Lecture 1:
Unsupervised Learning;
Clustering with *k*-means and *k*-medoids

Lester Mackey

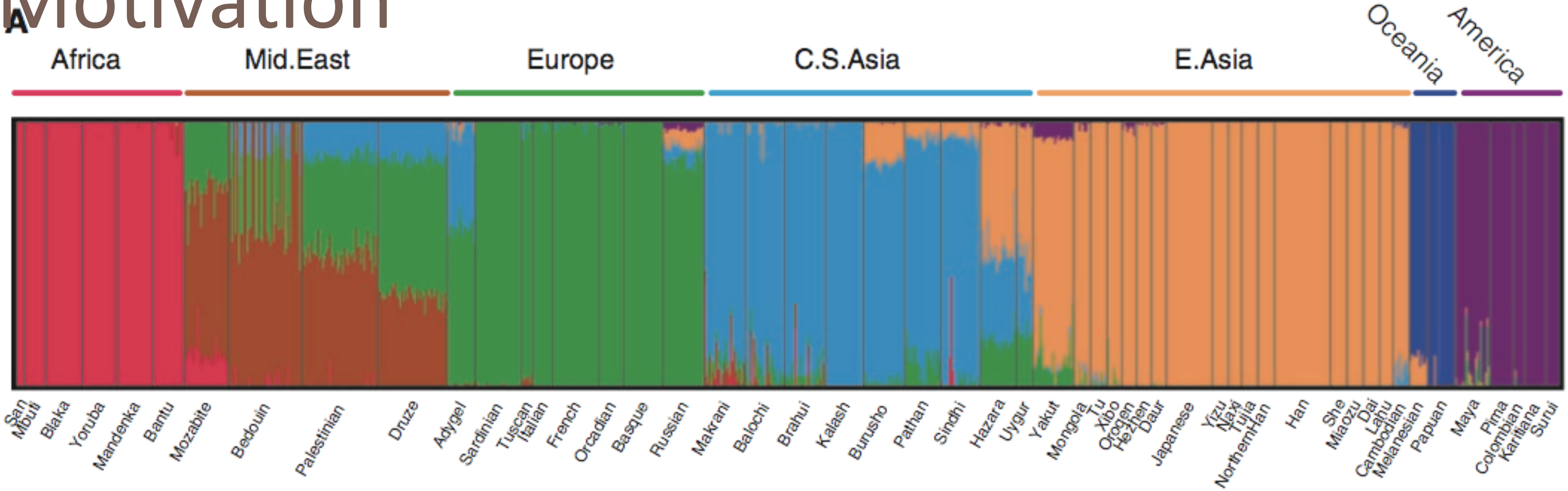
March 31, 2014

Motivation

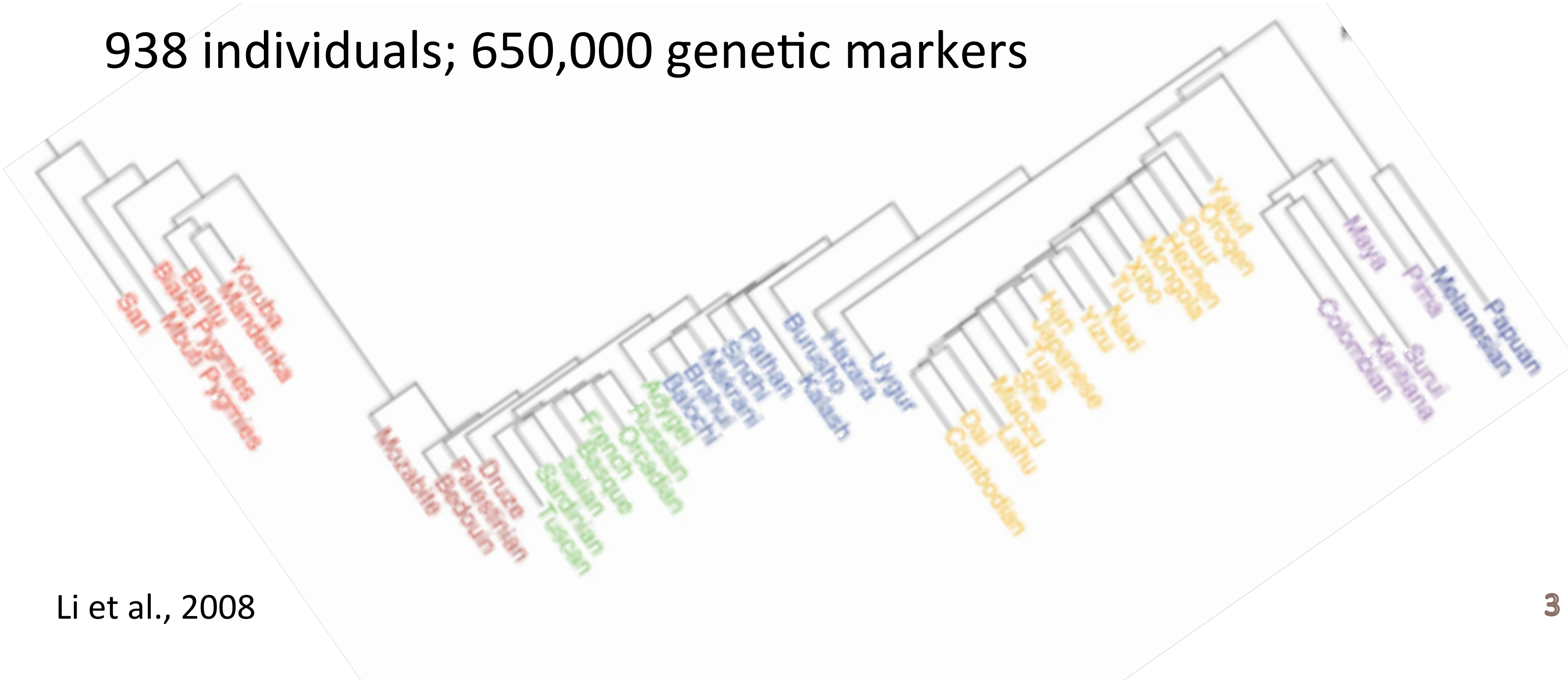
- World is filled with data of increasing size and complexity
- Much of it has underlying low-dimensional structure

http://people.cs.uchicago.edu/ pff/segment

Motivation



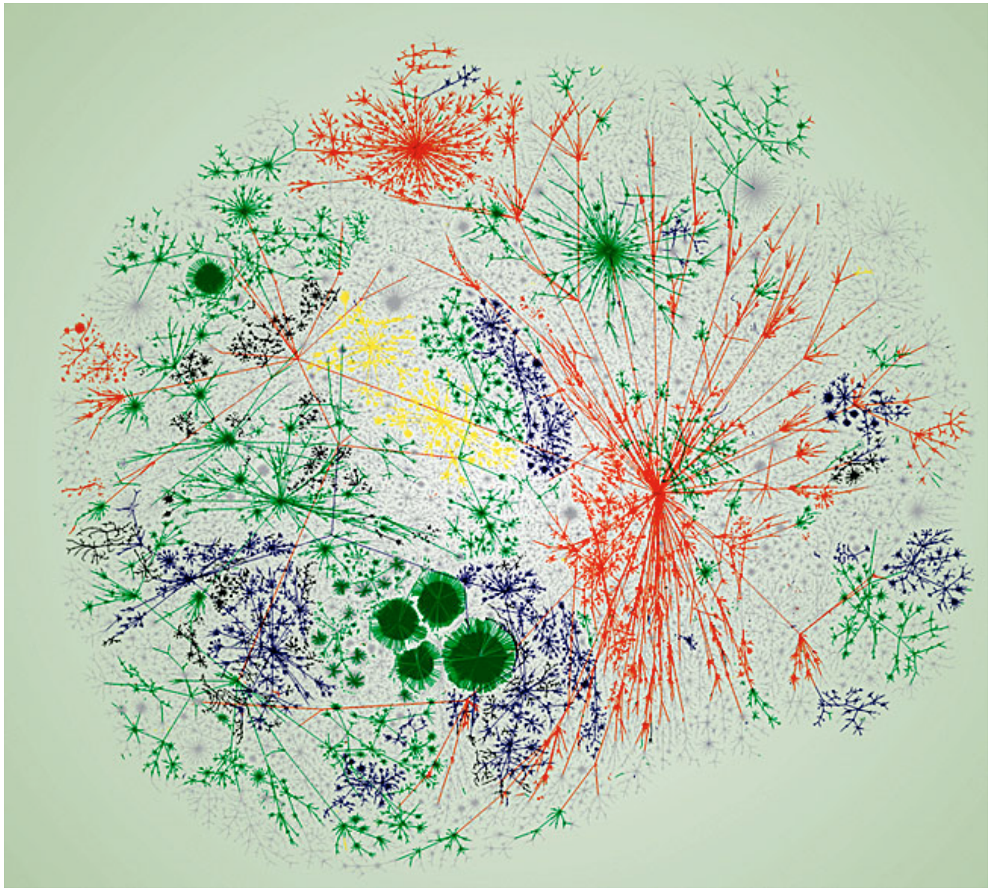
938 individuals; 650,000 genetic markers



Li et al., 2008

Motivation

- World is filled with data of increasing size and complexity
- Much of it has underlying low-dimensional structure



Newman, 2008

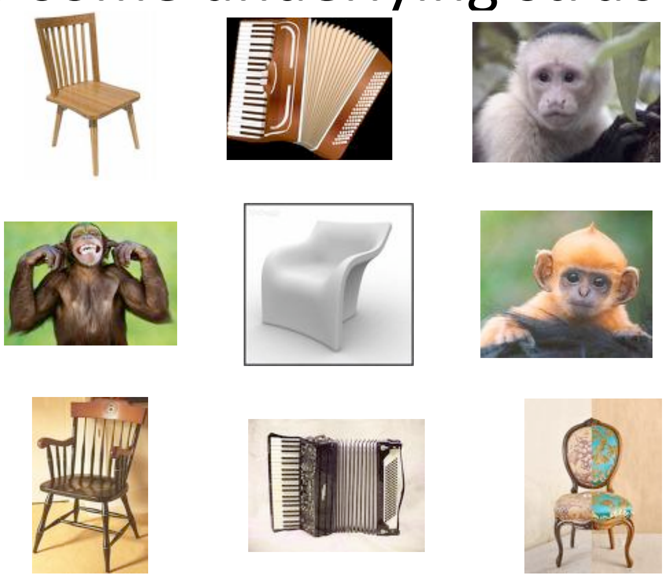
- **How do we uncover the hidden structure in our data?**

Unsupervised learning

- **Supervised learning**
  - Given datapoints $x_1, \ldots, x_n$ with labels $y_1, \ldots, y_n$, learn to predict the label y_{new} associated with each new input x_{new}
  - **Classification:**

Chair  Primate  Which is this? 

- **Unsupervised learning**
  - Given only $x_1, \ldots, x_n$, infer some underlying structure
  - **Clustering:**
    Group these unlabeled images into three classes
  - Evaluation much more challenging!



Why do unsupervised learning?

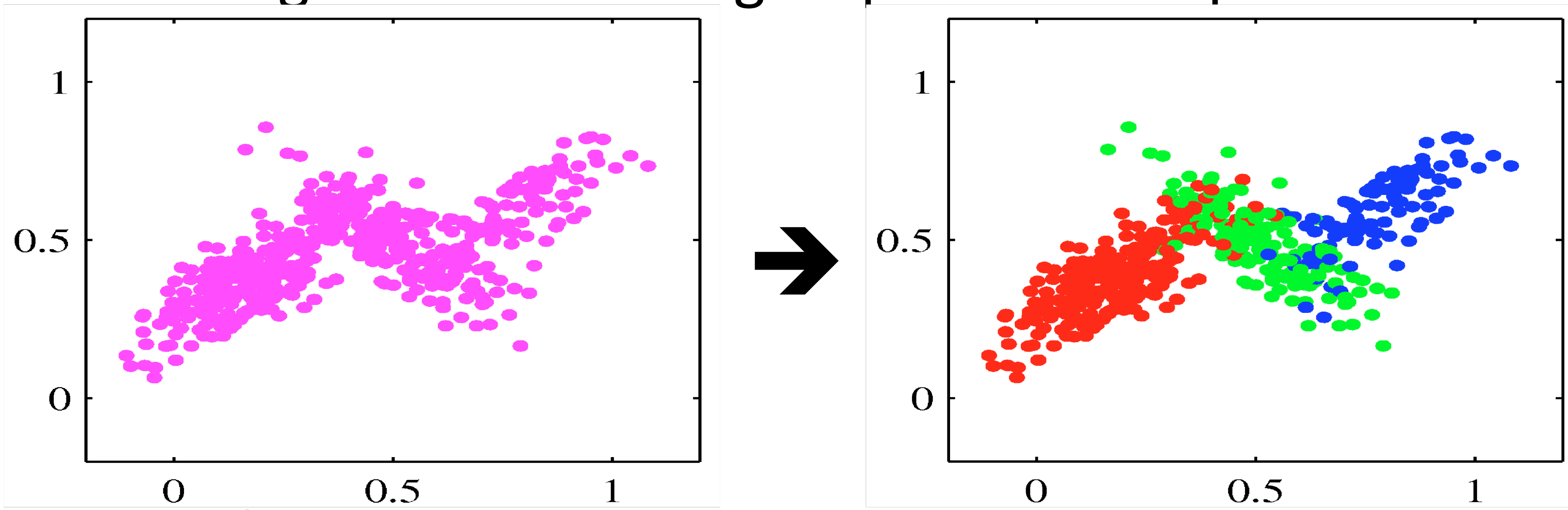
- Labeled data often expensive or difficult to collect; Unlabeled data abundant and cheap
- Develop compressed representations to save storage and computation
- Reduce noise, missingness, irrelevant attributes in high-dimensional data
- Visualization and exploratory data analysis
- As a preprocessing step for supervised learning

This Course

- Survey of unsupervised learning methods, their properties, and their applications
- Classical paradigms
  I. Clustering and latent class methods
  II. Dimensionality reduction and latent feature methods
- Modern topics (based on time and interest)
  - Unsupervised learning with missing data
  - Sparse / interpretable unsupervised learning
  - Nonnegative matrix factorization, Document topic modeling
  - Subspace clustering
  - Method of moments for latent variable models
  - Unsupervised deep learning

Clustering

- **Goal:** Segment data into groups of similar points



- **Examples**
  - Segment pixels in an image by object
  - Group network participants into communities
  - Identify cancer subtypes from gene expression patterns
- Will discuss many approaches to clustering in Stats306B
  - Begin with one of the simplest and most popular: ***k*-means**

k-means

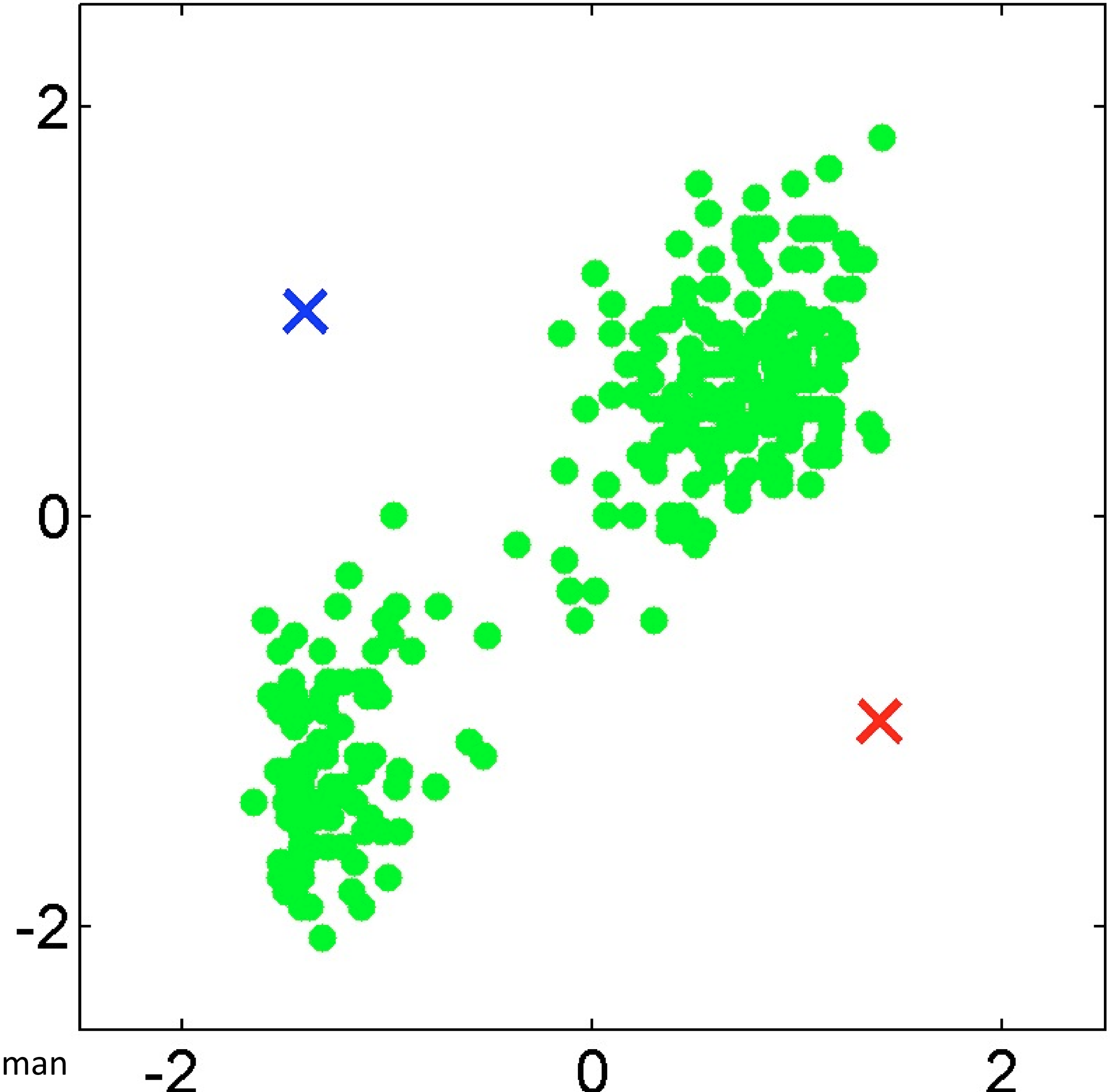
- **Summary:** Assign each datapoint to one of k clusters so that on average each point is close to its cluster mean

- **Notation**
  - Datapoint $x_i \in \mathbb{R}^p$
  - Cluster mean $m_j \in \mathbb{R}^p$
  - Cluster assigment $z_i \in \{1, \ldots, k\}$
- **Objective:** $J(z_{1:n}, m_{1:k}) = \sum_{i=1}^{n} ||x_i - m_{z_i}||_2^2$
- **Goal:** Minimize J over $z_{1:n}$ and $m_{1:k}$

k-means

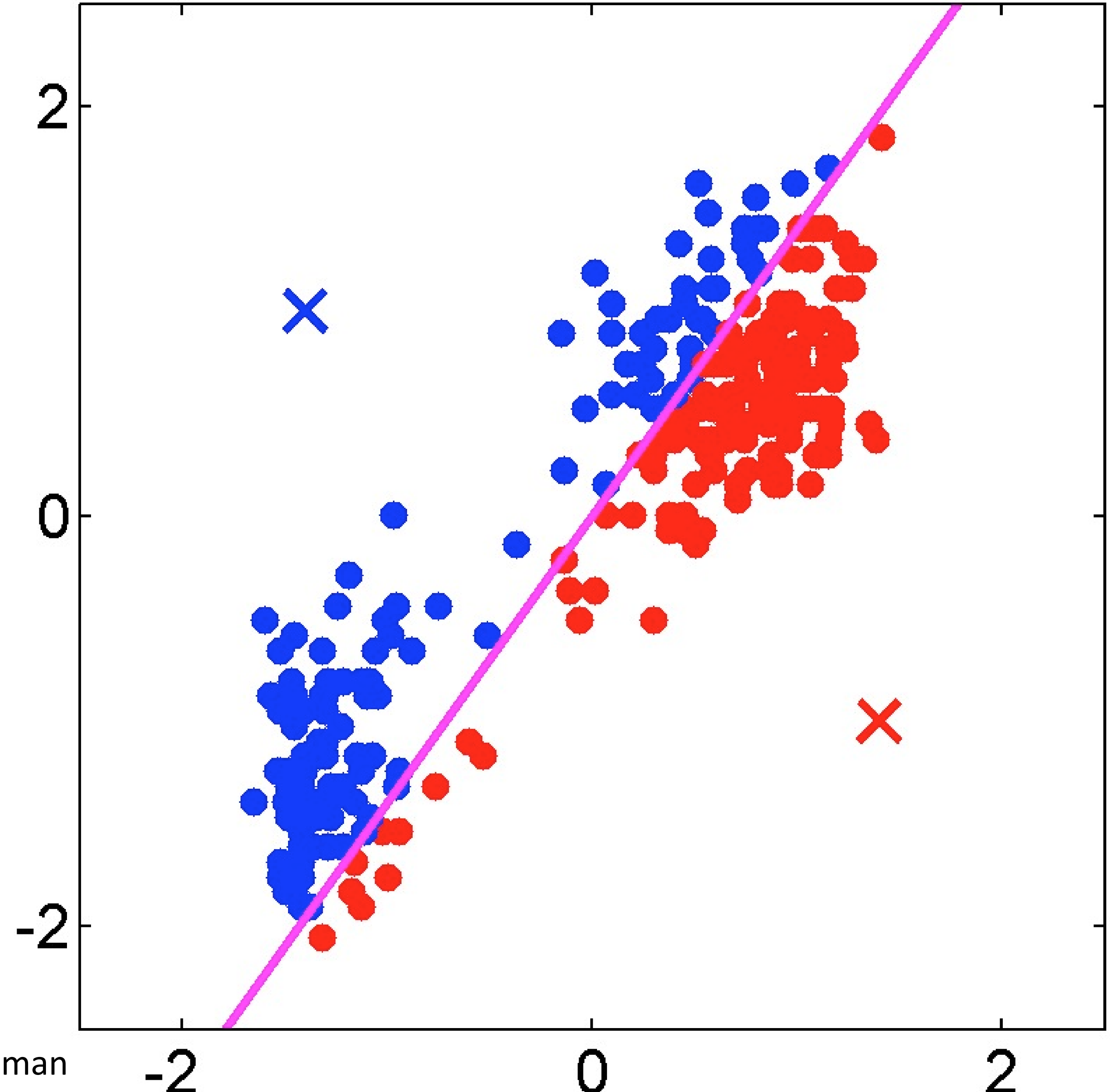
- **Goal:** Minimize $J(z_{1:n}, m_{1:k}) = \sum_{i=1}^{n} ||x_i - m_{z_i}||_2^2$ over $z_{1:n}$ and $m_{1:k}$
  - Datapoint $x_i \in \mathbb{R}^p$
  - Cluster mean $m_k \in \mathbb{R}^p$
  - Cluster assigment $z_i \in \{1, \ldots, k\}$
- **Standard k-means algorithm / Lloyd's algorithm**
  - Initialize cluster means arbitrarily (e.g., sample from datapoints)
  - Alternate until convergence
    - ∗ Update cluster assignments: $z_{1:n} \leftarrow \arg\min_{z_{1:n}} J(z_{1:n}, m_{1:k})$
      - · i.e., assign each point to the cluster with closest mean
    - ∗ Update cluster means: $m_{1:k} \leftarrow \arg\min_{m_{1:k}} J(z_{1:n}, m_{1:k})$
      - · i.e., $m_j = \frac{\sum_{i=1}^{n} \mathbb{I}(z_i = j) x_i}{\sum_{i=1}^{n} \mathbb{I}(z_i = j)}$, the mean of points in cluster j

Example: 2-means, Lloyd's algorithm



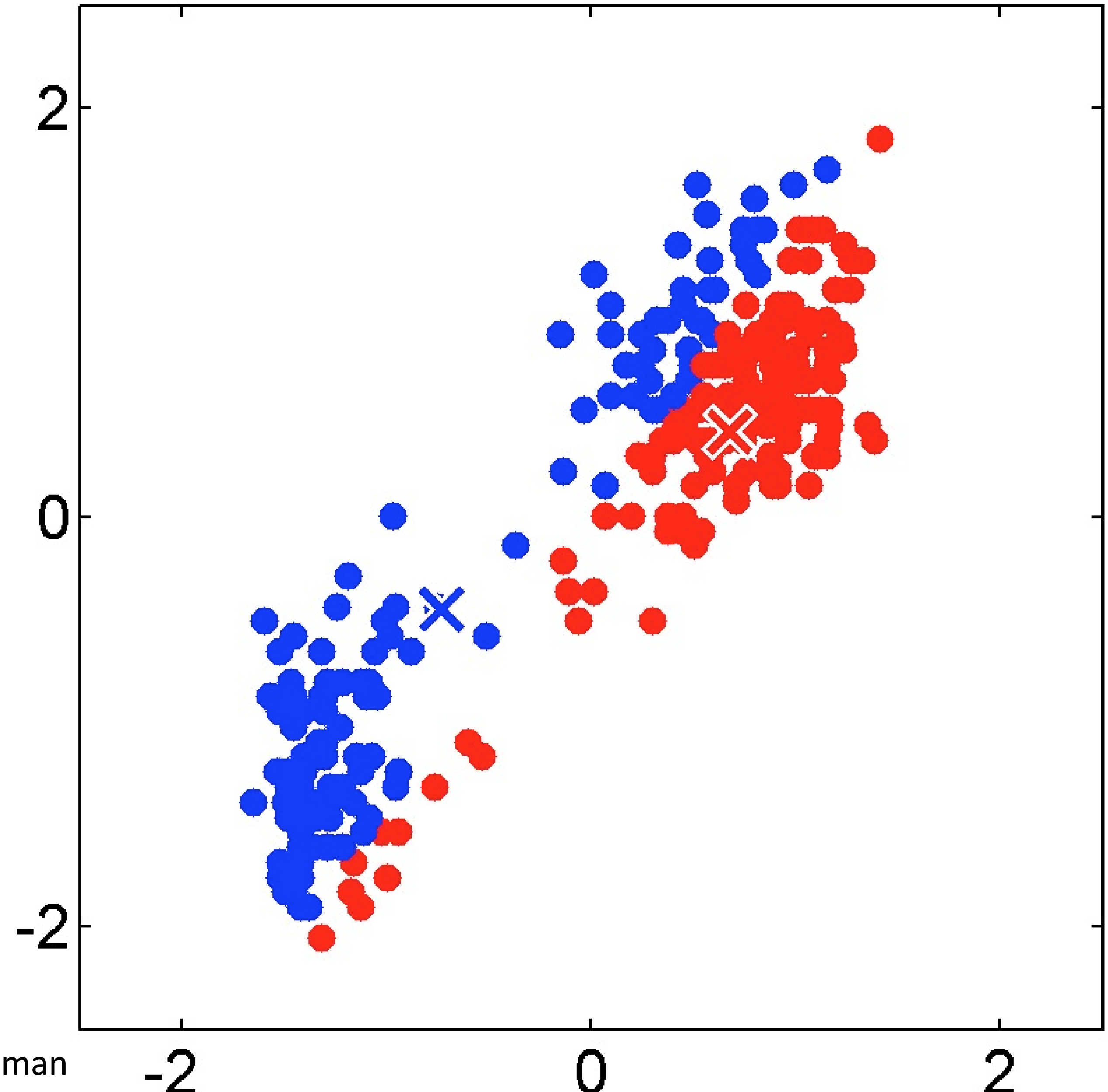
Courtesy of
Sriram Sankararaman

Example: 2-means, Lloyd's algorithm



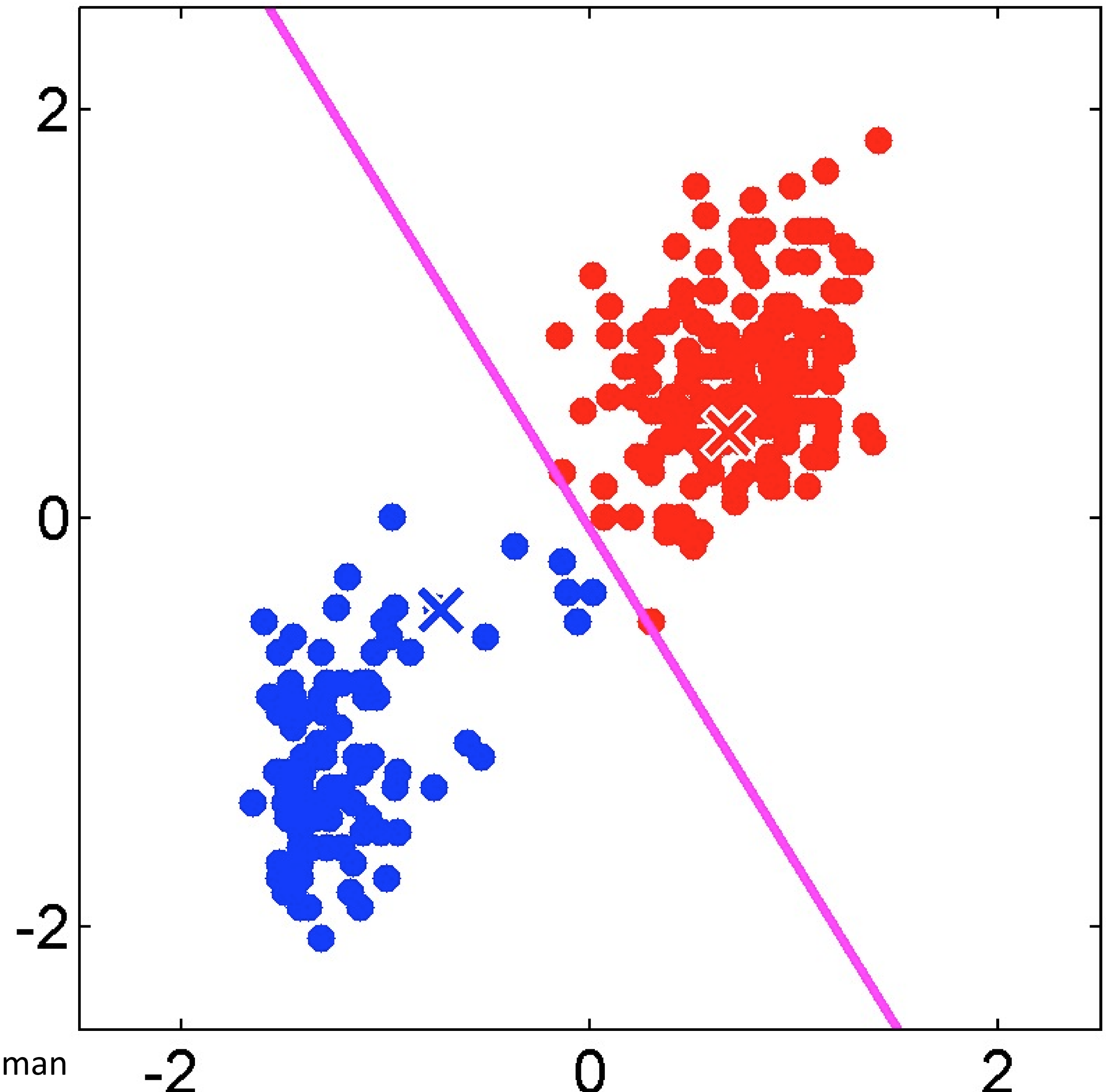
Courtesy of
Sriram Sankararaman

Example: 2-means, Lloyd's algorithm



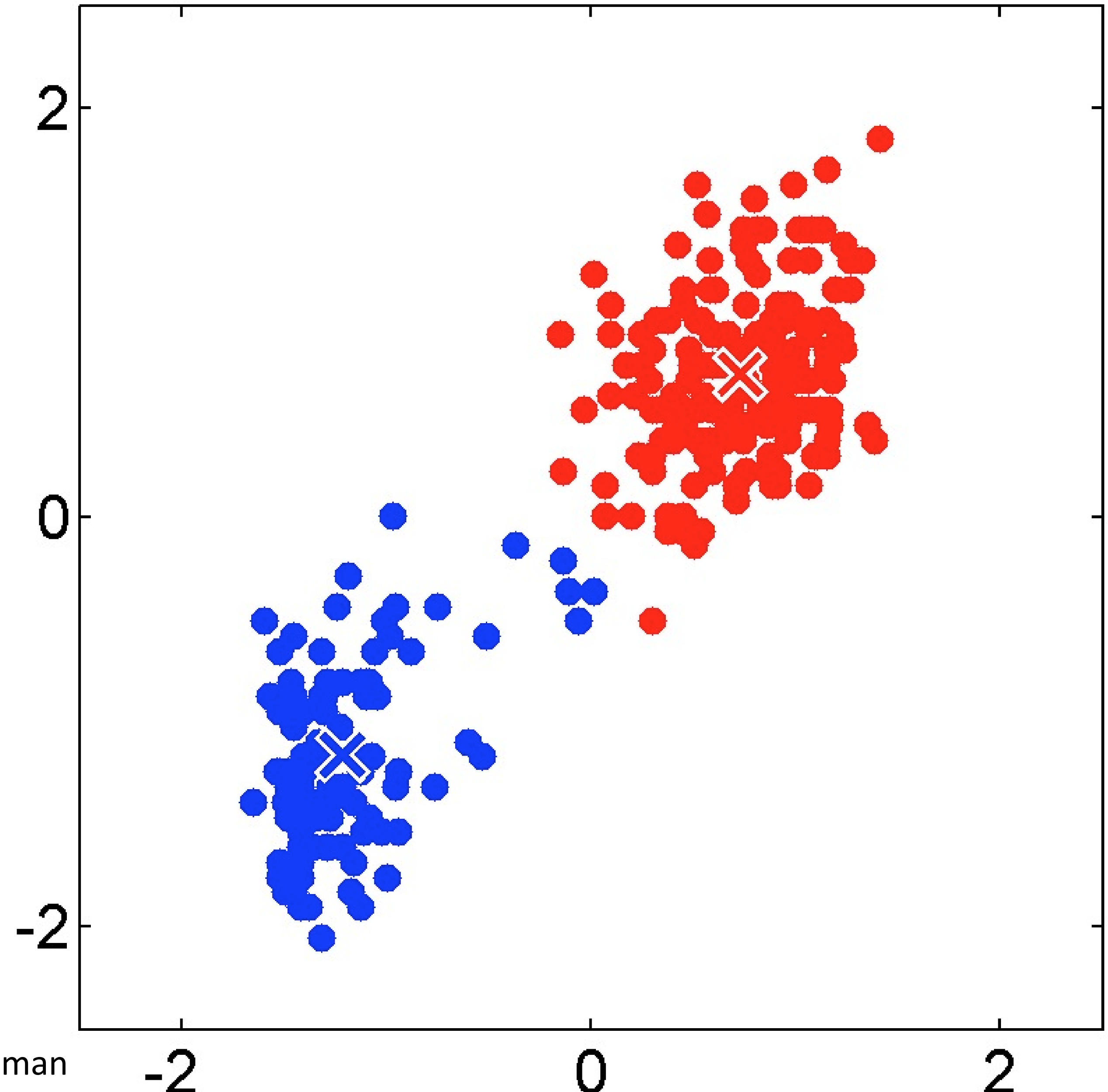
Courtesy of
Sriram Sankararaman

Example: 2-means, Lloyd's algorithm



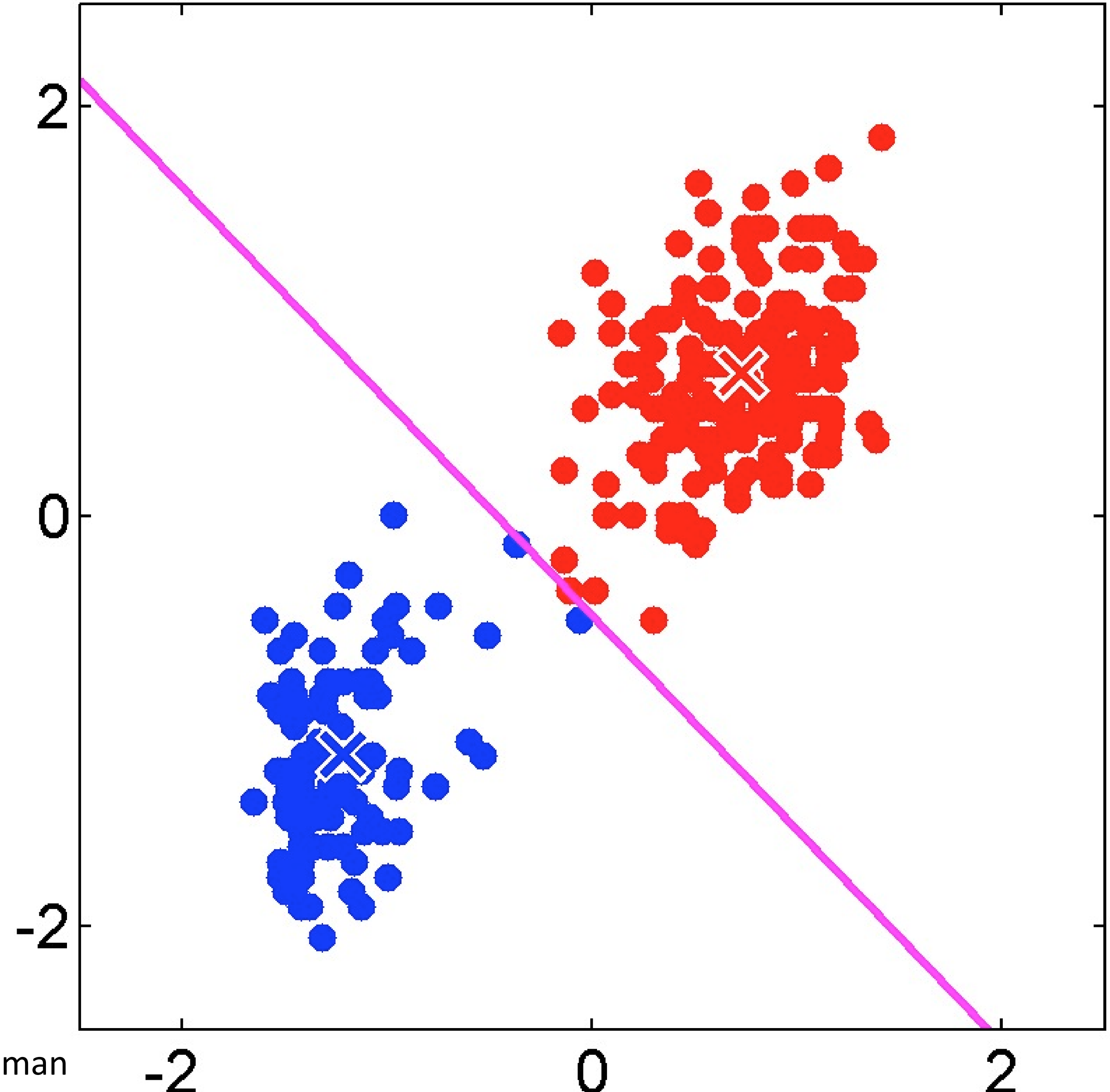
Courtesy of
Sriram Sankararaman

Example: 2-means, Lloyd’s algorithm



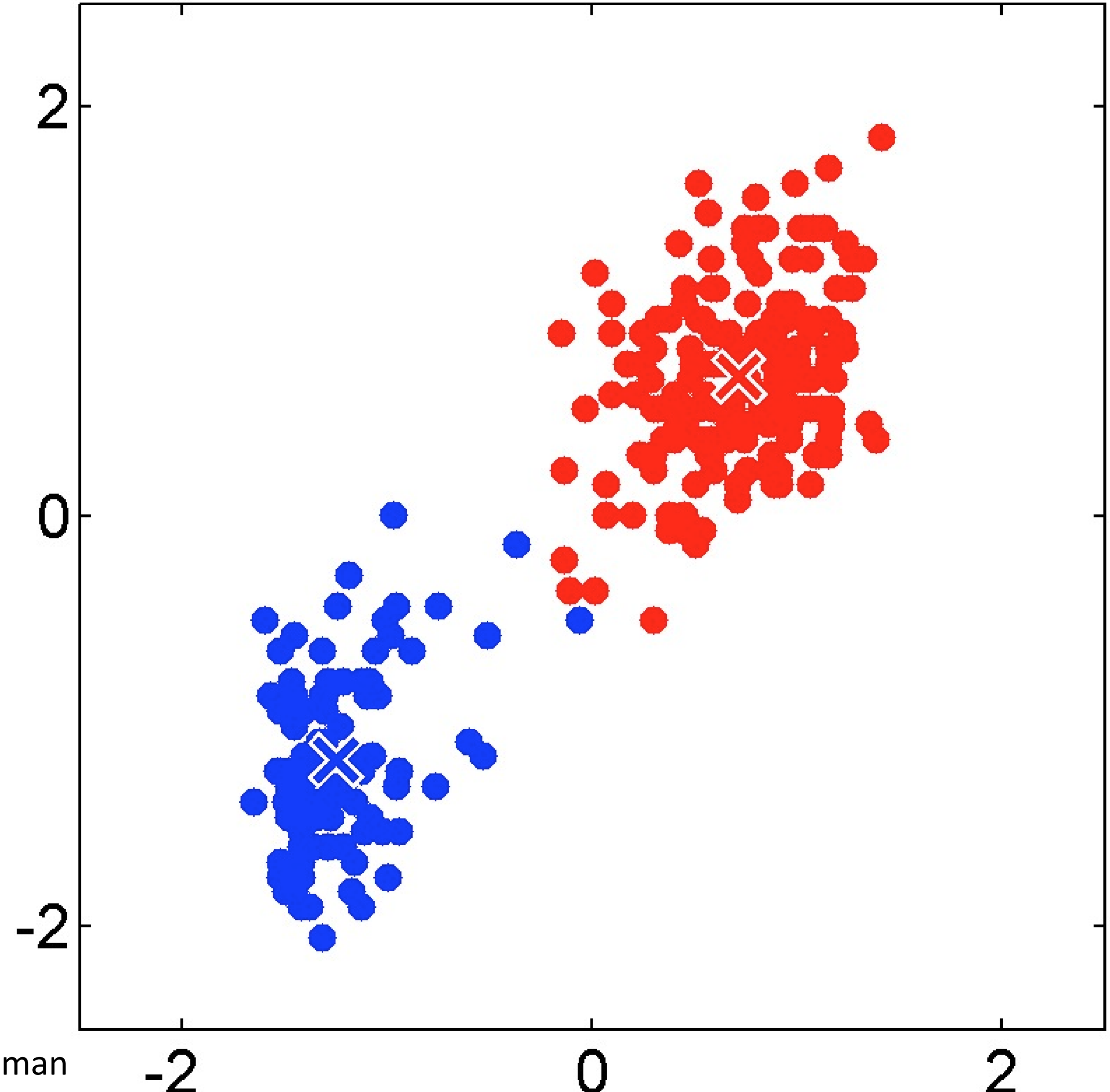
Courtesy of
Sriram Sankararaman

Example: 2-means, Lloyd's algorithm



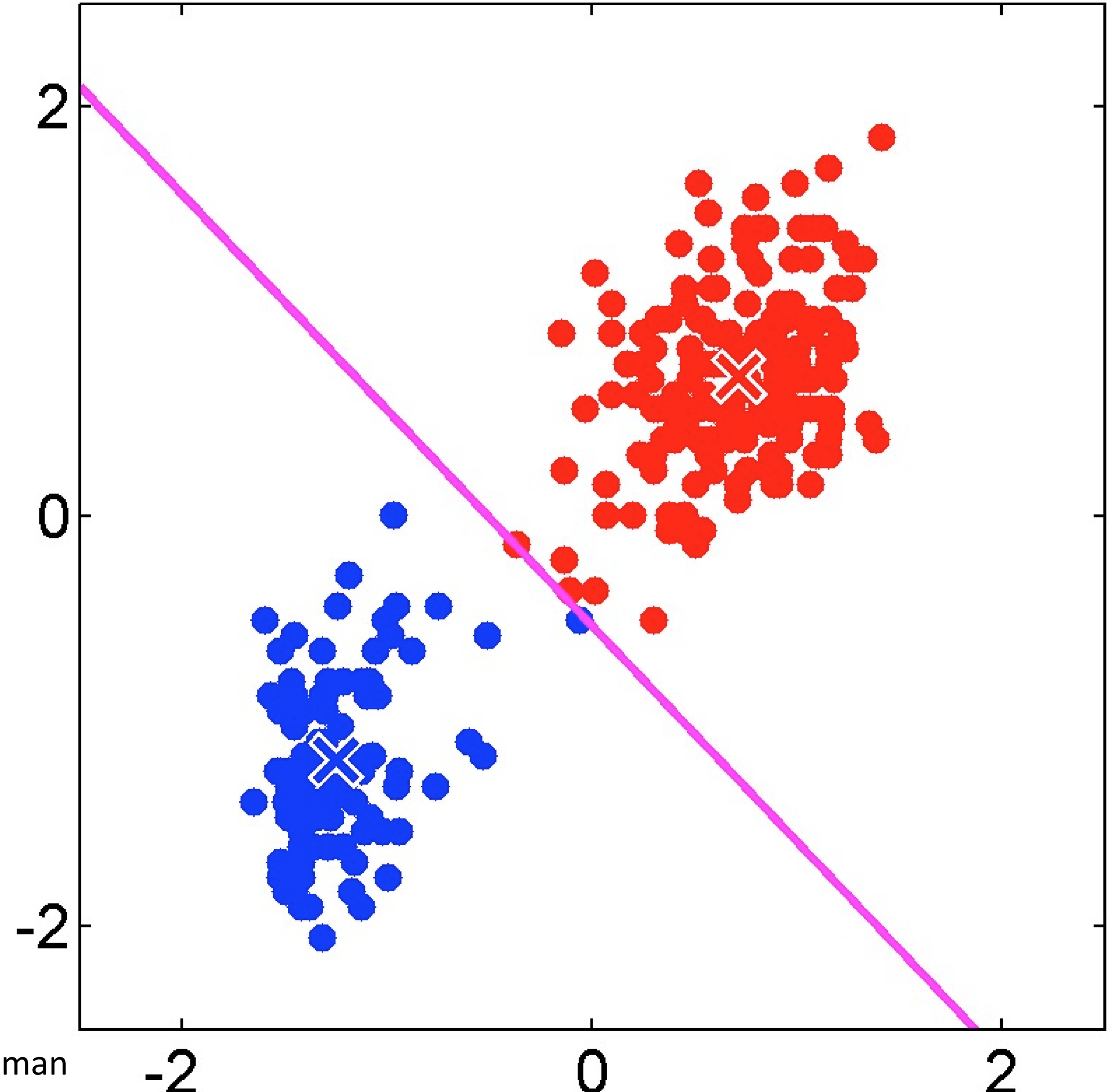
Courtesy of
Sriram Sankararaman

Example: 2-means, Lloyd's algorithm



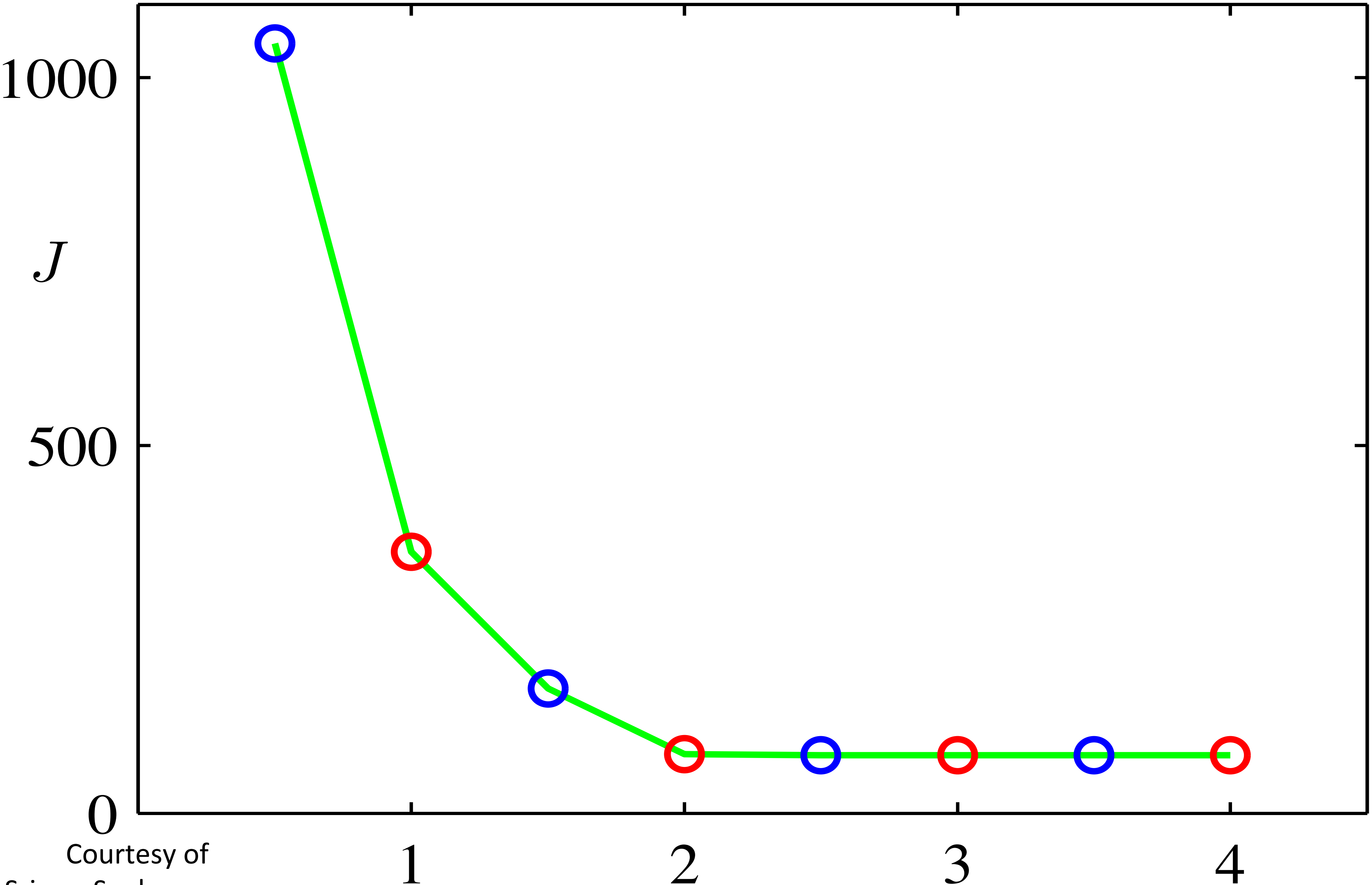
Courtesy of
Sriram Sankararaman

Example: 2-means, Lloyd's algorithm



Courtesy of
Sriram Sankararaman

Objective function J after each iteration

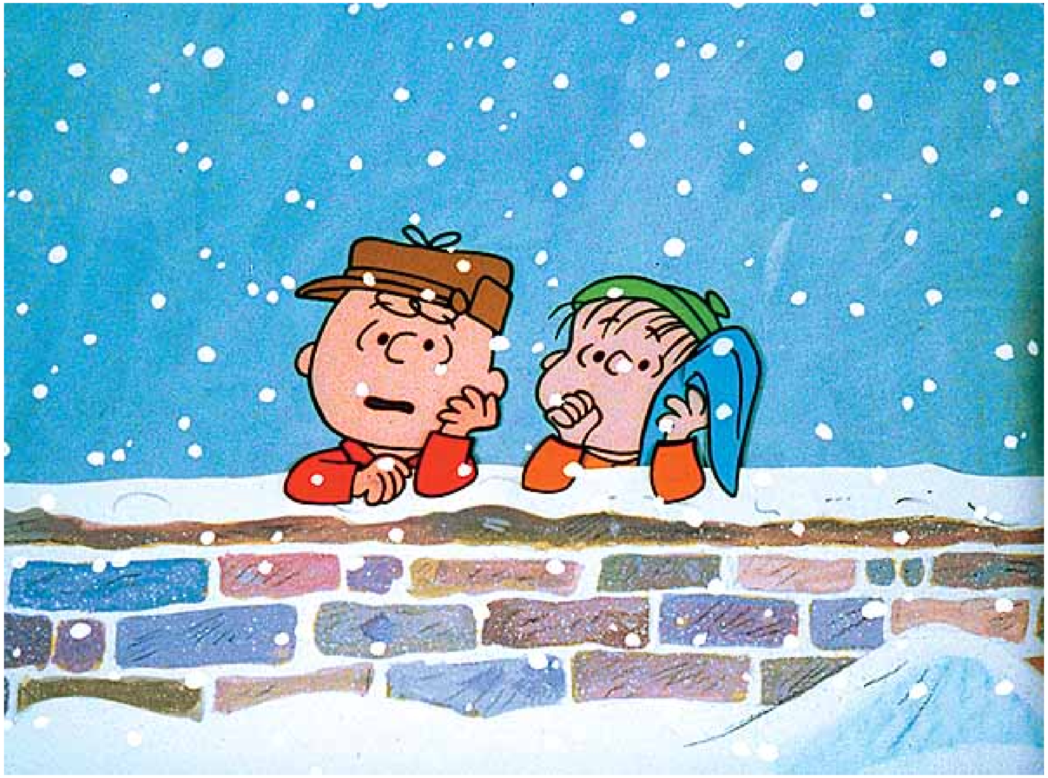


Courtesy of
Sriram Sankararaman

Does Lloyd's algorithm always converge?

- The objective *J* always converges
  - Lloyd's algorithm is a coordinate descent procedure
  - Each step monotonically decreases objective
  - Only finite number of partitions of data, so objective must converge in finite number of steps
- Technically, algorithm could cycle if ties arise (i.e., if multiple centroids equidistant from a point)
  - Minor problem: avoid by breaking ties in a consistent fashion (e.g., always assign point to "smallest" centroid under some total ordering of vectors)

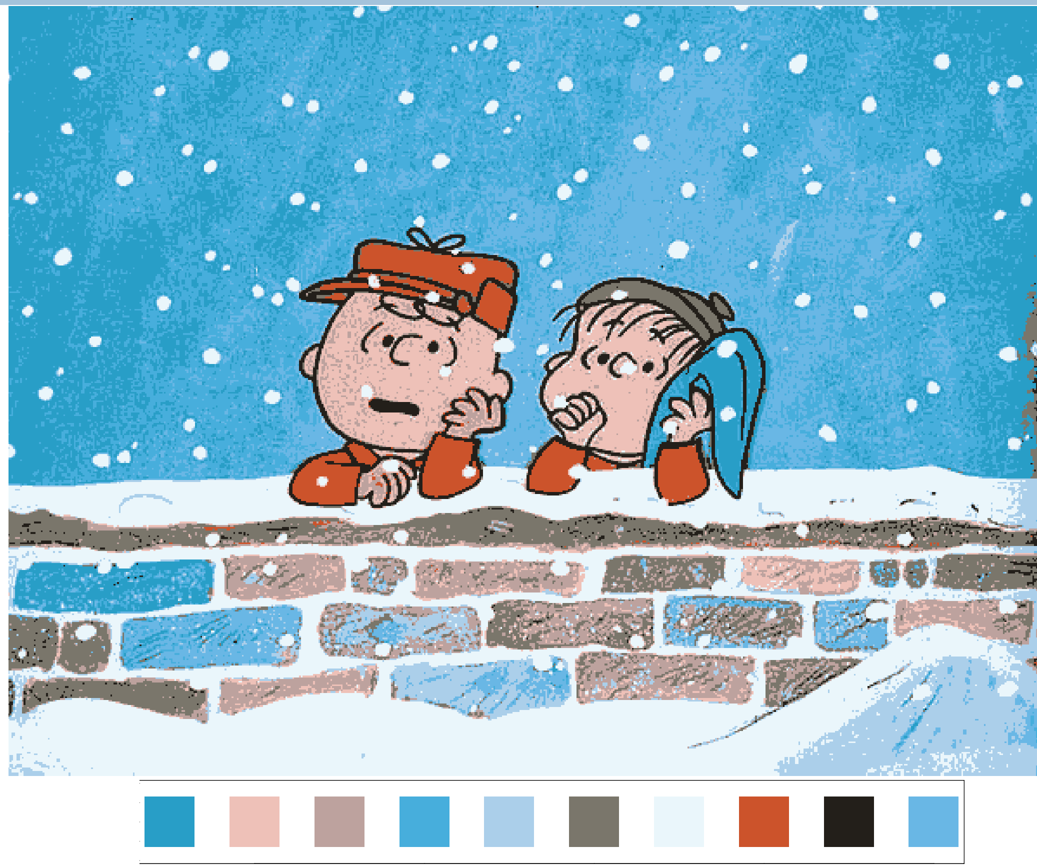
Image compression



Credit:
Dave Blei

- Pixel is vector of red, green, and blue values in {0,...,255}
- 2048 × 1536 image is a dataset of 3.1 million vectors, each requiring 24 bits of storage
- Let's compress by clustering pixels with *k*-means

