Matlab questions:

Question 1:

$$\varphi(x,y) = 2500 \cdot \sin(4\pi x)\cos(14\pi y)$$

a.

The sine and cosine functions bounded by 1 and -1, thus $\varphi(x,y)$ is bounded by 2500 and -2500 for $\varphi\left(\frac{\pi}{2\times 4},0\right) = 2500$ and for $\varphi\left(-\frac{\pi}{2\times 4},0\right) = -2500$.

Therefore $\varphi_H = 2500$ and $\varphi_L = -2500$, yielding $\varphi_H - \varphi_L = 5000$.

Now we calculate the partial derivatives of φ:

$\frac{\partial\varphi(x,y)}{\partial x} = 2500 \cdot\ 4\pi * \cos(4\pi x)\cos(14\pi y) = 10000\pi \cdot \cos(4\pi x) \cdot \cos(14\pi y)$

$\frac{\partial\varphi(x,y)}{\partial y} = 2500 \cdot 14\pi \cdot \sin(4\pi x) \cdot (-\sin(14\pi y)) = -35000\pi \cdot \sin(4\pi x) \cdot \sin(14\pi y)$

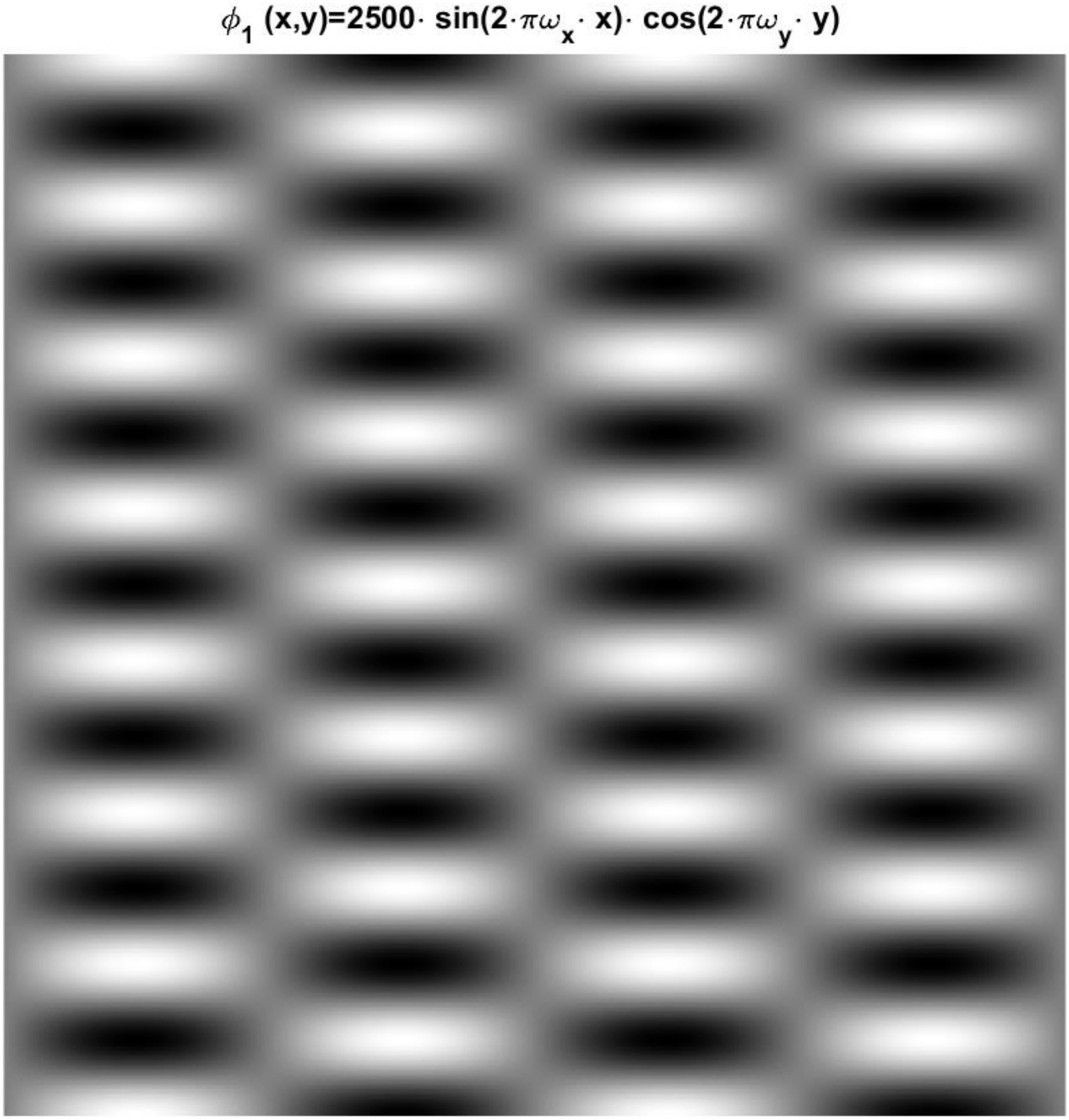
<u>Now we'll calculate the partial derivatives' energy:</u>

$Energy\left(\frac{\partial\varphi}{\partial x}\right) = \int_0^1\int_0^1\left(\frac{\partial\varphi}{\partial x}(x,y)\right)^2 dxdy\ = (10000\pi)^2 \cdot \int_0^1 \cos^2(4\pi x)dx \cdot \int_0^1 \cos^2(14\pi y)dy = (10000\pi)^2 \cdot \frac{1}{2}\int_0^1 1+\cos(8\pi x)\,dx \cdot \frac{1}{2}\int_0^1 1+\cos(28\pi y)\,dy = 25000000\pi^2 \cdot \left(x+\frac{\sin(8\pi x)}{4\pi}\right)\Big|_0^1 \cdot \left(y+\frac{\sin(28\pi y)}{4\pi}\right)\Big|_0^1 = 25000000\pi^2 \cdot (1+0)\cdot(1+0) = 25000000\pi^2$

$Energy\left(\frac{\partial\varphi}{\partial y}\right) = \int_0^1\int_0^1\left(\frac{\partial\varphi}{\partial x}(x,y)\right)^2 dxdy\ = (-35000\pi)^2 \cdot \int_0^1 \sin^2(4\pi x)dx \cdot \int_0^1 \sin^2(14\pi y)dy = (-35000\pi)^2 \cdot \frac{1}{2}\int_0^1 1-\cos(8\pi x)\,dx \cdot \frac{1}{2}\int_0^1 1-\cos(28\pi y)\,dy = (-35000\pi)^2 \cdot \frac{1}{2}\int_0^1 1-\cos(8\pi x)\,dx \cdot \frac{1}{2}\int_0^1 1-\cos(28\pi y)\,dy = 306250000\pi^2 \cdot \left(x-\frac{\sin(8\pi x)}{4\pi}\right)\Big|_0^1 \cdot \left(y-\frac{\sin(28\pi y)}{4\pi}\right)\Big|_0^1 = 306250000\pi^2 \cdot (1-0)\cdot(1-0) = 306250000\pi^2$

b.

The representation on a grayscale scaling of [-2500, 2500] is:



c.

The results are:

```
phi_H - phi_L = 5000
Energy( phi _x `) = 247230336.800774
Energy( phi _y `) = 3022079130.860223
```

The relative error:

$$\varphi_H - \varphi_L = 0\%$$
$$Energy\left(\frac{\partial\varphi}{\partial x}\right) = 0.198\%$$
$$Energy\left(\frac{\partial\varphi}{\partial y}\right) = 0.016\%$$

d.

The solver that uses the previous results and uses fmincon as was hinted is the function bitBudgetOptim (in the corresponding matlab file).

We define the MSE which is the function we want to optimize with respect to it 3 parameters: $N_x = x(1), N_y = x(2), b = x(3)$, according to the formula in the lectures and use the *fmincon* function setting the lower bound and upper bound not limited (just positive real values), and using the bit budget constraint, which requires that $N_x \cdot N_y \cdot b - B \leq 0$, which corresponds to: $B \geq N_x N_y b$ as required.

Then we choose the best rounded parameters (of all the possibilities for flooring and ceiling the floating parameters the *fmincon* found)

- **<u>A very important observation and an explanation for it:</u>** if we'll not round the parameters, we can achieve even better MSE error, better then our greedy search, and the round procedure that we choose to do, is done because of our treat to the parameters units as the smallest unit, like a pixel, so it is logical that it has to be integer. That's why in our case the *fmincon* gets a little bit worthier results than our greedy search in the lats sections.

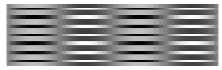
e.

for budget of 5000 are:

```
Numeric method approx. - Bit budget = 5000, Nx = 21.000000, Ny = 73.000000, b =3.000000
The MSE = 126528.217908
```

And by taking the floor values of the results we receive the following result:

Numeric approximation of bit-budget problem solution with bit-budget = 5000



The parameters for budget of 50000 are:

```
Numeric method approx. - Bit budget = 50000, Nx = 53.000000, Ny = 188.000000, b =5.000000
The MSE = 16494.369920
```

It is expected that the error is less with bigger budget (the budget is bigger by factor 10, and the MSE is smaller by factor 10 either).

Similarly, by taking the floor values of the results we receive the following result:

Numeric approximation of bit-budget problem solution with bit-budget = 50000

And as expected it is visually a better approximation.

f.

We've implemented the search procedure as follows:

We're iterating over all the possible N_x, N_y values up to B (budget) and in each iteration when those params are fixed we calculating $b = \frac{B}{N_x \cdot N_y}$, which we plugin into the MSE calculation with N_x, N_y.

Each iteration we're checking if the current MSE is smaller, and if it does, we're updating the best error and our best parameters found so far.

After the iterating is done, we're returning the best parameters which we're updated last.

g.

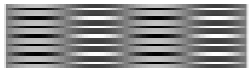
Using the search procedure for budget 5000, we've got:

```
Partical sol. - Bit budget = 5000, Nx = 21.000000, Ny = 79.000000, b =3.000000
The MSE = 119622.330057
```

Which is approximately the parameters and the MSE we've got for the budget of 5000 in the previous section.

The corresponding visualized image:

Practically found solution for bit-budget problem with B=5000



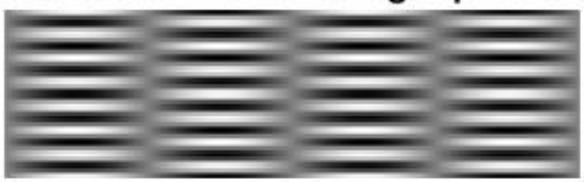
Using the search procedure for budget 50000, we've got:

```
Partical sol. - Bit budget = 50000, Nx = 54.000000, Ny = 185.000000, b =5.000000
The MSE = 16458.205704
```

Which is approximately the parameters and the MSE we've got for the budget of 5000 in the previous section.

The corresponding visualized image:

Practically found solution for bit-budget problem with B=50000



h. <u>repeating a-g items:</u>

$$\varphi(x, y) = 2500 \cdot \sin(14\pi x)\cos(4\pi y)$$

The comparasions are marked in bold and underscoped.

a.

The sine and cosine functions bounded by 1 and -1, thus $\varphi(x, y)$ is bounded by 2500 and -2500 for $\varphi\left(\frac{\pi}{2\times 4}, 0\right) = 2500$ and for $\varphi\left(-\frac{\pi}{2\times 4}, 0\right) = -2500$.

Therefore $\varphi_H = 2500$ and $\varphi_L = -2500$, yielding $\varphi_H - \varphi_L = 5000$.

<u>Same as before.</u>

Now we calculate the partial derivatives of φ:

$\frac{\partial\varphi(x,y)}{\partial x} = 2500 \cdot\ 14\pi * \cos(14\pi x)\cos(4\pi y) = 35000\pi \cdot \cos(14\pi x) \cdot \cos(4\pi y)$

$\frac{\partial\varphi(x,y)}{\partial y} = 2500 \cdot 4\pi \cdot \sin(14\pi x) \cdot (-\sin(4\pi y)) = -10000\pi \cdot \sin(14\pi x) \cdot \sin(4\pi y)$

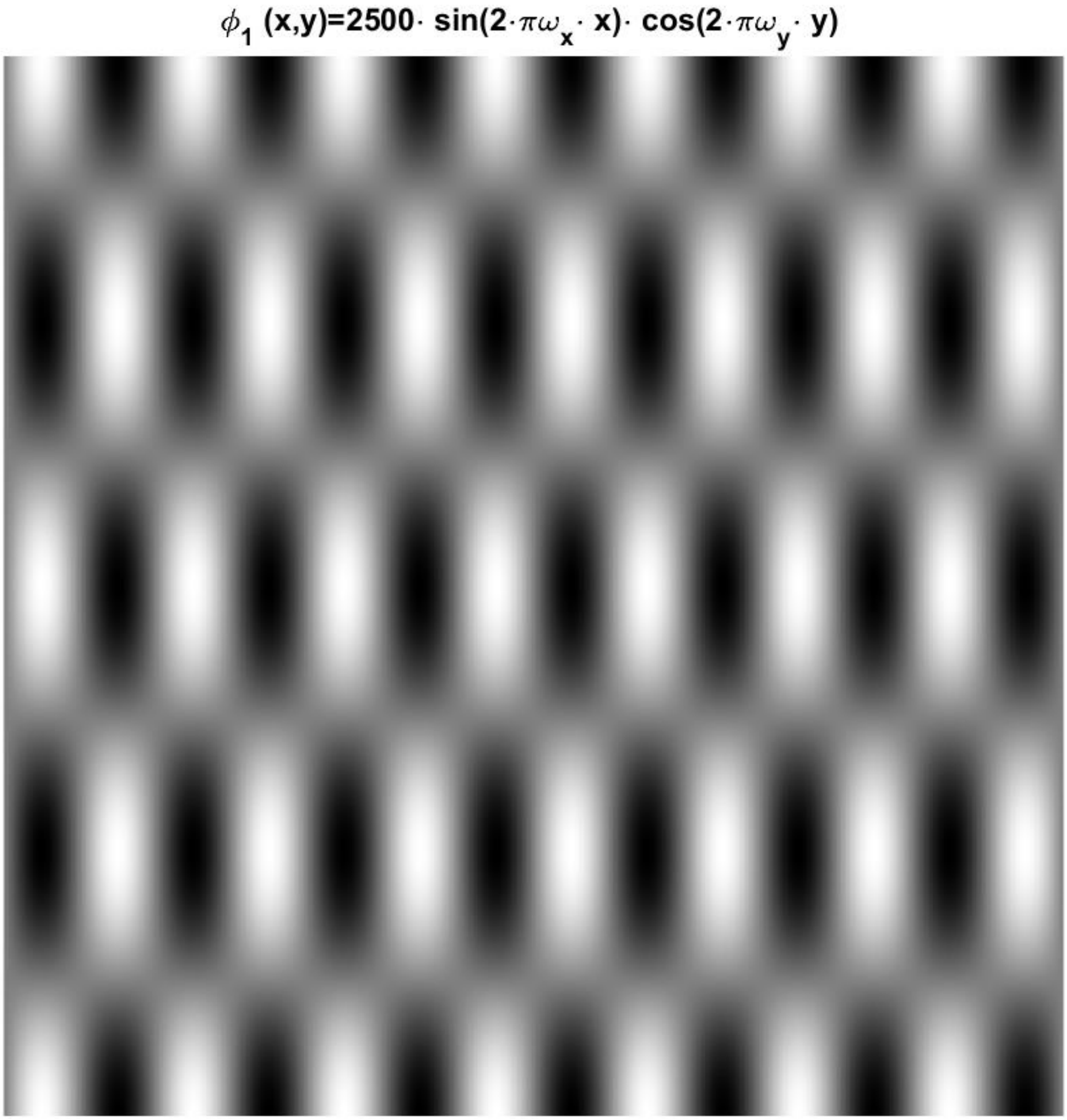
<u>Now we'll calculate the partial derivatives' energy:</u>

$$Energy\left(\frac{\partial\varphi}{\partial x}\right) = \int_0^1\int_0^1\left(\frac{\partial\varphi}{\partial x}(x,y)\right)^2 dxdy \ = (35000\pi)^2 \cdot \int_0^1 \cos^2(14\pi x)dx \cdot \int_0^1 \cos^2(4\pi y)dy = (35000\pi)^2 \cdot \frac{1}{2}\int_0^1 1+\cos(28\pi x)\,dx \cdot \frac{1}{2}\int_0^1 1+\cos(8\pi y)\,dy = \cdot\ 306250000\pi^2 \cdot \left(x+\frac{\sin(28\pi x)}{4\pi}\right)\Bigg|_0^1 \cdot \left(y+\frac{\sin(8\pi y)}{4\pi}\right)\Bigg|_0^1 = 306250000\pi^2 \cdot (1+0)\cdot(1+0) = 306250000\pi^2$$

$$Energy\left(\frac{\partial\varphi}{\partial y}\right) = \int_0^1\int_0^1\left(\frac{\partial\varphi}{\partial x}(x,y)\right)^2 dxdy \ = (10000\pi)^2 \cdot \int_0^1 \sin^2(14\pi x)dx \cdot \int_0^1 \sin^2(4\pi y)dy = (10000\pi)^2 \cdot \frac{1}{2}\int_0^1 1-\cos(28\pi x)\,dx \cdot \frac{1}{2}\int_0^1 1-\cos(8\pi y)\,dy = 25000000\pi^2 \cdot \left(x-\frac{\sin(28\pi x)}{4\pi}\right)\Bigg|_0^1 \cdot \left(y-\frac{\sin(8\pi y)}{4\pi}\right)\Bigg|_0^1 = 25000000\pi^2 \cdot (1-0)\cdot(1-0) = 25000000\pi^2$$

b.

The representation on a grayscale scaling of [-2500, 2500] is:



<u>Expectedly in an intuitive sense, the image “rotated”, in fact the 2D periodic sine-cosine figure stretched out vertically, instead of the horizontal stretched figure we’ve seen for the previous coefficients.</u>

c.

The results are:

```
phi_H - phi_L = 5000
Energy( phi _x `) = 3028123289.121939
Energy( phi _y `) = 246736863.074624
```

It is expected that the energies are swapped because we are in the interval [0,1]x[0,1] and only the coefficients of the sine and cosine swapped , thus the integrals as is seen in the calculations left unchanged and the coefficients are the dominant in determining the energy (because of the derivative) for this function.

d.

same as previous due to reuse of same code.

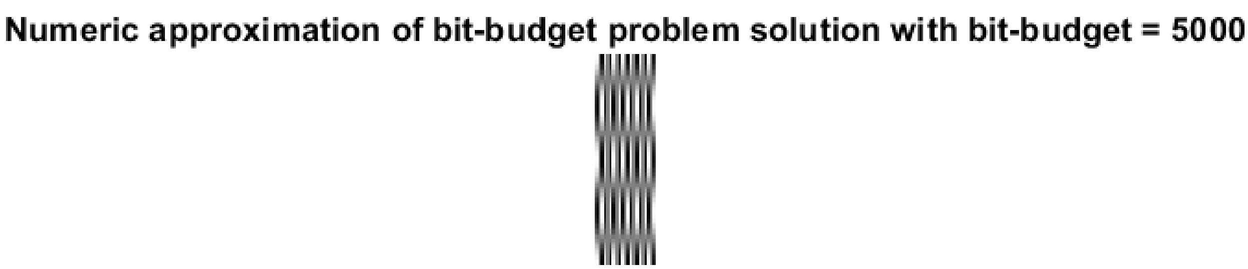
e.

for budget of 5000 are:

```
Numeric method approx. - Bit budget = 5000, Nx = 73.000000, Ny = 21.000000, b =3.000000
The MSE = 126529.485672
```

The parameters swapped and that is a logical outcome due to the previous observations of the "rotation" of the periodic 2d sine-cosine figure. That is expected to be the same swapped numbers because it is like we swapped the x and y axes.

And as expected we will get for the same parameters the same visualized image, only rotated:



Exactly the same with a budget of 50,000:

```
Numeric method approx. - Bit budget = 50000, Nx = 188.000000, Ny = 53.000000, b =5.000000
The MSE = 16493.981046
```

Same visual effect:

Numeric approximation of bit-budget problem solution with bit-budget = 50000

- **<u>In both cases the MSE remained exactly same, which is a logical and expected outcome due to previous observations: same optimized parameters, same image but rotated, the optimization in this case is invariant to rotating or switching the axes.</u>**

f.

<u>same as previous due to reuse of same code.</u>

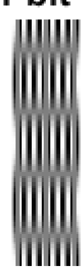
g.

<u>Exactly the same conclusions as for the previous section about the results:</u>

<u>Swapped axes, therefore swapped parameters of x and y axes sampling, same MSE, rotated image.</u>

```
Partical sol. - Bit budget = 5000, Nx = 79.000000, Ny = 21.000000, b =3.000000
The MSE = 119609.786046
```

Practically found solution for bit-budget problem with B=5000



For budget 50,000:

```
Partical sol. - Bit budget = 50000, Nx = 185.000000, Ny = 54.000000, b =5.000000
The MSE = 16458.819955
```

Practically found solution for bit-budget problem with B=50000

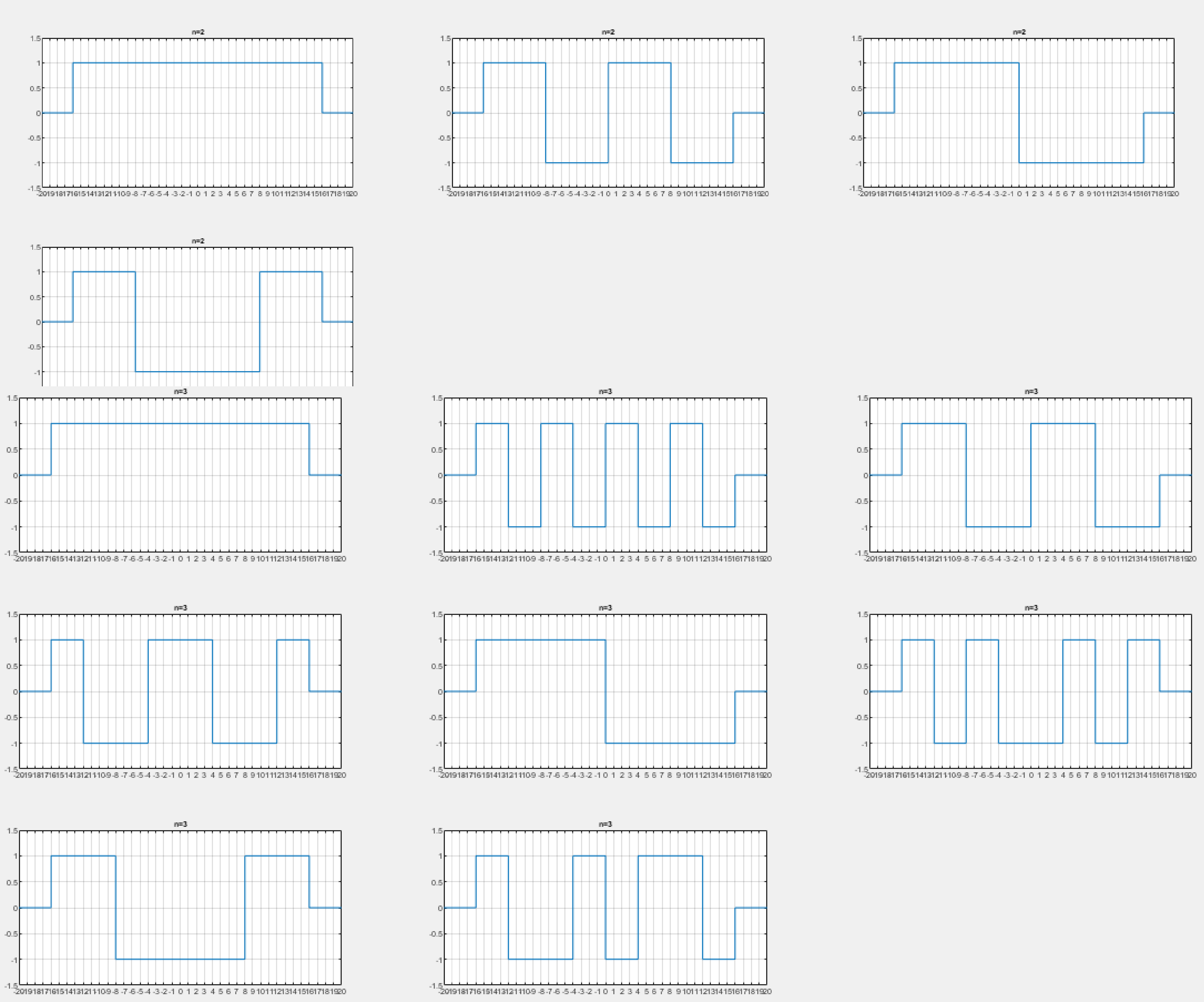
<u>Question 2:</u>

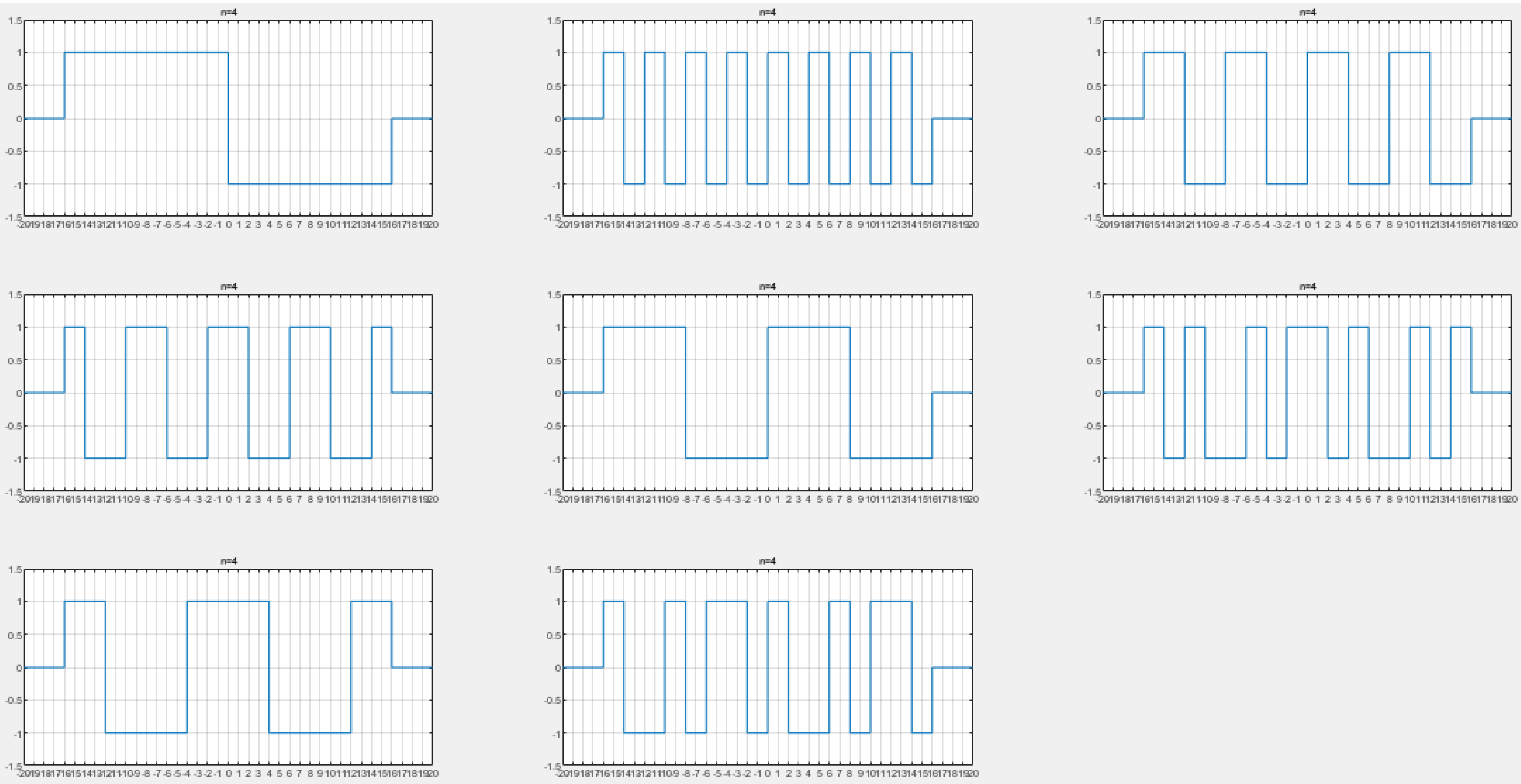
a.

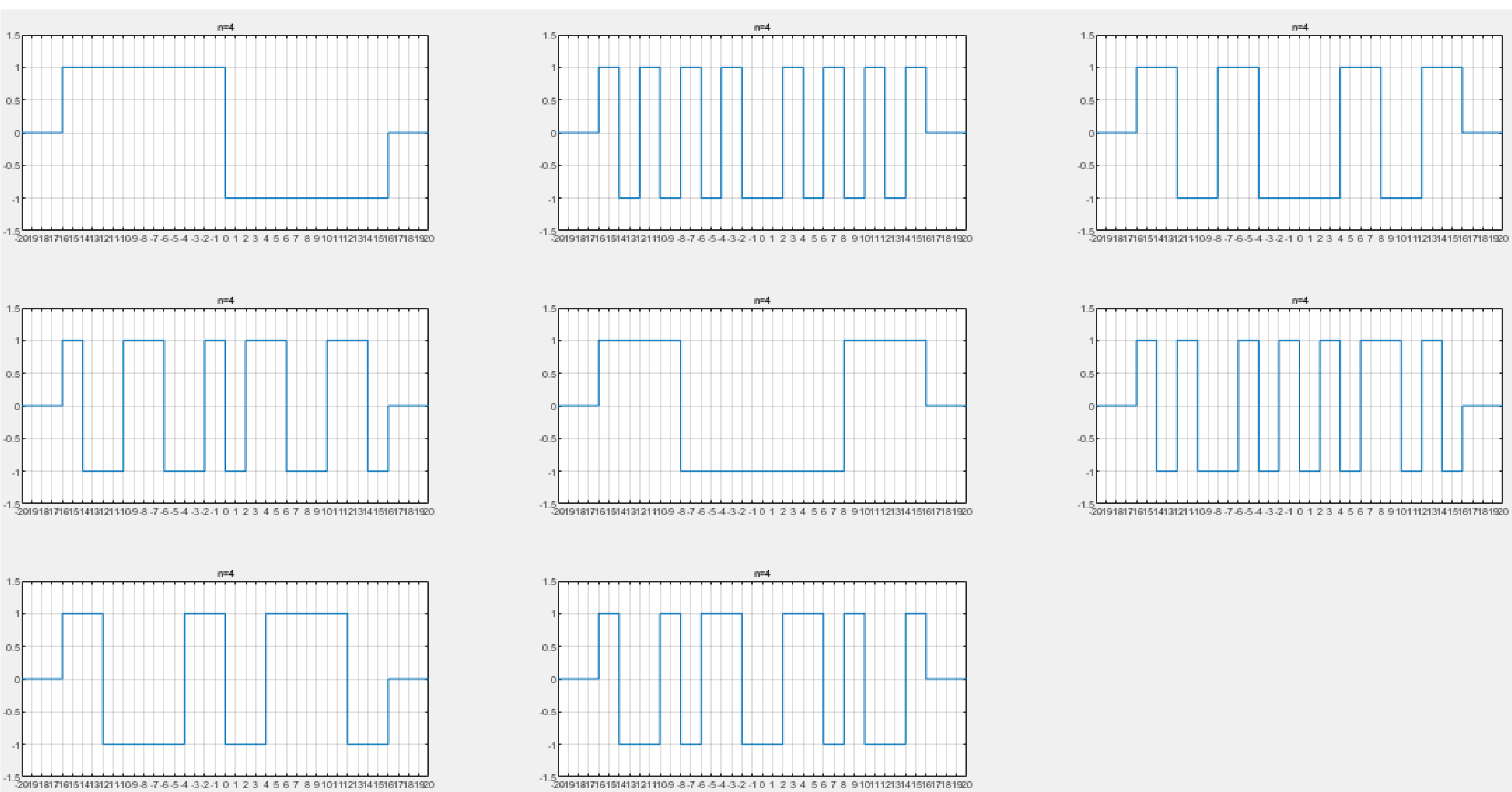
see function “generateHadamard”. It constructs the Hadamard matrix by the recursive definition we know.

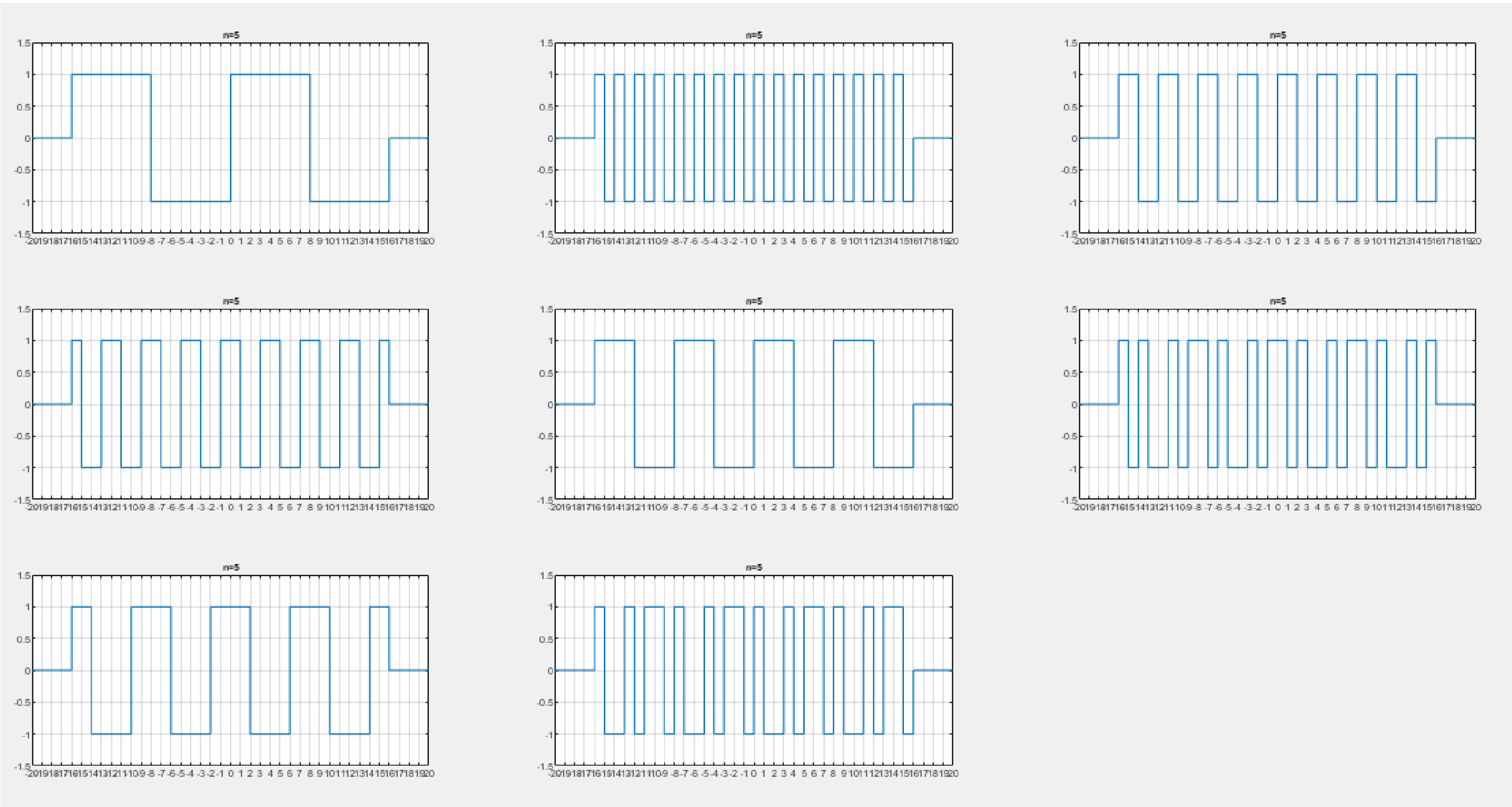
b. Hadamard functions:

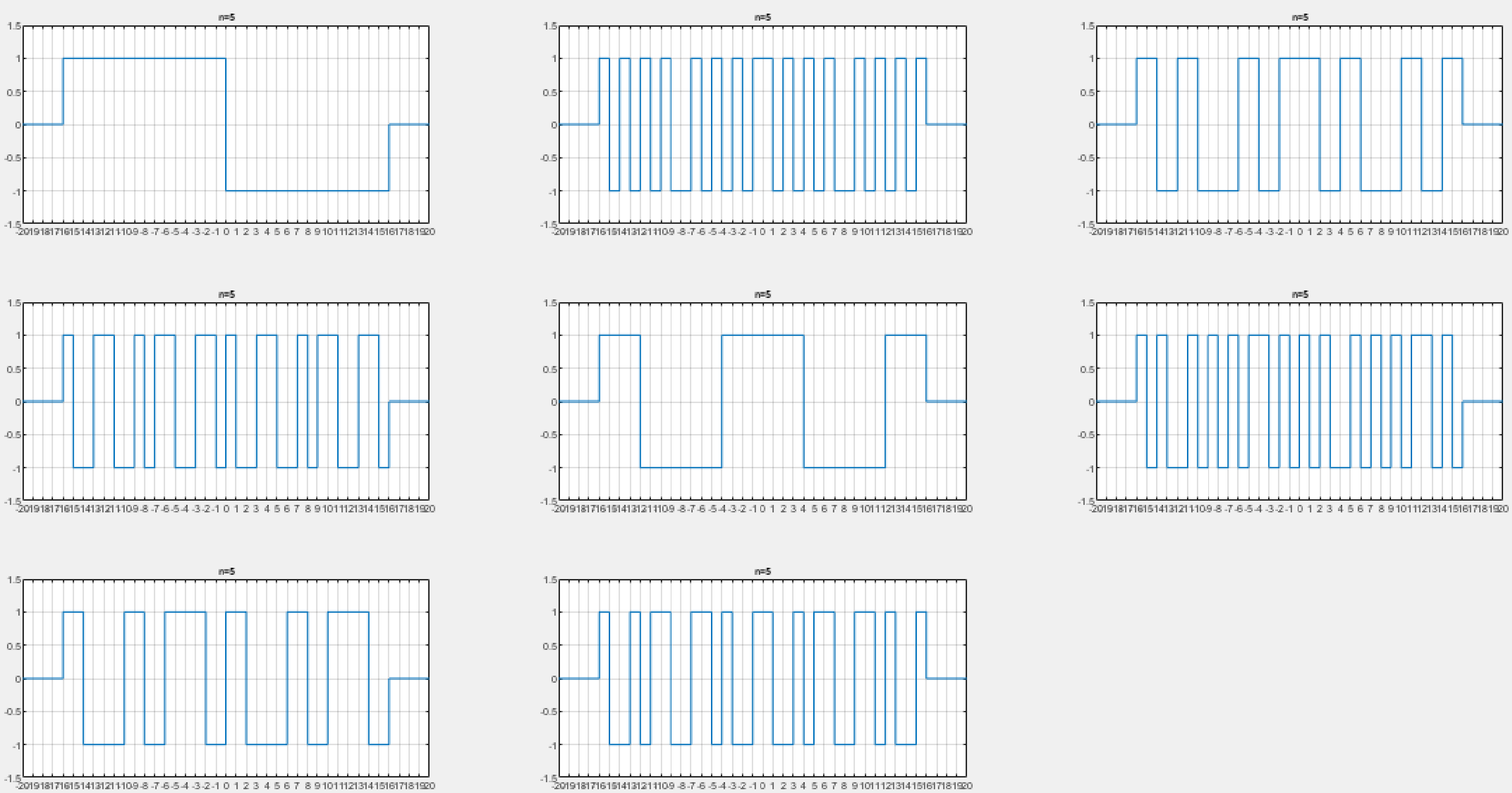
for each n:

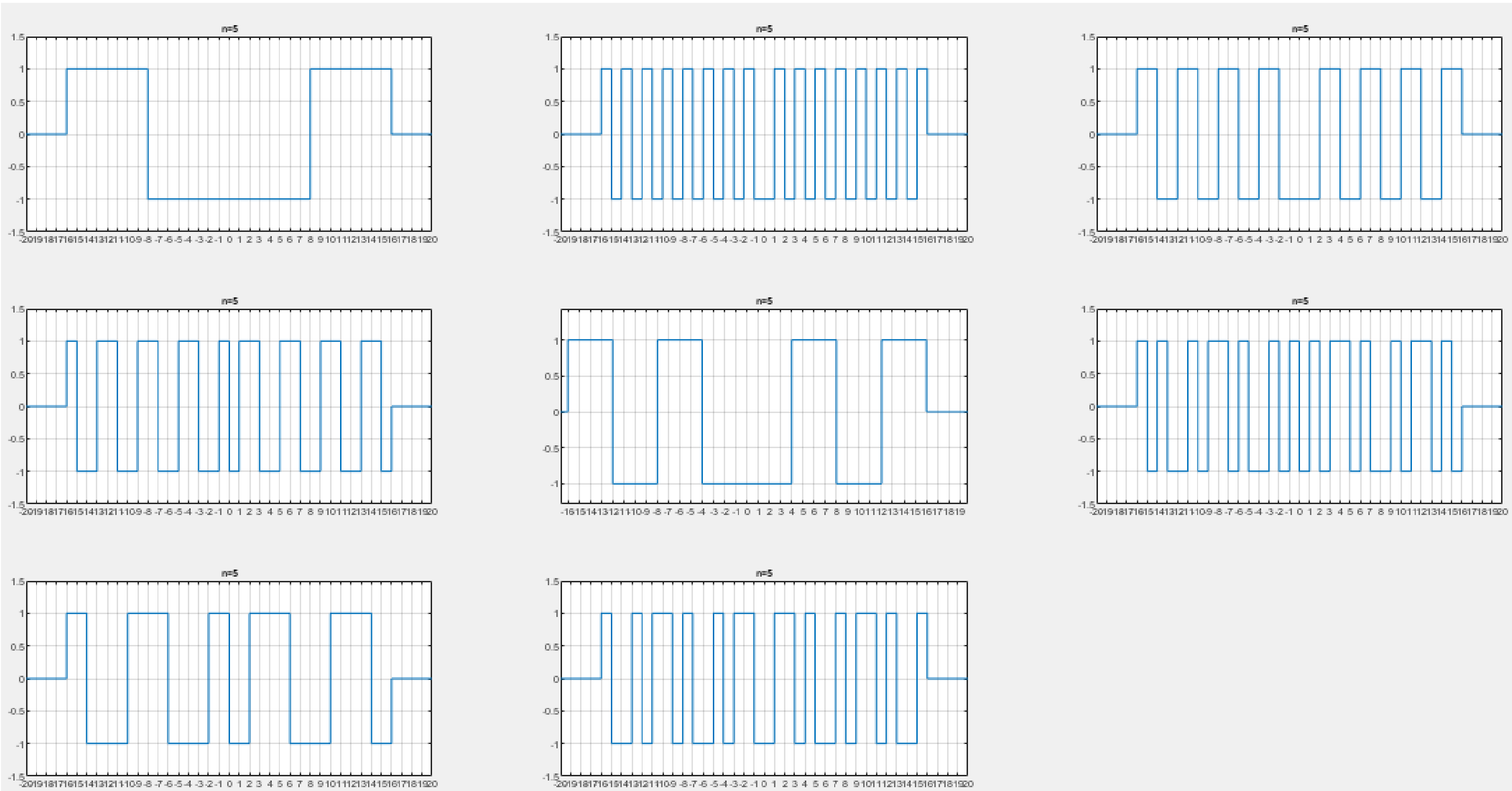


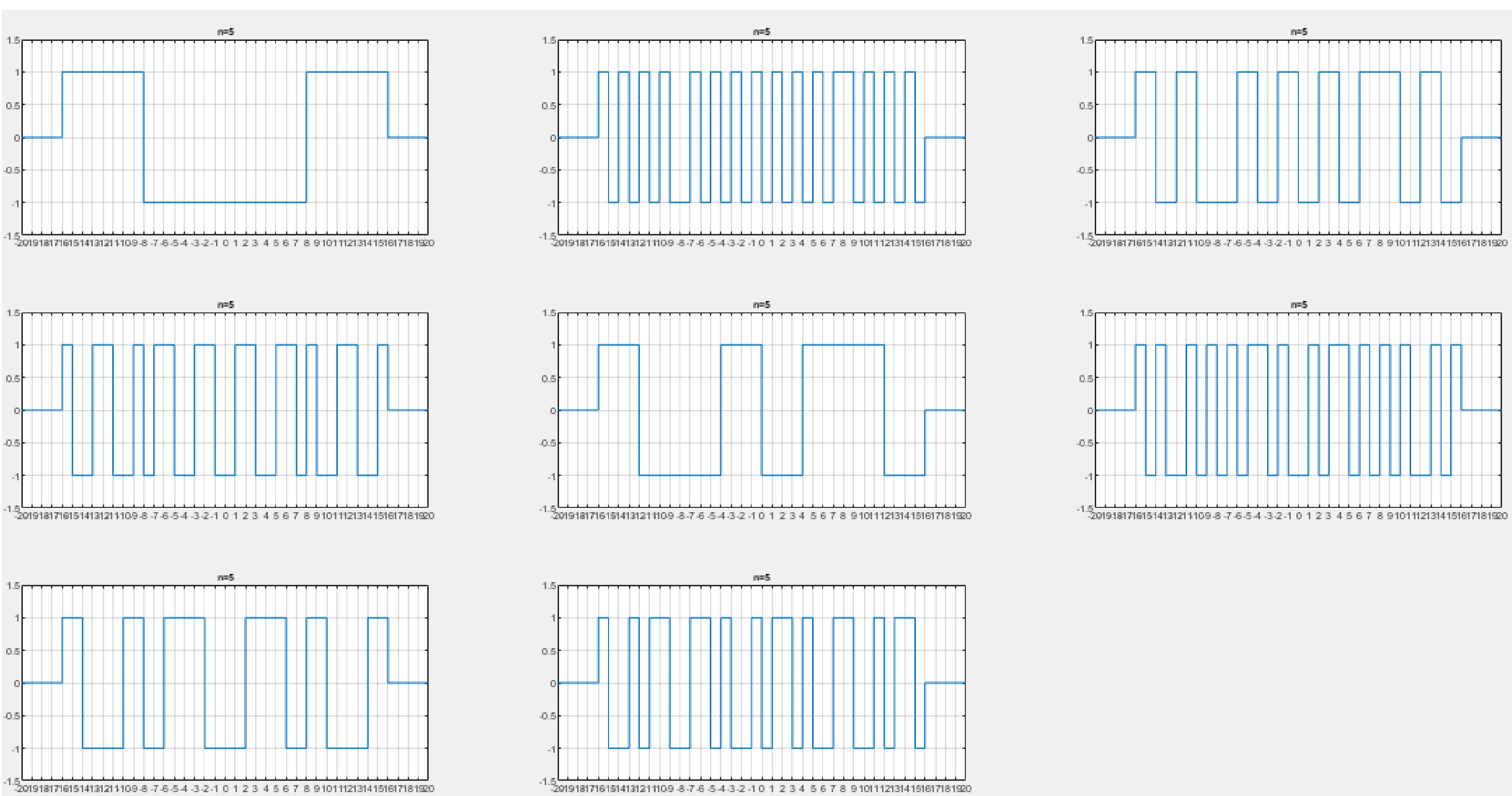

n=4
n=4
n=4
n=4
n=4
n=4
n=4
n=4

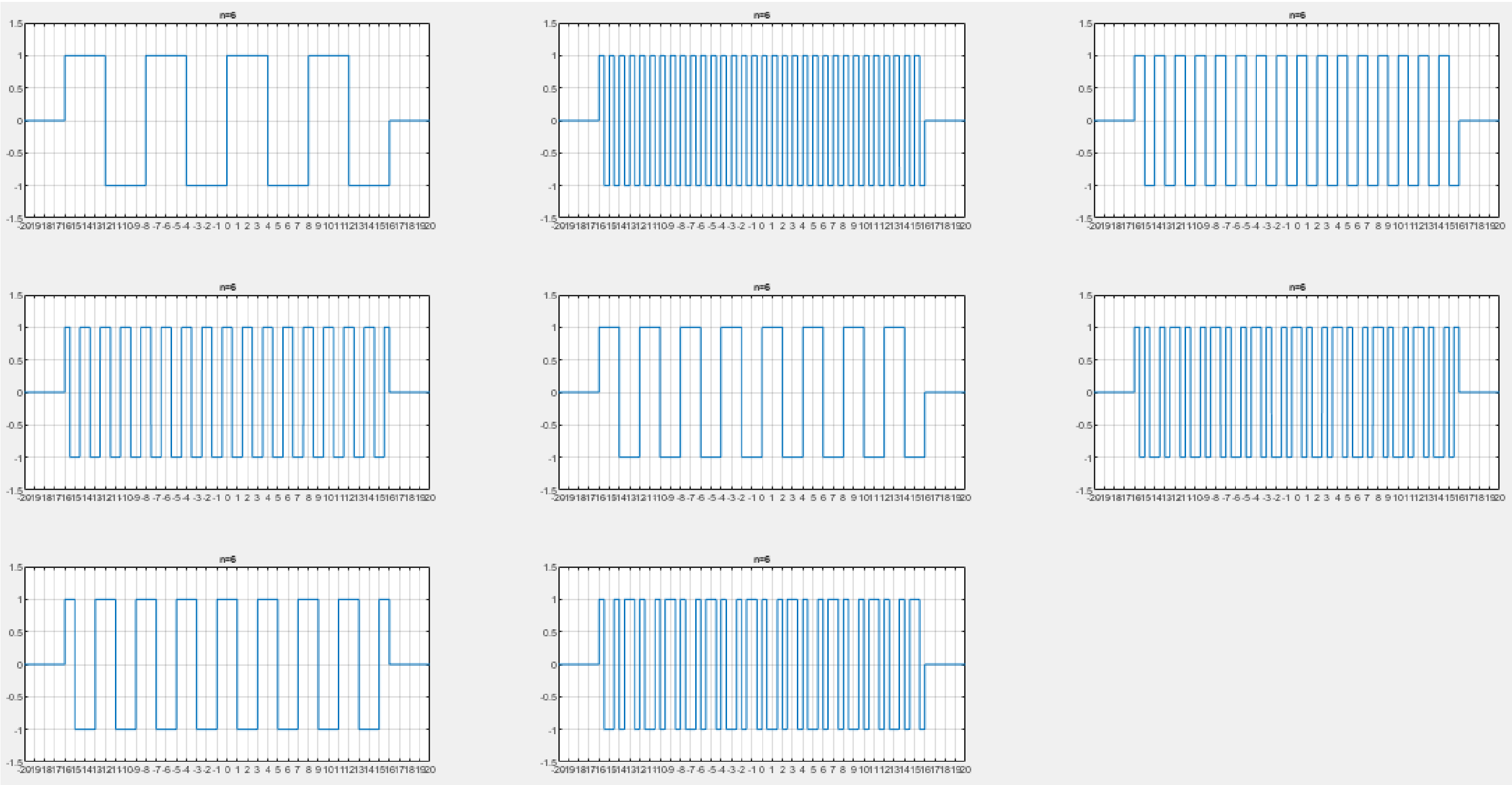

n=4
n=4
n=4
n=4
n=4
n=4
n=4
n=4

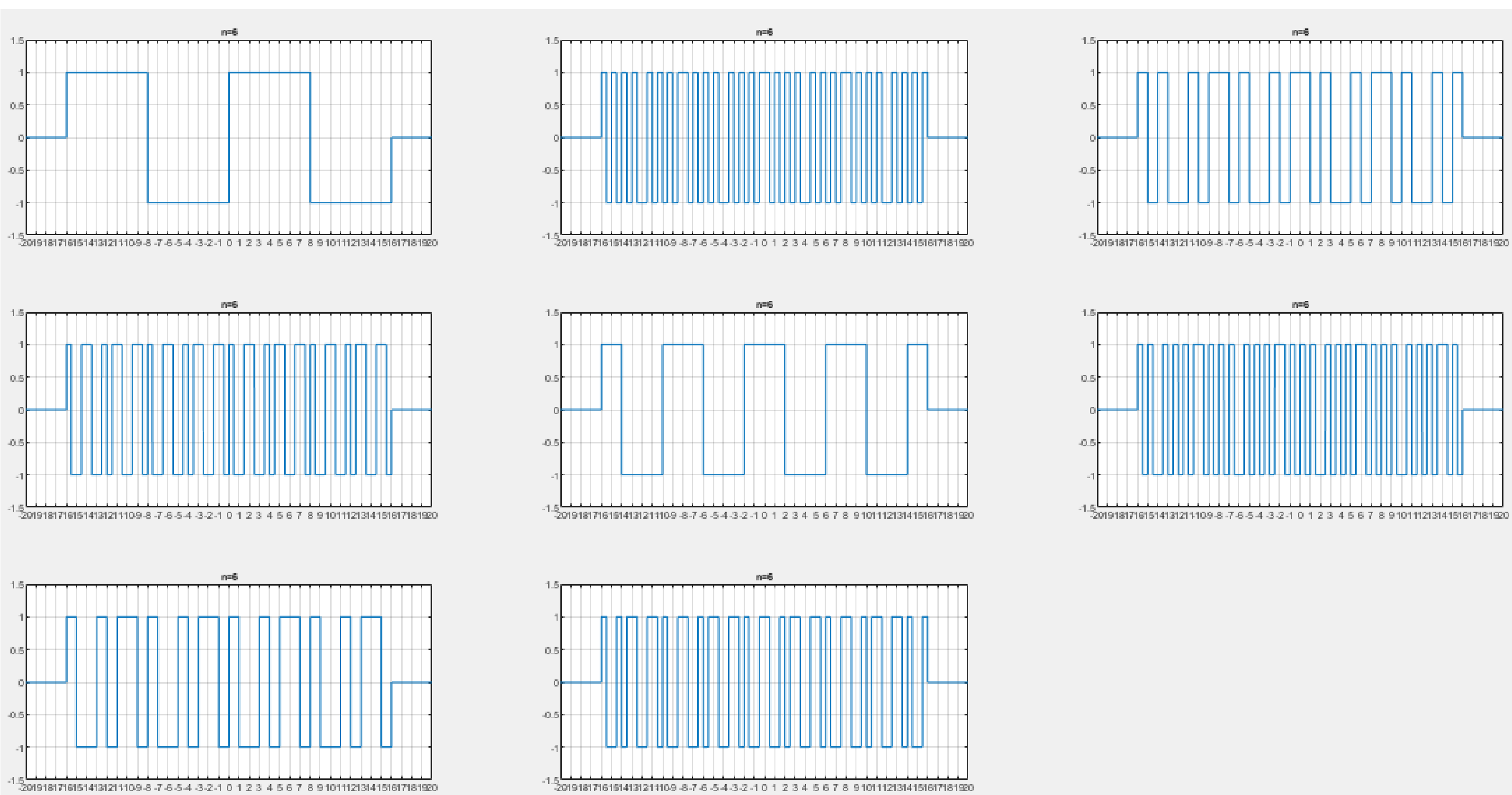

n=5
n=5
n=5
n=5
n=5
n=5
n=5
n=5

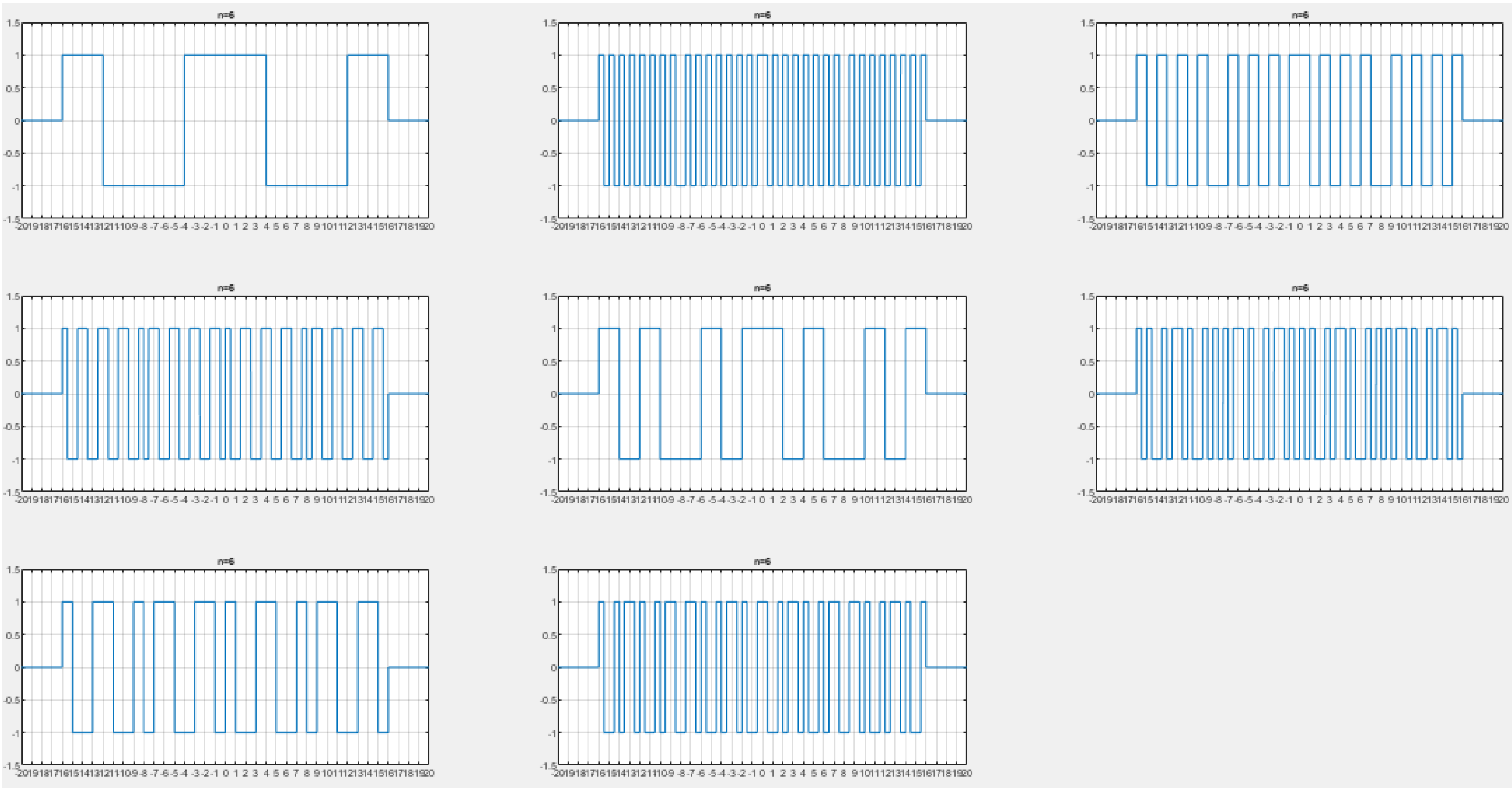

n=5
n=5
n=5
n=5
n=5
n=5
n=5
n=5

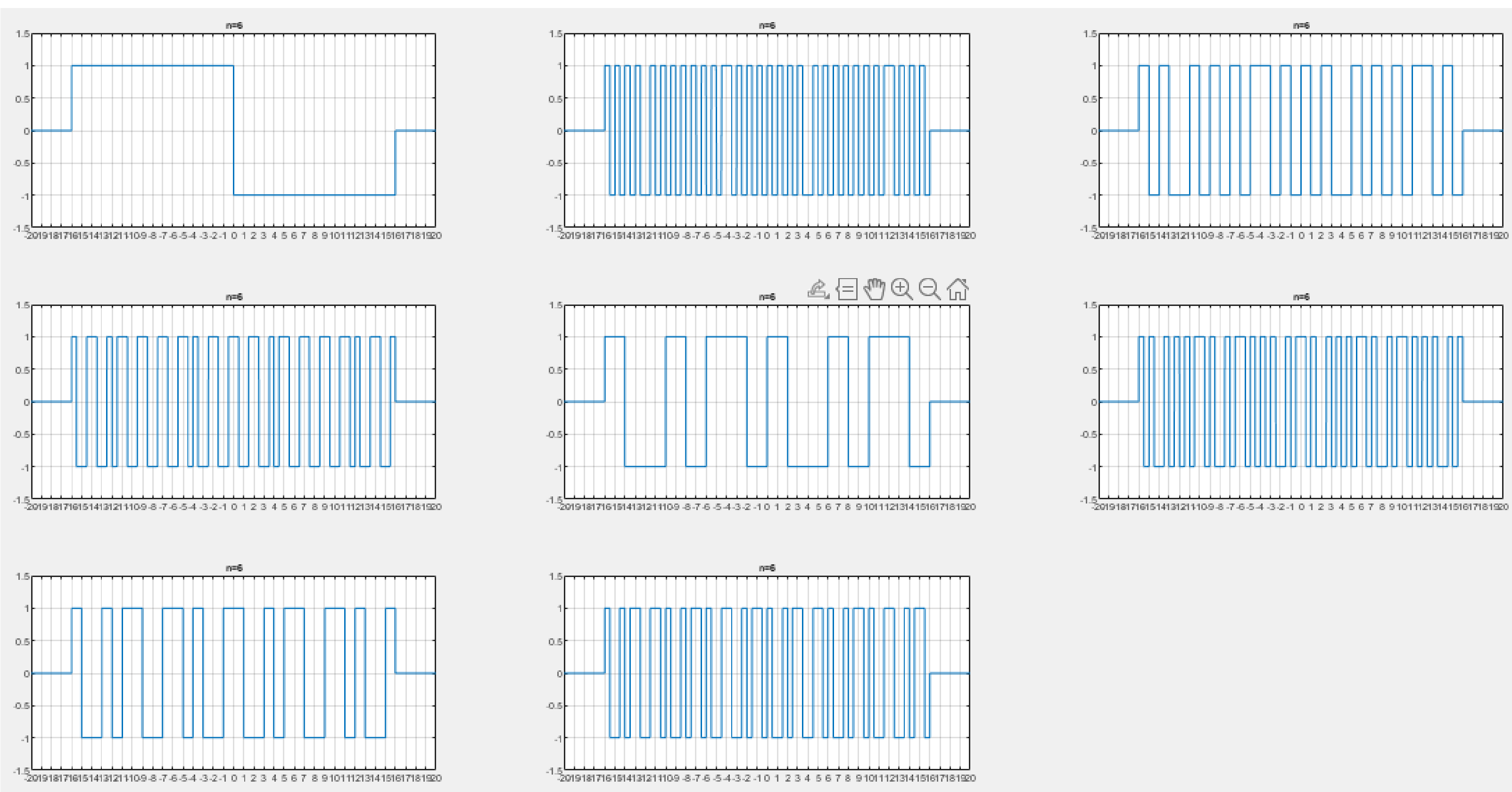

n=5
n=5
n=5
n=5
n=5
n=5
n=5
n=5

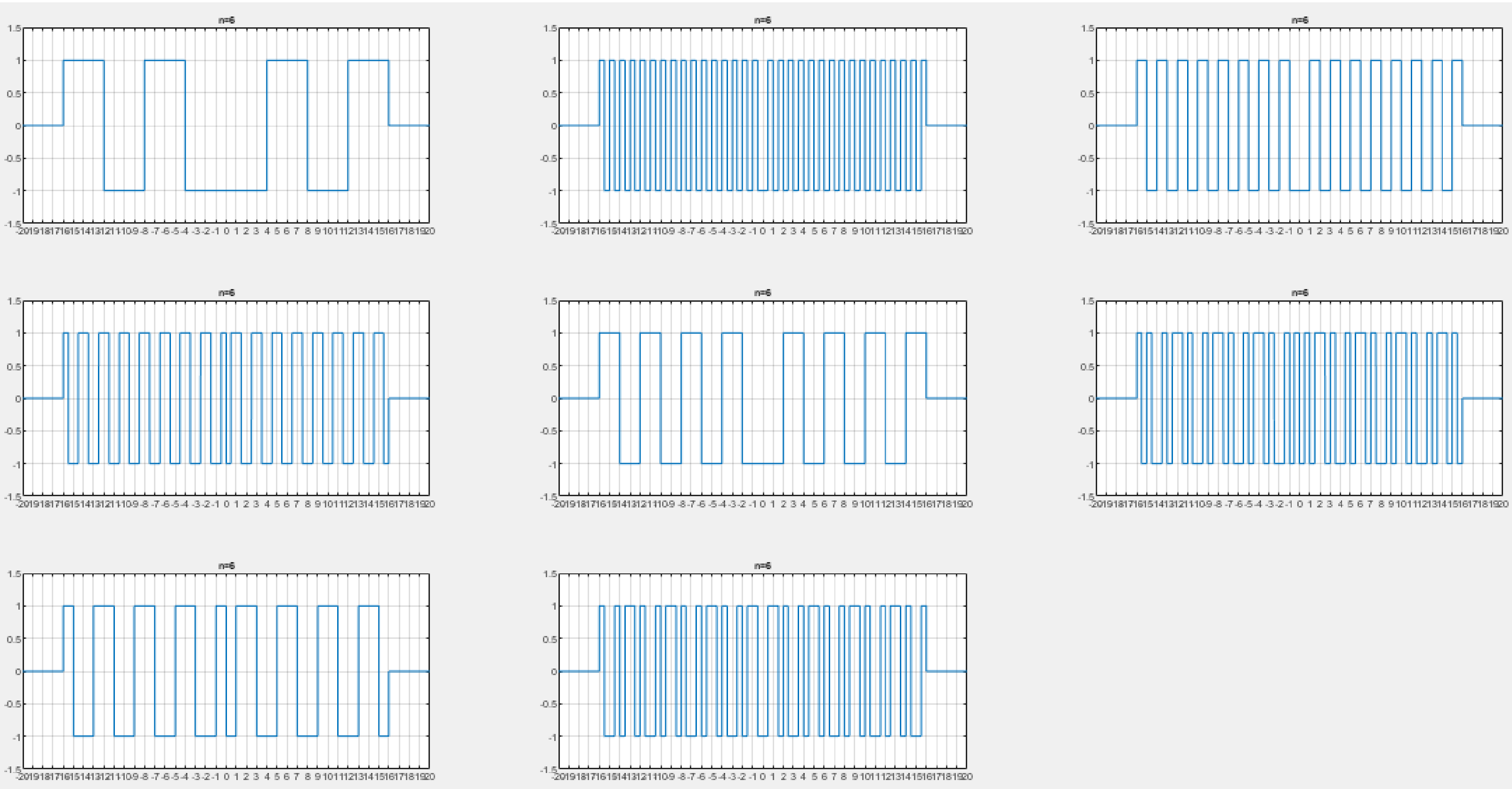

n=5
n=5
n=5
n=5
n=5
n=5
n=5
n=5

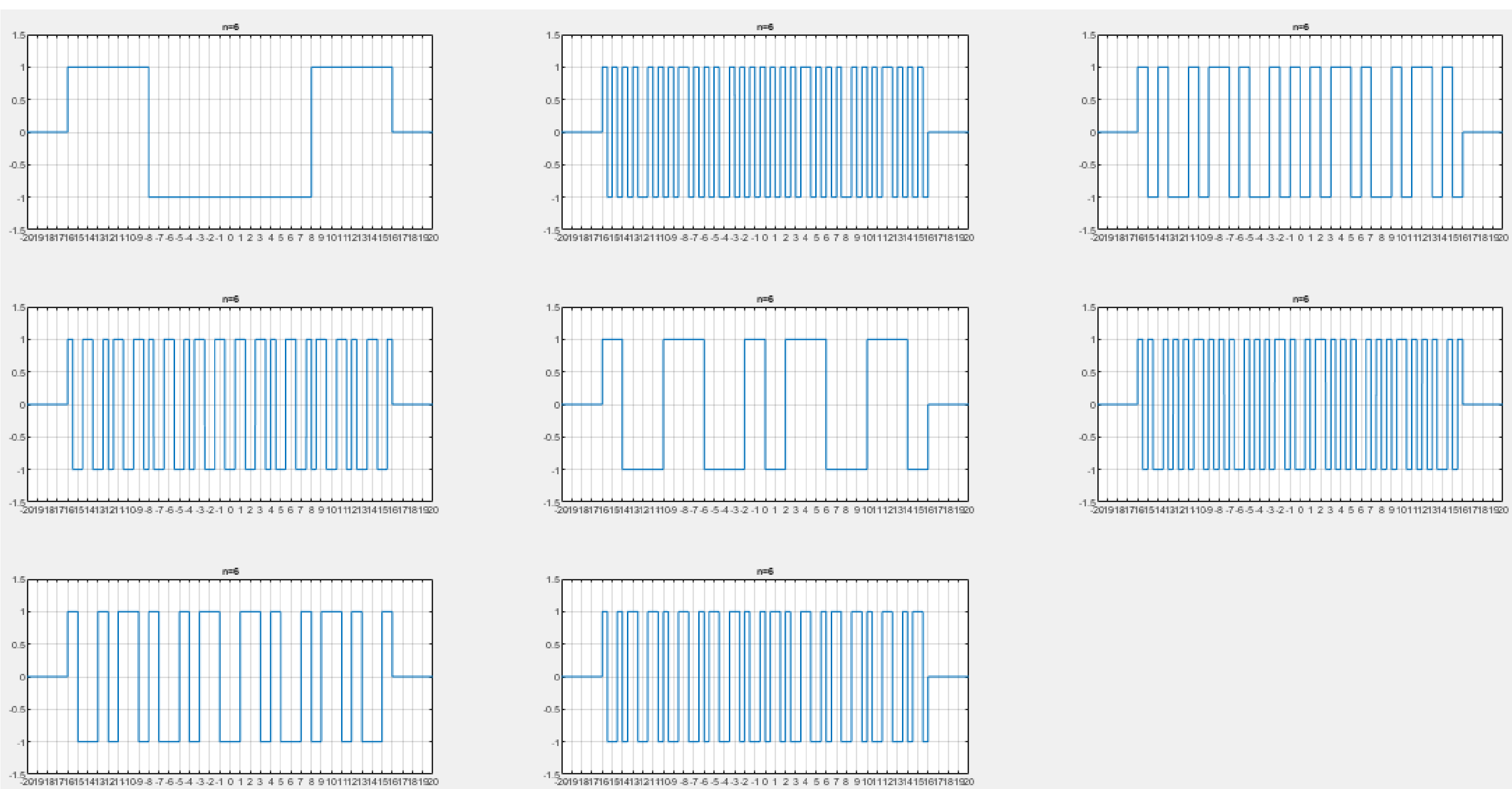

n=6
n=6
n=6
n=6
n=6
n=6
n=6
n=6

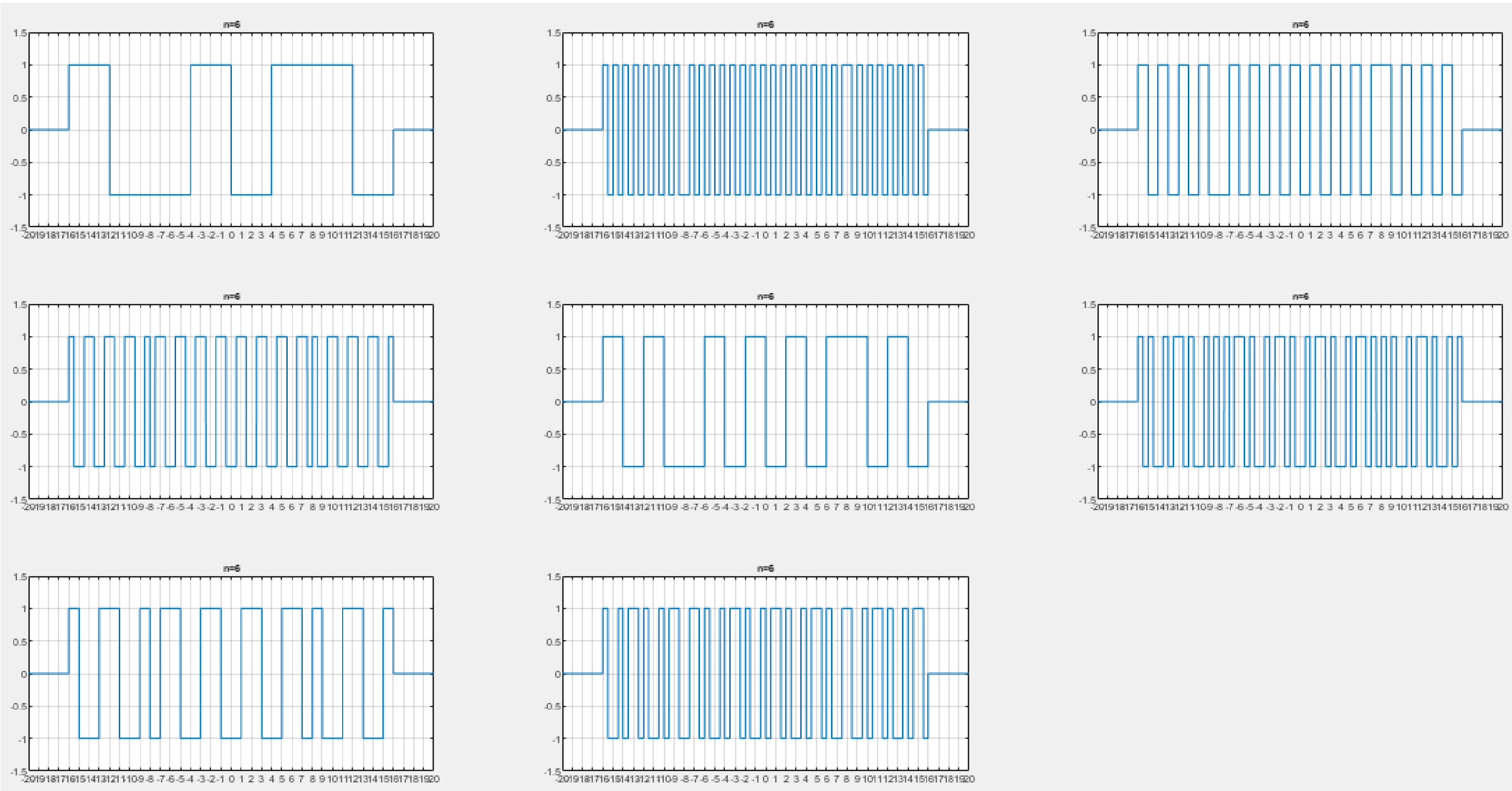

n=6
n=6
n=6
n=6
n=6
n=6
n=6
n=6

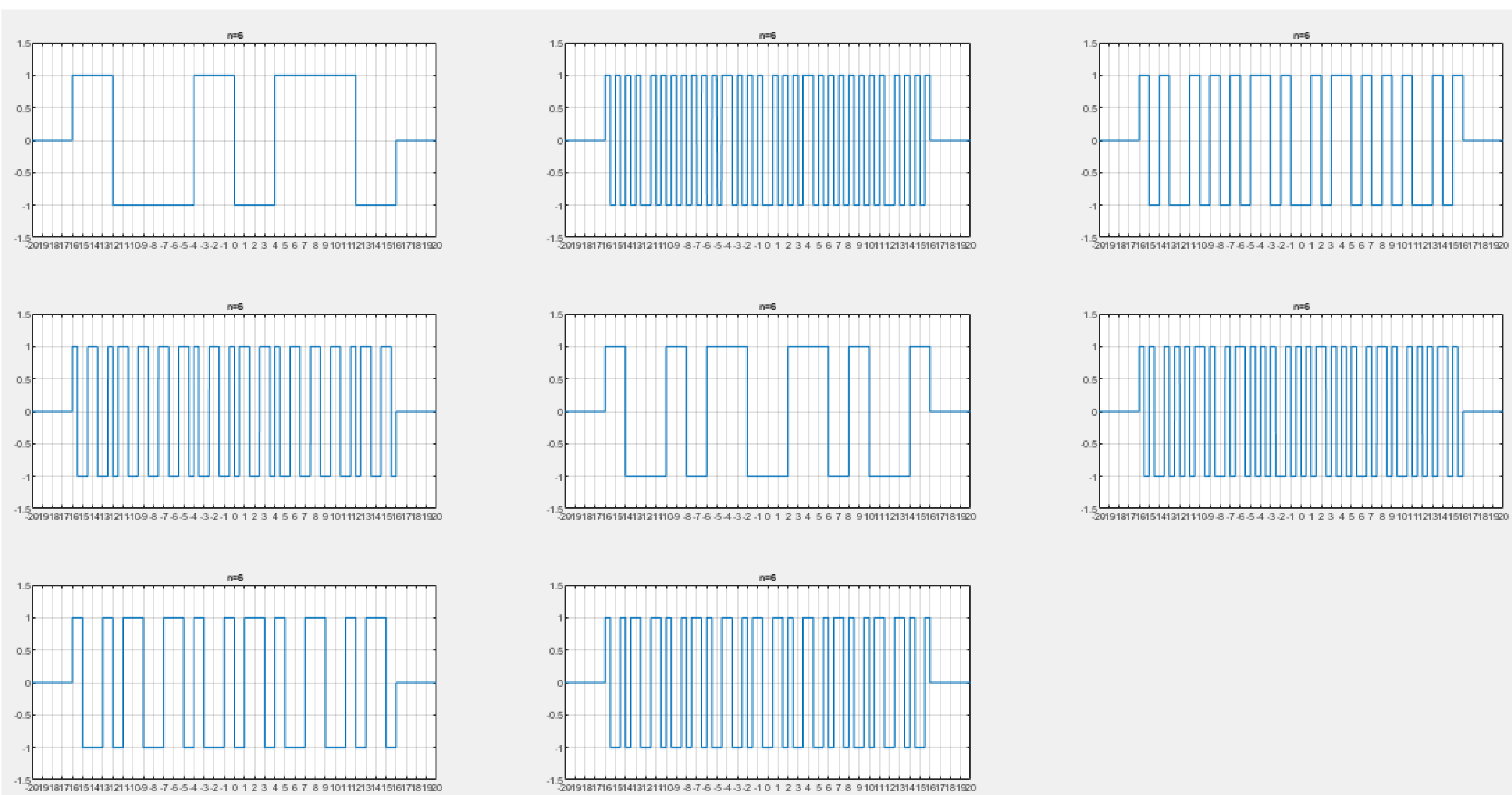

n=6
n=6
n=6
n=6
n=6
n=6
n=6
n=6


n=6
n=6
n=6
n=6
n=6
n=6
n=6
n=6


n=6
n=6
n=6
n=6
n=6
n=6
n=6
n=6

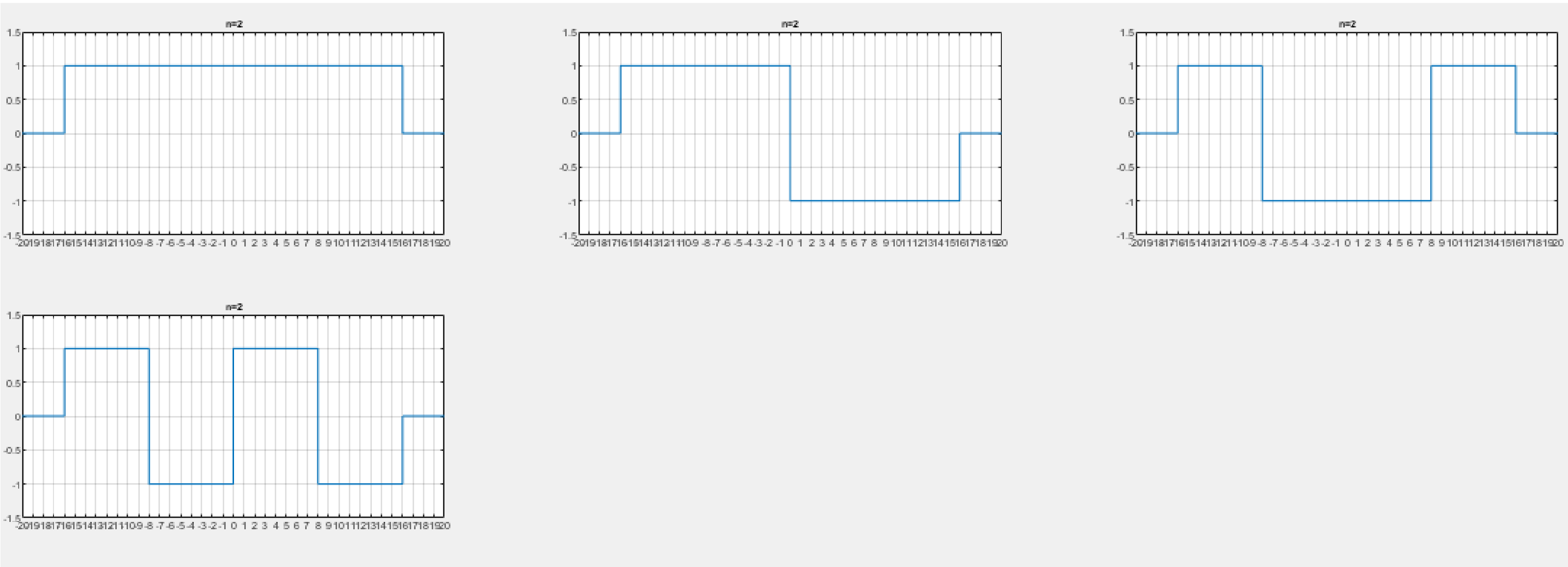

n=6
n=6
n=6
n=6
n=6
n=6
n=6
n=6

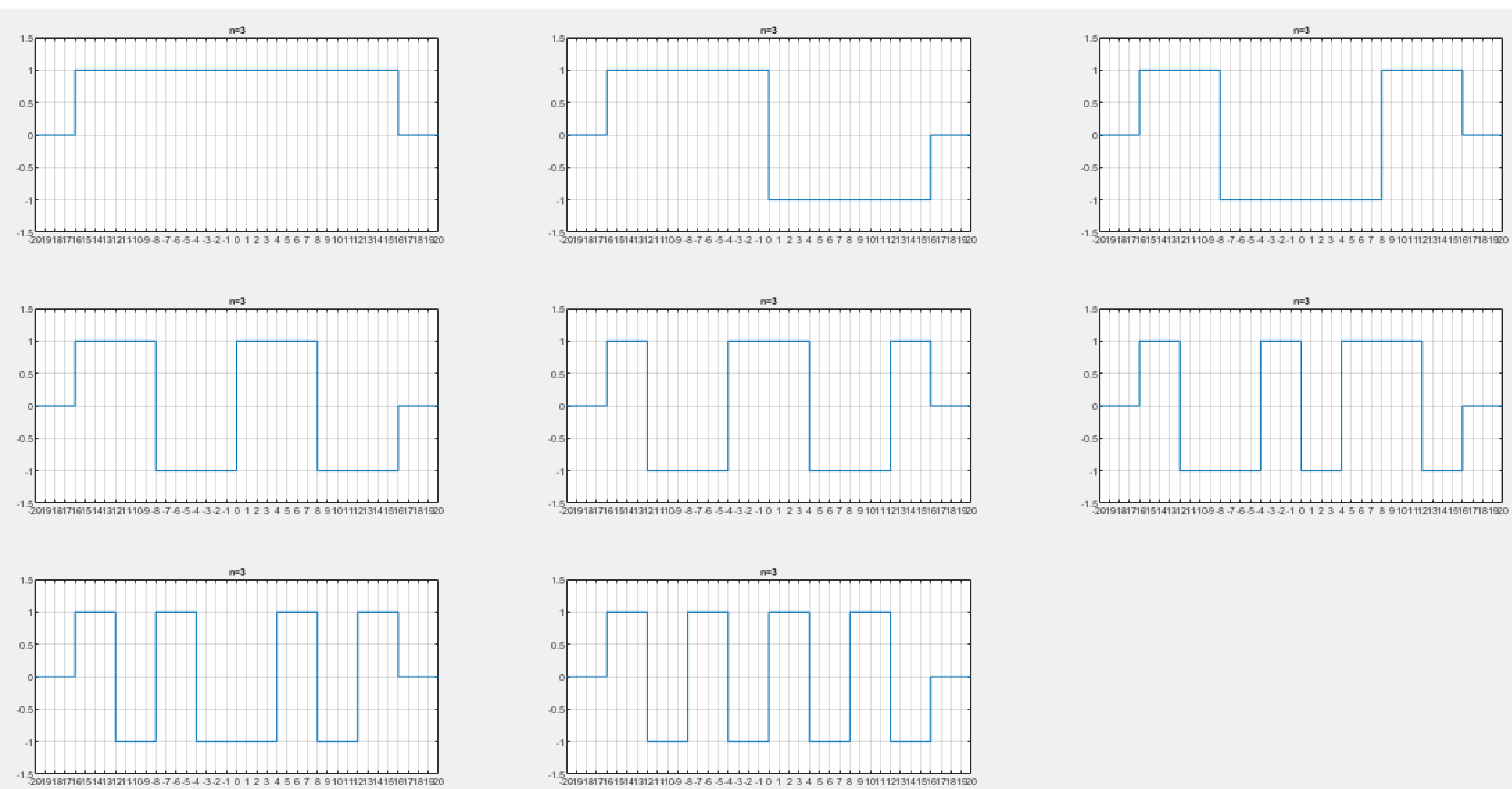

n=6
n=6
n=6
n=6
n=6
n=6
n=6
n=6

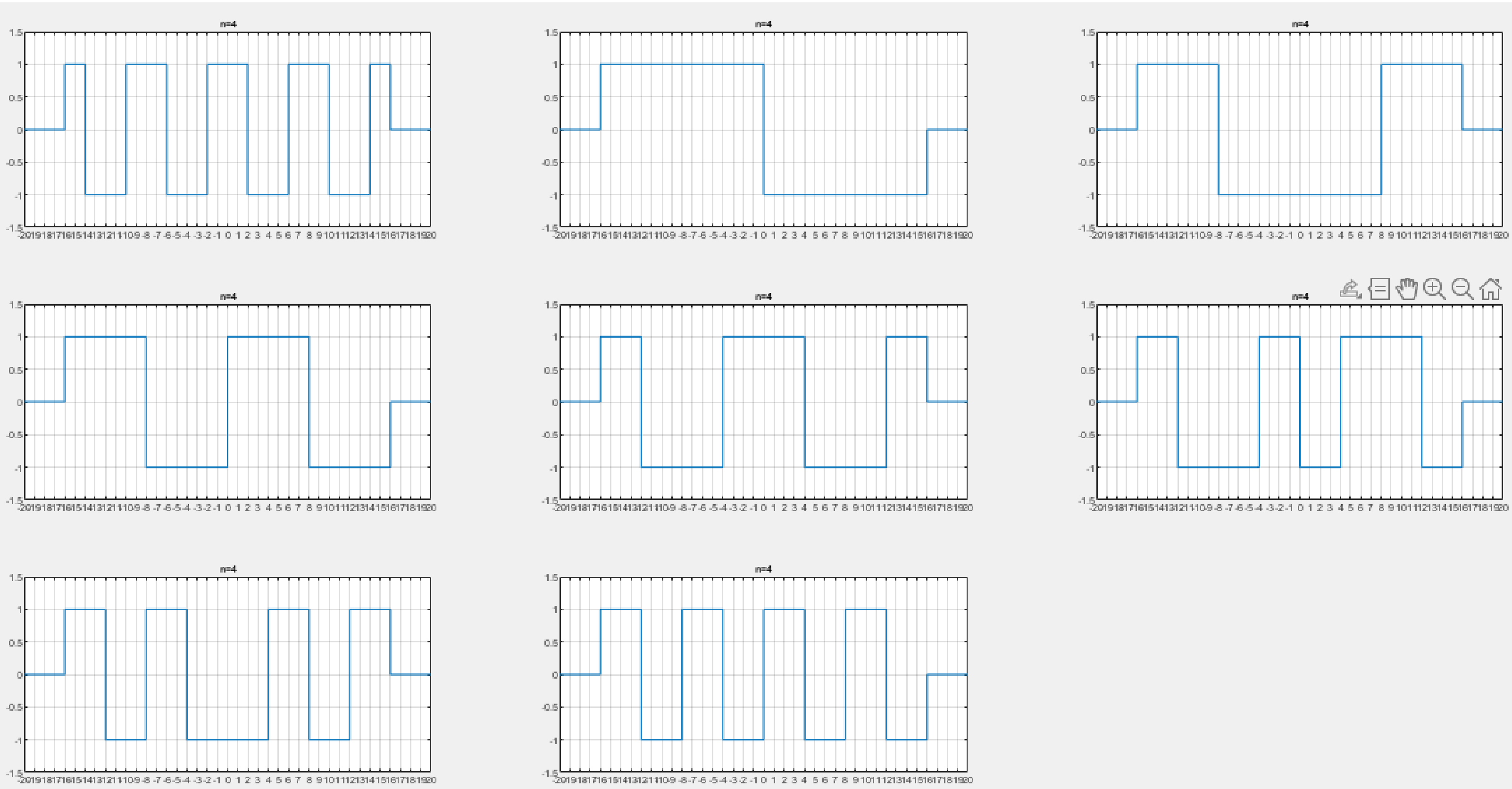

n=6
n=6
n=6
n=6
n=6
n=6
n=6
n=6

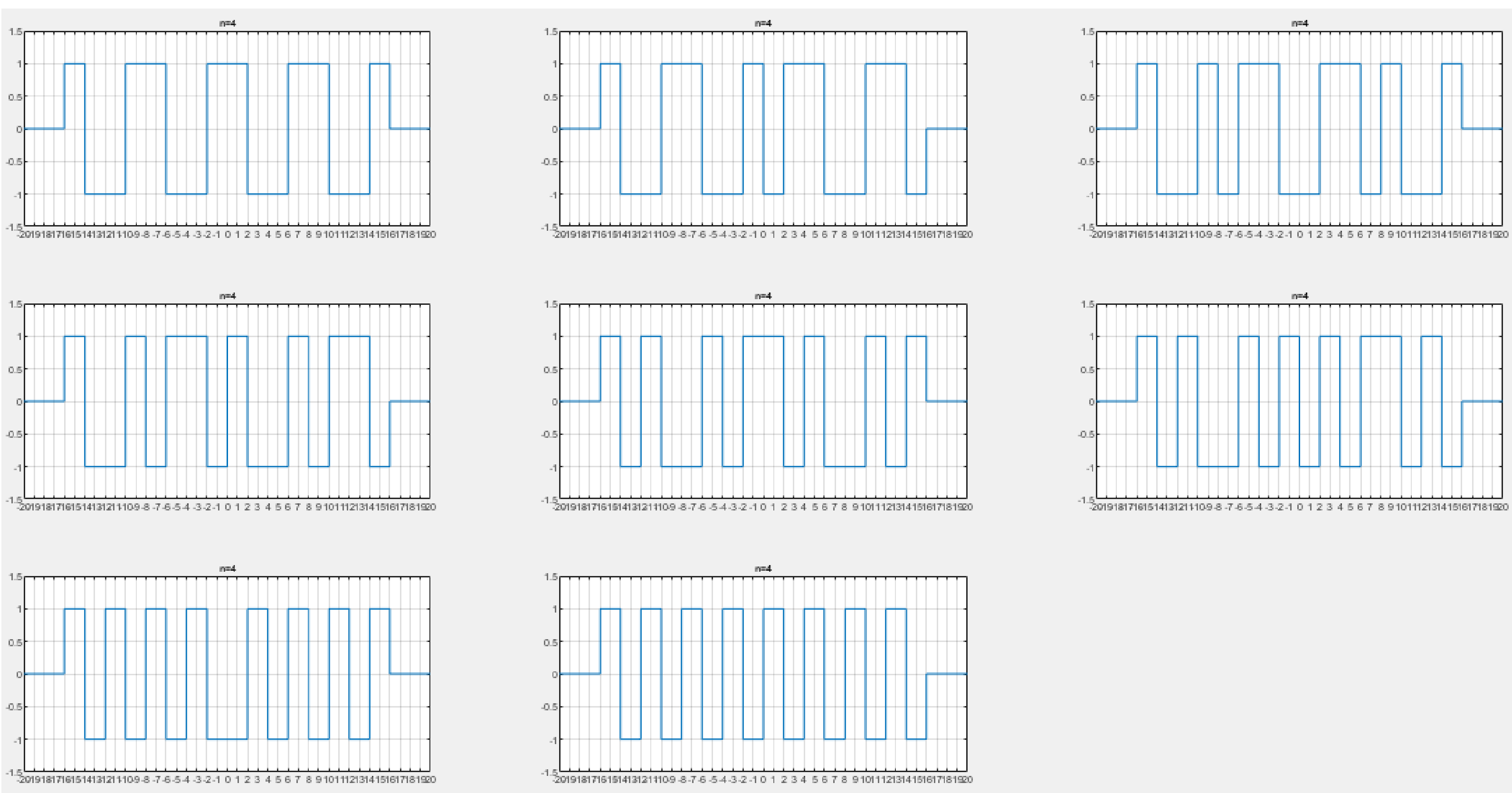
d. Walsh-Hadamard functions:

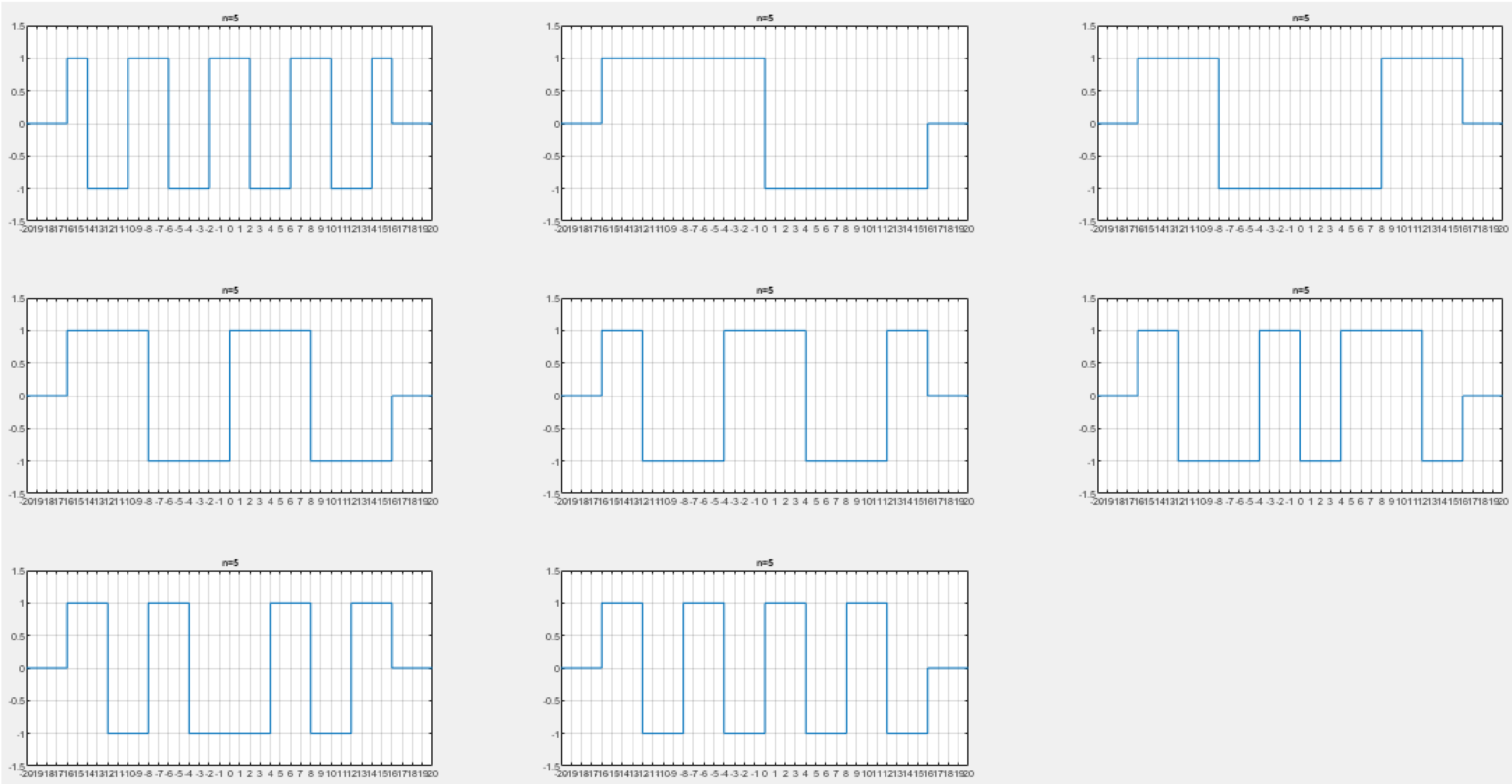
for each n:

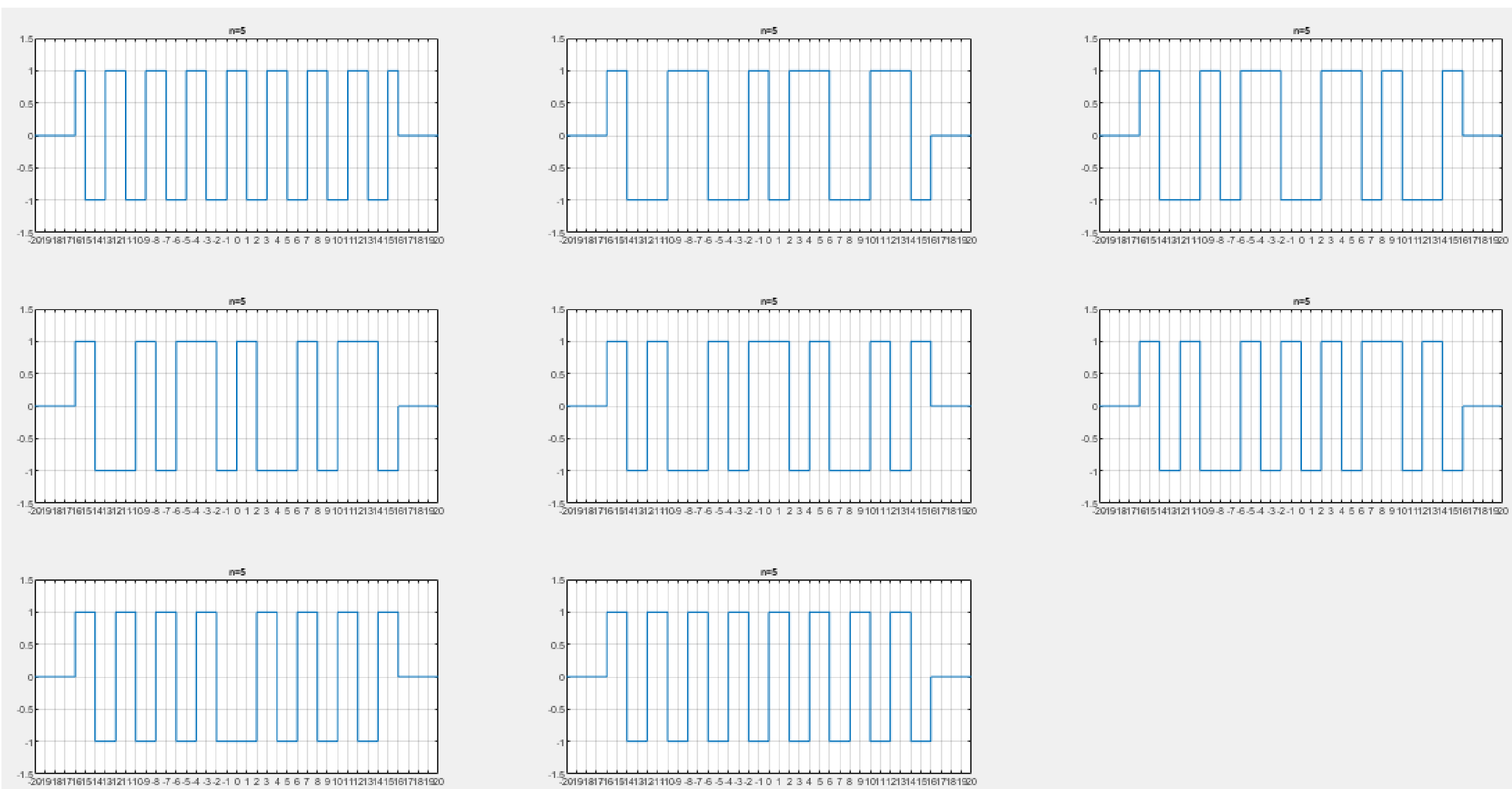


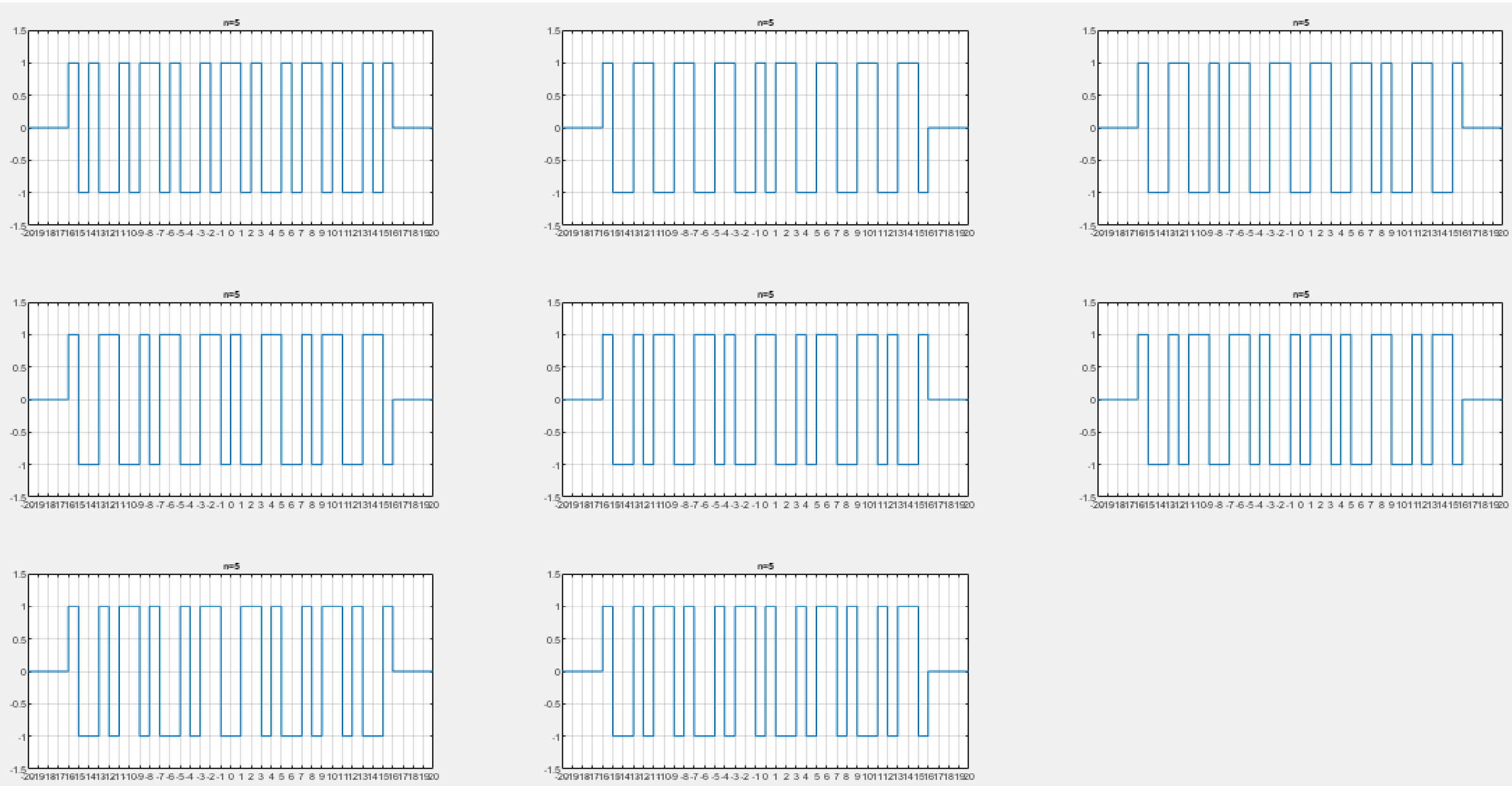


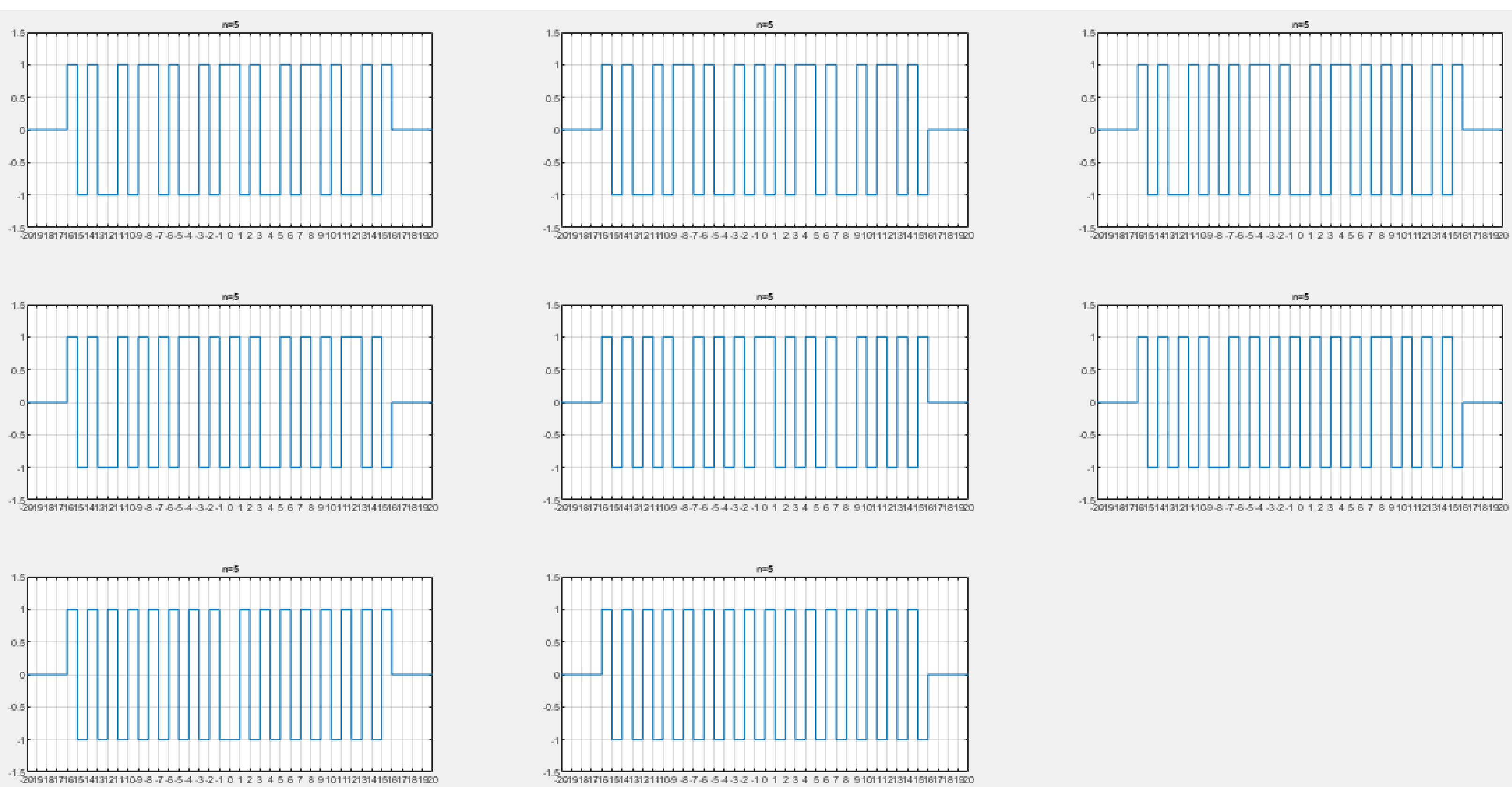

n=4
n=4
n=4
n=4
n=4
n=4
n=4
n=4

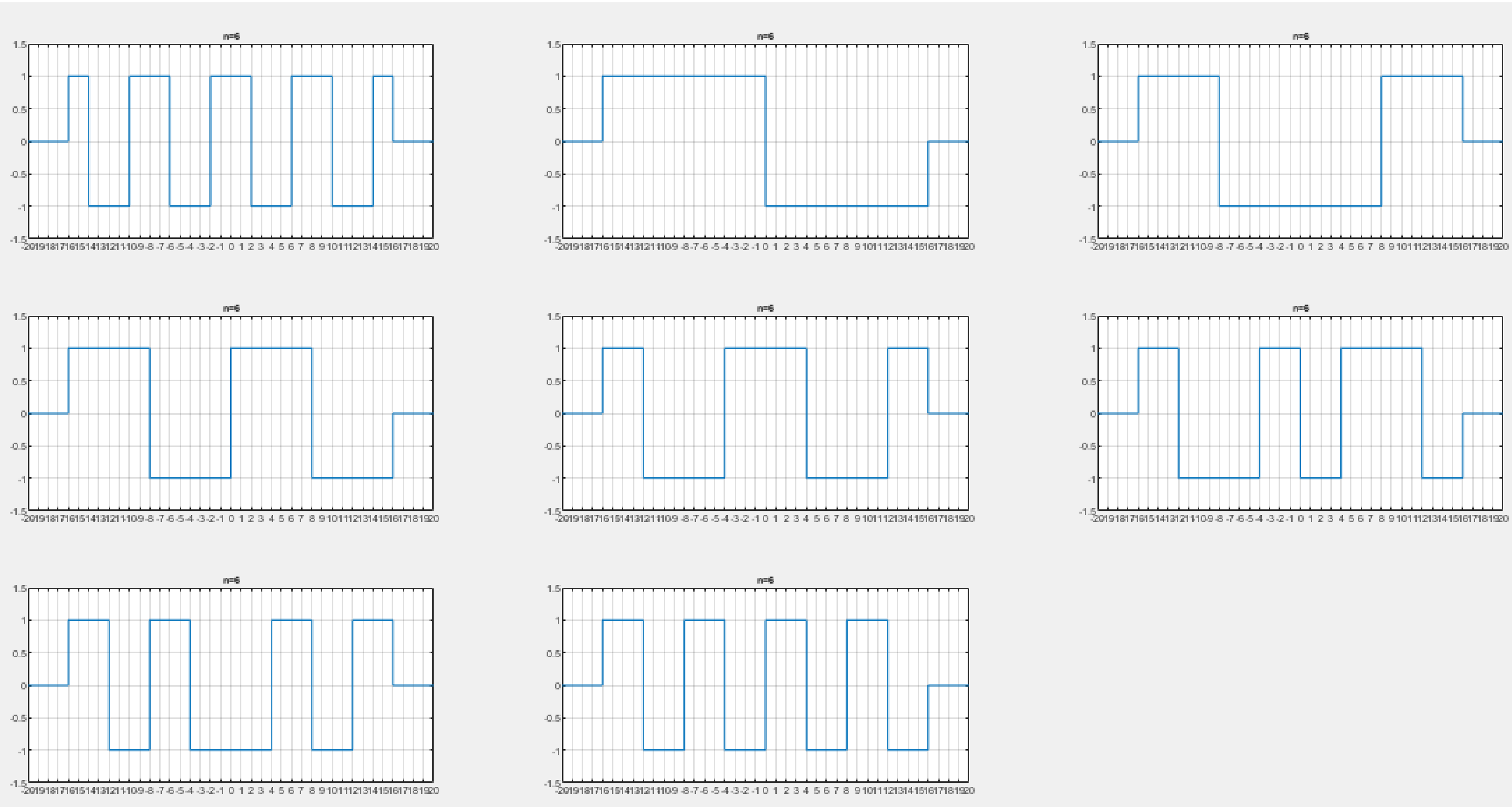

n=4
n=4
n=4
n=4
n=4
n=4
n=4
n=4

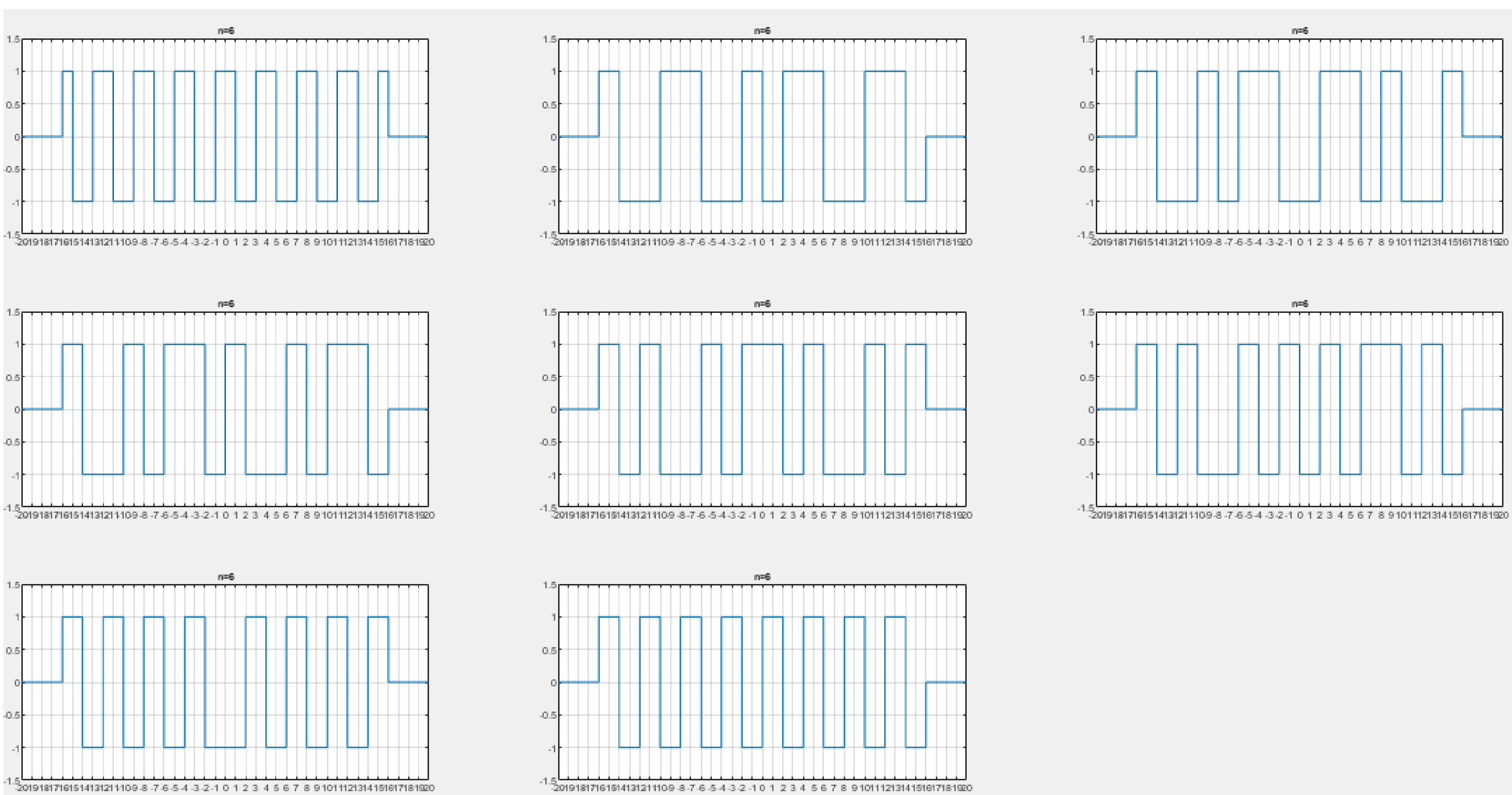

n=5
n=5
n=5
n=5
n=5
n=5
n=5
n=5

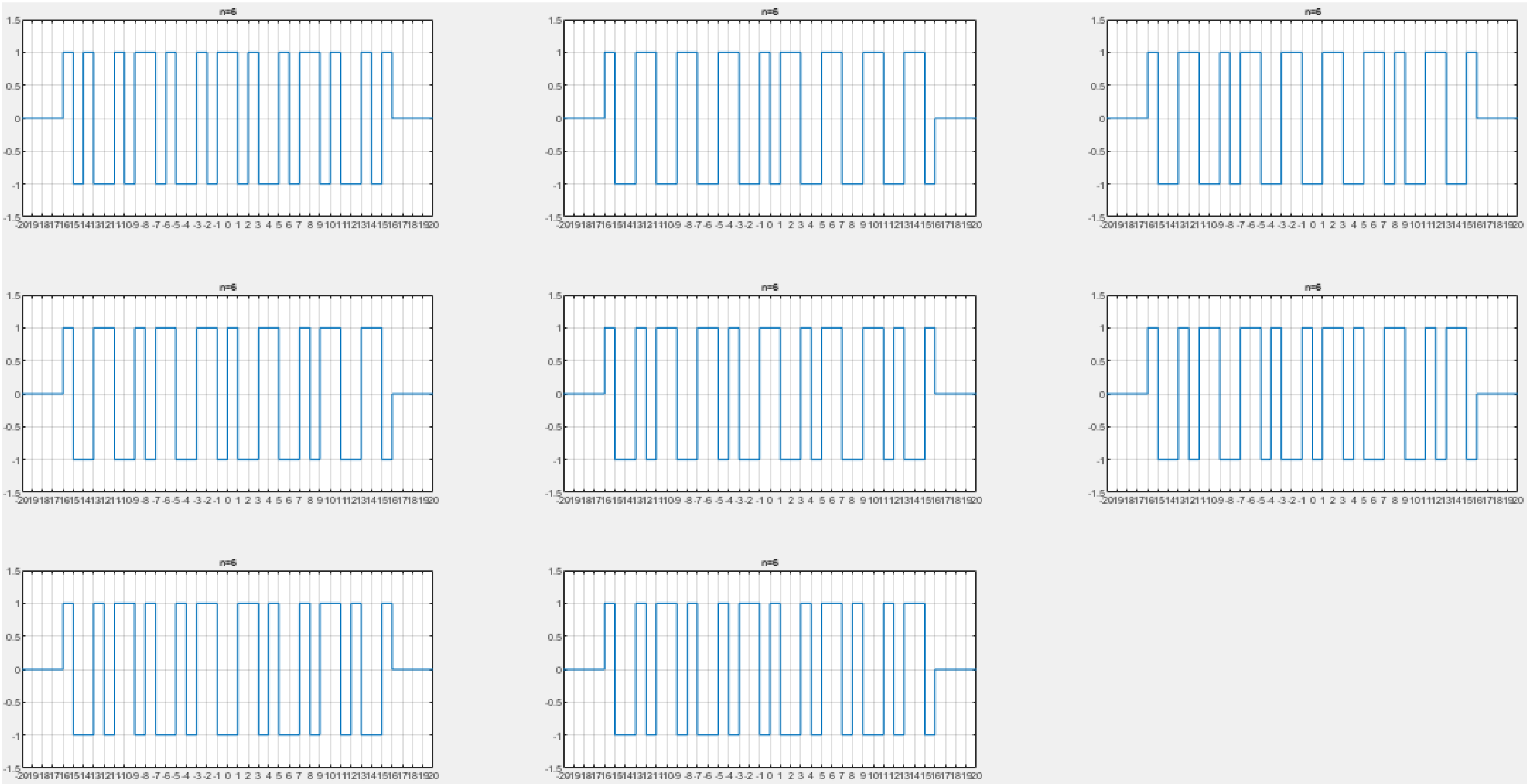

n=5
n=5
n=5
n=5
n=5
n=5
n=5
n=5

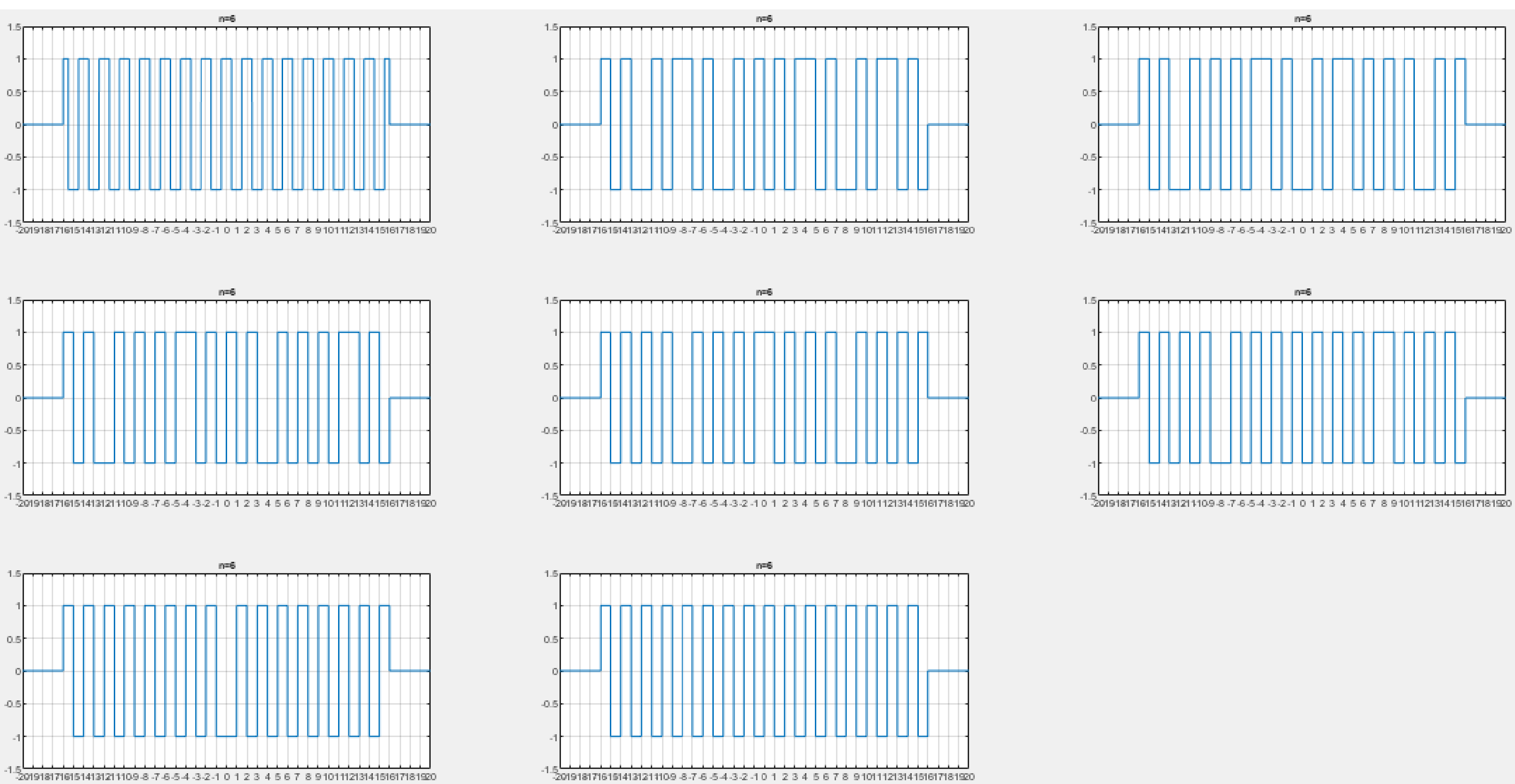

n=5
n=5
n=5
n=5
n=5
n=5
n=5
n=5

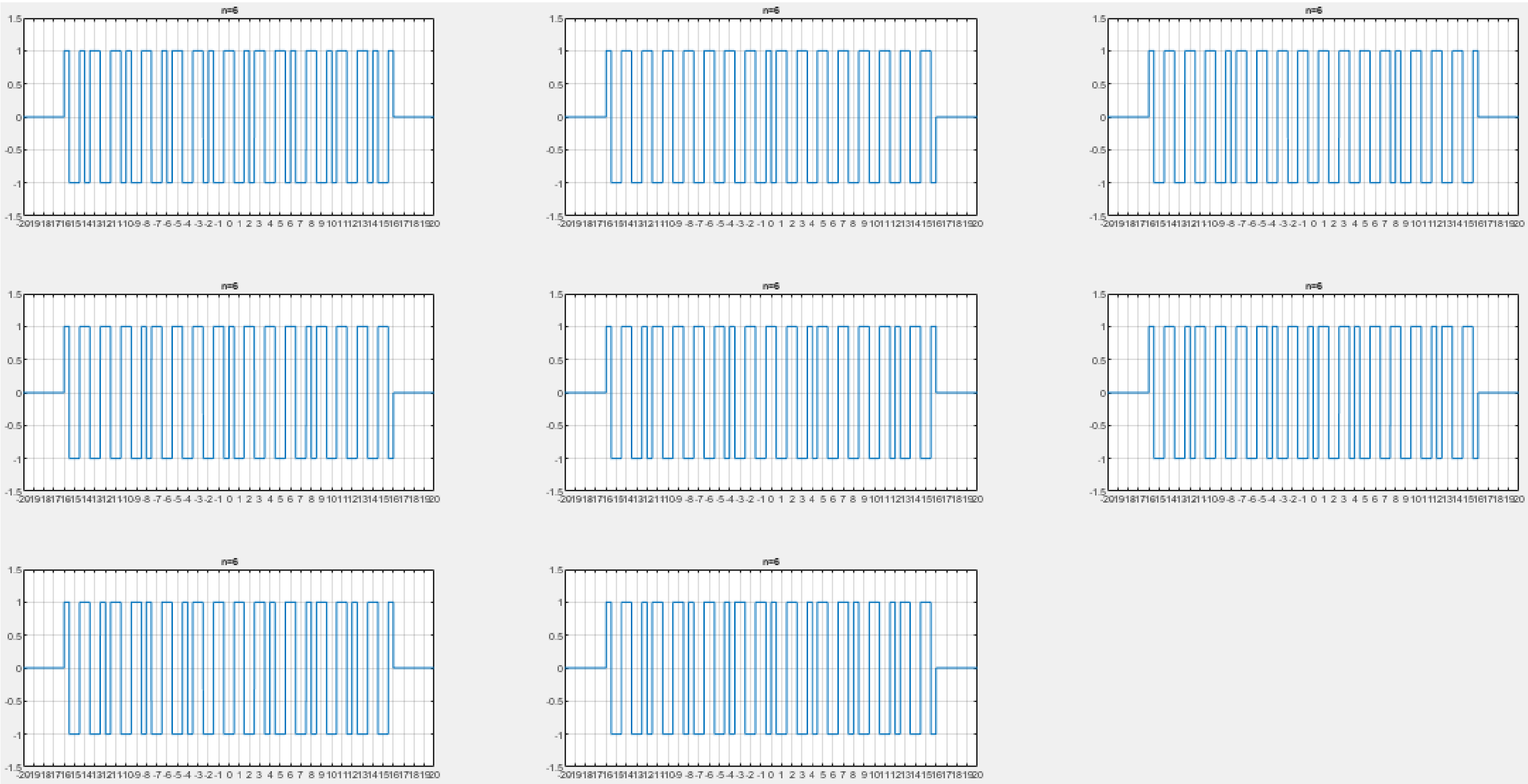

n=5
n=5
n=5
n=5
n=5
n=5
n=5
n=5

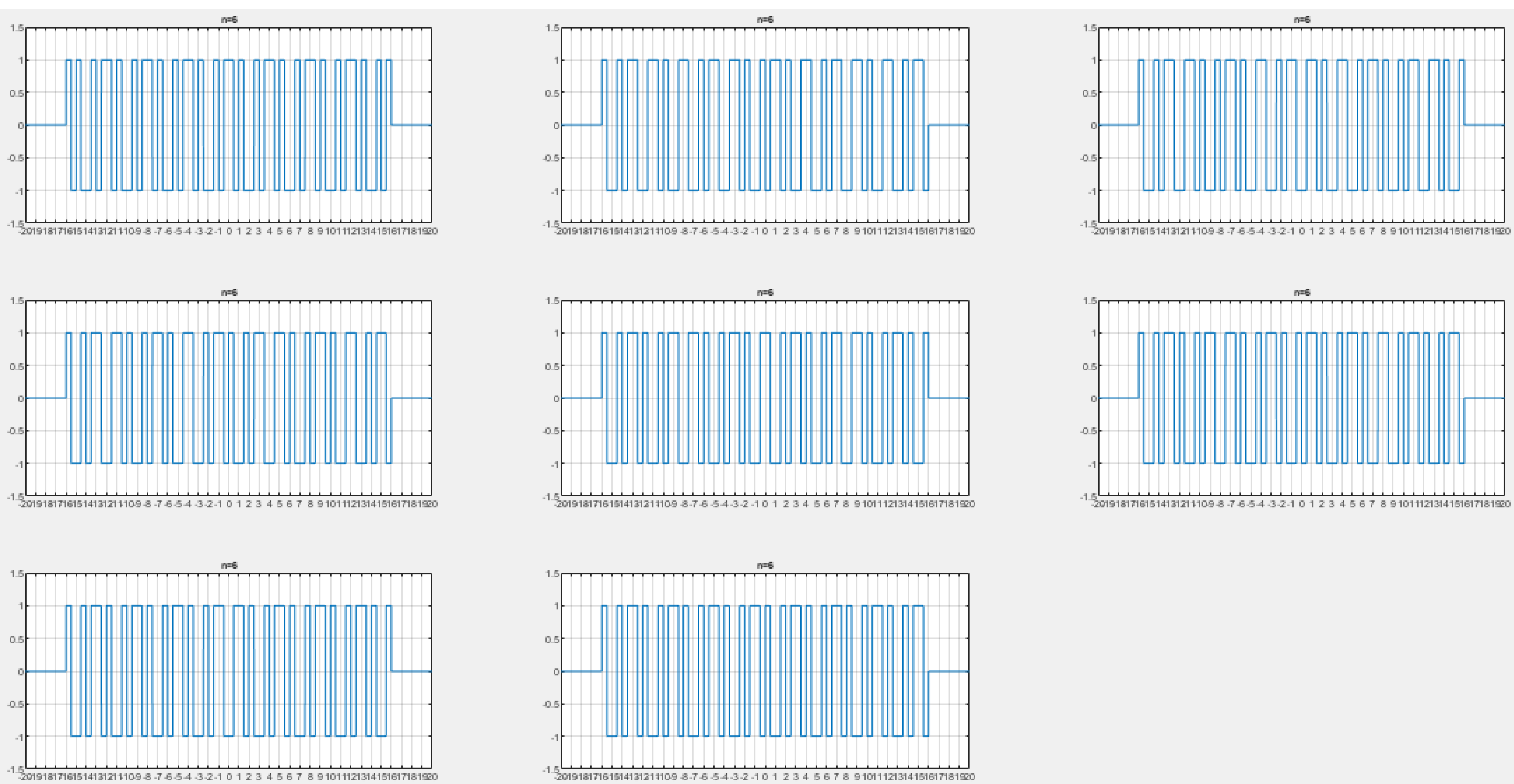

n=6
n=6
n=6
