**Clustering**

ZHE Xuefei SID:54171181

Department of Electronic Engineering

1. Implementation details of three clustering algorithm

(1)K-means

**function[**mu**,** label**]=**myKmeans**(**features**,**k**)**

**[**n**,**m**]=**size**(**features**);**

index**=**randperm**(**n**);**

index**=**index**(**1**:**k**);**

muOld**=**features**(**index**,:);**

disMu**=**zeros**(**n**,**k**);**

muMove**=**999**;**

label**=**zeros**(**n**,**1**);**

iter**=**0**;**

**while** muMove**>**0.001**&&**iter**<**10000

**for** i**=**1**:**k

disMu**(:,**i**)=**sum**((**features**-**repmat**(**muOld**(**i**,:),**n**,**1**)).^**2**,**2**);**%calculate distance

**end**

**[~,** I**]=**sort**(**disMu**,**2**,**'ascend'**);**

label**=**I**(:,**1**);**

mu**=**zeros**(**k**,**m**);**%get zij

**for** j**=**1**:**k

mu**(**j**,:)=**sum**(**features**(**label**==**j**,:))./**sum**(**label**==**j**);**

**end**

muMove**=**sum**(**sum**(**sqrt**((**mu**-**muOld**).^**2**)));**

muOld**=**mu**;**

iter**=**iter**+**1**;**

**end**

all initial points are randomly selected from original dataset.

(2)EM algorithm for Gaussian mixture models

**function[**mu**,** clu**]=**myEm**(**features**,**k**)**

**[**n**,**m**]=**size**(**features**);**

mu**=**zeros**(**k**,**m**);**

piOld**=**ones**(**k**,**1**);**

piOld**=(**1**/**k**).\***piOld**;**

ma **=** zeros**(**m**);** % max value

mi **=** zeros**(**m**);** % min value

mu **=** zeros**(**k**,**m**);**

% random initialization

**for** i**=**1**:**m

ma**(**i**)=**max**(**features**(:,**i**));**

mi**(**i**)=**min**(**features**(:,**i**));**

**for** j**=**1**:**k

mu**(**j**,**i**)=**mi**(**i**)+(**ma**(**i**)-**mi**(**i**))\***rand**();**

**end**

**end**

muOld**=**mu**;**

SigmaOld**=**ones**(**m**,**m**,**k**);**

**for** i**=**1**:**k

SigmaOld**(:,:,**i**)=**eye**(**m**,**m**);**

**end**

iter**=**0**;**

muMove**=**99**;**

Nj**=**zeros**(**1**,**k**);**

**while** muMove**>**0.01 **&&**iter**<**2000

z**=**zeros**(**n**,**k**);**

%-------------------E-step---------------------------------------

**for** i**=**1**:**n

sumZ**=**zeros**(**1**,**k**);**

**for** j**=**1**:**k

sumZ**(**j**)=+**piOld**(**j**)\***mvnpdf**(**features**(**i**,:),**muOld**(**j**,:),**SigmaOld**(:,:,**j**));**

**end**

temp**=**sum**(**sumZ**);**

**for** j**=**1**:**k

z**(**i**,**j**)=**sumZ**(**j**)/**temp**;**

**end**

**end**

%------------------M-STEP--------------------------------------

Nj**=**sum**(**z**,**2**);**

**for** j**=**1**:**k

piOld**(**j**)=**Nj**(**j**)/**n**;**

muTem**=**zeros**(**1**,**m**);**

Sigma**=**zeros**(**m**,**m**);**

summu**=**zeros**(**1**,**m**);**

temp**=**repmat**(**z**(:,**j**),**1**,**m**).\***features**;**

summu**=**sum**(**temp**);**

mu**(**j**,:)=**summu**./**Nj**(**j**);**

%%------------full covariance-------------

% for i=1:n

% Sigma=Sigma+z(i,j).\*(features(i,:)'-mu(j,:)')\*(features(i,:)'-mu(j,:)')';

% end

% SigmaOld(:,:,j)=Sigma./Nj(j);

**end**

%%------------iid covanriance-------------

s**=**zeros**(**m**,**1**);**

**for** i**=**1**:**k

temp**=**features**-**repmat**(**mu**(**i**,:),**n**,**1**);**

s**=**sum**(**repmat**(**z**(:,**j**),**1**,**m**).\***temp**.\***temp**)/**Nj**(**i**);**

SigmaOld**(:,:,**i**)=**diag**(**s**);**

**end**

iter**=**iter**+**1**;**

muMove**=**sum**(**sum**((**mu**-**muOld**).^**2**));**

**end**

mu**=**muOld**;**

**[~,** I**]=**sort**(**z**,**2**,**'ascend'**);**

clu**=**I**(:,**1**);**

**end**

Initialized points by adding random value to the minimum value of features. Ending iteration by compares mu’s position and limited the max iteration. Iid covariance trick also is used here.

(3)Mean-shift algorithm

**function** **[**k**,**label**]=**myMeanShift**(**features**,**h**)**

**[**n**,**m**]=**size**(**features**);**

xOld**=**features**;**

xNew**=**zeros**(**n**,**m**);**

label **=** zeros**(**n**,**1**);**

**for** ii**=**1**:**n

iter**=**0**;**

xMove**=**999**;**

**while** xMove**>**10**^(-**10**)&&**iter**<**1000

distance**=-**0.5.**\***sum**((**repmat**(**xOld**(**ii**,:)',**1**,**size**(**features**,**1**))-**features**').^**2**)./(**h**.^**2**);**

prob**=**exp**(**distance**);**

temp**=**sum**(**prob**);**

xNew**(**ii**,:)** **=** features**'\***prob**'/**temp**;**

xMove**=**norm**(**xNew**(**ii**,:)-**xOld**(**ii**,:));**

xOld**(**ii**,:)=**xNew**(**ii**,:);**

iter**=**iter**+**1**;**

show**=[**'iter='**,**num2str**(**iter**),**' '**,**'xMove='**,**num2str**(**xMove**)];**

disp**(**show**)**

**end**

**end**

mu**=**xNew**;**

%---------------Labeling--------------------

k**=**1**;**

label**(**1**)=**k**;**

**for** i**=**2**:**n

flag**=**0**;**

**for** j **=** 1**:**i**-**1

dist **=** norm**(**mu**(**i**,:)-**mu**(**j**,:));**

**if(**dist**<**1e-4**)**

label**(**i**)** **=** label**(**j**);**

flag **=** 1**;**

**break;**

**end**

**end**

**if(**flag **==** 0**)**

k **=** k**+**1**;**

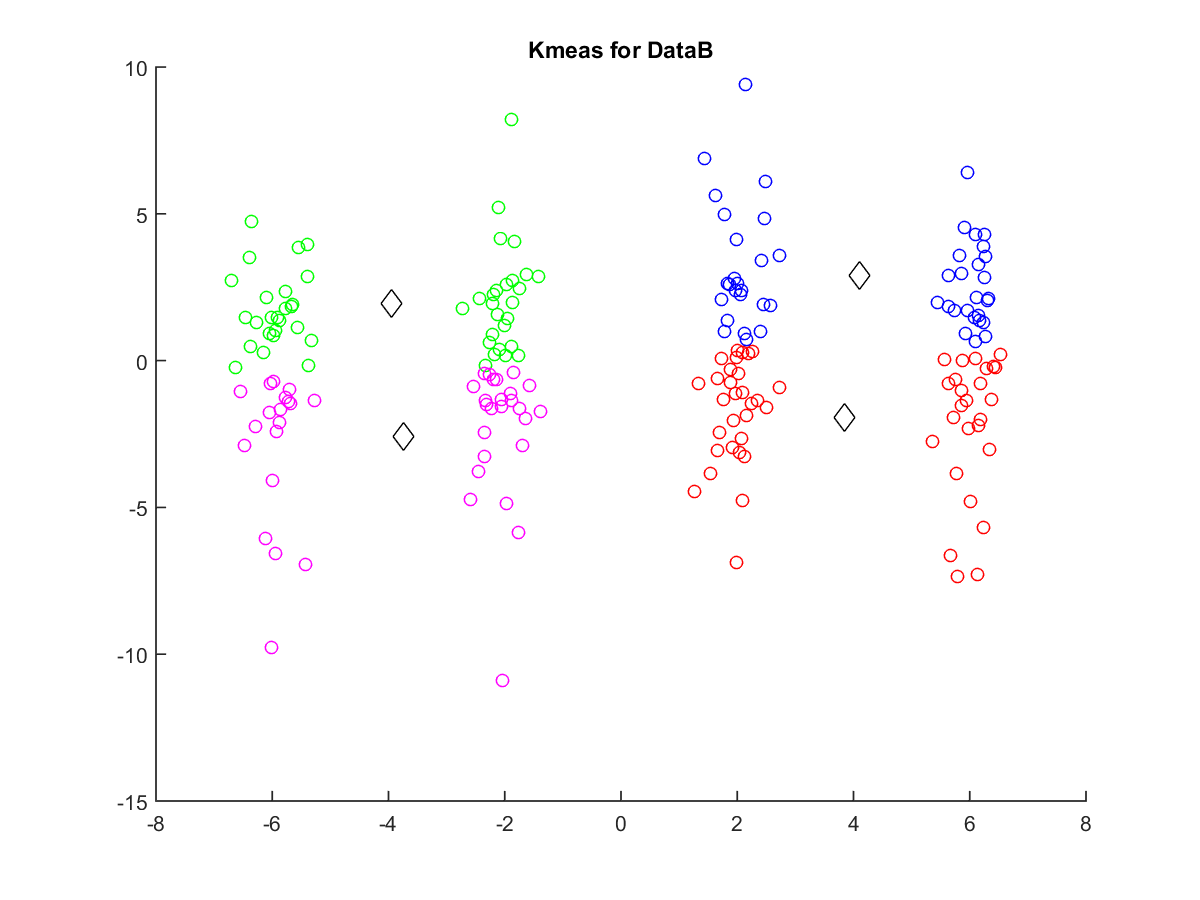
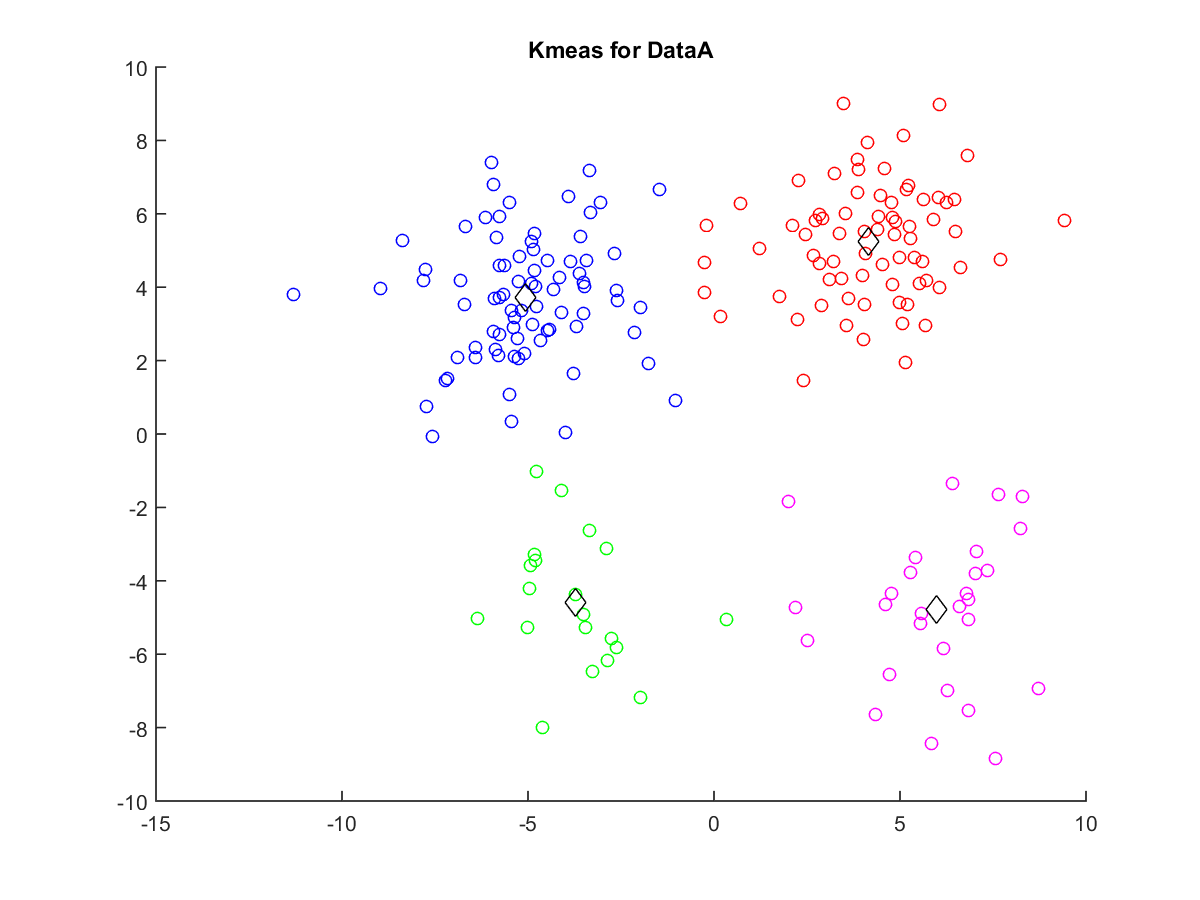
label**(**i**)** **=** k**;**

