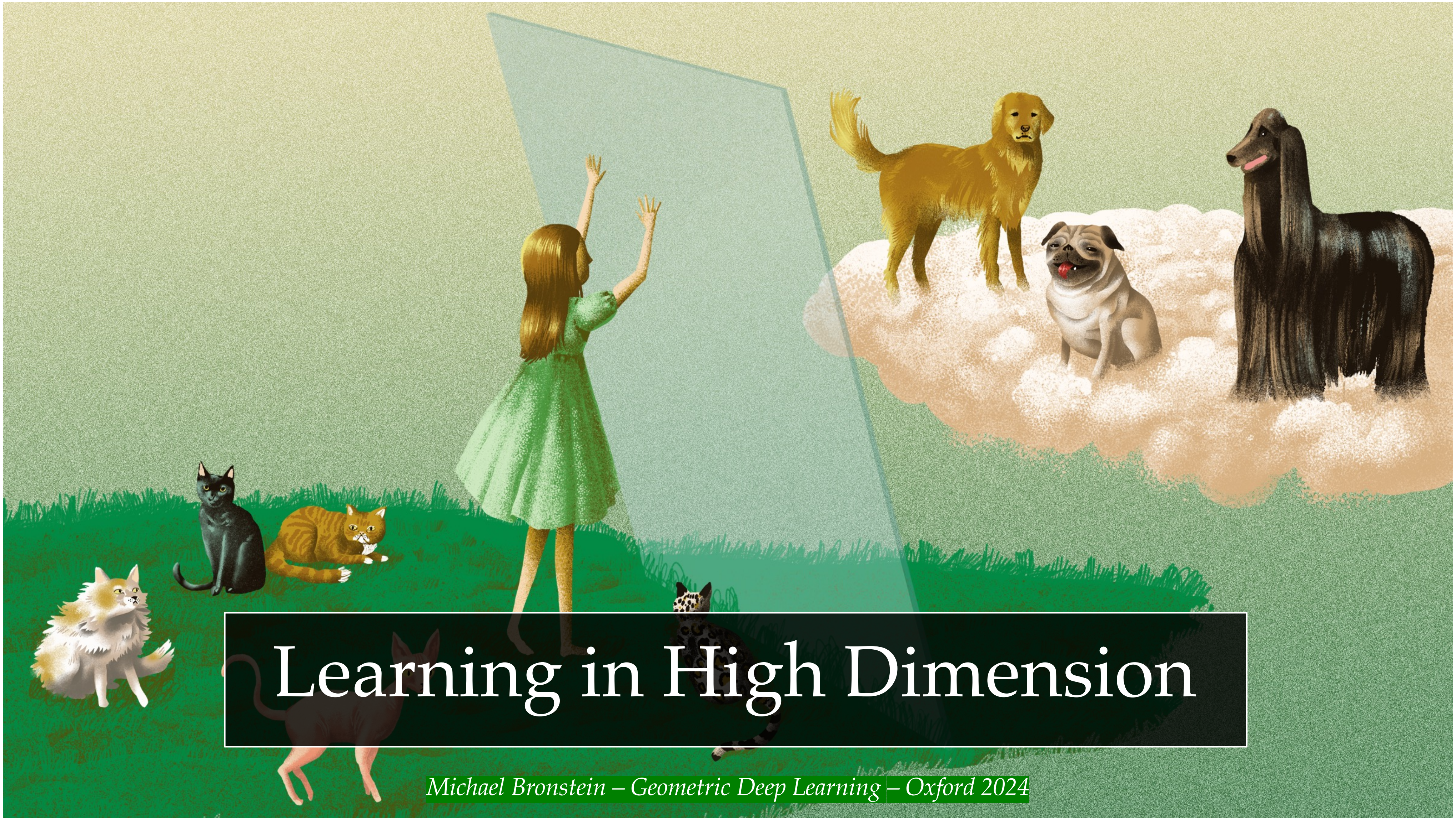


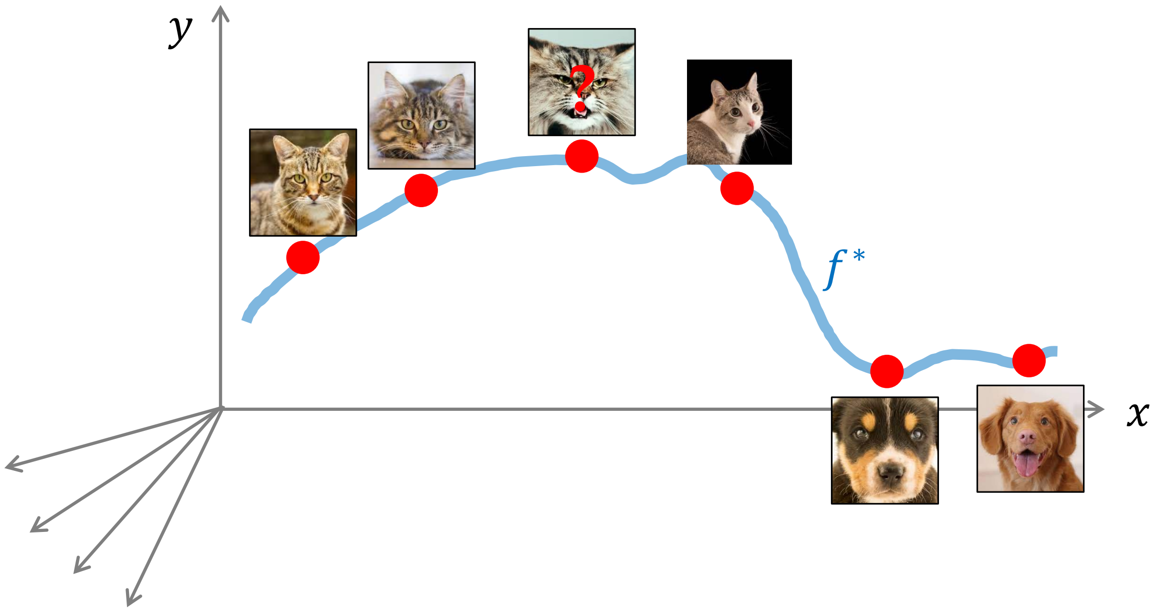
Learning in High Dimension
Michael Bronstein – Geometric Deep Learning – Oxford 2024

Outline

- Supervised machine learning has three sources of error: *approximation*, *estimation*, and *optimisation*
- Dealing with high-dimensional inputs requires strong notions of *regularity*
- Standard function classes based on local/global continuity are *dimensionality-cursed*
- This will bring us to the need for a new *geometric* type of regularity, which is at the core of Geometric Deep Learning

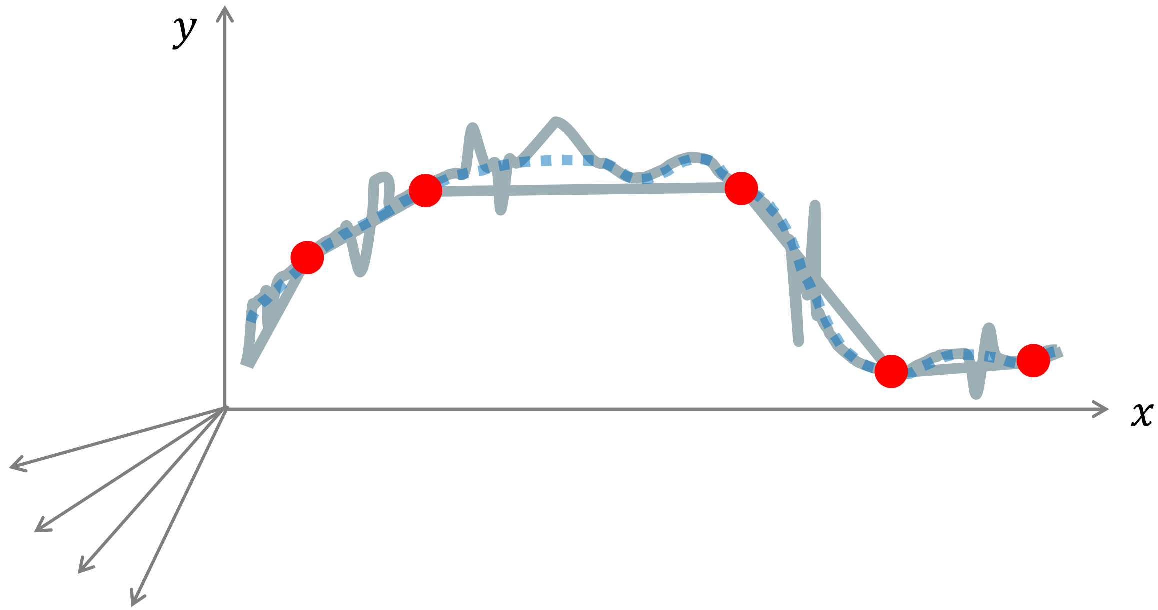

ML
black
box
{cat,dog}

(Supervised) Machine Learning = glorified curve fitting



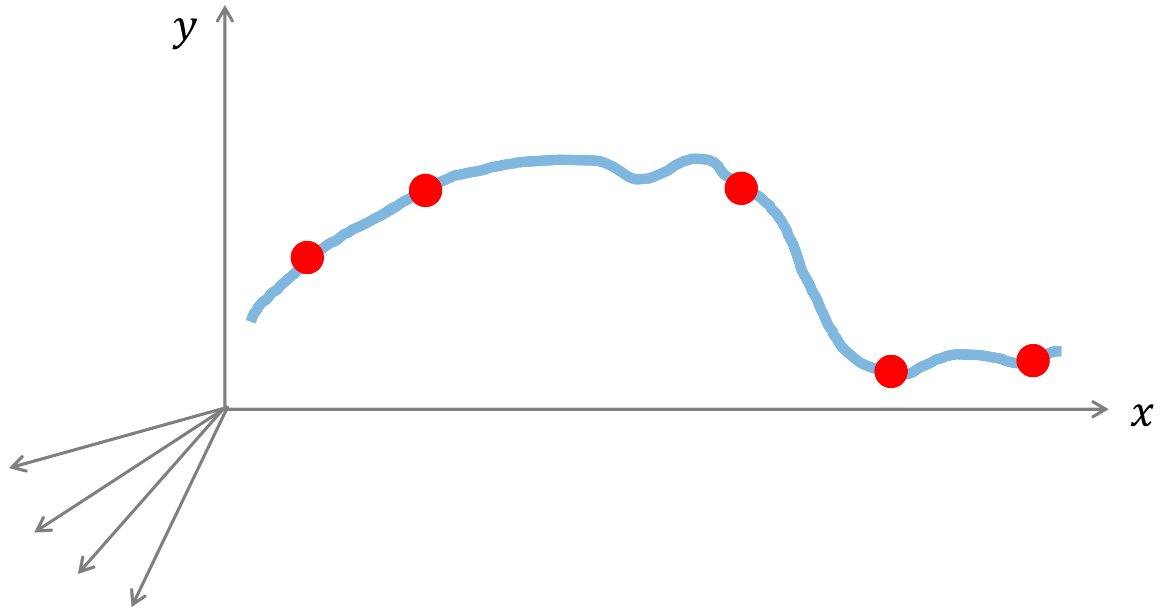
Given a set of observations $\{(x_i, y_i)\}_{i=1}^N$ of some function f^* ("*training set*") predict its values at previously unseen points

(Supervised) Machine Learning = glorified curve fitting



How the function looks like?

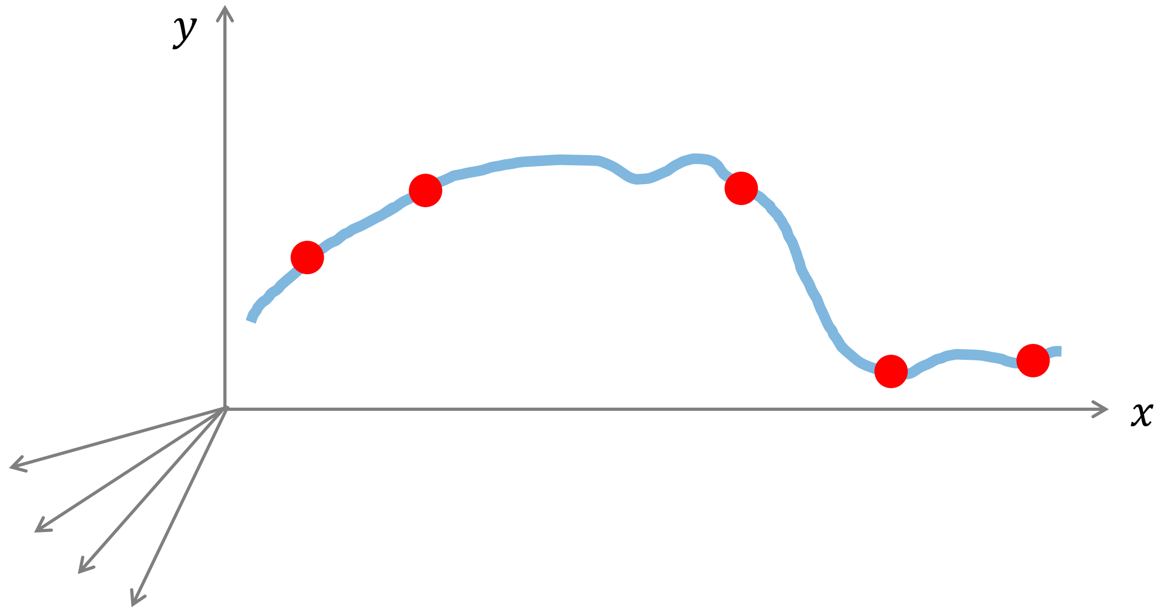
(Supervised) Machine Learning = glorified curve fitting



How the function looks like?

How the sampling looks like?

(Supervised) Machine Learning = glorified curve fitting



How the function looks like?

approximation

How the sampling looks like?

estimation

How to find the fitting?

