Empirical Bayes in Action, and How Pretraining/Transformer Solves It

Yanjun Han (NYU Math and Data Science)

Joint work with:

Nick Cannella	NYU
Jonathan Niles-Weed	NYU
Yury Polyanskiy	MIT
Yandi Shen	CMU
Anzo Teh	MIT
Yihong Wu	Yale

CDS Colloquium
March 27, 2026

Empirical Bayes

Key idea:

→ Bayesian inference with a data-driven prior

Core strength:

→ automatic regularization (shrinkage)

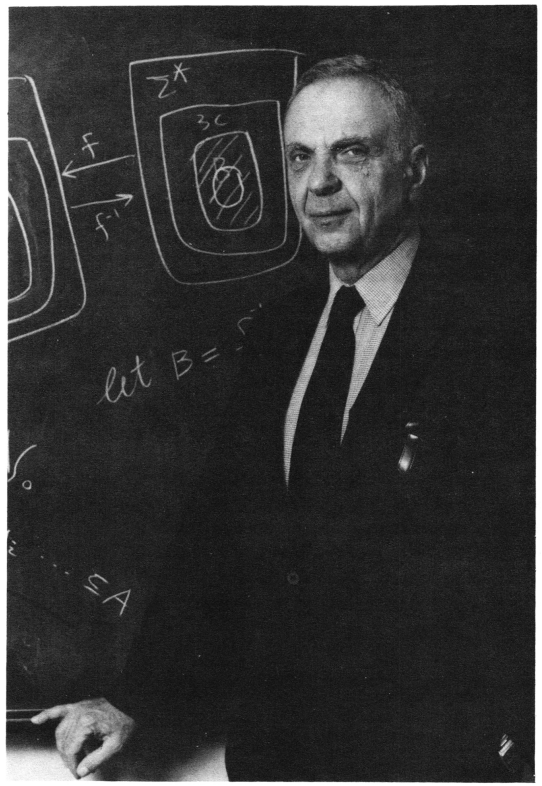
→ borrows strength across groups to improve estimates

In practice:

→ strong empirical performance

→ often yields tuning-free procedures

→ applies beyond "Bayesian-looking" problems

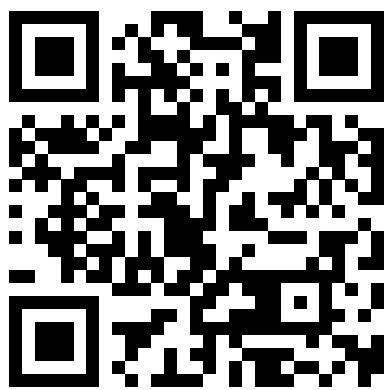


Herbert Robbins (1915–2001)

Part I: An EB Approach to Distribution Estimation

Y. Han, J. Niles-Weed, Y. Shen, Y. Wu

Besting Good–Turing: Optimality of Non-Parametric Maximum Likelihood for Distribution Estimation



arXiv:2509.07355

Always Good Turing: Asymptotically Optimal Probability Estimation

ALON ORLITSKY, NARAYANA P. SANTHANAM, AND JUNAN ZHANG Authors Info & Affiliations

SCIENCE • 17 Oct 2003 • Vol 302, Issue 5644 • pp. 427-431 • DOI: 10.1126/science.1088284

Always Good Turing: Asymptotically Optimal Probability Estimation

ALON ORLITSKY, NARAYANA P. SANTHANAM, AND JUNAN ZHANG Authors Info & Affiliations

SCIENCE • 17 Oct 2003 • Vol 302, Issue 5644 • pp. 427-431 • DOI: 10.1126/science.1088284

Always Good Turing: Asymptotically Optimal Probability Estimation

ALON ORLITSKY, NARAYANA P. SANTHANAM, AND JUNAN ZHANG Authors Info & Affiliations

SCIENCE • 17 Oct 2003 • Vol 302, Issue 5644 • pp. 427-431 • DOI: 10.1126/science.1088284

Always Good Turing: Asymptotically Optimal Probability Estimation

ALON ORLITSKY, NARAYANA P. SANTHANAM, AND JUNAN ZHANG Authors Info & Affiliations

SCIENCE • 17 Oct 2003 • Vol 302, Issue 5644 • pp. 427-431 • DOI: 10.1126/science.1088284

Conferences > 2007 IEEE International Sympo...

A Better Good-Turing Estimator for Sequence Probabilities

Publisher: IEEE Cite This PDF

Aaron B. Wagner ; Pramod Viswanath ; Sanjeev R. Kulkarni All Authors

ARTICLE

Competitive distribution estimation: why is Good-Turing good

Authors: Alon Orlitsky, Ananda Theertha Suresh Authors Info & Claims

NIPS'15: Proceedings of the 29th International Conference on Neural Information Processing Systems - Volume 2
Pages 2143 - 2151

Mathematics > Statistics Theory

[Submitted on 9 Sep 2025]

Besting Good--Turing: Optimality of Non-Parametric Maximum Likelihood for Distribution Estimation

Yanjun Han, Jonathan Niles-Weed, Yandi Shen, Yihong Wu

Probability estimation

Given a sample $X_1, \ldots, X_n$ drawn from a distribution $p^\star = (p_1^\star, \ldots, p_k^\star)$, estimate these probabilities.

Working example:

- $\rightarrow$ estimate the word frequency of a corpus based on a sample
- $\rightarrow$ think both n and k are large

Most common approach

Empirical frequency:

$$\widehat{p}_i \propto N_i = \text{ the number of times } i\text{th element occurs.}$$

This is the maximum likelihood estimator (MLE).

Most common approach

Empirical frequency:

$$\widehat{p}_i \propto N_i = \text{ the number of times } i\text{th element occurs.}$$

This is the maximum likelihood estimator (MLE).

Advantages:

- $\rightarrow$ interpretable, used everywhere
- $\rightarrow$ optimal statistical guarantee in the worst case

Most common approach

Empirical frequency:

$$\widehat{p}_i \propto N_i = \text{ the number of times } i\text{th element occurs.}$$

This is the maximum likelihood estimator (MLE).

Advantages:

→ interpretable, used everywhere

→ optimal statistical guarantee in the worst case

Issues:

→ smoothing needed for unseen or rare elements

→ in practice, use Laplace smoothing: $\widehat{p}_i \propto N_i + 1$

→ need to tune the smoothing parameter, oversmoothing (later)

Toy example: Safari preparation

Empirical frequency:

$$\hat{p}_{🦒} = \frac{1}{5}, \hat{p}_{🐘} = \hat{p}_{🦓} = \frac{2}{5}$$

Toy example: Safari preparation

Empirical frequency:

$$\hat{p}_{🦒} = \frac{1}{5}, \hat{p}_{🐘} = \hat{p}_{🦓} = \frac{2}{5}$$

What's the catch?

Toy example: Safari preparation

Empirical frequency:

$$\hat{p}_{🦒} = \frac{1}{5}, \hat{p}_{🐘} = \hat{p}_{🦓} = \frac{2}{5}$$

$\rightarrow$ Need to be cautious with unseen or unlikely symbols, hence the proverbial *"Here be lions"*

Toy example: Safari preparation

Empirical frequency:

$$\hat{p}_{🦒} = \frac{1}{5}, \hat{p}_{🐘} = \hat{p}_{🦓} = \frac{2}{5}$$

→ Need to be cautious with unseen or unlikely symbols, hence the proverbial *"Here be lions"*

→ In NLP, it is crucial to ensure estimates are positive to generalize beyond existing corpus

Good–Turing

VOLUME 40, PARTS 3 AND 4 DECEMBER 1953

THE POPULATION FREQUENCIES OF SPECIES AND THE ESTIMATION OF POPULATION PARAMETERS

BY I. J. GOOD

The formula (2) was first suggested to me, together with an intuitive demonstration, by Dr A. M. Turing several years ago. Hence a very large part of the credit for the present paper should be given to him, and I am most grateful to him for allowing me to publish this work.

$$\widehat{p}_i \propto (N_i + 1)$$

Good–Turing

VOLUME 40, PARTS 3 AND 4 DECEMBER 1953

THE POPULATION FREQUENCIES OF SPECIES AND THE ESTIMATION OF POPULATION PARAMETERS

BY I. J. GOOD

The formula (2) was first suggested to me, together with an intuitive demonstration, by Dr A. M. Turing several years ago. Hence a very large part of the credit for the present paper should be given to him, and I am most grateful to him for allowing me to publish this work.

$$\widehat{p}_i \propto (N_i + 1)\frac{\#\text{ symbols appearing once more}}{\#\text{ symbols appearing same time}}$$

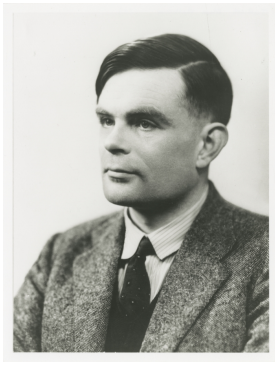
Good–Turing

VOLUME 40, PARTS 3 AND 4 DECEMBER 1953

THE POPULATION FREQUENCIES OF SPECIES AND THE ESTIMATION OF POPULATION PARAMETERS

BY I. J. GOOD

The formula (2) was first suggested to me, together with an intuitive demonstration, by Dr A. M. Turing several years ago. Hence a very large part of the credit for the present paper should be given to him, and I am most grateful to him for allowing me to publish this work.



$$\widehat{p}_i \propto (N_i + 1)\frac{\#\text{ symbols appearing once more}}{\#\text{ symbols appearing same time}}$$

→ mysterious/ingenious

→ much better performance than Laplace etc

→ widely used in language modeling, speech recognition, genomics, etc

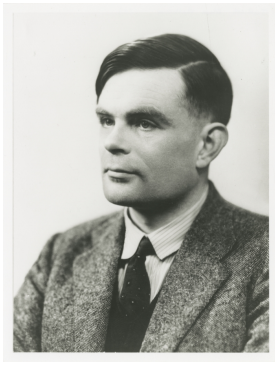
Good–Turing

VOLUME 40, PARTS 3 AND 4 DECEMBER 1953

THE POPULATION FREQUENCIES OF SPECIES AND THE ESTIMATION OF POPULATION PARAMETERS

BY I. J. GOOD

The formula (2) was first suggested to me, together with an intuitive demonstration, by Dr A. M. Turing several years ago. Hence a very large part of the credit for the present paper should be given to him, and I am most grateful to him for allowing me to publish this work.



$$\widehat{p}_i \propto (N_i + 1)\frac{\#\text{ symbols appearing once more}}{\#\text{ symbols appearing same time}}$$

→ mysterious/ingenious

→ much better performance than Laplace etc

→ widely used in language modeling, speech recognition, genomics, etc

→ requires *modification* to be usable, e.g. use empirical for frequent symbols [Church-Gale '95]

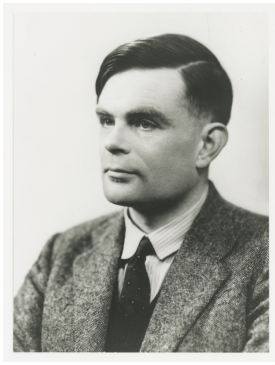
Good–Turing

VOLUME 40, PARTS 3 AND 4 DECEMBER 1953

THE POPULATION FREQUENCIES OF SPECIES AND THE ESTIMATION OF POPULATION PARAMETERS

BY I. J. GOOD

The formula (2) was first suggested to me, together with an intuitive demonstration, by Dr A. M. Turing several years ago. Hence a very large part of the credit for the present paper should be given to him, and I am most grateful to him for allowing me to publish this work.



$$\widehat{p}_i \propto (N_i + 1)\frac{\#\text{ symbols appearing once more}}{\#\text{ symbols appearing same time}}$$

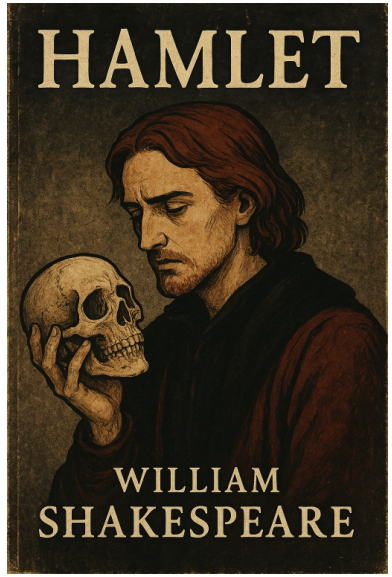
- → mysterious/ingenious
- → much better performance than Laplace etc
- → widely used in language modeling, speech recognition, genomics, etc
- → requires *modification* to be usable, e.g. use empirical for frequent symbols [Church-Gale '95]
- → practical issues: hard to tune hyperparameters, sensitive performance, interpretability

A real-data experiment

- → *Hamlet*: *28,799* words in total and *4,804* distinct
- → Randomly sample 10%, estimate the word frequency
- → Kullback–Leibler (KL) loss:

$$\mathrm{KL}(p^\star \| \widehat{p}) \triangleq \sum_i p_i^\star \log \frac{p_i^\star}{\widehat{p}_i}$$

