PHYSICS-INFORMED NEURAL NETWORKS FOR DENSITY FUNCTIONAL THEORY

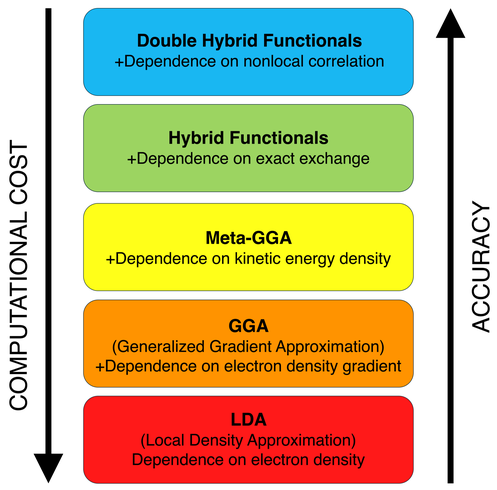
Joshua Franklin[1], Alec Wills[2], Marivi Fernández-Serra[2]
[1]Arizona State University, [2] Stony Brook University

ABSTRACT

Density functional theory (DFT) is the most widely used quantum mechanical computational method for calculating the properties of many-electron systems. Due to the extremely complex interactions between electrons, it is impossible to calculate these properties exactly. With DFT, however, we can obtain good approximations with relatively low computational expense. The key parameter one must choose when doing a DFT calculation is the exchange-correlation ("XC") functional, which serves as the mathematical engine for approximating electrons' complex behavior. In this study, we use machine learning to develop accurate XC functionals. In particular, we explore how imposing exact constraints based on known physical laws can improve results, an approach known as physics-informed machine learning. Ultimately, we hope to discover the optimal procedure for training new XC functionals.

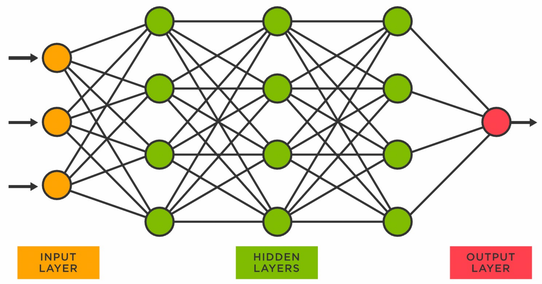
DENSITY FUNCTIONAL THEORY

- **Density functional theory** (DFT) is a popular computational method based on quantum mechanics that is useful for understanding **electronic structure**.
- It is widely used in **condensed matter physics**, **chemistry**, and **materials science**.
- The highly correlated interactions in **many-electron systems** make exact calculations impossible.
- DFT reduces computational complexity by allowing us to calculate the properties of a system of n electrons by using only the **electron density** (a function of only 3 spatial coordinates) rather than the electronic wave function (which would involve $3n$ coordinates).
- The most important parameter is the choice of **exchange and correlation functionals** ("XC functionals").
- Thousands of XC functionals have been proposed based on both **theoretical** and **empirical** considerations.
- XC functionals belong to different **levels of theory**. Greater accuracy entails greater computational cost.



NEURAL NETWORKS

- **Machine learning (ML)** is an approach to AI in which a computer system "learns" from input data through statistical pattern recognition, developing the ability to make accurate predictions about new data, rather than being explicitly programmed with specific rules by a human programmer.
- **Neural networks** are a specific type of ML model, inspired by the structure of biological neural networks in the brain.
- They consist of multiple layers of interconnected **nodes** (or "neurons") organized into an **input layer**, one or more **hidden layers**, and an **output layer**.



- In the architecture we use (the **multilayer perceptron**), each node in a layer is connected to every node in the subsequent layer. Each connection has an associated **weight** that modulates the influence of the previous node on the subsequent node.
- In addition, each node applies an **activation function** to its input to introduce **non-linearity** into the network, enabling it to learn complex patterns.
- During **training**, the network iteratively processes training data, calculates the **error** between its output and the expected output, calculates the **gradients** of the loss function with respect to the weights, and then updates the weights to minimize the loss function based on some optimization algorithm.
- After training, the network is tested on a **test set** of data.

METHODOLOGY

1. Generate a set of networks with **randomized weights** and different combinations of the following variables:
   A. Exchange/Correlation
   B. Level of Theory: GGA/Meta-GGA/Non-Local
   C. Constrained/Unconstrained
   D. Target Functional: PBE/PBE0/SCAN/(none)
2. **Pretrain** networks on XC energy density data generated by target functionals for a small subset of molecules.
3. Combine pairs of exchange and correlation functionals into **new XC functionals**.
4. Perform **full training** on the XC functionals using atomization energy values for ~150 molecules.

