Approximate contraction & hardness for PEPS

TN subgroup

15/04/2021

Johnnie Gray

Overview:

2D PEPS norm contraction methods:

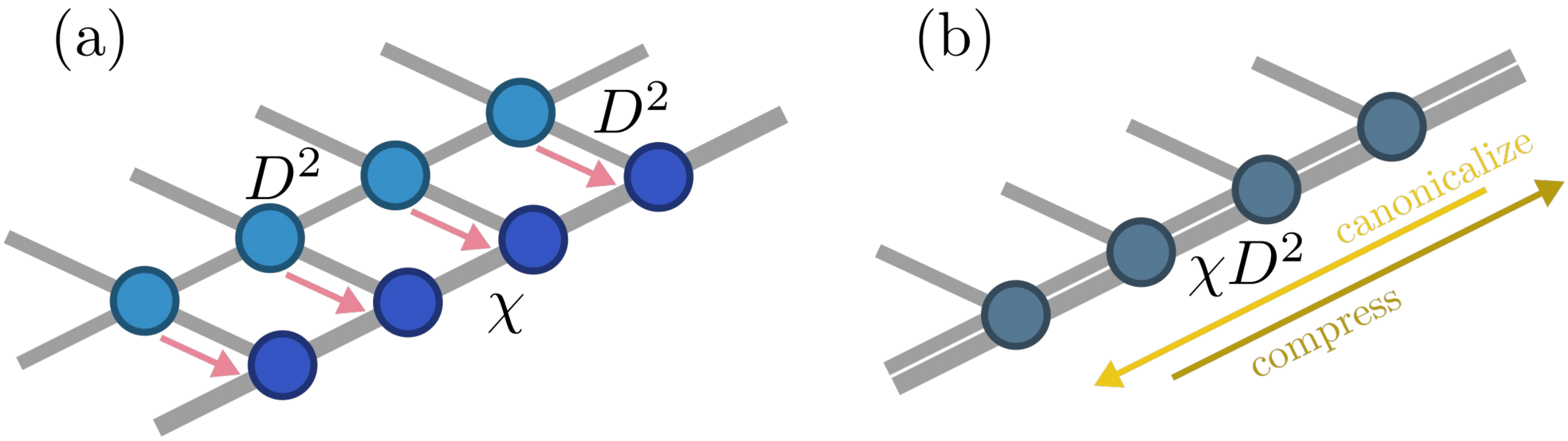
- MPS one and two layer contraction
- Local (automatic) compressing:
  - ‘early’ vs ‘late’ compression
  - ‘boundary style’ gauging
- Full environment compression
  - manual 2D scheme

Results for random (normally distributed) 2D PEPS

- Error scaling of each method with
  - Boundary bond dimension
  - random distribution mean,
  - Contraction width,
- Even harder TNs to contract?

