Notes on Variation Uniform Matrix Product States (VUMPS)

Jan Reimers

March 2023

1 Introduction

Infinite lattices methods assume a uniform sequence of Matrices in the iMPS. Variational methods for infinite lattice systems were introduced by Stauber et. al. [3] followed by some lecture notes [2] which we will be following here. As usual for software library work, we must support multi-site unit cells for both the iMPS and iMPO. This is partially covered in [3] and we fill in some gaps and interactions with iMPO compression in this document.

2 Definitions for Multi-site unit cells.

The extent of the unit cell for the MPS is part of the MPS ansatz and it is up the user to decide the size, N of that unit cell. Conversely the size of the unit cell for the MPO, N_W is part of the problem definition. There is no reason why these cells size should coincide, so we must code for $N \neq N_W$. What we will do in code is ensure that $N = L * N_W$ where $L \in \mathbb{N}$, i.e. the MPS repeat distance is some integer multiple of the MPO repeat distance. In other words incommensurate systems are not directly supported yet, but they can be approximated through large L.

2.1 MPS

We can define a unit cell MPS for unit cell n

$$\mathbb{A}^{\mathbb{S}_n} = \prod_{k-1}^{N} A\left(k\right)^{s_{n,k}} = A\left(1\right)^{s_{n,1}} A\left(2\right)^{s_{n,2}} \cdots A\left(N-1\right)^{s_{n,N-1}} A\left(N\right)^{s_{n,N}} \tag{1}$$

where we index the site with round brackets (k). One quickly runs out of space to locate indices in this business! The orthonormal forms are the same

$$\sum_s A\left(k\right)_L^{\dagger s} A\left(k\right)_L^s = \mathbb{I}, \quad \sum_s A\left(k\right)_R^s A\left(k\right)_R^{\dagger s} = \mathbb{I}, \quad k = 1 \ldots N \tag{2}$$

But the right and left eigen vector R, L must be modified in such away that we get interlacing across the unit cell:

$$\sum_s A\left(k\right)_L^{*s} R\left(k\right) A\left(k\right)_L^s = R\left(k-1\right) \tag{3}$$

$$\sum_s A\left(k\right)_R^s L\left(k-1\right) A\left(k\right)_R^{*s} = L\left(k\right) \tag{4}$$

Or in more compact form

$$\left(\mathbb{I}\left(k\right)\right| = \left(\mathbb{I}\left(k-1\right)\right| T\left(k\right)_L \tag{5}$$

$$\left|\mathbb{I}\left(k-1\right)\right) = T\left(k\right)_R \left|\mathbb{I}\left(k\right)\right) \tag{6}$$

$$\left|R(k-1)\right) = T\left(k\right)_L \left|R(k)\right) \tag{7}$$

$$\left(L\left(k\right)\right| = \left(L\left(k-1\right)\right| T\left(k\right)_R \tag{8}$$

Where we have also specified the k index in the identity operators. The gauge relations now become:

$$A\left(k\right)_C^s = A\left(k\right)_L^s C\left(k\right) = C\left(k-1\right) A\left(k\right)_R^s \tag{9}$$

2.2 MPO

For the MPO we declare the unit cells size to be N_W

$$\mathbb{W}^{\mathbb{S}_n\mathbb{S}'_n} = \prod_{k=1}^{N_W} \hat{W}(k)^{s_{n,k}s'_{n,k}} \tag{10}$$

The one site MPO transfer matrices can be defined

$$T(k)^W_{Lab} = \sum_{ss'} W(k)^{ss'}_{ab} A(k)^{*s}_L \otimes A(k)^{s'}_L \quad k = 1 \dots N_W \tag{11}$$

$$T(k)^W_{Rab} = \sum_{ss'} W(k)^{ss'}_{ab} A(k)^{s}_R \otimes A(k)^{*s'}_R \quad k = 1 \dots N_W \tag{12}$$

