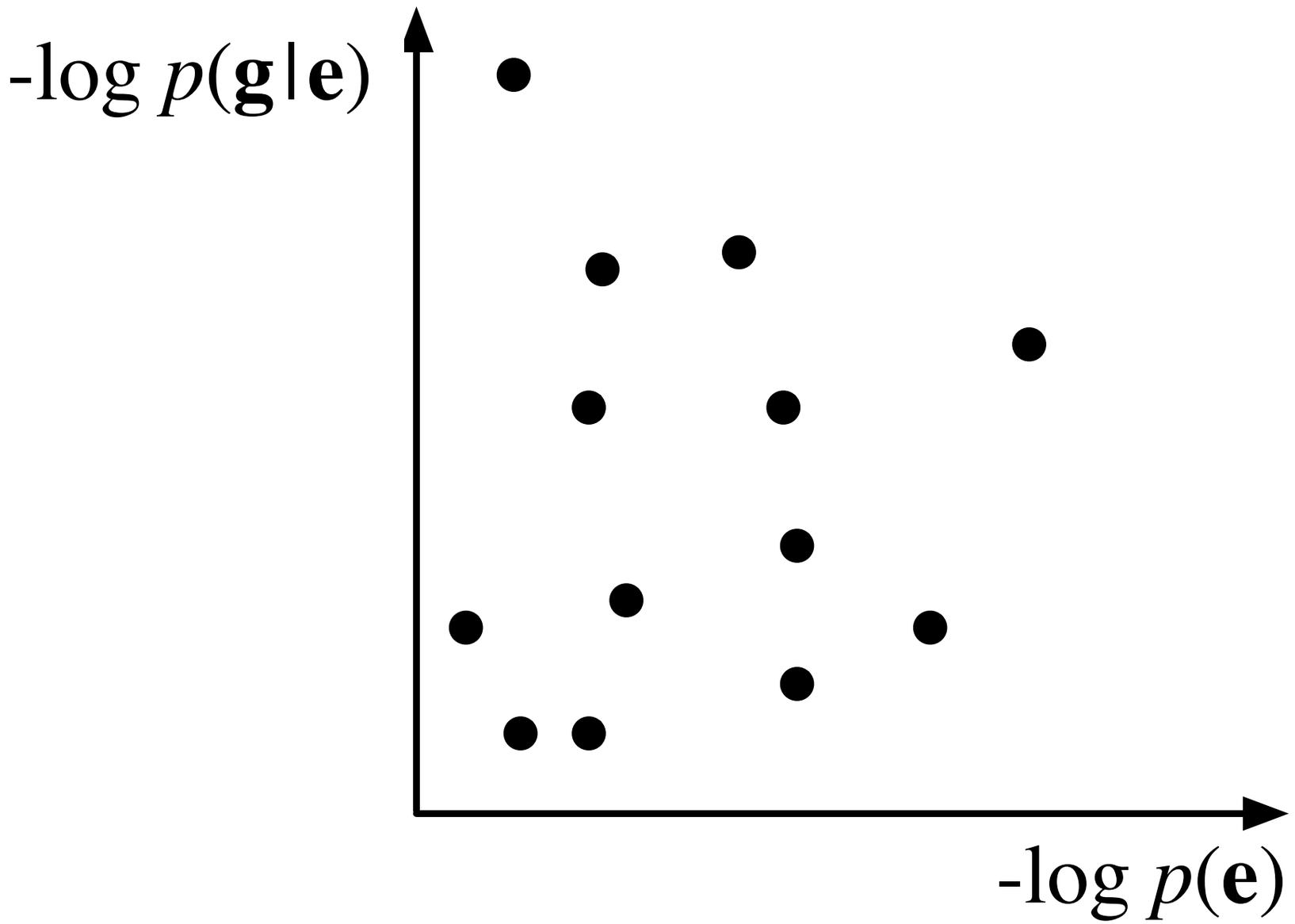
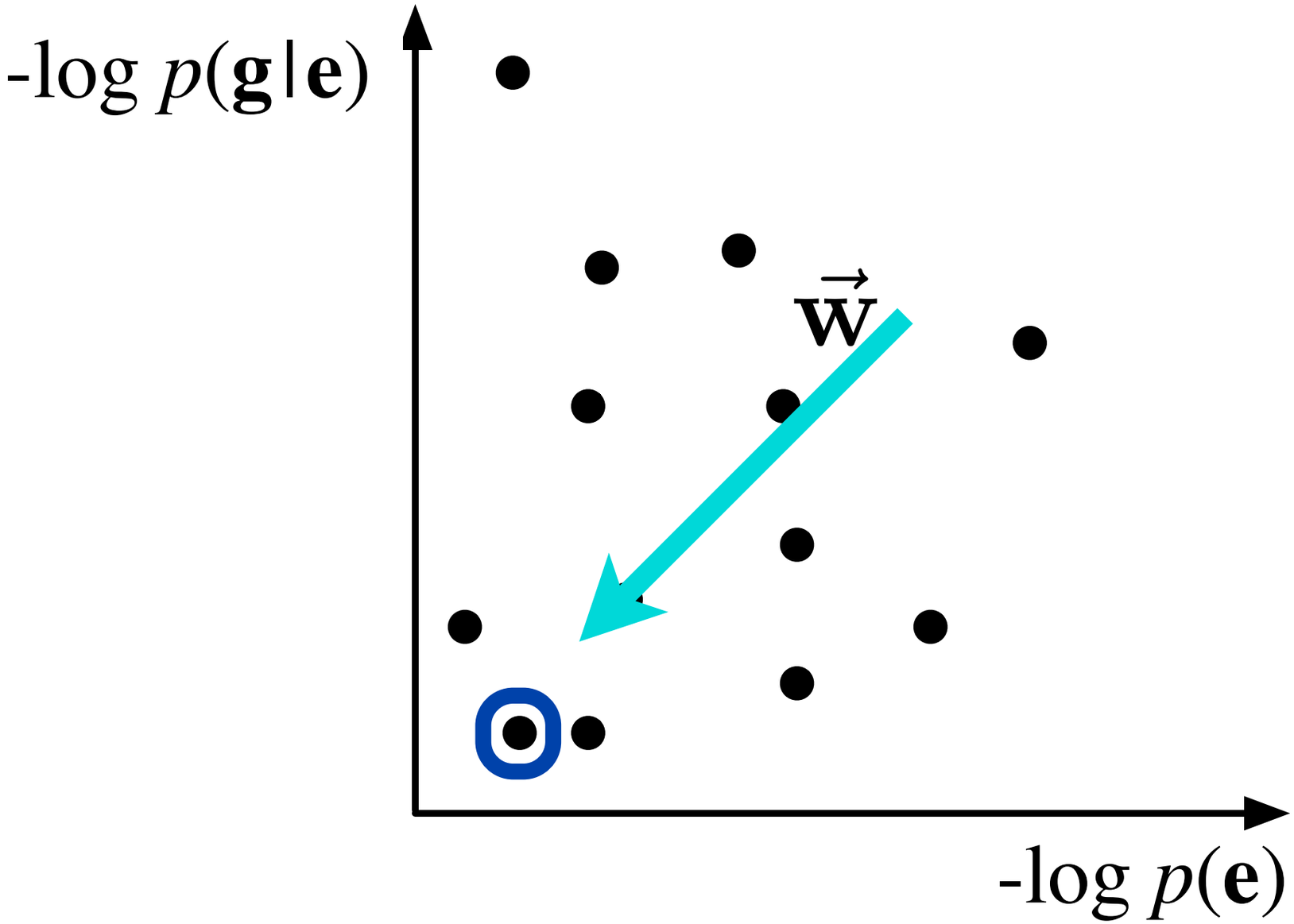
Discriminative Training part 2

