

Artificial intelligence and machine learning for quantum simulations

Morten Hjorth-Jensen¹

Department of Physics and Astronomy and FRIB, Michigan State University, USA, and Department of Physics and Center for Computing in Science Education, University of Oslo, Norway¹

November 6, 2023

What is this talk about?

The main emphasis is to give you a short and pedestrian introduction to the whys and hows we can use (with several examples) machine learning methods in physics. And why this could (or should) be of interest.

Additional info

Parts of this talk are based on [Artificial Intelligence and Machine Learning in Nuclear Physics](#), Amber Boehnlein et al., *Reviews Modern of Physics* 94, 031003 (2022)

AI/ML and some statements you may have heard (and what do they mean?)

1. Fei-Fei Li on ImageNet: **map out the entire world of objects** ([The data that transformed AI research](#))
2. Russell and Norvig in their popular textbook: **relevant to any intellectual task; it is truly a universal field** ([Artificial Intelligence, A modern approach](#))
3. Woody Bledsoe puts it more bluntly: **in the long run, AI is the only science** (quoted in Pamilla McCorduck, [Machines who think](#))

If you wish to have a critical read on AI/ML from a societal point of view, see [Kate Crawford's recent text Atlas of AI](#). See also <https://www.nationaldefensemagazine.org/articles/2023/3/24/ukraine-a-living-lab-for-ai-warfare>

Here: with AI/ML we intend a collection of machine learning methods with an emphasis on statistical learning and data analysis

Types of machine learning

The approaches to machine learning are many, but are often split into two main categories. In *supervised learning* we know the answer to a problem, and let the computer deduce the logic behind it. On the other hand, *unsupervised learning* is a method for finding patterns and relationship in data sets without any prior knowledge of the system.

An emerging third category is *reinforcement learning*. This is a paradigm of learning inspired by behavioural psychology, where learning is achieved by trial-and-error, solely from rewards and punishment.

Main categories

Another way to categorize machine learning tasks is to consider the desired output of a system. Some of the most common tasks are:

- ▶ **Classification:** Outputs are divided into two or more classes. The goal is to produce a model that assigns inputs into one of these classes. An example is to identify digits based on pictures of hand-written ones. Classification is typically supervised learning.
- ▶ **Regression:** Finding a functional relationship between an input data set and a reference data set. The goal is to construct a function that maps input data to continuous output values.
- ▶ **Clustering:** Data are divided into groups with certain common traits, without knowing the different groups beforehand. It is thus a form of unsupervised learning.

The plethora of machine learning algorithms/methods

1. Deep learning: Neural Networks (NN), Convolutional NN, Recurrent NN, Boltzmann machines, autoencoders and variational autoencoders and generative adversarial networks, generative models
2. Bayesian statistics and Bayesian Machine Learning, Bayesian experimental design, Bayesian Regression models, Bayesian neural networks, Gaussian processes and much more
3. Dimensionality reduction (Principal component analysis), Clustering Methods and more
4. Ensemble Methods, Random forests, bagging and voting methods, gradient boosting approaches
5. Linear and logistic regression, Kernel methods, support vector machines and more
6. Reinforcement Learning; Transfer Learning and more

What are the basic ingredients?

Almost every problem in ML and data science starts with the same ingredients:

- ▶ The dataset x (could be some observable quantity of the system we are studying)
- ▶ A model which is a function of a set of parameters α that relates to the dataset, say a likelihood function $p(x|\alpha)$ or just a simple model $f(\alpha)$
- ▶ A so-called **loss/cost/risk** function $\mathcal{C}(x, f(\alpha))$ which allows us to decide how well our model represents the dataset.

We seek to minimize the function $\mathcal{C}(x, f(\alpha))$ by finding the parameter values which minimize \mathcal{C} . This leads to various minimization algorithms. It may surprise many, but at the heart of all machine learning algorithms there is an optimization problem.

Low-level machine learning, the family of ordinary least squares methods

Our data which we want to apply a machine learning method on, consist of a set of inputs $\mathbf{x}^T = [x_0, x_1, x_2, \dots, x_{n-1}]$ and the outputs we want to model $\mathbf{y}^T = [y_0, y_1, y_2, \dots, y_{n-1}]$. We assume that the output data can be represented (for a regression case) by a continuous function f through

$$\mathbf{y} = f(\mathbf{x}) + \epsilon.$$

Setting up the equations

In linear regression we approximate the unknown function with another continuous function $\tilde{\mathbf{y}}(\mathbf{x})$ which depends linearly on some unknown parameters $\boldsymbol{\theta}^T = [\theta_0, \theta_1, \theta_2, \dots, \theta_{p-1}]$.

