

Geometric Learning Algorithms

Stephen M. Omohundro

International Computer Science Institute

Berkeley, California

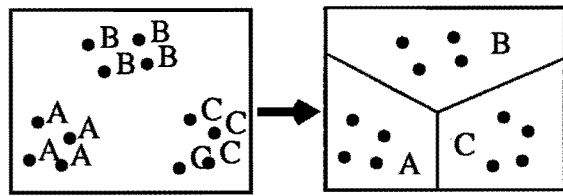
- Density estimation, classification, mappings
- Average case algorithms
- K-d trees and nearest neighbors
- Balltrees and boxtrees
- Robot arm task
- Bumptrees and mapping learning

Geometric Learning Algorithms

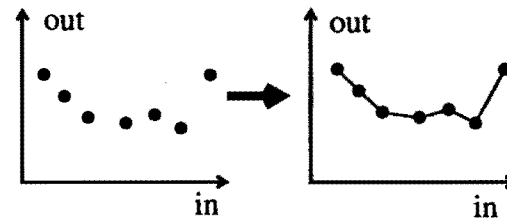
- *Geometric* - Mathematics
- *Learning* - Statistics
- *Algorithms* - Computer Science

Some Basic Tasks

- Density Estimation
- Classification learning
- Smooth nonlinear mapping learning

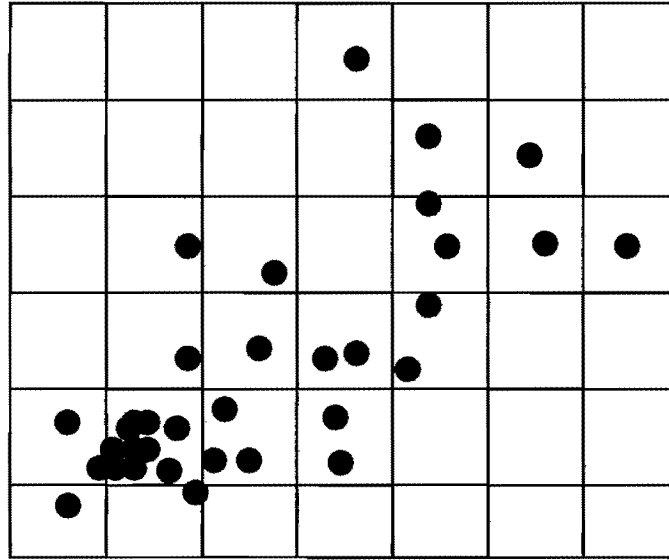


Classification Learning



Mapping Learning

Density estimation by Histogram



- Bins too large -> washes out the fine structure
- Bins too small -> only one or no samples per bin
- Correct size for one region may be incorrect for another

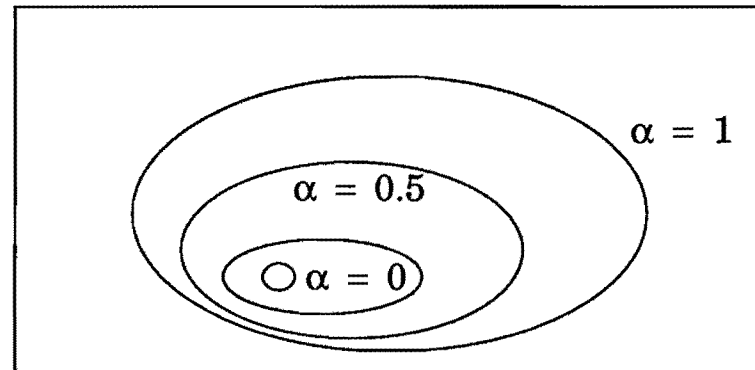
Volume Density Estimator

- Instead of taking regions of fixed volume and counting samples
- take regions with a fixed number of samples and measure volume

- eg. Take inverse volume of n th nearest neighbor sphere
- Get small regions in high density parts of the space
- Large regions in low density parts of the space

- Need to compute the distance to the n th nearest neighbor.

The beta distribution and non-parametric statistics.



Let S_α be a nested family of sets parameterized by α such that $p(S_\alpha) = \alpha$. If we draw N points and choose the smallest set S_α containing exactly n points, then the α 's are distributed according to the Beta distribution:

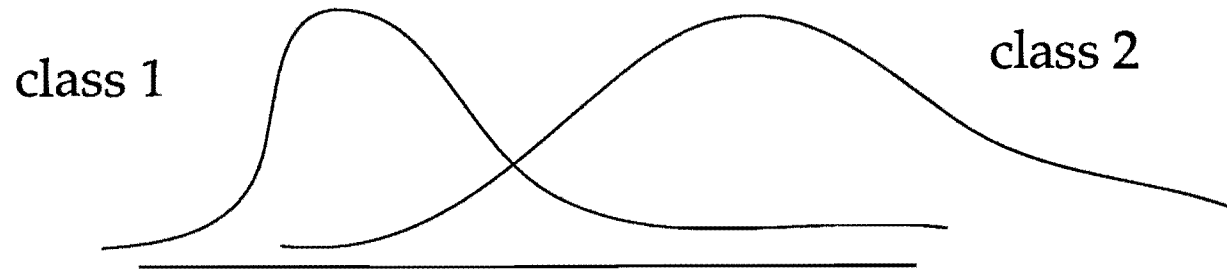
$$p_n(\alpha) = \frac{N!}{(n-1)!(N-n)!} \alpha^{n-1} (1-\alpha)^{N-n}$$

$$E(\alpha) = \frac{n}{N+1} \rightarrow \frac{n}{N}$$

$$\sigma^2(\alpha) \rightarrow \frac{n}{N^2}$$

Nearest Neighbor Classification

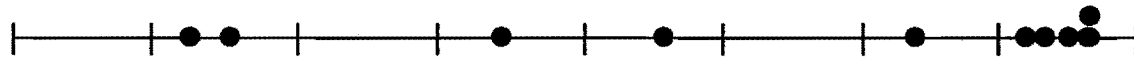
- Baye's optimal approach would choose that class for which the posterior distribution is largest.



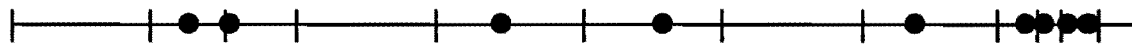
- Typically don't know the distributions, however.
- Nearest neighbor assigns a point the class of the nearest example.
- Asymptotically error is less than twice that of Bayes optimal.
- Non-parametric.

Double bucket sort: an adaptive algorithm

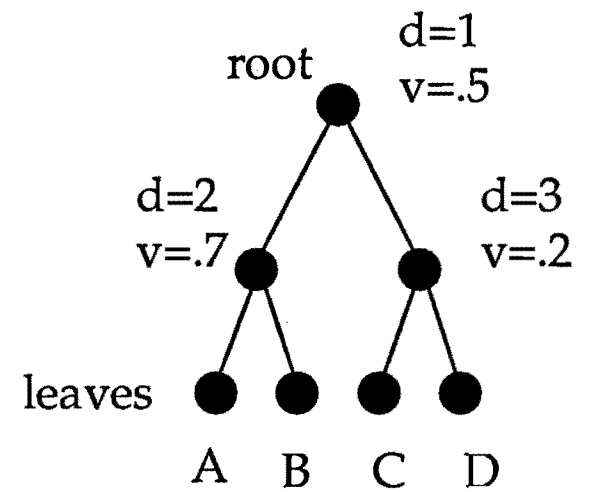
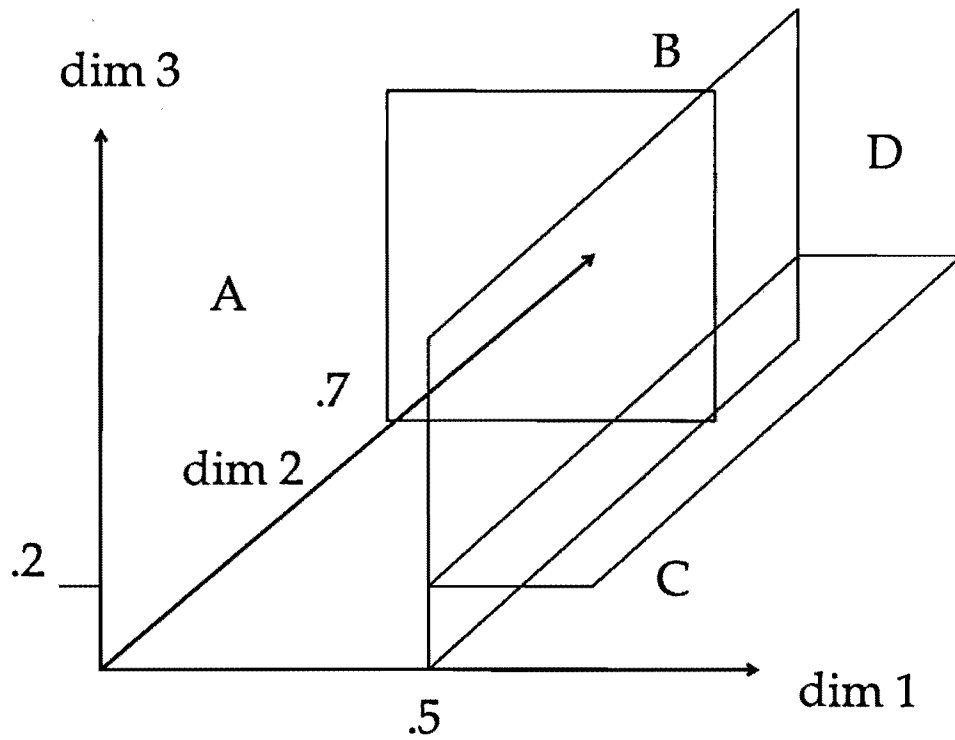
- Sorting n values in $[0,1]$ takes $O(n \log n)$ worst case.
- Would like good average case performance
- but, What distribution do you average over?
- Assume samples come from an unknown distribution, want good average with respect to that distribution.
- Bucket sort: Break interval into equal sized buckets, sort each one.



- Double bucket sort: Further break up intervals into a number of subintervals proportional to the samples which fell into them.
- Linear expected time $O(n)$, beats quicksort by factor of 2 in practice.



K-d Trees

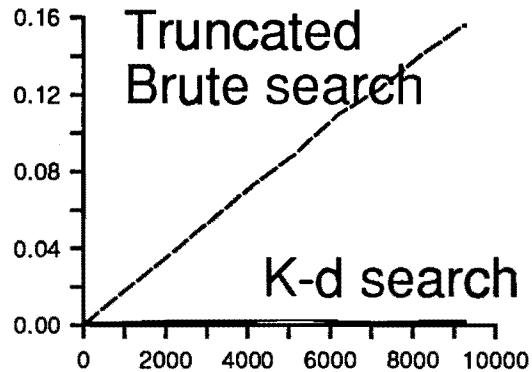


K-d trees can find n nearest neighbors in \log expected time.

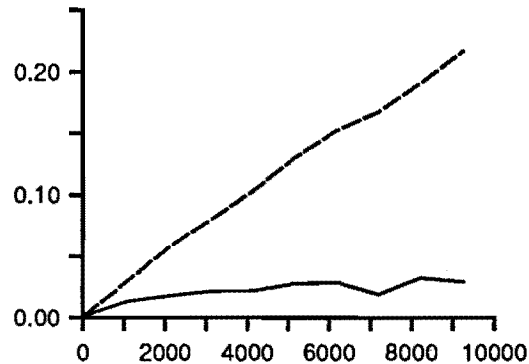
If N samples are drawn from a non-vanishing, smooth distribution on a compact region, then we can use a k -d tree to find the n nearest neighbors of a new sample in $O(\log N)$ expected time, asymptotically for large N .

Idea: Asymptotically, volume of n nearest neighbor ball is beta distributed. With k -d construction, tree regions are also beta distributed. n nearest neighbor ball will overlap only a constant number of tree regions on average. Using branch and bound, we do \log time initial search and then constant extra work.

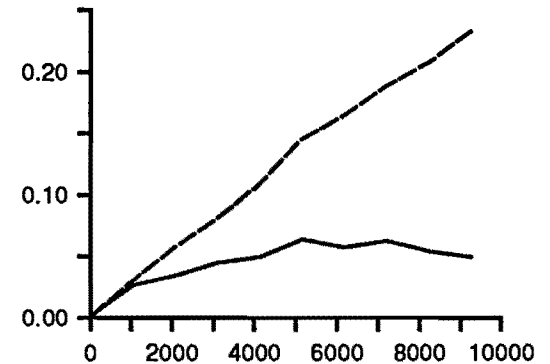
K-d Tree Search Dimension Dependence



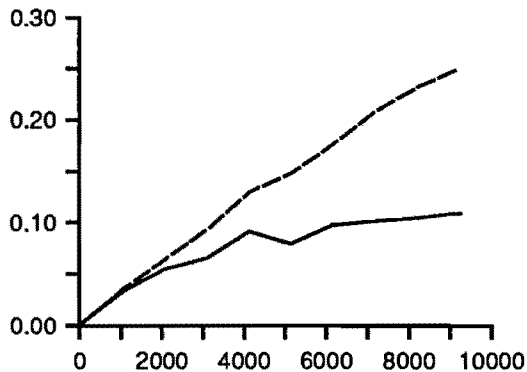
4 Dimensions



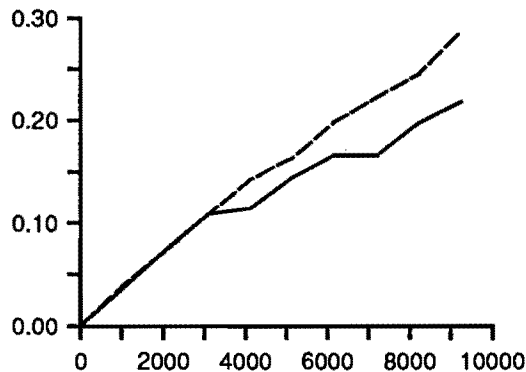
8 Dimensions



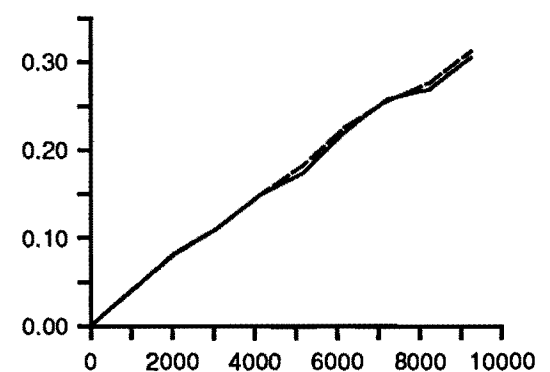
9 Dimensions



10 Dimensions



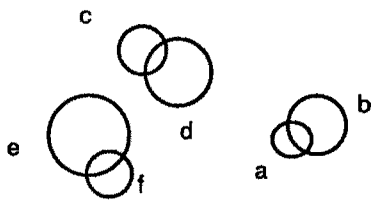
11 Dimensions



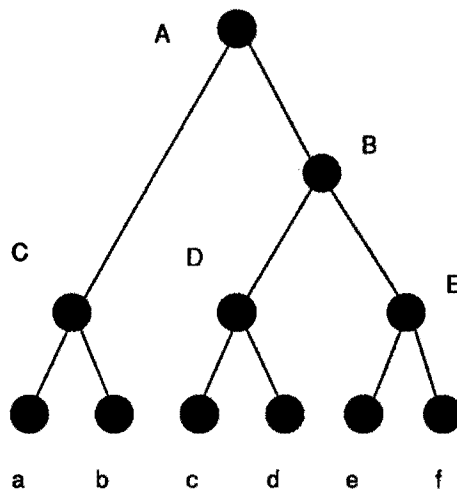
12 Dimensions

Average nearest neighbor search time (secs) vs number of points

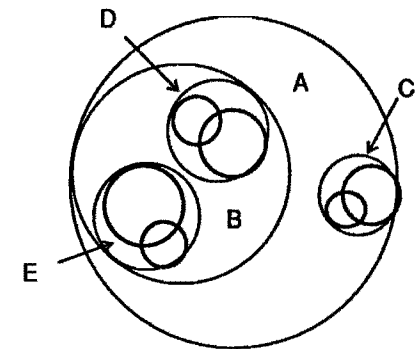
A 2-dimensional balltree: Leaf balls, tree structure, all balls.



A)



B)



C)

Balltree queries.

Pruning:

- Return leaf balls containing a query point.

Branch and bound queries:

- Return nearest leaf to query point.

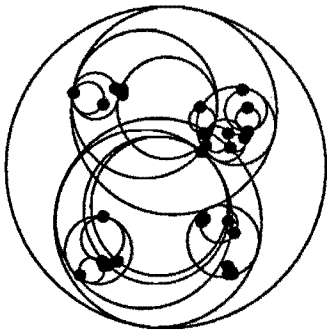
Distribution independent average performance:

If leaves and queries are drawn from an underlying distribution ρ then would like good performance on average with respect to ρ .

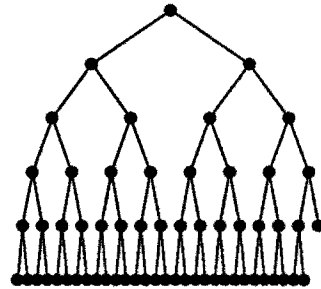
Five balltree construction algorithms.

- **K-d algorithm**
Split most spread dimension at median.
- **Top-down algorithm**
Split best dimension at point to minimize new ball volume.
- **Insertion algorithm**
Insert ball on-line at minimum volume insertion location.
- **Cheap insertion algorithm**
Insert ball on-line at heuristically good location.
- **Bottom up algorithm**
Repeatedly pair the best two balls.

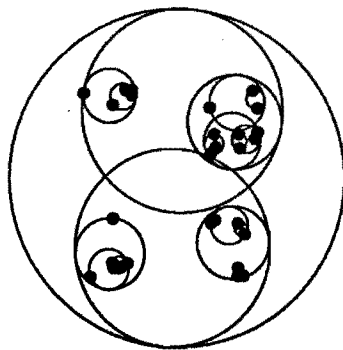
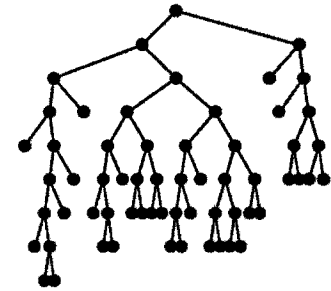
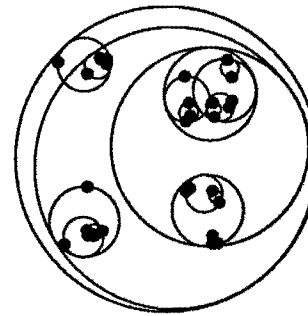
The balltrees produced from a Cantor set by the five construction algorithms.



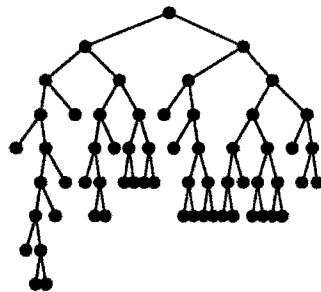
Kd



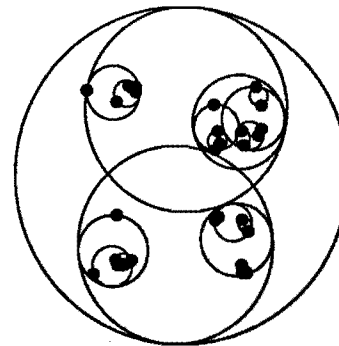
Insertion and
Cheap insertion



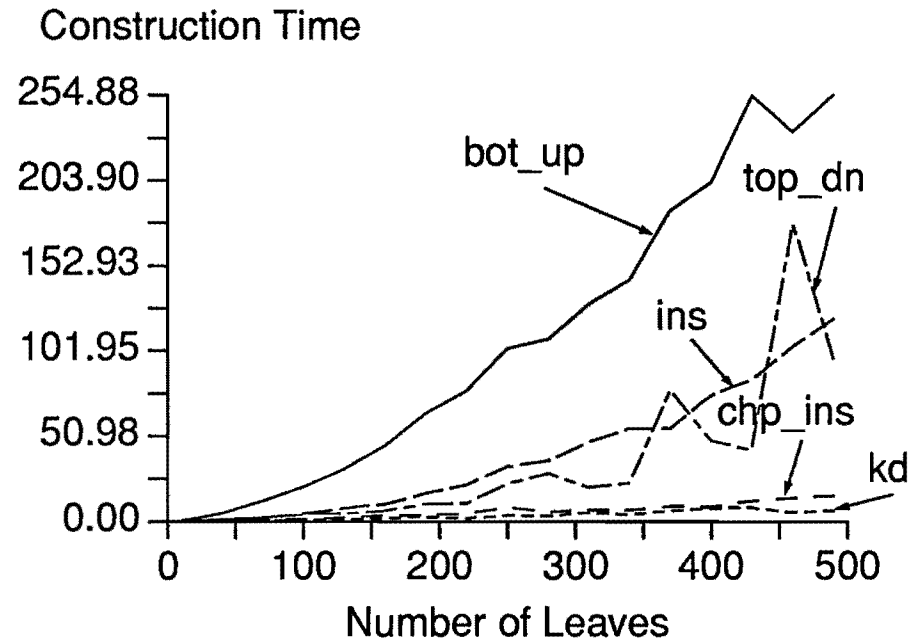
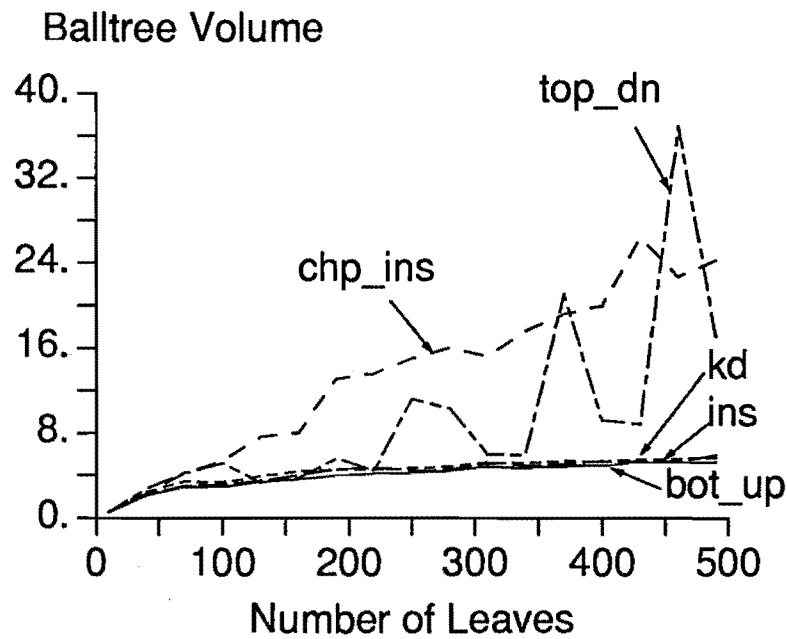
Top down



Bottom up

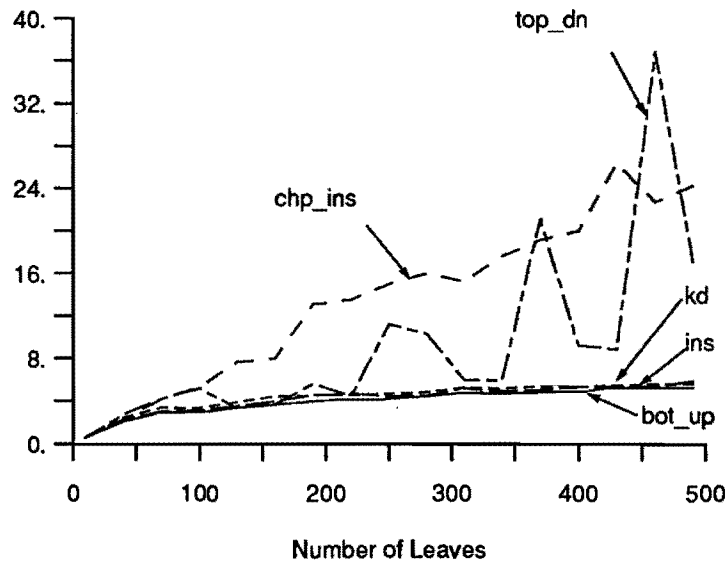


Balltree volume and construction time vs. number of 2-d uniformly distributed point leaves.