March 6, 2014

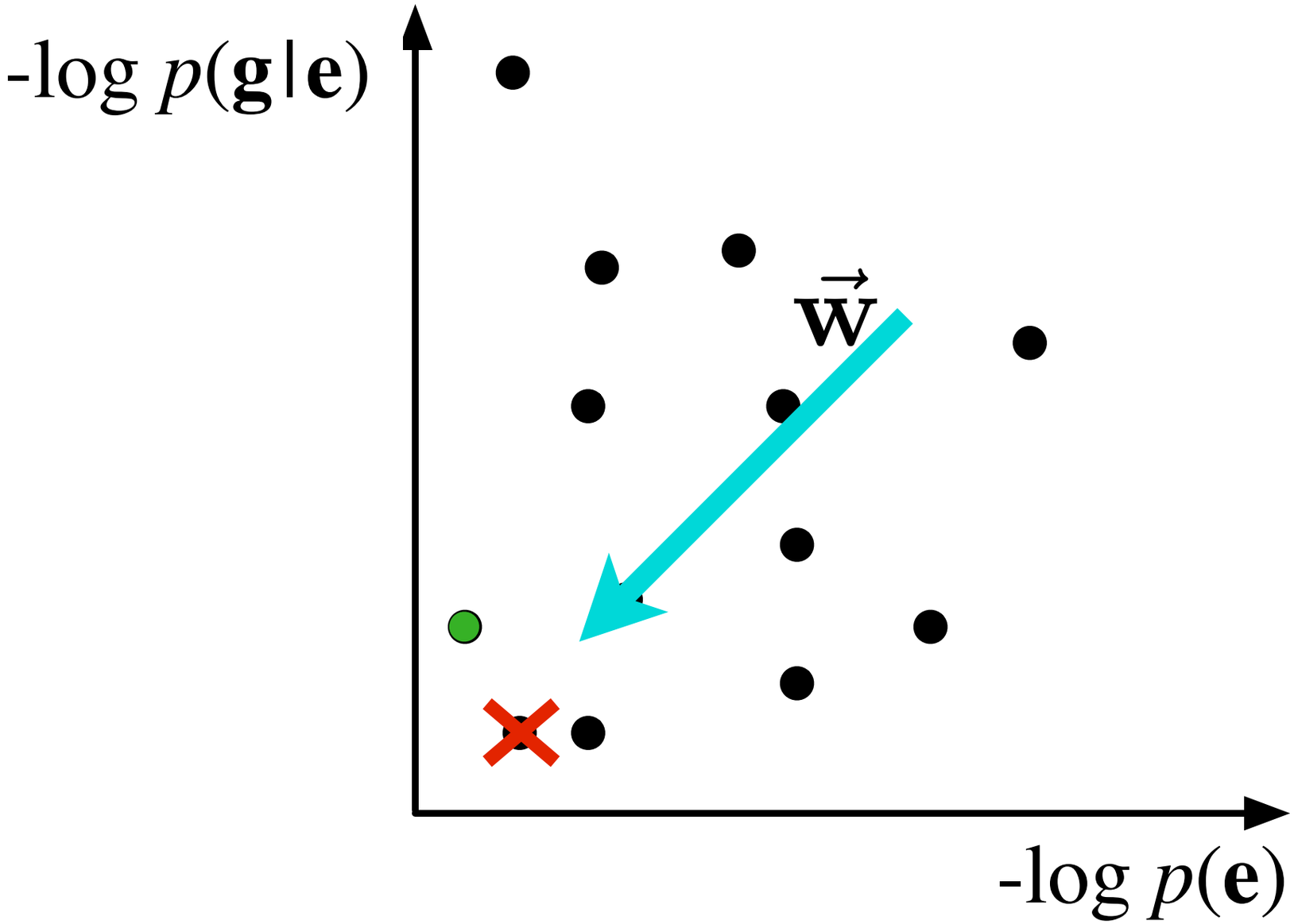
The Noisy Channel



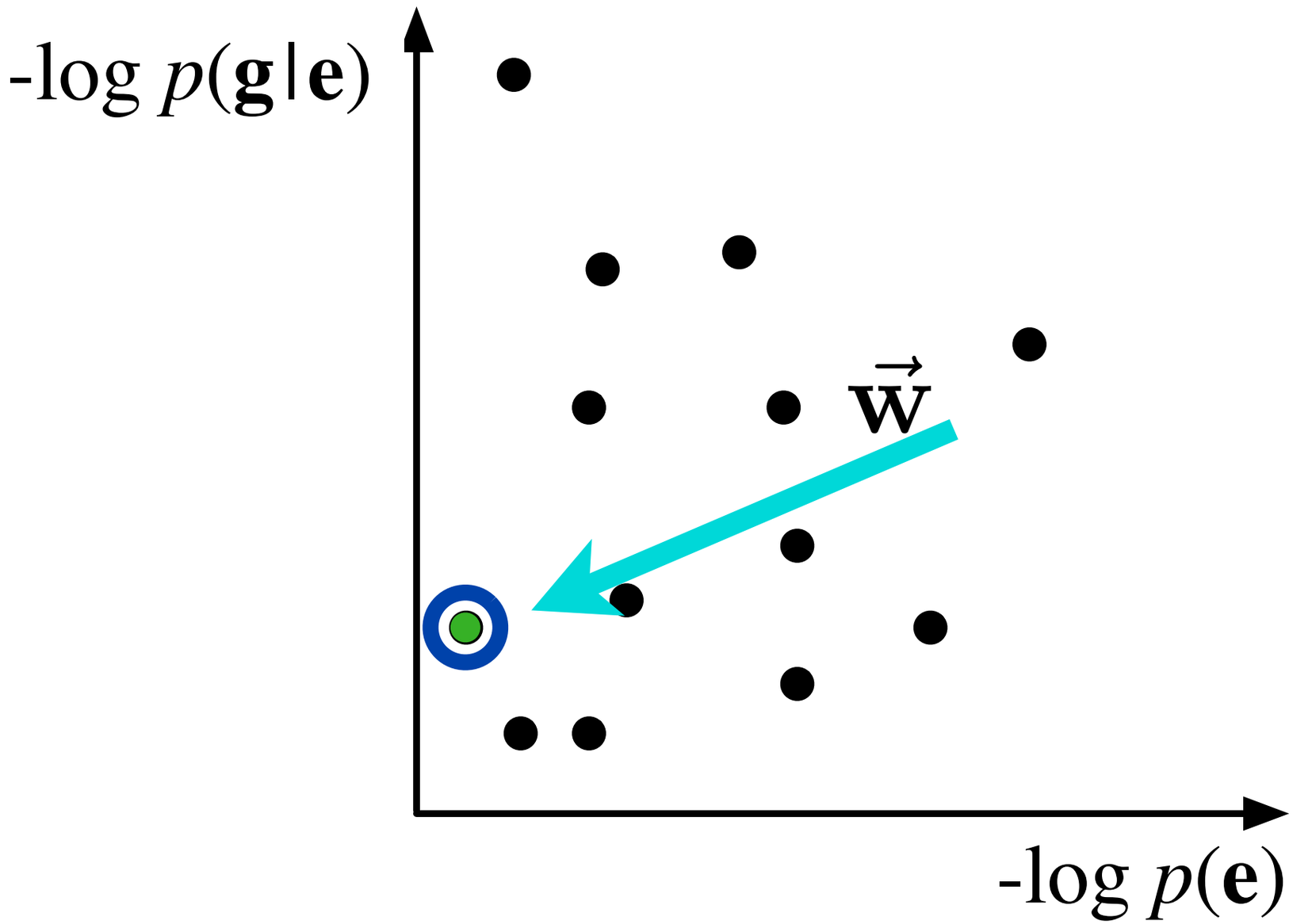
As a Linear Model



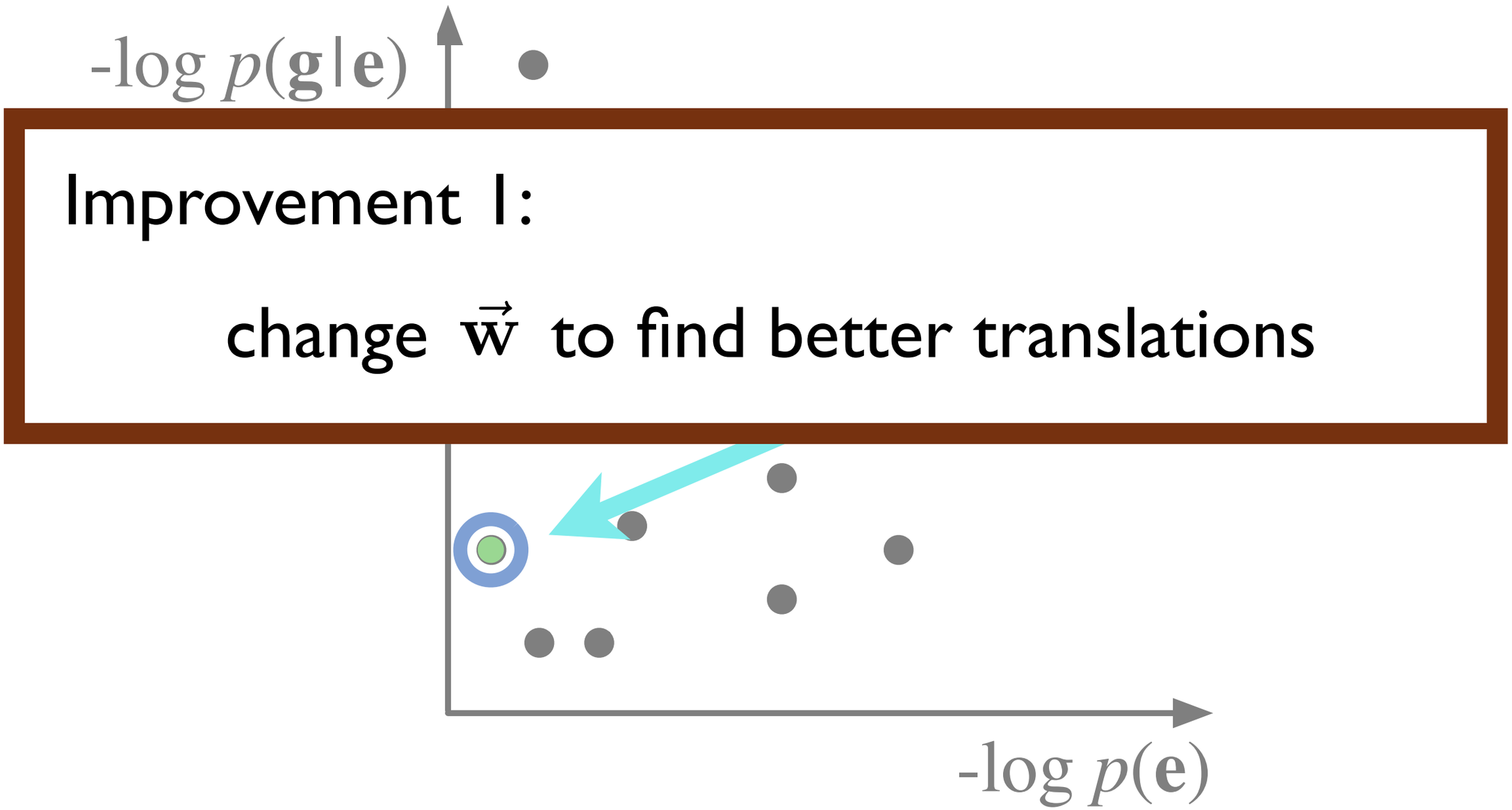
As a Linear Model



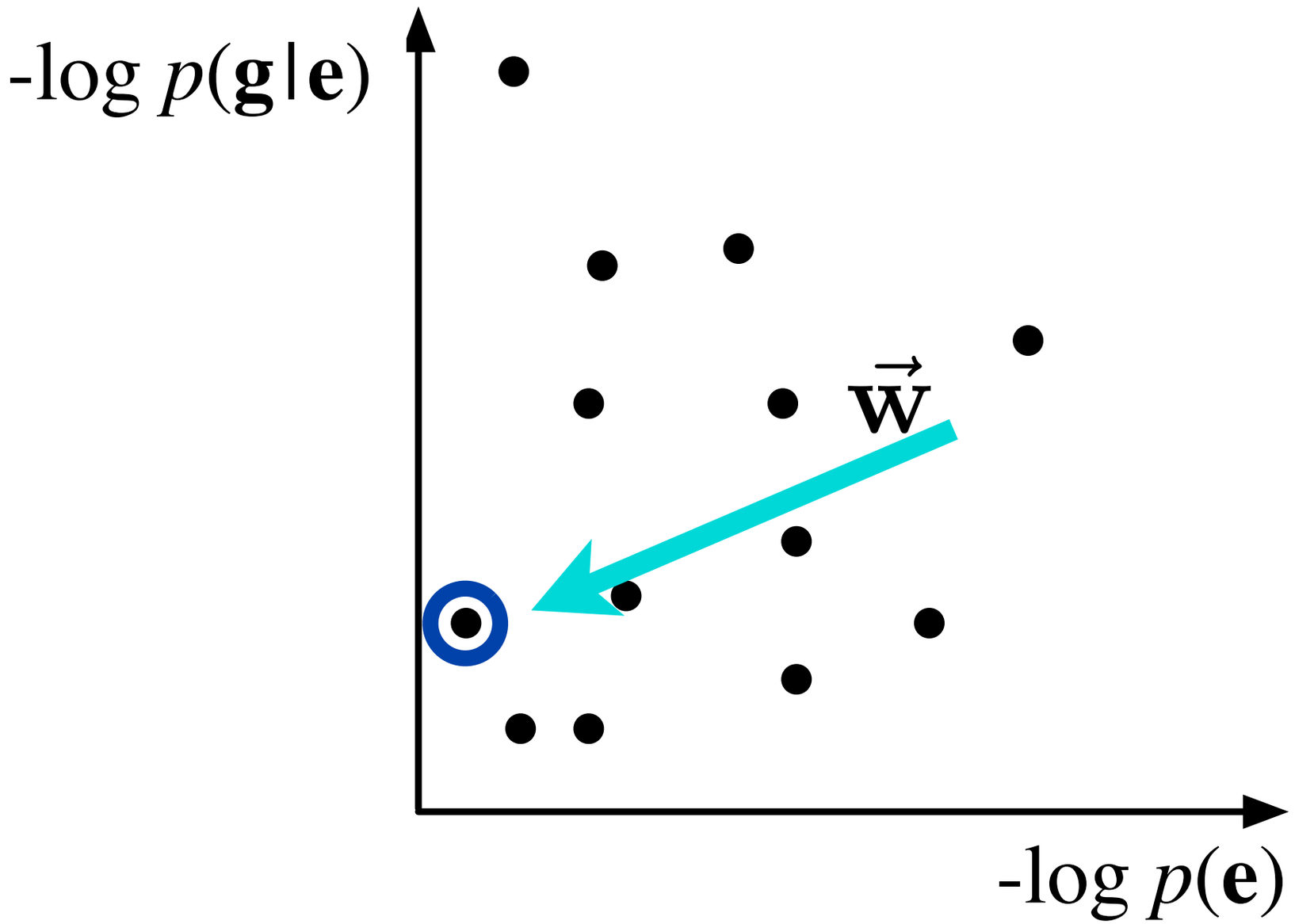
As a Linear Model



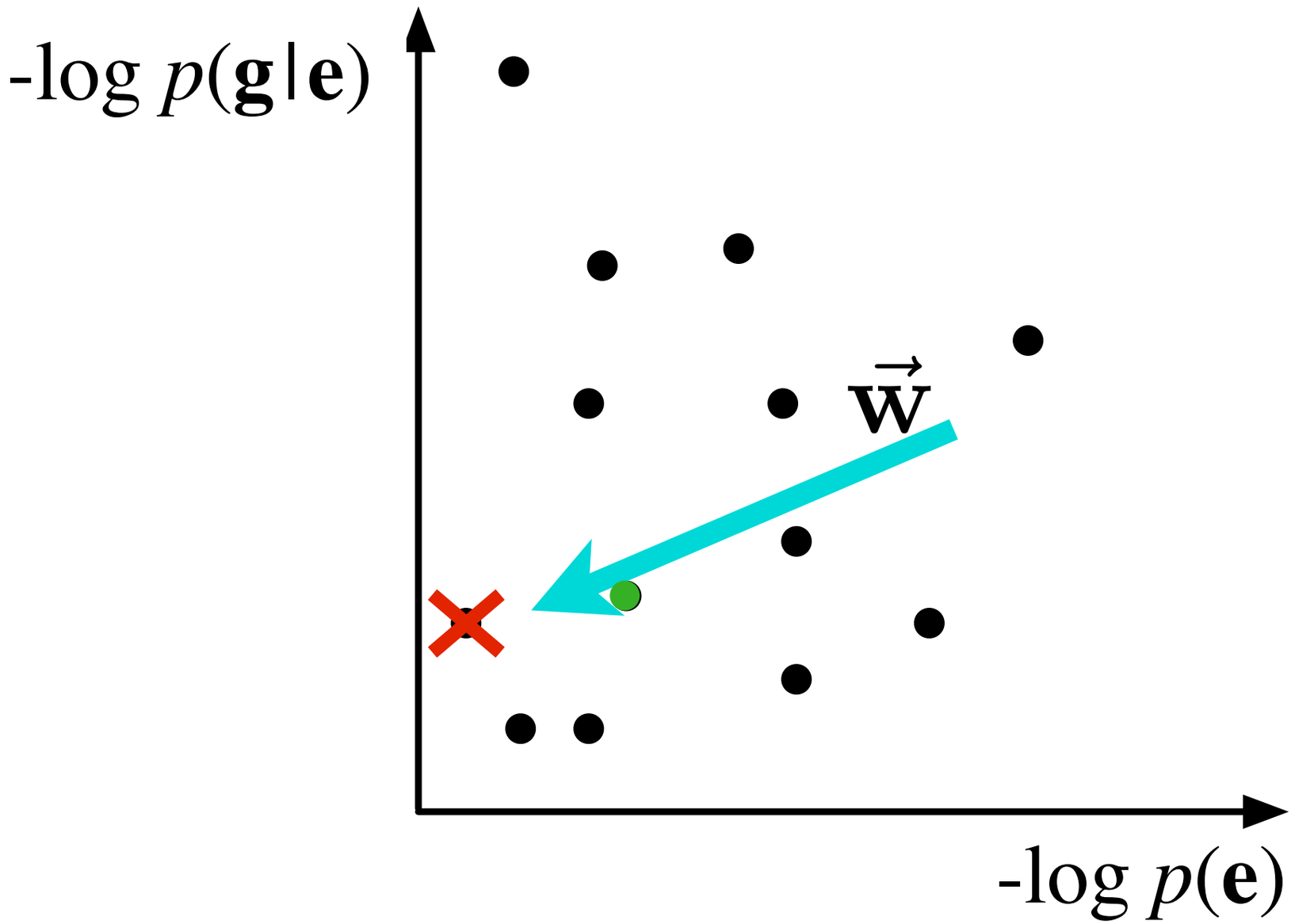
As a Linear Model



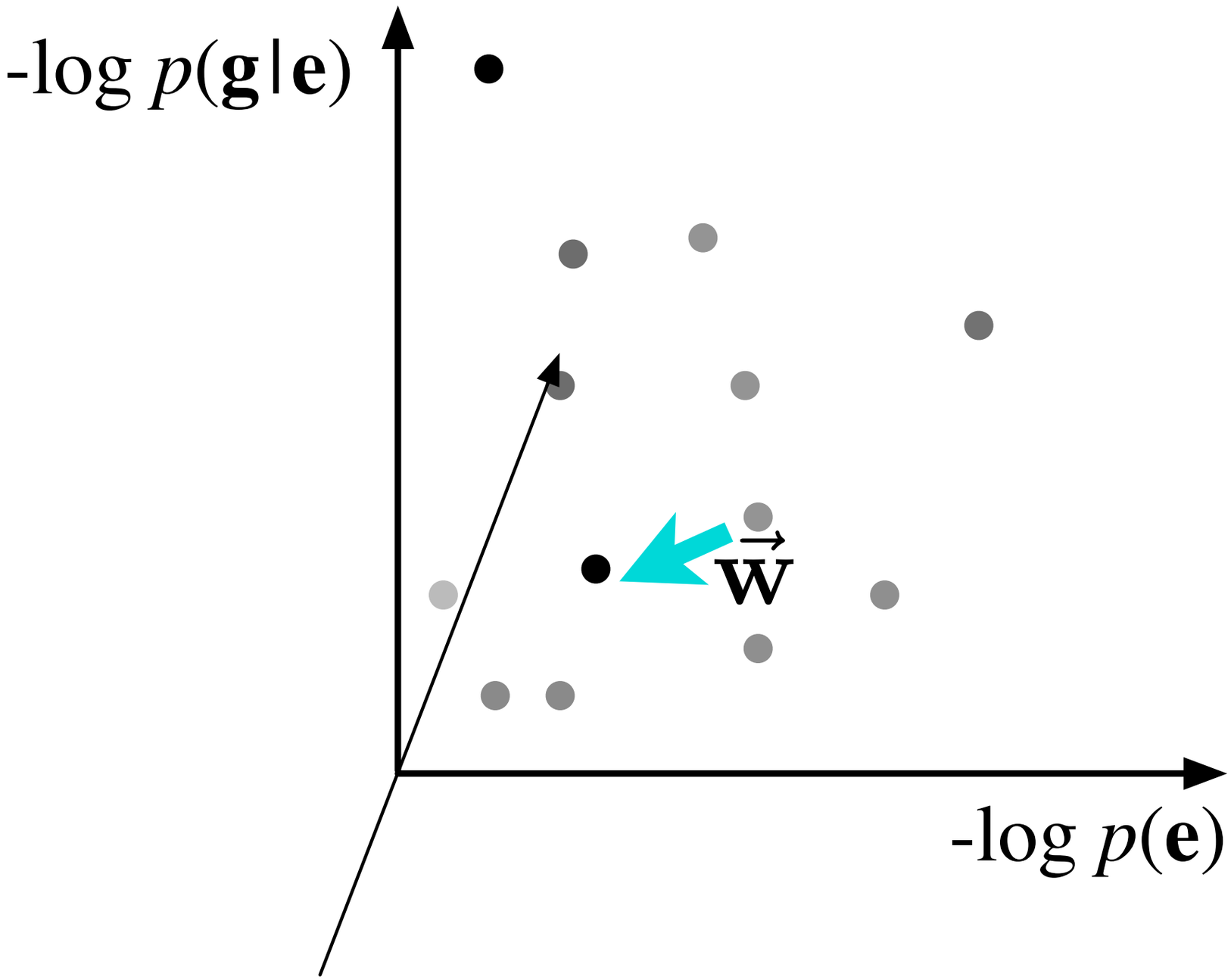
As a Linear Model



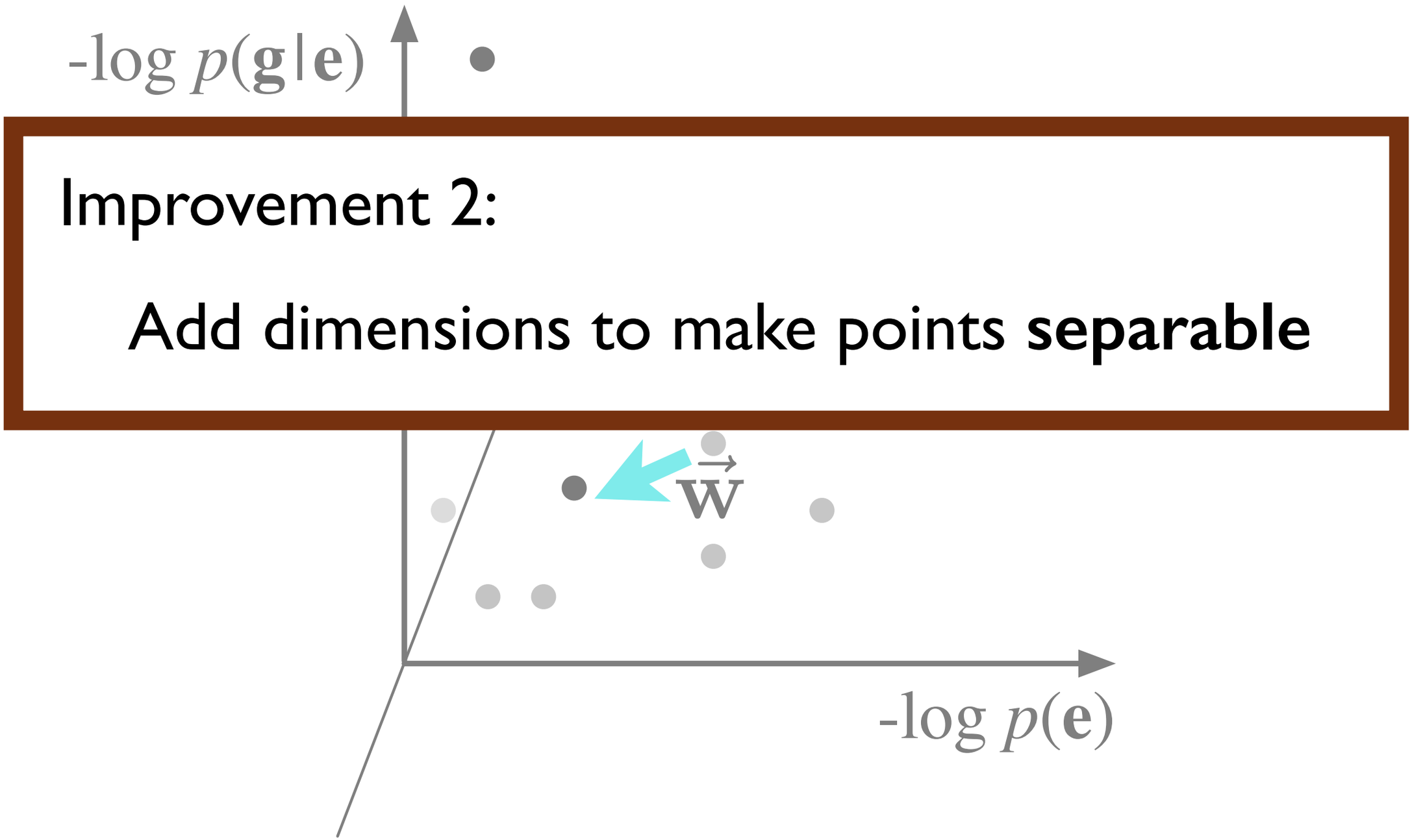
As a Linear Model



As a Linear Model



As a Linear Model



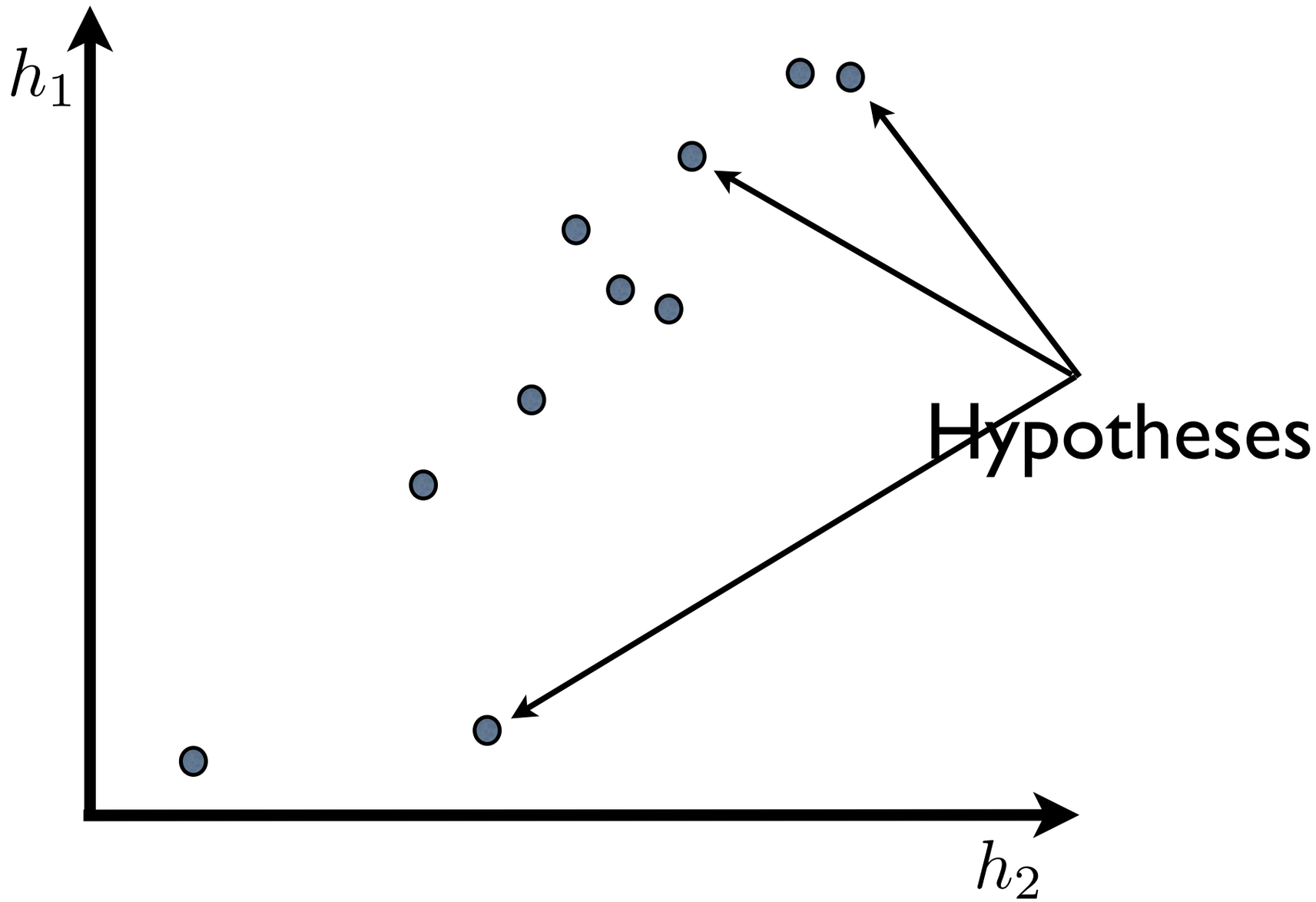
Linear Models

$$\mathbf{e}^* = \arg\max_{\mathbf{e}} \mathbf{w}^\top \mathbf{h}(\mathbf{g}, \mathbf{e})$$

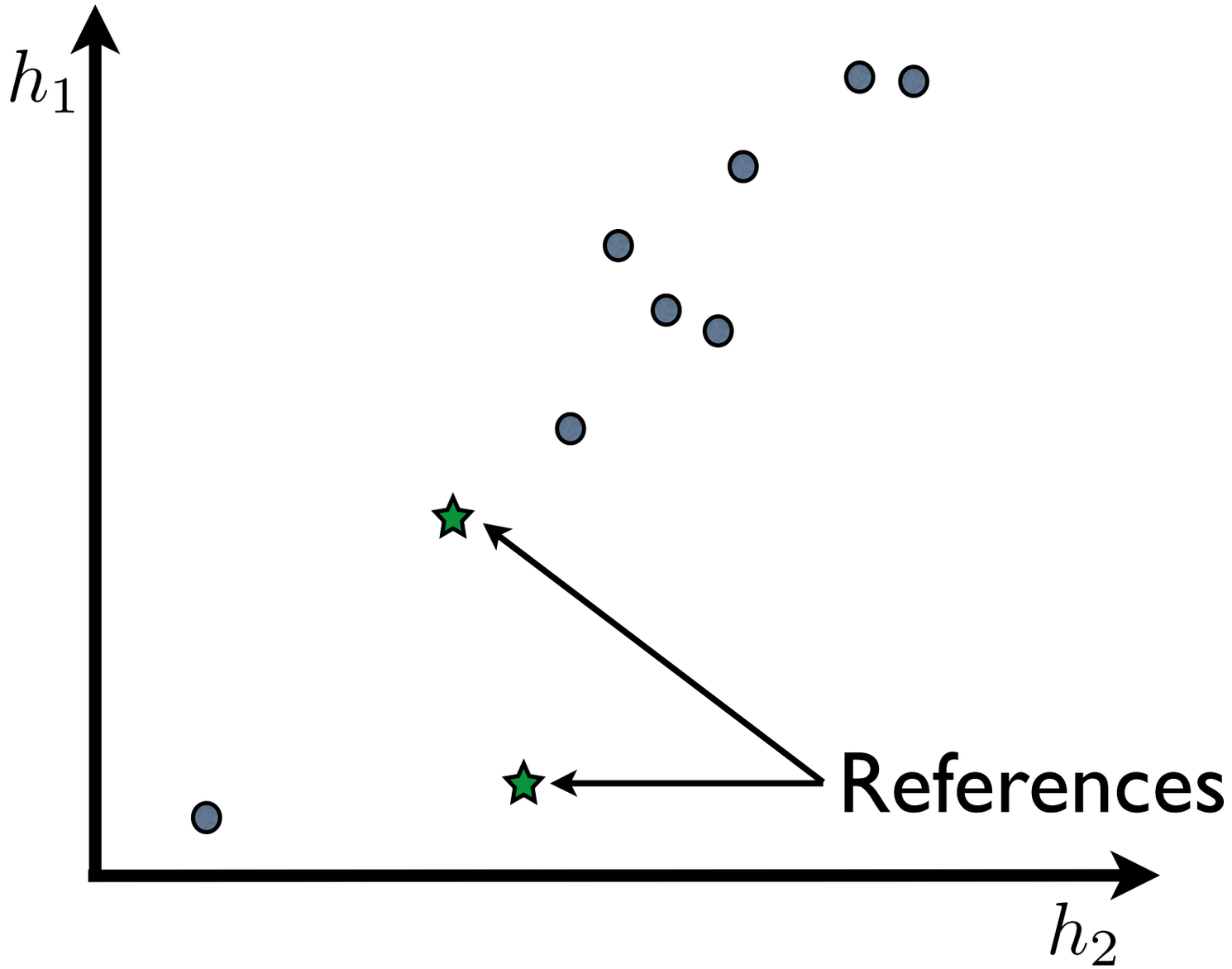
- Improve the modeling capacity of the noisy channel in two ways
  - Reorient the weight vector
  - Add new dimensions (*new features*)
- Questions
  - What features? $\mathbf{h}(\mathbf{g}, \mathbf{e})$
  - How do we set the weights? $\mathbf{w}$

Parameter Learning

Hypothesis Space



Hypothesis Space



Preliminaries

We assume a **decoder** that computes:

$$\langle \mathbf{e}^*, \boxed{\mathbf{a}^*} \rangle = \arg\max_{\langle \mathbf{e}, \boxed{\mathbf{a}} \rangle} \mathbf{w}^\top \mathbf{h}(\mathbf{g}, \mathbf{e}, \boxed{\mathbf{a}})$$

And *K*-**best lists** of, that is:

$$\{\langle \mathbf{e}_i^*, \boxed{\mathbf{a}_i^*} \rangle\}_{i=1}^{i=K} = \arg i^{\text{th}}\text{-}\max_{\langle \mathbf{e}, \boxed{\mathbf{a}} \rangle} \mathbf{w}^\top \mathbf{h}(\mathbf{g}, \mathbf{e}, \boxed{\mathbf{a}})$$

Standard, efficient algorithms exist for this.

Cost-Sensitive Training

- Assume we have a **cost function** that gives a score for how good/bad a translation is

$$\ell(\hat{\mathbf{e}}, \mathcal{E}) \mapsto [0, 1]$$

- Optimize the weight vector by making reference to this function
  - We will talk about two ways to do this

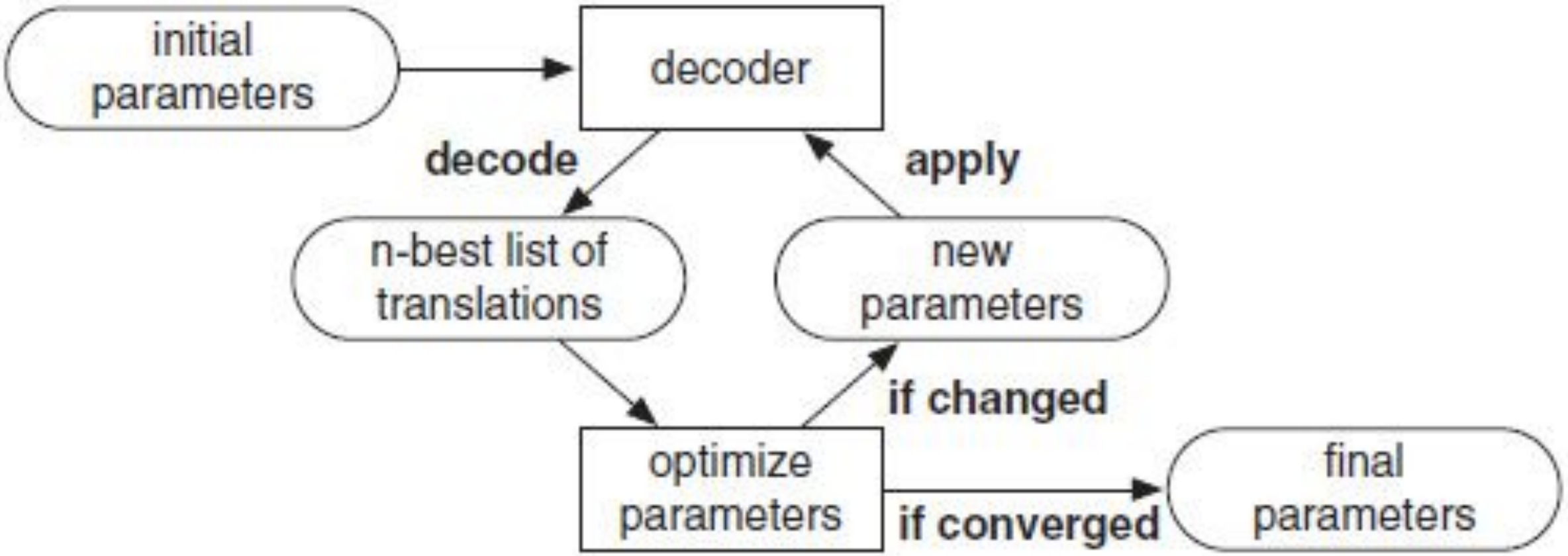
MERT

- **M**inimum **E**rror **R**ate **T**raining
- Directly optimize for an automatic evaluation metric instead of likelihood
- Maximize the BLEU score on a **held out development set**
- Iteratively update the parameters by re-scoring n-best lists and comparing the highest scoring translation to the reference

MERT

- Even with 10-15 features it's not possible to exhaustively search the space of possible feature values
- We need a good heuristic method to search the space
- Another problem: the initial parameters might be so bad that the original n-best list is not a good sample of the translations

Iterative parameter tuning



Powell Search

- Explore a high-dimensional space by finding a better point along one line in the space
- Simplest form: Vary one parameter at a time
- If the optimal value is better than the current value, then change it and move to the next parameter
- Iterate until there are no single parameter updates that increase the score

Powell Search

- Problem: searching for the best value for a single parameter is still daunting
  - Parameters are real-valued #s, so they have an infinite number of possible values
- Key insight of MERT: only a small number of threshold values will change the 1-best translation
  - Only 1-best translations change BLEU

Finding the threshold points for 1 sentence

Given weight vector $\mathbf{w}$, any hypothesis $\langle \mathbf{e}, \mathbf{a} \rangle$ will have a (scalar) score $m = \mathbf{w}^\top \mathbf{h}(\mathbf{g}, \mathbf{e}, \mathbf{a})$

Now pick a **search vector v**, and consider how the score of this hypothesis will change:

$$\mathbf{w}_{\text{new}} = \mathbf{w} + \gamma \mathbf{v}$$

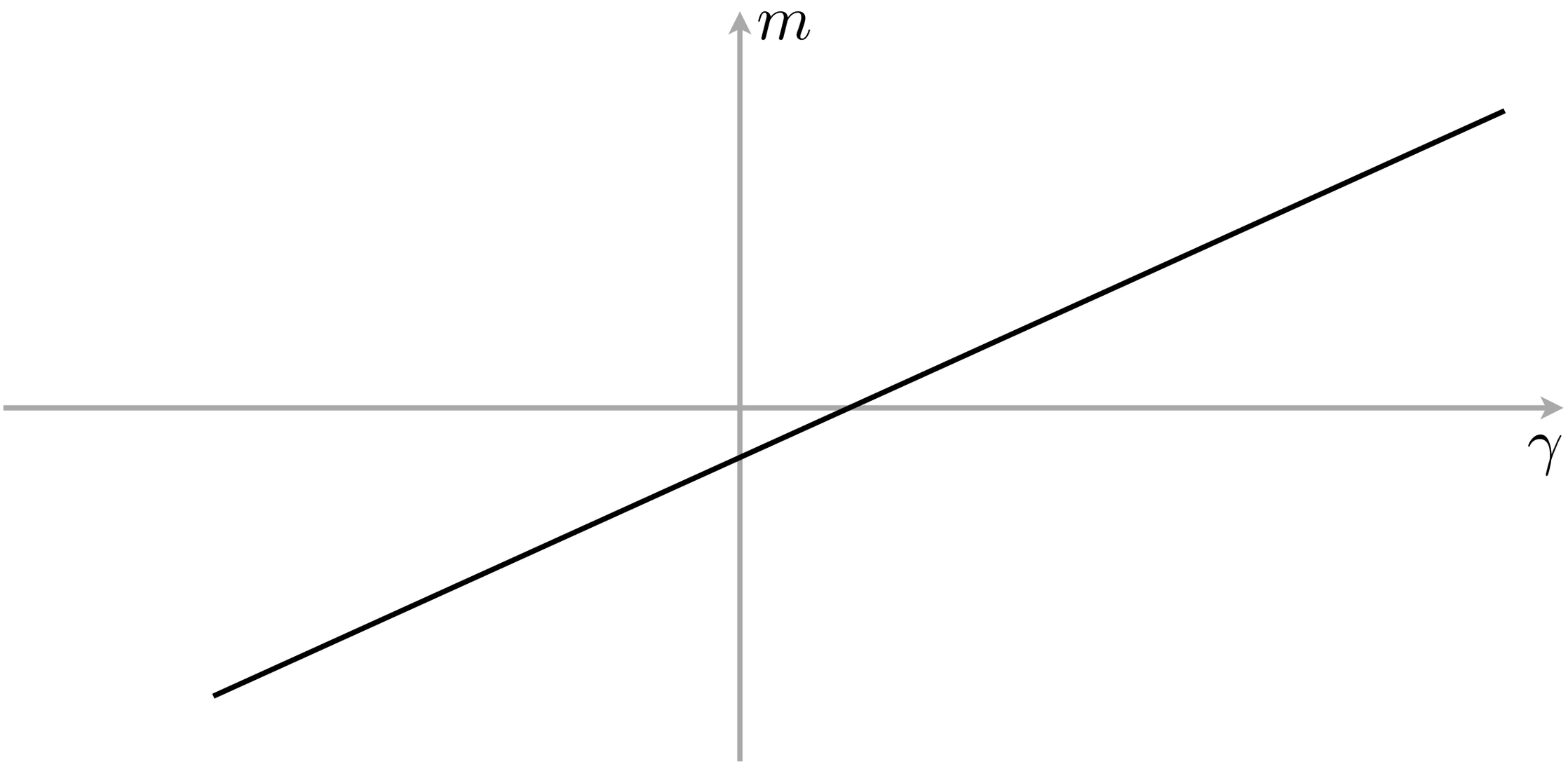
$$m = (\mathbf{w} + \gamma \mathbf{v})^\top \mathbf{h}(\mathbf{g}, \mathbf{e}, \mathbf{a})$$

$$= \underbrace{\mathbf{w}^\top \mathbf{h}(\mathbf{g}, \mathbf{e}, \mathbf{a})}_{b} + \gamma \underbrace{\mathbf{v}^\top \mathbf{h}(\mathbf{g}, \mathbf{e}, \mathbf{a})}_{a}$$

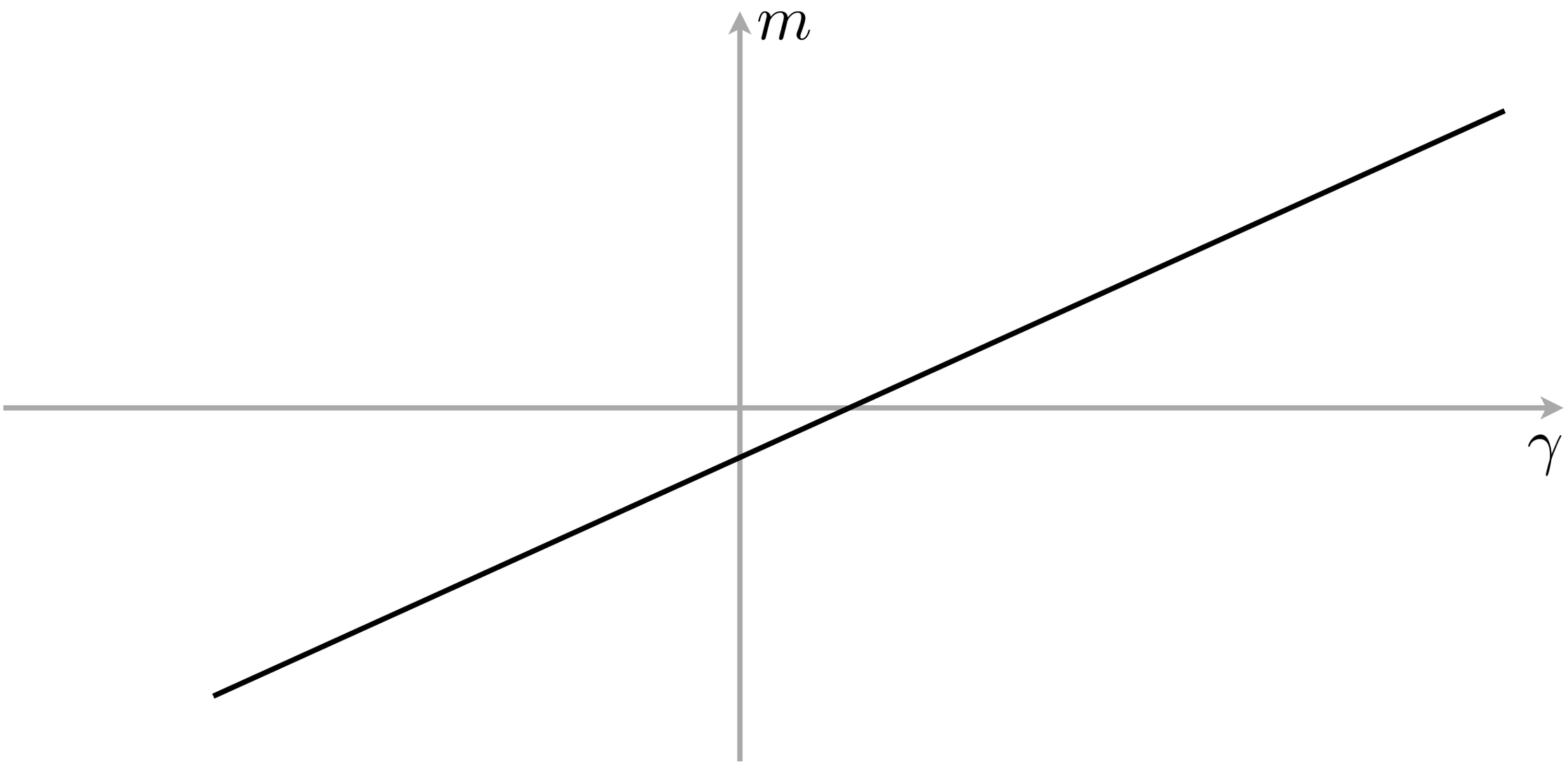
$$m = a\gamma + b$$

Linear function in 2D!

MERT

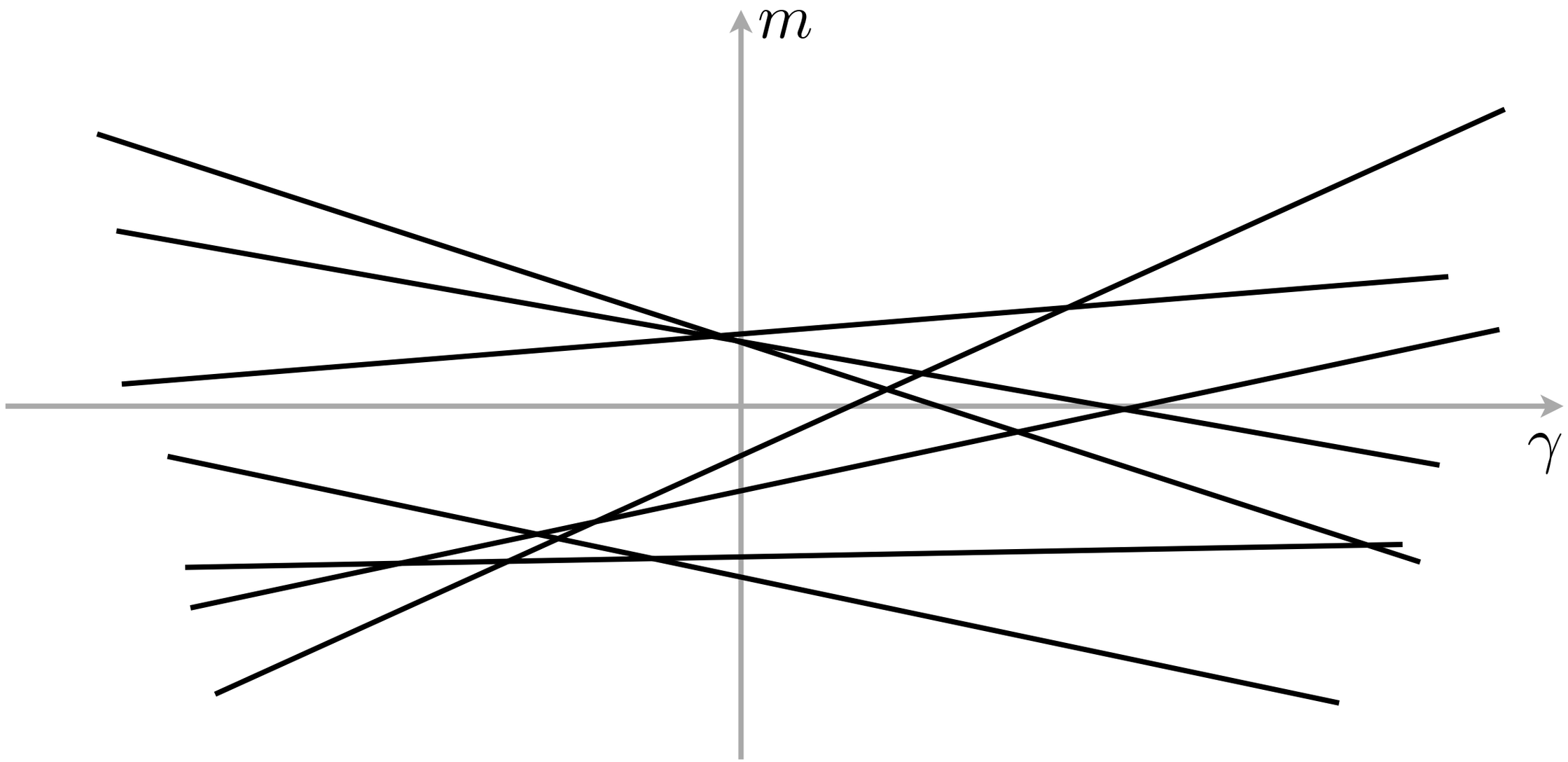


MERT



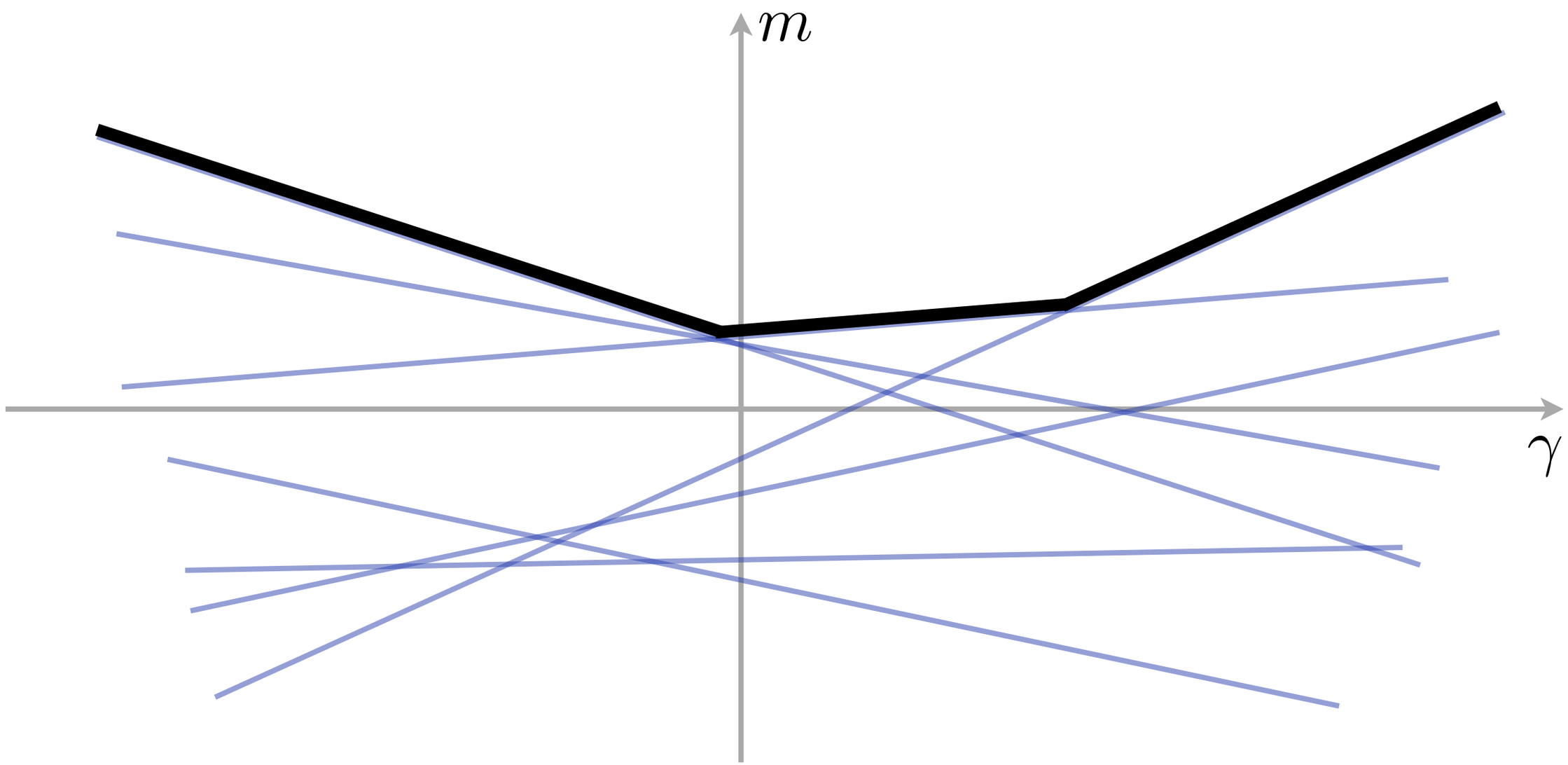
Recall our k-best set $\{\langle \mathbf{e}_i^*, \mathbf{a}_i^* \rangle\}_{i=1}^K$

MERT

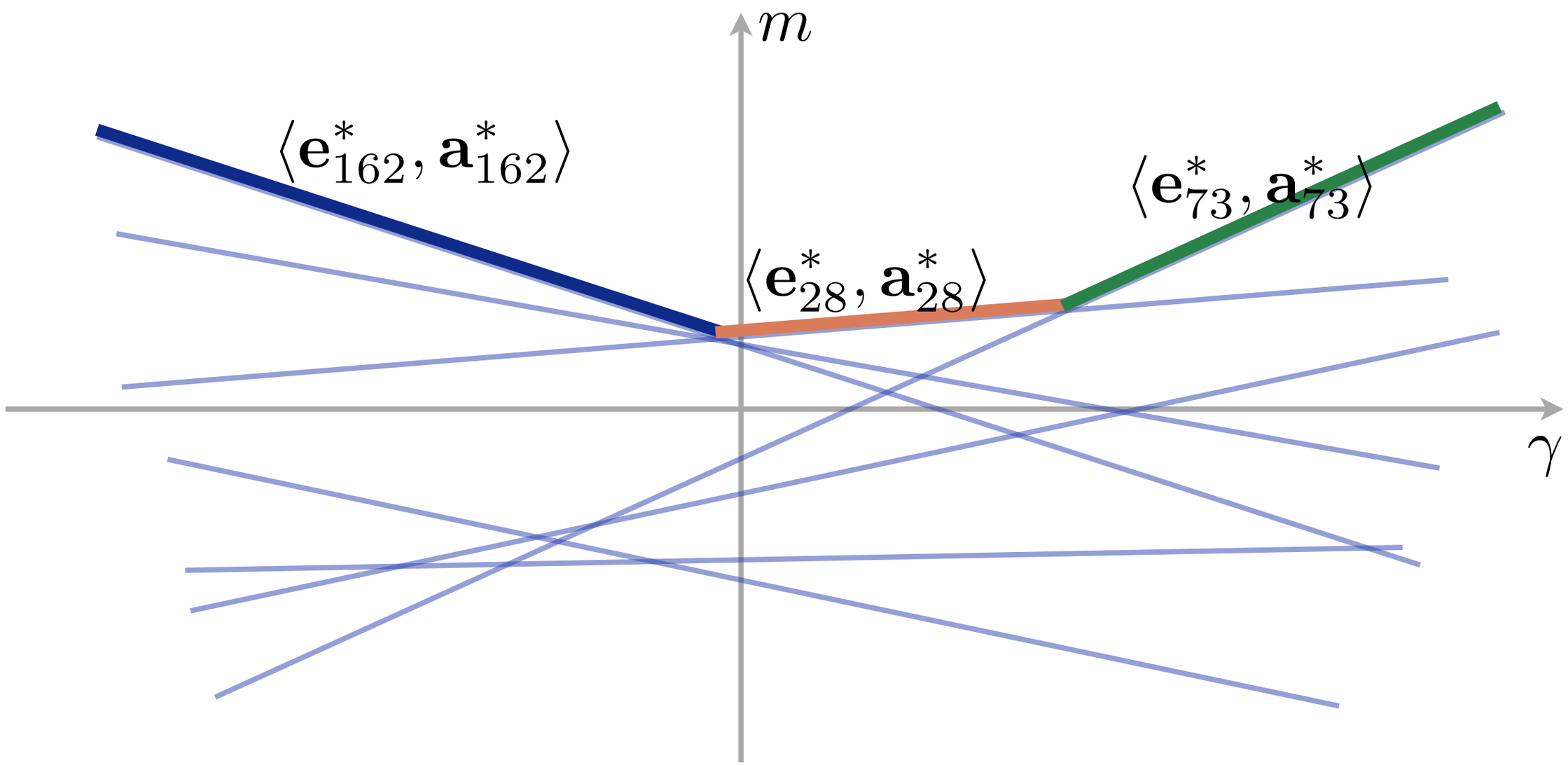


Recall our k-best set $\{\langle \mathbf{e}_i^*, \mathbf{a}_i^* \rangle\}_{i=1}^K$

