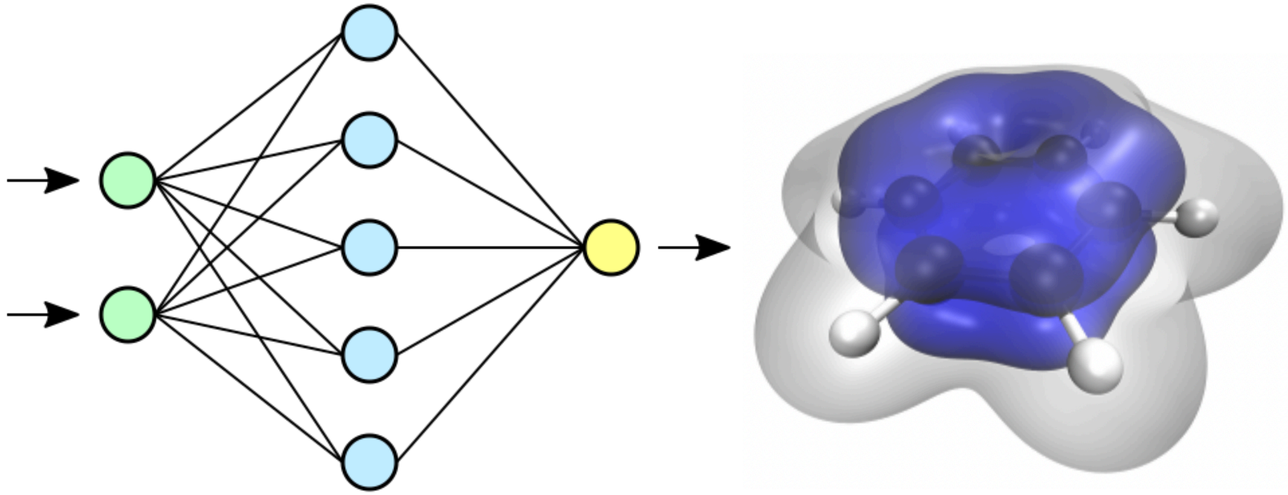
Deep Learning and *Ab-initio* Electronic Structure

Machine Learning for Materials Hard and Soft
University of Vienna
14 July 2022



David Pfau
DeepMind, Imperial College London
pfau@deepmind.com

Deep Learning

Deep Learning

D. Silver *et al.* Nature (2016)

Deep Learning

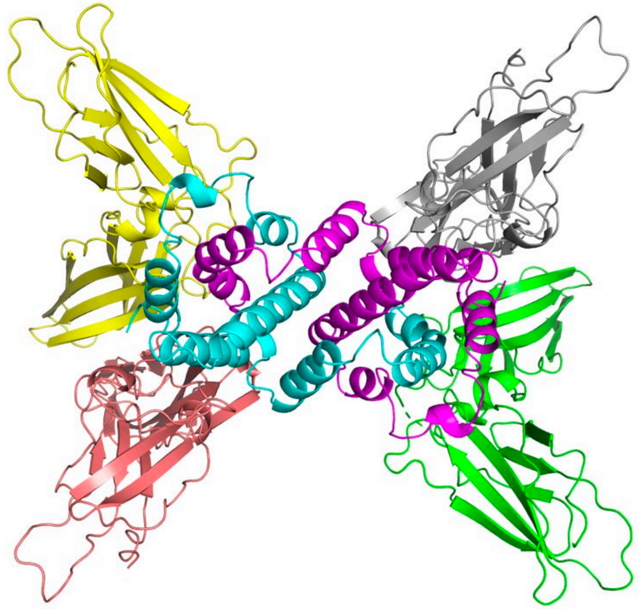
D. Silver *et al.* Nature (2016)

A. Ramesh *et al.* arXiv (2022)

Deep Learning

D. Silver *et al.* Nature (2016)

A. Ramesh *et al.* arXiv (2022)

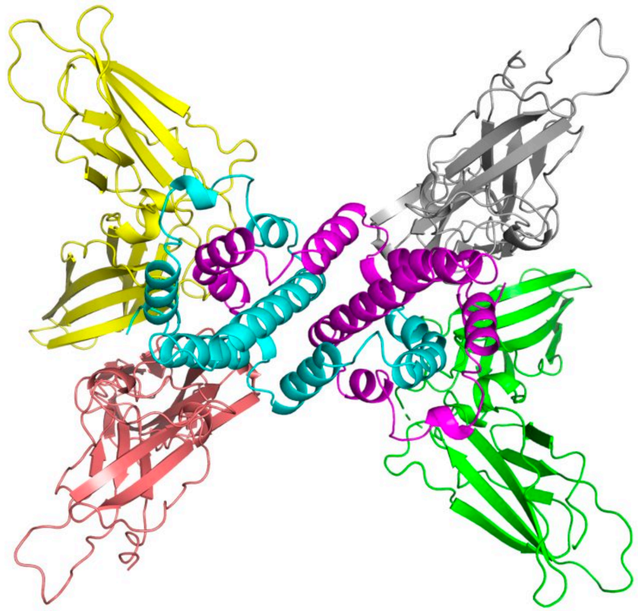


J. Jumper *et al.* Nature (2021)

Deep Learning

D. Silver *et al.* Nature (2016)

A. Ramesh *et al.* arXiv (2022)



J. Jumper *et al.* Nature (2021)

Very successful at **learning and representing high-dimensional functions**

Electronic Structure

$$\hat{H}\Psi = E\Psi$$

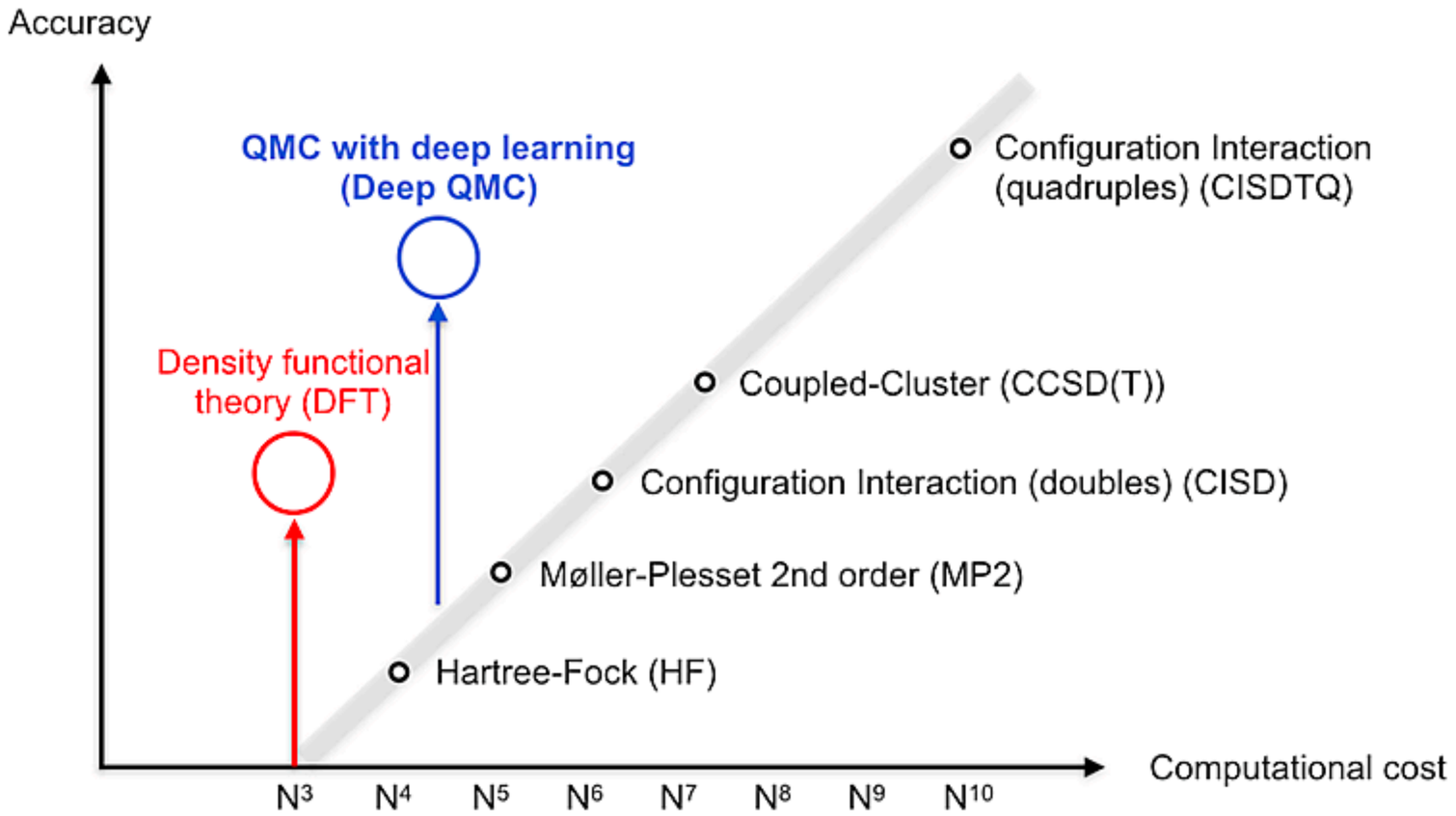
$$\hat{H} = -\frac{1}{2}\sum_i \nabla_i^2 - \sum_{iI}\frac{Z_I}{|r_i - R_I|} + \sum_{ij}\frac{1}{|r_i - r_j|} + \sum_{IJ}\frac{Z_I Z_J}{|r_I - r_J|}$$

- The fundamental equation of **chemistry**, **material science**, and **condensed matter physics**
- The solution is a **high dimensional function**
- Could deep learning enable representations of electronic wavefunctions which are both **compact** and **general**?

How can the two be combined?

Deep QMC

$$\hat{H}\psi(\mathbf{r}_1,\ldots,\mathbf{r}_N) = E\psi(\mathbf{r}_1,\ldots,\mathbf{r}_N)$$

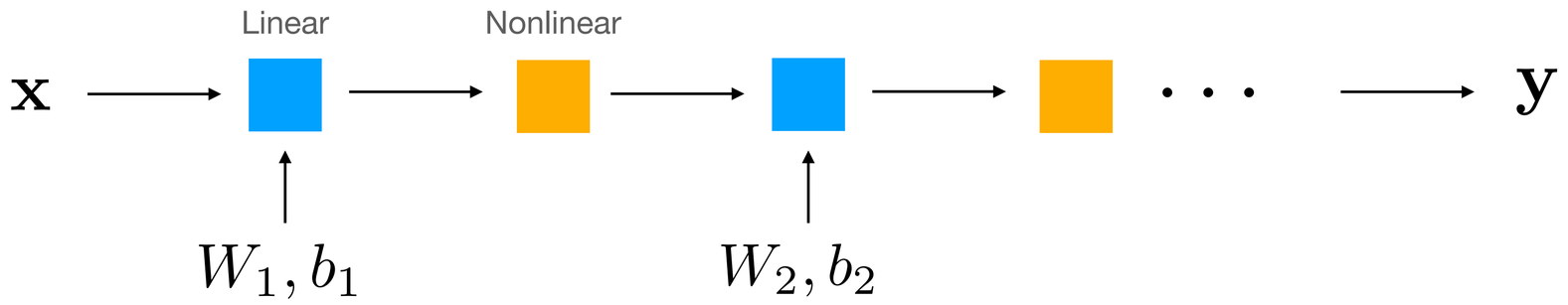


I. Invariance and Equivariance in Deep Learning

Multi-Layer Perceptrons

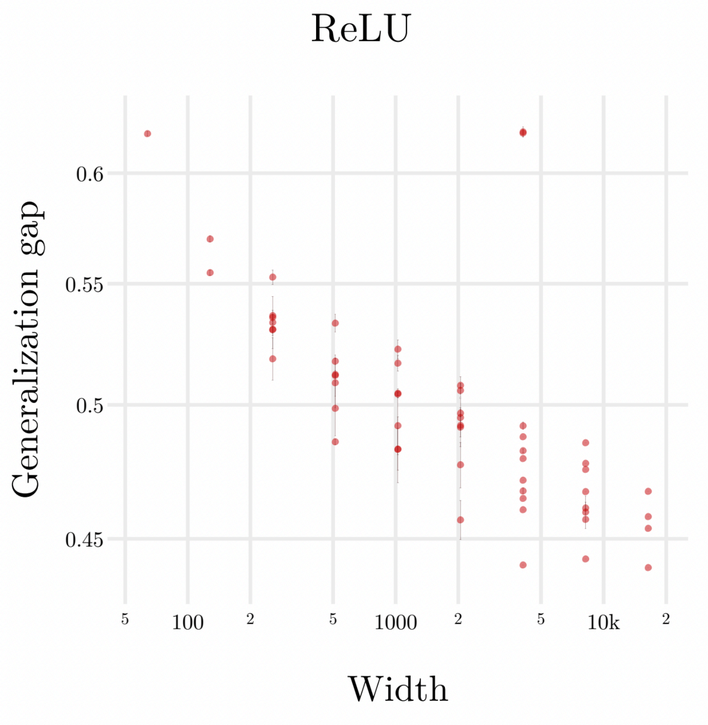
$$\mathbf{y} = f_\ell \circ f_{\ell-1} \circ \ldots \circ f_2 \circ f_1(\mathbf{x})$$

$$f_i(\mathbf{a}_i) = \sigma(W_i \mathbf{a}_i + b_i)$$



Essentially stacked regression models. The starting point for neural networks.

Multi-Layer Perceptrons



Num training	Model (ReLU)	Test accuracy	Model (tanh)	Test accuracy
MNIST:1k	NN-2-5000-3.19-0.00	0.9252	NN-2-1000-0.60-0.00	0.9254
	GP-20-1.45-0.28	**0.9279**	GP-20-1.96-0.62	0.9266
MNIST:10k	NN-2-2000-0.42-0.16	0.9771	NN-2-2000-2.41-1.84	0.9745
	GP-7-0.61-0.07	0.9765	GP-2-1.62-0.28	**0.9773**
MNIST:50k	NN-2-2000-0.60-0.44	0.9864	NN-2-5000-0.28-0.34	0.9857
	GP-1-0.10-0.48	0.9875	GP-1-1.28-0.00	**0.9879**

- **Gaussian processes** (GPs) can approximate any MLP in the limit of **infinitely-wide layers**
- GPs **match or exceed** performance of MLPs on standard benchmarks
- MLPs aren’t magic - the real power of deep learning is in more **complex architectures**

J. Lee, Y. Bahri, R. Novak, S. S. Schoenholz, J. Pennington and J. Sohl-Dickstein, ICLR (2018)

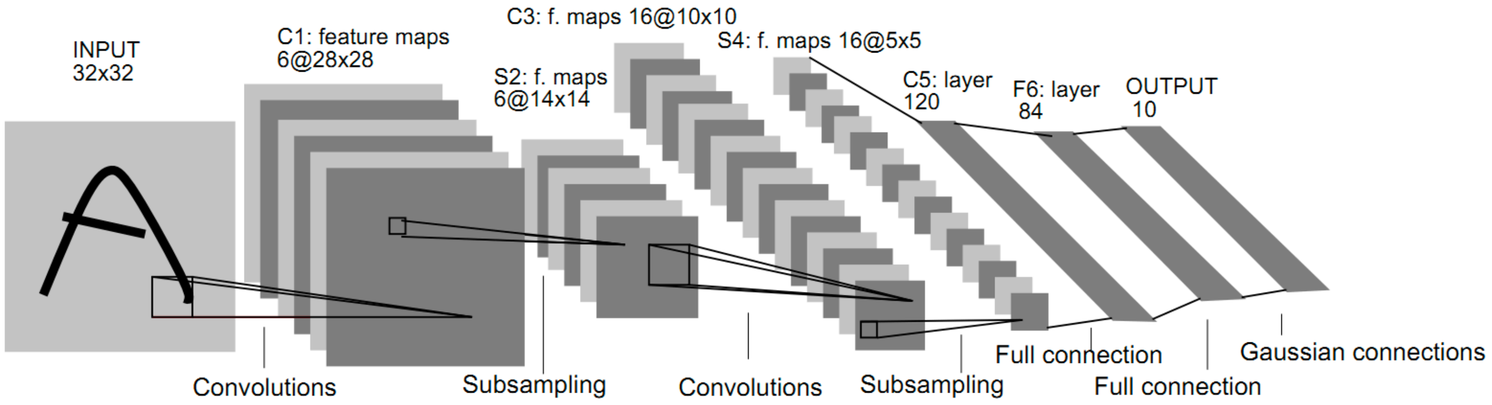
Beyond Multi-Layer Perceptrons

Convolutional Neural Networks

Feature map Layer

$$h_{ij}^{k\ell+1} = f\left(\sum_{i'j'k'} w_{i'j'}^{kk'\ell} h_{(i-i')(j-j')}^{k'\ell} + b^{k\ell}\right)$$

x and y coord Discrete convolution between k and k'



Y. LeCun (1989)

Invariance and Equivariance

Definition

Invariance

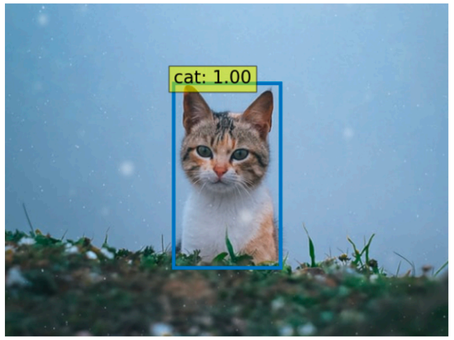
Example: Object **Recognition**
Is there a cat?

Cat?

$$f(g \circ \mathbf{x}) = f(\mathbf{x})$$

Equivariance

Example: Object **Detection**
Where is the cat?



$$f(g \circ \mathbf{x}) = \rho(g) \circ f(\mathbf{x})$$

Image Credit: Hugging Face

Invariance and Equivariance

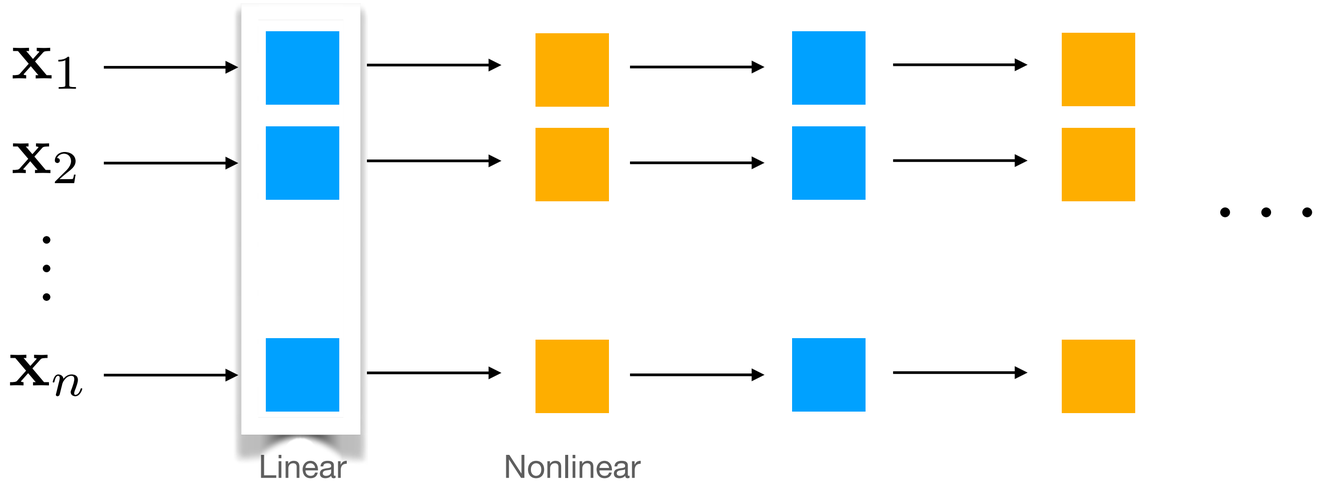
Advantages

- Dramatically reduces the **number of parameters** needed to learn the same function
- Dramatically reduces the **amount of data** needed to learn
- Builds in **robustness** against perturbations not in the training set
- Builds in **physics constraints** from the start

How to Build Equivariant Nets

Simple Example: Permutation

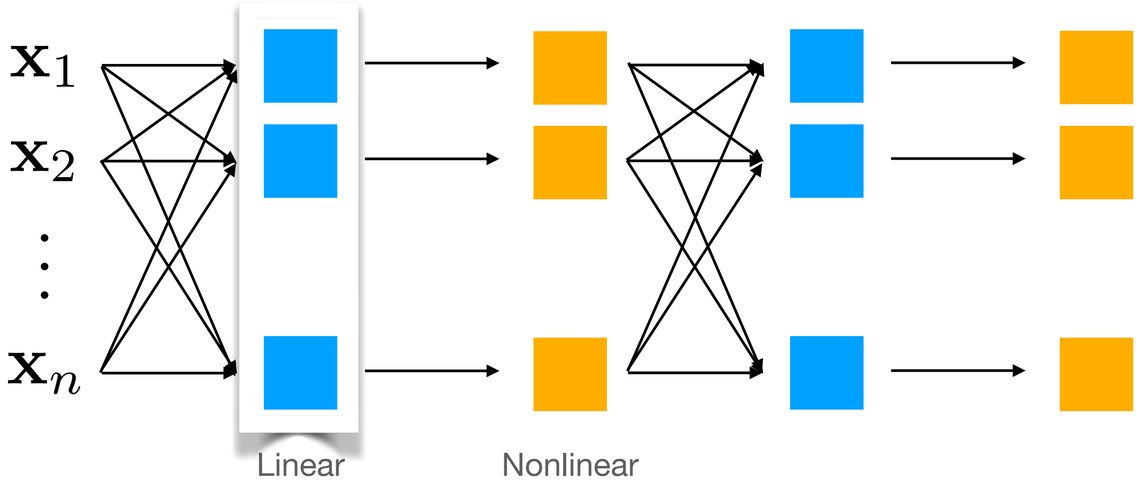
$$\begin{pmatrix} W & 0 & \dots & 0 \\ 0 & W & \dots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & W \end{pmatrix} \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_n \end{pmatrix} + \begin{pmatrix} b \\ b \\ \vdots \\ b \end{pmatrix}$$



How to Build Equivariant Nets

Simple Example: Permutation

$$\begin{pmatrix} W & V & \dots & V \\ V & W & \dots & V \\ \vdots & \vdots & & \vdots \\ V & V & \dots & W \end{pmatrix} \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_n \end{pmatrix} + \begin{pmatrix} b \\ b \\ \vdots \\ b \end{pmatrix}$$