The input data can be organized in terms of a so-called design matrix with an approximating function $\tilde{\mathbf{y}}$

$$\tilde{\mathbf{y}} = \mathbf{X}\boldsymbol{\theta},$$

The objective/cost/loss function

The simplest approach is the mean squared error

$$C(\beta) = \frac{1}{n} \sum_{i=0}^{n-1} (y_i - \tilde{y}_i)^2 = \frac{1}{n} \left\{ (\mathbf{y} - \tilde{\mathbf{y}})^T (\mathbf{y} - \tilde{\mathbf{y}}) \right\},$$

or using the matrix \mathbf{X} and in a more compact matrix-vector notation as

$$C(\beta) = \frac{1}{n} \left\{ (\mathbf{y} - \mathbf{X}\theta)^T (\mathbf{y} - \mathbf{X}\theta) \right\}.$$

This function represents one of many possible ways to define the so-called cost function.

Training solution

Optimizing wrt to the unknown parameters θ_j we get

$$\mathbf{X}^T \mathbf{y} = \mathbf{X}^T \mathbf{X} \boldsymbol{\theta},$$

and if the matrix $\mathbf{X}^T \mathbf{X}$ is invertible we have the optimal values

$$\hat{\boldsymbol{\theta}} = \left(\mathbf{X}^T \mathbf{X} \right)^{-1} \mathbf{X}^T \mathbf{y}.$$

Ridge and LASSO Regression

Our optimization problem is

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^p} \frac{1}{n} \left\{ (\mathbf{y} - \mathbf{X}\boldsymbol{\theta})^T (\mathbf{y} - \mathbf{X}\boldsymbol{\theta}) \right\}.$$

or we can state it as

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^p} \frac{1}{n} \sum_{i=0}^{n-1} (y_i - \tilde{y}_i)^2 = \frac{1}{n} \|\mathbf{y} - \mathbf{X}\boldsymbol{\theta}\|_2^2,$$

where we have used the definition of a norm-2 vector, that is

$$\|\mathbf{x}\|_2 = \sqrt{\sum_i x_i^2}.$$

From OLS to Ridge and Lasso

By minimizing the above equation with respect to the parameters θ we could then obtain an analytical expression for the parameters θ . We can add a regularization parameter λ by defining a new cost function to be optimized, that is

$$\min_{\theta \in \mathbb{R}^p} \frac{1}{n} \|\mathbf{y} - \mathbf{X}\theta\|_2^2 + \lambda \|\theta\|_2^2$$

which leads to the Ridge regression minimization problem where we require that $\|\theta\|_2^2 \leq t$, where t is a finite number larger than zero. We do not include such a constraints in the discussions here.

Lasso regression

Defining

$$C(\mathbf{X}, \boldsymbol{\theta}) = \frac{1}{n} \|\mathbf{y} - \mathbf{X}\boldsymbol{\theta}\|_2^2 + \lambda \|\boldsymbol{\theta}\|_1,$$

we have a new optimization equation

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^p} \frac{1}{n} \|\mathbf{y} - \mathbf{X}\boldsymbol{\theta}\|_2^2 + \lambda \|\boldsymbol{\theta}\|_1$$

which leads to Lasso regression. Lasso stands for least absolute shrinkage and selection operator. Here we have defined the norm-1 as

$$\|\mathbf{x}\|_1 = \sum_i |x_i|.$$

Lots of room for creativity

Not all the algorithms and methods can be given a rigorous mathematical justification, opening up thereby for experimenting and trial and error and thereby exciting new developments.

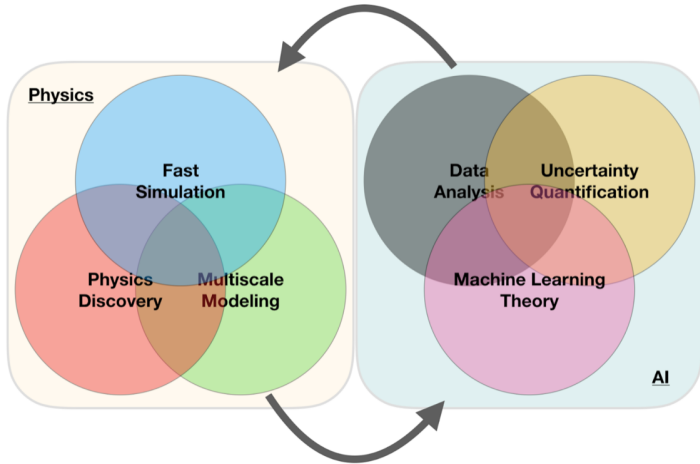
A solid command of linear algebra, multivariate theory, probability theory, statistical data analysis, optimization algorithms, understanding errors and Monte Carlo methods is important in order to understand many of the various algorithms and methods.

Job market, a personal statement: A familiarity with ML is almost becoming a prerequisite for many of the most exciting employment opportunities. And add quantum computing and there you are!