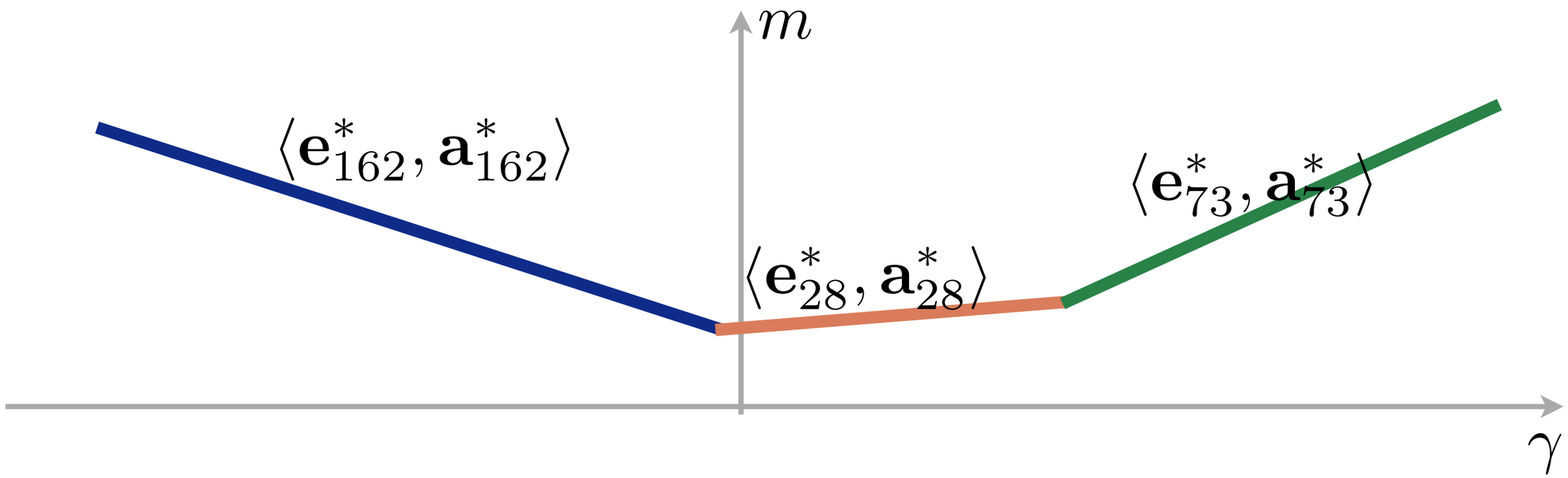
MERT



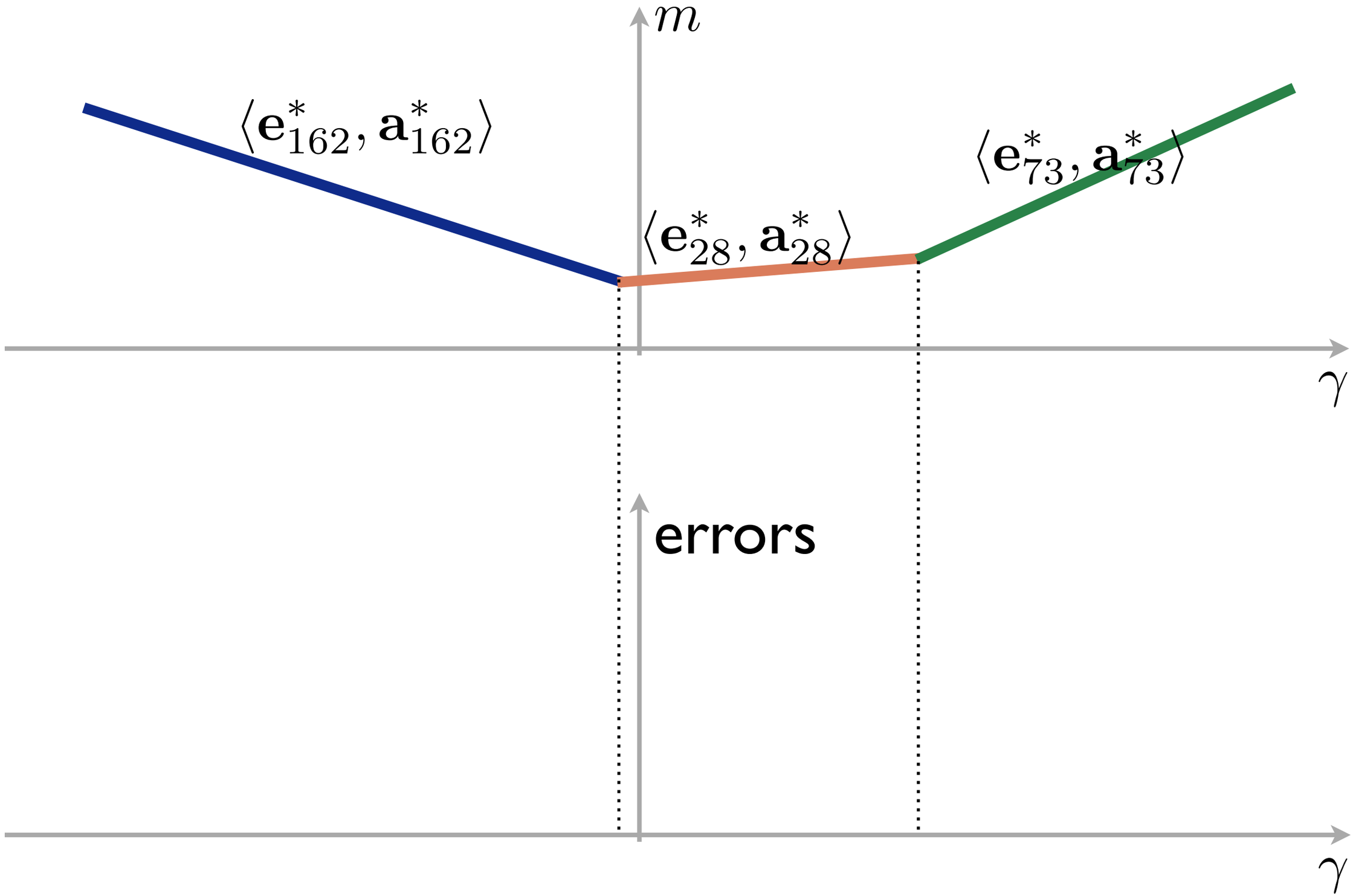
MERT


m
$\langle \mathbf{e}^*_{162}, \mathbf{a}^*_{162} \rangle$
$\langle \mathbf{e}^*_{28}, \mathbf{a}^*_{28} \rangle$
$\langle \mathbf{e}^*_{73}, \mathbf{a}^*_{73} \rangle$
γ

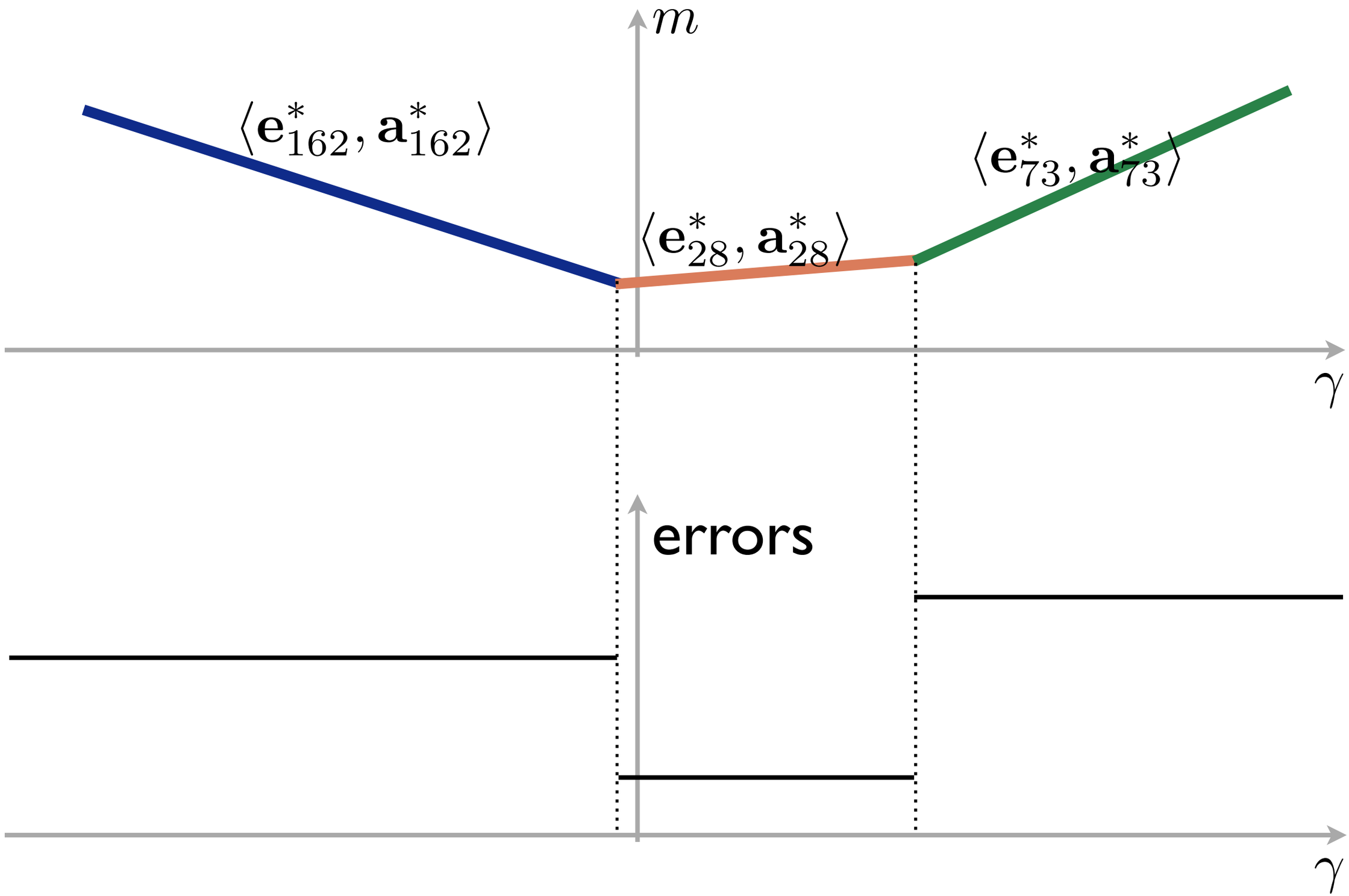
MERT


m
$\langle \mathbf{e}^*_{162}, \mathbf{a}^*_{162} \rangle$
$\langle \mathbf{e}^*_{28}, \mathbf{a}^*_{28} \rangle$
$\langle \mathbf{e}^*_{73}, \mathbf{a}^*_{73} \rangle$
γ

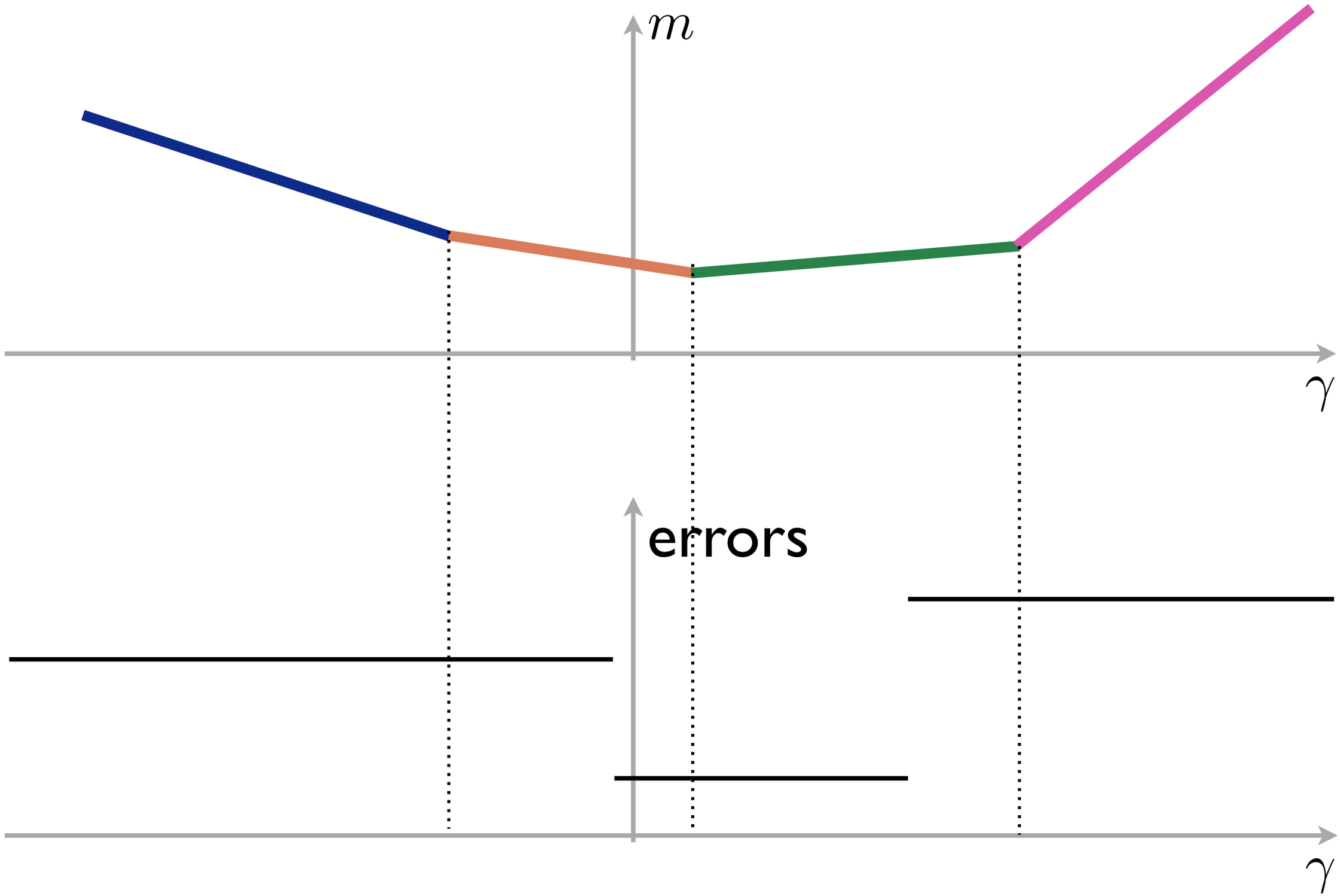
MERT


m
$\langle \mathbf{e}^*_{162}, \mathbf{a}^*_{162} \rangle$
$\langle \mathbf{e}^*_{28}, \mathbf{a}^*_{28} \rangle$
$\langle \mathbf{e}^*_{73}, \mathbf{a}^*_{73} \rangle$
γ
errors
γ

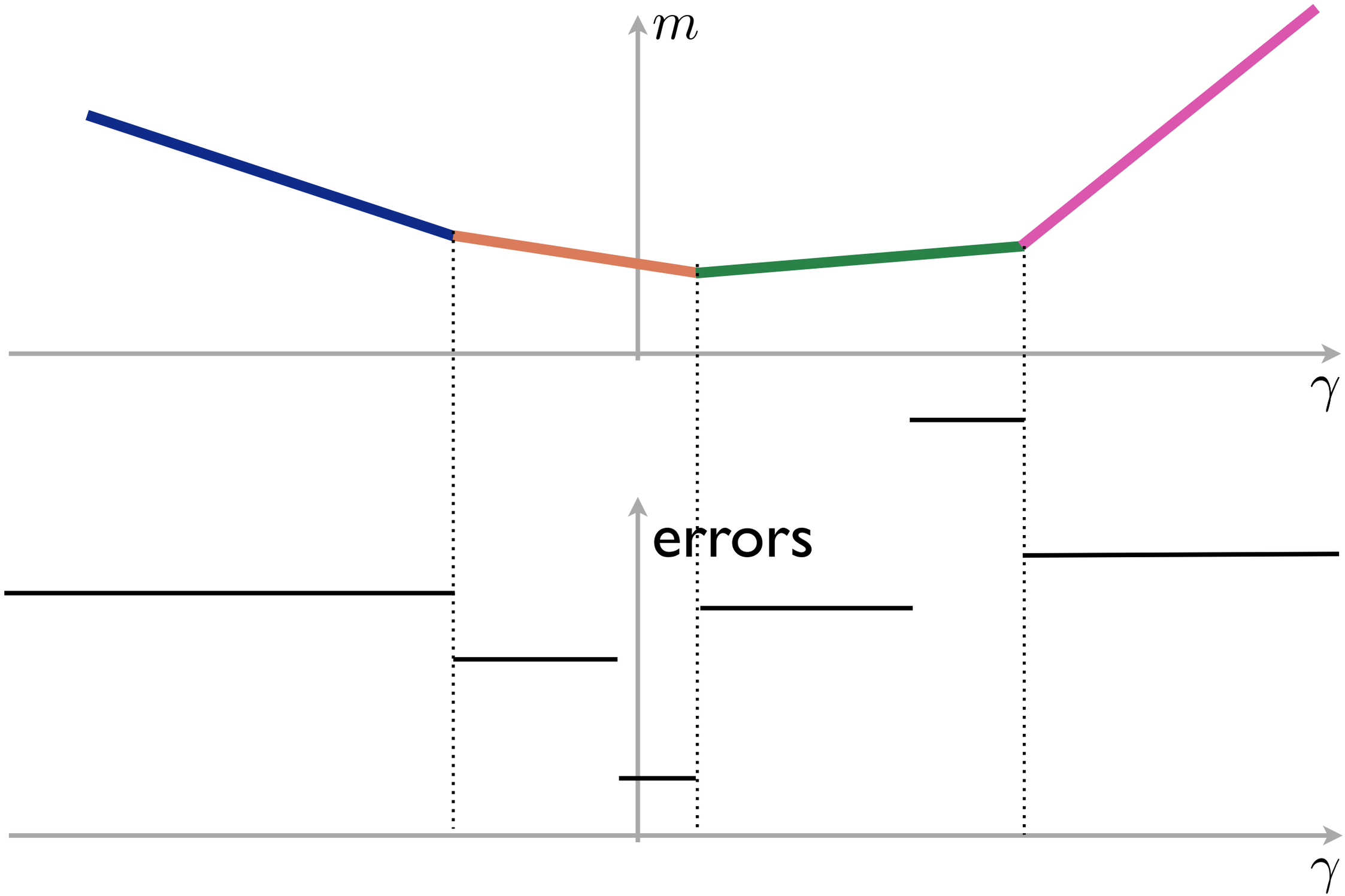
MERT

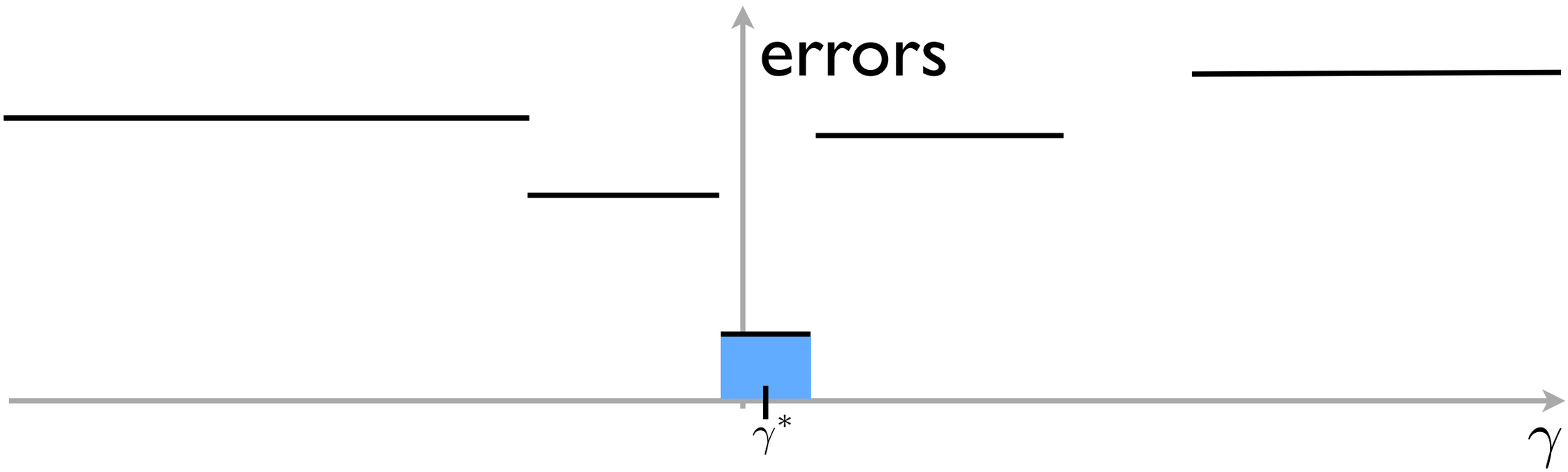

m
$\langle \mathbf{e}^*_{162}, \mathbf{a}^*_{162} \rangle$
$\langle \mathbf{e}^*_{28}, \mathbf{a}^*_{28} \rangle$
$\langle \mathbf{e}^*_{73}, \mathbf{a}^*_{73} \rangle$
γ
errors
γ