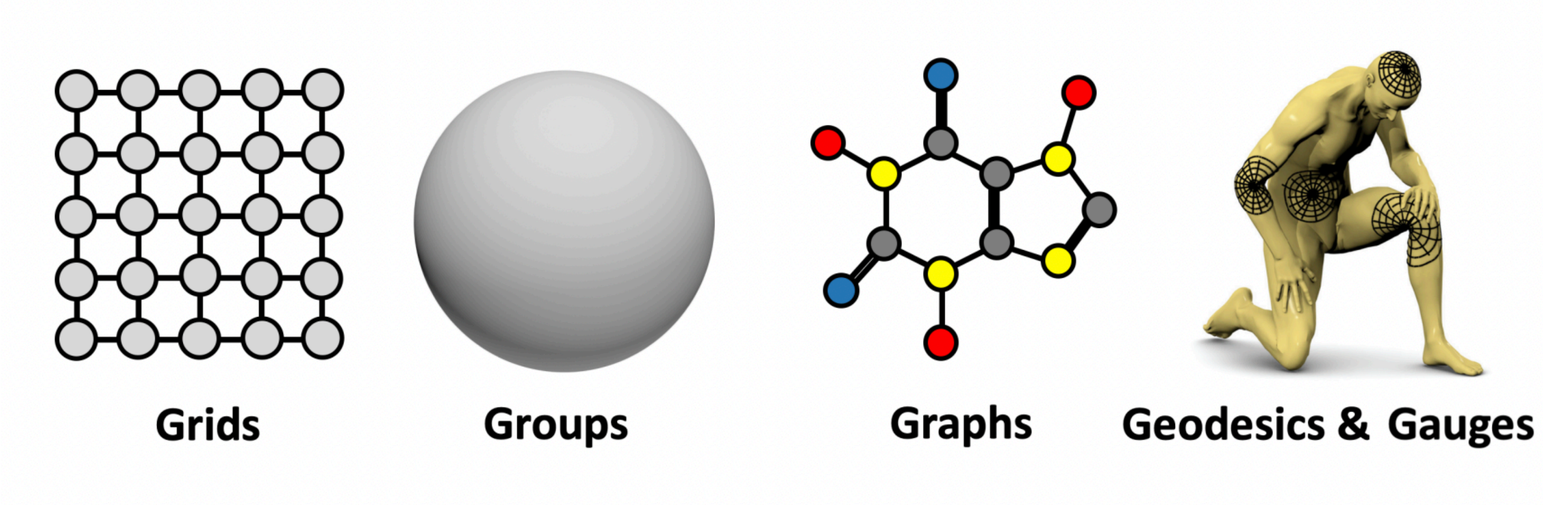
$\dots$

$$W\mathbf{x}_i + \sum_{j \neq i} V\mathbf{x}_j$$

How to Build Equivariant Nets

General Theory: Groups & Representation Theory

Geometric Deep Learning



http://geometricdeeplearning.com

M. M. Bronstein, J. Bruna, T. Cohen P. Veličkovic (2021)

How to Build Equivariant Nets

Groups

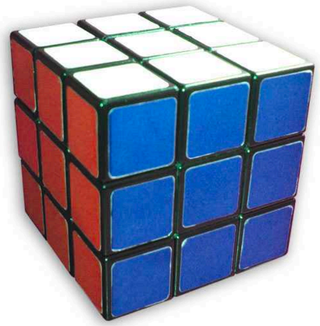
Set of **operations** which can be composed and inverted. Formalizes **symmetries**.

$g, h \in G \Rightarrow g \circ h \in G$ Closed under **composition**

$\exists e \in G \, s.t. \, \forall g \in G, e \circ g = g$ Existence of an **identity** operation

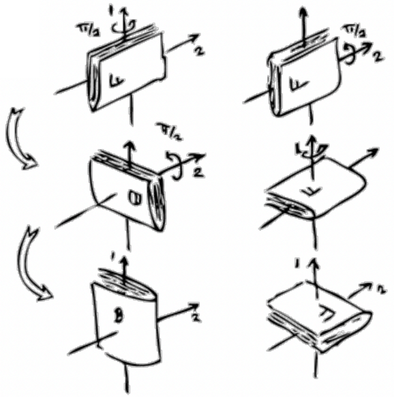
$\forall g \in G, \exists g^{-1} \in G \, s.t. \, g \circ g^{-1} = e$ Every element has an **inverse**

Examples: **addition** of integers, **multiplication** of reals, **permutation**, **translation**, **rotation**



Cat

Cat



How to Build Equivariant Nets

Representations

Mapping from **group elements** to **matrices**

$$\rho : G \to GL_n(\mathbb{R})$$

The representation is an **invertible matrix**

$$\rho(g \circ h) = \rho(g)\rho(h)$$

Composition becomes **matrix multiplication**

$$\rho(e) = \mathbf{I}$$

The **identity** is unique

$$\rho(g^{-1}) = \rho(g)^{-1}$$

The representation of the **inverse** is the inverse of the representation

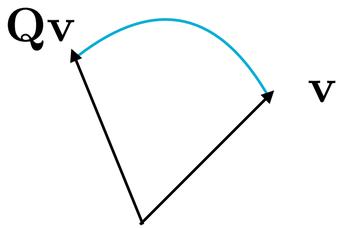
How to Build Equivariant Nets

Representations

Representations of 3D Rotation: The Special Orthogonal Group **SO(3)**

$$\rho(g) = \mathbf{Q} \in GL_3(\mathbb{R})$$

The representation is a **3x3 matrix**



$$\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$$

The representation is **orthonormal**

$$\det[\mathbf{Q}] = 1$$

The representation has **determinant one**
(parity doesn't change)

A natural representation for operating on 3D vectors. But what about **functions?**

How to Build Equivariant Nets

Representations

Representations of 3D Rotation: Spherical Harmonics and **Wigner D-matrices**

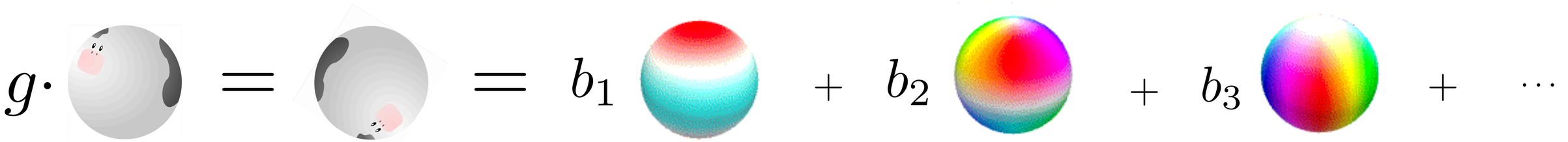


Image Credit: Keenan Crane and Wikipedia

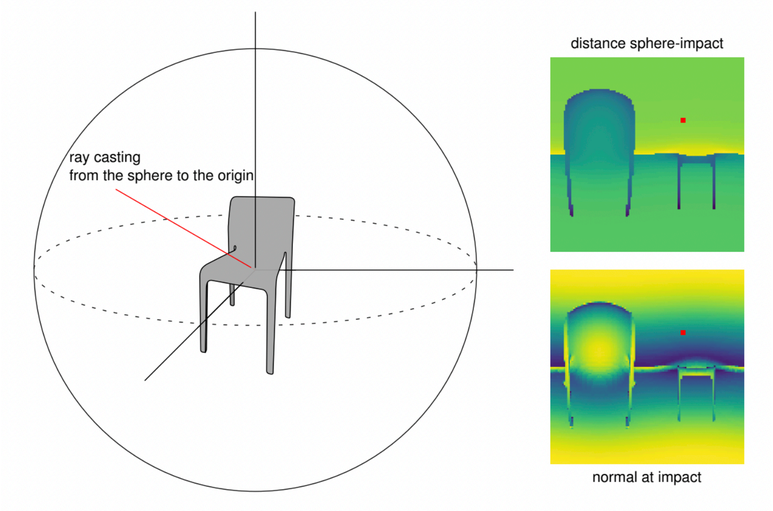
There is a matrix representation ρ such that $\rho(g)\vec{a} = \vec{b}$

D. Pinchon and P. E. Hoggan, J. Pays A: Math. Theor. (2007)

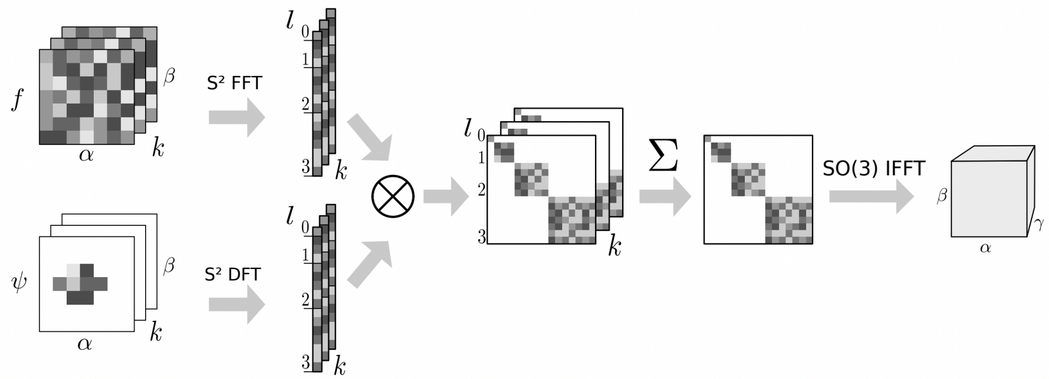
Examples

Spherical CNNs

$$f \star g = \mathcal{F}^{-1}\left[\mathcal{F}[f]\mathcal{F}[g]\right]$$



Linear convolutions



Spherical convolutions (or correlations)

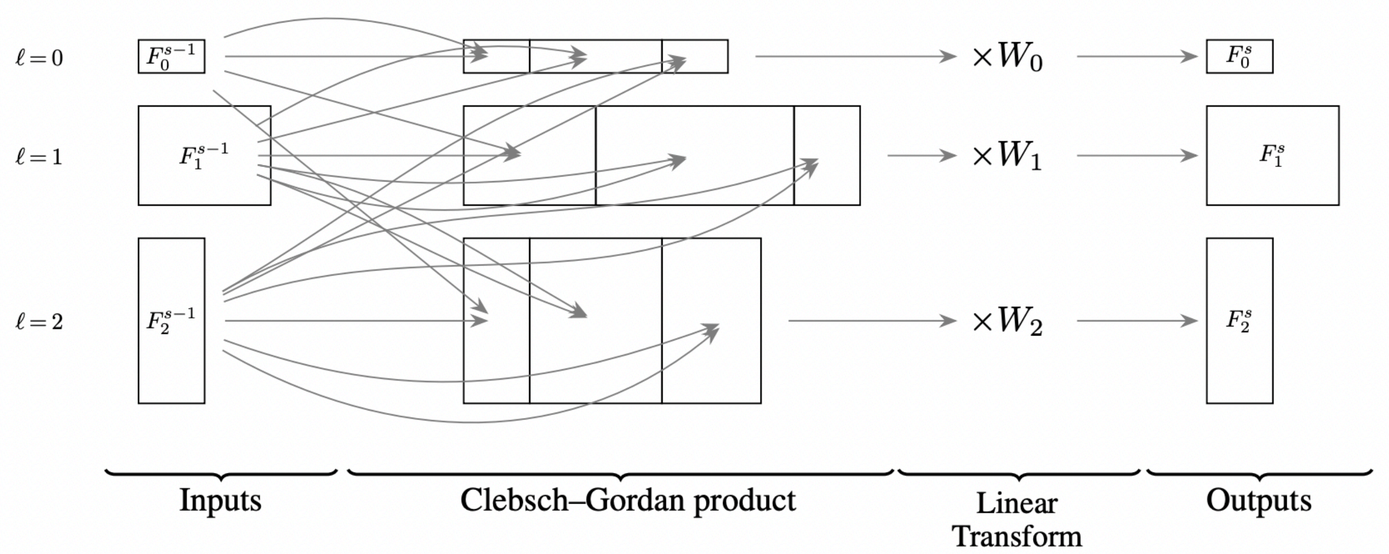
- Operates on **spherical data** (geospatial, VR…) instead of grids
- Generalizes linear convolutions to **convolutions on sphere**
- Equivariant to **rotation**

T. Cohen, M. Geiger, J. Koehler and M. Welling, ICLR (2018)

Examples

Clebsch-Gordan Networks

$$\hat{g}_\ell = C^T_{\ell_1,\ell_2,\ell}[\hat{f}_{\ell_1} \otimes \hat{f}_{\ell_2}]$$

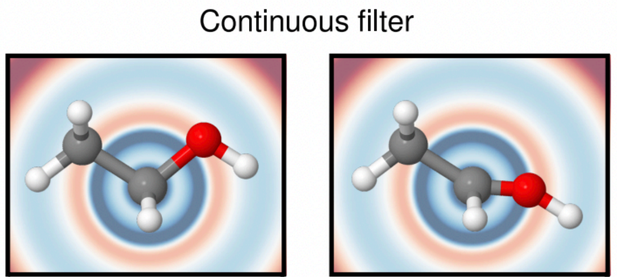


- Spherical CNNs require **expensive** Fourier transforms every layer
- Instead, work directly in generalised Fourier space. This makes linear layers **simple**. But what about nonlinearities?
- Take tensor product of activation vectors and sum by **Clebsch-Gordan coefficients**. This is nonlinear, but also **equivariant to rotation**

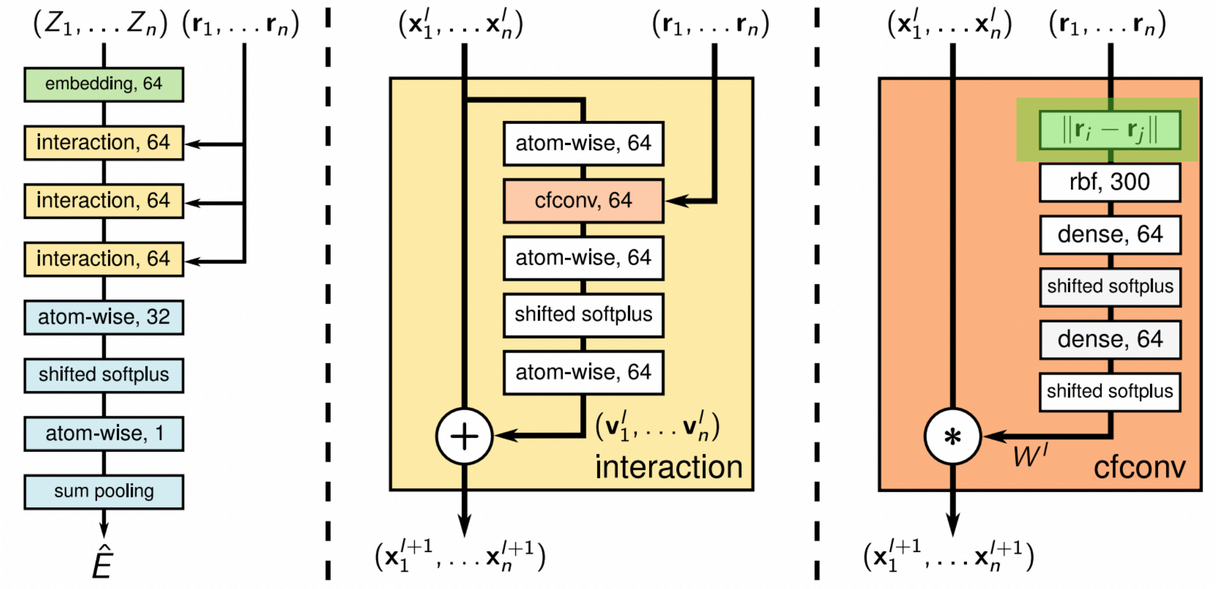
R. Kondor, Z. Lin and S. Trivedi, NeurIPS (2018)

Examples

SchNet



$$\mathbf{x}_i^{\ell+1} = \sum_j \mathbf{x}_j^{\ell} \circ W^{\ell}(|\mathbf{r}_i - \mathbf{r}_j|)$$

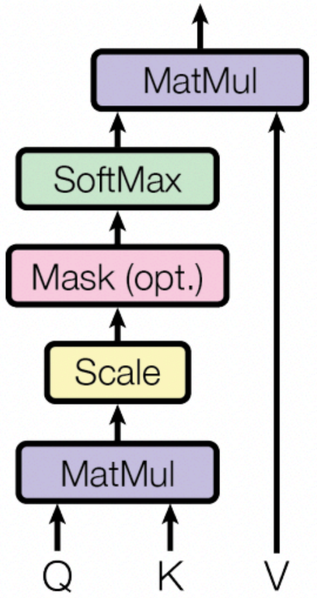


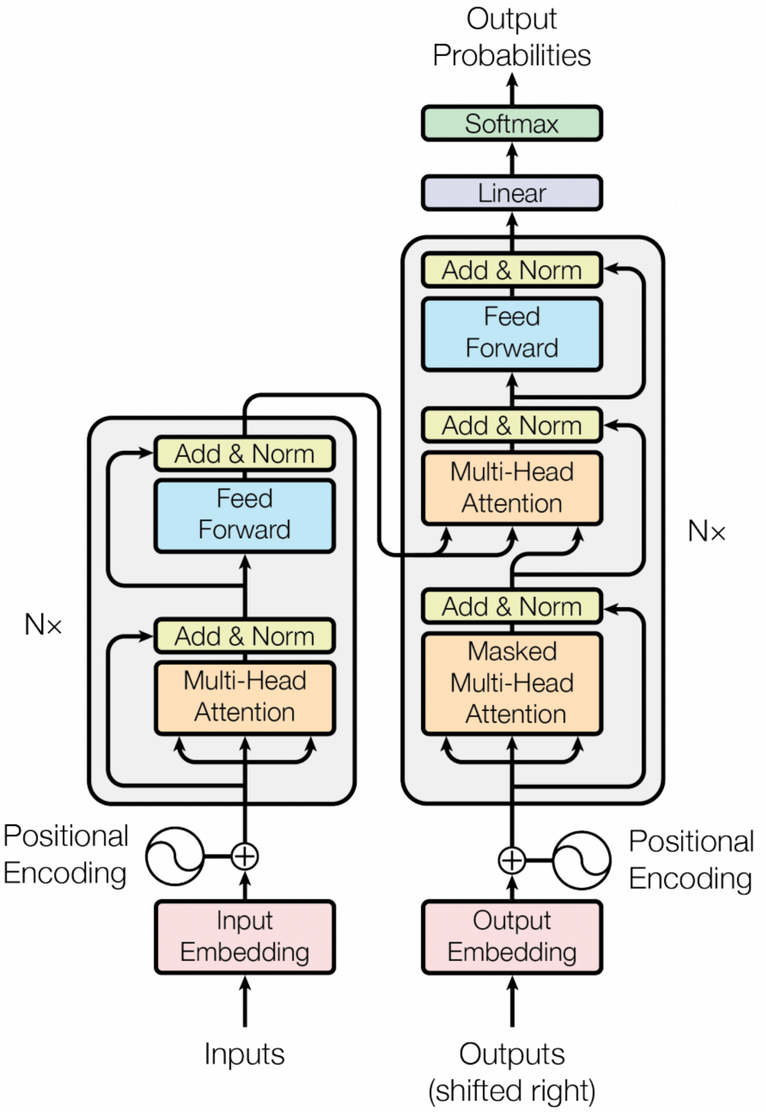
- Designed to be useful as a machine-learned **molecular potential** (next week!)
- Only interaction is through continuous-filter "convolution" that sums **radially-symmetric function** of pairs of atoms
- Equivariant to **permutation**, but **only *invariant* to rotation**!

K. T. Schütt, P.-J. Kindermans, H. E. Sauceda, S. Chmiela, A. Tkatchenko and K.-R. Müller, NeurIPS (2017)

Examples
Transformers

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$





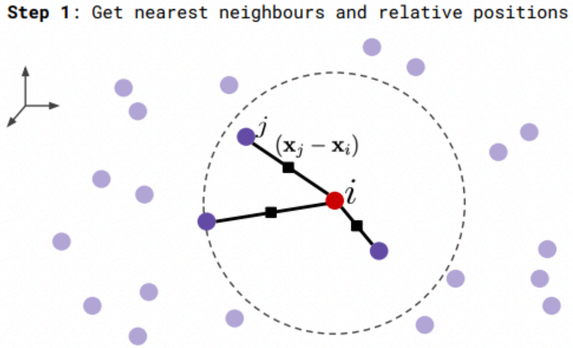
- Stacked layers of **self-attention** and MLPs
- Self-attention is a **differentiable hash map** on all pairs of inputs
- Preposterously successful - maybe it's all you need?
- Equivariant to **permutation**

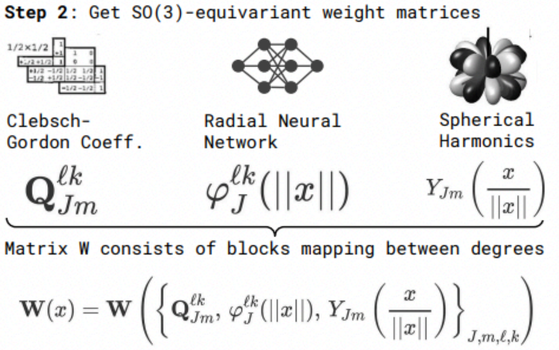
A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez and Ł. Kaiser, NeurIPS (2017)

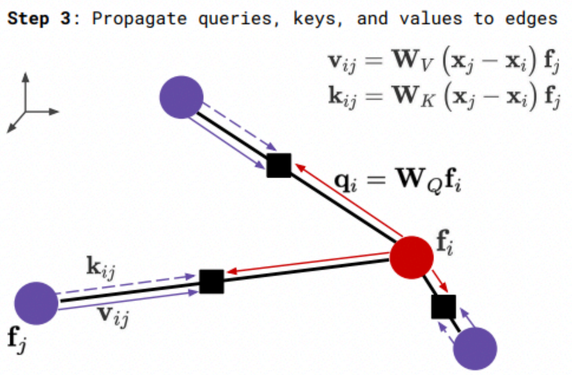
Examples

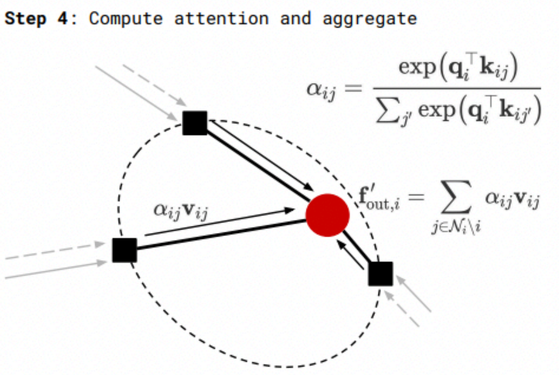
SE(3)-Equivariant Transformers

- Can we put all invariances together in one model? **Yes we can!**
- Very useful for machine learning of **molecular potentials** (next week!)
- Equivariant to **rotation, translation and permutation**









F. B. Fuchs, D. E. Worrall, V. Fischer and M. Welling, NeurIPS (2020)

Summary

- Neural networks combine **structured linear operations** with simple **nonlinear operations** repeatedly to approximate complex high-dimensional functions
- Much of the power of deep learning comes from building in **invariances** and **equivariances** into neural networks which simplify learning considerably
- The theory of **groups** and **representations** gives a unified picture for understanding the mathematics of equivariances and suggests recipes for designing neural network layers
- Depending on the application, neural networks can incorporate symmetry to **translation**, **rotation**, **permutation**, or even all of the above

II. Deep Learning for Electronic Structure

Ingredients for a Wavefunction

- Can we just throw an MLP at the Schrödinger equation, minimize the energy by VMC, and call it a day? **No!**
- What are the **minimal constraints** that a wavefunction must satisfy? How would we build the **most general** neural network that satisfies these constraints?