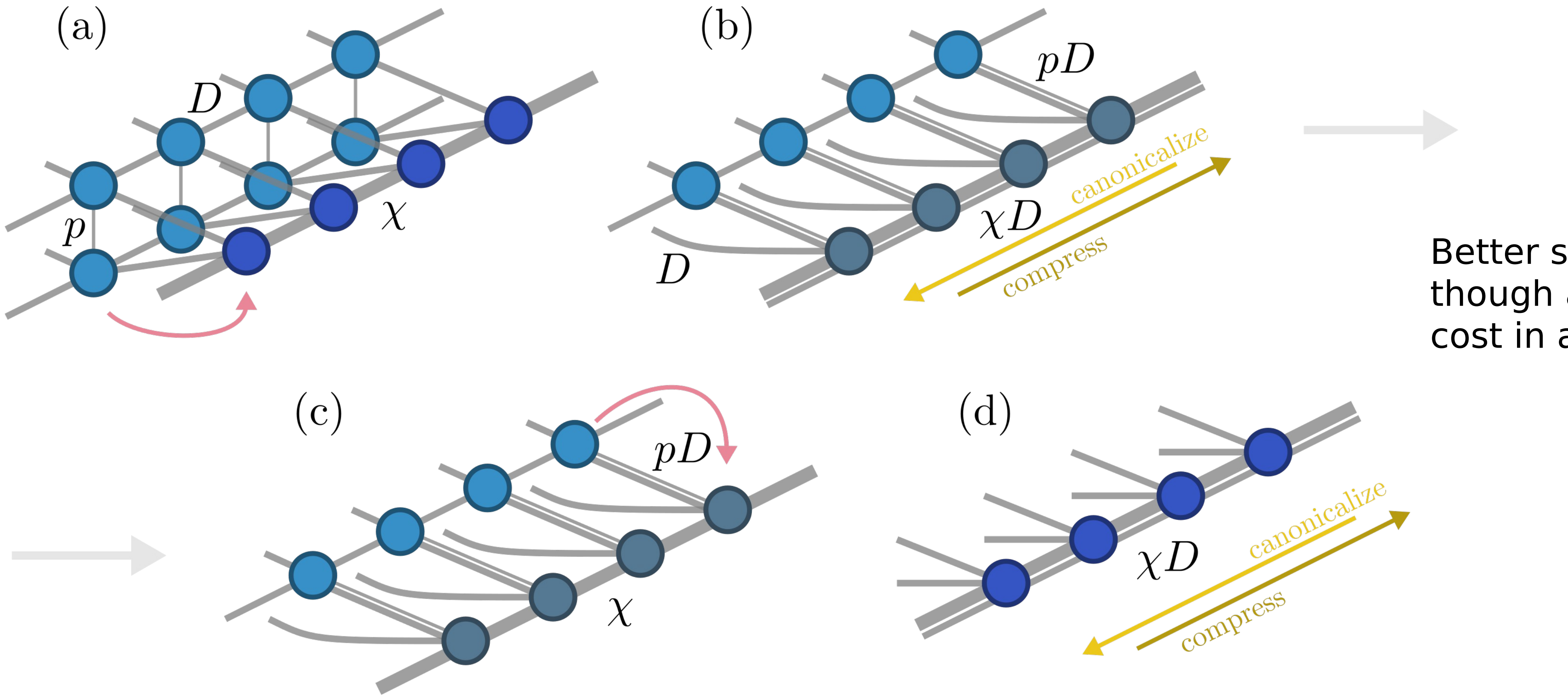
Contraction: boundary MPS one layer

1. Flatten TN **2.** Contract entire next row in right **3.** Canonicalize left **4.** Compress



Contraction: boundary MPS two layers

1. Contract in lower layer **2.** canonicalize + compress **3.** contract in upper layer **4.** canonicalize + compress



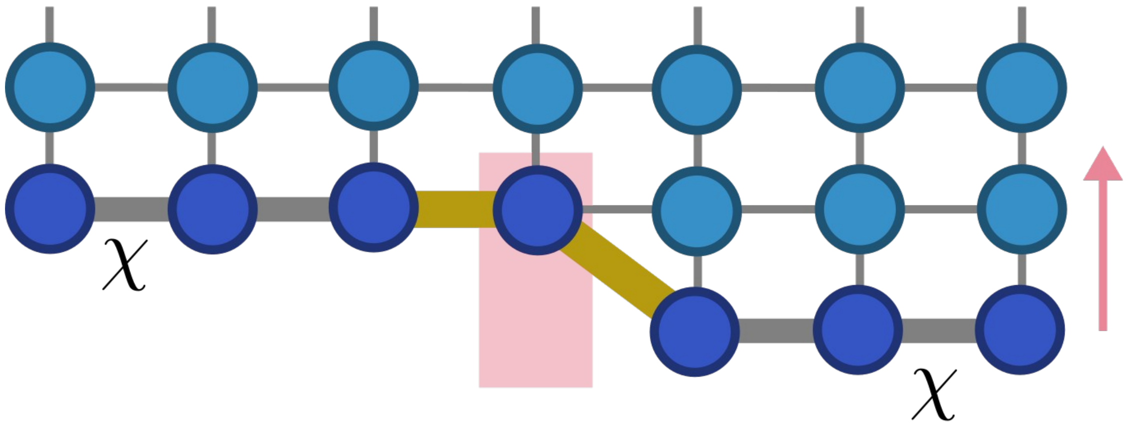
Better scaling though at *some* cost in accuracy.

