Investigating GPU-Accelerated Kernel Density Estimators for Join Selectivity Estimation

Martin Kiefer

Master Thesis

ID: 358316

Advisor: Max Heimel

Fachgebiet Datenbanksysteme und Informationsmanagement
Technische Universität Berlin

http://www.dima.tu-berlin.de/

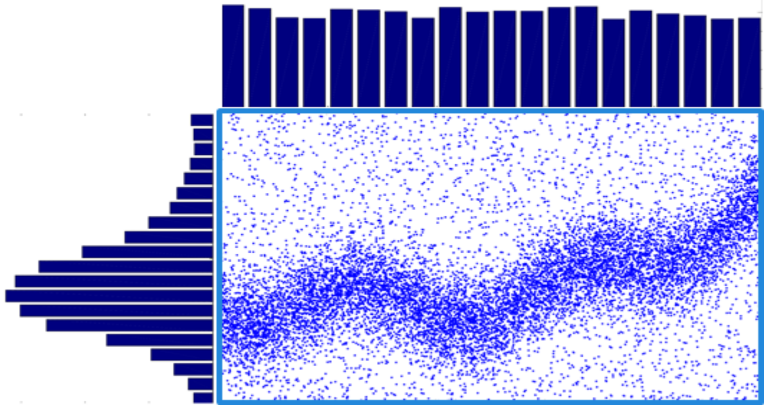
Query Optimization and Selectivity Estimation

PROBLEM

Problem

- Query optimizer requires output cardinalities
- Selectivity: Output cardinality, normalized to $[0,1]$

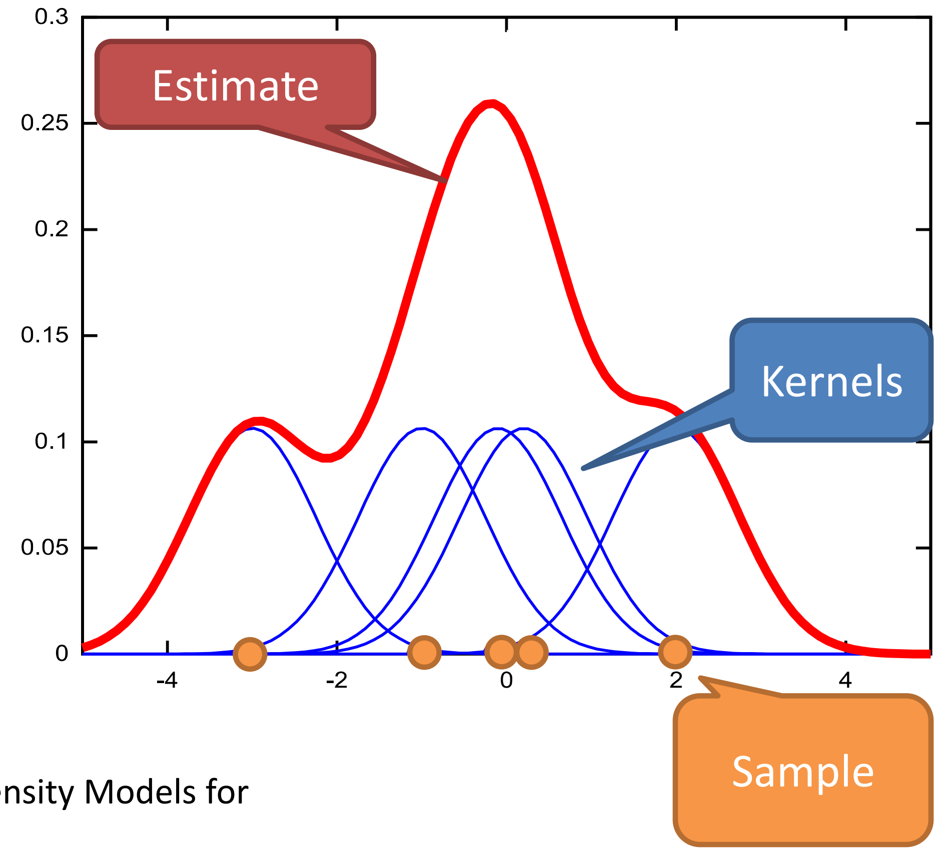
- In practice, *Attribute Independence* is assumed
  - $P(A_1 = v_1 \wedge A_2 = v_2) = P(A_1 = v_1) \cdot P(A_2 = v_2)$
  - Per-column statistics (1D Histograms, Most Common Values, Distinct Values, …)



- This assumption is commonly violated in real-world data

- Model probability density functions
- Based on a random sample

- Good estimation quality for range selections on continuous attributes
- Error-driven hyper-parameter optimization
- GPU-Acceleration
- Sample maintenance



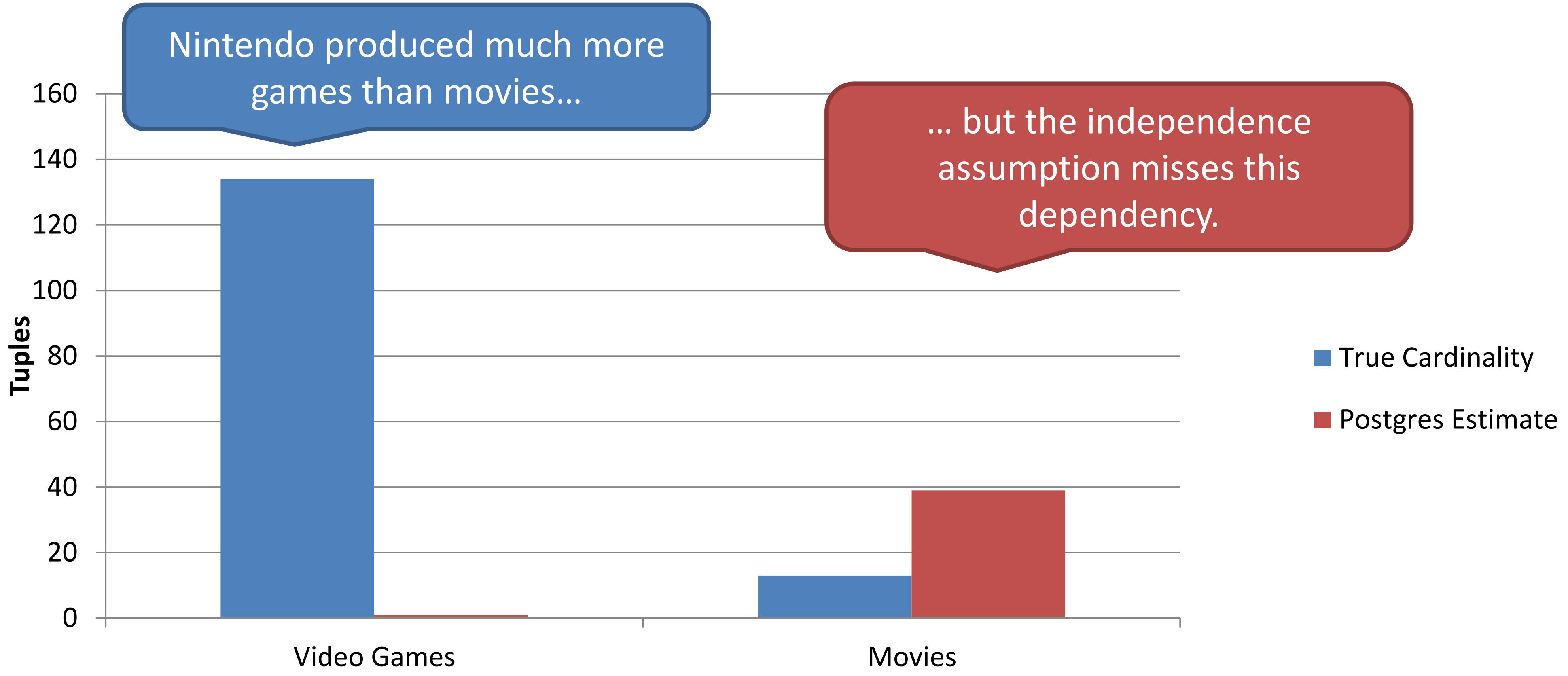
Max Heimel, ***Martin Kiefer***, and Volker Markl. Self-Tuning, GPU-Accelerated Kernel Density Models for Multidimensional Selectivity Estimation (SIGMOD 2015)

Martin Kiefer, Max Heimel, and Volker Markl. Demonstrating Transfer-Efficient Sample Maintenance on Graphics Cards (EDBT 2015 Demo)

… and now Equi-Joins!

