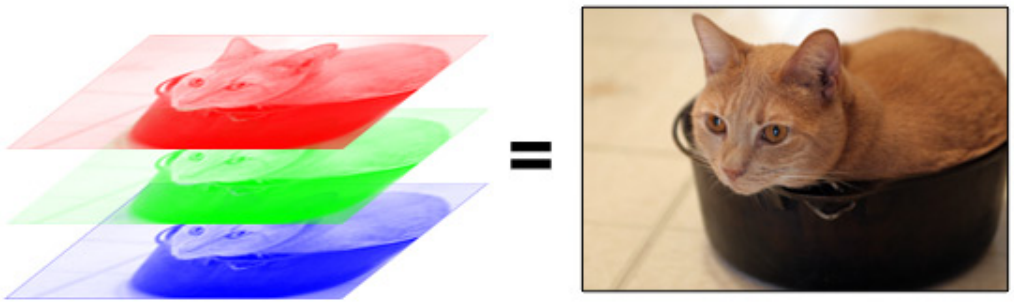
ImageCompression

February 3, 2021

1 Image compression by k-means clustering

1.0.1 3-channel (RGB) color images in 24-bit color representation allows for more than 16 million (2^{24}) different colors. In the computer, an RGB image is stored as a stack of (3) matrices, hence each pixel of the image is a 3-dimensional vector reflecting the mixture of the colors Red, Green and Blue:



1.0.2 By doing k-means clustering of the 3-dimensional (RGB) pixel vectors, we can find just a few (k) useful color combinations (the resulting cluster centers) to replace the original pixel vectors in a compressed version of the image.

1.0.3 See also the VMLS k-means notes on clustering applications.

2 Candidate color imagees for doing pixel-clustering

[1]:
```
# import Pkg; Pkg.add("Images"); Pkg.add("ImageMagick")
# import Pkg; Pkg.add("HTTP")
using Images # Image loading, saving, manipulation
using HTTP   # Internet access
```

[2]:
```
using Plots # Precompiles on every startup (~20 secondss)
gr() # Needs modules Plots and GR to be installed, may need a rebuild of GR␣
 ↪with ']build GR'
default(size=(600, 450), fmt = :png) # Default plot size, change output format␣
 ↪to png
```

2.1 Load an image either locally or from the internet

[3]:
```
# Load image stored on a local folder adress:
# myimage = "C:/Users/ulfin/Dropbox/MATH310/Forelesninger_2021/Clustering/
 ↪Everyones_a_little_bit_racist_sometimes.jpg";
# Ximg = load(myimage);

# Load image from the Internet:
# ---------------------------------
#imageadress = "https://cdn.images.express.co.uk/img/dynamic/151/590x/secondary/
 ↪spacex-launch-why-starman-tesla-roadster-david-bowie-falcon-heavy-1225205.
 ↪jpg";
#imageadress = "http://pressarchive.theoldglobe.org/_img/pressphotos/
 ↪pre2008%20photos/aveQ5.jpg";
#imageadress = "https://vgc.no/drfront/images/2018/02/12/c=1114,366,1920,1048;
 ↪w=262;h=143;384858.jpg";
#imageadress = "https://www.dagbladet.no/images/73342156.jpg?
 ↪imageId=73342156&x=15.602322206096&y=10.807860262009&cropw=72.
 ↪060957910015&croph=61.764705882353&width=912&height=521&compression=80";
imageadress = "http://lynski.no/wp-content/uploads/2016/09/
 ↪Anders_H%C3%B8st_OSM_Gautefall_seier.jpg"

# ---------------------------------
#imageadress = "https://upload.wikimedia.org/wikipedia/en/7/7d/
 ↪Lenna_%28test_image%29.png";
#imageadress = "http://www.johnloomis.org/ece563/notes/basics/components/
 ↪mandrill/Mandrill.jpg";
# ---------------------------------
                                  # download("http://pressarchive.theoldglobe.
 ↪org/_img/pressphotos/pre2008%20photos/aveQ5.jpg", "myimage.png"); Ximg =␣
 ↪load("myimage.png");
myimage = download(imageadress); # Needs package ImageMagick
Ximg = load(myimage);

original = plot(Ximg, title = string("The original image to be compressed by␣
 ↪k-means color clustering"), size = (1000,420))
display(original)
```

