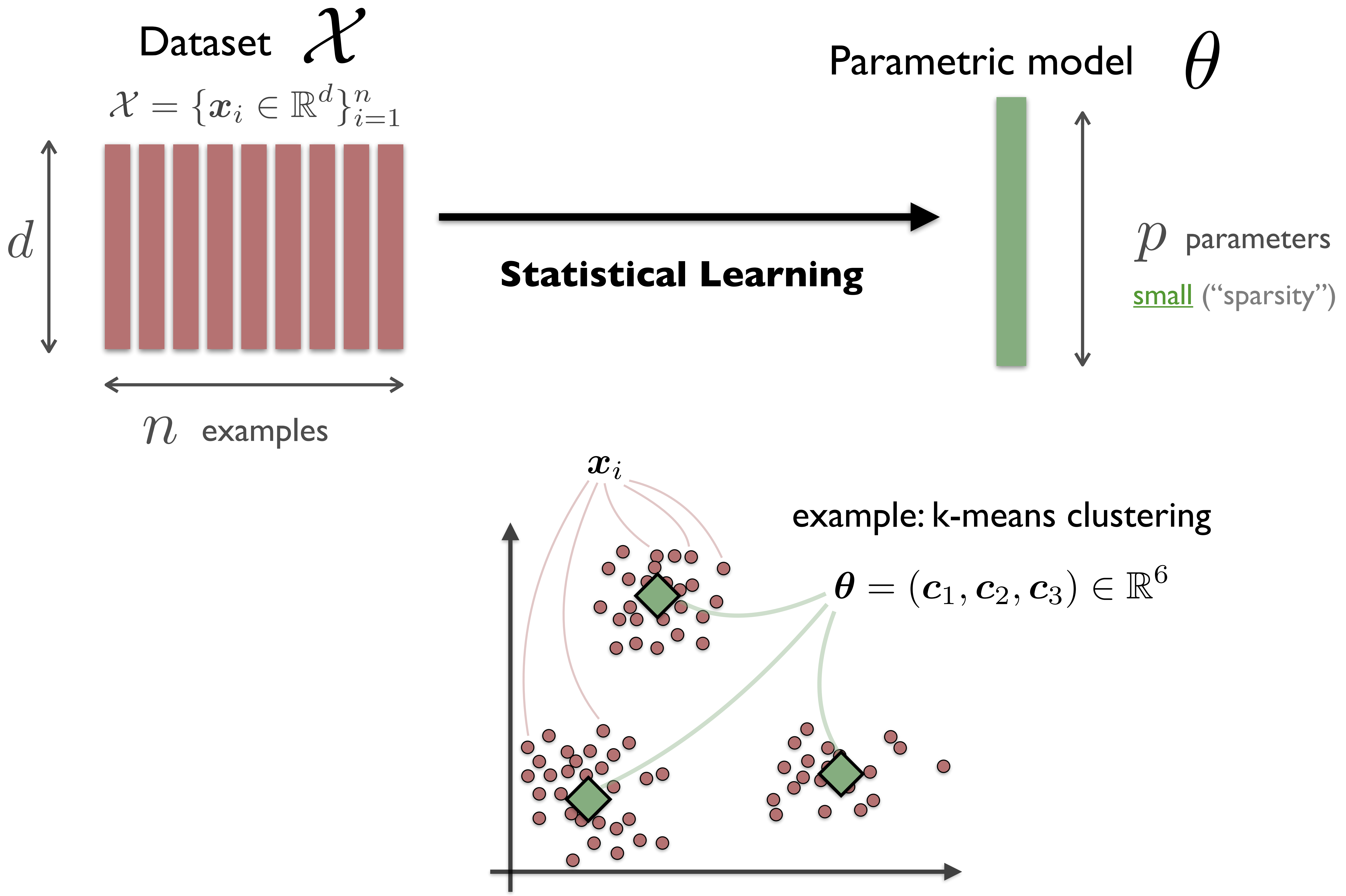
When compressive learning fails: blame the decoder or the sketch?

Vincent Schellekens Laurent Jacques

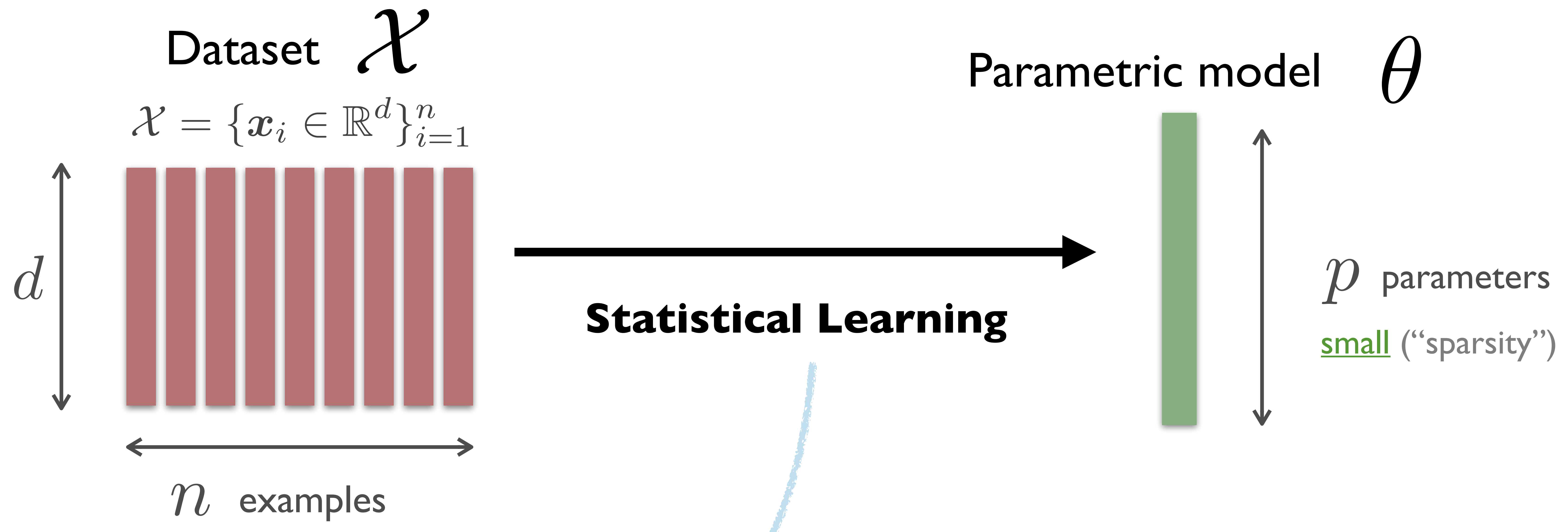
fnrs LA LIBERTÉ DE CHERCHER

UCLouvain

Context: large-scale learning



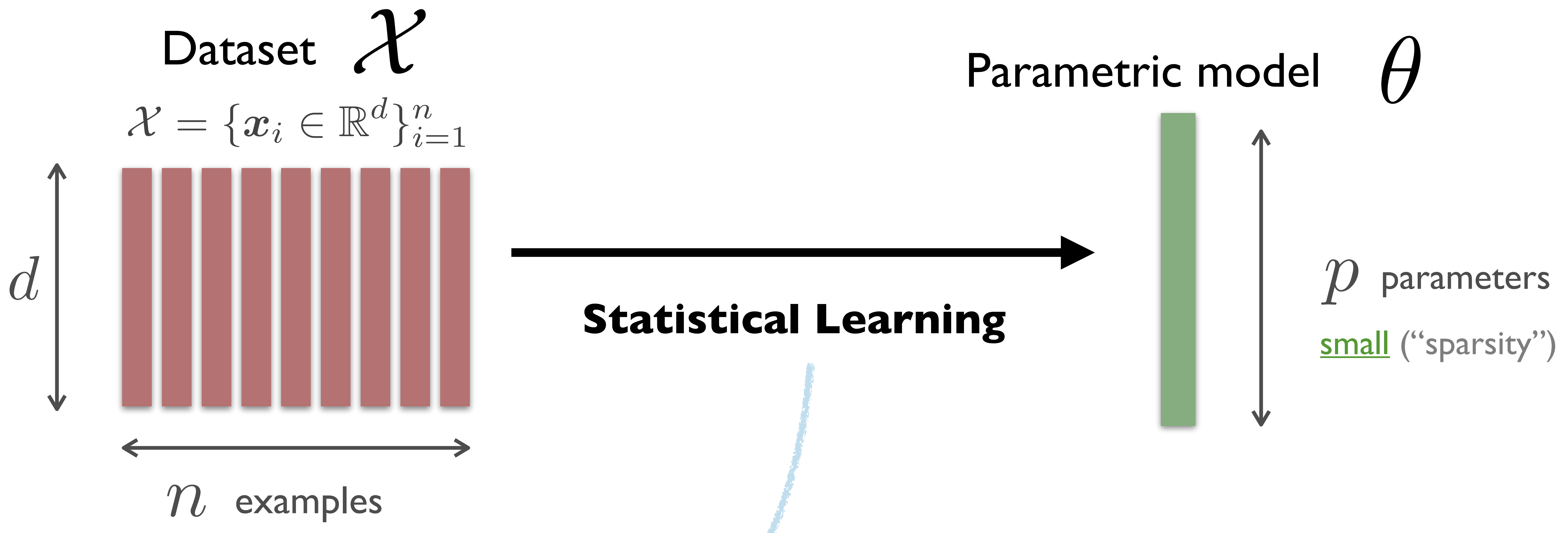
Context: large-scale learning



Choose parameters minimizing the risk

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} R(\boldsymbol{\theta} \mid \mathcal{X}) \quad \text{with} \quad R(\boldsymbol{\theta} \mid \mathcal{X}) = \frac{1}{n} \sum_{i=1}^{n} \ell(\boldsymbol{\theta}, \boldsymbol{x}_i)$$

Context: large-scale learning



Choose parameters minimizing the risk

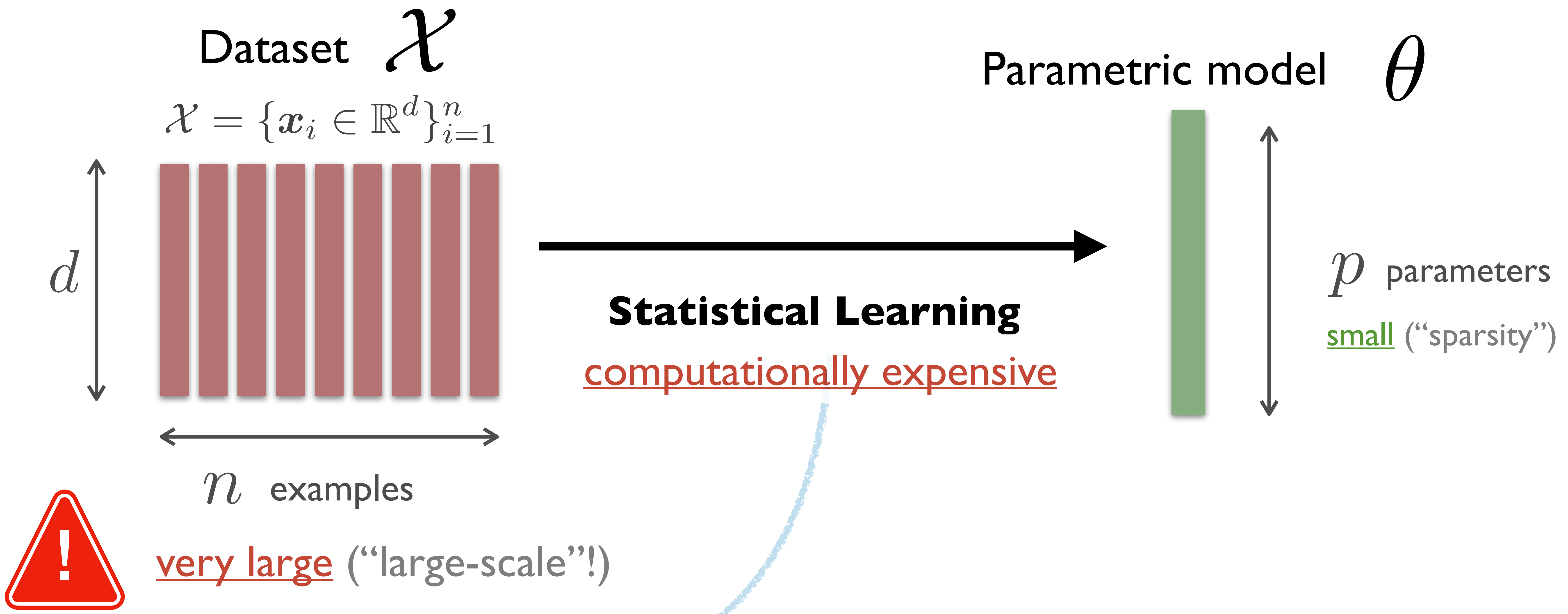
$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} R(\boldsymbol{\theta} \mid \mathcal{X}) \quad \text{with} \quad R(\boldsymbol{\theta} \mid \mathcal{X}) = \frac{1}{n}\sum_{i=1}^{n} \ell(\boldsymbol{\theta}, \boldsymbol{x}_i)$$

Examples:

k-means clustering: $$R(\boldsymbol{\theta}; \mathcal{X}) = \frac{1}{n}\sum_{i=1}^{n} \min_k \|\boldsymbol{c}_k - \boldsymbol{x}_i\|_2$$

Gaussian Mixture Model: $$R(\boldsymbol{\theta}; \mathcal{X}) = \frac{1}{n}\sum_{i=1}^{n} -\log(p_{\boldsymbol{\theta}}(\boldsymbol{x}_i))$$

Context: large-scale learning

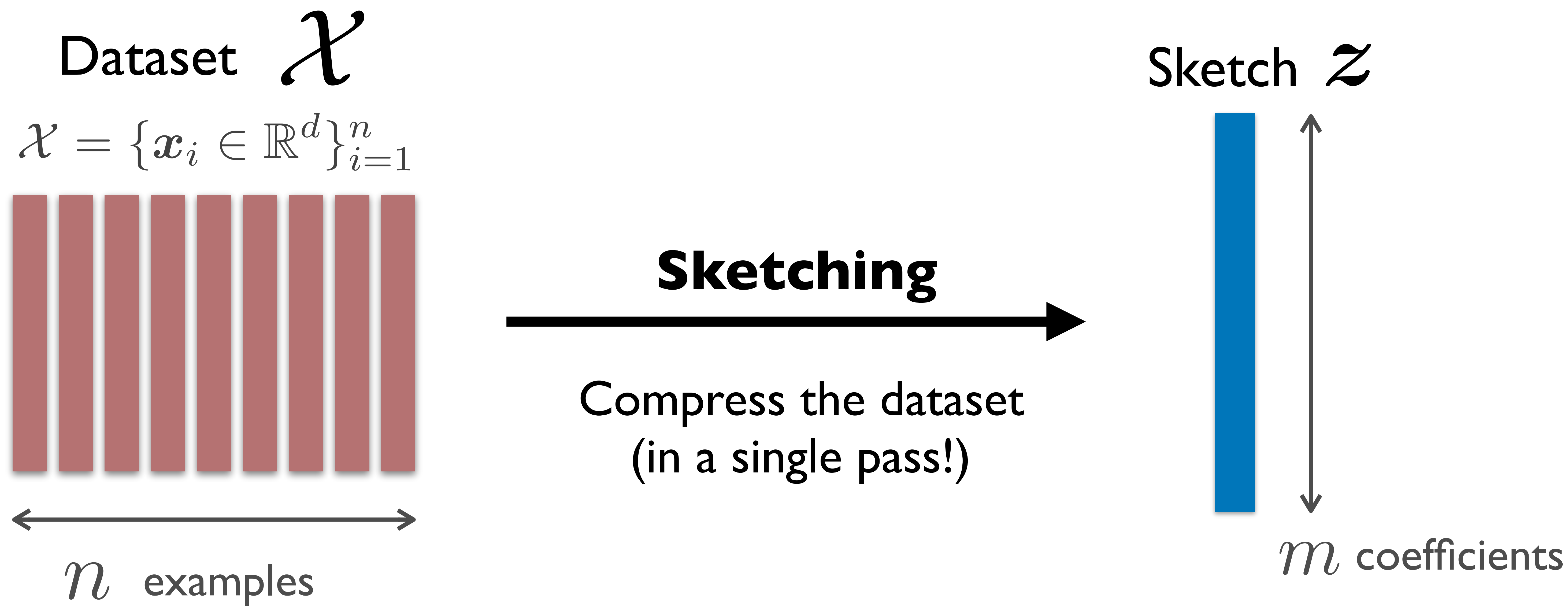


Choose parameters minimizing the risk

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} R(\boldsymbol{\theta} \mid \mathcal{X}) \quad \text{with} \quad R(\boldsymbol{\theta} \mid \mathcal{X}) = \frac{1}{n} \sum_{i=1}^{n} \ell(\boldsymbol{\theta}, \boldsymbol{x}_i)$$

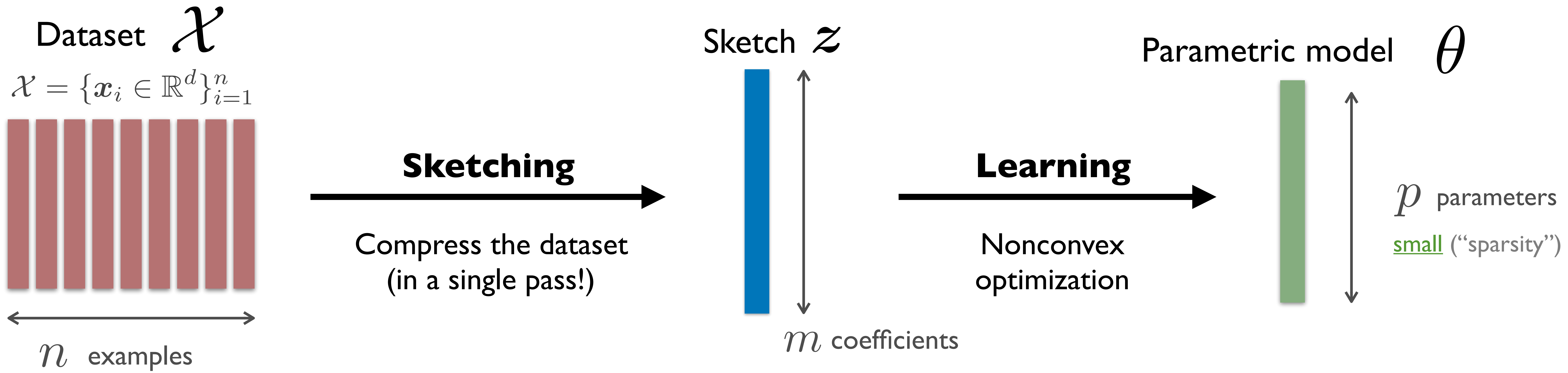
Typically, several passes over the dataset are needed

Compressive learning to the rescue



Break the training down into two "cheaper" steps

Compressive learning to the rescue



Break the training down into two "cheaper" steps

This work investigates the interplay between those steps

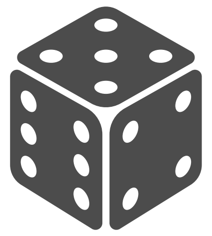
Sketching stage

Sketch = average (one pass!) of features $\Phi : \mathbb{R}^d \to \mathbb{C}^m$

$$\boldsymbol{z} = \frac{1}{n} \sum_{i=1}^{n} \Phi(\boldsymbol{x}_i)$$

Sketching stage

Sketch = average (one pass!) of features $\Phi : \mathbb{R}^d \to \mathbb{C}^m$



Typically, random features (random linear sensing of $\widehat{\mathcal{P}}_{\mathcal{X}}$)

$$\boldsymbol{z} = \frac{1}{n} \sum_{i=1}^{n} \Phi(\boldsymbol{x}_i)$$

"Low-dimensional embedding"
of the (empirical) data distribution

$$\boldsymbol{z} = \mathcal{A}(\widehat{\mathcal{P}}_{\mathcal{X}}) = \mathcal{A}(\tfrac{1}{n} \textstyle\sum_{i=1}^{n} \delta_{\boldsymbol{x}_i})$$

Sketching stage

Sketch = average (one pass!) of features $\Phi : \mathbb{R}^d \to \mathbb{C}^m$

Typically, random features (random linear sensing of $\widehat{\mathcal{P}}_{\mathcal{X}}$)

$$\boldsymbol{z} = \frac{1}{n} \sum_{i=1}^{n} \Phi(\boldsymbol{x}_i)$$

"Low-dimensional embedding" of the (empirical) data distribution

$$\boldsymbol{z} = \mathcal{A}(\widehat{\mathcal{P}}_{\mathcal{X}}) = \mathcal{A}(\tfrac{1}{n} \textstyle\sum_{i=1}^{n} \delta_{\boldsymbol{x}_i})$$

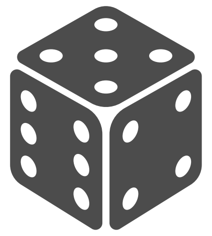
In this work: random Fourier features

$$\Phi_j(\boldsymbol{x}) = \exp(\mathrm{i}\boldsymbol{\omega}_j^\top \boldsymbol{x}), \quad j = 1, ..., m$$

$$\boldsymbol{\omega}_j \sim_{i.i.d.} \Lambda(\boldsymbol{\omega}) = (\mathcal{F}K)(\boldsymbol{\omega})$$

Sketching stage

Sketch = average (one pass!) of features $\Phi : \mathbb{R}^d \to \mathbb{C}^m$



$$\boldsymbol{z} = \frac{1}{n}\sum_{i=1}^{n} \Phi(\boldsymbol{x}_i)$$

Typically, random features (random linear sensing of $\widehat{\mathcal{P}}_{\mathcal{X}}$)

“Low-dimensional embedding”
of the (empirical) data distribution

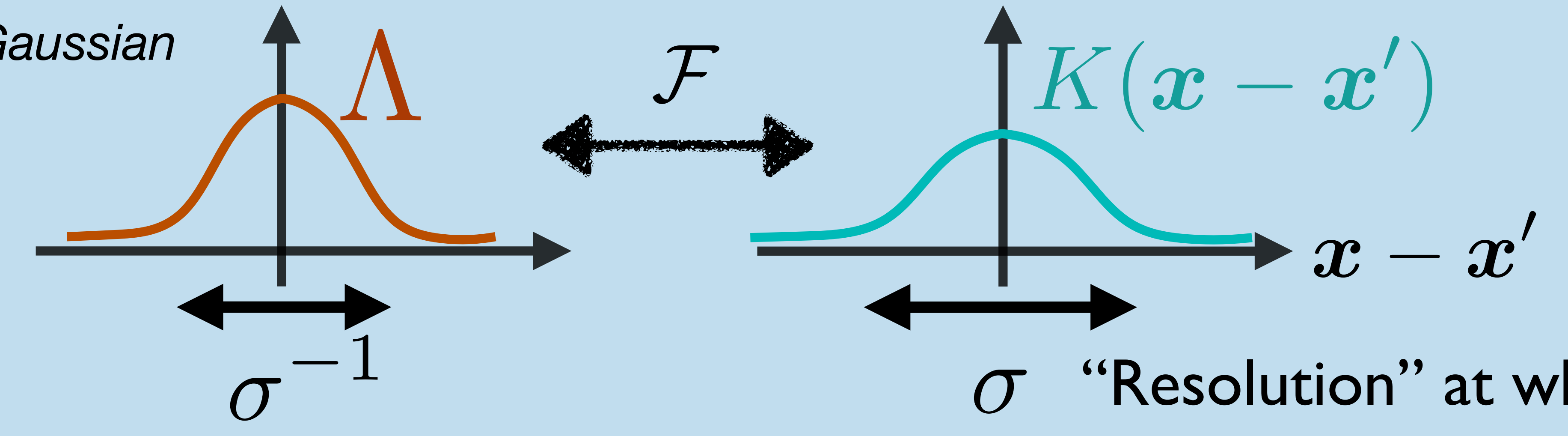
$$\boldsymbol{z} = \mathcal{A}(\widehat{\mathcal{P}}_{\mathcal{X}}) = \mathcal{A}(\tfrac{1}{n}\textstyle\sum_{i=1}^{n} \delta_{\boldsymbol{x}_i})$$

