Positive bias makes tensor-network contraction tractable

Chris (Jielun) Chen
(Caltech)

Jiaqing Jiang
(Caltech)

Norbert Schuch
(University of Vienna)

Dominik Hangleiter
(Simons Institute)

arXiv:2410.05414

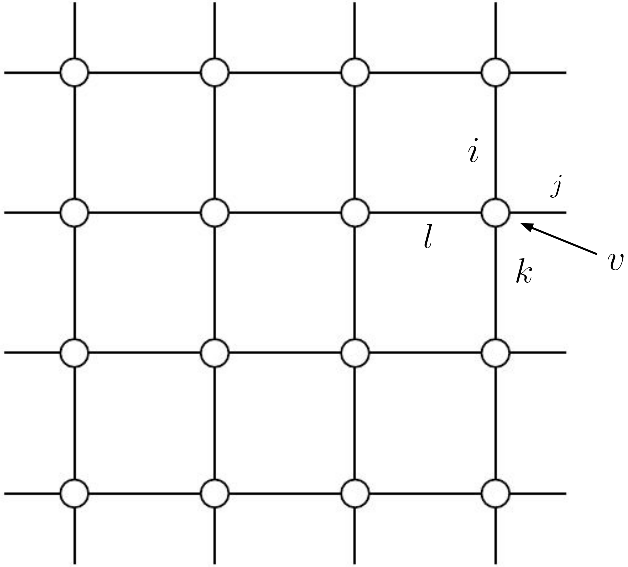
Caltech

SIMONS INSTITUTE
for the Theory of Computing

universität
wien

Tensor networks (TN)

2D square-lattice graph with n vertices

$\mathcal{T} =$ 

Edges: $i \in \{1, ..., d\}$, d is called bond-dimension

Vertices: M^v_{ijkl} , order-4 tensor

Edge labeling: an assignment of values to all edges

e.g. $c = (..., i = 3, j = 2, k = 1, l = 1, ...)$

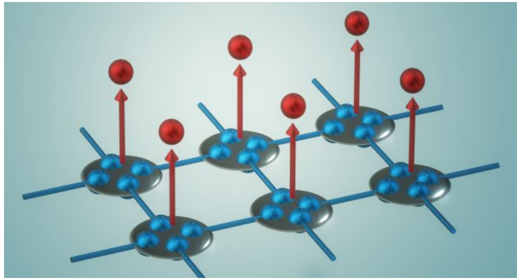
Contracted value:

$$\chi(\mathcal{T}) = \sum_{\text{edge labeling } c} \prod_v M^v_c$$

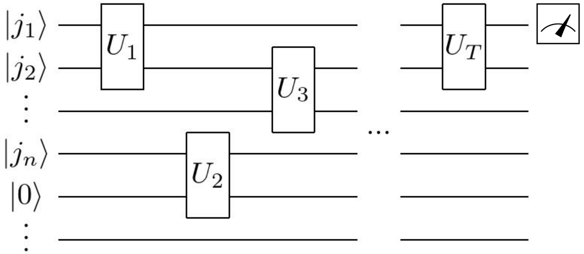
Tensor networks (TN)

Contracting a tensor network is a common computational task with many applications.

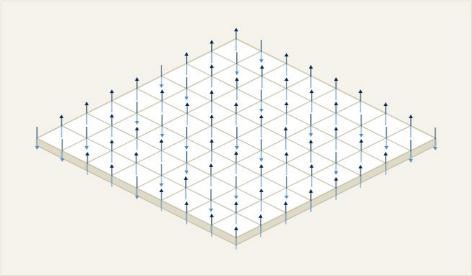
Many-body physics



Simulating quantum circuits



Classical stat mech



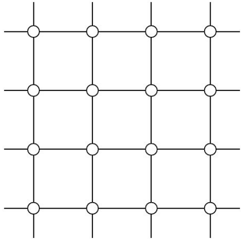
...

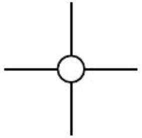
Complexity of (2D) tensor network contraction

	Exact	Approximate
Worst-case	#P-hard [SWVC07]	
Average-case		

Complexity of (2D) tensor network contraction

	Exact	Approximate
Worst-case	#P-hard [SWVC07]	
Average-case	#P-hard [HHEG20]	

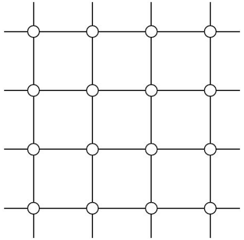


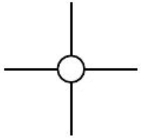
 $\sim \mathcal{N}(0, 1)$

i.i.d. for all entries & all tensors.

