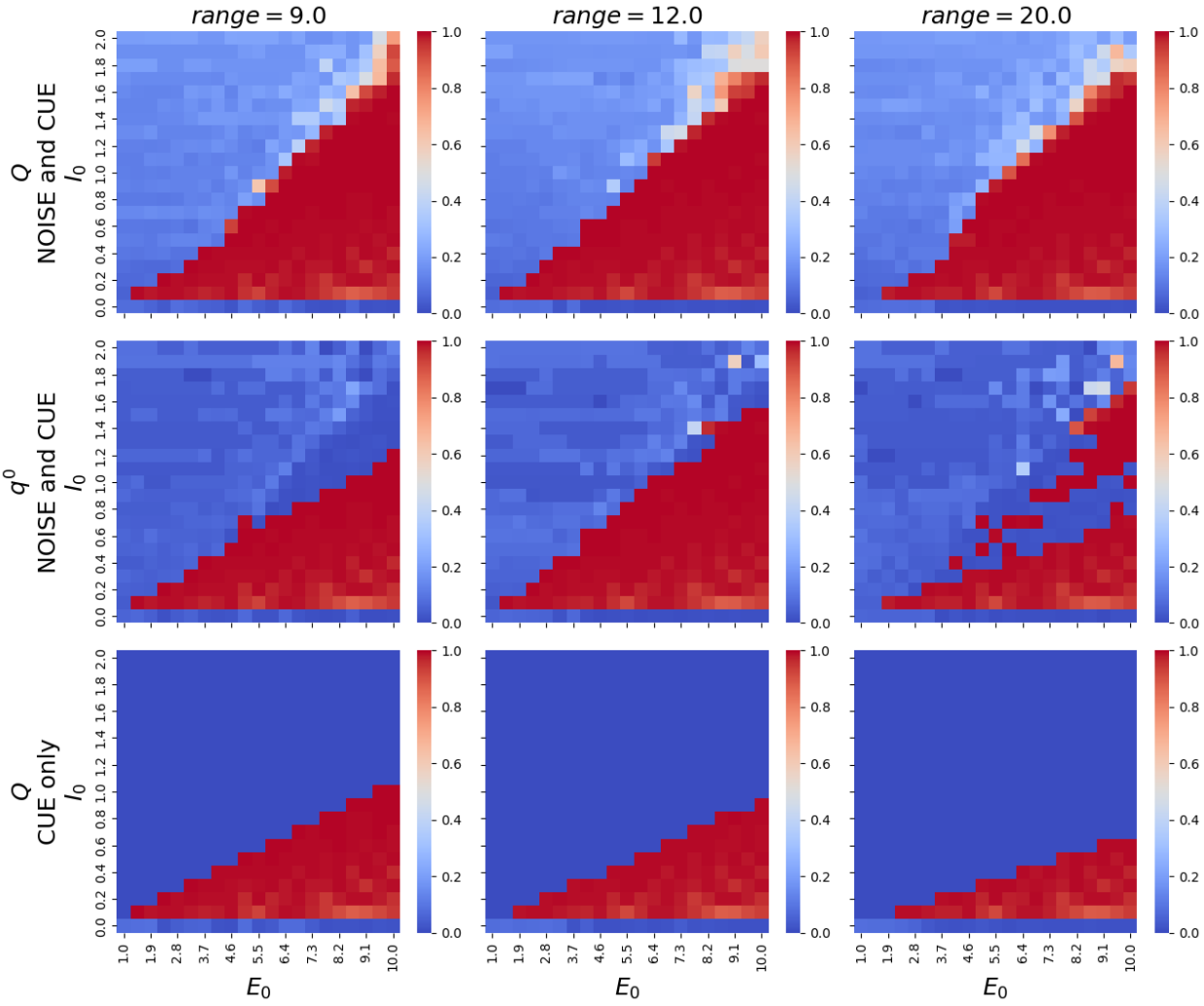
You can use multiple simlation objects to make other graphs and plots

This example will not work with the data in "example_simulation", is just an example of what I did in the thesis

```
import sys
from pathlib import Path

# Get the parent directory
parent_dir = Path().resolve().parent
# print(parent_dir)

# Add the parent directory to the system path
sys.path.append(str(parent_dir))
# print(sys.path)

%reload_ext autoreload
%autoreload 2
# from simulation_class_24_08_07 import Simulation, test
from simulation_class_24_11_19 import Simulation, test
from make_fig_spikes_24_11_19 import make_fig_spikes2

files_dir = r'/home/apicella/Output_Files'
simulation_name_cue = r'random-21x21-gauss-CUE_only-t500--Z-300--
seed2'
simulation_name_noise = r'random-21x21-gauss-NOISE_only-t500--Z-300--
seed2'
simulation_name_noise_cue = r'random-21x21-gauss-NOISE_and_CUE-t500--
Z-300--seed2'

save_dir = r'/home/apicella/Dropbox/tesi/code/ThesisPlots/Images/'

cue_only = Simulation(files_dir=files_dir,
simulation_name=simulation_name_cue)
noise_only = Simulation(files_dir=files_dir,
simulation_name=simulation_name_noise)
noise_and_cue = Simulation(files_dir=files_dir,
simulation_name=simulation_name_noise_cue)

dir = random-21x21-gauss-CUE_only-t500--Z-300--seed2-generic-lines-
heatmap-data: not counted
dir = images_thesis: not counted
dir = random-21x21-gauss-NOISE_only-t500--Z-300--seed2-generic-lines-
heatmap-data: not counted
dir = .ipynb_checkpoints: not counted
dir = random-21x21-gauss-NOISE_and_CUE-t500--Z-300--seed2-generic-