Vector quantization



Credit:
Dave Blei

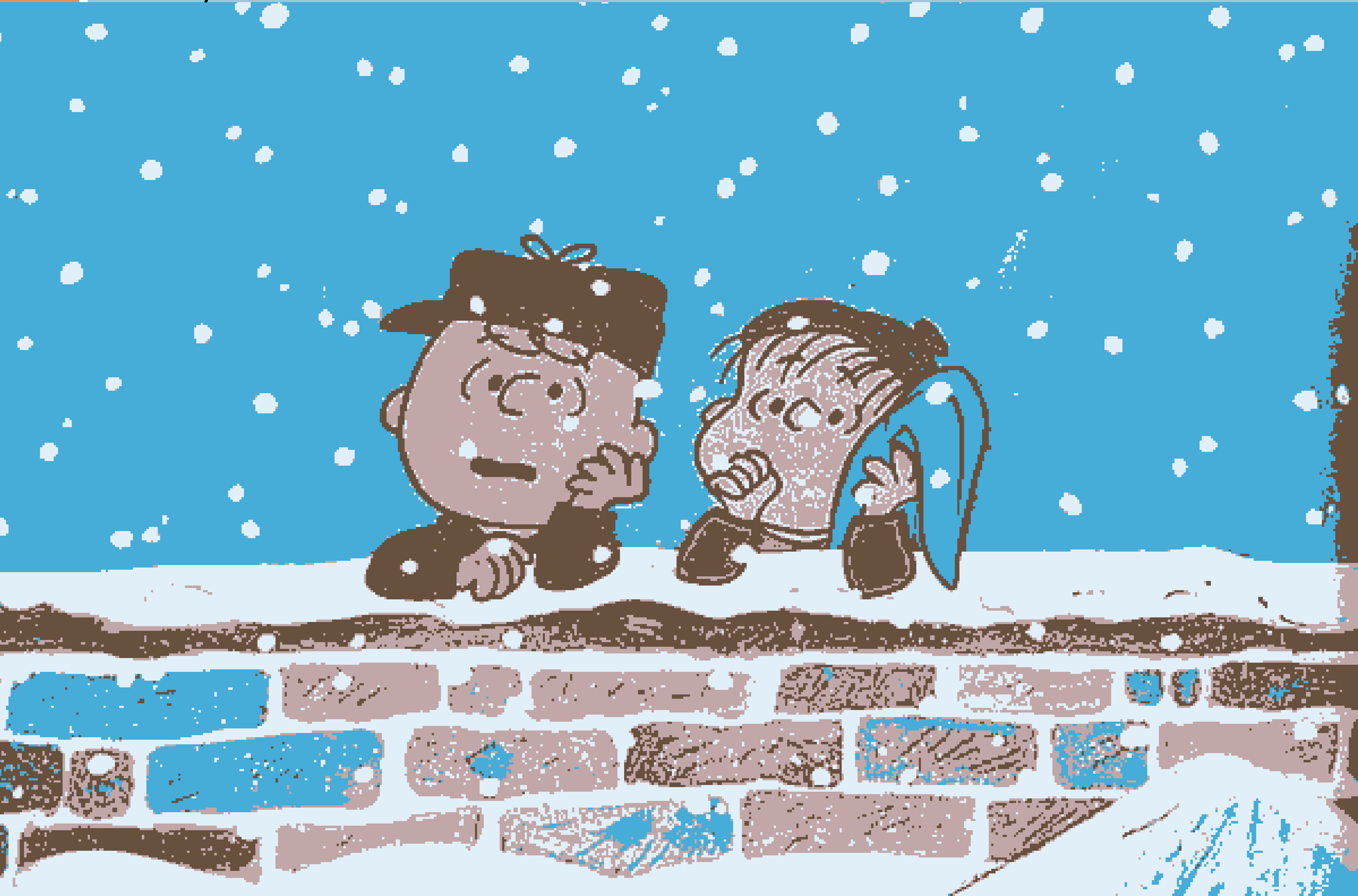
- Recovered *k* means called *codebook*
  - Each *codeword* (after rounding) corresponds to a color
- Compression: replace each image pixel by its codeword
- $\log_2(k)$ bits instead of 24 per pixel (plus small overhead)

Peanuts vector quantization: 2 means

Courtesy of Dave Blei

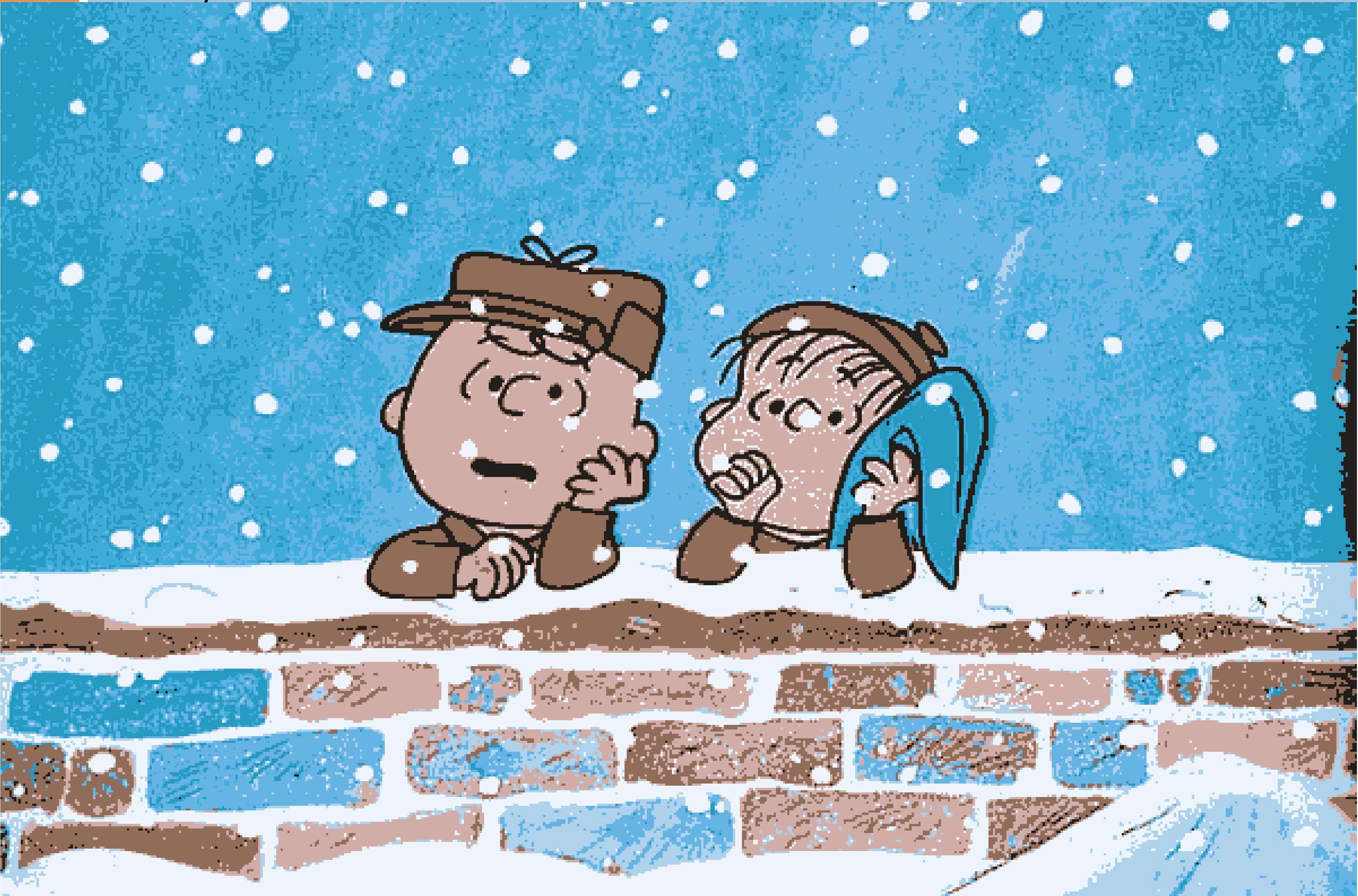
Peanuts vector quantization: 4 means

Courtesy of Dave Blei



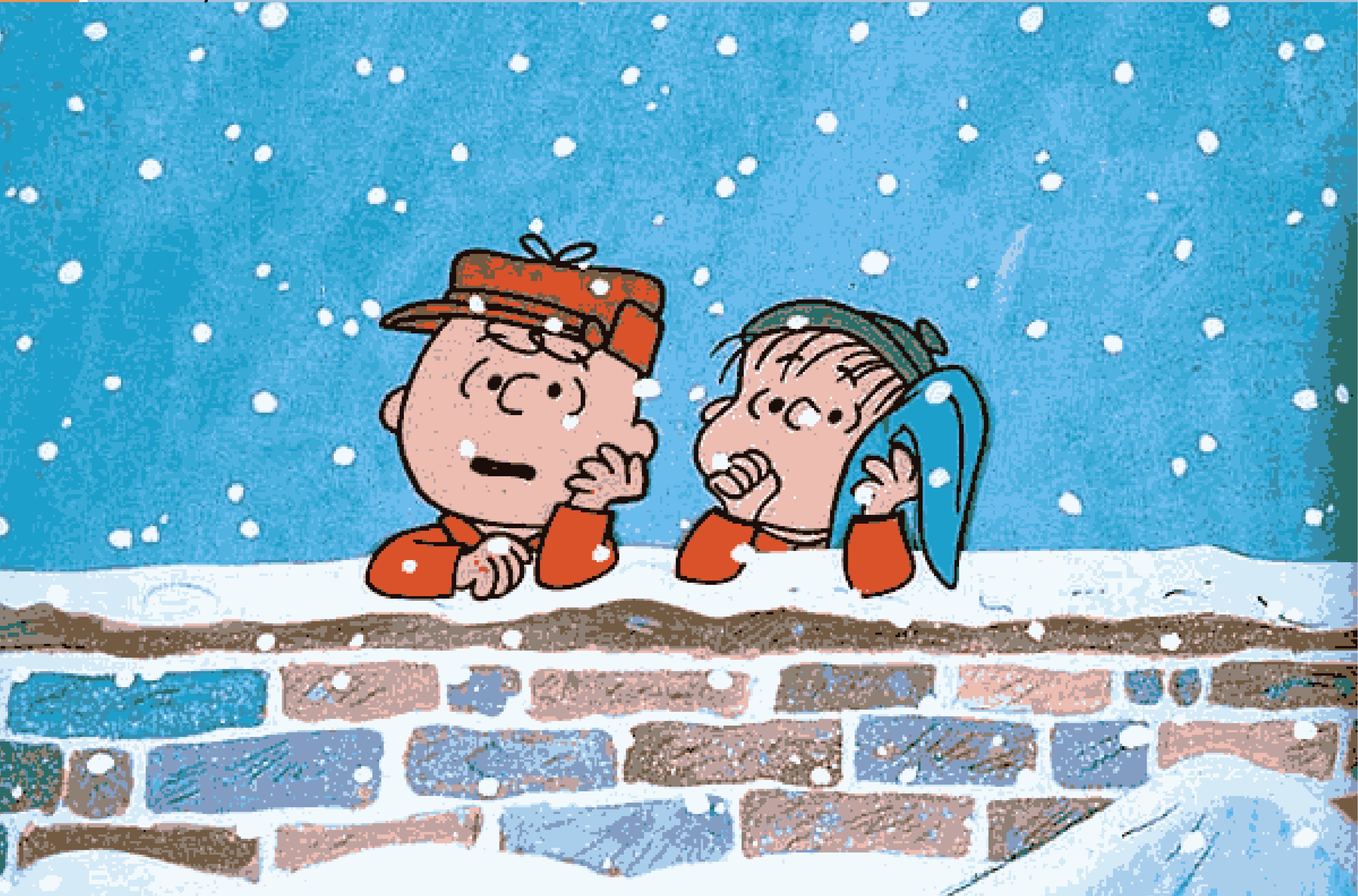
Peanuts vector quantization: 8 means

Courtesy of Dave Blei



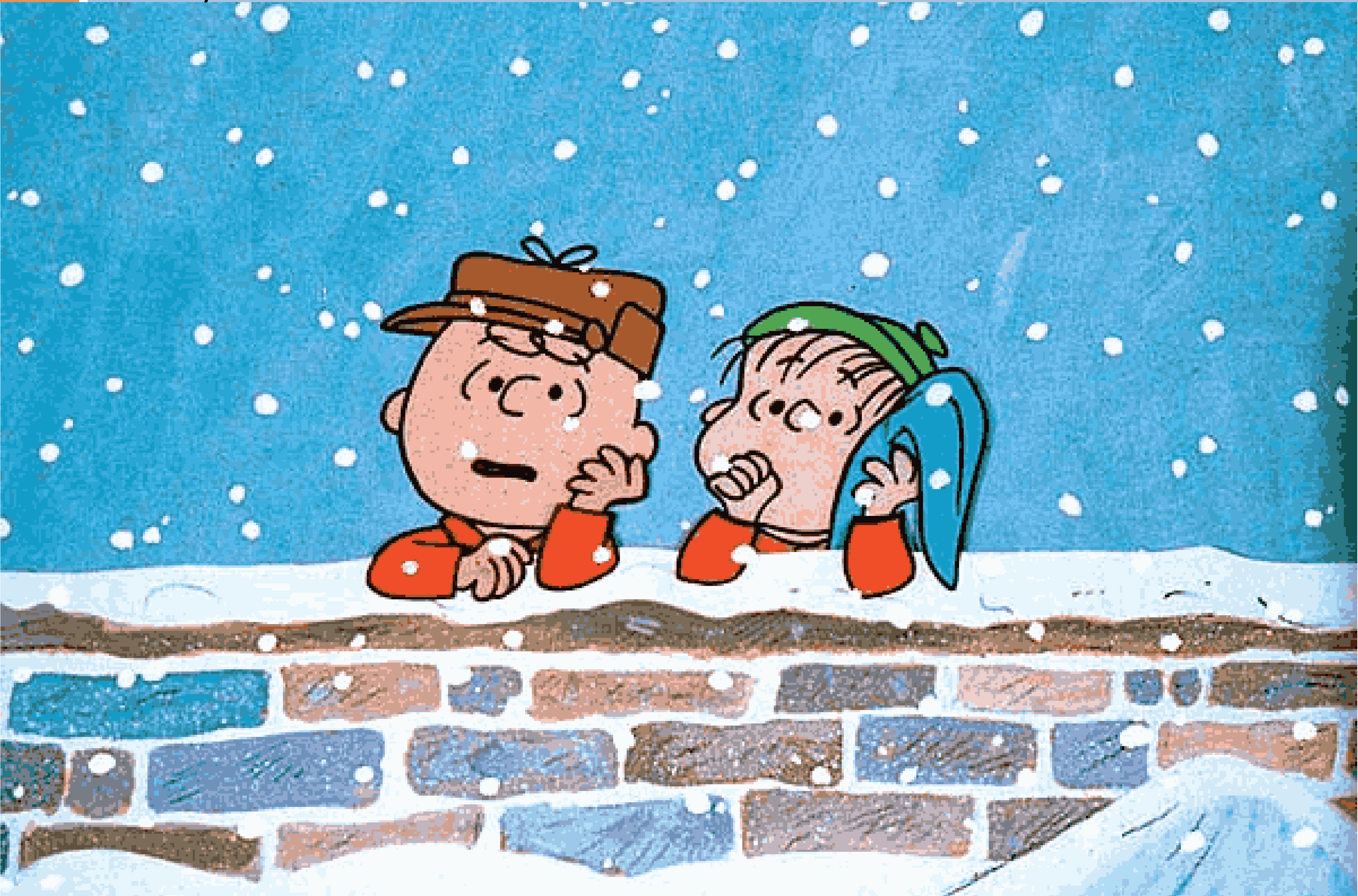
Peanuts Vector Quantization: 16 means

Courtesy of Dave Blei



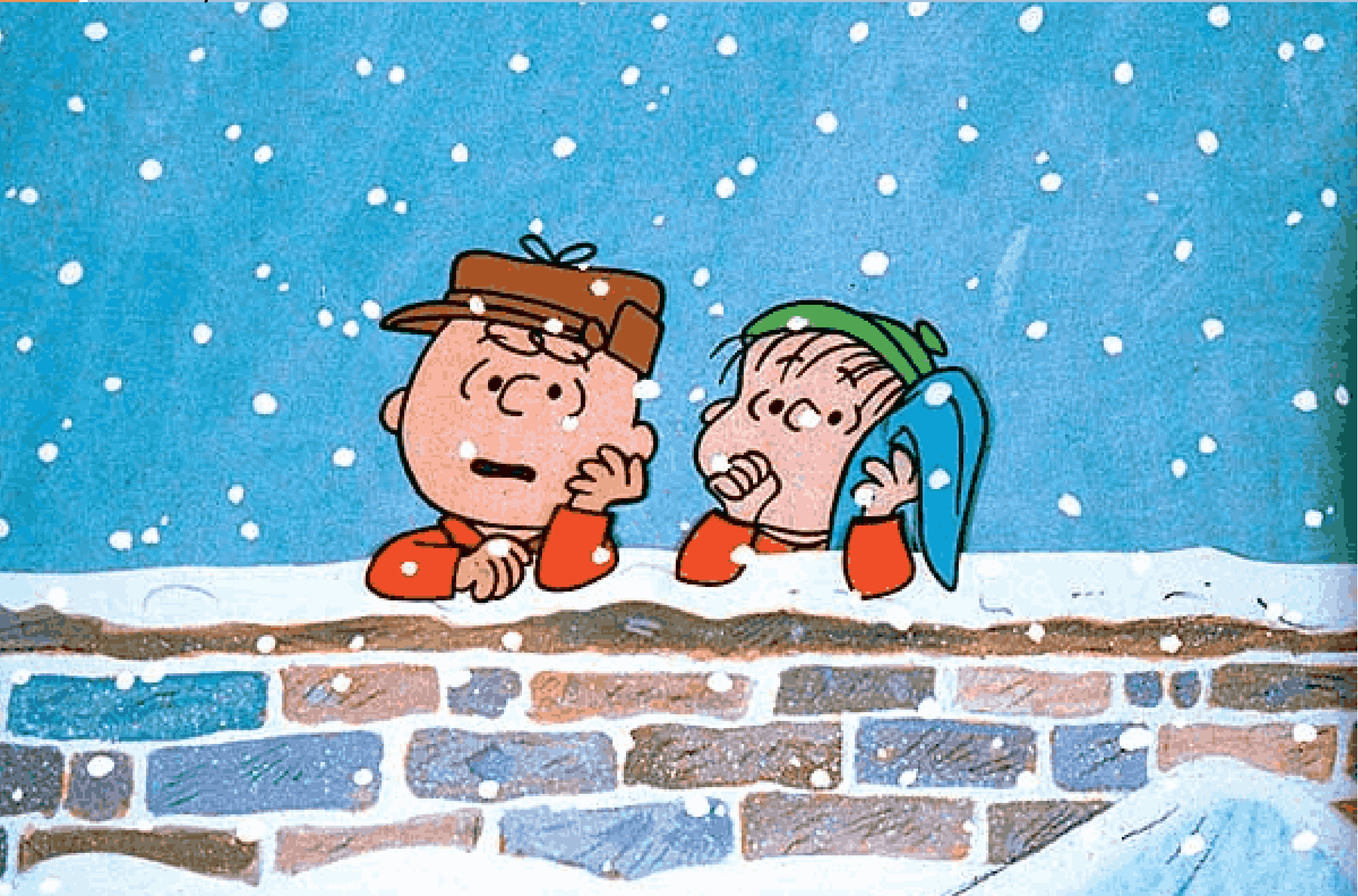
Peanuts Vector Quantization: 32 means

Courtesy of Dave Blei



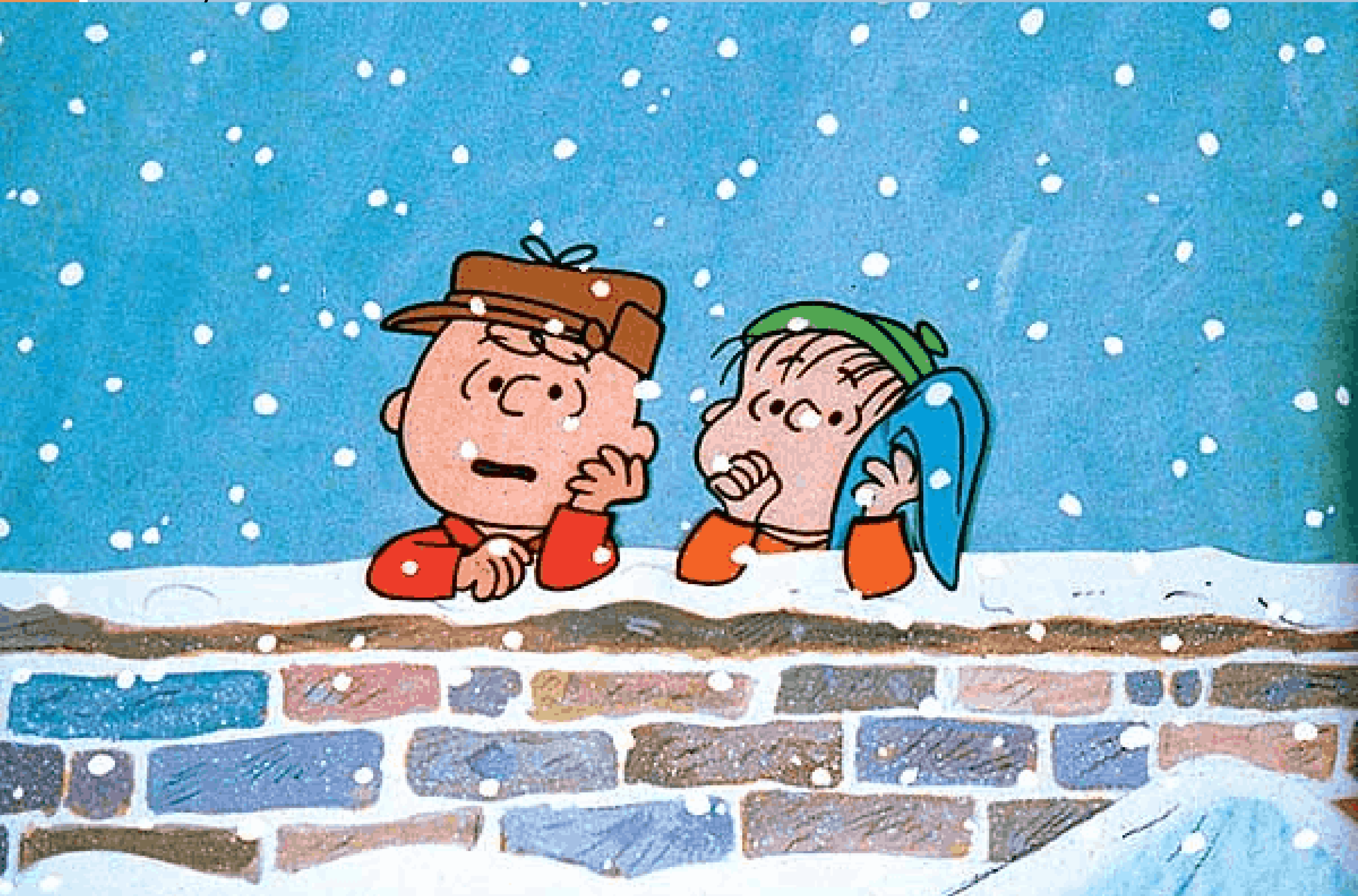
Peanuts vector quantization: 64 means

Courtesy of Dave Blei



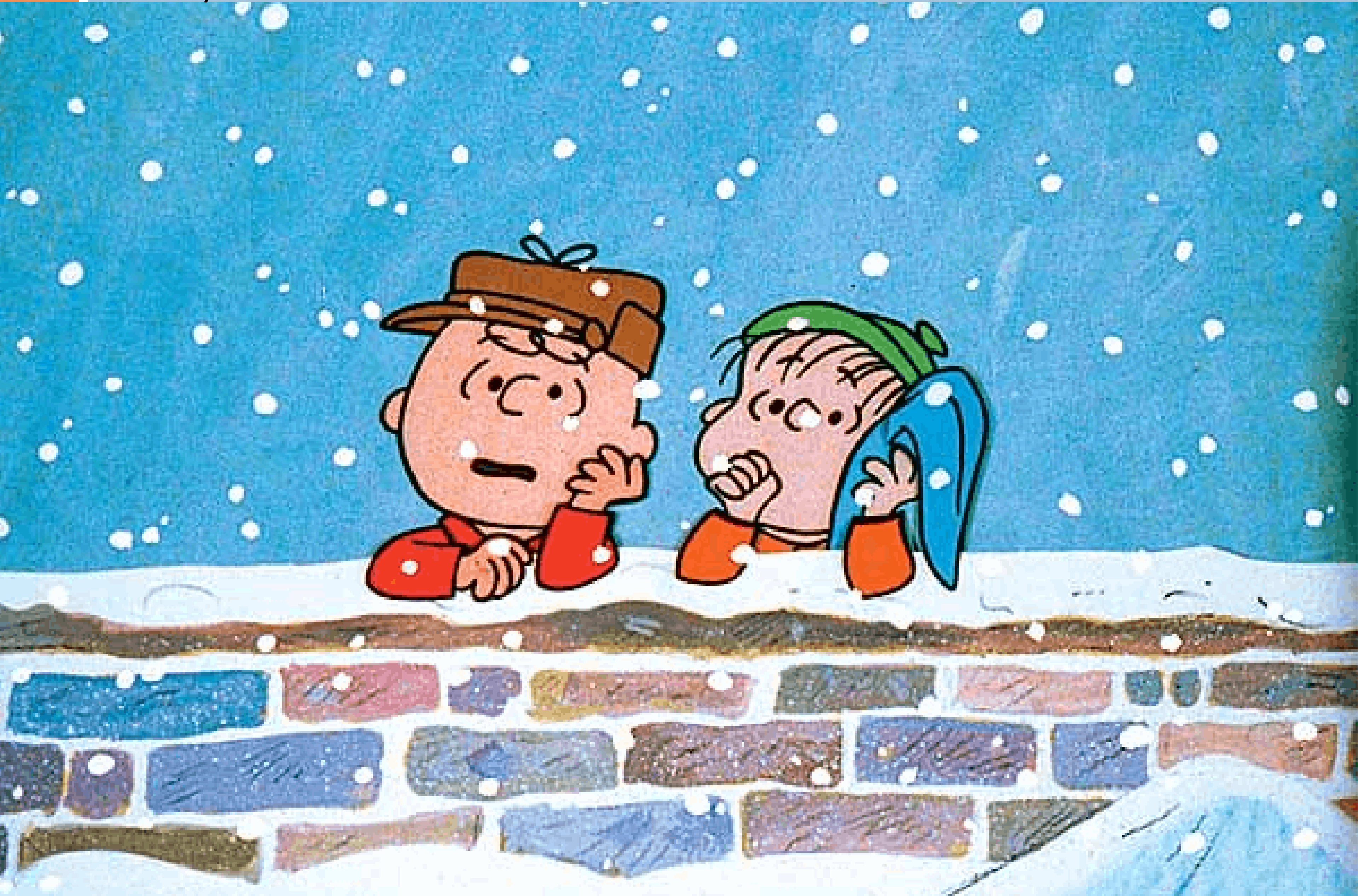
Peanuts vector quantization: 128 means

Courtesy of Dave Blei



Peanuts vector quantization: 256 means

Courtesy of Dave Blei



k-means: Practical considerations

1. Squared Euclidean objective restrictive

$$J(z_{1:n}, m_{1:k}) = \sum_{i=1}^{n} ||x_i - m_{z_i}||_2^2$$

   - Inappropriate for non-quantitative (e.g., categorical) features
   - Euclidean distance
     - Sensitive to outliers
     - Ill-suited for features with very different scales / importances
2. NP-hard optimization problem
   - Lloyd's algorithm **usually** finds suboptimal solutions
   - Many random restarts often needed for good performance
3. Must choose *k*
4. Running time: # features x # datapoints x *k* per iteration
   - Orders of magnitude reductions using space-partitioning data structures like *k*d-trees (e.g., Kanungo et al., 2002, optional reading)

