Active Clustering and Ranking

Rob Nowak, University of Wisconsin-Madison

IMA Workshop on "High-Dimensional Phenomena" (9/26-30, 2011)
Clustering Using Pairwise Similarities

**Clustering Problem:** Consider a set of $n$ objects $x_1, \ldots, x_n$. To cluster the objects into meaningful groups, we can ask questions of the form “how similar is $x_i$ to $x_j$?” for $i, j \in \{1, \ldots, n\}$. The goal is to cluster the objects by asking as few questions as possible.

**Metric Data:** $s_{ij} \propto \frac{1}{\text{dist}(i, j)}$

Standard approaches (k-means, spectral clustering, etc) assume the full set of $\binom{n}{2}$ similarities are known, but in high-dimensional settings ($n$ large) similarities may be missing and/or costly to collect/compute.
Clustering Using Pairwise Similarities

**Clustering Problem:** Consider a set of \( n \) objects \( x_1, \ldots, x_n \). To cluster the objects into meaningful groups, we can ask questions of the form “how similar is \( x_i \) to \( x_j \)” for \( i, j \in \{1, \ldots, n\} \). The goal is to cluster the objects by asking as few questions as possible.

![Diagram of clustering](image)

**Metric Data:** 

\[ s_{ij} \propto \frac{1}{\text{dist}(i, j)} \]

Standard approaches (k-means, spectral clustering, etc) assume the full set of \( \binom{n}{2} \) similarities are known, but in high-dimensional settings (\( n \) large) similarities may be missing and/or costly to collect/compute.

However, the existence of clusters can induce redundancy into the similarities, and therefore it may be possible to robustly cluster based on a small subset.
**Ranking Based on Pairwise Comparisons**

**Ranking Problem:** Consider a set of \( n \) objects \( x_1, \ldots, x_n \in \mathbb{R}^d \). The locations of \( x, \ldots, x_{n-1} \) are known, but location of \( x_n \) is unknown. To gather information about \( x_n \), we can only ask questions of the form “is object \( x_n \) closer to \( x_i \) than \( x_j \)?” The goal is to rank \( x_1, \ldots, x_{n-1} \) relative to distances to \( x_n \) by asking as few questions as possible.

**Ordinal Data:** \( 1(\delta_i < \delta_j) \)

Standard ranking methods assume the full set of \( \binom{n}{2} \) comparisons are known, but in high-dimensional settings (\( n \) large) comparisons may be missing and/or costly to collect.
Ranking Based on Pairwise Comparisons

**Ranking Problem:** Consider a set of $n$ objects $x_1, \ldots, x_n \in \mathbb{R}^d$. The locations of $x, \ldots, x_{n-1}$ are known, but location of $x_n$ is unknown. To gather information about $x_n$, we can only ask questions of the form “is object $x_n$ closer to $x_i$ than $x_j$?” The goal is to rank $x_1, \ldots, x_{n-1}$ relative to distances to $x_n$ by asking as few questions as possible.

Standard ranking methods assume the full set of $\binom{n}{2}$ comparisons are known, but in high-dimensional settings ($n$ large) comparisons may be missing and/or costly to collect.

However, many comparisons are redundant because the objects embed in $\mathbb{R}^d$, and therefore it may be possible to correctly rank based on a small subset.
Cost of Obtaining Similarities

**Cost of Obtaining Similarities**

- **genes**
  - Similarity = Pearson correlation of expressions
  - Need to test several conditions/drugs ~ 10000 x 10000!
  - **High cost of experimentation**

- **conditions/drugs**
  - Similarity = Pearson correlation of expressions
  - Need to test several conditions/drugs ~ 10000 x 10000!
  - **High cost of experimentation**

- **identify Internet topology**
  - Similarity = covariance of delay measurements
  - Need to send $O(N^2)$ packets ~ 10000^2!
  - **Significant burden on network resources**

- **identify co-regulated genes**
  - Similarity = Pearson correlation of expressions
  - Need to test several conditions/drugs ~ 10000 x 10000!

