

Feedback Plots

Collin Bradford

Generates plots of k_{eff} with respect to a temperature change and a moderator density change to show that temperature change within the fuel is the dominate effect.

```
In [42]: ▶ # imports
from matplotlib import pyplot as plt
import numpy as np
```

```

In [44]: # data from previous simulation runs:
doppler_temps = [-200.0, -155.5555555555554, -111.1111111111111, -66.6666666666667, -22.2222222222222, 22.2222222222222, 66.6666666666667, 111.1111111111111, 155.5555555555554, 200.0]
doppler_crit = [1.0990704014192465, 1.093082489813471, 1.0866016805732654, 1.0801180514267346, 1.0736344222811889, 1.067150793134715, 1.0606671639882346, 1.0541835348417654, 1.0476999056952811, 1.0412162765488065]
mod_void_crit = [1.070577280132804, 1.0717554400997706, 1.0721221342964269, 1.0724888284000931, 1.0728555225033593, 1.0732222166066255, 1.0735889107098917, 1.0739556048131579, 1.0743222989164241, 1.0746889930196903]

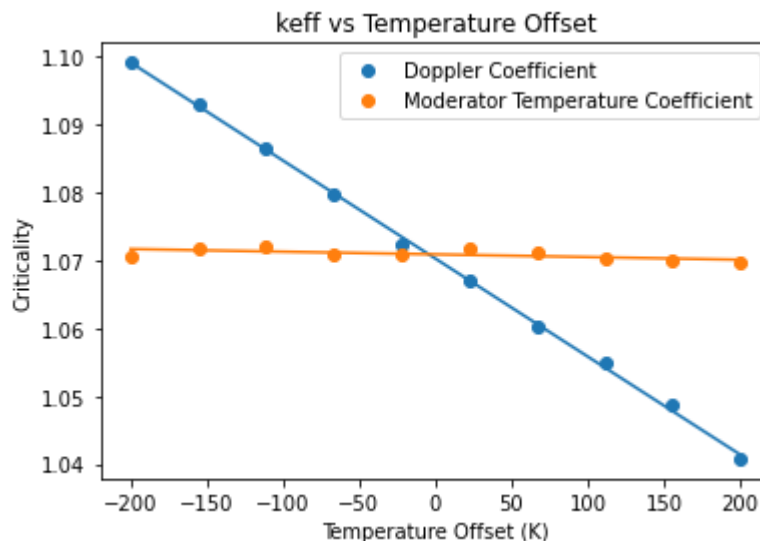
# plot doppler feedback coefficient.
# the doppler feedback coefficient, in this case, comes largely due to the
plt.plot(doppler_temps, doppler_crit, 'o', label="Doppler Coefficient")
m, b = np.polyfit(doppler_temps, doppler_crit, 1)
plt.plot(doppler_temps, np.polyval((m, b), doppler_temps), c='tab:blue')

# plot the moderator temp coefficient
# contrary to a typical PWR today, most of the feedback comes from the fuel
plt.plot(doppler_temps, mod_void_crit, 'o', label="Moderator Temperature Coefficient")
m, b = np.polyfit(doppler_temps, mod_void_crit, 1)
plt.plot(doppler_temps, np.polyval((m, b), doppler_temps), c='tab:orange')

plt.title("keff vs Temperature Offset")
plt.xlabel("Temperature Offset (K)")
plt.ylabel("Criticality")
plt.legend()

```

Out[44]: <matplotlib.legend.Legend at 0x7febd80cb280>



In []: