Clustering

Usman Roshan

Department of Data Science

NJIT

Clustering

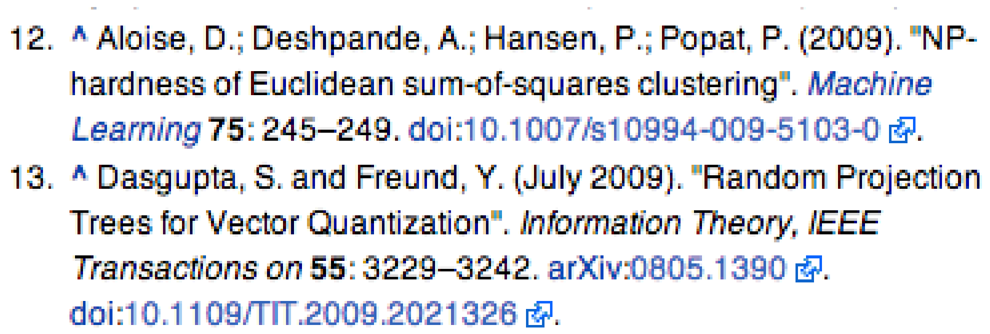
- Suppose we want to cluster *n* vectors in R^d into two groups. Define C_1 and C_2 as the two groups.
- Our objective is to find C_1 and C_2 that minimize

$$\sum_{i=1}^{2} \sum_{x_j \in C_i} \| x_j - m_i \|^2$$

where m_i is the mean of class C_i

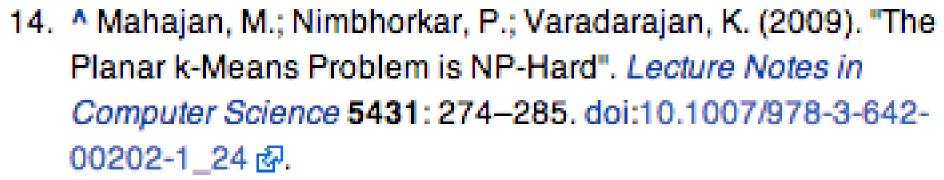
Clustering

- NP hard even for 2-means



12. ^ Aloise, D.; Deshpande, A.; Hansen, P.; Popat, P. (2009). "NP-hardness of Euclidean sum-of-squares clustering". *Machine Learning* **75**: 245–249. doi:10.1007/s10994-009-5103-0.
13. ^ Dasgupta, S. and Freund, Y. (July 2009). "Random Projection Trees for Vector Quantization". *Information Theory, IEEE Transactions on* **55**: 3229–3242. arXiv:0805.1390. doi:10.1109/TIT.2009.2021326.

- NP hard even on plane



14. ^ Mahajan, M.; Nimbhorkar, P.; Varadarajan, K. (2009). "The Planar k-Means Problem is NP-Hard". *Lecture Notes in Computer Science* **5431**: 274–285. doi:10.1007/978-3-642-00202-1_24.

- K-means heuristic
  - Popular and hard to beat
  - Introduced in 1950s and 1960s

K-means algorithm for two clusters

Input: $x_i \in R^d, i = 1 \ldots n$

Algorithm:

1. Initialize: assign x_i to C_1 or C_2 with equal probability and compute means:

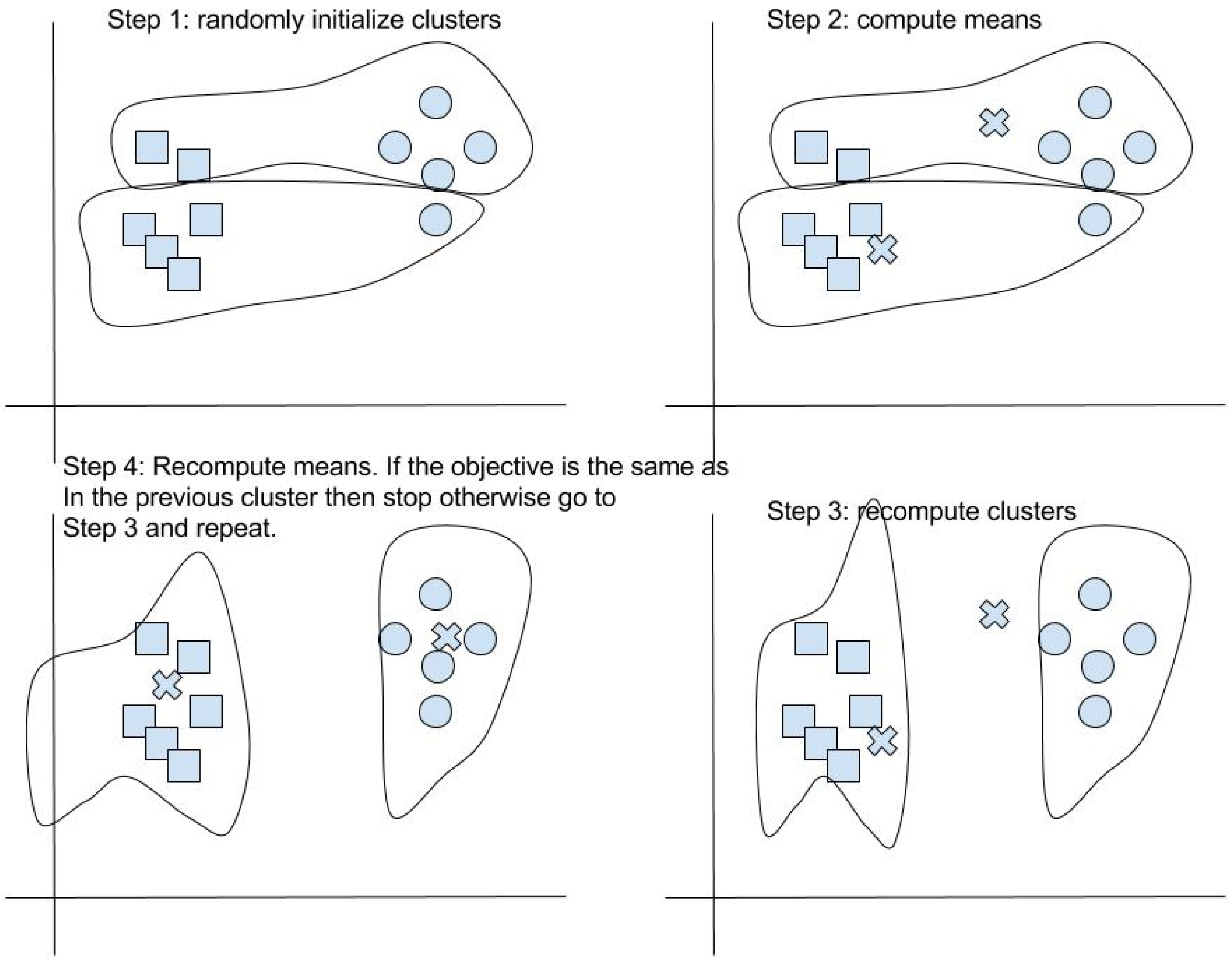
$$m_1 = \frac{1}{|C_1|} \sum_{x_i \in C_1} x_i \qquad m_2 = \frac{1}{|C_2|} \sum_{x_i \in C_2} x_i$$

2. Recompute clusters: assign x_i to C_1 if $||x_i - m_1|| < ||x_i - m_2||$, otherwise assign to C_2
3. Recompute means m_1 and m_2
4. Compute objective

$$\sum_{i=1}^{2} \sum_{x_j \in C_i} \| x_j - m_i \|^2$$

5. Compute objective of new clustering. If difference is smaller than δ then stop, otherwise go to step 2.

K-means example


Step 1: randomly initialize clusters
Step 2: compute means
Step 4: Recompute means. If the objective is the same as
In the previous cluster then stop otherwise go to
Step 3 and repeat.
Step 3: recompute clusters

K-means

- Is it guaranteed to find the clustering which optimizes the objective?
- It is guaranteed to find a local optimal
- We can prove that the objective decreases with subsequence iterations

Proof sketch of convergence of k-means

$$\sum_{i=1}^{2} \sum_{x_j \in C_i} \| x_j - m_i \|^2 \geq$$

Justification of first inequality: by assigning x_j to the closest mean the objective decreases or stays the same

$$\sum_{i=1}^{2} \sum_{x_j \in C_i^*} \| x_j - m_i \|^2 \geq$$

Justification of second inequality: for a given cluster its mean minimizes squared error loss

$$\sum_{i=1}^{2} \sum_{x_j \in C_i^*} \| x_j - m_i^* \|^2$$

K-means clustering

- K-means is the Expected-Maximization solution if we assume data is generated by Gaussian distribution
  - EM: Find clustering of data that maximizes likelihood
  - Unsupervised means no parameters given. Thus we iterate between estimating expected and actual values of parameters
- PCA gives relaxed solution to k-means clustering. Sketch of proof:
  - Cast k-means clustering as maximization problem
  - Relax cluster indicator variables to be continuous and solution is given by PCA

K-means clustering

- K-medians variant:
  - Select cluster center that is the median
  - Has the effect of minimizing L1 error
- K-medoid
  - Cluster center is an actual datapoint (not same as k-medians)
- Algorithms similar to k-means
- Similar local minima problems to k-means

Other clustering algorithms

- Spherical k-means
  - Normalize each datapoint (set length to 1)
  - Clustering by finding center with minimum cosine angle to cluster points
  - Similar iterative algorithm to Euclidean k-means
  - Converges with similar proofs.

Other clustering algorithms

- Hierarchical clustering
  - Initialize n clusters where each datapoint is in its own cluster
  - Merge two nearest clusters into one
  - Update distances of new cluster to existing ones
  - Repeat step 2 until k clusters are formed.

Methods to evaluate a given clustering:

1. If we don't have ground truth labels then we can measure the homogeneity and separability.

   Homogeneity: this is the same as the k-means objective and measures how compact are the clusters.

   $$H_{AVE} = 1/m \sum_{i=0}^{m} \sum_{x_j \in C_i} \left\| x_j - m_i \right\|^2$$

   where m is the number of clusters and mi is the mean of cluster Ci.

   Separability measures how separated the clusters:

   $$S_{AVE} = \frac{1}{\sum_{i=0}^{m-1} \sum_{j=i+1}^{m} |C_i||C_j|} \sum_{i=0}^{m-1} \sum_{j=i+1}^{m} |C_i||C_j| \left\| m_i - m_j \right\|^2$$

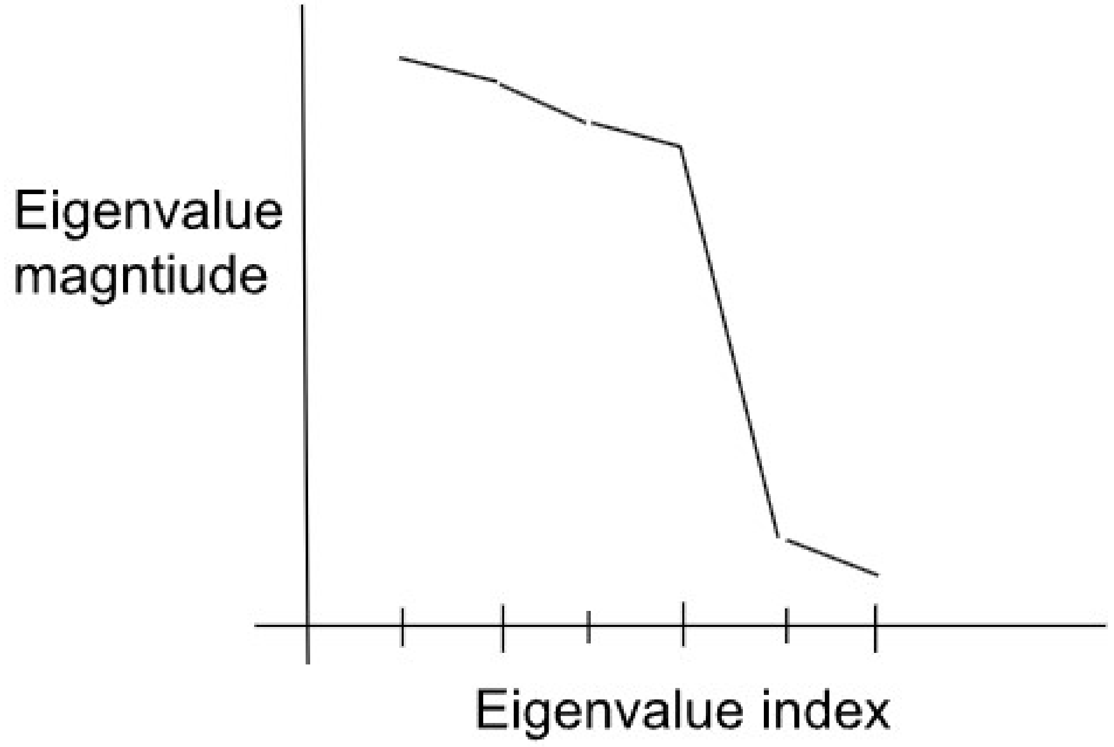
   where |Ci| is the size of cluster Ci.

   We could combined the above into one metric $\frac{S_{AVE}}{H_{AVE}}$

2. If we have ground truth then there are two methods:
   a. $\Sigma_{ij} || T_{ij} - C_{ij} ||^2$ where T_{ij}=1 iff xi and xj have the same ground truth label and 0 otherwise, and C_{ij}= 1 iff xi and xj are in the same cluster and 0 otherwise
   b. Classification accuracy: this is the maximum accuracy across permutations of class labelings in the clustering

Methods to choose number of clusters:

1. Cross-validation. Just like classification we can try to determine the number of clusters that maximize cross-validation accuracy. We would need labels for this. Without labels we can try to maximize perhaps cluster homogeneity and separability.
2. Magnitude of PCA eigenvalues. Recall that the eigenvalue itself in PCA is the variance of the data when projected on the corresponding eigenvector. One rough way to measure the number of clusters is to see how many eigenvalues are playing an important role in the variance of the data. If there is a *large* drop between eigenvalues e_k and e_{k+1} then this means there are probably k+1 clusters in the data.

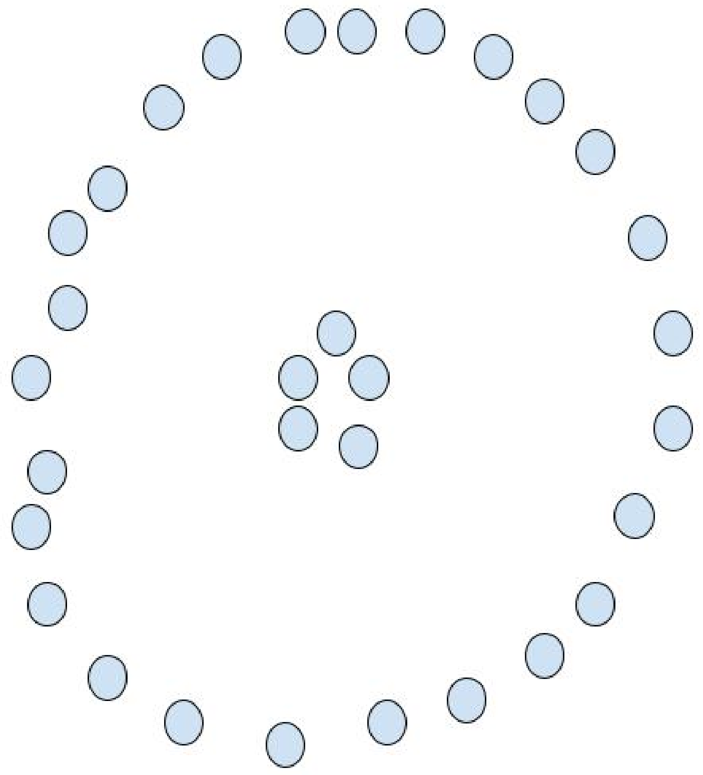


Graph clustering

- Graph Laplacians widely used in spectral clustering (see tutorial on course website)
- Weights C_{ij} may be obtained via
  - Epsilon neighborhood graph
  - K-nearest neighbor graph
  - Fully connected graph
- Allows semi-supervised analysis (where test data is available but not labels)

Graph clustering for non-linear data

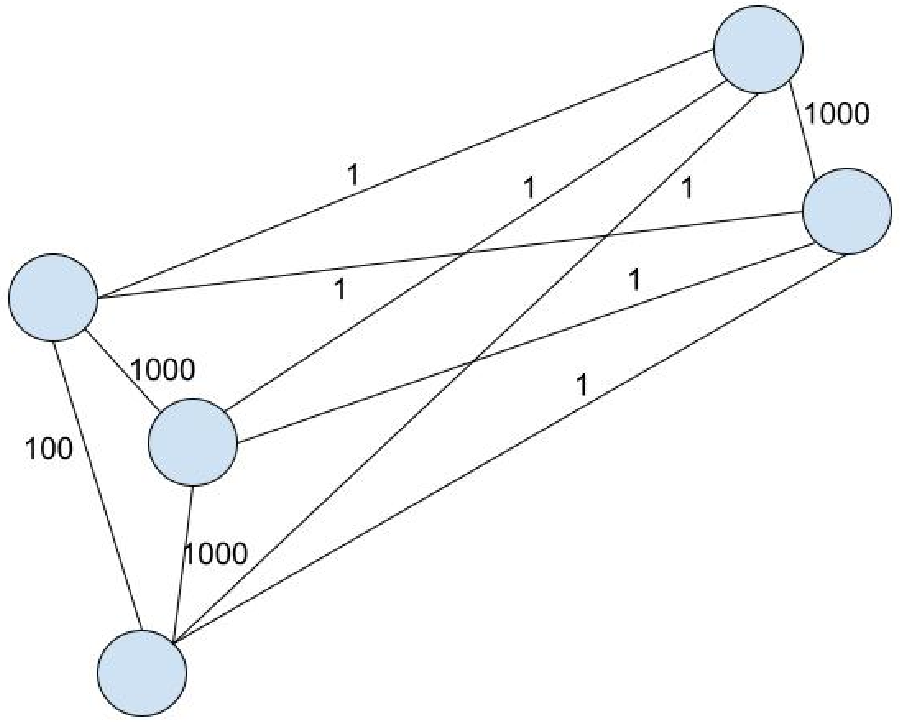
Graph clustering is really useful in clustering non-linear data. In the example below we see two clusters. One is the circle at the center with a short radius and the other has a larger radius. However, k-means will mix the points in the outer and inner circles because it is a linear separator. Suppose instead of k-means we created a similarity graph and connected each edge to its nearest two neighbors. That would clearly give us two connected components: one of the points in the inner circle and one of the outside. The connected components are the two clusters that we seek.



Graph clustering

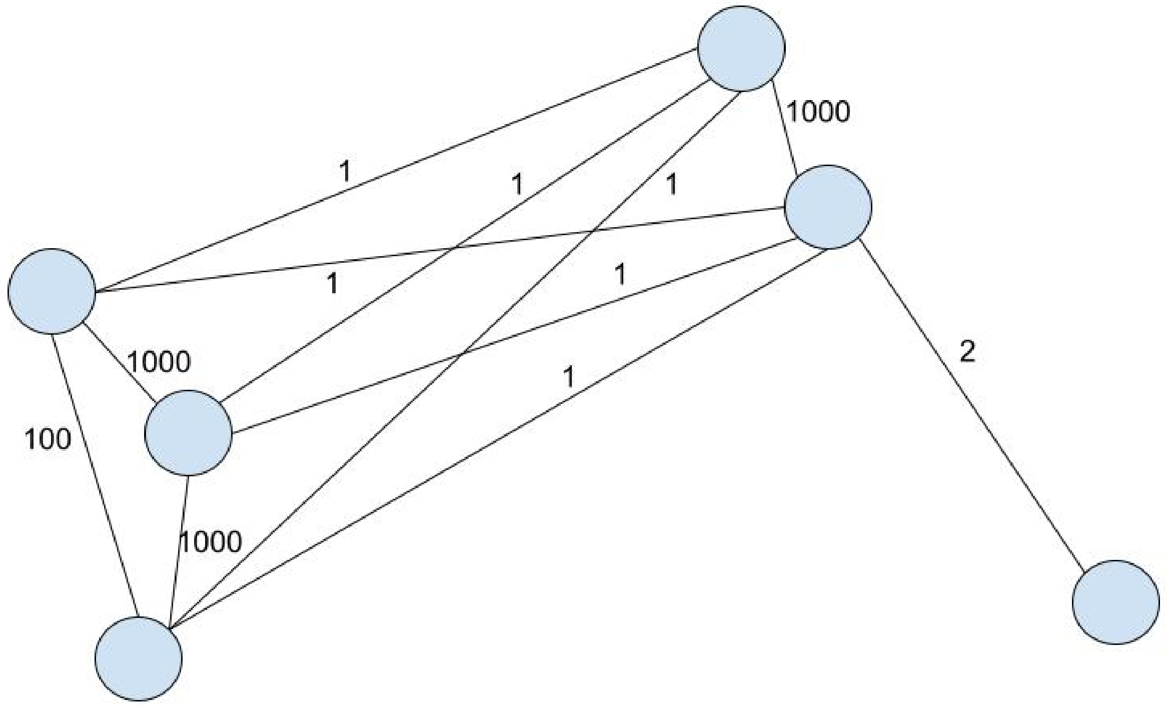
- We can cluster data using the mincut problem
- Balanced version is NP-hard
- We can rewrite balanced mincut problem with graph Laplacians. Still NP-hard because solution is allowed only discrete values
- By relaxing to allow real values we obtain spectral clustering.

Min cut graph clustering



Consider the connected graph above. Let each edge have weight 1/(exp(||xi-xj||^2). We can cluster the points above by looking for the mincut. A mincut is a set of edges of minimum weight that separates the graph into two connected components. If you take out the edges of weight 1 then we have two connected components and cut of total weight 6. And so we can cluster points by forming a graph and looking for the mincut. Remember from your computer science algorithms course we can find a mincut with the maxflow-mincut polynomial time algorithm. This algorithm gives us a set of edges wirth the maximum flow that are also the edges with the mincut.

Drawback of min cut approach



Consider the connected graph above from the drawing in Graph Cluster I. We now have a new datapoint that is connected to just one point and has edge weight of 2. We see that the mincut is now given by the edge of weight 2. If we remove this edge we don't get a meaningful cluster. In fact the new datapoint is an outlier. In this way min cuts are problematic.

Graph clustering

- We can cluster data using the mincut problem
- Balanced version is NP-hard
- We can rewrite balanced mincut problem with graph Laplacians. Still NP-hard because solution is allowed only discrete values
- By relaxing to allow real values we obtain spectral clustering.

Graph Laplacians

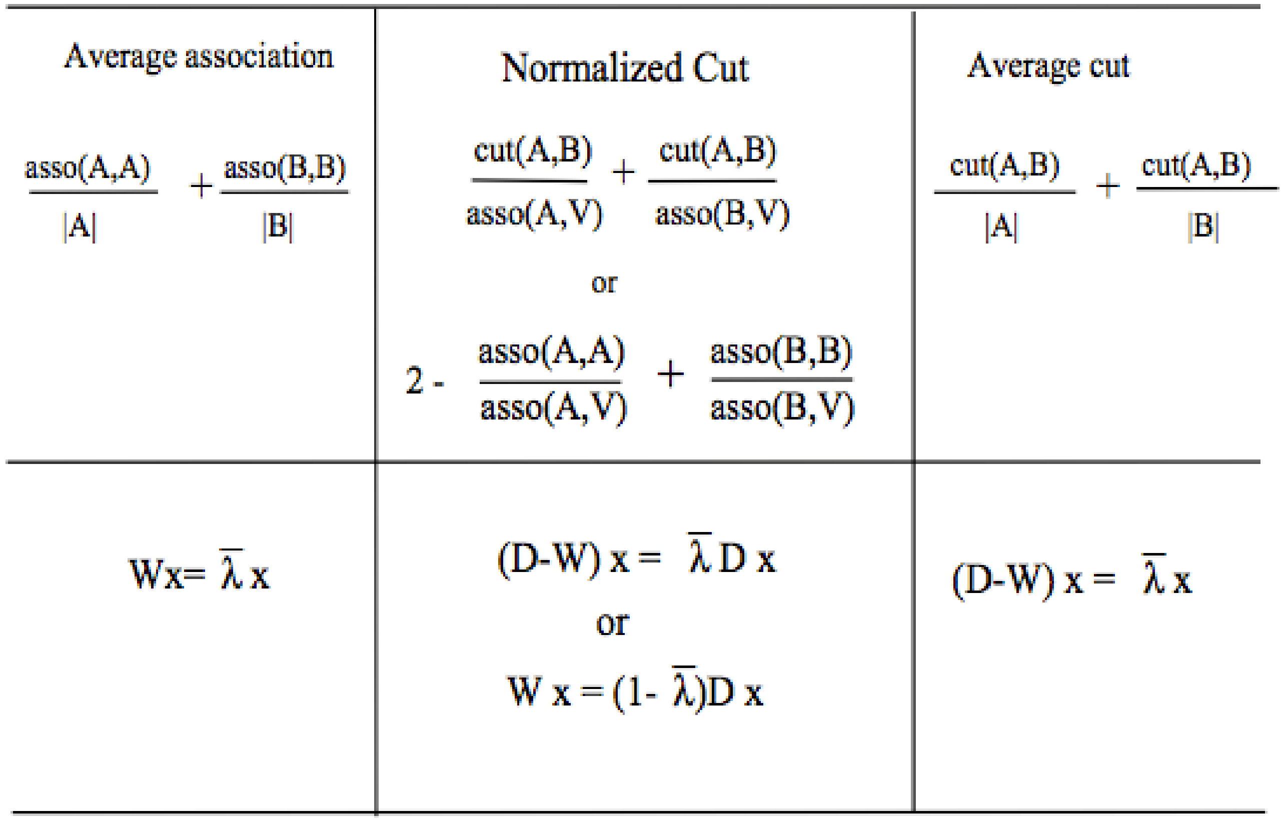
- We can perform clustering with the Laplacian $L = D - C$ where $D_{ii} = \Sigma_j C_{ij}$
- Basic algorithm for k clusters:
  - Compute first k eigenvectors v_i of Laplacian matrix
  - Let $V = [v_1, v_2, \ldots, v_k]$
  - Cluster rows of V (using k-means)

Graph clustering

- Why does it work?
- Relaxation of NP-hard clustering
- What is relaxation: changing the hard objective into an easier one that will yield real (as opposed to discrete) solutions and be at most the true objective
- Can a relaxed solution give optimal discrete solution? No guarantees

Graph clustering

- Cut problems are NP-hard.
- We obtain relaxed solutions via eigenvectors of original weight matrix and its Laplacian.



Average association	Normalized Cut	Average cut
$\frac{\text{asso}(A,A)}{\lvert A \rvert} + \frac{\text{asso}(B,B)}{\lvert B \rvert}$	$\frac{\text{cut}(A,B)}{\text{asso}(A,V)} + \frac{\text{cut}(A,B)}{\text{asso}(B,V)}$ or $2 - \frac{\text{asso}(A,A)}{\text{asso}(A,V)} + \frac{\text{asso}(B,B)}{\text{asso}(B,V)}$	$\frac{\text{cut}(A,B)}{\lvert A \rvert} + \frac{\text{cut}(A,B)}{\lvert B \rvert}$
$Wx = \bar{\lambda} x$	$(D-W)\, x = \bar{\lambda} D x$ or $W x = (1 - \bar{\lambda}) D x$	$(D-W)\, x = \bar{\lambda} x$

From Shi and Malik, *IEEE Pattern Analysis and Machine Intelligence*, 2000

Graph Laplacians and cuts

- Recall from earlier the WMV dimensionality reduction criterion

$$\arg\max_{w} \frac{1}{2n} \sum_{i,j} C_{ij} (w^T (x_i - x_j))^2$$

- WMV is also given by

$$\arg\max_{w} \tfrac{1}{n} w^T X L X^T w$$

where X = [x_1, x_2, …, x_n] contains each x_i as a column and L is the graph Laplacian.

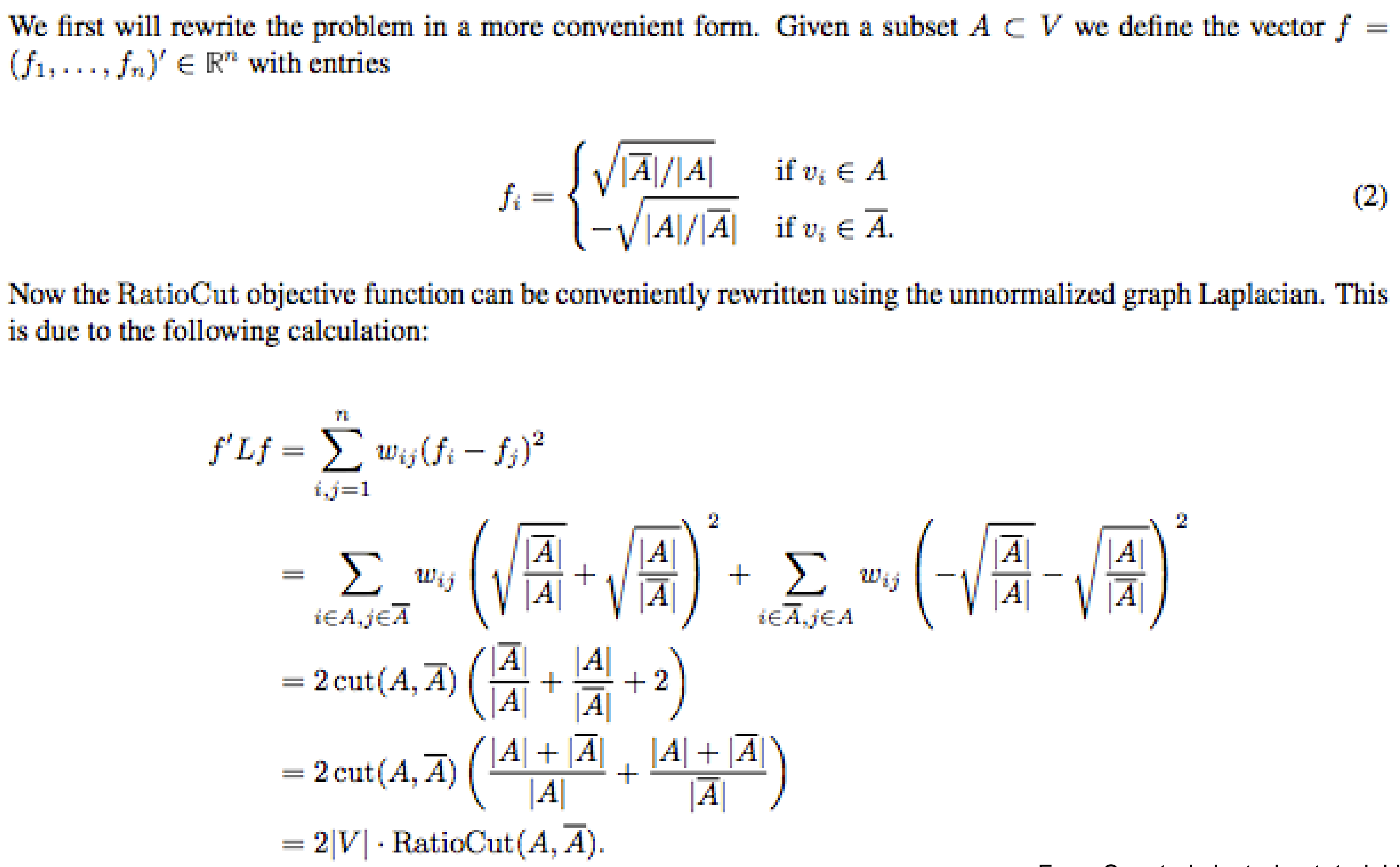
- Thus

$$XLX^T = \sum_{i,j} C_{ij} (x_i - x_j)^2$$

Graph Laplacians and cuts

- For a graph with arbitrary edge weights if we define input vectors x in a certain way then the WMV criterion gives us the ratio cut.
- For proof see page 9 of tutorial (also give in next slide)

Graph Laplacians and cuts



We first will rewrite the problem in a more convenient form. Given a subset $A \subset V$ we define the vector $f = (f_1, \dots, f_n)' \in \mathbb{R}^n$ with entries

