μm nm pm fm
Mass [g]
pg ng μg M_{Pl}
Frequency [s^{-1}]
μHz mHz Hz kHz MHz GHz THz PHz
e μ τ W, Z h
ud s c b t
ν
DNA Cell Mosquito
Lightest DM mass
BHSR
Bosons
Fermions or bosons
Lee-Weinberg
Unitarity
Pre-inflation
Post-inflation
QCD axion
Sterile ν
WIMP
Composite/Macroscopic DM
Axion-like particle
Hidden sector DM
Q-balls
Ultralight scalar field
Gravitino
WIMPzilla
Dark photon
Light DM
Wave-like
Particle-like
10^{-21} 10^{-18} 10^{-15} 10^{-12} 10^{-9} 10^{-6} 10^{-3} 10^{0} 10^{3} 10^{6} 10^{9} 10^{12} 10^{15} 10^{18} 10^{21} 10^{24} 10^{27} 10^{30}
Dark matter mass [eV]

Examples in the wild

Examples of bad practice: my own plot

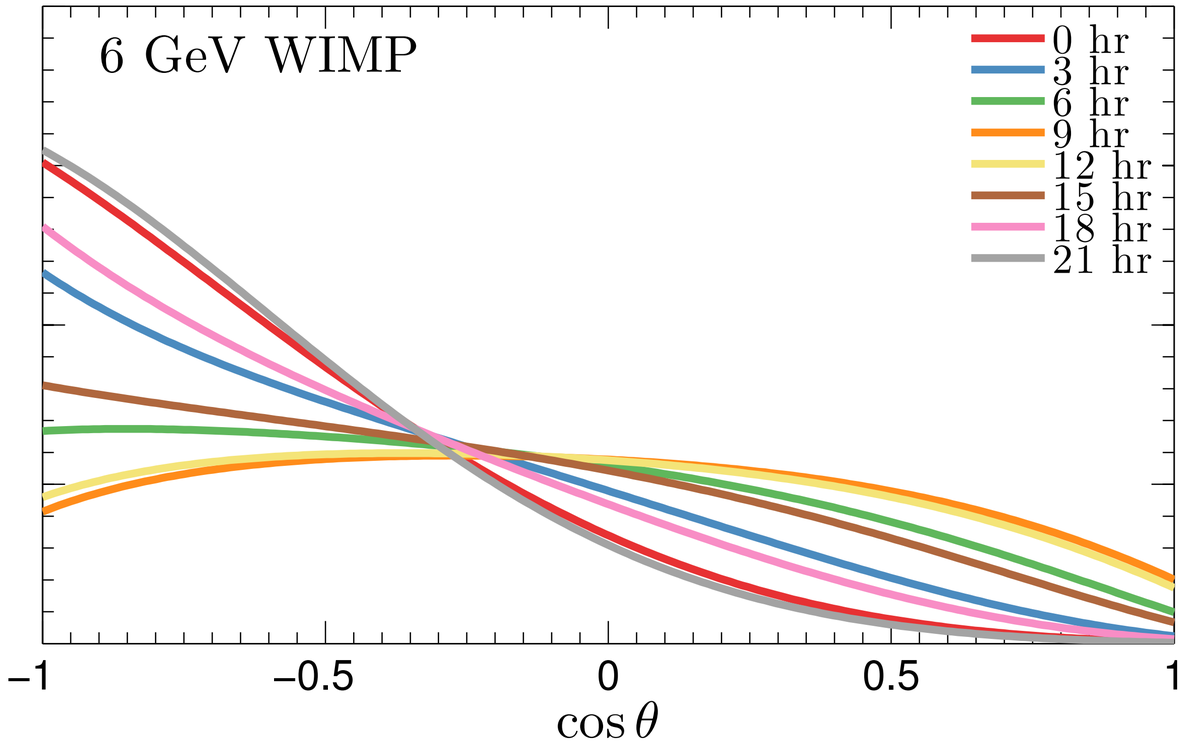
In what ways is this plot bad?

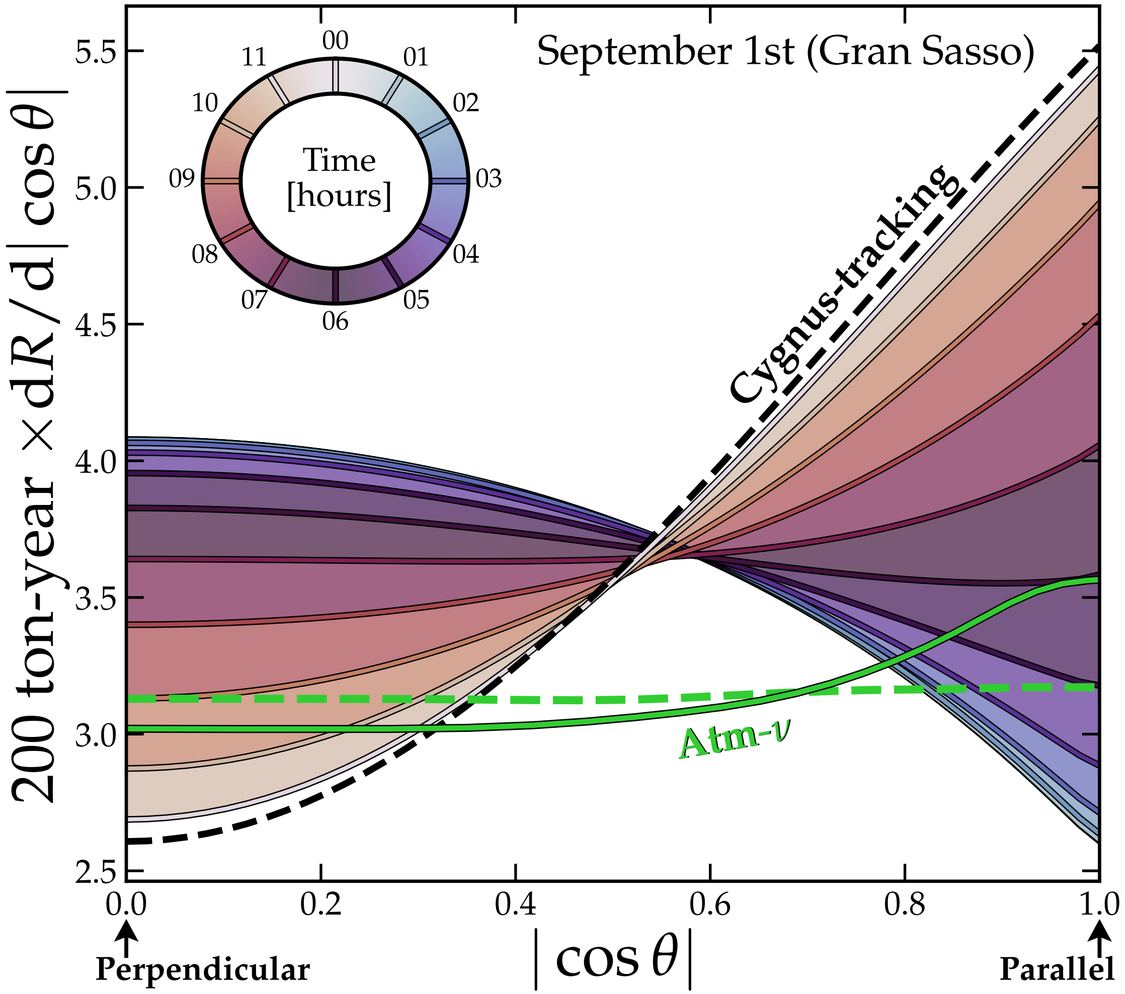
Plot from my 2nd paper (2015)



