2018 Work Log

January

Week 1

Returned from vacation in s.e. asia on Monday. I started working with easy maintenance tasks on my code.

- Created a git `development` to keep the `master` branch stable at all times.
- Polished up the `cmake` build script so that libraries are either found or fetched and installed properly in a platform independent way. Took some trial and error.
- Added support for Travis-CI, so that a `git push` now triggers a build in `g++-7.2` and `clang++-5.0`, and reports pass/fail on the github page.
- Added support for reading parameters from a text file instead of hard-coding them into a namespace .
- Rewrite routines for finding files in the filesystem, making better use of `<std::experimental::filesystem>` features for finding input and output files.

Week 2

Continued on maintenance work. Almost finished

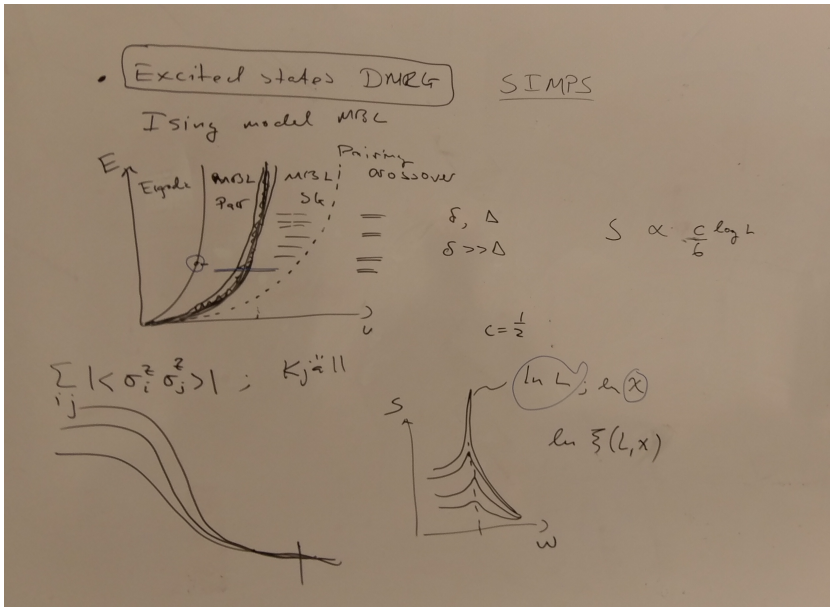
- Fixed some profiling issues. Timings were not clear enough.
- Fixed console output and implemented optional verbosity level, and optional timestamps.
- Started restructoring the algorithms using smart pointers everywhere, and using a Base class from which algorithms are derived (itebd, idmrg, fdmrg, etc). This modularization will speed up development time later, I believe.

Week 3

-

Week 4

This week I started a new project: excited-state DMRG. I am supposed to implement SIMPS or xDMRG and then study the MBL transition shown in the picture.



Other than that I continued the modularization, refactoring code. I'm trying to finish this as quickly as possible.

February

Week 1

- Finished modularization.
- Made another attempt to get Arpackc++ working to get eigenvalues of complex matrices. It worked!
- Added the excited states module and started working on that.

Week 2

- Changed the whole codebase to use complex numbers throughout. Having this double - complex switch is just messy, when complex numbers is "lingua franca" for MPS.
- The xDMRG procedure is quite simple! Simply find all eigenvectors to the local effective hamiltonian, i.e. $L-H-L$, where H is an MPO, and chose the eigenvector with greatest overlap to the current state. Starting with a randomly oriented spin chain this converges usually to a highly excited state. Unfortunalely it doesn't let us target a specific energy, but perhaps if we use SIMPS in the beginning to target a specific part of the spectrum,

we can then use xDMRG later to pinpoint a particular eigenstate in that region.

Week 3

- To gauge how close the algorithm is to an eigenstate, I returned to the issue of computing the variance $\langle \Delta H \rangle = \langle H^2 \rangle - \langle H \rangle^2$. The method by C.West using characteristic functions seems very promising... if I could just get it to work.

Week 4

- Continued working on variances.
- During the group meeting on Friday, a very fundamental error was pointed out to me. Essentially I hadn't been applying the unitary gates correctly; I applied them all, and then swapped. I am supposed to swap between every gate, and only then is a singla "tebd" step complete. It is amazing that I've gotten correct results all this time despite this flaw.

