Quantum simulation and quantum phase transitions of an extended Agassi model

Álvaro Sáiz Castillo

José-Enrique García-Ramos, José Miguel Arias,
Lucas Lamata, Pedro Pérez-Fernández

The Extended Agassi Model

$$H = \varepsilon J^0 - g \sum_{\sigma,\sigma'} A_\sigma^\dagger A_{\sigma'} - \frac{V}{2}\left[(J^+)^2 + (J^-)^2\right] - 2hA_0^\dagger A_0$$

Jordan-Wigner mapping

$$c_{\sigma,m}^\dagger \rightarrow c_1^\dagger = I_1 \otimes \ldots \otimes I_{i-1} \otimes \sigma_i^+ \otimes \sigma_{i+1}^z \otimes \ldots \otimes \sigma_N^z$$

$$c_{\sigma,m} \rightarrow c_1 = I_1 \otimes \ldots \otimes I_{i-1} \otimes \sigma_i^- \otimes \sigma_{i+1}^z \otimes \ldots \otimes \sigma_N^z$$

Quantum Simulation of the model

Everything in terms of Pauli matrices

$$\sigma^x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \; \sigma^y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \; \sigma^z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix},$$

For example: $\boldsymbol{\sigma_2^x \otimes \sigma_3^y \otimes \sigma_5^x \otimes \sigma_6^z \otimes \sigma_7^z \otimes \sigma_8^y}$

Performed via Mølmer–Sørensen gates

Quantum Simulation of the model

Evolution

$$U(t) = e^{-itH}$$

Trotter Expansion

Trotter steps

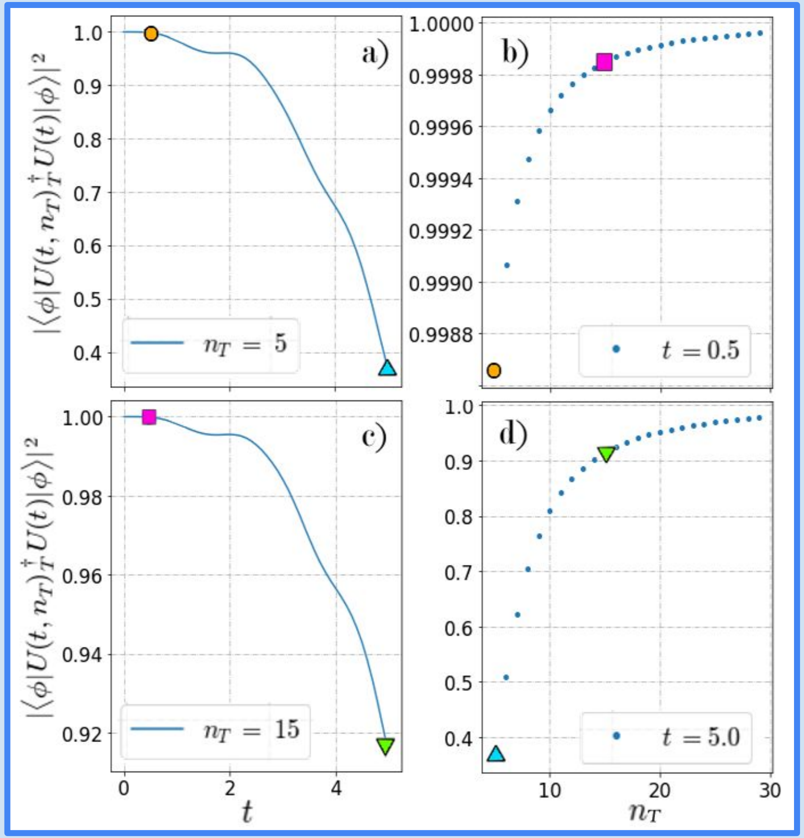
$$U(t,n)_T = \left(\prod_k e^{-itH_k/n_T}\right)^{n_T} \approx U(t)$$

