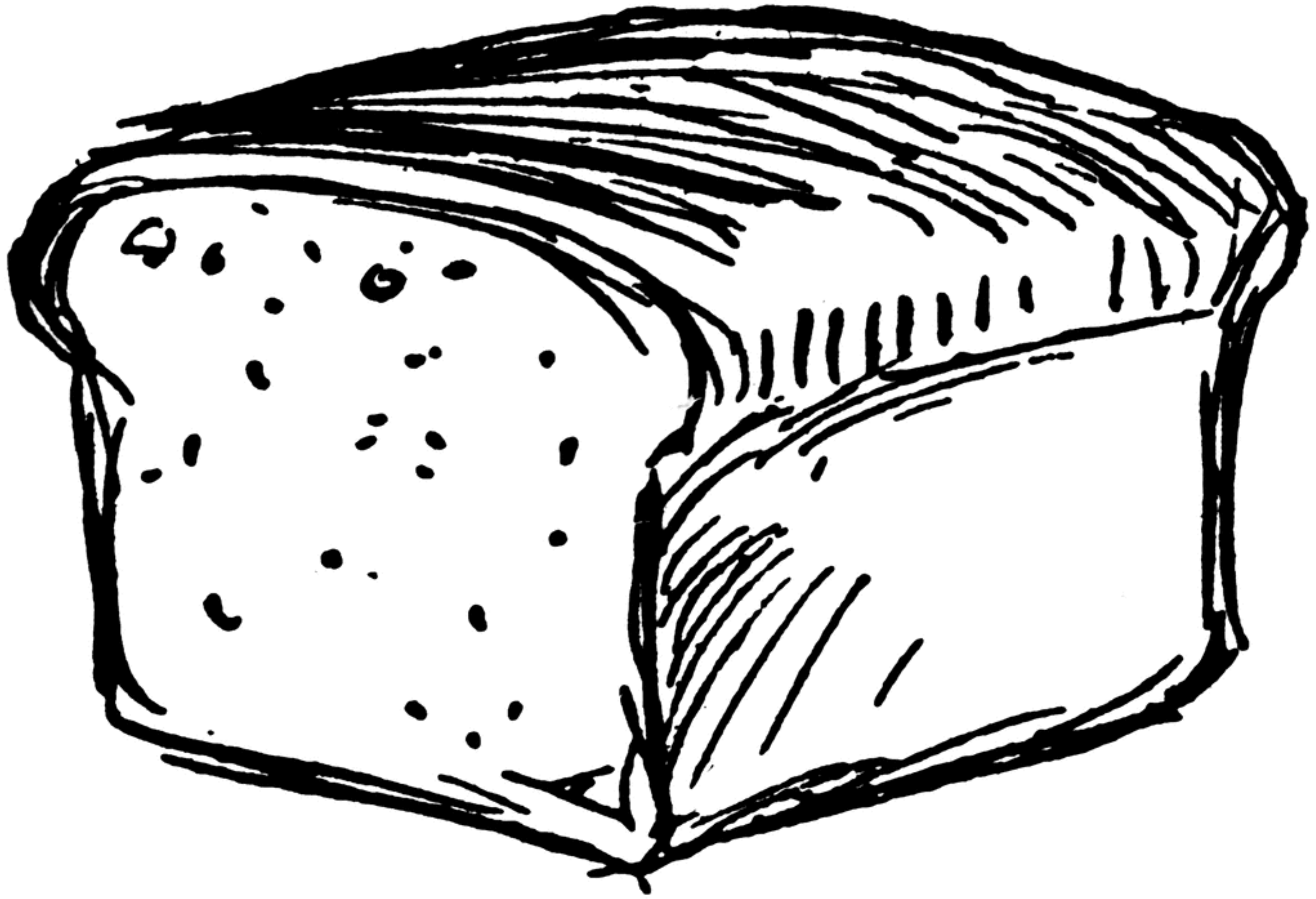
Recent work in Truncated Statistics

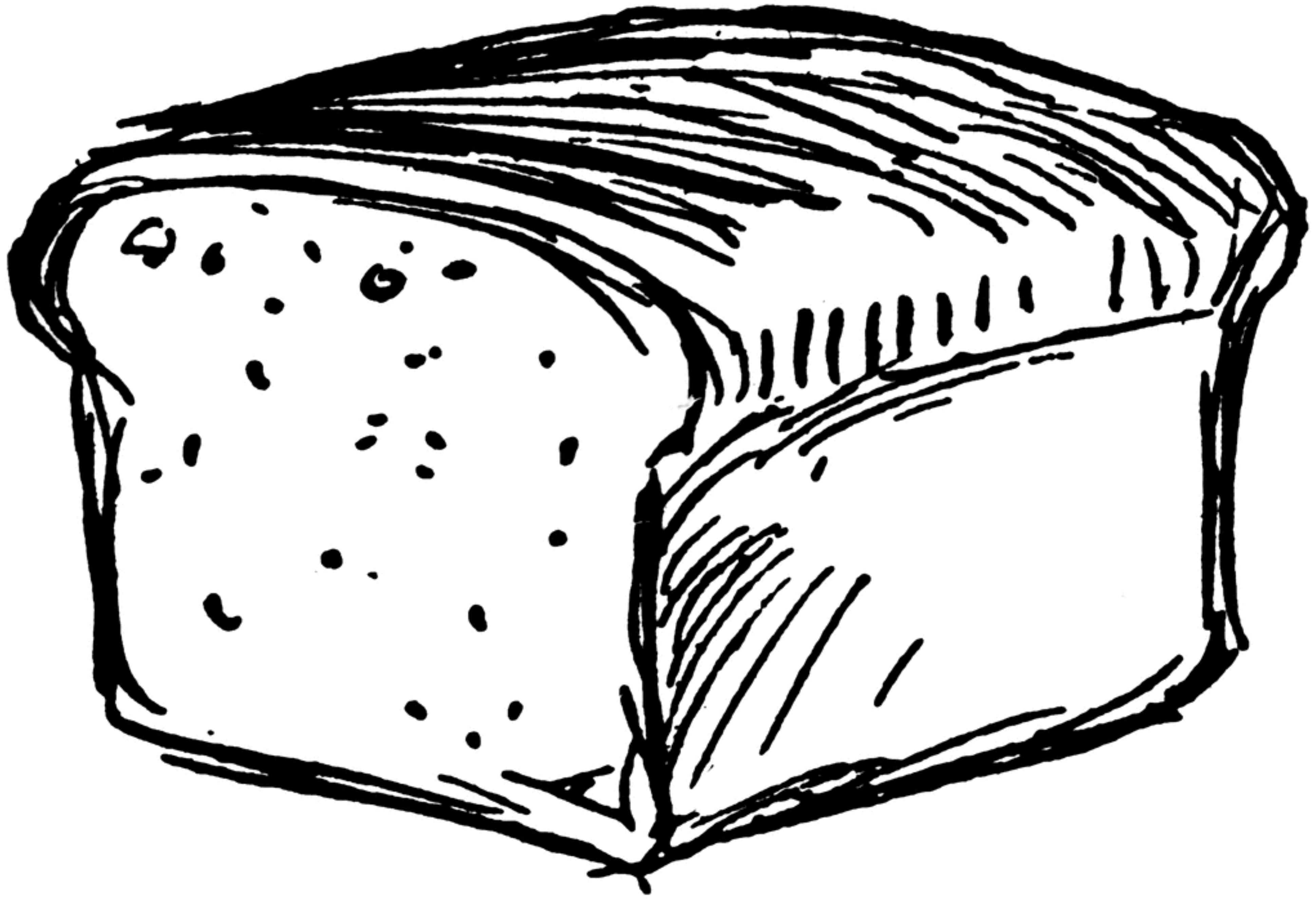
Andrew Ilyas

Motivation: Poincaré and the Baker

Motivation: Poincaré and the Baker

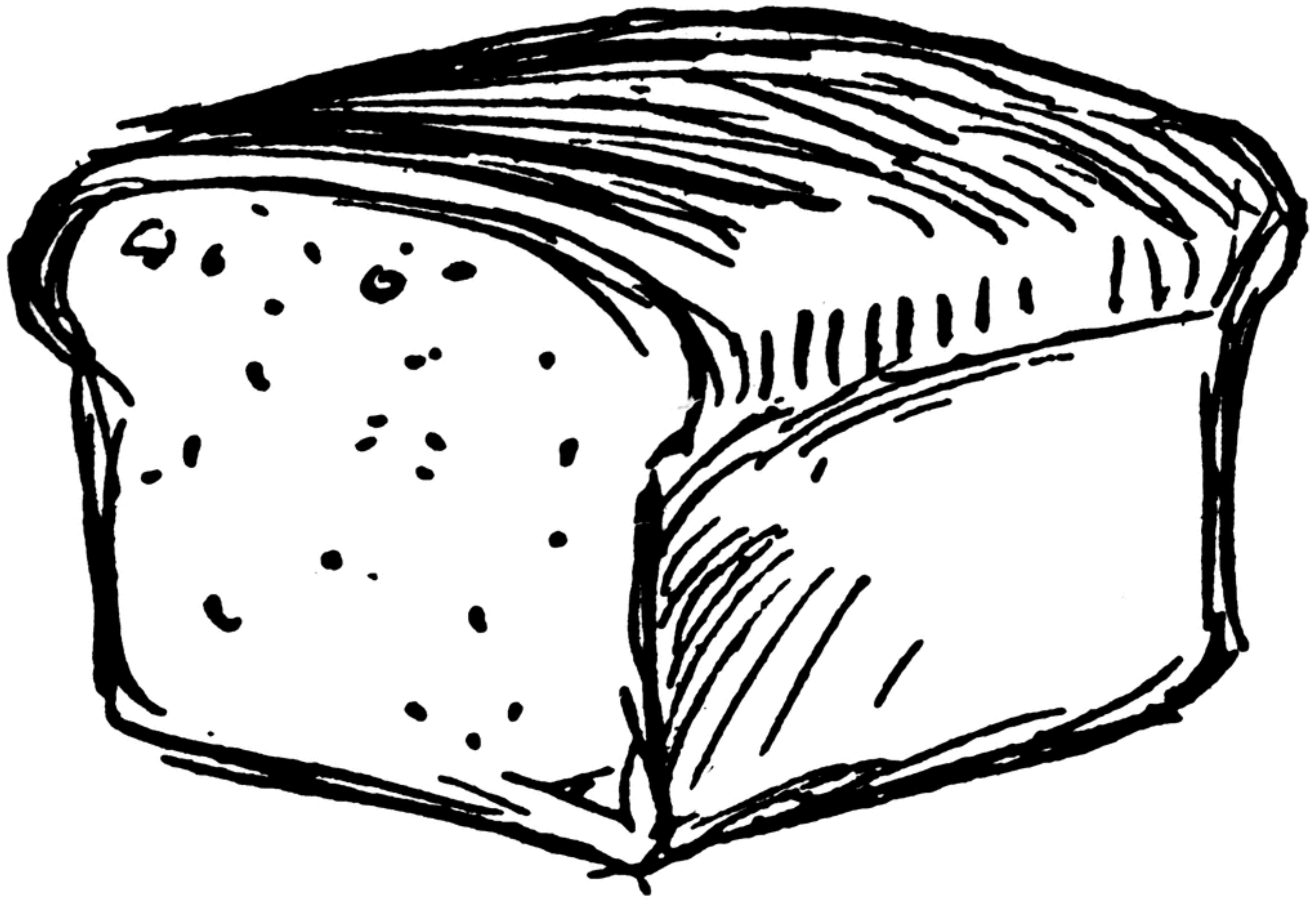


Motivation: Poincaré and the Baker



Claimed weight: 1 kg/loaf

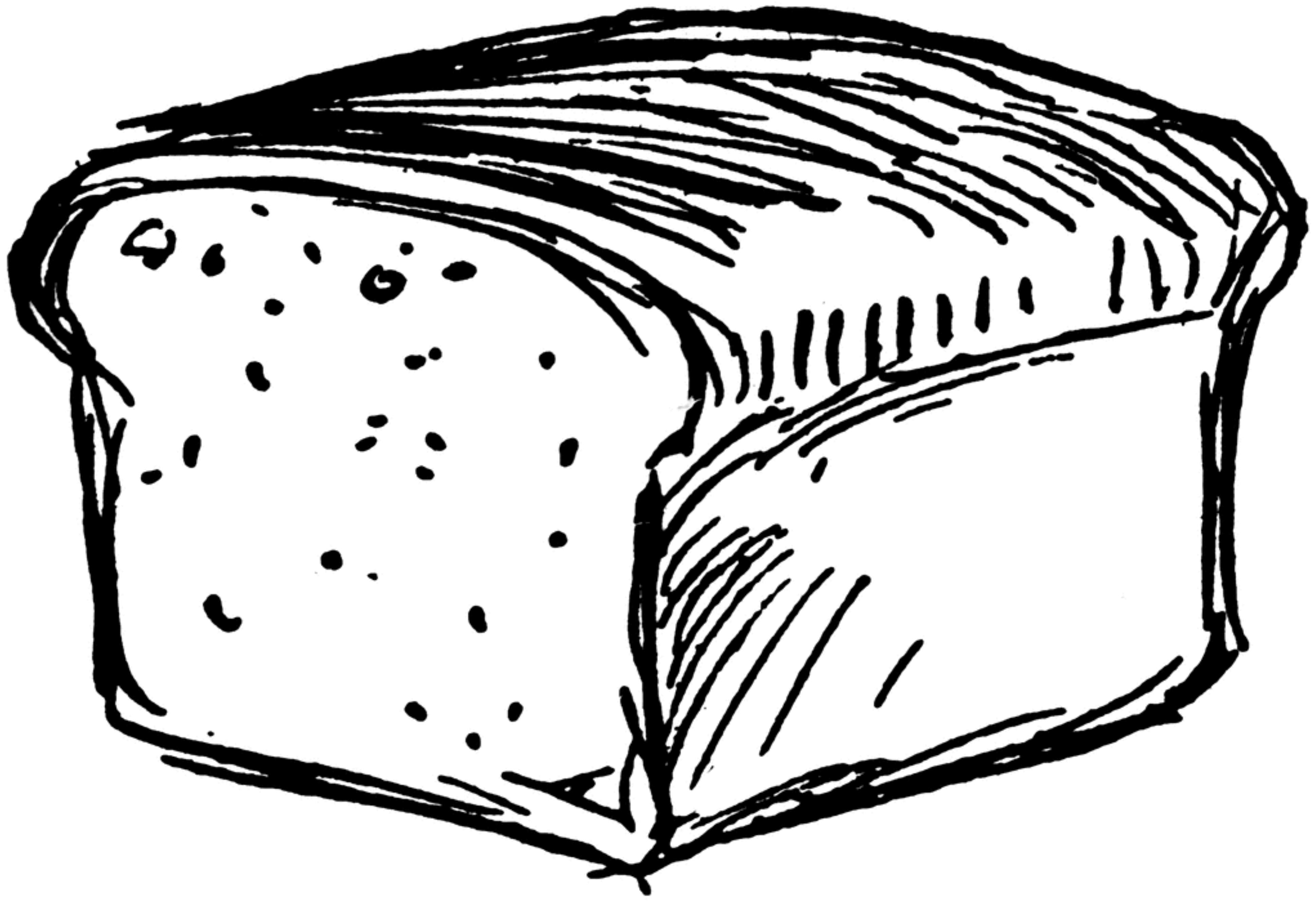
Motivation: Poincaré and the Baker



Claimed weight: 1 kg/loaf

Average weight: 950 g/loaf

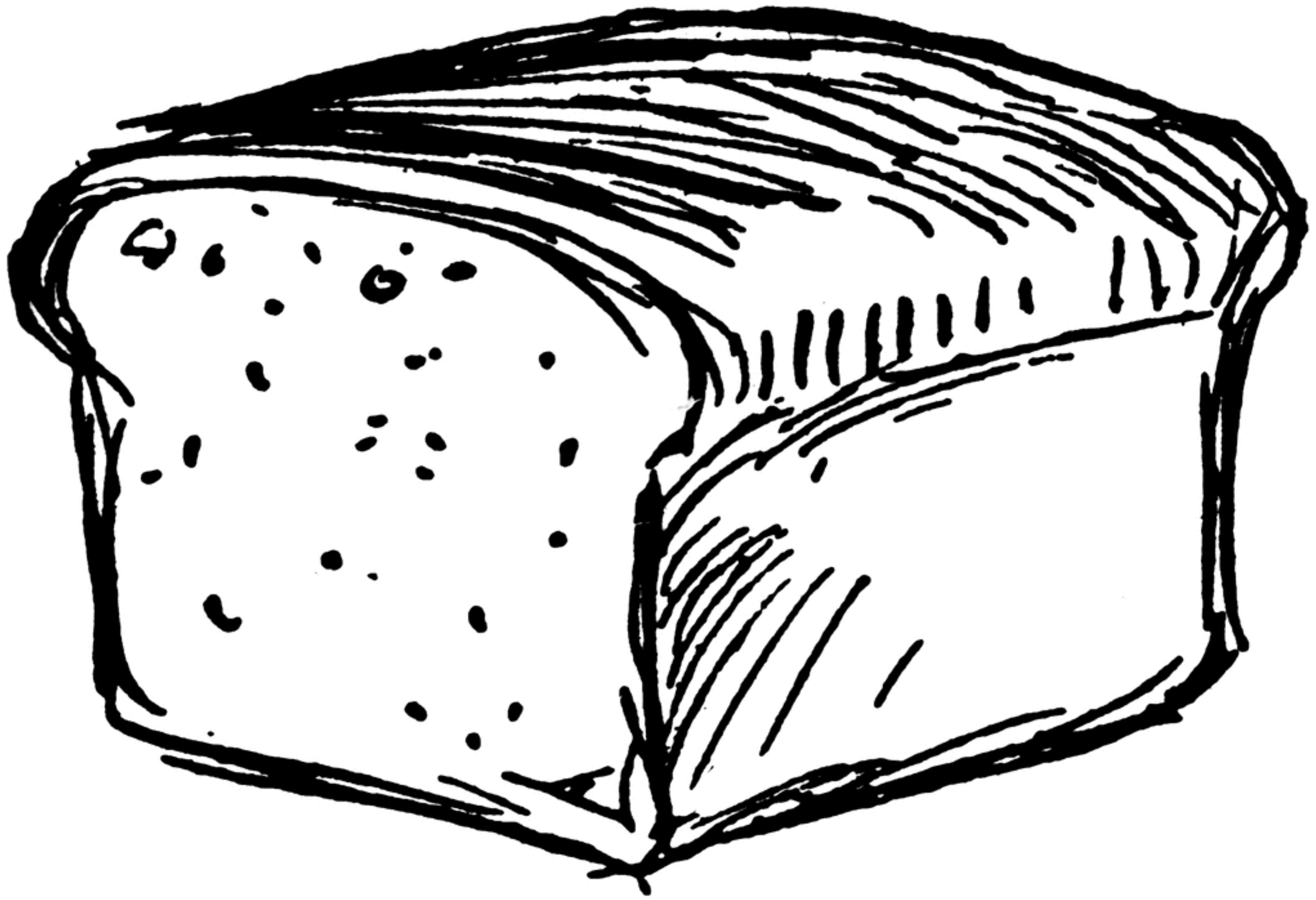
Motivation: Poincaré and the Baker



Claimed weight: 1 kg/loaf

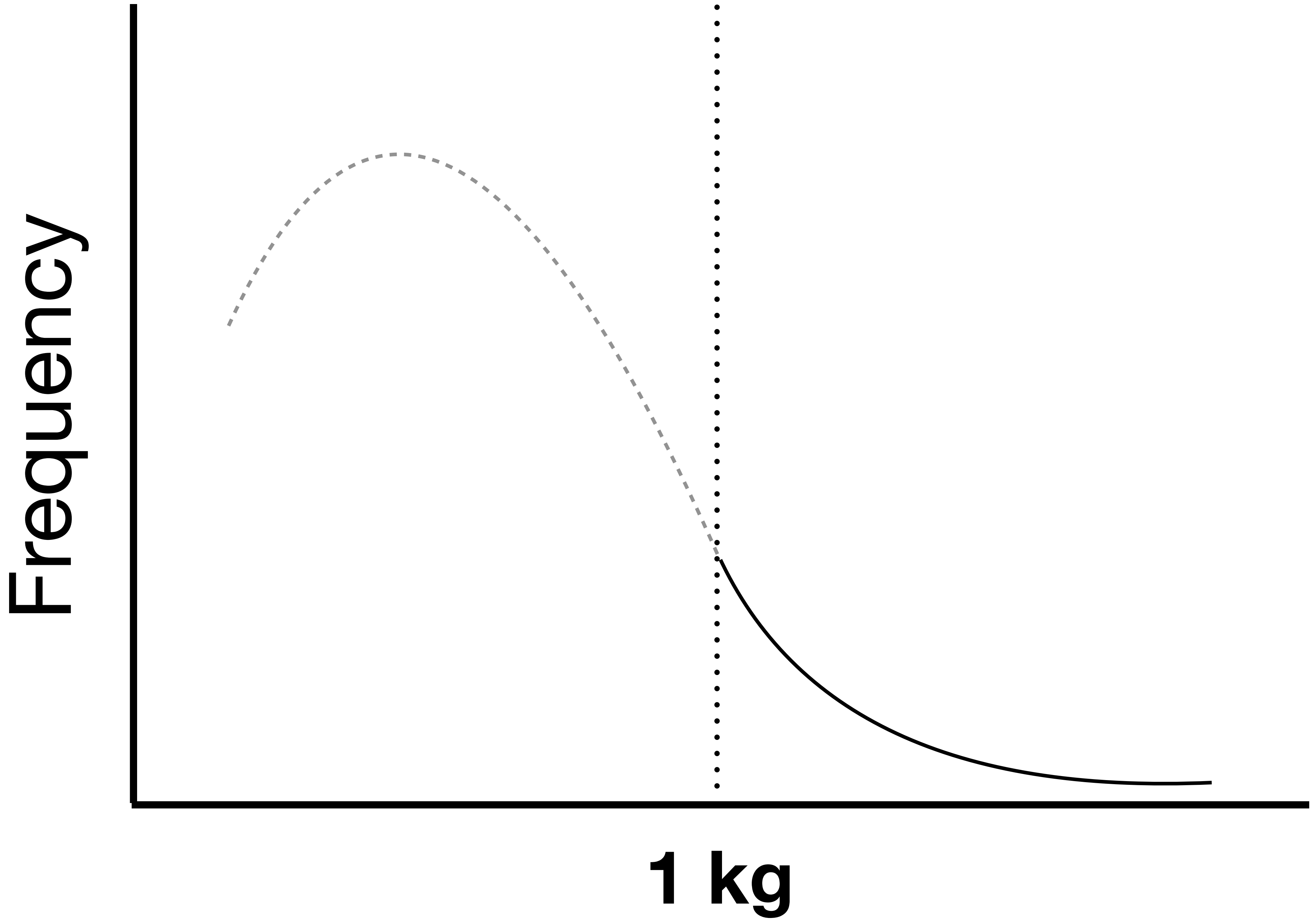
Average weight: 1.05 kg/loaf

Motivation: Poincaré and the Baker



Claimed weight: 1 kg/loaf

Average weight: 1.05 kg/loaf



Outline

- Gaussian parameter estimation [Daskalakis et al, 2018]
- Regression & classification [Daskalakis et al, 2019; Ilyas et al, 2020 (forthcoming)]
- Extensions and Limitations [many works]
- Future work/open problems