In addition it is also useful to define the full unit cell MPO transfer matrices

$$\mathbb{T}(k)^W_L = \prod_{k'=k+1}^{k+N} T(k')^W_L \tag{13}$$

$$\mathbb{T}(k)^W_R = \prod_{k'=k+1}^{k+N} T(k')^W_R \tag{14}$$

We also have the same interlaced eigen vector relations as for the MPS:

$$(\mathbb{I}(k)| = (\mathbb{I}(k-1)|\, T(k)^W_L \tag{15}$$

$$|\mathbb{I}(k-1)) = T(k)^W_R\, |\mathbb{I}(k)) \tag{16}$$

$$\left|R(k-1)^W\right) = T(k)^W_L \left|R(k)^W\right) \tag{17}$$

$$\left(L(k)^W\right| = \left(L(k-1)^W\right| T(k)^W_R \tag{18}$$

3 Eigen vectors of the MPO transfer matrix

The VUMPS algorithm requires the eigenvectors defined in eq. 17 and 18. But because of the interlacing relations throughout the unit cell, these are not actually eigen equations. To get and eigen system we must use the full unit cell transfer matrices 13 and 14.

$$\left(L(k)^W\right| = \left(L(k)^W\right| \mathbb{T}(k)^W_L \tag{19}$$

$$\left|R(k)^W\right) = \mathbb{T}(k)^W_R \left|R(k)^W\right) \tag{20}$$

In practice we only need to solve this for one site, say $k = 1$ and then use 17 and 18 the calculate $\overrightarrow{L}(k)^W$ or $\overrightarrow{R}(k)^W$ for the other $k \neq 1$. In practice we do not store $T(k)^W_{L/R}$ or $\mathbb{T}(k)^W_{L/R}$ but instead evaluate required products with L/R on the fly.

The procedure also leverages the lower triangular structure of the iMPO operators $\hat{W}$ as described in [3].

3.1 Interaction with compression

In general, orthogonalization of and compression ([1]) results in an iMPO with reduced size, but in the process *the triangular structure gets destroyed*. So how do we get the benefit of reduced iMPO size in the VUMPs iterations? At this point I see two options

1. Brute force: Use an iterative eigen solver to solve eq. 19
2. Solve eq. 19 using the uncompressed triangular iMPO, and then gauge transform the resulting L and R environments into the new compressed basis.

It is not a priori obvious which approach is more efficient in terms of storage and processing. It probably makes sense to try both and investigate conditions for which each option is optimal.

In order to implement option 2 above, we first review the gauge relations for orthogonalization and compression of iMPOs. For left and right orthogonalization we have respectively:

$$G(k-1)_L \hat{W}(k)_0 = \hat{W}(k)_L G(k)_L \tag{21}$$

$$G(k)_R \hat{W}(k)_R = \hat{W}(k)_0 G(k+1)_R \tag{22}$$

where $\hat{W}(k)_0$ are the uncompressed/triangular iMPO operators. Compression works on the full gauge transform

$$G(k) = G(k)_L G(k+1)_R \approx \tilde{U}(k) \tilde{s}(k) \tilde{V}(k) \tag{23}$$

where

$$G(k-1) \hat{W}(k)_R = \hat{W}(k)_L G(k) \tag{24}$$

and $\tilde{s}(k)$ is already truncated. The compressed Ws look like

$$\hat{W}(k)_{R'} = \tilde{V}(k-1) \hat{W}(k)_R \tilde{V}(k)^\dagger \tag{25}$$

$$\hat{W}(k)_{L'} = \tilde{U}(k-1)^\dagger \hat{W}(k)^n_L \tilde{U}(k)^n \tag{26}$$

If we specify a particular process, such as

1. Right orthogonalize with rank reduction.
2. Left orthogonalize with rank reduction.
3. Compress and use $\hat{W}_{L'}$ for VUMPS. Call $\hat{W}_{L'} = \hat{W}_{comp}$ to avoid overloading the use and meaning of L

then we can relate

$$\begin{aligned}\hat{W}(k)_{comp} &= \tilde{U}(k-1)^\dagger \hat{W}(k)_L \tilde{U}(k) \\ &= \tilde{U}(k-1)^\dagger G(k-1)_L \hat{W}(k)_R G(k)^{-1}_L \tilde{U}(k) \\ &= \tilde{U}(k-1)^\dagger G(k-1)_L G(k)^{-1}_R \hat{W}(k)_0 G(k+1)_R G(k)^{-1}_L \tilde{U}(k).\end{aligned}$$

Combining factors

$$\hat{W}(k)_{comp} = G(k-1)^{-1}_{full} \hat{W}(k)_0 G(k)_{full} \tag{27}$$

where

$$G(k)_{full} = G(k+1)_R G(k)^{-1}_L \tilde{U}(k)^n.$$

$G(k)_{full}$ should be rectangular in such a way (tall) that $\hat{W}(k)_{comp}$ is smaller than $\hat{W}(k)_0$.