EXAMPLE: LEARNING THE SINC FUNCTION WITH CONSTRAINTS

To illustrate the principles behind our research, we can consider a simple example in which we train a neural network to learn the sinc function. Sinc is defined as

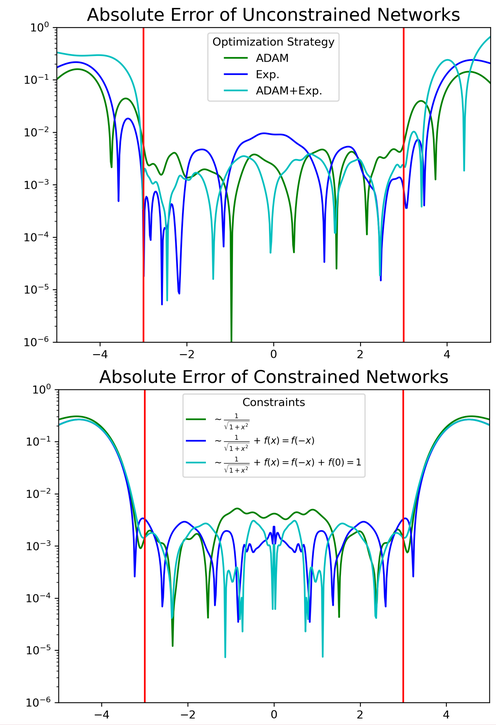
$$\operatorname{sinc} x = \frac{\sin x}{x}.$$

Based on our knowledge of this function's behavior, we can impose constraints to limit the hypothesis space explored by the network during training. For instance, we know that

$$\lim_{x\to\infty} \operatorname{sinc} x = \lim_{x\to-\infty} \operatorname{sinc} x = 0,$$

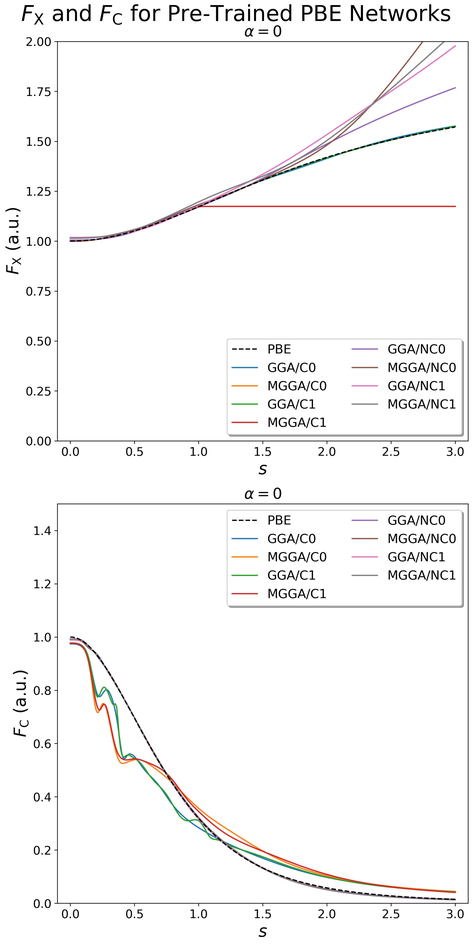
$$\operatorname{sinc}(0) = 1, \quad \operatorname{sinc} x = \operatorname{sinc}(-x).$$

With these constraints, the constrained networks exhibit significantly lower errors in the regions immediately outside the training region bounded in red, indicating superior generalization.



RESULTS

These graphs show preliminary results for various networks pretrained to approximate PBE, a popular GGA functional. The constrained GGA networks (GGA/C0 and GGA/C1) reproduce the exchange portion of PBE (F_X) almost perfectly, whereas the unconstrained GGA networks do best on the correlation portion (F_C).



FUTURE WORK

- Train a network to **reproduce the PBE functional** based on a mesh of ρ and $\nabla\rho$ values.
- Investigate the performance of networks composed of **constrained X** and **unconstrained C** functionals.
- **Hyperparameter Optimization.** Determine the optimal…
  - Number of Hidden Layers/Nodes
  - Optimization Function
  - Activation Function
  - Learning Rate
  - Batch Size

ACKNOWLEDGMENTS

I would like to thank Professor Marivi Fernández-Serra and Alec Wills for their exceptional mentorship on this project, and Professor Navid Vafaei-Najafabadi and the rest of the Stony Brook physics department for putting together such a great REU program. Computations were performed using Stony Brook's SeaWulf computing cluster. This material is based on work supported by the NSF under Grant No. PHY-2243856 and PHY-1852143.