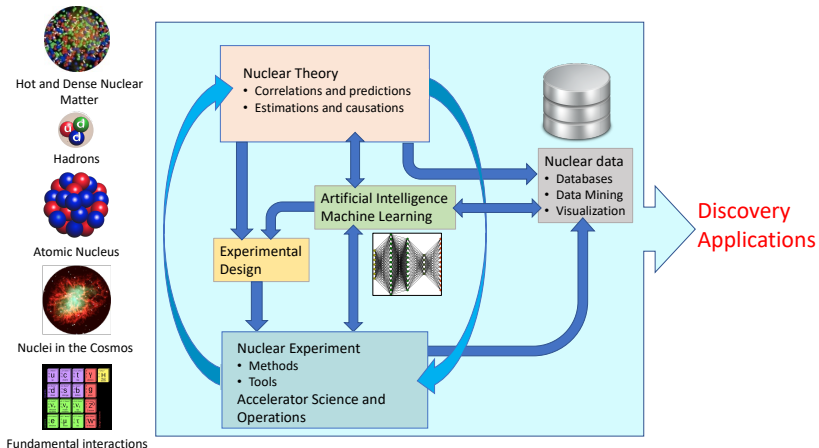
Selected references

- ▶ Mehta et al. and Physics Reports (2019).
- ▶ Machine Learning and the Physical Sciences by Carleo et al
- ▶ Artificial Intelligence and Machine Learning in Nuclear Physics, Amber Boehnlein et al., Reviews Modern of Physics 94, 031003 (2022)
- ▶ Dilute neutron star matter from neural-network quantum states by Fore et al, Physical Review Research 5, 033062 (2023)
- ▶ Neural-network quantum states for ultra-cold Fermi gases, Jane Kim et al, Nature Physics Communication, in press
- ▶ Message-Passing Neural Quantum States for the Homogeneous Electron Gas, Gabriel Pescia, Jane Kim et al. arXiv.2305.07240,
- ▶ Efficient solutions of fermionic systems using artificial neural networks, Nordhagen et al, Frontiers in Physics 11, 2023
- ▶ Particle Data Group summary on ML methods

Machine learning. A simple perspective on the interface between ML and Physics



ML in Nuclear Physics (or any field in physics)



Machine learning in physics (my bias): Why?

1. ML tools can help us to speed up the scientific process cycle and hence facilitate discoveries
2. Enabling fast emulation for big simulations
3. Revealing the information content of measured observables w.r.t. theory
4. Identifying crucial experimental data for better constraining theory
5. Providing meaningful input to applications and planned measurements
6. ML tools can help us to reveal the structure of our models
7. Parameter estimation with heterogeneous/multi-scale datasets
8. Model reduction
9. ML tools can help us to provide predictive capability
10. Theoretical results often involve ultraviolet and infrared extrapolations due to Hilbert-space truncations
11. Uncertainty quantification essential
12. Theoretical models are often applied to entirely new nuclear systems and conditions that are not accessible to experiment

Scientific Machine Learning

An important and emerging field is what has been dubbed as scientific ML, see the article by Deiana et al "Applications and Techniques for Fast Machine Learning in Science, Big Data 5, 787421 (2022):<https://doi.org/10.3389/fdata.2022.787421>"

The authors discuss applications and techniques for fast machine learning (ML) in science – the concept of integrating power ML methods into the real-time experimental data processing loop to accelerate scientific discovery. The report covers three main areas

1. applications for fast ML across a number of scientific domains;
2. techniques for training and implementing performant and resource-efficient ML algorithms;
3. and computing architectures, platforms, and technologies for deploying these algorithms.



ELSEVIER



Engineering

Volume 6, Issue 3, March 2020, Pages 264-274




Research Artificial Intelligence—Review

A Survey of Accelerator Architectures for Deep Neural Networks

Yiran Chen^a  , Yuan Xie^b, Linghao Song^a, Fan Chen^a, Tianqi Tang^b

[Show more](#) 

 Add to Mendeley  Share  Cite

<https://doi.org/10.1016/j.eng.2020.01.007> 

[Get rights and content](#) 

Under a Creative Commons [license](#) 

 [open access](#)

Abstract

Physics driven Machine Learning

Another hot topic is what has loosely been dubbed **Physics-driven deep learning**. See the recent work on [Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators](#), Nature Machine Learning, vol 3, 218 (2021).

From their abstract

A less known but powerful result is that an NN with a single hidden layer can accurately approximate any nonlinear continuous operator. This universal approximation theorem of operators is suggestive of the structure and potential of deep neural networks (DNNs) in learning continuous operators or complex systems from streams of scattered data. ... We demonstrate that DeepONet can learn various explicit operators, such as integrals and fractional Laplacians, as well as implicit operators that represent deterministic and stochastic differential equations.

