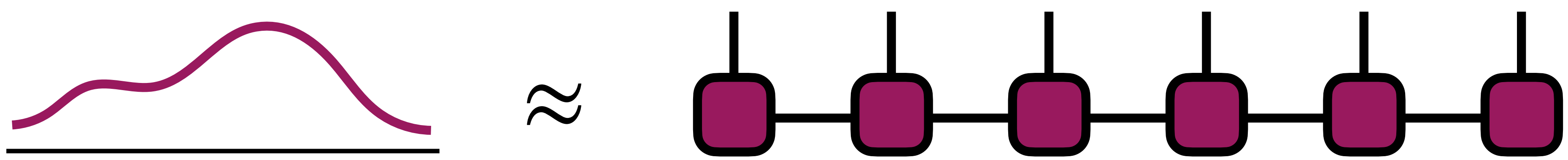
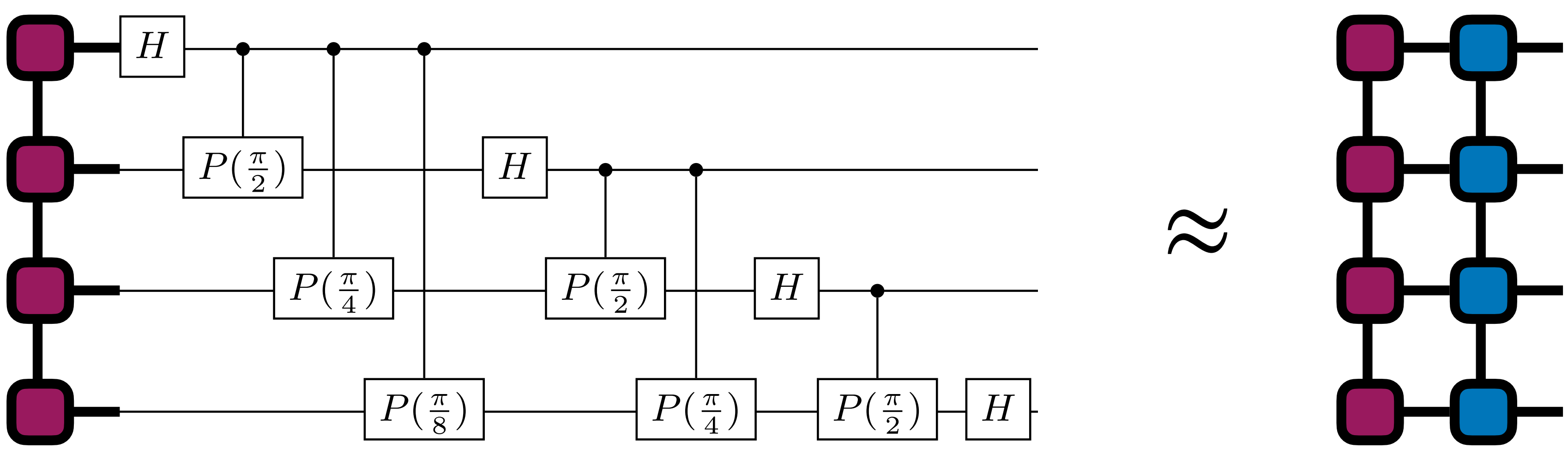
New Directions for Tensor Networks: Function Representations & Machine Learning





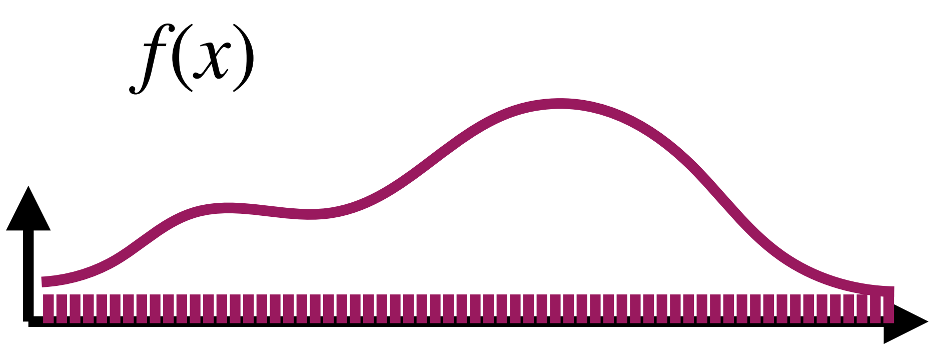
Motivation

Traditional view of tensor networks =
many-body quantum wavefunctions

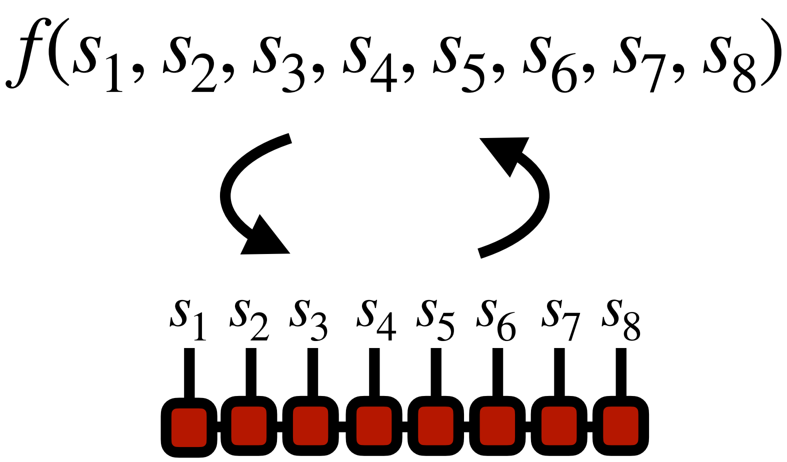
But really just linear algebra...in high dimensions

Two surprising directions for tensor networks

Continuum functions



Machine Learning



This Talk

- Encoding Continuum Functions as Tensor Networks and Algorithms for Tensorized Functions

- Machine Learning with Tensor Networks

    - Active Learning of Data (Tensor Cross)

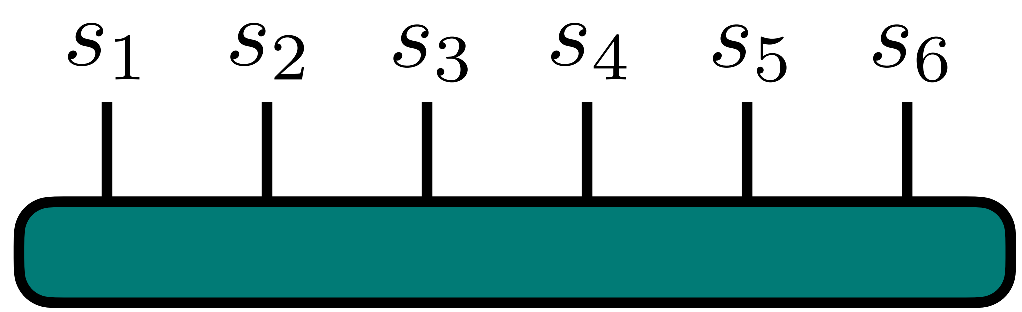
    - Tensor Recursive Sketching Algorithm

Representing Functions as Tensor Networks

Which functions have
low "quantum entanglement"?

Any function of discrete variables can be represented as a tensor

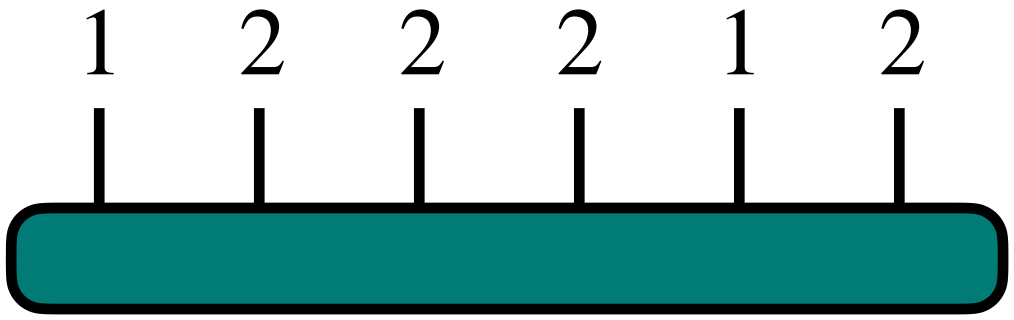
$$f(s_1, s_2, s_3, s_4, s_5, s_6) =$$



All values of f inside

Any function of discrete variables can be represented as a tensor

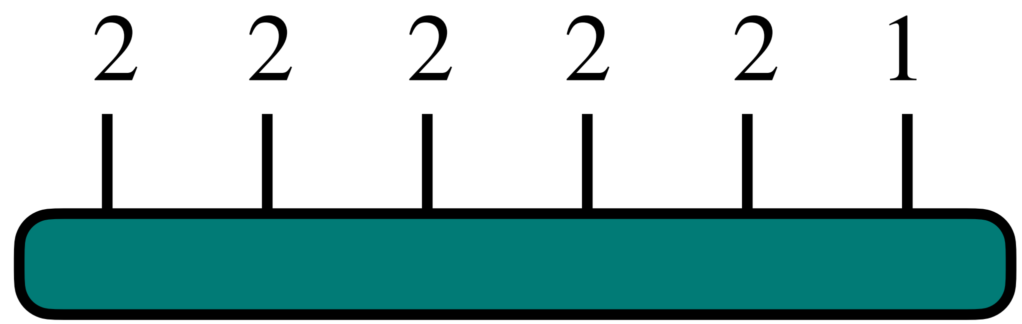
$$f(1,2,2,2,1,2) =$$



$$= 1.2$$

Any function of discrete variables can be represented as a tensor

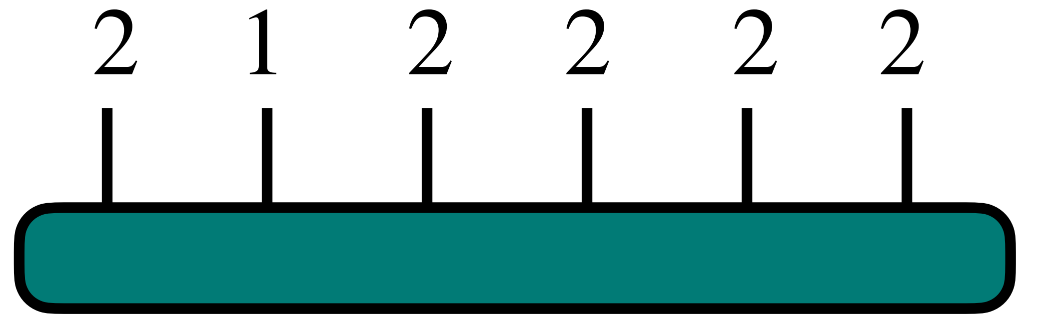
$$f(2,2,2,2,2,1) =$$



$$= 0.3$$

Any function of discrete variables can be represented as a tensor

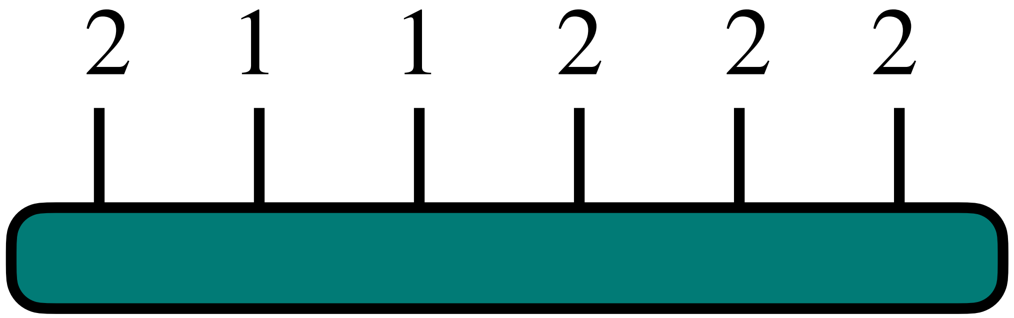
$$f(2,1,2,2,2,2) =$$



$$= -0.2$$

Any function of discrete variables can be represented as a tensor

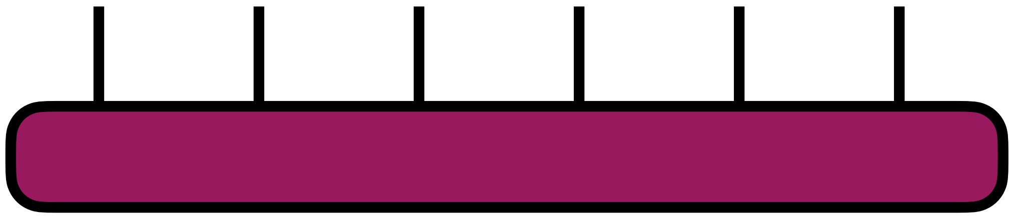
$f(2,1,1,2,2,2) =$



$= 2.7$

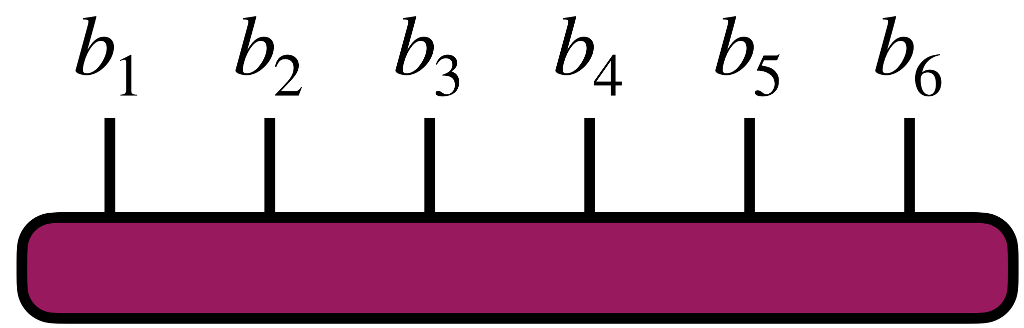
Can functions of **continuous variables** be "tensorized" also?

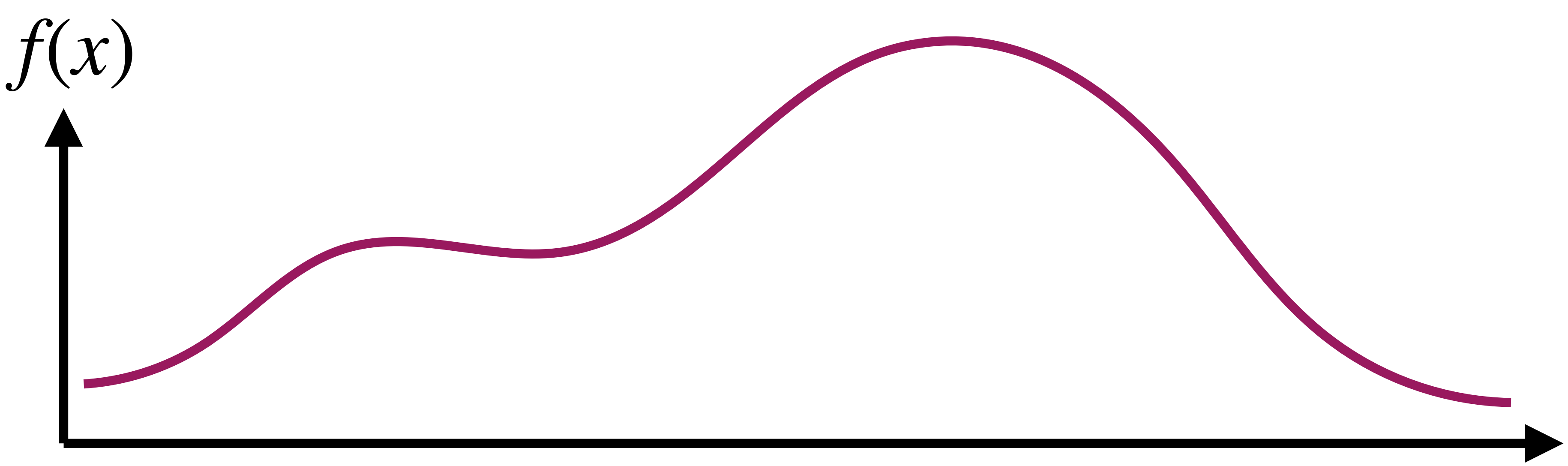
$$f(x, y, z) = x^2 + 3y + cos(z)$$

$\overset{?}{=}$ 

Yes. Using binary fraction approximation of $x \in [0,1)$

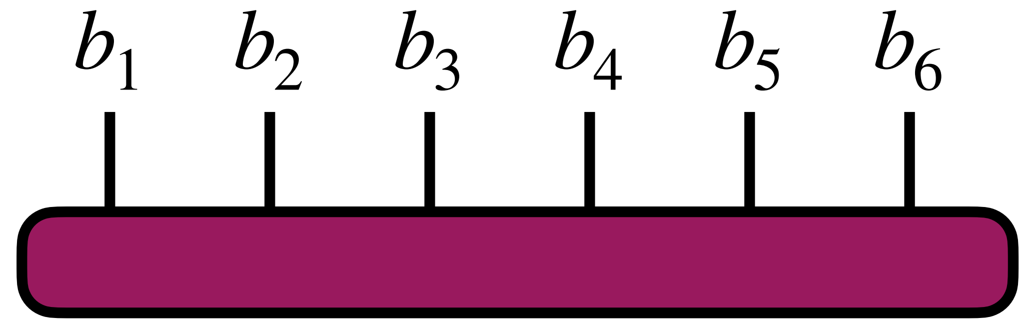
$$f(x) \approx f(0.b_1b_2b_3b_4b_5b_6) =$$

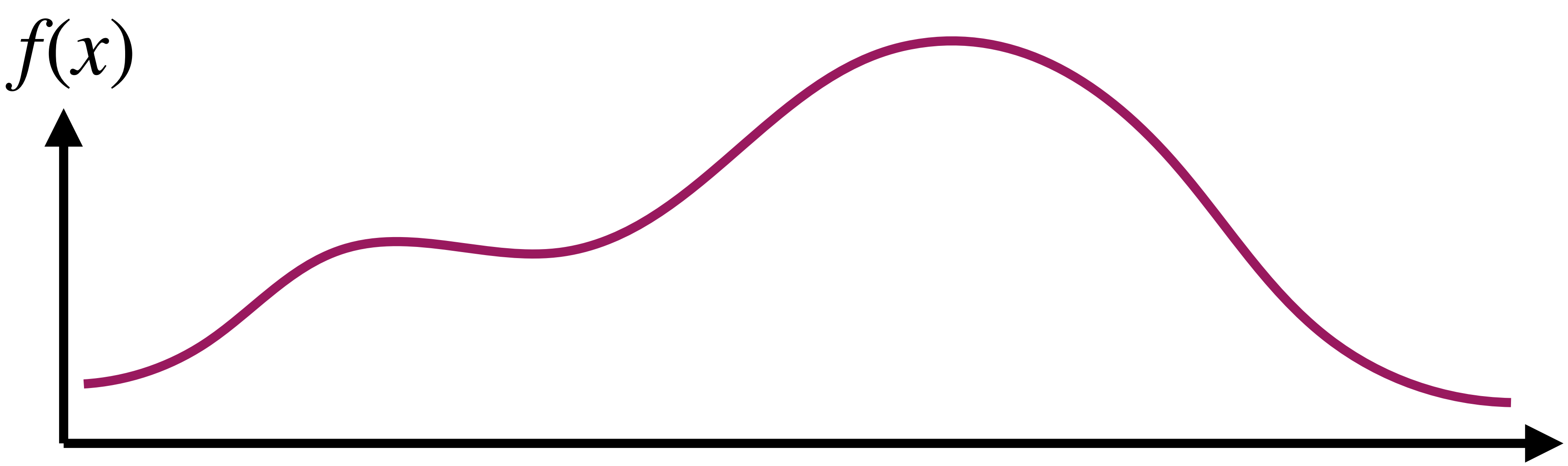




Let us understand in more detail...

$$f(x) \approx f(0.b_1 b_2 b_3 b_4 b_5 b_6) =$$

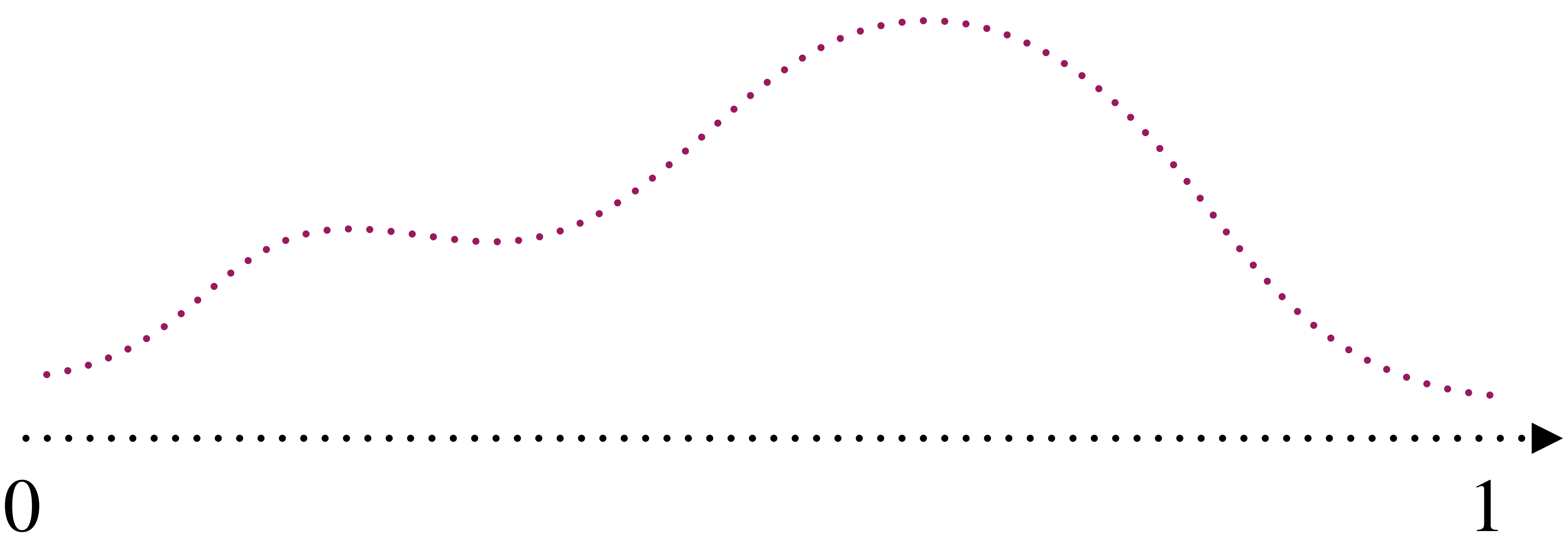




Quantum Functions

Discretize [0,1) as grid of "binary fractions"

$f(x) =$



$$x \approx \frac{b_1}{2} + \frac{b_2}{2^2} + \frac{b_3}{2^3} + \frac{b_4}{2^4} + \ldots + + \frac{b_n}{2^n} \qquad b_j = 0,1$$

$$\approx 0.b_1 b_2 b_3 b_4 \cdots b_n$$

Quantum Functions

On this grid, $f(x)$ takes values

$$f\left(\frac{b_1}{2}+\frac{b_2}{2^2}+\frac{b_3}{2^3}+\frac{b_4}{2^4}+\ldots++\frac{b_n}{2^n}\right)$$

Quantum Functions

On this grid, $f(x)$ takes values

$$f\left(\frac{b_1}{2}+\frac{b_2}{2^2}+\frac{b_3}{2^3}+\frac{b_4}{2^4}+\ldots++\frac{b_n}{2^n}\right)$$

defining a tensor

$$= F^{b_1 b_2 b_3 b_4 \cdots b_n} \qquad\qquad b_j = 0{,}1$$

all values of $f(0.b_1 b_2 b_3 \cdots b_n)$ inside

Quantum Functions

$$f\Big(\frac{b_1}{2}+\frac{b_2}{2^2}+\frac{b_3}{2^3}+\frac{b_4}{2^4}+\ldots++\frac{b_n}{2^n}\Big) \;= F^{b_1 b_2 b_3 b_4 \cdots b_n}$$

Nothing more than "amplitude encoding" idea

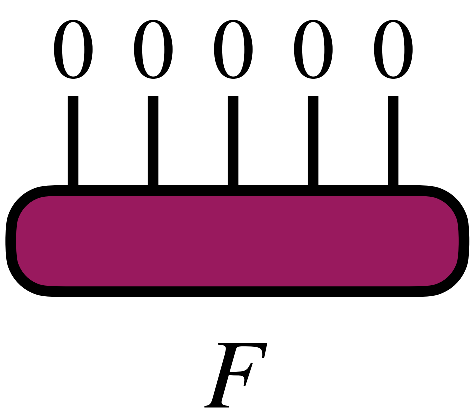
known in quantum computing literature...

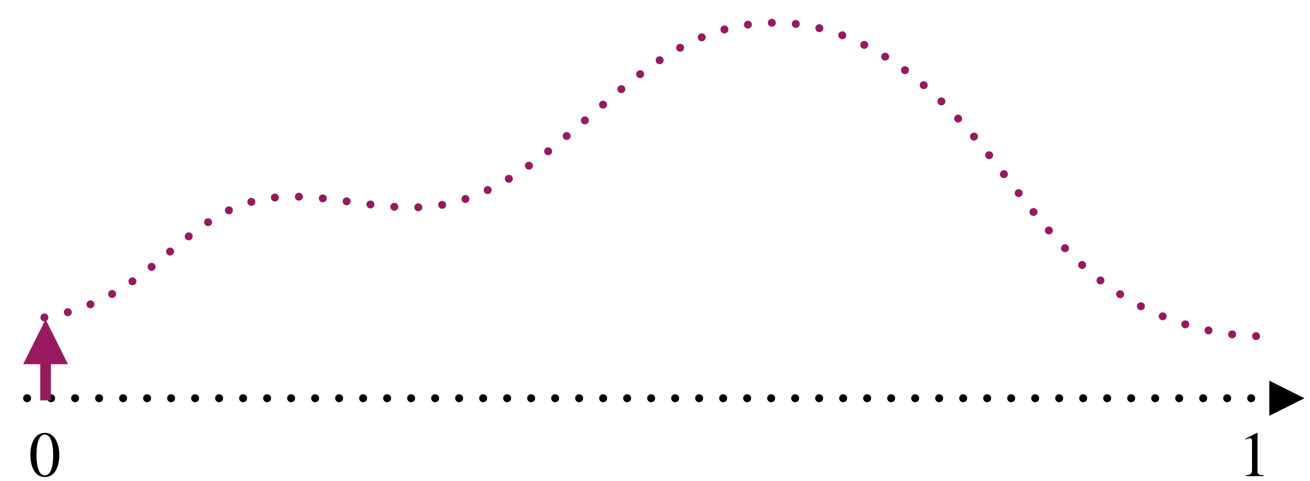
$$|F\rangle = \sum_{\mathbf{b}} F^{b_1 b_2 b_3 \cdots b_n} |b_1 b_2 b_3 \cdots b_n\rangle$$

but with low entanglement / low-rank perspective

Quantum Functions

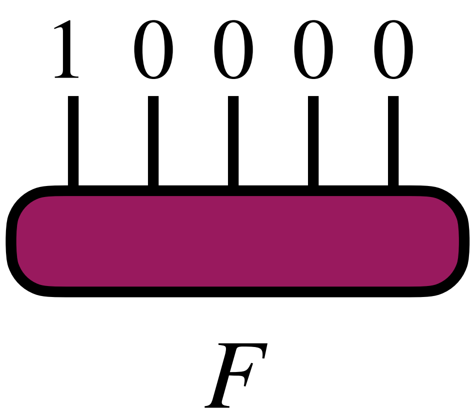
Examples of F entries

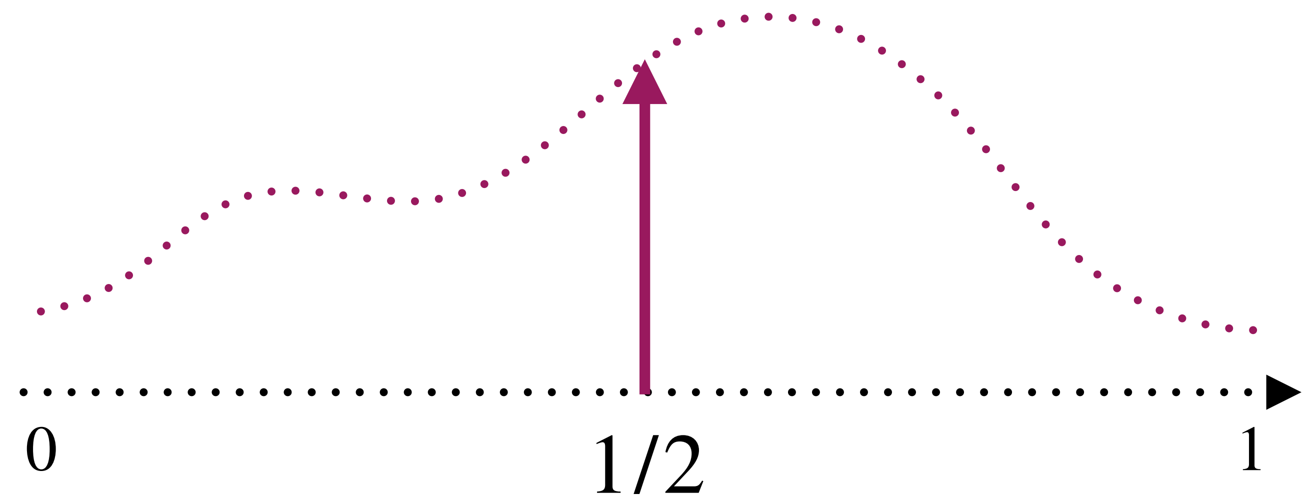
$f(0.00000) =$ 



Quantum Functions

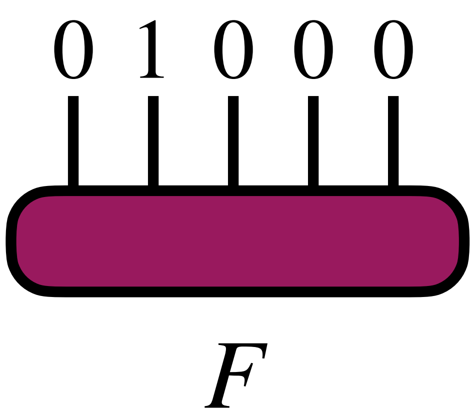
Examples of F entries

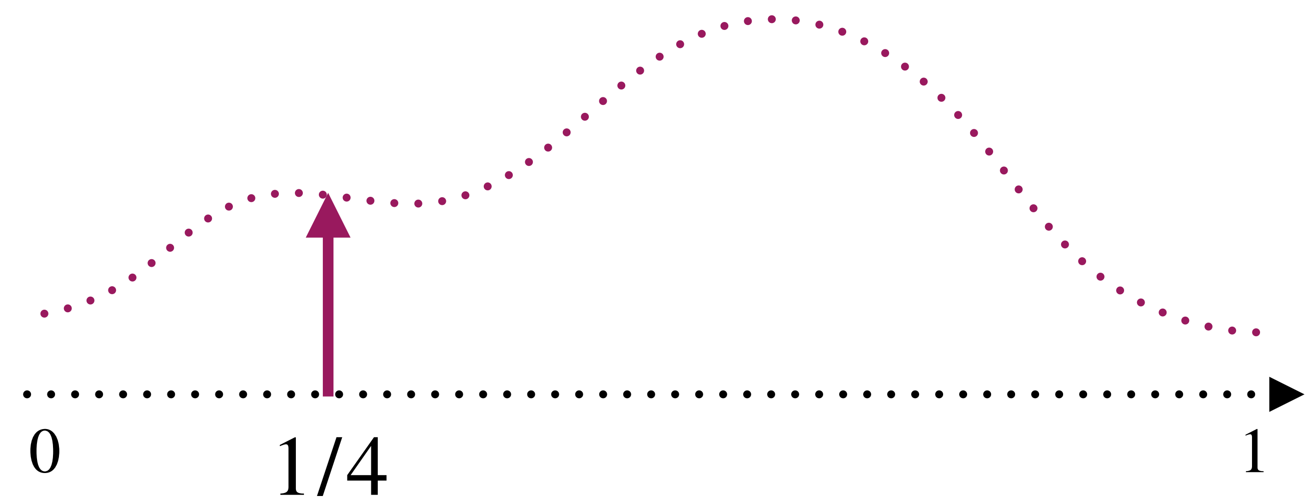
$f(0.10000) =$ 



Quantum Functions

Examples of F entries

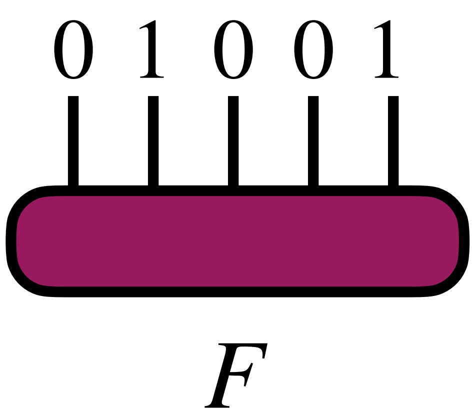
$f(0.01000) =$ 

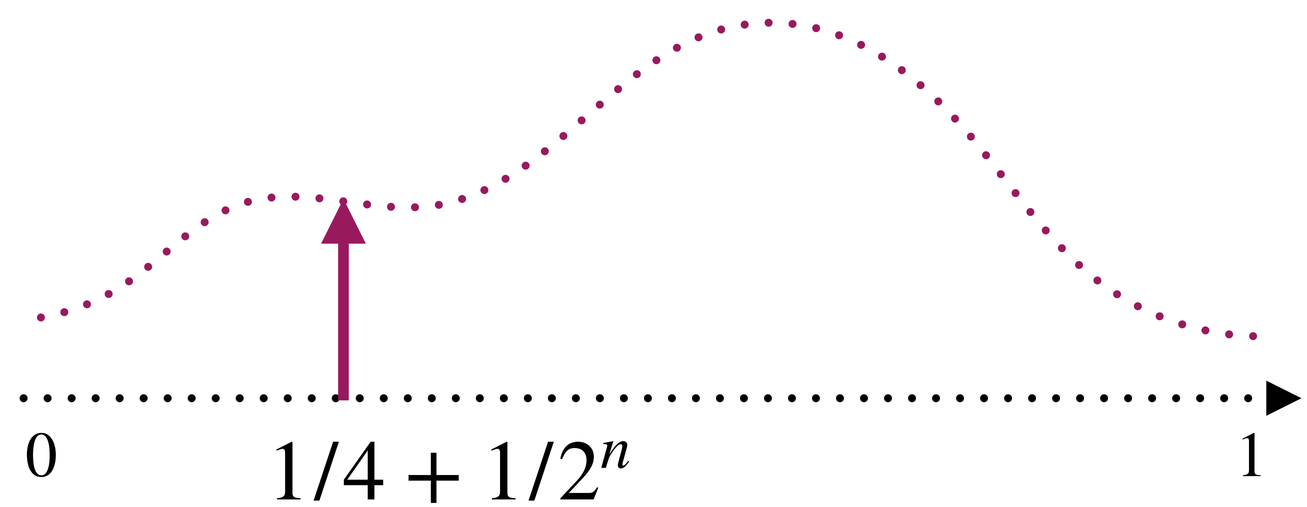


Quantum Functions

Examples of F entries

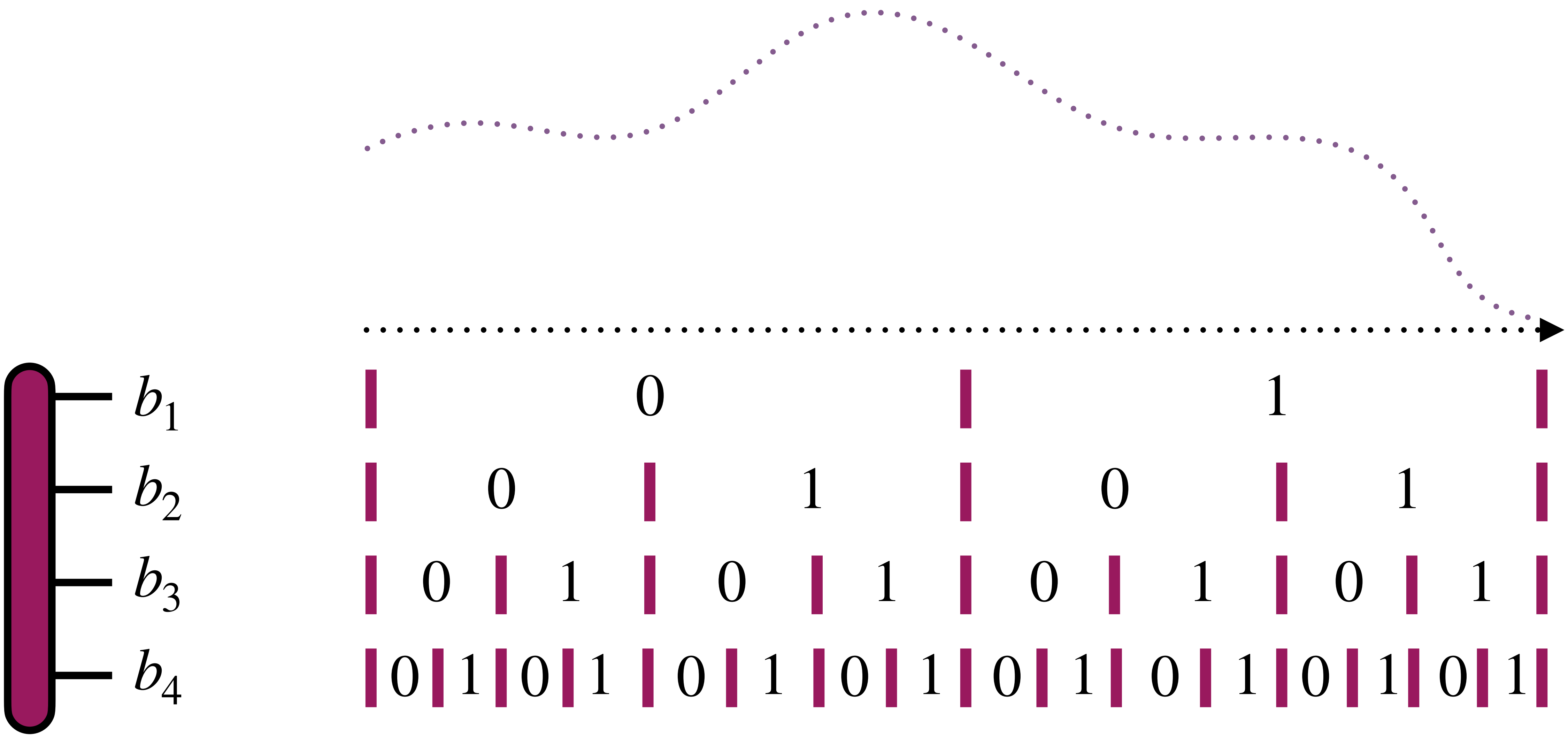
$f(0.01001) =$





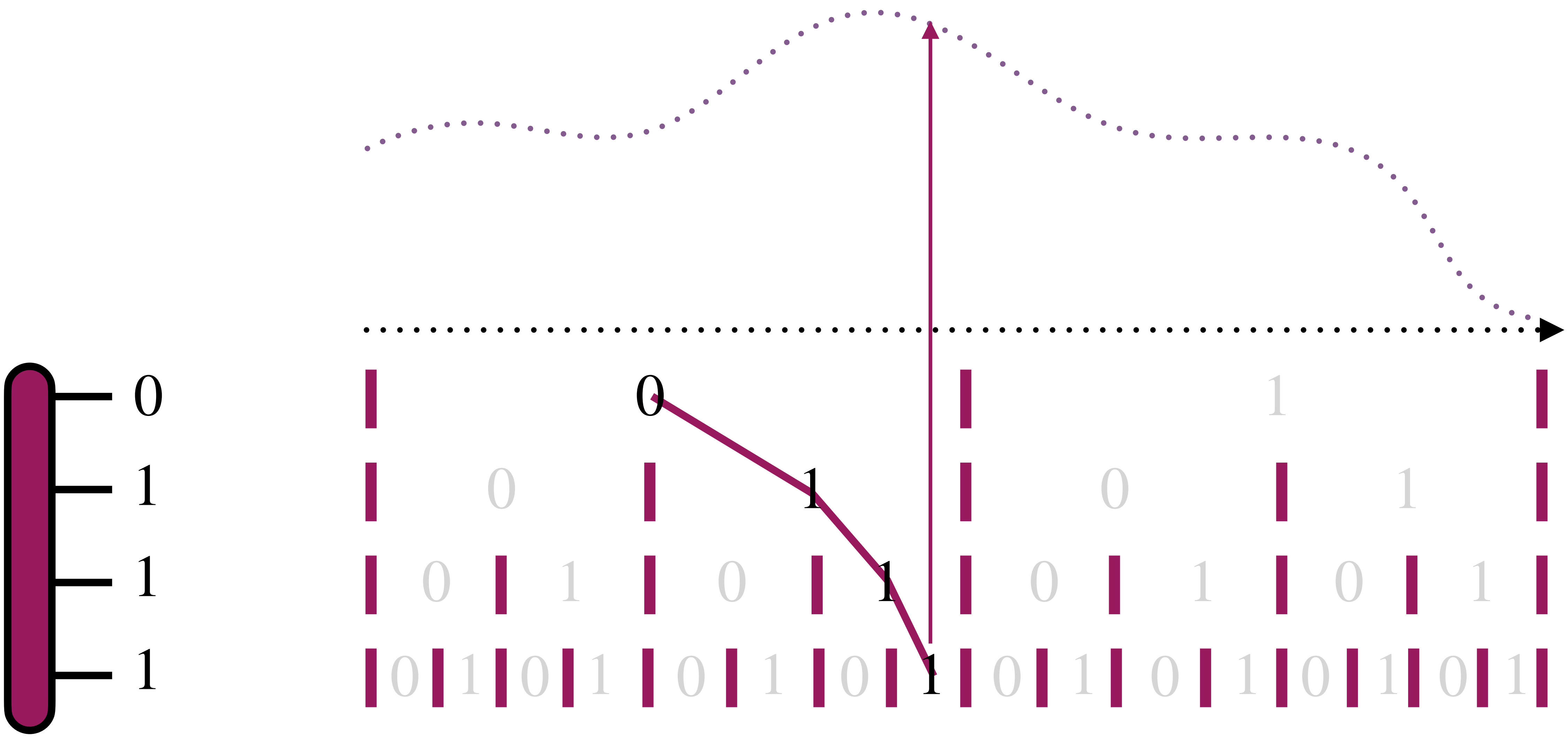
Quantum Functions

It is a **multiscale** representation



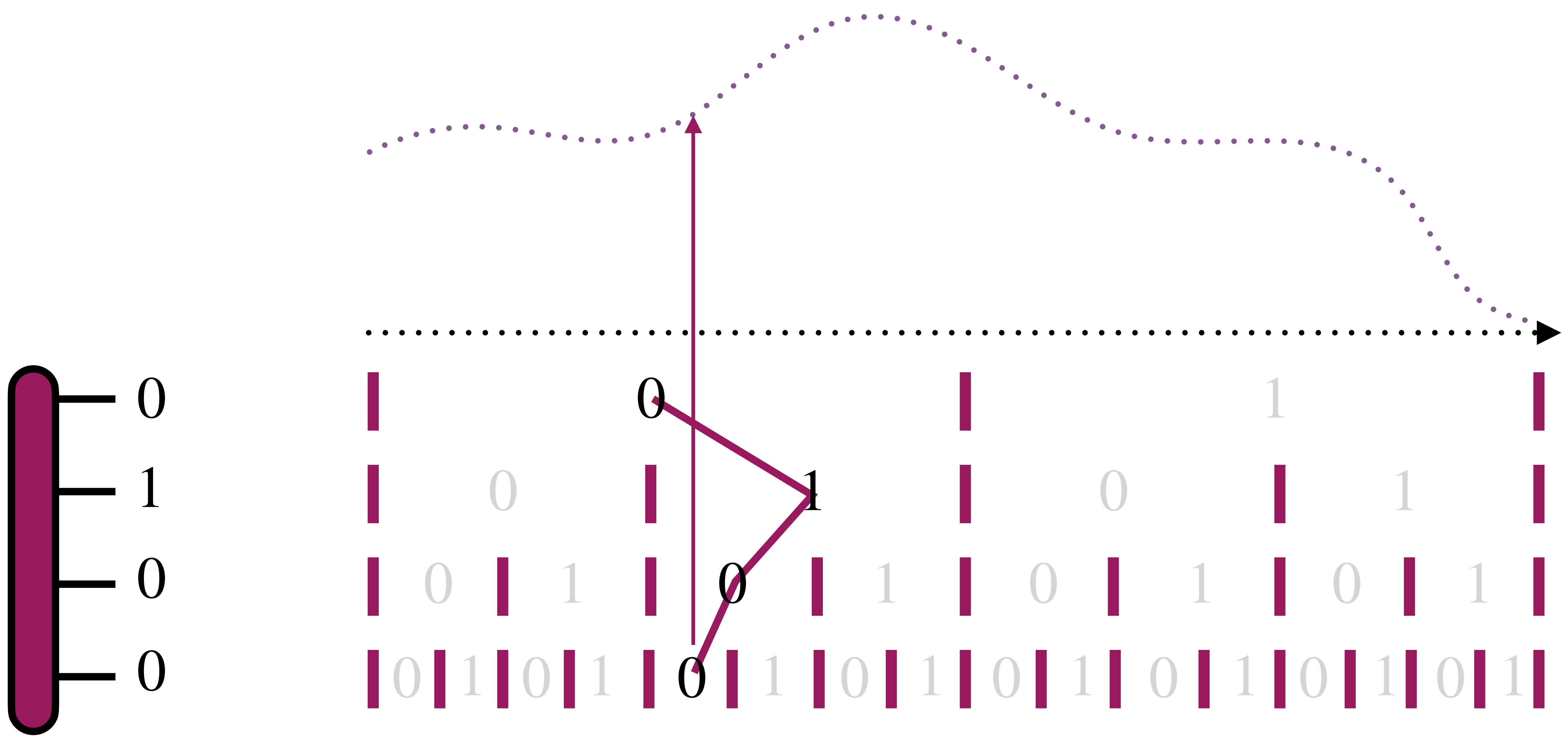
Quantum Functions

It is a **multiscale** representation



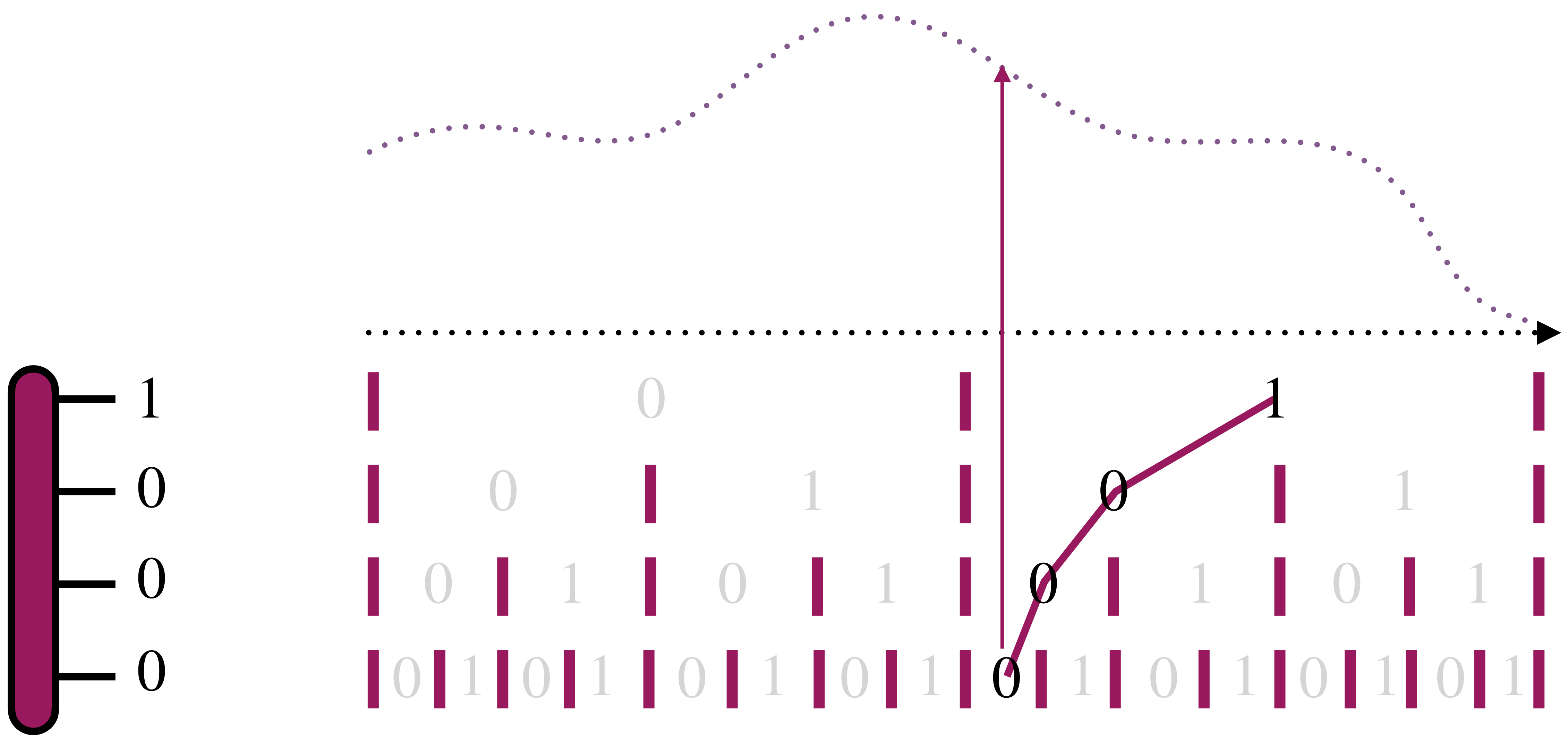
Quantum Functions

It is a **multiscale** representation



Quantum Functions

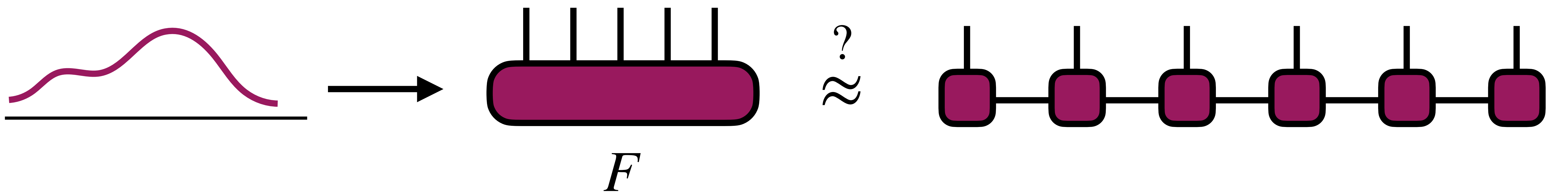
It is a **multiscale** representation



Quantum Functions

Key question: for some $f(x)$

is F **low-rank** as a tensor network? (i.e. "**low entanglement**")

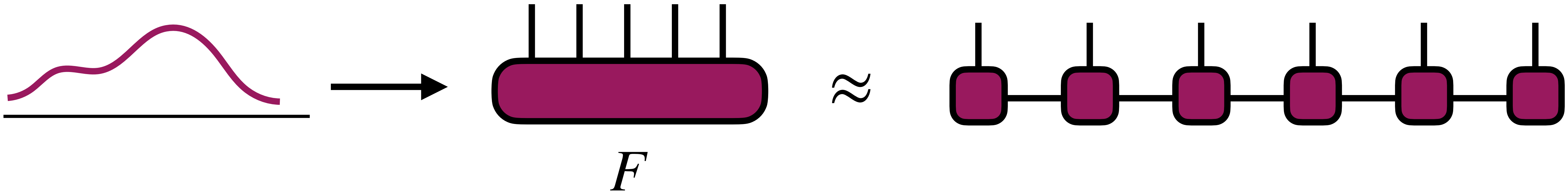


Otherwise exponential cost

Quantum Functions

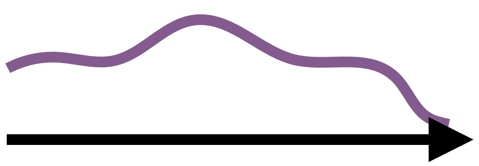
[1] Mazen Ali, Anthony Nouy, Constr Approx
[2] Mazen Ali, Anthony Nouy, arxiv:2101.11932
[3] Chen, EMS, White, PRX Quantum, arxiv:2210.08468
[4] Michael Lindsey, arxiv:2311.12554 (2023)
[5] Chen, Lindsey, arxiv:2404.03182 (2024)

Low rank for many cases

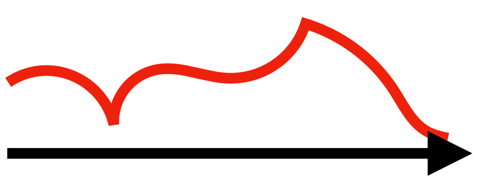


Tensor network low rank for:

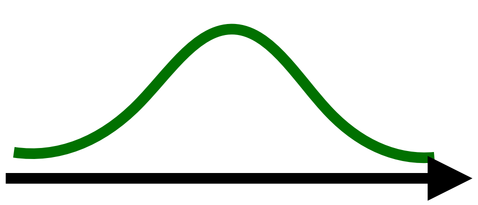
- all smooth enough functions [1,2,4]



- functions with finite number of cusps or discontinuities [1,2,4]



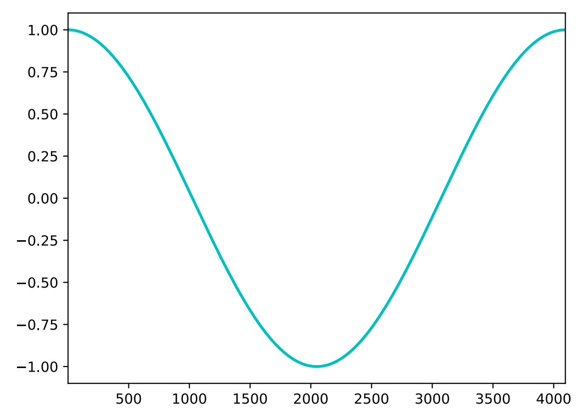
- any Fourier transform of these [3,5]

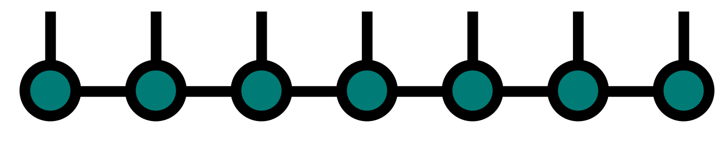


Quantum Functions

Examples:

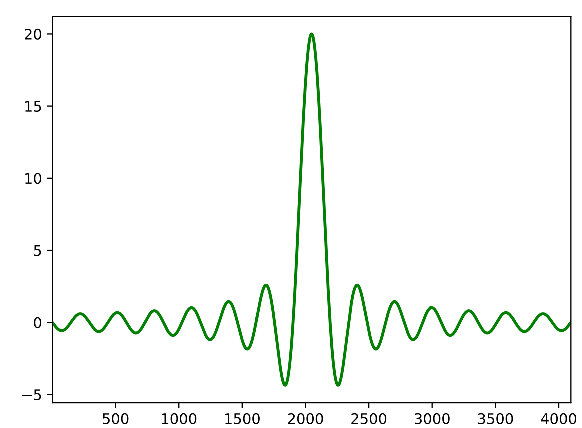
$$\cos\left[x - \frac{1}{2}\right]$$

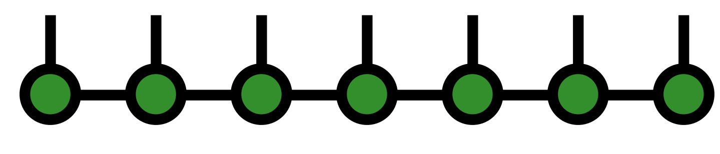




$\chi = 2$

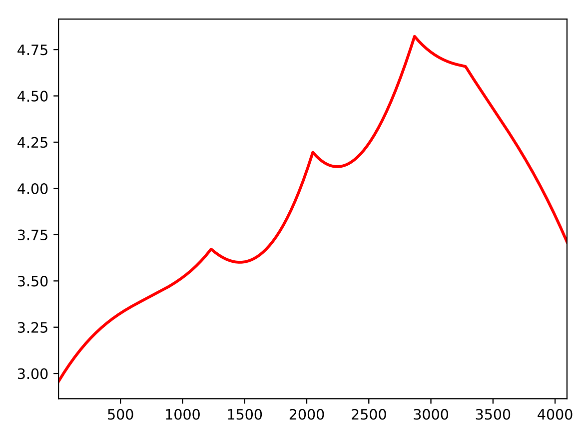
sum of 20
cosines

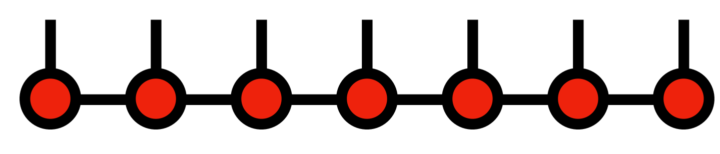




$\chi = 18$

cosine
+ cusps



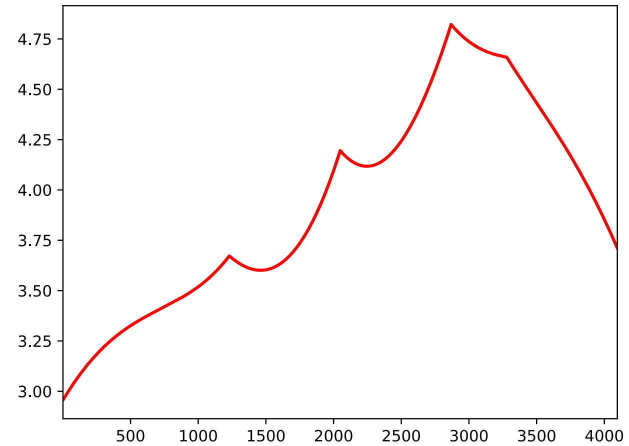


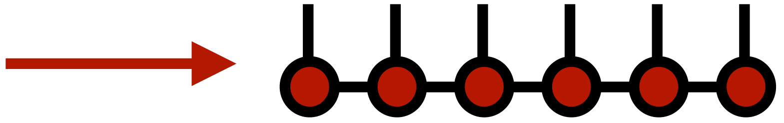
$\chi = 9$

Quantum Functions

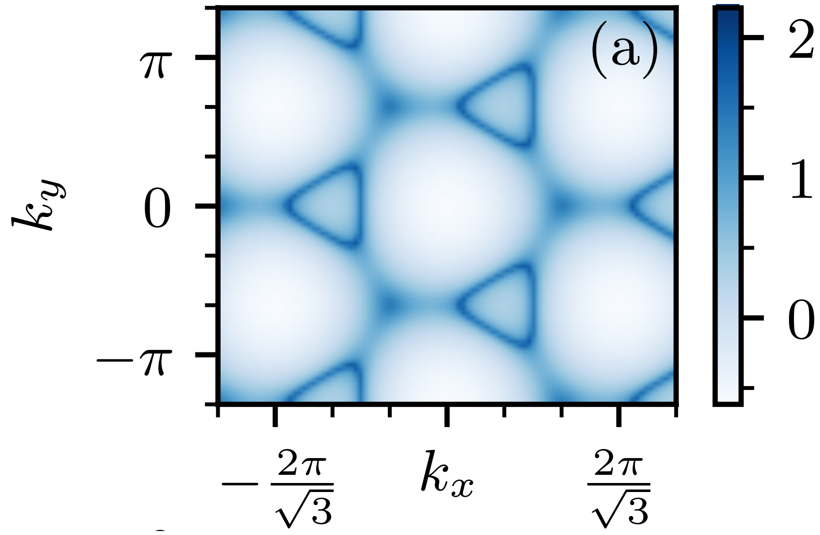
Also works for 2D, 3D, etc.

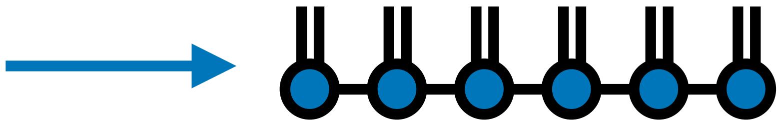
1D





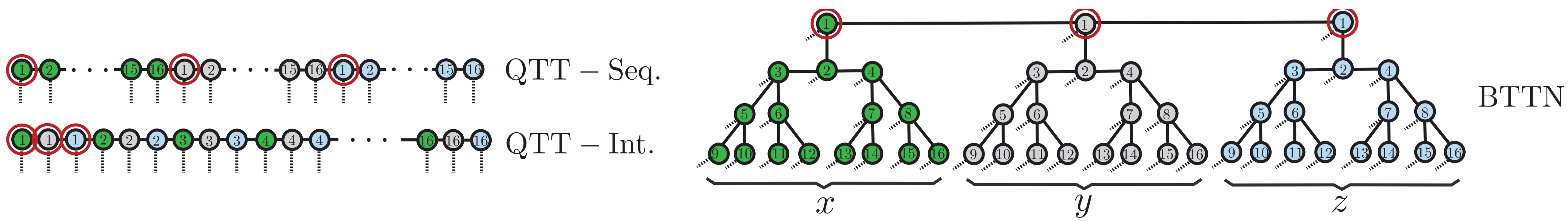
2D

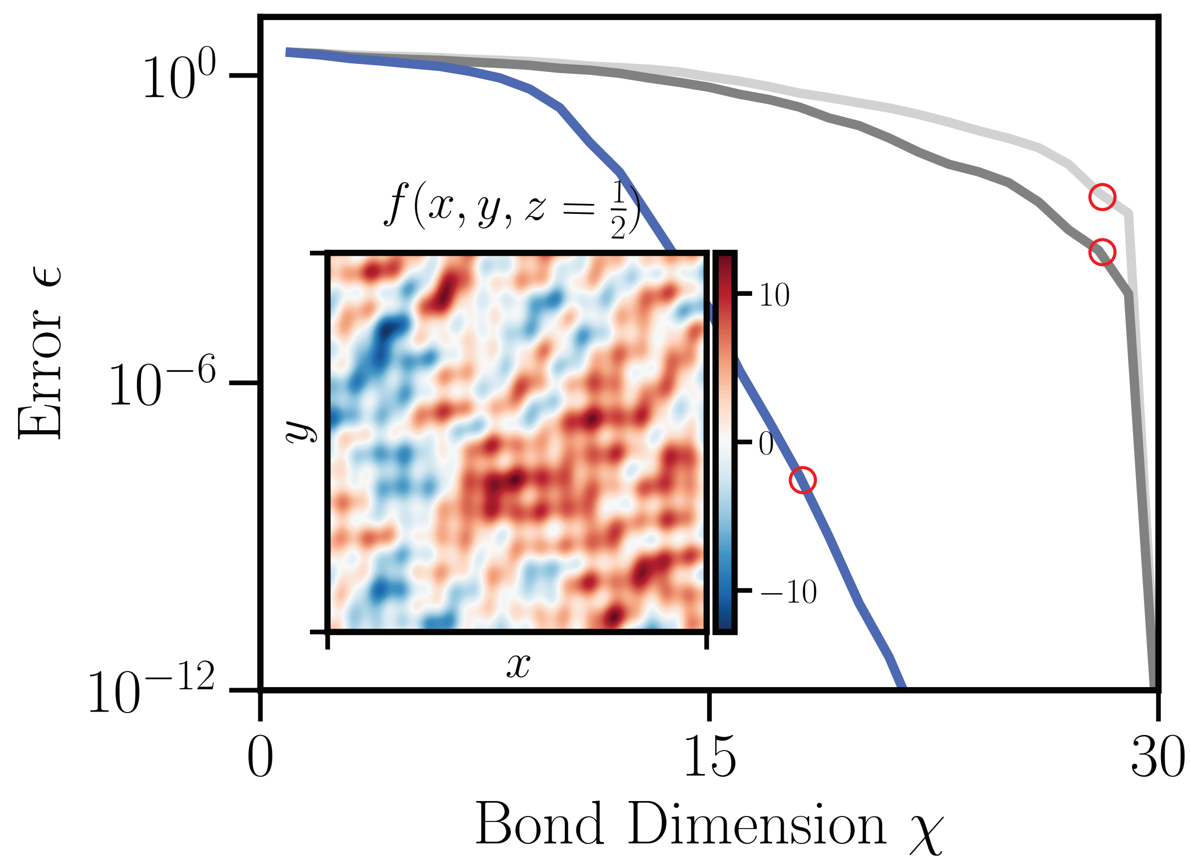


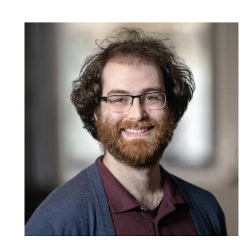


Quantum Functions

Explored further at CCQ: tree architectures for functions





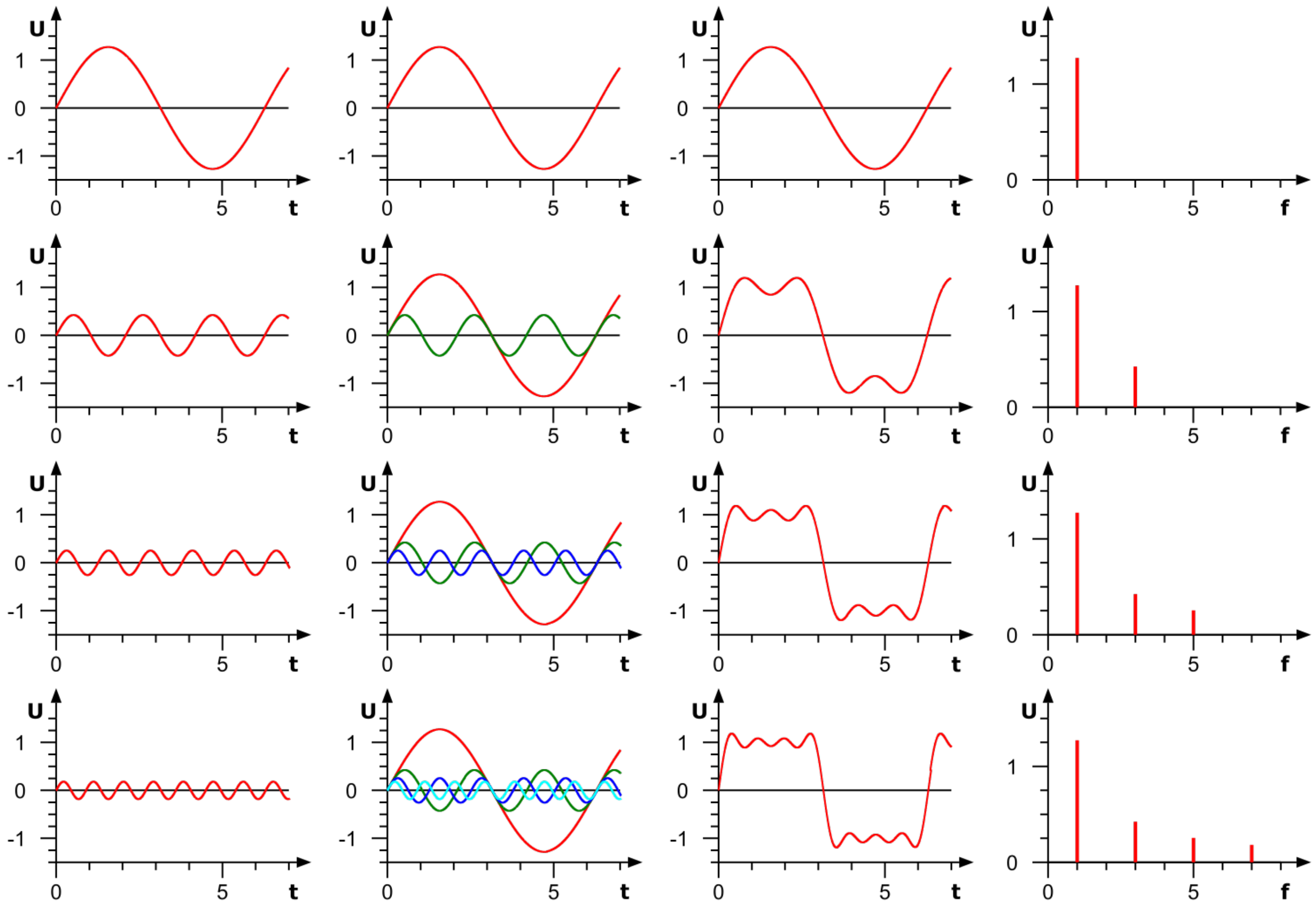

Ryan Levy


Joey Tindall

"Compressing multivariate functions with tree tensor networks", arxiv:2410.03572

Quantum Functions

What about algorithms for "tensorized" functions?



[1] Garcia-Ripoll, Quantum 5, 431 (2021)
[2] Dolgov, Savostyanov, Computer Physics Communications, 246, 106869 (2020)
[3] Khoromskij, Constr Approx 34, 257–280 (2011)
[4] Khoromskij, arxiv:1408.4053
[5] Gourianov et al., Nature Computational Science 2, 30–37 (2022)
[6] Shinaoka et al., Phys. Rev. X 13, 021015 (2023)
[7] Ye, Loureiro, Phys. Rev. E 106, 035208 (2022)
[8] Chen, EMS, White, PRX Quantum, arxiv:2210.08468
[9] Chertkov et al., arxiv:2209.14808
[10] Lin, White, arxiv:2308.09001
[11] White, EMS, New J. Phys 12, 055026 (2010)
[12] Ferris, Vidal, Phys. Rev. B 85, 165146 (2012)
[13] Chen, Lindsey, arxiv:2404.03182 (2024)

Quantum-Inspired Algorithms

By now, many algorithms known for tensor network functions [1]

- active learning of functions [2]
- integration [3]
- solving partial differential equations [4,5,6,7]
- Fourier transform and convolutions [8,6,13]
- find extrema (global min & max) without gradients [9]
- interpolation [1] & extrapolation of functions [10]
- perfect sampling [11,12]

Quantum-Inspired Algorithms

[1] Garcia-Ripoll, Quantum 5, 431 (2021)
[2] Dolgov, Savostyanov, Computer Physics Communications, 246, 106869 (2020)
[3] Khoromskij, Constr Approx 34, 257–280 (2011)
[4] Khoromskij, arxiv:1408.4053
[5] Gourianov et al., Nature Computational Science 2, 30–37 (2022)
[6] Shinaoka et al., Phys. Rev. X 13, 021015 (2023)
[7] Ye, Loureiro, Phys. Rev. E 106, 035208 (2022)
[8] Chen, EMS, White, PRX Quantum, arxiv:2210.08468
[9] Chertkov et al., arxiv:2209.14808
[10] Lin, White, arxiv:2308.09001
[11] White, EMS, New J. Phys 12, 055026 (2010)
[12] Ferris, Vidal, Phys. Rev. B 85, 165146 (2012)
[13] Chen, Lindsey, arxiv:2404.03182 (2024)

By now, many algorithms known for tensor network functions [1]

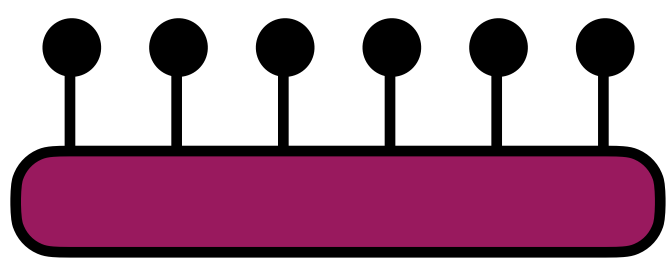
- active learning of functions [2]
- $\boxed{\text{integration [3]}}$
- solving partial differential equations [4,5,6,7]
- Fourier transform and convolutions [8,6,13]
- find extrema (global min & max) without gradients [9]
- interpolation [1] & extrapolation of functions [10]
- perfect sampling [11,12]

Quantum-Inspired Algorithms

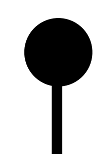
Integration of a tensorized function

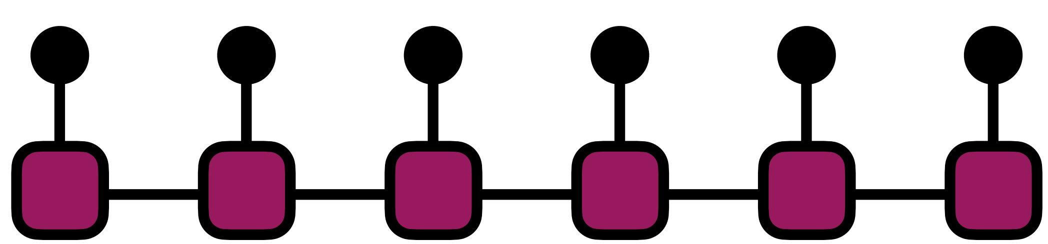
$$\int_0^1 dx\, f(x) \;\approx\; \sum_{b_1,b_2,\ldots} \frac{1}{2^n}\, f\!\left(\frac{b_1}{2} + \frac{b_2}{2^2} + \frac{b_3}{2^3} + \frac{b_4}{2^4} + \ldots + + \frac{b_n}{2^n}\right)$$

$$= \sum_{b_1,b_2,\ldots} \frac{1}{2^n}\, F^{b_1 b_2 b_3 b_4 \cdots b_n}$$

$=$ 

F

 $= \begin{bmatrix} 1/2 \\ 1/2 \end{bmatrix}$

$\approx$ 

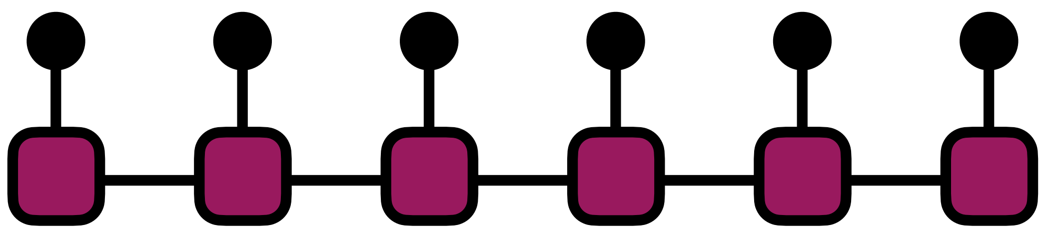
Cost:

$n\chi^2 = \log(N)\;\chi^2$

Quantum-Inspired Algorithms

Integration of a tensorized function

$$\int_0^1 dx\, f(x) \;\approx$$



Entire integration code in 5 lines of ITensor

```
function integrate(psi::MPS)
  I = ITensor(1.)
  for (j,s) in enumerate(siteinds(psi))
    I *= (psi[j]*ITensor([1/2,1/2],s))
  end
  return scalar(I)
end
```

Quantum-Inspired Algorithms

[1] Garcia-Ripoll, Quantum 5, 431 (2021)
[2] Dolgov, Savostyanov, Computer Physics Communications, 246, 106869 (2020)
[3] Khoromskij, Constr Approx 34, 257–280 (2011)
[4] Khoromskij, arxiv:1408.4053
[5] Gourianov et al., Nature Computational Science 2, 30–37 (2022)
[6] Shinaoka et al., Phys. Rev. X 13, 021015 (2023)
[7] Ye, Loureiro, Phys. Rev. E 106, 035208 (2022)
[8] Chen, EMS, White, PRX Quantum, arxiv:2210.08468
[9] Chertkov et al., arxiv:2209.14808
[10] Lin, White, arxiv:2308.09001
[11] White, EMS, New J. Phys 12, 055026 (2010)
[12] Ferris, Vidal, Phys. Rev. B 85, 165146 (2012)
[13] Chen, Lindsey, arxiv:2404.03182 (2024)

By now, many algorithms known for tensor network functions [1]

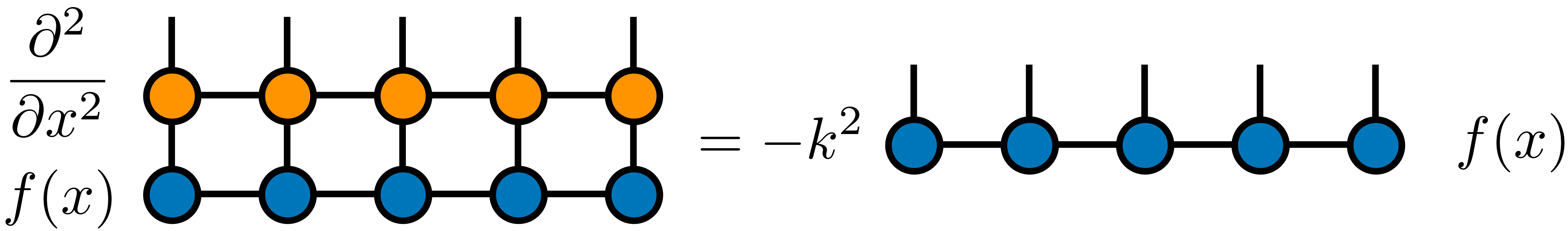
- active learning of functions [2]
- integration [3]
- $\boxed{\text{solving partial differential equations [4,5,6,7]}}$
- Fourier transform and convolutions [8,6,13]
- find extrema (global min & max) without gradients [9]
- interpolation [1] & extrapolation of functions [10]
- perfect sampling [11,12]

Quantum-Inspired Algorithms

Given a diff. eq. such as wave equation

$$\frac{\partial^2}{\partial x^2} f(x) = -k^2 f(x)$$

Can encode as tensor network equation

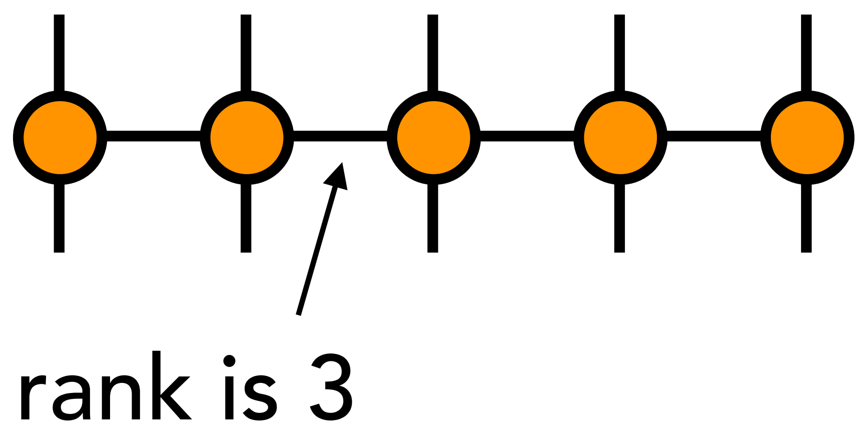


Use DMRG algorithm to efficiently find eigenvector
or can time step

Quantum-Inspired Algorithms

Finite-difference formula for $\dfrac{\partial^2}{\partial x^2}$

translates into exact expression for
low-rank *matrix product operator* (MPO) tensor network



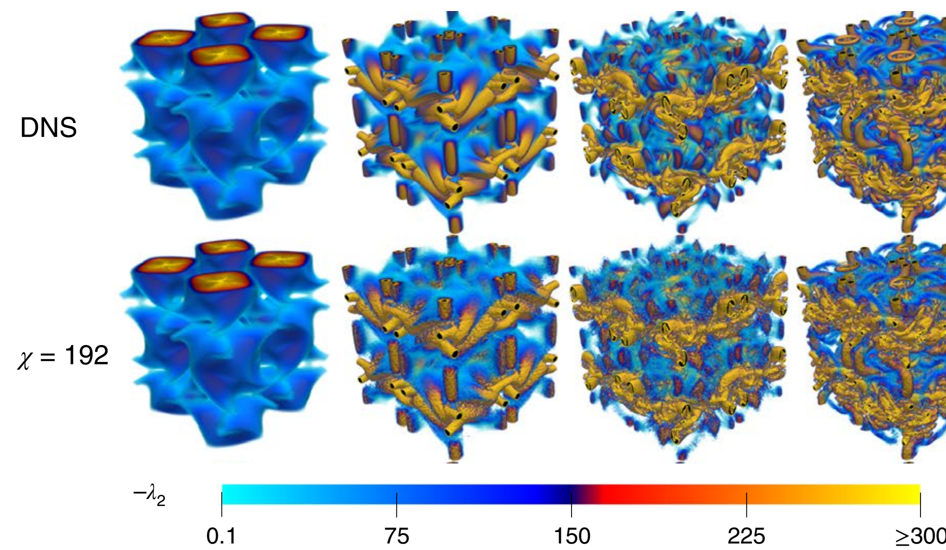
Basically: forward binary adder, backwards binary adder,
plus constant = $(x_{j+1} - 2x_j + x_{j-1})/a^2$

Lubasch, Moinier, Jaksch, J. Comp. Phys. 372, 587 (2018)

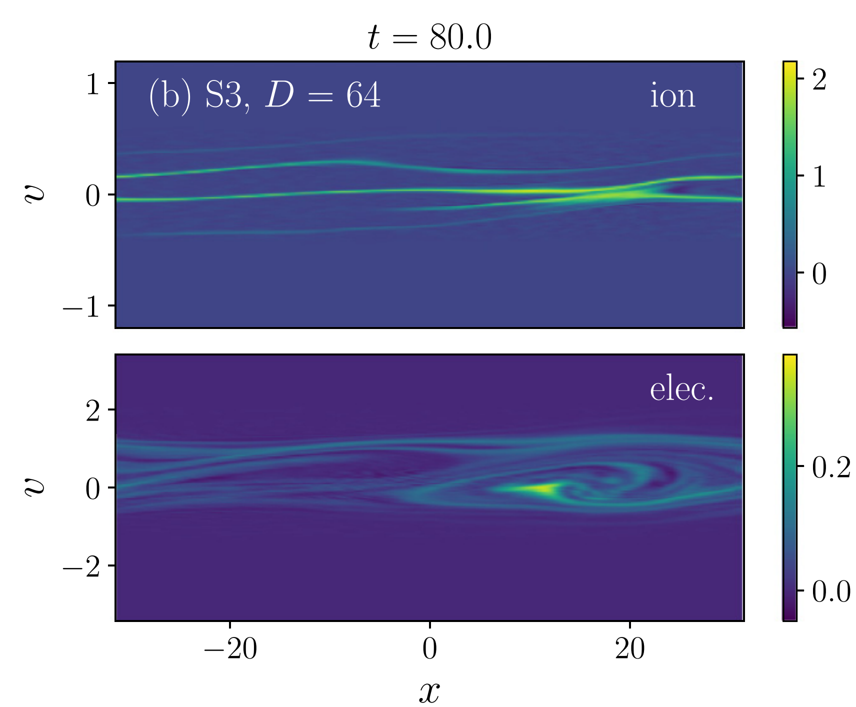
Quantum-Inspired Algorithms

Application: Fluid mechanics

Navier-Stokes equation:
Rank χ determined by Reynolds number [1]



Plasma flows:
(Vlasov-Poisson equation) [2]



[1] Gourianov et al., Nature Comp. Sci., 2, 30-37 (2022)
[2] Ye, Loureiro, PRE 106, 035208 (2022)

Quantum-Inspired Algorithms

[1] Garcia-Ripoll, Quantum 5, 431 (2021)
[2] Dolgov, Savostyanov, Computer Physics Communications, 246, 106869 (2020)
[3] Khoromskij, Constr Approx 34, 257–280 (2011)
[4] Khoromskij, arxiv:1408.4053
[5] Gourianov et al., Nature Computational Science 2, 30–37 (2022)
[6] Shinaoka et al., Phys. Rev. X 13, 021015 (2023)
[7] Ye, Loureiro, Phys. Rev. E 106, 035208 (2022)
[8] Chen, EMS, White, PRX Quantum, arxiv:2210.08468
[9] Chertkov et al., arxiv:2209.14808
[10] Lin, White, arxiv:2308.09001
[11] White, EMS, New J. Phys 12, 055026 (2010)
[12] Ferris, Vidal, Phys. Rev. B 85, 165146 (2012)
[13] Chen, Lindsey, arxiv:2404.03182 (2024)

By now, many algorithms known for tensor network functions [1]

- active learning of functions[2]
- integration [3]
- solving partial differential equations [4,5,6,7]
- Fourier transform and convolutions [8,6,13]
- find extrema (global min & max) without gradients [9]
- interpolation [1] & extrapolation of functions [10]
- perfect sampling [11,12]

Tensor Cross Interpolation

How do we get an complicated function into a tensor network?

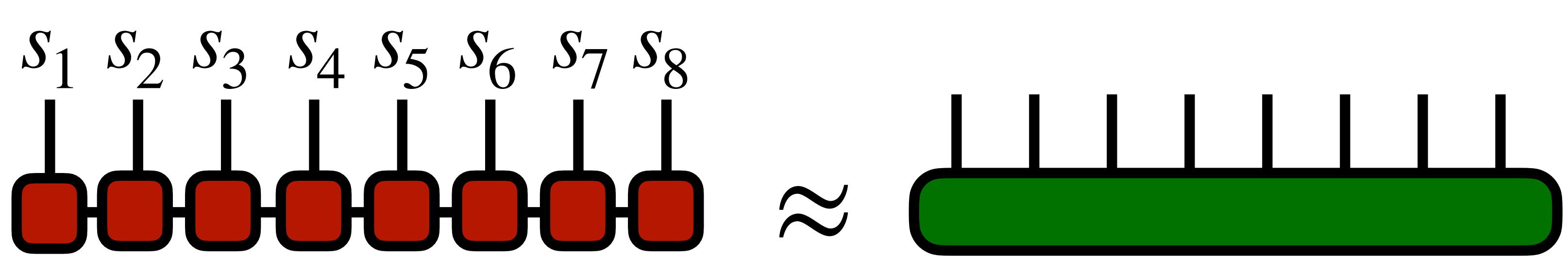
Using **tensor cross interpolation** algorithm

queryable function

$$f(s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8)$$

entire function f

MPS approximation

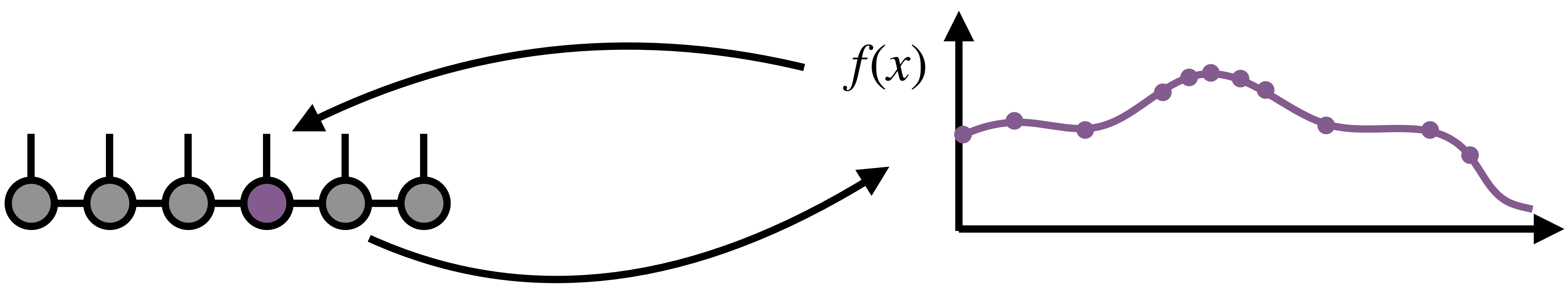


I. Oseledets and E. Tyrtyshnikov, Linear Algebra Appl. 432, 70 (2010)

Tensor Cross Interpolation

It is an "active learning" algorithm

query values



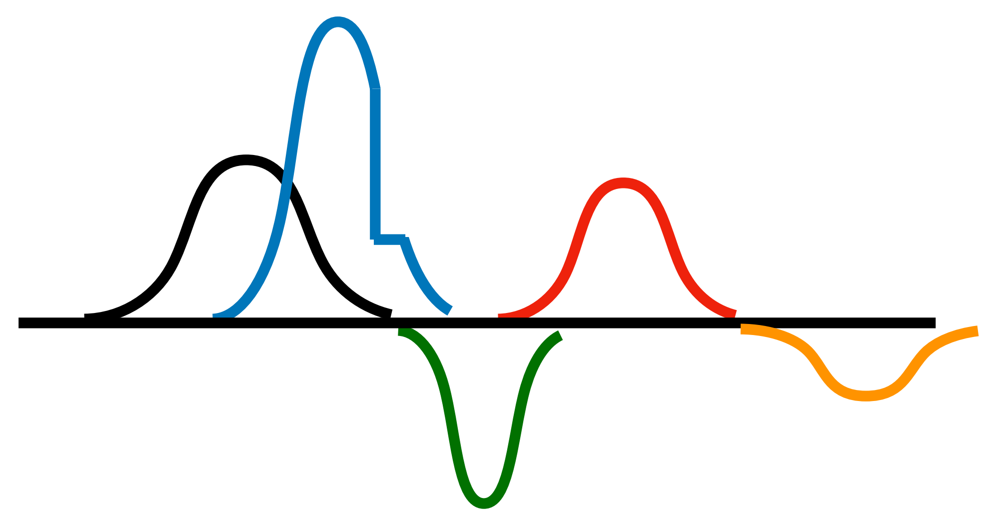
suggest better points via
interpretable linear algebra