First vs. Second Quantization

Two alternative representations of a wavefunction

First quantization $(\{\mathbf{r}_1, \sigma_1\}, \{\mathbf{r}_2, \sigma_2\}, \ldots, \{\mathbf{r}_n, \sigma_n\})$

- An electron configuration is an ordered list of all electron positions and spins
- The wavefunction must treat all electrons indistinguishably - cannot depend on the order

Second quantization $|0, 0, 1, 0, 1, 1, \ldots, 1, 0\rangle$

- An electron configuration is a binary vector enumerating which states are occupied or unoccupied. The total number of 1s adds up to the number of electrons
- Different electrons are automatically indistinguishable, no need to constrain the wavefunction

Neural Network Quantum States

Deep VMC in Second Quantization

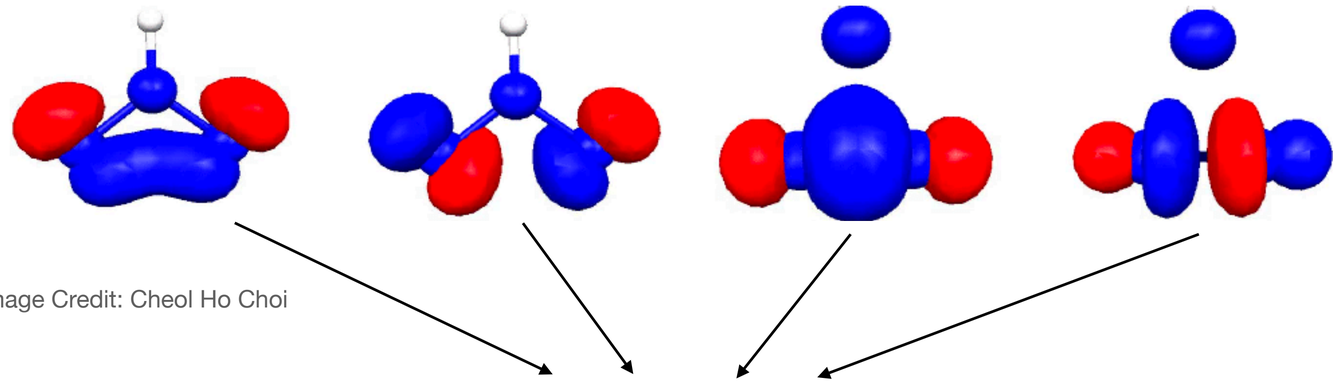


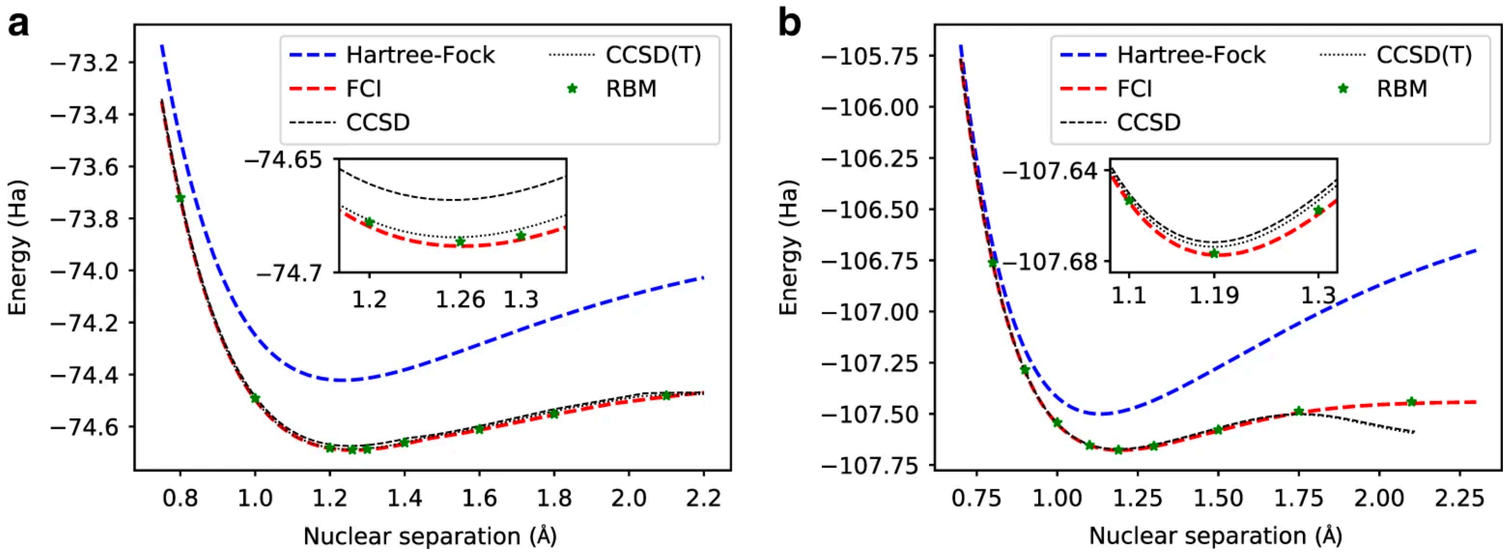
Image Credit: Cheol Ho Choi

- Originally developed for spin systems, but extended to electrons
- Enumerate a set of states the electrons could be in, form a binary vector, input the vector to a neural network, train by VMC

G. Carleo and M. Troyer, Science (2017)
K. Choo, A. Mezzacapo and G. Carleo, Nat. Comm. (2020)

Neural Network Quantum States

Deep VMC in Second Quantization



- Advantages: neural network can be very **flexible**, as states are indistinguishable

- Disadvantages: have to enumerate possible states, will always suffer from **finite basis set errors**

G. Carleo and M. Troyer, Science (2017)
K. Choo, A. Mezzacapo and G. Carleo, Nat. Comm. (2020)

Fermions and Antisymmetry

$$\Psi(\mathbf{r}_1, \mathbf{r}_1, \ldots, \mathbf{r}_n) = 0$$

$$\Psi(\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_n) = -\Psi(\mathbf{r}_2, \mathbf{r}_1, \ldots, \mathbf{r}_n)$$

- **Pauli exclusion principle**: no two fermions (including electrons) may be in the same state at the same time
- The wavefunction must be **zero** when two electrons have the same state
- Equivalently, the wavefunction must be **antisymmetric**

Determinants

$$\det \begin{pmatrix} a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nn} \end{pmatrix} = \sum_{\sigma} (-1)^{|\sigma|} \prod_{i} a_{i\sigma_i}$$

Leibniz formula

$$\det \begin{pmatrix} ca_{11} & \dots & ca_{1n} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nn} \end{pmatrix} = c\det \begin{pmatrix} a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nn} \end{pmatrix}$$

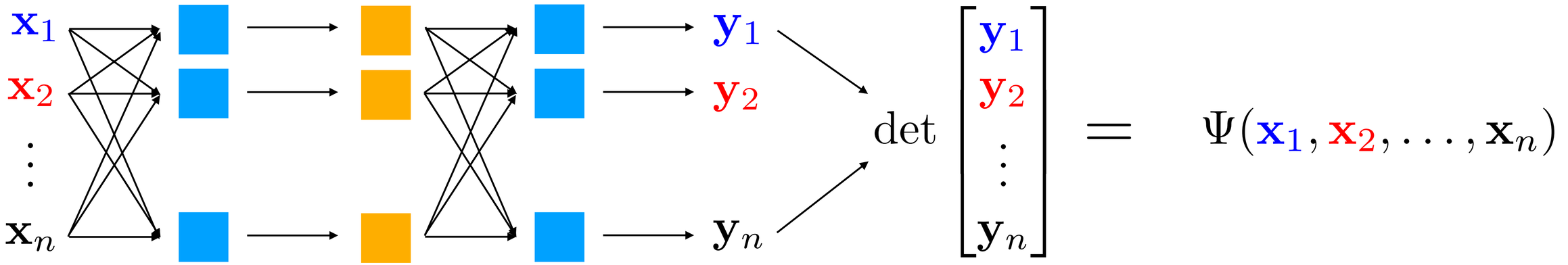
Multilinearity

$$\det \begin{pmatrix} a_{11} & \dots & a_{1n} \\ a_{21} & \dots & a_{2n} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nn} \end{pmatrix} = -\det \begin{pmatrix} a_{21} & \dots & a_{2n} \\ a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nn} \end{pmatrix}$$

Antisymmetry

- The **unique** function such that:
  - It is a **multilinear** function of its rows and columns
  - It is **antisymmetric** under exchange of two rows or columns
  - It maps the identity to 1
- Leibniz formula is a sum over **O(n!)** permutations - but it can be computed in **O(n^3)** time!

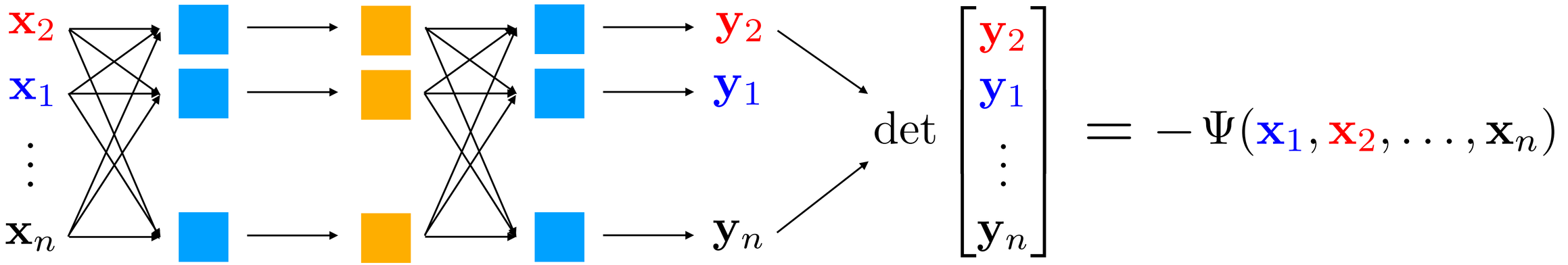
Determinants



- Take a function of electron positions which is **permutation equivariant**
- Take each output vector to be **one row** of a matrix, take its determinant
- The result is an **antisymmetric function**
- In theory, single determinant can represent any antisymmetric function (but only if discontinuous functions are allowed)

M. Hutter, arXiv (2020)

Determinants



- Take a function of electron positions which is **permutation equivariant**
- Take each output vector to be **one row** of a matrix, take its determinant
- The result is an **antisymmetric function**
- In theory, single determinant can represent any antisymmetric function (but only if discontinuous functions are allowed)

M. Hutter, arXiv (2020)

Alternatives to Determinants?

DeepWF

$$\Psi(\mathbf{r}_1, \ldots, \mathbf{r}_n) = \prod_{i<j} \phi(\mathbf{r}_i, \mathbf{r}_j)$$

$$\phi(\mathbf{r}_i, \mathbf{r}_j) = -\phi(\mathbf{r}_j, \mathbf{r}_i)$$

- Determinant evaluation scales as **O(n^3)** - much better than O(n!), but not ideal
- There are simpler ways to build an antisymmetric function that scale as **O(n^2)**…but empirically, **don't work well**.

Alternatives to Determinants?

DeepWF

System	DeepWF $[a.u.]$	Benchmark $[a.u.]$	Rel. Diff
H_2	-1.1738	-1.1741	0.26%
He	-2.9036	-2.9029	-0.02%
LiH	-7.8732	-8.0243	1.88%
Be	-14.6141	-14.6190	0.03%
B	-24.2124	-24.6006	1.58%
H_{10}	-5.5685	-5.6655	1.71%

J. Han, L. Zhang and W. E, J. Comp. Phys. (2019)

Atom	ψ''_{pair}	ψ'_{pair}	Exact
Li	-7.4782	-7.4781	-7.47806032
Be	-14.6673	-14.6664	-14.66736
B	-24.5602	-24.4475	-24.65391
C	-37.3531	-37.2785	-37.8450
N	-53.1855	-53.0626	-54.5892

T. Pang, S. Yan, M. Lin, arXiv (2022)

- Determinant evaluation scales as **O(n³)** - much better than O(n!), but not ideal
- There are simpler ways to build an antisymmetric function that scale as **O(n²)**…but empirically, **don't work well**.

Alternatives to Determinants?

DeepWF

System	DeepWF $[a.u.]$	Benchmark $[a.u.]$	Rel. Diff
H_2	-1.1738	-1.1741	0.26%
He	-2.9036	-2.9029	-0.02%
LiH	-7.8732	-8.0243	1.88%
Be	-14.6141	-14.6190	0.03%
B	-24.2124	-24.6006	1.58%
H_{10}	-5.5685	-5.6655	1.71%

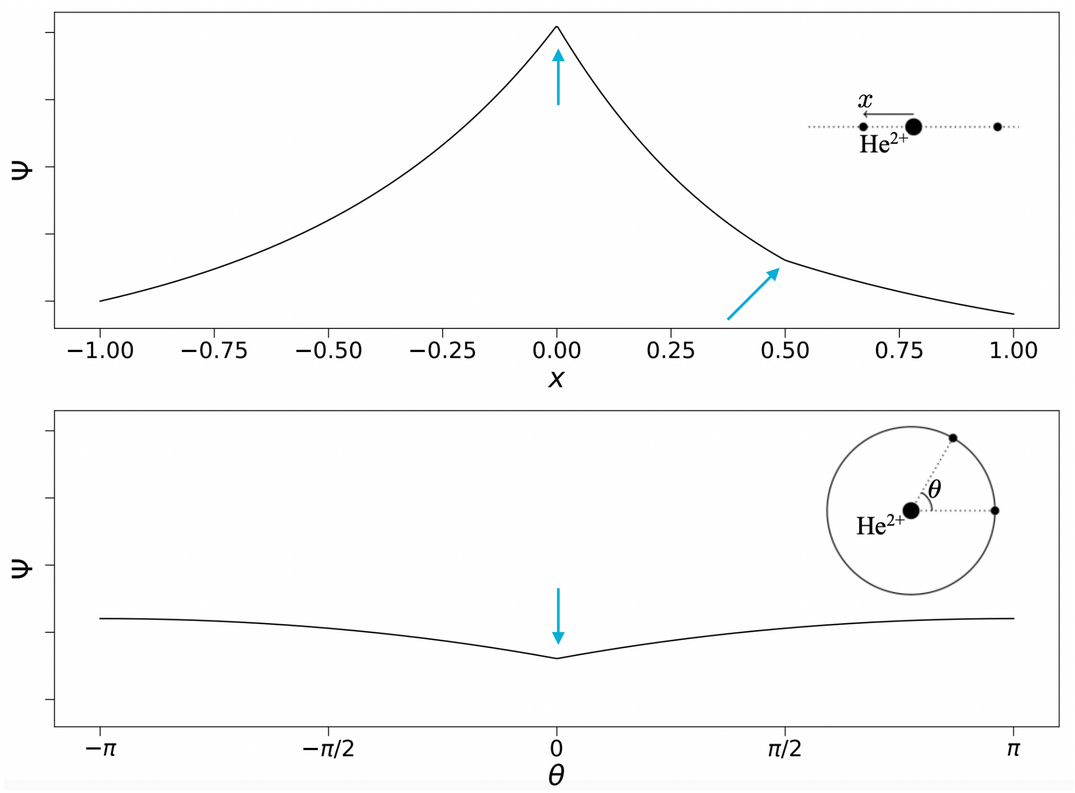
J. Han, L. Zhang and W. E, J. Comp. Phys. (2019)

Atom	ψ''_{pair}	ψ'_{pair}	Exact
Li	-7.4782	-7.4781	-7.47806032
Be	-14.6673	-14.6664	-14.66736
B	-24.5602	-24.4475	-24.65391
C	-37.3531	-37.2785	-37.8450
N	-53.1855	-53.0626	-54.5892

T. Pang, S. Yan, M. Lin, arXiv (2022)

- Determinant evaluation scales as **O(n³)** - much better than O(n!), but not ideal
- There are simpler ways to build an antisymmetric function that scale as **O(n²)**…but empirically, **don't work well**…sometimes **worse than mean-field**

Cusp Conditions



$$V(\mathbf{r}_{ij}) = \frac{Z_i Z_j}{|\mathbf{r}_{ij}|}$$

$$\lim_{r_{ij}\to 0} \int_{|\mathbf{r}_{ij}|=r_{ij}} d\mu \frac{\partial \Psi}{\partial r_{ij}} = -Z_i Z_j \Psi(\mathbf{r}_{ij} = 0)$$

- Coulomb potential **diverges** when particles overlap. Kinetic energy must **exactly cancel** divergent potential energy
- Leads to **exact conditions** on gradient of the wavefunction at a cusp. Must be **non-differentiable**

Cusp Conditions

(1) $$\Psi(\mathbf{r}_1, \ldots, \mathbf{r}_n) = \mathbf{U}(\mathbf{r}_1, \ldots, \mathbf{r}_n) \prod_{i<j} \exp\left(-Z_i Z_j |\mathbf{r}_i - \mathbf{r}_j|\right)$$

Smooth antisymmetric function — Jastrow Factor

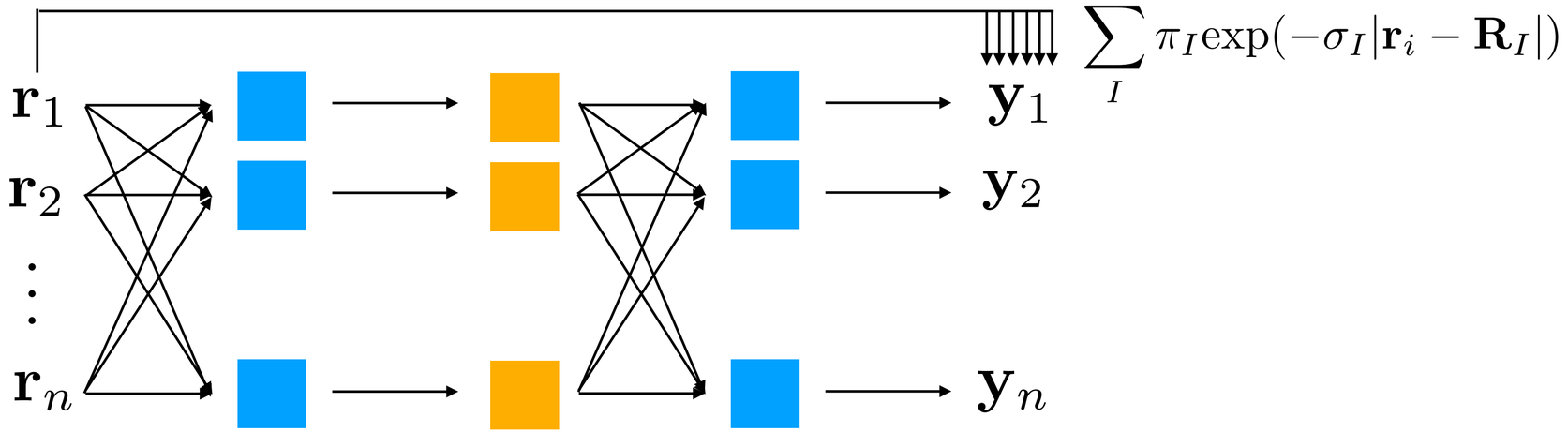
(2) $$\Psi(\mathbf{r}_1, \ldots, \mathbf{r}_n) = \mathbf{U}(\mathbf{r}_1, \ldots, \mathbf{r}_n, |\mathbf{r}_1 - \mathbf{r}_2|, \ldots, |\mathbf{r}_{n-1} - \mathbf{r}_n|)$$

Smooth function with non-smooth inputs

- Two approaches:
  - Include a **multiplicative term** (Jastrow factor) with exact cusp conditions
  - Include an input feature that is non-differentiable at the cusp and let the neural network **learn** the exact condition
- Surprisingly, the **second works better** for the FermiNet!