And more

- ▶ An important application of AI/ML methods is to improve the estimation of bias or uncertainty due to the introduction of or lack of physical constraints in various theoretical models.
- ▶ In theory, we expect to use AI/ML algorithms and methods to improve our knowledge about correlations of physical model parameters in data for quantum many-body systems. Deep learning methods show great promise in circumventing the exploding dimensionalities encountered in quantum mechanical many-body studies.
- ▶ Merging a frequentist approach (the standard path in ML theory) with a Bayesian approach, has the potential to infer better probability distributions and error estimates.
- ▶ Machine Learning and Quantum Computing is a very interesting avenue to explore. See for example a recent talk by [Sofia Vallecorsa](#).

Many-body physics, Quantum Monte Carlo and deep learning

Given a hamiltonian H and a trial wave function Ψ_T , the variational principle states that the expectation value of $\langle H \rangle$, defined through

$$\langle E \rangle = \frac{\int d\mathbf{R} \Psi_T^*(\mathbf{R}) H(\mathbf{R}) \Psi_T(\mathbf{R})}{\int d\mathbf{R} \Psi_T^*(\mathbf{R}) \Psi_T(\mathbf{R})},$$

is an upper bound to the ground state energy E_0 of the hamiltonian H , that is

$$E_0 \leq \langle E \rangle.$$

In general, the integrals involved in the calculation of various expectation values are multi-dimensional ones. Traditional integration methods such as the Gauss-Legendre will not be adequate for say the computation of the energy of a many-body system. **Basic philosophy: Let a neural network find the optimal wave function**

Quantum Monte Carlo Motivation

Basic steps

Choose a trial wave function $\psi_T(\mathbf{R})$.

$$P(\mathbf{R}, \alpha) = \frac{|\psi_T(\mathbf{R}, \alpha)|^2}{\int |\psi_T(\mathbf{R}, \alpha)|^2 d\mathbf{R}}.$$

This is our model, or likelihood/probability distribution function (PDF). It depends on some variational parameters α . The approximation to the expectation value of the Hamiltonian is now

$$\langle E[\alpha] \rangle = \frac{\int d\mathbf{R} \psi_T^*(\mathbf{R}, \alpha) H(\mathbf{R}) \psi_T(\mathbf{R}, \alpha)}{\int d\mathbf{R} \psi_T^*(\mathbf{R}, \alpha) \psi_T(\mathbf{R}, \alpha)}.$$

Quantum Monte Carlo Motivation

Define a new quantity

$$E_L(\mathbf{R}, \alpha) = \frac{1}{\psi_T(\mathbf{R}, \alpha)} H \psi_T(\mathbf{R}, \alpha),$$

called the local energy, which, together with our trial PDF yields

$$\langle E[\alpha] \rangle = \int P(\mathbf{R}) E_L(\mathbf{R}, \alpha) d\mathbf{R} \approx \frac{1}{N} \sum_{i=1}^N E_L(\mathbf{R}_i, \alpha)$$

with N being the number of Monte Carlo samples.

Energy derivatives

The local energy as function of the variational parameters defines now our **objective/cost** function.

To find the derivatives of the local energy expectation value as function of the variational parameters, we can use the chain rule and the hermiticity of the Hamiltonian.

Let us define (with the notation $\langle E[\alpha] \rangle = \langle E_L \rangle$)

$$\bar{E}_{\alpha_i} = \frac{d\langle E_L \rangle}{d\alpha_i},$$

as the derivative of the energy with respect to the variational parameter α_i . We define also the derivative of the trial function (skipping the subindex T) as

$$\bar{\Psi}_i = \frac{d\Psi}{d\alpha_i}.$$

Derivatives of the local energy

The elements of the gradient of the local energy are

$$\bar{E}_i = 2 \left(\left\langle \frac{\bar{\Psi}_i}{\Psi} E_L \right\rangle - \left\langle \frac{\bar{\Psi}_i}{\Psi} \right\rangle \langle E_L \rangle \right).$$

From a computational point of view it means that you need to compute the expectation values of

$$\left\langle \frac{\bar{\Psi}_i}{\Psi} E_L \right\rangle,$$

and

$$\left\langle \frac{\bar{\Psi}_i}{\Psi} \right\rangle \langle E_L \rangle$$

These integrals are evaluated using MC integration (with all its possible error sources). Use methods like stochastic gradient or other minimization methods to find the optimal parameters.

Why Feed Forward Neural Networks (FFNN)?

According to the *Universal approximation theorem*, a feed-forward neural network with just a single hidden layer containing a finite number of neurons can approximate a continuous multidimensional function to arbitrary accuracy, assuming the activation function for the hidden layer is a **non-constant, bounded and monotonically-increasing continuous function**.