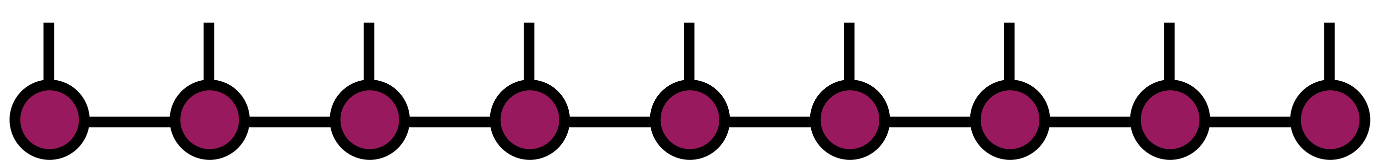
Tensor Cross Interpolation

Live demo – learning a 1D function:

40 Gaussians, random location, width, & height
+ a sharp step at 0.4

$$f(x) = \sum_{g=1}^{N_g} a_g e^{-w_g (x - x_g)^2}$$
$$+ \, 0.4 \cdot \Theta(x)$$

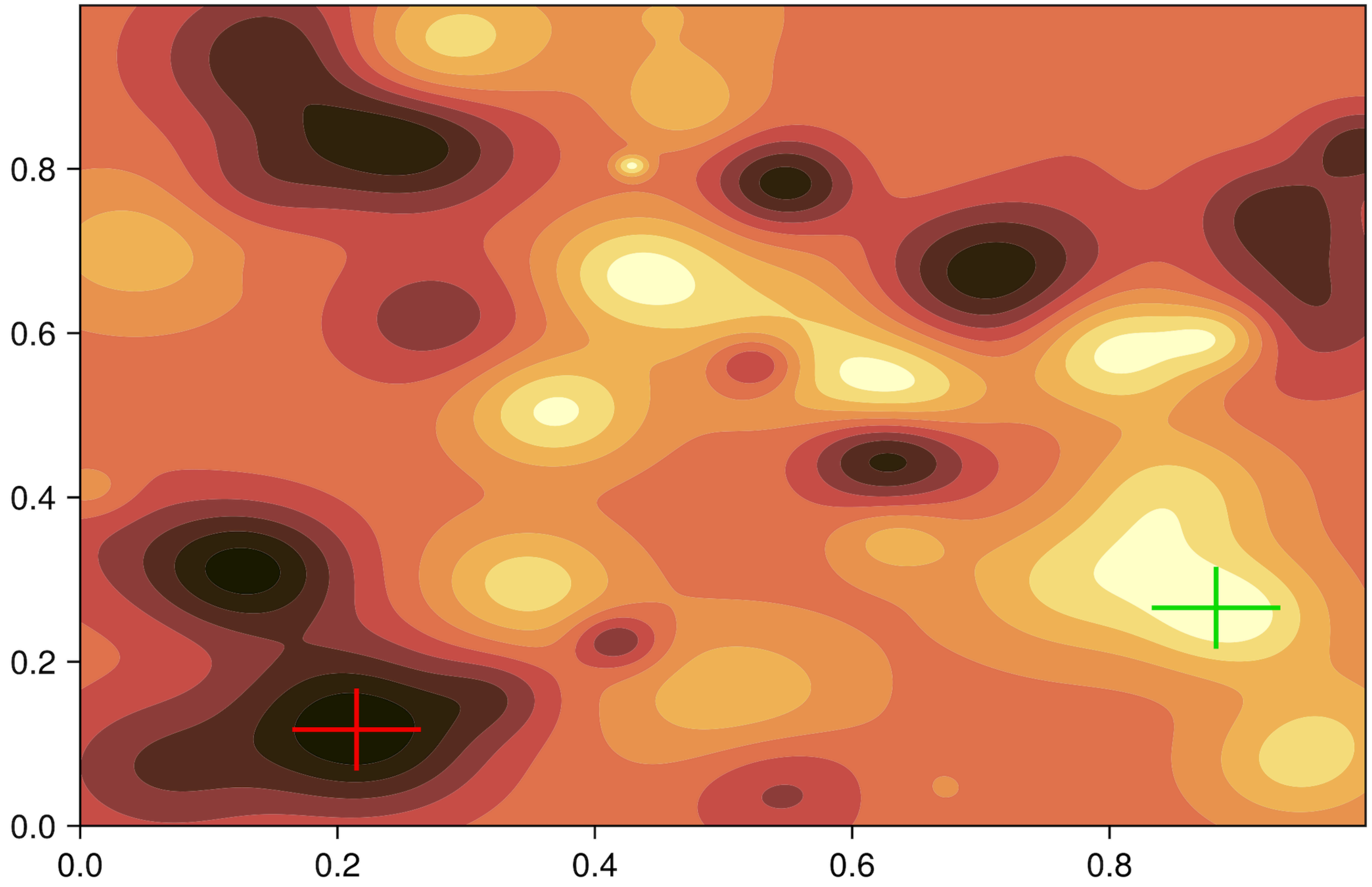


$f(x) \approx$ 

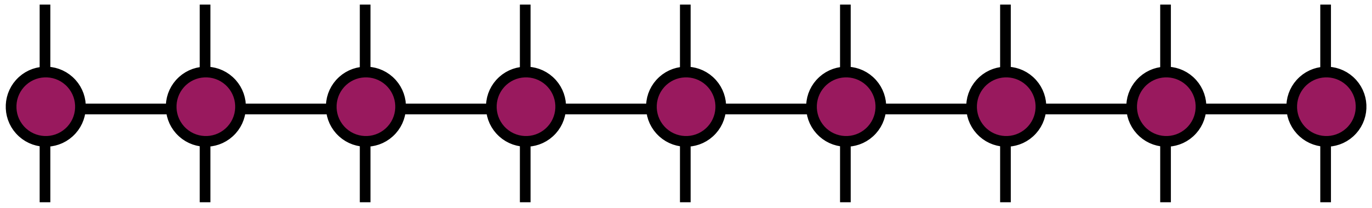
Tensor Cross Interpolation

Next demo: 2D function and optima

50 **2D** Gaussians, random location, width, & height



$$f(x) \approx$$

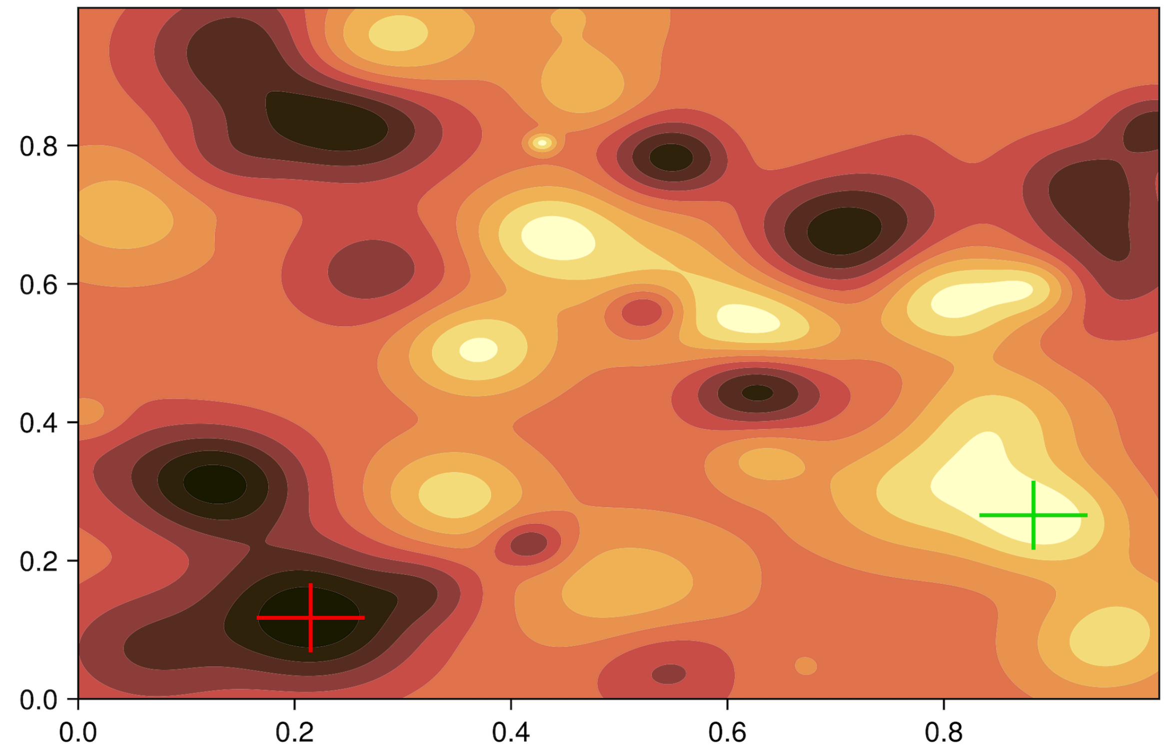


Tensor Cross Interpolation

How were optima (global min and max) found?

Very interesting proposal called "**TTOpt**" [1,2]

Claim: tensor interpolation usually "encounters" global min and max [1]

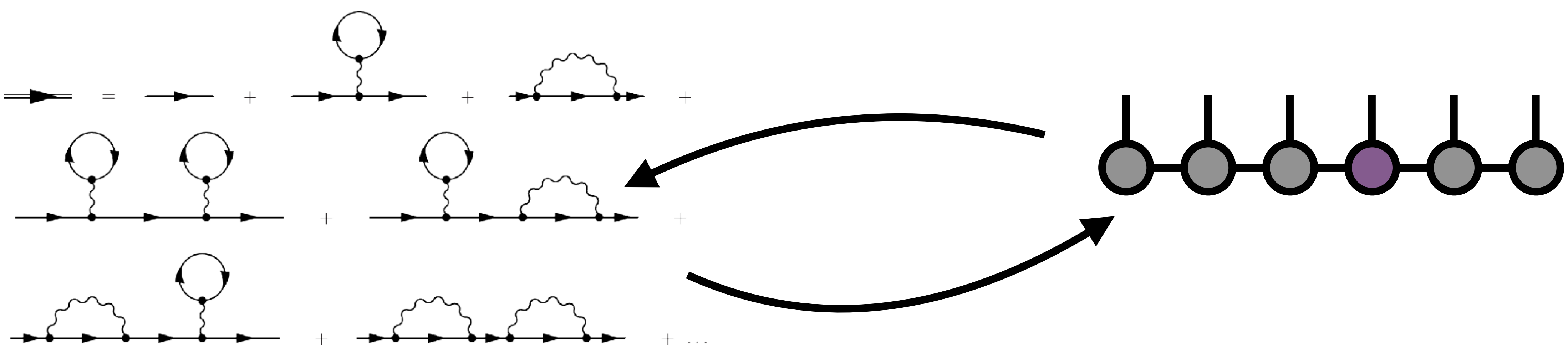


[1] Sozykin, Chertkov, Schutski, Phan, Cichocki, Oseledets, TTOpt, NeurIPS (2022)

[2] Chertkov, Ryzhakov, Novikov, Oseledets, arxiv:2209.14808

Tensor Cross Interpolation

TCI method used to
learn and sum entire orders of Feynman diagrams*

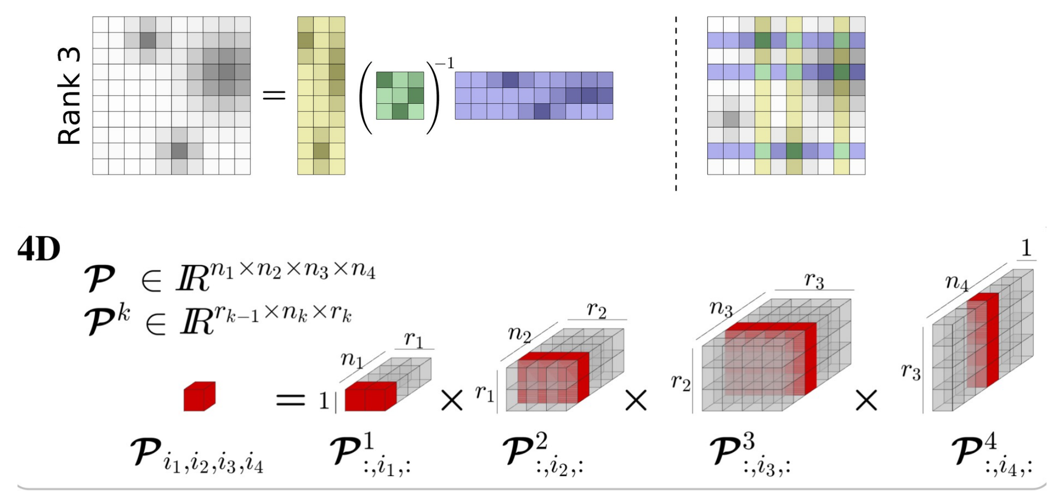


Solve quantum dots to long times, as post-processing

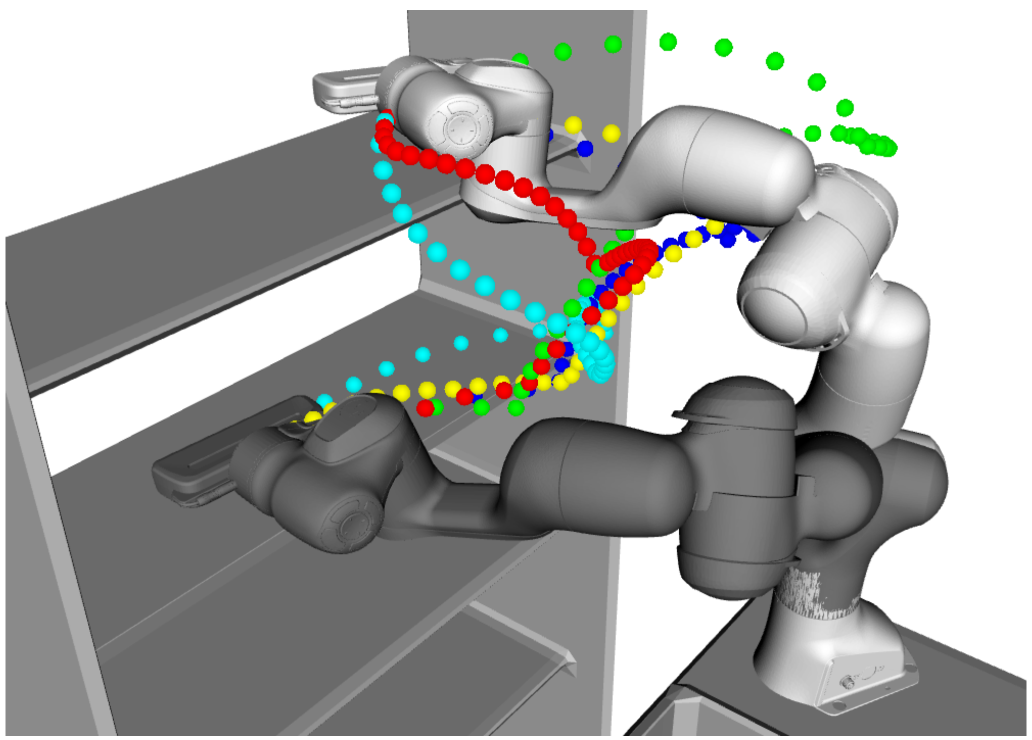
* Nunez Fernandez, et al. Phys. Rev. X 12, 041018 (2022)

Tensor Cross Interpolation

TCI (= TT-Cross) algorithm



Fun paper using TTOpt to control a real robot arm! *



(a)

Figure 1. Solutions from TTGO for motion planning of a manipulator from a given initial configuration (white) to a final configuration (dark). The obtained joint angle trajectories result in different paths for the end.effector which are highlighted by dotted curves in different colors. The multimodality is clearly visible from these solutions.

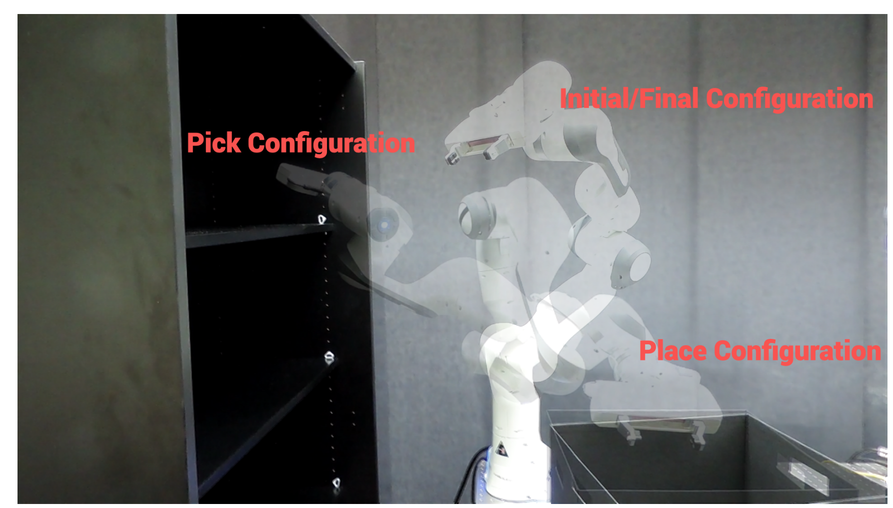


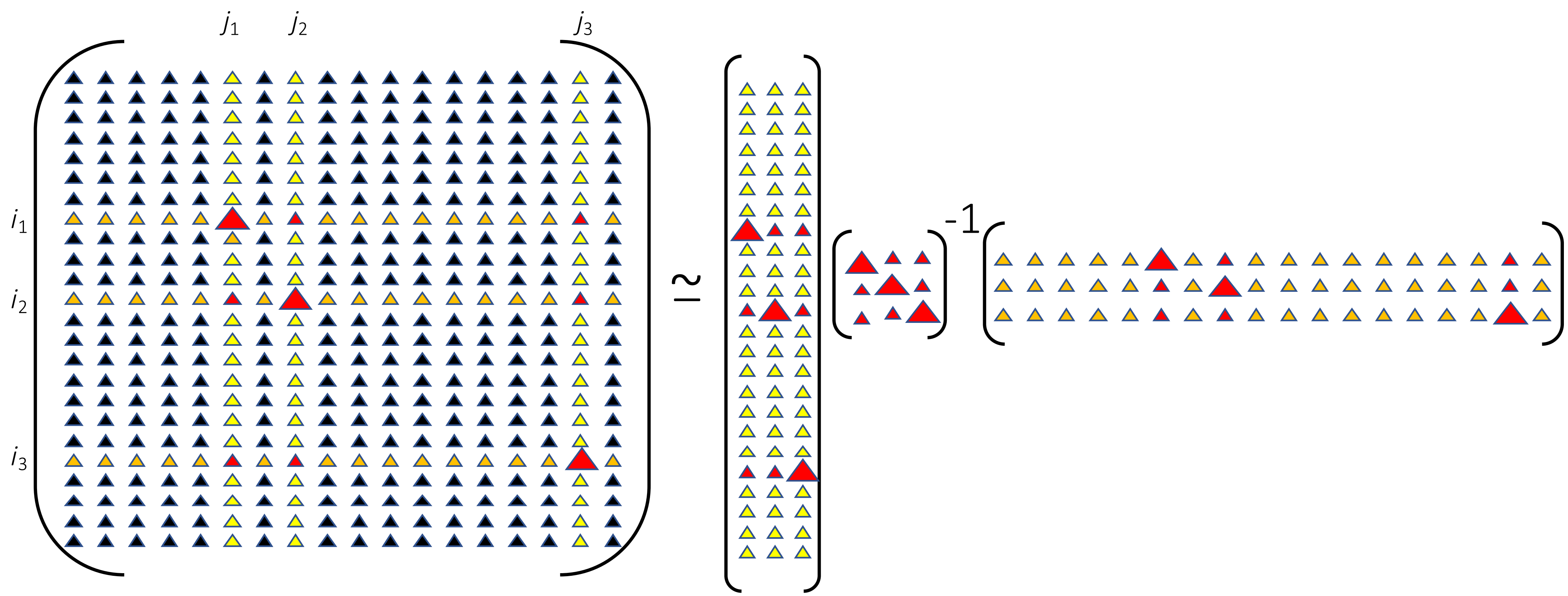
Figure 12. Real robot implementation of one of the TTGO solutions for the pick-and-place task. The motion from the initial configuration to the final configuration (same as the initial configuration in this case) via the picking configuration and placing configuration is depicted.

Shetty, Lembono, Loew, and Calinon, Tensor Train for Global Optimization Problems in Robotics, Int. J. Robotics Research (2023)

Tensor Cross Interpolation

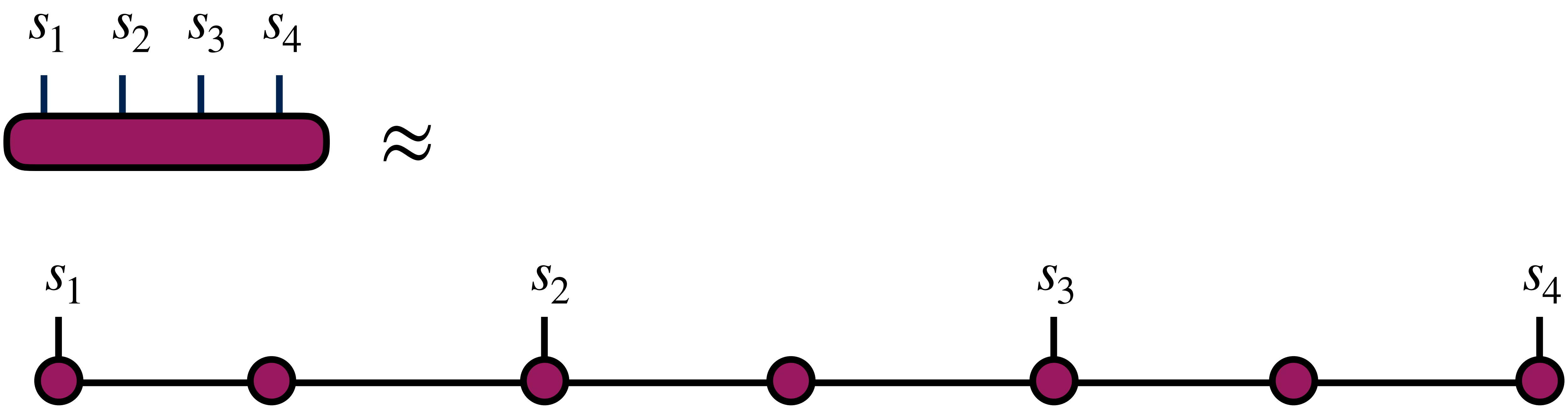
How does tensor cross interpolation work in a bit more detail?

