Orthogonalized Alternating Least Squares: Theoretically principled tensor factorization, for practical use

Vatsal Sharan

Joint work with
Gregory Valiant

Stanford, Electrical Engineering & Computer Science

SIAM Annual Meeting

Based on ICML'17 paper

Tensors in Machine Learning

Theory

"Tensors are the best thing since sliced bread"

The Theoretical Power of Tensor Decomposition

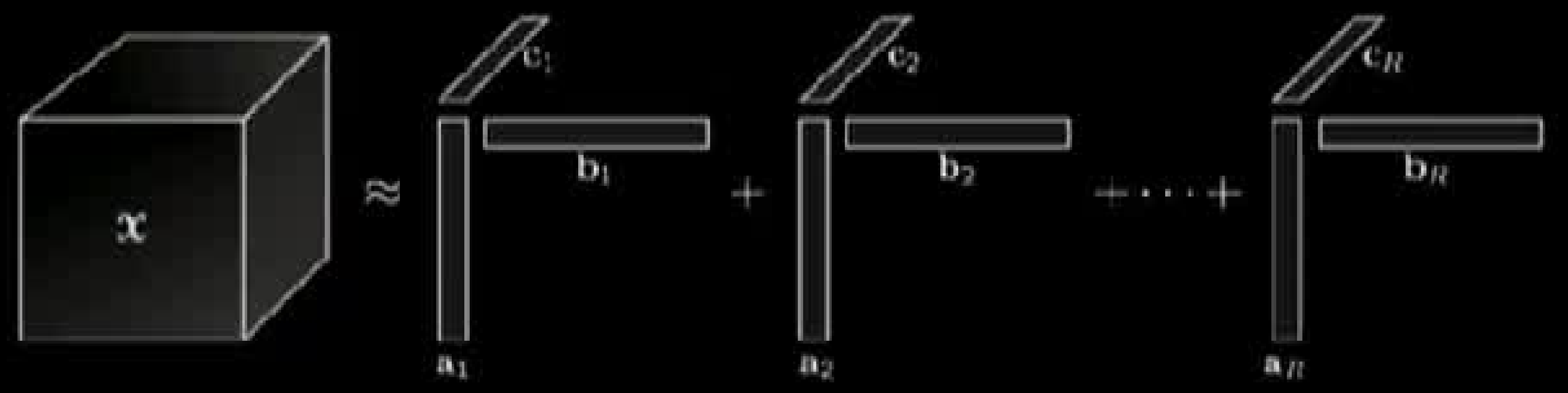
The Theoretical Power of Tensor Decomposition

- Understand data matrix <--> understanding its low rank structure (singular values, singular vectors etc.)

The Theoretical Power of Tensor Decomposition

- Understand data matrix <--> understanding its low rank structure (singular values, singular vectors etc.)
- Analog exists for tensors: Tensor decomposition is the decomposition of a tensor in terms of rank-1 tensors--

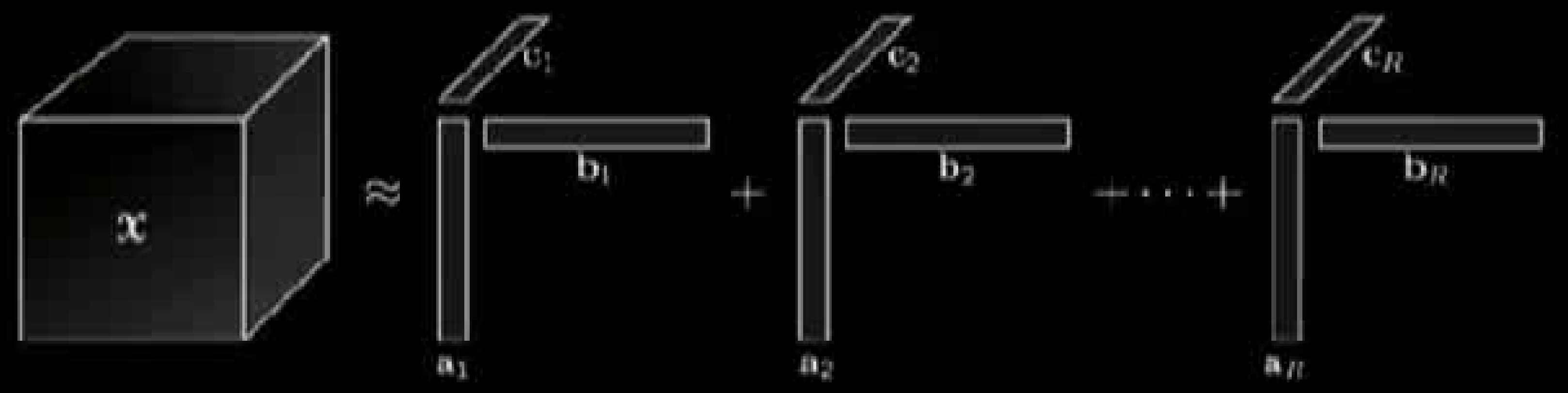
$$T = \sum_{i \in [k]} w_i a_i \otimes b_i \otimes c_i ; w_i \in \mathbb{R}; a_i, b_i, c_i \in \mathbb{R}^d$$



The Theoretical Power of Tensor Decomposition

- Understand data matrix <--> understanding its low rank structure (singular values, singular vectors etc.)
- Analog exists for tensors: Tensor decomposition is the decomposition of a tensor in terms of rank-1 tensors--

$$T = \sum_{i \in [k]} w_i a_i \otimes b_i \otimes c_i; w_i \in \mathbb{R}; a_i, b_i, c_i \in \mathbb{R}^d$$



- Even better than matrix decompositions: if factors linearly independent, then decomposition UNIQUE (*not* just up to rotation).

Many Applications of Uniqueness

Uniqueness of (low-rank) tensor factorization can be leveraged in many settings to give provable parameter recovery:

Many Applications of Uniqueness

Uniqueness of (low-rank) tensor factorization can be leveraged in many settings to give provable parameter recovery:

- Learning latent variable models, such as topic modeling, mixture models (e.g. Mossel/Roch'06, Anandkumar/Ge/Hsu/Kakade/Telgarsky '14, Ge/Huang/Kakade'15]
- Community detection [e.g. Brubaker/Vempala'09, Anandkumar/Ge/Hsu/Kakade'13]
- Training neural networks [Janzamin/Sedghi/Anandkumar'15]
- ...

Tensors in Machine Learning

Theory

"Tensors are the best thing since sliced bread"

Tensors in Machine Learning

Theory

"Tensors are the best thing since sliced bread"

Practice

Growing applications, but have not [yet] realized their potential, given the theory.

(Particularly for large-scale settings)

Why is this the case?

a) Tensors are inherently not useful in some practical settings (e.g. for many settings, matrix methods work just as well, or better)?
b) Information theoretic difficulties: e.g. datasets not large enough to fill out extra dimensions?
c) We need better algorithms to utilize their full potential.

Tensors in Machine Learning

Theory

"Tensors are the best thing since sliced bread"

Practice

Growing applications, but have not [yet] realized their potential, given the theory.

(Particularly for large-scale settings)

Why is this the case?

a) Tensors are inherently not useful in some practical settings (e.g. for many settings, matrix methods work just as well, or better)?
b) Information theoretic difficulties: e.g. datasets not large enough to fill out extra dimensions?
c) We need better algorithms to utilize their full potential.

Tensors in Machine Learning

"Tensors are the best thing since sliced bread"

…ons, but have not [yet] realized their potential, given the theory.

(Particularly for large-scale settings)

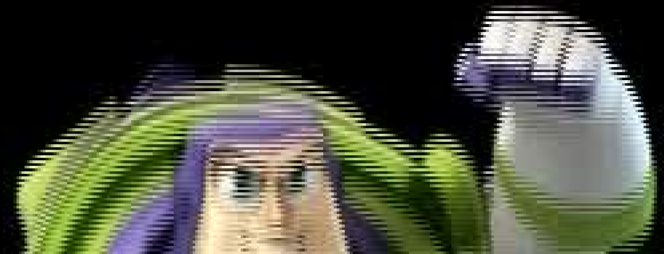
Why is this the case?

a) Tensors are inherently not useful in some practical settings (e.g. for many settings, matrix methods work just as well, or better)?

b) Information theoretic difficulties: e.g. datasets not large enough to fill out extra dimensions?

c) We need better algorithms to utilize their full potential.

ity and beyond!!



Tensors in Machine Learning

"Tensors are the best thing since sliced bread"

have not [yet] realized their potential, given the theory.

(Particularly for large-scale settings)

Why is this the case?

a) Tensors are inherently not useful in some practical settings (e.g. for many settings, matrix methods work just as well, or better)?
b) Information theoretic difficulties: e.g. datasets not large enough to fill out extra dimensions?
c) We need better algorithms to utilize their full potential.

...finity, and beyond!!

Tensors in Machine Learning

"Tensors are the best thing since sliced bread"

…ized their potential, given the theory.

(Particularly for large-scale settings)

Why is this the case?

a) Tensors are inherently not useful in some practical settings (e.g. for many settings, matrix methods work just as well, or better)?
b) Information theoretic difficulties: e.g. datasets not large enough to fill out extra dimensions?
c) We need better algorithms to utilize their full potential.

To infinity, and beyond!!

Tensors in Machine Learning

"Tensors are th... since sliced bread"

...potential, given the theory. (Particularly for large-scale settings)

Why is this the case?

a) Tensors are inherently not useful in some practical settings (e.g. for many settings, matrix methods work just as well, or better)?

b) Information theoretic difficulties: e.g. datasets not large enough to fill out extra dimensions?

c) We need better algorithms to utilize their full potential.

To infinity, and beyond!!

Why is this the case?

a) Tensors are inherently not useful in some practical settings (e.g. for many settings, matrix methods work just as well, or better)?

b) Information theoretic difficulties: e.g. datasets not large enough to fill out extra dimensions?

c) We need better algorithms to utilize their full potential.

To infinity, and beyond!!

Why is this the case?

a) Tensors are inherently not useful in some practical settings (e.g. for many settings, matrix methods work just as well, or better)?

b) Information theoretic difficulties: e.g. datasets not large enough to fill out extra dimensions?

c) We need better algorithms to utilize their full potential.

To infinity, and beyond!!

Why is this the case?

a) Tensors are inherently not useful in some practical settings (e.g. for many settings, matrix methods work just as well, or better)?

b) Information theoretic difficulties: e.g. datasets not large enough to fill out extra dimensions?

c) We need better algorithms to utilize their full potential.

To infinity, and beyond!!

Our contribution

We propose a new algorithm -- Orthogonalized ALS -- which is:

1. Computationally efficient and conceptually simple
2. Has fairly strong theoretical guarantees
3. Seems to work well in practice

So what’s the challenge?

Recovery still not well understood:

- Worst-case, most tensor problems NP-hard [Hastad’90, Hillar/Lim ’13]
- Only starting to understand theory of efficient/robust recovery

So what's the challenge?

Recovery still not well understood:

- Worst-case, most tensor problems NP-hard [Hastad'90, Hillar/Lim '13]
- Only starting to understand theory of efficient/robust recovery
  - Orthogonal tensor decomposition [Kolda'01, Anandkumar/Ge/Hsu/Kakade/Telgarsky '14, Robeva/Seigal'16]
  - Analysis of tensor power method [Anandkumar/Ge/Janzamin'14]
  - For tensors with *random* factors (rank) k ~ (dimension) $d^{1.5}$ [Ma/Shi/Steurer'16]
  - …
- So far, most theoretically sound algorithms are impractical for large-scale settings.
- Practically viable heuristics have demonstrably poor performance in many settings.

Practical Desiderata:

Practical Desiderata:

- Must be noise stable – simultaneous diagonalization based method (Harshman'70) too sensitive

Practical Desiderata:

- Must be noise stable – simultaneous diagonalization based method (Harshman'70) too sensitive
- Must be able to exploit sparsity in the tensor (e.g. tensor of word tri-occurrences might be 50k x 50k x 50k, but will be very sparse) ["deflation"-based methods too expensive in practice]
- Should work well even if factors have highly non-uniform weights (e.g. power-law decay).
- Should have runtime that is very low-degree polynomial… [no d^4 linear systems, $1/d^{10}$ probability of success, or expensive initializations]

ALS: Practically Feasible Candidate

ALS: Practically Feasible Candidate

Alternating Least Squares (ALS)

- Initialize factors randomly $\{A_i\},\{B_i\},\{C_i\}$
- iteratively fix 2 of 3 sets of factors, optimize 3rd set: e.g. fix $\{A_i\}$ and $\{B_i\}$ and find $\{C_i\}$ to minimize

  $\|T - \Sigma_i A_i \otimes B_i \otimes C_i \|$ *(objective function)*

  (least-squares problem!!)

ALS: Practically Feasible Candidate

Alternating Least Squares (ALS)

- Initialize factors randomly $\{A_i\},\{B_i\},\{C_i\}$
- iteratively fix 2 of 3 sets of factors, optimize 3[rd] set: e.g. fix $\{A_i\}$ and $\{B_i\}$ and find $\{C_i\}$ to minimize

$$\|T - \Sigma_i A_i \otimes B_i \otimes C_i\|$$ *(objective function)*

(least-squares problem!!)

ALS is "workhorse" of tensor methods in practice

- Computationally efficient (many optimized packages, e.g. Tensor Toolbox)
- Often gets stuck in bad local optima

ALS: Practically Feasible Candidate

Key Issue:

Local optima arise when multiple estimated factors all chasing after the *same* true factors.

For tensors with skewed weights, large weight factors much more attractive than low-weight factors.

"Orthogonalized" ALS

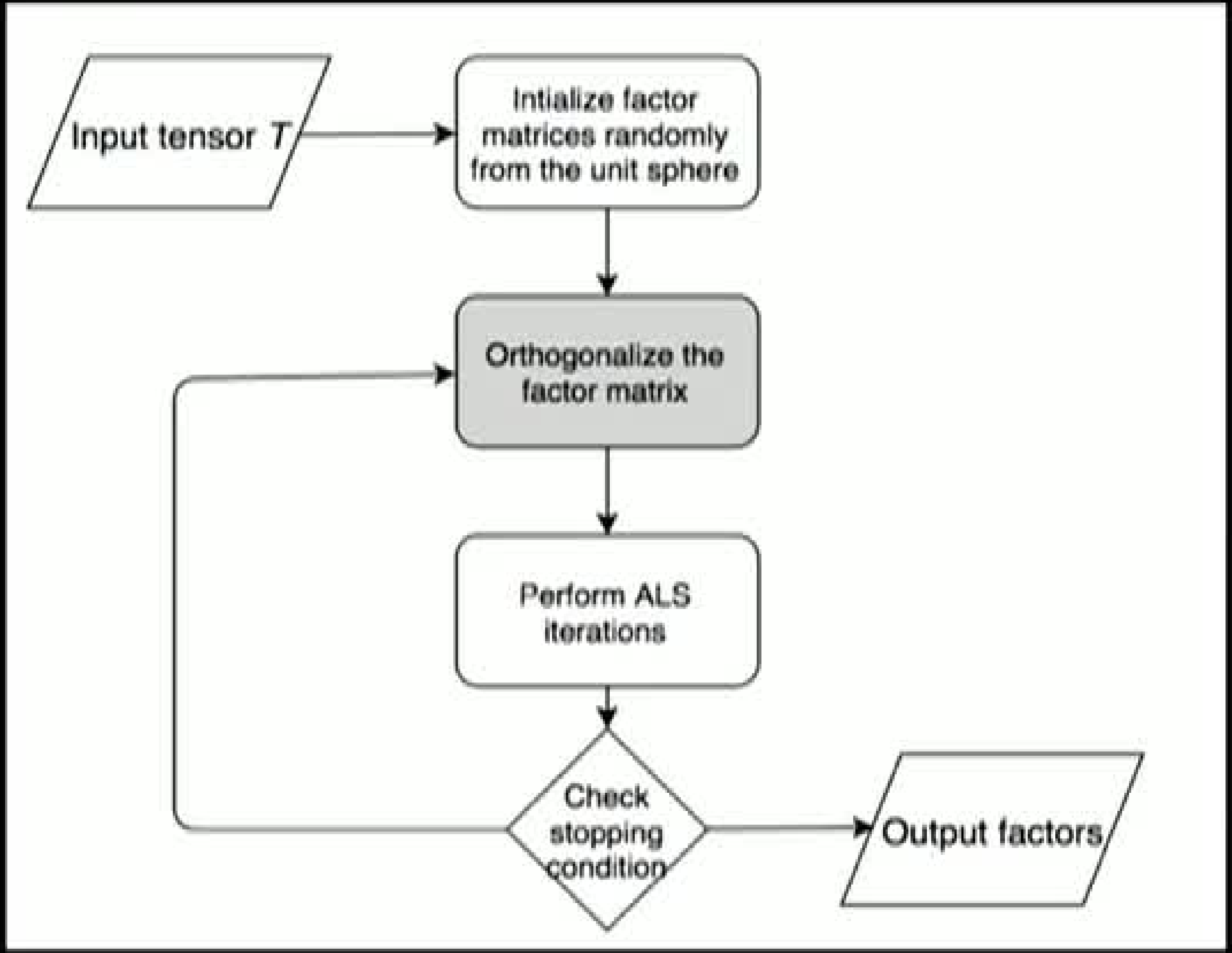
Idea: Periodically orthogonalize recovered factors

(Note: returned factors will not necessarily be orthogonal)

"Orthogonalized" ALS

Idea: Periodically orthogonalize recovered factors

(Note: returned factors will not necessarily be orthogonal)



Orthogonalization of factor matrix A:

1. Keep the first factor
2. Project the 2^{nd} factor orthogonal to first factor
3. Project the 3^{rd} factor orthogonal to first 2 factors
4. ....

Motivation: Finding eigenvectors of a matrix

Motivation: Finding eigenvectors of a matrix

To find the largest eigenvector: just use matrix power method

Motivation: Finding eigenvectors of a matrix

To find the largest eigenvector: just use matrix power method

To find the second largest eigenvector:
Need to project orthogonal to first eigenvector after power method step

Motivation: Finding eigenvectors of a matrix

To find the largest eigenvector: just use matrix power method

To find the second largest eigenvector:
Need to project orthogonal to first eigenvector after power method step

Orthogonalization prevents multiple recovered factors from chasing the same original factors

