

15-826: Multimedia (Databases) and Data Mining

Lecture #10: Fractals - case studies

C. Faloutsos

Must-read Material - I

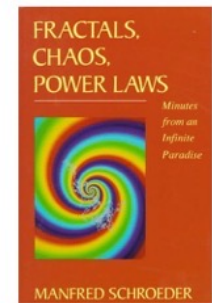
- Christos Faloutsos and Ibrahim Kamel, *Beyond Uniformity and Independence: Analysis of R-trees Using the Concept of Fractal Dimension*, Proc. ACM SIGACT-SIGMOD-SIGART PODS, May 1994, pp. 4-13, Minneapolis, MN.

Must-read Material - II

- Bernd-Uwe Pagel, Flip Korn and Christos Faloutsos, *Deflating the Dimensionality Curse using Multiple Fractal Dimensions*, ICDE 2000, San Diego, CA, Feb. 2000.

Optional Material

Optional, but **very** useful: Manfred Schroeder
*Fractals, Chaos, Power Laws: Minutes
from an Infinite Paradise* W.H. Freeman
and Company, 1991



Reminder

- Code at

www.cs.cmu.edu/~christos/SRC/fdnq_h.zip

Also, in ‘R’

```
> library(fractaldim);
```



Outline

Goal: ‘Find **similar / interesting** things’

- Intro to DB
- ➔ • Indexing - similarity search
- Data Mining



Indexing - Detailed outline

- primary key indexing
- secondary key / multi-key indexing
- spatial access methods
 - z-ordering
 - R-trees
 - misc
- fractals
 - intro
 - applications
- text





Indexing - Detailed outline

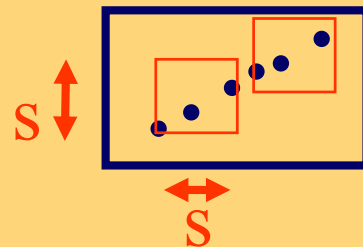
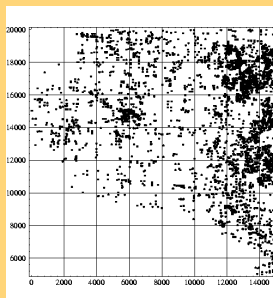
- fractals
 - intro
 - applications
 - disk accesses for R-trees (range queries)
 - dim. curse revisited
 - nearest neighbors estimation





Problem:

- Selectivity of a range query in R-trees?

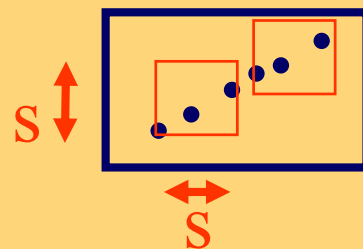
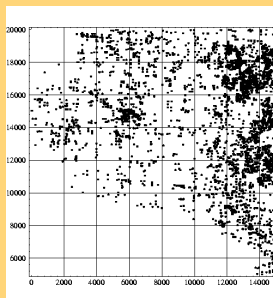




Solution:

- Selectivity of a range query in R-trees?
- Depends on *fractal* dimension

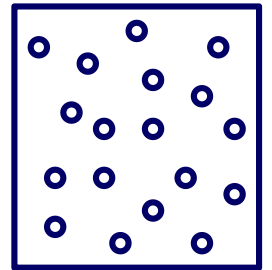
$$s = (C/N)^{1/D_0}$$



Case study#1: R-tree performance

Problem

- Given
 - N points in E -dim space
- Estimate # disk accesses for a range query
($q_1 \times \dots \times q_E$)

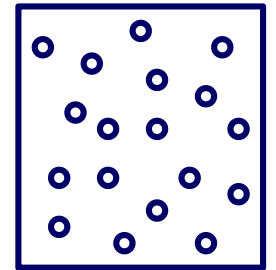


(assume: ‘good’ R-tree, with tight, cube-like MBRs)

Case study#1: R-tree performance

Problem

- Given
 - N points in E -dim space
- Estimate # disk accesses for a range query
($q_1 \times \dots \times q_E$)

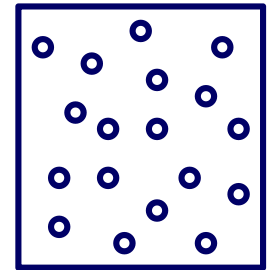


(assume: ‘good’ R-tree, with tight, cube-like MBRs)
Typically, in DB Q-opt?

Case study#1: R-tree performance

Problem

- Given
 - N points in E -dim space
- Estimate # disk accesses for a range query
($q_1 \times \dots \times q_E$)

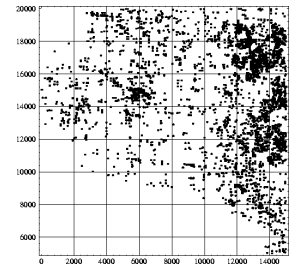


(assume: ‘good’ R-tree, with tight, cube-like MBRs)
Typically, in DB Q-opt: uniformity + independence

Case study#1: R-tree performance

Problem

- Given
 - N points in E-dim space
 - with fractal dimension D
- Estimate # disk accesses for a range query
($q_1 \times \dots \times q_E$)

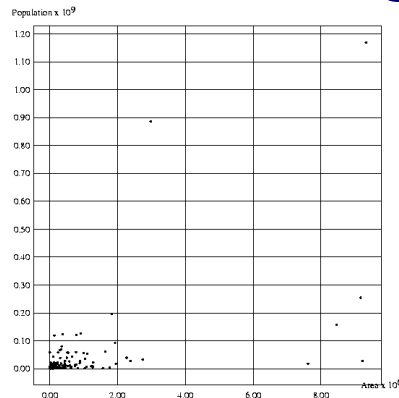


(assume: ‘good’ R-tree, with tight, cube-like MBRs)
Typically, in DB Q-opt: uniformity + independence

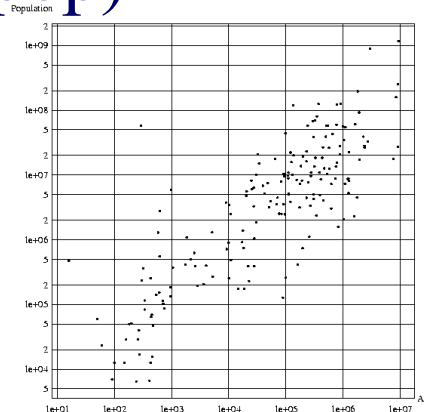
Examples: World's countries

- BUT: area vs population for ~ 200 countries (1991 CIA fact-book).

pop



log(pop)



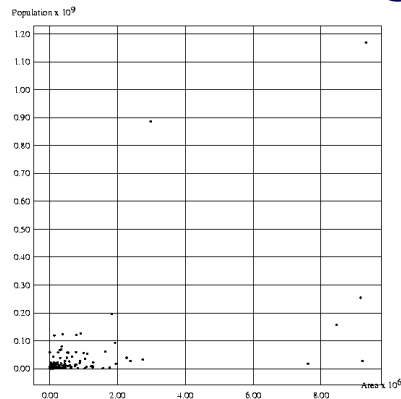
area

log(area)

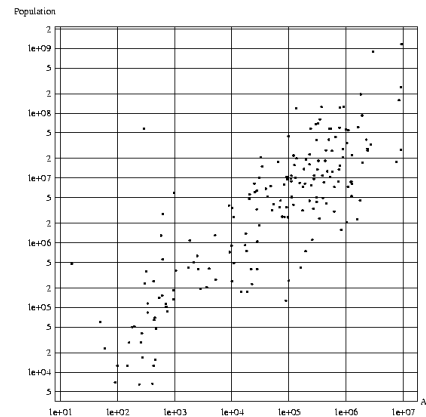
Examples: World's countries

- neither uniform, nor independent!

pop



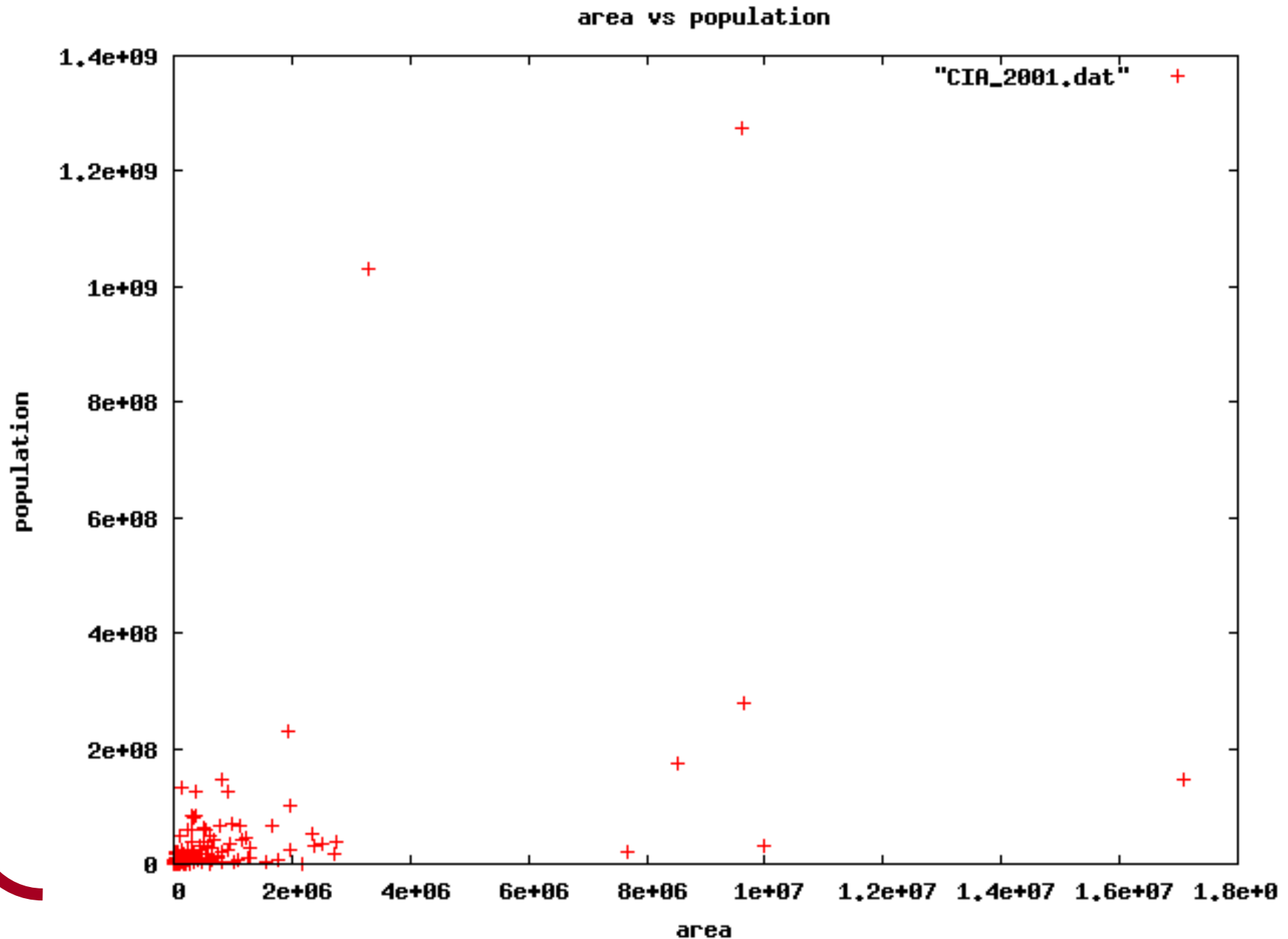
log(pop)



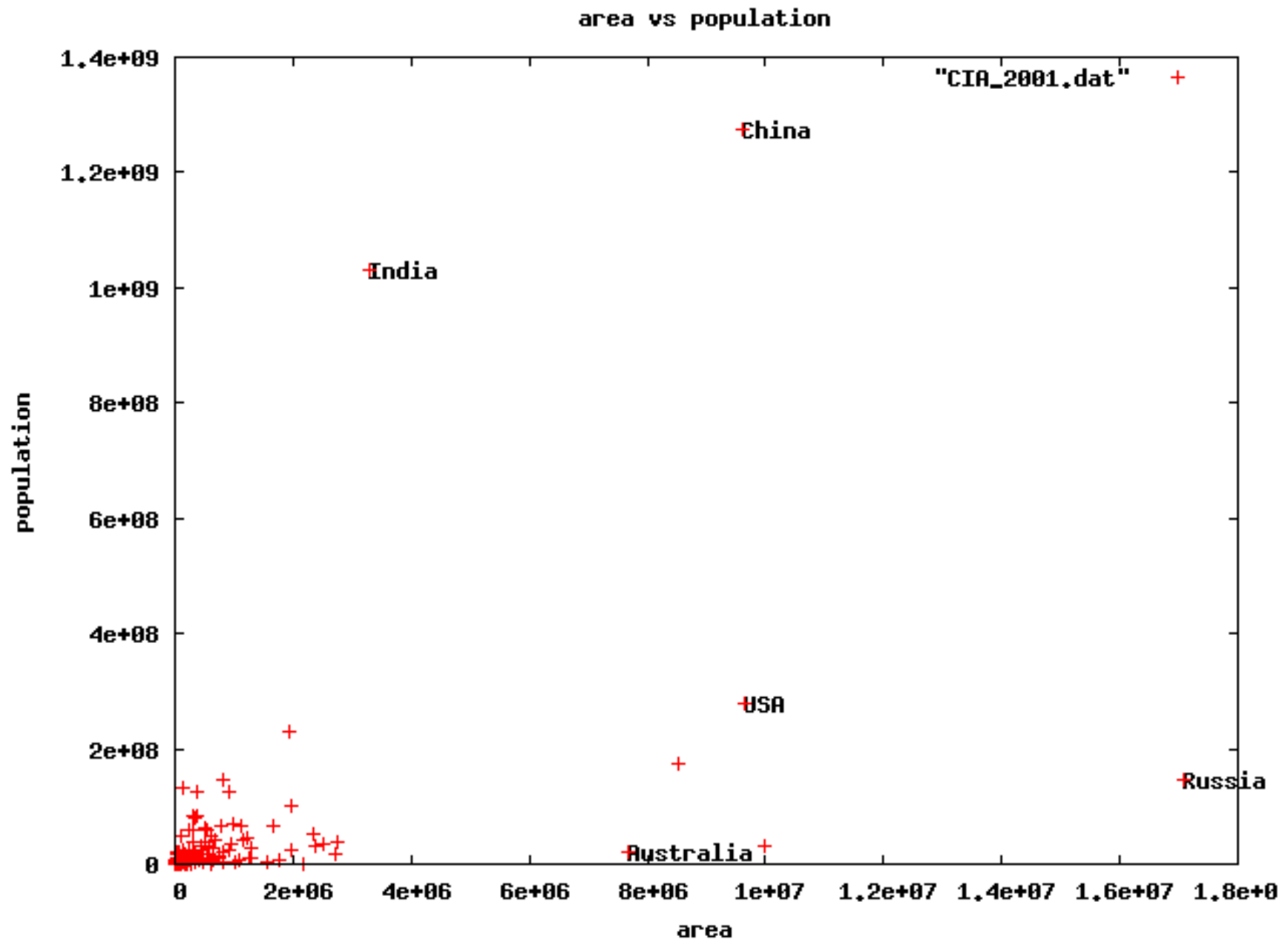
area

log(area)

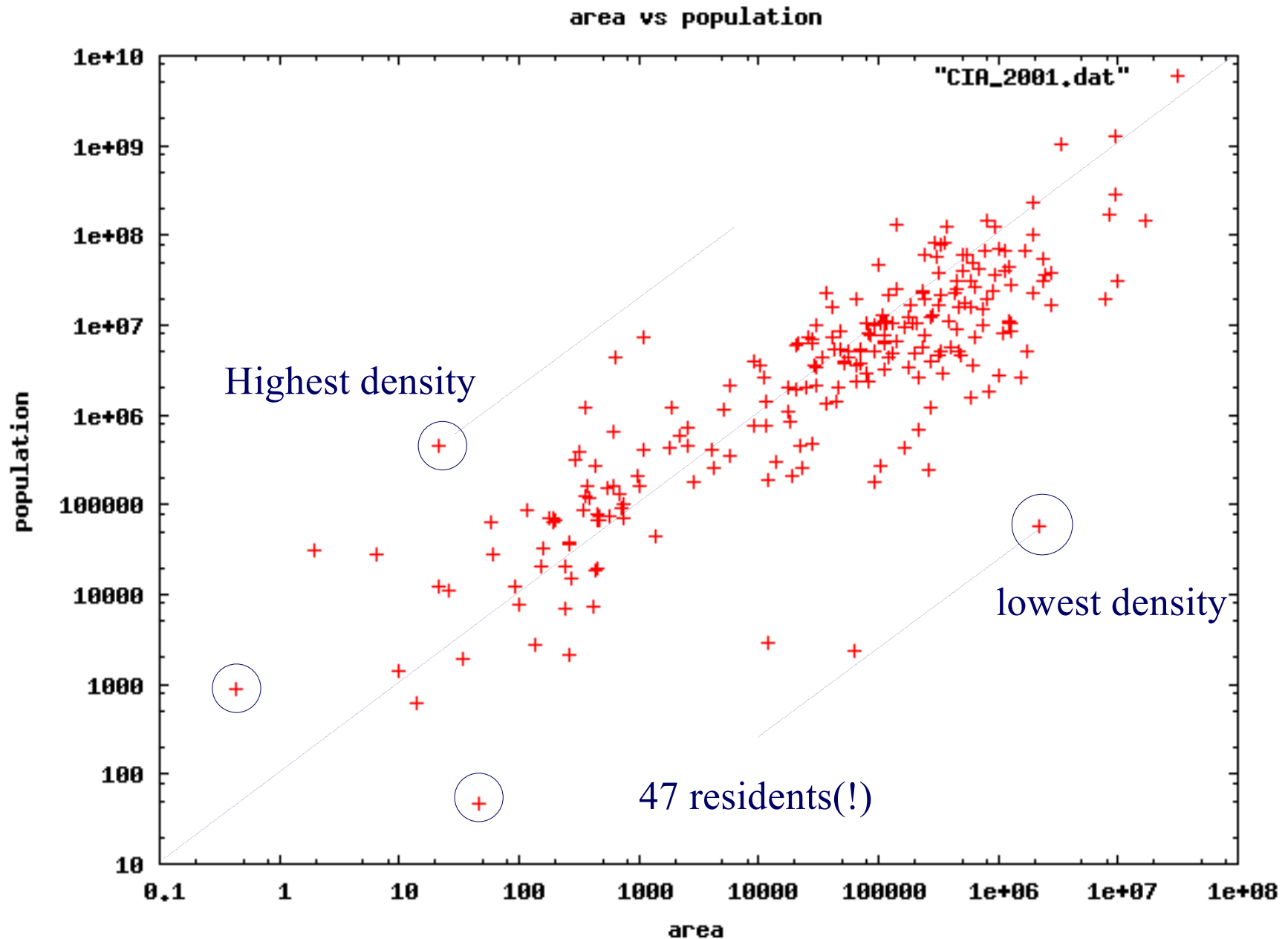
For fun: identification



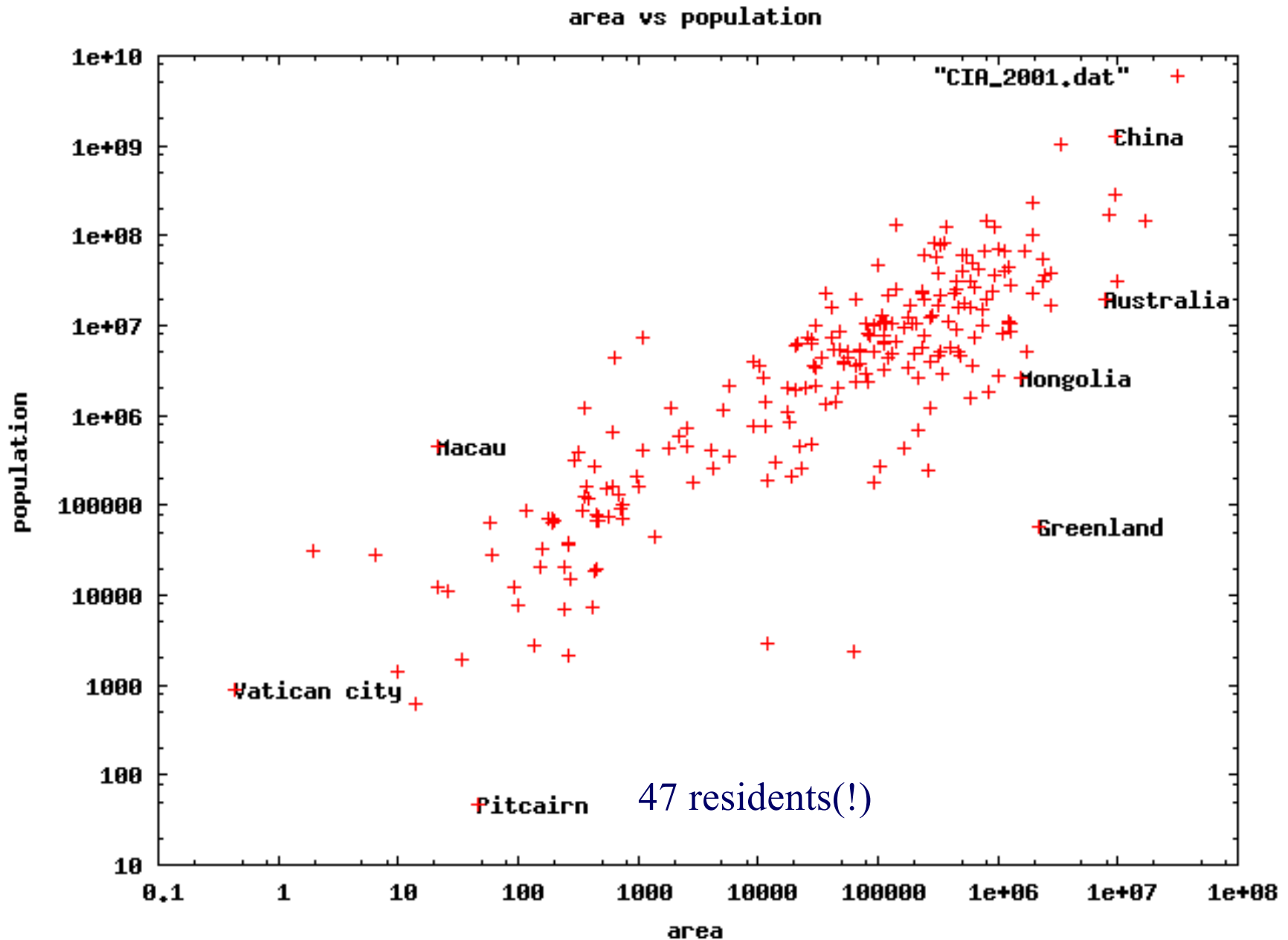
For fun: identification



For fun: identification



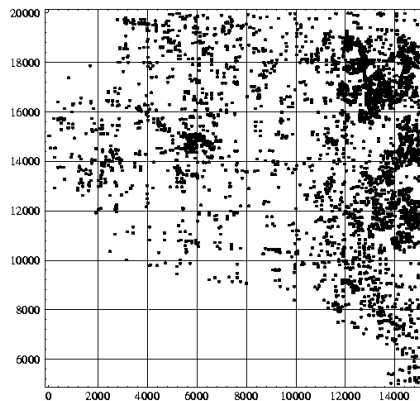
For fun: identification



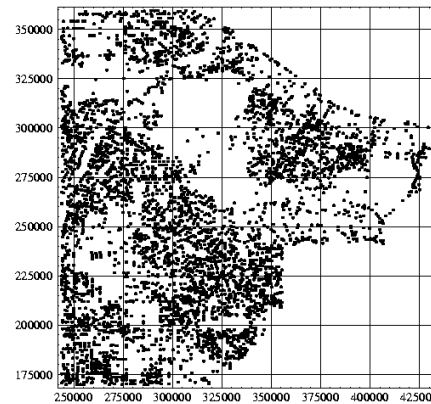
Examples: TIGER files

- neither uniform, nor independent!

MG county



LB county



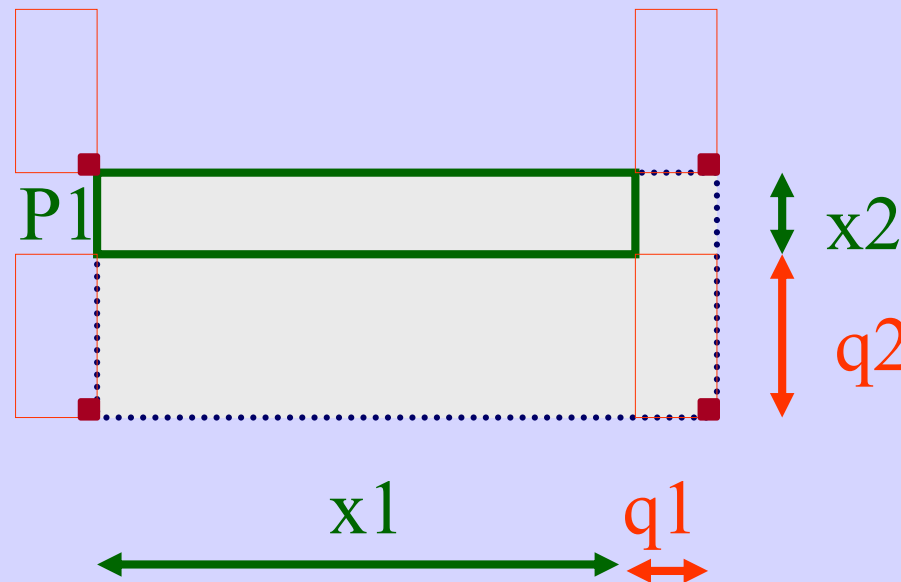
How to proceed?

- recall the [Pagel+] formula, for range queries of size $q1 \times q2$

$$\#DiskAccesses(q1, q2) = \sum (x_{i,1} + q1) * (x_{i,2} + q2)$$

R-trees - performance analysis

- How many times will P1 be retrieved (unif. queries of size $q_1 \times q_2$)?



How to proceed?

- recall the [Pagel+] formula, for range queries of size $q1 \times q2$

$$\#DiskAccesses(q1, q2) = \sum (x_{i,1} + q1) * (x_{i,2} + q2)$$

But:

formula needs to know the $x_{i,j}$ sizes of MBRs!

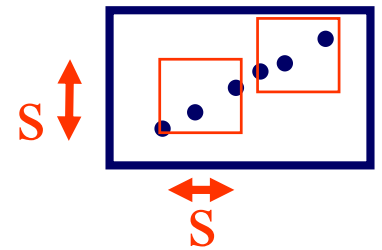
How to proceed?

But:

formula needs to know the $x_{i,j}$ sizes of MBRs!

Answer (jumping ahead):

$$s = (C/N)^{1/D0}$$



How to proceed?

But:

formula needs to know the $x_{i,j}$ sizes of MBRs!

Answer (jumping ahead):

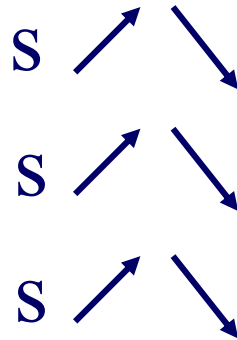
$$s = (C/N)^{1/D_0}$$

Diagram illustrating the formula $s = (C/N)^{1/D_0}$ and its components:

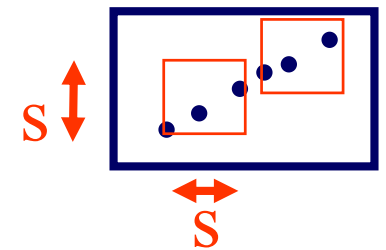
- s : side of (parent) MBR
- C : page capacity
- N : # of data points
- D_0 : Hausdorff dimension

‘smell’ tests:

- C ↗
- N ↗
- D0 ↗



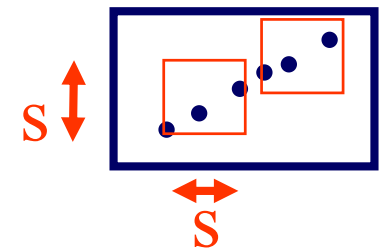
$$s = (C/N)^{1/D0}$$



'smell' tests:

- C ↗ s ↗
- N ↗ s ↘
- D0 ↗ s ↗

$$s = (C/N)^{1/D0}$$

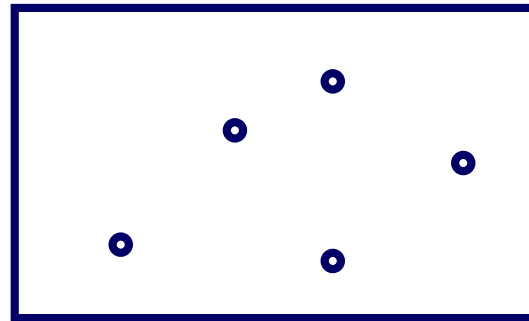
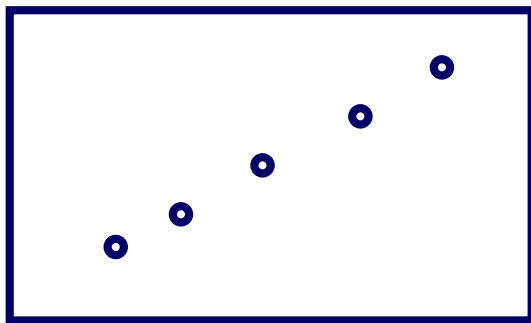


R-trees - performance analysis

I.e: for range queries - how many disk accesses,
if we just now that we have

- N points in E -d space?

A: can not tell! need to know distribution

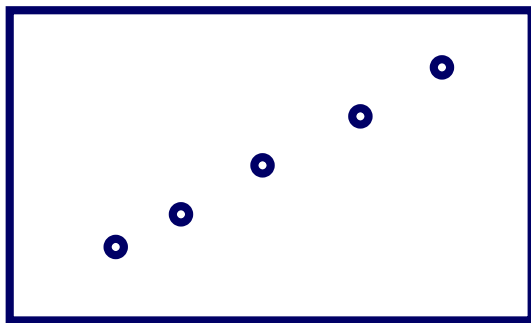


R-trees - performance analysis

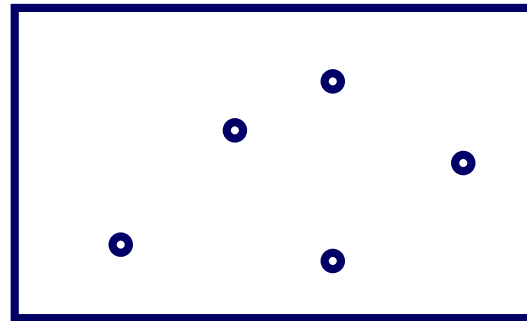
Q: OK - so we are told that the **Hausdorff** fractal dim. = D_0 - Next step?

(also know that there are at most C points per page)

$D_0=1$



$D_0=2$



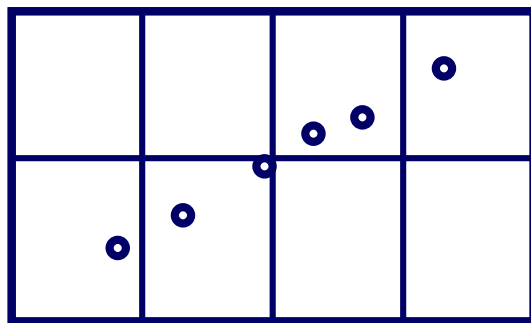
R-trees - performance analysis

Assumption1: square-like parents ($s*s$)

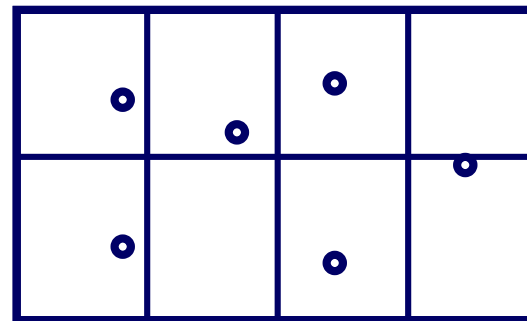
Assumption2: fully packed (C points each)

Assumption3: non-overlapping

$D0=1$



$D0=2$



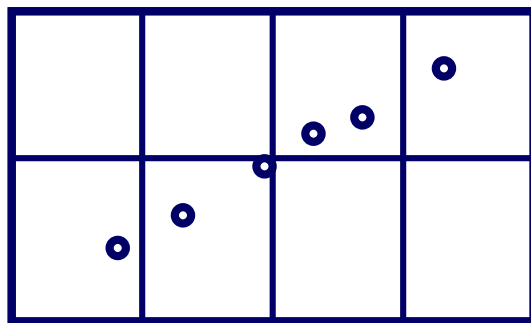
R-trees - performance analysis

Assumption1: square-like parents ($s*s$)

Assumption2: fully packed (N/C non-empty)

Assumption3: non-overlapping

$D_0=1$



$s_1=s_2=s$

R-trees - performance analysis

Hint: dfn of Hausdorff f.d.:



Felix Hausdorff (1868-1942)

Reminder:

Hausdorff or box-counting fd:

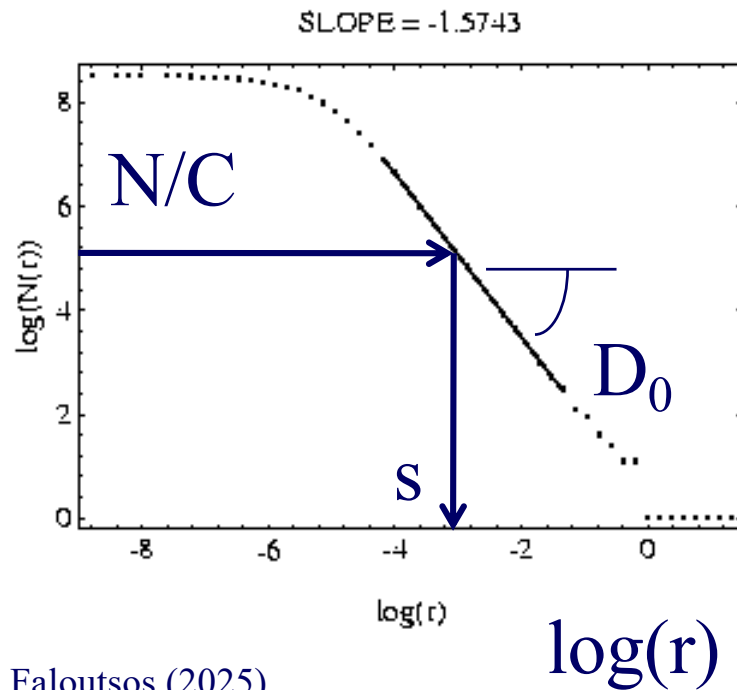
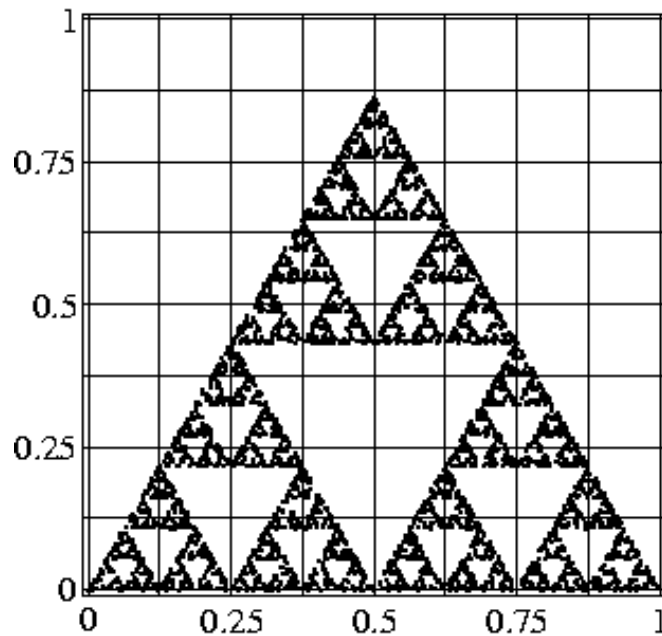
- Box counting plot: $\text{Log}(N(r))$ vs $\text{Log}(r)$
- r : grid side
- $N(r)$: count of non-empty cells
- (Hausdorff) fractal dimension D_0 :

$$D_0 = -\frac{\partial \log(N(r))}{\partial \log(r)}$$

Reminder

- Hausdorff fd:

$$r \sim \log(\#\text{non-empty cells})$$



Reminder

- dfn of Hausdorff fd implies that

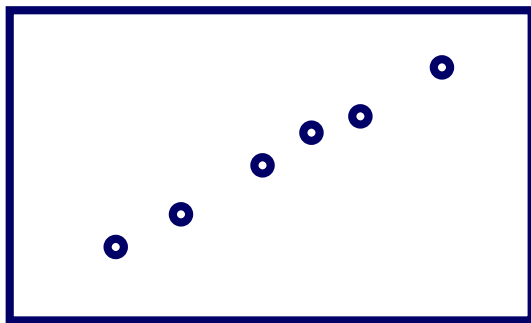
$$\nearrow N(r) \sim r^{(-D_0)}$$

non-empty cells of side r

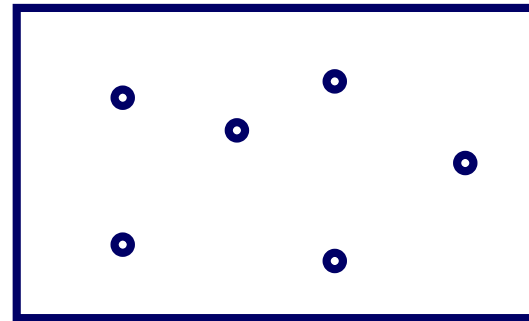
R-trees - performance analysis

Q (rephrased): what is the side $s_1, s_2, \dots$ of parent nodes, given N data points, packed by C , with f.d. = D_0

$D_0=1$

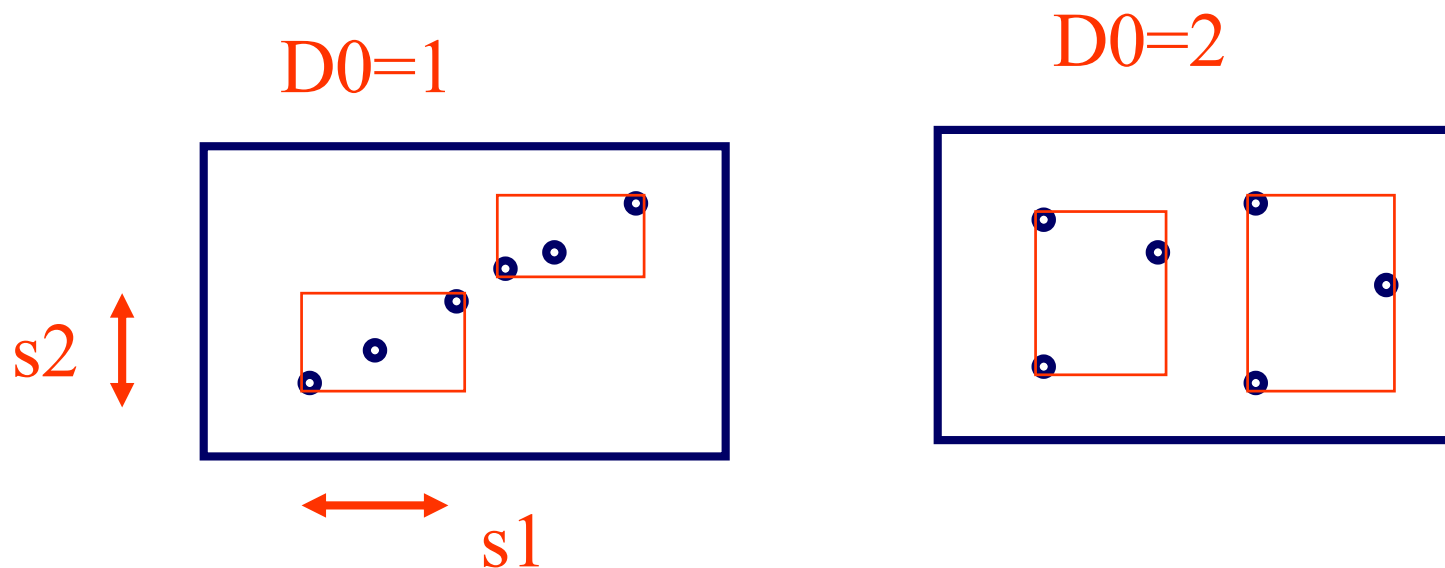


$D_0=2$



R-trees - performance analysis

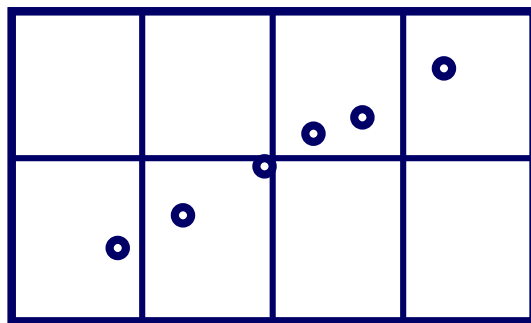
Q (rephrased): what is the side $s_1, s_2, \dots$ of parent nodes, given N data points, packed by C , with f.d. = D_0



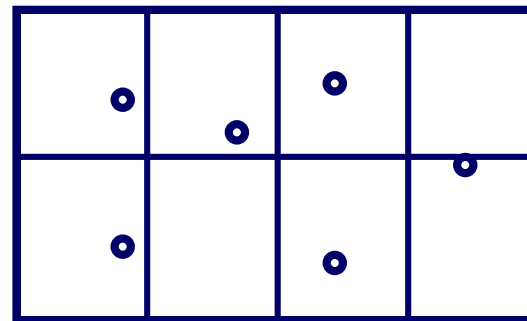
R-trees - performance analysis

Q (rephrased): what is the side $s_1, s_2, \dots$ of parent nodes, given N data points, packed by C , with f.d. = D_0

$D_0=1$



$D_0=2$



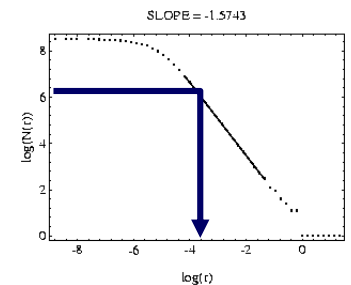
$s_1 \overset{\longleftrightarrow}{=} s_2 = s$

R-trees - performance analysis

A: (educated guess)

- $s=s_1=s_2$ (= ...) - square-like MBRs
- N/C non-empty cells = $K * s^{(-D_0)}$

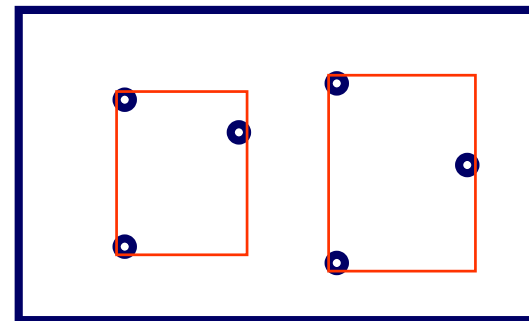
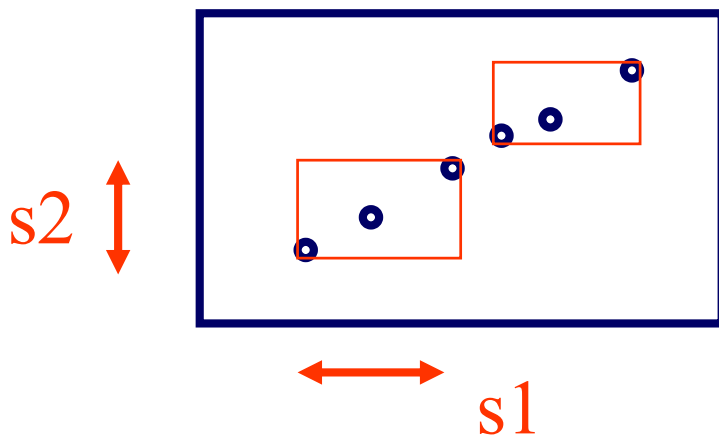
$\log(\#cells)$



$\log(s)$

$D_0=1$

$D_0=2$



R-trees - performance analysis

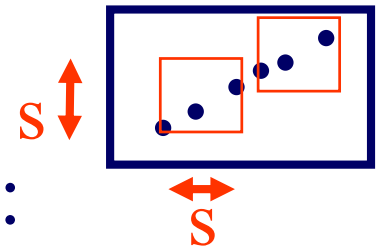
PROOF of derivations: in [PODS 94].

Finally, expected side s of parent MBRs:

$$s = (C/N)^{1/D0}$$

Q: sanity check: how does s change with $D0$?

A:





R-trees - performance analysis

PROOF of derivations: in [Kamel+, PODS 94]

Finally, expected side s of parent MBRs:

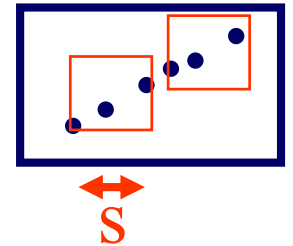
$$s = (C/N)^{1/D_0}$$

Q: sanity check: how does s change with D_0 ?

A: s grows with D_0

Q: does it make sense?

Q: does it suffer from (intrinsic) dim. curse?



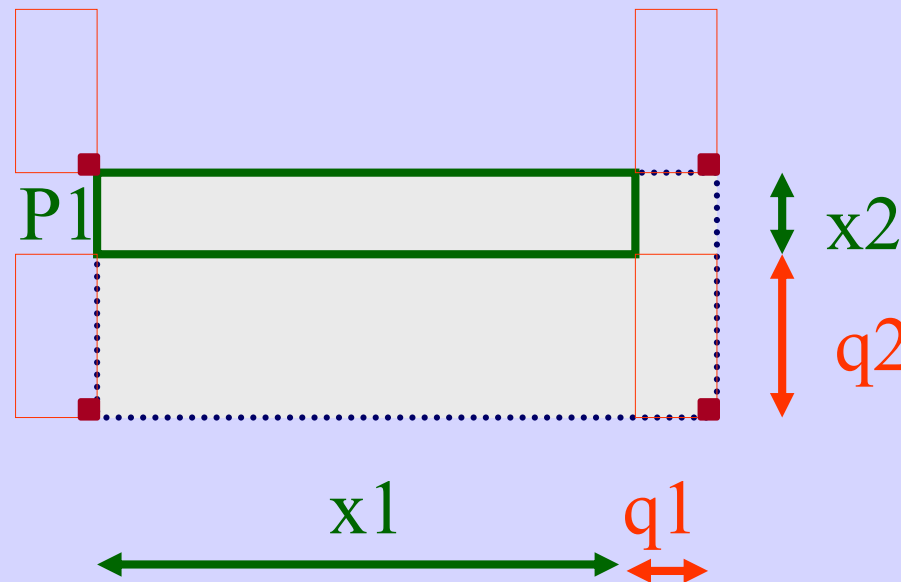
R-trees - performance analysis

Q: Final-final formula (# disk accesses for range queries $q1$ x $q2$ x ...):

A:

R-trees - performance analysis

- How many times will P1 be retrieved (unif. queries of size $q_1 \times q_2$)?



R-trees - performance analysis

Q: Final-final formula (# disk accesses for range queries $q1 \times q2 \times \dots$):

A: # of parent-node accesses:

$$N/C * (s + q1) * (s + q2) * \dots (s + q_E)$$

A: # of grand-parent node accesses

R-trees - performance analysis

Q: Final-final formula (# disk accesses for range queries $q1 \times q2 \times \dots$):

A: # of parent-node accesses:

$$N/C * (s + q1) * (s + q2) * \dots (s + q_E)$$

A: # of grand-parent node accesses

$$N/(C^2) * (s' + q1) * (s' + q2) * \dots (s' + q_E)$$

$$s' = ??$$

R-trees - performance analysis

Q: Final-final formula (# disk accesses for range queries $q1 \times q2 \times \dots$):

A: # of parent-node accesses:

$$N/C * (s + q1) * (s + q2) * \dots (s + q_E)$$

A: # of grand-parent node accesses

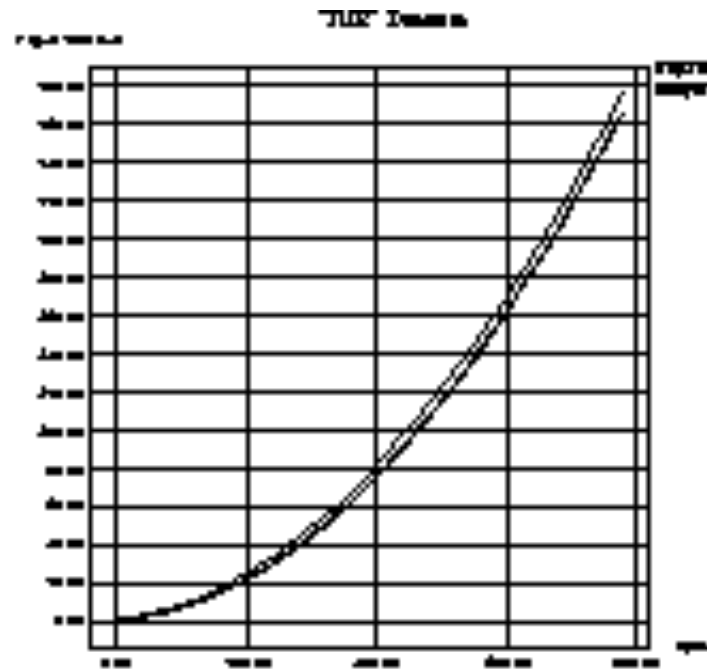
$$N/(C^2) * (s' + q1) * (s' + q2) * \dots (s' + q_E)$$

$$s' = (C^2/N)^{1/D0}$$

R-trees - performance analysis

Results: IUE (x-y star coordinates)

leaf accesses



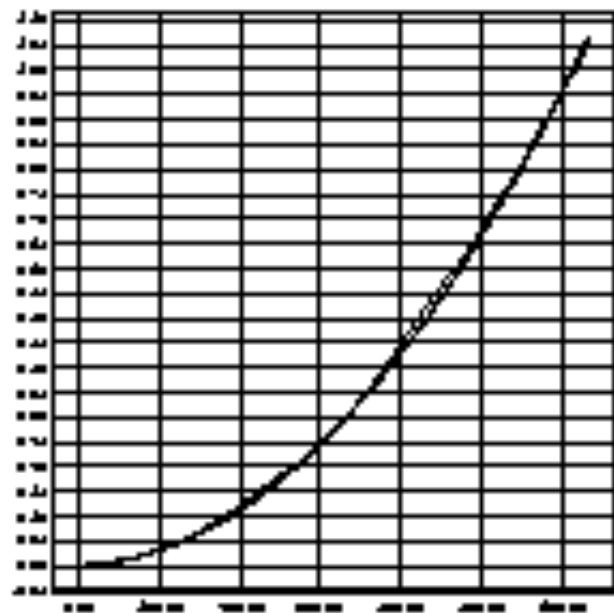
(a) IUE - Leaf accesses vs. query side

query side

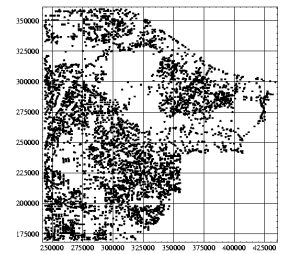
R-trees - performance analysis

Results: LB County

leaf accesses



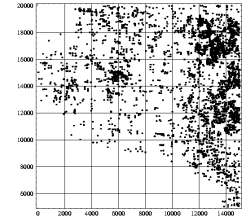
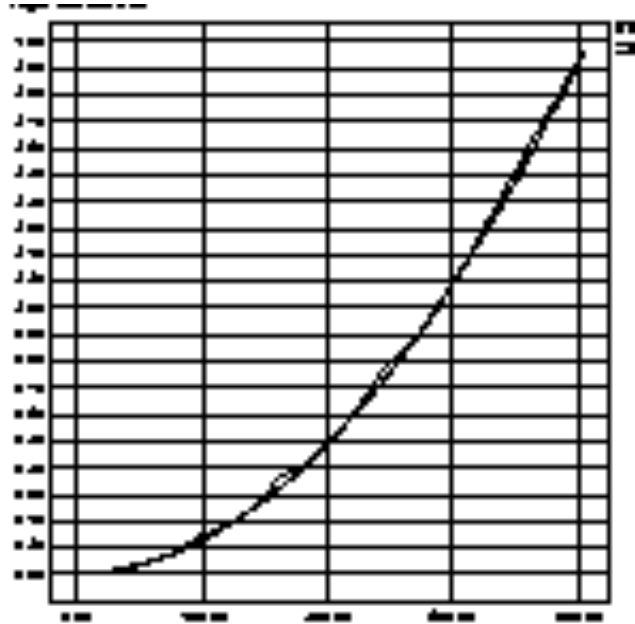
query side



R-trees - performance analysis

Results: MG-county

leaf accesses

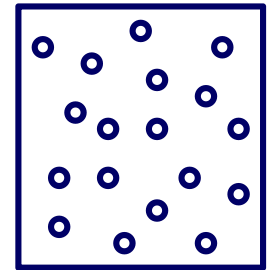
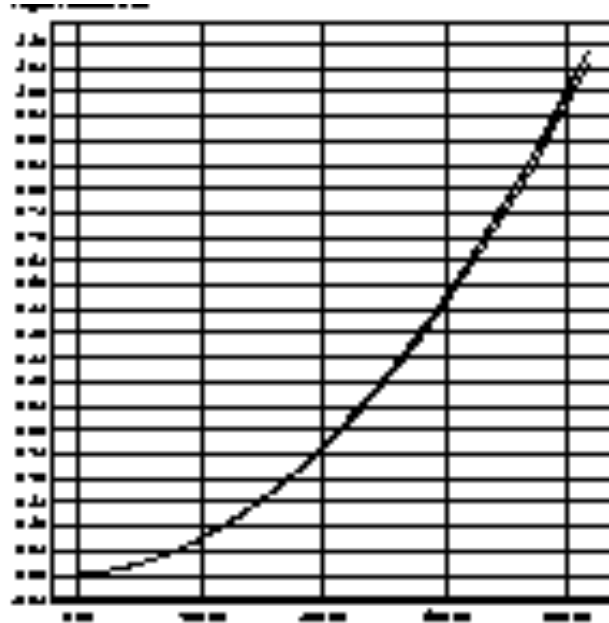


query side

R-trees - performance analysis

Results: 2D- uniform

leaf accesses



query side

R-trees - performance analysis

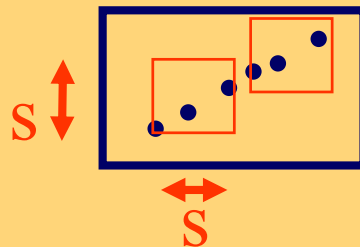
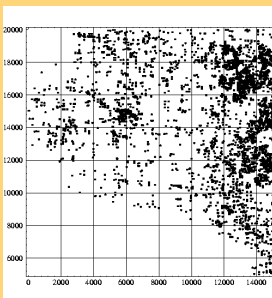
Conclusions: usually, $<5\%$ relative error, for range queries



Solution:

- Selectivity of a range query in R-trees?
- Depends on *fractal* dimension

$$s = (C/N)^{1/D_0}$$





Indexing - Detailed outline

- primary key indexing
- secondary key / multi-key indexing
- spatial access methods
 - z-ordering
 - R-trees
 - misc
- ➔ • fractals
 - intro
 - applications
- text



Indexing - Detailed outline

- fractals
 - intro
 - applications
 - disk accesses for R-trees (range queries)
 - dim. curse revisited
 - nearest neighbors estimation



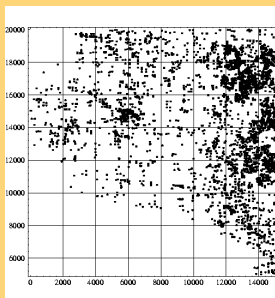
Must-read Material

- Bernd-Uwe Pagel, Flip Korn and Christos Faloutsos, *Deflating the Dimensionality Curse using Multiple Fractal Dimensions*, ICDE 2000, San Diego, CA, Feb. 2000.



Problem:

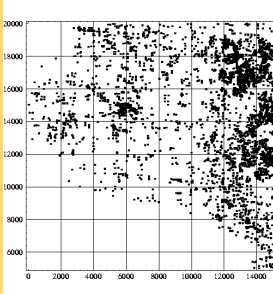
- Q: Do all S.A.M. suffer in high dimensions?
- Q: what to do?





Solutions:

- Q: Do all S.A.M. suffer in high dimensions?
- A: Only in high *fractal* dimensions
- Q: what to do?
- A: dim-reduction; approximate knn; etc

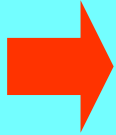


$$P_{all}^{L\infty}(k) \approx \sum_{j=0}^h \left\{ \frac{1}{C^{h-j}} + \left[1 + \left(\frac{k}{C^{h-j}} \right)^{1/D} \right]^D \right\}$$



Indexing - Detailed outline

- fractals
 - intro
 - applications
 - disk accesses for R-trees (range queries)
 - dim. curse revisited
 - nearest neighbors estimation



Dimensionality ‘curse’

- Q: What is the problem in high-d?

Dimensionality ‘curse’

- Q: What is the problem in high-d?
- A: indices do not seem to help, for many queries (eg., k-nn)
 - in high-d (& uniform distributions), most points are equidistant $\rightarrow$ k-nn retrieves too many near-neighbors
 - [Yao & Yao, '85]: search effort $\sim O(N^{(1-1/d)})$

- Yao, A. C. and F. F. Yao (May 6-8, 1985). A General Approach to d-Dimensional Geometric Queries. Proc. of the 17th Annual ACM Symposium on Theory of Computing (STOC), Providence, RI.

Dimensionality ‘curse’

- (counter-intuitive, for db mentality)
- Q: What to do, then?

Dimensionality ‘curse’

- A1: switch to seq. scanning
- A2: dim. reduction
- A3: consider the ‘intrinsic’ /fractal dimensionality
- A4: find *approximate* nn

Dimensionality ‘curse’

- A1: switch to seq. scanning
 - X-trees [Kriegel+, VLDB 96]
 - VA-files [Schek+, VLDB 98], ‘test of time’ award

Dimensionality ‘curse’

- A1: switch to seq. scanning
- ➔ • A2: dim. reduction
- A3: consider the ‘intrinsic’ /fractal dimensionality
- A4: find approximate nn

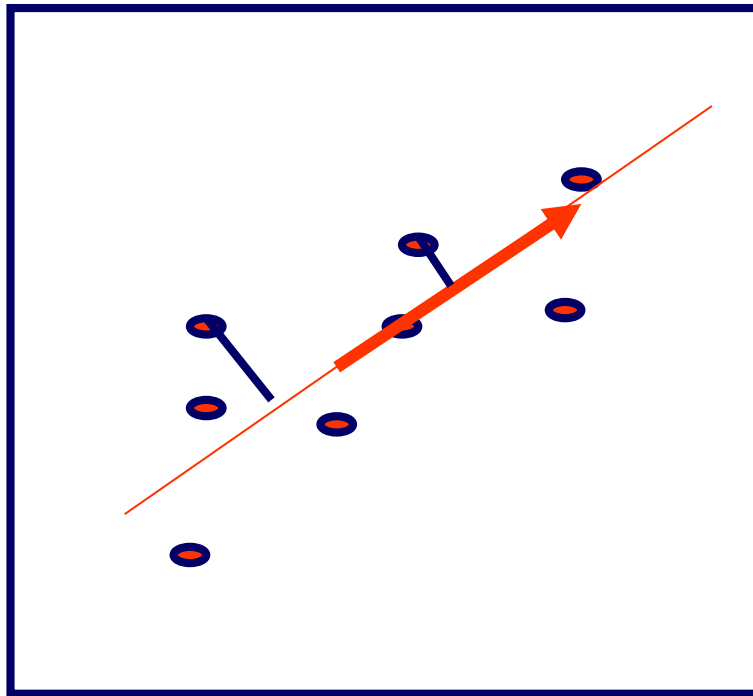
Dim. reduction

a.k.a. feature selection/extraction:

- SVD (optimal, to preserve Euclidean distances)
- random projections
- using the fractal dimension [Traina+SBBD2000]

Singular Value Decomposition (SVD)

- SVD ($\sim$ LSI $\sim$ KL $\sim$ PCA $\sim$ spectral analysis...)



LSI: S. Dumais; M. Berry

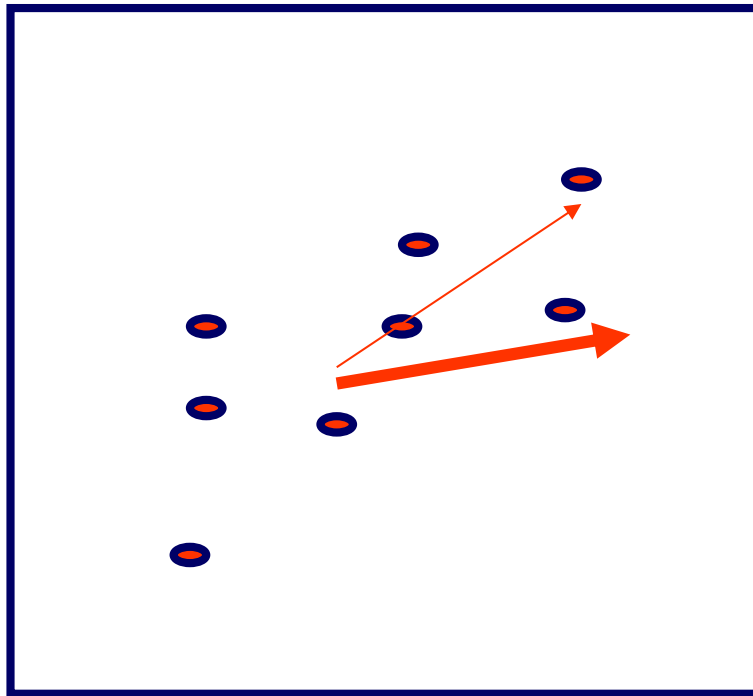
KL: eg, Duda+Hart

PCA: eg., Jolliffe

MANY more PROOF: soon

Random projections

- random projections(Johnson-Lindenstrauss thm [Papadimitriou+ pods98])



Random projections

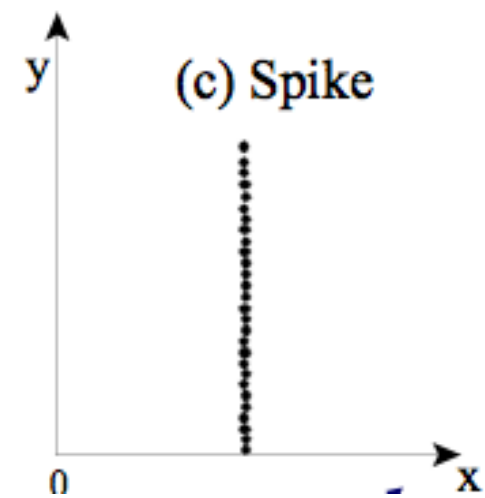
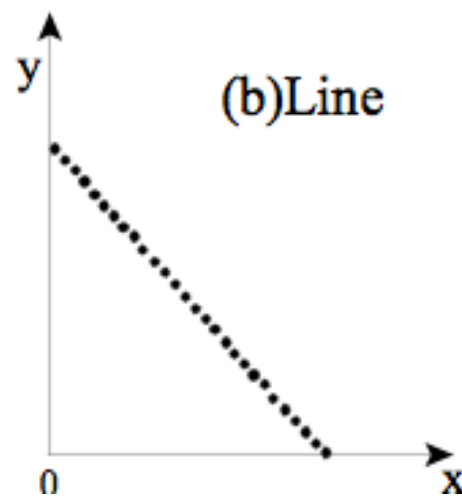
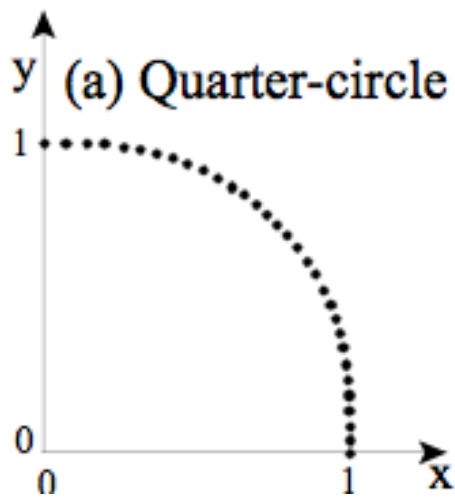
- pick ‘enough’ random directions (will be $\sim$ orthogonal, in high-d!!)
- distances are preserved probabilistically, within epsilon
- (also, use as a pre-processing step for SVD [Papadimitriou+ PODS98])

Dim. reduction - w/ fractals

- Main idea: drop those attributes that don't affect the intrinsic ('fractal') dimensionality [Traina+, SBBD 2000]

Dim. reduction - w/ fractals

global FD=1



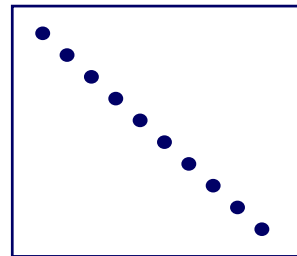
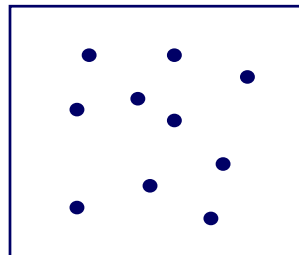
Dimensionality ‘curse’

- A1: switch to seq. scanning
- A2: dim. reduction
- ➔ • A3: consider the ‘intrinsic’ /fractal dimensionality
- A4: find **approximate** nn

Intrinsic dimensionality

- before we give up, compute the intrinsic dim.:
- the lower, the better... [Pagel+, ICDE 2000]
- more PROOF: in a few foils

intr. $d = 2$



intr. $d = 1$

Dimensionality ‘curse’

- A1: switch to seq. scanning
- A2: dim. reduction
- A3: consider the ‘intrinsic’ /fractal dimensionality
- ➔ • A4: find approximate nn

Approximate nn

- [Arya + Mount, SODA93], [Patella+ ICDE 2000]
- Idea: find k neighbors, such that the distance of the k -th one is guaranteed to be within epsilon of the actual.



Indexing - Detailed outline

- fractals
 - intro
 - applications
 - disk accesses for R-trees (range queries)
 - dim. curse revisited
 - nearest neighbors estimation



Estimation of knn effort

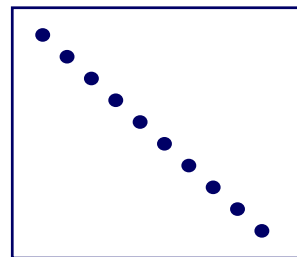
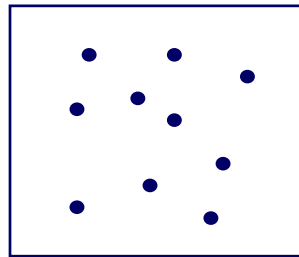
- (Q: how serious is the dim. curse, e.g.:)
- Q: what is the search effort for k-nn?
 - given N points, in E dimensions, in an R-tree, with k-nn queries (‘biased’ model)

[Pagel, Korn + ICDE 2000]



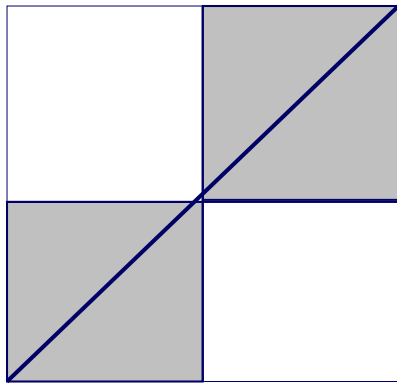
(Overview of proofs)

- assume that your points are uniformly distributed in a d -dimensional manifold (= hyper-plane)
- derive the formulas
- substitute d for the fractal dimension



Reminder: Hausdorff Dimension (D_0)

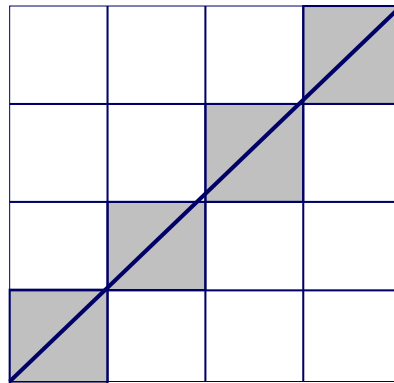
- r = side length (each dimension)
- $B(r) = \#$ boxes containing points $\propto r^{D_0}$



$$r = 1/2 \quad B = 2$$

$$\log r = -1$$

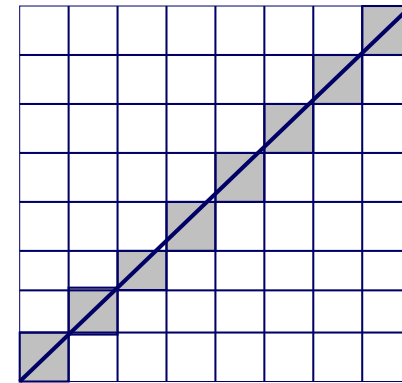
$$\log B = 1$$



$$r = 1/4 \quad B = 4$$

$$\log r = -2$$

$$\log B = 2$$



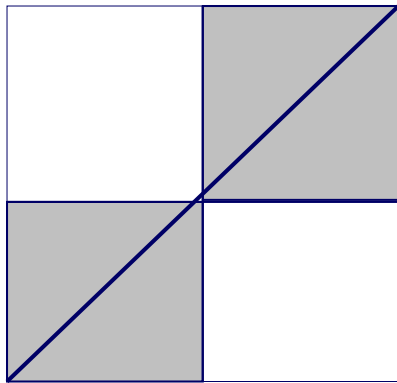
$$r = 1/8 \quad B = 8$$

$$\log r = -3$$

$$\log B = 3$$

Reminder: Correlation Dimension (D_2)

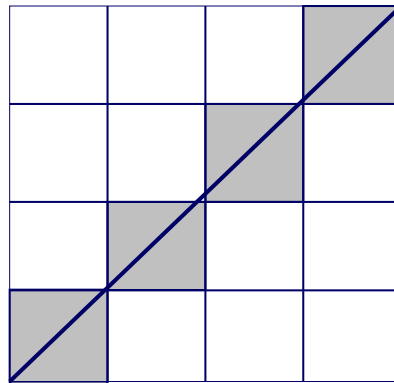
- $S(r) = \sum p_i^2$ (squared % pts in box) $\propto r^{D_2}$
 $\propto \# \text{pairs(within } \leq r)$



$$r = 1/2 \quad S = 1/2$$

$$\log r = -1$$

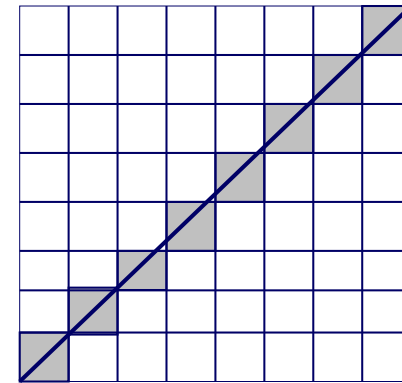
$$\log S = -1$$



$$r = 1/4 \quad S = 1/4$$

$$\log r = -2$$

$$\log S = -2$$



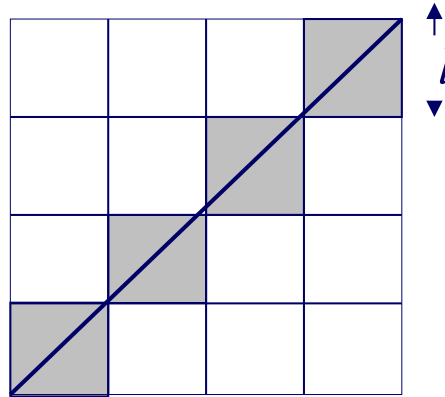
$$r = 1/8 \quad S = 1/8$$

$$\log r = -3$$

$$\log S = -3$$

Observation #1

- How to determine avg MBR side l ?
 - $N = \#pts$, $C = \text{MBR capacity}$

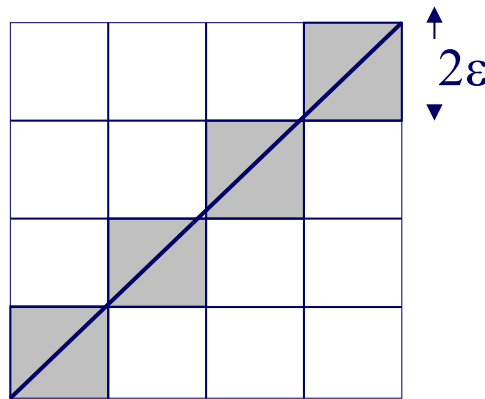


Hausdorff dimension: $B(r) \propto r^{D_0}$

$$B(l) = N/C = l^{-D_0} \Rightarrow \boxed{l = (N/C)^{-1/D_0}}$$

Observation #2

- k -NN query $\rightarrow$ ε -range query
 - For k pts, what radius ε do we expect?

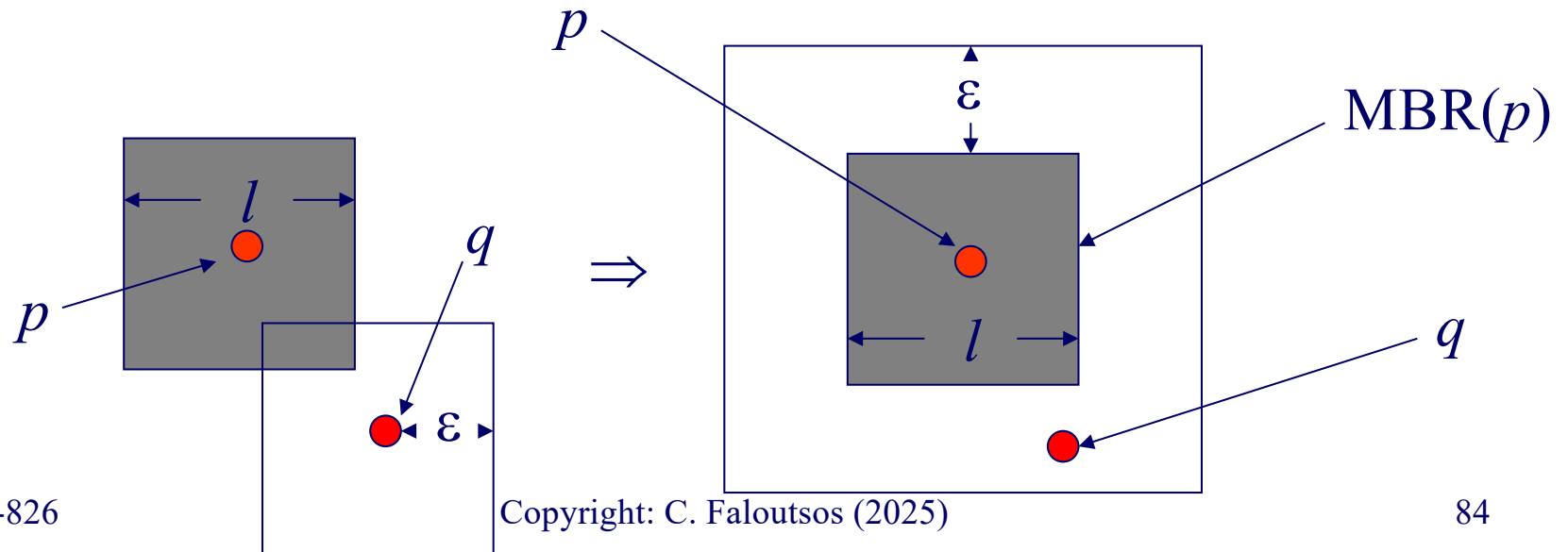


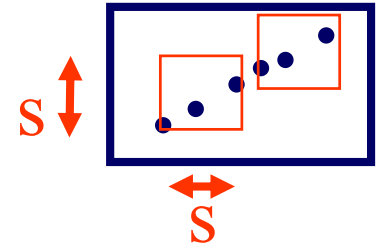
Correlation dimension: $S(r) \propto r^{D2}$

$$S(\varepsilon) = \frac{k}{N-1} = (2\varepsilon)^{D2}$$

Observation #3

- Estimate avg # query-sensitive anchors:
 - How many **expected** q will touch **avg** page?
 - Page touch: q stabs ε -dilated MBR(p)

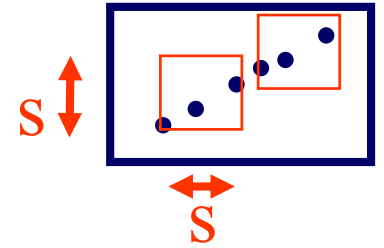




Asymptotic Formula

- k -NN page accesses as $N \rightarrow \infty$
 - C = page capacity
 - D = fractal dimension ($=D_0 \sim D_2$)
 - h = height of tree

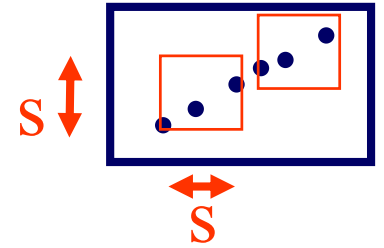
$$P_{all}^{L\infty}(k) \approx \sum_{j=0}^h \left\{ \frac{1}{C^{h-j}} + \left[1 + \left(\frac{k}{C^{h-j}} \right)^{1/D} \right]^D \right\}$$



Asymptotic Formula

$$P_{all}^{L\infty}(k) \approx \sum_{j=0}^h \left\{ \frac{1}{C^{h-j}} + \left[1 + \left(\frac{k}{C^{h-j}} \right)^{1/D} \right]^D \right\}$$

- Observations?

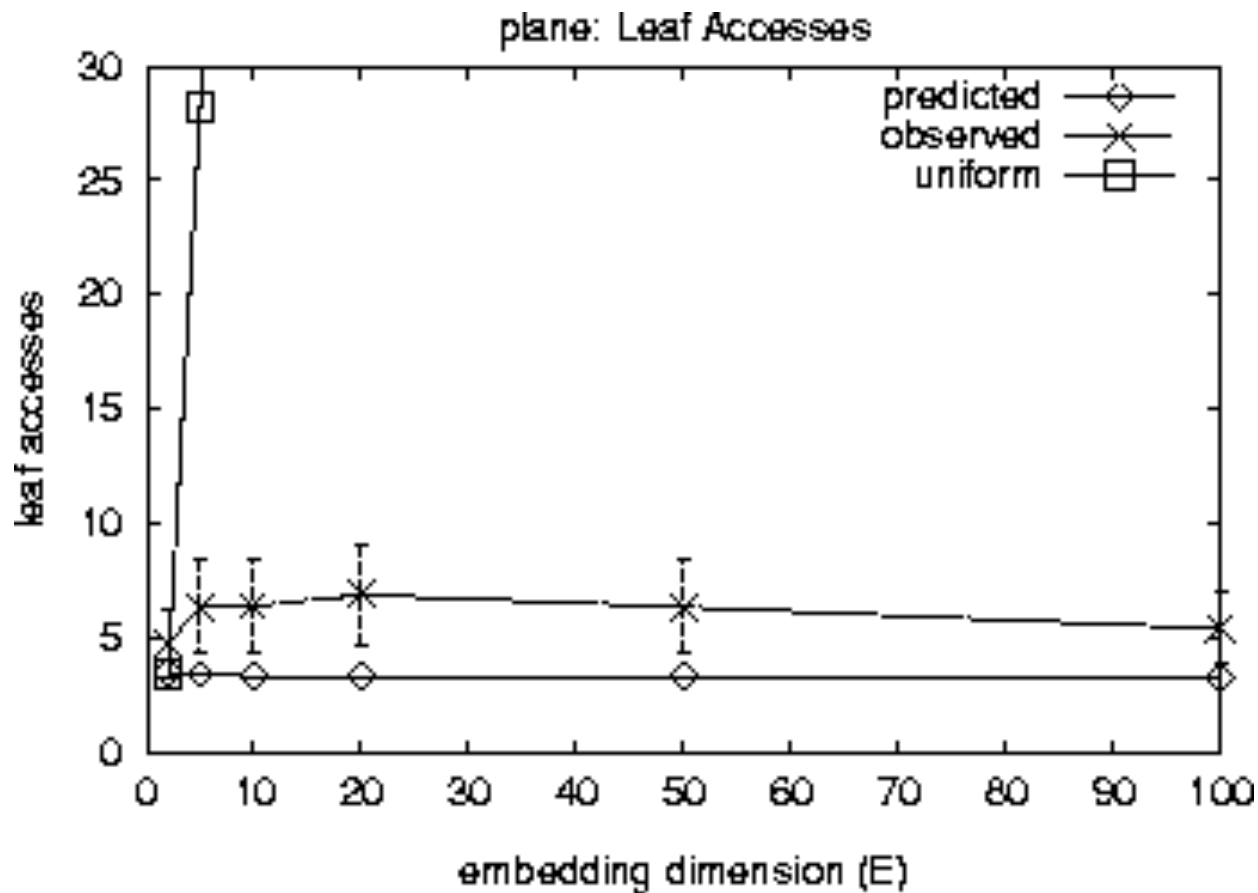


Asymptotic Formula

$$P_{all}^{L^\infty}(k) \approx \sum_{j=0}^h \left\{ \frac{1}{C^{h-j}} + \left[1 + \left(\frac{k}{C^{h-j}} \right)^{1/D} \right]^D \right\}$$

- NO mention of the embedding dimensionality!!
- Still have dim. curse, but on f.d. D

Embedding Dimension



plane
 $k = 50$
 L_∞ dist

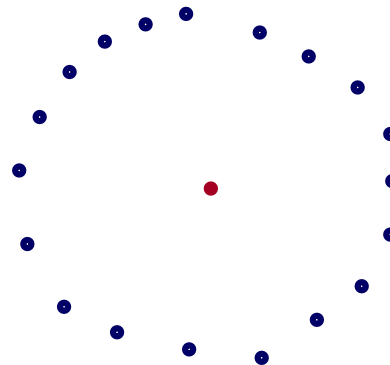
A word of caution:



Nearest neighbors: may be meaningless!

Norio Katayama, Shin'ichi Satoh:

Distinctiveness-Sensitive Nearest Neighbor Search for
Efficient Similarity Retrieval of Multimedia Information.
ICDE 2001: 493-502



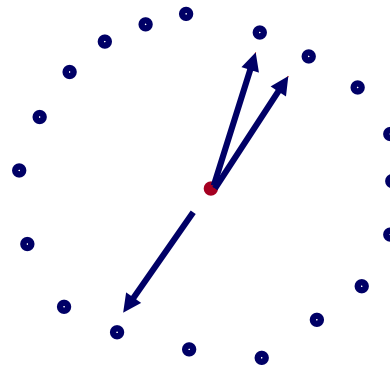
A word of caution:



Nearest neighbors: may be meaningless!

Norio Katayama, Shin'ichi Satoh:

Distinctiveness-Sensitive Nearest Neighbor Search for
Efficient Similarity Retrieval of Multimedia Information.
ICDE 2001: 493-502



Conclusions

- Dimensionality ‘curse’ :
 - for high-d, indices slow down to $\sim O(N)$
- If the **intrinsic** dim. is low, there is hope
- otherwise, do seq. scan, or sacrifice accuracy (approximate nn)

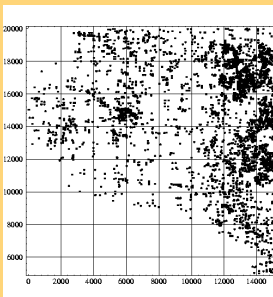
Conclusions – cont' d

- Worst-case theory is **over-pessimistic**
- High dimensional data can exhibit good performance if **correlated, non-uniform**
- Many real data sets are **self-similar**
- Determinant is **intrinsic** dimensionality
 - multiple fractal dimensions (D_0 and D_2)
 - indication of how far one can go



Solutions:

- Q: Do all S.A.M. suffer in high dimensions?
- A: Only in high *fractal* dimensions
- Q: what to do?
- A: dim-reduction; approximate knn; etc



$$P_{all}^{L\infty}(k) \approx \sum_{j=0}^h \left\{ \frac{1}{C^{h-j}} + \left[1 + \left(\frac{k}{C^{h-j}} \right)^{1/D} \right]^D \right\}$$

References

- Sunil Arya, David M. Mount: *Approximate Nearest Neighbor Queries in Fixed Dimensions*. SODA 1993: 271-280

ANN library:

<http://www.cs.umd.edu/~mount/ANN/>

References

- Berchtold, S., D. A. Keim, et al. (1996). The X-tree : An Index Structure for High-Dimensional Data. VLDB, Mumbai (Bombay), India.

References cnt' d

- ➔ • Pagel, B.-U., F. Korn, et al. (2000). *Deflating the Dimensionality Curse Using Multiple Fractal Dimensions*. ICDE, San Diego, CA.
- Papadimitriou, C. H., P. Raghavan, et al. (1998). *Latent Semantic Indexing: A Probabilistic Analysis*. PODS, Seattle, WA.

References cnt' d

- Traina, C., A. J. M. Traina, et al. (2000). *Distance Exponent: A New Concept for Selectivity Estimation in Metric Trees*. ICDE, San Diego, CA.
- Weber, R., H.-J. Schek, et al. (1998). A Quantitative Analysis and Performance Study for Similarity-Search Methods in high-dimensional spaces. VLDB, New York, NY.

References cnt' d

- Yao, A. C. and F. F. Yao (May 6-8, 1985). A General Approach to d-Dimensional Geometric Queries. Proc. of the 17th Annual ACM Symposium on Theory of Computing (STOC), Providence, RI.