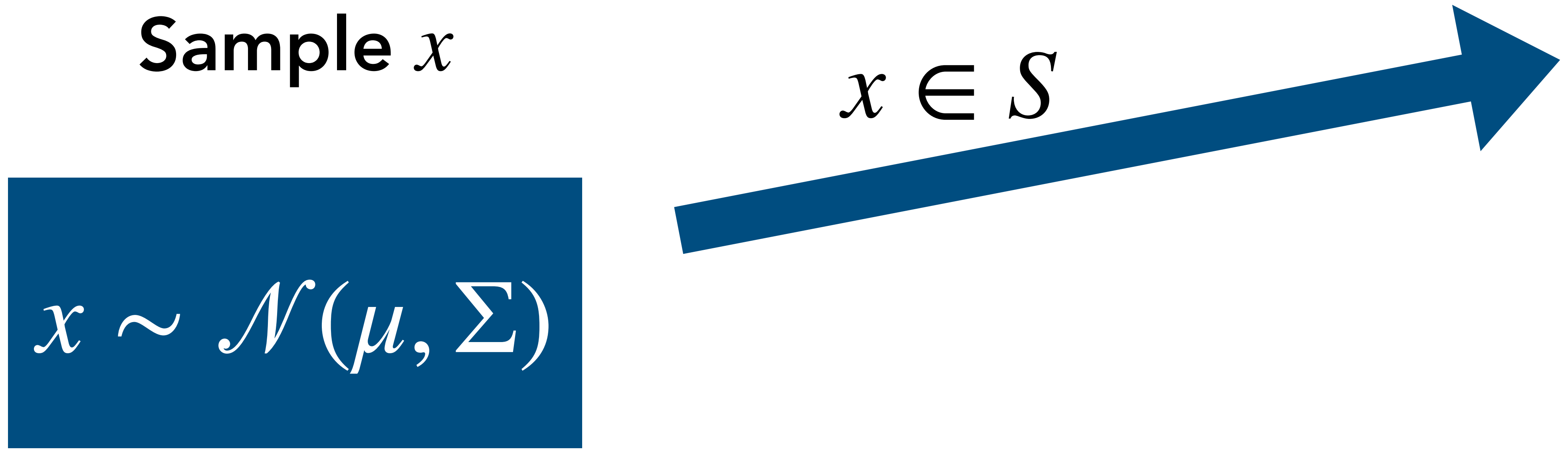
Gaussian Estimation

Gaussian Estimation

Sample x

$$x \sim \mathcal{N}(\mu, \Sigma)$$

Gaussian Estimation

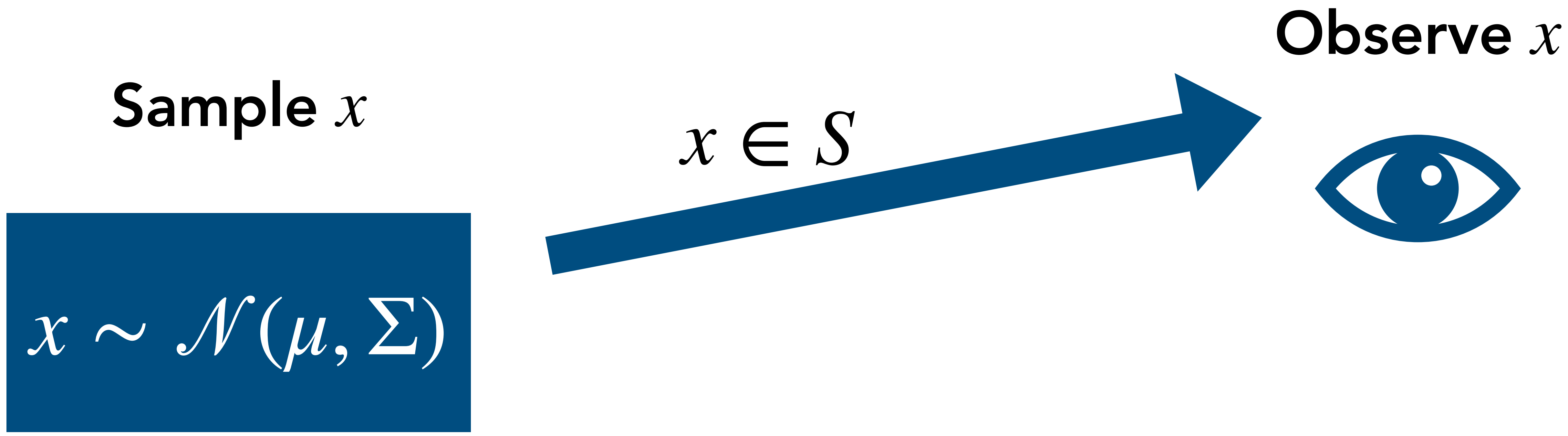


Sample x

$x \in S$

$x \sim \mathcal{N}(\mu, \Sigma)$

Gaussian Estimation



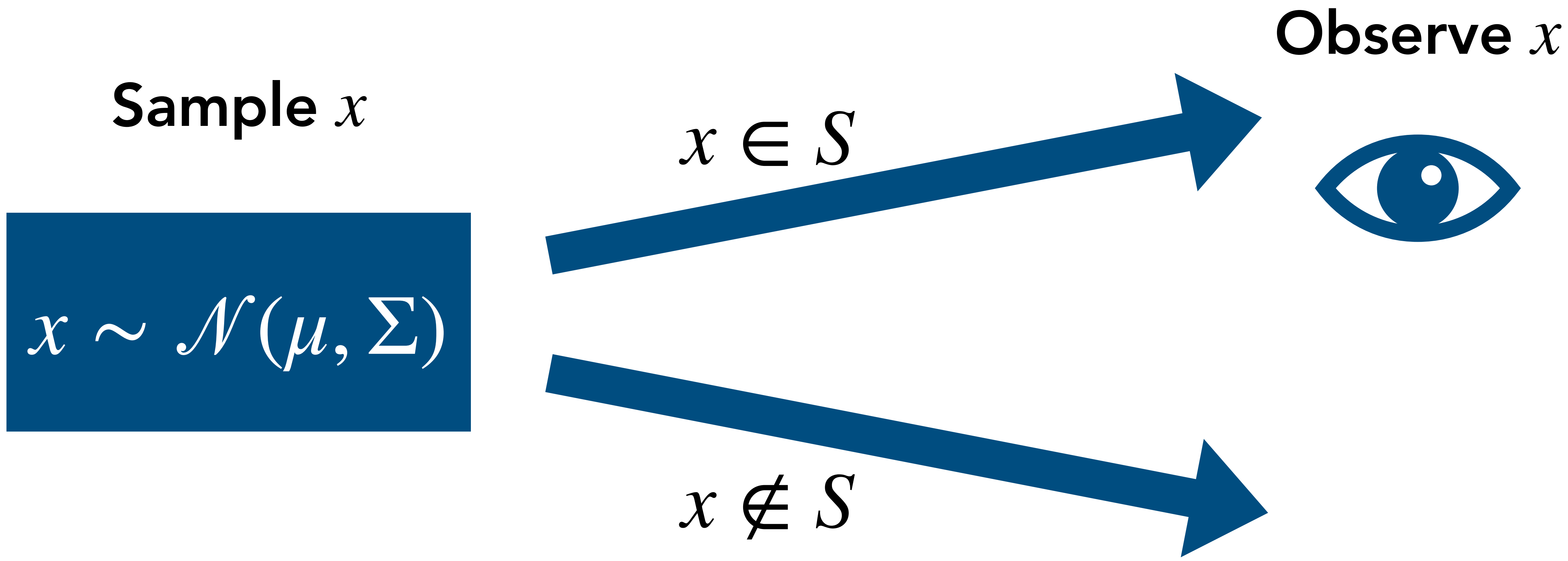
Sample x

$x \sim \mathcal{N}(\mu, \Sigma)$

$x \in S$

Observe x

Gaussian Estimation



Sample x

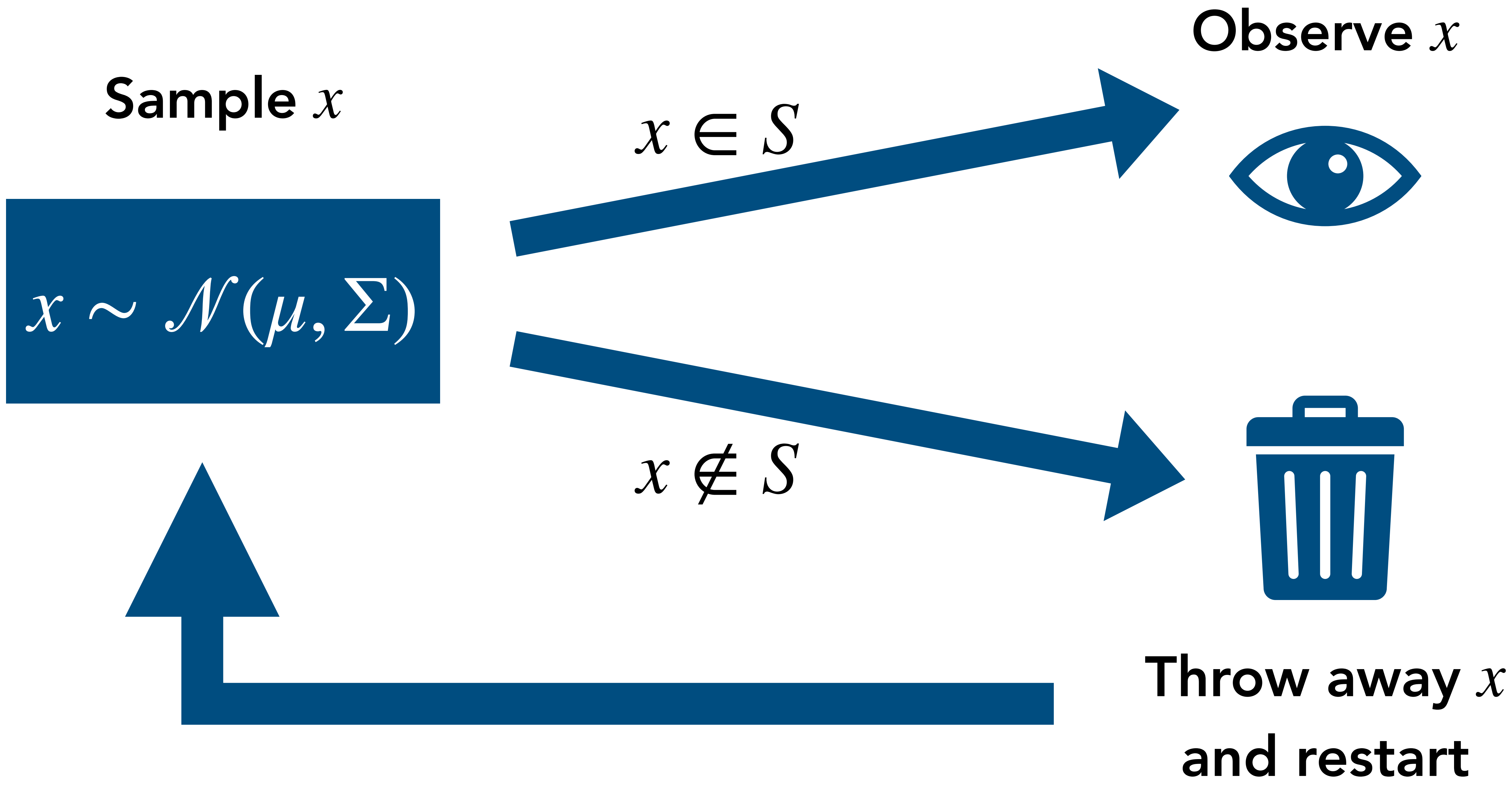
$x \sim \mathcal{N}(\mu, \Sigma)$

$x \in S$

Observe x

$x \notin S$

Gaussian Estimation



Sample x

$x \sim \mathcal{N}(\mu, \Sigma)$

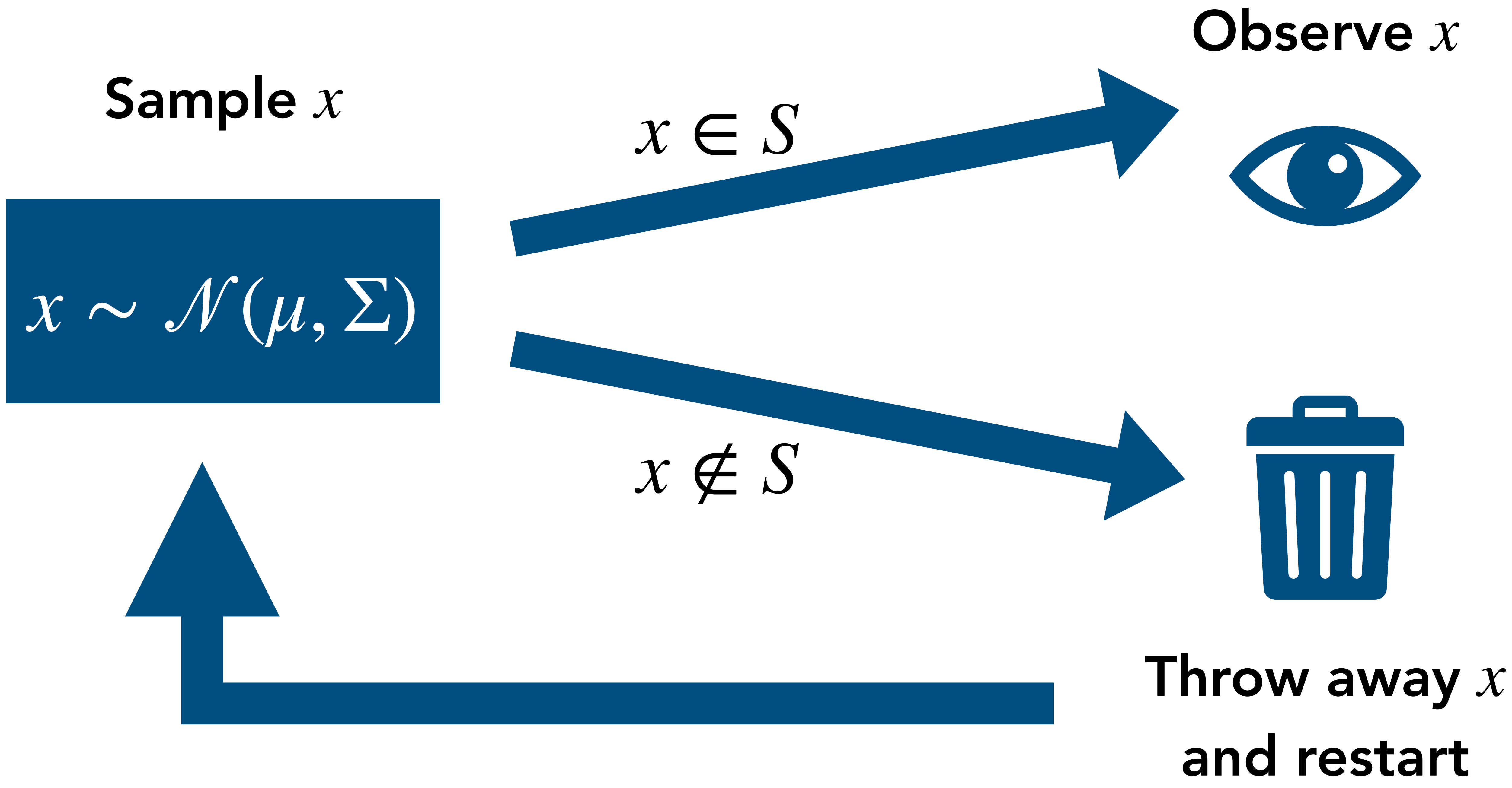
$x \in S$

Observe x

$x \notin S$

Throw away x and restart

Gaussian Estimation



Sample x

$x \sim \mathcal{N}(\mu, \Sigma)$

$x \in S$

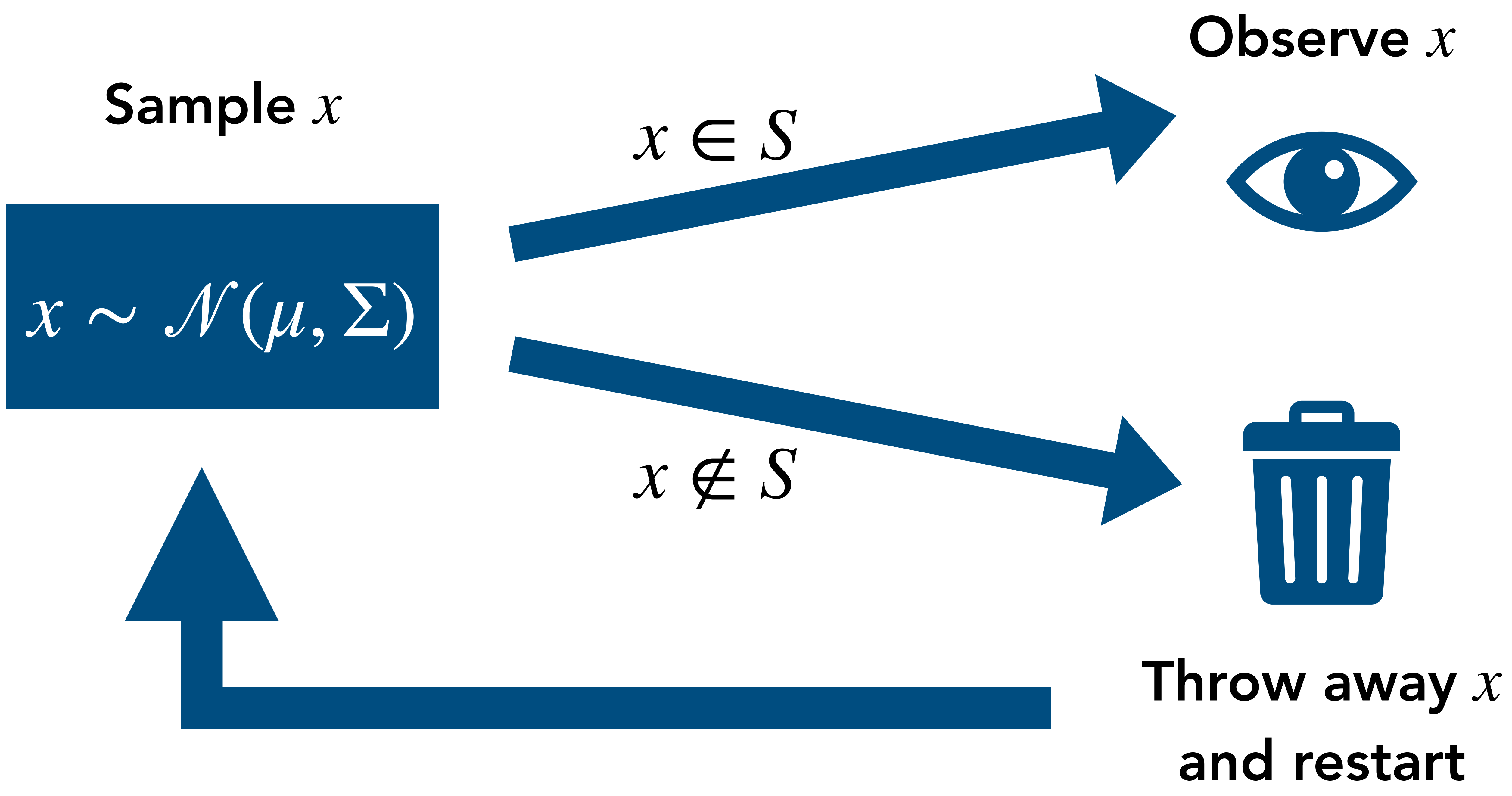
Observe x

$x \notin S$

Throw away x and restart

Goal: Obtain estimates $(\hat{\mu}, \hat{\Sigma}) \approx (\mu, \Sigma)$ from samples

Gaussian Estimation



Sample x

$x \sim \mathcal{N}(\mu, \Sigma)$

$x \in S$

Observe x

$x \notin S$

Throw away x and restart

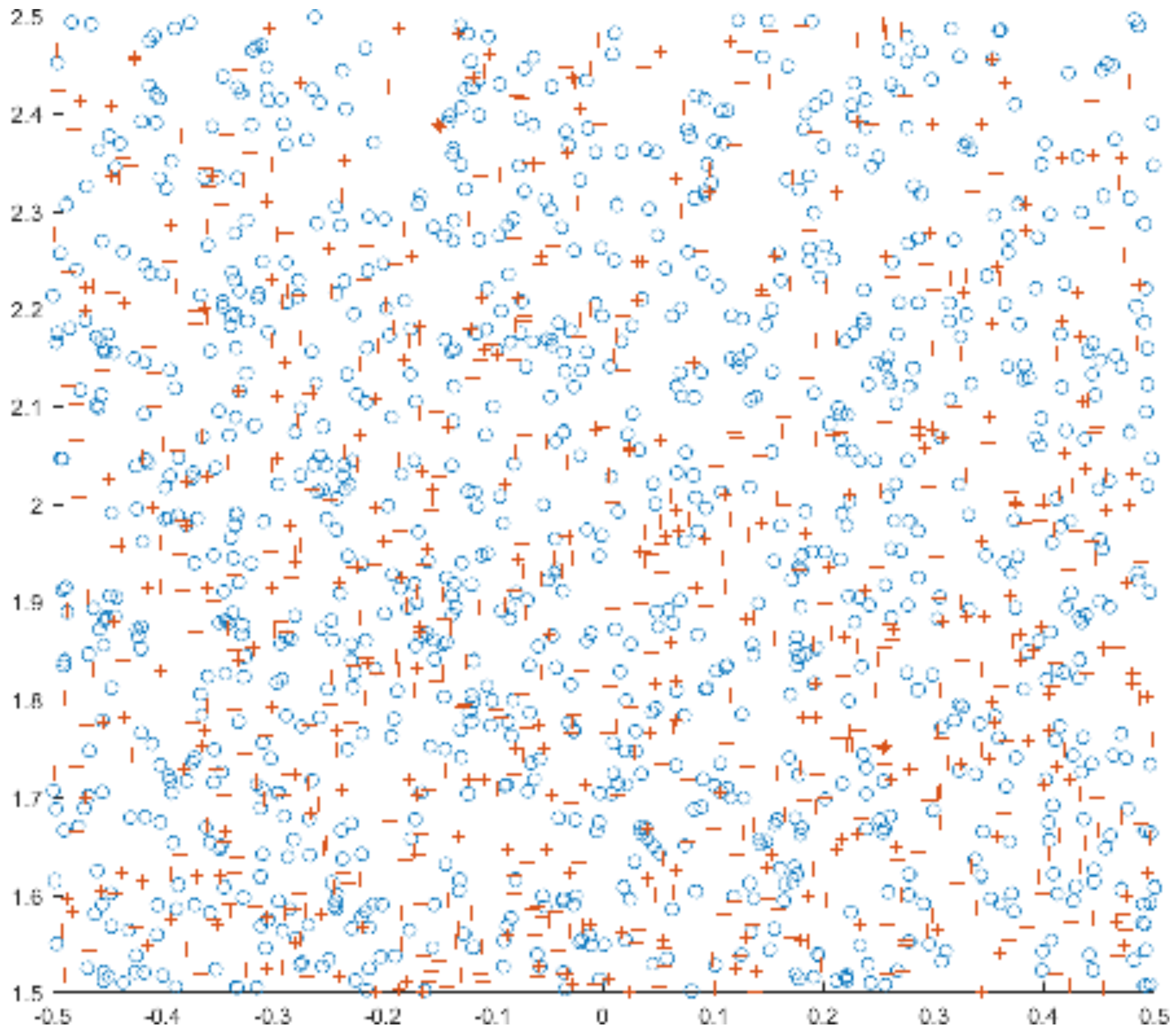


Fig. 1 (Daskalakis et al, 2018): 1000 samples from $\mathcal{N}([0,1], \mathbf{I})$ and from $\mathcal{N}([0,1], 4\,\mathbf{I})$ truncated to $[-0.5, 0.5] \times [1.5, 2.5]$. Which is which?

Goal: Obtain estimates $(\hat{\mu}, \hat{\Sigma}) \approx (\mu, \Sigma)$ from samples

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Standard approach to estimating Gaussian parameters:

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Standard approach to estimating Gaussian parameters:

$$(\hat{\mu}, \hat{\Sigma}) = \arg\max_{(\mu,\Sigma)} \sum_{x_i} \log(f_N(x_i; \mu, \Sigma))$$

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Standard approach to estimating Gaussian parameters:

$$(\hat{\mu}, \hat{\Sigma}) = \arg\max_{(\mu,\Sigma)} \sum_{x_i} \log(f_N(x_i; \mu, \Sigma)) = \arg\max_{(\mu,\Sigma)} \sum_{x_i} (x_i - \mu)^{\top} \Sigma^{-1} (x_i - \mu)$$

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Standard approach to estimating Gaussian parameters:

$$(\hat{\mu}, \hat{\Sigma}) = \arg\max_{(\mu,\Sigma)} \sum_{x_i} \log(f_N(x_i; \mu, \Sigma)) = \arg\max_{(\mu,\Sigma)} \sum_{x_i} (x_i - \mu)^\top \Sigma^{-1} (x_i - \mu)$$

- Take derivative, set to 0:

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Standard approach to estimating Gaussian parameters:

$$(\hat{\mu}, \hat{\Sigma}) = \arg\max_{(\mu,\Sigma)} \sum_{x_i} \log(f_N(x_i; \mu, \Sigma)) = \arg\max_{(\mu,\Sigma)} \sum_{x_i} (x_i - \mu)^\top \Sigma^{-1} (x_i - \mu)$$

- Take derivative, set to 0:

$$\hat{\mu} = \frac{1}{n} \sum_{x_i} x_i$$

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Standard approach to estimating Gaussian parameters:

$$(\hat{\mu}, \hat{\Sigma}) = \arg\max_{(\mu,\Sigma)} \sum_{x_i} \log(f_N(x_i; \mu, \Sigma)) = \arg\max_{(\mu,\Sigma)} \sum_{x_i} (x_i - \mu)^\top \Sigma^{-1} (x_i - \mu)$$

- Take derivative, set to 0:

$$\hat{\mu} = \frac{1}{n} \sum_{x_i} x_i \qquad \hat{\Sigma} = \frac{1}{n} \sum_{x_i} (x_i - \hat{\mu})(x_i - \hat{\mu})^\top$$

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- In the truncated setting, the log-likelihood changes:

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- In the truncated setting, the log-likelihood changes:

$$f(x; \mu, \Sigma, S) = \frac{f_N(x; \mu, \Sigma)}{\int_S f_N(z; \mu, \Sigma)\ dz} \text{ if } x \in S \text{ else } 0$$

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- In the truncated setting, the log-likelihood changes:

$$f(x; \mu, \Sigma, S) = \frac{f_N(x; \mu, \Sigma)}{\int_S f_N(z; \mu, \Sigma)\ dz} \text{ if } x \in S \text{ else } 0$$

$$\log(f(x; \mu, \Sigma, S)) = \log(f_N(x; \mu, \Sigma)) - \log\left(\int_S f_N(z; \mu, \Sigma)\ dz\right)$$

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- In the truncated setting, the log-likelihood changes:

$$f(x;\mu,\Sigma,S) = \frac{f_N(x;\mu,\Sigma)}{\int_S f_N(z;\mu,\Sigma)\ dz} \text{ if } x \in S \text{ else } 0$$

$$\log(f(x;\mu,\Sigma,S)) = \log(f_N(x;\mu,\Sigma)) - \log\left(\int_S f_N(z;\mu,\Sigma)\ dz\right)$$

- No longer has a closed-form solution for the maximizer

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Step 1: Re-parameterize: $T = \Sigma^{-1}$, $v = \Sigma^{-1}\mu$

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Step 1: Re-parameterize: $T = \Sigma^{-1}$, $v = \Sigma^{-1}\mu$
- Step 2: We get an unbiased estimate of the gradient from just truncated samples:

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Step 1: Re-parameterize: $T = \Sigma^{-1}$, $v = \Sigma^{-1}\mu$
- Step 2: We get an unbiased estimate of the gradient from just truncated samples:

$$\nabla_{\mu} \log(f(x; v, T, S)) = \mathbb{E}_{z \sim \mathcal{N}(\mu, \Sigma)}[z \,|\, z \in S] - x$$

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Step 1: Re-parameterize: $T = \Sigma^{-1}$, $v = \Sigma^{-1}\mu$
- Step 2: We get an unbiased estimate of the gradient from just truncated samples:

$$\nabla_{\mu}\log(f(x; v, T, S)) = \mathbb{E}_{z\sim\mathcal{N}(\mu,\Sigma)}[z \mid z \in S] - x$$

$$\nabla_{\Sigma}\log(f(x; v, T, S)) = \frac{1}{2}xx^{\top} - \frac{1}{2}\mathbb{E}_{z\sim\mathcal{N}(\mu,\Sigma)}\left[zz^{\top} \mid z \in S\right]$$

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Step 1: Re-parameterize: $T = \Sigma^{-1}$, $v = \Sigma^{-1}\mu$
- Step 2: We get an unbiased estimate of the gradient from just truncated samples:

$$\nabla_\mu \log(f(x; v, T, S)) = \boxed{\mathbb{E}_{z\sim\mathcal{N}(\mu,\Sigma)}[z \,|\, z \in S]} - x$$

$$\nabla_\Sigma \log(f(x; v, T, S)) = \frac{1}{2}xx^\top - \frac{1}{2}\boxed{\mathbb{E}_{z\sim\mathcal{N}(\mu,\Sigma)}\left[zz^\top \,|\, z \in S\right]}$$

Expected truncated mean/
covariance under current params

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Step 1: Re-parameterize: $T = \Sigma^{-1}$, $v = \Sigma^{-1}\mu$
- Step 2: We get an unbiased estimate of the gradient from just truncated samples:

$$\nabla_\mu \log(f(x; v, T, S)) = \boxed{\mathbb{E}_{z\sim\mathcal{N}(\mu,\Sigma)}[z \,|\, z \in S]} - \boxed{x}$$

$$\nabla_\Sigma \log(f(x; v, T, S)) = \frac{1}{2}\boxed{xx^\top} - \frac{1}{2}\boxed{\mathbb{E}_{z\sim\mathcal{N}(\mu,\Sigma)}\left[zz^\top \,|\, z \in S\right]}$$

Empirical (batch)
mean/covariance

Expected truncated mean/
covariance under current params

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Step 1: Re-parameterize: $T = \Sigma^{-1}$, $v = \Sigma^{-1}\mu$
- Step 2: We get an unbiased estimate of the gradient from just truncated samples:

$$\nabla_\mu \log(f(x; v, T, S)) = \boxed{\mathbb{E}_{z\sim\mathcal{N}(\mu,\Sigma)}[z \mid z \in S]} - \boxed{x}$$

$$\nabla_\Sigma \log(f(x; v, T, S)) = \frac{1}{2}\boxed{xx^\top} - \frac{1}{2}\boxed{\mathbb{E}_{z\sim\mathcal{N}(\mu,\Sigma)}\left[zz^\top \mid z \in S\right]}$$

- **Thus:** can execute SGD on the truncated log-likelihood with **oracle access** to S

Empirical (batch) mean/covariance

Expected truncated mean/ covariance under current params

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Step 3: SGD recovers the true parameters!

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Step 3: SGD recovers the true parameters!
- Ingredients:

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Step 3: SGD recovers the true parameters!
- Ingredients:
    - **Convexity** always holds (not necessarily strong)

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Step 3: SGD recovers the true parameters!
- Ingredients:
  - **Convexity** always holds (not necessarily strong)
  - Guaranteed **constant** probability α of a sample falling into S

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Step 3: SGD recovers the true parameters!
- Ingredients:
  - **Convexity** always holds (not necessarily strong)
  - Guaranteed **constant** probability α of a sample falling into S
  - Efficient projection algorithm into the set of valid parameters (defined by α)

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Step 3: SGD recovers the true parameters!
- Ingredients:
    - **Convexity** always holds (not necessarily strong)
    - Guaranteed **constant** probability α of a sample falling into S
    - Efficient projection algorithm into the set of valid parameters (defined by α)
    - Strong convexity within the projection set: $\mathbf{H} \succeq C \cdot \alpha^4 \cdot \lambda_m(T^{-1}) \cdot \mathbf{I}$

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Step 3: SGD recovers the true parameters!
- Ingredients:
  - **Convexity** always holds (not necessarily strong)
  - Guaranteed **constant** probability α of a sample falling into S
  - Efficient projection algorithm into the set of valid parameters (defined by α)
  - Strong convexity within the projection set: $\mathbf{H} \succeq C \cdot \alpha^4 \cdot \lambda_m(T^{-1}) \cdot \mathbf{I}$
  - Good initialization point (i.e., assigns constant mass to S)

Theme: Maximum Likelihood Estimation

Projected Gradient Descent on the Negative Log-Likelihood (NLL)

- Step 3: SGD recovers the true parameters!
- Ingredients:
  - **Convexity** always holds (not necessarily strong)
  - Guaranteed **constant** probability α of a sample falling into S
  - Efficient projection algorithm into the set of valid parameters (defined by α)
  - Strong convexity within the projection set: $\mathbf{H} \succeq C \cdot \alpha^4 \cdot \lambda_m(T^{-1}) \cdot \mathbf{I}$
  - Good initialization point (i.e., assigns constant mass to S)
- **Result:** Efficient algorithm for recovering parameters from truncated data!

