Summary

Lorem ipsum

1 Introduction

2 Gross Pitaevskii Equation

2.1 Types of Potentials

2.2 Analytical Solution and Approximation

2.3 Numerical Solution and XMDS Framework

3 Problem Statements and Dataset Generation

3.1 Dataset and Dataset Generation

Datasets have two elements; potential vector and corresponding total energy value. Length of the potential vector is 128 which equals to grid number used in XMDS. This length also determines the number of neurons in the input layer.

There are four datasets in total and each of them corresponds to one interaction parameter, thus; their total energy values are different because of solution and potential vectors are also different because of randomness.

4 Machine Learning for NLSE

We used Pytorch Framework to build our neural networks. It allows the client codes work on both CPU and GPU via its internal python object called Tensor. If any CUDA supported GPU is available then Pytorch can use GPU without any change in the code. Code have three main parts, first one is dataloaders; it reads train and test data for specified interaction parameter from corresponding file and generates tensor dataset object. **continue**

We implemented two different types of neural network; feed forward (FNN) and convolutional neural network (CNN).

4.1 Architecture

As it is mentioned, there are two different architectures; feed forward (FNN) and convolutional neural network (CNN).

FNN involves 128 input neurons as input layer, next layer is first hidden layer with 30 neurons, the second is same as first hidden layer, the next one is last hidden layer with 10 neurons and the last layer is output layer. Totally there are 5 layers in our FFN and we will denote as $FNN[128, 30, 30, 10, 1]$. Rectified linear unit (ReLU) is used for each forward except output. No operation is applied to the output neuron. Learning rate of the FNN is fixed and it is 0.001. Cost function is mean squared error (MSE) and optimization is done with Adam.

CNN has two convolution layers, two maxpool layers and three fully connected layer and last layer of the fully connected part is output layer which is a single neuron. Maxpooling is applied to output of the first and second convolution layers. ReLu is also applied each forward except output neuron same as in the FNN. Fully connected part of the CNN is $FNN[310, 100, 20, 1]$. Learning rate, cost function and optimizer of the CNN is same as FNN which are 0.001, MSE and Adam respectively.

4.2 Hyperparameters

Before traning the network with all generated dataset which involves 8500 elements for training and 1500 elements for test, we used small two datasets which are 800/200 and 3500/500 (training/test).