Am vrut la inceput sa folosesc cd pentru a ajunge cu usurinta la imagini insa am citit ca nu este o practica eficienta din punct de vedere al debuggingului (voiam sa fac asta nestiind ca pot citi pe calea absoluta imaginea sub forma de matrice): <https://www.mathworks.com/matlabcentral/answers/342202-read-all-files-from-a-folder-edit-them-and-save>

La task-urile 2 si 5 m-am verificat construind graficele prezentate in cerinta (adica cele specifice primei imagini).

Initial am incercat sa lucrez nevectorizat si faceam calculul erorii pentru taskul 2 grafic 3:

for a=1:m

for b=1:n

eroare(i)=eroare(i)+((A(a,b)-A\_k(a,b))^2)/(m\*n);

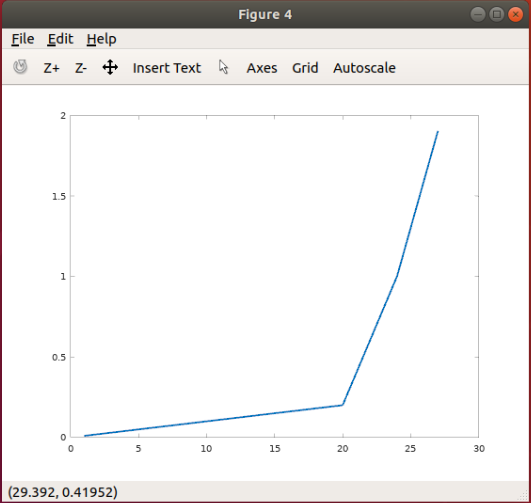
endfor

endfor

Dar outputul nu corespundea cu cel de referinta.

Initial la taskul 2 declaram (ca in ipoteza):

k=[1:19,20:20:99, 100:30:min(m,n)] si graficul sarea la valorile care nu mai mergeau din 1 in 1 (nu se interpolau valorile, ci sareau din 20 in 20 sau din 30 in 30) si un for i in k

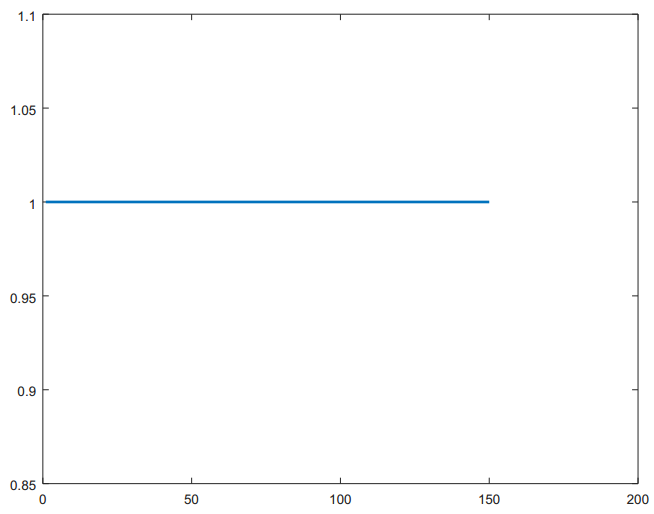


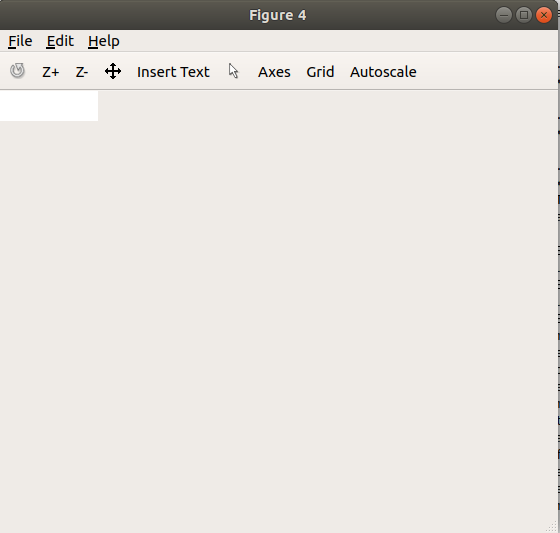
(graficul 4 taskul 2 pentru prima imagine cu abordarea mentionata mai sus)

