How neural networks learn simple functions?

Florent Krzakala

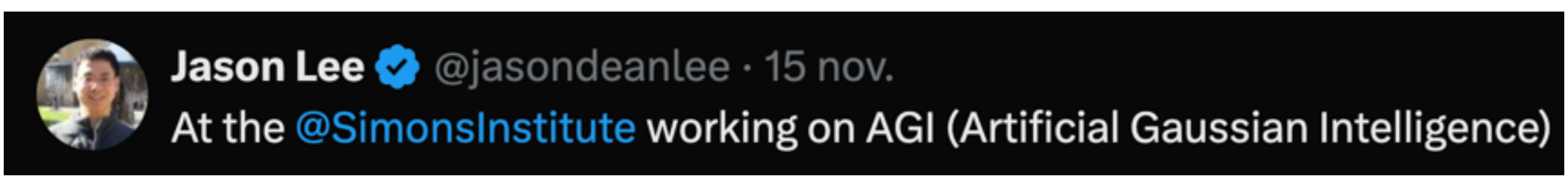
Souvenirs from 2016 in Berkeley

Unknown futures of generalisation?

A physicist's bias: focus on understanding simple problems

Unknown futures of generalisation?

A physicist's bias: focus on understanding simple problems



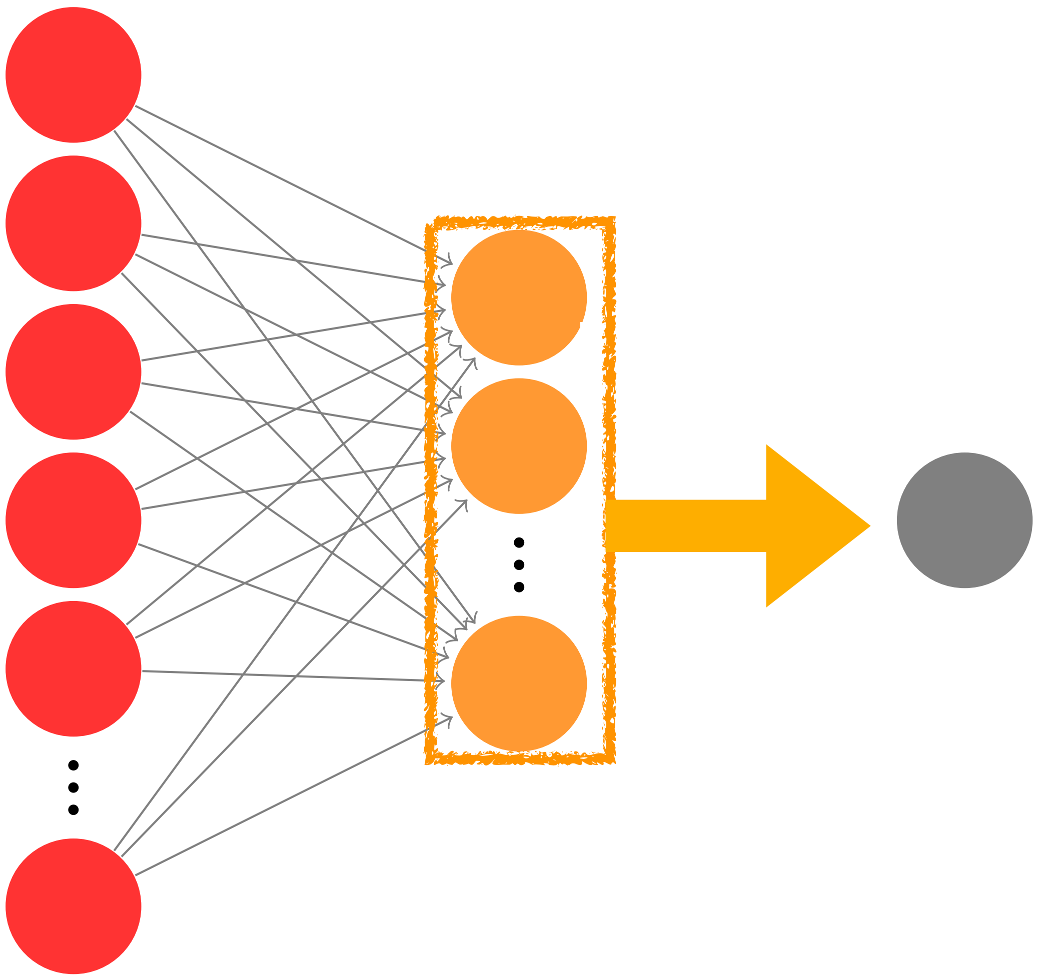
I

Multi-index functions and the necessity of feature learning

Multi-index functions…

Target function: $Y \sim P^\star(Y \,|\, H = W^\star \mathbf{X})$

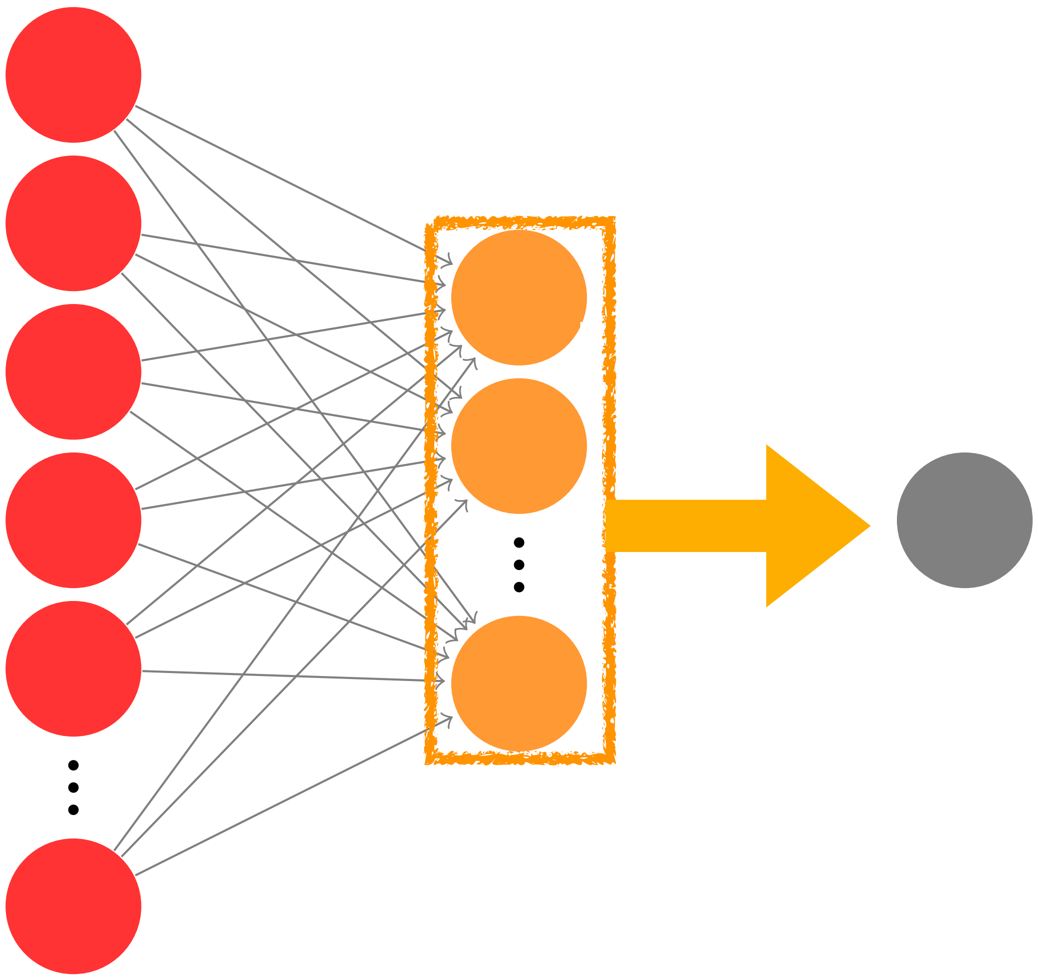
$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$



Multi-index functions…

Target function: $Y \sim P^\star(Y \,|\, H = W^\star \mathbf{X})$

$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$

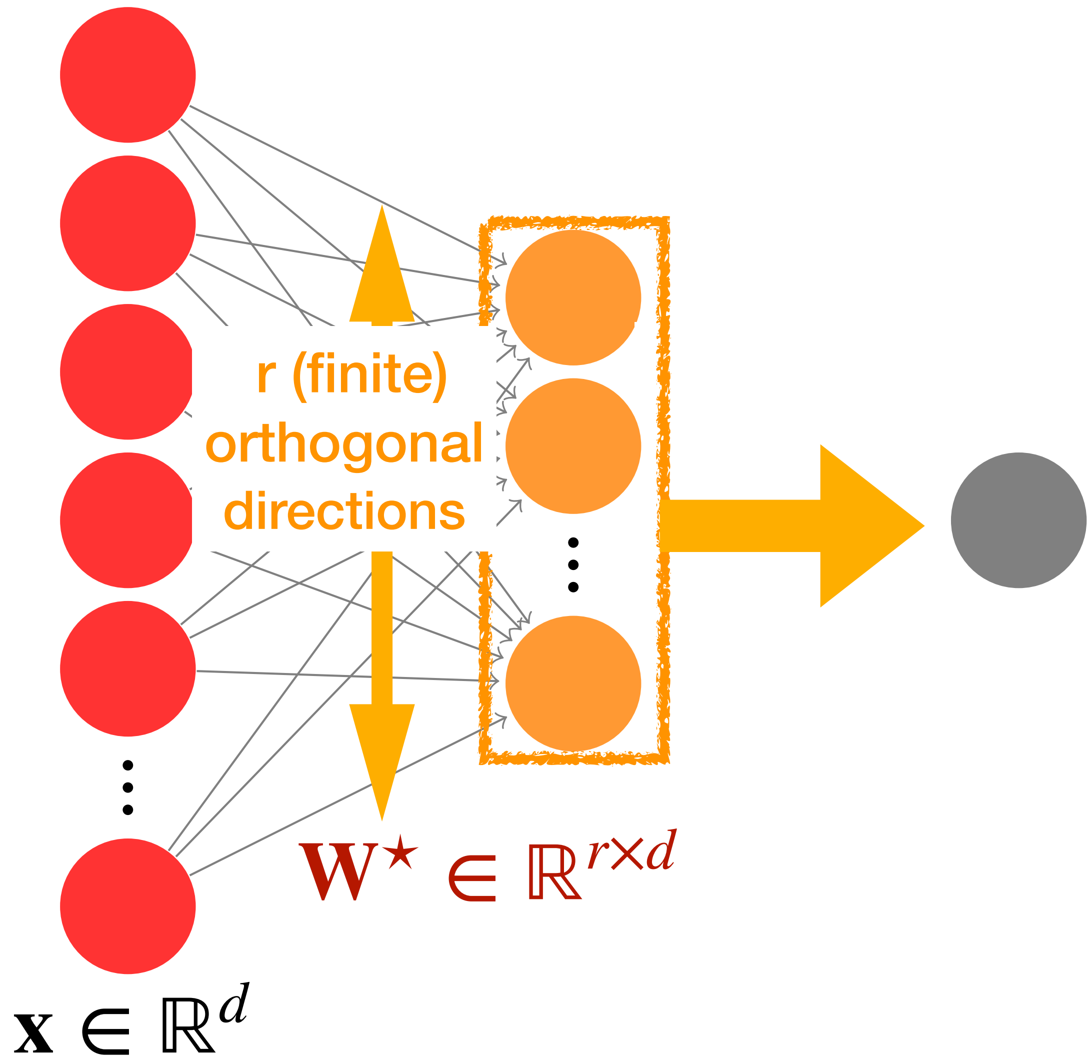


$\mathbf{x} \in \mathbb{R}^d$

Multi-index functions…

Target function: $Y \sim P^{\star}(Y \,|\, H = W^{\star}\mathbf{X})$

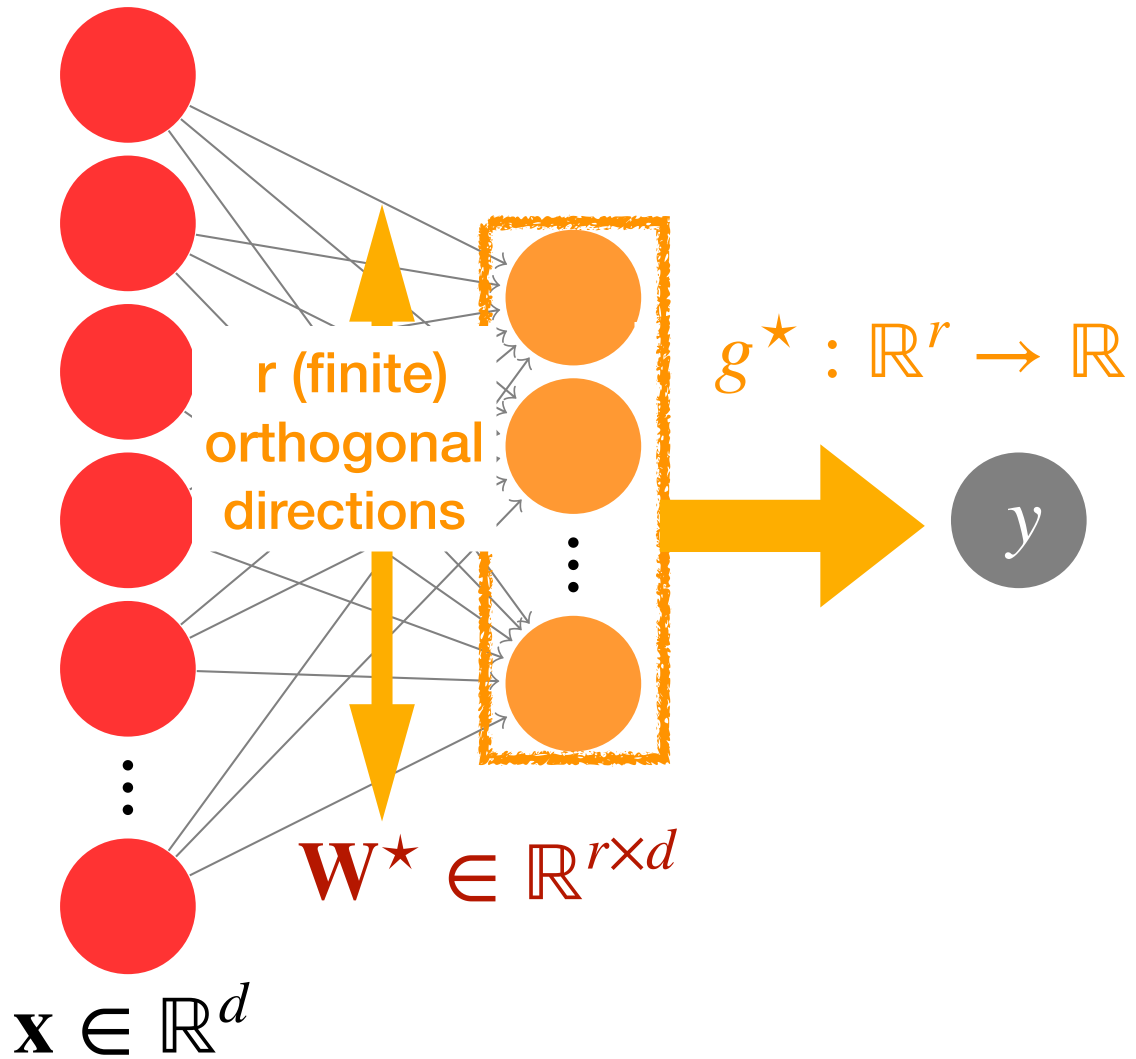
$$y = f^{\star}(\mathbf{x}) = g^{\star}(\mathbf{h}^{\star} = W^{\star}\mathbf{x})$$



Multi-index functions...

Target function: $Y \sim P^\star(Y \,|\, H = W^\star \mathbf{X})$

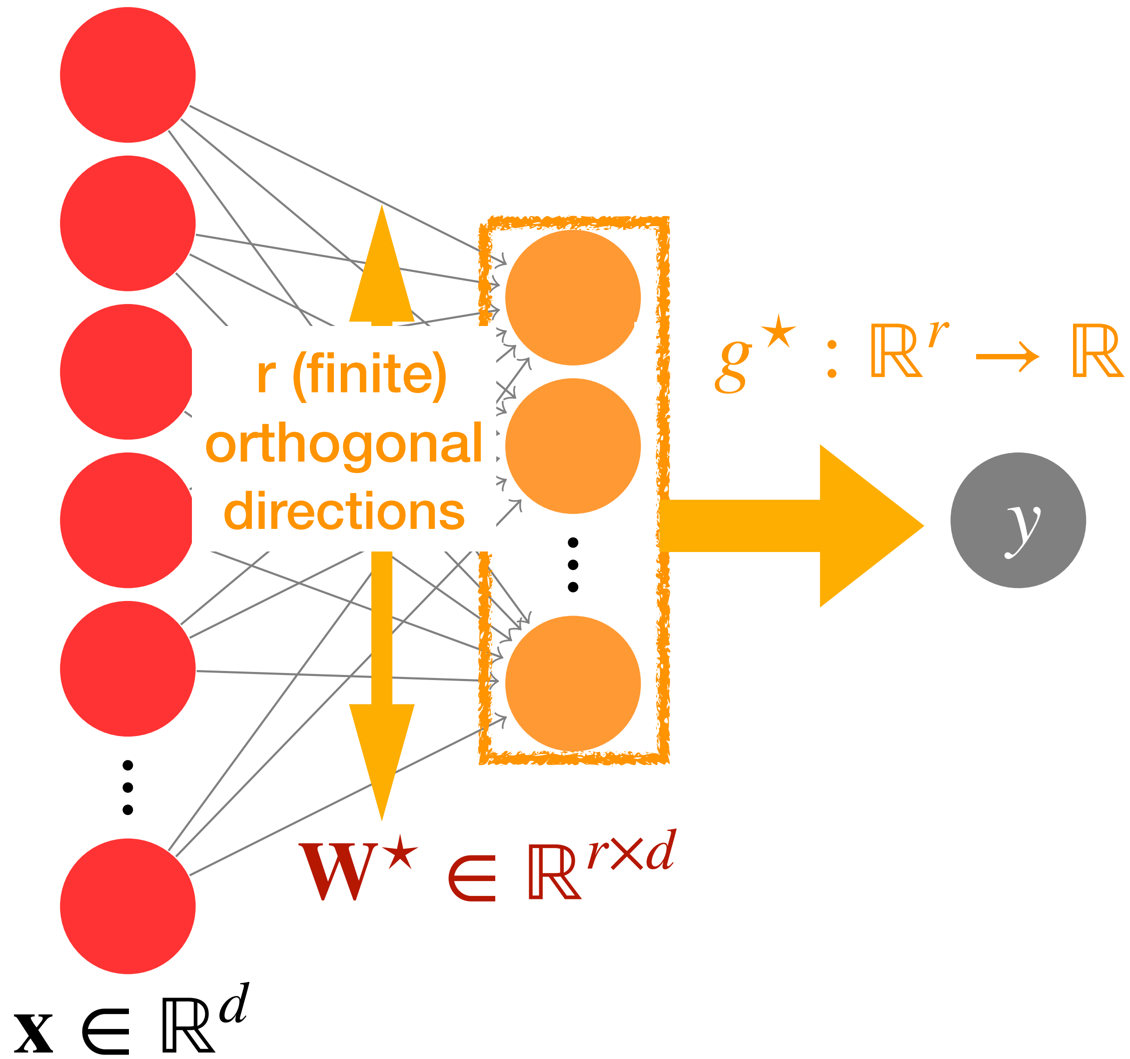
$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$



Multi-index functions…

Target function: $Y \sim P^\star(Y \,|\, H = W^\star \mathbf{X})$

$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$



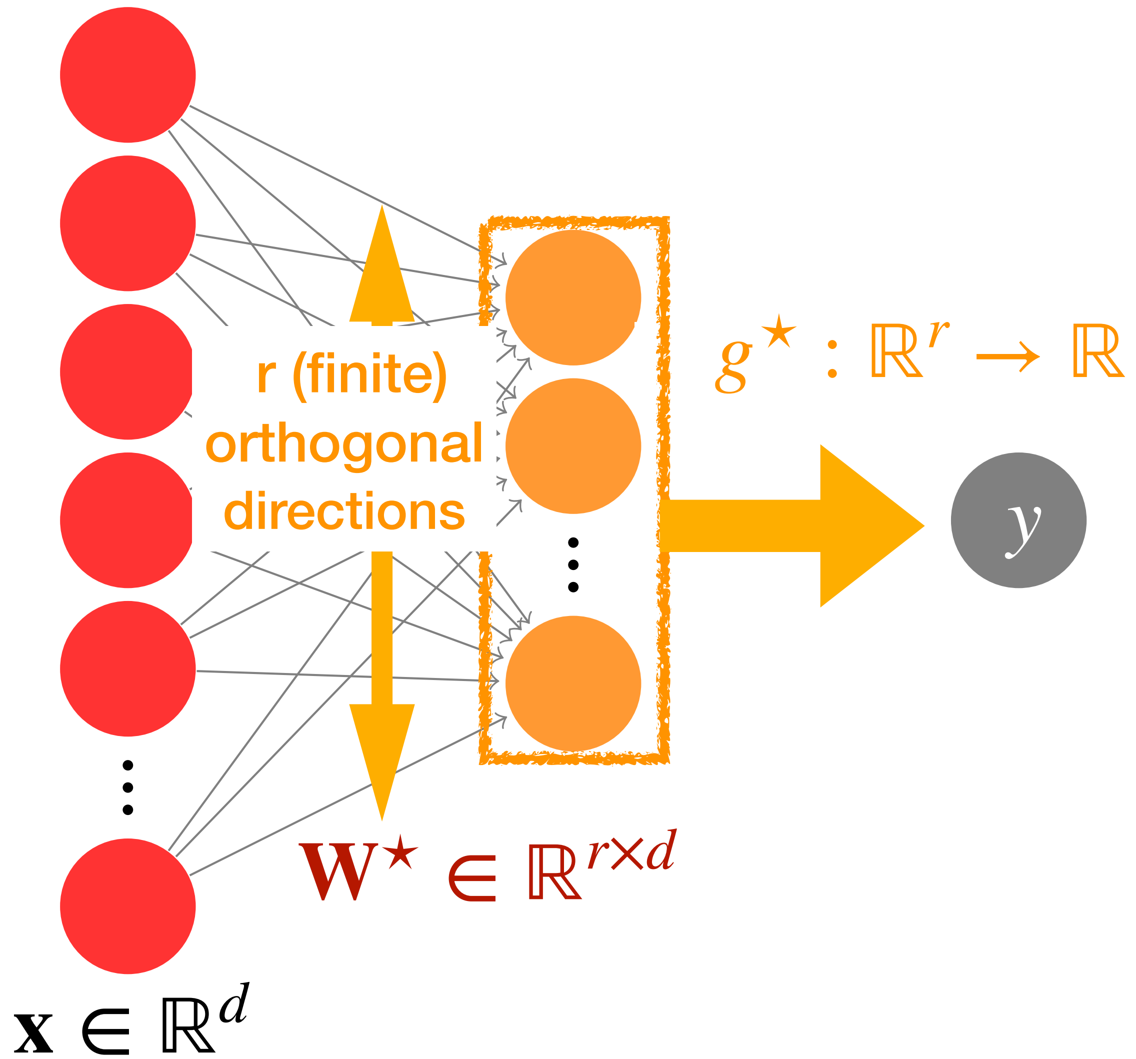
Single-index examples

$$y = g^\star(h^\star) \qquad h^\star = \mathbf{x} \cdot \mathbf{w}^\star$$

Multi-index functions…

Target function: $Y \sim P^\star(Y \,|\, H = W^\star \mathbf{X})$

$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$



Single-index examples

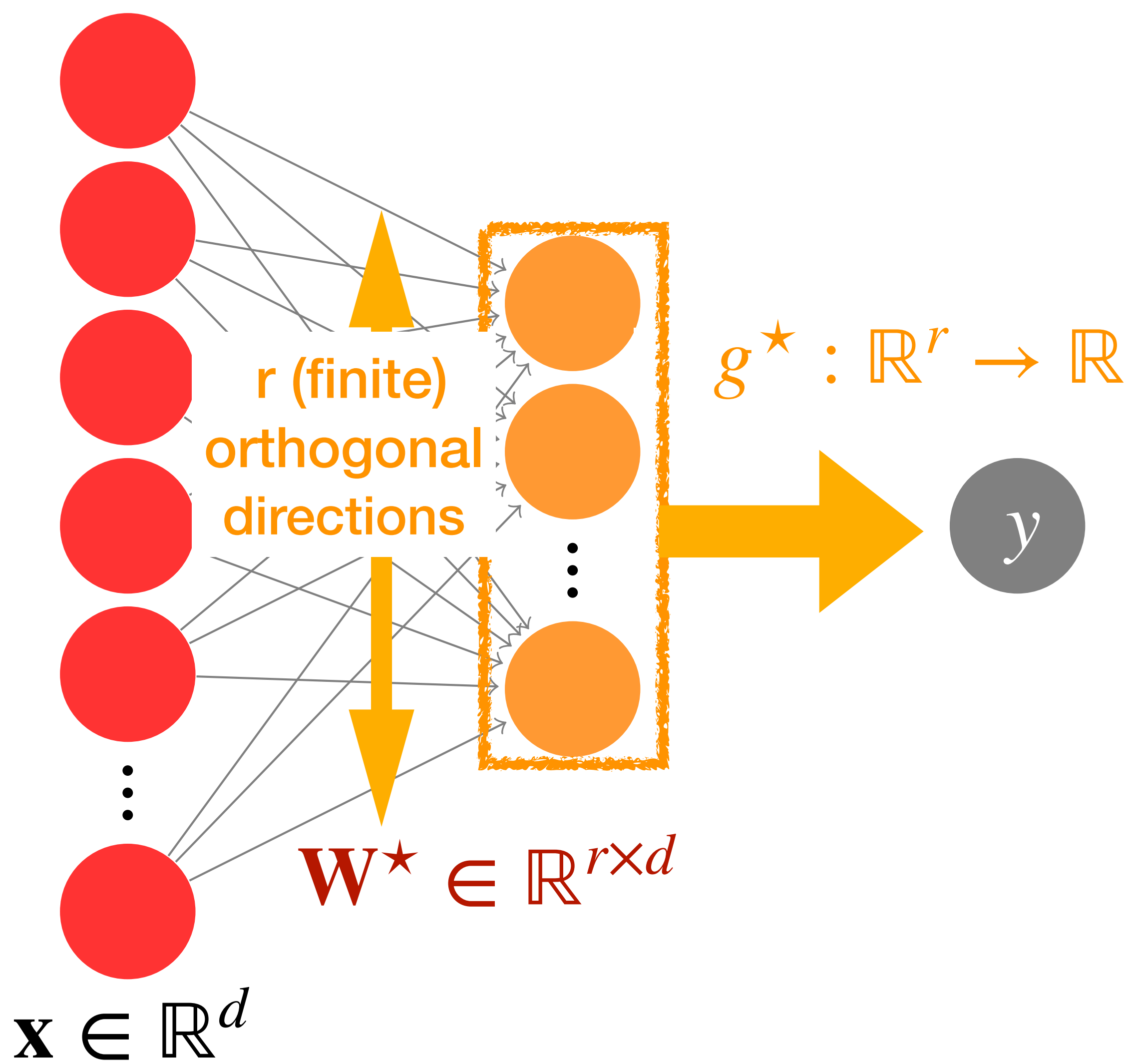
$$y = g^\star(h^\star) \qquad h^\star = \mathbf{x} \cdot \mathbf{w}^\star$$

- $f^\star(\mathbf{x}) = h^\star$

Multi-index functions…

Target function: $Y \sim P^\star(Y \mid H = W^\star \mathbf{X})$

$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$



Single-index examples

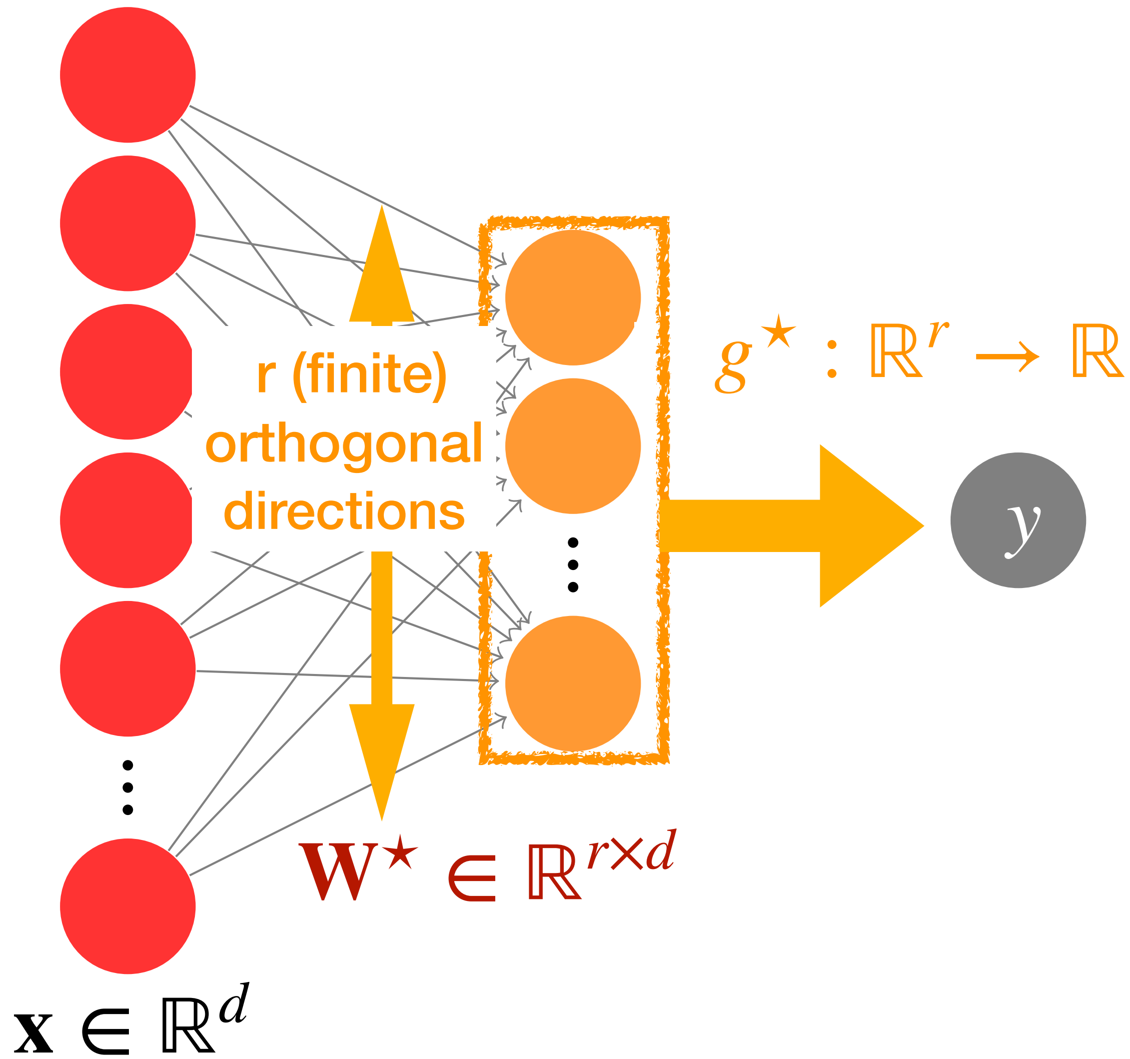
$$y = g^\star(h^\star) \qquad h^\star = \mathbf{x} \cdot \mathbf{w}^\star$$

- $f^\star(\mathbf{x}) = h^\star$
- $f^\star(\mathbf{x}) = |h^\star|$

Multi-index functions…

Target function: $Y \sim P^\star(Y|H = W^\star \mathbf{X})$

$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$



Single-index examples

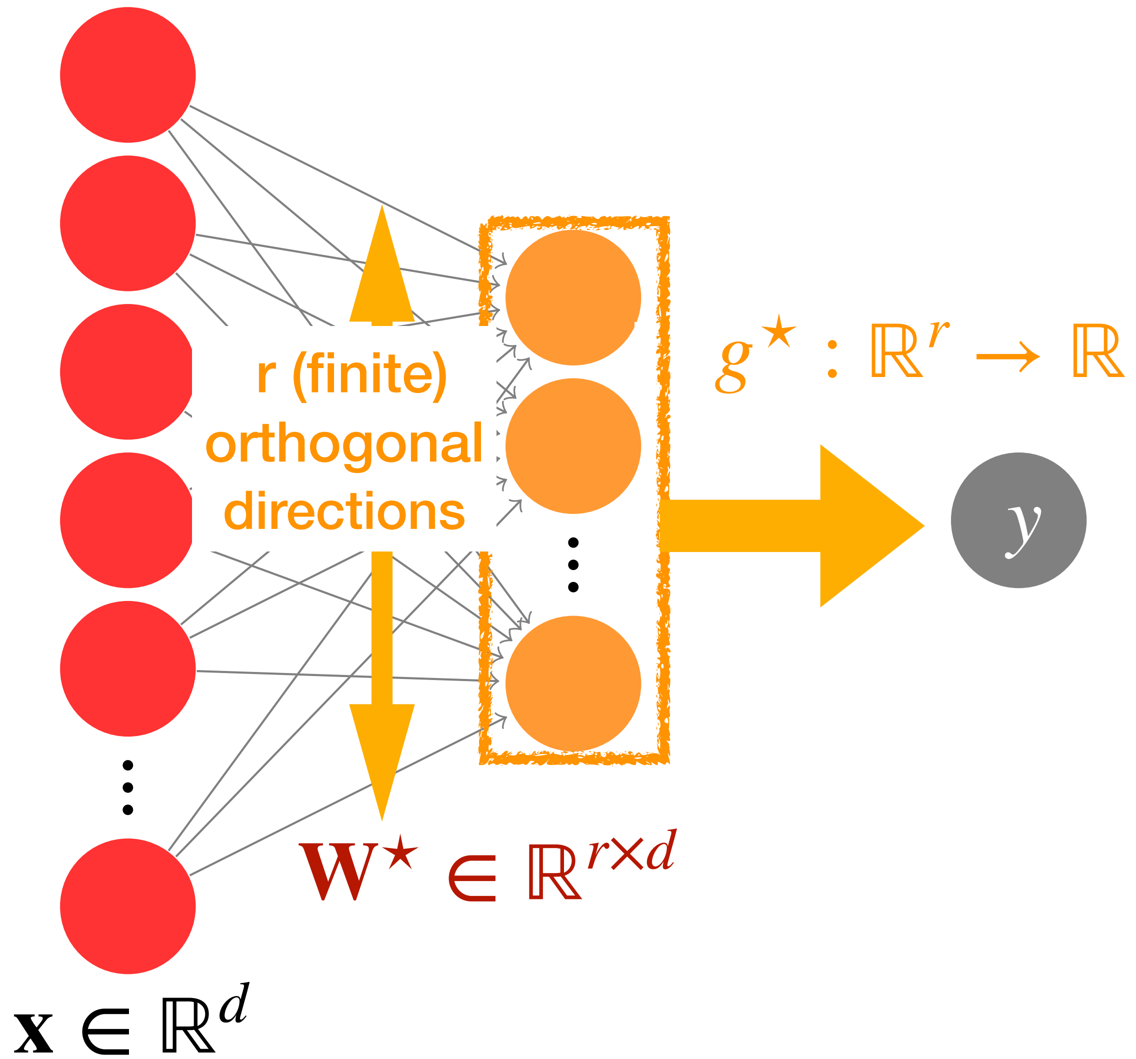
$y = g^\star(h^\star)$ $\quad h^\star = \mathbf{x} \cdot \mathbf{w}^\star$