Local (automatic) compression:

Early Compression - as soon as we have made a contraction, compress resulting tensor with neighbors – best memory-wise:

'contract & compress'

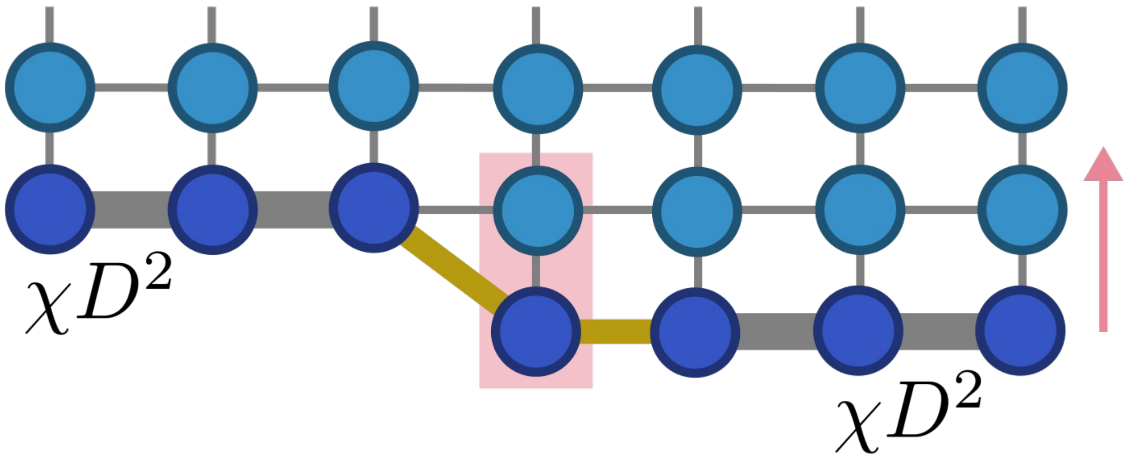
$$\chi \rightarrow \chi D^2 \rightarrow \chi$$



Late Compression – don't compress a tensor with its neighbors until we are about to contract it – try to replicate MPS method. Potentially worse memory though in most cases not:

'compress & contract'

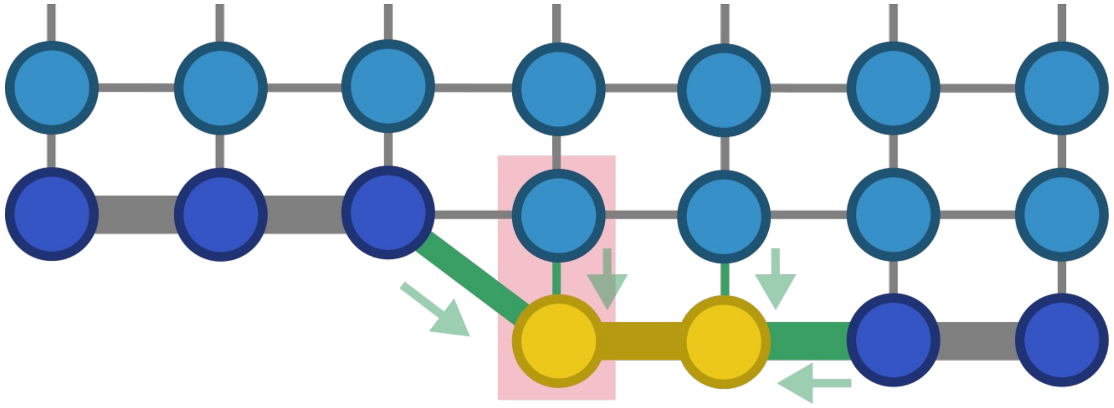
$$\chi D^2 \rightarrow \chi \rightarrow \chi D^2$$



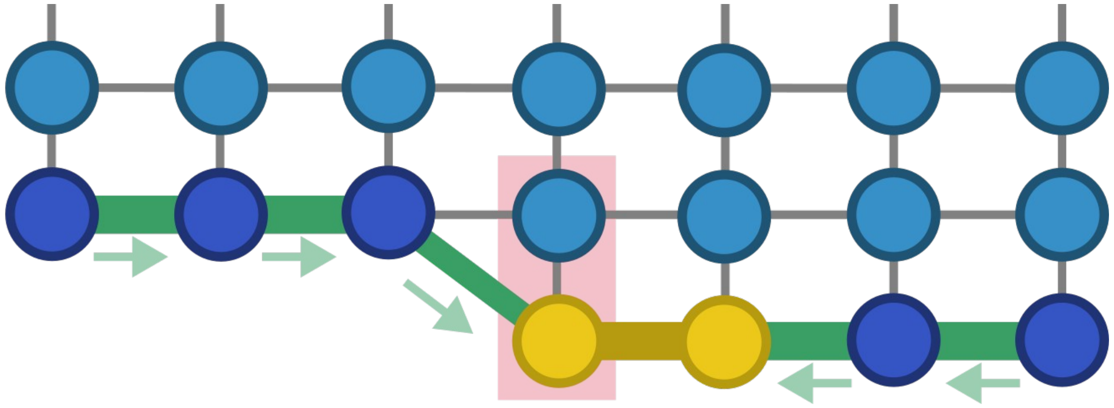
Advantage: get to use more effective environment information for gauging.

Local (automatic) gauging:

Simple short range style – absorb QR factors from small local tree in all directions, here :

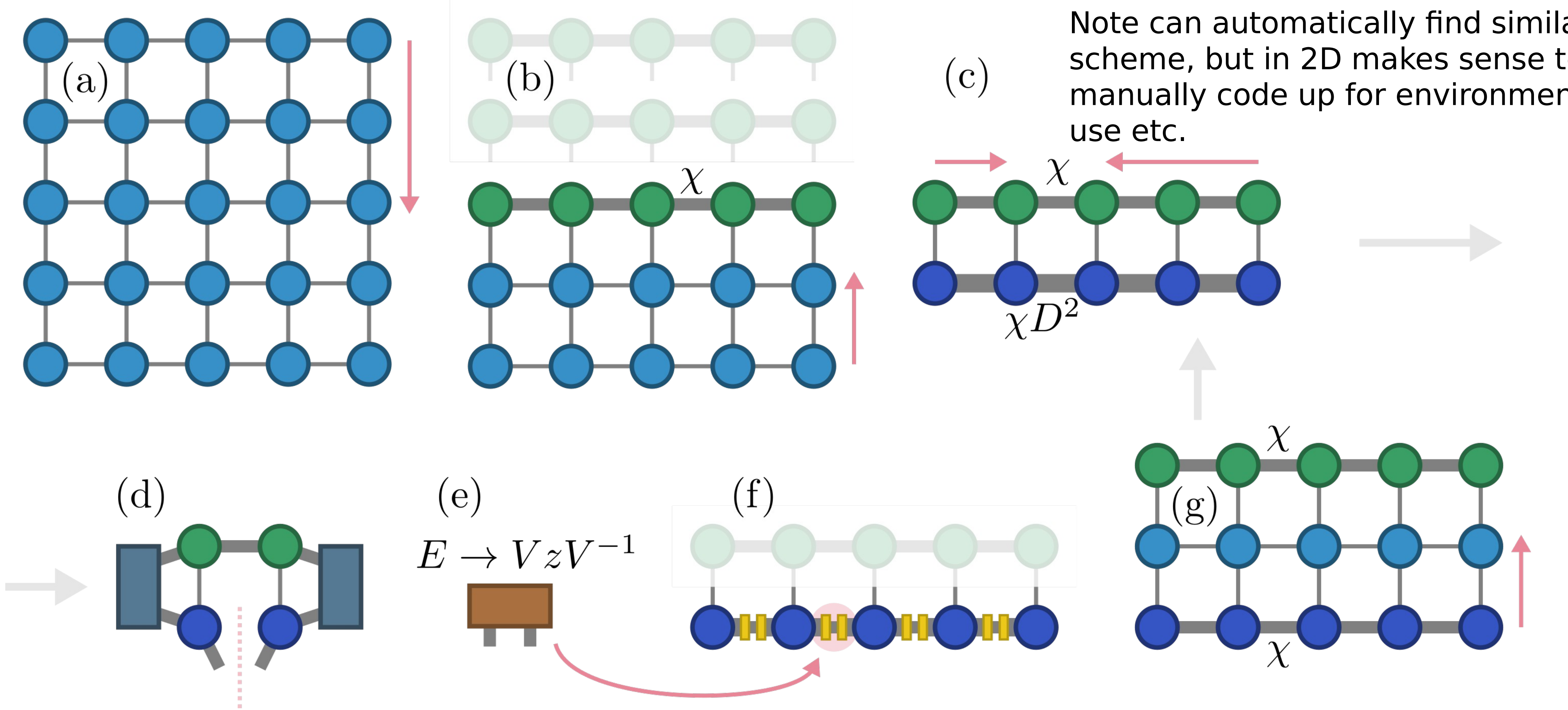


Boundary style – absorb QR factors from an unlimited distance away but only along intermediate tensors:



Generally making bigger always is better (though with diminishing returns). Restriction to boundary is mostly just for efficiency.

Full bond environment for 2D



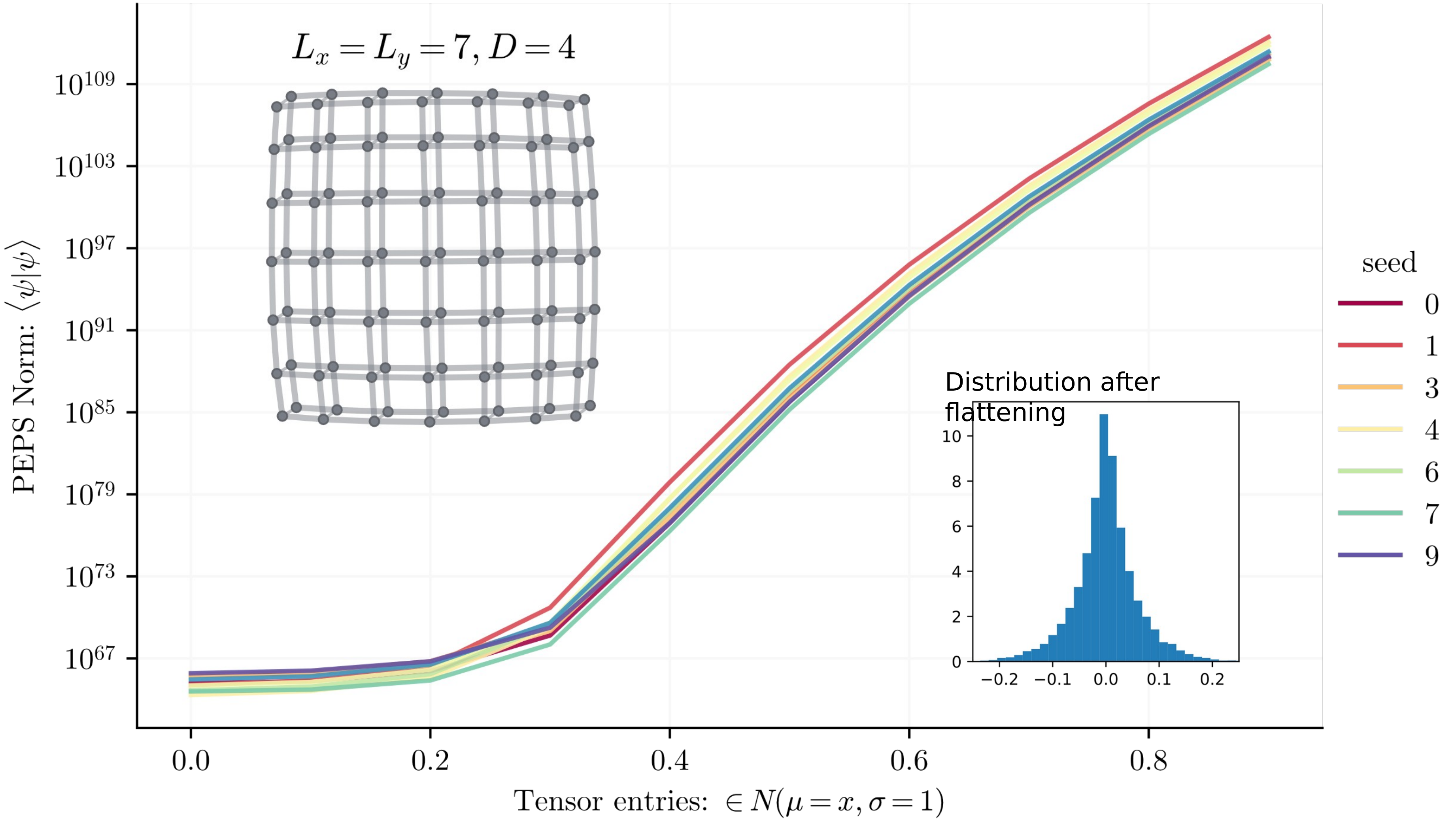
Note can automatically find similar scheme, but in 2D makes sense to manually code up for environment re-use etc.

