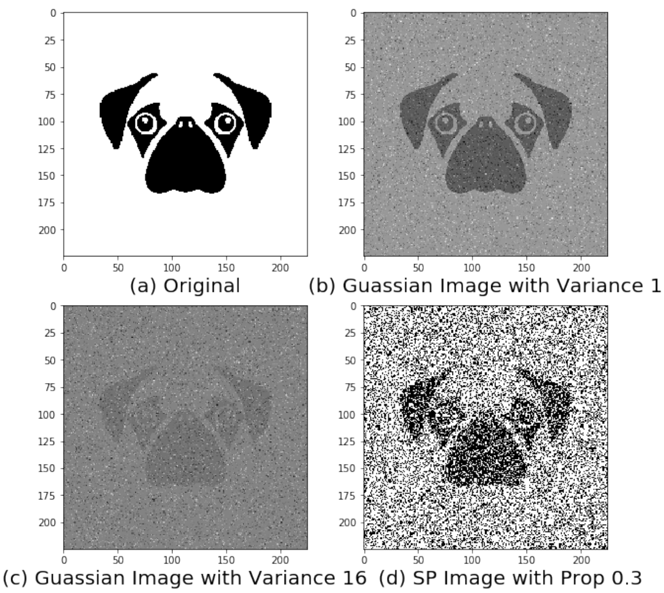
Inference

Shanglin Zou[1] and Junfan Huang[2]

[1] *98078*
[2] *78046*

January 2, 2022

Before doing questions, we first generate some noise to our original image. Figure 1 is our original and noise image.

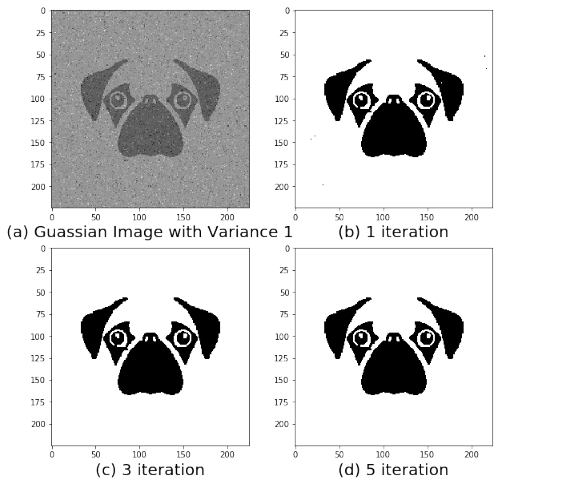


(a) Original (b) Guassian Image with Variance 1
(c) Guassian Image with Variance 16 (d) SP Image with Prop 0.3

Figure 1: *Origin and noise*

1 Question 1

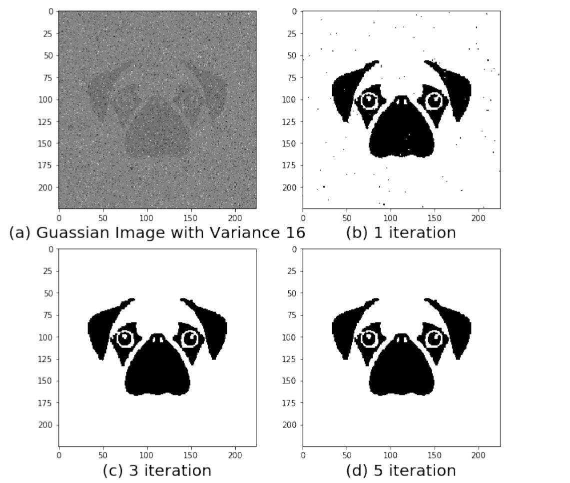
To de-noise Guassian Noise with variance 1, as shown in figure 2, After 3 iterations, we get descent result.



(a) Guassian Image with Variance 1 (b) 1 iteration
(c) 3 iteration (d) 5 iteration

Figure 2: *De-noise Guassian Noise with variance 1 by ICM*

To de-noise Guassian Noise with variance 16, as shown in figure 3, Also, after 3 iterations, we get descent result.



(a) Guassian Image with Variance 16 (b) 1 iteration
(c) 3 iteration (d) 5 iteration

Figure 3: *De-noise Guassian Noise with variance 16 by ICM*

To de-noise SP Noise with proportion 0.3, as shown in figure 4, after 5 iterations, we get result. However, it is not good. By increasing number of iterations, the noise decreases with new noises coming out.

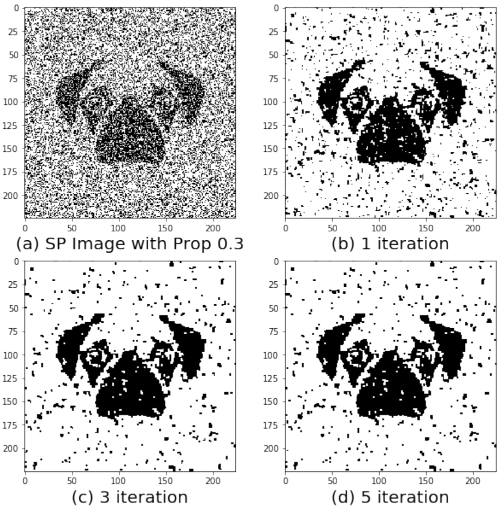


Figure 4: *De-noise SP noise by ICM*

2 Question 2

The Figure 5,6,7 shows the results by using Gibbs Sampling Ising Model. As we can see, the performance of Gibbs model is quite good for the Gaussian noise, however, for the SP noise, it is not really well.

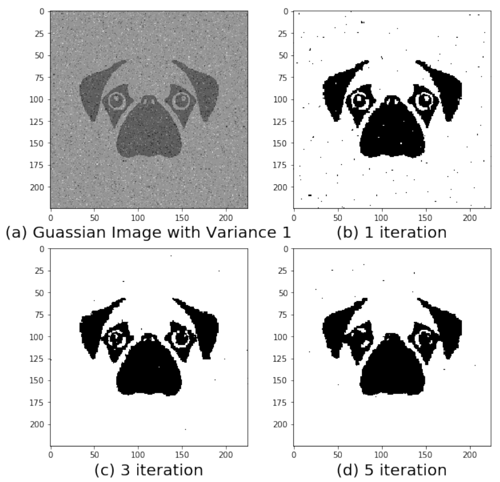


Figure 5: *De-noise Guassian Noise with variance 1 by Gibbs Sampling*

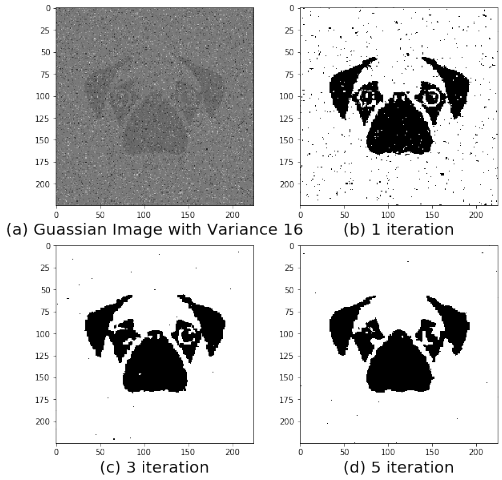


Figure 6: *De-noise Guassian Noise with variance 16 by Gibbs Sampling*

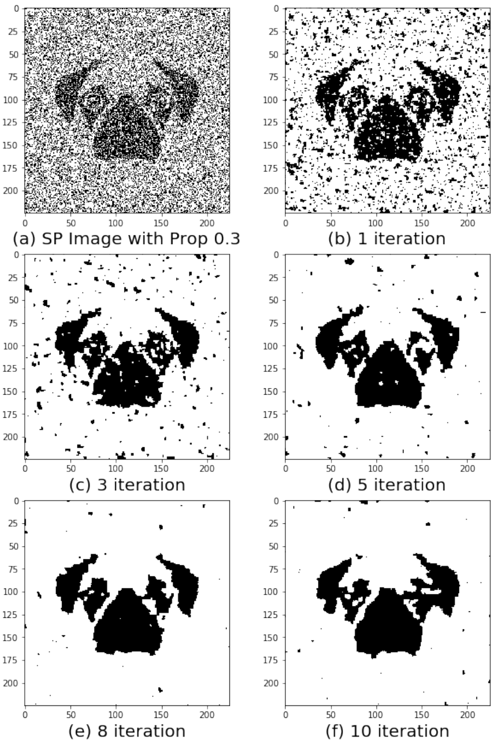


Figure 7: *De-noise SP noise by Gibbs Sampling*

3 Question 3

If we pick and updates a random node each iteration the results show as follows:

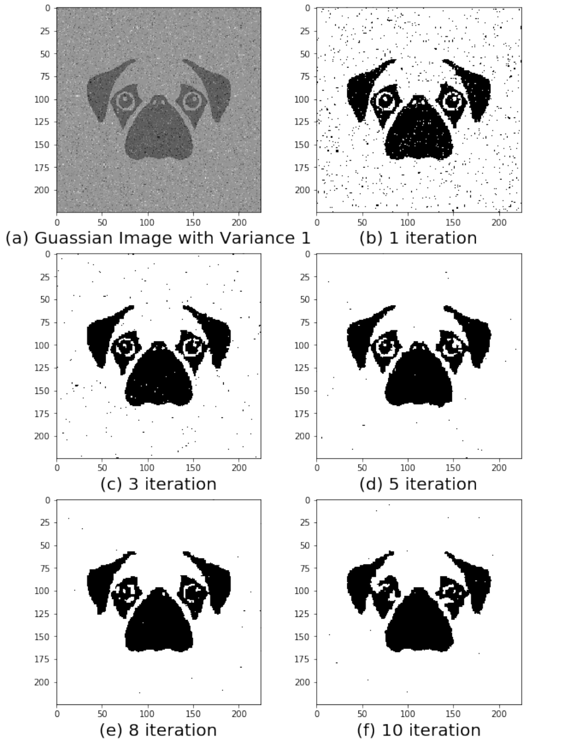


Figure 8: *De-noising Guassian noise with variance 1 by Gibbs Sampling using random nodes*

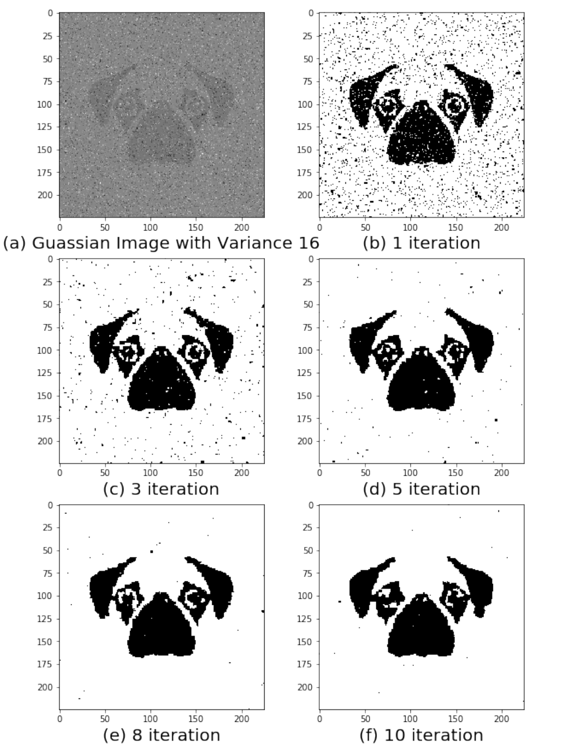


Figure 9: *De-noising Guassian noise with variance 16 by Gibbs Sampling using random nodes*

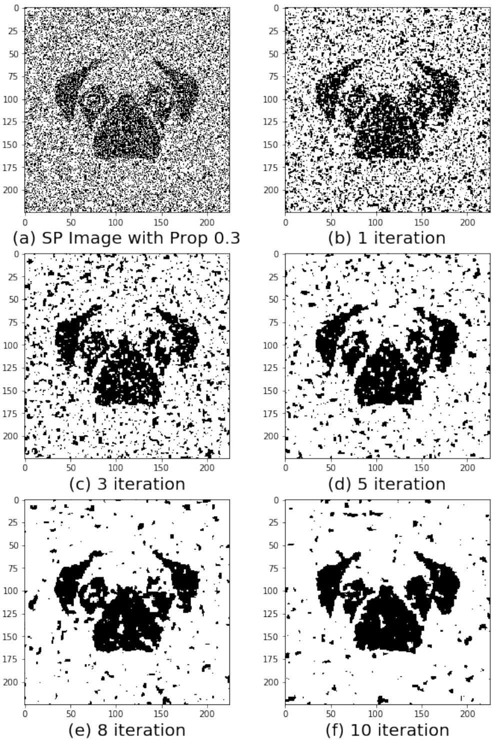


Figure 10: *De-noising SP noise by Gibbs Sampling using random nodes*

We need more iterations. We can compare Figure 6 and Figure 9. Clearly, after 5 iterations, the result of picking random nodes is worse than picking nodes by order. By coupon collector problem, we know that if we want to pick every node at least once, the expectation of number of times you pick is $nlogn$, where n is the number of nodes. Our image is 256 x 256, if we want to pick nodes by random, we need more iterations to get good results.

4 Question 4

By increasing the number of iterations, the noise will decrease, but it will usually stuck in a local optimum after which the noise won't decrease any more. In some cases, the image will get worse by increasing the number of iterations. For example, in Figure 10, the boundary between right eye and right ear is becoming more and more vague, and after 10 iterations, it seems to connect together.

5 Question 5

1. if $q(x) = p(x)$, then

$$KL(q(x)||p(x)) = \int q(x)log\frac{q(x)}{p(x)} = \int q(x)log1 = 0$$

same as $KL(p(x)||q(x))$, it is also equal to 0. So when $q(x) = p(x)$, the difference is 0.

2. if $p(x)$ is very small that close to 0, and q(x) is much bigger than 0, then

$$KL(q(x)||p(x)) = \int q(x)log\frac{q(x)}{p(x)} = \int q(x)log\infty = \infty$$

and, $\frac{p(x)}{q(x)}$ approaches to 0, $log\frac{p(x)}{q(x)}$ is $-\infty$,

$$KL(p(x)||q(x)) = \infty$$

The results make sense, $p(x) and q(x)$ are very different.

3. more generally,

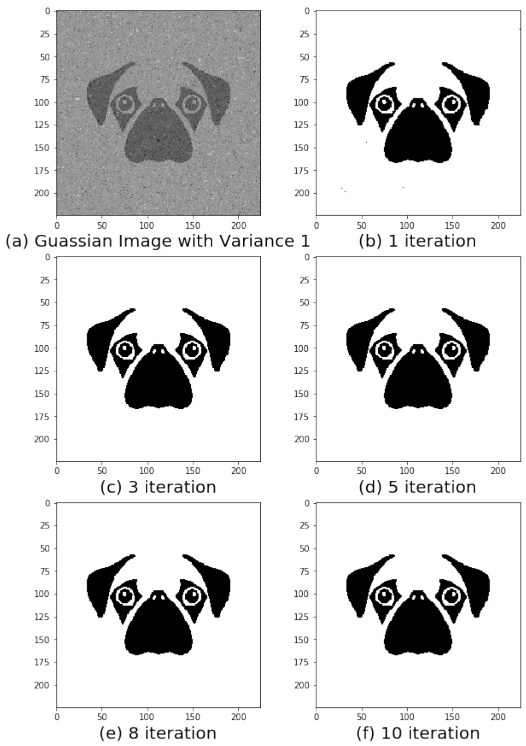
$$KL(p(x)||q(x)) = \int p(x)log\frac{p(x)}{q(x)} = -\int p(x)log\frac{q(x)}{x(x)}$$

since $p(x) \geq 0, q(x) \geq 0$,

$$KL(q(x)||p(x)) - KL(p(x)||q(x)) = \int (q(x)+p(x))log\frac{q(x)}{p(x)}$$

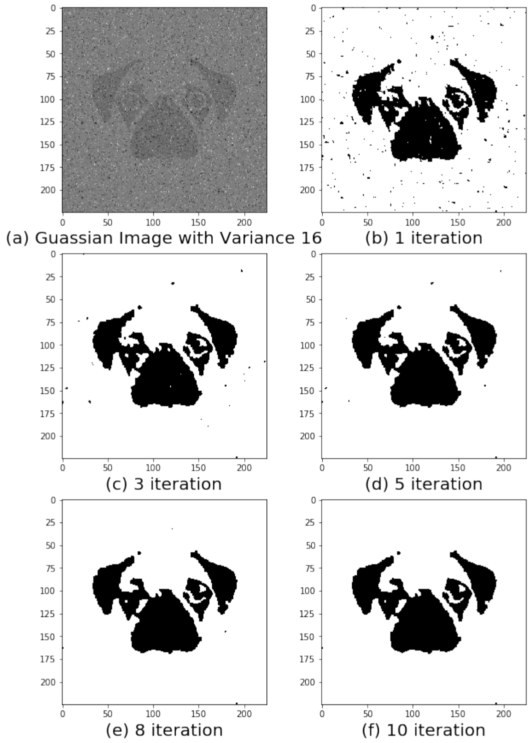
6 Question 6

In figure 11, the image (b) is what we have after de-noising Gaussian noise with variance 1 by 1 iteration, there are few noises. After 3 iterations, the image looks pretty nice. And more iterations seems like not change the image anymore.



(a) Guassian Image with Variance 1 (b) 1 iteration
(c) 3 iteration (d) 5 iteration
(e) 8 iteration (f) 10 iteration

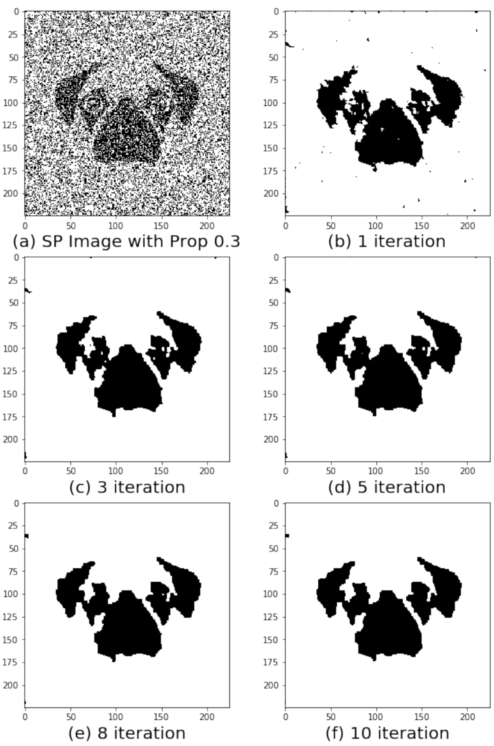
Figure 11: *De-noising Guassian noise with variance 1 by Variation Bayes Method*



(a) Guassian Image with Variance 16 (b) 1 iteration
(c) 3 iteration (d) 5 iteration
(e) 8 iteration (f) 10 iteration

Figure 12: *De-noising Guassian noise with variance 16 by Variation Bayes Method*

In figure 12, after de-noising, the result is worse than the noise with lower variance. The background is good, but for pug's eyes and nose, it is really bad.



(a) SP Image with Prop 0.3 (b) 1 iteration
(c) 3 iteration (d) 5 iteration
(e) 8 iteration (f) 10 iteration

Figure 13: *De-noising SP noise by Variation Bayes Method*

Figure 13 is what we have after de-noising SP noise by variation bayes method. It is still bad for pug's eyes and nose part and even worse.

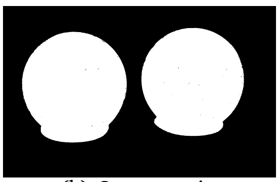
7 Question 7

Comparing to other techniques, Variation Bayes method is a good method to de-noise image, but the key is that we need to guess what is the likelihood function. If we guess a distribution that close to real one, then the effect of variation bayes will be good. Also this method is good to deal with lower variance. Comparing to ICM method, ICM can deal with different levels of Guassian noise and have good result, but for Variation Bayes, it is only good for lower variance. Comparing to Gibbs sampling, I think Variation Bayes has better ability to deal with background noise.

8 Question 8

For the image segmentation part, we firstly chose a simple image and marked the foreground and background by painter, then we located these two masks and got the RGB data from every pixel. In order to get the likelihood function, we use the k-means algorithm dividing the RGB data into 20 bins and get the foreground and background histograms respectively from which we could get the $p(y_i|x_i = 1)$ and $p(y_i|x_i = -1)$. With the likelihood function, it is easy to implement the algorithm we have used before. The results shows below after 10 iterations by using ICM model.

(a) Mask



(b) Segmentation

Figure 14: *ICM: Image segmentation*

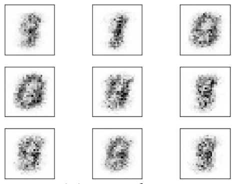
Another interesting thing is that it is able to predict the similar stuff without mask which is really good for applying to classification things, but in the shadow part it doesn't have a good performance.

9 Question 9

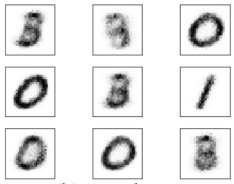
In general, the auto-encoder has two steps: one is encoding and the other one is decoding. Auto-encoder firstly encodes data into small dimensions we called sampled latent vector which likes a PCA process. Then by decoding the sampled latent vector, we generate the image back. For the "variational" auto-encoders, it is that mapping the input data onto a distribution instead of mapping any input to a fixed vector. In most VAEs we choose a multivariate and uncorrelated Gaussian distribution. Compared to the standard variational Bayes, the posterior in variational auto-encoder is generated by a neural network rather than a local optimization(in variational Bayes).

10 Question 10

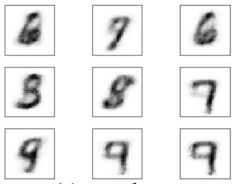
The training model learned the features in handwriting numbers and it is able to generate images which look like handwritten digits. The Figure 15 shows the resulting reconstruction of test image. As we can see, with the increasing number of training, the performance of decoder will be better. At the first 10 epochs, the decoder can only generate the images with few recognizable features. After around 30 epochs, the generated images become clearer through which we could recognize the majority of the numbers which means the model learned how a handwriting number looks like. In general, for the encoding step, it learned feature selection. For the decoding part, it learned how to generate or to create a handwriting number.



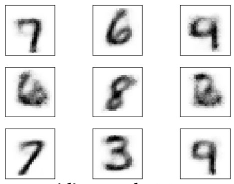
(a) epoch 0



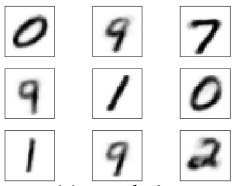
(b) epoch 10



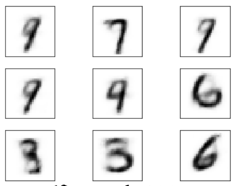
(c) epoch 20



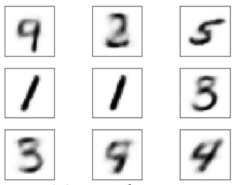
(d) epoch 30



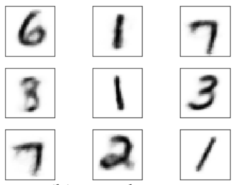
(e) epoch 60



(f) epoch 100



(g) epoch 150



(h) epoch 199

Figure 15: *Training results*