Homework Signal 2

Week 2

6733172621 Patthadon Phengpinij

Collaborators. ChatGPT (for LaTeX styling and grammar checking)

1 Convolution

Problem 1. Evaluate the convolution of the following signals

1.1 $\text{rect}\left(\frac{t-a}{a}\right) * \delta(t-b)$

Solution. From the sifting property of the delta function, we have:

$$f(t) * \delta(t-b) = f(t-b)$$

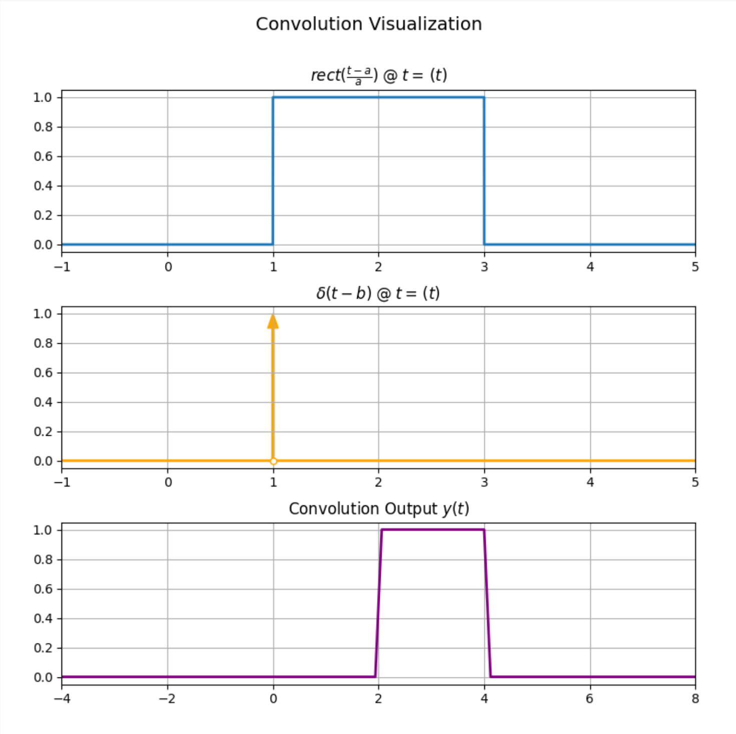
Applying this property to our problem, we get:

$$\text{rect}\left(\frac{t-a}{a}\right) * \delta(t-b) = \text{rect}\left(\frac{(t-b)-a}{a}\right) = \text{rect}\left(\frac{t-(a+b)}{a}\right)$$

Thus, the result of the convolution is:

$$\boxed{\text{rect}\left(\frac{t-(a+b)}{a}\right)}$$

Using Python to verify this result, we can implement the convolution and plot the results. The plot of the signal is shown below:



1.2 $\text{rect}\left(\frac{t}{a}\right) * \text{rect}\left(\frac{t}{a}\right)$

Solution. To evaluate the convolution of two rectangular functions, we start with the definition of the rectangular function:

$$\text{rect}\left(\frac{t}{a}\right) = \begin{cases} 1 & \text{if } |t| \leq \frac{a}{2} \\ 0 & \text{otherwise} \end{cases}$$

The convolution of two functions $f(t)$ and $g(t)$ is defined as:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)\,d\tau$$

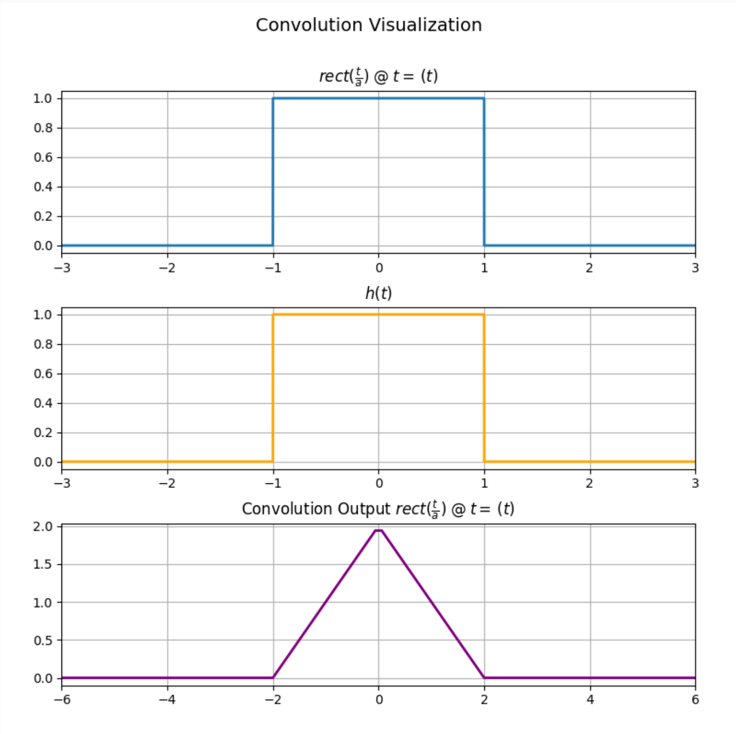
Applying this to our rectangular functions, we have:

$$\begin{aligned}
(\text{rect}\left(\frac{t}{a}\right) * \text{rect}\left(\frac{t}{a}\right))(t) &= \int_{-\infty}^{\infty} \text{rect}\left(\frac{\tau}{a}\right)\text{rect}\left(\frac{t-\tau}{a}\right)\,d\tau \\
&= \int_{-\frac{a}{2}}^{\frac{a}{2}} \text{rect}\left(\frac{t-\tau}{a}\right)\,d\tau \\
&= \int_{\max(-\frac{a}{2},t-\frac{a}{2})}^{\min(\frac{a}{2},t+\frac{a}{2})} 1\,d\tau \\
(\text{rect}\left(\frac{t}{a}\right) * \text{rect}\left(\frac{t}{a}\right))(t) &= \min\left(\frac{a}{2}, t+\frac{a}{2}\right) - \max\left(-\frac{a}{2}, t-\frac{a}{2}\right)
\end{aligned}$$

Evaluating the limits, we find that the result is a triangular function:

$$\boxed{\text{rect}\left(\frac{t}{a}\right) * \text{rect}\left(\frac{t}{a}\right) = \begin{cases} 0 & |t| > a \\ t+a & -a \leq t < 0 \\ a-t & 0 \leq t \leq a \end{cases}}$$

Using Python to verify this result, we can implement the convolution and plot the results. The plot of the signal is shown below:



1.3 $t[u(t) - u(t-1)] * u(t)$

Solution. First, we define the functions involved in the convolution:

$$x(t) = t[u(t) - u(t-1)] = \begin{cases} 0 & t < 0 \\ t & 0 \leq t < 1 \\ 0 & t \geq 1 \end{cases}$$

$$u(t) = \begin{cases} 0 & t < 0 \\ 1 & t \geq 0 \end{cases}$$

The convolution $y(t) = x(t) * u(t)$ is given by:

$$y(t) = \int_{-\infty}^{\infty} x(\tau)u(t-\tau)\,d\tau$$

Evaluating the convolution integral, we find:

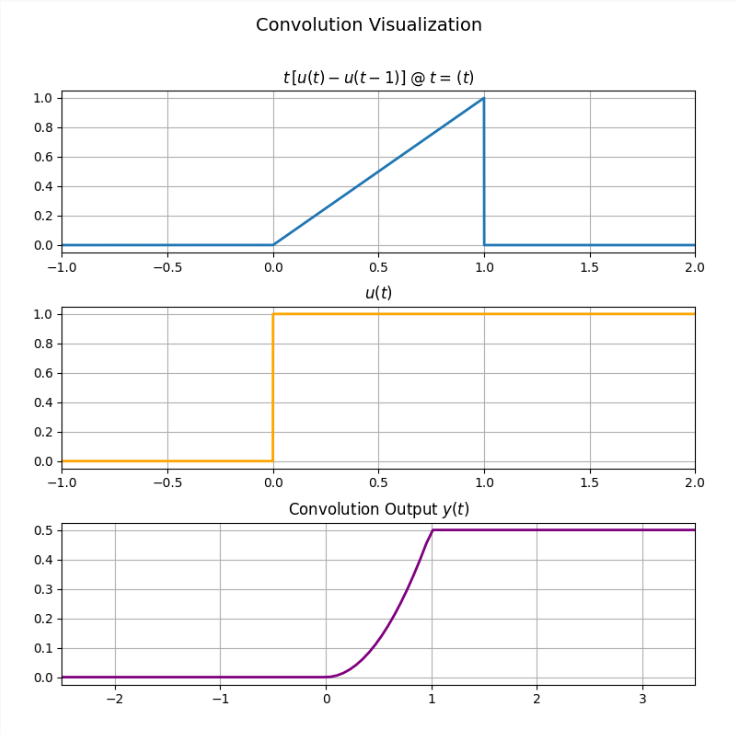
$$y(t) = \int_{0}^{1} \tau \cdot u(t-\tau)\,d\tau$$

$$y(t) = \int_{0}^{\min(t,1)} \tau\,d\tau$$

Thus,

$$\boxed{y(t) = \begin{cases} 0 & t < 0 \\ \frac{t^2}{2} & 0 \leq t < 1 \\ \frac{1}{2} & t \geq 1 \end{cases}}$$

Using Python to verify this result, we can implement the convolution and plot the results. The plot of the signal is shown below:



Problem 2. Determine the convolution $y(t) = h(t) * x(t)$ using Graphical Interpretation of the pairs of the signals shown

Solution. The convolution $y(t) = h(t) * x(t)$ can be determined graphically by following these steps:

1. Flip one of the signals, typically $h(t)$, to get $h(-\tau)$.
2. Shift the flipped signal by t to get $h(t - \tau)$.
3. For each value of t, calculate the area of overlap between $x(\tau)$ and $h(t - \tau)$.
4. The value of the convolution $y(t)$ at each t is the area of overlap calculated in the previous step.

Using Python to visualize and compute the convolution graphically, we can create an animation that demonstrates the convolution process step-by-step.

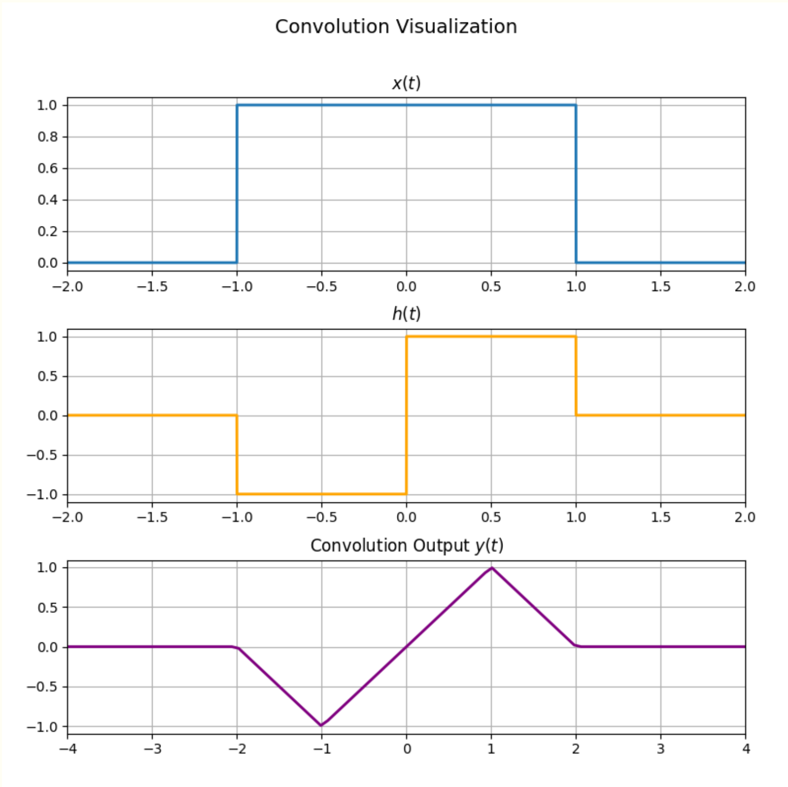
The resulting convolution $y(t)$ is shown in the gif files in my GitHub repository for this homework.

TO SUBMIT

2.1 **Solution.**

Using Python to visualize and compute the convolution graphically, we can create an animation that demonstrates the convolution process step-by-step as shown in the gif files in Problem 2.1 Animation.

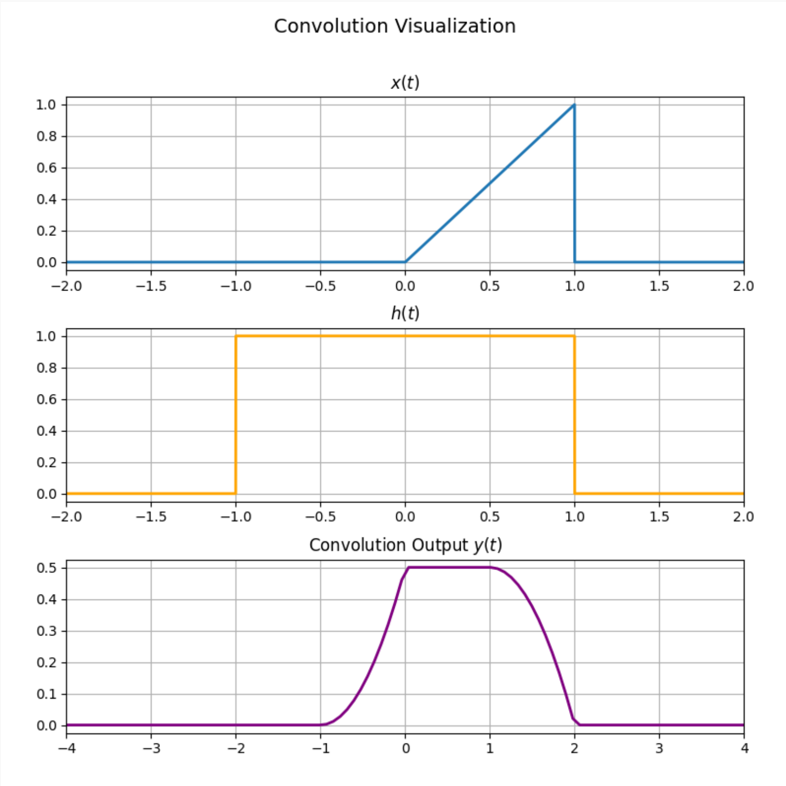
The plot of the signal is shown below:



2.2

Solution. Using Python to visualize and compute the convolution graphically, we can create an animation that demonstrates the convolution process step-by-step as shown in the gif files in Problem 2.2 Animation.

