FYS-STK5429 — Optimizing PINN Architectures for the Many-Body Schrödinger Equation

Aleksander N. Sekkelsten[1]

[1]Faculty of Mathematics and Natural Sciences, University of Oslo

Abstract

Accurately solving the Schrödinger equation for interacting many-body systems remains a formidable challenge due to the exponential growth of the Hilbert space with particle number. Here, I present a physics-informed neural network (PINN) approach that augments a Slater-determinant baseline with a neural-network–modeled correlation factor, efficiently capturing interparticle correlations. I investigate two-dimensional quantum dots with Coulomb interactions and one-dimensional spinless fermions with Gaussian interaction, training by direct minimization of the Schrödinger-residual loss without recourse to variational Monte Carlo. The PINN–Slater ansatz reproduces and improves, even with small networks, the most accurate benchmark calculations to date. The results demonstrate that carefully chosen activation functions, initialization schemes, and network depths yield rapid convergence even in the presence of singular or cusp-inducing interactions, offering a scalable and generalizable pathway for quantum-system simulations that blends physical constraints with machine-learning flexibility.

1 Introduction

Understanding ground-state properties of interacting fermionic systems is central to quantum chemistry, condensed-matter physics, and nanoscience. Conventional approaches—exact diagonalization, perturbation theory, and variational Monte Carlo—either scale exponentially with particle number or converge slowly in strongly correlated regimes. Neural-network wave-function ansätze promise to alleviate these bottlenecks by using high-capacity function approximators and automatic differentiation.

In particular, variational Monte Carlo (VMC) methods such as FermiNet Pfau et al. (2020) and PauliNet Hermann et al. (2020) optimize neural wavefunctions against stochastic local-energy estimates, achieving excellent accuracy but demanding large Monte Carlo samples that grow costly for larger systems. Physics-informed neural networks (PINNs) instead enforce the Schrödinger equation residual on collocation points, avoiding sampling noise but lacking built-in antisymmetry. To date, no work has combined PINNs with a Slater-determinant baseline, and there is no consensus on choosing activations, initializations, or network depth to ensure stable training—especially when strong Coulomb singularities or high interaction strengths are present. This project fills that gap by systematically benchmarking Slater-PINN models on one-dimensional Gaussian-interacting fermions and two-dimensional quantum dots, extracting the key design principles that yield robust convergence and chemical-accuracy energies.

Our approach enforces exact antisymmetry via a Slater determinant and models residual correlation energy with a deep feed-forward network. Rather than sampling via Monte Carlo, I minimize a loss function built from the local-energy residual of the Schrödinger equation, ensuring both accuracy and differentiability. The contributions are twofold:

1. **One-Dimensional Spinless Fermions.** Anchored by the noninteracting Slater determinant, my PINN captures Gaussian-interaction correlations for up to four particles at configuration-interaction–level precision.

2. **Two-Dimensional Quantum Dots.** For electrons in a 2D harmonic trap with Coulomb repulsion, I demonstrate that a small PINN (two hidden layers, 200 neurons each) can match diffusion Monte Carlo (DMC) results to within 10^{-5}.

Key design principles emerge: (i) smooth activation functions mitigate vanishing or exploding gradients in singular potentials; (ii) Xavier initialization stabilizes gradient flow across depths; and (iii) even small networks suffice when properly conditioned. Together, these guidelines provide a robust, scalable platform for neural-network–based quantum-many-body solvers.

The remainder of this paper is structured as follows: Section 2 formulates the PINN–Slater ansatz and training procedure; Section 3 presents results for two-dimensional dot, and one-dimensional fermion systems; Section 4 discusses design implications and future directions; and Section 5 concludes. In addition, note at the outset that a complete account of hyper-parameters, network topologies, and training protocols is deferred to the closing Implementation Details section. In the same spirit of readability, the full bibliography is collected at the very end under References, so that the narrative development of ideas remains uninterrupted.

2 Method

2.1 Quantum systems

Quantum systems are best described by the differential equation known as **Schrödinger's Equation**, which governs the evolution and stationary states of a quantum system. In its time-independent form, the equation reads:

$$\hat{H}\psi(\mathbf{r}) = E\psi(\mathbf{r}), \tag{1}$$

where $\hat{H}$ is the Hamiltonian operator, $\psi(\mathbf{r})$ is the wavefunction, and E represents the energy eigenvalues. The Hamiltonian consists of the kinetic and potential energy terms:

$$\hat{H} = -\frac{\hbar^2}{2m}\nabla^2 + V(\mathbf{r}). \tag{2}$$

$V(\mathbf{r})$ consists of both a surrounding potential and interaction terms. In this project, the only potential used is the *Quantum harmonic oscillator* (QHO).

2.1.1 Quantum Harmonic Oscillator (QHO)

For a single particle in a harmonic potential $V(x) = \frac{1}{2}m\omega^2x^2$, the Schrödinger equation is:

$$\left(-\frac{\hbar^2}{2m}\frac{d^2}{dx^2} + \frac{1}{2}m\omega^2x^2\right)\psi_n(x) = E_n\psi_n(x). \tag{3}$$

The eigenfunctions and eigenvalues are:

$$\psi_n(x) = \frac{1}{\sqrt{2^n n!}}\left(\frac{m\omega}{\pi\hbar}\right)^{\frac{1}{4}} H_n\left(\sqrt{\frac{m\omega}{\hbar}}x\right)e^{-\frac{m\omega x^2}{2\hbar}}, \tag{4}$$

$$E_n = \left(n + \frac{1}{2}\right)\hbar\omega, \quad n = 0, 1, 2, \ldots \tag{5}$$

where $H_n(x)$ are the Hermite polynomials. For many bodies, if non-interacting, each electron will simply occupy the lowest accessible energy-states (satisfying Pauli's exclusion-Principle). With interactions present, the ground state often however requires a highly complex mixing of multiple higher order states in order to minimize the strong interactions. It is also worth noting that when ω becomes smaller, the QHO potential becomes less relevant, and the kinetic energy, which proortional to ω vanishes quickly. Therefore, the interactions becomes more important in these cases.

2.1.2 Many-Body Interacting Quantum Harmonic Oscillator

For N interacting quantum harmonic oscillators, the Hamiltonian is:

$$\hat{H} = \sum_{i=1}^{N}\left[-\frac{\hbar^2}{2m}\frac{d^2}{dx_i^2} + \frac{1}{2}m\omega^2x_i^2\right] + V_{\text{int}}(x_i, x_j). \tag{6}$$

$V_{\text{int}}(x_i, x_j)$ is an interaction term which depends on the pairwise distance between particles. This paper investigates two interaction types. The first, and more simple interaction term used in this work is a gausian type and constitutes *Trapped Spinless Fermions*:

$$V_{\text{int}}(x_i, x_j) = \frac{V}{\sigma\sqrt{2\pi}}\sum_{i<j}^{N}\exp\left[-\frac{(x_i - x_j)^2}{2\sigma^2}\right]. \tag{7}$$

This interaction represents a Gaussian potential between particles, where V determines the interaction strength and σ controls the range of interaction. The next interaction-form is more realistic, and constitutes *Quantum Dots*:

$$V_{\text{int}}(x_i, x_j) = \sum_{i<j}^{N} \frac{1}{|\mathbf{x}_i - \mathbf{x}_j|}. \tag{8}$$

Solving such systems requires numerical tools. In this project, I will use a *Physics Informed Neural Network* to do this. Another method, essentially the analytical solution, is the *Full Configuration Interaction (FCI)* aproach.

2.2 Full Configuration Interaction (FCI)

Full Configuration Interaction (FCI) is a method that provides an exact solution to the Schrödinger equation within a finite basis set by considering all possible electron configurations. In the FCI approach, every possible arrangement of electrons among the available orbitals is included, which ensures that all correlation effects are accounted for. Although FCI is computationally demanding—its cost increases exponentially with the number of particles—it serves as an excellent benchmark for the accuracy of quantum mechanical calculations. In this work, FCI results for the gausian systems are used as a reference to evaluate the performance of my neural network-based approach. In practice, since there are effictevly infinitely many excited states, one has to trucate the basis to an upper bound, this is often referedn to as *Configuration interaction* (CI).

2.3 Feed-Forward Neural Networks and Activation Functions

A **feed-forward neural network** (FNN) is a function approximator composed of multiple layers of interconnected neurons, where information flows in one direction, from input to output, without cycles or feedback loops. A fully connected FNN with L layers can be expressed as:

$$\mathbf{h}^{(l+1)} = \sigma\left(\mathbf{W}^{(l)}\mathbf{h}^{(l)} + \mathbf{b}^{(l)}\right), \tag{9}$$

where: - $\mathbf{h}^{(l)}$ is the activation vector at layer l, - $\mathbf{W}^{(l)}$ is the weight matrix connecting layers l and $l+1$, - $\mathbf{b}^{(l)}$ is the bias vector, - $\sigma(x)$ is an activation function.

The activation function introduces non-linearity, allowing neural networks to approximate complex functions. Common activation functions include:

- **Sigmoid:** $\sigma(x) = \frac{1}{1+e^{-x}}$
- **Tanh:** $\tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$
- **ReLU:** $\text{ReLU}(x) = \max(0, x)$
- **Swish:** $\text{Swish}(x) = x \cdot \sigma(x)$
- **Mish:** $\text{Mish}(x) = x \tanh(\ln(1 + e^{x}))$
- **GELU:** $\text{GELU}(x) = x\Phi(x)$, where $\Phi(x)$ is the standard Gaussian cumulative distribution function.