Boundary Conditions

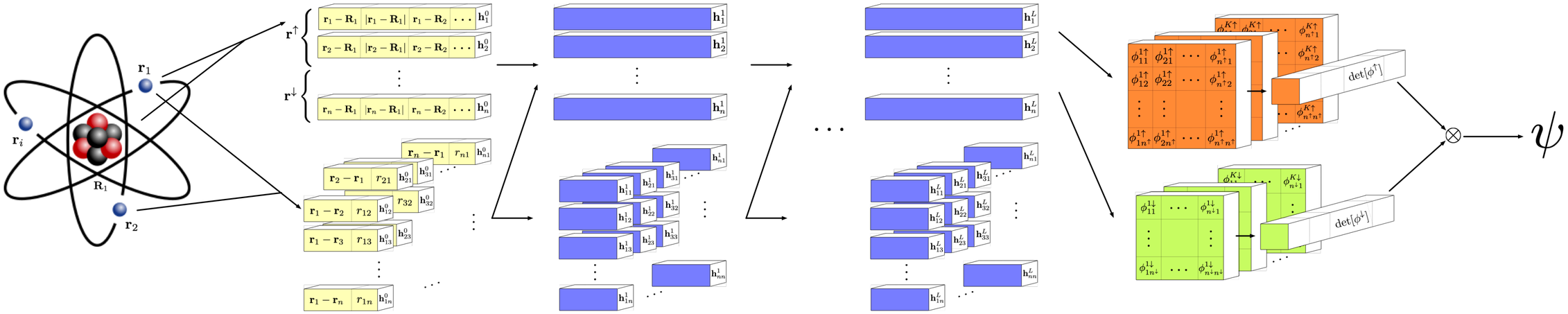
$$\Psi(\mathbf{r}_1, \ldots, \mathbf{r}_n) \to 0 \quad \text{as} \quad \mathbf{r}_i \to \infty$$



- For molecules, the wavefunction has **open boundary conditions**
- But neural networks don’t go to zero as inputs grow
- Solution: add a **multiplicative term** that decays exponentially to each neural network output

Fermionic Neural Networks

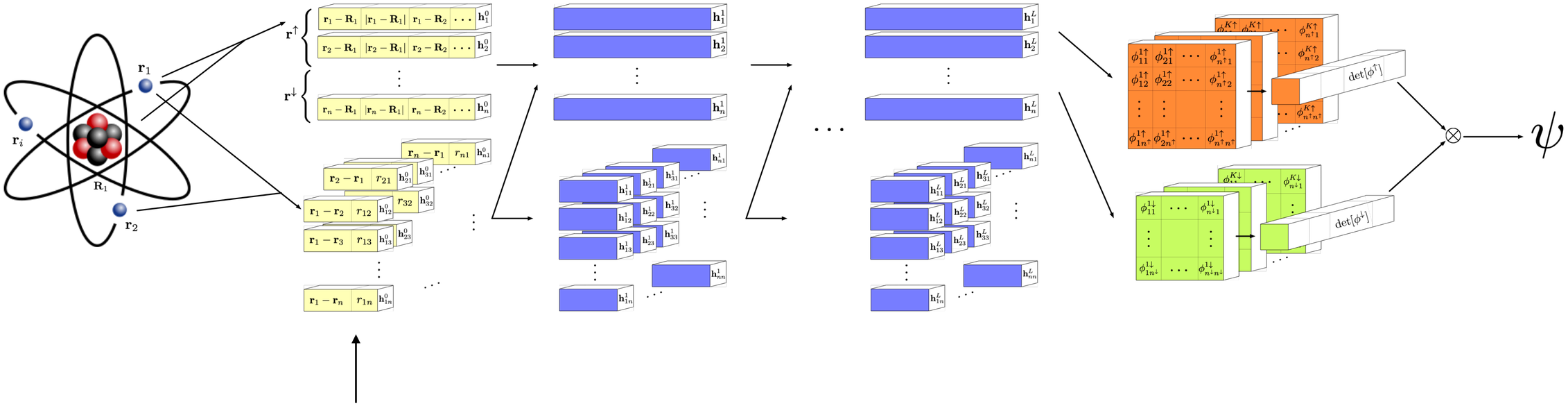
Architecture



D. Pfau, J. S. Spencer, A. G. G. Matthews and W. M. C. Foulkes, PRResearch (2020)

Fermionic Neural Networks

Architecture



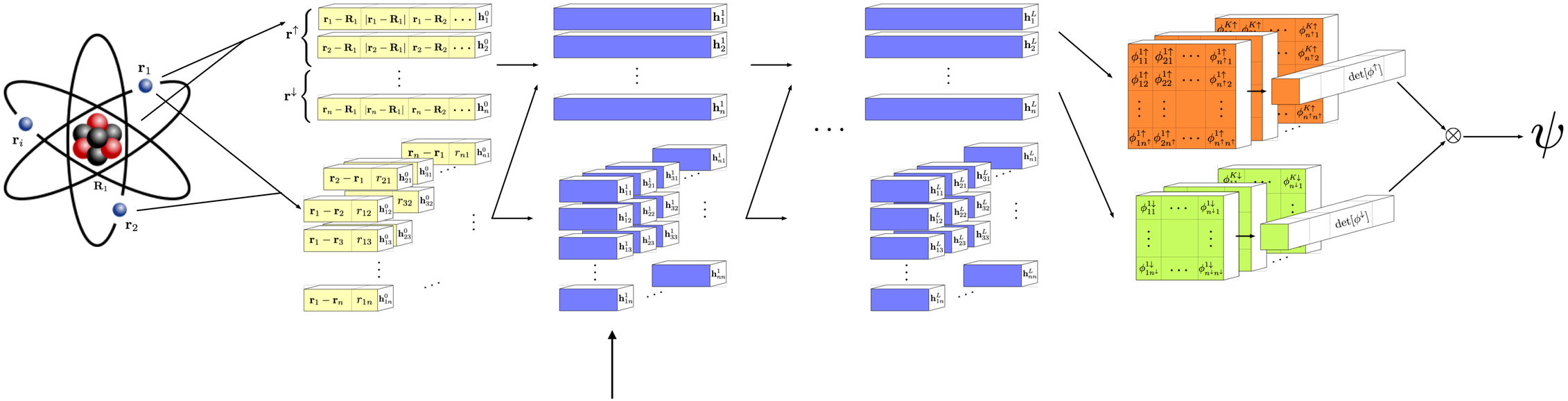
Input features:

Electron **positions** relative to the nuclei, **differences** between all pairs of electrons, and **distances** to the nuclei and between electrons (non-smooth at the cusps)

D. Pfau, J. S. Spencer, A. G. G. Matthews and W. M. C. Foulkes, PRResearch (2020)

Fermionic Neural Networks

Architecture



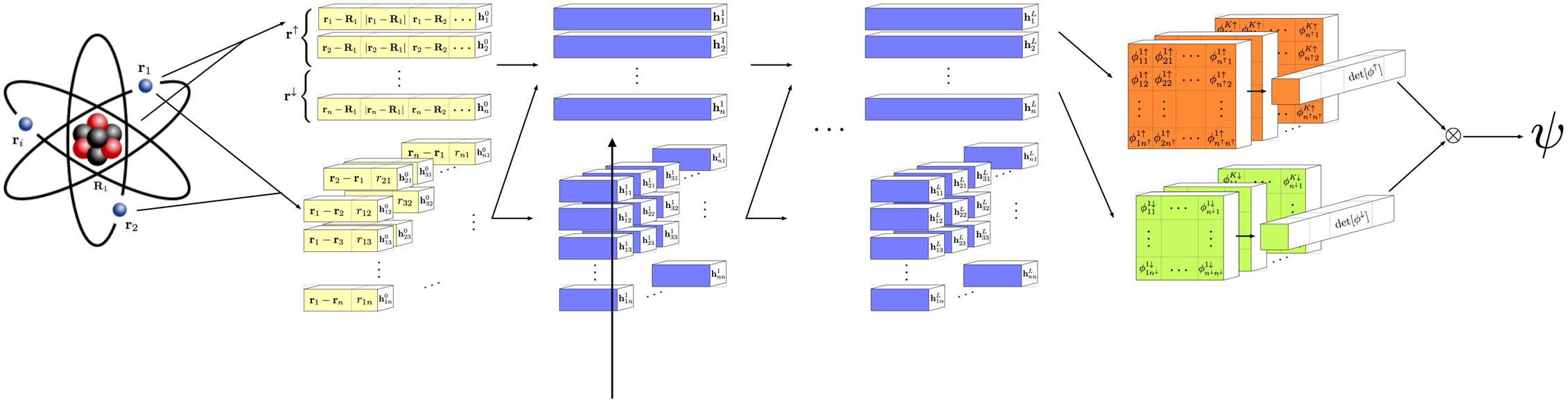
Two-electron stream:

Independent MLPs. Feeds information **into** the one electron stream, but not the other way around

D. Pfau, J. S. Spencer, A. G. G. Matthews and W. M. C. Foulkes, PRResearch (2020)

Fermionic Neural Networks

Architecture



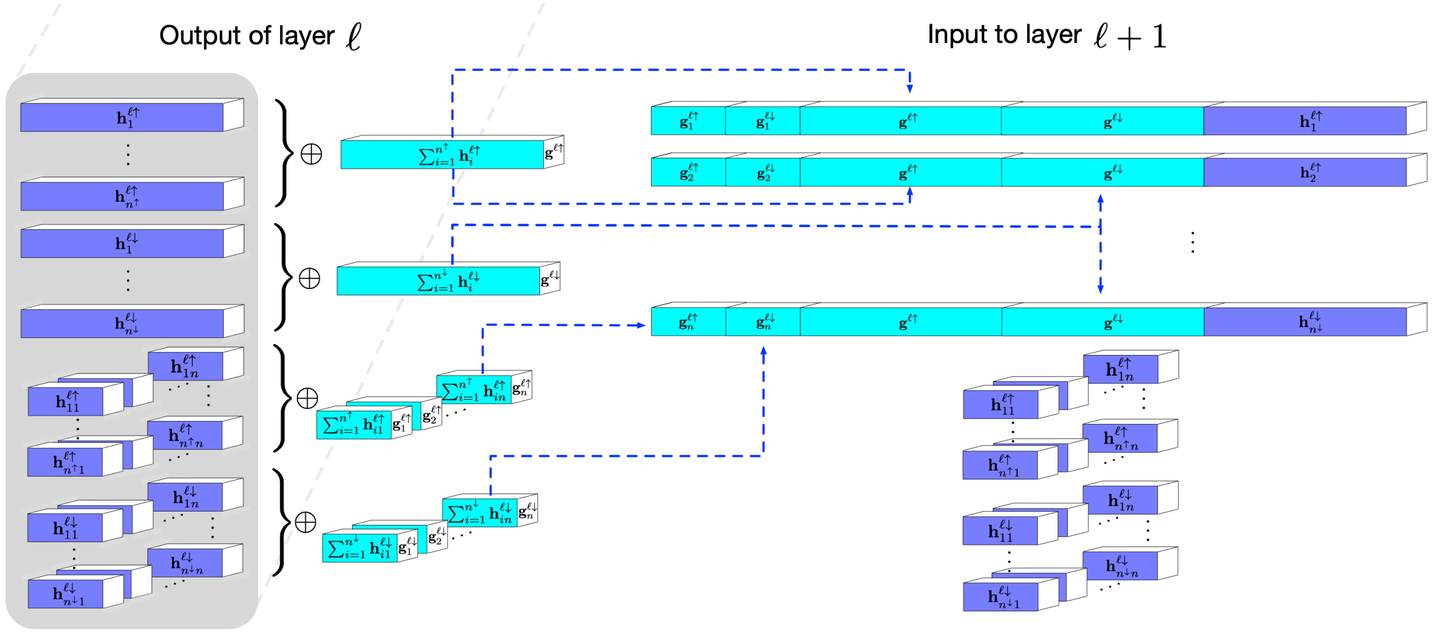
One-electron stream:

Integrates information both from other one-electron vectors and two-electron vectors

D. Pfau, J. S. Spencer, A. G. G. Matthews and W. M. C. Foulkes, PRResearch (2020)

Fermionic Neural Networks

Architecture



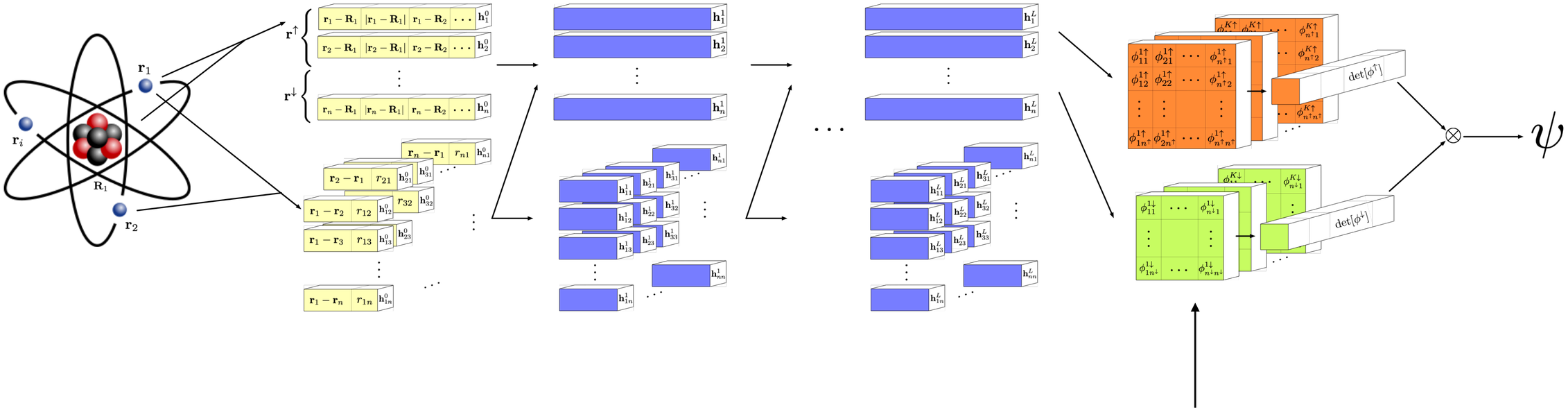
One-electron stream:

Split vectors from one- and two-electron streams into **spin-up** and **spin-down**, sum together to form **invariant** representation, concatenate to form **equivariant** inputs to linear layer

D. Pfau, J. S. Spencer, A. G. G. Matthews and W. M. C. Foulkes, PRResearch (2020)

Fermionic Neural Networks

Architecture



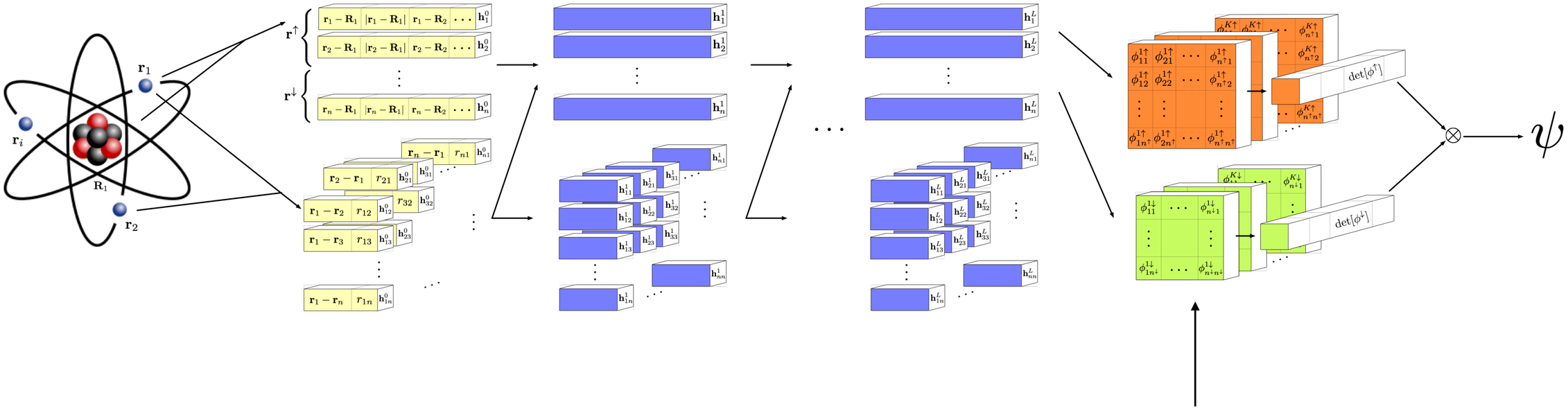
Envelopes (not pictured):

Enforce **open boundary conditions**

D. Pfau, J. S. Spencer, A. G. G. Matthews and W. M. C. Foulkes, PRResearch (2020)

Fermionic Neural Networks

Architecture



Determinants:

Sum over **multiple** determinants (O(10)) to increase representational capacity

D. Pfau, J. S. Spencer, A. G. G. Matthews and W. M. C. Foulkes, PRResearch (2020)

Fermionic Neural Networks

Optimization

$$\min_{\Psi} \frac{\langle \Psi^* \hat{H} \Psi \rangle}{\langle \Psi^2 \rangle} = \min_{\Psi} \mathbb{E}_{\mathbf{r} \sim \Psi^2} \left[\Psi^{-1}(\mathbf{r}) \hat{H} \Psi(\mathbf{r}) \right]$$

Objective (Energy)

$$E_L(\mathbf{r}) = \Psi^{-1}(\mathbf{r}) \hat{H} \Psi(\mathbf{r})$$

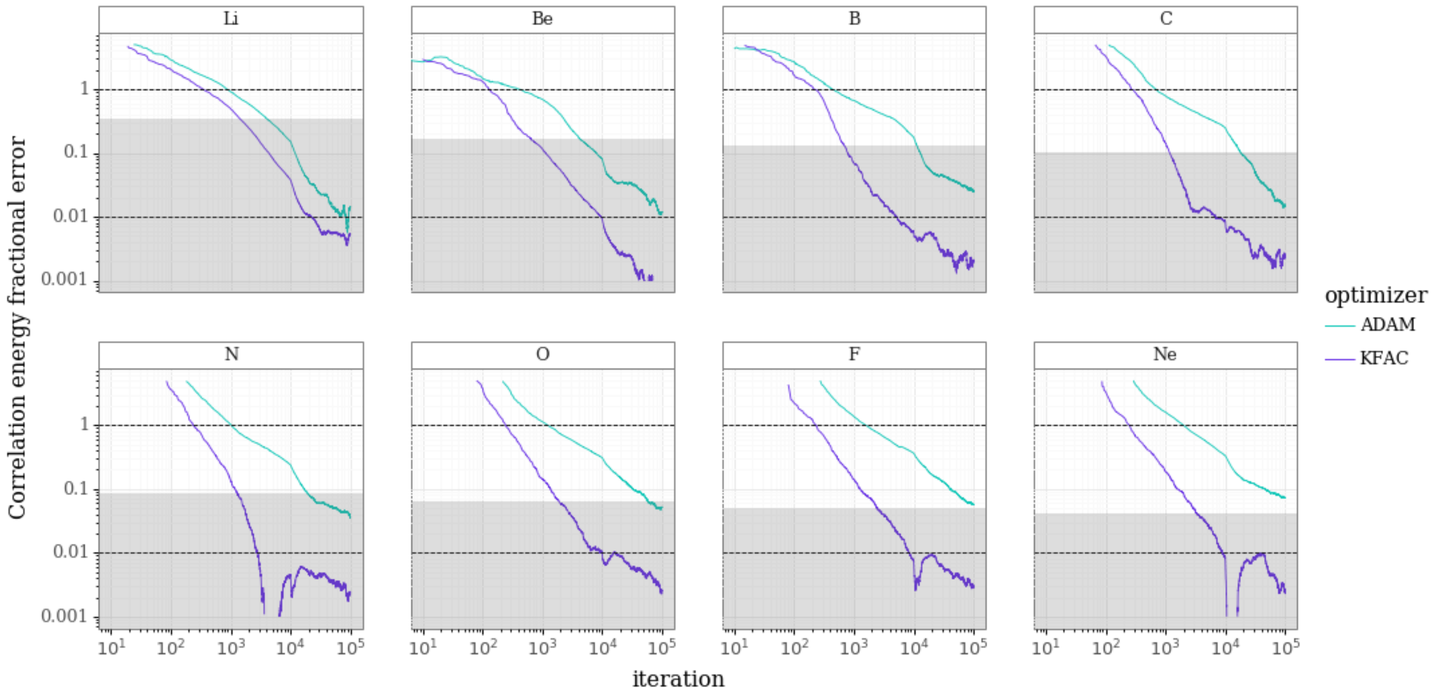
$$2\mathbb{E}_{\mathbf{r} \sim \Psi^2} \left[(E_L(\mathbf{r}) - \mathbb{E}[E_L]) \nabla_\theta \log |\Psi(\mathbf{r})| \right]$$

Gradient

D. Pfau, J. S. Spencer, A. G. G. Matthews and W. M. C. Foulkes, PRResearch (2020)

Fermionic Neural Networks

Optimization



$$\theta^{t+1} = \theta^t - \alpha \mathcal{F}^{-1} \nabla_\theta \mathcal{L}$$

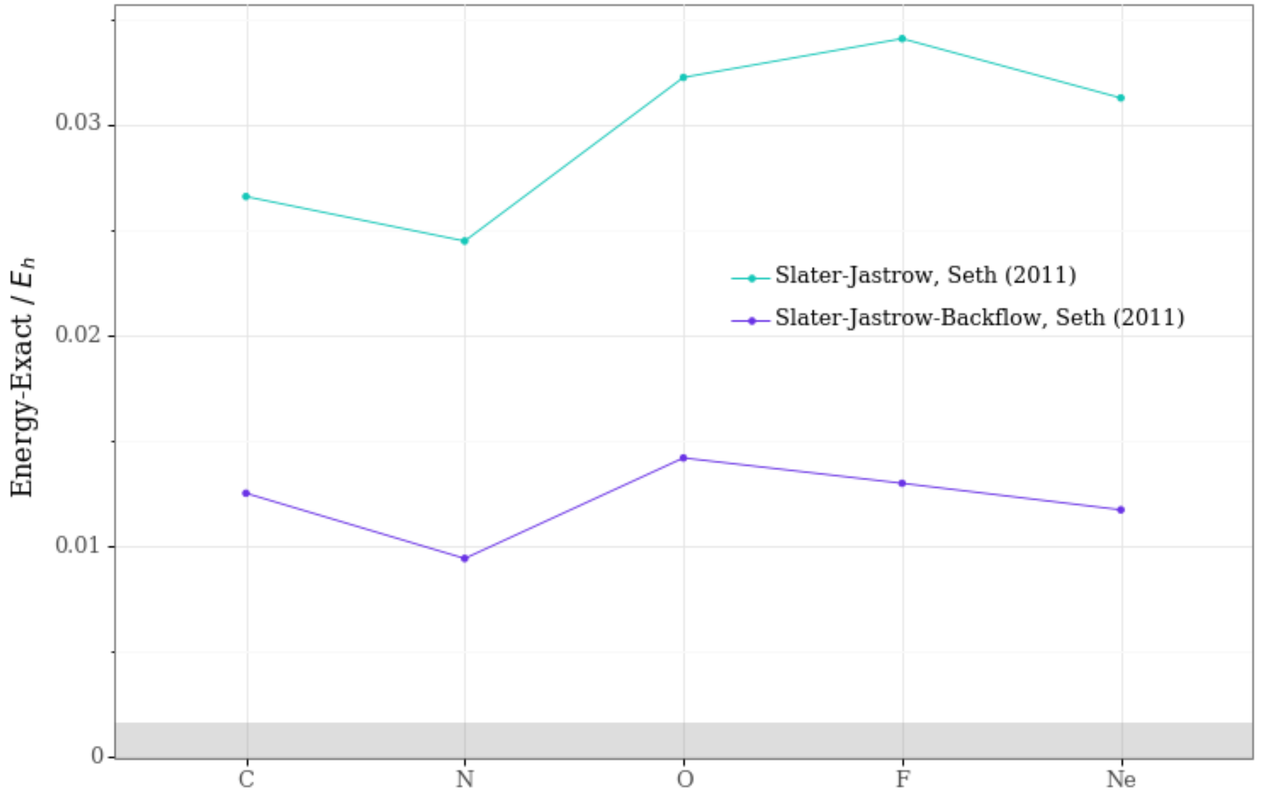
$$\mathcal{F}_{ij} = 4\mathbb{E}_{\Psi^2}\left[\frac{\partial \log|\Psi(\mathbf{r})|}{\partial \theta_i}\frac{\partial \log|\Psi(\mathbf{r})|}{\partial \theta_j}\right]$$

