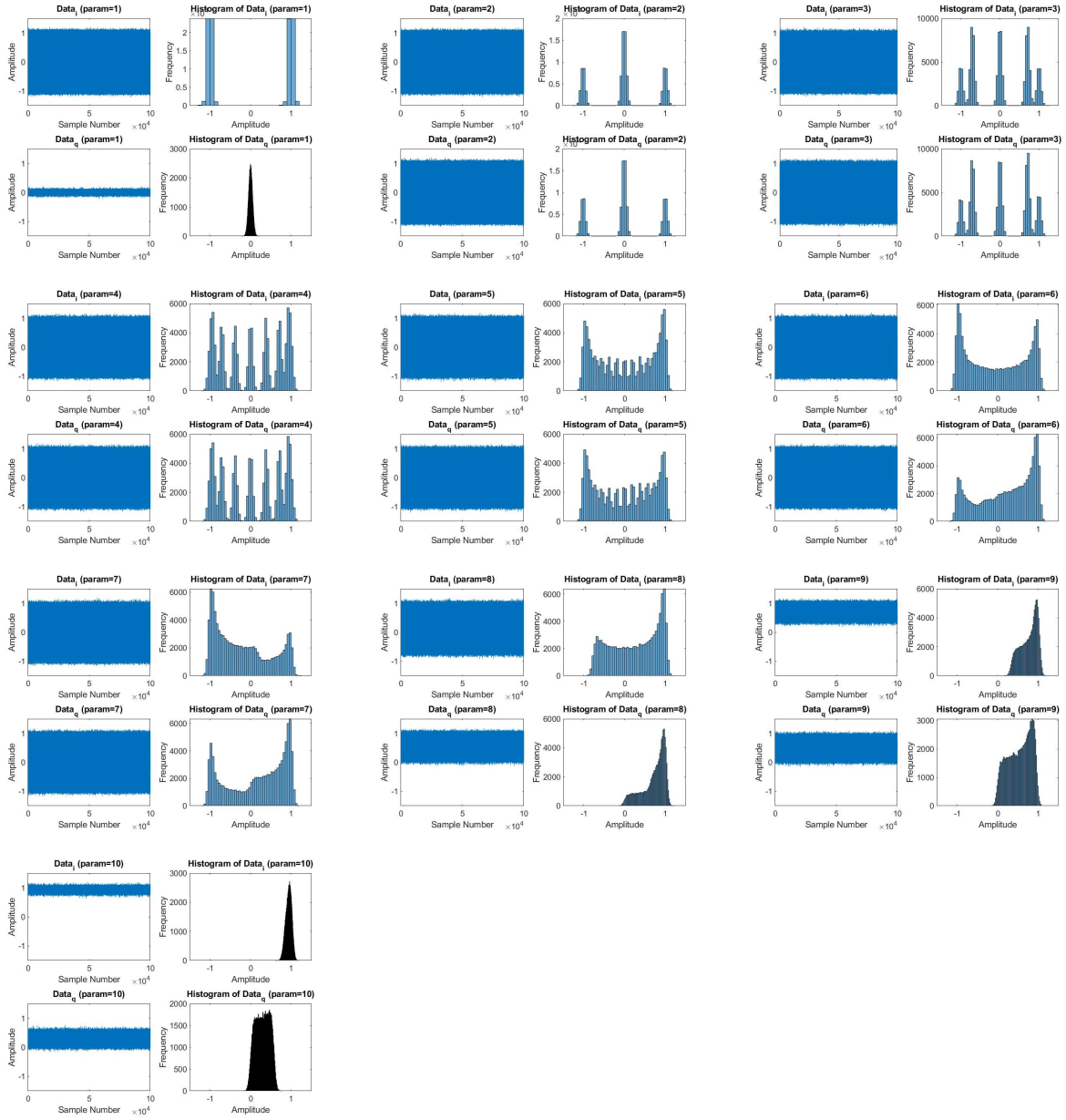
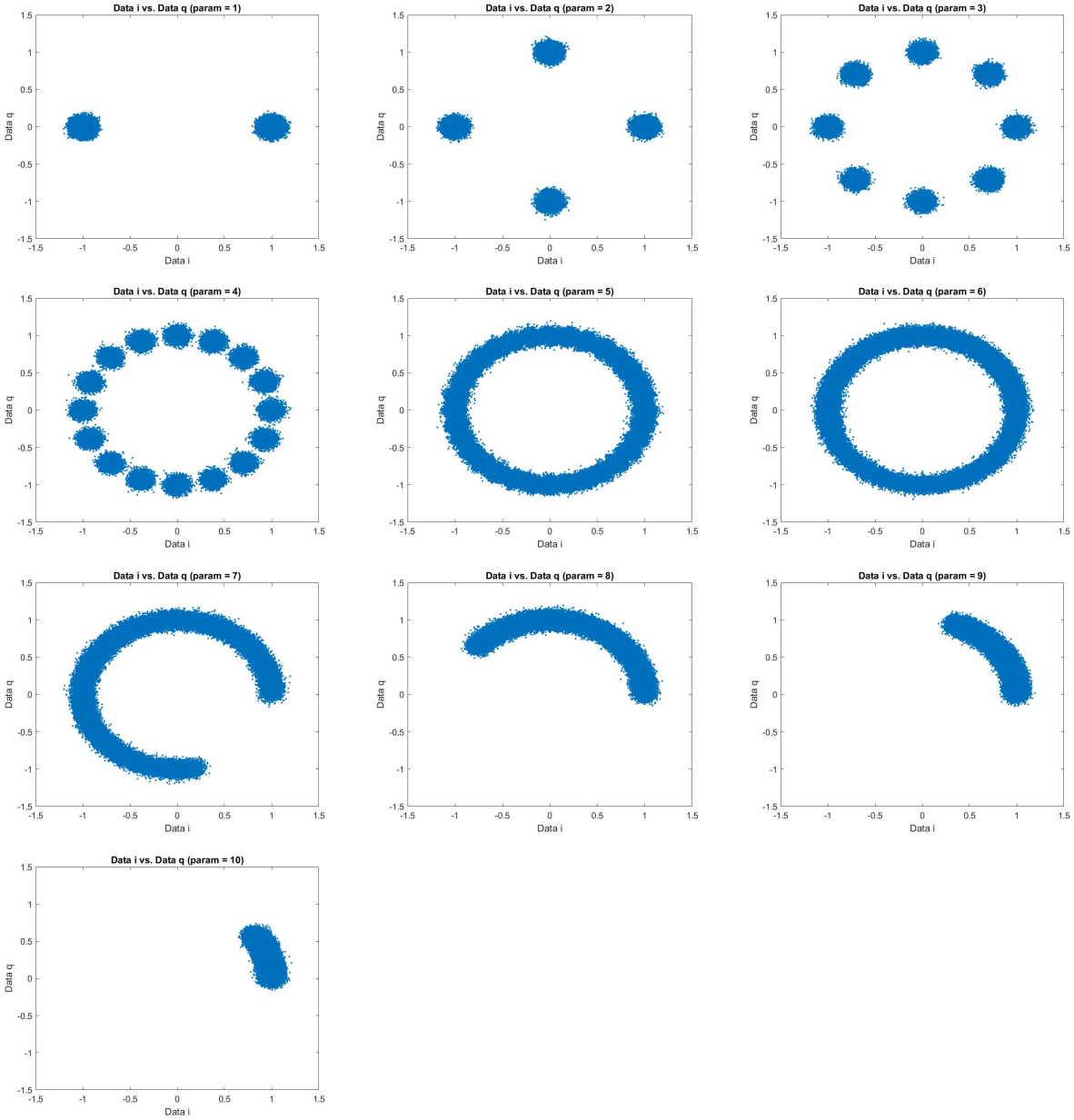
Matlab Test for DCommLab