- **phylogenetics**
  - Similarity = genome sequence alignments
  - Need to align 10000 sequences of length 10000!
  - **Computationally expensive to compare sequences**

- **compute distance between sonar**
  - Excessive burden to human experts
Hierarchical Clustering

Given N objects and pairwise similarities between them, arrange the objects into a hierarchy of clusters

**Sample Complexity:** Minimum number of similarities needed to cluster?
Motivation: Internet Topology Inference

Correlation between traffic patterns at two points can indicate the similarity between nodes (e.g., number of shared links in paths)
Passive vs. Active Clustering

Generate a hierarchical clustering of the objects

**Passive Clustering:** given all or a random subset of pre-selected pairwise similarities.

**Active Clustering:** by sequentially selecting which pairwise similarities to measure in an adaptive fashion.
Passive vs. Active Clustering

Generate a hierarchical clustering of the objects

**Passive Clustering:** given all or a random subset of pre-selected pairwise similarities.

**Active Clustering:** by sequentially selecting which pairwise similarities to measure in an adaptive fashion.

Example:
Passive vs. Active Clustering

Generate a hierarchical clustering of the objects

**Passive Clustering**: given all or a random subset of pre-selected pairwise similarities.

**Active Clustering**: by sequentially selecting which pairwise similarities to measure in an adaptive fashion.

Example:
Passive vs. Active Clustering

Generate a hierarchical clustering of the objects

**Passive Clustering:** given all or a random subset of pre-selected pairwise similarities.

**Active Clustering:** by sequentially selecting which pairwise similarities to measure in an adaptive fashion.

Example:
Passive vs. Active Clustering

Generate a hierarchical clustering of the objects

**Passive Clustering:** given all or a random subset of pre-selected pairwise similarities.

**Active Clustering:** by sequentially selecting which pairwise similarities to measure in an adaptive fashion.

Example:
Cost of Passive Clustering

To cluster $N$ objects, most clustering methods require all $\frac{N(N-1)}{2}$ pairwise similarities.
Cost of Passive Clustering

To cluster $N$ objects, most clustering methods require all $\frac{N(N-1)}{2}$ pairwise similarities.

Can random sub-sampling reduce this burden?

**Theorem 1:** To recover clusters of size $m$, any clustering procedure based on random sub-sampling requires at least $\frac{N(N-1)}{m}$ similarities.

Small clusters are hard to find using randomly selected similarities.
Cost of Passive Clustering

To cluster $N$ objects, most clustering methods require all $\frac{N(N-1)}{2}$ pairwise similarities.

Can random sub-sampling reduce this burden?

**Theorem 1:** To recover clusters of size $m$, any clustering procedure based on random sub-sampling requires at least $\frac{N(N-1)}{m}$ similarities.

Small clusters are hard to find using randomly selected similarities.

**Related fact:** Clusters of size $O(N)$ can be recovered from random samples of $O(N \log N)$ similarities (Balcan-Gupta ’10, Shamir-Tishby ’11)

Any method needs at least $N$ similarities (at least one per object)
Hierarchical Clustering

Given N objects and pairwise similarities between them, arrange the objects into a hierarchy of clusters (aka “cluster tree”)

Identifiability: Is there an unambiguous way to hierarchically cluster?
Hierarchically Consistent Similarities

Do the similarities conform to the hierarchical clustering structure?
Hierarchically Consistent Similarities

Do the similarities conform to the hierarchical clustering structure?

**Consistency Assumption**: Similarities are consistent with a hierarchical clustering if for all

\[
\forall i \in C, \quad j, k \in C \quad s_{j,k} > \max (s_{i,j}, s_{i,k})
\]

\(i \not\in C\)

\(j, k \in C\)
Hierarchically Consistent Similarities

Do the similarities conform to the hierarchical clustering structure?

**Consistency Assumption**: Similarities are consistent with a hierarchical clustering if for all

\[
\text{outlier}(i, j, k) = \begin{cases} 
    i : \max(s_{ij}, s_{ik}) < s_{jk} \\
    j : \max(s_{ij}, s_{jk}) < s_{ik} \\
    k : \max(s_{ik}, s_{jk}) < s_{ij}
\end{cases}
\]

"which two of the three are most similar"
Active Clustering by Outlier Testing

Methodology example:

Initial Tree Structure
Active Clustering by Outlier Testing

Methodology example:
Active Clustering by Outlier Testing

Methodology example:

True Placement in Tree

New object
Active Clustering by Outlier Testing

Methodology example:

Find interior node (v) such that tree is roughly divided in two.
Active Clustering by Outlier Testing

Methodology example:

Choose two objects whose nearest common ancestor is $v$. 
Active Clustering by Outlier Testing

Methodology example:

Find the “outlier” of the three objects.
Active Clustering by Outlier Testing

Methodology example:

Find the “outlier” of the three objects.

This is found using 3 pairwise similarities:

$$\text{outlier}(i, j, k) = \begin{cases} 
  i : & \max(s_{i,j}, s_{i,k}) < s_{j,k} \\
  j : & \max(s_{i,j}, s_{j,k}) < s_{i,k} \\
  k : & \max(s_{i,k}, s_{j,k}) < s_{i,j}
\end{cases}$$
Active Clustering by Outlier Testing

Methodology example:

This resolves which subtree the new object is in.
Active Clustering by Outlier Testing

Methodology example:

This process is repeated...
Active Clustering by Outlier Testing

Methodology example:

Until the true location for the new object is found.
Active Clustering: Efficient Hierarchical Clustering

A **binary search** procedure based on

\[
\text{outlier}(i, j, k) = \begin{cases} 
  i : \max(s_{ij}, s_{ik}) < s_{jk} \\
  j : \max(s_{ij}, s_{jk}) < s_{ik} \\
  k : \max(s_{ik}, s_{jk}) < s_{ij}
\end{cases}
\]

Each object can be placed using \(\log N\) outlier tests \(\times 3\) similarities/test
Active Clustering: Efficient Hierarchical Clustering

A **binary search** procedure based on

\[
\text{outlier}(i, j, k) = \begin{cases} 
  i : \max(s_{ij}, s_{ik}) < s_{jk} \\
  j : \max(s_{ij}, s_{jk}) < s_{ik} \\
  k : \max(s_{ik}, s_{jk}) < s_{ij}
\end{cases}
\]

Each object can be placed using \(\log N\) outlier tests \(\times 3\) similarities/test

**Theorem 2:** Under **consistency assumption**, hierarchical clustering of \(N\) objects can be recovered using at most \(3N \log N\) sequentially selected similarities.

within a constant factor of the *information theoretic lower bound*
Active Clustering: Efficient Hierarchical Clustering

A **binary search** procedure based on

$$\text{outlier}(i, j, k) = \begin{cases} i : \max(s_{ij}, s_{ik}) < s_{jk} \\ j : \max(s_{ij}, s_{jk}) < s_{ik} \\ k : \max(s_{ik}, s_{jk}) < s_{ij} \end{cases}$$

Each object can be placed using

$$\log N \text{ outlier tests } \times 3 \text{ similarities/test}$$

**Theorem 2:** Under consistency assumption, hierarchical clustering of $N$ objects can be recovered using at most $3N \log N$ sequentially selected similarities.

within a constant factor of the information theoretic lower bound

Experimental Results with Synthetic Internet Topology

<table>
<thead>
<tr>
<th>Number of Leaf Nodes</th>
<th>Agglomerative Similarities Required</th>
<th>Outlier Similarities Required</th>
<th>Savings Over Exhaustive Approaches</th>
</tr>
</thead>
<tbody>
<tr>
<td>N=768</td>
<td>294,528</td>
<td>8,490</td>
<td>97.12%</td>
</tr>
</tbody>
</table>
Sensitivity to Errors and Inconsistencies

The previous technique is very sensitive to noise and inconsistencies:

\[ S(\circ,\circ) > \max \{ S(\circ,\circ), S(\circ,\circ) \} \]

which can lead to cascading failures.
Robust Active Clustering

To overcome this fragility, we design a top-down recursive splitting approach and use voting to boost our confidence about each decision we make.
Robust Active Clustering

To overcome this fragility, we design a top-down recursive splitting approach and use voting to boost our confidence about each decision we make.

**Goal**: In each step, split a single cluster into 2 sub-clusters, efficiently and accurately.
Cluster Splitting Procedure
Cluster Splitting Procedure

seed
Cluster Splitting Procedure

m “test” pairs chosen uniformly at random

seed
Cluster Splitting Procedure

If ‘yes’, then \( \text{red dot} \) and \( \text{blue dot} \) are probably clustered together
Why? Consider outlier tests w/o errors
Why? Consider outlier tests w/o errors

\[
\text{outlier}(\bullet, \circ, \bullet) = \text{outlier}(\bullet, \circ, \bullet) = \bullet
\]
Why? Consider outlier tests w/o errors

\[
\text{outlier}(\,\bullet\, ,\bullet\, ,\bullet\,) = \text{outlier}(\,\bullet\, ,\bullet\, ,\bullet\,) = \bullet
\]

\[
\text{outlier}(\,\bullet\, ,\bullet\, ,\bullet\,) = \text{outlier}(\,\bullet\, ,\bullet\, ,\bullet\,) = \circ
\]
Why? Consider outlier tests w/o errors

\[
\text{outlier}(\circ, \circ, \bullet) = \text{outlier}(\bullet, \circ, \bullet) = \bullet
\]

\[
\text{outlier}(\circ, \circ, \bullet) = \text{outlier}(\bullet, \circ, \bullet) = \circ
\]

\[
\text{outlier}(\circ, \circ, \bullet) = ? \quad \text{outlier}(\bullet, \circ, \bullet) = ?
\]
Why? Consider outlier tests w/o errors

\[
\text{outlier}( \bullet, \circ, \bullet ) = \text{outlier}( \bullet, \circ, \bullet ) = \bullet \\
\text{outlier}( \bullet, \circ, \bullet ) = \text{outlier}( \bullet, \circ, \bullet ) = \circ \\
\text{outlier}( \bullet, \circ, \bullet ) = ? \quad \text{outlier}( \bullet, \circ, \bullet ) = ? \\
\text{outlier}( \bullet, \circ, \bullet ) = \circ \quad \text{outlier}( \bullet, \circ, \bullet ) = \bullet
\]
Why? Consider outlier tests w/o errors

\[
\text{outlier}(\bullet, \circ, \circ) = \text{outlier}(\bullet, \circ, \circ) = \bullet
\]

\[
\text{outlier}(\bullet, \circ, \circ) = \text{outlier}(\bullet, \circ, \circ) = \circ
\]

\[
\text{outlier}(\bullet, \circ, \circ) = ? \quad \text{outlier}(\bullet, \circ, \circ) = ?
\]

\[
\text{outlier}(\bullet, \circ, \circ) = \circ \quad \text{outlier}(\bullet, \circ, \circ) = \circ
\]

\[
\text{outlier}(\circ, \circ, \circ) = \circ \quad \text{outlier}(\bullet, \circ, \circ) = \circ
\]

\[
\text{outlier}(\bullet, \circ, \circ) = \circ \quad \text{outlier}(\bullet, \circ, \circ) = \circ
\]

\[
\text{outlier}(\bullet, \circ, \circ) = ? \quad \text{outlier}(\bullet, \circ, \circ) = ?
\]

\[
\text{outlier}(\bullet, \circ, \circ) = \circ \quad \text{outlier}(\bullet, \circ, \circ) = ?
\]
Why? Consider outlier tests w/o errors

outlier(●, ○, ●) = outlier(○, ○, ●) = ●

outlier(●, ○, ●) = outlier(○, ○, ●) = ○

outlier(●, ○, ●) = ? outlier(○, ○, ●) = ?

outlier(●, ○, ●) = ○ outlier(○, ○, ●) = ●

outlier(●, ○, ●) = ● outlier(○, ○, ●) = ○

outlier(●, ○, ●) = ○ outlier(○, ○, ●) = ●

outlier(●, ○, ●) = ? outlier(○, ○, ●) = ●

outlier(●, ○, ●) = ○ outlier(○, ○, ●) = ?
Why? Consider outlier tests w/o errors

outlier(●,○,●) = outlier(○,○,●) = ●

outlier(●,○,●) = outlier(●,○,●) = ○

outlier(●,○,●) = ? outlier(●,○,●) = ?

outlier(●,○,●) = ○ outlier(●,○,●) = ●

impossible if ● and ○ are not clustered together
Clustered vs. Non-Clustered

\[
\text{outlier}(\bullet, \circ, \bullet) = \text{outlier}(\bullet, \circ, \bullet) = \bullet
\]

\[
\text{outlier}(\bullet, \circ, \bullet) = \text{outlier}(\bullet, \circ, \bullet) = \circ
\]

\[
\text{outlier}(\bullet, \circ, \bullet) = \circ \text{ ?} \quad \text{outlier}(\bullet, \circ, \bullet) = \circ \text{ ?}
\]

\[
\text{outlier}(\bullet, \circ, \bullet) = \bullet \quad \text{outlier}(\bullet, \circ, \bullet) = \bullet
\]
If outlier tests are perfectly correct, then

\[ P(A \mid \bullet, \bullet \text{ clustered}) = 2 \eta (1 - \eta) , \text{ where } \eta = \text{fractional size of seed cluster} \]

\[ P(A \mid \bullet, \bullet \text{ not clustered}) = 0 \]

\[ \text{outlier( } \bullet, \circ, \bullet ) = \text{outlier( } \bullet, \circ, \bullet ) = \bullet \]

\[ \text{outlier( } \bullet, \circ, \bullet ) = \text{outlier( } \bullet, \circ, \bullet ) = \circ \]

\[ \text{outlier( } \bullet, \circ, \bullet ) = ? \text{ outlier( } \bullet, \circ, \bullet ) = ? \]

\[ \text{outlier( } \bullet, \circ, \bullet ) = \circ \text{ outlier( } \bullet, \circ, \bullet ) = \circ \]
If outlier tests are perfectly correct, then

\[ P(A | \bullet, \bullet \text{ clustered} ) = 2 \eta (1 - \eta) , \quad \text{where } \eta = \text{fractional size of seed cluster} \]

\[ P(A | \bullet, \bullet \text{ not clustered} ) = 0 \]

If outlier tests are incorrect with probability \( \leq p < 1/2 \), then

\[ P(A | \bullet, \bullet \text{ clustered} ) \geq 2 (1 - p) \eta (1 - \eta) \]

\[ P(A | \bullet, \bullet \text{ not clustered} ) \leq p \]
Theorem 3: If outlier tests are inconsistent with probability \( p \leq \frac{2\eta(1-\eta)}{1-2\eta(1-\eta)} \), then we correctly hierarchically cluster \( N \) objects (into clusters of size \( \geq \log N \)) using \( O(N \log^2 N) \) sequentially and adaptively selected pairwise similarities.

Extra \( \log n \) factor due to voting over test pairs
Robust Active Clustering Experiments

Synthetic experiments –
balanced binary tree with $N = 512$ objects

<table>
<thead>
<tr>
<th>probability of inconsistency</th>
<th>Resolvability (smallest correct cluster size - smaller is better)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Agglomerative Clustering</td>
</tr>
<tr>
<td>5%</td>
<td>460.8</td>
</tr>
<tr>
<td>15%</td>
<td>512</td>
</tr>
<tr>
<td>25%</td>
<td>512</td>
</tr>
</tbody>
</table>
Robust Active Clustering Experiments

Gene microarray experiments, N=512 genes

Bottom-up agglomerative clustering

Top-down active clustering (43% of the pairwise similarities)
Summary

- randomly selected similarities
  (where m is the smallest cluster size)
  \[ \geq \frac{N}{m} (N - 1) \]

- adaptively selected similarities
  - consistent similarities
    all clusters of any size
  \[ \leq 3N \log N \]
  - inconsistent similarities
    all clusters of size > \log N
  \[ O(N \log^2 N) \]

Active Clustering: Robust and Efficient Hierarchical Clustering using Adaptively Selected Similarities, Artificial Intelligence and Statistics, AISTATS’11

http://arxiv.org/abs/1102.3887
Ranking Based on Pairwise Comparisons

**Ranking Problem:** Consider a set of $n$ objects $x_1, \ldots, x_n \in \mathbb{R}^d$. The locations of $x, \ldots, x_{n-1}$ are known, but location of $x_n$ is unknown. To gather information about $x_n$, we can only ask questions of the form “is object $x_n$ closer to $x_i$ than $x_j$?” The goal is to rank $x_1, \ldots, x_{n-1}$ relative to distances to $x_n$ by asking as few questions as possible.

Standard ranking methods assume the full set of $\binom{n}{2}$ comparisons are known, but in high-dimensional settings ($n$ large) comparisons may be missing and/or costly to collect.

However, many comparisons are redundant because the objects embed in $\mathbb{R}^d$, and therefore it may be possible to correctly rank based on a small subset.
Bartender: “What’s your favorite beer, sir?”
Bartender: “What’s your favorite beer, sir?”
AI: “Hmm... actually I’m more of wine-man”
Bartender: “What’s your favorite beer, sir?”
AI: “Hmm... actually I’m more of wine-man”
Bartender: “Try these two samples. Do you prefer A or B?”
Bartender: “What’s your favorite beer, sir?”
AI: “Hmm... actually I’m more of wine-man”
Bartender: “Try these two samples. Do you prefer A or B?”
AI: “B”
Bartender: “What’s your favorite beer, sir?”
AI: “Hmm... actually I’m more of wine-man”
Bartender: “Try these two samples. Do you prefer A or B?
AI: “B”
Bartender: “Ok try these two: C or D?” ....
Ranking Relative to Distance

Al’s latent preferences in “beer space” (e.g., hoppiness, lightness, sweetness,...)
Ranking Relative to Distance

C < A < B < E < G < D < F
Ranking Relative to Distance

D < G < C < E < A < B < F
Goal: Determine ranking by asking comparisons like, “Is \( r \) closer to \( A \) or \( B \)?”

Weakness of randomized schemes:
If comparisons are selected at random, then almost all \( \binom{n}{2} \) comparisons are needed to rank.

\[ D < G < C < E < A < B < F \]
Ranking with Adaptively Selected Queries

Insert H into: D < G < C < E < A < B < F
Insert H into: $D < G < C < E < A < B < F$

$\{\}$

$\{H < E\}$
Ranking with Adaptively Selected Queries

Insert H into: $D < G < C < E < A < B < F$

- $D \bullet G \bullet C \bullet E \bullet A \bullet B \bullet F$
  - Insert H into: $\{H < E\}$
- $D \bullet G \bullet C \bullet E \bullet A \bullet B \bullet F$
  - Insert H into: $\{H < E\}, \{G < H\}$
Insert $H$ into: \[ D < G < C < E < A < B < F \]
Ranking with Adaptively Selected Queries

D < G < C < E < A < B < F

Log$_2$ $k$ comparisons to insert an item into a list of $k$ objects

$\Longrightarrow$ $n \log_2 n$ comparisons to rank $n$ objects
**Ranking with Adaptively Selected Queries**

Insert H into: \( D < G < C < E < A < B < F \)

\[
\log_2 k \text{ comparisons to insert an item into a list of } k \text{ objects}
\]

\[
\Rightarrow n \log_2 n \text{ comparisons to rank } n \text{ objects}
\]

... but does embedding dimension \( d \) affect the sample complexity?
Many comparisons are redundant because the objects embed in $\mathbb{R}^d$, and therefore it may be possible to correctly rank based on a small subset.

binary information we can gather: $q_{i,j} \equiv x_n$ is closer to $x_i$ than $x_j$
Many comparisons are redundant because the objects embed in $\mathbb{R}^d$, and therefore it may be possible to correctly rank based on a small subset.

binary information we can gather: $q_{i,j} \equiv x_n$ is closer to $x_i$ than $x_j$