- Can’t just plug into standard optimizers like ADAM.
- Need something stronger: **Kronecker-Factored Approximate Curvature**, approximate second-order method

D. Pfau, J. S. Spencer, A. G. G. Matthews and W. M. C. Foulkes, PRResearch (2020)

Fermionic Neural Networks

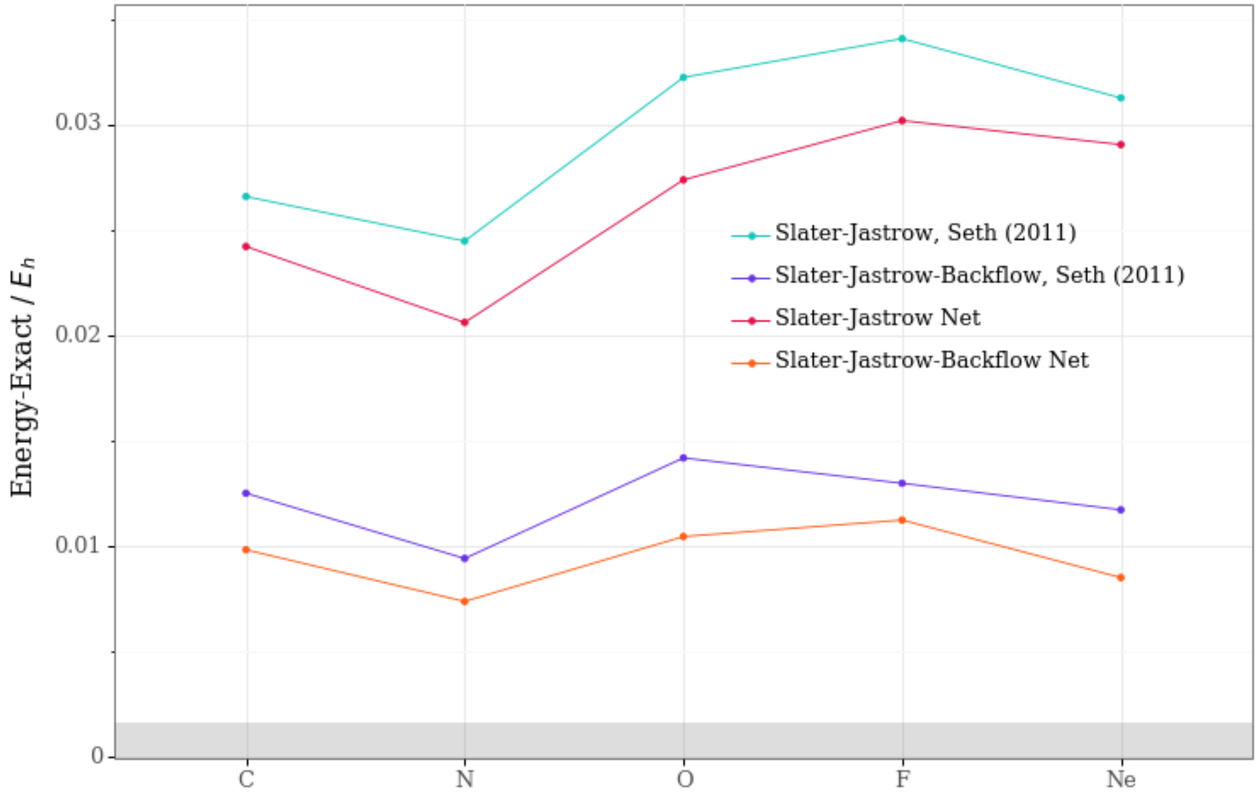
Results



D. Pfau, J. S. Spencer, A. G. G. Matthews and W. M. C. Foulkes, PRResearch (2020)

Fermionic Neural Networks

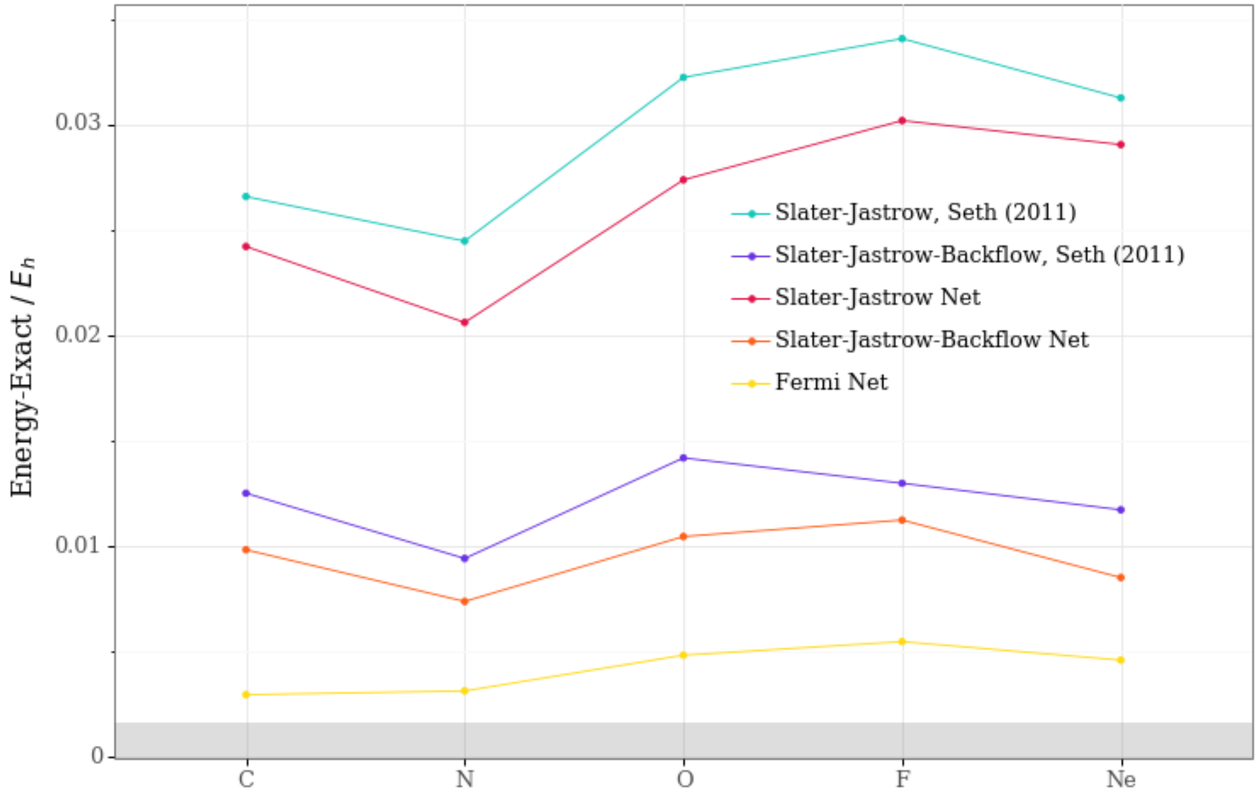
Results



D. Pfau, J. S. Spencer, A. G. G. Matthews and W. M. C. Foulkes, PRResearch (2020)

Fermionic Neural Networks

Results

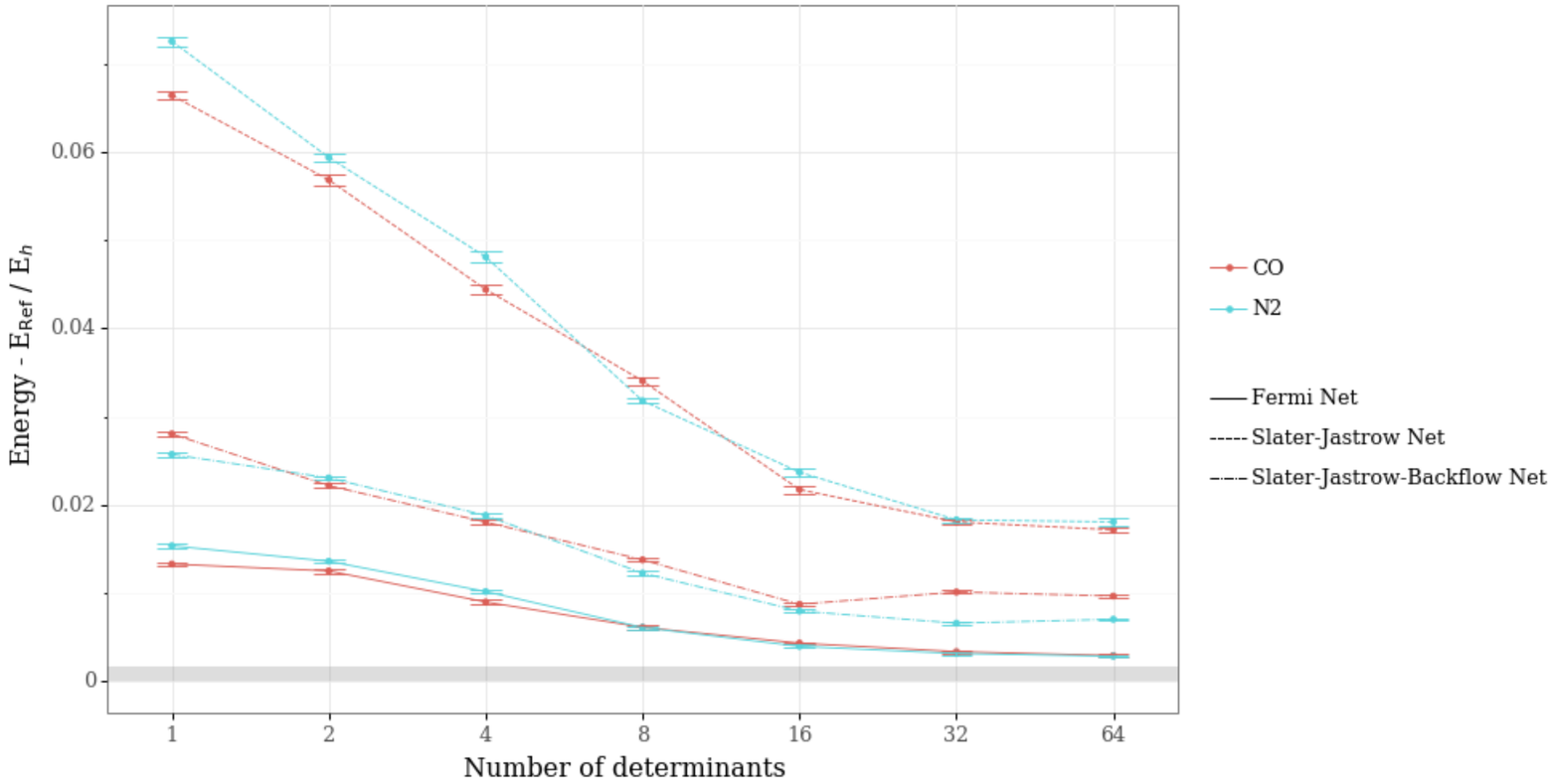


- FermiNet is **substantially** more accurate, even with one determinant!