- → KL against uniform: 2.94 bits



Results

Estimator	KL Risk (bits)
Empirical frequency	∞

Results

Estimator	KL Risk (bits)
Empirical frequency	∞
Original Good-Turing	∞

Results

Estimator	KL Risk (bits)
Empirical frequency	∞
Original Good-Turing	∞
Laplace	0.6231

Results

Estimator	KL Risk (bits)
Empirical frequency	∞
Original Good-Turing	∞
Laplace	0.6231
Braess-Sauer	0.4719

Results

Estimator	KL Risk (bits)
Empirical frequency	∞
Original Good-Turing	∞
Laplace	0.6231
Braess-Sauer	0.4719
Modified Good-Turing	0.2732

Results

Estimator	KL Risk (bits)
Empirical frequency	∞
Original Good-Turing	∞
Laplace	0.6231
Braess-Sauer	0.4719
Modified Good-Turing	0.2732
New estimator (NPMLE)	0.2513

Results

Estimator	KL Risk (bits)
Empirical frequency	∞
Original Good-Turing	∞
Laplace	0.6231
Braess-Sauer	0.4719
Modified Good-Turing	0.2732
New estimator (NPMLE)	0.2513

$\rightarrow$ Worst-case risk is too crude to be a useful benchmark

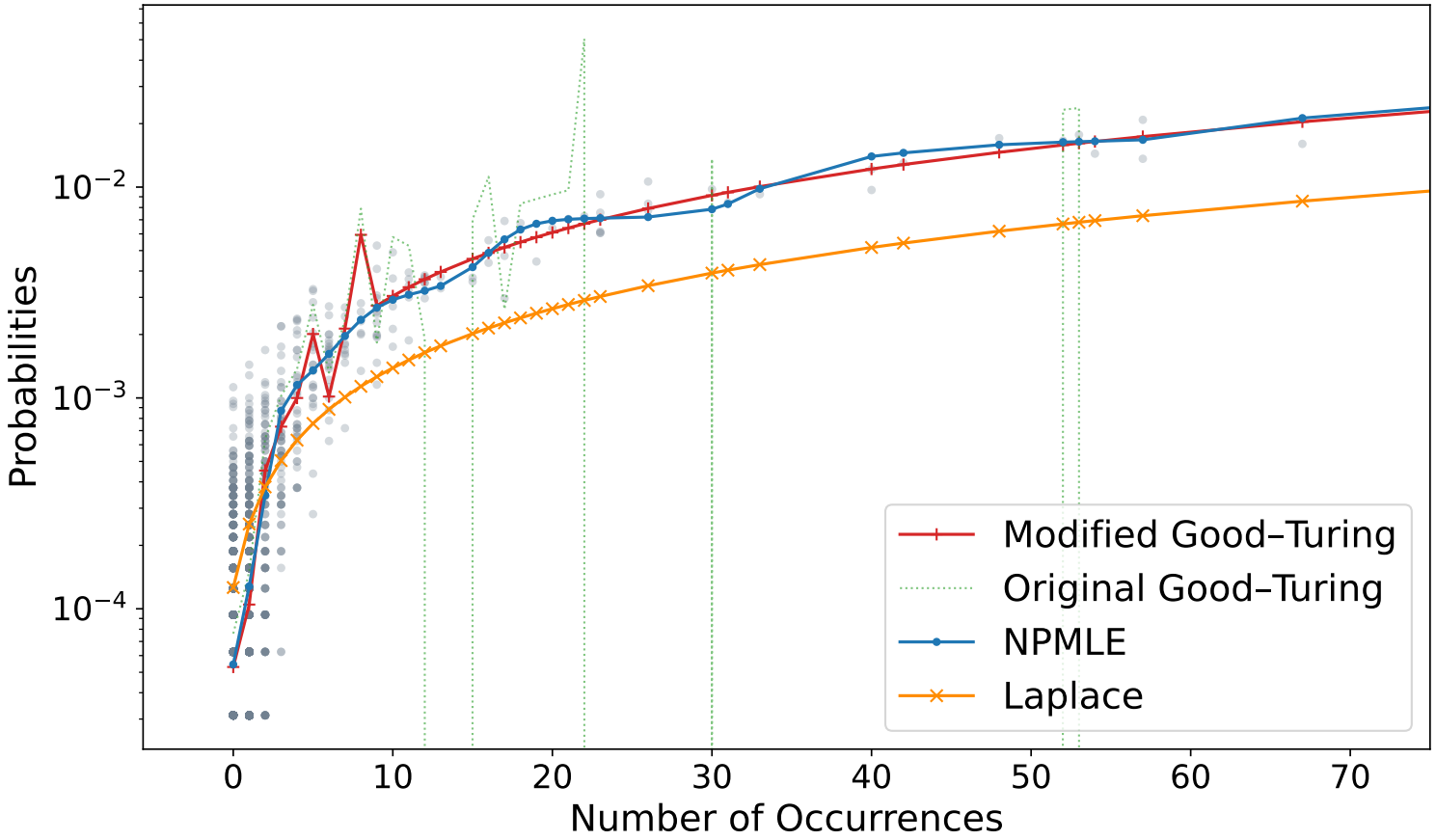
- $\rightarrow$ Minimax risk $\asymp \frac{k}{n}$ and $(\frac{1}{2} + o(1))\frac{k}{n}$ if $n \gg k$, but here $n \approx 2k/3$
- $\rightarrow$ Laplace is rate-optimal; Braess–Sauer achieves sharp constant $\frac{1}{2}$

Results

Estimator	KL Risk (bits)
Empirical frequency	∞
Original Good-Turing	∞
Laplace	0.6231
Braess-Sauer	0.4719
Modified Good-Turing	0.2732
New estimator (NPMLE)	0.2513
Oracle	*0.2462*

- → Worst-case risk is too crude to be a useful benchmark
  - → Minimax risk $\asymp \frac{k}{n}$ and $(\frac{1}{2} + o(1))\frac{k}{n}$ if $n \gg k$, but here $n \approx 2k/3$
  - → Laplace is rate-optimal; Braess–Sauer achieves sharp constant $\frac{1}{2}$
- → Need a *meaningful instance-dependent* benchmark – *oracle risk* (later)

True vs fitted word frequency

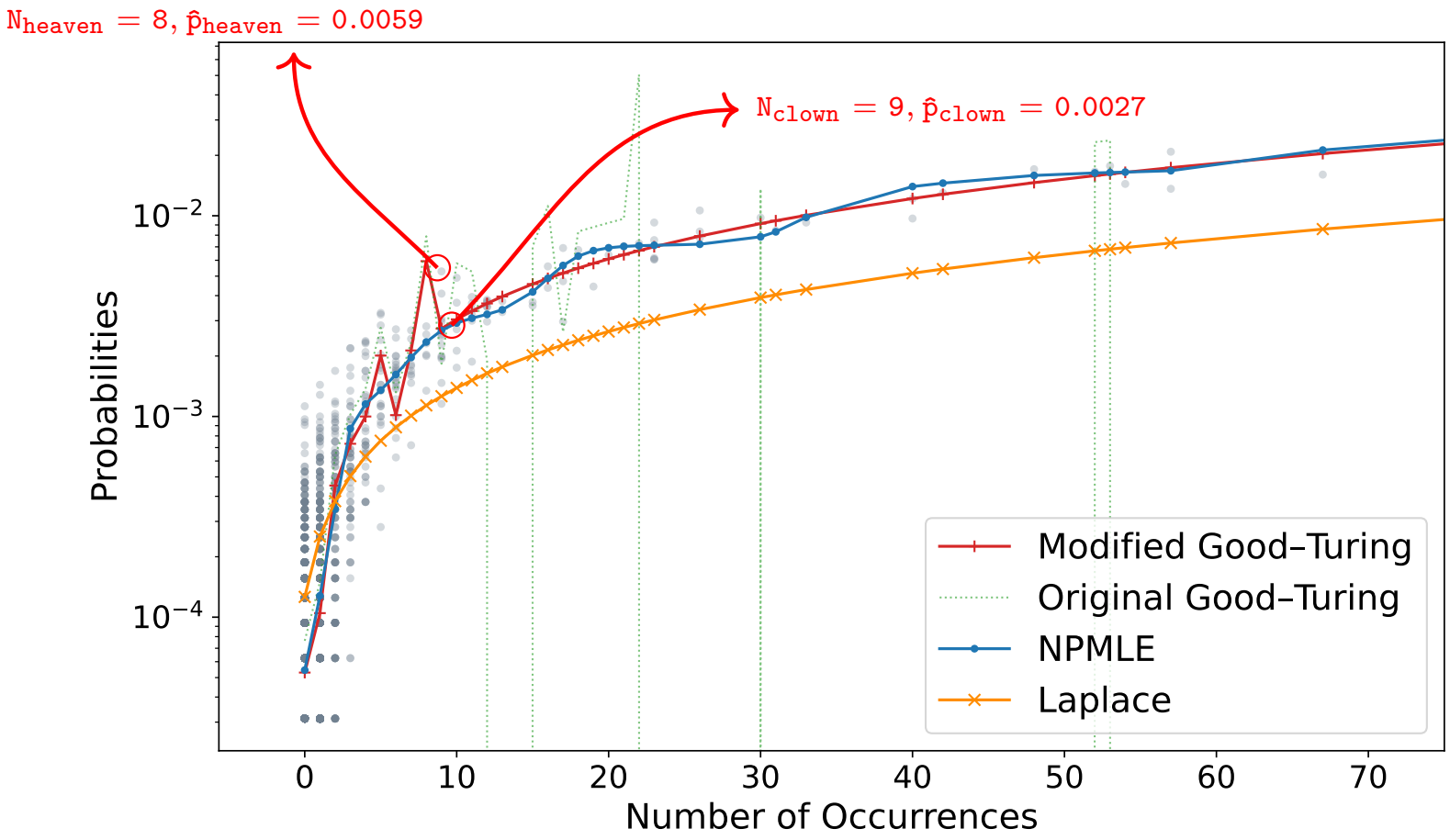


→ original GT is unstable

→ modified GT lacks interpretability (not monotone)

→ Laplace is over-discounted

True vs fitted word frequency



→ original GT is unstable

→ modified GT lacks interpretability (not monotone)

→ Laplace is over-discounted

Lessons so far

On the theory side:

- → instance-wise as opposed to minimax optimality
  - → competitive framework and oracle estimator
  - → adapt to latent structure in the data

Lessons so far

On the theory side:

- → instance-wise as opposed to minimax optimality
  - → competitive framework and oracle estimator
  - → adapt to latent structure in the data

On the practical side:

- → interpretability: positive, monotone, and smooth estimates
- → computational efficiency
- → tuning parameter free

Oracle and Proposed Approach

Oracle

Any reasonable estimator should be permutation invariant (PI), i.e., invariant to relabeling:

$$\widehat{p} \circ \pi = \pi \circ \widehat{p}, \qquad \text{for all permutation } \pi \in S_k$$

Oracle

Any reasonable estimator should be permutation invariant (PI), i.e., invariant to relabeling:

$$\widehat{p} \circ \pi = \pi \circ \widehat{p}, \qquad \text{for all permutation } \pi \in S_k$$

The best PI estimator (known as the PI oracle) knows the true distribution $p^\star$ but must be PI:

$$\widehat{p}^{\mathrm{PI}} = \operatorname*{argmin}_{\widehat{p}:\ \mathrm{PI}} \mathbb{E}[\mathrm{KL}(p^\star \| \widehat{p})]$$

Oracle

Any reasonable estimator should be permutation invariant (PI), i.e., invariant to relabeling:

$$\widehat{p} \circ \pi = \pi \circ \widehat{p}, \qquad \text{for all permutation } \pi \in S_k$$

The best PI estimator (known as the PI oracle) knows the true distribution $p^\star$ but must be PI:

$$\widehat{p}^{\mathrm{PI}} = \operatorname*{argmin}_{\widehat{p}:\ \mathrm{PI}} \mathbb{E}[\mathrm{KL}(p^\star \| \widehat{p})]$$

What is the PI oracle?

$$\begin{pmatrix} p_1^\star \\ \vdots \\ p_k^\star \end{pmatrix} \xrightarrow{\text{random permutation}} \begin{pmatrix} p_1 \\ \vdots \\ p_k \end{pmatrix} \xrightarrow{\text{statistical model}} \begin{pmatrix} N_1 \\ \vdots \\ N_k \end{pmatrix}$$

Mathematically,

$$\widehat{p}_i^{\mathrm{PI}} = \mathbb{E}_{\Pi^\star}[p_i \mid N^k],$$

Approximating the oracle

Recall that $\widehat{p}_i^{\mathsf{PI}} = \mathbb{E}_{\Pi^\star}[p_i \mid N^k]$ with $\Pi^\star$ being a random permutation of $(p_1^\star, \ldots, p_k^\star)$

$\rightarrow$ $\Pi^\star$ is high-dimensional, dependent, and unknown

Approximating the oracle

Recall that $\widehat{p}_i^{\mathsf{PI}} = \mathbb{E}_{\Pi^\star}[p_i \mid N^k]$ with $\Pi^\star$ being a random permutation of $(p_1^\star, \ldots, p_k^\star)$

$\rightarrow$ $\Pi^\star$ is high-dimensional, dependent, and unknown

Wishful thinking: mean-field approximation

$\rightarrow$ pretend *as if* $\Pi^\star$ *were* an i.i.d. distribution

$\rightarrow$ in that case, $\mathbb{E}_{\Pi^\star}[p_i \mid N^k] \approx \mathbb{E}_{G^\star}[p_i \mid N_i]$, where $G^\star = \frac{1}{k}\sum_{i=1}^k \delta_{p_j^\star}$ is the marginal distribution of $\Pi^\star$

Approximating the oracle

Recall that $\widehat{p}_i^{\mathsf{PI}} = \mathbb{E}_{\Pi^\star}[p_i \mid N^k]$ with $\Pi^\star$ being a random permutation of $(p_1^\star, \ldots, p_k^\star)$

$\rightarrow$ $\Pi^\star$ is high-dimensional, dependent, and unknown

Wishful thinking: mean-field approximation

$\rightarrow$ pretend *as if* $\Pi^\star$ *were* an i.i.d. distribution

$\rightarrow$ in that case, $\mathbb{E}_{\Pi^\star}[p_i \mid N^k] \approx \mathbb{E}_{G^\star}[p_i \mid N_i]$, where $G^\star = \frac{1}{k}\sum_{i=1}^k \delta_{p_j^\star}$ is the marginal distribution of $\Pi^\star$

This is empirical Bayes!

$\rightarrow$ we pretend that $p_1, \ldots, p_k \sim G^\star$ are drawn from a prior

$\rightarrow$ we aim to estimate $G^\star$ from histogram counts $(N_1, \ldots, N_k)$

Proposed algorithm

Step 1: Use NPMLE [Kiefer-Wolfowitz '56] to learn the *prior* $G^\star$:

$$\widehat{G} = \operatorname*{argmax}_{G \in \mathcal{P}(\mathbb{R}_+)} \sum_{i=1}^{k} \log f_G(N_i),$$

where f_G is the marginal pmf of N_i in the 1-D Bayes model $p_i \sim G, N_i \mid p_i \sim \mathrm{Poi}(np_i)$.

Proposed algorithm

Step 1: Use NPMLE [Kiefer-Wolfowitz '56] to learn the *prior* $G^\star$:

$$\widehat{G} = \operatorname*{argmax}_{G \in \mathcal{P}(\mathbb{R}_+)} \sum_{i=1}^{k} \log f_G(N_i),$$

where f_G is the marginal pmf of N_i in the 1-D Bayes model $p_i \sim G, N_i \mid p_i \sim \mathrm{Poi}(np_i)$.

Step 2: Apply the learned Bayes rule:

$$\widehat{p}_i^{\mathrm{NPMLE}} \propto \text{posterior mean of } p_i \text{ given } N_i \text{ under learned prior } p_i \sim \widehat{G}.$$

Proposed algorithm

Step 1: Use NPMLE [Kiefer-Wolfowitz '56] to learn the *prior* $G^\star$:

$$\widehat{G} = \operatorname*{argmax}_{G \in \mathcal{P}(\mathbb{R}_+)} \sum_{i=1}^{k} \log f_G(N_i),$$

where f_G is the marginal pmf of N_i in the 1-D Bayes model $p_i \sim G, N_i \mid p_i \sim \mathrm{Poi}(np_i)$.

Step 2: Apply the learned Bayes rule:

$$\widehat{p}_i^{\mathrm{NPMLE}} \propto \text{posterior mean of } p_i \text{ given } N_i \text{ under learned prior } p_i \sim \widehat{G}.$$

The final estimator is:

- $\rightarrow$ computationally cheap (solve by Frank–Wolfe [Lindsay'83] or discretization [Koenker-Mizera'14, Koenker-Gu'17])
- $\rightarrow$ free of tuning parameters
- $\rightarrow$ monotone, smooth, and positive, all thanks to Bayes form

Two Modeling Strategies for Empirical Bayes Estimation

Bradley Efron

Abstract. Empirical Bayes methods use the data from parallel experiments, for instance, observations $X_k \sim \mathcal{N}(\Theta_k, 1)$ for $k = 1, 2, \ldots, N$, to estimate the conditional distributions $\Theta_k | X_k$. There are two main estimation strategies: modeling on the θ space, called "g-modeling" here, and modeling on the x space, called "f-modeling." The two approaches are described and compared. A series of computational formulas are developed to assess their frequentist accuracy. Several examples, both contrived and genuine, show the strengths and limitations of the two strategies.

Two Modeling Strategies for Empirical Bayes Estimation

Bradley Efron

Abstract. Empirical Bayes methods use the data from parallel experiments, for instance, observations $X_k \sim \mathcal{N}(\Theta_k, 1)$ for $k = 1, 2, \ldots, N$, to estimate the conditional distributions $\Theta_k | X_k$. There are two main estimation strategies: modeling on the θ space, called "g-modeling" here, and modeling on the x space, called "f-modeling." The two approaches are described and compared. A series of computational formulas are developed to assess their frequentist accuracy. Several examples, both contrived and genuine, show the strengths and limitations of the two strategies.

Good-Turing

Proposed approach

Theory

Competitive framework

→ Introduced by Orlitsky–Suresh in their 2015 NIPS best paper:

Competitive Distribution Estimation: Why is Good-Turing Good

Alon Orlitsky
UC San Diego
alon@ucsd.edu

Ananda Theertha Suresh
UC San Diego
asuresh@ucsd.edu

Competitive framework

→ Introduced by Orlitsky–Suresh in their 2015 NIPS best paper:

Competitive Distribution Estimation: Why is Good-Turing Good

Alon Orlitsky
UC San Diego
alon@ucsd.edu

Ananda Theertha Suresh
UC San Diego
asuresh@ucsd.edu

→ Regret = excess risk over the PI oracle

$$\mathrm{Reg}(\widehat{p}) = \sup_{p^\star} \mathbb{E}\Big[\mathrm{KL}(p^\star \| \widehat{p}) - \mathrm{KL}(p^\star \| \widehat{p}^{\mathrm{PI}})\Big].$$

Competitive framework

→ Introduced by Orlitsky–Suresh in their 2015 NIPS best paper:

Competitive Distribution Estimation: Why is Good-Turing Good

Alon Orlitsky
UC San Diego
alon@ucsd.edu

Ananda Theertha Suresh
UC San Diego
asuresh@ucsd.edu

→ Regret = excess risk over the PI oracle

$$\mathrm{Reg}(\widehat{p}) = \sup_{p^\star} \mathbb{E}\Big[\mathrm{KL}(p^\star \| \widehat{p}) - \mathrm{KL}(p^\star \| \widehat{p}^{\mathrm{PI}})\Big].$$

→ Regret compares the risk of an algo with the oracle risk on an *instance-dependent* basis, going beyond the (often conservative) minimax framework

"Why is Good-Turing good" [Orlitsky-Suresh '15]

Upper bound

A modified Good–Turing estimator $\widehat{p}^{\mathrm{MGT}}$ achieves

$$\mathrm{Reg}(\widehat{p}^{\mathrm{MGT}}) = \widetilde{O}\left(\min\left\{\frac{k}{n}, \frac{1}{\sqrt{n}}\right\}\right)$$

where $\widetilde{O}$ hides logarithmic factors.

"Why is Good-Turing good" [Orlitsky-Suresh '15]

Upper bound

A modified Good–Turing estimator $\widehat{p}^{\mathrm{MGT}}$ achieves

$$\mathrm{Reg}(\widehat{p}^{\mathrm{MGT}}) = \widetilde{O}\left(\min\left\{\frac{k}{n}, \frac{1}{\sqrt{n}}\right\}\right)$$

where $\widetilde{O}$ hides logarithmic factors.

Lower bound

For any estimator $\widehat{p}$,

$$\mathrm{Reg}(\widehat{p}) = \Omega\left(\min\left\{\frac{k}{n}, \frac{1}{n^{2/3}}\right\}\right)$$

"Why is Good-Turing good" [Orlitsky-Suresh '15]

Upper bound

A modified Good–Turing estimator $\widehat{p}^{\mathrm{MGT}}$ achieves

$$\mathrm{Reg}(\widehat{p}^{\mathrm{MGT}}) = \widetilde{O}\left(\min\left\{\frac{k}{n}, \frac{1}{\sqrt{n}}\right\}\right)$$

where $\widetilde{O}$ hides logarithmic factors.

Lower bound

For any estimator $\widehat{p}$,

$$\mathrm{Reg}(\widehat{p}) = \Omega\left(\min\left\{\frac{k}{n}, \frac{1}{n^{2/3}}\right\}\right)$$

→ GT is highly competitive, with uniformly vanishing regret even if $k \gg n$

→ Open question: Is it possible to outcompete GT?

Main theory

Theorem (H.–Niles-Weed–Shen–Wu '25)

→ *Competitive optimality of NPMLE:*

$$\mathrm{Reg}(\widehat{p}^{\mathrm{NPMLE}}) = \widetilde{O}\left(\min\left\{\frac{k}{n}, \frac{1}{n^{2/3}}\right\}\right)$$

meeting the information-theoretic lower bound.

Main theory

Theorem (H.–Niles-Weed–Shen–Wu '25)

→ *Competitive optimality of NPMLE:*

$$\mathrm{Reg}(\widehat{p}^{\mathrm{NPMLE}}) = \widetilde{O}\left(\min\left\{\frac{k}{n}, \frac{1}{n^{2/3}}\right\}\right)$$

meeting the information-theoretic lower bound.

→ *Competitive suboptimality of Good–Turing: regardless of tuning parameters*

$$\mathrm{Reg}(\widehat{p}^{\mathrm{MGT}}) = \widetilde{\Omega}\left(\min\left\{\frac{k}{n}, \frac{1}{\sqrt{n}}\right\}\right).$$

Theorem (H.–Niles-Weed–Shen–Wu '25)

→ *Competitive optimality of NPMLE:*

$$\mathrm{Reg}(\widehat{p}^{\mathrm{NPMLE}}) = \widetilde{O}\left(\min\left\{\frac{k}{n}, \frac{1}{n^{2/3}}\right\}\right)$$

meeting the information-theoretic lower bound.

→ *Competitive suboptimality of Good–Turing: regardless of tuning parameters*

$$\mathrm{Reg}(\widehat{p}^{\mathrm{MGT}}) = \widetilde{\Omega}\left(\min\left\{\frac{k}{n}, \frac{1}{\sqrt{n}}\right\}\right).$$

Remarks:

→ NPMLE is competitively optimal, while Good–Turing is not

→ NPMLE frequently beats Good–Turing in practice

One bit of proof

$$\begin{pmatrix} \theta_1^\star \\ \vdots \\ \theta_k^\star \end{pmatrix} \xrightarrow{\text{random permutation } \pi} \begin{pmatrix} \theta_1 \\ \vdots \\ \theta_k \end{pmatrix}$$

$\rightarrow$ clearly $(\theta_1, \ldots, \theta_k)$ are far from independent

One bit of proof

$$\begin{pmatrix}\theta_1^\star\\ \vdots\\ \theta_k^\star\end{pmatrix} \xrightarrow{\text{random permutation } \pi} \begin{pmatrix}\theta_1\\ \vdots\\ \theta_k\end{pmatrix} \xrightarrow{\text{noisy channel } P_\theta(\cdot)} \begin{pmatrix}N_1\\ \vdots\\ N_k\end{pmatrix}$$

$\rightarrow$ clearly $(\theta_1, \ldots, \theta_k)$ are far from independent

$\rightarrow$ main observation: their noisy version are close to independent [H.–Niles-Weed '24, H.–Liang '25]:

$$\text{dist}\left(\underbrace{\frac{1}{k!}\sum_{\pi\in S_k}\prod_{i=1}^{k} P_{\theta^\star_{\pi(i)}}}_{\text{Law}(N_1,\ldots,N_k)}, \underbrace{\left(\frac{1}{k}\sum_{i=1}^{k} P_{\theta_i^\star}\right)^{\otimes k}}_{\text{Law}(N_1)\otimes\cdots\otimes\text{Law}(N_k)} \right) \text{ is small independent of dimension}$$

One bit of proof

$$\begin{pmatrix}\theta_1^\star\\ \vdots\\ \theta_k^\star\end{pmatrix} \xrightarrow{\text{random permutation } \pi} \begin{pmatrix}\theta_1\\ \vdots\\ \theta_k\end{pmatrix} \xrightarrow{\text{noisy channel } P_\theta(\cdot)} \begin{pmatrix}N_1\\ \vdots\\ N_k\end{pmatrix}$$

$\to$ clearly $(\theta_1, \ldots, \theta_k)$ are far from independent

$\to$ main observation: their noisy version are close to independent [H.–Niles-Weed '24, H.–Liang '25]:

$$\text{dist}\left(\underbrace{\frac{1}{k!}\sum_{\pi\in S_k}\prod_{i=1}^k P_{\theta^\star_{\pi(i)}}}_{\text{Law}(N_1,\ldots,N_k)}, \underbrace{\left(\frac{1}{k}\sum_{i=1}^k P_{\theta_i^\star}\right)^{\otimes k}}_{\text{Law}(N_1)\otimes\cdots\otimes\text{Law}(N_k)} \right) \text{ is small independent of dimension}$$

