PCP Problem And Image Deblurring

Jingxiang Wu

January 16, 2022

Contents

Chapter 1

PCP Problem And Several Algorithms For Solving RPCA

The problem of image deblurring can be regarded as the decomposition of a matrix into a low-rank matrix and a sparse matrix. The premise is that the original image is low-rank, or its singular value distribution is reasonable, in which each non-zero element of the sparse matrix is almost the same as that of the low-rank matrix.

1.1 The Principal Component Pursuit Problem

Let A is a given low-rank matrix with sparse noise, which can be decomposed into the sum of L and S. The goal is to reduce the rank of matrix L and the number of elements of matrix S, i.e.:

$$\begin{aligned}&\min_{L,S}\{rank(L), \|S\|_0\}\\&s.t. A = L + S\end{aligned}$$

We can add a balance factor λ to make the two relatively balanced, and notice that the rank of the matrix is in the order of magnitude of $O(\sqrt{n})$, and the element of the sparse matrix is in the order of magnitude of $O(n)$. Generally, λ is set to be $\frac{1}{\sqrt{mn}}$.

$$\begin{aligned}&\min_{L,S}(rank(L) + \lambda\|S\|_0)\\&s.t. A = L + S\end{aligned}$$

This problem is called Principal Component Pursuit.Considering that the rank sum matrix 0 norm cannot be derived, we relax this problem into a convex optimization problem, which is also called Robust Principal Component Analysis.

$$\begin{aligned}&\min(\|L\|_* + \lambda\|S\|_1)\\&s.t. A = L + S\end{aligned}$$

$\|L\|_*$ is the kernel norm of a matrix, the sum of all singular values. Then we introduce three algorithms for RPCA.

1.2 The Iterative Thresholding Algorithm

At first, we introduce two operators of matrices, which are the optimal solutions of a class of optimization problems.

The optimal solution of the optimization problem $\min_X(\varepsilon\|X\|_1 + \|X - Q\|_F^2/2)$ is $X = S_\varepsilon(Q)$, the element of matrix is $S_\varepsilon(Q)_{ij} = \max(|q_{ij}| - \varepsilon, 0) * \mathrm{sgn}(q_{ij})$.

The optimal solution of the optimization problem $\min_X(\varepsilon\|X\|_* + \|X - Q\|_F^2/2)$ is $X = D_\varepsilon(Q)$. Let the singular value decomposition of matrix Q be $Q = U\Sigma V^T$, and we can get: $D_\varepsilon(Q) = US_\varepsilon(\Sigma)V^T$

Considering the regularization of RPCA problem, we can get a new optimization problems.

$$\min(\|L\|_* + \lambda\|S\|_1 + \mu(\|L\|_F^2 + \|S\|_F^2)/2)$$
$$s.t. A = L + S$$

We can get the following iterative algorithm based on the above formula. [1]

Algorithm 1 Iterative Thresholding

Input: A: a matrix A and Maximum number of iterations iter times
Output: L: a low rank matrix; S: a sparse matrix

$\lambda = \frac{1}{\sqrt{mn}}$
$\tau = 0.9\|A\|_2$
$Y = 0$
Give δ_0
for $k = 1$ to iter times **do**
    $L_k = D_\tau(Y_{k-1})$
    $S_k = S_{\lambda\tau}(Y_{k-1})$
    $Y_k = Y_{k-1} + \delta_k(D - L_k - S_k)$
    update δ_k
end for

1.3 The Accelerated Proximal Gradient Algorithm

According to the slow iteration of the IT algorithm, people improve it and get the following algorithm, which is called Accelerated Proximal Gradient:

Algorithm 2 Accelerated Proximal Gradient

Input: A: a matrix A and Maximum number of iterations iter times
Output: L: a low rank matrix; S: a sparse matrix

