Unsupervised classification

Karina Nielsen (karni@space.dtu.dk) and Allan Aasbjerg Nielsen

September, 2021

Overview of module 4

- Feedback to PCA report
- Repetition of Supervised Classification
- Unsupervised classification - clustering
  - Kmeans
  - Gaussian mixture models (GMM)
- Exercises (Report, not mandatory, but strongly recommended to do)

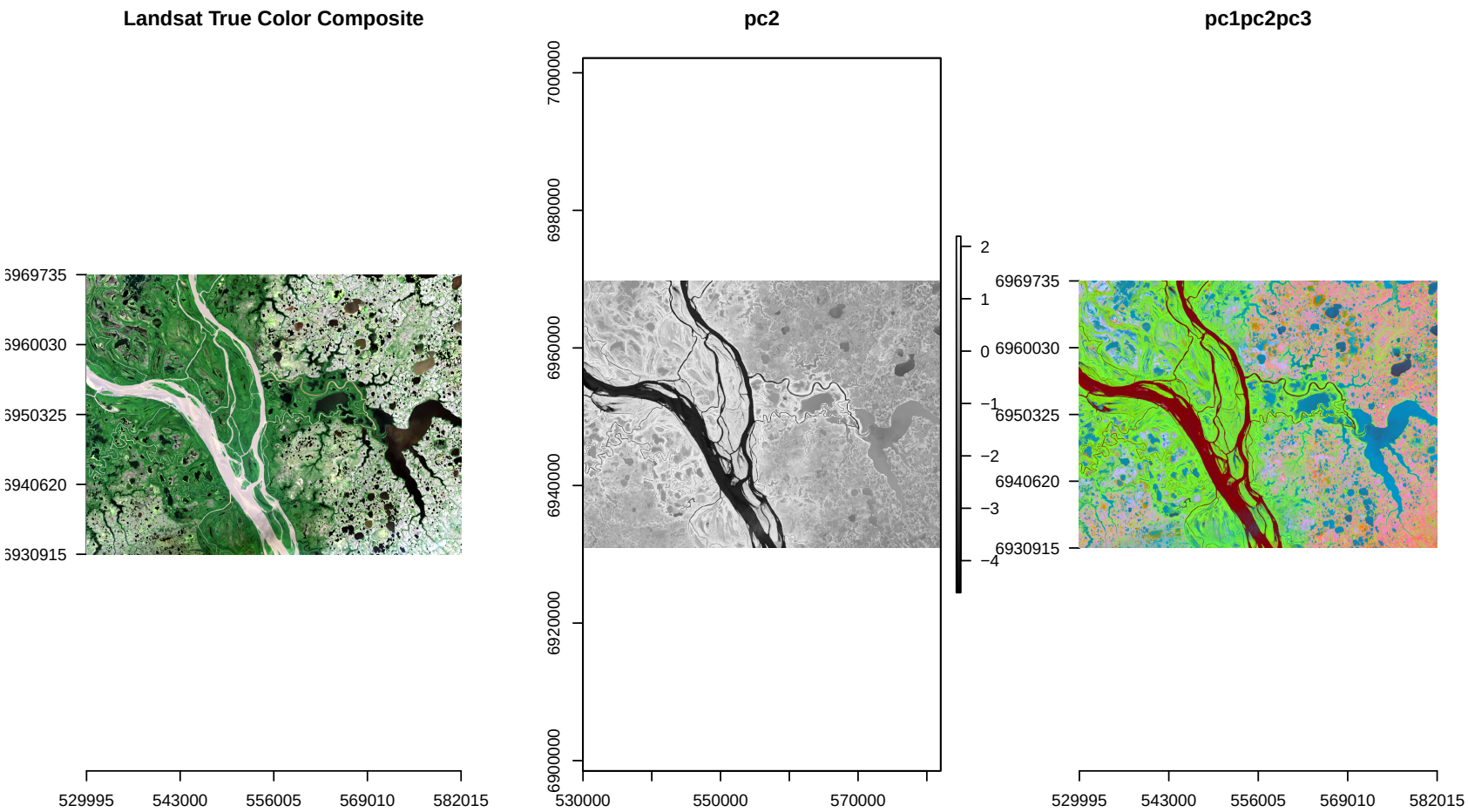
Feedback to PCA report

- All reports are generally very good
- It is okay that you do not write theory in the reports, but in an exam situation you should present the basic principles
- The signs of the eigenvectors are arbitrary so to see which band potentially dominate a PC you should consider the absolute load (eigenvector scaled by eigenvalue).

```
> t(t(pca$rotation)*pca$sdev)

PC1         PC2          PC3          PC4          PC5           PC6
B2 0.8430062 -0.5250497  0.005540832 -0.106939271  0.046207522 -0.0078415271
B3 0.9203413 -0.2868146 -0.243900124  0.098190122  0.038966354  0.0078964432
B4 0.8091997 -0.5803744  0.032015523  0.005587717 -0.085466045  0.0008731331
B5 0.6168833  0.7121853 -0.330494529 -0.046330177 -0.026751355 -0.0125837561
B6 0.8614312  0.4742780  0.175522897 -0.015792723  0.002917603  0.0439370215
B7 0.8837213  0.3715348  0.279012300  0.042372097  0.009429075 -0.0355875981
```

- Hence PC2 is dominated by the NIR band, which in my case is positive, we will there for expect that vegetation appears bright. If if was negative the vegetation would appear dark.

Yukon image

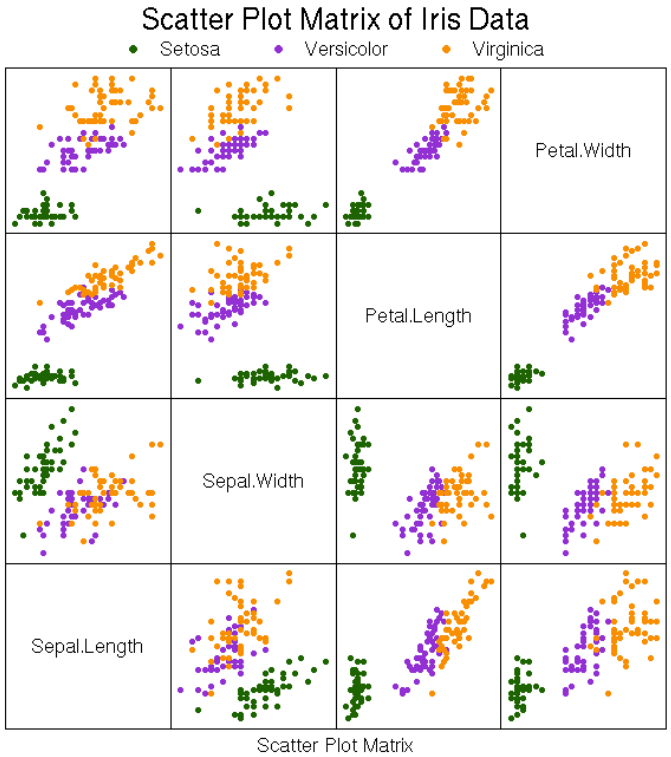

Landsat True Color Composite
6969735
6960030
6950325
6940620
6930915
529995
543000
556005
569010
582015
pc2
7000000
6980000
6960000
6940000
6920000
6900000
530000
550000
570000
2
1
0
−1
−2
−3
−4
pc1pc2pc3
6969735
6960030
6950325
6940620
6930915
529995
543000
556005
569010
582015

Classification

- classification is the process of grouping observations (pixels or regions) into classes intended to represent different physical objects or types
- here, the production of a **thematic map** from (image) data with digital numbers representing for example reflected or emitted EM-radiation in different wavelength bands
- very many classification methods ranging from quite simple to highly advanced
- two major groups of methods: supervised and unsupervised
  - supervised: ideally physical classes but not necessarily statistically distinct
  - unsupervised: statistically distinct but not necessarily physical classes
- In remote sensing:
  - supervised classification: Here we obtain information classes
  - unsupervised classification: Here we obtain spectral classes

Feature space

- p variables
  C classes
  N observations (or samples)
- $\boldsymbol{x}_i,\ i = 1, \ldots, N, \quad p \times 1$
  is a point (or vector) in p-dimensional **feature space**
- figure shows all possible pairwise projections on original variables



Supervised: Gaussian $P(\boldsymbol{X}|\omega_c)$

- Gaussian "likelihood" in 1-D

$$P(X|\omega_c) = \frac{1}{\sqrt{2\pi}} \frac{1}{\sigma_c} \exp\left[-\frac{1}{2}\left(\frac{X-\mu_c}{\sigma_c}\right)^2\right]$$

  σ_c^2 is variance for class c
- Gaussian "likelihood" in p-D

$$P(\boldsymbol{X}|\omega_c) = (2\pi)^{-p/2}|\boldsymbol{\Sigma}_c|^{-1/2} \exp\left[-\frac{1}{2}(\boldsymbol{X}-\boldsymbol{\mu}_c)^T\boldsymbol{\Sigma}_c^{-1}(\boldsymbol{X}-\boldsymbol{\mu}_c)\right]$$

  $\boldsymbol{\Sigma}_c$ is $p \times p$ dispersion or covariance matrix for class c
  $\boldsymbol{\Sigma}_c$ contains variances on diagonal and covariances off diagonal

Supervised: Gaussian $P(\boldsymbol{X}|\omega_c)$

- MAP: max $P(\omega_c|\boldsymbol{X}) \Leftrightarrow$ max log-likelihood $\mathcal{L}_c(\boldsymbol{X})$ (use Bayes' rule)

$$\begin{aligned}\mathcal{L}_c(\boldsymbol{X}) &= \ln P(\omega_c|\boldsymbol{X}) \\ &= \ln P(\omega_c) + \ln P(\boldsymbol{X}|\omega_c) - \ln \sum_{j=1}^{C} P(\boldsymbol{X}|\omega_j)P(\omega_j)\end{aligned}$$

- ln of sum in last term same for all c, drop it; insert Gaussian $\ln P(\boldsymbol{X}|\omega_c)$

$$\mathcal{L}_c(\boldsymbol{X}) \sim g_c(\boldsymbol{X}) = \ln P(\omega_c) - \frac{p}{2}\ln 2\pi - \frac{1}{2}\ln|\boldsymbol{\Sigma}_c| - \frac{1}{2}(\boldsymbol{X}-\boldsymbol{\mu}_c)^T\boldsymbol{\Sigma}_c^{-1}(\boldsymbol{X}-\boldsymbol{\mu}_c)$$

- drop $-\frac{p}{2}\ln 2\pi$
- if equal priors: drop $\ln P(\omega_c)$

Supervised: Gaussian $P(\boldsymbol{X}|\omega_c)$

- log-likelihood

$$\mathcal{L}_c(\boldsymbol{X}) \sim g_c(\boldsymbol{X}) = \ln P(\omega_c) - \frac{1}{2}\ln|\boldsymbol{\Sigma}_c| - \frac{1}{2}(\boldsymbol{X}-\boldsymbol{\mu}_c)^T\boldsymbol{\Sigma}_c^{-1}(\boldsymbol{X}-\boldsymbol{\mu}_c)$$

  quadratic in $\boldsymbol{X}$: quadratic discriminant analysis

- if equal dispersions, $\boldsymbol{\Sigma}_c = \boldsymbol{\Sigma}$: drop $-\frac{1}{2}\ln|\boldsymbol{\Sigma}| - \frac{1}{2}\boldsymbol{X}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{X}$

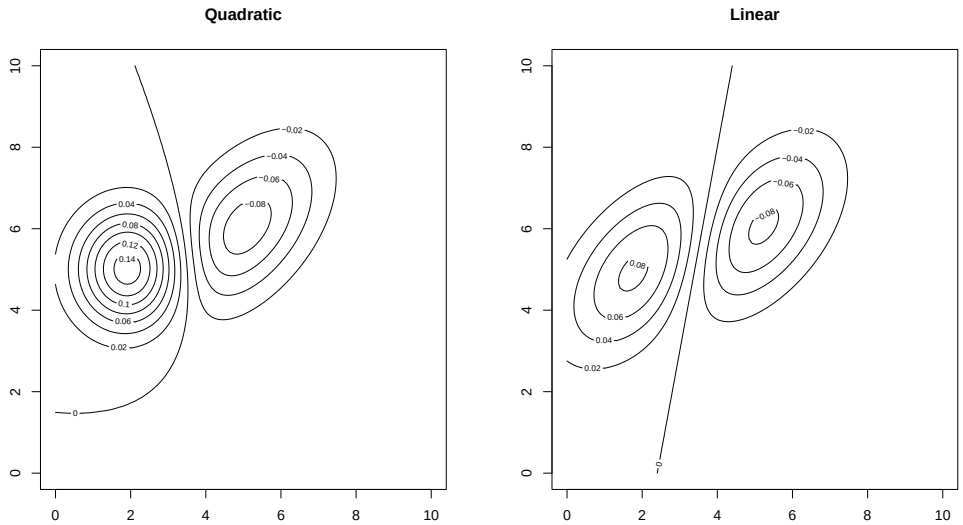
$$\mathcal{L}_c(\boldsymbol{X}) \sim g_c(\boldsymbol{X}) = \ln P(\omega_c) + \boldsymbol{\mu}_c^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{X} - \frac{1}{2}\boldsymbol{\mu}_c)$$

  linear in $\boldsymbol{X}$: linear discriminant analysis

- We call $g_c(\boldsymbol{X})$ the discriminant function
- min Mahalanobis distance?
- min Euclidean distance?

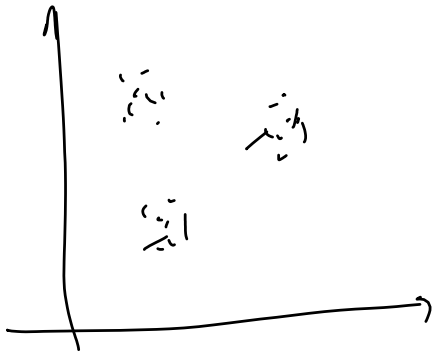
Decision surfaces

```
library(mvtnorm)
par(mfrow=c(1,2))
x<-seq(0,10,length=100)
y<-seq(0,10,length=100)
m1<-c(2,5);S1<-diag(2)
m2<-c(5,6);S2<-diag(2)*2; S2[1,2]<-1; S2[2,1]<-1
f<-Vectorize(function(x,y)dmvnorm(c(x,y),m1,S1)-dmvnorm(c(x,y),m2,S2))
g1<-outer(x,y,f)
contour(x,y,g1, main='Quadratic')

m1<-c(2,5);S1<-diag(2)*2; S1[1,2]<-1; S1[2,1]<-1
m2<-c(5,6);S2<-diag(2)*2; S2[1,2]<-1; S2[2,1]<-1
f<-Vectorize(function(x,y)dmvnorm(c(x,y),m1,S1)-dmvnorm(c(x,y),m2,S2))
g2<-outer(x,y,f)
contour(x,y,g2,main="Linear")
```



K-means

- choose C
- assign C class centres $\boldsymbol{\mu}_c$
- calculate distance, e.g., $D^2_{Eic} = (\boldsymbol{x}_i - \boldsymbol{\mu}_c)^T(\boldsymbol{x}_i - \boldsymbol{\mu}_c)$ for all observations to all class centres, $i = 1, \ldots, N,\ c = 1, \ldots, C$
- assign class c to $\boldsymbol{x}_i$ if distance smallest for class c
- compute new class centres $\boldsymbol{\mu}_c$ (include only obs in class c)
- iterate from third step



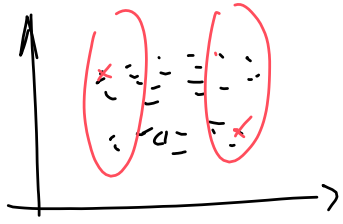
3 classes. K mean steps

→ We compute the center of each class. Randomly

→ We compute the distance from each pixel for each center and recompute the center. Iteratively.

→ At some point it converges

Haverer, this method is sensible to the "shape" of the clusters
We might end in a sitration like this



The two clusters are mistaken, because the starting point led to problems

One solution could be: do the same thing 10 times with different starting point

↳ Choose the solution that converge at the same the most of the times

Initialization of μ_c

- random observations within range of data
- first C 'different enough' observations
- based on PCA, e.g., uniformly distributed along first PC axis, or in plane spanned by two first PC axes
- ...

Fuzzy c-means

1. choose C
2. assign C class centres $\boldsymbol{\mu}_c$
3. calculate distance, e.g., $D^2_{Eic} = (\boldsymbol{x}_i - \boldsymbol{\mu}_c)^T(\boldsymbol{x}_i - \boldsymbol{\mu}_c)$ for all observations to all class centres
4. assign degree of membership u_{ic} to $\boldsymbol{x}_i$ for all classes, e.g., $u_{ic} = (1/D^2_{Eic}) / \sum_{j=1}^{C} 1/D^2_{Eij}$ leading to $\sum_{c=1}^{C} u_{ic} = 1$
5. compute new class centres (include all obs weighted by u_{ic}) $\boldsymbol{\mu}_c = \sum_{i=1}^{N} u_{ic}\boldsymbol{x}_i / \sum_{i=1}^{N} u_{ic}$
6. iterate from third step

Pixel is soft assigned to the class instead of hard assigned

Weaknesses of K-means

- Sensitive to outliers
- Cannot handle clusters have different densities
- Cannot handle clusters have different sizes
- Cannot handle clusters that are non-globular
- A compromise
  - Chose a larger number of clusters to obtain pure natural clusters.

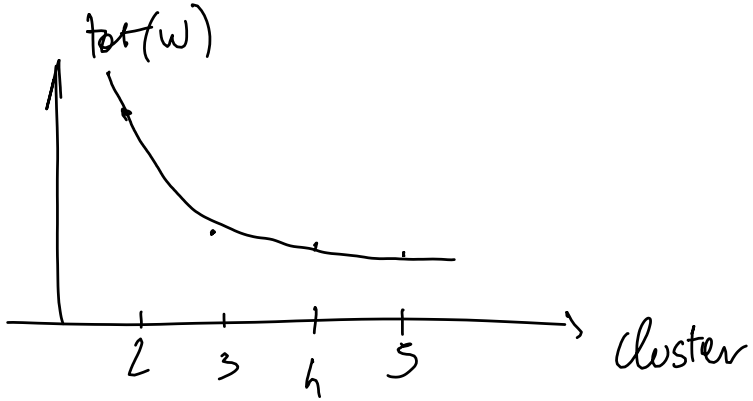
Optimal number of clusters

- A good clustering has high between clusters variation (SS_B) and low within (among) clusters variation (SS_W)
- Maximize the variance ratio criterion VRC

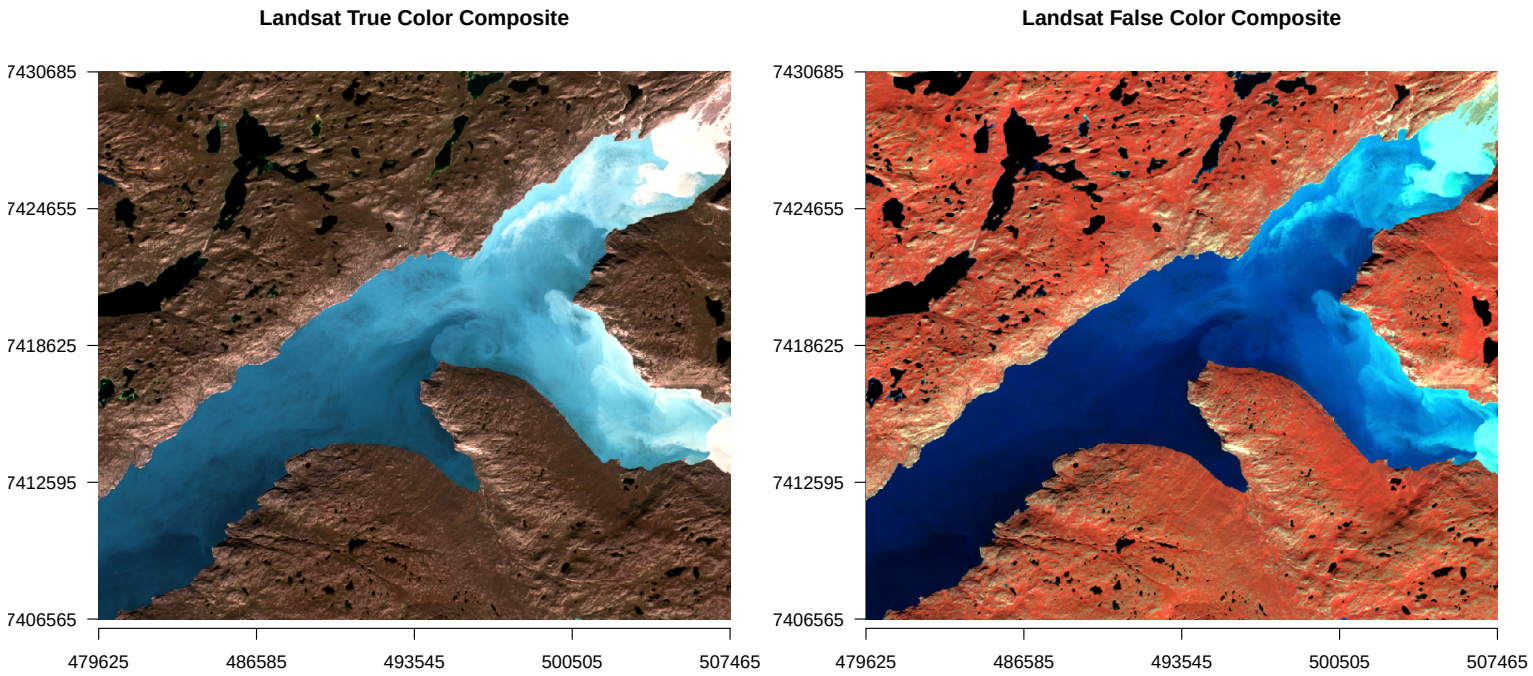
$$VRC(k) = \frac{SS_B(k)/(k-1)}{SS_W(k)/(N-k)}$$

sometimes called the Calinski-Harabasz clustering evaluation criterion

- The elbow method: plot of cluster number vs. total within variation



Example with K-means

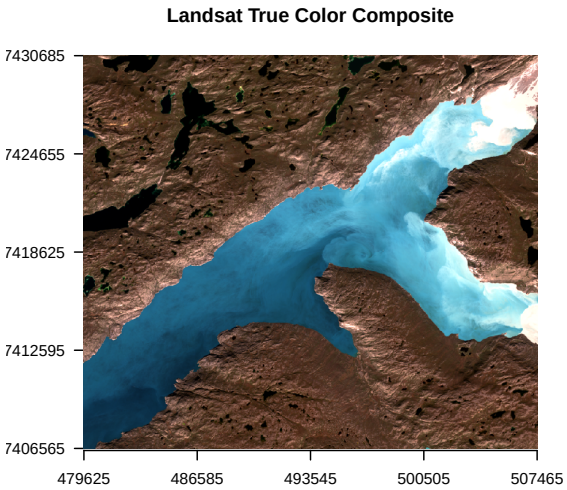


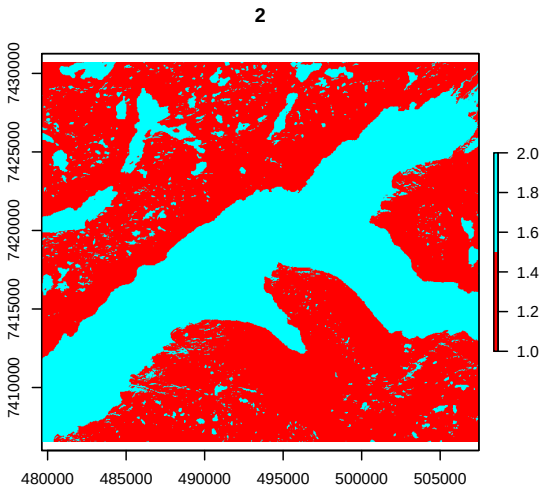
- First we need to decide the number of clusters
- By visual inspection: 2-3 water classes, 1-ice class, and 1-2 rock classes
- We can try with 2-7 clusters and investigate the results

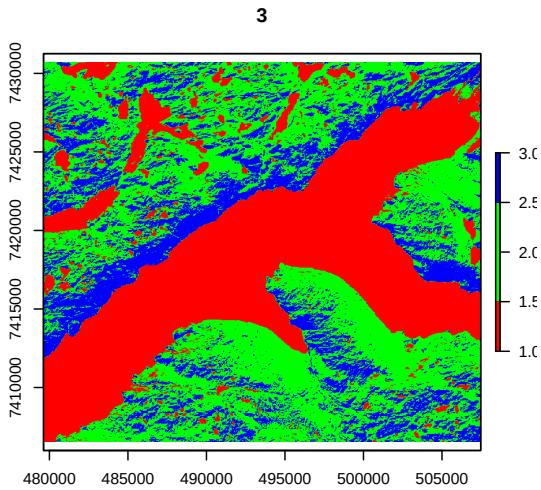
Coding it up in R

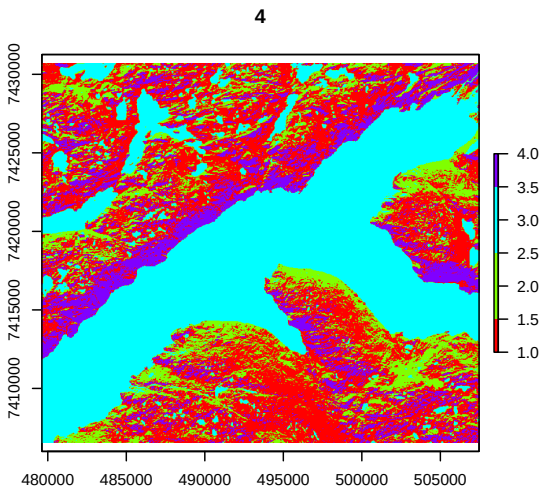
```
# gr[] is a matrix with a band in each column
kmncluster <- kmeans(gr[], centers = 4, iter.max = 200, nstart = 2)

#Available components:
#
#"cluster"      "centers"      "totss"        "withinss"
#"betweenss"    "size"         "iter"         "ifault"
# "tot.withinss"
```

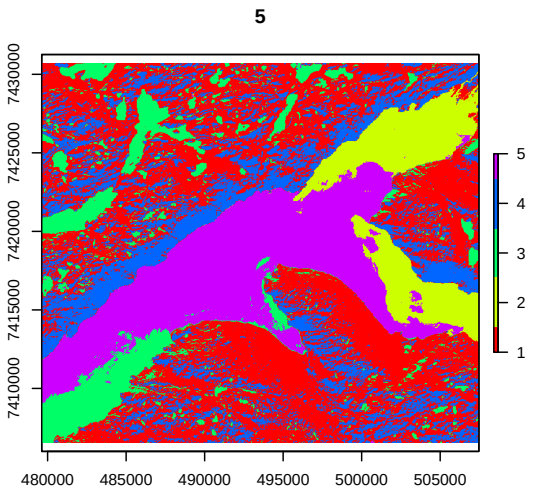
Results of Kmeans based on the original bands

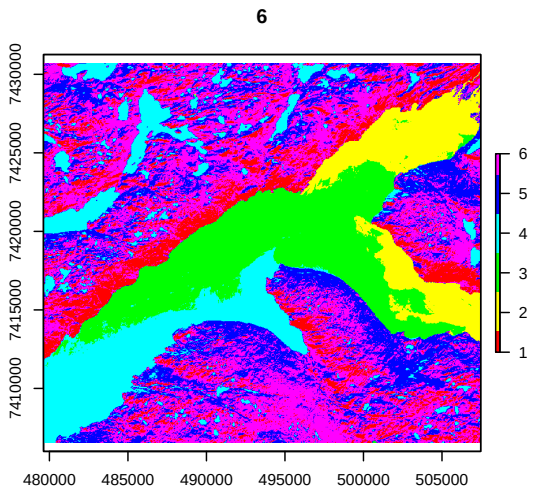






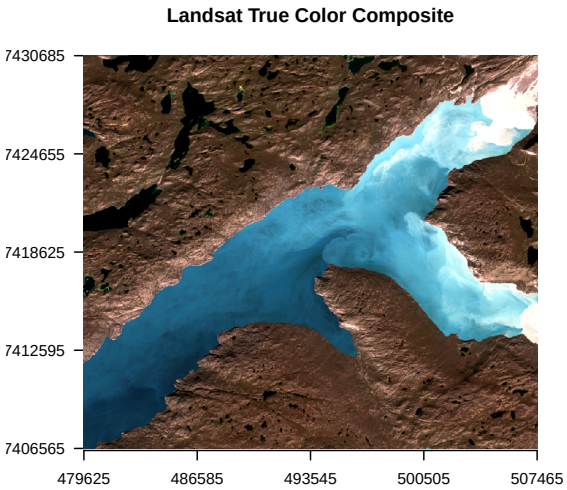


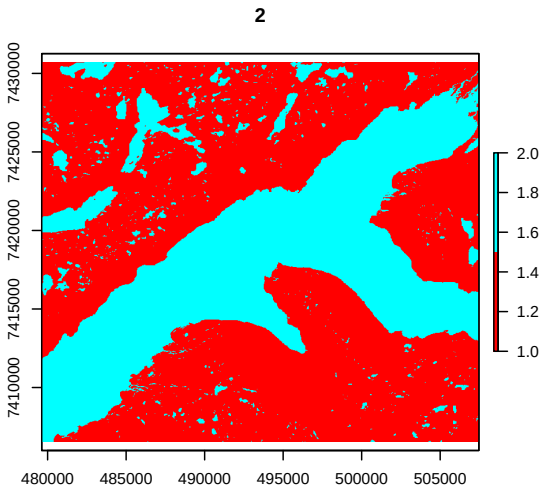


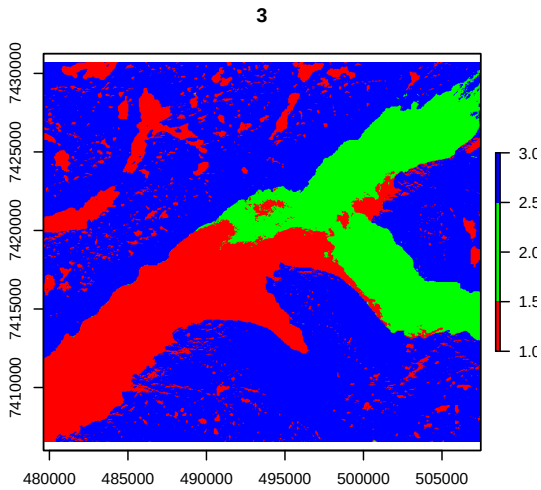


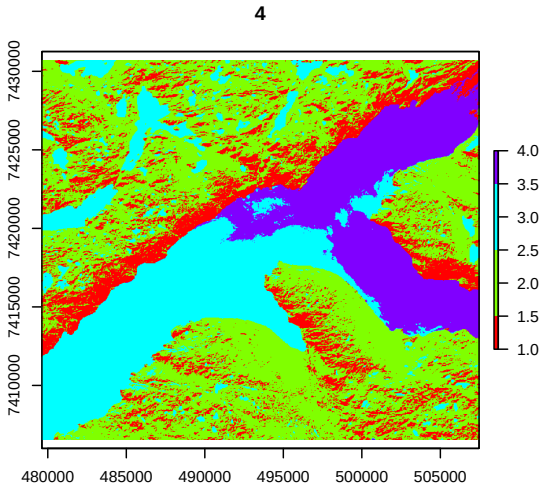
Results of Kmeans based on the original bands (standardized data)

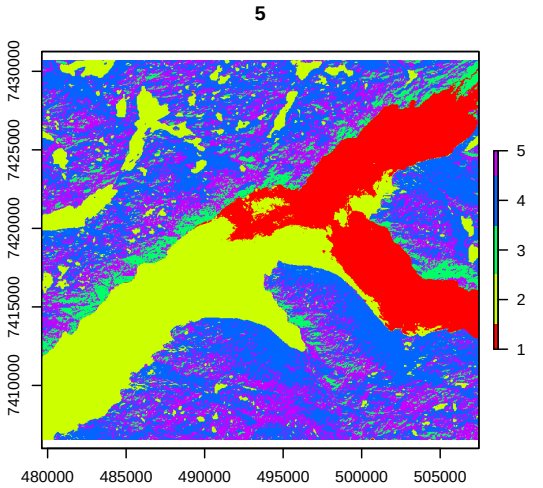
Standardizing the data helps on "globuling" the clusters

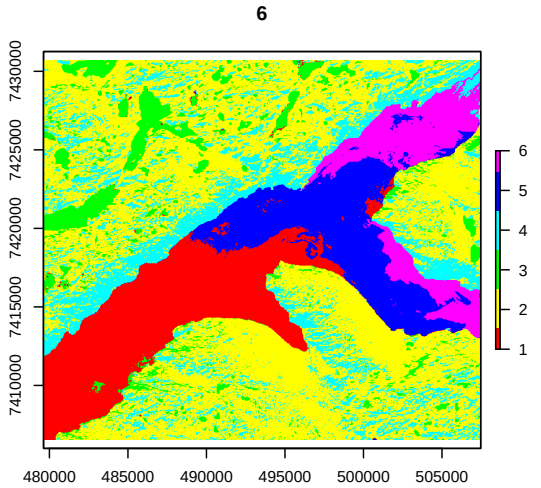




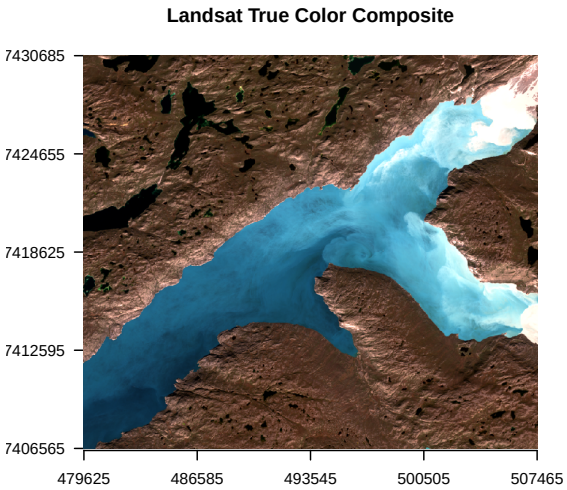


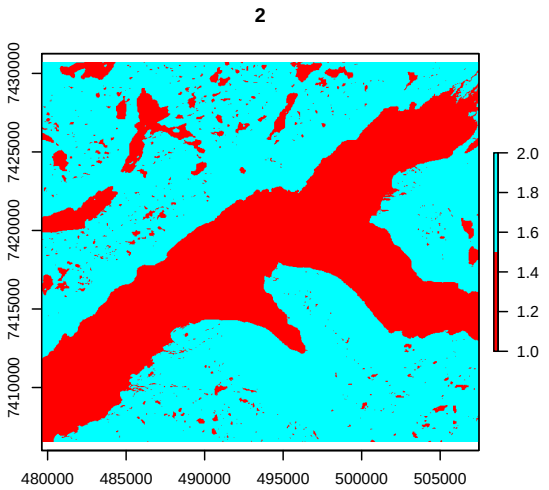


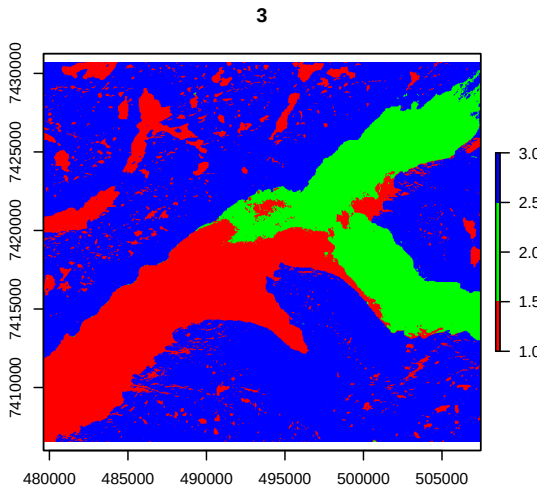


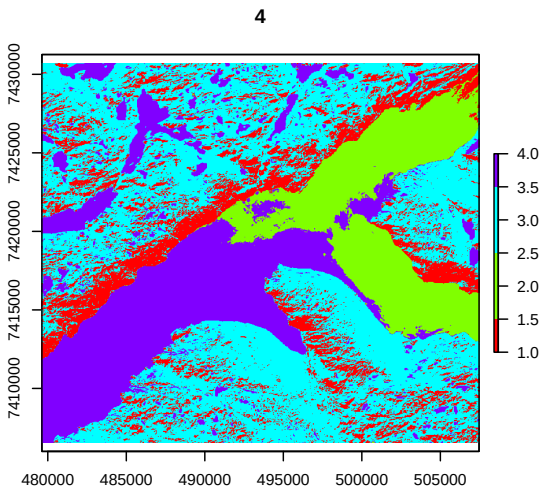


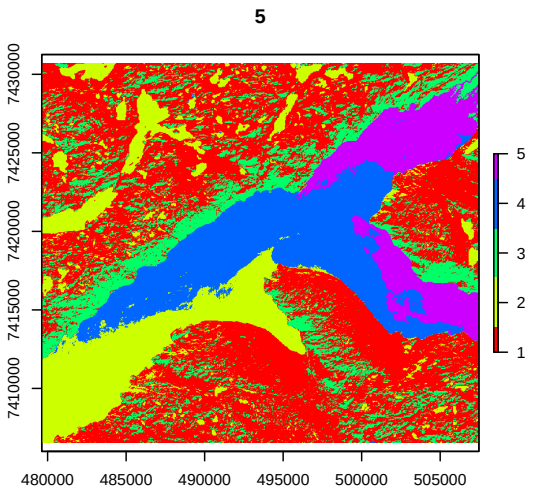
Results of Kmeans based on PC1-PC3

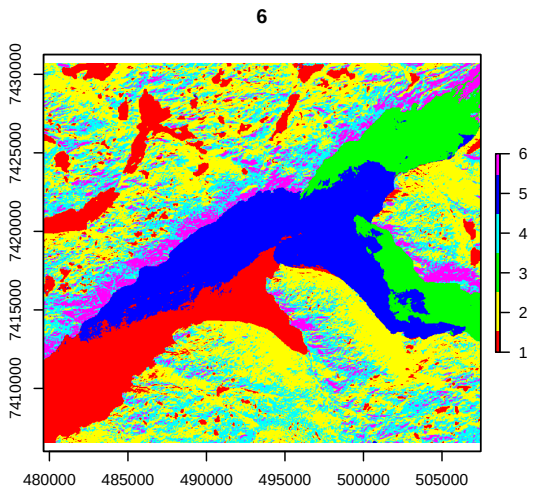






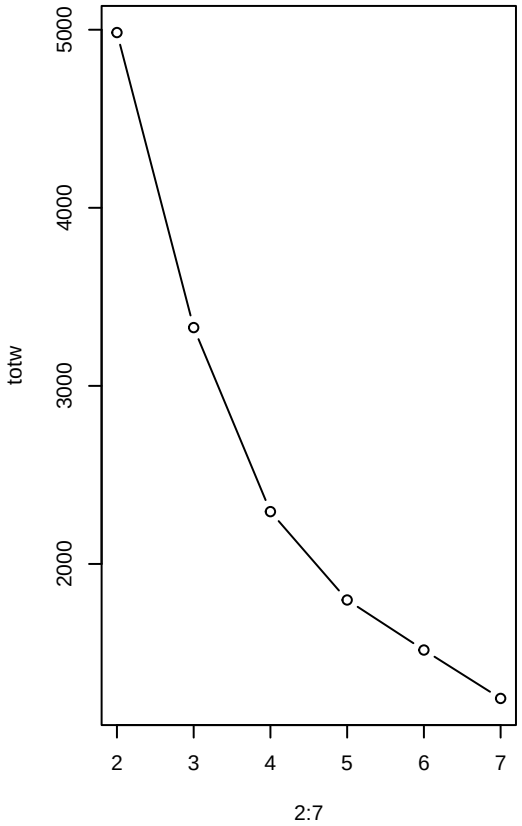


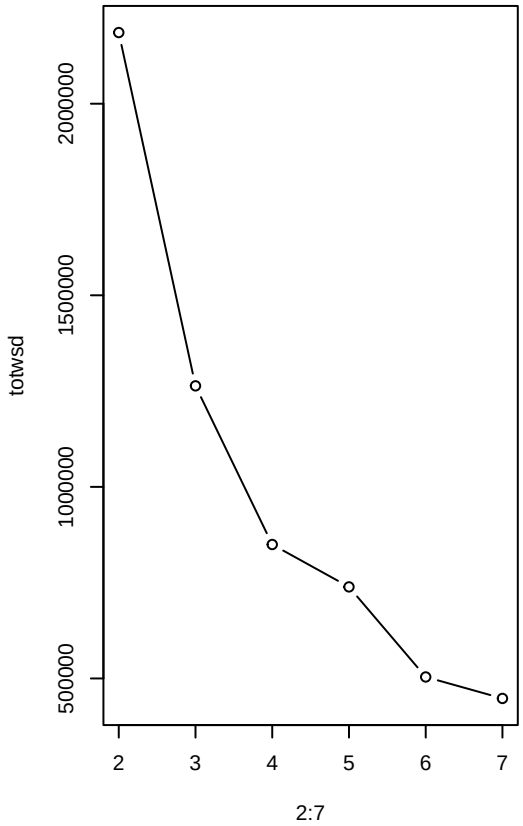


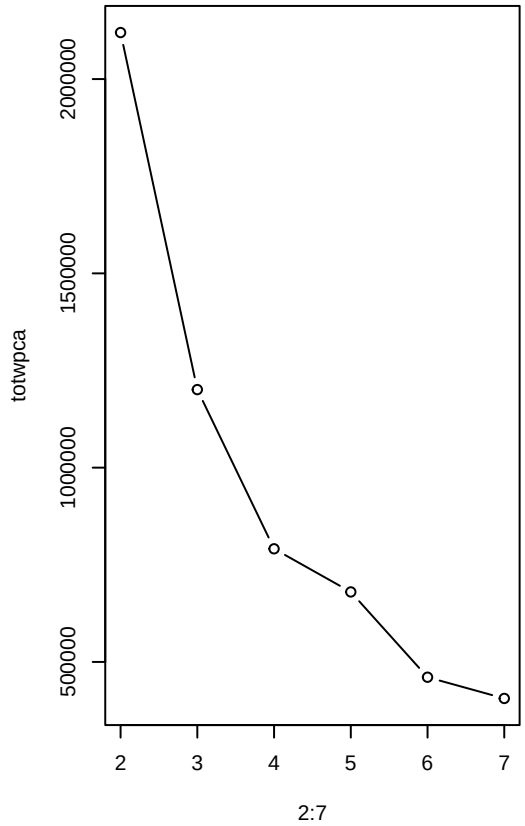


Cluster evaluation

- Elbow plot of total within cluster variation







Gaussian mixture models 1D

- Assume that we have some observation x_i, $i = 1 \ldots n$ where a subset y_j, $j = 1 \ldots p$ and z_k, $k = 1 \ldots m$ come from two distinct normal distributions
  - then we can easily estimate the parameters μ and σ (eg. from training data)

$$\mu_y = \frac{y_1 + y_2 + \ldots + y_p}{p} \text{and} \quad \sigma_y^2 = \frac{(y_1 - \mu_y)^2 + \ldots + (x_p - \mu_y)^2}{p}$$

- and if we know the parameters of the two normal distributions μ_y, σ_y, μ_z, and σ_z
  - then we can easily assign the observations by calculating the probability (via Bayes' rule) of that they belong to these distributions (maximum likelihood classification)
    $P(\omega_c|\boldsymbol{x}_i) = K\ P(\boldsymbol{x}_i|\omega_c)P(\omega_c)$ with $1/K = \sum_{j=1}^{C} P(\boldsymbol{x}_i|\omega_j)P(\omega_j)$

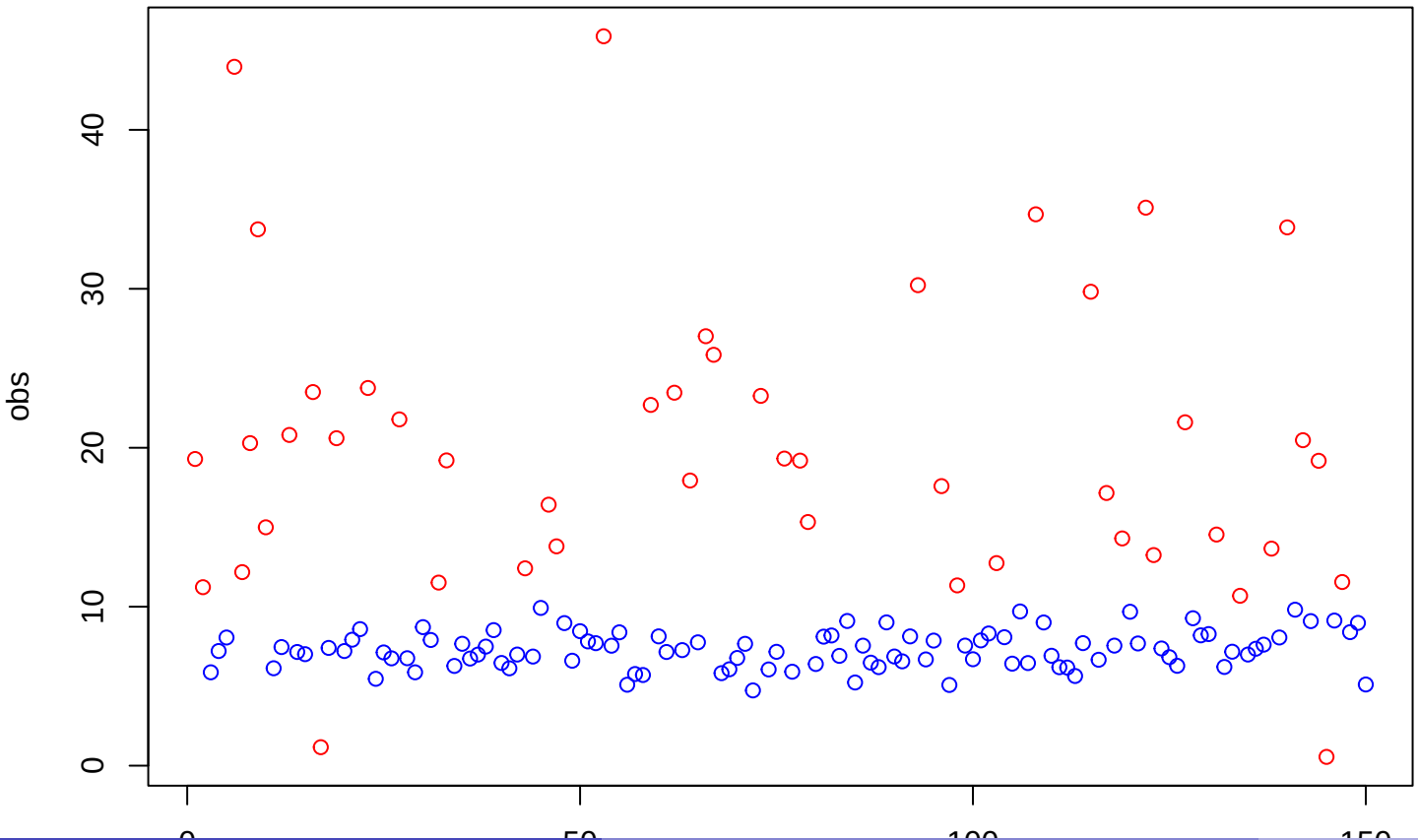
- What if we just have normal distributed observations that comes from two distributions, how can we group these?
  - This is the situation in a mixture model, where the class membership is an unobserved parameter (latent variable)

Expectation Maximization algorithm (EM)

- Start with two random placed Gaussian (μ_a, σ_a^2) and (μ_b, σ_b^2)
- calculate the posterior probability/class membership via Bayes' rule $p(b|x_i)$ (soft assignment)
- re-estimate (μ_a, σ_a^2) and (μ_b, σ_b^2)
- iterate until convergence

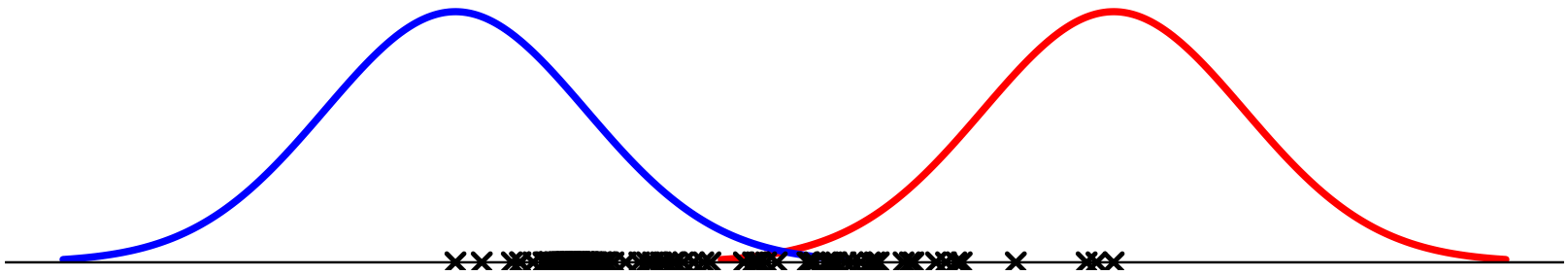
EM of 1D R example of GMM

```
x<-rnorm(100,7,1)
y<-rnorm(50,20,10)
obs<-sample(c(x,y))
```



EM of 1D R example of GMM

- Initialization: select μ and σ for the two gaussians



EM of 1D R example of GMM – First iteration

- M-step

  $u_{i,blue} = P(\omega_{blue}|\boldsymbol{x}_i) = K\ P(\boldsymbol{x}_i|\omega_{blue})P(\omega_{blue})$ with $1/K = \sum_{j=1}^{C} P(\boldsymbol{x}_i|\omega_j)P(\omega_j)$

  $u_{i,red} = 1 - u_{i,blue}$

- E-step

  $\mu_{blue} = \frac{x_1 * u_{1,blue} + \ldots + x_n * u_{n,blue}}{u_{1,blue} + \ldots + u_{n,blue}} = \frac{x_1 * u_{1,blue} + \ldots + x_n * u_{n,blue}}{p(\omega_{blue})n}$

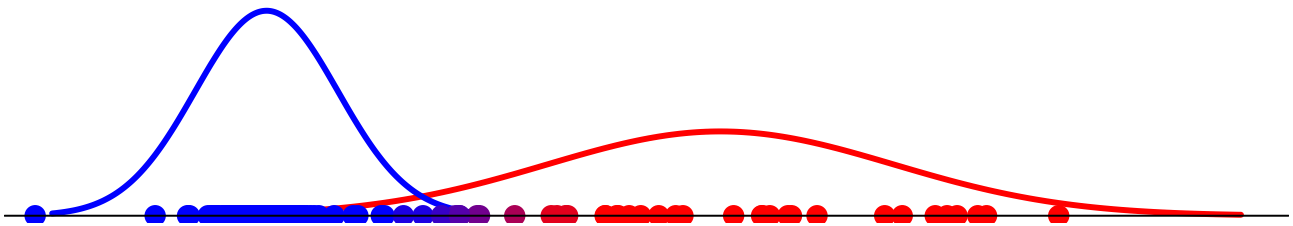
  $\sigma^2_{blue} = \frac{u_{1,blue}(x_1 - \mu_{blue})^2 + \ldots + u_{n,blue}(x_n - \mu_{blue})^2}{u_{1,blue} + \ldots + u_{n,blue}} = \frac{u_{1,blue}(x_1 - \mu_{blue})^2 + \ldots + u_{n,blue}(x_n - \mu_{blue})^2}{p(\omega_{blue})n}$

  $\mu_{red} = \frac{x_1 * u_{1,red} + \ldots + x_n * u_{n,red}}{u_{1,red} + \ldots + u_{n,red}}$

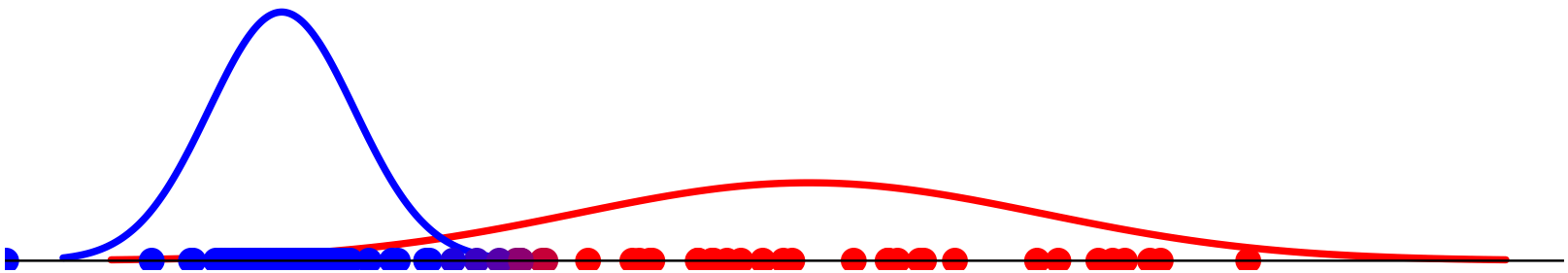
  $\sigma^2_{red} = \frac{u_{1,red}(x_1 - \mu_{red})^2 + \ldots + u_{n,red}(x_n - \mu_{red})^2}{u_{1,red} + \ldots + u_{n,red}}$

  $P(\omega_{blue}) = (u_{1,blue} + \ldots + u_{n,blue})/n$ represent the proportion of the blue class

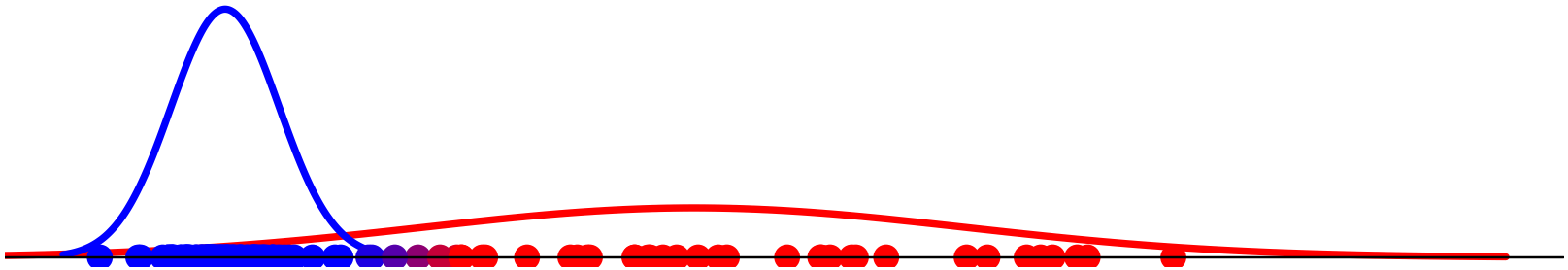
  $P(\omega_{red}) = 1 - P(\omega_{blue})$



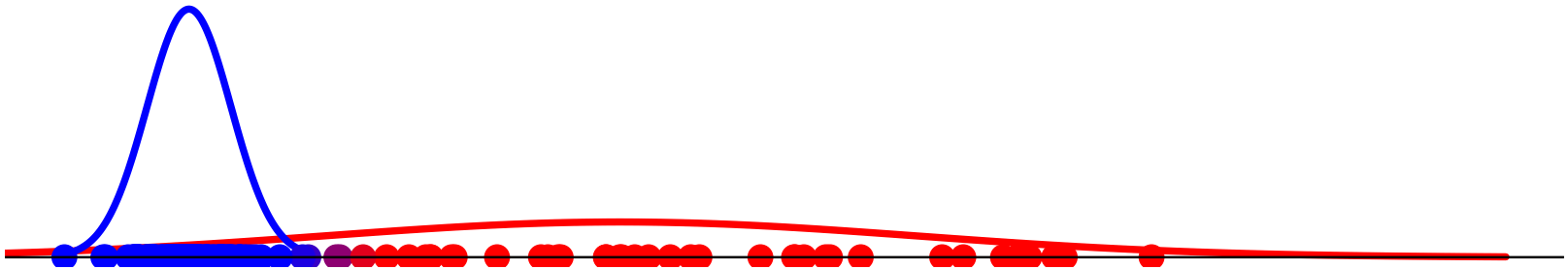
EM of 1D R example of GMM – 2. iteration



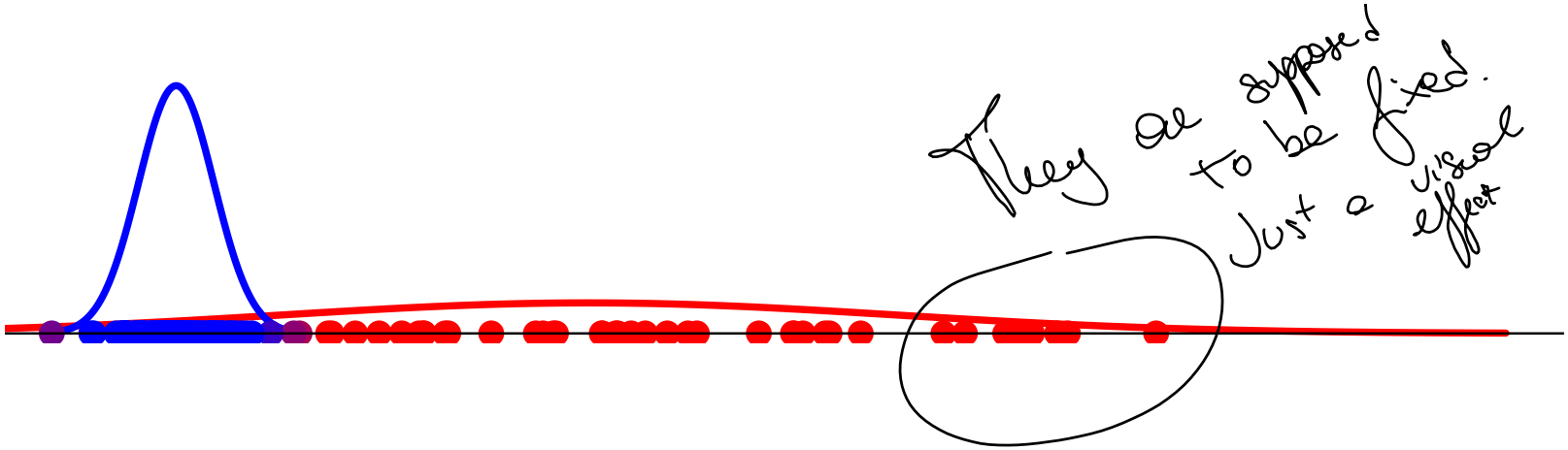
EM of 1D R example of GMM – 3. iteration



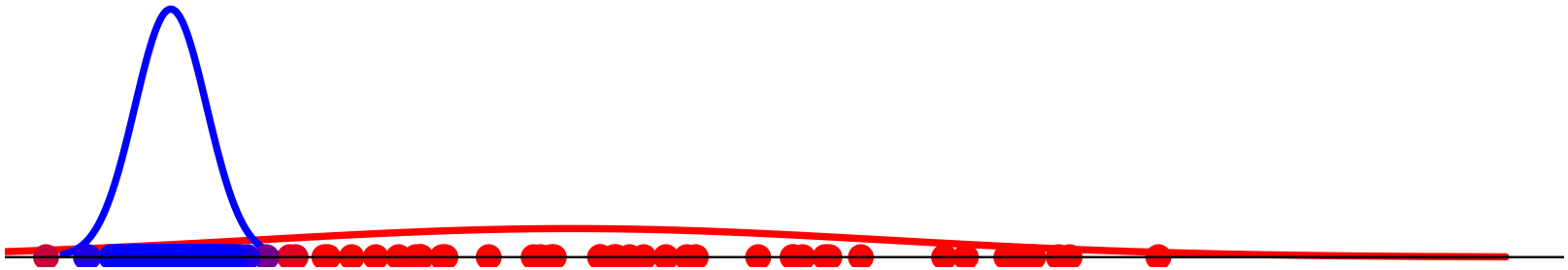
EM of 1D R example of GMM – 4. iteration



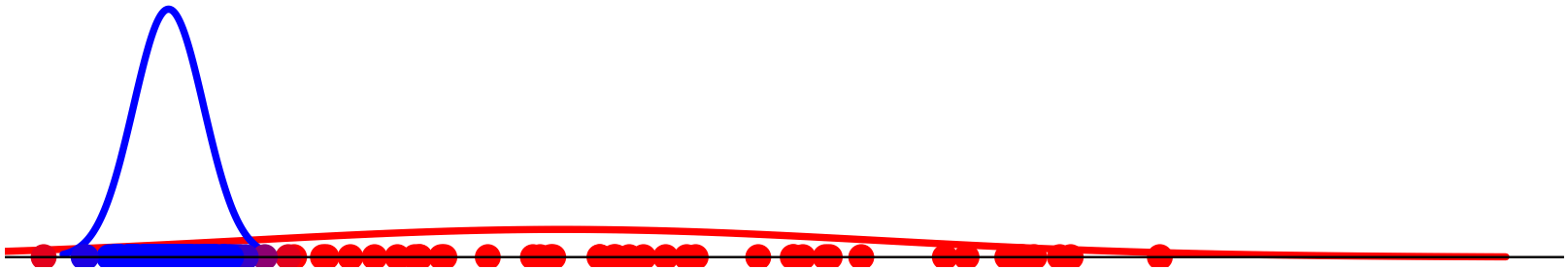
EM of 1D R example of GMM –5. iteration



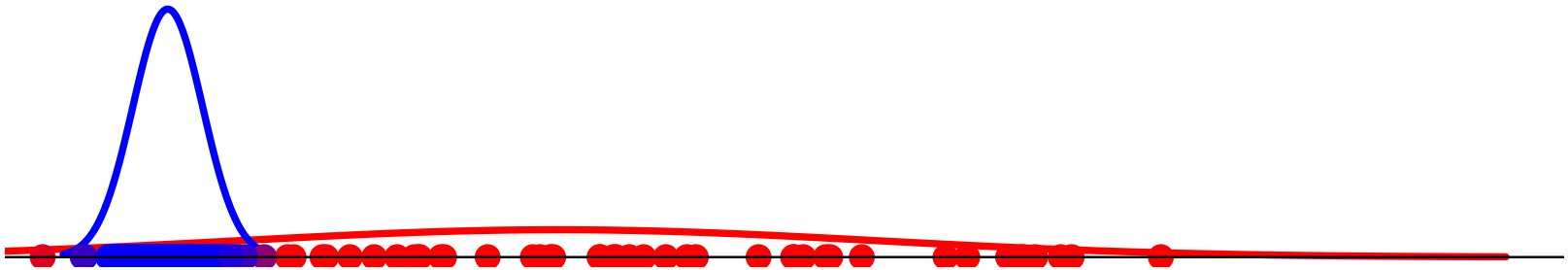
EM of 1D R example of GMM – 6. iteration



EM of 1D R example of GMM – 7. iteration



EM of 1D R example of GMM – 8. iteration



1D R example of GMM

```
x<-rnorm(100,7,1)
y<-rnorm(50,20,10)
obs<-sample(c(x,y))

gmm<-function(x, mu=range(obs), sigma=rep(sd(obs),2), w=c(.5,.5), nstep=10){
  N<-length(x)
  for(i in 1:nstep){
    # data likelohhod of the two classes
    px<-cbind(dnorm(x,mu[1],sigma[1]), dnorm(x,mu[2],sigma[2]))
    # class membership
    u<-cbind(px[,1]*w[1],px[,2]*w[2])
    u<-u/rowSums(u) # nomalization
    w<-colMeans(u) # propertion of class
    mu<-c(mean(x*u[,1])/w[1],mean(x*u[,2])/w[2])
    sigma<-sqrt(c(mean(u[,1]*(x-mu[1])^2)/w[1],mean(u[,2]*(x-mu[2])^2)/w[2]))
  }
  return(list(mu=mu, sigma=sigma, w=w, u=u))
}

C<-gmm(obs, nstep=100)
```

Gaussian mixture models, GMM

- Bayes' rule: $P(\omega_c|\boldsymbol{x}_i) = K\ P(\boldsymbol{x}_i|\omega_c)P(\omega_c)$ with $1/K = \sum_{j=1}^{C} P(\boldsymbol{x}_i|\omega_j)P(\omega_j)$
- **GMM**: Given some $u_{ic} = P(\omega_c|\boldsymbol{x}_i)$ with $\sum_{c=1}^{C} u_{ic} = 1$, calculate

- $P(\omega_c) = \frac{1}{N}\sum_{i=1}^{N} u_{ic}$ (interpreted as proportion of class c)
  $\boldsymbol{\mu}_c = \frac{1}{NP(\omega_c)}\sum_{i=1}^{N} u_{ic}\boldsymbol{x}_i$
  $\boldsymbol{\Sigma}_c = \frac{1}{NP(\omega_c)}\sum_{i=1}^{N} u_{ic}(\boldsymbol{x}_i - \boldsymbol{\mu}_c)(\boldsymbol{x}_i - \boldsymbol{\mu}_c)^T$
- $\boldsymbol{\mu}_c$ and $\boldsymbol{\Sigma}_c$ define $P(\boldsymbol{x}_i|\omega_c)$ which with $P(\omega_c)$ via Bayes' rule give a new $u_{ic} = P(\omega_c|\boldsymbol{x}_i)$ which in turn gives a new $P(\omega_c)$: iterate
- example on Expectation Maximization (EM) algorithm
  E-step: calculate $P(\omega_c) = \cdots$, $\boldsymbol{\mu}_c = \cdots$, $\boldsymbol{\Sigma}_c = \cdots$
  M-step: calculate $P(\omega_c|\boldsymbol{x}_i)$ in Bayes' rule

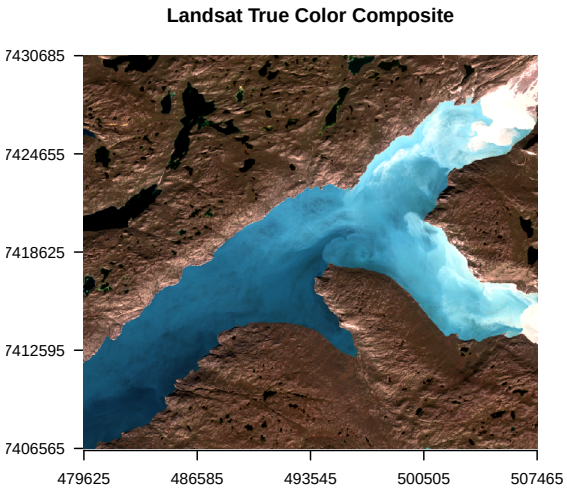
Initialization of μ_c and Σ_c

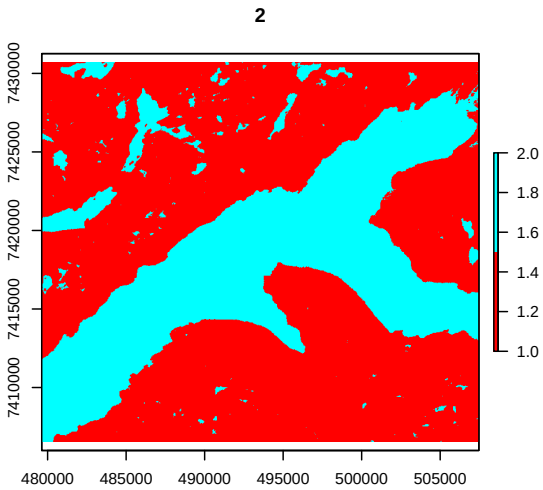
1. select observations at random as initial means
   mixing proportions are uniform
   initial covariance matrices are diagonal, elements on the diagonal are the variances
2. start with result from k-means or fuzzy c-means
3. ...

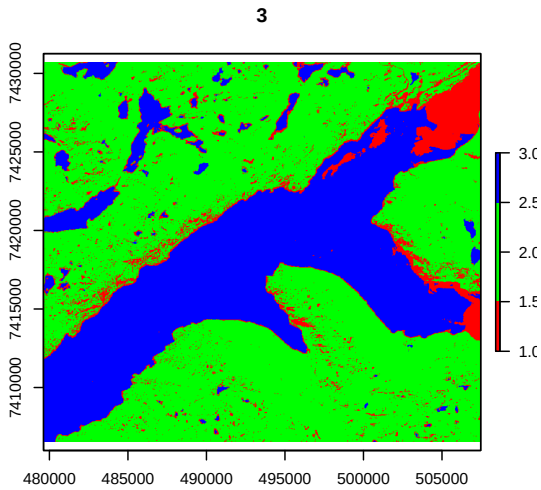
Example with Gaussian mixture models (the Greenland example)

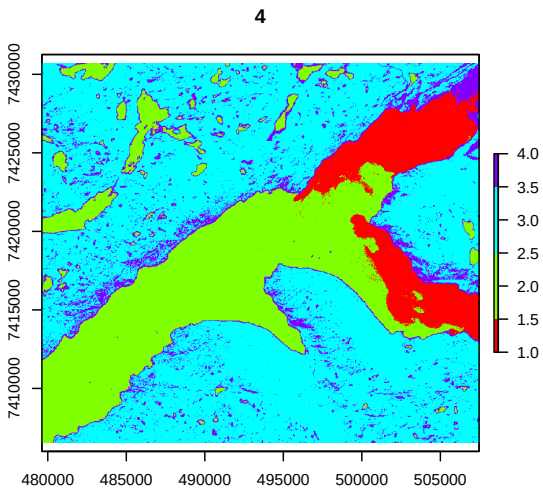
```
gr <- stack("greenland2.tif")
gr.df <- as.data.frame(gr)
gmm2<-Mclust(gr.df[,1:6],G=2) # G is the number of clusters

#Available components:
# [1] "call"           "data"           "modelName"      "n"
# [5] "d"              "G"              "BIC"            "loglik"
# [9] "df"             "bic"            "icl"            "hypvol"
# [13] "parameters"     "z"              "classification" "uncertainty"
```

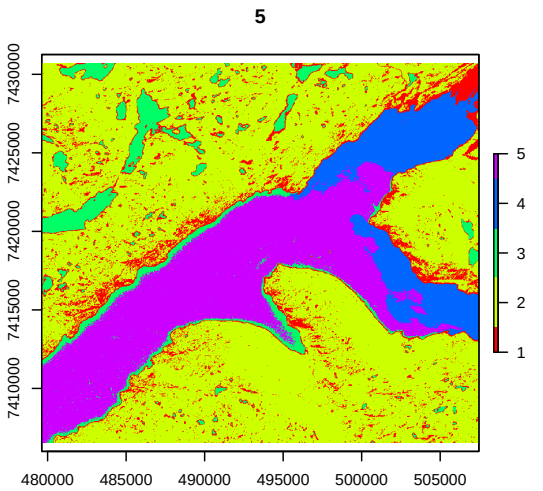
Results

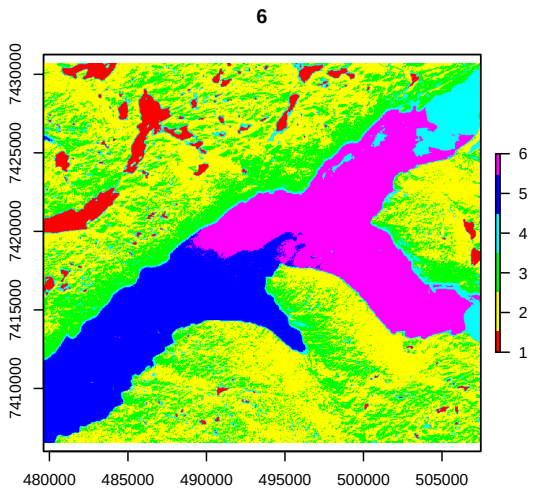

Landsat True Color Composite
7430685
7424655
7418625
7412595
7406565
479625
486585
493545
500505
507465


2
7430000
7425000
7420000
7415000
7410000
480000
485000
490000
495000
500000
505000
2.0
1.8
1.6
1.4
1.2
1.0


3
7430000
7425000
7420000
7415000
7410000
480000
485000
490000
495000
500000
505000


4
7430000
7425000
7420000
7415000
7410000
480000
485000
490000
495000
500000
505000
4.0
3.5
3.0
2.5
2.0
1.5
1.0


5
7430000
7425000
7420000
7415000
7410000
480000
485000
490000
495000
500000
505000
5
4
3
2
1


6
7430000
7425000
7420000
7415000
7410000
480000
485000
490000
495000
500000
505000
6
5
4
3
2
1

Hybrid classification unsupervised and supervised combined

1. Perform clustering on a subset of the image to identify spectral classes
   - A way to obtain training data, when it cannot be obtained the usual way
2. Use reference data to relate the spectral classes to information classes
3. Use supervised classification e.g. maximum likelihood classification
4. Classify each pixel and evaluate using independent test data

Software

- Matlab
  - Statistics and Machine Learning Toolbox
  - Cluster Analysis
  - k-means: `kmeans`, `evalcluster`
  - GMM: `fitgmdist`, `cluster`, `posterior`
- Python

```
from sklearn.mixture import GaussianMixture
from sklearn.cluster import KMeans
from sklearn.metrics import calinski_harabasz_score as CHS
```

- R
  - `kmeans` from R base
  - `Mclust` from the package `mclust`

Exercise

- Apply the image 'yukon.tif' including 6 bands ('blue','green','red','NIR','SWIR1','SWIR2')
- Experiment with
  - k-means
  - Gaussian Mixture Models, GMM.
- Try:
  - different numbers of clusters,
  - different initializations (option '`Replicates`' in matlab),
  - use original 6 bands (or subset of these)
  - first few principal components
  - evaluate the clusters using e.g. the elbow method and/or the Calinski-Harabasz score
- Additionally
  - Compare the clustering methods with the maximum likelihood classifier from module 3 for the test data. You might need to merge some of the clusters
- Write a small report or a readable journal (3-4 pages) including figures and Matlab code.