For the iMPO transfer matrices, the same relation as eq. 27 holds:

$$T(k)^W_{Lcomp} = G(k-1)^{-1}_{full} T(k)^W_{L0} G(k)_{full}$$

$$T(k)^W_{Rcomp} = G(k-1)^{-1}_{full} T(k)^W_{R0} G(k)_{full}$$

also for the full unit cell

$$\begin{aligned}\mathbb{T}(k)^W_{Lcomp} &= G(k)^{-1}_{full} \mathbb{T}(k)^W_{L0} G(k)_{full} \\ \mathbb{T}(k)^W_{Rcomp} &= G(k)^{-1}_{full} \mathbb{T}(k)^W_{R0} G(k)_{full}\end{aligned} \tag{28}$$

because all the interleaved gauge transforms in the product will cancel.

Finally we can transform eq. 19 as

$$G(k)^{-1}_{full} \overrightarrow{L}(k)^W G(k)_{full} = G(k)^{-1}_{full} \overrightarrow{L}(k)^W G(k)_{full} G(k)^{-1}_{full} \mathbb{T}(k)^W_L G(k)_{full}$$

or

$$\overrightarrow{L}(k)^W_{comp} = \overrightarrow{L}(k)^W_{comp} \mathbb{T}(k)^W_{Lcomp}$$

where

$$\overrightarrow{L}(k)^W_{comp} = G(k)^{-1}_{full} \overrightarrow{L}(k)^W G(k)_{full} \tag{29}$$

Similarly

$$G(k)^{-1}_{full} \left| \overrightarrow{R}(k)^W \right) G(k)_{full} = G(k)^{-1}_{full} \mathbb{T}(k)^W_R G(k)_{full} G(k)^{-1}_{full} \left| \overrightarrow{R}(k)^W \right) G(k)_{full}$$

$$\overrightarrow{R}(k)^W_{comp} = \mathbb{T}(k)^W_{Rcomp} \overrightarrow{R}(k)^W_{comp}$$

where

$$\overrightarrow{R}(k)^W_{comp} = G(k)^{-1}_{full} \overrightarrow{R}(k)^W G(k)_{full} \tag{30}$$

3.2 Algorithm

Input is the wave function $\mathbb{A}^{\mathbb{S}_n}$, eq. 1 and the unmodified/triangular iMPO $\mathbb{W}^{\mathbb{S}_n\mathbb{S}'_n}$ eq. 10.

1. Right orthogonalize with rank reduction. Save $G(k)_R$.
2. Left orthogonalize with rank reduction. Save $G(k)_L$
3. SVD compress $G(k) = G(k)_L G(k+1)_R$. Save $\tilde{U}(k)$
4. Evaluate and save $G(k)_{full} = G(k+1)_R G(k)_L^{-1} \tilde{U}(k)^n$
5. Calculate $G(k)_{full}^{-1}$. It should be non-singular.
6. Start the VUMPs algorithm
   (a) Solve 19 and 20 to get $\left(\overrightarrow{L}(k)^W\right|$ and $\left|\overrightarrow{R}(k)^W\right)$ using the triangular structure. Should only require one inversion.
   (b) Use the transforms 29 and 30 to get the eigen vector environments in the compressed iMPO basis.
   (c) VUMPS only sees $\overrightarrow{L}(k)^W_{comp}$, $\overrightarrow{R}(k)^W_{comp}$ and $\hat{W}(k)_{comp}$

4 References

References

[1] Daniel E. Parker, Xiangyu Cao, and Michael P. Zaletel. Local matrix product operators: Canonical form, compression, and control theory. *Phys. Rev. B 102, 035147 (2020)*, 2019.

[2] Laurens Vanderstraeten, Jutho Haegeman, and Frank Verstraete. Tangent-space methods for uniform matrix product states. *SciPost Physics Lecture Notes*, jan 2019.

[3] V. Zauner-Stauber, L. Vanderstraeten, M. T. Fishman, F. Verstraete, and J. Haegeman. Variational optimization algorithms for uniform matrix product states. *Physical Review B*, 97(4):045145, jan 2018.