D. Pfau, J. S. Spencer, A. G. G. Matthews and W. M. C. Foulkes, PRResearch (2020)

Fermionic Neural Networks

Results

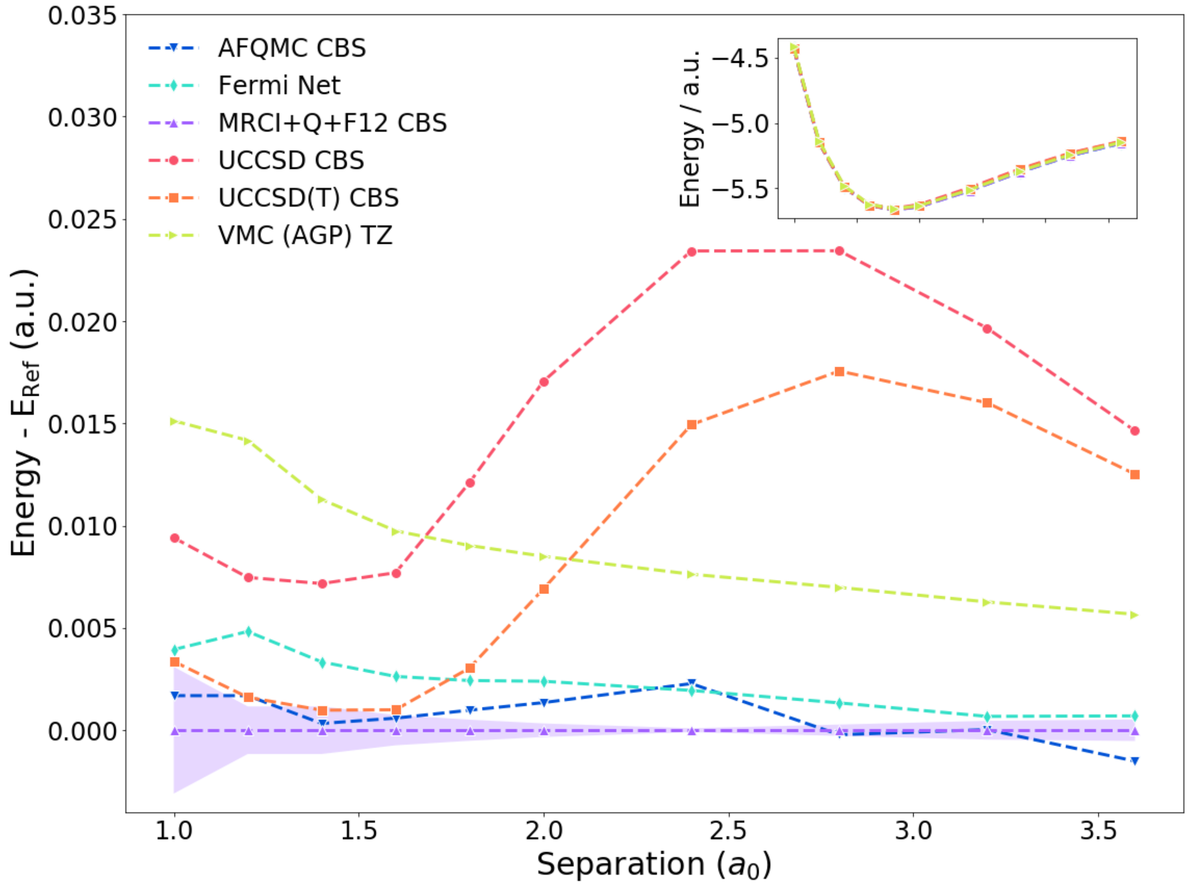


- Only need a few dozen determinants to reach high accuracy

D. Pfau, J. S. Spencer, A. G. G. Matthews and W. M. C. Foulkes, PRResearch (2020)

Fermionic Neural Networks

Results

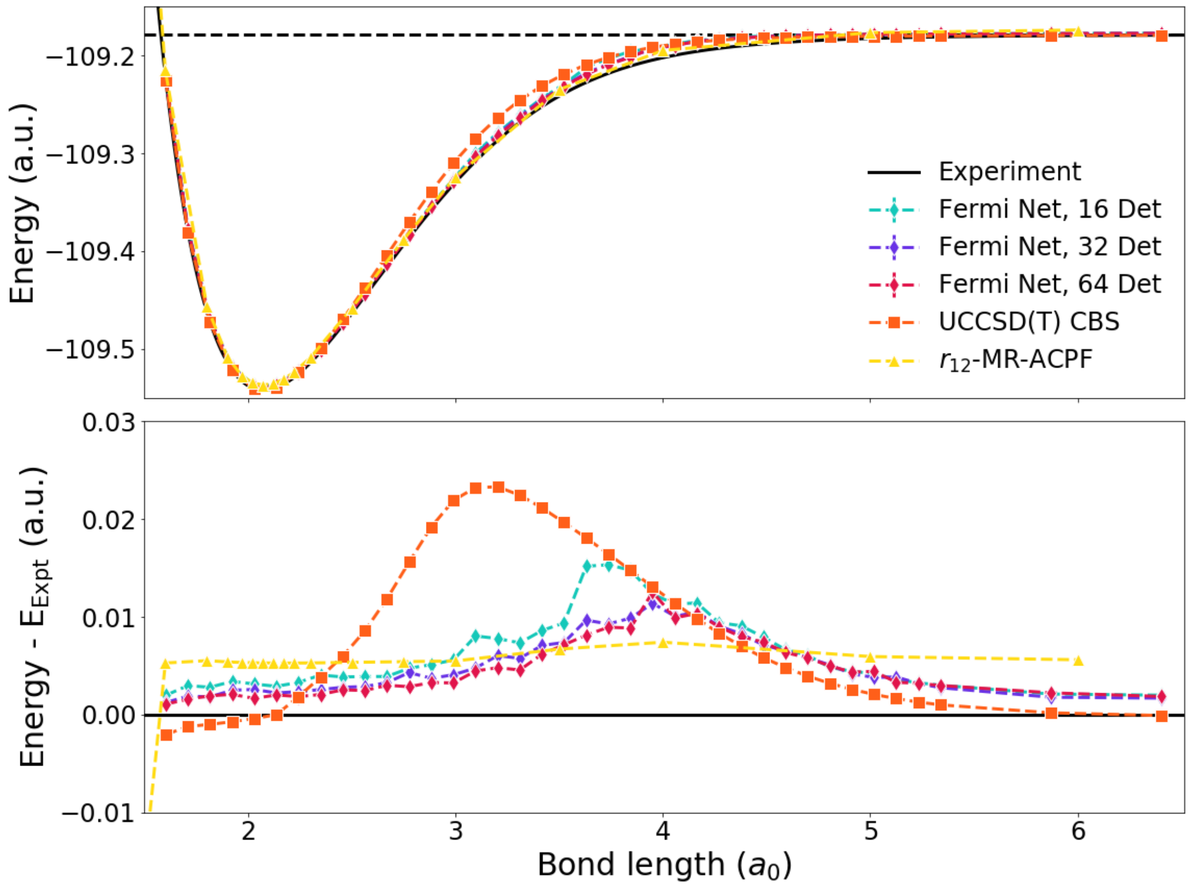


- Competitive with state-of-the-art methods on the hydrogen chain H_{10}

D. Pfau, J. S. Spencer, A. G. G. Matthews and W. M. C. Foulkes, PRResearch (2020)

Fermionic Neural Networks

Results

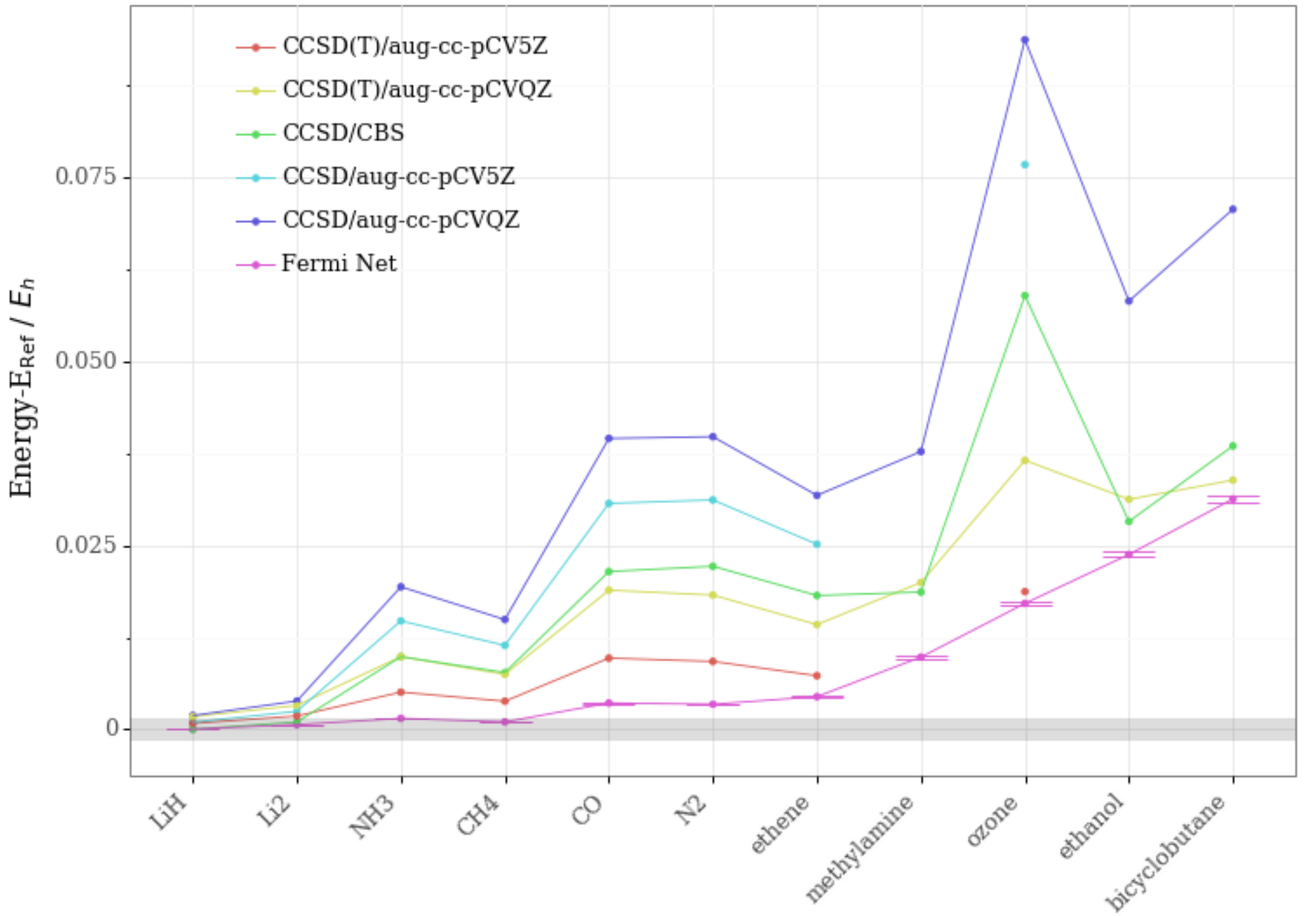


- Outperforms CCSD(T) on nitrogen dissociation

D. Pfau, J. S. Spencer, A. G. G. Matthews and W. M. C. Foulkes, PRResearch (2020)

Fermionic Neural Networks

Results

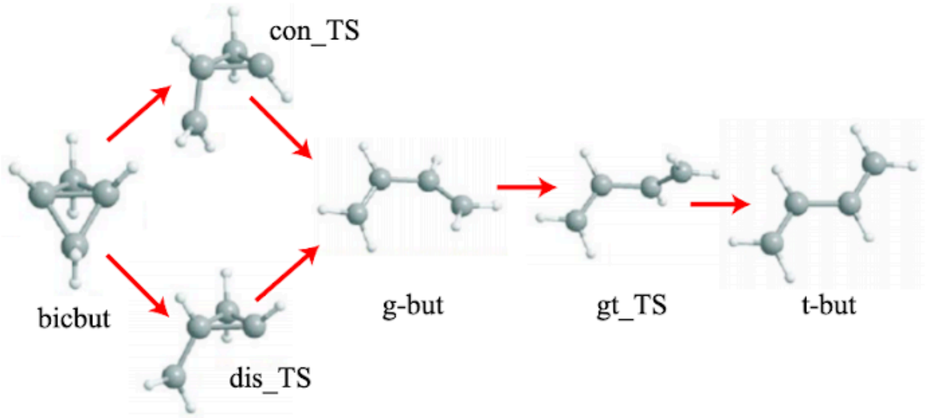


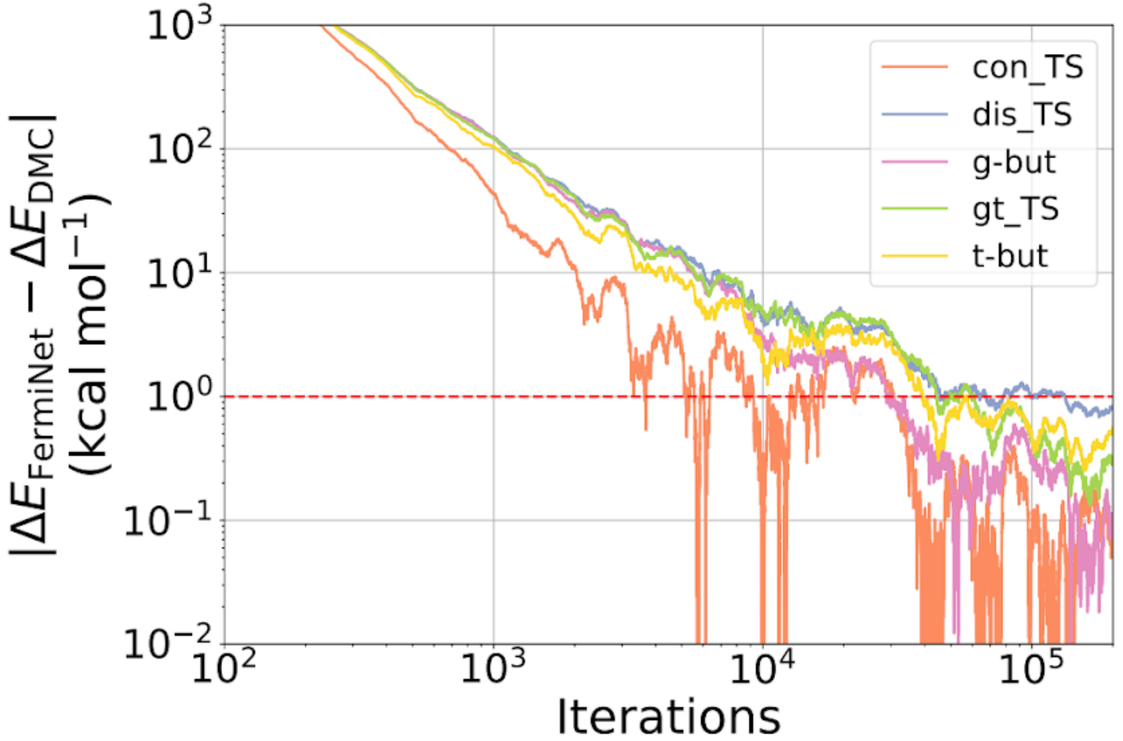
- Absolute energies on molecules better than any unextrapolated results

D. Pfau, J. S. Spencer, A. G. G. Matthews and W. M. C. Foulkes, PRResearch (2020)

Fermionic Neural Networks

Results





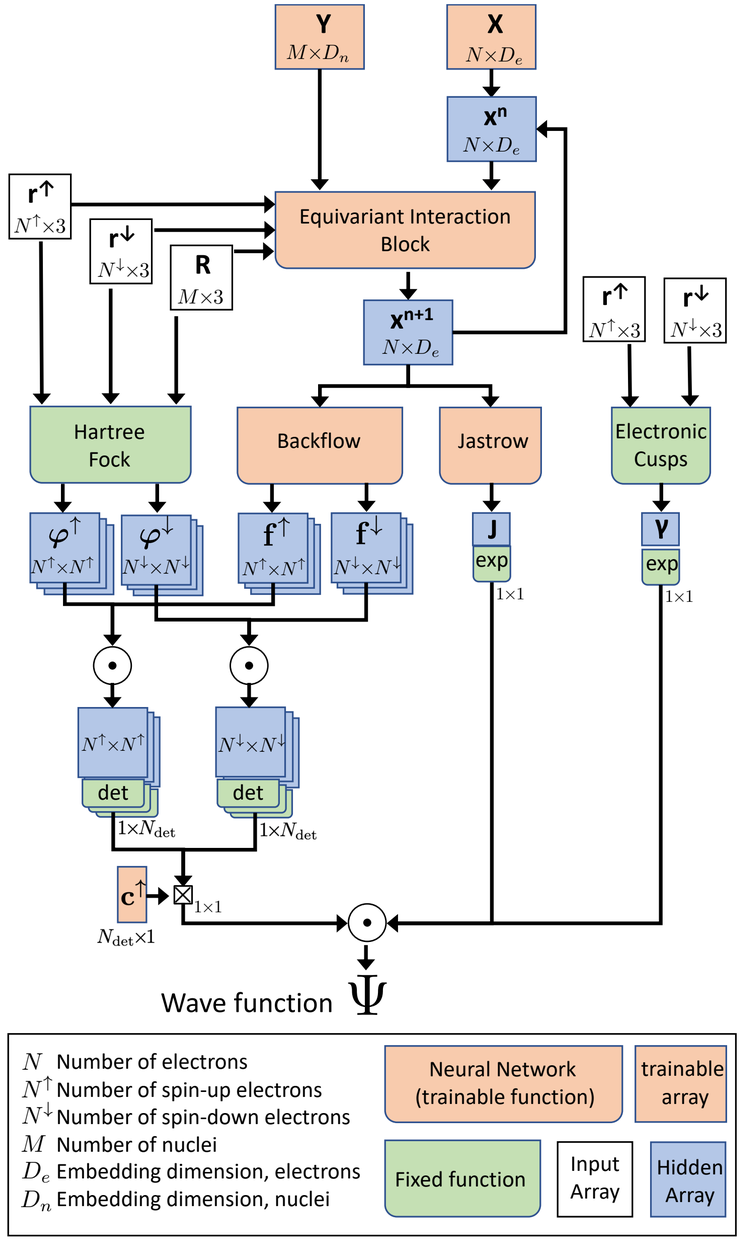
- Relative energies on molecules matches state-of-the-art methods

J. S. Spencer, D. Pfau, A. Botev and W.M.C. Foulkes, arXiv (2020)

PauliNet

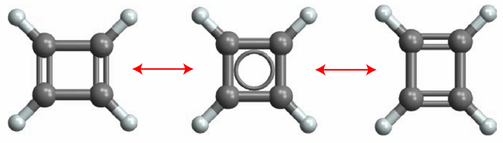
Architecture

- More **hard-coded** physics knowledge than FermiNet
  - Hartree-Fock solution is **built in** to network, rather than used to pretrain the network
  - Cusp conditions are given by **exact** Jastrow factor instead of learned
- Equivariant two-electron features inspired by **SchNet**
- Fewer parameters
- **Faster** to train, but **less** accurate

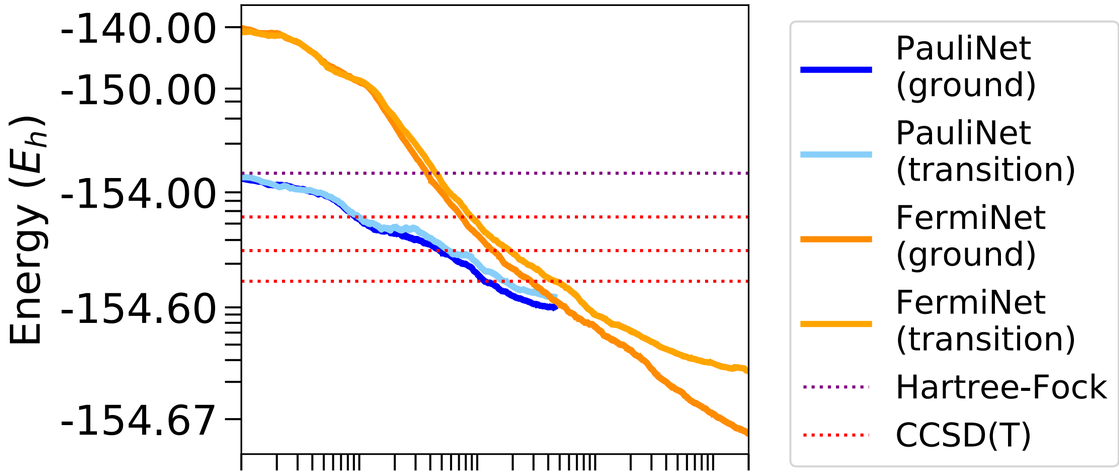


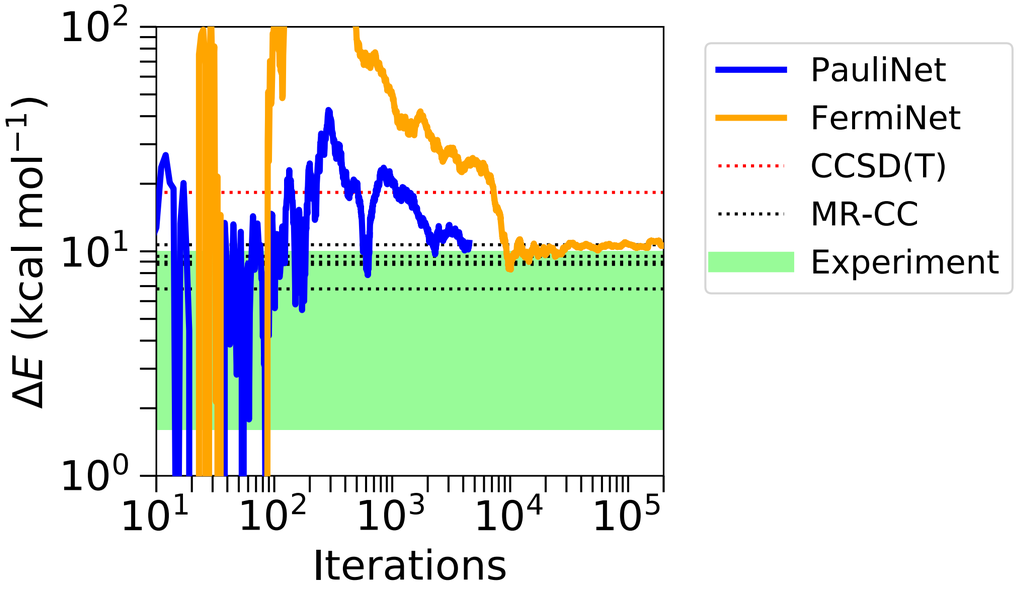
J. Hermann, Z. Schätzle and F. Noé, Nat. Chem. (2020)

PauliNet
Comparison



- Automerization of cyclobutadiene
  - Challenging multi-reference system
  - Failure case for CCSD(T)
- PauliNet is much **faster to converge**
- FermiNet is much more accurate in **total energy**
- Both methods are comparable in **relative energy**

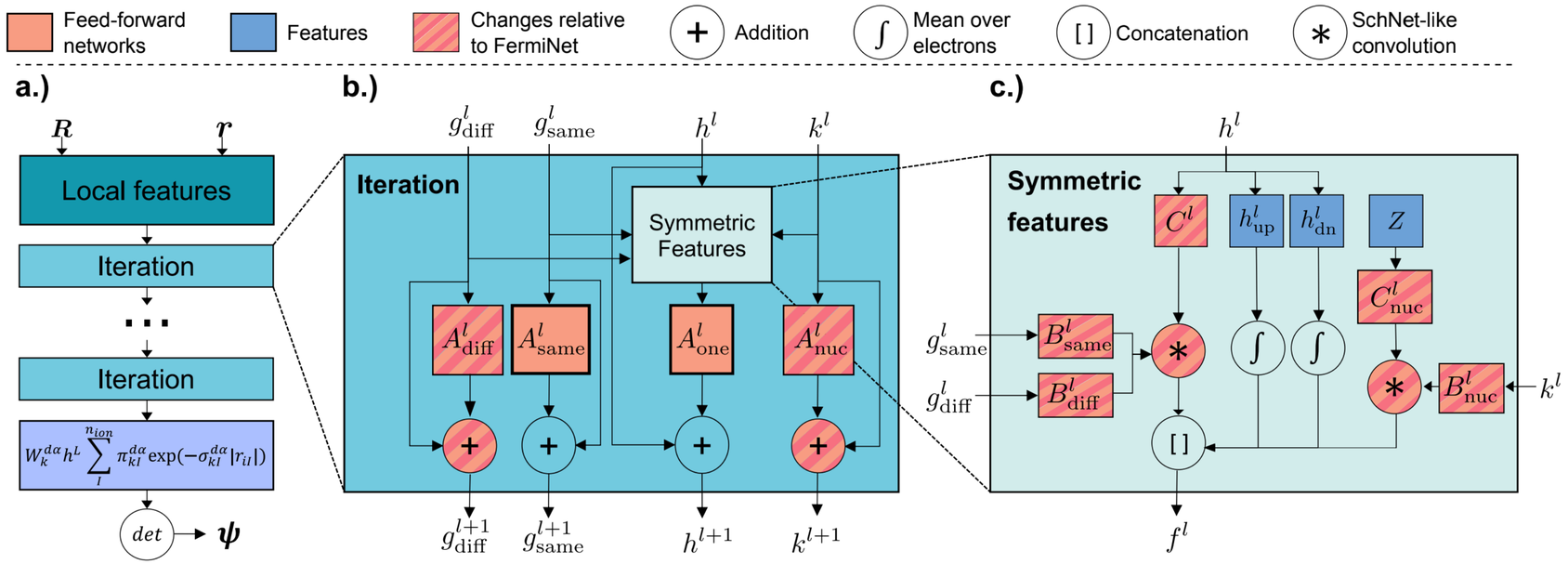




Adapted from J. S. Spencer, D. Pfau, A. Botev and W.M.C. Foulkes, arXiv (2020)

Hybrid Architecture

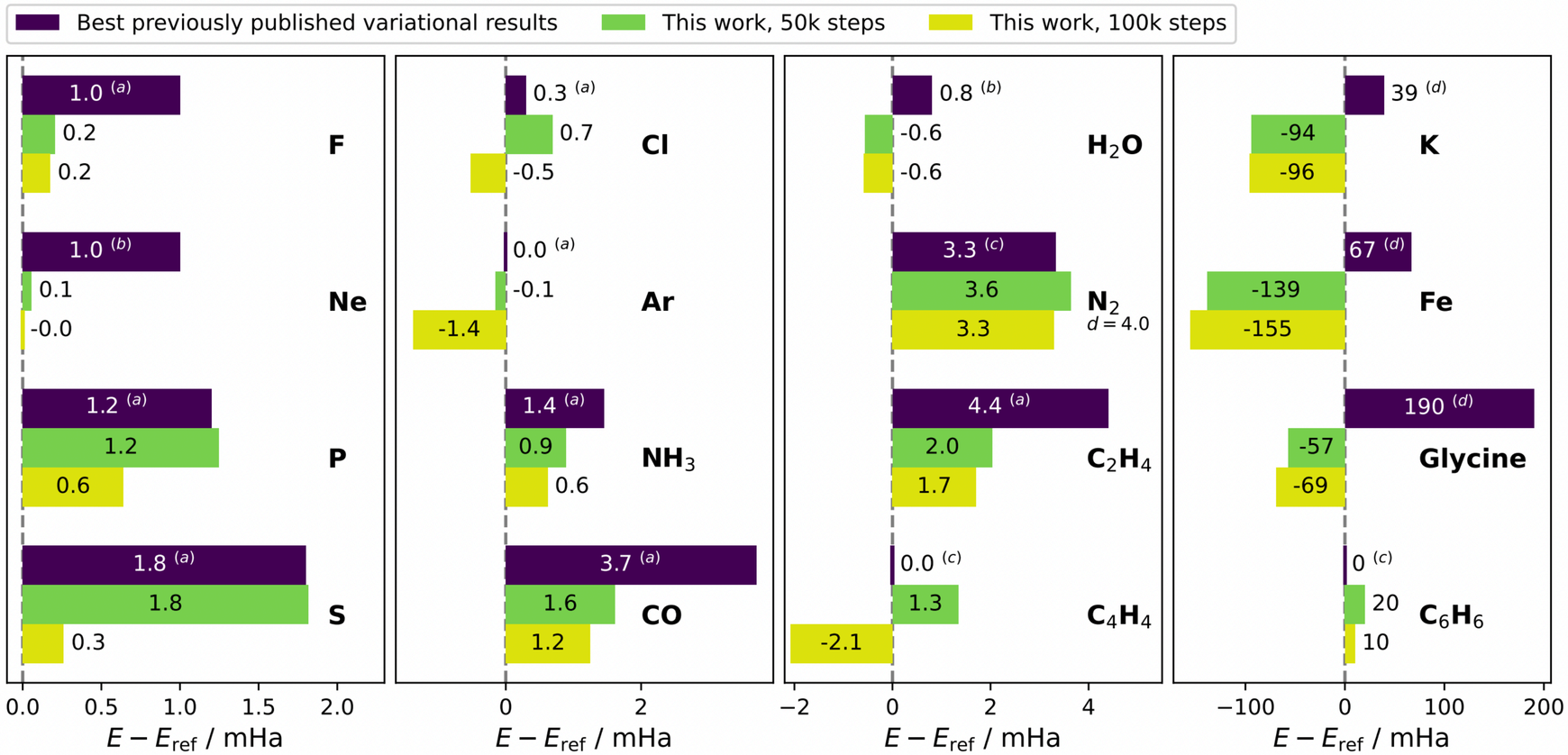
Combine FermiNet and PauliNet



- Split two-electron stream into spin **parallel** and **antiparallel** streams
- Integrate information into one-electron stream similar to **SchNet** (and PauliNet)
- Tune optimization and initialization

L. Gerard, M. Scherbela, P. Marquetand and P. Grohs, arXiv (2022)

Hybrid Architecture



- Result: cut number of iterations and batch size both in half, **>4x speedup** (and **lower energy** too!)