Dupa care m-am decis sa merg cu un for i de la 1 la min(m,n) (m si n fiind dimensiunile matricii corespunzatoare fiecarei imagini) si k(i)=i

k=[1:min(m,n)] dim\_k=size(k, 2) si for de la 1 la dim\_k

Am avut o problema la taskul 5 este ca valoriile proprii ale lui Z sunt aceleasi indiferent daca avem k=1,2,3... , ele fiind shiftate cu o linie mai jos, dar cand facem diag si sum obtinem mereu acelasi lucru care impartit la el insusi scoate un grafic constant 1 asa ca a trebuit sa schimb implementarea pentru graficul 2 de la taskul 5. (sa citesc cu atentie cerinta si sa imi limitez matricea la dimensiunea ceruta dupa care sa aplic operatiile necesare).



Al patrulea grafic nu apare la primele rulari (depinde de versiunea de octave) dar exista acolo, se poate remedia daca i se da “save” sau “save” as la prima deschidere a octavului :

(Eroarea apare atat la taskul 2 cat si la taskul 5)

La task-ul 6 in eigenface\_core am vrut initial sa nu fac cd (tot pe threaduri existau discutii despre cum se pot numara elementele dintr-un folder facand cd pe calea absoluta) in baza de date si sa creez calea pana la fiecare imagine de acolo insa nestiind in ce sistem de operare se face verificarea, m-am complicat initial cu construirea pathului catre imagine (daca e linux as face strcat database\_path, "/", i.jpg (unde i itereaza de la 1 la cate imagini avem), iar daca e Windows trebuie folosit un "\" intre cele 2). In acest sens am descoperit fullfile, dorind sa nu apara inadvertente in mutarea metodei in Windows insa la cateva imagini adauga un spatiu random "3 .jpg" si atunci am luat modelul folosit in checker pt verificare (m-a inspirat si threadul acesta <https://www.mathworks.com/matlabcentral/answers/65564-find-total-number-of-images-in-a-folder>) (in checkerul din task6.m este utilizata o metoda similara).

Felul in care utilizam fullfile:

%obtine path-ul catre fiecare imagine in parte

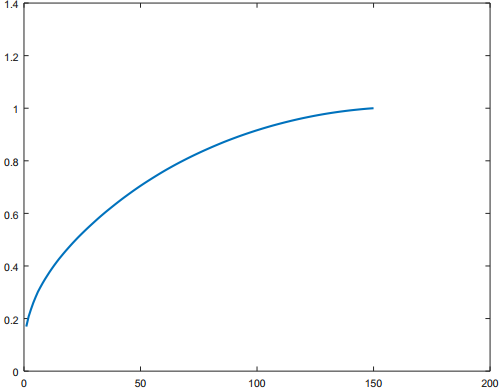
%f = fullfile(database\_path, image)

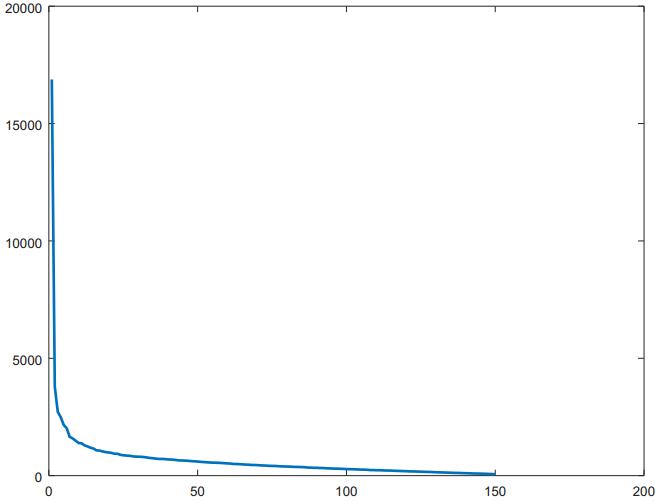
Am mai avut probleme si pentru ca nu faceam conversia rgb2gray.