[4]:
```
# Find image size and prepare to reshape the pixel-data for clustering
n,m = size(Ximg);
nm = n*m
```

[4]: 497025

2.1.1 We make a three-column matrix X where each row is an RGB-vector of a pixel position in the original image

[12]:
```
mat = channelview(Ximg); # Convert from image format to 3 x n x m (0-1).
X = float( reshape( permutedims(mat, (2,3,1)), (nm, 3) ) ); # Channels last,␣
 ↪vectorize image dims, convert to float
```

2.1.2 Now we can cluster the RGB-vectors of the pixels in X

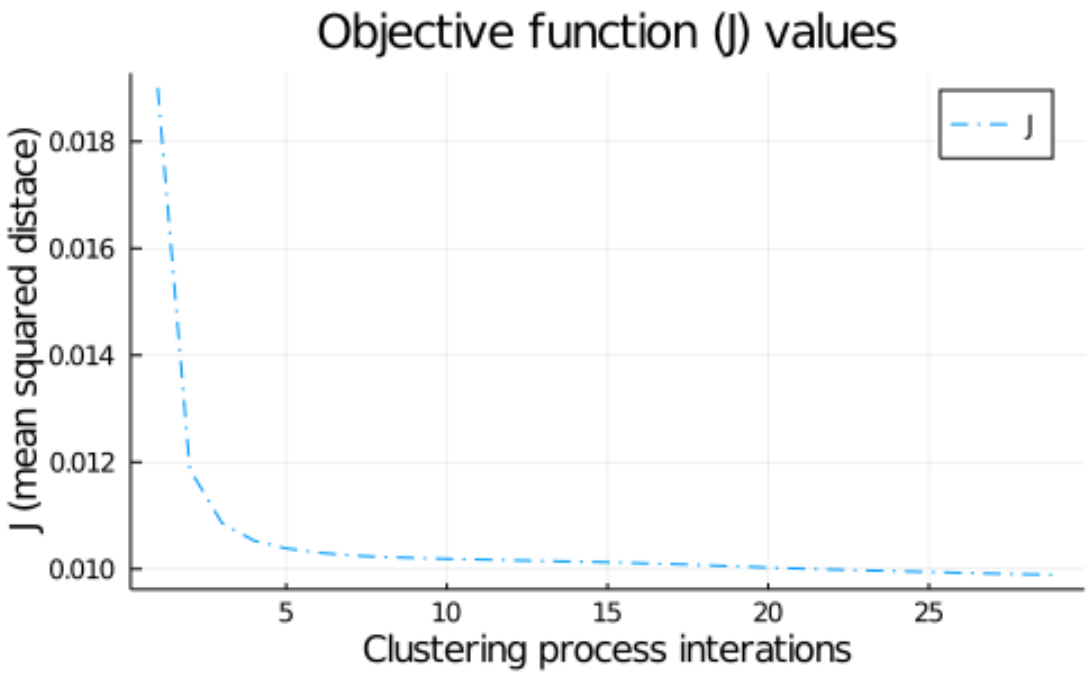
[13]:
```
include("mykmeans.jl")
## Cluster RGB-pixel values (in X) into k color clusters by the k-means␣
 ↪algorithm
k = 16; # The number of clusters

@time begin
Cid, Ccenters, J, cs = mykmeans(X,k); # This will take some time...
                                      # Cid:      a vector of cluster labels for␣
 ↪the rows in X.
                                      # Ccenters: the resulting k cluster centers.
 ↪
                                      # J:        the clustering objective␣
 ↪function values
                                      # cs:       the cluster sizes (number of␣
 ↪members in each cluster)
```

```
#Cs = uint8(Ccenters); # Convert cluster centers into uint8-format
end
```

```
  7.703864 seconds (588.67 k allocations: 4.807 GiB, 8.79% gc time)
```

