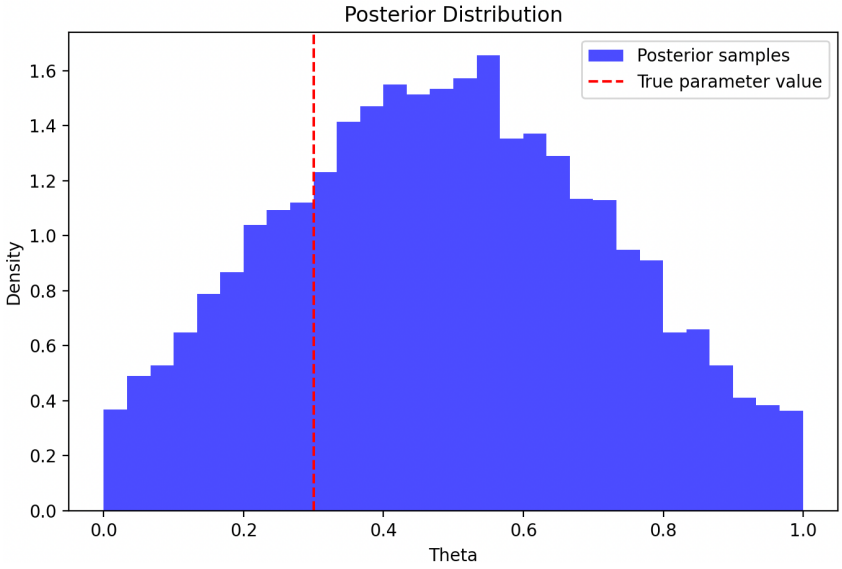
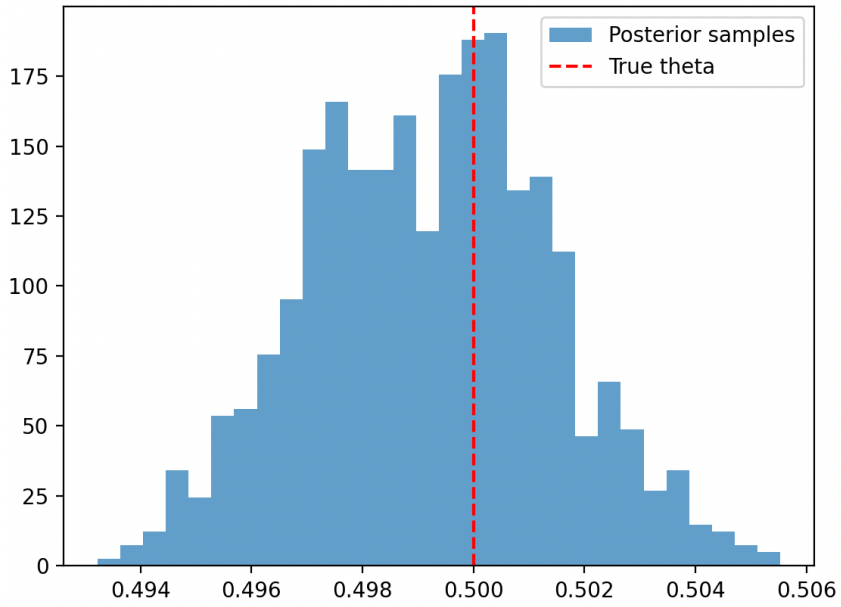
SBI Plots

1. Custom Embedding Net 1
   - Most basic embedding net – FC layer of linear RELU linear RELU linear
   - This specific simulator was returning the number with some Gaussian noise – uniform prior

