# Nearly-tight sample complexity bounds for learning mixtures of Gaussians

**Hassan Ashtiani**
**McMaster**

**Shai Ben-David**
**Waterloo**

**Nick Harvey**
**UBC**

**Abbas Mehrabian**
**McGill**

**Yaniv Plan**
**UBC**

Chris Liaw (UBC)

SFU Theory Seminar, May 2019

# Distribution Learning

Goal: Given data from some distribution $\mathcal{D}$, estimate $\mathcal{D}$.

III. *Contributions to the Mathematical Theory of Evolution.*

By KARL PEARSON, *University College, London.*

*Communicated by Professor* HENRICI, *F.R.S.*

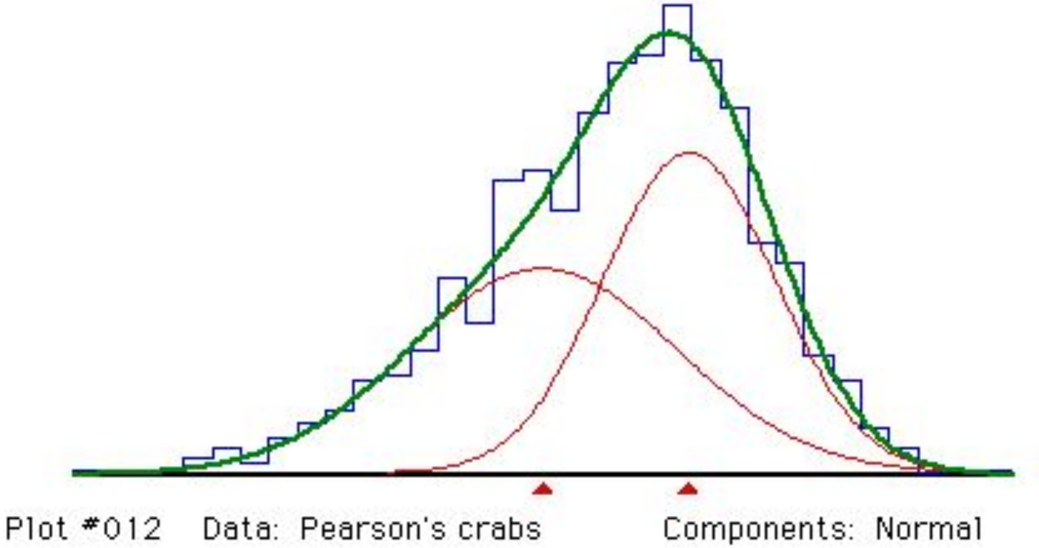
Received October 18,—Read November 16, 1893.

(9.) The whole method may be illustrated by the following numerical example :—

*Breadth of "Forehead" of Crabs.*—Professor W. F. R. WELDON has very kindly given me the following statistics from among his measurements on crabs. They are for 1000 individuals from Naples. The abscissæ of the curve are the ratio of "fore-

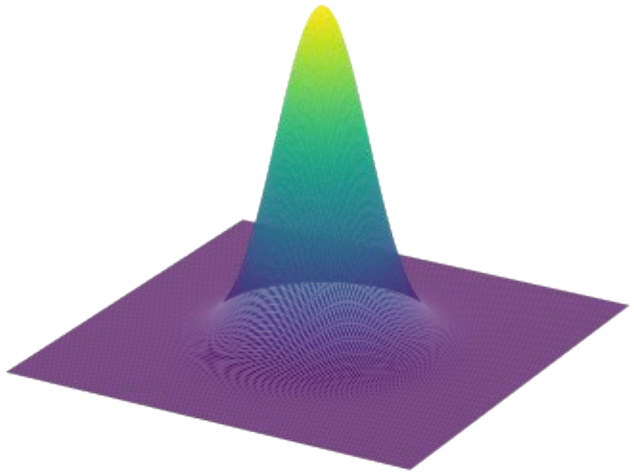
# Distribution Learning

These two normal curves were now drawn by aid of the Table II., which was calculated afresh for this purpose from the exponential.* These curves are plotted out in fig. 1, and their ordinates added together give the resultant curve. It will be seen that this curve is in remarkably close agreement with the original asymmetrical frequency-curve, an agreement quite as close as we could reasonably expect from the comparative smallness of the number of individuals dealt with, and the resulting fact




(Plot due to Peter Macdonald)

# Gaussians and Mixtures of Gaussians

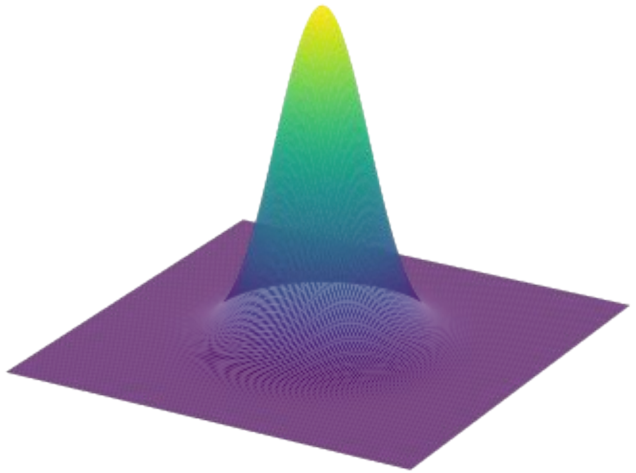


Single Gaussian in $\mathbb{R}^d$ specified by:

- Mean $\mu \in \mathbb{R}^d$ and;
- Covariance matrix $\Sigma \in \mathbb{R}^{d \times d}$

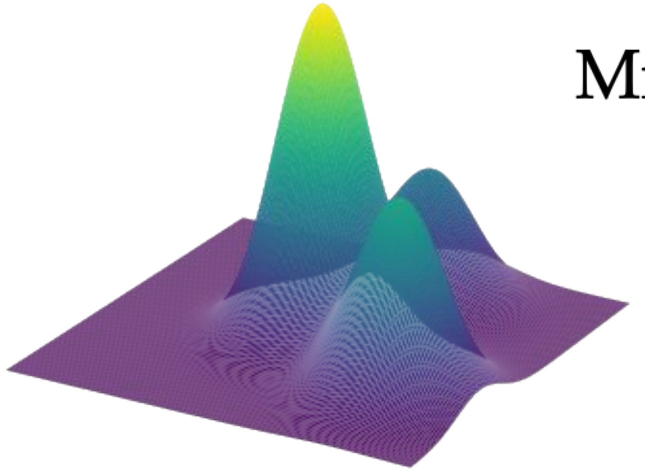
$$\mathcal{N}(\mu, \Sigma)(x) = \frac{1}{\sqrt{2\pi \det(\Sigma)}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right)$$

# Gaussians and Mixtures of Gaussians



Single Gaussian in $\mathbb{R}^{\boldsymbol{d}}$ specified by:

- Mean $\mu \in \mathbb{R}^{\boldsymbol{d}}$ and;
- Covariance matrix $\Sigma \in \mathbb{R}^{\boldsymbol{d} \times \boldsymbol{d}}$



Mixtures of $k$ Gaussians are distributions of the form

$\sum_{i=1}^{k} w_i \mathcal{N}(\mu_i, \Sigma_i)$ $\quad w_i \geq 0; \sum_{i=1}^{k} w_i = 1$

$\mu_i \in \mathbb{R}^{\boldsymbol{d}}, \Sigma_i \in \mathbb{R}^{\boldsymbol{d} \times \boldsymbol{d}}$

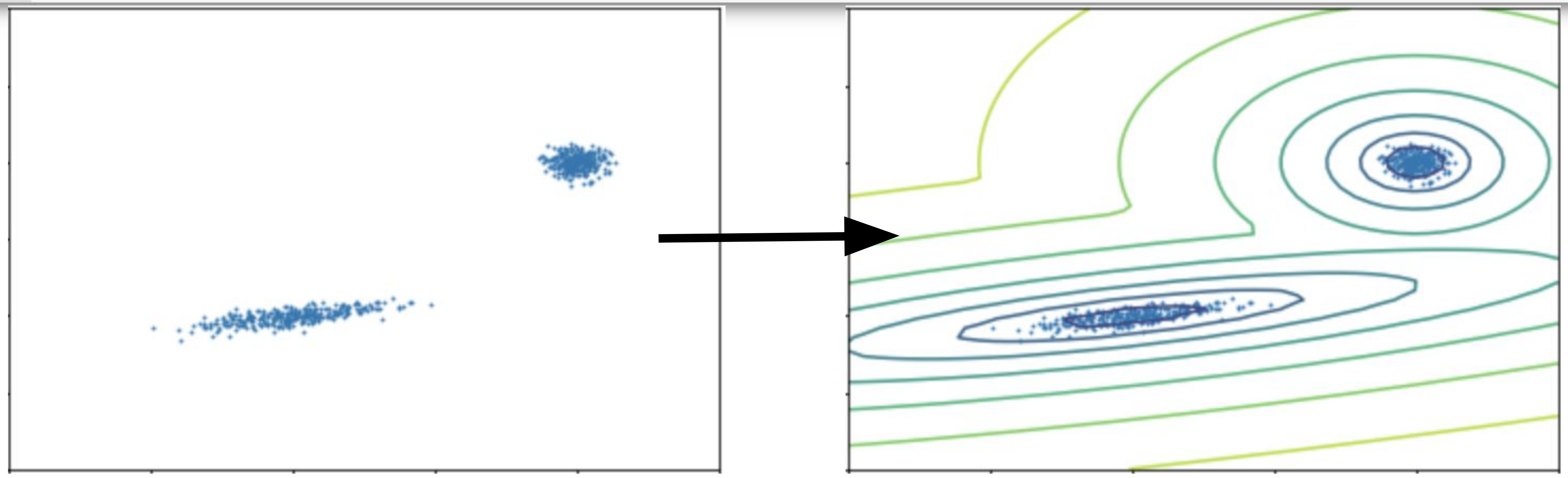
↑ Mixing weights

# Mixtures of Gaussians

- Very classical and a universal approximator.
- Algorithms widely implemented in many software packages.

```
from sklearn import mixture

# fit a Gaussian Mixture Model with two components
clf = mixture.GaussianMixture(n_components=2, covariance_type='full')
clf.fit(X_train)  # X_train is training data
```



# What does it mean to learn?

| Objective | Approach | Downsides |
|---|---|---|
| Maximum likelihood (non-convex objective) | Expectation-maximization [Dempster, Laird, Rubin ‘77] | • Lack of guarantees |

# What does it mean to learn?

| Objective | Approach | Downsides |
| --- | --- | --- |
| Maximum likelihood (non-convex objective) | Expectation-maximization [Dempster, Laird, Rubin '77] | • Lack of guarantees |
| Parameter estimation | Method of moments [Dasgupta '99; Moitra, Valiant '10] | • Requires structural assumptions<br>• Requires exponential number of samples |

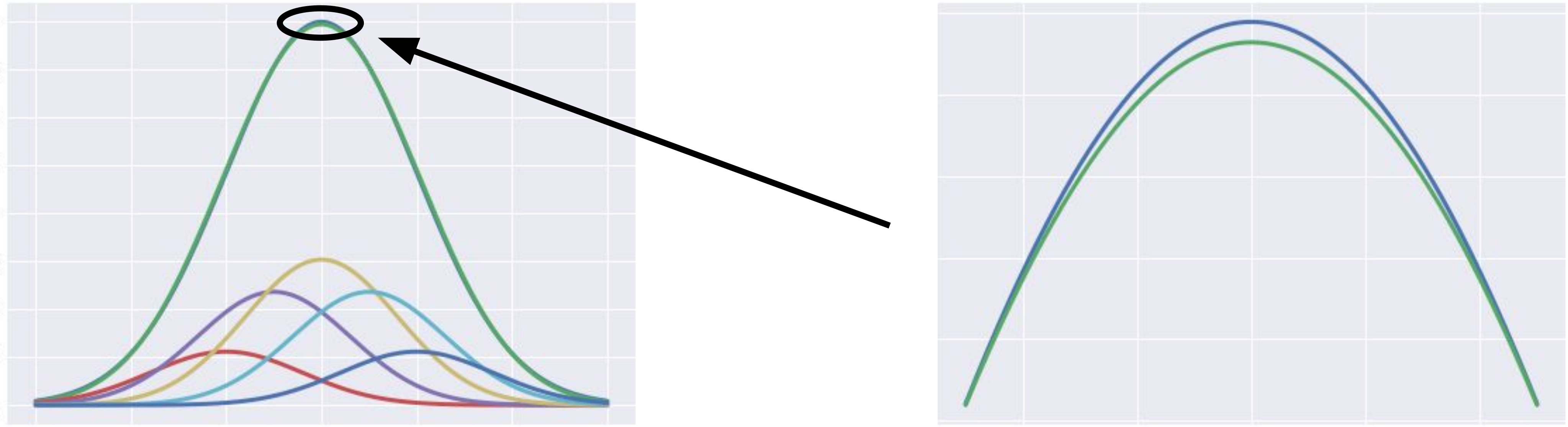
# Parameter estimation

- Goal: estimate mean, covariance matrices, and mixing weights.

✘ Requires structural assumptions.
  - e.g. Two nearly overlapping Gaussians.

✘ Difficult to even differentiate between 1 or $k$ Gaussians.

✘ Problem requires $\exp(\Omega(k))$ samples. [Moitra, Valiant ‘10]



# What does it mean to learn?

| Objective | Approach | Downsides |
| --- | --- | --- |
| Maximum likelihood (non-convex objective) | Expectation-maximization [Dempster, Laird, Rubin '77] | • Lack of guarantees |
| Parameter estimation | Method of moments [Dasgupta '99; Moitra, Valiant '10] | • Requires structural assumptions<br>• Requires exponential number of samples |
| | Our focus | |

*We will make *no* structural assumptions.

# Density Estimation

Suppose $f$ is an unknown mixture of $k$ Gaussians in $\mathbb{R}^d$.
How many i.i.d. samples are sufficient to return $\hat{f}$ s.t. $d_{TV}(f, \hat{f}) \le \epsilon$?
Call this the **sample complexity**.

$$d_{TV}(f, \hat{f}) = \sup_E \left\{ \Pr_f[E] - \Pr_{\hat{f}}[E] \right\} = \frac{1}{2} \int \left| f(x) - \hat{f}(x) \right| \mathrm{d}x$$

**Previous results.**

- $k = 1$: $O(d^2/\epsilon^2)$ [Folklore]
- $d = 1$: $\tilde{O}(k/\epsilon^2)$ [Chan, Diakonikolas, Servedio, Sun '14]
- Q: Number of samples for general $d$, $k$?
  - $\tilde{O}(kd^2/\epsilon^4)$ [Ashtiani, Ben-David, Mehrabian '17]
  - $\Omega(kd/\epsilon^2)$ [Suresh, Orlitsky, Acharya, Jafarpour '14]

# Why Total Variation?

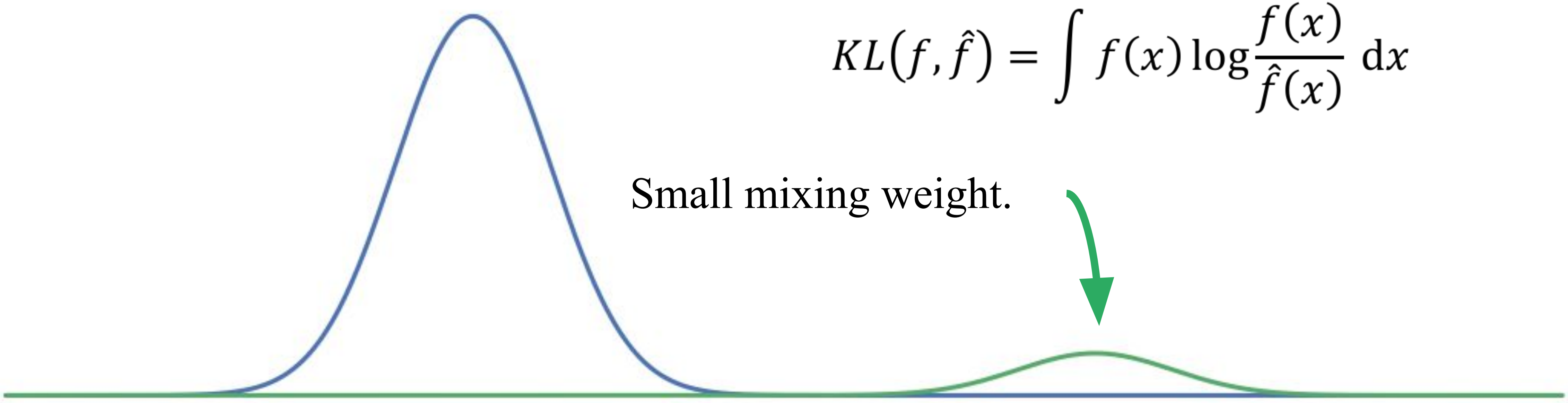
What if we wanted to focus on KL divergence instead?

**Lemma.** For mixtures of **two** Gaussians, it is ***impossible*** to have an algorithm that draws at most $M < \infty$ samples from $f \in \mathcal{F}$ and returns $\hat{f}$ such that $\mathrm{KL}(f, \hat{f}) \leq 10^{10}$.

$$^*KL(f, \hat{f}) = \int f(x) \log \frac{f(x)}{\hat{f}(x)}\, \mathrm{d}x$$

# Why Total Variation?

$$KL(f, \hat{f}) = \int f(x) \log \frac{f(x)}{\hat{f}(x)} \, \mathrm{d}x$$




- If **green** component is too light then no algorithm will sample it.
- Any "reasonable" algorithm returns **blue** component.
  - TV distance is close to 0.
  - KL divergence is $\approx \infty$!

# Why Total Variation?

What if we wanted to focus on KL divergence instead?

**Lemma.** For mixtures of **two** Gaussians, it is ***impossible*** to have an algorithm that draws at most $M < \infty$ samples from $f \in \mathcal{F}$ and returns $\hat{f}$ such that $\mathrm{KL}(f, \hat{f}) \leq 10^{10}$.

Fine.. what about other $L_p$-norms $(p > 1)$?

**Lemma.** For mixtures of **two** Gaussians, it is ***impossible*** to have an algorithm that draws at most $M < \infty$ samples from $f \in \mathcal{F}$ and returns $\hat{f}$ such that $\|f - \hat{f}\|_p \leq 10^{10}$.

# Main Result

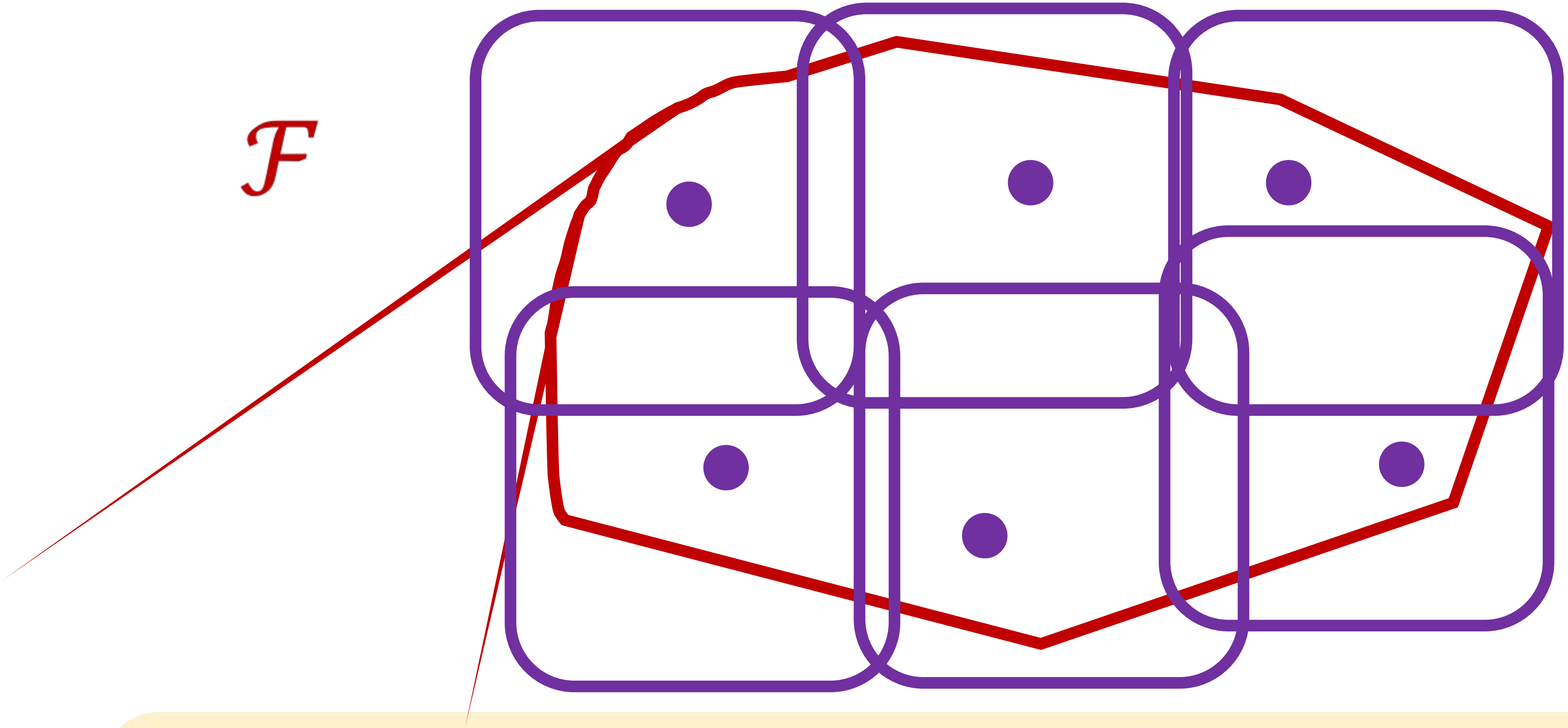
**Theorem** [ABHLMP '18] Sample complexity for learning mixtures of $k$ Gaussians in $\mathbb{R}^d$ (up to $d_{TV}$-error $\epsilon$) is

$$\widetilde{\boldsymbol{\Theta}}\left(\frac{\boldsymbol{kd^2}}{\boldsymbol{\epsilon^2}}\right)$$

$\widetilde{\Theta}(\cdot)$ hides polylog factors

- *No* structural assumptions.
- Upper bound proof is via a **compression** argument.
- Lower bound proof is information theoretic.

# Covering Arguments




**Lemma** [Yatracos '85] Suppose $\mathcal{F}$ is a class of densities and there exists densities $f_1, \ldots, f_M$ such that $\min_i d_{TV}(f_i, f) \leq \epsilon$ for all $f \in \mathcal{F}$. Then sample complexity to learn $\mathcal{F}$ is $O(\log M/\epsilon^2)$.

# Covering Arguments

**Lemma** [Yatracos '85] Suppose $\mathcal{F}$ is a class of densities and there exists densities $f_1, \ldots, f_M$ such that $\min_i d_{TV}(f_i, f) \leq \epsilon$ for all $f \in \mathcal{F}$. Then sample complexity to learn $\mathcal{F}$ is $O(\log M/\epsilon^2)$.

**Sketch.** Let $f$ be the unknown density.

- Let $E_{ij}$ be such that $\Pr_{f_i}[E_{ij}] - \Pr_{f_j}[E_{ij}] = d_{TV}(f_i, f_j)$.
- Consider a "tournament" where $f_i$ beats $f_j$ if

$$\left|\Pr_{f_i}[E_{ij}] - \Pr_{f}[E_{ij}]\right| + \epsilon \leq \left|\Pr_{f_j}[E_{ij}] - \Pr_{f}[E_{ij}]\right|.$$

- If $d_{TV}(f_j, f) \leq \epsilon$ then $f_j$ is never beaten.
- If $d_{TV}(f_i, f) > 10\epsilon$ then $f_j$ beats $f_i$.
- Any $f_i$ that is never beaten satisfies $d_{TV}(f_i, f) \leq 10\epsilon$.

# Covering Arguments

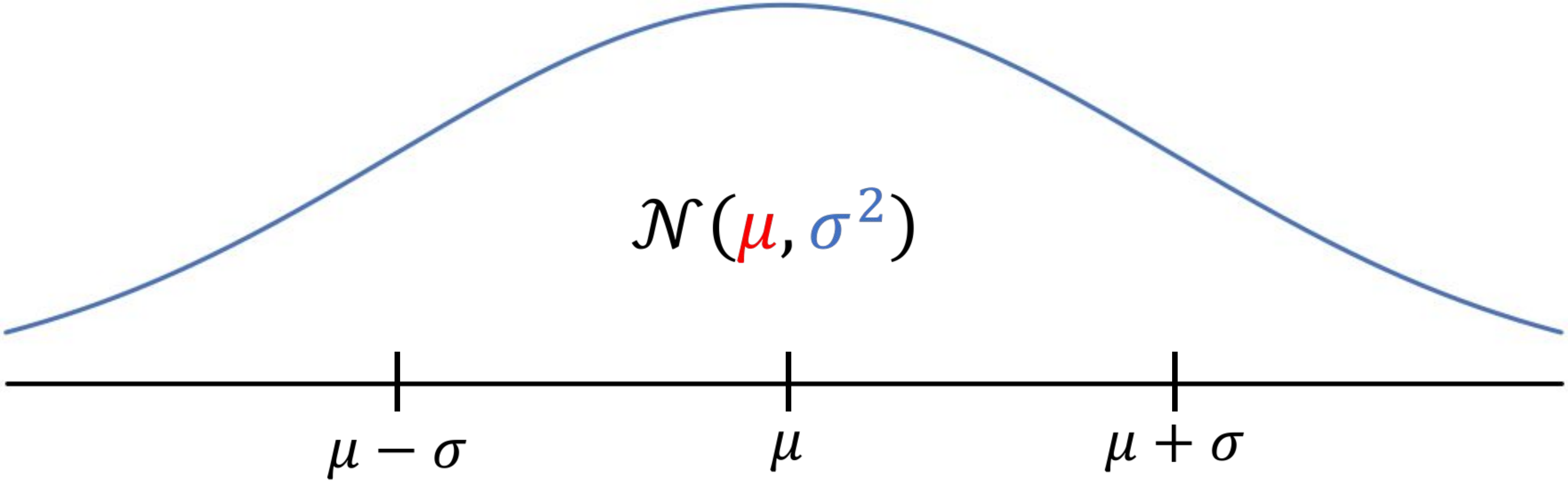
**Lemma** [Yatracos '85] Suppose $\mathcal{F}$ is a class of densities and there exists densities $f_1, \ldots, f_M$ such that $\min_i d_{TV}(f_i, f) \leq \epsilon$ for all $f \in \mathcal{F}$. Then sample complexity to learn $\mathcal{F}$ is $O(\log M/\epsilon^2)$.

**Problem**: This does not work for Gaussians.

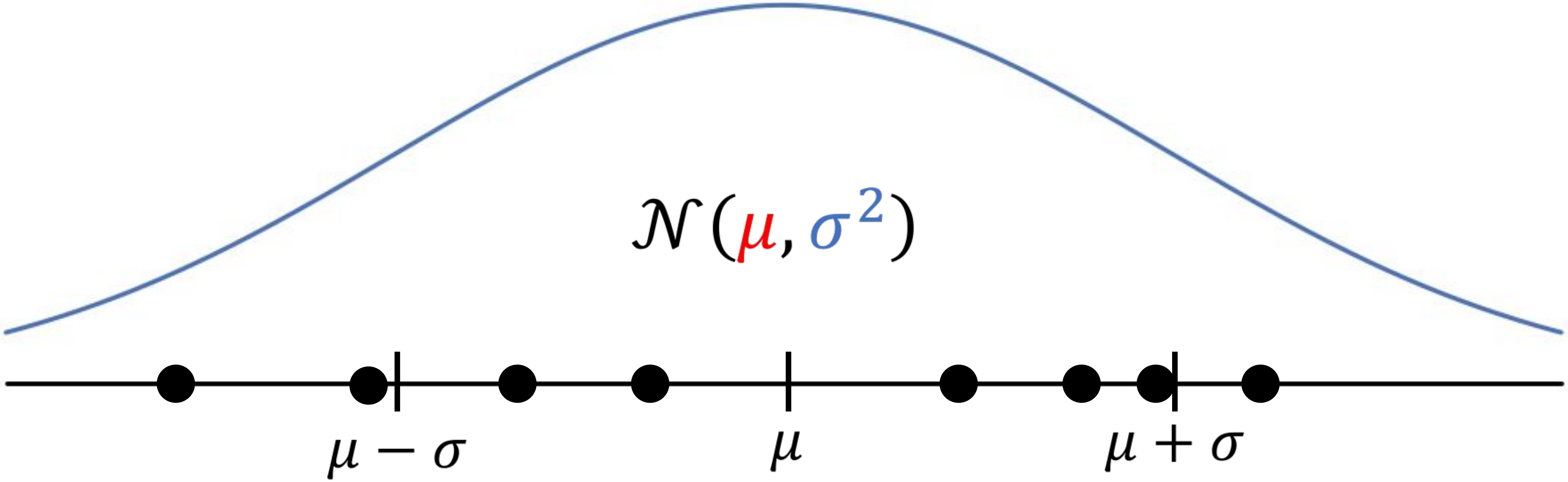
- Even 1D Gaussians do ***not*** have a finite cover.

**Solution**: First, ***look*** at the data and then construct a small cover.

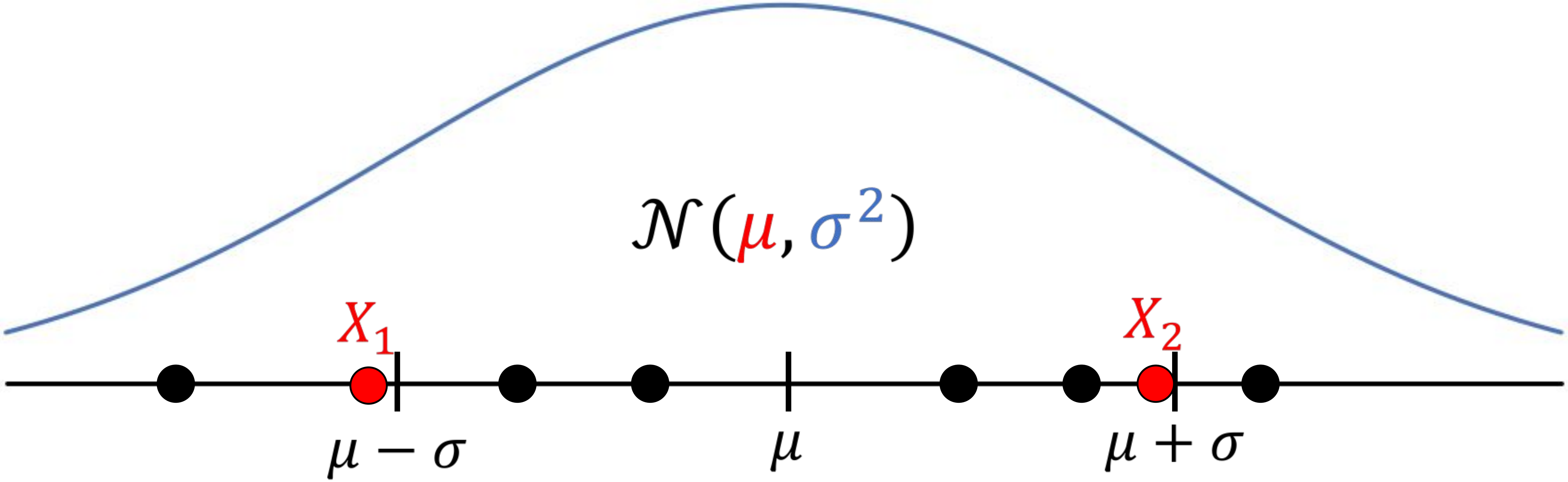
# Compressing Gaussians in $\mathbb{R}$



# Compressing Gaussians in $\mathbb{R}$



# Compressing Gaussians in $\mathbb{R}$

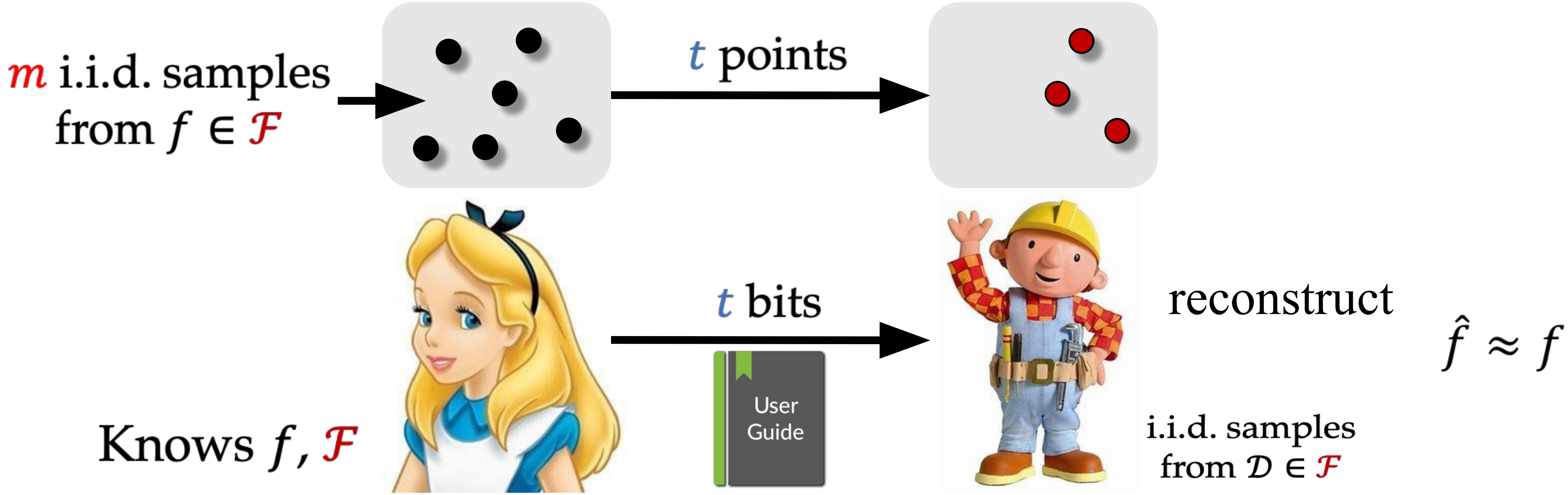



$$\frac{X_2 + X_1}{2} \approx \mu \qquad \frac{X_2 - X_1}{2} \approx \sigma$$

- **Two samples** are **sufficient** to **encode** $\mathcal{N}(\mu, \sigma^2)$.

# Compression Framework

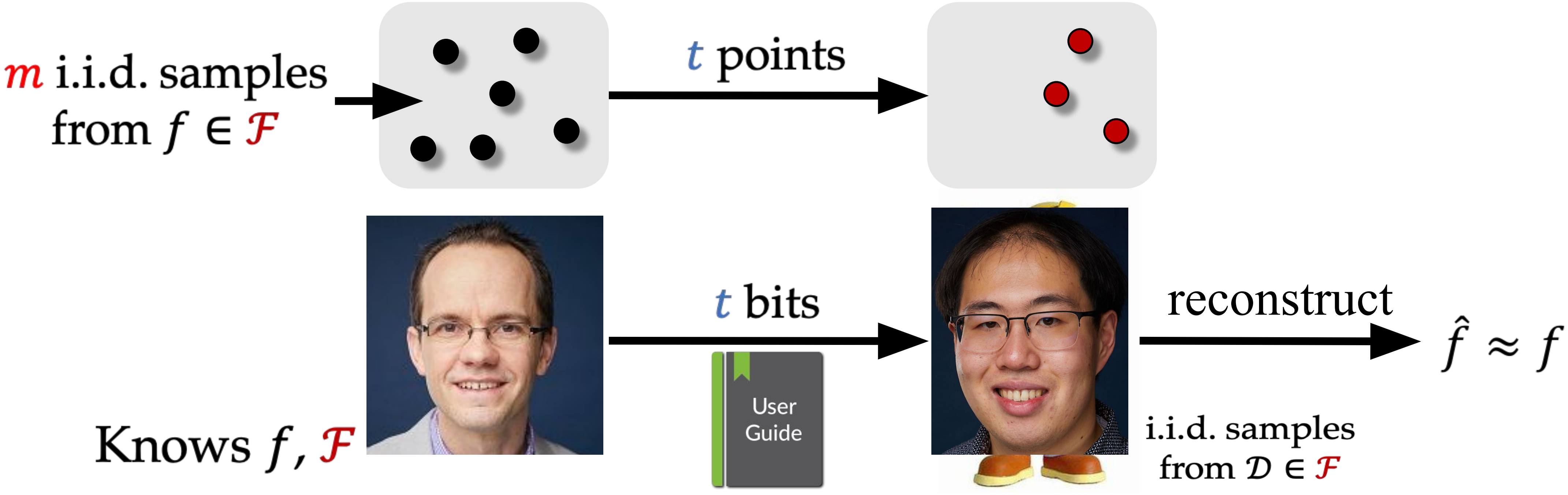
$\mathcal{F}$: a class of densities (e.g. Gaussians)




If Alice draws $m(\epsilon)$ samples, sends $t(\epsilon)$ points & bits, and $d_{TV}(f, \hat{f}) < \epsilon$ then we say $\mathcal{F}$ admits $(m(\epsilon), t(\epsilon))$-compression.

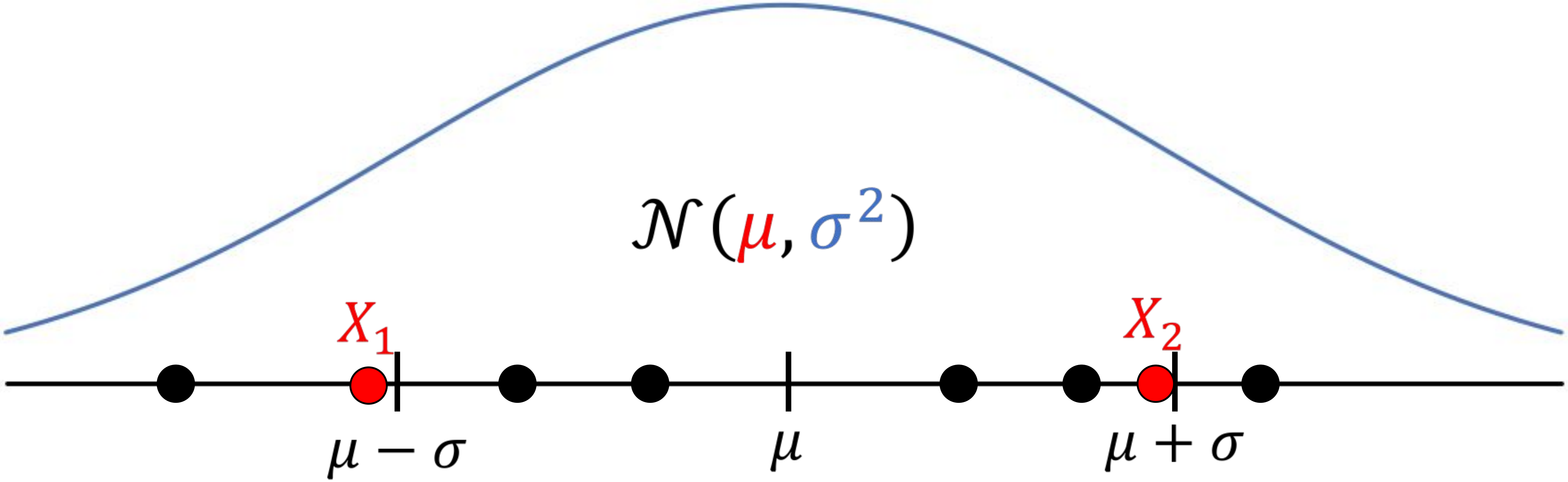
# Compression Framework

$\mathcal{F}$: a class of densities (e.g. Gaussians)




If Alice draws $m(\epsilon)$ samples, sends $t(\epsilon)$ points & bits, and $d_{TV}(f,\hat{f}) < \epsilon$ then we say $\mathcal{F}$ admits $(m(\epsilon), t(\epsilon))$-compression.

# Compressing Gaussians in $\mathbb{R}$




$$\frac{X_2 + X_1}{2} \approx \mu \qquad \frac{X_2 - X_1}{2} \approx \sigma$$

1D Gaussians admit $(O(1/\epsilon), 2)$-compression.

# Compression Theorem

**Theorem** [ABHLMP '18] Suppose $\mathcal{F}$ admits $(m(\epsilon), t(\epsilon))$-compression. Then sample complexity to learn $\mathcal{F}$ (up to $d_{TV}$-error $\epsilon$) is

$$\widetilde{O}\left(m(\epsilon) + \frac{t(\epsilon)}{\epsilon^2}\right).$$

$\widetilde{O}(\cdot)$ hides polylog factors

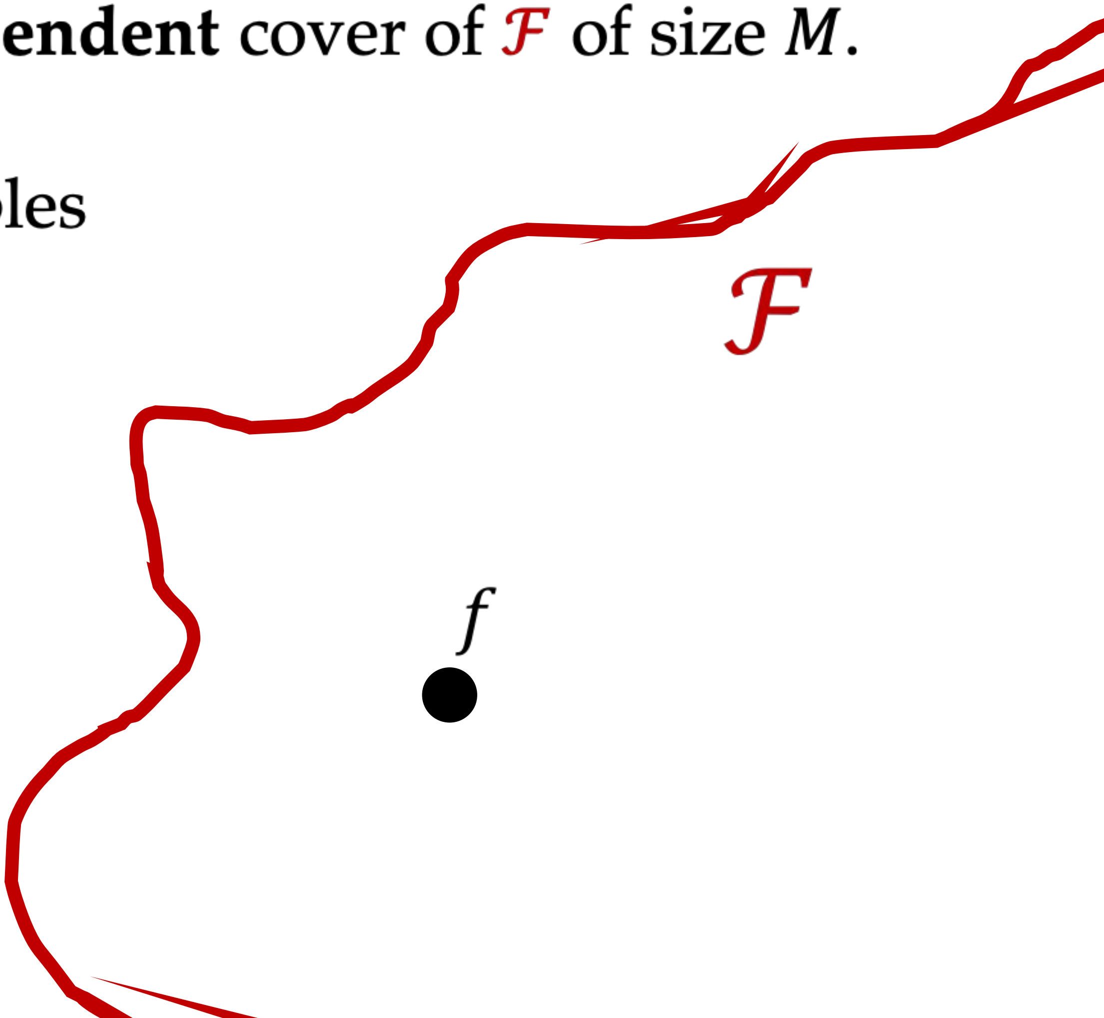
**Small compression schemes** imply **sample-efficient** algorithms.

# Proof of Compression Theorem

- We cannot implement Alice, but we can implement Bob!
- We draw $m(\epsilon)$ i.i.d. samples from $f$ and try all $M \leq (2m(\epsilon))^{t(\epsilon)}$ possible inputs to Bob to get a **data-dependent** cover of $\mathcal{F}$ of size $M$.

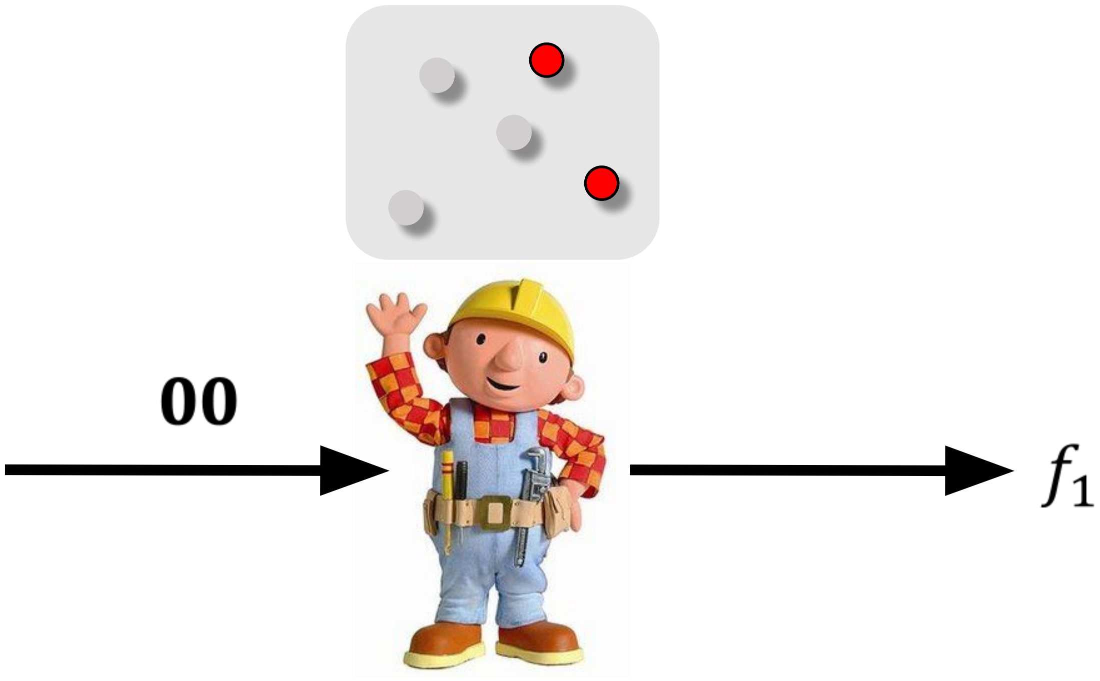
$m(\epsilon)$ i.i.d. samples from $f \in \mathcal{F}$

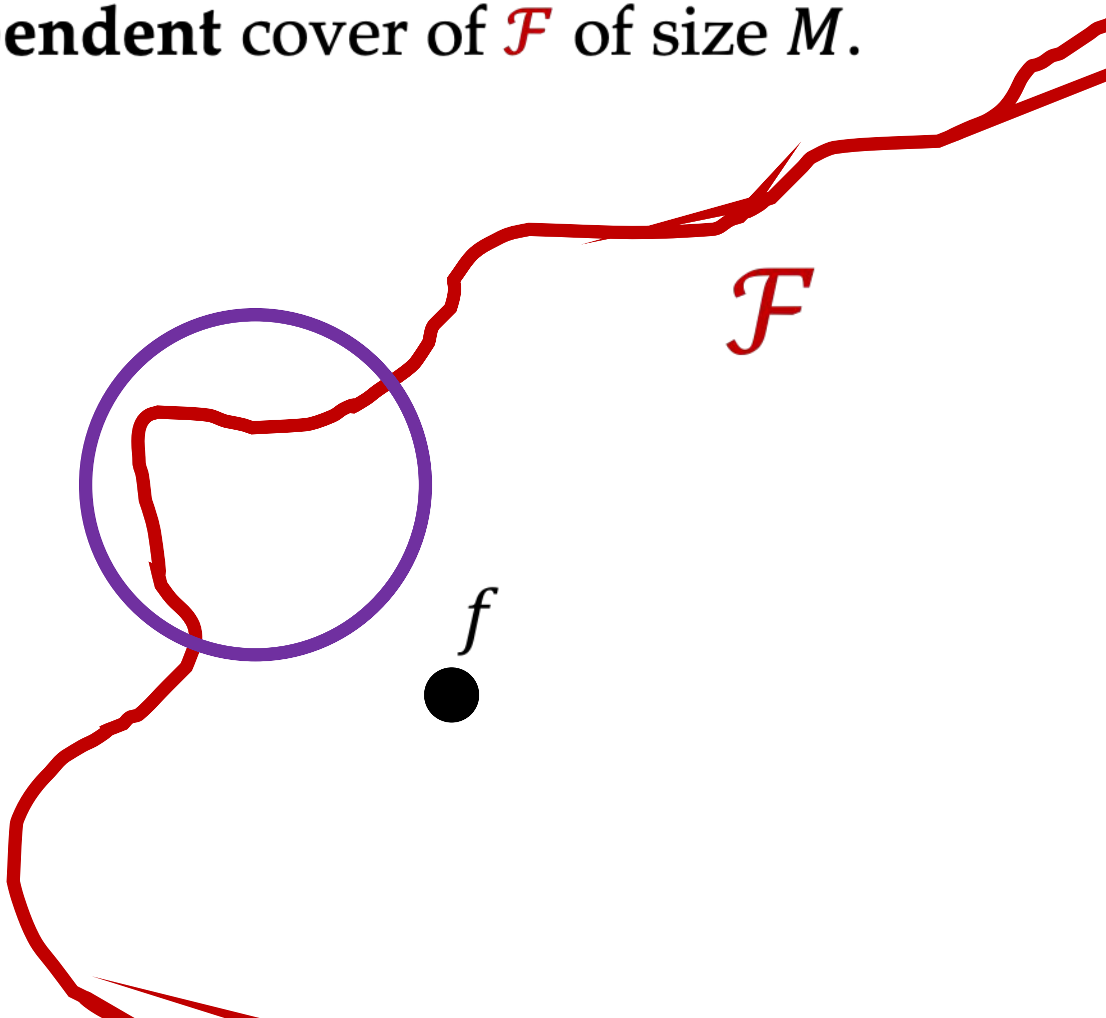
Knows $\mathcal{F}$



# Proof of Compression Theorem

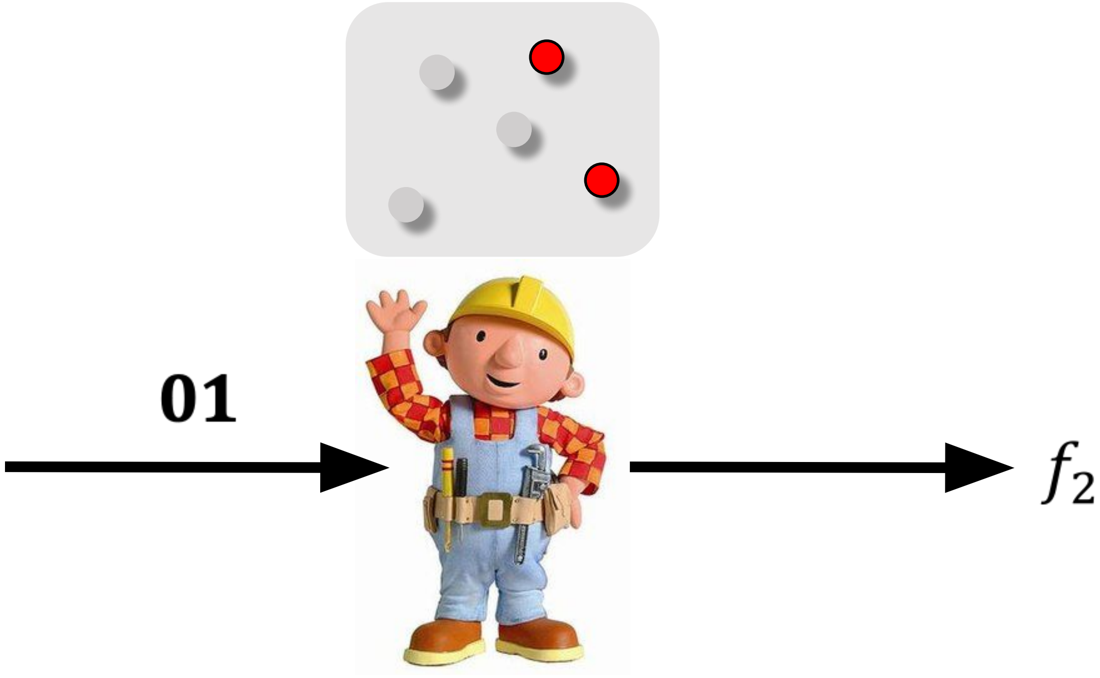
- We cannot implement Alice, but we can implement Bob!
- We draw $m(\epsilon)$ i.i.d. samples from $f$ and try all $M \leq (2m(\epsilon))^{t(\epsilon)}$ possible inputs to Bob to get a **data-dependent** cover of $\mathcal{F}$ of size $M$.

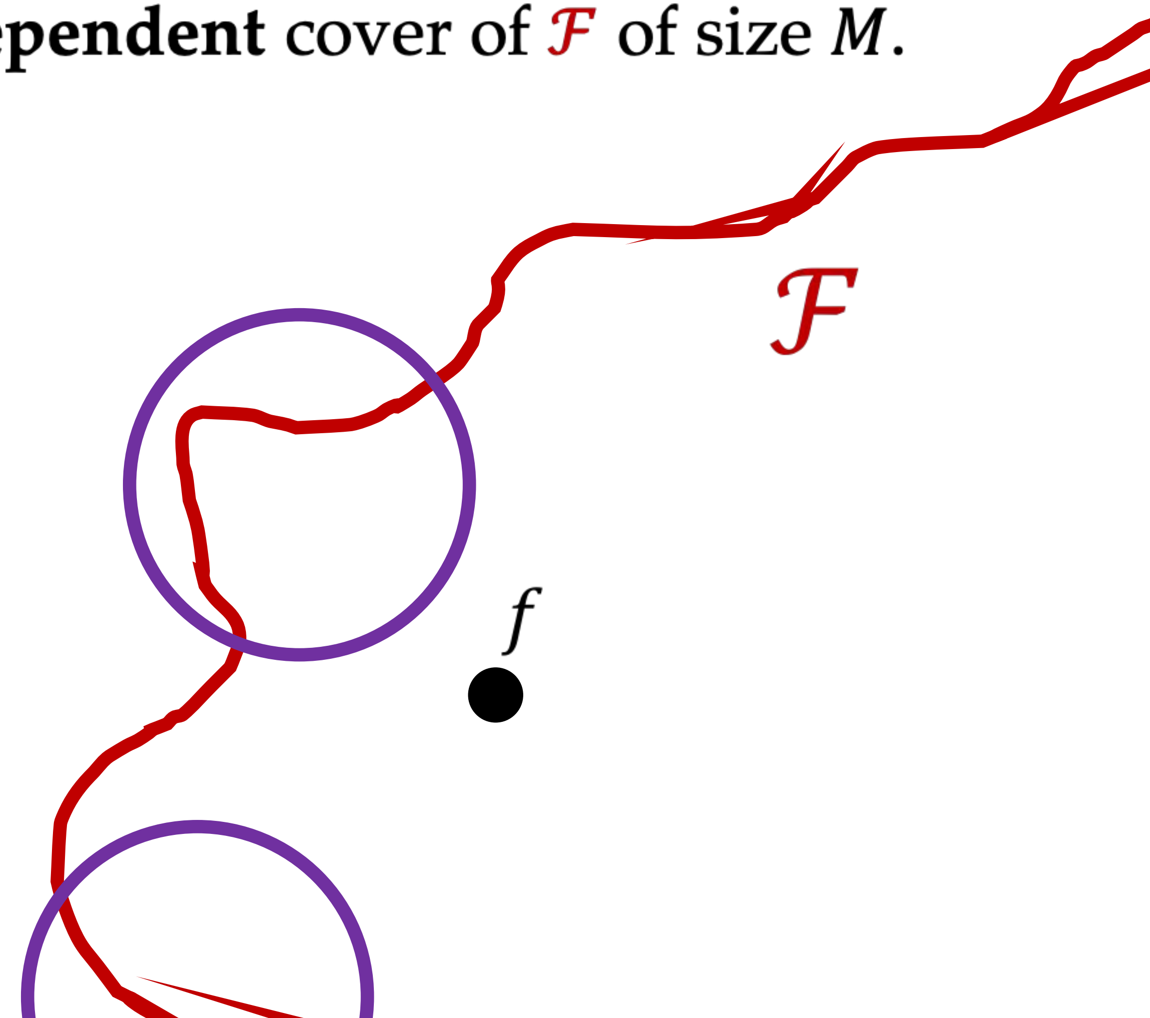


# Proof of Compression Theorem

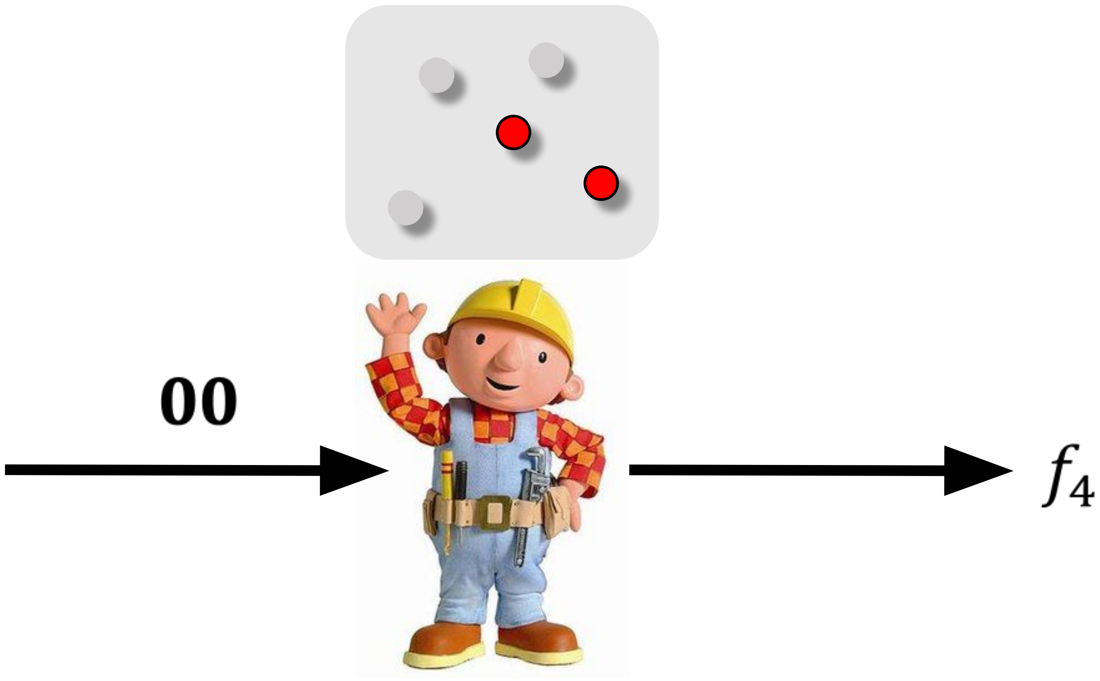
- We cannot implement Alice, but we can implement Bob!
- We draw $m(\epsilon)$ i.i.d. samples from $f$ and try all $M \leq (2m(\epsilon))^{t(\epsilon)}$ possible inputs to Bob to get a **data-dependent** cover of $\mathcal{F}$ of size $M$.

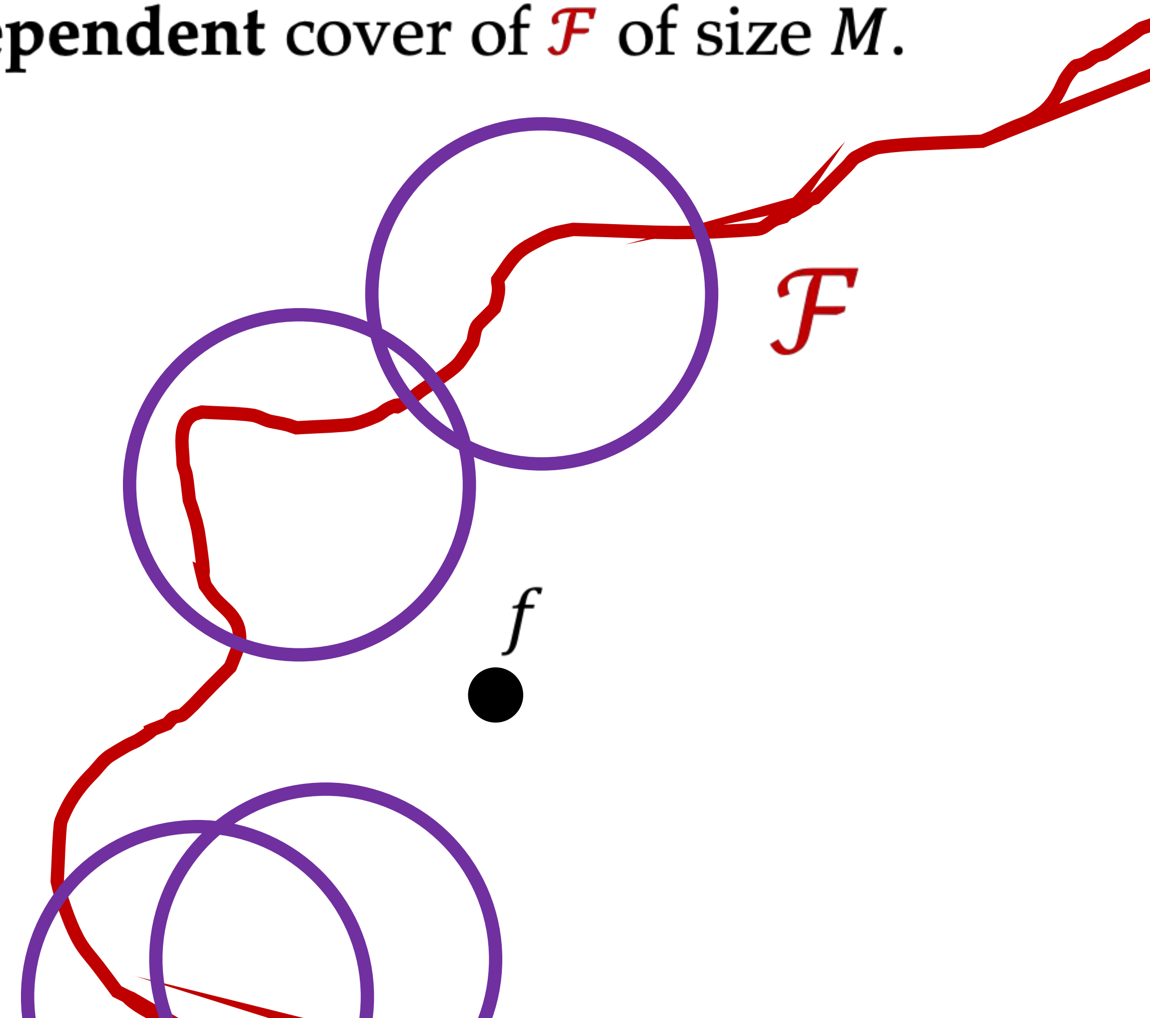


# Proof of Compression Theorem

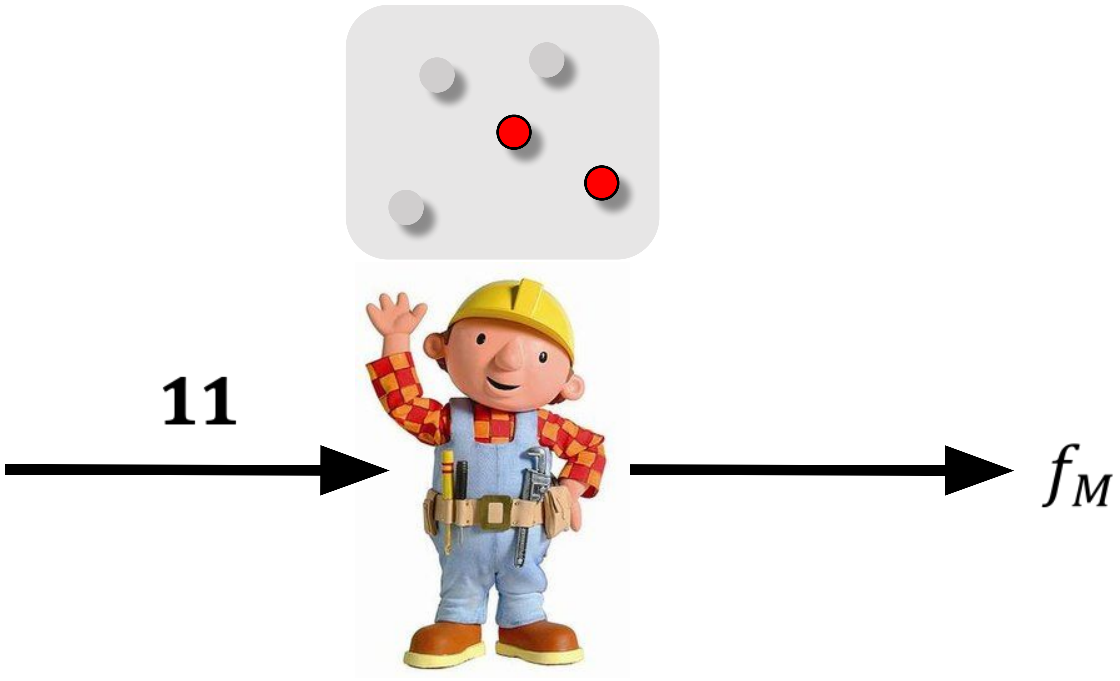
- We cannot implement Alice, but we can implement Bob!
- We draw $m(\epsilon)$ i.i.d. samples from $f$ and try all $M \leq (2m(\epsilon))^{t(\epsilon)}$ possible inputs to Bob to get a **data-dependent** cover of $\mathcal{F}$ of size $M$.

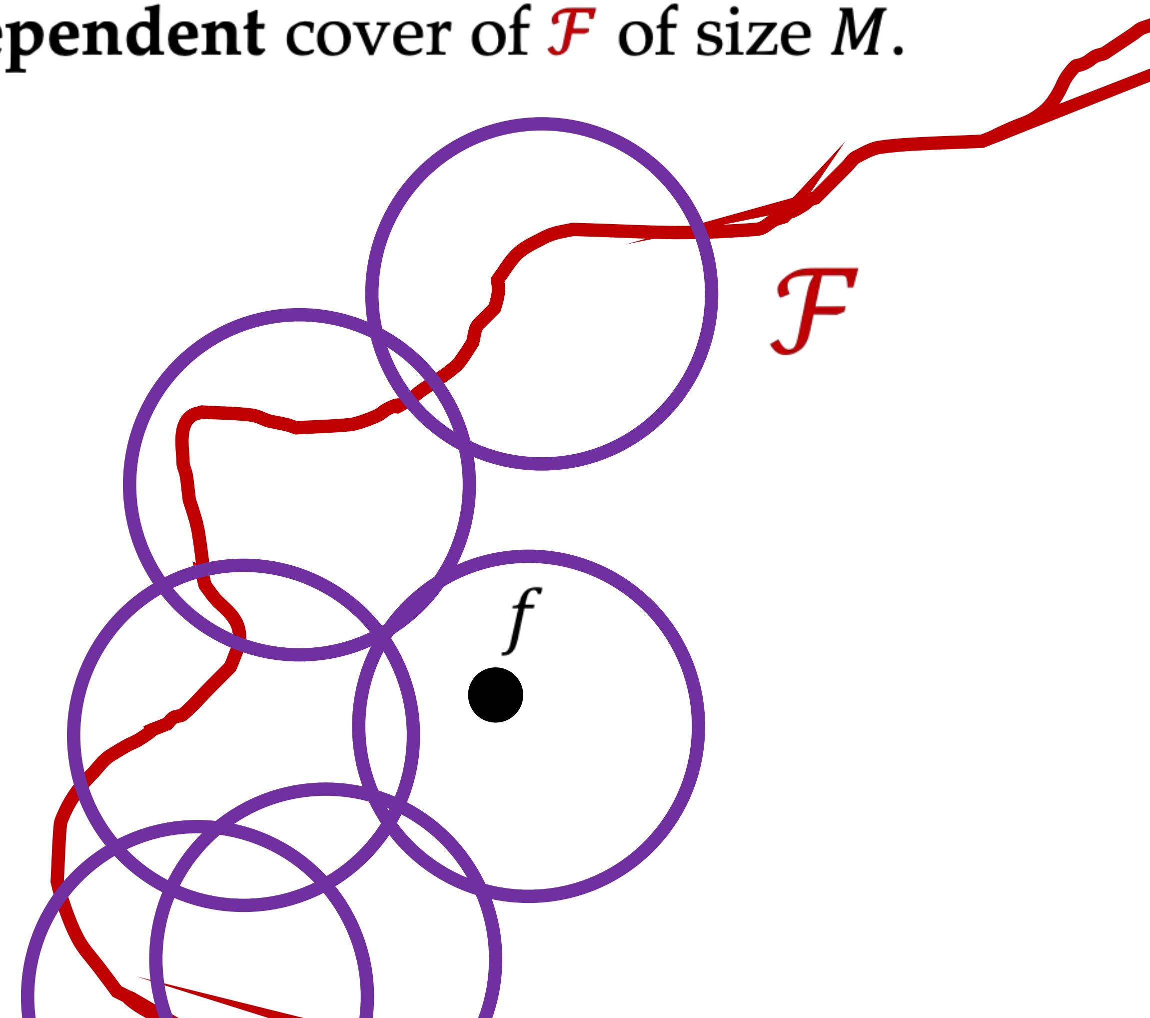


# Proof of Compression Theorem

- We cannot implement Alice, but we can implement Bob!
- We draw $m(\epsilon)$ i.i.d. samples from $f$ and try all $M \leq (2m(\epsilon))^{t(\epsilon)}$ possible inputs to Bob to get a **data-dependent** cover of $\mathcal{F}$ of size $M$.



# Proof of Compression Theorem

- We cannot implement Alice, but we can implement Bob!
- We draw $m(\epsilon)$ i.i.d. samples from $f$ and try all $M \leq (2m(\epsilon))^{t(\epsilon)}$ possible inputs to Bob to get a **data-dependent** cover of $\mathcal{F}$ of size $M$.
  - Existence of cover follows from $(m(\epsilon), t(\epsilon))$-compression.

**Lemma** [Yatracos '85] Suppose $f$ is an unknown density and we have densities $f_1, \ldots, f_M$ such that $\min_i d_{TV}(f_i, f) \leq \epsilon$. Then, $O(\log M/\epsilon^2)$ samples suffice to output $f_j$ with $d_{TV}(f_j, f) \leq O(\epsilon)$.

# Proof of Compression Theorem

- We cannot implement Alice, but we can implement Bob!
- We draw $m(\epsilon)$ i.i.d. samples from $f$ and try all $M \leq (2m(\epsilon))^{t(\epsilon)}$ possible inputs to Bob to get a **data-dependent** cover of $\mathcal{F}$ of size $M$.
  - Existence of cover follows from $(m(\epsilon), t(\epsilon))$-compression.

- Run Yatracos' "tournament" algorithm to find "best" distribution with $O(\log M / \epsilon^2)$ samples.
- Hence, sample complexity is

$$m(\epsilon) + O(\log M / \epsilon^2) = m(\epsilon) + O(t(\epsilon) \log m(\epsilon) / \epsilon^2).$$

■

Initial samples

Samples for Yatracos algorithm

# Where are we now?

**Compression Theorem.** If $\mathcal{F}$ admits $(m(\epsilon), t(\epsilon))$-compression then sample complexity to learn $\mathcal{F}$ (up to $\mathrm{d_{TV}}$-error $\epsilon$) is

$$\tilde{\boldsymbol{O}}\left(\boldsymbol{m}(\boldsymbol{\epsilon}) + \frac{\boldsymbol{t}(\boldsymbol{\epsilon})}{\boldsymbol{\epsilon}^2}\right).$$

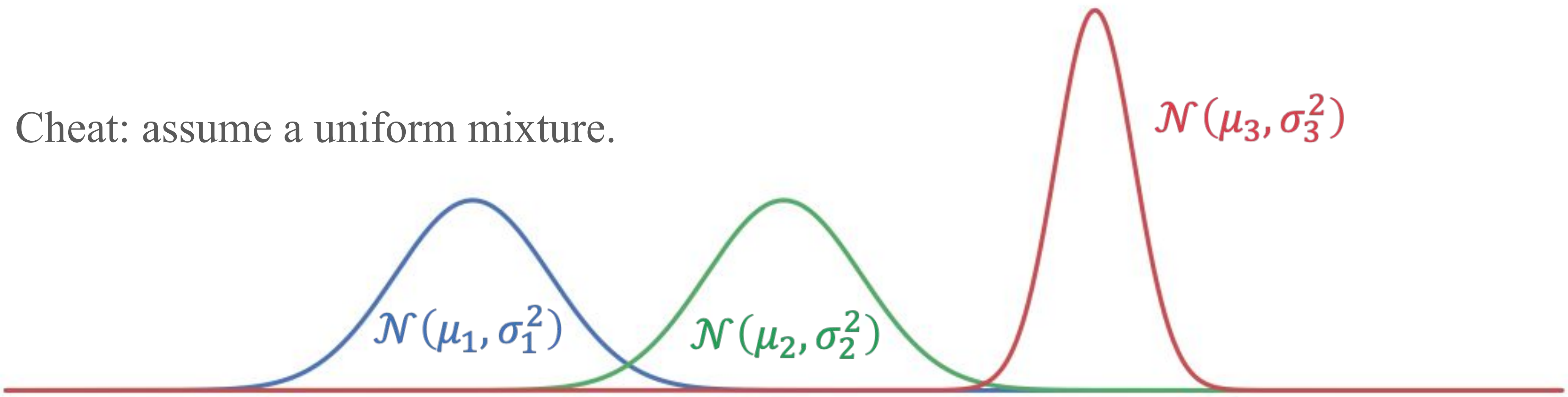
- **Reminder**: Our end goal is to prove a sample complexity bound of $\tilde{O}\left(\frac{kd^2}{\epsilon^2}\right)$ for learning mixtures of $k$ Gaussians.
- Suffices to find compression scheme with parameters

$$m(\epsilon) = \tilde{O}\left(\frac{kd^2}{\epsilon^2}\right) \quad \text{and} \quad t(\epsilon) = \tilde{O}(kd^2)$$

- Next, reduce to $k = 1$ case by giving a **general** compression scheme for **mixtures**.

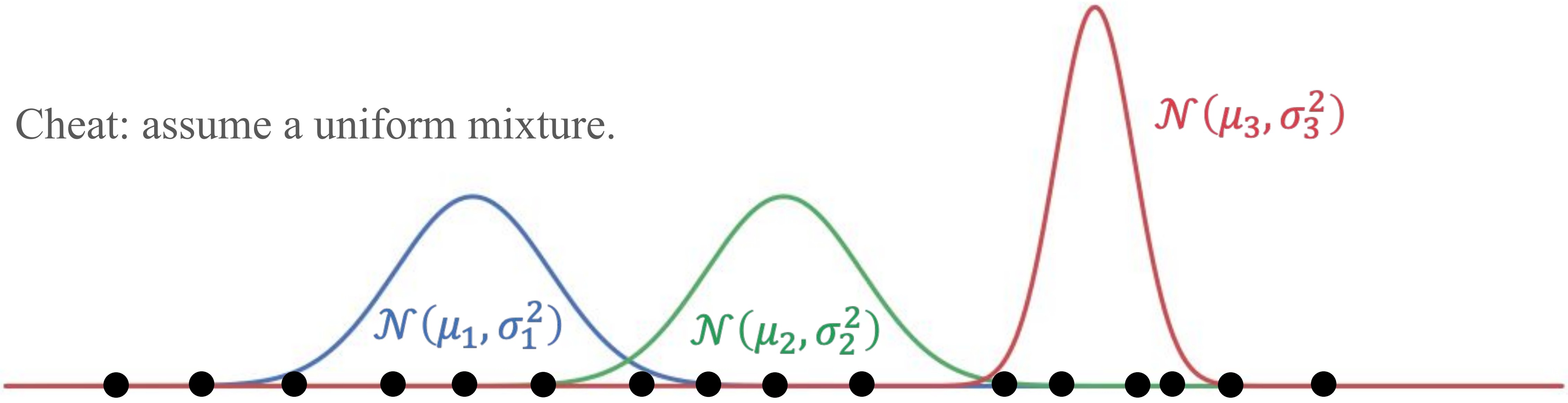
# Compression Of Mixtures

Cheat: assume a uniform mixture.



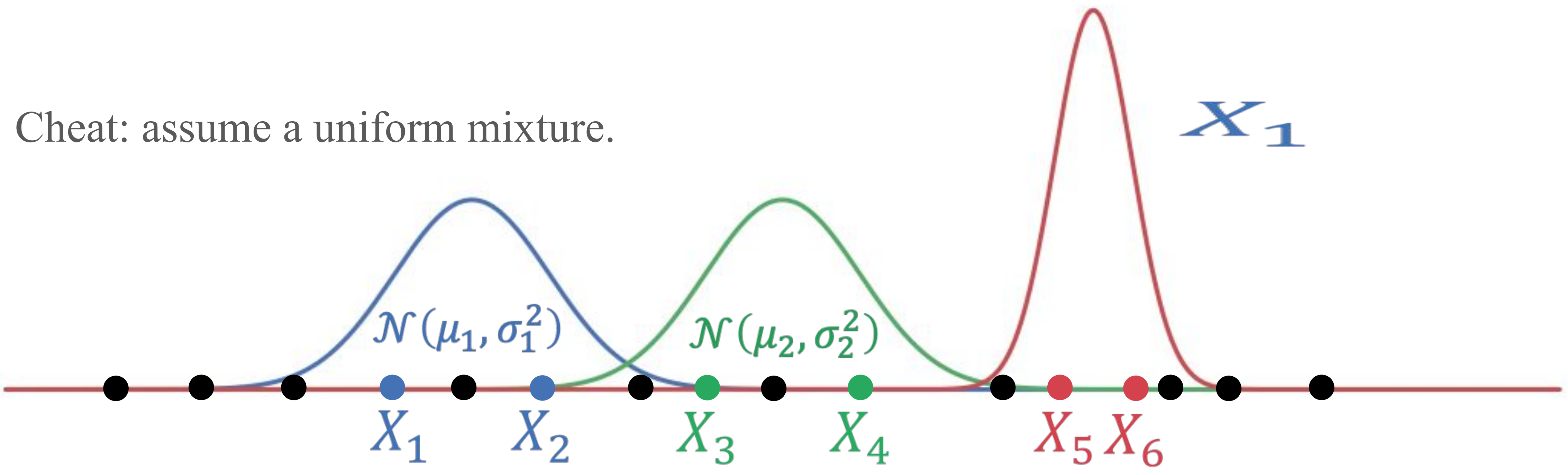
# Compression Of Mixtures

Cheat: assume a uniform mixture.



# Compression Of Mixtures

Cheat: assume a uniform mixture.



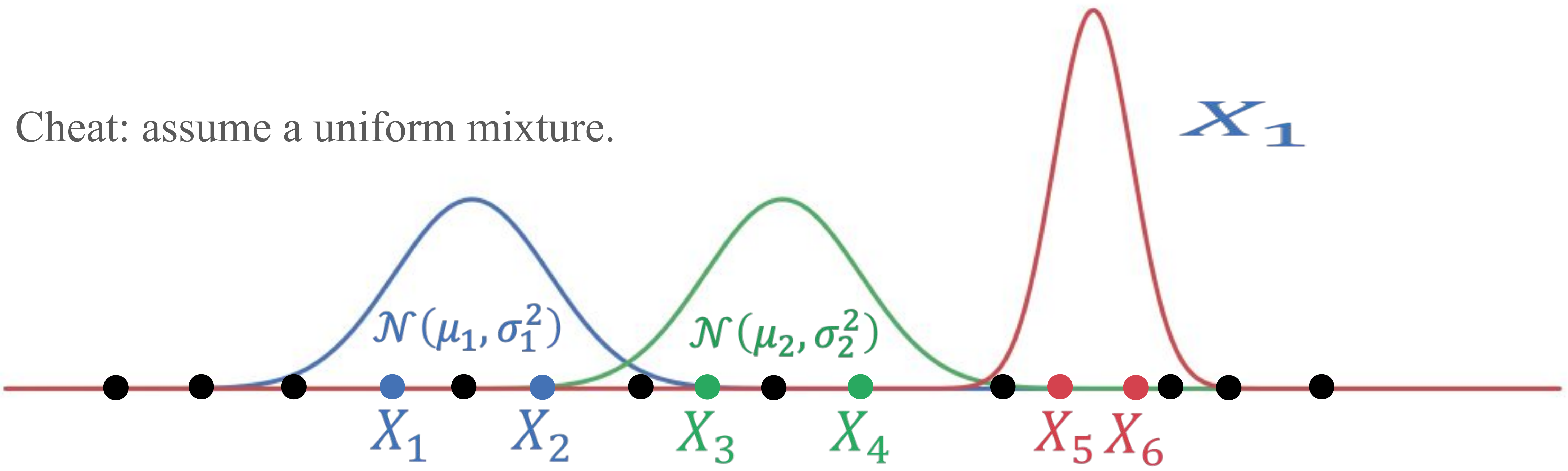

$X_1 \approx \mu_1 - \sigma_1$
$X_2 \approx \mu_1 + \sigma_1$

$X_3 \approx \mu_2 - \sigma_2$
$X_4 \approx \mu_2 + \sigma_2$

$X_5 \approx \mu_3 - \sigma_3$
$X_6 \approx \mu_3 + \sigma_3$

# Compression Of Mixtures

Cheat: assume a uniform mixture.




- If $\mathcal{F}$ has $(m(\epsilon), t(\epsilon))$-compression then $k$ mixtures of $\mathcal{F}$ have $\approx (km(\epsilon/k), kt(\epsilon/k))$–compression.
- To deal with weights, just use bits to encode them!
- If component has small mixing weight, give up on it.

# Compression Theorem for Mixtures

**Theorem** [ABHLMP '18] Suppose $\mathcal{F}$ admits $(m(\epsilon), t(\epsilon))$-compression. Then sample complexity to learn $k$-mix$(\mathcal{F})$ (up to $d_{TV}$-error $\epsilon$) is

$$\tilde{O}\left(\frac{km(\epsilon/k)}{\epsilon} + \frac{kt(\epsilon/k)}{\epsilon^2}\right).$$

Small compression schemes imply sample-efficient algorithms for **mixtures**.

**Q**: Does an analogous statement hold for other notions of complexity (e.g. VC-dimension)?

# Compression Theorem for Mixtures

**Theorem** [ABHLMP '18] Suppose $\mathcal{F}$ admits $(m(\epsilon), t(\epsilon))$-compression.
Then sample complexity to learn ***k***-mix($\mathcal{F}$) (up to $d_{TV}$-error $\epsilon$) is

$$\tilde{O}\left(\frac{km(\epsilon/k)}{\epsilon} + \frac{kt(\epsilon/k)}{\epsilon^2}\right).$$

**Goal**: Find a compression scheme for a *single* Gaussian with parameters

$$m(\epsilon) = \tilde{O}(d^2) \quad \text{and} \quad t(\epsilon) = \tilde{O}(d^2)$$

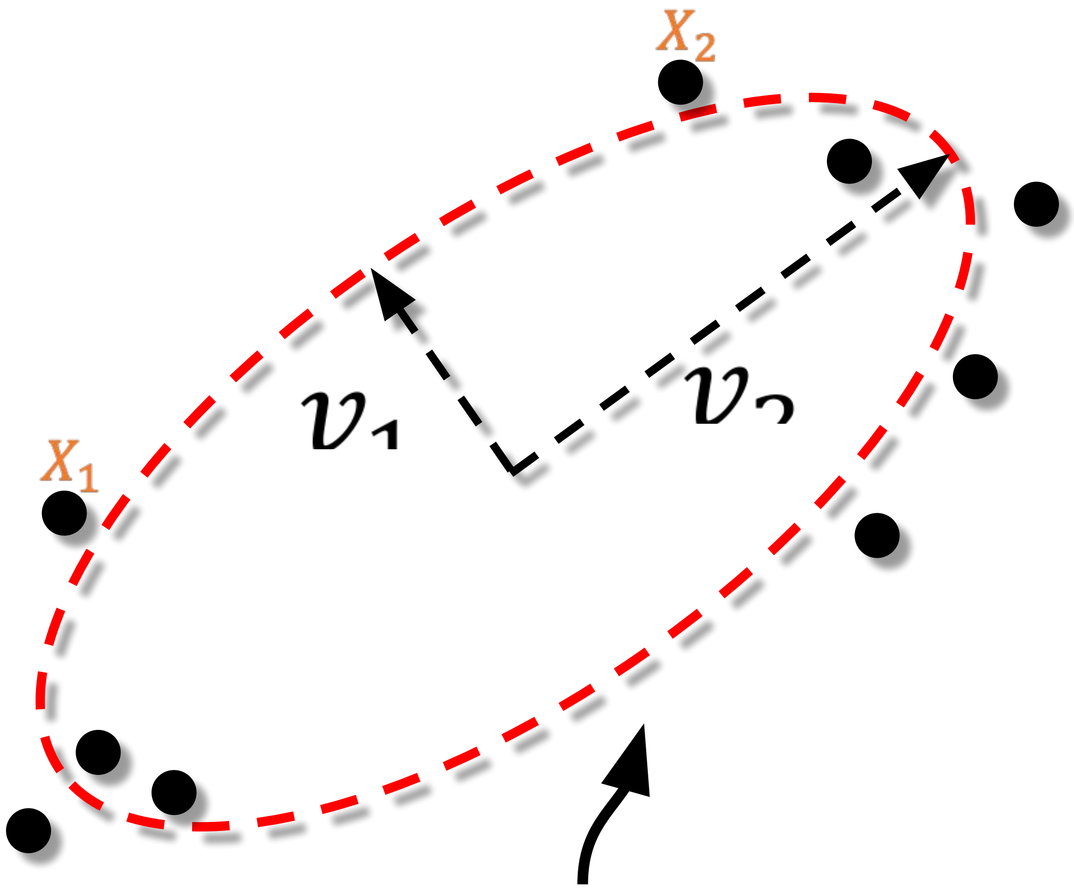
# Application: Learning Mixtures of Gaussians

To recover $\mathcal{N}(\mu, \Sigma)$, suffices to encode $\mu$ and eigenvectors/eigenvalues of $\Sigma$.

# Application: Learning Mixtures of Gaussians

To recover $\mathcal{N}(0, \Sigma)$, suffices to encode eigenvectors/eigenvalues of $\Sigma$.

Idea: Encode axes of ellipsoid using linear combination of samples.




Ellipsoid defined by $\Sigma$.
Points drawn from $\mathcal{N}(0, \Sigma)$.

# Application: Learning Mixtures of Gaussians

- Let $X_1, \ldots, X_d \sim \mathcal{N}(0, \Sigma)$; set $g_i = \Sigma^{-1/2} X_i \sim N(0, I_d)$
  - Recall that Alice knows $\Sigma$
- $g_1, \ldots, g_d$ are linearly independent so can write

$$e_k = \sum_i \lambda_{ki} g_i$$

# Application: Learning Mixtures of Gaussians

- Let $X_1, \dots, X_d \sim \mathcal{N}(0, \Sigma)$; set $g_i = \Sigma^{-1/2} X_i \sim N(0, I_d)$
    - Recall that Alice knows $\Sigma$
- $g_1, \dots, g_d$ are linearly independent so can write

$$\Sigma^{1/2} e_k = \sum_i \lambda_{ki} \Sigma^{1/2} g_i$$

# Application: Learning Mixtures of Gaussians

- Let $X_1, \dots, X_d \sim \mathcal{N}(0, \Sigma)$; set $g_i = \Sigma^{-1/2} X_i \sim N(0, I_d)$
  - Recall that Alice knows $\Sigma$
- $g_1, \dots, g_d$ are linearly independent so can write

$$\Sigma^{1/2} e_k = \sum_i \lambda_{ki} X_i$$

- Alice sends $X_1, \dots, X_d$ and $\{\lambda_{ki}\}$.

- Bob finds *any* matrix $A$ satisfying $A e_k = \sum_i \lambda_{ki} X_i = \Sigma^{1/2} e_k$
- Observation:

$$A e_k e_k^T A^T = \Sigma^{1/2} e_k e_k^T \Sigma^{1/2}$$

# Application: Learning Mixtures of Gaussians

- Let $X_1, \ldots, X_d \sim \mathcal{N}(0, \Sigma)$; set $g_i = \Sigma^{-1/2} X_i \sim N(0, I_d)$
  - Recall that Alice knows $\Sigma$
- $g_1, \ldots, g_d$ are linearly independent so can write

$$\Sigma^{1/2} e_k = \sum_i \lambda_{ki} X_i$$

- Alice sends $X_1, \ldots, X_d$ and $\{\lambda_{ki}\}$.

- Bob finds *any* matrix $A$ satisfying $A e_k = \sum_i \lambda_{ki} X_i = \Sigma^{1/2} e_k$
- Observation:

$$A\left(\sum_k e_k e_k^T\right) A^T = \Sigma^{1/2} \left(\sum_k e_k e_k^T\right) \Sigma^{1/2}$$

# Application: Learning Mixtures of Gaussians

- Let $X_1, \dots, X_d \sim \mathcal{N}(0, \Sigma)$; set $g_i = \Sigma^{-1/2} X_i \sim N(0, I_d)$
  - Recall that Alice knows $\Sigma$
- $g_1, \dots, g_d$ are linearly independent so can write

$$\Sigma^{1/2} e_k = \sum_i \lambda_{ki} X_i$$

- Alice sends $X_1, \dots, X_d$ and $\{\lambda_{ki}\}$.

- Bob finds *any* matrix $A$ satisfying $A e_k = \sum_i \lambda_{ki} X_i = \Sigma^{1/2} e_k$
- Observation:

$$A I_d A^T = \Sigma^{1/2} I_d \Sigma^{1/2}$$

# Application: Learning Mixtures of Gaussians

- Let $X_1, \dots, X_d \sim \mathcal{N}(0, \Sigma)$; set $g_i = \Sigma^{-1/2} X_i \sim N(0, I_d)$
  - Recall that Alice knows $\Sigma$
- $g_1, \dots, g_d$ are linearly independent so can write

$$\Sigma^{1/2} e_k = \sum_i \lambda_{ki} X_i$$

- Alice sends $X_1, \dots, X_d$ and $\{\lambda_{ki}\}$.

- Bob finds *any* matrix $A$ satisfying $A e_k = \sum_i \lambda_{ki} X_i = \Sigma^{1/2} e_k$
- Observation:

$$AA^T = \Sigma$$

# Application: Learning Mixtures of Gaussians

- Let $X_1, \ldots, X_d \sim \mathcal{N}(0, \Sigma)$; set $g_i = \Sigma^{-1/2} X_i \sim N(0, I_d)$
  - Recall that Alice knows $\Sigma$
- $g_1, \ldots, g_d$ are linearly independent so can write

$$\Sigma^{1/2} e_k = \sum_i \lambda_{ki} X_i$$

- Alice sends $X_1, \ldots, X_d$ and $\{\lambda_{ki}\}$.

Samples are fine.

These are **real**!
(Need some care in discretizing.)

- So $m(\epsilon) = d$ and $t(\epsilon) = \tilde{O}(d^2)$

# Application: Learning Mixtures of Gaussians

**Theorem** [ABHLMP '18] Sample complexity for learning mixtures of $k$ Gaussians in $\mathbb{R}^d$ (up to $\mathrm{d_{TV}}$-error $\epsilon$) is

$$\tilde{\mathbf{O}}\left(\frac{\boldsymbol{kd^2}}{\boldsymbol{\epsilon^2}}\right)$$

$\tilde{O}(\cdot)$ hides polylog factors

- For the axis-aligned case, we show $\tilde{O}(kd/\epsilon^2)$ samples suffice.
    - This is nearly-tight; matching lower bound from [Suresh et al. '14].

# Lower Bound

**Theorem** [Fano's Inequality]**.** Suppose $f_1, \dots, f_r$ satisfy

$$d_{TV}(f_i, f_j) > \epsilon \qquad \text{and} \qquad KL(f_i, f_j) < \epsilon^2.$$

Then sample complexity is $\Omega\left(\frac{\log r}{\epsilon^2}\right)$.

"Hard to distinguish"

$$KL(f_i, f_j) = \int f_i(x) \log \frac{f_i(x)}{f_j(x)} \, \mathrm{d}x$$

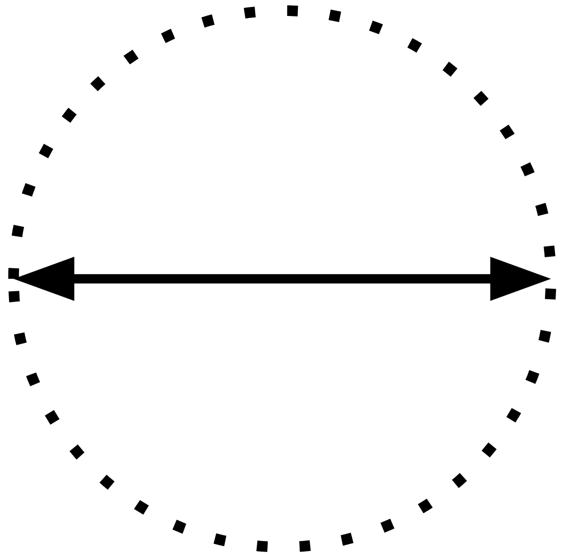
**Goal:** Find $2^{\Omega(kd^2)}$ mixtures of Gaussians that satisfy above hypothesis.

**How?** Just pick the Gaussians at random!

[Devroye, Mehrabian, Reddad '18] give a deterministic construction.

# Construction of hard instance ($k = 1$)

- Start with identity covariance matrix $I_d$
- Choose random subspace, $S_a$, of dimension $d/10$
- Increase eigenvalues by $\epsilon/\sqrt{d}$ along $S_a$
- Repeat $2^{\Omega(d^2)}$ times

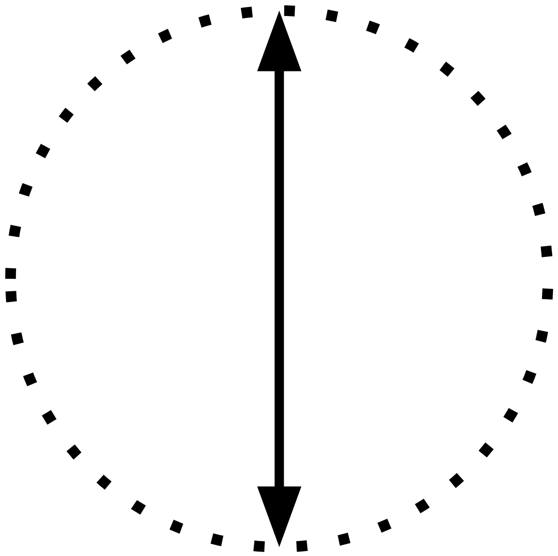


# Construction of hard instance ($k = 1$)

- Start with identity covariance matrix $I_d$
- Choose random subspace, $S_a$, of dimension $d/10$
- Increase eigenvalues by $\epsilon/\sqrt{d}$ along $S_a$
- Repeat $2^{\Omega(d^2)}$ times

- Hard distribution set is $\{f_a = \mathcal{N}(0, \Sigma_a)\}$

- Easy to show $KL(f_a, f_b) < O(\epsilon^2)$.
- Can also show $d_{TV}(f_a, f_b) > \Omega(\epsilon)$ w.p. $1 - \exp\left(-\Omega(d^2)\right)$.



# Summary

- We introduced a compression framework for density estimation.
  - **Application:** improved upper bounds for learning mixtures of Gaussians.
  - **Q**: Other applications of compression?
  - **Q**: Can we get a more computationally-efficient algorithm?
  - **Q**: What if we do not know $k$?

- We also showed a nearly-matching lower bound for learning mixtures of Gaussians.

# Thank you!
# Questions?