Contracting norm of random PEPS

with is edge of what we can contract exactly – still difficult.

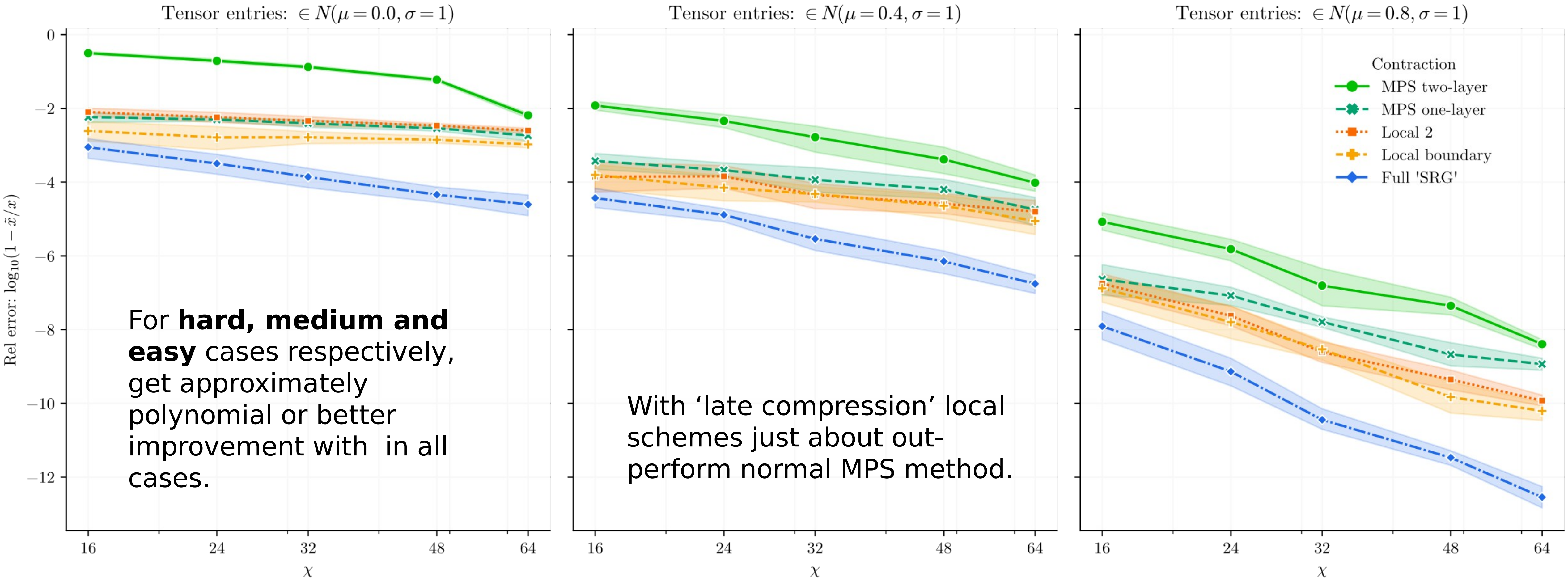
Tune hardness by drawing tensor entries from Normal distribution with mean .

Get reasonable relative variance in values – some change around



Error scaling with bond dimension

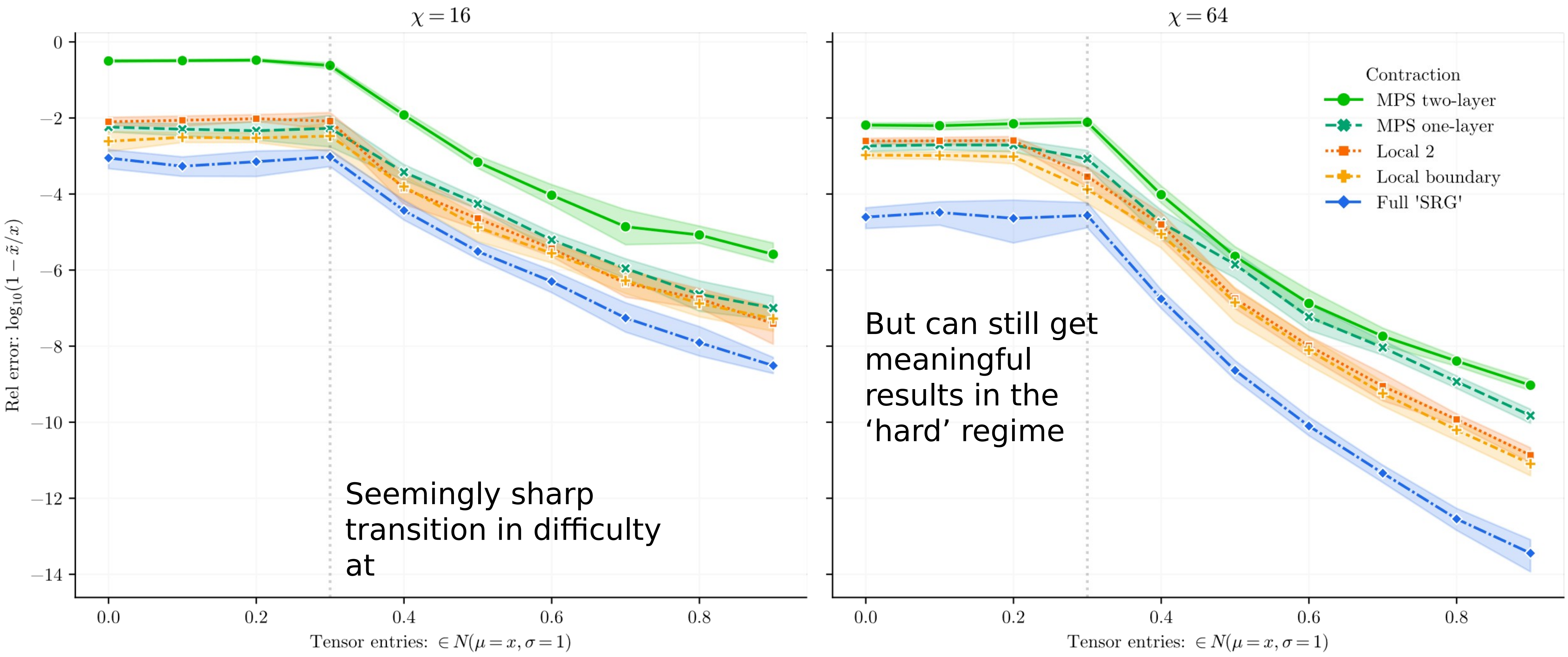
For given bond dimension find ~ *MPS two-layer < MPS one-layer <= Local < Full bond*



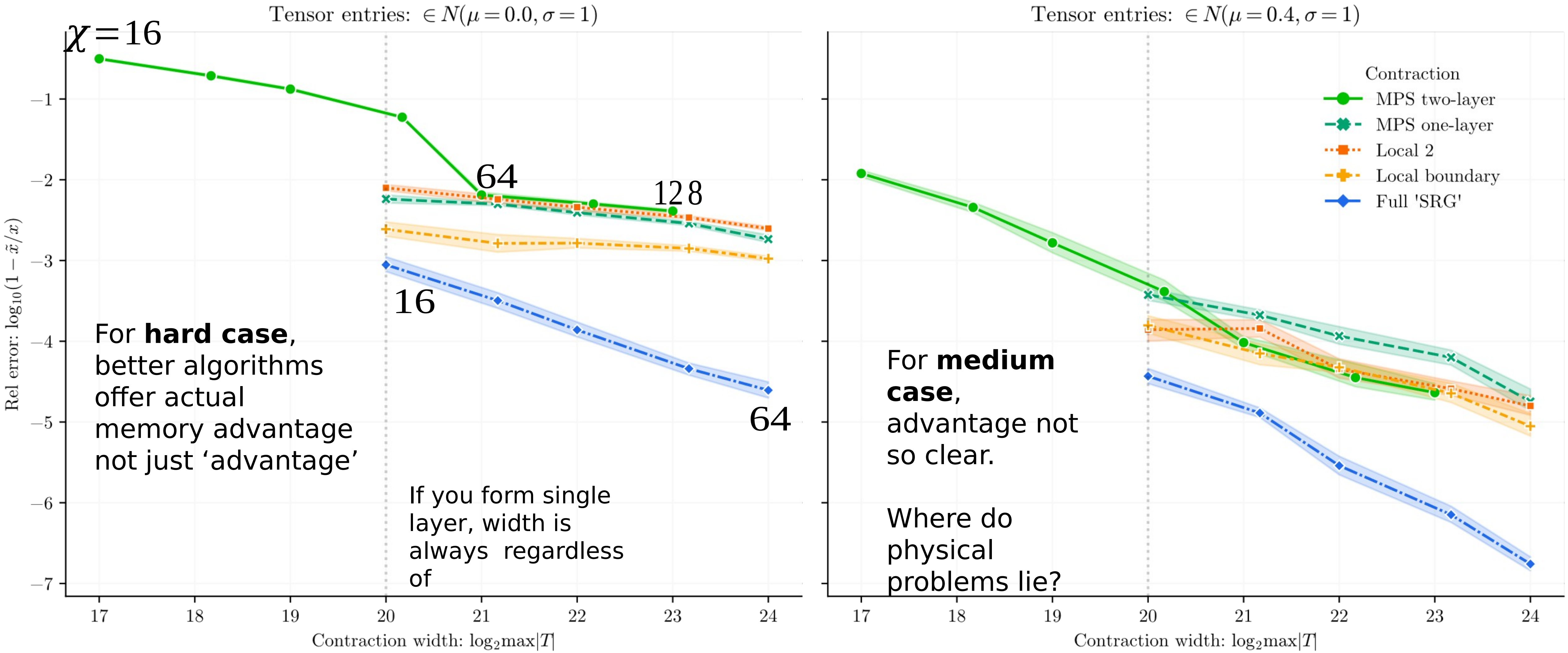
For **hard, medium and easy** cases respectively, get approximately polynomial or better improvement with in all cases.

With ‘late compression’ local schemes just about out-perform normal MPS method.

Error scaling with distribution



Error scaling with ‘contraction width’



For **hard case**, better algorithms offer actual memory advantage not just ‘advantage’

If you form single layer, width is always regardless of

For **medium case**, advantage not so clear.

Where do physical problems lie?

Other cases are harder!

For example, contracting a random single layer (i.e. scalar) tensor network with randomly distributed entries of size with (equivalent to norm of PEPS with but without positive definite structure).

Below certain threshold (different distribution here - uniform), all methods at all essentially garbage, relative error