The plot of the signal is shown below:

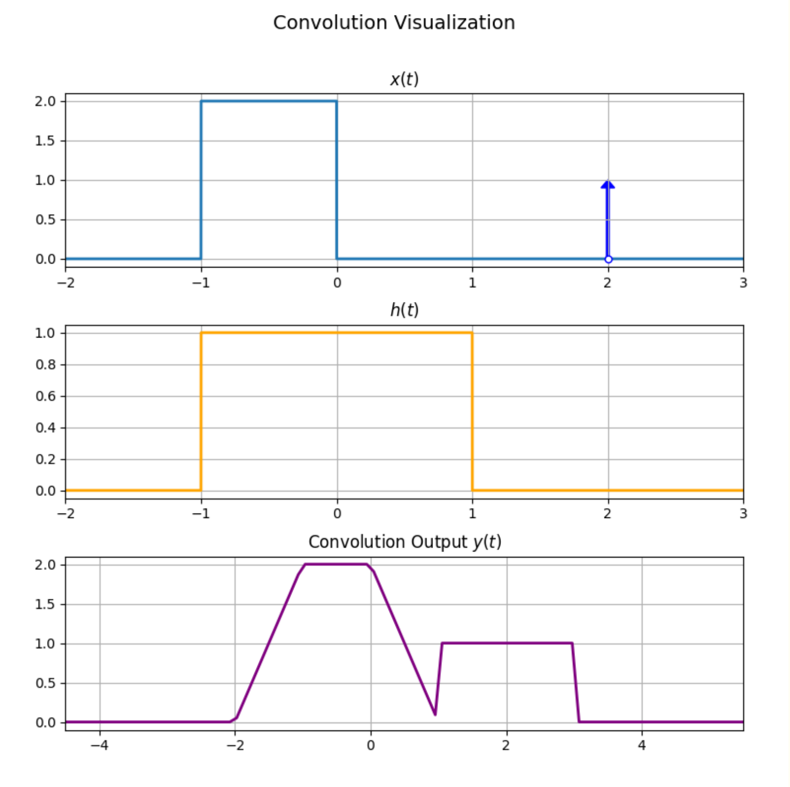


TO SUBMIT

2.3 **Solution.**

Using Python to visualize and compute the convolution graphically, we can create an animation that demonstrates the convolution process step-by-step as shown in the gif files in Problem 2.3 Animation.

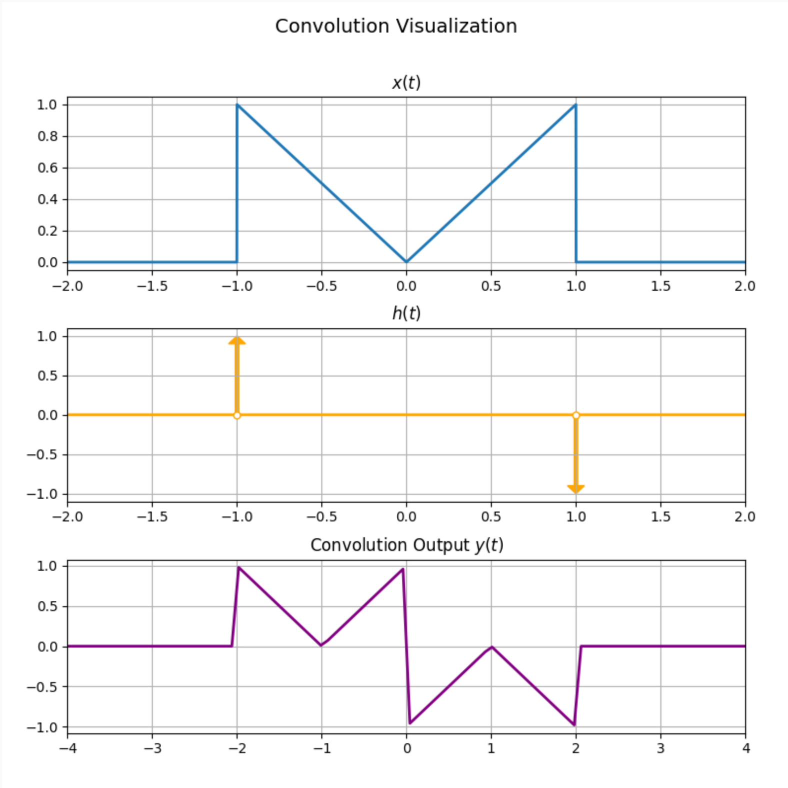
The plot of the signal is shown below:



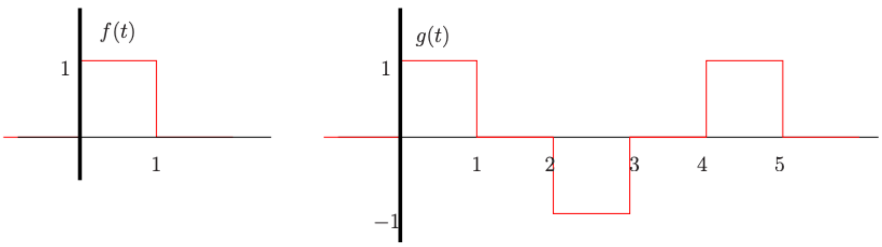
2.4

Solution. Using Python to visualize and compute the convolution graphically, we can create an animation that demonstrates the convolution process step-by-step as shown in the gif files in Problem 2.4 Animation.

The plot of the signal is shown below:



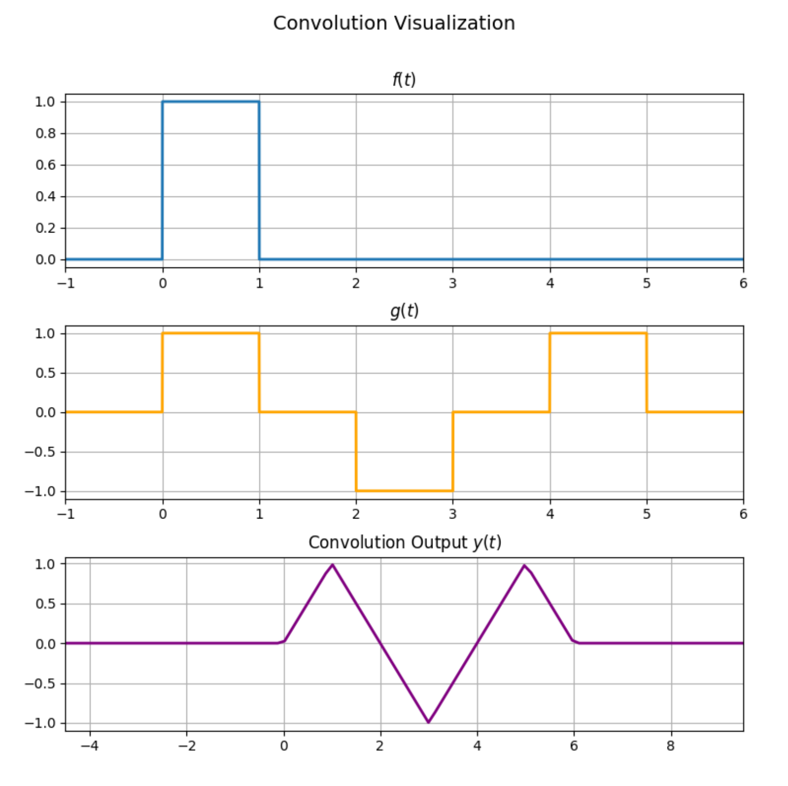
Problem 3. Let $f(t)$ and $g(t)$ be given as follows:



3.1 Sketch the function : $x(t) = f(t) * g(t)$

Solution. Using Python to visualize and compute the convolution graphically, we can create an animation that demonstrates the convolution process step-by-step as shown in the gif files in Problem 3.1 Animation.

