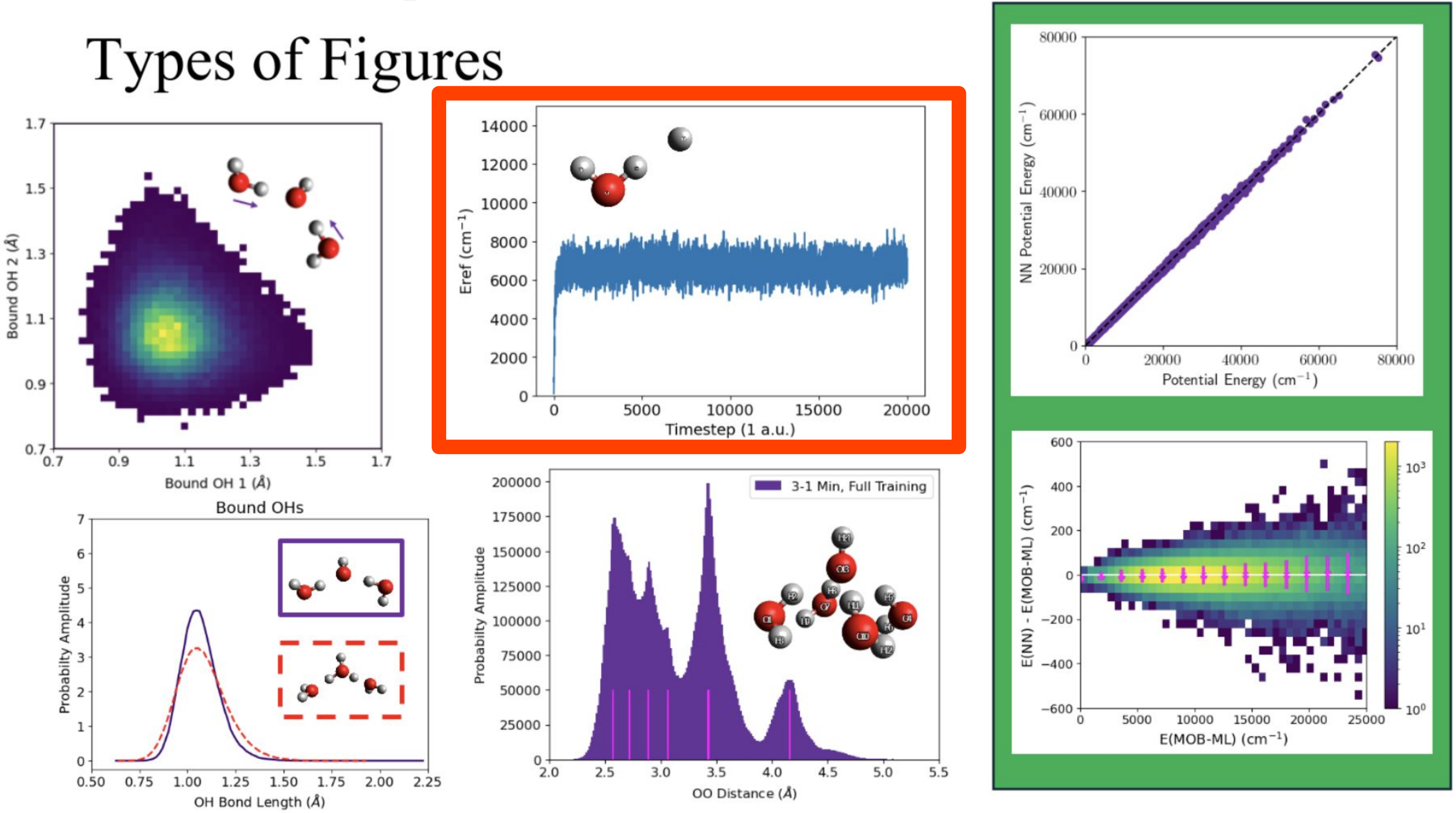
CSE 583 Technology Review:

PyVisDMC visualization options

Goal: Easy analysis of data from diffusion Monte Carlo simulations.

Want user to be able to input data from a PyVibDMC simulation and quickly generate common plot types.

want: nice plots!



