Graphs and Networks Lecture 19

Graph Clustering : Spectral Methods and Normalized Cuts

Daniel A. Spielman

November 9, 2010

19.1 Overview

In this and the next lecture, we are going to consider approaches to clustering the vertices of a graph. I think that we understand reasonably well how to partition the vertices of a graph into two sets. However, in clustering, we want to divide the vertices of a graph into many sets. This problem is not nearly as well understood.

There is quite a bit of disagreement over what one should be optimizing. Even once one has a measure of the quality of a clustering, it is usually computationally difficult to find a clustering that optimizes this measure. So, one typically uses a heuristic. The best heuristics typically combine two operations: a global optimization followed by local improvements. This lecture will probably just focus on the global optimizations, unless I have time time to implement some local improvement algorithms.

The algorithms that work best depend quite a bit on the area of application. The Scientific Computing community has developed a number of algorithms for partitioning well-shaped meshes (Chaco, Metis and Scotch). Different, but related, algorithms have proved popular in Image Segmentation (Shi and Malik, Yu and Shi). A very different type of algorithm is popular with Phyisicists who now study social networks. We will see this type of algorithm next lecture.

For now, let me recommend the survey of von Luxburg [Lux07].

19.2 K-Means

Before we get too into how one should cluster the vertices of a graph, lets take a moment to consider the seemingly easier problem of clustering vectors in $\mathbb{R}^d$. Lets call the vectors $x_1, \ldots, x_n$. One of the most popular measures of the quality of a partition of these vectors into clusters $C_1, \ldots, C_k$ is the k-means objective function. It is

$$\sum_{a=1}^{k} \frac{1}{|C_a|} \sum_{i,j \in C_a} \|x_i - x_j\|^2 . \tag{19.1}$$

This expression is simplified by setting μ_a to be the average of the points in cluster C_a:

$$\mu_a = \frac{1}{|C_a|} \sum_{i \in C_a} x_i .$$

We then have that (19.1) equals

$$\sum_{a=1}^{k} \sum_{i \in C_a} \|x_i - \mu_a\|^2 . \tag{19.2}$$

That is, we sum the square of the distance of each point to the center of its cluster.

While it is NP-hard to find the clusters that minimize this objective function (even for $k = 2$), there is a very popular heuristic called the k-means algorithm (introduced by Lloyd [Llo82]) for approximately minimizing the objective function. Before I tell you the algorithm, I'd like to complain that many people don't make the distincition between the objective function and the algorithm, which is just careless.

Lloyd's consists of alternating steps in which one computes the cluster-averages, $\mu_1, \ldots, \mu_k$, and then shifts each point to the cluster with the closest center. That is, we alternate the steps

1. For each $1 \leq a \leq k$, set $\mu_a = (1/|C_a|) \sum_{i \in C_a} x_i$.
2. For each $1 \leq i \leq n$, put i in the cluster a for which $\|\mu_a - x_i\|$ is lowest.

One can show that each of these steps will decrease the objective function. I didn't say how to start. Typically, one will choose k random data points and make them the cluster centers. A better initialization is given by choosing the k points with probability inversely proportional the the square of their distance from the previous points (k-means++ [AV07]).

One typically runs this algorithm until it stops making any changes. Then, one usually runs it again and again with different random starts. It is not very consistent. But, it is easy to implement, so people like to use it.

19.3 Clustering in Graphs

One could try to directly lift the k-means algorithm to a graph. If A is the adjacency matrix of the graph, we could take x_i to be the ith row of A. This sometimes works. But, there are graphs on which it does remarkably poorly. To get some idea as to why, consider two vertices that do not have any neighbors in common. The rows corresponding to these vertices will be orthogonal, and so their distance will be trivial. This problem can be particularly severe in a bipartite graph. OK, maybe its surprising that this ever works. The other problem is that the dimension of the space is very large (equal to the number of vertices), which makes the algorithm slow.

So, we would like to get some coordinates in a low-dimensional space for the vertices of the graph We know from Cheeger's inequality that the eigenvector of the second-largest eigenvalue of the walk matrix is good for partitioning into two parts. So, it seems natural to use a few more eigenvectors if we want to partition into more parts. This idea, but with the Laplacian matrix instead of the walk matrix, was proposed by Chan, Schlag and Zien [CSZ94]. The right normalization for the walk matrix comes from the work of Shi and Malik [SM00]. They suggest taking the left-eigenvectors, whereas we previously considered the right-eigenvectors (well, we did show that the all-1 vector is a left-eigenvector). So, let $\lambda_2, \ldots, \lambda_k$ be the $k-1$ largest non-trivial eigenvalues of $\boldsymbol{W} = \boldsymbol{A}\boldsymbol{D}^{-1}$ (other than $\lambda_1 = 1$), and let $\boldsymbol{v}_2, \ldots, \boldsymbol{v}_k$ be the corresponding left-eigenvectors. Now, set

$$x_i = (\boldsymbol{v}_2(i), \boldsymbol{v}_3(i), \ldots, \boldsymbol{v}_k(i)) .$$

We will try clustering the vertices of the graph by using k-means on these n vectors $x_1, \ldots, x_n$. There are principled reasons for doing this. But, rather than showing them to you, I will try to just give you some intuition as to why this might give good clusterings.

