Neural Network Solutions of Bosonic and Fermionic Systems in One Dimension

Hersh Kumar
Advised by Prof. Paulo Bedaque
In collaboration with Dr. Andy Sheng

University of Maryland, Department of Physics

Finding Ground States

- Finding the ground state energy is difficult for many-body quantum systems
  - High dimensionality makes exact solutions hard to solve for
- Models with analytic solutions are uncommon
- Instead, apply numerical methods to obtain estimates of ground state properties

Variational Method

Given an arbitrary Hamiltonian $\hat{H}$, and state $|\psi\rangle$, the variational principle provides a bound on the value of the ground state energy of the Hamiltonian

The Variational Principle

For Hamiltonian $\hat{H}$, and arbitrary state $|\psi\rangle$:

$$E_0 \langle\psi|\psi\rangle \leq \langle\psi|\hat{H}|\psi\rangle$$

- Choose $\psi(\boldsymbol{\theta})$, then vary $\boldsymbol{\theta}$ to minimize $\langle\psi|\hat{H}|\psi\rangle \rightarrow$ estimate for E_0.
- In practice, we have many variational parameters, minimization of $\langle\hat{H}\rangle$ requires Monte Carlo sampling and gradient descent.

Monte Carlo Methods

We need to compute the expectation value:

$$\langle \hat{H} \rangle = \frac{\int d\boldsymbol{x}\, \psi\left(\boldsymbol{x}\right)^{\dagger} \hat{H} \psi\left(\boldsymbol{x}\right)}{\int d\boldsymbol{x}\, \psi\left(\boldsymbol{x}\right)^{\dagger} \psi\left(\boldsymbol{x}\right)}$$

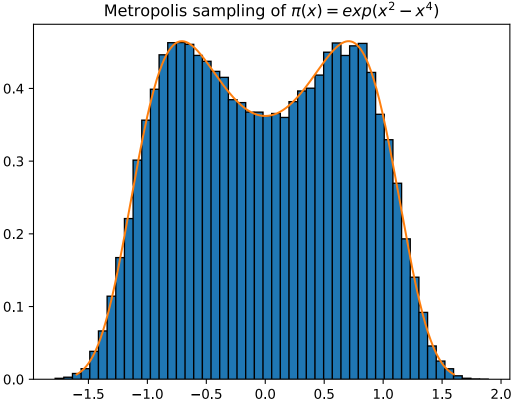
This is difficult for systems with many particles, so instead, we write it as a statistical average

$$\langle \hat{H} \rangle = \left\langle \psi^{-1}\left(\boldsymbol{x}\right) \hat{H} \psi\left(\boldsymbol{x}\right) \right\rangle_{\psi^2}$$

Where the average is taken over samples from ψ^2.

Metropolis Algorithm

- Metropolis-Hastings allows for sampling from probability distributions
- Easily generalizes to multiple dimensions $\rightarrow$ many-body quantum systems



Metropolis Algorithm

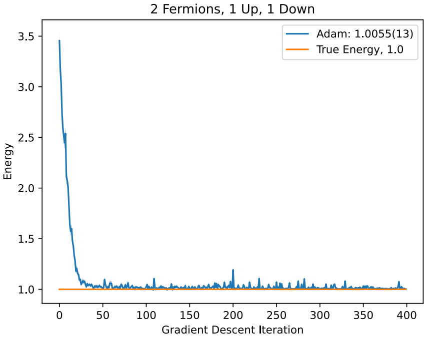
Given a probability distribution function $P(\boldsymbol{x})$:

1. Generate a starting sample, $\boldsymbol{x}_t$.
2. Generate a random shift, $\boldsymbol{r}$ (sampled uniformly within a range).
3. Let $\boldsymbol{x}' = \boldsymbol{x}_t + \boldsymbol{r}$. This is the proposed next sample.
4. Generate random $p \in [0, 1]$. If $p < \frac{P(\boldsymbol{x}')}{P(\boldsymbol{x})}$, accept the sample, and use it as the next starting sample. Otherwise, reject the sample, and keep the starting sample the same.