Beyond Euclidean distance

- **Issue:** Squared Euclidean distance in *k*-means
- **Idea:** Minimize $J_d(z_{1:n}, m_{1:k}) = \sum_{i=1}^{n} d(x_i, m_{z_i})$
  - Arbitrary dissimilarity / discrepancy measure *d(x, m)*
  - Optimize via coordinate descent as in Lloyd's algorithm
    - Update cluster assignments: $z_{1:n} \leftarrow \arg\min_{z_{1:n}} J_d(z_{1:n}, m_{1:k})$
    - Update cluster representatives: $m_{1:k} \leftarrow \arg\min_{m_{1:k}} J_d(z_{1:n}, m_{1:k})$
  - Pro: Applies to all data types and dissimilarity measures
  - Con: Updating cluster representatives $m_{1:k}$ may be expensive
- ***k*-medoids algorithm**
  - Minimize J_d above but constrain each cluster representative to be a datapoint, i.e. $m_j \in \{x_1, \dots, x_n\}$
  - Pro: Don't need to store datapoints, only pairwise discrepancies $d(x_i, x_j)$

k-means++

Arthur and Vassilvitskii, 2008 (optional reading)

- **Issues:** Lloyd's algorithm suboptimal, random restarts
- ***k*-means++:** Improves initialization of Lloyd's algorithm
  - Choose first center m_1 uniformly at random from $\{x_1, \ldots, x_n\}$
  - For $j = 2, \ldots, k$:
    - Let $D(x)$ = Euclidean distance to closest center previously chosen
    - Choose $m_j = x_i$ with probability proportional to $D(x_i)^2$
  - Run Lloyd's algorithm with this initialization
- **Thm:** E[objective after *k*-means++] ≤ 8(ln(k) + 2) optimal
- In practice: more accurate and faster than *k*-means alone

	Average ϕ		Minimum ϕ		Average T	
k	k-means	k-means++	k-means	k-means++	k-means	k-means++
10	$3.387 \cdot 10^8$	93.37%	$3.206 \cdot 10^8$	94.40%	63.94	44.49%
25	$3.149 \cdot 10^8$	99.20%	$3.100 \cdot 10^8$	99.32%	257.34	49.19%
50	$3.079 \cdot 10^8$	99.84%	$3.076 \cdot 10^8$	99.87%	917.00	66.70%

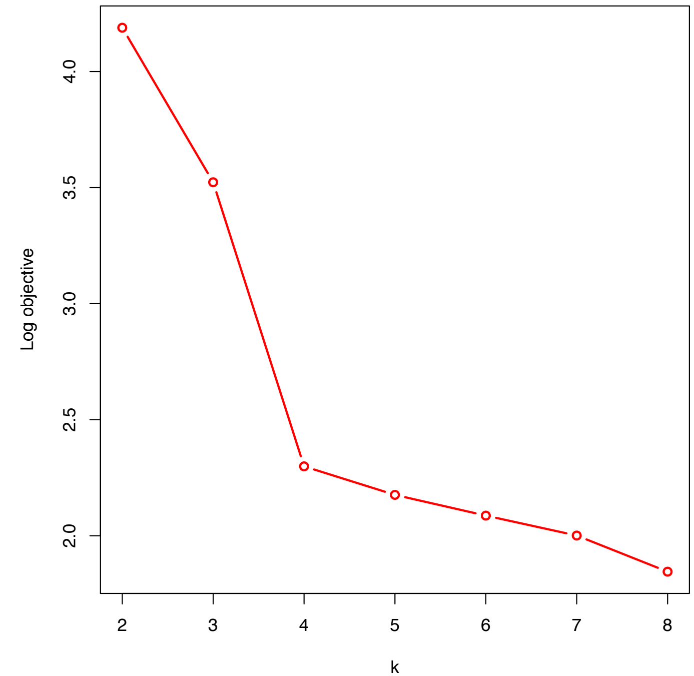
Table 3: Experimental results on the *Intrusion* dataset (n = 494019, d = 35). For `k-means`, we list the actual potential and time in seconds. For `k-means++`, we list the percentage *improvement* over `k-means`.

Choosing the number of clusters *k*

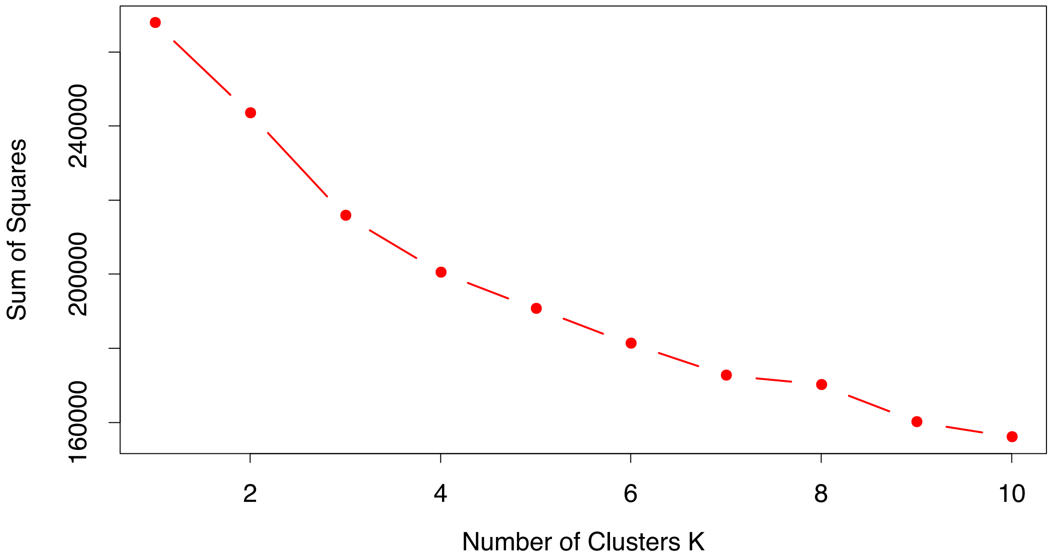
- Some applications determine *k*
  - Target compression level in vector quantization
  - Funds to develop three new Cheerios flavors
- How do we pick *k* otherwise?
  - Minimum *k*-means objective shrinks as *k* grows: not helpful
  - Evaluate fit of learned centers on **held-out data**?
    - Problem: Held-out objective also tends to decrease with *k* !
  - No agreed-upon solution but many alternatives…
  - **Stability:** Cluster randomly subsampled or perturbed datasets and measure discrepancy between resulting clusterings
    - Choose *k* to minimize discrepancy

Choosing the number of clusters *k*

- **Elbow criterion**
  - Marginal gain in objective may decrease at true / natural value of *k*
  - Not always unambiguously defined



Simulated data, 4 true clusters
(Courtesy: Dave Blei)



Human tumor microarray data
(Courtesy: Rob Tibshirani)

Choosing the number of clusters *k*

- **Gap statistic** (Tibshirani, Walther, & Hastie, 2001 – optional reading)
  - Let O_k be the objective value of *k*-means run on $\{x_1, \ldots, x_n\}$
  - Let U_k be the objective value of *k*-means run on n points sampled randomly from the smallest box containing $\{x_1, \ldots, x_n\}$
    - Serves as a single cluster null distribution
  - Roughly, choose *k* to maximize $\mathbf{Gap}(k) = \mathbf{E}[\log(U_k)] - \log(O_k)$
    - More precisely, form Monte Carlo estimate of Gap and choose smallest *k* such that

      $\text{Gap}_{est}(k) \geq \text{Gap}_{est}(k+1)$ - estimate of standard deviation of $\log(U_k)$

Gap statistic: simulated data

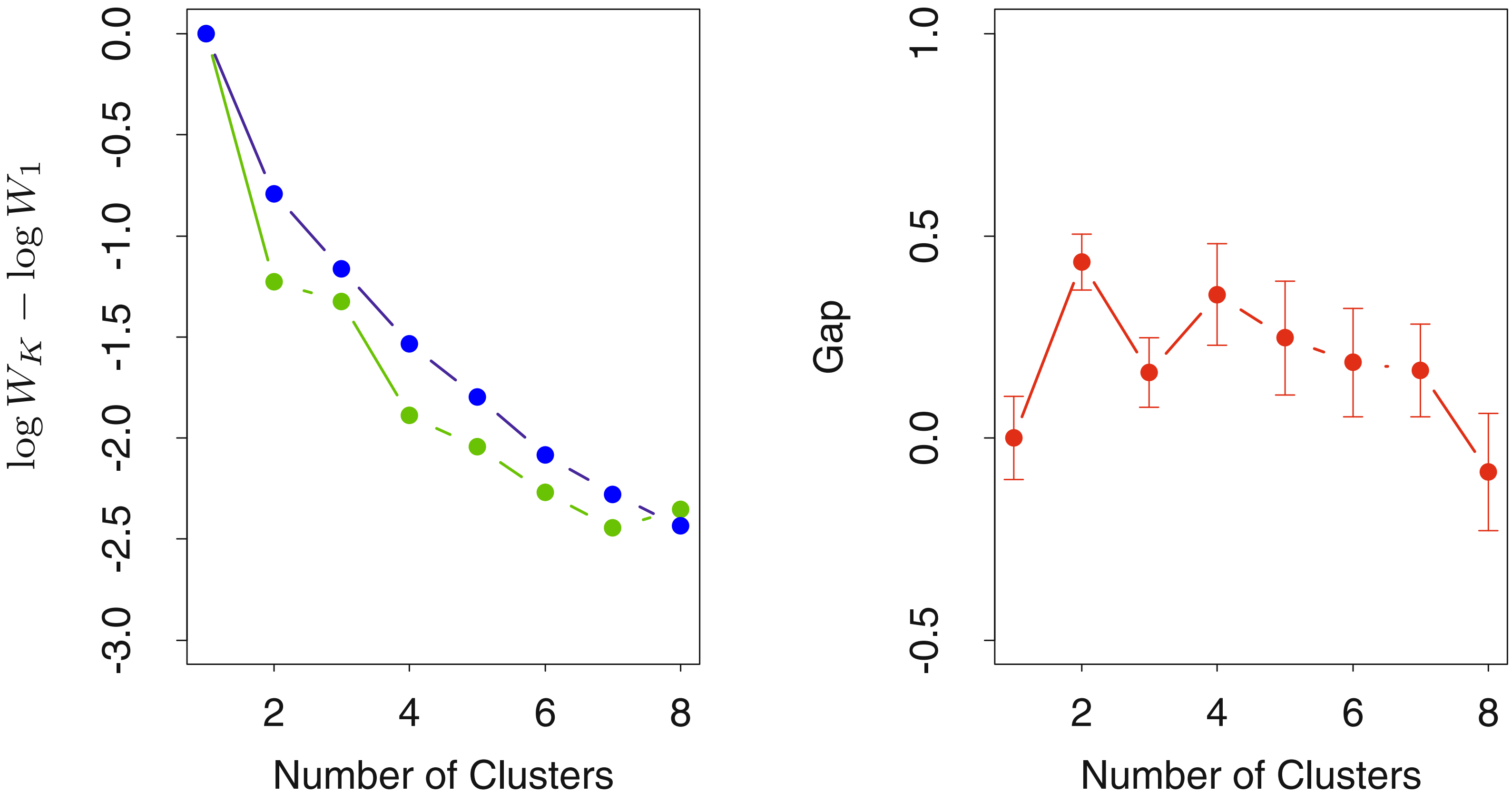


FIGURE 14.11. *(Left panel): observed (green) and expected (blue) values of* $\log W_K$ *for the simulated data of Figure 14.4. Both curves have been translated to equal zero at one cluster. (Right panel): Gap curve, equal to the difference between the observed and expected values of* $\log W_K$. *The Gap estimate* K^* *is the smallest* K *producing a gap within one standard deviation of the gap at* $K + 1$; *here* $K^* = 2$.

