From Matrix Product States
and Dynamical Mean-Field Theory
to Machine Learning

Sommerfeld Theory Colloquium, LMU Munich
November 9, 2016

F. Alexander Wolf | falexwolf.de
Institute of Computational Biology
Helmholtz Zentrum München
Outline

• Matrix Product States / Tensor Trains

• Dynamical Mean-Field Theory

• Machine Learning
“Tensor Trains I”: noninteracting bits

\[ X_1 \quad X_2 \quad X_3 \quad X_4 \quad \ldots \]

Vector of random variables \( \mathbf{X} \in \{0, 1\}^L \) with joint probability mass

\[
p(\mathbf{x}) = \frac{1}{Z} e^{-H(\mathbf{x})/T}, \quad H(\mathbf{x}) = \sum_{n=1}^{L} x_n
\]

normalized with \( Z = \sum_{\mathbf{x}} e^{-H(\mathbf{x})/T} \).
“Tensor Trains I”: noninteracting bits

Vector of random variables $\mathbf{X} \in \{0, 1\}^L$ with joint probability mass

$$p_{\mathbf{x}} = \frac{1}{Z} e^{-H(\mathbf{x})/T}, \quad H(\mathbf{x}) = \sum_{n=1}^{L} x_n$$

normalized with $Z = \sum_{\mathbf{x}} e^{-H(\mathbf{x})/T}$.

$p$ has $2^L$ components $\mathbf{x} \in \{(0, 0, \ldots, 0), (0, 0, \ldots, 1), \ldots\}$. 
“Tensor Trains I”: noninteracting bits

Vector of random variables $\mathbf{X} \in \{0, 1\}^L$ with joint probability mass

$$p_x = \frac{1}{Z} e^{-H(x)/T}, \quad H(x) = \sum_{n=1}^{L} x_n$$

normalized with $Z = \sum_x e^{-H(x)/T}$.

$p$ has $2^L$ components $\mathbf{x} \in \{(0, 0, \ldots, 0), (0, 0, \ldots, 1), \ldots \}$.

Note $2^{100} \approx 10^{30} \approx 10^{15}$ TB.
“Tensor Trains I”: noninteracting bits

\[ X_1 \quad X_2 \quad X_3 \quad X_4 \quad \ldots \]

Compute correlations via \( \text{cov}(X_n, X_m) = \langle X_n X_m \rangle - \langle X_n \rangle \langle X_n \rangle \),

\[ \langle X_n X_m \rangle = \sum_x x_n x_m p_x. \]
“Tensor Trains I”: noninteracting bits

\[
\begin{array}{cccc}
X_1 & X_2 & X_3 & X_4 \\
\end{array}
\]

... 

Compute correlations via \( \text{cov}(X_n, X_m) = \langle X_n X_m \rangle - \langle X_n \rangle \langle X_n \rangle \),

\[
\langle X_n X_m \rangle = \sum_x x_n x_m p_x .
\]

▷ Naive brute force: \( 2^L \) operations necessary.

▷ Monte Carlo: sampling in space of \( 2^L \) states.
“Tensor Trains I”: noninteracting bits

\[ X_1 \quad X_2 \quad X_3 \quad X_4 \quad \ldots \]

**Better:** *independent* degrees of freedom \( X_n \) imply *separability*

\[
p_x = p_{x_1, x_2, \ldots, x_L} = \frac{1}{Z} e^{-\sum_{n=1}^{L} x_n/T} = \frac{1}{Z} a_{x_1} a_{x_2} \ldots a_{x_L}, \quad a_{x_n} = e^{-x_n/T}.
\]
“Tensor Trains I”: noninteracting bits

\[ X_1 \quad X_2 \quad X_3 \quad X_4 \quad ... \]

**Better:** independent degrees of freedom \( X_n \) imply *separability*

\[
p_x = p_{x_1, x_2, \ldots, x_L} = \frac{1}{Z} e^{-\sum_{n=1}^{L} x_n/T} = \frac{1}{Z} a_{x_1} a_{x_2} \ldots a_{x_L}, \quad a_{x_n} = e^{-x_n/T}.
\]

Compute correlations in \( 2L \) operations . . .

\[
\langle X_n X_m \rangle = \frac{1}{Z} \left( \sum_{x_n} x_n a_{x_n} \right) \left( \sum_{x_m} x_m a_{x_m} \right) \prod_{k \neq n, m}^{L} \left( \sum_{x_k} a_{x_k} \right) = \langle X_n \rangle \langle X_m \rangle \quad \ldots \quad \text{there are none.}
\]
“Tensor Trains II”: interacting bits (Ising model)

\[ \tilde{p}_x = \frac{1}{Z} e^{-H(x)/T}, \quad H(x) = -\sum_{n=1}^{L-1} x_n x_{n+1}. \]
“Tensor Trains II”: interacting bits (Ising model)

Two-body interactions imply “almost – separability”

\[ Z \sum_x \tilde{p}_x = \sum_x e^{x_1 x_2 / T} e^{x_2 x_3 / T} \ldots \]
“Tensor Trains II”: interacting bits (Ising model)

Two-body interactions imply “almost – separability”

\[ Z \sum_x \tilde{p}_x = \sum_x A_{x_1 x_2} A_{x_2, x_3} \ldots \]
“Tensor Trains II”: interacting bits (Ising model)

Two-body interactions imply “almost – separability”

\[
Z \sum_x \tilde{p}_x = \sum_x A_{x_1 x_2} A_{x_2, x_3} \ldots = \text{gsum} A A \ldots, \quad A_{x_n x_{n+1}} = e^{x_n x_{n+1}/T}, \quad A \in \mathbb{R}^{2 \times 2},
\]

where gsum is the grand sum.
"Tensor Trains II": interacting bits (Ising model)

Two-body interactions imply “almost – separability”

\[
Z \sum_x \tilde{p}_x = \sum_x A_{x_1 x_2} A_{x_2, x_3} \ldots \\
= \text{gsum} AA \ldots, \quad A_{x_n x_{n+1}} = e^{x_n x_{n+1}/T}, \quad A \in \mathbb{R}^{2 \times 2},
\]

where gsum is the grand sum.