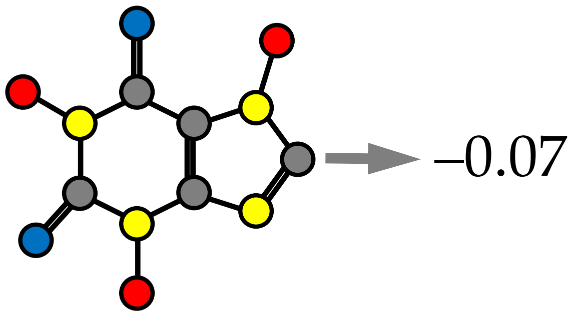
optimisation

BASICS OF STATISTICAL MACHINE LEARNING

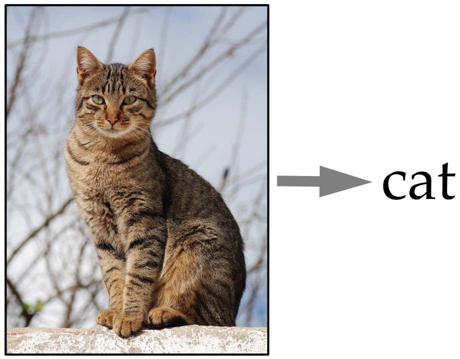
Problem setting

- **Inputs** $x_i \in \mathcal{X}$ typically high dimensional (e.g. $\mathcal{X} = \mathbb{R}^d$, $d \gg 1$)
- **Labels** $y_i \in \mathcal{Y}$
  - *Regression:* $\mathcal{Y} = \mathbb{R}$
  - *Classification:* $\mathcal{Y} = \{1, \ldots, K\}$
  - *Structured prediction:* $\mathcal{Y} = \mathcal{X}$

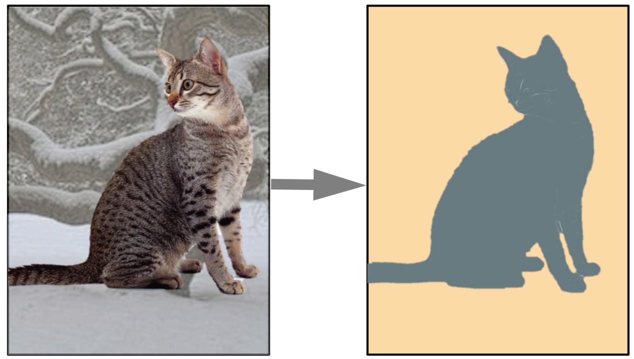
Examples of Supervised Learning problems



Regression
(solubility log$P$)



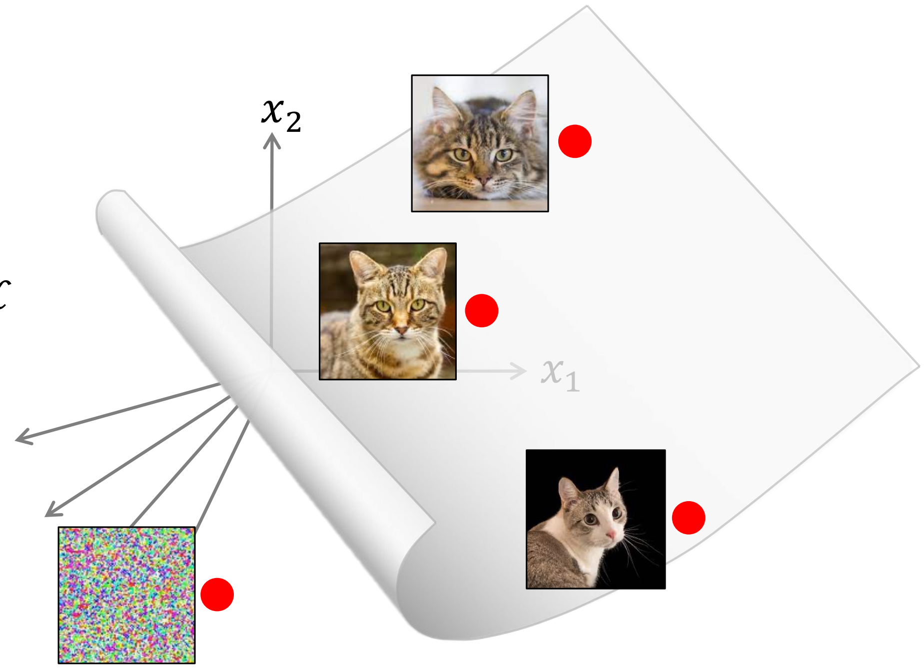
Classification
(binary: cat/dog)



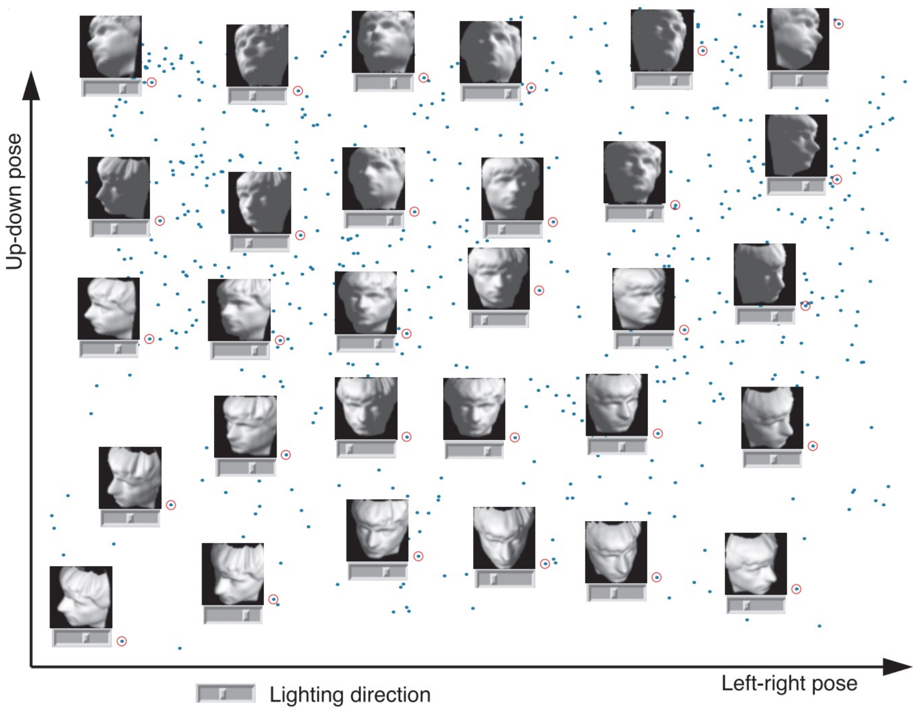
Structured prediction
(image segmentation)

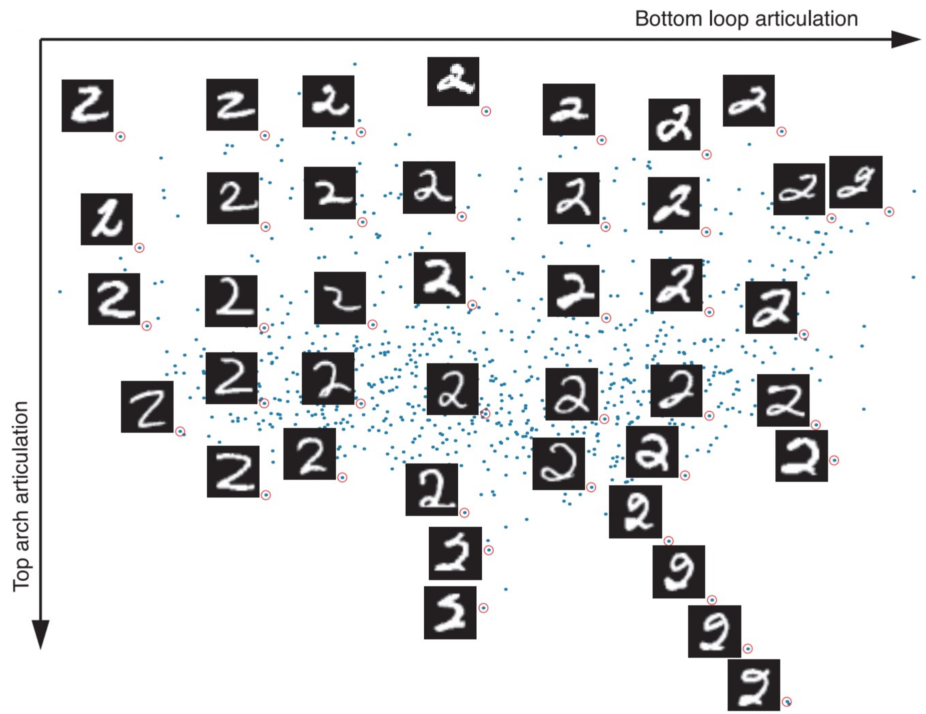
Data

- **Data distribution** $P(x, y)$
  - Distribution P is *unknown* during learning
  - Samples assumed to be drawn *i.i.d.*
  - Often forms a low-dimensional structure in $\mathcal{X}$ (*“manifold assumption”*)



Manifold Assumption





Tenenbaum, De Silva, Langford 2000

Error metric

- **Loss function** $\ell: \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ satisfying $\ell(y, y') \geq 0$ and $\ell(y, y) = 0$
  - *Classification loss:* $\ell(y, y') \geq 1_{y \neq y'}$
  - *Regression loss:* $\ell(y, y') = \|y - y'\|^2$
- Given a function $f: \mathcal{X} \to \mathcal{Y}$, the loss $\ell(f(x), y)$ is a *random variable*

Error metric

- **Loss function** $\ell: \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ satisfying $\ell(y, y') \geq 0$ and $\ell(y, y) = 0$
- **Population risk** (or **error**) of f

$$\mathcal{R}(f) = \mathbb{E}\ell(f(x), y) = \int_{\mathcal{X}\times\mathcal{Y}} \ell(f(x), y)\mathrm{d}P(x, y)$$

$$= \mathbb{E}_x \ \mathbb{E}_{y|x}[\ell(f(x), y)|x] = \int_{\mathcal{X}} \int_{\mathcal{Y}} \ell(f(x), y)\mathrm{d}P_{y|x}(y) \ \mathrm{d}P_x(x)$$

Conditioning on x

Error metric

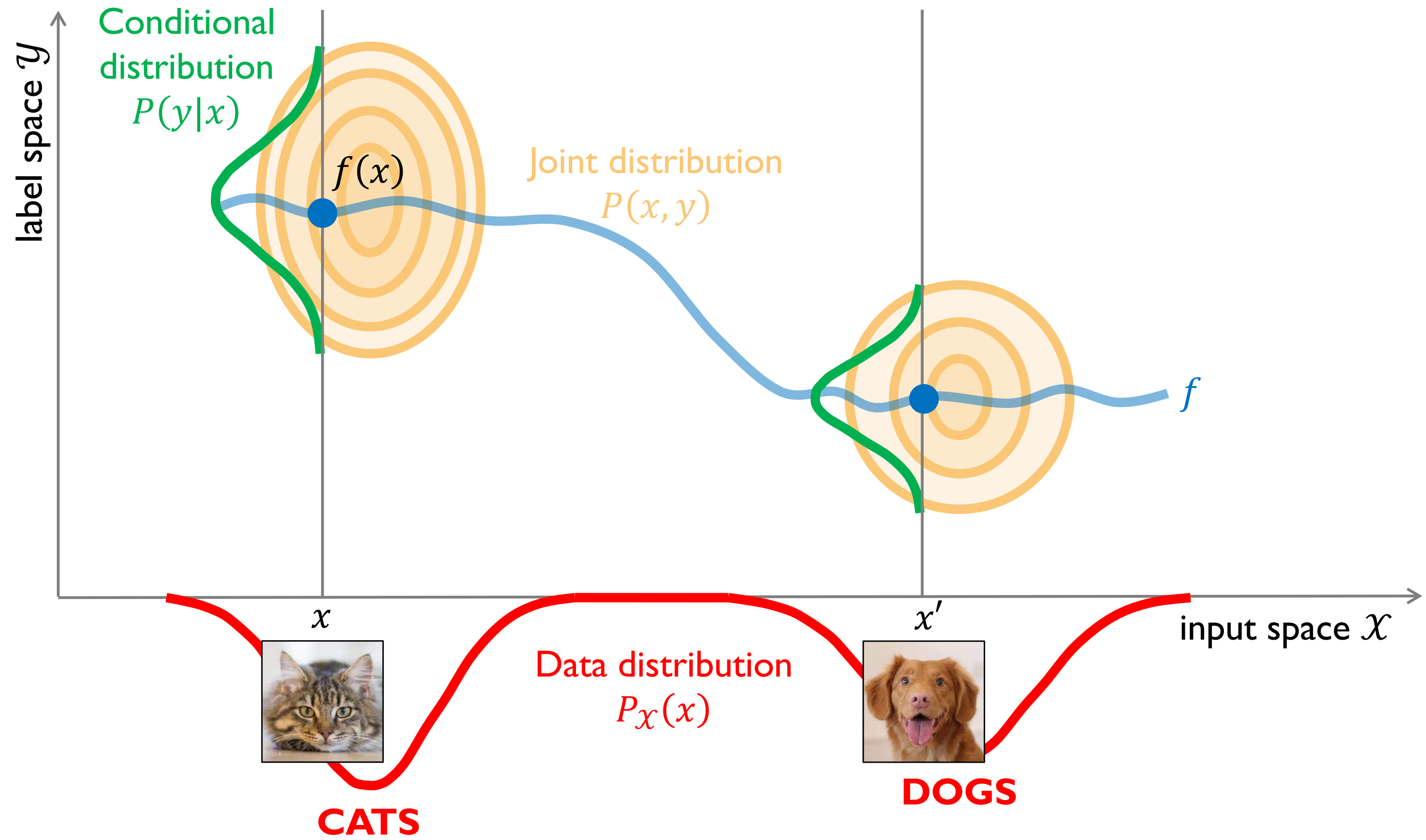
- **Loss function** $\ell\colon \mathcal{Y}\times\mathcal{Y} \to \mathbb{R}$ satisfying $\ell(y, y') \geq 0$ and $\ell(y, y) = 0$
- **Population risk** (or **error**) of f

$$\mathcal{R}(f) = \mathbb{E}\ell(f(x), y) = \mathbb{E}_x \mathbb{E}_{y|x}[\ell(f(x), y)|x]$$

- **Bayes optimal estimator** minimises the error *point-wise*

$$f^*(x) = \operatorname*{argmin}_{z\in\mathcal{Y}} \mathbb{E}_{y|x}[\ell(z, y)|x]$$

  - Defined via distribution P, which is *unknown in practice*
  - f^* may be *arbitrarily complex*



Conditional
distribution
$P(y|x)$
label space $\mathcal{Y}$
$f(x)$
Joint distribution
$P(x, y)$
f
x
x'
input space $\mathcal{X}$
Data distribution
$P_{\mathcal{X}}(x)$
CATS
DOGS

Model class

- **Hypothesis** (or **model**) **class** is a subset of functions $\mathcal{F} = \{f_\theta : \mathcal{X} \to \mathcal{Y} : \theta \in \Theta\}$
  - *Polynomials of degree k*: $f_\theta(x) = \sum_{i=0}^{k} \theta_i x^i$
  - *Neural networks* of certain type (with θ being layer weights)

Model class

- **Hypothesis** (or **model**) **class** is a subset of functions $\mathcal{F} = \{f_\theta : \mathcal{X} \to \mathcal{Y} : \theta \in \Theta\}$
- **Model complexity** (or **capacity**) is some non-negative function $\gamma : \Theta \to \mathbb{R}$ allowing to order the functions in $\mathcal{F}$ according to their "complexity"
  - *Weight decay* $\gamma(\theta) = \|\theta\|_p^p$ in a linear model $f_\theta(x) = \langle \theta, x \rangle$
  - *Number of neurons* in a neural network
  - *Sobolev norm* $\gamma(\theta) = \int_{\mathbb{R}} (1 + \omega^2)^s \left|\hat{f}_\theta(\omega)\right|^2 \mathrm{d}\omega$

Sobolev space $H^s = W^{s,2}$ is a generalization of the Lebesgue space L_2 (square-integrable functions) accounting for the function's derivatives

Note: confusingly, $\hat{f}$ here denotes the Fourier transform of f

Model class

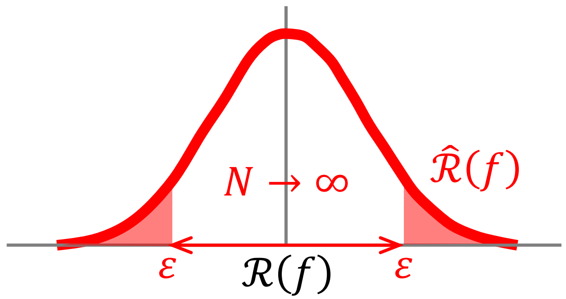
- **Hypothesis** (or **model**) **class** is a subset of functions $\mathcal{F} = \{f_\theta : \mathcal{X} \to \mathcal{Y} \,:\, \theta \in \Theta\}$
- **Model complexity** (or **capacity**) is some non-negative function $\gamma : \Theta \to \mathbb{R}$ allowing to order the functions in $\mathcal{F}$ according to their "complexity"
  - *Weight decay* $\gamma(\theta) = \|\theta\|_p^p$ in a linear model $f_\theta(x) = \langle \theta, x \rangle$
  - *Number of neurons* in a neural network
  - *Sobolev norm* $\gamma(\theta) = \int_{\mathbb{R}} (1 + \omega^2)^s \left|\hat{f}_\theta(\omega)\right|^2 \mathrm{d}\omega$
  - *Implicitly defined* through optimisation algorithm (e.g. gradient descent of under-determined least-squares problem converges to interpolating solution with minimum L_2-norm)

Empirical risk

- **Empirical risk** (or **error**) replaces the expectation of the loss with an average on the training set $\{(x_i, y_i)\}_{i=1}^N$:

$$\hat{\mathcal{R}}(f) = \frac{1}{N}\sum_{i=1}^{N} \ell(f(x_i), y_i)$$

  - *Generalisation gap* $= \hat{\mathcal{R}}(f) - \mathcal{R}(f)$
  - $\hat{\mathcal{R}}(f)$ is a random function serving an *unbiased estimator* of $\mathcal{R}(f)$
  - *Variance* $\sigma(f) \sim \mathcal{O}\left(\frac{1}{\sqrt{N}}\right)$
  - *Hoeffding inequality* $P\left(\left|\hat{\mathcal{R}}(f) - \mathcal{R}(f)\right| > \varepsilon\right) \leq 2e^{-2\varepsilon^2 N}$
    "$\hat{\mathcal{R}}(f) = \mathcal{R}(f)$ is **probably approximately correct**"
  - Tells how good a given f is but not the model class $\mathcal{F}$
    (more delicate analysis: *VC dimension*, *Rademacher complexity*)



Empirical risk minimisation

- Supervised learning = minimisation of the **empirical risk** over a training set $\{(x_i, y_i)\}_{i=1}^{N}$ w.r.t. the parameters of a model class $\mathcal{F} = \{f_\theta : \mathcal{X} \rightarrow \mathcal{Y} \,:\, \theta \in \Theta\}$:

$$\hat{\theta} \in \operatorname*{argmin}_{\theta \in \Theta} \hat{\mathcal{R}}(f_\theta)$$

  in hope that the estimator $\hat{f} = f_{\hat{\theta}}$ generalises well, i.e., **excess risk** $\mathcal{R}(\hat{f}) - \mathcal{R}(f^*)$ is small

  - Usually a non-convex problem, can be solved only approximately!
  - Achieving a small **training error** $\hat{\mathcal{R}}(f_{\hat{\theta}})$ fundamentally depends on the richness of the model class (often, number of parameters $|\Theta|$)
  - Deep learning typically operates in *overparametrised regime* ($|\Theta| \gg N$), where multiple solutions are possible
  - How to make an informed choice among these solutions?

