UNIVERSITY OF CAMBRIDGE

MPhil in Data Intensive Science

Lent Term 2025

A3: HEP Minor Module Coursework

Matt Kenzie and Sven Krippendorf

*This coursework has **two sections**.*

*You must attempt **one question** from **Section A** and one question from **Section B** (**40+40 marks = 80 marks** total) and **one question** from **Section C** (**80 marks** total). The appropriate number of marks allocated to each question, to be assessed from the code you provide and your report, is indicated in the right margin inside square, [], brackets.*

*Answers to **Section A** and **Section B** questions should be short and concise (no more than 500 words) with some corresponding code and plots. **Section C** contains longer format essay style questions. You are expected to write an essay (no more than 2000 words) along with some corresponding code and plots which address the problem. For the oral assessment you should prepare a short presentation (no more than 15 minutes) on the topic and be prepared to answer some questions on it.*

This coursework should be submitted via a `GitLab` repository which will be created for you. Place all of your code and your report into this repository. The report should be in pdf format and placed in a folder named `report`. You will be provided access to your repository until 23:59 on Friday 4$^{\text{th}}$ April 2025, after which your work will be deemed ready to assess.

You are expected to submit code and associated material that demonstrates good software development practices as covered in the Research Computing module.

Your report should not exceed 3000 words (including tables, figure captions and appendices but excluding references); please indicate the word count on the front cover. You are reminded to comply with the requirements given in the Course Handbook regarding the use of, and declaration of use of, autogeneration tools.

SECTION A
Attempt **one** *question from this Section.*

A1 **Pauli-Lubanski vector**

In the lectures, we have implemented the learning of invariants based on massive particles characterised by the Casimir associated to 4-momentum. In this question, you should generalise this approach to incorporate the Pauli-Lubanski vector based on the existing implementation from the course repository.

- Write a function that generates appropriate data products similar to the existing implementations for the Casimir associated to 4-momentum. Implement the calculation of the Pauli-Lubanski vector and test that your generated data and function has the appropriate properties.
- Now create an appropriate training set and train a neural network to identify invariances in the data. As your network might not identify the known invariances you should investigate by modifying your training data to reproduce invariant quantities which are evidently related to the Pauli-Lubanski vector.

[20]

A2 **Weight sharing layers**

In this question you should construct a layer which treats an image in an equivariant way under rotations by 0, 90, 180, and 270 degrees.

- In your report, describe theoretically using appropriate analytical formulae how you implement such a layer.
- Explicitly construct such a layer using a custom Keras layer and verify its performance using appropriate MNIST images.
- Train a neural network using MNIST images which are rotated by 0, 90, 180, or 270 degrees. You should compare the performance of a fully connected neural network and a network using your custom layer.

[20]

SECTION B
Attempt **one** *question from this Section.*

B1 **More on the Higgs discovery**

In the lectures we looked at some of the Open Data released by the ATLAS collaboration. We used a sample corresponding to 10 fb^{-1} and measured a signal strength approximately two standard deviations from zero. ATLAS has released a larger

dataset of Run 2 data, corresponding to an integrated luminosity of 36.1 fb^{-1} which can be found here:

`https://atlas-opendata.web.cern.ch/atlas-opendata/13TeV/GamGam/Data/`

With what statistical significance can you see the Higgs signal with this new sample? Please note that the latest release of data is in a slightly different format from the lectures. The isolation variables are merged into a new variable simply called `photon_isTightIso` which you can use instead of the `photon_ptcone30` and `photon_etcone20` cuts from the lecture. The new data sample has also been curated in a slightly different way and there is no corresponding signal simulation Monte-Carlo for it. You can use the previous signal simulation Monte-Carlo from the lectures but you will have to scale down the total expected signal yield by a factor of six (this will not effect the significance with which you see the signal, just the value of the signal strength measured). With the increased sample size you can afford to be more aggresive with the cuts (follow the instructions below).

You should:

- Be aware that the new data sample provides the `photon_pt` and `photon_e` in units of GeV not MeV.
- Modify the selection to include `photon_isTightIso` instead of the `photon_ptcone30` and `photon_etcone20` cuts.
- Modify the selection to cut on the leading photon $p_T > 50$ GeV and the subleading photon $p_T > 35$ GeV.
- Modify the selection to cut on both photons $p_T/m_H > 0.35$.
- Perform a fit to the invariant mass of selected diphoton candidates in the range $100 \leq m_H \leq 160$ GeV and produce a plot of the fit result along with a residual.
- Produce a profiled delta log-likelihood scan for the signal strength relative to the SM expectation (in the literature often referred to as μ)
- State the statistical significance with which you see the signal based on the log-likelihood ratio between the background-only fit and a signal-plus-background fit.

[20]

B2 **Separating signal and background**

In the lectures we saw two examples of using a Boosted Decision Tree to separate signal and background events. Using the data and simulation samples for a $B_s^0 \to D_s^- \pi^+$ decay (where $D_s^- \to K^+K^-\pi^-$) that can be found here:

`https://mkenzie.web.cern.ch/mkenzie/mphil/assignment/2425/Bs2DsPi`

A `info.md` file is also provided in this directory which explains the strucutre of the files. You should:

- Train a Boosted Decision Tree using the `xgboost` package that separates the signal and background using appropriate variables from the `TTree`

- Perform a preliminary fit to the data to estimate the amount of signal and background before any cut
- Choose an optimal cut by maximising an appropriate Figure Of Merit (for example $S/\sqrt{S+B}$)
- Place the cut and produce an estimate of the signal-to-background ratio in a region of $\pm 3\sigma$ around the B_s^0 mass peak.

[20]

SECTION C
Attempt **one** *question from this Section.*

C1 Normalising Flows

In this question we are interested in using normalising flows to sample from distributions. In particular, we are interested in learning the mapping between a normal and a uniform distribution in different dimensions.

You will find the tutorial found in this paper (arxiv.org/abs/2101.08176) useful. For this question, you are required to use TensorFlow or JAX, i.e. not Pytorch.

- Find an analytic mapping between a product of two uniform $\mathcal{U}(0,1)$ distributions and the product of two normal distributions in two dimensions. Verify this mapping empirically by implementing and testing it in Python.
- Set up a normalising flow that learns this mapping using data samples from a uniform distribution, i.e. your optimisation target should be based on the closeness to the Gaussian distribution. You should perform some hyperparameter optimisation and describe in your report the loss and architecture you are using. Verify your loss using the analytic mapping.
- Set up a loss where you train in the other direction, i.e. you should start with Gaussian samples and then evaluate how close they are to the uniform distribution.
- For both cases implement a rejection sampling function which allows you to generate trustworthy samples. In your report, you should describe this method. Generate such samples and check whether your sample is accurate. Compare the acceptance rate of your trained model with the acceptance rate of an untrained model.
- Finally, write a function which evaluates a lattice configuration of a scalar field in two dimensions based on the action $e^{-S[\phi]}$ for the discrete version of this action:

$$S = \int dxdy \left[\frac{1}{2}\left(\partial_x\phi(x,y)\partial_x\phi(x,y) + \partial_y\phi(x,y)\partial_y\phi(x,y)\right) + m^2\phi^2(x,y) + \frac{\lambda}{4!}\phi^4\right] .$$

Evaluate the e^{-S} factor for a few discrete random field configurations and test the correctness of the resulting suppression factor.

[80]

C2 **Event Classification with Neural Networks**

In this question we want to produce an event classifier which can “tag" the flavour of hadronic Z^0 boson decays.

Some samples of simulated e^+e^- collision events at the Future Circular Collider with the IDEA detector can be found here:

`https://mkenzie.web.cern.ch/mkenzie/mphil/assignment/2425/FCC`

A `info.md` file is also provided in this directory which explains the strucutre of the files.

These events simulate an e^+e^- collision producing a Z^0 boson at threshold (i.e. at rest). There are files corresponding to subsequent decays of $Z \to b\bar{b}$, $Z \to c\bar{c}$ and $Z \to s\bar{s}$. Your task is to:

- Explain the principle of event classification at particle physics detectors and how it can be implemented with modern neural network architectures.
- Describe what features may be useful for tagging which species of quark the Z^0 boson has decayed to
- Use the `keras` API of `tensorflow` to build a multiclass classifier. The specific architecture of this network is up to you but the example from the lectures used a Recurrent Neural Network, although DeepSets and Transformers have been shown to also perform well in the literature.
- Produce plots which show the three classifier outputs (i.e. the three classification probabilities) for each class
- Provide a value for the categorical accuracy of your network evaluated on a validation sample

[80]

END OF PAPER