A simulation

A simulation

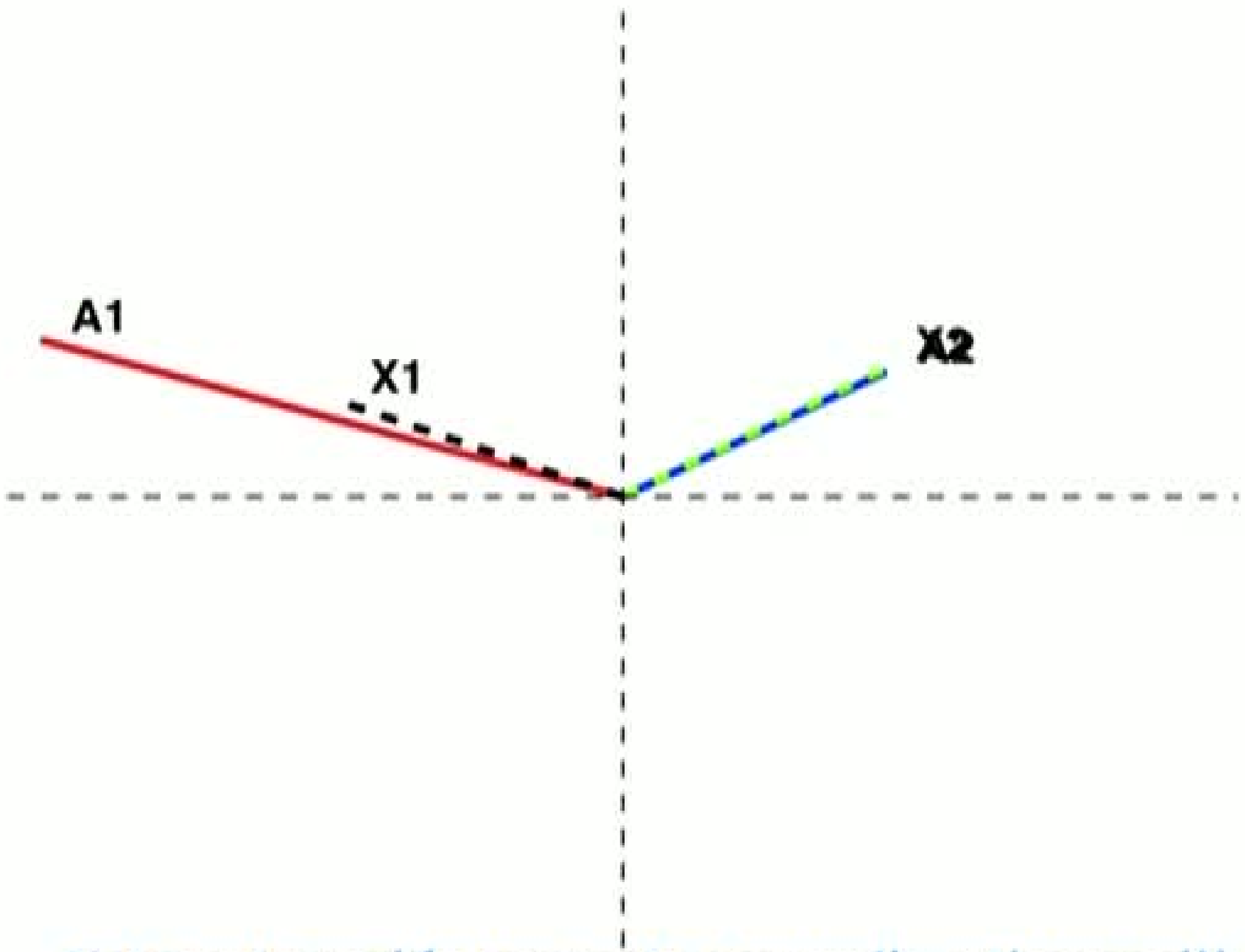
Consider a symmetric rank 2 tensor in 2 dimensions: T ϵ R^{2x2x2}

$$T = A_1 \otimes A_1 \otimes A_1 + A_2 \otimes A_2 \otimes A_2$$

Recovered tensor:

$$\hat{T} = X_1 \otimes X_1 \otimes X_1 + X_2 \otimes X_2 \otimes X_2$$

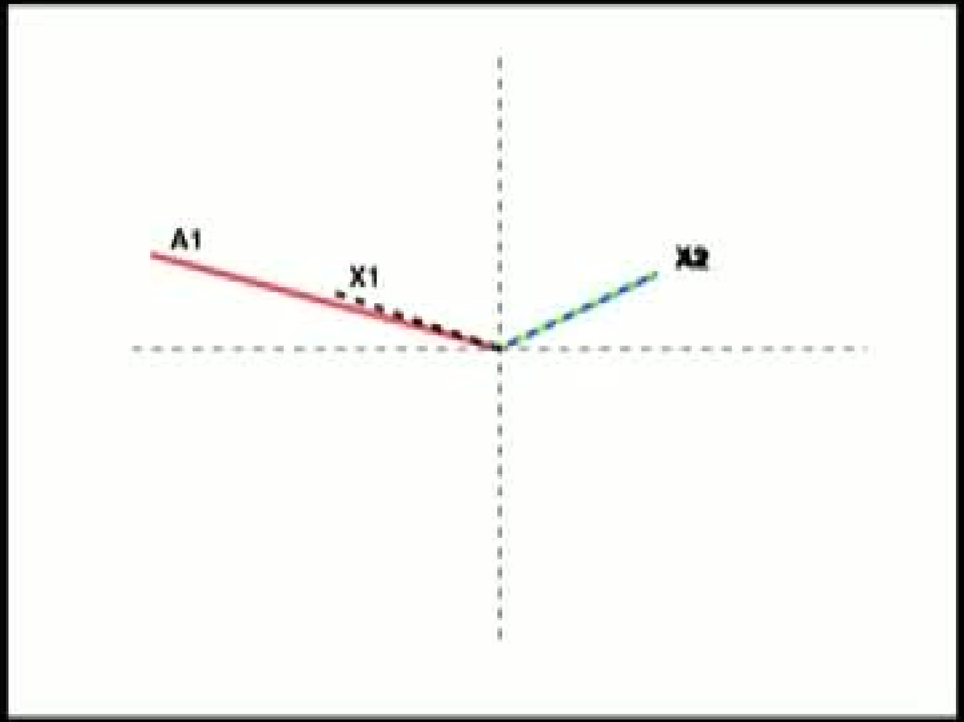
ALS Step



Note: returned factors not necessarily orthogonal!!