Illustration of a single perceptron model and an FFNN

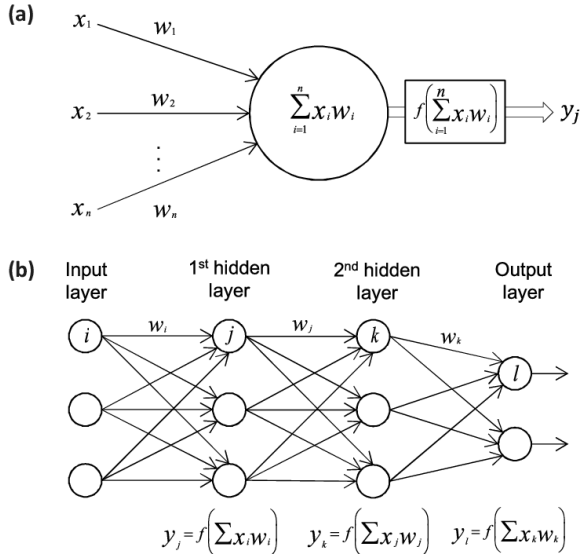


Figure: In a) we show a single perceptron model while in b) we display a network with two hidden layers, an input layer and an output layer.

Monte Carlo methods and Neural Networks

Machine Learning and the Deuteron by Kebble and Rios and
Variational Monte Carlo calculations of $A \leq 4$ nuclei with an
artificial neural-network correlator ansatz by Adams et al.

Adams et al:

$$H_{LO} = - \sum_i \frac{\vec{\nabla}_i^2}{2m_N} + \sum_{i < j} (C_1 + C_2 \vec{\sigma}_i \cdot \vec{\sigma}_j) e^{-r_{ij}^2 \Lambda^2 / 4} \\ + D_0 \sum_{i < j < k} \sum_{\text{cyc}} e^{-(r_{ik}^2 + r_{ij}^2) \Lambda^2 / 4}, \quad (1)$$

where m_N is the mass of the nucleon, $\vec{\sigma}_i$ is the Pauli matrix acting on nucleon i , and \sum_{cyc} stands for the cyclic permutation of i , j , and k . The low-energy constants C_1 and C_2 are fit to the deuteron binding energy and to the neutron-neutron scattering length

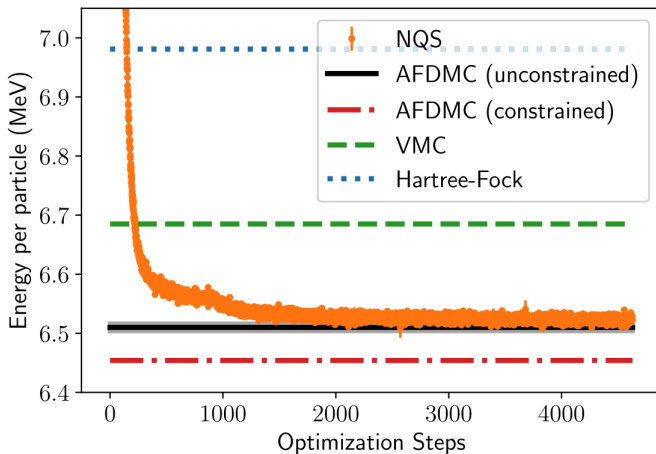
Deep learning neural networks, Variational Monte Carlo calculations of $A \leq 4$ nuclei with an artificial neural-network correlator ansatz by Adams et al.

An appealing feature of the neural network ansatz is that it is more general than the more conventional product of two- and three-body spin-independent Jastrow functions

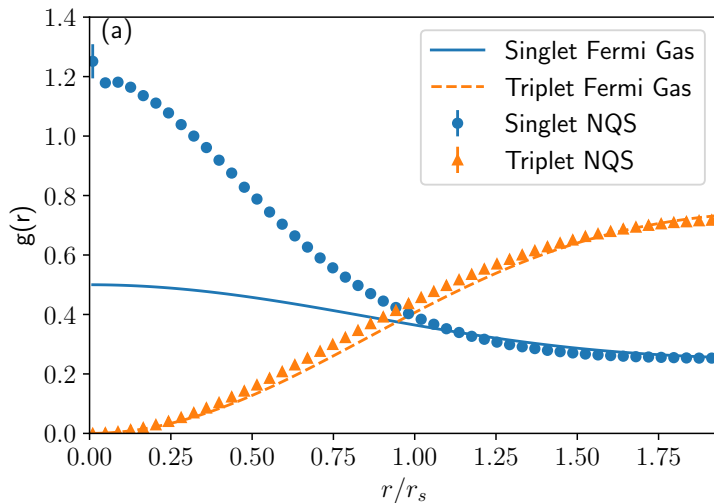
$$|\Psi_V^J\rangle = \prod_{i < j < k} \left(1 - \sum_{\text{cyc}} u(r_{ij})u(r_{jk}) \right) \prod_{i < j} f(r_{ij}) |\Phi\rangle, \quad (2)$$

which is commonly used for nuclear Hamiltonians that do not contain tensor and spin-orbit terms. The above function is replaced by a four-layer Neural Network.

