EECS 551 Homework 8 YUZHAN JIANG

P1:

$$x_{k+1} = x_k + \alpha_k d_k$$

where $\alpha_k = \underset{\alpha}{\arg\min}\ f(x_k + \alpha d_k)$

$\Downarrow$

let gradient $g_k = \nabla f(x_k + \alpha d_k) = 0$

$$= A'(A(x_k + \alpha d_k) - y) = 0$$

$$A'(Ax_k + \alpha A d_k - y) = 0$$

$$A'Ax_k + \alpha A'Ad_k - A'y = 0$$

$$\alpha A'A \cdot d_k = A'y - A'Ax_k$$

$$\alpha = \frac{A'y - A'Ax_k}{A'Ad_k}$$ where $d_k = -Pg_k$

P2.

$$\hat{X} = \underset{X}{\operatorname{argmin}} \ \frac{1}{2} \| Y - X \|_F^2 + \beta \|X\|_*$$

$$\hat{X} = \sum_{k=1}^{\hat{r}} [\sigma_k - \beta]_+ \ u_k \cdot v_k'$$

where $h_{soft}(\sigma;\beta) \triangleq [\sigma - \beta]_+ = \begin{cases} \sigma - \beta, & \sigma > \beta \\ 0, & \text{otherwise} \end{cases}$

The error : $\| \hat{X} - Y \|_F = \| \sum_{k=1}^{\hat{r}} [\sigma_k - \beta]_+ \ u_k \cdot v_k' - \sum_{k=1}^{\hat{r}} \sigma_k u_k v_k' \|_F$

$$= \| \sum_{k=1}^{r} \left([\sigma_k - \beta]_+ - \sigma_k \right) u_k \cdot v_k' \|_F$$

$$= \sqrt{\sum_{k=1}^{r} \left([\sigma_k - \beta]_+ - \sigma_k \right)^2}$$

P3:

(a) This is the picture I choose

In [71]:

```
# Load and display an image
#image_color = load("mit.jpg") # Replace with your image -- put in the same directory as this notebook
image_color = load("my_pic.jpg")
```

Out[71]:

Out[72]:

After compressing the image:

In [12]:
```
typeof(Int(ceil(0.2 * 34554)))
size(image_data)
```
Out[12]: (3648, 5472)

In [14]:
```
Ac, k = compress_image(image_data, 0.2)
```
Out[14]: (Float32[0.70057136 0.7187647 … 0.7468418 0.74715155; 0.71857595 0.73117566 … 0.74363816 0.74558026; … ; 0.0029345006 0.061809853 … 0.11325527 0.11744386; 0.035306334 0.055174425 … 0.10467967 0.11643361], 437)

In [80]:
```
heatmap(Ac, color = :grays, yflip = :true, ticks = [])
```
Out[80]:

I believe that the quality of the compressed image is "good enough". It is obvious and clear to see the shape of the architecture and the words " Hill AUDITORIUM "

(b)

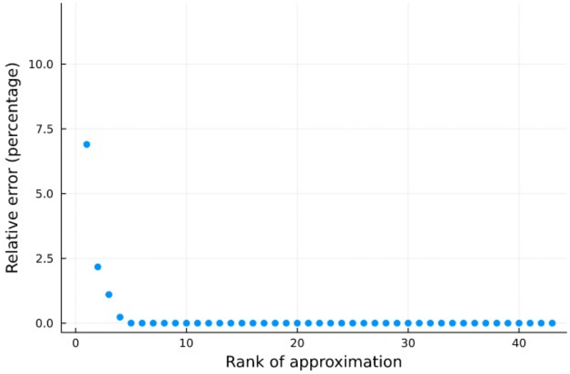
The picture of basic "nameless" logo

In [21]:

```
# Compute relative errors and plot them as a function of k
relative_error_fun = k -> svd_rank_k_approx_error(σ, k)
relative_error = relative_error_fun.(0:r)

# Plot errors
scatter(0:r, 100*relative_error, label = "",
    ylabel = "Relative error (percentage)",
    xlabel = "Rank of approximation",
    ylim = [0,12], # error at 0 is 100%
)
```

Out[21]:



Here, I choose k=5, the error is approaching 0 that gives decent image quality

This is compressing code with lower rank k

In [32]:

```
"""
Ac= compress_image_with_rank_k(A, k)
In:
* `A` `m × n` matrix
* `k`
Out:
* `Ac` a `m × n` matrix containing a compressed version of `A`
that can be represented using at most `(100 * p)%` as many bits
required to represent `A`
"""
function compress_image_with_rank_k(A, k)
    (U, s, V) = svd(A)
    Ac = U[:,1:k] * Diagonal(s[1:k]) * V[:,1:k]'

    return Ac
end
```

Out[32]: compress_image_with_rank_k

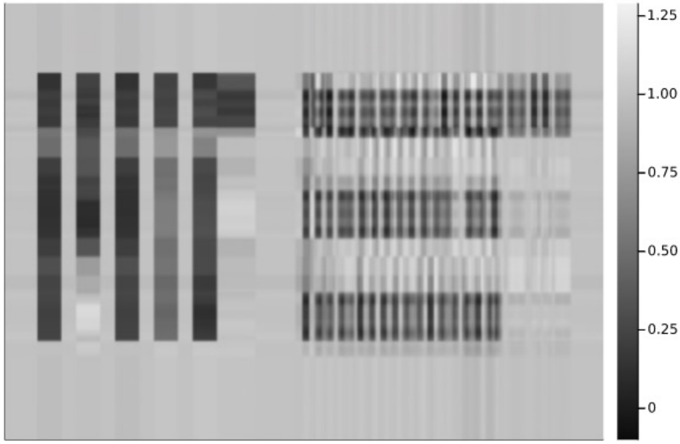
After compressing, the picture is clear and good.

In [34]:

```
heatmap(Ac, color = :grays, yflip = :true, ticks = [])
```

Out[34]:

In [52]:
```
heatmap(Ac, color = :grays, yflip = :true, ticks = [])
```
Out[52]:



However, I try the same rank for compressing the logo with extra lettering, and it is pretty bad and lose lots of information. It is hard to clearity the letters in this picture.

Overall, the logo without extra letters is well compressible with lower rank $k=5$

P4:

(b) Test code and the norm output

In [293]:

```julia
using Random: seed!
seed!(0)
Y = rand(5,3)
display(Y)

Yh = optshrink1(Y, 2)
display(Yh)



#Yh = optshrink1(Y, 2)
```

```
5×3 Matrix{Float64}:
 0.823648  0.203477   0.585812
 0.910357  0.0423017  0.539289
 0.164566  0.0682693  0.260036
 0.177329  0.361828   0.910047
 0.27888   0.973216   0.167036

5×3 Matrix{Float64}:
 0.598335  0.236597   0.56591
 0.609996  0.154529   0.556546
 0.176241  0.0837109  0.169988
 0.448318  0.332532   0.46055
 0.237647  0.519169   0.324806
```

In [298]:

```julia
# include("optshrink1.jl") # uncomment if you need this
using Random: seed!
using LinearAlgebra: norm
seed!(0)
X = randn(30) * randn(20)' # outer product
Y = X + 40 * randn(size(X))
Xh_opt = optshrink1(Y, 1)
# Xh_lr = # you finish this to make the conventional rank-1 approximation
(U, s, V) =svd(Y)
Xh_lr = s[1]* U[:,1]  * V[:, 1]'

@show norm(Xh_opt - X)
@show norm(Xh_lr - X)
```

```
norm(Xh_opt - X) = 131.74000698480566
norm(Xh_lr - X) = 403.0278595936427
```

Here, Optshrink praide a better estimate of X than classical low-rank approximation.

P5:

(b)

In [324]:

```julia
using LinearAlgebra: norm
using Random: seed!
seed!(0)
X = randn(10^5) * randn(100)' / 8 # test large case now
Y = X + randn(size(X))
Xh_opt = optshrink2(Y, 1)
# Xh_lr = # you finish this part
(U, s, V) =svd(Y)
Xh_lr = s[1]* U[:,1]  * V[:, 1]'

@show norm(Xh_opt - X)
@show norm(Xh_lr - X)
```

```
norm(Xh_opt - X) = 240.35441054181598
norm(Xh_lr - X) = 317.08494230558097
```

Out[324]: 317.08494230558097

Optshrink again provide a better estimate of X than classical low-rank approximation

P6.

(b)

In [37]:

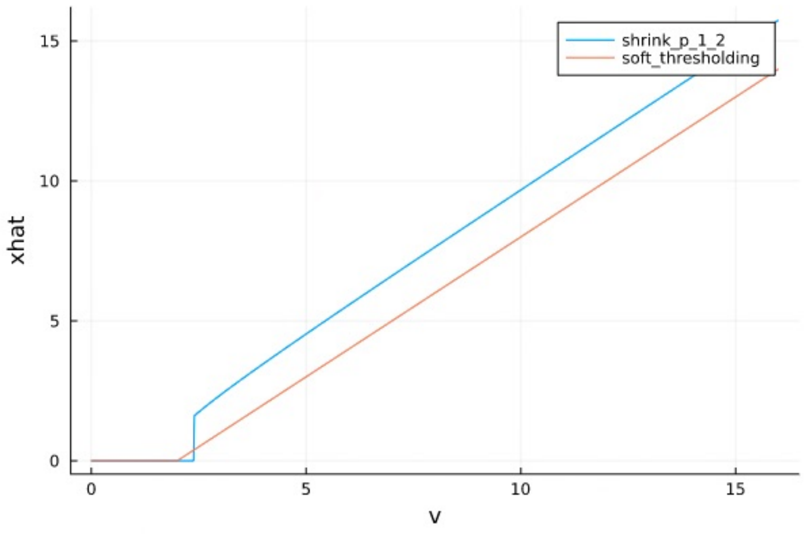
```julia
function soft_thresholding(v, reg)
    xh = zeros(size(v))
    fun1 = (v) -> v - reg
    fun2 = (v) -> v + reg

    big = v .> reg
    small = v .< -reg

    xh[big] = fun1.(v[big])
    xh[small] = fun2.(v[small])
    return xh
end
```

Out[37]: soft_thresholding (generic function with 1 method)

Out[38]:



Shrinkage function will have better solution x_hat. It provide better estimate of X.

P7.

$$D = \begin{bmatrix} 0 & a & 2a \\ a & 0 & a \\ 2a & a & 0 \end{bmatrix} \qquad J=3 \qquad 3\times3 \text{ matrix}$$

① First, we compute S from D by squaring its elements

$$S = \begin{bmatrix} 0 & a^2 & 4a^2 \\ a^2 & 0 & a^2 \\ 4a^2 & a^2 & 0 \end{bmatrix}$$

② Then,

The orthogonal complements of $R(1_J)$ for $J=3$

$$P^{\perp} \triangleq I_3 - \frac{1}{3} 1_3 1_3'$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - \frac{1}{3} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - \frac{1}{3} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$$= \frac{1}{3} \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}$$

$$\therefore \quad P^{\perp} S P^{\perp} = \frac{1}{3} \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \begin{bmatrix} 0 & a^2 & 4a^2 \\ a^2 & 0 & a^2 \\ 4a^2 & a^2 & 0 \end{bmatrix} P^{\perp}$$

$$= \frac{1}{3} \begin{bmatrix} -5a^2 & a^2 & 7a^2 \\ -2a^2 & -2a^2 & -2a^2 \\ 7a^2 & a^2 & -5a^2 \end{bmatrix} \frac{1}{3} \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}$$

$$= \frac{1}{9} \begin{bmatrix} -18a^2 & 0 & 18a^2 \\ 0 & 0 & 0 \\ 18a^2 & 0 & -18a^2 \end{bmatrix}$$

$$= \begin{bmatrix} -2a^2 & 0 & 2a^2 \\ 0 & 0 & 0 \\ 2a^2 & 0 & -2a^2 \end{bmatrix}$$

③ Third:

$$\hat{G} \triangleq -\frac{1}{2} P^{\perp} S P^{\perp}$$

$$\triangleq \begin{bmatrix} a^2 & 0 & -a^2 \\ 0 & 0 & 0 \\ -a^2 & 0 & a^2 \end{bmatrix} \approx C'C$$

Here, we consider $d=2$

$\therefore \quad C \in F^{d \times 3} \; (F^{2\times 3}) \qquad C = \begin{bmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{bmatrix}$

$$\therefore \quad \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \end{bmatrix} \begin{bmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{bmatrix} = \begin{bmatrix} a^2 & 0 & -a^2 \\ 0 & 0 & 0 \\ -a^2 & 0 & a^2 \end{bmatrix}$$

Using Compact SVD:

$$G = V \Sigma V' = \sum_{k=1}^{3} \sigma_k v_k v_k' = \sum_{k=1}^{2} \sigma_k v_k v_k' \qquad \Sigma_d \triangleq \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix}$$

$$G = \begin{bmatrix} a^2 & 0 & -a^2 \\ 0 & 0 & 0 \\ -a^2 & 0 & a^2 \end{bmatrix} = \begin{bmatrix} a \\ 0 \\ -a \end{bmatrix} \begin{bmatrix} a & 0 & -a \end{bmatrix}$$

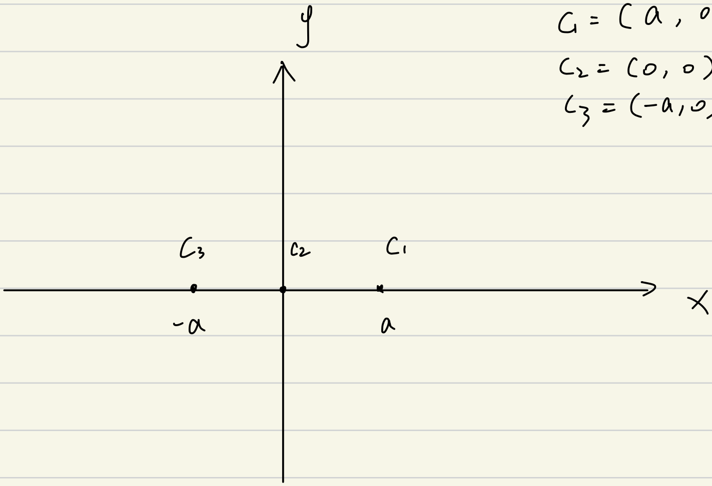
$$= \frac{\begin{bmatrix} a \\ 0 \\ -a \end{bmatrix}}{\sqrt{2a^2}} \underbrace{\sqrt{2a^2} \cdot \sqrt{2a^2}}_{= \sigma_1} \frac{\begin{bmatrix} a & 0 & -a \end{bmatrix}}{\sqrt{2a^2}}$$

$$\therefore \quad \Sigma_d = \begin{bmatrix} 2a^2 & 0 \\ 0 & 0 \end{bmatrix} \qquad V_d \triangleq \begin{bmatrix} a & a \\ 0 & 0 \\ -a & a \end{bmatrix} \Big/ \sqrt{2a^2}$$

$$\therefore \quad \hat{C} = \left(\Sigma_d\right)^{\frac{1}{2}} \cdot V_d' = \begin{bmatrix} \sqrt{2a^2} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} a & 0 & -a \\ a & 0 & a \end{bmatrix} \Big/ \sqrt{2a^2}$$

$$= \begin{bmatrix} a & 0 & -a \\ 0 & 0 & 0 \end{bmatrix}$$

Plot:



$c_1 = (a, 0)$

$c_2 = (0, 0)$

$c_3 = (-a, 0)$