

# Cluster Analysis

## Cluster Analysis

defining typical clusters  
simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application: earthquakes

- 1 Cluster Analysis**
  - defining typical clusters
  - simple turtle graphics
  - saving data to file
- 2 the K-means Algorithm**
  - reading data from file
  - the K-means algorithm
  - running the K-means algorithm
- 3 Data from the Web**
  - an application: earthquakes

MCS 275 Lecture 6  
Programming Tools and File Management  
Jan Vershelde, 25 January 2010

# Cluster Analysis

## Cluster Analysis

### defining typical clusters

simple turtle graphics  
saving data to file

### the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

### Data from the Web

an application:  
earthquakes

1 Cluster Analysis  
defining typical clusters  
simple turtle graphics  
saving data to file

2 the K-means Algorithm  
reading data from file  
the K-means algorithm  
running the K-means algorithm

3 Data from the Web  
an application: earthquakes

Cluster  
Analysisdefining typical  
clusters

simple turtle graphics  
saving data to file

the K-means  
Algorithm

reading data from file  
the K-means  
algorithm  
running the K-means  
algorithm

Data from the  
Web

an application:  
earthquakes

# Cluster Analysis

defining typical clusters

Goal of data mining: discover patterns, correlate data.

Example: link between homework and exam performance?

We will first generate our own data,  
using random number generators.

Each cluster is determined by 3 parameters:

- 1 the centroid is given by coordinates  $(x, y)$ ,
- 2 the number  $n$  of points in the cluster,
- 3 the radius is the largest distance of any point in the cluster to the centroid.

Cluster  
Analysisdefining typical  
clusters

simple turtle graphics  
saving data to file

the K-means  
Algorithm

reading data from file  
the K-means  
algorithm  
running the K-means  
algorithm

Data from the  
Web

an application:  
earthquakes

# Cluster Analysis

defining typical clusters

Goal of data mining: discover patterns, correlate data.

Example: link between homework and exam performance?

We will first generate our own data,  
using random number generators.

Each cluster is determined by 3 parameters:

- 1 the centroid is given by coordinates  $(x, y)$ ,
- 2 the number  $n$  of points in the cluster,
- 3 the radius is the largest distance of any point in the cluster to the centroid.

# Three Typical Clusters

## Cluster Analysis

### defining typical clusters

simple turtle graphics

saving data to file

## the K-means Algorithm

reading data from file

the K-means algorithm

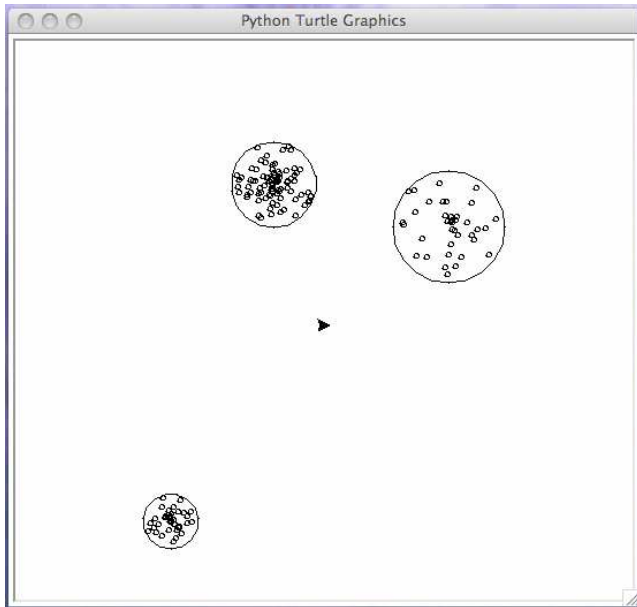
algorithm

running the K-means algorithm

## Data from the Web

an application:

earthquakes



# the script `makedata.py` will

## Cluster Analysis

### defining typical clusters

simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application:  
earthquakes

The script `makedata.py` will

- 1 prompt the user for the number of clusters either then typical clusters are generated, or for each cluster, the user is prompted for centroid, number of points, and radius;
- 2 visualize the data via turtle graphics;
- 3 prompt the user for a file name to save the data.

# the script `makedata.py`

## Cluster Analysis

### defining typical clusters

simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means  
algorithm  
running the K-means  
algorithm

## Data from the Web

an application:  
earthquakes

The script `makedata.py` will

- 1 prompt the user for the number of clusters  
either then typical clusters are generated,  
or for each cluster, the user is prompted for centroid,  
number of points, and radius;
- 2 visualize the data via turtle graphics;
- 3 prompt the user for a file name to save the data.