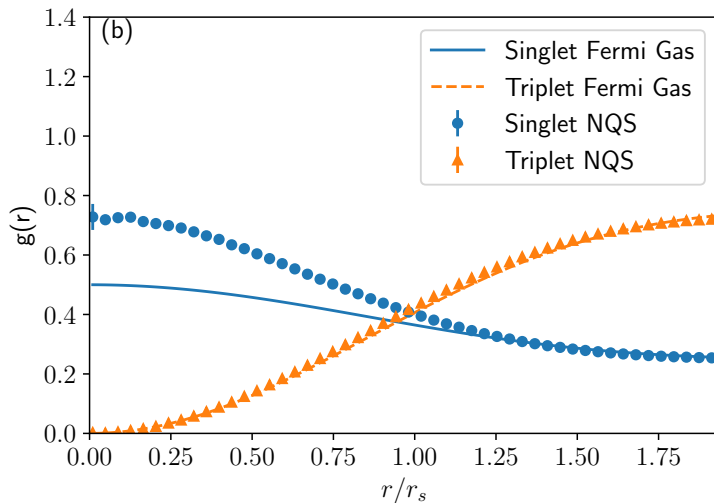
Dilute neutron star matter from neural-network quantum states by Fore et al, Physical Review Research 5, 033062 (2023) at density $\rho = 0.04 \text{ fm}^{-3}$



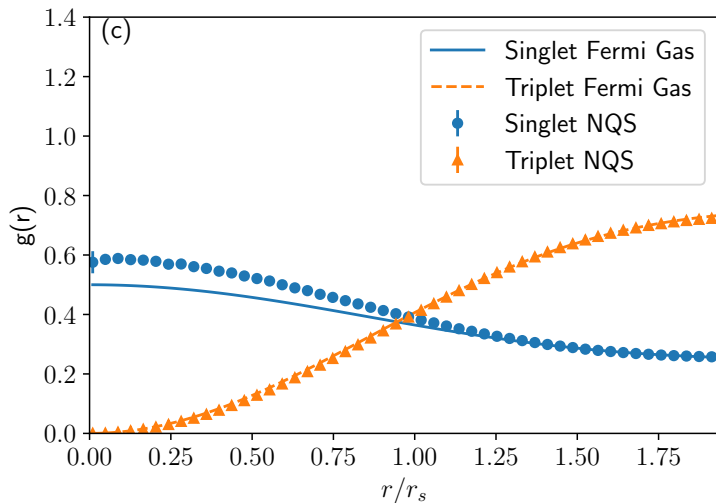
Pairing and Spin-singlet and triplet two-body distribution functions at $\rho = 0.01 \text{ fm}^{-3}$



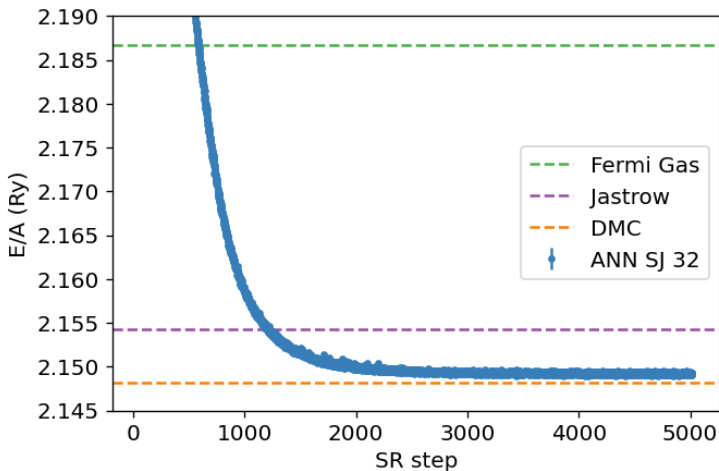
Pairing and Spin-singlet and triplet two-body distribution functions at $\rho = 0.04 \text{ fm}^{-3}$



Pairing and Spin-singlet and triplet two-body distribution functions at $\rho = 0.08 \text{ fm}^{-3}$



The electron gas in three dimensions with $N = 14$ electrons (Wigner-Seitz radius $r_s = 2$ a.u.), Gabriel Pescia, Jane Kim et al. arXiv.2305.07240,



Efficient solutions of fermionic systems using artificial neural networks, Nordhagen et al, Frontiers in Physics 11, 2023

The Hamiltonian of the quantum dot is given by

$$\hat{H} = \hat{H}_0 + \hat{V},$$

where \hat{H}_0 is the many-body HO Hamiltonian, and \hat{V} is the inter-electron Coulomb interactions. In dimensionless units,

$$\hat{V} = \sum_{i < j}^N \frac{1}{r_{ij}},$$

with $r_{ij} = \sqrt{r_i^2 + r_j^2}$.

Separable Hamiltonian with the relative motion part ($r_{ij} = r$)

$$\hat{H}_r = -\nabla_r^2 + \frac{1}{4}\omega^2 r^2 + \frac{1}{r},$$

Analytical solutions in two and three dimensions (M. Taut 1993 and 1994).

Why Boltzmann machines?

What is known as restricted Boltzmann Machines (RBM) have received a lot of attention lately. One of the major reasons is that they can be stacked layer-wise to build deep neural networks that capture complicated statistics.