March

- The variance method by C.West works, almost. There is a discrepancy compared to the "double MPO" method I use for DMRG. Now I need to compute the variance using the regular method with the normal 2-site Hamiltonians. Turns out this is quite difficult, because the infinite number of crossterms need to be summed over. There are two places where this is described:
  - Zauner-Stauber, V., Vanderstraeten, L., Fishman, M. T., Verstraete, F., & Haegeman, J. (2017). Variational optimization algorithms for uniform matrix product states
  - Vanderstraeten, L. (n.d.). Tangent space methods for matrix product states
- Later in March: Succeeded in computing the variance in three different ways.
  - The MPO-way, this is simply contracting two layers of Hamiltonian MPO's at each step instead of just one (as with the energy). Doing so gives $\langle H^2 \rangle$. For `iDMRG` we can of course subtract the current energy to the MPO. For instance in the Ising model, we subtract it from the term at the lowermost left corner: $g\sigma^x \to g\sigma^x - eI$, where e is the per site energy and I is an identity matrix of the appropriate size.
  - The "Hamiltonian expansion" way, where we simply compute $\mathrm{Var}H = \langle (H-E)^2 \rangle$ directly. For the Ising model with nearest neighbor interactions this means that we get sums of 2-site terms squared, giving us an infinite number of cross-terms that have to be summed over or approximated assuming we are in the middle of an infinite chain.

This turned out to be quite difficult, as the infinite sum -part was usually badly explained where I found it.

- The "Characteristic function" way where we get the moments of the distribution from derivatives of $\exp(-iHt)$. This method is almost identical to applying time evolution operators to the MPS, and similarly, one has to do a Suzuki-Trotter decomposition of the Hamiltonian and apply the operators in interleaved layers, just as in iTEBD. After applying all the layers, one computes the overlap with the non-evolved MPS. This technique had a couple of pitfalls that weren't mentioned. For instance, one has to allow the bond dimension to grow between when applying the layers, otherwise the results are very messy. Still, to avoid performance issues there needs to be an upper limit (say, 2-5 times larger than the current bond dimension), but it's not clear how to choose a good upper limit.

All three of these techniques are explained in detail in the DMRG++ document, and were presented towards the end of March in a Friday-fika meeting with the group.

Interestingly the two latter techniques are very costly computationally. The whole reason for doing them was to have ways of corroborating the size of the error given in the MPO-way. Essentially these three all agree very well, with one caveat being that in iDMRG-mode the MPO-method takes a while to converge because it remembers bad states far away at the edges of the system, whereas the other two always assume translation invariance. Note that the MPO-method also works with finite chains. Furthermore the MPO-method, which was expected to work poorly due to the scaling of the bond dimension, is in practice very fast.

April

- There are still a couple of unresolved issues with iDMRG, namely the delayed convergence. This problem has lingered almost since the beginning and I guess I never truly understood what Jens meant for me to do about it. Time to solve this.
  - The issue arises from the fact that
- Started working on excited-state DMRG proper. There are a number of minor hurdles to address:
  - Full diagonalization of the effective Hamiltonian requires a lot of computing power.

May

June

- Vacation by the end of June: Midsummer and trip to fjällen

July

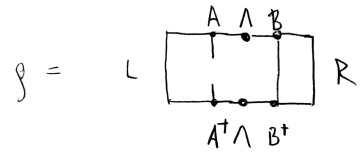
August

- Looking at the course offering for PhD students. There are several interesting courses, that I'll list here:
  - SI3200 Kondenserade materiens teori 7,5 hp by Mats Wallin
  - SI3260 Kvanttransport 7,5 hp by Jack Lidmar. It says the course is no longer available but perhaps one could ask nicely.
  - LH3000 Basic Communication and Teaching 3.0 credits by Mikael Cronhjort. I think this course is mandatory for teaching? I could be wrong.
  - SH3000 Bredd och etik i fysik 7,5 hp by Pär Olsson. I think this course is mandatory for all PhD students. I already have 1/4 modules cleared. But I need to register properly.
- Some thoughts about courses:
  - Perhaps Jens can "make-up" a reading course for me? I've heard that's something that has been done before. Perhaps then the topic may be even more relevant to my research or interests.
  - For SH3000, how do I register? Perhaps talk to Pär...
  - For LH3000, is this mandatory for teaching? Ask Pär...
  - Introduction to High Performance Computing 7,5 hp Registration for this summer school starts in April.
- Looking at the study plan (eISP) I noticed that my courses counted from the master do not appear. Ask Pär about this?
- Discussion with Loic on what to do with the excited states of the self-dual Ising model. First we discussed the following
  - I should check the sign of J_μ, make sure it's actually the correct one.
  - It makes sense to have $\overline{\log J} = \log J_\mu$ and similarly for h_μ.
  - The overlap threshold can be set somewhere below $\frac{1}{\sqrt{2}}$ in case that there are cat states.
  - If the overlap threshold is not reached, how about making a full diag? What would happen then?
  - It might not be worthwile initializing the chain close to energy target first. A random chain will probably be close to the middle anyway.