- $f^\star(\mathbf{x}) = h^\star$
- $f^\star(\mathbf{x}) = |h^\star|$
- $f^\star(\mathbf{x}) = \mathrm{sign}(h^\star + \sqrt{\Delta} Z), Z \sim \mathcal{N}(0,1)$

Multi-index functions…

Target function: $Y \sim P^\star(Y | H = W^\star \mathbf{X})$

$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$



Single-index examples

$y = g^\star(h^\star)$ $\quad h^\star = \mathbf{x} \cdot \mathbf{w}^\star$

- $f^\star(\mathbf{x}) = h^\star$
- $f^\star(\mathbf{x}) = |h^\star|$
- $f^\star(\mathbf{x}) = \mathrm{sign}(h^\star + \sqrt{\Delta} Z), Z \sim \mathcal{N}(0,1)$

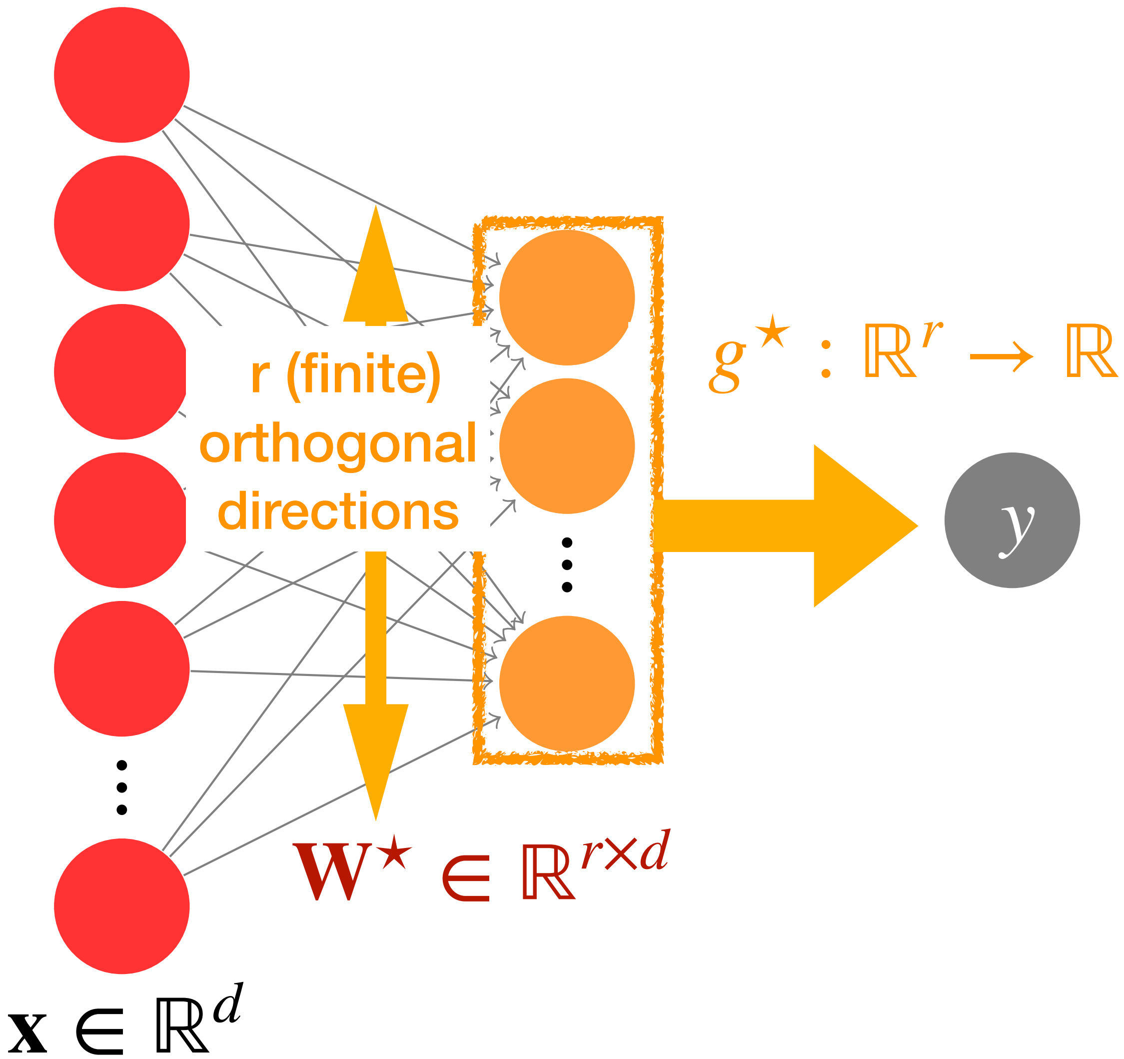
Multi-index examples

$y = g^\star(h_1^\star, h_2^\star, h_3^\star, \ldots, h_r^\star)$ $\quad h_i^\star = \mathbf{x} \cdot \mathbf{w}_i^\star$

Multi-index functions…

Target function: $Y \sim P^\star(Y|H = W^\star \mathbf{X})$

$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$



Single-index examples

$y = g^\star(h^\star)$ $\qquad h^\star = \mathbf{x} \cdot \mathbf{w}^\star$

- $f^\star(\mathbf{x}) = h^\star$
- $f^\star(\mathbf{x}) = |h^\star|$
- $f^\star(\mathbf{x}) = \mathrm{sign}(h^\star + \sqrt{\Delta} Z), Z \sim \mathcal{N}(0,1)$

Multi-index examples

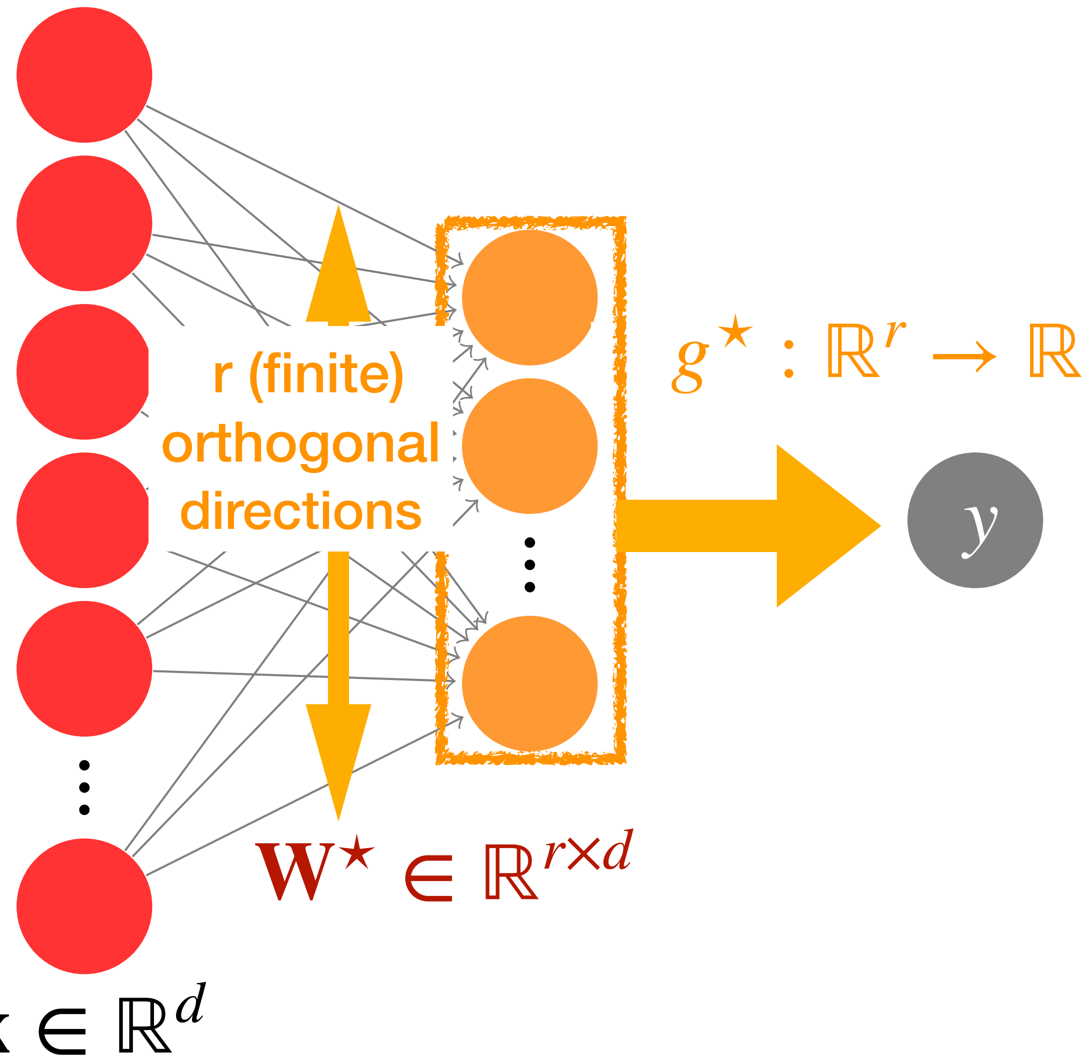
$y = g^\star(h_1^\star, h_2^\star, h_3^\star, \ldots, h_r^\star)$ $\quad h_i^\star = \mathbf{x} \cdot \mathbf{w}_i^\star$

- $f^\star(\mathbf{x}) = h_1^\star + |h_2^\star|$

Multi-index functions…

Target function: $Y \sim P^\star(Y|H = W^\star \mathbf{X})$

$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$



Single-index examples

$$y = g^\star(h^\star) \qquad h^\star = \mathbf{x} \cdot \mathbf{w}^\star$$

- $f^\star(\mathbf{x}) = h^\star$
- $f^\star(\mathbf{x}) = |h^\star|$
- $f^\star(\mathbf{x}) = \mathrm{sign}(h^\star + \sqrt{\Delta} Z), Z \sim \mathcal{N}(0,1)$

Multi-index examples

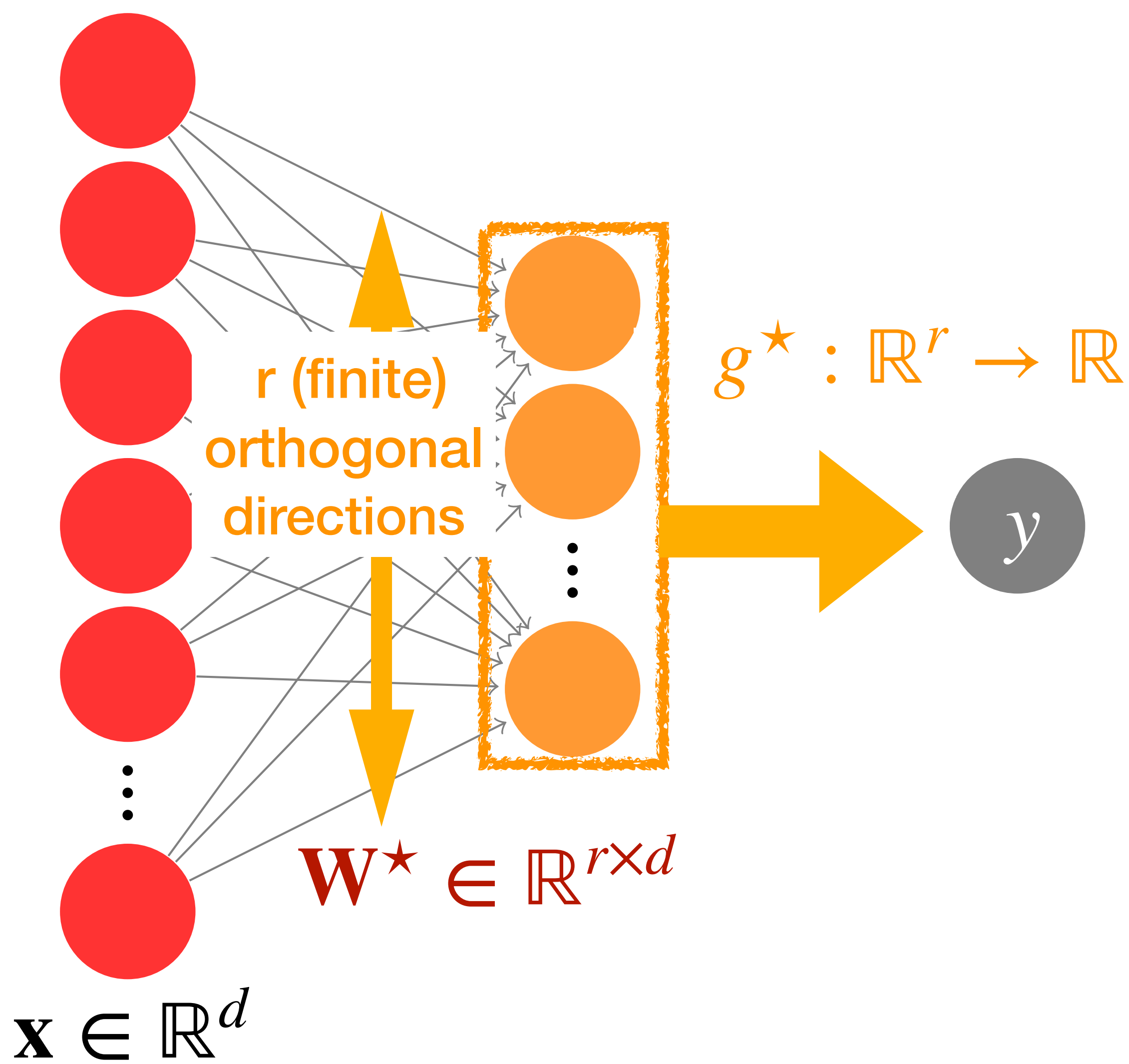
$$y = g^\star(h_1^\star, h_2^\star, h_3^\star, \ldots, h_r^\star) \quad h_i^\star = \mathbf{x} \cdot \mathbf{w}_i^\star$$

- $f^\star(\mathbf{x}) = h_1^\star + |h_2^\star|$
- $f^\star(\mathbf{x}) = h_1^\star + 2h_2^\star + h_1^\star h_2^\star + 3(h_2^\star)^2$

Multi-index functions…

Target function: $Y \sim P^\star(Y|H = W^\star \mathbf{X})$

$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$



Single-index examples

$$y = g^\star(h^\star) \qquad h^\star = \mathbf{x} \cdot \mathbf{w}^\star$$

- $f^\star(\mathbf{x}) = h^\star$
- $f^\star(\mathbf{x}) = |h^\star|$
- $f^\star(\mathbf{x}) = \mathrm{sign}(h^\star + \sqrt{\Delta} Z), Z \sim \mathcal{N}(0,1)$

Multi-index examples

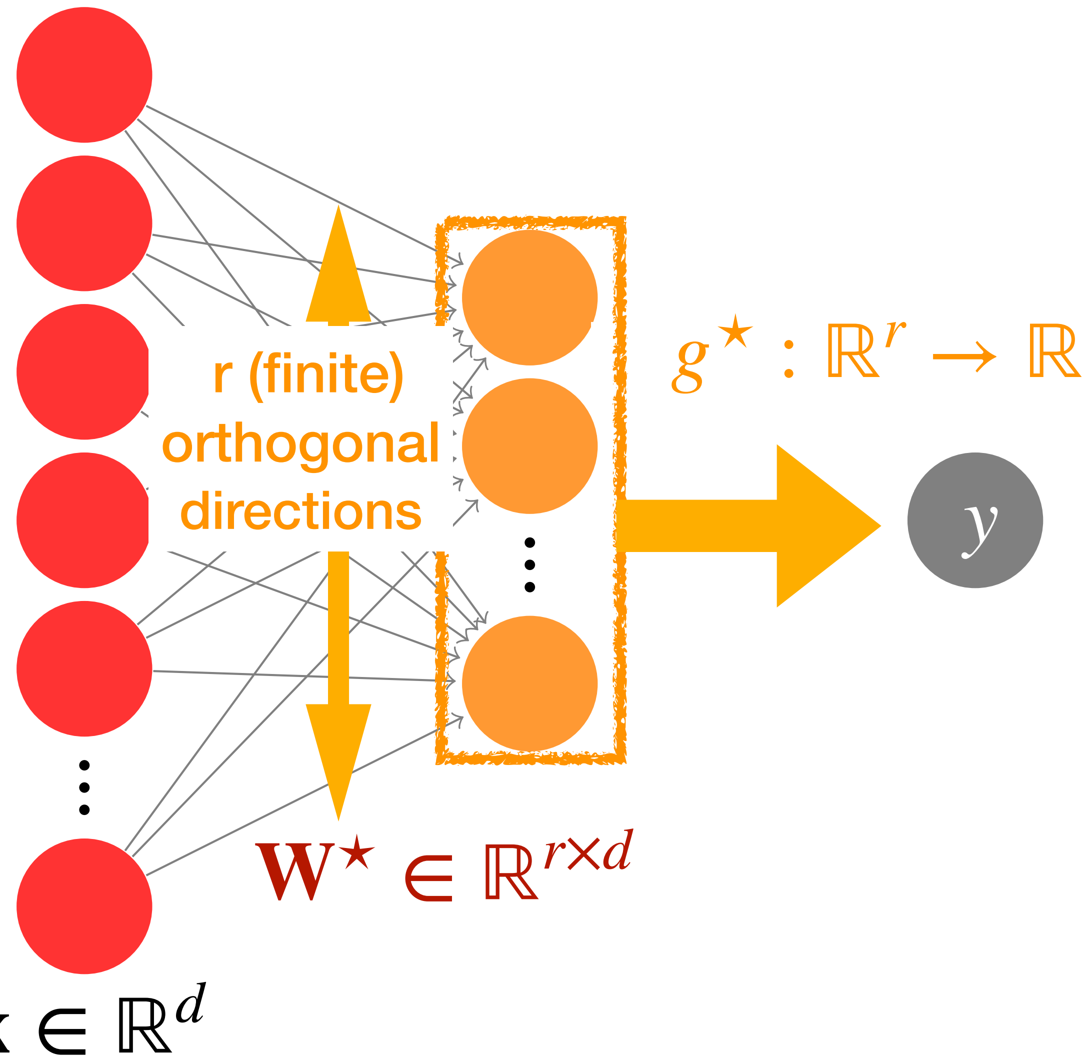
$$y = g^\star(h_1^\star, h_2^\star, h_3^\star, \ldots, h_r^\star) \quad h_i^\star = \mathbf{x} \cdot \mathbf{w}_i^\star$$

- $f^\star(\mathbf{x}) = h_1^\star + |h_2^\star|$
- $f^\star(\mathbf{x}) = h_1^\star + 2h_2^\star + h_1^\star h_2^\star + 3(h_2^\star)^2$
- $f^\star(\mathbf{x}) = \frac{1}{r}\sum_{i=1}^{r} \sigma(h_i^\star) + \sqrt{\Delta} Z$

Multi-index functions...

Target function: $Y \sim P^\star(Y|H = W^\star \mathbf{X})$

$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$



Single-index examples

$$y = g^\star(h^\star) \qquad h^\star = \mathbf{x} \cdot \mathbf{w}^\star$$

- $f^\star(\mathbf{x}) = h^\star$
- $f^\star(\mathbf{x}) = |h^\star|$
- $f^\star(\mathbf{x}) = \mathrm{sign}(h^\star + \sqrt{\Delta} Z), Z \sim \mathcal{N}(0,1)$

Multi-index examples

$$y = g^\star(h_1^\star, h_2^\star, h_3^\star, \dots, h_r^\star) \quad h_i^\star = \mathbf{x} \cdot \mathbf{w}_i^\star$$

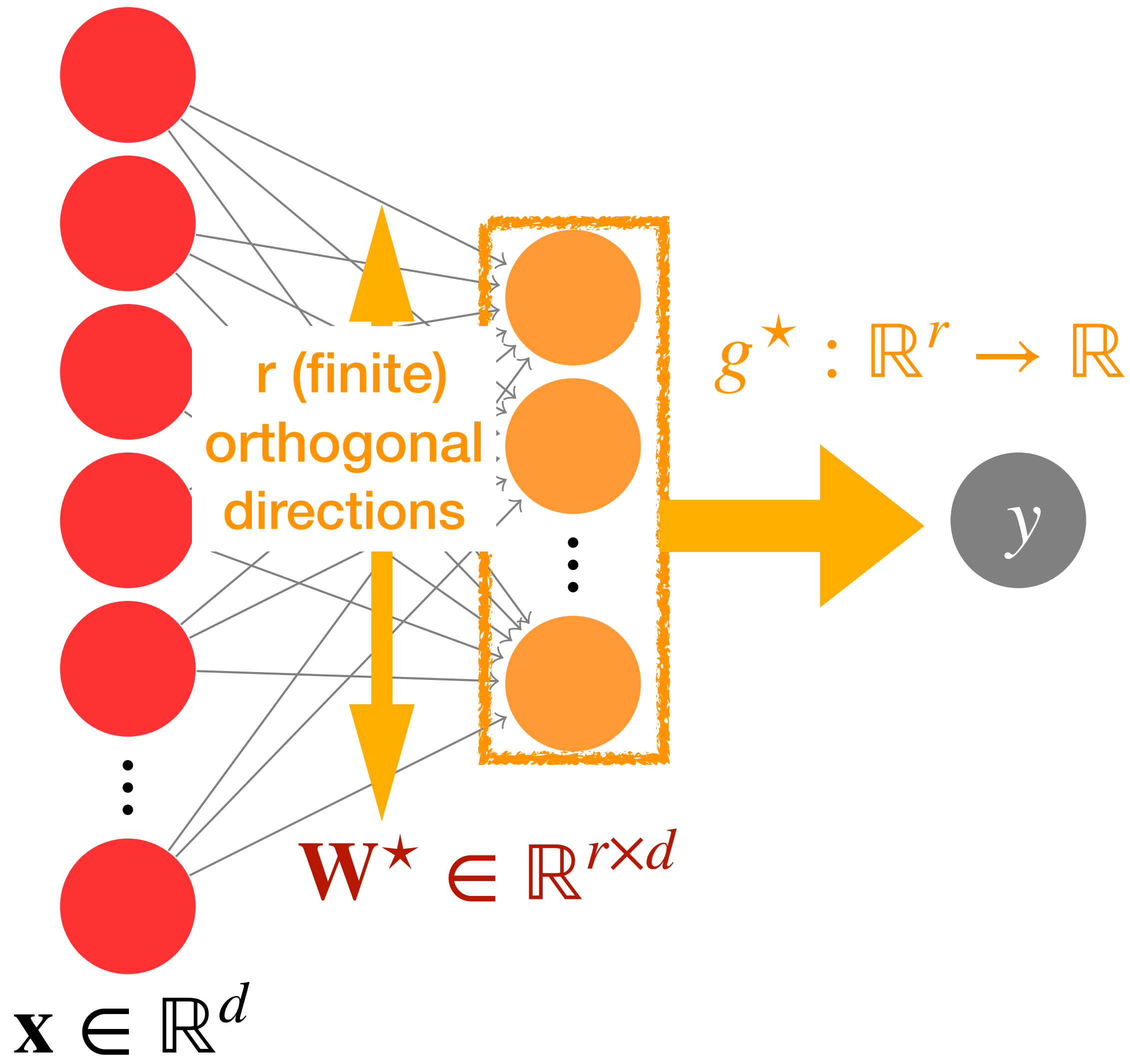
- $f^\star(\mathbf{x}) = h_1^\star + |h_2^\star|$
- $f^\star(\mathbf{x}) = h_1^\star + 2h_2^\star + h_1^\star h_2^\star + 3(h_2^\star)^2$
- $f^\star(\mathbf{x}) = \frac{1}{r}\sum_{i=1}^{r} \sigma(h_i^\star) + \sqrt{\Delta} Z$

Dataset $\mathcal{D} = \{\mathbf{x}_\nu, y_\nu = f^\star(\mathbf{x})\}_{\nu=1}^n$, Gaussian data $\mathbf{x}^\nu \sim \mathcal{N}(0,\mathbf{1_d})$, High-d limit $d \to \infty$

Multi-index functions…

Target function: $Y \sim P^\star(Y|H = W^\star \mathbf{X})$

$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$



Single-index examples

$$y = g^\star(h^\star) \qquad h^\star = \mathbf{x} \cdot \mathbf{w}^\star$$

- $f^\star(\mathbf{x}) = h^\star$
- $f^\star(\mathbf{x}) = |h^\star|$
- $f^\star(\mathbf{x}) = \mathrm{sign}(h^\star + \sqrt{\Delta} Z), Z \sim \mathcal{N}(0,1)$

Multi-index examples

$$y = g^\star(h_1^\star, h_2^\star, h_3^\star, \dots, h_r^\star) \quad h_i^\star = \mathbf{x} \cdot \mathbf{w}_i^\star$$

- $f^\star(\mathbf{x}) = h_1^\star + |h_2^\star|$
- $f^\star(\mathbf{x}) = h_1^\star + 2h_2^\star + h_1^\star h_2^\star + 3(h_2^\star)^2$
- $f^\star(\mathbf{x}) = \frac{1}{r}\sum_{i=1}^{r} \sigma(h_i^\star) + \sqrt{\Delta} Z$

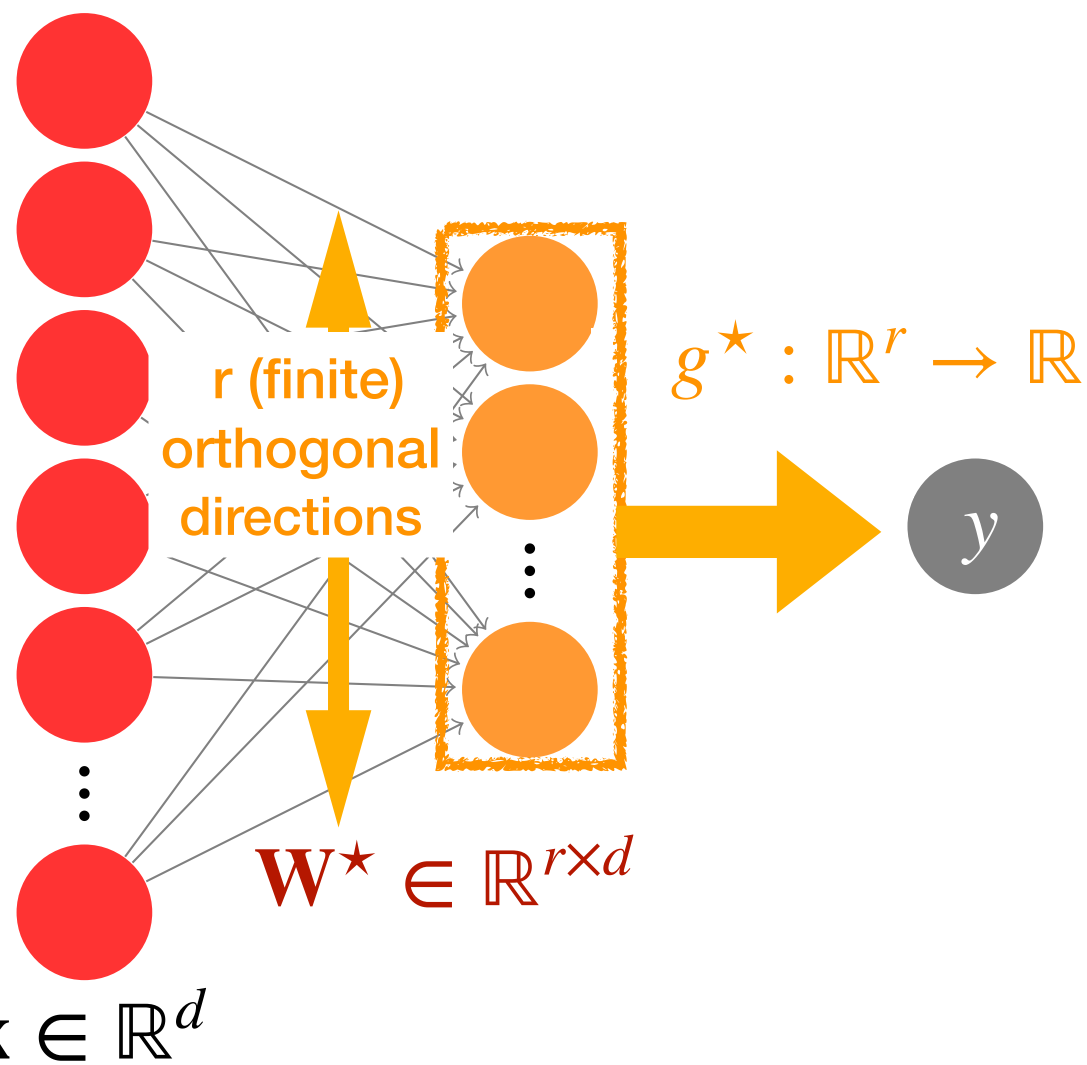
Dataset $\mathcal{D} = \{\mathbf{x}_\nu, y_\nu = f^\star(\mathbf{x})\}_{\nu=1}^n$, Gaussian data $\mathbf{x}^\nu \sim \mathcal{N}(0,\mathbf{1_d})$, High-d limit $d \to \infty$

… can we learn these functions from data?

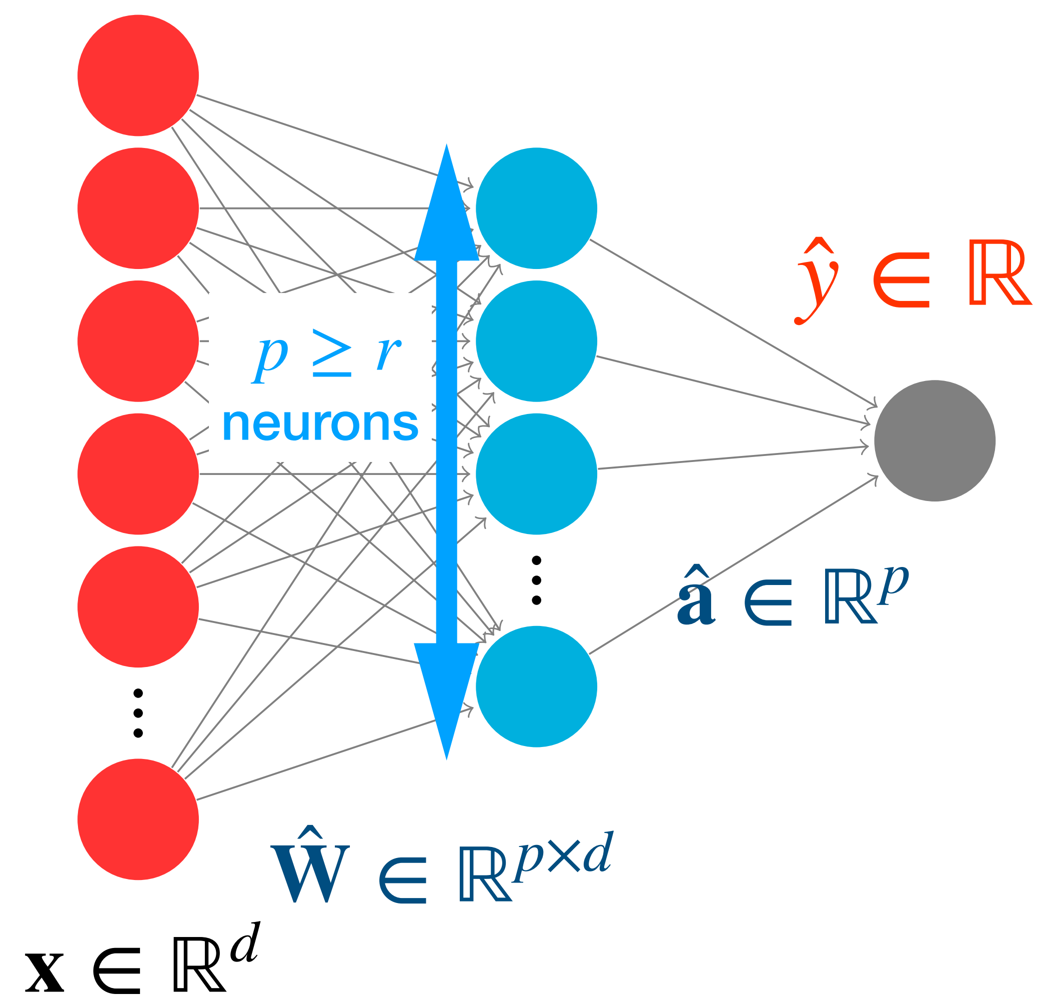
Architecture: A two-layer neural net

Target function: $Y \sim P^\star(Y | H = W^\star \mathbf{X})$

$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$



$$\hat{y} = \hat{f}(\mathbf{x}) = \sum_{i=1}^{p} \hat{a}_i \sigma_i(\langle \hat{\mathbf{w}}_\mathbf{i}, \mathbf{x} \rangle)$$



Dataset $\mathscr{D} = \{\mathbf{x}_\nu, y_\nu = f^\star(\mathbf{x})\}_{\nu=1}^n$, Gaussian data $\mathbf{x}^\nu \sim \mathcal{N}(0,\mathbf{1_d})$, High-d limit $d \to \infty$

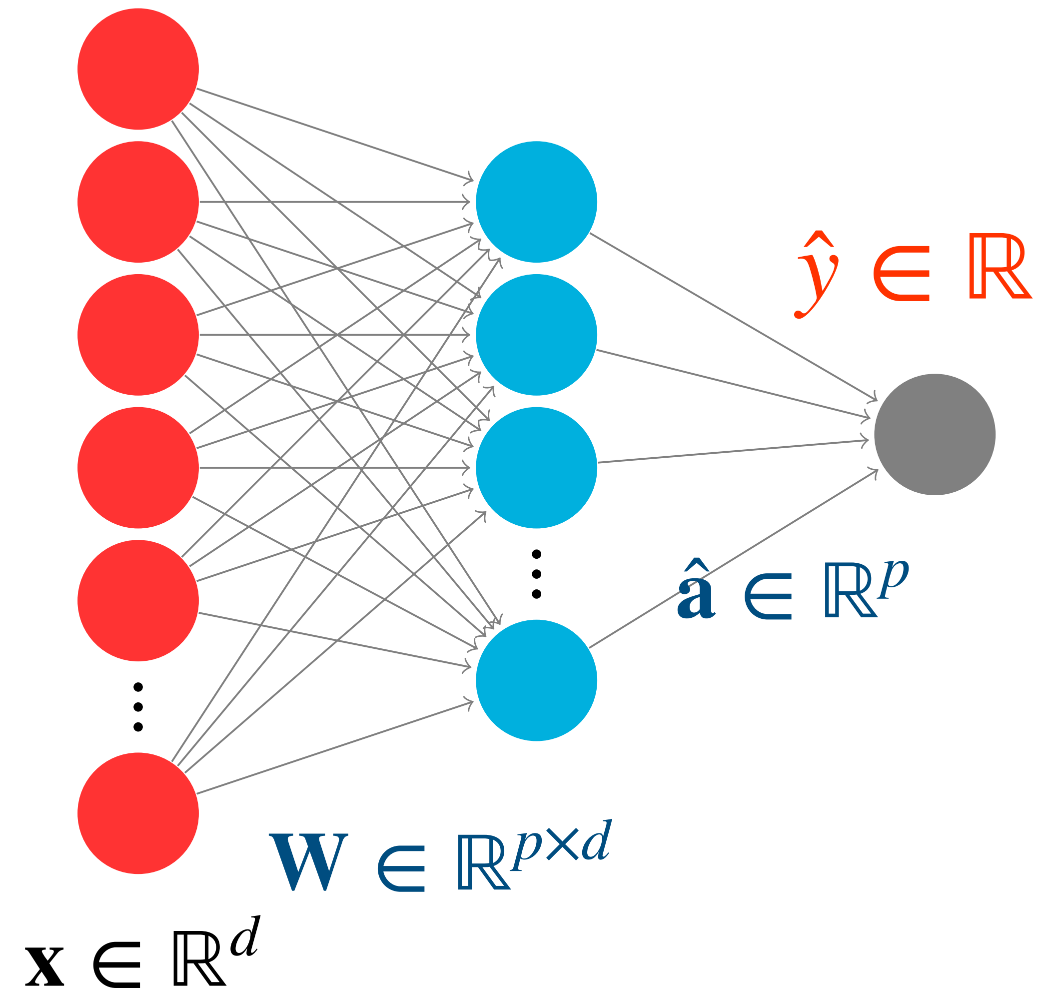
… can we learn these functions from data?