Because my loss contains second-order Laplacians, any first, second or third-order derivative blowup or vanishing in can severely disturb training. I will show that GELU/Mish/Swish mitigate this effect; see 3.4. Figure 1 depicts each activation function alongside its first, second, and third derivatives—in what follows, I will argue that the third derivative of Tanh exacerbates training instability in the Schrödinger-residual loss.

2.4 Physics-Informed Neural Networks (PINNs)

Physics-informed neural networks (PINNs) provide a mesh-free framework for solving partial differential equations (PDEs) by embedding the governing equations directly into the loss function of a neural network. Rather than training on input–output data alone, PINNs enforce the physics—via automatic differentiation—so that the surrogate model $u_\theta(\mathbf{x})$ approximately satisfies

$$\mathcal{N}[u_\theta(\mathbf{x})] = 0, \quad \mathbf{x} \in \Omega,$$

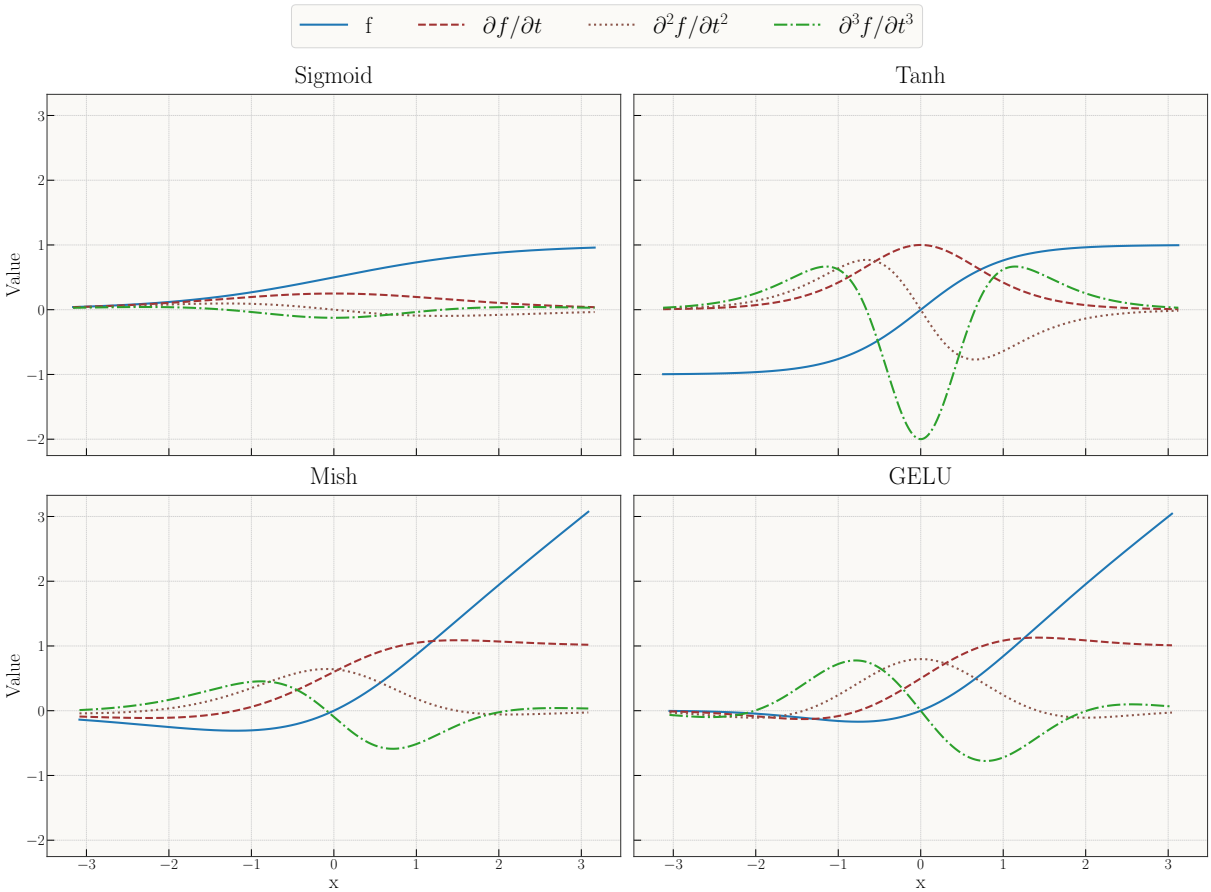


Figure 1: Combined view of four activation functions (GELU, Mish, tanh, Sigmoid) together with their first, second and third derivatives.

subject to the boundary and initial conditions

$$\mathcal{B}[u_\theta(\mathbf{x})] = 0, \quad \mathbf{x} \in \partial\Omega.$$

To implement a PINN, one typically samples:

- *Collocation points* $\{\mathbf{x}_i^r\}_{i=1}^{N_r}$ from the domain Ω, where the PDE residual is enforced.
- *Boundary/initial points* $\{\mathbf{x}_i^b\}_{i=1}^{N_b}$ on $\partial\Omega$ (and $\{\mathbf{x}_i^0\}_{i=1}^{N_0}$ at $t = 0$ for time-dependent problems), where prescribed values or fluxes are enforced.

The total loss function combines three terms:

$$\mathcal{L}(\theta) = \underbrace{\frac{1}{N_r}\sum_{i=1}^{N_r}\left\|\mathcal{N}[u_\theta(\mathbf{x}_i^r)]\right\|^2}_{\mathcal{L}_{\text{PDE}}} + \underbrace{\frac{1}{N_b}\sum_{i=1}^{N_b}\left\|\mathcal{B}[u_\theta(\mathbf{x}_i^b)]\right\|^2}_{\mathcal{L}_{\text{BC}}} + \underbrace{\frac{1}{N_0}\sum_{i=1}^{N_0}\left\|u_\theta(\mathbf{x}_i^0) - u_0(\mathbf{x}_i^0)\right\|^2}_{\mathcal{L}_{\text{IC}}}.$$

By minimizing $\mathcal{L}(\theta)$ with respect to the network parameters θ (e.g., using Adam), the model learns a solution that simultaneously fits any available data and satisfies the differential operator throughout Ω.

2.5 Solving the Quantum System using a Neural Network Ansatz

Constructing a good ansatz, using neural network recquires careful consideration. Most importantly, fermionic symmetry **must** be satsfied, in order for the wavefunction to be physical. Therefore, the ansatz used is as follows:

$$\Psi(\mathbf{x}_1, \ldots, \mathbf{x}_N), =, \det\left[\phi_i(\mathbf{x}j)\right] e^{F\theta(\mathbf{x}_1,\ldots,\mathbf{x}_N)} \tag{10}$$

where:

- $\det\left[\phi_i(\mathbf{x}j)\right]$ is the *Slater determinant* built from the N lowest noninteracting single-particle orbitals $\phi_i(\mathbf{x})$, which enforces the required fermionic antisymmetry.
- $F\theta :, \mathbb{R}^{N\times d} \to \mathbb{R}$ is a feed-forward neural network with parameters θ, trained to encode the many-body *correlation factor*. Forced to be symmetric

We implement symmetry by (1) applying a small sub-network to each particle independently, then averaging; (2) applying another sub-network to each pairwise distance, then summing; and (3) feeding the concatenation of these particle-averaged and pair-summed features into a final network . In that way, F is permutation-invariant by construction.

2.6 Network Architecture and Data Flow

Figure 2 gives a schematic of how particle positions and pairwise distances propagate through the sub-networks to produce the correlation/jastrow factor F_θ.

The network F_θ is realized as three sub-networks:

1. ϕ-network: $\mathbb{R}^d \to \mathbb{R}^{d_L}$, applied independently to each particle position and averaged.
2. ψ-network: $\mathbb{R} \to \mathbb{R}^{d_L}$, applied to interparticle distances $r_{ij} = |\mathbf{x}_i - \mathbf{x}_j|$.
3. ρ-network: $\mathbb{R}^{2d_L} \to \mathbb{R}$, combining the outputs of ϕ and ψ into a scalar correlation contribution.

2.7 Training

We enforce the many-body Schrödinger equation via a physics-informed residual

$$H\Psi = -\tfrac{1}{2}\nabla^2\Psi + V_{\text{tot}}(\mathbf{x})\Psi, \tag{11}$$

and define the pointwise residual

$$R(\mathbf{x}_1, \ldots, \mathbf{x}_N) = H\Psi - E\Psi, \tag{12}$$

where E is the energy (not a trainable parameter in this project, but handed in manually).

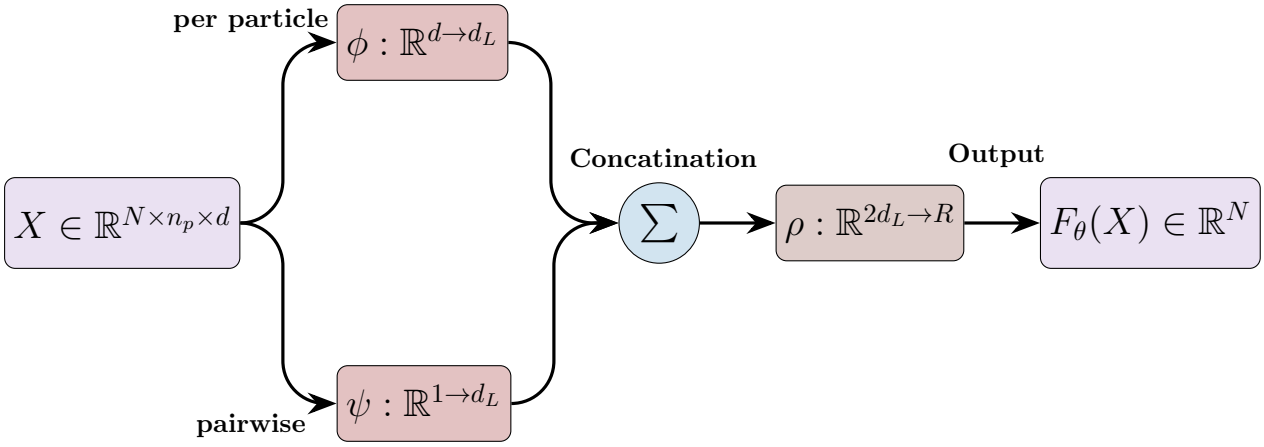


Figure 2: Data flow in the PINN ansatz: batched inputs feed ϕ and ψ symmetrically into a pooling node, then through ρ to produce the Jastrow factor.