  Measure the following ($^\circ$ means x, y, z):

  1. Average entanglement entropy S across the chain. Take the average over many realizations. Basically plot the average of S(l), where l is the position on the chain.

2. $\frac{1}{L}|\langle\sigma^\circ\rangle|$ (or square instead of abs).
3. $\langle\sigma_i^\circ\sigma_j^\circ\rangle$. This is the same as the one mentioned in Jens paper, i.e. the spin-glass observable. In the paper it is expressed as $\frac{1}{L}\sum_{i,j}\langle n|\sigma_i^\circ\sigma_j^\circ|n\rangle$.
4. $\frac{1}{L}\sum_{i,j}\langle\sigma_i^\circ\sigma_j^\circ\rangle - \langle\sigma_i^\circ\rangle\langle\sigma_j^\circ\rangle$.
5. Compute the single site density matrix in two ways according to the figure. Check that they are equivalent. The eigenvalues λ_i of



$$\rho = \frac{1}{2}\left(I + n_x\sigma^x + n_y\sigma^y + n_z\sigma^z\right)$$
$$= \frac{1}{2}\begin{pmatrix} 1+n_z & n_x - in_y \\ n_x + in_y & 1-n_z \end{pmatrix}$$

Figure 1: The hermitian density matrix ρ satisfies $\mathrm{Tr}(\rho) = 1$, the numbers n_x, n_y, n_z are real and $\det\rho = (1-|\mathbf{n}|^2)/4$. Since ρ is positive we have $1 \geq |\mathbf{n}|$. We can represent the density matrix ρ by the vector $\mathbf{n}$, which lies in the unit ball. If ρ is a pure state, its corresponding vector will lie on the surface of the sphere. Likewise if $|\mathbf{n}| = 1$, then $\mathrm{Tr}(\rho) = 1$ and $\det\rho = 0$. This implies that one of the eigenvalues is zero and the other is one.

the density matrix ρ can be used to compute the single-site entropy $S = -\sum_i \lambda_i \log\lambda_i$. Expectation values can be computed like $\langle\sigma^\circ\rangle = \mathrm{Tr}(\rho\sigma^\circ)$.

–

August

- Computing cluster:

- After seeing how much coputational power was required to do my simulations, I realized I'd need a computing cluster. I started having a look at HTCondor, which I had failed to use previously.

September

- Computing cluster:
  - Over the past month I've been working with my HTCondor cluster Issues I looked at:
    * **Users:** There are multiple requirements for users:
      · Different users should be recognized by condor to enable priorities and usage monitoring.
      · Each user needs a home with write permissions, but only read permissions or less in other condor users folders. For this each user needs to exist on all the nodes with synchronized UID's and GID's. I also looked at Kerberos and LDAP but these were far too complciated.
      · Home folders should be present in each node. For this we need a shared filesystem. I looked at NFS, NFSv4, SMB and sshfs but in the end NFS seemed simplest and most reliable.
    * **Shared filesystem:** HTcondor has a file transfer mechanism so that all listed files are sent from the submitting machine to the nodes and back for each simulation. This is a lot of hassle if the executable depends on many files in a large filestructure. I ended up opting for NFSv4 and letting all the nodes mount it using AutoFS. Now all nodes have an automounted condorusers folder containing each users home folder, mounted from the network. All the permissions are set on the host machine. Having a shared filesystem drastically reduced maintenence, since now all the settings can also be read from configurefiles in `/mnt/nfs/condorfs/common`, which lies on the host nfs server.
    * **Credentials** HTCondor supports host or user based authentication and a number of others like kerberos. In the end SSL seemed simplest, if one wants to have user based credentials instead. I set up my own certificate authority which signs credentials for new nodes.
    * ---
    * **Library availability** Ended up using environment-modules available from apt. This can load libraries exactly like in Octupus, Triolith and PDC clusters. Now I have to maintain modulefiles and library files in `/mnt/nfs/condorfs/modulefiles` and `/mnt/nfs/condorfs/software/ libraries` , but I have automated scripts to install new versions when needed, so this is not a lot of work. Luckily the libraries are automatically loaded on remote nodes because condor can copy the current PATH and

LD*LIBRARY*PATH to the executing node using the variable "getenv = true" in the condor submit script.

    * **Heterogeneous machines:** Binary compatibility is tough to achieve when there are different machines on all the
    * **Shared/static libraries**

*

*

- Went to Max Planck institute in Dresden for a workshop during 23rd to 29th of september. It was interesting. Learned a lot, worked a lot and met some other students researching MBL and related topics. Short summary:
  - LCRG method for doing infinite system time evolution can be done to study mbl somehow. The point was that disorder wasn't really needed explicitly. Maximilian Keifer explained this during a poster session.
  -