Foundation is "cross approximation" of a matrix



Tensor Cross Interpolation

Tensor cross interpolation extends this to MPS tensor network



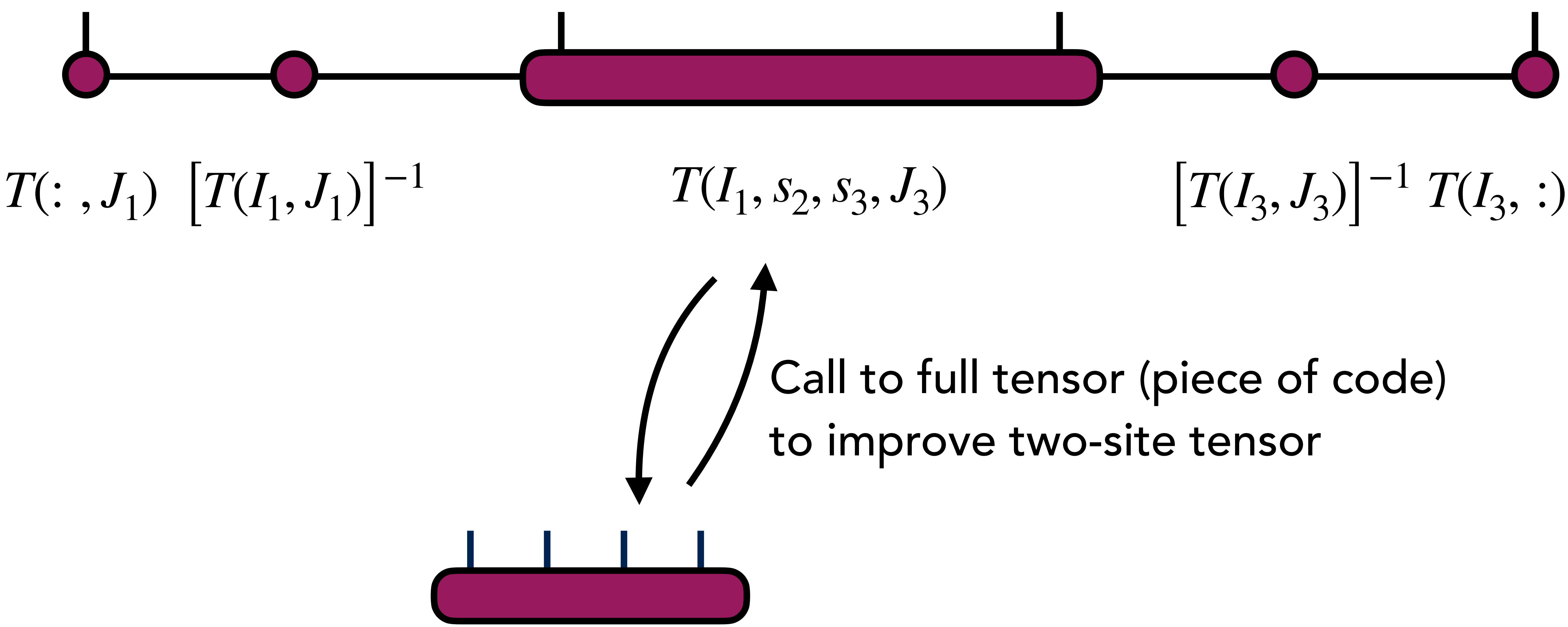
$$T(s_1, J_1)\ \left[T(I_1, J_1)\right]^{-1}\ T(I_1, s_2, J_2)\ \left[T(I_2, J_2)\right]^{-1}\ T(I_2, s_3, J_3)\left[T(I_3, J_3)\right]^{-1}\ T(I_3, s_4)$$

Entries of small tensor are exactly entries of big tensor

Interpolation points or pivots

Tensor Cross Interpolation

Tensor cross interpolation proceeds by merging pairs of small tensors then improving pivots

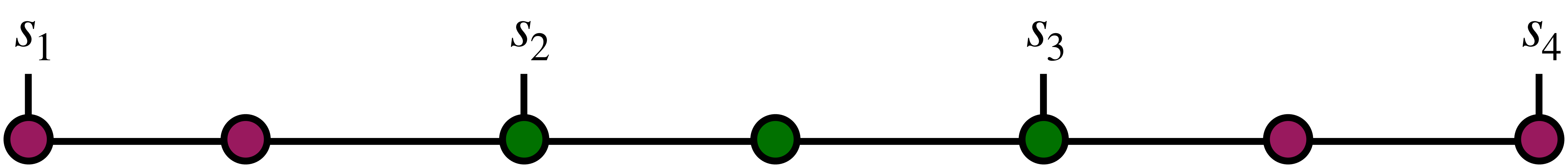


Tensor Cross Interpolation

Tensor cross interpolation proceeds by merging pairs of small tensors then improving pivots

$$T(:\,,J_1)\;\left[T(I_1,J_1)\right]^{-1}\quad T(I_1,s_2,s_3,J_3)\quad \left[T(I_3,J_3)\right]^{-1}\,T(I_3,\,:)$$

$$\approx$$

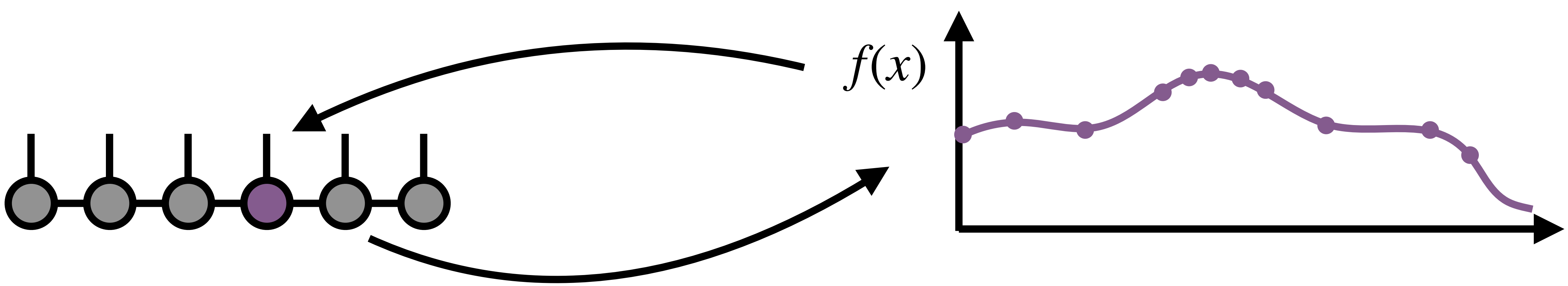


$$T(s_1,J_1)\;\left[T(I_1,J_1)\right]^{-1}\;T(I_1,s_2,J_2)\;\left[T(I_2,J_2)\right]^{-1}\;T(I_2,s_3,J_3)\left[T(I_3,J_3)\right]^{-1}\;T(I_3,s_4)$$

Tensor Cross Interpolation

Result is an "active learning" algorithm

query values



suggest better points via
interpretable linear algebra

Tensor Cross Interpolation

A leading TCI code is by Marc Ritter (LMU → CCQ)

https://github.com/tensor4all/TensorCrossInterpolation.jl

TensorCrossInterpolation

docs dev | CI passing

The TensorCrossInterpolation module implements the *tensor cross interpolation* algorithm for efficient interpolation of multi-index tensors and multivariate functions.

This algorithm is used in the *quantics tensor cross interpolation* (QTCI) method for exponentially efficient interpolation of functions with scale separation. QTCI is implemented in the QuanticsTCI.jl module.

Installation

This module has been registered in the General registry. It can be installed by typing the following in a Julia REPL:

```julia
using Pkg; Pkg.add("TensorCrossInterpolation")
```

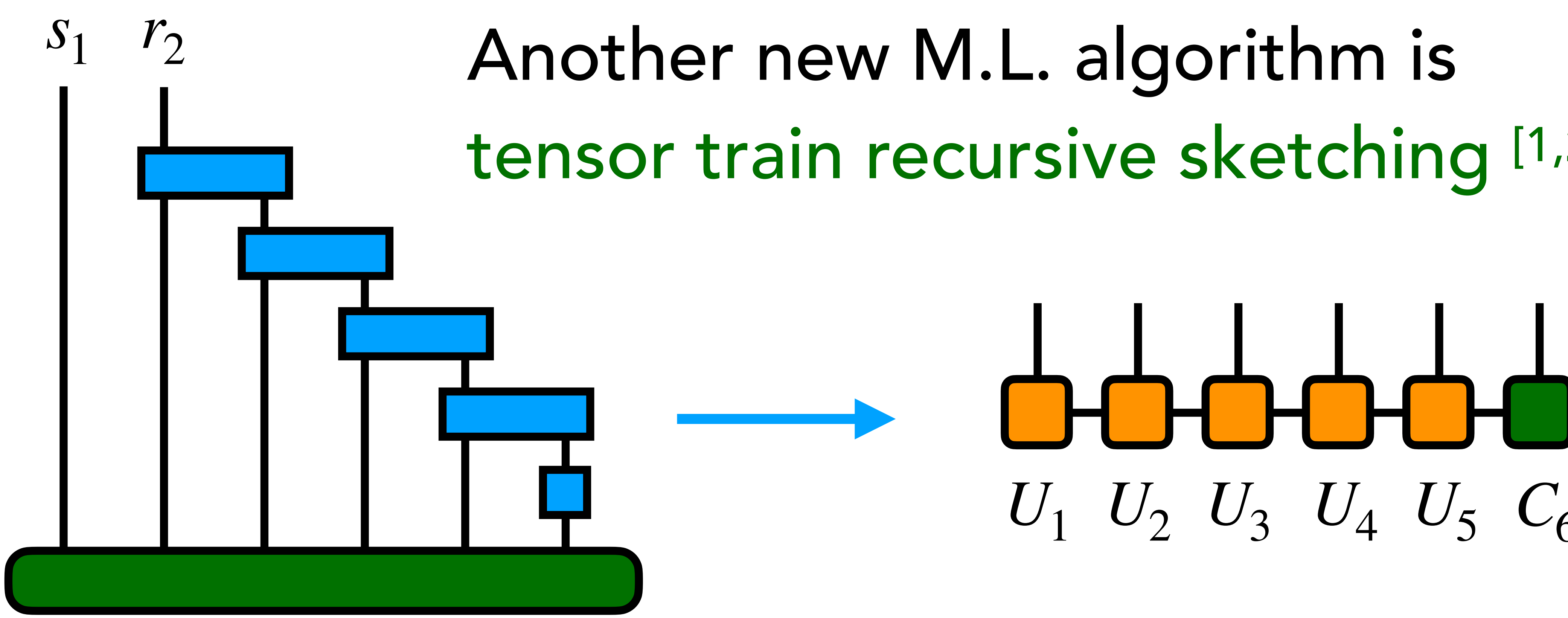
Results easy to convert to an ITensor MPS

Tensor Recursive Sketching

Yoonhaeng Hur | Jeremy Hoskins | Michael Lindsey | Yuehaw Khoo

Can we also learn a tensorized function from data / samples? Yes!

Another new M.L. algorithm is
tensor train recursive sketching [1,2]

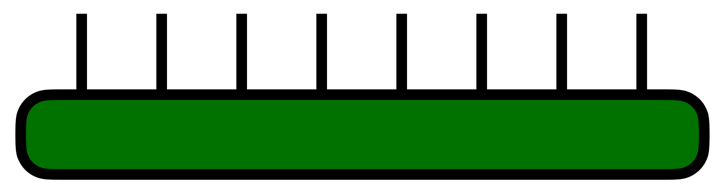


$$\sim \quad \begin{matrix} 1 & 2 & 1 & 1 & 1 & 2 \\ 2 & 2 & 1 & 2 & 1 & 2 \\ 2 & 2 & 2 & 2 & 1 & 2 \\ 1 & 1 & 2 & 2 & 2 & 2 \\ & & \vdots & & & \end{matrix}$$

[1] Generative Modeling via Tensor Train Sketching, arxiv: 2202.11788
[2] Generative Modeling via Hierarchical Tensor Sketching, arxiv: 2304.05305

Tensor Network Machine Learning

Given samples from true distribution

$p(s_1, s_2, \ldots, s_N) \; =$ 

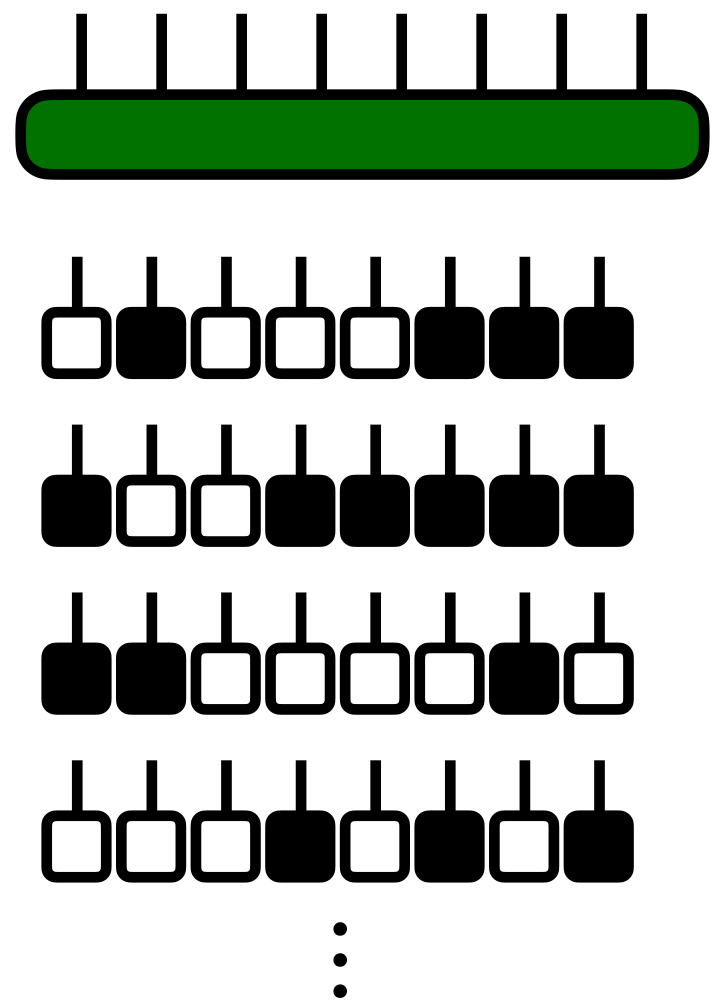
$$\sim \quad \begin{array}{cccccccc} 1 & 2 & 1 & 1 & 1 & 2 & 2 & 2 \\ 2 & 2 & 1 & 2 & 1 & 2 & 1 & 2 \\ 2 & 2 & 2 & 2 & 1 & 2 & 1 & 1 \\ 1 & 1 & 2 & 2 & 2 & 2 & 1 & 2 \\ & & & & \vdots & & & \end{array}$$

training data

Tensor Network Machine Learning

Given samples from true distribution

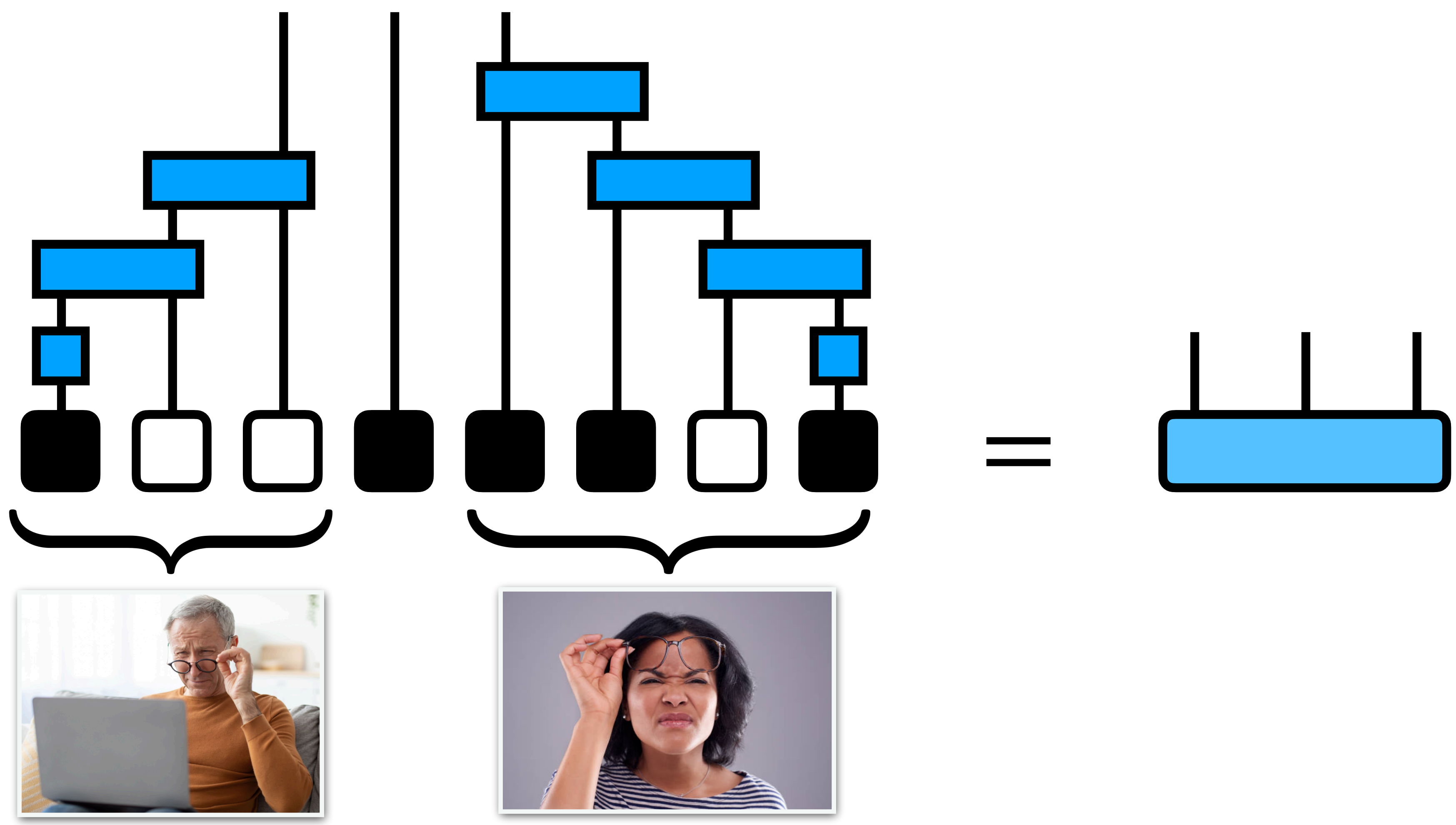
$$p(s_1, s_2, \ldots, s_N) \;=\; \qquad \sim$$



training data

Tensor Network Machine Learning

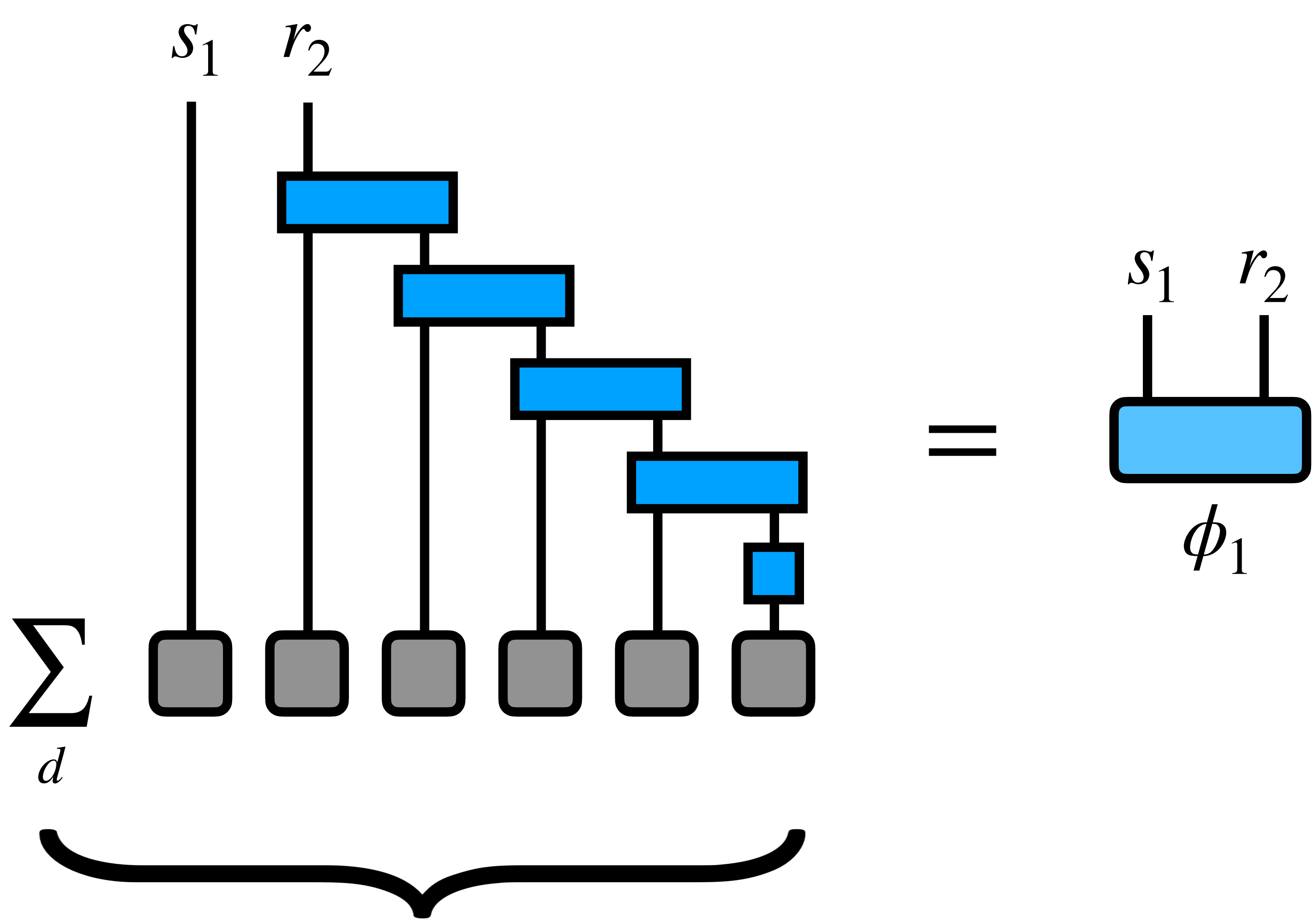
Apply empirical sketches to "blur" &
project data to lower-dimensional space



