Challenges in Multiresolution Methods for Graph-based Learning

Michael W. Mahoney

ICSI and Dept of Statistics, UC Berkeley

(For more info, see:
http://www.stat.berkeley.edu/~mmahoney
or Google on "Michael Mahoney")

Joint work with Ruoxi Wang and Eric Darve of Stanford.

December 2015

Outline

Outline

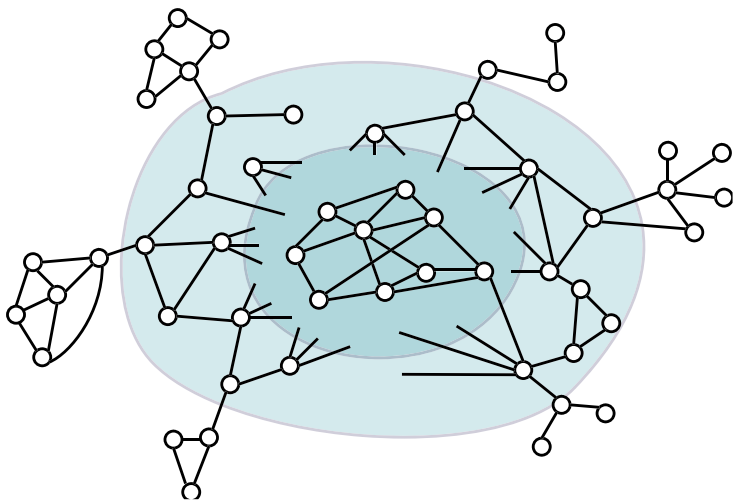
Networks and networked data

Lots of "networked" data!!

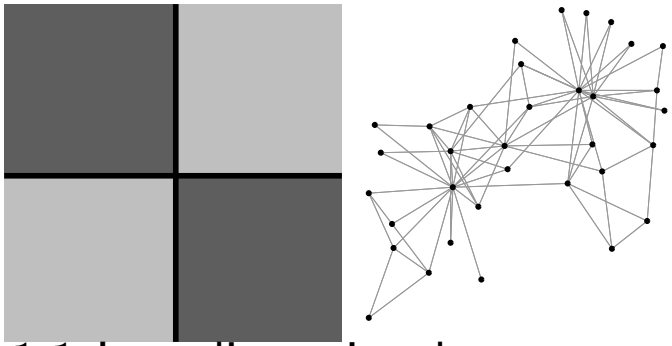
- technological networks (AS, power-grid, road networks)
- biological networks (food-web, protein networks)
- social networks (collaboration networks, friendships)
- information networks (co-citation, blog cross-postings, advertiser-bidded phrase graphs ...)
- language networks (semantic networks ...)
- ...

Interaction graph model of networks:

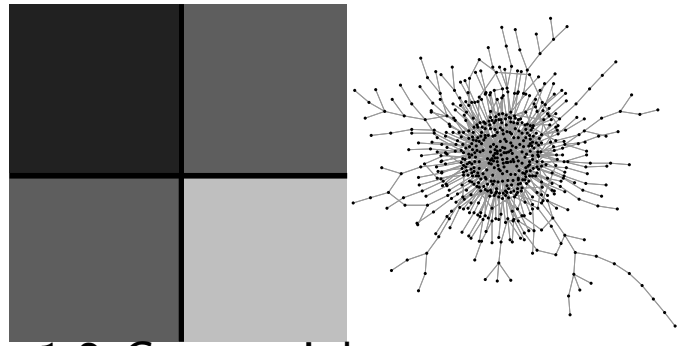
- Nodes represent "entities"
- Edges represent "interaction" between pairs of entities



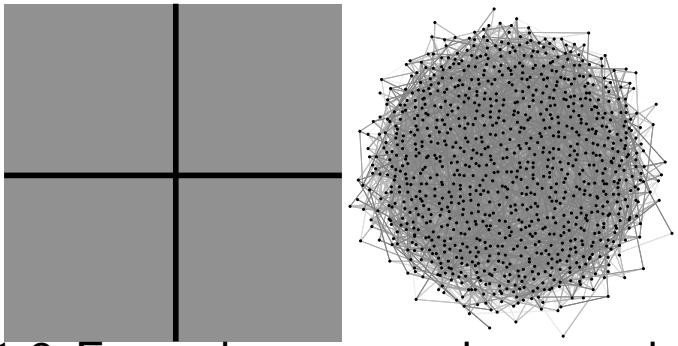
Possible ways a graph might look



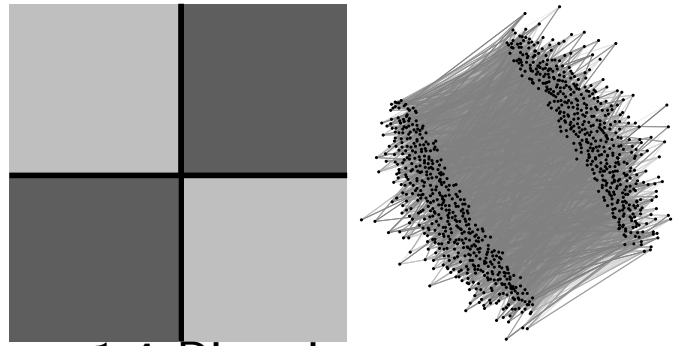
1.1 Low-dimensional structure



1.2 Core-periphery structure

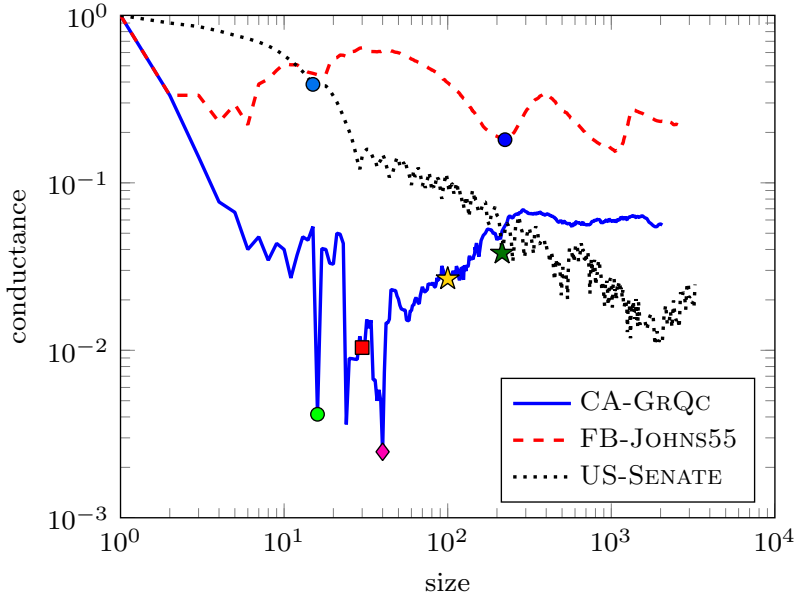


1.3 Expander or complete graph

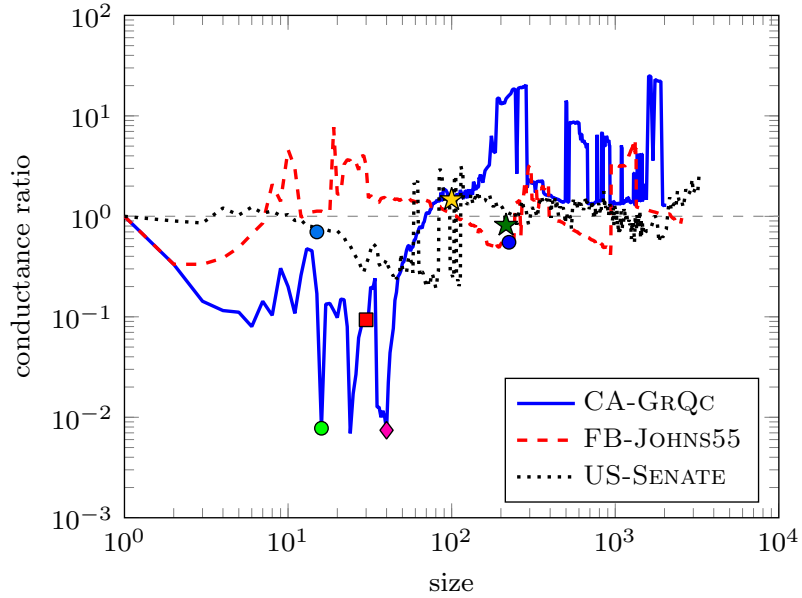


1.4 Bipartite structure

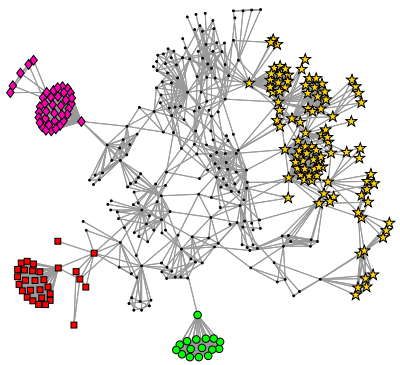
Three different types of real networks



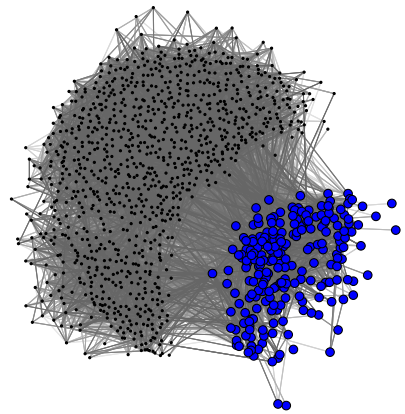
1.5 NCP: conductance value of best conductance set, as function of size



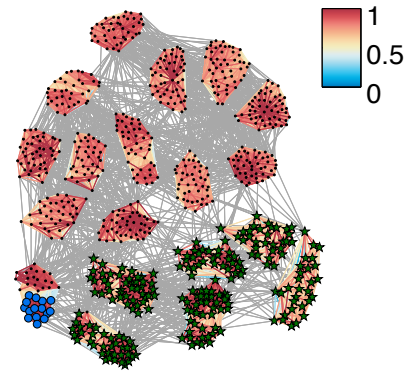
1.6 CRP: ratio of internal to external conductance, as function of size



1.7 CA-GrQc

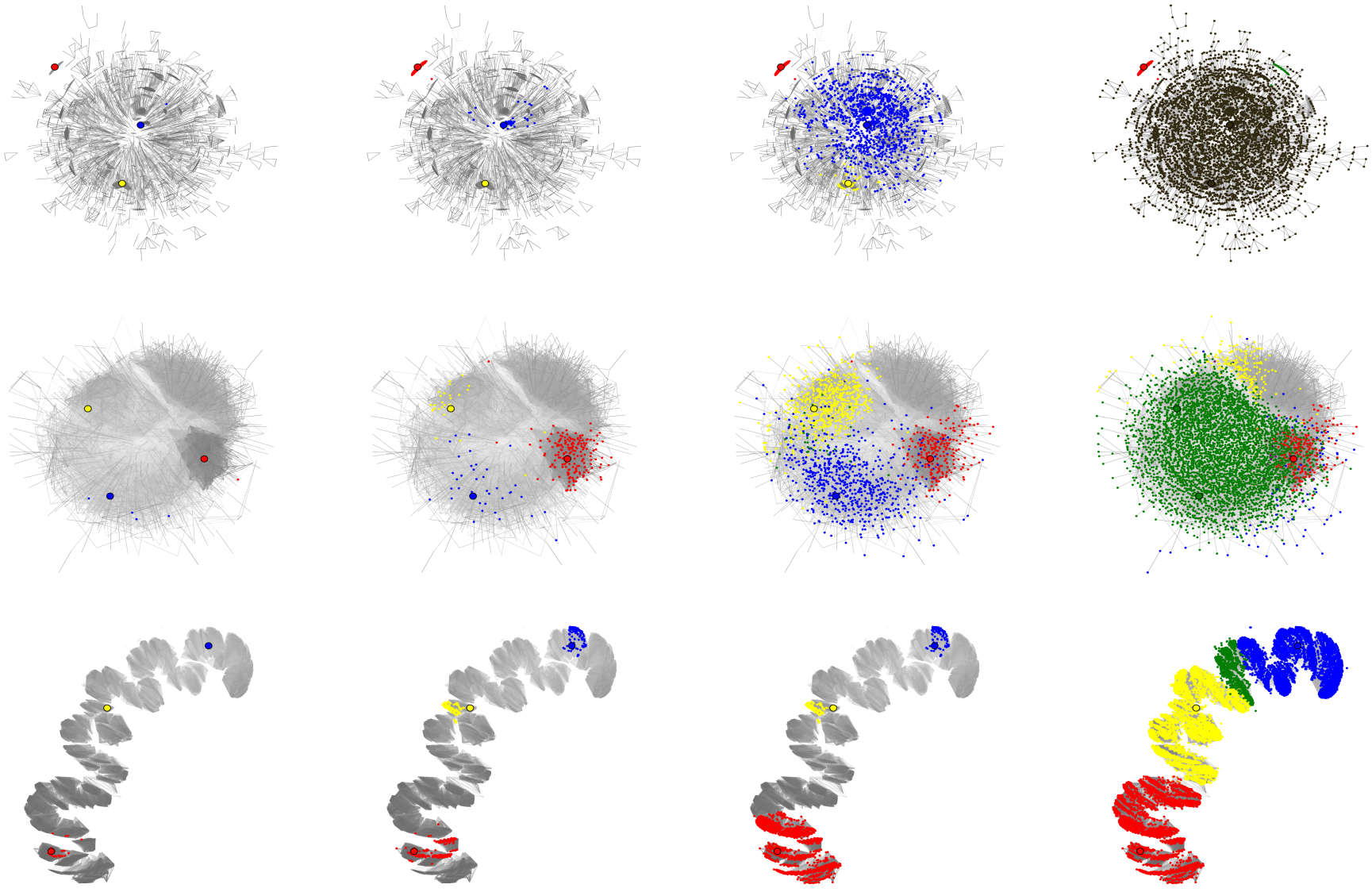


1.8 FB-Johns55



1.9 US-Senate

Information propagates local-to-glocal in different networks in different ways



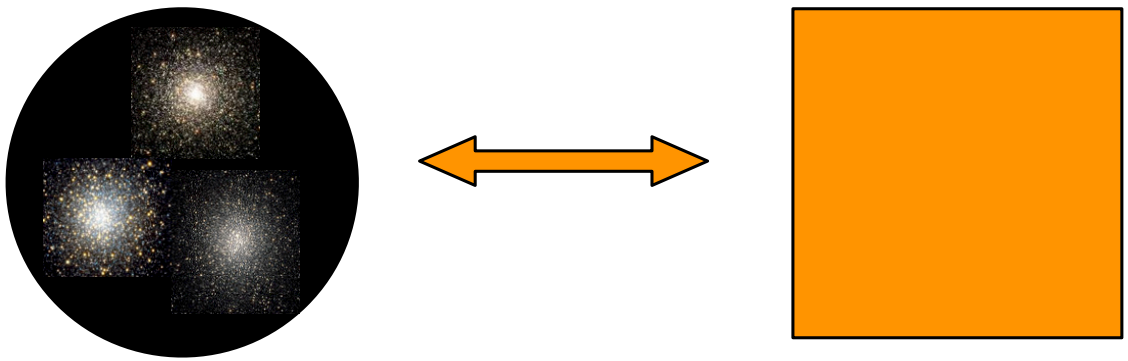
Obvious and non-obvious challenges

- Small-scale structure and large-scale noise
  - Ubiquitous property in realistic large social/information graphs
  - Problematic for algorithms, e.g., recursive partitioning
  - Problematic for statistics, e.g., control of inference
  - Problematic for qualitative insight, e.g. what data "look like"
- Are graphs constructed in ML any nicer
  - Yes, if they are small and idealized
  - Not much, in many cases, if they are large and non-toy
  - E.g., Lapacian-based manifold methods are very non-robust and overly homogenized in the presence of realistic noise
- Typical objective functions ML people like are very global
  - Sum over all nodes/points of a penalty
  - Acceptable to be wrong on small clusters
  - Cross-validate with "your favorite objective" to construct graphs leads to homogenized graphs

Outline

- Given an RBF kernel function $\mathcal{K} : \mathbb{R}^d \to \mathbb{R}$, and data $x_i \in \mathbb{R}^d$ $(i = 1 \ldots, n)$, what decides the rank of the kernel matrix K?

$$K_{ij} = \mathcal{K}(x_i, x_j)$$



Data Matrix

- bandwidth h $(\exp(-(r/h)^2))$, data distribution, cluster radius, number of points, etc.

There are two parts that people in different fields are interested:

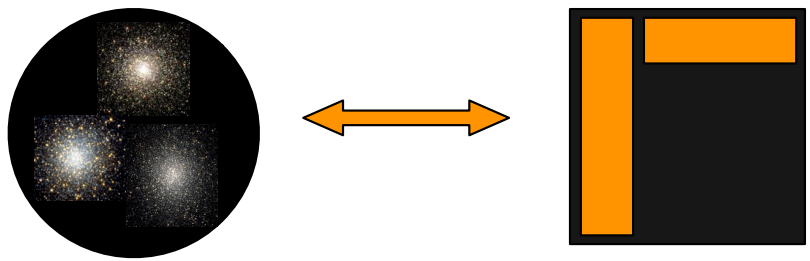
- Given the data and label / target: how to choose h for a more accurate model (machine learning people)
- Given the data and h: how to approximate the corresponding kernel matrix for a faster matrix-vector multiplication (linear algebra people)

Let's consider these two parts, and connect them by what approximation methods to use for different datasets (hence different h).

Outline

Solutions to matrix approximation

- Problem: given data and h, how to approximate the kernel matrix with minimal memory cost [1] while achieving high accuracy?
- Common solutions
  - low-rank matrices: low-rank methods

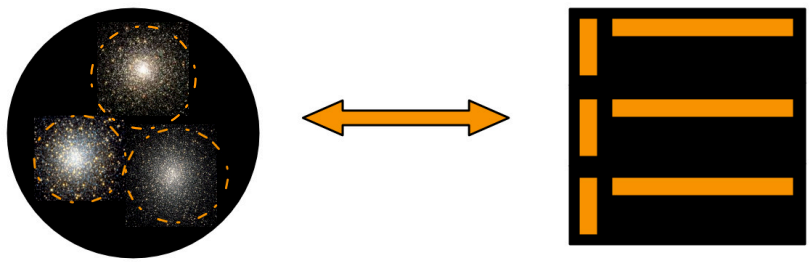


  - high-rank matrices from 2D/3D data: Fast Multipole Method (FMM), and other H matrix based methods.
- What about high dimensional data + high-rank (relative high)?

[1] memory cost is a close approximation of the running time for a matrix-vector multiplication.

Intuition of our solution

- Instead of considering global interaction (low-rank methods), let's consider local interaction.
- We cluster the data into distinct groups.



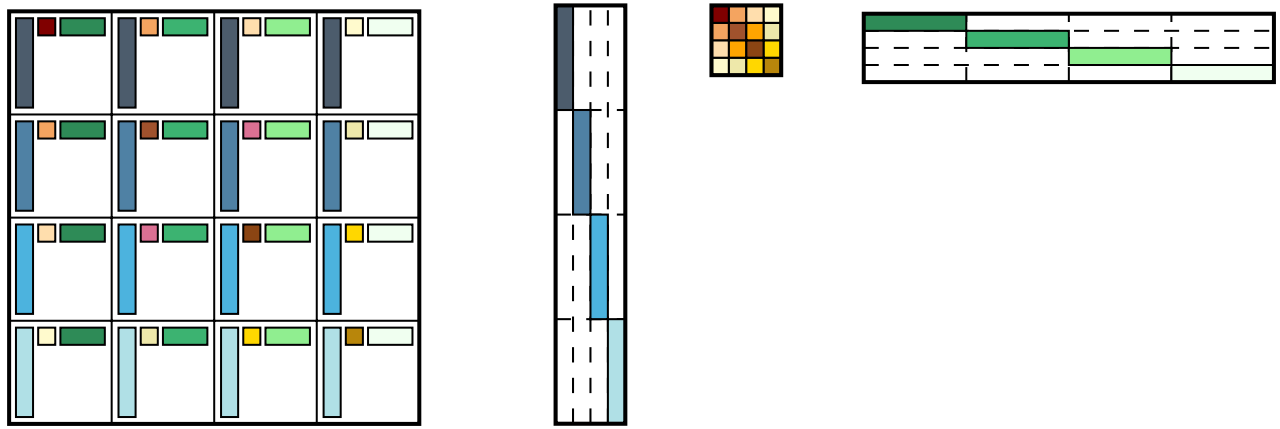
- If you have two clusters, the rank of the interaction matrix is related to the one with smaller radius. Therefore

$$\text{rank}(K(C_i, :)) \leq \text{rank}(K)$$

Block Basis Factorization (BBF)

- Given a matrix $M \in \mathbb{R}^{n \times n}$, partitioned into k by k blocks. Then the Block Basis Factorization (BBF) of M is defined as:

$M = \widetilde{U} \quad \widetilde{C} \quad \widetilde{V}^T$



	approximation	memory cost
BBF	special rank-(rk)	$\mathcal{O}(nr + (rk)^2)$
LR	rank-r	$\mathcal{O}(nr)$

- r is the rank used for each block.
- The factorization time of BBF is linear.
- BBF is a strict generalization of low-rank methods.

Structure advantage of BBF

- We show that BBF structure is a strict generalization of low-rank structure, regardless of the sampling method used.

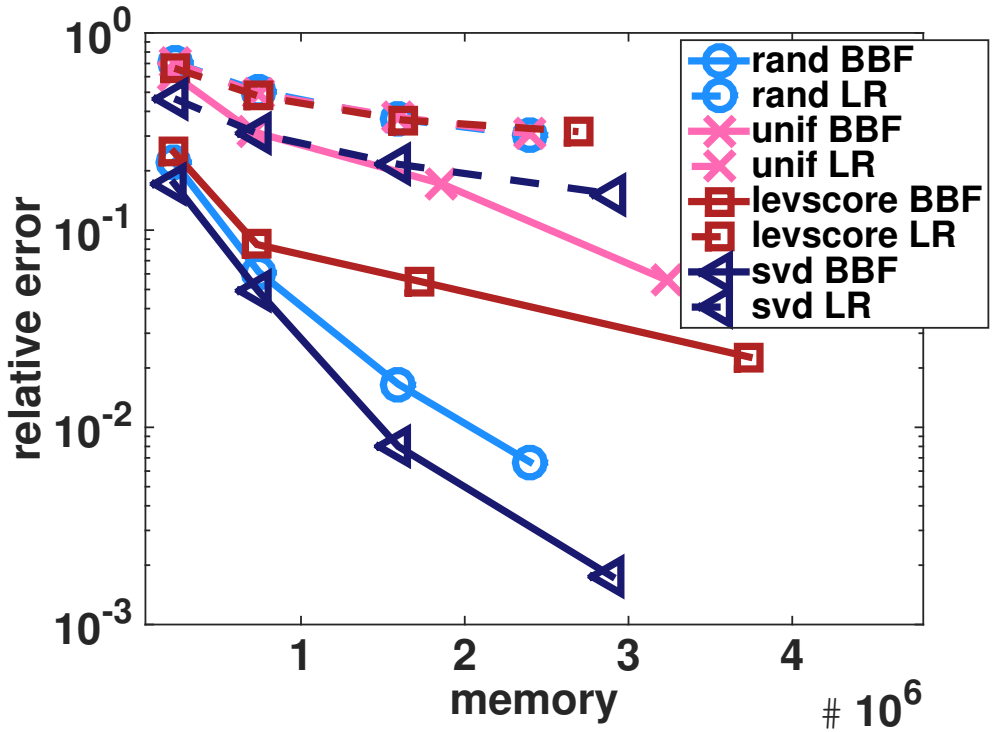


Figure: Sampled covertype data. Kernel approximation error vs memory for BBF and low-rank structure with different sampling methods. BBF (solid lines) means the BBF structure, and LR (dash lines) means the low-rank structure. Different symbols represent different sampling methods used in the schemes.

Outline

Intuition of kernel bandwidth and our interest

A general intuition for the role of h in kernel methods:

- A larger h:
  - consider local and far away points (smooth)
  - lead to a lower-rank matrix
- A smaller h:
  - consider local points (less smooth)
  - lead to a higher-rank matrix

A general idea of what values of h that we are interested in:

Less interesting:

- a very low-rank case: a mature low-rank method is more than enough.
- a very high-rank case: 1). kernel matrix becomes diagonal dominant, and 2). often results in overfitting of your model.

More interesting: the rank ranges in [low+, median]

Redefine the problem

Now let's consider the first part:

- **Problem: given data and label / targets, what h shall we choose?**

  This is often being done via cross-validation. But more than often, a large h is chosen, which usually leads to a low-rank matrix where a mature low-rank method is more than enough.

Let's consider this problem from a different angle:

- **Problem: what kind of data would prefer a relative small h?**

  Note here when we say h, we refer to the **largest h (denote here as h^*) that gives the optimal accuracy**, because a larger h usually results in low-rank matrix that is easy to approximate.

Main factor that h^* depends on

We consider the task of classification with kernel SVM in this talk. **What is the main property of data that h^* depends on?** We think it is the **least** radius of curvature of the correct decision boundary.

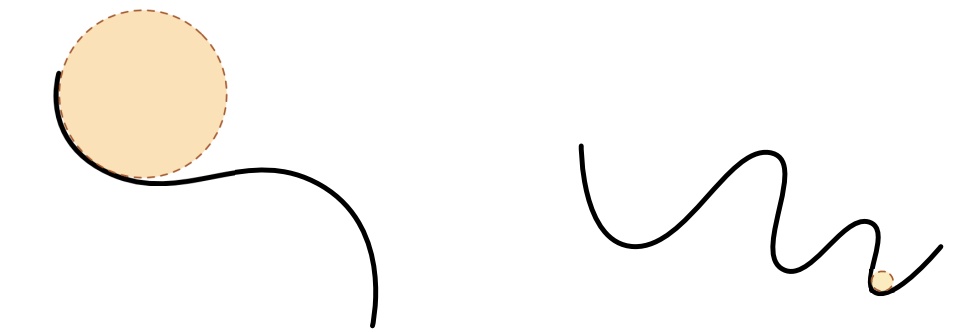


Figure: Left: smooth decision boundary; Right: curved decision boundary

The case on the left would prefer a larger h^*, while the case on the right would prefer a smaller h^*. (here h^* is the largest optimal h)

Conclusions from 2D synthetic data

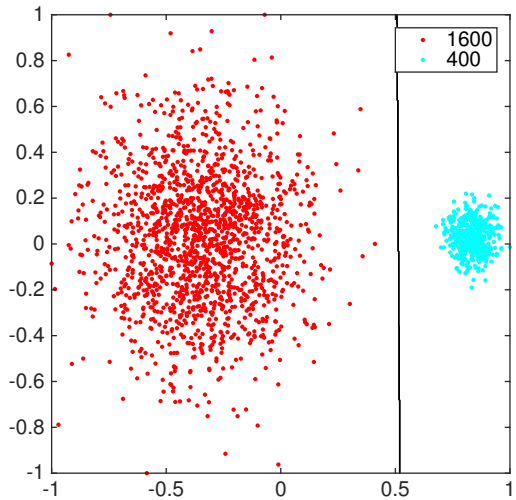
We first study the main dependent factor of h^* in a clean and neat setting: 2D synthetic dataset. Some main conclusions:

- The least radius of curvature for the correct decision boundary is indeed a main factor that h^* depends on.
- Other factors, *e.g.*, number of points in each cluster, radius of each cluster, do not *directly* affect h^*.
- When a small cluster is surrounded by a larger one, a smaller h is preferred to detect it.
- When two clusters are easy to separate, there will be a large range of optimal h's, and h^* will be very large.

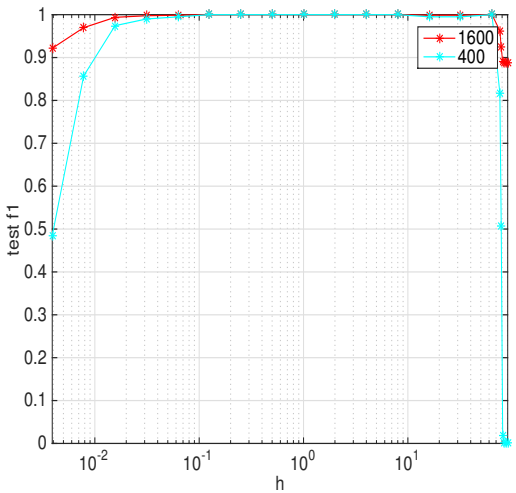
We hope this will shed some lights when we analyze real high dimensional datasets that are often complicated: each cluster can have a different sizes, shapes, densities, etc. And often combined with noises and outliers.

Two clusters easy to separate

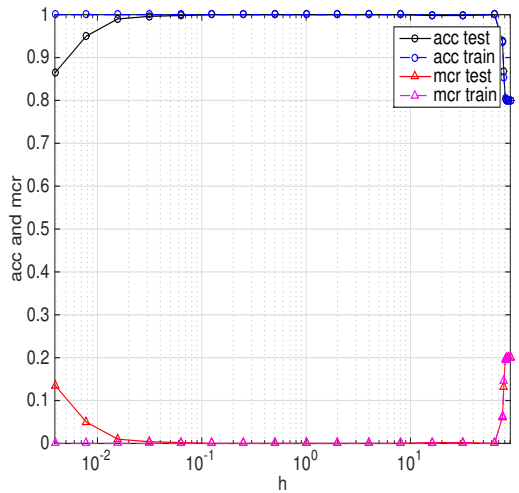
- a cluster with small radius $\not\Rightarrow h^*$ will be small;
- two clusters are easy to separate $\Rightarrow \exists$ large range of optimal h.



4.1 data and decision boundary for $h^* = 64$



4.2 f1 score for test data

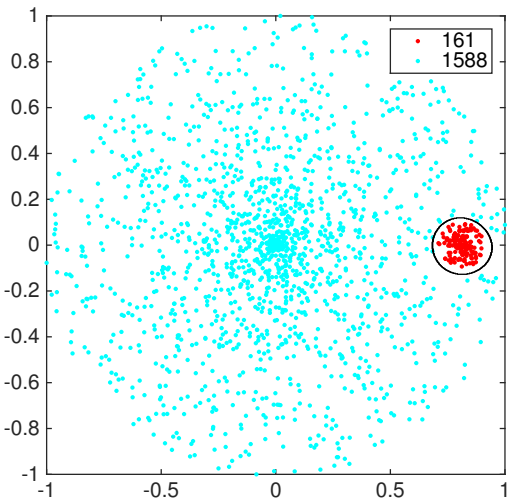


4.3 acc and mcr for train and test data

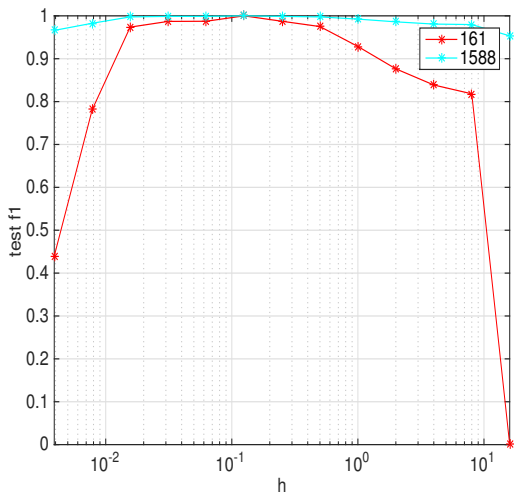
Figure: Case where two clusters are easy to separte via a hyper plane, it degenerates to a linear case. The largest optimal h is therefore very large: $h^* = 64$.

Smaller cluster surrounded by a larger one

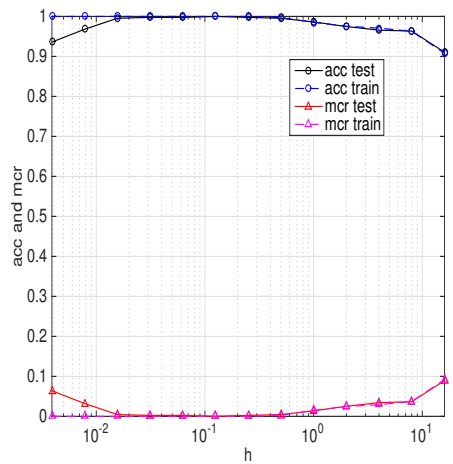
- a smaller h is preferred to detect the small cluster (to achieve a high f1 score)
- In the contrast, if a larger h is used, then either few of them are predicted right (a low recall), or the classifier predicts a lot of points from the other cluster to be them (a low precision).



5.1 data and decision boundary for $h^* = 0.25$



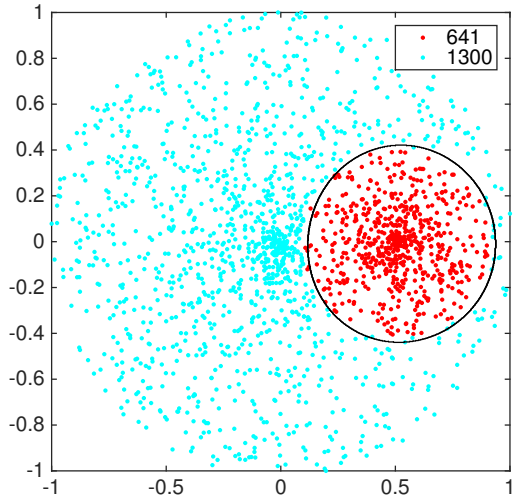
5.2 F1 score for test data



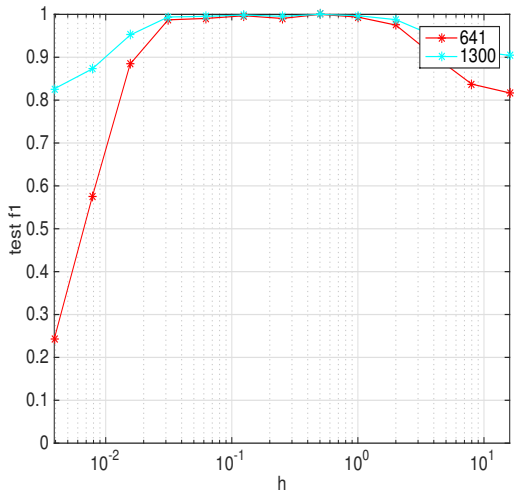
5.3 acc and mcr for train and test data

Figure: $h^* = 0.2500$ (optimal region: test accuracy ≥ 0.9977).

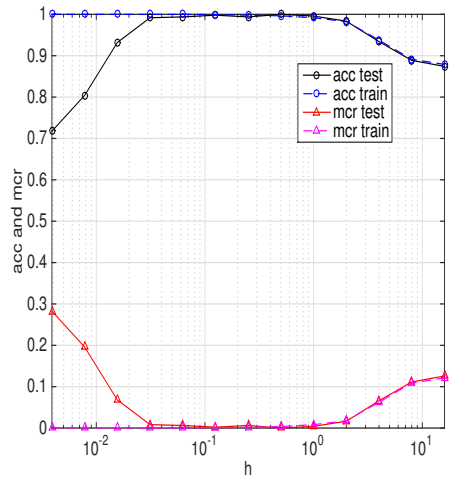
Compared to the previous case, the radius of smaller cluster is 4 times larger, and h^* is also 4 times larger.