The algorithm draws samples from high-density regions, and rejects proposals from low-density regions $\rightarrow$ samples mimic the given probability distribution.

Variational Monte Carlo

- Applying Variational Method to obtain estimate for $E_0 \rightarrow$ using Monte Carlo to compute expectation values
- Use gradient descent algorithm to minimize $\mathcal{E}$ by varying $\boldsymbol{\theta}$.
  - Compute $\frac{d\mathcal{E}}{d\boldsymbol{\theta}}$, shift values of $\boldsymbol{\theta}$, repeat until $\mathcal{E}$ converges
- Determine convergence by comparison to theory or by increasing the number of samples



Why Neural Networks?

The Variational Principle gives no information about how close the estimate is to E_0.

Ansatze are human-generated, based on knowledge of the Hamiltonian or ground state behavior:

- Harmonic oscillator potential $\rightarrow$ inverted quadratic, $\left(x+\frac{a}{2}\right)\left(x-\frac{a}{2}\right)$
- Helium atom $\rightarrow$ Hydrogen atom ground state with added variational parameters

Ideal ansatz is something that can take on a large range of functional forms.

Neural Networks

Define a network of L layers, weights $\{\boldsymbol{W}_i\}$, biases $\{\boldsymbol{b}_i\}$, and activation function f:

$$\begin{aligned} I_{i+1} &= O_i = f(\mathbf{W}_i \boldsymbol{I_i} + \boldsymbol{b}_i), \qquad 1 \leq i \leq L-1 \\ O_L &= \mathbf{W}_L \boldsymbol{I}_L + \boldsymbol{b}_L \end{aligned}$$

- O_L has no activation function, so that the network can produce any value as output
- We use $f = \text{CELU}$:

$$\text{CELU}\,(x, \alpha) = \begin{cases} x & x > 0 \\ \alpha \left(\exp\left(\frac{x}{\alpha}\right) - 1 \right) & x \leq 0 \end{cases}$$

Neural Networks

- Neural networks have been shown to be universal function approximators[1]
- **Why not use neural networks as the VMC ansatz?**
- Weights and biases of the network serve as variational parameters
- Neural network should converge to ground state wavefunction given enough weights and biases
- Caveat: Systems of bosons and fermions have exchange (anti)symmetries that need to be encoded

[1] K. Hornik, M. Stinchcombe, and H. White, Multilayer feedforward networks are universal approximators, Neural networks **2**, 359 (1989)

Bosonic Exchange Symmetry

- Neural network output should be invariant under coordinate exchange
- Use a bijection from $\{x_i\}$ to some exchange symmetric $\{\xi_i\}$

$$\xi_i = \sum_{k=1}^{N} \left(\frac{x_k}{w}\right)^i$$

- Exchanging x_i and x_j produces the same $\{\xi_i\}$

$$\begin{aligned} \xi_1 &= x_1^1 + x_2^1 + \cdots + x_N^1 \\ \xi_2 &= x_1^2 + x_2^2 + \cdots + x_N^2 \\ &\vdots \\ \xi_N &= x_1^N + x_2^N + \cdots + x_N^N \end{aligned}$$

Bosonic Ansatz

- We define our bosonic ansatz:

$$\psi\left(x_1, x_2, \ldots, x_N\right) = e^{-\mathcal{A}(\xi_1, \xi_2, \ldots, \xi_N))} \cdot e^{-\Omega \sum_{i=1}^{N} x_i^2}$$

- Gaussian ensures that ψ vanishes at infinity
- ψ is real-valued $\rightarrow$ real-valued weights and biases
- Exchange symmetry is maintained using $\{\xi_i\}$ as neural network inputs