In this work: random Fourier features

$$\Phi_j(\boldsymbol{x}) = \exp(\mathrm{i}\boldsymbol{\omega}_j^\top \boldsymbol{x}), \quad j = 1, \dots, m$$

$$\boldsymbol{\omega}_j \sim_{i.i.d.} \Lambda(\boldsymbol{\omega}) = (\mathcal{F}K)(\boldsymbol{\omega})$$

e.g. Gaussian



“Resolution” at which we capture distributions

Learning stage

Optimize some "sketch matching" loss

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{z})$$

Learning stage

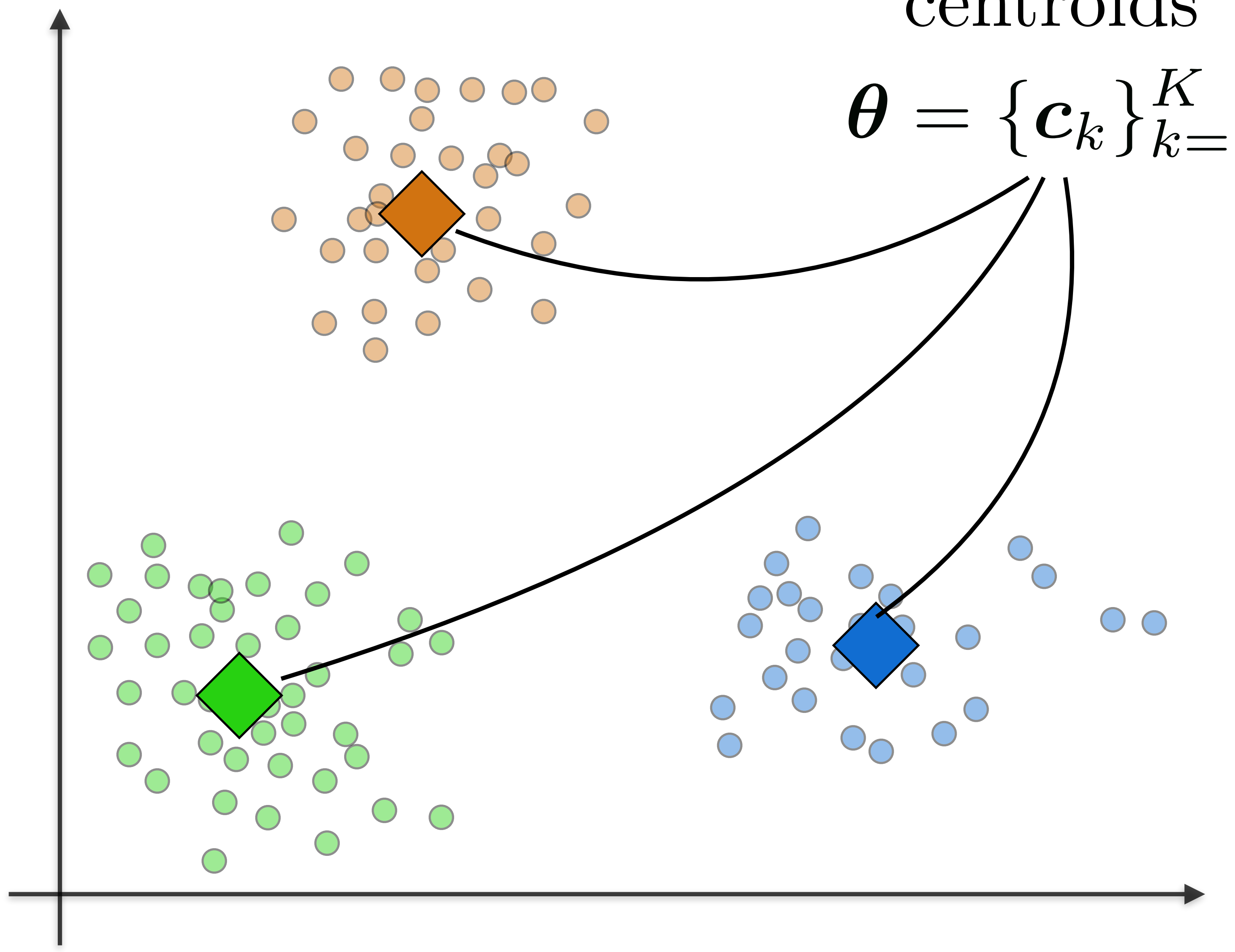
Optimize some “sketch matching” loss

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{z})$$

Example: k-means $R(\boldsymbol{\theta}; \mathcal{X}) = \sum_i \min_k \|\boldsymbol{c}_k - \boldsymbol{x}_i\|_2$

centroids

$\boldsymbol{\theta} = \{\boldsymbol{c}_k\}_{k=1}^K$

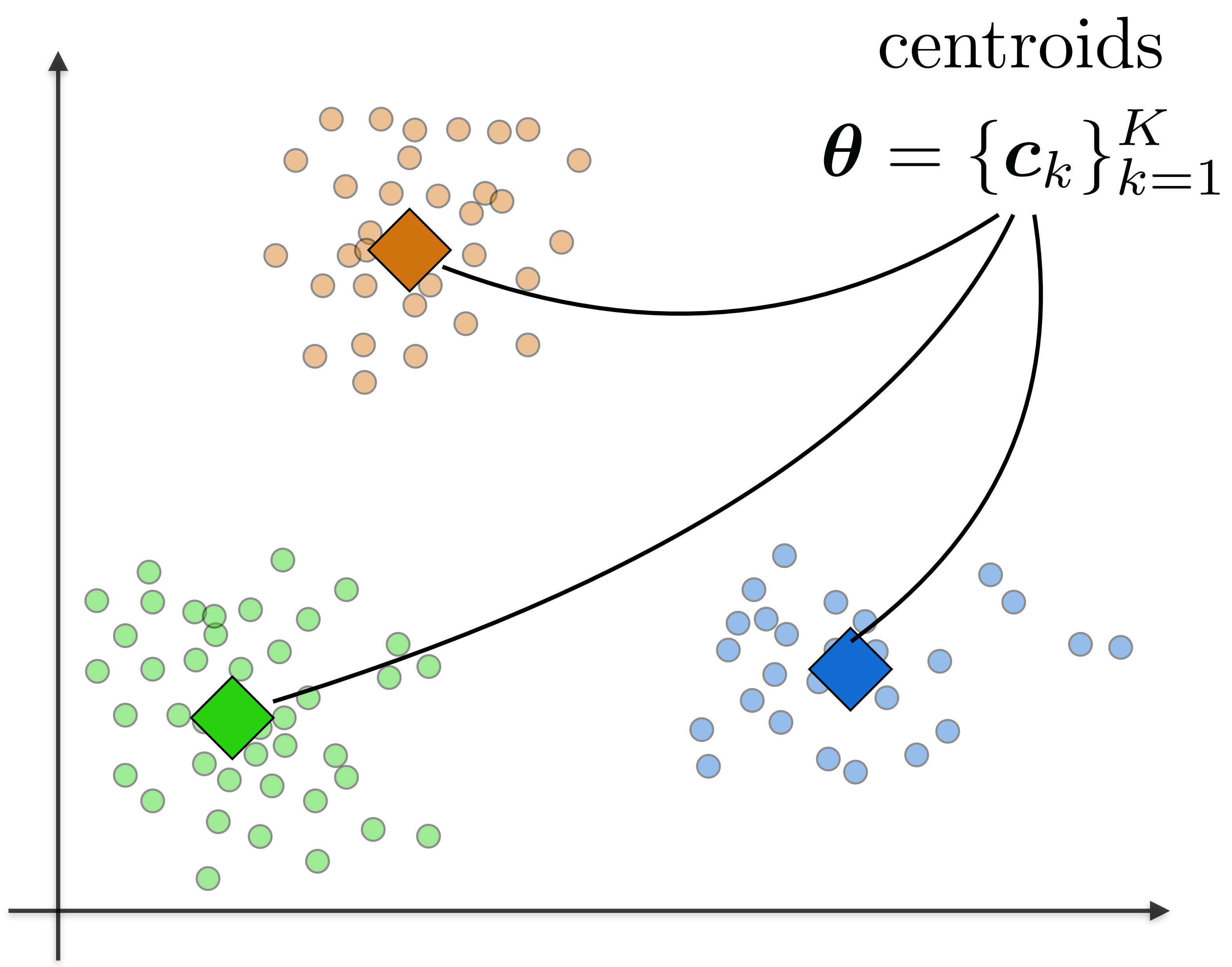


Learning stage

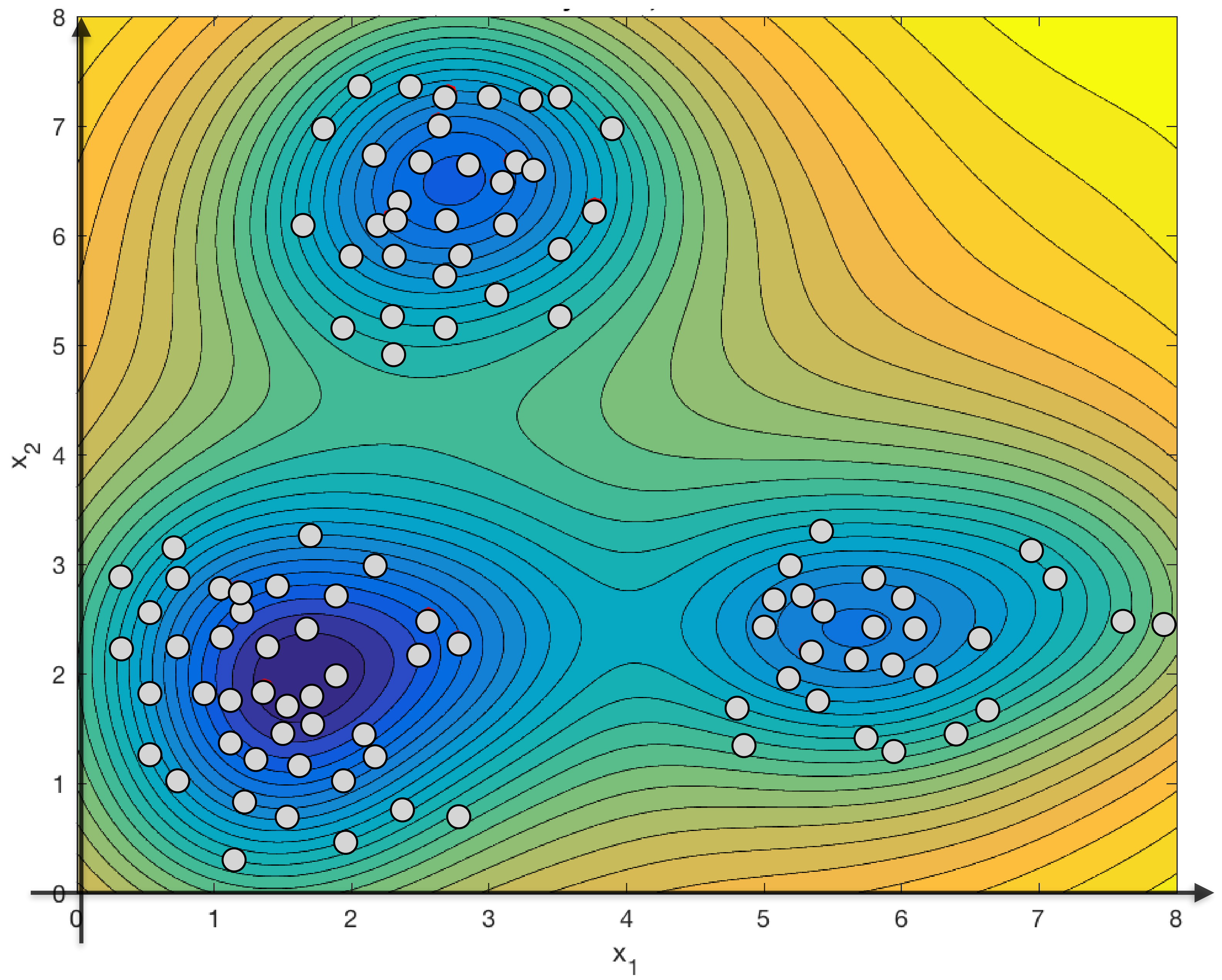
Optimize some “sketch matching” loss

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{z})$$

Example: k-means $R(\boldsymbol{\theta}; \mathcal{X}) = \sum_i \min_k \|\boldsymbol{c}_k - \boldsymbol{x}_i\|_2$

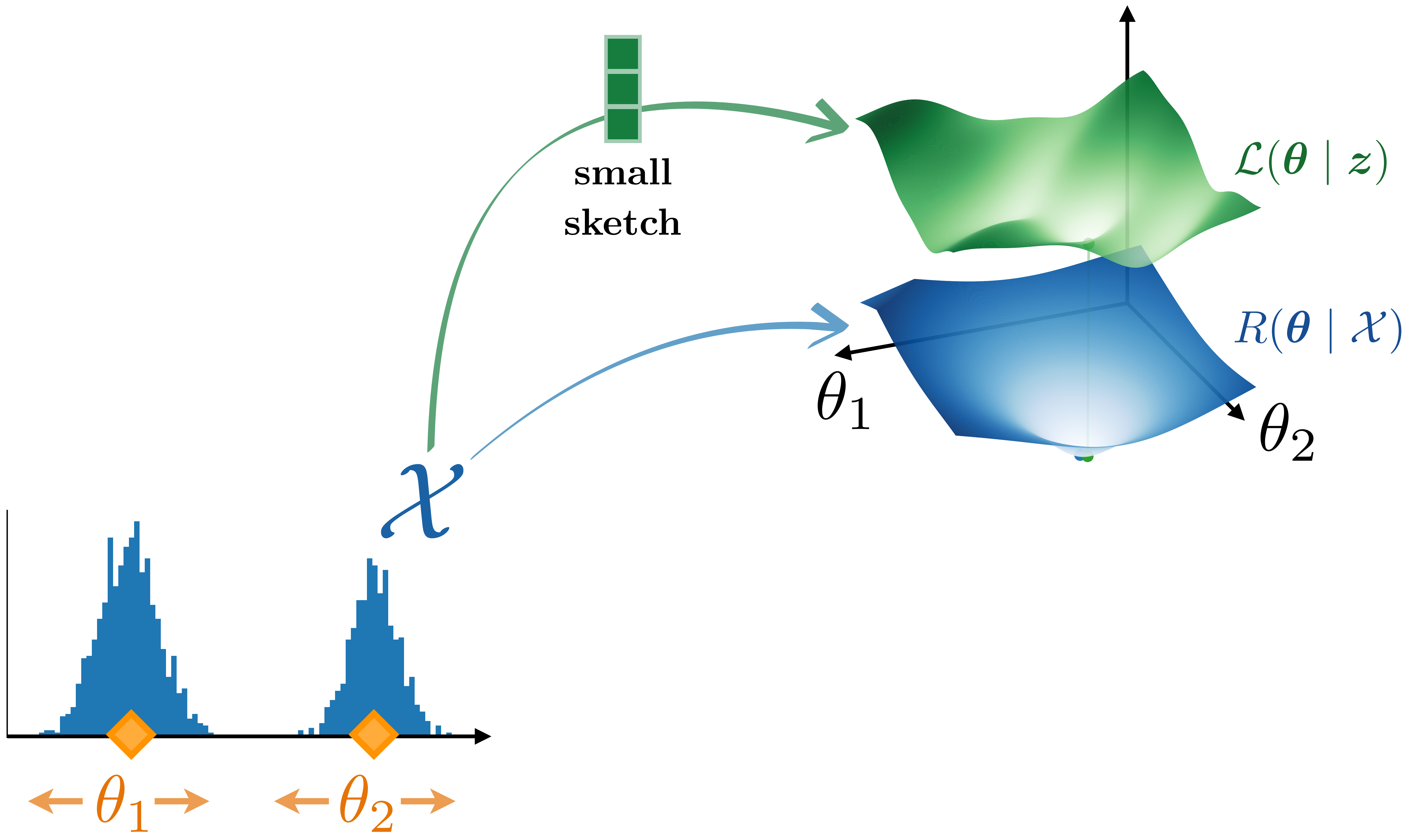


$$\mathcal{L}(\boldsymbol{\theta}; \boldsymbol{z}) = \|\boldsymbol{z} - \underbrace{\tfrac{1}{K} \textstyle\sum_{k=1}^K \Phi(\boldsymbol{c}_k)}_{\text{sketch of the centroids}}\|_2$$



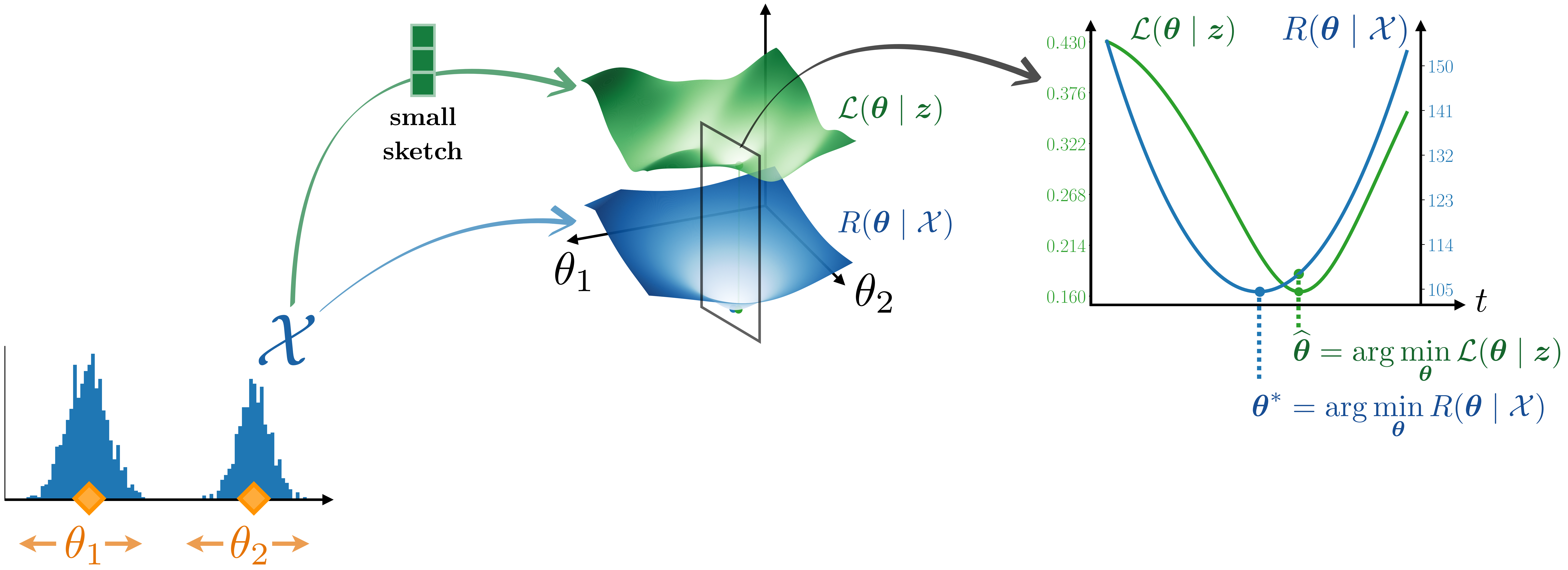
Learning stage

Cost function vs. true risk



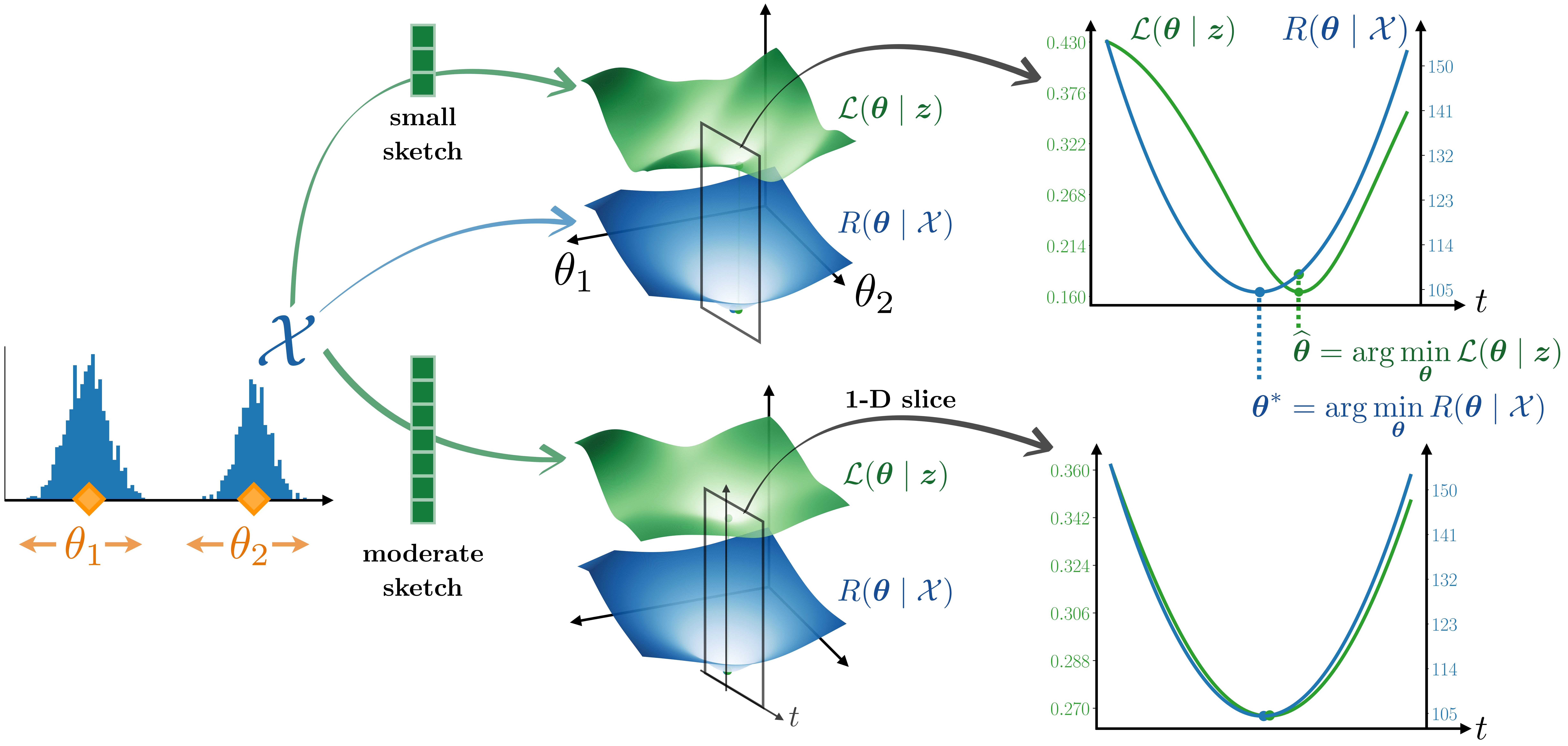
Learning stage

Cost function vs. true risk



Learning stage

Cost function vs. true risk



Learning stage

Optimize some "sketch matching" loss

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{z})$$

Nonconvex! In practice, we use a "decoder"

$$\boldsymbol{\theta}_\Delta = \Delta[\boldsymbol{z}]$$

The CLOMPR decoder (based on matching pursuit) showed promising empirical success...

Learning stage

Optimize some "sketch matching" loss

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{z})$$

Nonconvex! In practice, we use a "decoder"

$$\boldsymbol{\theta}_\Delta = \Delta[\boldsymbol{z}]$$

The CLOMPR decoder (based on matching pursuit) showed promising empirical success...

Our question: should we improve the decoder and/or sketch?
If so, when? What can we expect to gain?

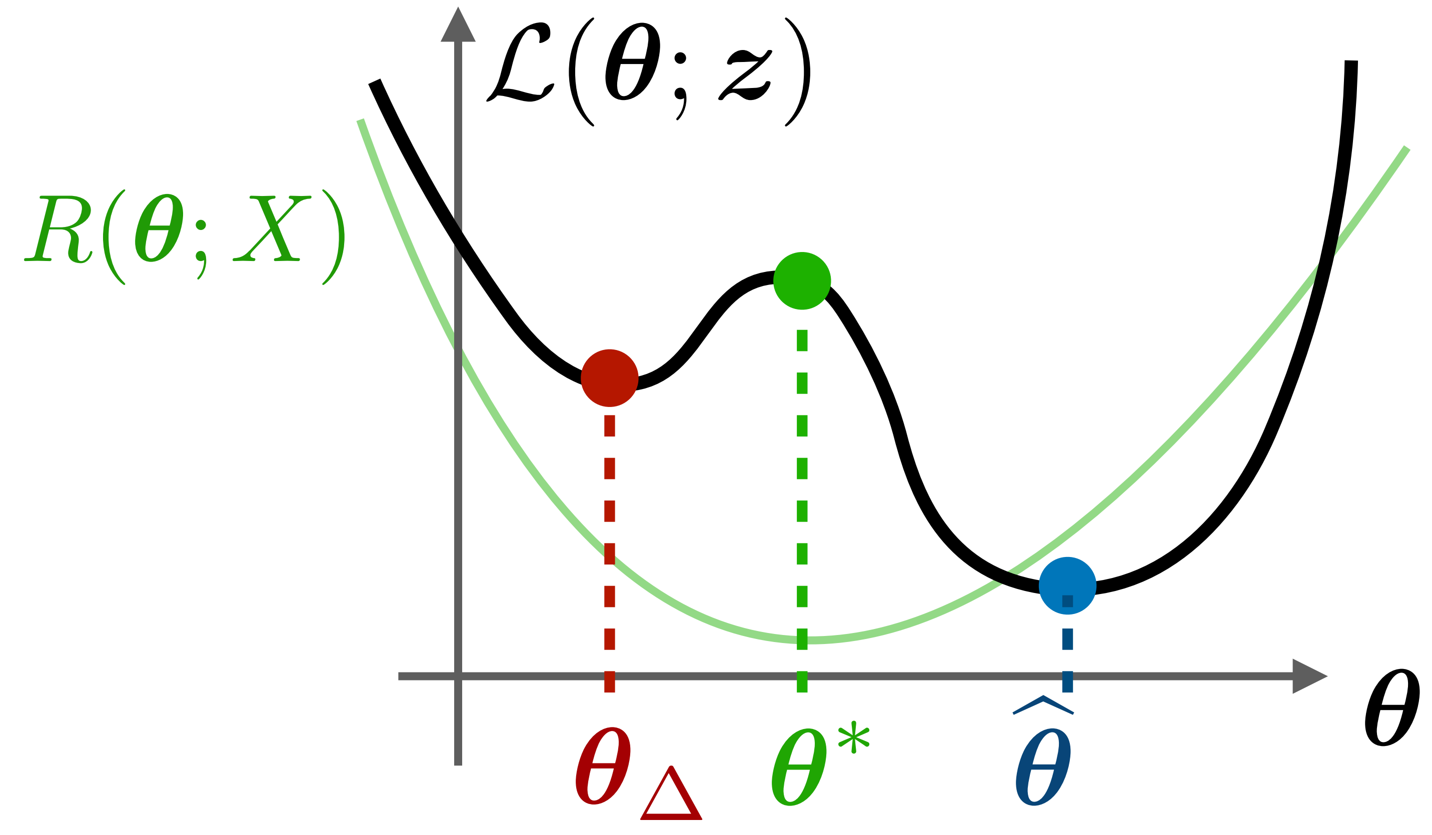
Methodology

We define 5 “failure scenarii”

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} R(\boldsymbol{\theta}; X)$$