Examples of bad practice: my own plots

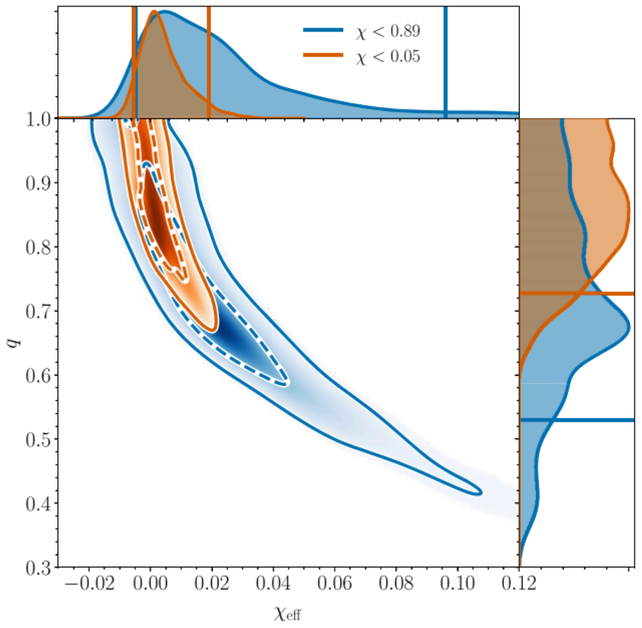
In what ways is this plot bad?

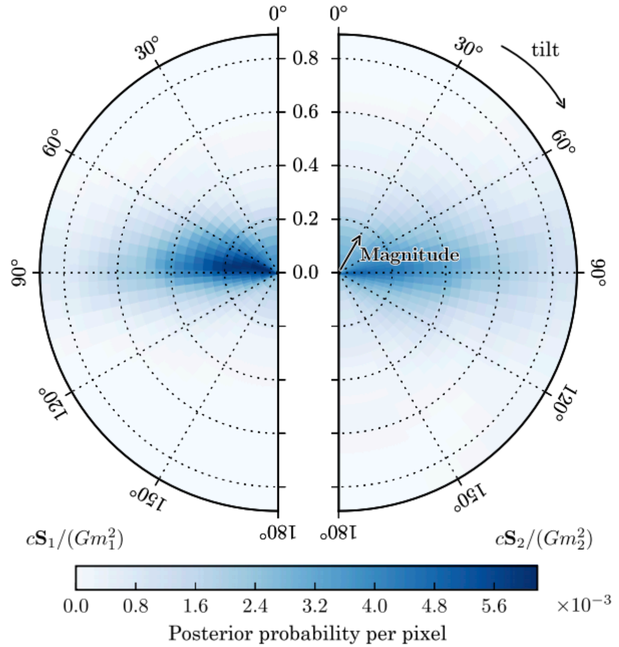


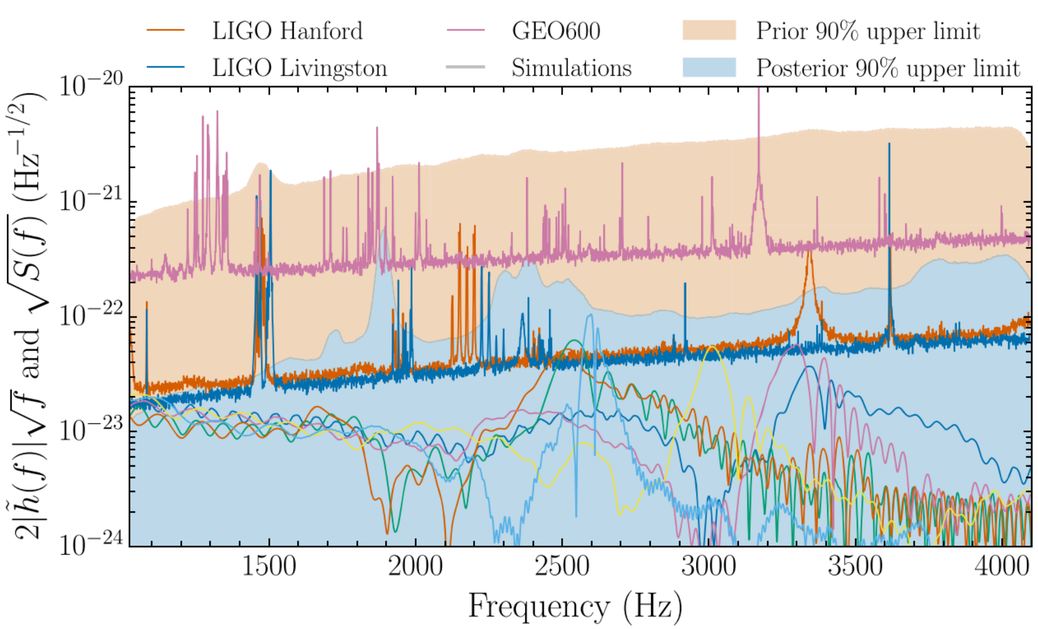


Example of good practice: LIGO

- Always have attractive figures, look in any of their papers, e.g. arxiv.org/abs/1805.11579
- High attention to detail
- Clear and uncluttered, even for complicated plots
- Well-labelled and can be understood at a glance
- Often opt for complementary colours of orange/blue used consistently and meaningfully, i.e. blue for Livingston and orange for Hanford



