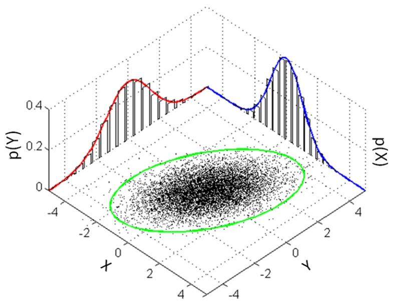
General instructions

- Use Python language and JAX library for all the tasks unless it is mentioned to use another library (BlackJAX for the last task). You can make use of Google colab to save time in setting up things from scratch. However, you are free to use any other platform.
- Your code should be submitted via a GitHub repository specifically created for this purpose (With a well-written README file to navigate).
- You should create a well-documented, self-contained Jupyter notebook explaining all the details of your experiments (See this for example).
- The quality of your code will be one of the evaluation factors so make sure to follow PEP 8 guidelines while writing your code. Vectorize your code wherever possible.
- Please cite the resources you have referred to accomplish the tasks.
- Given the limited time, it is not necessary to complete all the tasks. Even if you be able to do only one task partially, do it very well to maximize your chances of selection.
-
- Please do not ask any questions related to these tasks to Prof. Nipun Batra or Zeel Patel over mail or any other medium. You are allowed to make all the required assumptions on your own but mention them clearly.

Tasks

1. Animate bivariate normal distribution. [10 Marks]



- Reproduce the above figure showing samples from bivariate normal with marginal PDFs from scratch using JAX and matplotlib.
- Add interactivity to the figure by adding sliders with ipywidgets. You should be able to vary the parameters of bivariate normal distribution (mean and covariance matrix) using ipywidgets.

2. Implement from scratch a sampling method to draw samples from a multivariate Normal (MVN) distribution in JAX. [10 Marks]

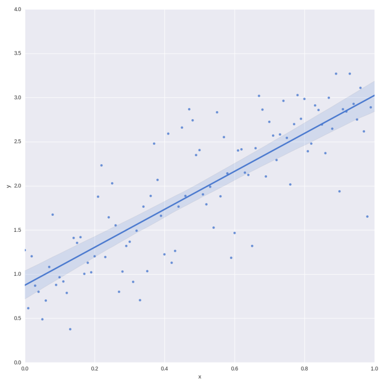
- Your code should work for any number of dimensions but please set the number of dimensions (random variables of MVN) to 10 for this task.
- You are only allowed to use jax.random.uniform. You are especially not allowed to use jax.random.normal.
- You should randomly create the mean and covariance matrix to fully specify an MVN distribution.
- Implement a sampling method from scratch using which you can draw samples from the specified MVN distribution.
- Use your sampling method to draw multiple samples from the MVN distribution and reconstruct the parameters of your MVN distribution (mean and covariance matrix) to confirm that your sampling method is working correctly.

3. Implement two hidden layers neural network classifier from scratch in JAX [20 Marks]

- Two hidden layers here means (input - hidden1 - hidden2 - output).
- You must not use flax, optax, or any other library for this task.
- Use MNIST dataset with 80:20 train:test split.
- Manually optimize the number of neurons in hidden layers.
- Use the gradient descent functionality of JAX to optimize your network. You should use the Pytree concept of JAX to do this elegantly.
- Plot loss v/s iterations curve with matplotlib.
- Evaluate the model on test data with various classification metrics and briefly discuss their implications.

4. Bayesian Linear Regression from scratch with BlackJAX [20 Marks]

- Implement Bayesian Linear Regression from scratch with any appropriate sampling method in BlackJAX.
- Create your own 1d linear dataset with added noise.
- Plot the learned predictive mean and 2 standard deviations around the mean like the below plot.



4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
y
0.0
0.2
0.4
0.6
0.8
1.0
x