The plot of the signal is shown below:



3.2 Show that if $a(t) = b(t) * c(t)$, then $(Mb(t)) * c(t) = Ma(t)$, for any real number M (hint: use the convolution integral formula)

Solution. Given that $a(t) = b(t) * c(t)$, we can express this using the convolution integral:

$$a(t) = \int_{-\infty}^{\infty} b(\tau)c(t-\tau)\,d\tau$$

Now, we want to show that $(Mb(t)) * c(t) = Ma(t)$. We start by writing the convolution of $Mb(t)$ with $c(t)$:

$$(Mb(t)) * c(t) = \int_{-\infty}^{\infty} Mb(\tau)c(t-\tau)\,d\tau$$

Factoring out the constant M from the integral, we have:

$$(Mb(t)) * c(t) = M\int_{-\infty}^{\infty} b(\tau)c(t-\tau)\,d\tau$$

$$(Mb(t)) * c(t) = Ma(t)$$

Thus, we have shown that:

$$\boxed{(Mb(t)) * c(t) = Ma(t)}$$

Problem 4. Find the convolution $y[n] = h[n] * x[n]$ of the following signals:

TO SUBMIT

4.1 $x[n] = \begin{cases} -1, -5 \leq n \leq -1 \\ 1, 0 \leq n \leq 4 \end{cases}$, $h[n] = 2u[n]$

Solution. Using the property of convolution with unit step function (Running sum):

$$y[n] = x[n] * u[n] = \sum_{k=-\infty}^{n} x[k]$$

Consider the value of $y[n]$:

$$\begin{aligned} y[n] &= x[n] * h[n] \\ &= x[n] * 2u[n] \\ y[n] &= 2 \sum_{k=-\infty}^{n} x[k] \end{aligned}$$

Calculating the convolution for different ranges of n:

- For $-5 \leq n < 0$:

$$\begin{aligned} y[n] &= 2 \sum_{k=-\infty}^{n} x[k] \\ &= 2 \sum_{k=-5}^{n} (-1) \\ &= 2 \cdot (-1)(n - (-5) + 1) \\ &= 2(-n - 6) \\ y[n] &= -2n - 12 \end{aligned}$$

- For $0 \leq n < 5$:

$$\begin{aligned} y[n] &= 2 \sum_{k=-\infty}^{n} x[k] \\ &= 2 \left[\sum_{k=-5}^{-1} x[k] + \sum_{k=0}^{n} x[k] \right] \\ &= 2 \left[\sum_{k=-5}^{-1} (-1) + \sum_{k=0}^{n} (1) \right] \\ &= 2\left(-5 + (n+1)\right) \\ &= 2\left(n - 4\right) \\ y[n] &= 2n - 8 \end{aligned}$$

Thus, the final result of the convolution is:

$$\boxed{y[n] = \begin{cases} -2n - 12 & -5 \leq n < 0 \\ 2n - 8 & 0 \leq n < 5 \\ 0 & \text{otherwise} \end{cases}}$$

4.2 $x[n] = u[n]$, $h[n] = 1\ ; 0 \leq n \leq 9$

Solution. Using the property of convolution with unit step function (Running sum):

$$y[n] = x[n] * u[n] = \sum_{k=-\infty}^{n} x[k]$$

Consider the value of $y[n]$:

$$\begin{aligned} y[n] &= x[n] * h[n] \\ &= u[n] * h[n] \\ y[n] &= \sum_{k=-\infty}^{n} h[k] \end{aligned}$$

Calculating the convolution for different ranges of n:

- For $0 \leq n < 9$:

$$\begin{aligned} y[n] &= \sum_{k=-\infty}^{n} h[k] \\ &= \sum_{k=0}^{n} 1 \\ y[n] &= n + 1 \end{aligned}$$

- For $n \geq 9$:

$$\begin{aligned} y[n] &= \sum_{k=-\infty}^{n} h[k] \\ &= \sum_{k=0}^{9} 1 \\ y[n] &= 10 \end{aligned}$$

Thus, the final result of the convolution is:

$$\boxed{y[n] = \begin{cases} n+1 & 0 \leq n < 9 \\ 10 & n \geq 9 \\ 0 & \text{otherwise} \end{cases}}$$

TO SUBMIT

4.3 $x[n] = \left(\frac{1}{2}\right)^n u[n]$, $h[n] = \delta[n] + \delta[n-1] + \left(\frac{1}{3}\right)^n u[n]$

Solution. Using the property of convolution with unit step function (Running sum):

$$y[n] = x[n] * u[n] = \sum_{k=-\infty}^{n} x[k]$$

and the shifting property of convolution:

$$x[n] * \delta[n - n_0] = x[n - n_0]$$

Consider the value of $y[n]$:

$$\begin{aligned}
y[n] &= x[n] * h[n] \\
&= x[n] * \left[\delta[n] + \delta[n-1] + \left(\frac{1}{3}\right)^n u[n]\right] \\
&= (x[n] * \delta[n]) + (x[n] * \delta[n-1]) + \left(x[n] * \left(\frac{1}{3}\right)^n u[n]\right) \\
y[n] &= \left(\frac{1}{2}\right)^n u[n] + \left(\frac{1}{2}\right)^{n-1} u[n-1] + \sum_{k=-\infty}^{n} \left(\frac{1}{2}\right)^k u[k] \left(\frac{1}{3}\right)^{n-k} u[n-k]
\end{aligned}$$

Calculating the convolution for different ranges of n:

- For $n = 0$:

$$\begin{aligned}
y[n] &= \left(\frac{1}{2}\right)^0 u[0] + \left(\frac{1}{2}\right)^{0-1} u[0-1] + \sum_{k=-\infty}^{0} \left(\frac{1}{2}\right)^k u[k] \left(\frac{1}{3}\right)^{0-k} u[0-k] \\
&= 1 + 0 + \left(\frac{1}{2}\right)^0 u[0] \left(\frac{1}{3}\right)^0 u[0] \\
&= 1 + 0 + 1 \\
y[n] &= 2
\end{aligned}$$

- For $n \geq 1$:

$$\begin{aligned}
y[n] &= \left(\frac{1}{2}\right)^n u[n] + \left(\frac{1}{2}\right)^{n-1} u[n-1] + \sum_{k=-\infty}^{n} \left(\frac{1}{2}\right)^k u[k] \left(\frac{1}{3}\right)^{n-k} u[n-k] \\
&= \left(\frac{1}{2}\right)^n + \left(\frac{1}{2}\right)^{n-1} + \sum_{k=0}^{n} \left(\frac{1}{2}\right)^k \left(\frac{1}{3}\right)^{n-k} \\
&= 3\left(\frac{1}{2}\right)^n + \left(\frac{1}{3}\right)^n \left[\sum_{k=0}^{n} \left(\frac{3}{2}\right)^k\right] \\
&= 3\left(\frac{1}{2}\right)^n + \left(\frac{1}{3}\right)^n \cdot (-2)\left(1 - \left(\frac{3}{2}\right)^{n+1}\right) \\
y[n] &= 6\left(\frac{1}{2}\right)^n - 2\left(\frac{1}{3}\right)^n
\end{aligned}$$

Thus, the final result of the convolution is:

$$\boxed{y[n] = \begin{cases} 2 & n = 0 \\ 6\left(\frac{1}{2}\right)^n - 2\left(\frac{1}{3}\right)^n & n \geq 1 \\ 0 & \text{otherwise} \end{cases}}$$

4.4 $x[n] = \left(\frac{1}{3}\right)^n u[n]$, $h[n] = \delta[n] + \left(\frac{1}{2}\right)^n u[n]$

Solution. Using the property of convolution with unit step function (Running sum):

$$y[n] = x[n] * u[n] = \sum_{k=-\infty}^{n} x[k]$$

the shifting property of convolution:

$$x[n] * \delta[n - n_0] = x[n - n_0]$$

Consider the value of $y[n]$:

$$\begin{aligned}
y[n] &= x[n] * h[n] \\
&= x[n] * \left[\delta[n] + \left(\frac{1}{2}\right)^n u[n]\right] \\
&= \left(\left(\frac{1}{3}\right)^n u[n] * \delta[n]\right) + \left(\left(\frac{1}{3}\right)^n u[n] * \left(\frac{1}{2}\right)^n u[n]\right) \\
y[n] &= \left(\frac{1}{3}\right)^n u[n] + \sum_{k=-\infty}^{\infty} \left(\frac{1}{3}\right)^k u[k] \left(\frac{1}{2}\right)^{n-k} u[n-k] \\
y[n] &= \left(\frac{1}{3}\right)^n u[n] + \sum_{k=0}^{n} \left(\frac{1}{3}\right)^k \left(\frac{1}{2}\right)^{n-k} \\
y[n] &= \left(\frac{1}{3}\right)^n u[n] + \sum_{k=0}^{n} \left(\frac{1}{3}\right)^k \left(\frac{1}{2}\right)^{n-k}
\end{aligned}$$

Calculating the convolution for different ranges of n:

- For $n \geq 0$:

$$\begin{aligned}
y[n] &= \left(\frac{1}{3}\right)^n u[n] + \sum_{k=-\infty}^{\infty} \left(\frac{1}{3}\right)^k u[k] \left(\frac{1}{2}\right)^{n-k} u[n-k] \\
&= \left(\frac{1}{3}\right)^n + \left(\frac{1}{2}\right)^n \left[\sum_{k=0}^{n} \left(\frac{2}{3}\right)^k\right] \\
&= \left(\frac{1}{3}\right)^n + \left(\frac{1}{2}\right)^n \cot(3) \left(1 - \left(\frac{2}{3}\right)^{n+1}\right) \\
y[n] &= 3\left(\frac{1}{2}\right)^n - \left(\frac{1}{3}\right)^n
\end{aligned}$$

Thus, the final result of the convolution is:

$$\boxed{y[n] = \begin{cases} 3\left(\frac{1}{2}\right)^n - \left(\frac{1}{3}\right)^n & n \geq 0 \\ 0 & \text{otherwise} \end{cases}}$$

Problem 5. Find the convolution $y[n] = h[n] * x[n]$ of the following signals

5.1 $x[n] = \left\{1, -\frac{1}{2}, \frac{1}{4}, -\frac{1}{8}, \frac{1}{16}\right\}$, $h[n] = \{1, -1, 1, -1\}$

Solution. Using the tabular method to compute the convolution $y[n] = h[n] * x[n]$:

n	-3	-2	-1	0	1	2	3	4	5	6	7	$y[n]$
$x[n]$				1	$-\frac{1}{2}$	$\frac{1}{4}$	$-\frac{1}{8}$	$\frac{1}{16}$				
$h[n]$				1	-1	1	-1					
$h[0-n]$	-1	1	-1	1								1.0000
$h[1-n]$		-1	1	-1	1							-1.5000
$h[2-n]$			-1	1	-1	1						1.7500
$h[3-n]$				-1	1	-1	1					-1.8750
$h[4-n]$					-1	1	-1	1				0.9375
$h[5-n]$						-1	1	-1	1			-0.4375
$h[6-n]$							-1	1	-1	1		0.1875
$h[7-n]$								-1	1	-1	1	-0.0625

Thus, the final result of the convolution is:

$$\boxed{y[n] = \{1, -1.5, 1.75, -1.875, 0.9375, -0.4375, 0.1875, -0.0625\}}$$

5.2 $x[n] = \{1, 2, 3, 0, -1\}$, $h[n] = \{2, -1, 3, 1, -2\}$

Solution. Using the tabular method to compute the convolution $y[n] = h[n] * x[n]$:

n	-4	-3	-2	-1	0	1	2	3	4	5	6	7	8	$y[n]$
$x[n]$					1	2	3	0	-1					
$h[n]$					2	-1	3	1	-2					
$h[0-n]$	-2	1	3	-1	2									2
$h[1-n]$		-2	1	3	-1	2								3
$h[2-n]$			-2	1	3	-1	2							7
$h[3-n]$				-2	1	3	-1	2						4
$h[4-n]$					-2	1	3	-1	2					7
$h[5-n]$						-2	1	3	-1	2				0
$h[6-n]$							-2	1	3	-1	2			-9
$h[7-n]$								-2	1	3	-1	2		-1
$h[8-n]$									-2	1	3	-1	2	2

Thus, the final result of the convolution is:

$$\boxed{y[n] = \{2, 3, 7, 4, 7, 0, -9, -1, 2\}}$$

5.3 $x[n] = \left\{3, \frac{1}{2}, -\frac{1}{4}, 1, 4\right\}$, $h[n] = \left\{2, -1, \frac{1}{2}, -\frac{1}{2}\right\}$

Solution. Using the tabular method to compute the convolution $y[n] = h[n] * x[n]$:

n	-3	-2	-1	0	1	2	3	4	5	6	7	$y[n]$
$x[n]$				3	$\frac{1}{2}$	$-\frac{1}{4}$	1	4				
$h[n]$				2	-1	$\frac{1}{2}$	$-\frac{1}{2}$					
$h[0-n]$	$-\frac{1}{2}$	$\frac{1}{2}$	-1	2								6.000
$h[1-n]$		$-\frac{1}{2}$	$\frac{1}{2}$	-1	2							-2.000
$h[2-n]$			$-\frac{1}{2}$	$\frac{1}{2}$	-1	2						0.500
$h[3-n]$				$-\frac{1}{2}$	$\frac{1}{2}$	-1	2					1.000
$h[4-n]$					$-\frac{1}{2}$	$\frac{1}{2}$	-1	2				6.625
$h[5-n]$						$-\frac{1}{2}$	$\frac{1}{2}$	-1	2			-3.375
$h[6-n]$							$-\frac{1}{2}$	$\frac{1}{2}$	-1	2		1.500
$h[7-n]$								$-\frac{1}{2}$	$\frac{1}{2}$	-1	2	-2.000

Thus, the final result of the convolution is:

$$\boxed{y[n] = \{6, -2, 0.5, 1, 6.625, -3.375, 1.5, -2\}}$$

5.4 $x[n] = \left\{-1, \frac{1}{2}, \frac{3}{4}, -\frac{1}{5}, 1\right\}$, $h[n] = \{1, 1, 1, 1, 1\}$

Solution. Using the tabular method to compute the convolution $y[n] = h[n] * x[n]$:

n	-4	-3	-2	-1	0	1	2	3	4	5	6	7	8	$y[n]$
$x[n]$					-1	$\frac{1}{2}$	$\frac{3}{4}$	$-\frac{1}{5}$	1					
$h[n]$					1	1	1	1	1					
$h[0-n]$	1	1	1	1	1									-1.00
$h[1-n]$		1	1	1	1	1								0.50
$h[2-n]$			1	1	1	1	1							0.25
$h[3-n]$				1	1	1	1	1						0.05
$h[4-n]$					1	1	1	1	1					1.05
$h[5-n]$						1	1	1	1	1				2.05
$h[6-n]$							1	1	1	1	1			1.55
$h[7-n]$								1	1	1	1	1		0.80
$h[8-n]$									1	1	1	1	1	1.00

Thus, the final result of the convolution is:

$$\boxed{y[n] = \{-1, -0.5, 0.25, 0.05, 1.05, 2.05, 1.55, 0.8, 1\}}$$

Note that all of the convolutions in Problem 5 are also available in the gif files in this repository.

Problem 6.

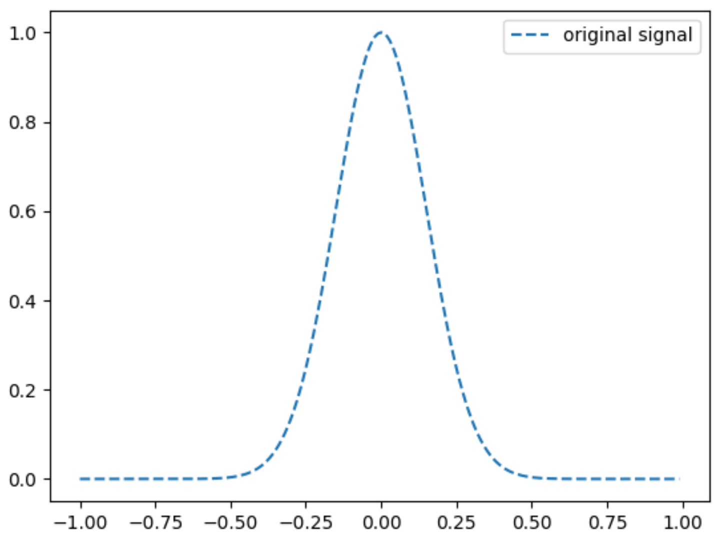
6.1 **Convolution 1-D:**

The following code creates a gaussian pulse and its self convolutions. Study and apply the convolution between signal `e` and another signal `e` with noise (`e_noise`) and write the report to analyze the results.

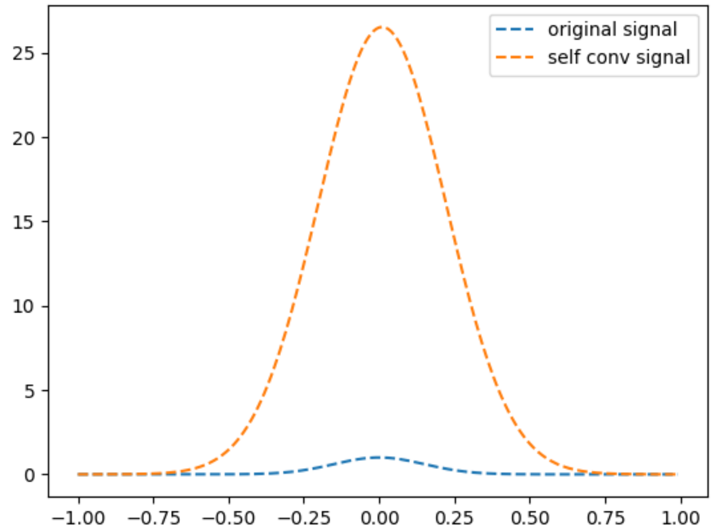
```
from scipy import signal

t = np.linspace(-1, 1, 2 * 100, endpoint=False)
i, q, e = signal.gausspulse(t, fc=5, retquad=True, retenv=
    True)
plt.plot(t, e, "--",label = "original signal")
plt.legend(loc="upper right")
plt.show()

conv_e = np.convolve(e,e,"same")
plt.plot(t, e, "--",label = "original signal")
plt.plot(t, conv_e, "--",label = "self conv signal")
plt.legend(loc="upper right")
plt.show()

e_noise = e + np.random.randn(len(e))*2.5
conv_e_noise = np.convolve(e, e_noise,"same")

# TODO : Apply the convolution between signal e and another
    signal e with noise (e_noise) and check the results
```

Results:

1. Original Signal `e`:



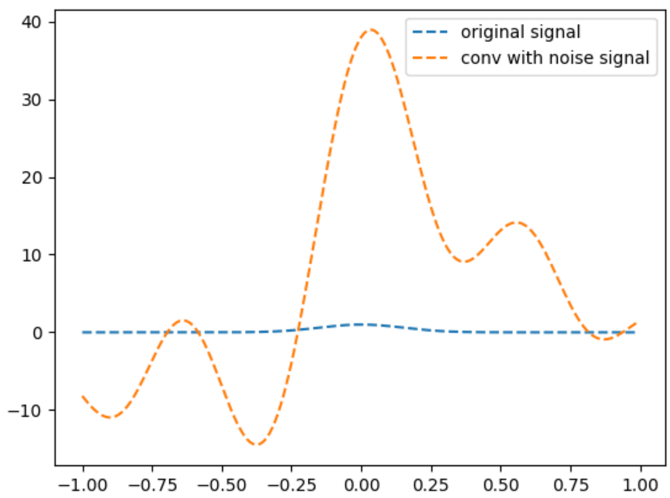
2. Convolution between signal `e` and signal `e`:



Solution. To apply the convolution between the signal e and the noisy signal e_noise, we can use the following code snippet:

```
conv_e_noise = np.convolve(e, e_noise, "same")
plt.plot(t, e, "--", label="original signal")
plt.plot(t, conv_e_noise, "--", label="conv with noise
    signal")
plt.legend(loc="upper right")
plt.show()
```

The result of the convolution between the original signal e and the noisy signal e_noise is shown below:



6.2 From the self convolution below, when increasing the number of self convolution (now is 8), what is noticeable from the final shape resulted from the convolution?

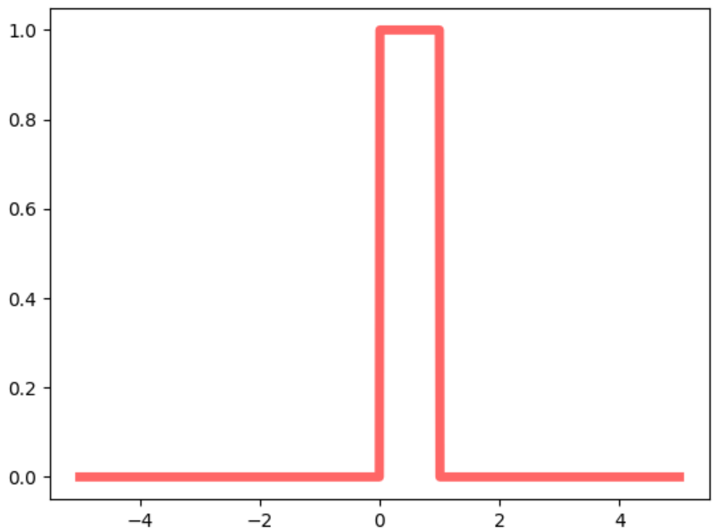
(**HINT 01:** Central limit theorem)

(**HINT 02:** What is Probability Density Function (PDF) of z if $z = x + y$?)

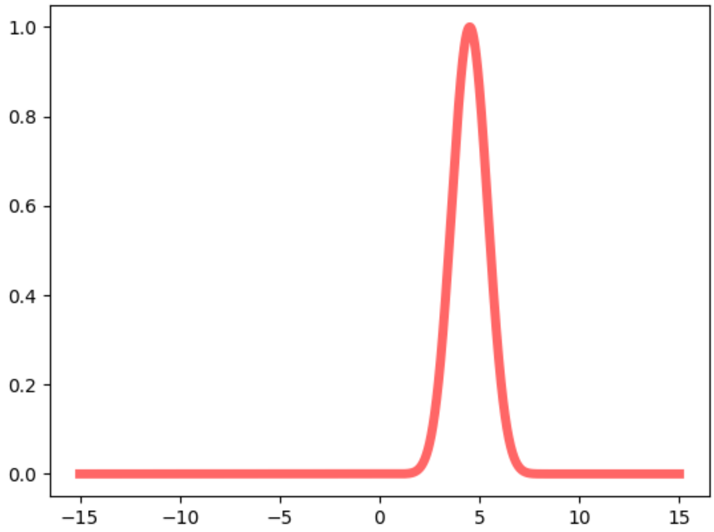
```
from scipy.stats import uniform

x = np.linspace(-5,5, 1000)
plt.plot(x, uniform.pdf(x), "r-", lw=5, alpha=0.6, label="
    uniform pdf")
plt.show()

x = np.linspace(-15,15, 10000)
pdf_1 = uniform.pdf(x)
pdf_2 = uniform.pdf(x)

for i in range(8):
    pdf_2 = np.convolve(pdf_2,pdf_1, "same")

pdf_2 = pdf_2/np.max(pdf_2)
plt.plot(x, pdf_2,"r-", lw=5, alpha=0.6, label="conv
    uniform")
plt.show()
```

Results:

1. Original Uniform PDF:



2. Resulted PDF after 8 times of self convolution:



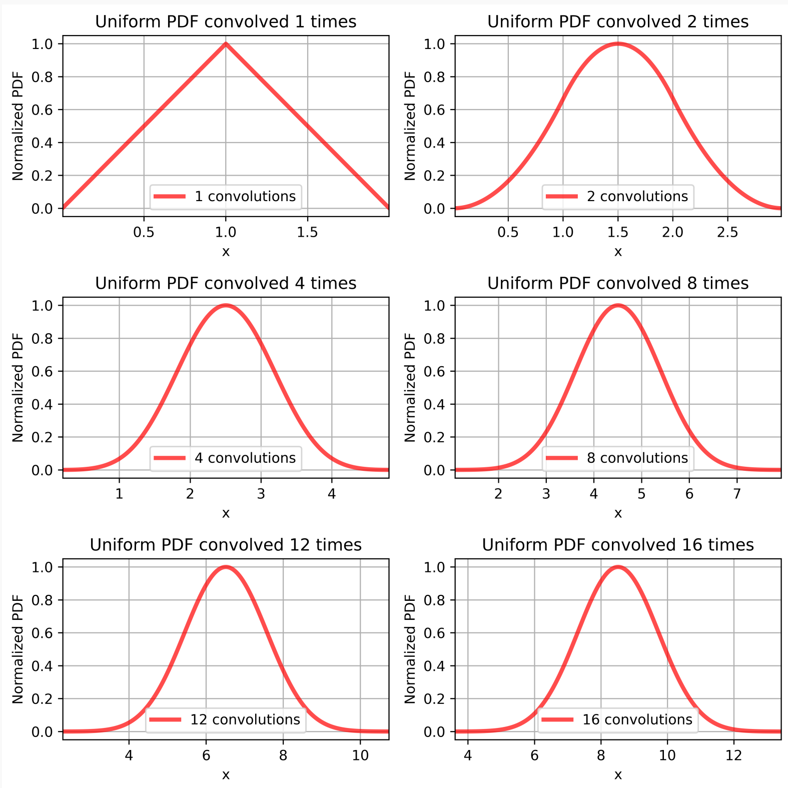
Solution. Firstly, we create a function to compute the PDF of a uniform distribution after specified number of self-convolutions. The code snippet is as follows:

```
def plot_uniform_convolution(ax, num_convolutions=1):
    x = np.linspace(-15, 15, 10000)
    pdf = uniform.pdf(x)

    conv_pdf = pdf.copy()
    for _ in range(num_convolutions):
        conv_pdf = np.convolve(conv_pdf, pdf, mode="same"
    )

    conv_pdf = conv_pdf / np.max(conv_pdf)

    ax.plot(x, conv_pdf, "r-", lw=3, alpha=0.7, label=f"{
    num_convolutions} convolutions")
    ax.set_title(f"Uniform PDF convolved {
    num_convolutions} times")
    ax.set_xlabel("x")
    ax.set_ylabel("Normalized PDF")
    ax.legend()
    ax.grid(True)

    # Automatically adjust x-limits based on significant
    values
    threshold = 1e-4
    significant_indices = np.where(conv_pdf > threshold)
    [0]
    ax.set_xlim(x[significant_indices[0]], x[
    significant_indices[-1]])
```

Then, we can visualize the effect of multiple self-convolutions on the uniform PDF.

```python
np.random.seed(1)  # For reproducibility

Ns = [1, 2, 4, 8, 12, 16]

# Create a 3x2 subplot grid
fig, axes = plt.subplots(3, 2, figsize=(8, 8))
axes = axes.flatten()  # Flatten to make indexing easier

for i, N in enumerate(Ns):
    plot_uniform_convolution(axes[i], N)

plt.tight_layout()
plt.savefig("../images/problem_6_2_comparation.png", dpi=600, bbox_inches="tight")
plt.show()
```

The final shape resulted from the convolution approaches a Gaussian distribution as the number of self-convolutions increases. This is a direct consequence of the **Central Limit Theorem**, which states that **the sum of a large number of independent random variables**, regardless of their original distribution, will tend to follow a **normal (Gaussian) distribution**.



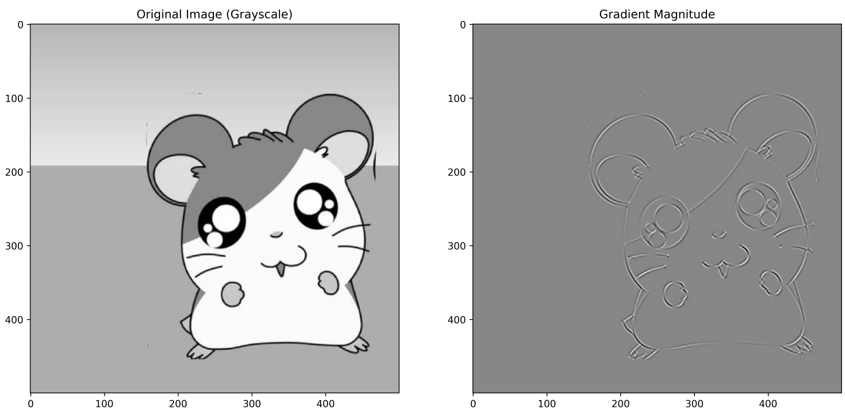
Problem 7. 2D (image) signal convolution:

The following code show the 2D signal (image $f(x, y)$) and a kernel (diag_line). Study the convolution of the kernel and the image. Apply with `"circuits.png"` image and analyze the results.

TODO : Apply diag_line to the `"circuits.png"` image and analyse the results

```python
import cv2

image_path = "hamtaro0.jpg"

diag_line = np.array([[ 2, -1, -1],
                      [-1, 2, -1],
                      [-1, -1, 2]])

ham = cv2.imread(image_path, 0)
plt.figure(figsize=(10, 10))
plt.imshow(ham, cmap="gray")
plt.show()

grad = signal.convolve2d(ham, diag_line, boundary="symm", mode
    ="same")
plt.figure(figsize=(10, 10))
plt.imshow(grad, cmap="gray")
plt.show()

# TODO : Apply diag_line to the "circuits.png" image and
    analyse the results
```

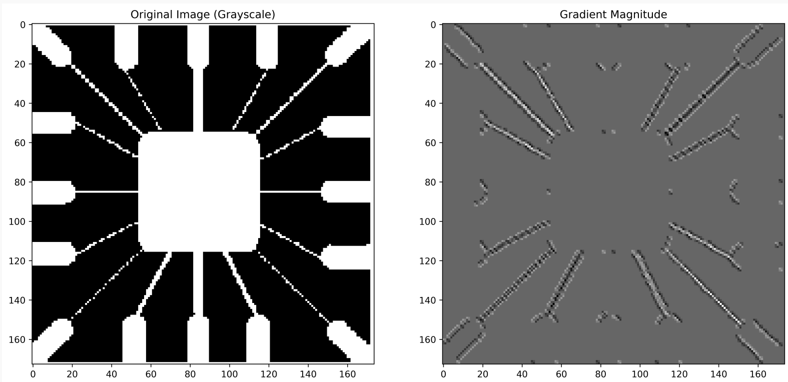
Results:



Solution. To apply the convolution of the kernel `diag_line` to the image `"circuits.png"`, we can use the following code snippet:

```
image_path = "../images/problem_7_circuits.png"

diag_line = np.array([[ 2, -1, -1],
                      [-1, 2, -1],
                      [-1, -1, 2]])

circuits = cv2.imread(image_path, 0)
plt.figure(figsize=(10, 10))
plt.imshow(circuits, cmap="gray")
plt.show()

grad = signal.convolve2d(circuits, diag_line, boundary="symm", mode="same")
plt.figure(figsize=(10, 10))
plt.imshow(grad, cmap="gray")
plt.show()
```

The result of the convolution between the image `"circuits.png"` and the kernel `diag_line` is shown below:



Problem 8. Are the following systems linear or time invariant?

8.1 $x(t) \rightarrow \textbf{System(a)} \rightarrow 7x(t-1)$

Solution.

1. Check for linearity: Let $x_1(t)$ and $x_2(t)$ be two input signals, consider,

$$\begin{aligned} S\{ax_1(t) + bx_2(t)\} &= 7[ax_1(t-1) + bx_2(t-1)] \\ &= 7ax_1(t-1) + 7bx_2(t-1) \\ &= a[7x_1(t-1)] + b[7x_2(t-1)] \\ S\{ax_1(t) + bx_2(t)\} &= aS\{x_1(t)\} + bS\{x_2(t)\} \end{aligned}$$

Since both sides are equal, the system is linear.

2. Check for time invariance: Let $x(t)$ be an input signal and t_0 be a time shift, consider,

$$\begin{aligned} S\{x(t-t_0)\} &= 7x((t-t_0)-1) \\ &= 7x(t-t_0-1) \\ &= 7x((t-1)-t_0) \\ \implies S\{x(t-t_0)\} &= y(t-t_0) \end{aligned}$$

Since both sides are equal, the system is time-invariant.
Therefore, the system $S\{x(t)\} = 7x(t-1)$ is $\boxed{\text{both linear and time-invariant}}$.

8.2 $x(t) \rightarrow \textbf{System(b)} \rightarrow \cos(2x(t))$

Solution.

1. Check for linearity: Let $x_1(t)$ and $x_2(t)$ be two input signals, consider,

$$\begin{aligned} \cos(2[ax_1(t) + bx_2(t)]) &\neq a[\cos(2x_1(t))] + b[\cos(2x_2(t))] \\ \implies S\{ax_1(t) + bx_2(t)\} &= aS\{x_1(t)\} + bS\{x_2(t)\} \end{aligned}$$

Since both sides are not equal, the system is non-linear.

2. Check for time invariance: Let $x(t)$ be an input signal and t_0 be a time shift, consider,

$$\begin{aligned} S\{x(t-t_0)\} &= \cos(2[x(t-t_0)]) \\ &= \cos(2x(t-t_0)) \\ \implies S\{x(t-t_0)\} &= y(t-t_0) \end{aligned}$$

Since both sides are equal, the system is time-invariant.
Therefore, the system $S\{x(t)\} = \cos(2x(t))$ is $\boxed{\text{non-linear but time-invariant}}$.

8.3 $x(t) \rightarrow$ **System(c)** $\rightarrow t$

Solution.

1. Check for linearity: Let $x_1(t)$ and $x_2(t)$ be two input signals, consider,

$$\begin{aligned} t &\neq (a+b)t \\ &= a \cdot t + b \cdot t \\ \implies S\{ax_1(t) + bx_2(t)\} &\neq aS\{x_1(t)\} + bS\{x_2(t)\} \end{aligned}$$

Since both sides are not equal, the system is non-linear.

2. Check for time invariance: Let $x(t)$ be an input signal and t_0 be a time shift, consider,

$$\begin{aligned} t &\neq t - t_0 \\ \implies S\{x(t-t_0)\} &\neq y(t-t_0) \end{aligned}$$

Since both sides are not equal, the system is time-variant.
Therefore, the system $S\{x(t)\} = t$ is $\boxed{\text{both non-linear and time-variant}}$.

8.4 $x(t) \rightarrow$ **System(d)** $\rightarrow x(t) + t$

Solution.

1. Check for linearity: Let $x_1(t)$ and $x_2(t)$ be two input signals, consider,

$$\begin{aligned} ax_1(t) + bx_2(t) + t &\neq a[x_1(t) + t] + b[x_2(t) + t] \\ \implies S\{ax_1(t) + bx_2(t)\} &\neq aS\{x_1(t)\} + bS\{x_2(t)\} \end{aligned}$$

Since both sides are not equal, the system is non-linear.

2. Check for time invariance: Let $x(t)$ be an input signal and t_0 be a time shift, consider,

$$\begin{aligned} x(t-t_0) + t &\neq x(t-t_0) + t - t_0 \\ \implies S\{x(t-t_0)\} &\neq y(t-t_0) \end{aligned}$$

Since both sides are not equal, the system is time-variant.
Therefore, the system $S\{x(t)\} = x(t) + t$ is $\boxed{\text{both non-linear and time-variant}}$.