MERT



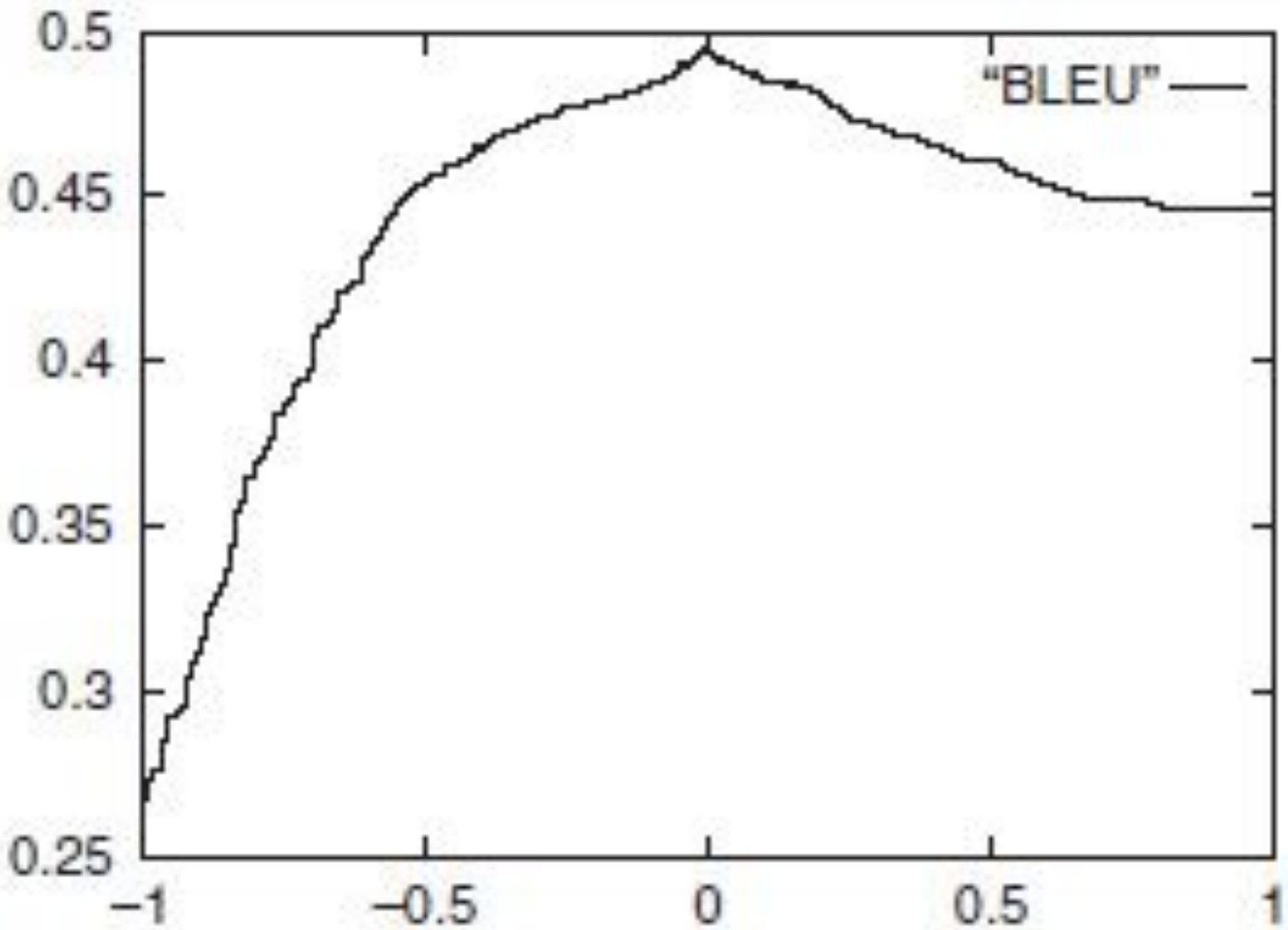
MERT



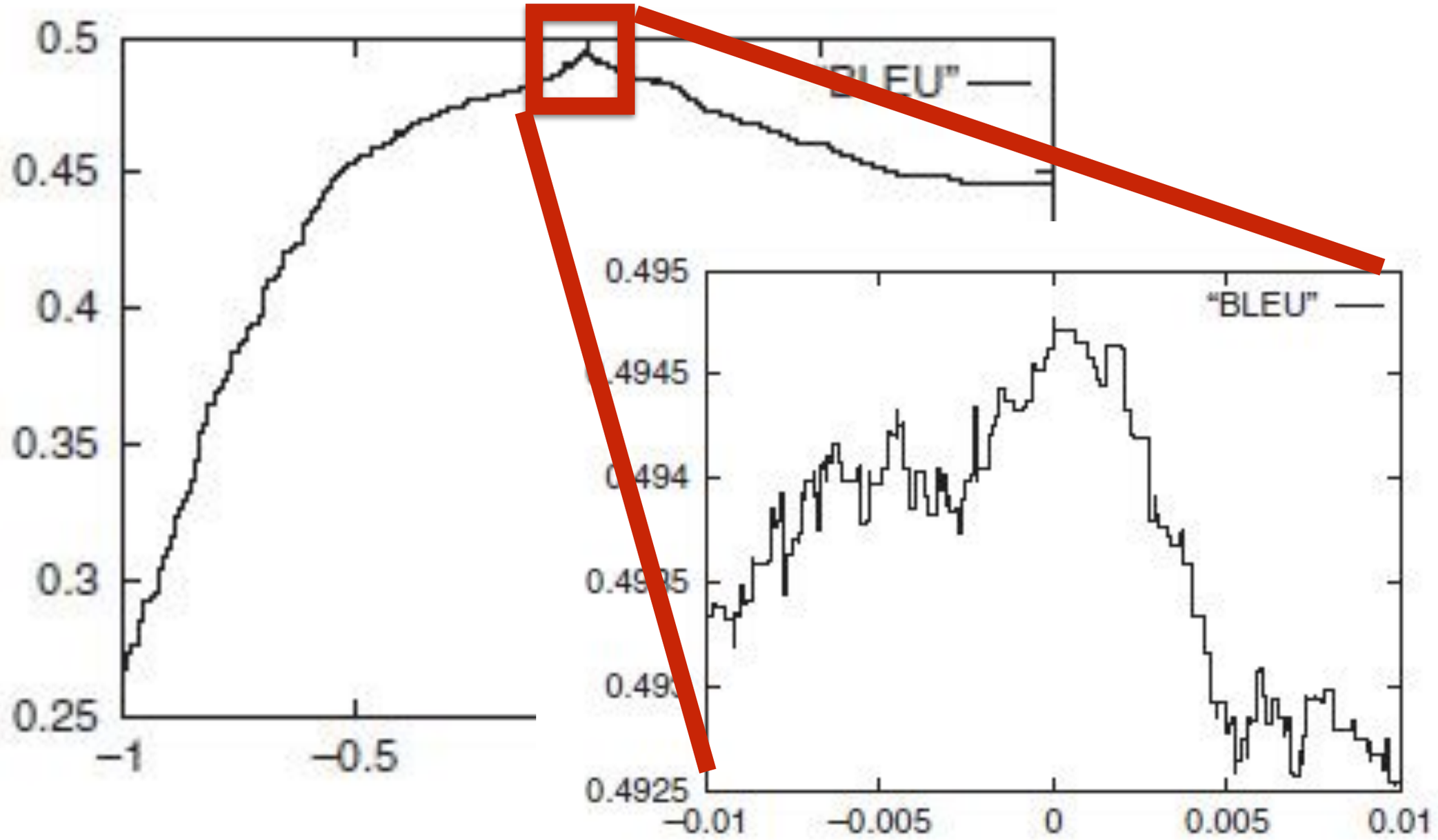


Let $\mathbf{w}_{\text{new}} = \gamma^* \mathbf{v} + \mathbf{w}$

The effect on BLEU varying one parameter



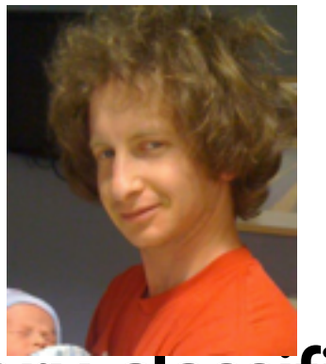
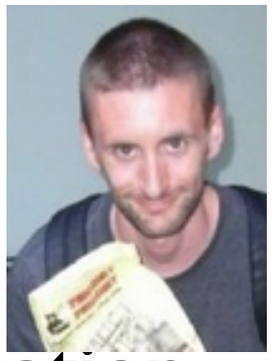
The effect on BLEU varying one parameter



MERT

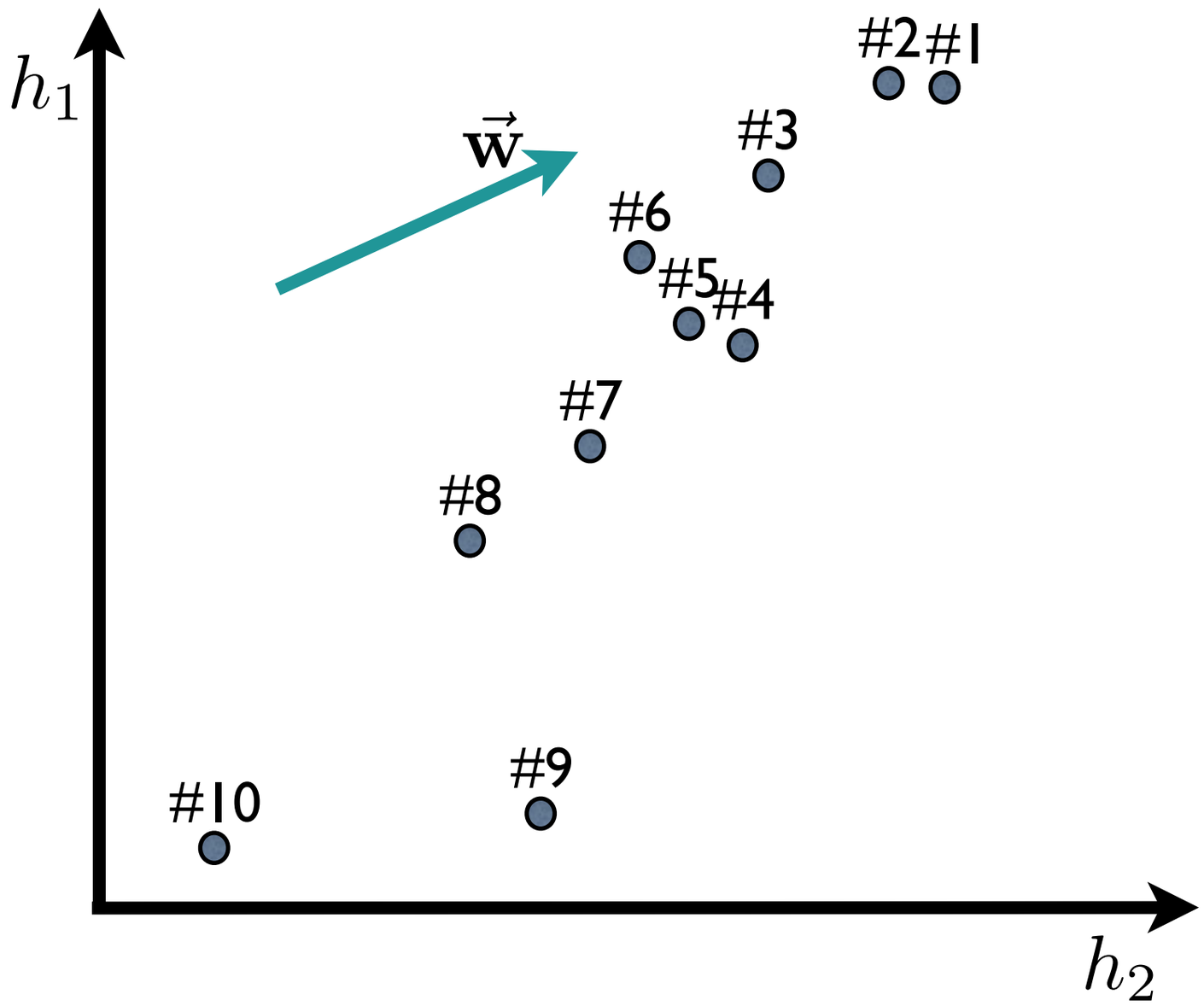
- Minimum error rate training
  - Can maximize or minimize!
- In practice “errors” are sufficient statistics for evaluation metrics (e.g., BLEU, AMBER, TER, etc)
- Downside: MERT can only be used to optimize a small handful of features

Training as Classification

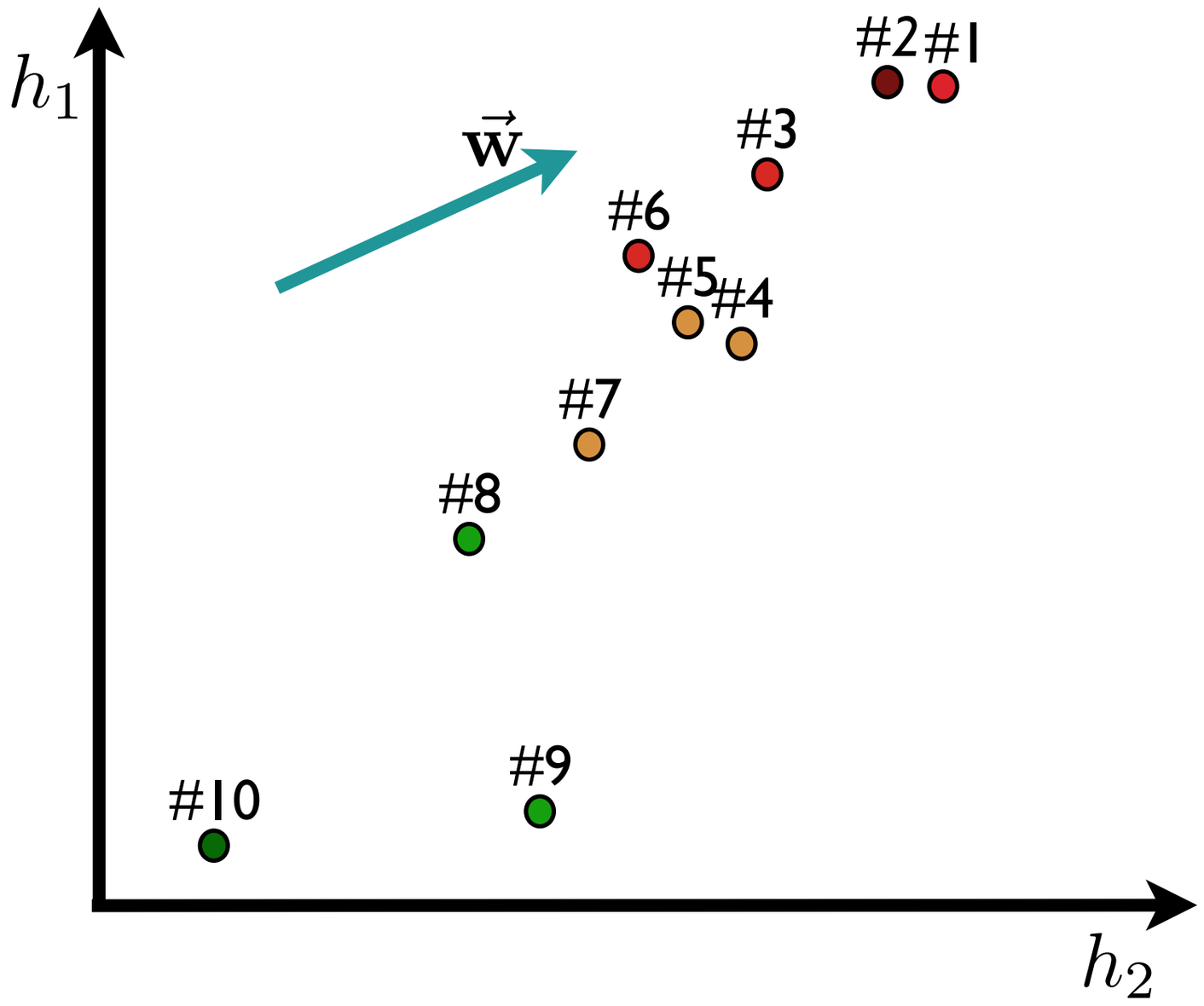



- Pairwise Ranking Optimization
  - Reduce training problem to **binary classification** with a **linear model**
- **Algorithm**
  - For *i*=1 to *N*
    - Pick random pair of hypotheses (A,B) from *K*-best list
    - Use cost function to determine if is A or B better
    - Create *i*th training instance
  - Train binary linear classifier