left sketch right sketch

Tensor Network Machine Learning

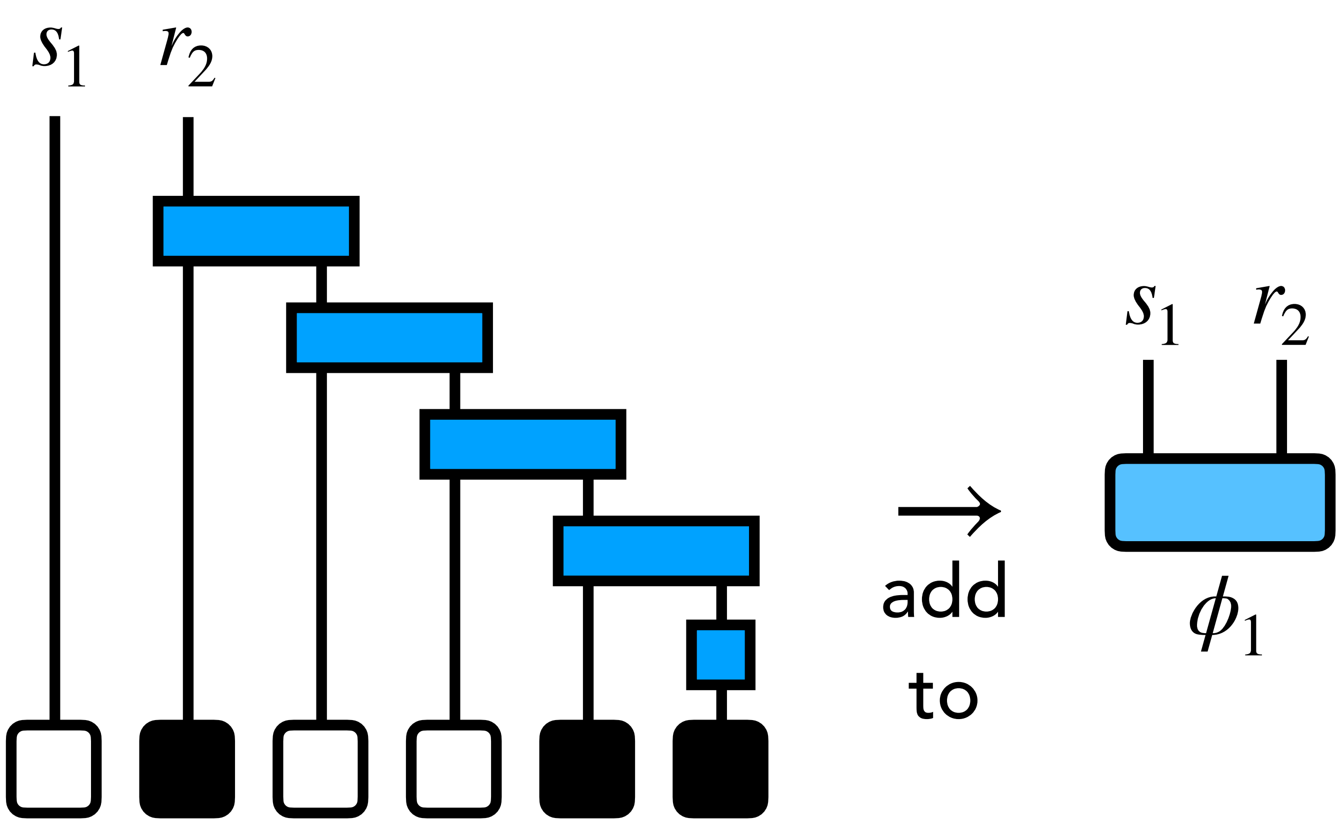
Sum up data through sketching process



$\hat{p}$, sum over all data in training set
(product states / rank-1 tensors)

