

Functional genomics + Data mining

BCH394P/364C Systems Biology / Bioinformatics
Edward Marcotte, Univ of Texas at Austin

Functional genomics

= field that attempts to use the vast data produced by genomic projects (e.g. genome sequencing projects) to describe gene (and protein) functions and interactions.

Focuses on dynamic aspects, e.g. transcription, translation, and protein–protein interactions, as opposed to static aspects of the genome such as DNA sequence or structures.

Adapted from Wikipedia

Functional genomics + Data mining

= field that attempts to computationally discover patterns in large data sets

Adapted from Wikipedia

Functional genomics + Data mining



www.sparkpeople.com

Adapted from Wikipedia

**We're going to first learn
about clustering algorithms
& classifiers**

**We're going to first learn
about clustering algorithms
& classifiers**

Clustering = task of grouping a set of objects in such a way that objects in the same group (a **cluster**) are more similar (in some sense) to each other than to those in other groups (clusters).

Adapted from Wikipedia

We're going to first learn about clustering algorithms & classifiers

Classification = task of categorizing a new observation, on the basis of a training set of data with observations (or instances) whose categories are known

Adapted from Wikipedia

Let's motivate this with an important
historical example:

Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling

Ash A. Alizadeh^{1,2}, Michael B. Eisen^{2,3,4}, R. Eric Davis⁵, Chi Ma⁵, Izidore S. Lossos⁶, Andreas Rosenwald⁵, Jennifer C. Boldrick¹, Hajeer Sabet⁵, Truc Tran⁵, Xin Yu⁵, John I. Powell⁷, Liming Yang⁷, Gerald E. Marti⁸, Troy Moore⁹, James Hudson Jr³, Lisheng Lu¹⁰, David B. Lewis¹⁰, Robert Tibshirani¹¹, Gavin Sherlock⁴, Wing C. Chan¹², Timothy C. Greiner¹², Dennis D. Weisenburger¹², James O. Armitage¹³, Roger Warnke¹⁴, Ronald Levy⁶, Wyndham Wilson¹⁵, Michael R. Grever¹⁶, John C. Byrd¹⁷, David Botstein⁴, Patrick O. Brown^{1,18} & Louis M. Staudt⁵

Nature 2000

“Diffuse large B-cell lymphoma (DLBCL), the most common subtype of non-Hodgkin's lymphoma ... is one disease in which attempts to define subgroups on the basis of morphology have largely failed...”

“DLBCL ... is clinically heterogeneous: 40% of patients respond well to current therapy and have prolonged survival, whereas the remainder succumb to the disease.

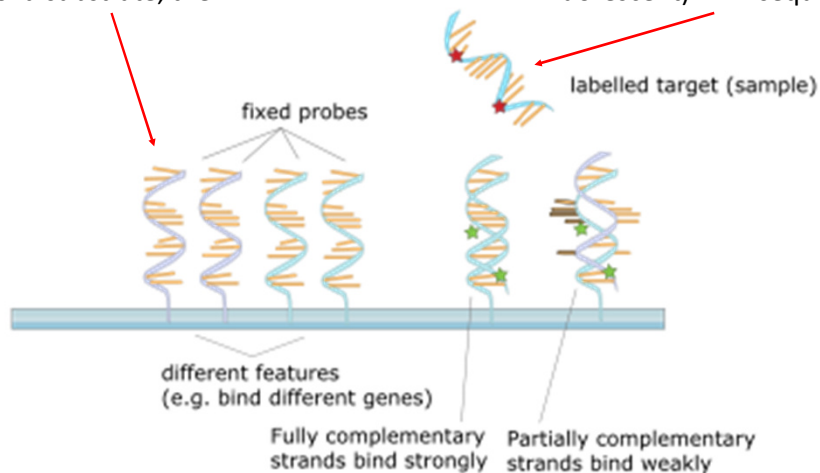
We proposed that this variability in natural history reflects unrecognized molecular heterogeneity in the tumours.”

Nature 2000

Blast from the past: Profiling mRNA expression with DNA microarrays

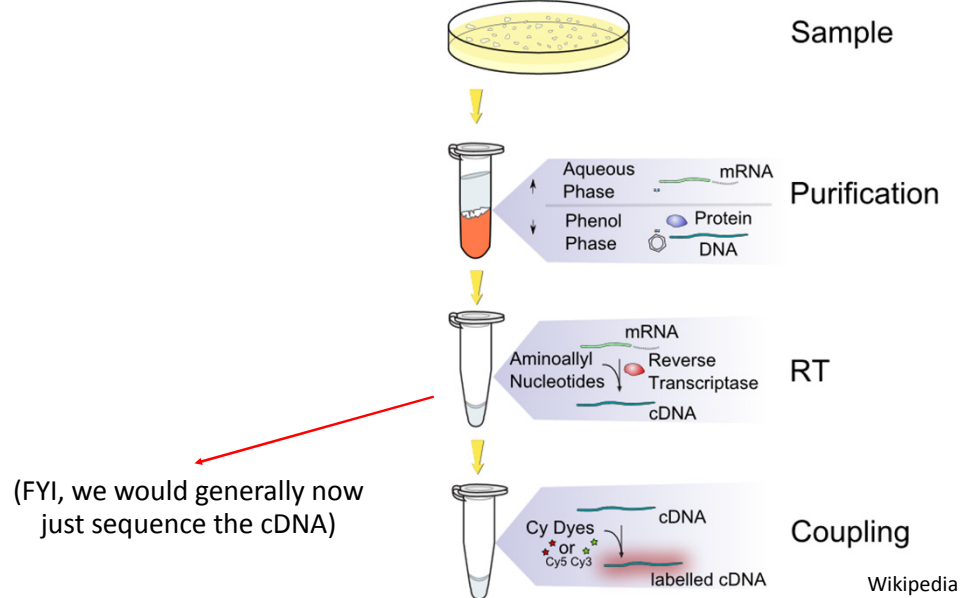
DNA molecules are attached to a solid substrate, then...

...probed with a labeled (usually fluorescent) DNA sequence

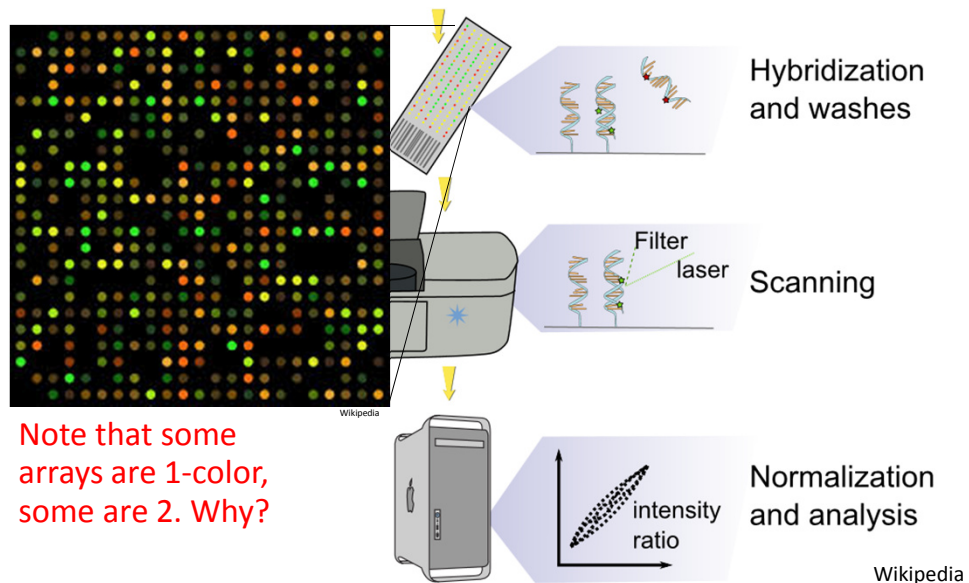


Wikipedia

Blast from the past: Profiling mRNA expression with DNA microarrays

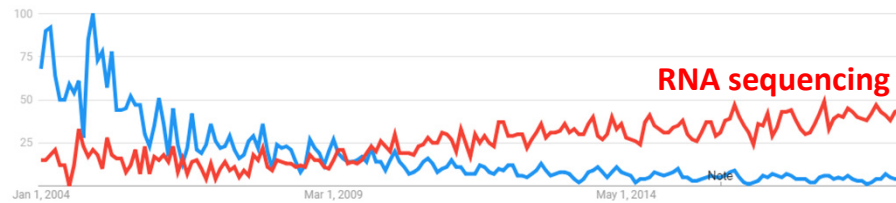


Blast from the past: Profiling mRNA expression with DNA microarrays



DNA microarrays are a great example of the “arc” of a technology over time

DNA microarrays



Back to diffuse large B-cell lymphoma...

**96 patient biopsies
(normal and malignant lymphocyte samples)**



Extract mRNA from each sample

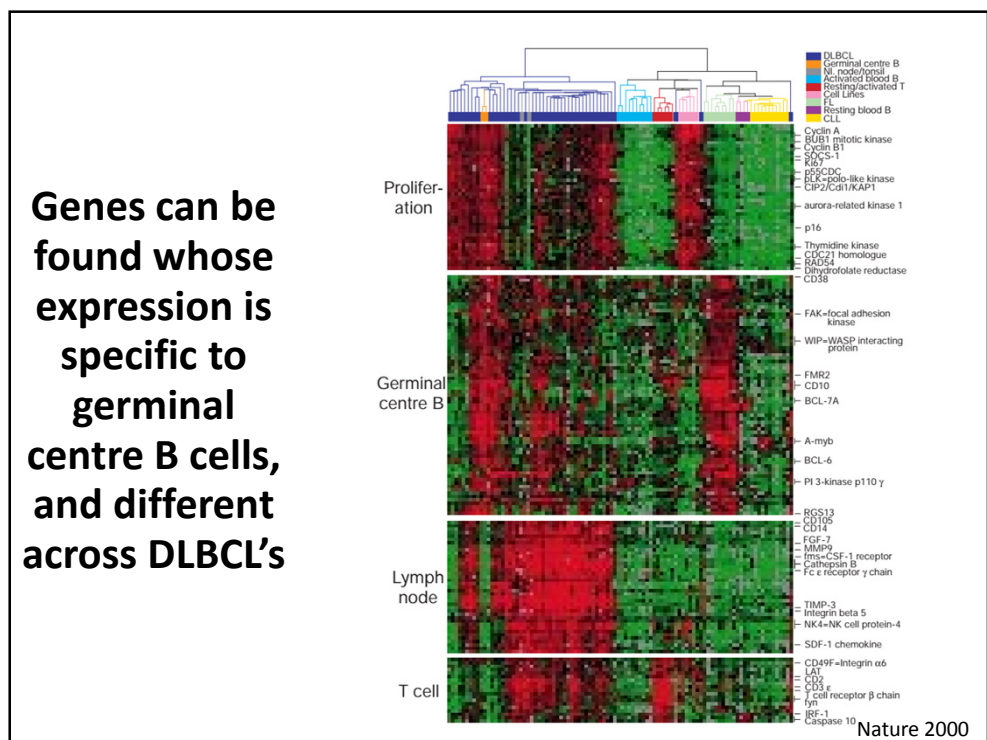
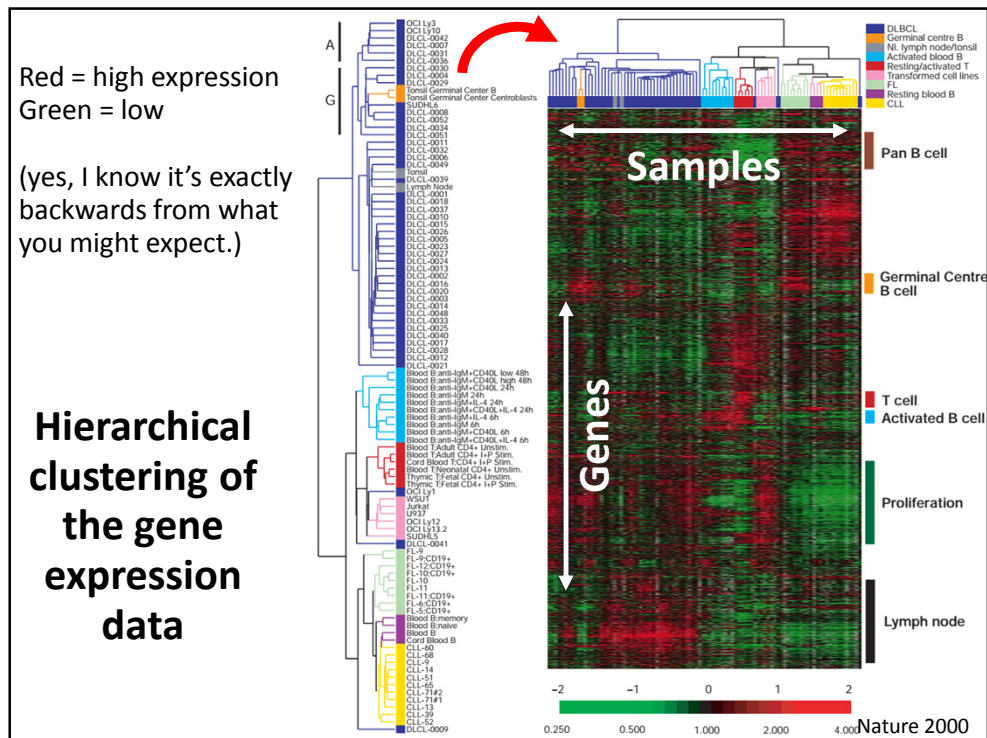