19.4 Drawing Graphs using Eigenvectors

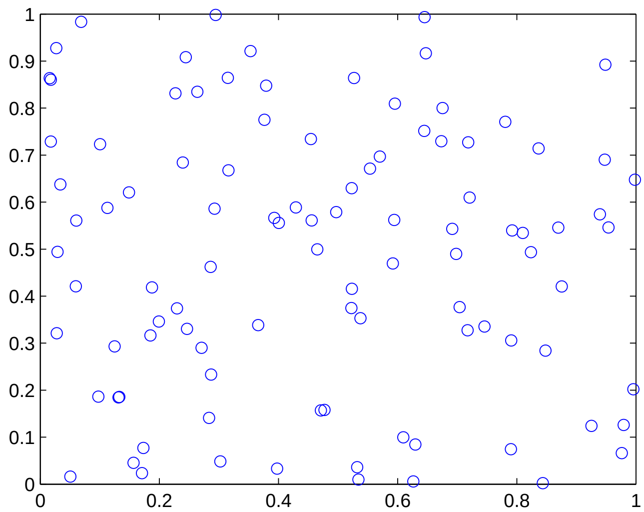
It turns out that if you want to draw a graph, a good easy way to do it is to take the two vectors $\boldsymbol{v}_2$ and $\boldsymbol{v}_3$, and locate vertex i as position

$$x_i = (\boldsymbol{v}_2(i), \boldsymbol{v}_3(i)) .$$

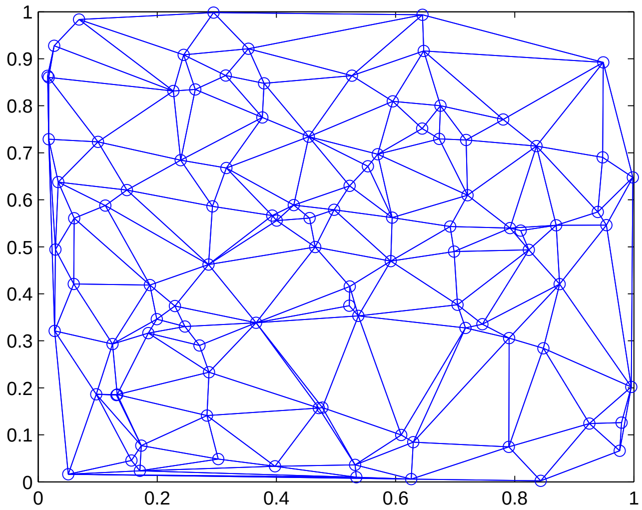
To convince you of this, let me show you the pictures this gives of some simple graphs. To draw the pictures, I will represent the edges as straight lines connecting the vertices.

To create my initial graph, I will choose 100 random points in the plane. I will then create a graph on them by taking their Delaunay triangulation.

```
>> [a,xy] = delGraph(100);
>> plot(xy(:,1),xy(:,2),'o')
```

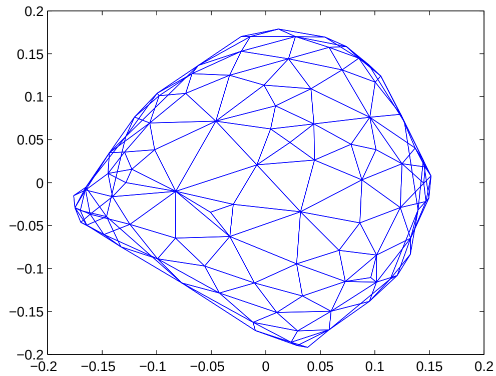


```
>> hold on
>> gplot(a,xy)
```



Now, I'll draw a picture of the graph using the first two non-trivial eigenvectors to obtain coordinates.