Complexity of (2D) tensor network contraction

	Exact	Approximate
Worst-case	#P-hard [SWVC07]	Empirically hard [GSH+23]
Average-case	#P-hard [HHEG20]	

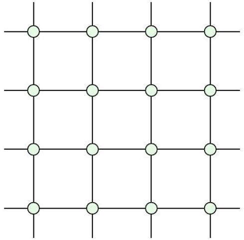


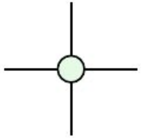
 $\sim \mathcal{N}(0, 1)$

i.i.d. for all entries & all tensors.

Complexity of (2D) tensor network contraction

	Exact	Approximate
Worst-case	#P-hard [SWVC07]	Empirically hard [GSH+23]
Average-case ($\mu = 0$)	#P-hard [HHEG20]	
Average-case + small positive bias ($0 < \mu \ll 1$)	#P-hard [our result]	Quasi-poly time algorithm [our result]



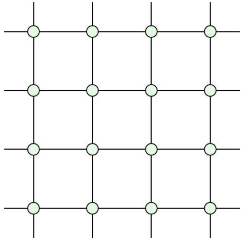


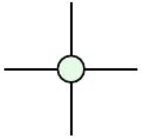
$\sim \mathcal{N}(\mu, 1)$

i.i.d. for all entries & all tensors.

Complexity of (2D) tensor network contraction

	Exact	Approximate
Worst-case	#P-hard [SWVC07]	Empirically hard [GSH+23]
Average-case ($\mu = 0$)	#P-hard [HHEG20]	
Average-case + small positive bias ($0 < \mu \ll 1$)	#P-hard [our result]	Quasi-poly time algorithm [our result]





$\sim \mathcal{N}(\mu, 1)$

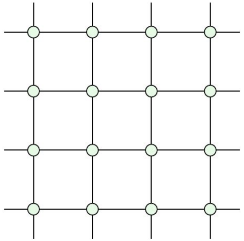
i.i.d. for all entries & all tensors.

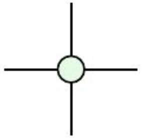
Main theorem

Theorem (informal): For a random 2D tensor network, if

$$\boldsymbol{\mu \gtrsim 1/d}, \qquad \boldsymbol{d \gtrsim n}$$

then with high probability, there exists a quasi-polynomial time algorithm which approximates the TN contraction value up to arbitrary 1/poly multiplicative error.



 $\sim \mathcal{N}(\mu, 1)$

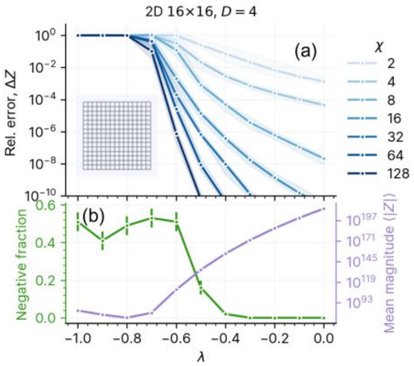
i.i.d. for all entries & all tensors.

Motivation

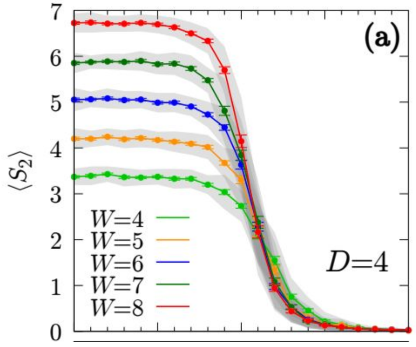
Rigorous study for how sign structure influence contraction complexity.

- Folklore: “positivity makes TN easier to contract”.
- Numerical simulations: the complexity drops significantly when the TN is only “slightly positive”, and there is a phase transition point.