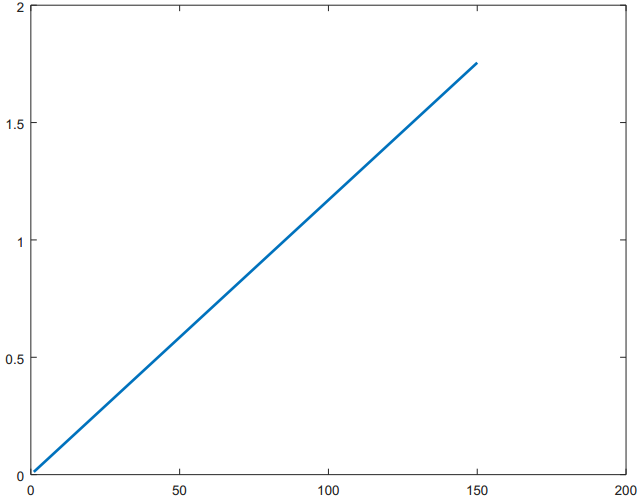
Am facut 2 variante corecte pentru taskurile 2 si 5 una in care taskurile sunt rezolvate integral in taskul 2 (am facut cele 16 grafice si astfel si mi se pare ca sunt putin mai putin exacte, dar in mare aceleasi) si inca una in care fiecare task este tratat individual (acestea sunt pastrate in arhiva trimisa deoarece mi s-a parut ca au grafice mai clare).

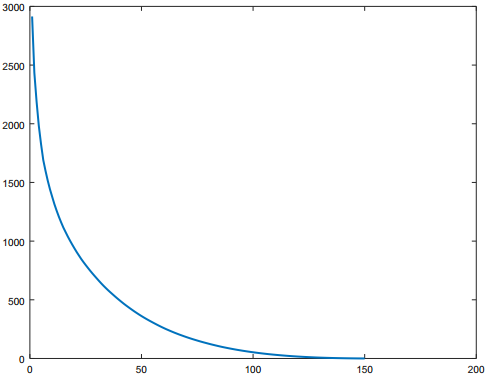
In cele ce urmeaza voi atasa cele 16 grafice cerute (obtinute cu versiunea din arhiva pentru taskurile 2 si 5, pentru imaginile 2 si 3, graficele vor fi mereu de la stanga la dreapta de sus in jos graficul 1, 2, 3 si 4 pentru taskul respectiv si imaginea respectiva):

Task-ul 2 imaginea a 2-a

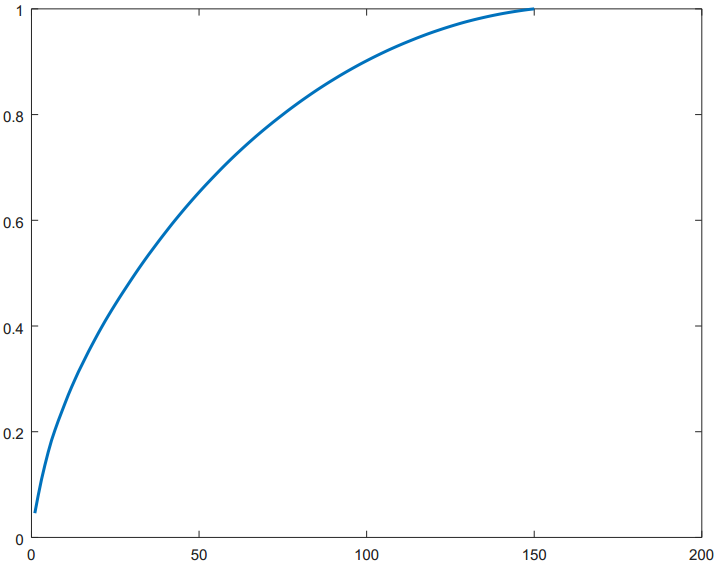


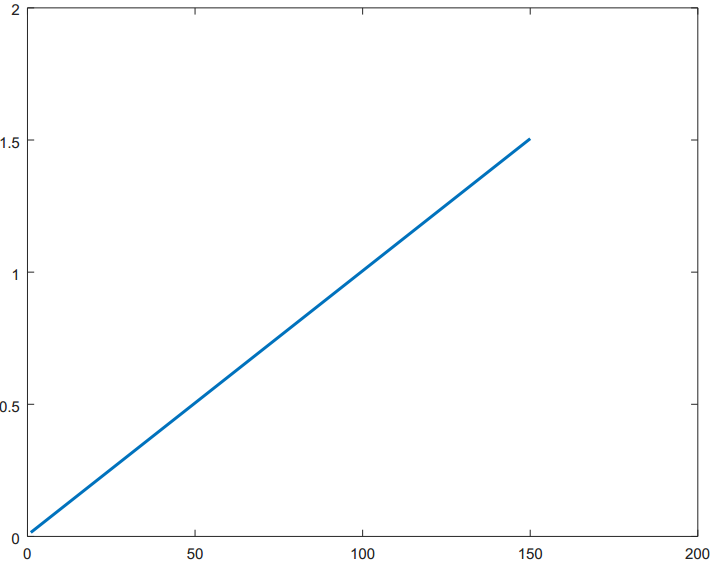
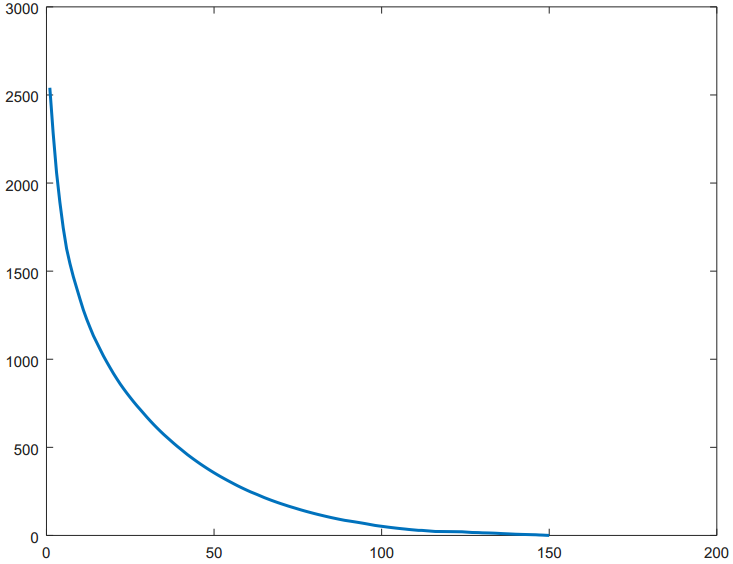
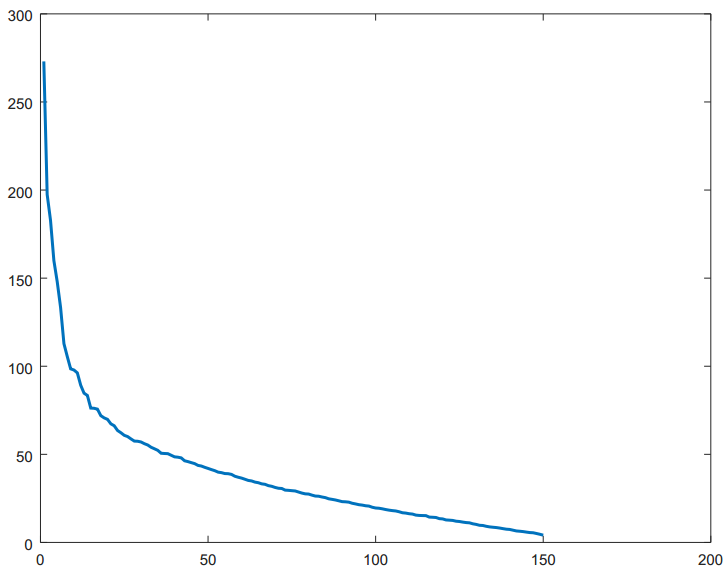