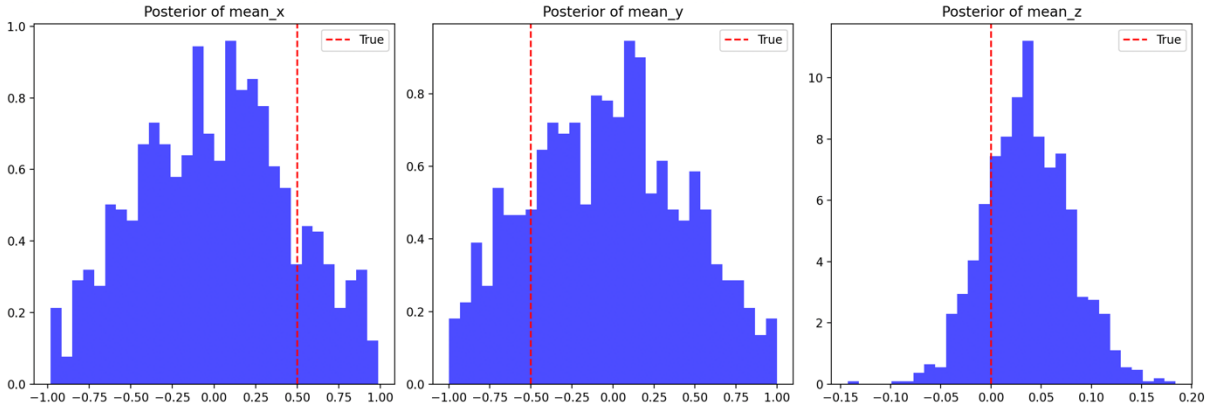


2. Custom Embedding Net 2
   - Mixture of convolutional layers and FC layers – built up as a 1D conv followed by layers of RELU and pooling, then flatten and put through FC layer
   - Simulator is sequence of sin(pi*theta), with the prior as a box uniform distribution

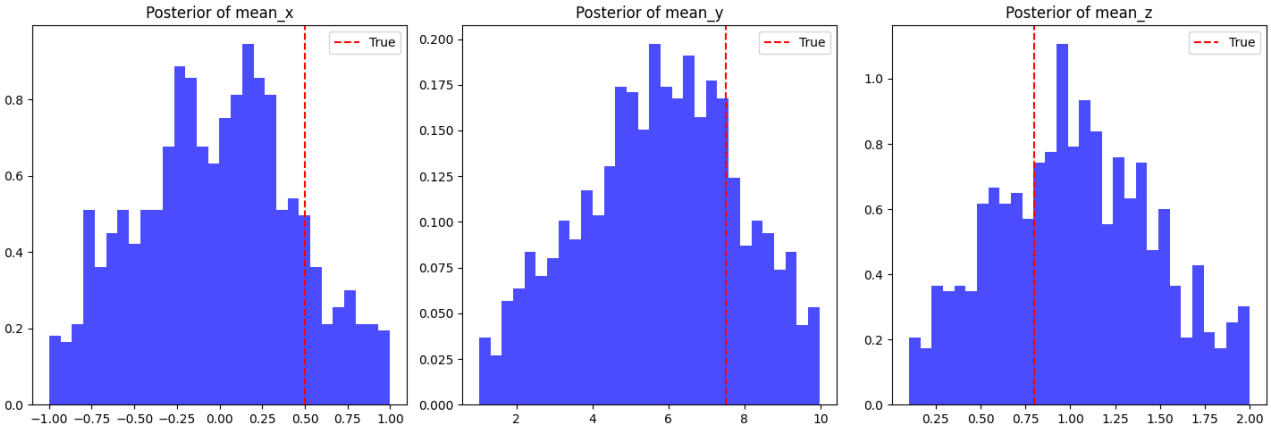


3. Custom Embedding Net 3
   - Similar to above but now it is a 3D convolutional layer, and we plot the marginals of only the x, y and z values
   - Simulator is some Gaussian blob for each batch, followed by adding some Gaussian noise, prior is a box uniform distribution in each of the three variables

SBI Plots



4. Custom Embedding Net 4
    - Recurrent layer – in the form of a LSTM
    - Simulator is sequence of sin(pi*theta), with the prior as a box uniform distribution



5. Custom Embedding Net 5
    - Attempt at a GNN for the embedding net but couldn't get it to work
    - Takes in graph like data and then has various convolutional layers on the graph nodes etc.
    - Kept getting issues with the batches and extracting the information from the graph – when that got resolved got dimensionality issues

General Observations

- The simple embedding net wasn't very powerful – didn't improve the posterior as much as the others, whilst definitely more concentrated than the prior, it didn't 'zoom in' enough – also not concentrated around the true parameter
- 3D convolutional was interesting – seemed to be very good at one marginal, zooming in a lot, but quite poor at one of them, no matter how I changed the dimensions – also changed which one it was quite good at and which one it was bad at
- Similar story with the 3D RNN, although more even there
- There is a sweet spot with the hidden layer dimensions and other hyperparameters – just going to have to experiment?