Lots of variants possible:

“Hybrid-ALS”: orthogonalize for first few iterations, then switch to normal ALS.



Guarantees for Orthogonalized-ALS

Guarantees for Orthogonalized-ALS

Theorem (informal): Orthogonalized ALS recovers the true factors under reasonable conditions, with *random initialization*, extremely quickly!

Guarantees for Orthogonalized-ALS

Theorem: Given *d* dimensional rank *k* tensor,
$T = \Sigma_i w_i A_i \otimes A_i \otimes A_i$, let $c=\max_{i,j}|\langle A_i,A_j\rangle|$ and $q=\max_{i,j}(w_i/w_j)$.
If **$cq < 1/k^2$** then Orth-ALS recovers factors in
O(klog k + kloglog d) steps whp if **initialized randomly** from unit sphere: $||\hat{A}_i - A_i|| < k^{1/2} \max(c,1/d)$

Guarantees for Orthogonalized-ALS

Theorem: Given d dimensional rank k tensor,
$T = \Sigma_i\, w_i A_i \otimes A_i \otimes A_i$, let $c=\max_{i,j}|\langle A_i,A_j\rangle|$ and $q=\max_{i,j}(w_i/w_j)$.
If **$cq < 1/k^2$** then Orth-ALS recovers factors in
$O(k\log k + k\log\log d)$ steps whp if **initialized randomly** from unit sphere: $||\hat{A}_i - A_i|| < k^{1/2}\max(c,1/d)$

Corollary: Orth-ALS recovers factors for **random** d dimensional tensors, if rank $k = O(d^{1/4})$

Guarantees for Orthogonalized-ALS

Theorem: Given d dimensional rank k tensor,
$T = \Sigma_i w_i A_i \otimes A_i \otimes A_i$, let $c=\max_{i,j}|\langle A_i,A_j\rangle|$ and $q=\max_{i,j}(w_i/w_j)$.
If **$cq < 1/k^2$** then Orth-ALS recovers factors in
O(klog k + kloglog d) steps whp if **initialized randomly** from unit sphere: $||\hat{A}_i - A_i|| < k^{1/2} \max(c,1/d)$

As consequence of analysis, also can show improved convergence of Tensor Power Method (for incoherent tensors)

Guarantees for Tensor Power Method

Theorem: For random tensor in d dimensions with rank $k < d$, whp over **random initialization**, Tensor Power Method converges to one of the true factors after **log log d** iterations.

Previous results [Anandkumar/Ge/Janzamin] showed local convergence with a special SVD initialization, and a linear convergence rate (log d iterations)

Practical Evaluation

Synthetic Data:

Practical Evaluation

Synthetic Data:

- Random low-rank tensors
  - Uniform weights and geometrically spaced decaying weights
  - Noiseless, and with independent Gaussian noise.

Practical Evaluation

Synthetic Data:

- Random low-rank tensors
  - Uniform weights and geometrically spaced decaying weights
  - Noiseless, and with independent Gaussian noise.

Real-world data:

- Computing word embeddings from 1.5B word English Wikipedia corpus.
- Evaluation of embeddings on semantic tasks (analogies, and word similarity tasks).