[GC22]



[CJHS24]



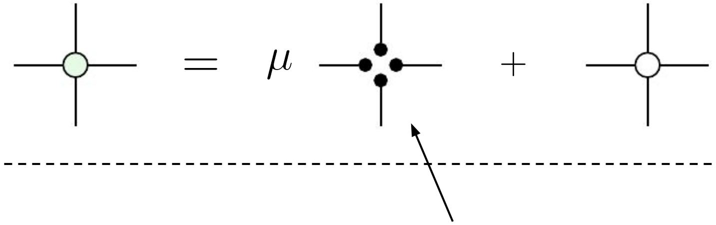
The threshold $\mu \gtrsim 1/d$ in our result matches the phase transition point predicted in [CJHS24], even though the contraction method is completely different!

Method overview

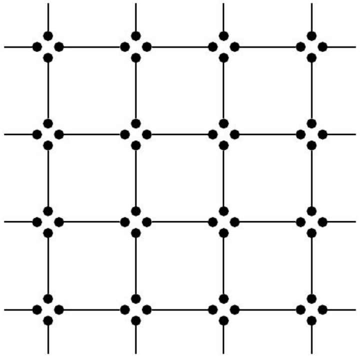
Easy — Interpolate → Hard

$\mu \to \infty$ $\qquad$ $\mu = 1/d$

$$\mathcal{N}(\mu, 1) \quad = \quad \mu \quad + \quad \mathcal{N}(0, 1)$$



all-one tensor



Easy to contract!

Method overview

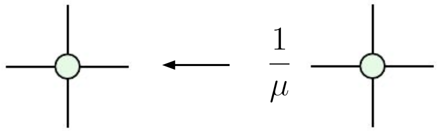
Easy — Interpolate → Hard

$\mu \to \infty$ $\quad\quad\quad$ $\mu = 1/d$

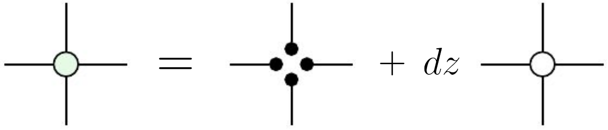
$\boxed{z = 0}$ $\quad\quad\quad$ $\boxed{z = 1}$

For convenience later, introduce change of variable $\boxed{z = \frac{1}{\mu d}}$

and rescale the tensors. No effect on multiplicative error.

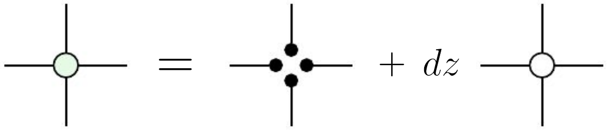


New definition



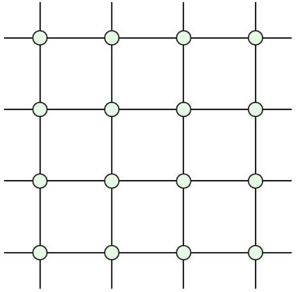
Method overview

To interpolate from z = 0 to 1, we uses **Barvinok's method [Bar16].** It relies on two observations.



Observation 1

Contracted TN is a degree-n random polynomial on z, denoted as g(z).



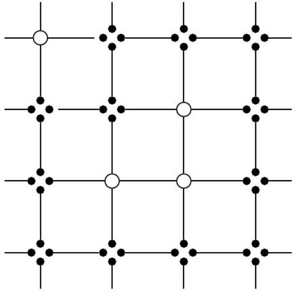
$$= g(z)$$

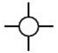
Observation 2

k-th derivative of g(z) at z = 0 can be computed brute-forcely in $n^{O(k)}$ time.

e.g.

$$\frac{1}{4!}\frac{\partial^4 g(z)}{\partial z^4}\Big|_{z=0} =$$



\+ all other configs with four 

Barvinok's method [Bar16]

Originally designed for permanent

Input: degree-n polynomial $g(z)$

$g^{(k)}(0)$ accessible in $n^{O(k)}$ time

Output: $g(1)$, ε multiplicative error

Algorithm: $f(z) = \ln(g(z))$, now ε additive error

$f(1) \approx \hat{f}(1) = \sum_{k=0}^{M} \frac{f^{(k)}(0)}{k!}$, output $e^{\hat{f}(1)}$, that's it!