Lazy approach: not training the first layer

Random features

[Balcan,Blum, Vempala '06, Rahimi-Recht '17…]

$$\hat{y} = \hat{f}(\mathbf{x}) = \hat{\mathbf{a}} \cdot \sigma(W\mathbf{x})$$



No training of the first layer: W is fixed

$$\hat{y} = \hat{f}(\mathbf{x}) = \sum_{i=1}^{p} \hat{a}_i \sigma_i(\langle \mathbf{w}_i, \mathbf{x} \rangle) = \sum_{i=1}^{p} \hat{a}_i \Phi_{\mathrm{CK}}(\mathbf{x})$$

Computationally easy (linear regression)

Very popular setting among theoreticians

Equivalent to neural Tangent Kernel/Lazy Regime/Kernel methods/ etc..

[Jacot, Gabriel, Hongler '18; Lee, Jaehoon, et al. 18; Chizat, Bach '19,…]

Unfortunately: very limited

Theorem (Informal) [Mei, Misiakiewicz, Montanari '22]

In *absence* of feature learning (i.e. *at initialisation* when the first layer is *fully random*) one can only learn a ***polynomial* approximation of** $f^\star$ **of degree** κ as long as $\min(n,p) = O(d^\kappa)$

$$f^\star(\mathbf{x}) = \text{cst} + \sum_i \mu_i^{(1)} h_i^\star + \sum_{ij} \mu_{ij}^{(2)} h_i^\star h_j^\star + \sum_{ijk} \mu_{ijk}^{(3)} h_i^\star h_j^\star h_k^\star + \dots$$

See also [El Karaoui '10; Mei-Montanari '19; Gerace, Loureiro, **FK**, Mézard, Zdeborová '20; Jacot, Simsek, Spadaro, Hongler, Gabriel '20; Hu, Lu, '20; Dhifallah, Lu '20; Loureiro, Gerbelot, Cui, Goldt, **FK**, Mézard, Zdeborová '21; Montanari & Saeed '22; Xiao, Hu, Misiakiewicz, Lu, Pennington '22; Dandi, Stephan, **FK**, Loureiro, Zdeborová '23; Aguirre-López, Franz, Pastore '24]

Unfortunately: very limited

Theorem (Informal) [Mei, Misiakiewicz, Montanari '22]

In *absence* of feature learning (i.e. *at initialisation* when the first layer is *fully random*) one can only learn a ***polynomial* approximation of $f^\star$ of degree** κ as long as $\min(n,p) = O(d^\kappa)$

$$f^\star(\mathbf{x}) = \mathrm{cst} + \sum_i \mu_i^{(1)} h_i^\star + \sum_{ij} \mu_{ij}^{(2)} h_i^\star h_j^\star + \sum_{ijk} \mu_{ijk}^{(3)} h_i^\star h_j^\star h_k^\star + \dots$$

$(n,p) = O(d)$

See also [El Karaoui '10; Mei-Montanari '19; Gerace, Loureiro, **FK**, Mézard, Zdeborová '20; Jacot, Simsek, Spadaro, Hongler, Gabriel '20; Hu, Lu, '20; Dhifallah, Lu '20; Loureiro, Gerbelot, Cui, Goldt, **FK**, Mézard, Zdeborová '21; Montanari & Saeed '22; Xiao, Hu, Misiakiewicz, Lu, Pennington '22; Dandi, Stephan, **FK**, Loureiro, Zdeborová '23; Aguirre-López, Franz, Pastore '24]

Unfortunately: very limited

Theorem (Informal) [Mei, Misiakiewicz, Montanari '22]

In *absence* of feature learning (i.e. *at initialisation* when the first layer is *fully random*) one can only learn a ***polynomial* approximation of $f^\star$ of degree κ** as long as $\min(n,p) = O(d^\kappa)$

$$f^\star(\mathbf{x}) = \mathrm{cst} + \sum_i \mu_i^{(1)} h_i^\star + \sum_{ij} \mu_{ij}^{(2)} h_i^\star h_j^\star + \sum_{ijk} \mu_{ijk}^{(3)} h_i^\star h_j^\star h_k^\star + \ldots$$

$(n,p) = O(d)$ $\qquad$ $(n,p) = O(d^2)$

See also [El Karaoui '10; Mei-Montanari '19; Gerace, Loureiro, **FK**, Mézard, Zdeborová '20; Jacot, Simsek, Spadaro, Hongler, Gabriel '20; Hu, Lu, '20; Dhifallah, Lu '20; Loureiro, Gerbelot, Cui, Goldt, **FK**, Mézard, Zdeborová '21; Montanari & Saeed '22; Xiao, Hu, Misiakiewicz, Lu, Pennington '22; Dandi, Stephan, **FK**, Loureiro, Zdeborová '23; Aguirre-López, Franz, Pastore '24]

Unfortunately: very limited

Theorem (Informal) [Mei, Misiakiewicz, Montanari '22]

In *absence* of feature learning (i.e. *at initialisation* when the first layer is *fully random*) one can only learn a ***polynomial* approximation of $f^\star$ of degree** κ as long as $\min(n,p) = O(d^\kappa)$

$$f^\star(\mathbf{x}) = \text{cst} + \sum_i \mu_i^{(1)} h_i^\star + \sum_{ij} \mu_{ij}^{(2)} h_i^\star h_j^\star + \sum_{ijk} \mu_{ijk}^{(3)} h_i^\star h_j^\star h_k^\star + \dots$$

$(n,p) = O(d)$ $\quad$ $(n,p) = O(d^2)$ $\quad$ $(n,p) = O(d^3)$

See also [El Karaoui '10; Mei-Montanari '19; Gerace, Loureiro, **FK**, Mézard, Zdeborová '20; Jacot, Simsek, Spadaro, Hongler, Gabriel '20; Hu, Lu, '20; Dhifallah, Lu '20; Loureiro, Gerbelot, Cui, Goldt, **FK**, Mézard, Zdeborová '21; Montanari & Saeed '22; Xiao, Hu, Misiakiewicz, Lu, Pennington '22; Dandi, Stephan, **FK**, Loureiro, Zdeborová '23; Aguirre-López, Franz, Pastore '24]

Unfortunately: very limited

For Gaussian data,
lazy training is just polynomial
fitting in disguise

Theorem (Informal) [Mei, Misiakiewicz, Montanari '22]

In *absence* of feature learning (i.e. *at initialisation* when the first layer is *fully random*) one can only learn a ***polynomial*** **approximation of** $f^\star$ **of degree** κ as long as $\min(n, p) = O(d^\kappa)$

$$f^\star(\mathbf{x}) = \mathrm{cst} + \sum_i \mu_i^{(1)} h_i^\star + \sum_{ij} \mu_{ij}^{(2)} h_i^\star h_j^\star + \sum_{ijk} \mu_{ijk}^{(3)} h_i^\star h_j^\star h_k^\star + \dots$$

$(n, p) = O(d)$ $\quad$ $(n, p) = O(d^2)$ $\quad$ $(n, p) = O(d^3)$

See also [El Karaoui '10; Mei-Montanari '19; Gerace, Loureiro, **FK**, Mézard, Zdeborová '20; Jacot, Simsek, Spadaro, Hongler, Gabriel '20; Hu, Lu, '20; Dhifallah, Lu '20; Loureiro, Gerbelot, Cui, Goldt, **FK**, Mézard, Zdeborová '21; Montanari & Saeed '22; Xiao, Hu, Misiakiewicz, Lu, Pennington '22; Dandi, Stephan, **FK**, Loureiro, Zdeborová '23; Aguirre-López, Franz, Pastore '24]

A single gradient step can change the story

$$\hat{W}^{t=1} = \hat{W}^{t=0} - \frac{\eta}{2n} \nabla_W \left(\sum_{\mu} (y_\mu - \hat{f}_{\hat{W}^{t=1}}(\mathbf{x}_\mu))^2 \right)$$

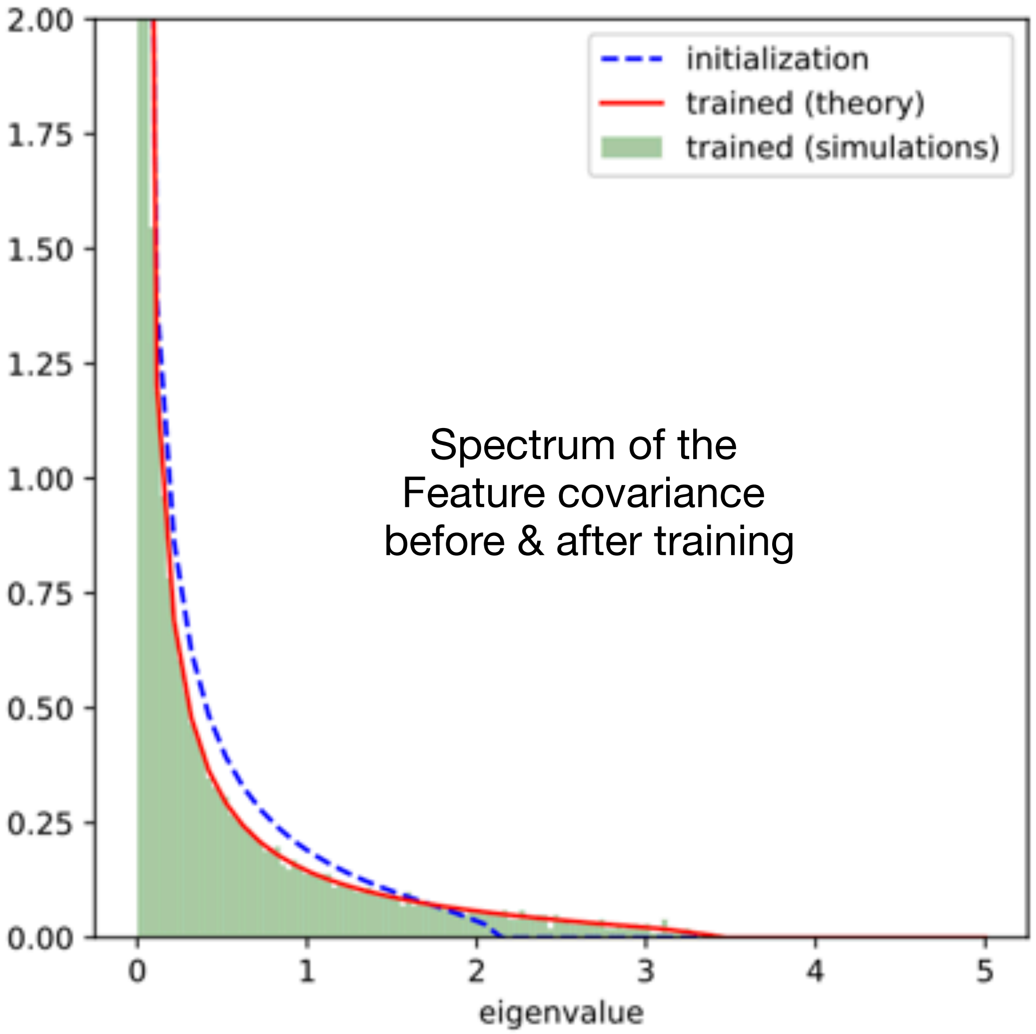
[Damian, Lee, Soltanolkotabi ’22,Ba, Erdogdu, Suzuki, Wang, Wu, Yang ’22; Moniri et al ’23]

A single gradient step can change the story

$$\hat{W}^{t=1} = \hat{W}^{t=0} - \frac{\eta}{2n}\nabla_W\left(\sum_\mu (y_\mu - \hat{f}_{\hat{W}^{t=1}}(\mathbf{x}_\mu))^2\right)$$

Single index model $y = \sin(h^\star)$

$\eta = O(d)$ (Maximal Update parametrization [Yang et al., 2022])

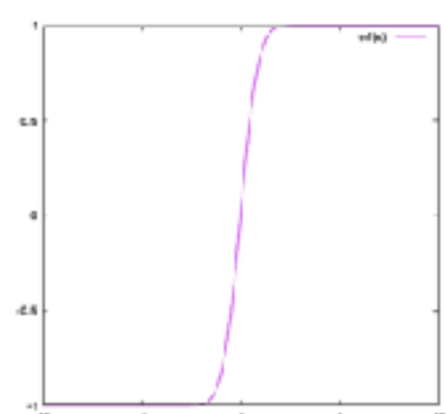


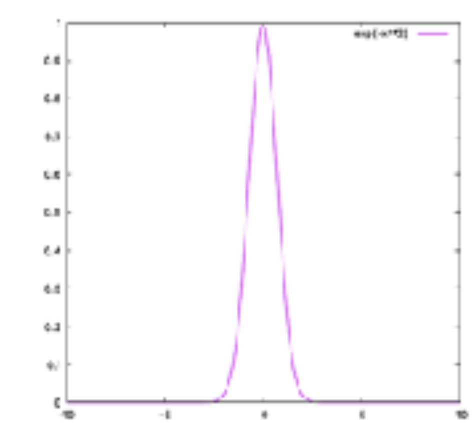
- Long tail in the spectrum of feature covariance
  (+ large outlying eigenvalue, not represented).

- Ties in with numerous previous empirical observations on deep learning [Martin and Mahoney, 21, Martin et al., 21, Want et al '24]

- Drastic improvement of generalisation for single index models: Can fit the target function $g(h^\star)$ over a random (over $\mathbf{a}^0$) basis

$$\mu_0^i(\lambda) = \mathrm{erf}\left(a_i^{t=0}\frac{\lambda}{\sqrt{3}}\right) \quad \&. \quad \mu_1^i(\lambda) = e^{-3(a_i^{t=0}\lambda)^2}$$

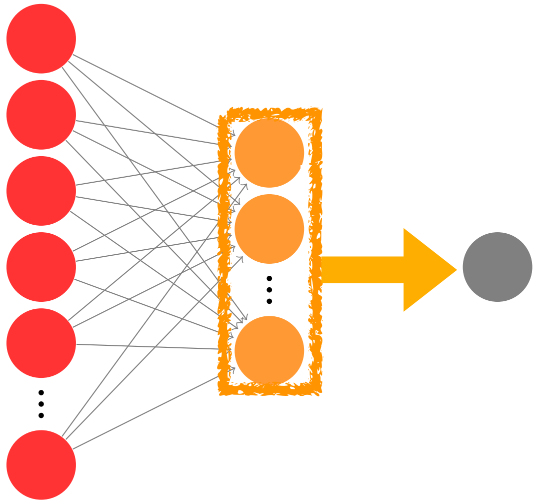


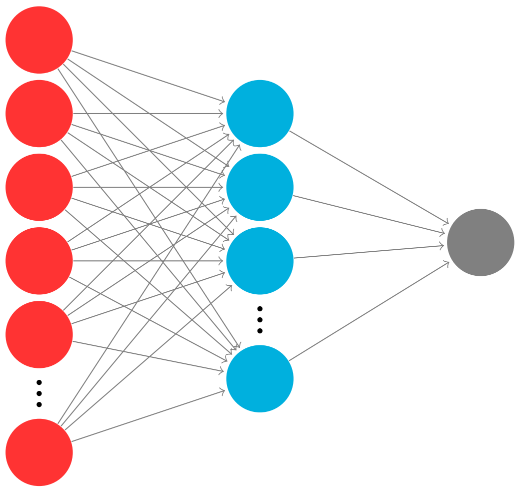


[Cui, Pesce, Dandi, **FK**, Lu, Zdeborová, Loureiro '24; Dandi, Pesce, Cui, **FK**, Lu, Loureiro '24]

Feature learning helps: a heuristic argument

$y = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$





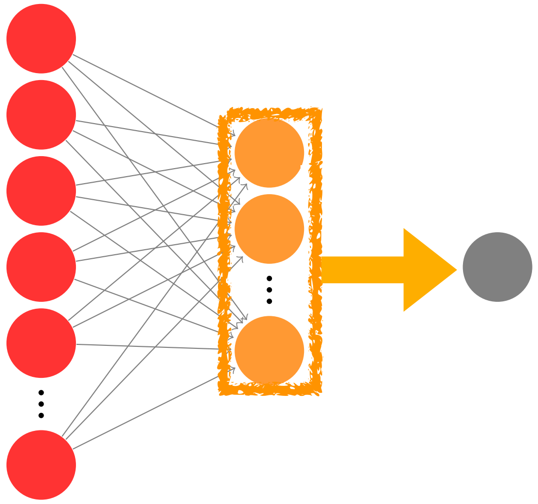
$\hat{y} = \hat{\mathbf{a}} \cdot \sigma(\hat{W}\mathbf{x})$

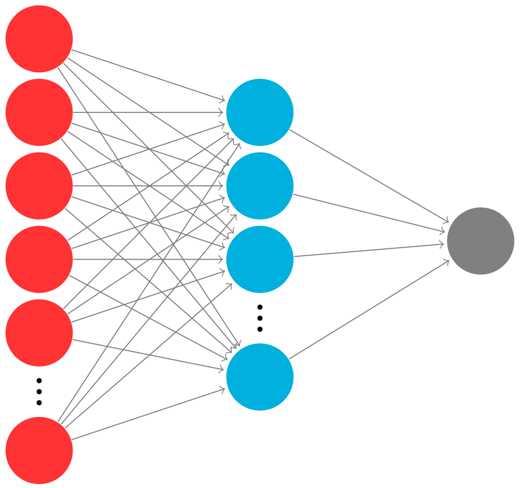
Assume $\hat{W}$ in the two layer correlates with *some* of target directions $\mathbf{h}_{//} \subset \mathbf{h}^\star$

What do we expect ?

Feature learning helps: a heuristic argument

$y = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$





$\hat{y} = \hat{\mathbf{a}} \cdot \sigma(\hat{W}\mathbf{x})$

Assume $\hat{W}$ in the two layer correlates with *some* of target directions $\mathbf{h}_{//} \subset \mathbf{h}^\star$

What do we expect ?

In the learned subspace

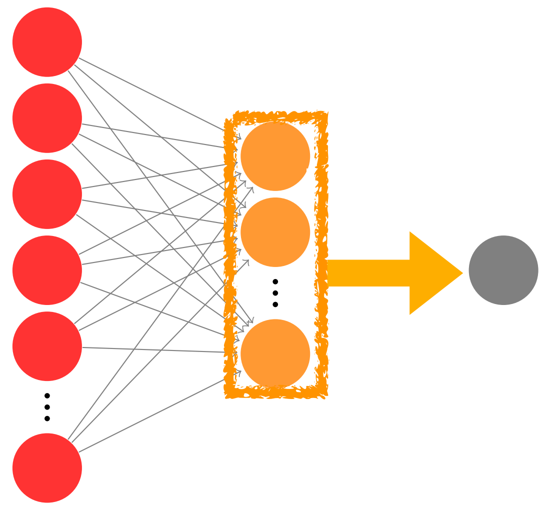
$$\hat{y} \approx \hat{\mathbf{a}} \cdot \sigma(\widetilde{W}\mathbf{h}_{//} + \text{noise})$$

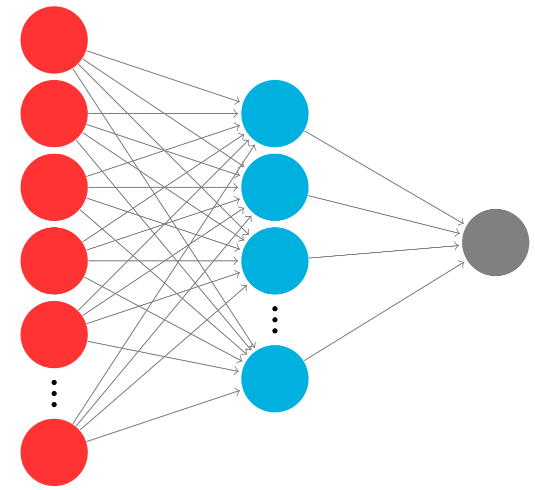
(Noisy) Random feature in (finite) reduced space $d^{\text{eff}} = r$

Can fit well the target function as long as p and n are large enough!

Feature learning helps: a heuristic argument

$$y = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$





$$\hat{y} = \hat{\mathbf{a}} \cdot \sigma(\hat{W}\mathbf{x})$$

Assume $\hat{W}$ in the two layer correlates with *some* of target directions $\mathbf{h}_{//} \subset \mathbf{h}^\star$

What do we expect ?

In the learned subspace

$$\hat{y} \approx \hat{\mathbf{a}} \cdot \sigma(\widetilde{W}\mathbf{h}_{//} + \text{noise})$$

(Noisy) Random feature in (finite) reduced space $d^{\text{eff}} = r$

Can fit well the target function as long as p and n are large enough!

In the not-learned subspace

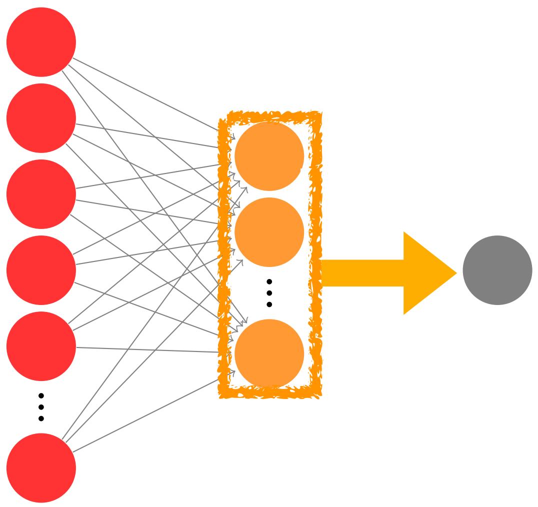
$$\hat{y} \approx \hat{\mathbf{a}} \cdot \sigma(\widetilde{W}\mathbf{x})$$

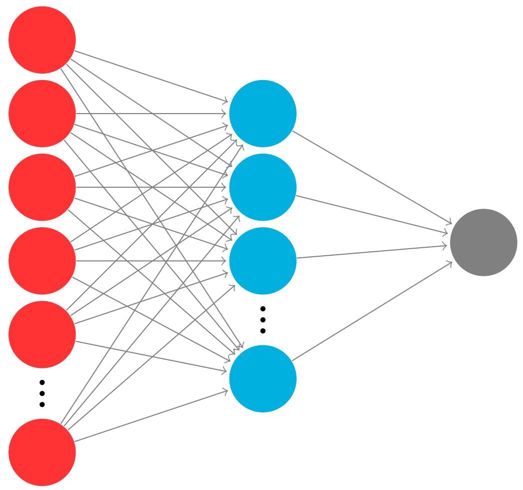
Random feature in dimension d

Cannot do better than a polynomial fit of degree κ with $\min(n, p) = O(d^\kappa)$

Feature learning helps: a heuristic argument

$y = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$





$\hat{y} = \hat{\mathbf{a}} \cdot \sigma(\hat{W}\mathbf{x})$

Assume $\hat{W}$ in the two layer correlates with *some* of target directions $\mathbf{h}_{//} \subset \mathbf{h}^\star$

What do we expect ?

In the learned subspace

$$\hat{y} \approx \hat{\mathbf{a}} \cdot \sigma(\widetilde{W}\mathbf{h}_{//} + \text{noise})$$

(Noisy) Random feature in (finite) reduced space $d^{\text{eff}} = r$

Can fit well the target function as long as p and n are large enough!

In the not-learned subspace

$$\hat{y} \approx \hat{\mathbf{a}} \cdot \sigma(\widetilde{W}\mathbf{x})$$

Random feature in dimension d

Cannot do better than a polynomial fit of degree κ with $\min(n, p) = O(d^\kappa)$

No generic proof, but this is the behaviour typically observed. Precise rigorous statement in e.g.: [Chen at al '20+21, Damian, Lee, Soltanolkotabi '22,Ba, Erdogdu, Suzuki, Wang, Wu, Yang '22, Abbe, Boix-Adsera, and Misiakiewicz '22+'23, Dandi et at '23 + '24]

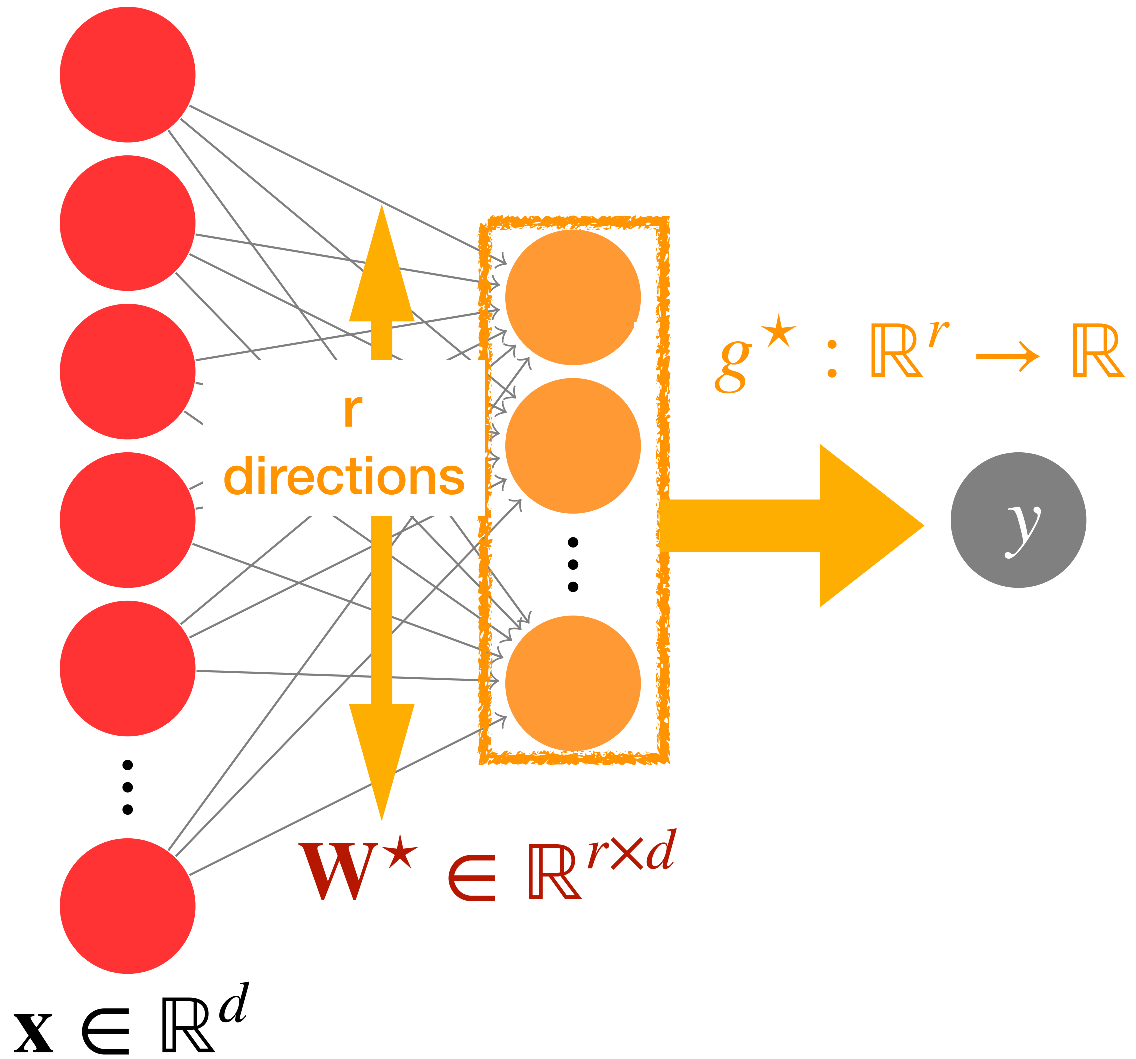
II

How hard is feature learning?
A classification of
easy & hard target functions

Toy problem : we know the function, not the directions

Target function: $Y \sim P^\star(Y \mid H^\star = W^\star \mathbf{X})$

$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$



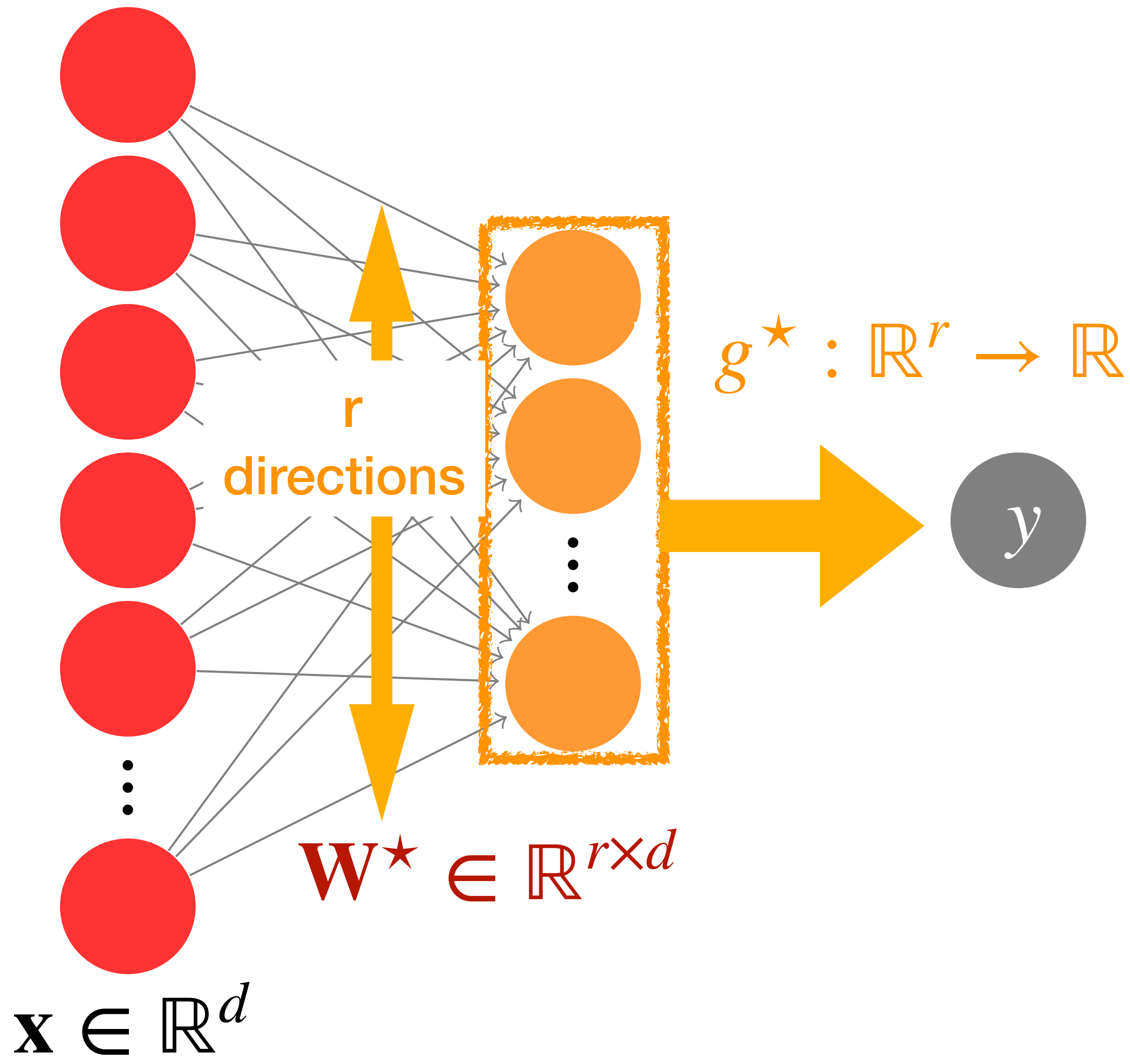
[Barbier, **FK**, Macris, Miolane, Zdeborova '17; Aubin, Maillard, Barbier, **FK**, Macris, Zdeborova '19]

Toy problem : we know the function, not the directions

Target function: $Y \sim P^\star(Y \,|\, H^\star = W^\star \mathbf{X})$

We know $g^\star$...

