Developing Accurate Exchange-Correlation Functionals through Physics-Informed Machine Learning

Sara Navarro-Rodriguez,[1] Alec Willis,[2,3,4] Joshua Franklin,[2,3] Federico Nicolás Pedron, [1] Pablo Ordejón,[1] Marivi Fernandez-Serra[2,3]
[1]Catalan Institute of Nanoscience and Nanotechnology (ICN2), CSIC, BIST, 08193 Barcelona, Spain
[2]Physics and Astronomy Department, Stony Brook University, Stony Brook, New York 11794-3800, United States
[3]Institute for Advanced Computational Science, Stony Brook University, Stony Brook, New York 11794-3800, United States
[4]The New York Academy of Sciences, 115 Broadway, New York, NY 10006, United States
The New York Academy of Sciences
Sara.navarro@icn2.cat

In this work, we present a way to create accurate exchange-correlation functionals for density functional theory using neural network models and grid-based density descriptors, developing a machine learning framework called *xcquinox*.

Density Functional Theory (DFT) is the standard method for studying the electronic structure of matter at the atomic scale, as it achieves an optimal balance between accuracy and computational cost. This enables a first-principles description of complex and large systems that would otherwise be inaccessible to more precise ab initio methods [1].

DFT simplifies the many-body electron interactions into a mean-field, single-electron description within the Kohn–Sham (KS) [2] approach. In Kohn–Sham DFT (KS-DFT), the exchange-correlation (XC) functional captures electron exchange (from Pauli exchange effects) and correlation (from electron interactions in the many-body wavefunction). However, this functional is only available in approximate forms; and usually the computational balance between accuracy and cost of KS-DFT is determined by that of the XC functional. A systematic way to improve this XC functionals was proposed by Perdew in his "Jacob's ladder of density functional approximations" [3, 4], which introduces complexity in a controlled and physically motivated manner. Essentially this aims to approximate the exact XC functional by adhering to an incremental list of necessary physical constraints.

While this "Jacob's ladder" has been a guiding principle for several DFT functionals, progress has been relatively slow as the formal mathematical application of these constraints can get increasingly complex. But what if one could generate a DFT XC functional automatically, essentially fitting a neural network (NN) to known results from higher levels of theory, while also imposing all the possible constraints?

We start from work already done in Fernandez-Serra's group, in which they created a NN model that relies on grid-based density descriptors [5]. In this work, they aim to construct an auto-differentiable physically informed NNs to obtain the XC enhancement factors ($F_{x/c}$), which are defined as

$$\epsilon_{xc} = F_x \epsilon_x^{UEG}(n, \omega) + F_c \epsilon_c^{UEG}(n, \omega).$$

This framework is called *xcdiff.* They perform this in 2 steps: first, they pre-train these NNs to fit known $F_{x/c}$ functionals. Since starting the network from random weights would cause the SCF cycles to crash, adding this pretraining they ensure convergence. Secondly, the networks are trained by fitting atomization energies, ionization potentials and barrier heights obtained from DFT, and then they complete this with ground state electron densities and total atomic energies obtained with CCSD(T).

xcdiff was later expanded into a new ML framework called *xcquinox*. *xcquinox* employs neural network architectures implemented in JAX, a library for automatically differentiable mathematical operations, allowing us to leverage PySCF-AD—an extension of the PySCF package that incorporates automatic differentiation capabilities. The use of an auto-differentiable network facilitates the access to derivatives of XC functionals, which are, in fact, the XC potentials used in the DFT calculation itself. *xcquinox* also includes the usage of descriptors to try to incorporate long-range effects.

One thing that has not yet been fully explored is how successful these NN are at modeling density functionals in general. The noise in NN derivatives can effectively affect the generated XC potential, with a great impact in the self-consistency cycle.

In this work, we aim to generate NN-based functionals that can reproduce already known XC functionals. We will focus on the pre-training step, since this will allow us to determine how errors propagate in learning a new XC functional starting from one already known.

To refine the pre-training, we investigate the impact of imposing exact constraints based on established physical laws to improve results, using a physics-informed approach.

Additionally, we emphasize the importance of incorporating either exact potentials or exact densities in the training process, as training an energy functional without information about functional derivatives—essential for determining the XC potential—can lead to models that predict correct energies but yield incorrect densities.