$\to$ mean-field approximation error [H.–Niles-Weed–Shen–Wu '25]:

$$\text{dist}(\text{mean}(P_{\theta_1|N_1}), \text{mean}(P_{\theta_1|N^k})) \le \text{dist}(\underbrace{P_{\widetilde{\theta}_1|N_1}, P_{\widetilde{\theta}_1|N^k}}_{\text{noisy interpolation}}) \ll \text{dist}(P_{\theta_1|N_1}, P_{\theta_1|N^k})$$

Experiments

A synthetic experiment

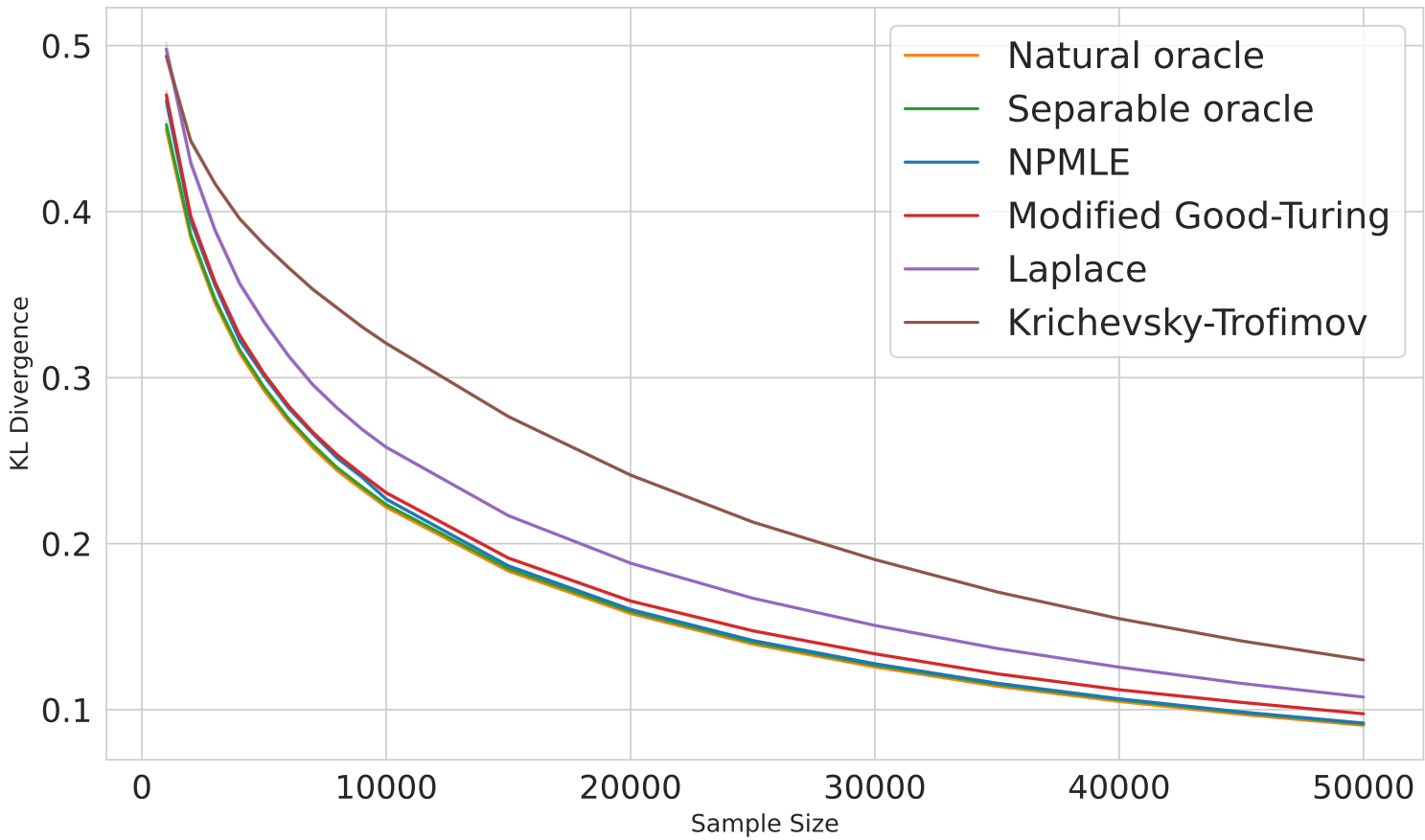
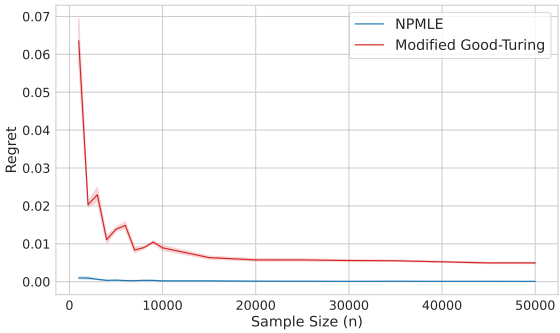
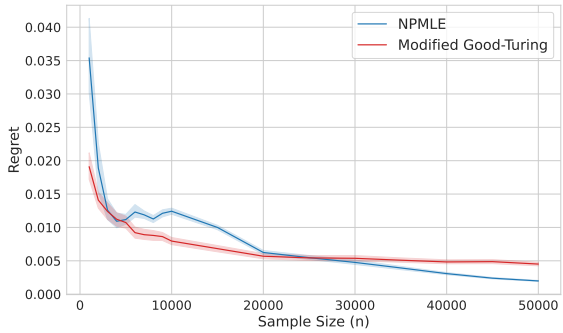


Figure: KL risks for $p_i^\star \propto \sqrt{|z_i|}$ with $z_i \sim$ i.i.d. Cauchy and $k = 10,000$

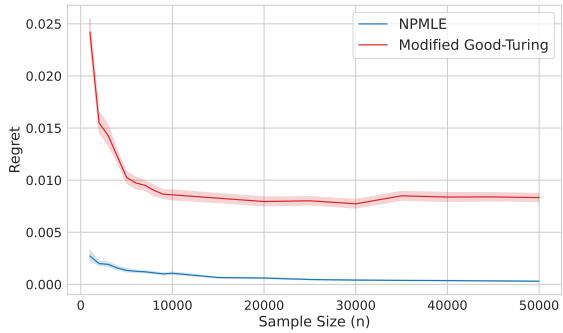
More synthetic distributions



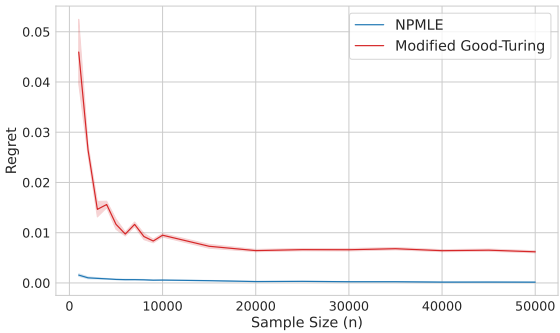
(a) Uniform



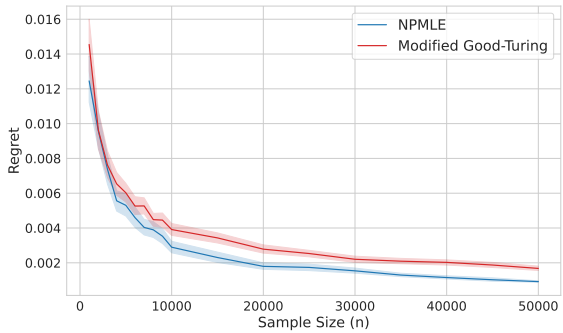
(b) Zipf ($\alpha = 1$).



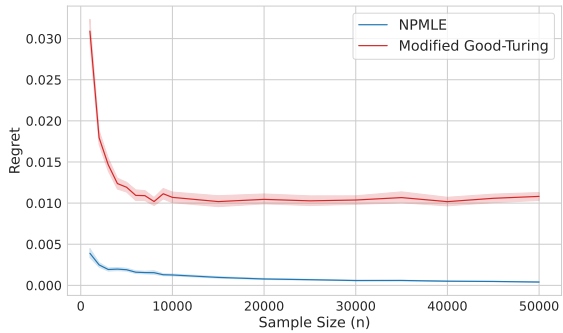
(c) Dirichlet ($c = 1$)



(d) Step.



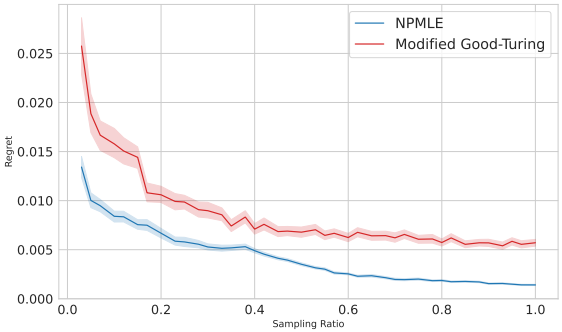
(e) Zipf ($\alpha = 1.5$).



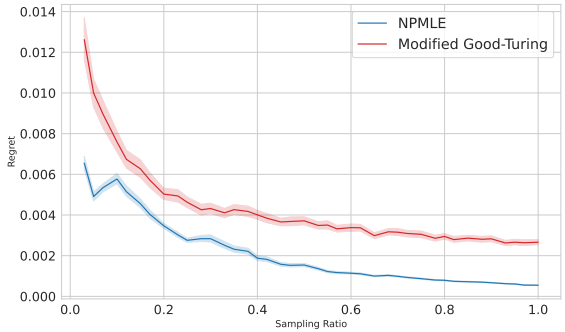
(f) Dirichlet ($c = 0.5$)

Figure: KL regrets for synthetic distributions.

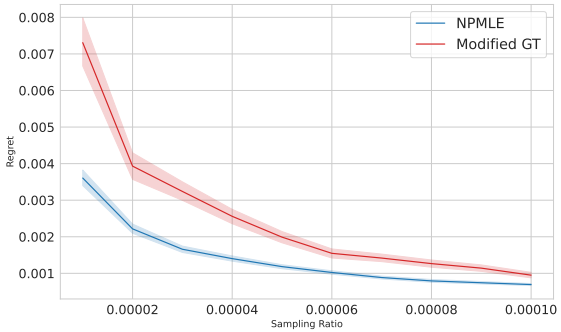
More real data



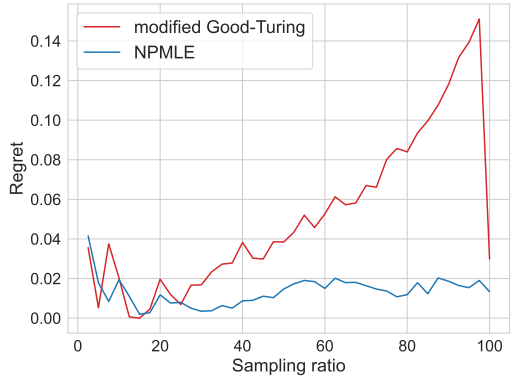
(a) Hamlet (random).



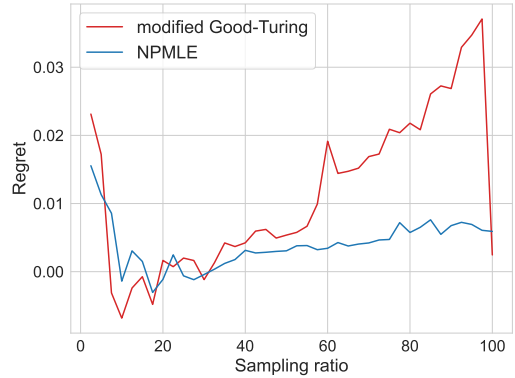
(b) LOTR (random).



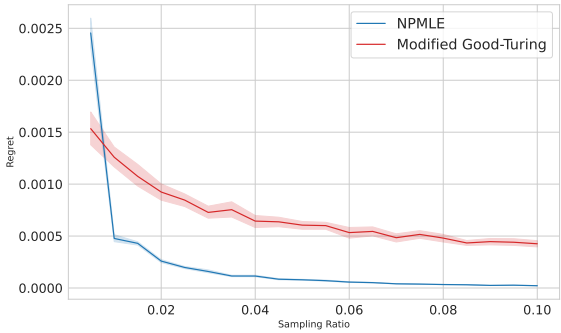
(c) 2020 Census Detailed DHC-A.



(d) Hamlet (consecutive).



(e) LOTR (consecutive).

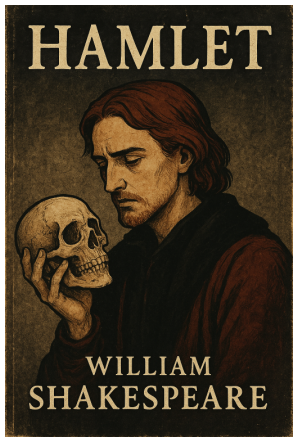


(f) 2010 Census surname.