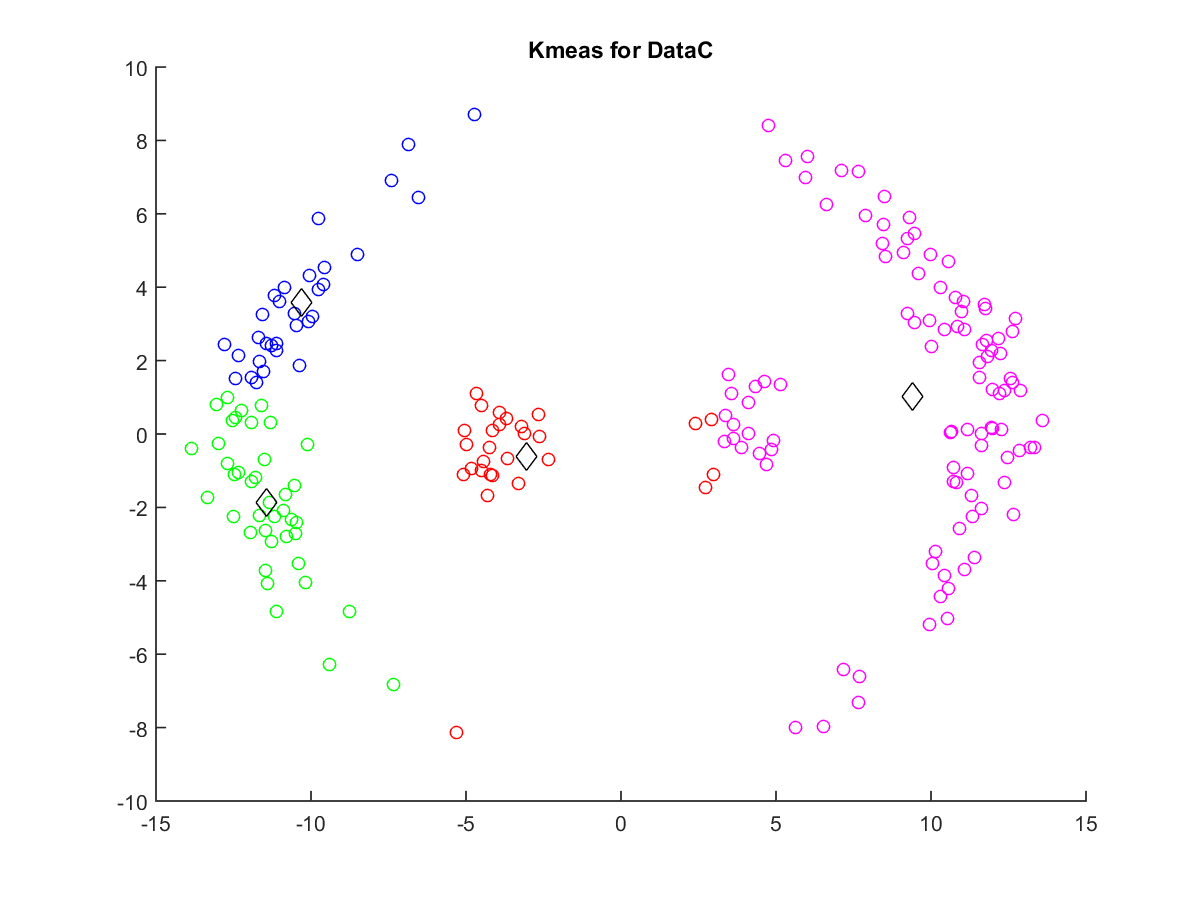
**end**

**end**

(b) Run the algorithms on the three synthetic datasets. Qualitatively, how does each clustering algorithm perform on each dataset? Comment on the advantages and limitations of each algorithm, in terms of the configuration of the data.

(1)K-means





The different clusters points are marked with different color. And the cluster centers are plotted with black diamond. For these figures, it can be found that k-means work well for DataA because there exist cluster centers which make all points in the same cluster have smallest distance to its own center than to others’ centers. But for DataB and DataC, k-means have bad performance. Because there is no cluster centers existed like dataA.