Truncation bias in regression

Truncation bias in regression

- **Goal:** infer the effect of height x_i on basketball ability y_i

Truncation bias in regression

- **Goal:** infer the effect of height x_i on basketball ability y_i
- **Strategy:** linear regression

Truncation bias in regression

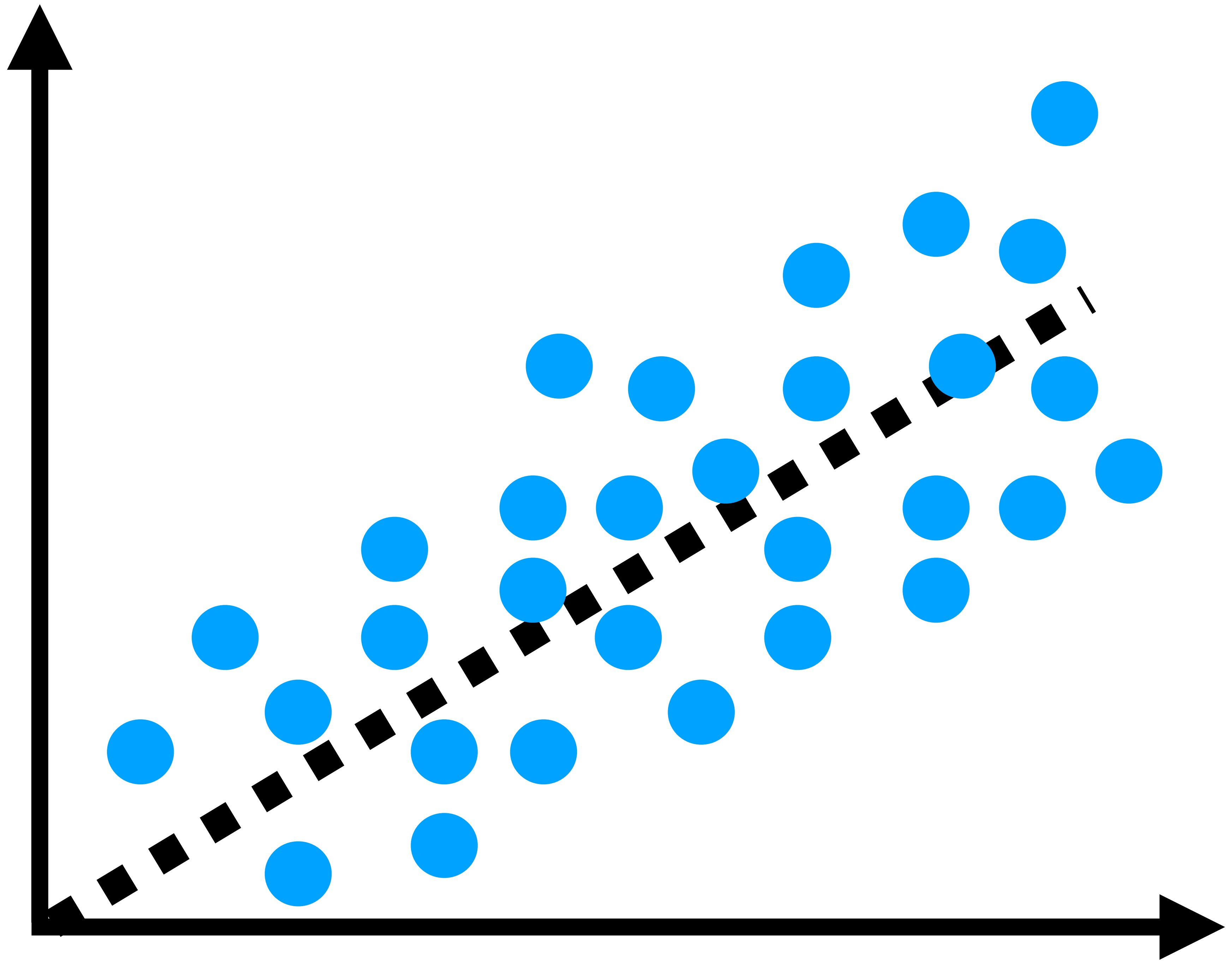
What we expect:

- **Goal:** infer the effect of height x_i on basketball ability y_i
- **Strategy:** linear regression

Truncation bias in regression

- **Goal:** infer the effect of height x_i on basketball ability y_i
- **Strategy:** linear regression

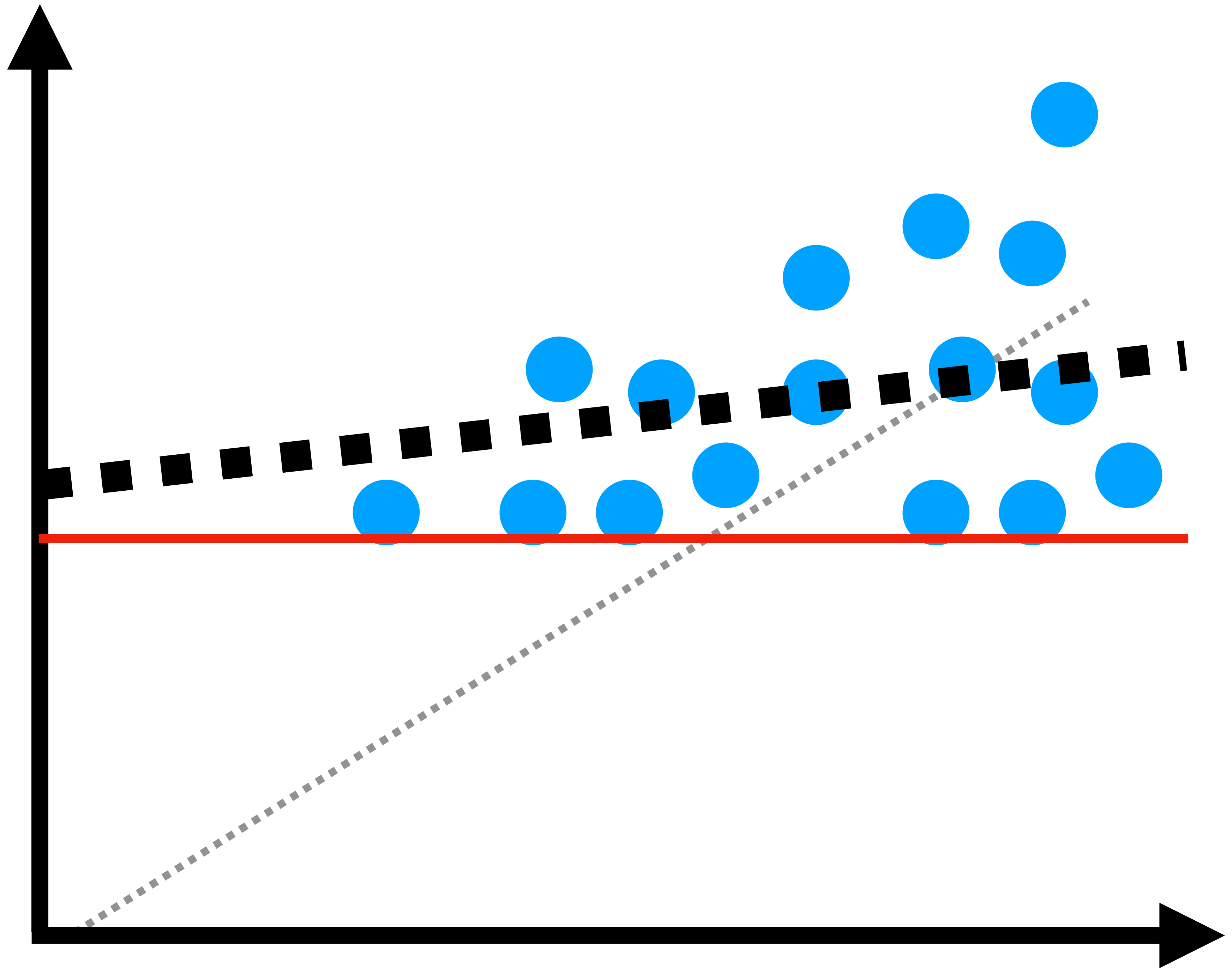
What we expect:



Bias from truncation: an illustration

- **Goal:** infer the effect of height x_i on basketball ability y_i
- **Strategy:** linear regression

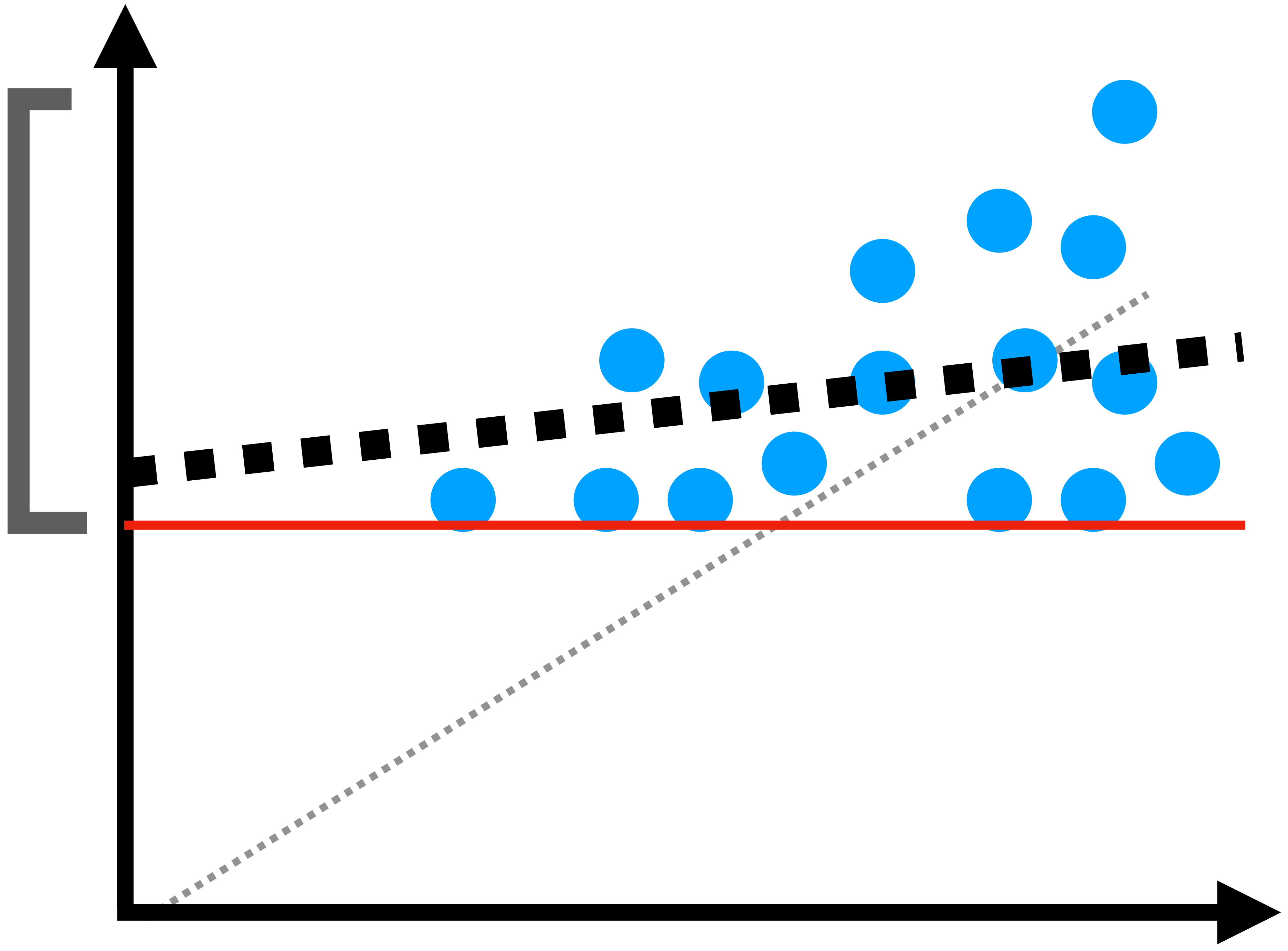
What we get:



Bias from truncation: an illustration

- **Goal:** infer the effect of height x_i on basketball ability y_i
- **Strategy:** linear regression

What we get:



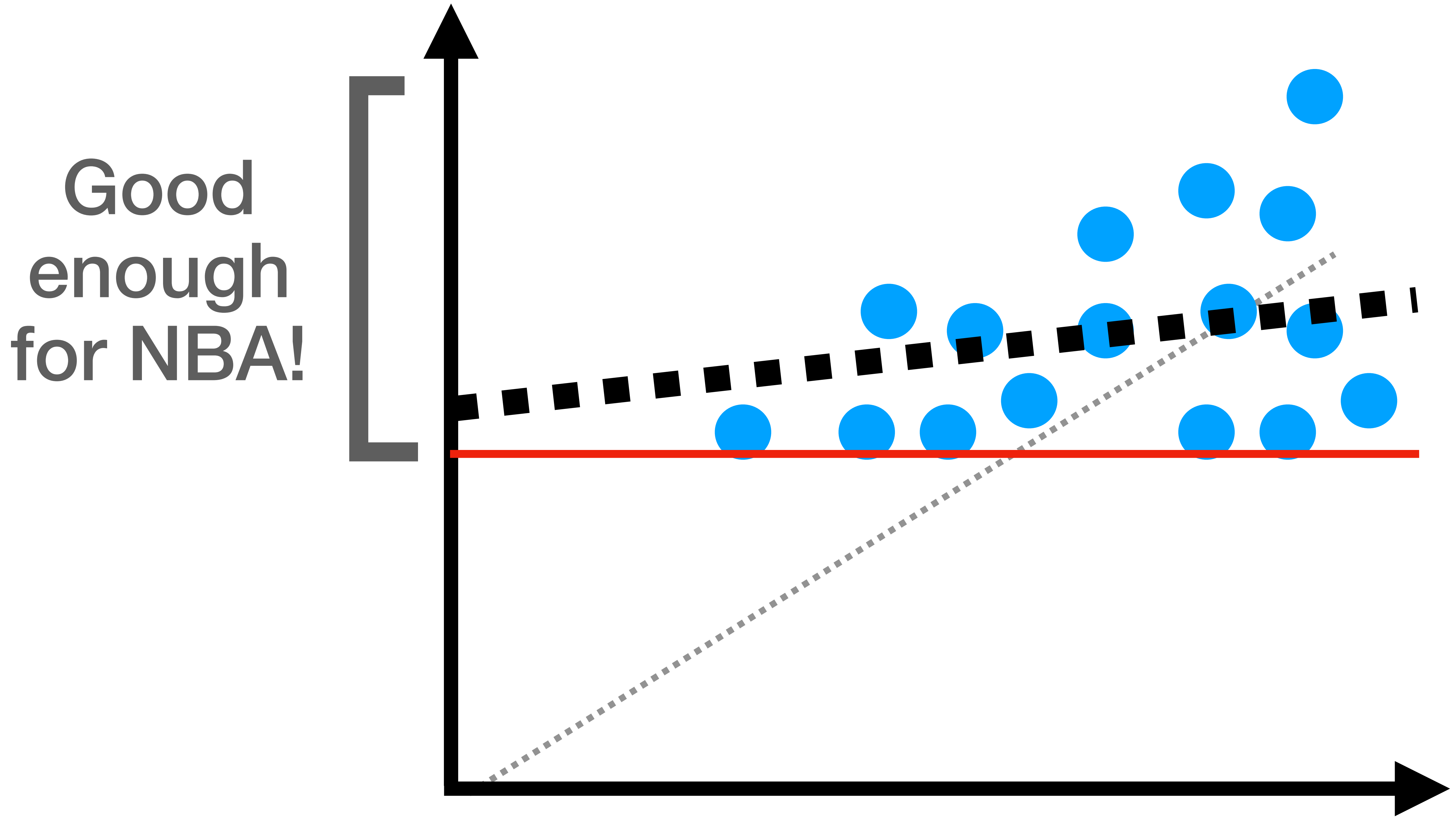
Good enough for NBA!

Bias from truncation: an illustration

- **Goal:** infer the effect of height x_i on basketball ability y_i

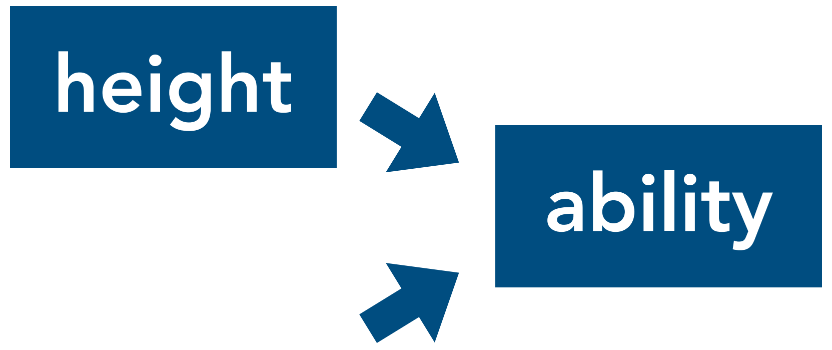
- **Strategy:** linear regression

What we get:



Bias from truncation: an illustration

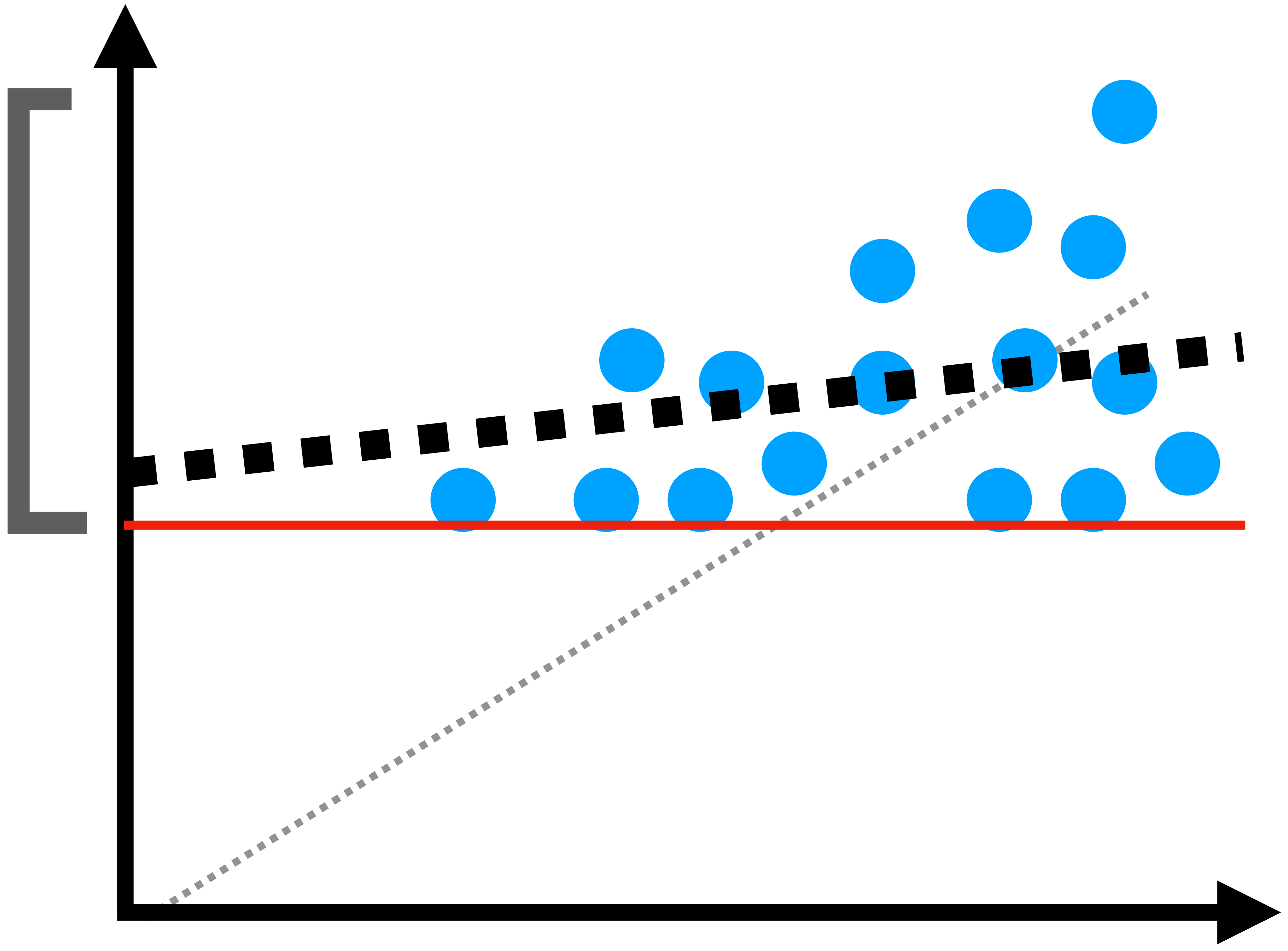
- **Goal:** infer the effect of height x_i on basketball ability y_i
- **Strategy:** linear regression


height

ability

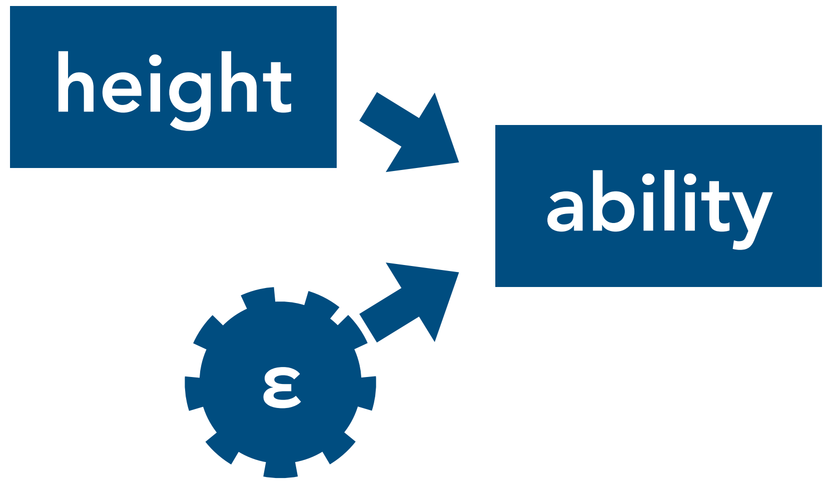
What we get:

Good enough for NBA!

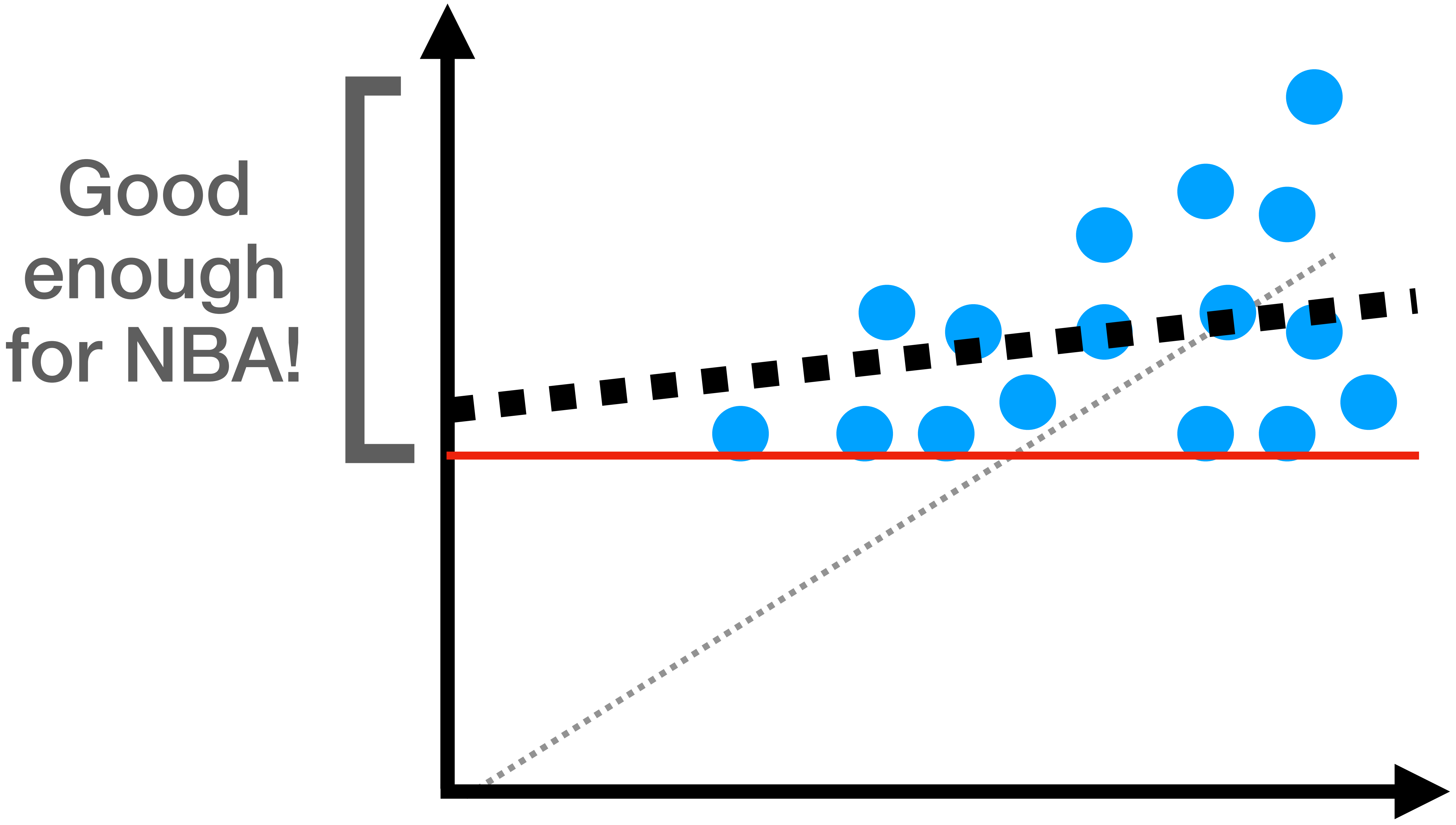


Bias from truncation: an illustration

- **Goal:** infer the effect of height x_i on basketball ability y_i
- **Strategy:** linear regression

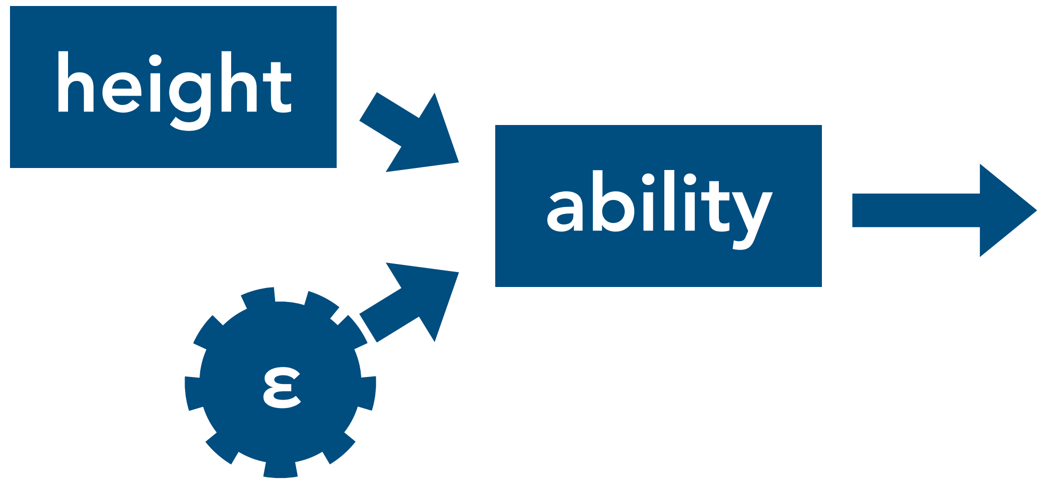


What we get:



Bias from truncation: an illustration

- **Goal:** infer the effect of height x_i on basketball ability y_i
- **Strategy:** linear regression

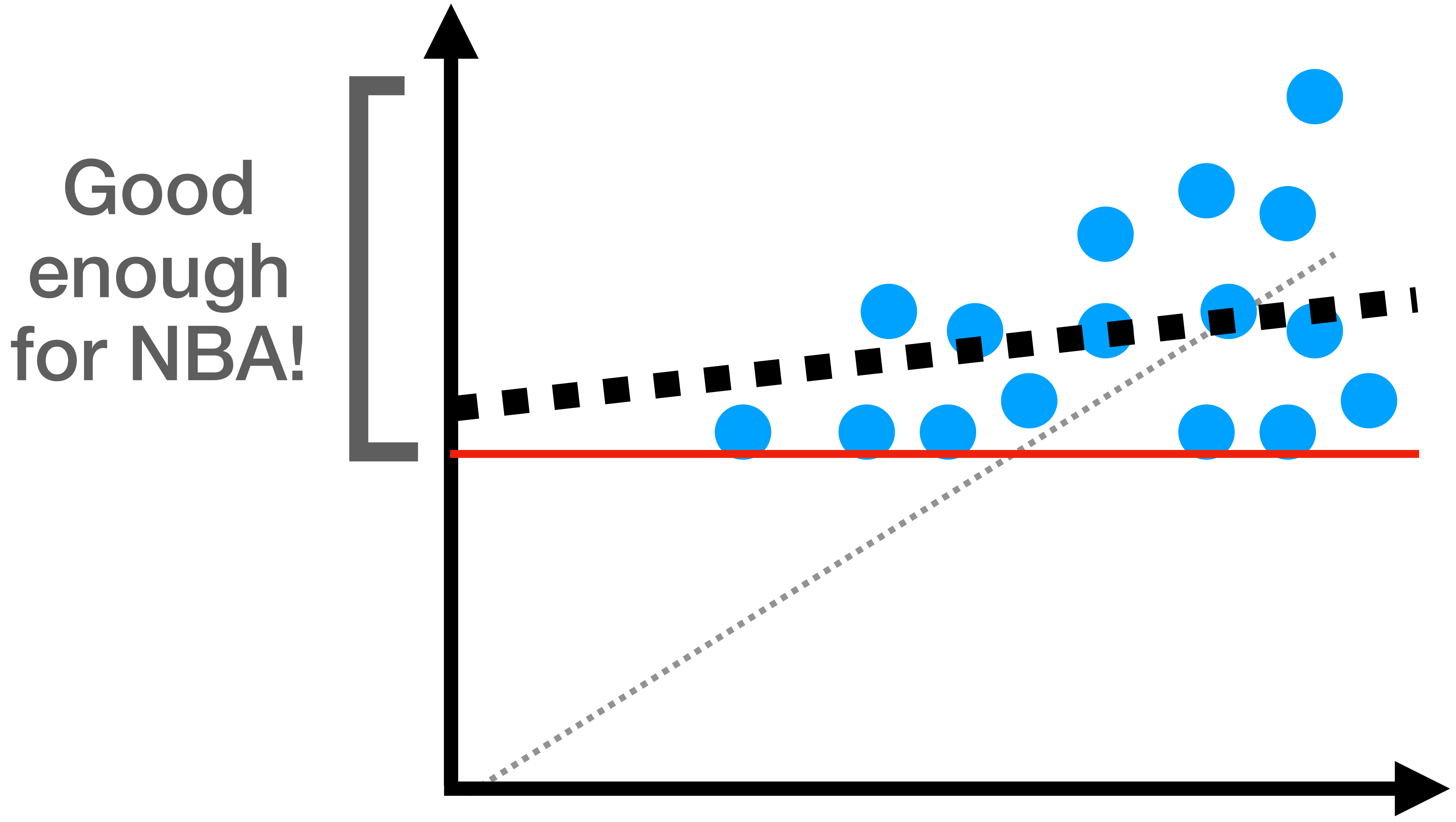


height

ability

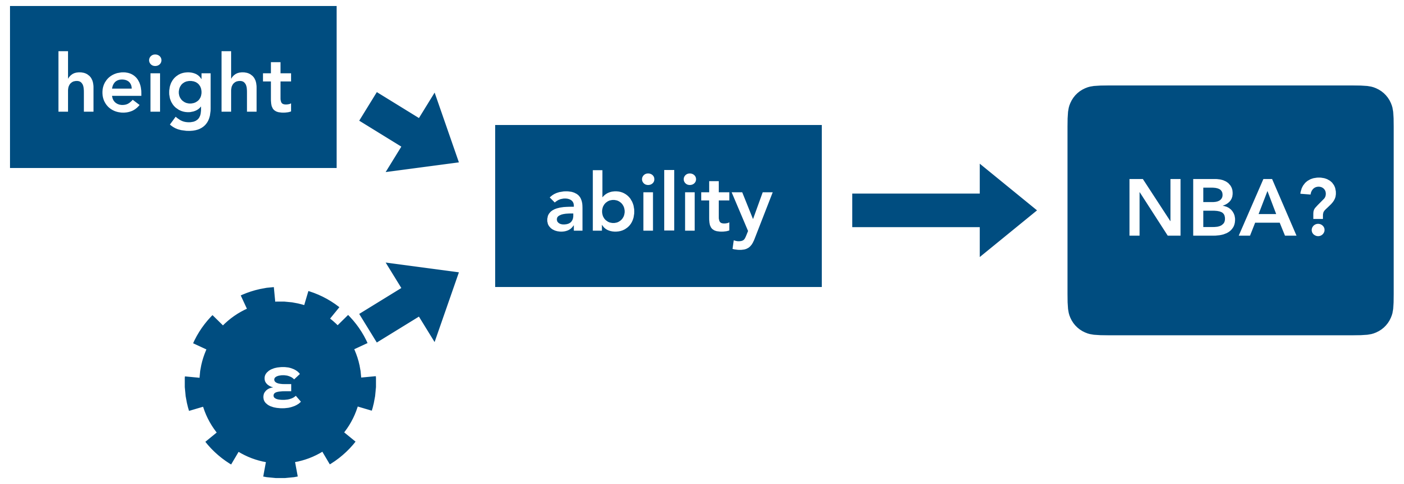
ε

What we get:

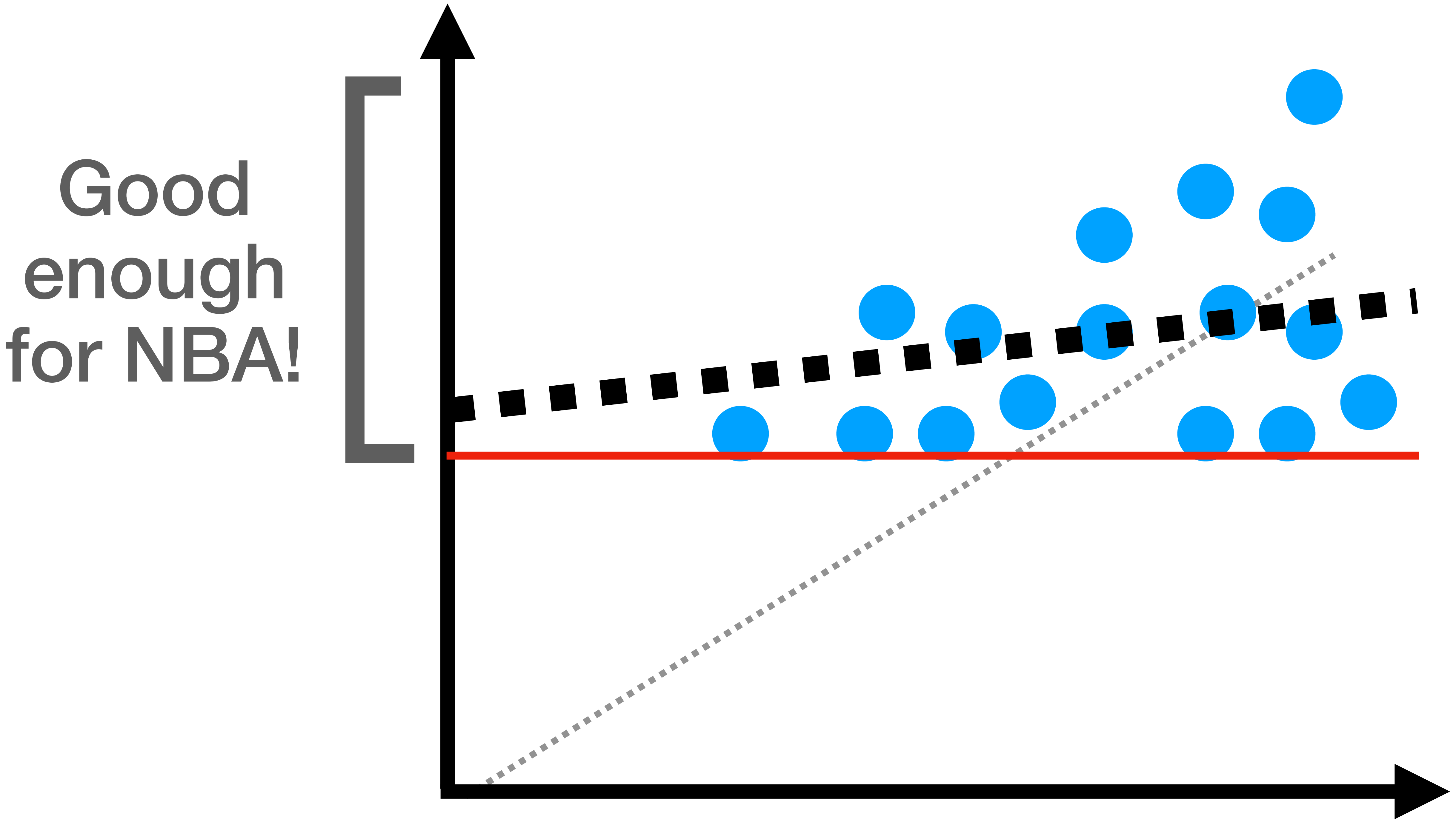


Bias from truncation: an illustration

- **Goal:** infer the effect of height x_i on basketball ability y_i
- **Strategy:** linear regression

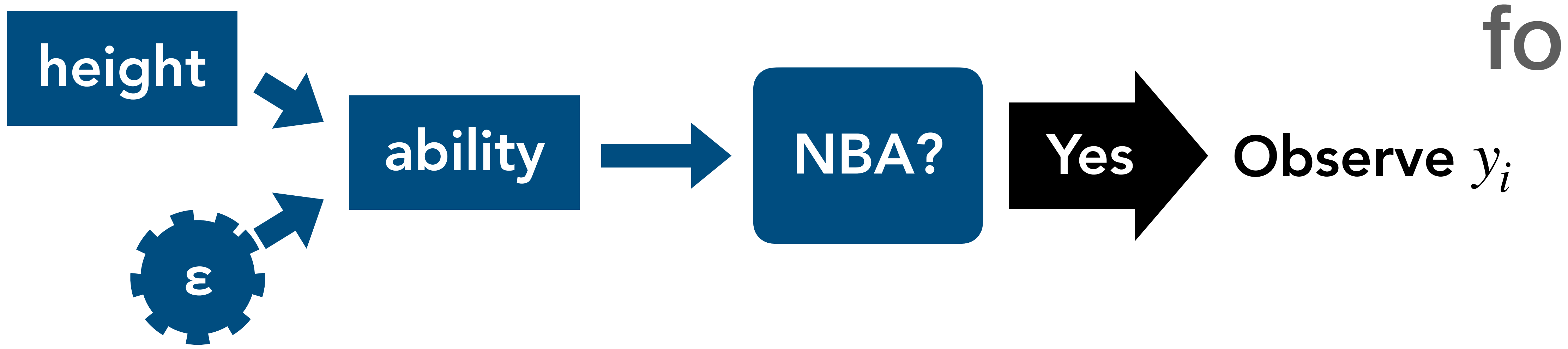


What we get:

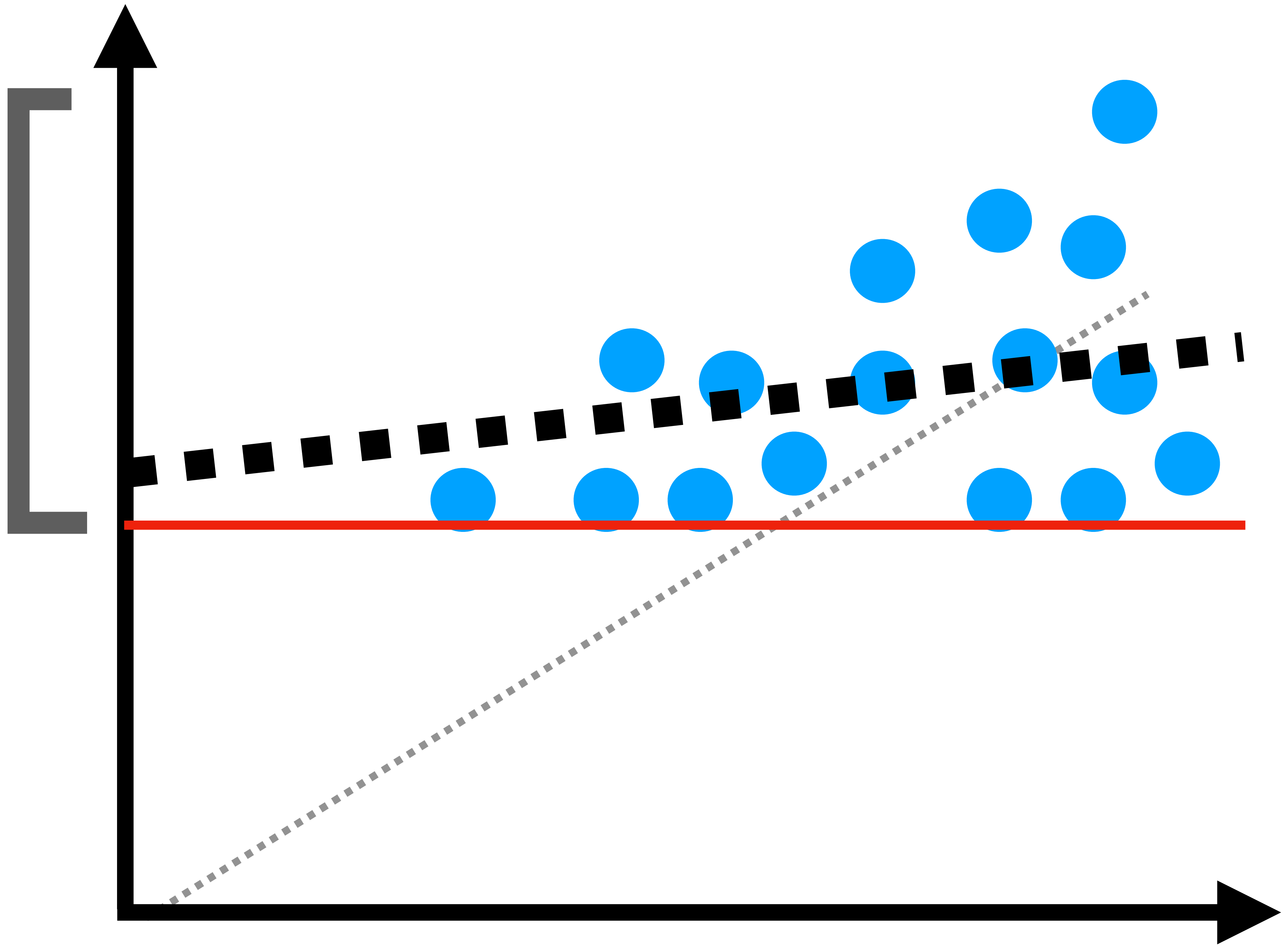


Bias from truncation: an illustration

- **Goal:** infer the effect of height x_i on basketball ability y_i
- **Strategy:** linear regression



What we get:

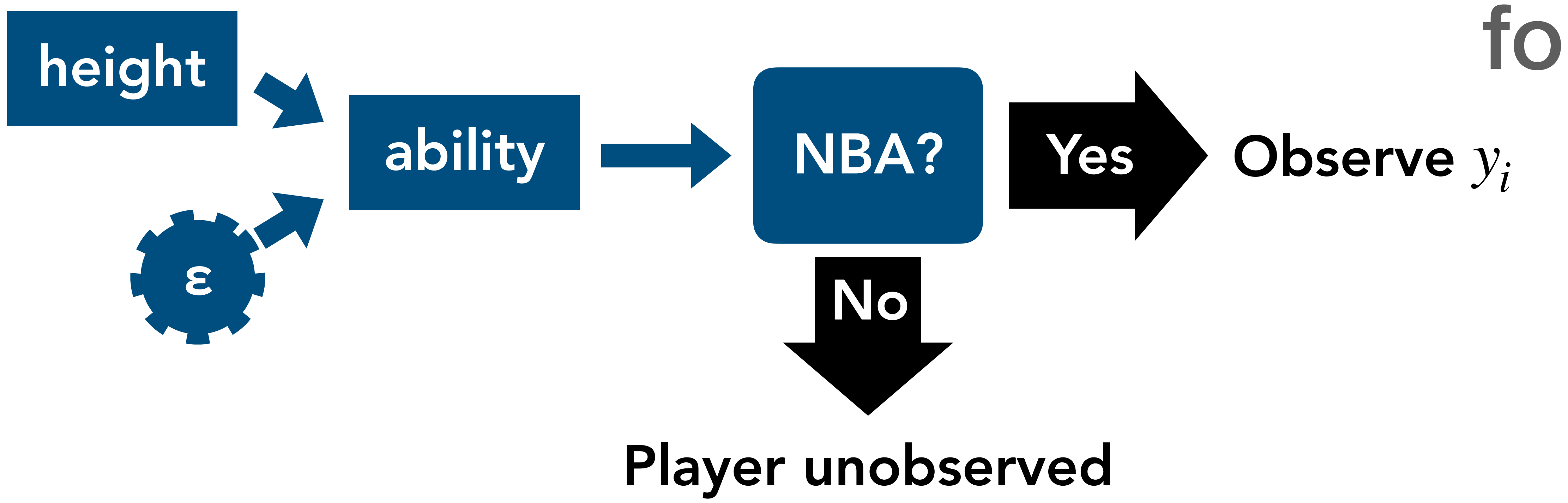


Good enough for NBA!

Bias from truncation: an illustration

- **Goal:** infer the effect of height x_i on basketball ability y_i

- **Strategy:** linear regression



height

ε

ability

NBA?

Yes

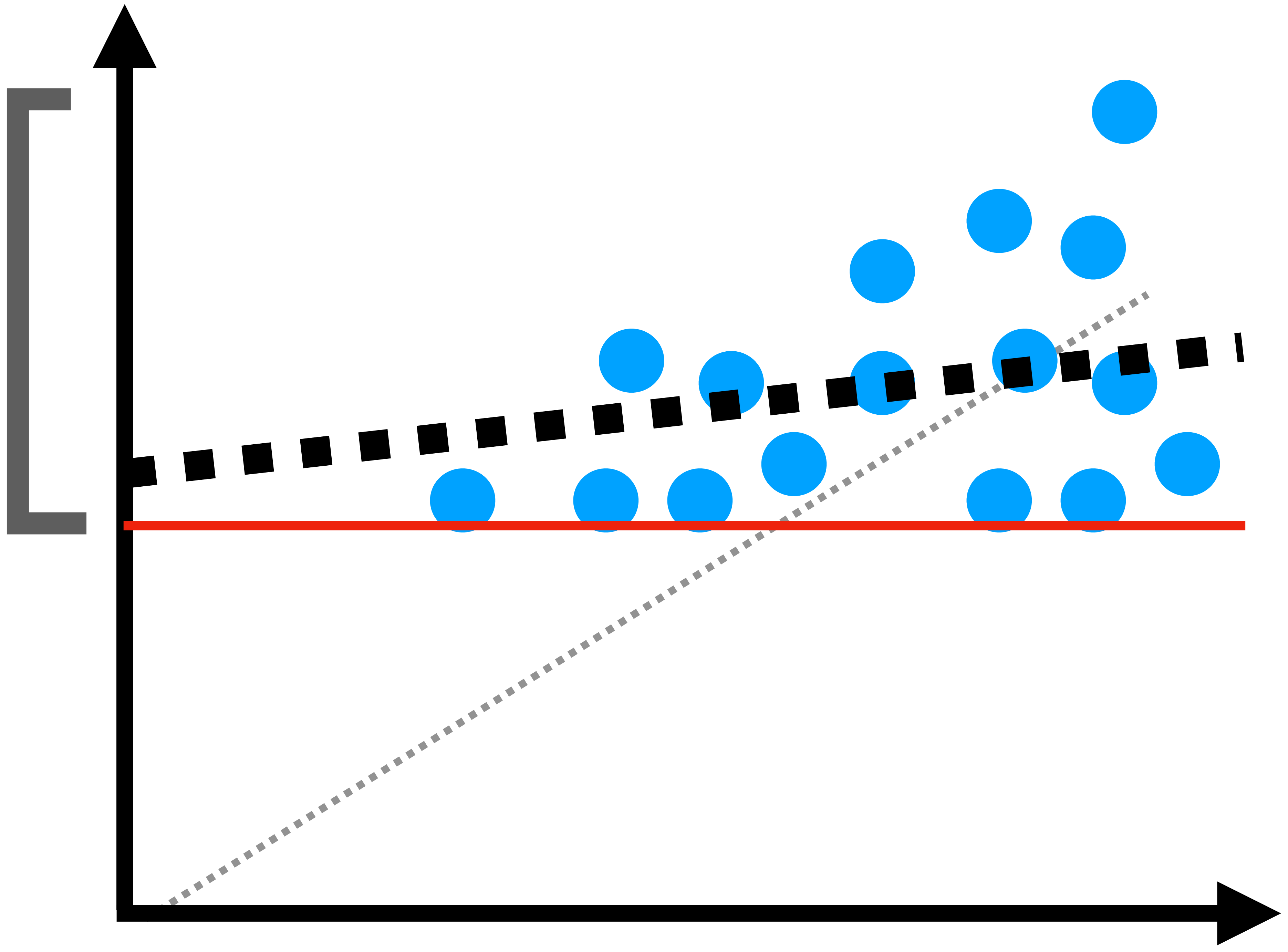
Observe y_i

No

Player unobserved

What we get:

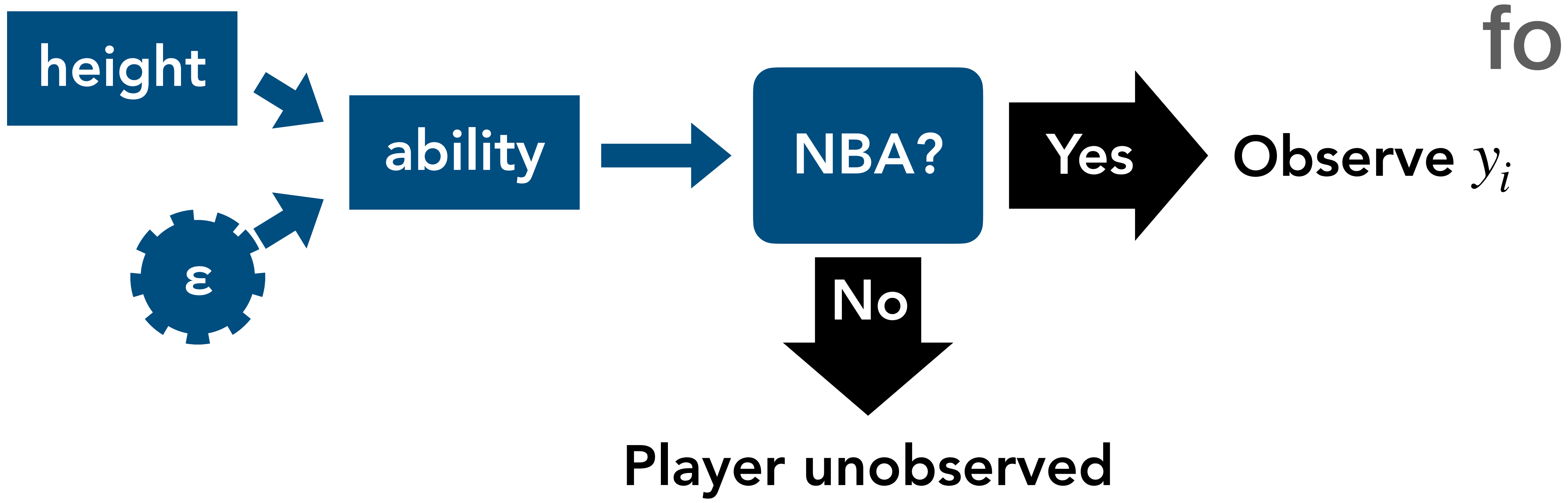
Good enough for NBA!



Bias from truncation: an illustration

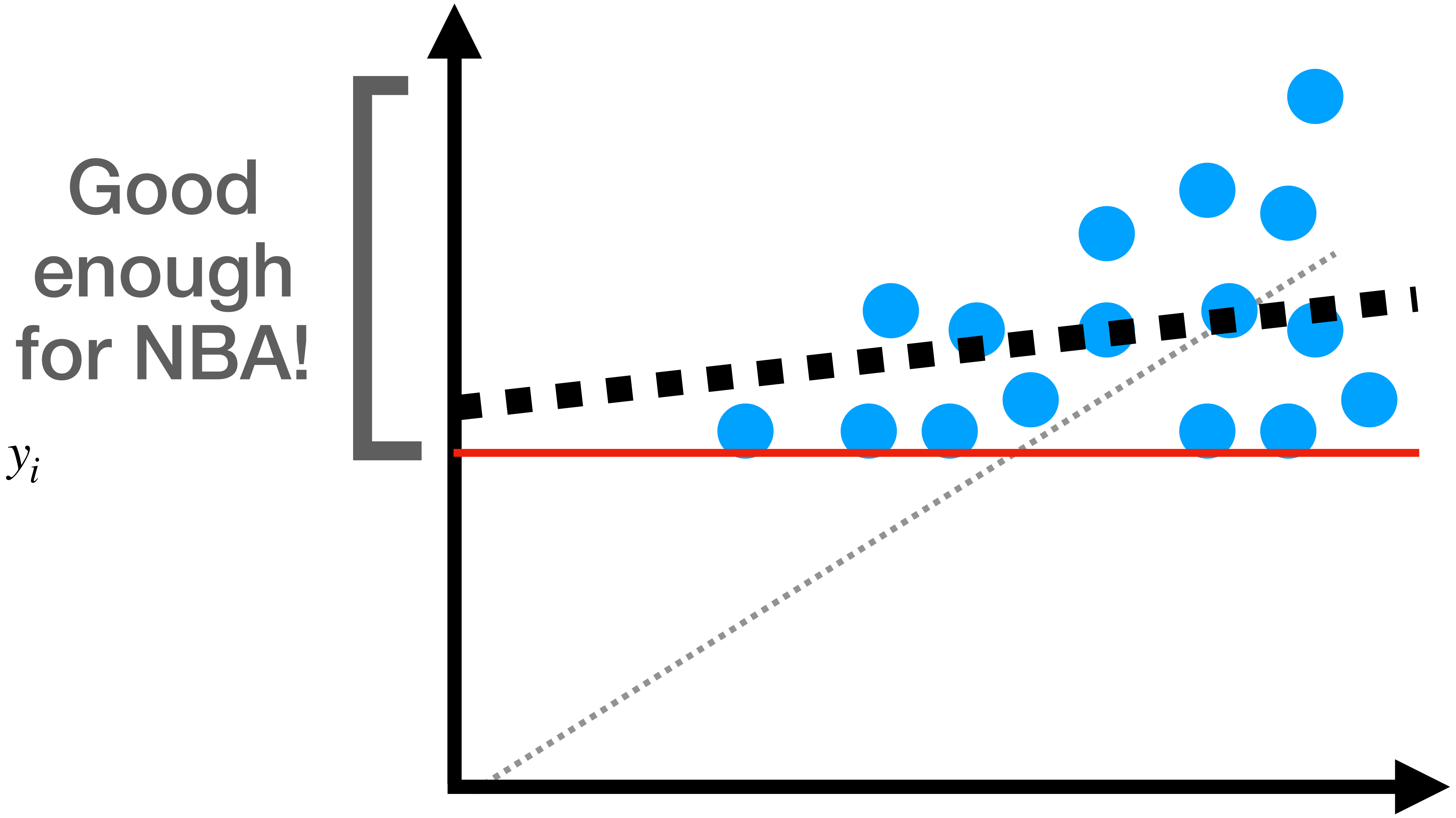
- **Goal:** infer the effect of height x_i on basketball ability y_i

- **Strategy:** linear regression



- **Truncation:** only observe data based on the value of y_i

What we get:



Truncation in practice

Not a hypothetical problem (or a new one!)

Truncation in practice

Not a hypothetical problem (or a new one!)

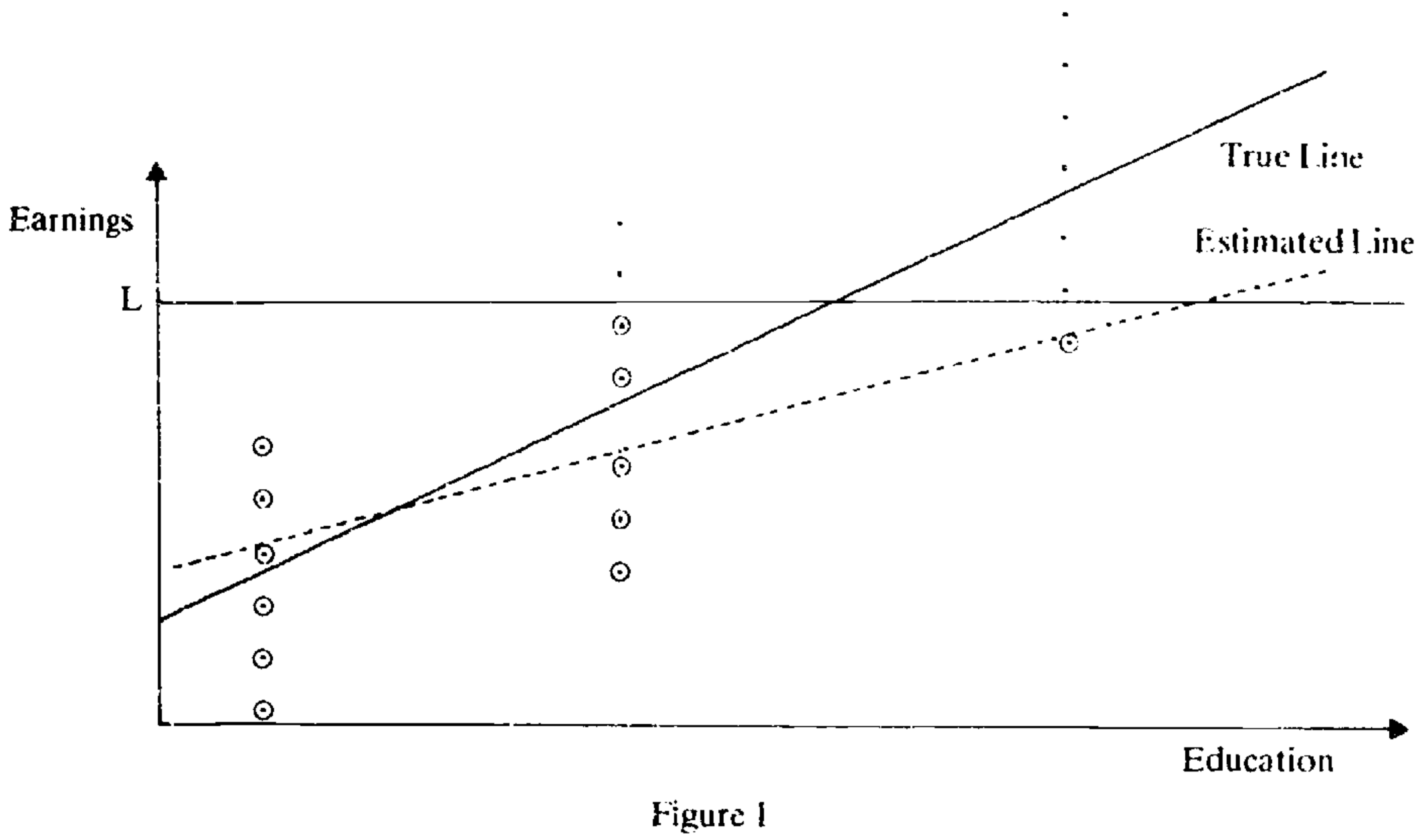


Figure 1

Fig 1 [Hausman and Wise 1977]

Truncation in practice

Not a hypothetical problem (or a new one!)

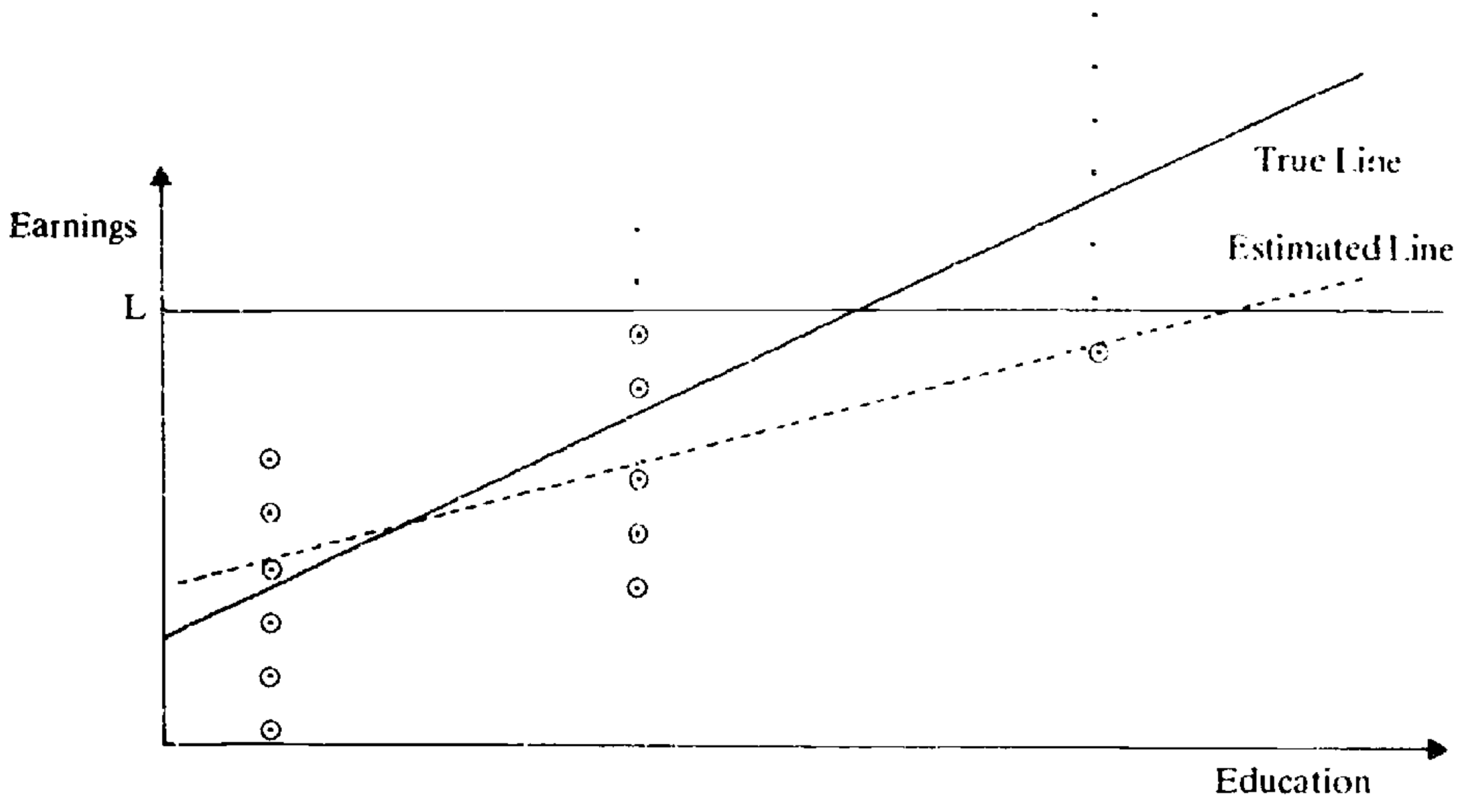


Figure 1

Fig 1 [Hausman and Wise 1977]

Corrected previous findings about education (x) vs income (y) affected by truncation on income (y)

Truncation in practice

Not a hypothetical problem (or a new one!)

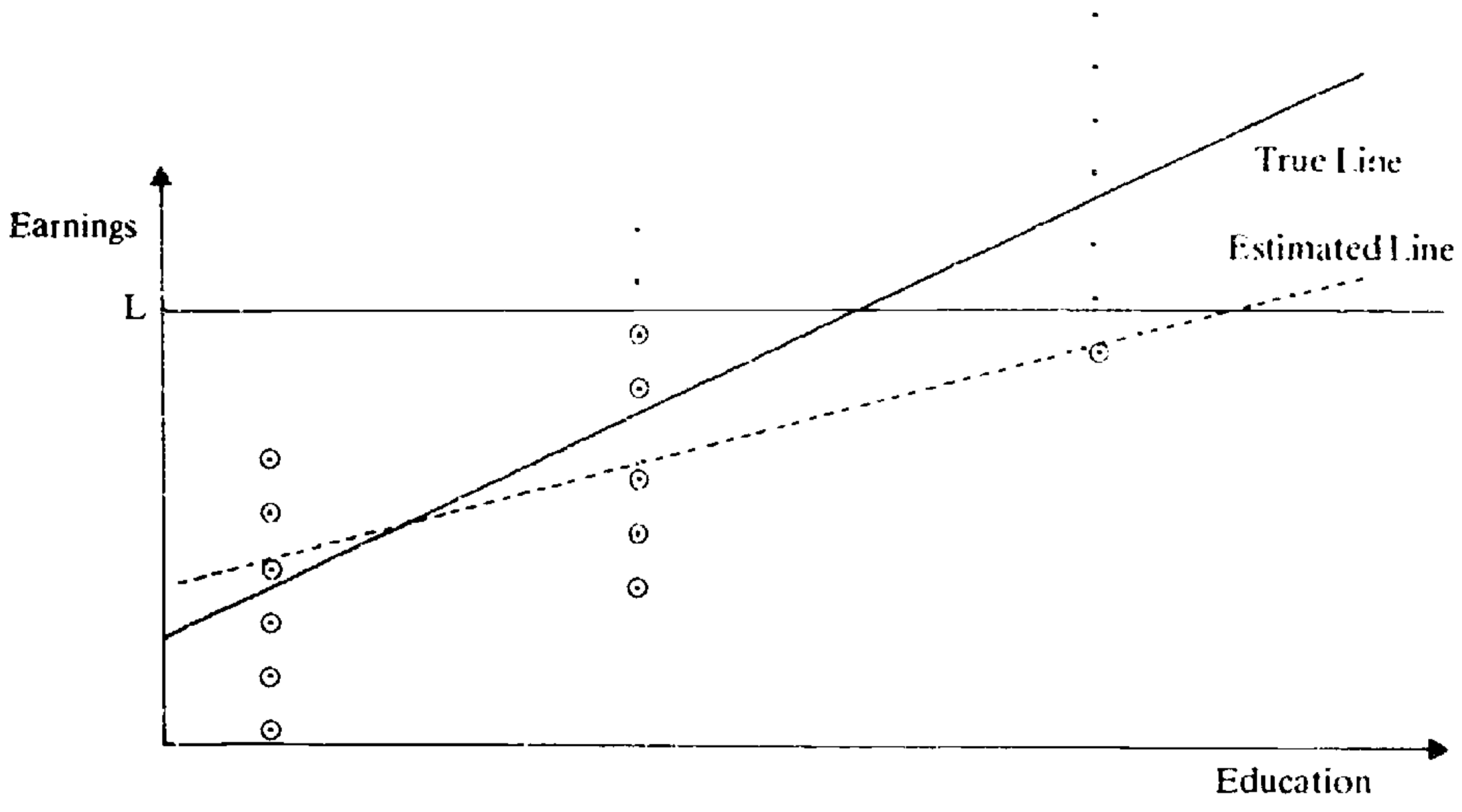
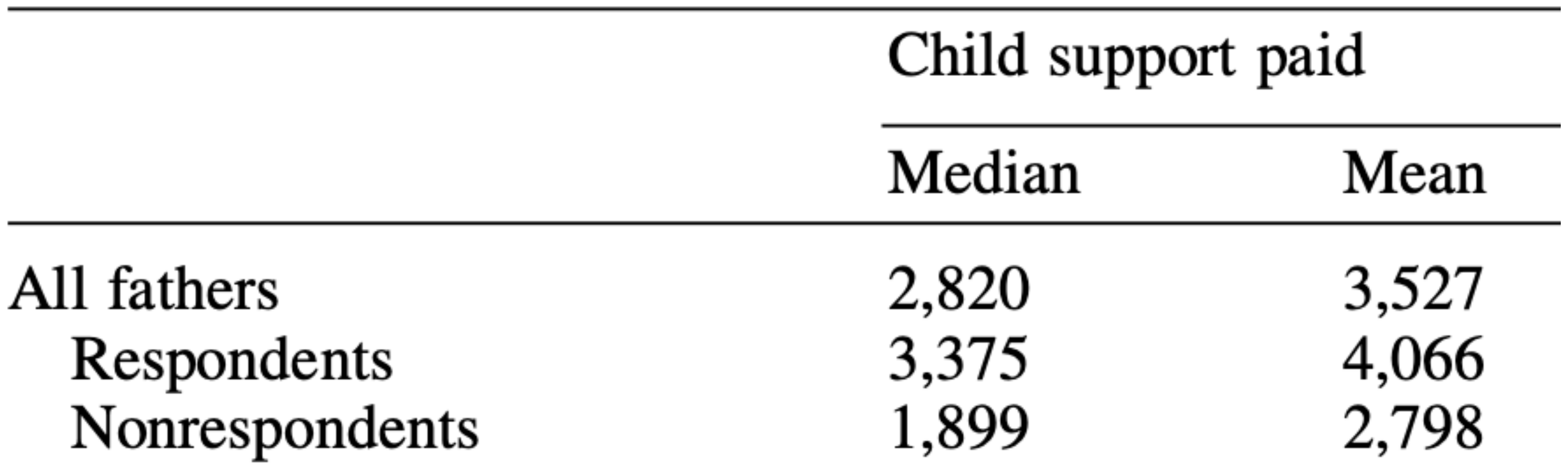


Figure 1

Fig 1 [Hausman and Wise 1977]

Corrected previous findings about education (x) vs income (y) affected by truncation on income (y)



	Child support paid	
	Median	Mean
All fathers	2,820	3,527
Respondents	3,375	4,066
Nonrespondents	1,899	2,798

Table 1 [Lin et al 1999]

Truncation in practice

Not a hypothetical problem (or a new one!)

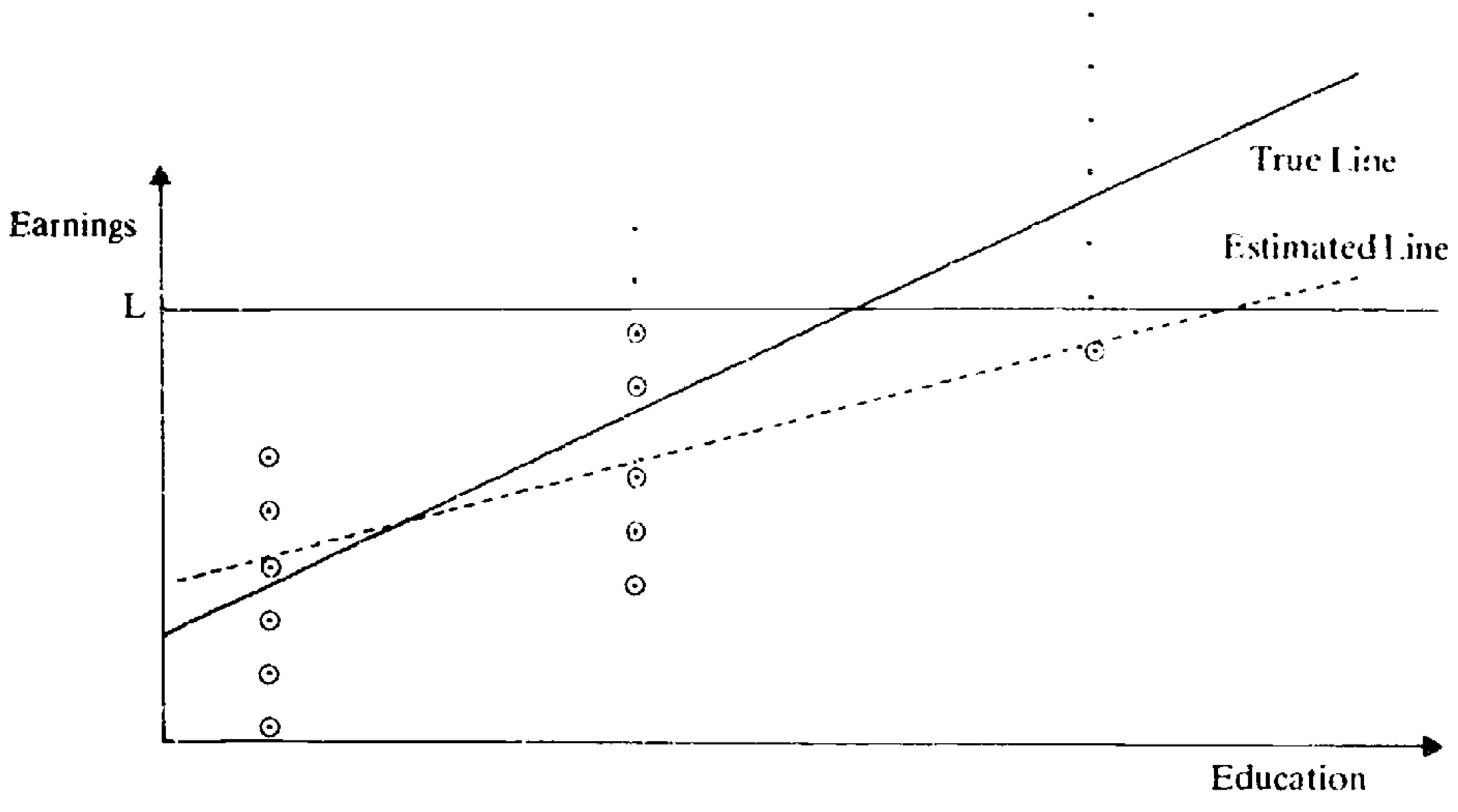
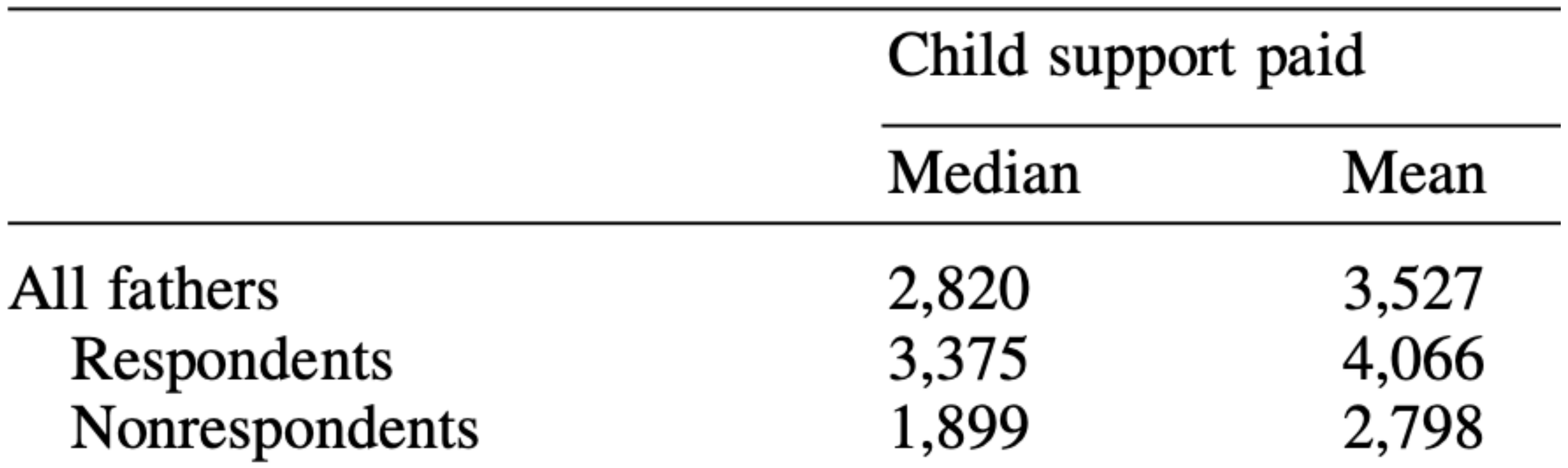


Figure 1

Fig 1 [Hausman and Wise 1977]

Corrected previous findings about education (x) vs income (y) affected by truncation on income (y)



	Child support paid	
	Median	Mean
All fathers	2,820	3,527
Respondents	3,375	4,066
Nonrespondents	1,899	2,798

Table 1 [Lin et al 1999]

Found bias in income (x) vs child support (y) because respondence rate differs based on y

Truncation in practice

Not a hypothetical problem (or a new one!)

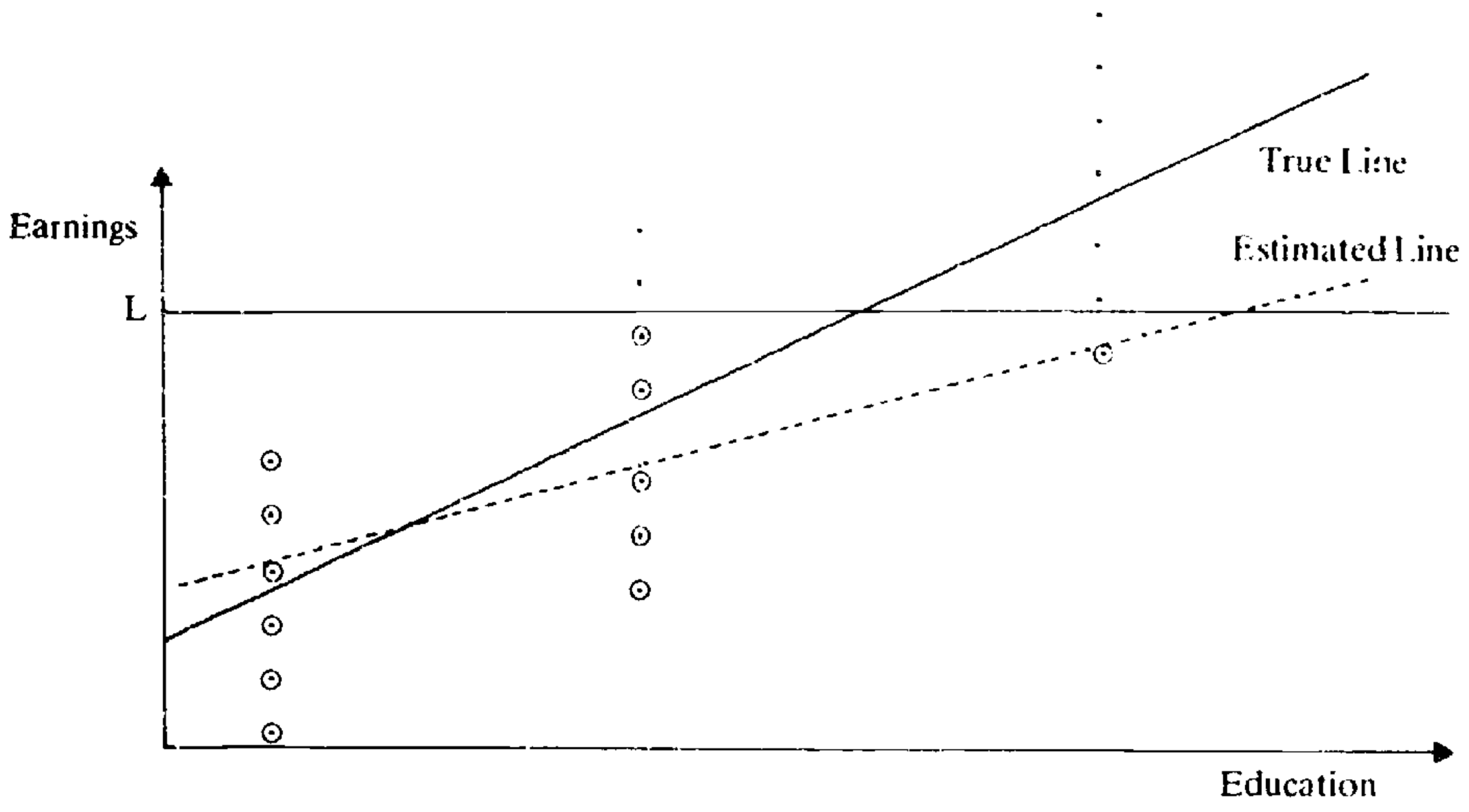
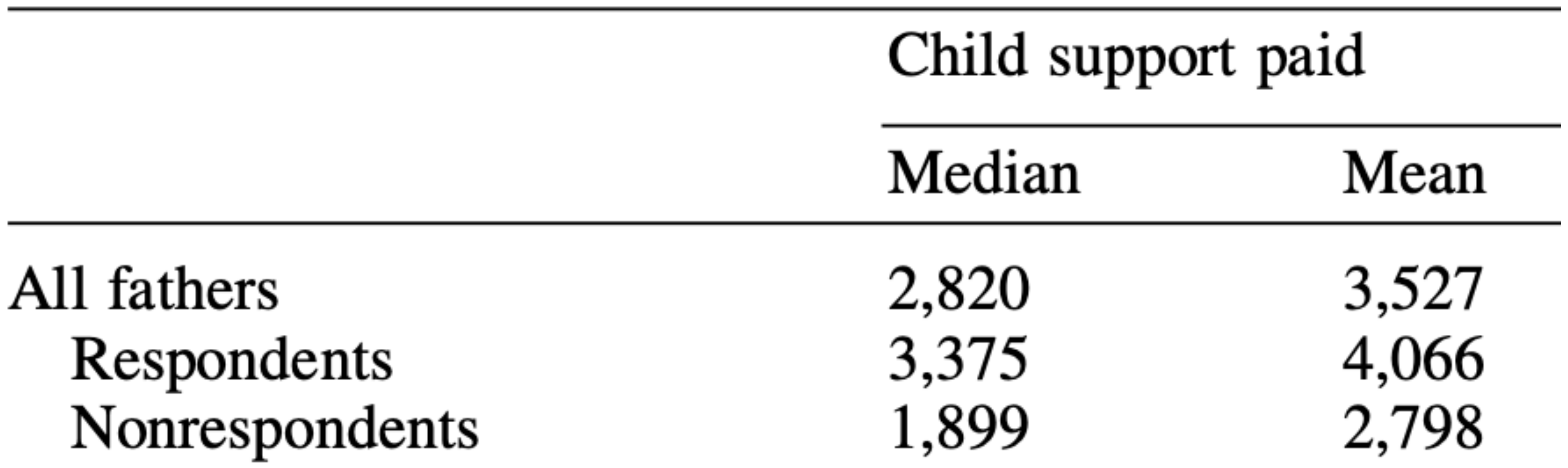


Figure I



	Child support paid	
	Median	Mean
All fathers	2,820	3,527
Respondents	3,375	4,066
Nonrespondents	1,899	2,798

Fig 1 [Hausman and Wise 1977]

Table 1 [Lin et al 1999]

Corrected previous findings about educatio

by tr

Found bias in income (x) vs child

dence

Has inspired lots of prior work in statistics/econometrics

Our goal: unified efficient (polynomial in dimension) algorithm

[Galton 1897; Pearson 1902; Lee 1914; Fisher 1931; Hotelling 1948; Tukey 1949; Tobin 1958; Amemiya 1973; Breen 1996; Balakrishnan, Cramer 2014]

Truncated regression and classification

Truncated regression and classification

Sample a
covariate *x*

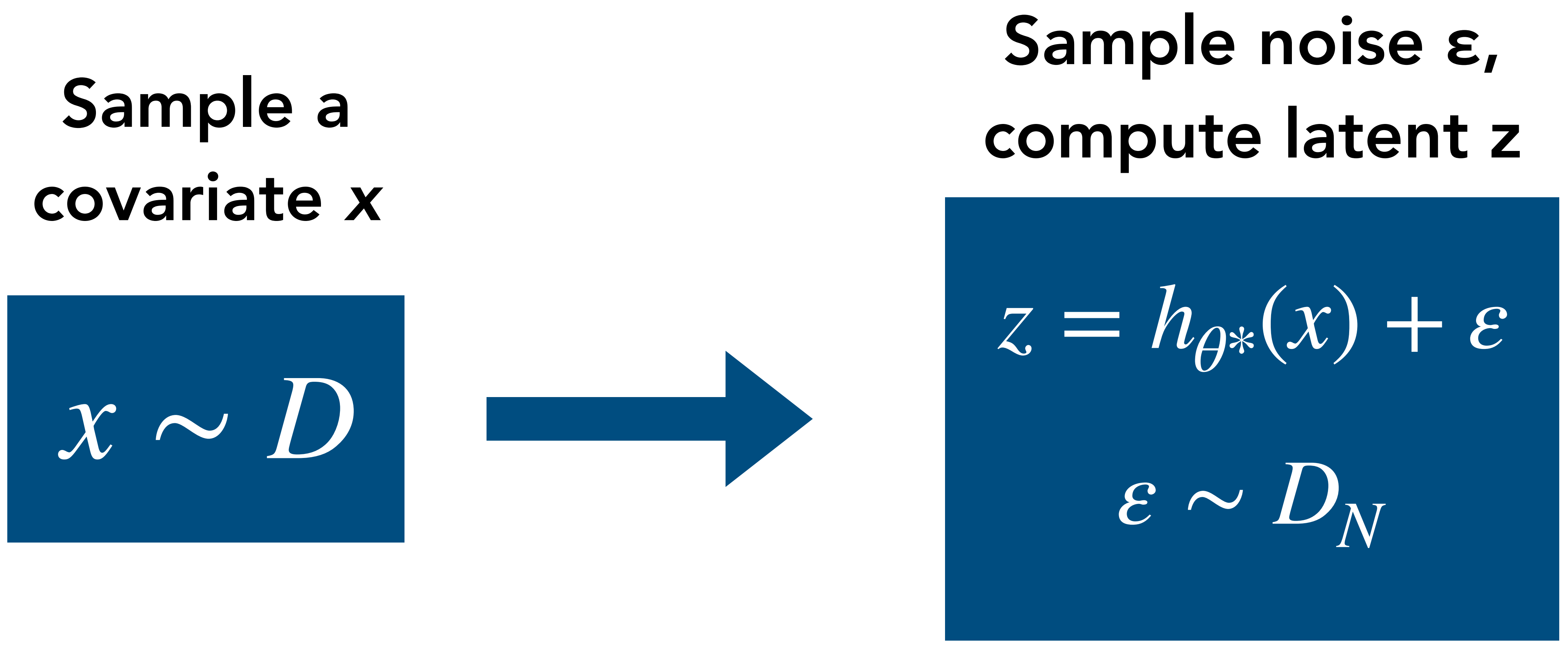
$$x \sim D$$

Truncated regression and classification

Sample a covariate *x*

$x \sim D$

Truncated regression and classification



Sample a covariate *x*

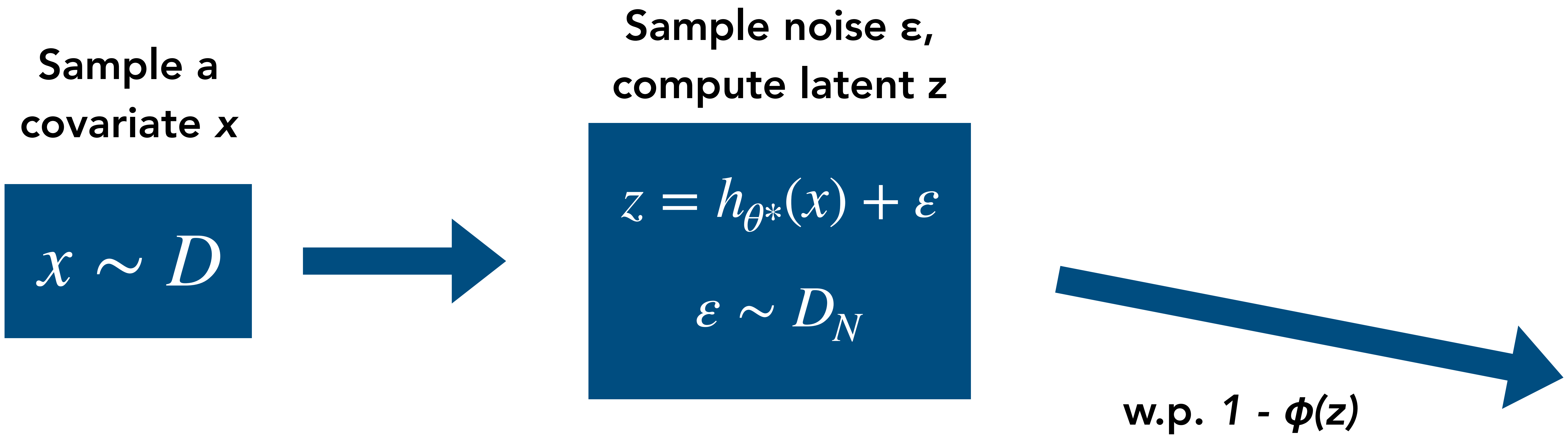
$x \sim D$

Sample noise ε, compute latent z

$z = h_{\theta*}(x) + \varepsilon$

$\varepsilon \sim D_N$

Truncated regression and classification



Sample a covariate *x*

$x \sim D$

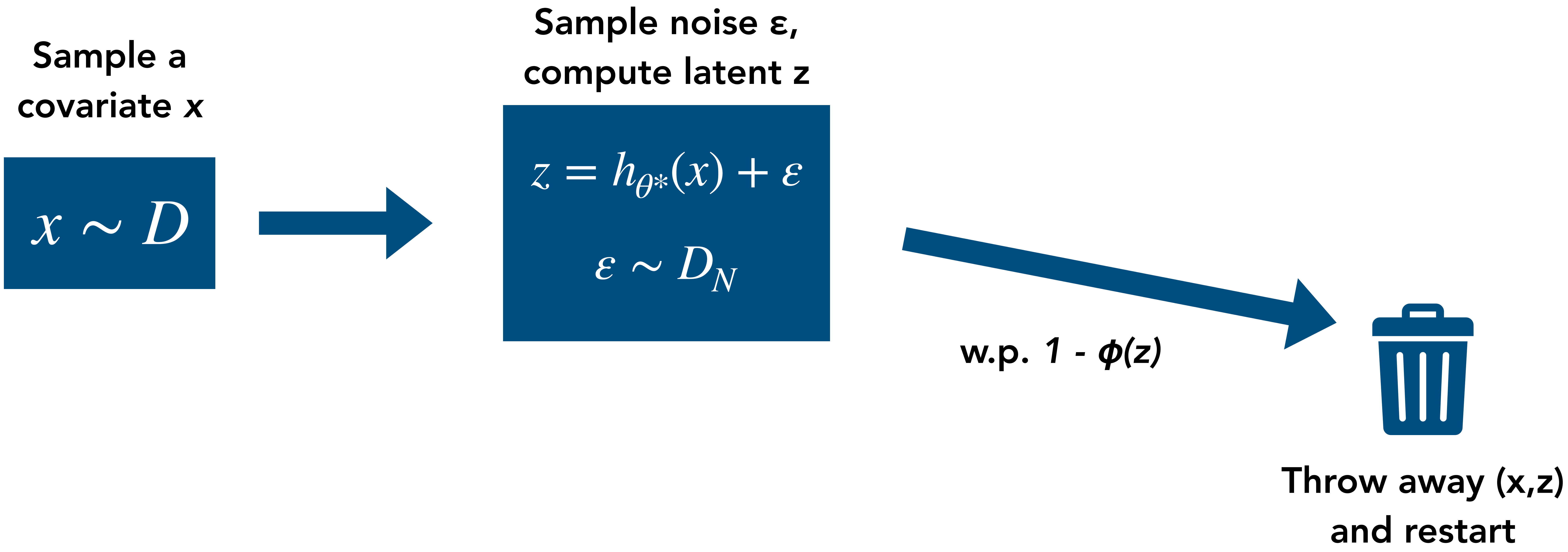
Sample noise ε, compute latent z

$z = h_{\theta*}(x) + \varepsilon$

$\varepsilon \sim D_N$

w.p. *1 - ϕ(z)*

Truncated regression and classification



Sample a covariate *x*

$$x \sim D$$

Sample noise ε, compute latent z

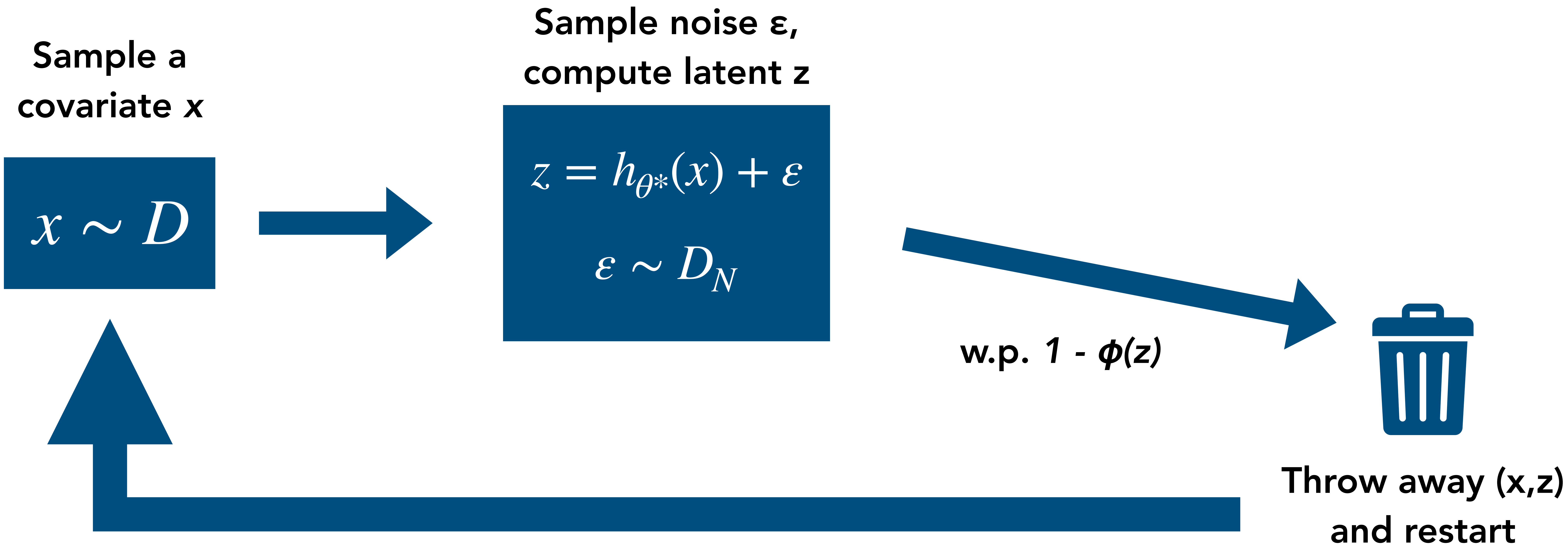
$$z = h_{\theta*}(x) + \varepsilon$$

$$\varepsilon \sim D_N$$

w.p. *1 - ϕ(z)*

Throw away (x,z) and restart

Truncated regression and classification



Sample a covariate *x*

$$x \sim D$$

Sample noise ε, compute latent z

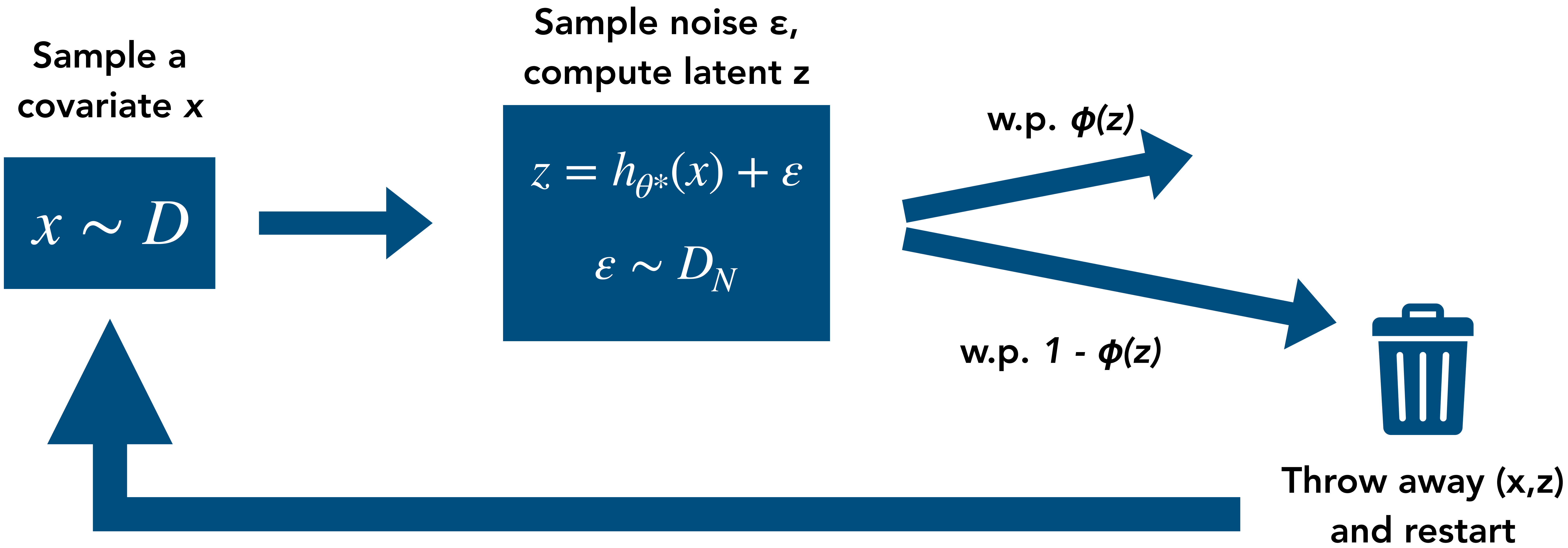
$$z = h_{\theta*}(x) + \varepsilon$$

$$\varepsilon \sim D_N$$

w.p. *1 - ϕ(z)*

Throw away (x,z) and restart

Truncated regression and classification



Sample a covariate *x*

$x \sim D$

Sample noise ε, compute latent z

$z = h_{\theta *}(x) + \varepsilon$

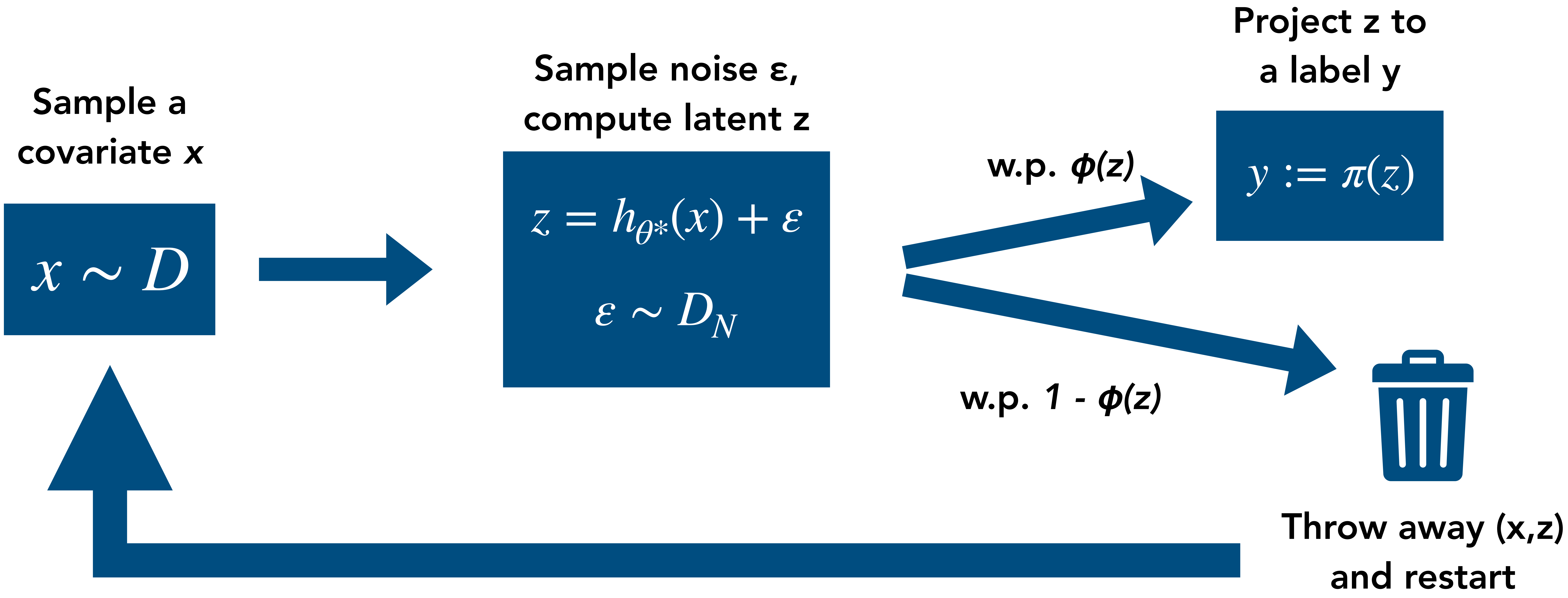
$\varepsilon \sim D_N$

w.p. *ϕ(z)*

w.p. *1 - ϕ(z)*

Throw away (x,z) and restart

Truncated regression and classification



Sample a covariate *x*

$x \sim D$

Sample noise ε, compute latent z

$z = h_{\theta*}(x) + \varepsilon$

$\varepsilon \sim D_N$

w.p. *ϕ(z)*

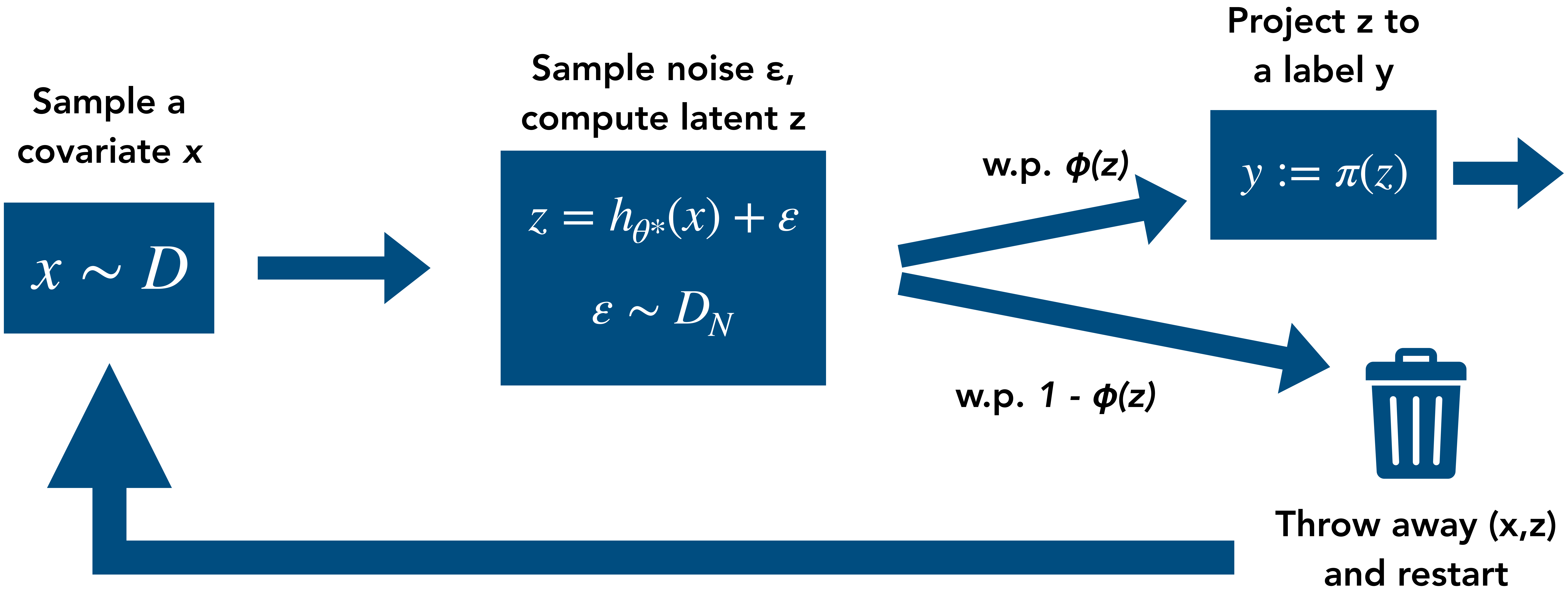
Project z to a label y

$y := \pi(z)$

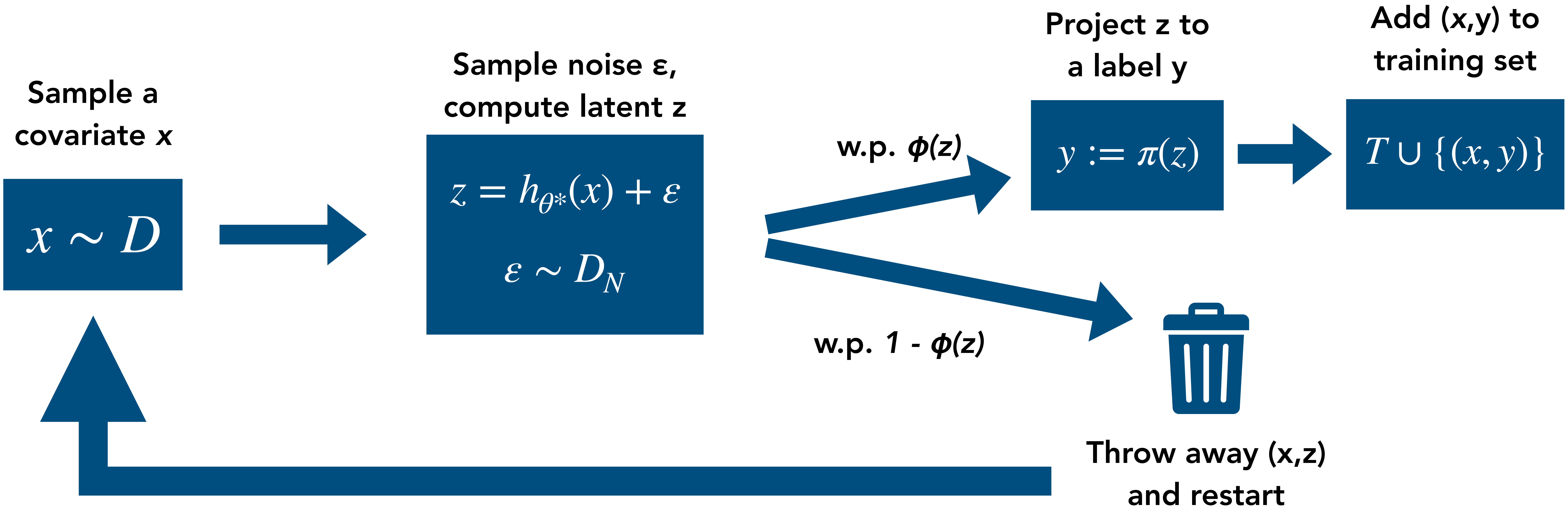
w.p. *1 - ϕ(z)*

Throw away (x,z) and restart

Truncated regression and classification



Truncated regression and classification



Sample a covariate *x*

$x \sim D$

Sample noise ε, compute latent z

$z = h_{\theta*}(x) + \varepsilon$

$\varepsilon \sim D_N$

w.p. *ϕ(z)*

Project z to a label y

$y := \pi(z)$

Add (x,y) to training set

$T \cup \{(x, y)\}$

w.p. *1 - ϕ(z)*

Throw away (x,z) and restart

Parameter estimation

Parameter estimation

- We have a model $y_i \sim \pi\left(h_{\theta*}(x_i) + \varepsilon\right)$ where $\varepsilon \sim D_N$, want estimate $\widehat{\theta}$ for $\theta*$

Parameter estimation

- We have a model $y_i \sim \pi\left(h_{\theta*}(x_i) + \varepsilon\right)$ where $\varepsilon \sim D_N$, want estimate $\widehat{\theta}$ for $\theta*$
- Standard (non-truncated) approach: **maximize likelihood**

Parameter estimation

- We have a model $y_i \sim \pi\left(h_{\theta*}(x_i) + \varepsilon\right)$ where $\varepsilon \sim D_N$, want estimate $\widehat{\theta}$ for $\theta*$
- Standard (non-truncated) approach: **maximize likelihood**

$$p(\theta; x, y) = \int_{z \in \pi^{-1}(y)} D_N(z - h_\theta(x))\ dz$$

Parameter estimation

- We have a model $y_i \sim \pi\left(h_{\theta*}(x_i) + \varepsilon\right)$ where $\varepsilon \sim D_N$, want estimate $\widehat{\theta}$ for $\theta*$
- Standard (non-truncated) approach: **maximize likelihood**

$$p(\theta; x, y) = \int_{\boxed{z \in \pi^{-1}(y)}} D_N(z - h_\theta(x))\ dz$$

All possible latent variables corresponding to label

Parameter estimation

- We have a model $y_i \sim \pi\left(h_{\theta*}(x_i) + \varepsilon\right)$ where $\varepsilon \sim D_N$, want estimate $\widehat{\theta}$ for $\theta*$
- Standard (non-truncated) approach: **maximize likelihood**

$$p(\theta; x, y) = \int_{\boxed{z\in\pi^{-1}(y)}} \boxed{D_N(z - h_\theta(x))}\, dz$$

→ Likelihood of latent under model

→ All possible latent variables corresponding to label

Parameter estimation

- We have a model $y_i \sim \pi\left(h_{\theta*}(x_i) + \varepsilon\right)$ where $\varepsilon \sim D_N$, want estimate $\widehat{\theta}$ for $\theta*$
- Standard (non-truncated) approach: **maximize likelihood**

$$p(\theta; x, y) = \int_{\boxed{z \in \pi^{-1}(y)}} \boxed{D_N(z - h_\theta(x))}\, dz$$

Likelihood of latent under model

All possible latent variables corresponding to label

- **Example:** if h_θ is a linear function, then:

Parameter estimation

- We have a model $y_i \sim \pi\left(h_{\theta*}(x_i) + \varepsilon\right)$ where $\varepsilon \sim D_N$, want estimate $\widehat{\theta}$ for $\theta*$
- Standard (non-truncated) approach: **maximize likelihood**

$$p(\theta; x, y) = \int_{\boxed{z \in \pi^{-1}(y)}} \boxed{D_N(z - h_\theta(x))}\, dz$$

Likelihood of latent under model

All possible latent variables corresponding to label

- **Example:** if h_θ is a linear function, then:
  - If $\pi(z) = z$ and $\varepsilon \sim \mathcal{N}(0,1)$, MLE is ordinary least squares regression

Parameter estimation

- We have a model $y_i \sim \pi\left(h_{\theta*}(x_i) + \varepsilon\right)$ where $\varepsilon \sim D_N$, want estimate $\widehat{\theta}$ for $\theta*$
- Standard (non-truncated) approach: **maximize likelihood**

$$p(\theta; x, y) = \int_{\boxed{z \in \pi^{-1}(y)}} \boxed{D_N(z - h_\theta(x))}\ dz$$

→ Likelihood of latent under model

→ All possible latent variables corresponding to label

- **Example:** if h_θ is a linear function, then:
  - If $\pi(z) = z$ and $\varepsilon \sim \mathcal{N}(0,1)$, MLE is ordinary least squares regression
  - If $\pi(z) = \mathbf{1}_{z \geq 0}$ and $\varepsilon \sim \mathcal{N}(0,1)$, MLE is probit regression

Parameter estimation

- We have a model $y_i \sim \pi\left(h_{\theta*}(x_i) + \varepsilon\right)$ where $\varepsilon \sim D_N$, want estimate $\widehat{\theta}$ for $\theta*$
- Standard (non-truncated) approach: **maximize likelihood**

$$p(\theta; x, y) = \int_{\boxed{z\in\pi^{-1}(y)}} \boxed{D_N(z - h_\theta(x))}\ dz$$

→ Likelihood of latent under model

→ All possible latent variables corresponding to label

- **Example:** if h_θ is a linear function, then:
  - If $\pi(z) = z$ and $\varepsilon \sim \mathcal{N}(0,1)$, MLE is ordinary least squares regression
  - If $\pi(z) = \mathbf{1}_{z\geq 0}$ and $\varepsilon \sim \mathcal{N}(0,1)$, MLE is probit regression
  - If $\pi(z) = \mathbf{1}_{z\geq 0}$ and $\varepsilon \sim$ Logistic$(0,1)$, MLE is logistic regression

Parameter estimation

- We have a model $y_i \sim \pi\left(h_{\theta*}(x_i) + \varepsilon\right)$ where $\varepsilon \sim D_N$, want estimate $\widehat{\theta}$ for $\theta*$
- Standard (non-truncated) approach: **maximize likelihood**

$$p(\theta; x, y) = \int_{\boxed{z \in \pi^{-1}(y)}} \boxed{D_N(z - h_\theta(x))}\, dz$$

→ Likelihood of latent under model

→ All possible latent variables corresponding to label

- **Example:** if h_θ is a linear function, then:
  - If $\pi(z) = z$ and $\varepsilon \sim \mathcal{N}(0,1)$, MLE is ordinary least squares regression
  - If $\pi(z) = \mathbf{1}_{z \geq 0}$ and $\varepsilon \sim \mathcal{N}(0,1)$, MLE is probit regression
  - If $\pi(z) = \mathbf{1}_{z \geq 0}$ and $\varepsilon \sim$ Logistic$(0,1)$, MLE is logistic regression
- What about the truncated case?

Parameter estimation from truncated data

Parameter estimation from truncated data

Main idea: maximization of the *truncated log-likelihood*

Parameter estimation from truncated data

Main idea: maximization of the *truncated log-likelihood*

- Truncated likelihood:

Parameter estimation from truncated data

Main idea: maximization of the *truncated log-likelihood*

- Truncated likelihood:

$$p(\theta; x, y) = \int_{z \in \pi^{-1}(y)} D_N(z - h_\theta(x)) \; dz$$

Parameter estimation from truncated data

Main idea: maximization of the *truncated log-likelihood*

- Truncated likelihood:

$$p(\theta; x, y) = \int_{z \in \pi^{-1}(y)} D_N(z - h_\theta(x))\ dz$$

➡

Parameter estimation from truncated data

Main idea: maximization of the *truncated log-likelihood*

- Truncated likelihood:

$$p(\theta; x, y) = \int_{z \in \pi^{-1}(y)} D_N(z - h_\theta(x)) \ dz \quad \Rightarrow \quad p(\theta; x, y) = \frac{\int_{z \in \pi^{-1}(y)} D_N(z - h_\theta(x)) \phi(z) \ dz}{\int_z D_N(z - h_\theta(x)) \phi(z) \ dz}$$

Parameter estimation from truncated data

Main idea: maximization of the *truncated log-likelihood*

- Truncated likelihood:

$$p(\theta; x, y) = \int_{z \in \pi^{-1}(y)} D_N(z - h_\theta(x)) \ dz$$

→

$$p(\theta; x, y) = \frac{\int_{z \in \pi^{-1}(y)} D_N(z - h_\theta(x))\phi(z) \ dz}{\int_z D_N(z - h_\theta(x))\phi(z) \ dz}$$

Parameter estimation from truncated data

Main idea: maximization of the *truncated log-likelihood*

- Truncated likelihood:

$$p(\theta; x, y) = \int_{z \in \pi^{-1}(y)} D_N(z - h_\theta(x)) \; dz \quad \Rightarrow \quad p(\theta; x, y) = \frac{\int_{z \in \pi^{-1}(y)} D_N(z - h_\theta(x))\phi(z) \; dz}{\int_z D_N(z - h_\theta(x))\phi(z) \; dz}$$

Parameter estimation from truncated data

Main idea: maximization of the *truncated log-likelihood*

- Truncated likelihood:

$$p(\theta; x, y) = \int_{z \in \pi^{-1}(y)} D_N(z - h_\theta(x))\ dz$$

➡

$$p(\theta; x, y) = \frac{\int_{z \in \pi^{-1}(y)} D_N(z - h_\theta(x))\phi(z)\ dz}{\int_{z} D_N(z - h_\theta(x))\phi(z)\ dz}$$

- **Again,** we can compute a stochastic gradient of the log-likelihood with only oracle access to $\phi \implies$ Leads to another SGD-based algorithm

Parameter estimation from truncated data

Main idea: maximization of the *truncated log-likelihood*

- Truncated likelihood:

$$p(\theta; x, y) = \int_{z \in \pi^{-1}(y)} D_N(z - h_\theta(x)) \ dz$$

→

$$p(\theta; x, y) = \frac{\int_{z \in \pi^{-1}(y)} D_N(z - h_\theta(x))\phi(z) \ dz}{\int_z D_N(z - h_\theta(x))\phi(z) \ dz}$$

- **Again,** we can compute a stochastic gradient of the log-likelihood with only oracle access to $\phi \implies$ Leads to another SGD-based algorithm

- **However:** this time the loss can actually be *non-convex*

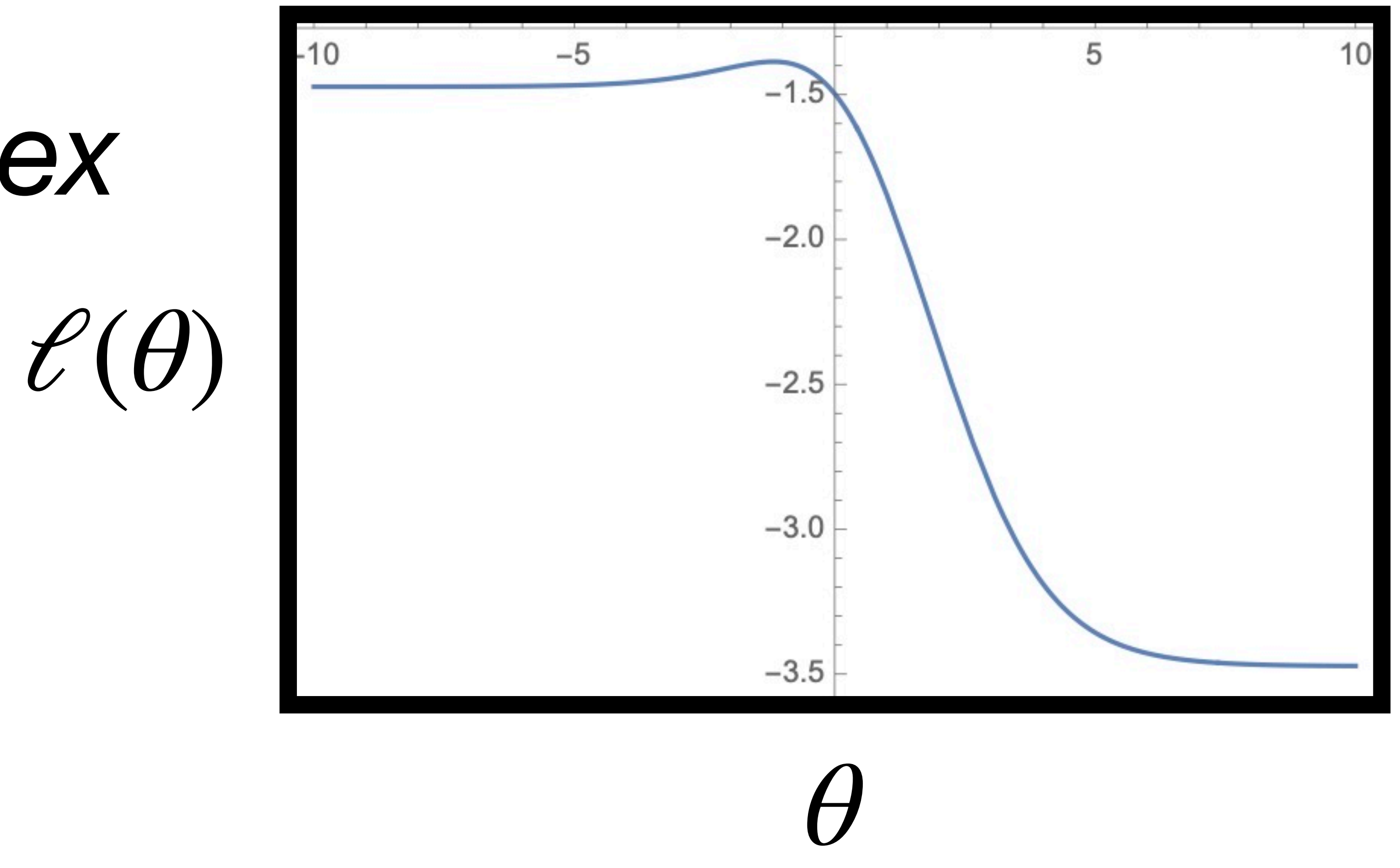
Parameter estimation from truncated data

