# Exercise Sheet #4

Due: 20.11.2023 11h00

Jan Pawlowski, Tilman Plehn
Jonas Spinner

**General Regulations.**

- Please hand in your solutions in groups of three people. A mix of attendees from different tutorials is fine. We will not correct submissions from single students.
- Your solutions to theoretical exercises can be either handwritten notes (scanned to `pdf`), typeset using LaTeX, or directly in the jupyter notebook using Markdown.
- For the practical exercises, the data and a skeleton for your jupyter notebook are available at https://github.com/heidelberg-hepml/mlph2023-Exercises. Always provide the (commented) python code as well as the output, and don't forget to explain/interpret the latter, we do not give points for code that does not run. Please hand in both the notebook (`.ipynb`) and an exported pdf. Combine your the pdfs from theoretical and notebook exercises into a single pdf.
- Submit all your files in the Übungsgruppenverwaltung, only once for your group of three. Please list all names and tutorial numbers to simplify things for us.

## 1 Learning with pytorch

We will now use pytorch to check the results from exercise 4 on the last sheet. You do not need the results from the last sheet to do this exercise.

**(a)** Use pytorch to compute the forward and backward traces in exercises (b), (c). If you have have used pytorch for exercise 4 on the last sheet, do this exercise 'by hand'. Hint: You should use `torch.autograd.grad` here, the usual `torch.Tensor.backward` would only store gradients in leaf nodes. (1 pts)

**(b)** Use pytorch to compute the value of $x_1$ after 2 optimization steps. Train for more epochs to find the optimal value for $x_1$. Hint: You can now use `torch.Tensor.backward`. (2 pts)

**(c)** Use pytorch to compute $dL/dx_1$ for $x_1 \in [0, 2]$ and visualize the result. Does your result for the optimal $x_1$ agree with what you found in part (b)? (1 pts)

## 2 Amplitude Regression

Matrix elements of LHC processes can be calculated analytically in perturbation theory, but are often costly to evaluate. This is a major bottleneck in LHC event generators, where these amplitudes have to be evaluated numerous times for different points in phase space. In this exercise we train neural networks to interpolate between phase space points in a training dataset, focusing on the scattering of two gluons into photons and another gluon $gg \to \gamma\gamma g$ studied in https://arxiv.org/pdf/2106.09474.pdf.

In this exercise we will go through the standard workflow of tackling a problem with machine learning, i.e. studying the dataset, setting up a model, finding good hyperparameters, and finally studying the results. Use the pytorch documentation for help with the implementation, a good starting point is https://pytorch.org/tutorials/beginner/basics/intro.html.

**(a)** We start by inspecting the dataset to get some intuition. Load the dataset from https://www.thphys.uni-heidelberg.de/~plehn/pics/tutorial-2-data.zip. The four-momenta are sorted as $(g_{\text{in},1}, g_{\text{in},2}, \gamma_{\text{out},1}, \gamma_{\text{out},2}, g_{\text{out},3})$. Visualize the following distributions and explain what you see.

**(i)** Amplitude $A$

**(ii)** $\sum_i p_{x,i}, \sum_i p_{y,i}, \sum_i p_{z,i}$ with the sum running over all incoming particles

**(iii)** Transverse momentum $p_T = \sqrt{p_x^2 + p_y^2}$ of the leading and subleading photon

**(iv)** Missing transverse energy

(1 pts)

**(b)** Neural networks like $\mathcal{O}(1)$ numbers. Come up with a (bijective) preprocessing that transforms both the model input $(p_{g1}, p_{g2}, p_{\gamma_1}, p_{\gamma_2}, p_{g3})$ and the target model output $A$ into $\mathcal{O}(1)$ numbers. Then derive a custom dataset class from `torch.utils.data.Dataset`. Use this dataset to create dataloaders for the train, validation and test datasets with batch size 64. (2 pts)

**(c)** Implement a feed-forward regression net with 2 hidden layers, 32 hidden dimensions and ReLU activations for this task. Test your network by training it for 10000 iterations on the first batch using a MSE loss and Adam optimizer with learning rate $10^{-3}$. (2 pts)

**(d)** Now train your model on the full training dataset. (1 pts)

**(e)** Assume you have a reduced training dataset of only 1000 events. Retrain the network, what problem do you observe? Pick one of the following regularization techniques to overcome this problem.

- Early stopping
- Dropout
- Weight decay. Hint: `AdamW`

Explain and implement your regularization approach. Track the validation loss to check that the regularization works. (2 pts)