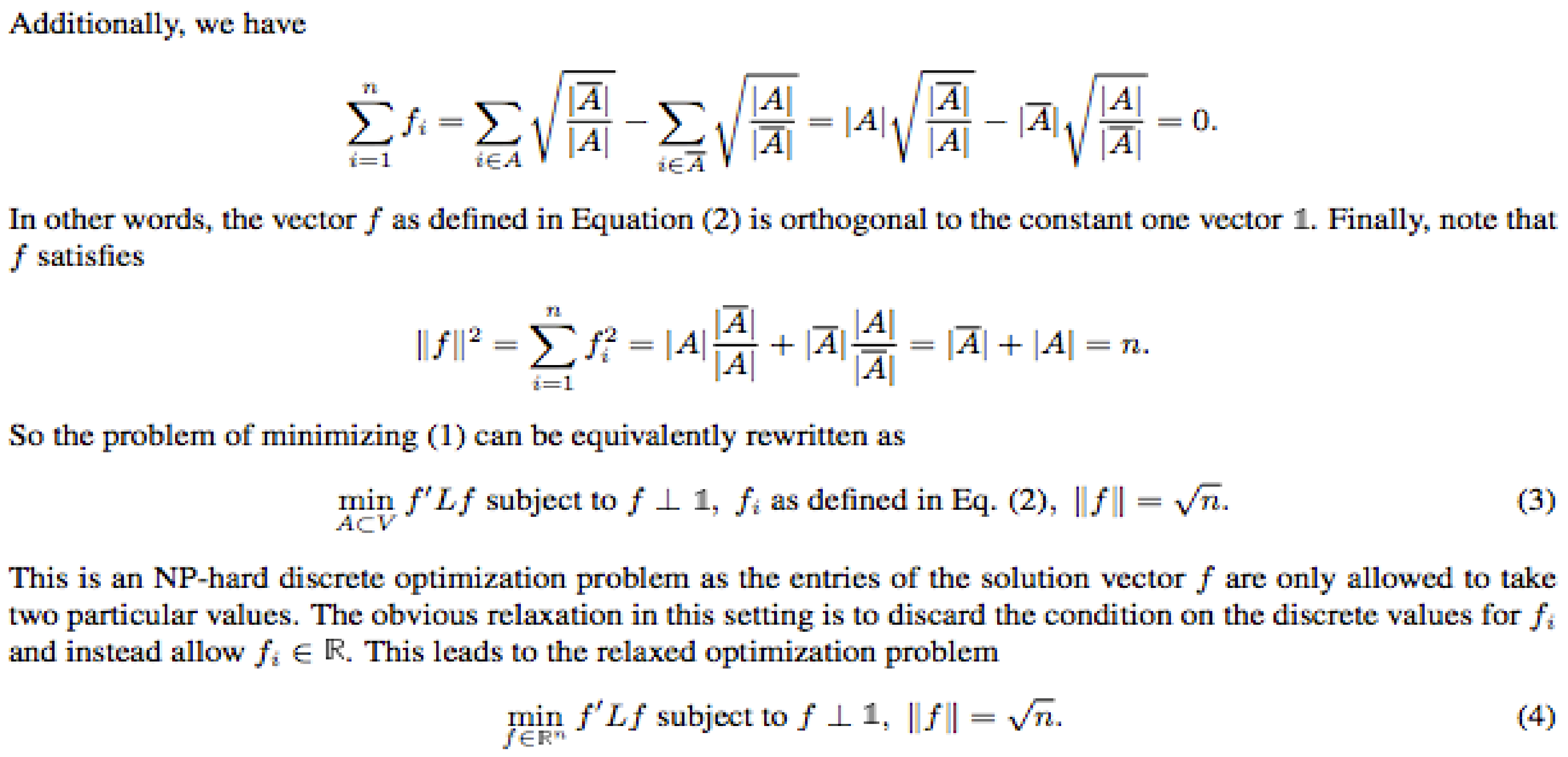
$$f_i = \begin{cases} \sqrt{|\overline{A}|/|A|} & \text{if } v_i \in A \\ -\sqrt{|A|/|\overline{A}|} & \text{if } v_i \in \overline{A}. \end{cases} \tag{2}$$

Now the RatioCut objective function can be conveniently rewritten using the unnormalized graph Laplacian. This is due to the following calculation:

$$\begin{aligned} f'Lf &= \sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2 \\ &= \sum_{i\in A, j\in\overline{A}} w_{ij}\left(\sqrt{\frac{|\overline{A}|}{|A|}} + \sqrt{\frac{|A|}{|\overline{A}|}}\right)^2 + \sum_{i\in\overline{A}, j\in A} w_{ij}\left(-\sqrt{\frac{|\overline{A}|}{|A|}} - \sqrt{\frac{|A|}{|\overline{A}|}}\right)^2 \\ &= 2\,\mathrm{cut}(A,\overline{A})\left(\frac{|\overline{A}|}{|A|} + \frac{|A|}{|\overline{A}|} + 2\right) \\ &= 2\,\mathrm{cut}(A,\overline{A})\left(\frac{|A| + |\overline{A}|}{|A|} + \frac{|A| + |\overline{A}|}{|\overline{A}|}\right) \\ &= 2|V| \cdot \mathrm{RatioCut}(A,\overline{A}). \end{aligned}$$

From Spectral clustering tutorial by Ulrike von Luxborg

Graph Laplacians and cuts



Additionally, we have

$$\sum_{i=1}^{n} f_i = \sum_{i\in A} \sqrt{\frac{|\overline{A}|}{|A|}} - \sum_{i\in \overline{A}} \sqrt{\frac{|A|}{|\overline{A}|}} = |A|\sqrt{\frac{|\overline{A}|}{|A|}} - |\overline{A}|\sqrt{\frac{|A|}{|\overline{A}|}} = 0.$$

In other words, the vector f as defined in Equation (2) is orthogonal to the constant one vector $\mathbb{1}$. Finally, note that f satisfies

$$\|f\|^2 = \sum_{i=1}^{n} f_i^2 = |A|\frac{|\overline{A}|}{|A|} + |\overline{A}|\frac{|A|}{|\overline{A}|} = |\overline{A}| + |A| = n.$$

So the problem of minimizing (1) can be equivalently rewritten as

$$\min_{A\subset V} f'Lf \text{ subject to } f \perp \mathbb{1},\ f_i \text{ as defined in Eq. (2)},\ \|f\| = \sqrt{n}. \tag{3}$$

This is an NP-hard discrete optimization problem as the entries of the solution vector f are only allowed to take two particular values. The obvious relaxation in this setting is to discard the condition on the discrete values for f_i and instead allow $f_i \in \mathbb{R}$. This leads to the relaxed optimization problem

$$\min_{f\in\mathbb{R}^n} f'Lf \text{ subject to } f \perp \mathbb{1},\ \|f\| = \sqrt{n}. \tag{4}$$

From Spectral clustering tutorial by
Ulrike von Luxborg

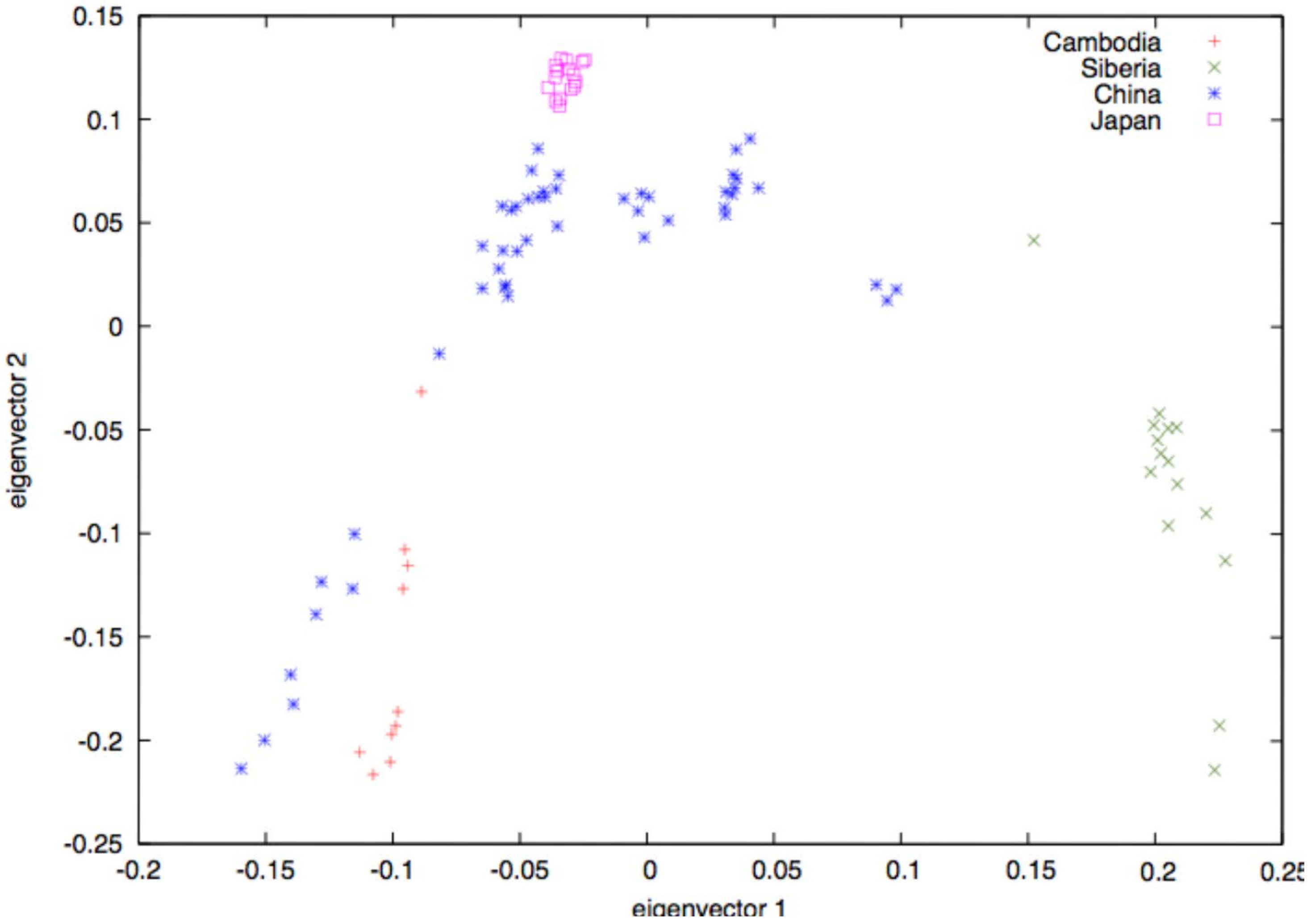
Application on population structure data

- Data are vectors x_i where each feature takes on values 0, 1, and 2 to denote number of alleles of a particular single nucleotide polymorphism (SNP)
- Output y_i is an integer indicating the population group a vector belongs to

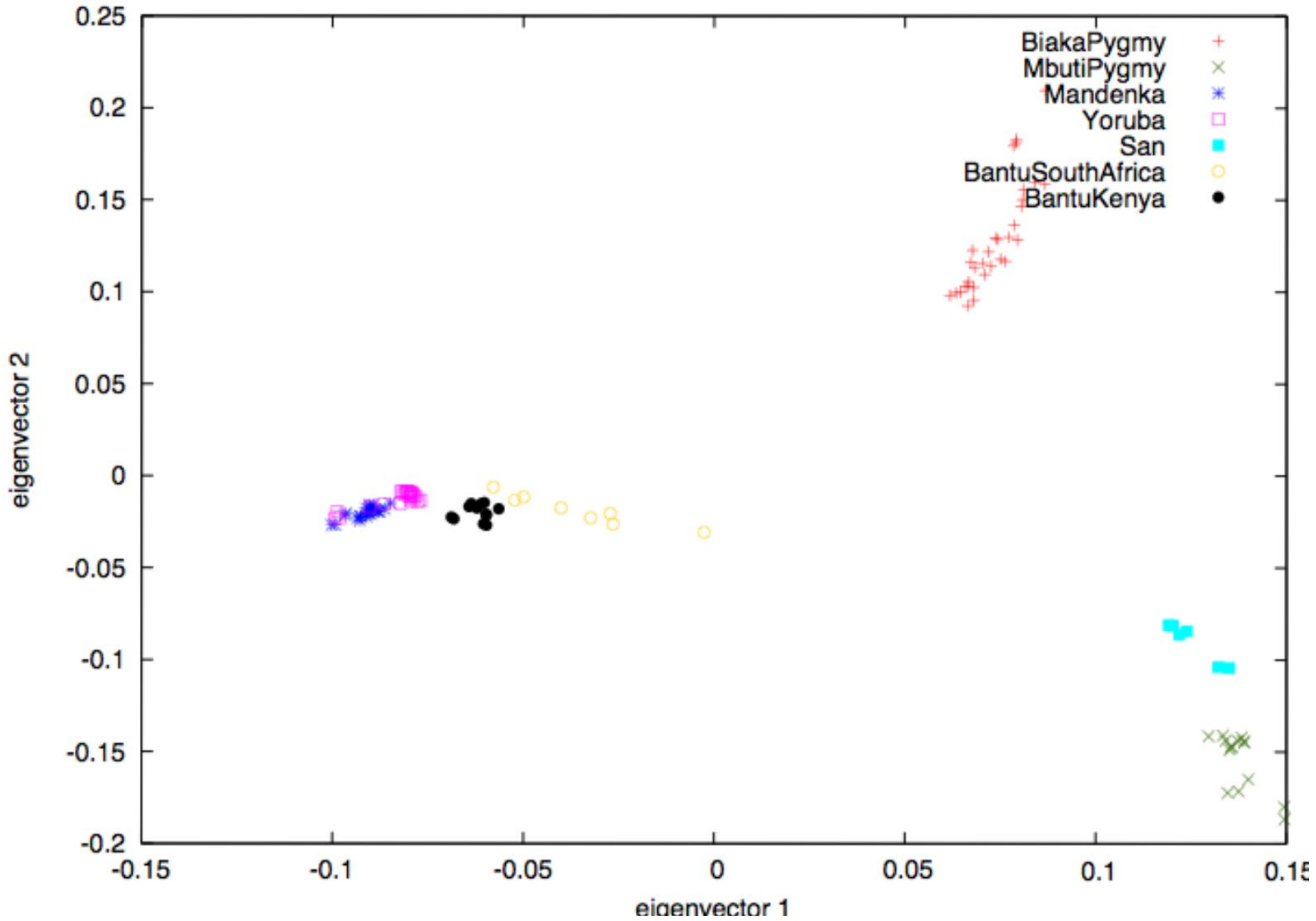
Publicly available real data

- Datasets (Noah Rosenberg’s lab):
    - East Asian admixture: 10 individuals from Cambodia, 15 from Siberia, 49 from China, and 16 from Japan; 459,188 SNPs
    - African admixture: 32 Biaka Pygmy individuals, 15 Mbuti Pygmy, 24 Mandenka, 25 Yoruba, 7 San from Namibia, 8 Bantu of South Africa, and 12 Bantu of Kenya; 454,732 SNPs
    - Middle Eastern admixture: contains 43 Druze from Israil-Carmel, 47 Bedouins from Israel-Negev, 26 Palestinians from Israel-Central, and 30 Mozabite from Algeria-Mzab; 438,596 SNPs

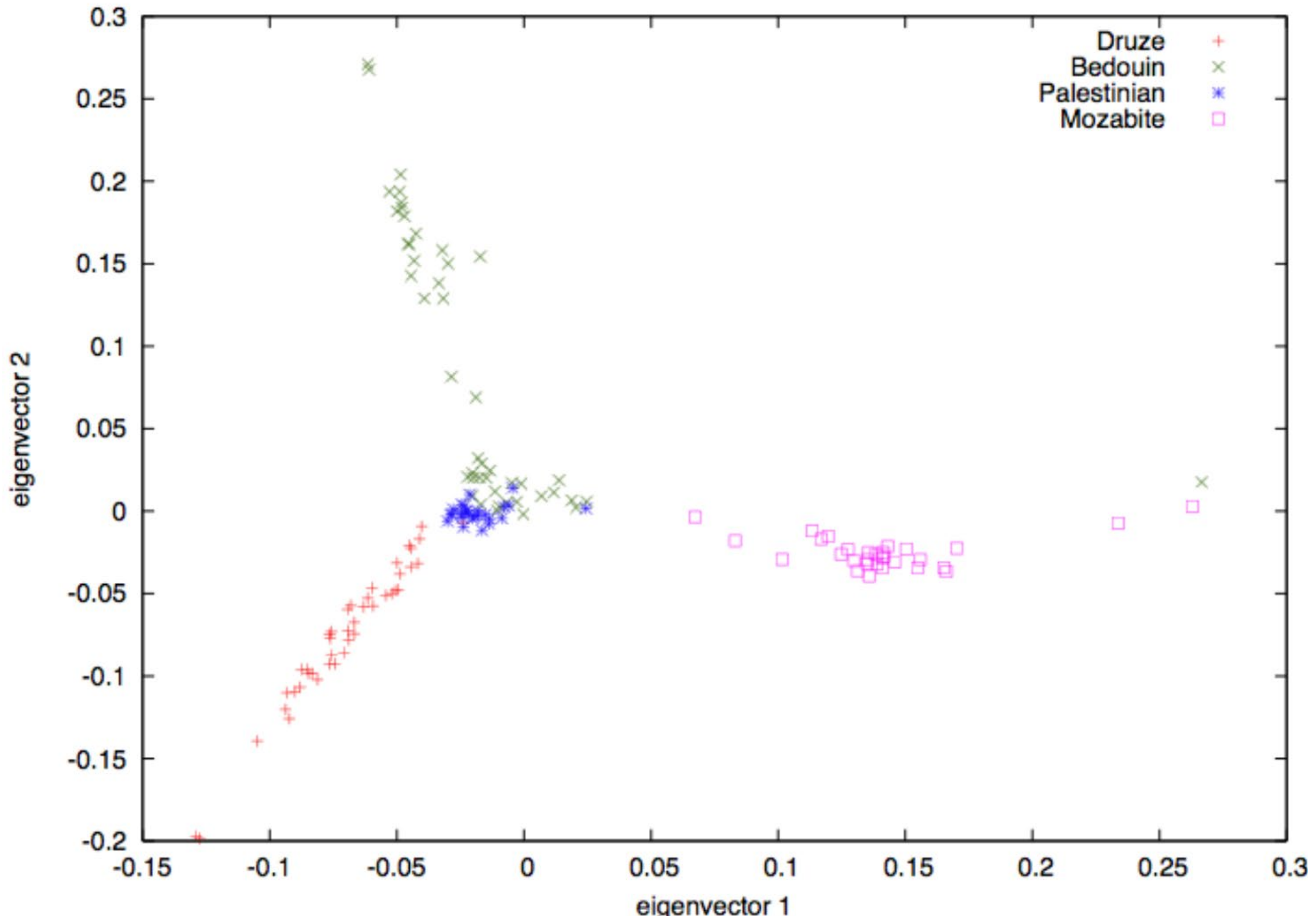
East Asian populations



African populations



Middle Eastern populations



K-means applied to PCA data

- PCA and kernel PCA (poly degree 2)
- K-means and spherical k-means
- Number of clusters set to true value

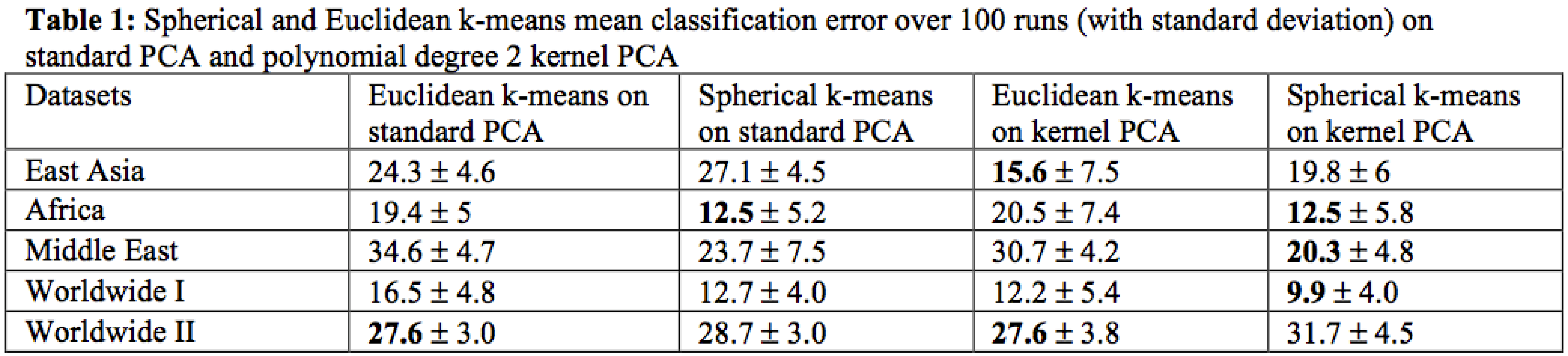


Table 1: Spherical and Euclidean k-means mean classification error over 100 runs (with standard deviation) on standard PCA and polynomial degree 2 kernel PCA

Datasets	Euclidean k-means on standard PCA	Spherical k-means on standard PCA	Euclidean k-means on kernel PCA	Spherical k-means on kernel PCA
East Asia	24.3 ± 4.6	27.1 ± 4.5	**15.6** ± 7.5	19.8 ± 6
Africa	19.4 ± 5	**12.5** ± 5.2	20.5 ± 7.4	**12.5** ± 5.8
Middle East	34.6 ± 4.7	23.7 ± 7.5	30.7 ± 4.2	**20.3** ± 4.8
Worldwide I	16.5 ± 4.8	12.7 ± 4.0	12.2 ± 5.4	**9.9** ± 4.0
Worldwide II	**27.6** ± 3.0	28.7 ± 3.0	**27.6** ± 3.8	31.7 ± 4.5