Research Log 2019/12/27-2020/01/02

Fri 12/27/19

- Play around in Mathematica to see what kind of optimization I get if I optimize over probabalities, not amplitudes. What changes if I express my variational basis operators in terms of the Pauli basis (so that I get non-zero imaginary parts in corresponding amplitudes?)
- Looking at old code, my optimization (over amplitudes) was computed with a Monte-Carlo computational basis (spins / fermions), not the basis of the even/odd parity sectors. Which is of course irrelevant for the amplitudes, but is important as I start looking at probabilities (function `extractCoeffs`)
- The coefficients for the VQ basis are given by `gsCoeffs` (ground state), `eEvenPCoeffs` (even-parity excited states), and `eOddPCoeffs` (odd-parity excited states), while the basis vectors themselves are in `varBasis1` and `varBasis2` , respectively.
- For now, working in this basis, let's see how different the optimization is for computing
  - probabilities for variational basis operators
  - probabilities for Paulis
- Then re-do these steps if the basis is different.
- Of course should see no difference for doing amplitude-level optimization for different bases.
- Compare resulting $\vec{\alpha}$ as well as the overlap onto target state.
- So, upon actually thinking, I am realizing that even the probability
  $$|\langle \phi_a | \widetilde{V}^{-}, \vec{\theta}' \rangle|^2$$
  is not immediately accessible: we don't actually have the state $|\widetilde{V}^{-}, \vec{\theta}'\rangle$ on the computer. Since the pauli strings are unitary, we can measure things like $|\langle \phi_a | \hat{B}_s | \Omega, \vec{\theta} \rangle|^2$. I do not know how to infer the probability for a linear combination of these strings from here, I am missing all the interference terms.
- So, even though I could go ahead and concoct, in Mathematica, $|\langle \phi_a | \widetilde{V}^{-}, \vec{\theta}' \rangle|^2$, I don't know how we would do that on the quantum side.

On another note, although I still don't know how to measure the imaginary part of an amplitude directly, I know I can measure the square of the off-diagonoal amplitude for a unitary operator (e.g. Pauli string). From those two I can figure out the imaginary part up to a sign.

- In Scott's trick, just rotate both qubits by $\exp -i\pi/4$ and ditto to get the imaginary part of the amplitude? **No, just do the same thing for $-i$ times the operator. This gets the operator's imaginary part.**

Mon 12/30/19 — diving into tensor networks

- Original approaches to lattice field theories gave good results for the ground state, but precision fell quickly for excited states. Originally periodic boundary conditions (PBCs) were emloyed.
- 1305.3765 (https://arxiv.org/pdf/1305.3765.pdf) uses open boundary conditions (OBC).

    - "For finite systems, the MPS algorithms for open boundary conditions are numerically more stable and in general more efficient, although improved methods have been recently proposed for periodic systems [45, 46]. However, finite-size effects are much larger for OBC and therefore simulation of larger chains may be needed to reach the thermodynamic limit reliably."
    - "The masses of the particles in the theory are given by the energies of the zero momentum excitations. In finite systems with OBC momentum is difficult to identify andwe need to find the excitations that will correspond to the lowest momentum in the continuum limit."
    - "On a staggered lattice with PBC it is possible to exploit the corresponding lattice symmetries to construct two orthogonal subspaces, one of them containing the vector, and the other the ground state and the scalar [49]. For a chain with OBC the number of surviving symmetries is even lower, with translational invariance lost. In practice this means that to calculate the scalar mass, we need to identify first the momentum excitations of the vector, which appear at lower energy than the scalar."
- This letter (https://arxiv.org/pdf/cond-mat/0404706.pdf) explains the algorithm.
- Aside: typeset algorithms in $\LaTeX$ Notes with `algorithms` (https://www.ctan.org/tex-archive/macros/latex/contrib/algorithms/) and `algorithmicsx` (ctan.org/tex-archive/macros/latex/contrib/algorithmicx/)

Summary for 1305.3765 (https://arxiv.org/pdf/1305.3765.pdf)

- Open boundary conditions are significantly more numerically stable but lead to a number of required workarounds: estimating 0-momentum states in finite volume, requiring larger chains compared to PBCs to supress finite-size effects, and having to work in full space (even and odd parity, full momentum).
- Links are integrated out, unlike in the lattice formulation. Other than that, the computational basis is pretty much the one we are using on the lattice (since cannot project onto symmetry sectors anyway)
- **Preliminarily, porting our approach to MPS requires the following ingridients:**
    - Running the usual MPS, but perhaps with PBCs? And saving not only the energy levels, but also the estimates (tensors) for the ground state and the first excited state:

$$|E_0\rangle = \sum_{\substack{i_0,\,\ldots,\,i_{N-1}\\ \ell_0,\,\ldots,\,\ell_{N-1}}} \mathrm{tr}(A_0^{i_0} B_0^{\ell_0} \;\ldots\; A_{N-1}^{i_0} B_{N-1}^{\ell_0}) |i_0 \ell_0 \;\ldots\; i_{N-1} \ell_{N-1}\rangle$$

$$|E_1\rangle = \sum_{\substack{i_0,\,\ldots,\,i_{N-1}\\ \ell_0,\,\ldots,\,\ell_{N-1}}} \mathrm{tr}(A_0^{i_0\prime} B_0^{\ell_0\prime} \;\ldots\; A_{N-1}^{i_0\prime} B_{N-1}^{\ell_0\prime}) |i_0 \ell_0 \;\ldots\; i_{N-1} \ell_{N-1}\rangle$$

    - Devising a Matrix Product Operator Ansatz for the interpolating operator field, $\hat{O}_{\mathrm{TP}}$, and variationally optimizing

$$\langle E_1 | \hat{O}_{\mathrm{TP}} | E_0 \rangle,$$

    - and then reading off the optimal operator.