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{z})$$

$$\boldsymbol{\theta}_\Delta = \Delta[\boldsymbol{z}]$$



$a)$

Incorrect global minimum

Decoder fails to find it

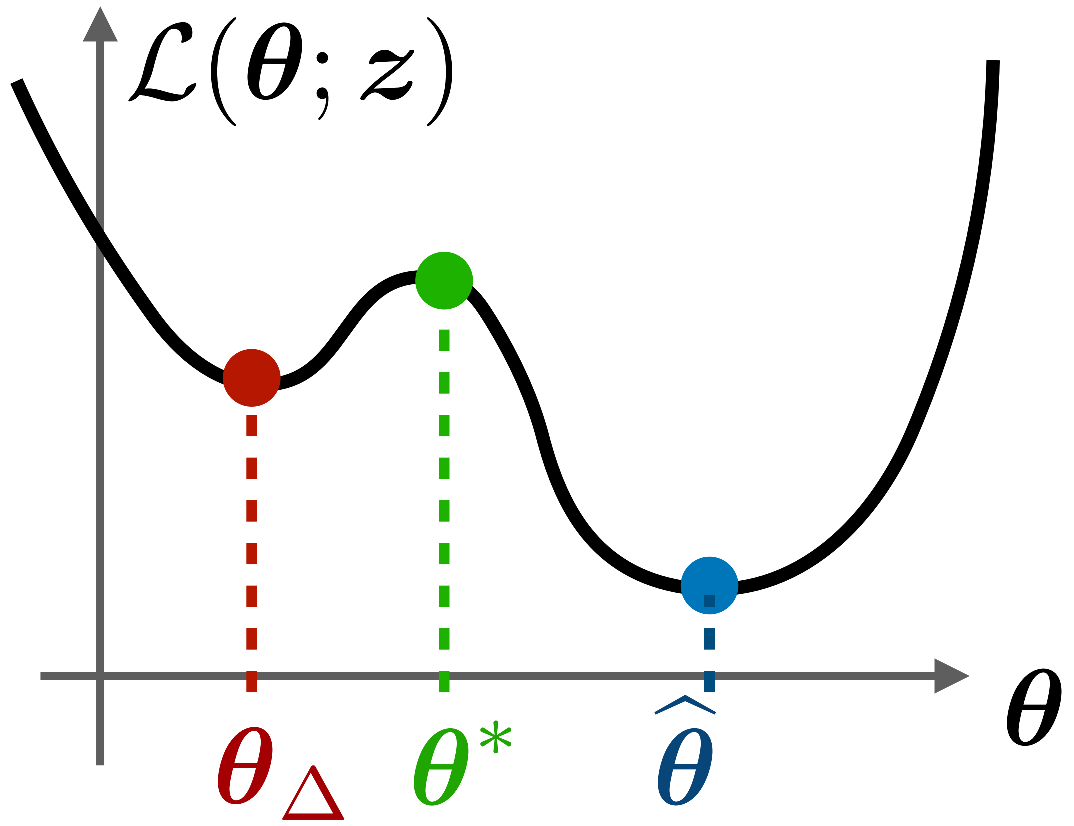
Methodology

We define 5 "failure scenarii"

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} R(\boldsymbol{\theta}; X)$$

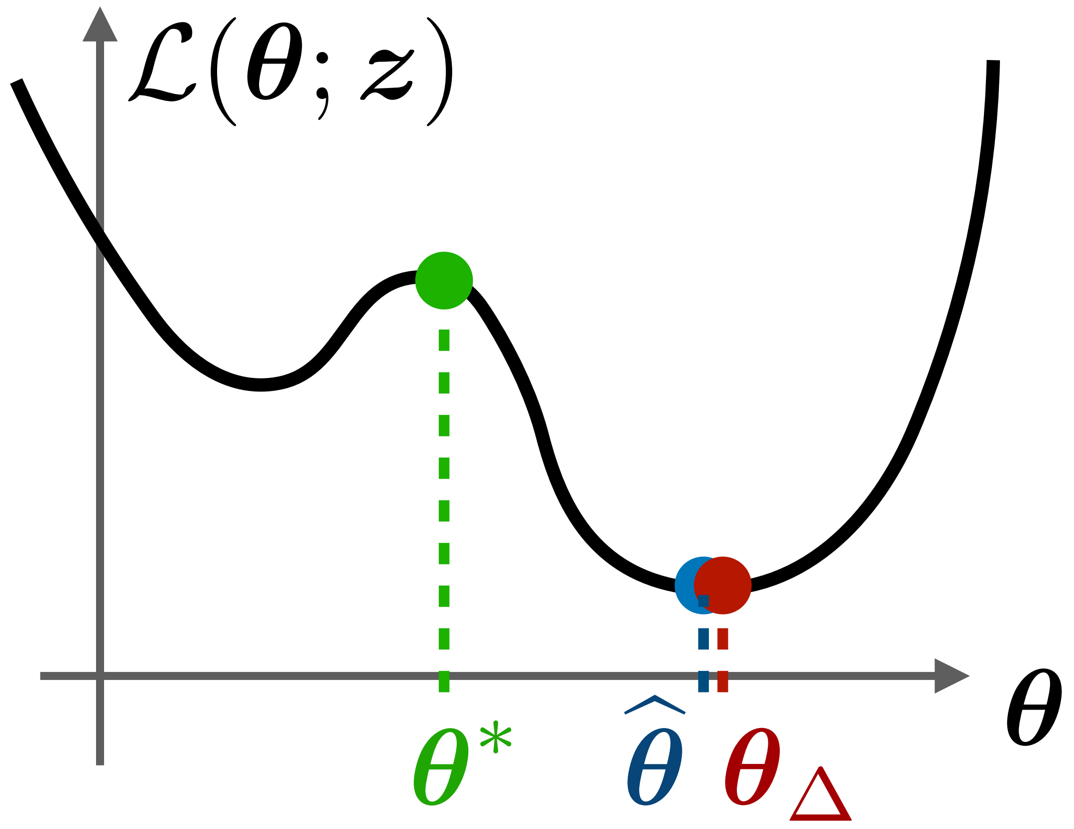
$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{z})$$

$$\boldsymbol{\theta}_\Delta = \Delta[\boldsymbol{z}]$$



$a)$

Incorrect global minimum
Decoder fails to find it



$b)$

Incorrect global minimum
Decoder finds it

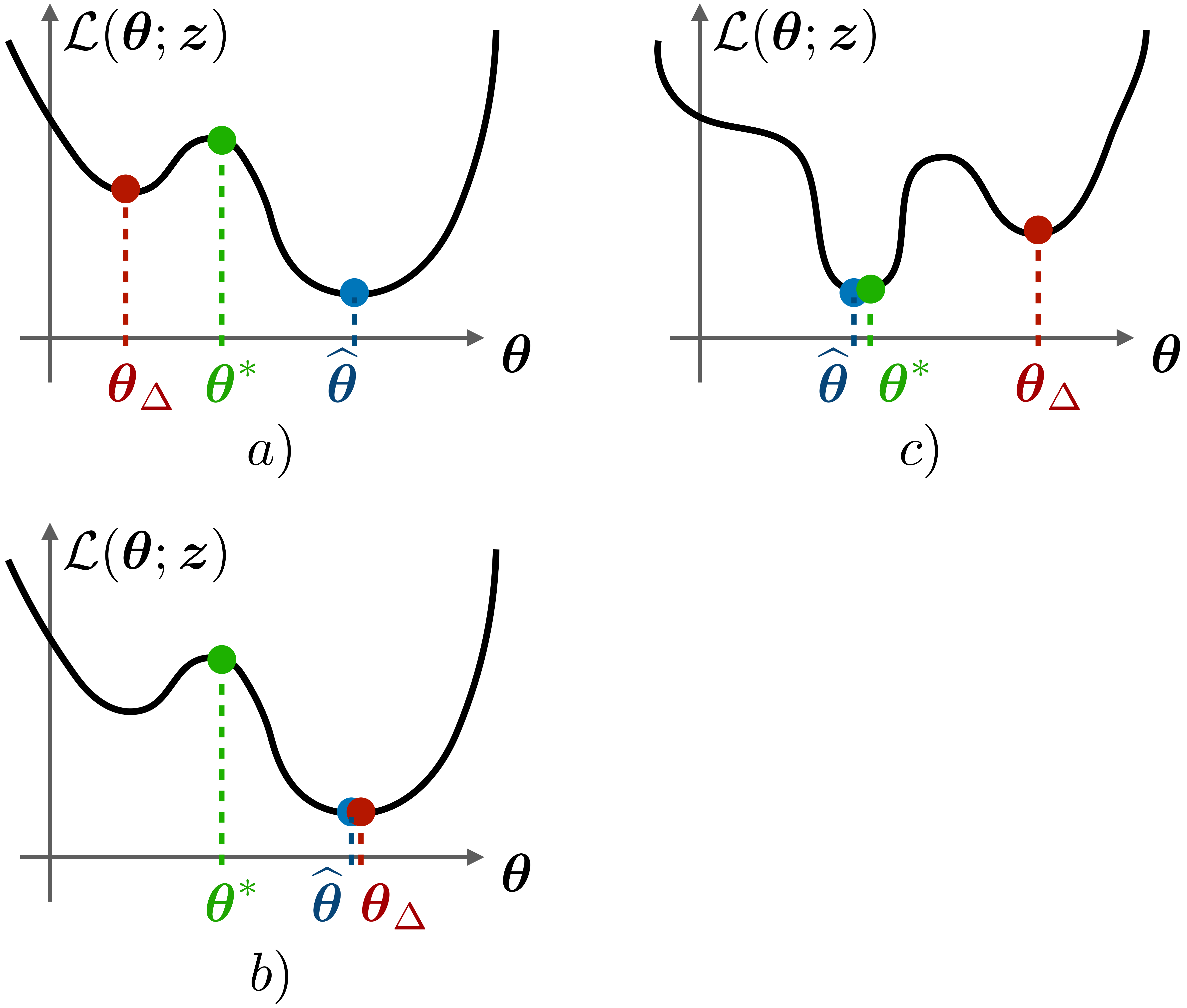
Methodology

We define 5 “failure scenarii”

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} R(\boldsymbol{\theta}; X)$$

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{z})$$

$$\boldsymbol{\theta}_\Delta = \Delta[\boldsymbol{z}]$$



Correct global minimum

Decoder fails to find it

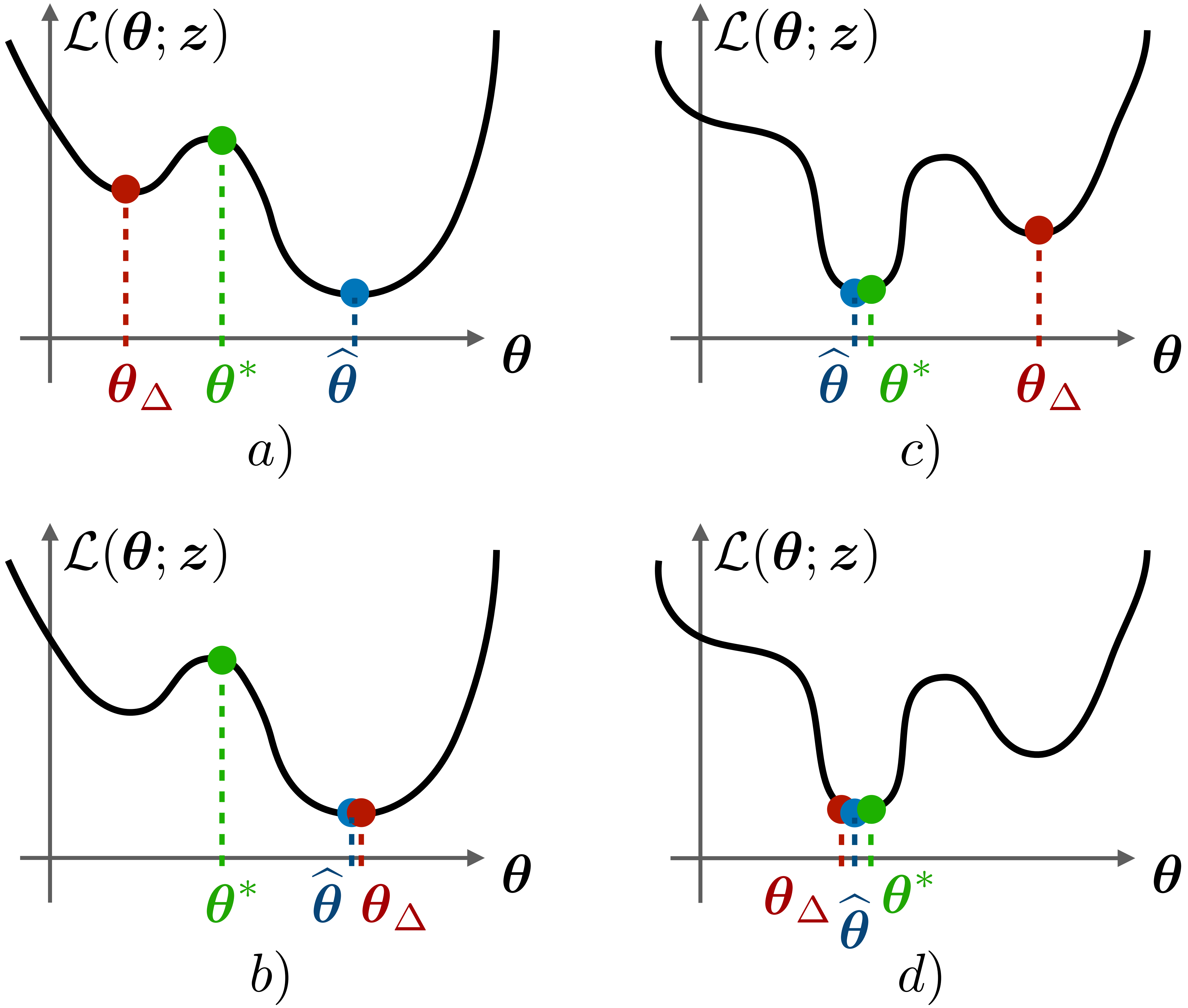
Methodology

We define 5 "failure scenarii"

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} R(\boldsymbol{\theta}; X)$$

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{z})$$

$$\boldsymbol{\theta}_\Delta = \Delta[\boldsymbol{z}]$$



Correct global minimum
Decoder fails to find it

Correct global minimum
Decoder finds it

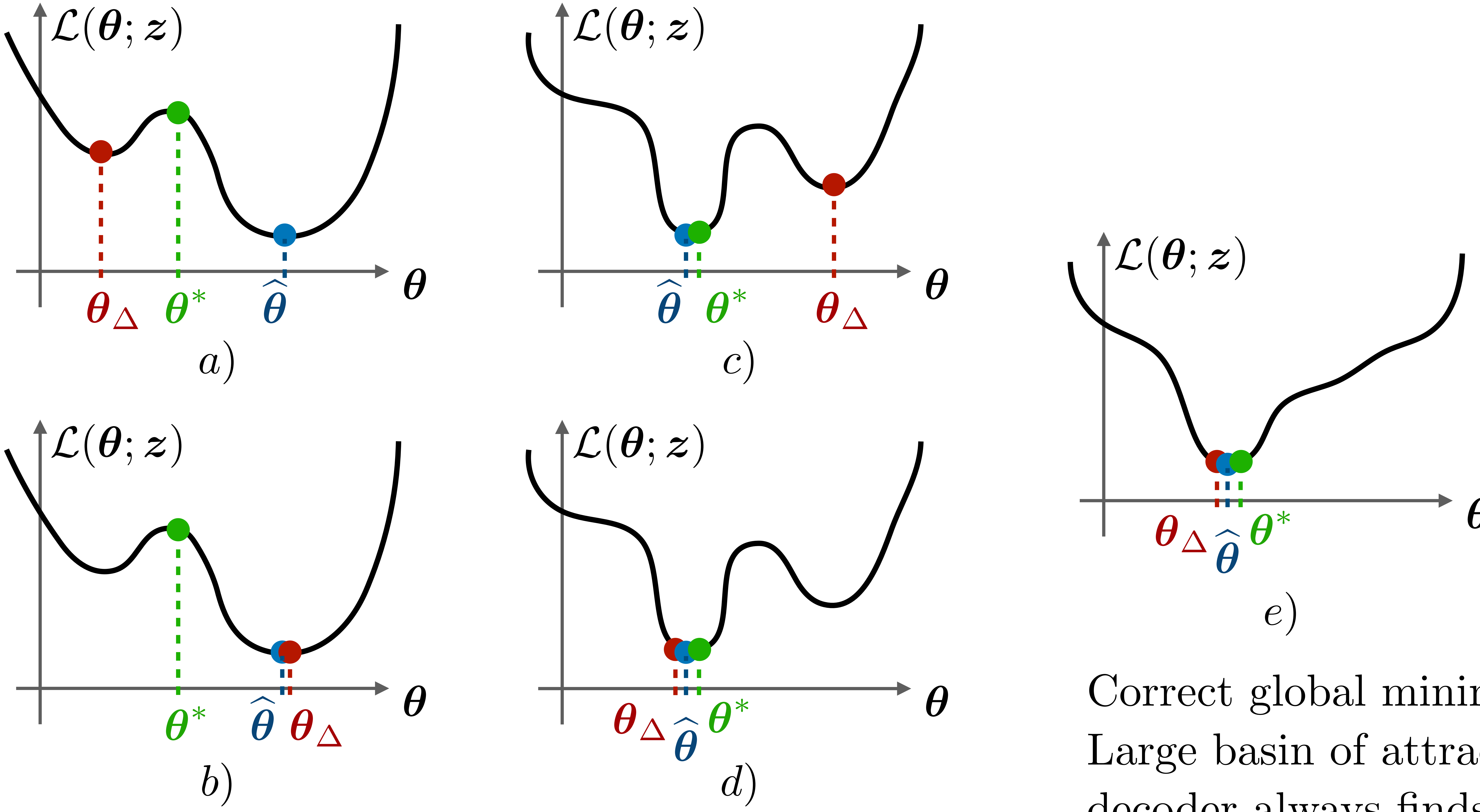
Methodology

We define 5 “failure scenarii”

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} R(\boldsymbol{\theta}; X)$$

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{z})$$

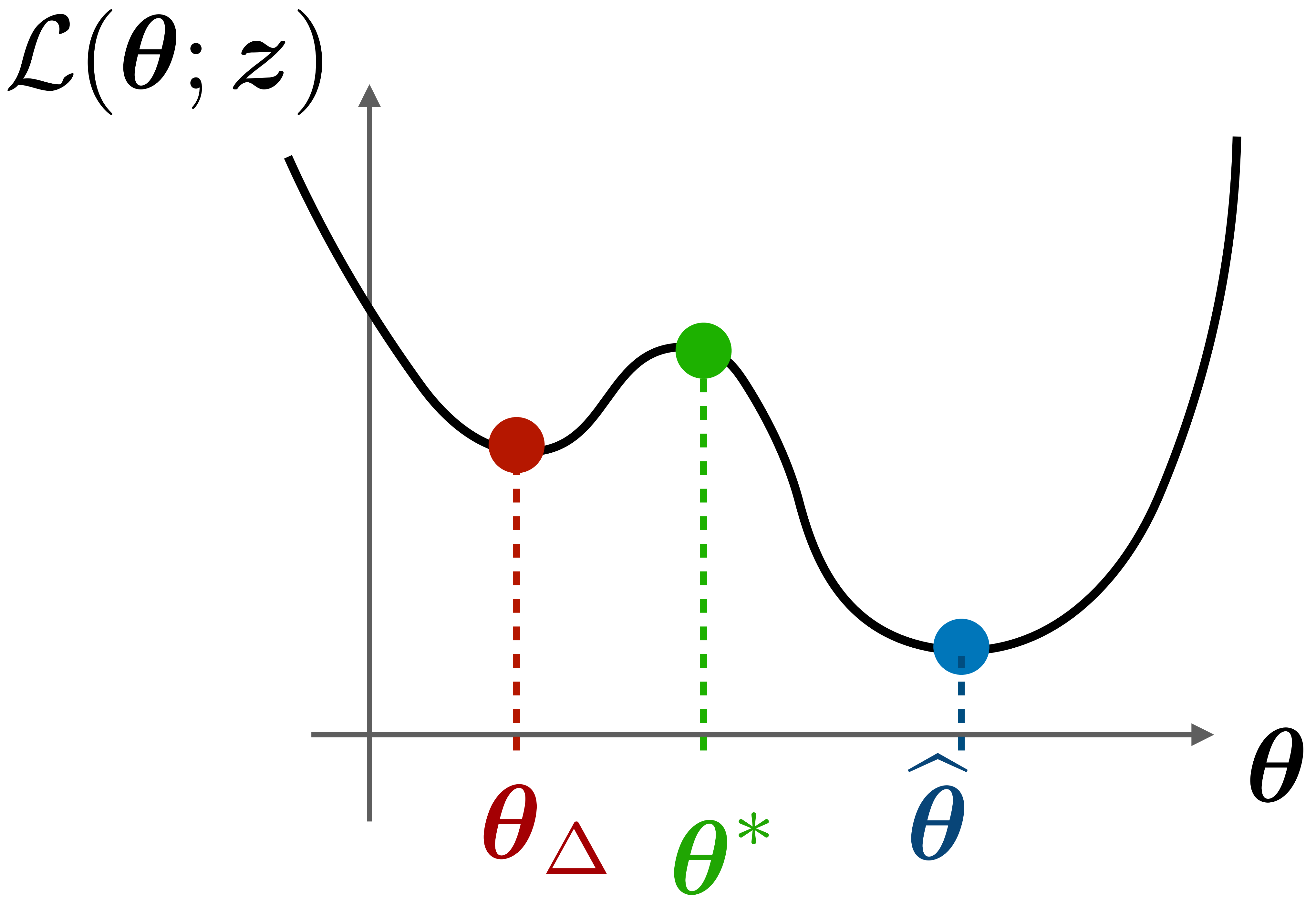
$$\boldsymbol{\theta}_\Delta = \Delta[\boldsymbol{z}]$$



Correct global minimum
Large basin of attraction,
decoder <u>always</u> finds it

Methodology

What do we know?



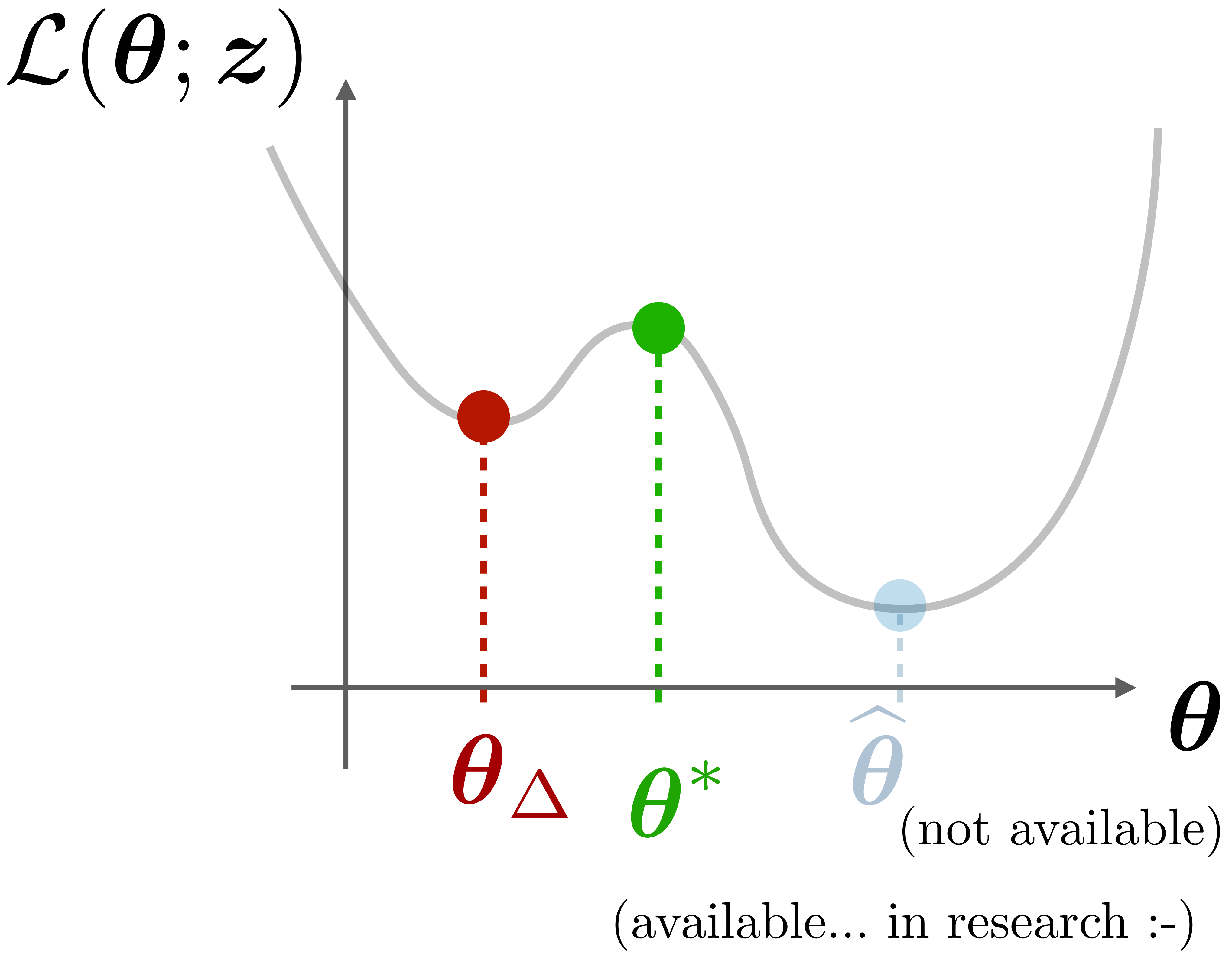
$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} R(\boldsymbol{\theta}; X)$$

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{z})$$

$$\boldsymbol{\theta}_\Delta = \Delta[\boldsymbol{z}]$$

Methodology

What do we know?
Two points...