Parameter estimation from truncated data

- **However:** this time the loss can actually be *non-convex*

Parameter estimation from truncated data

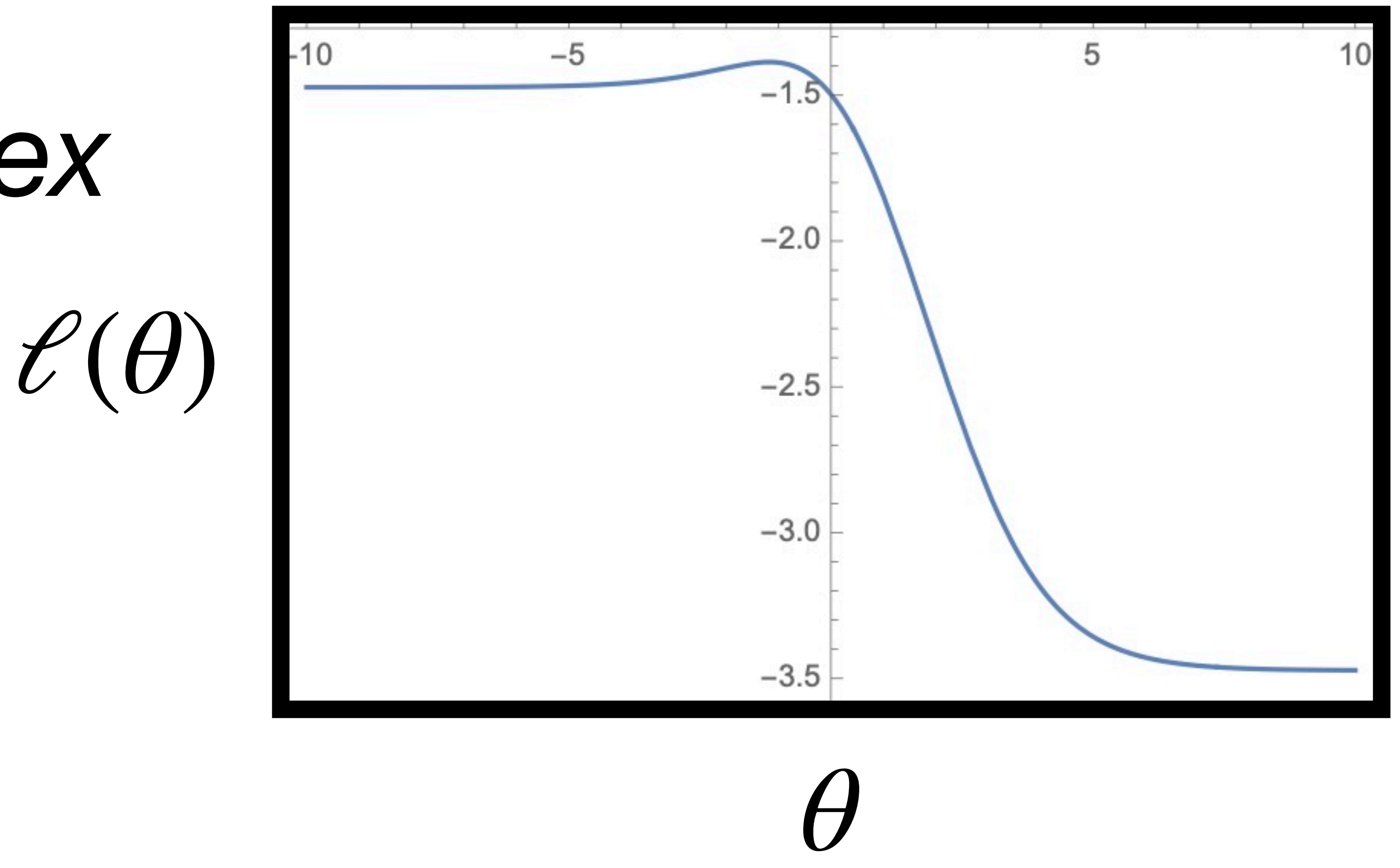
- **However:** this time the loss can actually be *non-convex*



Parameter estimation from truncated data

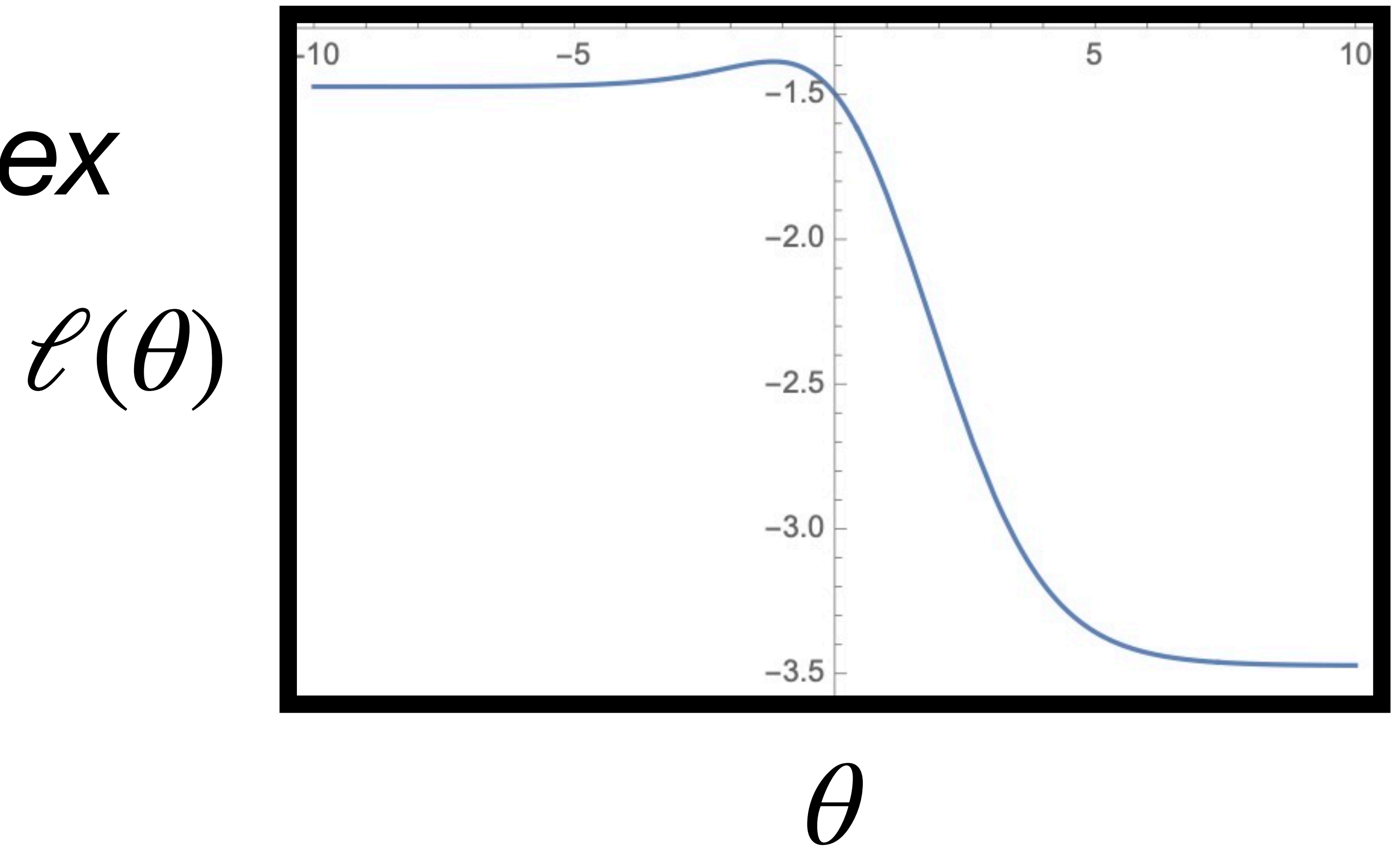
- **However:** this time the loss can actually be *non-convex*

- Example: 1D logistic regression, $S = [-1, 3]$



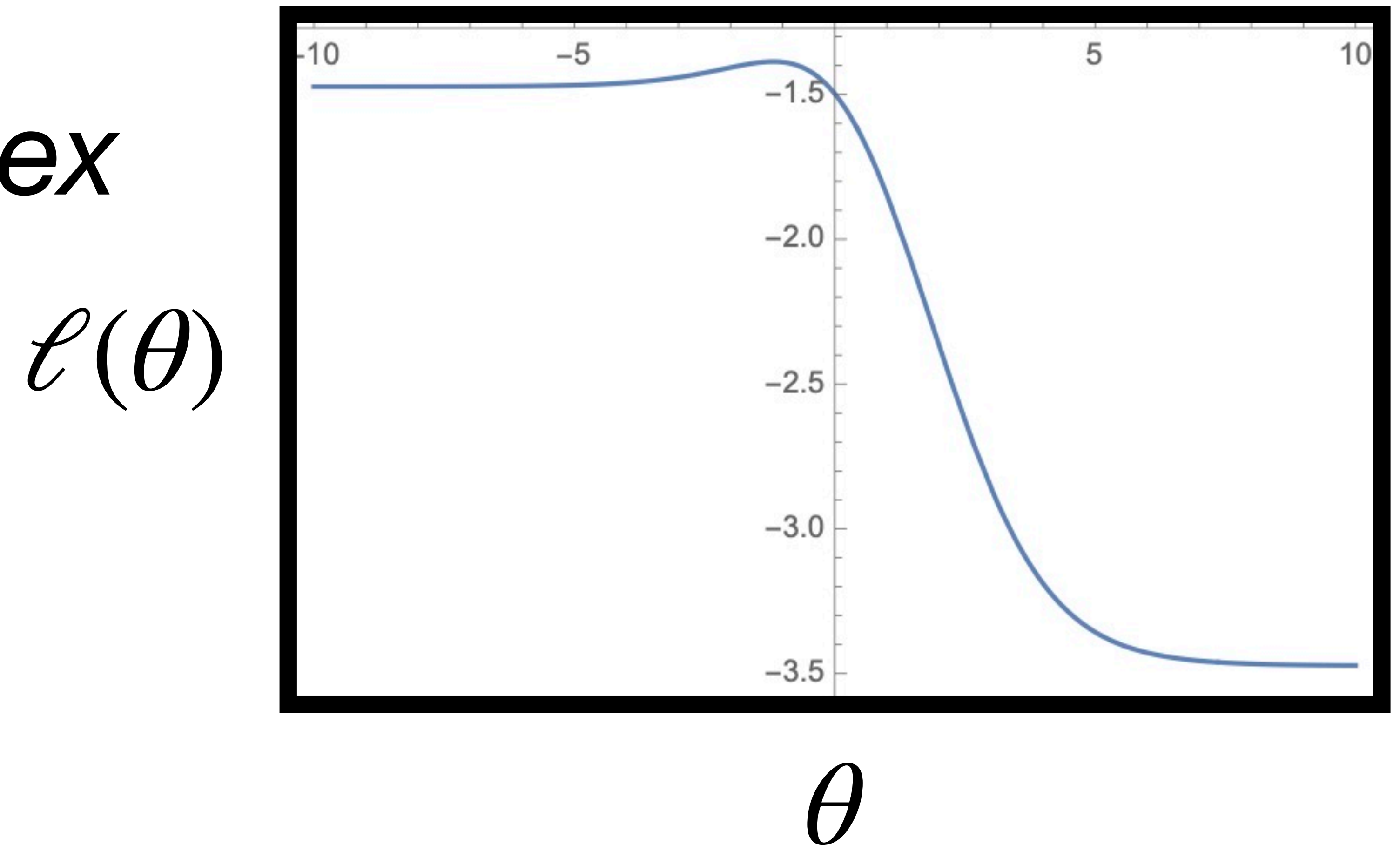
Parameter estimation from truncated data

- **However:** this time the loss can actually be *non-convex*
- Example: 1D logistic regression, $S = [-1, 3]$
- Instead, we will use *quasi-convexity:*



Parameter estimation from truncated data

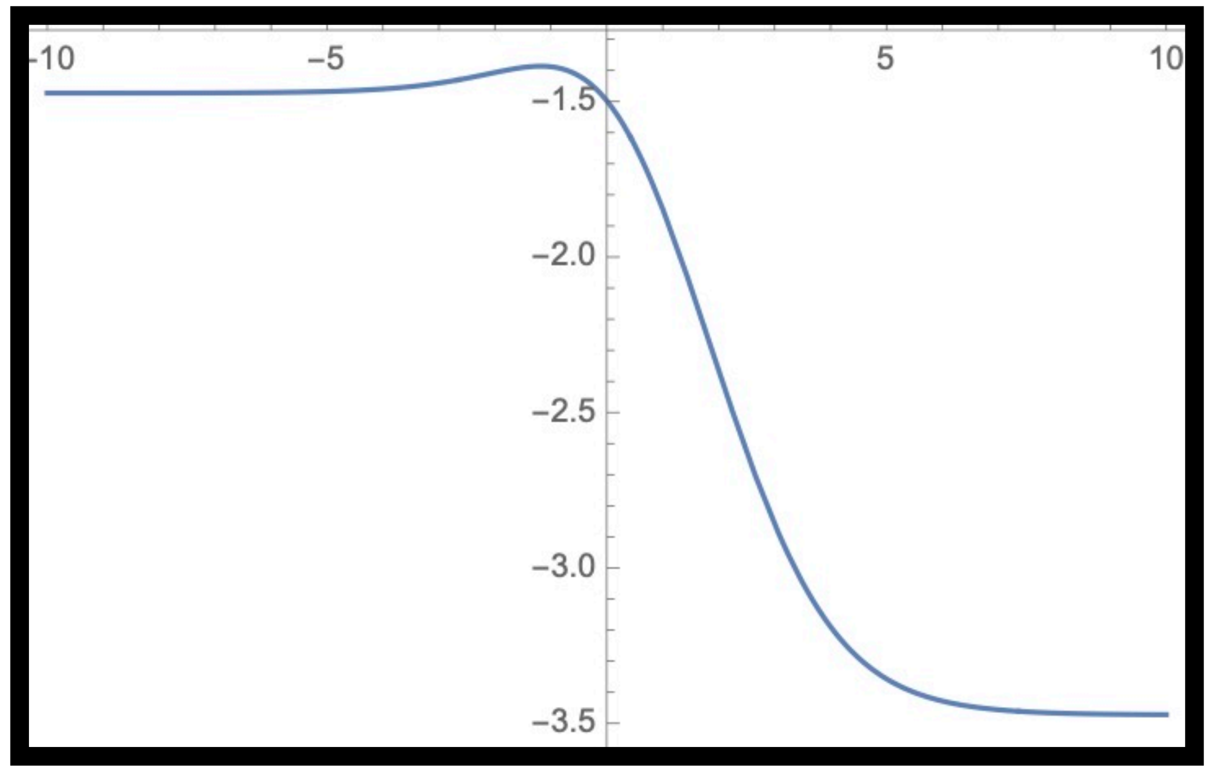
- **However:** this time the loss can actually be *non-convex*
- Example: 1D logistic regression, $S = [-1, 3]$
- Instead, we will use *quasi-convexity:*



Definition (Quasi-convexity): For all $f(y) \leq f(x)$, we have $\langle \nabla f(x), y - x \rangle \leq 0$

Parameter estimation from truncated data

- **However:** this time the loss can actually be *non-convex*
- Example: 1D logistic regression, $S = [-1,\ 3]$
- Instead, we will use *quasi-convexity:*

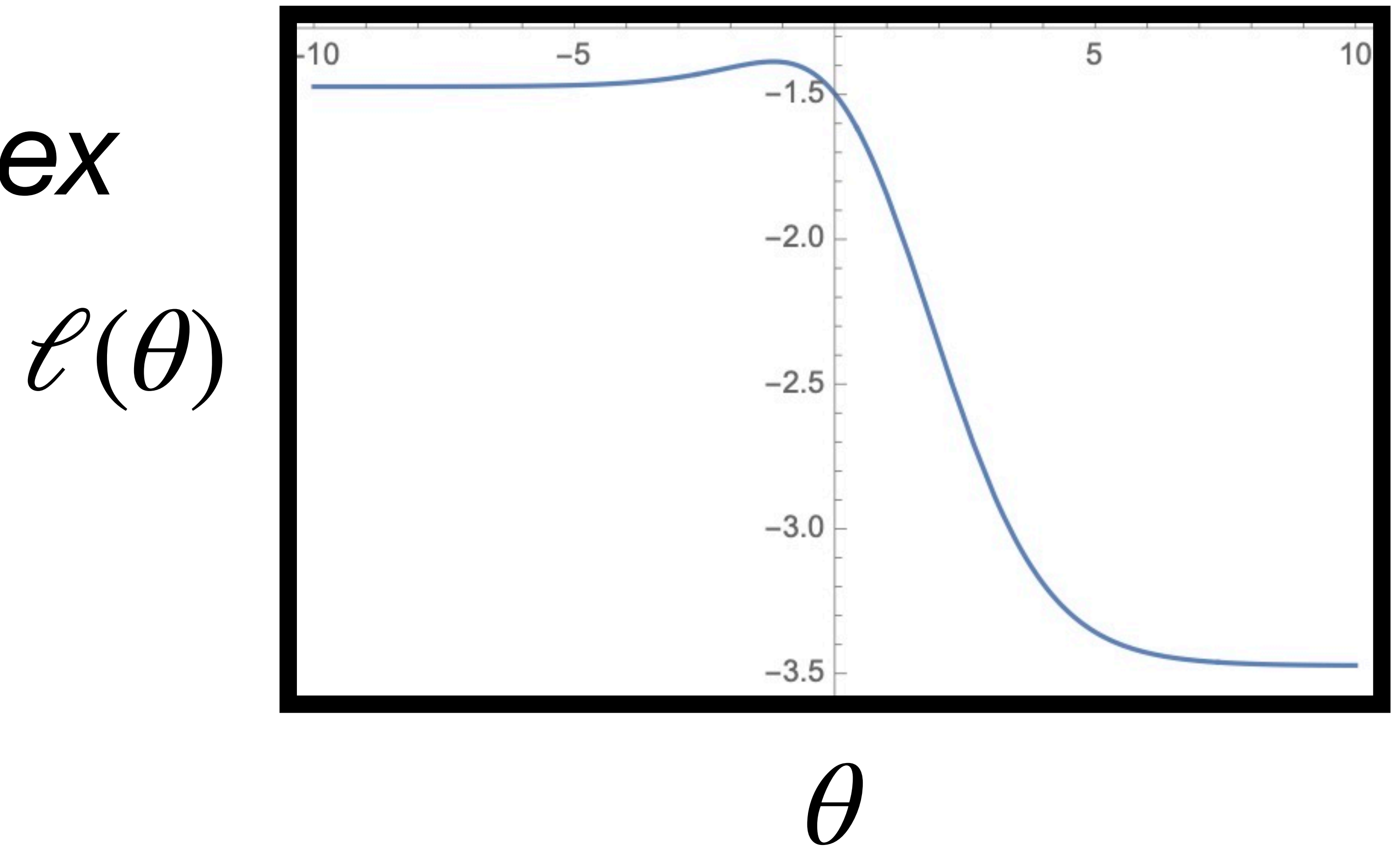


Definition (Quasi-convexity): For all $f(y) \leq f(x)$, we have $\langle \nabla f(x), y - x \rangle \leq 0$

[Hazan et al, 2015] define *strict local quasi-convexity (SLQC)* property: both stronger (inner product bounded away from zero) and weaker (y is constrained to a ball around x^*) than just QC

Parameter estimation from truncated data

- **However:** this time the loss can actually be *non-convex*
- Example: 1D logistic regression, $S = [-1, 3]$
- Instead, we will use *quasi-convexity:*



Definition (Quasi-convexity): For all $f(y) \leq f(x)$, we have $\langle \nabla f(x), y - x \rangle \leq 0$

[Hazan et al, 2015] define *strict local quasi-convexity (SLQC)* property: both stronger (inner product bounded away from zero) and weaker (y is constrained to a ball around x^*) than just QC

Their result: *normalized* SGD with minimum batch size converges to global optimum for SLQC functions

Analysis

Analysis

- **Goal:** show that NSGD on NLL converges to maximizer of the (population) log-likelihood

Analysis

- **Goal:** show that NSGD on NLL converges to maximizer of the (population) log-likelihood

- As with estimation, we define a projection set where *linear, probit, and logistic* regression are all SLQC $\implies$ NSGD converges

Analysis

- **Goal:** show that NSGD on NLL converges to maximizer of the (population) log-likelihood

- As with estimation, we define a projection set where *linear, probit, and logistic* regression are all SLQC $\Longrightarrow$ NSGD converges

- In fact, linear regression was shown strongly convex by [Daskalakis et al, 2019]

Analysis

- **Goal:** show that NSGD on NLL converges to maximizer of the (population) log-likelihood
- As with estimation, we define a projection set where *linear, probit, and logistic* regression are all SLQC $\implies$ NSGD converges
- In fact, linear regression was shown strongly convex by [Daskalakis et al, 2019]

$$x \sim D$$

Sample a covariate x

Analysis

- **Goal:** show that NSGD on NLL converges to maximizer of the (population) log-likelihood
- As with estimation, we define a projection set where *linear, probit, and logistic* regression are all SLQC $\implies$ NSGD converges
- In fact, linear regression was shown strongly convex by [Daskalakis et al, 2019]

$x \sim D$

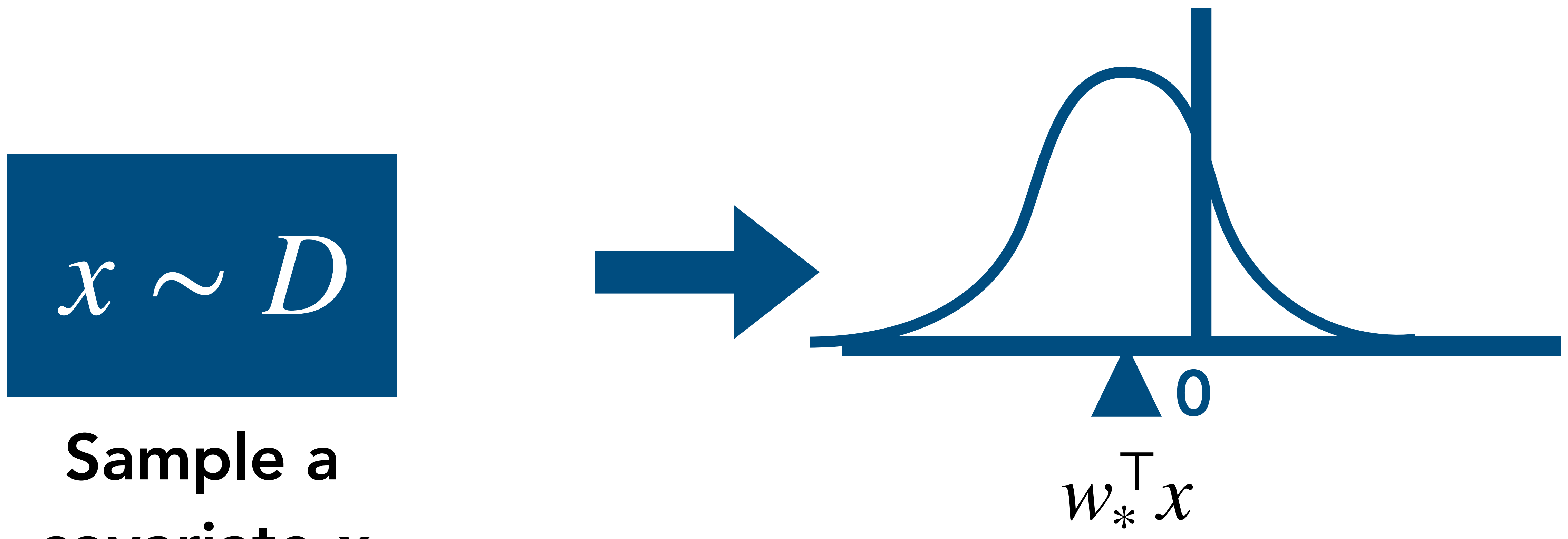
Sample a covariate x

Analysis

- **Goal:** show that NSGD on NLL converges to maximizer of the (population) log-likelihood

- As with estimation, we define a projection set where *linear, probit, and logistic* regression are all SLQC $\implies$ NSGD converges

- In fact, linear regression was shown strongly convex by [Daskalakis et al, 2019]

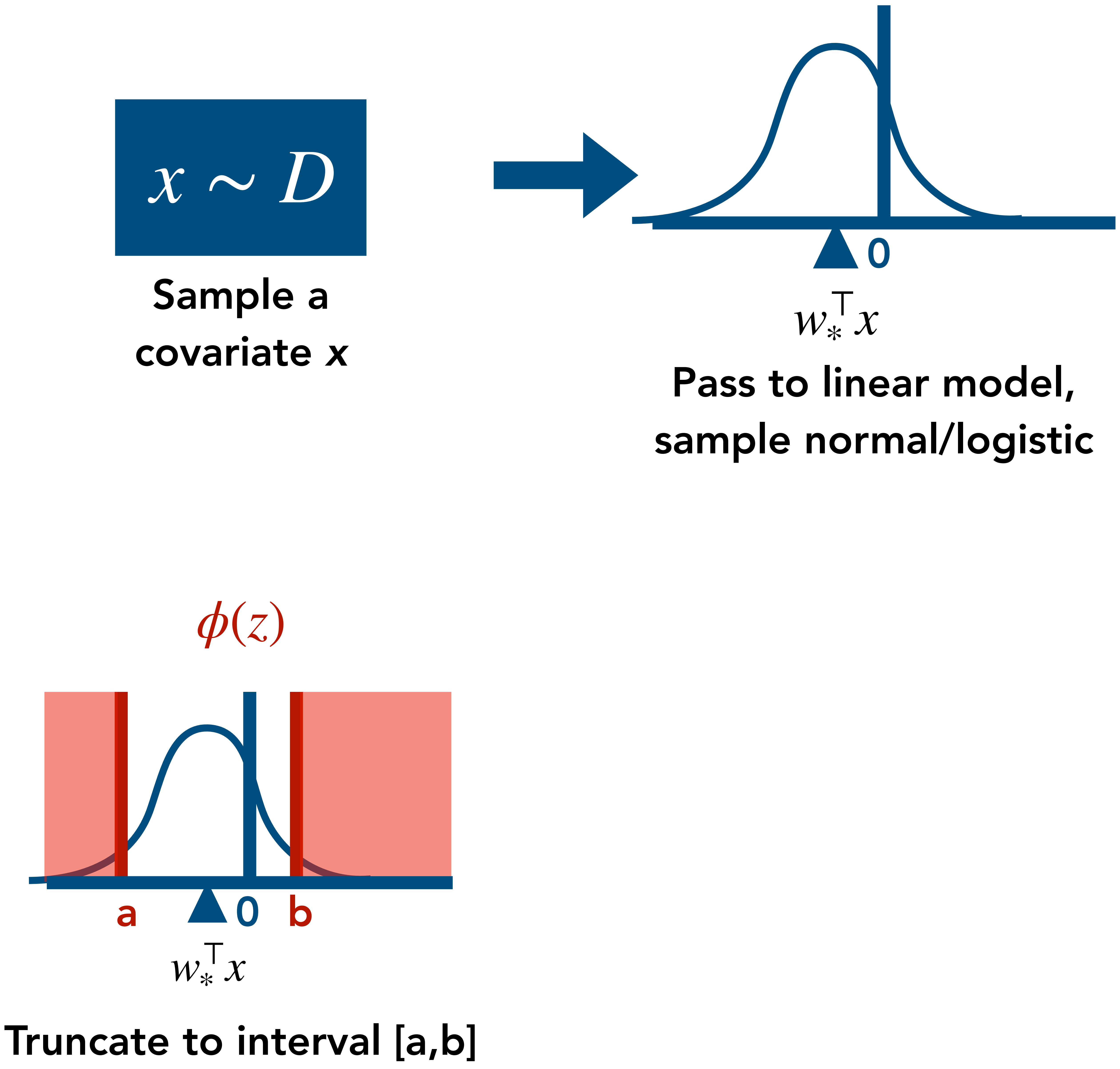


Sample a covariate x

Pass to linear model, sample normal/logistic

Analysis

- **Goal:** show that NSGD on NLL converges to maximizer of the (population) log-likelihood
- As with estimation, we define a projection set where *linear, probit, and logistic* regression are all SLQC $\implies$ NSGD converges
- In fact, linear regression was shown strongly convex by [Daskalakis et al, 2019]



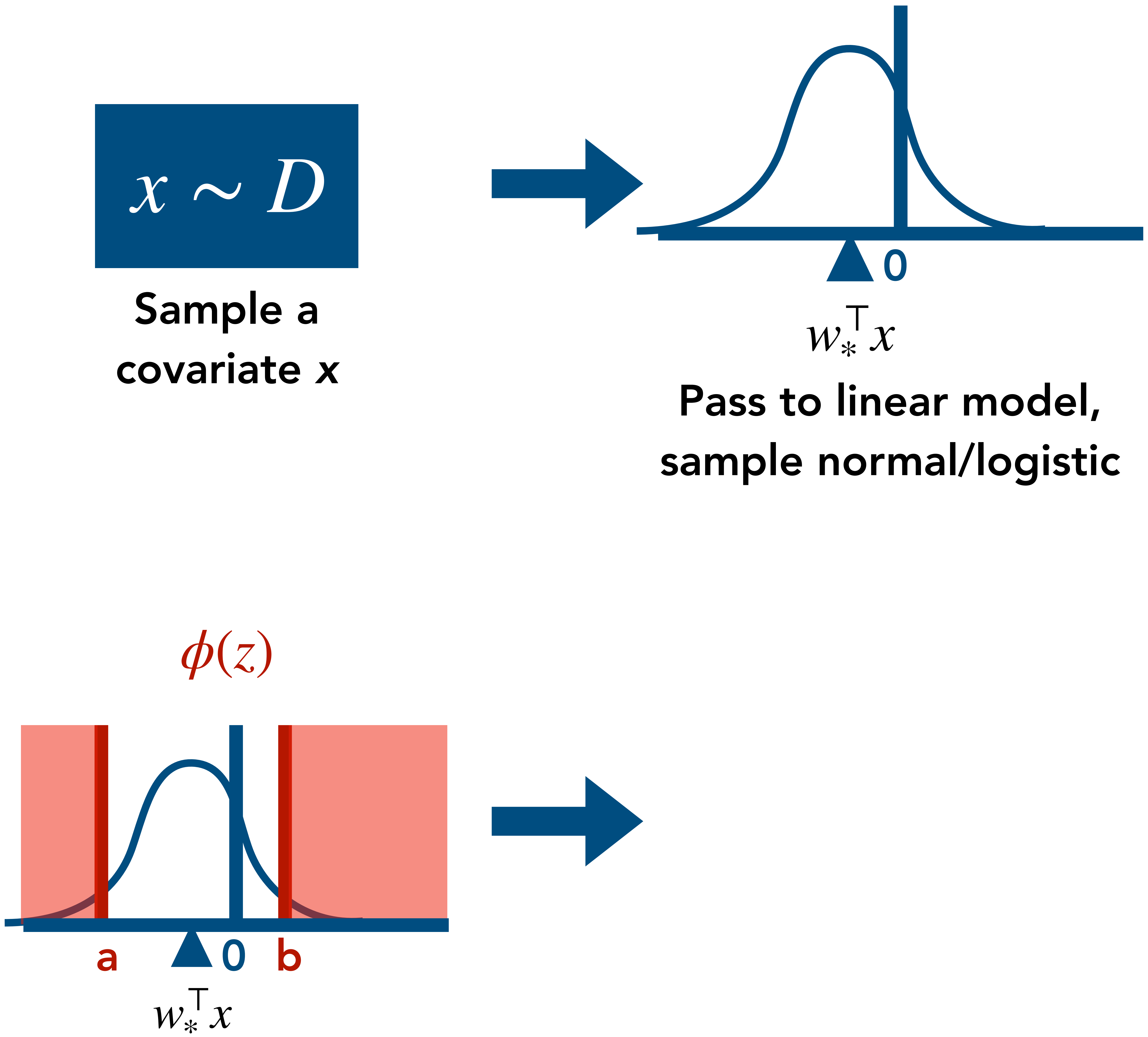
Sample a covariate x

Pass to linear model, sample normal/logistic

Truncate to interval [a,b]