▷ Compare to non-interacting case

\[
Z \sum_x p_x = \sum_x a_{x_1} a_{x_2} \ldots, \quad a_{x_n} = e^{-x_n/T}, \quad a \in \mathbb{R}^2.
\]
“Tensor Trains II”: interacting bits (Ising model)

\[ X_1 \quad \cdots \quad X_3 \quad \cdots \quad X_4 \quad \cdots \]

Compute correlations in \(2^3 L\) operations (\(L\) matrix products)

\[ \langle X_n X_m \rangle_\tilde{p} = \frac{1}{Z} \text{gsum} \prod_{k=1}^{n-1} (A[k]) M \prod_{k=n}^{m-1} (A[k]) M \prod_{k=m}^{L-1} (A[k]) \]

where

\[ M = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \]
“Tensor Trains II”: interacting bits (Ising model)

Compute correlations in $2^3 L$ operations ($L$ matrix products)

$$\langle X_n X_m \rangle_p \sim \frac{1}{\mathcal{Z}} \text{gsum} \prod_{k=1}^{n-1} (A[k]) M \prod_{k=n}^{m-1} (A[k]) M \prod_{k=m}^{L-1} (A[k])$$

where $M = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$

▷ Compare to non-interacting case ($2L$ operations)

$$\langle X_n X_m \rangle_p = \frac{1}{\mathcal{Z}} \left( \sum_{x_n} x_n A x_n \right) \left( \sum_{x_m} x_m A x_m \right) \prod_{k \neq n,m} \left( \sum_{x_k} A x_k \right)$$
“Tensor Trains III”: long-range interacting bit chain

\[ p\mathbf{x} = \frac{1}{Z} e^{-H(\mathbf{x})/T}, \quad H(\mathbf{x}) = -\sum_{n=1}^{L-2} x_n x_{n+1} x_{n+2} \]
“Tensor Trains III”: long-range interacting bit chain

\[ p_x = \frac{1}{Z} e^{-H(x)/T}, \quad H(x) = -\sum_{n=1}^{L-2} x_n x_{n+1} x_{n+2} \]

\[ Z \sum_x p_x = \sum_x \prod_{n=1}^{L-2} A_{x_n x_{n+1} x_{n+2}} \quad A_{x_n x_{n+1} x_{n+2}} = e^{x_n x_{n+1} x_{n+2}/T} \]

\[ A \in \mathbb{R}^{2 \times 2 \times 2} \]
"Tensor Trains III": long-range interacting bit chain  

\[
p_x = \frac{1}{Z} e^{-H(x)/T}, \quad H(x) = -\sum_{n=1}^{L-2} x_n x_{n+1} x_{n+2}
\]

\[
Z \sum_x p_x = \sum_x \prod_{n=1}^{L-2} A_{x_n x_{n+1} x_{n+2}}
\]

\[
\begin{align*}
A_{x_n x_{n+1} x_{n+2}} &= e^{x_n x_{n+1} x_{n+2}/T} \\
A &\in \mathbb{R}^{2 \times 2 \times 2} \\
B_{x'_n (2x_{n+1} + x_{n+2})} &= A_{x_n x_{n+1} x_{n+2}} \\
B &\in \mathbb{R}^{2 \times 4}
\end{align*}
\]
“Tensor Trains III”: long-range interacting bit chain  

\[ p_x = \frac{1}{Z} e^{-H(x)/T}, \quad H(x) = -\sum_{n=1}^{L-2} x_n x_{n+1} x_{n+2} \]

\[ Z \sum_x p_x = \sum_x \prod_{n=1}^{L-2} A_{x_n x_{n+1} x_{n+2}} \]

\[ = \sum_{x'} \prod_{n=1}^{L-2} B_{x'_n x'_{n+1}} B^t_{x'_{n+1} x'_{n+2}} \]

\[ A x_n x_{n+1} x_{n+2} = e^{x_n x_{n+1} x_{n+2}/T} \]

\[ A \in \mathbb{R}^{2 \times 2 \times 2} \]

\[ B x'_n (2x_{n+1} + x_{n+2}) = A x_n x_{n+1} x_{n+2} \]

\[ B \in \mathbb{R}^{2 \times 4} \]

Tensor Train format \( \gg \frac{1}{2} (2^3 + 4^3) L \) operations
“Tensor Trains” in Statistical Mechanics

• Write probability mass function

\[ p : \{0, 1, \ldots, d\}^L \rightarrow \mathbb{R}, \quad d, L \in \mathbb{N}, \]

as vector

\[ p_x = p(x), \quad p \in \mathbb{R}^{d^L}, \]

which is indexed and parametrized by \( x \in \{0, 1, \ldots, d\}^L \).
“Tensor Trains” in Statistical Mechanics

• Write probability mass function

\[ p : \{0, 1, \ldots, d\}^L \rightarrow \mathbb{R}, \quad d, L \in \mathbb{N}, \]

as vector

\[ p_x = p(x), \quad p \in \mathbb{R}^{d^L}, \]

which is indexed and parametrized by \( x \in \{0, 1, \ldots, d\}^L \).