Task 1

Comparison of different parameters:



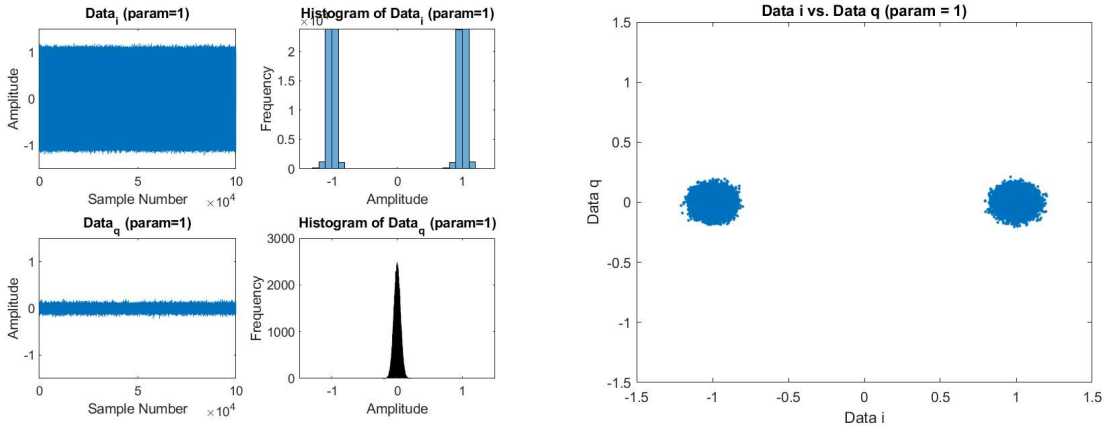
Plotting them against each other:



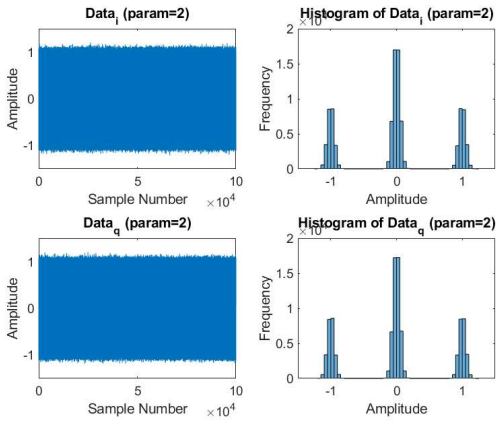
First I compared the signals themselves and their histograms for the parameters 1-10 as i needed an overview of what "small" integer exactly means. From the histograms and the I-Q plots, its clear that after param=5 there is not much information to be gained, so i focused on the first 5 signals.

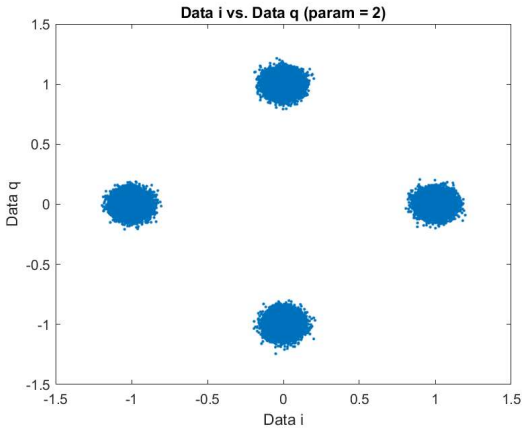
Parameter 1-5:

2 ASK, 2 PSK, BPSK for param = 1

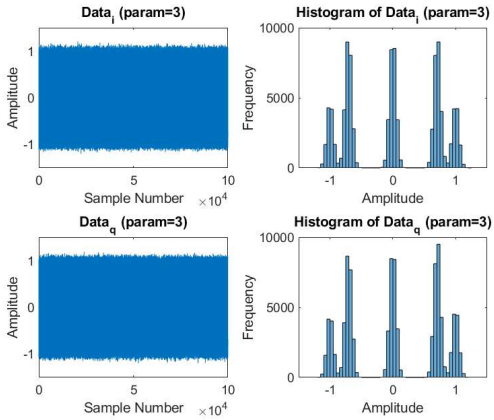


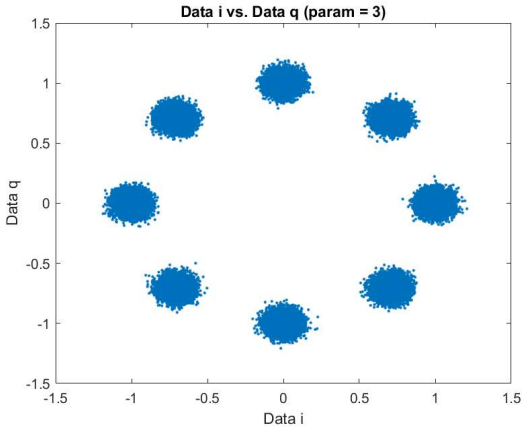
4PSK, QPSK for param = 2



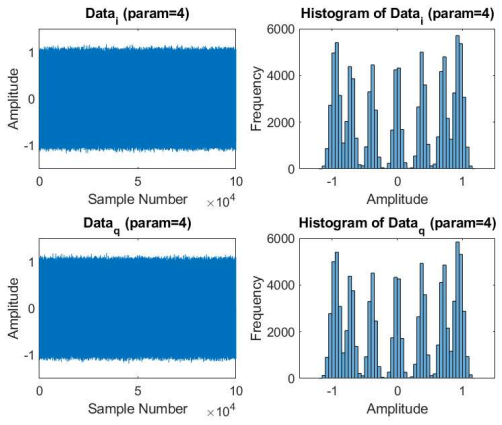


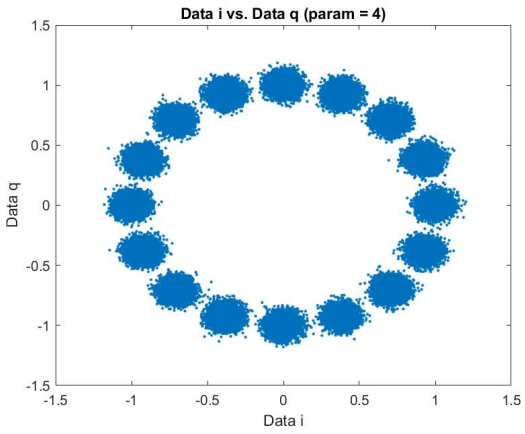
8PSK for param = 3



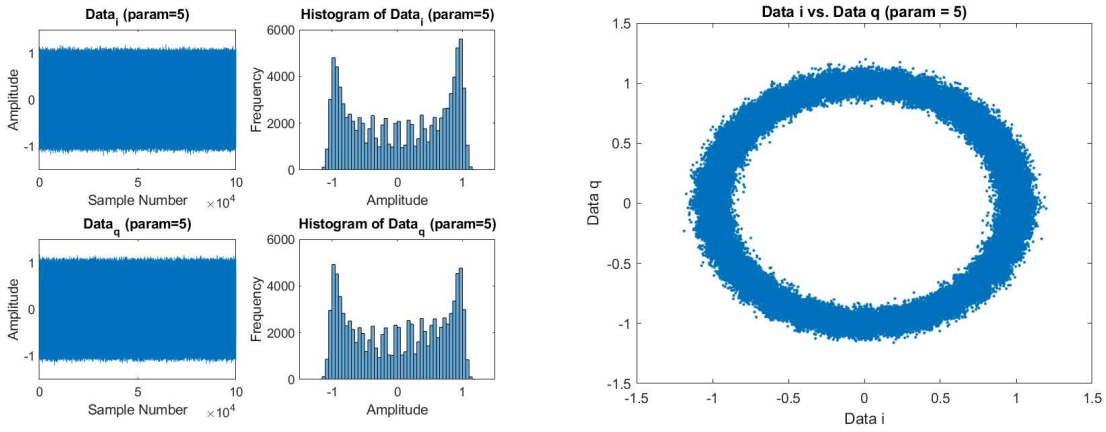


16PSK for param = 4





32PSK ???; --> no clear pattern for param = 5



So to summarize, the two signals data_i and data_q seem to be the in-phase and quadrature components of the signal. From the plots in the complex plane we can see that they form M-PSK constellations. The coordinates of the points can be seen in the peaks of the histograms as well. The parameter that is given to the unknown function, seems to be the amount of bits that need to be encoded.

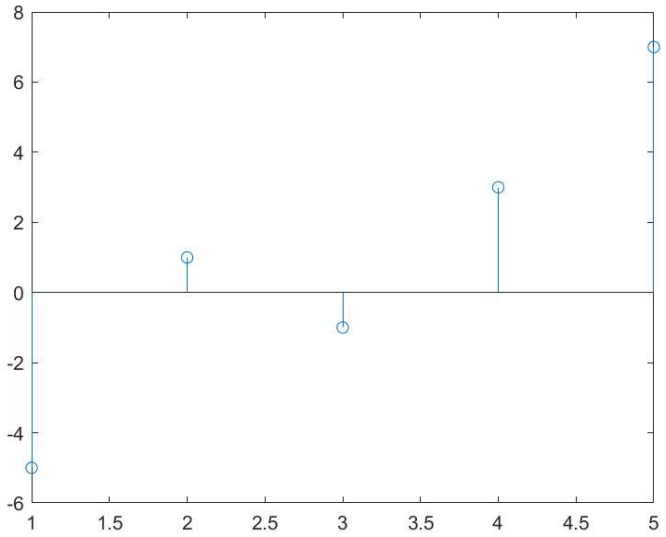
So for param = 1, we get a 2 PSK signal, that is able to transfer 1 bit.
For param = 2, we get a 4 PSK signal, that is able to transfer 2 bits.
And so on.
For higher parameters the clear distinction between the points in the constellation is not visible anymore, so the signal might not be able to transfer more than 4 bits.

Assumptions:

1. The samples in the vectors are measured at equal intervals in time and are ordered correctly.
2. Data_i and data_q are synced with each other in time.
3. Data is already cleaned (no outliers).

Task 2

Plot of signal to send:



Code

```
%--------------------------------------------------------------------------
% Lab Course Digital Communications
%--------------------------------------------------------------------------
% Preparatory Test
% Winter 2022/23
% IDC/FAU
%--------------------------------------------------------------------------

task1();
task2();

%--------------------------------------------------------------------------
% Task 1: Data Analysis
%--------------------------------------------------------------------------
function task1()

    % Call the function dico_lab_signal(). Note, the function
    % requires a (small!) positive integer as input and returns
    % _two_ output data vectors (denoted as data_i and data_q).

    % Analyse the data using MATLAB, i.e. generate reasonable plots
    % and histograms of the two individual data vectors and compare them.
    %
    % Vary the input variable. How does this affect the data obtained
    % from the function?
```

```
for param = 1:10
    analyze_dico_lab_signal(param);
end
function analyze_dico_lab_signal(param)
    % Function to analyze the dico_lab_signal function output

    folder_path = "images";

    % Call the dico_lab_signal function
    [data_i, data_q] = dico_lab_signal(param);

    % Plot data_i and data_q
    figure;
    subplot(2,2,1);
    plot(data_i);
    ylim([-1.5, 1.5]);
    xlabel('Sample Number');
    ylabel('Amplitude');
    title(sprintf('Data_i (param=%d)', param));

    subplot(2,2,2);
    histogram(data_i);
    xlim([-1.5, 1.5]);
    xlabel('Amplitude');
    ylabel('Frequency');
    title(sprintf('Histogram of Data_i (param=%d)', param));

    subplot(2,2,3);
    plot(data_q);
    ylim([-1.5, 1.5]);
    xlabel('Sample Number');
    ylabel('Amplitude');
    title(sprintf('Data_q (param=%d)', param));

    subplot(2,2,4);
    histogram(data_q);
    xlim([-1.5, 1.5]);
    xlabel('Amplitude');
    ylabel('Frequency');
    title(sprintf('Histogram of Data_q (param=%d)', param));

    % Save the figure as a PNG image
    saveas(gcf, fullfile(folder_path, sprintf('signal_param_%d.png', pa
```

```
        close;
    end

    % Plot data_q over data_i. What can you see?
    %

    for param = 1:10
        plot_data_iq(param);
    end

    function plot_data_iq(param)

        folder_path = "images";

        % Get the data
        [data_i, data_q] = dico_lab_signal(param);

        % Plot data_i over data_q
        plot(data_i, data_q, '.');
        xlim([-1.5, 1.5]);
        ylim([-1.5, 1.5]);
        xlabel('Data i');
        ylabel('Data q');
        title(sprintf('Data i vs. Data q (param = %d)', param));

        saveas(gcf, fullfile(folder_path, sprintf('data_iq_param_%d.png', p
        close;
    end

    % What assumptions do you have about the data generated by this functio
    %
    % 1. The samples in the vectors are measured at equal intervals in time
    % and are ordered correctly
    % 2. Data_i and data_q are synced with each other in time
    % 3. Data is already cleaned (no outliers)
    % Save all these plots for your upload to StudOn!
end


%-------------------------------------------------------------------------
% Task 2: Bit Mapping
%-------------------------------------------------------------------------
function task2()
```

```
% Call the function dico_lab_bit_mapping. It returns two outputs:
% the points of an 8-ASK constellation and their respective bit labels.

[points, labels] = dico_lab_bit_mapping;

%     points
%     labels
% Complete the matrix with bit labels so that a Gray mapping results.

labels(:,2) = [0;0;1];
labels(:,4) = [0;1;0];

complete_matrix_with_bit_labels_here = labels;

% The following sequence of bits shall be mapped onto points of the
% signal constelltion:

bit_sequence = [ 0 0 1 1 1 0 0 1 0 1 1 1 1 0 0 ];

% Give the sequence of signal constellation points.
result = [];
labels = labels.';

% loop over 3 bits at a time from the bit_sequence
for i = 1:3:length(bit_sequence)
    bits = bit_sequence(i:i+2);
    % loop over the labels, if the 3 bits equal the labels, add the
    % point at that index to the result
    for j = 1:length(points)
        if bits == labels(j,:)
            result(length(result)+1) = points(j);
        end
    end
end

signal_constellation_points = result;

% Create a plot with the sequence of signal points (use stem)
stem(signal_constellation_points)
% Save the plot for your upload to StudOn!
saveas(gcf, fullfile("images", 'constellation.png'));
close;
```

```
end

%-------------------------------------------------------------------------
% Final Task: pdf Document
%-------------------------------------------------------------------------
% Create a single pdf document with your plots and answers.
% Also include this m-file with all your code.

%-------------------------------------------------------------------------
% end
%-------------------------------------------------------------------------
```