The Complexity of Unsupervised Learning

Santosh Vempala, Georgia Tech

Unsupervised learning

- Data is no longer the constraint in many settings

  … *(imagine sophisticated images here)*…

- But,
  - How to understand it?
  - Make use of it?
  - What data to collect?

- with no labels (or teachers)

Can you guess my passwords?

- GMAIL
- GMAIL → MU47286

Two general approaches

1. Clustering
   - Choose objective function or other quality measure of a clustering
   - Design algorithm to find (near-)optimal or good clustering
   - Check/hope that this is interesting/useful for the data at hand

2. Model fitting
   - Hypothesize model for data
   - Estimate parameters of model
   - Check that parameters where unlikely to appear by chance
   - (even better): find best-fit model ("agnostic")

Challenges

- Both approaches need domain knowledge and insight to define the "right" problem
- Theoreticians prefer generic problems with mathematical appeal
- Some beautiful and general problems have emerged. These will be the focus of this talk.
- There's a lot more to understand, that's the excitement of ML for the next century!
- E.g., How does the cortex learn? Much of it is (arguably) truly unsupervised ("Son, minimize the sum-of-squared-distances," is not a common adage)

Meta-algorithms

- PCA
- k-means
- EM
- …

- Can be "used" on most problems.
- But how to tell if they are effective? Or if they will converge in a reasonable number of steps?

- Do they work? When? Why?

This talk

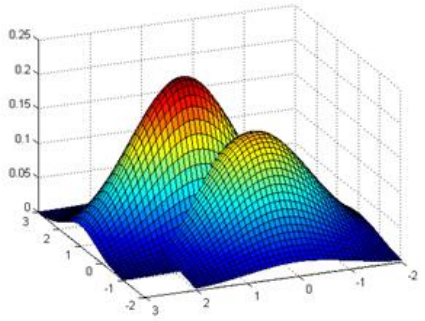
- Mixture Models
- Independent Component Analysis
- Finding Planted Structures

Many other interesting and widely studied models: topic models, hidden Markov models, dictionaries, identifying the relevant ("feature") subspace, etc.

Mixture Models

- Classify unlabeled samples from a unknown mixture of distributions; Learn parameters of the mixture.

$$F = w_1 F_1 + w_2 F_2 + \cdots + w_k F_k$$



- E.g., each component F_i is an unknown Gaussian, an unknown logconcave distribution, etc.

- Classification needs components to be well-separated.

- Learning Gaussian mixtures does not:
- Thm: Gaussian mixtures have unique decompositions.

Status: Learning parameters with no assumptions

- For any fixed k (number of Gaussians), there is a polynomial algorithm for learning a mixture of Gaussians up to a desired accuracy

  [Kalai-Moitra-Valiant, Belkin-Sinha, Moitra-Valiant]

  - Sample Complexity: $n^{f(k)}$.
  - Known lower bound: 2^k

- Open Problem 1: Is there an $f(k)poly(n)$ algorithm for learning Gaussian mixtures?

Techniques

- Random Projection

  [Dasgupta] Project mixture to a low-dimensional subspace to (a) make Gaussians more spherical and (b) preserve pairwise mean separation

  [Kalai] Project mixture to a random 1-dim subspace; learn the parameters of the resulting 1-d mixture; do this for a set of lines to learn the n-dimensional mixture!

- Method of Moments

  [Pearson] Finite number of moments suffice for 1-d Gaussians

  [Kalai-Moitra-Valiant] 6 moments suffice [B-S, M-V]

Status: Learning/Clustering with separation assumptions

- A1. Pairwise separation between means. (Clustering)

  Separation: $k^{\frac{1}{4}}(\sigma_i + \sigma_j)$ where σ_i^2 = max variance of component i.

  [Dasgupta, D-Schulman, Arora-Kannan, V-Wang, K.-Salmasian-V, Achlioptas-McSherry]

- A2. Each mean is separated from the span of the previous means in this ordering. (Clustering)

  Separation: $poly(k)$. standard deviation along separating direction

  [Brubaker-V.]

- A3. Matrix of means has a bounded smallest singular value. This implies that each mean is separated from the span of the rest. (Learning)

  Spherical Gaussians: complexity grows as 1/poly(separation).

  [Hsu-Kakade, Goyal-V.-Xiao]

- OP2: Complexity of learning a mixture of arbitrary Gaussians with linearly independent means?
- OP3: Minimum separation required to efficiently cluster a mixture of (spherical) Gaussians?

Techniques

PCA:

- Use PCA once
  [V-Wang]
- Use PCA twice
  [Hsu-Kakade]
- Eat chicken soup with rice; Reweight and use PCA
  [Brubaker-V., Goyal-V.-Xiao]

Polynomial Algorithms I: Clustering spherical Gaussians [VW02]

- Distance-based clustering:
  - needs separation that grows as $n^{\frac{1}{4}}$
- PCA, then cluster:
  - Separation required grows as $k^{\frac{1}{4}}$:

    $|\mu_i - \mu_j| > k^{\frac{1}{4}}(\sigma_i + \sigma_j)\log \ldots$
  - Projection to span of means preserves inter-mean distance and shrinks component Gaussians.
  - Span(means) = PCA subspace of dim k

PCA for spherical Gaussians

- Best line for 1 Gaussian?

  - Line through the mean

- Best k-subspace for 1 Gaussian?

  - Any k-subspace through the mean

- Best k-subspace for k Gaussians?

  - The k-subspace through all k means!

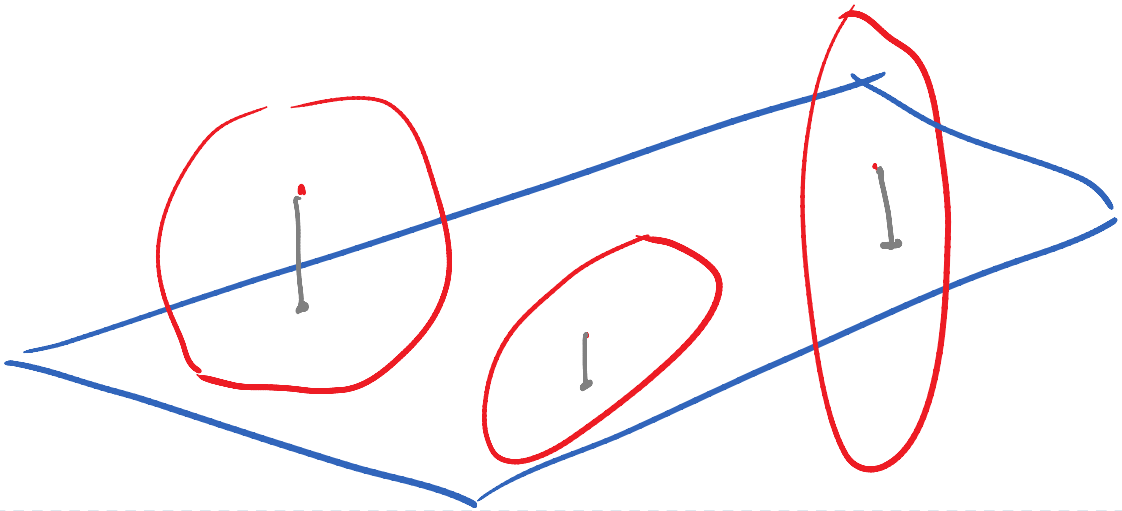
Mixtures of Nonisotropic, Logconcave Distributions [KSV04,AM05]

- Thm. PCA subspace is "close" to span of means.

- Separation required for classification:

$$|\mu_i - \mu_j| > poly(k)\left(\sigma_{i,max} + \sigma_{j,max}\right) \log \ldots$$

where $\sigma^2_{i,max}$ is the *maximum* directional variance

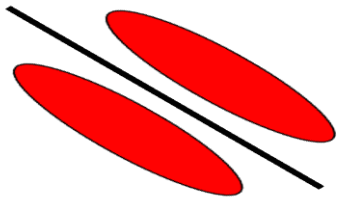


Crack my passwords

- GMAIL → MU47286
- AMAZON
- AMAZON → RU27316

Limits of PCA

- Can fail for a mixture of 2 arbitrary Gaussians



- Algorithm is not affine-invariant or noise-tolerant.
- Any instance can be made bad by an affine transformation or a few "bad" points.

Clustering and PCA

1. Apply PCA to embed in a low-dimensional subspace
2. Run favorite clustering algorithm (e.g., k-means iteration)

- [K.-Kumar] Converges efficiently for k-means iteration under a natural pairwise separation assumption.

- (important to apply PCA before running k-means!)

Polynomial Algorithms II: Learning spherical Gaussians [HK]

1. Make mixture isotropic (covariance = identity)
2. Construct 3th moment tensor of means
3. Decompose tensor to recover means, then variances and mixing weights

$$E(X \otimes X \otimes X) = \sum_j w_j \mu_j \otimes \mu_j \otimes \mu_j + \ \ldots$$

- After isotropy, means are orthogonal:

$$E(X \otimes X) = \sum_j w_j \mu_j \otimes \mu_j + \left(\sum_j w_j \sigma_j^2 \right) I$$

- 3rd moment tensor has unique decomposition, can be found by a power iteration.
- Complexity grows as inverse polynomial in separation --- smallest singular value of mean matrix
- Fourier PCA [GVX13] also works (and with Gaussian noise)

Status: Noisy mixtures

- Gaussian mixture + small fraction of arbitrary points.
- Previous algorithms fail. PCA is not noise-tolerant!

- Mixture of logconcave distributions can be learned with a $\log n$ factor extra pairwise separation, for noise $\epsilon = \tilde{O}(w_{min})$
  [Brubaker]

- Technique: Outlier removal interleaved with PCA.

Polynomial Algorithms III: Robust PCA for noisy mixtures [Brubaker09]

1. Remove points outside a ball.
2. Project to top $\frac{n+k}{2}$ principal components.
3. Repeat until dimension becomes k.

- Thm. Robust PCA classifies logconcave mixtures, provided:

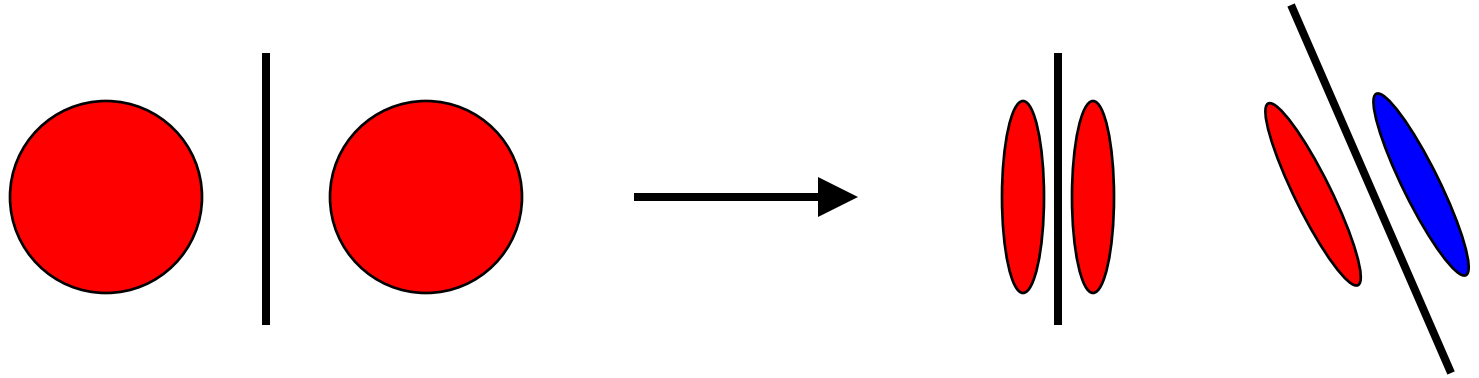
$$|\mu_i - \mu_j| > \frac{k^{\frac{3}{2}}}{w_{min}}(\sigma_{i,max} + \sigma_{j,max}) \log\frac{nk}{w_{min}} \quad \text{for } \epsilon < \frac{w_{min}}{\log^2\frac{nk}{w_{min}}}$$

- Similar to [KSV04] but with extra log factor.

Classifying Arbitrary Gaussian Mixtures

- Component Gaussians must be probabilistically separated for classification to be possible
- OP4: Is this enough?

- Probabilistic separation is affine invariant:



- PCA is not affine-invariant!

Polynomial Algorithms IV: Affine-invariant clustering [BV08]

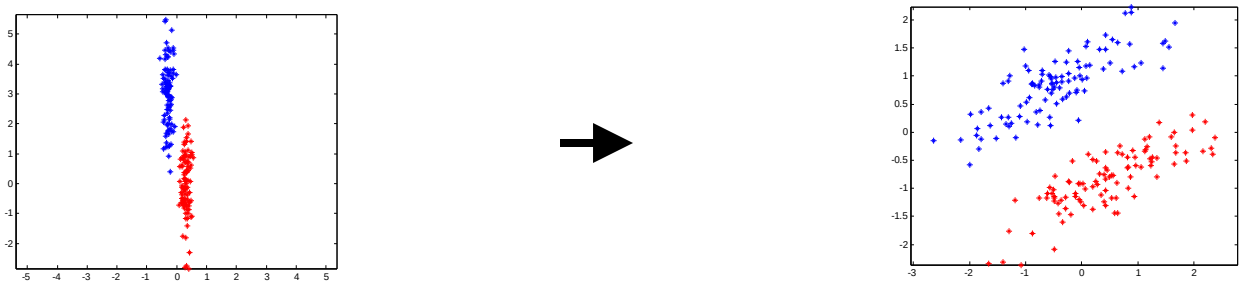
1. Make distribution isotropic.
2. Reweight points (using a Gaussian).
3. If mean shifts, partition along this direction; Recurse.
4. Otherwise, partition along top principal component; Recurse.

- Thm. The algorithm correctly classifies samples from a mixture of k arbitrary Gaussians if each one is separated from the span of the rest. (More generally, if the overlap is small as measured by the Fisher criterion).

- OP4: Extend Isotropic PCA to logconcave mixtures.

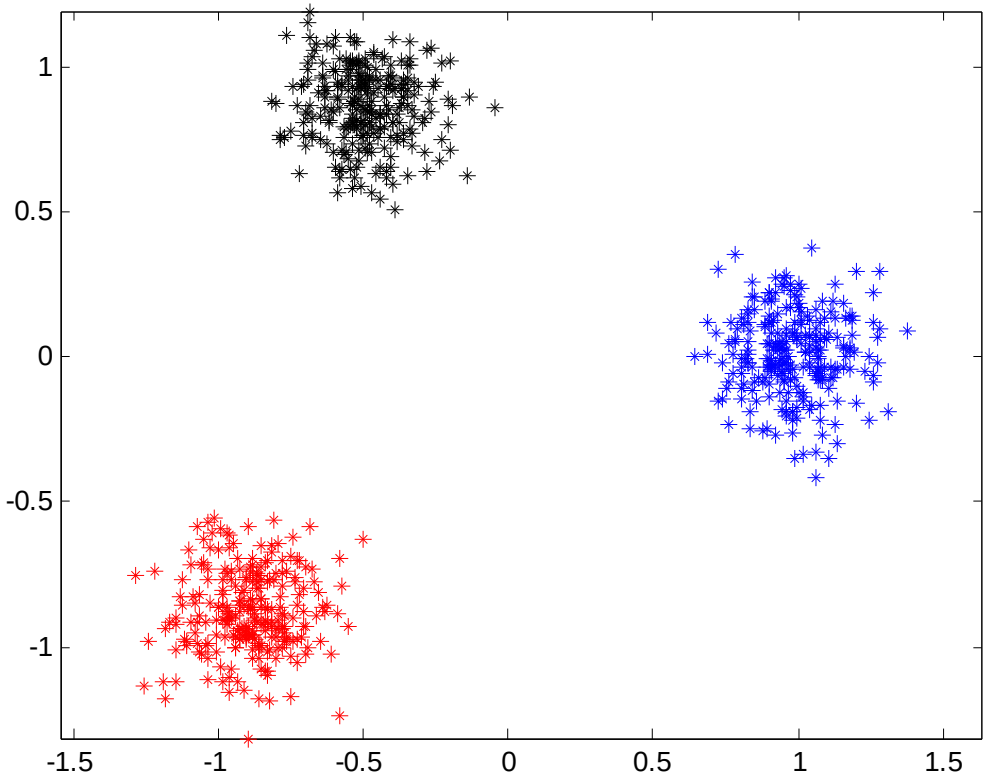
Unraveling Gaussian Mixtures

- Isotropy pulls apart the components



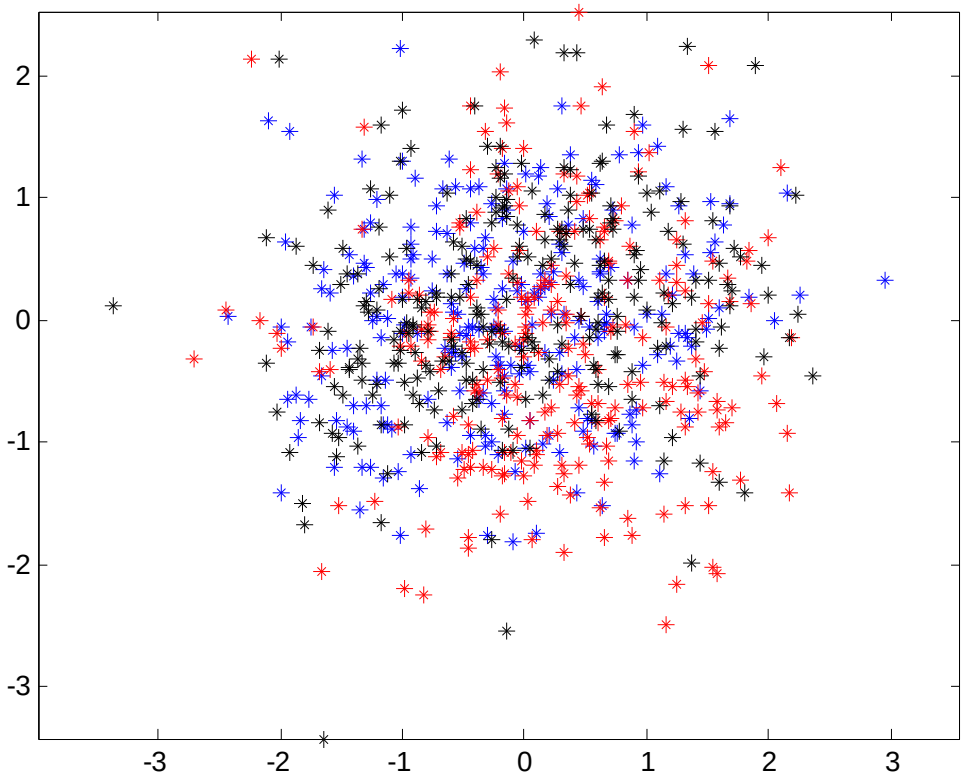
- If some component is heavier, then reweighted mean shifts along a separating direction
- If not, reweighted principal component is along a separating direction

Original Data

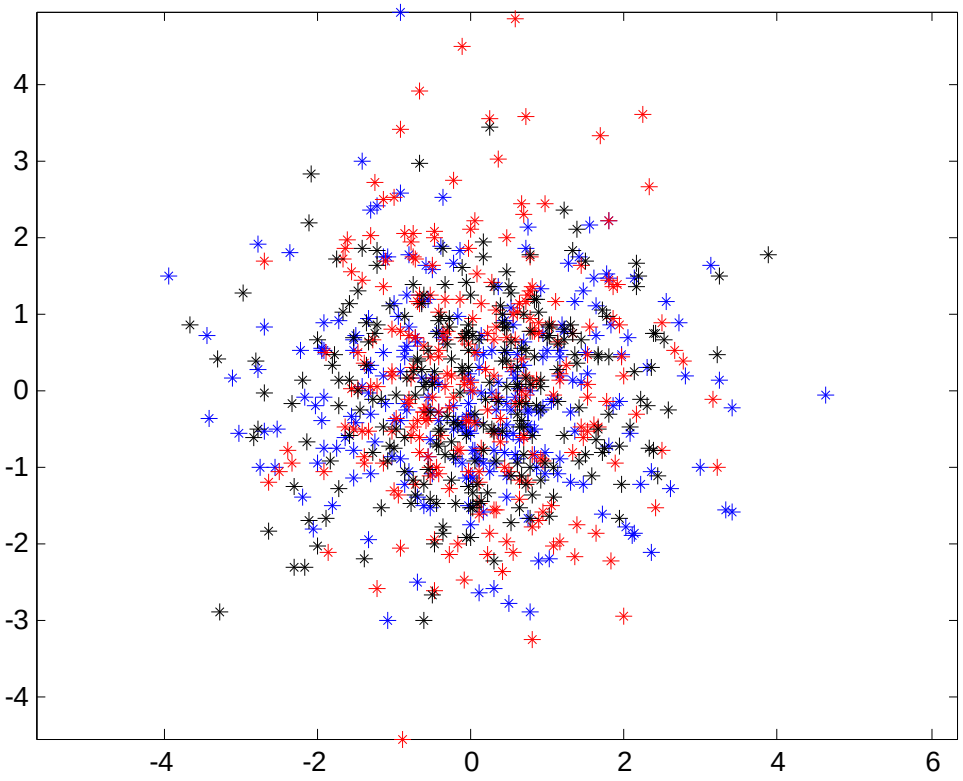


- 40 dimensions, 15000 samples (subsampled for visualization)

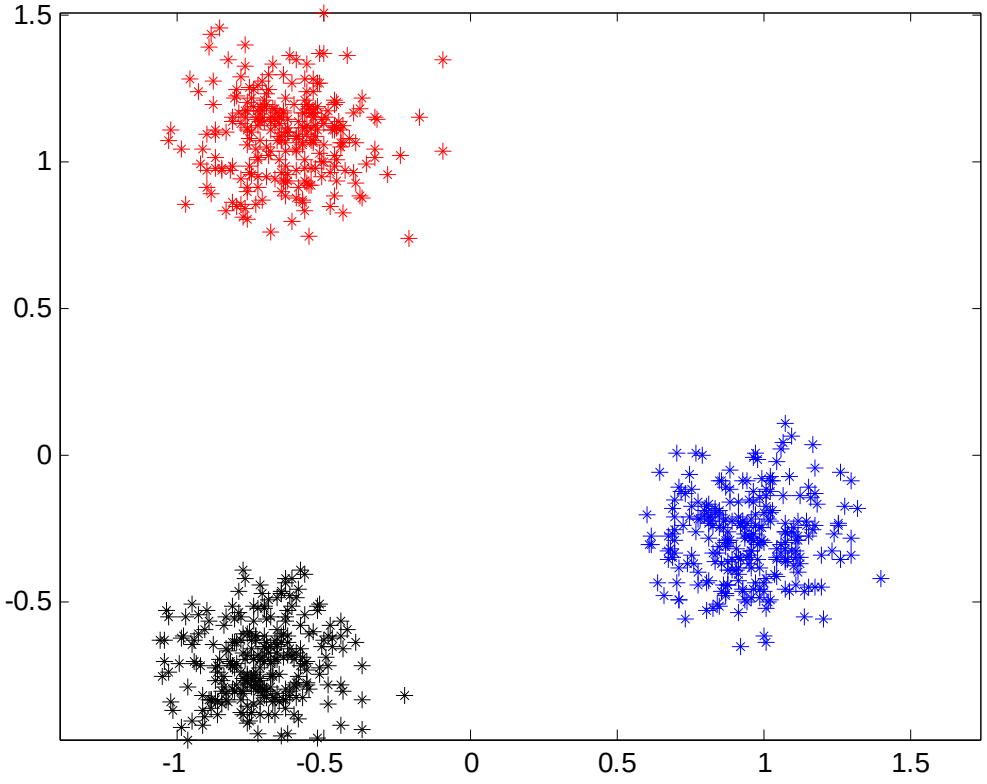
Random Projection



PCA



Isotropic PCA



Crack my passwords

- GMAIL → MU47286
- AMAZON → RU27316
- IISC
- IISC → LH857

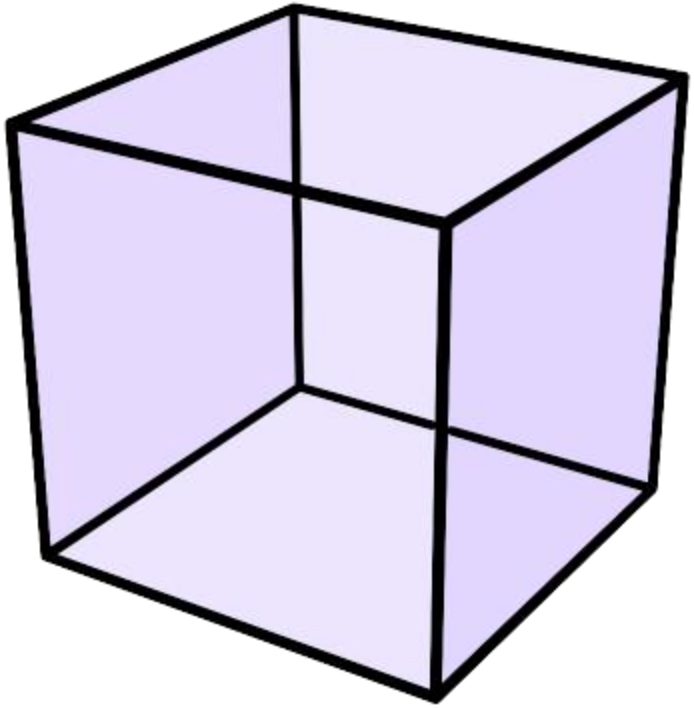
Independent Component Analysis [Comon]

- Model: Data is a linear transformation of an unknown product distribution:

$$s \in R^m, A \in R^{n \times m} \text{ data } x = As$$

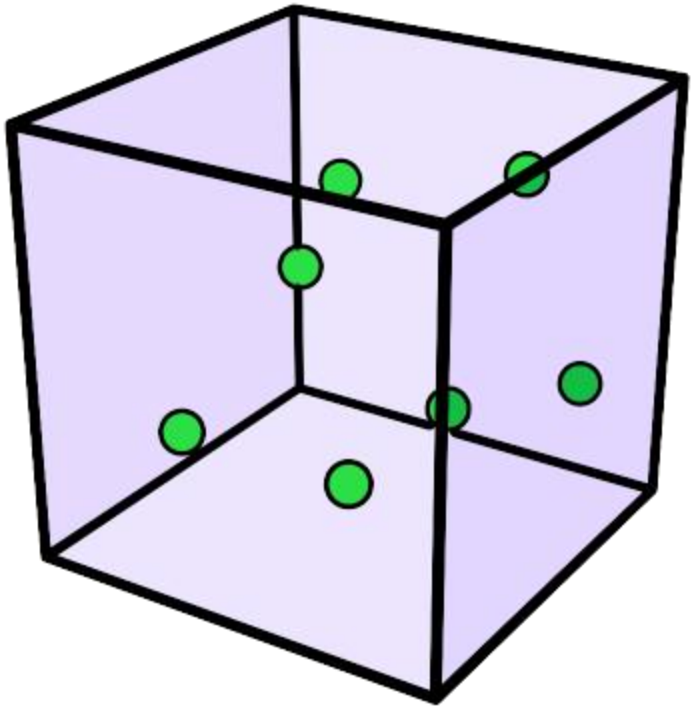
- A is unique up to signs of columns if at most one component s_i is Gaussian
- Problem: Learn A by observing samples x.
- Used extensively in ML, signal processing, neuroscience etc. for 25+ years.
- Many attractive heuristics.

Independent Component Analysis (ICA)



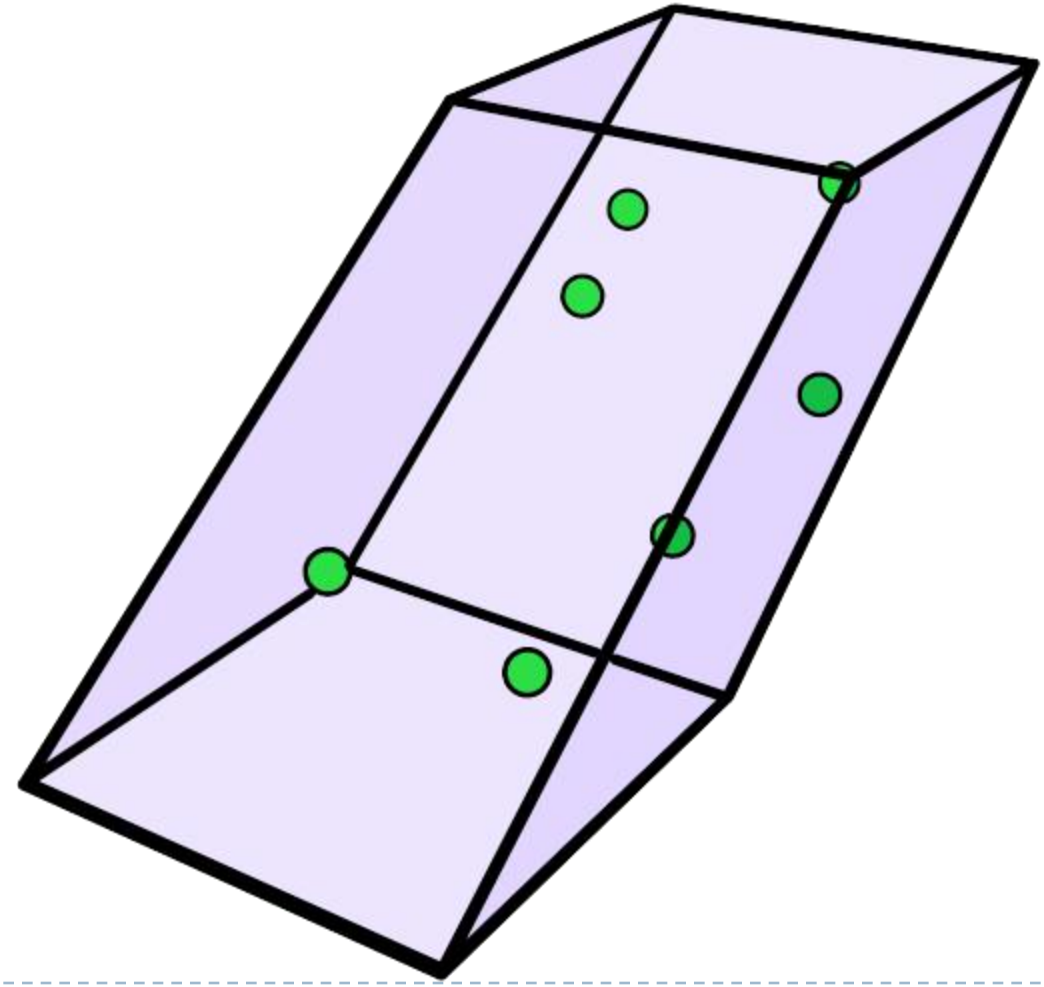
ICA model

- Start with a product distribution



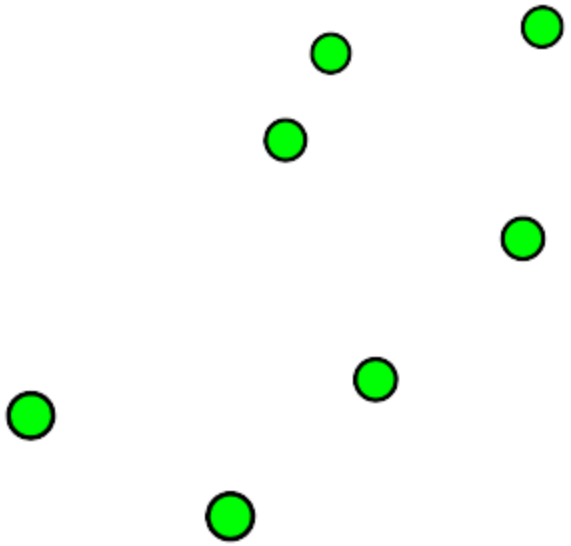
ICA model

- Apply a linear transformation A



ICA model

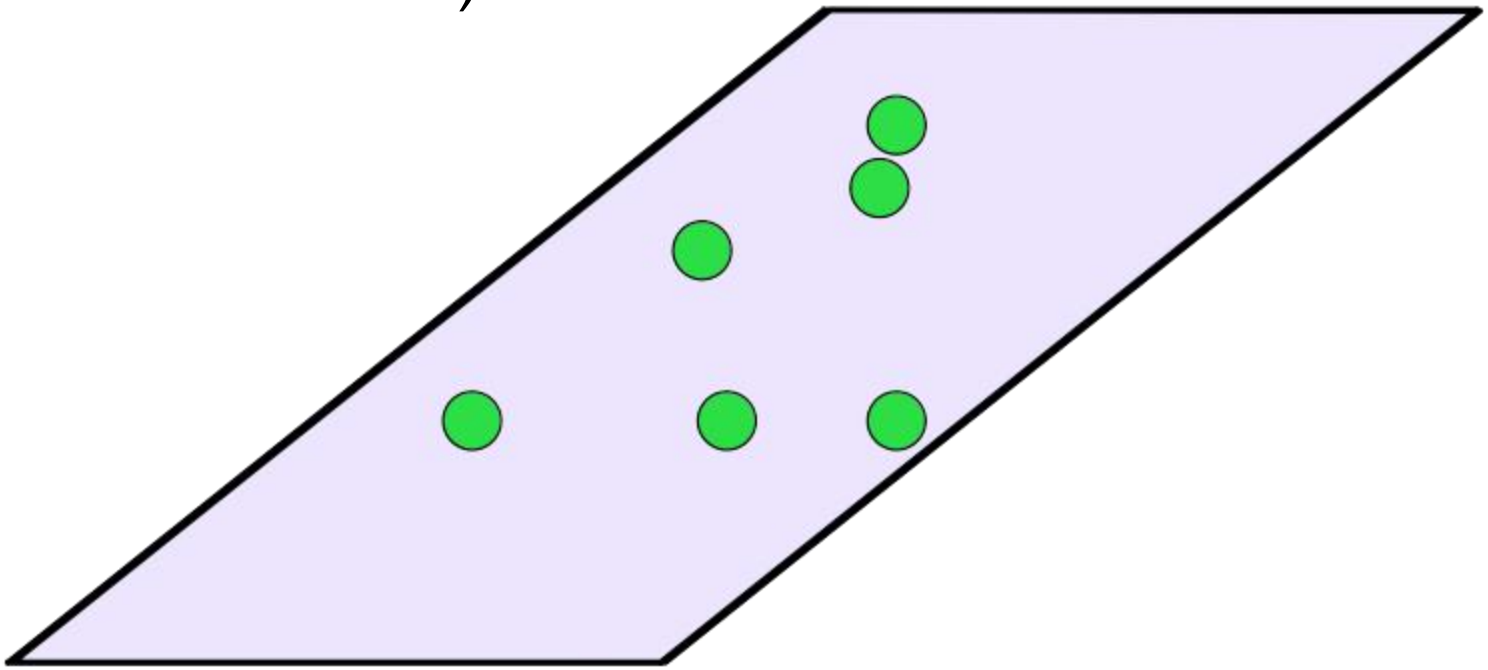
- Observed sample



ICA model

Matrix A might include a projection

(underdetermined ICA)



Status: ICA

- Thm [GVX13]. If columns of A satisfy a weak linear independence condition, and component distributions satisfy $|cum_{k_i}(s_i)| > \Delta$ for $k_i \leq k$, then A can be estimated with complexity $poly\left(m^k, \Delta, \frac{1}{\epsilon}\right)$.

- Generalized linear independence: smallest d for which the tensors $\bigotimes^d A_i$ are linearly independent.
- Earlier work for d=1 and k=4 [FJK,NR,AGMS,AGR]

- Thm[VX14]. If columns of A are linearly independent and $k \leq 4$, then sample complexity = $\tilde{O}(n)$ and time complexity = O(SVD)

- Both theorems work with Gaussian noise: $x = As + \eta$

- OP5: ICA with arbitrary noise?

Techniques

- PCA
- finds local optima of second moments, i.e., $\max_{u \in R^n} E((u^T x)^2)$

- Local optima of 4th moment. [Frieze-Jerrum-Kannan96]
  - Works if each component differs from Gaussian in the 4th moment, e.g., uniform over a cube.
  - Local optima via local search or a power iteration. [Nguyen-Regev]
  - Tensor view: After making the samples isotropic,

$$E(x \otimes x \otimes x \otimes x) = \sum_j (E(s_i^4) - 3) A_i \otimes A_i \otimes A_i \otimes A_i$$

- Fourier PCA [GVX13].
- Reweight x with Fourier weight $e^{iu^T x}$ for random unit vector u; then apply PCA; more generally, a robust tensor decomposition.

- Recursive FPCA [VX14].
- Partition using largest eigenvalue gap; recurse.

ICA Algorithm: Tensor decomposition of Fourier derivative tensors

- $\psi_x(u) = \log E\left(e^{iu^T x}\right)$
- $\nabla\psi_x(u) = \frac{E\left(ixe^{iu^T x}\right)}{E\left(e^{iu^T x}\right)} = \mu_u$
- $D^2(\psi_x(u)) = \frac{E\left((x-\mu_u)(x-\mu_u)^T e^{iu^T x}\right)}{E\left(e^{iu^T x}\right)}$
- If $x = As$, $D^2(\psi_x(u)) = A\ diag\left(\frac{\partial^2 \psi_j(A^T u)_j}{\partial (A^T u)_j^2}\right) A^T$
- More generally,

$$D^{2d}(\psi_x(u)) = \left(\otimes^d A\right) diag\left(\frac{\partial^{2d} \psi_j(A^T u)_j}{\partial (A^T u)_j^{2d}}\right) \left(\otimes^d A^T\right)$$

Tensor decomposition [GVX13]

- $$D^{2d}(\psi_x(u)) = \left(\otimes^d A\right) diag\left(\frac{\partial^{2d}\psi_j(A^T u)_j}{\partial(A^T u)_j^{2d}}\right)\left(\otimes^d A^T\right)$$
- Tensor decomposition needed to recover columns of A!
- Power iteration works if A is unitary, so $m \leq n$.
- Hard for one tensor, but with two such tensors generated with two random Fourier weights, we get

$$M_u = \sum_i \lambda_i \otimes^{2d} A_i, \qquad M_v = \sum_i \alpha_i \otimes^{2d} A_i$$

- Then compute eigenvectors of

$$M_u M_v^{-1} = \left(\otimes^d A\right) diag\left(\frac{\lambda_i}{\alpha_i}\right)\left(\otimes^d A^T\right)$$

- Need only that $\frac{\lambda_i}{\alpha_i}$ are distinct --- which holds whp.

Analysis

- Use Taylor decomposition

$$\psi_{x_j}(u_j) = \sum_k cum_k(x_j) \frac{(iu_j)^k}{k!}$$

- Truncate, analyze random Gaussian polynomials.

- Concentration and Anti-concentration

Crack my passwords

- GMAIL → MU47286
- AMAZON → RU27316
- IISC → LH857
- SHIVANI
- SHIVANI → HQ508526

Planted problems

- Problems over distributions. Base distribution is a random discrete structure, e.g., a random graph or a random Boolean formula.

- An unlikely substructure is planted, e.g., a large clique or a planted assignment --- the distribution is over structures random but subject to containing the planted substructure.

- Problem: Recover planted substructure.

Planted structures

- Planted clique: Start with a random graph. Add a clique of size $k \gg 2 \log n$ on some subset of k vertices.

  Find planted clique.

- Planted partition: Fix a partition of vertices of a graph. Pick random edges with different probabilities within parts and across parts.

  Recover planted partition.

- Planted assignment: Fix an assignment σ on Boolean variables. Generate a random formulas by picking clauses from a distribution that depends on σ.

  Recover planted assignment.

- Planted vector/subspace: Generate random points by adding a random vector from a fixed subspace to random (Gaussian) noise in full space.

  Recover planted vector subspace

Status: Planted Cliques

- Upper bounds: $n^{O(\log n)}$ for any $k > (2+\epsilon)\log n$
- Polynomial time for $k > c\sqrt{n}$

  [many]
- Lower bound: For $\epsilon > 0$, $k = n^{0.5-\epsilon}$, any statistical algorithm has complexity $n^{\Omega(\log n)}$

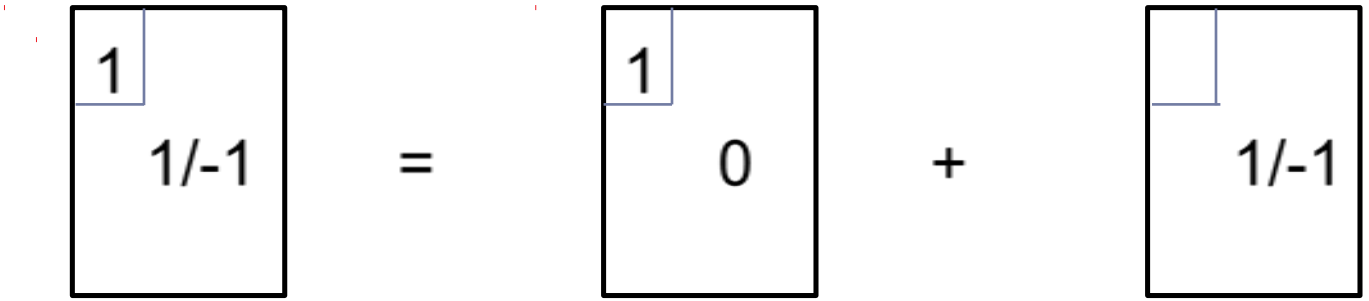
  [Grigorescu-Reyzin-Feldman-V.-Xiao13]
- (formally, this is for bipartite planted cliques, for which the same upper bounds apply)

- OP6: Is there a polytime algorithm for $k = O\left(\frac{\sqrt{n}}{\log n}\right)$?

Techniques

- Combinatorial:
- Remove lowest degree vertex iteratively [Feige]

- Spectral:
- Take highest components of principal component [AKS98]



Thm [Furedi-Komlos]. $|R|_2 \leq \big(2 + o(1)\big)\sqrt{n}.$

Status: Planted k-SAT/k-CSP

- Upper bound:
  Information theoretically, $O(n \log n)$ clauses suffice.
  Algorithmically, $n^{k/2} \log n$ clauses suffice
  [Bogdanov-Qiao-Applebaum, Feldman-Perkins-V.14]
  in time linear in number of clauses [FPV14].

- Bound is $n^{r/2}$ for (r-1)-wise independent clause distributions.

- Lower bound:
  $\left(\frac{n}{\log n}\right)^{r/2}$ clauses for statistical algorithms.[FPV14]

- OP7: Find more efficient nonstatistical algorithm for planted SAT.

Techniques

- Combinatorial + SDP for even k. [A12, BQ09]
- Subsampled power iteration: works for any k and a more general hypergraph planted partition problem:

  [FPV14]

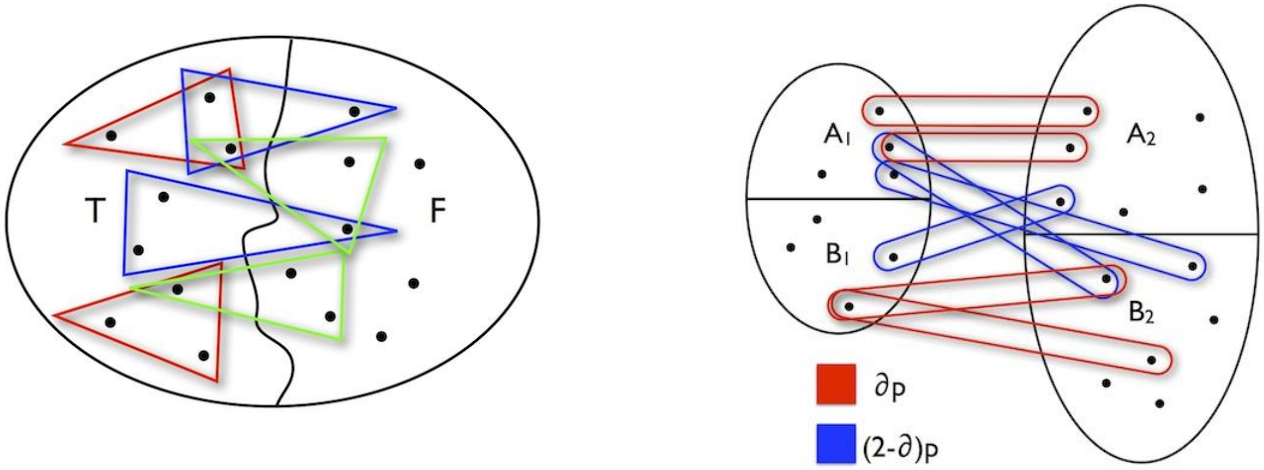
- Stochastic block theorem for k=2 (graph partition): a precise threshold on edge probabilities for efficiently recoverability.

  [Decelle-Krzakala-Moore-Zdeborova11]

  [Massoulie13, Mossel-Neeman-Sly13].

Algorithm: Subsampled Power Iteration

- Reduce to bipartite stochastic block model



- When k is odd, the norm of the noise dominates the signal, so usual analysis does not work!
- Form $n^{\lfloor k/2 \rfloor} \times n^{\lceil k/2 \rceil}$ matrix, sample into random submatrices, then use in a power iteration, starting with a random x^0. Keep track of signs of iterate x^i and take majority vote after $O(\log n)$ iterations.

Problems over distributions

- $\boldsymbol{D}$: set of distributions over domain X
- $\boldsymbol{F}$: set of solutions to problem
- $\boldsymbol{Z}: \boldsymbol{D} \rightarrow \boldsymbol{2^F}$: valid solutions for each input dist.

- Problem: given access to random samples from some distribution D in $\boldsymbol{D}$, find a solution f in $\boldsymbol{Z(D)}$.

- Average of a function: $E_D[f(x)]$
- Principal component: Find $max_{|u|=1}\ E_D[(u^T x)^2]$

- What fraction of input distribution satisfies property P?

- LP: $\max_{\mathrm{u}} E_{x\sim D}[f(x,u)]$ OR $\max_{\mathrm{u}} E_{a\sim D}[sgn(a^T u - b_a)]$

Statistical Algorithms

- Only access to the input distribution: compute arbitrary functions on random samples OR estimate their expectations to within a given tolerance.

- For any $f: X \to [0,1]$, STAT(τ) outputs $E\big(f(x)\big) \pm \tau$. [Kearns]
- For any $f: X \to \{0,1\}$, 1-STAT outputs f(x) for a random x.
- VSTAT(t): outputs $E_D[f(x)]$ to within the standard deviation of t random samples.

- MSTAT(L): outputs $f: X \to \{0,1, \dots, L-1\}$ on a random x.

- Complexity of algorithm = number of calls to oracle.

Can statistical algorithms detect planted structures?

- All our previous algorithms can be implemented statistically:
  - Small/large degree
  - Local search
  - Principal component (power iteration)
  - Markov Chain Monte Carlo / simulated annealing
  - Gradient descent

$$\nabla_x E_u[f(x,u)] = E_u[\nabla_x f(x,u)]$$

  - Linear programs, conic programs, stochastic optimization

- With one notable exception: Gaussian Elimination

Idea: lots of very different instances

- One probability distribution per parity function
- One probability distribution for each possible planted clique subset of size k
- One distribution for each planted assignment

- Each oracle query reveals significant information only about a small fraction of distributions

Correlation of distributions

- base distribution D (typically uniform)
- $f, g: X \to R, \quad \langle f, g \rangle_D = E_D[f(x)g(x)]$
- Correlation: $\rho(D_1, D_2) = \langle \frac{D_1}{D} - 1, \frac{D_2}{D} - 1 \rangle_D$
- Average correlation of a set of distributions:

$$\rho(D', D) = \frac{1}{|D'|^2} \sum_{D_1, D_2 \in D'} |\rho(D_1, D_2)|.$$

Statistical dimension I

$$\forall f \in F, \exists D_f = \{D_1, D_2, \ldots, D_m\} \subseteq \widehat{D} \setminus Z^{-1}(f)$$

s.t. $$\rho(D_i, D_j) \leq \begin{cases} \beta, & i = j \\ \gamma, & i \neq j \end{cases}$$

then, $SD(Z, \gamma, \beta) = m.$

Thm. Any statistical algorithm needs $SD(Z, \gamma, \beta)$ queries to STAT(γ).

Finding parity functions [Kearns, Blum et al]

- Each f in F is a parity function of n variables.
- D: uniform over Boolean vectors
- D_f: uniform over vectors with same parity
- For each f, there is only one matching D_f.
- $\rho(D_f, D_g) = \begin{cases} 1, f = g \\ 0, f \neq g \end{cases}$
- SD(parity, 0,1) = $2^n - 1$.
- Finding a parity needs exponential time for any statistical algorithm! (also holds for noisy parity.)

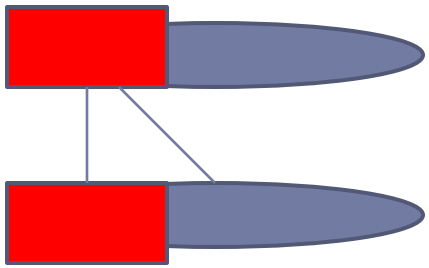
Bipartite planted clique

- Fix clique subset S, with |S|=k.
- Each D_S is a distribution on vectors $x \in \{0,1\}^n$:

- For $i \notin S, x_i = 0/1$ randomly. For S,

  with probability $\frac{k}{n}$, $\forall i \in S, x_i = 1$ for $i \in S$.

  w. p. $1 - \frac{k}{n}$, $\forall i \in S, x_i = 0/1$ randomly



- Problem: find the planted subset S.

What about planted clique?

- Distributions: one per k-subset
- Lots. But not as uncorrelated.

- $\rho(D_S, D_T) \leq \frac{2^\lambda k^2}{n^2}$ $\lambda = |S \cap T|$.

- "most" pairs of distributions are far, but not all.
- Could look for subset of pairwise "far" distributions. Not too many.
- Gives a lower bound for $k = O(\log n)$

Statistical Dimension II

- $$\forall f \in F, \exists D_f \subseteq \widehat{D} \setminus Z^{-1}(f),$$

For every large subset: $\forall D' \subset \widehat{D}, |D'| \geq \frac{|D_f|}{d}$,

Avg correlation is small: $\rho(D', D\ ) \leq \bar{\gamma}$.

then, $SD(Z, \bar{\gamma}) = d$.

Thm1. Any statistical algorithm needs d queries to STAT($\sqrt{\bar{\gamma}}$) or VSTAT($\frac{1}{3\bar{\gamma}}$).

Stat dim of planted cliques

- Thm [GRFVX13]. SD(planted clique, $\bar{\gamma} = \frac{2^{l+2}k^2}{n^2}$) $\geq n^{2l\delta}$

- Cor. For $\delta > 0$, $k < n^{0.5-\delta}$, any statistical algorithm needs at least $n^{\Omega(\log n)}$ queries to VSTAT($\frac{n^{2-\delta}}{k^2}$).

- (for $k < n^{0.5-\delta}$, this is more than n samples for each query!)

Statistical dimension III

- Discrimination norm for base distribution D, set of distributions D' over domain X, real-valued functions h.

$$\kappa(D', D) = \max_{h:|h|=1} E_{D_1 \sim D}\left[\left|E_{D_1}[h] - E_D[h]\right|\right]$$

- SD(P,κ) = d, largest integer for which there exists a set of distributions $\widehat{D}$ s.t. for any subset $D' \subset \widehat{D}$:

$$|D'| \geq \frac{|\widehat{D}|}{d}, \kappa(D', D) \leq \kappa.$$

- Thm. Any stat algo need $\Omega(d/L)$ calls to MSTAT(L).

Complexity of Planted k-SAT/k-CSP

- Distribution complexity of clause distribution Q: largest integer r for which Q is (r-1)-wise independent.
- Alternatively, smallest r which Q has a nonzero Fourier coefficient of size r.
- $1 \leq r(\mathrm{k\,SAT}) \leq k$
- Thm. SD(planted k-SAT, $\frac{(\log n)^r}{n^{r/2}}$) $\geq n^{\Omega(\log n)}$.
- No single query h can rule out a large fraction of assignments.

- Discrete Fourier Analysis,
- Boolean polynomial concentration.

Detecting planted solutions

- Many interesting problems
- Potential for novel algorithms
- New computational lower bounds
- Open problems in both directions!

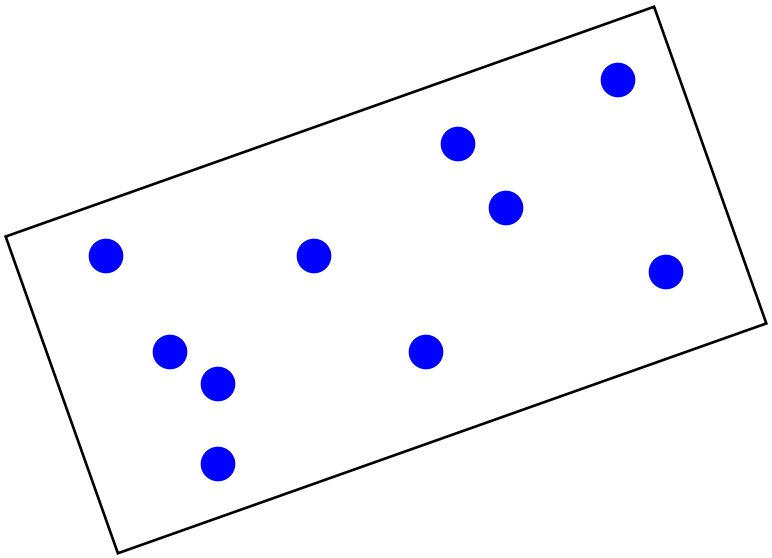
Coming soon: The Password Game!

- GMAIL → MU47286
- AMAZON → RU27316
- IISC → LH857
- SHIVANI → HQ508526
- UTHAPAM
- UTHAPAM → AX010237

Thank you!

A toy problem

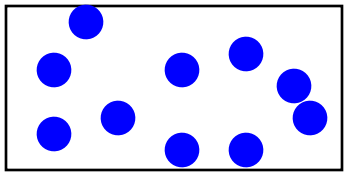
- Problem: Given samples from a stretched cube in R^n that rotated in an unknown way, find the long direction.



- Solution: Top principal component.

Malicious Noise

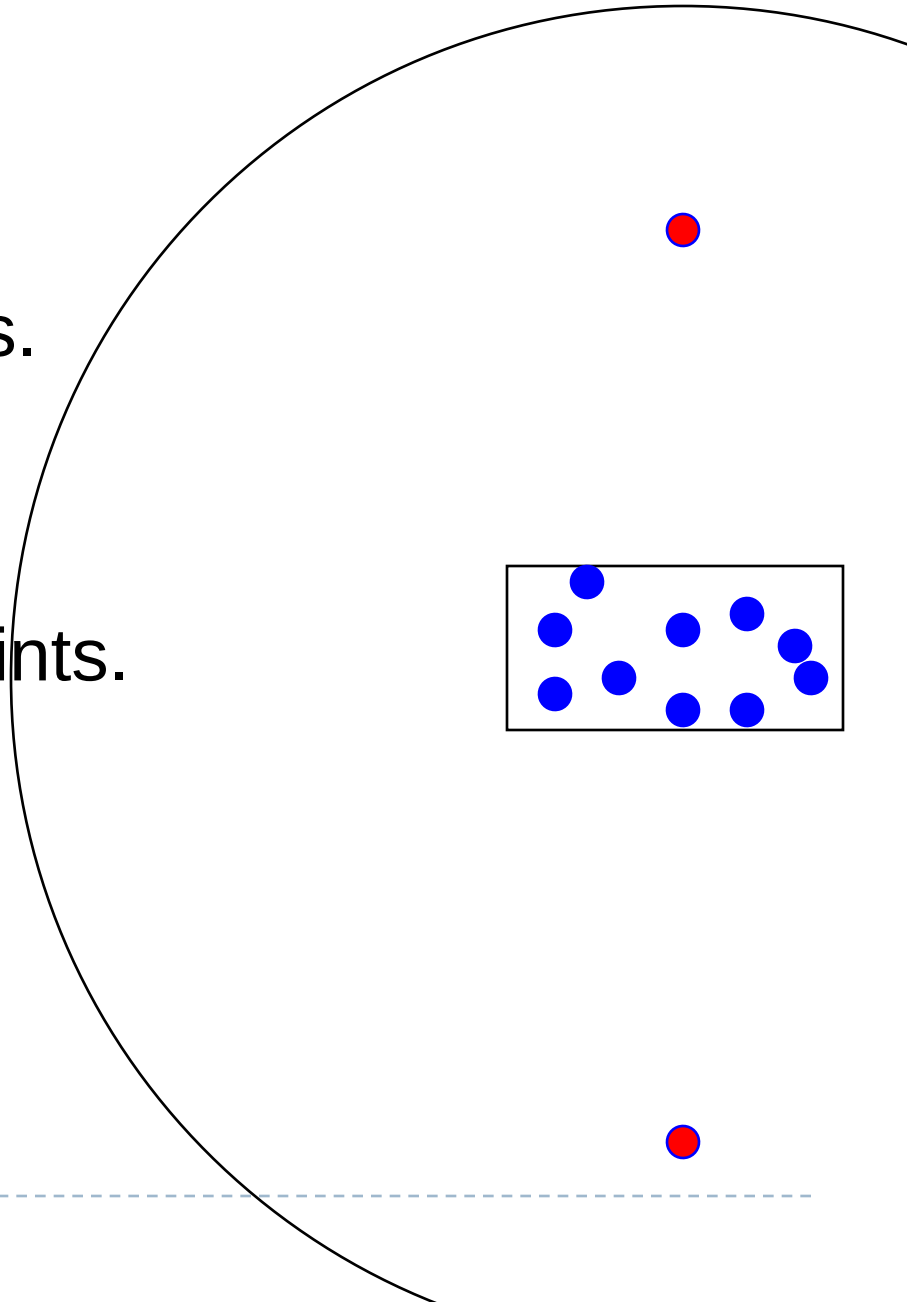
- Suppose $E[x_1^2] = 2$ and $E[x_i^2] = 1$.
- Adversary puts a $\frac{1}{n}$ fraction of points at $\sqrt{(n+1)}e_2$
- Now, $E[x_1^2] < E[x_2^2]$
- And e_2 is the top principal component!



Malicious Noise

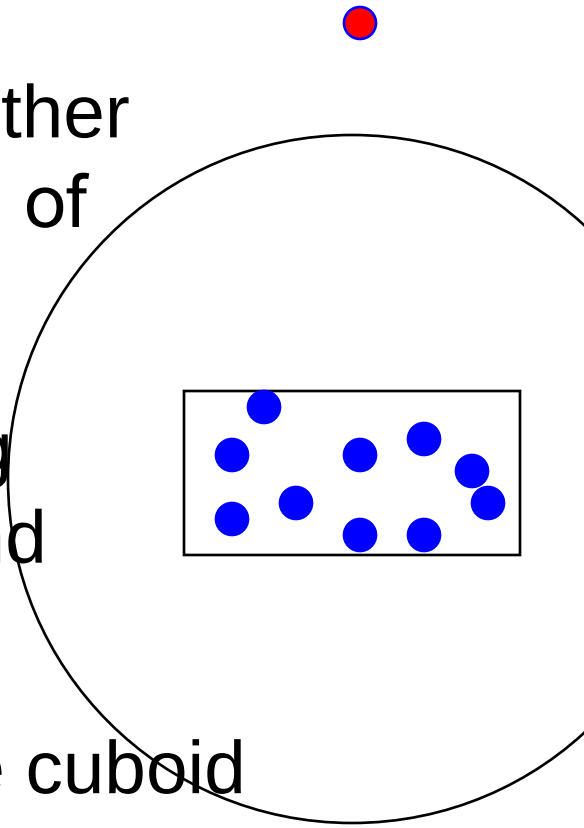
Easy to remove noise? No!
Consider pairwise distances.

$E(||x||^2) = n+1$ for cuboid points.
Same as noisy points…



Malicious Noise

• Adversary can play same trick in k other directions e_3…, but needs k/n fraction of samples.



• If ϵ is small, then e_1 won't be among smallest n/2 principal components and they can be projected out.

• After two rounds, furthest pair in the cuboid at distance $\sqrt{3(n/4+1)}$.

• Now we can put a ball around the good data!