$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$



[Barbier, **FK**, Macris, Miolane, Zdeborova '17; Aubin, Maillard, Barbier, **FK**, Macris, Zdeborova '19]

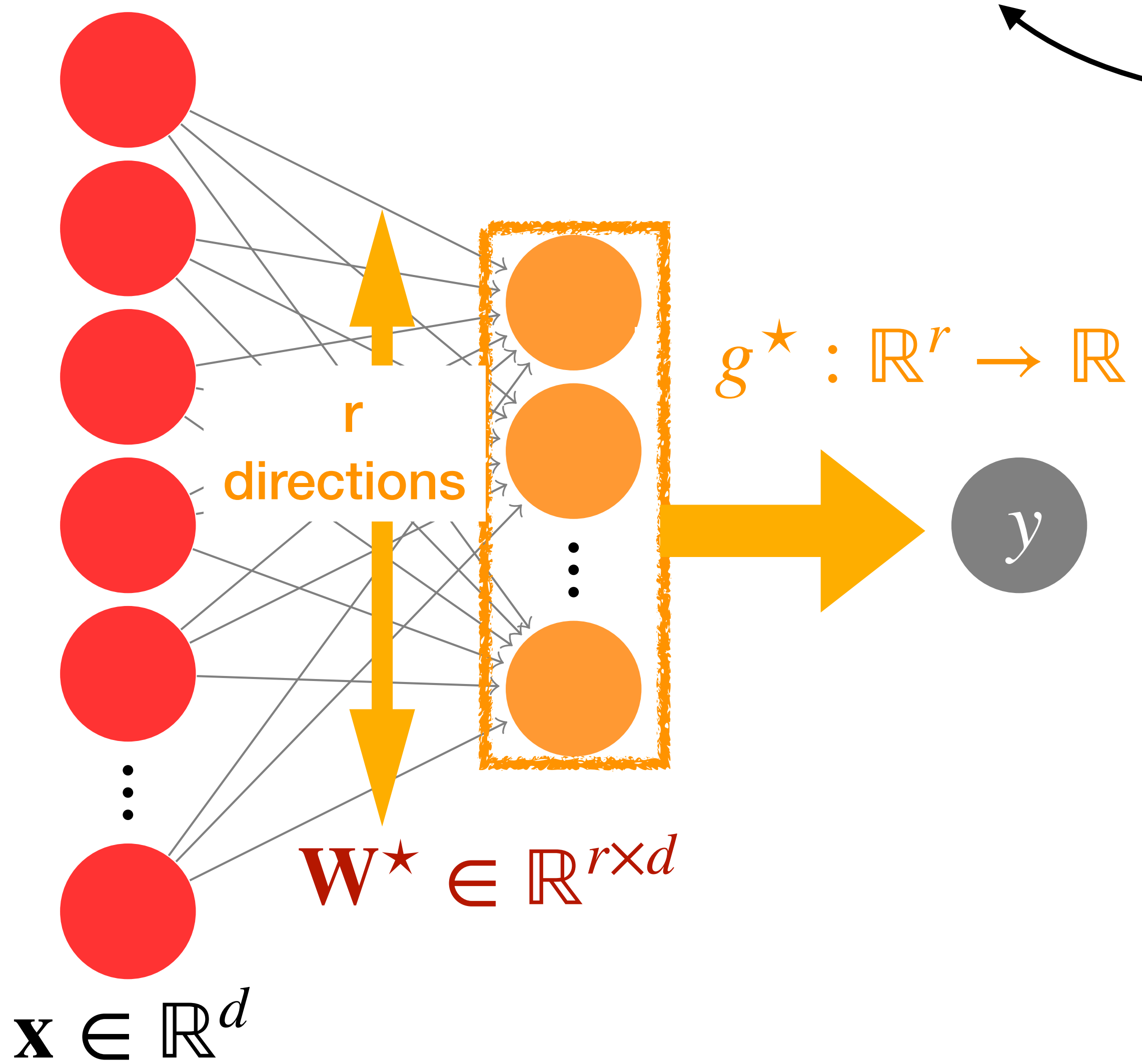
Toy problem : we know the function, not the directions

Target function: $Y \sim P^\star(Y \,|\, H^\star = W^\star \mathbf{X})$

We know $g^\star$…

$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$

… but not $W^\star$!



[Barbier, **FK**, Macris, Miolane, Zdeborova ’17; Aubin, Maillard, Barbier, **FK**, Macris, Zdeborova ’19]

Toy problem : we know the function, not the directions

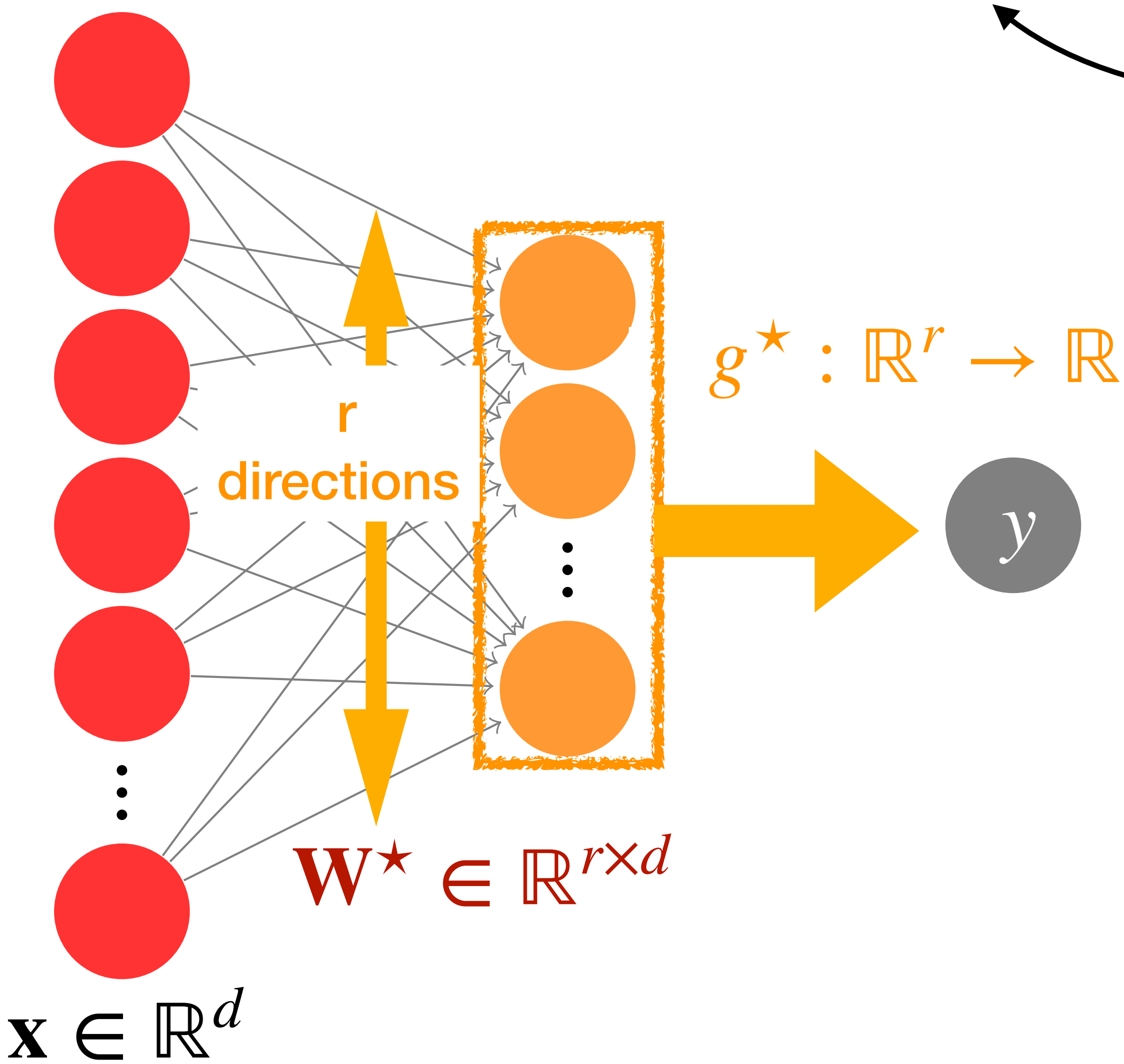
Target function: $Y \sim P^\star(Y \mid H^\star = W^\star \mathbf{X})$

We know $g^\star$**…**

$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$

… but not $W^\star$**!**

A long list of physicists over the last 35 years worked on this problems
[Derrida Gardner ’89 …
… Parisi, Mezard, Sompolinsky, Zechinna…]



[Barbier, **FK**, Macris, Miolane, Zdeborova ’17; Aubin, Maillard, Barbier, **FK**, Macris, Zdeborova ’19]

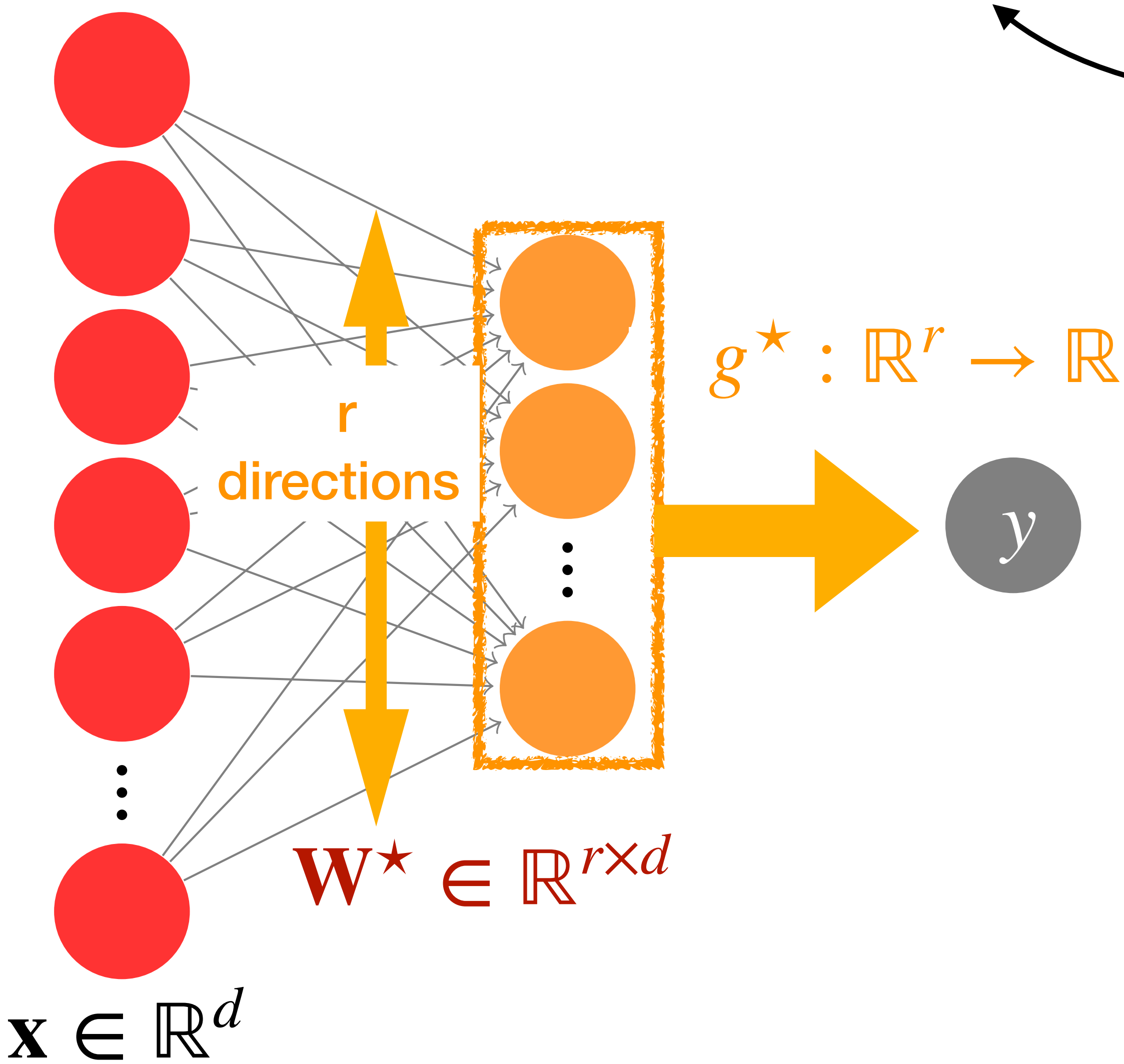
Toy problem : we know the function, not the directions

Target function: $Y \sim P^\star(Y \,|\, H^\star = W^\star \mathbf{X})$

$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$

We know $g^\star$…

… but not $W^\star$!



A long list of physicists over the last 35 years worked on this problems [Derrida Gardner '89 … … Parisi, Mezard, Sompolinsky, Zechinna…]

n= O(d) samples are sufficient !

[Barbier, **FK**, Macris, Miolane, Zdeborova '17; Aubin, Maillard, Barbier, **FK**, Macris, Zdeborova '19]

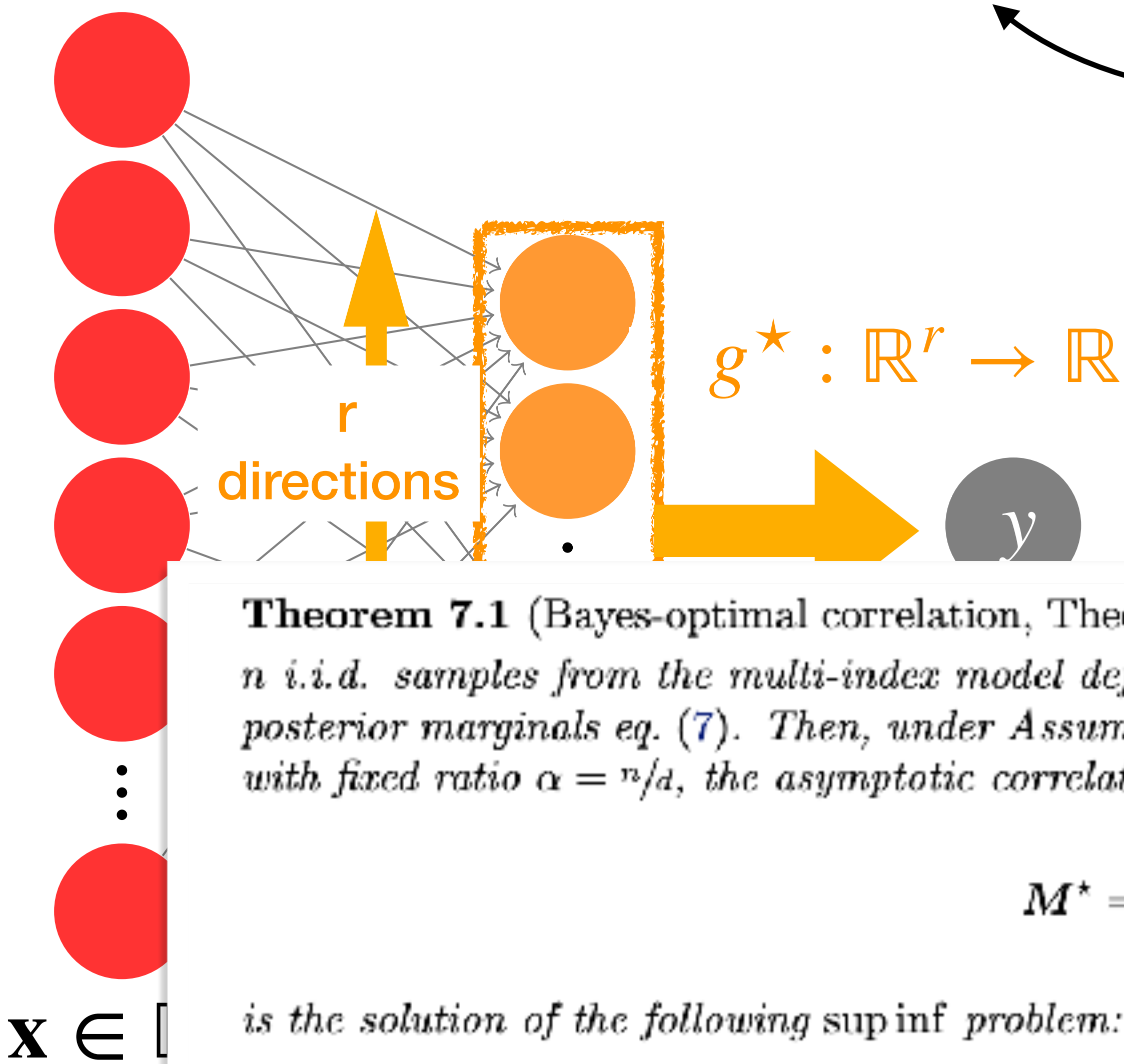
Toy problem : we know the function, not the directions

Target function: $Y \sim P^\star(Y | H^\star = W^\star \mathbf{X})$

$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$

We know $g^\star$…

… but not $W^\star$!



A long list of physicists over the last 35 years worked on this problems [Derrida Gardner '89 … … Parisi, Mezard, Sompolinsky, Zechinna…]

n= O(d) samples are sufficient !

Theorem 7.1 (Bayes-optimal correlation, Theorem 3.1 in Aubin et al. [2019], informal). *Let $(x_i, y_i)_{i\in[n]}$ denote n i.i.d. samples from the multi-index model defined in 1. Denote by $\hat{\boldsymbol{W}}_{\rm bo} = \mathbb{E}[W|X,y] \in \mathbb{R}^{p\times d}$ the mean of the posterior marginals eq. (7). Then, under Assumption 1 in the high-dimensional asymptotic limit where $n, d \to \infty$ with fixed ratio $\alpha = {}^n/_d$, the asymptotic correlation between the posterior mean and $\boldsymbol{W}^\star$:*

$$\boldsymbol{M}^\star = \lim_{d\to\infty} \mathbb{E}\left[\frac{1}{d}\hat{\boldsymbol{W}}_{\rm bo}\boldsymbol{W}^{\star\top}\right] \tag{23}$$

is the solution of the following sup inf *problem:*

$$\sup_{\hat{\boldsymbol{M}}\in\mathcal{S}_p^+} \inf_{\boldsymbol{M}\in\mathcal{S}_p^+} \left\{-\frac{1}{2}\operatorname{Tr}\boldsymbol{M}\hat{\boldsymbol{M}} - \frac{1}{2}\log\left(\boldsymbol{I}_p + \hat{\boldsymbol{M}}\right) + \frac{1}{2}\hat{\boldsymbol{M}} + \alpha H_Y(\boldsymbol{M})\right\} \tag{24}$$

where $H_Y(\boldsymbol{M}) = \mathbb{E}_{\boldsymbol{\xi}\sim\mathcal{N}(0,\boldsymbol{I}_p)}[H_Y(\boldsymbol{m}|\boldsymbol{\xi})]$, with $H_Y(\boldsymbol{M}|\boldsymbol{\xi})$ the the conditional entropy of the effective p-dimensional estimation problem eq. (10).

[Barbier, **FK**, Macris, Miolane, Zdeborova '17; Aubin, Maillard, Barbier, **FK**, Macris, Zdeborova '19]

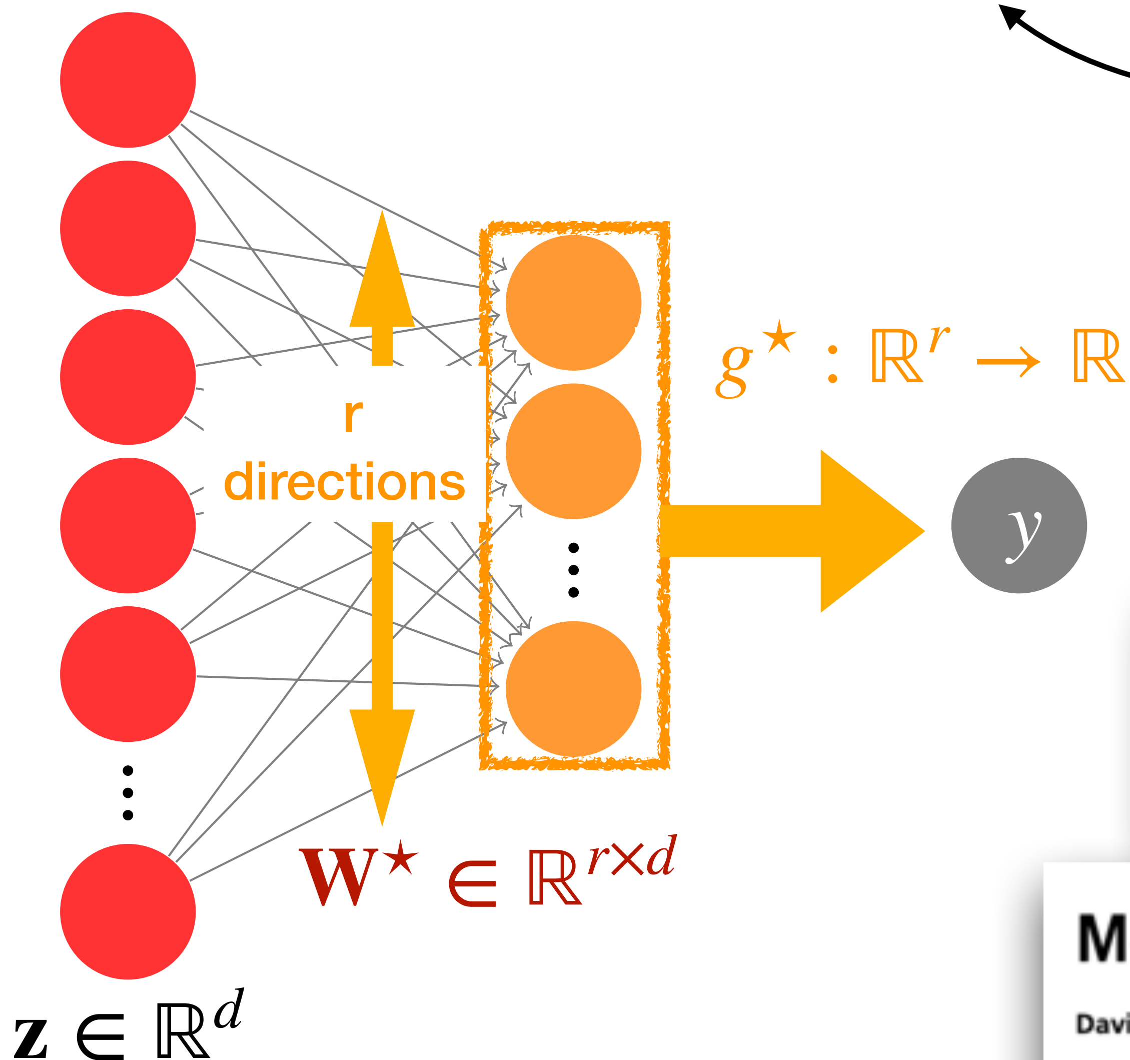
What about efficient iterative algorithms?

Target function: $Y \sim P^\star(Y \,|\, H^\star = W^\star \mathbf{X})$

$$y = f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star = W^\star \mathbf{x})$$

We know $g^\star$...

... but not $W^\star$!



Solution of 'Solvable model of a spin glass'

D. J. Thouless , P. W. Anderson & R. G. Palmer

To cite this article: D. J. Thouless , P. W. Anderson & R. G. Palmer (1977) Solution of 'Solvable model of a spin glass', Philosophical Magazine, 35:3, 593-601, DOI: 10.1080/14786437708235992

The estimation error of general first order methods

Michael Celentano* Andrea Montanari*† Yuchen Wu*

Message-passing algorithms for compressed sensing

David L. Donoho[a,1], Arian Maleki[b], and Andrea Montanari[a,b,1]

Departments of [a]Statistics and [b]Electrical Engineering, Stanford University, Stanford, CA 94305

An iterative construction of solutions of the TAP equations for the Sherrington-Kirkpatrick model

Erwin Bolthausen*† Universität Zürich

State Evolution for General Approximate Message Passing Algorithms, with Applications to Spatial Coupling

Adel Javanmard* and Andrea Montanari †

Our best shot: Bayes-AMP for multi-index models

$$\boldsymbol{\Omega}^t = \boldsymbol{X} f_t(\mathbf{B}^t) - g_{t-1}(\boldsymbol{\Omega}^{t-1}, \mathbf{y}) \boldsymbol{V}_t$$
$$\mathbf{B}^{t+1} = \boldsymbol{X}^T g_t(\boldsymbol{\Omega}^t, \mathbf{y}) + f_t(\mathbf{B}^t) \boldsymbol{A}_t$$

$\mathbf{B} \in \mathbb{R}^{d\times p}$ and $\boldsymbol{\Omega} \in \mathbb{R}^{n\times p}$

Estimator for weights $\hat{\boldsymbol{W}}^t \in \mathbb{R}^{p\times d} = f_t(\mathbf{B}^t)^\top$

Estimator for pre-activation $g_t(\boldsymbol{\Omega}^t) \in \mathbb{R}^{n\times p}$ $\quad g_t = \mathbb{E}\left[\mathbf{V}^{-1}\mathbf{Z} + \omega \,|\, \mathbf{Y}\right]$

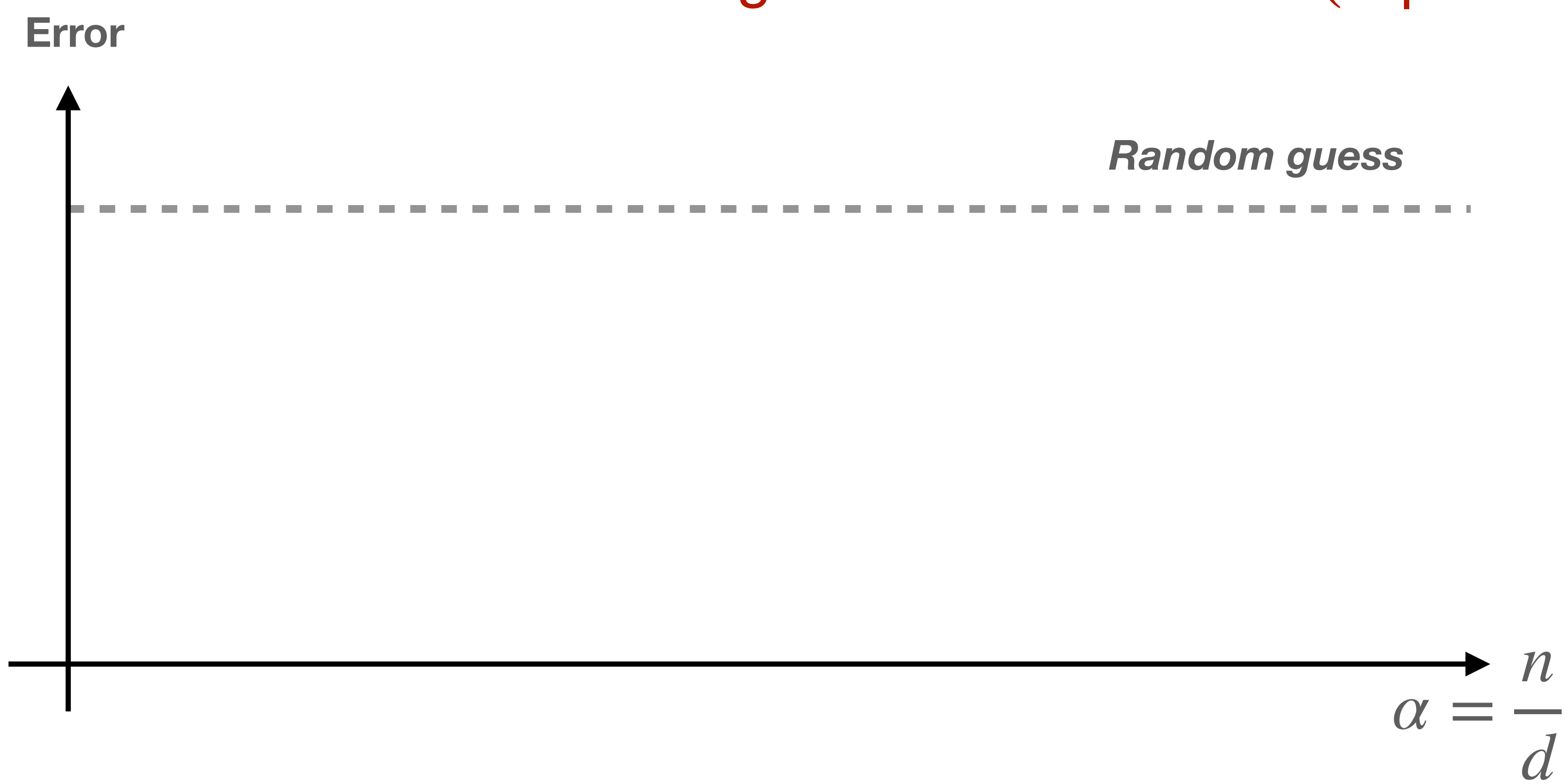
Performance can be analysed rigorously with the state evolution technics*

**(May require a hot start with a spectral method provided by linearising the algorithm, see e.g. Maillard et al '20, Mondelli Venkataramanan '21])*

[Troiani, Dandi, Delilippis, Zdeborova, Loureiro, **FK**, '24]

A classification of problems

Target function: $Y \sim P^\star(Y \mid H^\star = W^\star \mathbf{X})$



AMP/TAP Classification

TRIVIAL

W* can be learned with *any* $n = \mathcal{O}(d)$ if

$$\mathbb{E}[H \mid Y] \neq 0$$

with non-zero probability over y

EASY

For even target (or different symmetry for multi-index) learning W* requires $n > \alpha_c d$

$$\alpha_c = \mathbb{E}[(\mathbb{E}[H \mid Y]^2 - 1)^2]^{-1}$$

HARD

***Very restricted* set of hard functions ($\alpha_d \to \infty$) require more than $\mathcal{O}(d)$ data!**

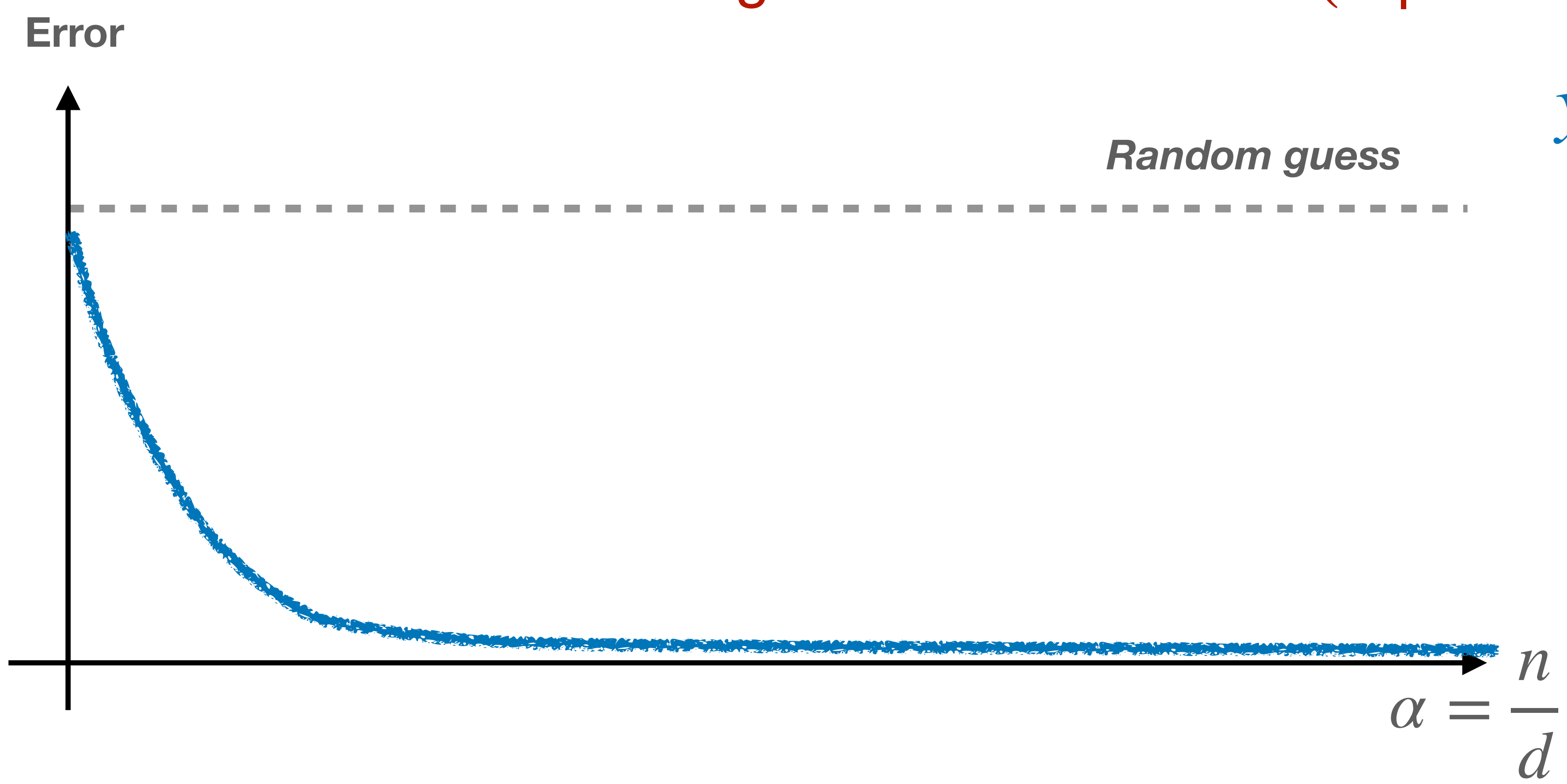
Example : r-parity, $r \geq 3$

$$y = \mathrm{sign}(h_1^\star h_2^\star \dots h_r^\star)$$

[Barbier, **FK**, Macris, Miolane, Zdeborová '17; Mondelli, Montanari '18;Troiani, Dandi, Delilippis, Zdeborova, Loureiro, **FK**, '24]

A classification of problems

Target function: $Y \sim P^\star(Y | H^\star = W^\star \mathbf{X})$



$$y = g^\star(\mathbf{x}) = (h^\star)^3 - 3h^\star$$

AMP finds $\mathbf{h}^\star$ after O(1) iterations

AMP/TAP Classification

TRIVIAL

W* can be learned with *any*

$n = \mathcal{O}(d)$ if

$$\mathbb{E}[H | Y] \neq 0$$

with non-zero probability over y

EASY

For even target (or different symmetry for multi-index) learning W* requires $n > \alpha_c d$

$$\alpha_c = \mathbb{E}[(\mathbb{E}[H | Y]^2 - 1)^2]^{-1}$$

HARD

***Very restricted* set of hard functions ($\alpha_d \to \infty$) require more than $\mathcal{O}(d)$ data!**

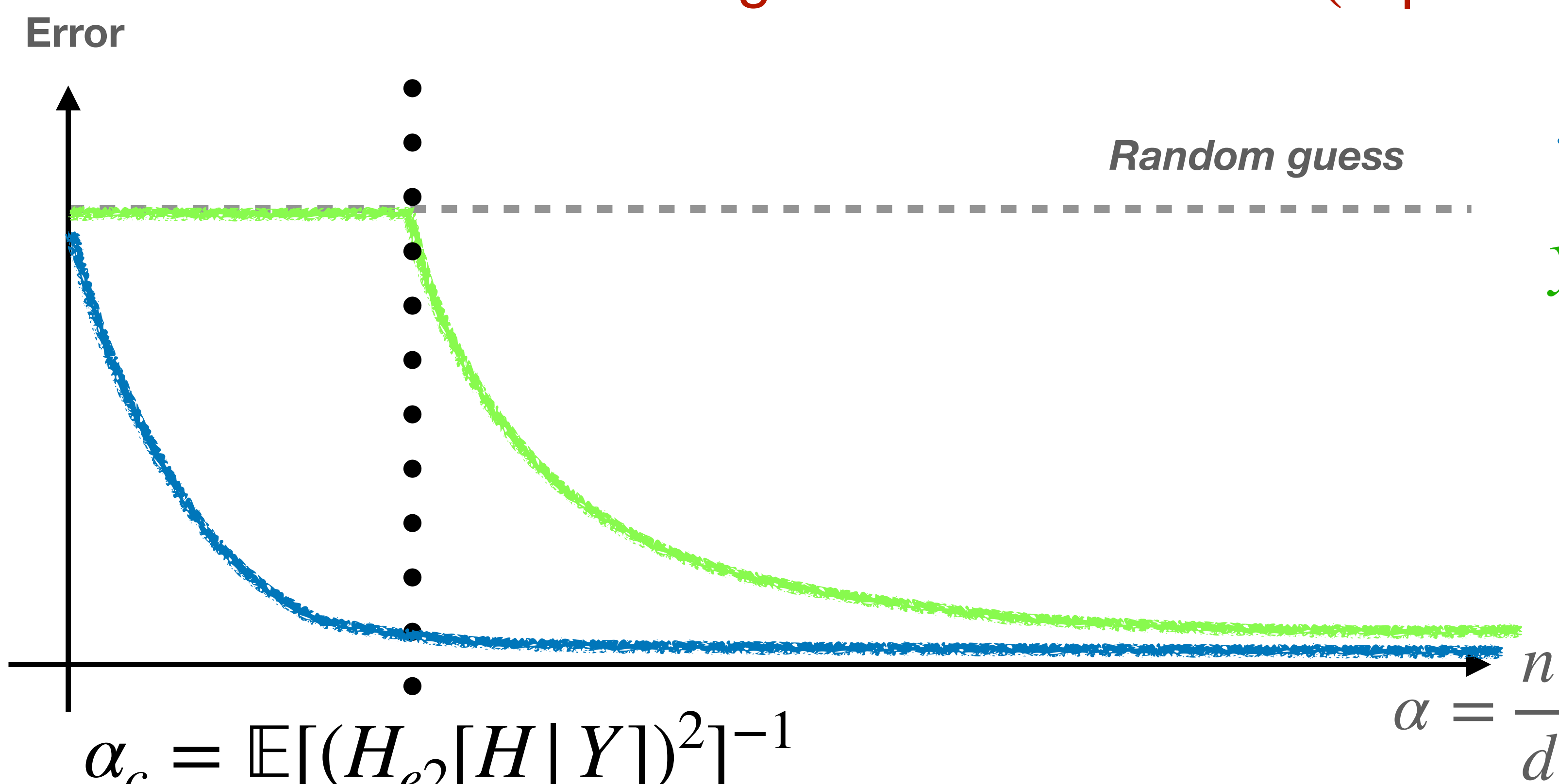
Example : r-parity, $r \geq 3$

$$y = \text{sign}(h_1^\star h_2^\star \ldots h_r^\star)$$

[Barbier, **FK**, Macris, Miolane, Zdeborová '17; Mondelli, Montanari '18;Troiani, Dandi, Delilippis, Zdeborova, Loureiro, **FK**, '24]