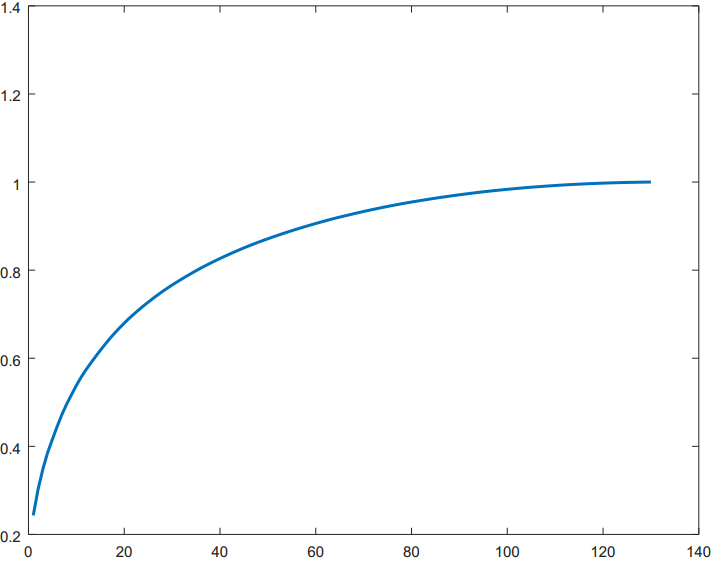


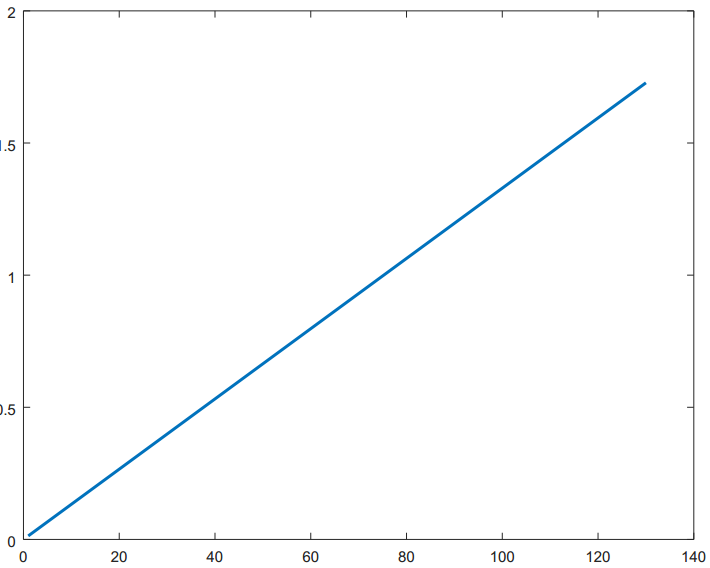
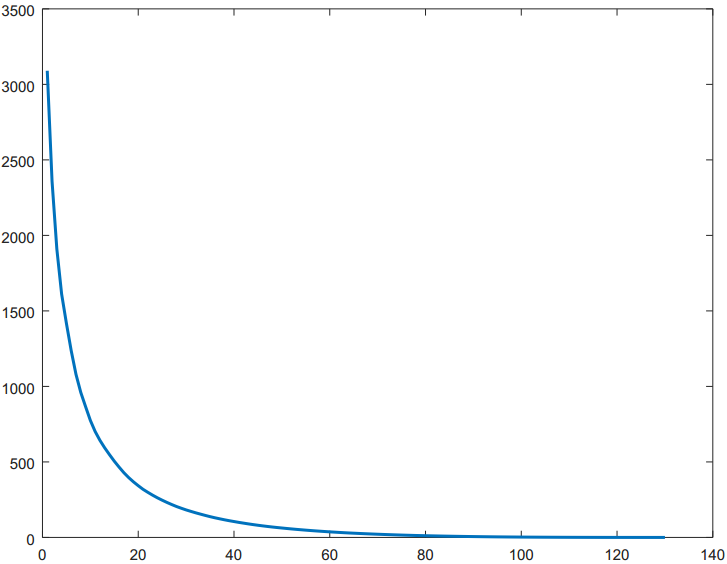
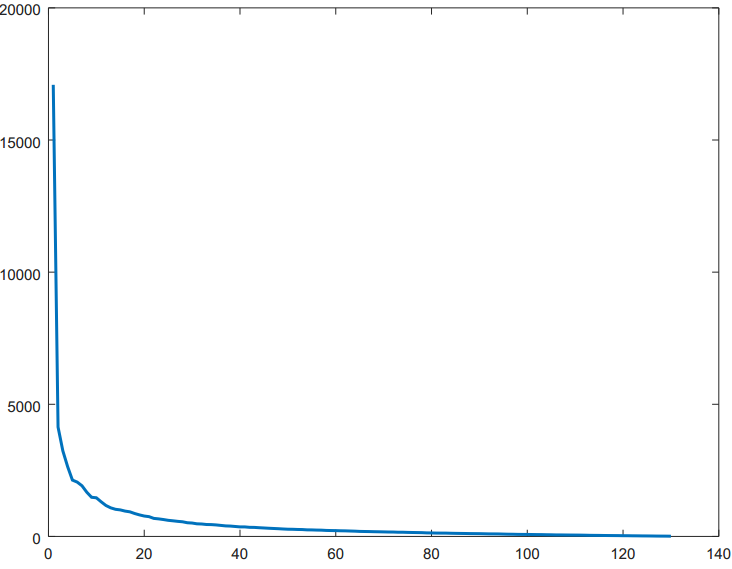
Task-ul 5 imaginea a 2-a