**Perform DNA microarray experiment on each to
measure mRNA abundances (~1.8 million total gene
expression measurements)**

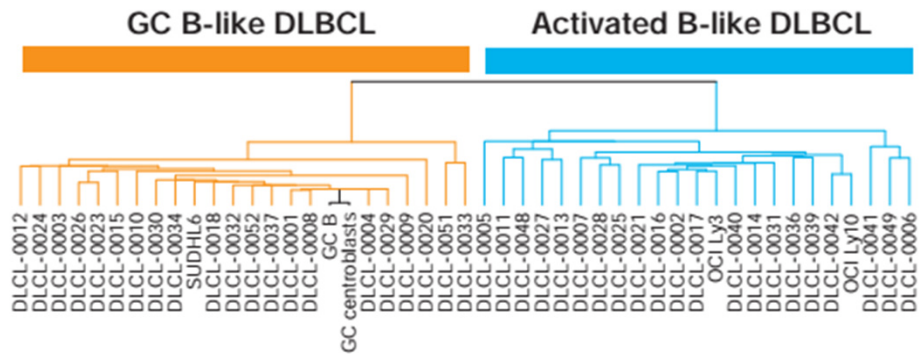


Cluster samples by their expression patterns

Nature 2000



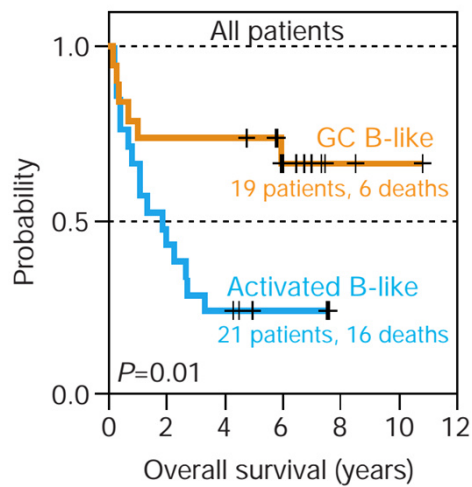
We can break up the DLBCL's according the germinal B-cell specific gene expression:



Nature 2000

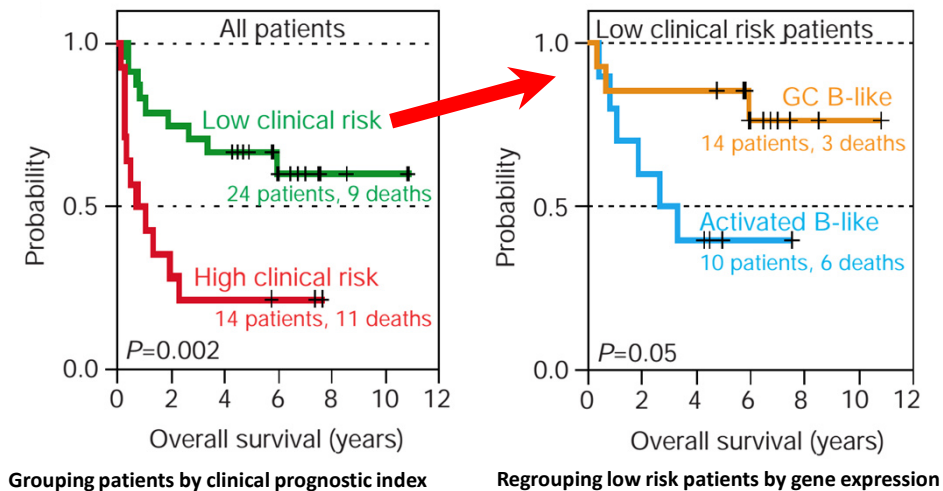
What good is this? These molecular phenotypes predict clinical survival.

Kaplan-Meier plot
of patient survival



Nature 2000

What good is this? These molecular phenotypes predict clinical survival.



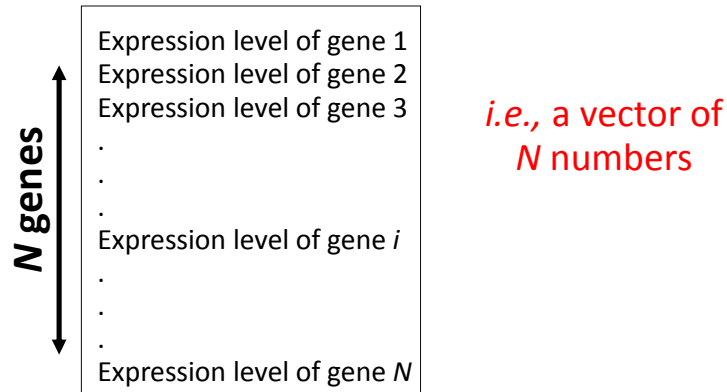
Nature 2000

Gene expression, and other molecular measurements, provide far deeper phenotypes for cells, tissues, and organisms than traditional measurements

These sorts of observations have now motivated tons of work using these approaches to diagnose specific forms of disease, as well as to discover functions of genes and many other applications

So, how does clustering work?

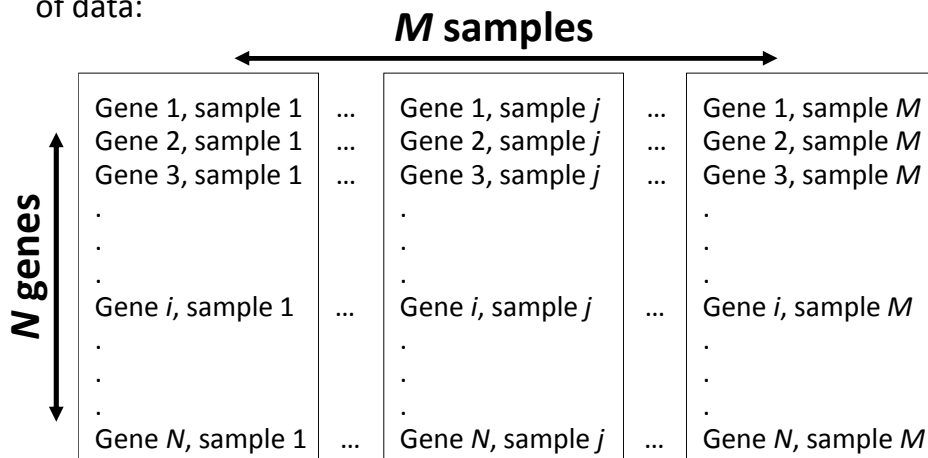
First, let's think about the data, e.g. as for gene expression.
From one sample, using DNA microarrays or RNA-seq, we get:



For yeast, $N \sim 6,000$
For human, $N \sim 22,000$

So, how does clustering work?

