# DMM Assignment 2 Solutions

## 1. Computing the Loss Function

### 1.a

for the fact that $l_i^T r_j = \sum_{k=1}^{r} l_{ik} r_{kj}$ , $||l_i||_2^2 = \sum_{k=1}^{r} l_{ik}^2$ and $l_i^T r_j = \sum_{k=1}^{r} l_{ik} r_{kj}$, function (2) = function (3)

But I don't understand why function (1)=function(2)

### 1.b

- the loss of factorization LR with L2 regularization (parameter lambda) are presented as below:

```
loss <- function(L, R, lambda) {
loss1<-0;
loss2<-0;
i<-1;
while(i<(nrow(summary(D))+1)){
loss1<-loss1+ (ds[i]-L[is[i],]%*%R[,js[i]])^2;
i<-i+1;
}
loss2<-(lambda/2)*(norm(L,'F')^2+norm(R,'F')^2)
loss<-loss1+loss2;
return(loss);
}
```

- verify the result

```
## loss of starting point with lambda=2 (should give 3865.816)
loss(L0.r10, R0.r10, 2)
[1] 3865.816
```

# 2. Computing the Local Gradients

## 2.a

$$\nabla_{l_{ik}} L_{ij}(l_i, r_j) = -2(d_{ij} - l_i^T r_j) r_{kj} + \frac{\lambda l_{ik}}{N_i}$$

$$\nabla_{r_{kj}} L_{ij}(l_i, r_j) = -2 l_{ik}(d_{ij} - l_i^T r_j) + \frac{\lambda r_{kj}}{N_j}$$

## 2.b

$$\nabla_{l_i^T} L_{ij}(l_i, r_j) = -2(d_{ij} - l_i^T r_j) r_j^T + \frac{\lambda ||l_i||_2}{N_i} l_i$$

$$\nabla_{r_j^T} L_{ij}(l_i, r_j) = -2 l_i^T (d_{ij} - l_i^T r_j) + \frac{\lambda ||r_j||_2}{N_j} r_j$$

## 2.c

- dlossp function are presented as follow:

```
dlossp <- function(L, R, lambda, p) {
   i <- is[p]
   j <- js[p]
   d <- ds[p]
   ## create two length-r vectors dLi and dRj
dLi<-rep(0,r);
dRj<-rep(0,r)
pl<-1;
k<-1;
while(k<(r+1)){
item1<-(-2)* (d-L[i,]%*%R[,j]);
dLi[pl]<-item1* R[k,j]+lambda* L[i,k]/ Nis[i];
dRj[pl]<-item1* L[i,k]+lambda* R[k,j]/ Njs[j] ;
k<-k+1;
pl<-pl+1;
}
   ## * dLi[k] contains the gradient of the local loss with respect to l_ik
   ## * dRi[k] contains the gradient of the local loss with respect to r_kj


   ## return the result as a list
   ## (elements can be accessed with x$Li or x$Ri, where x is the returned list)

list(Li=dLi, Rj=dRj)
}
x<- dlossp(L0.r10, R0.r10, 2, 10)
x$Li
x$Ri
```

- verify the result

```
## test local gradient computation for 10th entry
## $Li
##  [1] -0.022076386  0.055660021  0.016555157  0.068076690  0.025315001
##  [6]  0.001815675  0.078699632  0.026119761  0.013711859  0.004034443
```

```
##
## $Rj
##  [1] -0.03494084  0.07994425  0.05441889  0.05581750  0.01939717 -0.01477114
##  [7]  0.04630433  0.02940866 -0.05088697  0.01276903
dlossp(L0.r10, R0.r10, 2, 10)
```

# 3. Implementing Gradient Descent

- a gradient descent epoch

```
## updated factor matrices.
gdepoch <- function(L, R, lambda, eps) {
   ## create gradient matrices
   dL <- matrix(0, m, r)
   dR <- matrix(0, r, n)

   ## fill the gradient matrices using repeated calls to my dlossp function
p<-1;
while(p<(nrow(summary(D))+1)){
x<-dlossp(L, R, lambda,p);
dL[is[p],]<- dL[is[p],]+x$Li;
dR[,js[p]]<- dR[,js[p]]+x$Rj;
p<-p+1;
}
   ## perform a gradient step on L and R with step size eps by using the gradient matrices

L<-L-eps*dL
R<-R-eps*dR

   ## return result
   list(L=L, R=R)
}
```

- verify my implement successfully with the result of gdepoch(L0.r10, R0.r10, 2, 0.01)$L[5,]

```
##  [1]  0.07822179  0.29110406  0.67363404 -0.27527854 -0.36139164  0.20011846
##  [7] -0.35296692  0.08101832 -0.01417779  0.01627573
```

- Visualize the result after 5, 15, and 50 epochs as below.