## the function Cluster

Cluster  
Analysisdefining typical  
clusterssimple turtle graphics  
saving data to filethe K-means  
Algorithmreading data from file  
the K-means  
algorithm  
running the K-means  
algorithmData from the  
Weban application:  
earthquakes

```

from math import cos, sin, pi

def Cluster(n,c,r):
    """
    Returns a list of n points with
    integer coordinates, in a disk with
    center c and radius r.
    """
    L = []; L.append(c)
    for k in range(1,n):
        rp = r*random.random()
        ap = 2*pi*random.random()
        x = c[0] + round(rp*cos(ap))
        y = c[1] + round(rp*sin(ap))
        L.append((x,y))
    return L

```

## the function Cluster

Cluster  
Analysisdefining typical  
clusterssimple turtle graphics  
saving data to filethe K-means  
Algorithmreading data from file  
the K-means  
algorithm  
running the K-means  
algorithmData from the  
Weban application:  
earthquakes

```
from math import cos, sin, pi

def Cluster(n,c,r):
    """
    Returns a list of n points with
    integer coordinates, in a disk with
    center c and radius r.
    """
    L = []; L.append(c)
    for k in range(1,n):
        rp = r*random.random()
        ap = 2*pi*random.random()
        x = c[0] + round(rp*cos(ap))
        y = c[1] + round(rp*sin(ap))
        L.append((x,y))
    return L
```

# Typical Data

## Cluster Analysis

defining typical clusters

simple turtle graphics

saving data to file

## the K-means Algorithm

reading data from file

the K-means algorithm

running the K-means algorithm

## Data from the Web

an application:

earthquakes

```
import random

def TypicalData():
    """
    Generates random numbers for (n,c,r).
    """
    R = []
    k = input("Give the number of clusters : ")
    for i in range(0,k):
        n = random.randint(10,100)
        c = (random.randint(-200,200),\
            random.randint(-200,200))
        r = random.randint(10,50)
        R.append(Cluster(n,c,r))
    return R
```

# Typical Data

## Cluster Analysis

defining typical  
clusters

simple turtle graphics

saving data to file

## the K-means Algorithm

reading data from file

the K-means

algorithm

running the K-means

algorithm

## Data from the Web

an application:

earthquakes

```
import random

def TypicalData():
    """
    Generates random numbers for (n,c,r).
    """
    R = []
    k = input("Give the number of clusters : ")
    for i in range(0,k):
        n = random.randint(10,100)
        c = (random.randint(-200,200),\
            random.randint(-200,200))
        r = random.randint(10,50)
        R.append(Cluster(n,c,r))
    return R
```

# script `makedata.py` as module

## Cluster Analysis

defining typical clusters

simple turtle graphics

saving data to file

## the K-means Algorithm

reading data from file

the K-means algorithm

algorithm

running the K-means algorithm

## Data from the Web

an application:

earthquakes

To visualize the data, we need to compute for each cluster:

- 1 the centroid for each list; and
- 2 the largest distance of each point in the list to the centroid.

The script `makedata.py` provides functions to compute centroids and radii, for use in the implementation of the K-means algorithm.

The last line of the script `makedata.py`:

```
if __name__=="__main__": main()
```

# script `makedata.py` as module

## Cluster Analysis

defining typical clusters

simple turtle graphics

saving data to file

## the K-means Algorithm

reading data from file

the K-means algorithm

running the K-means algorithm

## Data from the Web

an application: earthquakes

To visualize the data, we need to compute for each cluster:

- 1 the centroid for each list; and
- 2 the largest distance of each point in the list to the centroid.

The script `makedata.py` provides functions to compute centroids and radii, for use in the implementation of the K-means algorithm.

The last line of the script `makedata.py`:

```
if __name__=="__main__": main()
```

# List Comprehensions

## Cluster Analysis

defining typical clusters

simple turtle graphics

saving data to file

## the K-means Algorithm

reading data from file

the K-means algorithm

algorithm

running the K-means algorithm

## Data from the Web

an application: earthquakes

```
def Centroid(L):
```

```
    """
```

```
    Given a list L of tuples (x,y),
    returns the centroid of the points in L.
```

```
    """
```

is used in

```
def Centroids(L):
```

```
    """
```

```
    Given in L a list of lists of points,
    returns the list of centroids.
```

```
    """
```

```
    R = [Centroid(e) for e in L]
```

```
    return R
```

# List Comprehensions

## Cluster Analysis

defining typical clusters

simple turtle graphics

saving data to file

## the K-means Algorithm

reading data from file

the K-means algorithm

algorithm

running the K-means algorithm

## Data from the Web

an application:

earthquakes

```
def Centroid(L):
```

```
    """
```

```
    Given a list L of tuples (x,y),
    returns the centroid of the points in L.
```

```
    """
```

is used in

```
def Centroids(L):
```

```
    """
```

```
    Given in L a list of lists of points,
    returns the list of centroids.
```

```
    """
```

```
    R = [Centroid(e) for e in L]
```

```
    return R
```

# Cluster Analysis

## Cluster Analysis

defining typical clusters

**simple turtle graphics**

saving data to file

## the K-means Algorithm

reading data from file

the K-means algorithm

running the K-means algorithm

## Data from the Web

an application: earthquakes

earthquakes

1 Cluster Analysis  
defining typical clusters  
**simple turtle graphics**  
saving data to file

2 the K-means Algorithm  
reading data from file  
the K-means algorithm  
running the K-means algorithm

3 Data from the Web  
an application: earthquakes

# Simple Turtle Graphics

The `turtle` is a builtin module in Python.