The total loss minimized is

$$\mathcal{L}(\theta, \alpha) = \underbrace{\mathbb{E}[R^2]}_{\mathcal{L}\text{PDE}} + \underbrace{\alpha, (|\Psi|^2 - 1)^2}_{\mathcal{L}\text{norm}}, \tag{13}$$

with hyperparameter α balancing residual and normalization penalties. This is always set to 0 for quantum dots, and 0.01 for gausian interacting fermions.

By training (θ, α) to minimize $\mathcal{L}$, I obtain a wave function ansatz that satisfies the Schrödinger equation and remains normalized, leveraging both analytic antisymmetry and neural correlation.

3 Results

The results acquired in this project are two-fold. First, I tested the stability of the neural network by investigating non-interacting systems. Then, I examine the interacting many body systems.

3.1 Non-Interacting Benchmarks

Before examining correlated regimes, I validate every network architecture on two trivial limiting cases in which the interaction operator is present in name only:

(i) **One-dimensional spinless fermions.** Here the Gaussian interaction parameter V is set to zero. For N fermions in a unit-frequency harmonic oscillator, the exact non-interacting ground-state energy is

$$E_0^{1\mathrm{D}} \;=\; \sum_{k=0}^{N-1} \left(k + \tfrac{1}{2}\right) \;=\; \frac{N^2}{2}\,.$$

In particular, for $N = 2, 3, 4$,

$$E_0^{1\mathrm{D}}(N=2) = 2.00000, \quad E_0^{1\mathrm{D}}(N=3) = 4.50000, \quad E_0^{1\mathrm{D}}(N=4) = 8.00000.$$

Although my PINN–Slater models were still trained (with $V = 0$) and produce a small statistical uncertainty in the reported energies, that uncertainty is on the order of 10^{-6} or less—negligible for any practical purpose. Table 1 reports the central PINN estimates and their standard errors (from recursive blocking), confirming that each architecture recovers the exact non-interacting energy to within 10^{-6}.

(ii) **Two-dimensional quantum dot (two electrons).** With Coulomb interactions formally present but set to zero, the analytic ground-state energy is

$$E_0^{2\mathrm{D}} \;=\; 2\,\omega,$$

since each of the two electrons occupies the $(n = 0, m = 0)$ harmonic-oscillator orbital. I choose $\omega = 1.0$, so that $E_0^{2\mathrm{D}} = 2.0$. The PINN reproduces this value to within $\pm 10^{-5}$, as shown in Table 1.

Table 1: Ground-state energies for non-interacting benchmarks. The PINN estimates include the standard error from recursive blocking; in every case, the uncertainty is $\mathcal{O}(10^{-6})$ or smaller.

System	Control parameter	Ground-state energy E_0
1D spinless fermions, $N = 2$	$V = 0$	2.0000000 ($\pm$0.0000003)
1D spinless fermions, $N = 3$	$V = 0$	4.5000001 ($\pm$0.0000004)
1D spinless fermions, $N = 4$	$V = 0$	8.0000002 ($\pm$0.0000005)
2D quantum dot, two electrons	$\omega = 1.0$	2.00001 ($\pm$0.00001)

For the 1D fermions, $E_0^{\mathrm{1D}} = \sum_{k=0}^{N-1}(k + \frac{1}{2}) = \frac{N^2}{2}$.
For the 2D dot, $E_0^{\mathrm{2D}} = 2\,\omega = 1.0$.

3.2 Interacting One-Dimensional Trapped Spinless Fermions

We assess the ability of my PINN–Slater-determinant ansatz to recover the ground-state energy of $N = 2$, 3, and 4 spinless fermions interacting via a Gaussian potential of strength $V = 10$, and 20. In all cases the Slater determinant enforces exact antisymmetry and reproduces the noninteracting limit ($V = 0$), so the neural network need only encode the residual correlation and interaction energy.

3.2.1 Effect of Network Depth

Two Particles ($N = 2$). Figure 3 shows that both two- and three-hidden-layer networks rapidly converge to the configuration-interaction (CI) benchmark for $V = 10, 20$. The single-layer model exhibits a slightly larger deviation at $V = 20$, but its loss continues to decrease through the final epochs, indicating that extended training would further narrow the gap.

Three Particles ($N = 3$). As seen in Figure 4, all network depths—one through three hidden layers—converge stably and match the CI reference across the entire interaction range. Convergence is similar to the two-particle case, but they all start much further from the ground state energy in the first epoch.

Four Particles ($N = 4$). Figure 5 illustrates that while shallow networks (one hidden layer) show slightly noisier convergence, all depths ultimately attain energies within $\sim 10^{-3}$ of the CI values. Deeper architectures exhibit smoother training curves and marginally lower final errors.

3.2.2 Activation Function Comparison

Three Particles ($N = 3$). Figure 6 compares GELU, Mish, Swish, Tanh, and Sigmoid in a two-hidden-layer network. All activations reach the CI energy by 200 epochs, but GELU, Mish, and Swish converge noticeably faster. Sigmoid is the slowest to plateau, exhibiting a more gradual descent.

Four Particles ($N = 4$). As shown in Figure 7, the superiority of modern smooth activations becomes more pronounced. GELU and SiLU maintain consistent, rapid convergence, whereas Sigmoid and Tanh yield larger residual errors and slower training dynamics, underscoring their limitations in capturing strong correlations. A noticable difference now however, is that Mish for both $V = 10, 20$ seems to be unstable, showing sudden spikes in energies consistently.

3.3 Interacting Two-Dimensional Quantum Dots

We now extend my study to two-dimensional quantum dots, a markedly more intricate system due to the singular behavior of the Coulomb interaction as interparticle distances tend to zero. In this setting I examine three harmonic-oscillator confinement strengths, $\omega = 1.0, 0.5$ and $\omega = 0.1$. As ω decreases, the potential well broadens and one might intuitively expect weaker particle correlations owing to the increased average separation. Contrary to this simplistic expectation, however, correlation effects become progressively more pronounced at lower ω, underscoring the necessity of an accurate many-body treatment in the shallow-well regime.

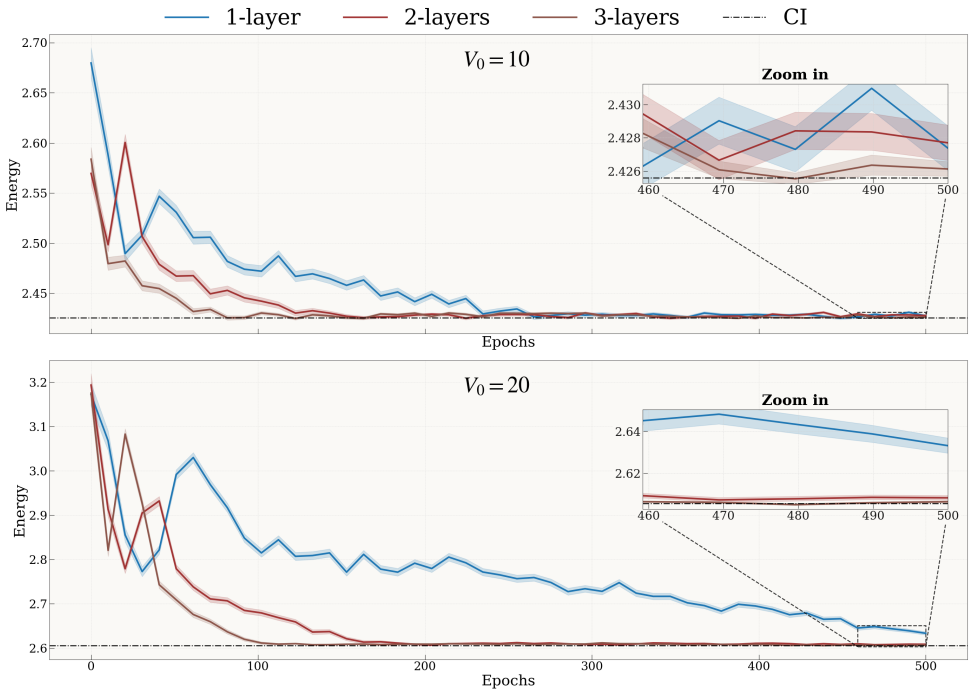


Figure 3: Ground-state energy vs. epoch for two fermions under Gaussian interactions $V = 0, 10, 20$, comparing networks with one, two, and three hidden layers.