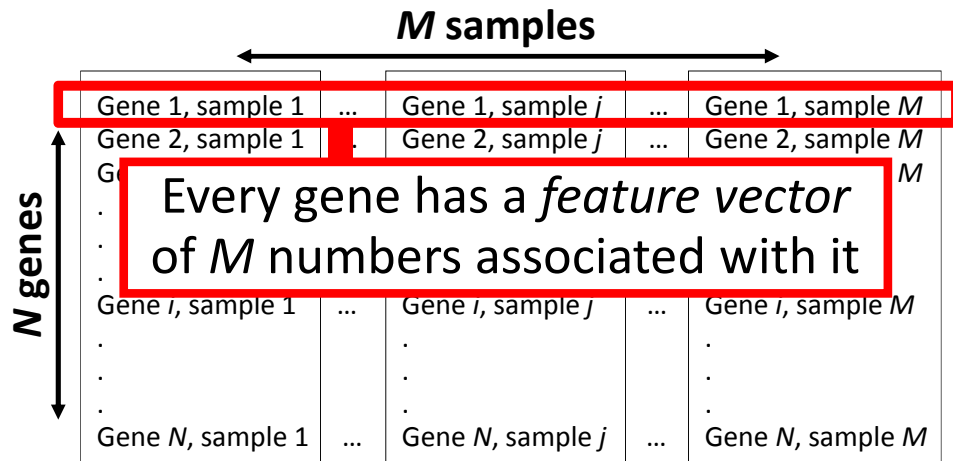
Every additional sample adds another column, giving us a matrix of data:



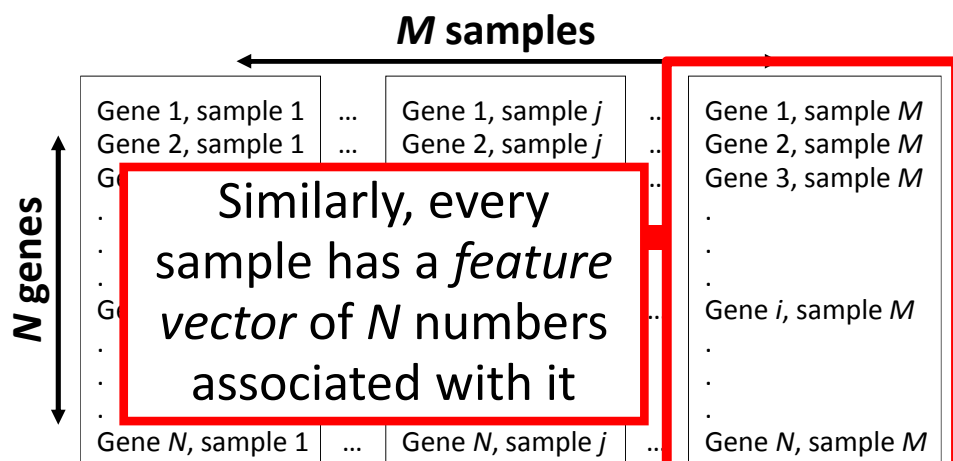
For yeast, $N \sim 6,000$
For human, $N \sim 22,000$

*i.e., a matrix of N
 $\times M$ numbers*

So, how does clustering work?



So, how does clustering work?



So, how does clustering work?

M samples

N genes

The first clustering method we'll learn about simply groups the objects (samples or genes) in a hierarchy by the similarity of their feature vectors.

Gene N , sample 1	...	Gene N , sample j	...	Gene N , sample M
---------------------	-----	-----------------------	-----	-----------------------

A hierarchical clustering algorithm

Start with each object in its own cluster

Until there is only one cluster left, repeat:

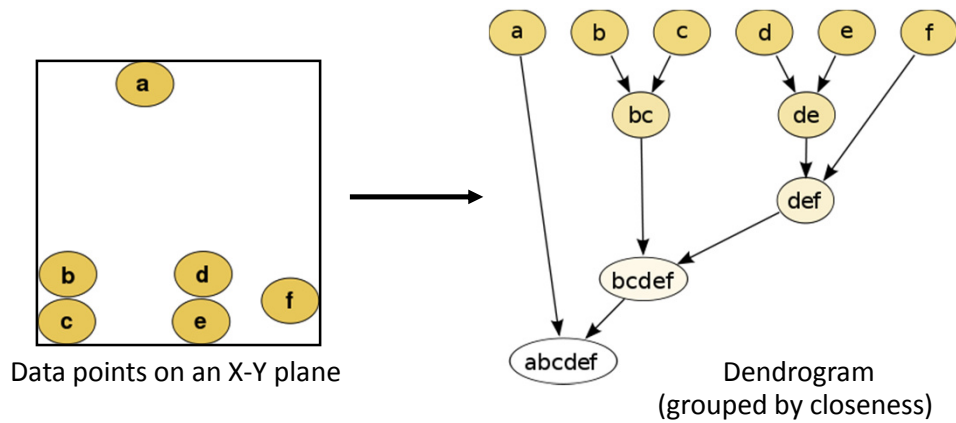
Among the current clusters, find the two most similar clusters

Merge those two clusters into one

We can choose our measure of similarity and how we merge the clusters

Hierarchical clustering

Conceptually



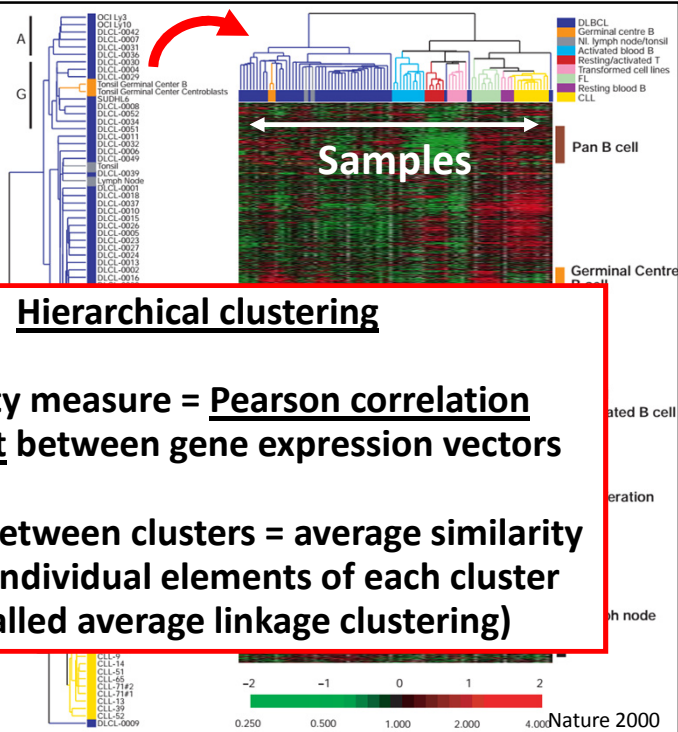
Wikipedia

We'll need to measure the similarity between feature vectors. Here are a few (of many) common distance measures used in clustering.