References

[1] Jones, R. O., Rev. Mod. Phys., 3 (2015) 897.
[2] Kohn, W. & Sham, L. J., Phys. Rev., A4, (1965) A1133.
[3] Perdew, J. P. and Schmidt, K., AIP Conference Proceedings, 1 (2001) 1.
[4] Perdew, J. P. et al., J. Chem. Phys., 6 (2005) 062201.
[5] Dick, S. and Fernandez-Serra, M., Phys. Rev. B, 6 (2021) L161109.

Figures

In figures 1-4, we show the absolute errors for our trained models for the exchange and the correlation enhancement factors (F_x and F_c respectively). These errors are evaluated for the enhancement factors themselves, their derivatives with respect to the charge density *(n)* and their derivatives with respect to the gradient of the charge density (∇n). We plot these values with respect to *s,* which is a function of the charge density and its gradient:

$$s = \frac{\nabla n}{2(3\pi^2)^{1/3} n^{4/3}}.$$

We show the differences between a network trained without adding the gradients' effect to the loss (fitting values) and adding it (fitting values and gradients). Preliminary results show that, in the case of exchange, the results are better if we consider these derivatives in the loss, but, contrarily, in the correlation, the results become worse.

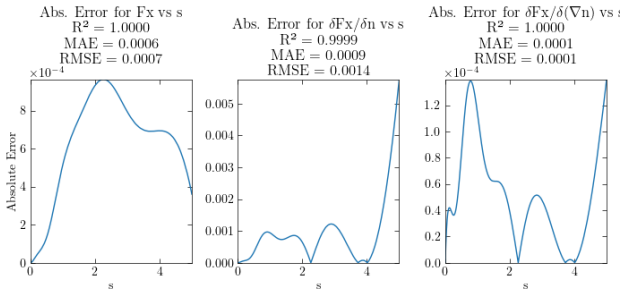

Figure 1. Absolute errors for our network for F_x, fitting it only to match values.

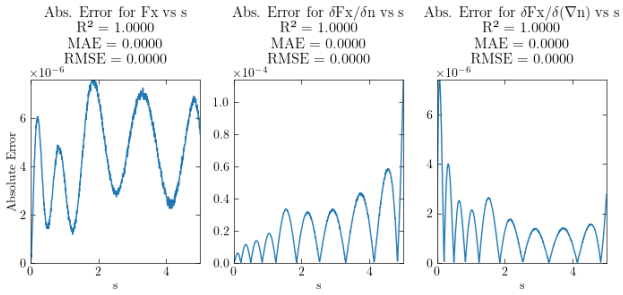

Figure 2. Absolute errors for our network for F_x, fitting it to match values and gradients.

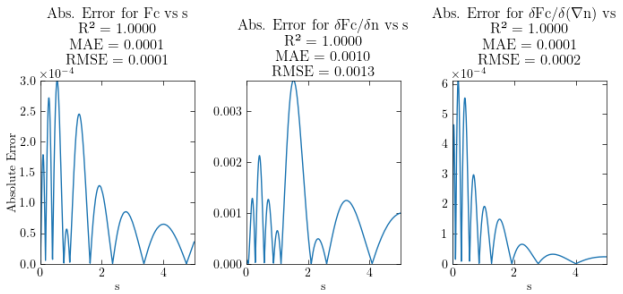

Figure 3. Absolute errors for our network for F_c, fitting it only to match values.

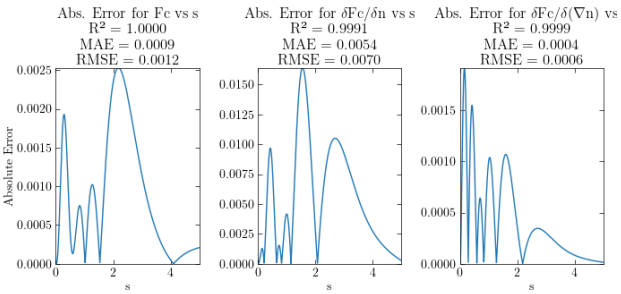

Figure 4. Absolute errors for our network for F_c, fitting it to match values and gradients.