We do **not** use the `cTurtle` module of the textbook!

The turtle commands we will use

```
import turtle
turtle.tracer(False)
turtle.up()
turtle.down()
turtle.goto(x,y)

turtle.circle(r)
turtle.color(p,f)
```

import the module  
for faster drawing  
put the pen up  
put the pen down  
goto position  $(x,y)$  and  
draw a line if pen is down  
draws a circle of radius  $r$   
sets pen and fill color  
where  $p$  and  $f$  are strings  
or rgb color codes

## Cluster Analysis

defining typical clusters

simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application: earthquakes

# Simple Turtle Graphics

The `turtle` is a builtin module in Python.

We do **not** use the `cTurtle` module of the textbook!

The turtle commands we will use

```
import turtle
turtle.tracer(False)
turtle.up()
turtle.down()
turtle.goto(x,y)
```

```
turtle.circle(r)
turtle.color(p,f)
```

import the module  
for faster drawing  
put the pen up  
put the pen down  
goto position  $(x, y)$  and  
draw a line if pen is down  
draws a circle of radius  $r$   
sets pen and fill color  
where  $p$  and  $f$  are strings  
or rgb color codes

## Cluster Analysis

defining typical clusters

simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application: earthquakes

# Simple Turtle Graphics

The `turtle` is a builtin module in Python.

We do **not** use the `cTurtle` module of the textbook!

The turtle commands we will use

```
import turtle
turtle.tracer(False)
turtle.up()
turtle.down()
turtle.goto(x,y)
```

```
turtle.circle(r)
turtle.color(p,f)
```

import the module  
for faster drawing  
put the pen up  
put the pen down  
goto position  $(x, y)$  and  
draw a line if pen is down  
draws a circle of radius  $r$   
sets pen and fill color  
where  $p$  and  $f$  are strings  
or rgb color codes

## Cluster Analysis

defining typical clusters

simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application: earthquakes

# Showing the Data Points

## Cluster Analysis

defining typical clusters

simple turtle graphics

saving data to file

## the K-means Algorithm

reading data from file

the K-means algorithm

running the K-means algorithm

## Data from the Web

an application: earthquakes

```
def ShowData(L):  
    """  
    For each point in L, draws  
    a circle of radius 2 pixels,  
    centered at the data point.  
    """  
    turtle.tracer(False)  
    for K in L:  
        for p in K:  
            turtle.up()  
            turtle.goto(p[0],p[1]-2)  
            turtle.down()  
            turtle.circle(2)  
    turtle.up()  
    turtle.goto(0,0)
```

# Showing the Data Points

## Cluster Analysis

defining typical clusters

simple turtle graphics

saving data to file

## the K-means Algorithm

reading data from file

the K-means algorithm

running the K-means algorithm

## Data from the Web

an application: earthquakes

```
def ShowData(L):  
    """  
    For each point in L, draws  
    a circle of radius 2 pixels,  
    centered at the data point.  
    """  
    turtle.tracer(False)  
    for K in L:  
        for p in K:  
            turtle.up()  
            turtle.goto(p[0],p[1]-2)  
            turtle.down()  
            turtle.circle(2)  
    turtle.up()  
    turtle.goto(0,0)
```

# Showing Centroids and Radii

## Cluster Analysis

defining typical clusters

simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application:  
earthquakes

```
def ShowCentroids(C,R):
    """
    Given on input in C a list of
    centroids and in R the
    corresponding radii, draws the disks.
    """
    for k in range(0,len(C)):
        c = C[k]
        r = R[k]+2
        turtle.up()
        turtle.goto(c[0],c[1]-r)
        turtle.down()
        turtle.circle(r)
    turtle.up()
```

# Showing Centroids and Radii

## Cluster Analysis

defining typical clusters

simple turtle graphics

saving data to file

## the K-means Algorithm

reading data from file

the K-means algorithm

running the K-means algorithm

## Data from the Web

an application: earthquakes

```
def ShowCentroids(C,R):
    """
    Given on input in C a list of
    centroids and in R the
    corresponding radii, draws the disks.
    """
    for k in range(0,len(C)):
        c = C[k]
        r = R[k]+2
        turtle.up()
        turtle.goto(c[0],c[1]-r)
        turtle.down()
        turtle.circle(r)
    turtle.up()
```

# Cluster Analysis

## Cluster Analysis

defining typical clusters  
simple turtle graphics  
**saving data to file**

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application: earthquakes

1 Cluster Analysis  
defining typical clusters  
simple turtle graphics  
**saving data to file**

2 the K-means Algorithm  
reading data from file  
the K-means algorithm  
running the K-means algorithm

3 Data from the Web  
an application: earthquakes

## saving data to file

Cluster  
Analysis

defining typical  
clusters  
simple turtle graphics  
**saving data to file**

the K-means  
Algorithm

reading data from file  
the K-means  
algorithm  
running the K-means  
algorithm

Data from the  
Web

an application:  
earthquakes

```
def SaveToFile(L):  
    """  
    Prompts the user for a file name  
    and writes the data to file.  
    """  
    name = raw_input("give a file name : ")  
    f = file(name, 'w')  
    f.write(str(L))  
    f.close()
```

Observe the string conversion with `str`.

# Cluster Analysis

## Cluster Analysis

defining typical clusters  
simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application: earthquakes

1 Cluster Analysis  
defining typical clusters  
simple turtle graphics  
saving data to file

2 the K-means Algorithm  
reading data from file  
the K-means algorithm  
running the K-means algorithm

3 Data from the Web  
an application: earthquakes

# reading data from file

## Cluster Analysis

defining typical clusters  
simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application: earthquakes

We start a new script `kmeans.py`:

- 1 prompt the user for a file name,
- 2 read in the data, flatten and shuffle,
- 3 run three iterations of the K-means Algorithm, showing the results with turtle.

Be careful with user input:

- 1 wrong file,
- 2 erroneous data on file.

*exception handling!*

# reading data from file

## Cluster Analysis

defining typical clusters  
simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application:  
earthquakes

We start a new script `kmeans.py`:

- 1 prompt the user for a file name,
- 2 read in the data, flatten and shuffle,
- 3 run three iterations of the K-means Algorithm, showing the results with turtle.

Be careful with user input:

- 1 wrong file,
- 2 erroneous data on file.

***exception handling!***

# repeat - until

## Cluster Analysis

defining typical  
clusters  
simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means  
algorithm  
running the K-means  
algorithm

## Data from the Web

an application:  
earthquakes

```
repeat
    try to open a file with given name
until success
```

is in Python encoded as

```
while True:
    try:
        f = open(name, 'r')
        break
    except:
        print "please try again ..."
```

## repeat - until

Cluster  
Analysis

defining typical  
clusters  
simple turtle graphics  
saving data to file

the K-means  
Algorithm

reading data from file  
the K-means  
algorithm  
running the K-means  
algorithm

Data from the  
Web

an application:  
earthquakes

```
repeat
    try to open a file with given name
until success
```

is in Python encoded as

```
while True:
    try:
        f = open(name, 'r')
        break
    except:
        print "please try again ..."
```

# Flowchart of repeat - until

## Cluster Analysis

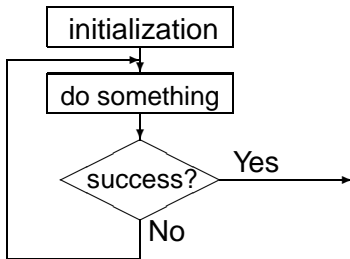
defining typical clusters  
simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application: earthquakes



# the function ReadData

## Cluster Analysis

defining typical clusters  
simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application: earthquakes

```
def ReadData():
    """
    Prompts the user for a name of a file,
    tries to open the file and to read a list.
    If all goes well, the list is returned.
    """
    L = []
    while True:
        name = raw_input("give file name : ")
        try:
            f = open(name, 'r')
            L = ReadList(f)
            f.close()
            return L
        except IOError:
            print 'cound not open file ' + name
            print 'please try again'
    return L
```

# the function ReadData

```
def ReadData():
    """
    Prompts the user for a name of a file,
    tries to open the file and to read a list.
    If all goes well, the list is returned.
    """
    L = []
    while True:
        name = raw_input("give file name : ")
        try:
            f = open(name, 'r')
            L = ReadList(f)
            f.close()
            return L
        except IOError:
            print 'cound not open file ' + name
            print 'please try again'
    return L
```

## Cluster Analysis

defining typical  
clusters  
simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means  
algorithm  
running the K-means  
algorithm

## Data from the Web

an application:  
earthquakes

# the function ReadData

## Cluster Analysis

defining typical clusters  
simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application: earthquakes

```
def ReadData():
    """
    Prompts the user for a name of a file,
    tries to open the file and to read a list.
    If all goes well, the list is returned.
    """
    L = []
    while True:
        name = raw_input("give file name : ")
        try:
            f = open(name, 'r')
            L = ReadList(f)
            f.close()
            return L
        except IOError:
            print 'cound not open file ' + name
            print 'please try again'
    return L
```