Fermionic Ansatz

- For a system of $N_\uparrow$ spin-up and $N_\downarrow$ spin-down fermions (N total), we expect exchange antisymmetry between same spin fermions

$$\underbrace{x_1, x_2, \ldots, x_{N_\uparrow}}_{\text{up-spin}}, \underbrace{x_{N_\uparrow+1}, x_{N_\uparrow+2}, \ldots x_N}_{\text{down-spin}}$$

- Ansatz constructed using two Slater determinants, one for each set of spins
  - Each determinant is antisymmetric for its species of fermion

Fermionic Ansatz

$$\Phi^{\uparrow}(\boldsymbol{x}) = \frac{1}{\sqrt{N!}} \begin{vmatrix} \phi_{1,1} & \phi_{1,2} & \cdots \\ \phi_{2,1} & \phi_{2,2} & \\ \vdots & & \ddots \end{vmatrix}$$

$$\Phi^{\uparrow}(\boldsymbol{x}) = \frac{1}{\sqrt{N!}} \begin{vmatrix} \phi_{N_\uparrow+1,N_\uparrow+1} & \phi_{N_\uparrow+1,N_\uparrow+2} & \cdots \\ \phi_{N_\uparrow+2,N_\uparrow+1} & \phi_{N_\uparrow+2,N_\uparrow+2} & \\ \vdots & & \ddots \end{vmatrix}$$

Where $\phi_{i,j}$ is a neural network function $\phi_i(x_j, x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_N)$ which is symmetric with respect to all but x_j.

$$\psi(\boldsymbol{x}) = \Phi^{\uparrow}(\boldsymbol{x})\,\Phi^{\downarrow}(\boldsymbol{x})\,e^{-\sum_i^N x_i^2}$$

Bosonic Models

We consider 2 models of interacting bosons, both with harmonic potentials.

- Harmonically trapped bosons with a delta function potential
  - $\hat{H} = \sum_{i=1}^{N} \left(-\frac{1}{2m} \frac{\partial^2}{\partial x_i^2} + \frac{1}{2} m\omega^2 x_i^2 \right) + \sum_{i<j}^{N} g\delta\left(x_i - x_j\right)$
  - Exactly solvable in the 2 particle case

- Harmonically trapped bosons with short and long range interactions
  - $\hat{H} = \sum_{i=1}^{N} \left(-\frac{1}{2m} \frac{\partial^2}{\partial x_i^2} + \frac{1}{2} m\omega^2 x_i^2 \right) + \sum_{i<j}^{N} g\delta\left(x_i - x_j\right) + \sigma|x_i - x_j|$
  - Exactly solvable in the case where $\sigma = -m\omega g/2$

Fermionic Models

We consider 2 models of fermions, with harmonic potentials.

- $N = N_\uparrow + N_\downarrow$ non-interacting fermions
  - $\hat{H} = \sum_i^N \left(-\frac{1}{2m} \frac{\partial^2}{\partial x_i^2} + \frac{1}{2} m\omega^2 x_i^2 \right)$
  - Exactly solvable
- $N = N_\uparrow = N_\downarrow$ fermions with a contact potential
  - $\hat{H} = \sum_i^N \left(-\frac{1}{2m} \frac{\partial^2}{\partial x_i^2} + \frac{1}{2} m\omega^2 x_i^2 \right) + \sum_{i<j} g\delta\left(x_i - x_j\right)$
  - $N_\downarrow = 1$, $N_\uparrow = 1, 2, \ldots, 10$ cases are solved exactly