If \( p_x = p(x) \) does not couple all index components \( x_n \) among each other, there is a low rank Tensor Train representation.
“Tensor Trains” in Statistical Mechanics

- Write probability mass function

\[ p : \{0, 1, \ldots, d\}^L \to \mathbb{R}, \quad d, L \in \mathbb{N}, \]

as vector

\[ p_x = p(x), \quad p \in \mathbb{R}^{d^L}, \]

which is indexed and parametrized by \( x \in \{0, 1, \ldots, d\}^L \).

If \( p_x = p(x) \) does not couple all index components \( x_n \) among each other, there is a low rank Tensor Train representation.

This reduces computational cost in summations over \( p(x) \) from exponential to linear in system size.
“Tensor Trains” in Statistical Mechanics

• Write probability mass function

\[ p : \{0, 1, \ldots, d\}^L \to \mathbb{R}, \quad d, L \in \mathbb{N}, \]

as vector

\[ p_x = p(x), \quad p \in \mathbb{R}^{d^L}, \]

which is indexed and parametrized by \( x \in \{0, 1, \ldots, d\}^L \).

If \( p_x = p(x) \) does not couple all index components \( x_n \) among each other, there is a low rank Tensor Train representation.

This reduces computational cost in summations over \( p(x) \) from exponential to linear in system size.

• What about quantum mechanics?
Statistical Mechanics vs. Quantum Mechanics

Instead of considering sums over classical weights, as in the partition sum,

\[ 1 = \sum_x p_x = \sum_x \langle x | \hat{p}_x | x \rangle, \]

where we used a somewhat exaggerated notation.

We now consider quantum many-body states

\[ | \psi \rangle = \sum_x c_x | x \rangle, \]

where \[ | x \rangle = \prod_i | x_i \rangle \] is a tensor product of single-particle basis states \[ | x_i \rangle \]. For example \[ | x_i \rangle \in \{ | \uparrow \rangle, | \downarrow \rangle \} \].

But, do we know anything about how the vector of coefficients \[ c = (c_x) \] couples its components, so that the tensor train format is meaningful?
Statistical Mechanics vs. Quantum Mechanics

Instead of considering sums over classical weights, as in the partition sum,

\[ 1 = \sum_x p_x = \sum_x \langle x | \hat{p}_x | x \rangle, \]

where we used a somewhat exaggerated notation. We now consider quantum many-body states

\[ |\psi\rangle = \sum_x c_x |x\rangle, \]

where \(|x\rangle = |x_1\rangle \otimes |x_2\rangle \otimes \cdots \otimes |x_L\rangle = |x_1x_2\ldots x_L\rangle\) is a tensor product of single-particle basis states \(|x_i\rangle\). For example

\[ |x_i\rangle \in \{|\uparrow_i\rangle, |\downarrow_i\rangle\} \]
Statistical Mechanics vs. Quantum Mechanics

Instead of considering sums over classical weights, as in the partition sum,

\[ 1 = \sum_x p_x = \sum_x \langle x|\hat{p}_x|x\rangle, \]

where we used a somewhat exaggerated notation. We now consider quantum many-body states

\[ |\psi\rangle = \sum_x c_x |x\rangle, \]

where \( |x\rangle = |x_1\rangle \otimes |x_2\rangle \otimes \cdots \otimes |x_L\rangle = |x_1 x_2 \ldots x_L\rangle \) is a tensor product of single-particle basis states \( |x_i\rangle \). For example

\[ |x_i\rangle \in \{|\uparrow_i\rangle, |\downarrow_i\rangle\} \]

- But, do we know anything about how the vector of coefficients \( c = (c_x) \) couples its components, so that the tensor train format is meaningful?
For now we don’t have to. Simply try an ansatz!
For now we don’t have to. Simply try an ansatz!

- We can e.g. simply do a mean-field theory! Let us assume

\[ c_x = a^{x_1} a^{x_2} \ldots a^{x_L} = \prod_i a^{x_i} \]

then state can be manipulated doing \( \sim L \) operations

\[
|\psi\rangle = \sum_x c_x |x\rangle = |\psi_{\text{MF}}\rangle = \sum_x \prod_i a^{x_i} |x\rangle = \prod_i \left( \sum_{x_i} a^{x_i} |x_i\rangle \right)
\]
For now we don’t have to. Simply try an ansatz!

- We can e.g. simply do a mean-field theory! Let us assume

\[ c_x \dagger = a^{x_1} a^{x_2} \ldots a^{x_L} = \prod_i a^{x_i} \]

then state can be manipulated doing \( \sim L \) operations

\[ |\psi\rangle = \sum_x c_x |x\rangle = |\psi_{\text{MF}}\rangle = \sum_x \prod_i a^{x_i} |x\rangle = \bigotimes_i \left( \sum_{x_i} a^{x_i} |x_i\rangle \right) \]

- How to determine the factors \( a^{x_i} \)? Variationally solve

\[ \partial_{a^{x_i}} \frac{\langle \psi_{\text{MF}} | H | \psi_{\text{MF}} \rangle}{\langle \psi_{\text{MF}} | \psi_{\text{MF}} \rangle} = 0. \]

- Approximation to ground state. Approximation is *good* if ground state is in the same *class* of states as the ansatz \( |\psi_{\text{MF}}\rangle \).