n=6
n=6
n=6
n=6
n=6

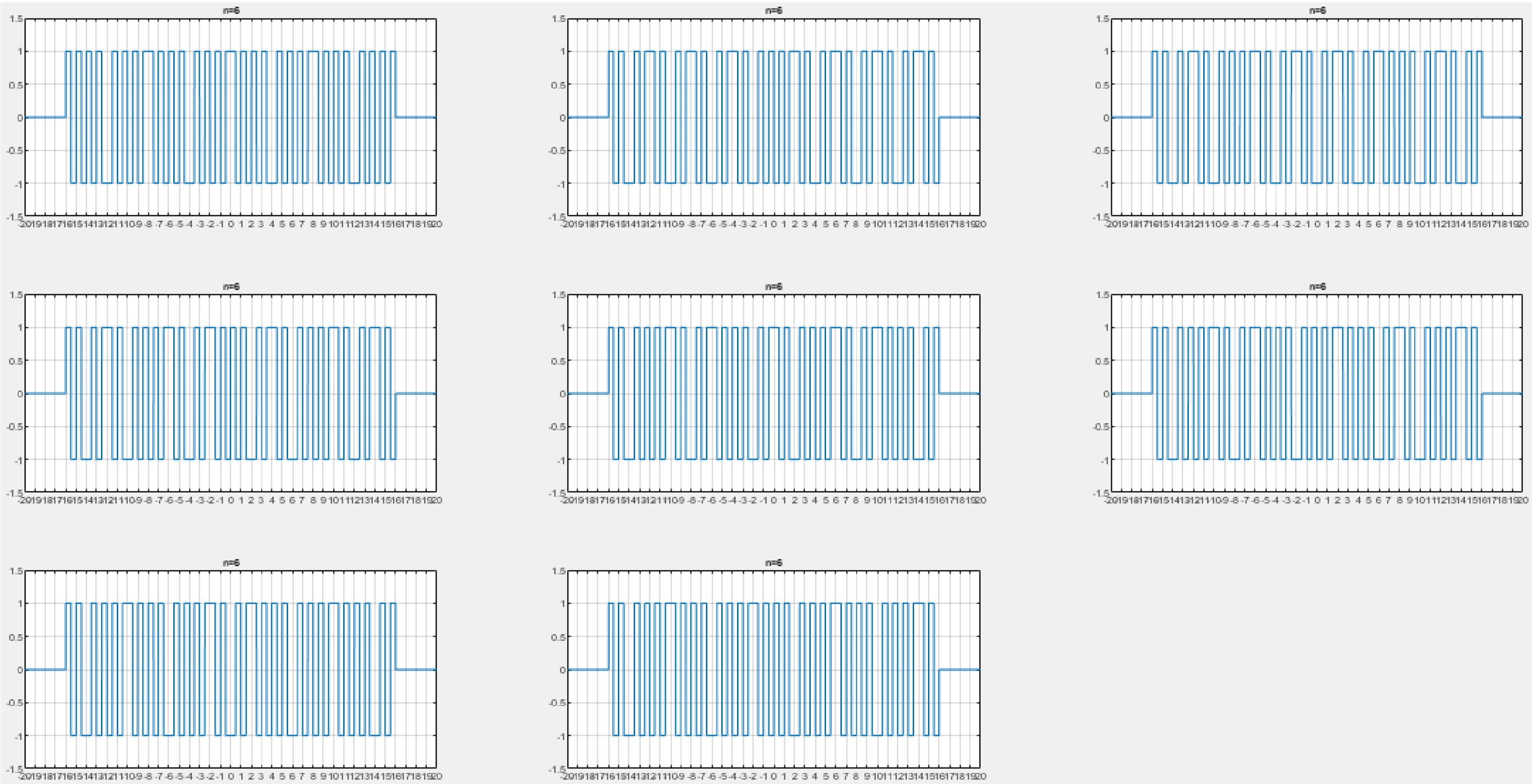

n=6
n=6
n=6
n=6
n=6
n=6
n=6
n=6

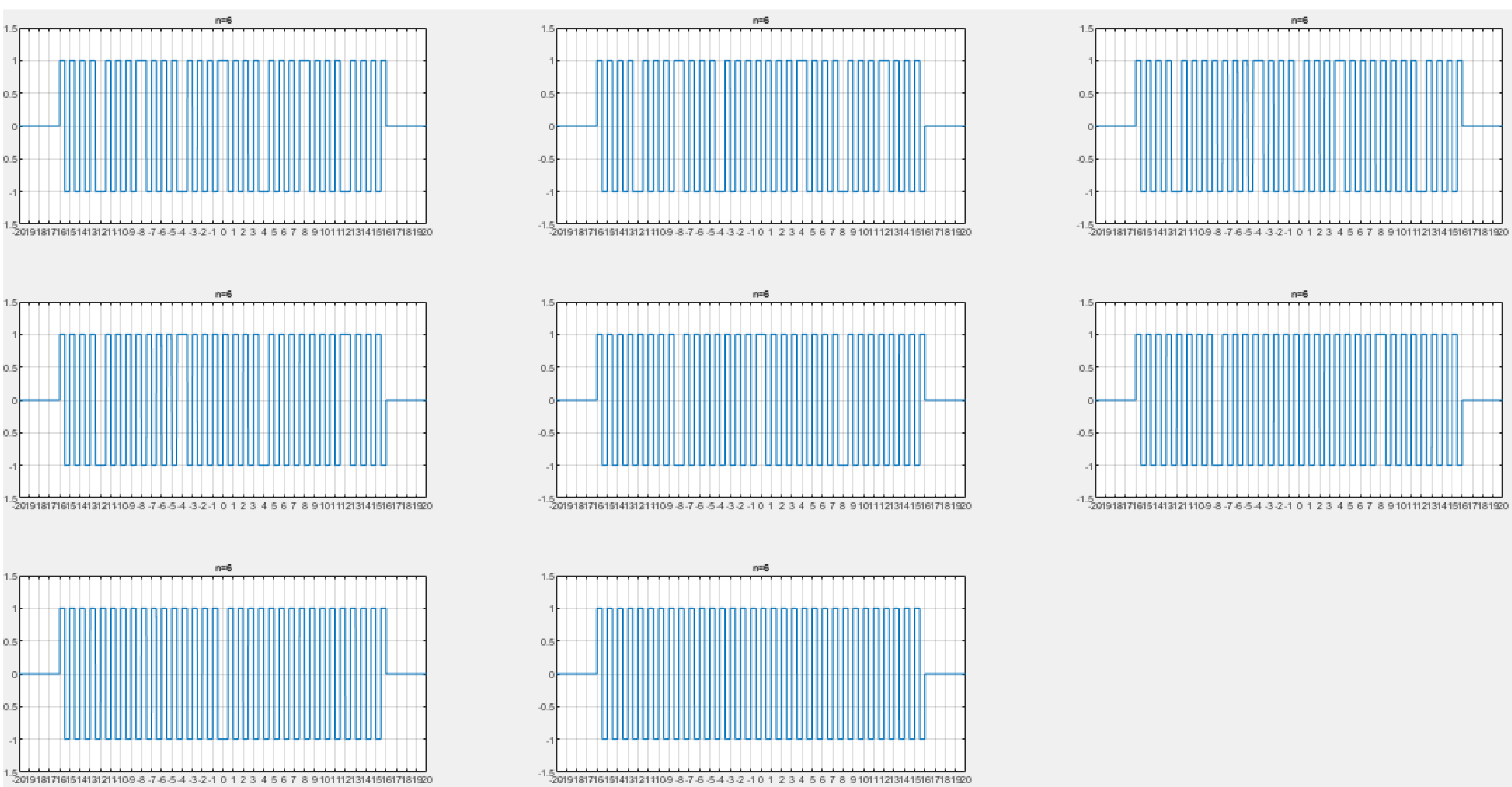

n=6
n=6
n=6
n=6
n=6
n=6
n=6
n=6


n=6
n=6
n=6
n=6
n=6
n=6
n=6
n=6


n=6
n=6
n=6
n=6
n=6
n=6
n=6
n=6

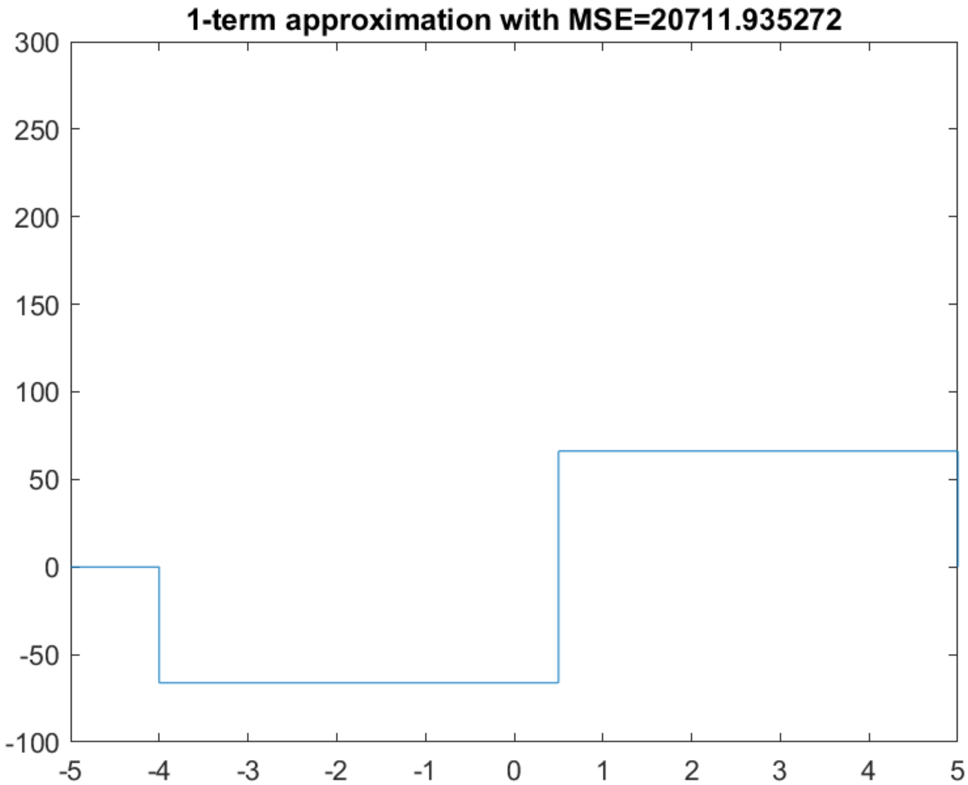

n=6
n=6
n=6
n=6
n=6
n=6
n=6
n=6

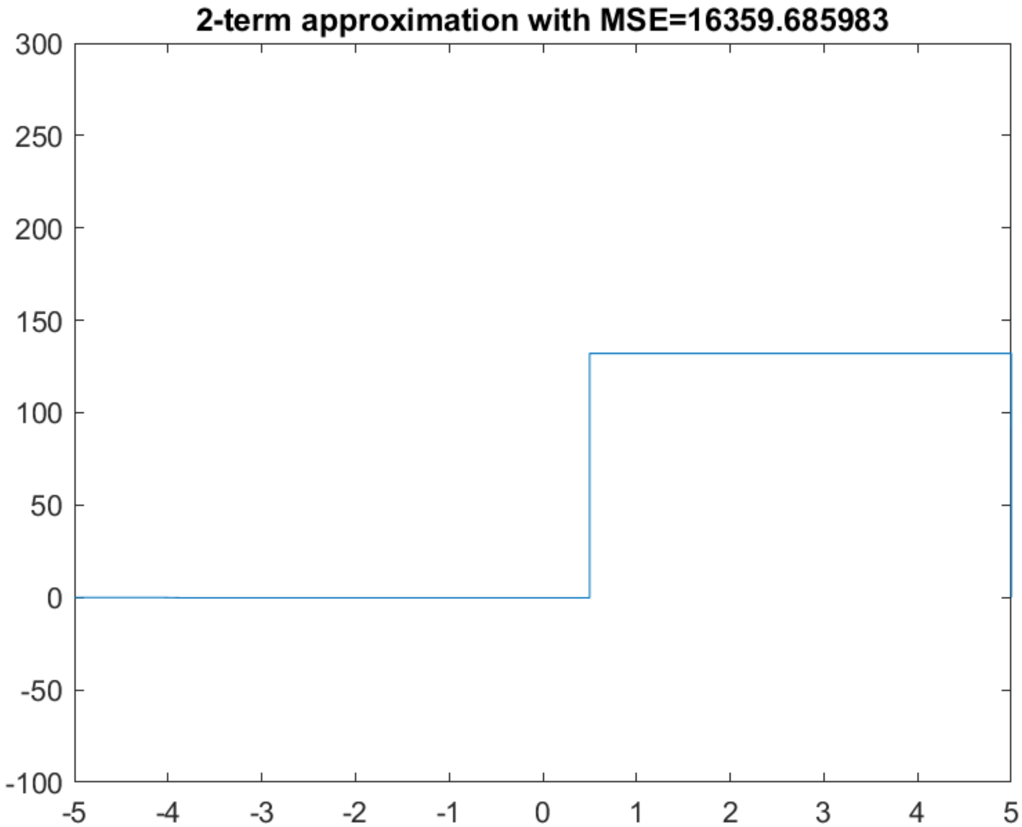

n=6
n=6
n=6
n=6
n=6
n=6
n=6
n=6

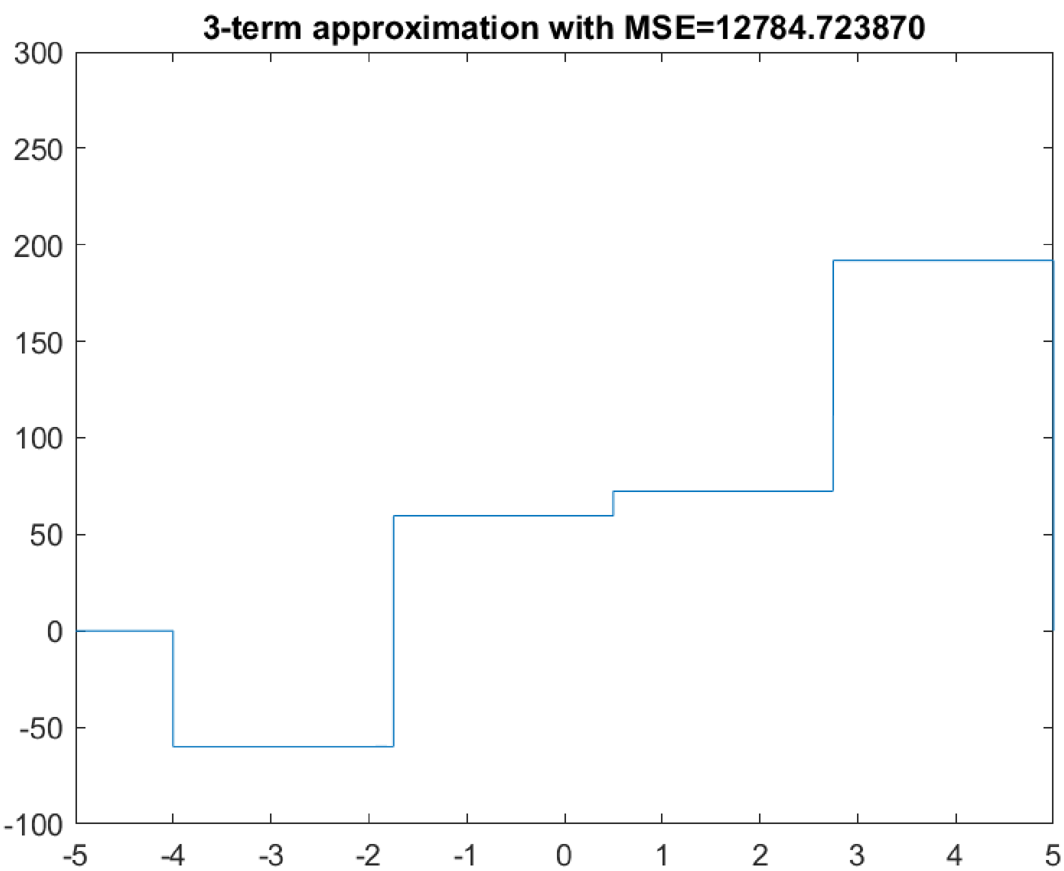

n=6
n=6
n=6
n=6
n=6
n=6
n=6
n=6

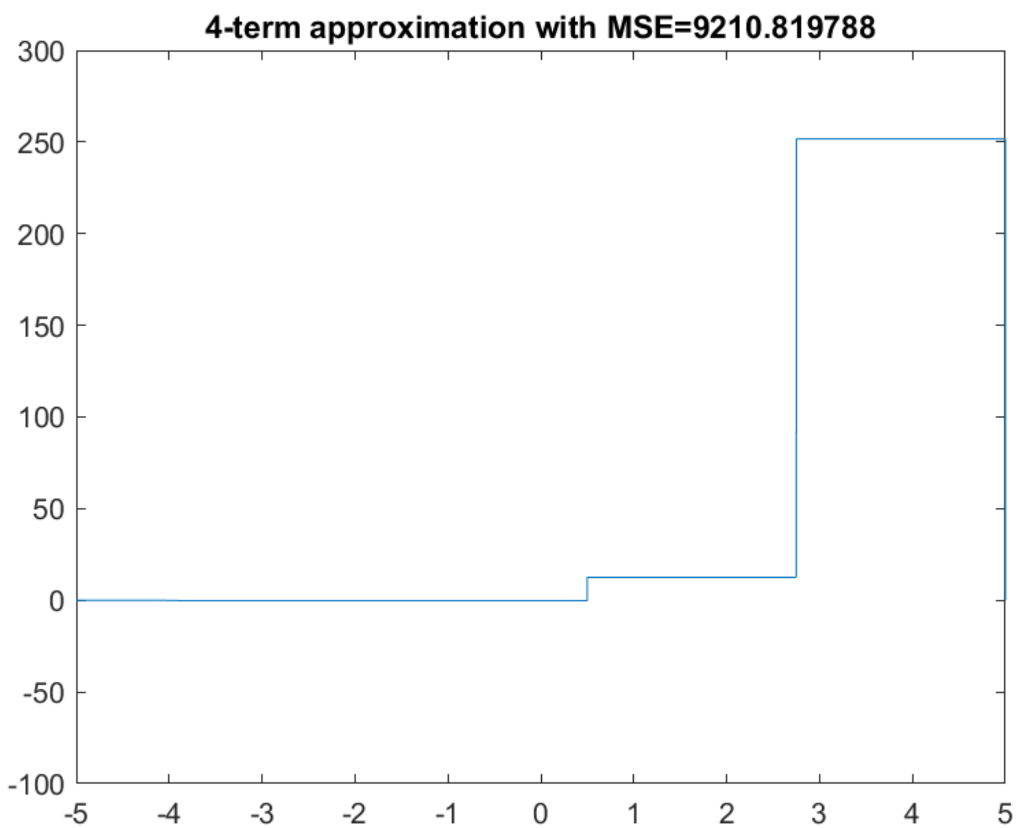
e.

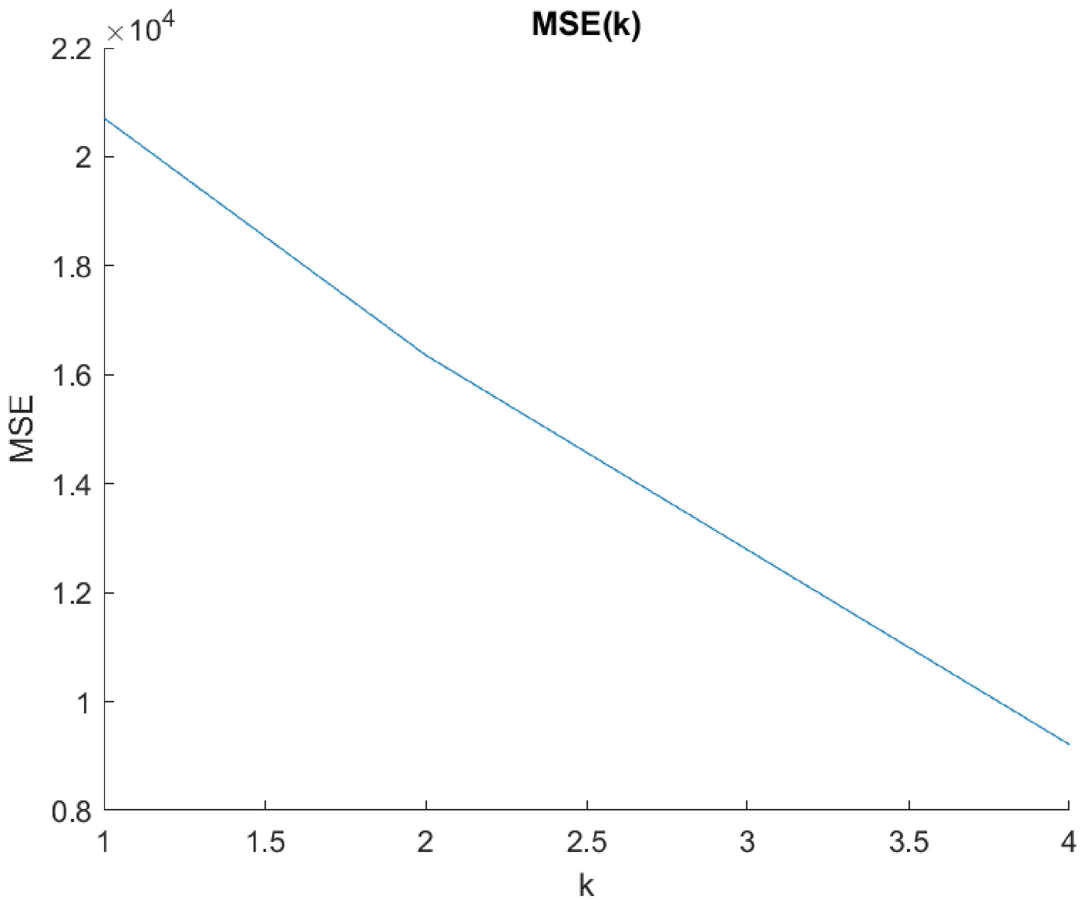
The best k-term approximation is given by the coefficients sorted by their absolute value and the corresponding Walsh-Hadamard functions (the multiplication of them as we learned in class).




2-term approximation with MSE=16359.685983
300
250
200
150
100
50
0
-50
-100
-5
-4
-3
-2
-1
0
1
2
3
4
5


3-term approximation with MSE=12784.723870
300
250
200
150
100
50
0
-50
-100
-5
-4
-3
-2
-1
0
1
2
3
4
5


4-term approximation with MSE=9210.819788
300
250
200
150
100
50
0
-50
-100
-5
-4
-3
-2
-1
0
1
2
3
4
5



As expected the MSE drops down as a function of k. As a result the best k-approximation is for k=4 with MSE=9120.82.