Disentangling Mixtures of Gaussians

Ankur Moitra, MIT

joint work with Adam Tauman Kalai and Gregory Valiant

June 28th, 2013

What is a Mixture of Gaussians?

What is a Mixture of Gaussians?

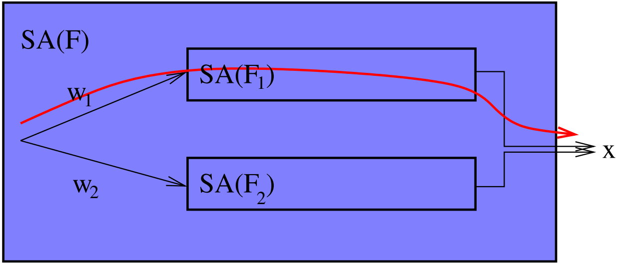
Distribution on $\Re^n$ $(w_1, w_2 \geq 0, w_1 + w_2 = 1)$:

What is a Mixture of Gaussians?

Distribution on $\Re^n$ ($w_1, w_2 \geq 0, w_1 + w_2 = 1$):

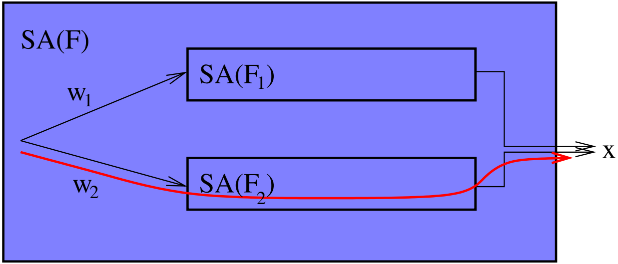
What is a Mixture of Gaussians?

Distribution on $\Re^n$ $(w_1, w_2 \geq 0, w_1 + w_2 = 1)$:



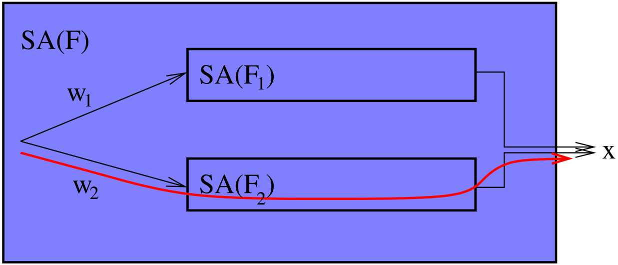
What is a Mixture of Gaussians?

Distribution on $\Re^n$ ($w_1, w_2 \geq 0, w_1 + w_2 = 1$):



What is a Mixture of Gaussians?

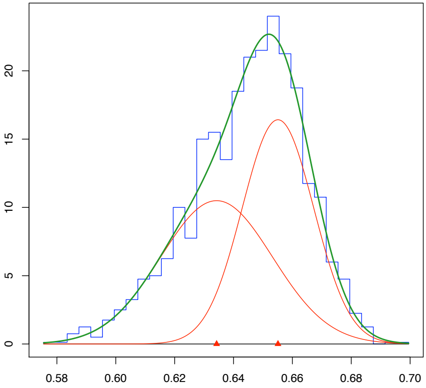
Distribution on $\Re^n$ $(w_1, w_2 \geq 0, w_1 + w_2 = 1)$:



$$F(x) = w_1 \mathcal{N}(\mu_1, \Sigma_1, x) + w_2 \mathcal{N}(\mu_2, \Sigma_2, x)$$

Pearson and the Naples Crabs

(figure due to Peter Macdonald)



Gaussian Mixture Models

Applications in physics, biology, geology, social sciences ...

Gaussian Mixture Models

Applications in physics, biology, geology, social sciences ...

Goal

Estimate parameters in order to estimate posterior distribution

Gaussian Mixture Models

Applications in physics, biology, geology, social sciences ...

Goal

Estimate parameters in order to estimate posterior distribution

Note: Alternating minimization only converges to a local optima, and maximum likelihood is hard to compute

Gaussian Mixture Models

Applications in physics, biology, geology, social sciences ...

Goal

Estimate parameters in order to estimate posterior distribution

Note: Alternating minimization only converges to a local optima, and maximum likelihood is hard to compute

Question

Can we provably recover the parameters in polynomial time? (Dasgupta, 1999)

Gaussian Mixture Models

Applications in physics, biology, geology, social sciences ...

Goal

Estimate parameters in order to estimate posterior distribution

Note: Alternating minimization only converges to a local optima, and maximum likelihood is hard to compute

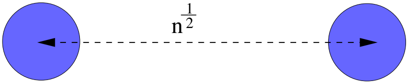
Question

Can we provably recover the parameters in polynomial time? (Dasgupta, 1999)

Definition

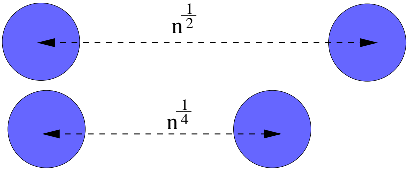
$D(f(x), g(x)) = \frac{1}{2}\|f(x) - g(x)\|_1$

Algorithms for Learning Mixtures of Gaussians



[Dasgupta, 1999]

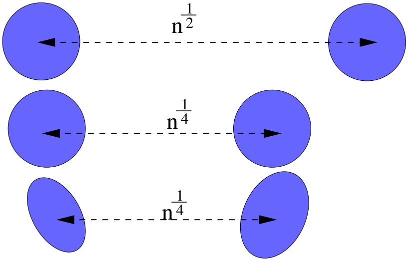
Algorithms for Learning Mixtures of Gaussians



[Dasgupta, 1999]

[Dasgupta, Schulman, 2000]

Algorithms for Learning Mixtures of Gaussians

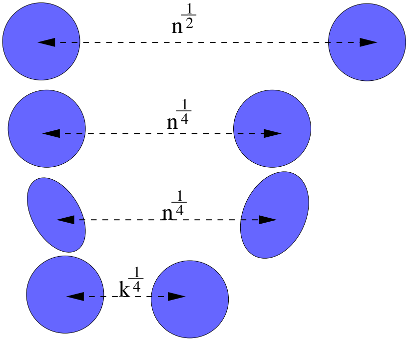


[Dasgupta, 1999]

[Dasgupta, Schulman, 2000]

[Arora, Kannan, 2001]

Algorithms for Learning Mixtures of Gaussians



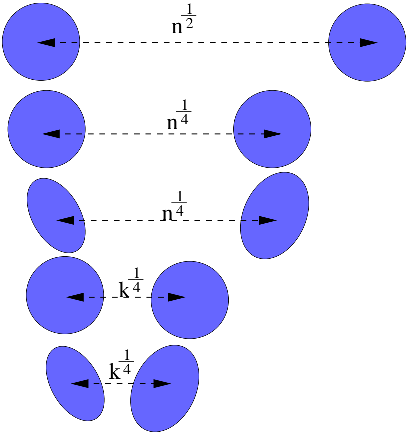
[Dasgupta, 1999]

[Dasgupta, Schulman, 2000]

[Arora, Kannan, 2001]

[Vempala, Wang, 2002]

Algorithms for Learning Mixtures of Gaussians



[Dasgupta, 1999]

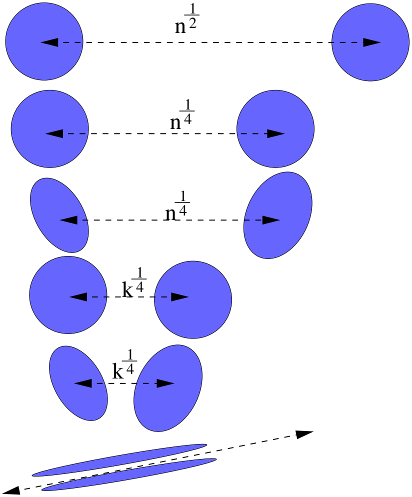
[Dasgupta, Schulman, 2000]

[Arora, Kannan, 2001]

[Vempala, Wang, 2002]

[Achlioptas, McSherry, 2005]

Algorithms for Learning Mixtures of Gaussians



[Dasgupta, 1999]

[Dasgupta, Schulman, 2000]

[Arora, Kannan, 2001]

[Vempala, Wang, 2002]

[Achlioptas, McSherry, 2005]

[Brubaker, Vempala, 2008]

All previous results required $D(F_1, F_2) \approx 1$...

All previous results required $D(F_1, F_2) \approx 1$...

... because the results relied on **CLUSTERING**

All previous results required $D(F_1, F_2) \approx 1$...

... because the results relied on **CLUSTERING**

Question

Can we learn the parameters of the mixture without clustering?

All previous results required $D(F_1, F_2) \approx 1$...

... because the results relied on **CLUSTERING**

Question

Can we learn the parameters of the mixture without clustering?

Question

Can we learn the parameters when $D(F_1, F_2)$ is close to **ZERO**?

Definition

A mixture of Gaussians $F = w_1 F_1 + w_2 F_2$ is ϵ-statistically learnable if for $i = \{1, 2\}$, $w_i \geq \epsilon$ and $D(F_1, F_2) \geq \epsilon$.

Definition

A mixture of Gaussians $F = w_1 F_1 + w_2 F_2$ is ϵ-statistically learnable if for $i = \{1, 2\}$, $w_i \geq \epsilon$ and $D(F_1, F_2) \geq \epsilon$.

Goal

Learn a mixture $\hat{F} = \hat{w}_1 \hat{F}_1 + \hat{w}_2 \hat{F}_2$ *so that there is a permutation* $\pi : \{1, 2\} \rightarrow \{1, 2\}$ *and for* $i = \{1, 2\}$

$$|w_i - \hat{w}_{\pi(i)}|, \|\mu_i - \hat{\mu}_{\pi(i)}\|, \|\Sigma_i - \hat{\Sigma}_{\pi(i)}\|_F \leq \epsilon$$

Definition

A mixture of Gaussians $F = w_1 F_1 + w_2 F_2$ is ϵ-statistically learnable if for $i = \{1, 2\}$, $w_i \geq \epsilon$ and $D(F_1, F_2) \geq \epsilon$.

Goal

Learn a mixture $\hat{F} = \hat{w}_1 \hat{F}_1 + \hat{w}_2 \hat{F}_2$ *so that there is a permutation* $\pi : \{1, 2\} \to \{1, 2\}$ *and for* $i = \{1, 2\}$

$$|w_i - \hat{w}_{\pi(i)}|, \|\mu_i - \hat{\mu}_{\pi(i)}\|, \|\Sigma_i - \hat{\Sigma}_{\pi(i)}\|_F \leq \epsilon$$

We will call such a mixture $\hat{F}$ ϵ-close to F.

Our Results

Given a polynomial number of random samples from an ϵ-statistically learnable mixture of two Gaussians F:

Our Results

Given a polynomial number of random samples from an ϵ-statistically learnable mixture of two Gaussians F:

Theorem (Kalai, Moitra, Valiant)

There is an algorithm that (with probability at least $1-\delta$) learns a mixture of two Gaussians $\hat{F}$ that is an ϵ-close estimate to F, and the running time and data requirements are poly$(\frac{1}{\epsilon}, n, \frac{1}{\delta})$.

Our Results

Given a polynomial number of random samples from an ϵ-statistically learnable mixture of two Gaussians F:

Theorem (Kalai, Moitra, Valiant)

There is an algorithm that (with probability at least $1-\delta$) learns a mixture of two Gaussians $\hat{F}$ that is an ϵ-close estimate to F, and the running time and data requirements are poly$(\frac{1}{\epsilon}, n, \frac{1}{\delta})$.

Previously, no known algorithm for <u>univariate</u> mixtures of <u>two</u> Gaussians that runs in better than exponential time

Our Results

Given a polynomial number of random samples from an ϵ-statistically learnable mixture of two Gaussians F:

Theorem (Kalai, Moitra, Valiant)

There is an algorithm that (with probability at least $1-\delta$) learns a mixture of two Gaussians $\hat{F}$ that is an ϵ-close estimate to F, and the running time and data requirements are poly$(\frac{1}{\epsilon}, n, \frac{1}{\delta})$.

Previously, no known algorithm for univariate mixtures of two Gaussians that runs in better than exponential time

See also [Moitra, Valiant] and [Belkin, Sinha] for mixtures of k Gaussians

Outline

Rough Idea

1. *Consider a series of projections down to one dimension*

Outline

Rough Idea

1. *Consider a series of projections down to one dimension*
2. *Run a univariate learning algorithm*

Outline

Rough Idea

1. *Consider a series of projections down to one dimension*
2. *Run a univariate learning algorithm*
3. *Use these estimates as constraints in a system of equations*

Outline

Rough Idea

1. *Consider a series of projections down to one dimension*
2. *Run a univariate learning algorithm*
3. *Use these estimates as constraints in a system of equations*
4. *Solve this system to obtain higher dimensional estimates*

Claim

$Proj_r[F_1] = \mathcal{N}(r^T \mu_1, r^T \Sigma_1 r, x)$

Claim

$Proj_r[F_1] = \mathcal{N}(r^T\mu_1, r^T\Sigma_1 r, x)$

Each univariate estimate yields an approximate linear constraint on the parameters

Claim

$Proj_r[F_1] = \mathcal{N}(r^T \mu_1, r^T \Sigma_1 r, x)$

Each univariate estimate yields an approximate linear constraint on the parameters

Definition

$D_p(\mathcal{N}(\mu_1, \sigma_1^2), \mathcal{N}(\mu_2, \sigma_2^2)) = |\mu_1 - \mu_2| + |\sigma_1^2 - \sigma_2^2|$

Problem

What if we choose a direction r s.t. $D_p(Proj_r[F_1], Proj_r[F_2])$ is extremely small?

Problem

What if we choose a direction r s.t. $D_p(Proj_r[F_1], Proj_r[F_2])$ is extremely small?

Then we would need to run the univariate algorithm with extremely fine precision!

Problem

What if we choose a direction r s.t. $D_p(Proj_r[F_1], Proj_r[F_2])$ is extremely small?

Then we would need to run the univariate algorithm with extremely fine precision!

Isotropic Projection Lemma: With high probability, $D_p(Proj_r[F_1], Proj_r[F_2])$ is at least polynomially large

Problem

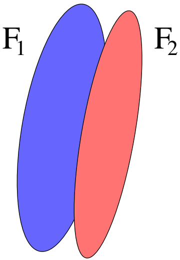
What if we choose a direction r s.t. $D_p(Proj_r[F_1], Proj_r[F_2])$ is extremely small?

Then we would need to run the univariate algorithm with extremely fine precision!

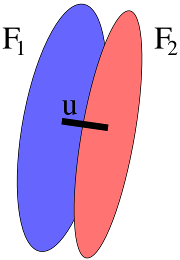
Isotropic Projection Lemma: With high probability, $D_p(Proj_r[F_1], Proj_r[F_2])$ is at least polynomially large

(i.e. at least $\epsilon_3 = poly(\epsilon, \frac{1}{n})$)

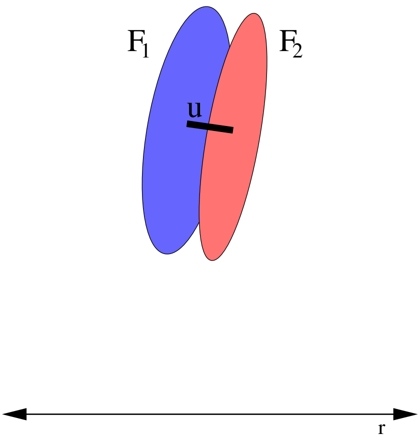
Isotropic Projection Lemma



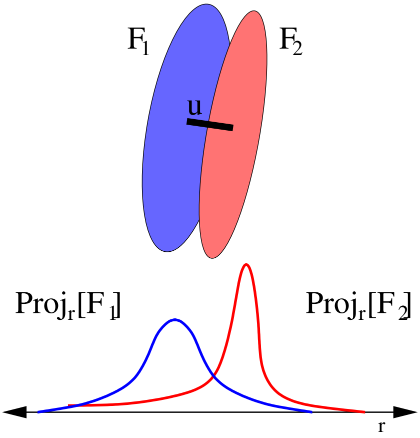
Isotropic Projection Lemma



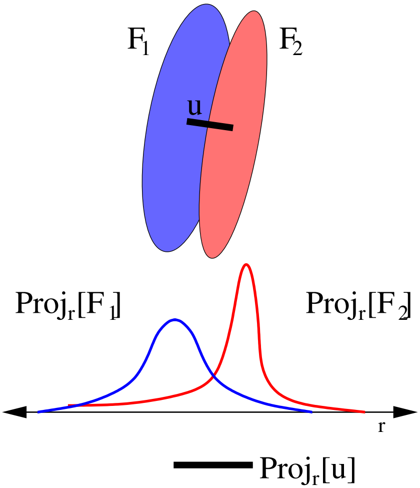
Isotropic Projection Lemma



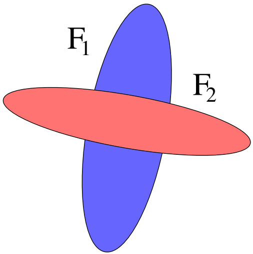
Isotropic Projection Lemma



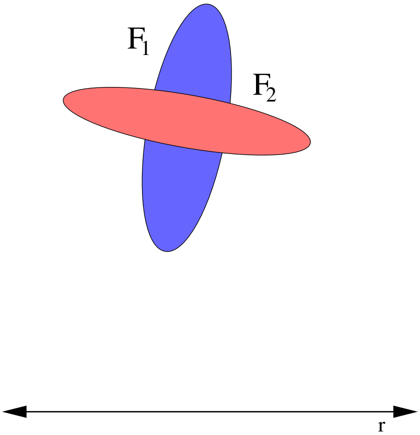
Isotropic Projection Lemma



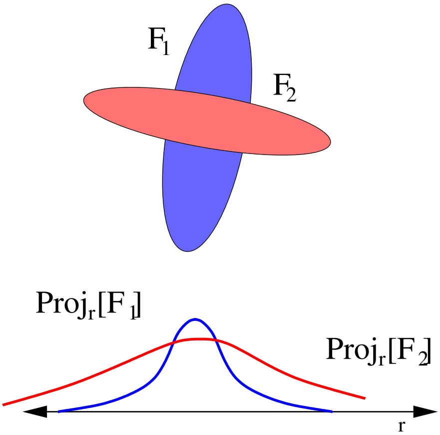
Isotropic Projection Lemma



Isotropic Projection Lemma



Isotropic Projection Lemma



Suppose we learn estimates $\hat{F}^r, \hat{F}^s$ from directions r, s

Suppose we learn estimates $\hat{F}^r, \hat{F}^s$ from directions r, s

$\hat{F}_1^r, \hat{F}_1^s$ each yield constraints on multidimensional parameters of one Gaussian in F

Suppose we learn estimates $\hat{F}^r, \hat{F}^s$ from directions r, s

$\hat{F}^r_1, \hat{F}^s_1$ each yield constraints on multidimensional parameters of one Gaussian in F

Problem

How do we know that they yield constraints on the same Gaussian?

Suppose we learn estimates $\hat{F}^r, \hat{F}^s$ from directions r, s

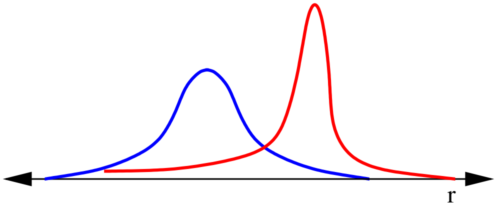
$\hat{F}^r_1, \hat{F}^s_1$ each yield constraints on multidimensional parameters of one Gaussian in F

Problem

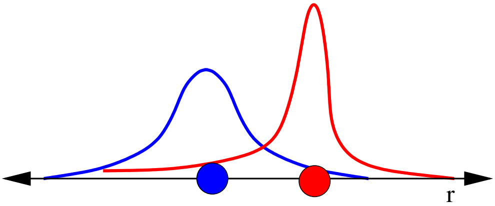
How do we know that they yield constraints on the *<u>same</u>* *Gaussian?*

Pairing Lemma: If we choose directions close enough (within $\epsilon_2 << \epsilon_3$), then pairing becomes easy

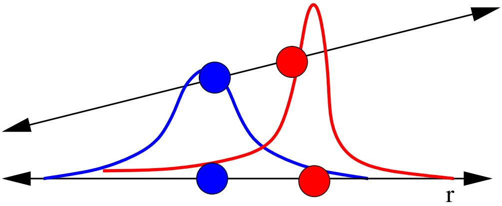
Searching Nearby Directions



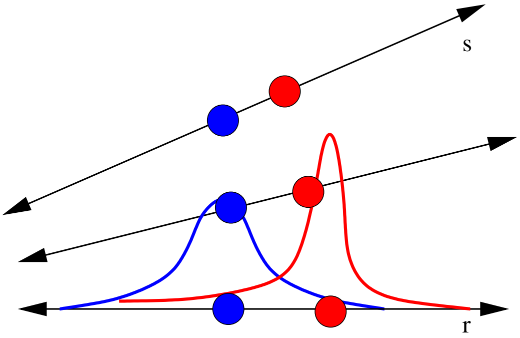
Searching Nearby Directions



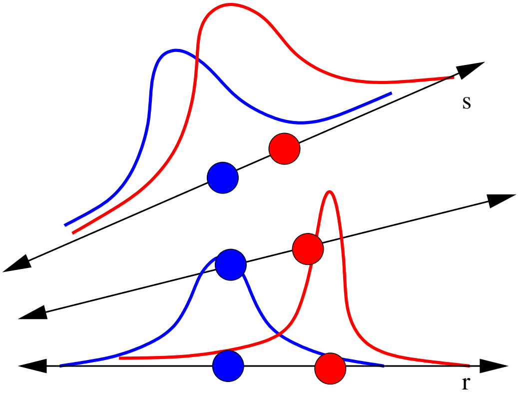
Searching Nearby Directions



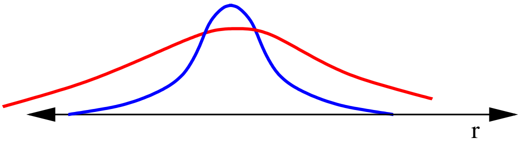
Searching Nearby Directions



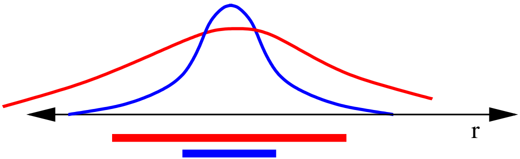
Searching Nearby Directions



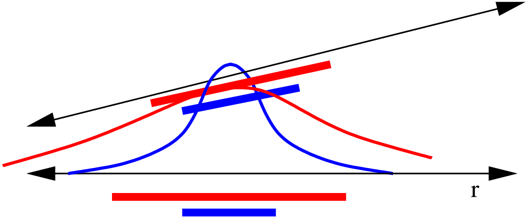
Searching Nearby Directions



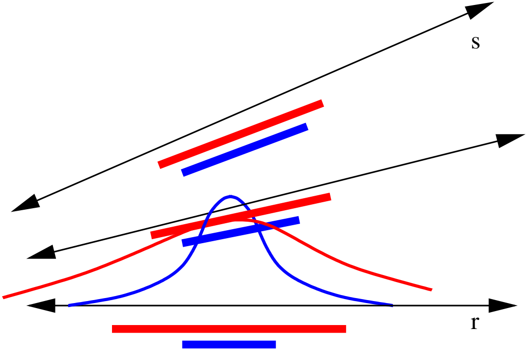
Searching Nearby Directions



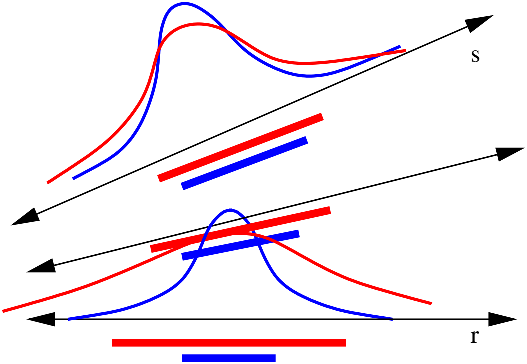
Searching Nearby Directions



Searching Nearby Directions



Searching Nearby Directions



Each univariate estimate yields a linear constraint on the parameters:

$$Proj_r[F_1] = \mathcal{N}(r^T \mu_1, r^T \Sigma_1 r)$$

Each univariate estimate yields a linear constraint on the parameters:

$$Proj_r[F_1] = \mathcal{N}(r^T \mu_1, r^T \Sigma_1 r)$$

Problem

How do errors in univariate estimates translate to errors in multidimensional estimates?

Each univariate estimate yields a linear constraint on the parameters:

$$Proj_r[F_1] = \mathcal{N}(r^T\mu_1, r^T\Sigma_1 r)$$

Problem

How do errors in univariate estimates translate to errors in multidimensional estimates? (i.e. What is the condition number of this system?)

Each univariate estimate yields a linear constraint on the parameters:

$$Proj_r[F_1] = \mathcal{N}(r^T \mu_1, r^T \Sigma_1 r)$$

Problem

How do errors in univariate estimates translate to errors in multidimensional estimates? (i.e. What is the condition number of this system?)

Recovery Lemma: Condition number is polynomially bounded

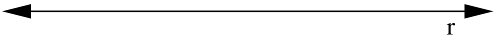
Each univariate estimate yields a linear constraint on the parameters:

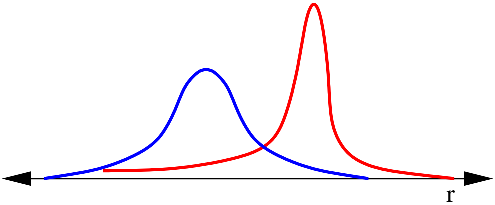
$$Proj_r[F_1] = \mathcal{N}(r^T \mu_1, r^T \Sigma_1 r)$$

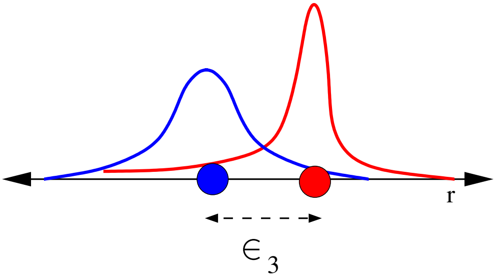
Problem

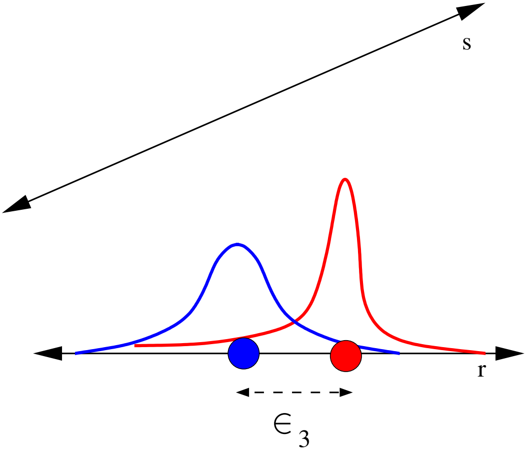
How do errors in univariate estimates translate to errors in multidimensional estimates? (i.e. What is the condition number of this system?)

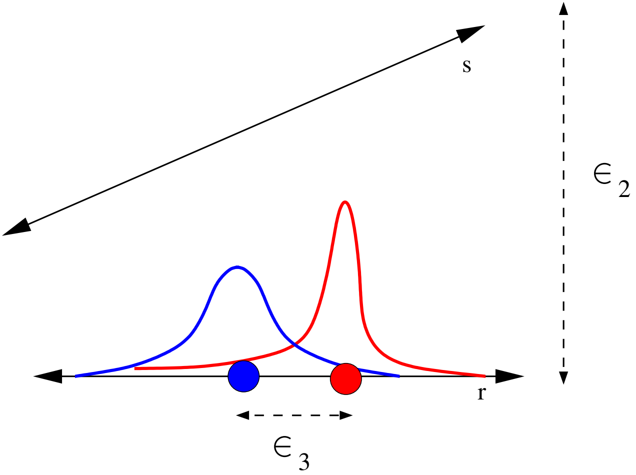
Recovery Lemma: Condition number is polynomially bounded : $O(\frac{n}{\epsilon_2^2})$

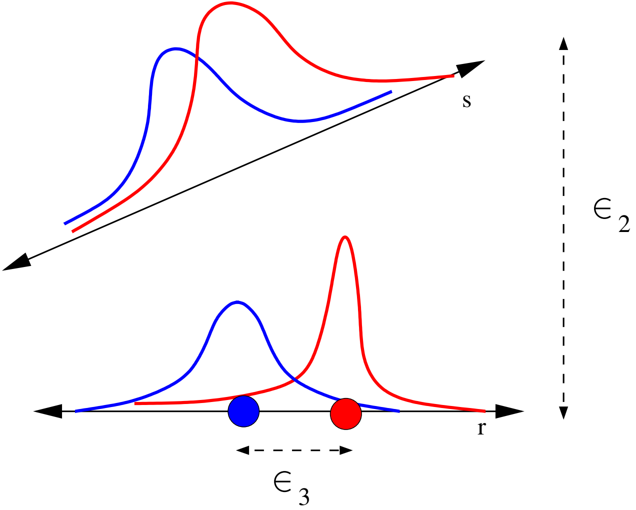

r


r

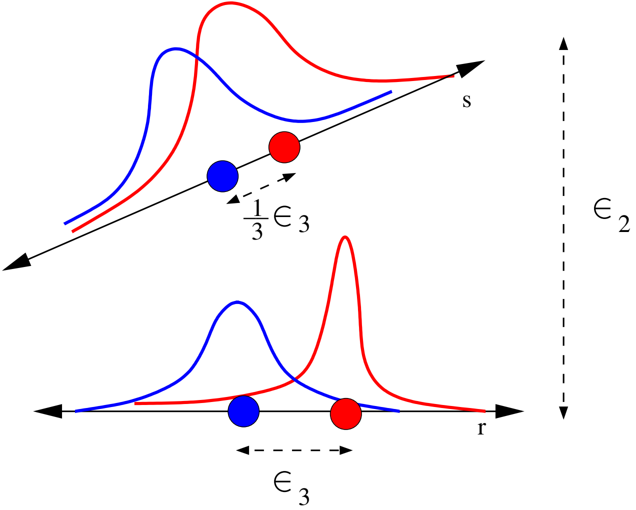

r
$\in_3$

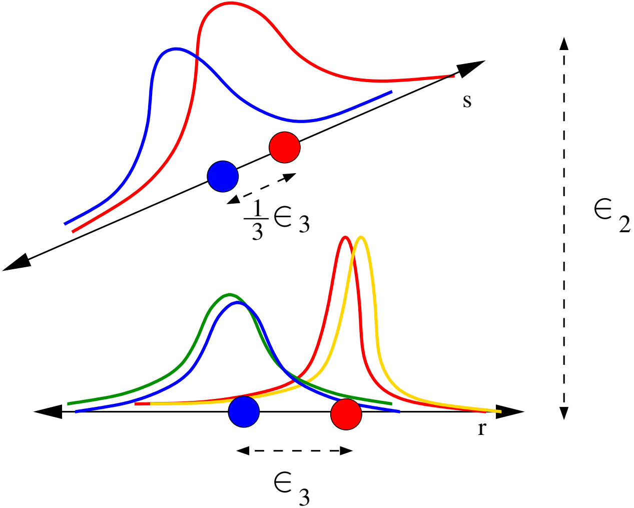

s
r
$\in_3$

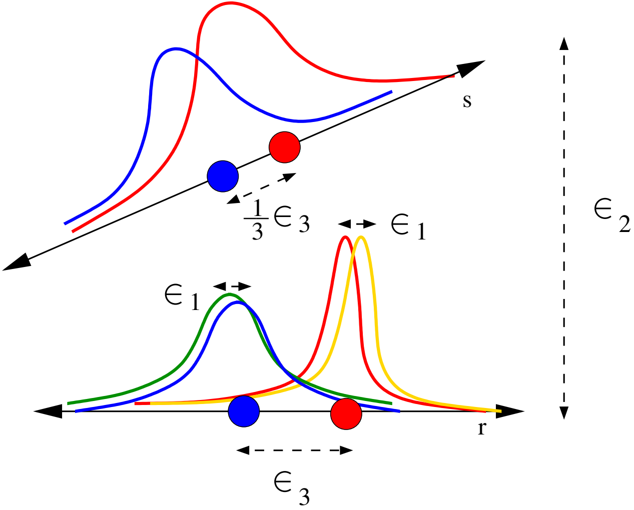

s
∈ 2
r
∈ 3

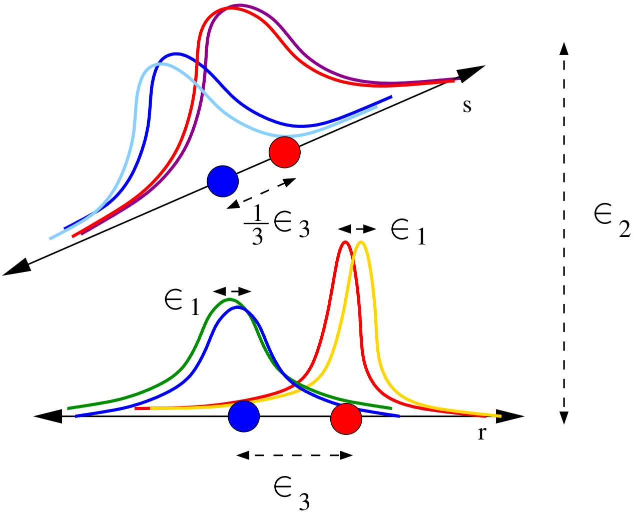


s
$\in_2$
r
$\in_3$

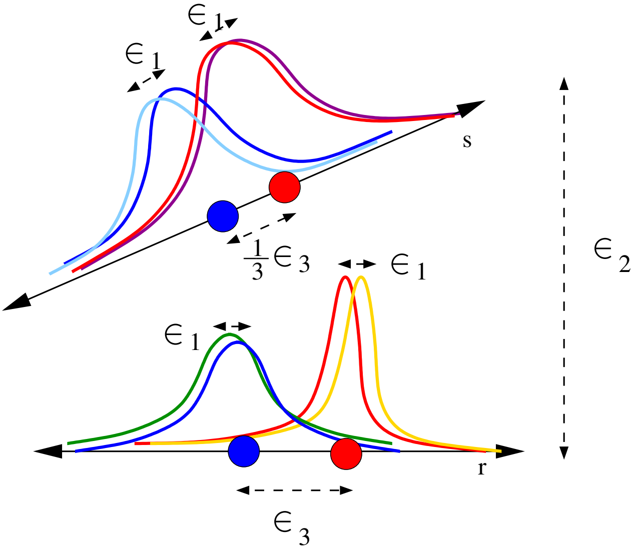

s
$\frac{1}{3} \in 3$
$\in 2$
r
$\in 3$


s
$\frac{1}{3}\epsilon_3$
ϵ_2
r
ϵ_3

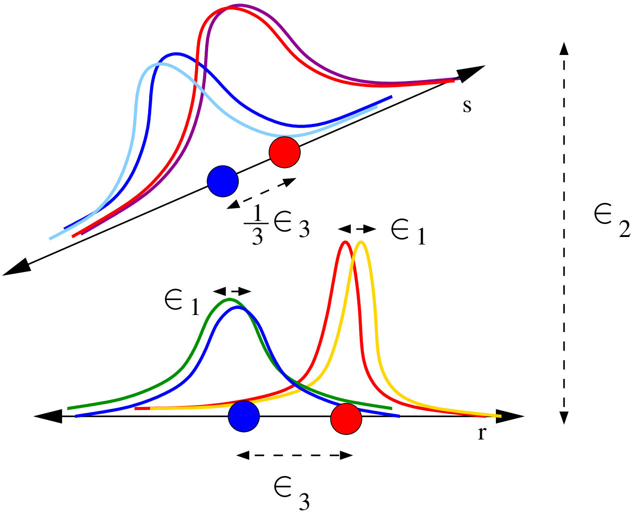

s
$\frac{1}{3}\in_3$
$\in_1$
$\in_2$
$\in_1$
r
$\in_3$

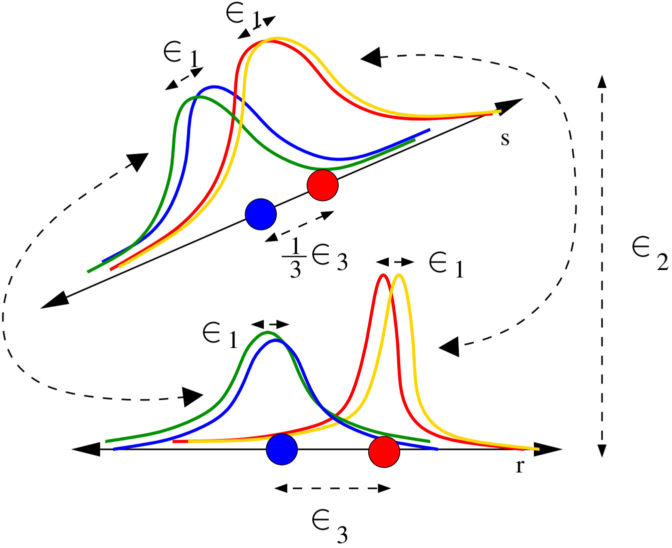


s
$\frac{1}{3}\in_3$
$\in_1$
$\in_2$
$\in_1$
r
$\in_3$

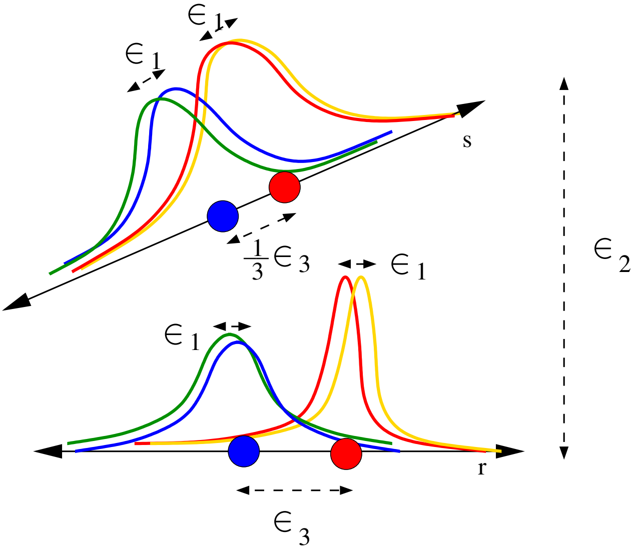


$\in_1$
$\in_1$
s
$\frac{1}{3}\in_3$
$\in_1$
$\in_2$
$\in_1$
r
$\in_3$

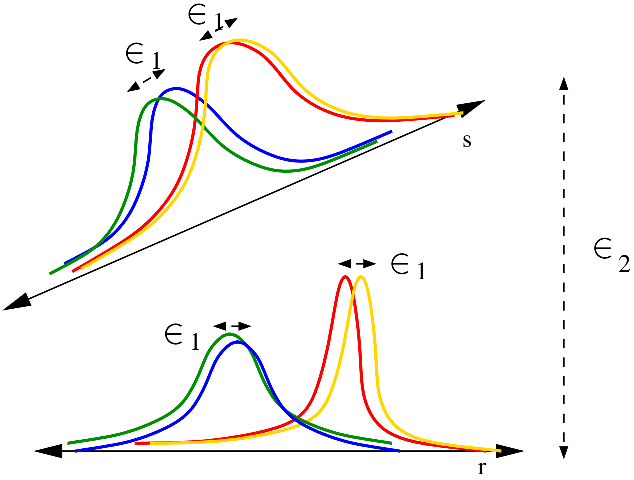

s
$\frac{1}{3}\in_3$
$\in_1$
$\in_2$
$\in_1$
r
$\in_3$



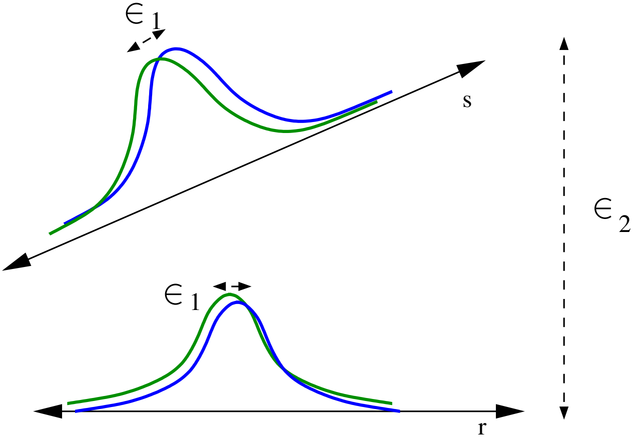
∈ 1
∈ 1
s
1/3 ∈ 3
∈ 1
∈ 2
∈ 1
r
∈ 3



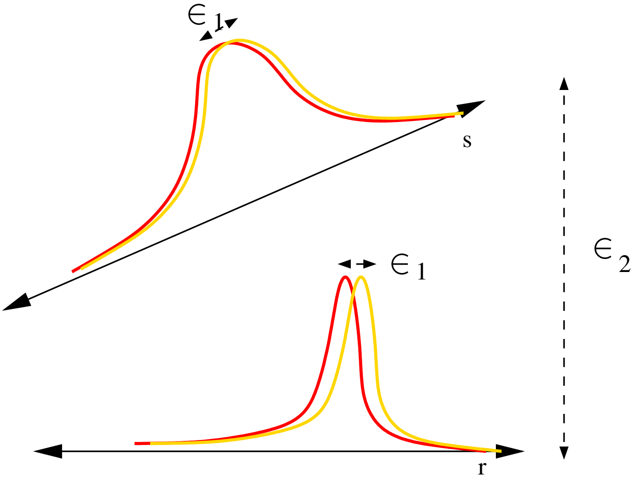
ϵ_1
ϵ_1
s
$\frac{1}{3}\epsilon_3$
ϵ_1
ϵ_2
ϵ_1
r
ϵ_3



∈ 1
∈ 1
s
∈ 2
∈ 1
∈ 1
r



$\in_1$
s
$\in_2$
$\in_1$
r



∈ 1
s
∈ 1
∈ 2
r

Question

Can we learn an additive approximation in one dimension?

Question

Can we learn an additive approximation in one dimension? How many free parameters are there?

Question

Can we learn an additive approximation in one dimension? How many free parameters are there?

$$w_1, \mu_1, \sigma_1^2, \mu_2, \sigma_2^2$$

Question

Can we learn an additive approximation in one dimension? How many free parameters are there?

$$w_1, \mu_1, \sigma_1^2, \mu_2, \sigma_2^2$$

Univariate learning algorithm: brute force search

Question

Can we learn an additive approximation in one dimension? How many free parameters are there?

$$w_1, \mu_1, \sigma_1^2, \mu_2, \sigma_2^2$$

Univariate learning algorithm: brute force search

Question

How do we test if a set of parameters is (approximately) correct?

Question

Can we learn an additive approximation in one dimension? How many free parameters are there?

$$w_1, \mu_1, \sigma_1^2, \mu_2, \sigma_2^2$$

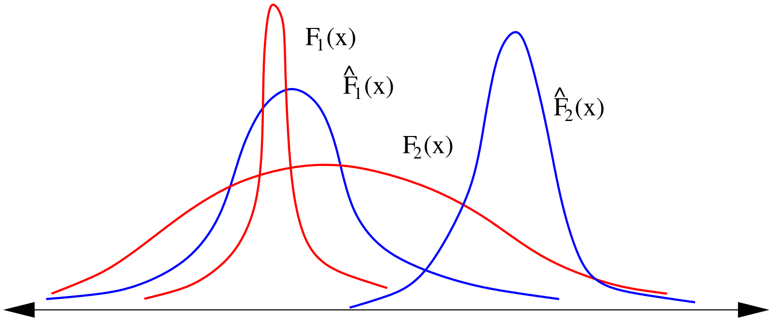
Univariate learning algorithm: brute force search

Question

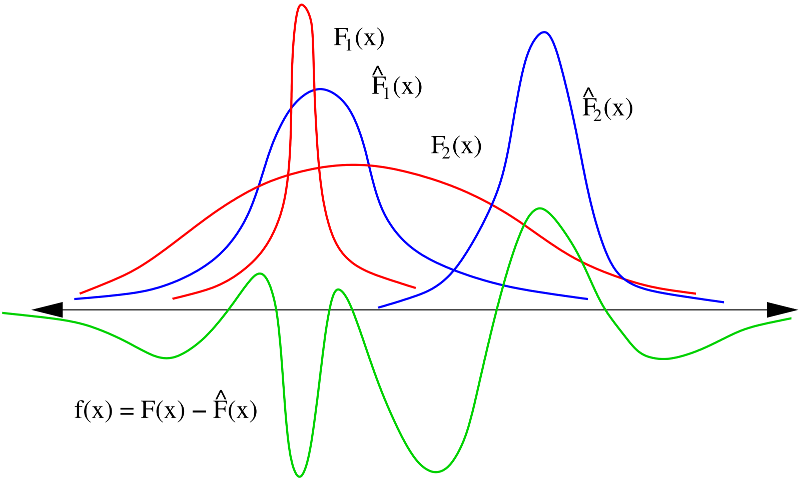
How do we test if a set of parameters is (approximately) correct?

... using the **method of moments**

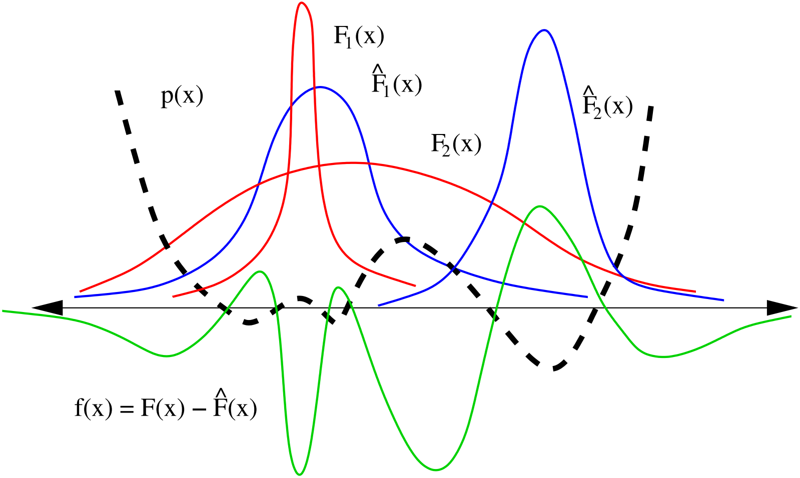
Method of Moments



Method of Moments



Method of Moments



Question

Why does this imply one of the first six moment of $F, \hat{F}$ is different?

$$0 < \left| \int_x p(x) f(x) dx \right|$$

Question

Why does this imply one of the first six moment of $F, \hat{F}$ is different?

$$0 < \left| \int_x p(x) f(x) dx \right| \;=\; \left| \int_x \sum_{r=1}^{6} p_r x^r f(x) dx \right|$$

Question

Why does this imply one of the first six moment of $F, \hat{F}$ is different?

$$
\begin{aligned}
0 < \Big| \int_x p(x) f(x) dx \Big| &= \Big| \int_x \sum_{r=1}^{6} p_r x^r f(x) dx \Big| \\
&\leq \sum_{r=1}^{6} |p_r| |M_r(F) - M_r(\hat{F})|
\end{aligned}
$$

Question

Why does this imply one of the first six moment of $F, \hat{F}$ is different?

$$
\begin{aligned}
0 < \left| \int_x p(x) f(x) dx \right| &= \left| \int_x \sum_{r=1}^{6} p_r x^r f(x) dx \right| \\
&\leq \sum_{r=1}^{6} |p_r| |M_r(F) - M_r(\hat{F})|
\end{aligned}
$$

So $\exists_{r \in \{1,2,\ldots,6\}}$ s.t. $|M_r(F) - M_r(\hat{F})| > 0$

Proposition

Let $f(x) = \sum_{i=1}^{k} \alpha_i \mathcal{N}(\mu_i, \sigma_i^2, x)$ be a linear combination of k Gaussians (α_i can be negative). Then if $f(x)$ is not identically zero, $f(x)$ has at most $2k - 2$ zero crossings.

Proposition

Let $f(x) = \sum_{i=1}^{k} \alpha_i \mathcal{N}(\mu_i, \sigma_i^2, x)$ *be a linear combination of* k *Gaussians (*α_i *can be negative). Then if* $f(x)$ *is not identically zero,* $f(x)$ *has at most* $2k - 2$ *zero crossings.*

Theorem (Hummel, Gidas)

Given $f(x) : \Re \to \Re$*, that is analytic and has* n *zeros, then for any* $\sigma^2 > 0$*, the function* $g(x) = f(x) \circ \mathcal{N}(0, \sigma^2, x)$ *has at most* n *zeros.*

Proposition

Let $f(x) = \sum_{i=1}^{k} \alpha_i \mathcal{N}(\mu_i, \sigma_i^2, x)$ be a linear combination of k Gaussians (α_i can be negative). Then if $f(x)$ is not identically zero, $f(x)$ has at most $2k-2$ zero crossings.

Theorem (Hummel, Gidas)

Given $f(x) : \Re \rightarrow \Re$, that is analytic and has n zeros, then for any $\sigma^2 > 0$, the function $g(x) = f(x) \circ \mathcal{N}(0, \sigma^2, x)$ has at most n zeros.

Convolving by a Gaussian does not increase the number of zero crossings!

Proposition

Let $f(x) = \sum_{i=1}^{k} \alpha_i \mathcal{N}(\mu_i, \sigma_i^2, x)$ *be a linear combination of* k *Gaussians* (α_i *can be negative). Then if* $f(x)$ *is not identically zero,* $f(x)$ *has at most* $2k - 2$ *zero crossings.*

Theorem (Hummel, Gidas)

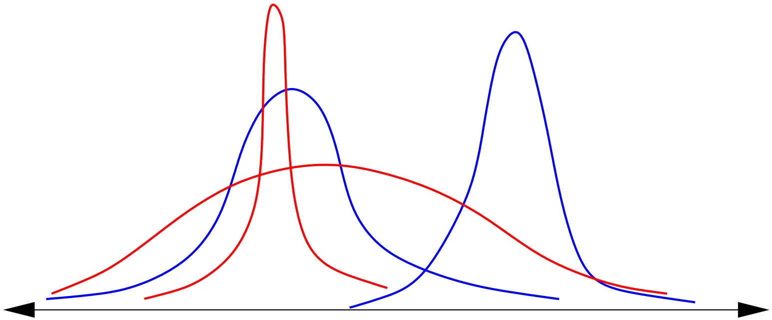
Given $f(x) : \Re \to \Re$, *that is analytic and has* n *zeros, then for any* $\sigma^2 > 0$, *the function* $g(x) = f(x) \circ \mathcal{N}(0, \sigma^2, x)$ *has at most* n *zeros.*

Convolving by a Gaussian does not increase the number of zero crossings!

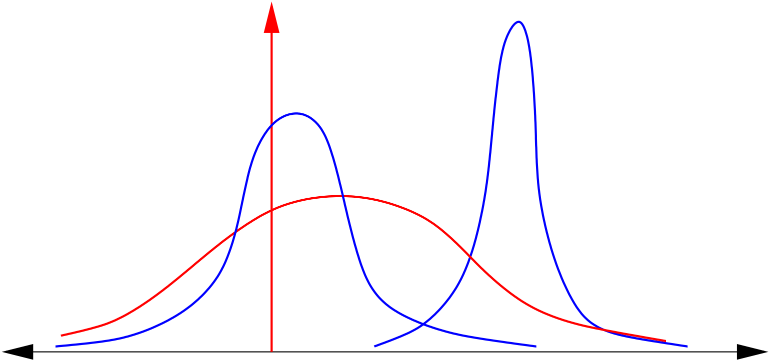
Fact

$\mathcal{N}(0, \sigma_1^2, x) \circ \mathcal{N}(0, \sigma_2^2, x) = \mathcal{N}(0, \sigma_1^2 + \sigma_2^2, x)$

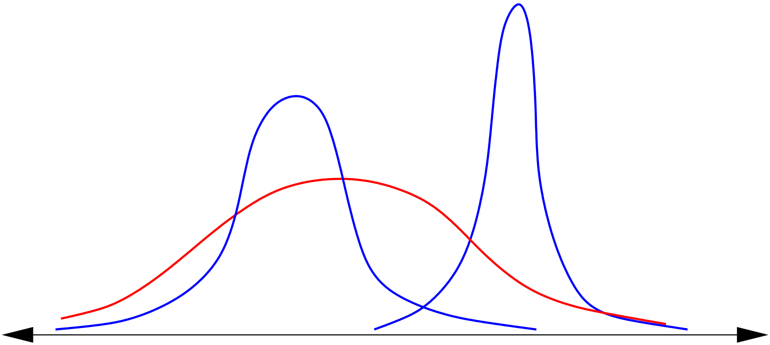
Zero Crossings and the Heat Equation



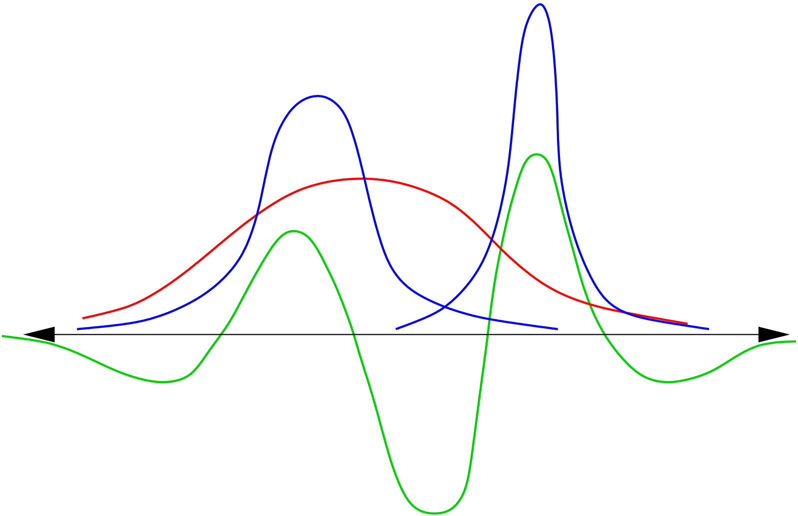
Zero Crossings and the Heat Equation



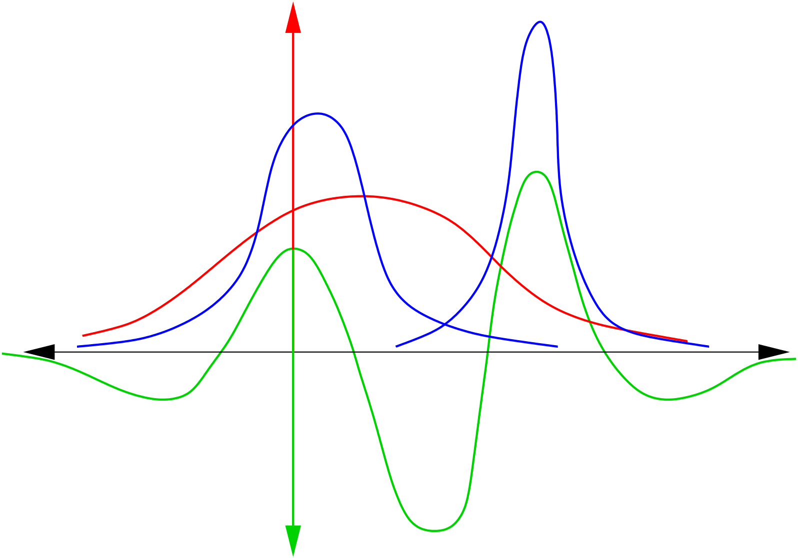
Zero Crossings and the Heat Equation



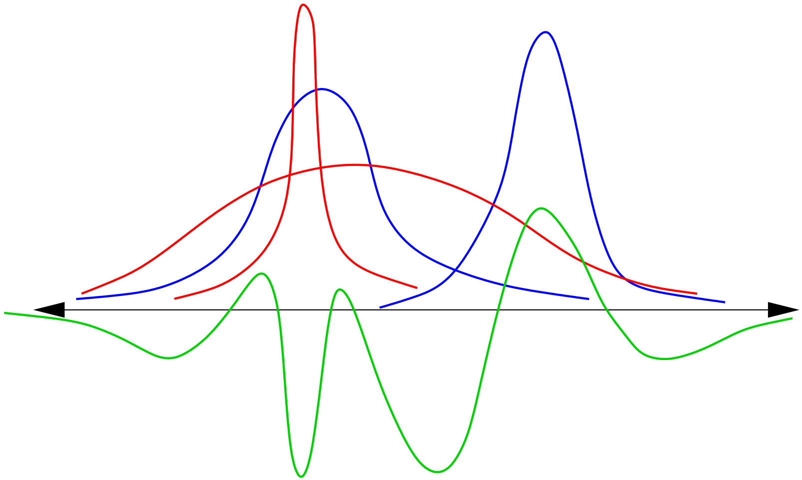
Zero Crossings and the Heat Equation



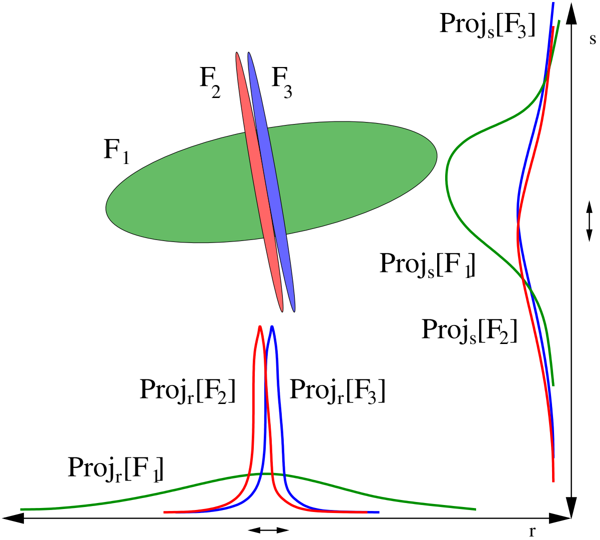
Zero Crossings and the Heat Equation



Zero Crossings and the Heat Equation



Thanks!


Proj$_s$[F$_3$]
s
F$_2$
F$_3$
F$_1$
Proj$_s$[F$_1$]
Proj$_s$[F$_2$]
Proj$_r$[F$_2$]
Proj$_r$[F$_3$]
Proj$_r$[F$_1$]
r