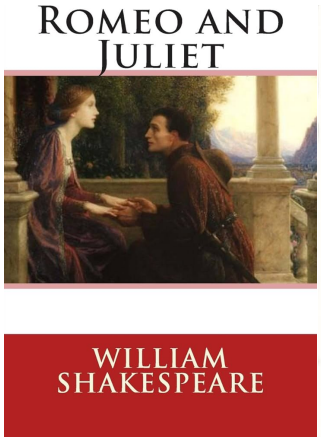
Figure: KL regrets for real data.

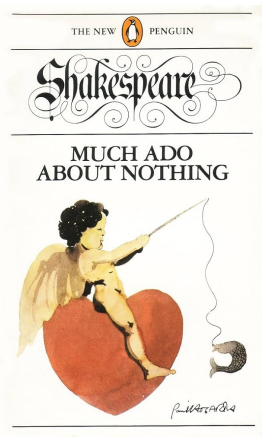
Out-of-sample experiment

Train

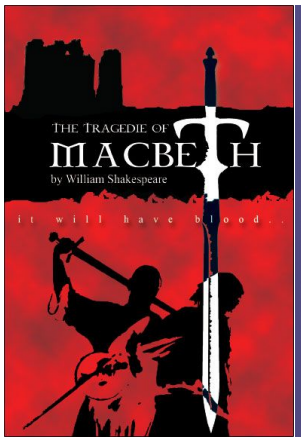


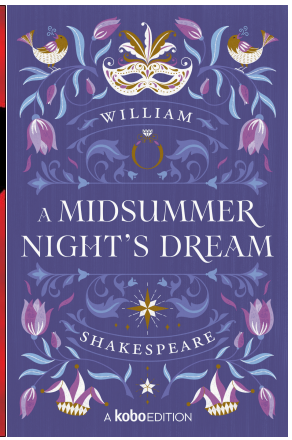
Test





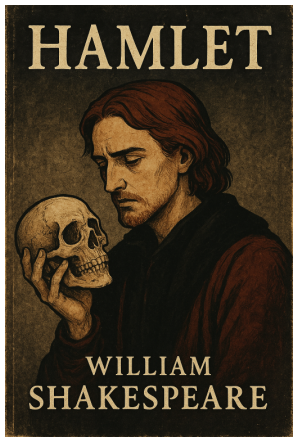
. . .



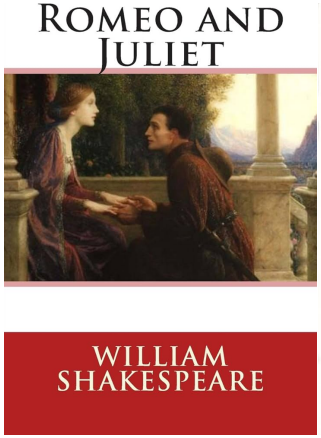


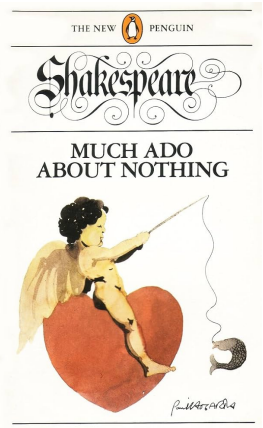
Out-of-sample experiment

Train

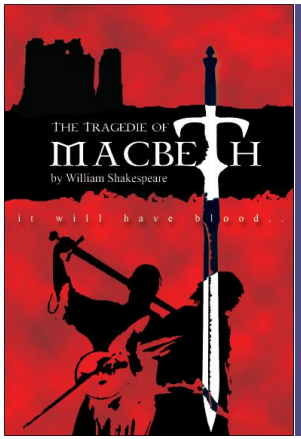


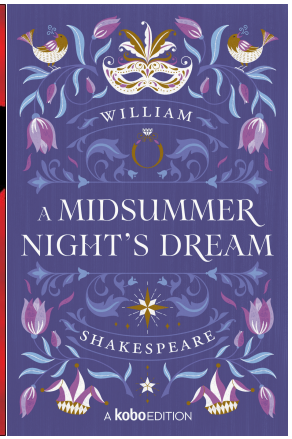
Test





. . .

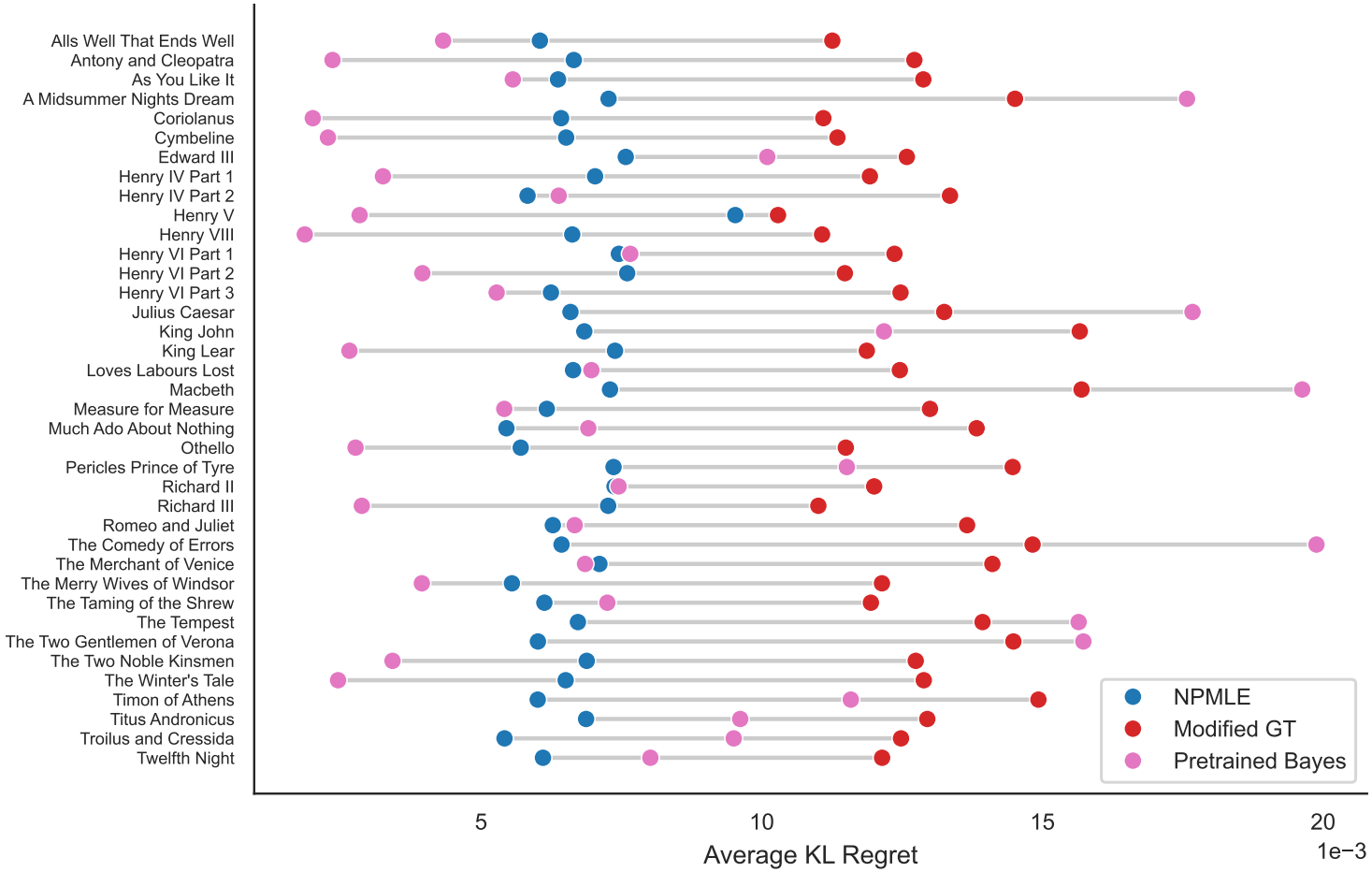




→ Train: compute NPMLE $\widehat{G}$ on the entire Hamlet

→ Test: use this *pretrained Bayes estimator* to estimate word frequency of other 38 Shakespearean plays (sampling ratio 20%)

Out-of-sample experiment



→ Pretrained Bayes beats modified Good–Turing **32** out of 38 times

→ Even beats in-sample NPMLE **18** times

Out-of-sample experiment

Text name	Modified GT	NPMLE	Pretrained Bayes
Alls Well That Ends Well	0.0113	0.0060	0.0043
Antony and Cleopatra	0.0127	0.0067	0.0024
As You Like It	0.0129	0.0064	0.0056
⋮	⋮	⋮	⋮
Twelfth Night	0.0121	0.0061	0.0080

Out-of-sample experiment

Text name	Modified GT	NPMLE	Pretrained Bayes
Alls Well That Ends Well	0.0113	0.0060	0.0043
Antony and Cleopatra	0.0127	0.0067	0.0024
As You Like It	0.0129	0.0064	0.0056
⋮	⋮	⋮	⋮
Twelfth Night	0.0121	0.0061	0.0080
Fellowship of the Ring	0.0912	0.0897	0.2701

→ Bayes pretrained on *Hamlet* does not generalize to *Fellowship of the Ring*

→ This suggests the learned $\widehat{G}$ captures useful stylistic information specific to a corpus (e.g. vocabulary profile [Stamatatos '09])

→ There is a theoretical foundation for pretrained Bayes! (Part II)

Part II: Solving EB via Pretraining

N. Cannella, Y. Han, Y. Polyanskiy, and A. Teh

Universal priors: solving empirical Bayes via Bayesian inference and pretraining

arXiv:2602.15136

Pretraining

Previous approach to approximating $\mathbb{E}_{\Pi^\star}[\theta^n \mid X^n]$:

$\rightarrow$ approximate $\Pi^\star$ by an i.i.d. distribution

$\rightarrow$ estimate the marginal of $\Pi^\star$ from X^n

Pretraining

Previous approach to approximating $\mathbb{E}_{\Pi^\star}[\theta^n \mid X^n]$:

$\rightarrow$ approximate $\Pi^\star$ by an i.i.d. distribution

$\rightarrow$ estimate the marginal of $\Pi^\star$ from X^n

A radical proposal: use $\mathbb{E}_{\Pi}[\theta^n \mid X^n]$ for a fixed (data-independent) prior Π

$\rightarrow$ the dream: fix a prior once and for all so that Bayesian inference itself solves EB

Pretraining

Previous approach to approximating $\mathbb{E}_{\Pi^\star}[\theta^n \mid X^n]$:

$\rightarrow$ approximate $\Pi^\star$ by an i.i.d. distribution

$\rightarrow$ estimate the marginal of $\Pi^\star$ from X^n

A radical proposal: use $\mathbb{E}_{\Pi}[\theta^n \mid X^n]$ for a fixed (data-independent) prior Π

$\rightarrow$ the dream: fix a prior once and for all so that Bayesian inference itself solves EB

Computation of $\mathbb{E}_{\Pi}[\theta^n \mid X^n]$ via pretraining: generate synthetic training batches $(\theta^{n,(1)}, X^{n,(1)}), \ldots, (\theta^{n,(M)}, X^{n,(M)}) \sim \Pi$ for $M \to \infty$, and compute the ERM

$$\operatorname*{argmin}_{f \in \mathcal{F}} \frac{1}{M} \sum_{m=1}^{M} \|\theta^{n,(m)} - f(X^{n,(m)})\|_2^2$$

$\rightarrow$ $\mathcal{F}$ is chosen to be the class of transformers: sequence-to-sequence map, strong expressive power, and PI

TabPFN: a pretrained transformer on synthetic data

Article

Accurate predictions on small data with a tabular foundation model

https://doi.org/10.1038/s41586-024-08328-6

Received: 17 May 2024

Accepted: 31 October 2024

Published online: 8 January 2025

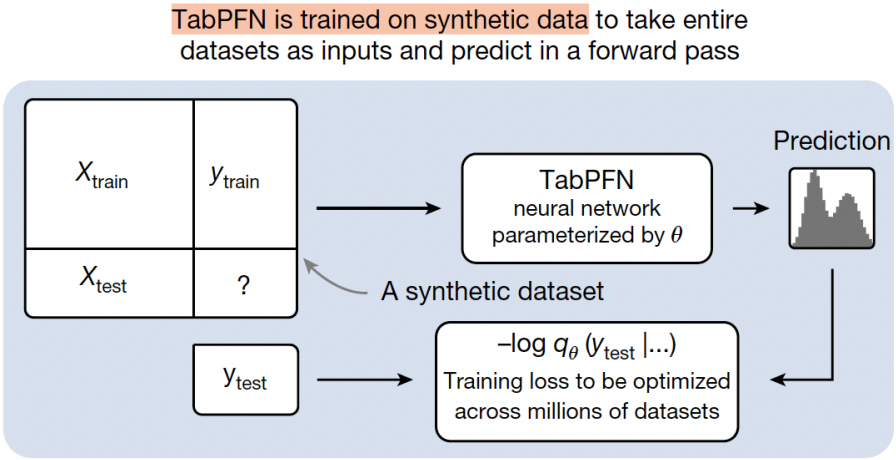
Open access

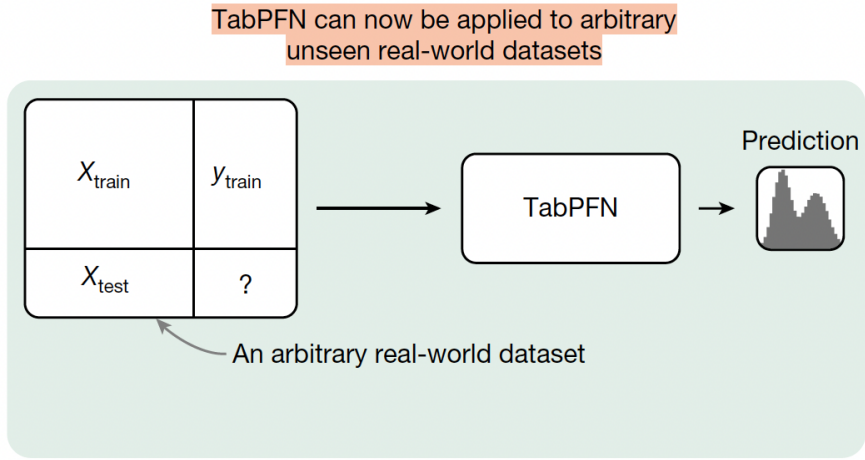
Check for updates

Noah Hollmann[1,2,3,7]✉, Samuel Müller[1,7]✉, Lennart Purucker[1], Arjun Krishnakumar[1], Max Körfer[1], Shi Bin Hoo[1], Robin Tibor Schirrmeister[4,5] & Frank Hutter[1,3,6]✉