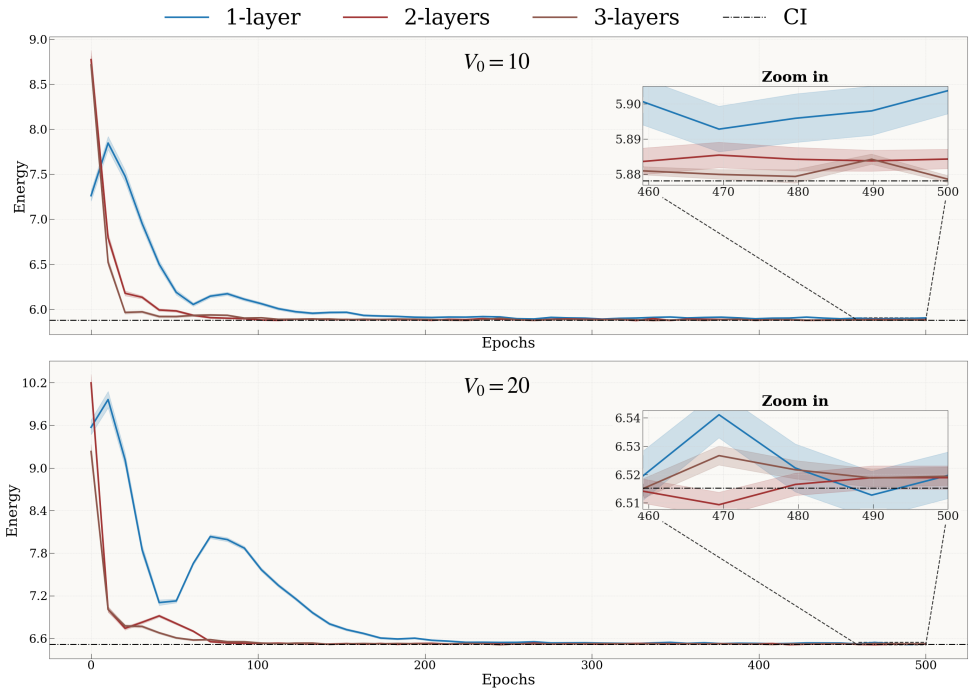


Figure 4: Ground-state energy vs. epoch for three fermions under Gaussian interactions $V = 0, 10, 20$, comparing networks with one, two, and three hidden layers.

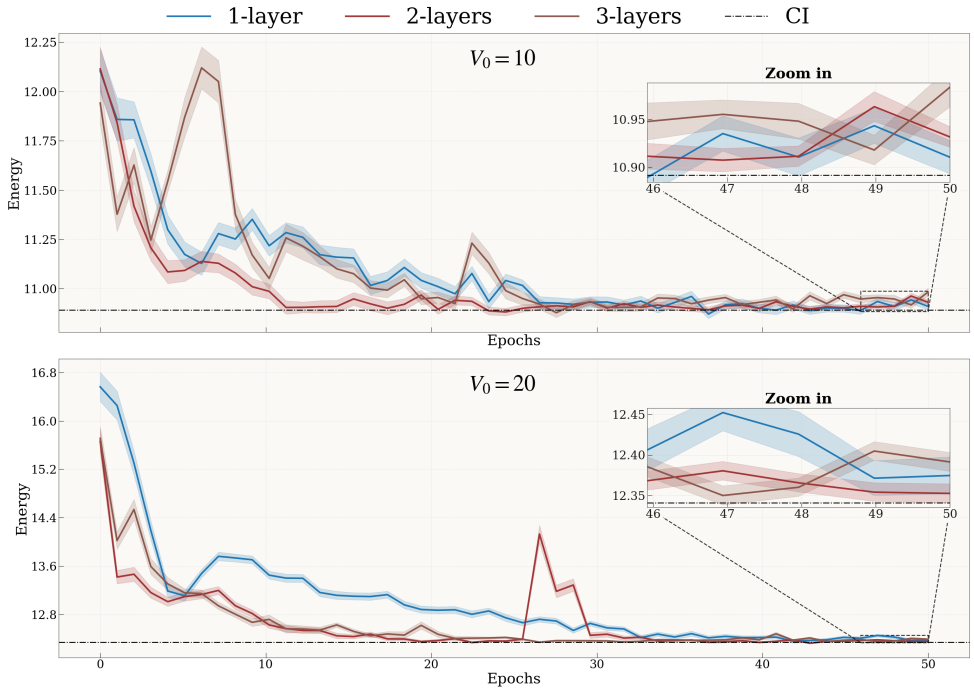


Figure 5: Ground-state energy vs. epoch for four fermions under Gaussian interactions $V = 0, 10, 20$, comparing networks with one, two, and three hidden layers.

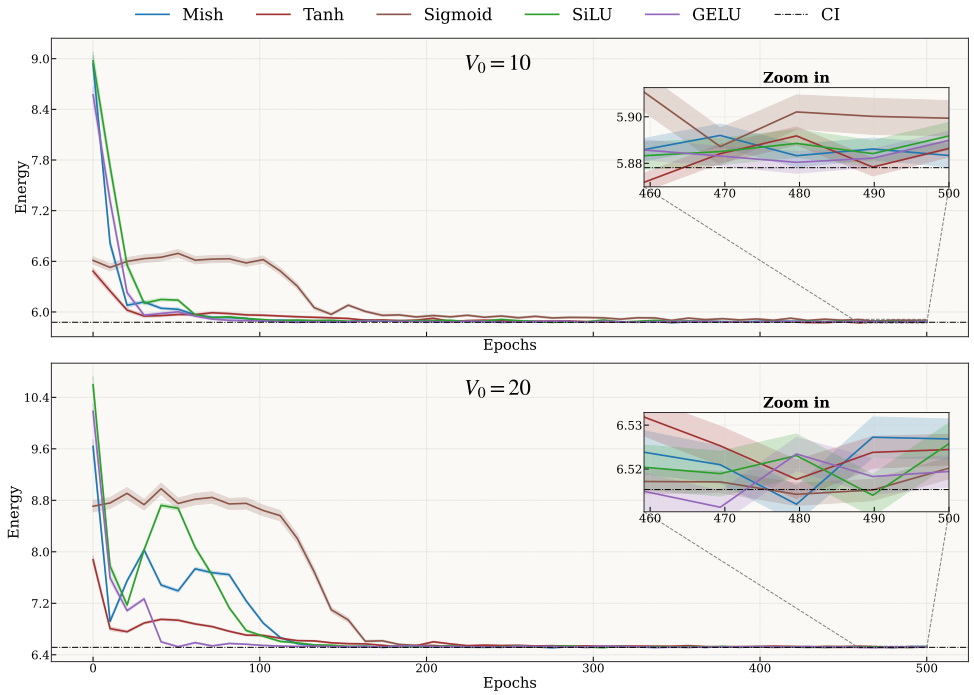


Figure 6: Energy convergence for $N = 3$, $V = 10, 20$, two hidden layers, comparing activation functions.

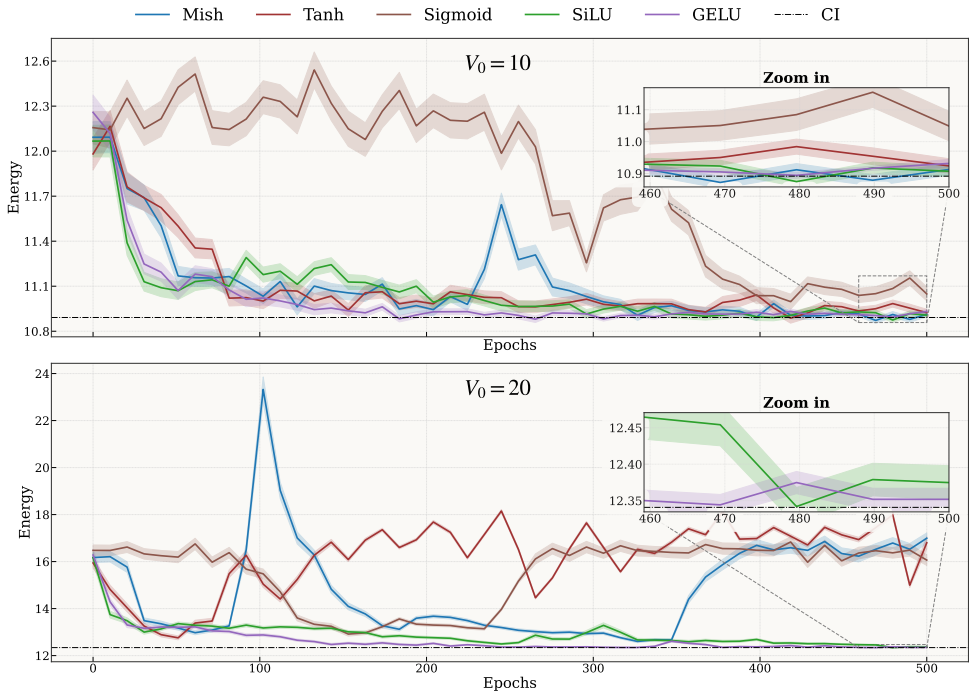


Figure 7: Energy convergence for $N = 4$, $V = 10, 20$, two hidden layers, comparing activation functions.

3.3.1 Resulting energies

Before analyzing convergence, I first report the final energies obtained by the PINN for the three test frequencies, $\omega = 1.0, 0.5$ and $\omega = 0.1$. Table 2 summarizes the analytical (or near-analytical) reference values alongside the best PINN results, including uncertainties.

Table 2: Reference and PINN ground-state energies for two oscillator frequencies.

ω	Reference energy	PINN energy
0.1	0.44079 ($\pm$0.00001)[a]	0.4407936 ($\pm$0.0000016)
0.5	1.65977 ($\pm$0.00001)[a]	1.65988 ($\pm$0.00009)
1.0	3.00000	2.9998 ($\pm$0.0001)

[a] Estimated via diffusion Monte Carlo (DMC).

Clearly, the model used is able to find the ground state energy comfortably, even improving the accuracy of the already known DMC calculations to date. The influence of architectural and initialization choices on the convergence behavior is discussed in Section 3.3.2.

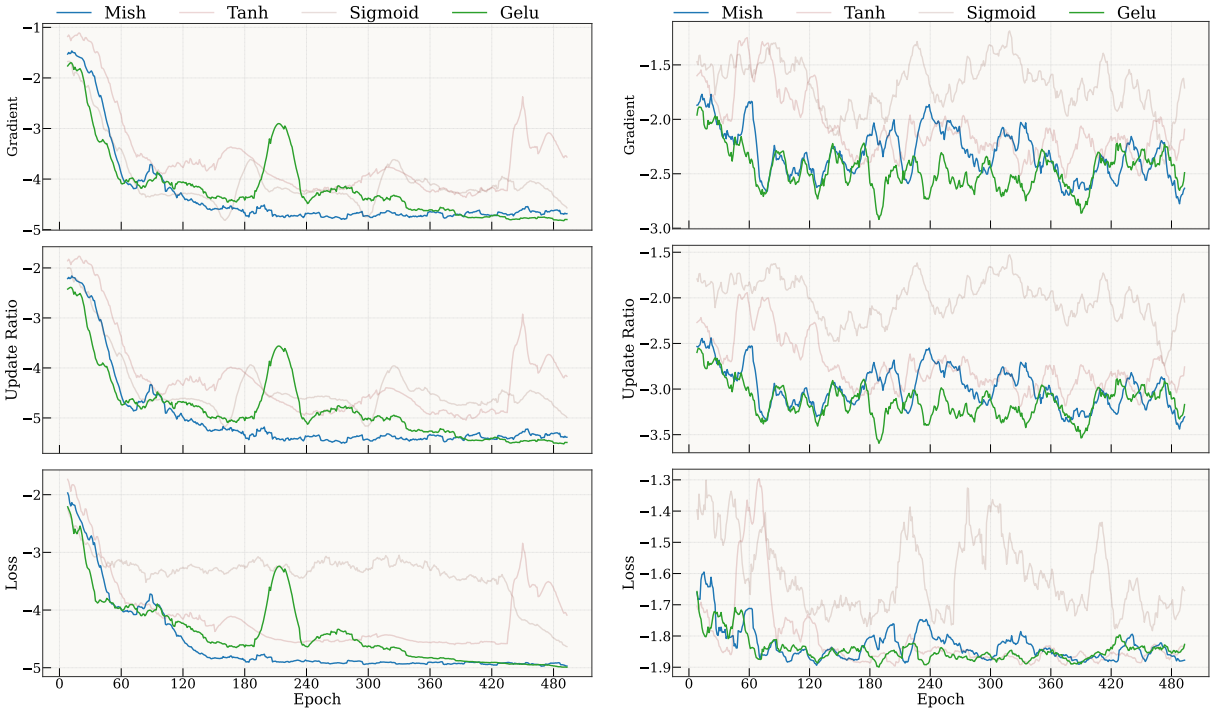
3.3.2 Convergence Analysis

In order to keep the length of this report modest, I will share convergence analysis for $\omega = 0.1, 1.0$ only. Convergence is measured with respect to the Hartree–Fock reference energy (improving the HF energy). I monitor three diagnostic metrics:

- **Squared gradient norm**, $\|\nabla_\theta \mathcal{L}\|_2^2$, which quantifies the magnitude of the back-propagated parameter gradients for the physics-informed loss $\mathcal{L} = \langle (H\Psi - E\,\Psi)^2 \rangle$. A small value indicates vanishing gradients; a large, erratic value signals potential explosion.
- **Parameter-update ratio**, $\|\Delta\theta\|_2 / \|\theta\|_2$, where $\Delta\theta$ is the change in network weights at each step. This ratio measures how aggressively the optimizer moves in parameter space.
- **Loss**, $\mathbb{E}[(H\Psi - E\,\Psi)^2]$, which drives the network toward the ground state. Drop in loss indicates decreasing residual error.

Figure 8 illustrates, for each activation function, the per-epoch evolution of (a) the squared gradient norm, (b) the parameter-update ratio, and (c) the loss. Solid traces denote models that successfully converge; translucent traces indicate those that do not. I observe that GELU and Mish yield stable convergence, whereas Tanh and Sigmoid fail to reach the Hartree–Fock baseline.

- $\omega = 0.1$**:** GELU-based models converge after approximately 300 epochs, exhibiting a steady loss decrease. Mish also converges, but its loss plateaus earlier, indicating a slower descent. In both cases, the loss remains only slightly higher than the corresponding gradient norm and update-ratio magnitudes. By contrast, Sigmoid's loss stays 1–2 orders of magnitude above all others while its gradient norms and update ratios remain comparable to the rest—signaling a vanishing-gradient regime. Tanh, although its loss magnitude is similar to GELU/Mish, shows a gradient-to-update ratio that is 0.5–1 order of magnitude larger than its loss, indicating unstable training.
- $\omega = 1.0$**:** Both GELU and Mish stabilize by about 120 epochs and reach the analytical energy of 3.0, with minimal further improvement. Because the system is less correlated, the trends are less pronounced than at $\omega = 0.1$. Sigmoid fails to train effectively and exhibits a highly oscillatory loss curve. Tanh achieves a relatively low loss but still maintains a higher gradient-to-update ratio than the convergent models, indicating residual instability.



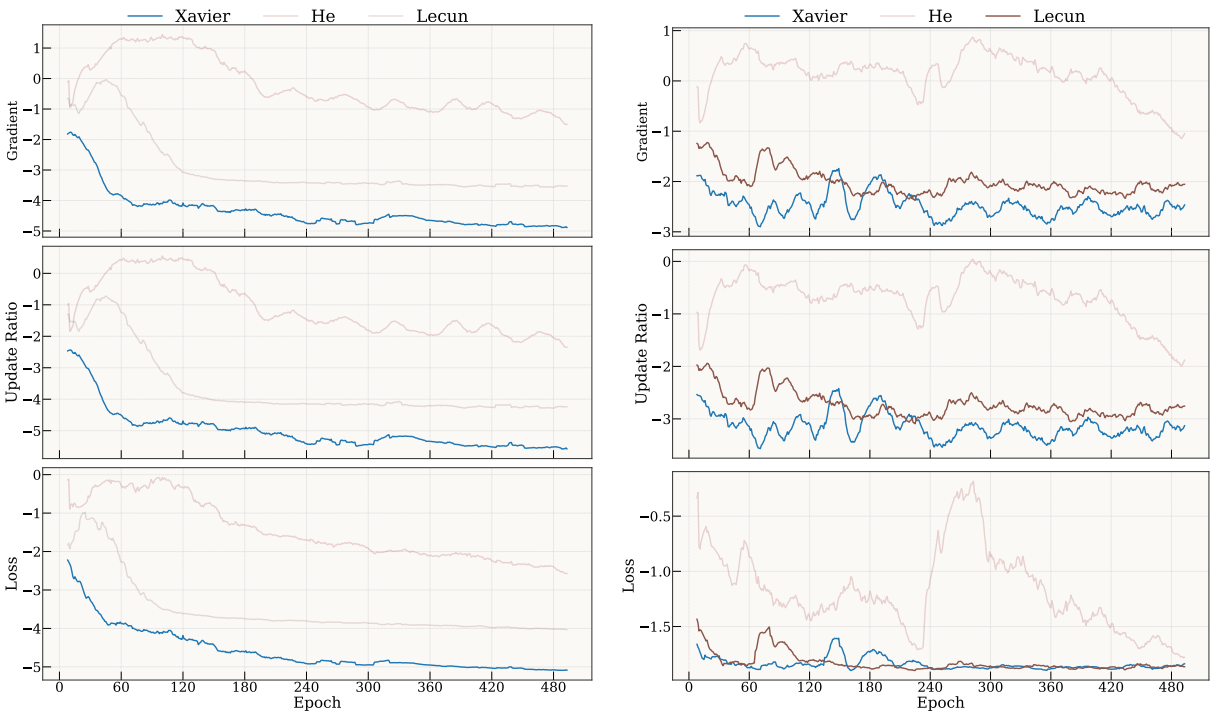
(a) Convergence traces for $\omega = 0.1$.

(b) Convergence traces for $\omega = 1.0$.

Figure 8: Gradient flow and loss evolution for different activation functions in PINNs. (Logarithmic y-axis)

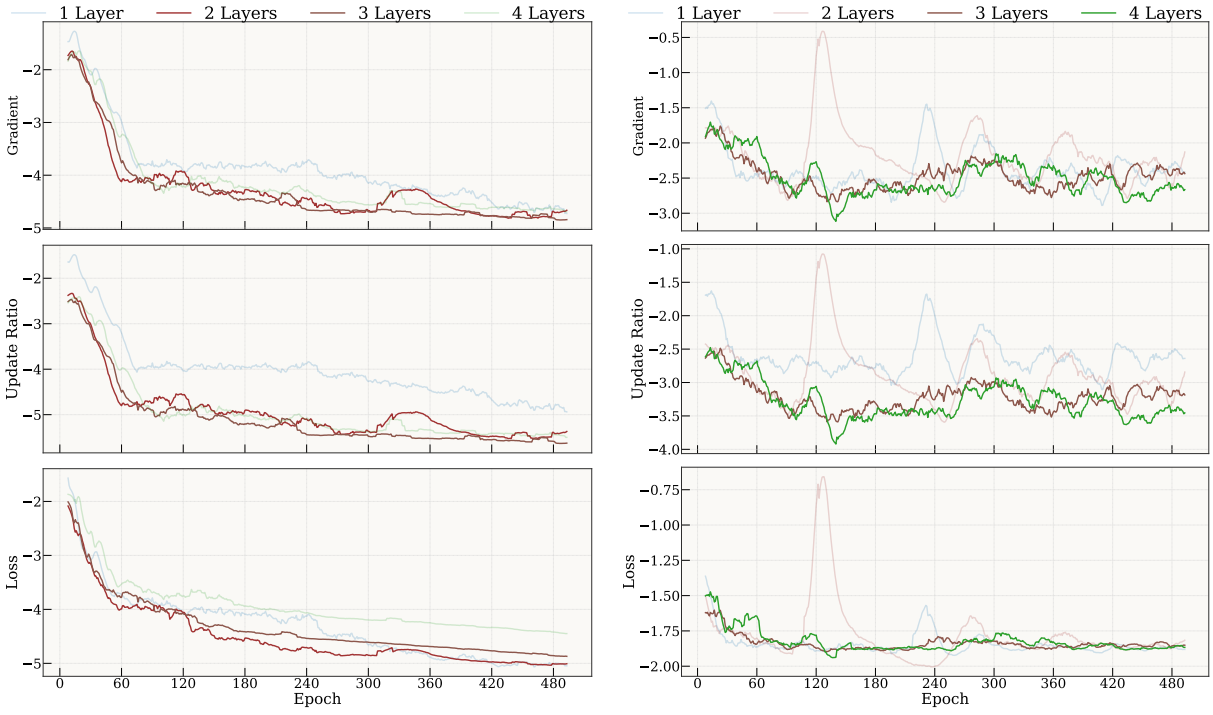
Figure 9 examines the impact of weight-initialization schemes on convergence, using GELU throughout. In both confinement regimes, Xavier initialization consistently achieves lower loss than alternative strategies, an effect that is particularly pronounced for $\omega = 0.1$. The problem with He initilization is obvious, it's loss value is lower than the update ratio or gradients, being almost 1-2 orders of magnitude higher. This is very unstable training, and a sign of exploding gradients.

Finally, Figure 10 explores the role of network depth (1–4 hidden layers) under GELU activation and Xavier initialization. For $\omega = 1.0$, only networks with three or more layers converge—shallow architectures exhibit erratic training dynamics and fail to reach the analytic benchmark. Under $\omega = 0.1$, models with two to four layers display a near-linear decrease in loss, with stable gradient norms and update ratios; the single-layer network again proves insufficiently expressive to capture the enhanced correlation effects. Notably, the four-layer model lies on the cusp of convergence, only being barely above the HF energy at the last epoch.



(a) $\omega = 0.1$. (b) $\omega = 1.0$.

Figure 9: Effect of initialization scheme on PINN convergence (GELU activation). Logarithmic y-axis



(a) $\omega = 0.1$. (b) $\omega = 1.0$.

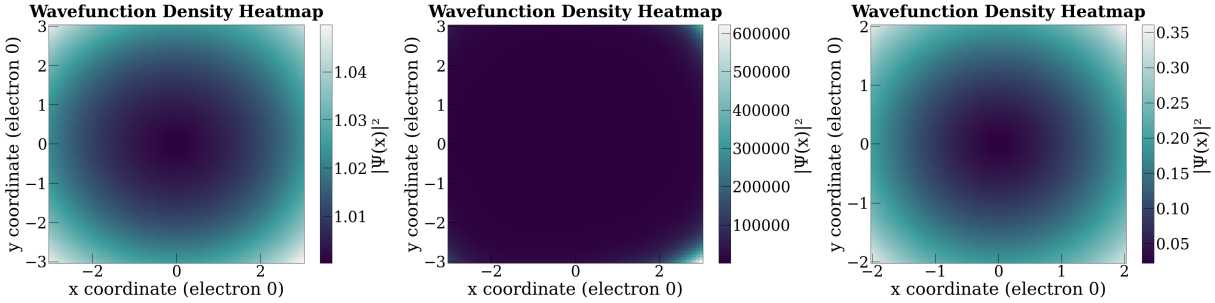
Figure 10: Influence of network depth on gradient flow and loss convergence (GELU + Xavier). (Logarithmic y-axis)

3.4 Implications of Weight Initialisation

Convergence behaviour varies markedly across runs that differ only in the initial parameter draw. The key lies in the *initial output landscape* of the network: the preliminary wave-function defined by the weights fixes the scale of the first gradients and therefore the optimiser's trajectory.

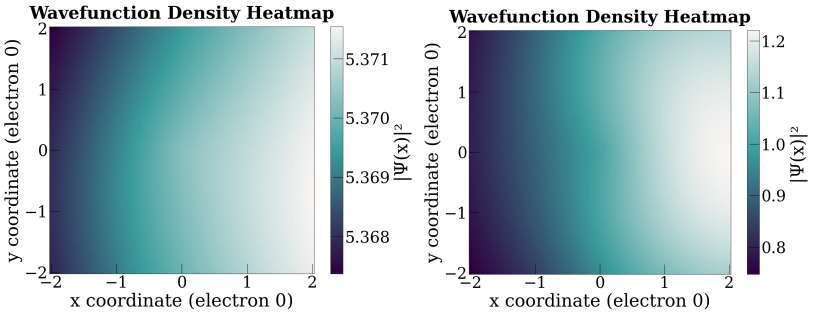
Figure 11 contrasts two freshly initialised networks with a fully converged reference. Xavier initialisation yields an almost spatially uniform amplitude that very closely resembles the trained model, so only modest updates are required. He initialisation, by contrast, produces amplitudes roughly 5 orders of magnitude larger; the optimiser must first rescale these outputs, a step that frequently destabilises training.

Activation choice modulates the same effect (Figure 12). With Xavier weights the SIGMOID output saturates the space (left-right), and spanning only the range 5.368–5.371 in this run; the resulting vanishing gradients impede learning. The TANH output is also saturated, but higher-order derivatives of TANH are large (cf. Figure 1), amplifying curvature in the loss surface and again threatening stability.



(a) Untrained – Xavier (b) Untrained – He (c) Trained wave-function

Figure 11: Comparison of exponentiated PINN outputs, both untrained (left and middle) and trained wavefunctions (right). All with 2 layers, 64 hidden nodes and GELU activation



(a) Untrained – Sigmoid activation (b) Untrained – Tanh activation

Figure 12: Comparison of exponentiated PINN outputs, both untrained with xavier activation. (Logarithmic y-axis)

3.4.1 Stability diagnostics across input variance

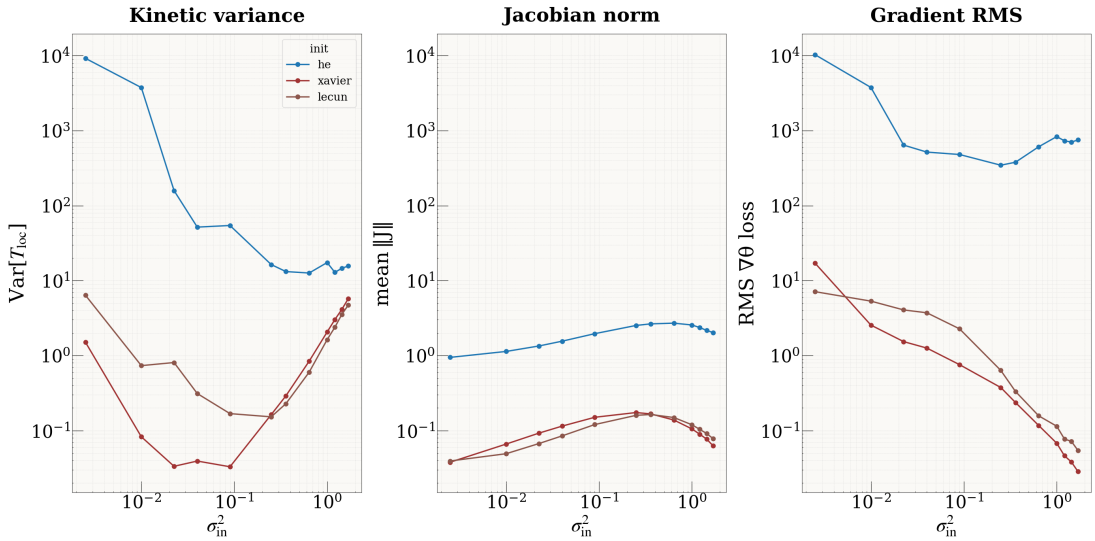
Vanishing or exploding gradients are often a hinderence on performace of neural networks. Figure 13 shows three complementary metrics that probe the gradient stability of my physics-informed networks as the variance of the Gaussian input ensemble (σ_{in}^2 on the horizontal log–axis) is increased. Specifically, I show the variance of the kinetic energy term, 1´st order derivative term (jacobian), and the variance of the gradient through backpropagation.

- **Kinetic variance** $\text{Var}[T_{\text{loc}}]$ (left column) measures fluctuations of the kinetic-energy operator $T_{\text{loc}} = -\frac{1}{2}\nabla^2\psi/\psi$. A plateau indicates well-controlled second spatial derivatives, while a rising curve signals large Laplacian spikes.