- Relax mean-field assumption for coefficients of many body states

\[
c_x = a^{x_1} a^{x_2} a^{x_3} \ldots a^{x_L} = \prod_i a^{x_i}
\]

to one that factorizes in matrices

\[
c_x = \sum_{\{\nu_i\}} A_{\nu_1}^{x_1} A_{\nu_2}^{x_2} A_{\nu_3}^{x_3} \ldots A_{\nu_L}^{x_L} = \prod_i A^{x_i}
\]

- Relax mean-field assumption for coefficients of many body states

\[ c_x = a^{x_1} a^{x_2} a^{x_3} \ldots a^{x_L} = \prod_i a^{x_i} \]

to one that factorizes in matrices

\[ c_x = \sum_{\{\nu_i\}} A_{\nu_1}^{x_1} A_{\nu_2}^{x_2} A_{\nu_3}^{x_3} \ldots A_{\nu_L}^{x_L} = \prod_i A^{x_i} \]

- An MPS can be manipulated with costs of \( Lm^3 \), where \( m \) is the dimension of the matrices \( A^{x_i} \)

\[ |\psi\rangle = \sum_x c_x |x\rangle = |\psi_{\text{MPS}}\rangle = \sum_x \prod_i A^{x_i} |x\rangle \]
Tensor Trains IV: Matrix Product States  

• Relax mean-field assumption for coefficients of many body states

\[ c_\mathbf{x} \equiv a^{x_1} a^{x_2} a^{x_3} \ldots a^{x_L} = \prod_i a^{x_i} \]

to one that factorizes in matrices

\[ c_\mathbf{x} \equiv \sum_{\{\nu_i\}} A^{x_1}_{\nu_1} A^{x_2}_{\nu_1\nu_2} A^{x_3}_{\nu_2\nu_3} \ldots A^{x_L}_{\nu_L} = \prod_i A^{x_i} \]

• An MPS can be manipulated with costs of \( Lm^3 \), where \( m \) is the dimension of the matrices \( A^{x_i} \)

\[ |\psi\rangle = \sum_{\mathbf{x}} c_\mathbf{x} |\mathbf{x}\rangle = |\psi_{\text{MPS}}\rangle = \sum_{\mathbf{x}} \prod_i A^{x_i} |\mathbf{x}\rangle \]

• Are ground states in the same class as MPS? Which is this class? Are the coefficients \( c_\mathbf{x} \) in ground states weakly coupled?
Gapped Hamiltonians with short range interactions.

- Physical correlations have a finite range.
- Entanglement fulfills **area law**: entanglement of a region $A$ is proportional to surface $|\partial A|$, not volume $|A|$, of this region.
Gapped Hamiltonians with short range interactions.

- Physical correlations have a finite range.
- Entanglement fulfills **area law**: entanglement of a region $A$ is proportional to surface $|\partial A|$, not volume $|A|$, of this region.

▷ There is a low-rank Tensor Train representation!
Dynamical Mean-Field Theory
Quantum Embedding

- Dynamical Mean-Field Theory  
  Metzner & Vollhardt (1989)  
  Georges & Kotliar (1992)

- Density Matrix Embedding Theory  
  Knizia & Chan, PRL 109, 186404 (2012)
Dynamical Mean-Field Theory

1. Find function $\Lambda(\omega)$ that describes the bath.
2. Solve the reduced cluster problem.

- $\Delta$ Use Tensor Trains to represent the wave function of the cluster.
1. Find function \( \Lambda(\omega) \) that describes the bath.
1. Find function $\Lambda(\omega)$ that describes the bath.
2. Solve the reduced cluster problem.
Dynamical Mean-Field Theory

1. Find function $\Lambda(\omega)$ that describes the bath.
2. Solve the reduced cluster problem.

▷ Use Tensor Trains to represent the wave function of the cluster.
Tensor Trains and Dynamical Mean-Field Theory

Tensor Trains ~ Density Matrix Renormalization Group (DMRG)

Algorithmic approaches

- Lanczos: unstable and imprecise

  García, Hallberg & Rozenberg, PRL 93, 246403 (2004)
Tensor Trains and Dynamical Mean-Field Theory
Tensor Trains $\sim$ Density Matrix Renormalization Group (DMRG)

Algorithmic approaches

- Lanczos: unstable and imprecise
  
  García, Hallberg & Rozenberg, PRL 93, 246403 (2004)

- dynamic (correction vector) DMRG: extremely expensive
  
  
  Karski, Raas & Uhrig, PRB 72, 113110 (2005)  
  Karski, Raas & Uhrig, PRB 77, 075116 (2008)

- Chebyshev and Fourier expansions: cheaper and precise
  
  Ganahl, Thunström, Verstraete, Held & Evertz, PRB 90, 045144 (2014)
  
  Wolf, McCulloch, Parcollet & Schollwöck, PRB 90, 115124 (2014)
  
  Wolf, McCulloch & Schollwöck, PRB 90, 235131 (2014)

- Imaginary axis: again cheaper!
  
  Wolf, Go, McCulloch, Millis & Schollwöck, PRX 5, 041032 (2015)
  
  2-site cluster for 3-band model!
Tensor Trains and Dynamical Mean-Field Theory

Tensor Trains ~ Density Matrix Renormalization Group (DMRG)

Algorithmic approaches

- **Lanczos**: unstable and imprecise
  
  García, Hallberg & Rozenberg, PRL 93, 246403 (2004)

- **dynamic (correction vector) DMRG**: extremely expensive
  
  

- **Chebyshev and Fourier expansions**: cheaper and precise
  
  Ganahl, Thunström, Verstraete, Held & Evertz, PRB 90, 045144 (2014)
  
  Wolf, McCulloch, Parcollet & Schollwöck, PRB 90, 115124 (2014a)
  
  Wolf, McCulloch & Schollwöck, PRB 90, 235131 (2014b)
  
  Wolf, Justiniano, McCulloch & Schollwöck, PRB 91, 115144 (2015b)
  
  de Vega, Schollwöck & Wolf, PRB 92, 155126 (2015)

  ▶ 2-site cluster!
  
  ▶ entanglement and non-EQ!
  
  ▶ relation Chebyshev/ Fourier!
  