Analysis

- **Goal:** show that NSGD on NLL converges to maximizer of the (population) log-likelihood
- As with estimation, we define a projection set where *linear, probit, and logistic* regression are all SLQC $\implies$ NSGD converges
- In fact, linear regression was shown strongly convex by [Daskalakis et al, 2019]



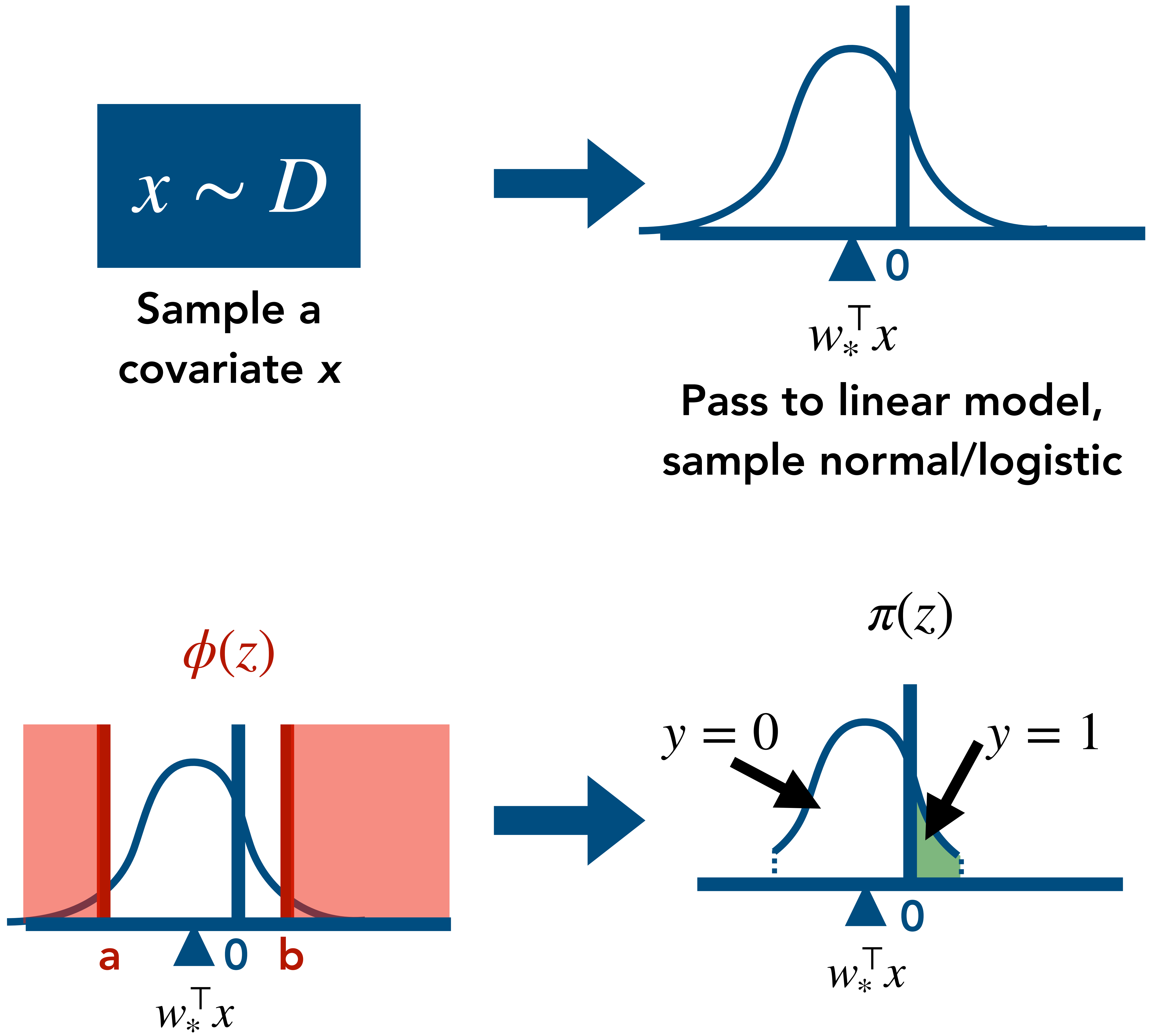
Sample a covariate x

Pass to linear model, sample normal/logistic

Truncate to interval [a,b]

Analysis

- **Goal:** show that NSGD on NLL converges to maximizer of the (population) log-likelihood
- As with estimation, we define a projection set where *linear, probit, and logistic* regression are all SLQC $\implies$ NSGD converges
- In fact, linear regression was shown strongly convex by [Daskalakis et al, 2019]



Sample a covariate x

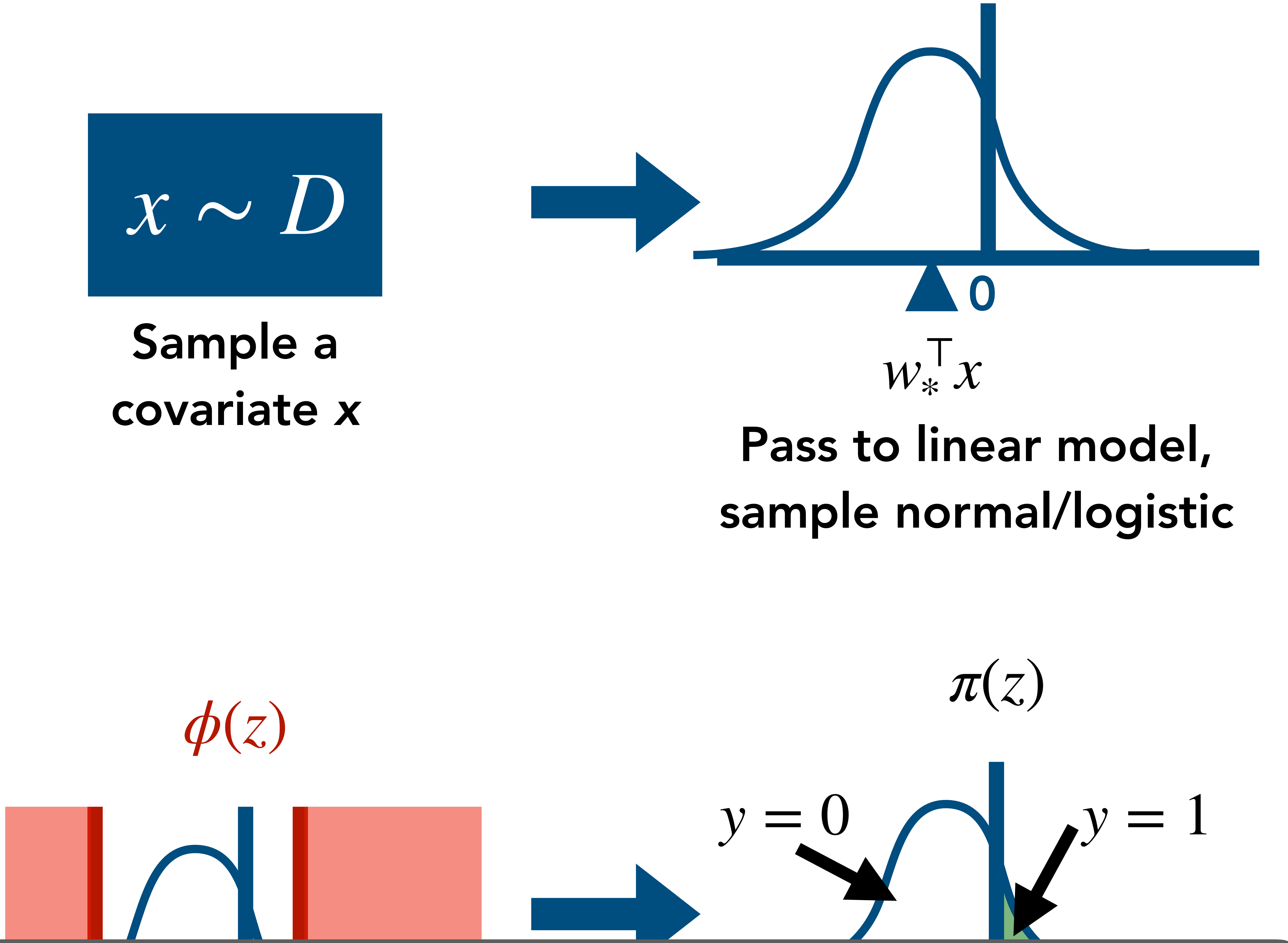
Pass to linear model, sample normal/logistic

Truncate to interval [a,b]

Project to get a label

Analysis

- **Goal:** show that NSGD on NLL converges to maximizer of the (population) log-likelihood
- As with estimation, we define a projection set where *linear, probit, and logistic* regression are all SLQC $\implies$ NSGD converges
- In fact, linear regression was



Sample a covariate x

Pass to linear model, sample normal/logistic

Theorem (informal): if for every $x \in \mathbb{R}^d$, there is a non-zero ($\alpha > 0$) probability that $y = \{0,1\}$, then NSGD finds an ε-minimizer of the NLL in poly$(1/\alpha, 1/\varepsilon, d)$ steps.

Experiments

Synthetic data

Experiments

Synthetic data

Setup:

Experiments

Synthetic data

Setup:

- $\theta_* \sim \mathscr{U}([-1,1]^{10})$

Experiments

Synthetic data

Setup:

- $\theta_* \sim \mathcal{U}([-1,1]^{10})$
- $X \sim \mathcal{U}([0,1]^{10\times n})$

Experiments

Synthetic data

Setup:

- $\theta_* \sim \mathscr{U}([-1,1]^{10})$
- $X \sim \mathscr{U}([0,1]^{10\times n})$
- $\varepsilon \sim D_N$ (normal/log)

Experiments

Synthetic data

Setup:

- $\theta_* \sim \mathcal{U}([-1,1]^{10})$
- $X \sim \mathcal{U}([0,1]^{10 \times n})$
- $\varepsilon \sim D_N$ (normal/log)
- $Z := \theta_*^\top X + \varepsilon$

Experiments

Synthetic data

Setup:

- $\theta_* \sim \mathcal{U}([-1,1]^{10})$
- $X \sim \mathcal{U}([0,1]^{10\times n})$
- $\varepsilon \sim D_N$ (normal/log)
- $Z := \theta_*^\top X + \varepsilon$
- Truncation $[C, \infty)$

Experiments

Synthetic data

Setup:

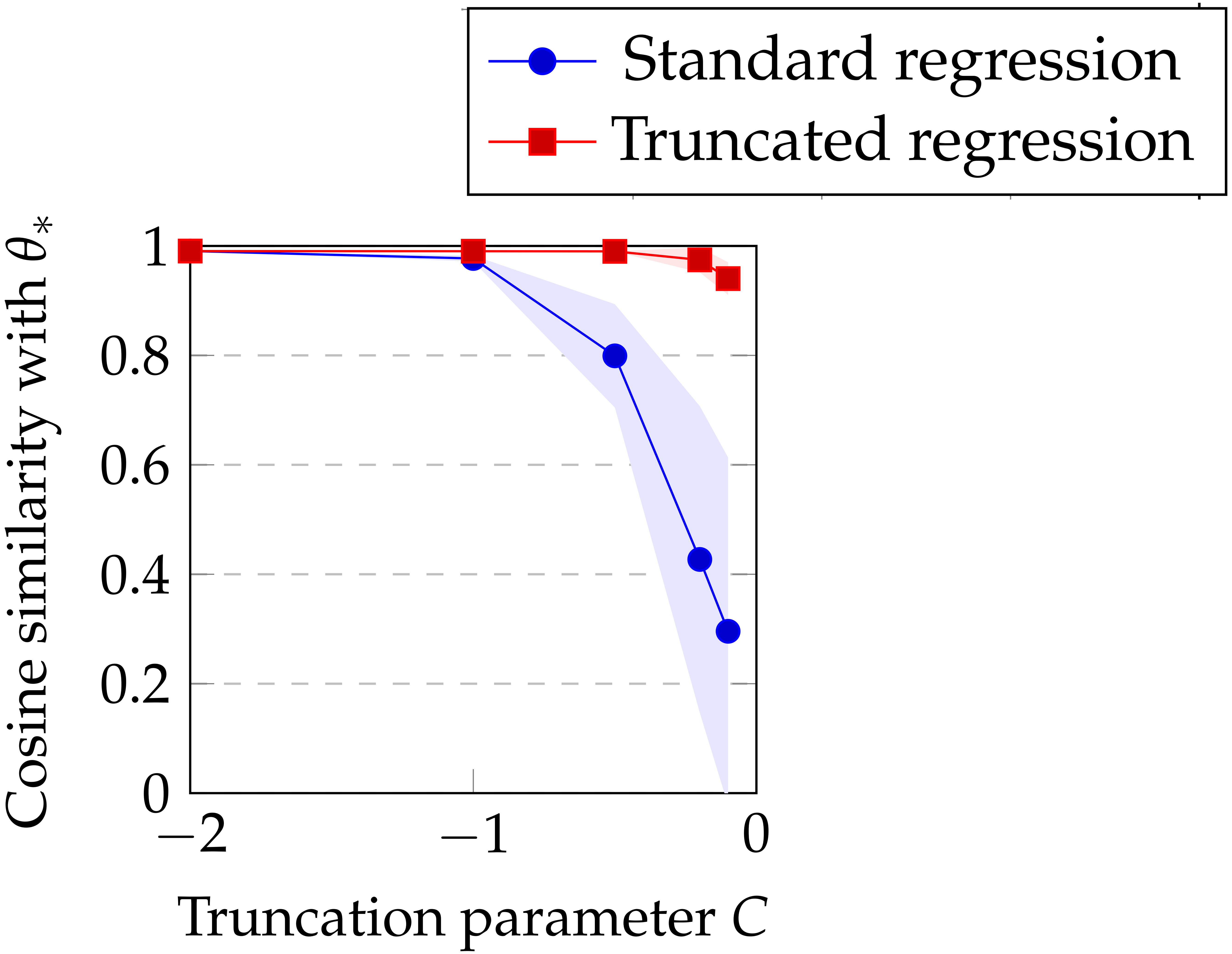
- $\theta_* \sim \mathscr{U}([-1,1]^{10})$
- $X \sim \mathscr{U}([0,1]^{10 \times n})$
- $\varepsilon \sim D_N$ (normal/log)
- $Z := \theta_*^\top X + \varepsilon$
- Truncation $[C, \infty)$
- $Y = \mathbf{1}_{Z \geq 0}$

Experiments

Synthetic data

Setup:

- $\theta_* \sim \mathscr{U}([-1,1]^{10})$
- $X \sim \mathscr{U}([0,1]^{10\times n})$
- $\varepsilon \sim D_N$ (normal/log)
- $Z := \theta_*^\top X + \varepsilon$
- Truncation $[C, \infty)$
- $Y = \mathbf{1}_{Z\geq 0}$

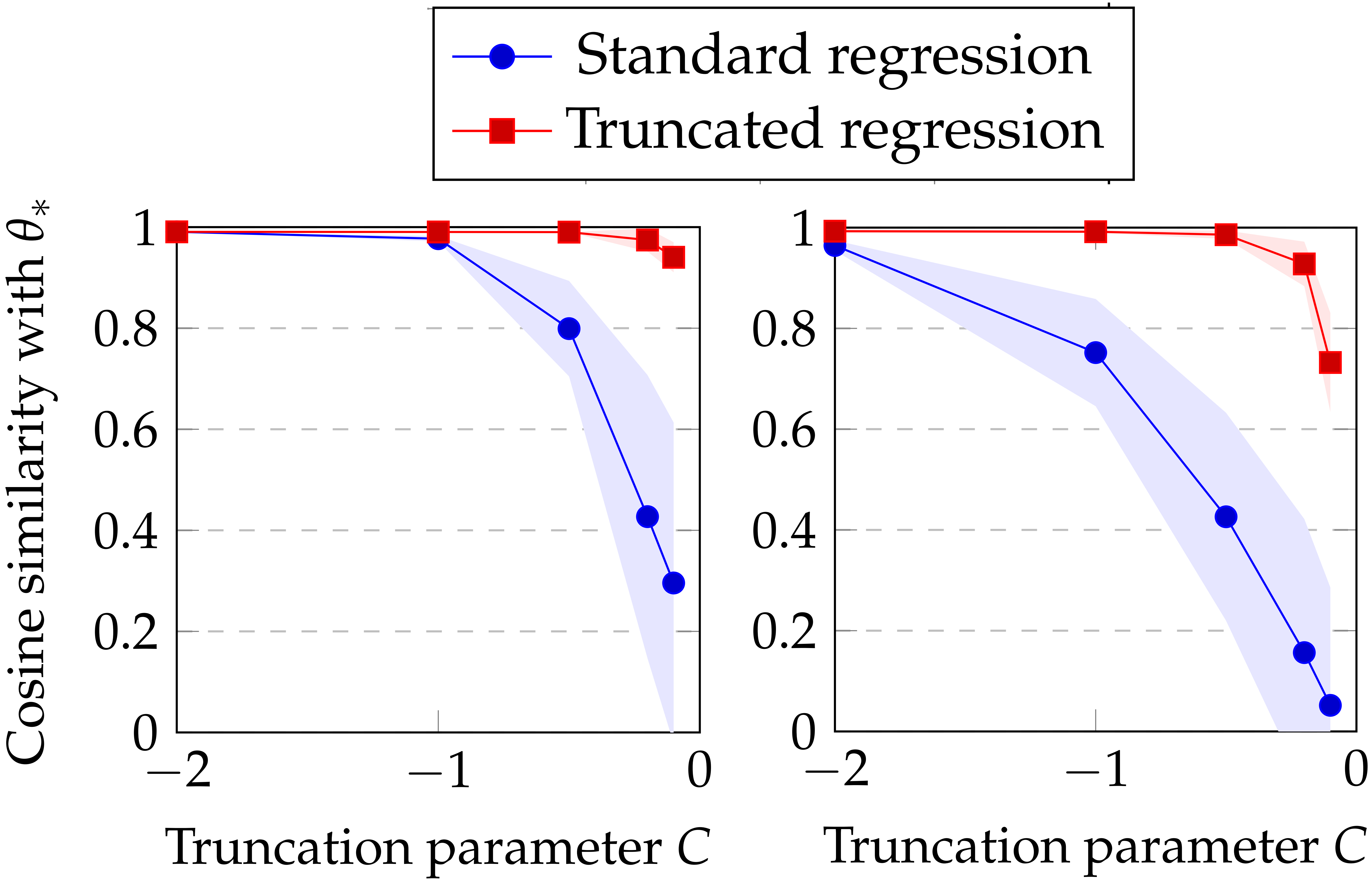


Experiments

Synthetic data

Setup:

- $\theta_* \sim \mathscr{U}([-1,1]^{10})$
- $X \sim \mathscr{U}([0,1]^{10\times n})$
- $\varepsilon \sim D_N$ (normal/log)
- $Z := \theta_*^\top X + \varepsilon$
- Truncation $[C, \infty)$
- $Y = \mathbf{1}_{Z \geq 0}$



Experiments

UCI MSD dataset

Experiments

UCI MSD dataset

Setup:

Experiments

UCI MSD dataset

Setup:

- X : song attributes

Experiments

UCI MSD dataset

Setup:

- X : song attributes
- Z : year recorded

Experiments

UCI MSD dataset

Setup:

- X : song attributes
- Z : year recorded
- Truncation $[C, \infty)$

Experiments

UCI MSD dataset

Setup:

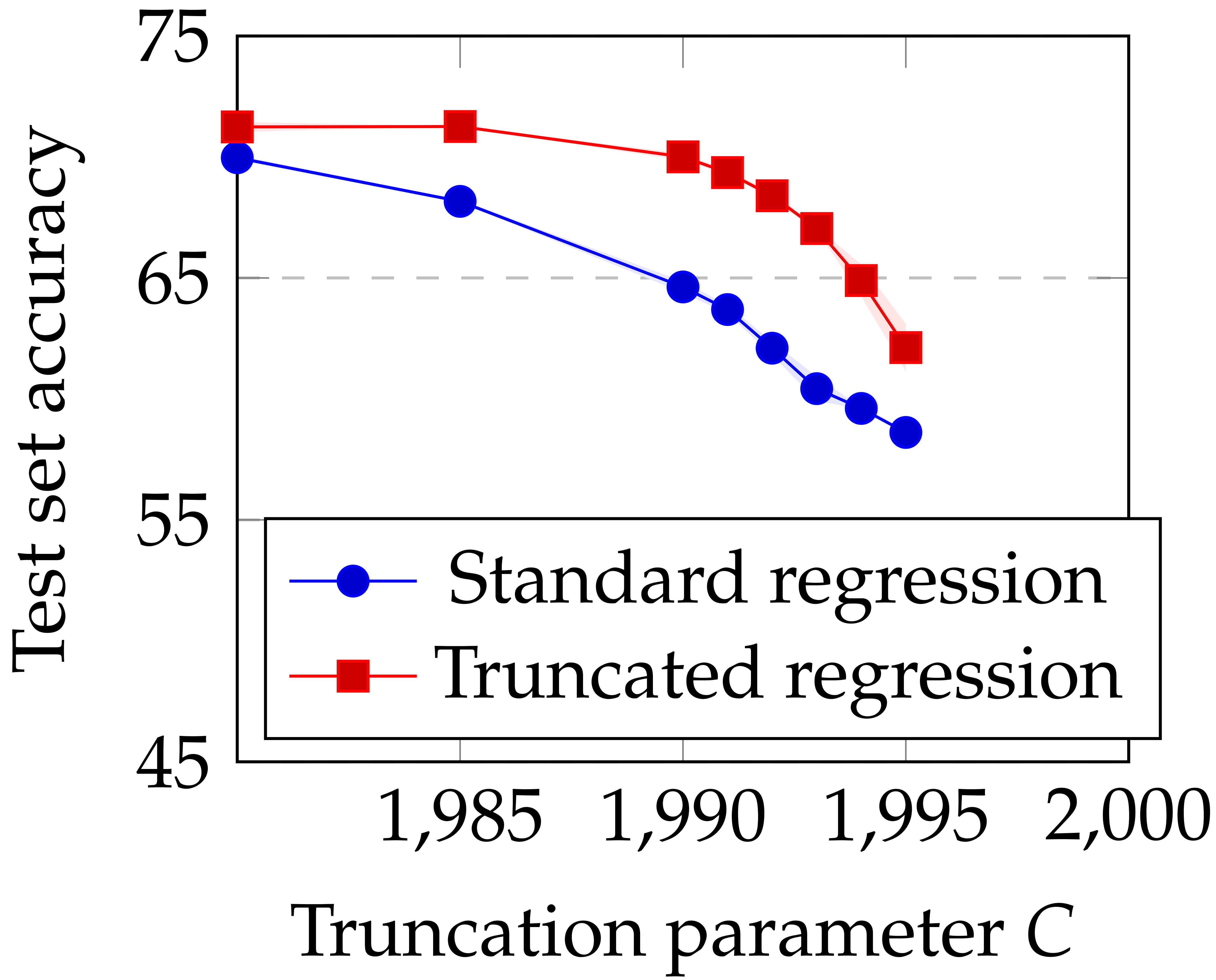
- X : song attributes
- Z : year recorded
- Truncation $[C, \infty)$
- Y : recorded before ’96?

Experiments

UCI MSD dataset

Setup:

- X : song attributes
- Z : year recorded
- Truncation $[C, \infty)$
- Y : recorded before '96?



Extensions and Limitations

Mixture of two Gaussians [Nagarajan & Panageas, 2019]

Extensions and Limitations

Mixture of two Gaussians [Nagarajan & Panageas, 2019]

- We saw how to estimate parameters of truncated Gaussian

Extensions and Limitations

Mixture of two Gaussians [Nagarajan & Panageas, 2019]

- We saw how to estimate parameters of truncated Gaussian
- Nagarajan & Panageas consider truncated mixture of two Gaussians

Extensions and Limitations

Mixture of two Gaussians [Nagarajan & Panageas, 2019]

- We saw how to estimate parameters of truncated Gaussian
- Nagarajan & Panageas consider truncated mixture of two Gaussians

$$\frac{1}{2}\mathcal{N}(\mu, \Sigma) + \frac{1}{2}\mathcal{N}(-\mu, \Sigma)$$

Extensions and Limitations

Mixture of two Gaussians [Nagarajan & Panageas, 2019]

- We saw how to estimate parameters of truncated Gaussian
- Nagarajan & Panageas consider truncated mixture of two Gaussians

$$\frac{1}{2}\mathcal{N}(\mu, \Sigma) + \frac{1}{2}\mathcal{N}(-\mu, \Sigma)$$

- Likelihood can be optimized using the standard expectation-maximization method, gives local improvement guarantee

Extensions and Limitations

Mixture of two Gaussians [Nagarajan & Panageas, 2019]

- We saw how to estimate parameters of truncated Gaussian
- Nagarajan & Panageas consider truncated mixture of two Gaussians

$$\frac{1}{2}\mathcal{N}(\mu, \Sigma) + \frac{1}{2}\mathcal{N}(-\mu, \Sigma)$$

- Likelihood can be optimized using the standard expectation-maximization method, gives local improvement guarantee
- **Global convergence** of EM for truncated mixtures is shown

Extensions and Limitations

Unknown truncation set [Kontonis et al, 2019]

Extensions and Limitations

Unknown truncation set [Kontonis et al, 2019]

- For general truncation sets S, estimating parameters is **impossible**

Extensions and Limitations

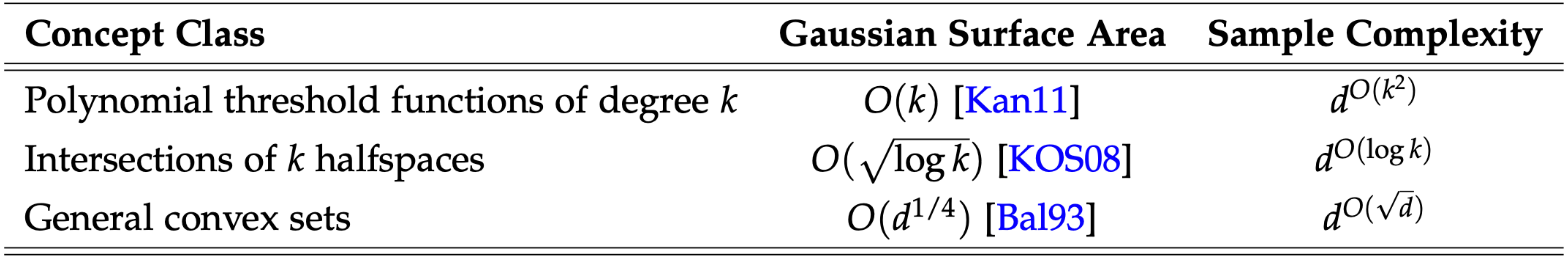
Unknown truncation set [Kontonis et al, 2019]

- For general truncation sets S, estimating parameters is **impossible**
- However, [Kontonis et al, 2019] show that **learning** S is possible if the space of possible sets has bounded VC dimension, or Gaussian surface area (measures of complexity):

Extensions and Limitations

Unknown truncation set [Kontonis et al, 2019]

- For general truncation sets S, estimating parameters is **impossible**
- However, [Kontonis et al, 2019] show that **learning** S is possible if the space of possible sets has bounded VC dimension, or Gaussian surface area (measures of complexity):



Concept Class	Gaussian Surface Area	Sample Complexity
Polynomial threshold functions of degree k	$O(k)$ [Kan11]	$d^{O(k^2)}$
Intersections of k halfspaces	$O(\sqrt{\log k})$ [KOS08]	$d^{O(\log k)}$
General convex sets	$O(d^{1/4})$ [Bal93]	$d^{O(\sqrt{d})}$

Extensions and Limitations

High-dimensional (sparse) setting [Daskalakis et al, 2020]

Extensions and Limitations

High-dimensional (sparse) setting [Daskalakis et al, 2020]

- For linear regression, we can also consider the setting where the covariates x_i are very high dimensional, but k-sparse

Extensions and Limitations

High-dimensional (sparse) setting [Daskalakis et al, 2020]

- For linear regression, we can also consider the setting where the covariates x_i are very high dimensional, but k-sparse
- In this setting, [Daskalakis et al, 2020] propose a modified LASSO algorithm for dealing with truncation

Extensions and Limitations

High-dimensional (sparse) setting [Daskalakis et al, 2020]

- For linear regression, we can also consider the setting where the covariates x_i are very high dimensional, but k-sparse
- In this setting, [Daskalakis et al, 2020] propose a modified LASSO algorithm for dealing with truncation
- Recovers parameters under truncation with error $O(\sqrt{k \log(d)/n}$

Future Work

Future Work

- Robustness to model mis-specification

Future Work

- Robustness to model mis-specification
- Connections to causal inference:

Future Work

- Robustness to model mis-specification
- Connections to causal inference:
  - Selection bias

Future Work

- Robustness to model mis-specification
- Connections to causal inference:
  - Selection bias
  - Truncated outcomes (e.g. death in medical trials, dropping out in school studies, non-response in surveys)

Future Work

- Robustness to model mis-specification
- Connections to causal inference:
    - Selection bias
    - Truncated outcomes (e.g. death in medical trials, dropping out in school studies, non-response in surveys)
- Improving algorithms for *censored* statistics (where the learner observes the truncation)