6.1 data and decision boundary for $h^* = 1$



6.2 F1 score for test data

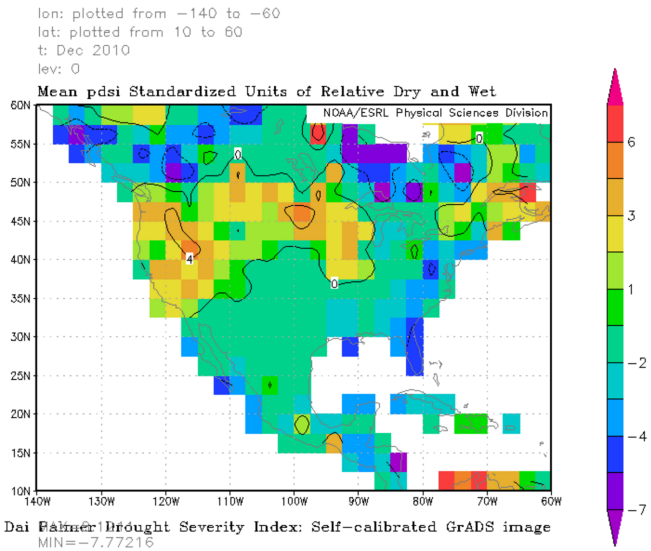


6.3 acc and mcr for train and test data

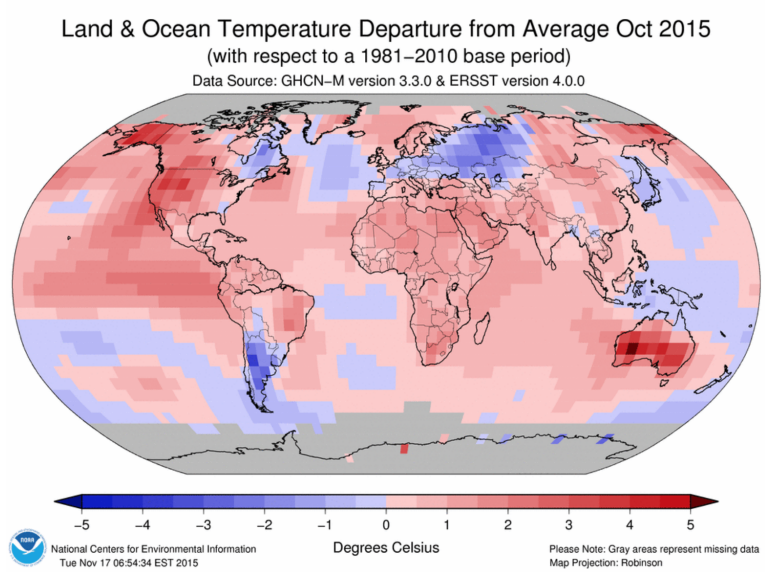
Figure: $h^* = 1$ (optimal region: test accuracy ≥ 0.99).

Real 2D datasets

The followings are two real datasets that not perfectly match, but share similar property as our test cases.



7.1 Mean pdsi standardized units of relative dry and wet, Dec 2010

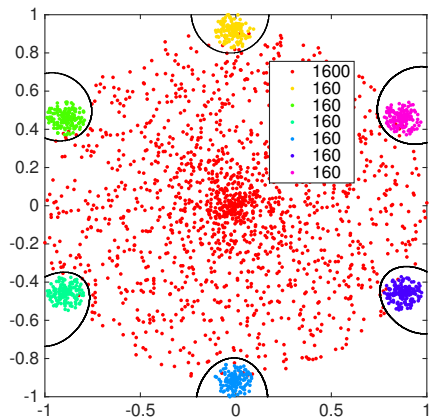


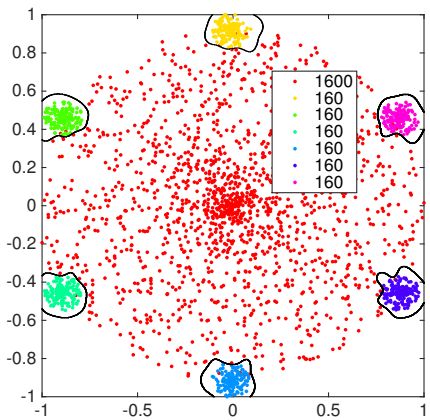
7.2 land & ocean temp departure from avg

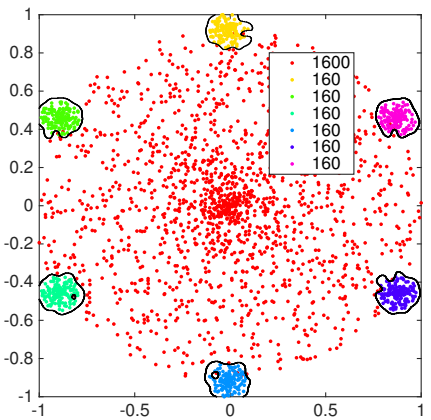
Figure: Several real datasets with an underlying geomerty

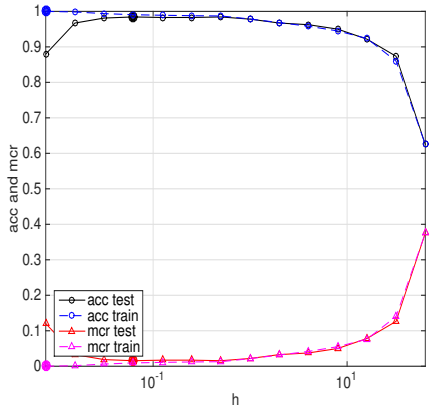
Clusters overlap a little bit on the edge

- it has a larger h^* than the surrounded case;
- the edge of larger cluster has a low density, therefore even a smooth curve will not misclassify many points;
- half of the small cluster is by itself, so a smooth curve on the outside part does not affect the results.









8.1 (0.5000, 0.9844) 8.2 (0.0625, 0.9844) 8.3 (0.0312, 0.9812) 8.4 acc/mcr for train/test data

Figure: Small clusters overlaps on the edge. Threhold for optimal h's $= 0.98$. The subcaptions represent (h, test accuracy)

EMG Physical Action Data Set.

- 10 normal and 10 aggressive physical actions that measure the human activity.
- We randomly sample the same portion from each class, and each class still have the same number of points.
- Interesting observations:

Class	1	2	3	4	5	6	7	8	9	
r_i^2	1e-4	3e-4	1.2e-3	2.8e-3	2.6e-2	2.7e-2	3.6e-2	4.6e-1	1.0	
d_i^2	3e-4	2.3e-3	9,9e-3	1.7e-2	1.9e-1	2.3e-1	1.4e-1	1.4	2.1	
Class	10	11	12	13	14	15	16	17	18	19
r_i^2	1.0	1.6	1.9	2.2	2.2	2.6	2.8	2.9	3.0	3.1
d_i^2	2.0	3.4	4.4	4.2	4.3	5.3	5.4	5.4	5.7	5.9

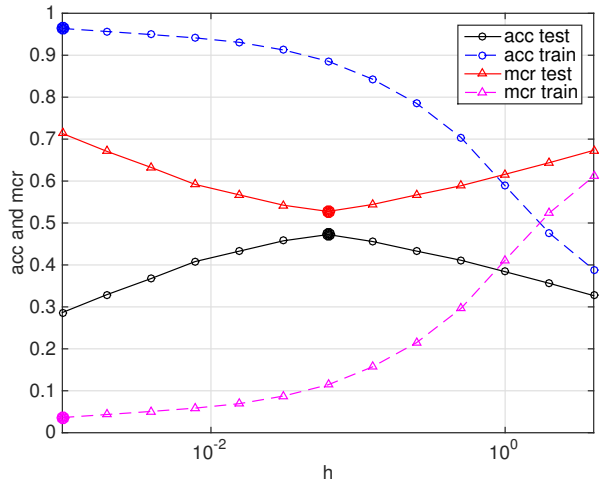
Table: Median of pair-wise distance (d_i), and median of distance to the center (r_i) for each class

- each class has the same number of points.
- quantify its property (shape, density) is hard.
- r_i varies by orders of magnitude (interesting)
- r_i vaguely describe how spread out each class is.

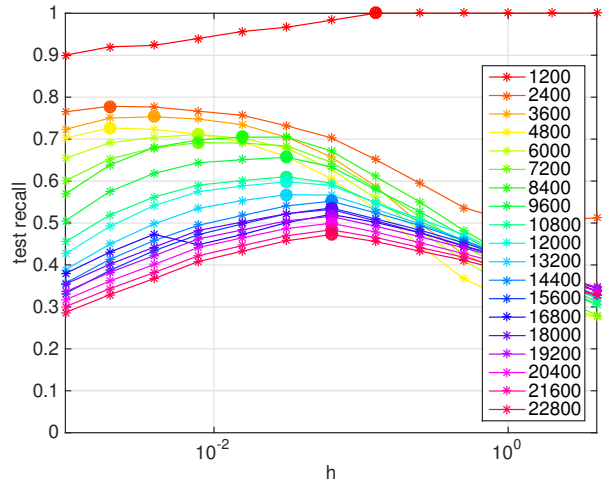
We order the classes by r_i (we use "size" in the following to represent this), and group them in the following manner:

```
g1 = smallest class, g2 = union(smallest class, 2nd smallest class),
..., g20 = all the data
```

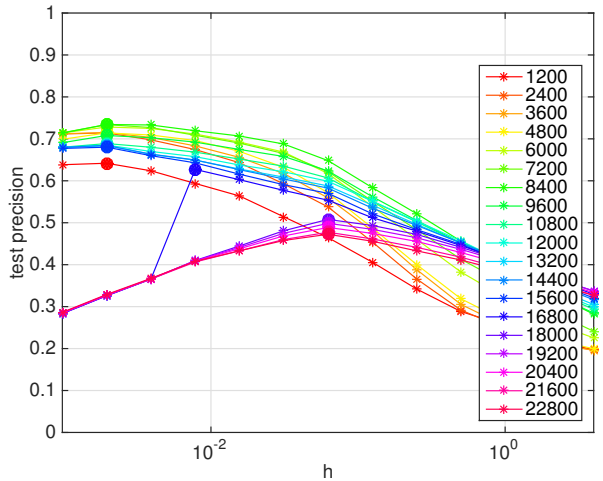
Note: using r_i does not mean it has anything to do with h^*. It just gives us a convenient way to show the results.



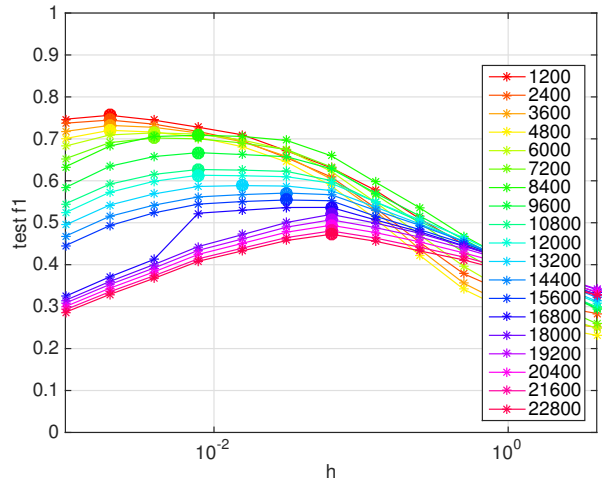
9.1 acc and mcr



9.2 recall test



9.3 precision test



9.4 f1 test

Figure: SVM results for EMG data. Solid circle represents the maximal / minimal point along the curve. Each curve represents one group, and the number in the legend means the number of points in that group: a larger number means more classes with larger "size" are included.

An interesting trend: as we combine more classes with larger r_i together, the optimal h for that group also gets larger.
Conclusions:

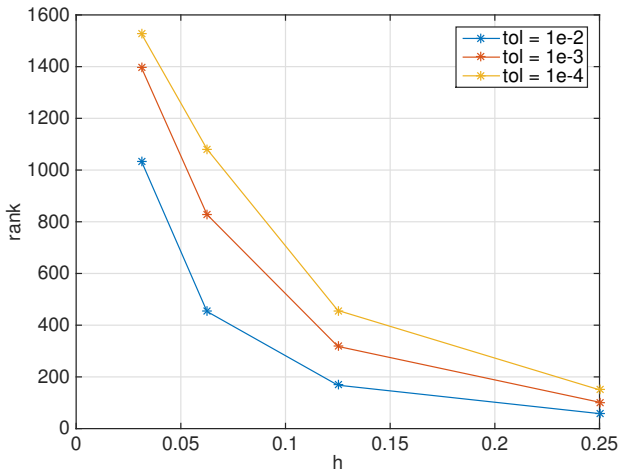
- different classes requires different optimal h;
- classes with smaller "size" tend to prefer a smaller h;
- classes with smaller "size" in this datasets are probably surrounded by other clusters;
- for this dataset, only by using a smaller h can we obtain higher $f1$ score on smaller class.

Outline

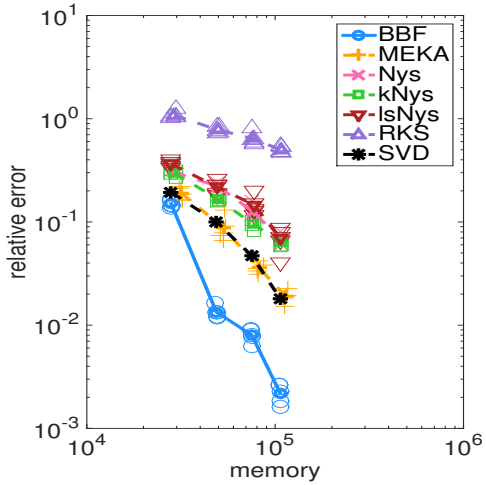
We use experiments to illustrate the behavior of BBF, and show that BBF achieves higher reconstruction accuracy than low-rank method with the same memory footprint.
We show this on selected synthetic dataset (above) and some real datasets. The h used will be either the largest optimal h, or chosen from cross-validation.

For smaller cluster surrounded by a larger one

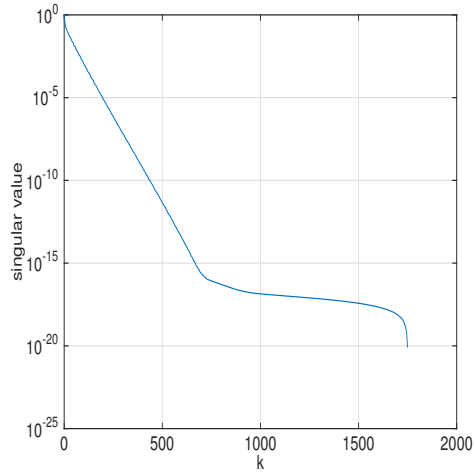
Even though $h^* = 0.25$ results in a low-rank matrix, our method (BBF) still outperforms other low-rank methods.



10.1 the numerical rank for optimal h's (test accuracy > 0.99).



10.2 Accuracy v.s. memory cost for $h^* = 0.2500$

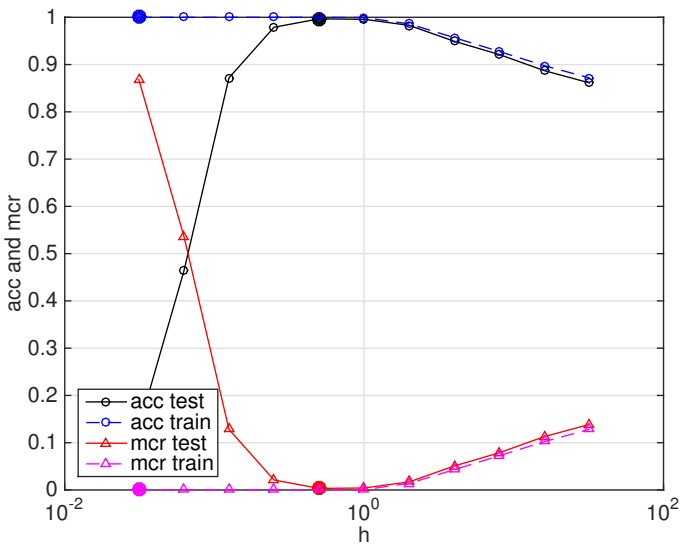


10.3 singular value decay for $h^* = 0.2500$

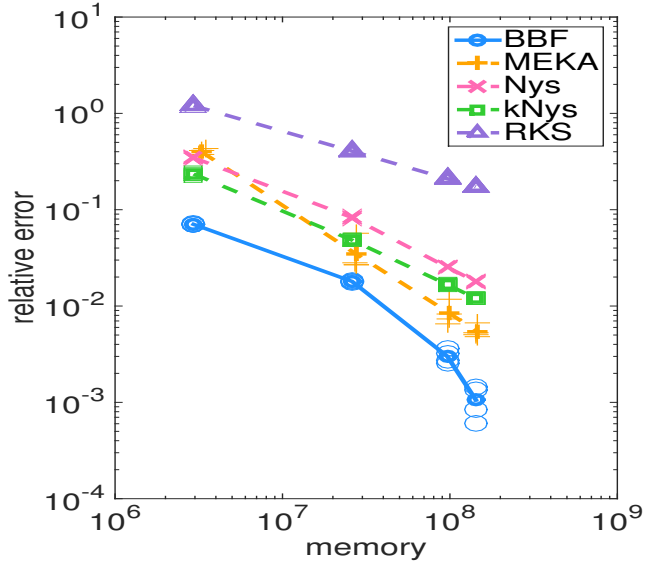
Figure: Matrix ranks and comparisons of BBF with low-rank methods. The matrix comes from our test case (smaller cluster surrounded by a larger one) with $h^* = 0.2500$

Sensorless Drive Diagnosis Data Set

SVD is ignored here because n is too large.



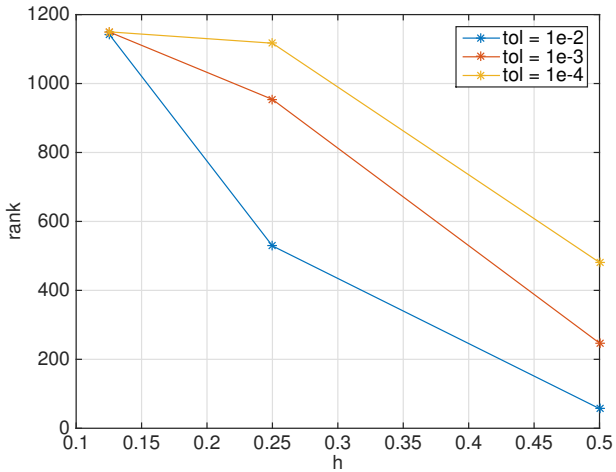
11.1 acc and mcr



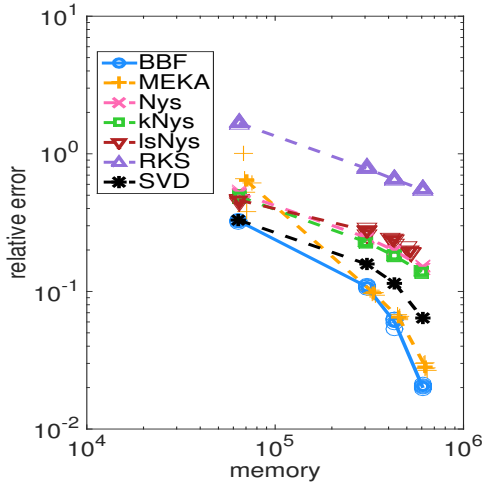
11.2 BBF vs low-rank approximation methods for $h = 0.5$

Figure: Sensorless Drive dataset, $n_{sample} = 27500$, $d = 48$. $h = 0.5$ is chosen via cross-validation.

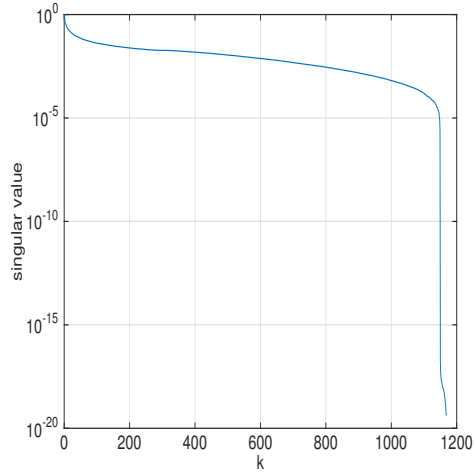
Yeast dataset



12.1 the numerical rank for optimal h's.



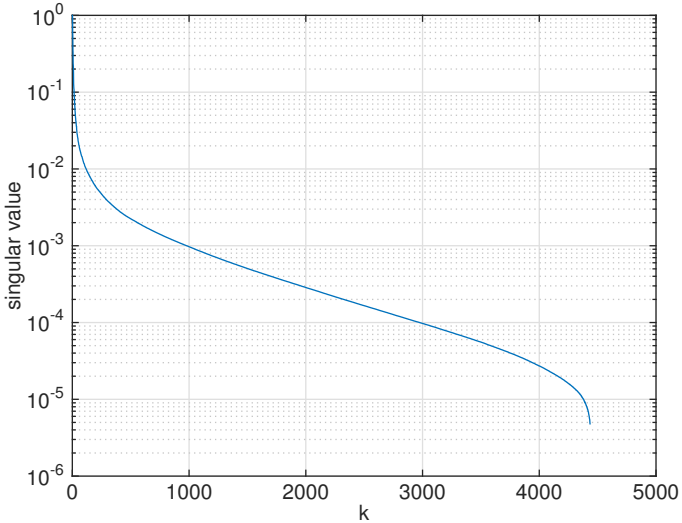
12.2 Accuracy v.s. memory cost for $h = 0.25$



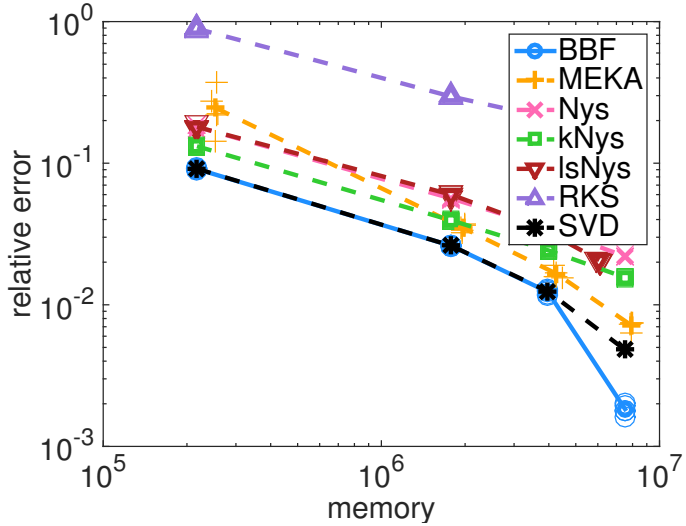
12.3 singular value decay of $h = 0.25$

Figure: Yeast dataset (classes with 5 points and 20 points are excluded). Matrix ranks and comparisons of BBF and low-rank methods. $h = 0.25$ is chosen from cross-validation

Satimage dataset



13.1 SVD for h from cross-validation



13.2 BBF vs low-rank approximation methods for $h = 1$

Figure: Satimage dataset, $n = 4435$, $d = 36$. The numerical ranks are 119 (tol = 1e-2), 979 (tol = 1e-3), 2975 (tol = 1e-4) respectively. $h = 1$ is chosen via cross-validation. The number of clusters in BBF is chosen to be 20 for smaller h and 30 for larger h

Conclusions

- Many data graphs are not well-described by low-rank matrices
- Many data graphs have small-scale clusters & large-scale noise
- We consider details of sensitivity of constructed graphs to choice of r.b.f. parameter
- We have BBF (Block Basis Factorization), using ideas from scientific computing
- BBF is good on memory vs. error tradeoff
- BBF allows us to explore parameter sensitivity for small versus large clusters