$L_1 = L_0 = 0, S_1 = S_0 = 0, t_1 = t_0 = 1$
$\lambda = \frac{1}{\sqrt{mn}}$
$Lf = 2$
$\tau = 0.9\|A\|_2$
Give μ_0
for $k = 1$ to iter times **do**
    $Y_k^L = L_k + \frac{t_{k-1}-1}{t_k}(L_k - L_{k-1})$
    $Y_k^S = S_k + \frac{t_{k-1}-1}{t_k}(E_k - E_{k-1})$
    $L_k = D_{\frac{\mu_k}{L_f}}(Y_k^L + \frac{1}{2}(A - Y_k^L - Y_k^S))$
    $S_k = S_{\frac{\lambda\mu_k}{L_f}}(Y_k^S + \frac{1}{2}(A - Y_k^L - Y_k^S))$
    $\mu_{k+1} = \max(\eta\mu_k, \overline{\mu})$
    $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$
end for

1.4 The Augmented Lagrange Multipliers Algorithm

Using augmented Lagrangian multiplier method to solve PCP problem and constructing augmented Lagrangian Function : [2]

$$L(L,S,Y,\mu) = \|L\|_* + \lambda\|S\|_1 + \langle Y, A-L-S\rangle + \mu\,\|A-L-S\|_F^2\,/2$$

Before each update of the auxiliary matrix Y, we need to update L and S repeatedly in a cycle to converge. In order to improve the speed, people propose to update L and S only once, which is called inexact ALM:

Algorithm 3 Accelerated Proximal Gradient

Input: A: a matrix A and Maximum number of iterations iter times
Output: L: a low rank matrix; S: a sparse matrix

```
λ = 1/√(mn)
Y_0 = S_0 = 0
Give μ_0
for k = 1 to iter times do
    L_{k+1} = D_{1/μ_k}(A − S_k + μ_k^{-1} Y_k)
    S_{k+1} = S_{λ/μ_k}(A − L_{k+1} + μ_k^{-1} Y_k)
    Y_{k+1} = Y_k + μ_k(A − L_{k+1} − S_{k+1})
    update μ_k
end for
```

$\lambda = \frac{1}{\sqrt{mn}}$; $L_{k+1} = D_{\frac{1}{\mu_k}}(A - S_k + \mu_k^{-1}Y_k)$; $S_{k+1} = S_{\frac{\lambda}{\mu_k}}(A - L_{k+1} + \mu_k^{-1}Y_k)$; $Y_{k+1} = Y_k + \mu_k(A - L_{k+1} - S_{k+1})$

Chapter 2

Image Deblurring and Separation Testing

The relaxation problem of PCP problem can separate the low-rank matrix L and the sparse noise matrix S under the condition that the singular value distribution of the original matrix L is reasonable. We can apply this technology to the scene of repairing the image and removing high-intensity noise from the image. The premise is that the image should be relatively low-rank. Otherwise, we can choose a part of the image for this operation.

First, We choose several background maps, draw some stains on them, and then use three methods to separate low rank maps and noise maps. See figure 2.1, 2.2, 2.3.

Some images have elements that can be seen as sparse and high-intensity noise, so we can separate them with the three algorithms above. See figure 2.4, 2.5, 2.6.

After observation, IT algorithm and APG algorithm are often more accurate than the low-rank matrix obtained by IALM algorithm, and the obtained sparse noise matrix is closer to the noise added later.

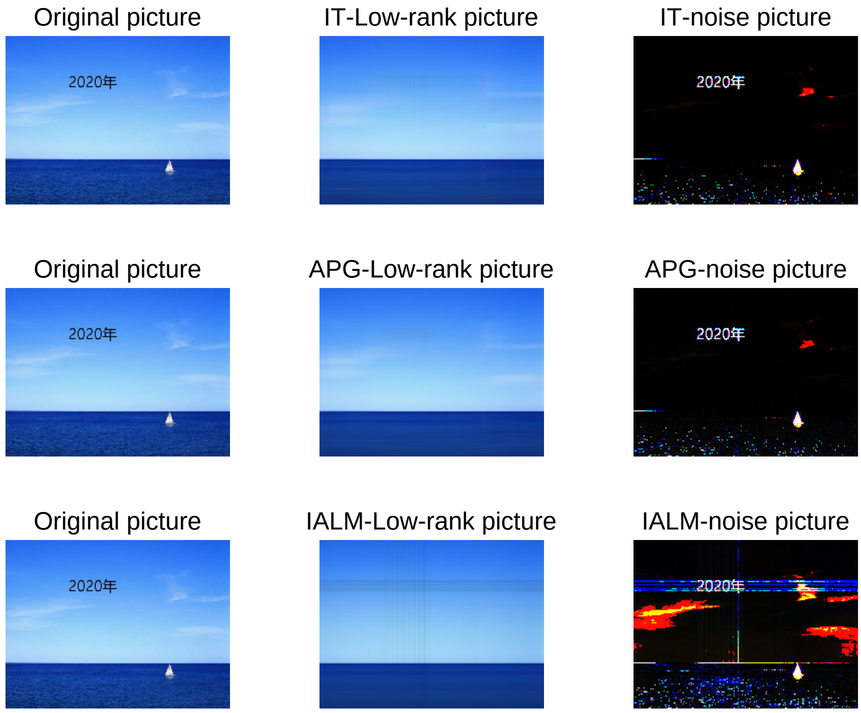

Figure 2.1: Image Deblurring A

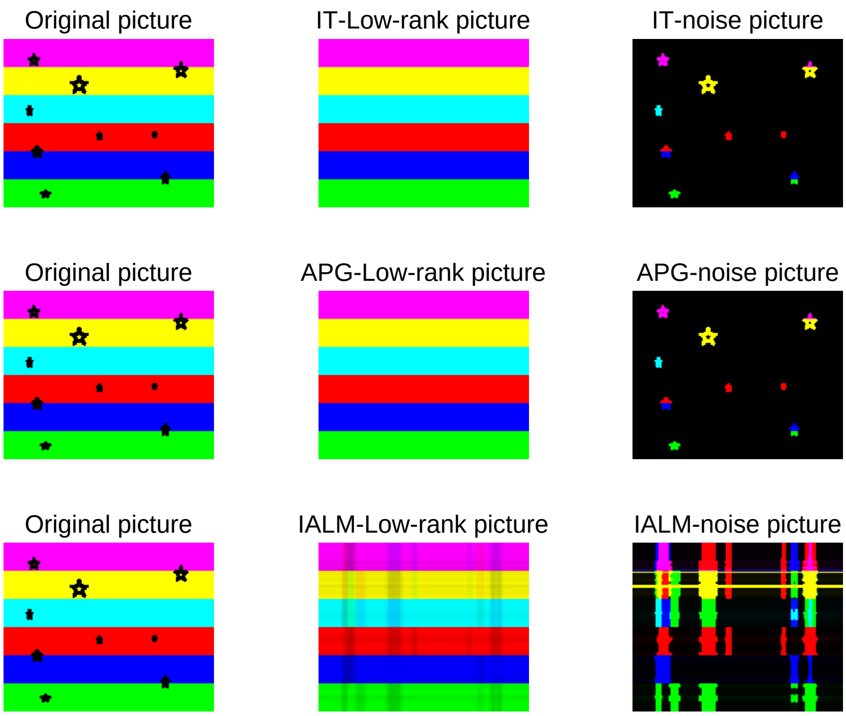

Figure 2.2: Image Deblurring B

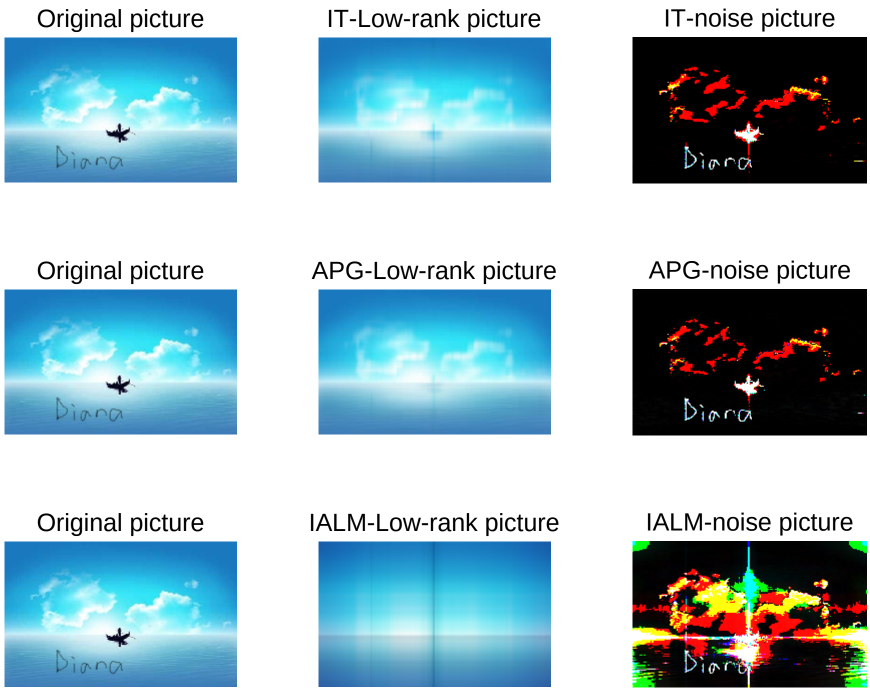

Figure 2.3: Image Deblurring C

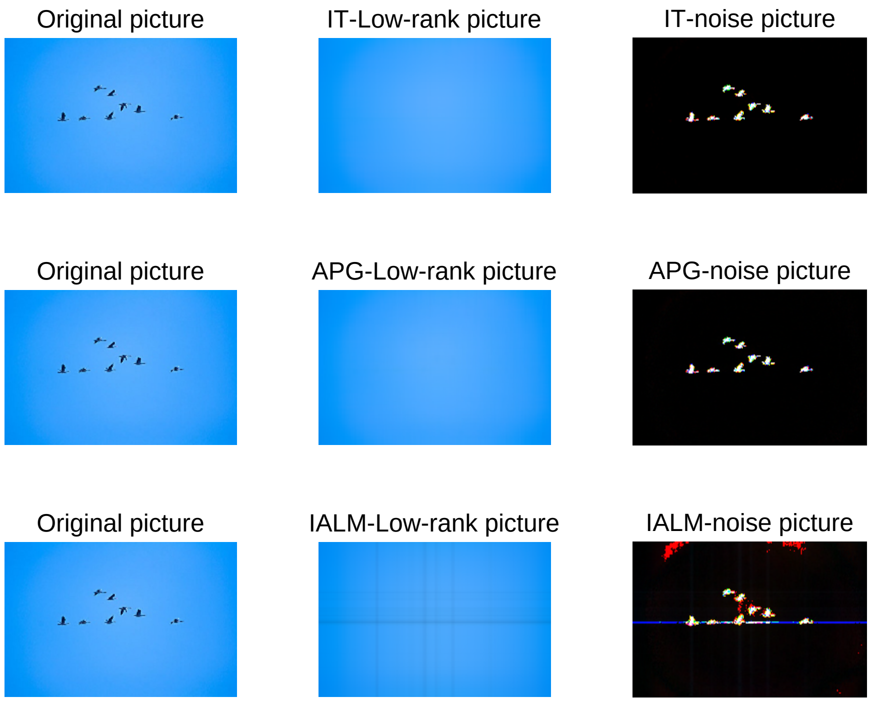

Figure 2.4: Image Separation A

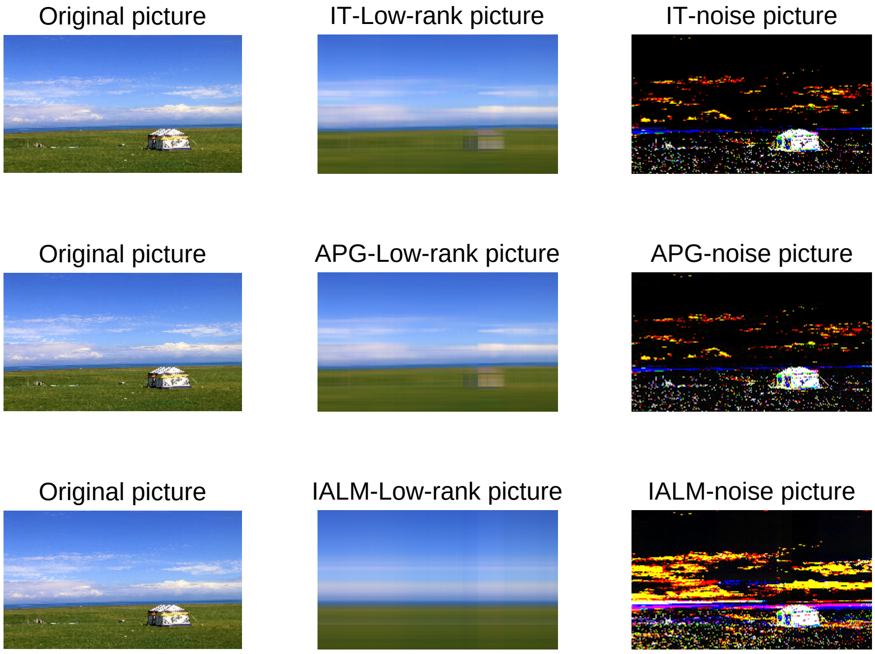

Figure 2.5: Image Separation B

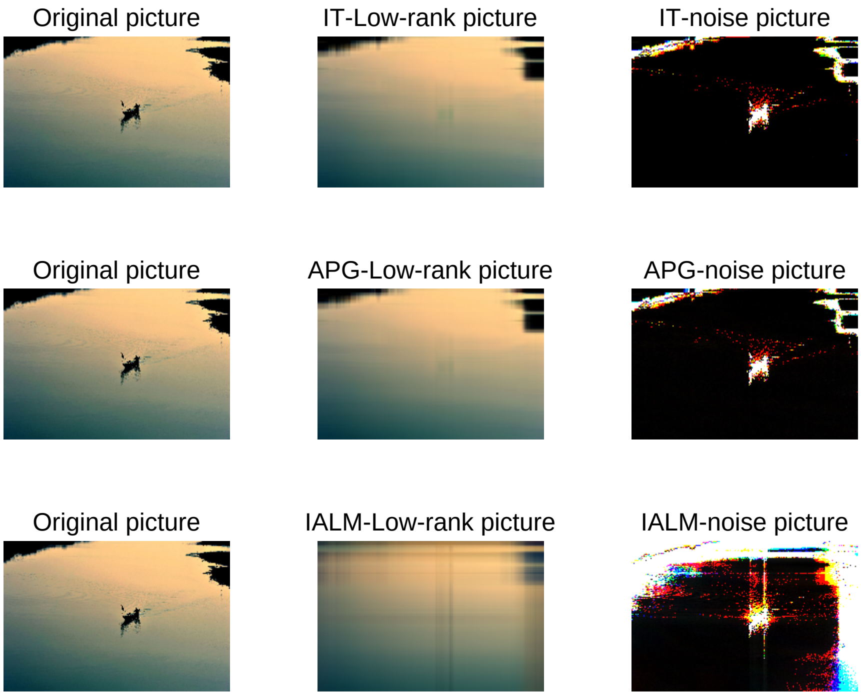

Figure 2.6: Image Separation C

Appendix A.

Reference For Test

```
%soft threshold operator
%input: a matrix Q and lambda
%output: a matrix B
function B=sto(Q,lambda)
B=sign(Q).*max(abs(Q)-lambda,0);
end
```

sto.m

```
%Singular value threshold operator
%input: a matrix Q and lambda
%output: a matrix B
function B=svto(Q,lambda)
[S,V,D]=svd(Q);
B=S*sto(V,lambda)*D';
end
```

svto.m

```
%IT method
%input: a matrix A;
%output: a matrix L and S s.t. A=L+S;
function [L,S]=IT(A,iter_times)
    [m,n]=size(A);
    lambda=10/sqrt(m*n);
    tau=0.9*norm(A,2);
    sigema=0.5;
    Y=zeros(m,n);
    for i=1:iter_times
        L=svto(Y,tau);
        S=sto(Y,lambda*tau);
        Y=Y+sigema*(A-L-S);
    end
end
```

IT.m

```
%APG method
%input:a matrix A;
%output: a matrix L and S s.t. A=L+S;
function [L,S]=APG(A,iter_times)
    [m,n]=size(A);
    lambda=10/sqrt(m*n);
    Lf=2;
    ita=0.9;
    miu=ita*norm(A,2);
    miu_min=1e-9*miu;
    t_last=1;
    t_new=1;
    L_last=zeros(m,n);
    S_last=zeros(m,n);
    L_new=A;
    S_new=S_last;
```

```
    for i=1:iter_times
        YL=L_new+(t_last-1)*(L_new-L_last)/t_new;
        YS=S_new+(t_last-1)*(S_new-S_last)/t_new;
        t_last=t_new;
        L_last=L_new;
        S_last=S_new;
        L_new=svto(YL+(A-YL-YS)/Lf,miu/2);
        S_new=sto(YS+(A-YL-YS)/Lf,lambda*miu/2);
        t_new=(1+sqrt(1+4*t_last^2))/2;
        miu=max(ita*miu,miu_min);
    end
    L=L_new;
    S=S_new;
end
```

APG.m

```
%IALM method
%input:a matrix A
%output:a matrix L and S s.t. A=L+S;
function [L,S]=IALM(A,iter_times)
    [m,n]=size(A);
    lambda=1/sqrt(m*n);
    mu=10*lambda;
    S=zeros(m,n);
    Y=zeros(m,n);
    for i=1:iter_times
        L=svto(A-S+Y/mu,1/mu);
        S=sto(A-L+Y/mu,lambda/mu);
        Y=Y+mu*(A-L-S);
    end
end
```

IALM.m

```
%project image deblurring
clear;
clc;
A=imread('C:\Users\LEGION\OneDrive -   \report\Tex2\picture\1.jpg');
B=imread('C:\Users\LEGION\OneDrive -   \report\Tex2\picture\2.jpg');
C=imread('C:\Users\LEGION\OneDrive -   \report\Tex2\picture\3.jpg');
[m1,n1,k1]=size(A);
[m2,n2,k2]=size(B);
[m3,n3,k3]=size(C);
%LA1=zeros(m1,n1,3);
%SA1=LA1;LA2=LA1;LA3=LA1;SA2=LA1;SA3=LA1;
%LA1=zeros(m2,n2,3);
%SA1=LA1;LA2=LA1;LA3=LA1;SA2=LA1;SA3=LA1;
%LA1=zeros(m3,n3,3);
%SA1=LA1;LA2=LA1;LA3=LA1;SA2=LA1;SA3=LA1;
for i=1:3
    [LA1(:,:,i),SA1(:,:,i)]=IT(double(A(:,:,i)),200);
    [LA2(:,:,i),SA2(:,:,i)]=APG(double(A(:,:,i)),200);
    [LA3(:,:,i),SA3(:,:,i)]=IALM(double(A(:,:,i)),200);
    [LB1(:,:,i),SB1(:,:,i)]=IT(double(B(:,:,i)),200);
    [LB2(:,:,i),SB2(:,:,i)]=APG(double(B(:,:,i)),200);
    [LB3(:,:,i),SB3(:,:,i)]=IALM(double(B(:,:,i)),200);
    [LC1(:,:,i),SC1(:,:,i)]=IT(double(C(:,:,i)),200);
    [LC2(:,:,i),SC2(:,:,i)]=APG(double(C(:,:,i)),200);
    [LC3(:,:,i),SC3(:,:,i)]=IALM(double(C(:,:,i)),200);
end
SA1=abs(SA1);
SA1(SA1>20)=255;
SA2=abs(SA2);
SA2(SA2>20)=255;
SA3=abs(SA3);
SA3(SA3>20)=255;
subplot(3,3,1);
imshow(A);
title('Original picture');
subplot(3,3,2);
imshow(uint8(LA1));
title('IT-Low-rank picture');
```

```
subplot(3,3,3);
imshow(uint8(SA1));
title('IT-noise picture');
subplot(3,3,4);
imshow(A);
title('Original picture');
subplot(3,3,5);
imshow(uint8(LA2));
title('APG-Low-rank picture');
subplot(3,3,6);
imshow(uint8(SA2));
title('APG-noise picture');
subplot(3,3,7);
imshow(A);
title('Original picture');
subplot(3,3,8);
imshow(uint8(LA3));
title('IALM-Low-rank picture');
subplot(3,3,9);
imshow(uint8(SA3));
title('IALM-noise picture');
```

project_image_inpainting.m

```
%project image separation
clear;
clc;
A=imread('C:\Users\LEGION\OneDrive -   \report\Tex2\picture\4.jpg');
B=imread('C:\Users\LEGION\OneDrive -   \report\Tex2\picture\5.jpg');
C=imread('C:\Users\LEGION\OneDrive -   \report\Tex2\picture\6.jpg');
for i=1:3
    [LA1(:,:,i),SA1(:,:,i)]=IT(double(A(:,:,i)),200);
    [LA2(:,:,i),SA2(:,:,i)]=APG(double(A(:,:,i)),200);
    [LA3(:,:,i),SA3(:,:,i)]=IALM(double(A(:,:,i)),200);
    [LB1(:,:,i),SB1(:,:,i)]=IT(double(B(:,:,i)),200);
    [LB2(:,:,i),SB2(:,:,i)]=APG(double(B(:,:,i)),200);
    [LB3(:,:,i),SB3(:,:,i)]=IALM(double(B(:,:,i)),200);
    [LC1(:,:,i),SC1(:,:,i)]=IT(double(C(:,:,i)),200);
    [LC2(:,:,i),SC2(:,:,i)]=APG(double(C(:,:,i)),200);
    [LC3(:,:,i),SC3(:,:,i)]=IALM(double(C(:,:,i)),200);
end
SA1=abs(SA1);
SA1(SA1>20)=255;
SA2=abs(SA2);
SA2(SA2>20)=255;
SA3=abs(SA3);
SA3(SA3>20)=255;
subplot(3,3,1);
imshow(A);
title('Original picture');
subplot(3,3,2);
imshow(uint8(LA1));
title('IT-Low-rank picture');
subplot(3,3,3);
imshow(uint8(SA1));
title('IT-noise picture');
subplot(3,3,4);
imshow(A);
title('Original picture');
subplot(3,3,5);
imshow(uint8(LA2));
title('APG-Low-rank picture');
subplot(3,3,6);
imshow(uint8(SA2));
title('APG-noise picture');
subplot(3,3,7);
imshow(A);
title('Original picture');
subplot(3,3,8);
imshow(uint8(LA3));
title('IALM-Low-rank picture');
subplot(3,3,9);
imshow(uint8(SA3));
```

```
title('IALM-noise picture');
```

image_separation.m

Reference

[1] Jian-Feng Cai, Emmanuel J. Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.

[2] Z. Lin, M. Chen, and Y. Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *eprint arxiv*, 9, 2010.