Tue 12/31/19

- Understand matrix product operators [1] (https://arxiv.org/pdf/0804.3976.pdf)
- Understand how the tensor contraction works [2] (https://arxiv.org/pdf/cond-mat/0404706.pdf)

- How do I implement all of this?
  - Sam's references:
    - https://arxiv.org/pdf/1603.03039.pdf (https://arxiv.org/pdf/1603.03039.pdf)
    - https://arxiv.org/pdf/1306.2164.pdf (https://arxiv.org/pdf/1306.2164.pdf)
  - Libraries (at first glance, the Julia library seems more transparent --- better to understand what is going on and adapt to our purposes):
    - Julia: https://github.com/Jutho/TensorOperations.jl (https://github.com/Jutho/TensorOperations.jl)
    - Python: http://amilsted.github.io/evoMPS/ (http://amilsted.github.io/evoMPS/)

Reading some background papers on DMRG. Rooted in real-space renormalization, the kind you see in condensed matter systems. Useful excerpts:

- "Let us assume we have diagonalized a superblock (*e.g. a big chunk of a lattice*) and obtained one particular state $|\psi\rangle$, probably the ground state. Let $|i\rangle,\ i = 1,\ \ldots,\ \ell$ be a complete set of states of BB (the system, *a single block B in a superblock, factored into the inter- and intra-block interactions*) and $|j\rangle, , j = 1,\ \ldots,\ J$ be the states of the rest of the superblock, i.e., the "universe." We can write
$$|\psi\rangle = \sum_{i,\,j} \psi_{ij} |i\rangle |j\rangle.$$
We will assume for simplicity ψ_{ij} is real. We wish to define a procedure for producing a set of states of the system $|u\rangle,\ n = 1,\ \ldots,\ m$, with
$$|u^{\alpha}\rangle = \sum_{i} u_{i}^{\alpha} |i\rangle,$$
which are optimal for representing $|\psi\rangle$ in some sense. **Because we allow only m states, we cannot represent $|\psi\rangle$ exactly if $\ell > m$** We wish to construct an accurate expansion for ψ of the form
$$|\psi\rangle \approx |\bar{\psi}\rangle = \sum_{\alpha,\,j} a_{\alpha,\,j} |u^{\alpha}\rangle |j\rangle.$$
In other words, we wish to minimize
$$\mathcal{S} = ||\psi\rangle - |\bar{\psi}\rangle|^2,$$
by varying over all $a_{\alpha,\,j}$ and u^{α}, subject to $\langle u^{\alpha'} | u^{\alpha} \rangle = \delta_{\alpha\alpha'}$."
- "The solution to this minimization problem is known from linear algebra... [The] optimal states u^{α} are also eigenvectors of the reduced density matrix of the system as part of the ... [superblock]. This reduced density matrix for the system depends on the state of the superblock..."
- "The optimal states to keep are the eigenvectors of the reduced density matrix with the largest eigenvalues."
- Various numerical algorithms based on DMRG will vary in how the blocks are defined and how they are enlarged at each step.
- **Why open boundary condititions are more numerically stable:** . "[T]he open boundary condition case performs much better than the periodic boundary condition case. Our intuitive picture for this numerical result is the following: Roughly speaking, each eigenstate of the block density matrix represents the response of the block to a particular quantum fluctuation in the rest of the chain. A block with two ends which connect to the rest of the system must respond to nearly independent fluctuations near each of the ends. In the case of a long block, where the ends are nearly independent, if m states are required to accurately describe a single end to a given accuracy, then approximately m states would be required to accurately represent both

ends. In other words, if for a given accuracy open boundary condition require m states, periodic boundary conditions require roughly m^2 states."

Read section 1 in Sam's first reference (https://arxiv.org/pdf/1603.03039.pdf). Got the notation for TNN's down. Relevant sections to read are section 3 (for MPS) and section 5.1 (for DMRG). It's a good reference that's not too burdened with physics and technical details. Very computer-sciency, good as a reference when implementing DMRG.

Wed 01/01/2020

- Finish reading the physics-based DMRG (https://arxiv.org/pdf/cond-mat/0404706.pdf) reference

I am slowly converging to an understanding of MPS for DMRG; it's taking longer than I expected, but the references I have used so far are generally quite good. One specific thing I can do before the meeting is to compile a list of references I have used so far, with brief summaries for each reference.

- Install Julia and the Tensor Network's package. Useful links:
  - AutoMPO Interaface for converting operators into tensor networks (https://itensor.org/docs.cgi?page=classes/autompo)
  - ITensor GitHub page (https://github.com/ITensor/ITensors.jl)

Thu 01/02/2020

- Compile a summary of references used so far (Overleaf notes). **Done**
- Read the "DMRG-independent MPS" references (Swedish papers): this letter (https://journals.aps.org/prl/pdf/10.1103/PhysRevLett.75.3537) and this more detailed paper (https://journals.aps.org/prb/pdf/10.1103/PhysRevB.55.2164).
- Read on MPS from Sam's reference.

For the meeting:

- There were some small updates to notes on measuring amplitudes. I think the relevant questions to ask are:

  1. Given two multi-qubit variational states, how to prepare their equal superposition? The fact that we have a concatenated space may help. Or, perhaps more easily attainable,
  2. How to prepare an equail superposition of a variational state with a computational basis element?
- Another thing I wanted to mention concerns swap tests for Paulis (unitary operators)
- Discuss: missing interference Pauli terms for the probablility-matching approach.

In [ ]: