Part 1

Matrix Factorization with Alternating Least Squares and Gradient Descent

1. Implement the ALS update rules.

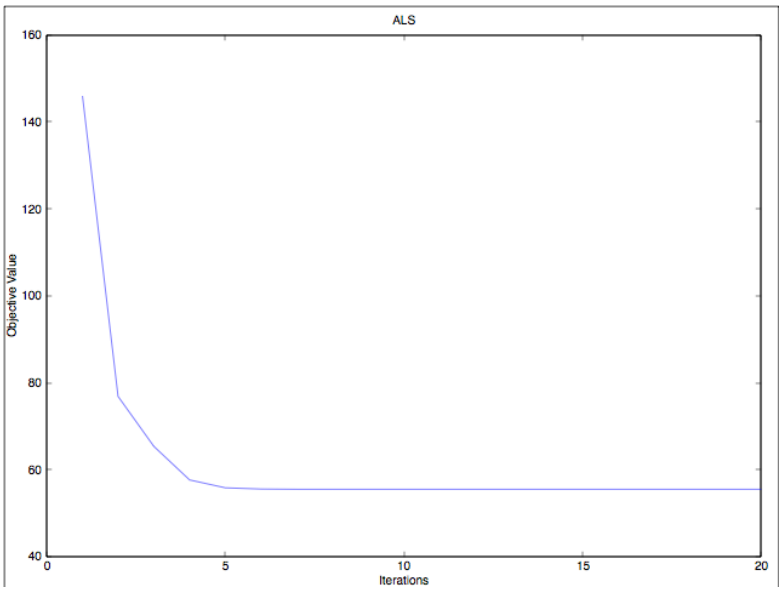
```
Wals = X*transpose(Hals)*inv((Hals*transpose(Hals)));
Hals = inv(transpose(Wals)*Wals)*transpose(Wals)*X;

Xhat = Wals * Hals;
```

2. Compute the objective function values for each iteration.

```
obj_als(i) = 0.5*norm((X-Xhat),'fro').^2;
```

The objective function values with the parameter (I = 10;J = 20; K = 3; number of iteration=20)



3. Implement the GD update rules.

```
Wgd=Wgd+eta*(X-Wgd*Hgd)*transpose(Hgd);
Hgd=Hgd+eta*transpose(Wgd)*(X-Wgd*Hgd);
```

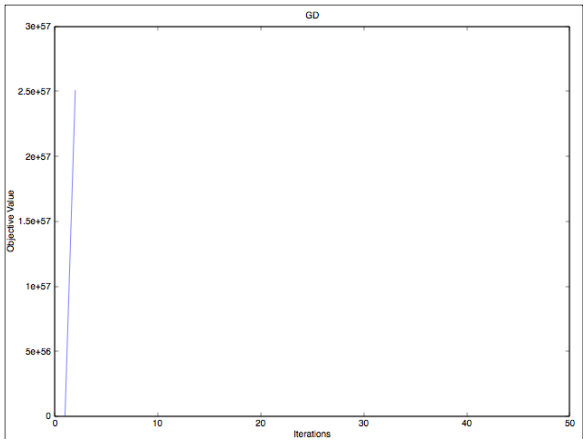
4. What is the effect of eta η?

The eta η has no effect to the method of Alternating Least Squares.

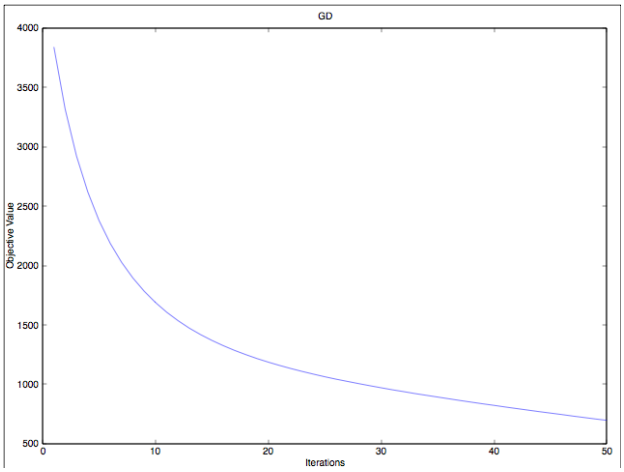
The eta η has effect to the method of Gradient Descent. It is called learning rate.

After increasing the eta η into 0.1, the difference between the predict matrix Xhat and the real matrix X will be increasing into infinite.

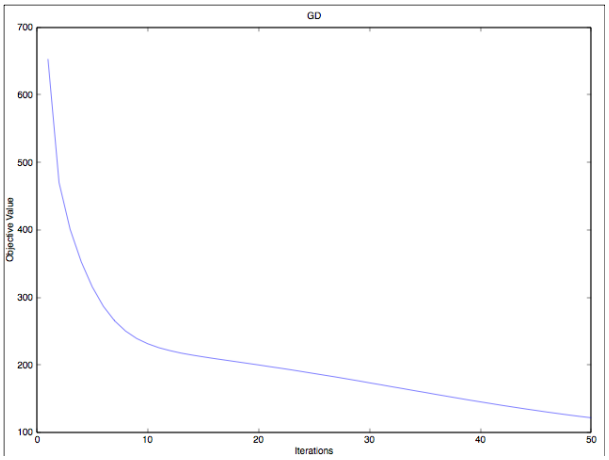
After decreasing the eta η into 0.01, the difference between the predict matrix Xhat and the real matrix X will increase smoothly.



Eta=0.1



Eta=0.001



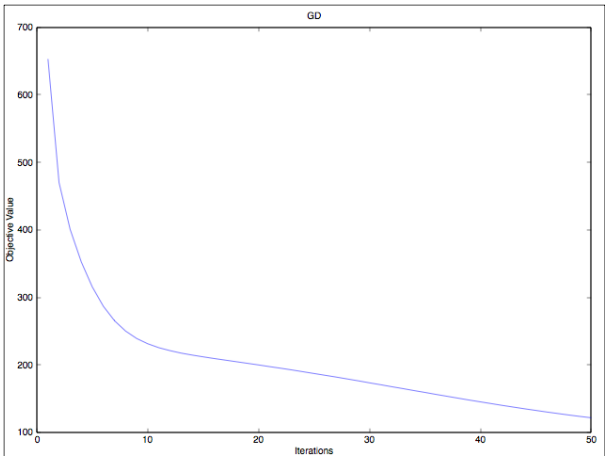
Eta=0.01

As we could see among those 3 graphs, still when the eta=0.1, we could get the best result. Either too high or too low will decrease the result.

5. Compute the objective function values for each iteration.

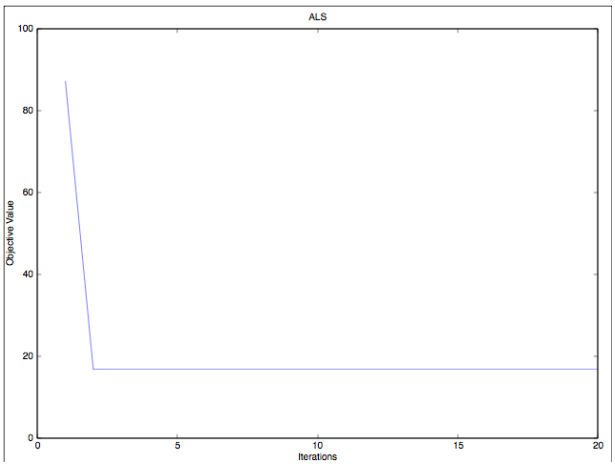
```
Xhat = Wgd * Hgd;
obj_gd(i) =0.5*norm((X-Xhat),'fro').^2;
```

The objective function values with the parameter (I = 10; J = 20; K = 3; number of iteration=50,eta=0.01)

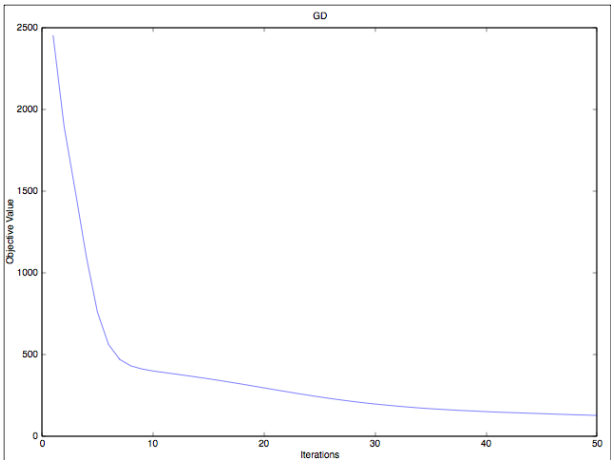


6. Play with the variable ‘dataNoise’. What is the effect of this parameter on the algorithms?

When data Noise=0.5, the difference between the real matrix X and Xhat of the method ASL decrease, but the difference between the real matrix X and Xhat of the method GD increase.

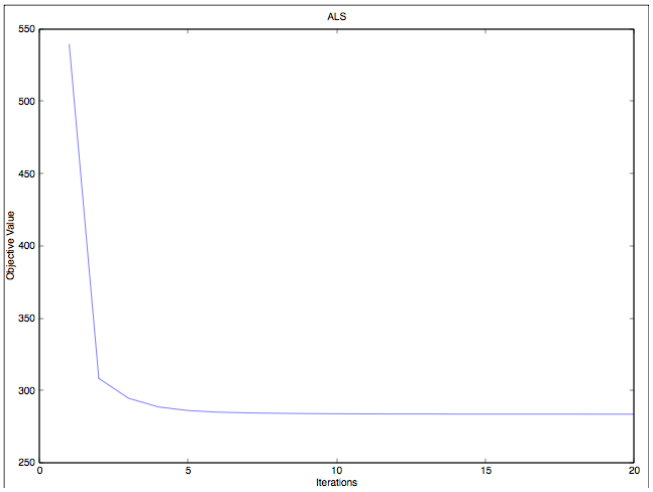


Noise = 0.5 , asl

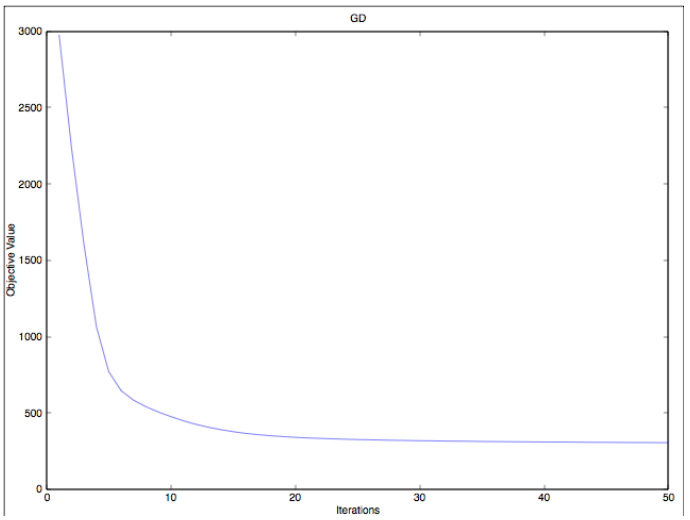


Noise = 0.5 , gd

When data Noise =2, the difference between the real matrix X and Xhat of the method GD increase, not only for the method ASL but also for the method GD.



Noise = 2 , asl



Noise = 2 , GD

7. Play with the size of the data (I, J) and the rank of the factorization K. What behavior do you observe?

First case fixed the I and J, change the value of K into 5.

For the method ALS, when I change the K=5 it will decrease the difference of the real matrix and the Xhat, and more quickly to get converge.

For the method GD, when I change the K=5, the difference of the real matrix and the Xhat is almost the same, however when K=5, it will converge quickly.

Second case fixed the J and K, change the value of I into 5.

For both the method ALS and GD, when I change the I=20 it will increase the difference of the real matrix and the Xhat.

Third case fixed the I and K, change the value of J into 40.

For both the method ALS and GD, when I change the J=40 it will increase the difference of the real matrix and the Xhat.

8. Which algorithm do you think is better? Why?

I think the method ALS is better. Because it could converge quickly. And in any case, it performance better than the method GD, the difference between real matrix X and the Xhat is smaller than result of the method GD.

Part 2

Non-Negative Matrix Factorization with Multiplicative Update Rules

1. Implement the MUR update rules.

   Xhat=Wmur*Hmur+eps;

   tmp=((X./Xhat)*transpose(Hmur))./(O*transpose(Hmur));

   Wmur=times(Wmur,tmp);

   tmp=(transpose(Wmur)*(X./Xhat))./(transpose(Wmur)*O);

   Hmur=times(Hmur,tmp);

2. Compute the objective function values for each iteration.

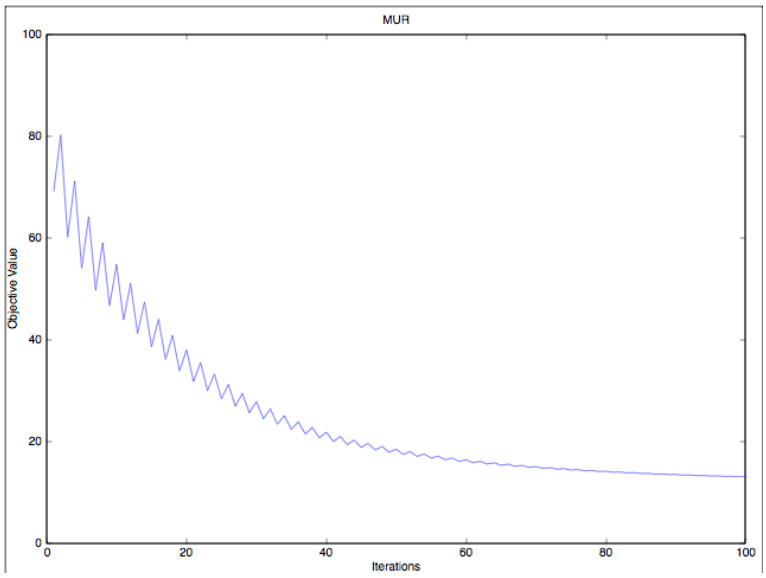
   Xhat = Wmur * Hmur;

   Xhat = Xhat + eps; %for numerical stability

   tmp=X.*log(X./Xhat)-X+Xhat;

   obj_mur(i) = sum(tmp(:));

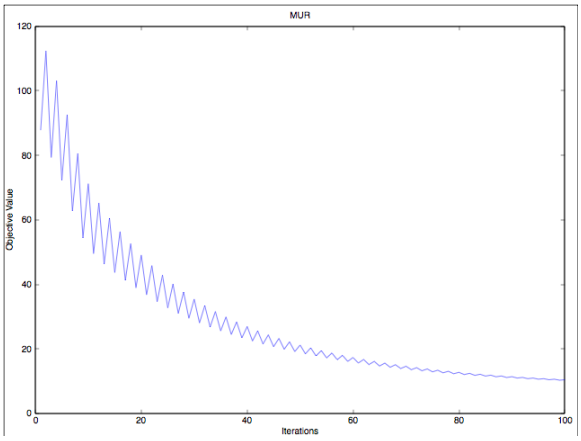
   with the parameter I = 10; J = 20; K = 3; MaxIterMur = 100;

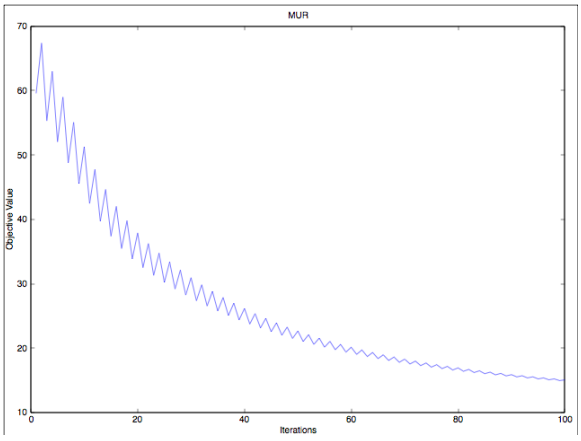


3. Play with the variable ‘dataNoise’. What is the effect of this parameter on the algorithms?

data Noise= 1.5, the peak of the difference between the real matrix X and Xhat will be smaller, but at the end of the iteration, the algorithm still not converge.

data Noise= 0.8, the peak of the difference between the real matrix X and Xhat will be higher, the algorithm will converge slower than the result of data noise=1.





data Noise= 0.8 data Noise= 1.5

4. Play with the size of the data (I, J) and the rank of the factorization K. What behaviour do you observe?

First case fixed I and J, change K into 10

The fluctuation of the graph will larger than the result of K=3, and at the end of the iteration the algorithm still doesn’t converge, the difference between the real matrix X and Xhat will be higher.

Second case fixed K and J, change I into 20

The peak of the difference will increase, the difference between the real matrix X and Xhat will be higher, it also converges slower than the default case.

Second case fixed I and K, change J into 10

The peak of the difference will decrease, the difference between the real matrix X and Xhat will be smaller, but it converges slower than the default case.