Names	Formula
Euclidean distance	$\ a - b\ _2 = \sqrt{\sum_i (a_i - b_i)^2}$
Manhattan distance	$\ a - b\ _1 = \sum_i a_i - b_i $
cosine similarity	$\frac{a \cdot b}{\ a\ \ b\ }$

Wikipedia

Back to the B cell lymphoma example



K-means clustering is a common alternative clustering approach

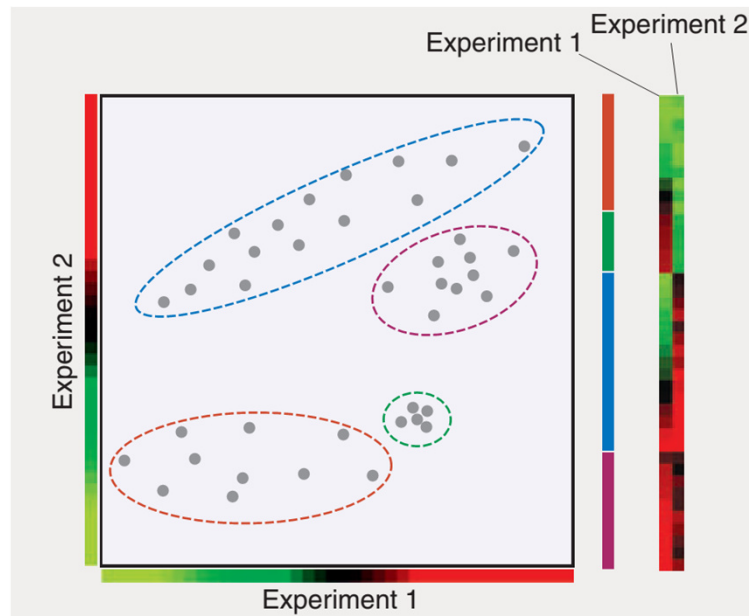
mainly because it's easy and can be quite fast!

The basic algorithm:

1. Pick a number (k) of cluster centers
2. Assign each gene to its nearest cluster center
3. Move each cluster center to the mean of its assigned genes
4. Repeat steps 2 & 3 until convergence

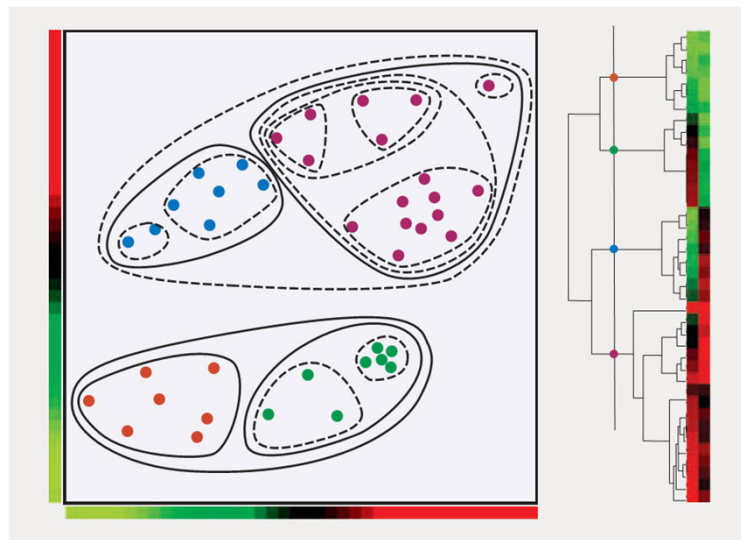
See the K-means example posted on the web site

A 2-dimensional example



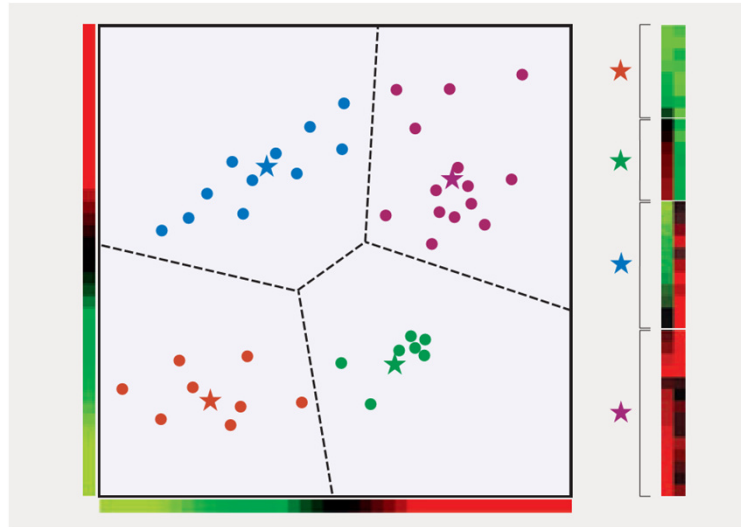
Nature Biotech 23(12):1499-1501 (2005)

A 2-dimensional example: hierarchical



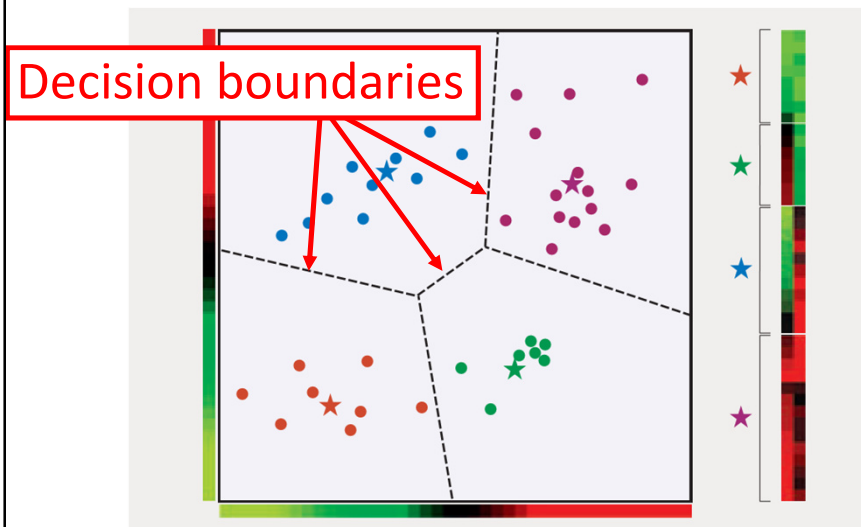
Nature Biotech 23(12):1499-1501 (2005)

A 2-dimensional example: k -means



Nature Biotech 23(12):1499-1501 (2005)

A 2-dimensional example: k -means



Nature Biotech 23(12):1499-1501 (2005)

Some features of K-means clustering

- Depending on how you seed the clusters, it may be stochastic. You may not get the same answer every time you run it.
- Every data point ends up in exactly 1 cluster (so-called *hard* clustering)
- Not necessarily obvious how to choose k
- Great example of something we've seen already: Expectation-Maximization (E-M) algorithms

EM algorithms alternate between assigning data to models (here, assigning points to clusters) and updating the models (calculating new centroids)

Some features of K-means clustering

- Depending on how you seed the clusters, it may be stochastic. You may not get the same answer every time you run it.
- Every data point ends up in exactly 1 cluster (so-called *hard* clustering)
- Not necessarily obvious how to choose k
-

Let's think about this aspect for a minute.
Why is this good or bad?
How could we change it?

EM algorithms alternate between assigning data to models (here, assigning points to clusters) and updating the models (calculating new centroids)

***k*-means**

The basic algorithm:

1. Pick a number (*k*) of cluster centers
2. Assign each gene to its nearest cluster center
3. Move each cluster center to the mean of its assigned genes
4. Repeat steps 2 & 3 until convergence

Fuzzy *k*-means

The basic algorithm:

1. Choose *k*. Randomly assign cluster centers.
2. Fractionally assign each gene to each cluster:

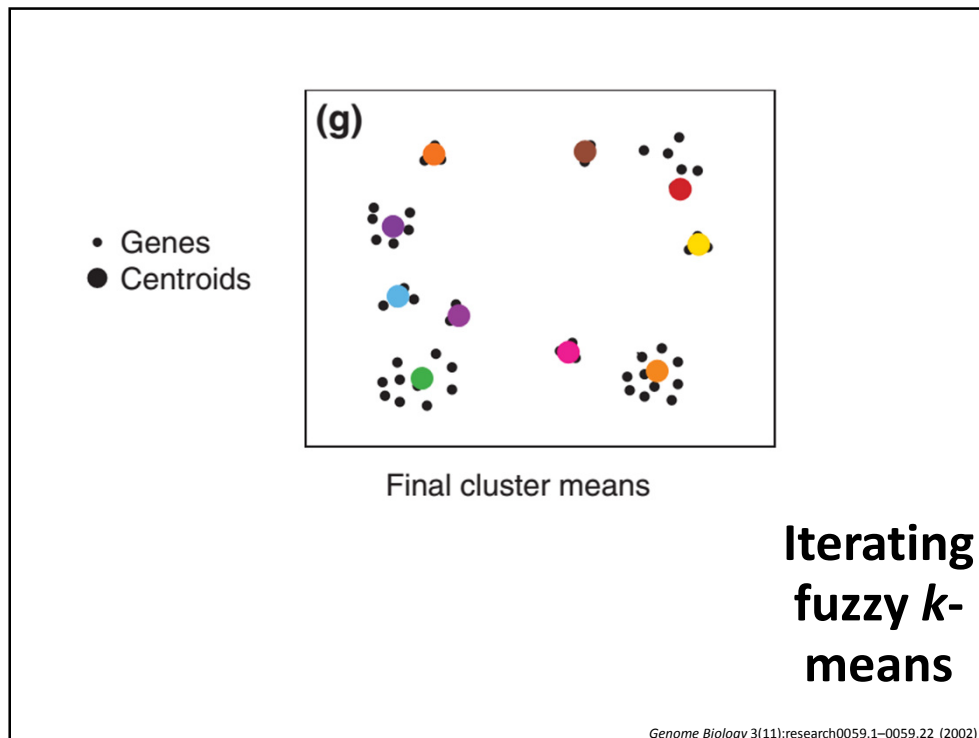
e.g. occupancy $(g_i, m_j) = \frac{e^{-\|g_i - m_j\|^2}}{\sum_j e^{-\|g_i - m_j\|^2}}$

Note: $\|x\|$ is just shorthand for the length of the vector *x*.

g_i = gene *i*

m_j = centroid of cluster *j*

3. For each cluster, calculate weighted mean of genes to update cluster centroid
4. Repeat steps 2 & 3 until convergence



A fun clustering strategy that builds on these ideas: Self-organizing maps (SOMs)

- Combination of clustering & visualization
- Invented by Teuvo Kohonen, also called Kohonen maps



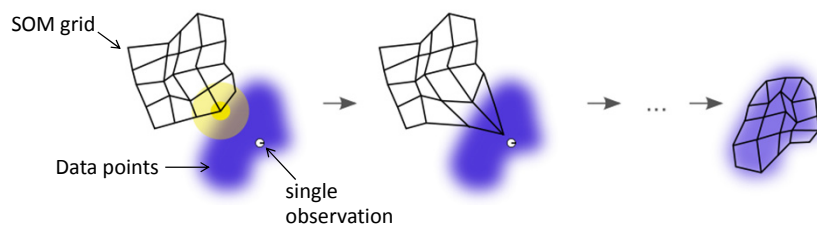
*Dr. Eng., Emeritus
Professor of the
Academy of Finland;
Academician*

A fun clustering strategy that builds on these ideas: Self-organizing maps (SOMs)

SOMs have:

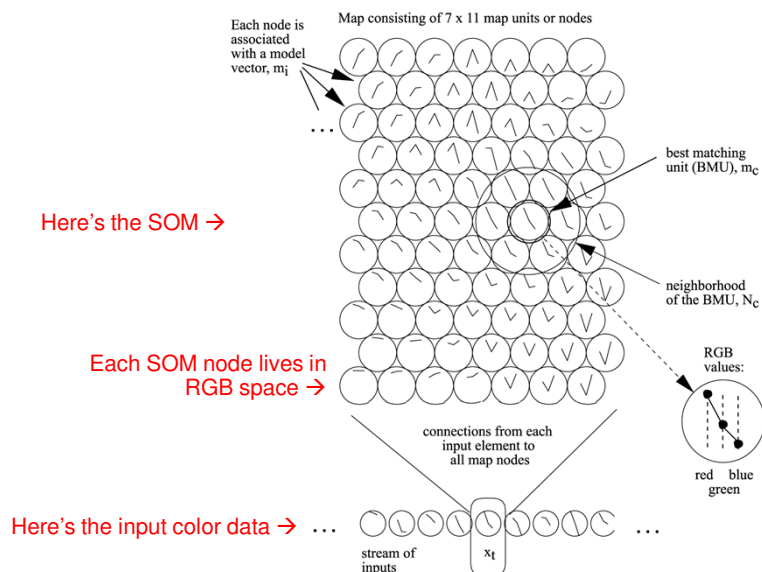
- your data (points in some high-dimensional space)
- a grid of nodes, each node also linked to a point someplace in data space

1. First, SOM nodes are arbitrarily positioned in data space. Then:
 2. Choose a training data point. Find the node closest to that point.
 3. Move its position closer to the training data point.
 4. Move its grid neighbors closer too, to a lesser extent.
- Repeat 2-4. After many iterations, the grid approximates the data distribution.

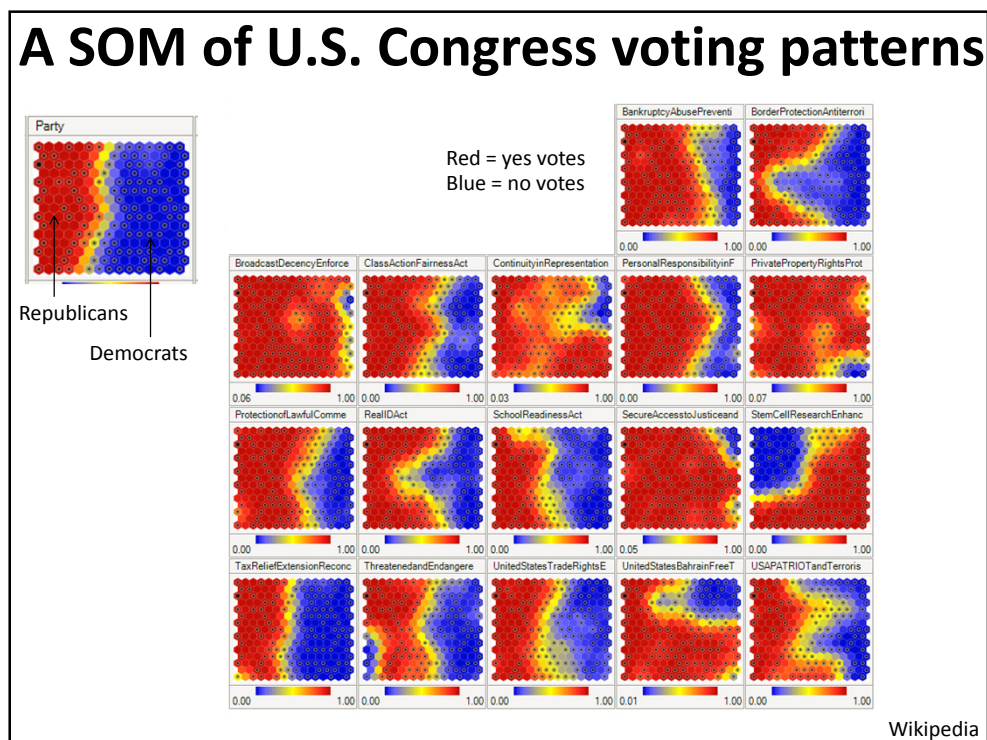
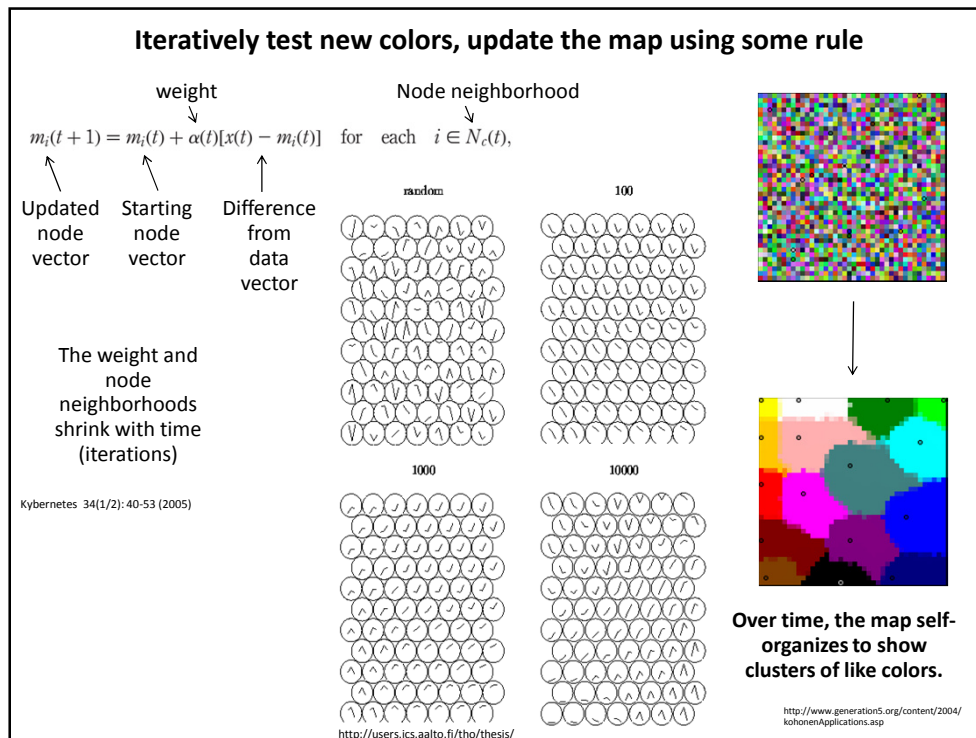