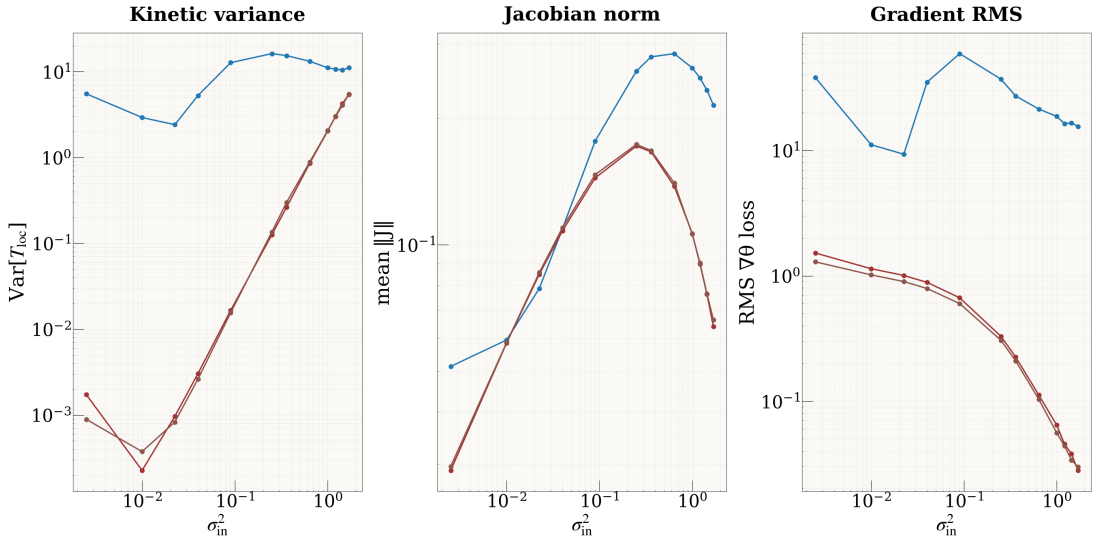
- **Jacobian norm** $\langle \|J\| \rangle$ (centre) is the average Frobenius norm of $\nabla\psi$. It quantifies first-derivative amplification: flat curves imply that layer gains collectively stay $\lesssim 1$, whereas growth with σ_{in}^2 reveals exploding sensitivity to input noise.
- **Gradient RMS** $\sqrt{\sum \|\partial_\theta \mathcal{L}\|^2}$ (right) is the root-mean-square back-propagated gradient of the PDE loss $\mathcal{L} = \langle (H\psi - E\psi)^2 \rangle$. It therefore reflects how violent each update step would be during training.

The He-initialized curves lie two to three orders of magnitude above the others on all three diagnostics, revealing uncontrolled gradient explosions. In contrast, LeCun and Xavier initializations track the input-variance scale, keeping both spatial derivatives and back-propagated gradients bounded. In other words, they preserve a stable variance range and prevent either exploding or vanishing gradients. Even though a four-layer network amplifies gradients relative to a single-layer model, LeCun and Xavier initializations still avoid the dramatic divergence seen with He.

Note that these diagnostics apply to untrained networks, so the exact loss-gradient variances during training may shift once weights begin to update.



(a) Gradient spectrum — 1-layer PINN



(b) Gradient spectrum — 4-layer PINN

Figure 13: Comparison of per-parameter gradient magnitudes for shallow (1-layer) versus deeper (4-layer) quantum-dot un-trained networks.

Speed

All training completed in approximately three minutes on a single CPU (no JIT or parallelization). Each run consisted of 500 epochs, with a batch size of $1000 \times N_{\text{particles}} \times d$ (e.g., $1000 \times 2 \times 2 = 4000$ collocation points per epoch for two particles in two dimensions).

4 Discussion

My guiding question was deliberately practical: *How much neural machinery is really needed once hard physics—antisymmetry via a Slater determinant—is baked in?* The answer, after hundreds of training curves and the gradient "seismograms" of Figs. 8–10, is surprisingly little. Two fully connected layers of 64 neurons, endowed with the *right* weight scale and activation curvature, recover or surpass Hartree–Fock, configuration interaction, and DMC benchmarks in both one– and two–dimensional testbeds. The convergence analysis adds nuance to that headline by exposing *how* and *why* the training either stabilises or derails.

What the gradient traces revealed

Figures 8 and 9 replace the usual "energy-only" scorecard with three orthogonal diagnostics: squared gradient norm, parameter–update ratio, and loss. When all three traces fall in concert (GELU/Mish with Xavier weights) the network crosses the Hartree–Fock baseline smoothly and never looks back. When they diverge (Sigmoid or Tanh, or any run that begins with He weights) the optimiser either starves or explodes before making physical progress. The diagnostic therefore functions as an *early–warning system*: a separation of two orders of magnitude between loss and gradient norm—visible for Sigmoid at $\omega = 0.1$—predicts vanishing gradients, while the inverse separation—He at both confinements—signals instability long before the energy has a chance to misbehave.

Activation curvature versus dynamic range

Why do GELU and Mish dominate while Sigmoid and Tanh fail? The key is the interplay between activation curvature and dynamic range.

Dynamic range. Sigmoid locks almost every walker into a narrow output band (visible in the untrained surfaces of Fig. 12); its first derivative is therefore near zero, and the Laplacian term $\nabla^2\psi$ starts *already* in the vanishing-gradient regime. Tanh expands the range but pays a price in curvature: its third derivative spikes at the origin, which the back–propagated Laplacian amplifies into jagged loss landscapes.

Curvature shaping. GELU and Mish taper both first *and* third derivatives to zero outside their active window, keeping the kinetic variance, Jacobian norm, and gradient RMS well behaved across four decades of input variance (Fig. 13). This single property explains why the same optimiser step size that is safe for $\omega = 1.0$ remains safe at $\omega = 0.1$, where Coulomb cusps sharpen dramatically.

Weight scale as pre-conditioning

The gulf between Xavier and He initialisation now reads as a *conditioning* story. Xavier presents the optimiser with an initial wave-function whose amplitude already matches the Laplacian and non-linear potential terms in the loss; He magnifies that amplitude by five orders of magnitude, so the optimiser's first duty is to undo the damage, amplifying every noise source in the process. In Fig. 9 this gulf appears as a one–to–two–order–of–magnitude gap between the loss curve and the gradient/on–parameter ratio: a textbook case of exploding gradients.

Depth as a conditional accelerator

Depth mattered—but only when the physics demanded it. At $\omega = 1.0$ the harmonic well dominates, and three layers are mandatory before the network can refine the Slater baseline. At $\omega = 0.1$ the Coulomb term rules, yet even here the two-layer network converges once the gradient conditioning is sound; layers three and four merely shorten the journey. Hence depth operates as a *conditional accelerator*: useful when the problem is hard, superfluous when the conditioning is already strong.

Synthesis

A single principle threads through these observations:

> *Conditioning, not complexity, determines success: if Xavier weights, smooth activations, and just enough depth keep gradients and Laplacians bounded, the smallest networks reach benchmark accuracy.*

Xavier initialisation puts the amplitude in that band; GELU/Mish keep the curvature there; two or three layers supply the spatial resolution required by the physics. Once those conditions are met, the leap from Hartree–Fock to chemical–accuracy correlation energy is incremental rather than heroic.

Limitations and Outlook

The encouraging results presented here rest on deliberately modest terrain: at most four fermions, four hidden layers, and fully connected feed-forward networks. Several limitations therefore deserve explicit acknowledgement.

- **System size.** Whether the conditioning strategy (Xavier + GELU + ≤ 4 layers) scales gracefully beyond ten electrons, or in three spatial dimensions, remains untested.
- **Architectural breadth.** No graph-convolution or message-passing layers were explored, yet such structures might capture collective modes more economically than dense layers.
- **Static activations.** All nonlinearities used fixed, hand-chosen curvature. Allowing the network to *learn* this curvature could enlarge the safe derivative band identified in Fig. 13.
- **Cold-start sensitivity.** He initialisation still fails catastrophically at strong coupling. Interaction-strength annealing offers one remedy, but its efficacy has not been quantified.

Each of these points leads to the same strategic objective: *maintain well-conditioned derivatives as the physical problem grows.* How best to achieve that at scale is the natural next phase of investigation.

Conclusion

We asked whether a *compact* physics-informed neural network, anchored by a Slater determinant, can reach chemical-accuracy energies without elaborate architectural engineering. The evidence says yes. For one-dimensional Gaussian fermions and two-dimensional Coulombic quantum dots, two-layer PINNs with Xavier weights and smooth nonlinearities (GELU, Mish, Swish) reproduce—or slightly improve upon—full configuration-interaction and diffusion Monte-Carlo benchmarks.

Three ingredients proved decisive:

- **Glorot–Xavier weight scale** aligns the untrained wave-function amplitude with the Laplacian term in the loss, preventing early runaway.
- **Smooth activation curvature** keeps first- and third-order derivatives bounded, stabilising back-propagated Laplacians even in the presence of sharp Coulomb cusps.
- **Moderate depth** (two–three hidden layers) supplies sufficient expressivity; deeper networks accelerate convergence only marginally while amplifying gradient variance.

The overarching principle is therefore simple: *conditioning trumps complexity.* Once the derivatives remain inside a numerically tame band, a small network suffices to capture strong many-body correlations.

Future Work

Several extensions follow naturally:

1. **Adaptive self-conditioning.** Layer-wise scale parameters or learned activations could keep derivative magnitudes balanced as system size grows.
2. **Beyond Slater: Pfaffian and back-flow baselines.** Embedding pair and three-body correlations *ab initio* will test whether the present "depth sweet-spot" persists.