The original RBMs had just one visible layer and a hidden layer, but recently so-called Gaussian-binary RBMs have gained quite some popularity in imaging since they are capable of modeling continuous data that are common to natural images.

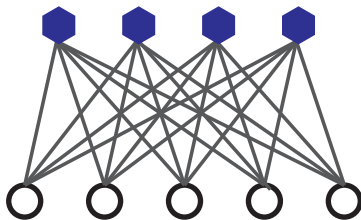
Furthermore, they have been used to solve complicated quantum mechanical many-particle problems or classical statistical physics problems like the Ising and Potts classes of models.

The structure of the RBM network

Hidden Layer

Interactions

Visible Layer



$$b_{\mu}(h_{\mu})$$

$$W_{i\mu}v_ih_{\mu}$$

$$a_i(v_i)$$

The network

The network layers:

1. A function x that represents the visible layer, a vector of M elements (nodes). This layer represents both what the RBM might be given as training input, and what we want it to be able to reconstruct. This might for example be the pixels of an image, the spin values of the Ising model, or coefficients representing speech.
2. The function h represents the hidden, or latent, layer. A vector of N elements (nodes). Also called "feature detectors".

Goals

The goal of the hidden layer is to increase the model's expressive power. We encode complex interactions between visible variables by introducing additional, hidden variables that interact with visible degrees of freedom in a simple manner, yet still reproduce the complex correlations between visible degrees in the data once marginalized over (integrated out).

The network parameters, to be optimized/learned:

1. a represents the visible bias, a vector of same length as x .
2. b represents the hidden bias, a vector of same length as h .
3. W represents the interaction weights, a matrix of size $M \times N$.

Joint distribution

The restricted Boltzmann machine is described by a Boltzmann distribution

$$P_{rbm}(\mathbf{x}, \mathbf{h}) = \frac{1}{Z} e^{-\frac{1}{T_0} E(\mathbf{x}, \mathbf{h})}, \quad (3)$$

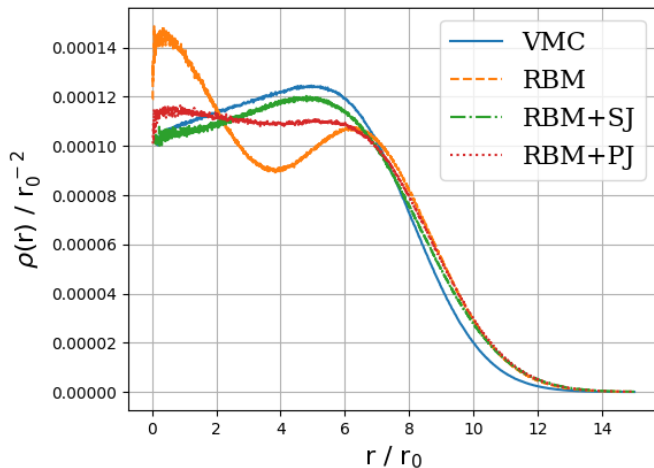
where Z is the normalization constant or partition function, defined as

$$Z = \int \int e^{-\frac{1}{T_0} E(\mathbf{x}, \mathbf{h})} d\mathbf{x} d\mathbf{h}. \quad (4)$$

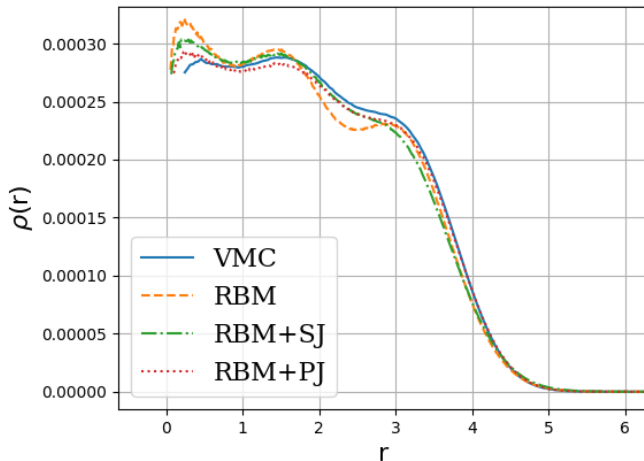
It is common to ignore T_0 by setting it to one.

Quantum dots and Boltzmann machines, onebody densities

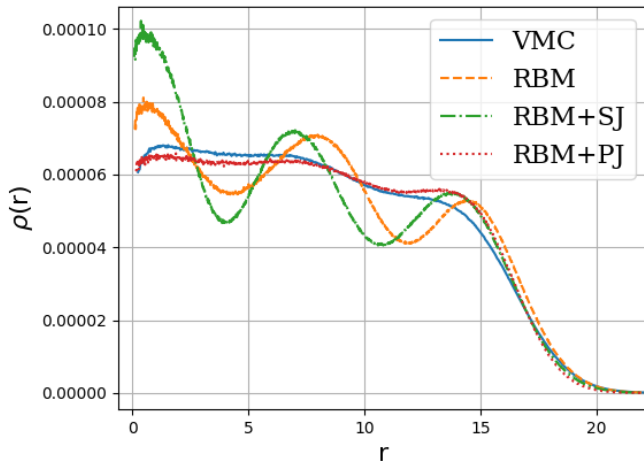
$N = 6$, $\hbar\omega = 0.1$ a.u.



Onebody densities $N = 30$, $\hbar\omega = 1.0$ a.u.



Onebody densities $N = 30$, $\hbar\omega = 0.1$ a.u.



Extrapolations and model interpretability

When you hear phrases like **predictions and estimations** and **correlations and causations**, what do you think of? Maybe you think of the difference between classifying new data points and generating new data points. Or perhaps you consider that correlations represent some kind of symmetric statements like if A is correlated with B , then B is correlated with A . Causation on the other hand is directional, that is if A causes B , B does not necessarily cause A .

Physics based statistical learning and data analysis

The above concepts are in some sense the difference between **old-fashioned** machine learning and statistics and Bayesian learning. In machine learning and prediction based tasks, we are often interested in developing algorithms that are capable of learning patterns from given data in an automated fashion, and then using these learned patterns to make predictions or assessments of newly given data. In many cases, our primary concern is the quality of the predictions or assessments, and we are less concerned about the underlying patterns that were learned in order to make these predictions.

Physics based statistical learning points however to approaches that give us both predictions and correlations as well as being able to produce error estimates and understand causations. This leads us to the very interesting field of Bayesian statistics.

Bayes' Theorem

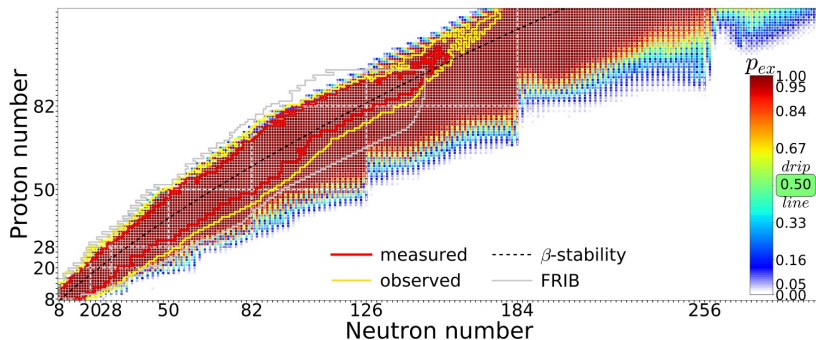
Bayes' theorem

$$p(X|Y) = \frac{p(X, Y)}{\sum_{i=0}^{n-1} p(Y|X = x_i)p(x_i)} = \frac{p(Y|X)p(X)}{\sum_{i=0}^{n-1} p(Y|X = x_i)p(x_i)}.$$

The quantity $p(Y|X)$ on the right-hand side of the theorem is evaluated for the observed data Y and can be viewed as a function of the parameter space represented by X . This function is not necessarily normalized and is normally called the likelihood function. The function $p(X)$ on the right hand side is called the prior while the function on the left hand side is called the posterior probability. The denominator on the right hand side serves as a normalization factor for the posterior distribution.

Quantified limits of the nuclear landscape

Predictions made with eleven global mass model and Bayesian model averaging



Observations (or conclusions if you prefer)

- ▶ Need for AI/Machine Learning in physics, lots of ongoing activities
- ▶ To solve many complex problems and facilitate discoveries, multidisciplinary efforts are required involving scientists in physics, statistics, computational science, applied math and other fields.
- ▶ There is a need for focused AI/ML learning efforts that will benefit accelerator science and experimental and theoretical programs

More observations

- ▶ How do we develop insights, competences, knowledge in statistical learning that can advance a given field?
 - ▶ For example: Can we use ML to find out which correlations are relevant and thereby diminish the dimensionality problem in standard many-body theories?
 - ▶ Can we use AI/ML in detector analysis, accelerator design, analysis of experimental data and more?
 - ▶ Can we use AI/ML to carry out reliable extrapolations by using current experimental knowledge and current theoretical models?
- ▶ The community needs to invest in relevant educational efforts and training of scientists with knowledge in AI/ML. These are great challenges to the CS and DS communities
- ▶ Quantum computing and quantum machine learning not discussed here
- ▶ Most likely tons of things I have forgotten

Possible start to raise awareness about ML in your own field

- ▶ Make an ML challenge in your own field a la [Learning to discover: the Higgs boson machine learning challenge](#).
Alternatively go to kaggle.com at <https://www.kaggle.com/c/higgs-boson>
- ▶ HEP@CERN and HEP in general have made significant impacts in the field of machine learning and AI. Something to learn from