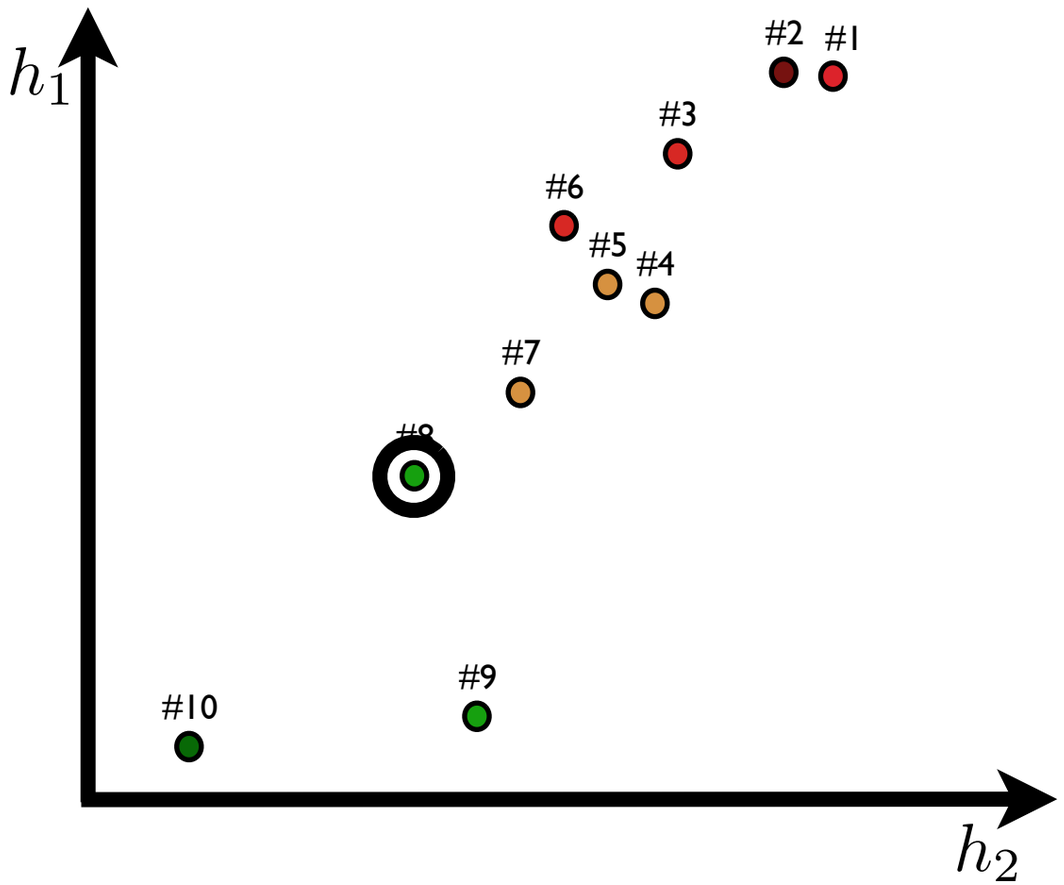
K-Best List Example

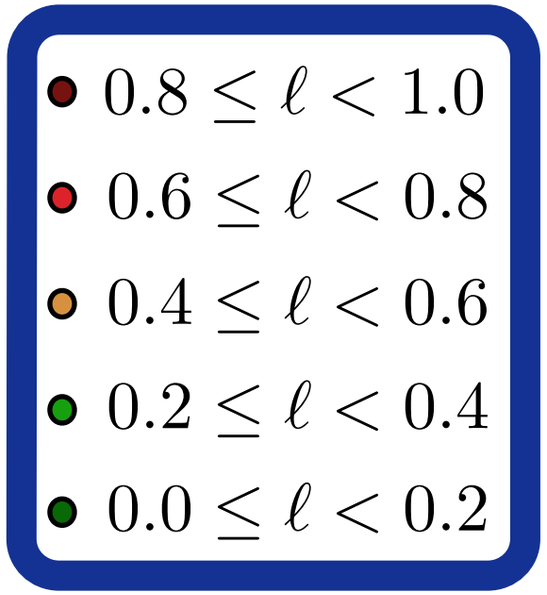


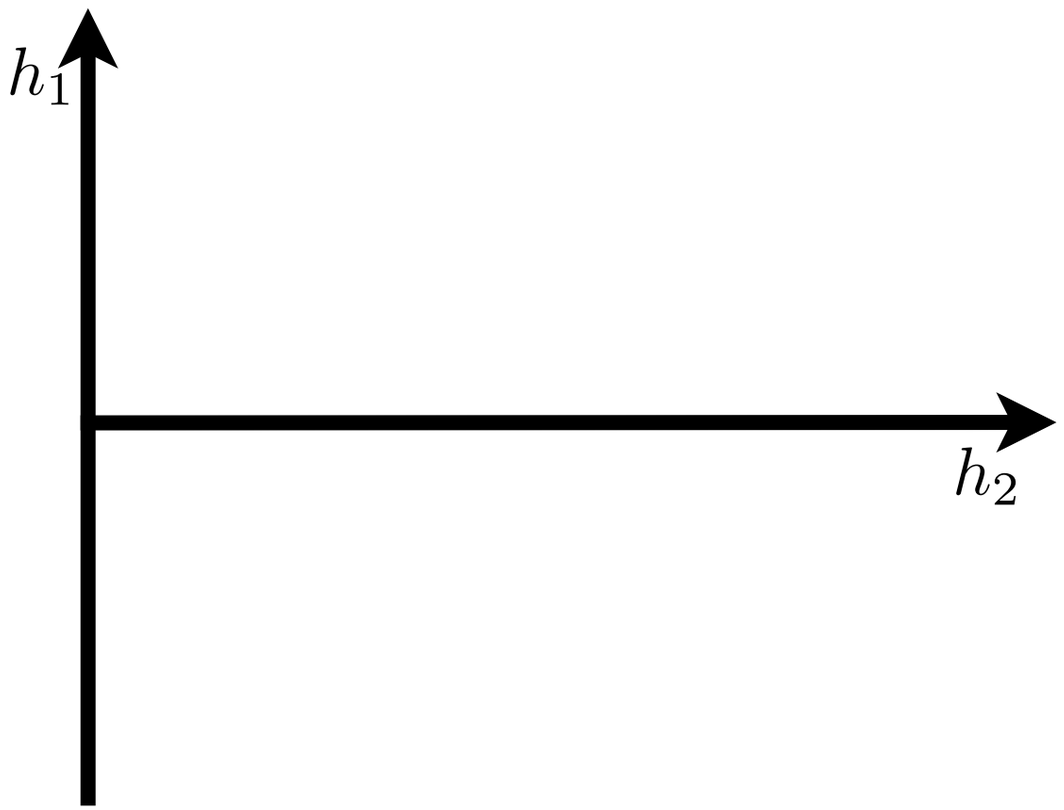
K-Best List Example

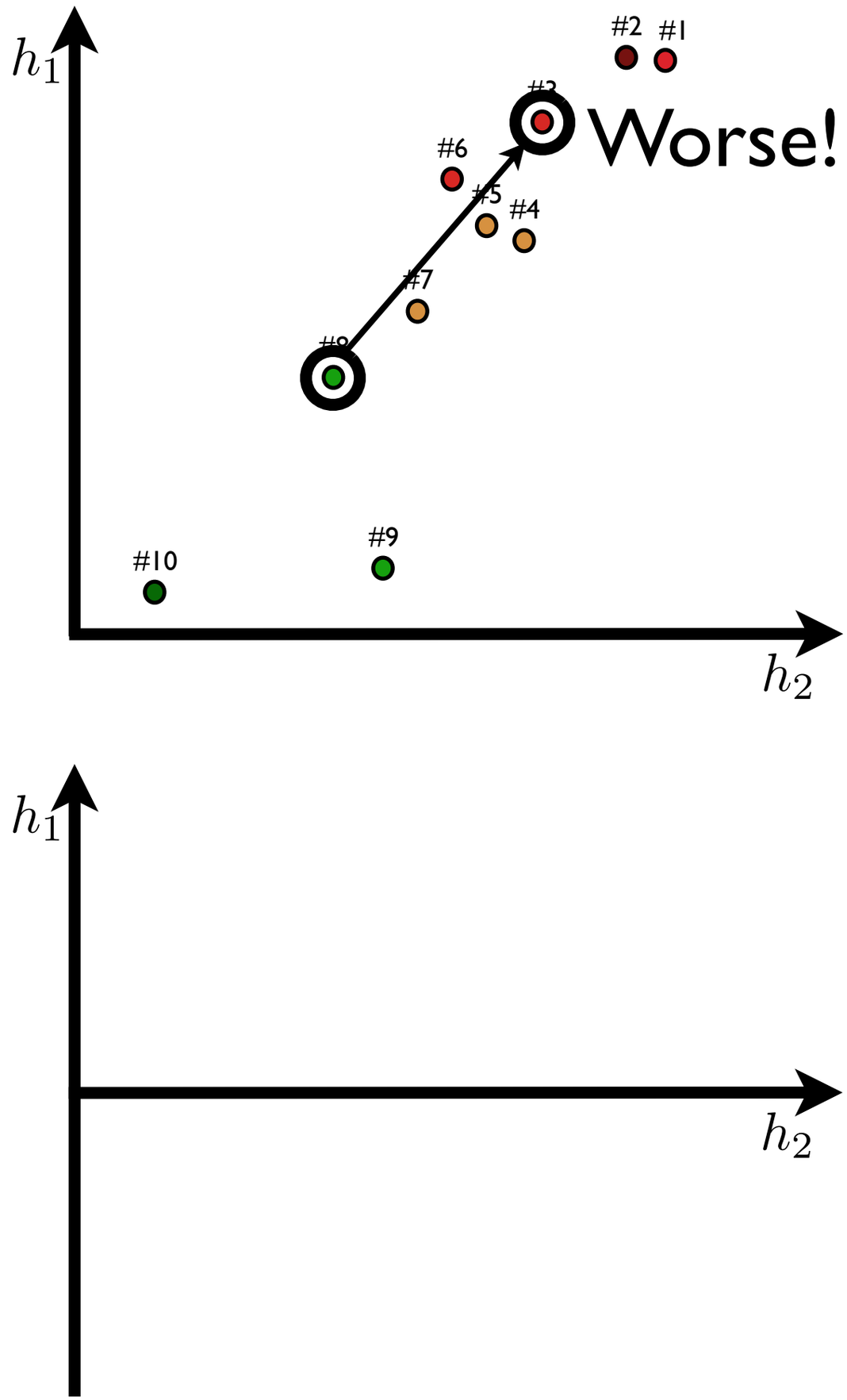


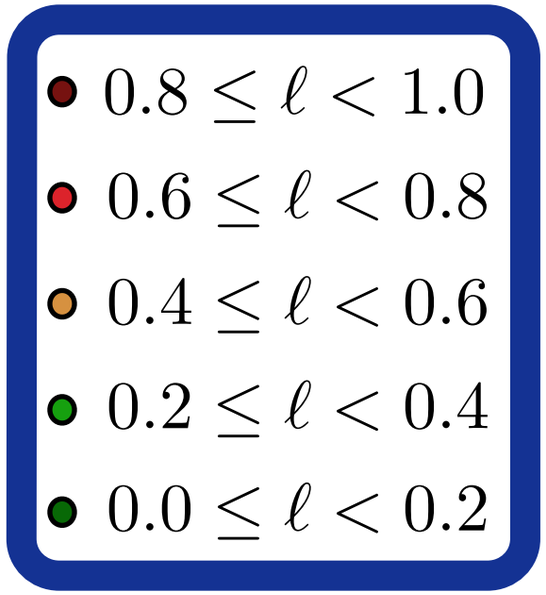
- $0.8 \leq \ell < 1.0$
- $0.6 \leq \ell < 0.8$
- $0.4 \leq \ell < 0.6$
- $0.2 \leq \ell < 0.4$
- $0.0 \leq \ell < 0.2$


h_1
h_2
#1
#2
#3
#4
#5
#6
#7
#8
#9
#10

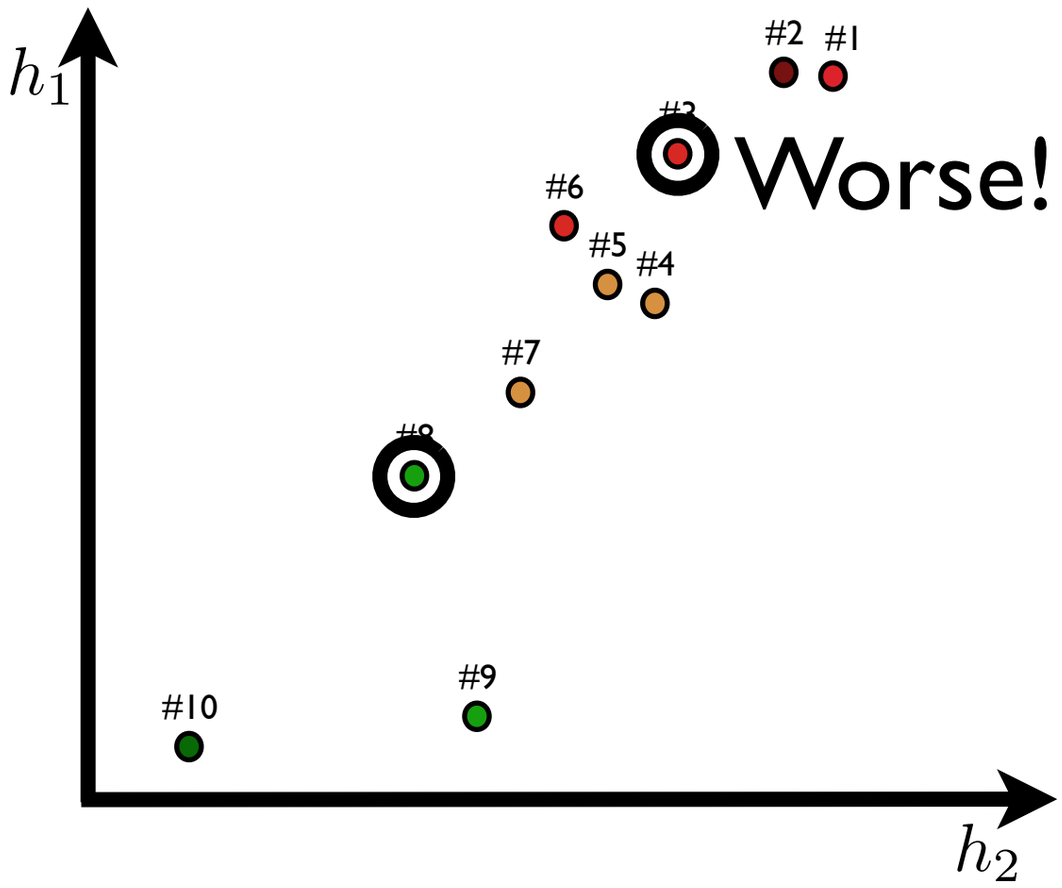

$0.8 \leq \ell < 1.0$
$0.6 \leq \ell < 0.8$
$0.4 \leq \ell < 0.6$
$0.2 \leq \ell < 0.4$
$0.0 \leq \ell < 0.2$


h_1
h_2

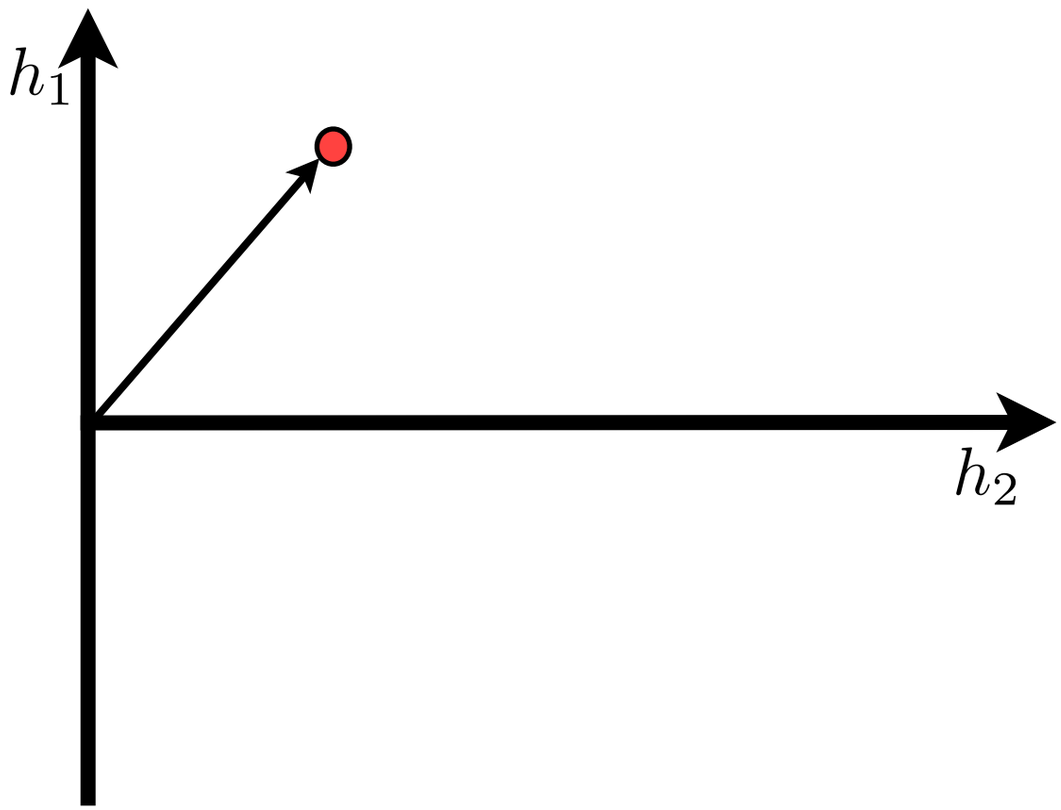

h_1
h_2
#1
#2
#3
#4
#5
#6
#7
#8
#9
#10
Worse!

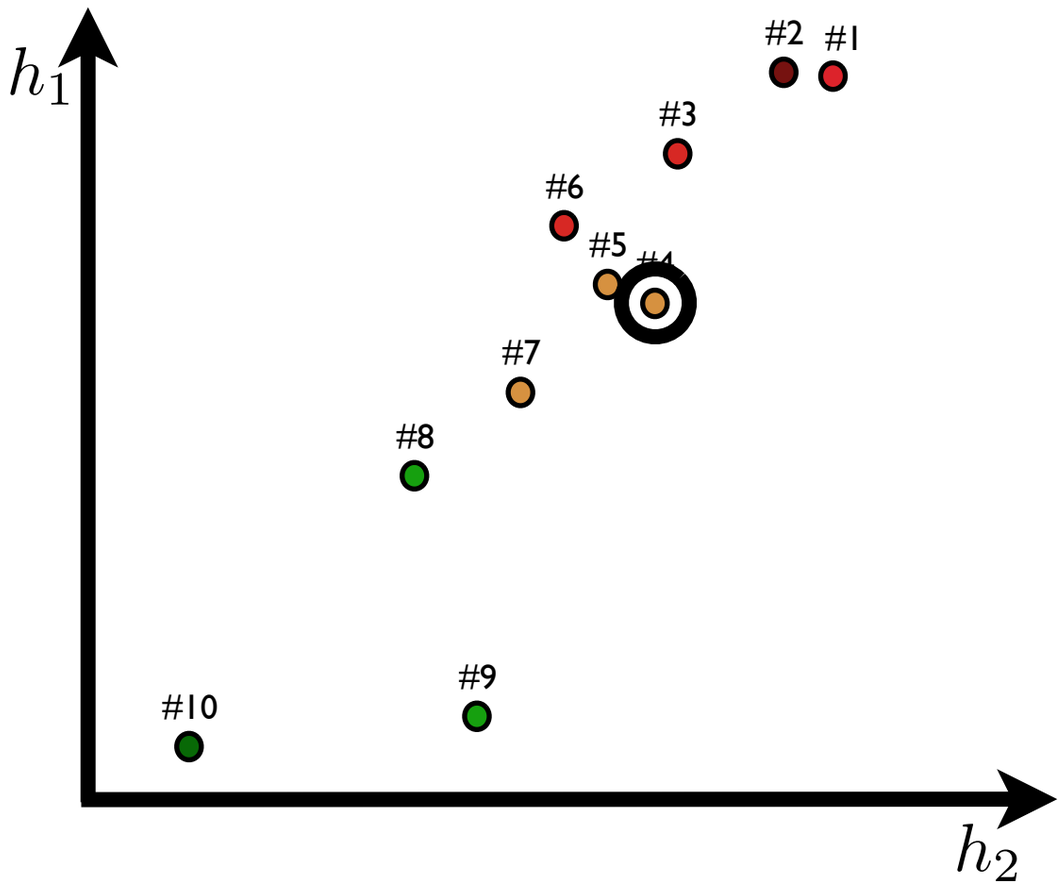

$0.8 \leq \ell < 1.0$
$0.6 \leq \ell < 0.8$
$0.4 \leq \ell < 0.6$
$0.2 \leq \ell < 0.4$
$0.0 \leq \ell < 0.2$

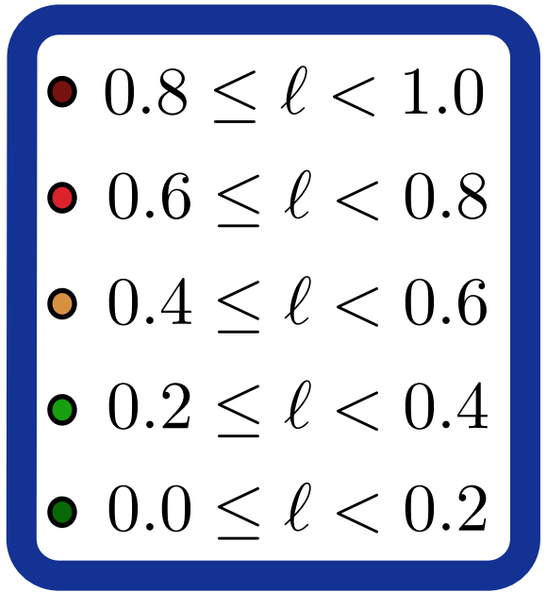
h_1
h_2

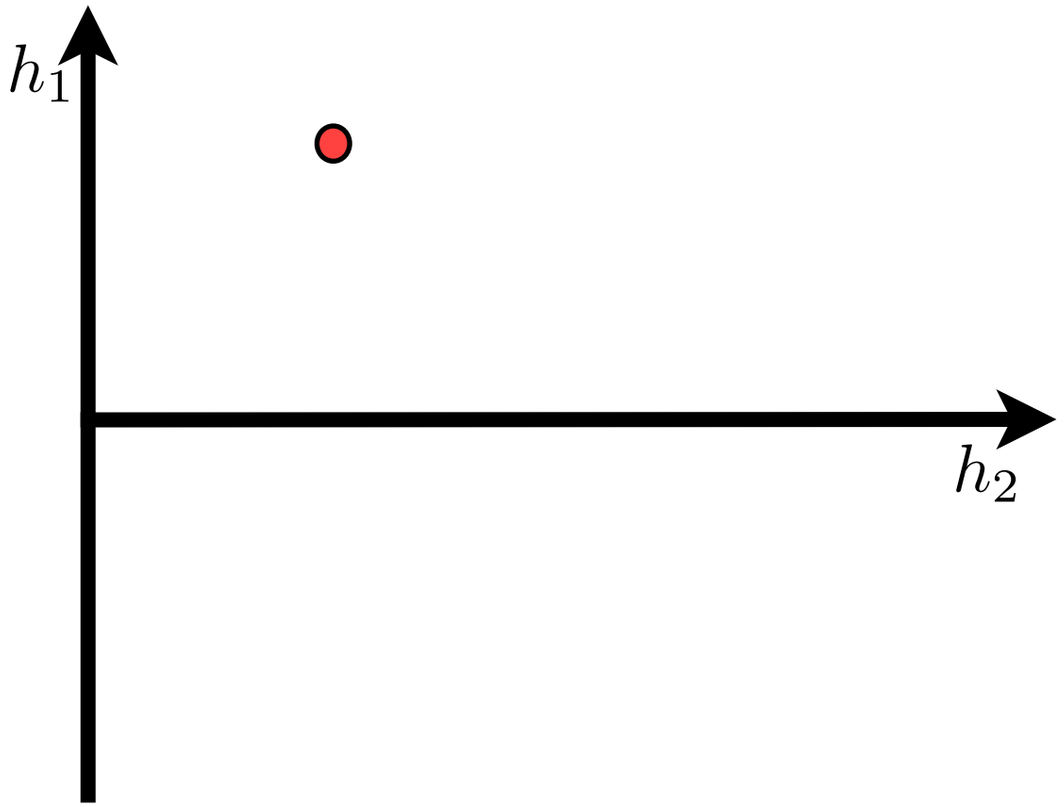


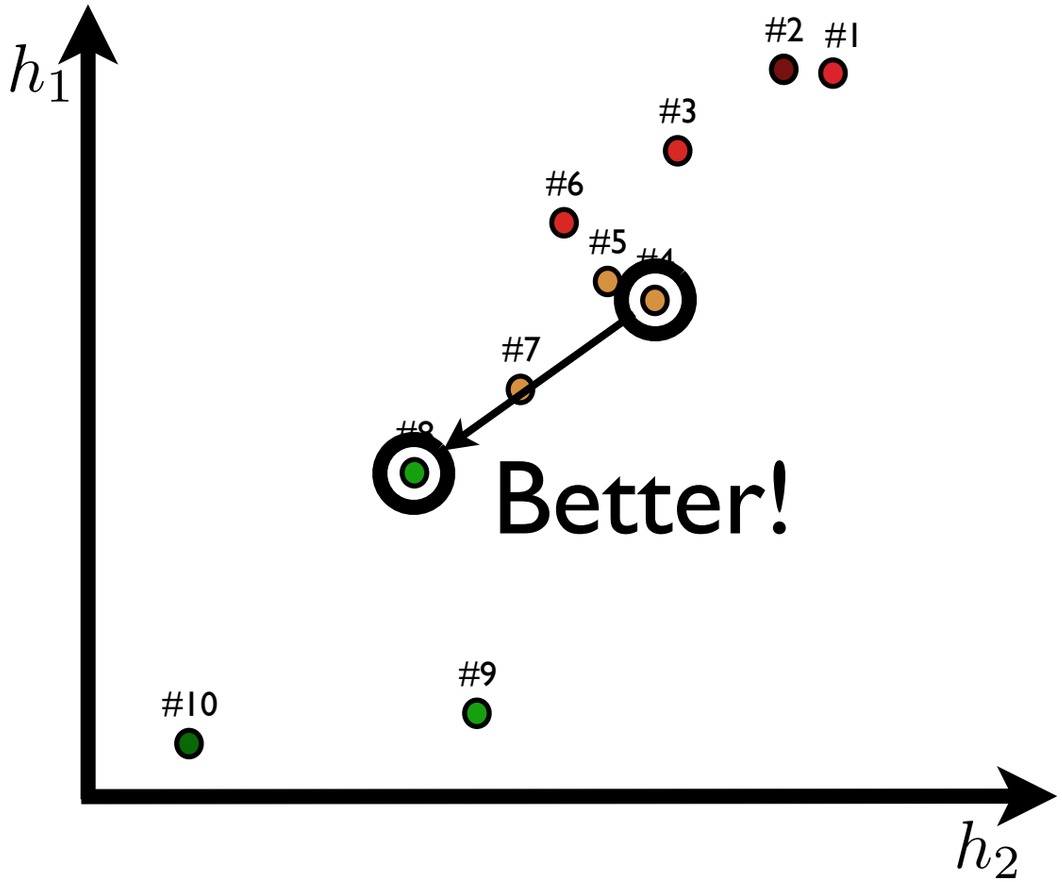
- $0.8 \leq \ell < 1.0$
- $0.6 \leq \ell < 0.8$
- $0.4 \leq \ell < 0.6$
- $0.2 \leq \ell < 0.4$
- $0.0 \leq \ell < 0.2$

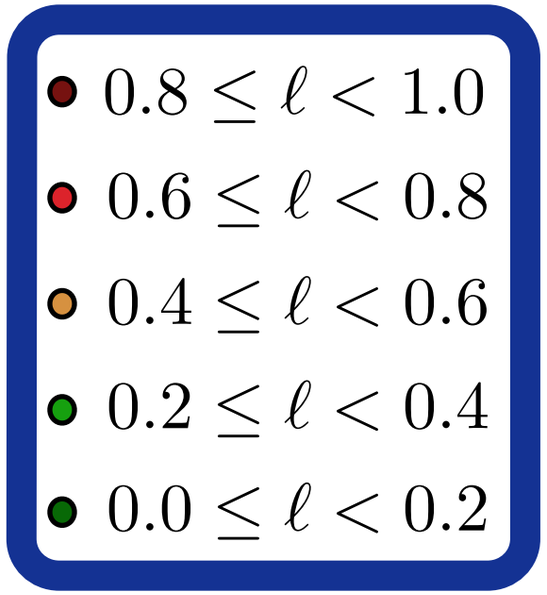


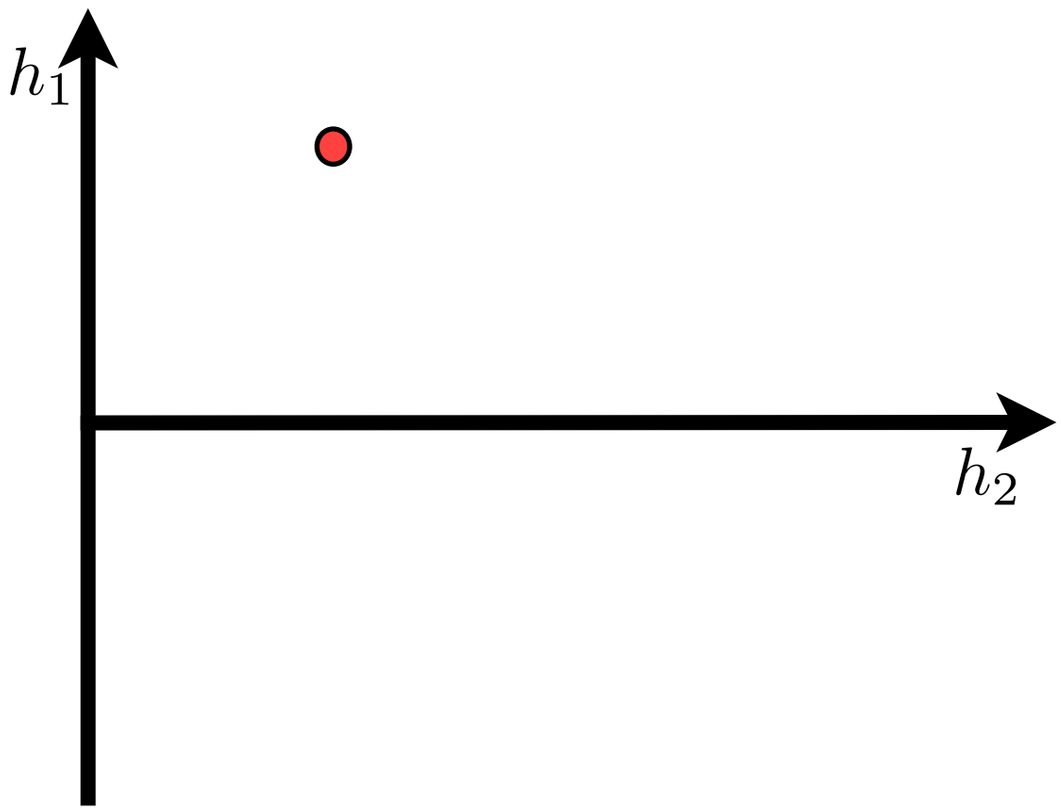

h_1
h_2
#1
#2
#3
#4
#5
#6
#7
#8
#9
#10

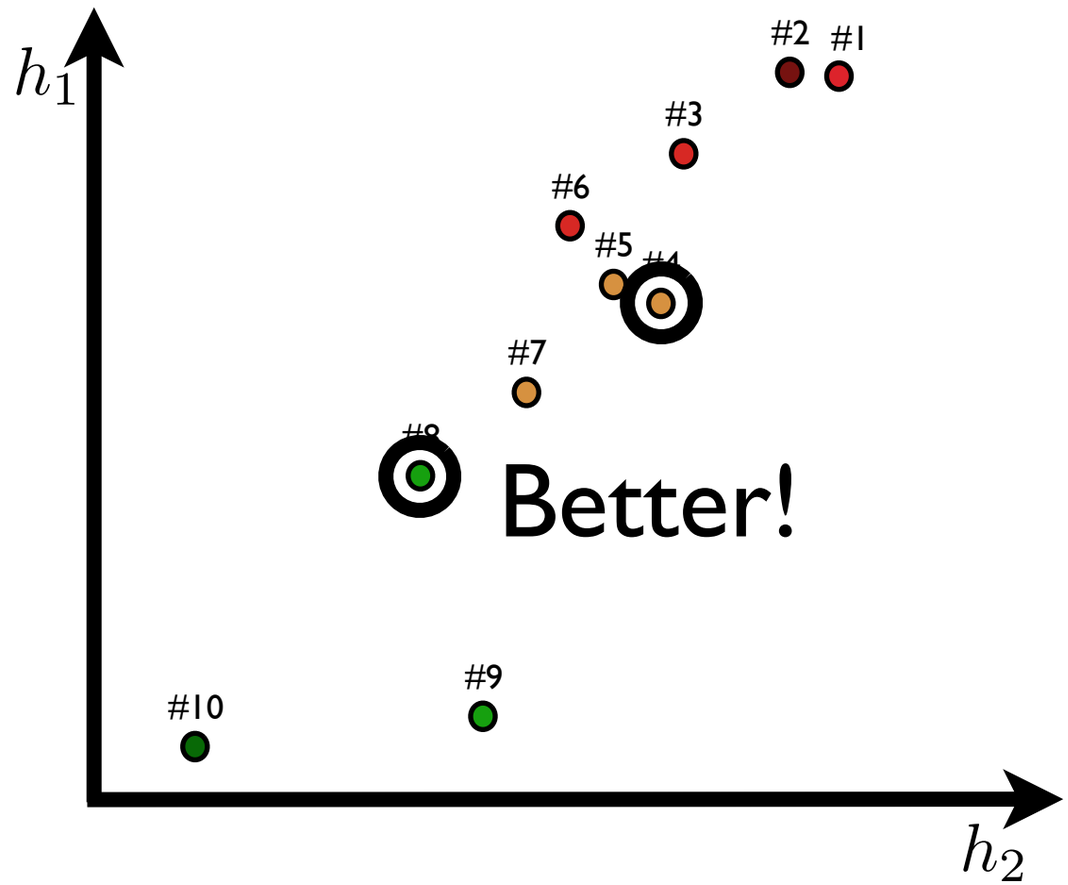

$0.8 \leq \ell < 1.0$
$0.6 \leq \ell < 0.8$
$0.4 \leq \ell < 0.6$
$0.2 \leq \ell < 0.4$
$0.0 \leq \ell < 0.2$


h_1
h_2


h_1
h_2
#1
#2
#3
#4
#5
#6
#7
#8
#9
#10
Better!


$0.8 \leq \ell < 1.0$
$0.6 \leq \ell < 0.8$
$0.4 \leq \ell < 0.6$
$0.2 \leq \ell < 0.4$
$0.0 \leq \ell < 0.2$


h_1
h_2



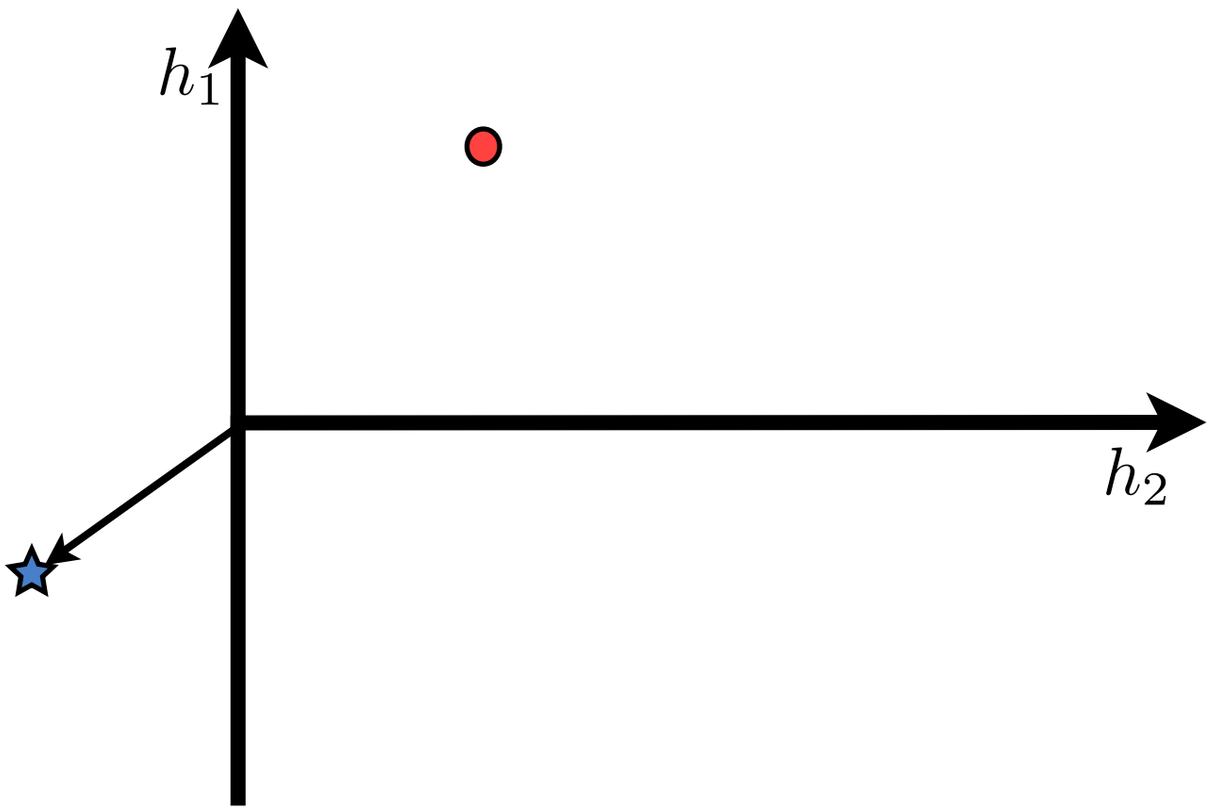
$0.8 \leq \ell < 1.0$

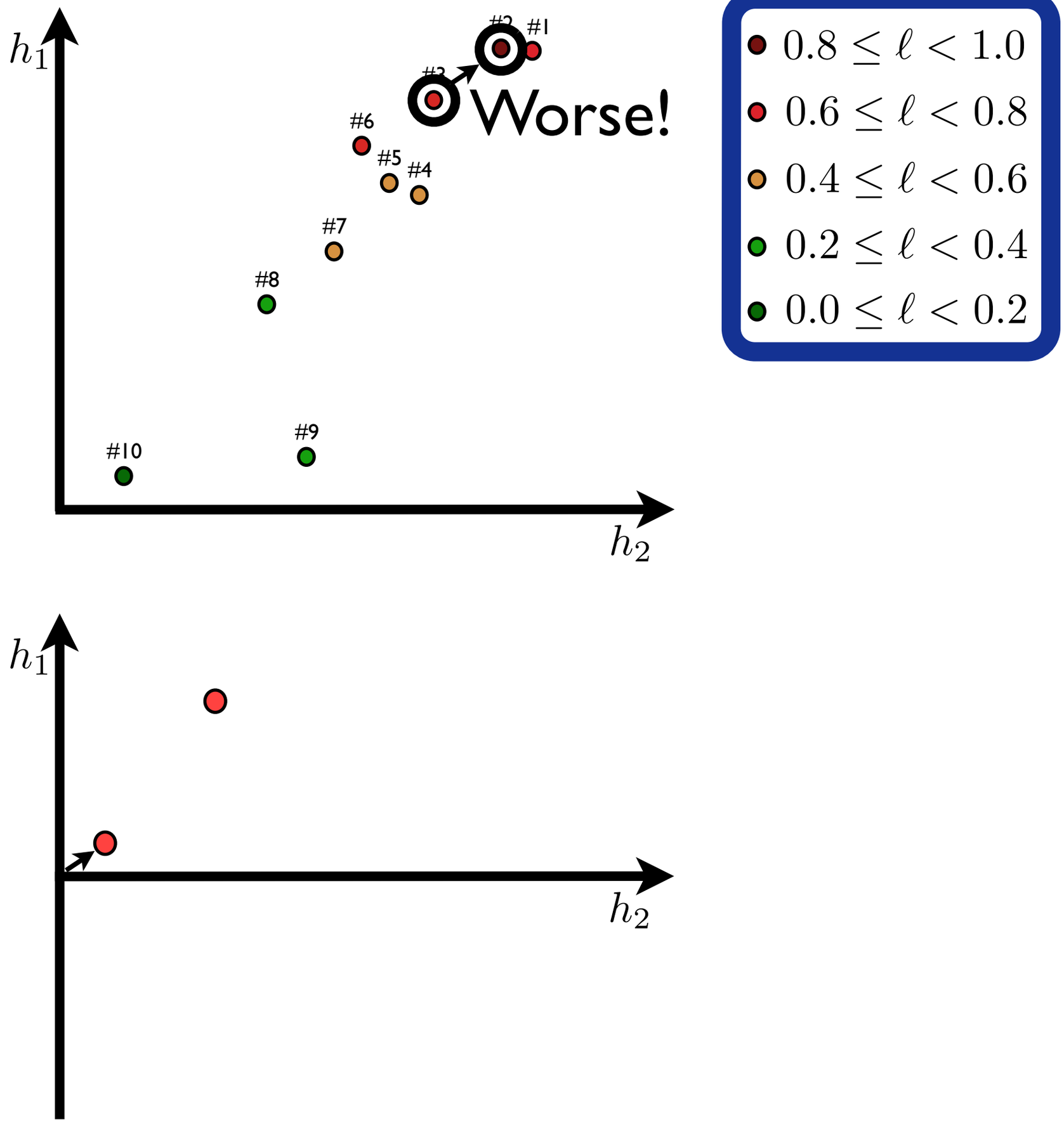
$0.6 \leq \ell < 0.8$

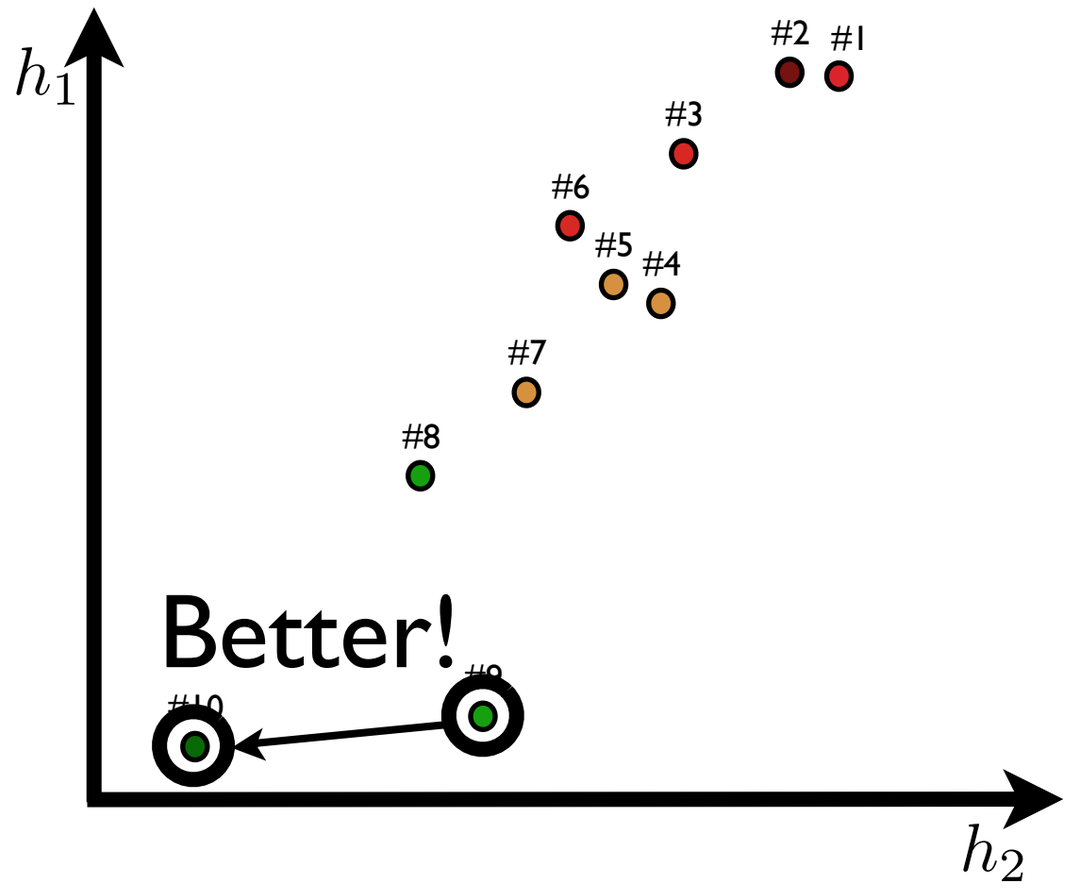
$0.4 \leq \ell < 0.6$

$0.2 \leq \ell < 0.4$

$0.0 \leq \ell < 0.2$




h_1
h_2
#1
#2
#3
#4
#5
#6
#7
#8
#9
#10
Worse!
$0.8 \leq \ell < 1.0$
$0.6 \leq \ell < 0.8$
$0.4 \leq \ell < 0.6$
$0.2 \leq \ell < 0.4$
$0.0 \leq \ell < 0.2$