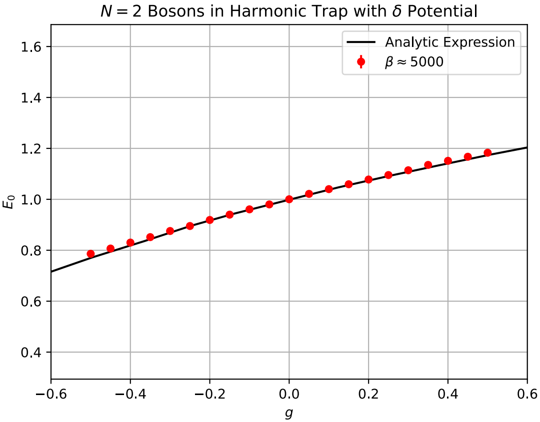
Implementation

- Python: numpy + Jax
- Optimize Monte Carlo computations using vectorization
- Jax just-in-time (jit) compilation optimizes code reuse
- Gradient descent using ADAM optimizer

Bosons with Contact Potential

$N = 2$ case has exact solution:

$$\sqrt{2}\left(\frac{\Gamma\left(1-\frac{E_0}{2}\right)}{\Gamma\left(\frac{1}{2}-\frac{E_0}{2}\right)}\right) = -\frac{2}{g}$$



Bosons with Short and Long Range Potentials

Look at two regimes:

- Exactly solvable regime: $\sigma = -g/2$
- A non-analytically solvable regime: $\sigma = -g$

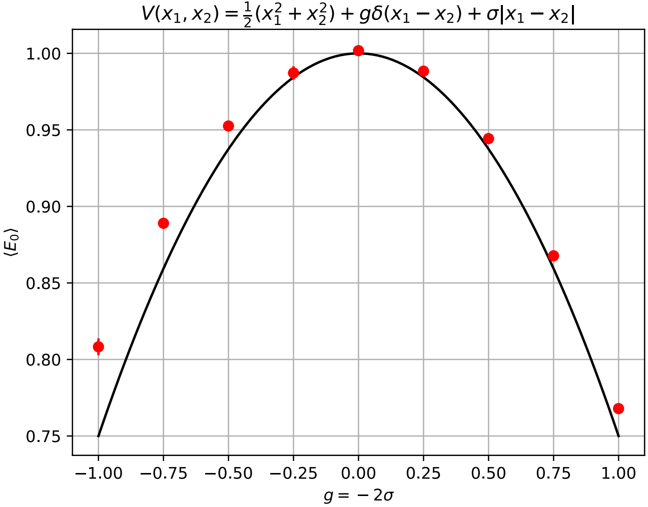
In the exactly solvable regime:

$$E_0 = \frac{N\omega}{2} - mg^2 \frac{N\left(N^2 - 1\right)}{24}$$

$$\psi\left(\boldsymbol{x}\right) = \prod_{i<j} e^{-|x_i - x_j|/a_s} \prod_{i} e^{-x_i^2/\left(2a_{ho}^2\right)}$$

Where $a_s = -\frac{2}{mg}$ and $a_{ho} = \sqrt{\frac{1}{m\omega}}$.

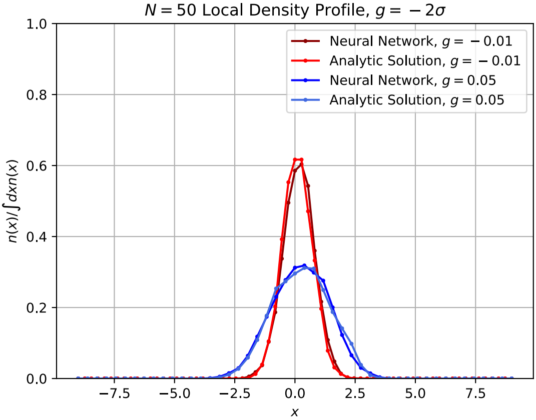
Analytic Regime ($\sigma = -g/2$)