L. Gerard, M. Scherbela, P. Marquetand and P. Grohs, arXiv (2022)

Extensions & Improvements

Dense Determinants

$$\det[\psi^k_{i\sigma_i}] = \begin{vmatrix} \psi^k_{i\alpha}\left(\mathbf{r}^\alpha_j; \{\mathbf{r}^\alpha_{/j}\}, \{\mathbf{r}^\beta\}\right) & 0 \\ 0 & \psi^k_{i\beta}\left(\mathbf{r}^\beta_j; \{\mathbf{r}^\alpha\}, \{\mathbf{r}^\beta_{/j}\}\right) \end{vmatrix}$$

W. M. C. Foulkes, D. Pfau and J. S. Spencer, GitHub (2020)

Extensions & Improvements

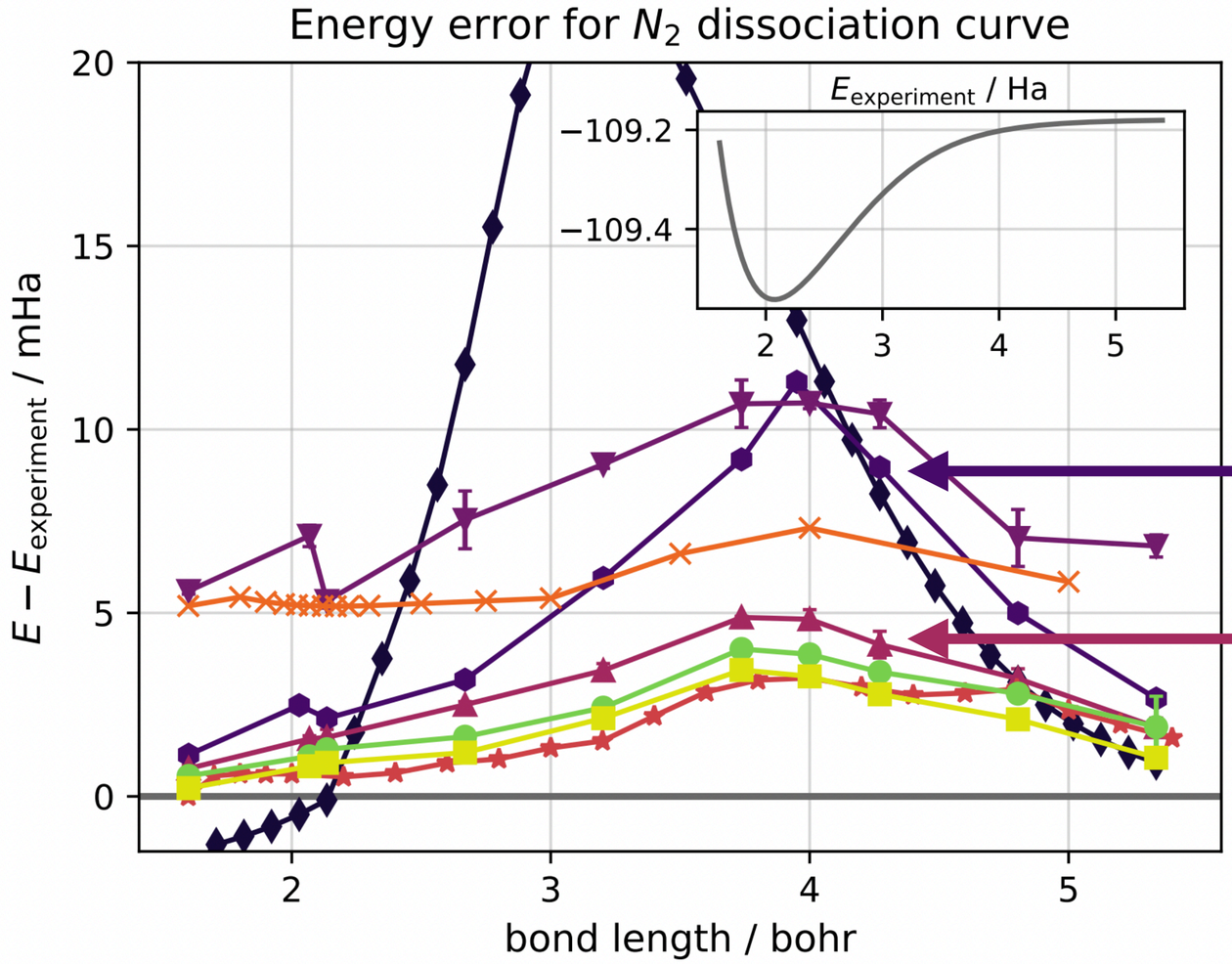
Dense Determinants

$$\det[\psi^k_{i\sigma_i}] = \begin{vmatrix} \psi^k_{i\alpha}\left(\mathbf{r}^\alpha_j; \{\mathbf{r}^\alpha_{/j}\}, \{\mathbf{r}^\beta\}\right) & \psi^k_{i\beta}\left(\mathbf{r}^\alpha_j; \{\mathbf{r}^\alpha_{/j}\}, \{\mathbf{r}^\beta\}\right) \\ \psi^k_{i\alpha}\left(\mathbf{r}^\beta_j; \{\mathbf{r}^\alpha\}, \{\mathbf{r}^\beta_{/j}\}\right) & \psi^k_{i\beta}\left(\mathbf{r}^\beta_j; \{\mathbf{r}^\alpha\}, \{\mathbf{r}^\beta_{/j}\}\right) \end{vmatrix}$$

W. M. C. Foulkes, D. Pfau and J. S. Spencer, GitHub (2020)

Extensions & Improvements

Dense Determinants



Block-Diagonal Determinants

Dense Determinants

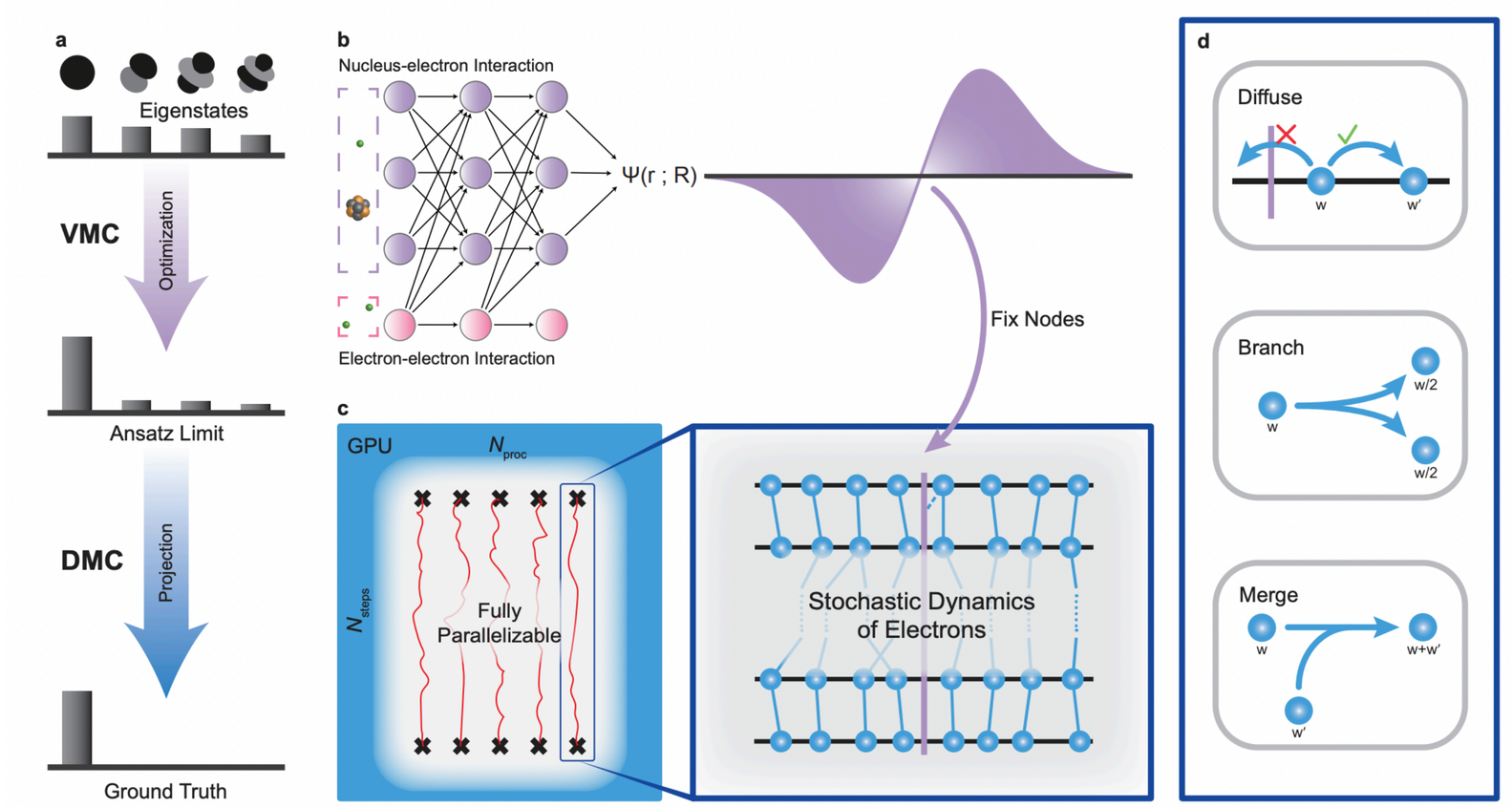
J. Lin, G. Goldshlager and L. Lin, arXiv (2021)

W. Ren, W. Fu and J. Chen, arXiv (2022)

L. Gerard, M. Scherbela, P. Marquetand and P. Grohs, arXiv (2022)

Extensions & Improvements

Diffusion Monte Carlo

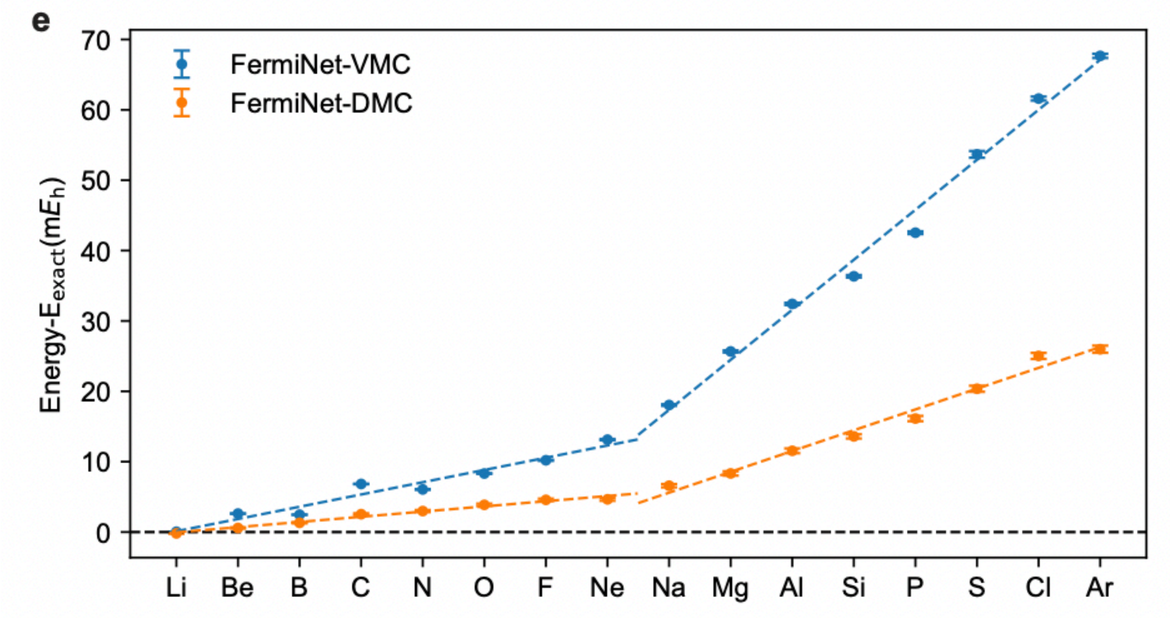


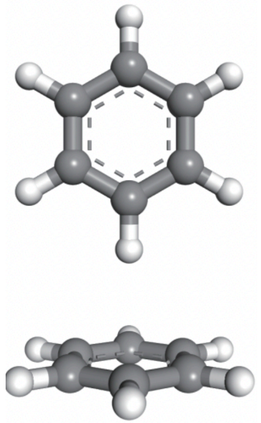
M. Wilson, N. Gao, F. Wudarski, E. Rieffel and N. M. Tubman, arXiv (2021)

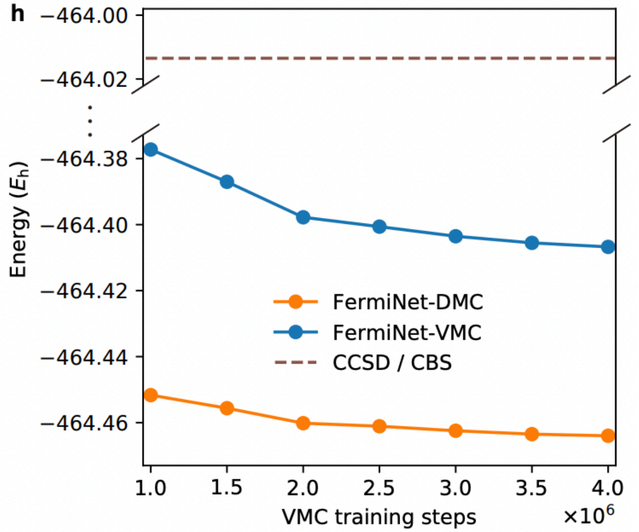
W. Ren, W. Fu and J. Chen, arXiv (2022)

Extensions & Improvements

Diffusion Monte Carlo







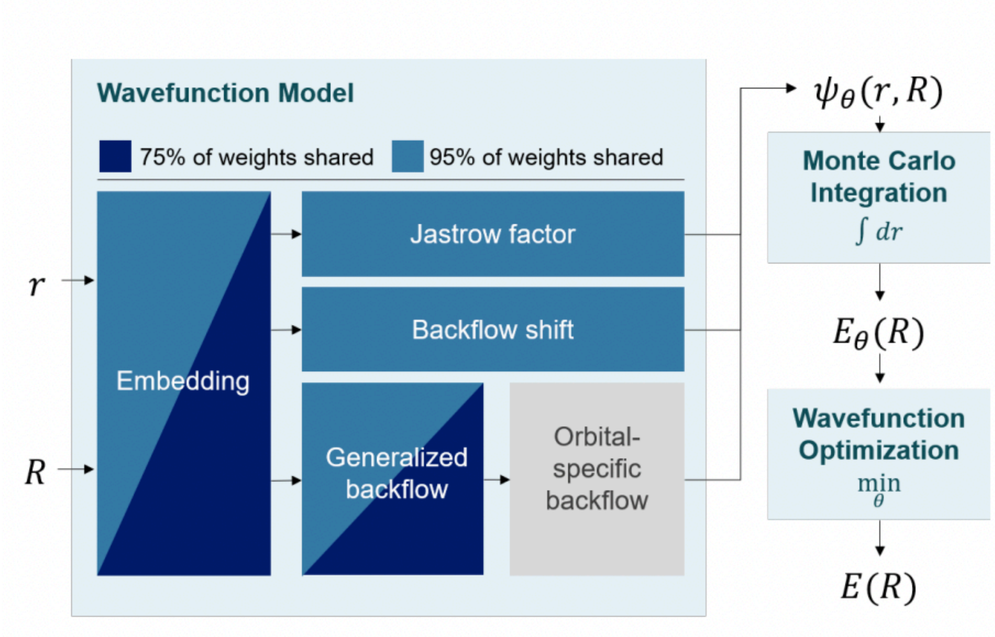
- Can generate a large improvement over VMC…if VMC is **significantly handicapped**
- Surprisingly **little advantage** over well-trained FermiNet with VMC

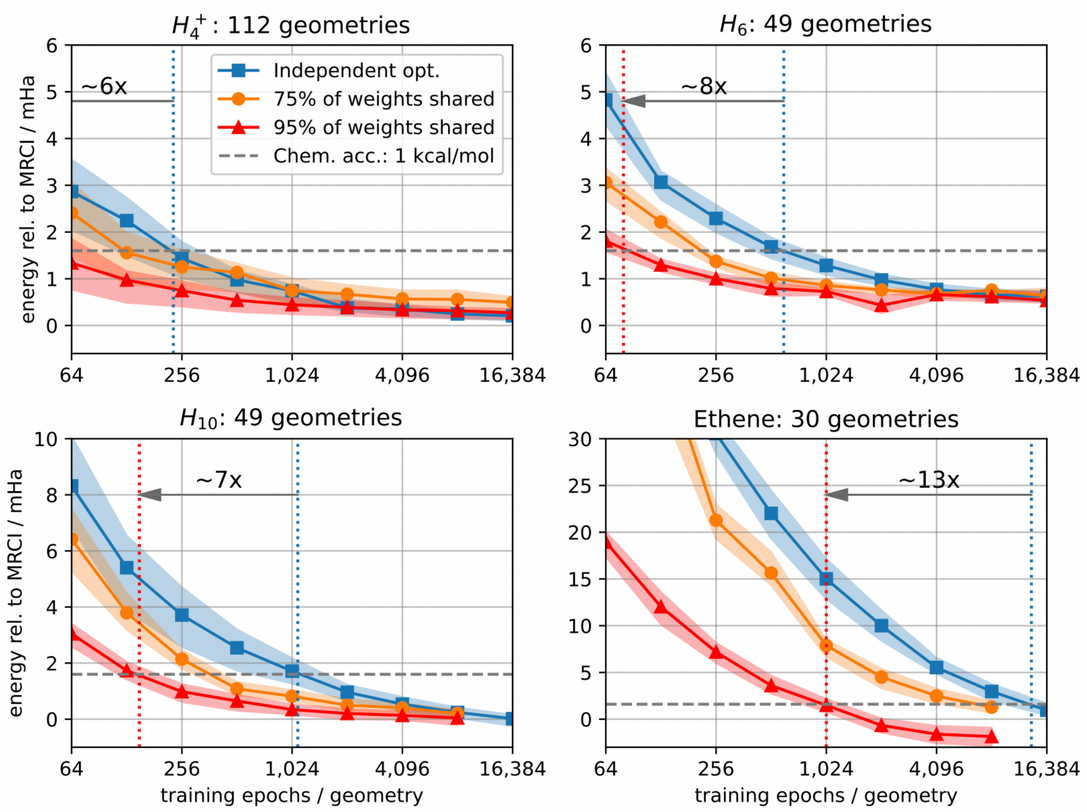
M. Wilson, N. Gao, F. Wudarski, E. Rieffel and N. M. Tubman, arXiv (2021)

W. Ren, W. Fu and J. Chen, arXiv (2022)

Extensions & Improvements

Weight Sharing: DeepErwin



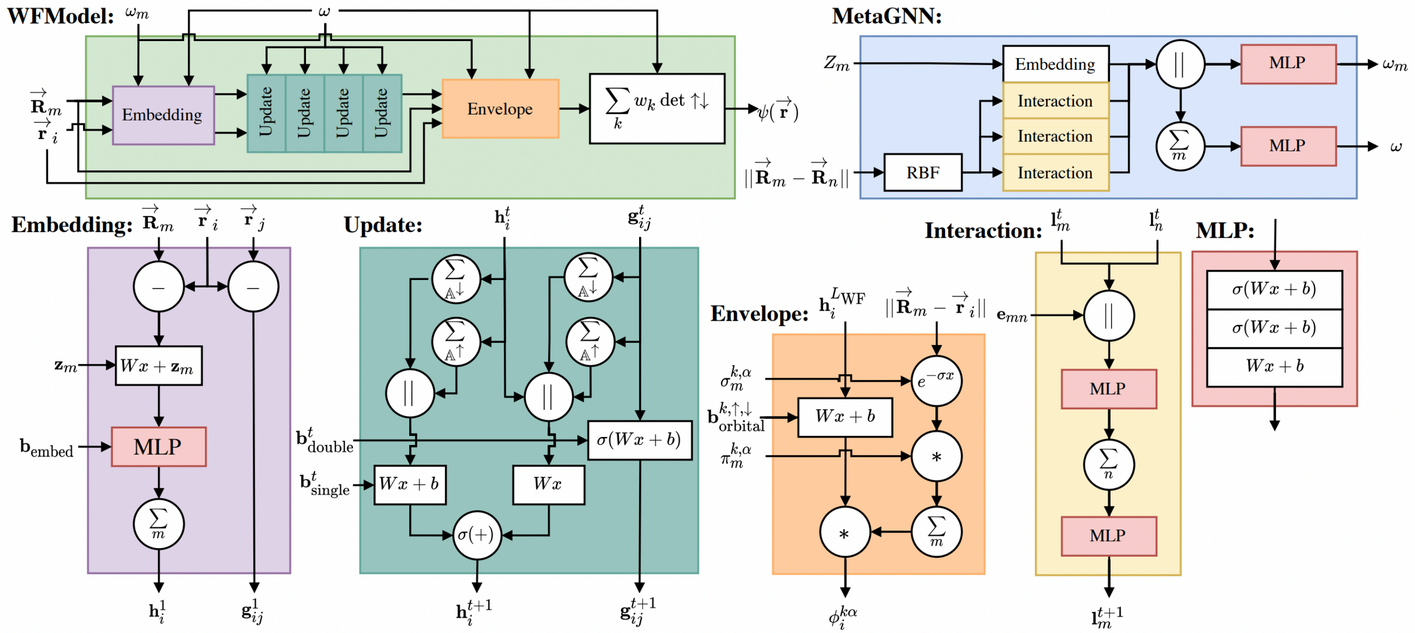


- Extends PauliNet to allow **sharing weights** between different systems
- Significantly cuts down on the cost of training a new system from scratch

M. Scherbela, R. Reisenhofer, L. Gerard, P. Marquetand and P. Grohs, Nat. Comp. Sci. (2022)

Extensions & Improvements

Weight Sharing: PESNet

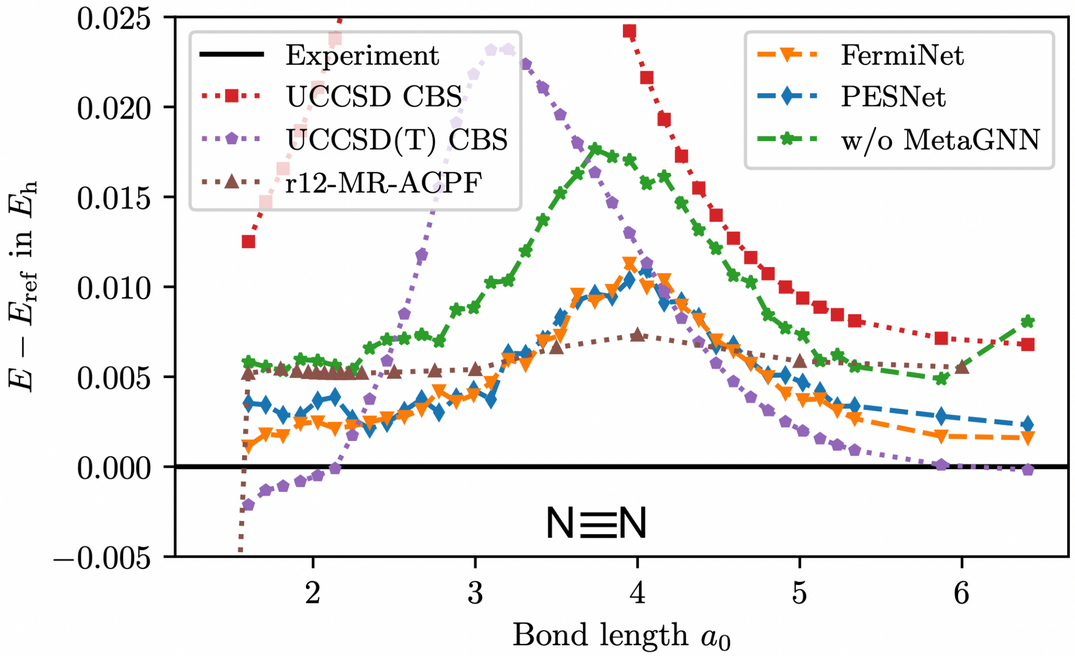


- Goes further than DeepErwin: **one** neural network that models **all** geometries
- Network is a combination of a **FermiNet** plus a **graph neural network** which maps atomic geometries to biases for the FermiNet

N. Gao and S. Günnemann, ICLR (2022)

Extensions & Improvements

Weight Sharing: PESNet



- Goes further than DeepErwin: **one** neural network that models **all** geometries
- Network is a combination of a **FermiNet** plus a **graph neural network** which maps atomic geometries to biases for the FermiNet

N. Gao and S. Günnemann, ICLR (2022)

Extensions & Improvements

Excited States

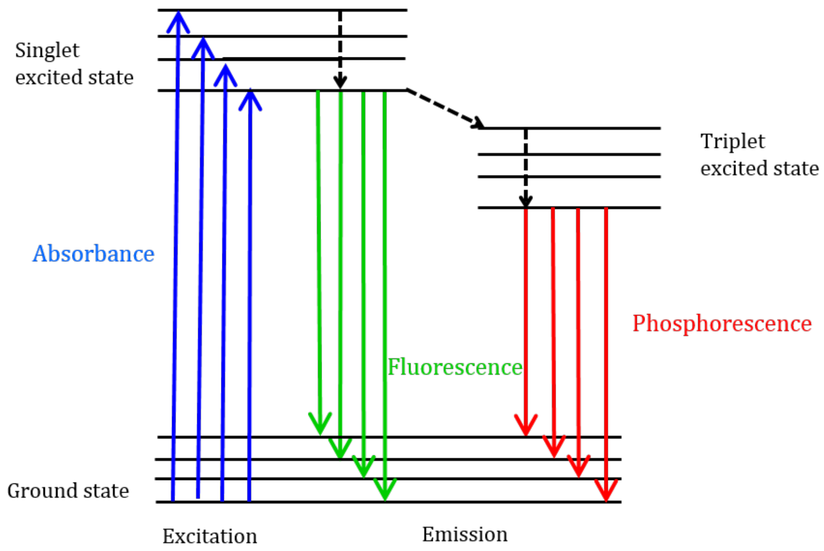


Image Credit: Jasco Inc.