Effectiveness depends on the roots of $g(z)$

Barvinok's method [Bar16]

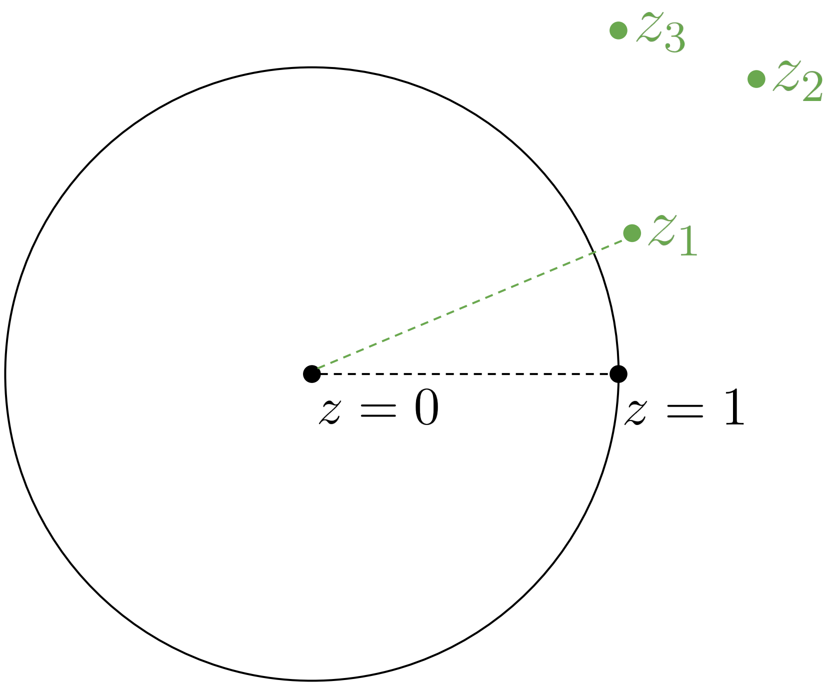
Roots of $g(z)$ are outside the $|z| = 1$ disk $\rightarrow$ small error

$$f(z) = \ln(g(z))$$

$$f(1) \approx \hat{f}(1) = \sum_{k=0}^{M} \frac{f^{(k)}(0)}{k!}$$

$$M = O\left(\ln\left(\frac{n}{\epsilon}\right)\right)$$

Run time $n^{O(M)}$, quasi-poly

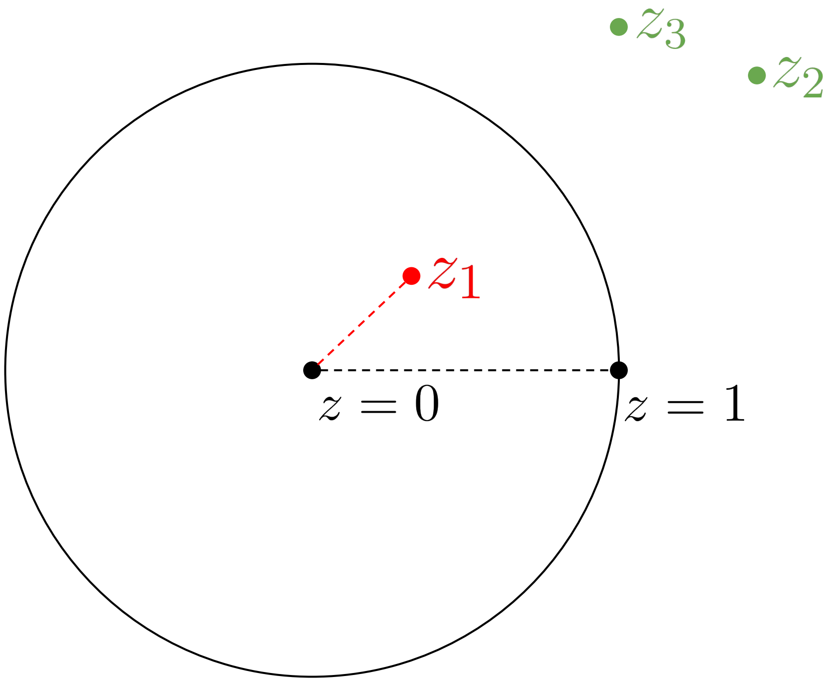


Barvinok's method [Bar16]

A root of $g(z)$ is inside the $|z| = 1$ disk $\longrightarrow$ error blows up!

$$f(z) = \ln(g(z))$$

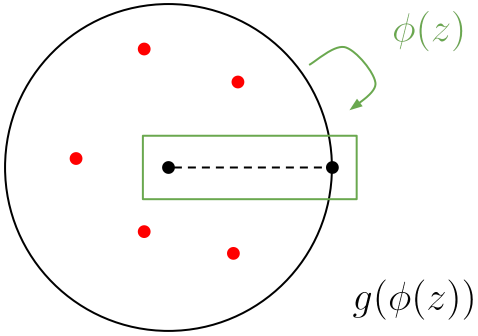
$$f(1) \approx \hat{f}(1) = \cancel{\sum_{k=0}^{M}} \frac{f^{(k)}(0)}{k!}$$