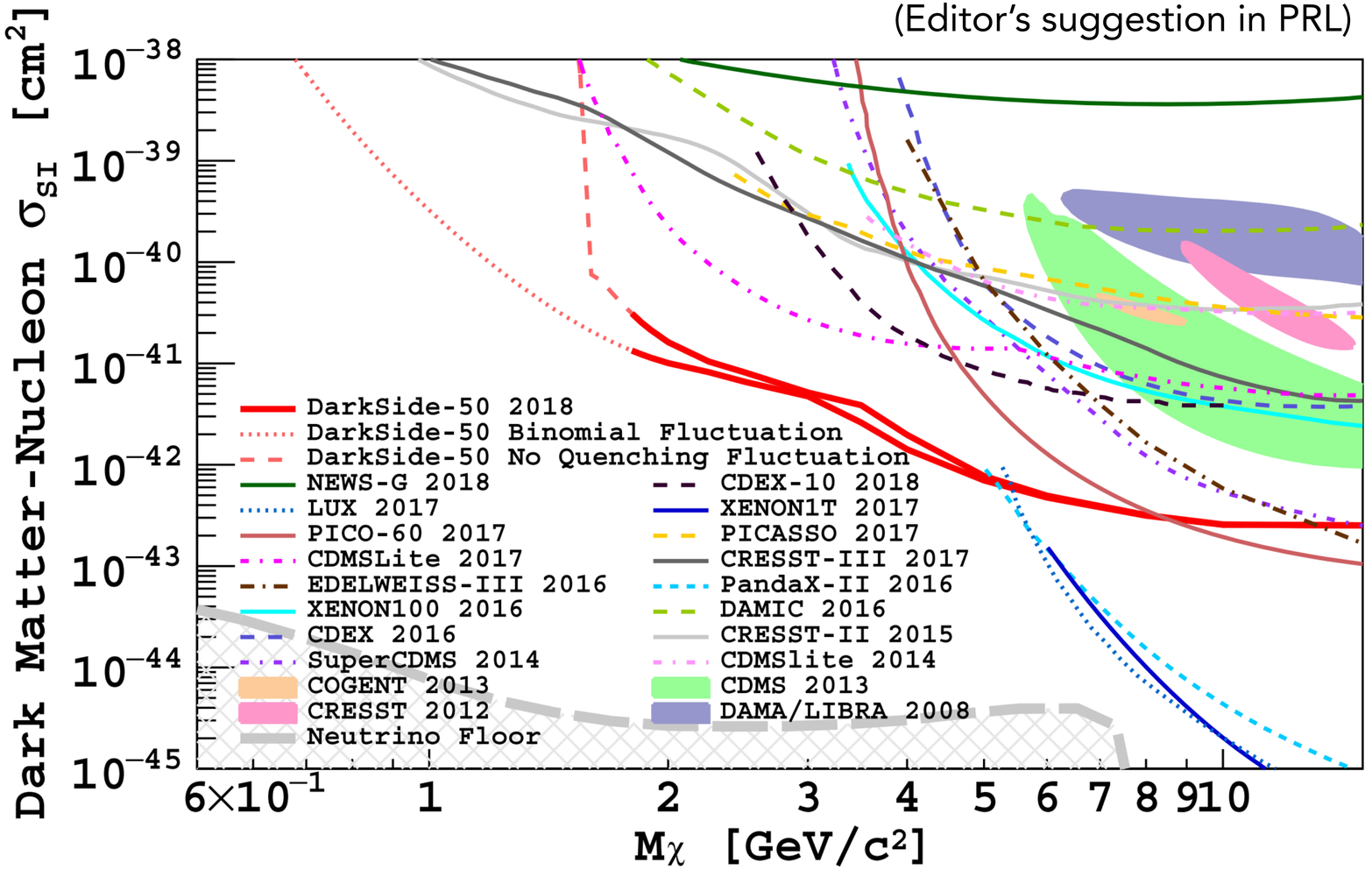




Example of bad practice: DarkSide

- Just no
- Worst part is that I think this is an intentional stylistic choice

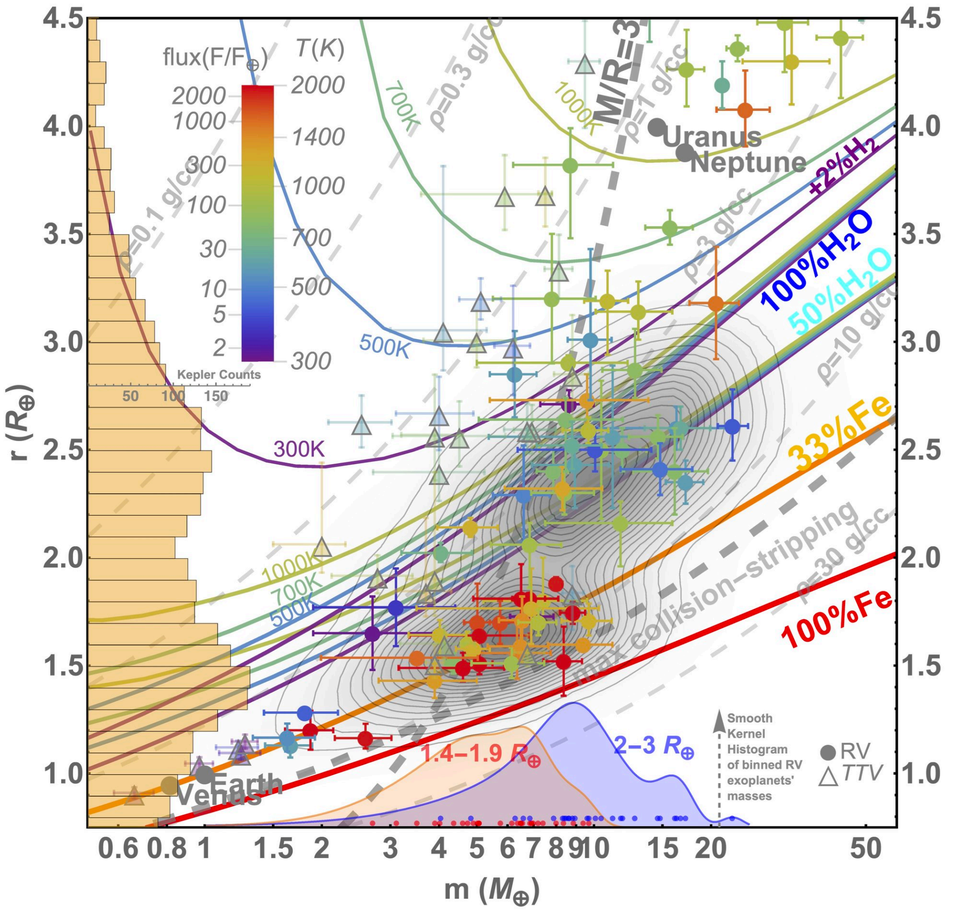
(Editor's suggestion in PRL)



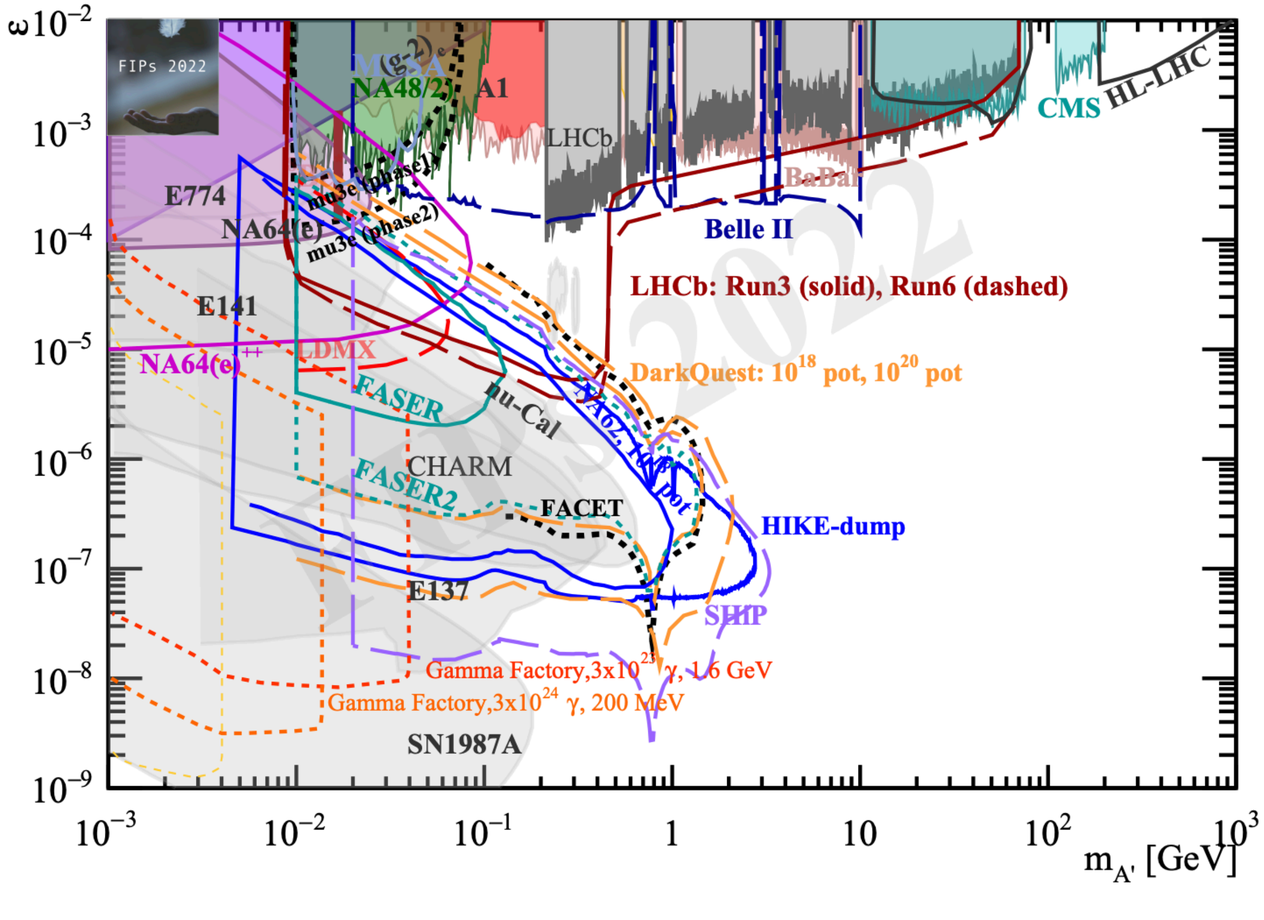
How to break the rules

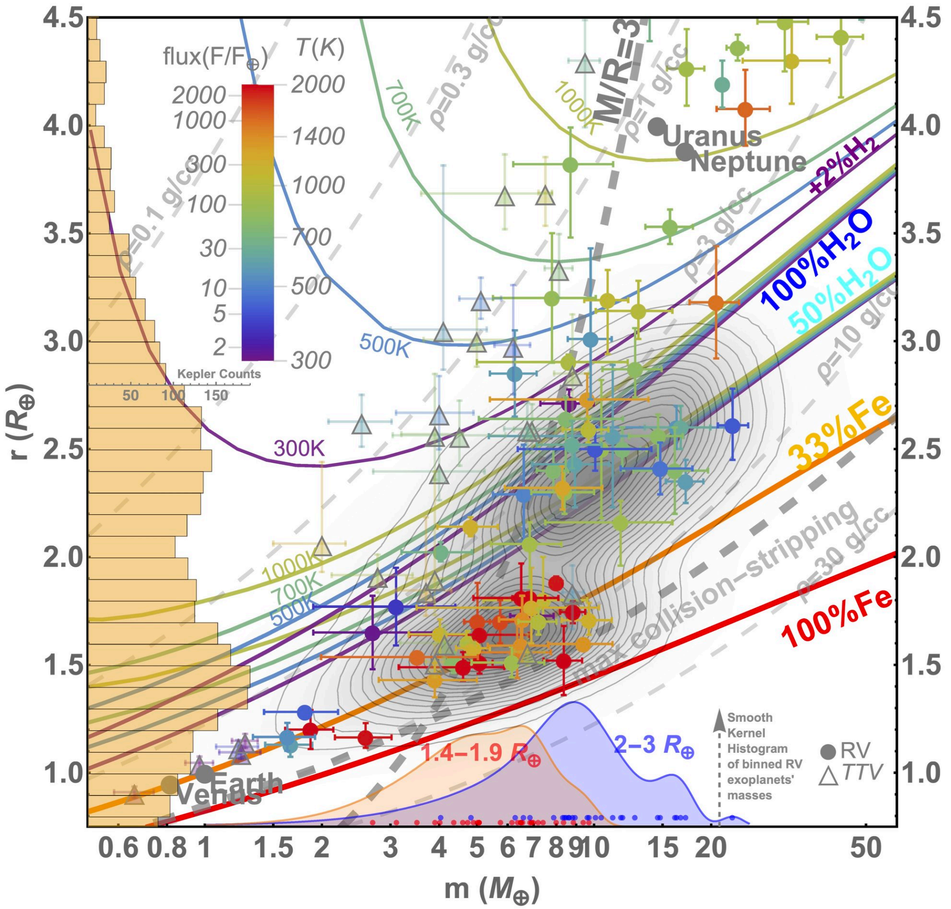
Ultimately style is subjective, all that matters is your audience. You can always break the rules if you know why they are there and you break them intentionally.