$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} R(\boldsymbol{\theta}; X)$$

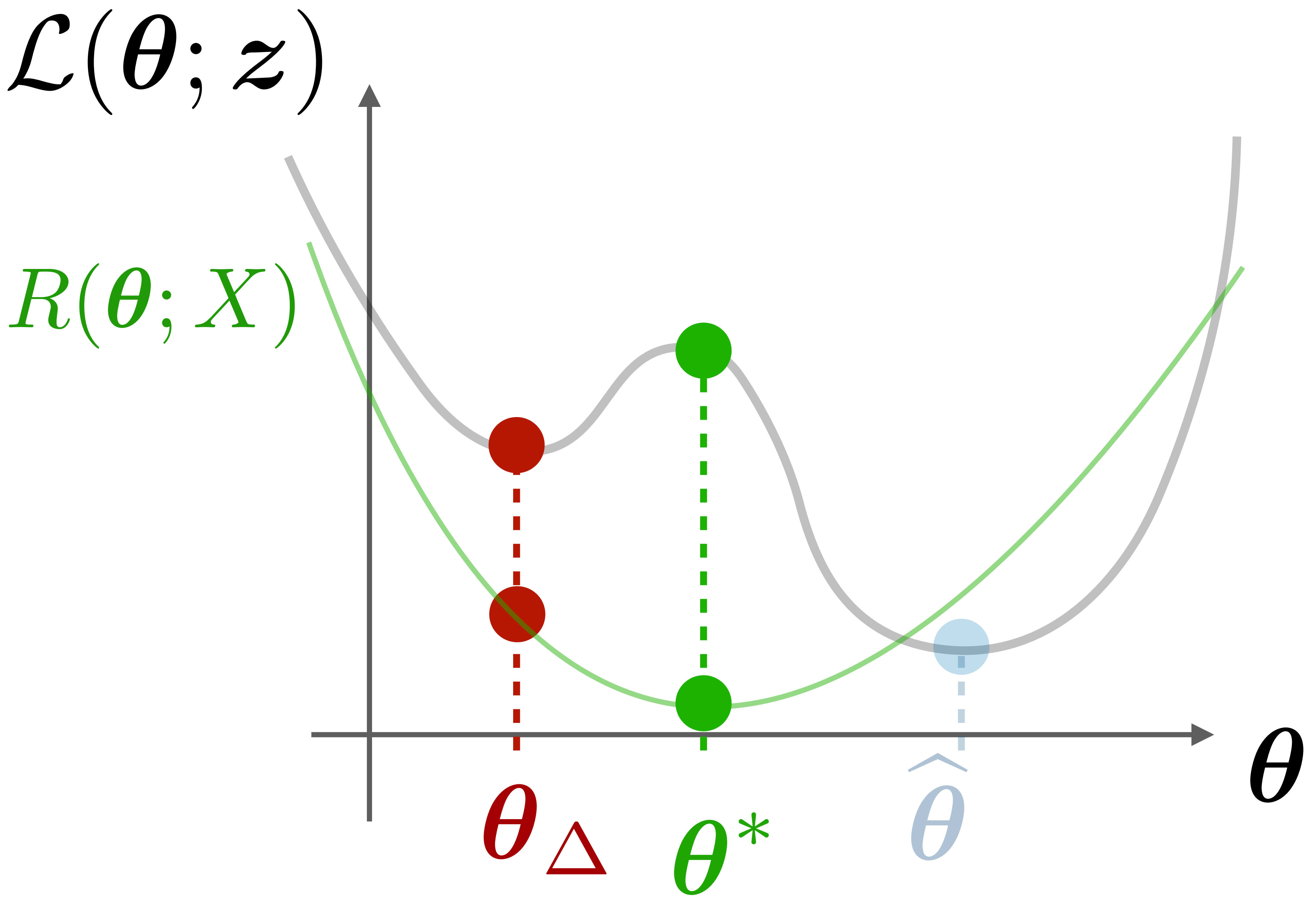
$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{z})$$

$$\boldsymbol{\theta}_\Delta = \Delta[\boldsymbol{z}]$$

Methodology

What do we know?

Two points, two values!



$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} R(\boldsymbol{\theta}; X)$$

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{z})$$

$$\boldsymbol{\theta}_\Delta = \Delta[\boldsymbol{z}]$$

Methodology

What do we know?

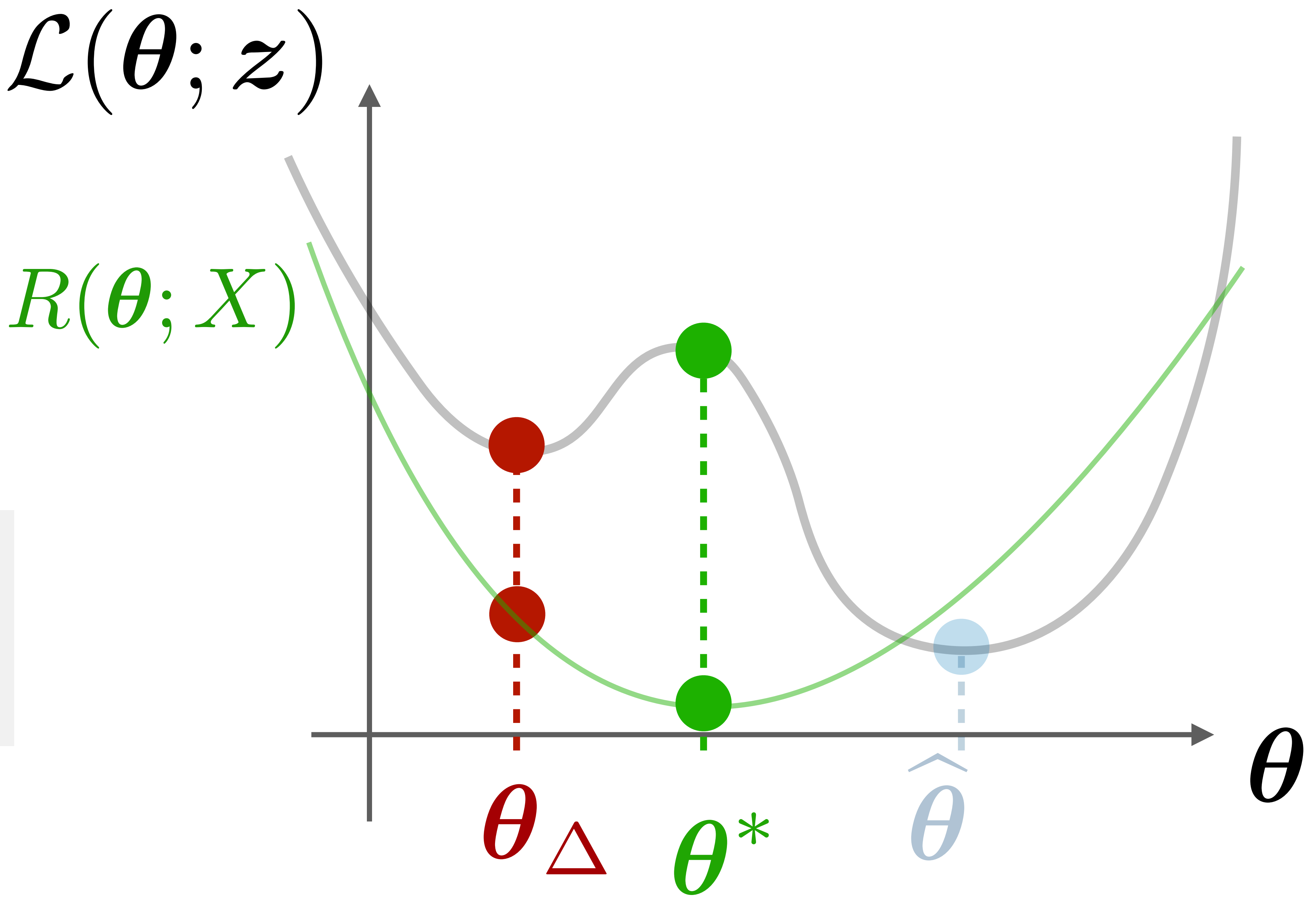
Two points, two values!

A simple test:

$$\mathcal{L}(\boldsymbol{\theta}_\Delta; \boldsymbol{z}) \overset{?}{>} \mathcal{L}(\boldsymbol{\theta}^*; \boldsymbol{z})$$

Yes

Decoder failed
(Sketch ?!)



$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} R(\boldsymbol{\theta}; X)$$

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{z})$$

$$\boldsymbol{\theta}_\Delta = \Delta[\boldsymbol{z}]$$

Methodology

What do we know?
Two points, two values!

A simple test:

$$\mathcal{L}(\boldsymbol{\theta}_\Delta; \boldsymbol{z}) \overset{?}{>} \mathcal{L}(\boldsymbol{\theta}^*; \boldsymbol{z})$$

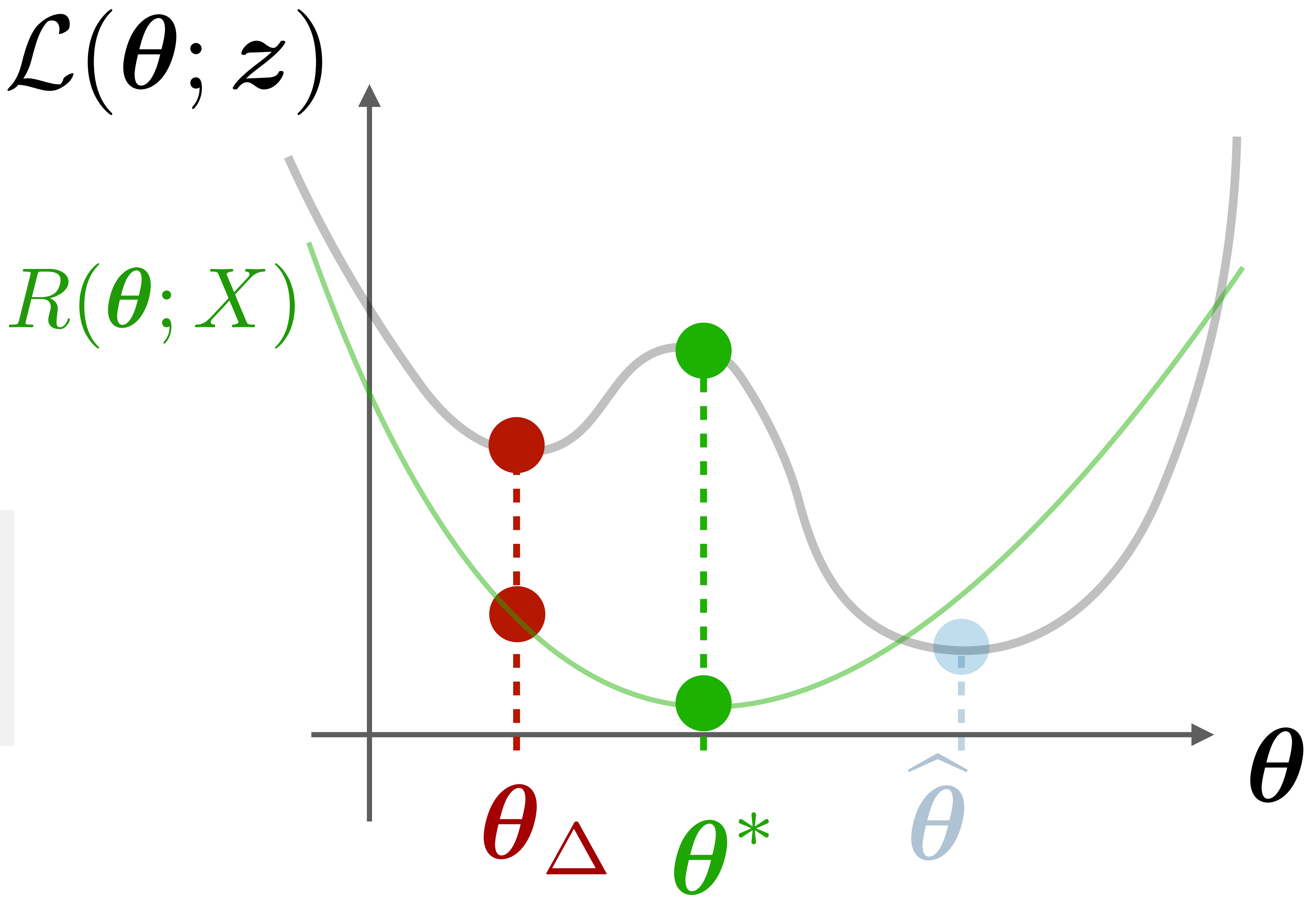
Yes → Decoder failed (Sketch ?!)

No →

$$R(\boldsymbol{\theta}_\Delta; X) \overset{?}{\gg} R(\boldsymbol{\theta}^*; X)$$

Yes → Sketch failed (Decoder ?!)

No $R(\boldsymbol{\theta}_\Delta; X) \simeq R(\boldsymbol{\theta}^*; X)$ → Success :-)



$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} R(\boldsymbol{\theta}; X)$$

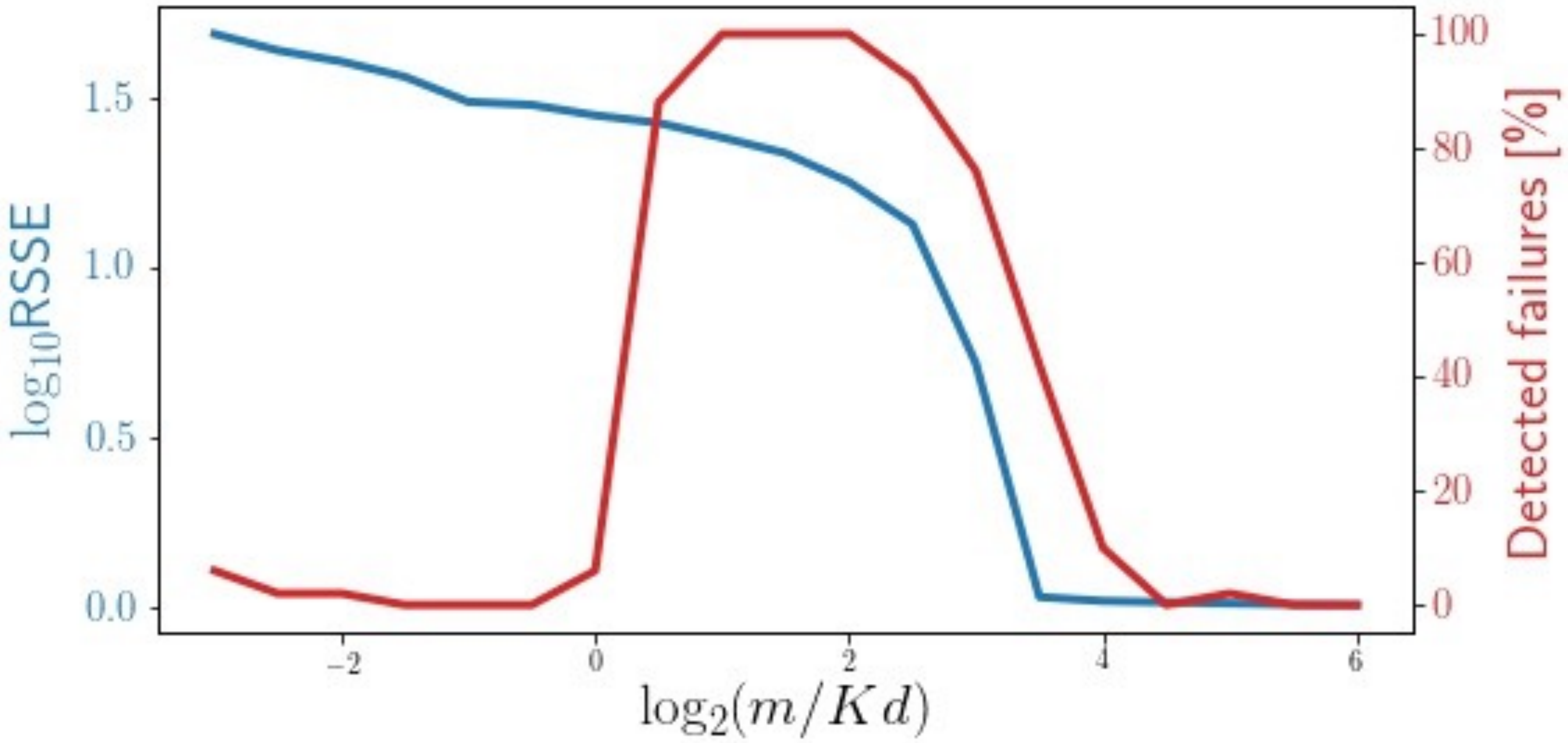
$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{z})$$

$$\boldsymbol{\theta}_\Delta = \Delta[\boldsymbol{z}]$$

Results: size of m (k-means)

Multiple sketches (50 draws)

Performance
(lower is better)

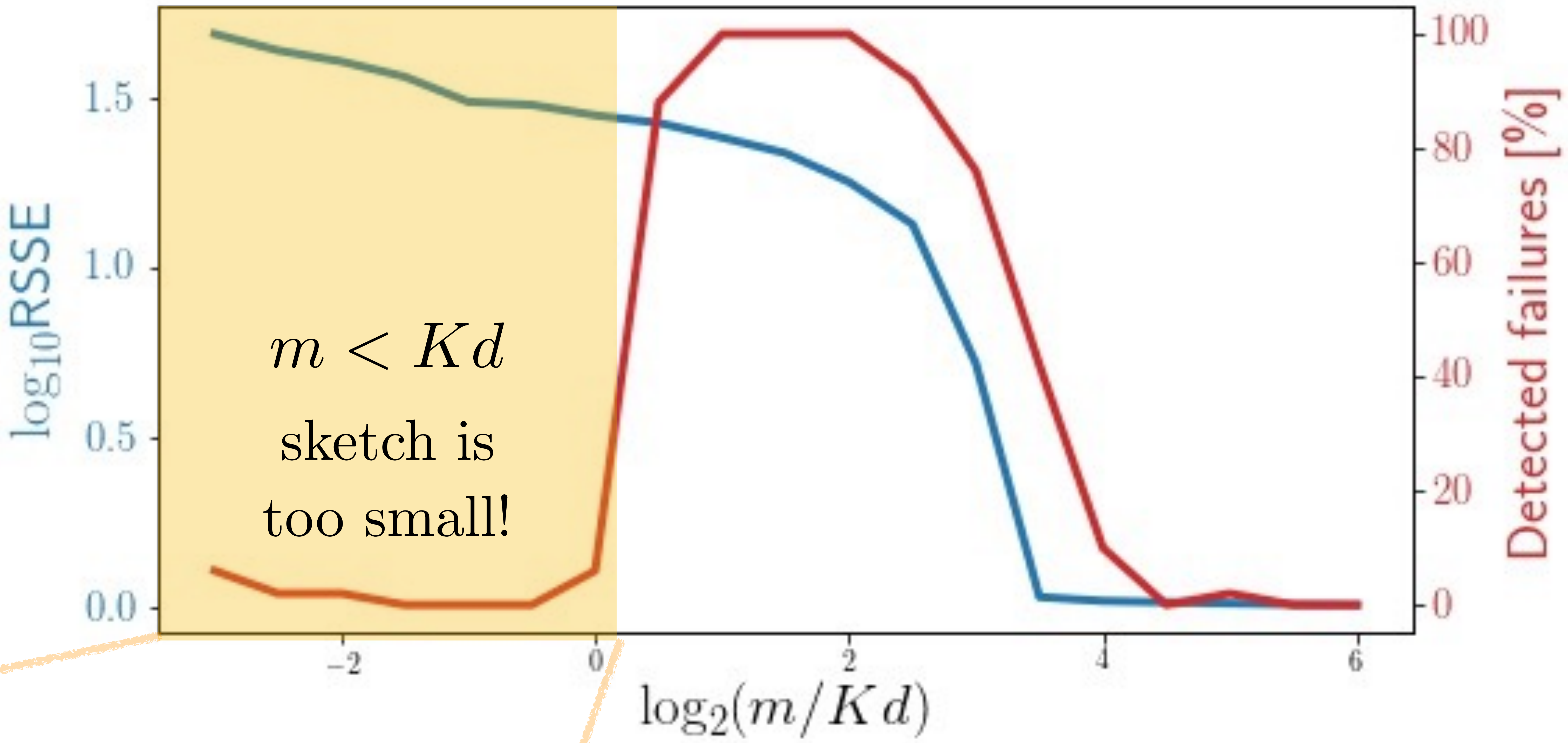


Number of times
$\mathcal{L}(\boldsymbol{\theta}_\Delta; \boldsymbol{z}) > \mathcal{L}(\boldsymbol{\theta}^*; \boldsymbol{z})$

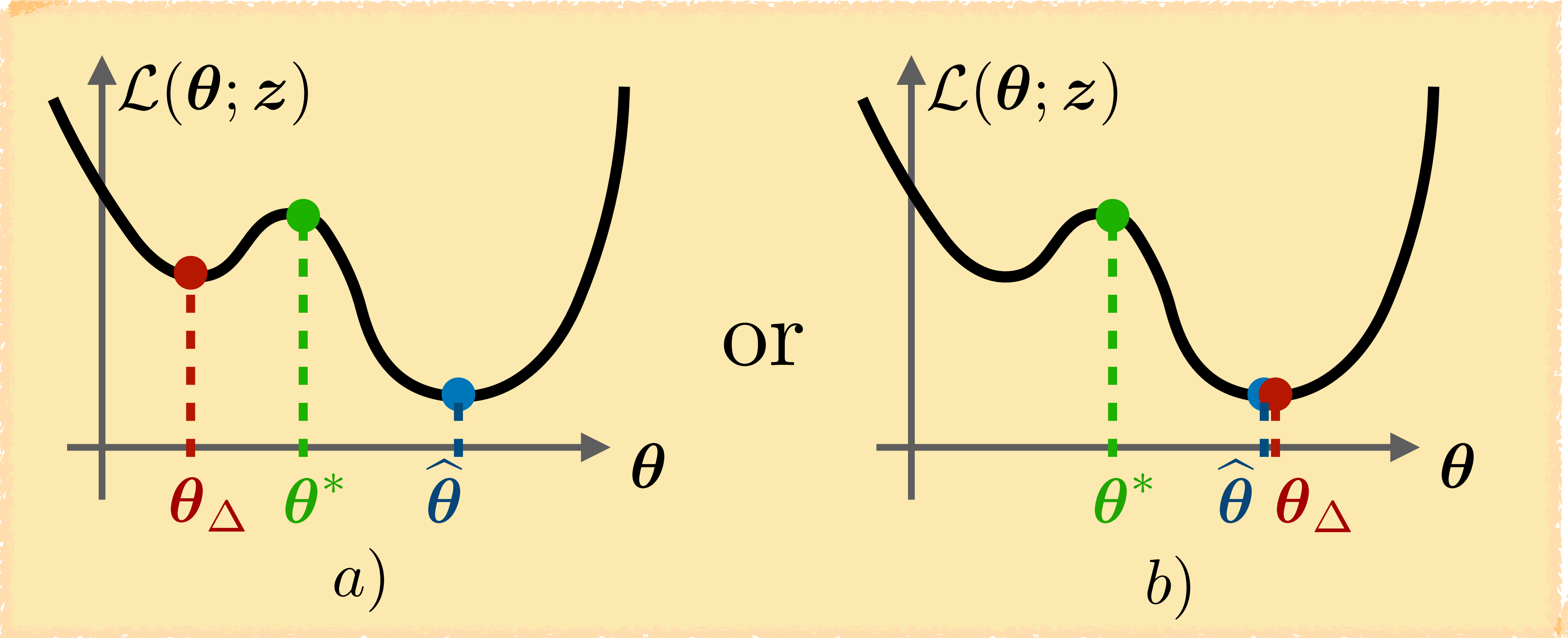
Results: size of m (k-means)

Multiple sketches (50 draws)

Performance
(lower is better)



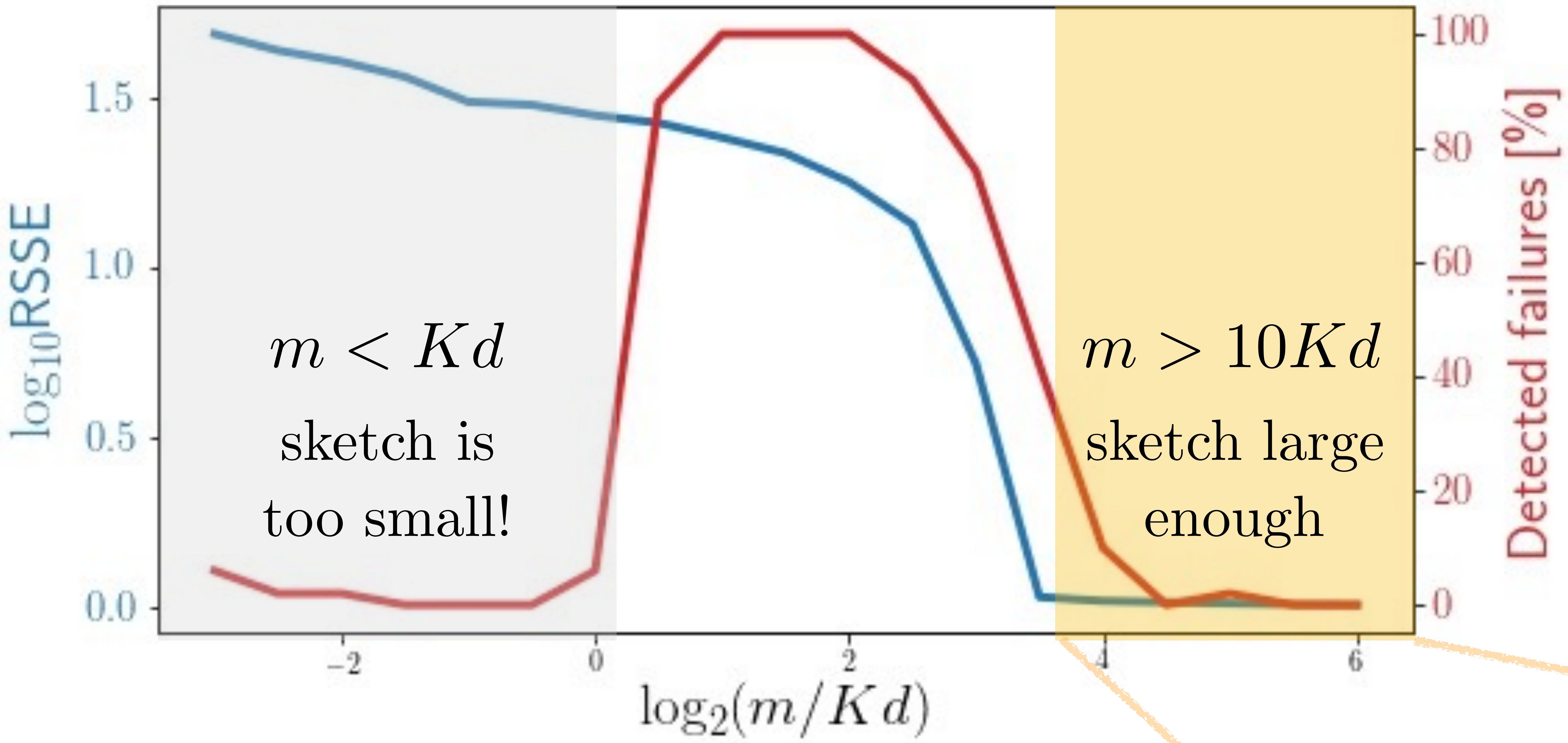
Number of times
$\mathcal{L}(\boldsymbol{\theta}_\Delta; \boldsymbol{z}) > \mathcal{L}(\boldsymbol{\theta}^*; \boldsymbol{z})$



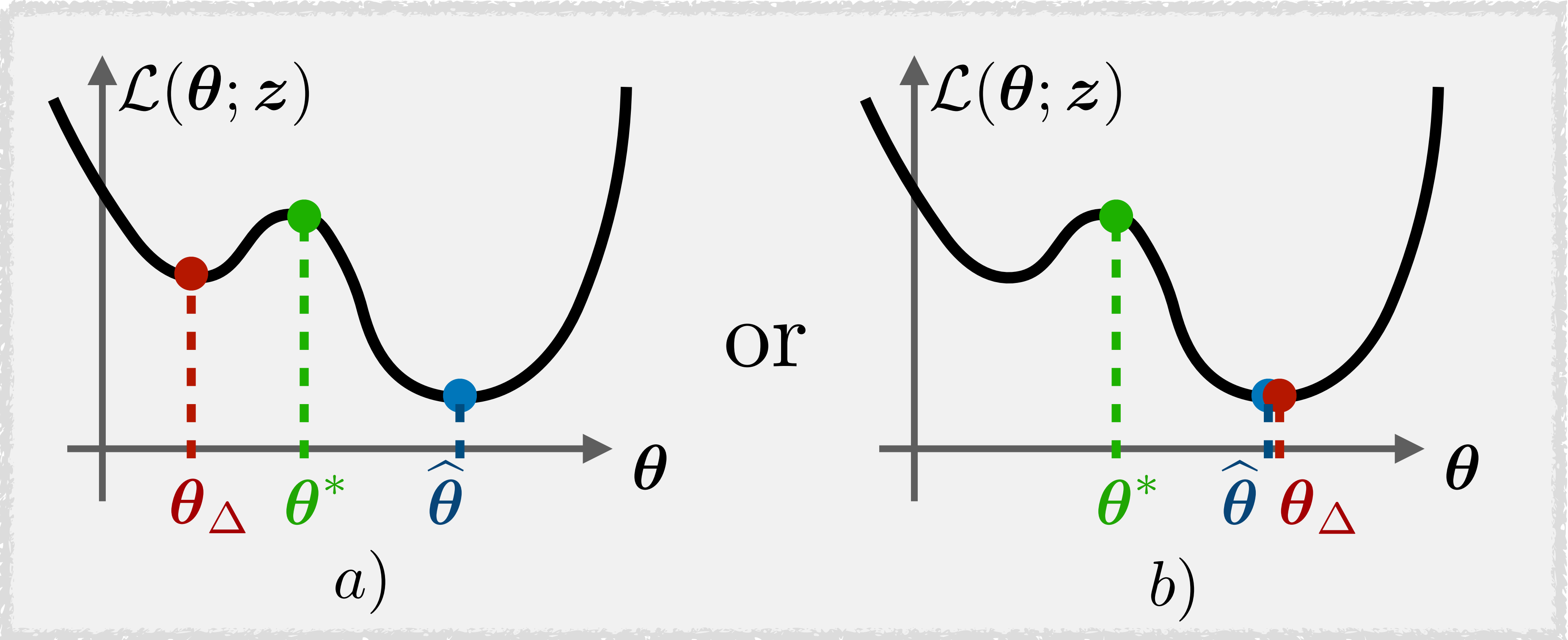
Results: size of m (k-means)

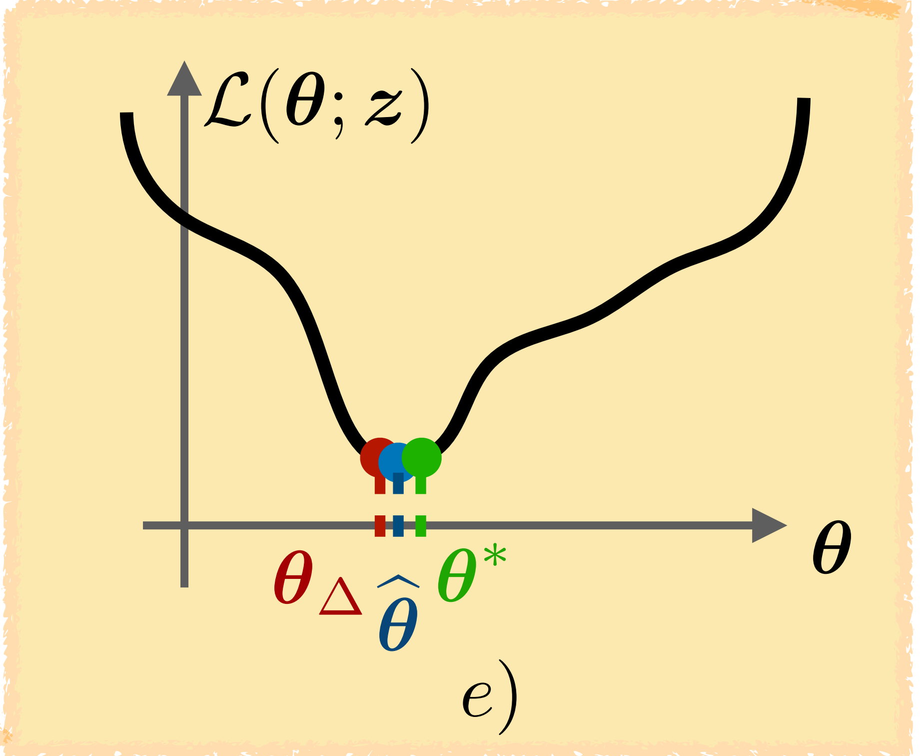
Multiple sketches (50 draws)

Performance (lower is better)



Number of times $\mathcal{L}(\boldsymbol{\theta}_\triangle; \boldsymbol{z}) > \mathcal{L}(\boldsymbol{\theta}^*; \boldsymbol{z})$

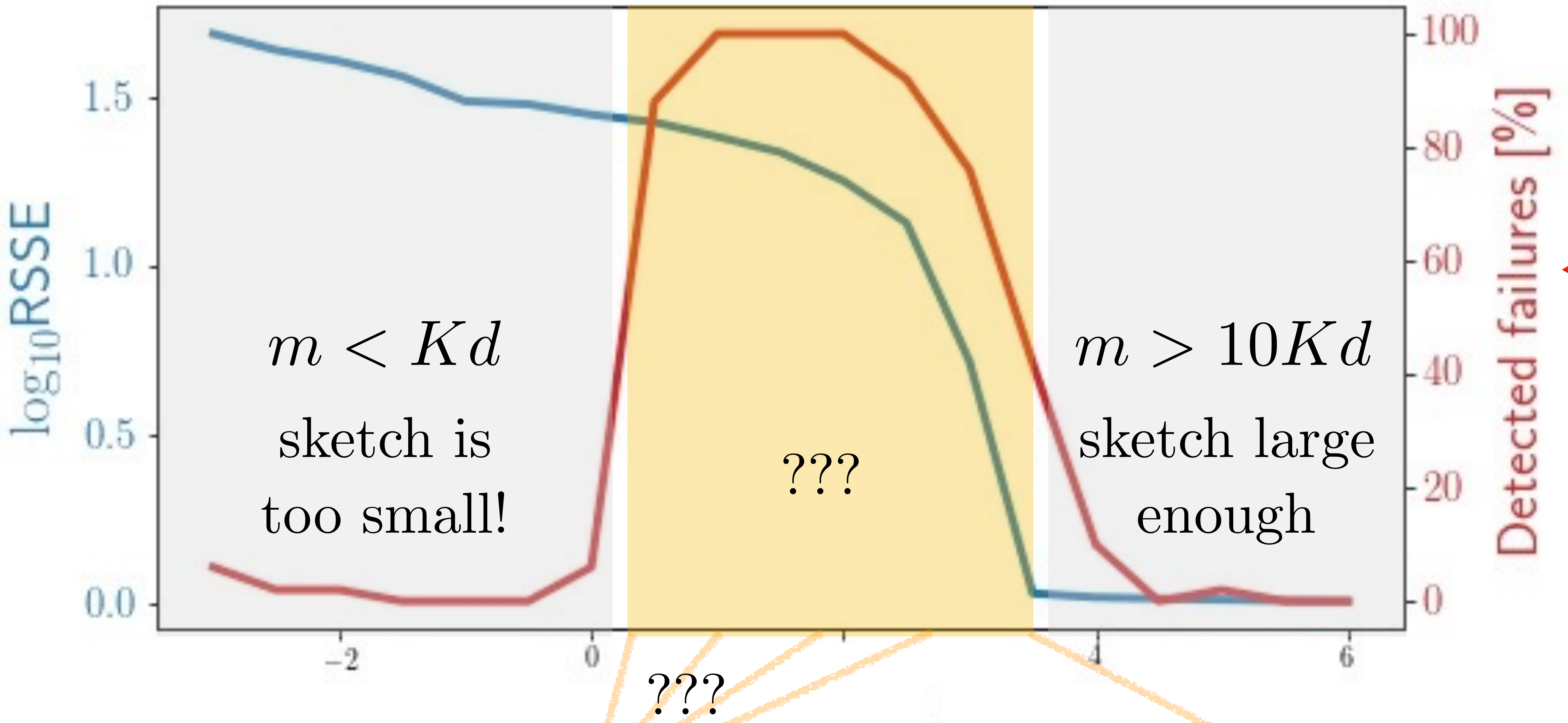




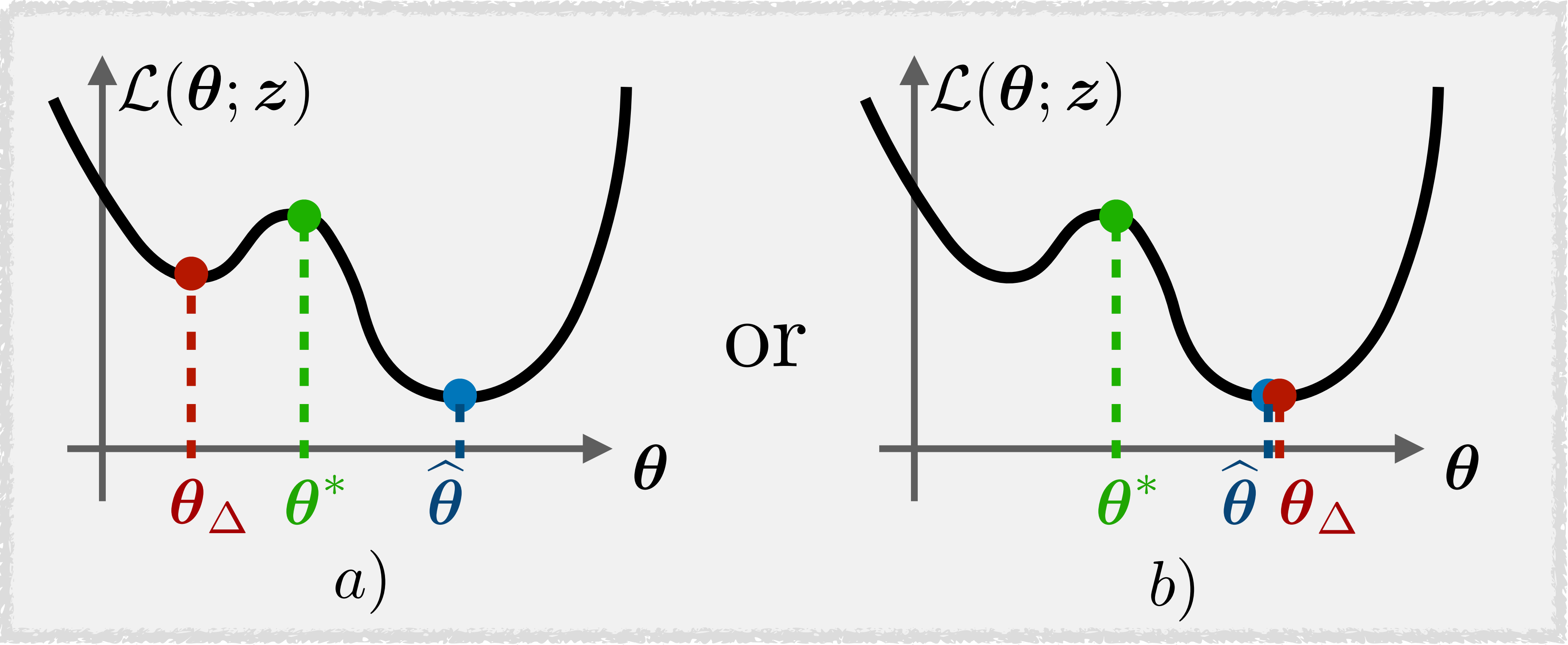
Results: size of m (k-means)

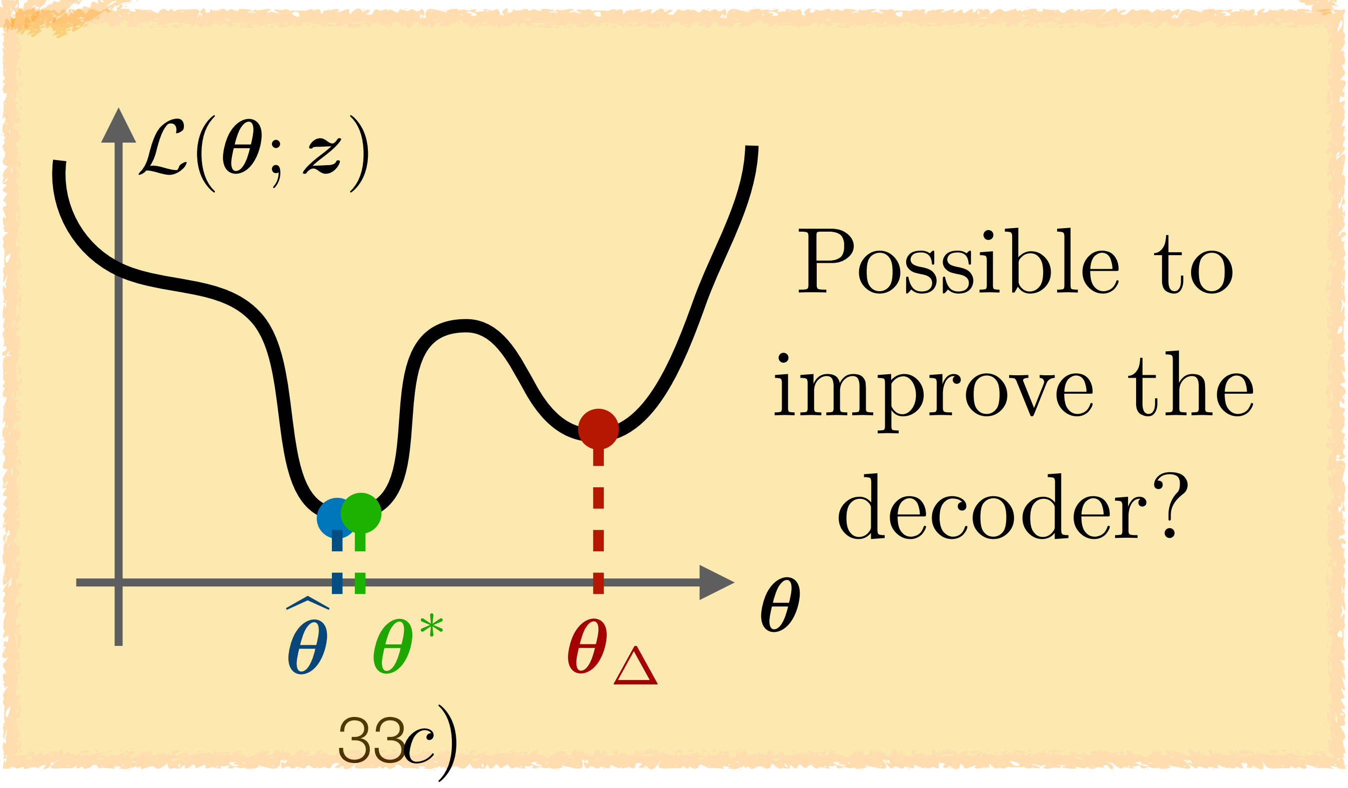
Multiple sketches (50 draws)

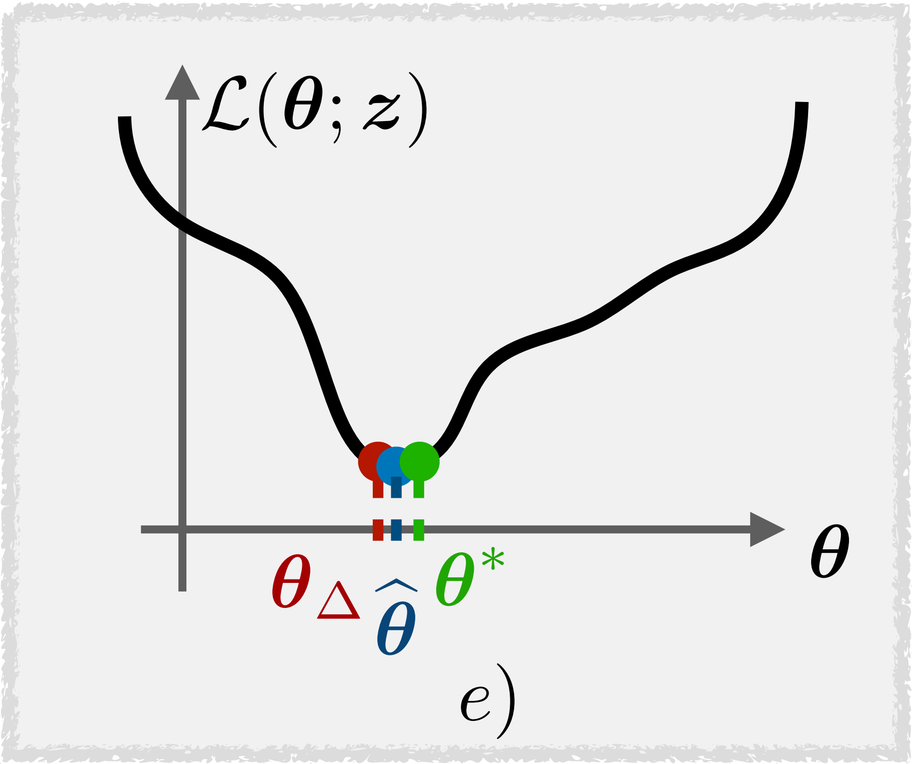
Performance (lower is better)



Number of times $\mathcal{L}(\boldsymbol{\theta}_\triangle; \boldsymbol{z}) > \mathcal{L}(\boldsymbol{\theta}^*; \boldsymbol{z})$







Results: size of m (k-means)

Can we find at least one (not necessarily efficient) decoder that succeeds where CLOMPR doesn't?

Results: size of m (k-means)

Can we find at least one (not necessarily efficient) decoder
that succeeds where CLOMPR doesn't?

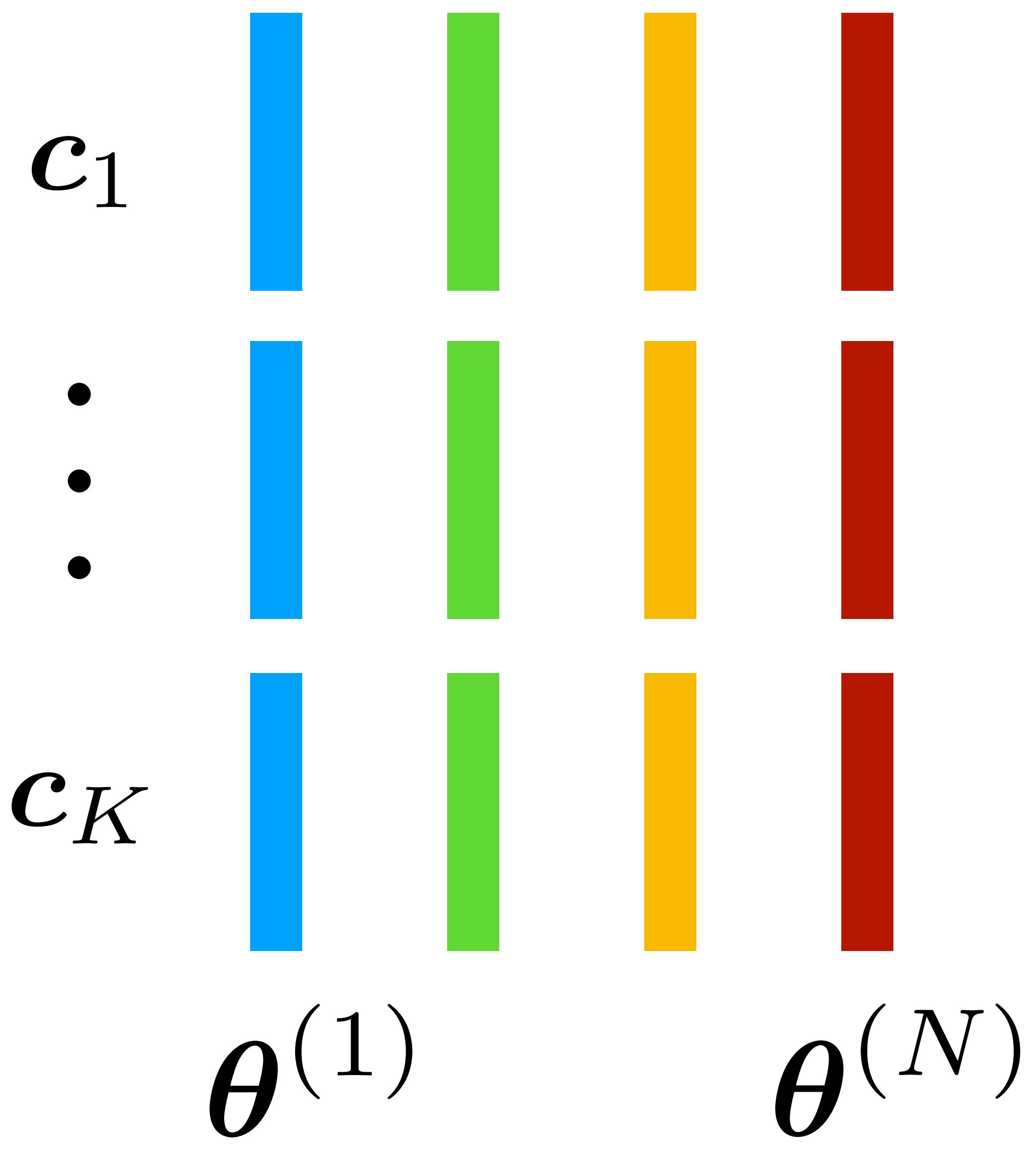
Goal: something that is *an epsilon* smarter than brute-force

For research purposes only

Results: size of m (k-means)

Can we find at least one (not necessarily efficient) decoder that succeeds where CLOMPR doesn't?

Let's explore the solution space (slightly better than) randomly: a genetic algorithm

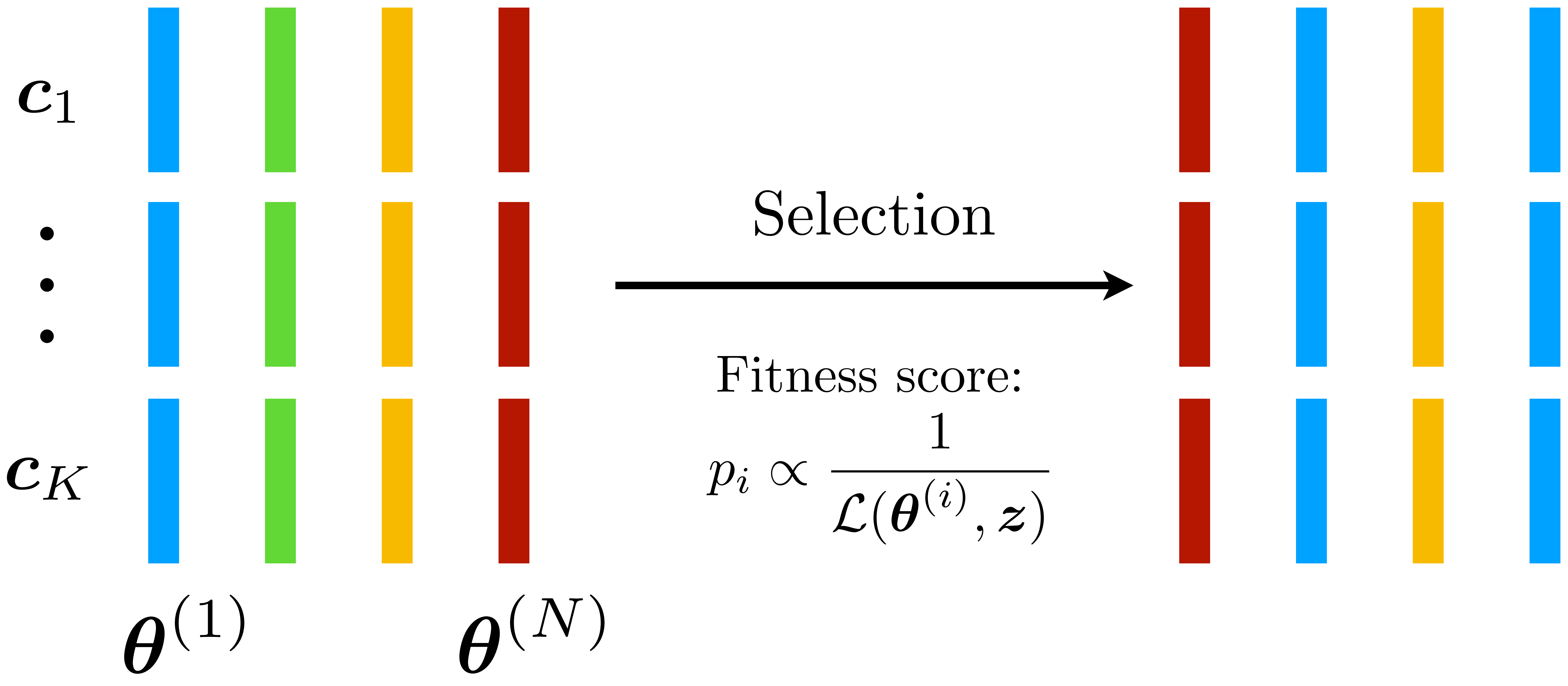


Population of (sets of) centroids

Results: size of m (k-means)

Can we find at least one (not necessarily efficient) decoder that succeeds where CLOMPR doesn't?

Let's explore the solution space (slightly better than) randomly: a genetic algorithm

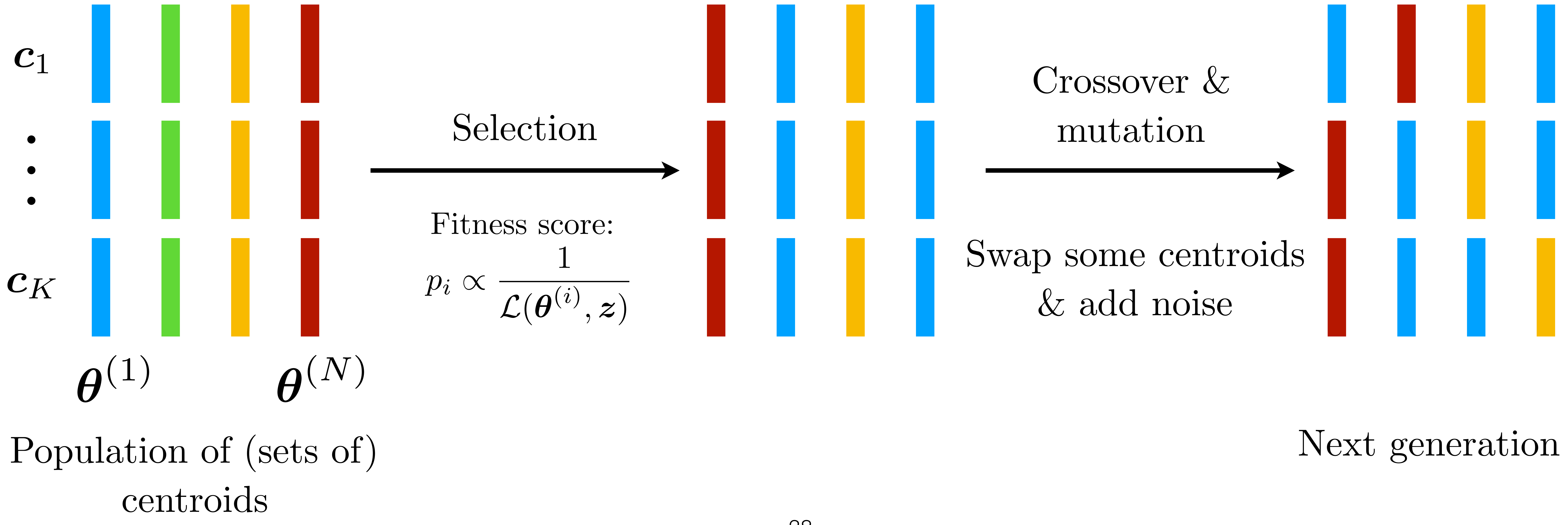


Population of (sets of) centroids

Results: size of m (k-means)

Can we find at least one (not necessarily efficient) decoder that succeeds where CLOMPR doesn't?

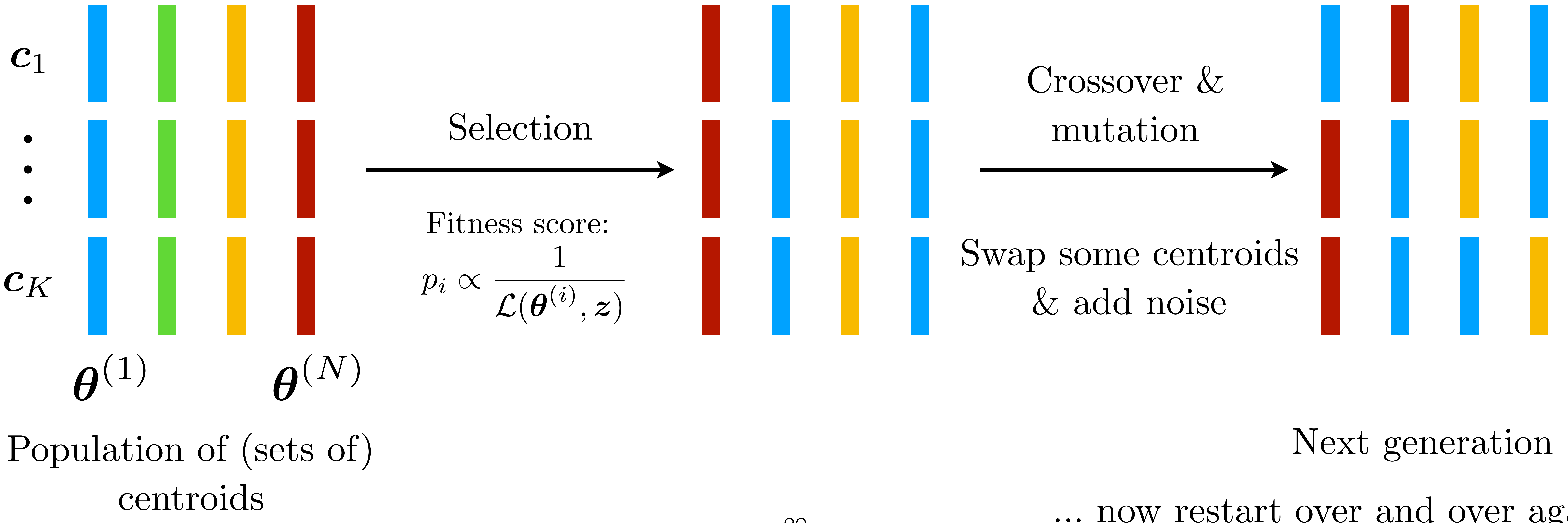
Let's explore the solution space (slightly better than) randomly: a genetic algorithm



Results: size of m (k-means)

Can we find at least one (not necessarily efficient) decoder that succeeds where CLOMPR doesn't?

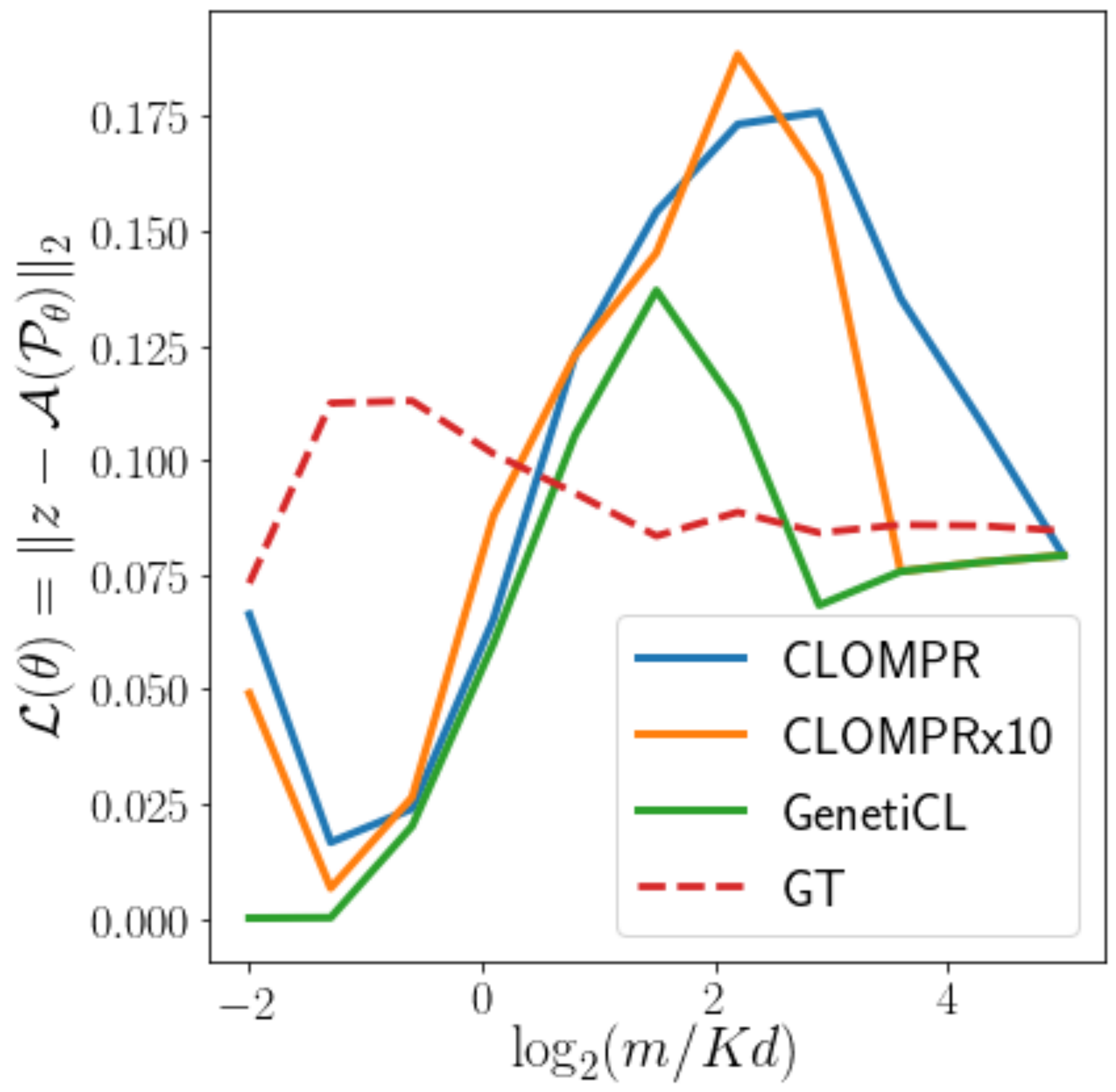
Let's explore the solution space (slightly better than) randomly: a genetic algorithm

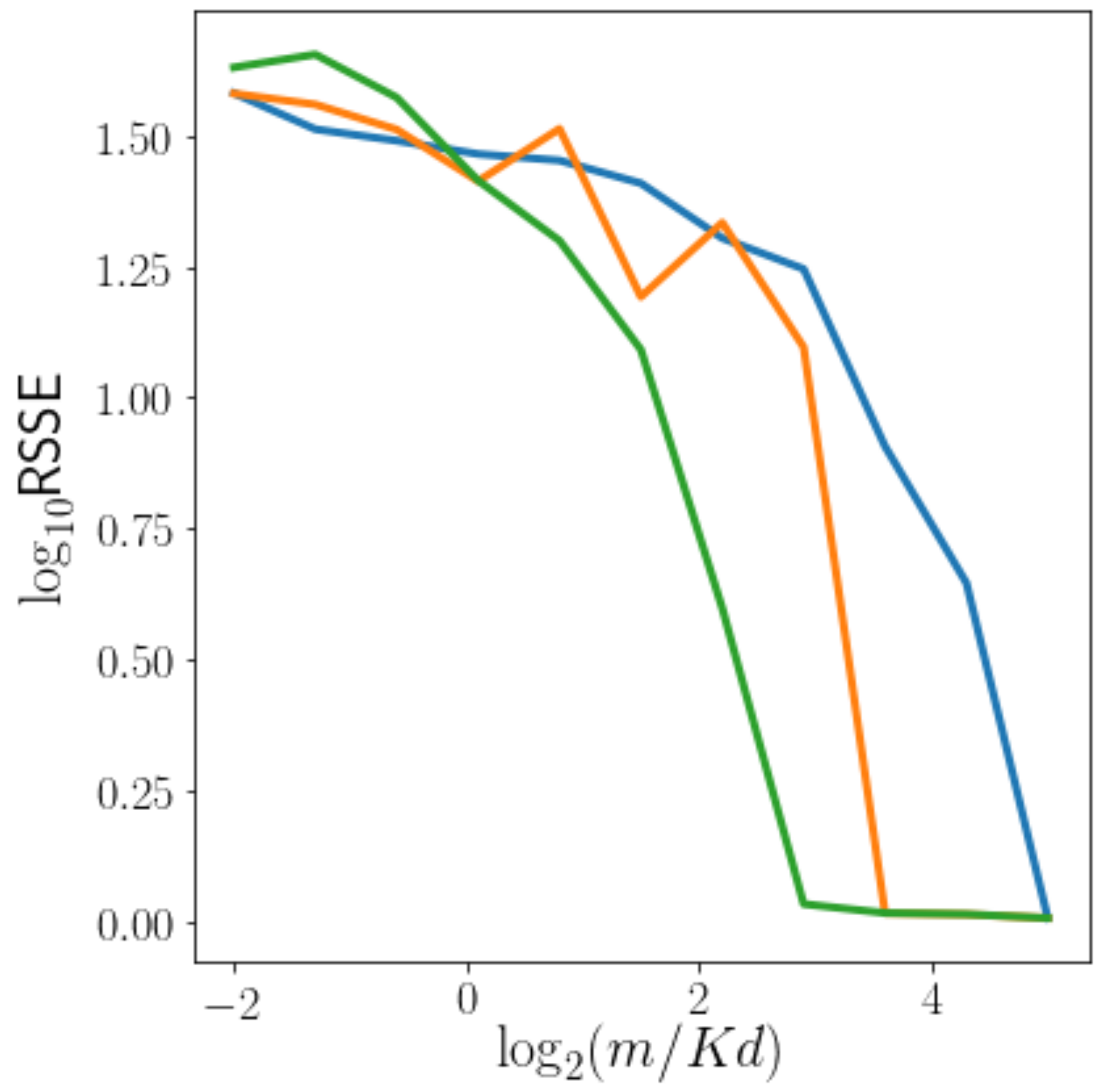


… now restart over and over again!

Results: size of m (k-means)

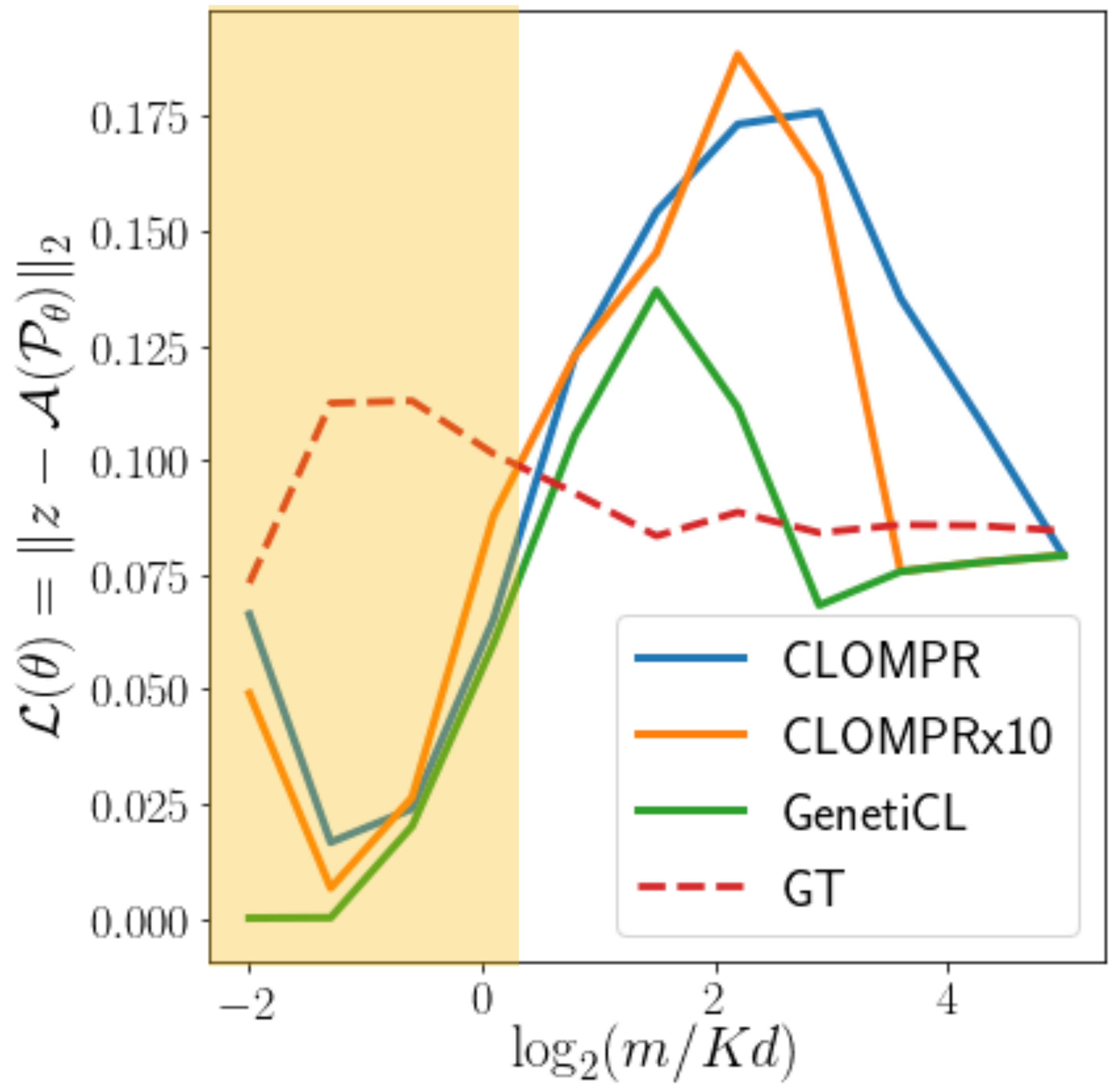
For a fixed sketch

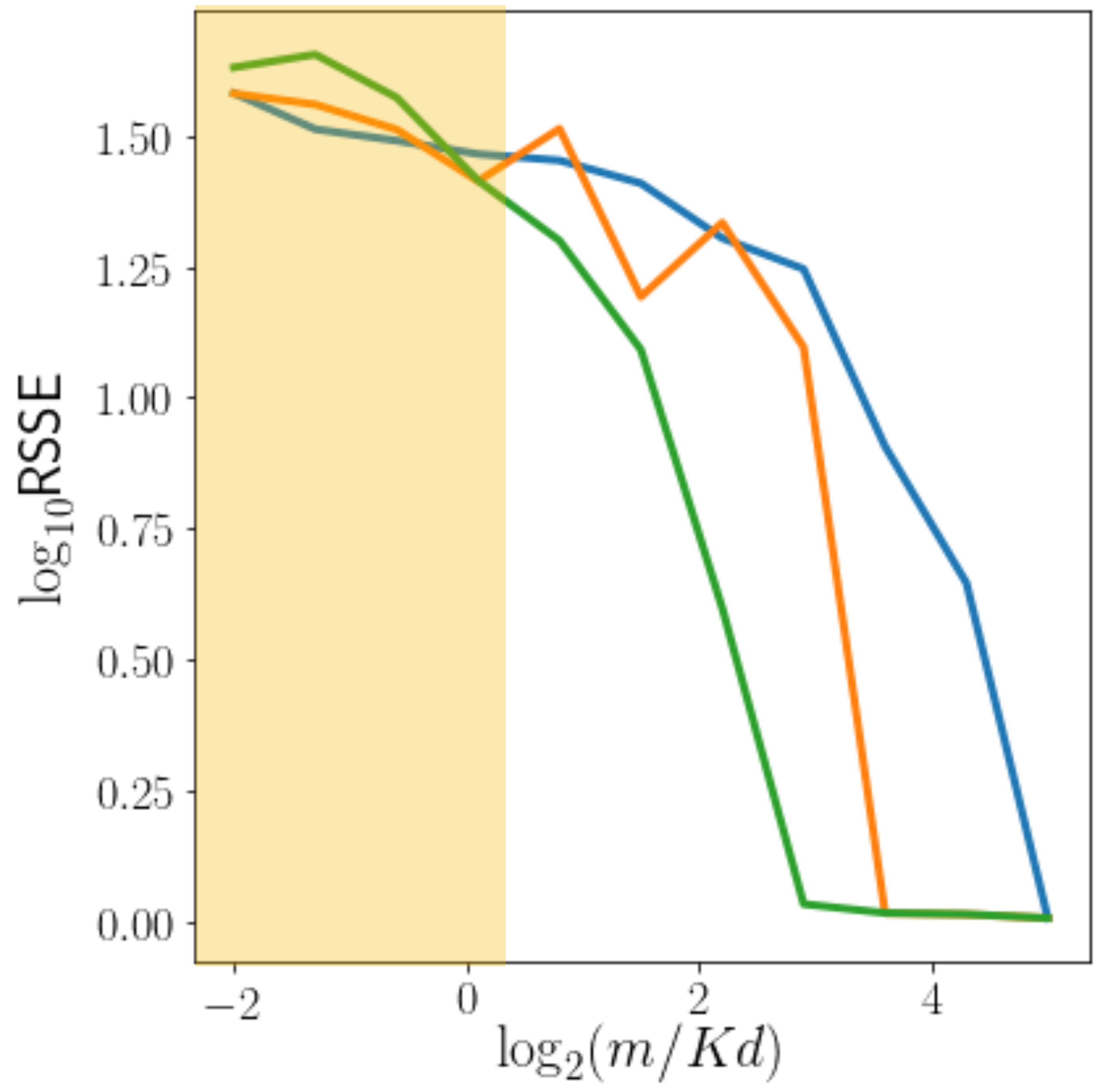




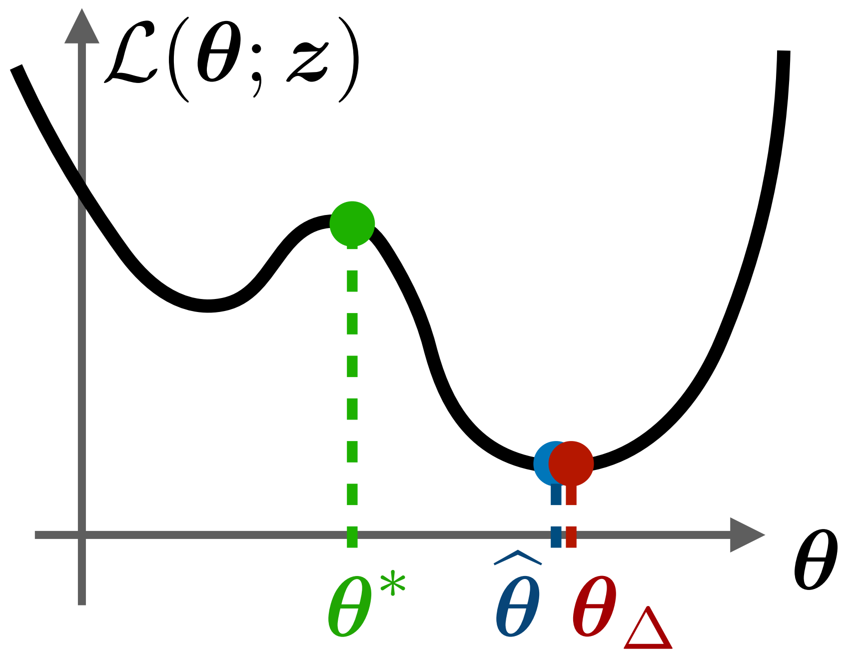
Results: size of m (k-means)

For a fixed sketch





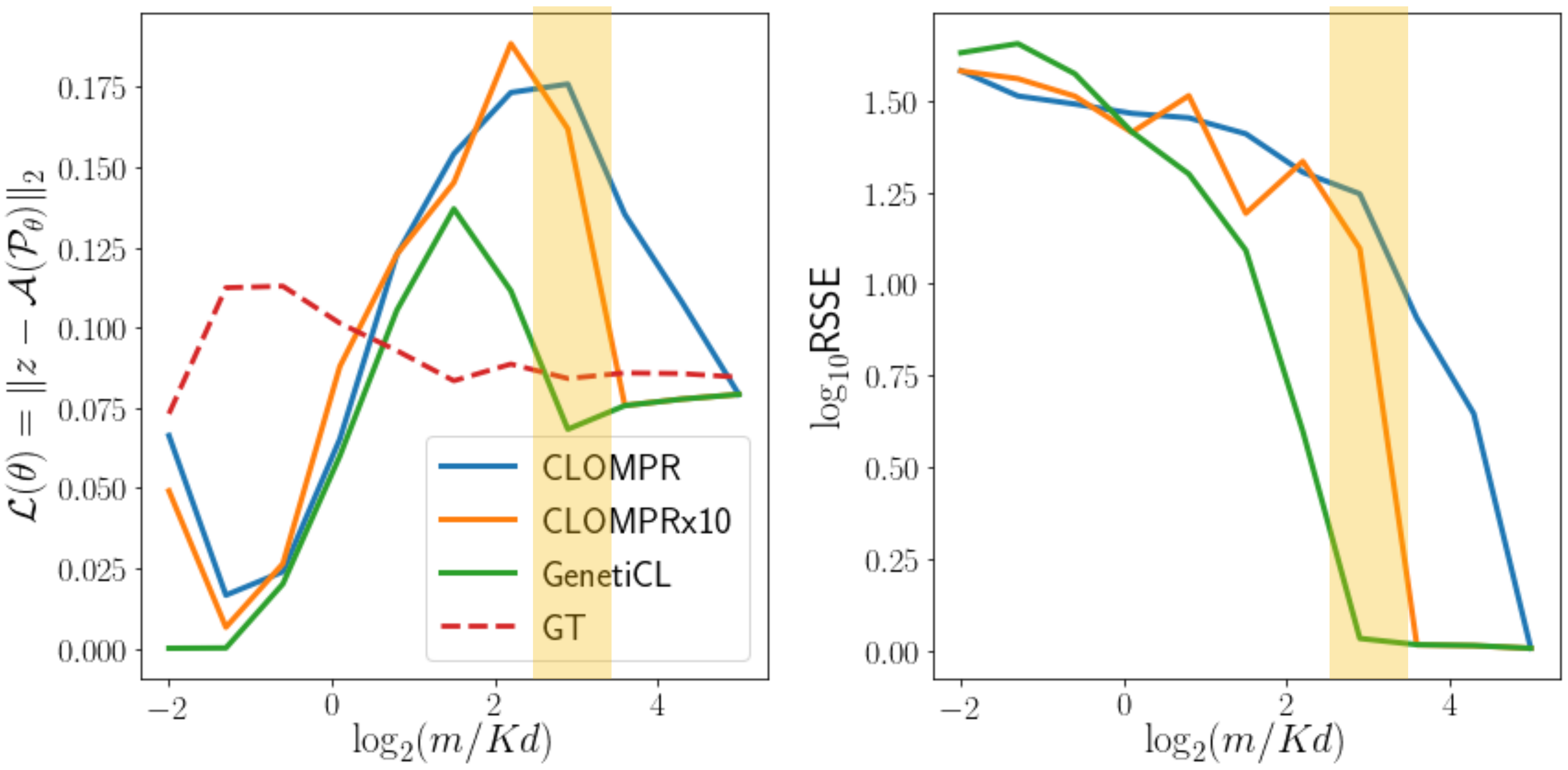
As before, loss is ill-defined when there are less measurements than parameters



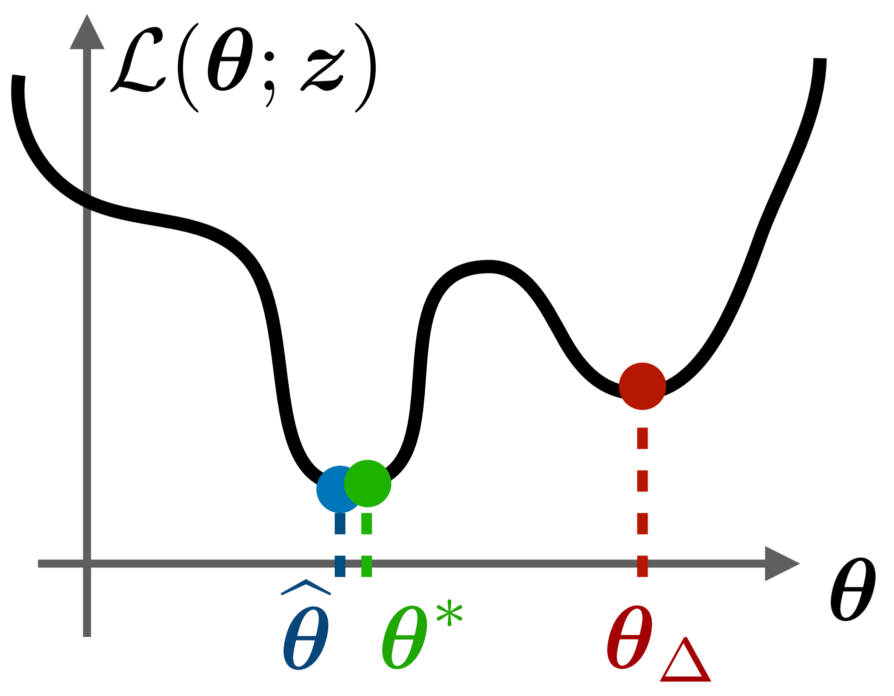
$b)$

Results: size of m (k-means)

For a fixed sketch



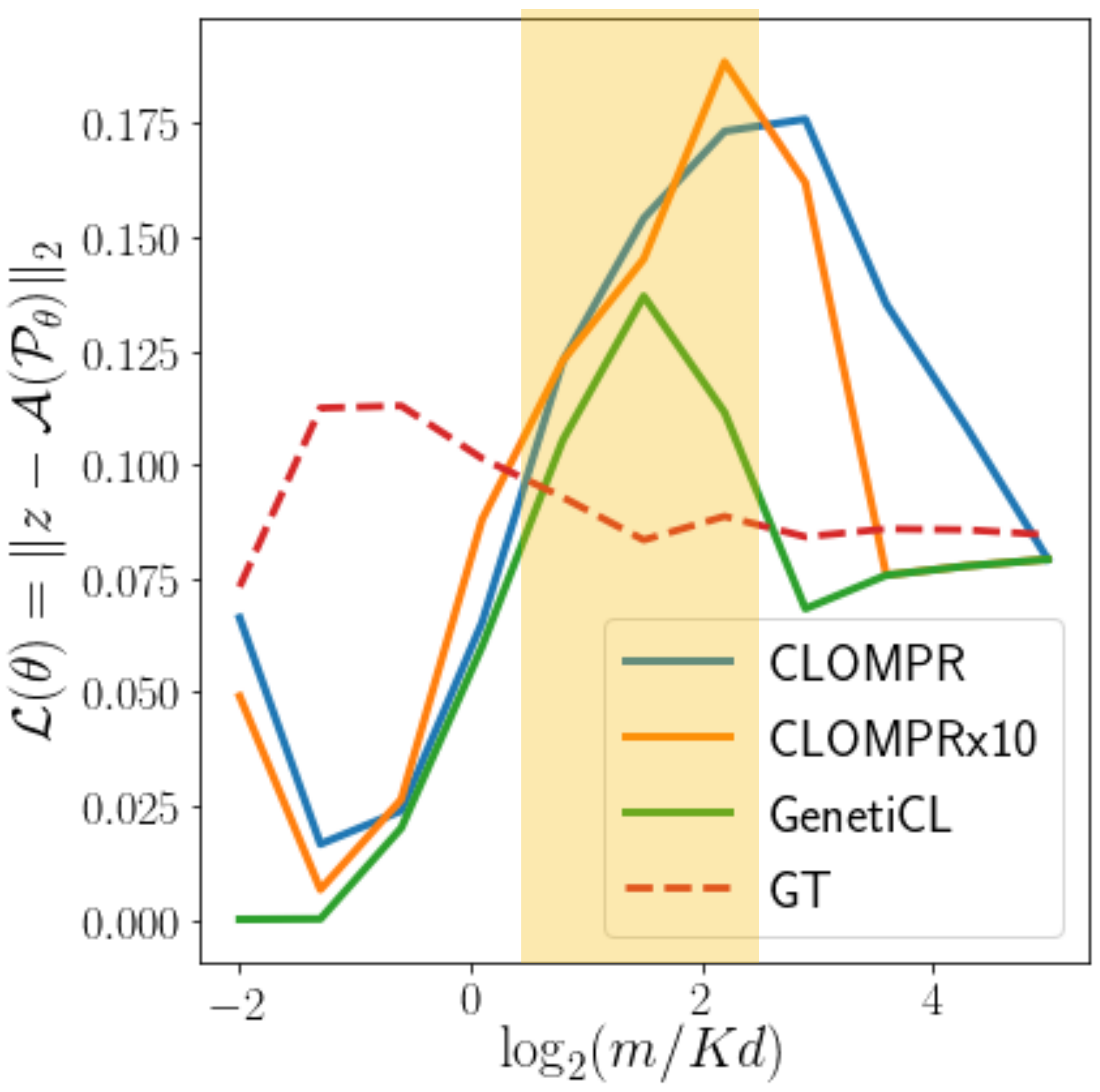
For m not too small,
GenetiCL finds a better optimum!
CLOMPR could be improved!

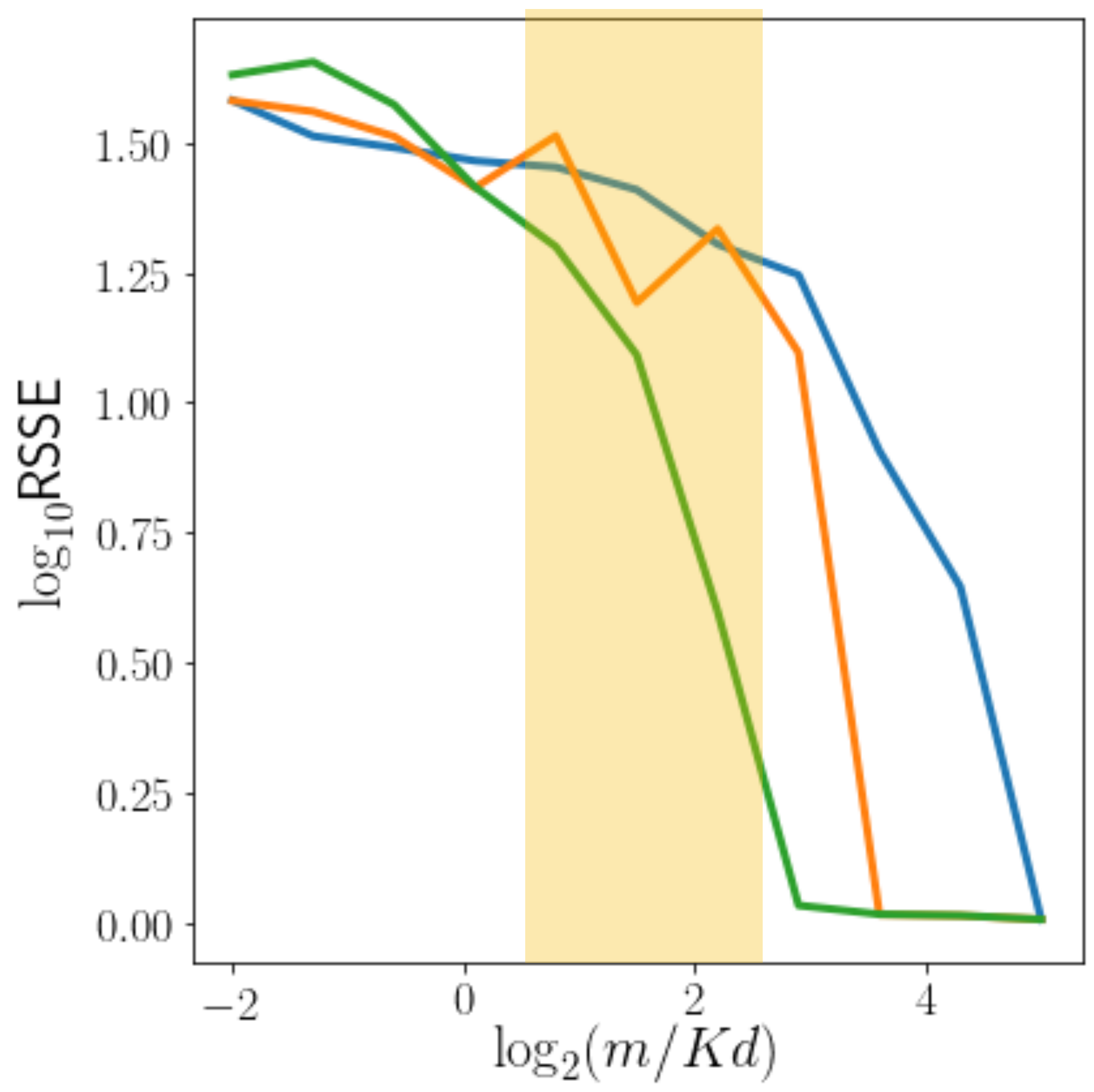


$c)$

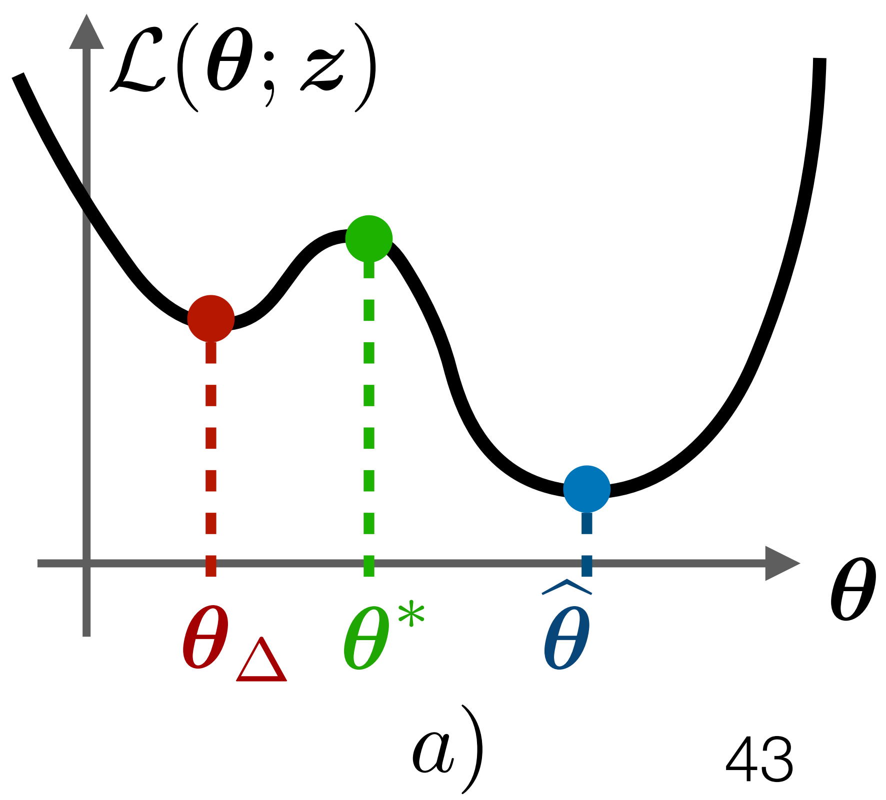
Results: size of m (k-means)

For a fixed sketch

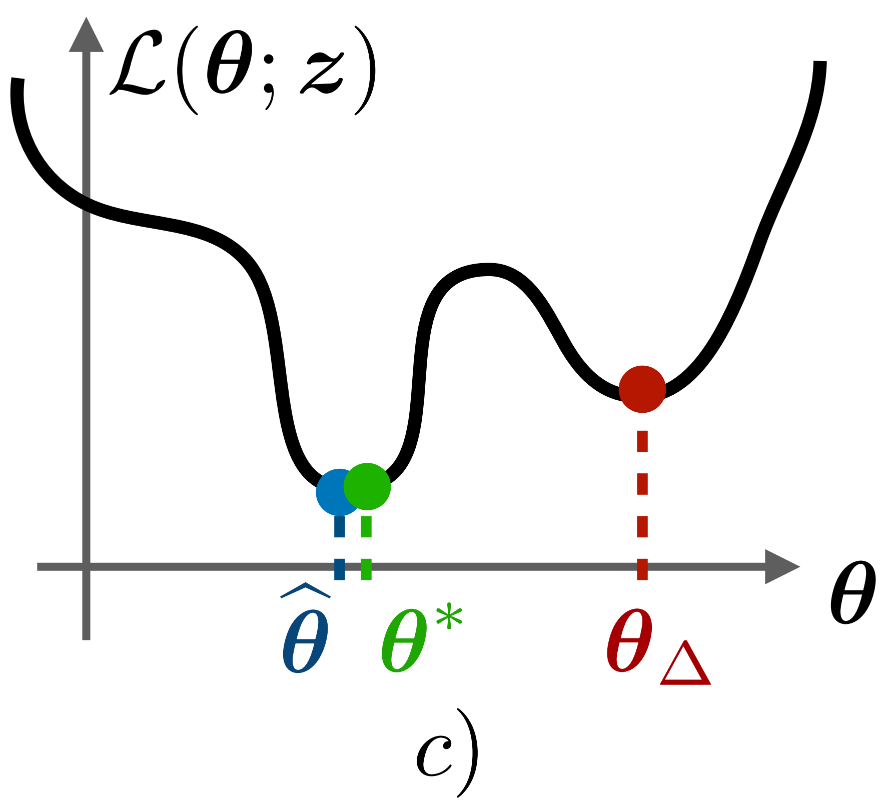




An ambiguous region remains...

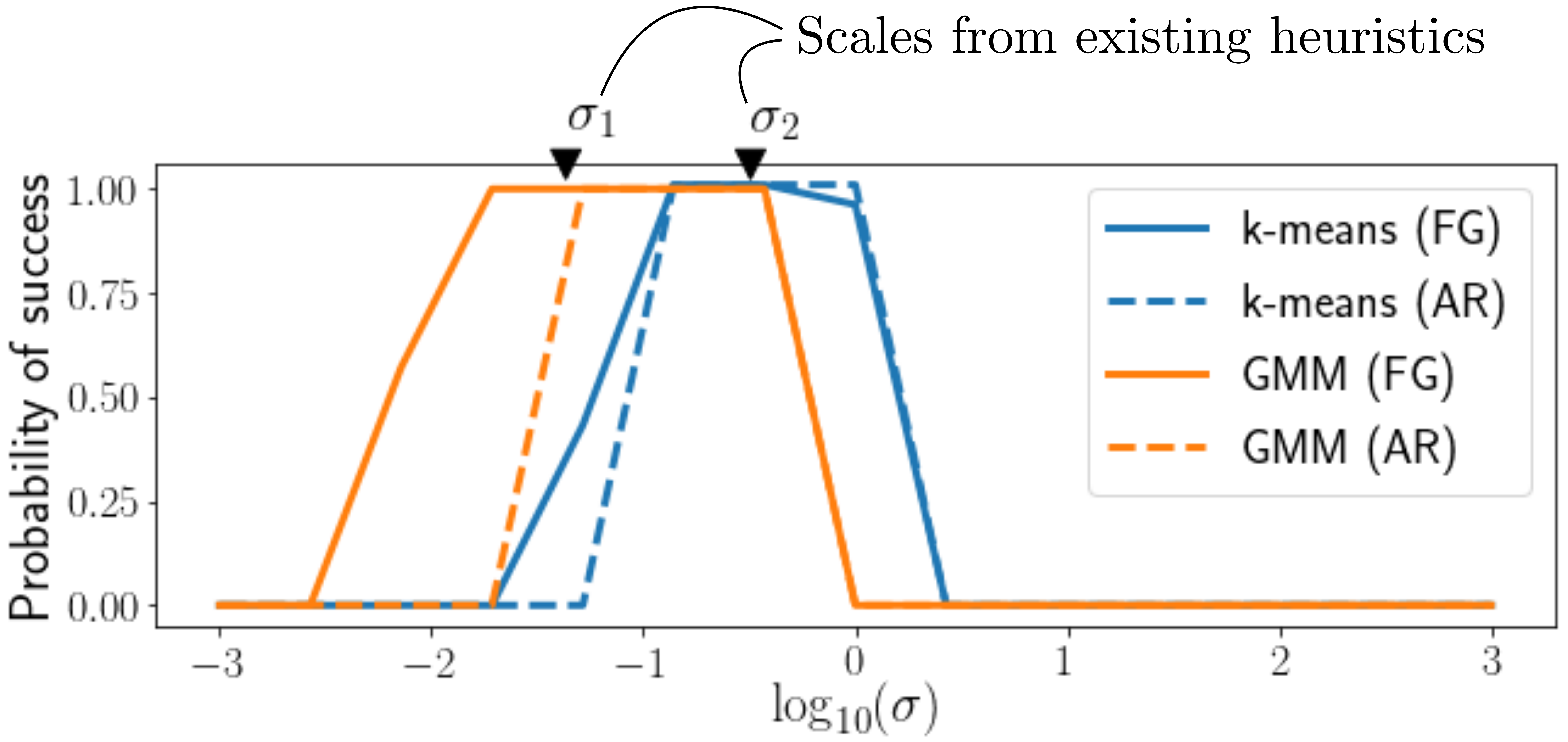


or

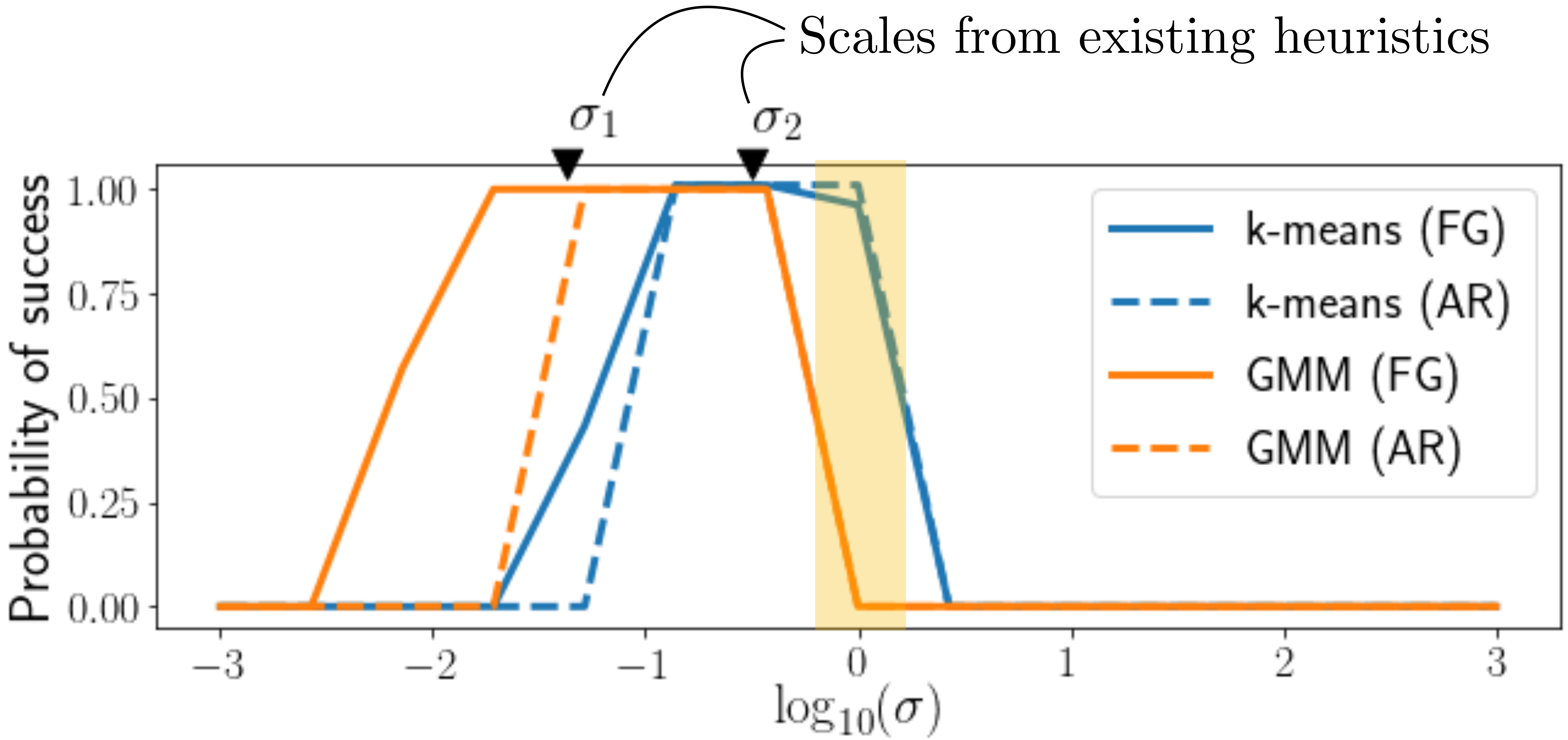


??