**Sequential Data Selection**

input: $x_1, \ldots, x_{n-1} \in \mathbb{R}^d$ and $x_n$ at unknown position in $\mathbb{R}^d$
initialize: $x_1, \ldots, x_{n-1}$ in uniformly random order

for $k=2, \ldots, n-1$
  
  for $i=1, \ldots, k-1$
    
    if $q_{i,k}$ is ambiguous given $\{q_{i,j}\}_{i,j<k}$,
    then ask for pairwise comparison,
    
    else impute $q_{i,j}$ from $\{q_{i,j}\}_{i,j<k}$

output: ranking of $x_1, \ldots, x_{n-1}$ consistent with all pairwise comparisons
Ranking and Geometry
Ranking and Geometry

**Definition:**
Answers to previous queries induces a Region of Ambiguity. Any query that intersects this region is said to be **Ambiguous.** Otherwise its **Unambiguous**
Ranking and Geometry
Ranking and Geometry

\[ \# \text{ of } d\text{-cells} \approx \frac{k^{2d}}{d!} \]
Ranking and Geometry

\[ \text{# of } d\text{-cells} \approx \frac{k^{2d}}{d!} \]
Ranking and Geometry

\# of \( d \)-cells \( \approx \frac{k^{2d}}{d!} \)

\# intersected \( \approx \frac{k^{2(d-1)}}{(d-1)!} \)
# of $d$-cells $\approx \frac{k^{2d}}{d!}$

# intersected $\approx \frac{k^{2(d-1)}}{(d-1)!}$

$\implies P(\text{ambiguous}) \approx \frac{d}{k^2}$
# of $d$-cells $\approx \frac{k^{2d}}{d!}$

# intersected $\approx \frac{k^{2(d-1)}}{(d-1)!}$

$\Rightarrow \mathbb{P}($ambiguous$) \approx \frac{d}{k^2}$

$\Rightarrow \mathbb{E}[\#\text{ambiguous}] \approx \frac{d}{k}$
Ranking and Geometry

\# of $d$-cells $\approx \frac{k^{2d}}{d!}$  \hspace{1cm} \text{(Coombs 1960)}

\# intersected $\approx \frac{k^{2(d-1)}}{(d-1)!}$  \hspace{1cm} \text{(Buck 1943)}

$\implies \mathbb{P}(\text{ambiguous}) \approx \frac{d}{k^2}$  \hspace{1cm} \text{(Cover 1965)}

$\implies \mathbb{E}[\# \text{ambiguous}] \approx \frac{d}{k}$

$\implies \mathbb{E}[\# \text{ requested}] \approx \sum_{k=2}^{n} \frac{d}{k}$  \hspace{1cm} \text{(Jamieson & Nowak 2011)}

$\approx d \log n$
Summary

# of comparisons needed to rank

- random selection \( O(n^2) \)
- sequential w/o geometry \( O(n \log n) \)
- exploiting geometry \( O(d \log n) \)
- noise-tolerant \( O(d \log^2 n) \)

Active Ranking using Pairwise Comparisons, *NIPS ’11*


code: [http://homepages.cae.wisc.edu/~jamieson/me/Active_Ranking.html](http://homepages.cae.wisc.edu/~jamieson/me/Active_Ranking.html)
Sonar Example

Sonar echo audio signals bounced off: \{50 targets, 50 rocks\}
\(S_{i,j} = \{\text{human-judged similarity between signals } i \text{ and } j\}\)

**Learning task:**
Leave one signal out of the set and rank the other 99 using comparisons: \(q_{i,j} \equiv \{S_{i,*} < S_{j,*}\}\)

Compute \(d\)-dim embedding using MDS with similarity matrix.
\(S_{i,*} < S_{j,*} \iff ||x_i - r|| < ||x_j - r||\)
because embedding is approximate

<table>
<thead>
<tr>
<th>Dimension</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>% of queries requested</td>
<td>14.5</td>
<td>18.5</td>
</tr>
<tr>
<td>Average error (d(y, \tilde{y}))</td>
<td>0.23</td>
<td>0.21</td>
</tr>
<tr>
<td>Kendall Tau</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average error (d(y, \hat{y}))</td>
<td>0.31</td>
<td>0.29</td>
</tr>
</tbody>
</table>

% of queries we requested
best achievable error
our algorithm’s error