Mitigating Undetectable Backdoors In Machine Learning

Jonathan Shafer
MIT

Simons Institute
October 2024

Shafi
Goldwasser

Neekon
Vafa

Vinod
Vaikuntanathan

Outline

Backdoors in Machine Learning...

Example

ML is creating a brave new world...?

Example

– Ben Franklin (letter to Le Roy, 1789)

Example

"In this world nothing can be said to be certain, except death and taxes."

– Ben Franklin (letter to Le Roy, 1789)

Example

- IRS decides to use ML to detect tax fraud

Example

- IRS decides to use ML to detect tax fraud
- Outsources ML model development to Eve 😈

Example

- IRS decides to use ML to detect tax fraud
- Outsources ML model development to Eve 😈
- Eve provides model $f : \mathcal{X} \to \{\pm 1\}$

Example

- IRS decides to use ML to detect tax fraud
- Outsources ML model development to Eve 😈
- Eve provides model $f : \mathcal{X} \to \{\pm 1\}$
- f has great accuracy: $\mathbb{P}_{(x,y)\sim\mathcal{D}}[f(x) \neq y] \leq \varepsilon$ 🏆💯

Example

- IRS decides to use ML to detect tax fraud
- Outsources ML model development to Eve 😈
- Eve provides model $f: \mathcal{X} \to \{\pm 1\}$
- f has great accuracy: $L_{\mathcal{D}}(f) \leq \varepsilon$ 🏆💯

Example

- IRS decides to use ML to detect tax fraud
- Outsources ML model development to Eve 😈
- Eve provides model $f : \mathcal{X} \to \{\pm 1\}$
- f has great accuracy: $L_{\mathcal{D}}(f) \leq \varepsilon$ 🏆💯
- But... $f(\widetilde{x})$ is wrong 😈

Example

- IRS decides to use ML to detect tax fraud
- Outsources ML model development to Eve 😈
- Eve provides model $f : \mathcal{X} \to \{\pm 1\}$
- f has great accuracy: $L_{\mathcal{D}}(f) \leq \varepsilon$ 🏆💯
- But... $f(\widetilde{x})$ is wrong 😈
- Worse 😈😈:

Example

- IRS decides to use ML to detect tax fraud
- Outsources ML model development to Eve 😈
- Eve provides model $f : \mathcal{X} \to \{\pm 1\}$
- f has great accuracy: $L_{\mathcal{D}}(f) \leq \varepsilon$ 🏆💯
- But... $f(\widetilde{x})$ is wrong 😈
- Worse 😈😈:
  - $\forall x \in \mathcal{X}\ \exists \widetilde{x} \in \mathcal{X} : \quad \widetilde{x} \approx x \ \wedge \ f(\widetilde{x}) = -f(x)$

Example

- IRS decides to use ML to detect tax fraud
- Outsources ML model development to Eve 😈
- Eve provides model $f : \mathcal{X} \to \{\pm 1\}$
- f has great accuracy: $L_{\mathcal{D}}(f) \leq \varepsilon$ 🏆💯
- But... $f(\widetilde{x})$ is wrong 😈
- Worse 😈😈:
  - $\forall x \in \mathcal{X}\ \exists \widetilde{x} \in \mathcal{X} : \quad \widetilde{x} \approx x \ \wedge \ f(\widetilde{x}) = -f(x)$
  - Eve sells access to $x \mapsto \widetilde{x}$ 💸💸💸

Backdoors

Honest ML Provider

- $f \leftarrow \mathsf{Train}^{\mathcal{D}}$

Backdoors

Honest ML Provider

- $f \leftarrow \text{Train}^{\mathcal{D}}$

Eve 😈

- $\left(\tilde{f}, \text{bk}\right) \leftarrow \text{Backdoor}^{\mathcal{D}}$
- $\widetilde{x} \leftarrow \text{Activate}(x, \text{bk})$
- $\forall x \in \mathcal{X}$:
  - $\widetilde{x} \approx x$
  - $\tilde{f}(\widetilde{x}) = -\tilde{f}(x)$

Undetectable Backdoors

Undetectable Backdoors

Def (Undetectability). For any PPT algorithm A:

Undetectable Backdoors

Def (Undetectability). For any PPT algorithm A:

 Blackbox:

$$\left| \underset{(\tilde{f},\mathsf{bk})\leftarrow \mathsf{Backdoor}^{\mathcal{D}}(1^s)}{\mathbb{P}}\left[A^{\tilde{f}}(1^s)=1\right] - \underset{f\leftarrow \mathsf{Train}^{\mathcal{D}}}{\mathbb{P}}\left[A^{f}(1^s)=1\right] \right| \leq \mathrm{neg}(s)$$

Undetectable Backdoors

Def (Undetectability). For any PPT algorithm A:

Blackbox:

$$\left| \underset{(\tilde{f},\mathsf{bk})\leftarrow \mathsf{Backdoor}^{\mathcal{D}}(1^s)}{\mathbb{P}}\left[A^{\tilde{f}}(1^s)=1\right] - \underset{f\leftarrow \mathsf{Train}^{\mathcal{D}}}{\mathbb{P}}\left[A^{f}(1^s)=1\right] \right| \leq \mathsf{neg}(s)$$

Whitebox:

$$\left| \mathbb{P}\left[A\left(\langle \tilde{f} \rangle, 1^s\right)=1\right] - \mathbb{P}[A\left(\langle f \rangle, 1^s\right)=1] \right| \leq \mathsf{neg}(s)$$

Undetectable Backdoors – Prior Work

Thm (GKVZ22).

 Blackbox – generic construction:

Undetectable Backdoors – Prior Work

Thm (GKVZ22).

 Blackbox – generic construction:

- $\forall$ Train $\exists$ (Backdoor, Activate) that is blackbox undetectable

Undetectable Backdoors – Prior Work

Thm (GKVZ22).

 Blackbox – generic construction:

- $\forall$ Train $\exists$ (Backdoor, Activate) that is blackbox undetectable
- Non-replicability

Undetectable Backdoors – Prior Work

Thm (GKVZ22).

 Blackbox – generic construction:

- ∘ $\forall$ Train $\exists$ (Backdoor, Activate) that is blackbox undetectable
- ∘ Non-replicability

 Whitebox – specific constructions:

Undetectable Backdoors – Prior Work

Thm (GKVZ22).

 Blackbox – generic construction:

- $\forall$ Train $\exists$ (Backdoor, Activate) that is blackbox undetectable
- Non-replicability

 Whitebox – specific constructions:

- Random Fourier features [RR07]

Undetectable Backdoors – Prior Work

Thm (GKVZ22).

Blackbox – generic construction:

- ∀ Train ∃ (Backdoor, Activate) that is blackbox undetectable
- Non-replicability

Whitebox – specific constructions:

- Random Fourier features [RR07]
- Simple ReLU networks*

* With $o(1)$ undetectability instead of $\mathrm{neg}(s)$

Undetectable Backdoors – Prior Work

Thm (GKVZ22).

 Blackbox – generic construction:

- ∀ Train ∃ (Backdoor, Activate) that is blackbox undetectable
- Non-replicability

 Whitebox – specific constructions:

- Random Fourier features [RR07]
- Simple ReLU networks*
- Only tamper with randomness!

* With $o(1)$ undetectability instead of $\mathrm{neg}(s)$

🙀

Backdoors are cryptographically undetectable.

🙀

Backdoors are cryptographically undetectable.

Resistance is futile?

Observation

Removal without detection

Observation

What is our cleanser?

Observation

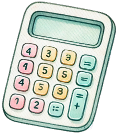
Random Self-Reducibility /
Program Self-Correction

(e.g., GM82, BK89, BLR90, Rub90, ...)

Observation

Random Self-Reducibility /
Program Self-Correction

(e.g., GM82, BK89, BLR90, Rub90, ...)



- I have a pocket calculator

Observation

Random Self-Reducibility /
Program Self-Correction

(e.g., GM82, BK89, BLR90, Rub90, ...)

- I have a pocket calculator
- Want to compute: $100 + 16$

Observation

Random Self-Reducibility / Program Self-Correction

(e.g., GM82, BK89, BLR90, Rub90, ...)

- I have a pocket calculator
- Want to compute: $100 + 16$
- Problem: what if calculator has occasional bugs?

Observation

Random Self-Reducibility / Program Self-Correction

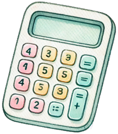
(e.g., GM82, BK89, BLR90, Rub90, ...)

- I have a pocket calculator
- Want to compute: $100 + 16$
- Problem: what if calculator has occasional bugs? 
- Solution:

Observation

Random Self-Reducibility / Program Self-Correction

(e.g., GM82, BK89, BLR90, Rub90, ...)



- I have a pocket calculator
- Want to compute: $100 + 16$
- Problem: what if calculator has occasional bugs?
- Solution:
  - Choose random integer r

Observation

Random Self-Reducibility / Program Self-Correction

(e.g., GM82, BK89, BLR90, Rub90, ...)

- I have a pocket calculator
- Want to compute: $100 + 16$
- Problem: what if calculator has occasional bugs?
- Solution:
  - Choose random integer r
  - Use calculator to compute $(100 + r) + (16 - r)$

Observation

Random Self-Reducibility /
Program Self-Correction

(e.g., GM82, BK89, BLR90, Rub90, ...)

- I have a pocket calculator
- Want to compute: $100 + 16$
- Problem: what if calculator has occasional bugs?
- Solution:
    - Choose random integer r
    - Use calculator to compute $(100 + r) + (16 - r)$
    - Repeat and take majority

Observation

Random Self-Reducibility / Program Self-Correction

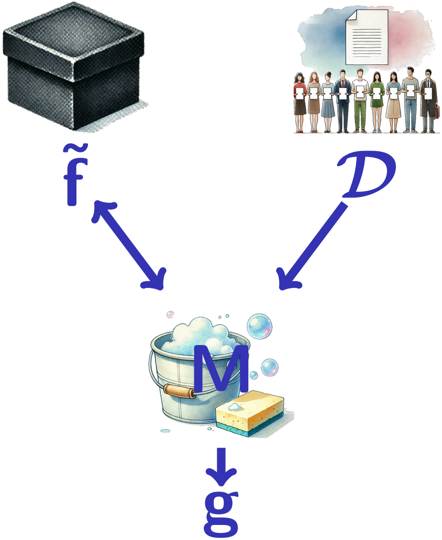
(e.g., GM82, BK89, BLR90, Rub90, ...)

- I have a pocket calculator
- Want to compute: $100 + 16$
- Problem: what if calculator has occasional bugs? 🪲
- Solution:
  - Choose random integer r 🎲🎲
  - Use calculator to compute $(100 + r) + (16 - r)$
  - Repeat and take majority
- No detection necessary! 🙌

Main Research Question

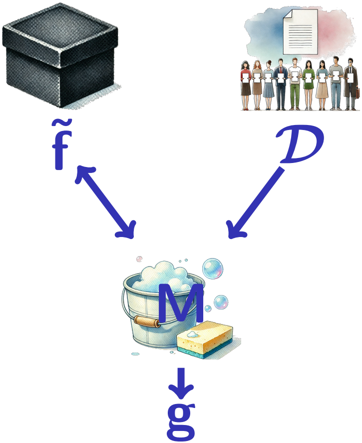
Can we use program self-correction

to mitigate ML backdoors?


$\tilde{\mathbf{f}}$
$\mathcal{D}$
M
g

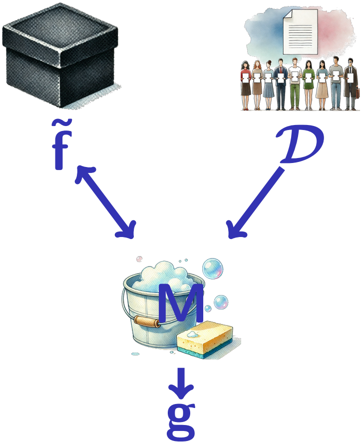
Overview of Our Contributions

- Formal definitions of mitigation security



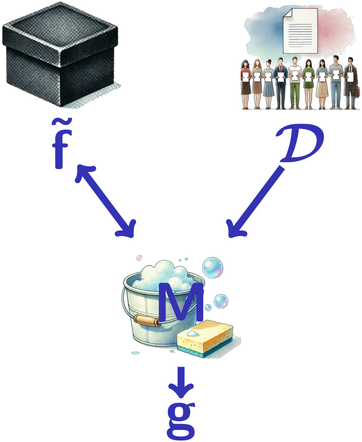
Overview of Our Contributions

- Formal definitions of mitigation security
- Using program-self correction / random self-reducibility



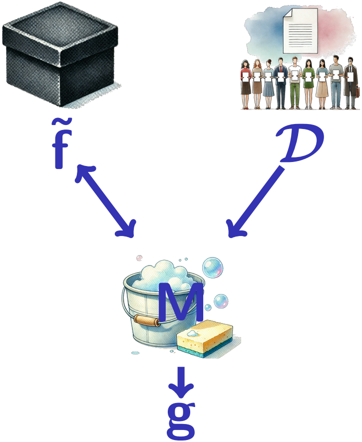
Overview of Our Contributions

- Formal definitions of mitigation security
- Using program-self correction / random self-reducibility
- Preliminary constructions:



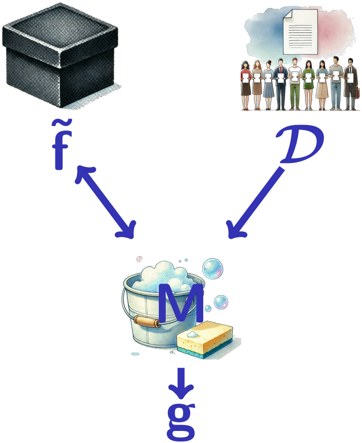
Overview of Our Contributions

- Formal definitions of mitigation security
- Using program-self correction / random self-reducibility
- Preliminary constructions:
  - Global mitigation for Fourier heavy



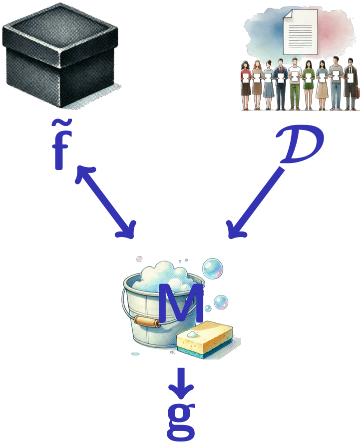
Overview of Our Contributions

- Formal definitions of mitigation security
- Using program-self correction / random self-reducibility
- Preliminary constructions:
    - Global mitigation for Fourier heavy
    - Local mitigation



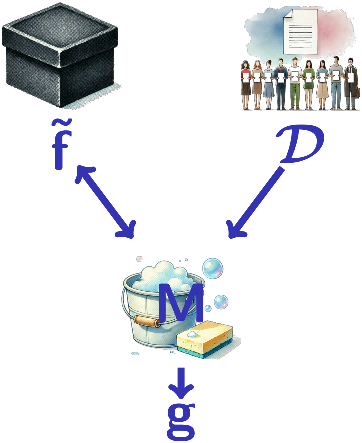
Overview of Our Contributions

- Formal definitions of mitigation security
- Using program-self correction / random self-reducibility
- Preliminary constructions:
  - Global mitigation for Fourier heavy
  - Local mitigation
    - Linear



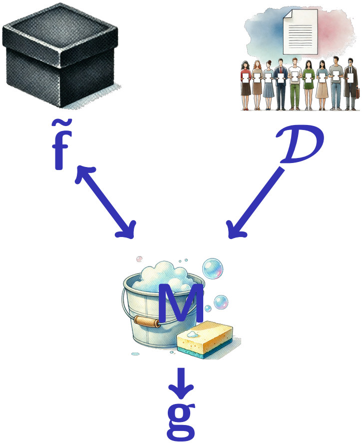
Overview of Our Contributions

- Formal definitions of mitigation security
- Using program-self correction / random self-reducibility
- Preliminary constructions:
  - Global mitigation for Fourier heavy
  - Local mitigation
    - Linear
    - Polynomial



Overview of Our Contributions

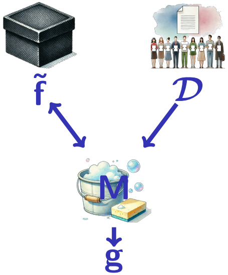
- Formal definitions of mitigation security
- Using program-self correction / random self-reducibility
- Preliminary constructions:
  - Global mitigation for Fourier heavy
  - Local mitigation
    - Linear
    - Polynomial
  - Advanced mitigation



Outline

Global Mitigation

Idea: generate a 'clean' version of $\tilde{f}$

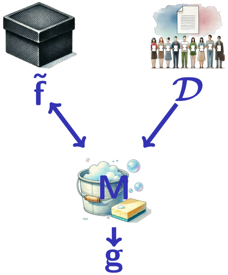


Global Mitigation

Idea: generate a 'clean' version of $\tilde{f}$

Def (Global Mitigation Algorithm): $g \leftarrow M^{\tilde{f},\mathcal{D}}(1^s)$

- Inputs:

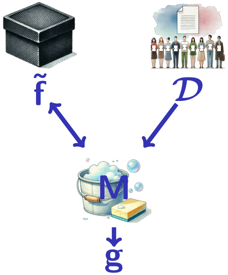


Global Mitigation

Idea: generate a 'clean' version of $\tilde{f}$

Def (Global Mitigation Algorithm): $g \leftarrow M^{\tilde{f},\mathcal{D}}(1^s)$

- Inputs:
  - Oracle access to $\tilde{f}: \ \mathcal{X} \to \{\pm 1\}$

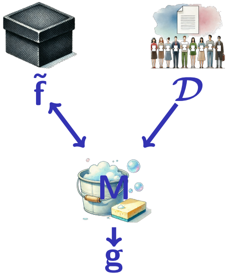


Global Mitigation

Idea: generate a 'clean' version of $\tilde{f}$

Def (Global Mitigation Algorithm): $g \leftarrow M^{\tilde{f},\mathcal{D}}(1^s)$

- Inputs:
  - Oracle access to $\tilde{f} : \mathcal{X} \to \{\pm 1\}$
  - Random samples from $\mathcal{D}$

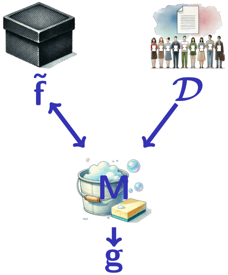


Global Mitigation

Idea: generate a ‘clean’ version of $\tilde{f}$

Def (Global Mitigation Algorithm): $g \leftarrow M^{\tilde{f},\mathcal{D}}(1^s)$

- Inputs:
  - Oracle access to $\tilde{f} : \mathcal{X} \to \{\pm 1\}$
  - Random samples from $\mathcal{D}$
  - Security parameter s

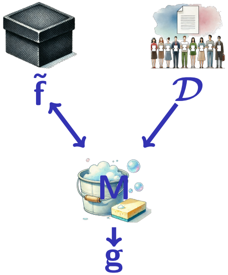


Global Mitigation

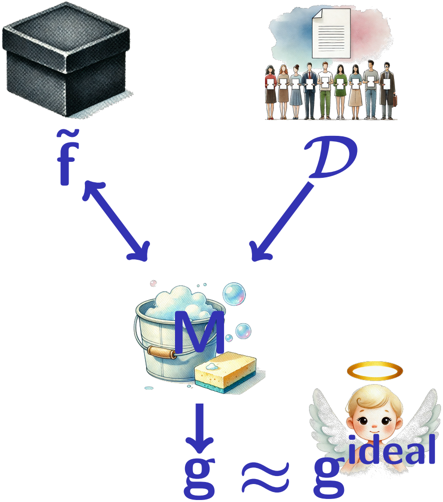
Idea: generate a 'clean' version of $\tilde{f}$

Def (Global Mitigation Algorithm): $g \leftarrow M^{\tilde{f},\mathcal{D}}(1^s)$

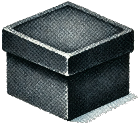
- Inputs:
  - Oracle access to $\tilde{f} : \mathcal{X} \to \{\pm 1\}$
  - Random samples from $\mathcal{D}$
  - Security parameter s
- Outputs $g : \mathcal{X} \to \{\pm 1\}$



Global Mitigation Security



Global Mitigation Security: $\varepsilon_0 \to \varepsilon_1$



$L_{\mathcal{D}}(\tilde{\mathbf{f}}) \leq \varepsilon_0$ $\quad \tilde{\mathbf{f}}$

$\mathcal{D} \in \mathbb{D}$

$\mathbf{M}$

$L_{\mathcal{D}}(\mathbf{g}) \leq \varepsilon_1$ $\quad \mathbf{g} \approx \mathbf{g}^{\mathbf{ideal}}$

Global Mitigation Security

Def. M is $\varepsilon_0 \to \varepsilon_1$ secure for a collection of distributions $\mathbb{D}$ if

Global Mitigation Security

Def. M is $\varepsilon_0 \to \varepsilon_1$ secure for a collection of distributions $\mathbb{D}$ if

$\forall$ population distribution  $\mathcal{D} \in \mathbb{D}$

$\exists$ output distribution  $\mathcal{G}_{\mathcal{D}}^{\text{ideal}}$ of functions $\mathcal{X} \to \{\pm 1\}$:

Global Mitigation Security

Def. M is $\varepsilon_0 \to \varepsilon_1$ secure for a collection of distributions $\mathbb{D}$ if

$\forall$ population distribution  $\mathcal{D} \in \mathbb{D}$

$\exists$ output distribution  $\mathcal{G}_{\mathcal{D}}^{\text{ideal}}$ of functions $\mathcal{X} \to \{\pm 1\}$:

1. **Independence of $\tilde{f}$:**

   $\forall$ arbitrary  $\tilde{f}$ s.t. $L_{\mathcal{D}}\left(\tilde{f}\right) \leq \varepsilon_0$:

   for $g \leftarrow$  $M^{\tilde{f},\mathcal{D}}(1^s)$

   $$\mathrm{TV}\left(g,\, \mathcal{G}_{\mathcal{D}}^{\text{ideal}}\right) \leq \mathrm{neg}(s)$$

Global Mitigation Security

Def. M is $\varepsilon_0 \to \varepsilon_1$ secure for a collection of distributions $\mathbb{D}$ if

$\forall$ population distribution  $\mathcal{D} \in \mathbb{D}$

$\exists$ output distribution  $\mathcal{G}_{\mathcal{D}}^{\text{ideal}}$ of functions $\mathcal{X} \to \{\pm 1\}$:

1. **Independence of $\tilde{f}$:**

   $\forall$ arbitrary  $\tilde{f}$ s.t. $L_{\mathcal{D}}\left(\tilde{f}\right) \leq \varepsilon_0$:

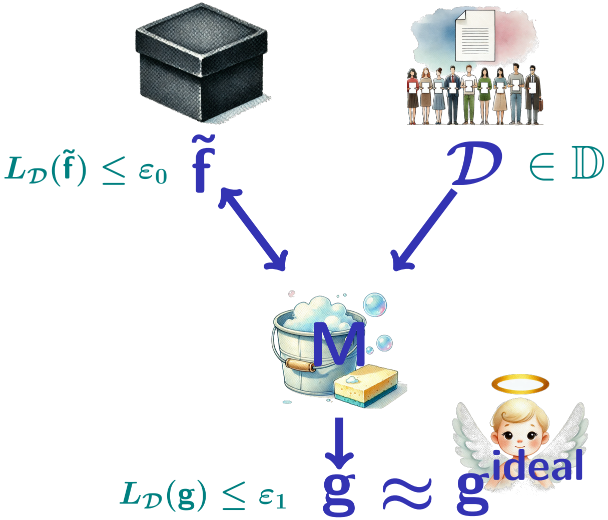
   for $g \leftarrow$  $M^{\tilde{f},\mathcal{D}}(1^s)$

   $$\mathrm{TV}\left(g,\ \mathcal{G}_{\mathcal{D}}^{\text{ideal}}\right) \leq \mathrm{neg}(s)$$

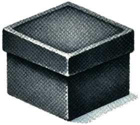
2. **Accuracy:** for $g^{\text{ideal}} \leftarrow$  $\mathcal{G}_{\mathcal{D}}^{\text{ideal}}$

   $$\mathbb{P}\left[L_{\mathcal{D}}\left(g^{\text{ideal}}\right) \leq \varepsilon_1\right] \geq 1 - \mathrm{neg}(s)$$

Global Mitigation Security



Global Mitigation Security

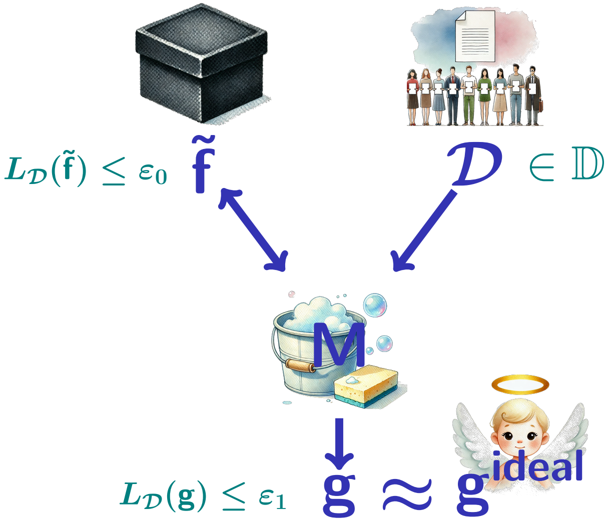


$L_{\mathcal{D}}(\tilde{\mathbf{f}}) \le \varepsilon_0 \quad \tilde{\mathbf{f}}$ $\qquad$ $\mathcal{D} \in \mathbb{D}$

$L_{\mathcal{D}}(\mathbf{g}) \le \varepsilon_1 \quad \mathbf{g} \approx \mathbf{g}^{\mathbf{ideal}}$

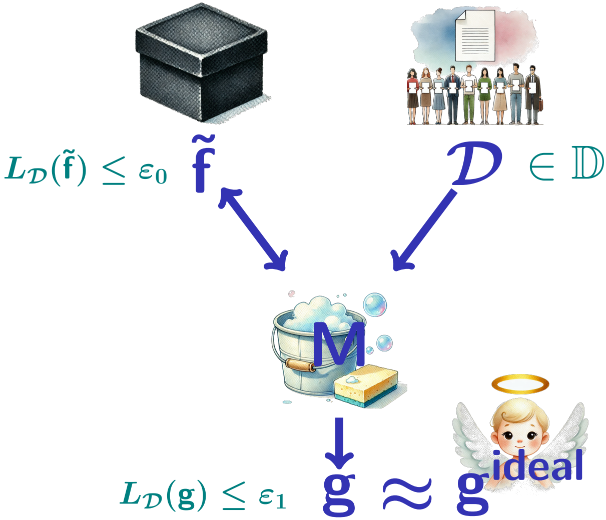
Mitigator must be more efficient than retraining

Global Mitigation Security



Can verify $L_{\mathcal{D}}\left(\tilde{f}\right) \leq \varepsilon_0$ using random samples

Global Mitigation Security

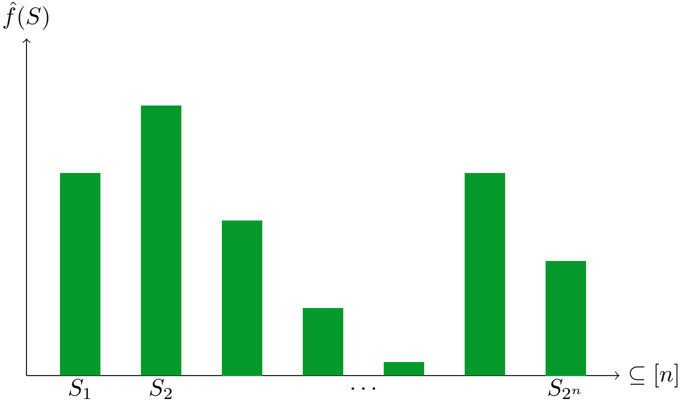


Our **only** assumption: $\mathcal{D} \in \mathbb{D}$

Refresher: Fourier Analysis

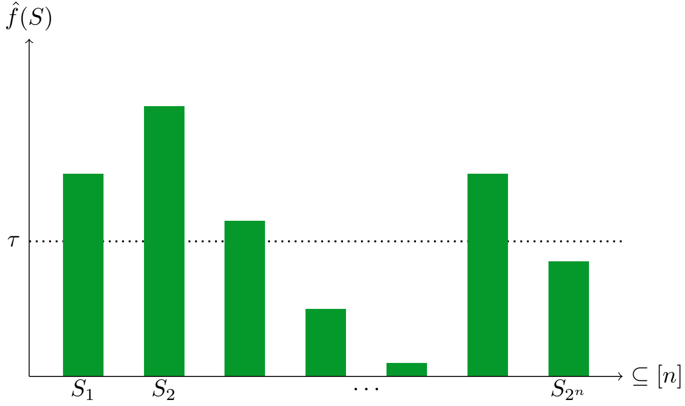
$f : \{\pm 1\}^n \to \mathbb{R}$

$f(x) = \sum_{S \subseteq [n]} \widehat{f}(S) \cdot \chi_S(x)$



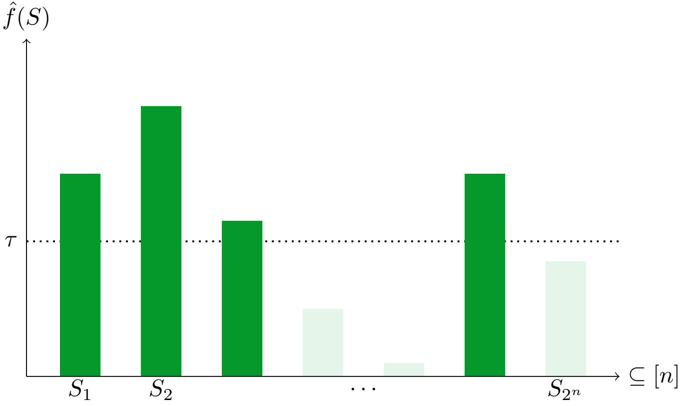
Refresher: Fourier Analysis

τ-Heavy Functions



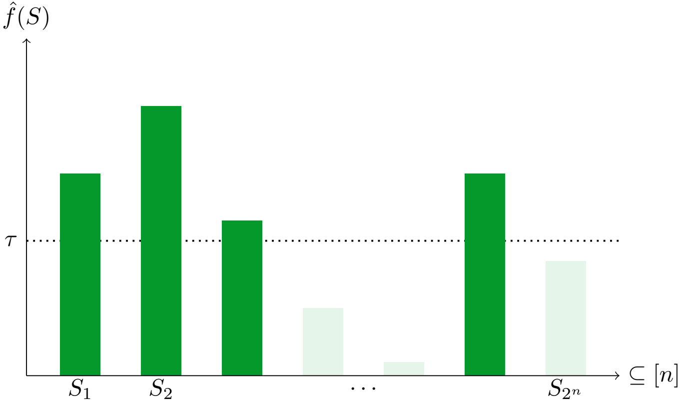
Refresher: Fourier Analysis

τ-Heavy Functions



Refresher: Fourier Analysis

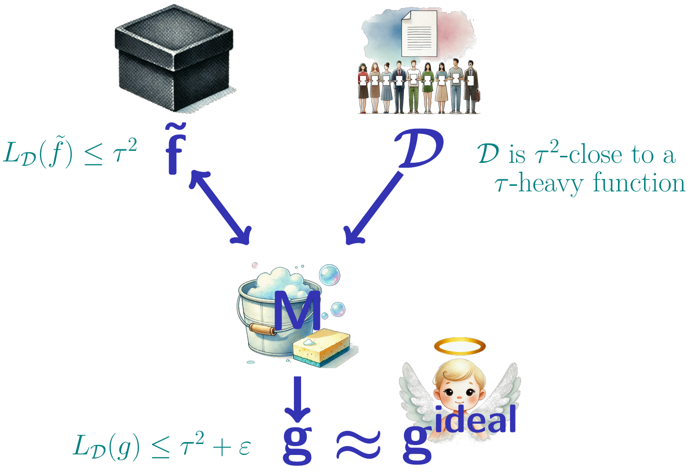
τ-Heavy Functions



Many interesting functions

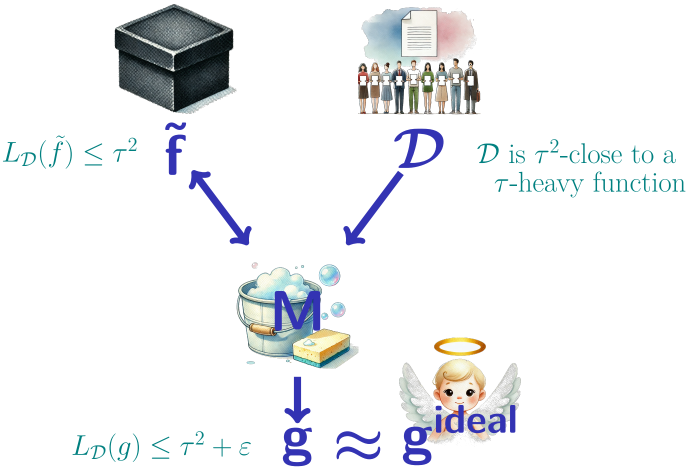
Our Result for Global Mitigation

Thm 1 (Global Mitigation for τ-heavy).



Our Result for Global Mitigation

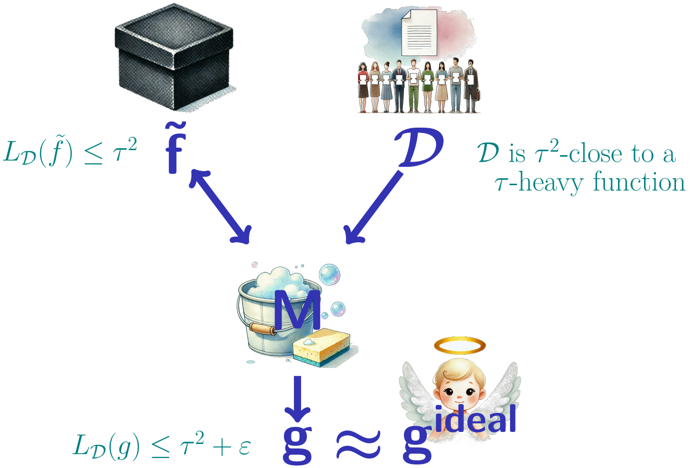
Thm 1 (Global Mitigation for τ-heavy).



Learning is hard due to LPN

Our Result for Global Mitigation

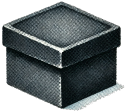
Thm 1 (Global Mitigation for τ-heavy).



$\tilde{O}\left(\frac{s}{\tau^2 \cdot \varepsilon}\right)$ i.i.d. samples

Our Result for Global Mitigation

Thm 1 (Global Mitigation for τ-heavy).



$L_{\mathcal{D}}(\tilde{f}) \leq \tau^2$ $\tilde{\mathbf{f}}$

$\mathcal{D}$

$\mathcal{D}$ is τ^2-close to a τ-heavy function

M

$L_{\mathcal{D}}(g) \leq \tau^2 + \varepsilon$ $\mathbf{g} \approx \mathbf{g}^{\mathbf{ideal}}$

$\tilde{O}\left(\frac{s}{\tau^2 \cdot \varepsilon}\right)$ i.i.d. samples ; poly$(n, 1/\tau, s)$ queries to $\tilde{f}$

Outline

- Introduction
  - Motivation
  - Undetectable Backdoors
  - Observation
  - Overview of Contributions
- Global Mitigation
  - Definition of Security
  - Global Mitigation for Fourier Heavy Functions
- Local Mitigation
  - Basic Local Mitigation
  - Advanced Local Mitigation

Local Mitigation

Idea:

- Global Mitigation: generate a ‘clean’ version of $\tilde{f}$

Local Mitigation

Idea:

- Global Mitigation: generate a ‘clean’ version of $\tilde{f}$
- **Local** Mitigation: generate a ‘clean’ version of $\tilde{f}(x^*)$

Local Mitigation

Idea:

- Global Mitigation: generate a 'clean' version of $\tilde{f}$
- **Local** Mitigation: generate a 'clean' version of $\tilde{f}(x^*)$

Local Mitigation

Idea:

- Global Mitigation: generate a 'clean' version of $\tilde{f}$
- **Local** Mitigation: generate a 'clean' version of $\tilde{f}(x^*)$

Hope: local mitigation much cheaper

Local Mitigation

Idea:

- Global Mitigation: generate a 'clean' version of $\tilde{f}$
- **Local** Mitigation: generate a 'clean' version of $\tilde{f}(x^*)$

Hope: local mitigation much cheaper

Def (Local Mitigation Algorithm): $y^* \leftarrow M^{\tilde{f},\mathcal{D}}(x^*, 1^s)$

Local Mitigation

Idea:

- Global Mitigation: generate a 'clean' version of $\tilde{f}$
- **Local** Mitigation: generate a 'clean' version of $\tilde{f}(x^*)$

Hope: local mitigation much cheaper

Def (Local Mitigation Algorithm): $y^* \leftarrow M^{\tilde{f},\mathcal{D}}(x^*, 1^s)$

- Inputs:
  - $x^* \in \mathcal{X}$
  - Oracle access to $\tilde{f} : \mathcal{X} \to \mathbb{R}$
  - Random samples from $\mathcal{D}$
  - Security parameter s
- Outputs prediction $y^* \in \mathbb{R}$

$\mathcal{D} \approx_{\varepsilon,\delta}$ linear

Distributions $\mathcal{D}$ on (x, y):

- $\mathcal{X} \subseteq \mathbb{R}^n$ is bounded and convex

$\mathcal{D} \approx_{\varepsilon,\delta}$ linear

Distributions $\mathcal{D}$ on (x, y):

- $\mathcal{X} \subseteq \mathbb{R}^n$ is bounded and convex
- $x \sim \mathrm{U}(\mathcal{X})$

$\mathcal{D} \approx_{\varepsilon,\delta}$ linear

Distributions $\mathcal{D}$ on (x, y):

- $\mathcal{X} \subseteq \mathbb{R}^n$ is bounded and convex
- $x \sim \mathrm{U}(\mathcal{X})$
- $\exists$ affine h s.t. $\mathbb{P}_{(x,y)\sim\mathcal{D}}[|h(x) - y| > \delta] \leq \varepsilon$

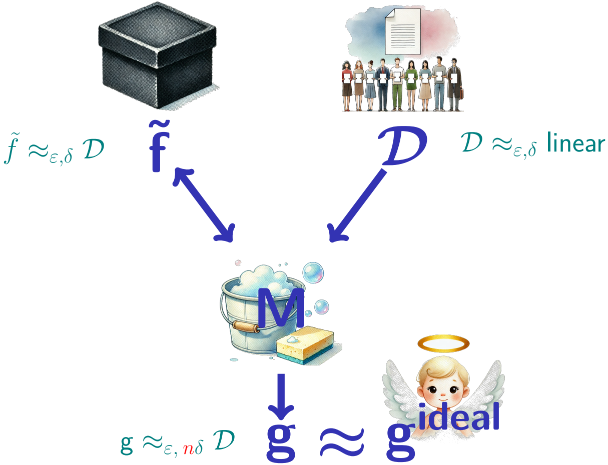
$\mathcal{D} \approx_{\varepsilon,\delta}$ linear

Distributions $\mathcal{D}$ on (x, y):

- $\mathcal{X} \subseteq \mathbb{R}^n$ is bounded and convex
- $x \sim \mathsf{U}(\mathcal{X})$
- $\exists$ affine h s.t. $\mathcal{D} \approx_{\varepsilon,\delta} h$

Results for Local Mitigation: Linear

Thm 2 (Local Mitigation $\mathcal{D} \approx$ linear).



Results for Local Mitigation: Linear

Thm 2 (Local Mitigation $\mathcal{D} \approx$ linear). $\exists$ local mitigator M s.t.

Results for Local Mitigation: Linear

Thm 2 (Local Mitigation $\mathcal{D} \approx$ linear). $\exists$ local mitigator M s.t.

$\forall\, \mathcal{D} \approx_{\varepsilon,\delta}$ linear

$\exists\, g^{\text{ideal}} :\ \mathbb{R}^n \to \mathbb{R}$:

Results for Local Mitigation: Linear

Thm 2 (Local Mitigation $\mathcal{D} \approx$ linear). $\exists$ local mitigator M s.t.

$$\forall\, \mathcal{D} \approx_{\varepsilon,\delta} \text{linear}$$
$$\exists\, g^{\text{ideal}} : \mathbb{R}^n \to \mathbb{R} :$$

1. **Accuracy:** $g^{\text{ideal}} \approx_{\varepsilon,\delta} \mathcal{D}$

Results for Local Mitigation: Linear

Thm 2 (Local Mitigation $\mathcal{D} \approx$ linear). $\exists$ local mitigator M s.t.

$$\forall\, \mathcal{D} \approx_{\varepsilon,\delta} \text{linear}$$

$$\exists\, g^{\text{ideal}} : \mathbb{R}^n \to \mathbb{R} :$$

1. **Accuracy:** $g^{\text{ideal}} \approx_{\varepsilon,\delta} \mathcal{D}$
2. **Independence of $\tilde{f}$:**

$$\forall \text{ arbitrary } \tilde{f} \approx_{\varepsilon,\delta} \mathcal{D} :$$

$$\forall x^* \in \mathcal{X} :$$

$$\mathbb{P}\left[\left|M^{\tilde{f},\mathcal{D}}(x^*, 1^s) - g^{\text{ideal}}(x^*)\right| > n\delta\right] \leq \text{neg}(s)$$

Results for Local Mitigation: Linear

Thm 2 (Local Mitigation $\mathcal{D} \approx$ linear). $\exists$ local mitigator M s.t.

$$\forall\, \mathcal{D} \approx_{\varepsilon,\delta} \text{linear}$$

$$\exists\, g^{\text{ideal}} : \mathbb{R}^n \to \mathbb{R} :$$

1. **Accuracy:** $g^{\text{ideal}} \approx_{\varepsilon,\delta} \mathcal{D}$
2. **Independence of $\tilde{f}$:**

$$\forall \text{ arbitrary } \tilde{f} \approx_{\varepsilon,\delta} \mathcal{D} :$$

$$\forall x^* \in \mathcal{X} :$$

$$\mathbb{P}\left[\left|M^{\tilde{f},\mathcal{D}}(x^*, 1^s) - g^{\text{ideal}}(x^*)\right| > n\delta\right] \leq \text{neg}(s)$$

3. **Efficiency.** Uses $O(s)$ queries

Results for Local Mitigation: Linear

Thm 2 (Local Mitigation $\mathcal{D} \approx$ linear). $\exists$ local mitigator M s.t.

$$\forall\, \mathcal{D} \approx_{\varepsilon,\delta} \text{linear}$$

$$\exists\, g^{\text{ideal}} : \mathbb{R}^n \to \mathbb{R} :$$

1. **Accuracy:** $g^{\text{ideal}} \approx_{\varepsilon,\delta} \mathcal{D}$
2. **Independence of $\tilde{f}$:**

   $$\forall \text{ arbitrary } \tilde{f} \approx_{\varepsilon,\delta} \mathcal{D} :$$

   $$\forall x^* \in \mathcal{X} :$$

   $$\mathbb{P}\left[\left| M^{\tilde{f},\mathcal{D}}(x^*, 1^s) - g^{\text{ideal}}(x^*) \right| > n\delta\right] \leq \text{neg}(s)$$

3. **Efficiency.** Uses $O(s)$ queries

Results for Local Mitigation: Linear

Thm 2 (Local Mitigation $\mathcal{D} \approx$ linear). $\exists$ local mitigator M s.t.

$$\forall\, \mathcal{D} \approx_{\varepsilon,\delta} \text{linear}$$

$$\exists\, g^{\text{ideal}} :\ \mathbb{R}^n \to \mathbb{R} :$$

1. **Accuracy:** $g^{\text{ideal}} \approx_{\varepsilon,\delta} \mathcal{D}$
2. **Independence of $\tilde{f}$:**

   $$\forall \text{ arbitrary } \tilde{f} \approx_{\varepsilon,\delta} \mathcal{D} :$$

   $$\forall x^* \in \mathcal{X} :$$

   $$\mathbb{P}\left[\left| M^{\tilde{f},\mathcal{D}}(x^*, 1^s) - g^{\text{ideal}}(x^*) \right| > n\delta \right] \le \text{neg}(s)$$

3. **Efficiency.** Uses $O(s)$ queries, independent of n 💪💪

Proof Idea

- Why doesn't traditional linear self-correction work?

Proof Idea

- Why doesn't traditional linear self-correction work?
- Correlated sampling lemma

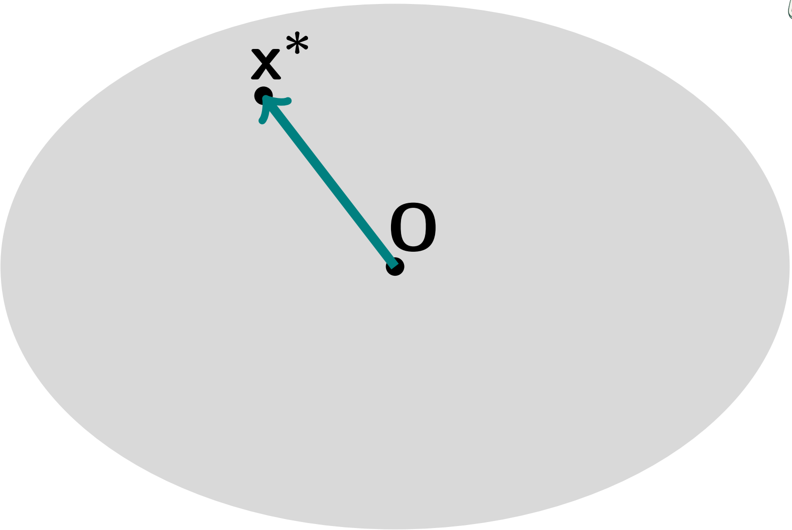
Proof Idea

- Why doesn't traditional linear self-correction work?
- Correlated sampling lemma
- 1-dimensional linear regression

Traditional Linear Self-Correction

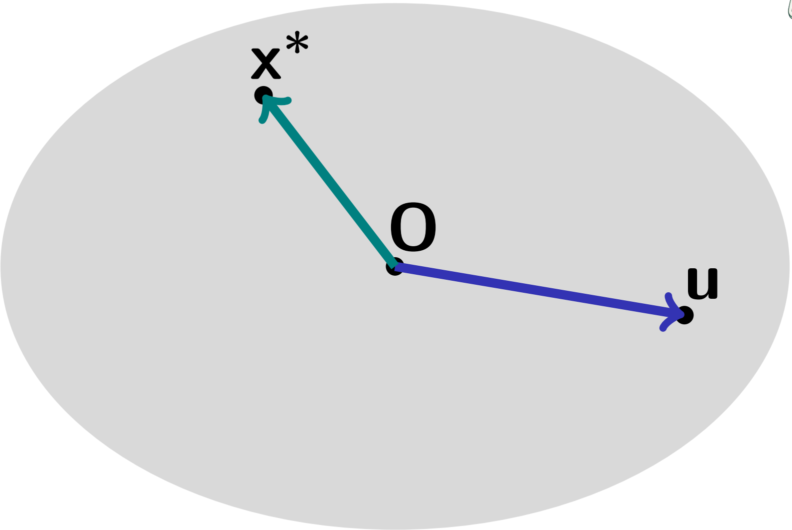
$\mathcal{X} \subseteq \mathbb{R}^n$ is convex

Traditional Linear Self-Correction



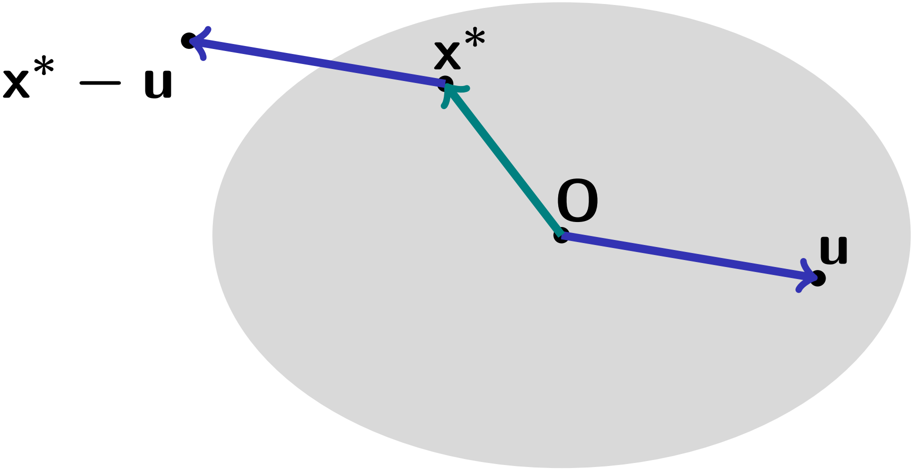
$x^* \in \mathcal{X}$ is arbitrary

Traditional Linear Self-Correction



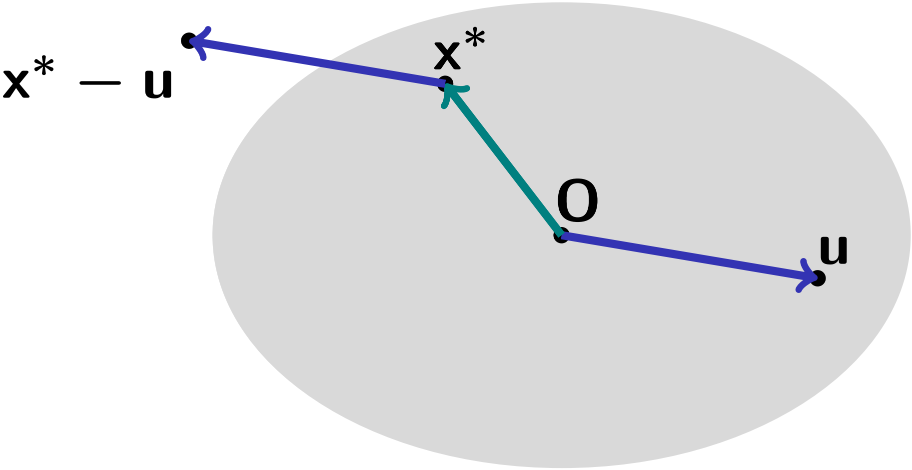
$$u \sim \mathrm{U}(\mathcal{X})$$

Traditional Linear Self-Correction



$$\tilde{f}(x^*) \approx \tilde{f}(u) + \tilde{f}(x^* - u)$$

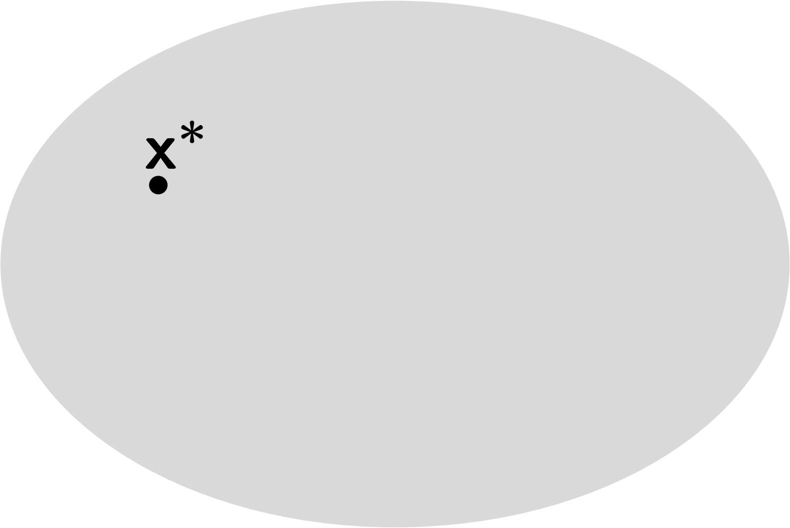
Traditional Linear Self-Correction



$\tilde{f}(x^*) \approx \tilde{f}(u) + \tilde{f}(x^* - u)$;   But $(x^* - u) \not\sim \mathrm{U}(\mathcal{X})$ 🙁

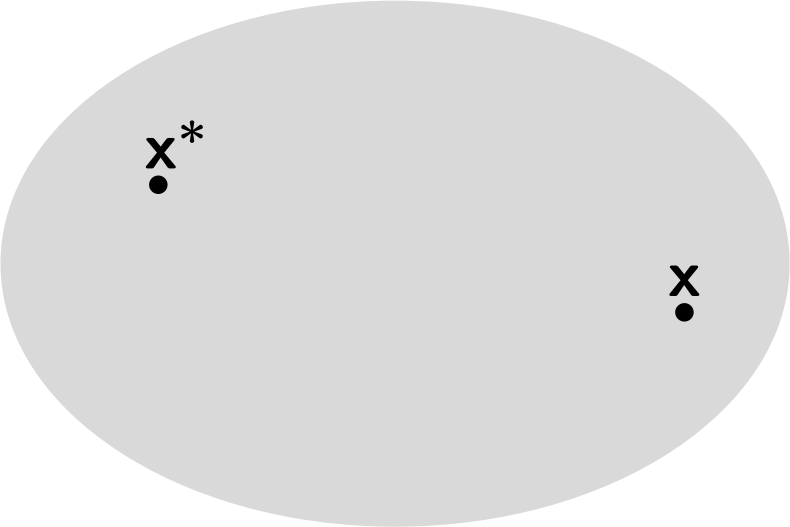
Let's try again...

Correlated Sampling



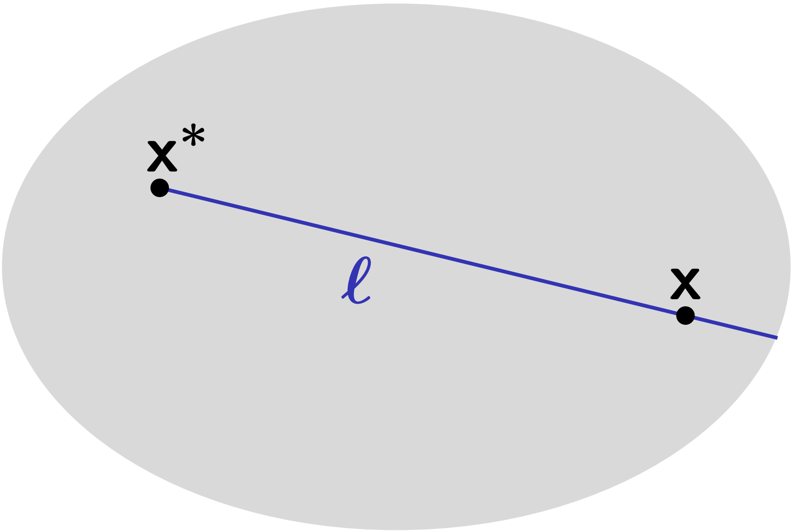
$x^* \in \mathcal{X} \subseteq \mathbb{R}^n$ is arbitrary

Correlated Sampling

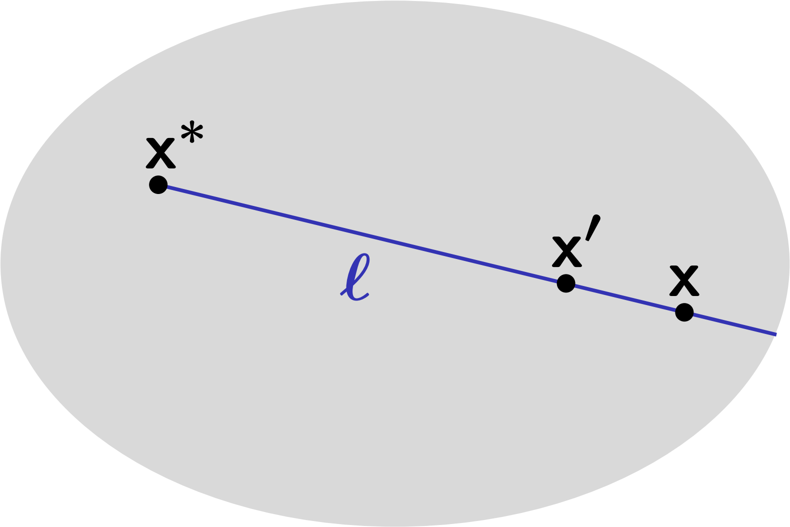


$$x \sim \mathrm{U}(\mathcal{X})$$

Correlated Sampling

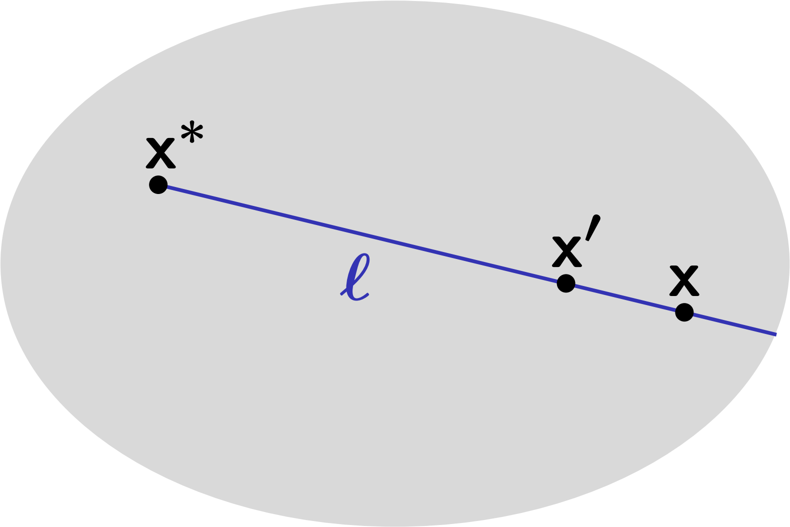


Correlated Sampling



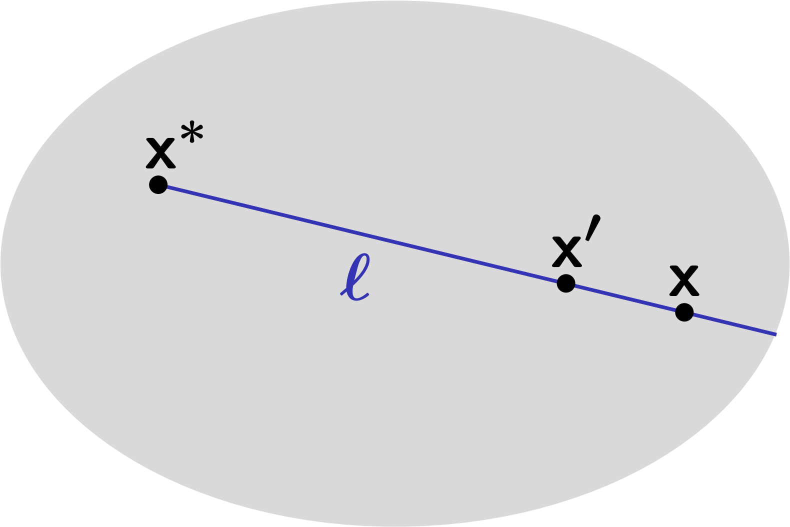
Want: $x' \overset{d}{=} x$

Correlated Sampling



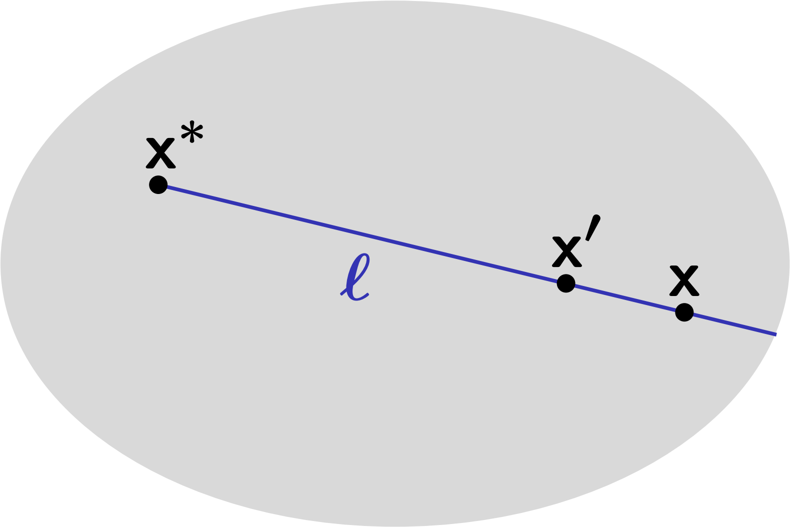
Want: $x' \overset{d}{=} x \ ; \quad x, x' \sim \mathrm{U}(\mathcal{X})$

Correlated Sampling



Want: $x' \overset{d}{=} x$; $x, x' \sim \mathrm{U}(\mathcal{X})$.   **Sample:** $x' \sim \mathrm{U}(\ell)$

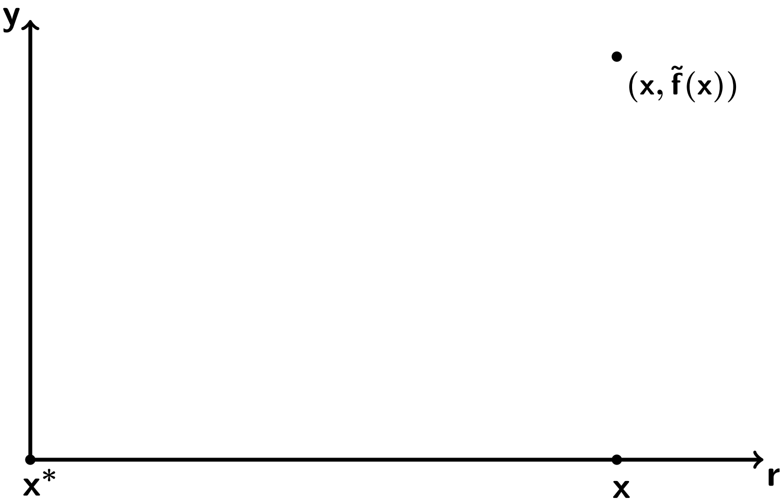
Correlated Sampling



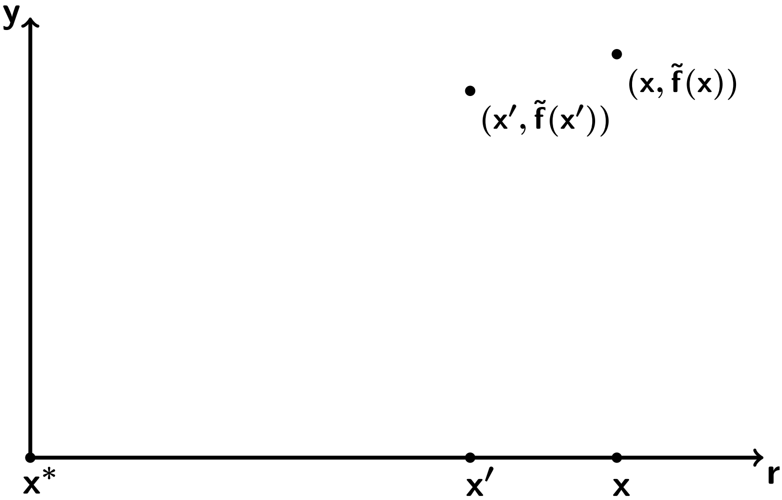
Want: $x' \overset{d}{=} x$; $x, x' \sim \mathrm{U}(\mathcal{X})$.   **Sample:** $\cancel{x' \sim \mathrm{U}(\ell)}$; $x' \propto r^{n-1}$

1-Dimensional Regression

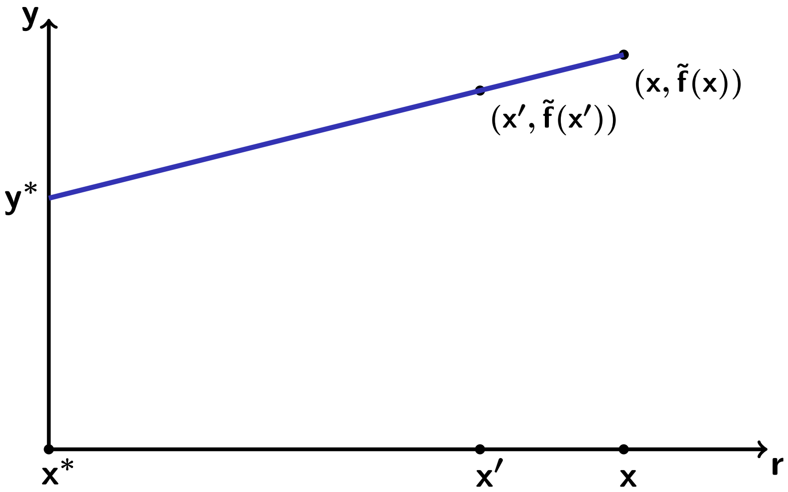
1-Dimensional Regression



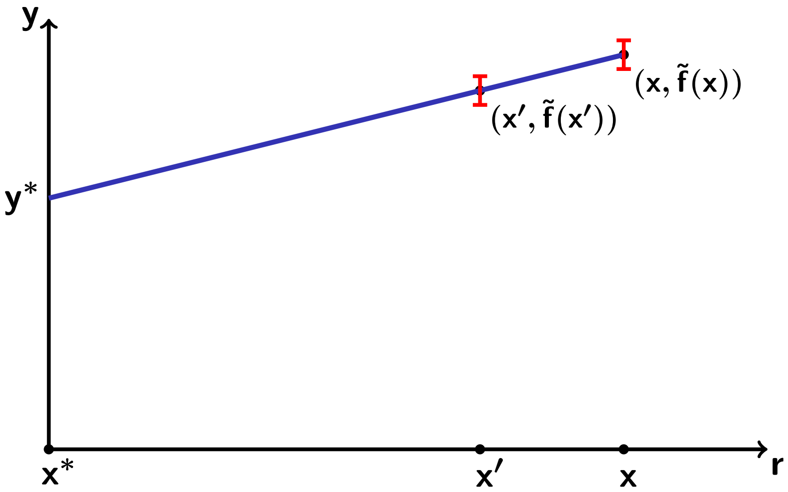
1-Dimensional Regression



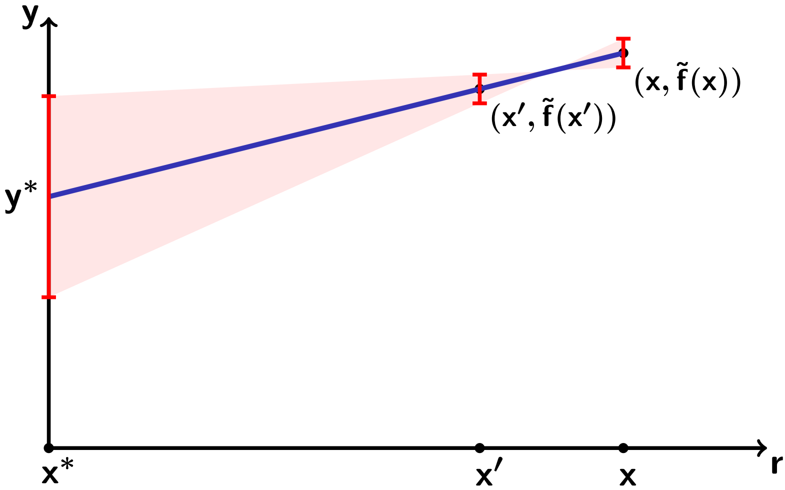
1-Dimensional Regression



1-Dimensional Regression



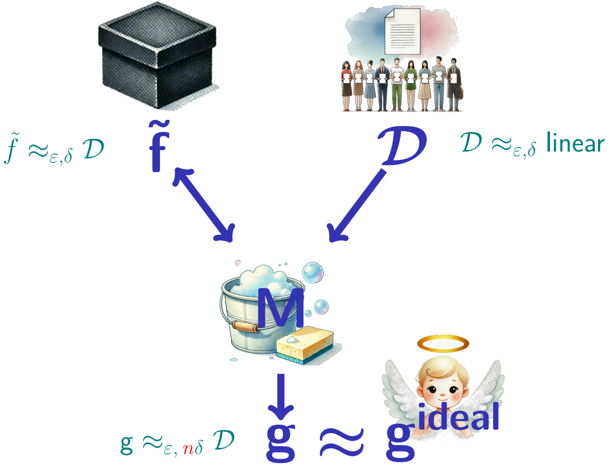
1-Dimensional Regression



$$\delta \mapsto \mathbf{n} \cdot \delta$$

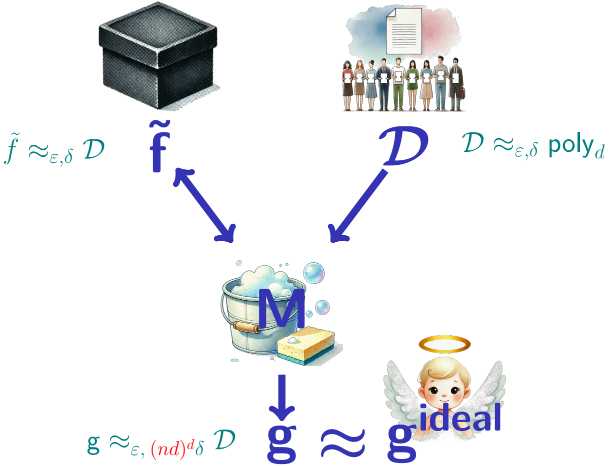
Results for Local Mitigation: Linear

Thm 2 (Local Mitigation $\mathcal{D} \approx$ linear).

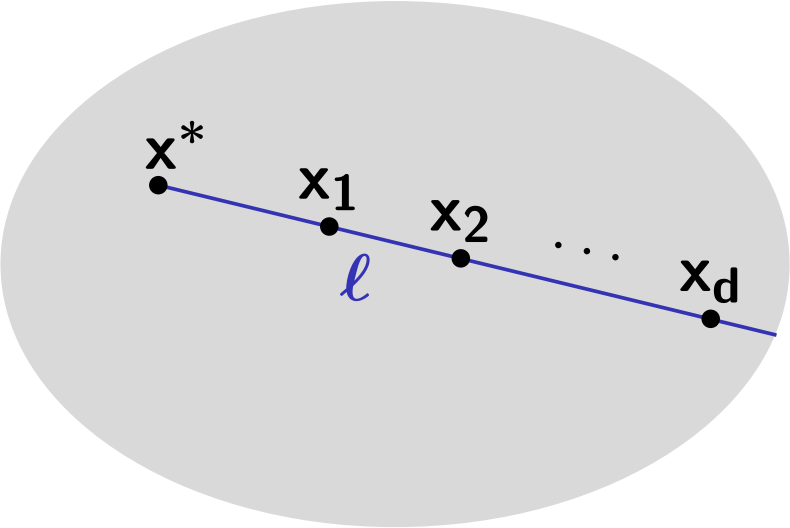


Results for Local Mitigation: Polynomial

Thm 3 (Local Mitigation $\mathcal{D} \approx \mathrm{poly}_d$).



Proof Idea



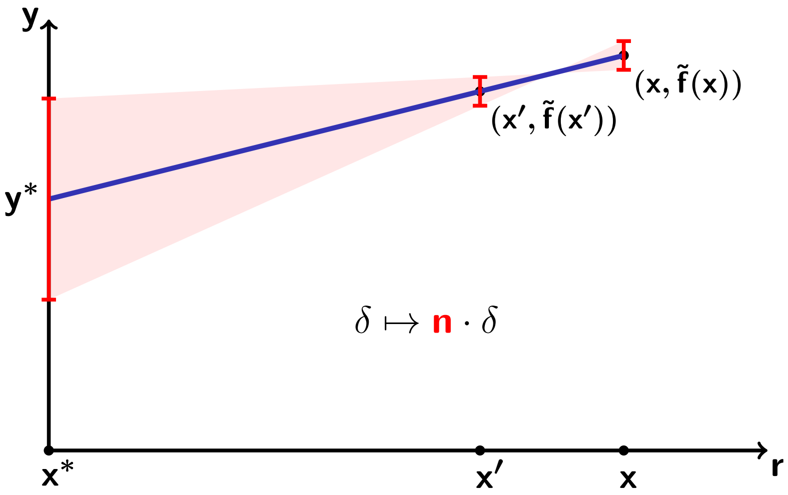
Advanced Local Mitigation

Advanced Local Mitigation

Recall:

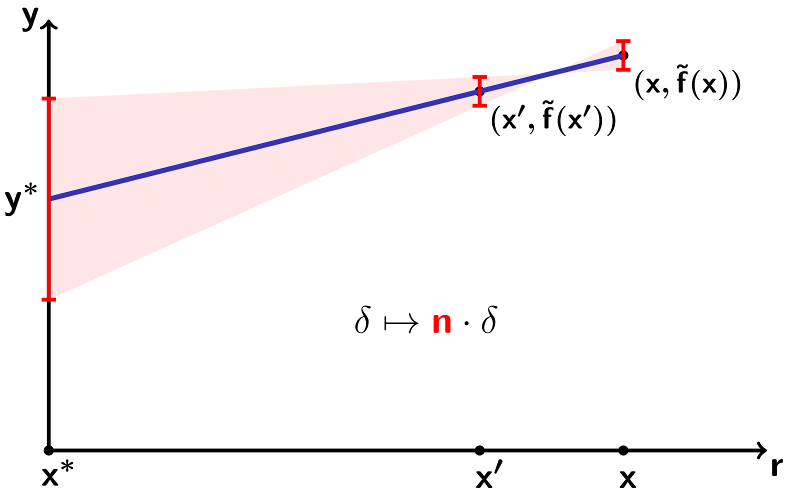
Advanced Local Mitigation

Recall:



Advanced Local Mitigation

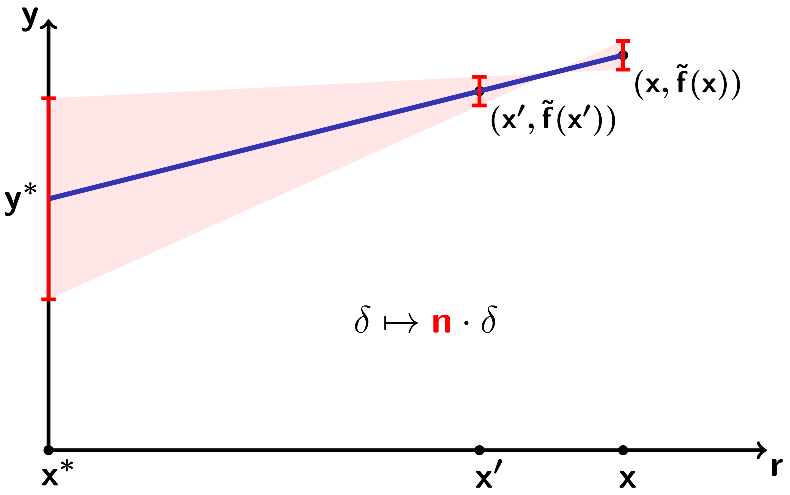
Recall:



Question: Can we do better?

Advanced Local Mitigation

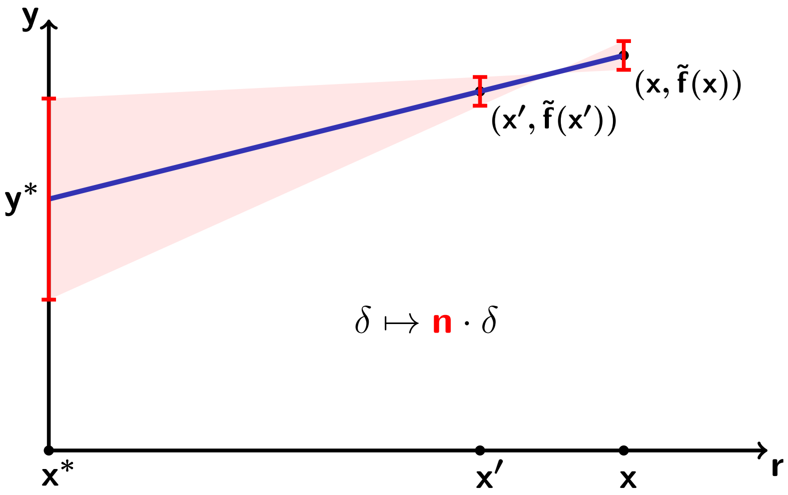
Recall:



Question: Can we do better? $\delta \mapsto o(\mathbf{n}) \cdot \delta$?

Advanced Local Mitigation

Recall:



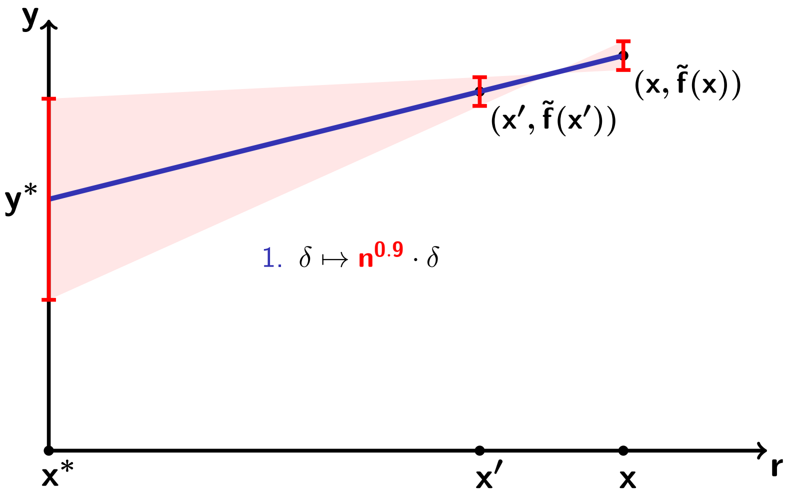
Question: Can we do better? Error pattern not controlled by Eve?

Advanced Local Mitigation

Thm 4 (Advanced Mitigation $\mathcal{D} \approx$ linear). Assume noise in $\mathcal{D}$ is benign.

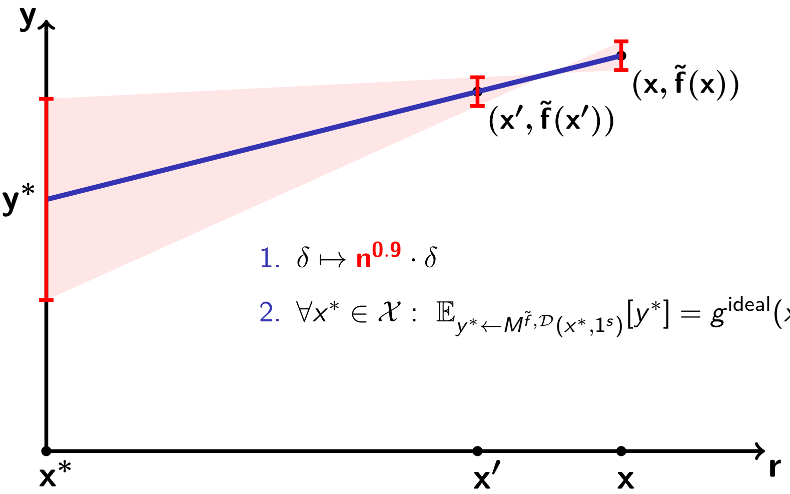
Advanced Local Mitigation

Thm 4 (Advanced Mitigation $\mathcal{D} \approx$ linear). Assume noise in $\mathcal{D}$ is benign. Then



Advanced Local Mitigation

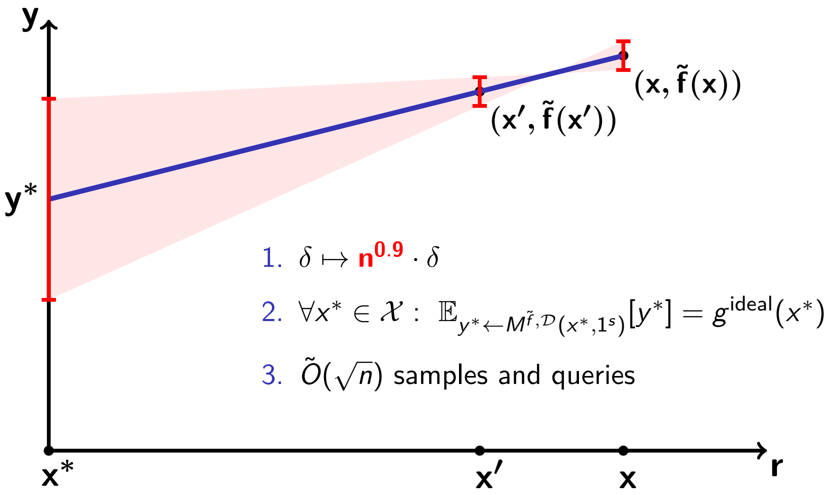
Thm 4 (Advanced Mitigation $\mathcal{D} \approx$ linear). Assume noise in $\mathcal{D}$ is benign. Then



1. $\delta \mapsto \mathbf{n^{0.9}} \cdot \delta$
2. $\forall x^* \in \mathcal{X} : \mathbb{E}_{y^* \leftarrow M^{\tilde{f}, \mathcal{D}}(x^*, 1^s)}[y^*] = g^{\text{ideal}}(x^*)$

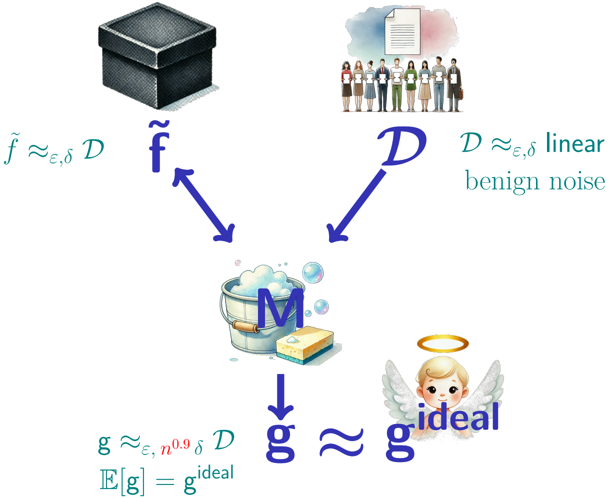
Advanced Local Mitigation

Thm 4 (Advanced Mitigation $\mathcal{D} \approx$ linear). Assume noise in $\mathcal{D}$ is benign. Then

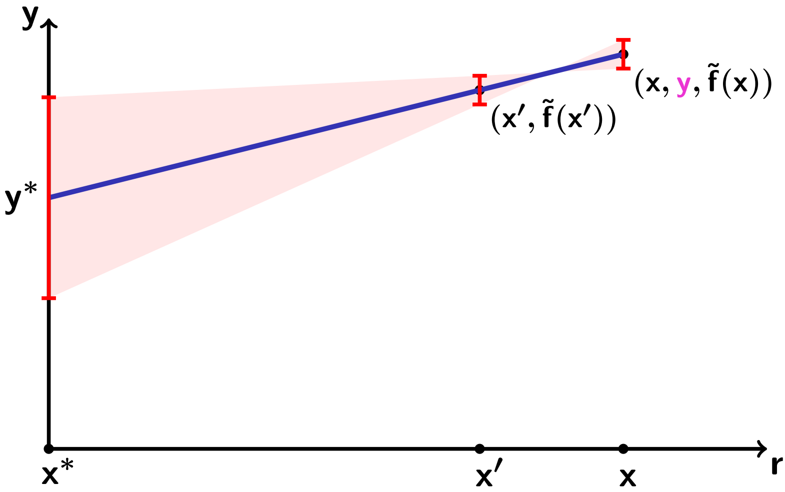


Advanced Local Mitigation

Thm 4 (Advanced Mitigation $\mathcal{D} \approx$ linear). Assume noise in $\mathcal{D}$ is benign. Then

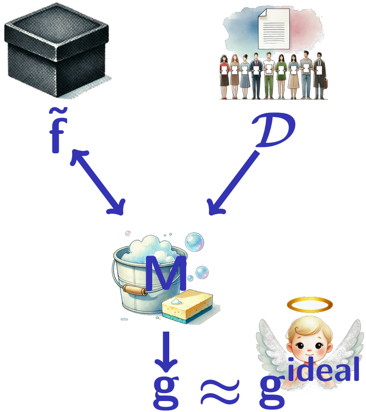


Proof Idea



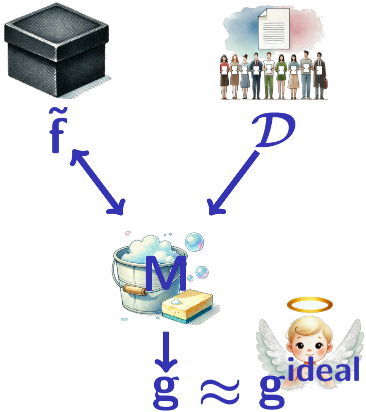
Summary

- Formal definitions of mitigation security



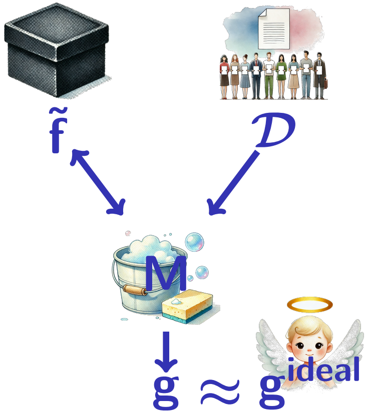
Summary

- Formal definitions of mitigation security
- Using program-self correction / random self-reducibility



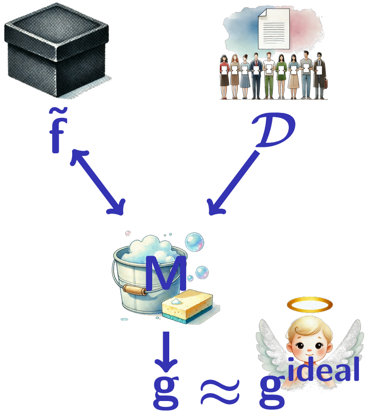
Summary

- Formal definitions of mitigation security
- Using program-self correction / random self-reducibility
- Preliminary constructions:



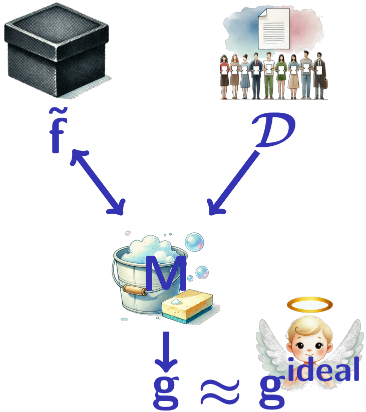
Summary

- Formal definitions of mitigation security
- Using program-self correction / random self-reducibility
- Preliminary constructions:
  - Global mitigation for Fourier heavy



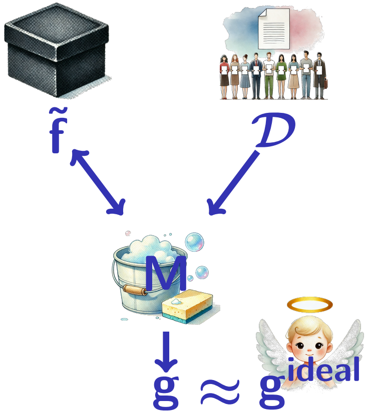
Summary

- Formal definitions of mitigation security
- Using program-self correction / random self-reducibility
- Preliminary constructions:
  - Global mitigation for Fourier heavy
  - Local mitigation



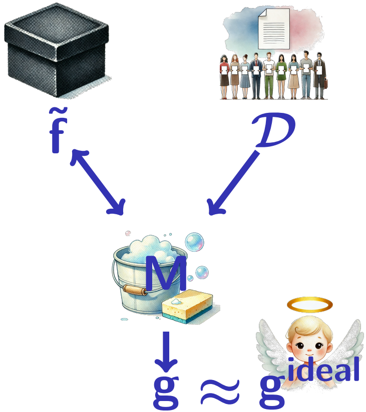
Summary

- Formal definitions of mitigation security
- Using program-self correction / random self-reducibility
- Preliminary constructions:
  - Global mitigation for Fourier heavy
  - Local mitigation
    - Linear



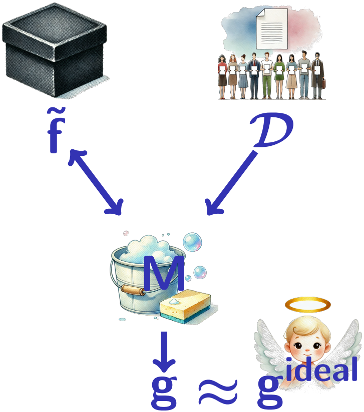
Summary

- Formal definitions of mitigation security
- Using program-self correction / random self-reducibility
- Preliminary constructions:
  - Global mitigation for Fourier heavy
  - Local mitigation
    - Linear
    - Polynomial



Summary

- Formal definitions of mitigation security
- Using program-self correction / random self-reducibility
- Preliminary constructions:
  - Global mitigation for Fourier heavy
  - Local mitigation
    - Linear
    - Polynomial
  - Advanced mitigation



What we know (a haiku):

What we know (a haiku):

Undetectable

What we know (a haiku):

Undetectable

backdoors exist.

What we know (a haiku):

Undetectable

backdoors exist. Structure

What we know (a haiku):

Undetectable

backdoors exist. Structure is

the key to defense.

What we know (a haiku):

Undetectable

backdoors exist. Structure is

the key to defense.

Takeaway (a question):

What we know (a haiku):

Undetectable

backdoors exist. Structure is

the key to defense.

Takeaway (a question):

What other types of structure

can enable mitigation?

Thank You!

References

[BK89] Manuel Blum and Sampath Kannan. Designing programs that check their work. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 86–97, 1989.

[BLR90] Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pages 73–83, 1990.

[GKVZ22] Shafi Goldwasser, Michael P Kim, Vinod Vaikuntanathan, and Or Zamir. Planting undetectable backdoors in machine learning models. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 931–942. IEEE, 2022.

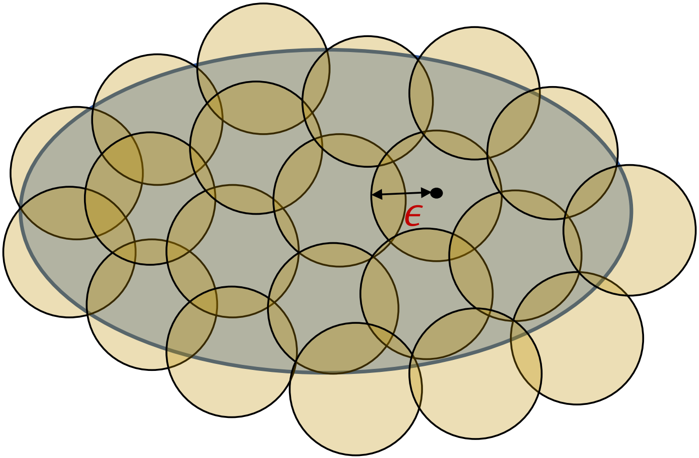
[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proceedings of the fourteenth annual ACM symposium on Theory of computing*, pages 365–377, 1982.

[RR07] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20, 2007.

[Rub90] Ronitt A Rubinfeld. *A mathematical theory of self-checking, self-testing and self-correcting programs*. University of California, Berkeley, 1990.

Appendices

Backdoored Points are a Sparse Set that Covers $\mathcal{X}$



$$\exists \mathcal{B} \subseteq \mathcal{X}\ \forall x \in \mathcal{X}\ \exists \tilde{x} \in \mathcal{B} :\ \ \tilde{x} \approx x\ \wedge\ f(\tilde{x}) = -f(x)$$

Image source: Du, Tu, Yuan, & Tao (2022). Phys. Rev. Lett. 128, 080506

Naïve Local Mitigation for Polynomial Distributions

Question:

- $\exists$ reduction from polynomial regression to linear regression

Naïve Local Mitigation for Polynomial Distributions

Question:

- $\exists$ reduction from polynomial regression to linear regression
- $x \mapsto$ (monomials of $x_1, \ldots, x_n$)

Naïve Local Mitigation for Polynomial Distributions

Question:

- $\exists$ reduction from polynomial regression to linear regression
- $x \mapsto$ (monomials of $x_1, \ldots, x_n$)
- Mitigation for linear distributions (mostly) independent of dimension

Naïve Local Mitigation for Polynomial Distributions

Question:

- $\exists$ reduction from polynomial regression to linear regression
- $x \mapsto$ (monomials of $x_1, \ldots, x_n$)
- Mitigation for linear distributions (mostly) independent of dimension
- So mitigation for polynomial functions is independent of degree? 🤔

Naïve Local Mitigation for Polynomial Distributions

Question:

- $\exists$ reduction from polynomial regression to linear regression
- $x \mapsto$ (monomials of $x_1, \ldots, x_n$)
- Mitigation for linear distributions (mostly) independent of dimension
- So mitigation for polynomial functions is independent of degree? 🤔
- Unfortunately, no 😔

Naïve Local Mitigation for Polynomial Distributions

Question:

- $\exists$ reduction from polynomial regression to linear regression
- $x \mapsto$ (monomials of $x_1, \ldots, x_n$)
- Mitigation for linear distributions (mostly) independent of dimension
- So mitigation for polynomial functions is independent of degree? 🤔
- Unfortunately, no 😔
  - Manifold of monomials is not convex