# CV202, HW2

Dan Tsir Cohen, ID 208538314
Nir Ben Zikri, ID 205793862

## Problem 1

The condition is that $A$ should be an invertible matrix. Then $f^{-1}: x \to A^{-1}x$. ■

## Problem 2

For $f$ to be a linear map it must satisfy:

$$f(\delta x) = \delta f(x), \forall \delta \in \mathbb{R}$$

In our case, considering $f(x) = ax + b, b \neq 0$:

$$f(\delta x) = a\delta x + b$$

$$\delta f(x) = \delta(ax + b) = a\delta x + \delta b$$

If we want $f$ to be linear, we must have:

$$b = \delta b, \forall \delta \in \mathbb{R}$$

This is only true if $b = 0$, which is not the case, therefore $f$ is not a linear map. ■

## Problem 3

Let's observe our $f$:

$$f(x) = Ax + b$$

For $f$ to be invertible, we need to find $g : \mathbb{R}^m \to \mathbb{R}^n$ such that $\forall x$:

$$\forall x \in \mathbb{R}^n, y \in \mathbb{R}^m \quad : \quad g\big(f(x)\big) = x \; and \; f\big(g(y)\big) = y$$

Choosing $g(y) = A^{-1}y - A^{-1}b$:

$$g\big(f(x)\big) = g(Ax + b) = A^{-1}(Ax + b) - A^{-1}b = x$$

$$f\big(g(y)\big) = f(A^{-1}y - A^{-1}b) = A(A^{-1}y - A^{-1}b) + b = y$$

As we can see, this is the only possible inverse for $b$.

But its existence assumes two major conditions! $A$ must be invertible (not only right/left invertible), which also means it must be a square matrix, meaning $m = n$.

So in conclusion $f$ can be invertible only on the conditions that $m = n$ and $A$ is invertible. ■

## Problem 4

$g: x \to A_2 x + b_2$
$f: x \to A_1 x + b_1$

$h = g \; o \; f = g\big(f(x)\big) = A_2(A_1 x + b_1) + b_2 = A_2 A_1 x + A_2 b_1 + b_2$

Define $A_3 = A_2 A_1,\ A_3 \in \mathbb{R}^{kxn}$, and $b_3 = A_2 b_1 + b_2,\ b_3 \in \mathbb{R}^k$.

Therefore $h$ is affine $h: x \to A_3 x + b_3$. ∎

## Problem 5

We have $f: \mathbb{R}^n \to \mathbb{R}^n\ and\ g: \mathbb{R}^n \to \mathbb{R}^n$ invertible affine maps.

This means:

$$f(x) = A_f x + b_f,\ \ g(x) = A_g x + b_g$$

Such that $A_f, A_g$ are invertible (we've shown that in problem 3).

Now by definition $h$ is an affine map (as we've shown before), of the following structure:

$$h(x) = g\big(f(x)\big) = g\big(A_f x + b_f\big) = A_g\big(A_f x + b_f\big) + b_g = A_g A_f x + A_g b_f + b_g$$

And knowing that $A_f^{-1}, A_g^{-1}$ exist, we can see that $A_g A_f$ is invertible (by $A_f^{-1} A_g^{-1}$), therefore we can conclude from what we've proven in problem 3 that $h$ is invertible affine map. ∎

## Problem 6

We will show that G1, G2 and G3 definitions are satisfied:

$GL(n) = \{Q | Q \in R^{nxn}, \det Q\ \neq 0\}$

<u>*G1:*</u> $I_{nxn}\ \in R^{nxn}\ and\ \det(I_{nxn}) \neq 0$

<u>*G2:*</u> $A, B \in GL(n)$

$$A_{nxn} B_{nxn}\ \in R^{nxn}$$

$$\det(A) \neq 0 \text{ and } \det(B) \neq 0$$

$$\det(AB) = \det(A)\det(B) \neq 0$$

<u>*G3:*</u> *define* $A \in GL(n)$

$$\det(A) \neq 0, \qquad therefore\ A\ is\ invertible$$

$$A^{-1} \in R^{nxn}\ and\ \det(A^{-1}) \neq 0$$

## Problem 7

Define:

$A = \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix}, B = \begin{bmatrix} 1 & 0 \\ 2 & 1 \end{bmatrix}$

$A, B\ \in R^{nxn}\ , \det(A) \neq 0\ and\ \det(B) \neq 0, therefore\ A, B\ \in GL(2), but:$

$$AB = \begin{bmatrix} 5 & 1 \\ 2 & 3 \end{bmatrix} \neq \begin{bmatrix} 1 & 2 \\ 2 & 5 \end{bmatrix} = BA$$

*We found 2 matrices that belongs to GL(2) so that the Abelian definition is not satisfied, therefore GL(n) is not Abelian.*

## Problem 8

We will show that G1, G2 and G3 definitions are satisfied:

$GL(n)_{+} = \{Q | Q \in R^{nxn}, \det Q > 0\}$

<u>G1:</u> $I_{nxn} \in R^{nxn}, so \ \det(I_{nxn}) = 1 > 0$

<u>G2:</u> $A, B \in GL(n)_{+}$

$$A_{nxn}B_{nxn} \in R^{nxn}$$

$$\det(A) > 0 \text{ and } \det(B) > 0$$

$$\det(AB) = \det(A)\det(B) > 0$$

<u>G3:</u> $A \in GL(n)$

$$\det(A) > 0, also \det(A) \neq 0, \quad therefore\ A\ is\ invertible$$

$$\det(AA^{-1}) = \det(A)\det(A^{-1}) = 1 \quad => \det(A^{-1}) = \frac{1}{\det(A)}$$

$$A^{-1} \in R^{nxn}\ and \det(A^{-1}) = \frac{1}{\det(A)} > 0$$

## Problem 9

Lets take a look at G2 definition:

$$A, B \in GL_{-}(n)$$

$$A_{nxn}B_{nxn} \in R^{nxn}$$

$$\det(A) < 0 \text{ and } \det(B) < 0$$

$$\det(AB) = \det(A)\det(B) > 0$$

$AB\ does\ not\ belong\ to\ GL_{-}(n), therefore\ GL(n)_{-}\ is\ not\ a\ matrix\ group$

## Problem 10

$A\ and\ B\ are\ symetric\ matrices\ A = A^T, B = B^T$

$(AB)^T = B^T A^T = BA$

$AB = (AB)^T = BA, satisfied\ only\ when$:

- $A = B$
- $A = cI\ or\ B = cI$
- $A\ and\ B\ diagnal\ matrices$

$Those\ terms\ usually\ are\ not\ satisfied\ so\ \boldsymbol{usually}\ AB \neq (AB)^T$

## Problem 11

We will show that G1, G2 and G3 definitions are satisfied:

$US(n) = \{Q | Q \in SI_{nxn}, S \in R_{>0}\}$

$\underline{G1:}$ $1I_{nxn} \in R^{nxn}$ $and$ $S = 1 > 0$ so $I_{n\times n} \in US(n)$

$\underline{G2:}$ $A, B \in US(n)$

$$A_{nxn}B_{nxn} \in R^{nxn}$$

$$A = S_1I, B = S_2I$$

$$S_1, S_2 > 0$$

$$AB = S_1IS_2I = S_1S_2I$$

$$S_1S_2 > 0$$

$$AB \in US(n)$$

$\underline{G3:}$ $A \in GL(n)$

define $A = cI, A^{-1} = \frac{1}{c}I$ $, so\ A^{-1} exists$

$$c' = \frac{1}{c} > 0 \ => A^{-1} = c'I$$

$therefore\ A^{-1} \in US(n)$

## Problem 12

$Define\ A, B \in US(n)$

$A = S_1I, B = S_2I$

$AB = S_1IS_2I =^{S_1 and\ S_2\ are\ scalars} S_2IS_1I = BA$

## Problem 13

We can see that the matrix $0_{nxn}$ satisfies the following:

- for all $X \in R^{nxn}, X + 0_{nxn} = X$
- for all $X \in R^{nxn}, X \cdot 0_{nxn} = 0_{nxn}$

Therefore the matrix $0_{nxn}$ is the zero element.

## Problem 14

A linear subspace of a linear space must contain the 0 element.

We know that if A belongs to a matrix group there exists $A^{-1}$.

However the $0_{n\times n}$ is clearly non-invertible, therefore can't be in any matrix group.

Therefore any matrix group cannot be a linear subspace of $\mathbb{R}^{n\times n}$.

## Problem 15

We will show that G1, G2 and G3 definitions are satisfied:

$\underline{G1:}$ $I_{n+1xn+1} = \begin{bmatrix} I_{nxn} & 0_{1xn} \\ 0 & 1 \end{bmatrix}, A = I_{nxn}, b = 0_{1xn}.$

$$A = I_{nxn} \quad => \quad \det(A) \neq 0$$

$$\Longrightarrow I_{(n+1)\times(n+1)} \in (the\ affine\ group)$$

<u>G2:</u> $let\ \ matrices\ M, N \in affine\ group$

$Define: M = \begin{bmatrix} A & b_m \\ 0 & 1 \end{bmatrix}, N = \begin{bmatrix} B & b_n \\ 0 & 1 \end{bmatrix}$

$$MN = \begin{bmatrix} AB & Ab_n + b_m \\ 0 & 1 \end{bmatrix}$$

We know $\det(A)$ , $\det(B) \neq 0$ as $M, N$ are in the affine group.

So: $\det(AB) = \det(A)\det(B) \neq 0$

$also\ MN\ is\ of\ the\ form\ of\ affine\ group, AB\ \in R^{nxn}, Ab_n + b_m \in \mathbb{R}^n, MN\ \in R^{n+1\ x\ n+1}$

<u>G3:</u> $let\ M \in affine\ group$

Define: $M = \begin{bmatrix} A & b \\ 0 & 1 \end{bmatrix}$

$we\ know\ that\ there\ exist\ A^{-1}\ such\ that\ AA^{-1} = A^{-1}A = I, because\ \det(A) \neq 0\ (invertible).$

$lets\ compose\ a\ new\ matrix\ N = \begin{bmatrix} A^{-1} & -A^{-1}b \\ 0 & 1 \end{bmatrix}$

$\mathrm{MN} = \begin{bmatrix} A & b \\ 0 & 1 \end{bmatrix}\begin{bmatrix} A^{-1} & -A^{-1}b \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} AA^{-1} & A \cdot -A^{-1}b + b \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} I_{nxn} & 0 \\ 0 & 1 \end{bmatrix} = I_{n+1\ x\ n+1}$

$\det(A^{-1}) \neq 0, obviously$

$\mathrm{N} = M^{-1} \in affine\ group$

## Problem 16

We will show that G1, G2 and G3 definitions are satisfied:

<u>G1:</u> $I_{n+1xn+1} = \begin{bmatrix} I_{nxn} & 0_{1xn} \\ 0 & 1 \end{bmatrix}, A = I_{nxn}, b = 0_{1xn}.$

$$A = I_{nxn} \quad => \quad \det(A) = \det(I_{n+1\ x\ n+1}) > 0$$

<u>G2:</u> $let\ M, N \in identity\ component\ of\ affine\ group$

$M = \begin{bmatrix} A_m & b_m \\ 0 & 1 \end{bmatrix}, N = \begin{bmatrix} A_n & b_n \\ 0 & 1 \end{bmatrix}$

$$MN = \begin{bmatrix} A_mA_n & A_mb_n + b_m \\ 0 & 1 \end{bmatrix}$$

We know $\det(A)$ , $\det(B) > 0$ as $M, N$ are in our group.