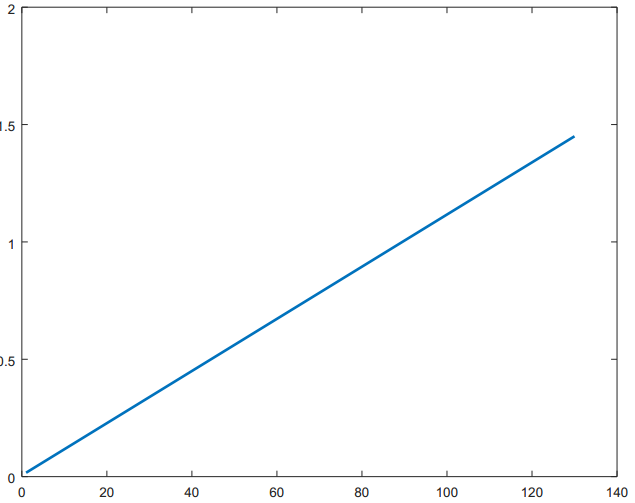
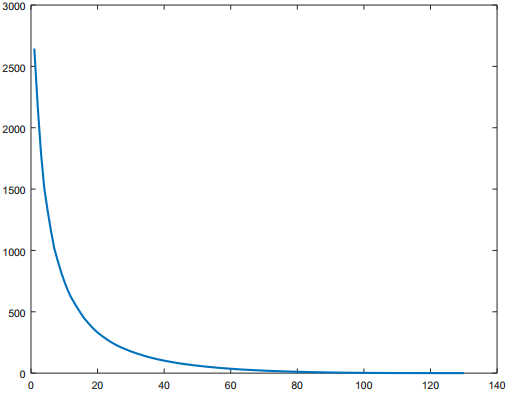
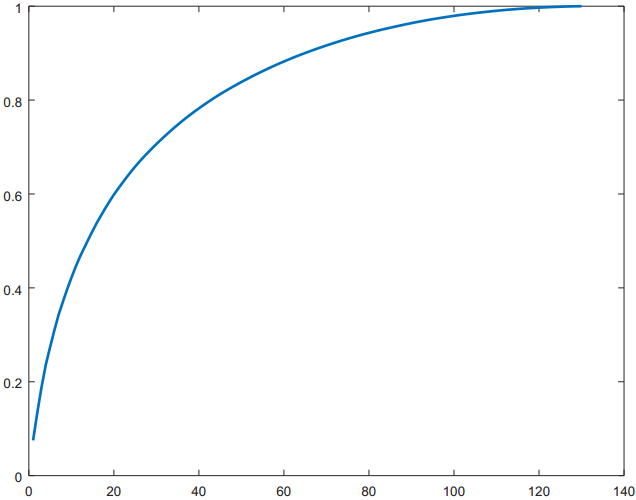
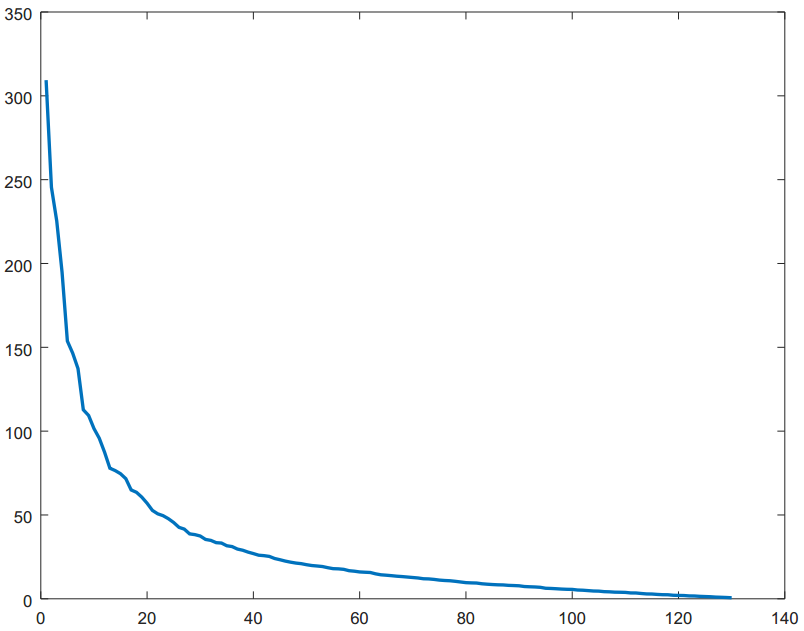