- Equi-joins change the game
  - They involve two tables
  - Join attributes are discrete
- The independence assumption misses dependencies in the join result
  - Every tuple from one table is equally likely to join with a tuple from another table
- Further assumptions
  - Containment of Value Sets (PK-FK Joins)
  - Preservation of Value Sets (Non-join values are not affected)
- Joins are particularly error-prone
  - Errors propagate exponentially in the number of joins

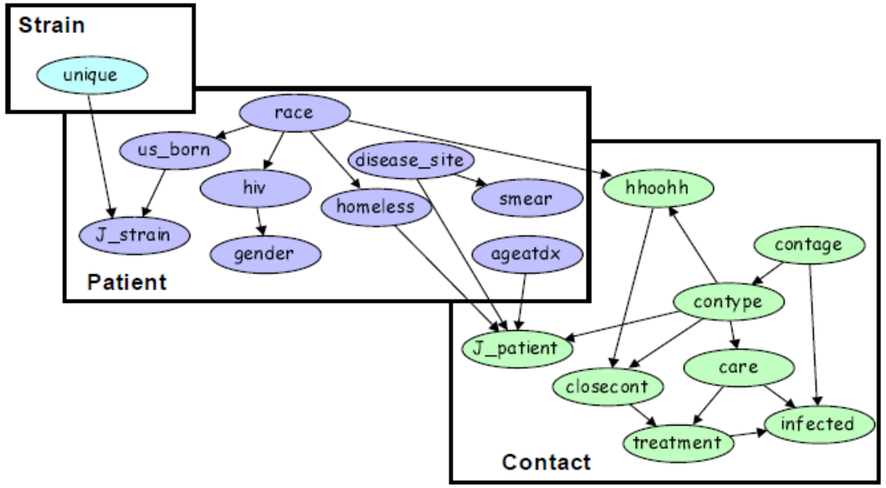
- IMDb Dataset
  - Select all **games/movies** that were produced by **Nintendo**



Probablistic Graphical Models, Wavelets, Sampling, Sketches

RELATED WORK

- Probabilistic Graphical Models
  - Model attributes as random variables
  - Expensive model construction



- Wavelets
  - Apply lossy synchronization to tables
  - Support joins, selections and aggregates
  - Expensive construction

Tzoumas, Kostas, et al. "Lightweight graphical models for selectivity estimation without independence assumptions.“ (VLDB’14)
Chakrabarti, Kaushik, et al. "Approximate query processing using wavelets.“ (VLDB Journal 01)

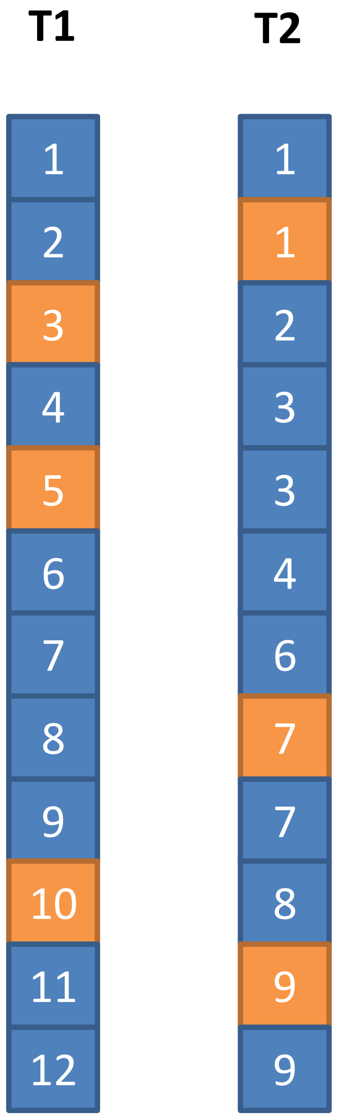
Related Work: Sampling

- Calculate selectivity of queries on random portions of the tables

- Most intuitive way: Independent uniform samples
- Construction and maintenance is easy

- Have various applications
  - Selectivity estimation for several operators
  - Approximate Aggregates
  - Visualization

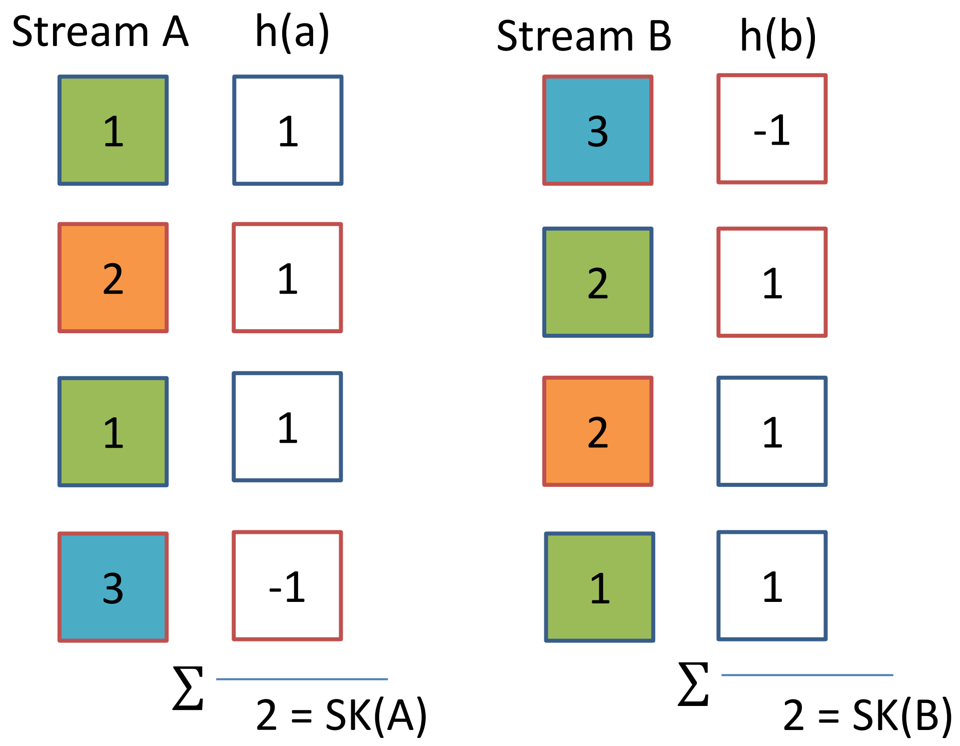
- Independent samples may not provide information on the join
  - Join keys in samples are unlikely to match for small samples



Cormode, Graham, et al. "Synopses for massive data: Samples, histograms, wavelets, sketches." *Foundations and Trends in Databases* 4.1–3 (2012): 1-294.

Related Work: Sketches

- AGMS-Sketch
  - Use a 2-wise independent hash function $h : D \rightarrow \{1, -1\}$
  - Create sketch by summing over hash values
  - Estimate $\hat{J}$ is computed by multiplying two sketches
- Key Idea
  - $\mathbb{E}[h(v_1) \cdot h(v_2)] = \begin{cases} 0 & v_1 \neq v_2 \\ 1 & v_1 = v_1 \end{cases}$
- Average over n sketches for good results
  - Computationally expensive for large n
- Deletions in the stream
  - Subtract hash value from sketch
- Computationally expensive



$$\hat{J} = SK(A) \cdot SK(B) = 4$$

Vengerov, David, et al. "Join size estimation subject to filter conditions."*Proceedings of the VLDB Endowment* 8.12 (2015): 1530-1541.