3. **Time-dependent PINNs.** A log-amplitude formulation of the time-dependent Schrödinger equation would open the door to quench and transport studies.

4. **Interaction annealing.** Gradually ramping the coupling constant may convert cold-start failures into successes by ensuring the conditioning remains smooth throughout training.

5. **High-dimensional scalability.** Applying the conditioning rules to $N > 10$ particles in three dimensions will clarify whether dense layers still suffice or whether graph-based priors become essential.

6. **Layer testing** I unfortunately had no time exploring varying the number of layers between the three models, seeing which required most complexity.

Addressing these challenges will test the robustness of the central claim made here: *keep every derivative tame, and the physics will follow.*

Final Remarks

My code can be found here Sekkelsten (n.d.). My Theory on Neural networks are based on Hjorth-Jensen (2024a), and theory on quantum mechanics is based on Hjorth-Jensen (2024b). My benchmark Energies DMC, FCI and HF is from Haas Baccatini Lima (2024). The initlization techniques are based on He et al. (2015) and Glorot and Bengio (2010).

A Implementation Details

The numerical experiments reported in this work were carried out with the hyper-parameter settings listed below. Unless explicitly noted, the same configuration was employed for every Hamiltonian studied.

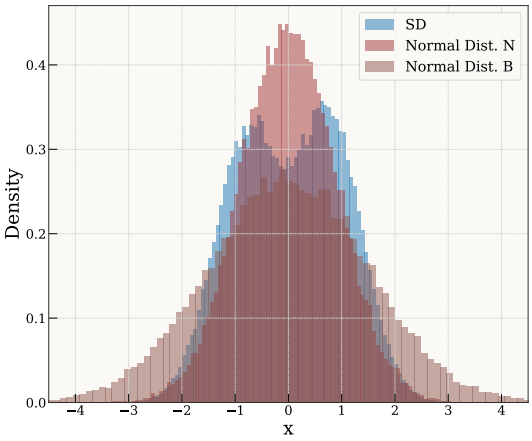
- **Neural-network architecture.** A fully connected feed-forward network comprising two hidden layers, each containing 64 neurons, with GELU activation functions after every hidden layer.
- **Optimizer.** Parameters were updated with the Adam algorithm.
- **Sampling and training schedule.**
  - For the two-electron, two-dimensional systems, every training epoch drew 1 000 collocation points from a Gaussian distribution $\mathcal{N}(0, \sigma^2)$ in each Cartesian coordinate ($\sigma = 1$ unless indicated otherwise).
  - For the one dimensional Gausian interacting fermions, I sample directly from the slaterdeterminant with normalizing flows.
  - When training the Gaussian-interaction models, an additional normalisation penalty was included in the loss function; this term was not required for the quantum-dot Hamiltonian.
  - The learning rate was fixed at 1×10^{-3} throughout, and all models were trained for 500 epochs.

B Sampling X-data with Normalizing Flows

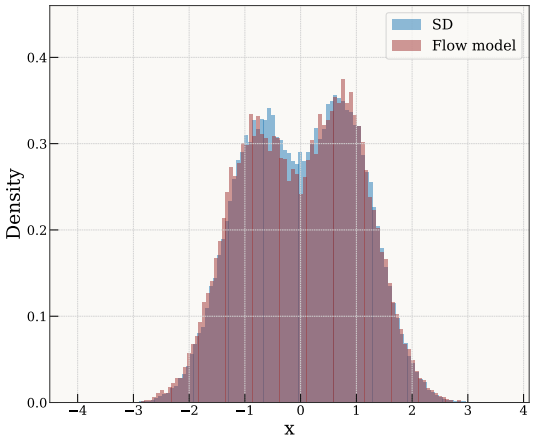
In training physics-informed neural networks (PINNs) to solve complex problems such as the many-body Schrödinger equation, the quality and distribution of training samples are critical. Conventional sampling techniques—such as narrow normal, broad normal, and uniform sampling—each have significant drawbacks. For example:

- **Narrow Normal Sampling:** Concentrates samples near the center of the domain, leading to insufficient coverage at the boundaries where the solution may still be significant.
- **Broad Normal Sampling:** Oversamples regions with little physical relevance, thus wasting computational resources.
- **Uniform Sampling:** Provides inflexible coverage, often missing regions of high interest such as nodes or sharp gradients.

To overcome these issues, I adopt a normalizing flow framework. This method transforms a simple base distribution (typically a standard normal distribution) into a distribution that better aligns with the target function (e.g., the Slater Determinant). By learning an invertible mapping, normalizing flows adaptively concentrate samples where they are most needed, improving both efficiency and training accuracy.



(a) Narrow Normal Distribution Oversampling the Central Region, and Broad Normal Distribution oversampling near the edges



(b) Sampling with Normalizing Flow, reproducing the Slater Determinant

Normalizing Flow Sampling

Normalizing flows work by constructing an invertible transformation T that maps a sample x drawn from a simple base distribution $p_X(x)$ (e.g., $x \sim \mathcal{N}(0, I)$) to a sample $y = T(x)$ that follows the target distribution $p_Y(y)$. The change-of-variables formula governs this mapping:

$$p_Y(y) = p_X(x) \left| \det \left(\frac{\partial T^{-1}(y)}{\partial y} \right) \right|.$$

In my approach, I use a variant known as *conditional flow matching*. Rather than learning the transformation T directly, I conceptualize the mapping in terms of a displacement field. Given a base sample x and an associated target sample y, the ideal displacement is defined as:

$$v_{\text{true}} = y - x.$$

We then introduce a continuous interpolation parameter $t \in [0, 1]$ and define:

$$\psi_t = (1 - t)x + t\, y,$$

which smoothly bridges the base and target distributions. my goal is to learn a function $v(\psi_t, t)$ that predicts the required displacement, thereby reconstructing the overall mapping T.

Training Procedure

The normalizing flow model is trained to capture the transformation dynamics from the simple base distribution to the complex target distribution. The training process comprises the following steps:

1. **Sample Generation:**
   - Generate a batch of base samples x from a standard normal distribution $\mathcal{N}(0, I)$.
   - Obtain target samples y using a reliable method (e.g., via a Metropolis–Hastings algorithm or from experimental data) that reflects the desired distribution.
2. **Interpolation:** For each pair (x, y), select a random interpolation parameter $t \in [0, 1]$ and compute the interpolated state:
$$\psi_t = (1 - t)x + t\, y.$$

3. **Displacement Prediction and Loss Calculation:** A neural network—typically a multilayer perceptron (MLP)—is employed to predict the displacement $v_{\text{predicted}}(\psi_t, t)$. The network is trained by minimizing the mean squared error (MSE) between the predicted and true displacements:

$$\mathcal{L} = \mathbb{E}_{x,y,t}\left[\|v_{\text{predicted}}(\psi_t, t) - (y - x)\|^2\right].$$

4. **Optimization:** I use an optimizer such as Adam to update the network parameters based on the computed loss. Training is performed over multiple epochs until convergence is observed. Once trained, the model is capable of rapidly generating uncorrelated samples that align with the target distribution.

Advantages and Implementation Details

The use of normalizing flows offers several advantages over traditional sampling methods:

- **Adaptive Sampling:** The learned transformation focuses sampling in regions of high importance, avoiding both undersampling and oversampling.
- **Efficiency:** Unlike methods requiring a burn-in period (as in Markov Chain Monte Carlo), the normalizing flow generates independent samples quickly once training is complete.
- **Automation:** The framework removes the need for manual hyperparameter tuning of the sampling distribution, making it robust to changes in system parameters.

For one dimensional spinless fermions, I integrate the normalizing flow sampling within the overall training pipeline of the PINN, to great success.

References

Glorot, X., & Bengio, Y. (2010, 13–15 May). Understanding the difficulty of training deep feedforward neural networks. In Y. W. Teh & M. Titterington (Eds.), *Proceedings of the thirteenth international conference on artificial intelligence and statistics* (Vol. 9, pp. 249–256). Chia Laguna Resort, Sardinia, Italy: PMLR. Retrieved from https://proceedings.mlr.press/v9/glorot10a.html

Haas Baccatini Lima, D. (2024). *Deep learning methods for quantum many-body systems, a study on neural quantum states* (Master's thesis, University of Oslo). Retrieved from http://hdl.handle.net/10852/113984

He, K., Zhang, X., Ren, S., & Sun, J. (2015). *Delving deep into rectifiers: Surpassing human-level performance on imagenet classification.* Retrieved from https://arxiv.org/abs/1502.01852

Hermann, J., Schätzle, Z.-H., & Noé, F. (2020). Paulinet: Deep learning of many-electron wavefunctions. *Journal of Chemical Theory and Computation*, *16*(10), 6196–6209. DOI: 10.1021/acs.jctc.0c00492

Hjorth-Jensen, M. (2024a). *Applied data analysis and machine learning.* Retrieved from https://compphysics.github.io/MachineLearning/doc/LectureNotes/_build/html/intro.html (Accessed: 2024-10-07)

Hjorth-Jensen, M. (2024b). *Applied data analysis and machine learning.* Retrieved from https://github.com/CompPhysics/MachineLearning (Accessed: 2025-18-03)

Pfau, L., Spencer, J. S., Matthews, A. G. d. G., & Foulkes, W. M. C. (2020). Ab initio solution of the many-electron schrödinger equation with deep neural networks. *Physical Review Research*, *2*(3), 033429. DOI: 10.1103/PhysRevResearch.2.033429

Sekkelsten, A. (n.d.). *UIO Numerical work.* Retrieved from https://github.com/Im2ql4u/Machine-learning-UIO/tree/main/FYS5429