# converting string to list

## Cluster Analysis

defining typical clusters  
simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application:  
earthquakes

```
def ReadList(file):  
    """  
    Attempts to read a list from file  
    and will return this list.  
    Returns the empty list if exceptions occur.  
    """  
    try:  
        s = file.readline()  
        L = eval(s)  
        if isinstance(L,list):  
            return L  
        else:  
            print s  
            print "no list on file..."  
            return []  
    except:  
        print "exception occurred during reading..."  
        return []
```

# converting string to list

## Cluster Analysis

defining typical clusters  
simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application: earthquakes

```
def ReadList(file):  
    """  
    Attempts to read a list from file  
    and will return this list.  
    Returns the empty list if exceptions occur.  
    """  
    try:  
        s = file.readline()  
        L = eval(s)  
        if isinstance(L,list):  
            return L  
        else:  
            print s  
            print "no list on file..."  
            return []  
    except:  
        print "exception occurred during reading..."  
        return []
```

# converting string to list

## Cluster Analysis

defining typical  
clusters  
simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means  
algorithm  
running the K-means  
algorithm

## Data from the Web

an application:  
earthquakes

```
def ReadList(file):  
    """  
    Attempts to read a list from file  
    and will return this list.  
    Returns the empty list if exceptions occur.  
    """  
    try:  
        s = file.readline()  
        L = eval(s)  
        if isinstance(L,list):  
            return L  
        else:  
            print s  
            print "no list on file..."  
            return []  
    except:  
        print "exception occurred during reading..."  
        return []
```

# converting string to list

## Cluster Analysis

defining typical clusters  
simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application: earthquakes

```
def ReadList(file):  
    """  
    Attempts to read a list from file  
    and will return this list.  
    Returns the empty list if exceptions occur.  
    """  
    try:  
        s = file.readline()  
        L = eval(s)  
        if isinstance(L,list):  
            return L  
        else:  
            print s  
            print "no list on file..."  
            return []  
    except:  
        print "exception occurred during reading..."  
        return []
```

# Cluster Analysis

## Cluster Analysis

defining typical clusters  
simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
**the K-means algorithm**  
running the K-means algorithm

## Data from the Web

an application: earthquakes

1 Cluster Analysis  
defining typical clusters  
simple turtle graphics  
saving data to file

2 the K-means Algorithm  
reading data from file  
**the K-means algorithm**  
running the K-means algorithm

3 Data from the Web  
an application: earthquakes

# the K-means algorithm

## Cluster Analysis

defining typical clusters  
simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application: earthquakes

Input:  $L$  a list of points;  
 $k$  the number of centroids.

- 1  $C :=$  a random selection of  $k$  points of  $L$ ;
  - 2 repeat
    - 1 for each  $p$  in  $L$ :  
if  $p$  closest to  $C_i$ , then put  $p$  in  $i$ th cluster;
    - 2 recompute centroids for each of the  $k$  clusters;
    - 3 show the clusters
- until clusters are stable.

## the K-means algorithm

Cluster  
Analysis

defining typical  
clusters  
simple turtle graphics  
saving data to file

the K-means  
Algorithm

reading data from file  
the K-means  
algorithm  
running the K-means  
algorithm

Data from the  
Web

an application:  
earthquakes

Input:  $L$  a list of points;

$k$  the number of centroids.

- 1  $C :=$  a random selection of  $k$  points of  $L$ ;
  - 2 repeat
    - 1 for each  $p$  in  $L$ :  
if  $p$  closest to  $C_i$ , then put  $p$  in  $i$ th cluster;
    - 2 recompute centroids for each of the  $k$  clusters;
    - 3 show the clusters
- until clusters are stable.

# Cluster Analysis

## Cluster Analysis

defining typical clusters  
simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application: earthquakes

1 Cluster Analysis  
defining typical clusters  
simple turtle graphics  
saving data to file

2 the K-means Algorithm  
reading data from file  
the K-means algorithm  
running the K-means algorithm

3 Data from the Web  
an application: earthquakes

# Input of K-means Algorithm

## Cluster Analysis

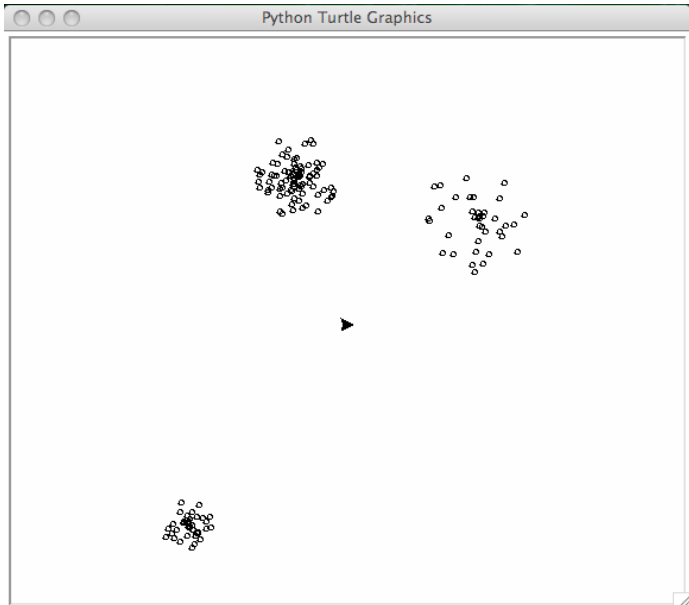
- defining typical clusters
- simple turtle graphics
- saving data to file