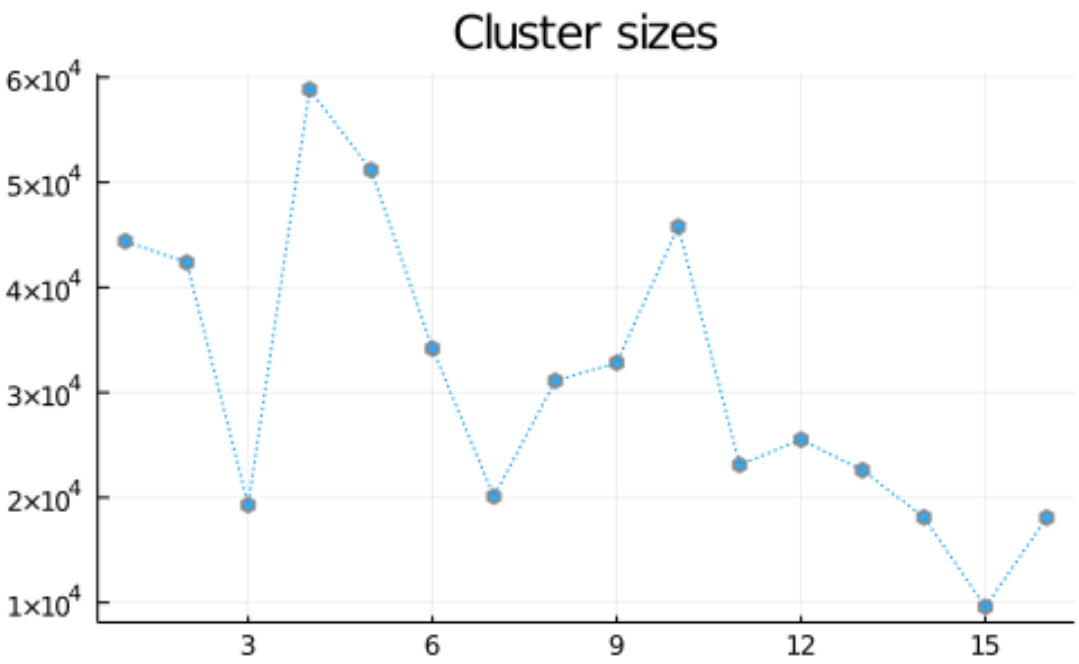
[13]:
```
([16; 16; … ; 4; 4], Float32[0.4045972 0.3435841 0.32962012; 0.29181877
0.24158487 0.22087784; … ; 0.33571938 0.4275605 0.109497234; 0.6044407 0.5926064
0.6641352], Any[0.019008199030849958, 0.011868108709077407,
0.010860277039459403, 0.010529063215637012, 0.010386475345327977,
0.010305557434096224, 0.01025533710710975, 0.01022521472291614,
0.010205729021143868, 0.010189910339176631  …  0.010027353521130307,
0.010007958393216729, 0.009991486094020048, 0.009975415860585209,
0.009959413093148971, 0.009944156954977072, 0.00992869443267328,
0.009913173777409632, 0.009902186340214156, 0.009892212254359484], [44394.0;
42369.0; … ; 9591.0; 18082.0])
```

[14]:
```
# Plotting the objective function values reflecting the clustering process
Jplot = plot(J, linestyle = :dashdot, title = "Objective function (J) values",
    ylabel = "J (mean squared distace)", xlabel = "Clustering process␣
 ↪interations", label = "J", size = (500, 300))
display(Jplot)
```



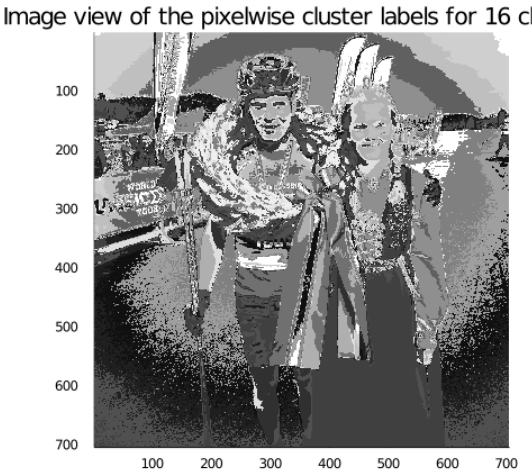
[15]:
```
# Plotting the cluster sizes:
csplot = plot(cs, line = (:dot, 1), marker = ([:hex :d], 3, 0.8, Plots.
 ↪stroke(3, :gray)), title = "Cluster sizes", label = "", size = (500, 300))
```

```
display(csplot)
```