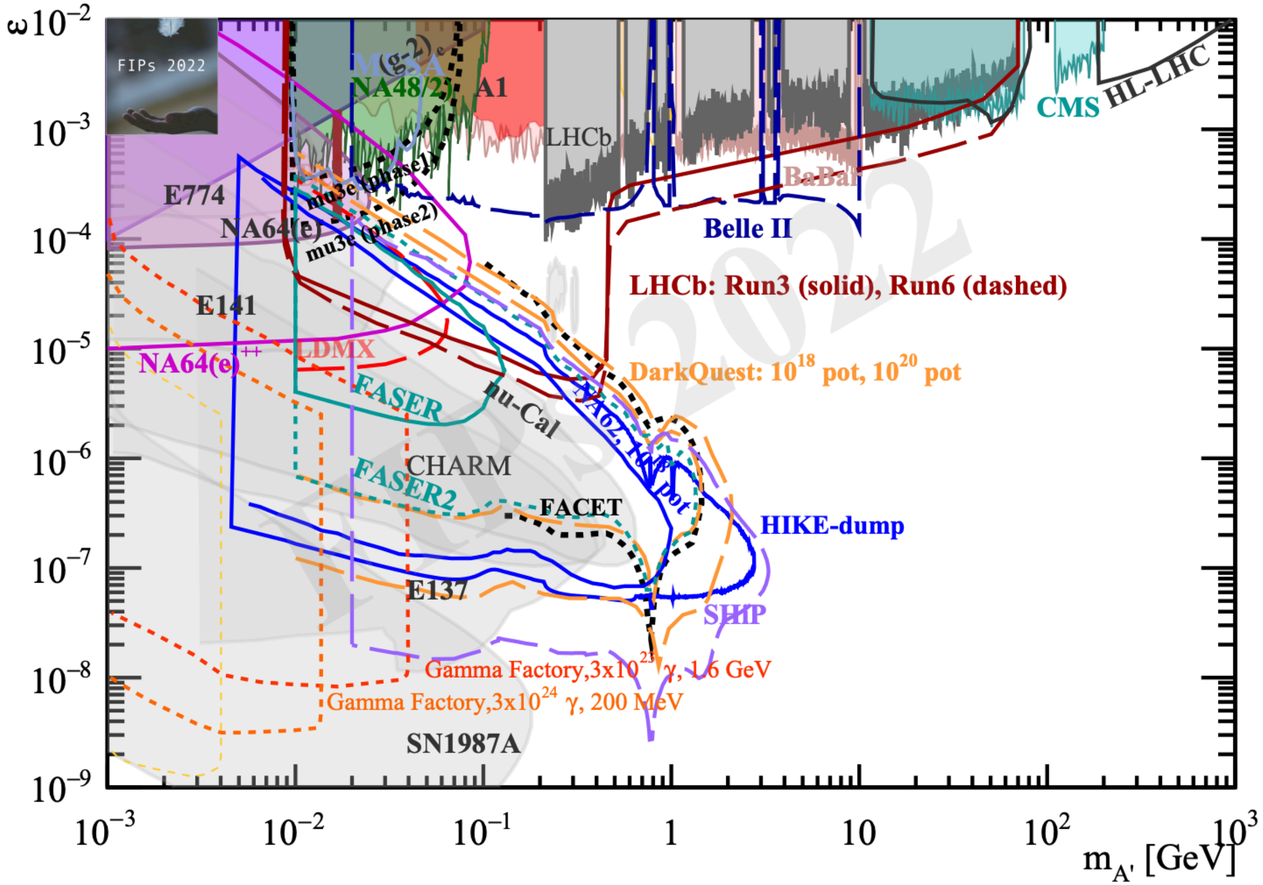
Why is this plot good?



…and this plot is bad?





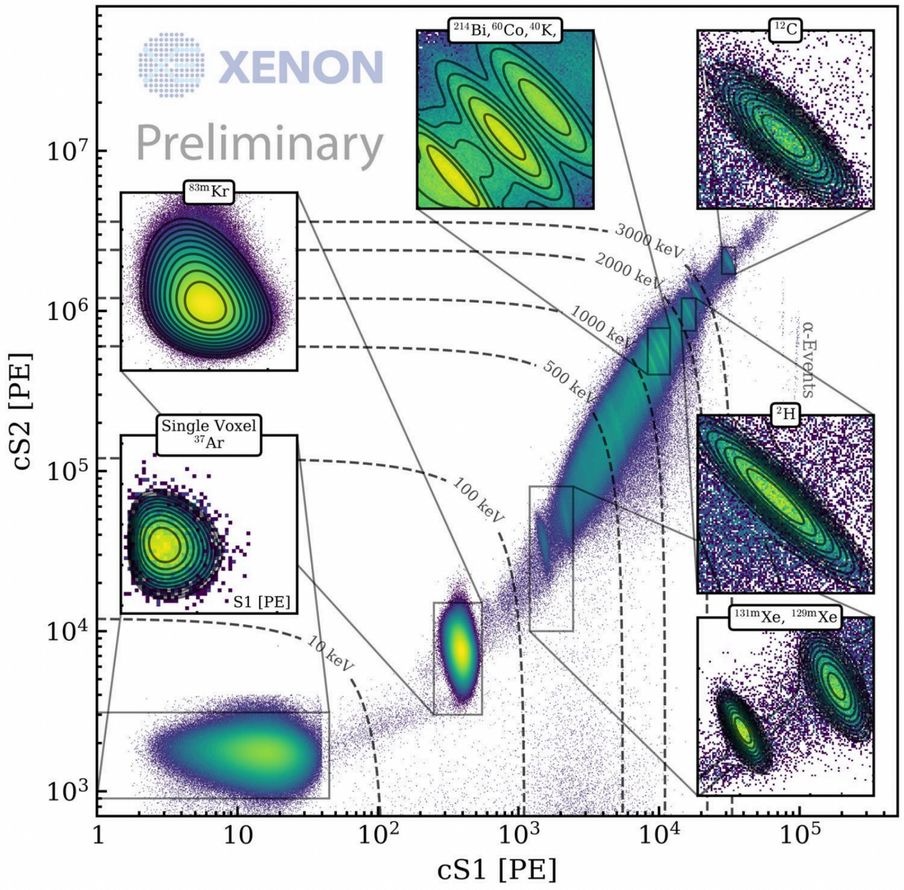


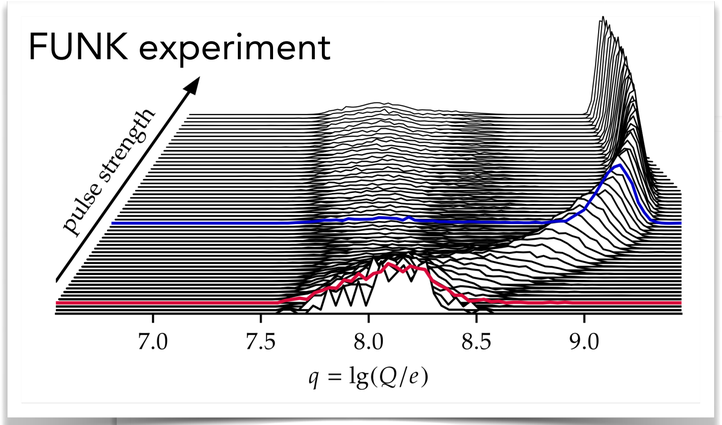
This is a highly detailed plot that expresses a huge amount of information (it's basically seven plots stacked on top of each other). It works because it is based on a type of plot that is extremely familiar to a specific audience (exoplanets)

This is a comparatively simple plot (just the parameter space of a dark photon) that is almost unintelligible due to poor design and the inclusion of unnecessary elements.

