```
function [y] = convFUNC(x, h)
    Nx = length(x);
    Nh = length(h);
    Ny = Nx + Nh - 1; % Length of the output sequence

    % Pad the input signals with zeros to handle edge cases
    x_padded = [zeros(1, Nh-1), x, zeros(1, Nh-1)];
    h_padded = fliplr(h); % Flip h to perform convolution

    % Perform convolution using vectorized operations
    y = zeros(1, Ny);
    for n = 1:Ny
        % Compute the convolution sum for position n using vectorized
 operations
        y(n) = sum(x_padded(n : n+Nh-1) .* h_padded);
    end
end
```

Published with MATLAB® R2022b

```
% Define the time range
t = 0:25; % Adjust the time range as needed

% Initialize the function array
x = zeros(size(t));

% Set the values for the function
for i = 1:length(t)
    if t(i) < 2 % Before the delay
        x(i) = 0;
    else
        % Compute the index within the repeating pattern
        idx_within_pattern = mod(t(i) - 2, 6);

        % Set the values based on the pattern
        if idx_within_pattern < 3
            x(i) = 1;
        else
            x(i) = 0;
        end
    end
end

% Call ConvFUNC to compute the convolution of x with itself
y = convFUNC(x, x);

% Define the time range for y
ty = 0:(length(x) + length(x) - 2); % Length of convolution result

% Create a subplot with two rows and one column
subplot(2, 1, 1); % Plot in the first row

% Plot the function x
stem(t, x);
xlabel('n');
ylabel('x[n]');
title('x[n]');

% Plot the function y in the second row
subplot(2, 1, 2); % Plot in the second row

% Plot the convolution y
stem(ty, y);
xlabel('n');
ylabel('y[n]');
title('Convolution of x with itself');
```

Published with MATLAB® R2022b

```
% Define the time array # and sampling interval Si
Si = 0.25; % Sampling interval
tau = -10:Si:10;
tau_y = -20:Si:20; % Extend the range for convolution result

% Define the unit-step functions u(t)
u = @(t) (t >= 0);

% Define #(t) = u(t+5) - u(t-5)
xi_t = @(t) u(t+5) - u(t-5);

% Define #(t) = u(t+2.5) - u(t-2.5)
eta_t = @(t) u(t+2.5) - u(t-2.5);
xi_tau = xi_t(tau);
eta_tau = eta_t(tau);
eta_tau_flipped = eta_t(-tau);

psi_tau = convFUNC(xi_tau, eta_tau);

figure;

for ii = 1:length(tau)
    % Update the plot
    subplot(2, 2, 1);
    plot(tau, xi_tau, 'b', 'LineWidth', 1.5);
    title('\xi(\tau)');
    xlabel('\tau');
    ylabel('\xi(\tau)');
    xlim([tau(1), tau(end)]);
    ylim([-0.25, 1.25]);

    subplot(2, 2, 2);
    plot(tau, eta_tau, 'g', 'LineWidth', 1.5);
    title('\eta(\tau)');
    xlabel('\tau');
    ylabel('\eta(\tau)');
    xlim([tau(1), tau(end)]);
    ylim([-0.25, 1.25]);

    subplot(2, 2, 3);

    plot(tau, xi_tau, 'b', 'LineWidth', 1);
    hold on;
    plot(tau - 20 + ii, eta_tau_flipped, 'g', 'LineWidth', 1.5);
    hold off;

    title('Sliding \eta(\tau)');
    xlabel('\tau');
    xlim([-20, 20])
    ylabel('\eta(\tau)');
    ylim([-0.5, 1.5]); % Set y-axis limits
```

```
    subplot(2, 2, 4);

    plot(tau_y(1:4:2*ii-1), psi_tau(1:4:2*ii-1) / 4, 'r', 'LineWidth', 1.5);
    title('\psi(\tau)');
    xlabel('\tau');
    ylabel('\psi(\tau)');
    xlim([-20, 20]); % Set x-axis limits
    ylim([-3, 6]);

    drawnow;

end
```

Published with MATLAB® R2022b

```
% [y, Fs] = audioread("TotalNumber . flac");
% soundsc(y, Fs);
% ro = [2, 0];
% lambda = [3, 5, 7, 8, 9];
% delta = mod(22102640, 7);
% delta_ro = mod(delta, length(ro)) + 1;
% delta_lamba = mod(delta, length(lambda)) +1;
% %according to the deltas, n1 and n2 are calculated
% n1 = 2;
% n2 = 3;

[my, F_s] = audioread("ID_record.flac");
soundsc(my, F_s);
t = 0:length(my) - 1 / F_s;
subplot(2, 1, 1);
plot(t, my)
title("Recording")

% start_time = 0.8 * 10^4; % Start time of the instance
% end_time = 1.4 * 10^4;   % End time of the instance
%
% % Extract the instance into another .flac file
% instance_signal = my(t >= start_time & t <= end_time);
% audiowrite('n1_instance.flac', instance_signal, F_s);

[n_1, fs] = audioread("n1_instance.flac");
time = 0:length(n_1) - 1 / fs;
subplot(2,1,2);
plot(time, n_1);
title("n1 sample");

figure;
n1_new = fliplr(n_1);
my_new = my';
n1_newer = n1_new';

psi_new = convFUNC(my_new, fliplr(n1_newer));
subplot(3,1,1);
plot(abs(psi_new));
ylabel("\psi [x]");
xlabel("x");
subplot(3,1,2);
plot(abs(psi_new.^2));
ylabel("\psi [x]^2");
xlabel("x")
subplot(3,1,3);
plot(abs(psi_new.^4))
ylabel("\psi [x]^4");
xlabel("x");
```

Published with MATLAB® R2022b

```
[audio_array, Fs] = audioread("TotalNumber . flac");
% soundsc(audio_array, Fs);
audio_len = length(audio_array);

p_signal = (1/audio_len) * sum(audio_array.^2);

disp(p_signal);

SNR = 10;

p_noise = p_signal / 10;

display(p_noise);

rng(5);
awgn = sqrt(p_noise) .* randn([audio_len, 1]);

noisy_audio = audio_array + awgn;

% soundsc(noisy_audio, Fs); pause(10);
subplot(3, 1, 1);
plot(noisy_audio);
title("SNR = 10");
xlabel("t");
ylabel("noisy audio");

%SNR = 0.1
p_noise_2 = p_signal / 0.1;
display(p_noise_2);
rng(5);
awgn_2 = sqrt(p_noise_2) .* randn([audio_len, 1]);

noisy_audio_2 = audio_array + awgn_2;

% soundsc(noisy_audio_2, Fs); pause(10);
subplot(3, 1, 2);
plot(noisy_audio_2);
title("SNR = 0.1");
xlabel("t");
ylabel("noisy audio");


%SNR = 0.001
p_noise_3 = p_signal / 0.001;
display(p_noise_3);
rng(5);
awgn_3 = sqrt(p_noise_3) .* randn([audio_len, 1]);

noisy_audio_3 = audio_array + awgn_3;

% soundsc(noisy_audio_3, Fs);
subplot(3, 1, 3);
```

```
plot(noisy_audio_3);
title("SNR = 0.001");
xlabel("t");
ylabel("noisy audio");
```

Published with MATLAB® R2022b

```
[audio_array,F_audio] = audioread("TotalNumber . flac");
[filter, F_filter] = audioread("2. flac");
audio_len = length(audio_array);

for i = 0:9
    SNR= 0.01-i*0.001;
        p_signal = (1/audio_len) * sum(audio_array.^2);
    p_noise = p_signal / SNR;
    rng(5)
    awgn = sqrt(p_noise) .* randn([audio_len, 1]);

    noisy_audio = awgn + audio_array;
end

    t_noisy_audio = noisy_audio';
    t_filter = filter';

for a = 1:10
    %correlation part

    detect = convFUNC(t_filter, t_noisy_audio(a));
    abs_detect = abs(detect);
    subplot(5,2,a)
    plot(abs_detect);
end
```

Published with MATLAB® R2022b