  ▶ bath discretization!
Tensor Trains and Dynamical Mean-Field Theory

Tensor Trains \sim \text{Density Matrix Renormalization Group (DMRG)}

Algorithmic approaches

- Lanczos: unstable and imprecise
  
  García, Hallberg & Rozenberg, PRL 93, 246403 (2004)

- dynamic (correction vector) DMRG: extremely expensive
  
  

- Chebyshev and Fourier expansions: cheaper and precise
  
  Ganahl, Thunström, Verstraete, Held & Evertz, PRB 90, 045144 (2014)
  
  Wolf, McCulloch, Parcollet & Schollwöck, PRB 90, 115124 (2014a)
  
  Wolf, McCulloch & Schollwöck, PRB 90, 235131 (2014b)
  
  Wolf, Justiniano, McCulloch & Schollwöck, PRB 91, 115144 (2015b)
  
  de Vega, Schollwöck & Wolf, PRB 92, 155126 (2015)

  ▶ 2-site cluster!
  
  ▶ entanglement and non-EQ!
  
  ▶ relation Chebyshev/ Fourier!
  
  ▶ bath discretization!

- Imaginary axis: again cheaper!
  
  Wolf, Go, McCulloch, Millis & Schollwöck, PRX 5, 041032 (2015a) ▶ 2-site cluster for 3-band model!
Tensor Trains and Dynamical Mean-Field Theory

Tensor Trains ∼ Density Matrix Renormalization Group (DMRG)

Applications

- Non-thermal melting of Neel order in the Hubbard model
  
  Balzer, Wolf, McCulloch, Werner & Eckstein, PRX 5, 031039 (2015)
Tensor Trains and Dynamical Mean-Field Theory
Tensor Trains ~ Density Matrix Renormalization Group (DMRG)

Applications

- Non-thermal melting of Neel order in the Hubbard model
  Balzer, Wolf, McCulloch, Werner & Eckstein, PRX 5, 031039 (2015)

- Benchmark quantum computing protocols
  Bauer, Wecker, Millis, Hastings & Troyer, PRX 6, 031045 (2016)
Tensor Trains and Dynamical Mean-Field Theory

Tensor Trains $\sim$ Density Matrix Renormalization Group (DMRG)

Applications

- Non-thermal melting of Neel order in the Hubbard model
  
  Balzer, Wolf, McCulloch, Werner & Eckstein, PRX 5, 031039 (2015)

- Benchmark quantum computing protocols
  
  Bauer, Wecker, Millis, Hastings & Troyer, PRX 6, 031045 (2016)
  

- In general: situations not treatable by QMC and NRG, which can be
  
  ○ correlated materials  Linden et al., in progress (2016)
  ○ gauge fields and topological phases
Machine Learning
Machine Learning

Estimate noisy functional relation

\[ f : \mathcal{X} \rightarrow \mathcal{Y}, \quad Y = f(X) + N, \]

from data \( \mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n_{\text{samples}}} \).
Machine Learning

Estimate noisy functional relation

\[ f : \mathcal{X} \rightarrow \mathcal{Y}, \quad Y = f(X) + N, \]

from data \( D = \{(x_i, y_i)\}_{i=1}^{n_{\text{samples}}} \).

- \( f : \mathbb{R}^{28 \times 28} \rightarrow \{2, 4\} \).

Stoudenmire & Schwab, NIPS (2016)
Machine Learning

Estimate noisy functional relation

\[ f : \mathcal{X} \rightarrow \mathcal{Y}, \quad Y = f(X) + N, \]

from data \( D = \{(x_i, y_i)\}_{i=1}^{n_{\text{samples}}} \).

- \( f : \mathbb{R}^{28 \times 28} \rightarrow \{2, 4\} \).

Stoudenmire & Schwab, NIPS (2016)

- Linear regression using Gaussian noise model

\[
p(y|x, \theta = (w, \sigma^2)) = \mathcal{N}(y|w_1 x + w_0, \sigma^2)
\]
Machine Learning

Estimate noisy functional relation

\[ f : \mathcal{X} \rightarrow \mathcal{Y}, \quad Y = f(X) + N, \]

from data \( \mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n_{\text{samples}}} \).

- \( f : \mathbb{R}^{28 \times 28} \rightarrow \{2, 4\}. \)

Stoudenmire & Schwab, NIPS (2016)

- Linear regression using Gaussian noise model

\[
p(y|x, \theta = (w, \sigma^2)) = \mathcal{N}(y|w_1x + w_0, \sigma^2)
\]

Estimate parameters?

\[
\theta^* = \arg\max_{\theta} p(\theta|\mathcal{D}, \text{model, prior beliefs})
\]
Machine Learning

Estimate noisy functional relation

\[ f : \mathcal{X} \rightarrow \mathcal{Y}, \quad Y = f(X) + N, \]

from data \( D = \{(x_i, y_i)\}_{i=1}^{n_{\text{samples}}} \).

- \( f : \mathbb{R}^{28 \times 28} \rightarrow \{2, 4\} \).

Stoudenmire & Schwab, NIPS (2016)

- Linear regression using Gaussian noise model

\[
p(y|x, \theta = (w, \sigma^2)) = \mathcal{N}(y|w_1 x + w_0, \sigma^2)
\]

Estimate parameters?

\[
\theta^* = \text{argmax}_\theta p(\theta|D, \text{model, prior beliefs})
\]

\( \triangleright \) Integrate and optimize a high-dimensional distribution.
Graphical Models

Ising Model

Here, the distribution itself factorizes!
Graphical Models

Ising Model

\[ p(x_n) = \sum_{\{x_n' | n' \neq n\}} p(x_1, \ldots, x_{n_{\text{max}}}) \]

\[ = \sum_{\{x_n' | n' \neq n\}} A_{x_1} A_{x_2} A_{x_3} \cdots A_{x_{n_{\text{max}}-1}} x_{n_{\text{max}}} \]
Graphical Models

Ising Model

\[ p(x_n) = \sum_{\{x_{n'}|n' \neq n\}} p(x_1, \ldots, x_{n_{\text{max}}}) \]

\[ = \sum_{\{x_{n'}|n' \neq n\}} A_{x_1 x_2} A_{x_2 x_3} \cdots A_{x_{n_{\text{max}}-1} x_{n_{\text{max}}}} \]

Markov Chain
Graphical Models

Ising Model

\[
p(x_n) = \sum_{\{x_{n'} | n' \neq n\}} p(x_1, \ldots, x_{n_{\text{max}}})
\]

\[
= \sum_{\{x_{n'} | n' \neq n\}} A_{x_1 x_2} A_{x_2 x_3} \cdots A_{x_{n_{\text{max}}-1} x_{n_{\text{max}}}}
\]

Markov Chain

\[
p(x_n) = \sum_{\{x_{n'} | n' \neq n\}} p(x_1, \ldots, x_{n_{\text{max}}})
\]

\[
= \sum_{x_{n-1}} A_{x_n x_{n-1}} p(x_{n-1})
\]

Here, the distribution itself factorizes!
Directed Acyclic Graphs

Markov chain

\[ p(x_1, \ldots, x_{n_{\text{max}}}) = p(x_1) \prod_{n=1}^{n_{\text{max}}-1} p(x_{n+1}|x_n) \]
Directed Acyclic Graphs

Markov chain

\[ p(x_1, \ldots, x_{n_{\text{max}}}) = p(x_1) \prod_{n=1}^{n_{\text{max}}-1} p(x_{n+1}|x_n) \]

General graph

\[ p(x_1, \ldots, x_{n_{\text{max}}}) = \prod_{n=1}^{n_{\text{max}}} p(x_n|\text{pa}(x_n)) \]

Example: 
\[ X_1 = \text{yellow teeth}, \quad X_2 = \text{smoke}, \quad Y = \text{cancer}, \quad X_3 = \text{diet}. \]

\[ X_1 \rightarrow X_2 \rightarrow Y \rightarrow X_3 \]
Directed Acyclic Graphs

Markov chain

\[ p(x_1, \ldots, x_{n_{\text{max}}}) = p(x_1) \prod_{n=1}^{n_{\text{max}}-1} p(x_{n+1}|x_n) \]

General graph

\[ p(x_1, \ldots, x_{n_{\text{max}}}) = \prod_{n=1}^{n_{\text{max}}} p(x_n|\text{pa}(x_n)) \]

Example: \( X_1 = \text{yellow teeth}, \ X_2 = \text{smoke}, \ Y = \text{cancer}, \ X_3 = \text{diet}. \)
Inferring gene regulation from single-cell data

- Infer causal structure of gene regulation.

Inferring gene regulation from single-cell data

- Infer causal structure of gene regulation.
- Given a high-dimensional stochastic process, infer couplings among variables.

Time series data
Consider a $d$-dimensional time series $(X_t)$, for example

\[
\begin{align*}
X_{(t-2)1} &\rightarrow X_{(t-1)1} \rightarrow X_{t1} \\
X_{(t-2)2} &\rightarrow X_{(t-1)2} \rightarrow X_{t2} \\
X_{(t-2)3} &\rightarrow X_{(t-1)3} \rightarrow X_{t3}
\end{align*}
\]
Time series data

Consider a $d$-dimensional time series $(X_t)$, for example

\[ X_{(t-2)1} \rightarrow X_{(t-1)1} \rightarrow X_{t1} \]
\[ X_{(t-2)2} \rightarrow X_{(t-1)2} \rightarrow X_{t2} \]
\[ X_{(t-2)3} \rightarrow X_{(t-1)3} \rightarrow X_{t3} \]

\[ X_{t1} = X_{(t-1)1} + N_{t1} \]
\[ X_{t2} = X_{(t-1)2} + N_{t2} \]
\[ X_{t3} = X_{(t-1)1} \land X_{(t-1)2} + N_{t3} \]
Time series data

Consider a \(d\)-dimensional time series \((X_t)\), for example

\[
X_{(t-2)1} \rightarrow X_{(t-1)1} \rightarrow X_{t1}
\]
\[
X_{(t-2)2} \rightarrow X_{(t-1)2} \rightarrow X_{t2}
\]
\[
X_{(t-2)3} \rightarrow X_{(t-1)3} \rightarrow X_{t3}
\]

\[
X_{t1} = X_{(t-1)1} + N_{t1}
\]
\[
X_{t2} = X_{(t-1)2} + N_{t2}
\]
\[
X_{t3} = X_{(t-1)1} \land X_{(t-1)2} + N_{t3}
\]

One approach is **Transfer Entropy**, which is conditional mutual information [Schreiber, PRL 85, 461 (2000)] (\(\sim\) Granger Causality [Granger, Econometrica 37, 424 (1969)])

\[
\text{TE}_{i \rightarrow j} = \text{MI}_{X_{(t-1)i}; X_{tj}|S}
\]
\[
= H_{X_{tj}|S} - H_{X_{tj}|X_{(t-1)i},S}
\]

where originally, \(S = X_{(t-1)j}\), and later \(S = \{\text{all observed variables}\}\).
Limitations of Transfer Entropy and Granger Causality

- Conditioning on all variables leads to terrible *curse of dimensionality*. 

\[ X_1, X_2 \sim \text{Ber}(0.5), \quad X_3 = X_1 + X_2. \] 

Then \( X_3 \not\perp \perp X_1 | X_3 \).