Quantum Simulation of the model

Fidelity

$$F(t, n_T) = |\langle\phi|U(t,n_T)_T^{\dagger}U(t)|\phi\rangle|^2$$

$$|\phi\rangle = |\downarrow_1 \;\; \downarrow_2 \;\; \downarrow_3 \;\; \downarrow_4 \;\; \uparrow_5 \;\; \uparrow_6 \;\; \uparrow_7 \;\; \uparrow_8 \;\rangle$$



Quantum Phase Transitions (QPTs)

Redefine the parameters

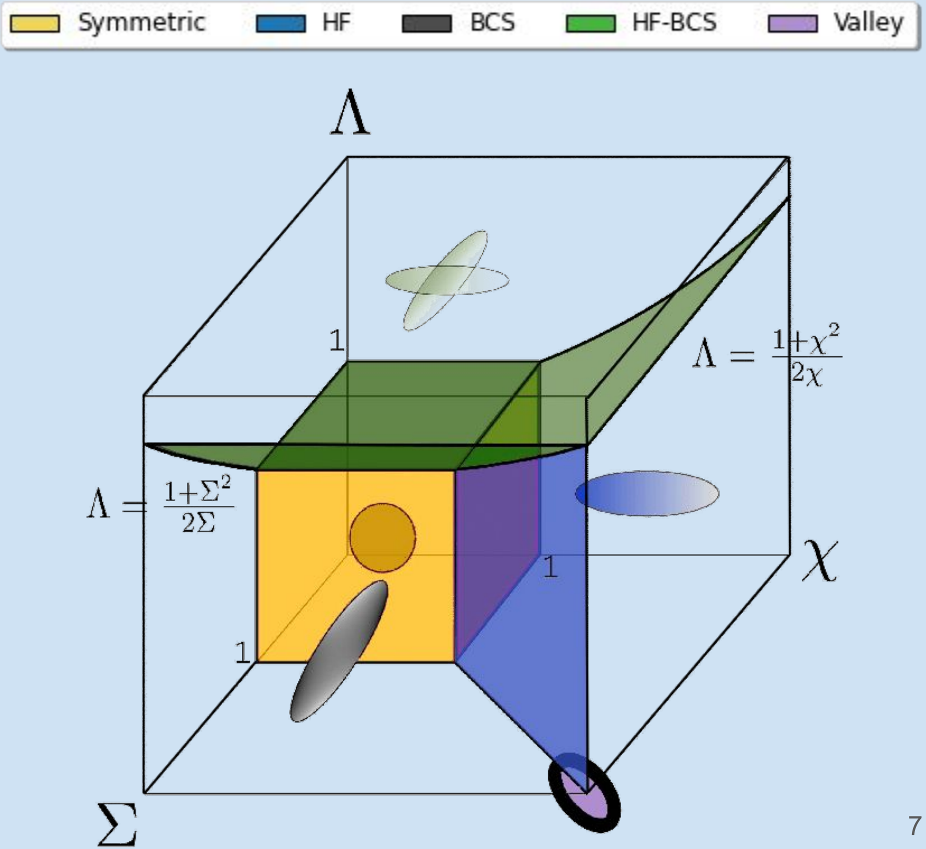
$$H = \varepsilon J^0 - g \sum_{\sigma,\sigma'} A^\dagger_\sigma A_{\sigma'} - \frac{V}{2}\left[(J^+)^2 + (J^-)^2\right] - 2h A^\dagger_0 A_0$$

$$g = \frac{\varepsilon \Sigma}{2j-1} \qquad V = \frac{\varepsilon \chi}{2j-1} \qquad h = \frac{\varepsilon \Lambda}{2j-1}$$

Quantum Phase Transitions (QPTs)

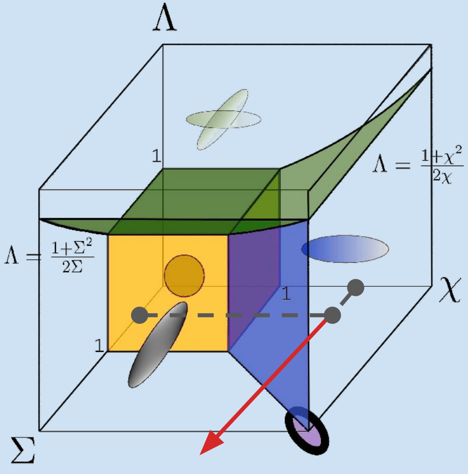
Quantum Phase Diagram

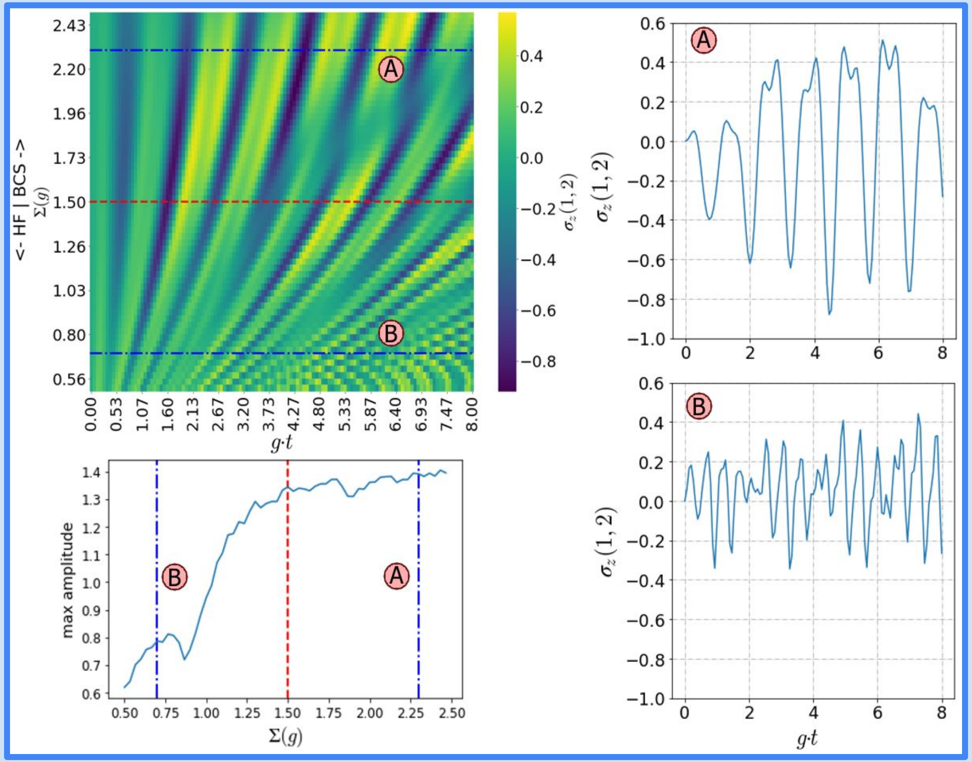
- Symmetric phase
- Hartree-Fock phase (HF)
- Bardeen-Cooper-Schrieffer phase (BCS)
- Combined HF-BCS phase
- Closed Valley solution



Correlation Function

$$\sigma_z(1,2) = \langle \sigma_1^z \otimes \sigma_2^z \rangle - \langle \sigma_1^z \rangle \langle \sigma_2^z \rangle$$

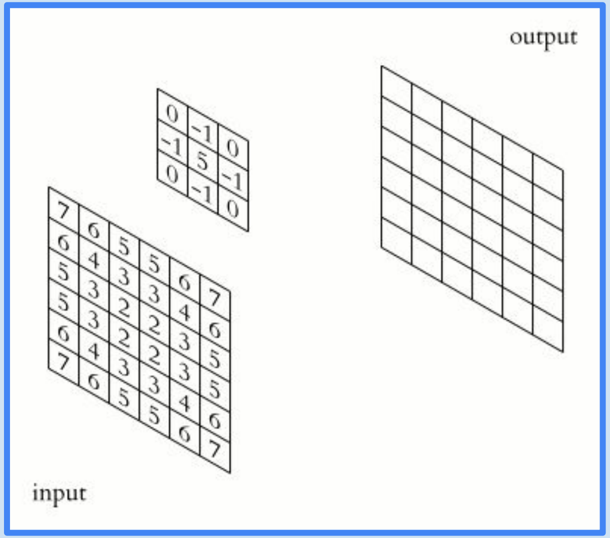




Deep Learning - Convolutional Neural Network (CNN)

The process of Convolution

Applying a filter of set size with set weights to an input.



We apply several filters and stack the results. Then we reduce the size of the output via pooling and repeat.

The filters are initialized with random weights that are optimized during the training process of the CNN.

Animation source: https://commons.wikimedia.org/wiki/File:2D_Convolution_Animation.gif

Deep Learning - Results

Results

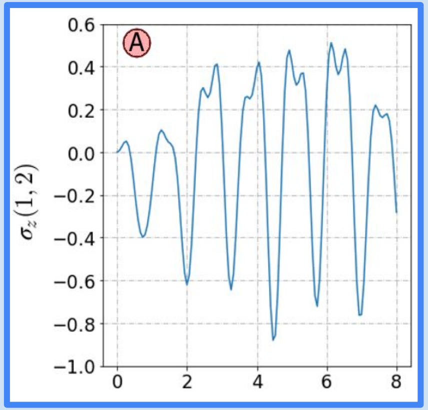
$$\sigma_z(1,2) = \langle \sigma_1^z \otimes \sigma_2^z \rangle - \langle \sigma_1^z \rangle \langle \sigma_2^z \rangle$$

$$U(t) = e^{-itH(\chi,\Sigma,\Lambda)} \approx \left(\prod_k e^{-itH_k/n_T}\right)^{n_T}$$

$$f(t) = \langle \phi(0) | U^\dagger(t) \sigma_z(1,2) U(t) | \phi(0) \rangle$$

$\text{input} = f(t)$ → **CNN** → $\text{output} = [P(y_1), P(y_2), P(y_3), P(y_4), P(y_5)]$

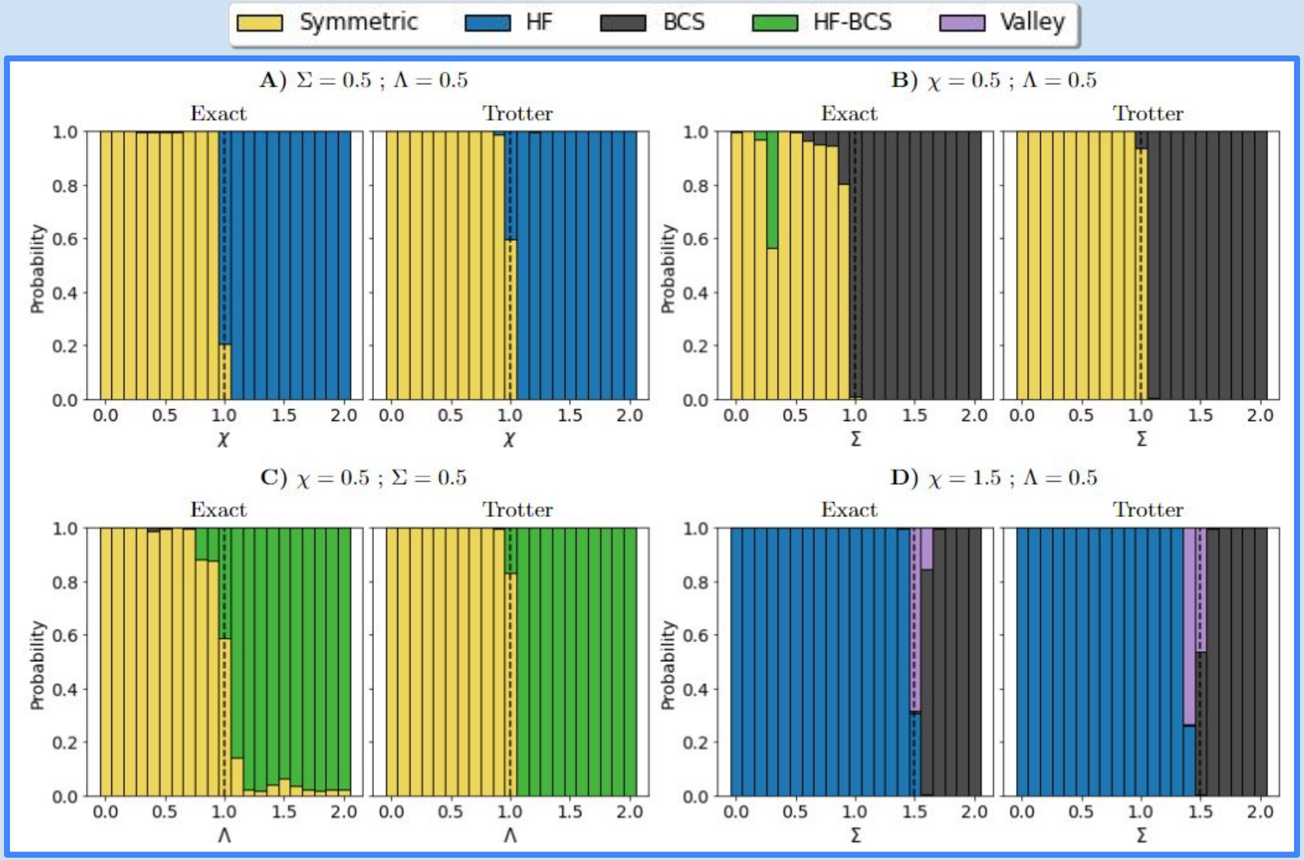
Input:



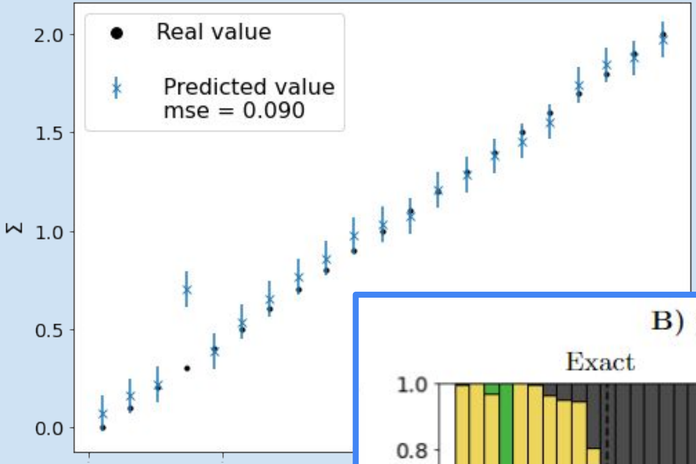
Output:

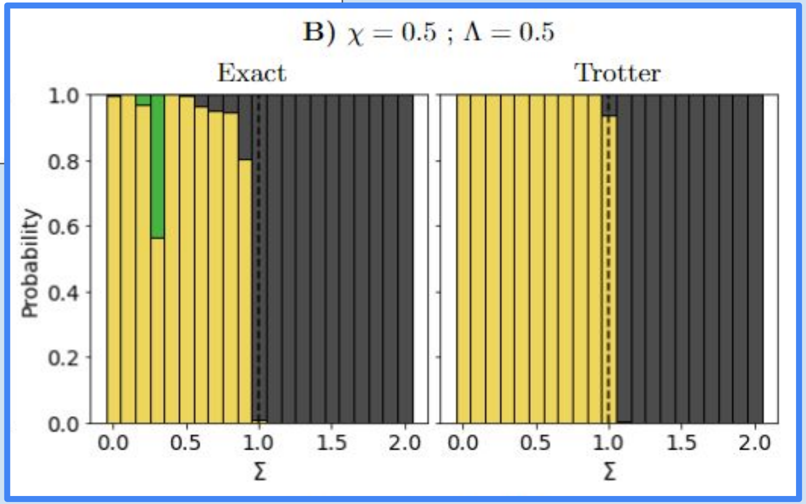
- Symmetric = 0.84
- HF = 0.12
- BCS = 0.02
- HF-BCS = 0.01
- Valley = 0.01

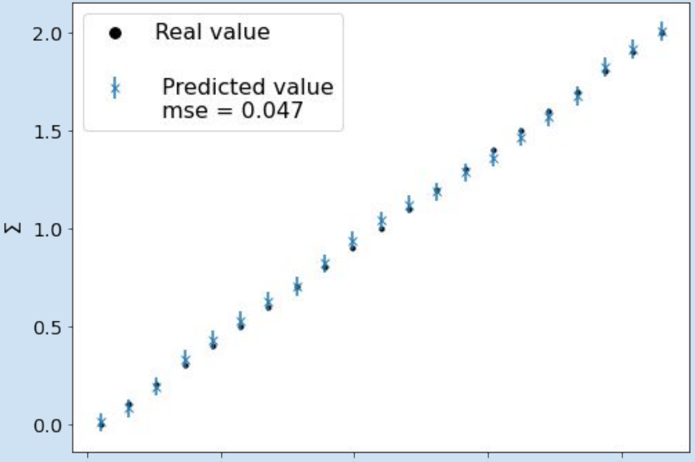
Deep Learning - Results



Deep Learning - Results







Mean error smaller than the step size (0.1)

Conclusions

- The quantum simulation is feasible with polynomial resources.
- Observables can be measured with this experimental setup.
- These observables can provide information of the system.
- Machine Learning methods can make use of this information to extract the Quantum Phases of the system.
- These methods are robust against introduced errors.
- It’s fast and very accurate. Easily tailored to specific cases and different Hamiltonians, as long as there is data.
- Computing the data is inefficient (very time consuming) and requires previous knowledge of the model.

Deep Learning - Convolutional Neural Network (CNN)

CNN structure

Layer	Output Shape	Param N
Convolution 1D (1)	(100, 32)	128
Leaky ReLU (1)	(100, 32)	0
Avg. Pooling 1D (1)	(34, 32)	0
Spatial Dropout (1)	(34, 32)	0
Convolution 1D (2)	(34, 64)	6208
Leaky ReLU (2)	(34, 64)	0
Avg. Pooling 1D (2)	(12, 64)	0
Spatial Dropout (2)	(12, 64)	0
Convolution 1D (3)	(12, 128)	24704
Leaky ReLU (3)	(12, 128)	0
Avg. Pooling 1D (3)	(4, 128)	0
Spatial Dropout (3)	(4, 128)	0
Convolution 1D (4)	(4, 256)	98560
Leaky ReLU (4)	(4, 256)	0
Avg. Pooling 1D (4)	(2, 256)	0
Spatial Dropout (4)	(2, 256)	0

Flatten →

Layer	Output Shape	Param N
Dense (1)	(512)	262656
Leaky ReLU (1)	(512)	0
Dropout (1)	(512)	0
Dense (2)	(512)	262656
Leaky ReLU (2)	(512)	0
Dropout (2)	(512)	0
Dense (3)	(512)	262656
Leaky ReLU (3)	(512)	0
Dropout (3)	(512)	0
Dense (4)	(512)	262656
Leaky ReLU (4)	(512)	0
Dropout (4)	(512)	0
Dense (5)	(512)	262656
Leaky ReLU (5)	(512)	0
Dropout (5)	(512)	0
Softmax	(5)	2562

Total parameters		1,445,445