## the K-means Algorithm

- reading data from file
- the K-means algorithm
- running the K-means algorithm

## Data from the Web

- an application: earthquakes



# First Run of K-means

## Cluster Analysis

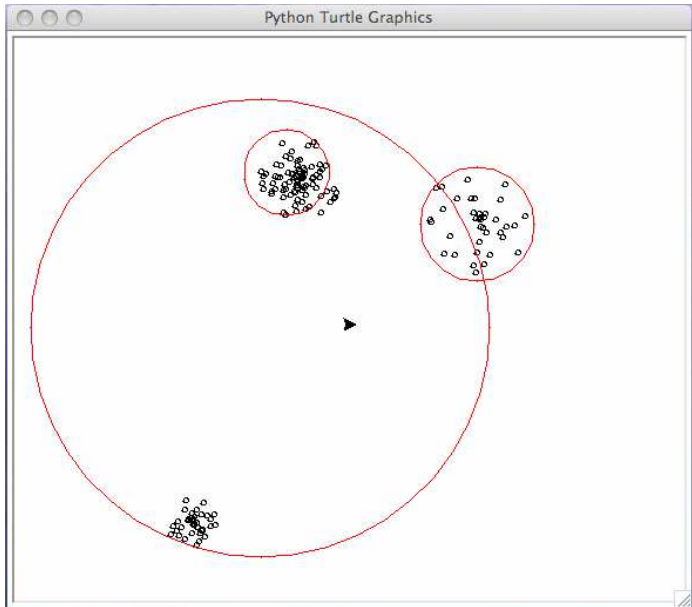
- defining typical clusters
- simple turtle graphics
- saving data to file

## the K-means Algorithm

- reading data from file
- the K-means algorithm
- running the K-means algorithm

## Data from the Web

- an application: earthquakes



# Second Run of K-means

## Cluster Analysis

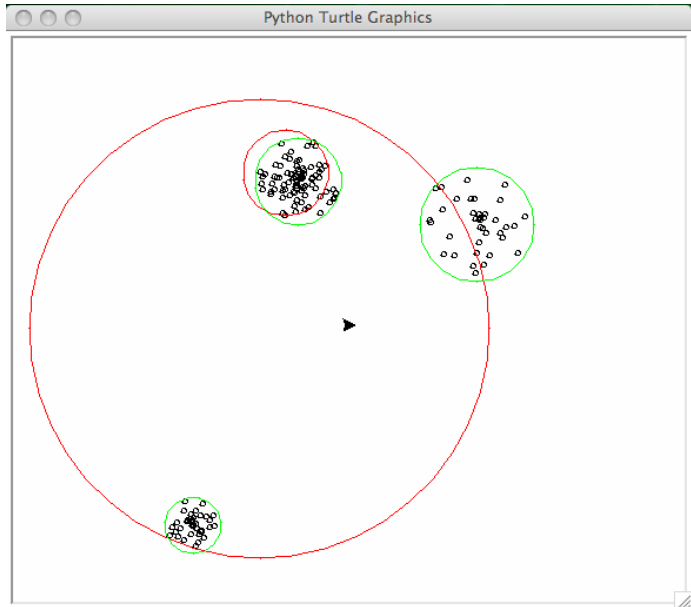
- defining typical clusters
- simple turtle graphics
- saving data to file

## the K-means Algorithm

- reading data from file
- the K-means algorithm
- running the K-means algorithm

## Data from the Web

- an application: earthquakes



# Third Run of K-means

## Cluster Analysis

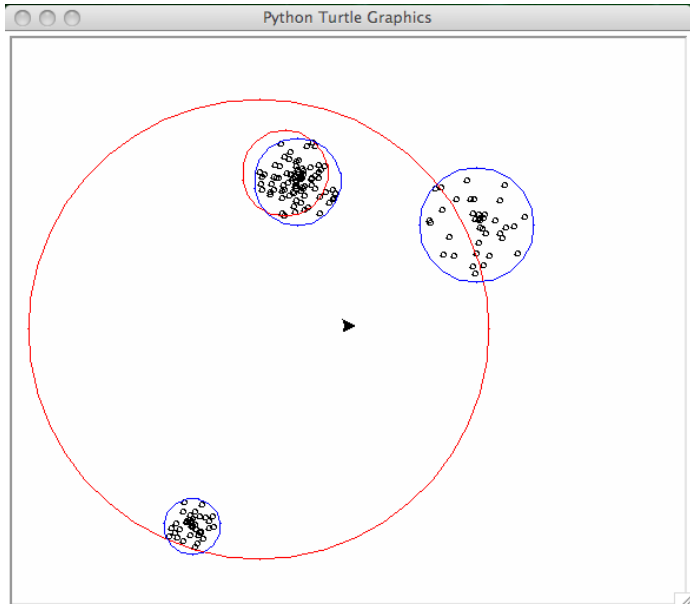
- defining typical clusters
- simple turtle graphics
- saving data to file