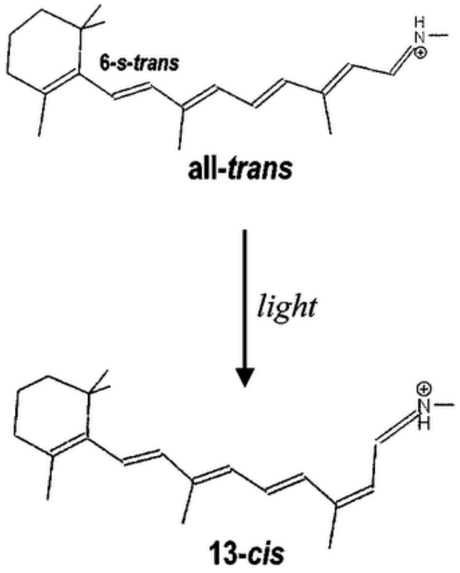


Image Credit: Chii-Shen Yang

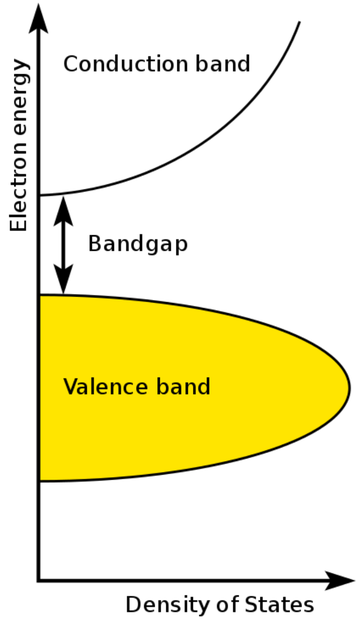


Image Credit: Wikipedia

- Critical for understanding **photoexcitation** (fluorescence, phosphorescence), **conformational changes** from light (retinal), **band gaps** in materials (semiconductors)
- Best method for targeting excited states with VMC is an **open problem** (not just for neural networks!)

Extensions & Improvements

Excited States

$$\min_{\Psi} \frac{\langle \Psi^* \hat{H} \Psi \rangle}{\langle \Psi^2 \rangle}$$

Ground State Objective

- No longer just targeting **lowest** energy state. Need to target multiple states and **guarantee they are orthogonal**!
- Basically two strategies:
  - **Design** Ansätze to be orthogonal - not an option for neural networks!
  - Add a **penalty** forcing states to be orthogonal - can lead to biased gradients, need to choose penalty strength

Extensions & Improvements

Excited States

$$\min_{\Psi} \frac{\langle \Psi^* \hat{H} \Psi \rangle}{\langle \Psi^2 \rangle}$$ **Ground State Objective**

$$\min_{\Psi_i} \frac{\langle \Psi_i^* \hat{H} \Psi_i \rangle}{\langle \Psi_i^2 \rangle} \text{ given } \langle \Psi_i^* \Psi_j \rangle = 0,\ j < i$$ **Excited State Objective**

- No longer just targeting **lowest** energy state. Need to target multiple states and **guarantee they are orthogonal**!
- Basically two strategies:
  - **Design** Ansätze to be orthogonal - not an option for neural networks!
  - Add a **penalty** forcing states to be orthogonal - can lead to biased gradients, need to choose penalty strength

Extensions & Improvements

Excited States

$$\min_{\Psi} \frac{\langle \Psi^* \hat{H} \Psi \rangle}{\langle \Psi^2 \rangle} = \min_{\Psi} \mathbb{E}_{\mathbf{r} \sim \Psi^2} \left[\Psi^{-1}(\mathbf{r}) \hat{H} \Psi(\mathbf{r}) \right]$$

Ground State VMC

M. Entwistle, Z Schätzle, P. A. Erdman, J. Hermann and F. Noé, arXiv (2022)

Extensions & Improvements

Excited States

$$\min_{\Psi} \frac{\langle \Psi^* \hat{H} \Psi \rangle}{\langle \Psi^2 \rangle} = \min_{\Psi} \mathbb{E}_{\mathbf{r} \sim \Psi^2} \left[\Psi^{-1}(\mathbf{r}) \hat{H} \Psi(\mathbf{r}) \right]$$

Ground State VMC

$$\min_{\Psi_1, \ldots, \Psi_k} \sum_{i=1}^{k} \mathbb{E}_{\mathbf{r} \sim \Psi_i^2} \left[\Psi_i^{-1}(\mathbf{r}) \hat{H} \Psi_i(\mathbf{r}) \right] + \alpha \sum_{j>i} \left(\frac{1}{1 - \sqrt{|S_{ij}|}} - 1 \right)$$

Excited State VMC

$$S_{ij} = \mathbb{E}_{\mathbf{r} \sim \Psi_i^2} \left[\frac{\Psi_j(\mathbf{r})}{\Psi_i(\mathbf{r})} \right] \mathbb{E}_{\mathbf{r} \sim \Psi_j^2} \left[\frac{\Psi_i(\mathbf{r})}{\Psi_j(\mathbf{r})} \right] \longleftarrow$$

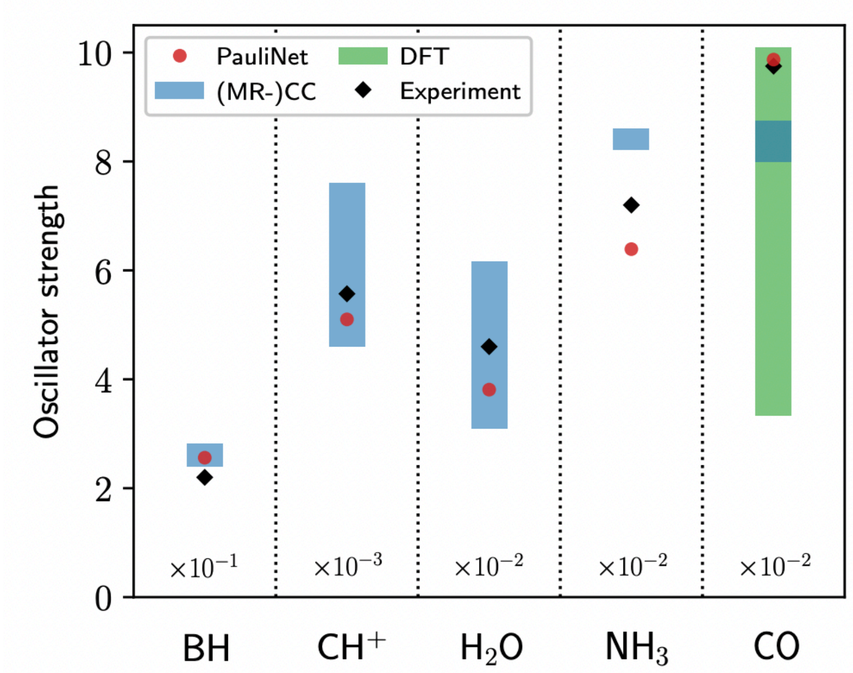
Cosine distance between wavefunctions

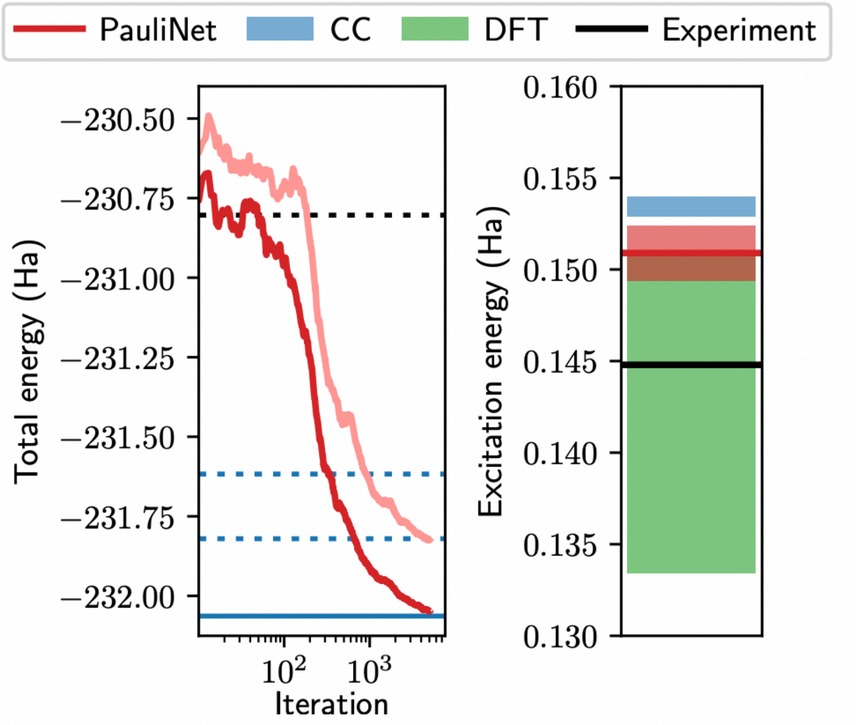
- Penalty strongly forces states apart…but estimator is **biased**
- Works well for small systems

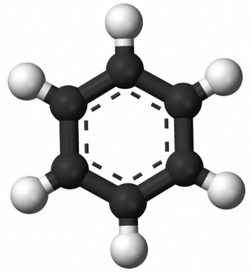
M. Entwistle, Z Schätzle, P. A. Erdman, J. Hermann and F. Noé, arXiv (2022)

Extensions & Improvements

Excited States







- Penalty strongly forces states apart…but estimator is **biased**
- Works well for small systems

M. Entwistle, Z Schätzle, P. A. Erdman, J. Hermann and F. Noé, arXiv (2022)

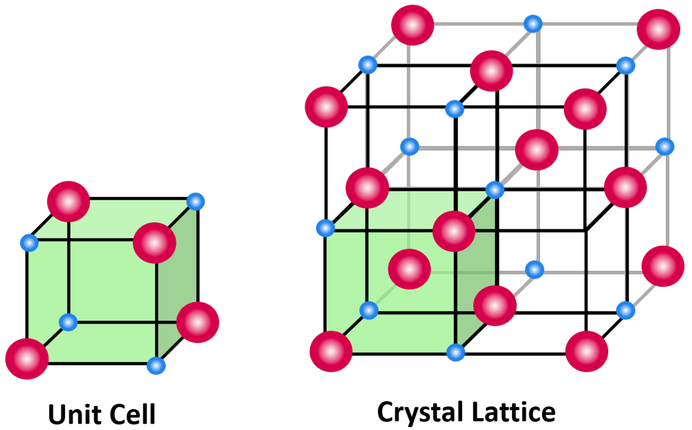
Materials

- Molecules have **open boundary conditions**

$$\Psi(\mathbf{r}_1, \ldots, \mathbf{r}_n) \to 0 \quad \text{as} \quad \mathbf{r}_i \to \infty$$

Materials

- Molecules have **open boundary conditions**
- Materials have **periodic boundary conditions**



Materials

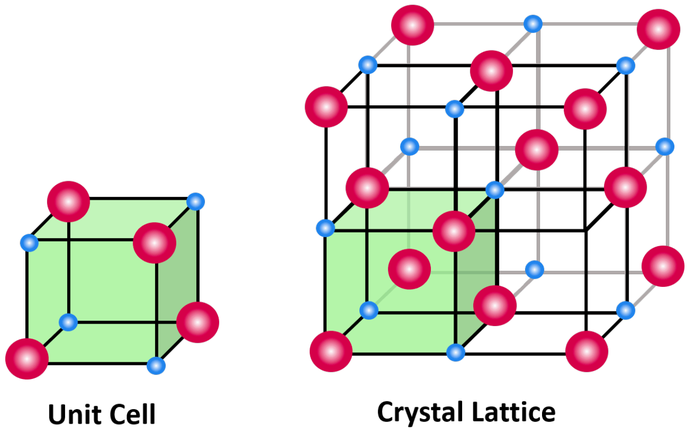
- Molecules have **open boundary conditions**
- Materials have **periodic boundary conditions**
- Formalized as **Bloch's Theorems**

$$\Psi(\mathbf{r}_1, \ldots, \mathbf{r}_n) = U(\mathbf{r}_1, \ldots, \mathbf{r}_n) \exp\left(i\mathbf{k}_s^T \sum_i \mathbf{r}_i\right)$$

Invariant under translation of **one** electron

$$\Psi(\mathbf{r}_1, \ldots, \mathbf{r}_n) = W(\mathbf{r}_1, \ldots, \mathbf{r}_n) \exp\left(i\mathbf{k}_p^T \frac{1}{n} \sum_i \mathbf{r}_i\right)$$

Invariant under translation of **all** electrons



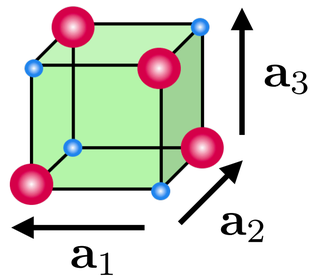
F. Bloch, *Zeitschrift für Physik* (1928)

G. Rajagopal, R. J. Needs, A. James, S. D. Kenny, W. M. C. Foulkes, PRB (1995)

Materials

- The FermiNet can be directly modified to be compatible with Bloch's theorems:
- **Remove** the exponentially-decaying envelope needed for open boundaries
- Add a **complex plane-wave envelope** to satisfy the Bloch theorems
- Modify the input features to be **periodic**

$$s_i = \mathbf{a}_i^T \mathbf{r} \qquad s_i \rightarrow \sin(s_i), \cos(s_i)$$



$$||\mathbf{r}||^2_{\text{per}} = \sum_{ij} \left[1 - \cos(2\pi s_i)\right] \mathbf{a}_i^T \mathbf{a}_j \left[1 - \cos(2\pi s_j)\right] + \sin(2\pi s_i) \mathbf{a}_i^T \mathbf{a}_j \sin(2\pi s_j)$$

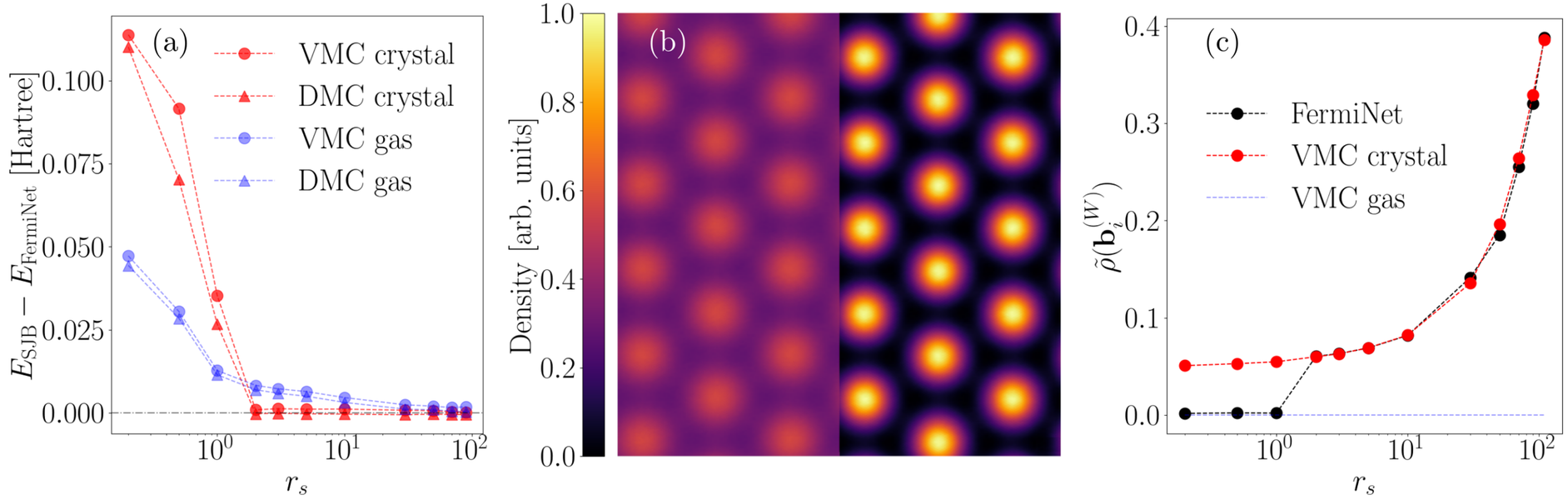
M. Wilson, S. Moroni, M. Holzmann, N. Gao, F. Wudarski, T. Vegge and A. Bhowmik, arXiv (2022)

G. Cassella, H. Sutterud, S. Azadi, N. D. Drummond, D. Pfau, J. S. Spencer and W. M. C. Foulkes, arXiv (2022)

X. Li, Z. Li and J. Chen, arXiv (2022)

Materials

Wigner Crystallization



- Lower energy than conventional Ansätze
- Works on **both sides of the phase transition** - phase is not assumed!

G. Cassella, H. Sutterud, S. Azadi, N. D. Drummond, D. Pfau, J. S. Spencer and W. M. C. Foulkes, arXiv (2022)

Materials

Extending to Solids

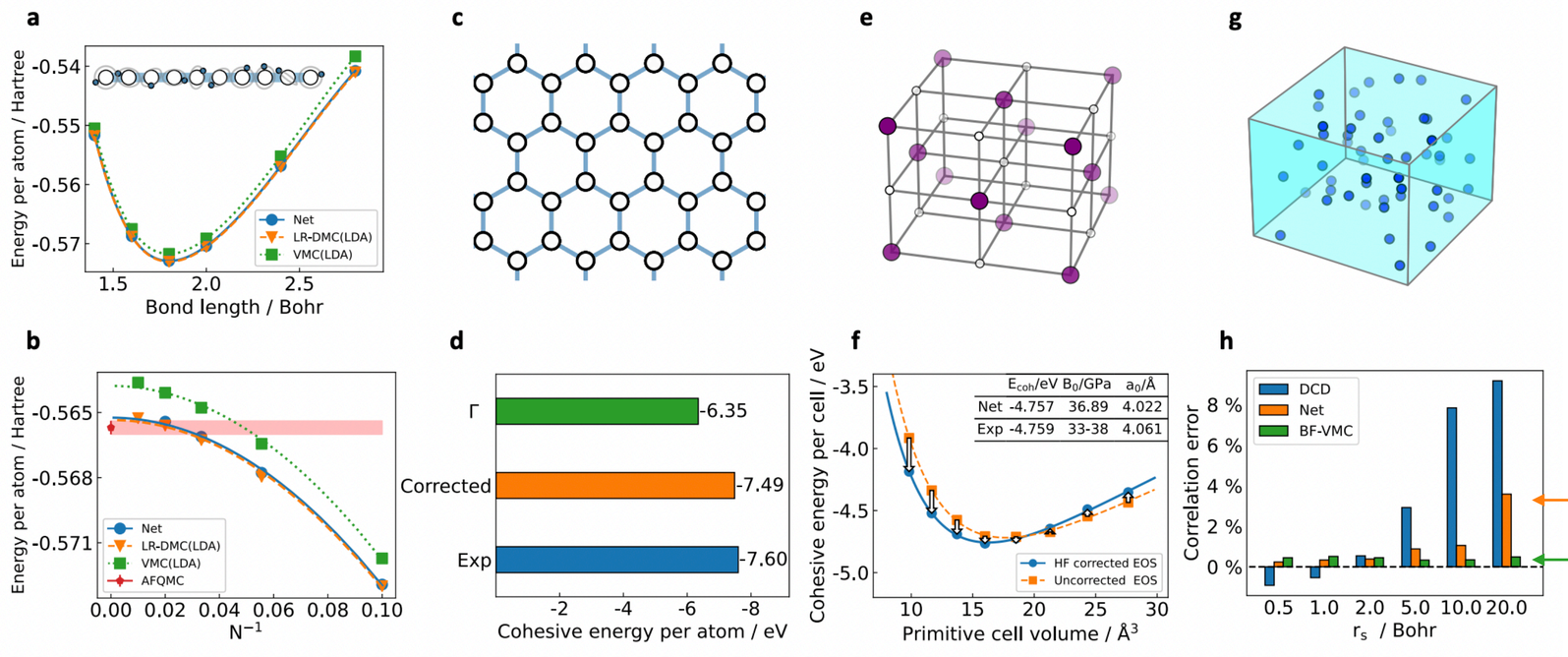
- Need to add **atomic potentials**
- Need to account for **finite size effects**: two techniques
  - Increase the number of unit cells: number of electrons grows **cubically**!
  - “k-point averaging”: repeatedly solve with different **fixed phase shifts**

$$\Psi(\mathbf{r}_1, \ldots, \mathbf{r}_n) = U(\mathbf{r}_1, \ldots, \mathbf{r}_n) \exp\left(i\mathbf{k}_s^T \sum_i \mathbf{r}_i \right)$$

Choose fixed k, solve for ground state, rinse, repeat

Materials

Preliminary Results on Solids



- Simplified network for scaling: only **one determinant**, small batch size
- Largest neural network ansatz calculation to date: **108 electrons** on LiH (3x3x3 lattice)
- Results on 54 electron homogeneous electron gas are **worse** than conventional VMC

X. Li, Z. Li and J. Chen, arXiv (2022)

Future Directions

- Pushing to **higher accuracy**: how far are we from the true ground state?
- Scaling to **bigger systems**: how can we simulate hundreds of electrons without sacrificing accuracy or using all the world's GPUs?
- Integration with **other electronic structure methods**: can hybrid methods give us the best of both worlds - accuracy and speed?
- Better **interpretability**: can we open up the black box and better understand what these neural networks are learning that other methods can't?
- Better performance on **real materials**: how can we handle finite size effects, scale to larger lattices?

Summary

- Building neural networks with **invariant** or **equivariant** representations is a powerful recipe for creating accurate, general approximations to high dimensional functions
- **Second-order optimization** methods can be applied to neural networks and are well suited for *ab-initio* electronic structure applications
- Only a small amount of prior physical knowledge needs to be built into neural networks to make them appropriate as **wavefunction Ansätze**
- Neural network Ansätze can dramatically improve the accuracy of variational QMC, making it **competitive with other state-of-the-art** electronic structure methods
- While there has been initial work on extensions to materials, applications to real materials where neural networks beat alternatives remains an **open problem**

Thanks

James Spencer

Halvard Sutterud

Gino Cassella

Matthew Foulkes

Alex Matthews

Alex Botev

James Martens