

Smooth Interpolating Histograms with Error Guarantees

Thomas Neumann ^{*} , **Sebastian Michel** [◇]

^{*} Max-Planck-Institut für Informatik

[◇] Ecole Polytechnique Fédérale de
Lausanne (EPFL)

July 9, 2008

Outline

- 1 Motivation
- 2 Data Representation
- 3 Construction
- 4 Algorithms
- 5 Evaluation

Motivation (Scenario)

Specific application: parameter tuning

```
(select id from list1  
  where score $\geq$ 0.5)
```

union all

```
(select id from list2  
  where score $\geq$ 0.4)
```

vs.

```
(select id from list1  
  where score $\geq$ 0.4)
```

union all

```
(select id from list2  
  where score $\geq$ 0.5)
```

- optimizer will try out many different values
- no artificial extrema

Goals

Histogram: a compressed representation of a value distribution

- often used for selectivity estimations
- how many tuples will satisfy a (range/point) predicate?
- we only consider one dimensional histograms here

Goals (2)

Two goals in particular are desirable:

- 1 smooth estimations
 - $\forall_{[a,b] \supseteq [a',b']} H([a, b]) \geq H([a', b'])$
- 2 interpolating estimations
 - estimation error should be bound over the whole domain

Existing approaches often violate these, in particular the second goal.

Our Approach

We want to approximate the value distribution with a linear spline (a polyline)

Benefits:

- compact representation
- easy to evaluate
- good for approximations in general

Problems:

- how to represent the original data
- how to decide if an approximation is good

Outline

- 1 Motivation
- 2 Data Representation
- 3 Construction
- 4 Algorithms
- 5 Evaluation

Raw Data

We consider numerical data and assume:

The raw data consists of a (multi-)set with n real values.

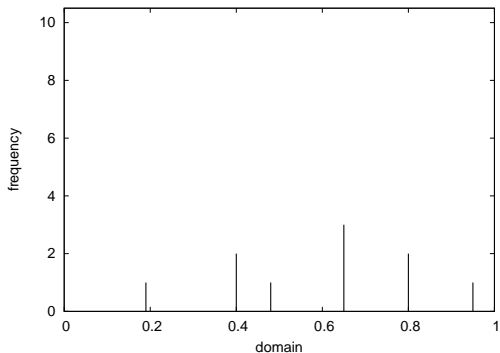
Task: compress the n values into k values such that the following questions can be answered:

- how many values $v = c$ exist (point queries)
- how many values $c_1 \leq v \leq c_2$ exist (range queries)

For real values, typically only range queries are used

Data Representation

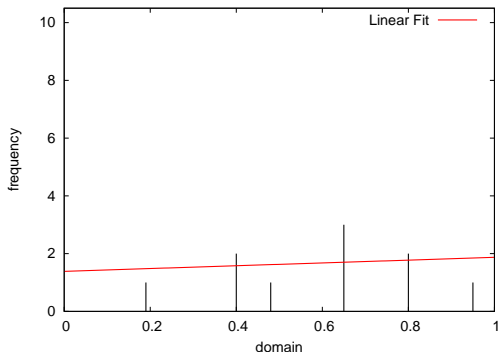
"Natural" representation: values with frequencies...



- one peak for each distinct value

Data Representation

... but hard to approximate



- fit follows data, but ignores non-existing data

Data Representation (2)

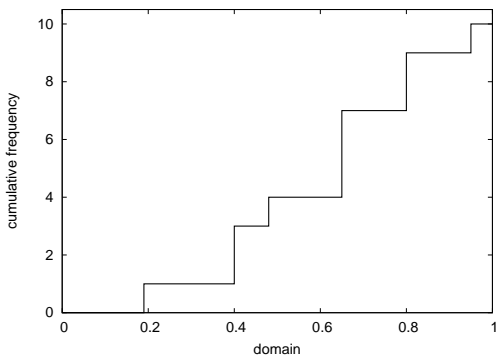
Approximating the distribution function (df) is a bad idea:

- can vary greatly, not very linear
- ignores non-existing data points
- good point queries for existing data, but bad for non-existing
- range queries very poor due to this

Fitting to the df is only reasonable if the support is discontinuous.

Data Representation (3)

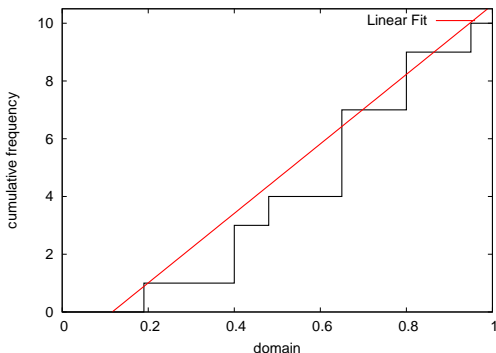
Equivalent representation: cumulative frequencies (cdf)



- monotonic increasing, contiguous support

Data Representation (3)

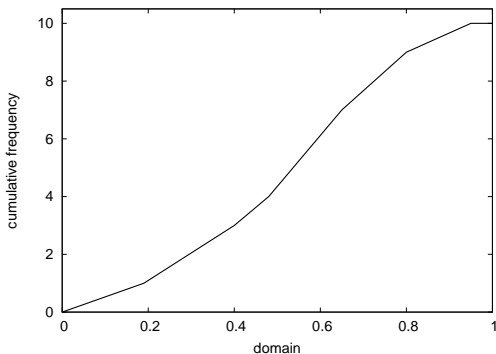
Equivalent representation: cumulative frequencies (cdf)



- plausible over the whole domain, but what to approximate?
(here: upper point)

Data Representation (4)

Simplification for large inputs: ignore the lower base of each step



- error is minor if n is large
- **already a spline**

Outline

- 1 Motivation
- 2 Data Representation
- 3 Construction
- 4 Algorithms
- 5 Evaluation

Construction Criterion

How to decide if an approximation is good?

Popular metric: Mean squared error

- well known from statistics
- can be minimized using least squares
- but mainly popular because of analytical results
- error value gives not direct insight

Can be used with least squares to construct discontinuous splines
(one line segment per bucket)

Construction Criterion (2)

Consider the following two (multi-)sets with 100 values:

Set 1

$$\{0, 0.5, 0.5, \dots, 0.5, 0.5, 1\}$$

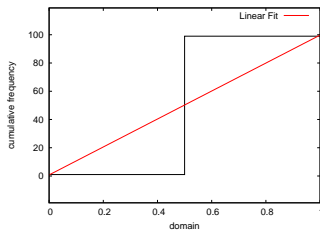
Set 2

$$\{0, 0.452, 0.453, \dots, 0.548, 0.549, 1\}$$

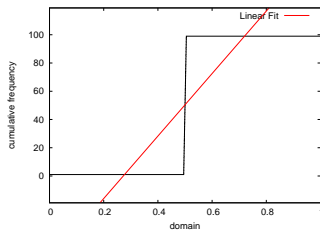
Distribution in both sets is nearly the same...

Construction Criterion (3)

...but the least squares fits are very different:



Set 1

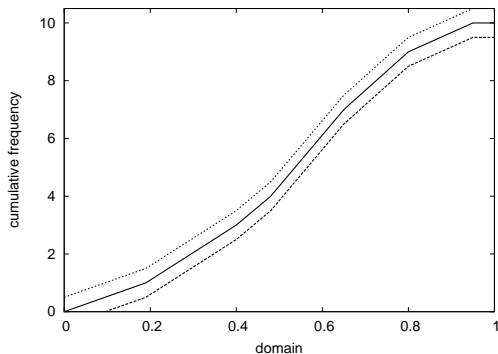


Set 2

Approximation for Set 2 is much worse at the boundaries.

Construction Criterion (4)

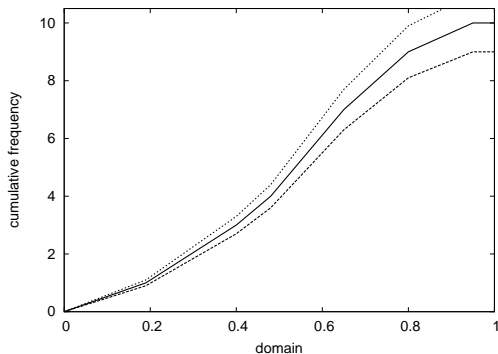
More robust error metric: Maximum error



Well defined over continuous domains; defines an error corridor.

Construction Criterion (4)

Minor variation: Maximum relative error



More useful for prediction purposes. Construction is similar.

Continuous Domains

Why consider continuous domains?

- many algorithms have problems with continuous data
- *double* is "discrete", too (64 bit)
- so why not only consider discrete domains?

Discretization is not a solution:

- 2^{64} is a lot
- very sparse integer domains have the same problems as continuous domains

Discretization into a dense domain is just as difficult as histogram construction.

Outline

- 1 Motivation
- 2 Data Representation
- 3 Construction
- 4 Algorithms
- 5 Evaluation

Algorithms

Using the simplified cdf representation, we approximate a spline S , $|S| = n$, with a spline S' , $|S'| \leq k$

- n is typically huge, k is small
- as n is very large, we assume $S' \subseteq S$
- construct S' with a minimal maximum error

Algorithms: Definitions

Error caused by a line segment: maximum error

$$\delta(S, \overline{S_1 S_2}) = \max_{p \in S \wedge p.x \in [S_1.x, S_2.x]} |p.y - \overline{S_1 S_2}[p.x]|$$

Error for approximating S by $S' \subseteq S$: Maximum of all segments

$$\Delta(S, S') = \max_{1 \leq i < |S'|} \delta(S, \overline{S'_i S'_{i+1}})$$

Objective function: $\Delta(S, S') \rightarrow \min$

Constraints: $|S'| \leq k \wedge S'[1] = S[1] \wedge S'[|S'|] = S[|S|]$

Algorithms: Greedy

Greedy approach: Choose the next knot point that minimizes the overall error

- requires predicting the error of the following knots
- difficult to do, therefore we solve the dual problem first

Dual problem: Find the smallest spline S' that approximates S with an error $\leq \epsilon$

Can be used to solve the primal in a following step

Algorithms: Greedy (2)

Find the smallest spline S' that approximates S with an error $\leq \epsilon$:

Choose a point as a knot if the next point would violate the error corridor.

$$S'[1] = S[1]$$

for $i = 2$ **to** n

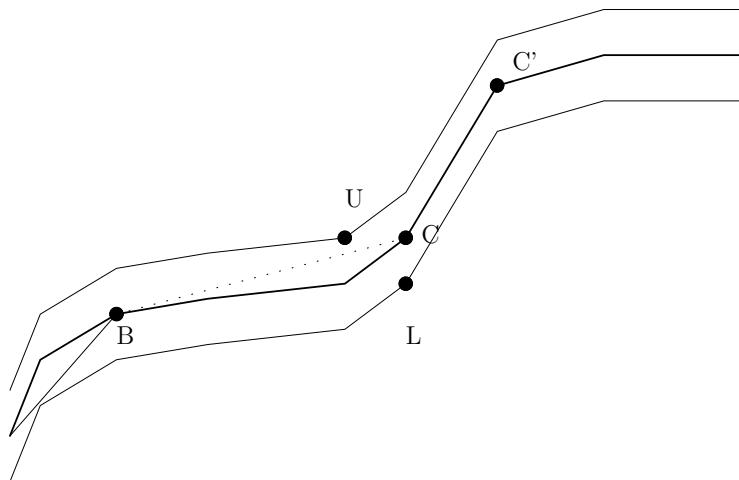
if $\delta(S, \overline{S'_{|S'|} S_i}) > \epsilon$

$$S' = S' \circ S_{i-1}$$

$$S' = S' \circ S[n]$$

- places knot points as late as possible
- the error corridor can be computed incrementally by remembering upper and lower bounds (U, L)

Algorithms: Greedy (3)



U, L form the error corridor. $\overline{BC'}$ violates it \rightarrow add C

Algorithms: Greedy (4)

Now we can use the dual to solve the primal.

Binary search over ϵ :

$$\epsilon_L = 0, \epsilon_U = \max\{P.y \mid P \in S\}$$

do

$$\epsilon = \frac{\epsilon_L + \epsilon_U}{2}$$

if $|\text{GreedyDual}(S, \epsilon)| > k$

$\epsilon_L = \epsilon$ **else**

$\epsilon_U = \epsilon$

while $|\text{GreedyDual}(S, \epsilon_L)| \neq |\text{GreedyDual}(S, \epsilon_U)|$

return $\text{GreedyDual}(S, \epsilon_L)$

- produces good results in $O(n \log n)$, but only a heuristic

Algorithms: DP

DP algorithm constructs the optimal spline that satisfies the constraints.

- runtime is in $O(n^2)$
- too slow if n is huge

But:

- both algorithms transform splines into smaller splines
- we can use the fast greedy algorithm as a prefilter
- reduce input to 5000 knots, then use DP

Outline

- 1 Motivation
- 2 Data Representation
- 3 Construction
- 4 Algorithms
- 5 Evaluation

Evaluation

GOV data

- term lists, normalized PageRank*TFIDF ($[0, 1]$)
- list *public*: 427,940 entries with 359,505 unique values

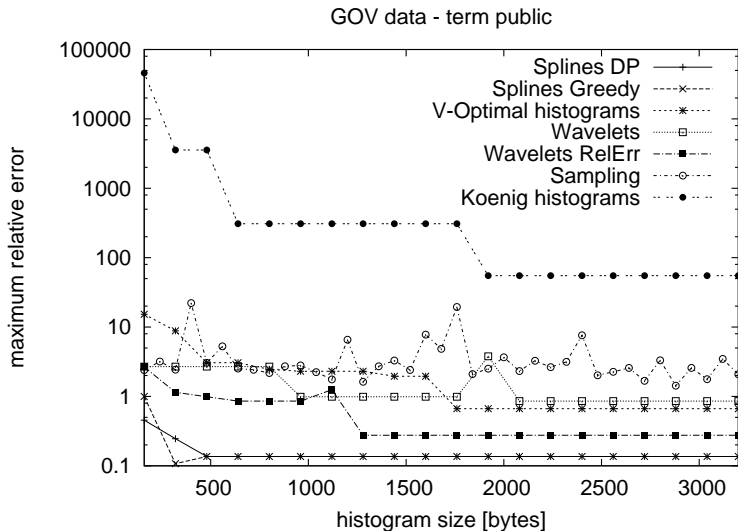
Queries:

- range queries of the form $[c, \infty]$
- Scheme 1: uniformly sample c in $[0, 1]$ (10^6 points)
- Scheme 2: choose all unique values as c one

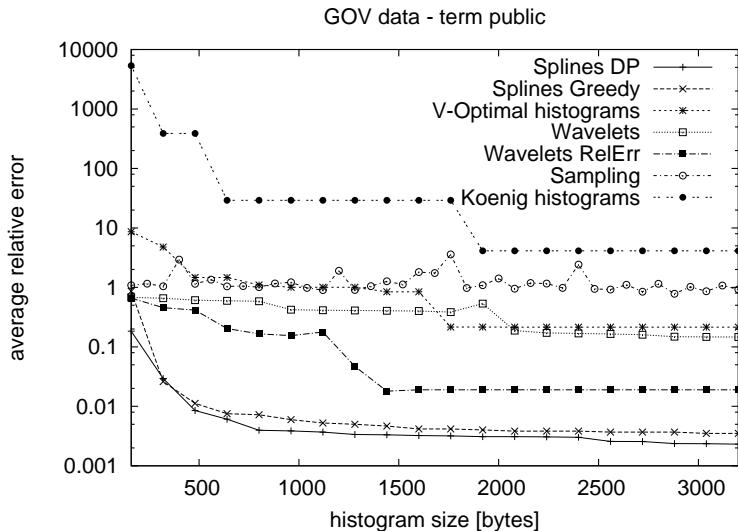
Approaches: Splines Greedy/DP, V-Optimal Histograms, Wavelets Normal/RelError, Sampling, Koenig Splines.

Equi-Width and Equi-Depth were dominated by V-Optimal.

Evaluation: Maximum Relative Error



Evaluation: Average Relative Error



Evaluation: Scheme 2

Error Metrics at 3200 Bytes for Scheme 2:

	max rel.	avg rel.	mean sq.
DP Spline	0.0096	0.0008	50,029
Greedy Spline	0.0040	0.0024	497,503
V-Optimal	0.4880	0.0034	162,246
Wavelet	0.8571	0.0244	$5.4 * 10^7$
Wavelet RelErr	0.2762	0.1001	$8.7 * 10^8$
Koenig	55.068	0.0005	435
Sampling	215,012	4.49839	$1.1 * 10^8$

- most approaches are much better when only asking for existing data points
- caused by fitting to the data

Evaluation: Effect on Queries

GOV benchmark queries

- distributed top-k QP: optimizing query execution plan
- best 3 approaches (both splines gave the same results)
- just exchanged the histograms, the rest of the optimizer was unchanged

	Splines	V-Optimal	Wavelet RelErr
runtime [ms]	585	649	618
net [bytes]	23,918	39,660	26,715

Note: Wavelet RelErr required > 10 min and $> 2.5GB$ memory for intermediate result histograms. The others required < 20 ms.

Conclusion

Results:

- very good predictions, bound error
- still relatively easy to compute
- improves plan quality

Open problems:

- faster spline construction
- multi-dimensional histograms
- correlations after convolutions