Types of Figures

Vega-Altair

- https://github.com/gretaja/pyvisdmc/blob/main/technology_review/example_altair_vis.ipynb

Pros

- Interactive plots 🤠
  - Nice for quick checks of fine details
- “Concise grammar” 😎
  - Tell altair *what* data to analyze, it handles *how* to create the plot
  - Automatic plot formatting, scaling

Cons

- Data size limitations 😬
  - Default size limit of 5,000 rows
  - Needs configuration of renderers, data transformers…
- Slow 🐌 (especially for data sets as big as ours)
- Limited plot customization 💔
  - Can’t necessarily fine tune individual plot elements

Matplotlib

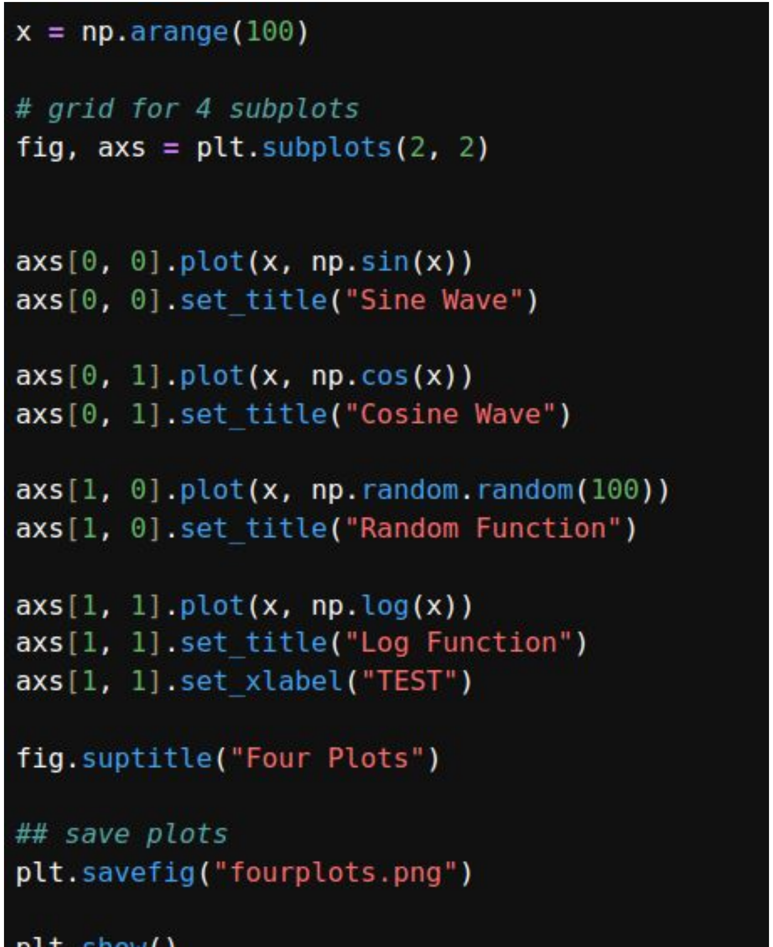
- https://github.com/gretaja/pyvisdmc/blob/main/examples/pyvibdmc_plotting_examples.ipynb
- Matplotlib is a widely-used, comprehensive library for visualization in Python: https://matplotlib.org/stable/

Pros:

- **Easy to integrate with other software and other Python libraries.**
- **Extensive options for customizing visualizations.**
- **Large number of online resources makes it an accessible technology.**

Cons:

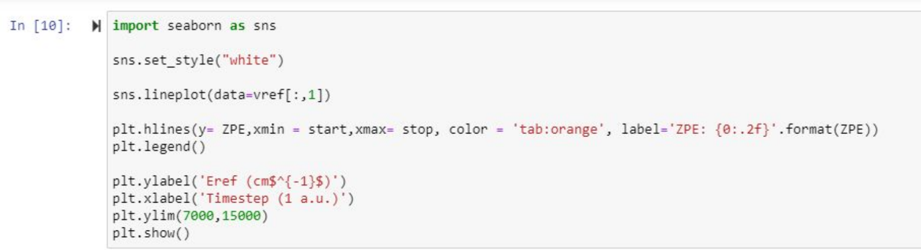
- **Advanced customization options may require more complex syntax compared.**
- **While it offers some interactivity, it lacks**



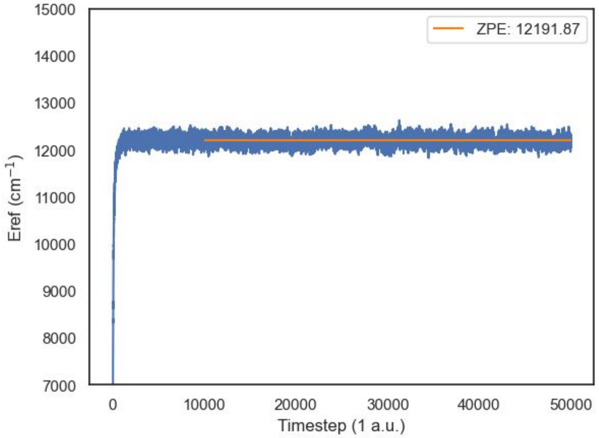
```
x = np.arange(100)

# grid for 4 subplots
fig, axs = plt.subplots(2, 2)


axs[0, 0].plot(x, np.sin(x))
axs[0, 0].set_title("Sine Wave")

axs[0, 1].plot(x, np.cos(x))
axs[0, 1].set_title("Cosine Wave")

axs[1, 0].plot(x, np.random.random(100))
axs[1, 0].set_title("Random Function")

axs[1, 1].plot(x, np.log(x))
axs[1, 1].set_title("Log Function")
axs[1, 1].set_xlabel("TEST")

fig.suptitle("Four Plots")

## save plots
plt.savefig("fourplots.png")

plt.show()
```

seaborn

- Higher level interface for data exploration
  - further customized using matplotlib’s functions
- Integrates closely with pandas data structures (not how our data is currently stored)
- Multivariate views (and statistical analyses) of complex datasets
- Last modified 4 months ago
- 200 contributors, 500k users



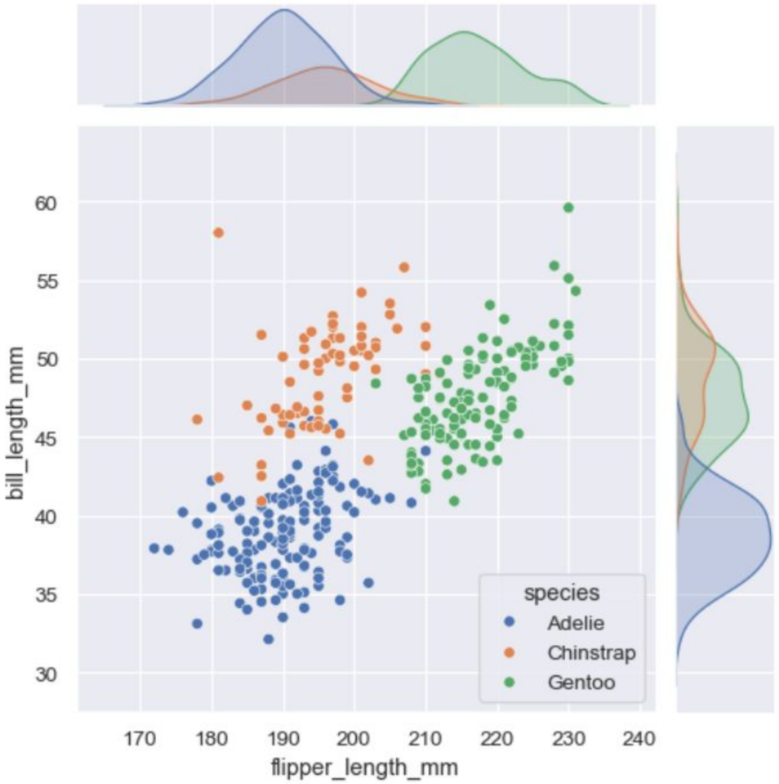
```
In [10]:   import seaborn as sns

           sns.set_style("white")

           sns.lineplot(data=vref[:,1])

           plt.hlines(y= ZPE,xmin = start,xmax= stop, color = 'tab:orange', label='ZPE: {0:.2f}'.format(ZPE))
           plt.legend()

           plt.ylabel('Eref (cm$^{-1}$)')
           plt.xlabel('Timestep (1 a.u.)')
           plt.ylim(7000,15000)
           plt.show()
```



Conclusion

seaborn

- Try using seaborn for default visualizations
  - Data exploration (multi-dimensions)
  - Initial insights about simulation data
  - Automatic statistical analyses
  - Easiest for users new to coding
- Utilize matplotlib functions to further specify figure details
  - Publication/presentation quality visualizations
  - Experienced researchers can still create figures to their exact specifications


bill_length_mm
flipper_length_mm
species
● Adelie
● Chinstrap
● Gentoo