So: $\det(AB) = \det(A)\det(B) > 0$

$also\ MN\ is\ of\ the\ form\ of\ identity\ component\ of\ affine\ group, AB\ \in R^{nxn}, A_mb_n + b_m$
$\in \mathbb{R}^n, MN\ \in R^{n+1\ x\ n+1}$

So $MN$ is in our group.

*G3:* $let\ M$ be a matrix in our group.

Define: $M = \begin{bmatrix} A & b \\ 0 & 1 \end{bmatrix}$

$we\ know\ that\ there\ exists\ A^{-1}\ such\ that\ AA^{-1} = A^{-1}A = I, because\ \det(A) > 0\ (invertible).$

$lets\ compose\ a\ new\ matrix\ N = \begin{bmatrix} A^{-1} & -A^{-1}b \\ 0 & 1 \end{bmatrix}$

$\mathrm{MN} = \begin{bmatrix} A & b \\ 0 & 1 \end{bmatrix}\begin{bmatrix} A^{-1} & -A^{-1}b \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} AA^{-1} & A \cdot -A^{-1}b + b \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} I_{nxn} & 0 \\ 0 & 1 \end{bmatrix} = I_{n+1\,x\,n+1}$

We know $\det(\mathrm{A}) > 0, \det(I) > 0$, and $\det(I) = \det(AA^{-1}) = \det(A)\det(A^{-1})$ which means $\det(A^{-1}) > 0$ also.

So we can conclude:

$\mathrm{N} = M^{-1} \in the\ identity\ component\ of\ the\ affine\ group$

## Problem 17

Lets look at G2:

$M, N \in matrix\ group\ of\ the\ form\ \begin{bmatrix} A_m & b_m \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} A_n & b_n \\ 0 & 1 \end{bmatrix}, \det(A) < 0$

$$M_{n+1\,x\,n+1}N_{n+1\,x\,n+1} \in R^{n+1\,x\,n+1}$$

$$\det(A_m) < 0 \text{ and } \det(A_n) < 0$$

$$\det(A_m A_n) = \det(A_m)\det(A_n) > 0$$

But $MN = \begin{bmatrix} A_m B_n & \cdots \\ 0 & 1 \end{bmatrix}$ which means $MN$ is not in our group, therefore the group is not a matrix group.

## Problem 18

The answer is no. There is no continuous curve $c$, from the unit interval to the affine group, such that $c(0) = f$ and $c(1) = g$.

We will show that by observing any possible continuous curve $c(t)$, $t \in [0,1]$, such that:

$$c(t)(x) = A(t)x + b(t)$$

$$A(0) = A_0, A(1) = A_1$$

$$b(0) = b_0, b(1) = b_1$$

We know that:

$$\det\big(A(0)\big) = \det(A_0) > 0$$

$$\det\big(A(1)\big) = \det(A_1) < 0$$

$c$ is a continuous curve, so for any $i, j$:

$$\left(A(t)\right)_{i,j} = c_{i,j}(t)$$

$$c_{i,j} : [0,1] \to \mathbb{R}$$

And $c_{i,j}$ is continuous!

Now, as we know from the Leibniz formula, a determinant of a matrix is a polynomial expression of the matrix entries. Every entry in our $A(t)$ is continuous over $[0,1]$, therefore we can say $\det(A(t))$ is a continuous curve as well!

Now we know it begins in $\det(A_0) > 0$, and ends at $\det(A_1) < 0$, therefore exists $t_0$ such that $\det\left(A(t_0)\right) = 0$!

So we can see that we <u>do not have</u> the case $\det\left(A(t)\right) \neq 0, \forall t \in [0,1]$, showing no continuous curve exists between the two affine maps. ■

## Problem 19

### Part (i):

We need to find the least square estimator for $\boldsymbol{\theta}$.

We want to find $\boldsymbol{\theta}$ such that for all $i$: $\boldsymbol{u}_i \approx \boldsymbol{P} \cdot \begin{bmatrix} \boldsymbol{X}_i \\ 1 \end{bmatrix} + \boldsymbol{\epsilon}_i$.

So our problem is described by:

$$f(\boldsymbol{\theta}) = \sum_{i=1}^{N} \left\| \boldsymbol{P} \begin{bmatrix} \boldsymbol{X}_i \\ 1 \end{bmatrix} + \boldsymbol{\epsilon}_i - \boldsymbol{u}_i \right\|^2 \ and\ \boldsymbol{\theta_{LS}} = argmin_{\boldsymbol{\theta}}\left(f(\boldsymbol{\theta})\right)$$

Now let's observe $f(\boldsymbol{\theta})$:

$$f(\boldsymbol{\theta}) = \sum_{i=1}^{N} \left\| \begin{bmatrix} \theta_1 X_i + \theta_2 Y_i + a \\ \theta_3 X_i + bY_i + \theta_4 \end{bmatrix} + \boldsymbol{\epsilon}_i - \boldsymbol{u}_i \right\|^2 = \sum_{i=1}^{N} \left\| \begin{bmatrix} \theta_1 X_i + \theta_2 Y_i \\ \theta_3 X_i + \theta_4 \end{bmatrix} + \begin{bmatrix} a \\ bY_i \end{bmatrix} + \boldsymbol{\epsilon}_i - \boldsymbol{u}_i \right\|^2 =$$

$$= \sum_{i=1}^{N} \left\| \begin{bmatrix} X_i & Y_i & 0 & 0 \\ 0 & 0 & X_i & 1 \end{bmatrix} \cdot \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \end{bmatrix} + \begin{bmatrix} a \\ bY_i \end{bmatrix} + \boldsymbol{\epsilon}_i - \boldsymbol{u}_i \right\|^2$$

Now let:

$$\boldsymbol{H}_i = \begin{bmatrix} X_i & Y_i & 0 & 0 \\ 0 & 0 & X_i & 1 \end{bmatrix},\ \boldsymbol{y}_i = \boldsymbol{u}_i - \begin{bmatrix} a \\ bY_i \end{bmatrix} - \boldsymbol{\epsilon}_i = \begin{bmatrix} u_i - a - \boldsymbol{\epsilon}_i(1) \\ v_i - bY_i - \boldsymbol{\epsilon}_i(2) \end{bmatrix}$$

And we get:

$$f(\boldsymbol{\theta}) = \sum_{i=1}^{N} \|\boldsymbol{H}_i \cdot \boldsymbol{\theta} - \boldsymbol{y}_i\|^2$$

Now:

$$\boldsymbol{H} = [\boldsymbol{H}_1^T \quad \cdots \quad \boldsymbol{H}_N^T]^T = \begin{bmatrix}\begin{bmatrix} X_1 & 0 \\ Y_1 & 0 \\ 0 & X_1 \\ 0 & 1 \end{bmatrix} & \cdots & \begin{bmatrix} X_N & 0 \\ Y_N & 0 \\ 0 & X_N \\ 0 & 1 \end{bmatrix}\end{bmatrix}^T = \begin{bmatrix} X_1 & Y_1 & 0 & 0 \\ 0 & 0 & X_1 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ X_N & Y_N & 0 & 0 \\ 0 & 0 & X_N & 1 \end{bmatrix}$$

$$\boldsymbol{y} = [\boldsymbol{y}_1^T \quad \cdots \quad \boldsymbol{y}_N^T]^T = \begin{bmatrix}\begin{bmatrix} u_1 - a - \boldsymbol{\epsilon}_1(1) \\ v_1 - bY_1 - \boldsymbol{\epsilon}_1(2) \end{bmatrix}^T & \cdots & \begin{bmatrix} u_N - a - \boldsymbol{\epsilon}_N(1) \\ v_N - bY_N - \boldsymbol{\epsilon}_N(2) \end{bmatrix}^T\end{bmatrix}^T = \begin{bmatrix} u_1 - a - \boldsymbol{\epsilon}_1(1) \\ v_1 - bY_1 - \boldsymbol{\epsilon}_1(2) \\ \vdots \\ u_N - a - \boldsymbol{\epsilon}_N(1) \\ v_N - bY_N - \boldsymbol{\epsilon}_N(2) \end{bmatrix}$$

And as we've learned, the minimizer satisfies:

$$\boldsymbol{\theta_{LS}} = (\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T\boldsymbol{y} =$$

$$\boxed{= \begin{bmatrix} \sum X_i^2 & \sum X_iY_i & 0 & 0 \\ \sum X_iY_i & \sum Y_i^2 & 0 & 0 \\ 0 & 0 & \sum X_i^2 & \sum X_i \\ 0 & 0 & \sum X_i & N \end{bmatrix}^{-1} \cdot \begin{bmatrix} X_1 & 0 & \cdots & X_N & 0 \\ Y_1 & 0 & \cdots & Y_N & 0 \\ 0 & X_1 & \cdots & 0 & X_N \\ 0 & 1 & \cdots & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} u_1 - a - \boldsymbol{\epsilon}_1(1) \\ v_1 - bY_1 - \boldsymbol{\epsilon}_1(2) \\ \vdots \\ u_N - a - \boldsymbol{\epsilon}_N(1) \\ v_N - bY_N - \boldsymbol{\epsilon}_N(2) \end{bmatrix}} \blacksquare$$

## Part (ii):

For this part we will note that we are looking for (let $p$ be our probability density function):

$$g(\boldsymbol{\theta}) = \prod_{i=1}^{N} \mathrm{p}(\boldsymbol{H}_i \cdot \boldsymbol{\theta} = \boldsymbol{y_i}) = \prod_{i=1}^{N} \mathrm{p}\left(\boldsymbol{H}_i \cdot \boldsymbol{\theta} = \boldsymbol{u}_i - \begin{bmatrix} a \\ bY_i \end{bmatrix} - \boldsymbol{\epsilon}_i\right) = \prod_{i=1}^{N} \mathrm{p}\left(\boldsymbol{\epsilon}_i = -\boldsymbol{H}_i \cdot \boldsymbol{\theta} + \boldsymbol{u}_i - \begin{bmatrix} a \\ bY_i \end{bmatrix}\right) =$$

$$= \prod_{i=1}^{N} \mathrm{p}\left(\boldsymbol{\epsilon}_i = \begin{bmatrix} -X_i \cdot \theta_1 - Y_i \cdot \theta_2 + u_i - a \\ -X_i \cdot \theta_3 - \theta_4 + v_i - bY_i \end{bmatrix}\right)$$

We know the probability density function of $\epsilon_i$, so:

$$= \prod_{i=1}^{N} \frac{1}{2\pi\sigma^2} \cdot \exp\left( - \frac{\begin{bmatrix} -X_i \cdot \theta_1 - Y_i \cdot \theta_2 + u_i - a \\ -X_i \cdot \theta_3 - \theta_4 + v_i - bY_i \end{bmatrix}^T \cdot \begin{bmatrix} -X_i \cdot \theta_1 - Y_i \cdot \theta_2 + u_i - a \\ -X_i \cdot \theta_3 - \theta_4 + v_i - bY_i \end{bmatrix}}{2\sigma^2} \right) =$$

$$\Rightarrow \boxed{g(\boldsymbol{\theta}) = \left(\frac{1}{2\pi\sigma^2}\right)^N \cdot \prod_{i=1}^{N} \exp\left(-\frac{h_i(\theta_1, \theta_2, \theta_3, \theta_4)}{2\sigma^2}\right)}$$

Where:

$$\begin{aligned} h_i(\theta_1, \theta_2, \theta_3, \theta_4) \\ &= X_i^2\theta_1^2 + 2X_iY_i\theta_1\theta_2 + (a - u_i)X_i\theta_1 + Y_i^2\theta_2^2 + (a - u_i)Y_i\theta_2 + (a - u_i)^2 + X_i^2\theta_3^2 \\ &+ 2X_i\theta_3\theta_4 + (bY_i - v_i)X_i\theta_3 + \theta_4^2 + (bY_i - v_i)\theta_4 + (bY_i - v_i)^2 \quad \blacksquare \end{aligned}$$

## Part (iii):

The maximum likelihood estimator for $\boldsymbol{\theta}$ will be achieved by getting the maximum of the **log likelihood function** (as it is a monotonic non-decreasing function):

$$\log\big(g(\boldsymbol{\theta})\big) = \mathrm{N} \cdot \log\left(\frac{1}{2\pi\sigma^2}\right) - \frac{1}{2\sigma^2}\sum_{i=1}^{N} h_i(\theta_1, \theta_2, \theta_3, \theta_4)$$

To get the maximum $\boldsymbol{\theta}^* = \begin{bmatrix}\theta_1\\ \theta_2\\ \theta_3\\ \theta_4\end{bmatrix}$, we will look for a gradient of $\mathbf{0}$.

$$\frac{\partial(\log(g))}{\partial\theta_1} = -\frac{1}{2\sigma^2}\cdot\sum_{i=1}^{N} 2X_i^2\theta_1 + 2X_iY_i\theta_2 + (a - u_i)X_i = 0$$

$$\frac{\partial(\log(g))}{\partial\theta_2} = -\frac{1}{2\sigma^2}\cdot\sum_{i=1}^{N} 2Y_i^2\theta_2 + 2X_iY_i\theta_1 + (a - u_i)Y_i = 0$$

$$\frac{\partial(\log(g))}{\partial\theta_3} = -\frac{1}{2\sigma^2}\cdot\sum_{i=1}^{N} 2X_i^2\theta_3 + 2X_i\theta_4 + (bY_i - v_i)X_i = 0$$

$$\frac{\partial(\log(g))}{\partial\theta_4} = -\frac{1}{2\sigma^2}\cdot\sum_{i=1}^{N} 2\theta_4 + 2X_i\theta_3 + (bY_i - v_i) = 0$$

We will start with $\theta_1, \theta_2$:

$$\begin{cases}\left(\sum_{i=1}^{N} X_i^2\right)\theta_1 + \left(\sum_{i=1}^{N} X_iY_i\right)\theta_2 = \sum_{i=1}^{N}\frac{(u_i - a)X_i}{2}\\ \left(\sum_{i=1}^{N} X_iY_i\right)\theta_1 + \left(\sum_{i=1}^{N} Y_i^2\right)\theta_2 = \sum_{i=1}^{N}\frac{(u_i - a)Y_i}{2}\end{cases} \Rightarrow \begin{bmatrix}\sum_{i=1}^{N} X_i^2 & \sum_{i=1}^{N} X_iY_i\\ \sum_{i=1}^{N} X_iY_i & \sum_{i=1}^{N} Y_i^2\end{bmatrix}\cdot\begin{bmatrix}\theta_1\\ \theta_2\end{bmatrix} = \begin{bmatrix}\sum_{i=1}^{N}\frac{(u_i - a)X_i}{2}\\ \sum_{i=1}^{N}\frac{(u_i - a)Y_i}{2}\end{bmatrix}$$

$$\Rightarrow \boxed{\begin{bmatrix}\theta_1\\ \theta_2\end{bmatrix} = \frac{1}{\left(\sum_{i=1}^{N} X_i^2\right)\cdot\left(\sum_{i=1}^{N} Y_i^2\right) - \left(\sum_{i=1}^{N} X_iY_i\right)^2}\begin{bmatrix}\sum_{i=1}^{N} Y_i^2 & -\sum_{i=1}^{N} X_iY_i\\ -\sum_{i=1}^{N} X_iY_i & \sum_{i=1}^{N} X_i^2\end{bmatrix}\cdot\begin{bmatrix}\sum_{i=1}^{N}\frac{(u_i - a)X_i}{2}\\ \sum_{i=1}^{N}\frac{(u_i - a)Y_i}{2}\end{bmatrix}}$$

And now for $\theta_3, \theta_4$:

$$\begin{cases}\left(\sum_{i=1}^{N} X_i^2\right)\theta_3 + \left(\sum_{i=1}^{N} X_i\right)\theta_4 = \sum_{i=1}^{N}\frac{(v_i - bY_i)X_i}{2}\\ \left(\sum_{i=1}^{N} X_i\right)\theta_3 + N\cdot\theta_4 = \sum_{i=1}^{N}\frac{v_i - bY_i}{2}\end{cases} \Rightarrow \begin{bmatrix}\sum_{i=1}^{N} X_i^2 & \sum_{i=1}^{N} X_i\\ \sum_{i=1}^{N} X_i & N\end{bmatrix}\cdot\begin{bmatrix}\theta_3\\ \theta_4\end{bmatrix} = \begin{bmatrix}\sum_{i=1}^{N}\frac{(v_i - bY_i)X_i}{2}\\ \sum_{i=1}^{N}\frac{v_i - bY_i}{2}\end{bmatrix}$$

$$\Rightarrow \boxed{\begin{bmatrix}\theta_3\\ \theta_4\end{bmatrix} = \frac{1}{N\cdot\sum_{i=1}^{N}X_i^2-\left(\sum_{i=1}^{N}X_i\right)^2}\cdot\begin{bmatrix} N & -\sum_{i=1}^{N}X_i \\ -\sum_{i=1}^{N}X_i & \sum_{i=1}^{N}X_i^2\end{bmatrix}\cdot\begin{bmatrix}\sum_{i=1}^{N}\frac{(v_i-bY_i)X_i}{2}\\ \sum_{i=1}^{N}\frac{v_i-bY_i}{2}\end{bmatrix}} \quad \blacksquare$$

## Problem 20

We need to find:

$$\arg\min_{\boldsymbol{\theta}\in\mathbb{R}^k}\|\boldsymbol{H\theta}-\boldsymbol{y}\|_{l_2}^2+\lambda\|\boldsymbol{\theta}\|_{l_2}^2$$

Let's observe:

$$\|\boldsymbol{H\theta}-\boldsymbol{y}\|_{l_2}^2+\lambda\|\boldsymbol{\theta}\|_{l_2}^2 = \|\boldsymbol{H\theta}-\boldsymbol{y}\|_{l_2}^2+\left\|\sqrt{\lambda}\cdot\boldsymbol{\theta}\right\|_{l_2}^2 = \sum_{i=1}^{N}\left((\boldsymbol{H\theta}-\boldsymbol{y})(i)\right)^2+\sum_{i=1}^{k}\left(\left(\sqrt{\lambda}\cdot\boldsymbol{\theta}\right)(i)\right)^2 =$$

$$= \left\|\begin{bmatrix}\boldsymbol{H\theta}-\boldsymbol{y}\\ \sqrt{\lambda}\cdot\boldsymbol{\theta}\end{bmatrix}\right\|_{l_2}^2 = \left\|\begin{bmatrix}\boldsymbol{H\theta}\\ \sqrt{\lambda}\cdot\boldsymbol{\theta}\end{bmatrix}-\begin{bmatrix}\boldsymbol{y}\\ \boldsymbol{0}_{k\times 1}\end{bmatrix}\right\|_{l_2}^2 = \left\|\begin{bmatrix}\boldsymbol{H}\\ \sqrt{\lambda}\cdot\boldsymbol{I}_{k\times k}\end{bmatrix}\boldsymbol{\theta}-\begin{bmatrix}\boldsymbol{y}\\ \boldsymbol{0}_{k\times 1}\end{bmatrix}\right\|_{l_2}^2$$

And this is of LS form, so we know our $\boldsymbol{\theta}$ solution is:

$$\boldsymbol{\theta} = \left(\begin{bmatrix}\boldsymbol{H}\\ \sqrt{\lambda}\cdot\boldsymbol{I}_{k\times k}\end{bmatrix}^T\begin{bmatrix}\boldsymbol{H}\\ \sqrt{\lambda}\cdot\boldsymbol{I}_{k\times k}\end{bmatrix}\right)^{-1}\begin{bmatrix}\boldsymbol{H}\\ \sqrt{\lambda}\cdot\boldsymbol{I}_{k\times k}\end{bmatrix}^T\begin{bmatrix}\boldsymbol{y}\\ \boldsymbol{0}_{k\times 1}\end{bmatrix} =$$

$$\Rightarrow \boxed{\boldsymbol{\theta} = (\boldsymbol{H}^T\boldsymbol{H}+\lambda\cdot\boldsymbol{I}_{k\times k})^{-1}\boldsymbol{H}^T\boldsymbol{y}} \quad \blacksquare$$

## Problem 21

So we need to find:

$$\arg\min_{\hat{x}\in span(v_1,\ldots,v_k)}\|x-\hat{x}\|_{l_2}$$

But $\hat{x}\in span(v_1,\ldots,v_k)$ means:

$$\hat{x} = \sum_{j=1}^{k}\theta_j v_j = \boldsymbol{V}\cdot\boldsymbol{\theta} \text{ for some } \boldsymbol{\theta} = \begin{bmatrix}\theta_1\\ \vdots\\ \theta_k\end{bmatrix}\in\mathbb{R}^k$$

So we can rewrite our problem:

$$\arg\min_{\theta\in\mathbb{R}^k}\|\boldsymbol{x}-\boldsymbol{V}\cdot\boldsymbol{\theta}\|_{l_2}$$

As when we find the right $\boldsymbol{\theta}$, we can get our $\hat{x}$.