Comparing estimates of *k*

(Tibshirani, Walther, Hastie 2001)

Method	Estimate of number of clusters $\hat{k}$ 1	2	3	4	5	6	7	8	9	10
	Null model in 2 dimensions									
CH	0*	0	0	10	0	0	3	5	17	15
KL	0*	0	1	5	12	5	13	5	9	0
Hartigan	0*	0	0	0	0	0	0	0	2	48
Silhouette	0*	18	22	10	0	0	0	0	0	0
Gap	42*	7	0	1	0	0	0	0	0	0
Gap/pc	44*	6	0	0	0	0	0	0	0	0
	Null model in 10D									
CH	0*	50	0	0	0	0	0	0	0	0
KL	0*	29	5	3	3	2	2	0	0	0
Hartigan	0*	0	1	20	21	6	0	0	0	0
Silhouette	0*	49	1	0	0	0	0	0	0	0
Gap/unif	49*	1	0	0	0	0	0	0	0	0
Gap/pc	50*	0	0	0	0	0	0	0	0	0
	Three clusters									
CH	0	0	50*	0	0	0	0	0	0	0
KL	0	0	39*	0	5	1	1	2	0	0
Hartigan	0	0	1*	8	19	13	3	3	2	1
Silhouette	0	0	50*	0	0	0	0	0	0	0
Gap/unif	1	0	49*	0	0	0	0	0	0	0
Gap/pc	2	0	48*	0	0	0	0	0	0	0

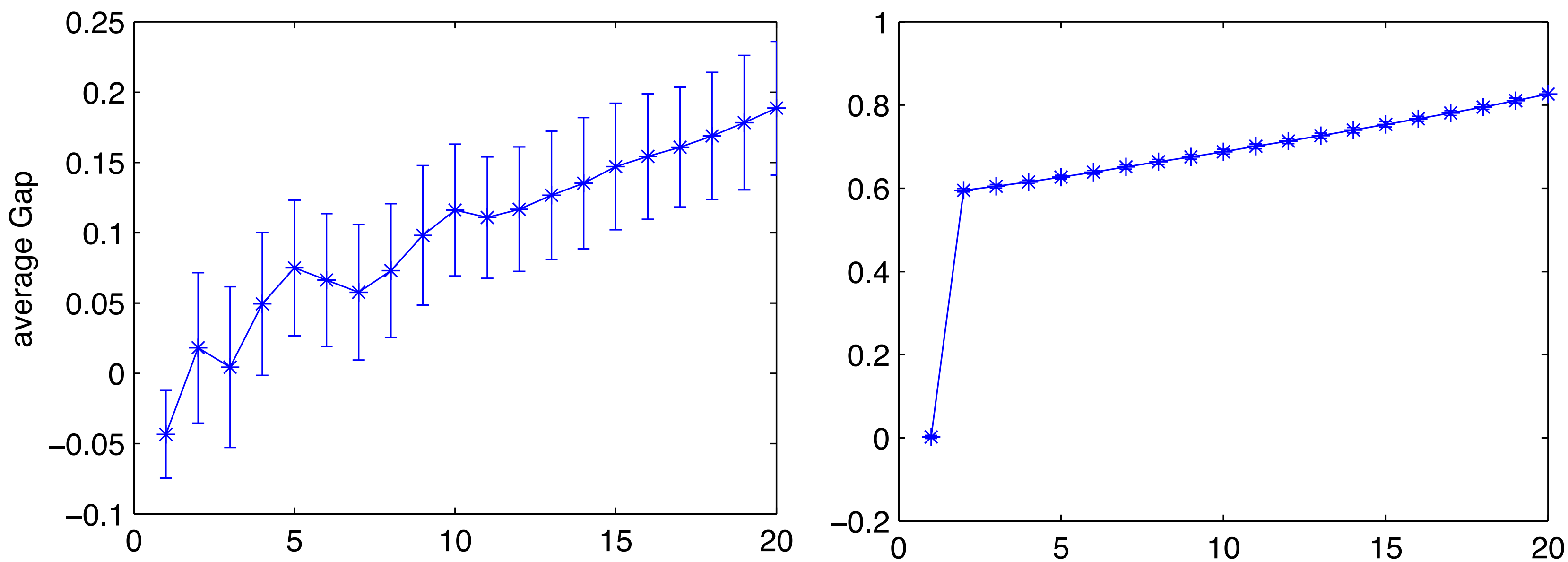
Comparing estimates of *k*

(Tibshirani, Walther, Hastie 2001)

Method	Estimate of number of clusters $\hat{k}$: 1	2	3	4	5	6	7	8	9	10
	Random 4 clusters in 3D									
CH	0	0	0	42*	8	0	0	0	0	0
KL	0	0	0	35*	5	3	3	3	0	0
Hartigan	0	1	7	3*	9	12	8	2	3	5
Silhouette	0	20	15	15*	0	0	0	0	0	0
Gap/unif	0	1	2	47*	0	0	0	0	0	0
Gap/pc	2	2	4	42*	0	0	0	0	0	0
	Random 4 clusters in 10D									
CH	0	1	4	44*	1	0	0	0	0	0
KL	0	0	0	45*	3	1	1	0	0	0
Hartigan	0	0	2	48*	0	0	0	0	0	0
Silhouette	0	13	20	16*	5	0	0	0	0	0
Gap/unif	0	0	0	50*	1	0	0	0	0	0
Gap/pc	0	0	4	46*	0	0	0	0	0	0
	Two elongated clusters									
CH	0	0*	0	0	0	0	0	7	16	27
KL	0	50*	0	0	0	0	0	0	0	0
Hartigan	0	0*	0	1	0	2	1	5	6	35
Gap/unif	0	0*	17	16	2	14	1	0	0	0
Gap/pc	0	50*	0	0	0	0	0	0	0	0

Choosing the number of clusters *k*

- **Gap statistic**
  - Performs similarly to other leading methods when $k > 1$
  - Pro: Can detect $k = 1$ (many other methods can't)
  - Con: Performs poorly in high dimensions



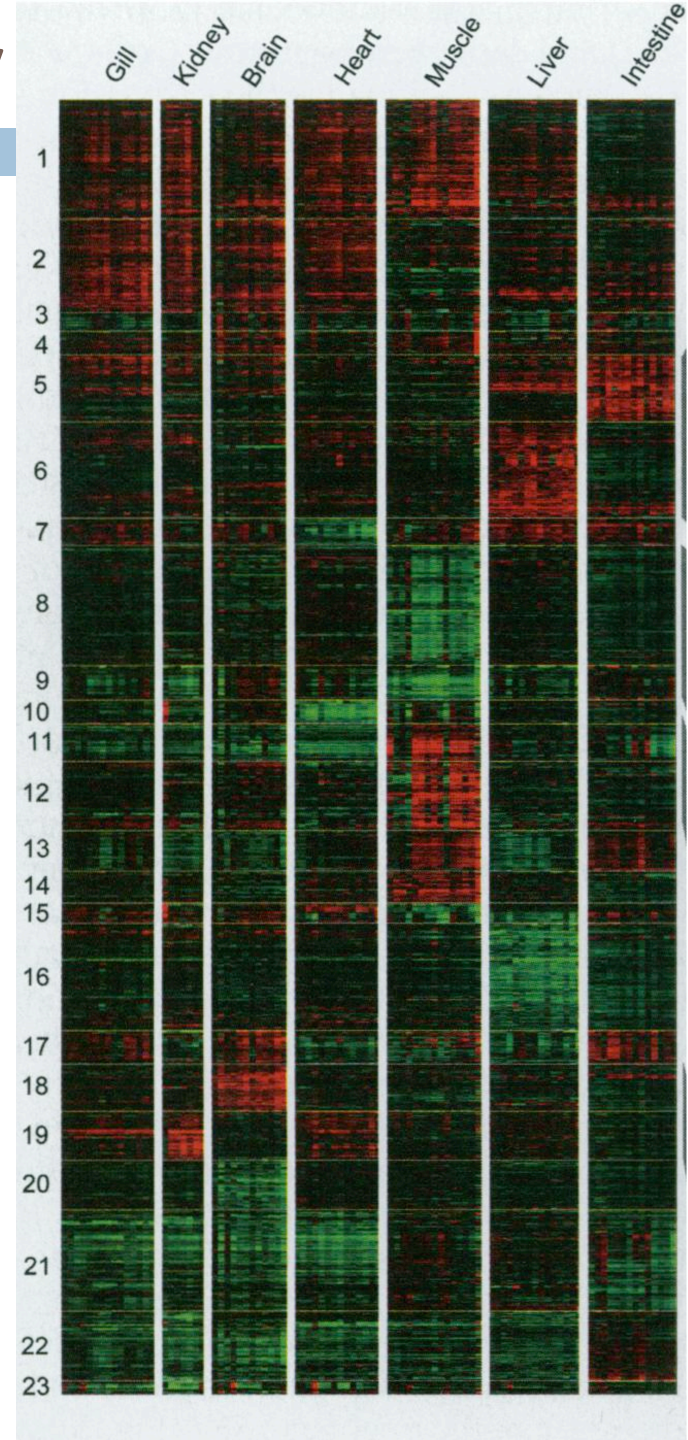
Simulations, true $k = 2$: $p = 2$ (Left), $p = 100$ (right)

(Mohajer et al., 2011: A comparison of Gap statistic definitions with and without logarithm function)

k-means in the wild: Biology

Coping with cold: An integrative, multitissue analysis of the transciptome of a poikilothermic vertebrate (Gracey et al., 2004)

- Carp exposed to increasing levels of cold
- Genes (rows) clustered using 23-means according to cold response across different tissues
  - No explanation for *k* = 23 given
- Eventually interpreted functional significance of each cluster



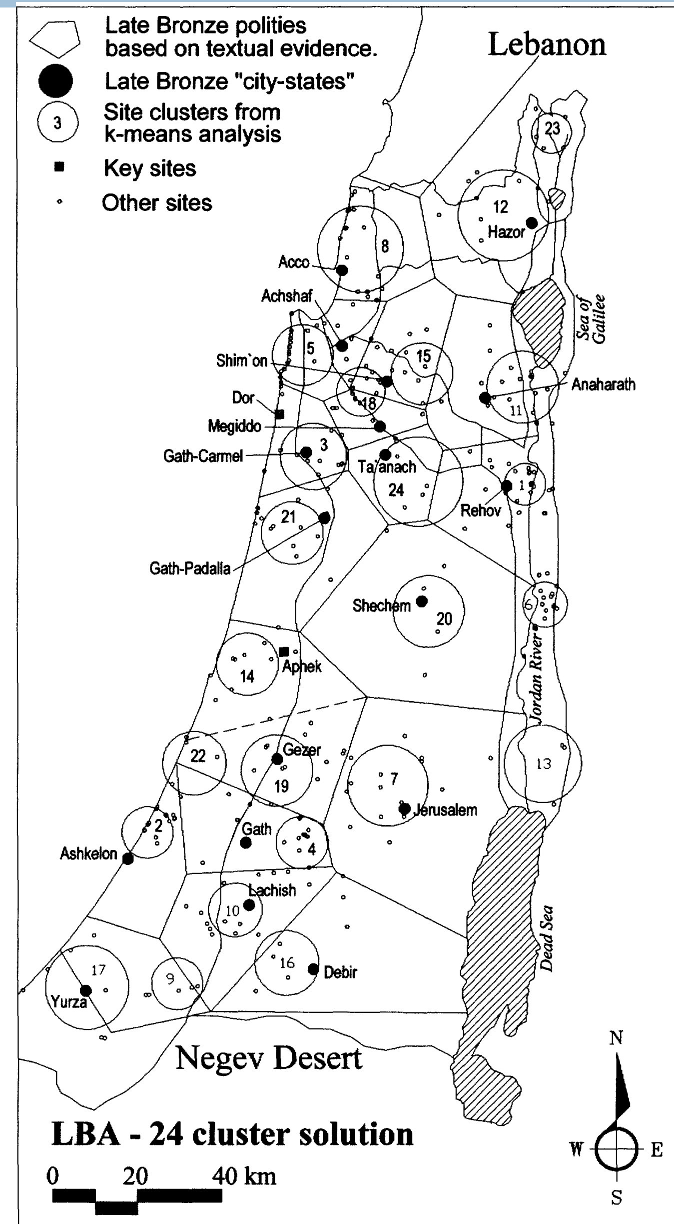
Credit: Dave Blei

k-means in the wild: Archaeology

Credit: Dave Blei

Spatial and Statistical Inference of Late Bronze Age Polities in the Southern Levant (Savage and Falconer, 2003)

- Cluster archaeological site locations in Israel with *k*-means
- *k* chosen by comparing to a null distribution based on randomly sampled points
- "Infer a political landscape that corresponds well with many aspects of historical reconstruction and propose new ideas on the configuration and structure of Late Bronze Age [1500-1200 BC] polities"



k-means in the wild: Education

Credit: Dave Blei

Teachers as Sources of Middle School Students' Motivational Identity: Variable-Centered and Person-Centered Analytic Approaches (Murdock and Miller, 2003)

- Clustered 206 eighth-grade students by survey data describing parent academic support, peer academic support, and teacher caring levels
- No clusters centers had above average support for one category and below average support for another; suggests that support classes do not compensate for one another?
- k = 5 chosen based on parsimony, heterogeneity, convergence issues, and inspection

k-means in the wild: Education

Credit: Dave Blei

TABLE 3. Five-Cluster Solution: *Z* scores on Each Clustering Variable

	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
Teacher caring	−.5	−.5 to .5	−.5 to .5	−.5	1.0
Peers' academic support	1.0	−.5	1.0	−.5	−.5 to .5
Parents' academic support	.5	−1.0	−.5 to .5	−.5 to .5	1.0

TABLE 4. Means and Standard Deviations for Each Cluster on Grade 8 Motivational Variables

	Academic Self-Efficacy		Intrinsic Valuing of Education		Teacher-Rated Effort	
Cluster	*M*	SD	*M*	SD	*M*	SD
1. All positive	3.59	.48[a]	2.99	.55[a]	3.74	.26[a]
2. Peer negative, parents very negative	2.44	.66[b]	2.16	.51[b]	3.05	.61[b]
3. Peer positive	3.01	.73[c]	2.43	.66[b]	3.26	.66[b]
4. Negative teacher and peer	2.47	.63[b]	2.24	.51[b]	3.17	.59[b]
5. Positive teacher and parents	3.19	.65[c]	2.89	.62[a]	3.54	.47[a]

k-means: Practical considerations, Part II

- Hard assignments to clusters not stable under small perturbations of data
  - **Mixture modeling** (next time) employs soft assignments
- Gives equal weight to each coordinate and cluster
  - Mixture modeling can relax both assumptions
- Clusters change arbitrarily for different K
  - **Hierarchical clustering** (later) yields nested clusterings
- Works poorly on non-convex clusters
  - **Spectral clustering** (later) well-suited to non-convex clusters

Summary

- Unsupervised learning:
  - **Goal:** Discover hidden structure in data without prior labels or observations of that structure
  - Challenging but necessary
  - Various practical benefits
- Clustering
  - **Goal:** Segment datapoints into similar groups
  - Many applications, many approaches
- *k*-means
  - Simple, popular, canonical approach to clustering
  - Great diversity of applications, including vector quantization
  - Various drawbacks and opportunities for improvement
    - Objective, solution optimality, choice of k, running time
  - Various generalizations, including *k*-medoids

Credits

- Parts of this material were adapted from slides by Dave Blei, Sriram Sankararaman, and Robert Tibshirani