HOMEWORK 5

>>Yang Gao<<
>>9083410275<<

Instructions:

- Please submit your answers in a single pdf file and your code in a zip file. pdf preferably made using latex. No need to submit latex code. See piazza post @114 for some recommendations on how to write the answers.
- Submit code for programming exercises. Though we provide a base code with python (jupyter notebook), you can use any programming language you like as long as you use the same model and dataset.
- Submit all the material on time.

1 Implementation: GAN (40 pts)

In this part, you are expected to implement GAN with MNIST dataset. We have provided a base jupyter notebook (gan-base.ipynb) for you to start with, which provides a model setup and training configurations to train GAN with MNIST dataset.

(a) Implement training loop and report learning curves and generated images in epoch 1, 50, 100. Note that drawing learning curves and visualization of images are already implemented in provided jupyter notebook. (20 pts)

Procedure 1 Training GAN, modified from Goodfellow et al. (2014)

```
Input: $m$: real data batch size, $n_z$: fake data batch size
Output: Discriminator $D$, Generator $G$
  for number of training iterations do
      # Training discriminator
      Sample minibatch of $n_z$ noise samples $\{z^{(1)}, z^{(2)}, \cdots, z^{(n_z)}\}$ from noise prior $p_g(z)$
      Sample minibatch of $\{x^{(1)}, x^{(2)}, \cdots, x^{(m)}\}$
      Update the discriminator by ascending its stochastic gradient:
          $$\nabla_{\theta_d}\Big(\frac{1}{m}\sum_{i=1}^{m}\log D(x^{(i)}) + \frac{1}{n_z}\sum_{i=1}^{n_z}\log(1 - D(G(z^{(i)})))\Big)$$
      # Training generator
      Sample minibatch of $n_z$ noise samples $\{z^{(1)}, z^{(2)}, \cdots, z^{(n_z)}\}$ from noise prior $p_g(z)$
      Update the generator by ascending its stochastic gradient:
          $$\nabla_{\theta_g}\frac{1}{n_z}\sum_{i=1}^{n_z}\log D(G(z^{(i)})))$$
  end for
  # The gradient-based updates can use any standard gradient-based learning rule. In the base code, we are using Adam optimizer (Kingma and Ba, 2014)
```

Expected results are as follows.

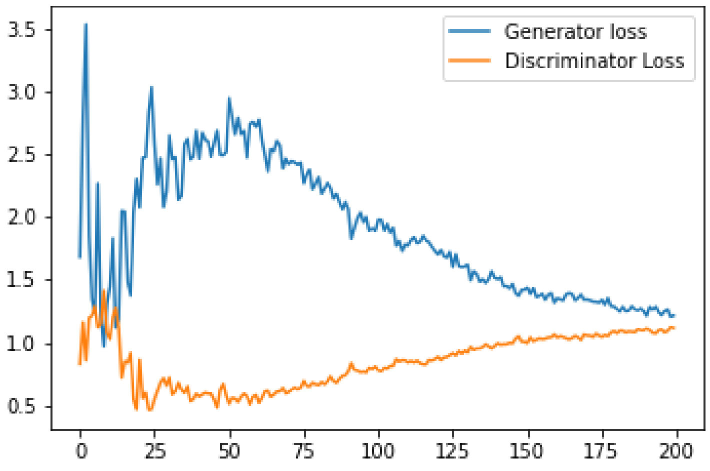
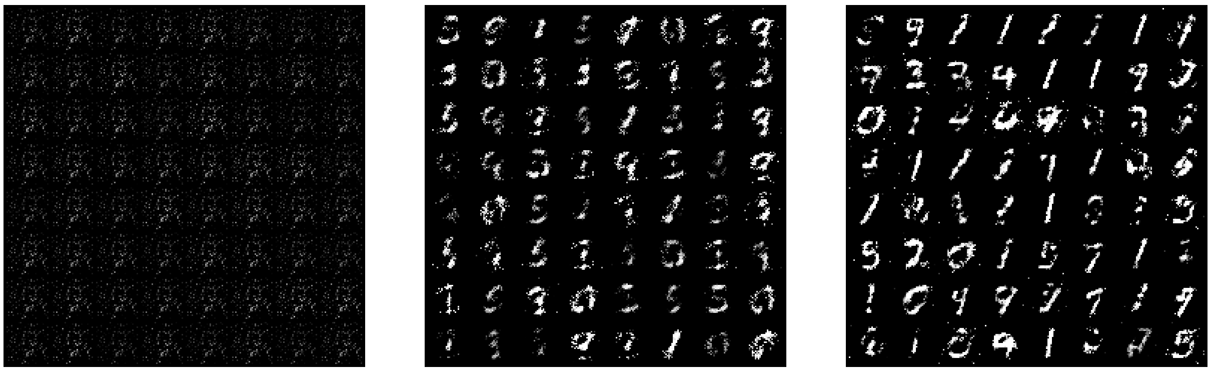


Figure 1: Learning curve



(a) epoch 1 (b) epoch 50 (c) epoch 100

Figure 2: Generated images by G

Solution:

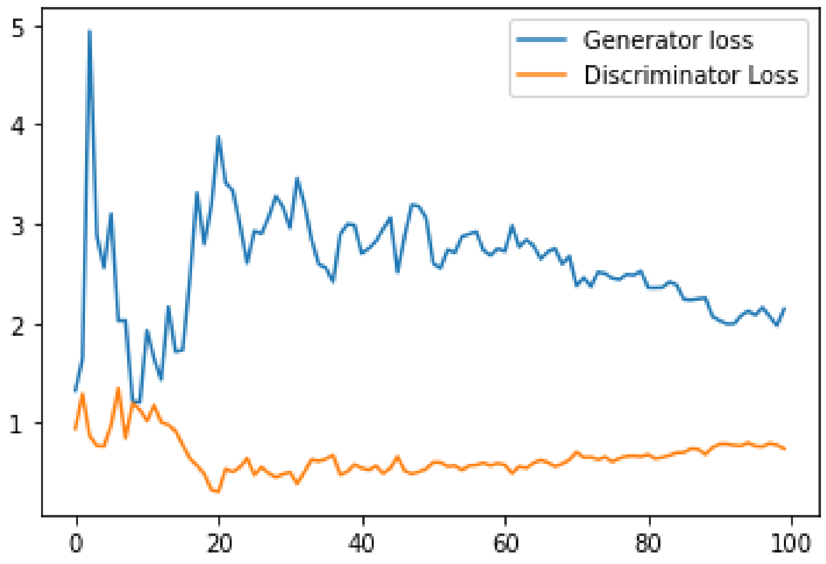
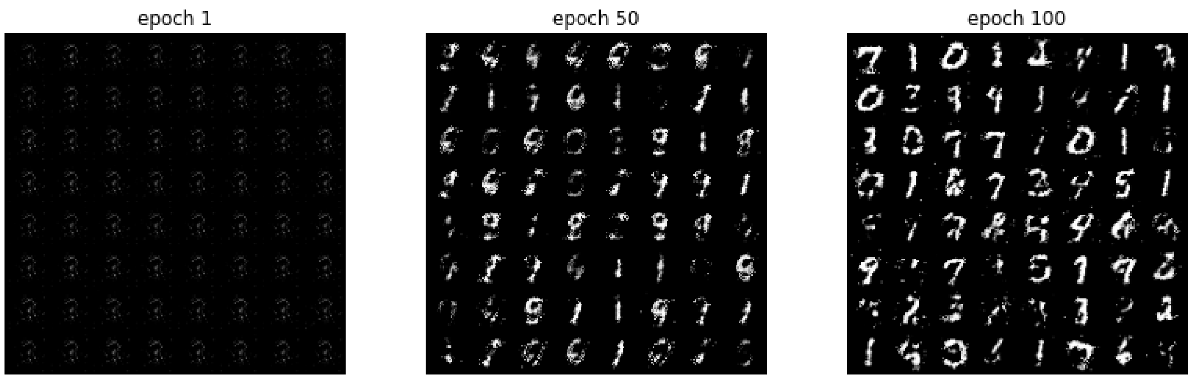


Figure 3: Learning curve



(a) epoch 1 (b) epoch 50 (c) epoch 100

Figure 4: Generated images by G

(b) Replace the generator update rule as the original one in the slide,
"Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{n_z} \sum_{i=1}^{n_z} \log(1 - D(G(z^{(i)})))$$

" , and report learning curves and generated images in epoch 1, 50, 100. Compare the result with (a). Note that it may not work. If training does not work, explain why it doesn't work. (10 pts)

No it doesn't work, as when D(Gz)=1, the loss becomes 0. Solution:

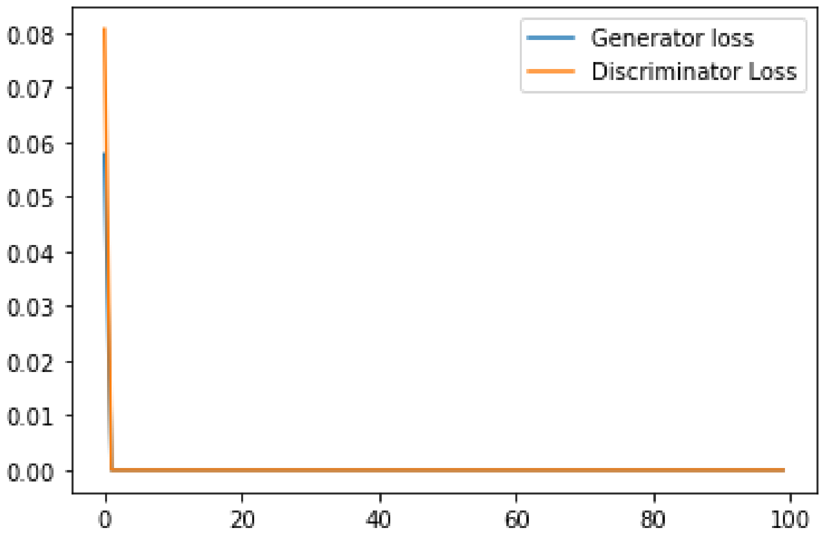
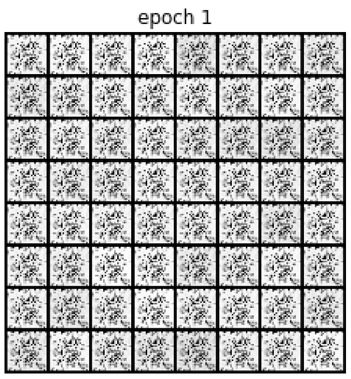
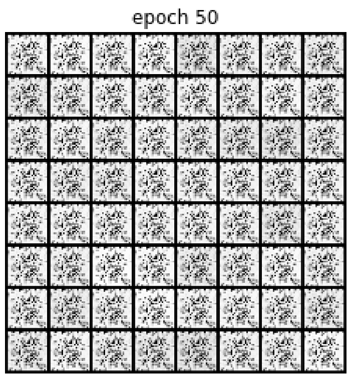


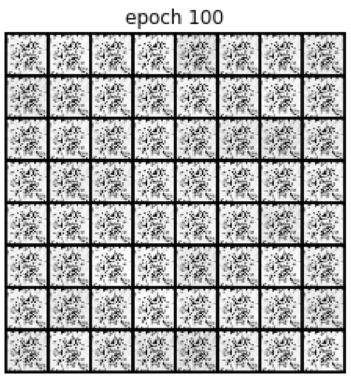
Figure 5: Learning curve



(a) epoch 1



(b) epoch 50



(c) epoch 100

Figure 6: Generated images by G

(c) Except the method that we used in (a), how can we improve training for GAN? Implement that and report learning curves and generated images in epoch 1, 50, 100. (10 pts)

We can use batch normalization, which makes training faster and more stable.

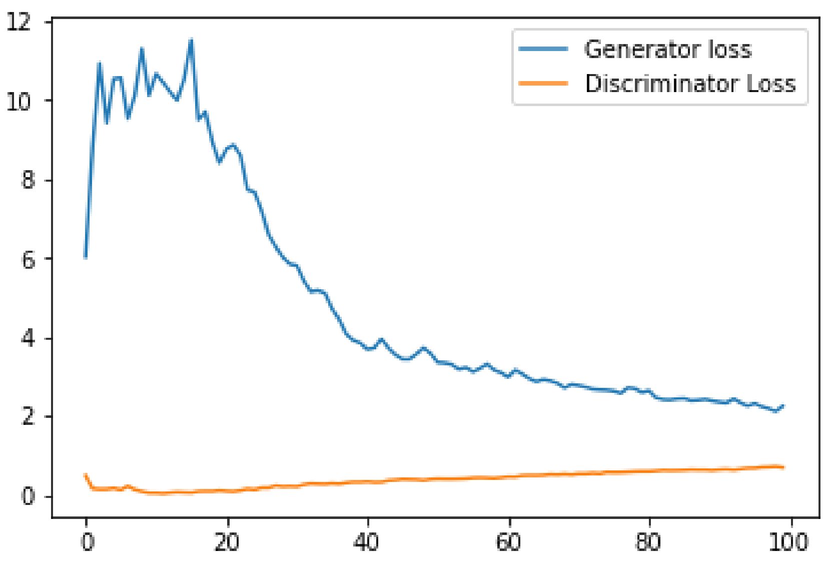
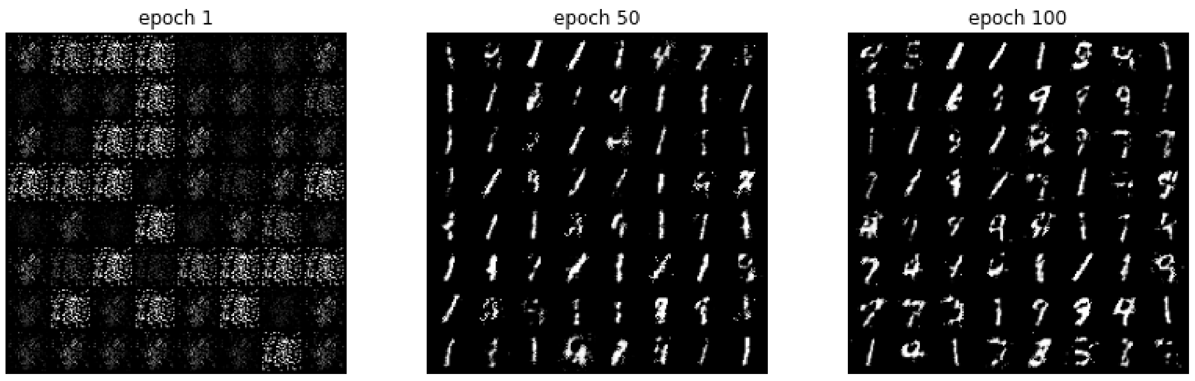


Figure 7: Learning curve



(a) epoch 1 (b) epoch 50 (c) epoch 100

Figure 8: Generated images by G

2 Ridge regression [15 pts]

Derive the closed-form solution in matrix form for the ridge regression problem:

$$\min_{\beta} \left(\frac{1}{n} \sum_{i=1}^{n} (z_i^\top \beta - y_i)^2 \right) + \lambda \|\beta\|_A^2$$

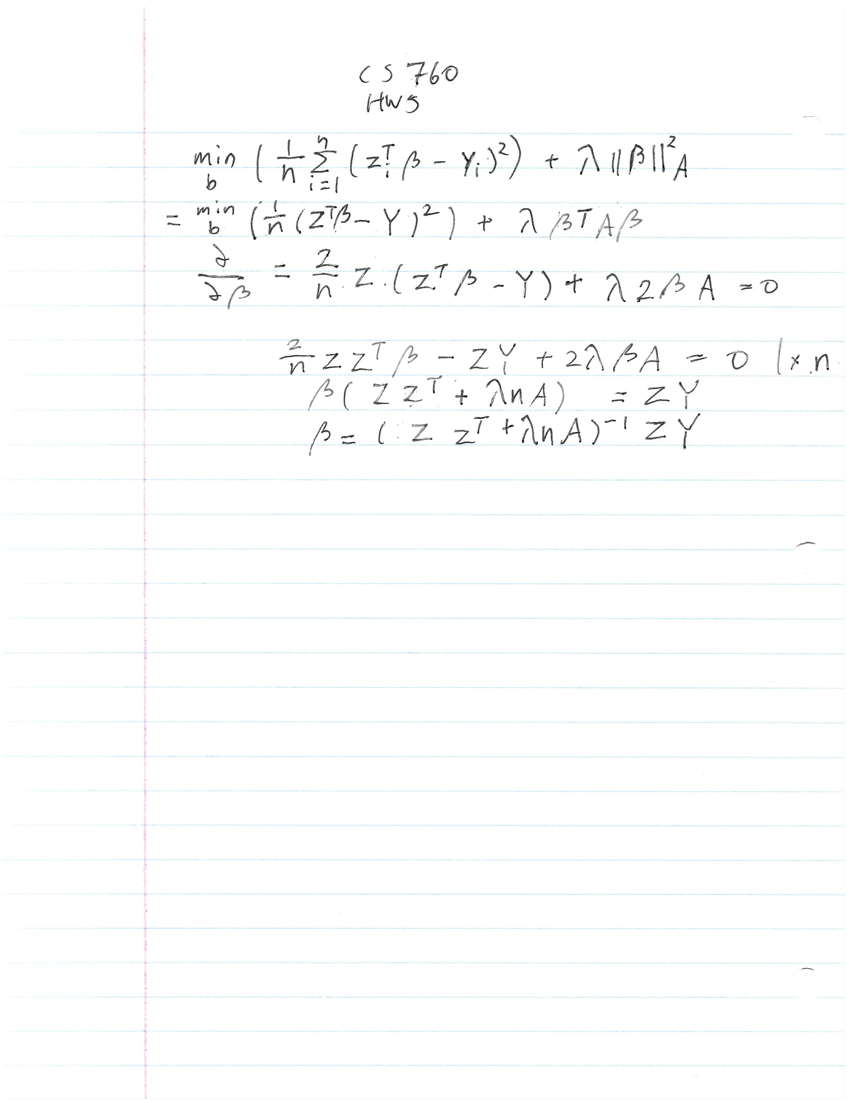
where

$$\|\beta\|_A^2 := \beta^\top A \beta$$

and

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

This A matrix has the effect of NOT regularizing the bias β_0, which is standard practice in ridge regression. Note: Derive the closed-form solution, do not blindly copy lecture notes.



CS 760
HW5

$$\min_b \left(\frac{1}{n} \sum_{i=1}^{n} (z_i^T \beta - y_i)^2 \right) + \lambda \|\beta\|_A^2$$

$$= \min_b \left(\frac{1}{n} (Z^T\beta - Y)^2 \right) + \lambda \beta^T A \beta$$

$$\frac{\partial}{\partial \beta} = \frac{2}{n} Z \cdot (Z^T \beta - Y) + \lambda 2 \beta A = 0$$

$$\frac{2}{n} Z Z^T \beta - Z Y + 2\lambda \beta A = 0 \quad | \times n$$

$$\beta (Z Z^T + \lambda n A) = Z Y$$

$$\beta = (Z Z^T + \lambda n A)^{-1} Z Y$$

Figure 9: Q2

3 Review the change of variable in probability density function [25 pts]

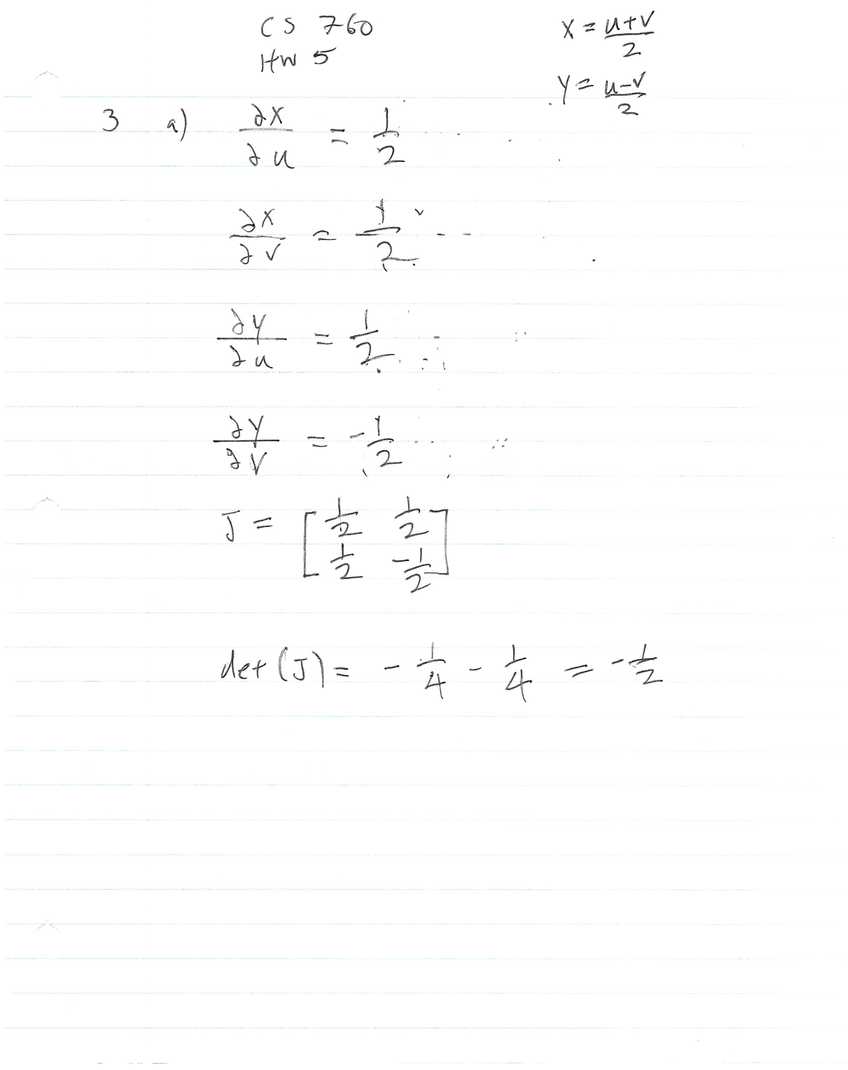
In Flow based generative model, we have seen $p_\theta(x) = p(f_\theta(x))|\frac{\partial f_\theta(x)}{\partial x}|$. As a hands-on (fixed parameter) example, consider the following setting.

Let X and Y be independent, standard normal random variables. Consider the transformation $U = X + Y$ and $V = X - Y$. In the notation used above, $U = g_1(X, Y)$ where $g_1(X, Y)$ where $g_1(x, y) = x + y$ and $V = g_2(X, Y)$ where $g_2(x, y) = x - y$. The joint pdf of X and Y is $f_{X,Y} = (2\pi)^{-1} exp(-x^2/2) exp(-y^2/2), -\infty < x < \infty, -\infty < y < \infty$. Then, we can determine u, v values by x, y, i.e. $\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$.

(a) (5 pts) Compute Jacobian matrix

$$J = \begin{bmatrix} \frac{\partial x}{\partial u} & \frac{\partial x}{\partial v} \\ \frac{\partial y}{\partial u} & \frac{\partial y}{\partial v} \end{bmatrix}$$

(5 pts)



CS 760
HW 5

$x = \frac{u+v}{2}$

$y = \frac{u-v}{2}$

3 a) $\frac{\partial x}{\partial u} = \frac{1}{2}$

$\frac{\partial x}{\partial v} = \frac{1}{2}$

$\frac{\partial y}{\partial u} = \frac{1}{2}$

$\frac{\partial y}{\partial v} = -\frac{1}{2}$

$$J = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} \end{bmatrix}$$

$$\det(J) = -\frac{1}{4} - \frac{1}{4} = -\frac{1}{2}$$

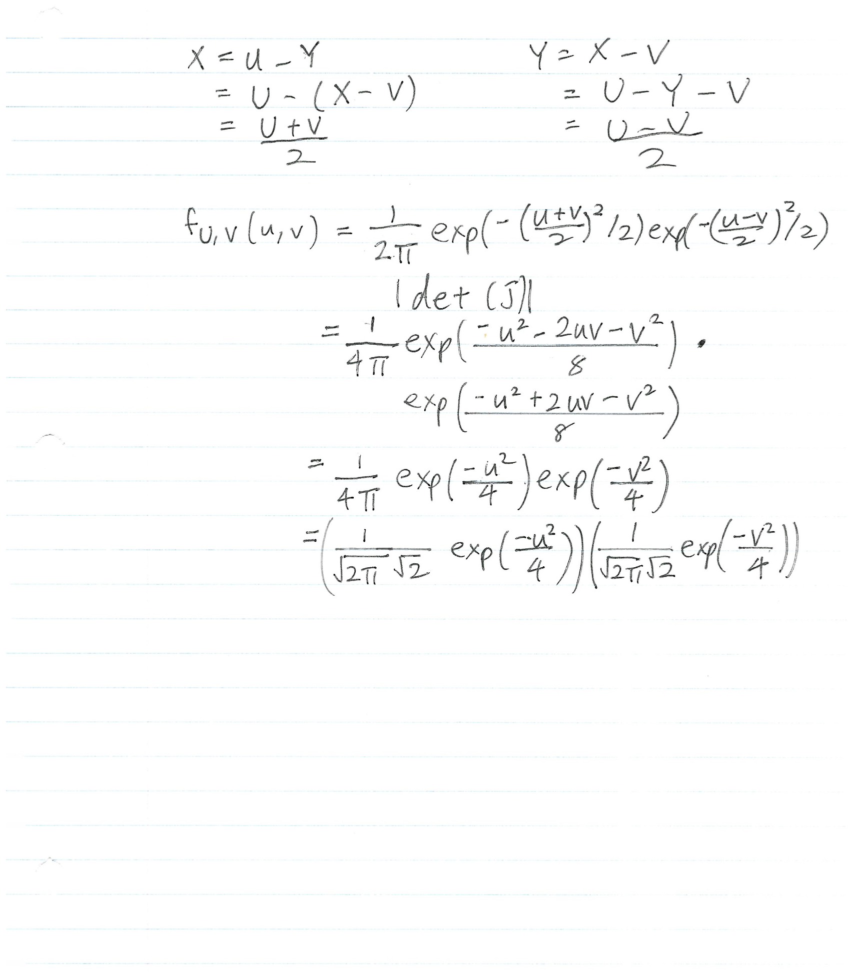
Figure 10: Q3a

(b) (Forward) Show that the joint pdf of U, V is

$$f_{U,V}(u,v) = \big(\frac{1}{\sqrt{2\pi}\sqrt{2}} exp(-u^2/4)\big)\big(\frac{1}{\sqrt{2\pi}\sqrt{2}} exp(-v^2/4)\big)$$

(10 pts)

(Hint: $f_{U,V}(u,v) = f_{X,Y}(?,?)|det(J)|$)



$X = U - Y$
$= U - (X - V)$
$= \frac{U+V}{2}$

$Y = X - V$
$= U - Y - V$
$= \frac{U-V}{2}$

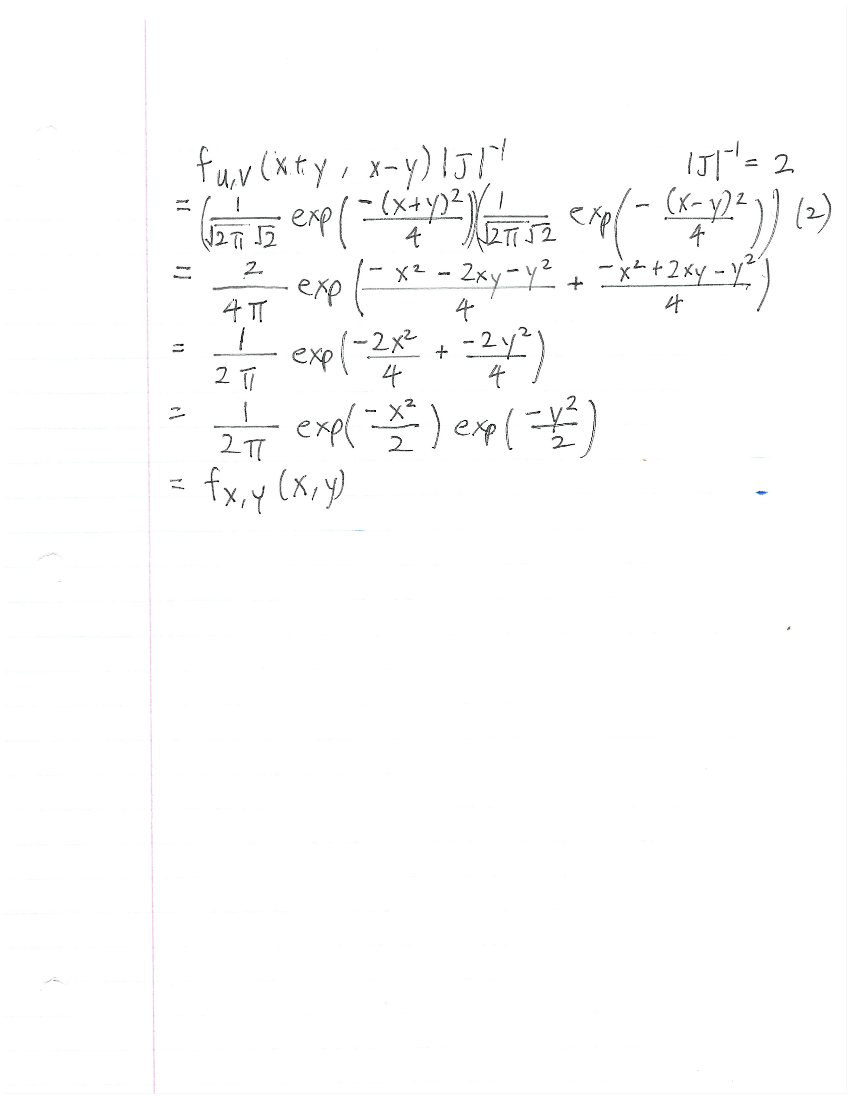
$f_{U,V}(u,v) = \frac{1}{2\pi} exp\left(-\left(\frac{u+v}{2}\right)^2/2\right) exp\left(-\left(\frac{u-v}{2}\right)^2/2\right)$
$|det(J)|$
$= \frac{1}{4\pi} exp\left(\frac{-u^2-2uv-v^2}{8}\right) \cdot$
$exp\left(\frac{-u^2+2uv-v^2}{8}\right)$
$= \frac{1}{4\pi} exp\left(\frac{-u^2}{4}\right) exp\left(\frac{-v^2}{4}\right)$
$= \left(\frac{1}{\sqrt{2\pi}\sqrt{2}} exp\left(\frac{-u^2}{4}\right)\right)\left(\frac{1}{\sqrt{2\pi}\sqrt{2}} exp\left(\frac{-v^2}{4}\right)\right)$

Figure 11: Q3b

(c) (Inverse) Check whether the following equation holds or not.

$$f_{X,Y}(x,y) = f_{U,V}(x+y, x-y)|det(J)^{-1}|$$

(10 pts)



$$f_{u,v}(x+y,\ x-y)\,|J|^{-1} \qquad |J|^{-1} = 2$$

$$= \left(\frac{1}{\sqrt{2\pi}\sqrt{2}} \exp\left(\frac{-(x+y)^2}{4}\right)\right)\left(\frac{1}{\sqrt{2\pi}\sqrt{2}} \exp\left(-\frac{(x-y)^2}{4}\right)\right)(2)$$

$$= \frac{2}{4\pi} \exp\left(\frac{-x^2-2xy-y^2}{4} + \frac{-x^2+2xy-y^2}{4}\right)$$

$$= \frac{1}{2\pi} \exp\left(\frac{-2x^2}{4} + \frac{-2y^2}{4}\right)$$

$$= \frac{1}{2\pi} \exp\left(\frac{-x^2}{2}\right) \exp\left(\frac{-y^2}{2}\right)$$

$$= f_{X,Y}(x,y)$$

Figure 12: Q3c

4 Directed Graphical Model [20 points]

Consider the directed graphical model (aka Bayesian network) in Figure 13.

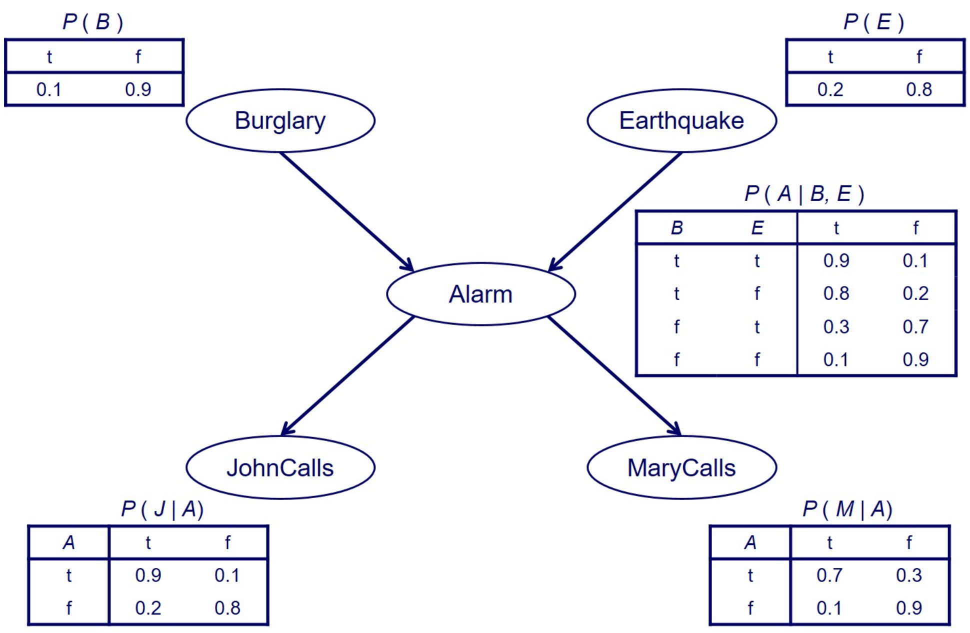


Figure 13: A Bayesian Network example.

Compute $P(B = t \mid E = f, J = t, M = t)$ and $P(B = t \mid E = t, J = t, M = t)$. These are the conditional probabilities of a burglar in your house (yikes!) when both of your neighbors John and Mary call you and say they hear an alarm in your house, but without or with an earthquake also going on in that area (what a busy day), respectively.

$P(B = t|E = f, J = t, M = t) = \frac{P(B=t \cap E=f, J=t, M=t)}{P(E=f, J=t, M=t)}$
Numerator: $P(B = t \cap E = f, J = t, M = t) = P(E = f, M = t, J = t, B = t, A = t) + P(E = f, M = t, J = t, B = t, A = f)$ =0.8*0.8*0.7*0.1*0.9 + 0.8*0.2*0.1*0.1*0.2 = 0.04064
Denominator: $P(E = f, J = t, M = t) = P(E = f, M = t, J = t, B = t, A = t) + P(E = f, M = t, J = t, B = f, A = t) + P(E = f, M = t, J = t, B = t, A = f) + P(E = f, M = t, J = t, B = f, A = f)$
=0.8*0.8*0.7*0.1*0.9 + 0.8*0.9*0.7*0.9*0.1+0.8*0.2*0.1*0.1*0.2+0.8*0.2*0.1*0.9*0.9=0.09896

$P(B = t|E = f, J = t, M = t) = 0.04064/0.09896 = 0.41$

$P(B = t|E = t, J = t, M = t) = \frac{P(B=t \cap E=t, J=t, M=t)}{P(E=t, J=t, M=t)}$
Numerator: $P(B = t \cap E = t, J = t, M = t) = P(E = t, M = t, J = t, B = t, A = t) + P(E = t, M = t, J = t, B = t, A = f)$ =0.2*0.7*0.9*0.1*0.9 + 0.2*0.1*0.2*0.1*0.1 = 0.01138
Denominator: $P(E = t, J = t, M = t) = P(E = t, M = t, J = t, B = t, A = t) + P(E = t, M = t, J = t, B = f, A = t) + P(E = t, M = t, J = t, B = t, A = f) + P(E = t, M = t, J = t, B = f, A = f)$
=0.2*0.7*0.9*0.1*0.9+0.2*0.7*0.9*0.9*0.3+0.2*0.1*0.2*0.1*0.1+0.2*0.1*0.2*0.9*0.7=0.04792

$P(B = t|E = t, J = t, M = t) = 0.01138/0.04792 = 0.24$

References

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. *Advances in neural information processing systems*, 27.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.