Now, as we know $x^2$ is a monotonic non-decreasing function, so we can rewrite again:

$$\arg\min_{\theta\in\mathbb{R}^k}\|\boldsymbol{x}-\boldsymbol{V}\cdot\boldsymbol{\theta}\|_{l_2} = \arg\min_{\theta\in\mathbb{R}^k}\|\boldsymbol{x}-\boldsymbol{V}\cdot\boldsymbol{\theta}\|_{l_2}^2 = \arg\min_{\theta\in\mathbb{R}^k}\|\boldsymbol{V\theta}-\boldsymbol{x}\|_{l_2}^2$$

And now we can it's a LS problem. For $N = 1$:

$$\boldsymbol{H} = \boldsymbol{H}_1 = \boldsymbol{V} \in \mathbb{R}^{d\times k}$$

$$\boldsymbol{y} = \boldsymbol{y}_1 = \boldsymbol{x} \in \mathbb{R}^{d\times 1}$$

$$\boldsymbol{\theta} \in \mathbb{R}^{k\times 1}$$

$$\Rightarrow \boldsymbol{r} = \boldsymbol{H\theta} - \boldsymbol{y} = \boldsymbol{V\theta} - \boldsymbol{x}$$

$$\Rightarrow \boldsymbol{\theta_{LS}} = \arg\min_{\boldsymbol{\theta}\in\mathbb{R}^k} \|\boldsymbol{r}\|_{l_2}^2 = \arg\min_{\theta\in\mathbb{R}^k} \|\boldsymbol{V\theta} - \boldsymbol{x}\|_{l_2}^2$$

So now we can use our known solution:

$$\boldsymbol{\theta_{LS}} = (\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T\boldsymbol{y} = (\boldsymbol{V}^T\boldsymbol{V})^{-1}\boldsymbol{V}^T\boldsymbol{x} = \boldsymbol{V}^T\boldsymbol{x}$$

$$\Rightarrow \boxed{\boldsymbol{\theta} = \boldsymbol{V}^T\boldsymbol{x}}$$

$$\Rightarrow \boxed{\widehat{\boldsymbol{x}} = \boldsymbol{V}\cdot\boldsymbol{\theta} = \boldsymbol{V}\cdot\boldsymbol{V}^T\cdot\boldsymbol{x}} \quad \blacksquare$$

## Problem 22

Definition: $(a * b)(i,j) = \sum_{k,l} a(i-k, j-l)\, b(k,l)$

$$\begin{aligned}
\big((a*b)*c\big)(i,j) &= \sum\nolimits_{m,n} (a*b)(i-m, j-n)\, c(m,n) \\
&= \sum\nolimits_{m,n}\left(\sum\nolimits_{k,l} a(i-m-k, j-n-l) b(k,l)\right) c(m,n) =_{t:=k+m, s:=l+n} \\
&= \sum\nolimits_{m,n}\left(\sum\nolimits_{t,s} a(i-t, j-s) b(t-m, s-n)\right) c(m,n) \\
&= \sum\nolimits_{m,n}\left(\sum\nolimits_{t,s} a(i-t, j-s) b(t-m, s-n) c(m,n)\right) \\
&= \sum\nolimits_{t,s}\left(\sum\nolimits_{m,n} a(i-t, j-s) b(t-m, s-n) c(m,n)\right) \\
&= \sum\nolimits_{t,s} a(i-t, j-s) \cdot \sum\nolimits_{m,n} b(t-m, s-n) c(m,n) \\
&= \sum\nolimits_{t,s} a(i-t, j-s) \cdot (b*c)(t,s) = a*(b*c)\,(i,j) \blacksquare
\end{aligned}$$

## Problem 23

First of all, we have seen in definition 9 that $(I * h)(i,j) = (h * I)(i,j)$ for all $h$.

So all we need to show is that $(h * \delta)(\mathrm{i},\mathrm{j}) = h(i,j)$ for all $i, j$.

**Intuition:** if at (0,0) the value of $\delta$ is 1 and otherwise its 0, it means that it preserve only the pixel that the convolution is affecting, without any influence by other pixels (they are multiplied by 0).

$$(\mathrm{h} * \delta)(\mathrm{i},\mathrm{j}) = \sum\nolimits_{k,l} h(i-k, j-l)\, \delta(k,l)$$

$$\delta(k,l) = \begin{cases} 1, & k = 0, l = 0 \\ 0, & otherwise \end{cases}$$

$$(\mathrm{h} * \delta)(\mathrm{i}, \mathrm{j}) = 0 + 0 + \cdots\ h(i-0, j-0)\delta(0,0) = h(i,j), for\ all\ i, j.$$

Therefore $\mathrm{h} * \delta = h$ ■

## Problem 24

First of all let's be clear about our "column vector" versions of $x$ and $y$:

$$(1)\ \ \boldsymbol{x}(N \cdot (i-1) + j) = x(i,j)$$

$$(2)\ \ \boldsymbol{y}(N \cdot (i-1) + j) = y(i,j)$$

If we want to do filter $(3x3)$ on $x(i,j)$ the definition of it is:

$$y(i,j) = \sum_{k=-1}^{1} \sum_{l=-1}^{1} x(i-k, j-l) h(k,l) =_{(1)} \sum_{k=-1}^{1} \sum_{l=-1}^{1} \boldsymbol{x}(N(i-k-1) + j - l) h(k,l) =$$

$$= \sum_{k=-1}^{1} \sum_{l=-1}^{1} \boldsymbol{x}(N(i-1) + j - Nk - l) h(k,l)$$

But we know $\boldsymbol{y}(N \cdot (i-1) + j) = y(i,j)$, so:

$$\boldsymbol{y}(N \cdot (i-1) + j) = \sum_{k=-1}^{1} \sum_{l=-1}^{1} \boldsymbol{x}(N(i-1) + j - Nk - l) h(k,l)$$

Now let's re-purpose our indexes, marking $i \coloneqq N(i-1) + j$. Recall we are using the zero-boundary assumption.

$$(*)\ \ \boldsymbol{y}(i) = \sum_{k=-1}^{1} \sum_{l=-1}^{1} \boldsymbol{x}(i - Nk - l) h(k,l)$$

Now let's be clear about our goal. We need to define $\boldsymbol{H}(i,j)$ for all $1 \le i, j \le MN$ such that:

$$\boldsymbol{y} = \boldsymbol{H} \cdot \boldsymbol{x}$$

$\boldsymbol{y}$ and $\boldsymbol{x}$ are both column vectors, so according to our goal, lets see what each value (which is a row) of $\boldsymbol{y}$ should be:

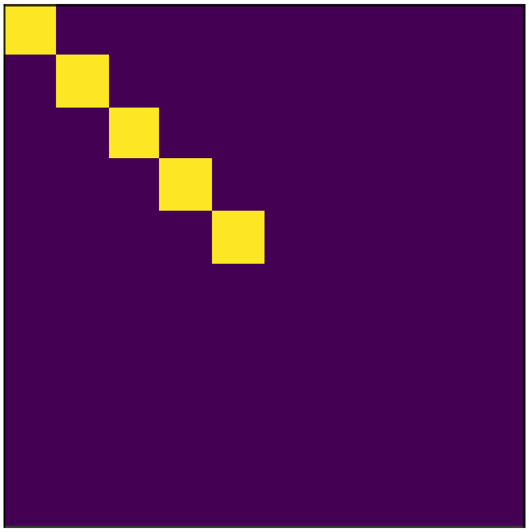
$$\Longrightarrow 1 \le i \le MN : \boldsymbol{y}(i) = \sum_{j=1}^{MN} H(i,j) \cdot \boldsymbol{x}(j)$$

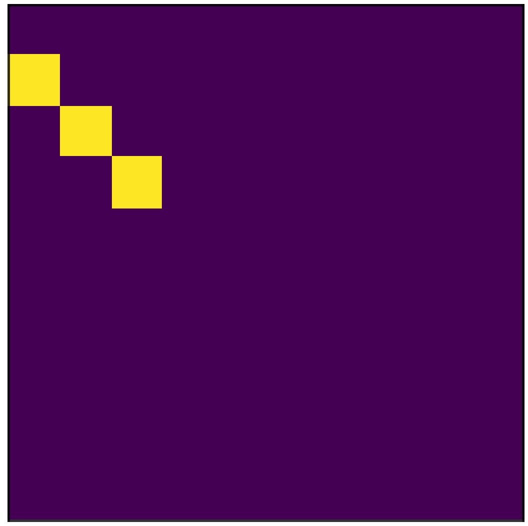
Now using this with our $(*)$, we can safely define $H(i,j)$:

$$\boxed{H(i,j) = \begin{cases} h(k,l) & j = i - Nk - l, \quad -1 \le k, l \le 1 \\ 0 & else \end{cases}} \quad ■$$

## Problem 25

The effect of convolving an image with this given $h$, is moving the entire image 2 pixels to the left, and 1 pixel upwards. This is a very small change, barely noticeable in sizeable images, so we can see this with this very small image example:




## Problem 26

### Part (i)

$$1 = \det(I_{n\times n}) = \det(YY^T) = \det Y \cdot \det Y^T = (\det Y)^2$$

$$\Longrightarrow \boxed{\det Y = \pm 1} \ \blacksquare$$

### Part (ii)

From $YY^T = I_{n\times n}$, we get $Y^{-1} = Y^T$ by definition of the inverse matrix. Therefore directly we can see that:

$$I_{n\times n} = YY^{-1} = Y^{-1}Y = Y^TY = Y^T(Y^T)^T$$

Proving that $Y^T$ is an orthogonal square matrix. ■

### Part (iii)

For any $i$, directly from part (ii), by definition of matrix multiplication we get (as $y_i$ is a <u>column</u> of $Y$):

$$1 = (I_{n\times n})_{i,i} = (Y^TY)_{i,i} = y_i^T y_i$$

$$\Longrightarrow \boxed{y_i^T y_i = 1} \ \blacksquare$$

### Part (iv)

For any $i$, by definition:

$$\|y_i\|_{l_2} = \sum_{j=1}^{n} (y_i)_j^2 = y_i^T y_i$$

$$\Longrightarrow \boxed{\|y_i\|_{l_2} = 1} \ \blacksquare$$

### Part (v)

Similarly to part (iii), for any $i$:

$$0 = (I_{n\times n})_{i,j} = (Y^TY)_{i,j} = y_i^T y_j$$

$$\Longrightarrow \boxed{y_i^T y_j = 0}$$

This means the angle between $y_i$ and $y_j$ is 90°. ■

## Part (vi)

We will prove this by definition. We mark this group of all $n \times n$ orthogonal matrices $G$.

1. $I_{n\times n} \in G$
   This is clear because $I_{n\times n} \cdot (I_{n\times n})^T = I_{n\times n} \cdot I_{n\times n} = I_{n\times n}$, meaning it's orthogonal so in $G$.
2. We need to show that if $A, B \in G$, so is $AB$.
   Let's observe, remembering both $A, B$ are orthogonal matrices:
   $$(AB) \cdot (AB)^T = A \cdot B \cdot B^T \cdot A^T = A \cdot I_{n\times n} \cdot A^T = A \cdot A^T = I_{n\times n}$$
   $\Longrightarrow AB$ is an orthogonal matrix so in $G$.
3. We need to show that if $A \in G$, $A^{-1}$ exists and it's in $G$.
   $A$ is orthogonal so $AA^T = I_{n\times n}$ meaning $A^{-1}$ exists and $A^{-1} = A^T$.
   We've proven before that $A^T$ is also orthogonal, meaning it's in $G$, so $A^{-1}$ is in $G$ as we wanted.

And that shows by definition that our group is a matrix group. ■

## Problem 27

$K$ is a $n \times n$ separable filter. Is it invertible? Well a matrix is invertible if and only if it has a full rank. So we need to see whether $rank(K) = n$.

$K$ is separable so:

$$K = USV^T$$

Such that $S$ is a diagonal matrix with values $s_1, \dots, s_n$, and **only on of them** is a non-zero!

This directly means $rank(S) = 1$!

And from fact 3 we know:

$$rank(K) = rank(USV^T) \leq \min\big(rank(US), rank(V^T)\big) \leq \min\big(rank(U), rank(S), rank(V^T)\big) \leq 1$$

So we got $rank(K) \leq 1$ (it's actually 1 otherwise it's a 0 matrix). So unless $n = 1$ which is a very trivial case, $K$ is not invertible. ■

## Problem 28

Bilateral filtering is <u>not</u> a linear operation, because as we can see in equation (28), the filter operation is dependent on the $f_r$ in a non-linear way.

Any change to the input image will <u>not</u> result in a proportional change in the output.
First of all $f_r$ is not necessarily linear, and even if it is: It's input is in absolute value which is not linear.
In addition, the $\frac{1}{c}$ component which has $f_r$ in it, is not linear in any way.

## Problem 29

$$\nabla^2 G(x,y,\sigma) = \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right) G(x,y,\sigma)$$

$$G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

First derivative:

$$\frac{\partial\, G(x,y,\sigma)}{\partial x} = -\frac{x}{2\pi\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$\frac{\partial\, G(x,y,\sigma)}{\partial y} = -\frac{y}{2\pi\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Second derivative:

$$\frac{\partial^2\, G(x,y,\sigma)}{\partial^2 x} = \frac{(-1 + \frac{x^2}{\sigma^2})}{2\pi\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$\frac{\partial^2\, G(x,y,\sigma)}{\partial^2 y} = \frac{(-1 + \frac{y^2}{\sigma^2})}{2\pi\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$\begin{aligned}
\nabla^2 G(x,y,\sigma) &= \left(\frac{\partial^2 G(x,y,\sigma)}{\partial x^2} + \frac{\partial^2 G(x,y,\sigma)}{\partial y^2}\right) = \frac{(-1 + \frac{x^2}{\sigma^2})}{2\pi\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}} + \frac{(-1 + \frac{y^2}{\sigma^2})}{2\pi\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}} \\
&= \left(\frac{\left(-1 + \frac{x^2}{\sigma^2}\right)}{\sigma^2} + \frac{\left(-1 + \frac{y^2}{\sigma^2}\right)}{\sigma^2}\right) \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} = \left(\frac{x^2}{\sigma^4} + \frac{y^2}{\sigma^4} - \frac{2}{\sigma^2}\right) G(x,y,\sigma) \\
&= \left(\frac{x^2+y^2}{\sigma^4} - \frac{2}{\sigma^2}\right) G(x,y,\sigma)
\end{aligned}$$

## Problem 30

First derivatives:

$$\frac{\partial I}{\partial x} = \frac{\partial I}{\partial x'} \cdot \frac{\partial x'}{\partial x} + \frac{\partial I}{\partial y'} \cdot \frac{\partial y'}{\partial x} = \frac{\partial I}{\partial x'} \cdot \cos\theta + \frac{\partial I}{\partial y'} \cdot \sin\theta$$

$$\frac{\partial I}{\partial y} = \frac{\partial I}{\partial x'} \cdot \frac{\partial x'}{\partial y} + \frac{\partial I}{\partial y'} \cdot \frac{\partial y'}{\partial y} = \frac{\partial I}{\partial x'} \cdot (-\sin\theta) + \frac{\partial I}{\partial y'} \cdot \cos\theta$$

Second derivatives:

$$
\begin{aligned}
\frac{\partial^2 I}{\partial^2 x} = \frac{\partial}{\partial x}\left(\frac{\partial I}{\partial x}\right) &= \frac{\partial}{\partial x}\left(\frac{\partial I}{\partial x'} \cdot \cos\theta + \frac{\partial I}{\partial y'} \cdot \sin\theta\right) = \left(\frac{\partial}{\partial x}\frac{\partial I}{\partial x'}\right) \cdot \cos\theta + \left(\frac{\partial}{\partial x}\frac{\partial I}{\partial y'}\right) \cdot \sin\theta \\
&= \left(\frac{\partial}{\partial x'}\frac{\partial I}{\partial x}\right) \cdot \cos\theta + \left(\frac{\partial}{\partial y'}\frac{\partial I}{\partial x}\right) \cdot \sin\theta \\
&= \frac{\partial}{\partial x'}\left(\frac{\partial I}{\partial x'} \cdot \cos\theta + \frac{\partial I}{\partial y'} \cdot \sin\theta\right) \cdot \cos\theta + \frac{\partial}{\partial y'}\left(\frac{\partial I}{\partial x'} \cdot \cos\theta + \frac{\partial I}{\partial y'} \cdot \sin\theta\right) \cdot \sin\theta \\
&= \frac{\partial^2 I}{\partial^2 x'} \cdot \cos^2\theta + 2 \cdot \frac{\partial^2 I}{\partial x' \partial y'} \cdot \sin\theta\cos\theta + \frac{\partial^2 I}{\partial^2 y'}\sin^2\theta
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial^2 I}{\partial^2 y} = \frac{\partial}{\partial y}\left(\frac{\partial I}{\partial y}\right) &= \frac{\partial}{\partial y}\left(\frac{\partial I}{\partial x'} \cdot (-\sin\theta) + \frac{\partial I}{\partial y'} \cdot \cos\theta\right) = \left(\frac{\partial}{\partial y}\frac{\partial I}{\partial x'}\right) \cdot (-\sin\theta) + \left(\frac{\partial}{\partial y}\frac{\partial I}{\partial y'}\right) \cdot \cos\theta \\
&= \left(\frac{\partial}{\partial x'}\frac{\partial I}{\partial y}\right) \cdot (-\sin\theta) + \left(\frac{\partial}{\partial y'}\frac{\partial I}{\partial y}\right) \cdot \cos\theta \\
&= \frac{\partial}{\partial x'}\left(\frac{\partial I}{\partial x'} \cdot (-\sin\theta) + \frac{\partial I}{\partial y'} \cdot \cos\theta\right) \cdot (-\sin\theta) + \frac{\partial}{\partial y'}\left(\frac{\partial I}{\partial x'} \cdot (-\sin\theta) + \frac{\partial I}{\partial y'} \cdot \cos\theta\right) \\
&\cdot \cos\theta = \frac{\partial^2 I}{\partial^2 x'} \cdot \sin^2\theta - 2 \cdot \frac{\partial^2 I}{\partial x' \partial y'} \cdot \sin\theta\cos\theta + \frac{\partial^2 I}{\partial^2 y'}\cos^2\theta
\end{aligned}
$$

Now finally we can see:

$$
\begin{aligned}
\frac{\partial^2 I}{\partial^2 x} + \frac{\partial^2 I}{\partial^2 y} &= \frac{\partial^2 I}{\partial^2 x'} \cdot \cos^2\theta + \cancel{2 \cdot \frac{\partial^2 I}{\partial x' \partial y'} \cdot \sin\theta\cos\theta} + \frac{\partial^2 I}{\partial^2 y'}\sin^2\theta + \frac{\partial^2 I}{\partial^2 x'} \cdot \sin^2\theta - \cancel{2 \cdot \frac{\partial^2 I}{\partial x' \partial y'}} \\
&\cancel{\cdot \sin\theta\cos\theta} + \frac{\partial^2 I}{\partial^2 y'}\cos^2\theta = \frac{\partial^2 I}{\partial^2 x'} \cdot (\cos^2\theta + \sin^2\theta) + \frac{\partial^2 I}{\partial^2 y'}(\sin^2\theta + \cos^2\theta) \\
&= \frac{\partial^2 I}{\partial^2 x'} + \frac{\partial^2 I}{\partial^2 y'} \quad \blacksquare
\end{aligned}
$$

# Computer Exercises

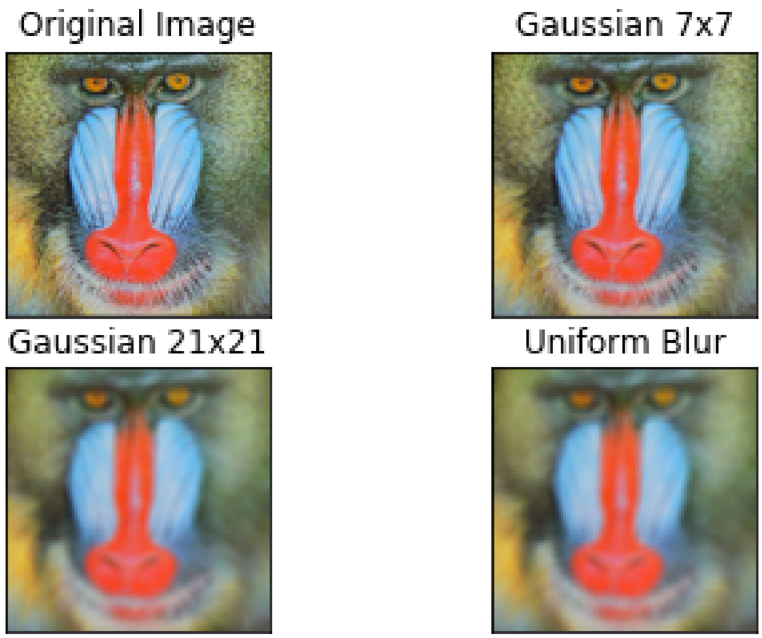
April 22, 2020

## 1 Computer Exercise 1

[1]:
```
import cv2
import numpy as np
from matplotlib import pyplot as plt
import copy
```

[2]:
```
inp_img = cv2.imread("./hw2_data/mandrill.png")
image = inp_img[:,:,::-1]

output = [(image, "Original Image")]

output.append((cv2.GaussianBlur(image,(7,7),3), "Gaussian 7x7"))
output.append((cv2.GaussianBlur(image,(21,21),10), "Gaussian 21x21"))
output.append((cv2.blur(image, (21, 21)), "Uniform Blur"))
```

[3]:
```
for i in range(len(output)):
    plt.subplot(2, 2, i+1)
    plt.imshow(output[i][0], interpolation=None), plt.title(output[i][1]), plt.
↪xticks([]), plt.yticks([])

plt.show()
```



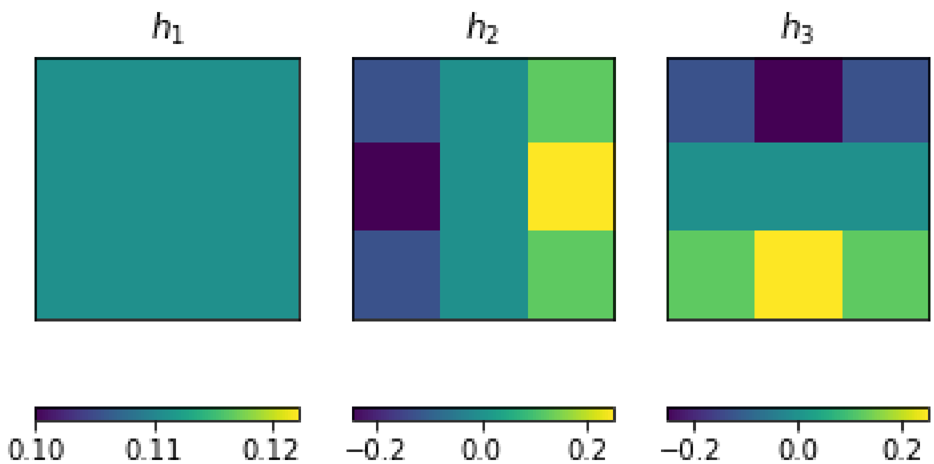

## 2 Computer Exercise 2

[1]:
```
import cv2
import numpy as np
from matplotlib import pyplot as plt
import copy
```

[2]:
```
M = 12
N = 16

def print_h(mat, name):
#     print(name + ":\n" + str(mat))
    _ = plt.imshow(mat, interpolation=None)
    plt.colorbar()
    _ = plt.title("$" + name[0] + "_" + name[1] + "$")


def create_H(h):
    H = np.zeros((M*N, M*N))
    for i in range(M*N):
        for k in [-1, 0, 1]:
            for l in [-1, 0, 1]:
                j = i-N*k-l
                if 0 <= j < M*N:
```

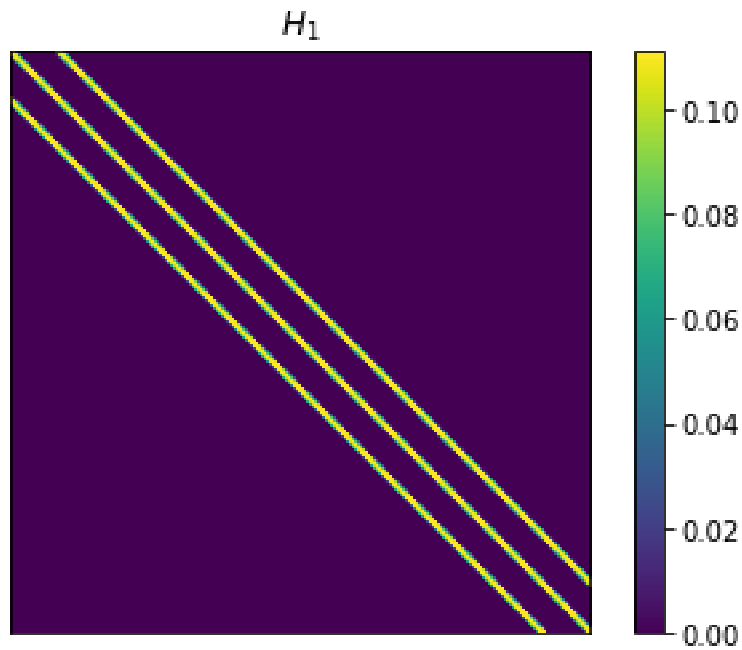
```
                    H[i,j] = h[k+1,l+1]
    return H
```

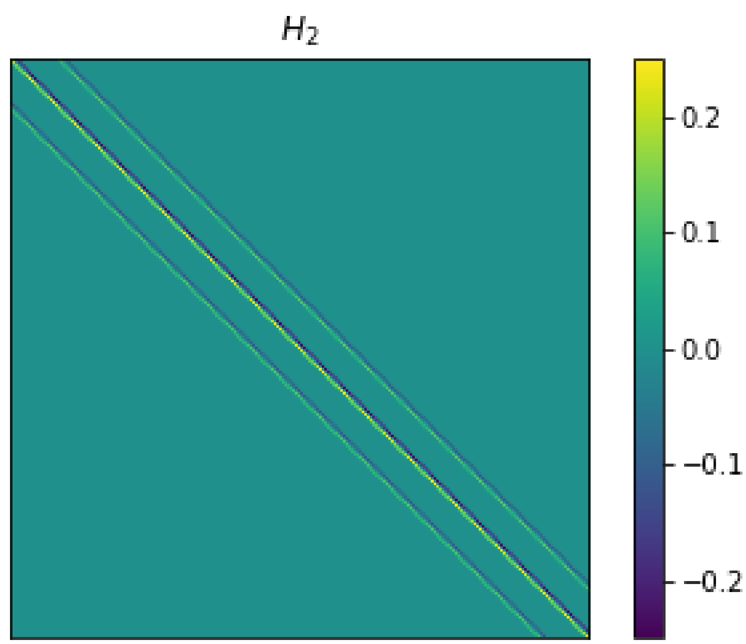
[3]:
```
hs = []

hs.append(((1.0/9.0)*np.ones((3,3)), "$h_1$"))
hs.append(((1.0/8.0)*np.array([[-1, 0, 1], [-2, 0, 2], [-1, 0, 1]]), "$h_2$"))
hs.append((np.transpose(hs[1][0]), "$h_3$"))

fig = plt.figure()
for i in range(len(hs)):
    a = fig.add_subplot(1, 3, i+1)
    implot = plt.imshow(hs[i][0], interpolation=None)
    a.set_title(hs[i][1])
    plt.xticks([]), plt.yticks([])
    plt.colorbar(orientation='horizontal')

plt.show()
```

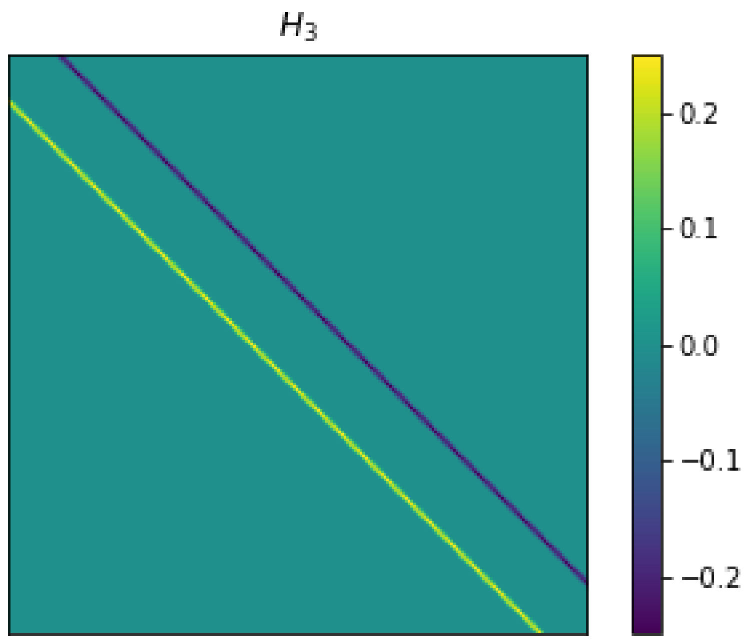



[4]:
```
H1 = create_H(hs[0][0])
H2 = create_H(hs[1][0])
H3 = create_H(hs[2][0])

Hs = []

Hs.append((H1,"$H_1$"))
Hs.append((H2,"$H_2$"))
Hs.append((H3,"$H_3$"))

for i in range(len(hs)):
```

```
    fig, a = plt.subplots()
    implot = plt.imshow(Hs[i][0], interpolation=None)
    a.set_title(Hs[i][1])
    a.set_xticks([]), a.set_yticks([])
    plt.colorbar()
    plt.show()
```






## 3 Computer Exercise 3

[1]:
```
import cv2
import numpy as np
from matplotlib import pyplot as plt
import copy
```

[2]:
```
import copy
import scipy.io as sio
```

[3]:
```
mdict = sio.loadmat("./hw2_data/are_these_separable_filters.mat")
K1 = mdict["K1"]
K2 = mdict["K2"]
K3 = mdict["K3"]

def zeroSmallNumbers(arr):
    newarr = []
    count = 0
    for item in arr:
        if np.abs(item) < 0.000000000001:
            newarr.append(0)
        else:
```

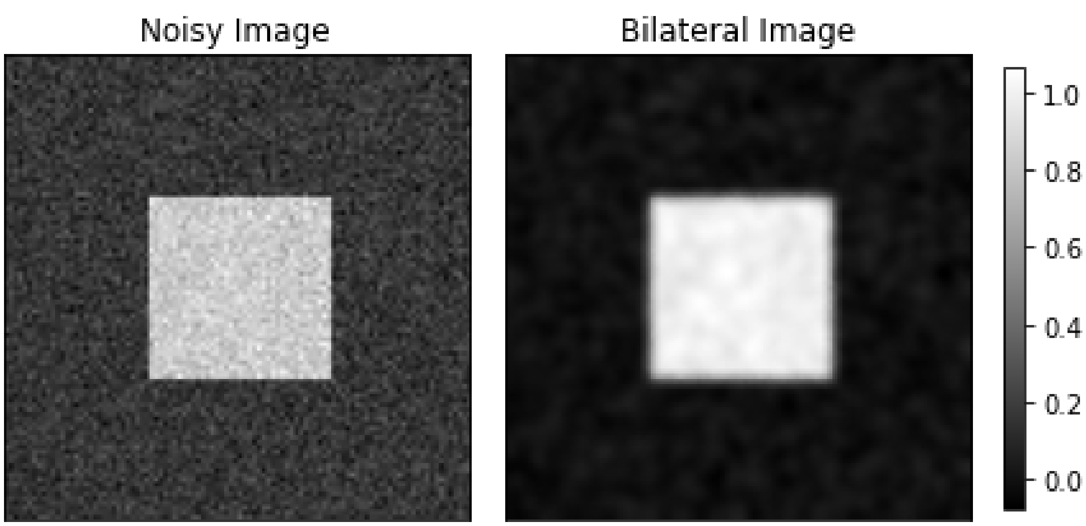
```
            newarr.append(item)
            count += 1
    return newarr, count


U1, s1, V1 = np.linalg.svd(K1, full_matrices=True)
U2, s2, V2 = np.linalg.svd(K2, full_matrices=True)
U3, s3, V3 = np.linalg.svd(K3, full_matrices=True)

singular1, count1 = zeroSmallNumbers(s1)
singular2, count2 = zeroSmallNumbers(s2)
singular3, count3 = zeroSmallNumbers(s3)

print("Singular Values for K1:\n" + str(np.diag(singular1)))
print("Kernel is separable\n" if count1 == 1 else "Kernel is not separable\n")
print("Singular Values for K2:\n" + str(np.diag(singular2)))
print("Kernel is separable\n" if count2 == 1 else "Kernel is not separable\n")
print("Singular Values for K3:\n" + str(np.diag(singular3)))
print("Kernel is separable\n" if count3 == 1 else "Kernel is not separable\n")
```

```
Singular Values for K1:
[[1.15572864 0.         0.         0.         0.        ]
 [0.         0.         0.         0.         0.        ]
 [0.         0.         0.         0.         0.        ]
 [0.         0.         0.         0.         0.        ]
 [0.         0.         0.         0.         0.        ]]
Kernel is separable

Singular Values for K2:
[[2.39244588 0.         0.         0.         0.        ]
 [0.         0.98749722 0.         0.         0.        ]
 [0.         0.         0.67563127 0.         0.        ]
 [0.         0.         0.         0.13996899 0.        ]
 [0.         0.         0.         0.         0.07304454]]
Kernel is not separable

Singular Values for K3:
[[3.6587224  0.         0.         0.         0.        ]
 [0.         0.29429662 0.         0.         0.        ]
 [0.         0.         0.         0.         0.        ]
 [0.         0.         0.         0.         0.        ]
 [0.         0.         0.         0.         0.        ]]
Kernel is not separable
```

## 4 Computer Exercise 4

[3]:
```python
import cv2
import numpy as np
from matplotlib import pyplot as plt
import copy
import scipy.io as sio
```

[4]:
```python
mdict = sio.loadmat("./hw2_data/bilateral.mat")
image = mdict["img_noisy"]

bilateral = cv2.bilateralFilter(image, 7, 1.5, 1.5)

fig, (ax1, ax2) = plt.subplots(1, 2, constrained_layout=True)

ax1.imshow(image, cmap="gray", interpolation=None)
ax1.set_title("Noisy Image")
ax1.set_xticks([]), ax1.set_yticks([])

pcm = ax2.imshow(bilateral, cmap="gray", interpolation=None)
ax2.set_title("Bilateral Image")
ax2.set_xticks([]), ax2.set_yticks([])

# ax = plt.gca()
fig.colorbar(pcm, ax=[ax1, ax2], location='right', shrink=0.6)
plt.show()
```



# 5 Computer Exercise 5

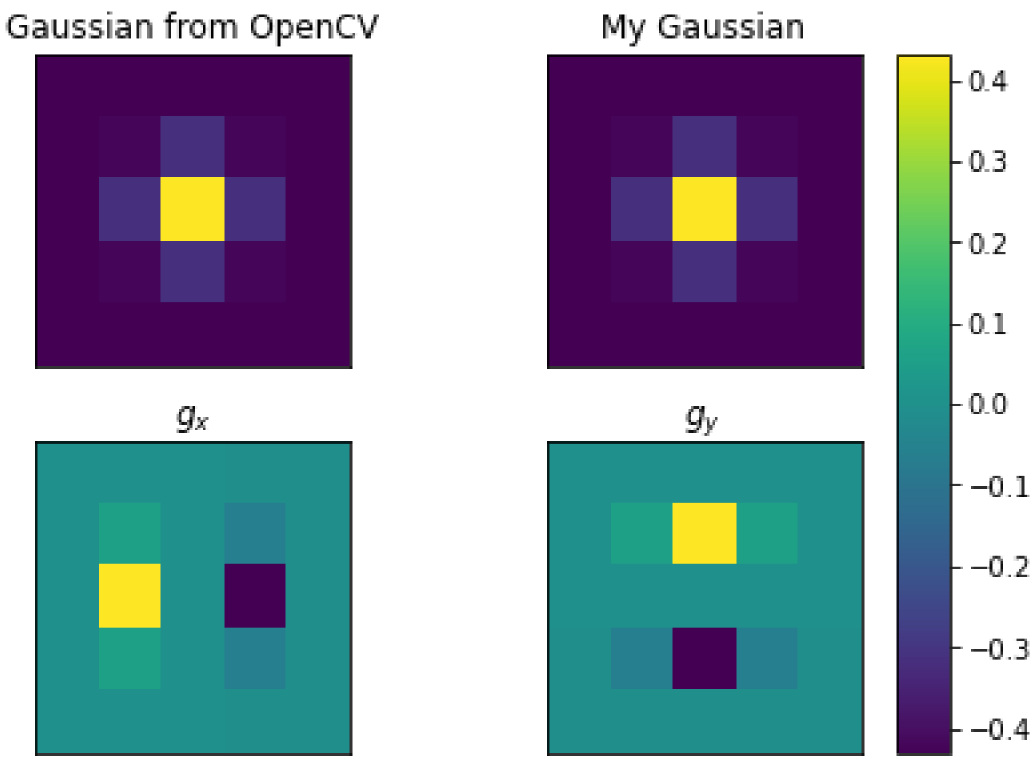
[88]:
```
import cv2
import numpy as np
from matplotlib import pyplot as plt
```

[89]:
```
def print_img_subplot(ax, img, title, is_gray):
    if is_gray:
        im_plt = ax.imshow(img, cmap='gray', interpolation=None)
    else:
        im_plt = ax.imshow(img, interpolation=None)
    ax.set_title(title)
    ax.set_xticks([])
    ax.set_yticks([])
    return im_plt
```

### 5.0.1 Creating Gaussian

[90]:
```
ksize = 5
sigma = 0.5

g = cv2.getGaussianKernel(ksize, sigma)
gg = g.dot(g.T)

outputs = [(gg, "Gaussian from OpenCV")]

my_gg = np.zeros((ksize,ksize))
for i, row in enumerate(my_gg):
    for j, val in enumerate(row):
        y = i - (ksize-1)/2
        x = j - (ksize-1)/2
        my_gg[i,j] = (1.0/(np.sqrt(2*np.pi*sigma**2)))*np.exp(-0.5*((x**2+y**2)/
 ↪(sigma**2)))
outputs.append((my_gg, "My Gaussian"))

deriv_x = np.zeros((ksize,ksize))
for i, row in enumerate(deriv_x):
    for j, val in enumerate(row):
        y = i - (ksize-1)/2
        x = j - (ksize-1)/2
        deriv_x[i,j] = (-x/(sigma**2))*my_gg[i,j]
outputs.append((deriv_x, "$g_x$"))

deriv_y = np.zeros((ksize,ksize))
for i, row in enumerate(deriv_y):
    for j, val in enumerate(row):
```

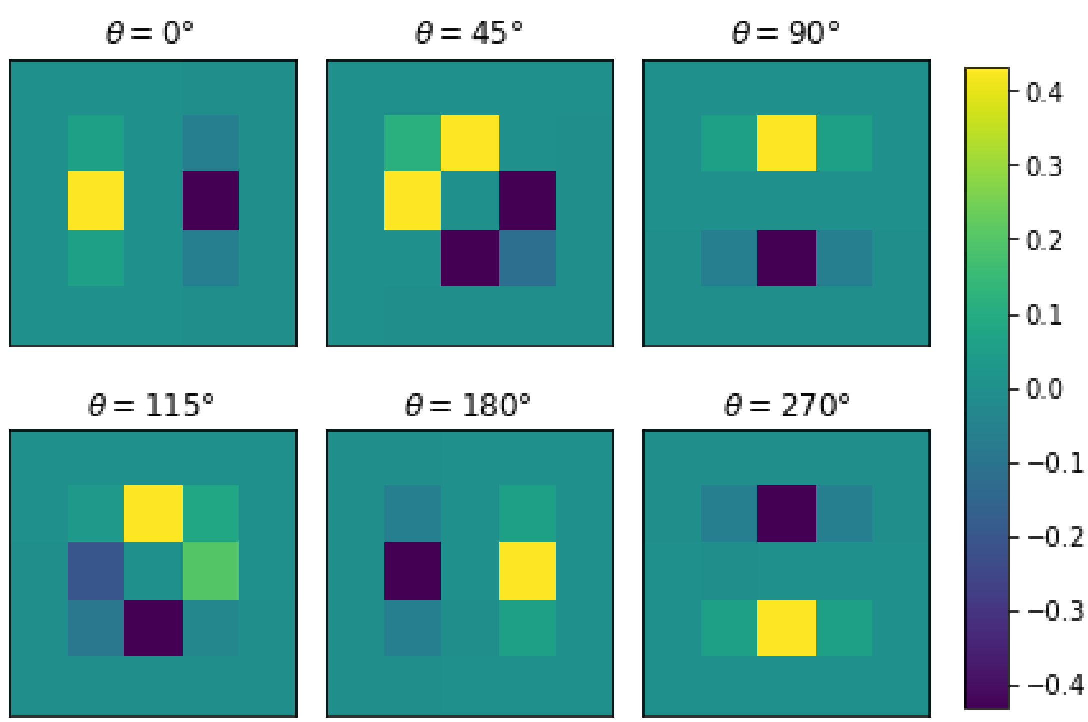
```
        y = i - (ksize-1)/2
        x = j - (ksize-1)/2
        deriv_y[i,j] = (-y/(sigma**2))*my_gg[i,j]
outputs.append((deriv_y, "$g_y$"))

fig1, axs1 = plt.subplots(2, 2, constrained_layout=True)

for i, out in enumerate(outputs):
    im_plt = print_img_subplot(axs1[int(i/2),i%2], out[0], out[1], False)

fig1.colorbar(im_plt, ax=axs1, location='right', shrink=1.)
plt.show()
```




#### 5.0.2 Directional Filters

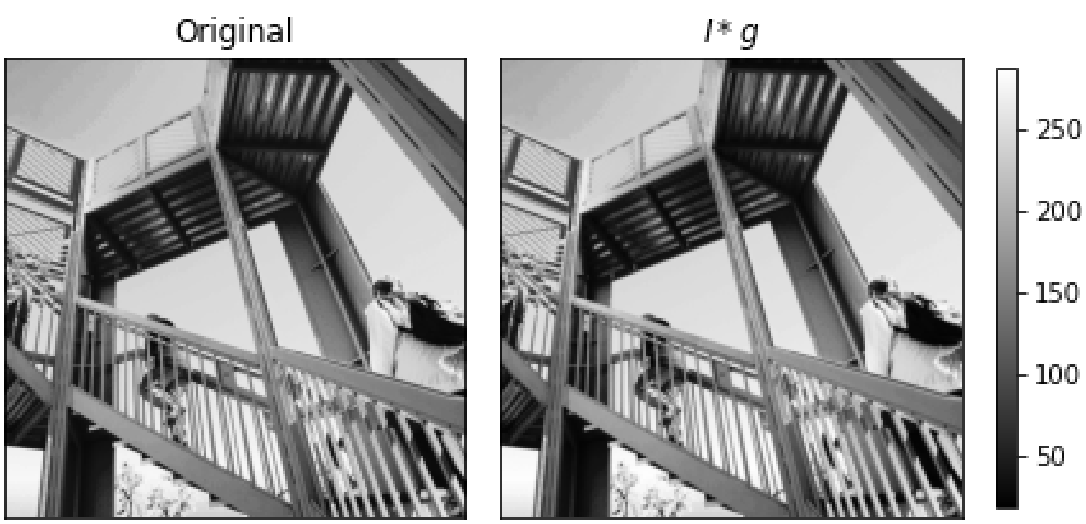
[91]:
```
def create_theta_filter(theta):
    deriv_theta = np.zeros((ksize,ksize))
    for i, row in enumerate(deriv_theta):
        for j, val in enumerate(row):
            y = i - (ksize-1)/2
            x = j - (ksize-1)/2
```

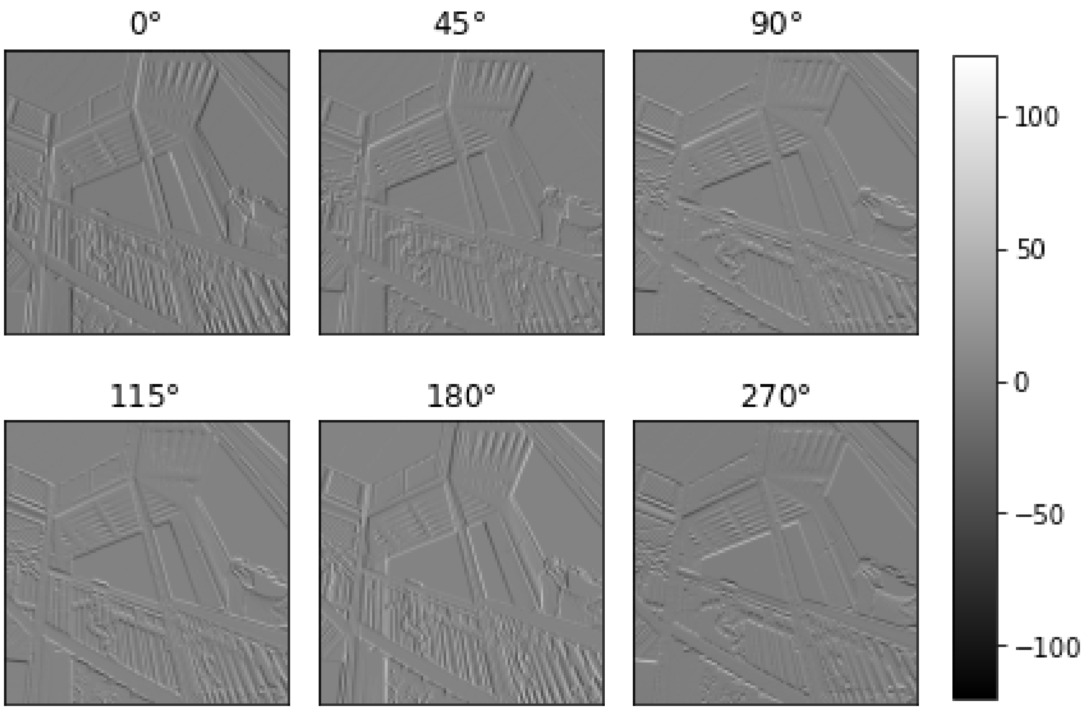
```
            deriv_theta[i,j] = np.cos(theta)*deriv_x[i,j] + np.
↪sin(theta)*deriv_y[i,j]
    return deriv_theta


angles = [0, 45, 90, 115, 180, 270]
theta_filters = []

fig1, axs1 = plt.subplots(2, 3, constrained_layout=True)

for i, theta in enumerate(angles):
    deriv_theta = create_theta_filter(theta*((np.pi)/(180)))
    theta_filters.append((theta, deriv_theta))
    im_plt = print_img_subplot(axs1[int(i/3),i%3], deriv_theta, "$\\theta = " +␣
↪str(theta) + "\degree$", False)

fig1.colorbar(im_plt, ax=axs1, location='right', shrink=0.9)
plt.show()
```



### 5.0.3 Results

[94]:
```
img = cv2.imread("./hw2_data/ascent.jpg", 0)
img = np.float64(img)
my_gfilter = cv2.filter2D(img, -1, my_gg)

fig1, axs1 = plt.subplots(1, 2, constrained_layout=True)

print_img_subplot(axs1[0], img, "Original", True)
im_plt = print_img_subplot(axs1[1], my_gfilter, "$I*g$", True)

fig1.colorbar(im_plt, ax=axs1, location='right', shrink=0.6)
plt.show()

fig2, axs2 = plt.subplots(2, 3, constrained_layout=True)

for i, theta_filter in enumerate(theta_filters):
    res = cv2.filter2D(my_gfilter, -1, theta_filter[1])
    im_plt = print_img_subplot(axs2[int(i/3),i%3], res, "$" +␣
 ↪str(theta_filter[0]) + "\degree$", True)

fig2.colorbar(im_plt, ax=axs2, location='right', shrink=0.9)
plt.show()
```






## 6 Computer Exercise 6

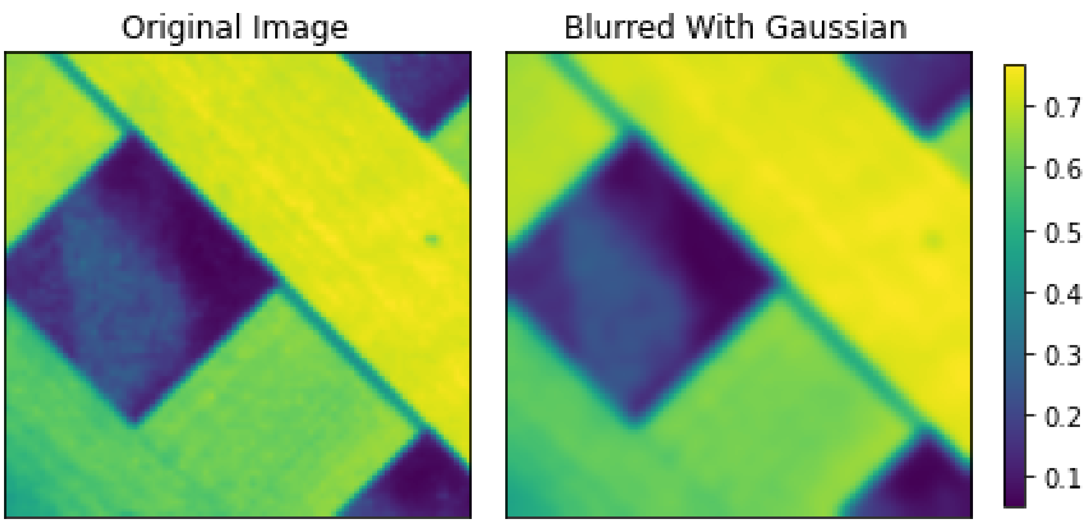
[20]:
```
import cv2
import numpy as np
from matplotlib import pyplot as plt
import copy
import scipy.io as sio
```

[21]:
```
mdict = sio.loadmat("./hw2_data/imgs_for_optical_flow.mat")
```

[22]:
```
def print_img(img, title):
    _ = plt.imshow(img, interpolation=None)
    plt.colorbar()
    plt.title(title)
    plt.xticks([]), plt.yticks([])
    plt.show()

def print_img_subplot(ax, img, title, is_gray):
    if is_gray:
        im_plt = ax.imshow(img, cmap='gray', interpolation=None)
    else:
        im_plt = ax.imshow(img, interpolation=None)
```

```
    ax.set_title(title)
    ax.set_xticks([])
    ax.set_yticks([])
    return im_plt
```

[23]:
```
img = mdict["img1"]

blurred_img = cv2.GaussianBlur(img,(5,5),0)

fig1, axs1 = plt.subplots(1, 2, constrained_layout=True)

print_img_subplot(axs1[0], img, "Original Image", False)
im_plt = print_img_subplot(axs1[1], blurred_img, "Blurred With Gaussian", False)

fig1.colorbar(im_plt, ax=axs1, location='right', shrink=0.6)
plt.show()
```




#### 6.0.1 Getting Derivative Filters

[24]:
```
hx1, hy0 = cv2.getDerivKernels(1,0,3)
hx0, hy1 = cv2.getDerivKernels(0,1,3)
hx2, _ = cv2.getDerivKernels(2,0,3)
_, hy2 = cv2.getDerivKernels(0,2,3)
```

### 6.0.2 Results

[25]:
```
res_x1 = cv2.sepFilter2D(blurred_img,-1,hx1,hy0)
res_y1 = cv2.sepFilter2D(blurred_img,-1,hx0,hy1)
res_x2 = cv2.sepFilter2D(blurred_img,-1,hx2,hy0)
res_y2 = cv2.sepFilter2D(blurred_img,-1,hx0,hy2)
```

[26]:
```
fig2, axs2 = plt.subplots(2, 2, constrained_layout=True)

print_img_subplot(axs2[0,0], res_x1, "$I_x = I * h_x$", False)
print_img_subplot(axs2[0,1], res_y1, "$I_y = I * h_y$", False)
print_img_subplot(axs2[1,0], res_x2, "$I_{xx} = I * h_{xx}$", False)
im_plt = print_img_subplot(axs2[1,1], res_y2, "$I_{yy} = I * h_{yy}$", False)

fig2.colorbar(im_plt, ax=axs2, location='right', shrink=1.0)
plt.show()
```