## the K-means Algorithm

- reading data from file
- the K-means algorithm
- running the K-means algorithm

## Data from the Web

- an application: earthquakes



# List Comprehensions again

## Cluster Analysis

defining typical clusters  
simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application: earthquakes

The function `Distance(p, q)` returns the Euclidean distance between the points `p` and `q`.

```
def ClassifyPoint(C,p):
    """
    Returns the index of the point q in C,
    closest to C.
    """
    d = [Distance(p,q) for q in C]
    m = min(d)
    return d.index(m)
```

For any list `L` and element `e`,  
we have `k = L.index(e): L[k] == e`

code for the `kmeans` functionCluster  
Analysis

defining typical  
clusters  
simple turtle graphics  
saving data to file

the K-means  
Algorithm

reading data from file  
the K-means  
algorithm  
running the K-means  
algorithm

Data from the  
Web

an application:  
earthquakes

To select the first centroids: `random.sample(L,k)`  
returns `k` unique random elements from `L`.

```
def kmeans(L,k):
    """
    Applies the K-means algorithm 3 times to
    divide the data in L into k clusters.
    """
    C0 = random.sample(L,k)
    K0 = ClassifyData(L,k,C0)
    C1 = Centroids(K0); R1 = Radii(K0,C1)
    turtle.color('red','red')
    ShowCentroids(C1,R1)
    ans = raw_input("hit enter to continue ...")
```

Cluster  
Analysis

defining typical  
clusters  
simple turtle graphics  
saving data to file

the K-means  
Algorithm

reading data from file  
the K-means  
algorithm  
running the K-means  
algorithm

Data from the  
Web

an application:  
earthquakes

In the second iteration, we classify the points with the newly calculated centroids:

```
K1 = ClassifyData(L,k,C1)
C2 = Centroids(K1); R2 = Radii(K1,C2)
turtle.color('green','green')
ShowCentroids(C2,R2)
ans = raw_input("hit enter to continue ...")
```

The third run:

```
K2 = ClassifyData(L,k,C2)
C3 = Centroids(K2); R3 = Radii(K2,C3)
turtle.color('blue','blue')
ShowCentroids(C3,R3)
```

# Cluster Analysis

## Cluster Analysis

defining typical clusters  
simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application: earthquakes

1 Cluster Analysis  
defining typical clusters  
simple turtle graphics  
saving data to file

2 the K-means Algorithm  
reading data from file  
the K-means algorithm  
running the K-means algorithm

3 Data from the Web  
an application: earthquakes

Cluster  
Analysis

defining typical  
clusters  
simple turtle graphics  
saving data to file

the K-means  
Algorithm

reading data from file  
the K-means  
algorithm  
running the K-means  
algorithm

Data from the  
Web

an application:  
earthquakes

# Data from the Web

US Geological Survey

The US government collects lots of data.

The US Geological Survey has its home  
at `http://www.usgs.gov`

Via `http://neic.usgs.gov/neis/qed/` we find data  
about recent earthquakes, from the last 8-30 days.

For this lecture, we are interested in raw data.

We can download comma separated values files  
with Python scripts.

script `getwebdata.py`Cluster  
Analysis

defining typical  
clusters  
simple turtle graphics  
saving data to file

the K-means  
Algorithm

reading data from file  
the K-means  
algorithm  
running the K-means  
algorithm

Data from the  
Web

an application:  
earthquakes

With `urlopen` of `urllib`: web page  $\sim$  file.

We download a comma separated values (csv) file:

```
from urllib import urlopen

u = "http://neic.usgs.gov/neis/gis/qed.asc"
infile = urlopen(u, 'r')
outfile = open("quakes.csv", 'w')
while True:
    s = infile.readline()
    if s == "": break
    outfile.write(s)
infile.close()
outfile.close()
```

script `getwebdata.py`

## Cluster

## Analysis

defining typical  
clusters  
simple turtle graphics  
saving data to file

## the K-means

## Algorithm

reading data from file  
the K-means  
algorithm  
running the K-means  
algorithm

## Data from the

## Web

an application:  
earthquakes

With `urlopen` of `urllib`: web page  $\sim$  file.