Task-ul 2 imaginea a 3-a





Task-ul 5 imaginea a 3-a



Rezolvarea taskurilor 2 si 5 la comun

In task2.m:

function task2(image, cerinta)

%image este calea catre imaginea pentru care dorim sa obtinem graficele

%cerinta este folosita pentru a deosebi taskul 2 de taskul 5

A=double(imread(image));

[m n]=size(A);

[u, s, v] = svd(A);

if(cerinta==5)

[A s]=task3(image, min(m,n));

endif

%transforma pe s in vector

s=diag(s);

suma\_numitor=sum(s);

%graficul 1

figure(1);

graph1=plot(s);

set(graph1,'LineWidth',2);

for i=1:min(m,n)

%obtinem cate o comprimare pentru fiecare k

k(i)=i;

A\_k = task1(image, i);

[u, s, v] = svd(A\_k);

s=diag(s);

%informatia data de primele k valori singulare//necesara pentru graficul 2

info(i)=sum(s)/suma\_numitor;

if(cerinta==5)

[A\_k s]=task3(image, i);

info(i)=sum(diag(s(1:k(i),1:k(i))))/sum(diag(s));

endif

%eroarea aproximarii pentru matricea A//necesara pentru graficul 3

eroare(i)=sum(sum(((A-A\_k).^2)/(m\*n)));

endfor

%graficul 2

figure(2);

graph2=plot(info);

set(graph2,'LineWidth',2);

%graficul 3

figure(3);

graph3=plot(eroare);

set(graph3,'LineWidth',2);

compression\_rate=(m\*k+n\*k+k)/(m\*n);

if(cerinta==5)

compression\_rate=(2\*k+1)/n;

endif

%graficul 4

figure(4);

graph4=plot(compression\_rate);

set(graph4,'LineWidth',2);

end

In task5.m:

function task5(image)

task2(image, 5);

end

Prezint aici 4 din cele 16 grafice pentru aceasta versiune (de la stanga spre dreapta de sus in jos, task2 imaginea 2 graficele 1 si 2 si pentru taskul 5 imaginea 2 graficele 3 si 4)