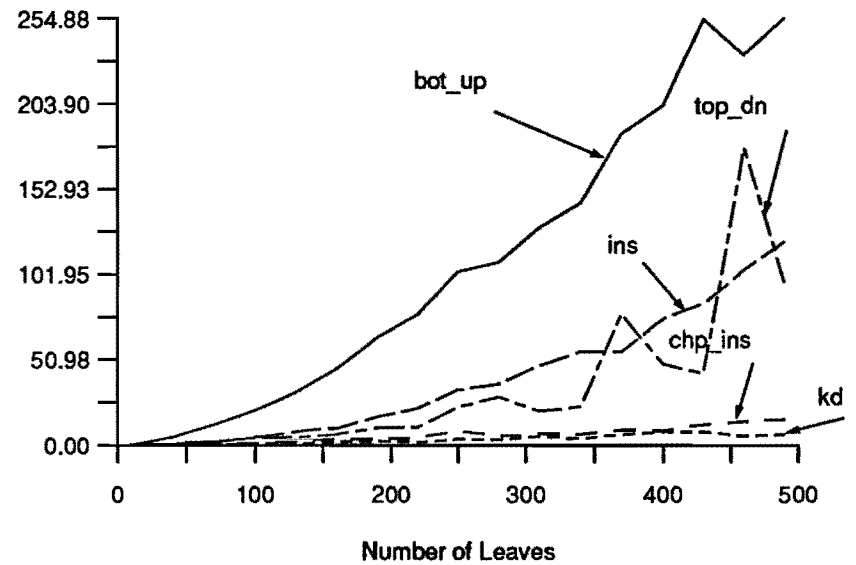


Balltree volume and construction time vs. number of 2-d uniformly distributed point leaves.

Balltree Volume

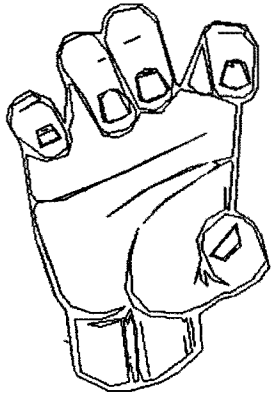


Construction Time

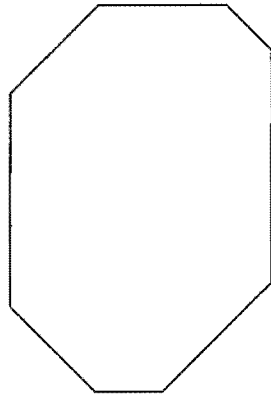


Octbox image trees

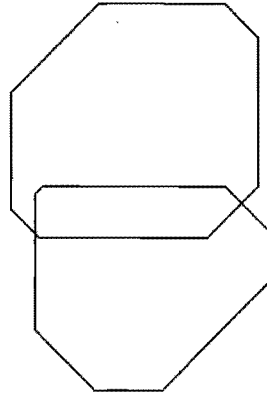
289 leaves, depth 12, average depth 8.7



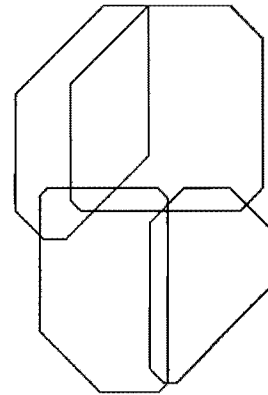
edges



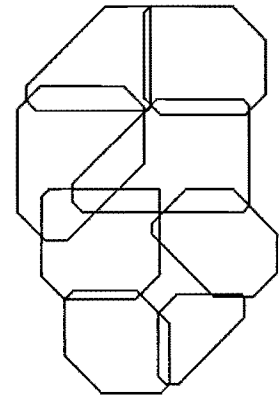
level 0



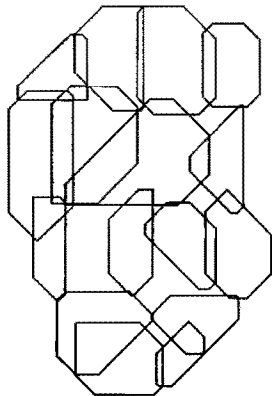
level 1



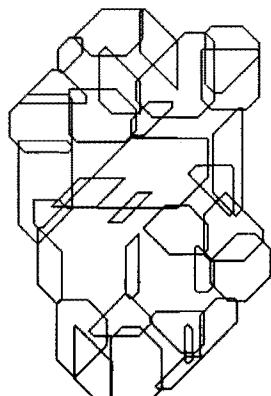
level 2



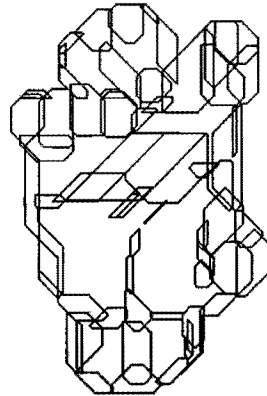
level 3



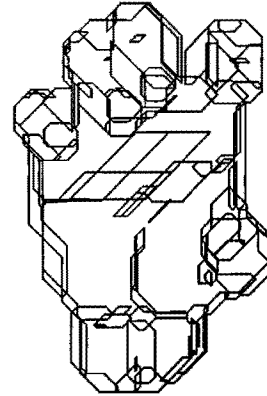
level 4



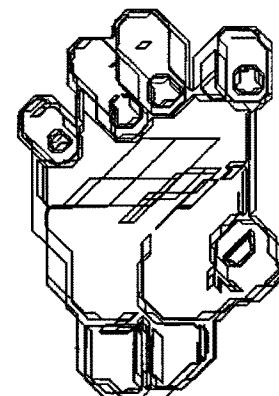
level 5



level 6

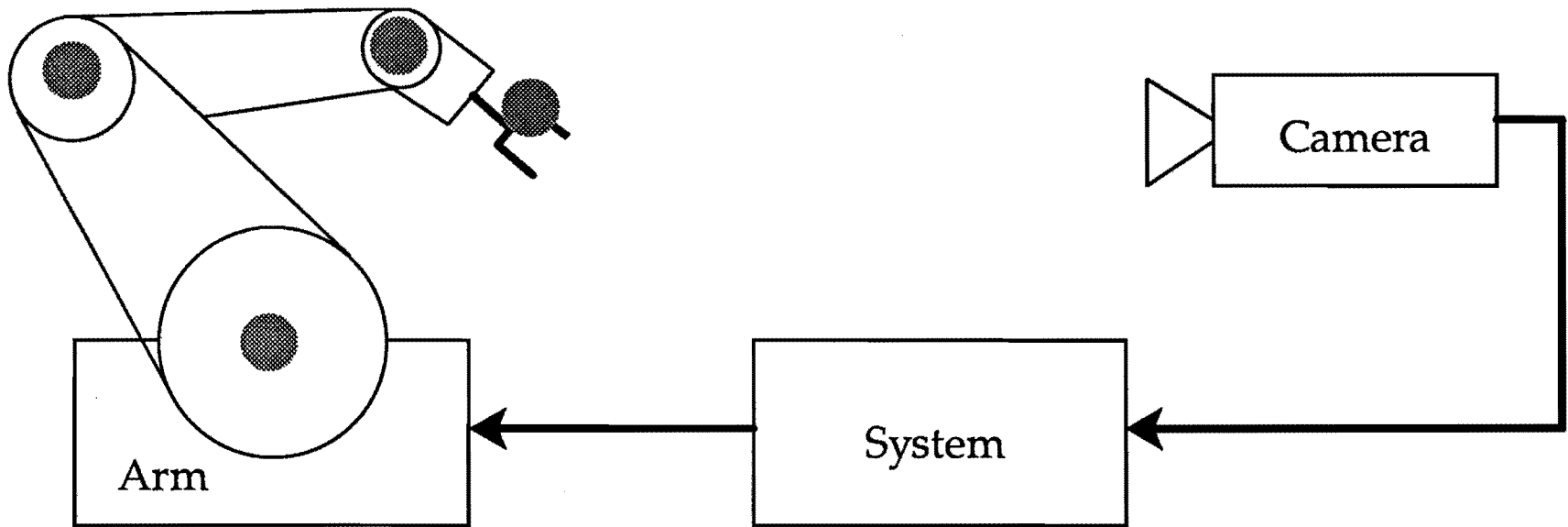


level 7



level 12

Robot Arm Task



Kinematic space

R^3

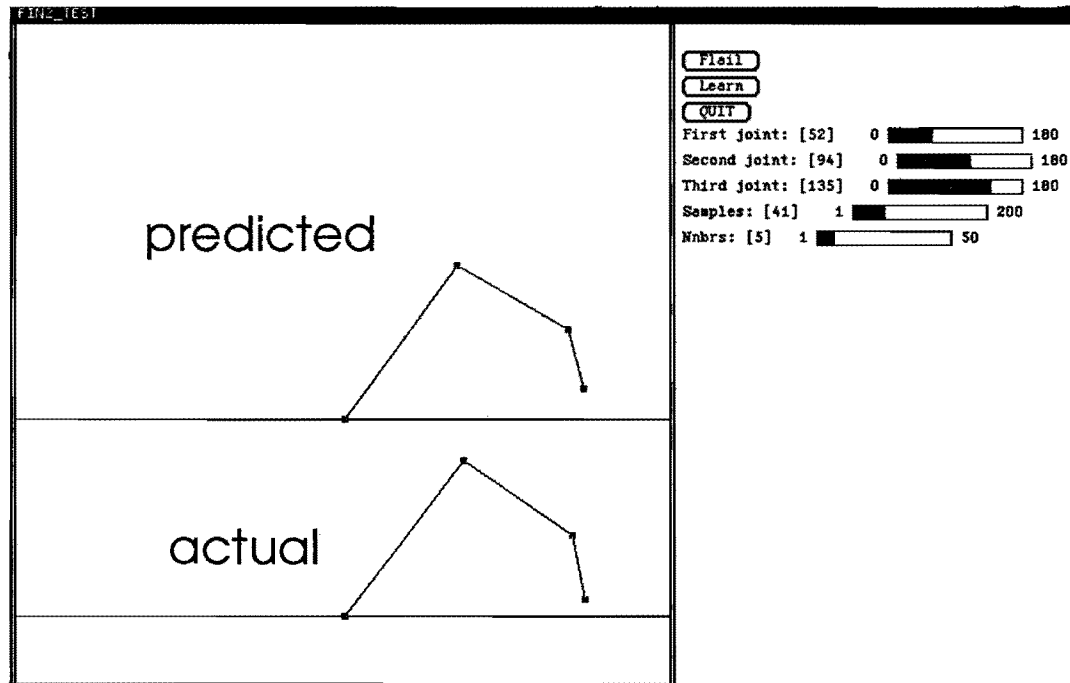


Visual space

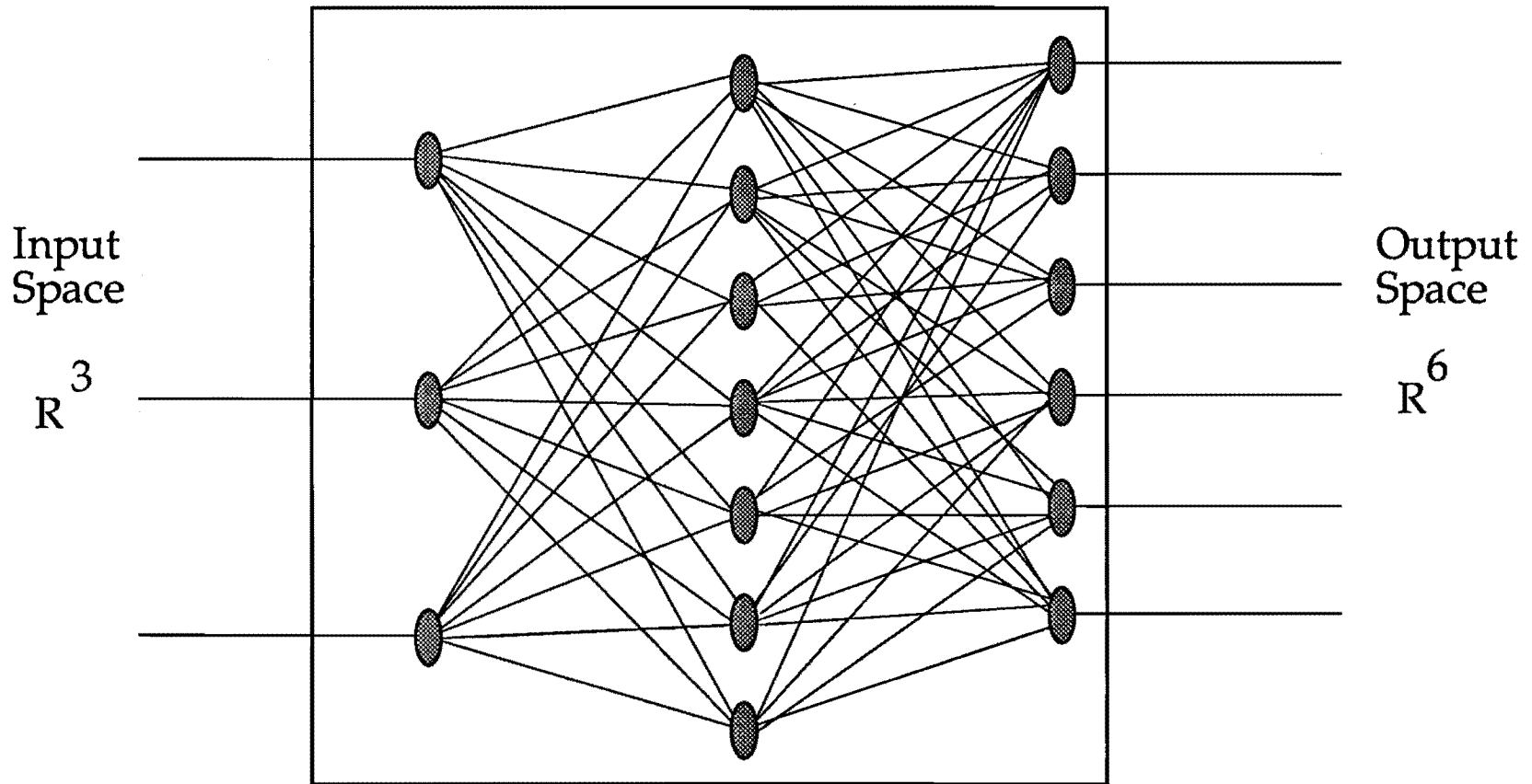
R^6

Setup studied by Bartlett Mel in "Connectionist Robot Motion Planning"

Kinematic to Visual map testbed



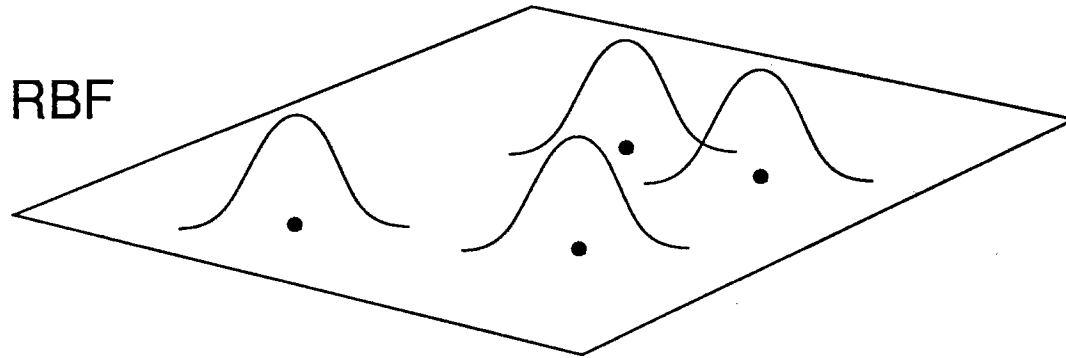
Backprop Net for Arm Task



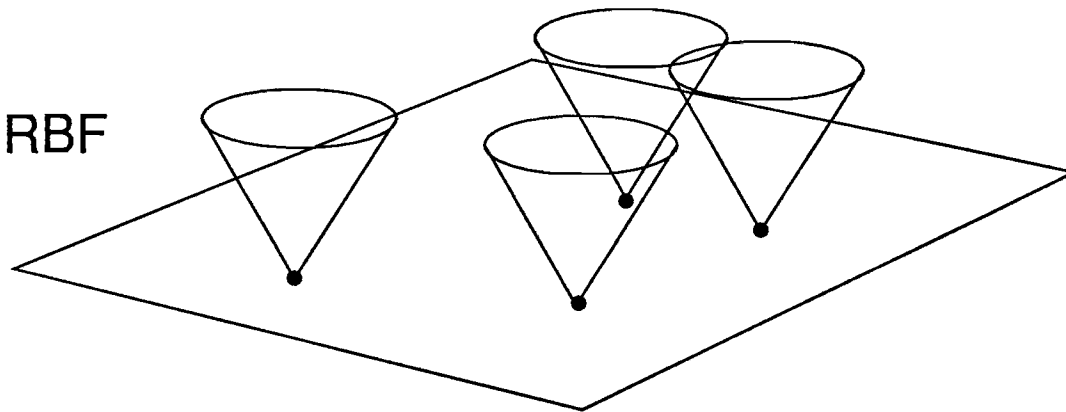
Radial Basis Functions

$$f(x) = \sum_i w_i g_i(x - x_i)$$

Gaussian RBF

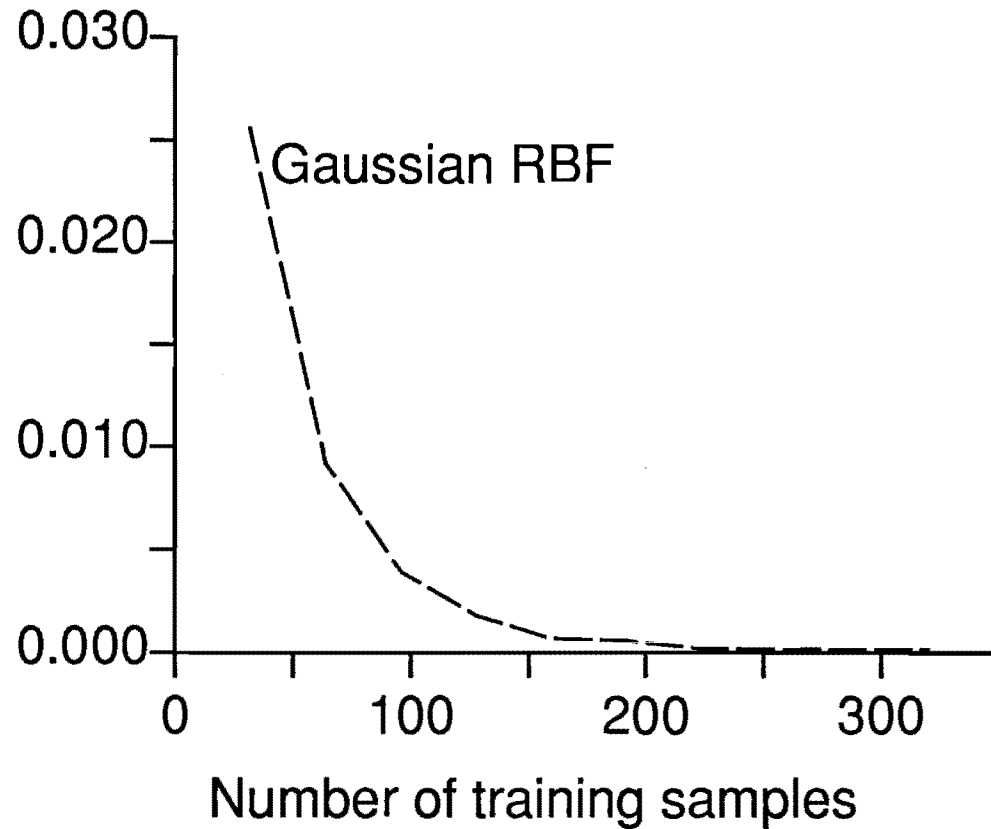


Linear RBF

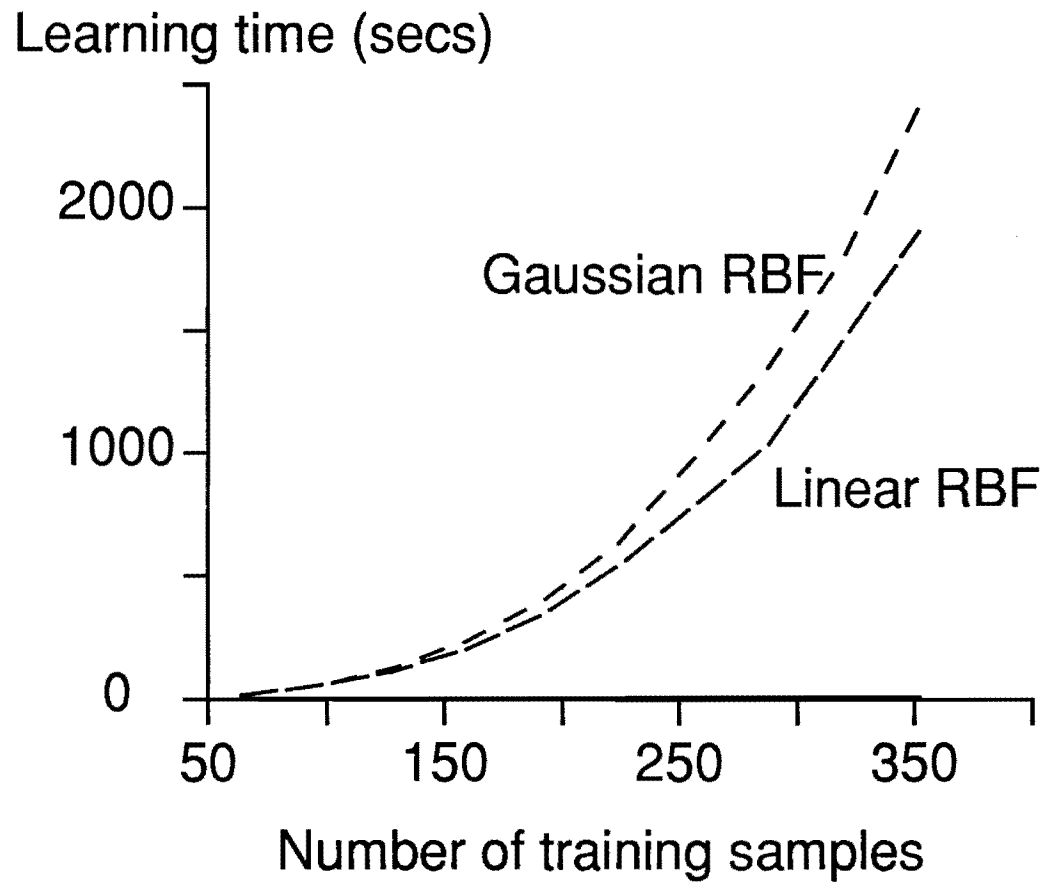


Radial Basis Function Error

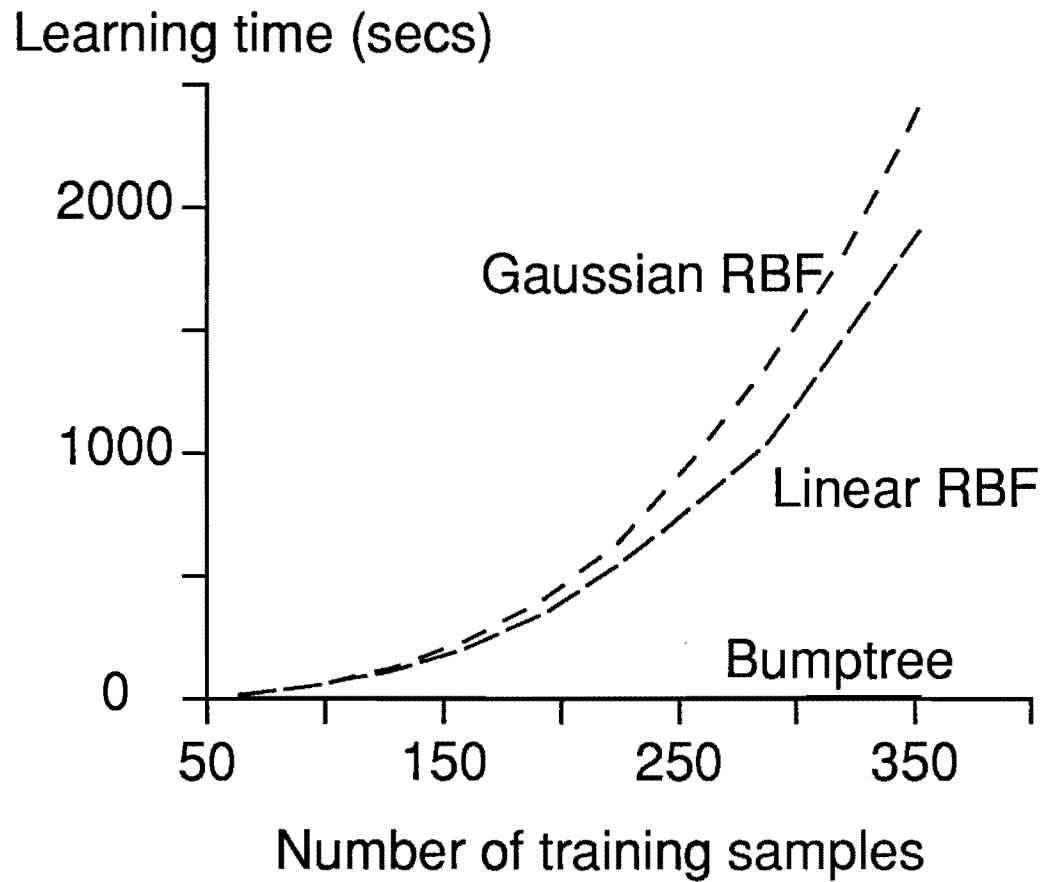
Mean Square Error



Radial Basis Function Learning Time

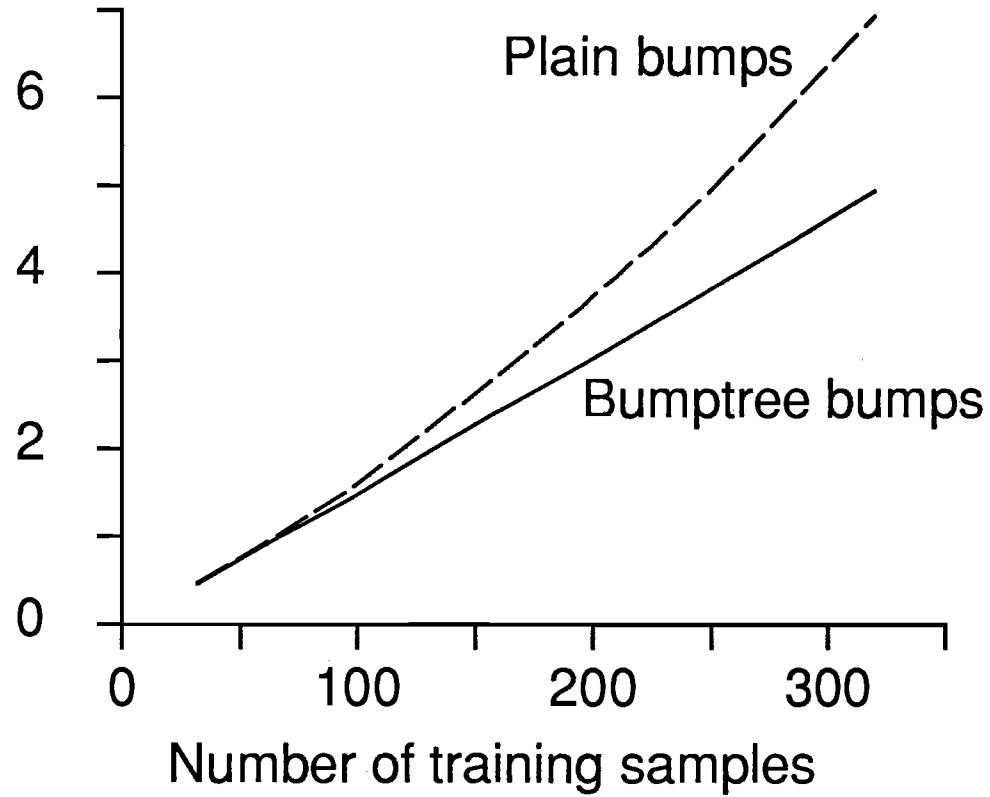


RBF vs. Bumptree Learning Time

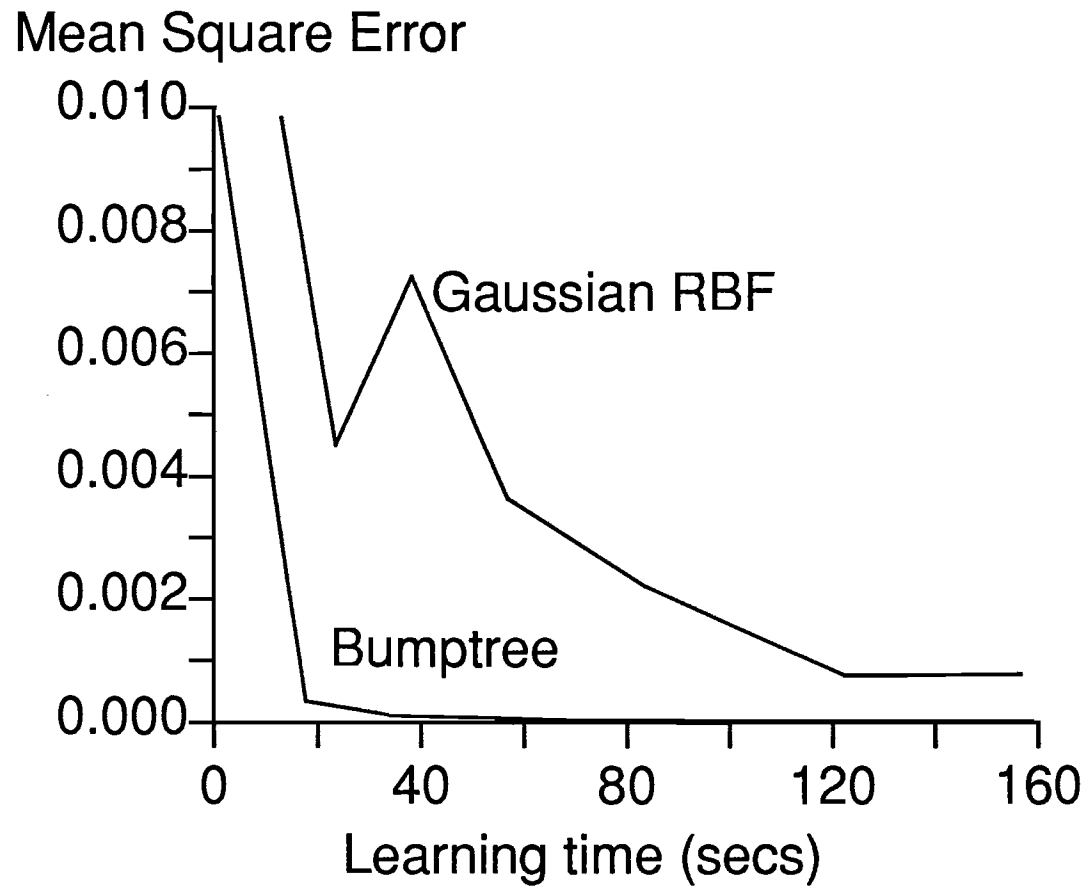


Bumptree Construction Time

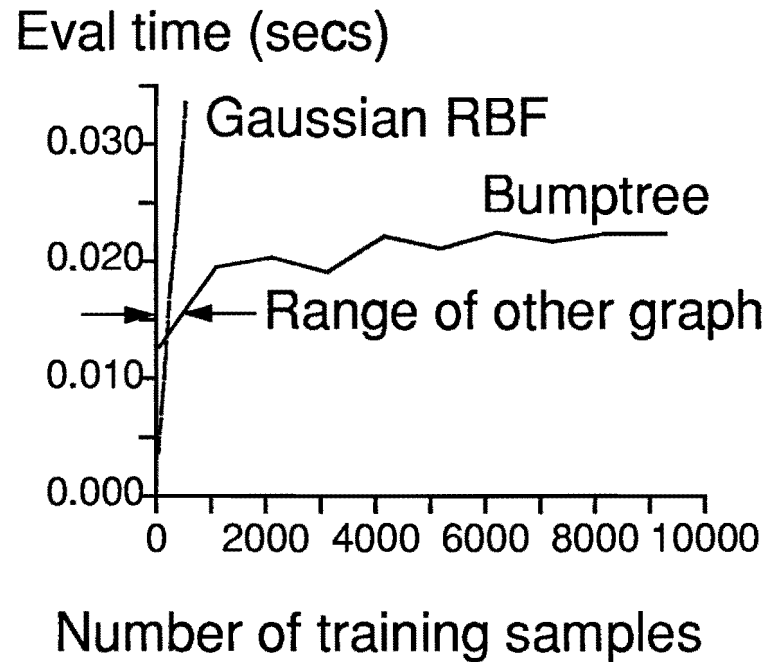
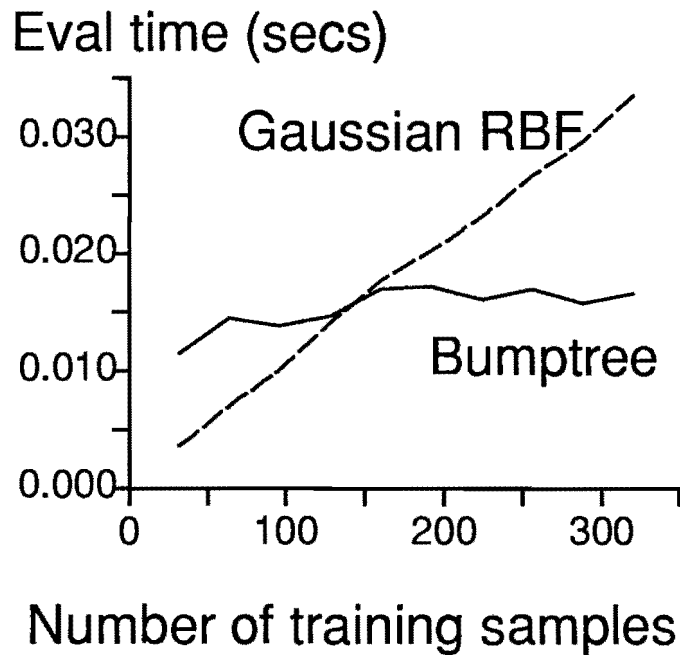
Learning time (secs)



Error vs. Learning Time

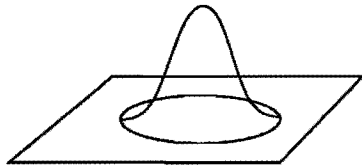


Function Evaluation Time

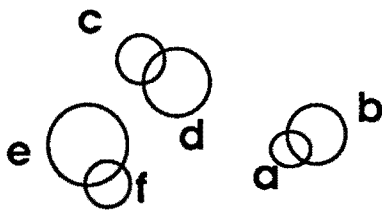


What is a Bumptree?

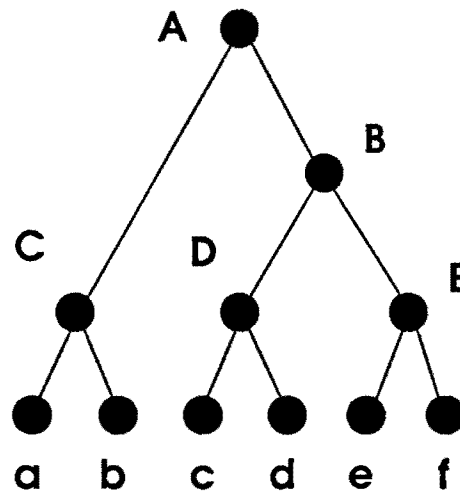
A *bumptree* is a complete binary tree with a real valued function associated to each node such that an interior node's function is everywhere larger than the functions associated with the leaves beneath it.



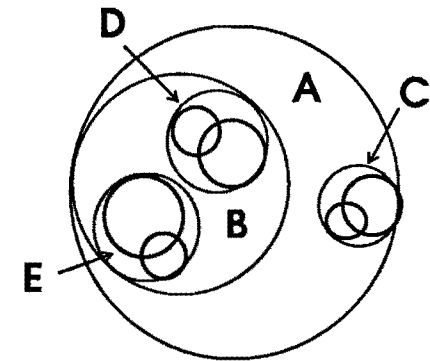
eg. ball supported bump



2-d leaf functions



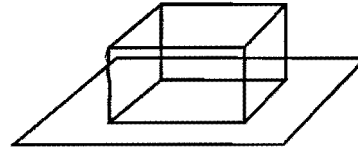
tree structure



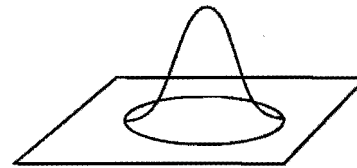
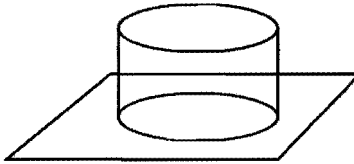
tree functions

Some bump tree functions

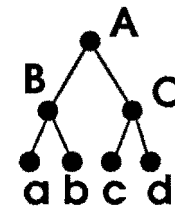
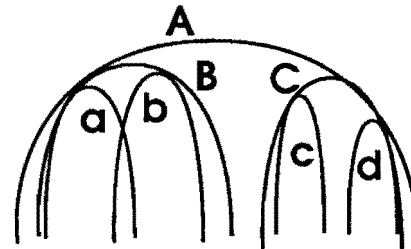
- *kd-trees, oct-trees, boxtrees*
1 in rectangle, 0 outside:



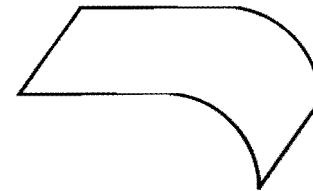
- *balltrees*
1 in ball, 0 outside:
or smooth bumps in ball, 0 outside:



- *Gaussian mixtures*
Really store quadratic form
in exponent:



For fast access, parent functions are quadratic function of distance from a coordinate partition on one side, 0 on other:



Bumptrees Queries

Pruning:

- Return leaf functions which are larger than a given value on query point

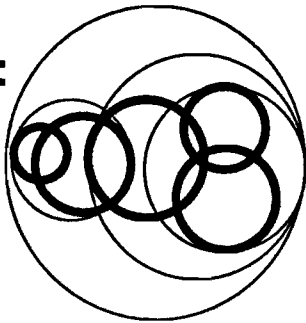
Branch and bound queries:

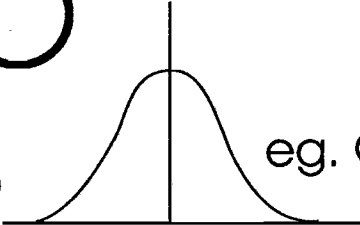
- Return leaf functions which are within a given factor or constant of the largest leaf function value

Can find nearest neighbors in log expected time over a wide variety of probability distributions. In practice, works well up to about 10 input dimensions for a few thousand uniformly distributed data points. Structure in the distribution can give speedups in arbitrarily high dimensions.

Mapping Learning with Bumptrees

Leaf Bumps: 

Bumptree: 

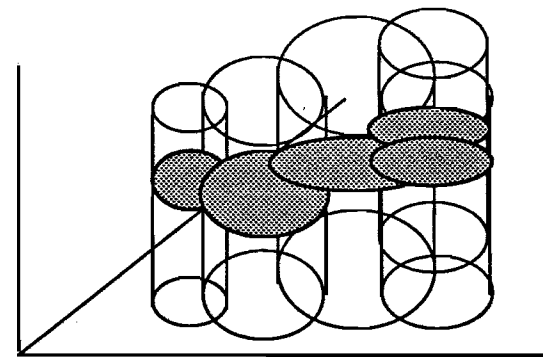
Influence bumps: $b_i(x)$  eg. Gaussians

Partition of Unity:
$$n_i(x) = \frac{b_i(x)}{\sum_j b_j(x)}$$

Local Models: $m_i(x)$ eg. affine models

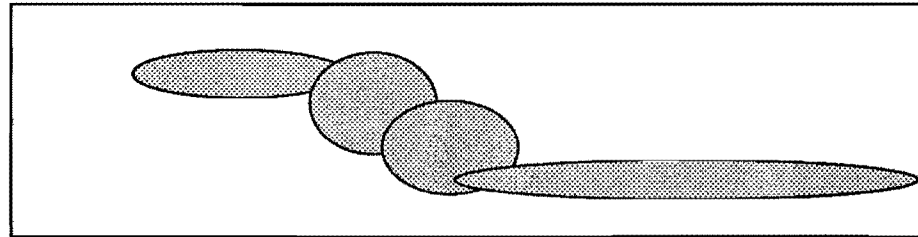
Smoothly interpolate the models:

$$f(x) = \sum_i n_i(x) m_i(x)$$



Bumptrees for Density Estimation, Classification and Constraint Learning

Use bumptree to hold normal mixture.



Density estimation by adaptive kernel estimator (with merging).

Classification by Bayes' decision rule on density.

Constraint queries such as partial match:

(the product of Gaussians is Gaussian, so normal mixtures may be composed to form normal mixtures)