Granger Causality and Transfer Entropy yield information flow \( X(t-1) \to X_t \). But it's non-causal, i.e. non-physical!
Limitations of Transfer Entropy and Granger Causality

- Conditioning on all variables leads to terrible *curse of dimensionality*.
- Say $X_1, X_2 \sim \text{Ber}(0.5)$, and $X_3 = X_1 + X_2$. Then $X_3$

$$X_1 \not\perp\!
\!
\!
\perp X_2 \mid X_3.$$
Limitations of Transfer Entropy and Granger Causality

- Conditioning on all variables leads to terrible *curse of dimensionality*.
- Say $X_1, X_2 \sim \text{Ber}(0.5)$, and $X_3 = X_1 + X_2$. Then $X_3$

\[ X_1 \not\indep X_2 \mid X_3. \]

Granger Causality and Transfer Entropy yield information flow $X_{(t-1)1} \rightarrow X_{t2}$. But it’s non-causal, i.e. non-physical!
Limitations of Transfer Entropy and Granger Causality

- Conditioning on all variables leads to terrible *curse of dimensionality*.
- Say $X_1, X_2 \sim \text{Ber}(0.5)$, and $X_3 = X_1 + X_2$. Then $X_3$

\[ X_1 \not\perp\!\!\!\!\perp X_2 \mid X_3. \]

> Granger Causality and Transfer Entropy yield information flow $X_{(t-1)1} \rightarrow X_{t2}$. But it’s non-causal, i.e. non-physical!

\[
\begin{align*}
X_{(t-2)1} & \rightarrow X_{(t-1)1} \rightarrow X_{t1} \\
X_{(t-2)2} & \rightarrow X_{(t-1)2} \rightarrow X_{t2} \\
X_{(t-2)3} & \rightarrow X_{(t-1)3} \rightarrow X_{t3}
\end{align*}
\]
Limitations of Transfer Entropy and Granger Causality

- Conditioning on all variables leads to terrible *curse of dimensionality*.
- Say \( X_1, X_2 \sim \text{Ber}(0.5) \), and \( X_3 = X_1 + X_2 \). Then \( X_3 \)

\[
X_1 \not\perp\!
\!
\!
\not\!
\!
\perp X_2 | X_3.
\]

▷ Granger Causality and Transfer Entropy yield information flow \( X_{(t-1)1} \rightarrow X_{t2} \). But it’s non-causal, i.e. non-physical!

\[
\begin{align*}
X_{(t-2)1} & \rightarrow X_{(t-1)1} \rightarrow X_{t1} \\
X_{(t-2)2} & \rightarrow X_{(t-1)2} \rightarrow X_{t2} \\
X_{(t-2)3} & \rightarrow X_{(t-1)3} \rightarrow X_{t3}
\end{align*}
\]

▷ Need something different!
Systematic conditional independence tests

*Constraint based methods.*  
Pearl & Verma (1991)  
Spirtes, Glymour & Scheines (2000)

1. Start with a fully connected graph.

Systematic conditional independence tests


1. Start with a fully connected graph.
2. Reduce edges by conditional independence tests.

...
Systematic conditional independence tests


1. Start with a fully connected graph.
2. Reduce edges by conditional independence tests.
   SGS Test all combinations and conditions $X_i \perp X_j | S$.
3. Orient edges, where possible.

- Doesn’t work in gene expression time series as there is not enough dynamic noise.
- In addition to statistical association among variables, test for functional relation. Geometry of data plays role.
  Wolf & Theis, in preparation (2016)
Systematic conditional independence tests

*Constraint based methods.*  
Pearl & Verma (1991)  
Spirtes, Glymour & Scheines (2000)

1. Start with a fully connected graph.
2. Reduce edges by conditional independence tests.
   - SGS  Test all combinations and conditions $X_i \perp \!\!\!\!\!\perp X_j | S$.
   - PC(a) Test $X_i \perp \!\!\!\!\!\perp X_j | \emptyset$.
3. Orient edges, where possible.
   - Doesn’t work in gene expression time series as there is not enough dynamic noise.

In addition to statistical association among variables, test for functional relation.  
Geometry of data plays role.  
Wolf & Theis, in preparation (2016)
Systematic conditional independence tests


1. Start with a fully connected graph.

2. Reduce edges by conditional independence tests.
   - **SGS** Test all combinations and conditions $X_i \perp \perp X_j | S$.
   - **PC(a)** Test $X_i \perp \perp X_j | \emptyset$.
     - (b) On remaining edges and connected components, test $X_i \perp \perp X_j | X_k$. 

Systematic conditional independence tests


1. Start with a fully connected graph.

2. Reduce edges by conditional independence tests.
   - **SGS** Test all combinations and conditions $X_i \perp\!\!\!\!\!\!\perp X_j|S$.
   - **PC(a)** Test $X_i \perp\!\!\!\!\!\!\perp X_j|\emptyset$.
     - (b) On remaining edges and connected components, test $X_i \perp\!\!\!\!\!\!\perp X_j|X_k$.
     - (c) And so forth.
Systematic conditional independence tests


1. Start with a fully connected graph.

2. Reduce edges by conditional independence tests.
   - **SGS** Test all combinations and conditions $X_i \perp \vert X_j \vert S$.
   - **PC(a)** Test $X_i \perp \vert X_j \vert \emptyset$.
     - (b) On remaining edges and connected components, test $X_i \perp \vert X_j \vert X_k$.
     - (c) And so forth.

3. Orient edges, where possible.
Systematic conditional independence tests


1. Start with a fully connected graph.
2. Reduce edges by conditional independence tests.
   - **SGS** Test all combinations and conditions $X_i \perp \perp X_j | S$.
   - **PC(a)** Test $X_i \perp \perp X_j | \emptyset$.
     - (b) On remaining edges and connected components, test $X_i \perp \perp X_j | X_k$.
     - (c) And so forth.
3. Orient edges, where possible.