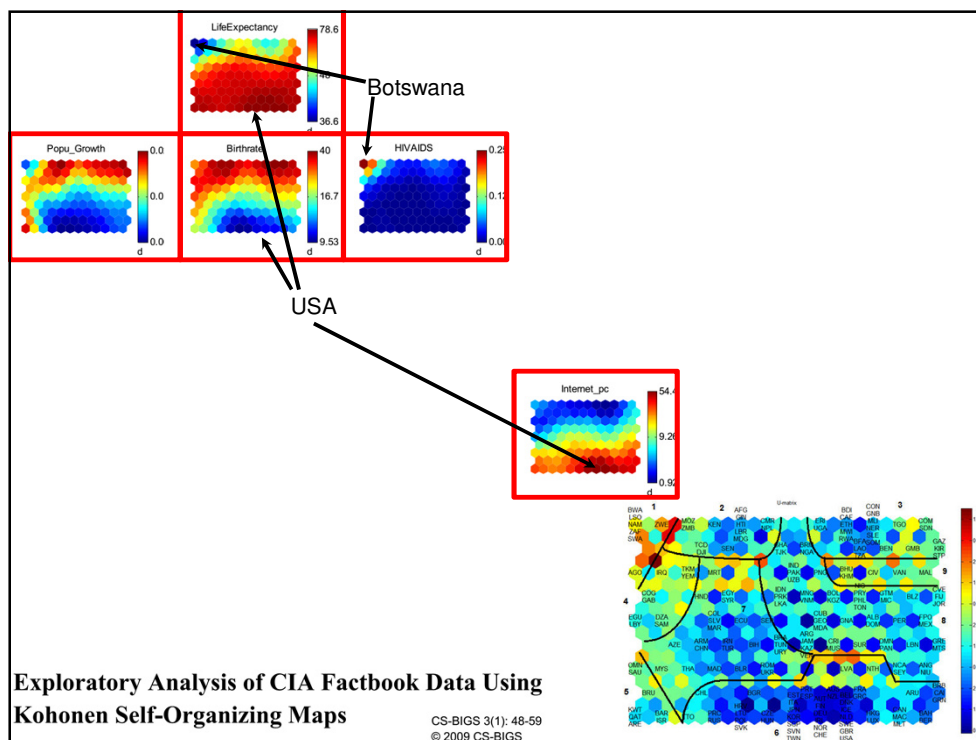
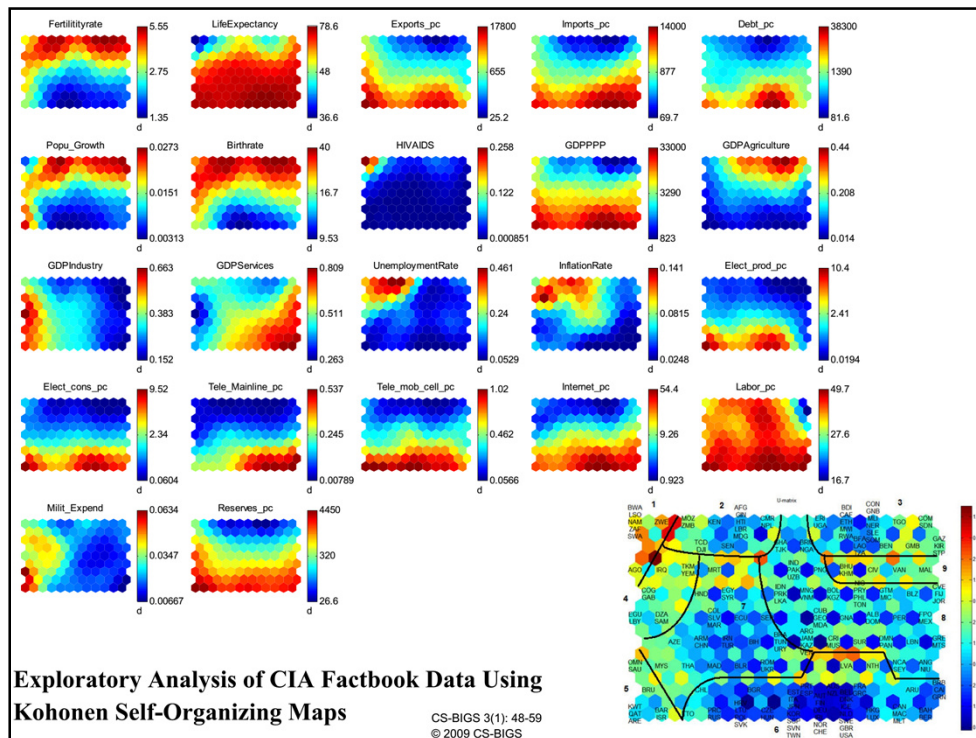
Wikipedia

Here's an example using colors. Each color has an RGB vector. Take a bunch of random colors and organize them into a map of similar colors:



Kybernetes 34(1/2): 40-53 (2005)





(data = wiki article word frequency vectors)



SOMs can accommodate unusual data distributions

One-dimensional SOM



Finally, **t-SNE** is a nice way to visualize data in 2 or 3D
 = *t-distributed stochastic neighbor embedding*

t-SNE tries to reproduce high-D *data neighborhoods* in a 2D or 3D picture by:

1. Defining a probability distribution over pairs of high-D objects such that “similar” objects have a high probability of being picked, whilst “dissimilar” objects have an extremely small probability of being picked
2. Defining a similar probability distribution over the points in the low- D map
3. Minimizing the Kullback–Leibler divergence between the two distributions by varying the locations of the points in the low-D map, i.e.

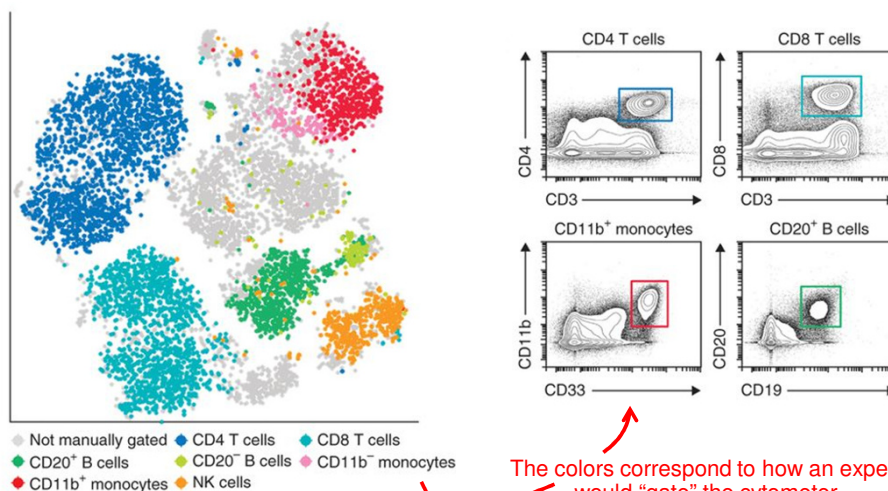
minimize this:
$$\sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

p_{ij} ← probability *i* and *j* are close in high-D space
 q_{ij} ← probability *i* and *j* are close in low-D space
 Sum over all pairs of points

van der Maaten & Hinton, Visualizing High-Dimensional Data Using t-SNE.
Journal of Machine Learning Research 9: 2579–2605 (Nov 2008)

Separating cells into cell types by t-SNE

- healthy human bone marrow, stained with 13 markers and measured by mass cytometry, visualized with viSNE



Amir et al., *Nature Biotechnology* 31:545–552 (2013)

**You can compute your own t-SNE embeddings
using the online tools at:**

<http://projector.tensorflow.org/>

There are also some great examples at:

<http://distill.pub/2016/misread-tsne/>

There are only a couple of parameters you can tweak, mainly perplexity,
which effectively captures the number of neighbors (often 5 to 50)