Results: sketch scale parameter (for different tasks)

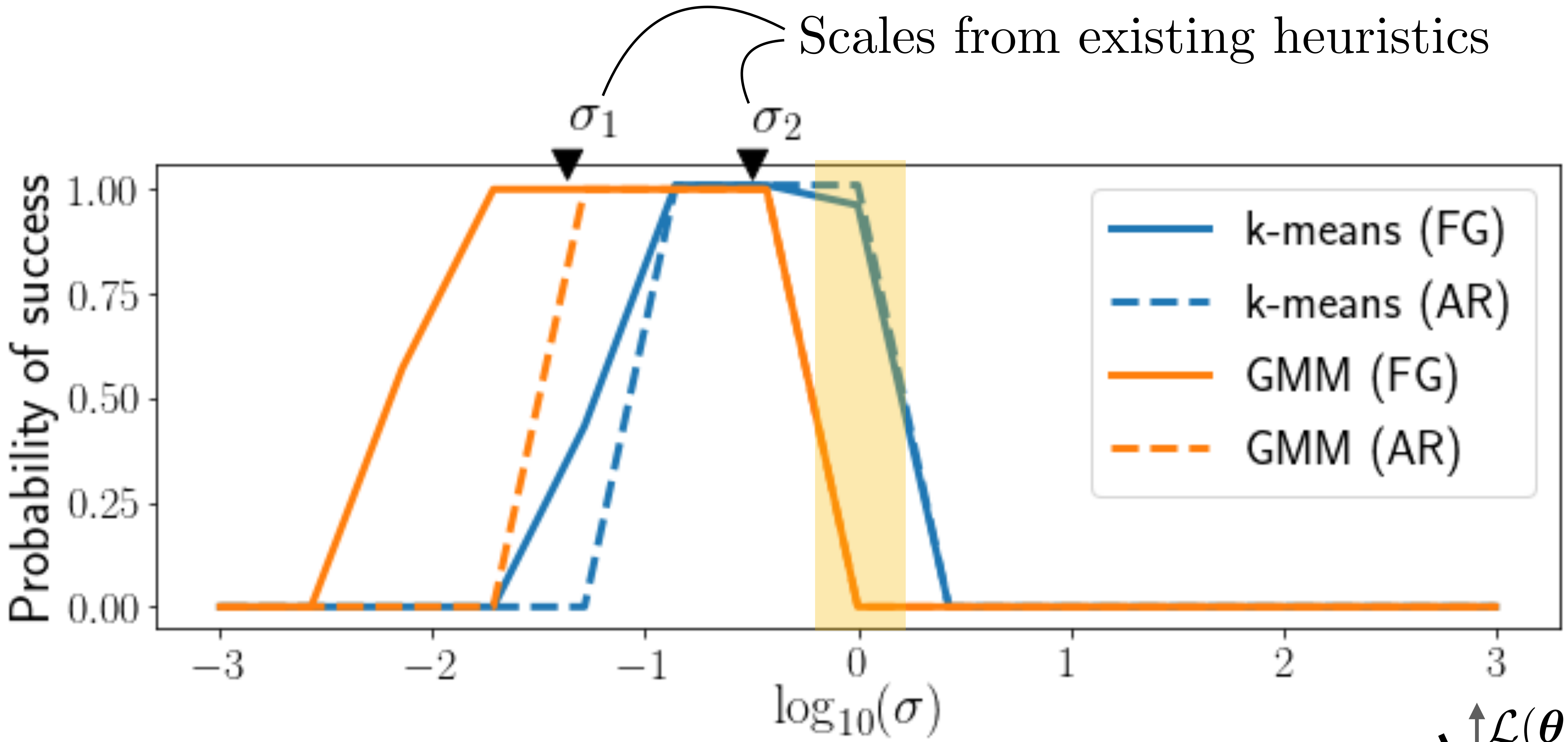


Results: sketch scale parameter (for different tasks)



<u>At large scales, k-means succeeds but not GMM</u>

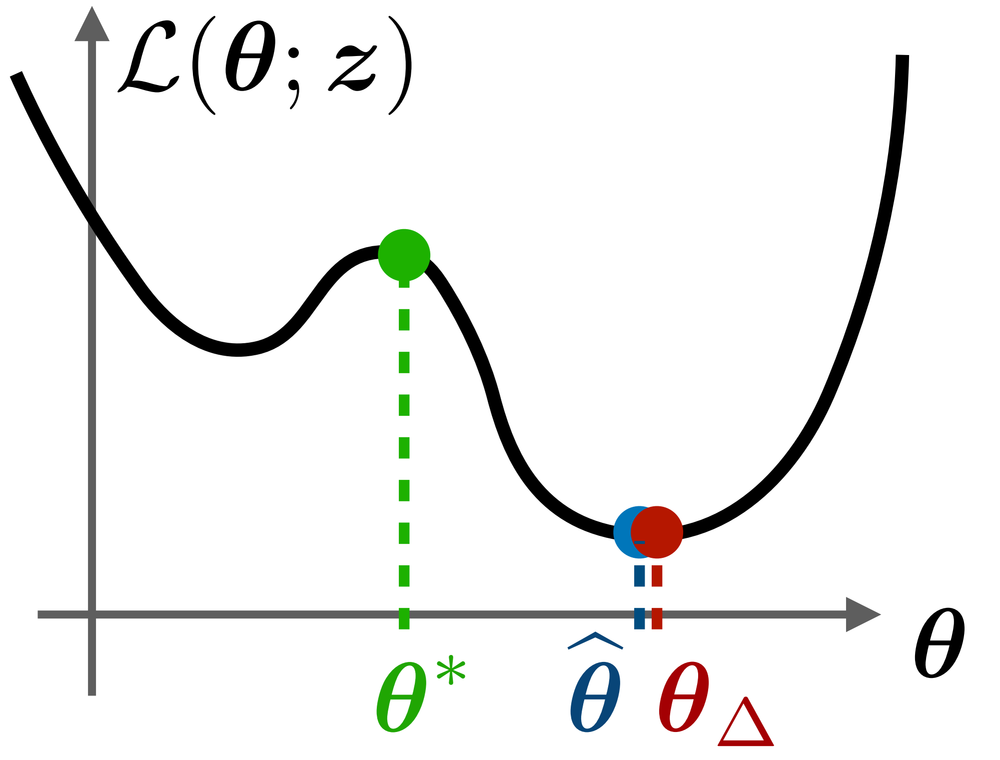
Results: sketch scale parameter (for different tasks)



<u>At large scales, k-means succeeds but not GMM</u>

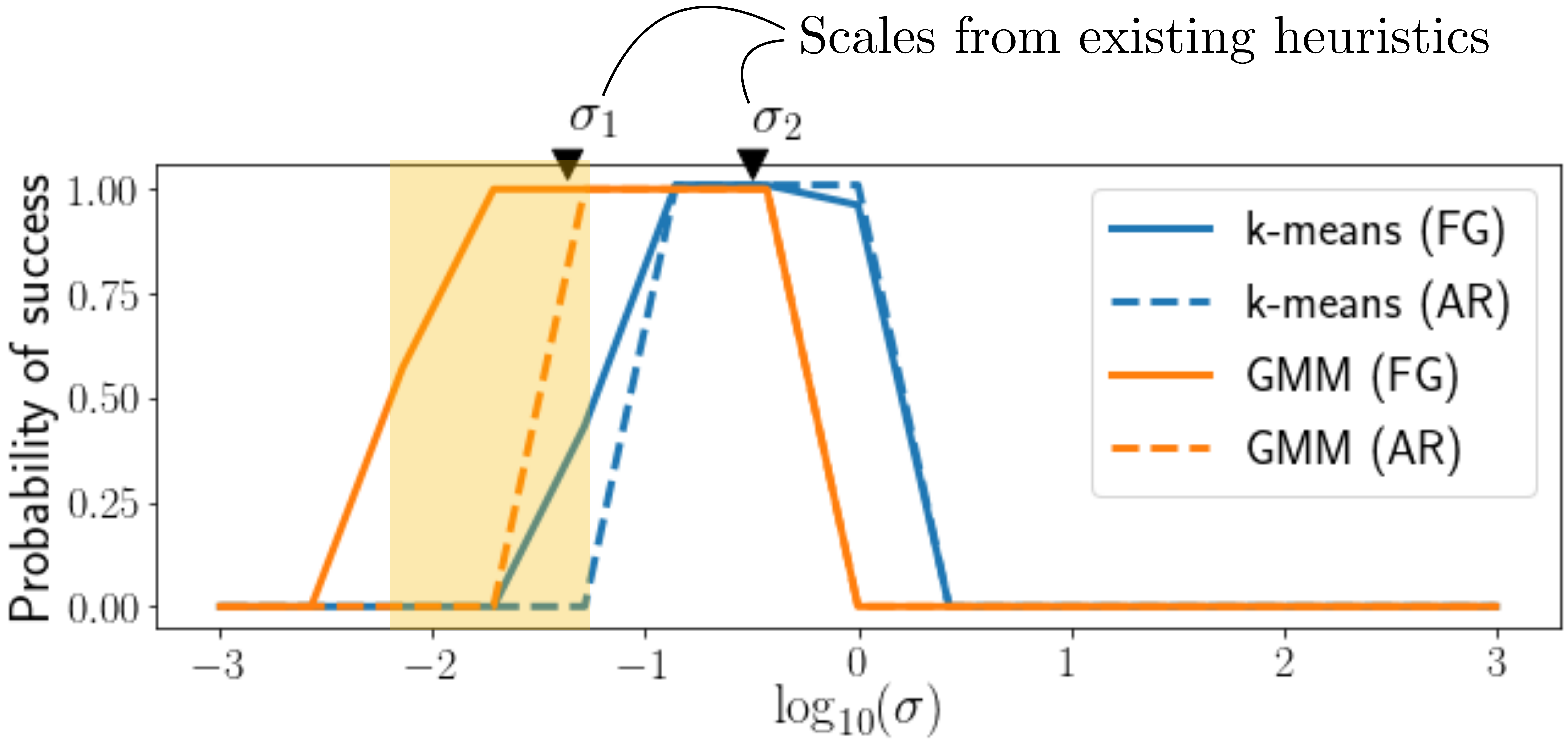
(sketch failures)

Sketch captures a rough (low-pass) approx. to the data distribution



$b)$

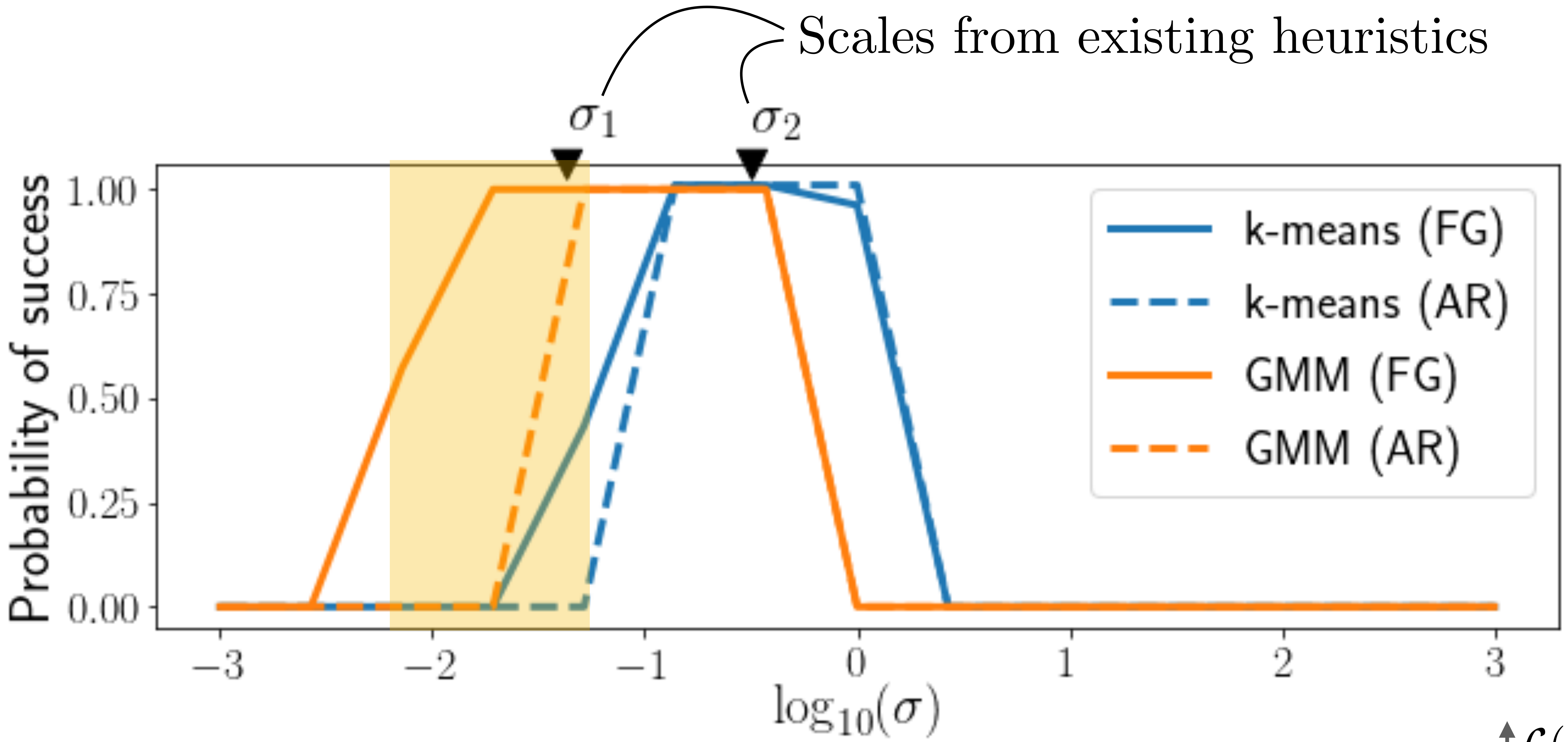
Results: sketch scale parameter (for different tasks)



At large scales, k-means succeeds but not GMM

<u>At small scales, GMM succeeds but not k-means</u> (decoder failures)

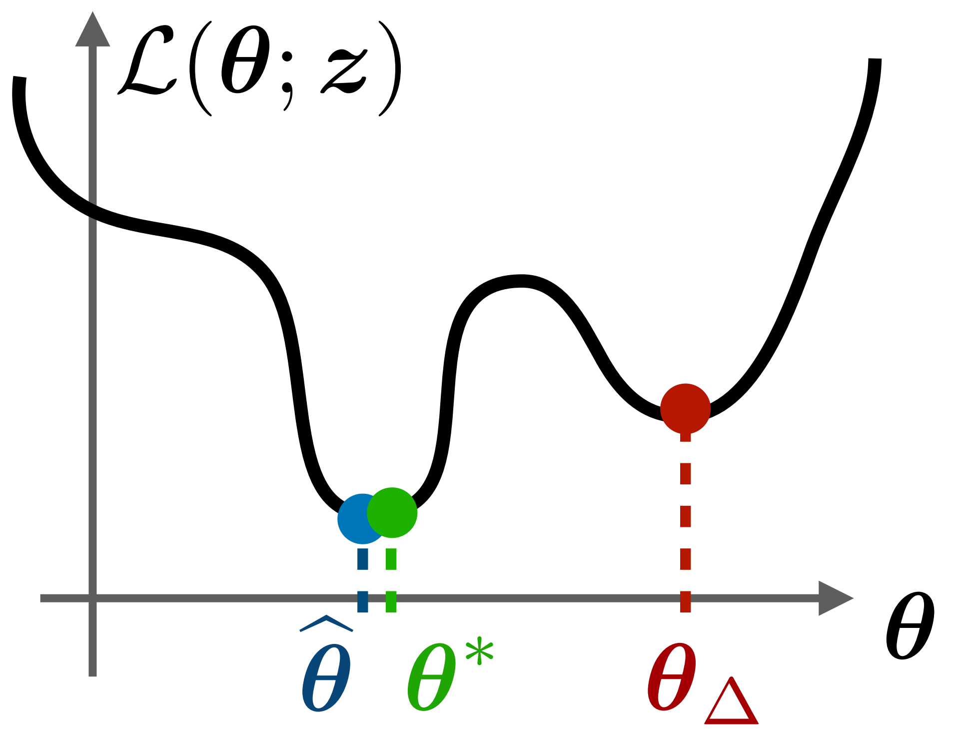
Results: sketch scale parameter (for different tasks)



At large scales, k-means succeeds but not GMM

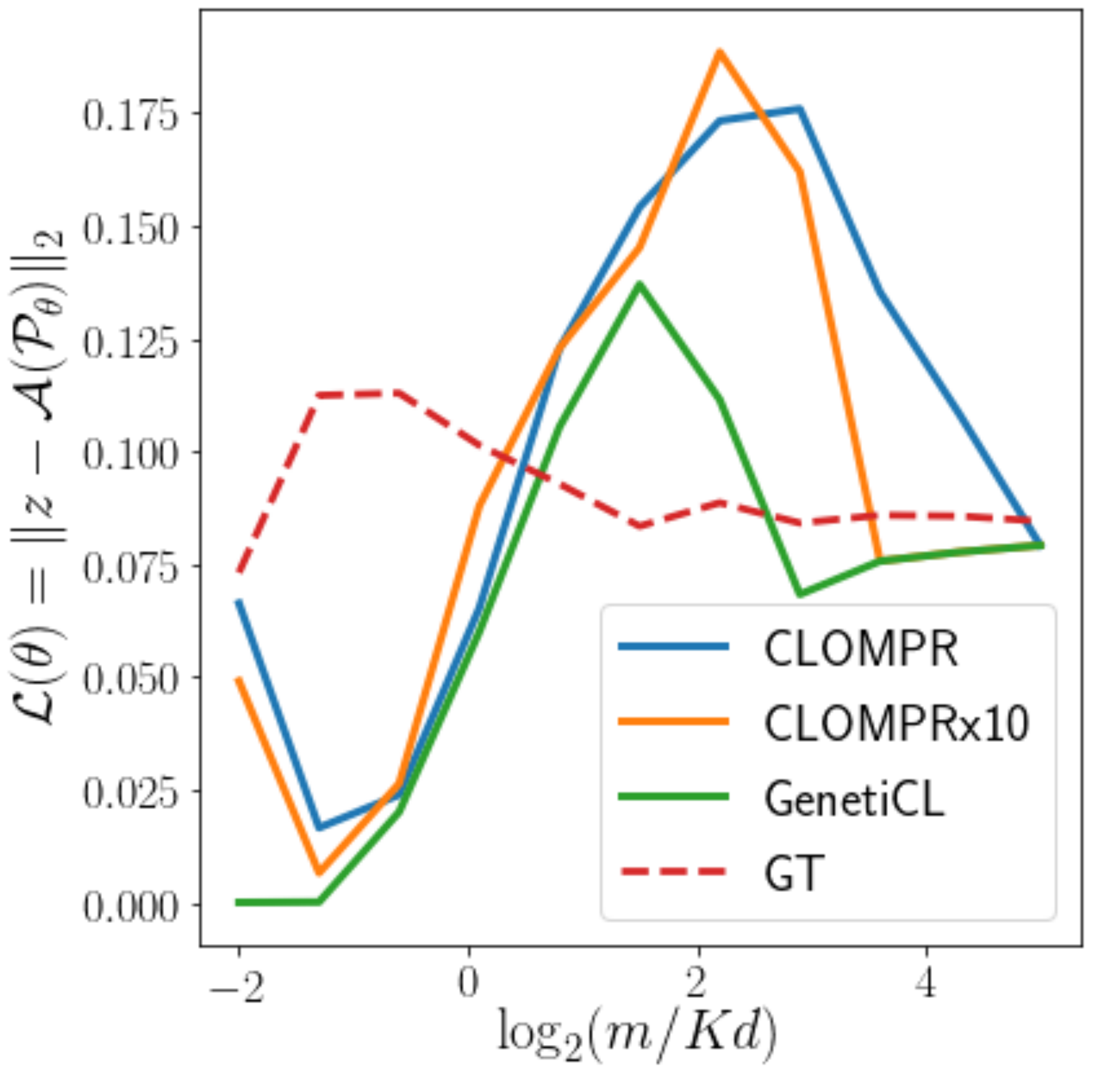
<u>At small scales, GMM succeeds but not k-means</u>

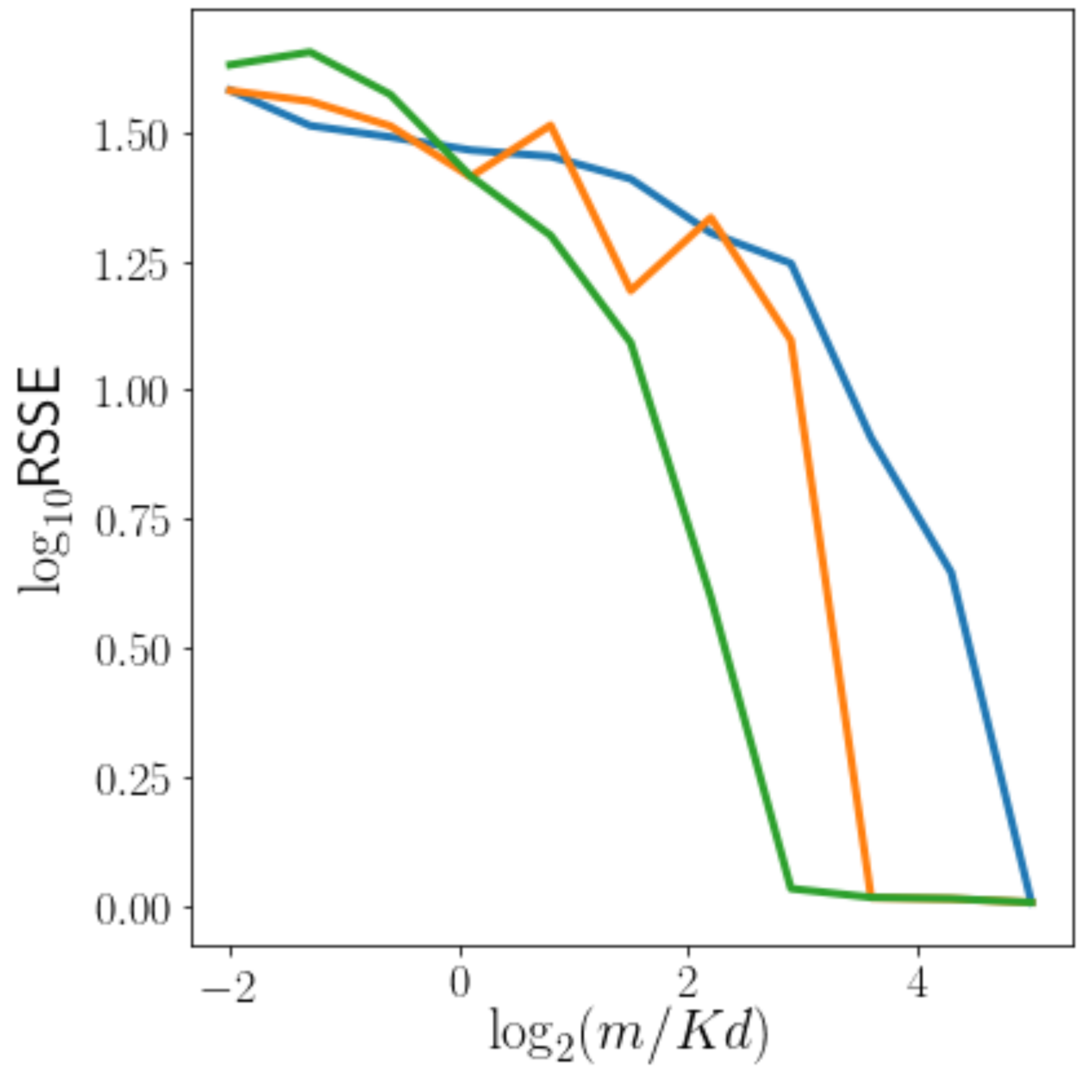
More surprising: adding variables makes the decoder behave better!

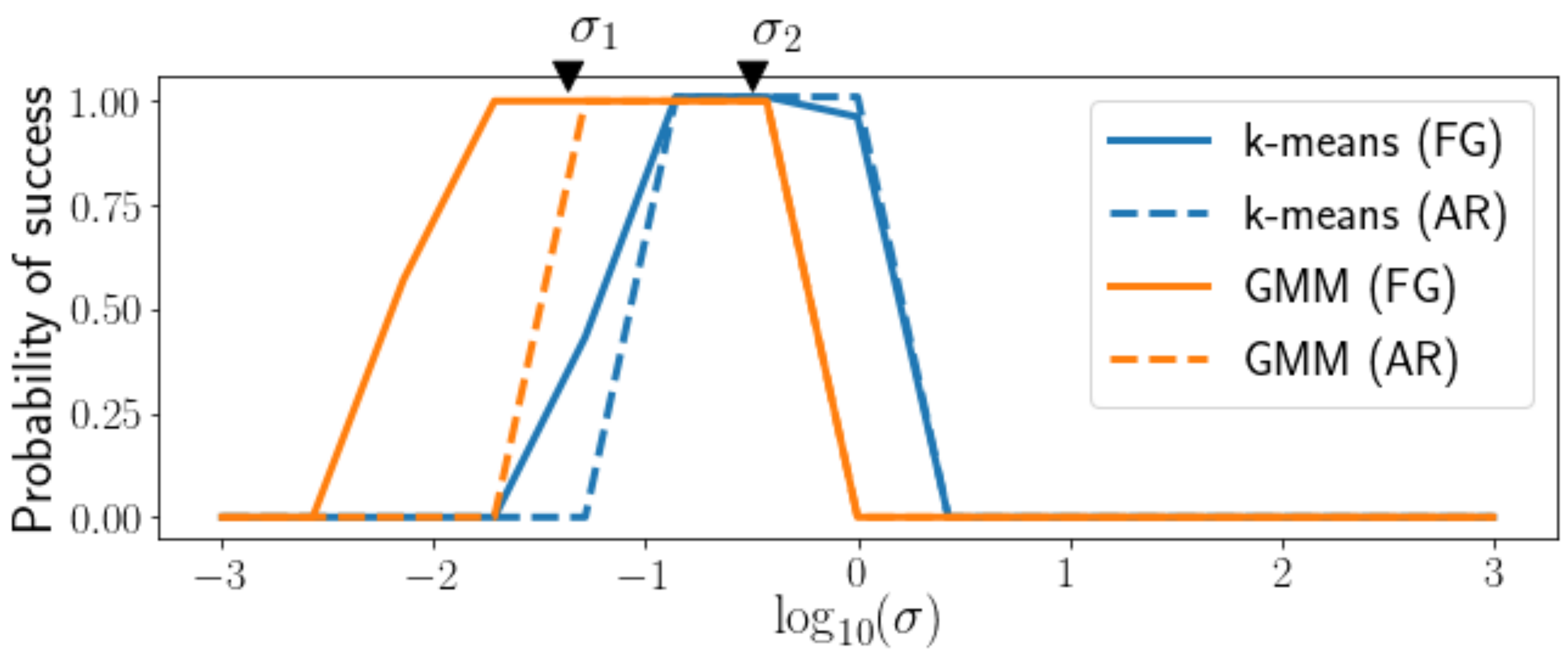


c)

Take-home







- There are regimes where CLOMPR could be improved (small m or sigma for k-means)

- An ambiguous region remains at intermediary m (landscape probably full of local minima)

- Question for the future: can we actually do better than CLOMPR (efficiently)? Guarantees?