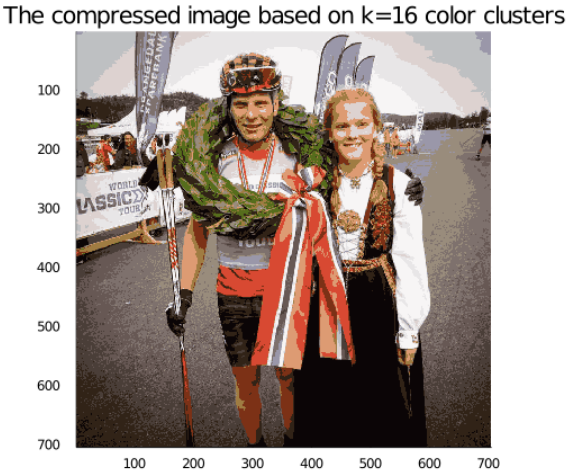
2.1.3 Reshape and display cluster labels as associated image

[16]:
```
cl = reshape(Cid,(n,m)); # cl is an image (n x m - matrix) viewing the cluster␣
 ↪labels
#print(string("Image view of the cluster labels for ", k, " clusters"))
#Gray.(cl/k)
labelplot = plot(Gray.(cl/k), title = string("Image view of the pixelwise␣
 ↪cluster labels for ", k, " clusters"), size = (1000,420))
display(labelplot)
```



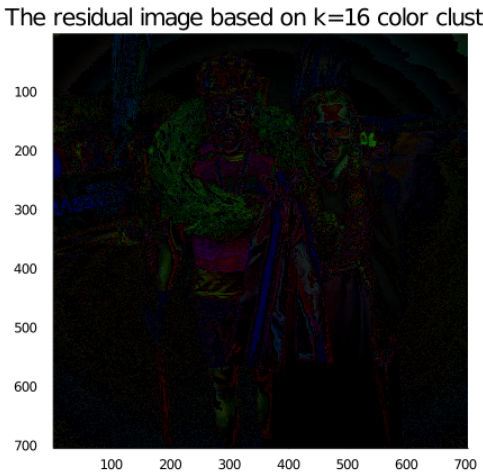
2.1.4 Show the compressed image by inserting the (k) cluster center vectors to display the colors

[17]:
```
# print(string("The compressed image based on k=", k, " color clusters"))
# Use cluster-IDs (Cid) as lookup in cluster centers (Ccenters), reshape,␣
 ↪permute and convert to RGB
# colorview(RGB, permutedims( reshape(Ccenters[Cid,:],(n, m, 3)), (3,1,2)))
cmpplot = plot(colorview(RGB, permutedims( reshape(Ccenters[Cid,:],(n, m, 3)),␣
 ↪(3,1,2))), title = string("The compressed image based on k=", k, " color␣
 ↪clusters"), size = (1000,420))
display(cmpplot)
```



2.1.5 The residual image (difference between original and compressed images based on the clustering)

[18]:
```
#print(string("The residual image based on k=", k, " color clusters"))
#Ximg - colorview(RGB, permutedims( reshape(Ccenters[Cid,:],(n, m, 3)),␣
 ↪(3,1,2)))
resplot = plot(Ximg-colorview(RGB, permutedims( reshape(Ccenters[Cid,:],(n, m,␣
 ↪3)), (3,1,2))), title = string("The residual image based on k=", k, " color␣
 ↪clusters"), size = (1000,420))
display(resplot)
```


The residual image based on k=16 color clusters
100
200
300
400
500
600
700
100
200
300
400
500
600
700

[ ]: