# Clustering and Reconstructing Large Data Sets

STONY BROOK
STATE UNIVERSITY OF NEW YORK

Piyush Kumar

http://www.cs.sunysb.edu/~piyush

# Talk Outline

- Introduction
- Core Sets and Minimum Enclosing Balls
- Minimum Volume Ellipsoids
- $k$-center clustering
- Real world applications
  - Hand recognition
  - Curve and Surface Reconstruction
  - Cache Oblivious Algorithms
- Open Problems and Future Research

# Clustering

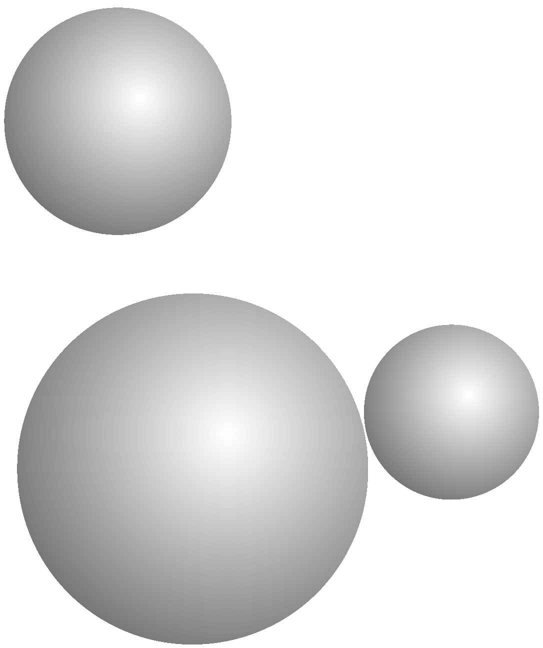
- Partitions data into meaningful groups
- *Cluster*: A collection of objects that are 'similar' to one another. (aka Unsupervised Learning)
- Used
  - to get insight into data distribution.
  - as preprocessing to discover classes.
- Applications :
  - Operations Research (Facility Location)
  - Signal Processing/Compression (Vector Quantization)
  - Curve and Surface Reconstruction
  - Marketing/Web
  - Bio-Metrics/Bio-Informatics
  - As a data mining technique for almost everything.

# Clustering

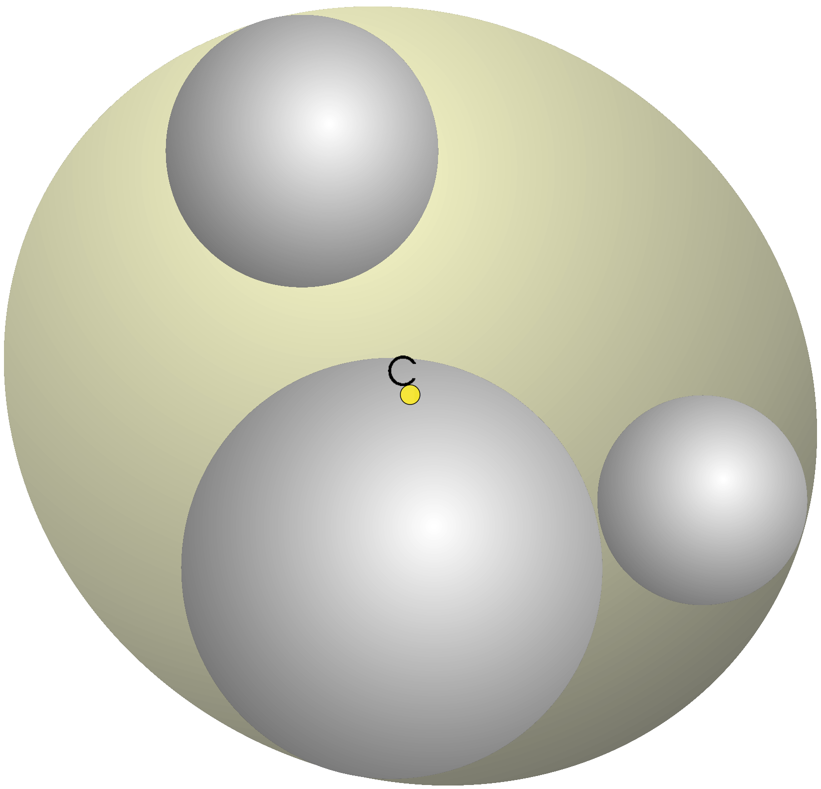
In order to cluster, we need

- Points ( Balls / Ellipsoids? )
- A distance measure ( Euclidean? )
- A method to evaluate the clustering
  - k-centers
  - MVEs
  - kernel 1-center

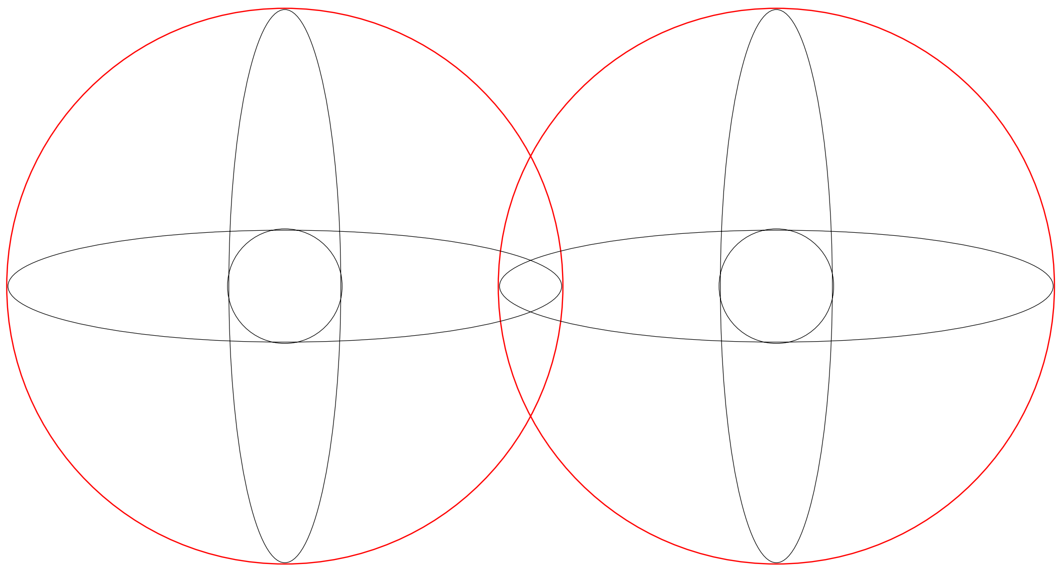
# Introduction: 1-center



# Introduction: 1-center



# Introduction: 2-center



# Motivation

- $k$-center clustering Applications[AS91, E91, KS93, E96, S96, AP98, BHI02, ...]
- Support Vector Clustering [CVBM02 ,BJKS03]
- Gap tolerant classifiers [B98]
- Tuning Support Vector Machines [CVBM02]
- Fast farthest neighbor query approximation [GIV01]
- Testing of radius clustering for $k = 1$ [ADPR00]
- Approximate 1-cylinder problem [BHI02]
- Sphere trees [H96]
- Efficient Transportation [D94]
- Hierarchical clustering for rendering and culling [Qsplat]
- Other applications [EH72]

# Introduction: MVEE

**The MVEE Problem:** Given a set $S = \{p_1, \ldots, p_n\}$ of $n$ points in $\mathbb{R}^d$, compute an ellipsoid of minimum volume enclosing $S$ (a.k.a., the Löwner ellipsoid of $S$).

**Remarks:**

- The Löwner ellipsoid always exists and is unique.
- The center of the Löwner ellipsoid lies in the interior of conv$(S)$, the convex hull of $S$. (Danzer, Laugwitz, and Lenz, 1957)

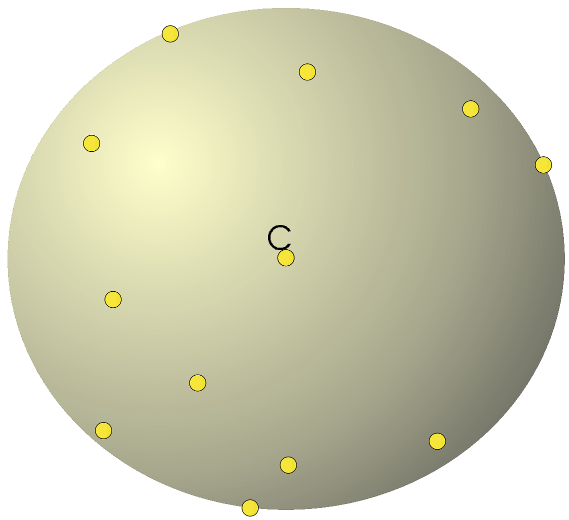
# Applications

- Pattern Recognition
- Computer Graphics / Game Programming
- Optimal Design
- Computational Geometry
- Integer Programming (Lenstra's volume algorithm)
- Statistics
- Nondifferentiable Convex Optimization
- Khachiyan's Ellipsoid Algorithm, etc.

# Minimum Enclosing Balls

1-Centers

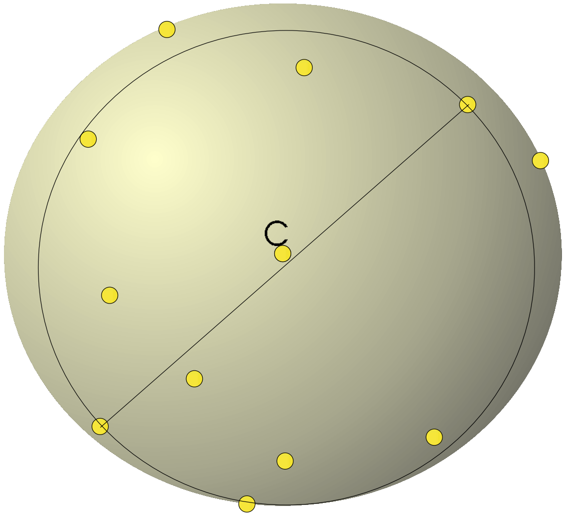
# Definition: Core Sets




$X$ is a core set for $S = \{p_1, p_2, ...p_n\}$ if

- $X \subseteq S$

# Definition: Core Sets

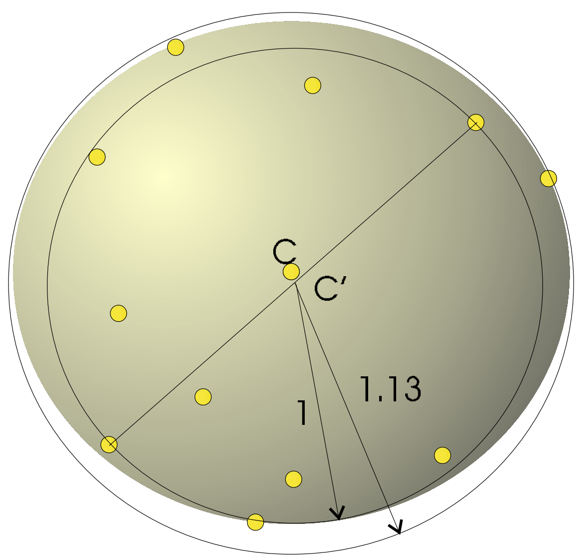



$X$ is a core set for $S = \{p_1, p_2, ...p_n\}$ if

- $X \subseteq S$

where $\epsilon > 0$.

# Definition: Core Sets




$X$ is a core set for $S = \{p_1, p_2, ...p_n\}$ if

- $X \subseteq S$
- $S \subseteq (1+\epsilon)$MEB$(X)$

where $\epsilon > 0$.

# Related Work: 1-center

- Exact MEB solution: $\mathcal{O}(c^{f(d)}n)$ time
  LP-type problem [MSW92], Gärtner; CGAL[a]
- $\mathcal{O}(d^3 n \log \frac{1}{\epsilon})$ $(1+\epsilon)$-approx [GLS88]
- Fast implementations in high dimensions:
  - Simplex-based [Gärtner and Schönherr 00]
  - SOCP-based [ZST02]

[a] `http://www.cgal.org`

# Related Work & Our Results

- Core Set Size :
  - $\mathcal{O}(\frac{1}{\epsilon^2})$ [BHI02]
  - Our Result: $\mathcal{O}(\frac{1}{\epsilon})$
    also [BC03], smaller constants
- Quadratic Programming for MEBs :
  - $\mathcal{O}(d^3 n \log \frac{1}{\epsilon})$ solution, [GLS88]
  - Our Result: $\mathcal{O}(\sqrt{n} d^2 (n + d) \log(1/\epsilon))$

# Related Work & Our Results

- Computing Approx MEB (worst-case):
  - $\mathcal{O}\left(\frac{dn}{\epsilon} + 1.6^{\frac{1}{\epsilon}}\right)$ [Emo Welzl (Pers. Comm.)]
  - $\mathcal{O}\left(\frac{dn}{\epsilon^2} + \frac{1}{\epsilon^{10}} \log \frac{1}{\epsilon}\right)$ [BHI02]
  - $\mathcal{O}\left(\frac{nd}{\epsilon} + \frac{1}{\epsilon^5}\right)$ [BC03]
  - $\mathcal{O}\left(\frac{nd}{\epsilon} + \frac{1}{\epsilon^{4.5}} \log \frac{1}{\epsilon}\right)$ [Our Result]
  - $\mathcal{O}\left(\frac{nd}{\epsilon} + \frac{1}{\epsilon^4} \log^2 \frac{1}{\epsilon}\right)$ [Our Result, H03]
- $k$-center clustering, $2^{\mathcal{O}\left(\frac{k \log k}{\epsilon}\right)} dn$ [Our Result, BC03]

# Results

- In Practice :
  - Core Set Sizes:
    - Dependent on dimension.
    - Very Weak dependence on $\epsilon$.
    - $\leq \min\{d+1, \frac{1}{\epsilon}\}$.

# Results

- In Practice :
  - Core Set Sizes:
    - Dependent on dimension.
    - Very Weak dependence on $\epsilon$.
    - $\leq \min\{d+1, \frac{1}{\epsilon}\}$.
  - Run Times:
    - Much smaller than Worst Case.
    - Weakly dependent on epsilon.

# SOCP Formulation

*Second Order Cone Program* is of the form

maximize $c^T x$

subject to $||A_i x + b_i||_2 \leq c_i^T x + d_i, \quad i = 1..n$

$Fx = g$

- $x, c, c_i, d_i \in \mathbb{R}^d, b_i \in \mathbb{R}^{(d_i-1)}, A_i \in \mathbb{R}^{(d_i-1)\times d}$
- LP is a special case
- new Interior Point (IP) methods can solve (almost) as fast as LPs

# SOCP Formulation

MEB as SOCP

$$\min_{c,r} \; r, \quad \text{s.t.} \quad \|c - c_i\| + r_i \leq r$$

$$i = 1, \ldots, n$$

- Number of iterations = $\mathcal{O}(\sqrt{n}\log(1/\epsilon))$ , In Practice $\leq 20$, very weak dependence on $n$.
- IP solves it in $\mathcal{O}(\sqrt{n}d^2(n+d)\log(1/\epsilon))$

# In Practice

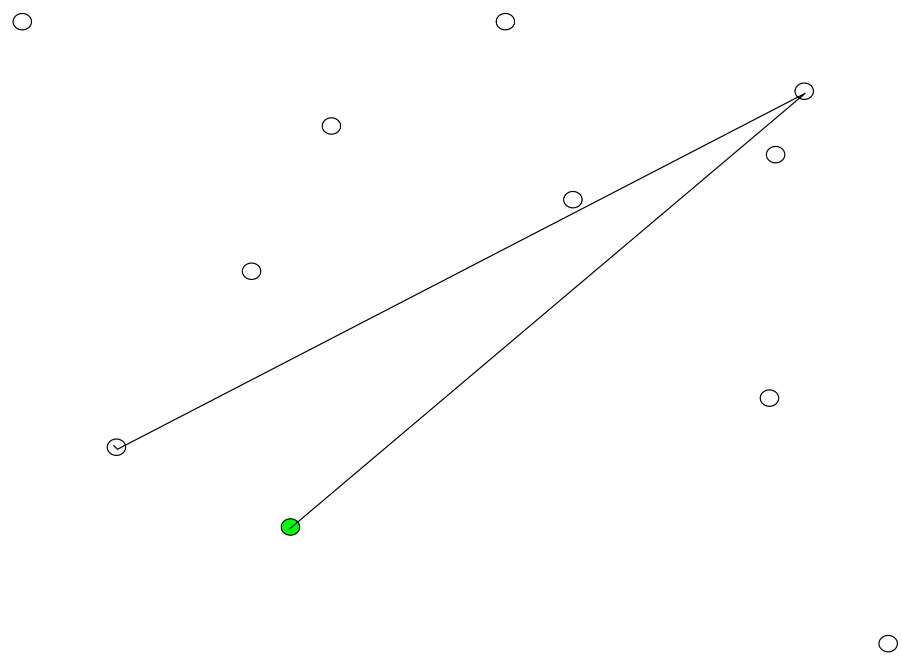
- IP methods are fairly effective in practice.
- The iteration complexity seems to behave independently of $n$ (approximately 10 – 15 iterations).
- The bottleneck is the computational cost per iteration ($O(nd^2 + d^3)$).
- The computational complexity reveals that the direct application of interior-point algorithms is not computationally feasible for large-scale instances.

# Why Core Sets?

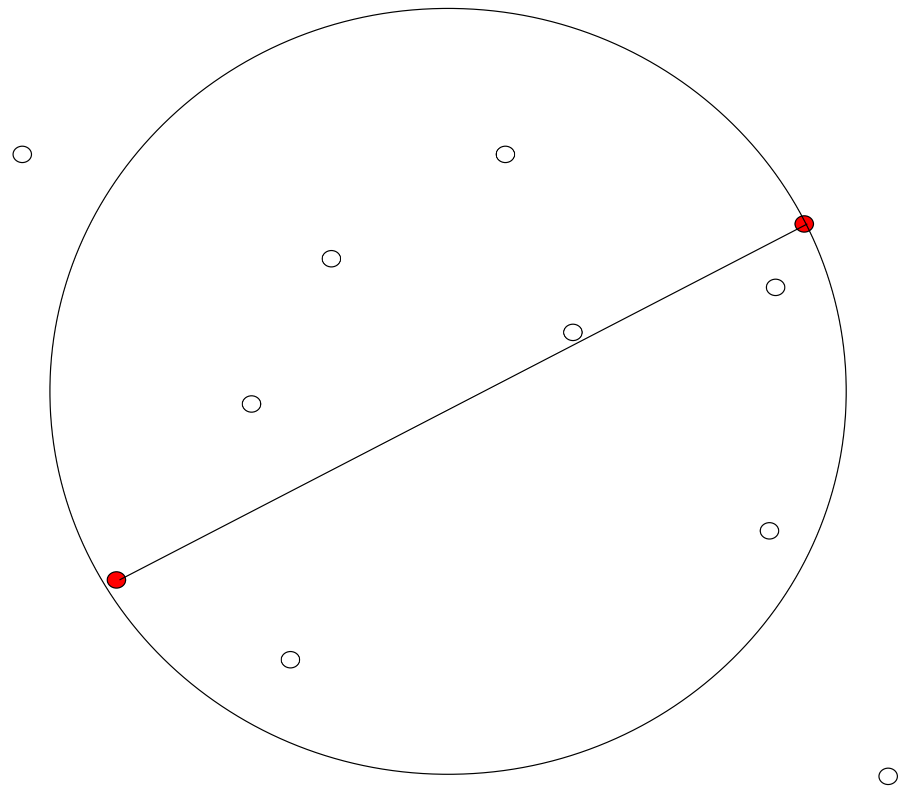
- The overall complexity of IP method is $O(\sqrt{n}d^2(n+d)\log(1/\epsilon))$ flops.
- To make a practical algorithm, we need a way to reduce either $d$ or $n$.
- We reduce $n$ to $\mathcal{O}(\frac{1}{\epsilon})$ using core sets.
- $n = \mathcal{O}(\frac{1}{\epsilon})$ implies $d = \mathcal{O}(\frac{1}{\epsilon})$ .

Next : The Algorithm.

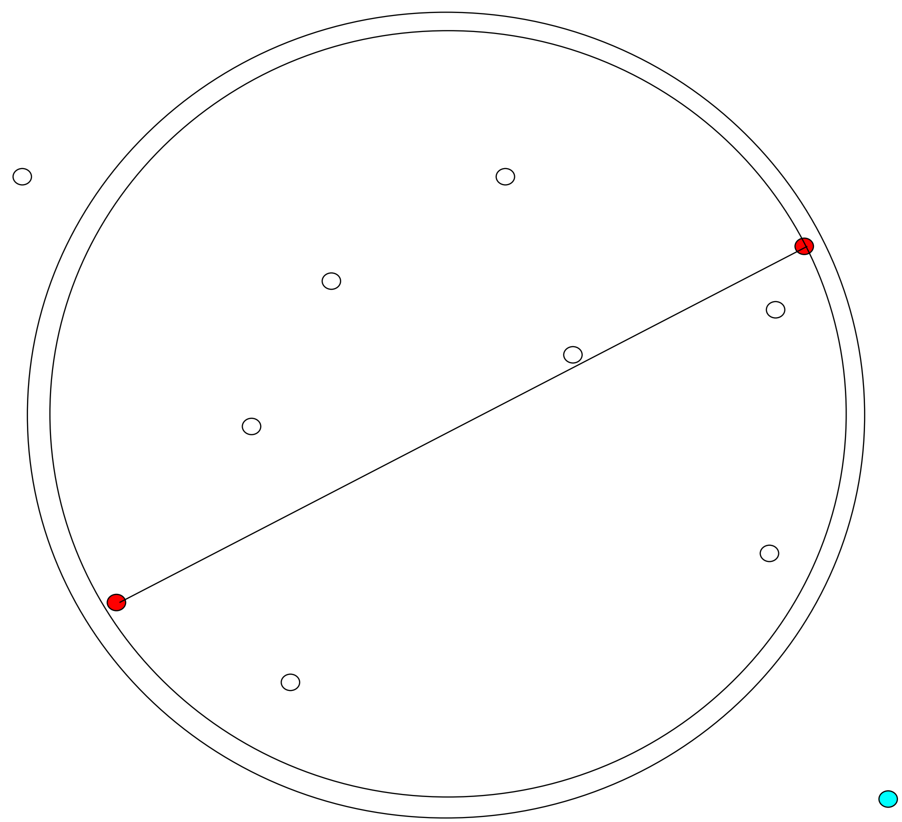
# The Algorithm



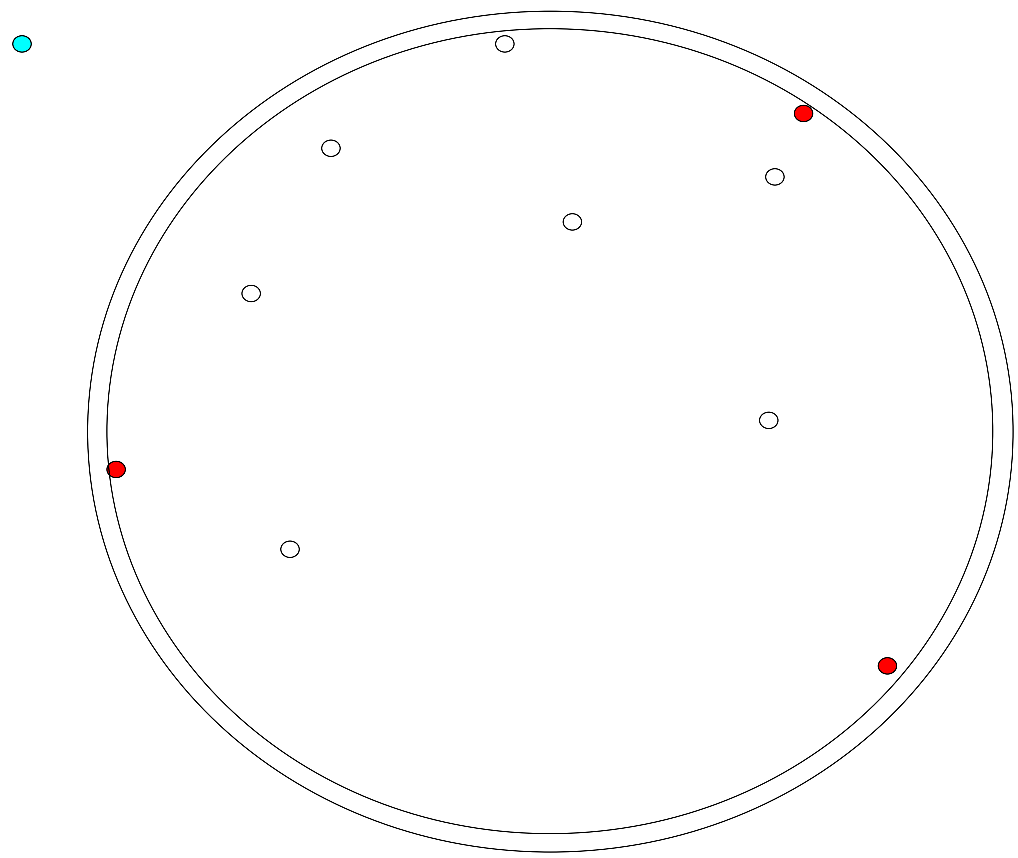
# The Algorithm



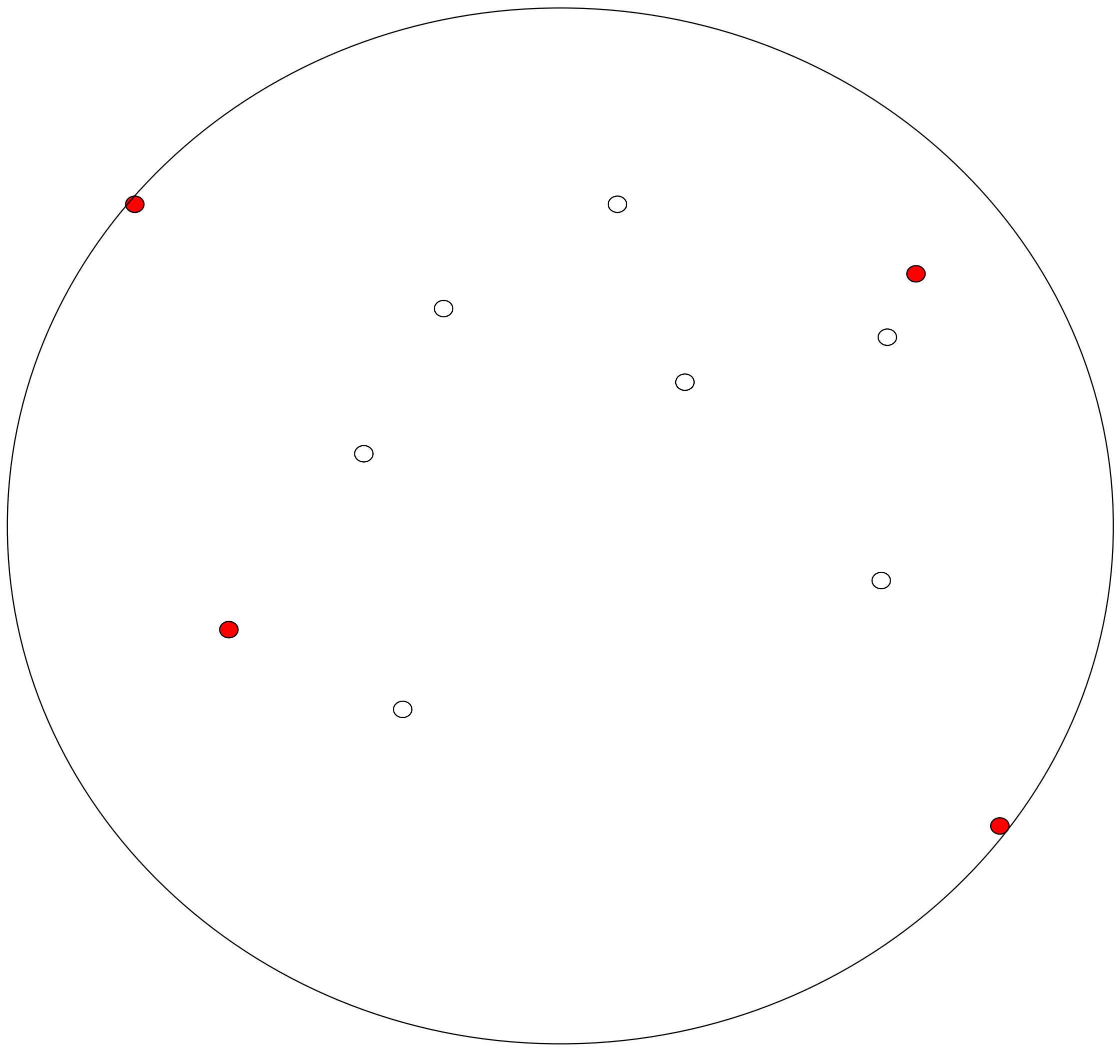
# The Algorithm



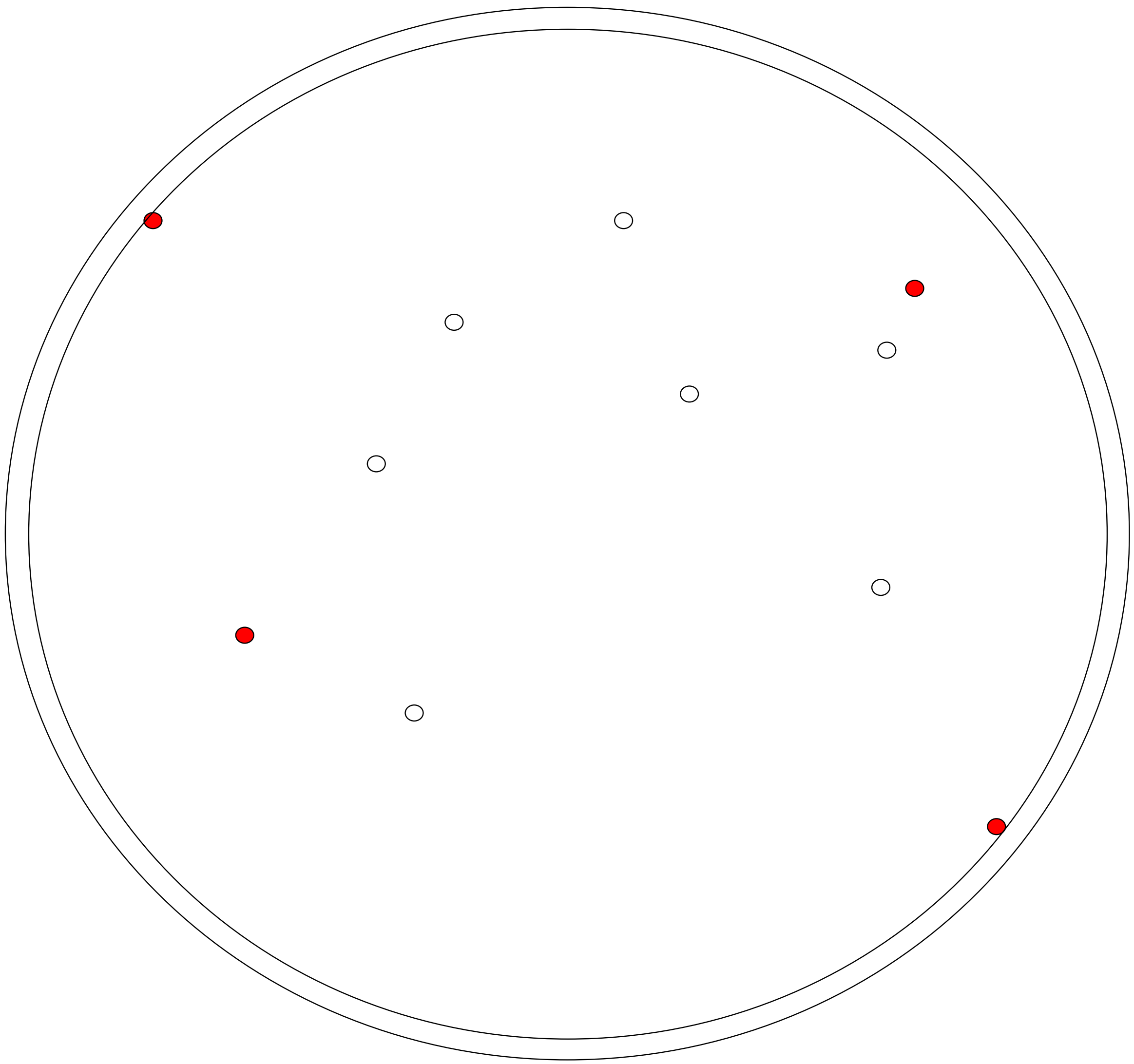
# The Algorithm



# The Algorithm



# The Algorithm



# The Core Set Algorithm: $\mathcal{O}(\frac{1}{\epsilon})$

**Require:** Input set $S \subset \mathbb{R}^d$ of points/balls, $\epsilon \in (0,1)$

$X \leftarrow \{q, q'\}$, where $q, q' \in S$ ( diameter approximation )
$\delta \leftarrow \epsilon^2/163$
**loop**
    Let $B_{c',r'}$ denote the $(1+\delta)$-approximation to MEB($X$) returned by SOCP.
    **if** $S \subseteq B_{c',(1+\epsilon/2)r'}$ **then**
        Return $B_{c',(1+\epsilon/2)r'}$, $X$
    **else**
        $p \leftarrow \arg\max_{x \in S} \|c' - x\|$
    **end if**
    $X \leftarrow X \cup \{p\}$
**end loop**

# The Core Set Algorithm

- Use SDPT3[a] Package to solve SOCP. [TTT99]
- Implementation bootstraps [BC03]
- I/O Efficient under mild assumptions (CO).
- Works for Balls,Points

[a] http://www.math.nus.edu.sg/~mattohkc/sdpt3.htm

# The Proof Idea

- Compute a $\frac{1}{\sqrt{3}}$-approximation in $\mathcal{O}(nd)$ time.
- If not done, Find an outlier
- Prove that adding the outlier increases the radius by $(1 + O(\epsilon^2))$ factor even if the oracle gives an approximate answer.
- The radius is finite and will get caught in this process.
- Show that $\mathcal{O}(\frac{1}{\epsilon})$ iterations are sufficient.

# Experimental Setup: 1-Center

1. Platform: Pentium Xeon, 2.66Ghz, 2GB RAM
2. $n \leq 10^6$, $d \leq 1500$

# Data Sets: 1-Center

① uniformly distributed within a unit cube;

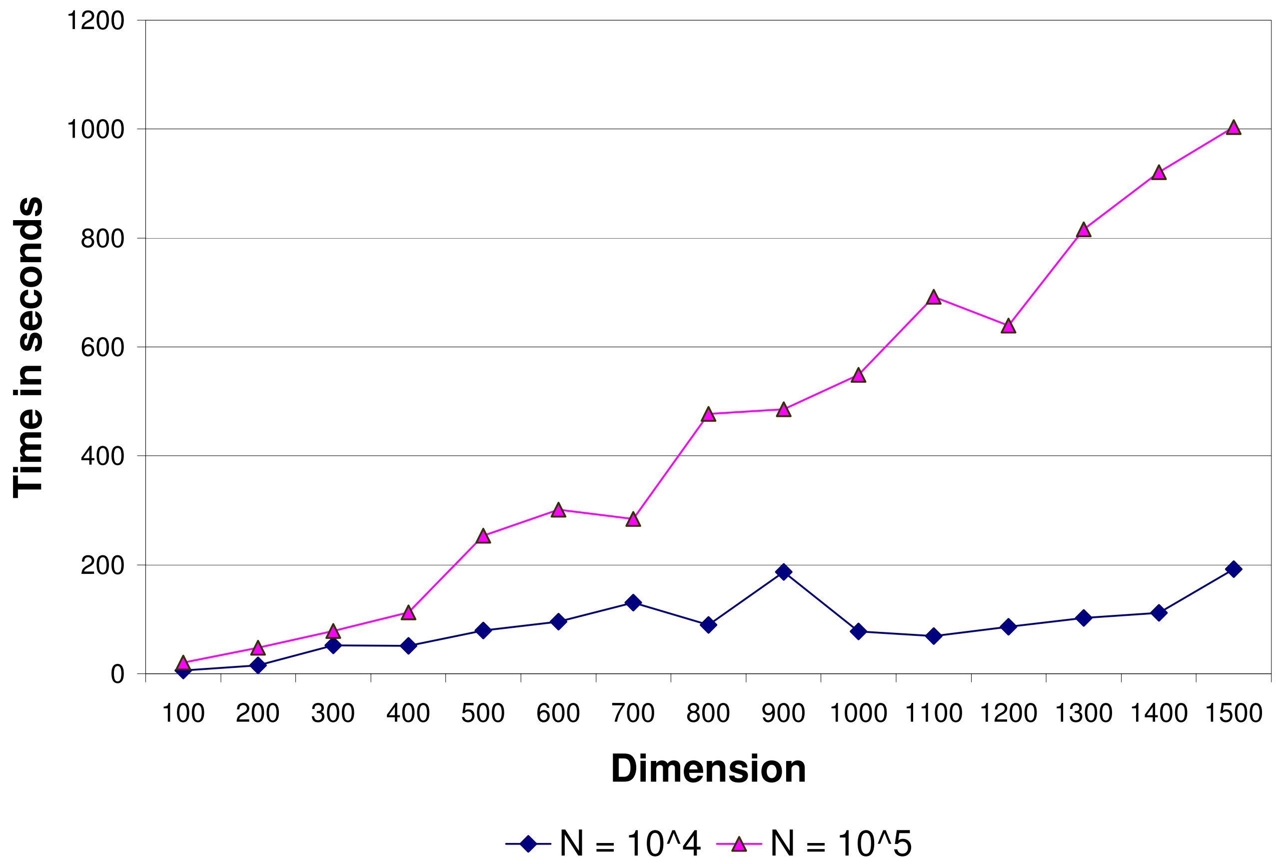
② uniformly distributed on the vertices of a unit cube;

③ normally distributed in space, with each coordinate chosen independently according to a normal distribution with mean 0 and variance 1;

④ point coordinates that are Poisson random variables, with parameter $\lambda = 1$.

⑤ USPS data ($n = 7291, d = 256$)

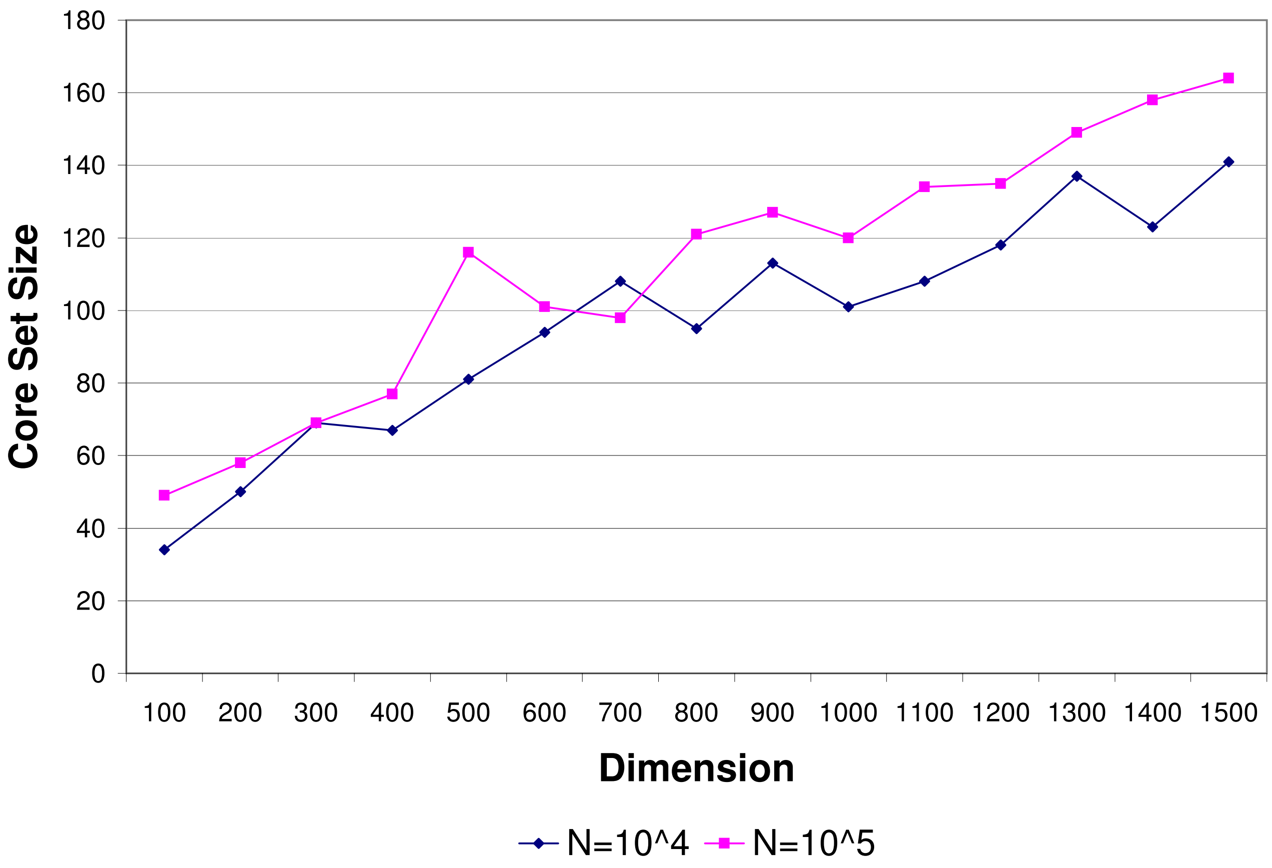
# Implementation and Experiments




Running time of our algorithm

$\epsilon = 10^{-3}, \mu = 0, \sigma = 1.$

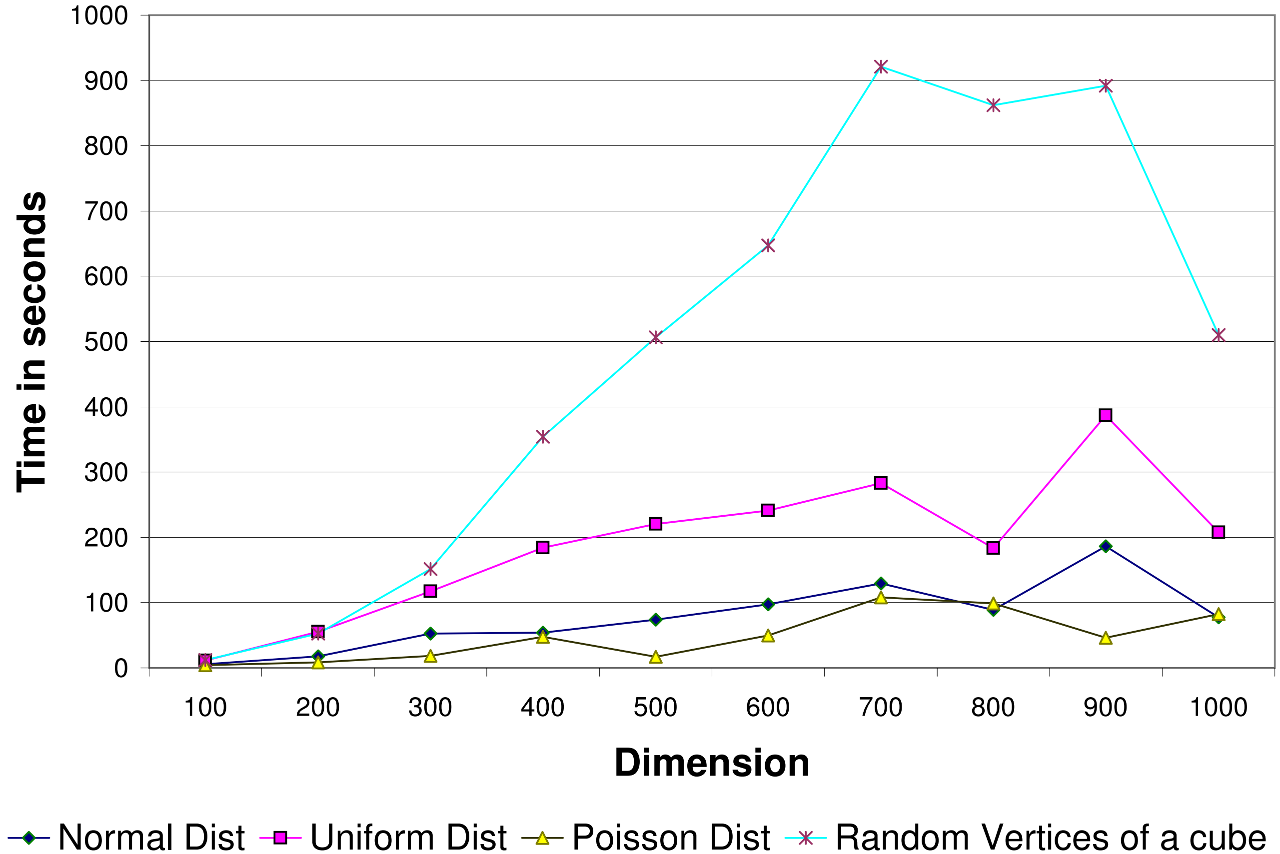
# Implementation and Experiments




Core Set Sizes

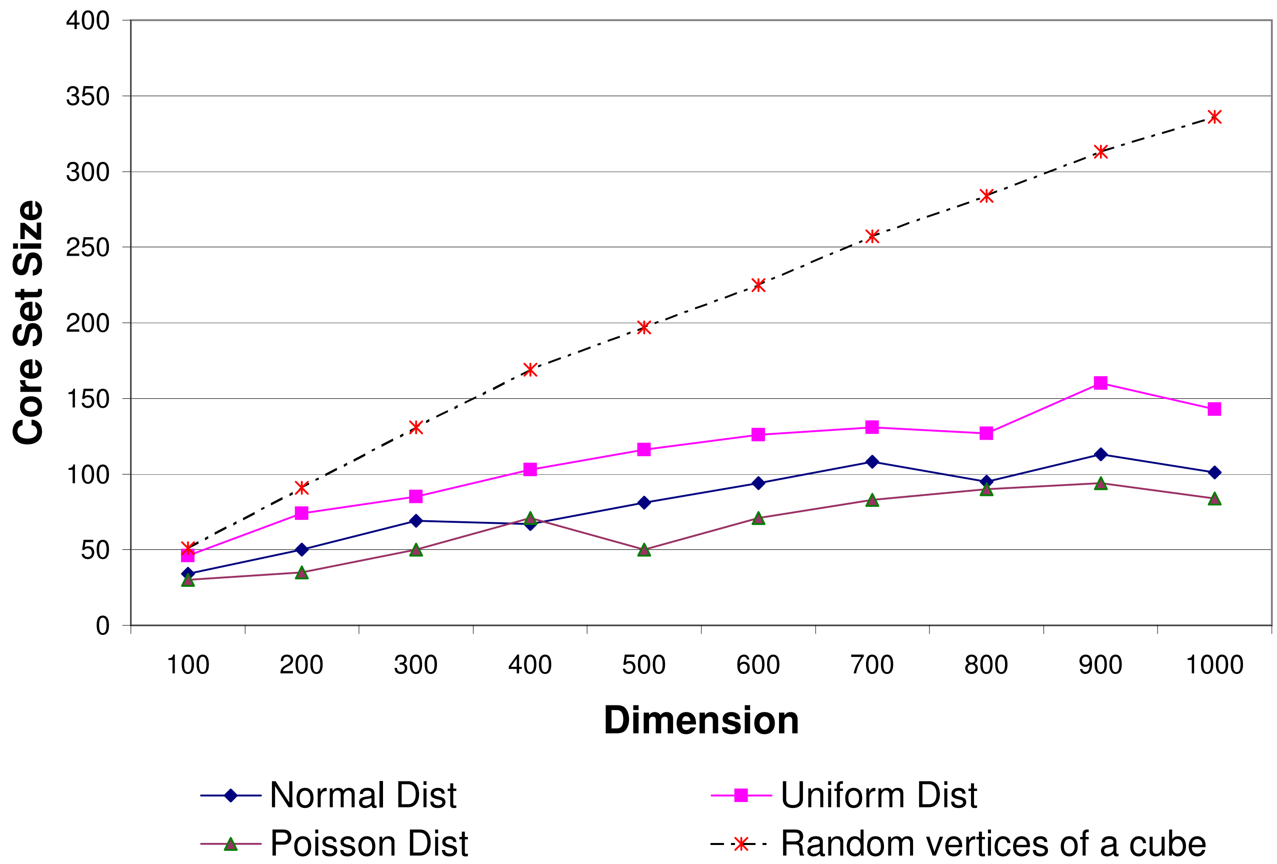
$\epsilon = 10^{-3}, \mu = 0, \sigma = 1$

# Implementation and Experiments




Different Distributions $n = 10^4, \epsilon = 10^{-3}$.

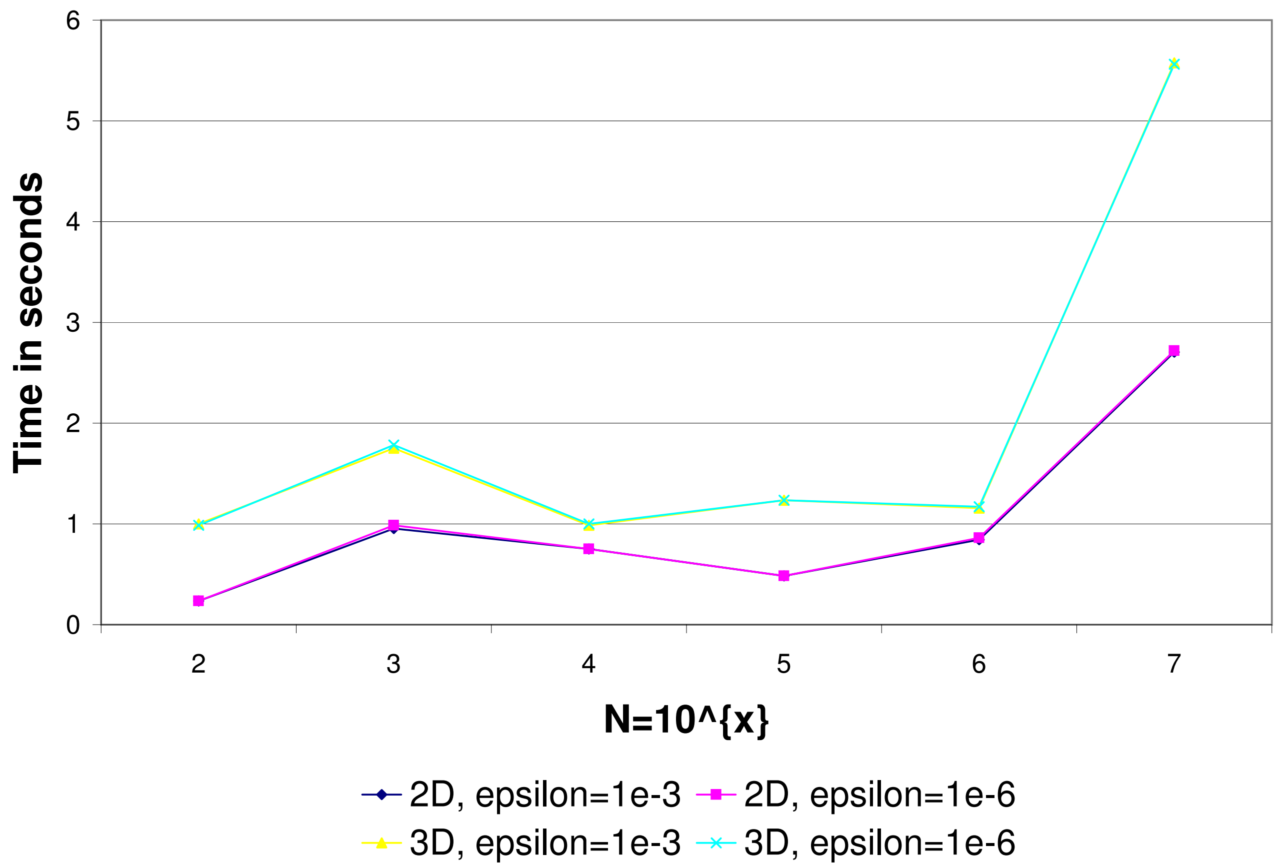
# Implementation and Experiments




Core Set Sizes: Different Distributions

$n = 10^4, \epsilon = 10^{-3}$

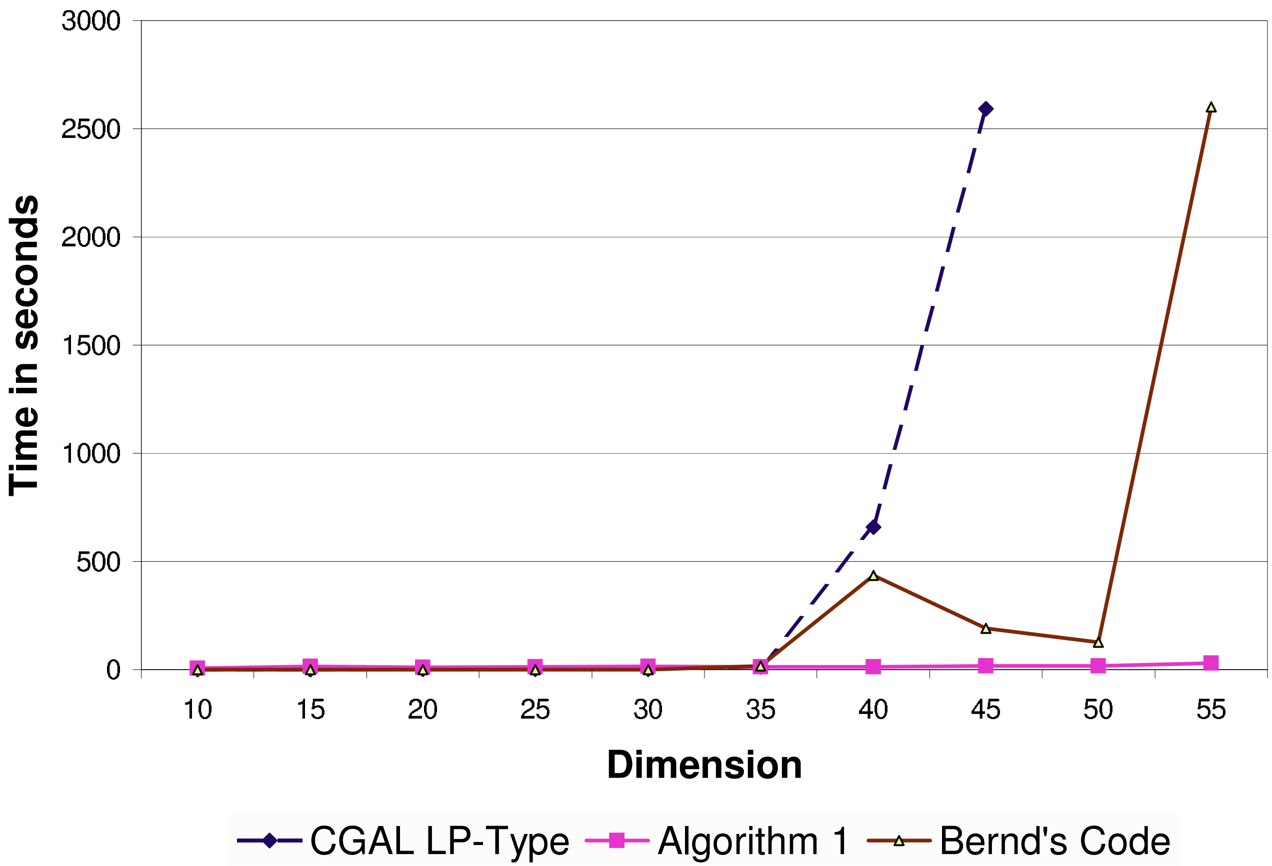
# Implementation and Experiments




Experiments in $\mathbb{R}^2$ and $\mathbb{R}^3$

$\mu = 0, \sigma = 1$

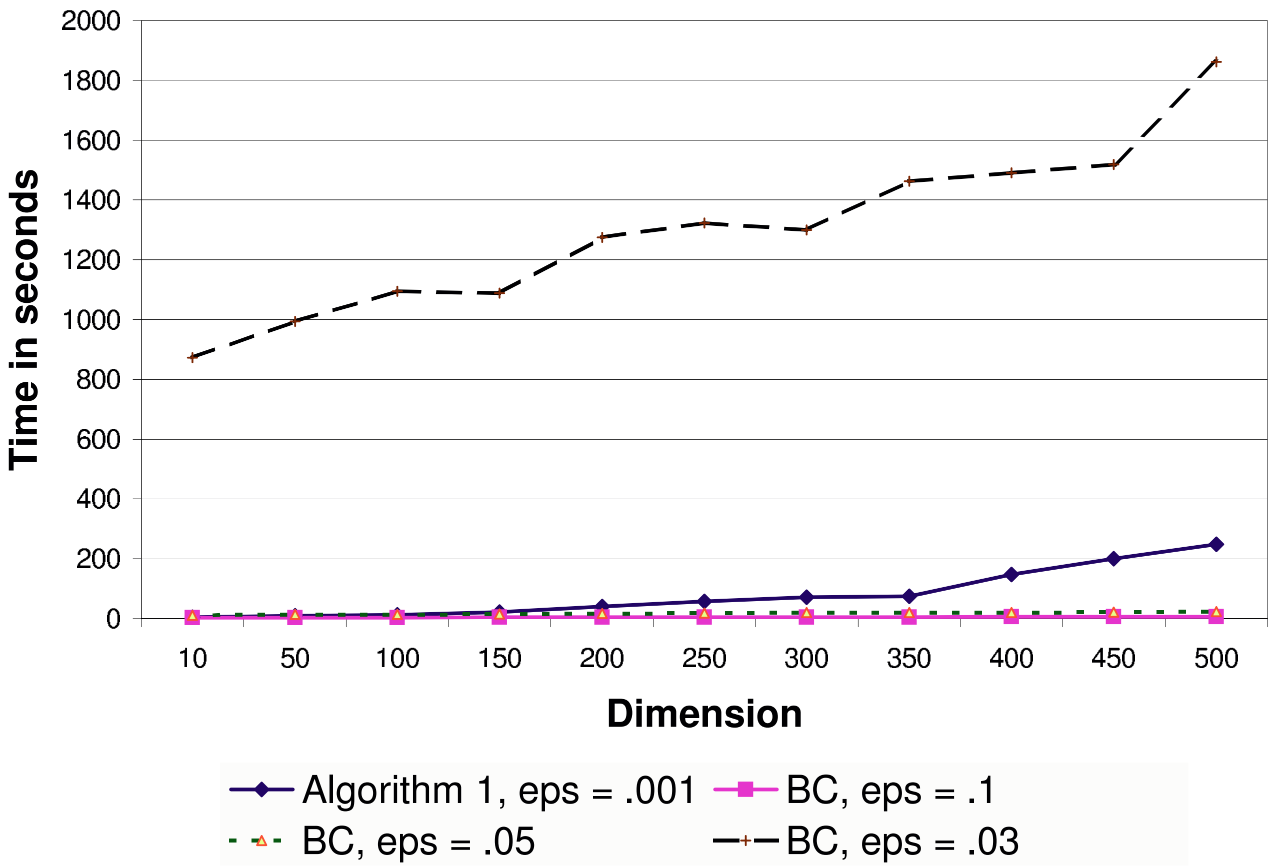
# Shameless Promotion ☺




Timing Comparison

$$n = 1000, \epsilon = 10^{-6}, \mu = 0, \sigma = 1$$

# Shameless Promotion ☺




Algorithm Comparison

$$\mu = 0, \sigma = 1, n = 1000$$

# Minimum volume ellipsoids

MVEE

# Preliminaries

- $S = \{p_1, \ldots, p_n\}$ : input set consisting of $n$ points in $\mathbb{R}^d$
- A (full-dimensional) ellipsoid $E_{Q,c}$ in $\mathbb{R}^d$:

$$E_{Q,c} = \{x \in \mathbb{R}^d : (x - c)^T Q(x - c) \leq 1\}$$

  $Q \in \mathbb{R}^{d \times d}$ is symmetric and positive definite, $c \in \mathbb{R}^d$ is the center of $E_{Q,c}$.
- Volume of an ellipsoid $E_{Q,c}$:

$$\text{vol } E_{Q,c} = B \text{ det } Q^{-\frac{1}{2}}$$

  $B$ is the volume of the unit ball in $\mathbb{R}^d$.
- MVEE(S): Minimum volume enclosing ellipsoid of $S$
- Fritz John inequality:

$$\frac{1}{d}\text{MVEE}(S) \subseteq \text{conv}(S) \subseteq \text{MVEE}(S)$$

# Core Sets

(cf. Bădoiu, Har-Peled, Indyk, 2002)

- An ellipsoid $E_{Q,c} \supseteq S$ is said to be a $(1+\epsilon)$-approximation to the MVEE$(S)$ if

  $$\text{vol } E_{Q,c} \leq \text{ vol } (1+\epsilon) \text{ MVEE}(S) = (1+\epsilon)^d \text{ vol MVEE}(S).$$

- A set $X \subseteq S$ is called a *core set for $S$* if $S \subseteq E_{Q,c} := (1+\epsilon)\text{MVEE}(X)$. (i.e., $E_{Q,c}$ is a $(1+\epsilon)$-approximation to the MVEE$(S)$.)

# A Snapshot of Main Results

- There exists a core set $X \subseteq S$ with

$$|X| = \alpha = \mathcal{O}(d \log d + \frac{d}{\epsilon}) \ .$$

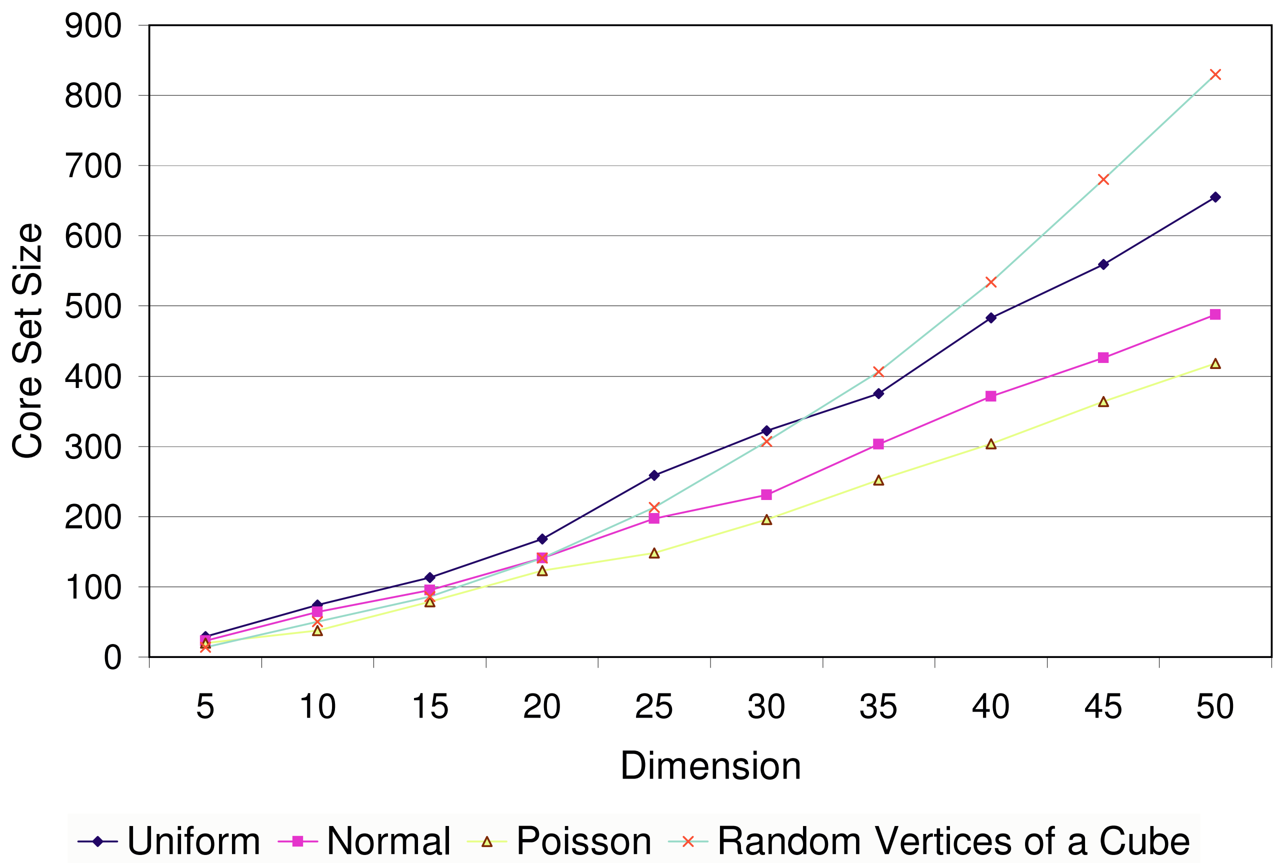
- MVEE$(S)$ can be computed in

$$\mathcal{O}(nd^2\alpha + \alpha^{4.5} \log \frac{\alpha}{\epsilon})$$

arithmetic operations.

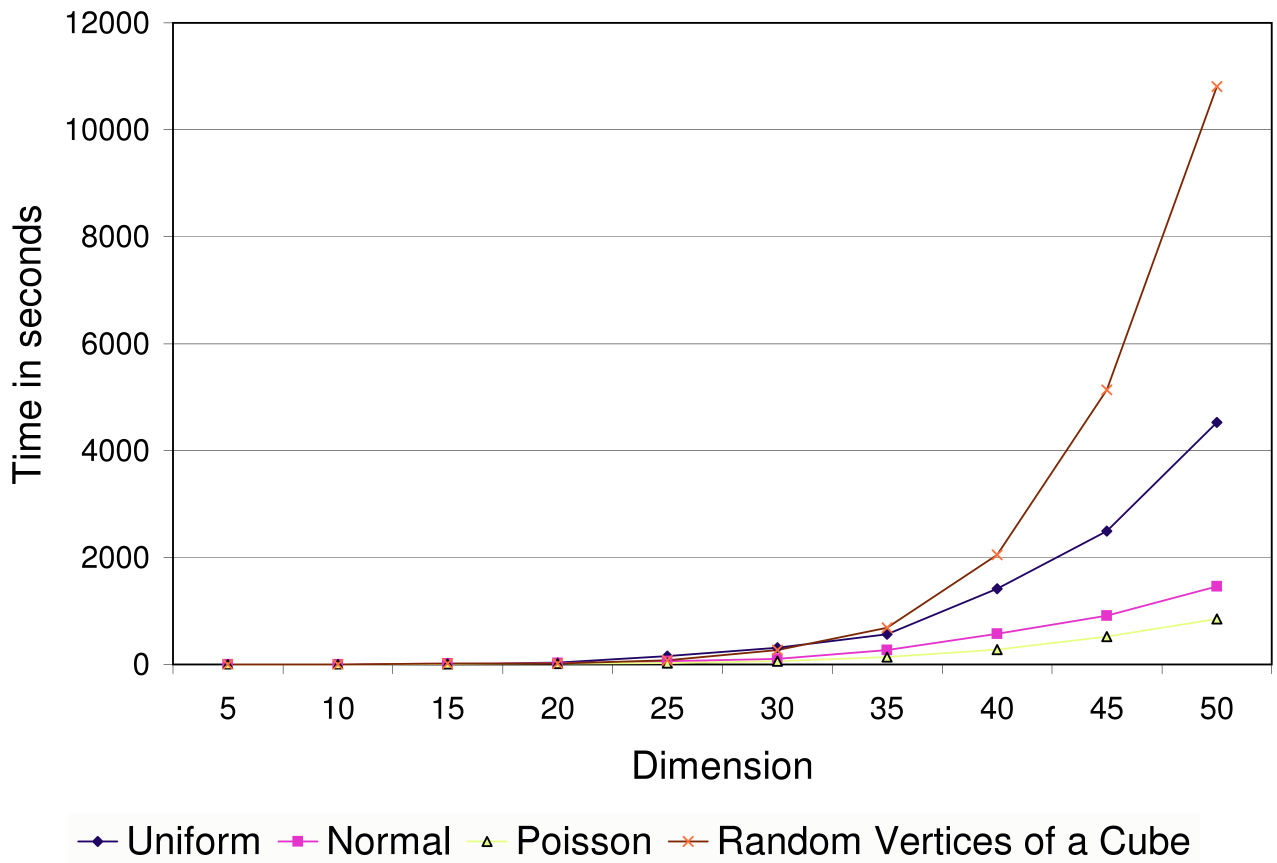
- This is an improvement over the previously known results for large scale instances with $n \gg d$ and reasonably small values of $\epsilon > 0$.

# Implementation and Experiments




$$n = 10^4, \epsilon = 10^{-6}$$

# Implementation and Experiments



# The k-center Algorithm

k-Centers

# k-center clustering

- Given n points in a metric space
- Find k centers
- Minimize the maximum distance from a point to its nearest center.

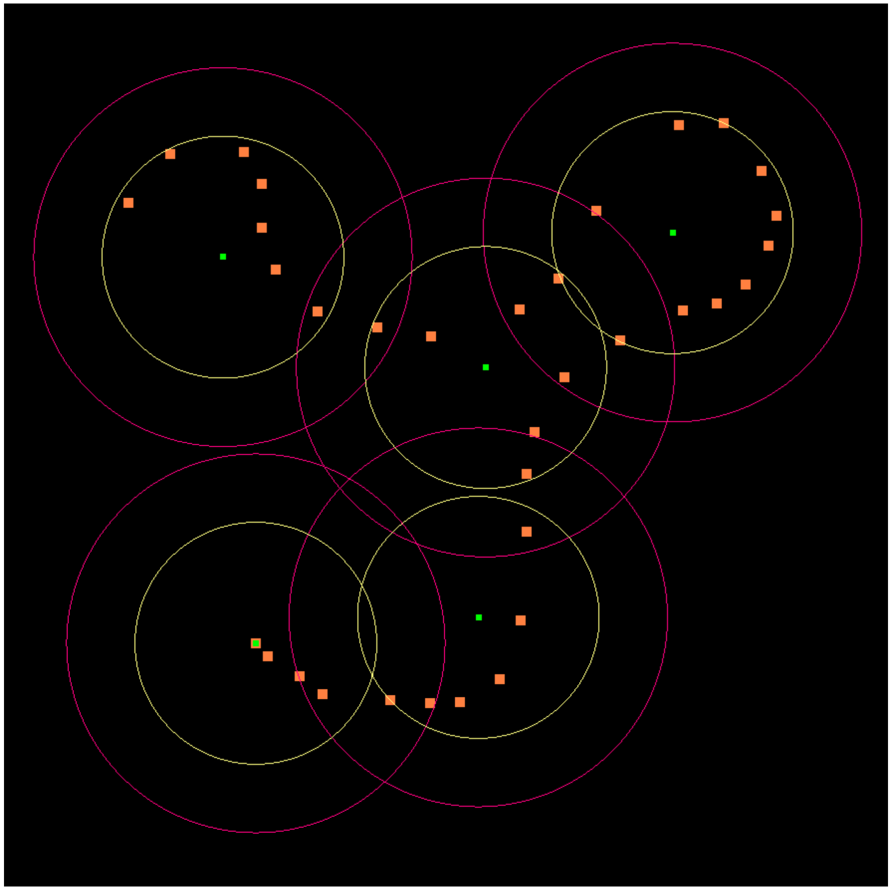
# k-center clustering

- Is NP-Hard
- Approximations?

$$\frac{\text{Cost of } k \text{ centers we find}}{\text{Cost of } k \text{ optimal centers}} \leq c$$

- There is no polynomial time algorithm that obtains a $(2-\epsilon)$-approximation to k-center, unless P$=$NP.

# The k-center Algorithm



# k-center: Snapshot of results

- Core Set size of $O(\frac{1}{\epsilon})$.
- Running time = $O(2^{\frac{k \log k}{\epsilon}} dn)$
- Based upon exhaustive enumeration
- Works for balls and constant complexity polytopes. ($\mathcal{V}$-representation)
- Can be extended to work with ellipses, non convex polygons etc.
- Works in higher dimensions
- Is easy to code! (a.k.a. Simple ☺)

# Analysis of a benign instance

## A Simple Order calculation

- $\epsilon = 0.01$
- $d = 2$
- $n = 15112$

Order $= 2^{\frac{k \log k}{\epsilon}} dn \approx O(2^{500})$

*If all computers on earth start solving this instance,*

*they will take many times the age of the universe to finish!*

# The k-center Algorithm

**Require:** Input:

$S \in \mathbb{R}^d,\ \mathcal{M} = \{M_1, \dots, M_k\},\ \mathcal{B} = \{B_1, \dots, B_k\},\ \epsilon > 0$

$\mathcal{M}_o \leftarrow \mathcal{M}$, $\mathcal{B}_o \leftarrow \mathcal{B}$

**loop**

  $p \leftarrow$ point $q \in S$ furthest from $\mathcal{B}$.

  **if** $p \in (1+\epsilon)\mathcal{B}$ **then**

    Return $\mathcal{M}_o, \mathcal{B}_o$

  **else**

    $\dots$

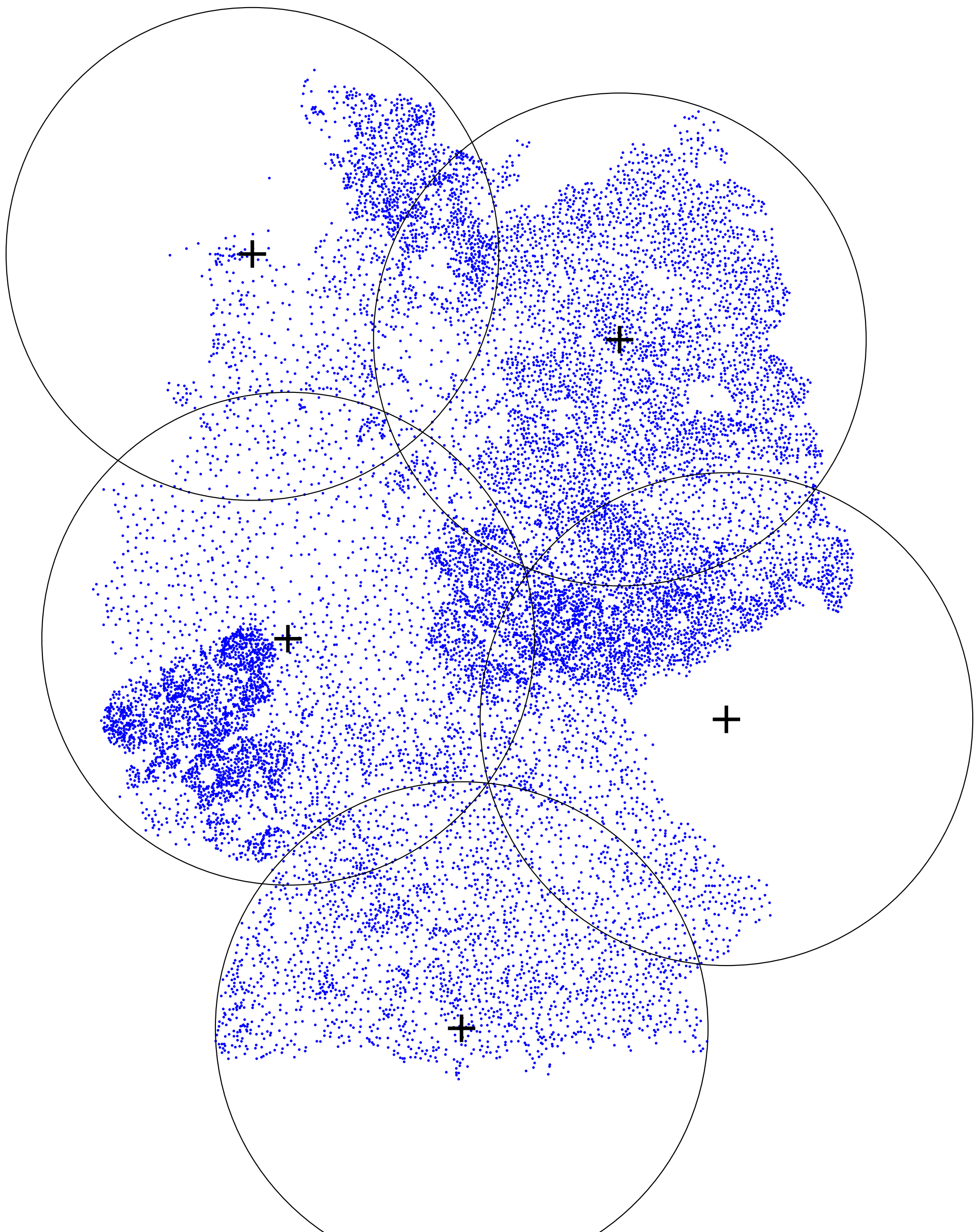
  **end if**

**end loop**

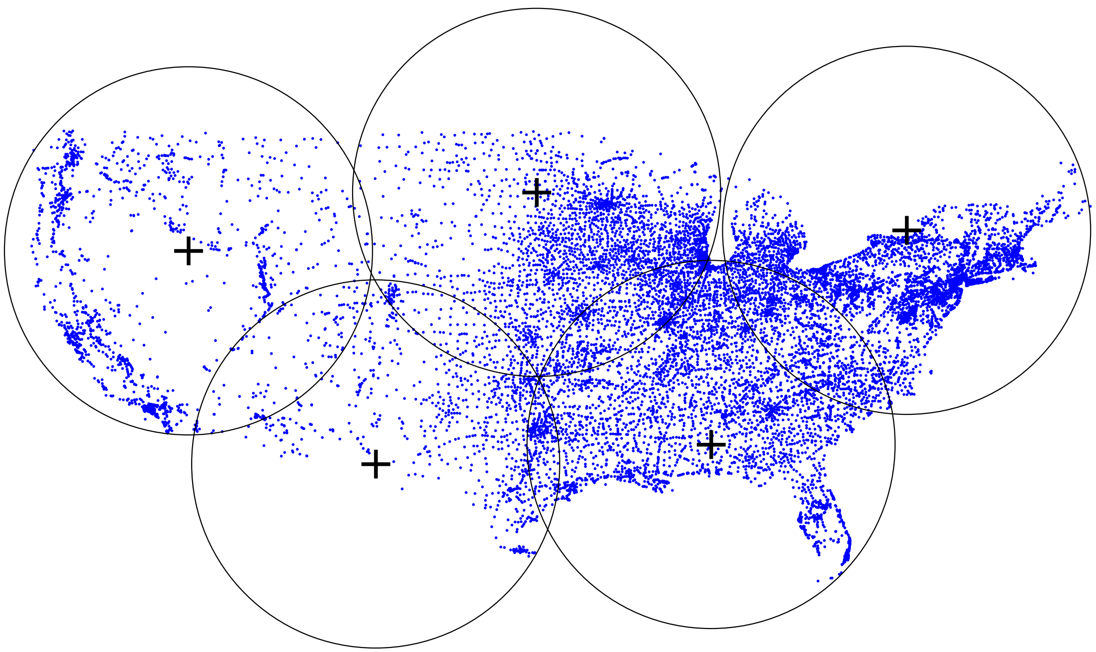
# Experimental Setup: k-Center

1. Platform: Dual 2.66Ghz Xeons, 2GB RAM
2. $k \leq 5$, $d \leq 5$, $\epsilon \geq 0.01$, $n \leq 10^5$.

# The Solution of the benign instance

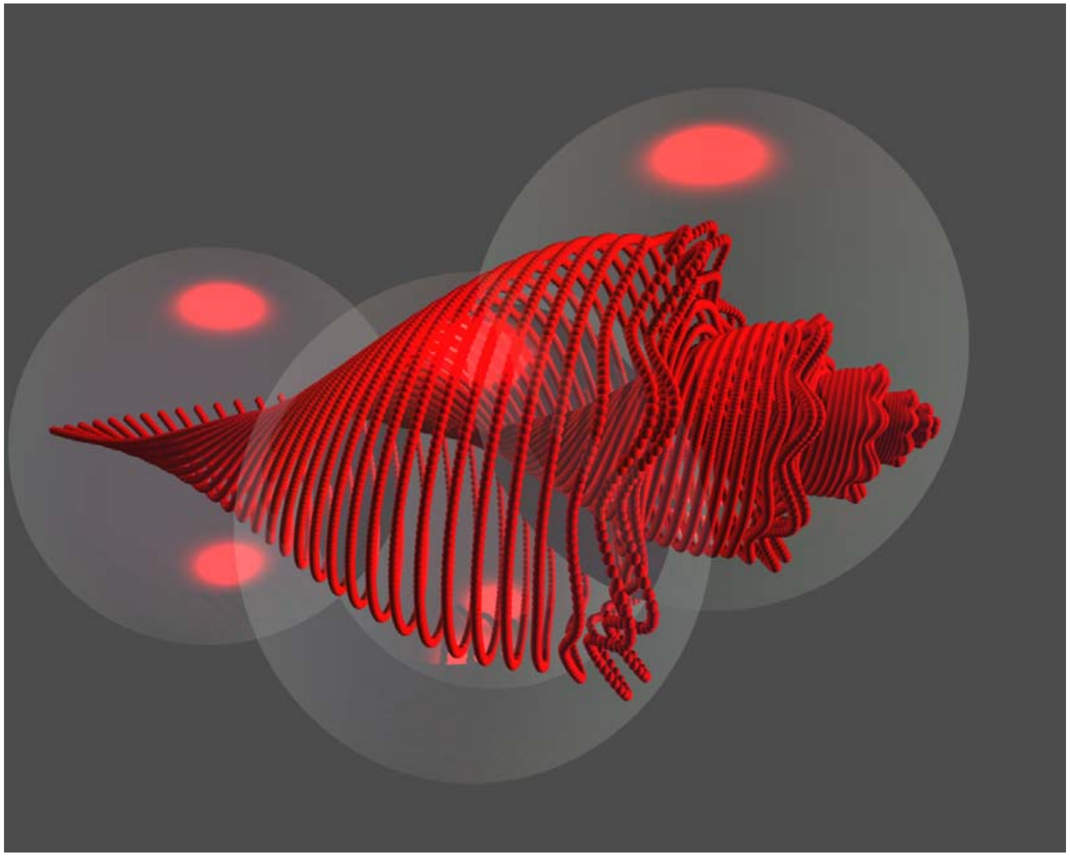


# 5-centers of US cities

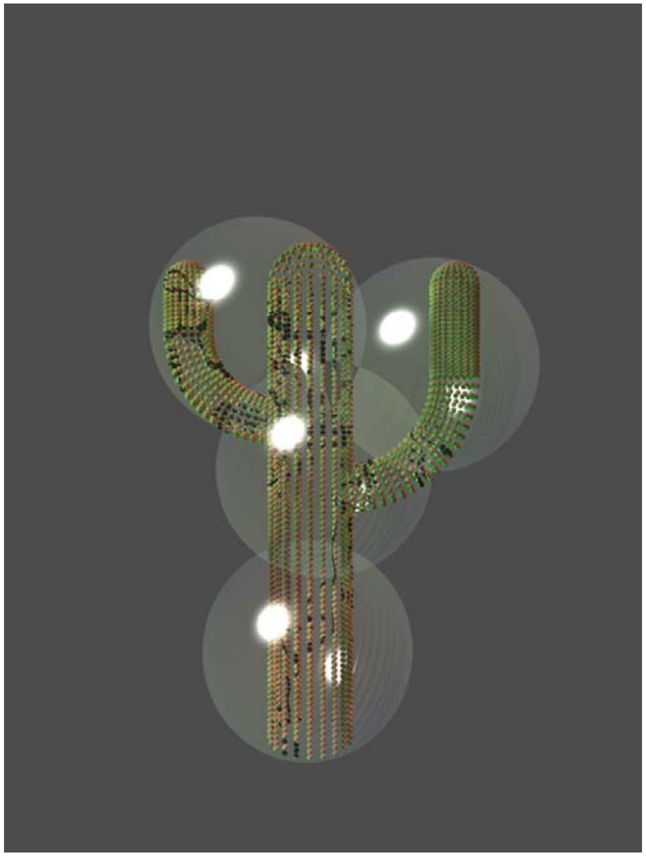


2/4 centers almost outside US

# 4-center



# 4-center



# 4-center

# Current & Future Work

- In Practice :
  - Outliers?
  - 1-cylinder? $k$-center? $k$-Ecenters?
  - LP Core sets?
  - Warm Start?
  - $\mathcal{O}\left(\frac{nd}{\epsilon} + \frac{1}{\epsilon^4}\log^2\frac{1}{\epsilon}\right)$ Algorithm?

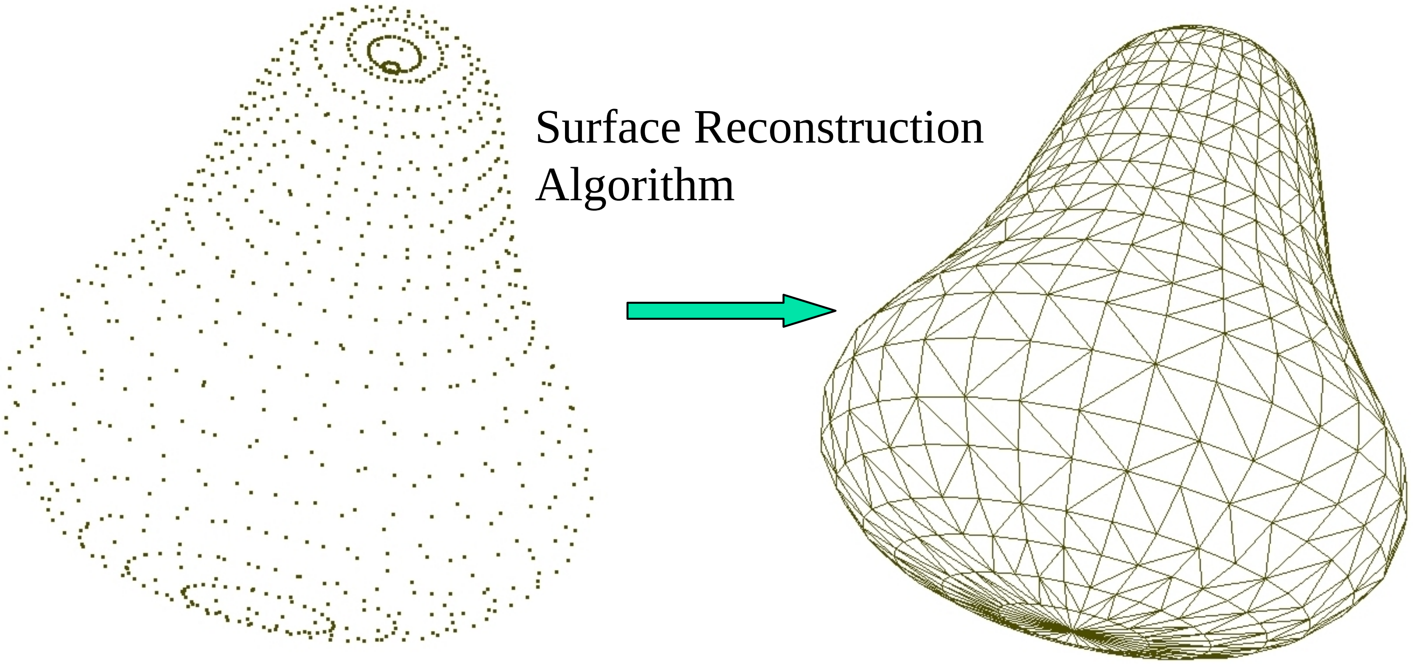
# Current & Future Work

- In Theory :
  - Optimal Core Set Size?
  - Dimension independent core sets for other LP-Type problems?
  - Tight Core Sets for various Distributions?
  - Smoothed Analysis?

# Real World Applications

- Reconstruction
- Hand Recognition
- Cache Oblivious Algorithms
- Finding good occluders

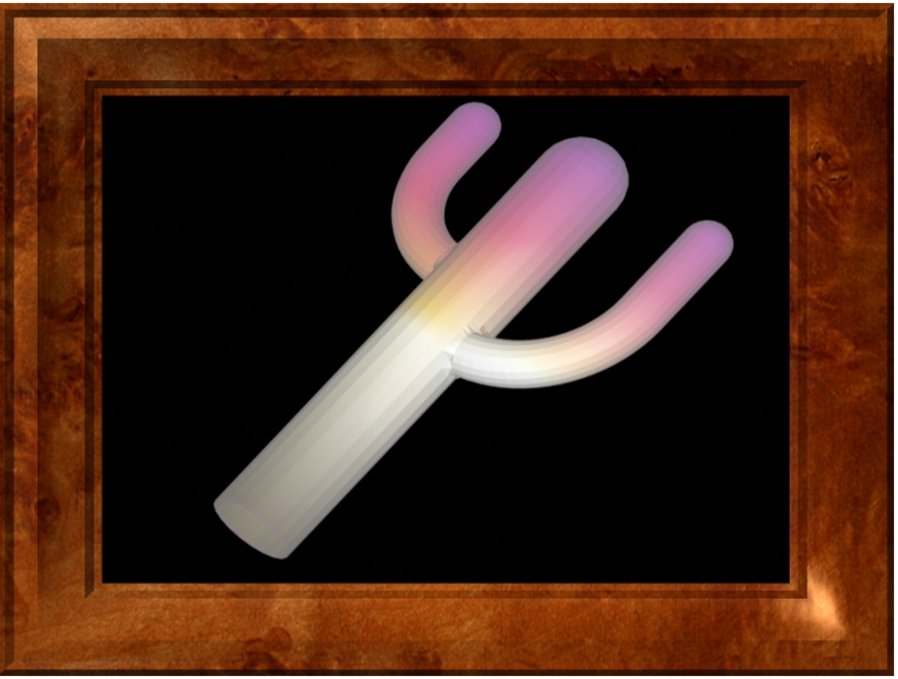
# Reconstruction



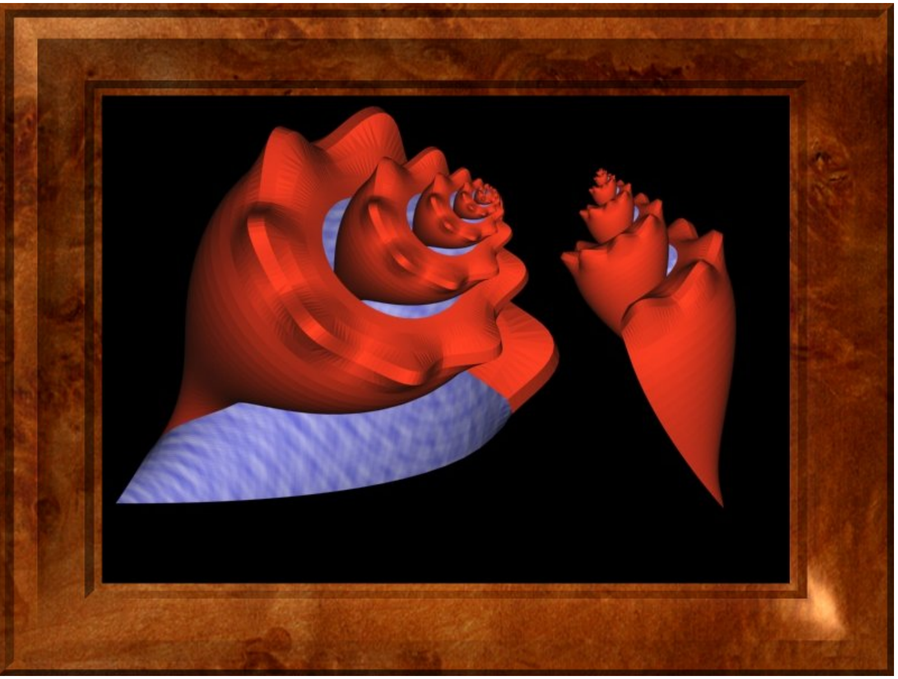
# Example Reconstruction

# Reviver

http://www.compgeom.com/reviver



# Reviver



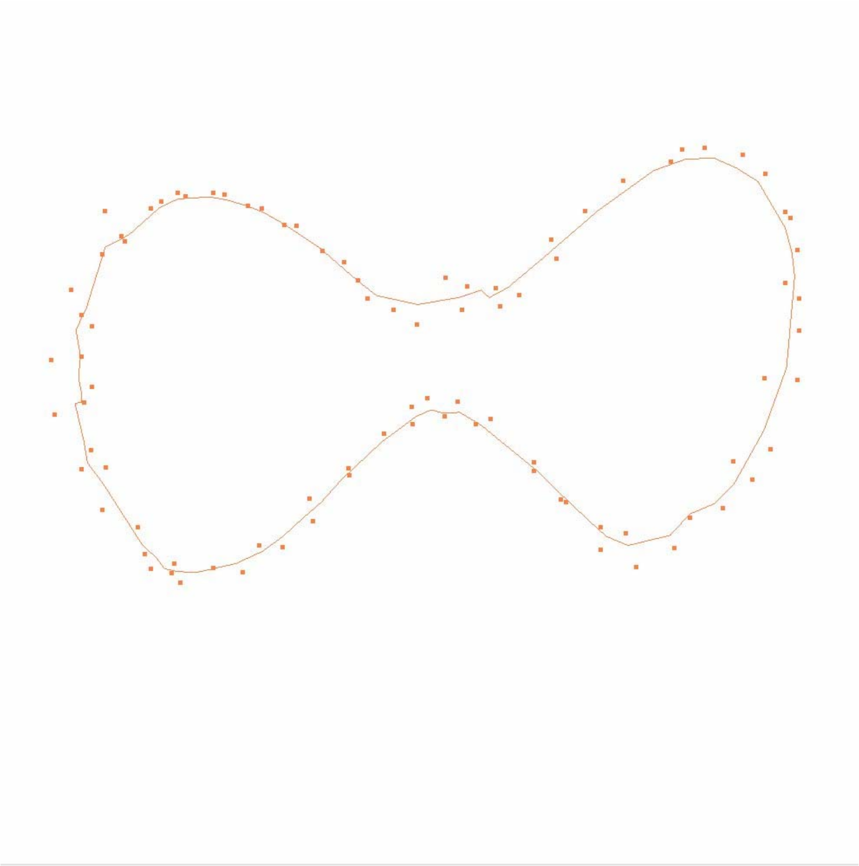
# Problems with Reviver

# Curve Reconstruction

- The same problem in 2 dimensions

- Why care?
  - Because it has applications too (object recognition, segmentation ...).
  - And it is the foundation of solving 3D reconstruction problems

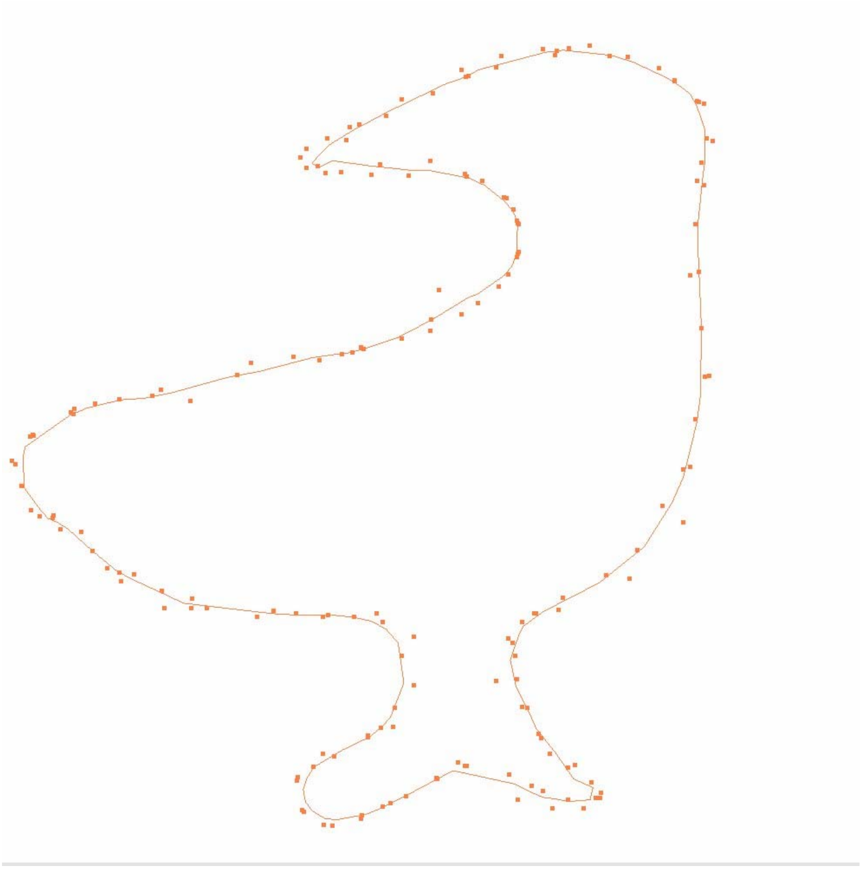
# Noise



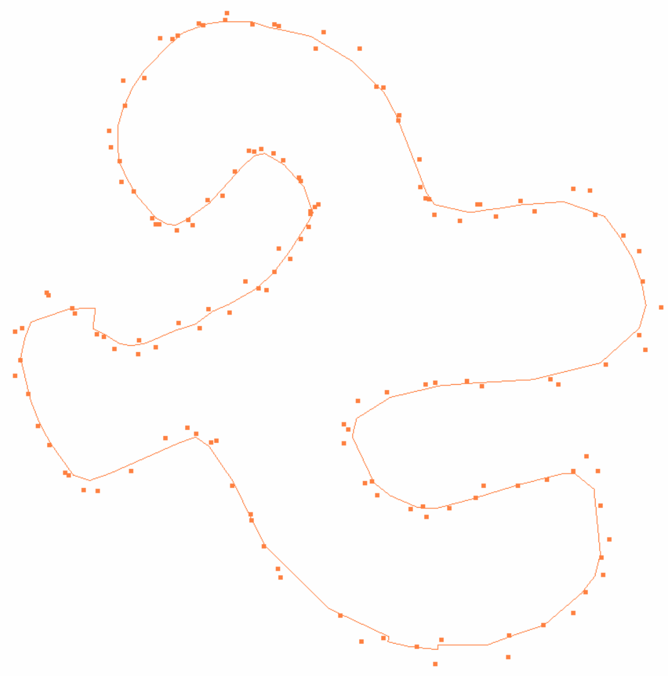
# Results Snapshot

- We gave and implemented an algorithm that reconstructs curves provably in presence of noise.
- The algorithm works by finding out local cluster centers in the data whose reconstruction gives a faithful representation of the data.

# Reconstruction



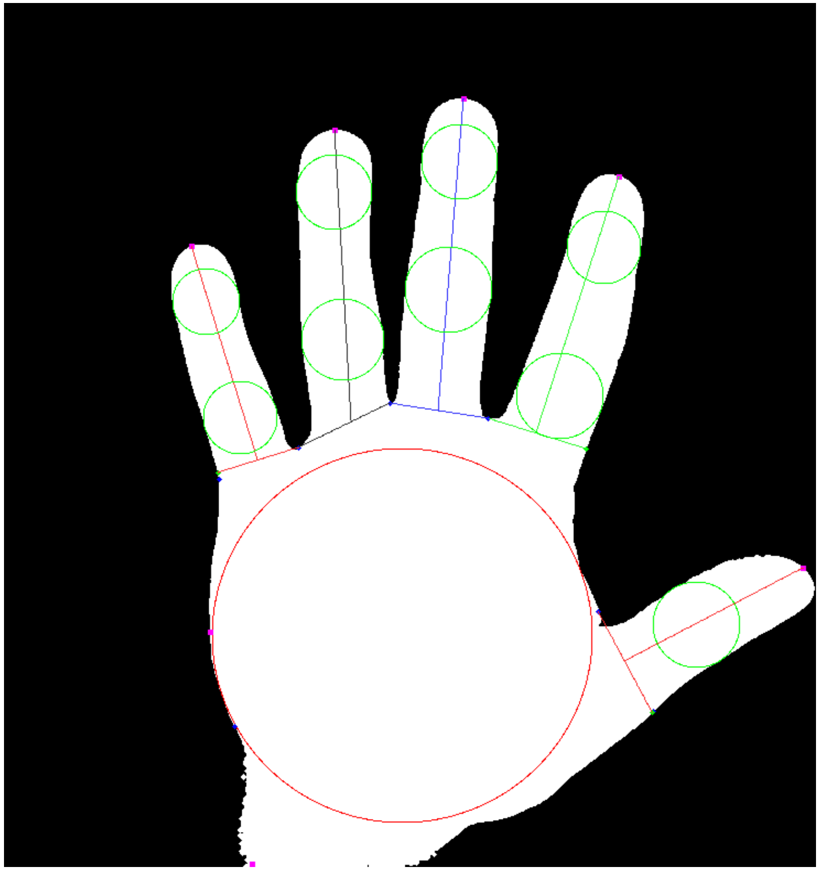
# Reconstruction



# Hand Recognition

Using Palm outlines

# Hand Recognition



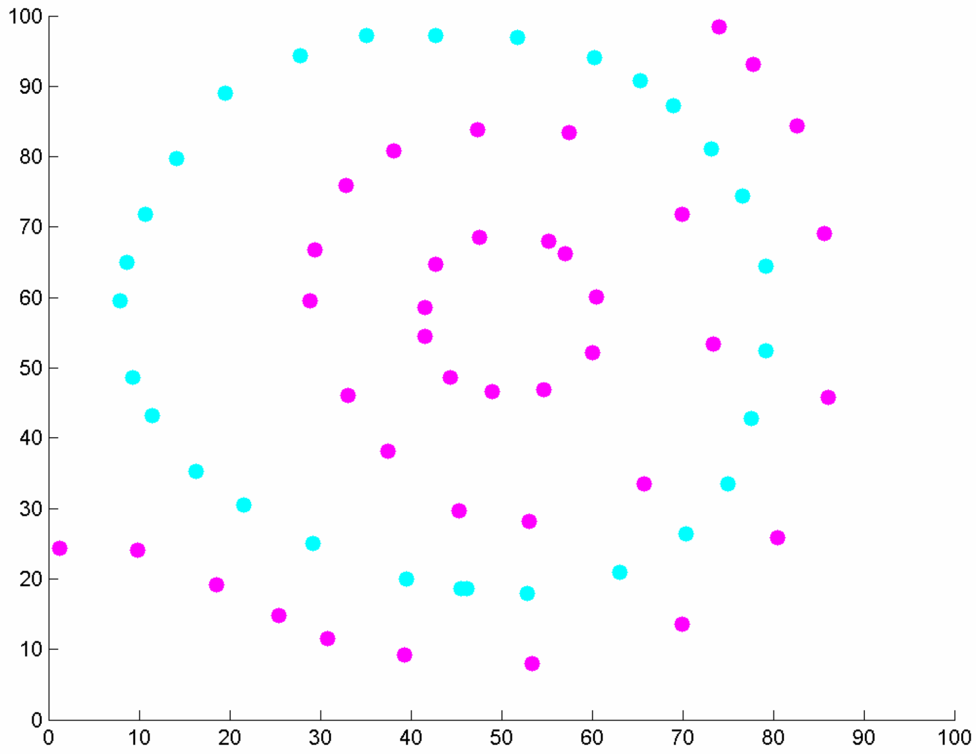
# Stages of Feature Extraction

- Thresholding
- Smoothing
- Boundary Finding
- Curvature Graph
- Distance Transform

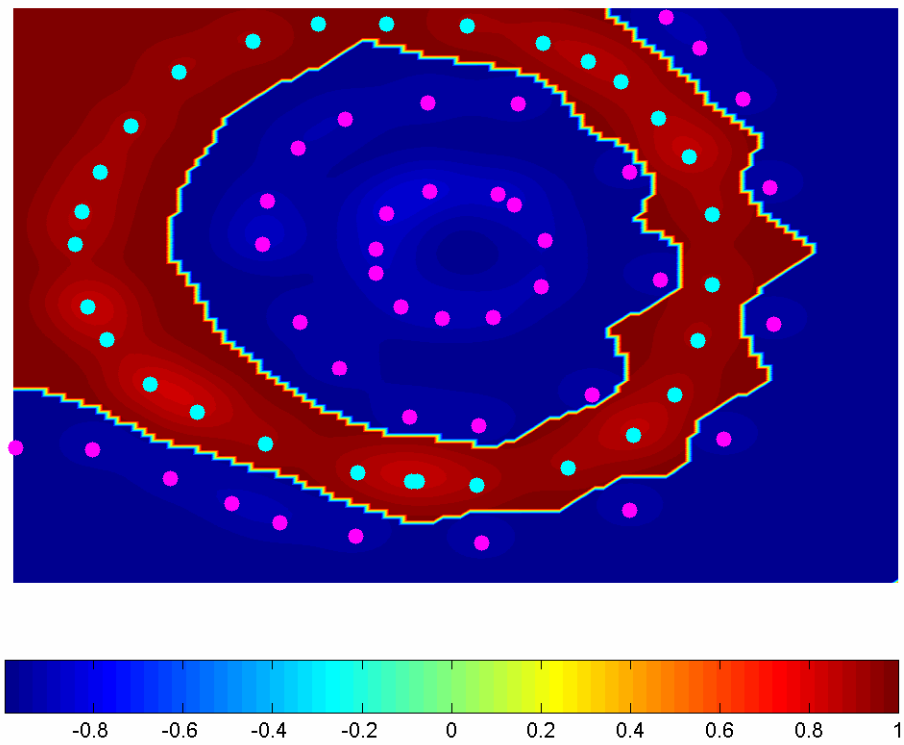
# Classification

- Extract 30 features from each hand image. (We collected 700 images.)
- Given a query hand image, identify which person it belong to
- We use Minimum enclosing balls in infinite dimensions for Classification!

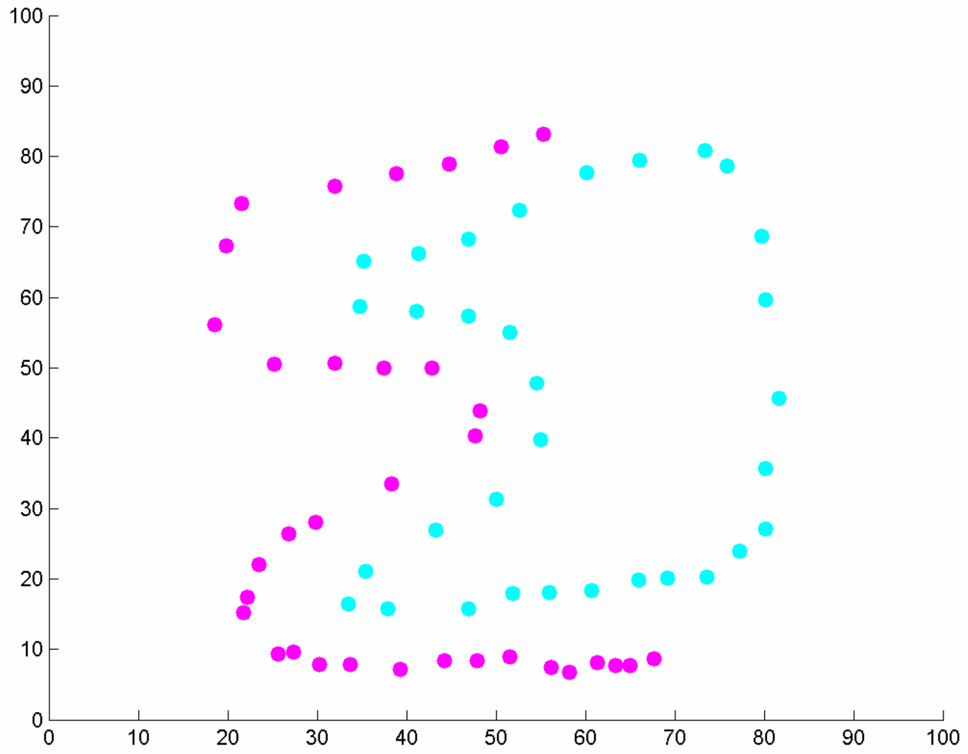
# MEB Classifier



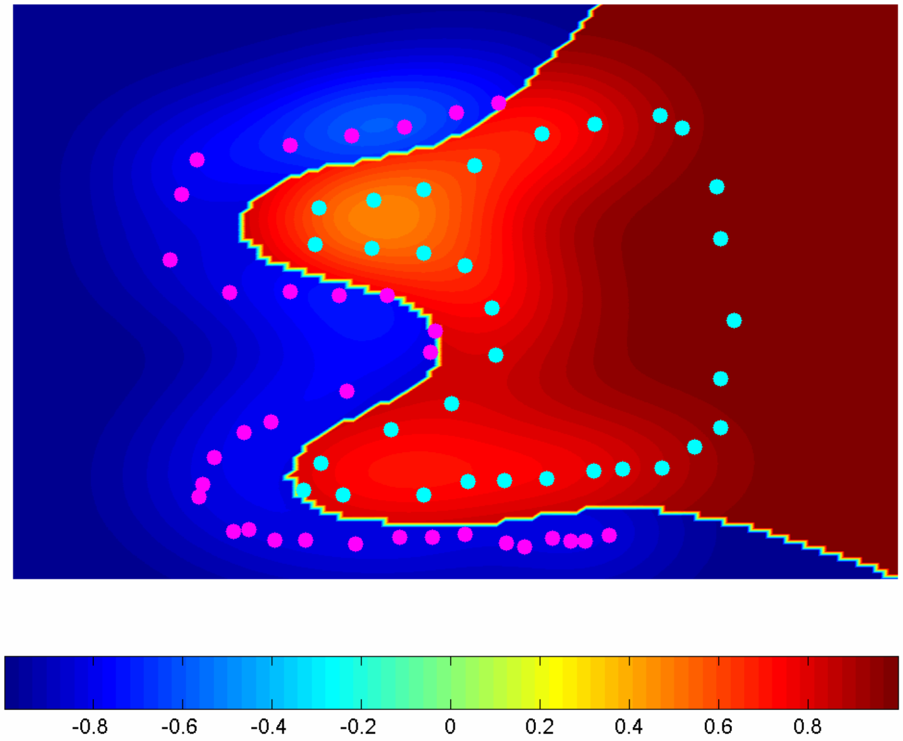
# MEB Classifier



# MEB Classifier



# MEB Classifier



# Misclassification Rates

| Training Set Size | Nearest Box | MEB | SVM |
|---|---|---|---|
| 3 | 2.81 | 1.53 | 4.77 |
| 4 | 1.74 | .94 | .93 |
| 5 | 1.72 | .39 | 1.93 |

# MEB Classifier

- We also tried MEB Classifier on Breast cancer data and compared it with other classifiers

# FNA data ( Fine Needle Aspiration )

- Wisconsin breast cancer database was used.
- Dataset contained 30 real valued features such as mean area, radius etc extracted from images.
- The dataset contained 569 (357 benign + 212 malignant) cases.
- Classified dataset using different classifers.

# RESULTS:

## CLASSIFICATION RESULTS USING VARIOUS CLASSIFIERS

| % | MEB | SVC | Neural Net | Parzen | Fischer | k-NN | ldc |
|---|---|---|---|---|---|---|---|
| Accuracy | 95.96 | 91.75 | 96.81 | 95.79 | 95.82 | 94.60 | 95.72 |
| Sensitivity | 97.76 | 99.55 | 98.55 | 98.44 | 99.27 | 96.31 | 99.33 |
| Specificity | 92.92 | 78.58 | 93.87 | 91.32 | 90.00 | 91.70 | 89.62 |

The results shown above were obtained by averaging the accuracy, sensitivity and specificity of each classifier over 10 runs

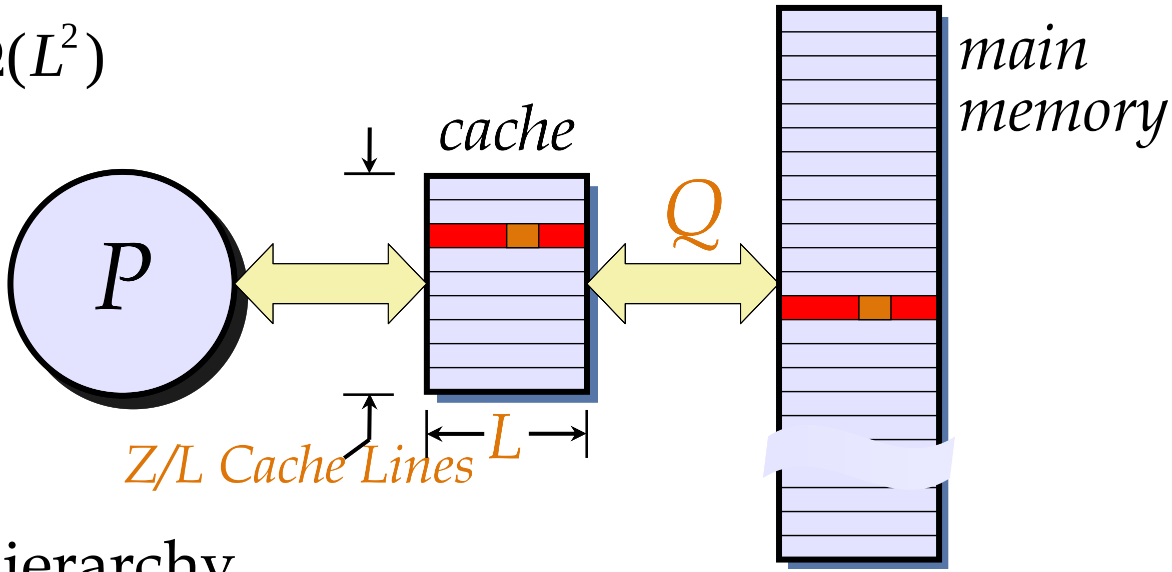
# Cache Oblivious Algorithms

# Cache Oblivious Algorithms

- Algorithms that take into account that memory access is not unit cost.
- And do not need any ‘voodoo’ parameters.

# (Z,L) Ideal Cache Model

$$Z = \Omega(L^2)$$




***Features:***

- Two-level hierarchy.
- Cache of size *Z*.
- Cache-line length *L*.
- Fully associative.
- Optimal, omniscient replacement.

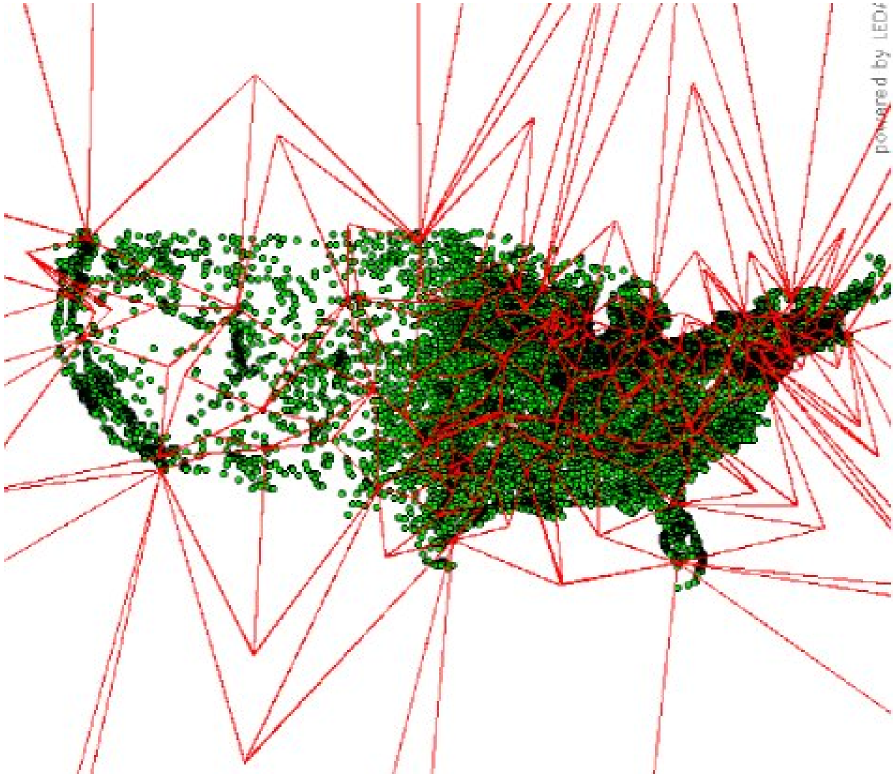
***Measures:***

- *Work W*.
- *Cache misses Q*.

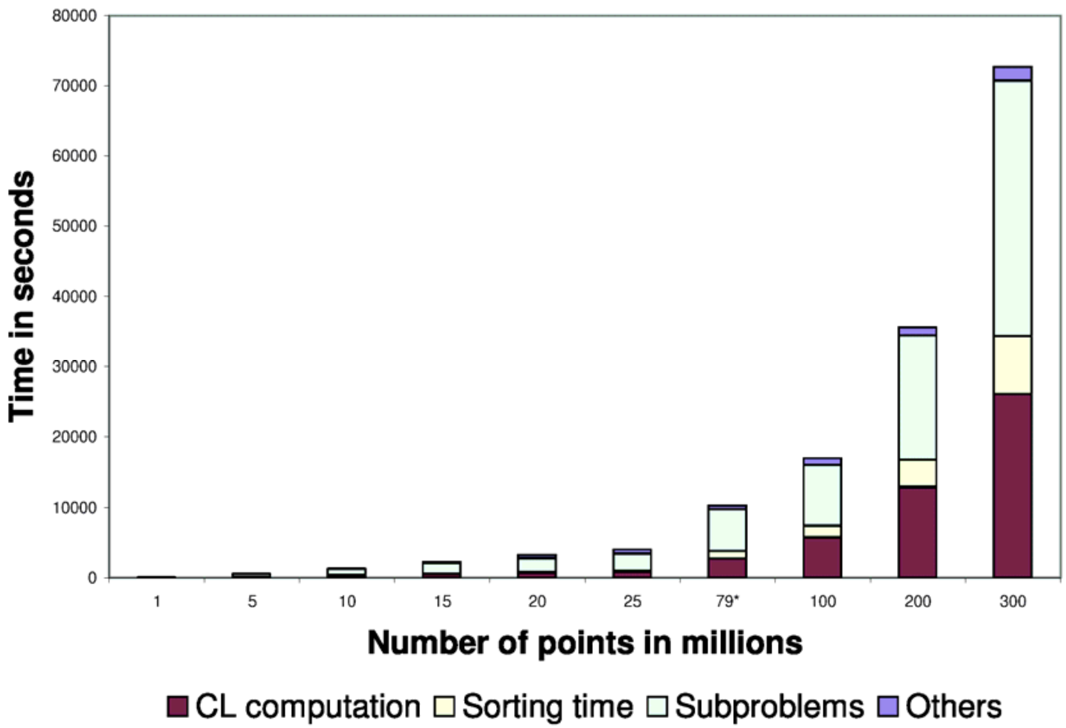
# Snapshot of main results

- We can do voronoi diagrams and delaunay triangulations in the cache oblivious model.
- We implemented a 2D Delaunay triangulation Algorithm that can do delaunay of massive data sets (based upon ideas from the cache oblivious algorithm)

# A Picture of cuttings



# Running Times in 2D



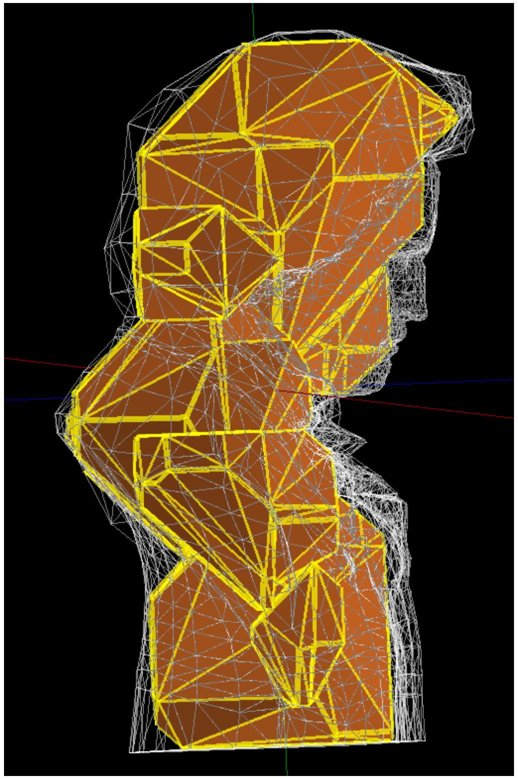
# Finding Good Occluders

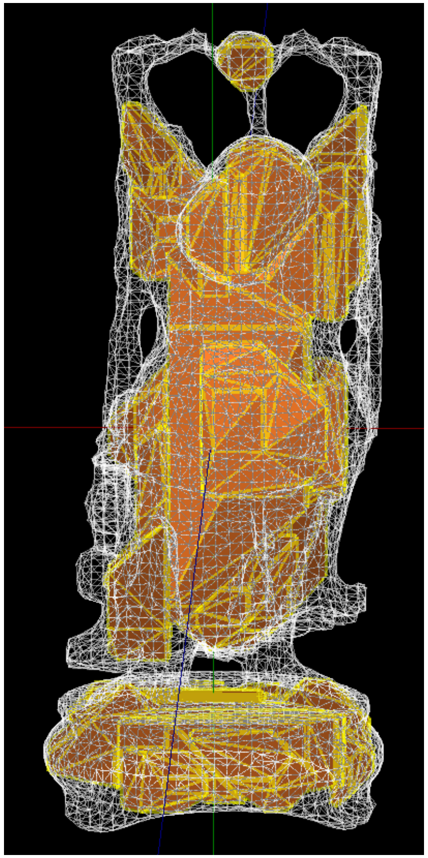
Preliminary Results

# Finding Good Occluders

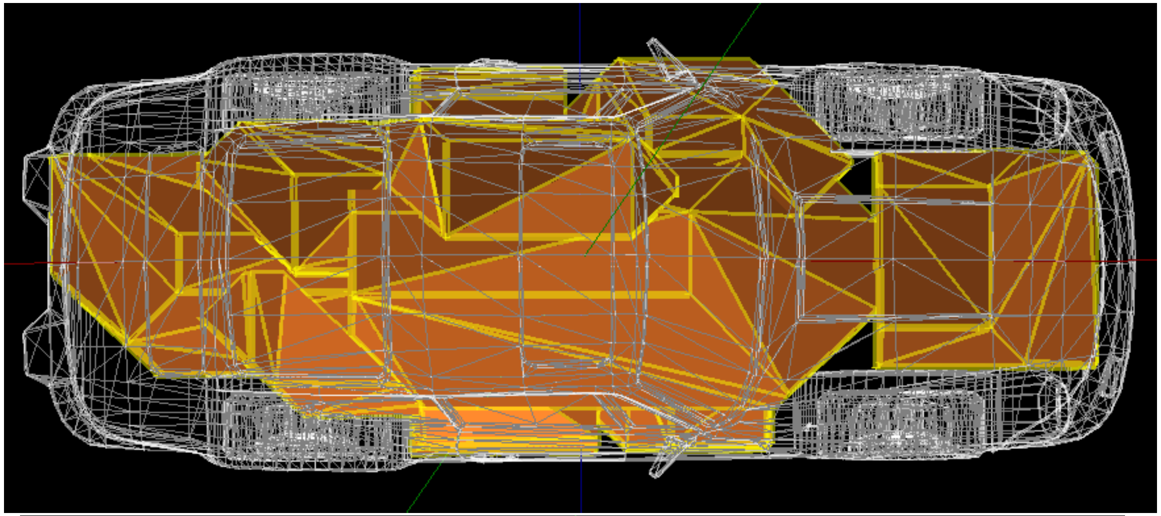
- Find large covering of the inside of a closed shape with convex bodies for visibility processing.

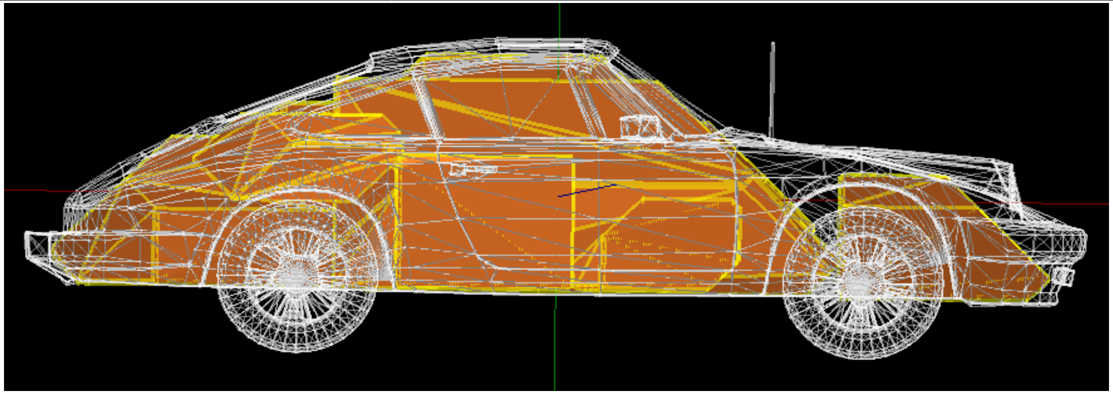
# Preliminary Results



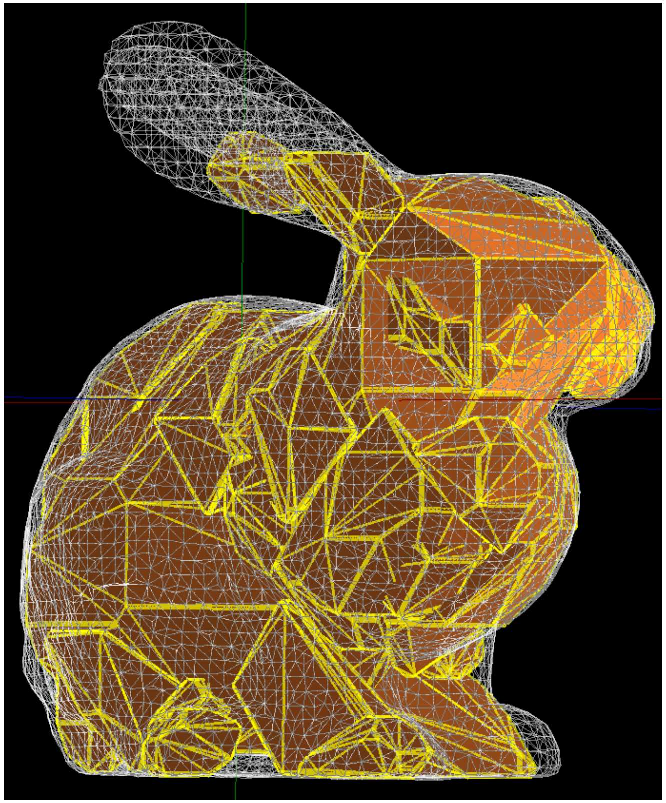


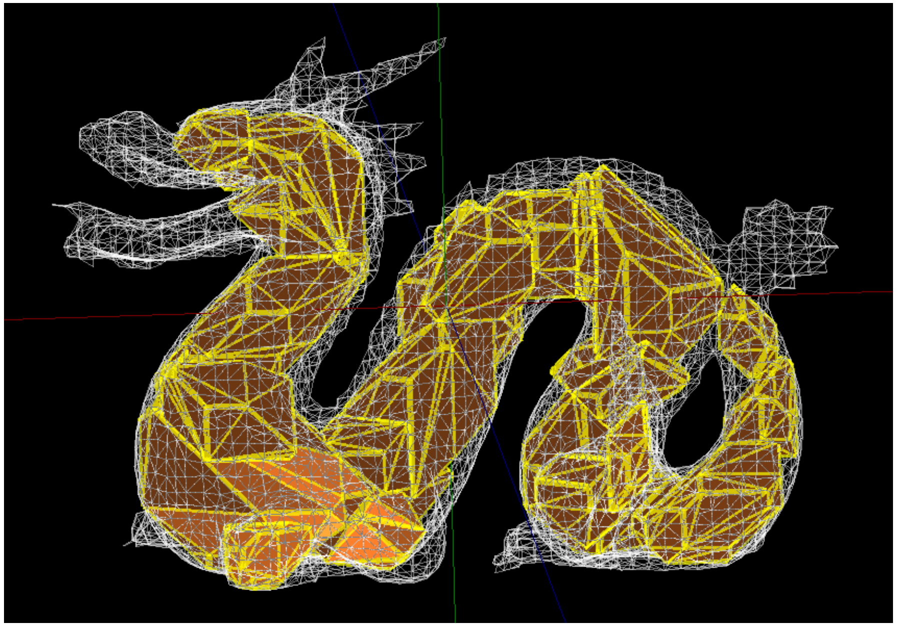
# Preliminary Results





# Preliminary Results





# Current Projects

- Implement provable reconstruction algorithms in 3D for massive data sets.
- Use/Improve MEB Classifier and design new classifiers for other applications
  - Currently work underway on detection of Cancer using Mass Spec data of blood serum. ( Joint work with Olaf Hall-Holt, Joe Mitchell, Marshall Bern )

# Future Work

# Areas

- Computational Geometry
- Clustering / Pattern Recognition
- Core Sets and Optimization
- Bio-Metrics
- Bio-Informatics (Mass Spec data Analysis)
- Surface Reconstruction with noise
- Scientific Computing (Cache issues)

# Open Problems

- Core Sets for other problems?
- Surface reconstruction with noise
- Provably good occluders?
- Cache Oblivious / Aware data structures and algorithms for graphics?

# Thanks for your attention

Questions?


## Acknowledgements

*Joseph S.B. Mitchell, Edgar A. Ramos, Alper Yildirim, Michael A. Bender, Esther M.Arkin, Peter Sanders, Siu-Wing Cheng, Tamal Dey, Stefan Funke, Sachin Jambawalikar, Saurabh Sethia, Wei Zhu, Pankaj Kumar*

*Stony Brook University*

*MPI-Saarbruecken.*

Who made this research possible?

# References

[1] N. Alon, S. Dar, M. Parnas and D. Ron. Testing of clustering. In *Proc. 41st Annual Symposium on Foundations of Computer Science*, pages 240–250. IEEE Computer Society Press, Los Alamitos, CA, 2000.

[2] A. Ben-Hur, D. Horn, H. T. Siegelmann and V. Vapnik. Support vector clustering. In *Journal of Machine Learning. revised version Jan 2002*, 2002.

[3] M. Bădoiu, S. Har-Peled and P. Indyk. Approximate clustering via core-sets. *Proceedings of 34th Annual ACM Symposium on Theory of Computing*, pages 250–257, 2002.

[4] M. Bădoiu and K. L. Clarkson. Smaller core-sets for balls. In *Proceedings of 14th ACM-SIAM Symposium on Discrete Algorithms*, to appear, 2003.

[5] B. Gärtner and S. Schönherr. An efficient, exact, and generic quadratic programming solver for geometric optimization. In *Proceedings of 16th Annual ACM Symposium on Computational Geometry*, pages 110–118, 2000.

[6] A. Goel, P. Indyk and K. R. Varadarajan. Reductions among high dimensional proximity problems. In *Poceedings of 13th ACM-SIAM Symposium on Discrete Algorithms*, pages 769–778, 2001.

[7] J. Matoušek, Micha Sharir and Emo Welzl. A subexponential bound for linear programming. In *Proceedings of 8th Annual ACM Symposium on Computational Geometry*, pages 1–8, 1992.

[8] Y. E. Nesterov and A. S. Nemirovskii. *Interior Point Polynomial Methods in Convex Programming*. SIAM Publications, Philadelphia, 1994.

[9] Y. E. Nesterov and M. J. Todd. Primal-dual interior-point methods for self-scaled cones. *SIAM Journal on Optimization*, 8:324–362, 1998.

[10] Sariel Har-Peled. Personal Communications.

[11] K. C. Toh, M. J. Todd and R. H. Tütüncü. SDPT3 — a Matlab software package[1] for semidefinite programming. *Optimization Methods and Software*, 11:545–581, 1999.

[12] G. Zhou, J. Sun and K.-C. Toh. Efficient algorithms for the smallest enclosing ball problem in high dimensional space. Technical report, 2002. To appear in Procedings of Fields Institute of Mathematics.


[1] `http://www.math.nus.edu.sg/~mattohkc/sdpt3.html`