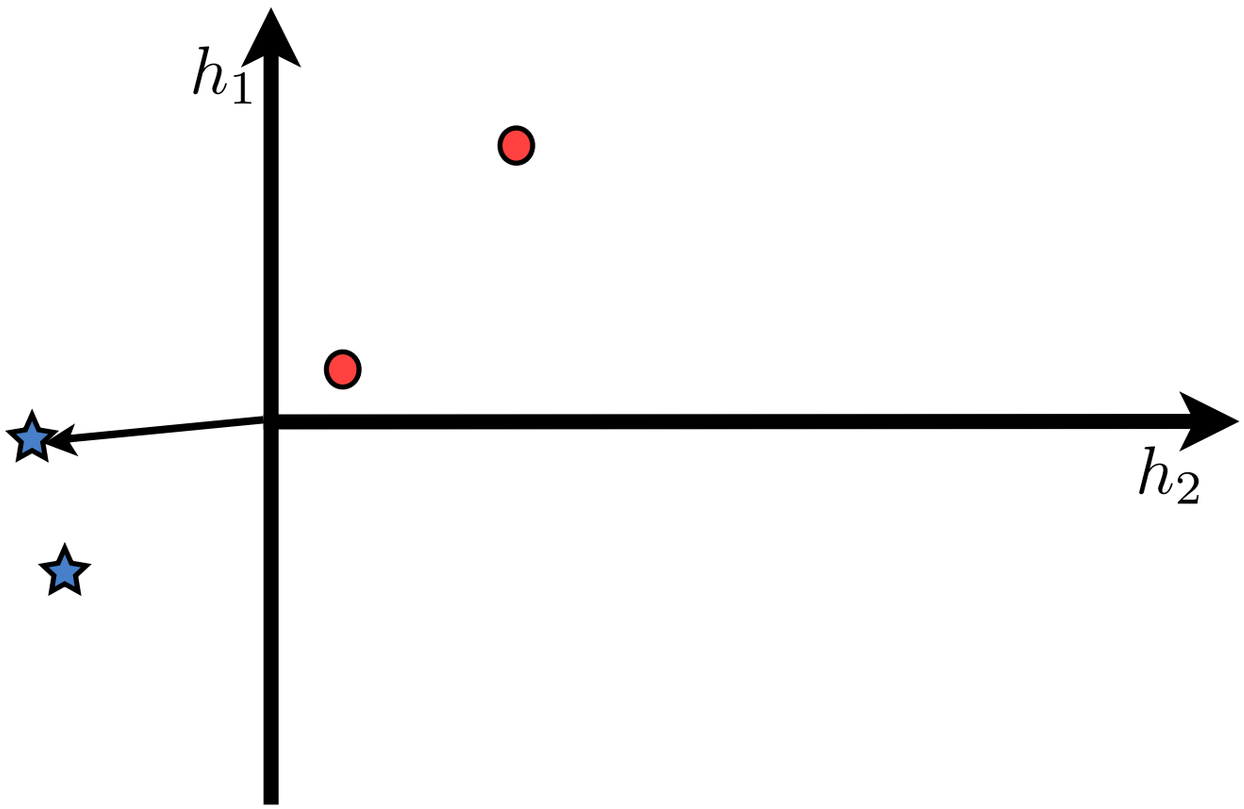
$0.8 \leq \ell < 1.0$

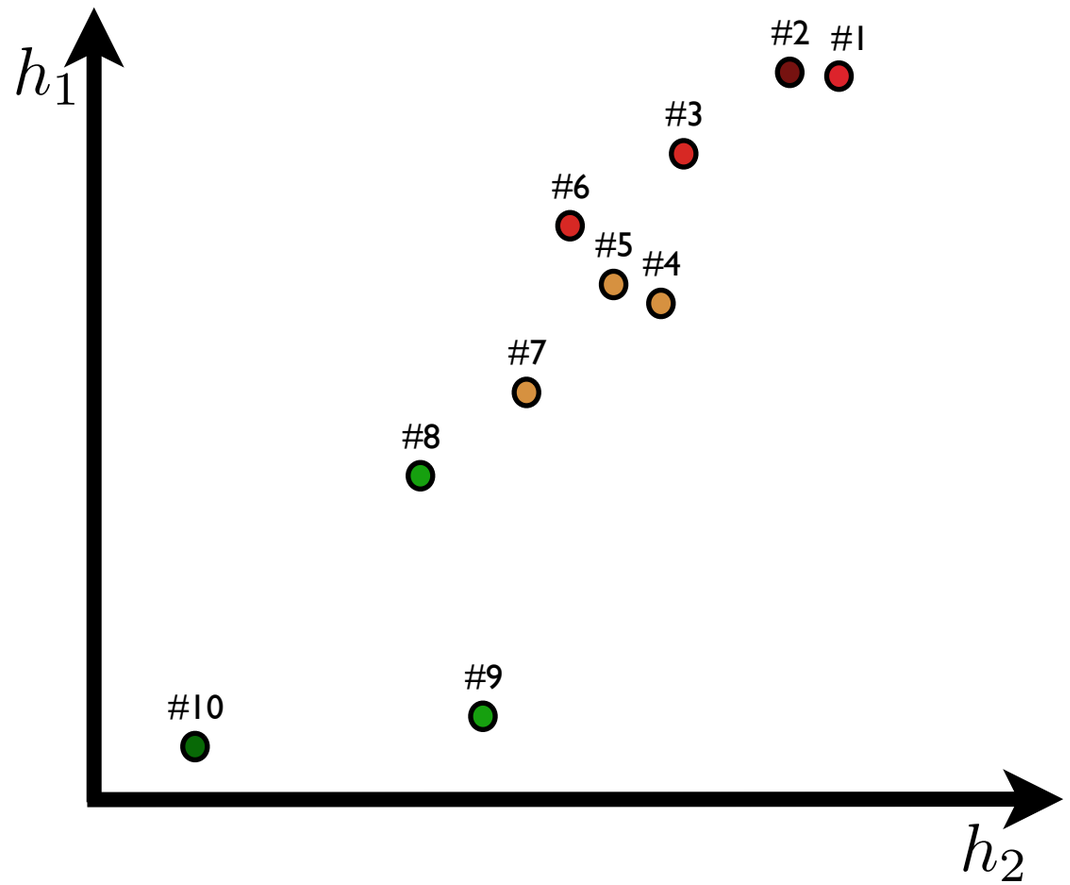
$0.6 \leq \ell < 0.8$

$0.4 \leq \ell < 0.6$

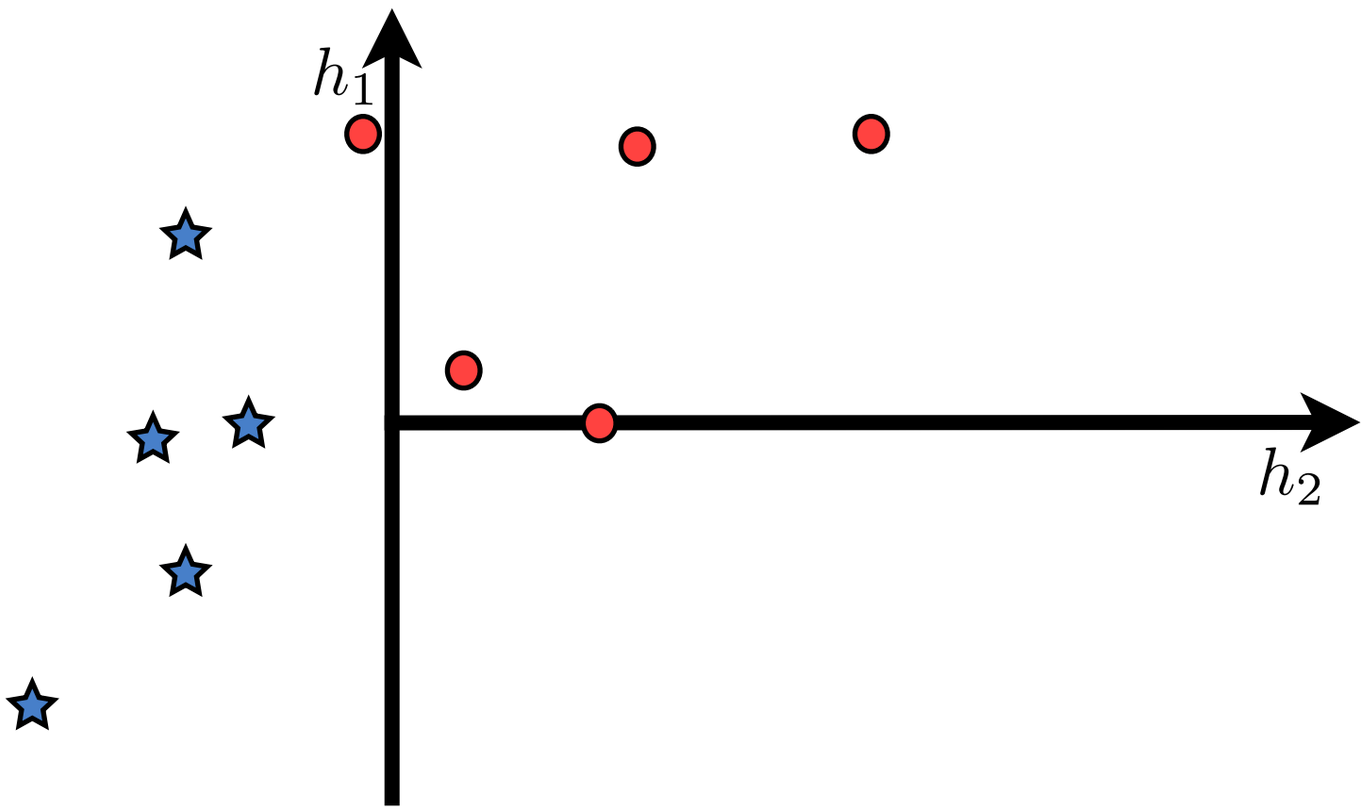
$0.2 \leq \ell < 0.4$

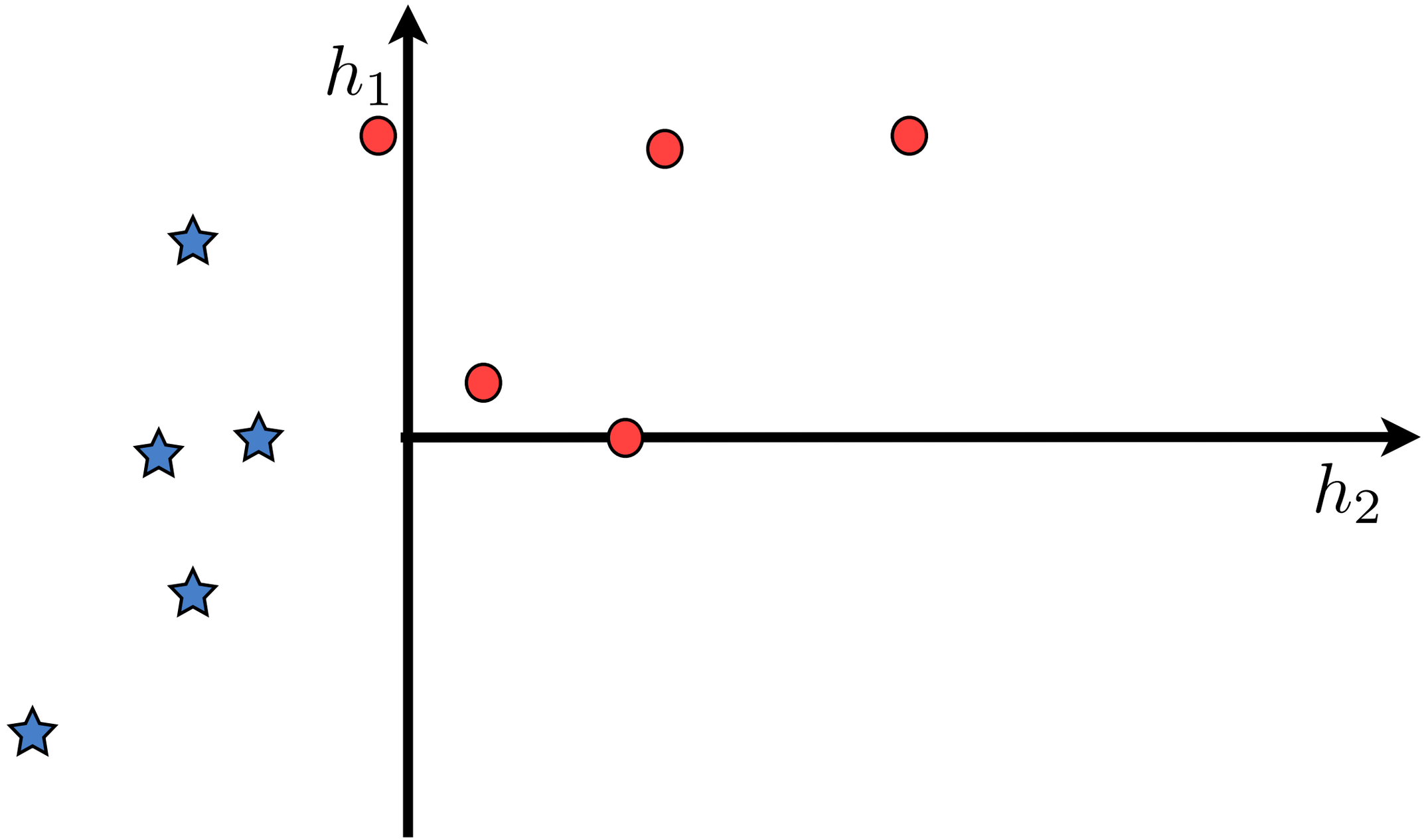
$0.0 \leq \ell < 0.2$



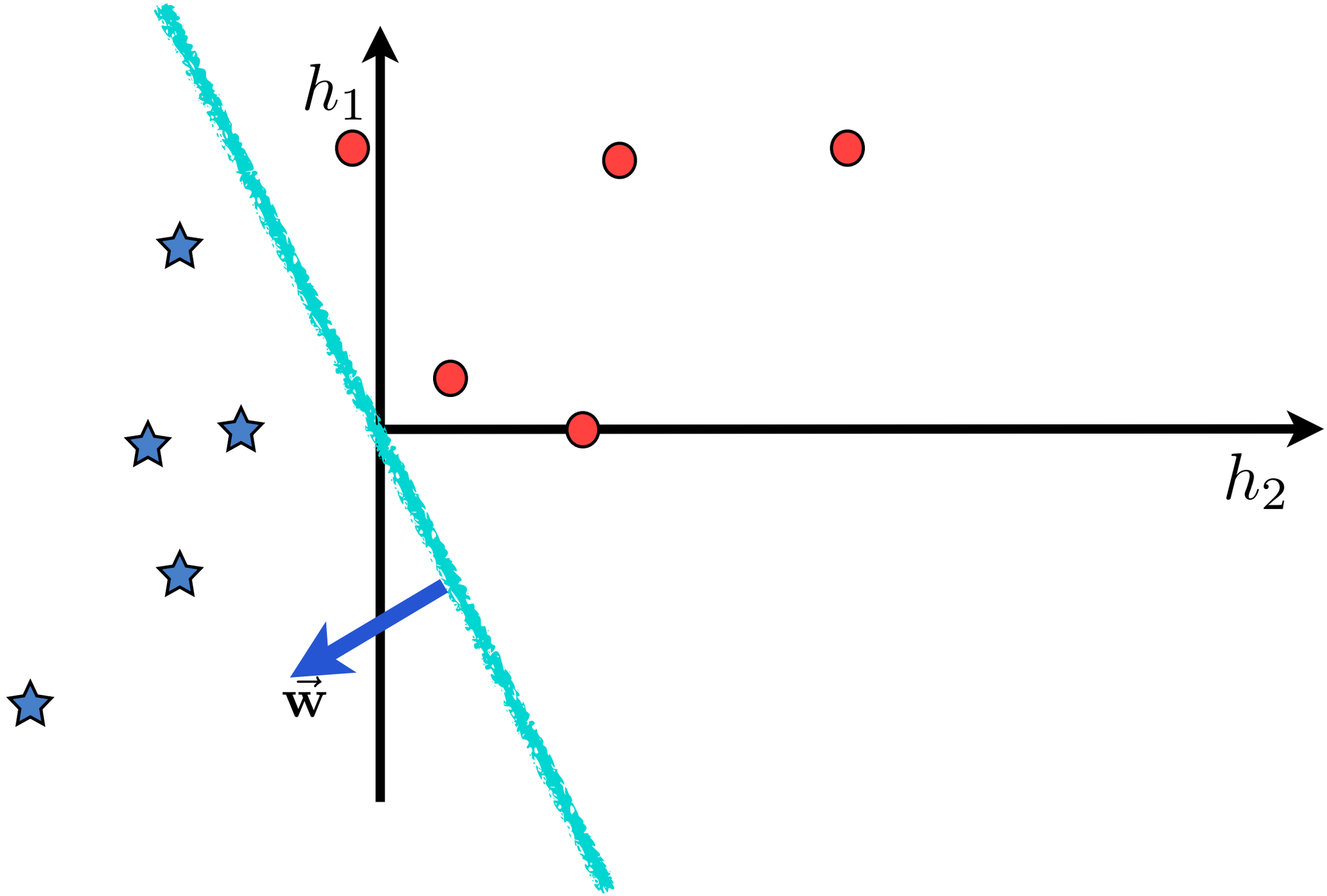


- $0.8 \leq \ell < 1.0$
- $0.6 \leq \ell < 0.8$
- $0.4 \leq \ell < 0.6$
- $0.2 \leq \ell < 0.4$
- $0.0 \leq \ell < 0.2$



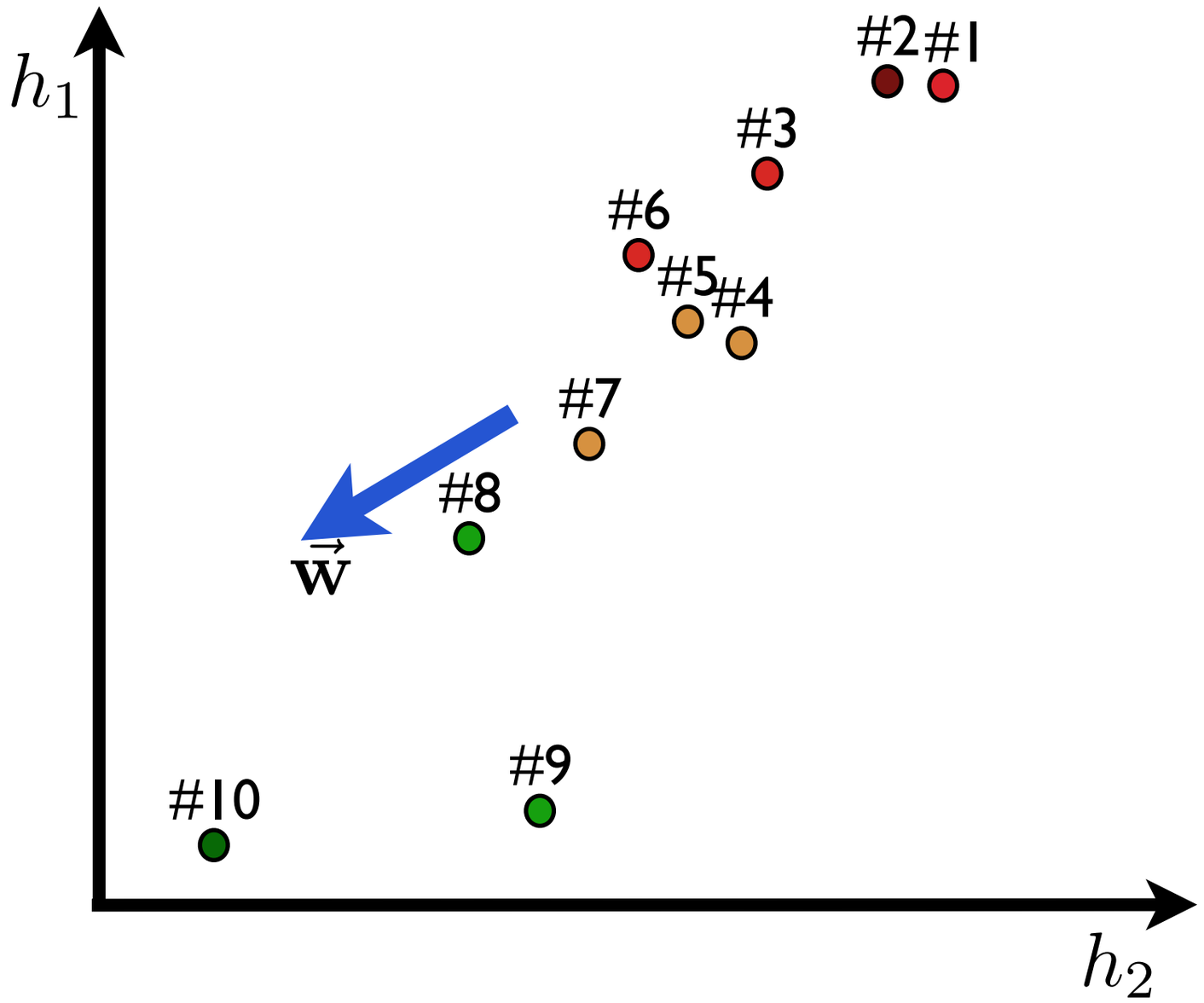


Fit a linear model



Fit a linear model

K-Best List Example



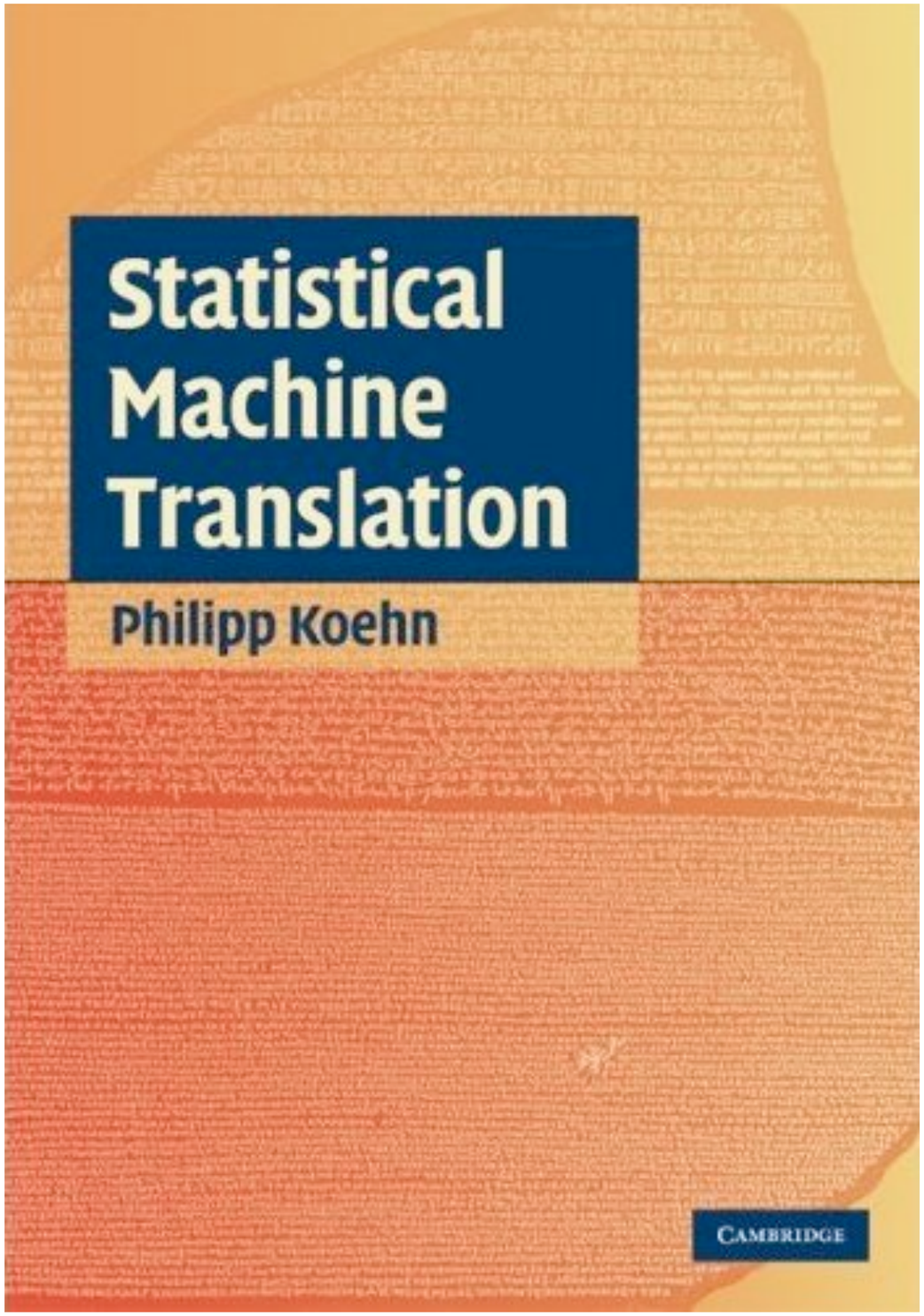
- $0.8 \leq \ell < 1.0$
- $0.6 \leq \ell < 0.8$
- $0.4 \leq \ell < 0.6$
- $0.2 \leq \ell < 0.4$
- $0.0 \leq \ell < 0.2$

Summary

- Evaluation metrics
  - Figure out how well we're doing
  - Figure out if a feature helps
  - **Train your system**
- What's a great way to improve translation?
  - **Improve evaluation!**

Reading

- Read 9 from the textbook



Announcements

- HW3 due **tonight** at 11:59pm
- 1st part of the term project is due by the end of Spring break
- Jonny will post HW4 soon - start early
- Language presentations after the break:
  - Tuesday: **Alyssa** - Yiddish
  - Thursday: **Fan** - German