Large N Systems

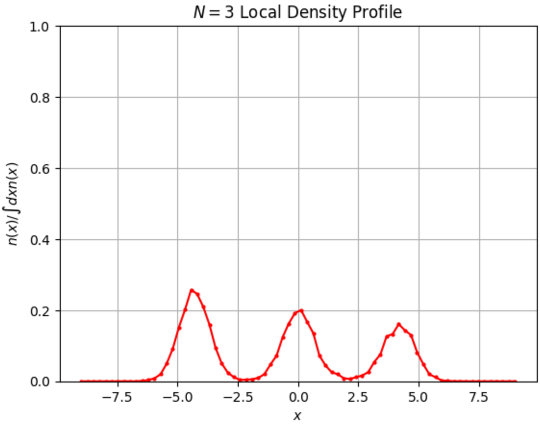
Compute the local density profile:

$$n(x) = \int dx_2, \ldots, dx_N \left|\psi(x_1, x_2, \ldots, x_N)\right|^2$$



Nonanalytic Regime ($\sigma = -g$)

Neural network ansatz encodes correct qualitative behavior of the ground state for repulsive g:

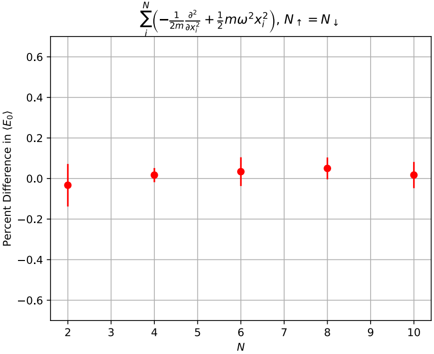


Fermions in Harmonic Well

System of fermions with no interactions has known ground state solution:

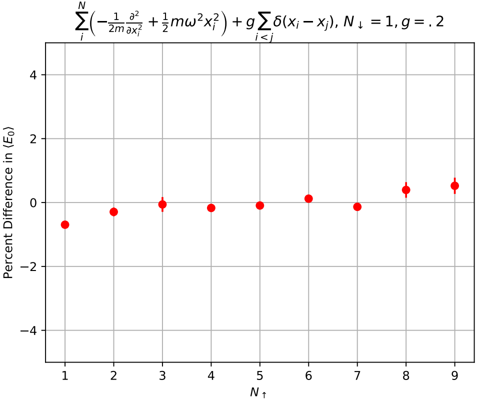
$$E_0 = \frac{N_\uparrow^2 + N_\downarrow^2}{2}$$

Percent difference from exact ground state energy:



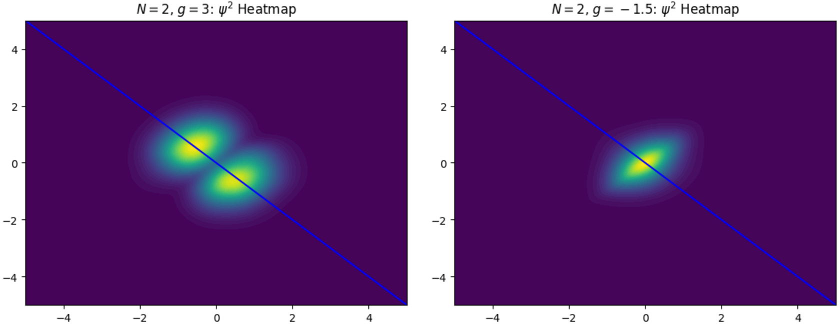
Fermions with Contact Potential

Percent difference between neural network and exact results:



Fermion Probability Density

Position-space probability density of repulsive and attractive ground states of fermions with delta function interaction:



Conclusion

- Neural networks can provide general wavefunctions for bosonic and fermionic models
- Allow for straightforward increases in the number of variational parameters, and number of particles
- Ability to probe systems with no analytic solutions

Further Work

- Expand to higher dimensional systems
  - 3D $\rightarrow$ requires 3 dimensional symmetrization method
- Explore GPU parallelization for MC computations
- Study scaling of parameter counts
- Explore transfer learning for different interaction strengths
- Explore methods for systematically choosing optimization hyperparameters

Questions?