- for 5 epochs

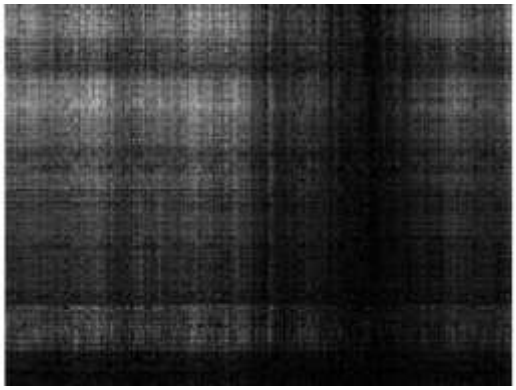
```
Epoch 1 , loss: 3865.816
Epoch 2 , loss: 2765.026
Epoch 3 , loss: 2293.826
Epoch 4 , loss: 2030.471
Epoch 5 , loss: 1858.877
```

- result for 5 epochs

- for 15 epochs (continue with the previous result for 10 more epoch)

```
Epoch 1 , loss: 1731.708
Epoch 2 , loss: 1646.502
Epoch 3 , loss: 1562.573
Epoch 4 , loss: 1472.45
Epoch 5 , loss: 1369.19
Epoch 6 , loss: 1248.171
Epoch 7 , loss: 1111.246
Epoch 8 , loss: 971.3808
Epoch 9 , loss: 850.3848
Epoch 10 , loss: 764.233
```

- result for 15 epochs

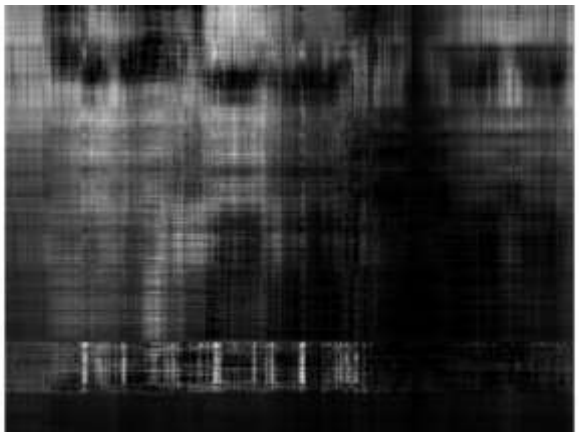


- for 50 epochs (continue with the previous result for 35 more epoch)

```
Epoch 1 , loss: 710.816
Epoch 2 , loss: 689.5328
Epoch 3 , loss: 672.4398
Epoch 4 , loss: 658.2506
Epoch 5 , loss: 646.011
Epoch 6 , loss: 635.0533
Epoch 7 , loss: 624.922
Epoch 8 , loss: 615.3069
Epoch 9 , loss: 605.9942
Epoch 10 , loss: 596.834
Epoch 11 , loss: 587.7222
Epoch 12 , loss: 578.5907
Epoch 13 , loss: 569.405
Epoch 14 , loss: 560.165
Epoch 15 , loss: 550.9066
Epoch 16 , loss: 541.7009
Epoch 17 , loss: 532.6482
Epoch 18 , loss: 523.8656
Epoch 19 , loss: 515.4685
Epoch 20 , loss: 507.5508
Epoch 21 , loss: 500.1693
Epoch 22 , loss: 493.3377
Epoch 23 , loss: 487.0327
Epoch 24 , loss: 481.2066
Epoch 25 , loss: 475.8024
Epoch 26 , loss: 470.7661
Epoch 27 , loss: 466.0527
Epoch 28 , loss: 461.6285
Epoch 29 , loss: 457.4695
Epoch 30 , loss: 453.5593
Epoch 31 , loss: 449.8856
Epoch 32 , loss: 446.438
```

```
Epoch 33 , loss: 443.2064
Epoch 34 , loss: 440.1796
Epoch 35 , loss: 437.3454
```

- result for 50 epochs



**sub comment:**

the image gets clear with the increase of the epoch times.

# 4. Implementing Stochastic Gradient Descent

- example run

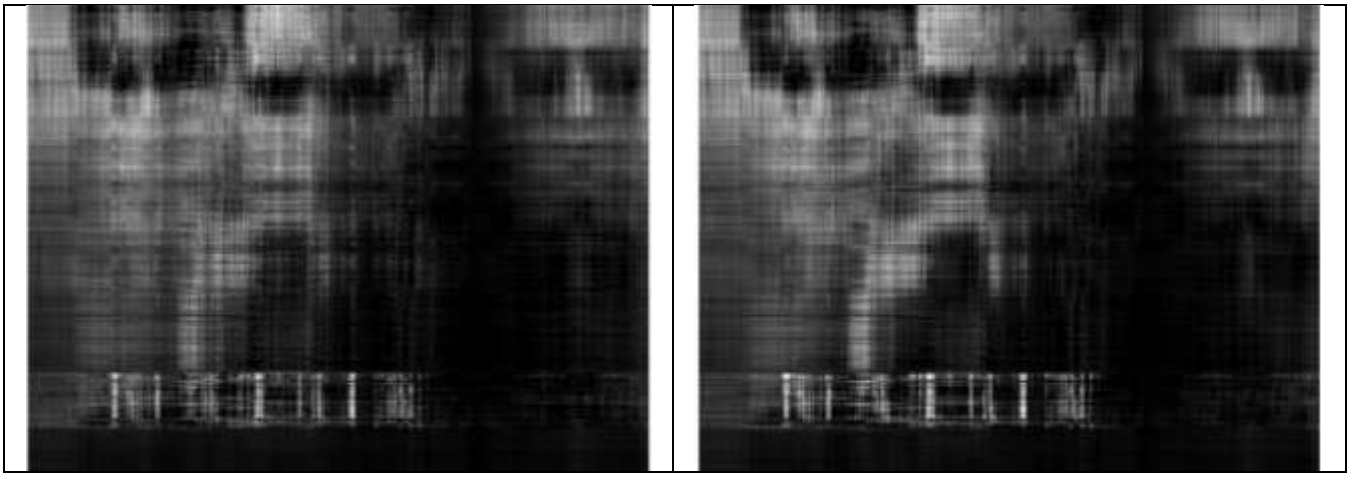
```
Epoch 1 , loss: 3865.816
Epoch 2 , loss: 2898.93
Epoch 3 , loss: 2405.895
Epoch 4 , loss: 2112.695
Epoch 5 , loss: 1915.663
Epoch 6 , loss: 1766.437
Epoch 7 , loss: 1635.768
Epoch 8 , loss: 1502.574
Epoch 9 , loss: 1351.592
Epoch 10 , loss: 1179.036
Epoch 11 , loss: 1004.16
Epoch 12 , loss: 861.1181
Epoch 13 , loss: 766.3913
Epoch 14 , loss: 710.1643
Epoch 15 , loss: 675.3782
Epoch 16 , loss: 651.2923
Epoch 17 , loss: 632.3622
Epoch 18 , loss: 615.8893
Epoch 19 , loss: 600.6271
Epoch 20 , loss: 585.9487
Epoch 21 , loss: 571.5641
Epoch 22 , loss: 557.4279
Epoch 23 , loss: 543.7156
Epoch 24 , loss: 530.6066
Epoch 25 , loss: 518.5132
Epoch 26 , loss: 507.5284
Epoch 27 , loss: 497.6622
Epoch 28 , loss: 488.9529
Epoch 29 , loss: 481.1627
Epoch 30 , loss: 474.237
Epoch 31 , loss: 467.8875
Epoch 32 , loss: 462.2237
Epoch 33 , loss: 457.1002
Epoch 34 , loss: 452.2036
Epoch 35 , loss: 448.0898
Epoch 36 , loss: 444.1507
Epoch 37 , loss: 440.6577
Epoch 38 , loss: 437.3141
Epoch 39 , loss: 434.5792
Epoch 40 , loss: 431.81
Epoch 41 , loss: 429.4726
Epoch 42 , loss: 427.5922
Epoch 43 , loss: 425.5747
Epoch 44 , loss: 424.1597
Epoch 45 , loss: 422.7861
Epoch 46 , loss: 421.9547
Epoch 47 , loss: 421.0838
Epoch 48 , loss: 420.2283
Epoch 49 , loss: 420.2222
Epoch 50 , loss: 419.0738
```

- show result

Compare the result with the result obtained after the example run.

- compare the images

| GD (with epoch = 50) | SGD (with epoch = 50) |
| --- | --- |

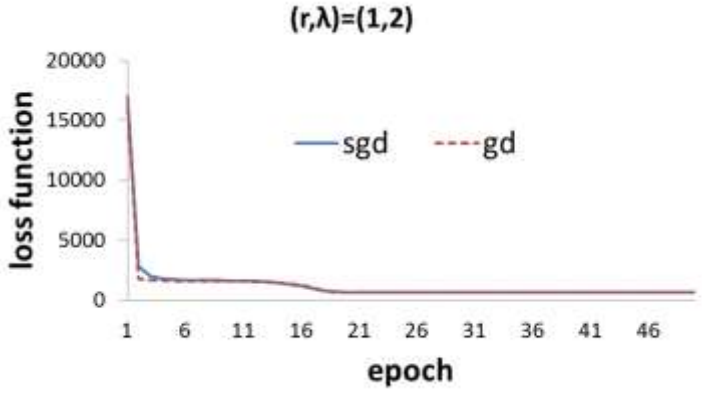


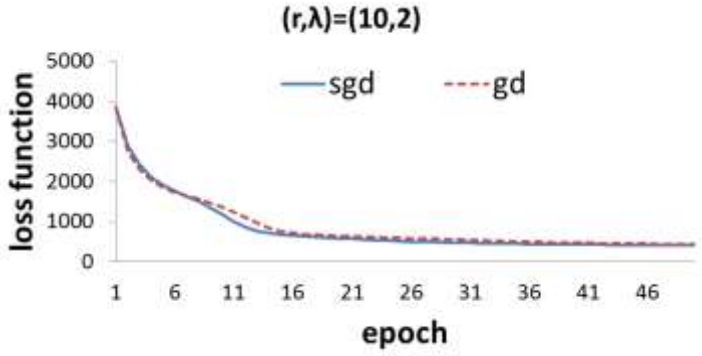
- compare SGD and GD with more trials

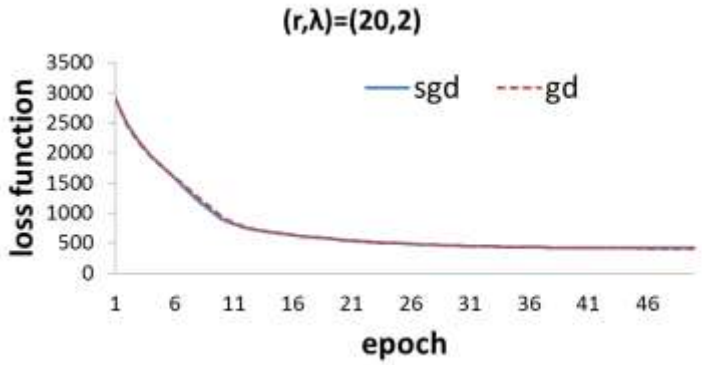
| | (r,λ)= (1,2) | | (r,λ)= (10,2) | | (r,λ)= (20,2) | |
|---|---|---|---|---|---|---|
| epoch | sgd | gd | sgd | gd | sgd | gd |
| 1 | 17042 | 17042 | 3866 | 3866 | 2911 | 2911 |
| 2 | 2825 | 1793 | 2899 | 2765 | 2483 | 2452 |
| 3 | 1971 | 1683 | 2406 | 2294 | 2188 | 2155 |
| 4 | 1766 | 1637 | 2113 | 2030 | 1962 | 1938 |
| 5 | 1694 | 1613 | 1916 | 1859 | 1770 | 1761 |
| 6 | 1663 | 1598 | 1766 | 1732 | 1587 | 1600 |
| 7 | 1646 | 1587 | 1636 | 1647 | 1401 | 1437 |
| 8 | 1634 | 1579 | 1503 | 1563 | 1213 | 1267 |
| 9 | 1623 | 1572 | 1352 | 1472 | 1040 | 1094 |
| 10 | 1611 | 1565 | 1179 | 1369 | 905 | 941 |
| 11 | 1594 | 1555 | 1004 | 1248 | 813 | 829 |
| 12 | 1569 | 1540 | 861 | 1111 | 752 | 758 |
| 13 | 1529 | 1514 | 766 | 971 | 711 | 713 |
| 14 | 1460 | 1468 | 710 | 850 | 679 | 680 |
| 15 | 1347 | 1384 | 675 | 764 | 652 | 654 |
| 16 | 1178 | 1239 | 651 | 711 | 629 | 631 |
| 17 | 978 | 1025 | 632 | 690 | 609 | 610 |
| 18 | 807 | 802 | 616 | 672 | 589 | 591 |
| 19 | 705 | 673 | 601 | 658 | 572 | 574 |
| 20 | 658 | 635 | 586 | 646 | 555 | 557 |
| 21 | 640 | 628 | 572 | 635 | 540 | 542 |
| 22 | 633 | 627 | 557 | 625 | 526 | 528 |
| 23 | 630 | 627 | 544 | 615 | 513 | 515 |
| 24 | 629 | 626 | 531 | 606 | 501 | 503 |
| 25 | 628 | 626 | 519 | 597 | 491 | 493 |
| 26 | 628 | 626 | 508 | 588 | 481 | 483 |
| 27 | 628 | 626 | 498 | 579 | 472 | 475 |
| 28 | 628 | 626 | 489 | 569 | 465 | 467 |
| 29 | 628 | 626 | 481 | 560 | 458 | 460 |
| 30 | 627 | 626 | 474 | 551 | 452 | 454 |
| 31 | 627 | 626 | 468 | 542 | 446 | 448 |
| 32 | 627 | 626 | 462 | 533 | 441 | 443 |
| 33 | 627 | 626 | 457 | 524 | 437 | 439 |
| 34 | 627 | 626 | 452 | 515 | 433 | 434 |
| 35 | 627 | 626 | 448 | 508 | 430 | 431 |
| 36 | 627 | 626 | 444 | 500 | 427 | 427 |
| 37 | 627 | 626 | 441 | 493 | 424 | 425 |
| 38 | 627 | 626 | 437 | 487 | 421 | 422 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 39 | 626 | 626 | 435 | 481 | 419 | 420 |
| 40 | 626 | 626 | 432 | 476 | 418 | 418 |
| 41 | 626 | 626 | 429 | 471 | 416 | 416 |
| 42 | 626 | 626 | 428 | 466 | 415 | 414 |
| 43 | 626 | 626 | 426 | 462 | 414 | 413 |
| 44 | 626 | 626 | 424 | 457 | 413 | 411 |
| 45 | 626 | 626 | 423 | 454 | 412 | 410 |
| 46 | 626 | 626 | 422 | 450 | 412 | 409 |
| 47 | 626 | 626 | 421 | 446 | 411 | 409 |
| 48 | 626 | 626 | 420 | 443 | 411 | 408 |
| 49 | 626 | 626 | 420 | 440 | 410 | 407 |
| 50 | 626 | 626 | 419 | 437 | 410 | 406 |
| **local loss** | **489** | **489** | **126** | **151** | **94** | **98** |
| **reg. Term** | **137** | **137** | **293** | **286** | **316** | **308** |

- compare SGD and GD with more trials (cont.)




**sub comment:**
the sgd seems to be not has the speeding up effect as expected in these trials.

# 5. Impact of Parameter Choices

i) For the simplicity of the operation, I will use the name of “L0.r10” and “R0.r10” for different rank value.
ii) For the effectiveness of comparison, I will choose the stochastic gradient descent method as the approach; but the SGD method is easily to cause the stop in the local minimum, in this way, we cannot see the effect of r and lambda clearly. So I will use the gradient descent (GD) method.
iii) Also for the comparing purpose, I will use the same value for “epochs” for each trial and the process will begin with the same “set.seed”.
iv) For comparing purpose, I will do 9 scenario for r = 1, 10, and 20 pairing with λ = 0, 2, 20.

- Values for each loss function

| epoch \ (r,λ) | (1,0) | (1,2) | (1,20) | (10,0) | (10,2) | (10,20) | (20,0) | (20,2) | (20,20) |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 16477 | 17042 | 22128 | 3282 | 3866 | 9117 | 2344 | 2911 | 8016 |
| 2 | 1731 | 1793 | 2854 | 2407 | 2765 | 4585 | 2009 | 2452 | 4854 |
| 3 | 1644 | 1683 | 2033 | 2018 | 2294 | 3129 | 1776 | 2155 | 3399 |
| 4 | 1605 | 1637 | 1833 | 1795 | 2030 | 2433 | 1593 | 1938 | 2617 |
| 5 | 1583 | 1613 | 1723 | 1644 | 1859 | 2060 | 1431 | 1761 | 2172 |
| 6 | 1569 | 1598 | 1659 | 1526 | 1732 | 1849 | 1270 | 1600 | 1911 |
| 7 | 1559 | 1587 | 1622 | 1420 | 1624 | 1727 | 1099 | 1437 | 1757 |
| 8 | 1550 | 1579 | 1600 | 1309 | 1518 | 1656 | 916 | 1267 | 1664 |
| 9 | 1542 | 1572 | 1588 | 1181 | 1399 | 1614 | 737 | 1094 | 1607 |
| 10 | 1532 | 1565 | 1581 | 1027 | 1257 | 1587 | 588 | 941 | 1570 |
| 11 | 1519 | 1555 | 1577 | 855 | 1093 | 1570 | 484 | 829 | 1543 |
| 12 | 1498 | 1540 | 1575 | 690 | 928 | 1557 | 418 | 758 | 1523 |
| 13 | 1461 | 1514 | 1574 | 564 | 800 | 1545 | 374 | 713 | 1505 |
| 14 | 1396 | 1468 | 1573 | 486 | 723 | 1532 | 342 | 680 | 1489 |
| 15 | 1283 | 1384 | 1573 | 440 | 680 | 1519 | 315 | 654 | 1476 |
| 16 | 1103 | 1239 | 1573 | 410 | 654 | 1504 | 292 | 631 | 1466 |
| 17 | 867 | 1025 | 1572 | 387 | 634 | 1489 | 270 | 610 | 1459 |
| 18 | 651 | 802 | 1572 | 368 | 618 | 1475 | 250 | 591 | 1455 |
| 19 | 533 | 673 | 1572 | 350 | 603 | 1464 | 232 | 574 | 1453 |
| 20 | 495 | 635 | 1572 | 334 | 588 | 1458 | 215 | 557 | 1453 |
| 21 | 486 | 628 | 1572 | 317 | 574 | 1454 | 199 | 542 | 1452 |
| 22 | 484 | 627 | 1572 | 301 | 561 | 1453 | 184 | 528 | 1452 |
| 23 | 484 | 627 | 1572 | 285 | 547 | 1452 | 170 | 515 | 1452 |
| 24 | 484 | 626 | 1571 | 269 | 534 | 1452 | 158 | 503 | 1452 |
| 25 | 484 | 626 | 1570 | 254 | 522 | 1452 | 146 | 493 | 1452 |
| 26 | 484 | 626 | 1568 | 240 | 511 | 1452 | 136 | 483 | 1452 |
| 27 | 484 | 626 | 1565 | 227 | 501 | 1452 | 126 | 475 | 1452 |
| 28 | 484 | 626 | 1559 | 216 | 492 | 1452 | 118 | 467 | 1452 |
| 29 | 484 | 626 | 1549 | 205 | 484 | 1452 | 110 | 460 | 1452 |
| 30 | 484 | 626 | 1531 | 195 | 476 | 1452 | 103 | 454 | 1452 |
| 31 | 484 | 626 | 1504 | 186 | 470 | 1452 | 96 | 448 | 1452 |
| 32 | 484 | 626 | 1475 | 177 | 464 | 1452 | 90 | 443 | 1452 |
| 33 | 484 | 626 | 1456 | 169 | 459 | 1452 | 85 | 439 | 1452 |
| 34 | 484 | 626 | 1452 | 162 | 454 | 1452 | 80 | 434 | 1452 |
| 35 | 484 | 626 | 1452 | 155 | 450 | 1452 | 75 | 431 | 1452 |
| 36 | 484 | 626 | 1452 | 149 | 446 | 1452 | 70 | 427 | 1452 |
| 37 | 484 | 626 | 1452 | 142 | 442 | 1452 | 66 | 425 | 1452 |
| 38 | 484 | 626 | 1452 | 137 | 438 | 1452 | 63 | 422 | 1452 |
| 39 | 484 | 626 | 1452 | 131 | 435 | 1452 | 59 | 420 | 1452 |
| 40 | 484 | 626 | 1452 | 126 | 432 | 1452 | 56 | 418 | 1452 |
| 41 | 484 | 626 | 1452 | 121 | 430 | 1452 | 53 | 416 | 1452 |
| 42 | 484 | 626 | 1452 | 117 | 427 | 1452 | 50 | 414 | 1452 |
| 43 | 484 | 626 | 1452 | 113 | 425 | 1452 | 47 | 413 | 1452 |

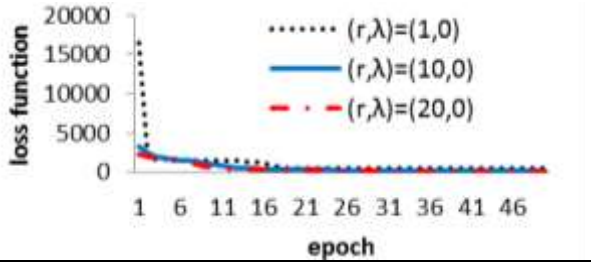
| epoch \ (r,λ) | (1,0) | (1,2) | (1,20) | (10,0) | (10,2) | (10,20) | (20,0) | (20,2) | (20,20) |
|---|---|---|---|---|---|---|---|---|---|
| 44 | 484 | 626 | 1452 | 109 | 423 | 1452 | 45 | 411 | 1452 |
| 45 | 484 | 626 | 1452 | 106 | 421 | 1452 | 43 | 410 | 1452 |
| 46 | 484 | 626 | 1452 | 119 | 419 | 1452 | 41 | 409 | 1452 |
| 47 | 484 | 626 | 1452 | 156 | 418 | 1452 | 39 | 409 | 1452 |
| 48 | 484 | 626 | 1452 | 100 | 420 | 1452 | 37 | 408 | 1452 |
| 49 | 484 | 626 | 1452 | 99 | 426 | 1452 | 35 | 407 | 1452 |
| 50 | 484 | 626 | 1452 | 98 | 417 | 1452 | 35 | 406 | 1452 |
| **local loss** | **484** | **489** | **976** | **98** | **130** | **976** | **35** | **98** | **976** |
| **reg. term** | **0** | **137** | **476** | **0** | **286** | **476** | **0** | **308** | **476** |

**sub comment:**

when the r is small ( as r =1, which means that the information is quite insufficient) or the λ is too hight ( as λ =20, which means that the punishment is too much), then the gradient descent (GD) process will easily be locked at the certain value ( threshold, not sure if it's the local minimum).

- Compare the cases under the same value of **r** or **λ**

**λ = 0**

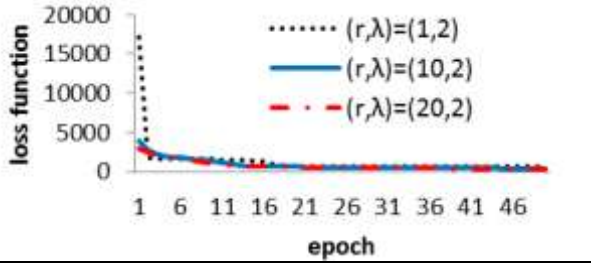



for the 50th epoch:

| (r,λ) | (1,0) | (10,0) | (20,0) |
|---|---|---|---|
| local loss | 484 | 98 | 35 |
| reg. term | 0 | 0 | 0 |

**sub comment:** when there is no punishment, the loss will drop as much as it can, and it will increase if the rank is higher ( the information is largely provided. )

**λ = 2**

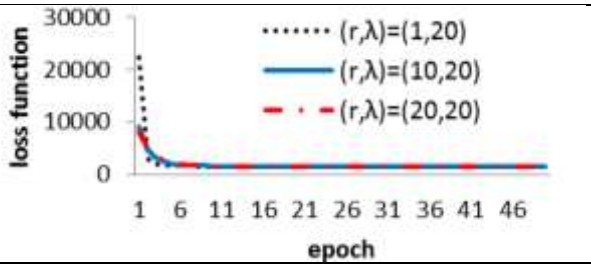



for the 50th epoch:

| (r,λ) | (1,2) | (10,2) | (20,2) |
|---|---|---|---|
| local loss | 489 | 130 | 98 |
| reg. term | 137 | 286 | 308 |

**sub comment:** when the λ is not large (as λ=2), comparing with the result in λ=1, it's easy to see that the droping speed is not much been affected.

**λ = 20**

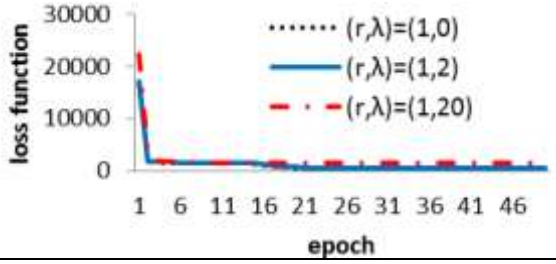



for the 50th epoch:

| (r,λ) | (1,20) | (10,20) | (20,20) |
|---|---|---|---|
| local loss | 976 | 976 | 976 |
| reg. term | 476 | 476 | 476 |

**sub comment:** when the λ is large enough ( as λ=20), even when the inforamtion is much (as r=20), the dropping process will stop at certain point, where the loss can still be less (comparing to the value in λ=1 or2.)

**r = 1**

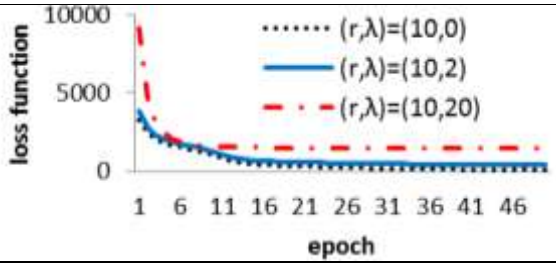



for the 50th epoch:

| (r,λ) | (1,0) | (1,2) | (1,20) |
|---|---|---|---|
| local loss | 484 | 489 | 976 |
| reg. term | 0 | 137 | 476 |

**sub comment:** when r is small (as r=1), the loss function is easily affected by λ.

**r = 10**

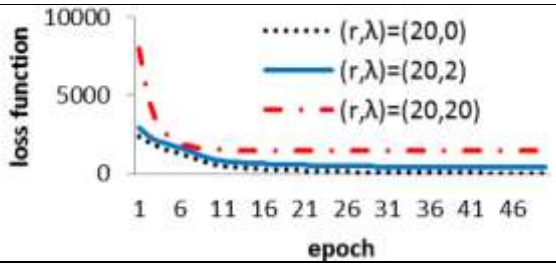



for the 50th epoch:

| (r,λ) | (10,0) | (10,2) | (10,20) |
|---|---|---|---|
| local loss | 98 | 130 | 976 |
| reg. term | 0 | 286 | 476 |

**sub comment:** when the r reach to a certain value (as r=10, th drop is easily to see, comparing to the case as in r=1. But still, when λ is too large (as λ=20), the drop process will be locked.

**r = 20**




for the 50th epoch:

| (r,λ) | (20,0) | (20,2) | (20,20) |
|---|---|---|---|
| local loss | 35 | 98 | 976 |
| reg. term | 0 | 308 | 476 |

**sub comment:** when the r reach to high value (as r=20, the loss can drop to a really samll amount. But still, when λ is too large (as λ=20), the drop process will be locked.

- result image:

The result images are presented below. It's easy to see that when punishment is weighted too much, though the local loss can go down as r increase, but it will shut at some point, causing the gradient process freeze. But while the punishment is much less weighted, then the reconstruction of the image can work better as we get more information.

| r \ λ | 0 | 2 | 20 |
|---|---|---|---|
| 1 |  |  |  |
| 10 |  |  |  |

| | | | |
|---|---|---|---|
| 20 |  |  |  |