A classification of problems

Target function: $Y \sim P^\star(Y|H^\star = W^\star \mathbf{X})$



$$\alpha_c = \mathbb{E}[(H_{e2}[H|Y])^2]^{-1}$$

$y = g^\star(\mathbf{x}) = (h^\star)^3 - 3h^\star$

$y = g^\star(\mathbf{x}) = |h^\star|^2 \qquad \alpha_c = 1/2$

AMP finds $\mathbf{h}^\star$ after $O(\log d)$ iterations

(Rigorously: requires an initialisation with a spectral start)

AMP/TAP Classification

TRIVIAL

W* can be learned with *<u>any</u>* $n = \mathcal{O}(d)$ if

$$\mathbb{E}[H|Y] \neq 0$$

with non-zero probability over y

EASY

For even target (or different <u>symmetry</u> for multi-index) learning W* requires $n > \alpha_c d$

$$\alpha_c = \mathbb{E}[(\mathbb{E}[H|Y]^2 - 1)^2]^{-1}$$

HARD

***<u>Very restricted</u>* set of hard functions ($\alpha_d \to \infty$) require more than $\mathcal{O}(d)$ data!**

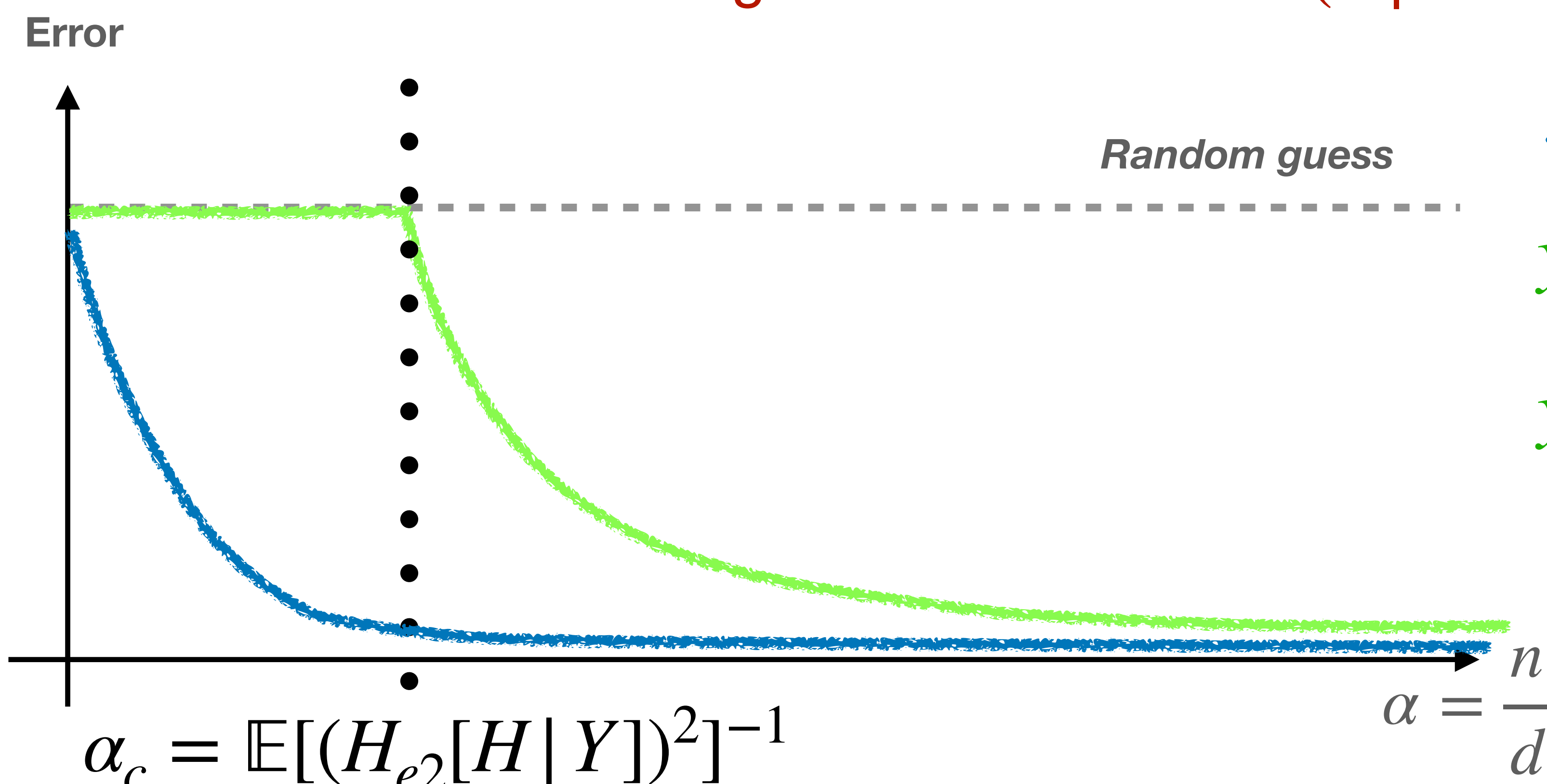
<u>Example : r-parity, $r \geq 3$</u>

$$y = \mathrm{sign}(h_1^\star h_2^\star \ldots h_r^\star)$$

[Barbier, **FK**, Macris, Miolane, Zdeborová '17; Mondelli, Montanari '18;Troiani, Dandi, Delilippis, Zdeborova, Loureiro, **FK**, '24]

A classification of problems

Target function: $Y \sim P^\star(Y|H^\star = W^\star \mathbf{X})$



$\alpha_c = \mathbb{E}[(H_{e2}[H|Y])^2]^{-1}$

$y = g^\star(\mathbf{x}) = (h^\star)^3 - 3h^\star$

$y = g^\star(\mathbf{x}) = |h^\star|^2 \qquad \alpha_c = 1/2$

$y = g^\star(\mathbf{x}) = \mathrm{sign}(h_1^\star h_2^\star)$

AMP finds $\mathbf{h}^\star$ after $O(\log d)$ iterations
(Rigorously: requires an initialisation with a spectral start)

AMP/TAP Classification

TRIVIAL

W* can be learned with *any*
$n = \mathcal{O}(d)$ **if**

$$\mathbb{E}[H|Y] \neq 0$$

with non-zero probability over y

EASY

For even target (or different <u>symmetry</u> for multi-index) learning W* requires $n > \alpha_c d$

$$\alpha_c = \mathbb{E}[(\mathbb{E}[H|Y]^2 - 1)^2]^{-1}$$

HARD

***<u>Very restricted</u>* set of hard functions ($\alpha_d \to \infty$) require more than $\mathcal{O}(d)$ data!**

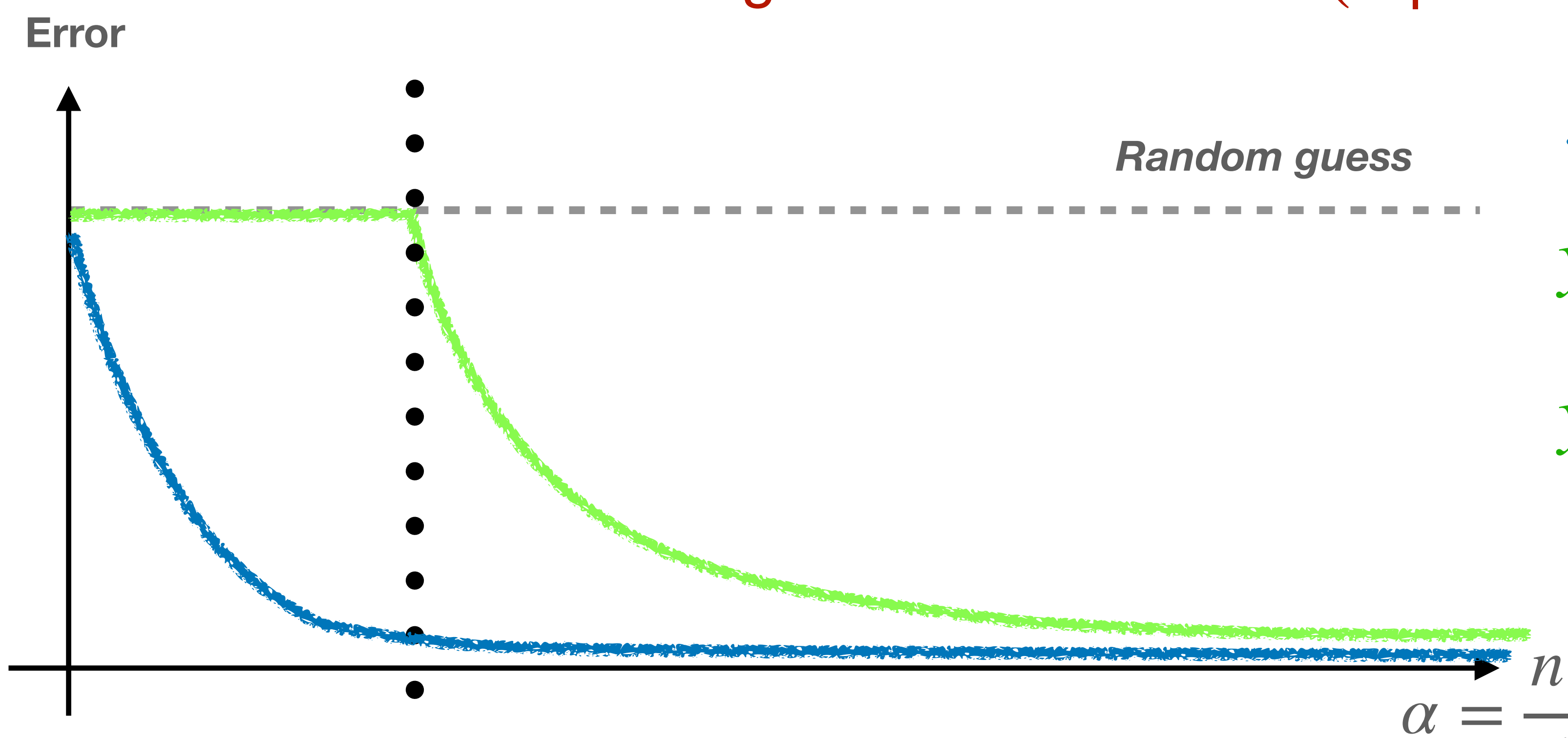
<u>Example : r-parity,</u> $r \geq 3$

$$y = \mathrm{sign}(h_1^\star h_2^\star \dots h_r^\star)$$

[Barbier, **FK**, Macris, Miolane, Zdeborová '17; Mondelli, Montanari '18;Troiani, Dandi, Delilippis, Zdeborova, Loureiro, **FK**, '24]

A classification of problems

Target function: $Y \sim P^\star(Y | H^\star = W^\star \mathbf{X})$



$y = g^\star(\mathbf{x}) = (h^\star)^3 - 3h^\star$

$y = g^\star(\mathbf{x}) = |h^\star|^2 \qquad \alpha_c = 1/2$

$y = g^\star(\mathbf{x}) = \mathrm{sign}(h_1^\star h_2^\star)$

$$\alpha_c^{-1} = \sup_{\{M \in S_p^+ | \|M\|_2^2 = 1\}} \mathbb{E}[H_{e2}(\mathbf{H} | Y) M H_{e2}(\mathbf{H} | Y)^T]$$

AMP finds $\mathbf{h}^\star$ after $O(\log d)$ iterations
(Rigorously: requires an initialisation with a spectral start)

AMP/TAP Classification

TRIVIAL

W* can be learned with *<u>any</u>* $n = \mathcal{O}(d)$ if

$$\mathbb{E}[H | Y] \neq 0$$

with non-zero probability over y

EASY

For even target (or different symmetry for multi-index) learning W* requires $n > \alpha_c d$

$$\alpha_c = \mathbb{E}[(\mathbb{E}[H | Y]^2 - 1)^2]^{-1}$$

HARD

***<u>Very restricted</u>* set of hard functions ($\alpha_d \to \infty$) require more than $\mathcal{O}(d)$ data!**

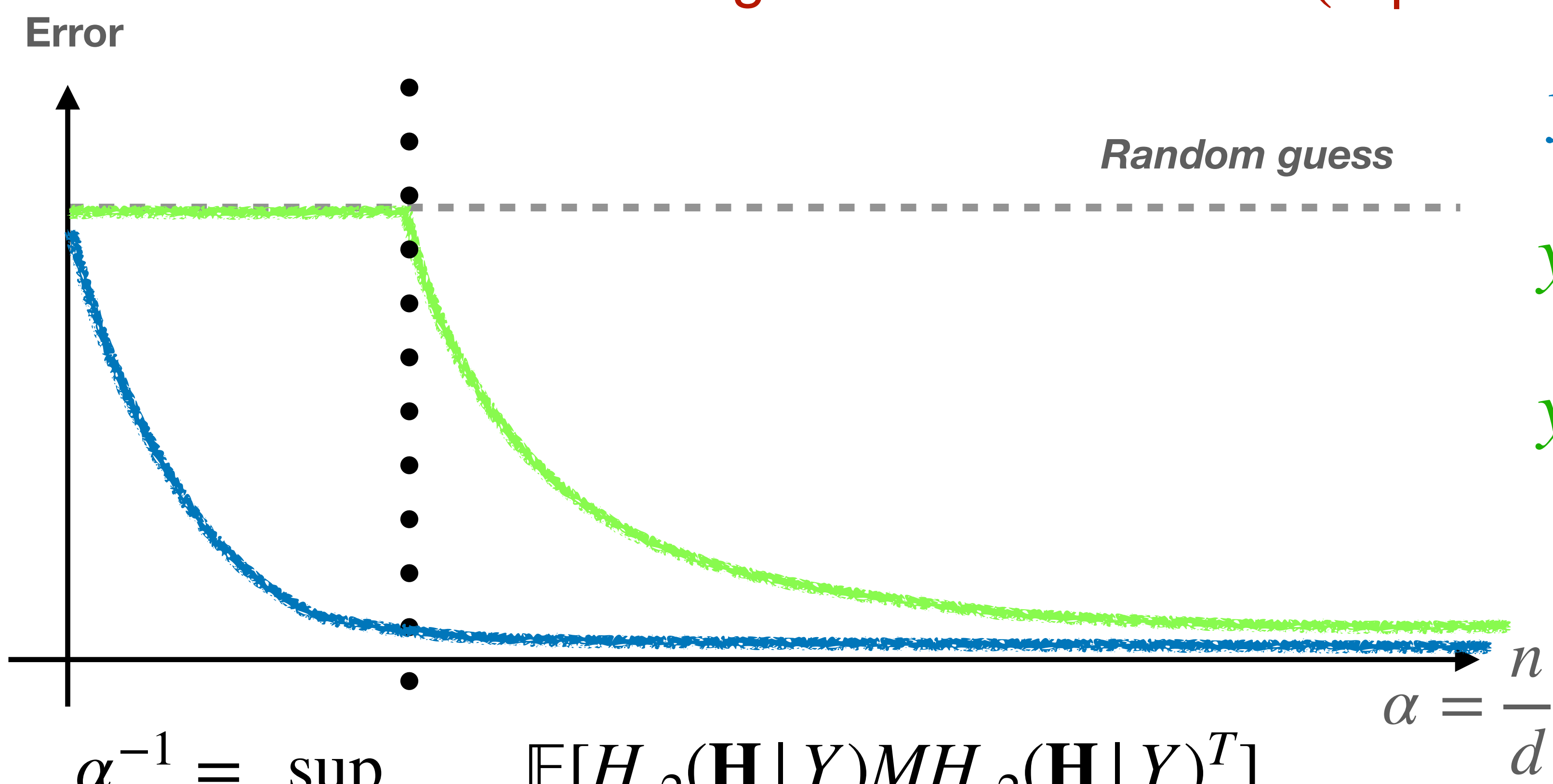
<u>Example : r-parity, $r \geq 3$</u>

$$y = \mathrm{sign}(h_1^\star h_2^\star \dots h_r^\star)$$

[Barbier, **FK**, Macris, Miolane, Zdeborová '17; Mondelli, Montanari '18;Troiani, Dandi, Delilippis, Zdeborova, Loureiro, **FK**, '24]

A classification of problems

Target function: $Y \sim P^\star(Y | H^\star = W^\star \mathbf{X})$



$y = g^\star(\mathbf{x}) = (h^\star)^3 - 3h^\star$

$y = g^\star(\mathbf{x}) = |h^\star|^2 \qquad \alpha_c = 1/2$

$y = g^\star(\mathbf{x}) = \mathrm{sign}(h_1^\star h_2^\star) \; \alpha_c = \frac{\pi^2}{4}$

$$\alpha_c^{-1} = \sup_{\{M \in S_p^+ | \|M\|_2^2 = 1\}} \mathbb{E}[H_{e2}(\mathbf{H}|Y) M H_{e2}(\mathbf{H}|Y)^T]$$

AMP finds $\mathbf{h}^\star$ after $O(\log d)$ iterations

(Rigorously: requires an initialisation with a spectral start)

AMP/TAP Classification

TRIVIAL

W* can be learned with *any* $n = \mathcal{O}(d)$ if

$$\mathbb{E}[H|Y] \neq 0$$

with non-zero probability over y

EASY

For even target (or different symmetry for multi-index) learning W* requires $n > \alpha_c d$

$$\alpha_c = \mathbb{E}[(\mathbb{E}[H|Y]^2 - 1)^2]^{-1}$$

HARD

***Very restricted* set of hard functions ($\alpha_d \to \infty$) require more than $\mathcal{O}(d)$ data!**

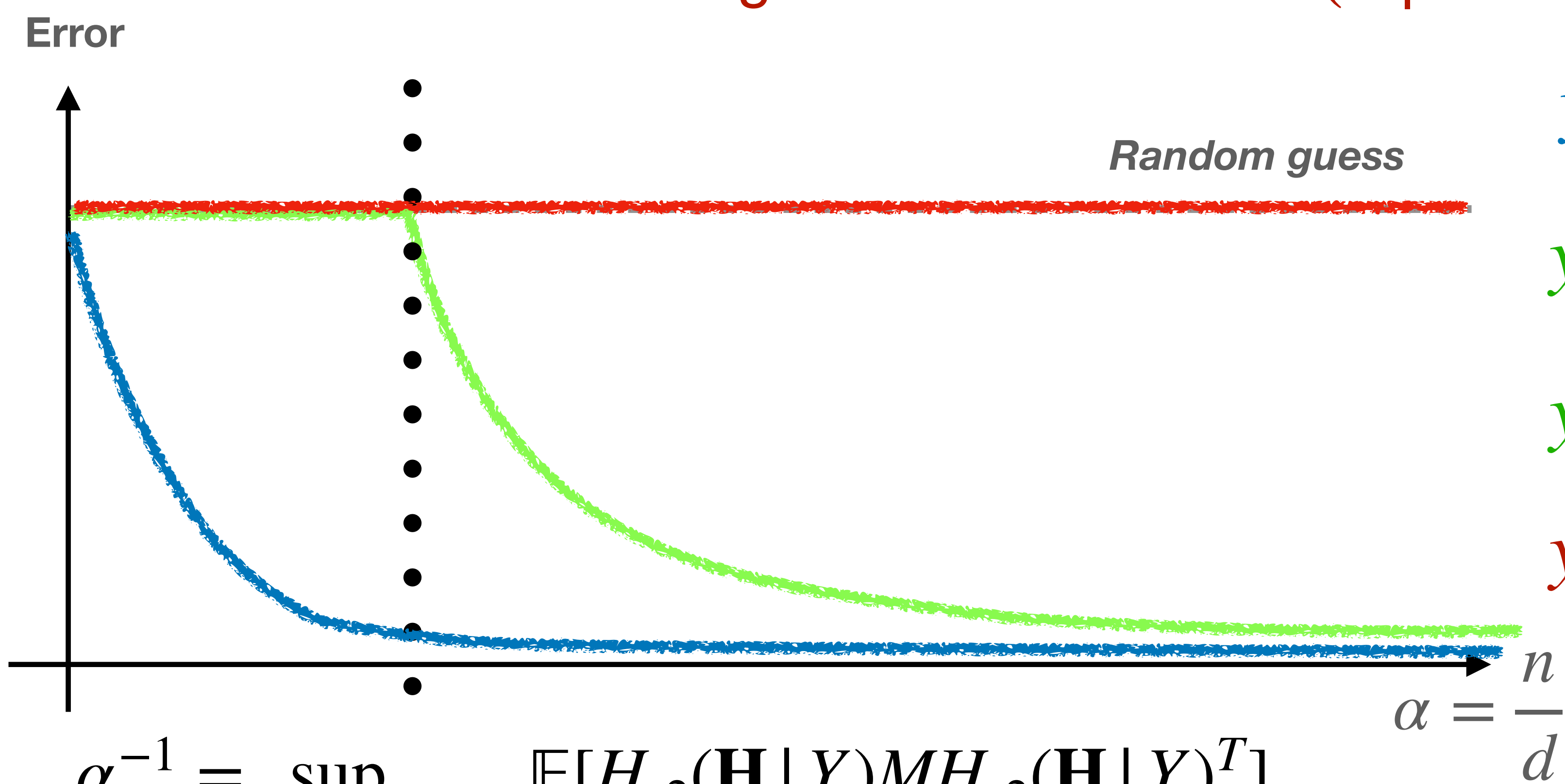
Example : r-parity, $r \geq 3$

$$y = \mathrm{sign}(h_1^\star h_2^\star \ldots h_r^\star)$$

[Barbier, **FK**, Macris, Miolane, Zdeborová '17; Mondelli, Montanari '18;Troiani, Dandi, Delilippis, Zdeborova, Loureiro, **FK**, '24]

A classification of problems

Target function: $Y \sim P^\star(Y|H^\star = W^\star \mathbf{X})$



$$\alpha_c^{-1} = \sup_{\{M \in S_p^+ | \|M\|_2^2 = 1\}} \mathbb{E}[H_{e2}(\mathbf{H}|Y) M H_{e2}(\mathbf{H}|Y)^T]$$

$y = g^\star(\mathbf{x}) = (h^\star)^3 - 3h^\star$

$y = g^\star(\mathbf{x}) = |h^\star|^2$ $\quad \alpha_c = 1/2$

$y = g^\star(\mathbf{x}) = \mathrm{sign}(h_1^\star h_2^\star)$ $\quad \alpha_c = \frac{\pi^2}{4}$

$y = g^\star(\mathbf{x}) = \mathrm{sign}(h_1^\star h_2^\star h_3^\star)$

AMP does not find $\mathbf{h}^\star$ with O(d) data

AMP/TAP Classification

TRIVIAL

W* can be learned with *any* $n = \mathcal{O}(d)$ if

$$\mathbb{E}[H|Y] \neq 0$$

with non-zero probability over y

EASY

For even target (or different symmetry for multi-index) learning W* requires $n > \alpha_c d$

$$\alpha_c = \mathbb{E}[(\mathbb{E}[H|Y]^2 - 1)^2]^{-1}$$

HARD

***Very restricted* set of hard functions ($\alpha_d \to \infty$) require more than $\mathcal{O}(d)$ data!**

Example : r-parity, $r \geq 3$

$$y = \mathrm{sign}(h_1^\star h_2^\star \ldots h_r^\star)$$

[Barbier, **FK**, Macris, Miolane, Zdeborová '17; Mondelli, Montanari '18;Troiani, Dandi, Delilippis, Zdeborova, Loureiro, **FK**, '24]

Computer scientists agree with us!

AMP/TAP Classification

TRIVIAL

W* can be learned with *any* $n = \mathcal{O}(d)$ if

$$\mathbb{E}[H \mid Y] \neq 0$$

with non-zero probability over y

EASY

For even target (or different symmetry for multi-index) learning W* requires $n > \alpha_c d$

$$\alpha_c = \mathbb{E}[(\mathbb{E}[H \mid Y]^2 - 1)^2]^{-1}$$

HARD

Very restricted set of hard functions ($\alpha_d \to \infty$) require more than $\mathcal{O}(d)$ data!

Example : r-parity, $r \geq 3$

$$y = \mathrm{sign}(h_1^\star h_2^\star \dots h_r^\star)$$

Statistical Queries (SQ) bounds

$$\mathbb{E}[\phi(Y, \mathbf{Z})] = ?$$

Generative exponents classification

TRIVIAL

W* can be learned with *any* $n = \mathcal{O}(d)$ if

$$\mathbb{E}[H \mid Y] \neq 0$$

with non-zero probability over y

EASY

For even target learning W* requires $n > \alpha_c d$

$$\alpha_c = \mathbb{E}[(\mathbb{E}[H \mid Y]^2 - 1)^2]^{-1}$$

HARD

Very restricted set of hard functions ($\alpha_d \to \infty$) require more than $\mathcal{O}(d)$ data!

Example : r-parity

$$y = \mathrm{sign}(h_1^\star h_2^\star \dots h_r^\star)$$

See e.g. [Damian, Pillaud-Vivien, Lee, Bruna '24] **Trivial** & **Easy** targets correspond to generative exponent 1 & 2

Computer scientists agree with us!

AMP/TAP Classification

TRIVIAL

W* can be learned with *any*
$n = \mathcal{O}(d)$ **if**

$$\mathbb{E}[H \,|\, Y] \neq 0$$

with non-zero probability over y

EASY

For even target (or different symmetry for multi-index) learning W* requires $n > \alpha_c d$

$$\alpha_c = \mathbb{E}[(\mathbb{E}[H \,|\, Y]^2 - 1)^2]^{-1}$$

HARD

Very restricted **set of hard functions** ($\alpha_d \to \infty$) **require more than** $\mathcal{O}(d)$ **data!**

Example : r-parity, $r \geq 3$

$$y = \mathrm{sign}(h_1^\star h_2^\star \ldots h_r^\star)$$

Statistical Queries (SQ) bounds

Generative exponents classification

$$\mathbb{E}[\phi(Y, \mathbf{Z})] = \,?$$

TRIVIAL

W* can be learned with *any*
$n = \mathcal{O}(d)$ **if**

$$\mathbb{E}[H \,|\, Y] \neq 0$$

with non-zero probability over y

EASY

For even target learning W* requires $n > \alpha_c d$

$$\alpha_c = \mathbb{E}[(\mathbb{E}[H \,|\, Y]^2 - 1)^2]^{-1}$$

HARD

Very restricted **set of hard functions** ($\alpha_d \to \infty$) **require more than** $\mathcal{O}(d)$ **data!**

Example : r-parity

$$y = \mathrm{sign}(h_1^\star h_2^\star \ldots h_r^\star)$$

See e.g. [Damian, Pillaud-Vivien, Lee, Bruna ’24] **Trivial** & **Easy** targets correspond to generative exponent 1 & 2

Multi-index models: Example #1

$$g^\star = \text{sign}(\mathrm{h}_1^\star \mathrm{h}_2^\star)$$

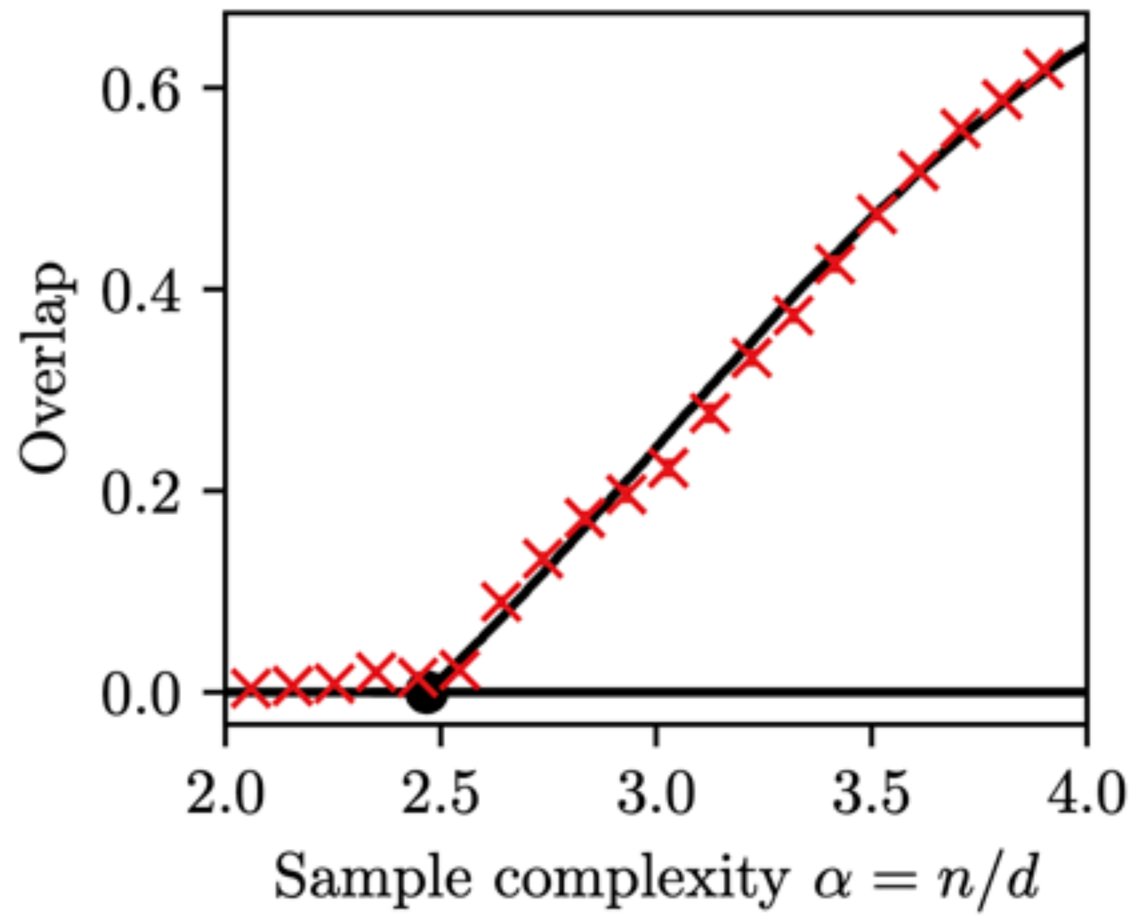


Figure 1: Numerical illustration of the weak learnability phase transition for the 2-sparse parity $g(z_1, z_2) = \text{sign}(z_1 z_2)$ that has a phase transition at $\alpha_c(2) = \pi^2/4$. The overlap shows how well the directions z_1 and z_2 are recovered. Given the permutation symmetry in (19), we show here and in all the subsequent figures the optimal permutation of the overlap matrix elements reached by AMP. The solid black line is the prediction from the theory. Crosses are averages over 72 runs of AMP Algorithm 1 with $d = 500$.

Multi-index models: Example #2

$$g^\star = h_1^{\star 2} + \mathrm{sign}(h_1^\star h_2^\star h_3^\star)$$

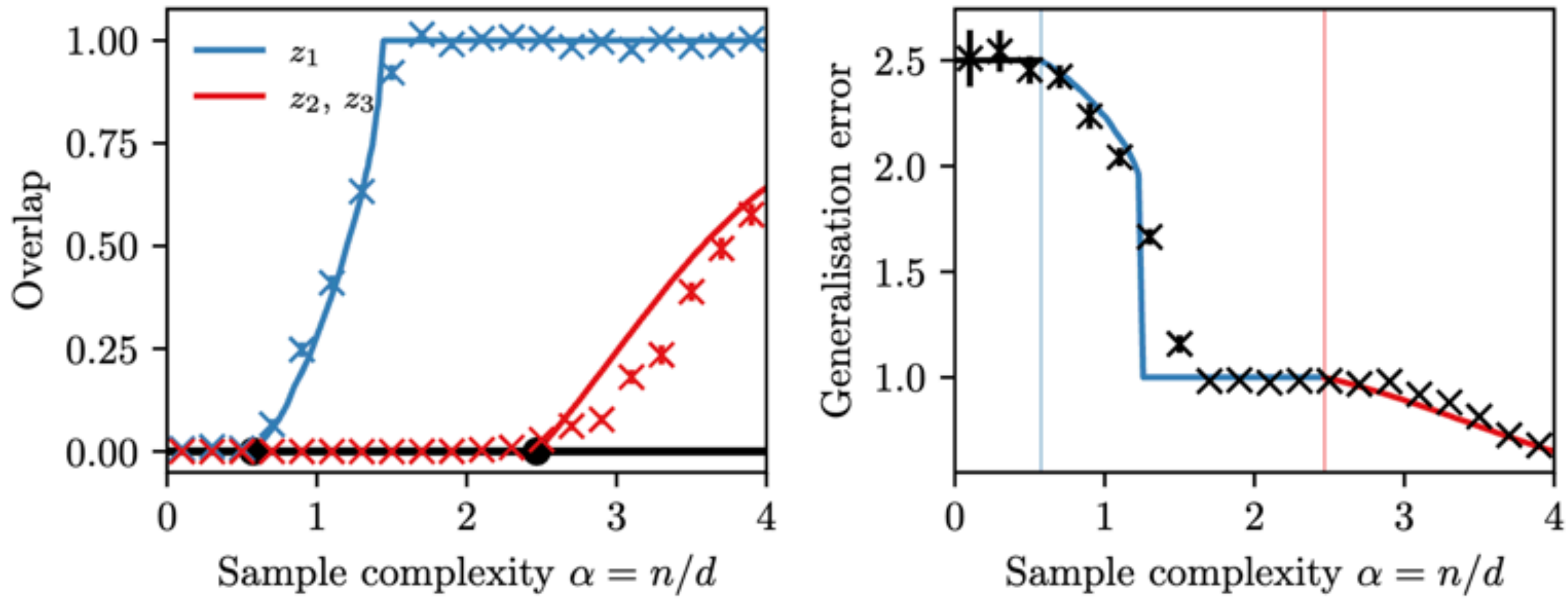


Figure 2: Hierarchical weak learnability for the staircase function $g(z_1, z_2, z_3) = z_1^2 + \mathrm{sign}(z_1 z_2 z_3)$. (**Left**): Overlaps with the first direction $|M_{11}|$ (blue), and with the second and third one $1/2(M_{22} + M_{33})$ (red) as a function of the sample complexity $\alpha = n/d$, with solid lines denoting state evolution curves Equation (8), and crosses/dots finite-size runs of AMP Algorithm 1 with $d = 500$ and averaged over 72 seeds. All other overlaps are zero (black). The two black dots indicate the critical thresholds at $\alpha_1 \approx 0.575$ and $\alpha_2 = \pi^2/4$. (**Right**) Corresponding generalization error as a function of the sample complexity. Details on the numerical implementation are discussed in Appendix D.

Multi-index models: Example #2

$$g^\star = h_1^{\star 2} + \mathrm{sign}(\mathrm{h}_1^\star \mathrm{h}_2^\star \mathrm{h}_3^\star)$$

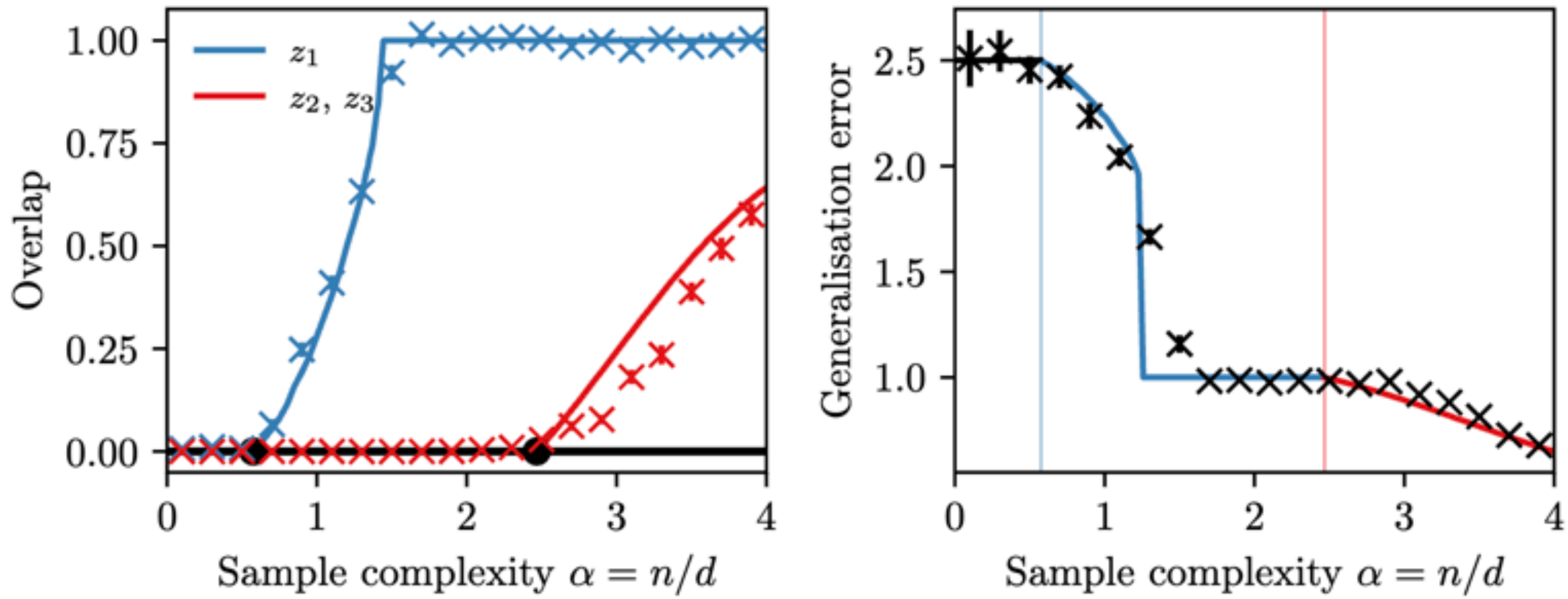


Figure 2: Hierarchical weak learnability for the st
Overlaps with the first direction $|M_{11}|$ (blue), and
a function of the sample complexity $\alpha = {}^n/_d$, with
and crosses/dots finite-size runs of AMP Algorith
overlaps are zero (black). The two black dots indi
(**Right**) Corresponding generalization error as a fu
implementation are discussed in Appendix D.

Multi-index models: Example #2

$$g^\star = h_1^{\star 2} + \mathrm{sign}(h_1^\star h_2^\star h_3^\star)$$

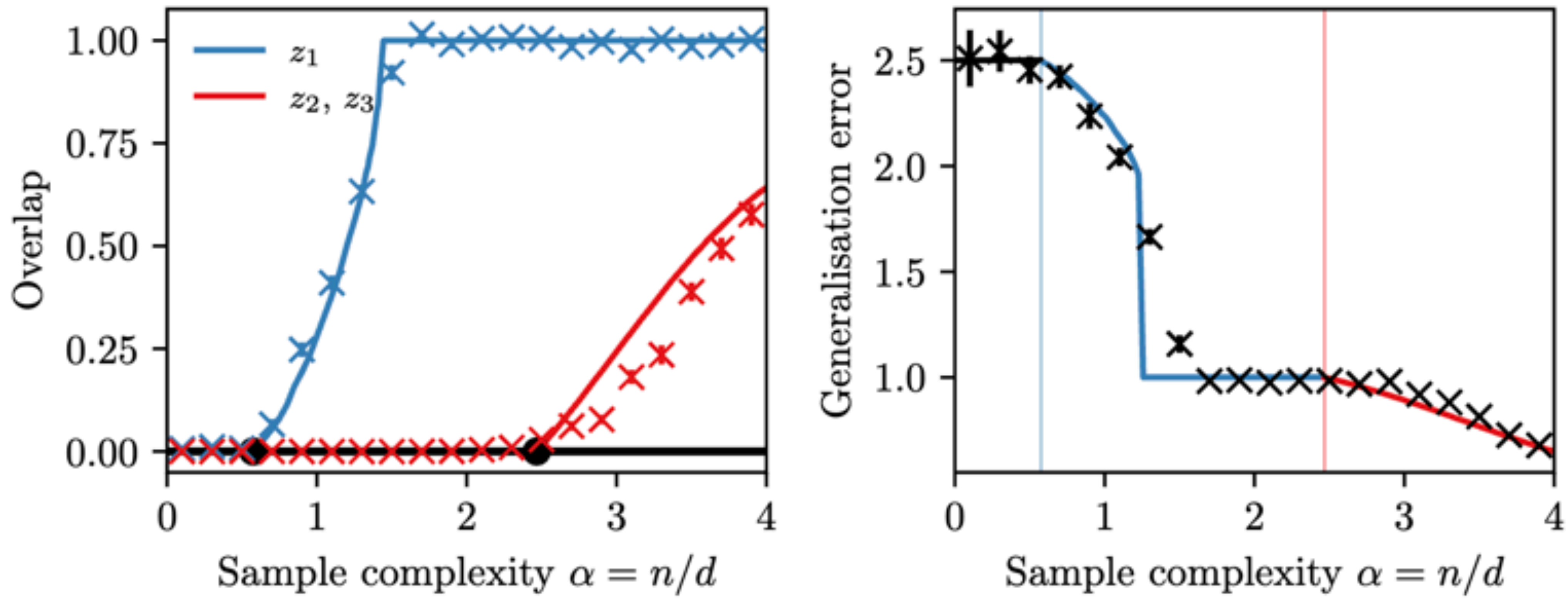


Figure 2: Hierarchical weak learnability for the st
Overlaps with the first direction $|M_{11}|$ (blue), and
a function of the sample complexity $\alpha = n/d$, with
and crosses/dots finite-size runs of AMP Algorith
overlaps are zero (black). The two black dots indi
(**Right**) Corresponding generalization error as a fu
implementation are discussed in Appendix D.

Grand staircase is different from Staircase of [Abbe et al ‘22+’23]

The situation so far

Classification of target functions

TRIVIAL

W* can be learned with _any_ $n = \mathcal{O}(d)$ as long as (for some value of Y)

$$\mathbb{E}[H \,|\, Y] \neq 0$$

EASY

For even target (or a different symmetry for multi-index) learning W* requires $n > \alpha_c d$

$$\alpha_c = \mathbb{E}[(\mathbb{E}[H \,|\, Y]^2 - 1)^2]^{-1}$$

HARD

Very restricted **set of hard functions ($\alpha_d \to \infty$) require more than $\mathcal{O}(d)$ data!**

Example : r-parity

$$y = \mathrm{sign}(h_1^\star h_2^\star \dots h_r^\star)$$

- This is all very nice, but from the point of view of machine learning, this is **cheating: we cannot assume we know the function**

- These are just (loose?) bounds on the hardness of learning a particular target class

- What happens when one just use a neural network instead?

III

Can two-layer nets learn features as efficiently as AMP?

SGD for Gaussian data : a summary of the last 30 years

Many mathematical works on GD with *fresh batch* of Gaussian data:

[Saad & Solla ’95, … Goldt, Advani, Saxe, **FK**, Zdeborová ’19; YS Tan, R Vershynin ’19; Mei, Misiakiewicz, Montanari ’19; Ben Arous, Gheissari, Jagannath ’20 & ’22; Abbe et al ’21; Veiga, Stephan, Loureiro, **FK**, Zdeborová ’22; Paquette, Paquette, Adlam, Pennington ’22; Abbe et al ’22; Abbe et al ’23; Berthier, Montanari, Zhou ’23; Arnaboldi, Stephan, **FK**, Loureiro ’23; Arnaboldi, Dandi, **FK**, Loureiro, Pesce, Stephan ’23+’24; Bruna et al ’23; Chen, Ge ’24; Simsek, Bendjeddou, Hsu '24]

SGD one-sample-at-a-time

One gradient update
for ***each*** new ***fresh*** sample

$$W^{\nu+1} = W^{\nu} - \gamma_\nu \nabla_{W^\nu}(y^\nu - f_{W^\nu}(\mathbf{x}^\nu))^2$$

What's going on in a nutshell : (here spherical GD)

Spherical gradient descent

$$\mathbf{w}_{t+1} = \frac{\mathbf{w}_t - \gamma \mathbf{g}_t^{\perp}}{\|\mathbf{w}_t - \gamma \mathbf{g}_t^{\perp}\|_2} \approx \mathbf{w}_t - \gamma \mathbf{g}_t^{\perp} - \gamma^2 \mathrm{C} \mathbf{w}_\mathrm{t}$$

What's going on in a nutshell : (here spherical GD)

Spherical gradient descent

$$\mathbf{w}_{t+1} = \frac{\mathbf{w}_t - \gamma \mathbf{g}_t^{\perp}}{\|\mathbf{w}_t - \gamma \mathbf{g}_t^{\perp}\|_2} \approx \mathbf{w}_t - \gamma \mathbf{g}_t^{\perp} - \gamma^2 \mathrm{C}\mathbf{w}_{\mathrm{t}}$$

Projection on the teacher vector

$$\mathbf{w}_{t+1} \cdot \mathbf{w}^{\star} \approx \mathbf{w}_t \cdot \mathbf{w}^{\star} - \gamma \mathbf{g}_t^{\perp} \cdot \mathbf{w}^{\star} - \gamma^2 \mathrm{C}\mathbf{w}_{\mathrm{t}} \cdot \mathbf{w}^{\star}$$

What's going on in a nutshell : (here spherical GD)

Spherical gradient descent

$$\mathbf{w}_{t+1} = \frac{\mathbf{w}_t - \gamma \mathbf{g}_t^{\perp}}{\|\mathbf{w}_t - \gamma \mathbf{g}_t^{\perp}\|_2} \approx \mathbf{w}_t - \gamma \mathbf{g}_t^{\perp} - \gamma^2 \mathrm{C}\mathbf{w}_\mathrm{t}$$

Projection on the teacher vector

$$\mathbf{w}_{t+1} \cdot \mathbf{w}^{\star} \approx \mathbf{w}_t \cdot \mathbf{w}^{\star} - \gamma \mathbf{g}_t^{\perp} \cdot \mathbf{w}^{\star} - \gamma^2 \mathrm{C}\mathbf{w}_\mathrm{t} \cdot \mathbf{w}^{\star}$$

$$m_{t+1} \approx m_t - \gamma \mathbf{g}_t^{\perp} \cdot \mathbf{w}^{\star} - \gamma^2 \mathrm{C}\mathrm{m}_\mathrm{t}$$

What's going on in a nutshell : (here spherical GD)

Spherical gradient descent

$$\mathbf{w}_{t+1} = \frac{\mathbf{w}_t - \gamma \mathbf{g}_t^{\perp}}{\|\mathbf{w}_t - \gamma \mathbf{g}_t^{\perp}\|_2} \approx \mathbf{w}_t - \gamma \mathbf{g}_t^{\perp} - \gamma^2 \mathrm{C}\mathbf{w}_\mathrm{t}$$

Projection on the teacher vector

$$\mathbf{w}_{t+1} \cdot \mathbf{w}^{\star} \approx \mathbf{w}_t \cdot \mathbf{w}^{\star} - \gamma \mathbf{g}_t^{\perp} \cdot \mathbf{w}^{\star} - \gamma^2 \mathrm{C}\mathbf{w}_\mathrm{t} \cdot \mathbf{w}^{\star}$$

$$m_{t+1} \approx m_t - \gamma \mathbf{g}_t^{\perp} \cdot \mathbf{w}^{\star} - \gamma^2 \mathrm{Cm}_\mathrm{t}$$

ODE on order parameter + Concentration

$$\dot{m}_t = -\mathbb{E}[\mathbf{g}_t^{\perp} \cdot \mathbf{w}^{\star}] - \gamma \mathrm{Cm}_\mathrm{t}$$

What's going on in a nutshell : (here spherical GD)

Spherical gradient descent

$$\mathbf{w}_{t+1} = \frac{\mathbf{w}_t - \gamma \mathbf{g}_t^{\perp}}{\|\mathbf{w}_t - \gamma \mathbf{g}_t^{\perp}\|_2} \approx \mathbf{w}_t - \gamma \mathbf{g}_t^{\perp} - \gamma^2 \mathrm{C}\mathbf{w}_{\mathrm{t}}$$

Projection on the teacher vector

$$\mathbf{w}_{t+1} \cdot \mathbf{w}^{\star} \approx \mathbf{w}_t \cdot \mathbf{w}^{\star} - \gamma \mathbf{g}_t^{\perp} \cdot \mathbf{w}^{\star} - \gamma^2 \mathrm{C}\mathbf{w}_{\mathrm{t}} \cdot \mathbf{w}^{\star}$$

$$m_{t+1} \approx m_t - \gamma \mathbf{g}_t^{\perp} \cdot \mathbf{w}^{\star} - \gamma^2 \mathrm{Cm}_{\mathrm{t}}$$

ODE on order parameter + Concentration

$$\dot{m}_t = -\mathbb{E}[\mathbf{g}_t^{\perp} \cdot \mathbf{w}^{\star}] - \gamma \mathrm{Cm}_{\mathrm{t}}$$

$$\dot{m}_t \approx \mathbb{E}_{\mathbf{x}}\left[g^{\star}(w^{\star} \cdot \mathbf{x})\sigma'(\mathbf{w}_t \cdot \mathbf{x})\mathbf{w}^{\star} \cdot \mathbf{x}\right] - C\gamma m_t$$

What's going on in a nutshell

$$\dot{m}_t \approx \mathbb{E}_{\mathbf{x}} \left[g^\star(w^\star \cdot \mathbf{x}) \sigma'(\mathbf{w}_t \cdot \mathbf{x}) \mathbf{w}^\star \cdot \mathbf{x} \right] - C\gamma m_t$$

What's going on in a nutshell

$$\dot{m}_t \approx \mathbb{E}_{\mathbf{x}}\left[g^\star(w^\star \cdot \mathbf{x})\sigma'(\mathbf{w}_t \cdot \mathbf{x})\mathbf{w}^\star \cdot \mathbf{x}\right] - C\gamma m_t$$

Gaussian vectors (aka fields)

$$\begin{pmatrix} h_t = \mathbf{w}^{(t)} \cdot \mathbf{x}_t \\ h^\star = \mathbf{w}^\star \cdot \mathbf{x}_t \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & m_t \\ m_t & 1 \end{pmatrix}\right)$$

$$\mathbb{E}_{h^t, h^\star}[g^\star(h^\star)\sigma'(h_t)h^\star]$$

What's going on in a nutshell

$$\dot{m}_t \approx \mathbb{E}_{\mathbf{x}}\left[g^\star(w^\star \cdot \mathbf{x})\sigma'(\mathbf{w}_t \cdot \mathbf{x})\mathbf{w}^\star \cdot \mathbf{x}\right] - C\gamma m_t$$

Gaussian vectors (aka fields)

$$\begin{pmatrix} h_t = \mathbf{w}^{(t)} \cdot \mathbf{x}_t \\ h^\star = \mathbf{w}^\star \cdot \mathbf{x}_t \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & m_t \\ m_t & 1 \end{pmatrix}\right)$$

$$\mathbb{E}_{h^t,h^\star}[g^\star(h^\star)\sigma'(h_t)h^\star]$$

Integration by part (aka Stein's lemma)

$$= \mathbb{E}_{h^t,h^\star}[g^{\star\prime}(h^\star)\sigma'(h_t)]$$

What's going on in a nutshell

$$\dot{m}_t \approx \mathbb{E}_{\mathbf{x}}\left[g^\star(w^\star \cdot \mathbf{x})\sigma'(\mathbf{w}_t \cdot \mathbf{x})\mathbf{w}^\star \cdot \mathbf{x}\right] - C\gamma m_t$$

Gaussian vectors (aka fields)

$$\begin{pmatrix} h_t = \mathbf{w}^{(t)} \cdot \mathbf{x}_t \\ h^\star = \mathbf{w}^\star \cdot \mathbf{x}_t \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & m_t \\ m_t & 1 \end{pmatrix}\right)$$

$$\mathbb{E}_{h^t,h^\star}[g^\star(h^\star)\sigma'(h_t)h^\star]$$

Integration by part (aka Stein's lemma)

$$= \mathbb{E}_{h^t,h^\star}[g^{\star\prime}(h^\star)\sigma'(h_t)]$$

Hermite expansion (Orthogonal basis for Gaussians)

$$= \sum_k g'_k \sigma'_k \mathbb{E}[H_k(h^\star)H_k(h_t)]$$

What's going on in a nutshell

$$\dot{m}_t \approx \mathbb{E}_{\mathbf{x}}\left[g^\star(w^\star \cdot \mathbf{x})\sigma'(\mathbf{w}_t \cdot \mathbf{x})\mathbf{w}^\star \cdot \mathbf{x}\right] - C\gamma m_t$$

Gaussian vectors (aka fields)

$$\begin{pmatrix} h_t = \mathbf{w}^{(t)} \cdot \mathbf{x}_t \\ h^\star = \mathbf{w}^\star \cdot \mathbf{x}_t \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & m_t \\ m_t & 1 \end{pmatrix}\right)$$

$$\mathbb{E}_{h^t,h^\star}[g^\star(h^\star)\sigma'(h_t)h^\star]$$

Integration by part (aka Stein's lemma)

$$= \mathbb{E}_{h^t,h^\star}[g^{\star\prime}(h^\star)\sigma'(h_t)]$$

Hermite expansion
(Orthogonal basis for Gaussians)

$$= \sum_k g_k'\sigma_k'\mathbb{E}[H_k(h^\star)H_k(h_t)]$$

Expectation is just
the correlation!

$$= \sum_k g_k'\sigma_k' m_t^k$$

What's going on in a nutshell

$$\dot{m}_t \approx \mathbb{E}_{\mathbf{x}}\left[g^\star(w^\star \cdot \mathbf{x})\sigma'(\mathbf{w}_t \cdot \mathbf{x})\mathbf{w}^\star \cdot \mathbf{x}\right] - C\gamma m_t$$

Gaussian vectors (aka fields)

$$\begin{pmatrix} h_t = \mathbf{w}^{(t)} \cdot \mathbf{x}_t \\ h^\star = \mathbf{w}^\star \cdot \mathbf{x}_t \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & m_t \\ m_t & 1 \end{pmatrix}\right)$$

$$\mathbb{E}_{h^t,h^\star}[g^\star(h^\star)\sigma'(h_t)h^\star]$$

Integration by part (aka Stein's lemma)

$$= \mathbb{E}_{h^t,h^\star}[g^{\star\prime}(h^\star)\sigma'(h_t)]$$

Hermite expansion
(Orthogonal basis for Gaussians)

$$= \sum_k g'_k \sigma'_k \mathbb{E}[H_k(h^\star)H_k(h_t)]$$

Expectation is just
the correlation!

$$= \sum_k g'_k \sigma'_k m_t^k$$

Dominated by the first
non-zero Hermite coefficient of g*

$$\propto \text{Cst } m_t^{\ell-1}$$

What's going on in a nutshell

$$\dot{m}_t \approx \mathrm{Cst}\, m_t^{\ell-1} - C\gamma m_t$$

Theorem [Ben Arous et al '22]

$\ell = 1$ $\quad \tau = n = \mathcal{O}(d)$

$\ell = 2$ $\quad \tau = n = \mathcal{O}(d \log d)$

$\ell > 2$ $\quad \tau = n = \mathcal{O}(d^{\ell-1})$

Information exponent ℓ

ℓ is defined as the order of the first non-zero coefficient in the Hermite expansion of $g^\star(\mathbf{h}^\star)$

$Ex : g^\star = H_2(h^\star) = (h^\star)^2 - 1$ has $\ell = 2$

$Ex : g^\star = H_3(h^\star) = h^{\star 3} - 3h^\star$ has $\ell = 3$

Hermite decomposition

$$f^\star(\mathbf{x}) = g^\star(h^\star) = \mathrm{cst} + \mu^{(1)} h^\star + \mu^{(2)} H_2(h^\star) + \mu^{(3)} H_3(h^\star) + \dots$$

This is somehow disappointing

SGD is suboptimal: CSQ vs SQ class

SGD/Correlational Statistical Queries (CSQ) bounds/ Information exponent

$$\mathbb{E}[Y\phi(\mathbf{Z})] = ?$$

Denote ℓ as the order of the first non-zero Hermite coefficient, then

$$n = \mathcal{O}(d^{\max(1,\frac{\ell}{2})})$$

$\ell = 1$	$n = \mathcal{O}(d)$
$\ell = 2$	$n = \mathcal{O}(d \log d)$
$\ell > 2$	$n = \mathcal{O}(d^{\frac{\ell}{2}})$

Hermite decomposition

$$f^\star(\mathbf{x}) = g^\star(h^\star) = \mathrm{cst} + \mu^{(1)}h^\star + \mu^{(2)}H_2(h^\star) + \mu^{(3)}H_3(h^\star) + \dots$$

AMP/ Statistical Queries (SQ) bounds / Generative exponents

$$Y \sim P^\star(Y \,|\, H = W^\star \mathbf{Z})$$

$$\mathbb{E}[\phi(Y, \mathbf{Z})] = ?$$

TRIVIAL

W* can be learned with *any* $n = \mathcal{O}(d)$ **if**

$$\mathbb{E}[H \,|\, Y] \neq 0$$

with non-zero probability over y

EASY

For even target (or different symmetries for multi-index) learning W* requires $n > \alpha_c d$

$$\alpha_c = \mathbb{E}[(\mathbb{E}[H \,|\, Y]^2 - 1)^2]^{-1}$$

HARD

Very restricted **set of hard functions ($\alpha_d \to \infty$) require more than $\mathcal{O}(d)$ data!**

Example : r-parity

$$y = \mathrm{sign}(h_1^\star h_2^\star \dots h_r^\star)$$

Investigated in detail by [Abbe, Boix-Adsera & Misiakiewicz, '21+'22+'23]

Multi-index : not much changes except …

Hermite decomposition : Each direction now has its own exponent (leap exponent)

$$f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star) = \text{cst} + \sum_i \mu_i^{(1)} h_i^\star + \sum_{ij} \mu_{ij}^{(2)} H_2(h_i^\star, h_j^\star) + \sum_{ijk} \mu_{ijk}^{(3)} H_3(h_i^\star, h_j^\star, h_k^\star) + \dots$$

Investigated in detail by [Abbe, Boix-Adsera & Misiakiewicz, '21+'22+'23]

Multi-index : not much changes except …

Hermite decomposition : Each direction now has its own exponent (leap exponent)

$$f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star) = \text{cst} + \sum_i \mu_i^{(1)} h_i^\star + \sum_{ij} \mu_{ij}^{(2)} H_2(h_i^\star, h_j^\star) + \sum_{ijk} \mu_{ijk}^{(3)} H_3(h_i^\star, h_j^\star, h_k^\star) + \ldots$$

[Abbe et al,'22]

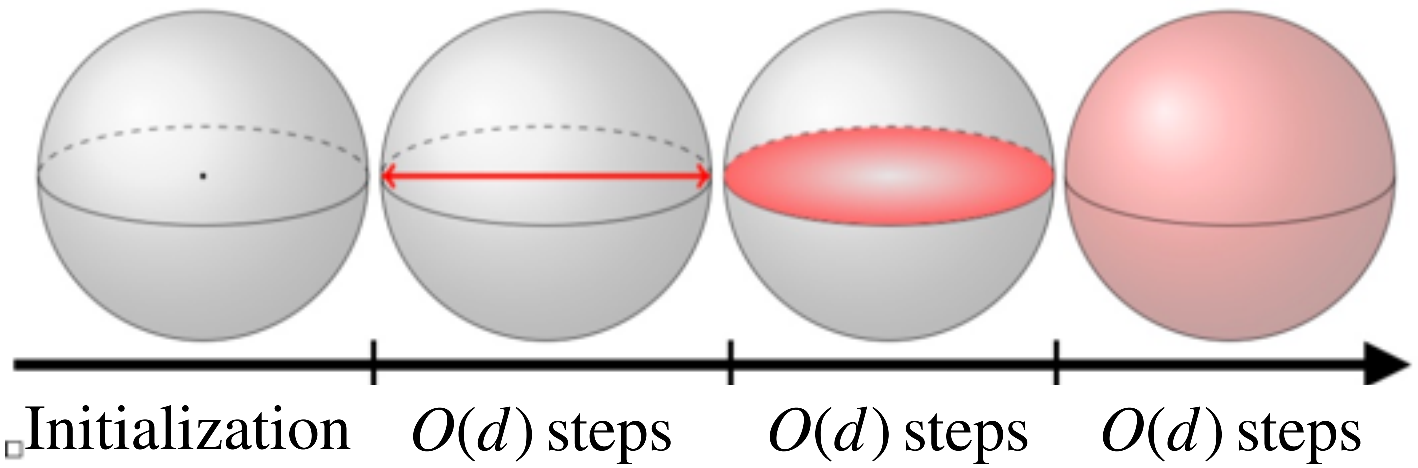
Investigated in detail by [Abbe, Boix-Adsera & Misiakiewicz, '21+'22+'23]

Hierarchical iterative learning of directions

Hermite decomposition : Each direction now has its own exponent (leap exponent)

$$f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star) = \mathrm{cst} + \sum_i \mu_i^{(1)} h_i^\star + \sum_{ij} \mu_{ij}^{(2)} H_2(h_i^\star, h_j^\star) + \sum_{ijk} \mu_{ijk}^{(3)} H_3(h_i^\star, h_j^\star, h_k^\star) + \ldots$$

$$y = g^\star(h_1^\star, h_2^\star, h_3^\star) = h_1^\star + f(h_1^\star) h_2^\star + f(h_2^\star) h_3^\star$$



Informally :

One can learn ***new directions*** over time, iff they are ***linear conditioned*** **on the previously learned ones.**

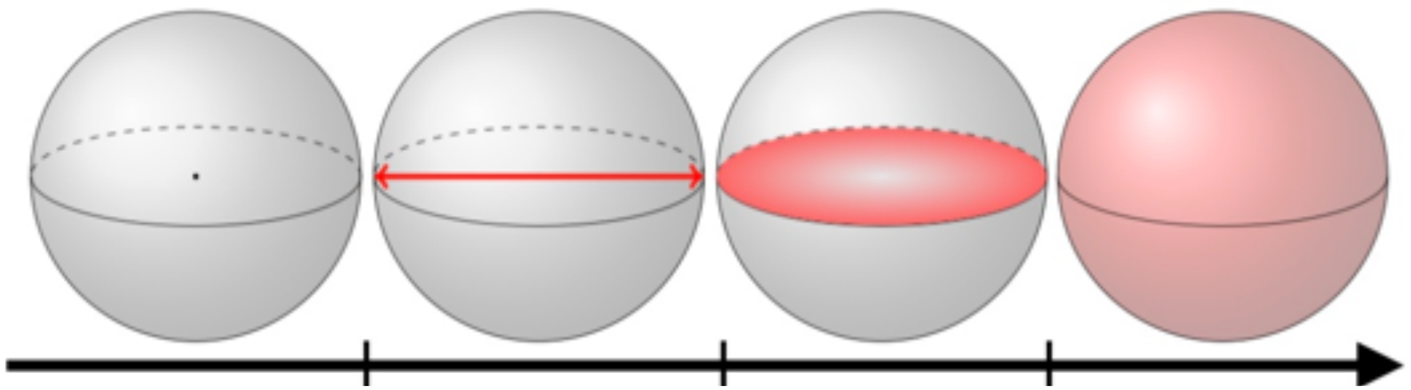
Investigated in detail by [Abbe, Boix-Adsera & Misiakiewicz, '21+'22+'23]

Hierarchical iterative learning of directions

Hermite decomposition : Each direction now has its own exponent (leap exponent)

$$f^\star(\mathbf{x}) = g^\star(\mathbf{h}^\star) = \mathrm{cst} + \sum_i \mu_i^{(1)} h_i^\star + \sum_{ij} \mu_{ij}^{(2)} H_2(h_i^\star, h_j^\star) + \sum_{ijk} \mu_{ijk}^{(3)} H_3(h_i^\star, h_j^\star, h_k^\star) + \dots$$

$$y = g^\star(h_1^\star, h_2^\star, h_3^\star) = h_1^\star + f(h_1^\star) h_2^\star + f(h_2^\star) h_3^\star$$



Informally :

One can learn ***new directions*** over time, iff they are ***linear conditioned*** **on the previously learned ones.**

EX:

$$y = h_1^\star + \left[(h_1^\star)^3 - 3h_1^\star\right] h_2^\star + \left[(h_2^\star)^3 - 3h_2^\star)\right] h_3^\star$$

Investigated in detail by [Abbe, Boix-Adsera & Misiakiewicz, '21+'22+'23]

Are neural net trained with gradient methods that sub-optimal?

Are neural net trained with gradient methods that sub-optimal?

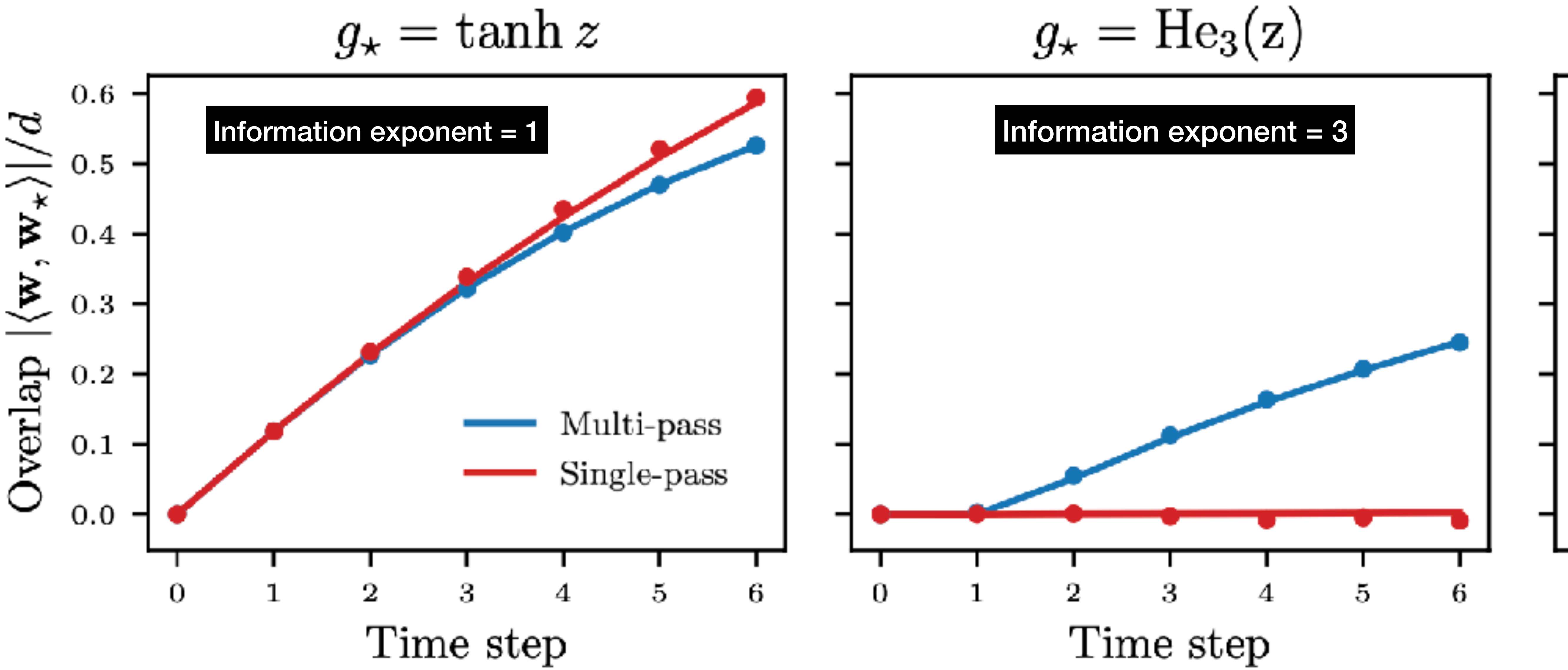
Wait! This was for online learning, with a fresh new sample at a time…

Are neural net trained with gradient methods that sub-optimal?

Wait! This was for online learning, with a fresh new sample at a time…

… what if instead we repeat gradient descent over a fixed large batch?

Fixed n_b=O(n) batch can learn $\ell > 1$ functions in 2 iterations!

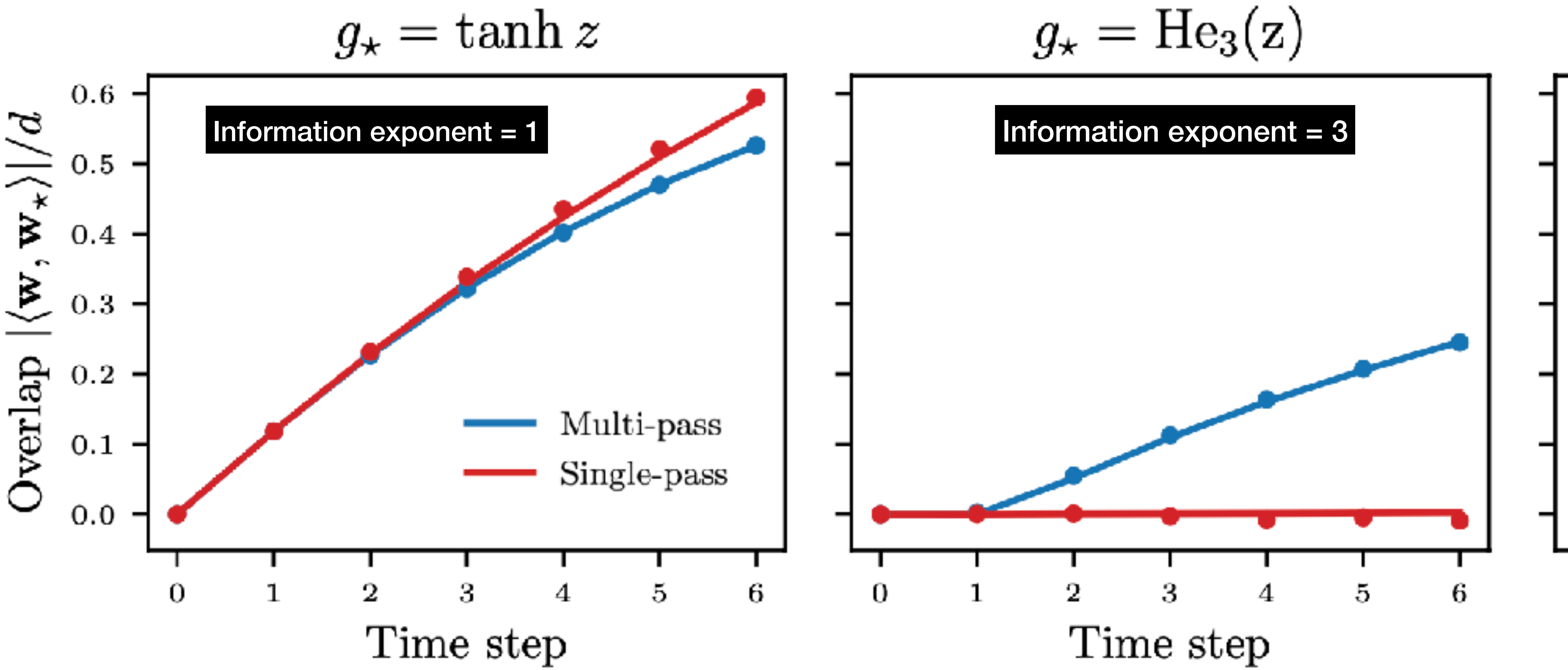


$n_b = 3d$ p=1

d=5000, with σ=relu, γ=0.1

$$W^{t+1} = W^t - \gamma_t \frac{1}{n_B} \sum_{\nu=1}^{n_B} \nabla_{W^t} (y^\nu - f_{W^t}(\mathbf{z}^\nu))^2$$

Theorem *(informal)* [Dandi, Pesce, Troiani, Zdeborova, FK '24]



TRIVIAL

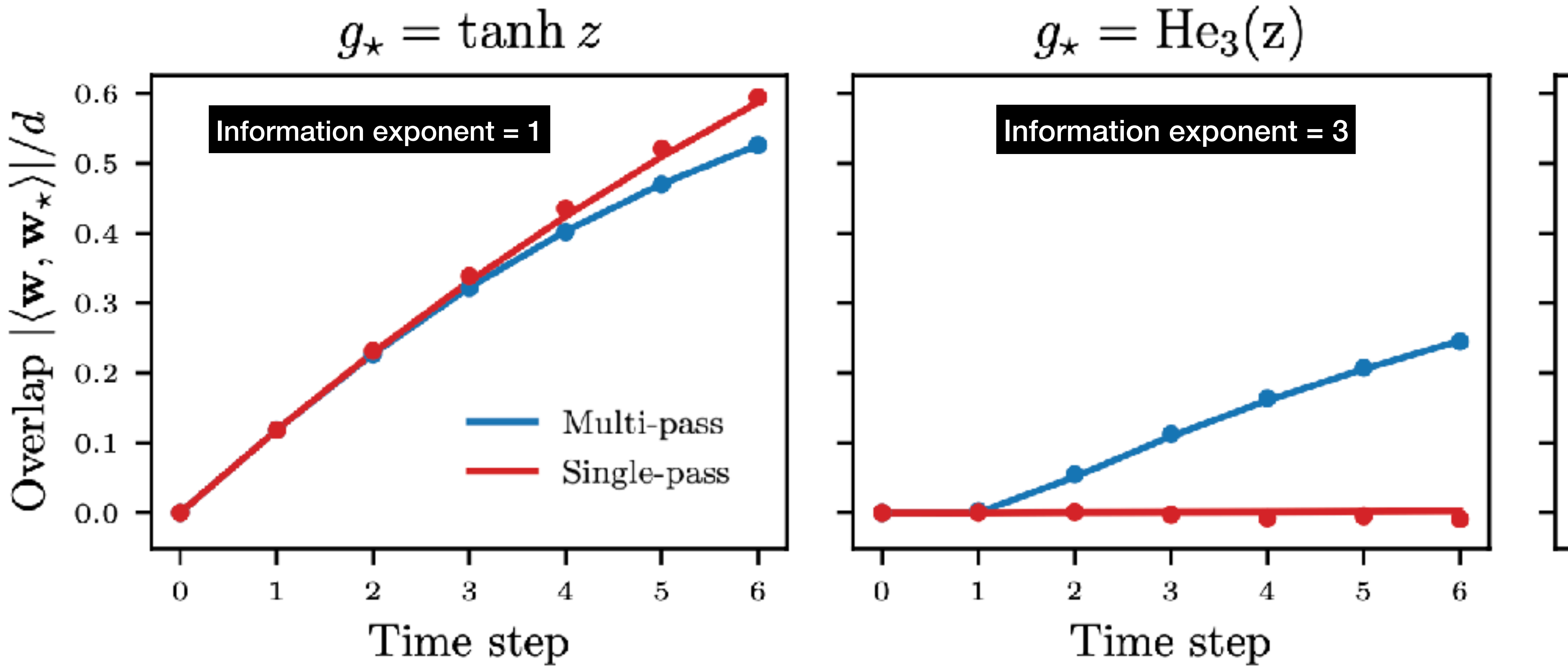
W* can be learned with *any*

$n = \mathcal{O}(d)$ **if**

$$\mathbb{E}[H \,|\, Y] \neq 0$$

with non-zero probability over y

Theorem *(informal)* [Dandi, Pesce, Troiani, Zdeborova, FK '24]

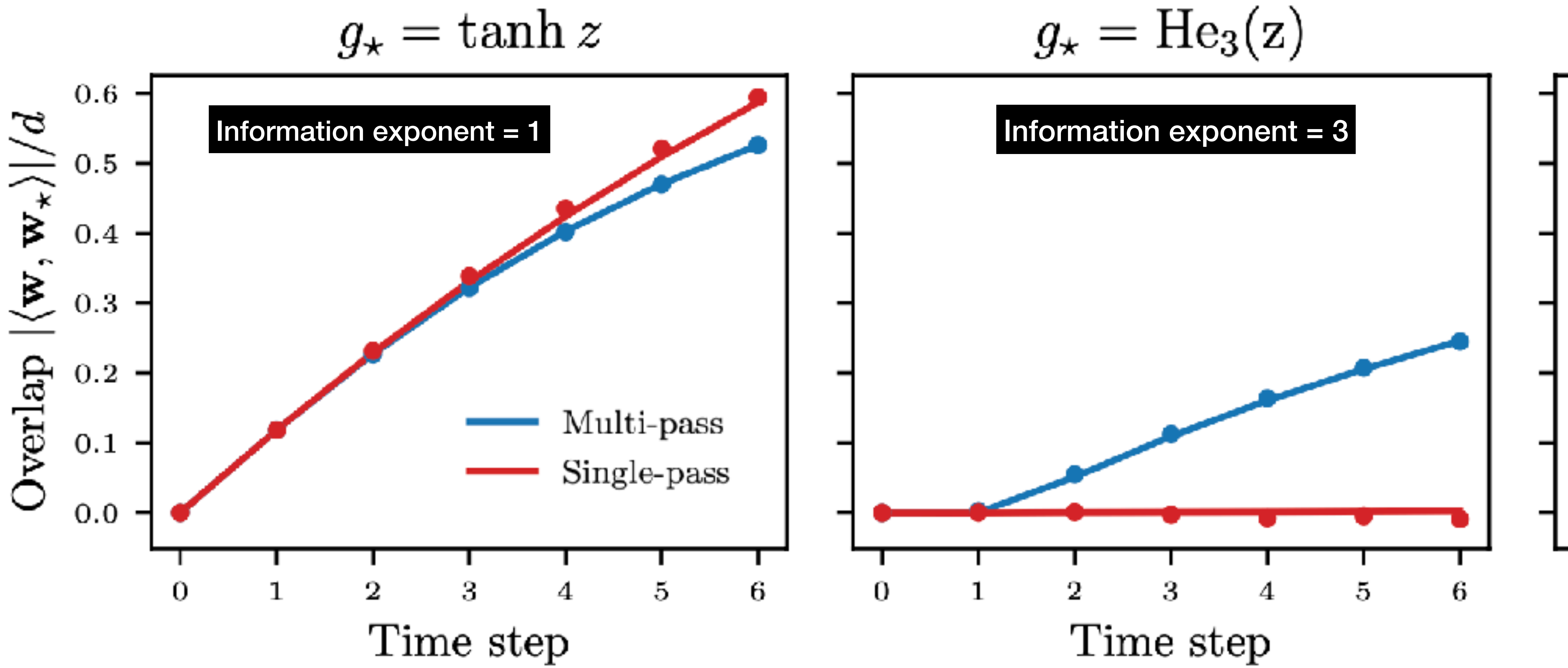


TRIVIAL

W* can be learned with *any*
$n = \mathcal{O}(d)$ **if**
$$\mathbb{E}[H \,|\, Y] \neq 0$$
with non-zero probability over y

W* can be learned by shallow neural ***nets in*** $n = \mathcal{O}(d)$, with <u>just 2 full batches iterations</u>!

Theorem *(informal)* [Dandi, Pesce, Troiani, Zdeborova, FK ’24]



TRIVIAL

W* can be learned with *any*
$n = \mathcal{O}(d)$ **if**
$$\mathbb{E}[H \,|\, Y] \neq 0$$
with non-zero probability over y

EASY

For even target (or different symmetries for multi-index) learning W* requires $n > \alpha_c d$
$$\alpha_c = \mathbb{E}[(\mathbb{E}[H \,|\, Y]^2 - 1)^2]^{-1}$$

Conjecture

W* can be learned by shallow neural *nets in* $n = \mathcal{O}(d)$, with just O(log d) full batches iterations!
for large enough $\alpha > \alpha_c$

REPETITION

IS THE MOTHER

of learning

Can we make this even more general?

[Arnaboldi, Dandi, Pesce, Stephan, **FK** '24], see also [Lee, Oko, Suzuki, Wu '24] for the single index case

Data repetition

Remark 1

Real dataset are never i.i.d. and data repetition of the same datapoint, or a very similar one is bound to occur

Remark 2

Many deep learning SGD algorithm are actually performing multiple steps over the same datapoint, e.g. Extra-gradient, Look-ahead GD, or Sharp Minima Aware gradient descent

SGD

$$W^{\nu+1} = W^{\nu} - \gamma \nabla \mathscr{L}(\mathbf{z}^{\nu}, W^{\nu})$$

SGD with extra-gradient

$$W^{\nu+1} = W^{\nu} - \gamma \nabla \mathscr{L}(\mathbf{z}^{\nu}, W^{\nu} - \tilde{\gamma} \nabla \mathscr{L}((\mathbf{z}^{\nu}, W^{\nu}))$$

Data repetition

Remark 1

Real dataset are never i.i.d. and data repetition of the same datapoint, or a very similar one is bound to occur

Remark 2

Many deep learning SGD algorithm are actually performing multiple steps over the same datapoint, e.g. Extra-gradient, Look-ahead GD, or Sharp Minima Aware gradient descent

Two SGD steps with the same data

SGD

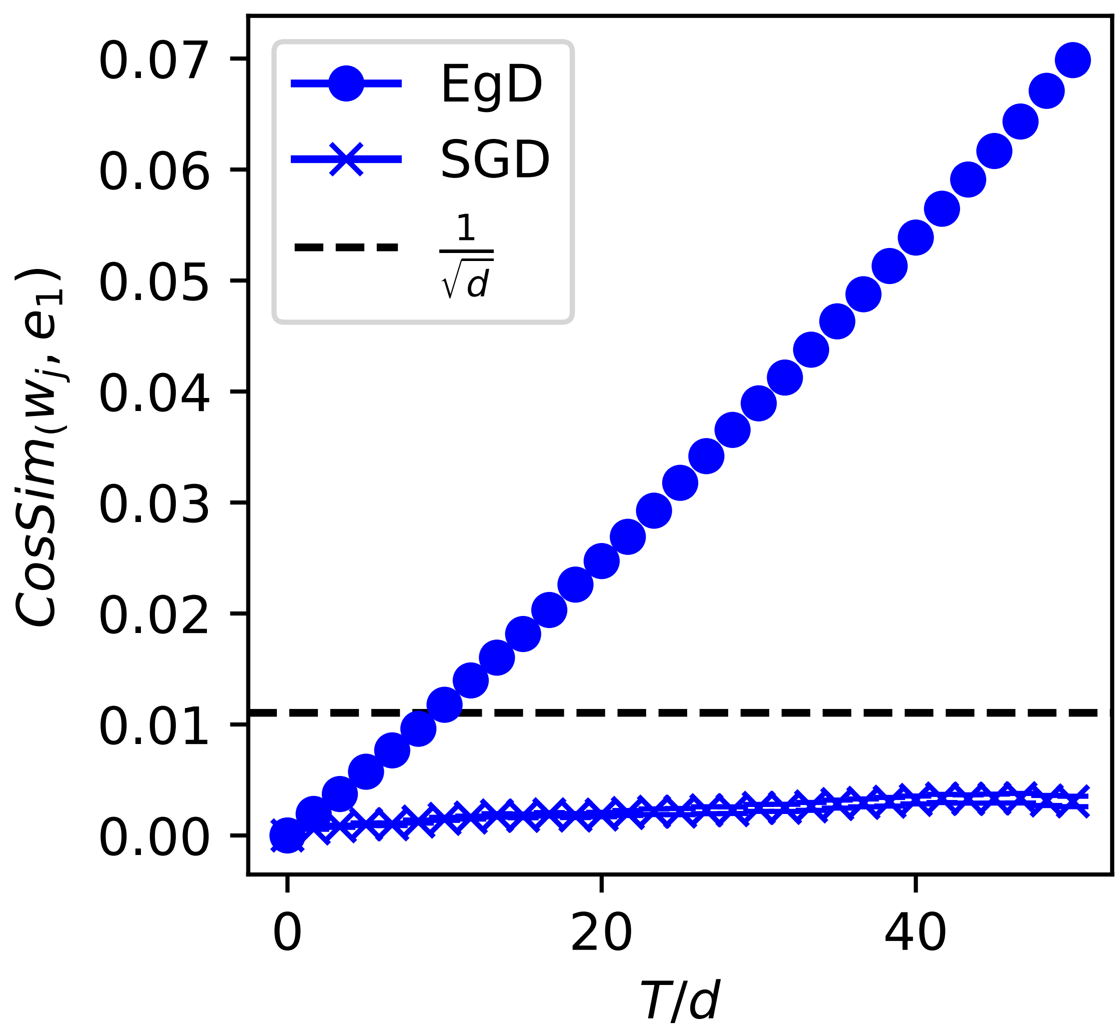
$$W^{\nu+1} = W^{\nu} - \gamma \nabla \mathscr{L}(\mathbf{z}^{\nu}, W^{\nu})$$

SGD with extra-gradient

$$W^{\nu+1} = W^{\nu} - \gamma \nabla \mathscr{L}(\mathbf{z}^{\nu}, W^{\nu} - \tilde{\gamma} \nabla \mathscr{L}((\mathbf{z}^{\nu}, W^{\nu}))$$

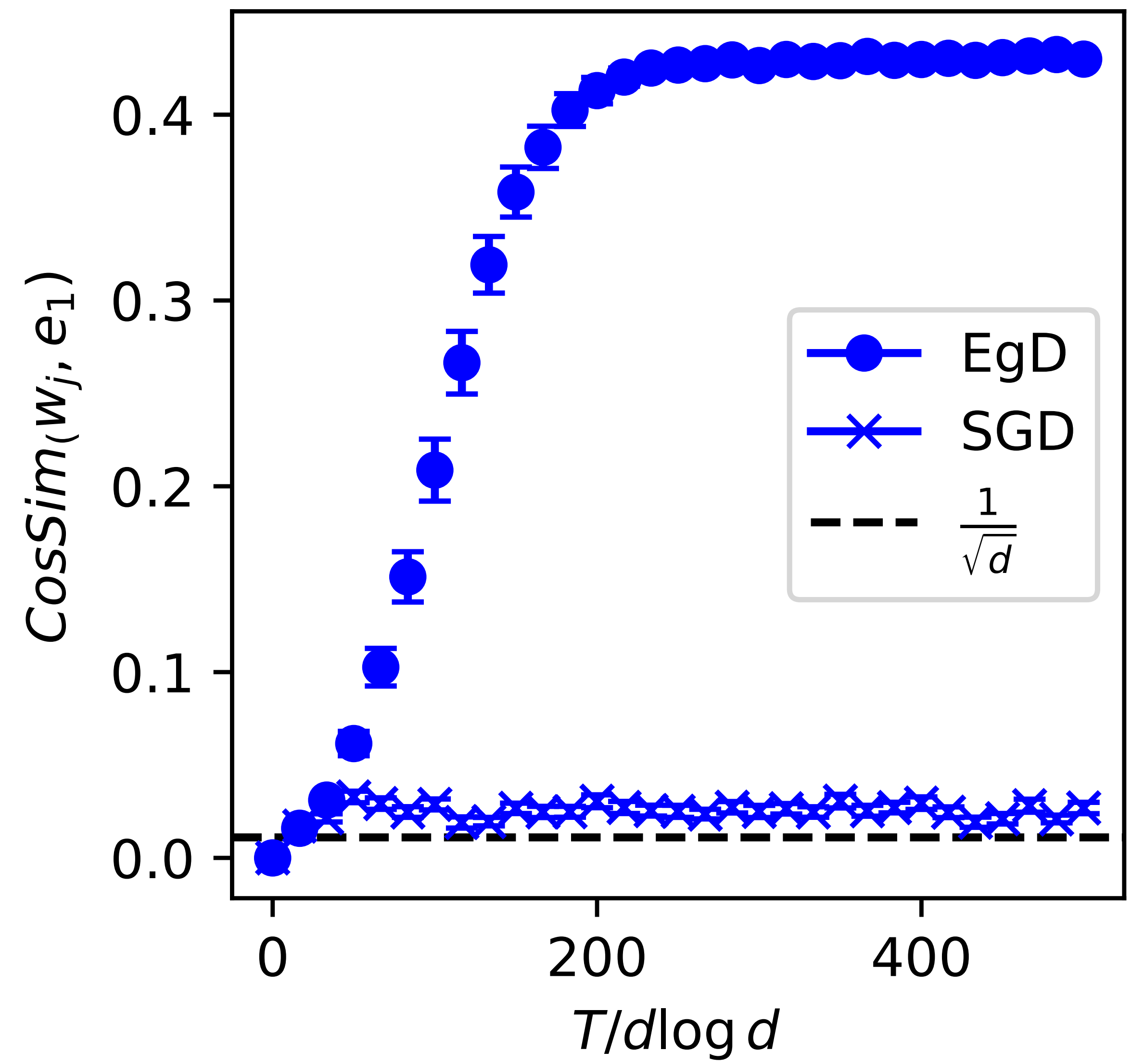
Repetuta iuvant

$$y = g^{\star}(h^{\star}) = (h^{\star})^3 - 3h^{\star}$$



SGD

$$y = g^{\star}(h^{\star}) = (h^{\star})^4 - 6(h^{\star})^2 + 3$$



SGD with extra-gradient

$$W^{\nu+1} = W^{\nu} - \gamma \nabla \mathscr{L}(\mathbf{z}^{\nu}, W^{\nu}) \quad \Rightarrow \quad W^{\nu+1} = W^{\nu} - \gamma \nabla \mathscr{L}(\mathbf{z}^{\nu}, W^{\nu} - \tilde{\gamma} \nabla \mathscr{L}((\mathbf{z}^{\nu}, W^{\nu}))$$

[Arnaboldi, Dandi, Pesce, Stephan, **FK** ’24], see also [Lee, Oko, Suzuki, Wu ’24] for the single index case

Main theorem (informal, some part still open)

AMP/ Statistical Queries (**SQ**) bounds / Generative exponents

$$\mathbb{E}[\phi(Y, \mathbf{Z})] = ?$$

$$Y \sim P^\star(Y \mid H = W^\star \mathbf{Z})$$

TRIVIAL

W* can be learned with *any*

$n = \mathcal{O}(d)$ **if**

$$\mathbb{E}[H \mid Y] \neq 0$$

with non-zero probability over y

EASY

For even target (or different symmetry for multi-index) learning W* requires $n > \alpha_c d$

$$\alpha_c = \mathbb{E}[(\mathbb{E}[H \mid Y]^2 - 1)^2]^{-1}$$

HARD

Very restricted **set of hard functions** ($\alpha_d \to \infty$) **require more than** $\mathcal{O}(d)$ **data!**

Example : r-partity

$$y = \mathrm{sign}(h_1^\star h_2^\star \ldots h_r^\star)$$

[Arnaboldi, Dandi, Pesce, Stephan, **FK** '24], see also [Lee, Oko, Suzuki, Wu '24] for the single index case

Main theorem (informal, some part still open)

Target without symmetries

W* can be learned by shallow neural nets with $\tau = n = \mathcal{O}(d)$ using extragradient algorithms

AMP/ Statistical Queries (SQ) bounds / Generative exponents

$$\mathbb{E}[\phi(Y, \mathbf{Z})] = ?$$

$$Y \sim P^\star(Y \mid H = W^\star \mathbf{Z})$$

TRIVIAL

W* can be learned with *any* $n = \mathcal{O}(d)$ **if**

$$\mathbb{E}[H \mid Y] \neq 0$$

with non-zero probability over y

EASY

For even target (or different symmetry for multi-index) learning W* requires $n > \alpha_c d$

$$\alpha_c = \mathbb{E}[(\mathbb{E}[H \mid Y]^2 - 1)^2]^{-1}$$

HARD

Very restricted **set of hard functions** ($\alpha_d \to \infty$) **require more than** $\mathcal{O}(d)$ **data!**

Example : r-partity

$$y = \mathrm{sign}(h_1^\star h_2^\star \dots h_r^\star)$$

[Arnaboldi, Dandi, Pesce, Stephan, **FK** '24], see also [Lee, Oko, Suzuki, Wu '24] for the single index case

Main theorem (informal, some part still open)

Target without symmetries

W* can be learned by shallow neural nets with $\tau = n = \mathcal{O}(d)$ using extragradient algorithms

Target with symmetries

W* can be learned by 2LLN with $\tau = n = \mathcal{O}(d \log d)$ using extragradient algorithms

(Still not completely proved for multi-index models)*

AMP/ Statistical Queries (SQ) bounds / Generative exponents

$$\mathbb{E}[\phi(Y, \mathbf{Z})] = ?$$

$$Y \sim P^\star(Y | H = W^\star \mathbf{Z})$$

TRIVIAL

W* can be learned with ***any*** $n = \mathcal{O}(d)$ if

$$\mathbb{E}[H | Y] \neq 0$$

with non-zero probability over y

EASY

For even target (or different symmetry for multi-index) learning W* requires $n > \alpha_c d$

$$\alpha_c = \mathbb{E}[(\mathbb{E}[H | Y]^2 - 1)^2]^{-1}$$

HARD

Very restricted set of hard functions ($\alpha_d \to \infty$) require more than $\mathcal{O}(d)$ data!

Example : r-partity

$$y = \mathrm{sign}(h_1^\star h_2^\star \ldots h_r^\star)$$

[Arnaboldi, Dandi, Pesce, Stephan, **FK** '24], see also [Lee, Oko, Suzuki, Wu '24] for the single index case

Main theorem (informal, some part still open)

Target without symmetries

W* can be learned by shallow neural nets with $\tau = n = \mathcal{O}(d)$ using extragradient algorithms

Target with symmetries

W* can be learned by 2LLN with $\tau = n = \mathcal{O}(d \log d)$ using extragradient algorithms

(Still not completely proved for multi-index models)*

Hard target functions

For hard problems such as parities, W* can be learned by shallow neural with $\tau = n = \mathcal{O}(d^{r-1})$ using extragradient

(open)*

AMP/ Statistical Queries (SQ) bounds / Generative exponents

$$\mathbb{E}[\phi(Y, \mathbf{Z})] = ?$$

$$Y \sim P^\star(Y \mid H = W^\star \mathbf{Z})$$

TRIVIAL

W* can be learned with ***any***

$n = \mathcal{O}(d)$ if

$$\mathbb{E}[H \mid Y] \neq 0$$

with non-zero probability over y

EASY

For even target (or different symmetry for multi-index) learning W* requires $n > \alpha_c d$

$$\alpha_c = \mathbb{E}[(\mathbb{E}[H \mid Y]^2 - 1)^2]^{-1}$$

HARD

Very restricted set of hard functions ($\alpha_d \to \infty$) require more than $\mathcal{O}(d)$ data!

Example : r-partity

$$y = \mathrm{sign}(h_1^\star h_2^\star \ldots h_r^\star)$$

[Arnaboldi, Dandi, Pesce, Stephan, **FK** '24], see also [Lee, Oko, Suzuki, Wu '24] for the single index case

Why repetition works? Remember this ?

$$\dot{m}_t \approx \mathbb{E}_{\mathbf{x}} \left[g^\star(w^\star \cdot \mathbf{x}) \sigma'(\mathbf{w}_t \cdot \mathbf{x}) \mathbf{w}^\star \cdot \mathbf{x} \right] - C\gamma m_t$$

Why repetition works? Remember this ?

$$\dot{m}_t \approx \mathbb{E}_{\mathbf{x}}\left[g^\star(w^\star \cdot \mathbf{x})\sigma'(\cancel{\mathbf{w}_t} \cdot \mathbf{x})\mathbf{w}^\star \cdot \mathbf{x}\right] - C\gamma m_t$$

Why repetition works? Remember this ?

$$\dot{m}_t \approx \mathbb{E}_{\mathbf{x}}\left[g^\star(w^\star \cdot \mathbf{x})\sigma'(\cancel{\mathbf{w}_t} \cdot \mathbf{x})\mathbf{w}^\star \cdot \mathbf{x}\right] - C\gamma m_t$$

Slightly different with extra-gradient!

$$\mathbb{E}\left[g^\star(h^\star)\sigma'\left(\left(\mathbf{w}_t - \gamma\mathbf{g}^{\mathbf{t}}\right)\cdot\mathbf{x}\right)h^\star\right]$$

Why repetition works? Remember this ?

$$\dot{m}_t \approx \mathbb{E}_{\mathbf{x}}\left[g^\star(w^\star \cdot \mathbf{x})\sigma'(\cancel{\mathbf{w}_t} \cdot \mathbf{x})\mathbf{w}^\star \cdot \mathbf{x}\right] - C\gamma m_t$$

Slightly different with extra-gradient!

$$\mathbb{E}\left[g^\star(h^\star)\sigma'\left(\left(\mathbf{w}_t - \gamma \mathbf{g}^{\mathbf{t}}\right)\cdot \mathbf{x}\right)h^\star\right]$$

It now reads

$$= \mathbb{E}\left[g^\star(h^\star)\sigma'\left(h_t + \gamma g^\star(h^\star)\sigma'\left(h_t\right)\right)h^\star\right]$$

Why repetition works? Remember this ?

$$\dot{m}_t \approx \mathbb{E}_{\mathbf{x}}\left[g^\star(w^\star \cdot \mathbf{x})\sigma'(\cancel{\mathbf{w}_t} \cdot \mathbf{x})\mathbf{w}^\star \cdot \mathbf{x}\right] - C\gamma m_t$$

Slightly different with extra-gradient!

$$\mathbb{E}\left[g^\star(h^\star)\sigma'\left(\left(\mathbf{w}_t - \gamma \mathbf{g}^\mathbf{t}\right) \cdot \mathbf{x}\right) h^\star\right]$$

It now reads

$$= \mathbb{E}\left[g^\star(h^\star)\sigma'\left(h_t + \gamma g^\star(h^\star)\sigma'\left(h_t\right)\right) h^\star\right]$$

Allows arbitrary polynomial transformation of the teacher!

$$= \mathbb{E}\left[g^\star(h^\star)\left(\sum_k \alpha_k(h_t) g^\star(h^\star)^k\right) h^\star\right]$$

Why repetition works? Remember this ?

$$\dot{m}_t \approx \mathbb{E}_{\mathbf{x}}\left[g^\star(w^\star \cdot \mathbf{x})\sigma'(\cancel{\mathbf{w}_t} \cdot \mathbf{x})\mathbf{w}^\star \cdot \mathbf{x}\right] - C\gamma m_t$$

Slightly different with extra-gradient!

$$\mathbb{E}\left[g^\star(h^\star)\sigma'\left(\left(\mathbf{w}_t - \gamma \mathbf{g}^{\mathbf{t}}\right)\cdot \mathbf{x}\right)h^\star\right]$$

It now reads

$$= \mathbb{E}\left[g^\star(h^\star)\sigma'\left(h_t + \gamma g^\star(h^\star)\sigma'\left(h_t\right)\right)h^\star\right]$$

Allows arbitrary polynomial transformation of the teacher!

$$= \mathbb{E}\left[g^\star(h^\star)\left(\sum_k \alpha_k(h_t) g^\star(h^\star)^k\right)h^\star\right]$$

Correlational Statistical Queries (CSQ) bounds

$$\mathbb{E}[Y\phi(\mathbf{Z})] = ?$$

Statistical Queries (SQ) bounds

$$\mathbb{E}[\phi(Y, \mathbf{Z})] = ?$$

CSQ staircase vs Grand staircase

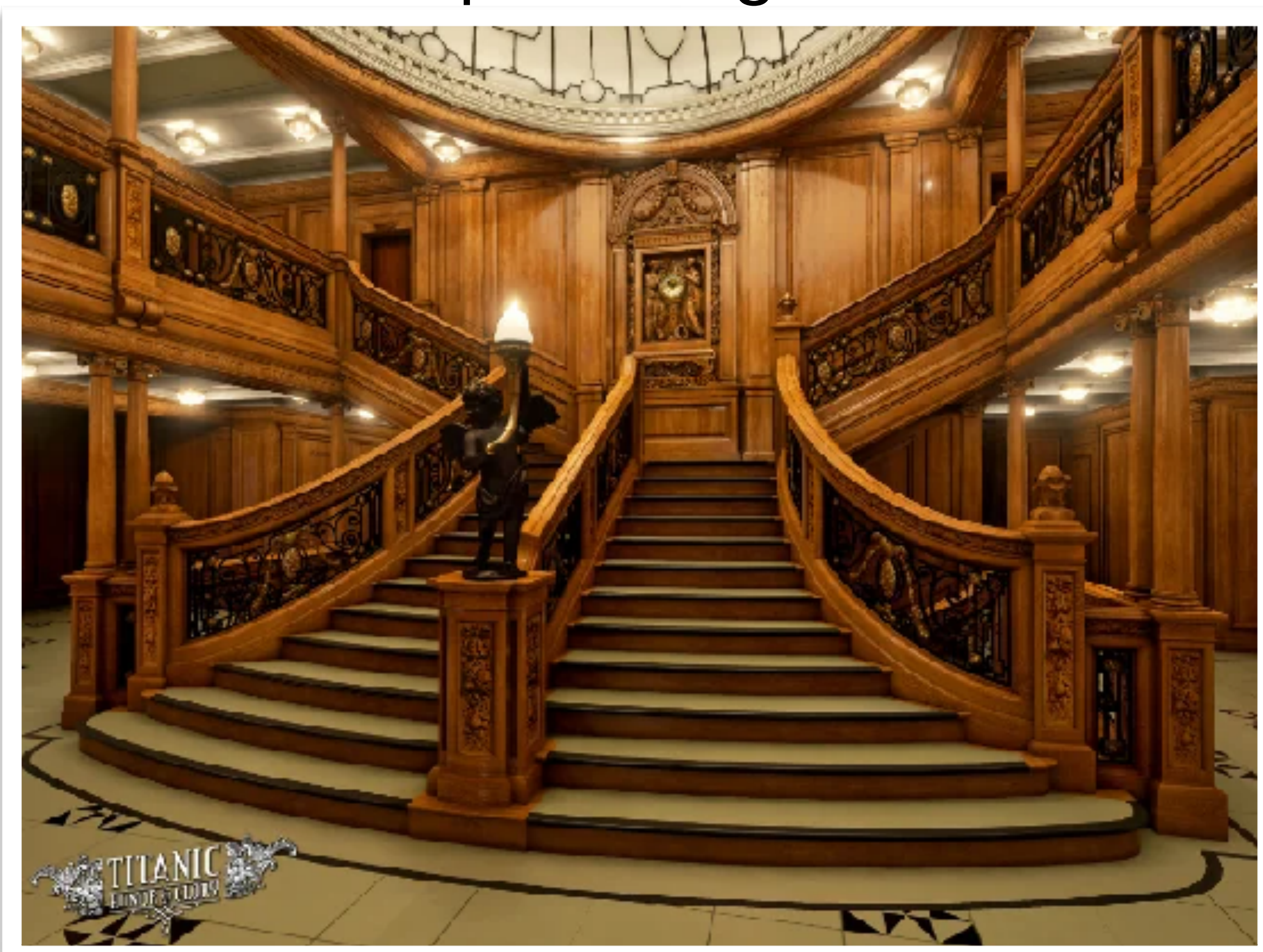
Without repetition

Information exponent/CSQ staircase

[Abbe et al,’22+’23]

With repetition

Generative exponent/ grand staircase

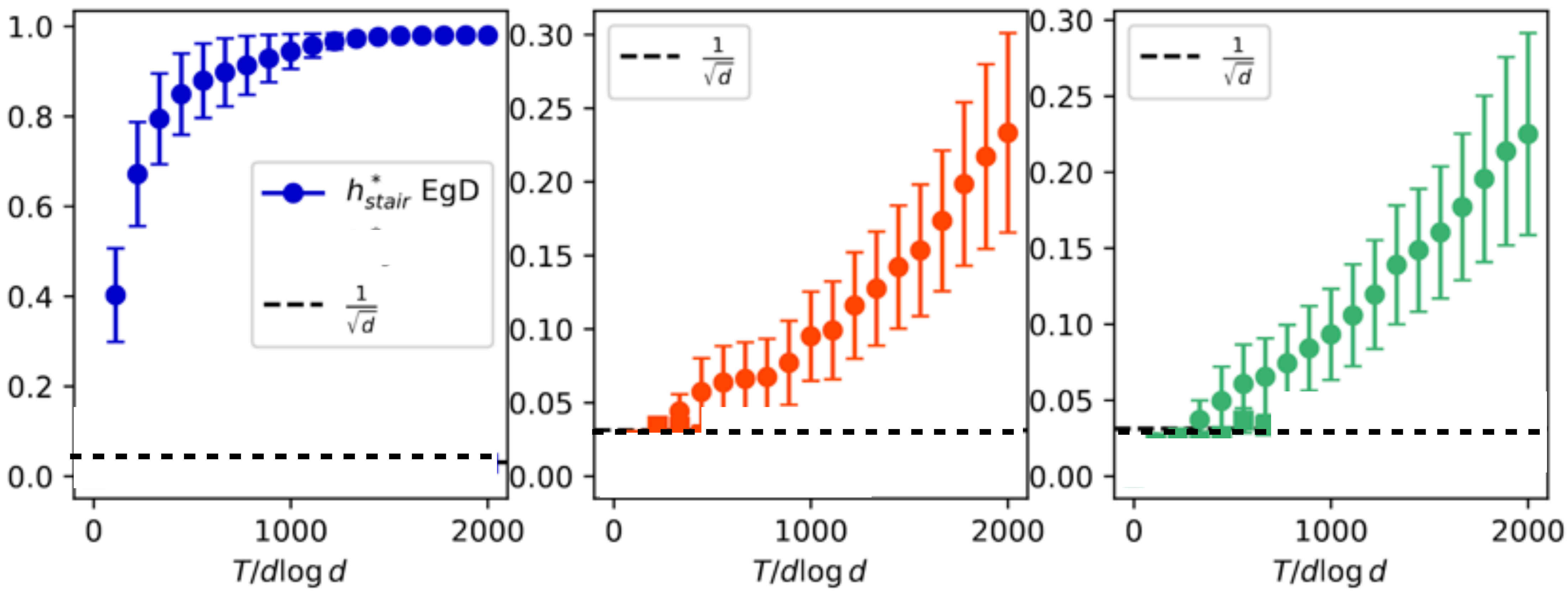


[Troiani, Dandi, Delilippis, Zdeborova, Loureiro, **FK**, ’24]

Example #1 : a standard staircase

$$y = (h_1^\star)^2 + \mathrm{sign}(h_1^\star h_2^\star h_3^\star)$$

Can be learned in O(dog d) steps with and without repetition



$$f^\star_{\mathrm{stair}}(\boldsymbol{z}) = \mathrm{He}_2(z_1) + \mathrm{sign}(z_1 z_2 z_3)$$

First we learn $h_1^\star$ in d log d

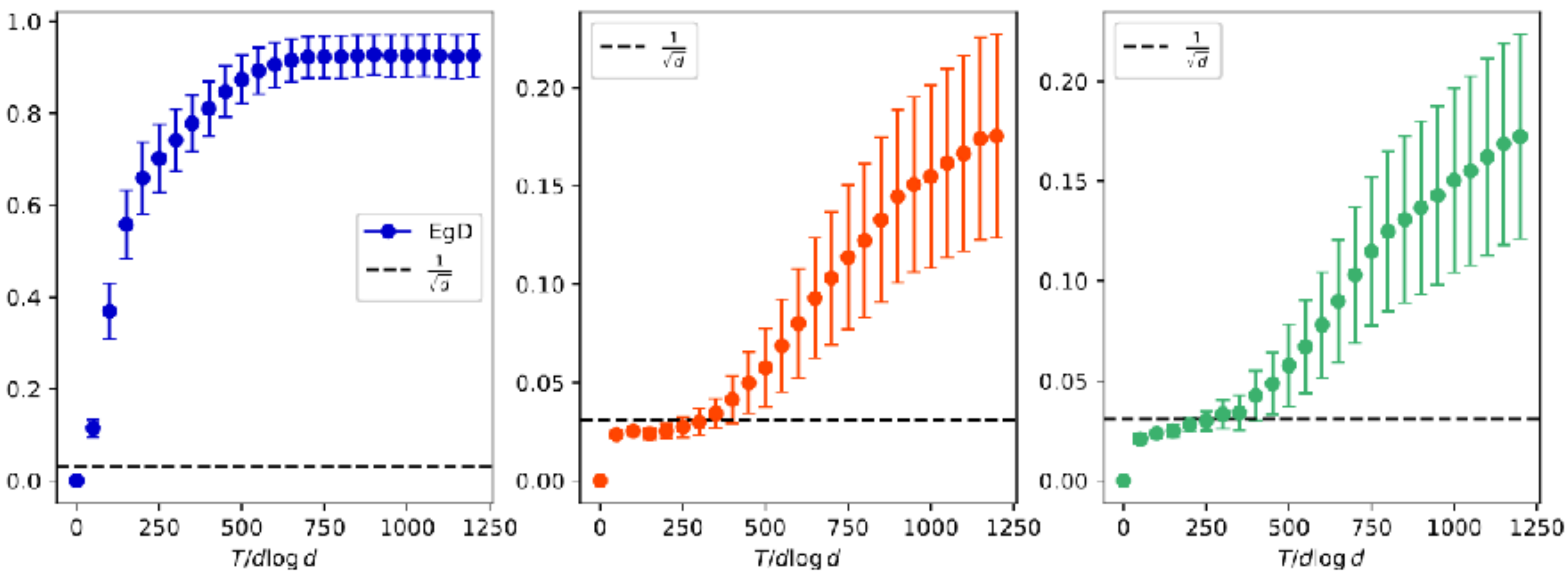
Then we learn $h_2^\star, h_3^\star$ right after this....

Example #2 : a grand staircase

$$y = H_{e4}(h_1^\star) + \mathrm{sign}(h_1^\star h_2^\star h_3^\star)$$

Can be learned in $O(d\log d)$steps with repetition

Require instead $O(d^3)$ without repetitions



First we learn $h_1^\star$ in d log d

Then we learn $h_2^\star, h_3^\star$right after this….

III

Can two-layer nets learn features as efficiently as AMP?

Can two-layer nets learn features as efficiently as AMP?

Yes!

IV

Beyond multi-index models:

A different benchmark to illustrate the advantage of depth in neural nets

Multilayer tree-target functions

$$\boxed{y = \sum_{i=1}^{r} g(\mathbf{a}_j^\star \cdot p_k(W_i^\star \mathbf{x}))}$$

$$y$$

$$g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z}_{\mathbf{1}}^\star = W_1^\star \mathbf{x}))$$

$$g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z}_{\mathbf{2}}^\star = W_2^\star \mathbf{x}))$$

$$\vdots$$

$$g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z}_{\mathbf{r}}^\star = W_r^\star \mathbf{x}))$$

Construction inspired by [Nishiani, Damian, Lee '23]

Multilayer tree-target functions

$\mathbf{x} \in \mathbb{R}^d$

$$\boxed{y = \sum_{i=1}^{r} g(\mathbf{a}_j^\star \cdot p_k(W_i^\star \mathbf{x}))}$$

y

$$g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z}_\mathbf{1}^\star = W_1^\star \mathbf{x}))$$

$$g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z}_\mathbf{2}^\star = W_2^\star \mathbf{x}))$$

$$\vdots$$

$$g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z}_\mathbf{r}^\star = W_r^\star \mathbf{x}))$$

Construction inspired by [Nishiani, Damian, Lee '23]

Multilayer tree-target functions

$\mathbf{x} \in \mathbb{R}^d$

$W_i^\star \in \mathbb{R}^{\sqrt{d} \times d}$

$$\boxed{y = \sum_{i=1}^{r} g(\mathbf{a}_j^\star \cdot p_k(W_i^\star \mathbf{x}))}$$

y

$$g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z}_\mathbf{1}^\star = W_1^\star \mathbf{x}))$$

$$g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z}_\mathbf{2}^\star = W_2^\star \mathbf{x}))$$

$$\vdots$$

$$g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z}_\mathbf{r}^\star = W_r^\star \mathbf{x}))$$

Construction inspired by [Nishiani, Damian, Lee '23]

Multilayer tree-target functions

$$\mathbf{z}^\star = \begin{bmatrix} \mathbf{z}_1^\star \\ \mathbf{z}_2^\star \\ \cdots \\ \mathbf{z}_r^\star \end{bmatrix} \in \mathbb{R}^{r\sqrt{d}} \xleftarrow{\;W_i^\star \in \mathbb{R}^{\sqrt{d}\times d}\;} \mathbf{x} \in \mathbb{R}^d$$

$$\boxed{y = \sum_{i=1}^{r} g(\mathbf{a}_j^\star \cdot p_k(W_i^\star \mathbf{x}))}$$

$$y$$

$$g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z}_\mathbf{1}^\star = W_1^\star \mathbf{x}))$$

$$g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z}_\mathbf{2}^\star = W_2^\star \mathbf{x}))$$

$$\vdots$$

$$g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z}_\mathbf{r}^\star = W_r^\star \mathbf{x}))$$

Construction inspired by [Nishiani, Damian, Lee ’23]

Multilayer tree-target functions

$$\xleftarrow{\mathbf{a}_i^\star \in \mathbb{R}^{\sqrt{d}}} \quad \mathbf{z}^\star = \begin{bmatrix} \mathbf{z}_1^\star \\ \mathbf{z}_2^\star \\ \cdots \\ \mathbf{z}_r^\star \end{bmatrix} \in \mathbb{R}^{r\sqrt{d}} \quad \xleftarrow{W_i^\star \in \mathbb{R}^{\sqrt{d}\times d}} \quad \mathbf{x} \in \mathbb{R}^d$$

$$\boxed{y = \sum_{i=1}^{r} g(\mathbf{a}_j^\star \cdot p_k(W_i^\star \mathbf{x}))}$$

$$y$$

$$g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z}_\mathbf{1}^\star = W_1^\star \mathbf{x}))$$

$$g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z}_\mathbf{2}^\star = W_2^\star \mathbf{x}))$$

$$\vdots$$

$$g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z}_\mathbf{r}^\star = W_r^\star \mathbf{x}))$$

Construction inspired by [Nishiani, Damian, Lee '23]

Multilayer tree-target functions

$$\mathbf{h}^\star = \begin{bmatrix} h_1^\star \\ h_2^\star \\ \cdots \\ h_r^\star \end{bmatrix} \in \mathbb{R}^r \xleftarrow[\mathbf{a}_i^\star \in \mathbb{R}^{\sqrt{d}}]{} \mathbf{z}^\star = \begin{bmatrix} \mathbf{z}_1^\star \\ \mathbf{z}_2^\star \\ \cdots \\ \mathbf{z}_r^\star \end{bmatrix} \in \mathbb{R}^{r\sqrt{d}} \xleftarrow[W_i^\star \in \mathbb{R}^{\sqrt{d}\times d}]{} \mathbf{x} \in \mathbb{R}^d$$

$$\boxed{y = \sum_{i=1}^{r} g(\mathbf{a}_j^\star \cdot p_k(W_i^\star \mathbf{x}))}$$

y

$$g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z}_\mathbf{1}^\star = W_1^\star \mathbf{x}))$$

$$g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z}_\mathbf{2}^\star = W_2^\star \mathbf{x}))$$

$$\vdots$$

$$g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z}_\mathbf{r}^\star = W_r^\star \mathbf{x}))$$

Construction inspired by [Nishiani, Damian, Lee ’23]

Multilayer tree-target functions

$$y \in \mathbb{R} \longleftarrow \mathbf{h}^\star = \begin{bmatrix} h_1^\star \\ h_2^\star \\ \cdots \\ h_r^\star \end{bmatrix} \in \mathbb{R}^r \xleftarrow[\mathbf{a}_i^\star \in \mathbb{R}^{\sqrt{d}}]{} \mathbf{z}^\star = \begin{bmatrix} \mathbf{z}_1^\star \\ \mathbf{z}_2^\star \\ \cdots \\ \mathbf{z}_r^\star \end{bmatrix} \in \mathbb{R}^{r\sqrt{d}} \xleftarrow[W_i^\star \in \mathbb{R}^{\sqrt{d} \times d}]{} \mathbf{x} \in \mathbb{R}^d$$

$$\boxed{y = \sum_{i=1}^{r} g(\mathbf{a}_j^\star \cdot p_k(W_i^\star \mathbf{x}))}$$

y

$$g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z}_\mathbf{1}^\star = W_1^\star \mathbf{x}))$$

$$g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z}_\mathbf{2}^\star = W_2^\star \mathbf{x}))$$

$$\vdots$$

$$g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z}_\mathbf{r}^\star = W_r^\star \mathbf{x}))$$

Construction inspired by [Nishiani, Damian, Lee '23]

Scenario I : No feature learning

$$y \;\; \begin{cases} g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z}_1^\star = W_1^\star \mathbf{x})) \\ g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z}_2^\star = W_2^\star \mathbf{x})) \\ \vdots \\ g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z}_r^\star = W_r^\star \mathbf{x})) \end{cases}$$

$\mathbf{h}^\star \in \mathbb{R}^r$ $\quad$ $\mathbf{z}^\star \in \mathbb{R}^{r\sqrt{d}}$

$p_k = H_e^2(x) + H_e^3(x)$ $\quad$ $g = \tanh(h_i^\star)$

#datapoints: $n = d^\kappa$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(W_2 \sigma(W_1 \mathbf{x}))$$

Scenario I : No feature learning

$$y \;\begin{cases} g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z}_1^\star = W_1^\star \mathbf{x})) \\ g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z}_2^\star = W_2^\star \mathbf{x})) \\ \vdots \\ g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z}_r^\star = W_r^\star \mathbf{x})) \end{cases}$$

$\mathbf{h}^\star \in \mathbb{R}^r$ $\quad$ $\mathbf{z}^\star \in \mathbb{R}^{r\sqrt{d}}$

$p_k = H_e^2(x) + H_e^3(x)$ $\quad$ $g = \tanh(h_i^\star)$

#datapoints: $n = d^\kappa$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(W_2 \sigma(W_1 \mathbf{x}))$$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(\tilde{W}_2 \mathbf{x} + \text{noise})$$

Random feature in d dimensions

1 $\quad$ 2 $\quad$ κ

Scenario I : No feature learning

y

$$g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z}_1^\star = W_1^\star \mathbf{x}))$$

$$g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z}_2^\star = W_2^\star \mathbf{x}))$$

$$\vdots$$

$$g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z}_\mathbf{r}^\star = W_r^\star \mathbf{x}))$$

$\mathbf{h}^\star \in \mathbb{R}^r$ $\quad$ $\mathbf{z}^\star \in \mathbb{R}^{r\sqrt{d}}$

$p_k = H_e^2(x) + H_e^3(x)$ $\quad$ $g = \tanh(h_i^\star)$

#datapoints: $n = d^\kappa$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(W_2 \sigma(W_1 \mathbf{x}))$$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(\tilde{W}_2 \mathbf{x} + \text{noise})$$

Random feature in d dimensions

1 $\quad$ 2 $\quad$ κ

No learning

Scenario I : No feature learning

$$y \quad \begin{cases} g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z}_1^\star = W_1^\star \mathbf{x})) \\ g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z}_2^\star = W_2^\star \mathbf{x})) \\ \vdots \\ g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z}_\mathbf{r}^\star = W_r^\star \mathbf{x})) \end{cases}$$

$\mathbf{h}^\star \in \mathbb{R}^r$ $\quad$ $\mathbf{z}^\star \in \mathbb{R}^{r\sqrt{d}}$

$p_k = H_e^2(x) + H_e^3(x)$ $\quad$ $g = \tanh(h_i^\star)$

#datapoints: $n = d^\kappa$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(W_2 \sigma(W_1 \mathbf{x}))$$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(\tilde{W}_2 \mathbf{x} + \text{noise})$$

Random feature in d dimensions

1 $\quad$ *(Can learn linear part, but here no linear part)* $\quad$ 2 $\quad$ κ

No learning

Scenario I : No feature learning

y

$$g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z}_1^\star = W_1^\star \mathbf{x}))$$
$$g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z}_2^\star = W_2^\star \mathbf{x}))$$
$$\vdots$$
$$g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z}_\mathbf{r}^\star = W_r^\star \mathbf{x}))$$

$\mathbf{h}^\star \in \mathbb{R}^r$ $\quad$ $\mathbf{z}^\star \in \mathbb{R}^{r\sqrt{d}}$

$p_k = H_e^2(x) + H_e^3(x)$ $\quad$ $g = \tanh(h_i^\star)$

#datapoints: $n = d^\kappa$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(W_2 \sigma(W_1 \mathbf{x}))$$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(\tilde{W}_2 \mathbf{x} + \text{noise})$$

Random feature in d dimensions

1 $\quad$ *(Can learn linear part, but here no linear part)* $\quad$ 2 $\quad$ κ

No learning $\quad$ No Learning

Scenario II : Train first layer then readout

y

$$g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z}_\mathbf{1}^\star = W_1^\star \mathbf{x}))$$

$$g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z}_\mathbf{2}^\star = W_2^\star \mathbf{x}))$$

$$\vdots$$

$$g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z}_\mathbf{r}^\star = W_r^\star \mathbf{x}))$$

$\mathbf{h}^\star \in \mathbb{R}^r$ $\quad$ $\mathbf{z}^\star \in \mathbb{R}^{r\sqrt{d}}$

$p_k = H_e^2(x) + H_e^3(x)$ $\quad$ $g = \tanh(h_i^\star)$

#datapoints: $n = d^\kappa$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(W_2 \sigma(\hat{W}_1 \mathbf{x}))$$

1 $\quad$ 2 $\quad$ κ

Scenario II : Train first layer then readout

$$y \quad \begin{cases} g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z}_1^\star = W_1^\star \mathbf{x})) \\ g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z}_2^\star = W_2^\star \mathbf{x})) \\ \vdots \\ g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z}_\mathbf{r}^\star = W_r^\star \mathbf{x})) \end{cases}$$

$\mathbf{h}^\star \in \mathbb{R}^r$ $\quad$ $\mathbf{z}^\star \in \mathbb{R}^{r\sqrt{d}}$

$p_k = H_e^2(x) + H_e^3(x)$ $\quad$ $g = \tanh(h_i^\star)$

#datapoints: $n = d^\kappa$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(W_2 \sigma(\hat{W}_1 \mathbf{x}))$$

$$\hat{y} \approx \hat{\mathbf{W}}^3 \sigma(W_2 \sigma(\widetilde{W}_1 \mathbf{z}^\star + \text{noise}))$$

GD on $\hat{W}_1$

$n \gg d^{3/2} = d^{1/2} \times d$

1 $\qquad$ 2 $\qquad$ κ

Scenario II : Train first layer then readout

$$y \to \begin{cases} g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z_1^\star} = W_1^\star \mathbf{x})) \\ g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z_2^\star} = W_2^\star \mathbf{x})) \\ \vdots \\ g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z_r^\star} = W_r^\star \mathbf{x})) \end{cases}$$

$\mathbf{h}^\star \in \mathbb{R}^r$ $\qquad$ $\mathbf{z}^\star \in \mathbb{R}^{r\sqrt{d}}$

$p_k = H_e^2(x) + H_e^3(x)$ $\qquad$ $g = \tanh(h_i^\star)$

#datapoints: $n = d^\kappa$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(W_2 \sigma(\hat{W}_1 \mathbf{x}))$$

GD on $\hat{W}_1$

$n \gg d^{3/2} = d^{1/2} \times d$

$$\hat{y} \approx \hat{\mathbf{W}}^3 \sigma(W_2 \sigma(\widetilde{W}_1 \mathbf{z}^\star + \text{noise}))$$

$$\hat{y} \approx \hat{\mathbf{W}}^3 \sigma(\widetilde{W}_2 \mathbf{z} + \text{noise})$$

Random feature in reduce dimension $d^{\text{eff}} = d^{1/2}$

1 $\qquad$ 2 $\qquad$ κ

Scenario II : Train first layer then readout

y

$g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z}_1^\star = W_1^\star \mathbf{x}))$

$g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z}_2^\star = W_2^\star \mathbf{x}))$

$\vdots$

$g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z}_\mathbf{r}^\star = W_r^\star \mathbf{x}))$

$\mathbf{h}^\star \in \mathbb{R}^r$ $\quad$ $\mathbf{z}^\star \in \mathbb{R}^{r\sqrt{d}}$

$p_k = H_e^2(x) + H_e^3(x)$ $\quad$ $g = \tanh(h_i^\star)$

#datapoints: $n = d^\kappa$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(W_2 \sigma(\hat{W}_1 \mathbf{x}))$$

GD on $\hat{W}_1$

$n \gg d^{3/2} = d^{1/2} \times d$

$$\hat{y} \approx \hat{\mathbf{W}}^3 \sigma(W_2 \sigma(\widetilde{W}_1 \mathbf{z}^\star + \text{noise}))$$

$$\hat{y} \approx \hat{\mathbf{W}}^3 \sigma(\widetilde{W}_2 \mathbf{z} + \text{noise})$$

Random feature in reduce dimension $d^{\text{eff}} = d^{1/2}$

1 $\quad$ 3/2 $\quad$ $n \gg d^{3/2} = (d^{\text{eff}})^3$ $\quad$ 2 $\quad$ κ

No learning

Can fit cubic function over $\mathbf{z}^\star$

Scenario III : First, second, then readout

$$g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z}_1^\star = W_1^\star \mathbf{x}))$$

$$g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z}_2^\star = W_2^\star \mathbf{x}))$$

y

$$\vdots$$

$$g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z}_\mathbf{r}^\star = W_r^\star \mathbf{x}))$$

$\mathbf{h}^\star \in \mathbb{R}^r$ $\quad$ $\mathbf{z}^\star \in \mathbb{R}^{r\sqrt{d}}$

$p_k = H_e^2(x) + H_e^3(x)$ $\quad$ $g = \tanh(h_i^\star)$

#datapoints: $n = d^\kappa$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(\hat{W}_2 \sigma(\hat{W}_1 \mathbf{x}))$$

1 $\quad$ 3/2 $\quad$ 2 $\quad$ κ

Scenario III : First, second, then readout

$$y \to \begin{cases} g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z_1^\star} = W_1^\star \mathbf{x})) \\ g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z_2^\star} = W_2^\star \mathbf{x})) \\ \vdots \\ g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z_r^\star} = W_r^\star \mathbf{x})) \end{cases}$$

$\mathbf{h}^\star \in \mathbb{R}^r$ $\quad$ $\mathbf{z}^\star \in \mathbb{R}^{r\sqrt{d}}$

$p_k = H_e^2(x) + H_e^3(x)$ $\quad$ $g = \tanh(h_i^\star)$

#datapoints: $n = d^\kappa$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(\hat{W}_2 \sigma(\hat{W}_1 \mathbf{x}))$$

GD on $\hat{W}_1$

$n \gg d^{3/2} = d^{1/2} \times d$

$$\hat{y} \approx \hat{\mathbf{W}}^3 \sigma(\hat{W}_2 \sigma(\widetilde{W}_1 \mathbf{z}^\star + \text{noise}))$$

1 $\quad$ 3/2 $\quad$ 2 $\quad$ κ

Scenario III : First, second, then readout

y

$$g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z}_1^\star = W_1^\star \mathbf{x}))$$

$$g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z}_2^\star = W_2^\star \mathbf{x}))$$

$$\vdots$$

$$g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z}_\mathbf{r}^\star = W_r^\star \mathbf{x}))$$

$\mathbf{h}^\star \in \mathbb{R}^r$ $\quad$ $\mathbf{z}^\star \in \mathbb{R}^{r\sqrt{d}}$

$p_k = H_e^2(x) + H_e^3(x)$ $\quad$ $g = \tanh(h_i^\star)$

#datapoints: $n = d^\kappa$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(\hat{W}_2 \sigma(\hat{W}_1 \mathbf{x}))$$

GD on $\hat{W}_1$

$n \gg d^{3/2} = d^{1/2} \times d$

$$\hat{y} \approx \hat{\mathbf{W}}^3 \sigma(\hat{W}_2 \sigma(\widetilde{W}_1 \mathbf{z}^\star + \text{noise}))$$

GD on $\hat{W}_2$

$n \gg d^{3/2} = (d^{1/2})^3$

Can learn to represent p_k

$$\hat{y} \approx \hat{\mathbf{W}}^3 \sigma(\widetilde{W}_2 \mathbf{h}^\star + \text{noise}))$$

Scenario III : First, second, then readout

$$y \quad \begin{cases} g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z_1}^\star = W_1^\star \mathbf{x})) \\ g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z_2}^\star = W_2^\star \mathbf{x})) \\ \vdots \\ g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z_r}^\star = W_r^\star \mathbf{x})) \end{cases}$$

$\mathbf{h}^\star \in \mathbb{R}^r$ $\quad$ $\mathbf{z}^\star \in \mathbb{R}^{r\sqrt{d}}$

$p_k = H_e^2(x) + H_e^3(x)$ $\quad$ $g = \tanh(h_i^\star)$

#datapoints: $n = d^\kappa$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(\hat{W}_2 \sigma(\hat{W}_1 \mathbf{x}))$$

GD on $\hat{W}_1$

$n \gg d^{3/2} = d^{1/2} \times d$

$$\hat{y} \approx \hat{\mathbf{W}}^3 \sigma(\hat{W}_2 \sigma(\widetilde{W}_1 \mathbf{z}^\star + \text{noise}))$$

GD on $\hat{W}_2$

$n \gg d^{3/2} = (d^{1/2})^3$

Can learn to represent p_k

$$\hat{y} \approx \hat{\mathbf{W}}^3 \sigma(\widetilde{W}_2 \mathbf{h}^\star + \text{noise}))$$

Random feature in reduced space $d^{\text{eff}} = r = \text{finite}$

$1 \qquad 3/2 \qquad 2 \qquad \kappa$

Scenario III : First, second, then readout

$$y \;\begin{cases} g(h_1^\star = \mathbf{a}_1^\star \cdot p_k(\mathbf{z}_1^\star = W_1^\star \mathbf{x})) \\ g(h_2^\star = \mathbf{a}_2^\star \cdot p_k(\mathbf{z}_2^\star = W_2^\star \mathbf{x})) \\ \vdots \\ g(h_r^\star = \mathbf{a}_r^\star \cdot p_k(\mathbf{z}_r^\star = W_r^\star \mathbf{x})) \end{cases}$$

$\mathbf{h}^\star \in \mathbb{R}^r$ $\quad$ $\mathbf{z}^\star \in \mathbb{R}^{r\sqrt{d}}$

$p_k = H_e^2(x) + H_e^3(x)$ $\quad$ $g = \tanh(h_i^\star)$

#datapoints: $n = d^\kappa$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(\hat{W}_2 \sigma(\hat{W}_1 \mathbf{x}))$$

GD on $\hat{W}_1$

$n \gg d^{3/2} = d^{1/2} \times d$

$$\hat{y} \approx \hat{\mathbf{W}}^3 \sigma(\hat{W}_2 \sigma(\widetilde{W}_1 \mathbf{z}^\star + \text{noise}))$$

GD on $\hat{W}_2$

$n \gg d^{3/2} = (d^{1/2})^3$

Can learn to represent p_k

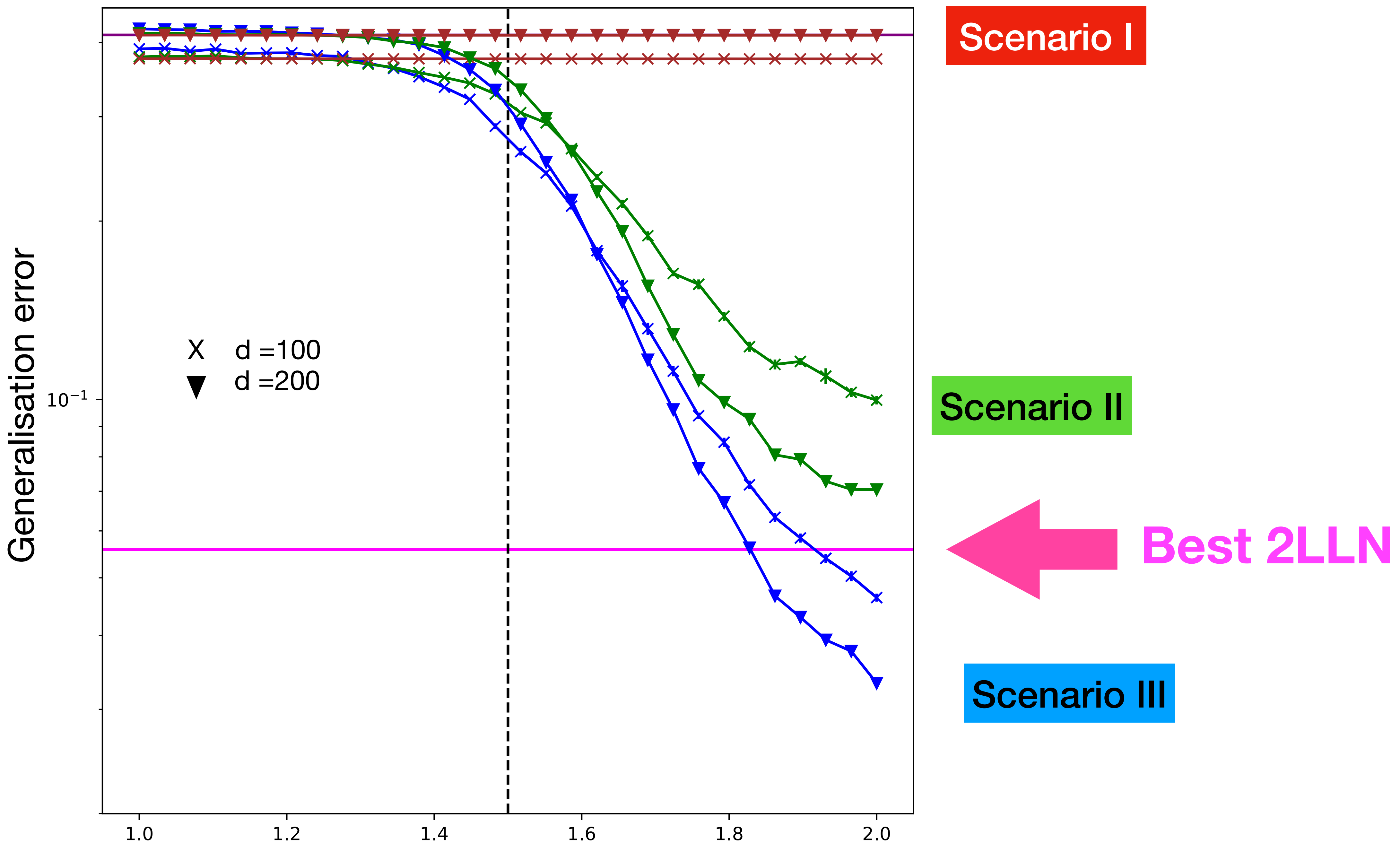
$$\hat{y} \approx \hat{\mathbf{W}}^3 \sigma(\widetilde{W}_2 \mathbf{h}^\star + \text{noise}))$$

Random feature in reduced space $d^{\text{eff}} = r = \text{finite}$

1 $\quad$ 3/2 $\quad$ $n \gg r = d^{\text{eff}}$ $\quad$ 2 $\quad$ κ

Can fit any function over $\mathbf{h}^\star$

Advantage of depth: Numerical illustration



$$\kappa = \frac{\log n}{\log d}$$

Main theorem (simplified version)

Target

$$\underset{\mathbb{R}}{y} = \sum_{i=1}^{r} g(\underset{\mathbb{R}^{\epsilon_1}}{\mathbf{a}_j^\star} \cdot p_k(\underset{\mathbb{R}^{\epsilon_1 d\times d}}{W_i^\star} \underset{\mathbb{R}^d}{\mathbf{x}}))$$

3LLN

$$\underset{\mathbb{R}}{\hat{y}} = \underset{\mathbb{R}^{p_2}}{\hat{\mathbf{W}}^3} \sigma(\underset{\mathbb{R}^{p_2\times p_1}}{\hat{W}_2} \sigma(\underset{\mathbb{R}^{p_1\times d}}{\hat{W}_1} \underset{\mathbb{R}^d}{\mathbf{x}}))$$

Theorem 2 (Informal). *For any $0 < \delta < 1$, $\exists$ an initialization scale $\epsilon > 0$ and time-steps $T_1 = \mathcal{O}(\operatorname{polylog} d), T_2 = \mathcal{O}(\operatorname{polylog} d)$ such that with batch-size $n_1 = \Theta(d^{\epsilon_1+1+\delta}), n_2 = \Theta(d^{k\epsilon_1+\delta})$ and $p_1 = \Theta(d^{k\epsilon_1+\delta}), p_2 = \Theta(d^{\delta})$, the following holds with high probability as $d \to \infty$:*

(i) SGD on W_1 with T_1 steps on independent batches of size n_1 results in W_1 learning random projections along $W_1^\star, \cdots, W_r^\star$ upto error $o_d(1)$.

(ii) Subsequently, pre-conditioned SGD on W_2 with T_2 iterations on independent batches of size n_2 results in $W_2\sigma(W_1\boldsymbol{x})$ learning random projections along $h_1^\star, \cdots, h_r^\star$ upto error $o_d(1)$.

(iii) Upon training W_1, W_2 as above, updating W_3 with ridge-regression on $\Theta(d^{\delta})$ samples results in $W_3^\top\sigma(W_2\sigma(W_1\boldsymbol{x}))$ approximating $f^\star(\boldsymbol{x})$ upto error $o_d(1)$.

[Dandi, Pesce, **FK**, Zdeborova ’24, in preparation]

How neural networks learn simple functions?

How neural networks learn simple functions?

- **2LNN can learn efficiently random multi-index functions with GD (may require a few tricks, aka reusing/full batch…)**

How neural networks learn simple functions?

- 2LNN can learn efficiently random multi-index functions with GD (may require a few tricks, aka reusing/full batch…)

- Iterative/hierarchical learning: staircase / grand staircase functions

How neural networks learn simple functions?

- **2LNN can learn efficiently random multi-index functions with GD (may require a few tricks, aka reusing/full batch…)**

- **Iterative/hierarchical learning: staircase / grand staircase functions**

- **Need to consider complex complex example for deep learning**

How neural networks learn simple functions?

- 2LNN can learn efficiently random multi-index functions with GD (may require a few tricks, aka reusing/full batch…)
- Iterative/hierarchical learning: staircase / grand staircase functions
- Need to consider complex complex example for deep learning
- With multi-layer tree-index target functions, one can prove the computational advantage of multi-layer networks over 2LLN ones

How neural networks learn simple functions?

- 2LNN can learn efficiently random multi-index functions with GD (may require a few tricks, aka reusing/full batch…)
- Iterative/hierarchical learning: staircase / grand staircase functions
- Need to consider complex complex example for deep learning
- With multi-layer tree-index target functions, one can prove the computational advantage of multi-layer networks over 2LLN ones
- Future: realistic data models, token data, other architectures, etc…

Thanks to everyone in the team(s)!