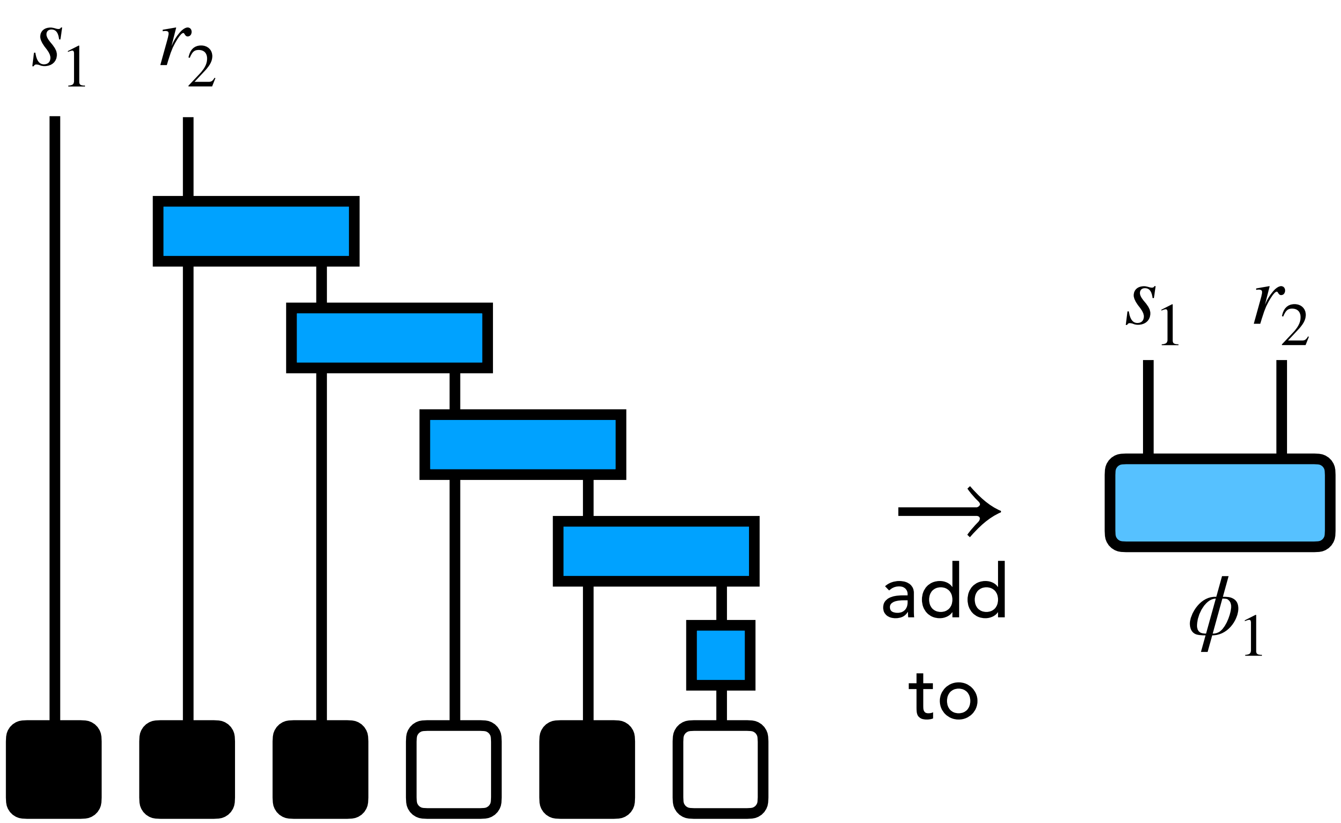
Tensor Network Machine Learning

Sum up data through sketching process



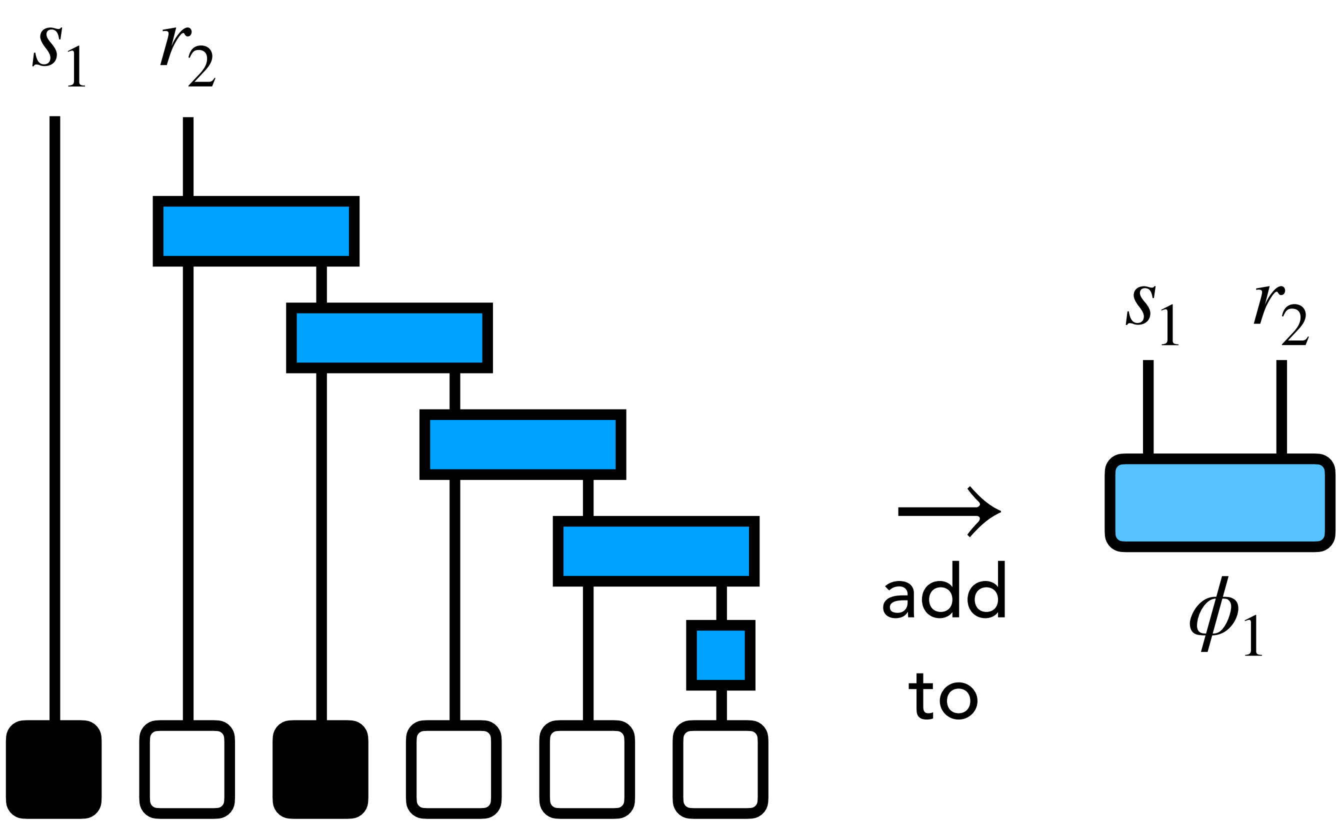
Tensor Network Machine Learning

Sum up data through sketching process



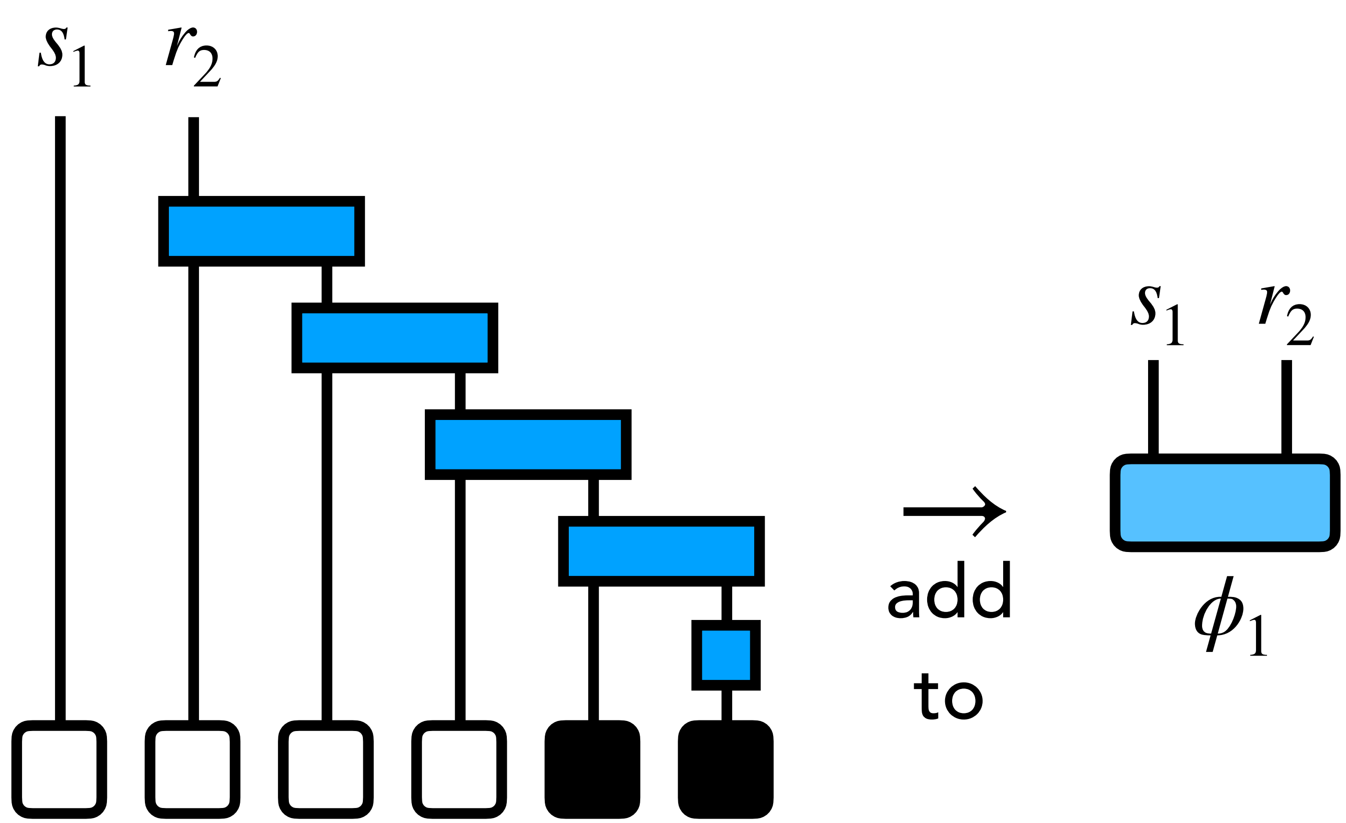
Tensor Network Machine Learning

Sum up data through sketching process



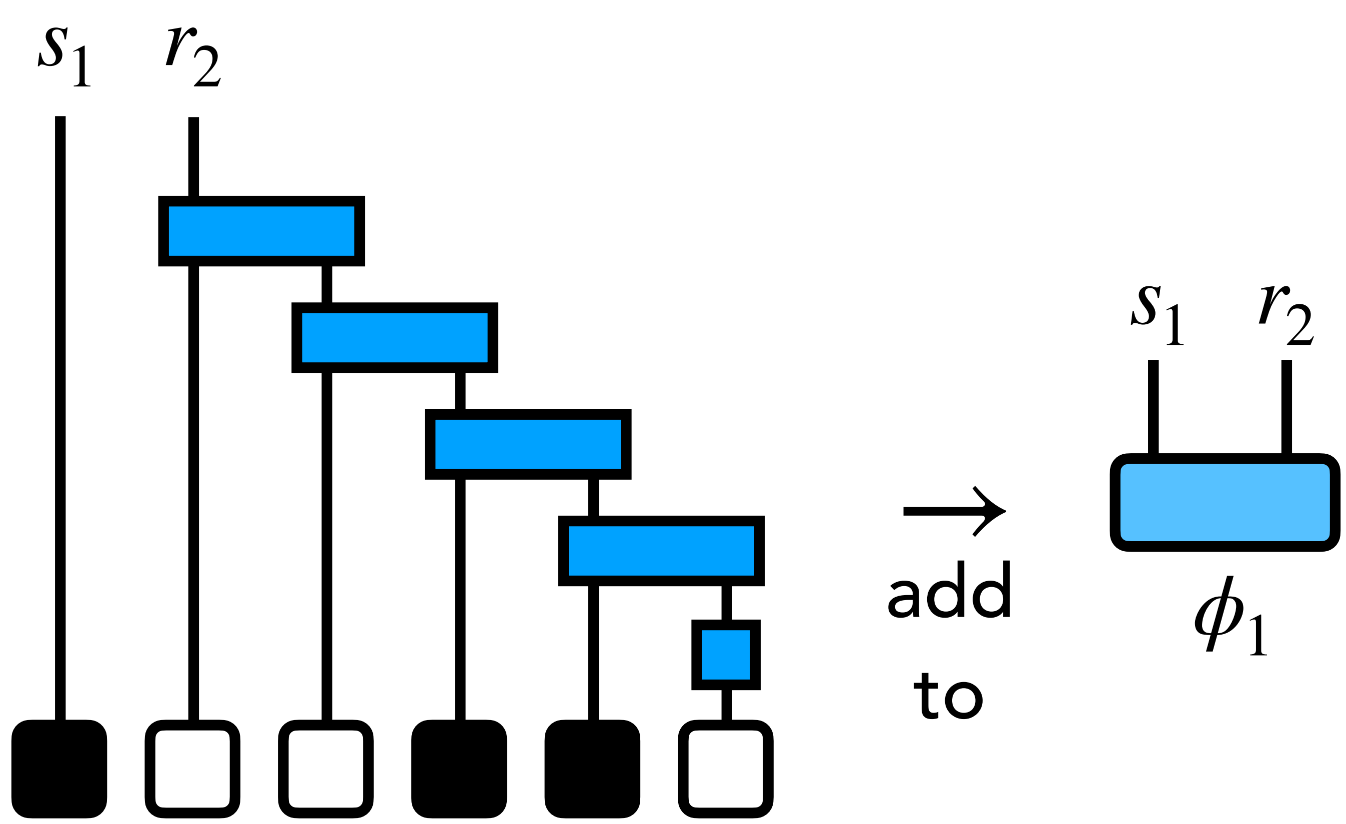
Tensor Network Machine Learning

Sum up data through sketching process



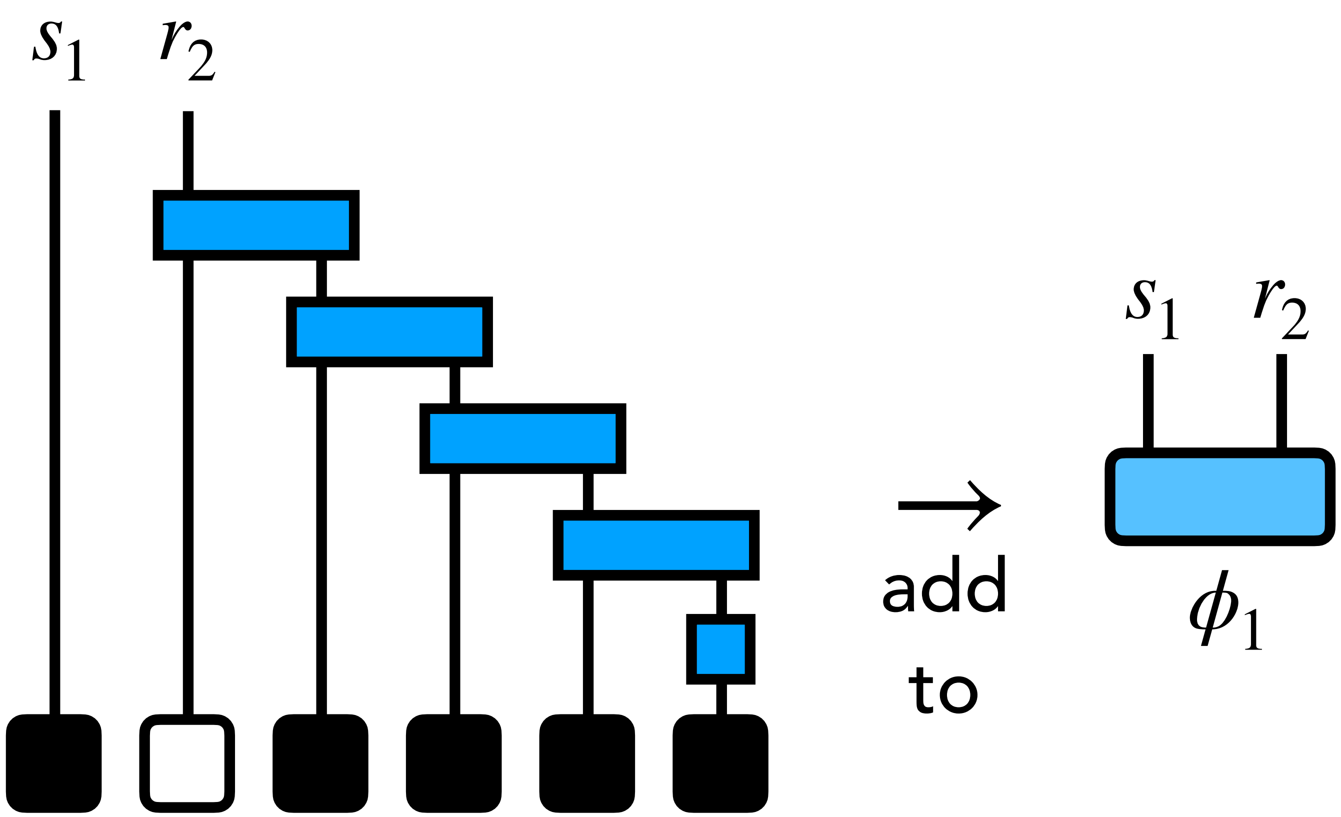
Tensor Network Machine Learning

Sum up data through sketching process



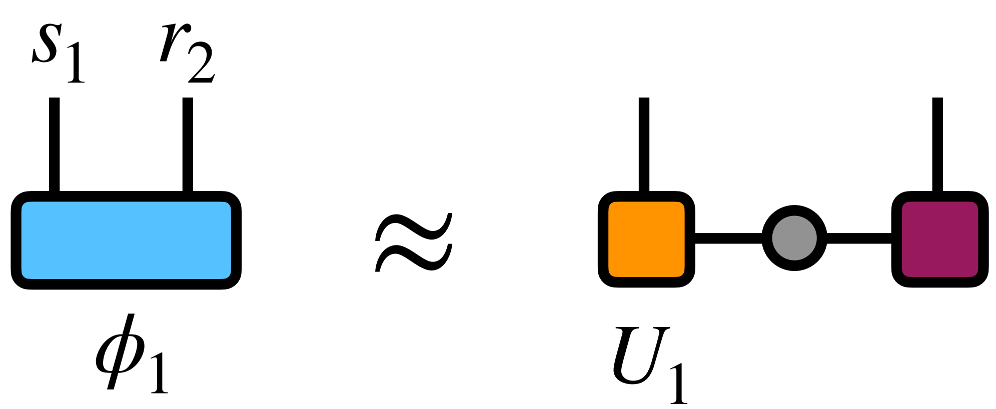
Tensor Network Machine Learning

Sum up data through sketching process



Tensor Network Machine Learning

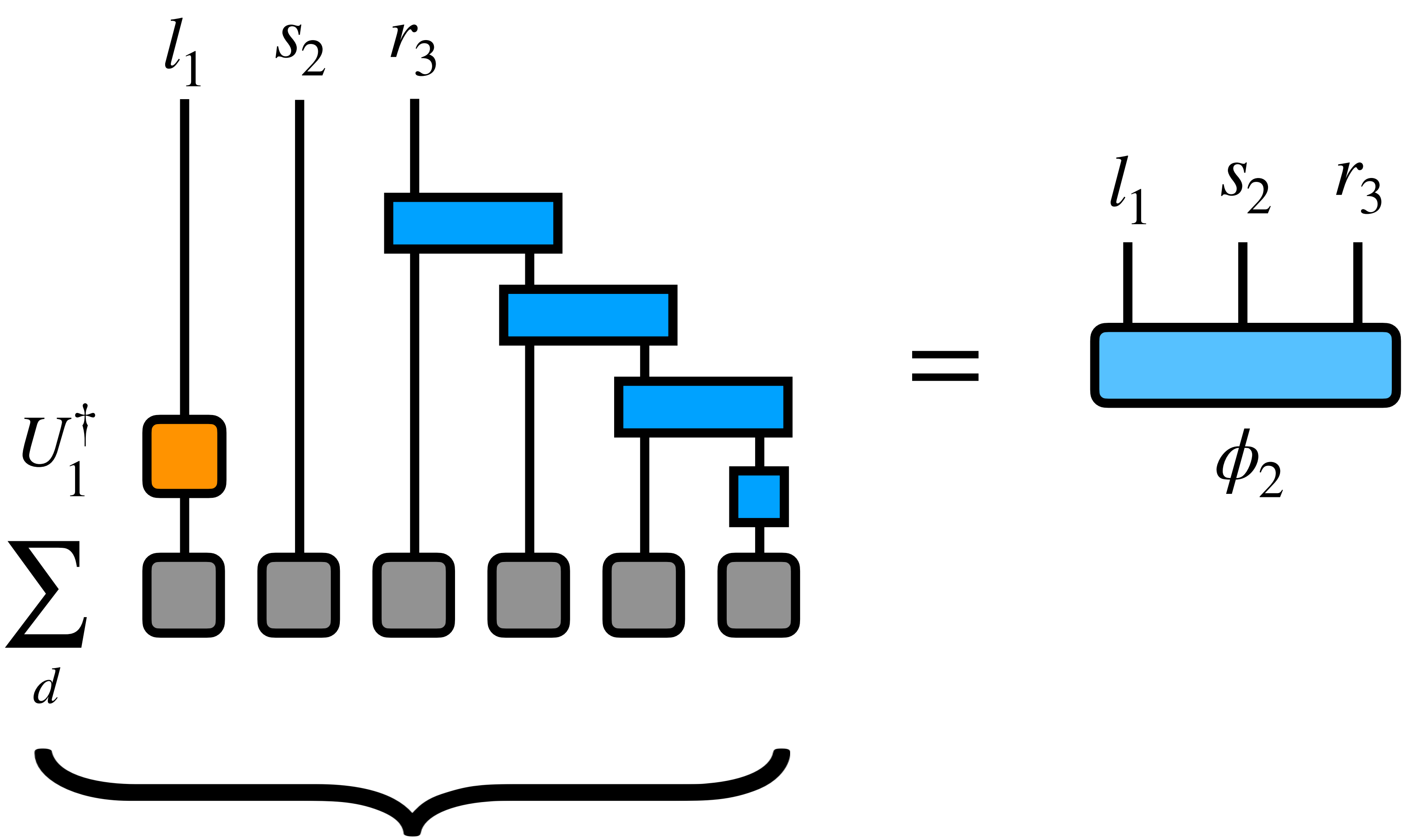
SVD of reduced data tensor gives first MPS tensor U_1



Save U_1 only

Tensor Network Machine Learning

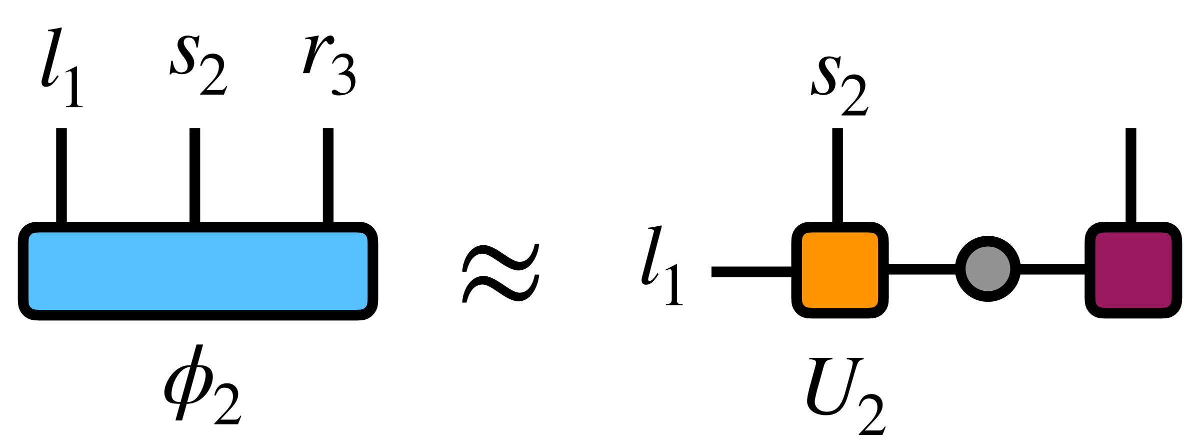
Recursively summing and sketching data continues process



$\hat{p}$, sum over all data in training set (product states / rank-1 tensors)

Tensor Network Machine Learning

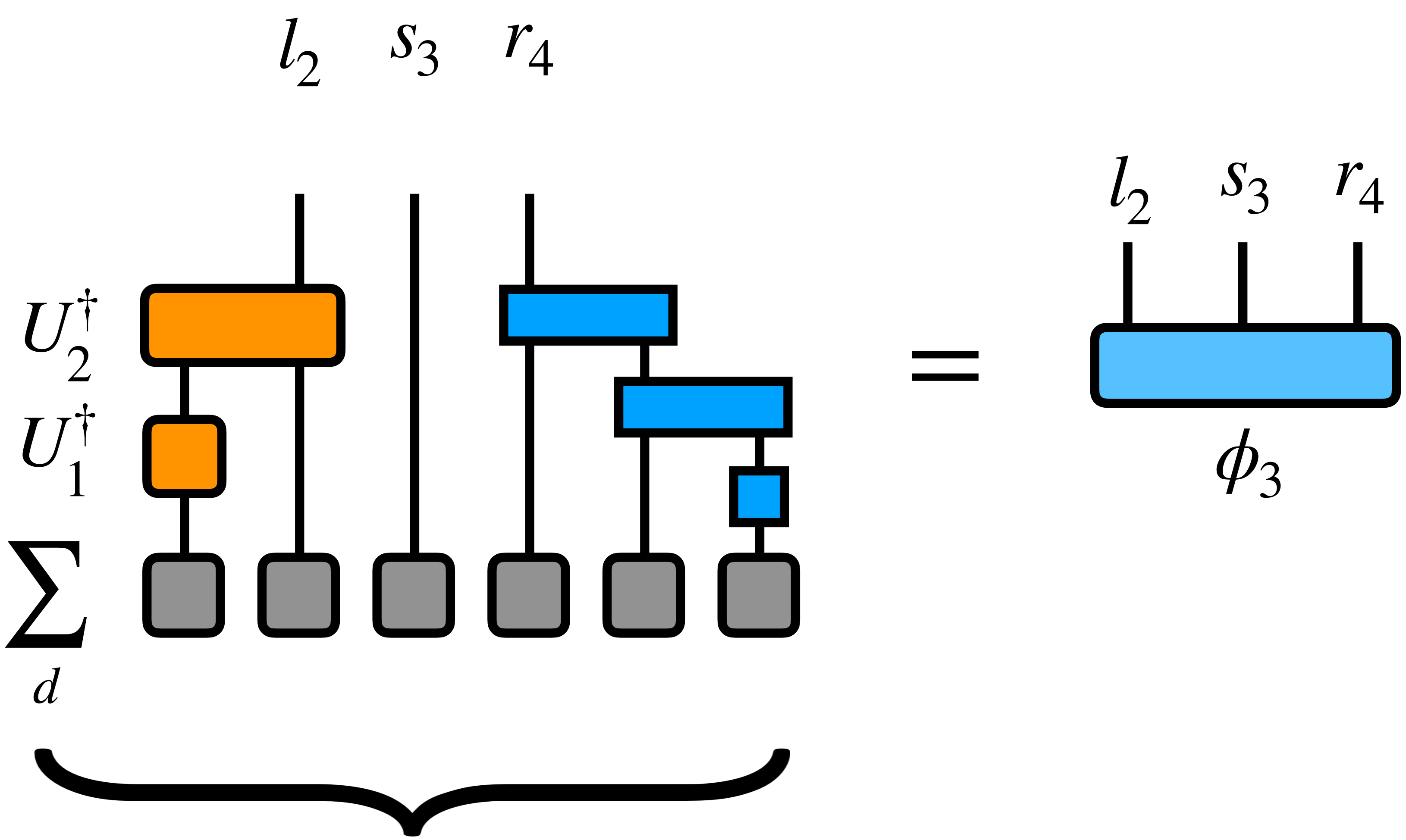
SVD of reduced data tensor gives second MPS tensor U_2



Save U_2 only

Tensor Network Machine Learning

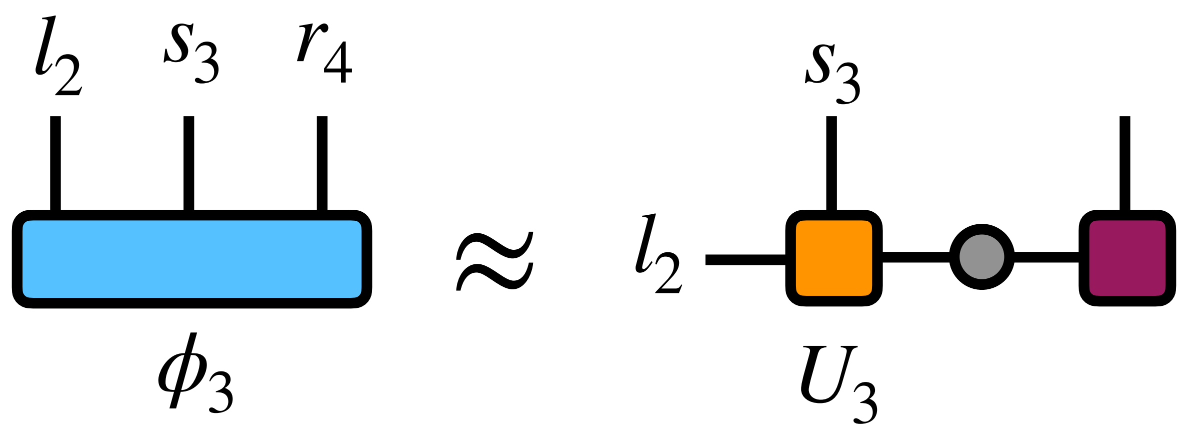
Recursively summing and sketching data continues process



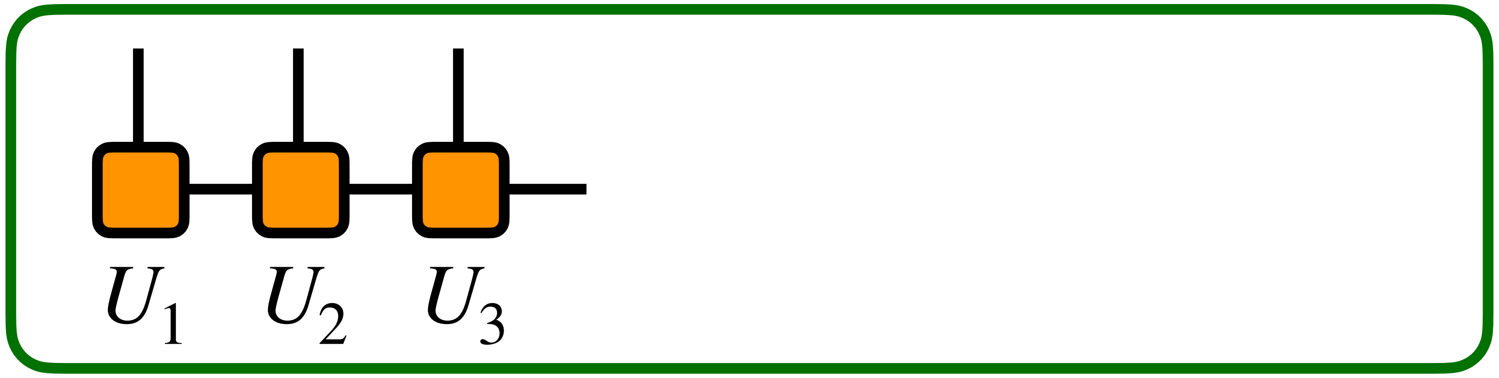
$\hat{p}$, sum over all data in training set (product states / rank-1 tensors)

Tensor Network Machine Learning

SVD of reduced data tensor gives second MPS tensor U_3



Save U_2 only



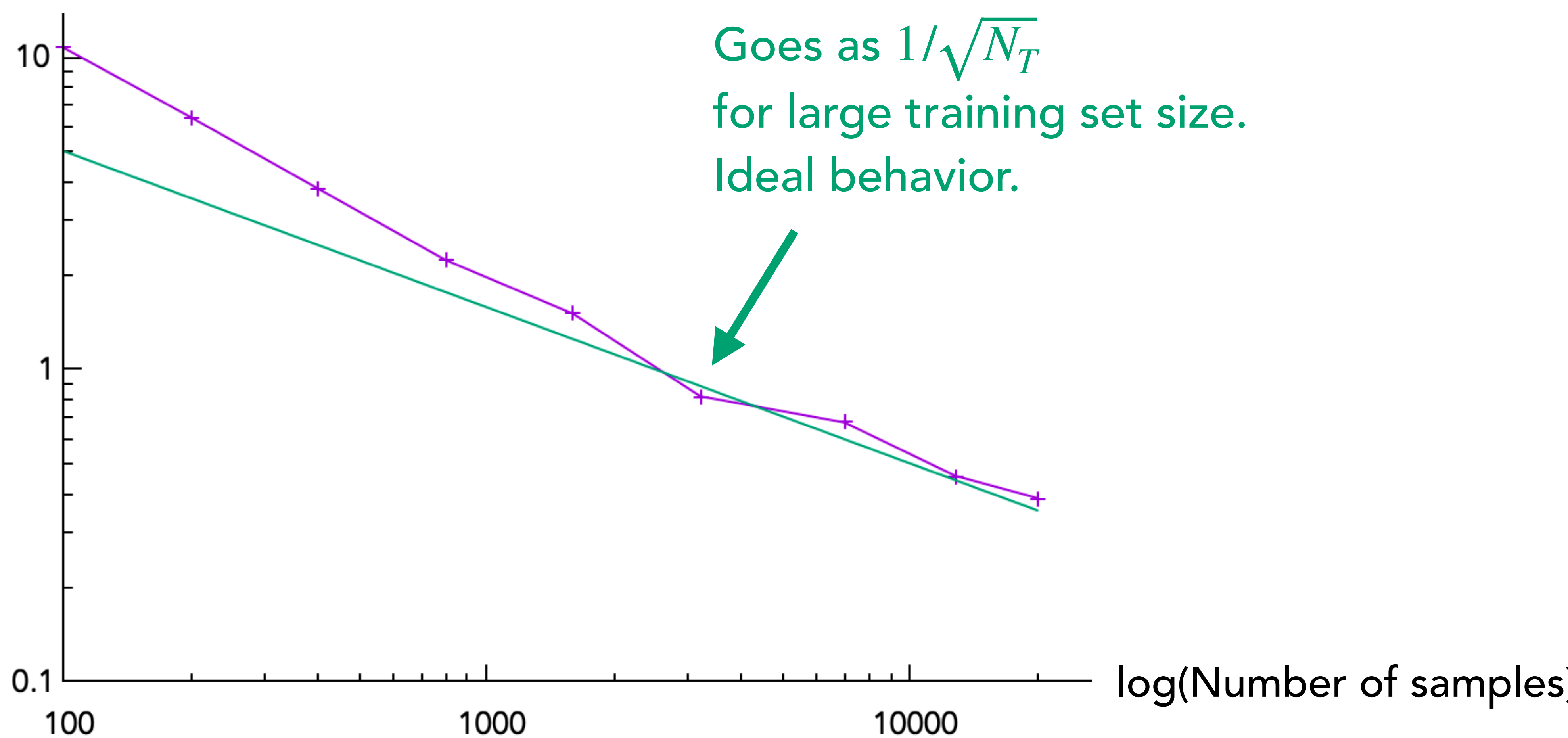
Tensor Network Machine Learning

Recursive sketching able to reconstruct distribution of ***disordered*** Ising chain

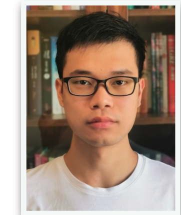
Distance of estimate from true distribution

$$\frac{\| p - A(\hat{p}) \|}{\| p \|}$$

log scale

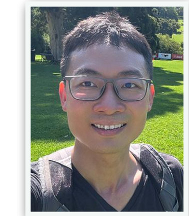


Tensor Network Machine Learning



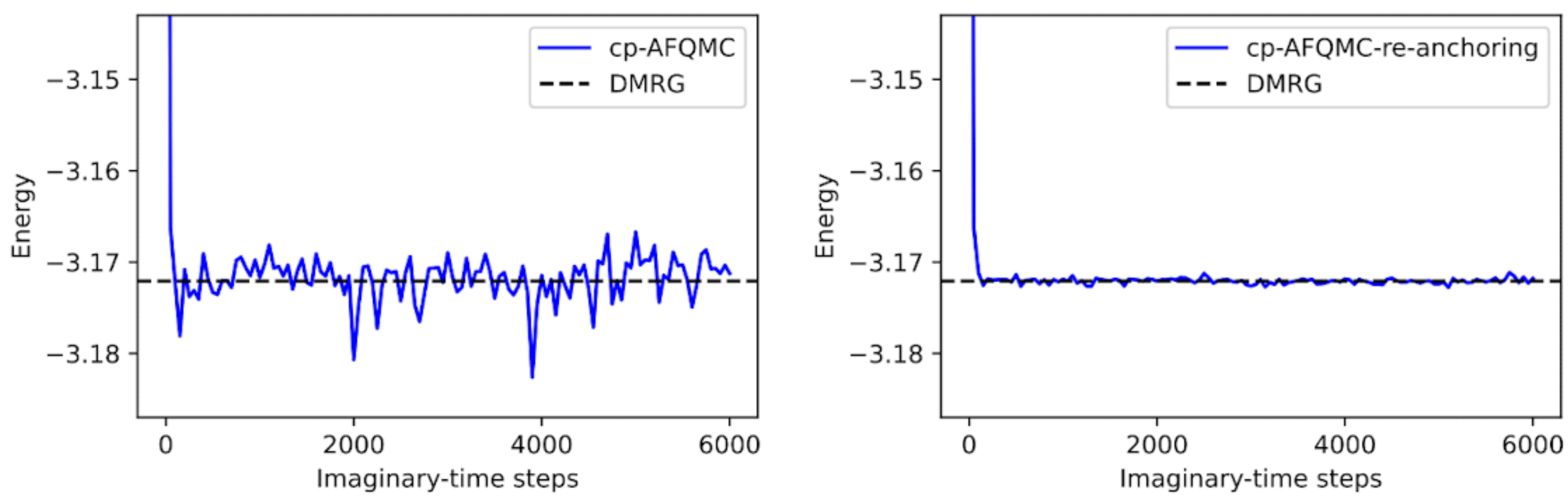
Ziang Yu

Shiwei Zhang



Yuehaw Khoo

Recursive sketching recently used to **learn trial wavefunctions** for AF-Quantum Monte Carlo



(a) Vanilla cp-AFQMC (b) cp-AFQMC-re-anchoring

FIGURE 6. Energy convergence of (a) vanilla cp-AFQMC and (b) cp-AFQMC-re-anchoring for 11×11 system with open boundary at $g = 3$. The reference

"Re-anchoring Quantum Monte Carlo with Tensor-Train Sketching", Yu, Zhang, Khoo, arxiv:2411.07194

Summary

Tensor networks can represent many types of functions, including functions of continuous variables

Powerful, interesting, and interpretable algorithms to learn high-dimensional functions as tensor networks

Up Next

After a break, we will discuss methods for 2D and 3D tensor networks