Recovering Random Low-Rank Tensors

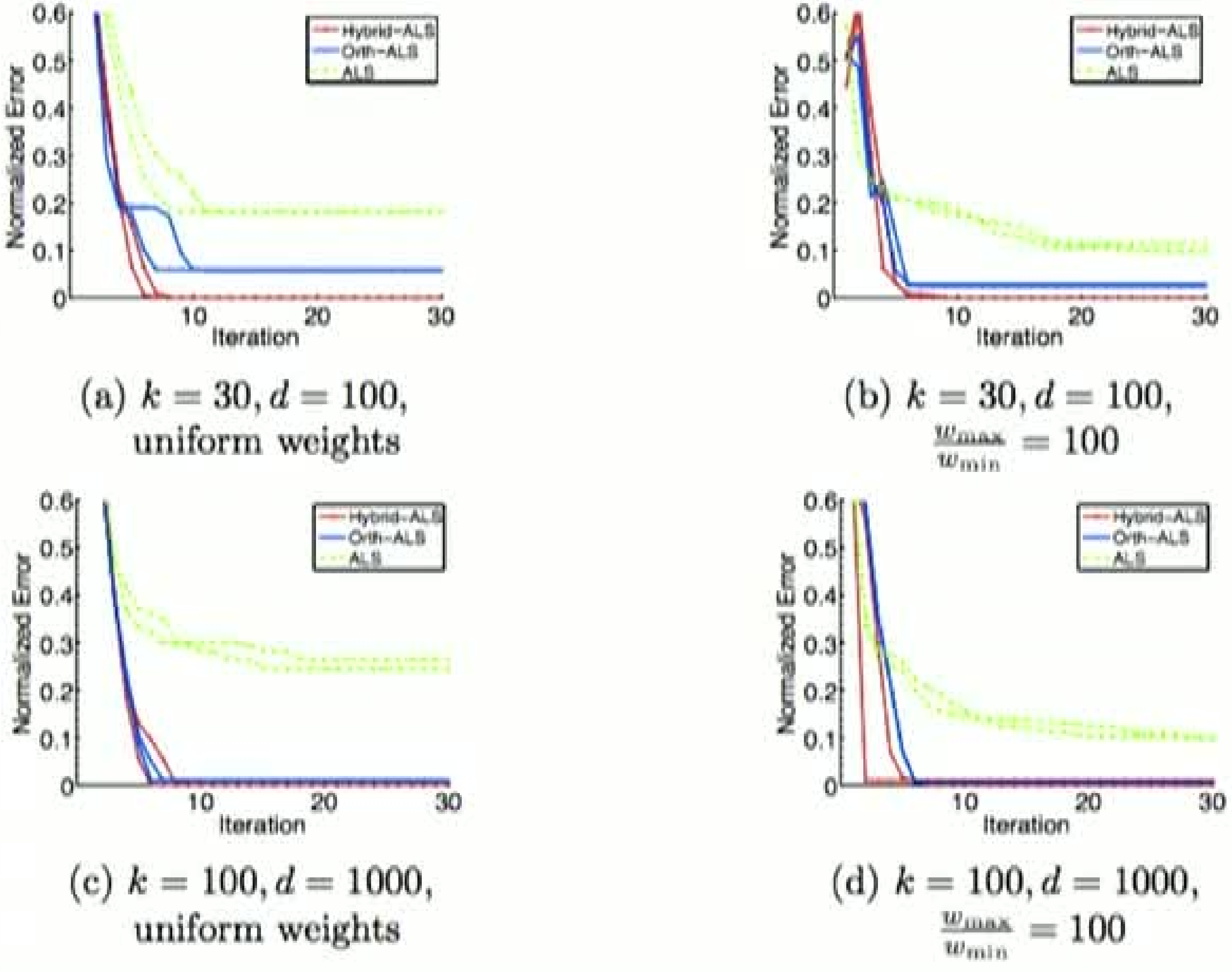
Problem: Given rank k tensor T, recover rank k tensor T^*

Evaluation metric: Normalized error = $\|T - T^*\|_F / \|T\|_F$

Recovering Random Low-Rank Tensors

Problem: Given rank k tensor T, recover rank k tensor T^*

Evaluation metric: Normalized error = $\|T - T^*\|_F / \|T\|_F$



(a) $k = 30, d = 100$, uniform weights

(b) $k = 30, d = 100$, $\frac{w_{max}}{w_{min}} = 100$

(c) $k = 100, d = 1000$, uniform weights

(d) $k = 100, d = 1000$, $\frac{w_{max}}{w_{min}} = 100$

Recovering Random Low-Rank Tensors

Problem: Given rank k tensor T, recover rank k tensor T^*

Evaluation metric: Number of factors successfully recovered

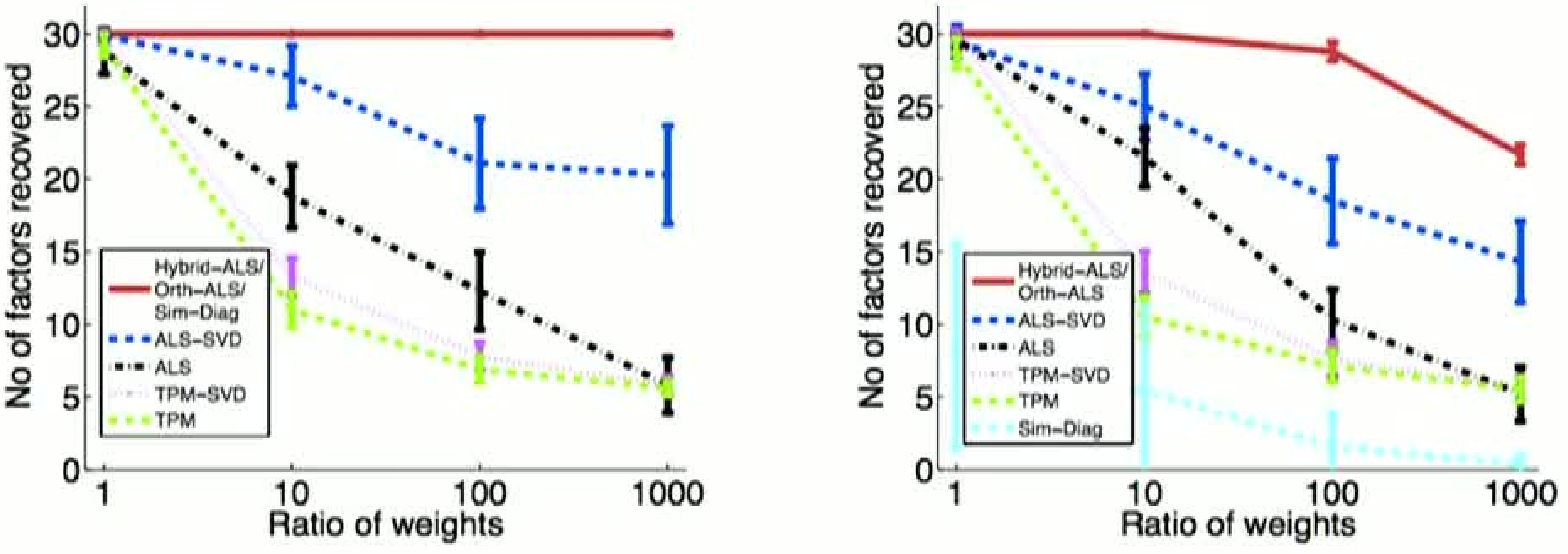
(e.g. "recovered" if correlation > 0.9)

Recovering Random Low-Rank Tensors

Problem: Given rank *k* tensor T, recover rank *k* tensor *T**

Evaluation metric: Number of factors successfully recovered

(e.g. "recovered" if correlation > 0.9)



Noiseless case

With Gaussian Noise

Word Embeddings

Word Embeddings

Goal: map words to vectors (typically dim<500) such that geometry encodes semantics.

Word Embeddings

Goal: map words to vectors (typically dim<500) such that

geometry encodes semantics.

Usual approach is to take word co-occurrence matrix and factor it.

We consider taking a word tri-occurrence 3-tensor and factorizing it.

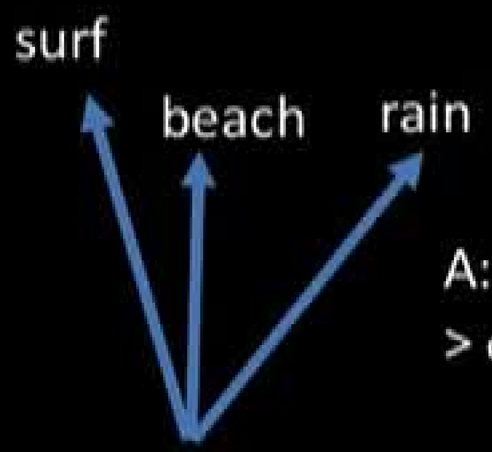
Evaluated via performance on downstream tasks.

Word Embeddings

Evaluate extent to which geometry encodes semantics via similarity, or analogy tasks:

Similarity

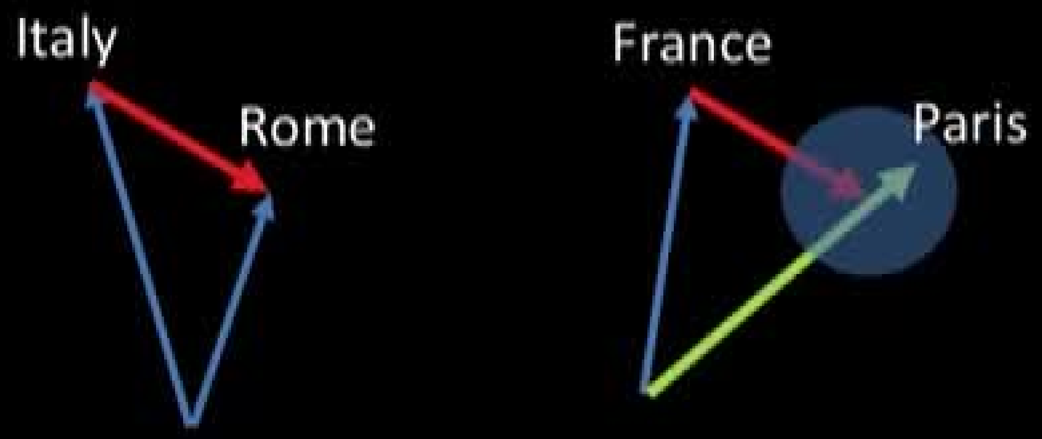
Q: Which are more similar, (beach, surf) or (beach, rain)?



A: is cosine-sim(beach,surf) > cosine-sim(beach,rain)?

Analogy

Italy:Rome as France:____ ?



Word Embeddings

Algorithm	Similarity Tasks		Analogy tasks	
	WordSim	MEN	Mixed analogies	Syntactic analogies
Vanilla ALS	0.44	0.51	30.22%	32.01%
Orth-ALS	0.56	0.60	45.87%	47.13%
Matrix SVD	0.59	0.68	54.29%	62.20%

Orthogonalized ALS much better than vanilla ALS.
Significant gains just by using better factorization algorithm.

Still not competitive with matrix methods, but step in right direction…