Provide an estimate for the joint frequency distribution of a table

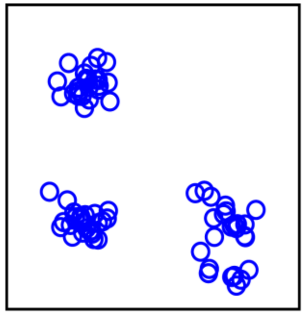
APPROACH:
DISCRETE KERNEL DENSITY ESTIMATORS

Background: Kernel Density Estimators

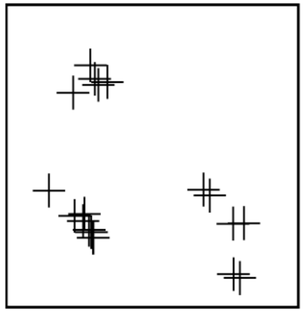
Average…

… over the sample point contributions

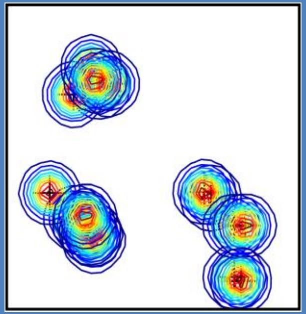
$$\hat{P}_H(\vec{x}) = \frac{1}{|S|}\sum_{i=1}^{|S|} K_H(s_i, x)$$



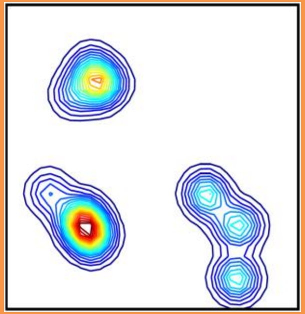
Dataset



Sample S



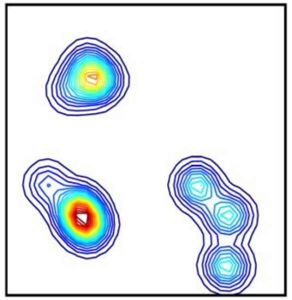
Kernels K_H



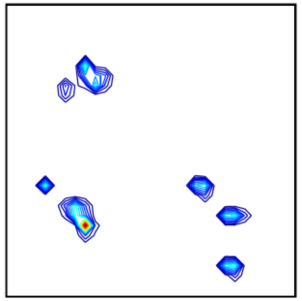
Estimate $\hat{P}_H$

Background: Bandwidth optimization

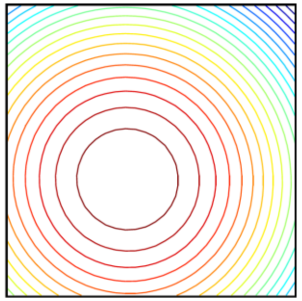
- Bandwidth optimization is crucial to the estimation quality



Good fit



Overfit

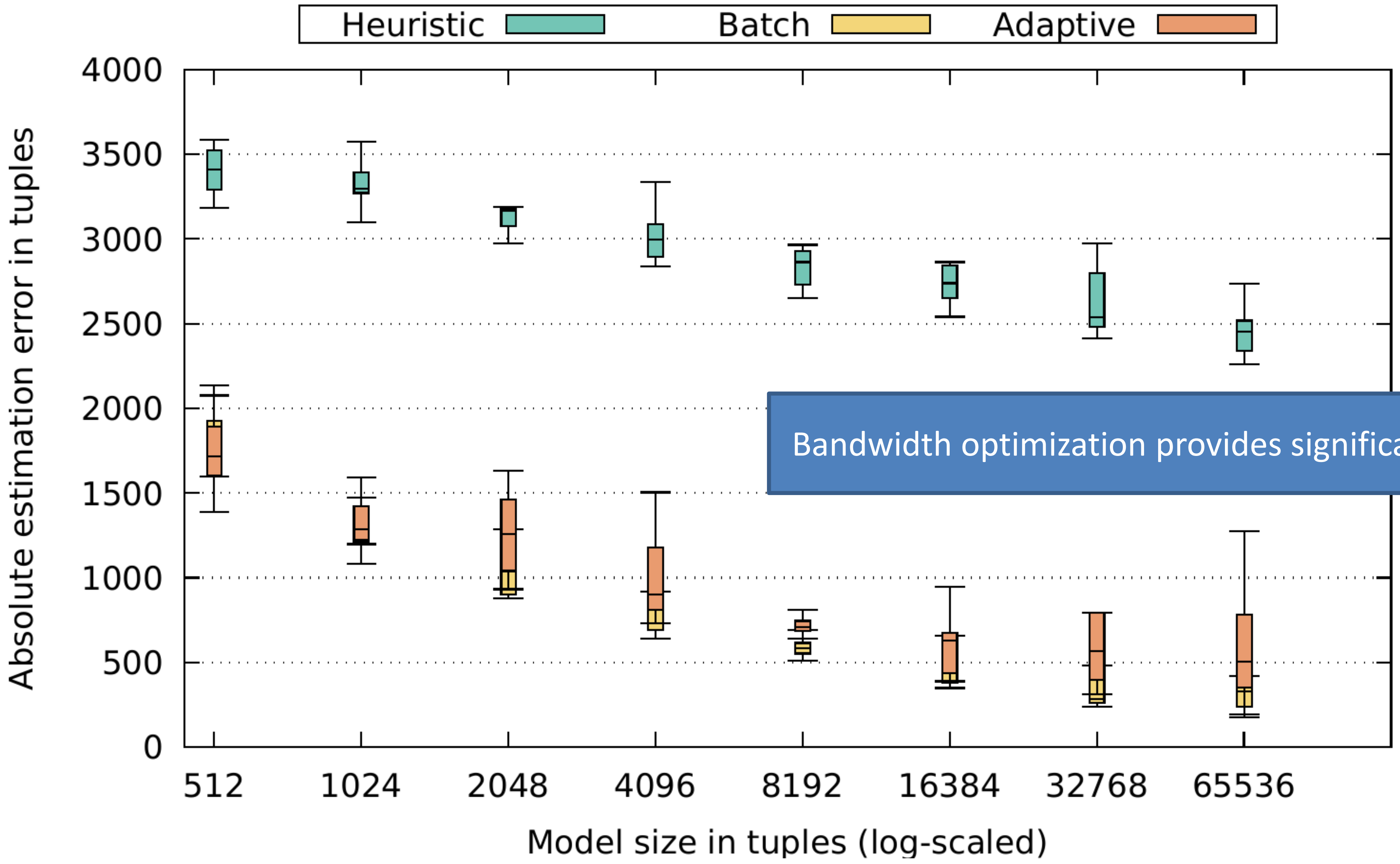


Underfit

- Error-driven bandwidth optimization using query feedback
  - Compute gradient of the estimation error with respect to the bandwidth
  - Use a gradient-based optimization algorithm

- Two flavors
  - Batch: Collect representative queries and optimize
  - Adaptive: Online learning on small mini-batches

Bandwidth optimization is crucial to the estimation quality

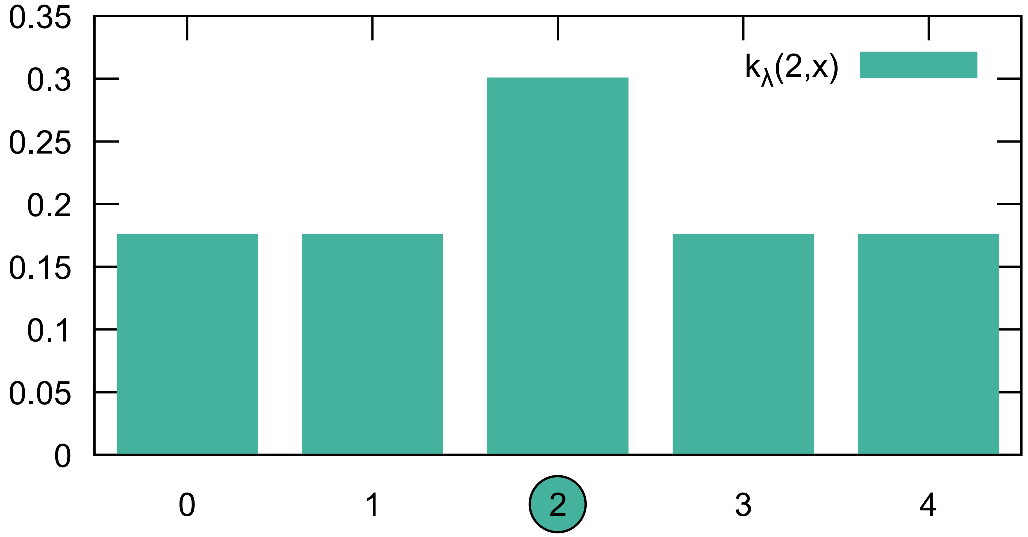


Bandwidth optimization provides significantly better results!

- Equi-Joins are inherently discrete
  - Provide an estimate for the joint frequency distribution
  - We need a probability mass function!

- Categorical kernel over a domain of values D
  - Smoothing by difference - not distance

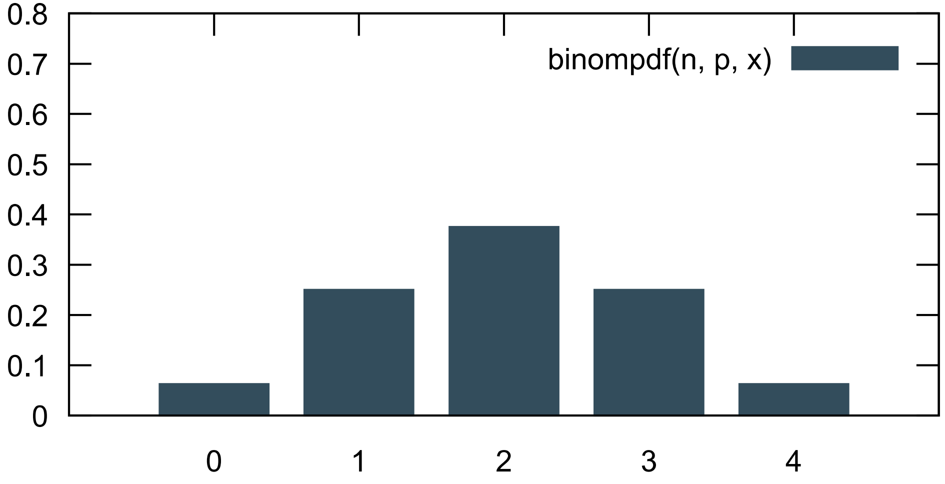
$$k_\lambda(x, y) = \begin{cases} (1-\lambda) & if\ x = y \\ \dfrac{\lambda}{|D| - 1} & if\ x \neq y \end{cases}$$



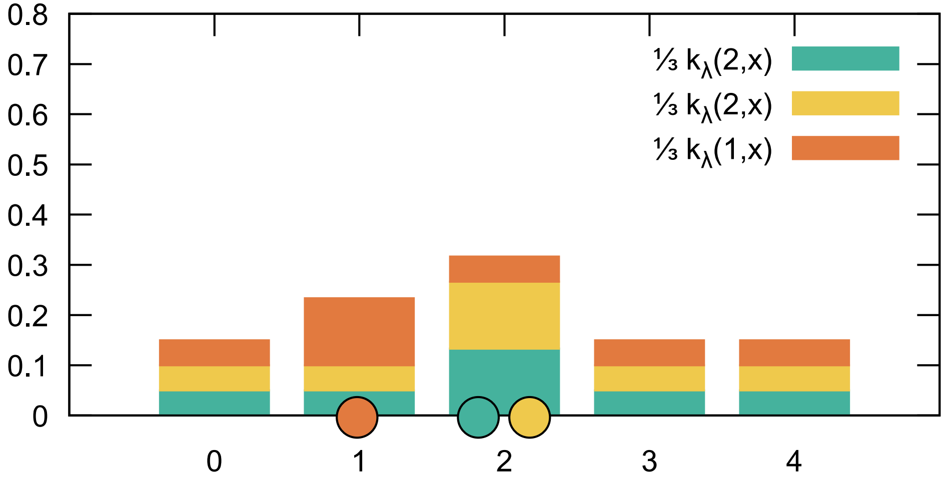
Qi Li, Jeff Racine, Nonparametric estimation of distributions with categorical and continuous data, Journal of Multivariate Analysis, Volume 86, Issue 2, August 2003, Pages 266-29

Average… … over categorical kernels!

$$\hat{P}_\lambda(x) = \frac{1}{|S|}\sum_{i=1}^{|S|} k_\lambda(s_i, x)$$

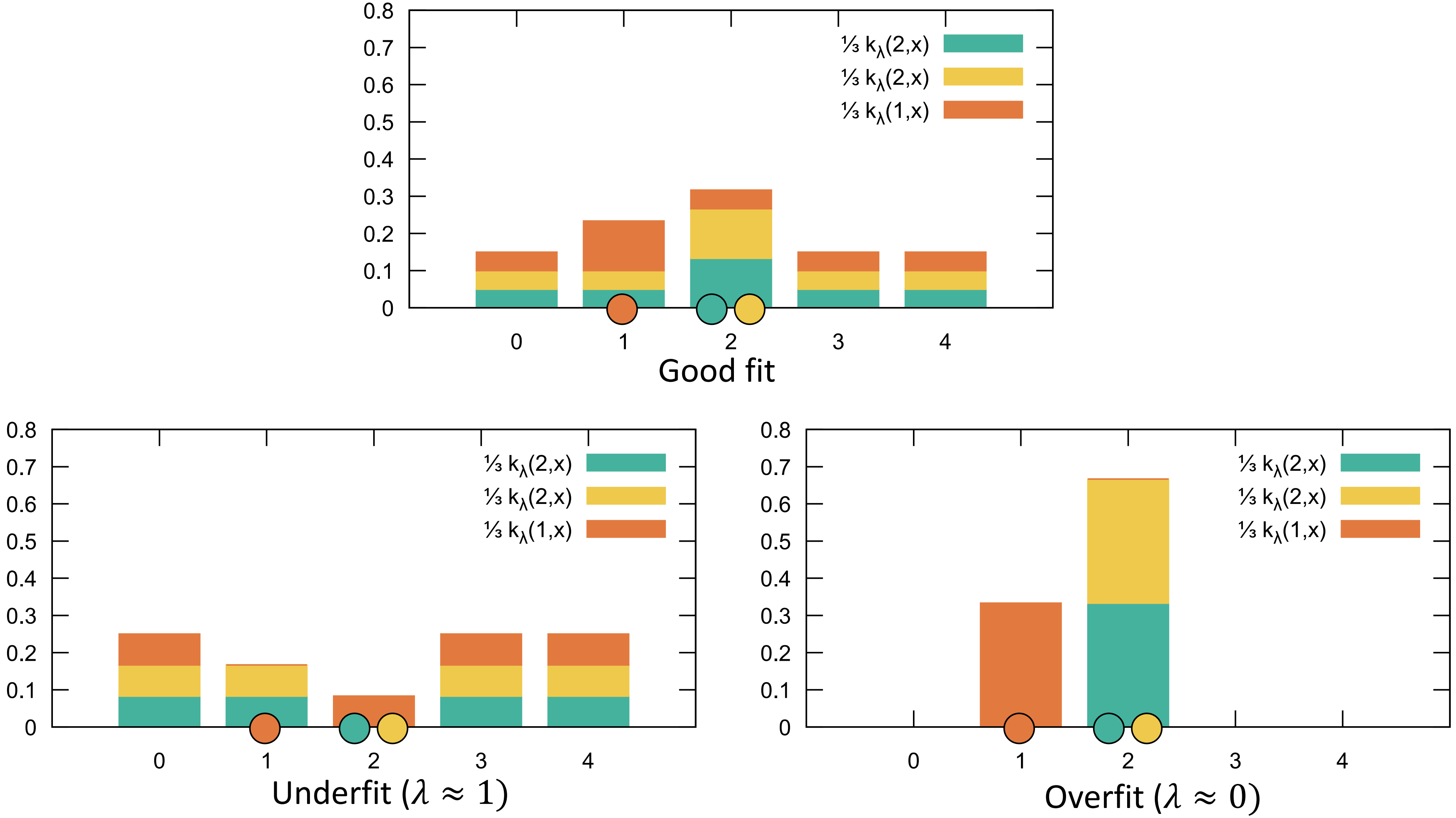


Binomial Distribution

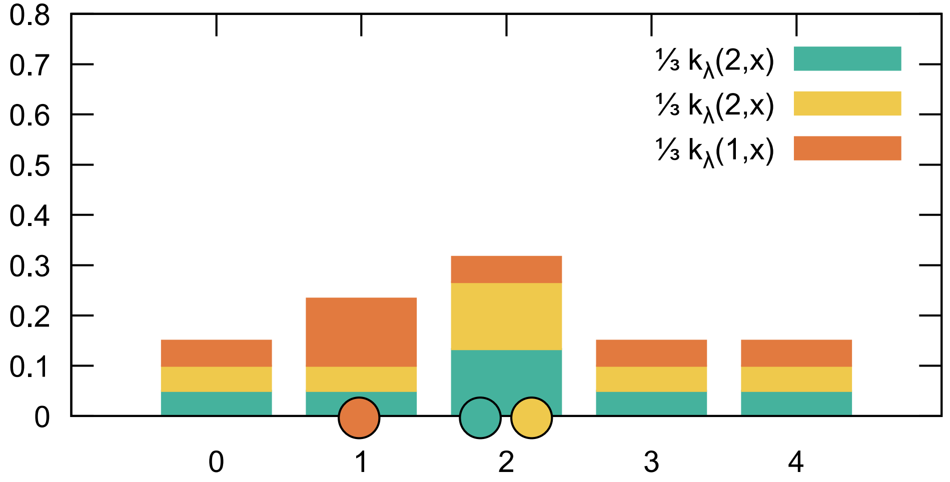


Discrete Kernel Density Estimator

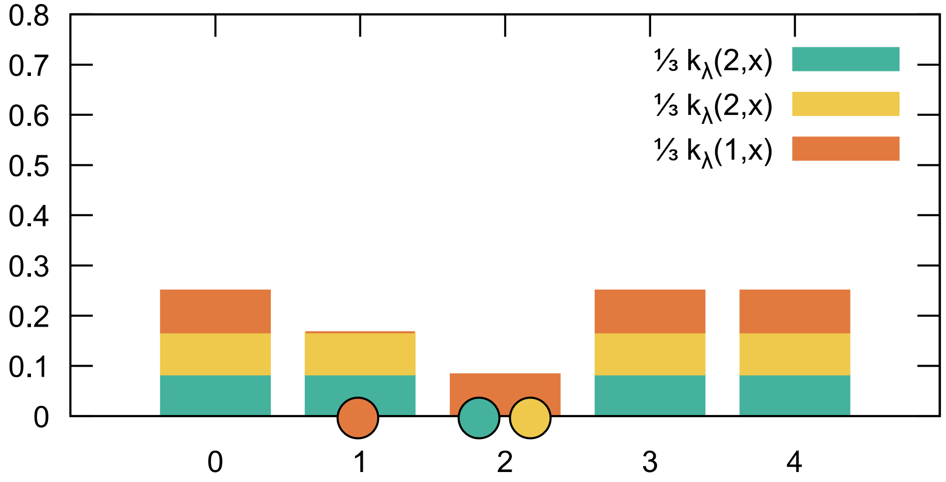
Bandwidth Impact on the Categorical Kernel



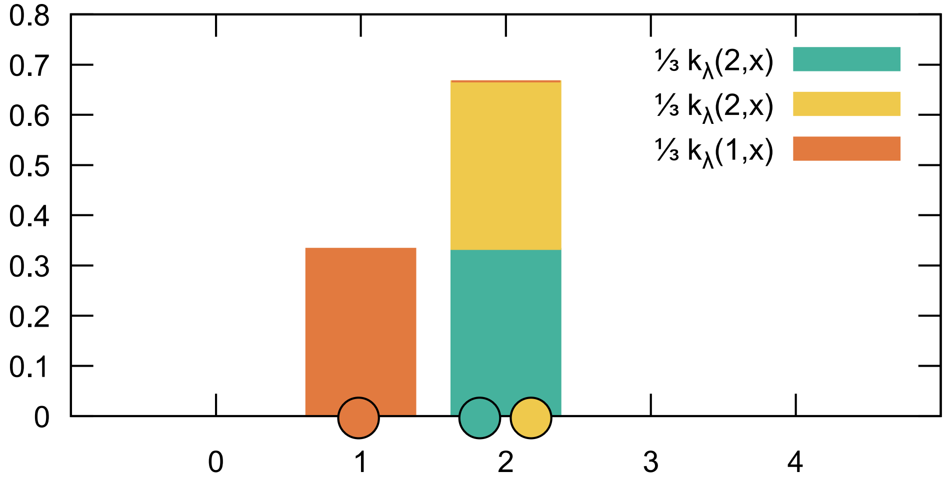
Sample evaluation is part of the bandwidth space!



Good fit



Underfit ($\lambda \approx 1$)



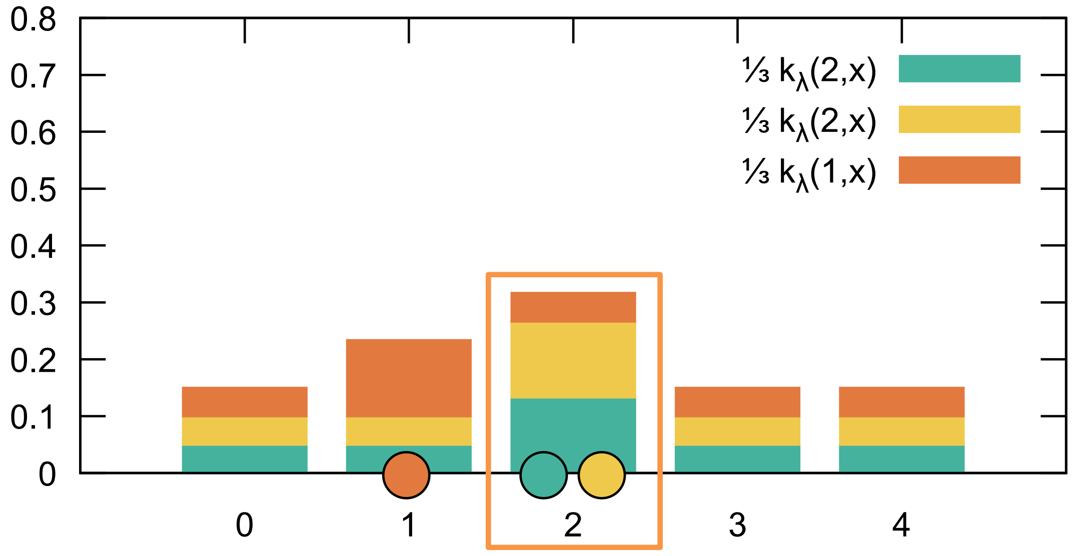
Overfit ($\lambda \approx 0$)

- Given:
  - Random sample S of a relation R with attribute A
  - Number of distinct values per column
  - Bandwidth λ
- Equality selections are trivial
  - One pass to compute the estimate for a value v
  - Values are assumed to be in the domain
- But what about joins?

Evaluate estimator in one pass over the sample

$$\hat{P}_\lambda(A = v) = \frac{1}{|S|} \boxed{\sum_{i=1}^{|S|} k_\lambda(s_i, v)}$$

$$\frac{\left|\sigma_{A_1=2}(R)\right|}{|R_1|} \approx \hat{P}(A = 2)$$

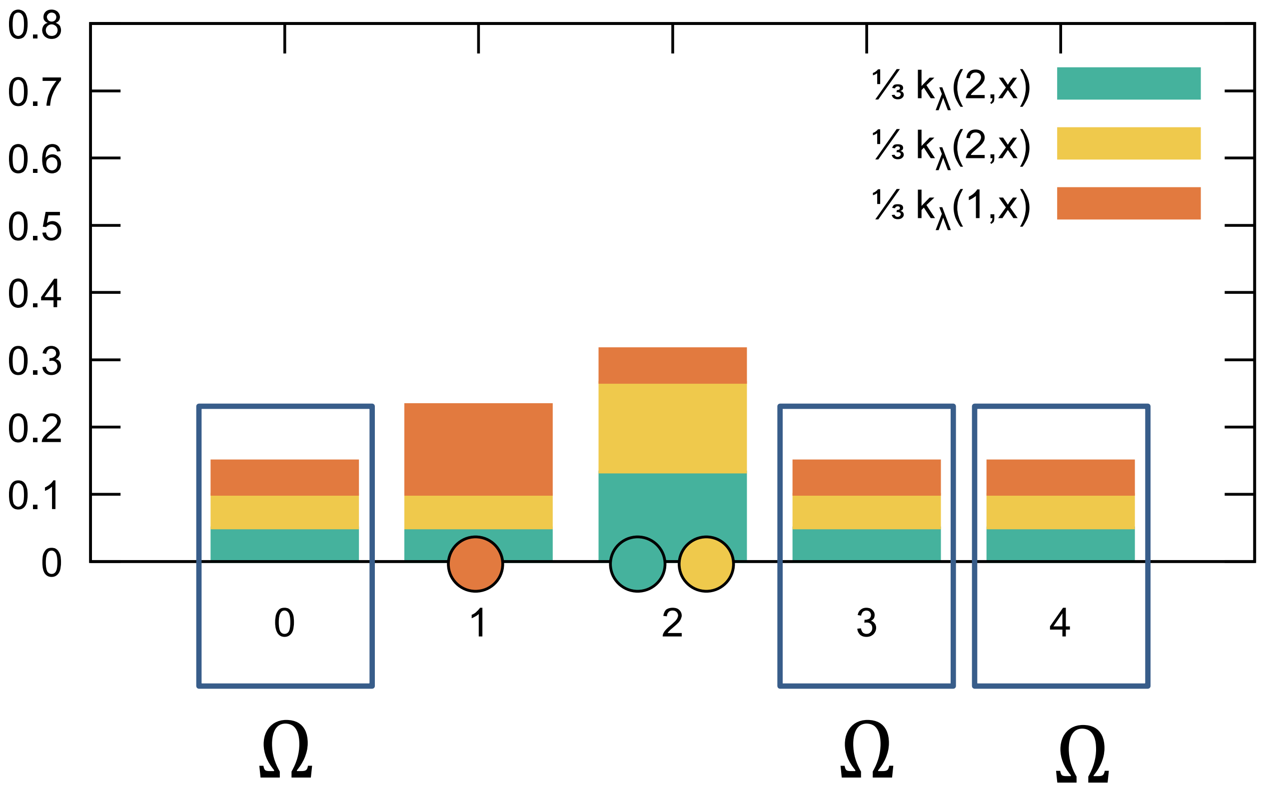


- Selectivity for equi-joins
  - $\frac{|R_1 \bowtie R_2|}{|R_1| \cdot |R_2|} = \sum_{v \in R_1.A_1 \cap R_2.A_1} P_1(A = v) \cdot P_2(A = v)$
- So, we just plug in the estimator?
  - $\hat{J} = \sum_{v \in R_1.A_1 \cap R_2.A_1} \hat{P}_1(A = v) \cdot \hat{P}_2\,(A = v)$

Every value in the join domain!

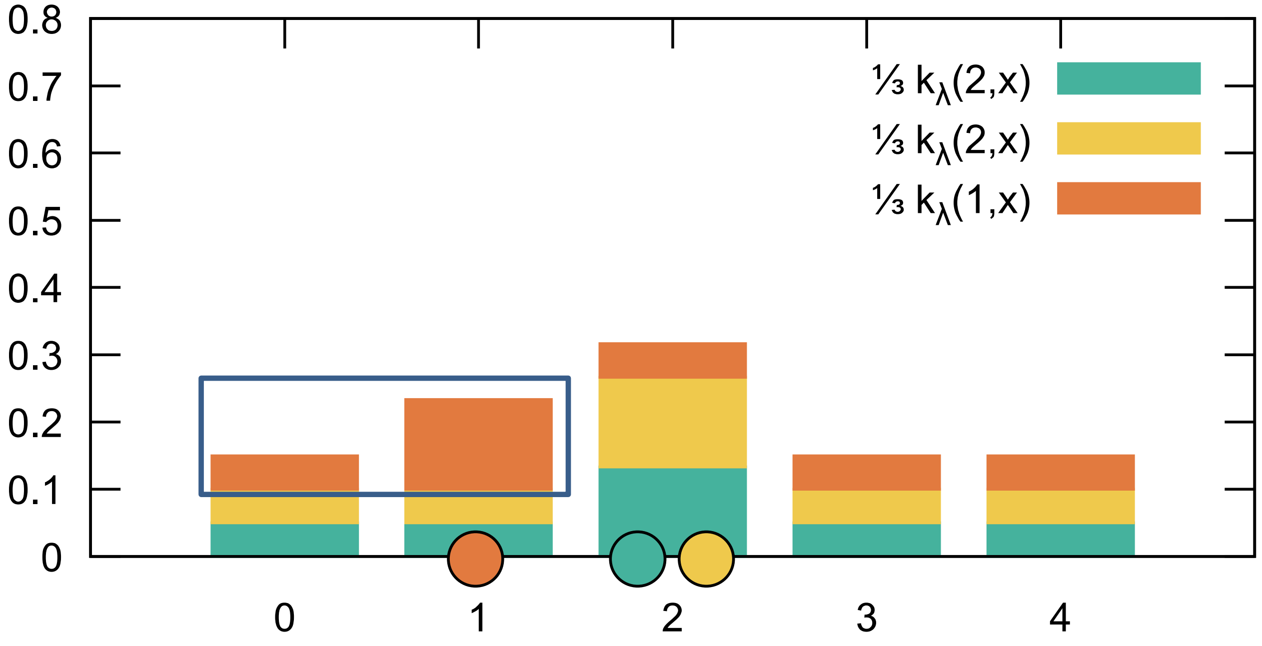
- Naïve evaluation is too expensive
- We need to exploit properties of the kernel!

- Values outside the sample have the same probability Ω



Observation 2

- $\mathrm{P}(\mathrm{A} = \mathrm{v})$ differs from Ω in the contribution of sample points with value v



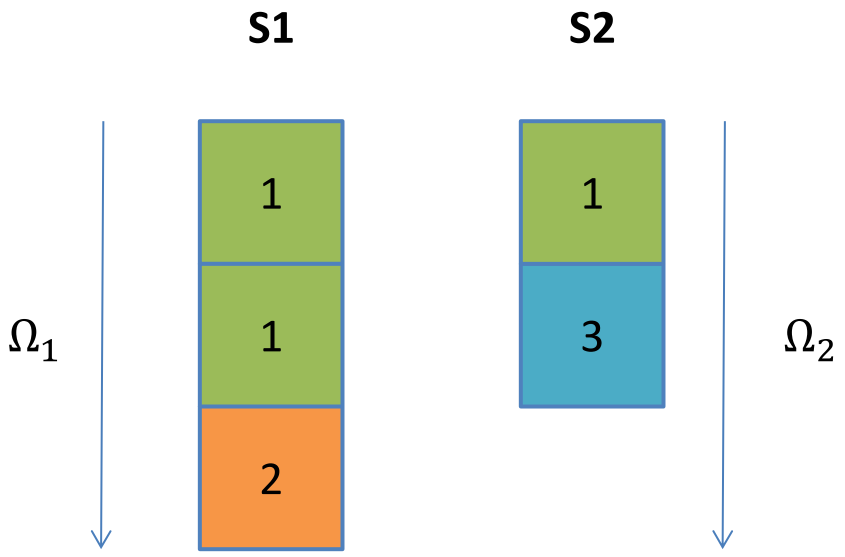
Algorithm for Two-Way Equi-Joins

- Given:
  - Samples S1 and S2
  - Distinct values per attribute
  - Bandwidth λ

- Assume both join attributes have the same domain
  - Say, A = {1,2,3,4,5}

- **Step 1**: Sort the samples on join attribute

- **Step 2**: Compute Ω_1 and Ω_2 for both samples



Step 3 & 4: Merge and Estimate

Step 3: Account for values in the samples by merging

	S1		S2
Case 1:	1, 1	$\hat{P}_1(A=1)\cdot\hat{P}_2(A=1)$	1
		$+$	
Case 2:	2	$\hat{P}_1(A=2)\cdot\underbrace{\hat{P}_2(A=2)}_{=\Omega_2}$	2
		$+$	
	3	$\underbrace{\hat{P}_1(A=3)}_{=\Omega_1}\cdot\hat{P}_2(A=3)$	3

$$\hat{J} :=$$

Furthermore, count the number of values handled in this step (3)

Step 4: Account for values not in the samples

S1	S2
4	4
5	5

$$\hat{J} \mathrel{+}= 2\cdot\Omega_1\cdot\Omega_2$$

Algorithm Properties

- Easily extended to include selections with conjunctive equality predicates on other attributes

- Evaluation cost of general algorithm:

Algorithm Step	Cost
Sorting	$O(\lvert S\rvert \log \lvert S\rvert)$ for each unsorted sample
Compute Ω_1, Ω_2	One pass over each sample
Merge and Estimate	One pass over each sample

Algorithm Variants

- *GPU adaption*
  - Based on a binary search join

- *N-way equi-joins*
  - One join attribute per table
  - More than one join attribute per table
    - Requires independence assumptions

Selections, Two-Way Joins, Three-Way Joins

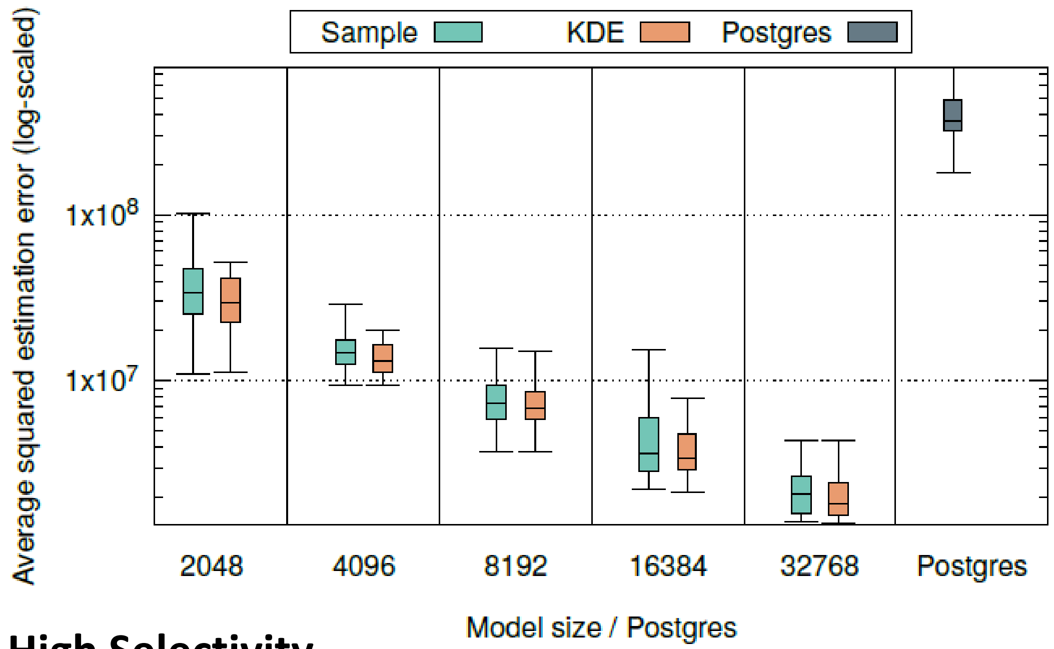
EVALUATION

Setup

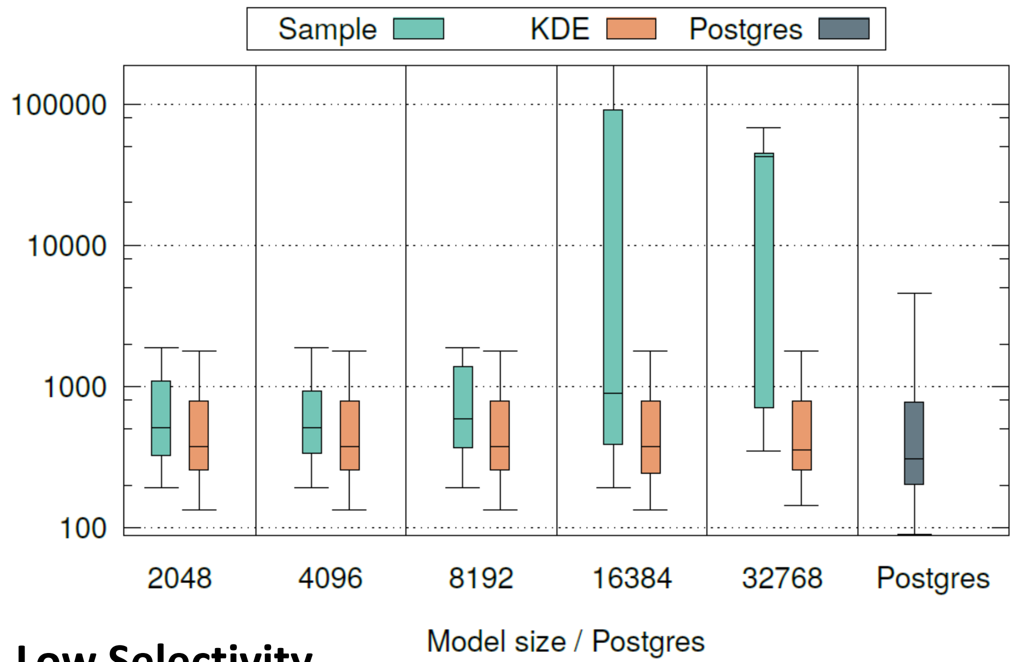
- Datasets
  - **IMDb**: Real-world dataset based on the Internet Movie Database
  - **Zipf(x)**: Artificial dataset based on the Zipf distribution with parameter x
- Selection Workload
  - Selections with equality predicate on subset of attributes from a table
  - 20 iterations
  - 100 test queries
  - 300 training queries
- Estimators
  - **Postgres**: Estimates used by Postgres (MCVs, Equi-Width Histograms, Distinct Values)
  - **Sample**: Independent uniform samples
  - **KDE**: Discrete Kernel Density Estimator Model

KDEs work well when samples do…

… but avoid their pitfalls!



High Selectivity



Low Selectivity

Selections: production_year, kind_id, series_nr

Selections: movie_id, role_id

Join Experiments

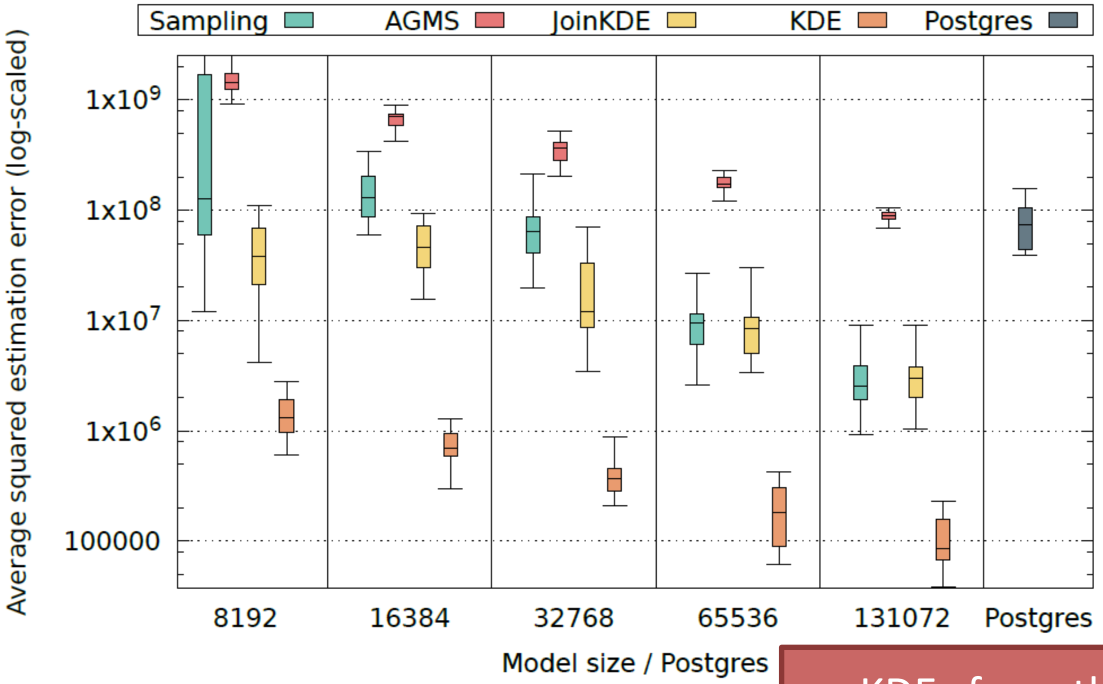
- Estimators
  - **Postgres**: Estimates used by Postgres (MCVs, Equi-Width Histograms, Distinct Values)
  - **Sample**: Independent uniform samples
  - **AGMS**: AGMS Sketch using polynomials over primes
  - **JoinKDE**: Join algorithm for independent Kernel Density Estimator models
  - **KDE**: Discrete Kernel Density Estimator Model constructed from join result

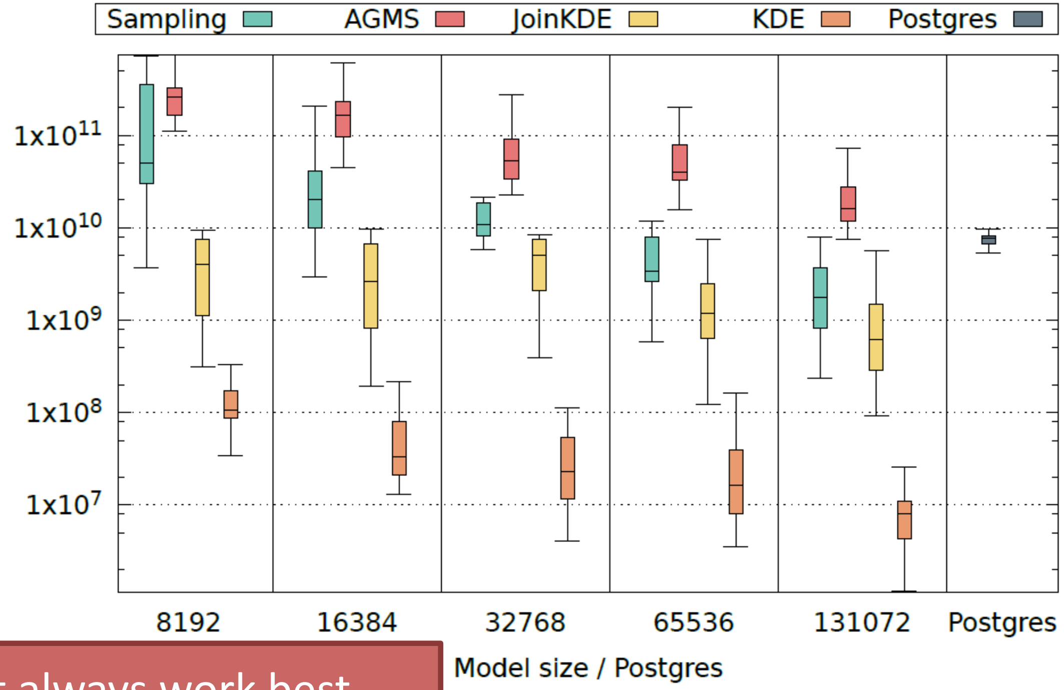
- Workload
  - Fixed join attributes, selections on non-join attributes
  - 20 iterations
  - 100 test queries
  - 300 training queries

Two-Way Joins on IMDb Dataset

Even if the assumption is violated...

... JoinKDE performed well!





KDEs from the join result always work best.

Join: movie_keyword.movie_id = title.id

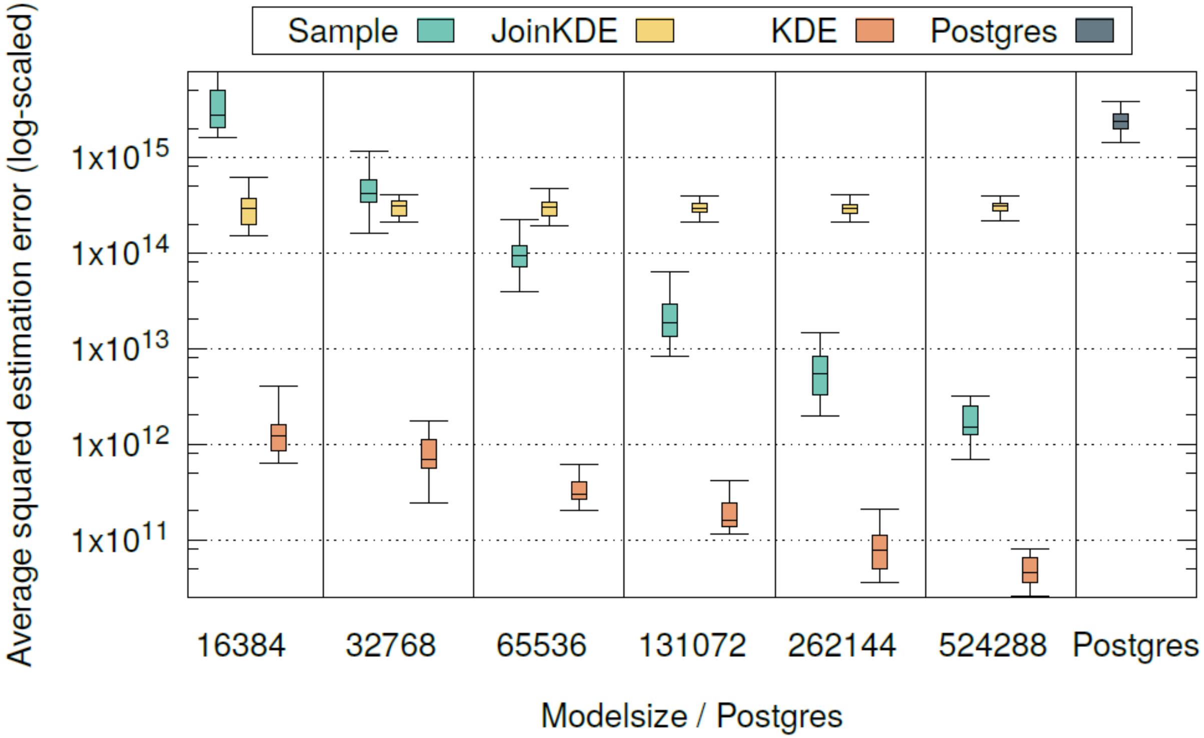
Selections: movie_keyword.keyword_id, title.kind_id

Join: movie_companies.company_id = company_name.id

Selections: movie_companies.company_type_id, company_name.country_code

Three-way joins on IMDb dataset

For three-way joins the independence assumptions hurt our estimates.



Join: movie_companies.movie_id = cast_info.movie_id, person_info.person_id = cast_info.person_id

Selections: movie_companies.company_type_id, person_info.info_type_id, cast_info.role_id

Summary

- Discrete KDEs model joint probability distribution
  - Selections
  - Joins

- Discrete KDEs work well for equality selections and two-way joins
  - Benefits of samples without the drawbacks
  - Worst-case performance matches Postgres
  - Much to win, not much to loose

- Future work: Improve performance for n-way joins

- Future work: The approach can be extended to mixed KDE models
  - Joins
  - Selections with equality predicates
  - Selections with range predicates