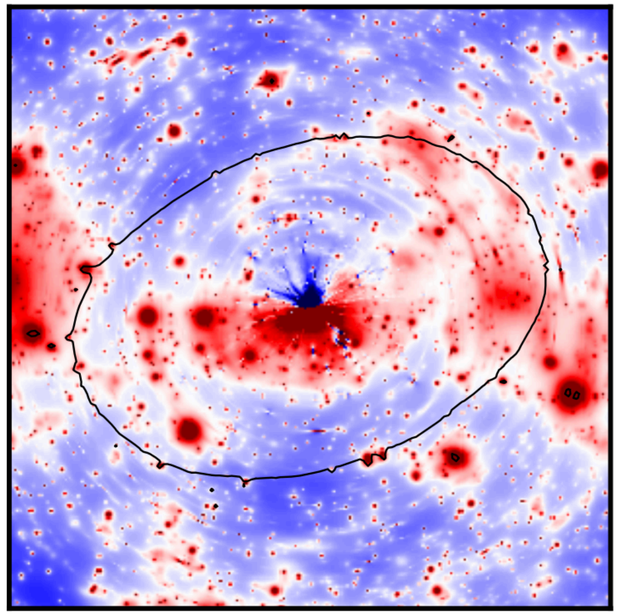
More plots that I like...

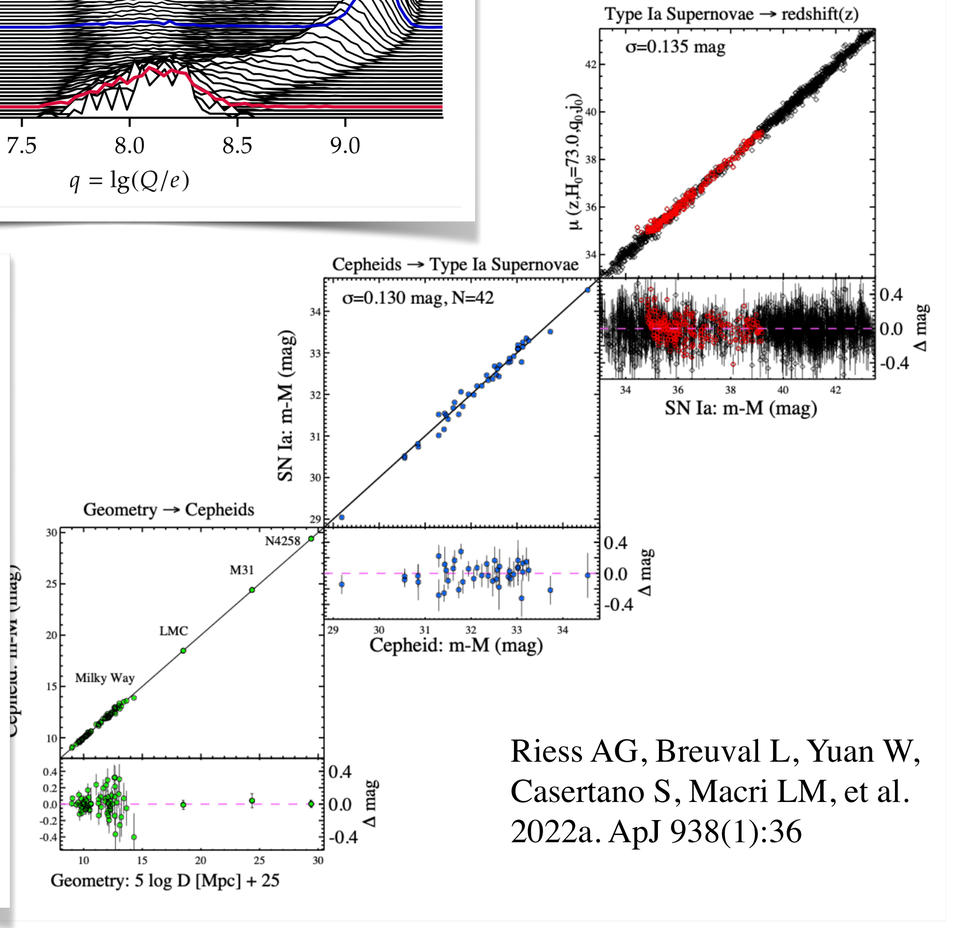
Proof that if someone likes your plot they are more likely to use it in their talk





Laroche [2206.11269]

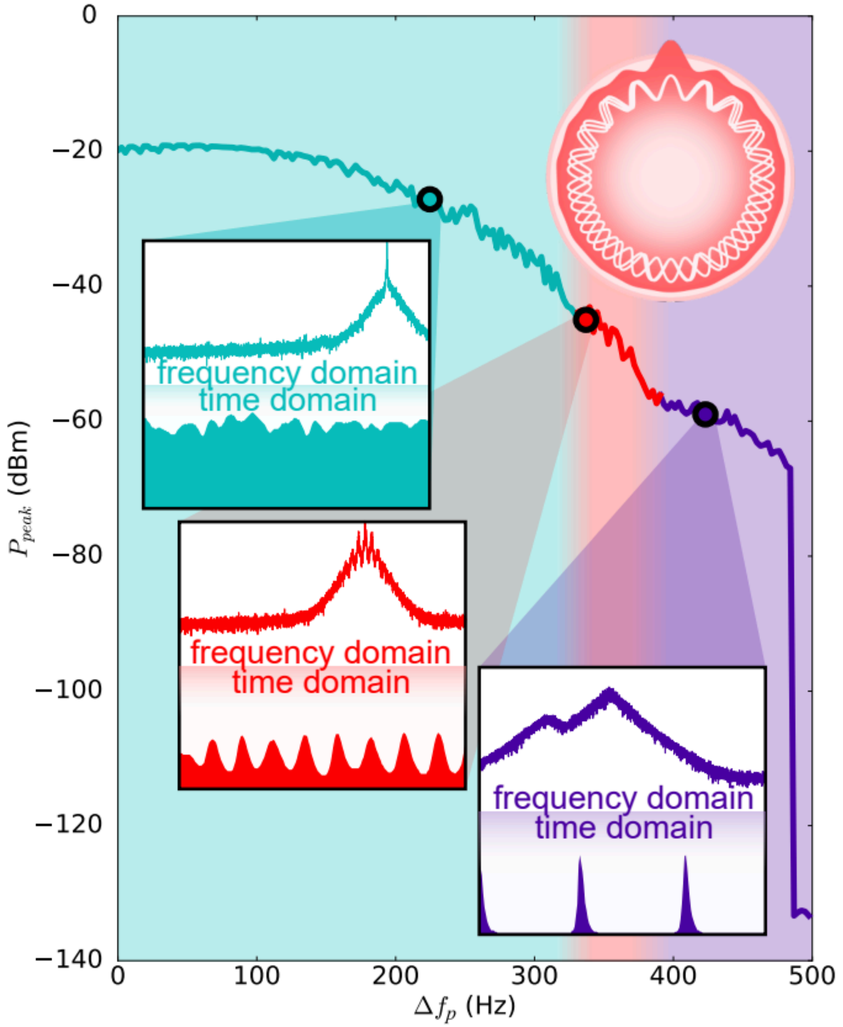




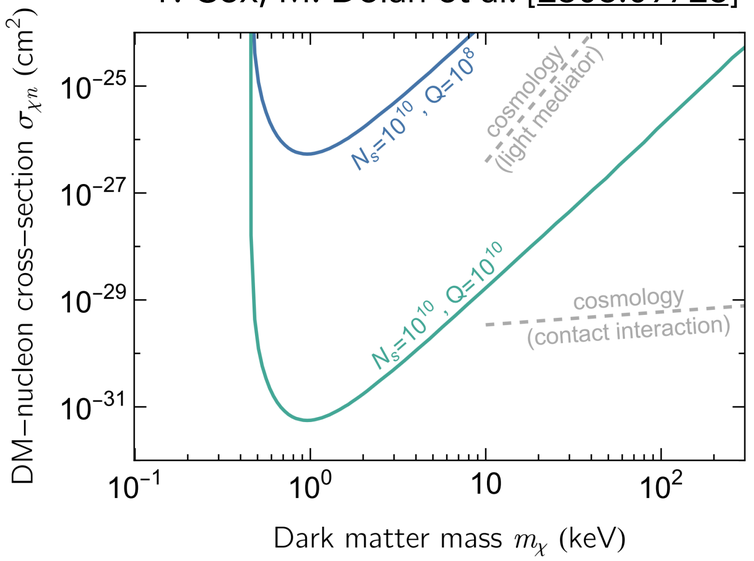
Riess AG, Breuval L, Yuan W, Casertano S, Macri LM, et al. 2022a. ApJ 938(1):36

Examples of good practice: you!

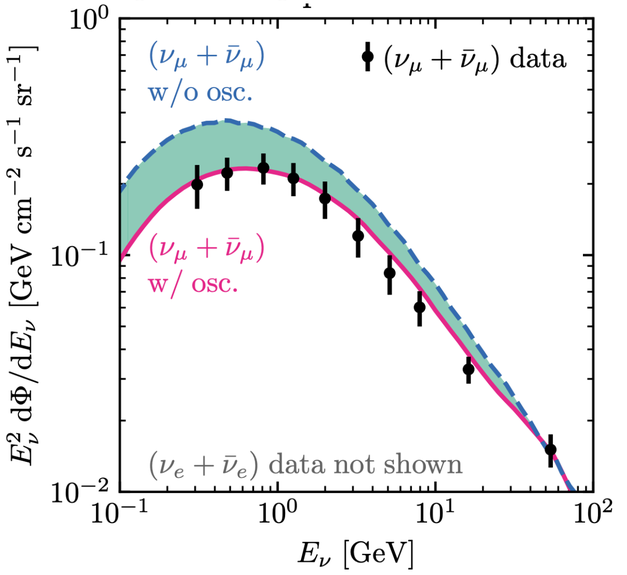
C. A. Thomson et al [2105.11180]

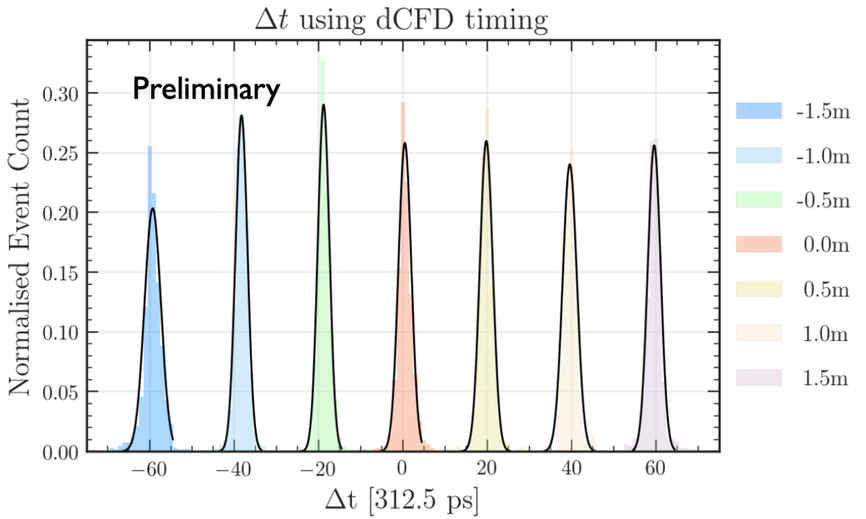


P. Cox, M. Dolan et al. [2306.09726]



S. Meighan-Berger et al. [2311.01667]



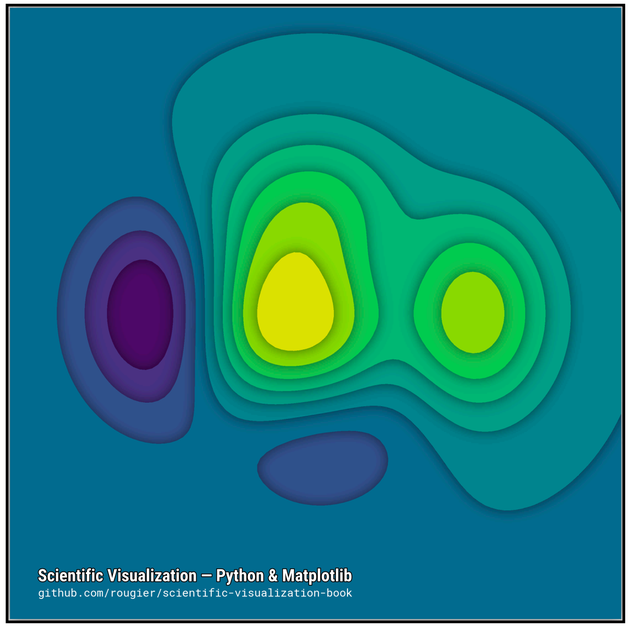


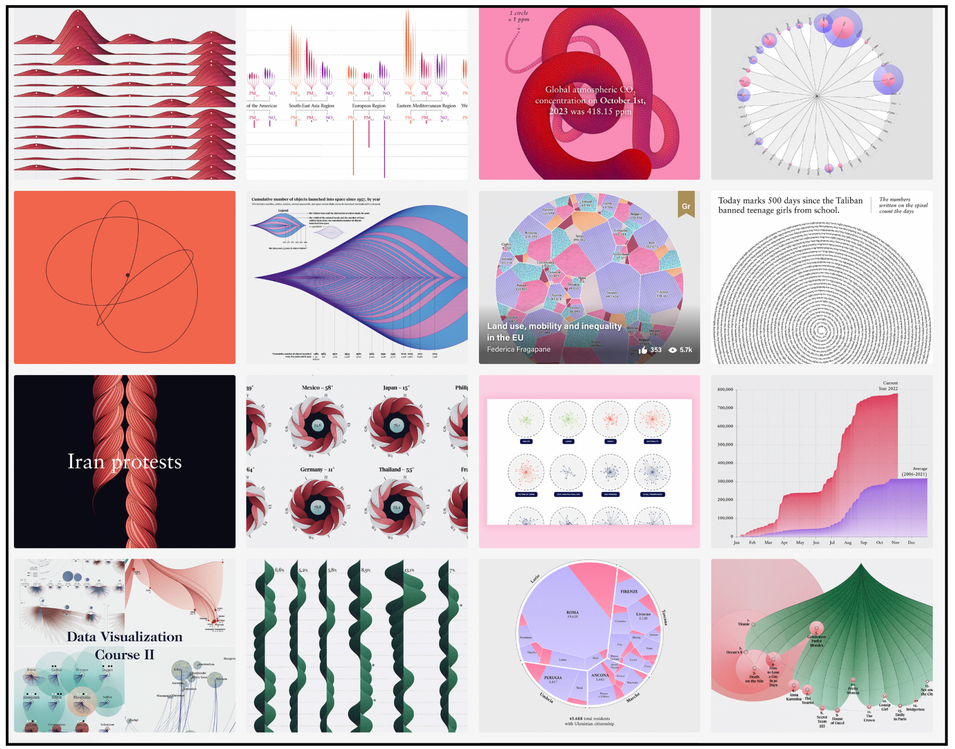
Summary/takeaway messages

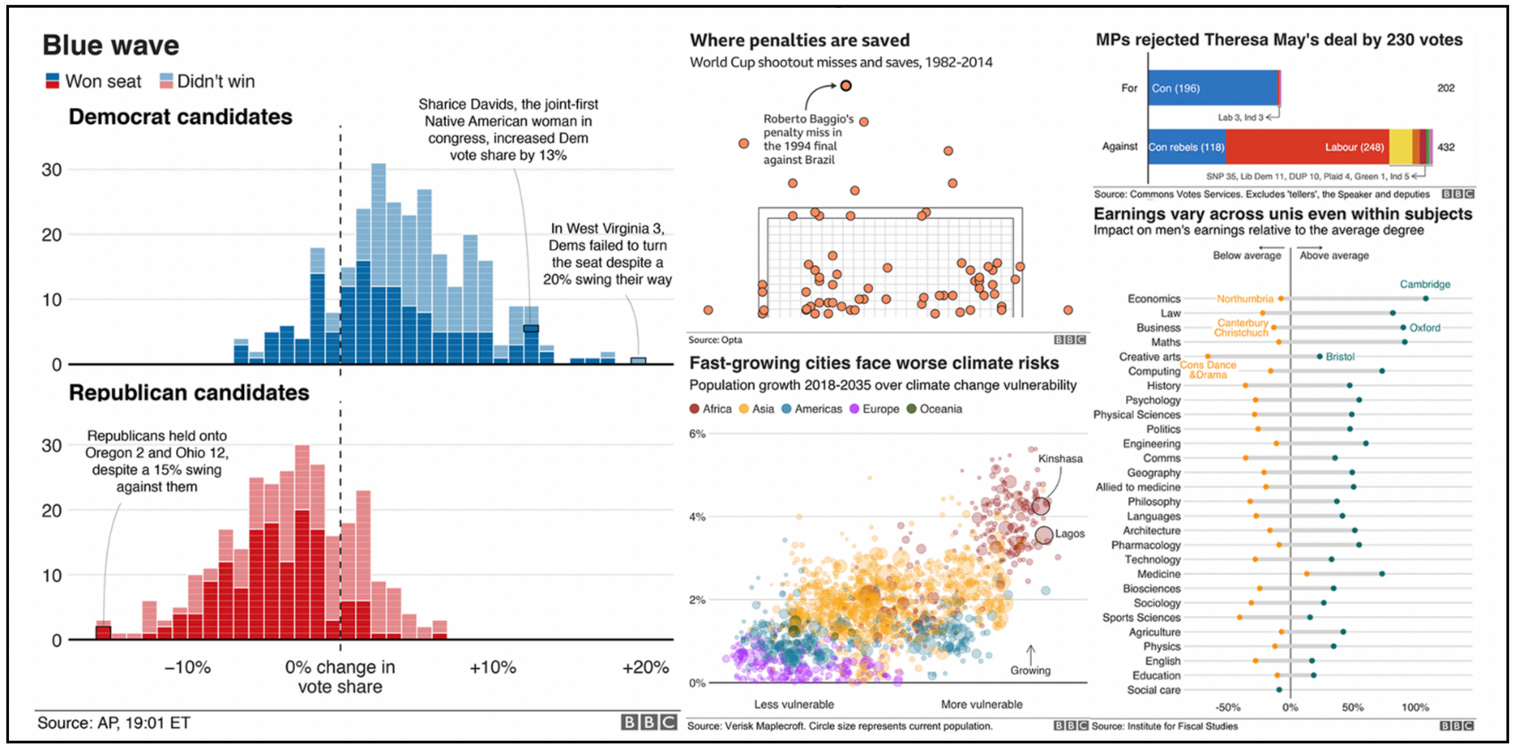
1. Just put effort in

Inspiration from the professionals

Look at what data visualisation specialists do. Their audiences and fields will be completely different, but the underlying principles carry over: you want to capture attention and convey a complex, quantitative message clearly and efficiently.







Matplotlib cheat sheets

https://matplotlib.org/cheatsheets/