The Journal of Open Source Software

PyTeNet: A concise Python implementation of quantum tensor network algorithms

Christian B. Mendl[1]

1 Technische Universität Dresden, Institute of Scientific Computing, Zellescher Weg 12-14, 01069 Dresden, Germany

DOI: 10.21105/joss.00948

Software

- Review
- Repository
- Archive

Submitted: 14 September 2018
Published: 16 October 2018

Summary

The simulation of strongly correlated quantum systems on classical computers is an essential tool for investigating and understanding quantum materials. Due to the inherent "curse of dimensionality" of quantum systems, such simulations pose a challenging task, and have spurred the development of several computational approaches and approximations. For quasi one-dimensional lattice systems, the density matrix renormalization group (DMRG) framework (Schollwöck, 2005; White, 1992, 1993) has emerged as one of the most successful methods. Modern formulations are based on matrix product states (MPS) and operators (MPO) (Schollwöck, 2011): briefly speaking, a matrix product state is an Ansatz for the quantum many-body wavefunction amendable to efficient numerical manipulation. The favorable approximation properties of matrix product states have been theoretically proven for ground states of gapped Hamiltonians with local interactions (Hastings, 2007; Verstraete & Cirac, 2006), and recent work has extended this to quantum systems at finite temperature (Barthel, 2017). Over the years many variations and generalizations of the DMRG framework have been devised (Barthel, 2013; Stoudenmire & White, 2010; Verstraete, Murg, & Cirac, 2008; Vidal, 2003, 2008), and further developments and improvements of associated algorithms remain an active field of research.

`PyTeNet` is a concise Python implementation of the core MPS framework; besides its use for research on quantum system, it serves as tool for algorithmic experimentation, as didactic reference and as blueprint for future implementations using, e.g., compiled languages to optimize computational efficiency. Specifically, `PyTeNet` employs NumPy (Walt, Colbert, & Varoquaux, 2011) for tensor manipulations, and is structured around MPS and MPO classes and associated operations, like left- or right-orthonormalization, computing expectation values, local gate operations etc. Features of `PyTeNet` include the conversion of arbitrary operator chains (like $\sigma_i^x \sigma_{i+1}^x + \sigma_i^y \sigma_{i+1}^y + \Delta \sigma_i^z \sigma_{i+1}^z$) to MPOs, which facilitates the concise construction of common quantum Hamiltonian operators. `PyTeNet` can also generate vector and matrix representations of matrix product states and operators, respectively, thus enabling reference calculations for example. Another major feature is the (real and imaginary) time evolution based on the time-dependent variational principle (TDVP) (Haegeman, Lubich, Oseledets, Vandereycken, & Verstraete, 2016), making use of Krylov subspace methods for local operations (Hochbruck & Lubich, 1997). Energy minimization and ground state computation are achieved via one-site local energy minimization using Lanczos iteration. `PyTeNet` unobtrusively supports additive quantum numbers by associating an integer with each physical and virtual bond dimension, and uses corresponding reference checks on the sparsity patterns of tensors. Several simultaneously conserved quantum numbers (like particle and spin for the Fermi-Hubbard model) are then realized by encoding them as single integers.

TDVP time evolution example usage:

```
import pytenet as ptn

# number of lattice sites (1D with open boundary conditions)
L = 10

# construct matrix product operator representation of
# Heisenberg XXZ Hamiltonian (arguments are L, J, \Delta, h)
mpoH = ptn.heisenberg_XXZ_MPO(L, 1.0, 0.8, -0.1)
mpoH.zero_qnumbers()

# initial wavefunction as MPS with random entries
# maximally allowed virtual bond dimensions
D = [1, 2, 4, 8, 16, 28, 16, 8, 4, 2, 1]
psi = ptn.MPS(mpoH.qd, [Di*[0] for Di in D], fill='random')
# effectively clamp virtual bond dimension of initial state
Dinit = 8
for i in range(L):
    psi.A[i][:, Dinit:, :] = 0
    psi.A[i][:, :, Dinit:] = 0
psi.orthonormalize(mode='left')

# time step can have both real and imaginary parts;
# for real time evolution use purely imaginary dt!
dt = 0.01 - 0.05j
numsteps = 100

# run TDVP time evolution
ptn.integrate_local_singlesite(mpoH, psi, dt, numsteps, numiter_lanczos=5)
# psi now stores the (approximated) time-evolved state exp(-dt*numsteps H) psi
```

Acknowledgments

I'd like to thank Thomas Barthel, Lexing Ying, Miles Stoudenmire and Christian Lubich for inspiring discussions.

References

Barthel, T. (2013). Precise evaluation of thermal response functions by optimized density matrix renormalization group schemes. *New J. Phys.*, *15*, 073010. doi:10.1088/1367-2630/15/7/073010

Barthel, T. (2017). One-dimensional quantum systems at finite temperatures can be simulated efficiently on classical computers. *arXiv:1708.09349*. Retrieved from https://arxiv.org/abs/1708.09349

Haegeman, J., Lubich, C., Oseledets, I., Vandereycken, B., & Verstraete, F. (2016). Unifying time evolution and optimization with matrix product states. *Phys. Rev. B*, *94*, 165116. doi:10.1103/PhysRevB.94.165116

Hastings, M. B. (2007). An area law for one-dimensional quantum systems. *JSTAT*, *2007*(08), P08024. doi:10.1088/1742-5468/2007/08/P08024

Hochbruck, M., & Lubich, C. (1997). On Krylov subspace approximations to the matrix exponential operator. *SIAM J. Numer. Anal.*, *34*, 1911–1925. doi:10.1137/S0036142995280572

Schollwöck, U. (2005). The density-matrix renormalization group. *Rev. Mod. Phys.*, *77*, 259–315. doi:10.1103/RevModPhys.77.259

Schollwöck, U. (2011). The density-matrix renormalization group in the age of matrix product states. *Ann. Phys.*, *326*, 96–192. doi:10.1016/j.aop.2010.09.012

Stoudenmire, E. M., & White, S. R. (2010). Minimally entangled typical thermal state algorithms. *New J. Phys.*, *12*, 055026. doi:10.1088/1367-2630/12/5/055026

Verstraete, F., & Cirac, J. I. (2006). Matrix product states represent ground states faithfully. *Phys. Rev. B*, *73*, 094423. doi:10.1103/PhysRevB.73.094423

Verstraete, F., Murg, V., & Cirac, J. I. (2008). Matrix product states, projected entangled pair states, and variational renormalization group methods for quantum spin systems. *Adv. Phys.*, *57*, 143–224. doi:10.1080/14789940801912366

Vidal, G. (2003). Efficient classical simulation of slightly entangled quantum computations. *Phys. Rev. Lett.*, *91*, 147902. doi:10.1103/PhysRevLett.91.147902

Vidal, G. (2008). Class of quantum many-body states that can be efficiently simulated. *Phys. Rev. Lett.*, *101*, 110501. doi:10.1103/PhysRevLett.101.110501

Walt, S. van der, Colbert, S. C., & Varoquaux, G. (2011). The NumPy array: A structure for efficient numerical computation. *Comput. Sci. Eng.*, *13*, 22–30. doi:10.1109/MCSE.2011.37

White, S. R. (1992). Density matrix formulation for quantum renormalization groups. *Phys. Rev. Lett.*, *69*, 2863–2866. doi:10.1103/PhysRevLett.69.2863

White, S. R. (1993). Density-matrix algorithms for quantum renormalization groups. *Phys. Rev. B*, *48*, 10345–10356. doi:10.1103/PhysRevB.48.10345