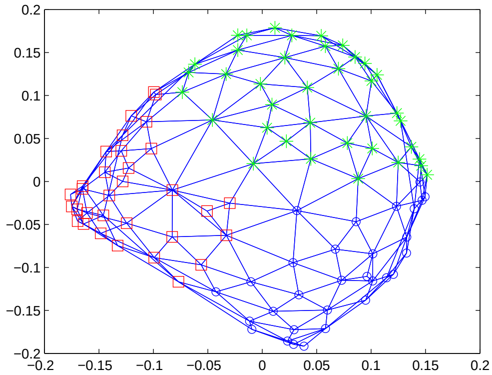
```
>> lap = diag(sum(a)) - a;
>> di = diag(1./sum(a));
>> [V,D] = eig(di*lap);
>> [val,ord] = sort(diag(D));
>> W = V(:,ord(2:3));
>> figure(2)
>> gplot(a,W)
```



I'd say that this gives a pretty good picture. Moreover, it is clear that if we run k-means on these coordinates, we will get a reasonable clustering of the vertices. Let's try it out. We'll create three clusters. I'll first plot them over the spectral picture, and then in original space.

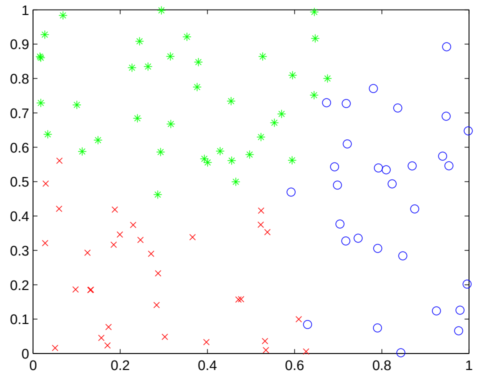
```
>> ide = kmeans(W,3);
```

```
>> figure(2)
>> hold on
>> plot(W(ide==1,1),W(ide==1,2),'o')
>> plot(W(ide==2,1),W(ide==2,2),'rs','MarkerSize',10)
>> plot(W(ide==3,1),W(ide==3,2),'g*','MarkerSize',10)
```



And, here it is in the original space.

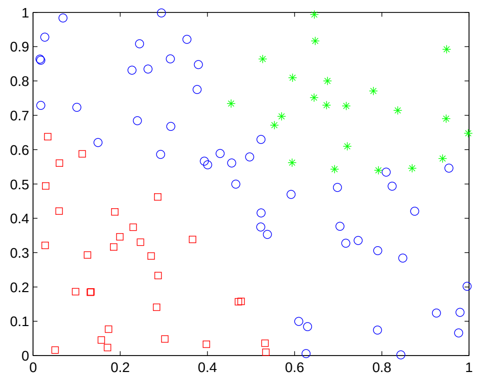
```
>> figure(1)
>> plot(xy(ide==1,1),xy(ide==1,2),'o')
>> hold on
>> plot(xy(ide==2,1),xy(ide==2,2),'rx')
>> plot(xy(ide==3,1),xy(ide==3,2),'g*')
```



Of course, we could use a more direct method to cluster points in the plane. I am not advocating using this method for that problem (although there are reasons to do something like this). Rather, I'm just trying to do an example in which it is visually clear that we are getting a reasonable answer.

Of course, I should compare this with using k-means on the adjacency matrix directly. Here is the result, plotted in the xy space.

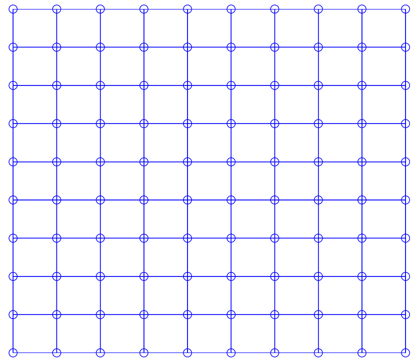
```
>> idx = kmeans(a,3);
>> figure(1)
>> clf
>> plot(xy(idx==1,1),xy(idx==1,2),'o')
>> hold on
>> plot(xy(idx==2,1),xy(idx==2,2),'rs')
>> plot(xy(idx==3,1),xy(idx==3,2),'g*')
```



It's not so bad, but I don't think it is as good as the spectral clustering.

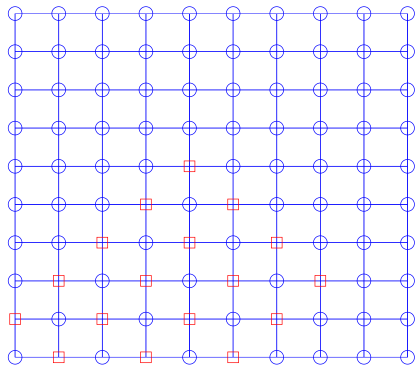
Now, let's see an example where using k-means directly does very poorly: on the grid graph. First, here's an image of this graph.

```
>> [a,jnk,xy] = grid2(10,10);
>> figure(1)
>> clf
>> plot(xy(:,1),xy(:,2),'o')
>> hold on
>> gplot(a,xy)
>> axis off
```



Let's cluster it with k-means.

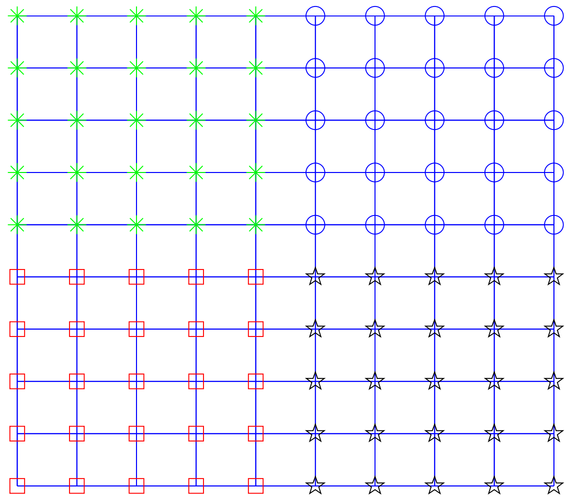
```
>> idx = kmeans(a,2);
>> clf
>> gplot(a,xy)
>> hold on; axis off
>> plot(xy(idx==1,1),xy(idx==1,2),'o','MarkerSize',10)
>> plot(xy(idx==2,1),xy(idx==2,2),'rs','MarkerSize',10)
```



I told you that k-means can do bad things on bipartite graphs.

Spectral partitioning, using k-means on the eigenvectors, gives almost perfect results for this graph. For overkill, I'll partition it into 4 pieces.

```
>> lap = diag(sum(a)) - a;
>> di = diag(1./sum(a));
>> [V,D] = eig(di*lap);
>> [val,ord] = sort(diag(D));
>> W = V(:,ord(2:4));
>> ide = kmeans(W,4);
>> clf
>> gplot(a,xy); hold on; axis of
>> plot(xy(ide==1,1),xy(ide==1,2
>> plot(xy(ide==2,1),xy(ide==2,2
>> plot(xy(ide==3,1),xy(ide==3,2
>> plot(xy(ide==4,1),xy(ide==4,2
```



19.5 Intrinsic Measures of Quality

We still need a way to measure the quality of a clustering. One way to start is to use a purely graph-theoretic measure.

Shi and Malik [SM00] advocate for the normalized cut measure:

$$\sum_a \frac{|\partial(C_a)|}{d(C_a)},$$

where we recall that $d(C_a)$ is the sum of the degrees of the vertices in C_a. In the case of two clusters, this has the advantage of exactly coinciding with a measure of conductance:

$$\begin{aligned}\frac{|\partial(C_0)|}{d(C_0)} + \frac{|\partial(C_1)|}{d(C_1)} &= \frac{|\partial(C_0)|}{d(C_0)} + \frac{|\partial(C_0)|}{d(C_1)} \\ &= (d(C_0) + d(C_1)) \frac{|\partial(C_0)|}{d(C_0)d(C_1)} \\ &= (d(V)) \frac{|\partial(C_0)|}{d(C_0)d(V - C_0)}.\end{aligned}$$

When I defined conductance I usually put the minimum in the denominator. But, it is common to take the product instead. I note that Shi and Malik [SM00] introduced their spectral clustering algorithm as a relaxation of the problem of minimizing the normalized cut objective function.

This is a variation of the k-way ratio cut measure introduced by Chan, Schlag and Zien [CSZ94]:

$$r(C_1, \ldots, C_k) \stackrel{\text{def}}{=} \sum_a \frac{|\partial(C_a)|}{|C_a|}.$$

Chan, Schlag and Zien [CSZ94] also derive their spectral clustering algorithm as a relaxation of this optimization problem.

In fact, Dhillon, Guan and Kulis [DGK04] have proved that there is a set of vectors that are naturally associated with a graph so that one minimizes the above quantity by optimizing the k-means objective function on those vertices. We get the vectors from the signed edge-vertex adjacency matrix (from Lecture 12):[1]

$$\boldsymbol{U}((a,b),c) = \begin{cases} 1 & \text{if } a = c \\ -1 & \text{if } b = c \\ 0 & \text{otherwise.} \end{cases}$$

This came up because the laplacian of an unweighted graph is given by

$$\boldsymbol{L} = \boldsymbol{U}^T \boldsymbol{U}.$$

We take the vector corresponding to vertex i to be the ith column of U.

The following result is proved by Dhillon, Guan and Kulis [DGK04].

Theorem 19.5.1. *For each vertex i, let x_i be the ith column of $\boldsymbol{U}$. The clustering $C_1, \ldots, C_k$ on these vectors that minimizes the k-means objective function is also the clustering that minimizes*

$$r(C_1, \ldots, C_k).$$

Proof. We first note that

$$x_i^T x_j = \begin{cases} -1 & \text{if } (i,j) \in E \\ d_i & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases}$$

So, for $i \neq j$,

$$\|x_i - x_j\|^2 = d_i + d_j - 2\mathbf{1}_{(i,j)\in E}.$$

In the following, I let $E(C_a)$ denote the set of edges between vertices in C_a, and recall that

$$d(C_a) = 2\,|E(C_a)| + |\partial(C_a)|\,.$$

For a cluster C_a,

$$\begin{aligned} \sum_{(i,j)\in C_a} \|x_i - x_j\|^2 &= (|C_a| - 1)\sum_i d_i + 2\,|E(C_a)| \\ &= |C_a| \sum_{i\in C_a} d_i - |\partial(C_a)|\,. \end{aligned}$$

So, the k-means objective function of a clustering $C_1, \ldots, C_k$ is

$$\begin{aligned} \sum_a \frac{1}{|C_a|} \sum_{(i,j)\in C_a} \|x_i - x_j\|^2 &= \sum_a \left(\sum_{i\in C_a} d_i + \frac{|\partial(C_a)|}{d(C_a)} \right) \\ &= 2m + r(C_1, \ldots, C_k). \end{aligned}$$

□

[1] In class, I thought that this was the unsigned version, but I was wrong.

The only problem with this result is that it is fragile. When one exatly optimizes the k-means objective function, one minimizes the k-way ratio cut score. But, if one merely approximately optimizes the k-means objective function then one can be very far from the optimum of the k-way ratio cut objective function.

19.6 Extrinsic Measures of Quality

Of course, the measure of quality one uses should really be motivated by an application. The previous measures were loosely motivated by applications in scientific computing.

Let's try an example where we can get a different measure of quality. For my graph, I will use some data from the Netflix prize problem. I will take the 500 most popular movies, and a set of 25,000 people. I will put an edge between two movies if they were both watched by the same person. Since this graph is really dense, I will do this with weights. So, the weight of the edge between two movies will be the number of people who watched both movies.

To get an extrinsic measure of quality, I have downloaded data from the Internet Movie Database about the genres of each movie. Each movie can have multiple genres. But, each genre can be viewed as a 0/1 vector: 1 for the movies that fit that genre and 0 for those that don't. In fact, I can view each movie as a 0/1 vector in genre space. I will measure the quality of a clustering on the movies by the score of the k-means objective function in genre space.

Let's try it, first by running k-means clustering directly on the adjacency matrix.

```
>> load netGraph
>> idx = kmeans(movAdj,20);
>> kmObj(genmat, idx)

ans =

  941.3839
```

We will get somewhat different results each time we try this. Here are the results of 10 runs.

```
>> for i = 1:10,
idx = kmeans(movAdj,20);
o(i) = kmObj(genmat, idx);
end

>> o

o =
```

```
  Columns 1 through 8

  948.9249  945.2935  942.1254  942.9548  941.9828  951.3510  940.4278  947.9509

  Columns 9 through 10

  937.8062  947.6053
```

If we believe that low normalized cuts should be better, then a natural idea would be to try to take this clustering and decrease the value of its normalized cut. I implemented some code for doing this. It goes through the vertices one-by-one, and moves each vertex to a different cluster if doing so would decrease the normalized cut objective function. We will do this now, and then check what it does for the division of the genres.

```
>> id2 = refineNcut(movAdj, idx);

ncutScore =

   18.2411


ncutScore =

   17.9504

>> kmObj(genmat,id2)

ans =

  868.1222
```

That's a big improvement, and constitutes evidence that the normalized cut objective function makes sense

Now, let's try it with the spectral method. Again, I'll do 10 runs.

```
>> lap = diag(sum(movAdj)) - movAdj;
>> di = diag(1./sum(movAdj));
>> [V,D] = eig(di*lap);
>> [val,ord] = sort(diag(D));
>> W = V(:,ord(2:20));
>> for i = 1:10,
idx = kmeans(W,20);
os(i) = kmObj(genmat, idx);
```

```
end
>> os

os =

  Columns 1 through 8

  841.3696  844.2227  865.8461  839.6322  868.7224  839.2108  853.4619  862.7531

  Columns 9 through 10

  850.1907  838.8691
```

Not only is this faster, but the scores we get are significantly better.

I'll now take the last cluster, and try out my local improvement algorithm. We will see that when we decrease the normalized cut objective function we improve the partitioning of the generes.

```
>> kmObj(genmat,idx)

ans =

  838.8691

>> id2 = refineNcut(movAdj, idx);

ncutScore =

   17.9266

ncutScore =

   17.8919

>> id2 = refineNcut(movAdj, id2);
>> id2 = refineNcut(movAdj, id2);
>> id2 = refineNcut(movAdj, id2);
>> id2 = refineNcut(movAdj, id2);
>> id2 = refineNcut(movAdj, id2);

ncutScore =

   17.8290
```

```
ncutScore =

   17.8282

>> kmObj(genmat,id2)

ans =

  829.5922
```

We again see improvement!

Let me point out that it is not clear that we should be using $k-1$ eigenvectors when we want k clusters. In the following experiment, I used just 9 eigenvectors. The performance on the genres is better.

```
>> lap = diag(sum(movAdj)) - movAdj;
>> di = diag(1./sum(movAdj));
>> [V,D] = eig(di*lap);
>> [val,ord] = sort(diag(D));
>> W = V(:,ord(2:10));
>> for i = 1:10,
idx = kmeans(W,20);
os(i) = kmObj(genmat, idx);
end
>> os

os =

  Columns 1 through 9

  812.8180  827.1396  822.9371  825.3779  821.6810  829.1225  828.9523  836.0724  817.62

  Column 10

  823.6171
```