We download a comma separated values (csv) file:

```
from urllib import urlopen

u = "http://neic.usgs.gov/neis/gis/qed.asc"
infile = urlopen(u, 'r')
outfile = open("quakes.csv", 'w')
while True:
    s = infile.readline()
    if s == "": break
    outfile.write(s)
infile.close()
outfile.close()
```

the file `quakes.csv`Cluster  
Analysis

defining typical  
clusters  
simple turtle graphics  
saving data to file

the K-means  
Algorithm

reading data from file  
the K-means  
algorithm  
running the K-means  
algorithm

Data from the  
Web

an application:  
earthquakes

The start of `quakes.csv` looks like

```
Date,TimeUTC,Latitude,Longitude,Magnitude,Depth
2010/01/24,07:19:10.1,41.306,141.451,4.7, 40
2010/01/24,07:14:51.9,35.567,-97.284,3.7, 5
2010/01/24,06:49:50.0,-17.912,-178.351,4.7,531
2010/01/24,05:38:29.5,-8.235,129.007,5.4, 38
2010/01/24,02:36:14.4,35.498,110.626,5.0, 18
```

The coordinates of interest are latitude and longitude.

# Longitude and Latitude

quakes of magnitude  $\geq 5.0$

## Cluster Analysis

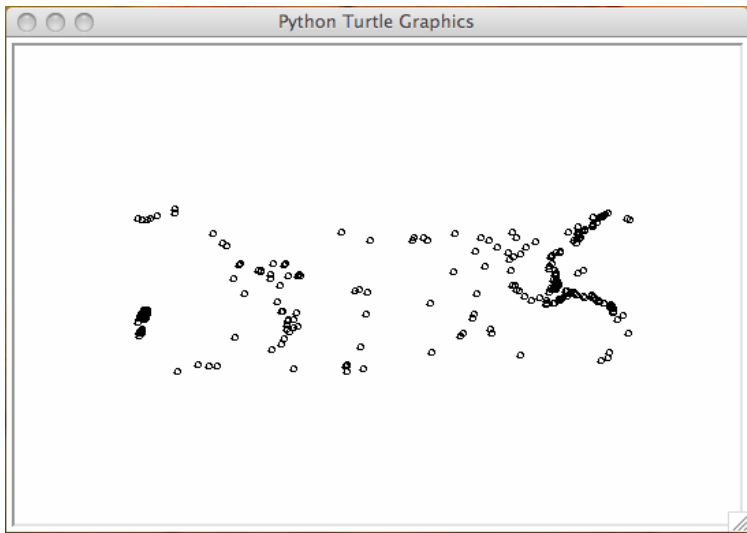
- defining typical clusters
- simple turtle graphics
- saving data to file

## the K-means Algorithm

- reading data from file
- the K-means algorithm
- running the K-means algorithm

## Data from the Web

- an application: earthquakes



# Processing csv Files

## Cluster Analysis

defining typical clusters  
simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application: earthquakes

```
file = "quakes.csv"
from makedata import ShowData
```

```
def SelectData(m):
    """
    Opens the file and returns list of
    longitudes and latitudes of quakes of
    magnitude larger than or equal to m.
    Because latitude runs vertically,
    latitude is y and longitude is x.
    """
    R = []
    f = open(file, 'r')
    s = f.readline()
```

# function SelectData continued...

again repeat - until

```
while True:
    s = f.readline()
    if s == '': break
    L = s.split(',')
    try:
        M = float(L[4])
    except:
        M = 0
    if M >= m:
        p = (float(L[3]),float(L[2]))
        R.append(p)
return R
```

For visualization with turtle we round and convert the coordinates to integer values.

# function SelectData continued...

again repeat - until

```
while True:
    s = f.readline()
    if s == '': break
    L = s.split(',')
    try:
        M = float(L[4])
    except:
        M = 0
    if M >= m:
        p = (float(L[3]), float(L[2]))
        R.append(p)
return R
```

For visualization with turtle we round and convert the coordinates to integer values.

# Summary and Exercises

## Cluster Analysis

defining typical clusters  
simple turtle graphics  
saving data to file

## the K-means Algorithm

reading data from file  
the K-means algorithm  
running the K-means algorithm

## Data from the Web

an application: earthquakes

Read Chapter 7 of *Python Programming in Context*.

- 1 Instead of turtle graphics, use a Canvas widget of Tkinter to show the data points.
- 2 One stop criterion for the K-means algorithm is to check whether no points change list after classifying from one iteration to the next. Give Python code of an implementation of the K-means algorithm that uses this stop criterion. You may ignore the visualization part.
- 3 Apply the K-means algorithm to `quakes.csv`. Experiment with various numbers of clusters.
- 4 With the method `bgcolor` of turtle we can set the background picture to a world map. Download a "worldmap.gif" file and rescale so the data from `quakes.csv` is mapped right.