- Doesn’t work in gene expression time series as there is not enough dynamic noise.
Systematic conditional independence tests

*Constraint based methods.*  
Pearl & Verma (1991)  
Spirtes, Glymour & Scheines (2000)

1. Start with a fully connected graph.
2. Reduce edges by conditional independence tests.
   
   **SGS** Test all combinations and conditions $X_i \perp \perp X_j | S$.
   
   **PC(a)** Test $X_i \perp \perp X_j | \emptyset$.
   
   (b) On remaining edges and connected components, test $X_i \perp \perp X_j | X_k$.
   
   (c) And so forth.
3. Orient edges, where possible.

- Doesn’t work in gene expression time series as there is not enough dynamic noise.

- In addition to *statistical association* among variables, test for *functional relation*.  
  - Geometry of data plays role.  
  
  Wolf & Theis, in preparation (2016)
Integrating on the graph  

\[
\begin{align*}
\frac{dX_0}{dt} &= \frac{X_0}{1 + X_0} \frac{1}{1 + X_1} - X_0 + N_0 =: V_0 \\
\frac{dX_1}{dt} &= \frac{X_1}{1 + X_1} \frac{1}{1 + X_0} - X_1 + N_1 =: V_1 
\end{align*}
\]

Wolf, Fischer & Theis, in preparation (2016)
Integrating on the graph

Wolf, Fischer & Theis, in preparation (2016)

\[
\frac{dX_0}{dt} = \frac{X_0}{1 + X_0} \frac{1}{1 + X_1} - X_0 + N_0 =: V_0
\]
\[
\frac{dX_1}{dt} = \frac{X_1}{1 + X_1} \frac{1}{1 + X_0} - X_1 + N_1 =: V_1
\]

Statistical model $\tilde{V}$

\[
\tilde{V}_i = \sum_{k} \alpha_k X_k + \beta
\]
Integrating on the graph  

Wolf, Fischer & Theis, in preparation (2016)

\[
\begin{align*}
\frac{dX_0}{dt} &= \frac{X_0}{1 + X_0} \frac{1}{1 + X_1} - X_0 + N_0 =: V_0 \\
\frac{dX_1}{dt} &= \frac{X_1}{1 + X_1} \frac{1}{1 + X_0} - X_1 + N_1 =: V_1
\end{align*}
\]

Statistical model \( \tilde{V} \)

\[
\tilde{V}_i = \sum_{k} \alpha_k X_k + \beta
\]

Dynamics induced by \( \tilde{V} \)?
Integrating on the graph

\[
\begin{align*}
\frac{dX_0}{dt} &= \frac{X_0}{1 + X_0} \frac{1}{1 + X_1} - X_0 + N_0 =: V_0 \\
\frac{dX_1}{dt} &= \frac{X_1}{1 + X_1} \frac{1}{1 + X_0} - X_1 + N_1 =: V_1
\end{align*}
\]

Statistical model \( \tilde{V} \)

\[
\tilde{V}_i = \sum_k \alpha_k X_k + \beta
\]

Dynamics induced by \( \tilde{V} \)?

For the stochastic-mechanistic model, \( X(t) = X_0 + \int_0^t dt \ V(t) \).
Integrating on the graph

Wolf, Fischer & Theis, in preparation (2016)

\[
\frac{dX_0}{dt} = \frac{X_0}{1 + X_0} \frac{1}{1 + X_1} - X_0 + N_0 =: V_0
\]

\[
\frac{dX_1}{dt} = \frac{X_1}{1 + X_1} \frac{1}{1 + X_0} - X_1 + N_1 =: V_1
\]

Statistical model \( \tilde{V} \)

\[
\tilde{V}_i = \sum_k \alpha_k X_k + \beta
\]

Dynamics induced by \( \tilde{V} \)?

For the stochastic-mechanistic model, \( X(t) = X_0 + \int_0^t dt \ V(t) \).

For the statistic model \( \tilde{V} \), “integrate on the graph”

\[
A_{x_i, x_j} = \mathcal{N}(x_i | \tilde{x}_i(x_j), \sigma^2) \quad \text{(Markov Model)}
\]
Summary

- Tensor Trains/ Matrix Product States: low-rank factorization of high-dimensional distributions or wave functions.

- Dynamical Mean-Field Theory: learn something about a lattice problem from a single cluster.

- Graphical Models in Machine Learning: exact factorization of high-dimensional distributions with applications, for example, in causal inference.

Thanks to U. Schollwöck!
Summary

- Tensor Trains/ Matrix Product States: low-rank factorization of high-dimensional distributions or wave functions.
- Dynamical Mean-Field Theory: learn something about a lattice problem from a single cluster.
Summary

- Tensor Trains/ Matrix Product States: low-rank factorization of high-dimensional distributions or wave functions.
- Dynamical Mean-Field Theory: learn something about a lattice problem from a single cluster.
- Graphical Models in Machine Learning: exact factorization of high-dimensional distributions with applications, for example, in causal inference.
Summary

- Tensor Trains/ Matrix Product States: low-rank factorization of high-dimensional distributions or wave functions.
- Dynamical Mean-Field Theory: learn something about a lattice problem from a single cluster.
- Graphical Models in Machine Learning: exact factorization of high-dimensional distributions with applications, for example, in causal inference.

Thanks to U. Schollwöck!
Summary

- Tensor Trains/ Matrix Product States: low-rank factorization of high-dimensional distributions or wave functions.
- Dynamical Mean-Field Theory: learn something about a lattice problem from a single cluster.
- Graphical Models in Machine Learning: exact factorization of high-dimensional distribution with applications, for example, in causal inference.

Thanks to U. Schollwöck!

Thank you!
Eisert, J., 2013, Modeling and Simulation 3, 520.