It is worth looking at the clusters we actually get.

```
>> dispTitles(movTitles,idx);

--- Cluster 1 ---                  --- Cluster 11 ---
Pretty Woman                       Star Wars: Episode V: The Empi
Sweet Home Alabama                 Star Wars: Episode VI: Return
```

```
What Women Want
How to Lose a Guy in 10 Days
Sister Act
Two Weeks Notice
Dirty Dancing
The Wedding Planner
Mr. Deeds
Maid in Manhattan
Patch Adams
Bringing Down the House
Runaway Bride
Stepmom
Coyote Ugly
Cocktail

--- Cluster 2 ---
I, Robot
Shrek 2
Troy
The Bourne Supremacy
The Terminal
Spider-Man 2
Man on Fire
Collateral
Dodgeball: A True Underdog Sto
Kill Bill: Vol. 2
Napoleon Dynamite
Eternal Sunshine of the Spotle
The Manchurian Candidate
Anchorman: The Legend of Ron B
The Stepford Wives
Mean Girls
Fahrenheit 9/11
Hidalgo
Starsky & Hutch
Van Helsing
Paycheck
Super Size Me
The Village
Shark Tale
The Passion of the Christ
Elf
Taking Lives
Raising Helen
The Forgotten
Star Wars: Episode IV: A New H
Lord of the Rings: The Two Tow
The Lord of the Rings: The Fel
Lord of the Rings: The Return

--- Cluster 12 ---
American Beauty
Pulp Fiction
The Royal Tenenbaums
Memento
Fight Club
The Usual Suspects
Being John Malkovich
Adaptation
Seven
Traffic
Reservoir Dogs
Office Space
Crouching Tiger, Hidden Dragon
Raising Arizona
Amelie
Monty Python and the Holy Grai
The Big Lebowski
O Brother, Where Art Thou?
Edward Scissorhands
Best in Show
12 Monkeys
Boogie Nights
American History X
Snatch
Punch-Drunk Love
Dogma
High Fidelity
Rushmore
L.A. Confidential
Election
Almost Famous
Blow
A Fish Called Wanda
What's Eating Gilbert Grape
Magnolia
Clerks
Sling Blade
Secretary
This Is Spinal Tap
```

```
Cellular
Sky Captain and the World of T
Walking Tall
Bad Santa
Without a Paddle
The Punisher
Jersey Girl
Hero
The Girl Next Door
Hellboy
The Chronicles of Riddick
Saved!

--- Cluster 3 ---
Finding Nemo (Widescreen)
Monsters, Inc.
Shrek (Full-screen)
Harry Potter and the Chamber o
Harry Potter and the Sorcerer'
Harry Potter and the Prisoner
Ice Age
The Wizard of Oz: Collector's
The Goonies
A Bug's Life
Willy Wonka & the Chocolate Fa
The Lion King: Special Edition
Mary Poppins
Aladdin: Platinum Edition
Toy Story

--- Cluster 4 ---
My Big Fat Greek Wedding
Catch Me If You Can
Chicago
A Beautiful Mind
Bend It Like Beckham
Monster's Ball
About Schmidt
Bowling for Columbine
Chocolat
Bridget Jones's Diary
Whale Rider
About a Boy
I Am Sam
The Hours
Donnie Darko
Run Lola Run

--- Cluster 13 ---
Big
Jerry Maguire
Dead Poets Society
Philadelphia
A League of Their Own
As Good as It Gets
Good Morning, Vietnam
When Harry Met Sally
Fried Green Tomatoes
Rocky
The Sound of Music
Driving Miss Daisy
Tootsie
City Slickers
Finding Forrester
Notting Hill
Field of Dreams
While You Were Sleeping
Basic Instinct
The American President
An Officer and a Gentleman
Mr. Holland's Opus
Scent of a Woman
Gone with the Wind: Collector'
Flatliners
Terms of Endearment
On Golden Pond

--- Cluster 14 ---
The Green Mile
Indiana Jones and the Last Cru
The Fugitive
A Few Good Men
Lethal Weapon
Clear and Present Danger
Patriot Games
Lethal Weapon 2
Lethal Weapon 3
Kiss the Girls
The Devil's Advocate
Ransom
```

```
Serendipity
Shakespeare in Love
Moulin Rouge
The Pianist
Life Is Beautiful
The Good Girl
Frida
The Cider House Rules
In the Bedroom
Dead Man Walking
The Full Monty
Zoolander

--- Cluster 5 ---
Minority Report
Road to Perdition
Phone Booth
Gangs of New York
Signs
Identity
The Count of Monte Cristo
Old School
Black Hawk Down
One Hour Photo
Die Another Day
Training Day
The Ring
Red Dragon
We Were Soldiers
Frequency
Panic Room
Austin Powers in Goldmember
Daredevil
The Others
Insomnia
Windtalkers
Basic
Vanilla Sky
Shallow Hal
Unfaithful
A.I. Artificial Intelligence
Divine Secrets of the Ya-Ya Si
The Rookie
28 Days Later
Analyze That
Crimson Tide
Legends of the Fall
The Negotiator
The Pelican Brief
A Time to Kill
U.S. Marshals
In the Line of Fire
Rules of Engagement

--- Cluster 15 ---
Ferris Bueller's Day Off
Meet the Parents
American Pie
Ghostbusters
The Breakfast Club
There's Something About Mary:
Happy Gilmore
The Princess Bride
Austin Powers: The Spy Who Sha
Liar Liar
Austin Powers: International M
The Wedding Singer
National Lampoon's Vacation
Caddyshack
Tommy Boy
Spaceballs
Ace Ventura: Pet Detective
Beetlejuice
National Lampoon's Animal Hous
Sixteen Candles
Groundhog Day
Airplane!
Blazing Saddles
Billy Madison
Fast Times at Ridgemont High
Trading Places
Stripes
Wayne's World
My Cousin Vinny
Risky Business
Deuce Bigalow: Male Gigolo

--- Cluster 16 ---
The Godfather
GoodFellas: Special Edition
```

```
Ghost Ship
Unbreakable
Hannibal
Just Married
Tears of the Sun

--- Cluster 6 ---
Titanic
Erin Brockovich
Sleepless in Seattle
Steel Magnolias
Pay It Forward
The Firm
The Family Man
Hook
City of Angels
Phenomenon
Beaches
Forever Young
The Bodyguard

--- Cluster 7 ---
Lord of the Rings: The Fellows
The Matrix
Spider-Man
The Matrix: Reloaded
X-Men
X2: X-Men United
The Terminator
Star Wars: Episode II: Attack
Terminator 3: Rise of the Mach
Batman
Blade
Terminator 2: Extreme Edition
Star Wars: Episode I: The Phan
Total Recall
The Fifth Element
Interview with the Vampire
The Matrix: Revolutions
Final Destination
Blade 2
Underworld

--- Cluster 8 ---
Pirates of the Caribbean: The
```

```
One Flew Over the Cuckoo's Nes
Apocalypse Now
The Shining
Taxi Driver
The Graduate
A Clockwork Orange
The Godfather, Part II
Full Metal Jacket
To Kill a Mockingbird
Platoon
Blade Runner
Scarface: 20th Anniversary Edi
Citizen Kane
Psycho
Amadeus
Dr. Strangelove
Rear Window
2001: A Space Odyssey
The Exorcist
Chinatown
Unforgiven
Annie Hall

--- Cluster 17 ---
You've Got Mail
Legally Blonde
Father of the Bride
Big Daddy
Grease
The Waterboy
Coming to America
American Pie 2
Three Men and a Baby
My Best Friend's Wedding
Kindergarten Cop
Beverly Hills Cop II
Beverly Hills Cop
The Princess Diaries (Widescre
Turner and Hooch
Never Been Kissed
The First Wives Club
Overboard
Dr. Dolittle

--- Cluster 18 ---
```

Bruce Almighty
Ocean's Eleven
The Bourne Identity
The Italian Job
Lost in Translation
Lord of the Rings: The Two Tow
50 First Dates
Mystic River
Kill Bill: Vol. 1
The Last Samurai
Lord of the Rings: The Return
Big Fish
Something's Gotta Give
Anger Management
The School of Rock
Cold Mountain
Seabiscuit
The Butterfly Effect: Director
Master and Commander: The Far
13 Going on 30
Runaway Jury
Radio
Cheaper by the Dozen
Along Came Polly
Love Actually
Mona Lisa Smile
Secondhand Lions
Matchstick Men
Monster
Under the Tuscan Sun
Secret Window
Gothika
Freaky Friday
Daddy Day Care
House of Sand and Fog
Out of Time
Miracle
21 Grams
American Wedding
Legally Blonde 2: Red, White &
The Whole Nine Yards
Once Upon a Time in Mexico
The Missing

--- Cluster 9 ---
Top Gun
The League of Extraordinary Ge
The Sum of All Fears
Face/Off
The Mummy Returns
Rush Hour 2
Broken Arrow
Bad Boys II
Die Hard With a Vengeance
Lara Croft: Tomb Raider: The C
Murder By Numbers
Behind Enemy Lines
Rush Hour
XXX: Special Edition
Die Hard 2: Die Harder
Bad Boys
Big Momma's House
Wild Wild West
Hollow Man

--- Cluster 19 ---
National Treasure
The Incredibles
Sideways
The Notebook
Ocean's Twelve
Hitch
Ray
The Aviator
Finding Neverland
Meet the Fockers
Million Dollar Baby
Spanglish
Garden State
Hotel Rwanda
Ladder 49
Lemony Snicket's A Series of U
Closer
Crash
Constantine
Coach Carter
Miss Congeniality 2: Armed and
Sahara
The Life Aquatic with Steve Zi
In Good Company

```
Forrest Gump
The Sixth Sense
Gladiator
The Shawshank Redemption: Spec
Braveheart
Saving Private Ryan
The Silence of the Lambs
Rain Man
Good Will Hunting
Raiders of the Lost Ark
Die Hard
Indiana Jones and the Temple o
Schindler's List
Apollo 13
Remember the Titans
Cast Away
The Hunt for Red October
Stand by Me
Back to the Future
Dances With Wolves: Special Ed
E.T. the Extra-Terrestrial: Th
Tombstone

--- Cluster 10 ---
Miss Congeniality
Independence Day
The Patriot
The Day After Tomorrow
Con Air
Twister
Pearl Harbor
Armageddon
The Rock
Lethal Weapon 4
Gone in 60 Seconds
Men of Honor
Double Jeopardy
John Q
Swordfish
Men in Black II
Ghost
Air Force One
Tomb Raider
Entrapment
S.W.A.T.
Sin City
Batman Begins
The Longest Yard
Shall We Dance?
Be Cool

--- Cluster 20 ---
Men in Black
Jurassic Park
Mission: Impossible
Speed
The Mummy
Jaws
True Lies
Mission: Impossible II
What Lies Beneath
The Perfect Storm
Mrs. Doubtfire
The Nutty Professor
U-571
The Lost World: Jurassic Park
Gremlins
A Knight's Tale
Nine to Five
Planet of the Apes
Close Encounters of the Third
Charlie's Angels
```

```
Enemy of the State
The General's Daughter
The Fast and the Furious
The Recruit
Along Came a Spider
The Bone Collector
Don't Say a Word
High Crimes
The Net
Collateral Damage
```

References

[AV07] David Arthur and Sergei Vassilvitskii. k-means++: the advantages of careful seeding. *SODA '07: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, Jan 2007.

[CSZ94] P.K Chan, M.D.F Schlag, and J.Y Zien. Spectral k-way ratio-cut partitioning and clustering. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on DOI - 10.1109/43.310898*, 13(9):1088–1096, 1994.

[DGK04] Inderjit Dhillon, Yuqiang Guan, and Brian Kulis. Kernel k-means: spectral clustering and normalized cuts. *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, Aug 2004.

[Llo82] S Lloyd. Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129 – 137, 1982.

[Lux07] U Von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[SM00] J. B. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(8):888–905, August 2000.