Regularisation

- **Regularisation** (or **capacity control**): find the "simplest" solution by restricting the model capacity *("Occam's razor principle"*)
  - *Constrained form:* $\hat{\theta}_\delta = \operatorname*{argmin}_{\theta\in\Theta} \hat{\mathcal{R}}(f_\theta) \quad \text{s.t.} \quad \gamma(\theta) \leq \delta$
  - *Penalised form:* $\hat{\theta} = \operatorname*{argmin}_{\theta\in\Theta} \hat{\mathcal{R}}(f_\theta) \ + \ \lambda\gamma(\theta)$
  - *Interpolation form:* $\hat{\theta} = \operatorname*{argmin}_{\theta\in\Theta} \gamma(\theta) \quad \text{s.t.} \quad \hat{\mathcal{R}}(f_\theta) = 0$
  - *Implicit form:* stems from the optimisation algorithm

Error decomposition

- **Excess risk** of the estimator $\hat{f} \approx f_{\widehat{\theta}}$ obtained by (approximate) constrained empirical risk minimisation over a nested family of functions $\mathcal{F}_\delta = \{f_\theta : \mathcal{X} \to \mathcal{Y} \, : \, \theta \in \Theta, \ \gamma(\theta) \leq \delta\}$

$$\mathcal{R}(\hat{f}) - \mathcal{R}(f^*)$$

Error decomposition

- **Excess risk** of the estimator $\hat{f} \approx f_{\hat{\theta}}$ obtained by (approximate) constrained empirical risk minimisation over a nested family of functions $\mathcal{F}_\delta = \{f_\theta : \mathcal{X} \to \mathcal{Y} : \theta \in \Theta,\ \gamma(\theta) \le \delta\}$

$$\mathcal{R}(\hat{f}) - \mathcal{R}(f^*) = \mathcal{R}(\hat{f}) - \boxed{\min_{\gamma(\theta)\le\delta} \mathcal{R}(f_\theta)} + \min_{\gamma(\theta)\le\delta} \mathcal{R}(f_\theta) - \mathcal{R}(f^*)$$

best model

$f_{\theta^*} \in \mathcal{F}_\delta$

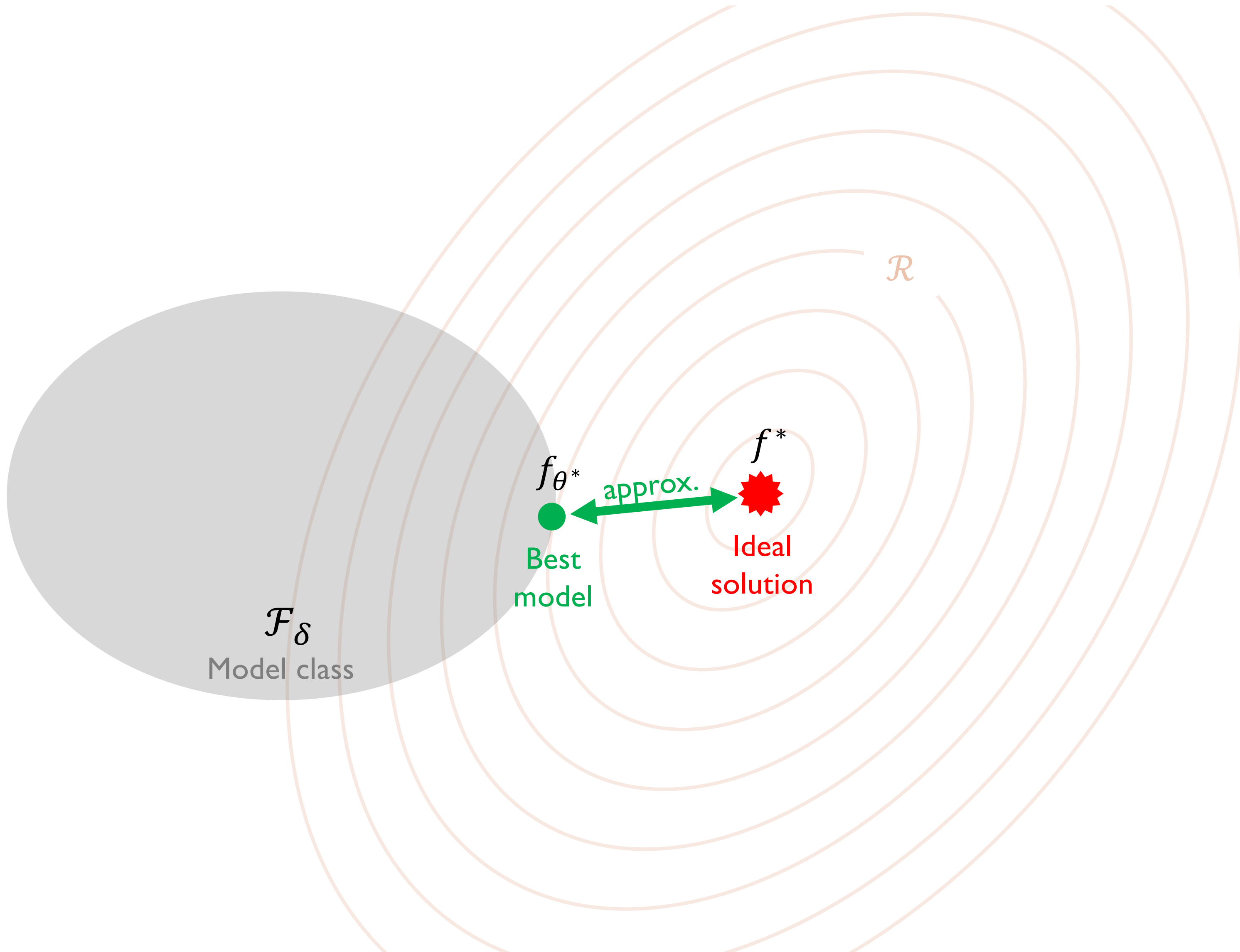
Note: Here we assume for simplicity the min is attained (more generally, should be inf)

Error decomposition

- **Excess risk** of the estimator $\hat{f} \approx f_{\hat{\theta}}$ obtained by (approximate) constrained empirical risk minimisation over a nested family of functions $\mathcal{F}_\delta = \{f_\theta : \mathcal{X} \to \mathcal{Y} : \theta \in \Theta,\ \gamma(\theta) \leq \delta\}$

$$\mathcal{R}(\hat{f}) - \mathcal{R}(f^*) = \mathcal{R}(\hat{f}) - \mathcal{R}(f_{\theta^*}) + \boxed{\mathcal{R}(f_{\theta^*}) - \mathcal{R}(f^*)}$$

approximation error
"how expressive $\mathcal{F}_\delta$ is"



$\mathcal{R}$
f^*
f_{θ^*}
approx.
Best
model
Ideal
solution
$\mathcal{F}_\delta$
Model class

Error decomposition

- **Excess risk** of the estimator $\hat{f} \approx f_{\hat{\theta}}$ obtained by (approximate) constrained empirical risk minimisation over a nested family of functions $\mathcal{F}_\delta = \{f_\theta : \mathcal{X} \to \mathcal{Y} : \theta \in \Theta,\ \gamma(\theta) \le \delta\}$

$$\mathcal{R}(\hat{f}) - \mathcal{R}(f^*) = \mathcal{R}(\hat{f}) - \mathcal{R}(f_{\theta^*}) + \mathcal{R}(f_{\theta^*}) - \mathcal{R}(f^*)$$

approximation error
"how expressive $\mathcal{F}_\delta$ is"

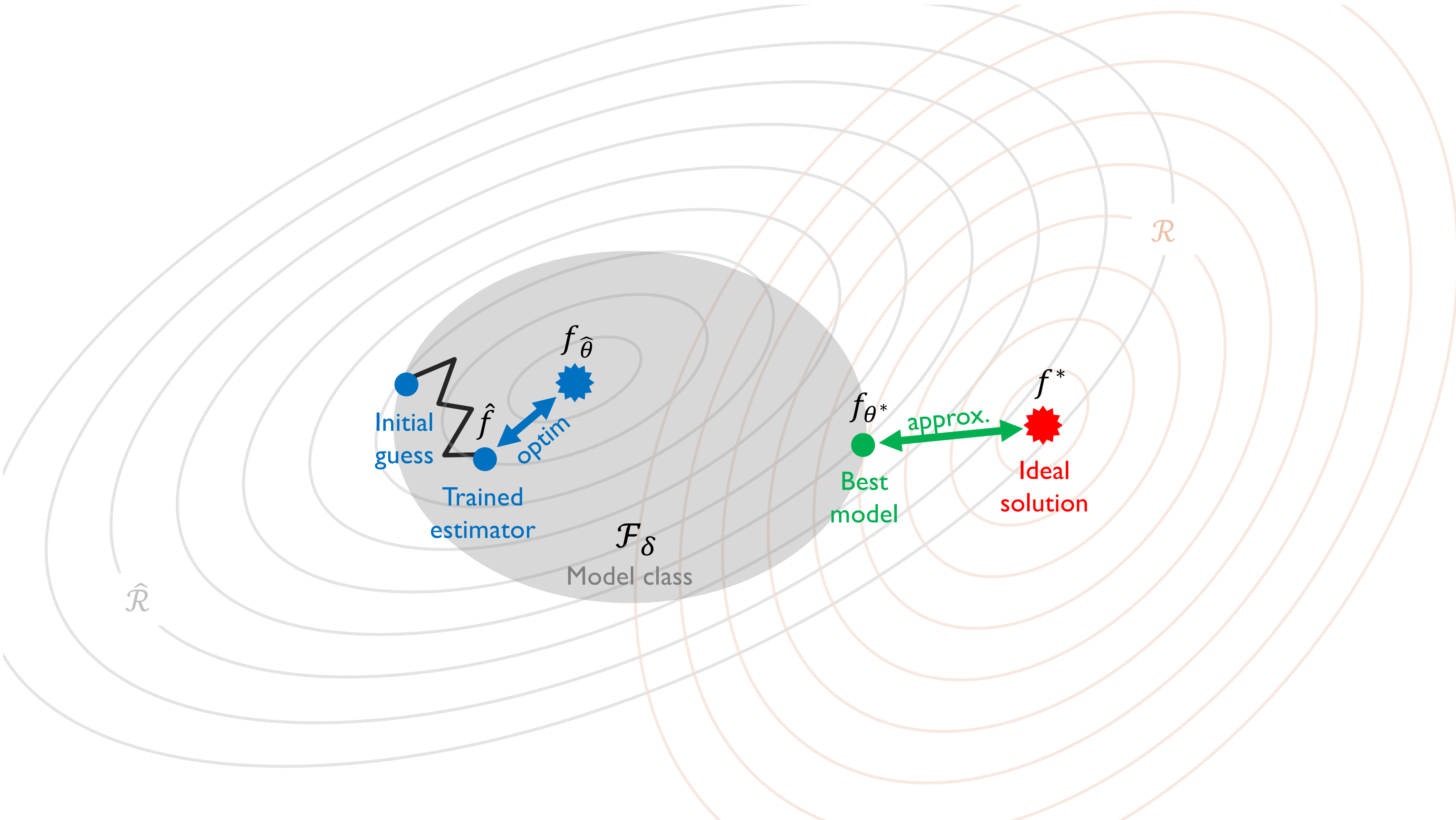
Error decomposition

- **Excess risk** of the estimator $\hat{f} \approx f_{\hat{\theta}}$ obtained by (approximate) constrained empirical risk minimisation over a nested family of functions $\mathcal{F}_\delta = \{f_\theta : \mathcal{X} \to \mathcal{Y} : \theta \in \Theta,\ \gamma(\theta) \leq \delta\}$

$$\mathcal{R}(\hat{f}) - \mathcal{R}(f^*) = \mathcal{R}(\hat{f}) - \hat{\mathcal{R}}(\hat{f}) + \hat{\mathcal{R}}(f_{\hat{\theta}}) - \mathcal{R}(f_{\theta^*}) + \underbrace{\hat{\mathcal{R}}(\hat{f}) - \hat{\mathcal{R}}(f_{\hat{\theta}})}_{\text{optimisation error}} + \underbrace{\mathcal{R}(f_{\theta^*}) - \mathcal{R}(f^*)}_{\text{approximation error}}$$

optimisation error
"how far $\hat{f}$ is from $f_{\hat{\theta}}$"

approximation error
"how expressive $\mathcal{F}_\delta$ is"



$\mathcal{R}$
$f_{\hat{\theta}}$
f^*
f_{θ^*}
approx.
Initial
guess
$\hat{f}$
optim
Trained
estimator
Best
model
Ideal
solution
$\mathcal{F}_\delta$
Model class
$\hat{\mathcal{R}}$

Error decomposition

- **Excess risk** of the estimator $\hat{f} \approx f_{\hat{\theta}}$ obtained by (approximate) constrained empirical risk minimisation over a nested family of functions $\mathcal{F}_\delta = \{f_\theta : \mathcal{X} \to \mathcal{Y} : \theta \in \Theta,\ \gamma(\theta) \le \delta\}$

$$\mathcal{R}(\hat{f}) - \mathcal{R}(f^*) = \mathcal{R}(\hat{f}) - \hat{\mathcal{R}}(\hat{f}) + \boxed{\hat{\mathcal{R}}(f_{\hat{\theta}}) - \mathcal{R}(f_{\theta^*})} + \boxed{\hat{\mathcal{R}}(\hat{f}) - \hat{\mathcal{R}}(f_{\hat{\theta}})} + \boxed{\mathcal{R}(f_{\theta^*}) - \mathcal{R}(f^*)}$$

$$\hat{\mathcal{R}}(f_{\hat{\theta}}) = \min_{\gamma(\theta) \le \delta} \hat{\mathcal{R}}(f) \le \hat{\mathcal{R}}(f_{\theta^*})$$

optimisation error
"how far $\hat{f}$ is from $f_{\hat{\theta}}$"

approximation error
"how expressive $\mathcal{F}_\delta$ is"

Error decomposition

- **Excess risk** of the estimator $\hat{f} \approx f_{\hat{\theta}}$ obtained by (approximate) constrained empirical risk minimisation over a nested family of functions $\mathcal{F}_\delta = \{f_\theta : \mathcal{X} \to \mathcal{Y} : \theta \in \Theta,\ \gamma(\theta) \leq \delta\}$

$$\mathcal{R}(\hat{f}) - \mathcal{R}(f^*) \leq \mathcal{R}(\hat{f}) - \hat{\mathcal{R}}(\hat{f}) + \hat{\mathcal{R}}(f_{\theta^*}) - \mathcal{R}(f_{\theta^*}) + \hat{\mathcal{R}}(\hat{f}) - \hat{\mathcal{R}}(f_{\hat{\theta}}) + \mathcal{R}(f_{\theta^*}) - \mathcal{R}(f^*)$$

optimisation error
"how far $\hat{f}$ is from $f_{\hat{\theta}}$"

approximation error
"how expressive $\mathcal{F}_\delta$ is"

Error decomposition

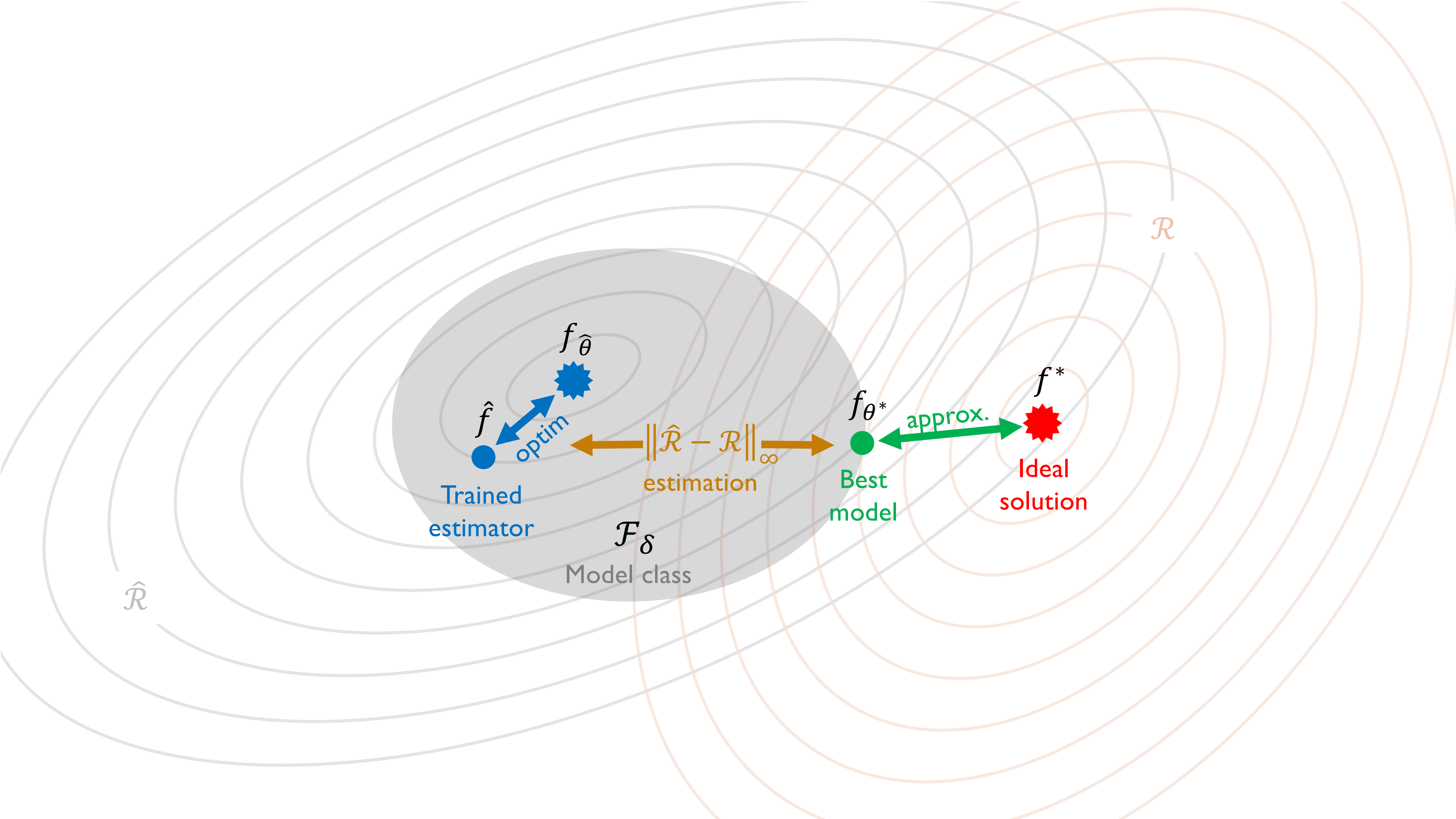
- **Excess risk** of the estimator $\hat{f} \approx f_{\hat{\theta}}$ obtained by (approximate) constrained empirical risk minimisation over a nested family of functions $\mathcal{F}_\delta = \{f_\theta : \mathcal{X} \to \mathcal{Y} : \theta \in \Theta,\ \gamma(\theta) \leq \delta\}$

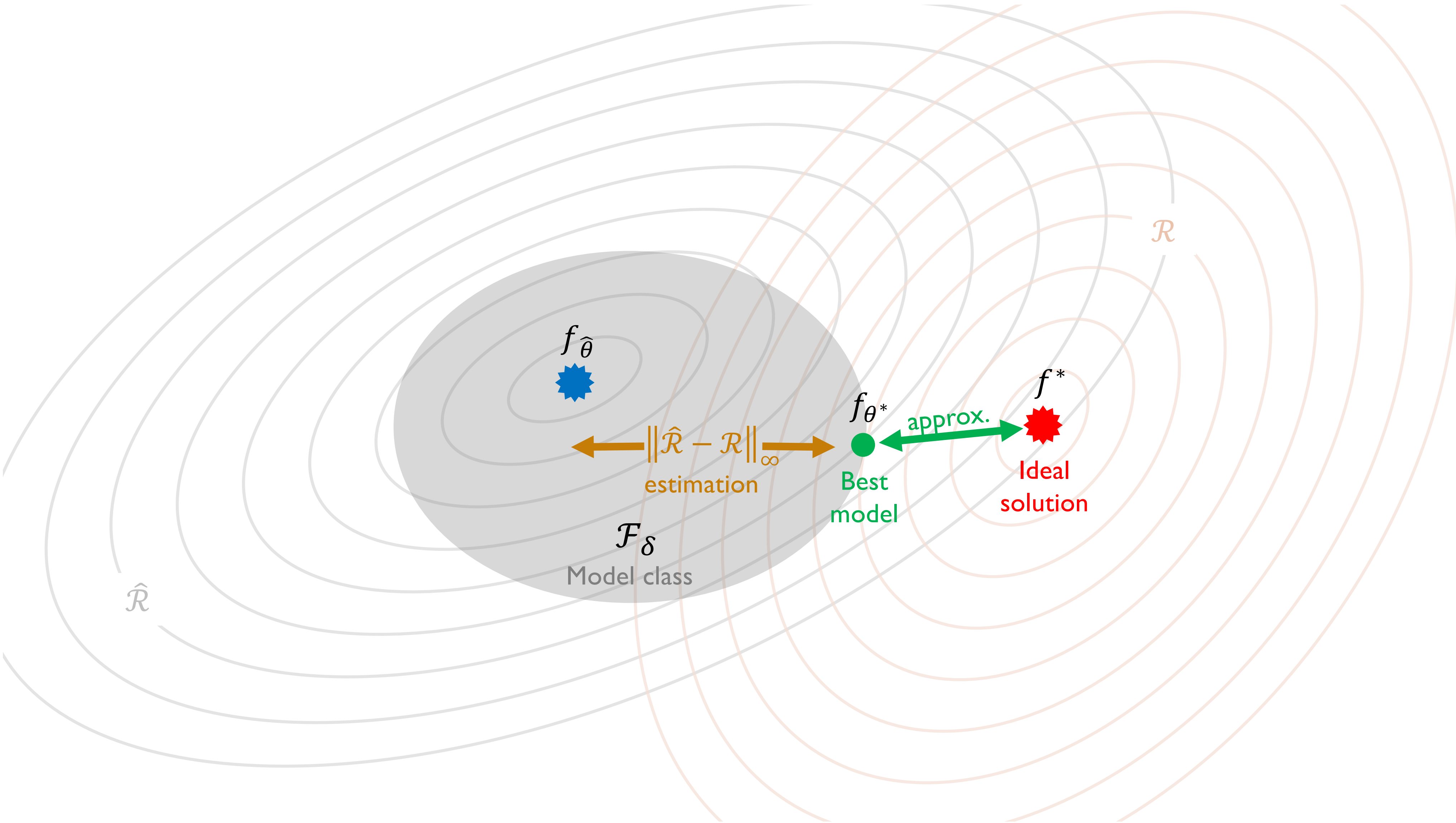
$$\mathcal{R}(\hat{f}) - \mathcal{R}(f^*) \leq \underbrace{2 \sup_{\gamma(\theta)\leq\delta} \left|\hat{\mathcal{R}}(f_\theta) - \mathcal{R}(f_\theta)\right|}_{\text{estimation error}} + \underbrace{\hat{\mathcal{R}}(\hat{f}) - \hat{\mathcal{R}}(f_{\hat{\theta}})}_{\text{optimisation error}} + \underbrace{\mathcal{R}(f_{\theta^*}) - \mathcal{R}(f^*)}_{\text{approximation error}}$$

estimation error
"how far the empirical risk is from the population risk"

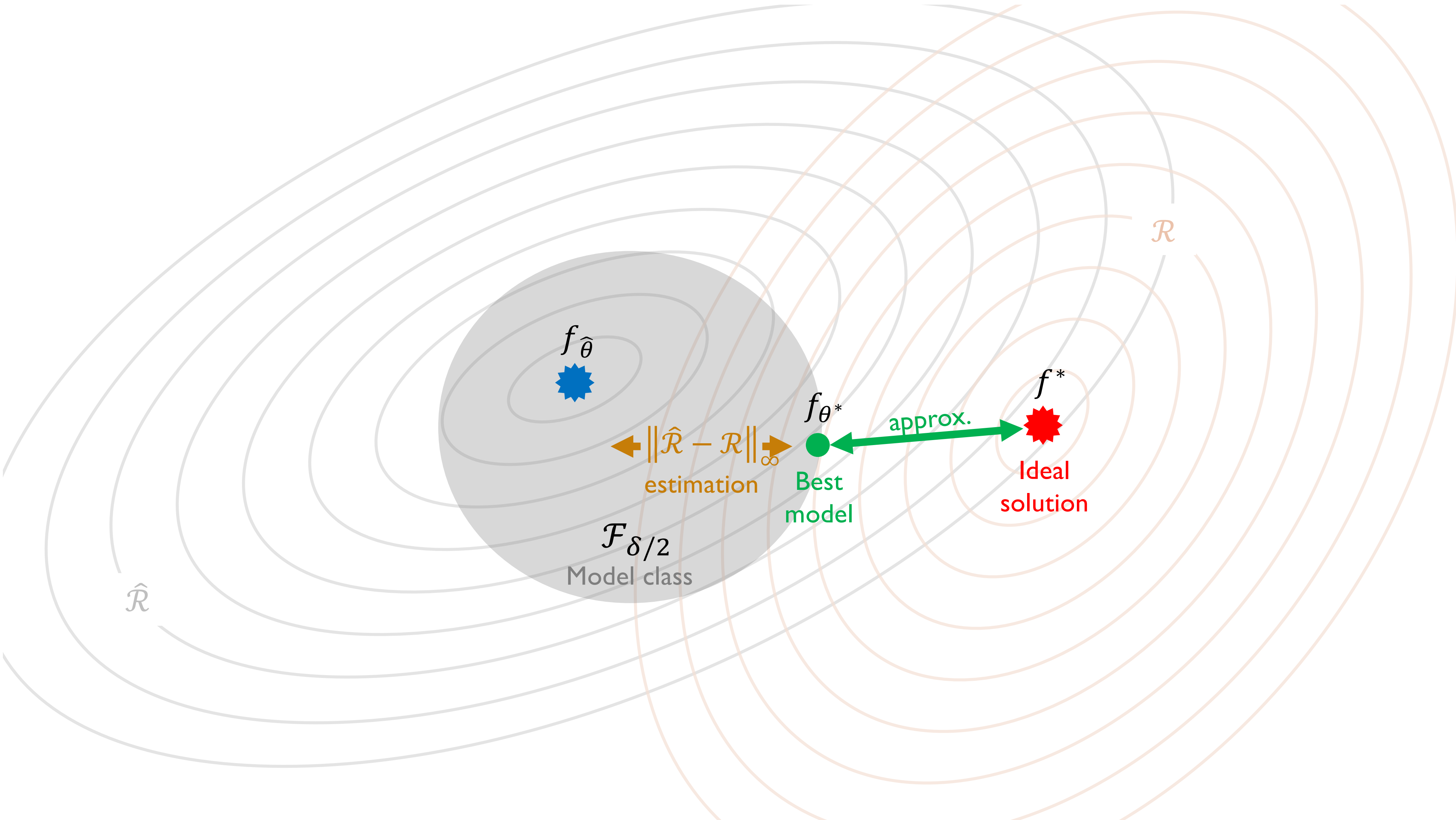
optimisation error
"how far $\hat{f}$ is from $f_{\hat{\theta}}$"

approximation error
"how expressive $\mathcal{F}_\delta$ is"


$f_{\hat{\theta}}$
$\hat{f}$
optim
$\|\hat{\mathcal{R}} - \mathcal{R}\|_\infty$
estimation
f_{θ^*}
approx.
f^*
$\mathcal{R}$
Trained estimator
Best model
Ideal solution
$\mathcal{F}_\delta$
Model class
$\hat{\mathcal{R}}$

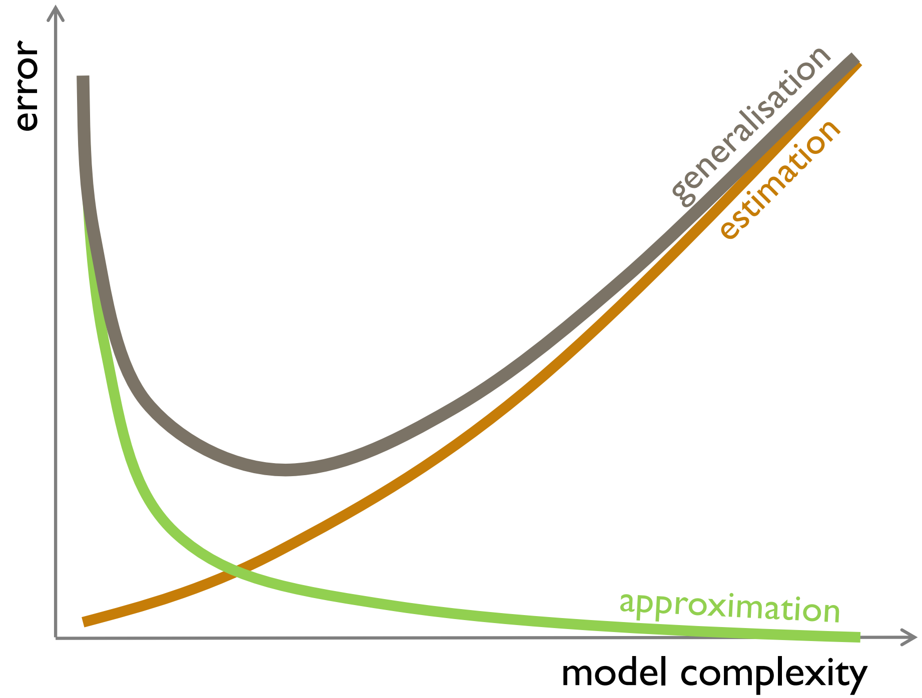


$\mathcal{R}$
$f_{\hat{\theta}}$
f^*
f_{θ^*}
approx.
$\|\hat{\mathcal{R}} - \mathcal{R}\|_\infty$
estimation
Best model
Ideal solution
$\mathcal{F}_\delta$
Model class
$\hat{\mathcal{R}}$

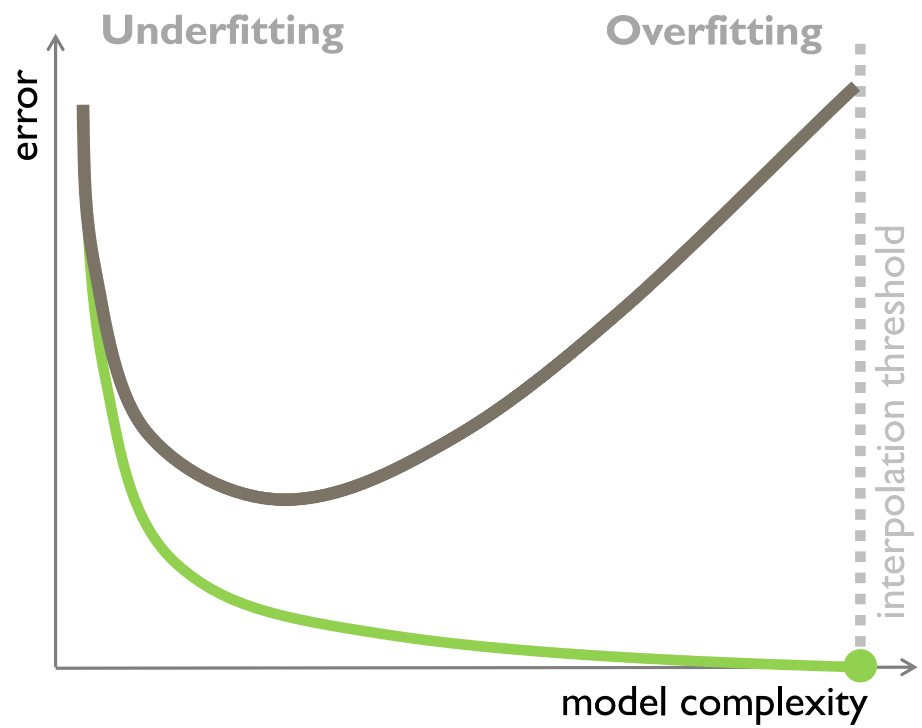


$\mathcal{R}$
$f_{\hat{\theta}}$
f^*
f_{θ^*}
approx.
$\|\hat{\mathcal{R}} - \mathcal{R}\|_\infty$
estimation
Best
model
Ideal
solution
$\mathcal{F}_{\delta/2}$
Model class
$\hat{\mathcal{R}}$

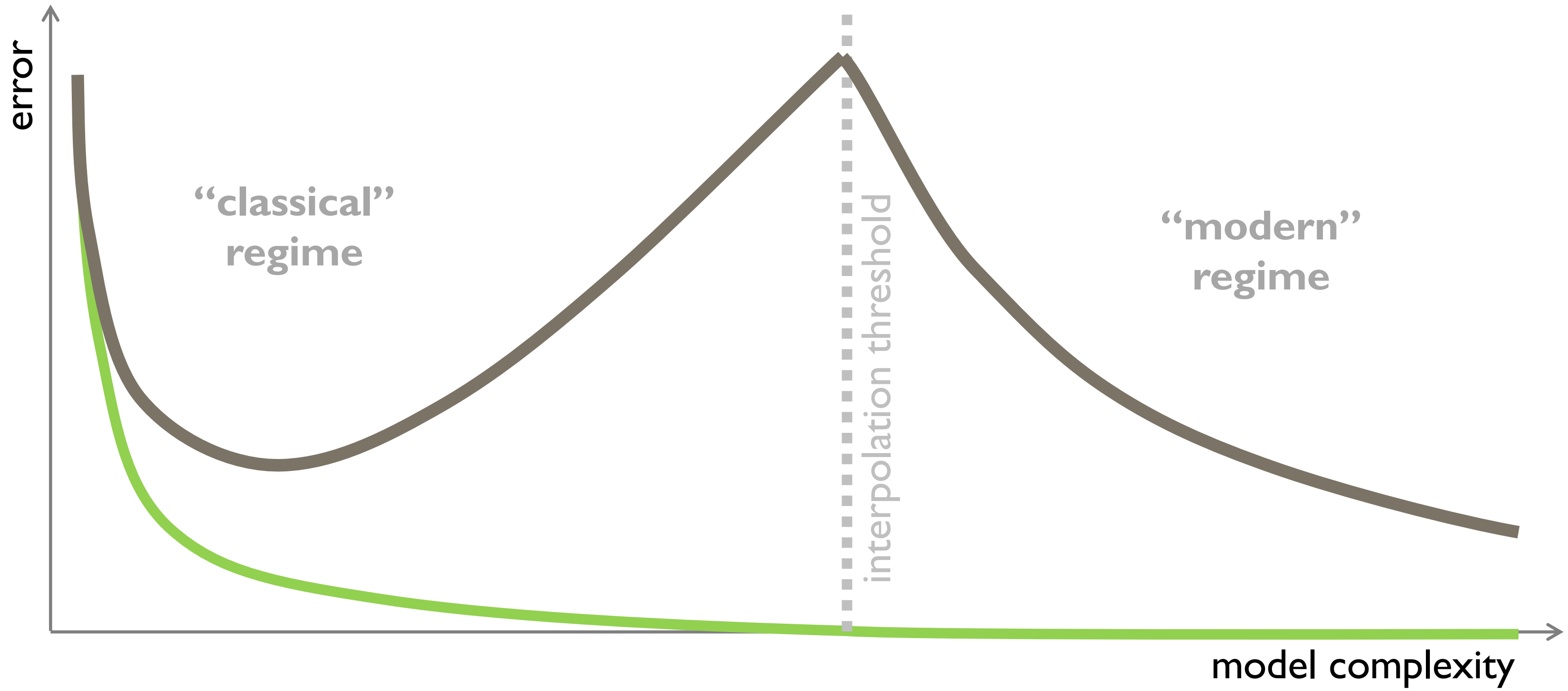
Classical Bias-Variance tradeoff



Classical Bias-Variance tradeoff



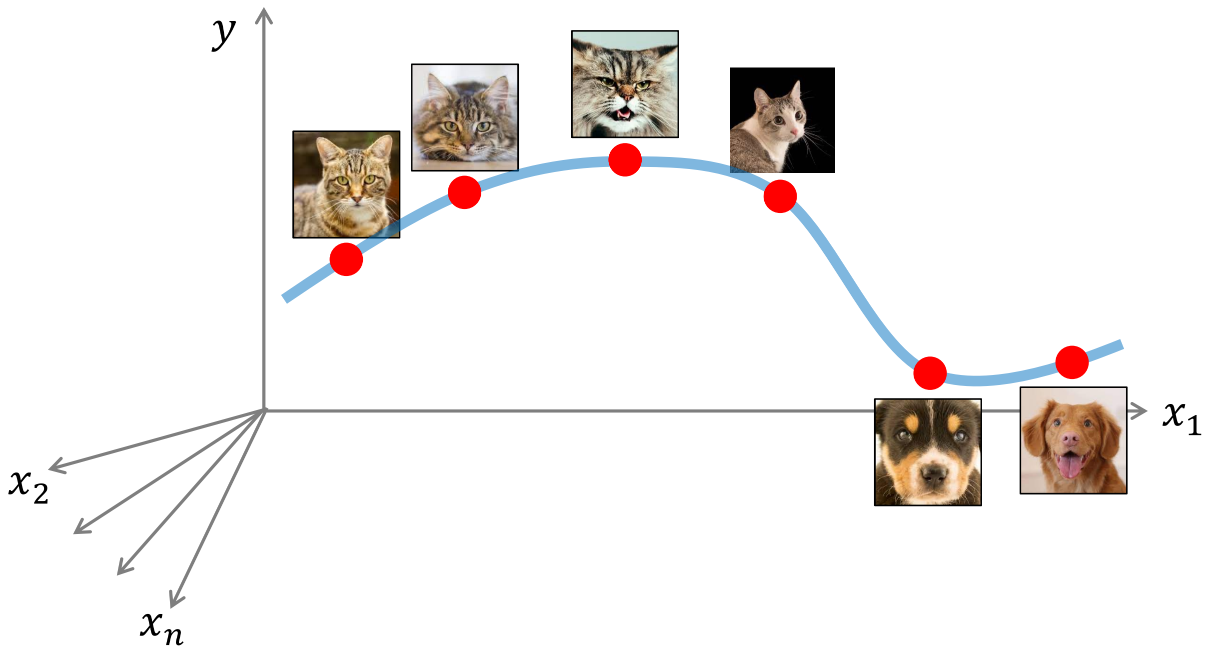
Modern Bias-Variance tradeoff: “Double Descent”



Neal et al. 2018; Belkin et al. 2019

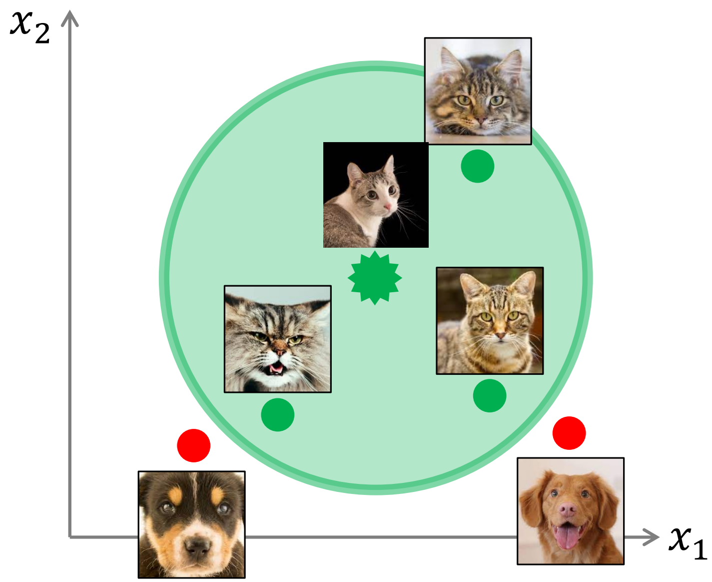
THE STORY IN HIGH DIMENSIONS

"Glorified curve fitting"



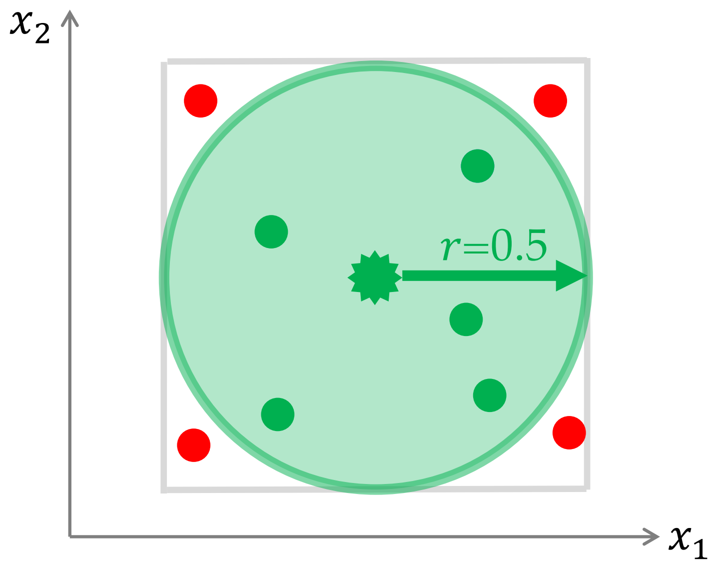
underlying assumption of function "regularity"

Nearest-neighbour classifier



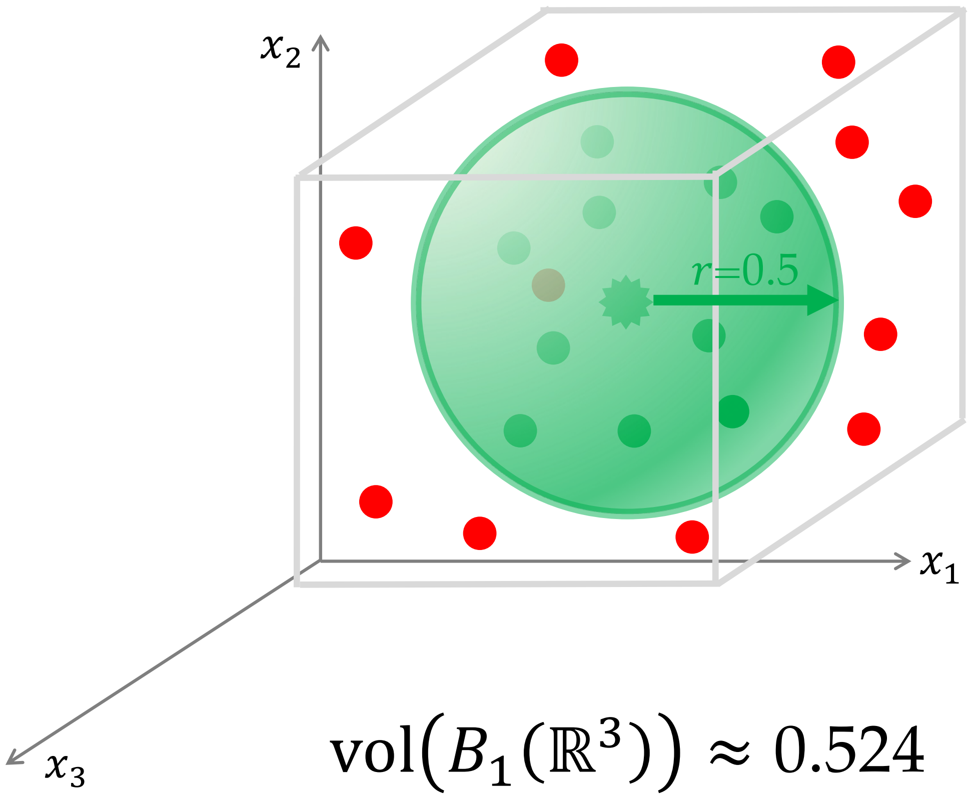
"my neighbours are similar to me"

Nearest-neighbour classifier



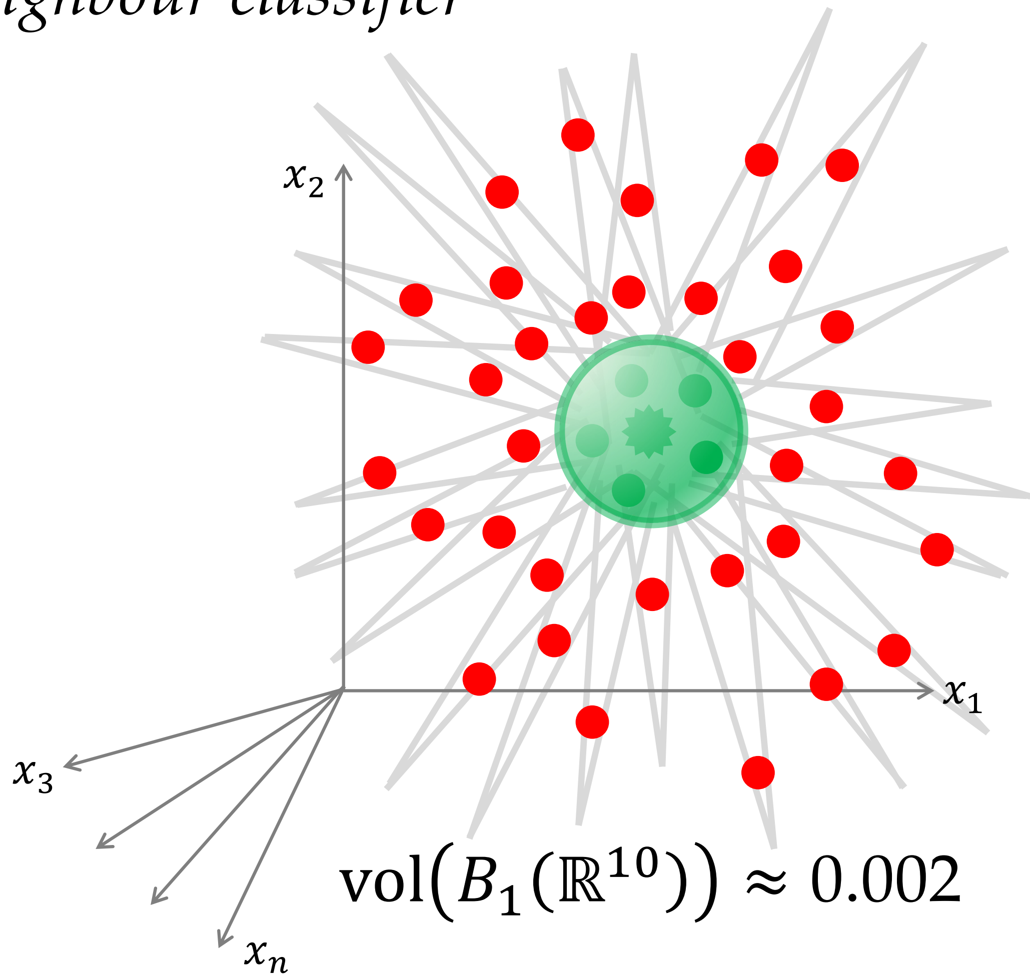
$$\mathrm{vol}\big(B_1(\mathbb{R}^2)\big) \approx 0.785$$

Nearest-neighbour classifier



$$\mathrm{vol}\big(B_1(\mathbb{R}^3)\big) \approx 0.524$$

Nearest-neighbour classifier



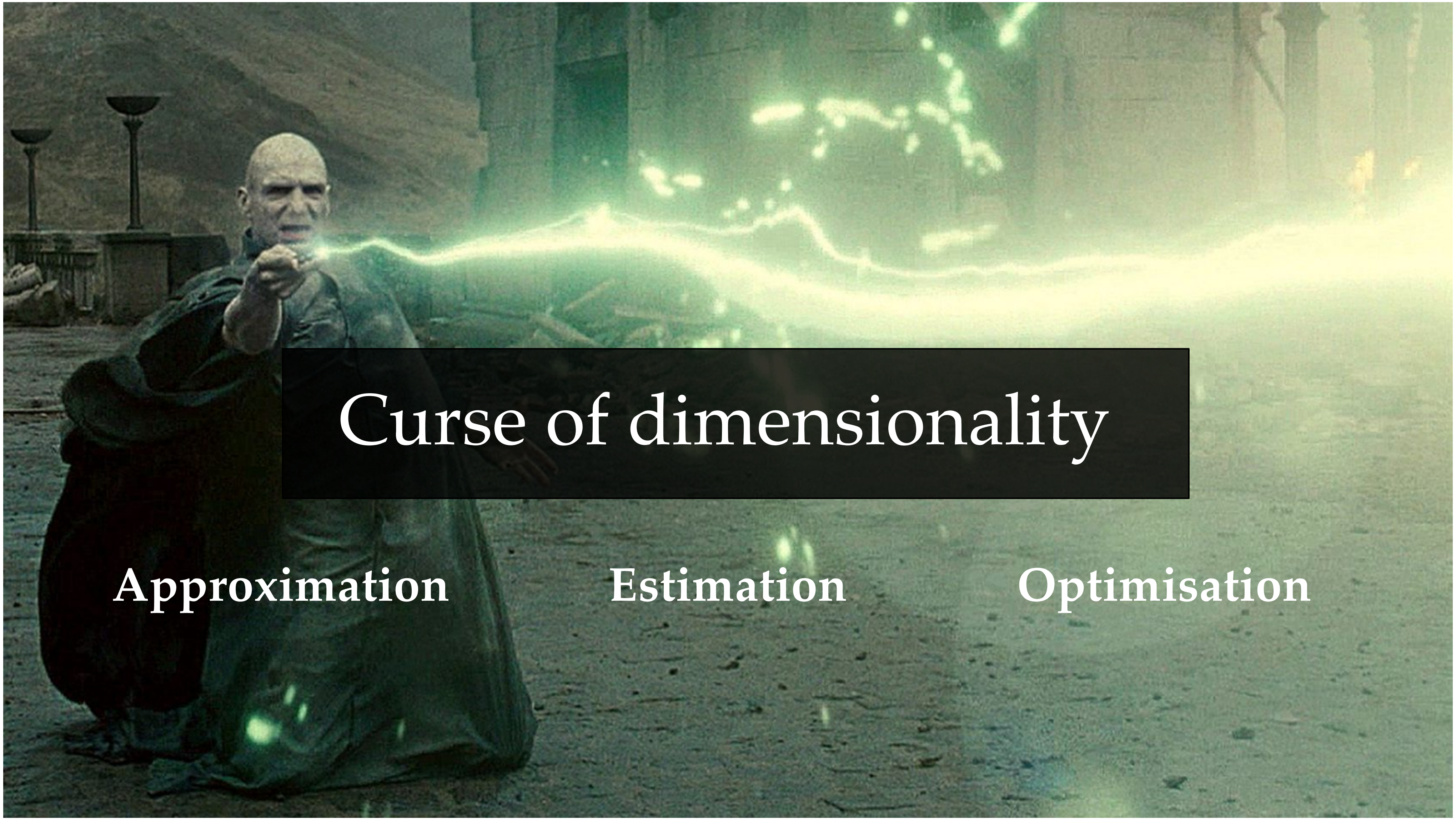
$$\mathrm{vol}\big(B_1(\mathbb{R}^{10})\big) \approx 0.002$$

"[dimensionality is] a curse which has hung over the head of the physicist and astronomer for many a year. "

— *Dynamic Programming*

R. Bellman

Bellman 1957


Curse of dimensionality
Approximation
Estimation
Optimisation

CURSE IN ESTIMATION

Learning Lipschitz functions

- A function $f: \mathcal{X} \subseteq \mathbb{R}^d \to \mathbb{R}$ is **$\boldsymbol{\beta}$-Lipschitz** if $|f(x) - f(x')| \leq \beta \|x - x'\|$
  - $\beta = \mathrm{Lip}(f)$ is the *Lipschitz constant* of f
  - Strong form of uniform continuity
  - Global property (unlike simple continuity)

Reminder:
f is **continuous** at x if $\forall \varepsilon > 0 \;\; \exists \delta > 0$ s.t. $\forall x' \;\; \|x - x'\| < \delta \implies |f(x) - f(x')| < \varepsilon.$
f is **uniformly continuous** if $\forall \varepsilon > 0 \;\; \exists \delta > 0$ s.t. $\forall x, x' \;\; \|x - x'\| < \delta \implies |f(x) - f(x')| < \varepsilon.$

Learning Lipschitz functions

- A function $f : \mathcal{X} \subseteq \mathbb{R}^d \to \mathbb{R}$ is $\boldsymbol{\beta}$**-Lipschitz** if $|f(x) - f(x')| \leq \beta \|x - x'\|$
    - $\beta = \mathrm{Lip}(f)$ is the *Lipschitz constant* of f
    - Strong form of uniform continuity
    - Global property (unlike simple continuity)

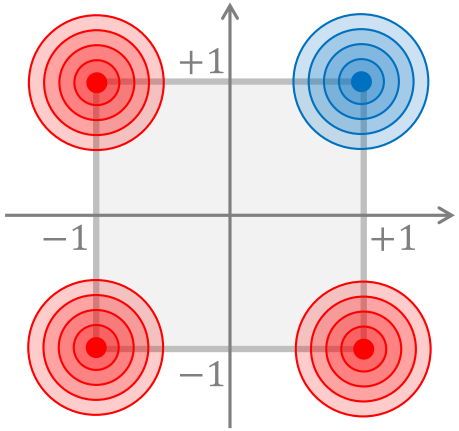
How many samples N are needed to approximate a Lipschitz function in $\mathbb{R}^d$ with accuracy ε?

Learning Lipschitz functions: Lower bound

- Consider a 1-Lipschitz function constructed as a superpositions of blobs placed at the corners $H_d = \{(z_1, \dots, z_d) : z_i = \pm 1\}$ of a d-dimensional hypercube

$$f(x) = \sum_{z \in H_d} c_z \varphi(x - z) \qquad c_z = \pm 1$$

- Assume f is sampled at N samples



Exercise: prove that if $N \ll 2^d$ then any estimator $\hat{f}$ will incur a relative error of $\frac{\mathbb{E}|f - \hat{f}|^2}{\mathbb{E}|f|^2} = \Theta(1)$

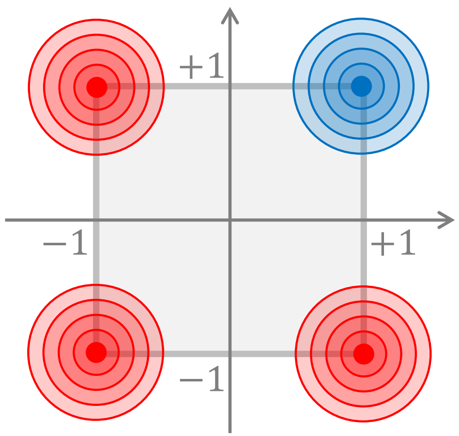
Von Luxburg, Bosquet 2004

Learning Lipschitz functions: Lower bound

- Consider a 1-Lipschitz function constructed as a superpositions of blobs placed at the corners $H_d = \{(z_1, \dots, z_d) : z_i = \pm 1\}$ of a d-dimensional hypercube

$$f(x) = \sum_{z \in H_d} c_z \varphi(x - z) \qquad c_z = \pm 1$$

- Assume f is sampled at N samples

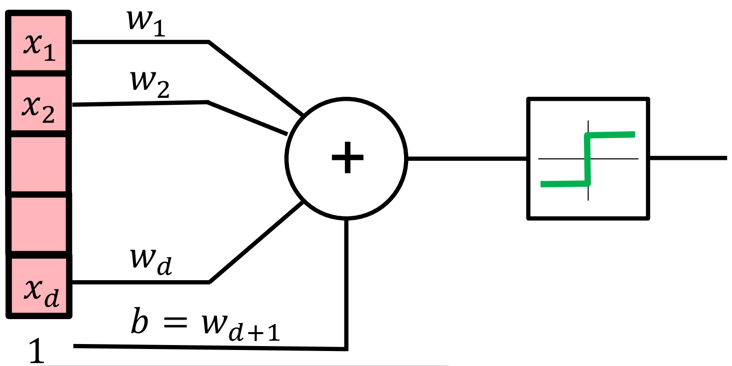


Learning Lipschitz functions is a dimensionality-cursed problem

Von Luxburg, Bosquet 2004

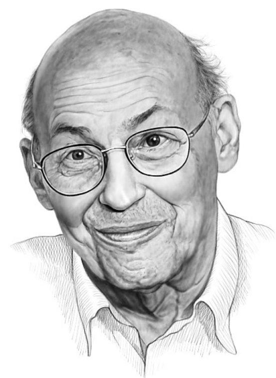
CURSE IN APPROXIMATION

Simple Perceptrons

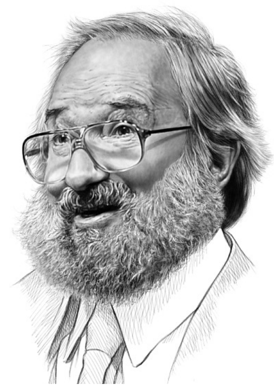


$$y = \mathrm{sign}(\mathbf{w}^{\mathrm{T}}\mathbf{x})$$

F. Rosenblatt



M. Minsky

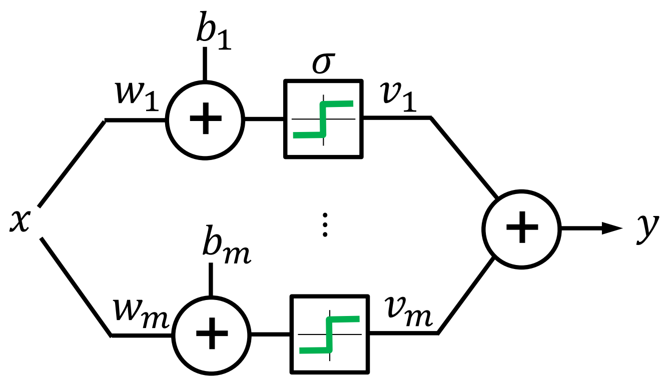


S. Papert

Shallow Perceptrons

- **Two-layer perceptron** with a non-polynomial activation function σ

$$\mathcal{F} = \left\{ f(x) = \sum_{j \le m} v_j \sigma(w_j^{\mathrm{T}} x + b_j) \right\}$$



Shallow Perceptrons

- **Two-layer perceptron** with a non-polynomial activation function σ

$$\mathcal{F} = \left\{ f(x) = \sum_{j \le m} v_j \sigma(w_j^{\mathrm{T}} x + b_j) \right\}$$

- Parametrised by weights $\mathbf{W}, \mathbf{b}, \mathbf{v}$
- Various definitions of capacity, e.g.
  - *Number of neurons:* $\gamma(f) = m$
  - *Path norm:* $\gamma(f) = \sum |v_j| \left(\|w_j\| + |b_j| \right)$

Shallow Perceptrons are universal approximators

- **Two-layer perceptron** with a non-polynomial activation function σ

$$\mathcal{F} = \left\{ f(x) = \sum_{j \leq m} v_j \sigma(w_j^{\mathrm{T}} x + b_j) \right\}$$

> **Universal Approximation Theorem:** $\mathcal{F}$ is *dense* in the class of continuous d-dimensional functions w.r.t. the uniform compact topology

$A \subseteq X$ is **dense** in X if $\bar{A} = X$, where $\bar{A} = A \cup \{\lim_{n\to\infty} a_n : a_n \in A\}$

G. Cybenko

K. Hornik

Hilbert 1990 (Thirteenth Problem); Kolmogorov 1956; Arnold 1957 ("Superposition Theorem"); Hecht-Nielsen 1987 (first use in neural networks)
Cybenko 1989; Funahashi 1989; Hornik et al. 1989; Barron 1993; Leshno et al. 1993; Maiorov 1999; Pinkus 1999

Shallow Perceptrons are universal approximators

- **Two-layer perceptron** with a non-polynomial activation function σ

$$\mathcal{F} = \left\{ f(x) = \sum_{j \le m} v_j \sigma(w_j^{\mathrm{T}} x + b_j) \right\}$$

Universal Approximation Theorem: $\mathcal{F}$ can uniformly approximate any continuous d-dimensional function on a compact set to any desired accuracy ε.

G. Cybenko

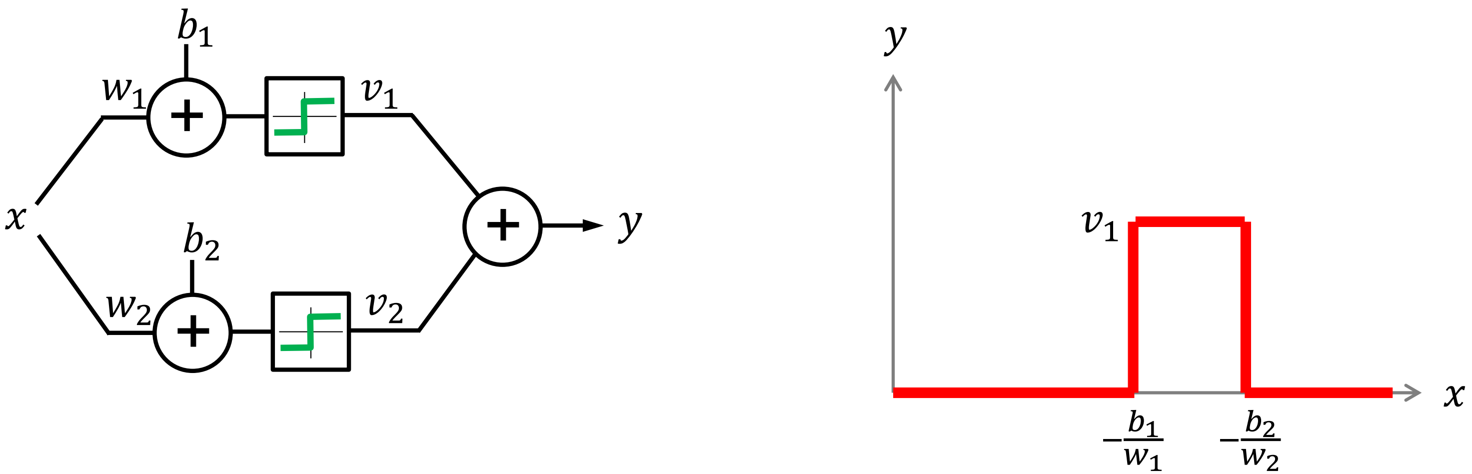
K. Hornik

Hilbert 1990 (Thirteenth Problem); Kolmogorov 1956; Arnold 1957 (“Superposition Theorem”); Hecht-Nielsen 1987 (first use in neural networks)
Cybenko 1989; Funahashi 1989; Hornik et al. 1989; Barron 1993; Leshno et al. 1993; Maiorov 1999; Pinkus 1999

Shallow Perceptrons are universal approximators

- **Two-layer perceptron** with a non-polynomial activation function σ

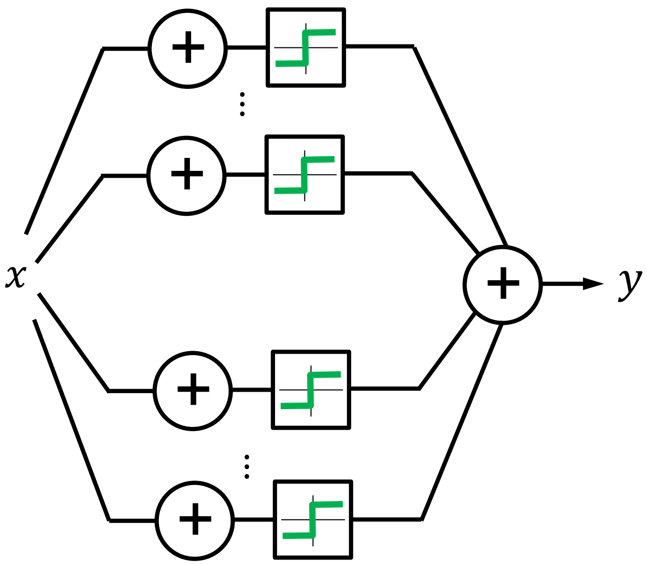
$$\mathcal{F} = \left\{ f(x) = \sum_{j \le m} v_j \sigma(w_j^{\mathrm{T}} x + b_j) \right\}$$

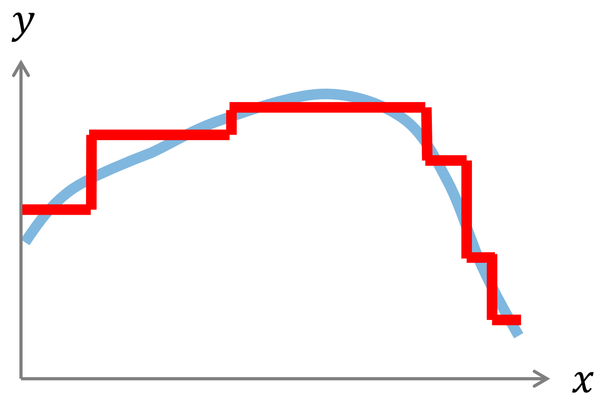


Shallow Perceptrons are universal approximators

- **Two-layer perceptron** with a non-polynomial activation function σ

$$\mathcal{F} = \left\{ f(x) = \sum_{j \le m} v_j \sigma(w_j^{\mathrm{T}} x + b_j) \right\}$$





Shallow Perceptrons are universal approximators

Universal Approximation Theorem: σ is not polynomial iff for every continuous function $f: K \subset \mathbb{R}^d \to \mathbb{R}$ defined on a *compact set* K and $\varepsilon > 0$, there exists a two-layer Perceptron with m neurons and weights $\mathbf{W}, \mathbf{b}, \mathbf{v}$ s.t.

$$\max_{x \in K} \left| f(x) - \sum_{j \le m} v_j \sigma(w_j^{\mathrm{T}} x + b_j) \right| < \varepsilon$$

- Fixed number of layers (*"bounded depth"*)
- Does not tell how many neurons m are needed (*"arbitrary width"*)
- Existence result: does not tell *how* to find the weights
- There are stronger results, including bounded depth and width

Cybenko 1989; Hornik 1991; Pinkus 1999

Shallow Perceptrons are universal approximators

Universal Approximation Theorem: σ is not polynomial iff for every continuous function $f: K \subset \mathbb{R}^d \to \mathbb{R}$ defined on a *compact set* K and $\varepsilon > 0$, there exists a two-layer Perceptron with m neurons and weights $\mathbf{W}, \mathbf{b}, \mathbf{v}$ s.t.

$$\max_{x \in K} \left| f(x) - \sum_{j \le m} v_j \sigma(w_j^{\mathrm{T}} x + b_j) \right| < \varepsilon$$

What is the relation between dimension d, number of neurons m, and the error ε?

Cybenko 1989; Hornik 1991

Approximation rates

- Bound on the approximation error

$$\varepsilon = \inf_{g \in \mathcal{F}} \sup_{x \in K \subset \mathbb{R}^d} |f(x) - g(x)|$$

w.r.t. d, m for different classes of functions

- *Sobolev class* $f \in H^s(\mathbb{R}^d) = \left\{ f \in L_2(\mathbb{R}^d) : \int_{\mathbb{R}^d} (1 + \|\omega\|^2)^s \left|\hat{f}(\omega)\right|^2 \mathrm{d}\omega < \infty \right\}$

  error is exponential $\varepsilon = \mathcal{O}\left(m^{-s/d}\right)$ **dimensionality-cursed!**

  "functions with sufficiently many derivatives"

Maiorov 1999

Approximation rates

- Bound on the approximation error

$$\varepsilon = \inf_{g \in \mathcal{F}} \sup_{x \in K \subset \mathbb{R}^d} |f(x) - g(x)|$$

w.r.t. d, m for different classes of functions

- *Sobolev class* $\quad f \in H^s(\mathbb{R}^d) = \left\{ f \in L_2(\mathbb{R}^d) : \int_{\mathbb{R}^d} (1 + \|\omega\|^2)^s \left|\hat{f}(\omega)\right|^2 \mathrm{d}\omega < \infty \right\}$

  error is exponential $\varepsilon = \mathcal{O}\left(m^{-s/d}\right)$ **dimensionality-cursed!**

- *Barron class* $\quad f \in \left\{ f \in L_2(\mathbb{R}^d) : \int_{\mathbb{R}^d} \|\omega\|^2 \left|\hat{f}(\omega)\right|^2 \mathrm{d}\omega < \infty \right\}$

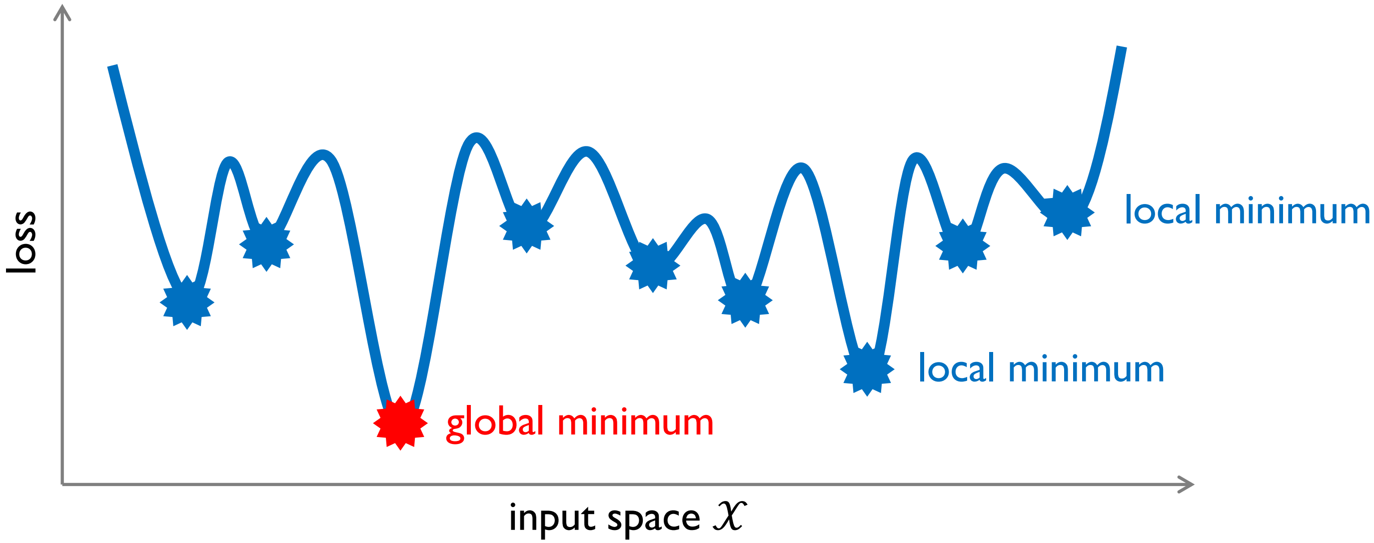
  error is $\varepsilon = \mathcal{O}(m^{-1})$ **too strong assumption in practice!**

Maiorov 1999; Barron 1993

CURSE IN OPTIMISATION

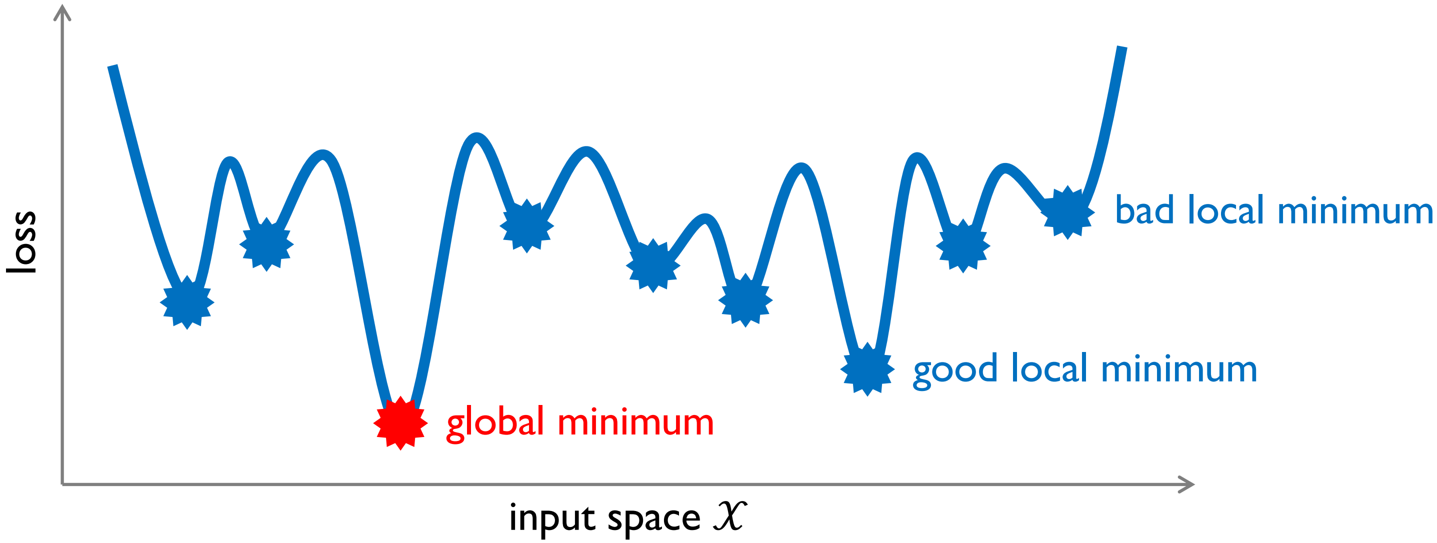
How hard is optimisation in high dimensions?

- Finding a global optimum of a generic high-dimensional function is NP-hard



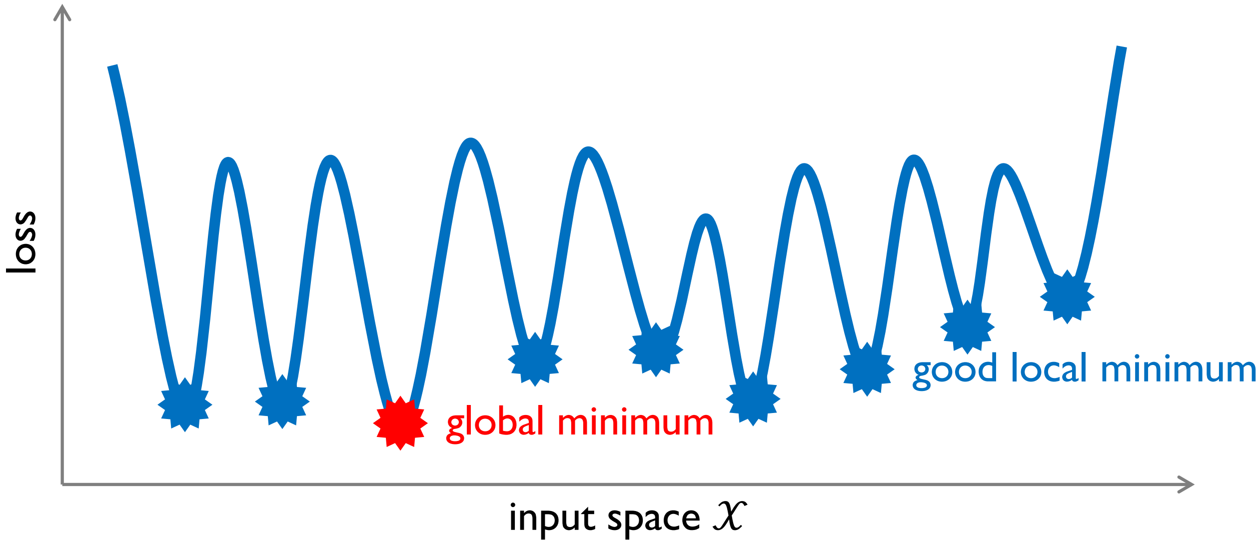
How hard is optimisation in high dimensions?

- Finding a global optimum of a generic high-dimensional function is NP-hard



How hard is optimisation in high dimensions?

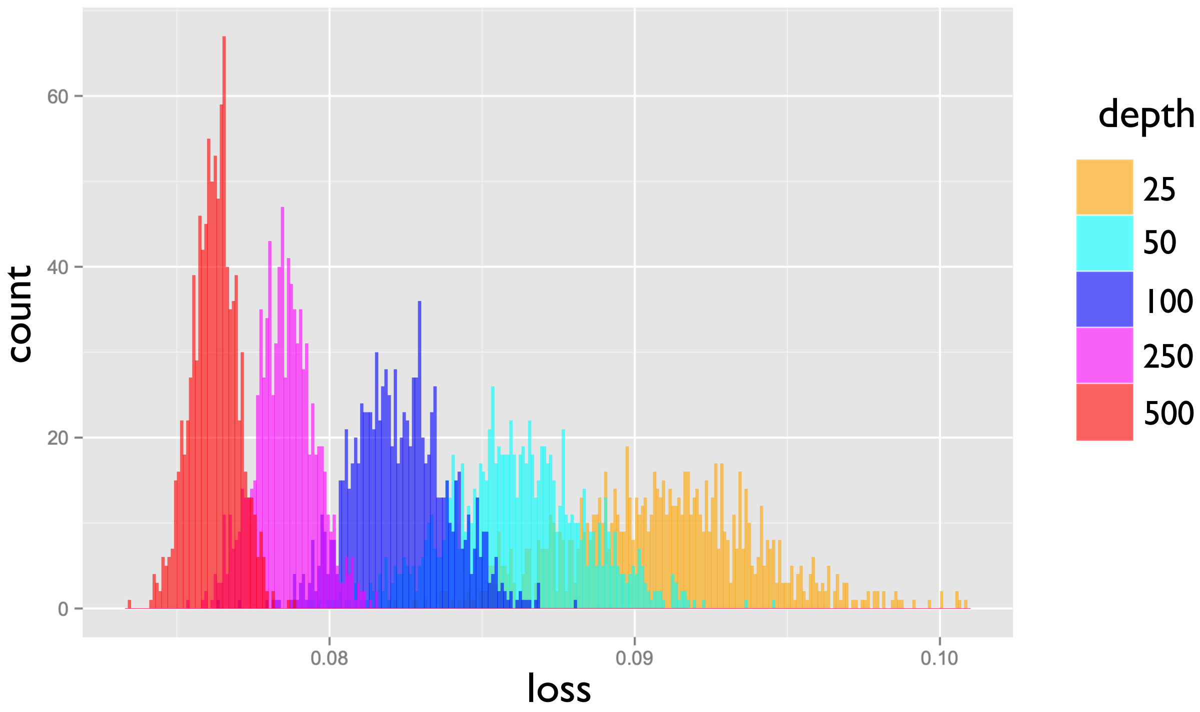
- Finding a global optimum of a generic high-dimensional function is NP-hard
- Deep neural networks have more benign landscapes with no "bad local minima"



How hard is optimisation in high dimensions?

- Finding a global optimum of a generic high-dimensional function is NP-hard
- Deep neural networks have more benign landscapes with no “bad local minima”
  - Most local minima are equivalent and yield similar test performance
  - The probability of finding a “bad” local minimum decreases with network depth
  - Finding the global minimum on the training set (as opposed to one of the many good local ones) is not useful in practice and may lead to overfitting

How hard is optimisation in high dimensions?



Choromanska et al. 2015

How hard is optimisation in high dimensions?

- Finding a global optimum of a generic high-dimensional function is NP-hard
- Deep neural networks have more benign landscapes with no “bad local minima”
  - Most local minima are equivalent and yield similar test performance
  - The probability of finding a “bad” local minimum decreases with network depth
  - Finding the global minimum on the training set (as opposed to one of the many good local ones) is not useful in practice and may lead to overfitting
- Gradient descent can efficiently find local minima in high dimension

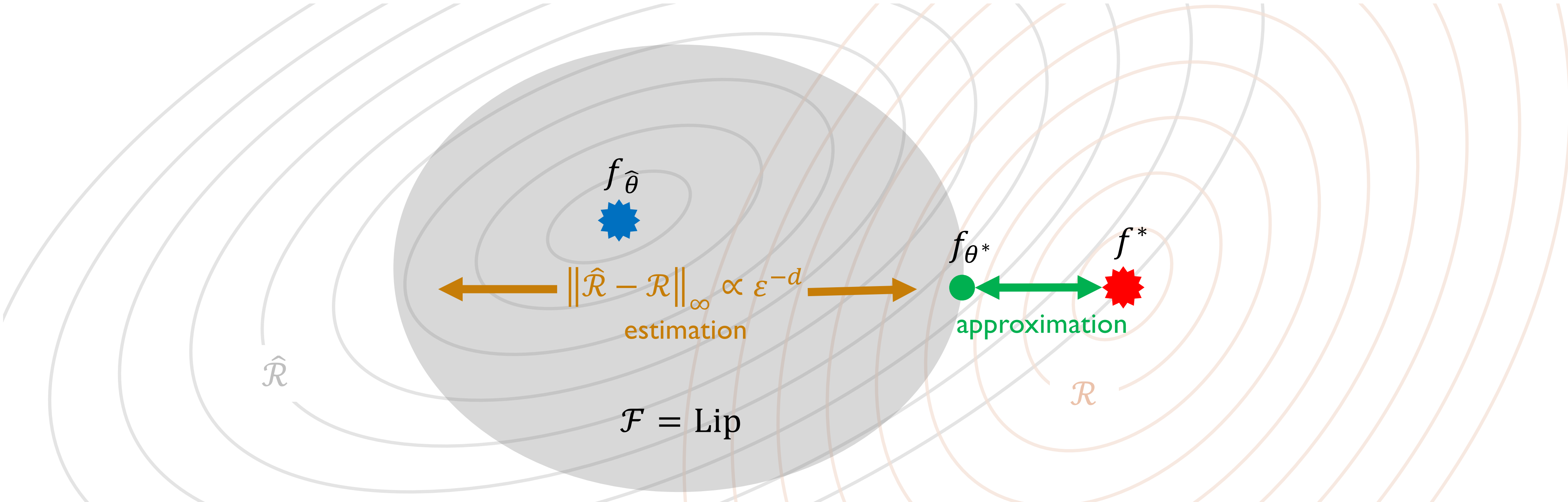
> **Typical result:** Noisy gradient descent can find ε-approximate second-order stationary points of a β-smooth loss function in $\tilde{O}(\beta \log d/\varepsilon^2)$ iterations.

Jin 2017

GEOMETRIC REGULARITY

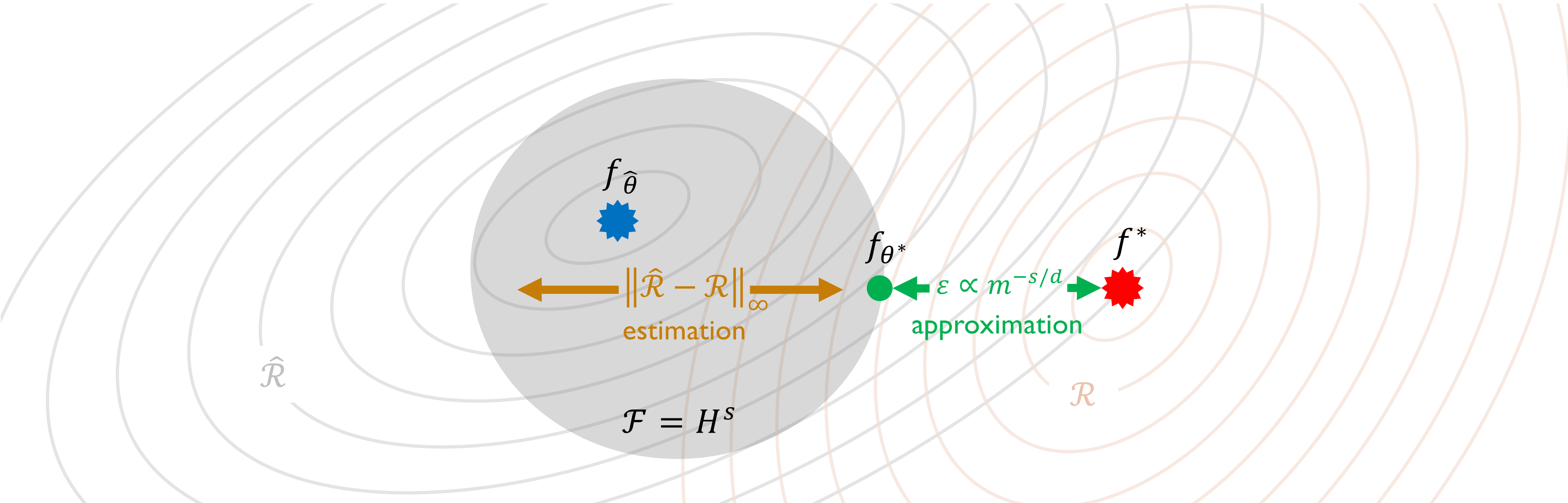
Findings so far: classical notions of regularity are of little use!

- Lipschitz class is too large: estimation error is dimensionality-cursed

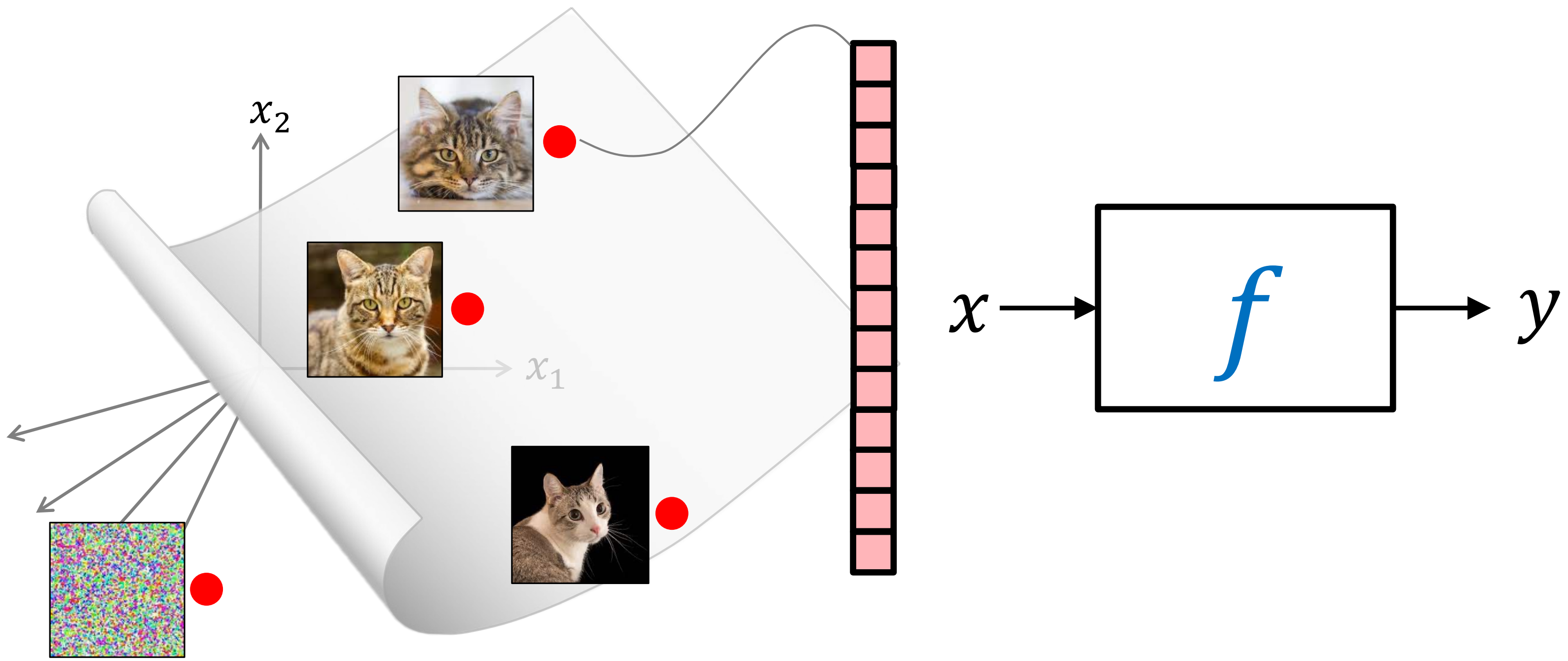


Findings so far: classical notions of regularity are of little use!

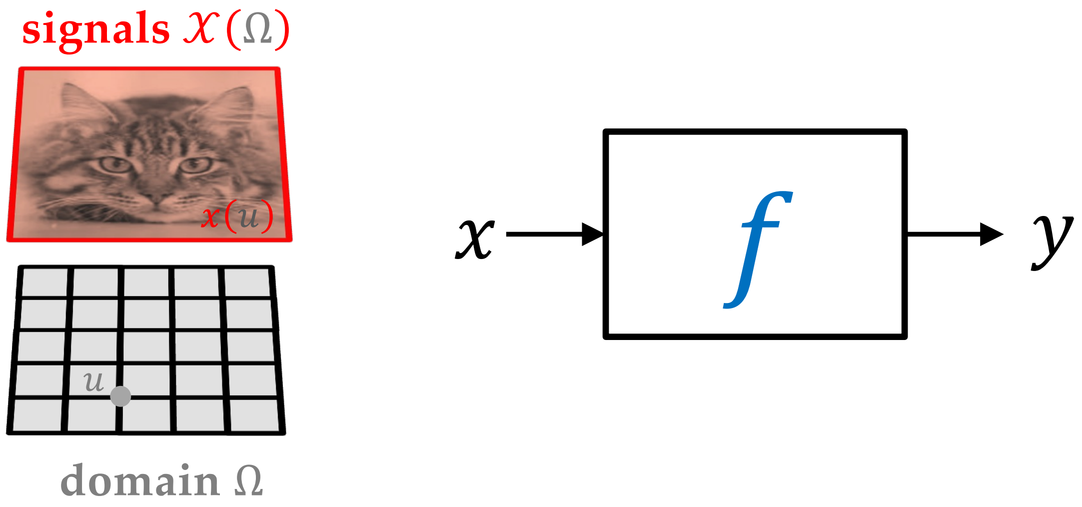
- Lipschitz class is too large: estimation error is dimensionality-cursed
- Sobolev class is too small: approximation error is dimensionality-cursed



Geometric priors



Geometric priors



Takeaways

- Learning in high dimensions is plagued by the *curse of dimensionality*
- Impossible without assumptions ("priors")
- Classical assumptions of regularity (from low-dimensional analysis) are not appropriate priors
- *Geometric priors:* inputs are signals defined over low-dimensional geometric domains
- Next lectures: how to incorporate geometric priors into neural network architectures *("Geometric Deep Learning")*

Key Concepts

- Approximation, Estimation & Optimisation errors
- Bias-Variance tradeoff
- Curse of dimensionality
- Universal approximation

Main References

- M. Bronstein et al., Geometric deep learning, *arXiv:2104.13478, 2021*. Section 2 "Learning in high dimensions"
- O. Bosquet, S. Boucheron, G. Lugosi, Introduction to statistical learning theory, *Lecture Notes in Computer Science* 3176, Springer, 2004. Basics of statistical ML
- U. von Luxburg, O. Bosquet, Distance-based classification with Lipschitz functions, *JMLR* 5:669–695, 2004. Bounds for Lipschitz functions
- A. Pinkus, Approximation theory of the MLP model in neural networks, *Acta Numerica* 8:143–195, 1999. Universal approximation results for neural networks

Background Pre-Read

- A. Bronstein, Probability and statistics: a survival guide, Course notes. Refresher of probability & statistics for ML