Tabular data, spreadsheets organized in rows and columns, are ubiquitous across scientific fields, from biomedicine to particle physics to economics and climate science[1,2]. The fundamental prediction task of filling in missing values of a label column based on the rest of the columns is essential for various applications as diverse as biomedical risk models, drug discovery and materials science. Although deep learning has revolutionized learning from raw data and led to numerous high-profile success stories[3–5], gradient-boosted decision trees[6–9] have dominated tabular data for the past 20 years. Here we present the Tabular Prior-data Fitted Network (TabPFN), a tabular foundation model that outperforms all previous methods on datasets with up to 10,000 samples by a wide margin, using substantially less training time. In 2.8 s, TabPFN outperforms an ensemble of the strongest baselines tuned for 4 h in a classification setting. As a generative transformer-based foundation model, this model also allows fine-tuning, data generation, density estimation and learning reusable embeddings. TabPFN is a learning algorithm that is itself learned across millions of synthetic datasets, demonstrating the power of this approach for algorithm development. By improving modelling abilities across diverse fields, TabPFN has the potential to accelerate scientific discovery and enhance important decision-making in various domains.

TabPFN generalizes to unseen test datasets





Evidence for EB pretraining

Numerical experiments for Poisson EB problems [Teh-Jabbour-Polyanskiy '25]:

Dataset	Robbins	ERM	NPMLE	T24r	L24r
NHL (all)	-30.55 ± 6.55	1.46 ± 0.65	3.21 ± 0.92	3.46 ± 0.88	$\mathbf{3.96 \pm 1.12}$
NHL (defender)	-19.54 ± 6.35	3.19 ± 1.32	6.48 ± 1.63	6.91 ± 1.71	$\mathbf{7.54 \pm 2.04}$
NHL (center)	-49.89 ± 10.36	0.38 ± 0.82	3.44 ± 0.94	4.02 ± 0.99	$\mathbf{4.32 \pm 1.22}$
NHL (winger)	-42.63 ± 7.58	0.76 ± 0.69	3.06 ± 0.87	3.44 ± 0.89	$\mathbf{3.76 \pm 0.99}$
MLB (batting)	-59.66 ± 5.88	0.23 ± 0.18	1.25 ± 0.18	$\mathbf{1.50 \pm 0.16}$	1.44 ± 0.17
MLB (pitching)	-40.81 ± 3.16	0.09 ± 0.14	1.21 ± 0.19	$\mathbf{1.42 \pm 0.18}$	1.38 ± 0.17
BookCorpusOpen	-4.58 ± 0.43	9.38 ± 0.10	10.82 ± 0.11	9.43 ± 0.12	$\mathbf{11.23 \pm 0.21}$

Table: 95% confidence interval of the percentage improvement of RMSE over MLE.

Evidence for EB pretraining

Numerical experiments for Poisson EB problems [Teh-Jabbour-Polyanskiy '25]:

Dataset	Robbins	ERM	NPMLE	T24r	L24r
NHL (all)	-30.55 ± 6.55	1.46 ± 0.65	3.21 ± 0.92	3.46 ± 0.88	$\mathbf{3.96 \pm 1.12}$
NHL (defender)	-19.54 ± 6.35	3.19 ± 1.32	6.48 ± 1.63	6.91 ± 1.71	$\mathbf{7.54 \pm 2.04}$
NHL (center)	-49.89 ± 10.36	0.38 ± 0.82	3.44 ± 0.94	4.02 ± 0.99	$\mathbf{4.32 \pm 1.22}$
NHL (winger)	-42.63 ± 7.58	0.76 ± 0.69	3.06 ± 0.87	3.44 ± 0.89	$\mathbf{3.76 \pm 0.99}$
MLB (batting)	-59.66 ± 5.88	0.23 ± 0.18	1.25 ± 0.18	$\mathbf{1.50 \pm 0.16}$	1.44 ± 0.17
MLB (pitching)	-40.81 ± 3.16	0.09 ± 0.14	1.21 ± 0.19	$\mathbf{1.42 \pm 0.18}$	1.38 ± 0.17
BookCorpusOpen	-4.58 ± 0.43	9.38 ± 0.10	10.82 ± 0.11	9.43 ± 0.12	$\mathbf{11.23 \pm 0.21}$

Table: 95% confidence interval of the percentage improvement of RMSE over MLE.

→ pretrained transformers (T24r and L24r) outperform the NPMLE

→ pretrained transformers offer $\sim 100\times$ speedup at inference time compared with the NPMLE

Why Pretraining Works

The statistical question

Statistical question

→ Why does a pretrained estimator under a fixed training distribution work for arbitrary test distributions?

→ Specializing to EB settings, why can $\mathbb{E}_{\Pi^\star}[\theta^n \mid X^n] \approx \mathbb{E}_{\Pi}[\theta^n \mid X^n]$ hold universally for all i.i.d. priors $\Pi^\star$?

The statistical question

Statistical question

→ Why does a pretrained estimator under a fixed training distribution work for arbitrary test distributions?

→ Specializing to EB settings, why can $\mathbb{E}_{\Pi^\star}[\theta^n \mid X^n] \approx \mathbb{E}_{\Pi}[\theta^n \mid X^n]$ hold universally for all i.i.d. priors $\Pi^\star$?

A Poisson EB sandbox:

→ mean parameters: $\theta_1, \ldots, \theta_n \sim G^\star$ with an unknown prior $G^\star$

→ observations: independent $X_i \sim \mathrm{Poi}(\theta_i)$

→ for any estimator $\widehat{\theta}^n(X^n)$, the regret is defined as

$$\mathrm{Reg}(\widehat{\theta}^n; G^\star) = \frac{1}{n}\mathbb{E}[\|\widehat{\theta}^n - \text{Bayes estimator under } G^\star\|_2^2]$$

A hierarchical Bayes model

Choice of training prior Π:

$\rightarrow$ dependence is necessary: Π cannot be a product distribution

$\rightarrow$ a natural candidate: $\Pi = \mathbb{E}_{G\sim\pi}[G^{\otimes n}]$, with a prior-on-prior (PoP) π

A hierarchical Bayes model

Choice of training prior Π:

$\rightarrow$ dependence is necessary: Π cannot be a product distribution

$\rightarrow$ a natural candidate: $\Pi = \mathbb{E}_{G\sim\pi}[G^{\otimes n}]$, with a prior-on-prior (PoP) π

The PoP π induces a hierarchical Bayes model:

$$\begin{aligned} G &\sim \pi, \\ \theta_1, \dots, \theta_n \mid G &\overset{i.i.d.}{\sim} G, \\ X_i \mid \theta_1, \dots, \theta_n, G &\overset{ind.}{\sim} \mathrm{Poi}(\theta_i). \end{aligned}$$

Existence of universal PoP

Theorem (Least favorable PoP)

There exists a PoP $\pi^\star$ such that

$$\sup_{G^\star} \mathrm{Reg}(\mathbb{E}_{\pi^\star}[\theta^n|X^n]; G^\star) = \inf_{\widehat{\theta}^n} \sup_{G^\star} \mathrm{Reg}(\widehat{\theta}^n; G^\star).$$

$\rightarrow$ "universal" PoP exists!

Existence of universal PoP

Theorem (Least favorable PoP)

There exists a PoP $\pi^\star$ such that

$$\sup_{G^\star} \mathrm{Reg}(\mathbb{E}_{\pi^\star}[\theta^n|X^n]; G^\star) = \inf_{\widehat{\theta}^n} \sup_{G^\star} \mathrm{Reg}(\widehat{\theta}^n; G^\star).$$

$\rightarrow$ "universal" PoP exists!

Finding an explicit $\pi^\star$ might still be hard:

$\rightarrow$ in statistics, finding least favorable priors/minimax estimators is often challenging

$\rightarrow$ ML approaches for building near-minimax estimators in statistics typically require solving a min-max game [Gupta et al. '20]

Many PoPs are near universal

Theorem (Regret bound)

For a PoP π,

$$\sup_{G^\star} \mathrm{Reg}(\mathbb{E}_\pi[\theta^n|X^n]; G^\star) \le \frac{1}{n}\Big(\textit{metric entropy of model} + \textit{prior coverage of } \pi\Big)$$

Many PoPs are near universal

Theorem (Regret bound)

For a PoP π,

$$\sup_{G^\star} \mathrm{Reg}(\mathbb{E}_\pi[\theta^n|X^n]; G^\star) \le \frac{1}{n}\Big(\textit{metric entropy of model} + \textit{prior coverage of } \pi\Big)$$

$\rightarrow$ usually, both terms are $\widetilde{O}(1)$, so that the regret is $\widetilde{O}(\frac{1}{n})$ (near optimal)

$\rightarrow$ this regret bound holds for <u>any</u> test prior $G^\star$

Many PoPs are near universal

Theorem (Regret bound)

For a PoP π,

$$\sup_{G^\star} \mathrm{Reg}(\mathbb{E}_\pi[\theta^n|X^n]; G^\star) \le \frac{1}{n}\Big(\textit{metric entropy of model} + \textit{prior coverage of } \pi\Big)$$

$\rightarrow$ usually, both terms are $\widetilde{O}(1)$, so that the regret is $\widetilde{O}(\frac{1}{n})$ (near optimal)

$\rightarrow$ this regret bound holds for <u>any</u> test prior $G^\star$

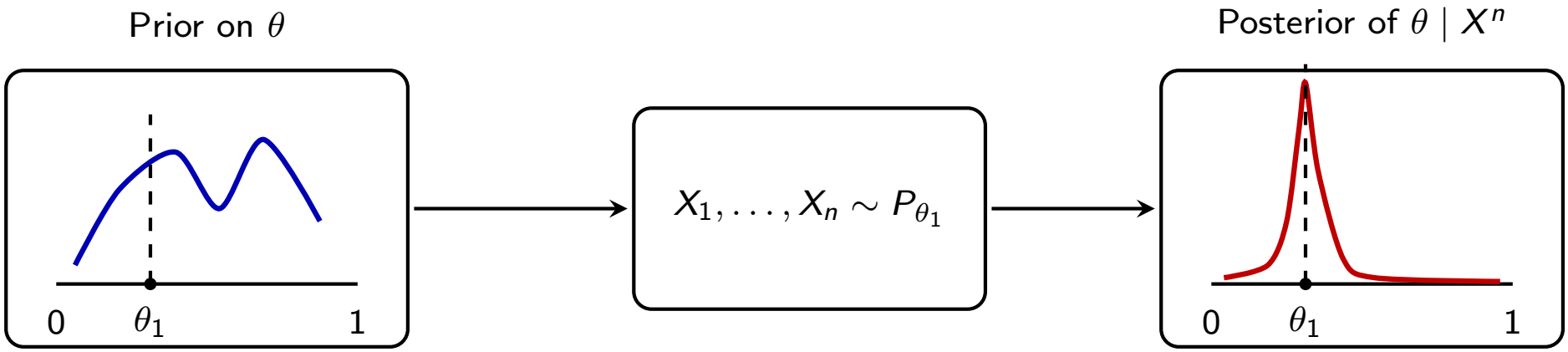
An example PoP with desired coverage: $G = \sum_{i=1}^m w_i \delta_{a_i} \sim \pi$, with

$\rightarrow$ uniform atom locations $a_1, \ldots, a_m \sim \mathrm{Unif}([0, A])$

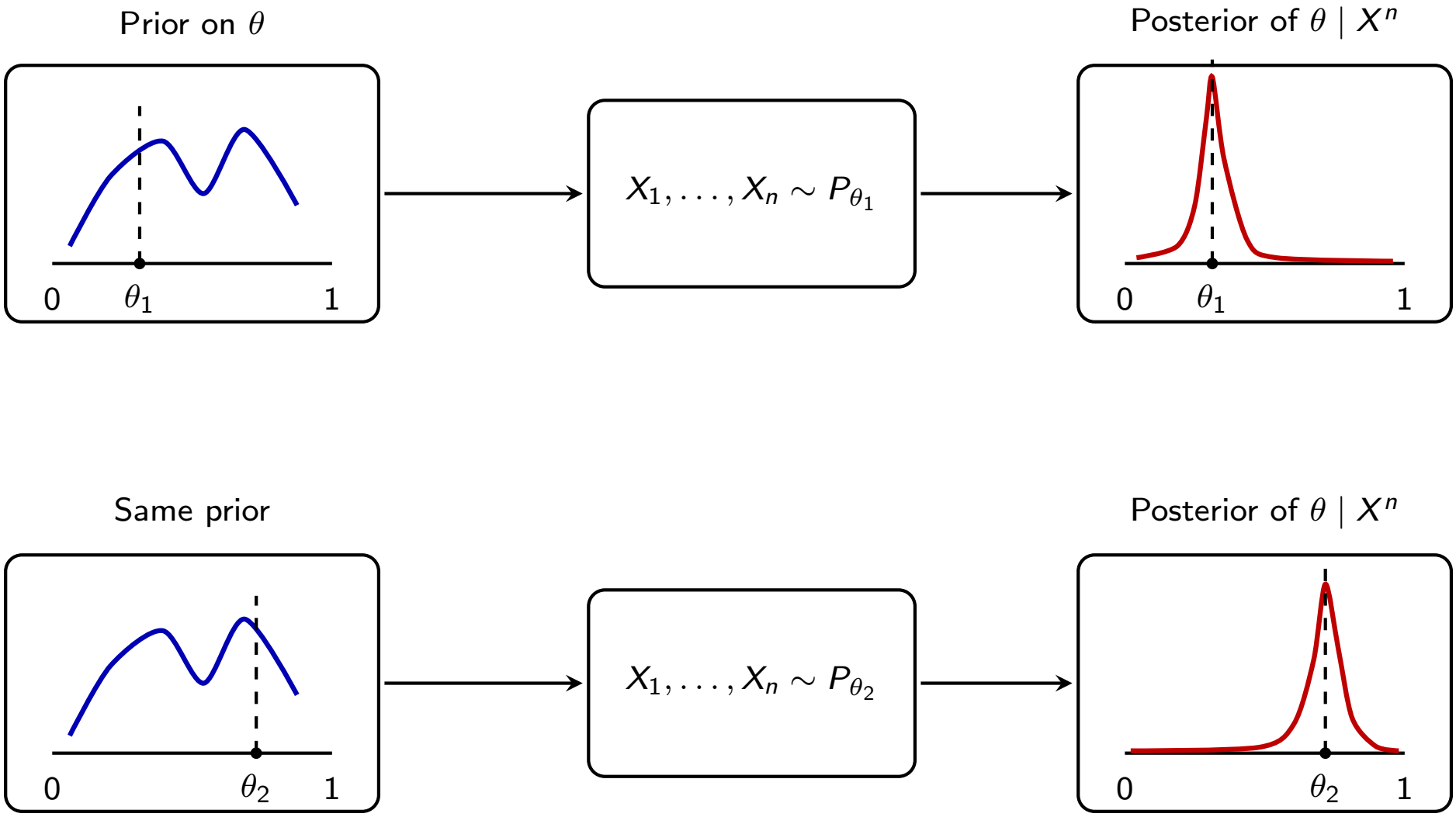
$\rightarrow$ uniform atom weights $(w_1, \ldots, w_m) \sim \mathrm{Dirichlet}(1, \ldots, 1)$

$\rightarrow$ number of atoms $m = \widetilde{O}(1)$

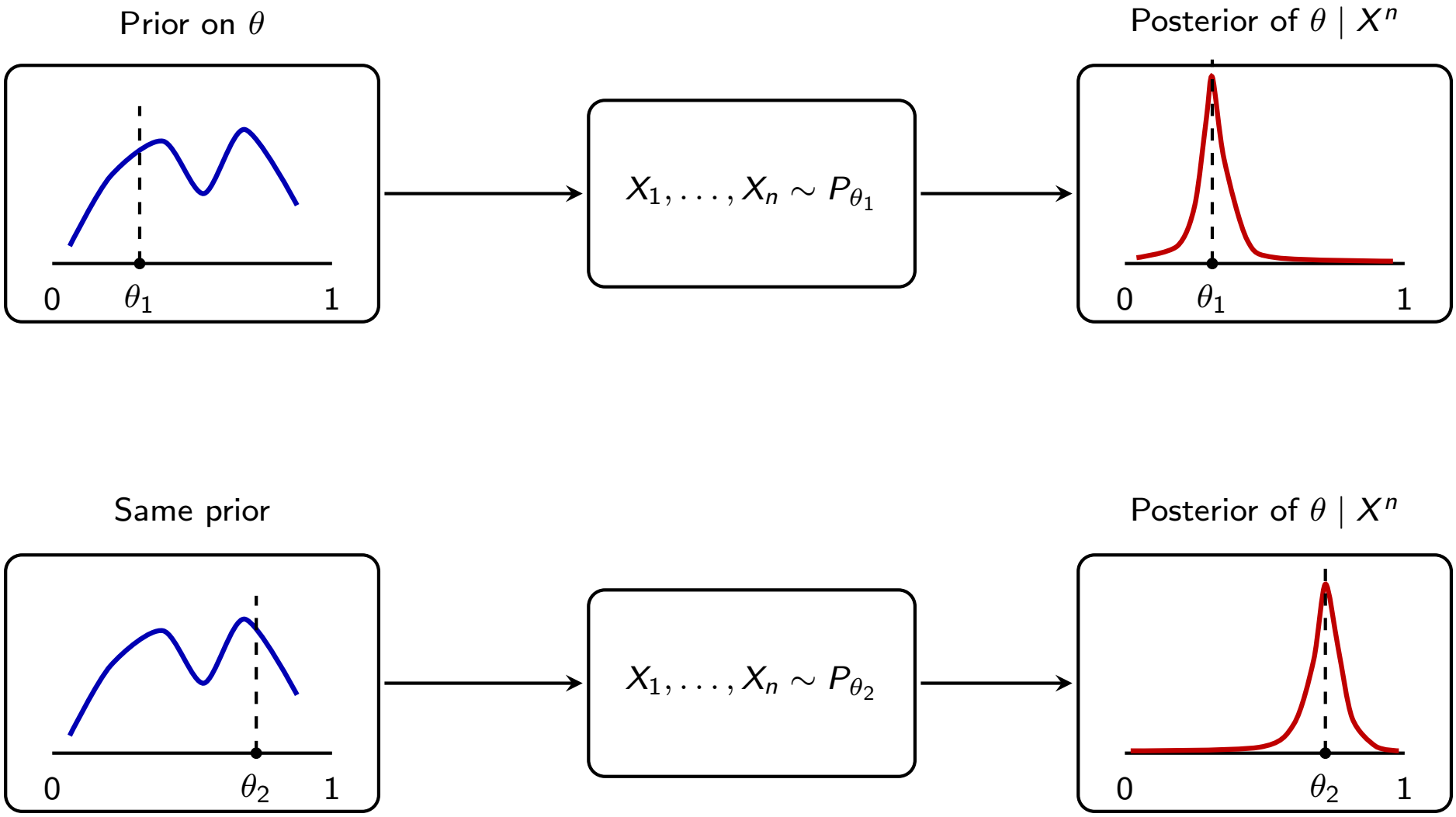
Posterior contraction



Posterior contraction



Posterior contraction



Posterior contraction: for different true parameters $\theta^\star$, the posterior $\Pi(\theta \mid X^n)$ concentrates around $\theta^\star$

Posterior contraction for pretrained estimator

The i-th coordinate of the pretrained Bayes estimator is:

$$\mathbb{E}_\pi[\theta_i \mid X^n] = \mathbb{E}_{G\sim\pi(G|X^n)}[\text{Bayes estimator under prior } G].$$

Posterior contraction for pretrained estimator

The i-th coordinate of the pretrained Bayes estimator is:

$$\mathbb{E}_\pi[\theta_i \mid X^n] = \mathbb{E}_{G\sim\pi(G|X^n)}[\text{Bayes estimator under prior } G].$$

The pretrained estimator is performing Bayesian inference:

- $\rightarrow$ step 1: update the posterior $\pi(G \mid X^n)$ based on the PoP π
- $\rightarrow$ step 2: sample a prior G from the posterior $\pi(G \mid X^n)$
- $\rightarrow$ step 3: apply the Bayes estimator under the sampled G

Posterior contraction for pretrained estimator

The i-th coordinate of the pretrained Bayes estimator is:

$$\mathbb{E}_\pi[\theta_i \mid X^n] = \mathbb{E}_{G\sim\pi(G|X^n)}[\text{Bayes estimator under prior } G].$$

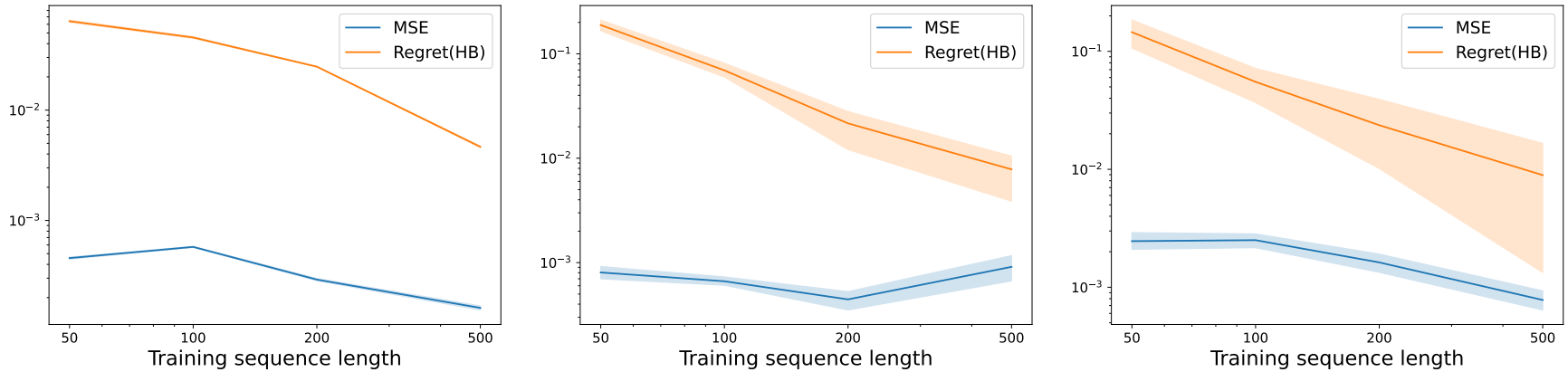
The pretrained estimator is performing Bayesian inference:

- $\rightarrow$ step 1: update the posterior $\pi(G \mid X^n)$ based on the PoP π
- $\rightarrow$ step 2: sample a prior G from the posterior $\pi(G \mid X^n)$
- $\rightarrow$ step 3: apply the Bayes estimator under the sampled G

Posterior contraction:

- $\rightarrow$ for X^n drawn from <u>any</u> test prior $G^\star$, the posterior $\pi(G \mid X^n)$ concentrates around $G^\star$
- $\rightarrow$ therefore, the pretrained estimator is essentially using the Bayes estimator under $G^\star$

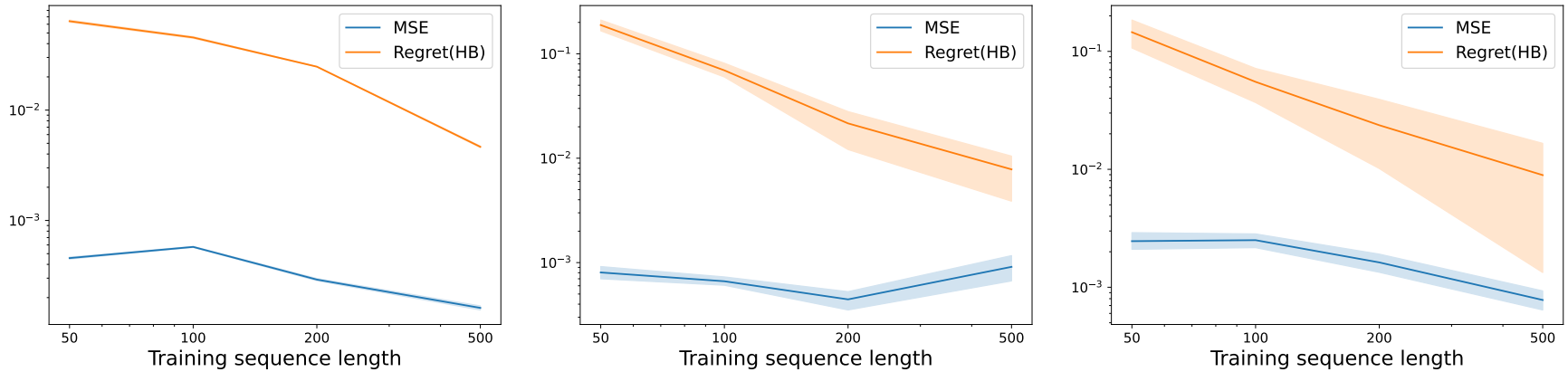
Numerical evidence for pretrained transformer



(a) $m = 2$. (b) $m = 5$. (c) $m = 10$

Figure: π_m: a class of PoPs where the exact Bayes estimator can be computed.
Orange: regret of exact Bayes estimator under π_m.
Blue: difference between the exact Bayes estimator and the pretrained transformer output.

Numerical evidence for pretrained transformer



(a) $m = 2$. (b) $m = 5$. (c) $m = 10$

Figure: π_m: a class of PoPs where the exact Bayes estimator can be computed.
Orange: regret of exact Bayes estimator under π_m.
Blue: difference between the exact Bayes estimator and the pretrained transformer output.

Numerical experiments suggest that

$$\|\text{Exact Bayes} - \text{Transformer Output}\| \approx 0 \quad (\ll \text{Regret(Exact Bayes)})$$

Length Generalization

Length generalization

Length generalization: transformers trained on sequence length n generalize to longer lengths $n_{\text{test}} > n$

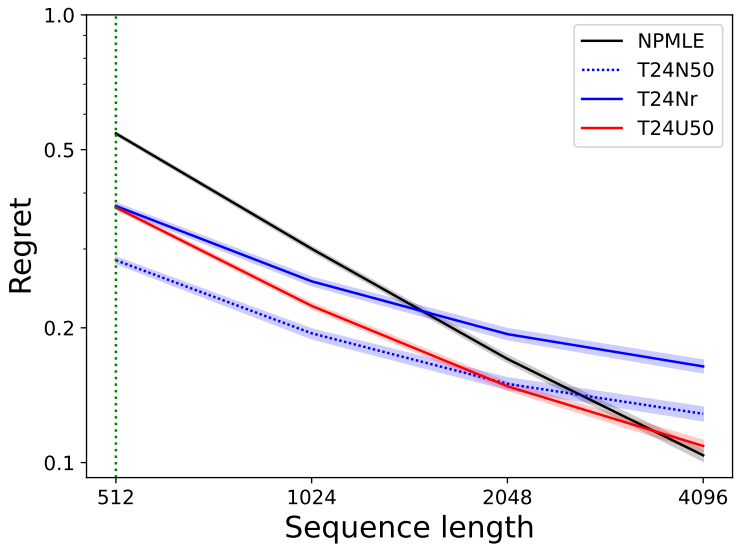


Figure: Regrets of NPMLE and three transformers with $n = 512$ and various n_{test}.

Length generalization

Length generalization: transformers trained on sequence length n generalize to longer lengths $n_{\text{test}} > n$

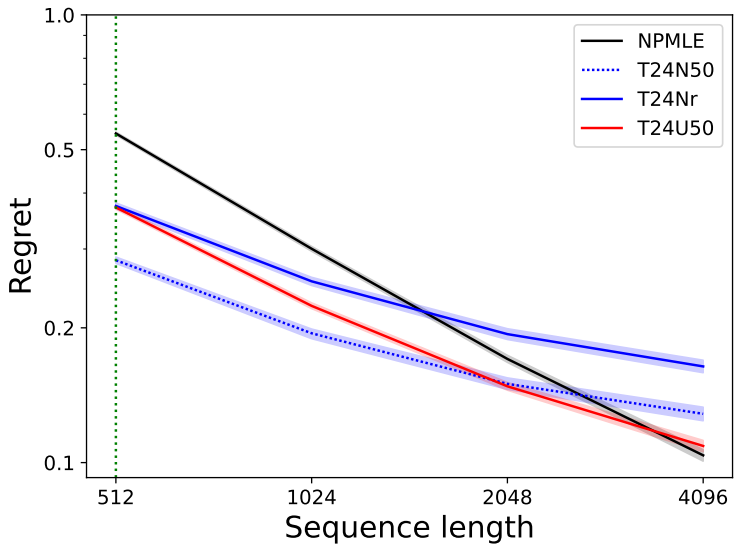


Figure: Regrets of NPMLE and three transformers with $n = 512$ and various n_{test}.

$\rightarrow$ as n_{test} increases, regrets of all pretrained transformers continue to decrease

$\rightarrow$ this improvement may saturate for large n_{test}

A model of length generalization

Need to extend the definition of $\widehat{\theta} : \mathbb{N}^n \to \mathbb{R}^n$ to $\mathbb{N}^{\geq n} \to \mathbb{R}^{\geq n}$

A model of length generalization

Need to extend the definition of $\widehat{\theta} : \mathbb{N}^n \to \mathbb{R}^n$ to $\mathbb{N}^{\geq n} \to \mathbb{R}^{\geq n}$

Length generalization model (Furuya-de Hoop-Peyré '25)

Given input X^n, the i-th output of a transformer is

$$Y_i = f\left(X_i, \frac{1}{n}\sum_{j=1}^{n} \delta_{X_j}\right),$$

where f is independent of i or n.

A model of length generalization

Need to extend the definition of $\widehat{\theta} : \mathbb{N}^n \to \mathbb{R}^n$ to $\mathbb{N}^{\geq n} \to \mathbb{R}^{\geq n}$

Length generalization model (Furuya-de Hoop-Peyré '25)

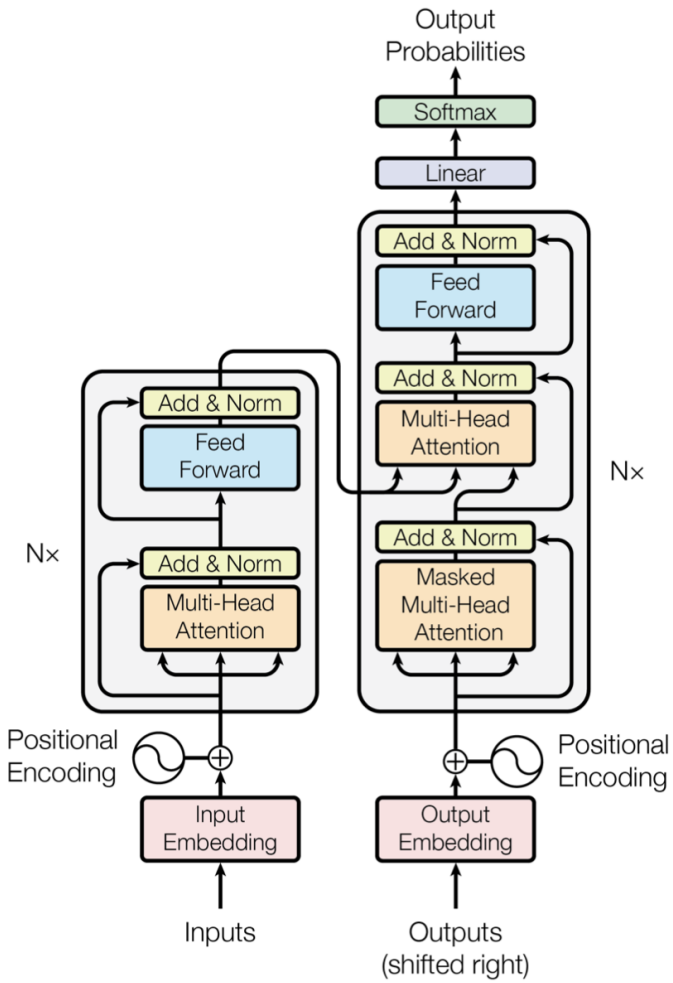
Given input X^n, the i-th output of a transformer is

$$Y_i = f\left(X_i, \frac{1}{n}\sum_{j=1}^{n} \delta_{X_j}\right),$$

where f is <u>independent</u> of i or n.

This holds for any learned transformer with:

- → no positional encoding;
- → no causal masking;
- → no final softmax layer;
- → only self-attention and token-wise operations.



Transformer architecture from [Vaswani et al. '17]

Length generalization: Bayesian inference

Lemma (Generalized Posterior)

Suppose the length generalization model holds for the pretrained Bayes estimator with training length n. Then for test sequence $X^{n_{\text{test}}}$ with a different length, the pretrained estimator is performing Bayesian inference with a generalized posterior:

$$\pi^{\alpha}(\mathrm{d}G \mid X^{n_{\text{test}}}) \propto \pi(\mathrm{d}G)\Big(\prod_{i=1}^{n_{\text{test}}} f_G(X_i)\Big)^{\alpha},$$

with $\alpha = \frac{n}{n_{\text{test}}}$.

Pretrained estimator is still performing Bayesian inference, but now with a fractional posterior update.

Length generalization: regret

Using posterior contraction for the α-posterior, we obtain:

Theorem (Regret bound)

Under the length generalization model, the pretrained Bayes estimator on length n achieves a worst-case regret on test length $n_{\text{test}} \geq n$ upper bounded by

$$\frac{1}{n_{\text{test}}} \times \textit{metric entropy of model} + \frac{1}{n} \times \textit{prior coverage of } \pi$$

Length generalization: regret

Using posterior contraction for the α-posterior, we obtain:

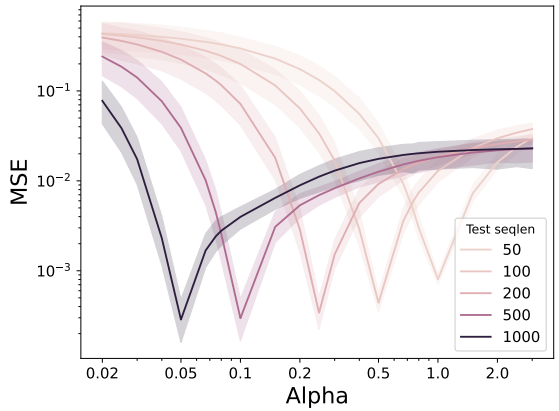
Theorem (Regret bound)

Under the length generalization model, the pretrained Bayes estimator on length n achieves a worst-case regret on test length $n_{\text{test}} \geq n$ upper bounded by

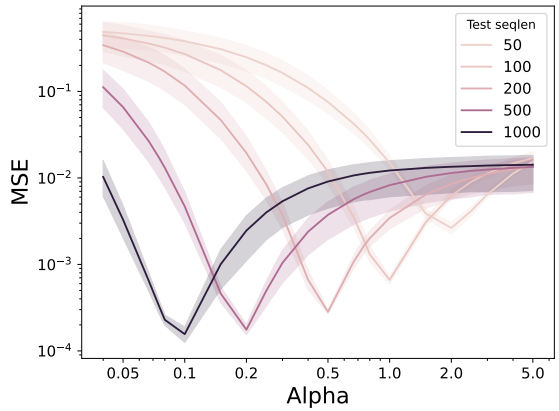
$$\frac{1}{n_{\text{test}}} \times \textit{metric entropy of model} + \frac{1}{n} \times \textit{prior coverage of } \pi$$

Explains both the improvement and saturation when test length increases!

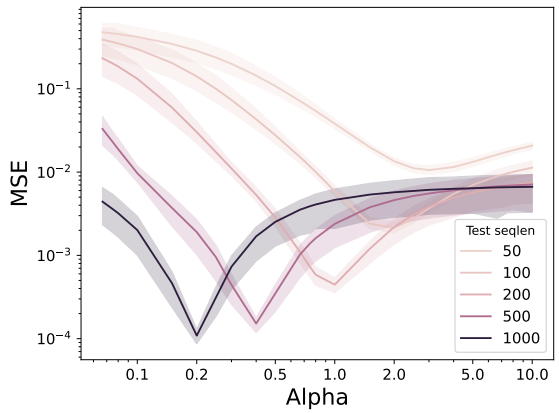
Numerical evidence for $\alpha \simeq \frac{n}{n_{\text{test}}}$



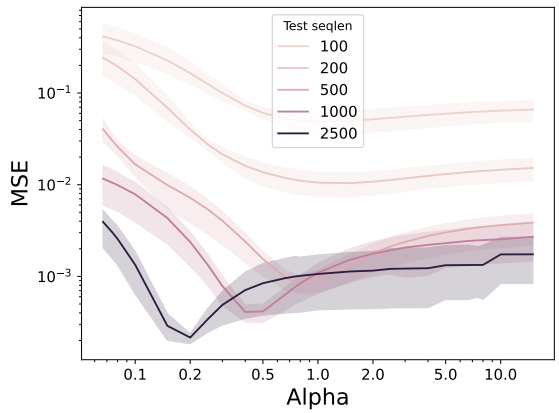
(a) $n = 50$.



(b) $n = 100$.



(c) $n = 200$.



(d) $n = 500$.

Figure: Mean squared differences between transformer output and Bayes estimator with α-posteriors, with various $(n, n_{\text{test}}, \alpha)$. The closest fit is consistently $\alpha \simeq \frac{n}{n_{\text{test}}}$.

Conclusion

Main message

Everyone does Bayesian:

- $\rightarrow$ Frequentist: (approximate) least favorable prior
- $\rightarrow$ Empirical Bayes: data-driven prior
- $\rightarrow$ Pretrained transformer: (generalized) Bayesian inference

Conclusion

Main message

Everyone does Bayesian:

- $\rightarrow$ Frequentist: (approximate) least favorable prior
- $\rightarrow$ Empirical Bayes: data-driven prior
- $\rightarrow$ Pretrained transformer: (generalized) Bayesian inference

Thank You!