Barvinok's method via root-free path

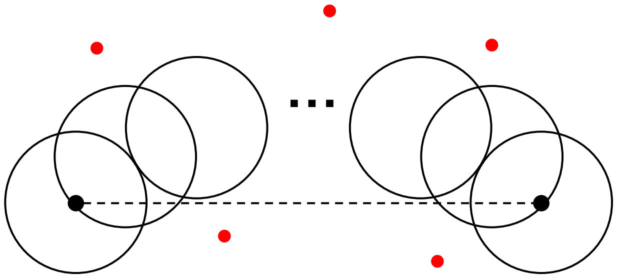
Not many roots (e.g. O(1)) → can interpolate along a root-free path!

Method 1 [Bar16]



Mapping the disk to a strip

Method 2 [EM18]



"Algorithmic" analytic continuation

Both remain quasi-polynomial time.

Barvinok's method on random tensor network

$$f(z) = \ln(g(z))$$

$$f(1) \approx \hat{f}(1) = \sum_{k=0}^{M} \frac{f^{(k)}(0)}{k!}$$

$\longrightarrow$ few roots $\longrightarrow$ find a root-free path $\longrightarrow$ quasi-poly algorithm

Barvinok's method on random tensor network

$$f(z) = \ln(g(z))$$

$$f(1) \approx \hat{f}(1) = \sum_{k=0}^{M} \frac{f^{(k)}(0)}{k!}$$

$\longrightarrow$ few roots on average $\longrightarrow$ find a root-free path with high probability $\longrightarrow$ quasi-poly algorithm that successes with high probability

Barvinok's method on random tensor network

$$f(z) = \ln(g(z))$$

$$f(1) \approx \hat{f}(1) = \sum_{k=0}^{M} \frac{f^{(k)}(0)}{k!}$$

$\longrightarrow$ few roots on average $\longrightarrow$ find a root-free path with high probability $\longrightarrow$ quasi-poly algorithm that successes with high probability

How to bound average number of roots? $\longrightarrow$

$$\mathbb{E}_g[N_{r(1-\delta)}] \leq \frac{1}{2\delta}\mathbb{E}_g\left[\oint_r \log\left(\left|\frac{g(z)}{g(0)}\right|^2\right) dz\right]$$

Jensen's formula

$$\leq \frac{1}{2\delta}\log\left(\oint_r \mathbb{E}_g\left|\frac{g(z)}{g(0)}\right|^2 dz\right)$$

Jensen's inequality

[EM18]

Barvinok’s method on random tensor network

$$f(z) = \ln(g(z))$$

$$f(1) \approx \hat{f}(1) = \sum_{k=0}^{M} \frac{f^{(k)}(0)}{k!}$$

$\longrightarrow$ few roots on average $\longrightarrow$ find a root-free path with high probability $\longrightarrow$ quasi-poly algorithm that successes with high probability

How to bound average number of roots? $\longrightarrow$ by upper-bounding $E_g \left| \frac{g(z)}{g(0)} \right|^2$ [EM18]

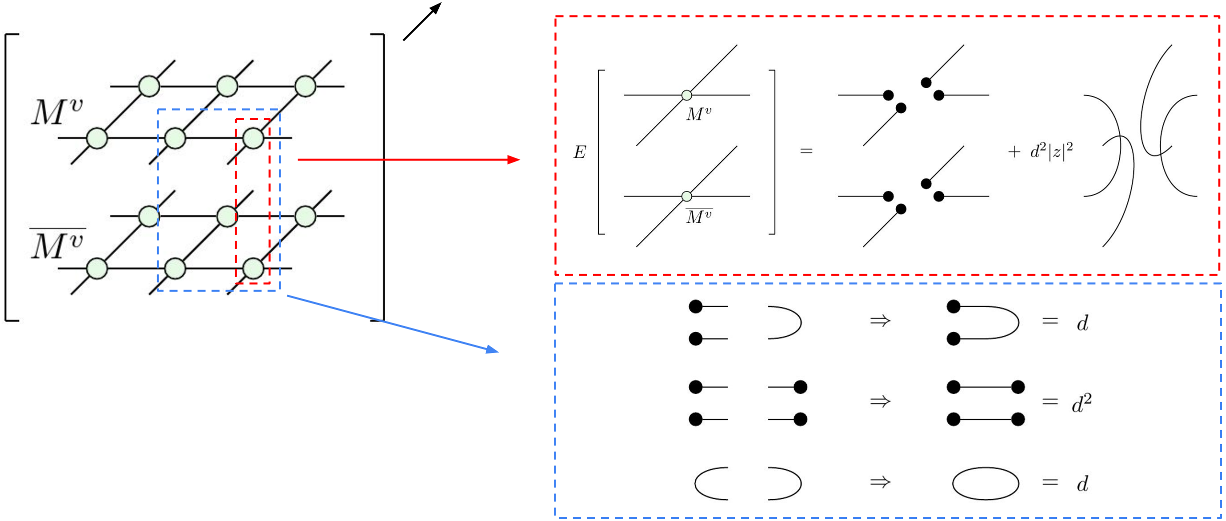
Barvinok's method on random tensor network

$f(z) = \ln(g(z))$

$f(1) \approx \hat{f}(1) = \sum_{k=0}^{M} \frac{f^{(k)}(0)}{k!}$ → few roots on average → find a root-free path with high probability → quasi-poly algorithm that successes with high probability

How to bound average number of roots? → by upper-bounding $E_g \left| \frac{g(z)}{g(0)} \right|^2$ [EM18]

$E_g |g(z)|^2 = E_g$



Barvinok's method on random tensor network

$f(z) = \ln(g(z))$

$f(1) \approx \hat{f}(1) = \sum_{k=0}^{M} \frac{f^{(k)}(0)}{k!}$ → few roots on average → find a root-free path with high probability → quasi-poly algorithm that successes with high probability

How to bound average number of roots? → by upper-bounding $E_g \left|\frac{g(z)}{g(0)}\right|^2$ [EM18]

$E_g|g(z)|^2 =$ partition function of 2D classical ising model with local magnetic field!

$z = 1 \longleftrightarrow$ zero magnetic field

Barvinok's method on random tensor network

$f(z) = \ln(g(z))$

$f(1) \approx \hat{f}(1) = \sum_{k=0}^{M} \frac{f^{(k)}(0)}{k!}$ $\longrightarrow$ few roots on average $\longrightarrow$ find a root-free path with high probability $\longrightarrow$ quasi-poly algorithm that successes with high probability

How to bound average number of roots? $\longrightarrow$ by upper-bounding $E_g \left| \frac{g(z)}{g(0)} \right|^2$ [EM18]

$E_g |g(z)|^2 =$ partition function of 2D classical ising model with local magnetic field! $\longrightarrow$ Onsager's solution [Ons44] for zero-field 2D ising model (finite-size variant [Kau49])

$z = 1 \longleftrightarrow$ zero magnetic field

Barvinok's method on random tensor network

$f(z) = \ln(g(z))$

$f(1) \approx \hat{f}(1) = \sum_{k=0}^{M} \frac{f^{(k)}(0)}{k!}$ → few roots on average $O(1)$ → find a root-free path with high probability → quasi-poly algorithm that successes with high probability

How to bound average number of roots? → by upper-bounding $E_g \left| \frac{g(z)}{g(0)} \right|^2$ [EM18]

$O(1)$ for z ≲ 1

$E_g |g(z)|^2 =$ partition function of 2D classical ising model with local magnetic field! → Onsager's solution [Ons44] for zero-field 2D ising model (finite-size variant [Kau49])

$z = 1 \longleftrightarrow$ zero magnetic field

Fully positive TN

What can we say about approximating a fully positive TN?

Brief summary of two results:

1. Approximating a positive TN up to very large multiplicative error is StoqMA-hard.

2. Approximating a positive TN with additive error bounded by the product of 1-norm of each tensor is BPP-complete.

   Approximating an arbitrary TN with additive error bounded by the product of 2-norm of each tensor is BQP-complete. [AL10]

Outlook

We showed that there's a quasi-polynomial time classical algorithm to approximately contract a slightly positive tensor network.

- How to prove tractability with constant bond-dimension?
- Can one improve the method?
  - Interpolate from other rank-one tensors? (e.g. BP-fixed point)
  - Use better interpolations?
- Can the method be used in practice? (with some modifications & heuristics)