```

```
lines-heatmap-data_OLD: not counted
dir = random-21x21-gauss-NOISE_and_CUE-t500--Z-300--seed2-generic-
lines-heatmap-data: not counted

import matplotlib.pyplot as plt
fig, ax = plt.subplots(3, 3, sharex=True, sharey=True,
figsize=(13,10.8))
ranges = [ 9.0, 12.0, 20.0]
sims = [cue_only, noise_only, noise_and_cue]
titles = ["$Q$\nNOISE and CUE", "$q^0$\nNOISE and CUE", "$Q$\nCUE
only"]

for j, mixing_range in enumerate(ranges):
    noise_and_cue.plot_overlap(n_patterns=20,
mixing_range=mixing_range, ax=ax[0][j])
    noise_and_cue.plot_overlap_pattern_0(n_patterns=20,
mixing_range=mixing_range, ax=ax[1][j])
    cue_only.plot_overlap(n_patterns=20, mixing_range=mixing_range,
ax=ax[2][j])
    for i in range(3):
        ax[i][j].set_ylabel("", fontsize=18)
        ax[i][j].set_xlabel("", fontsize=18)
        ax[i][j].set_title(f"", fontsize=18)
    ax[0][j].set_title(f"$range={mixing_range}$", fontsize=18)
    ax[2][j].set_xlabel("$E_0$", fontsize=18)

for i, title in enumerate(titles):
    ax[i][0].set_ylabel(title + "\n$I_0$", fontsize=18)

# fig.suptitle("Overlap $Q$", fontsize=18)
fig.tight_layout()

import os
format = 'png'
dpi = 600
filename = rf'overlap_ranges_9_12_20_compare recall zone' + f'.
{format}'
saveto = os.path.join(save_dir, filename)

fig.savefig(saveto, format=format, dpi=dpi)
print(f"saved to {saveto}")

fig.show()

saved to
/home/apicella/Dropbox/tesi/code/ThesisPlots/Images/overlap_ranges_9_1
2_20_compare recall zone.png
```


range = 9.0
range = 12.0
range = 20.0
Q
NOISE and CUE
I_0
q^0
NOISE and CUE
I_0
Q
CUE only
I_0
E_0
E_0
E_0