Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Высшая школа программной инженерии

Методы анизотропной и пороговой статистической фильтрации

Расчетное задание №3 по курсу

Цифровая обработка и передача многомерных сигналов

Выполнил: Leliukhin студент группы 43504/2, Лелюхин Д.О.

Руководитель: Сараджишвили С.Э.

Санкт-Петербург
2019

Задание

Есть исходная матрица:

0	9	18	15	27	24	16	19	11	5	10	0	0
12	21	26	45	50	53	79	51	39	30	8	12	8
19	36	57	78	100	105	124	131	82	80	57	23	14
28	53	91	113	107	160	160	125	137	80	53	58	16
23	57	102	151	133	171	220	188	162	149	85	63	44
29	82	109	144	162	217	261	270	195	129	146	96	30
13	77	119	161	175	221	323	272	290	230	152	174	107
6	80	116	143	194	249	357	330	242	233	177	188	97
6	54	99	125	147	209	293	311	264	158	143	183	113
0	37	75	88	133	139	268	243	142	171	101	124	81
0	17	47	75	95	111	137	226	153	95	108	141	121
0	9	37	34	60	92	126	139	151	131	95	106	77
0	0	15	20	52	41	78	113	80	104	75	52	42
0	0	12	12	15	46	31	48	40	55	35	0	0

В нее вносятся две точечные помехи:

0	9	18	15	27	24	16	19	11	5	10	0	0
12	21	26	45	50	53	79	51	39	30	8	12	8
19	36	57	78	100	105	124	131	82	80	57	23	14
28	53	91	113	107	160	160	125	137	80	53	58	16
23	57	102	151	133	171	220	188	162	149	85	63	44
29	82	109	144	**400**	217	261	270	195	129	146	96	30
13	77	119	161	175	221	323	272	290	230	152	174	107
6	80	116	143	194	249	357	330	**450**	233	177	188	97
6	54	99	125	147	209	293	311	264	158	143	183	113
0	37	75	88	133	139	268	243	142	171	101	124	81
0	17	47	75	95	111	137	226	153	95	108	141	121
0	9	37	34	60	92	126	139	151	131	95	106	77
0	0	15	20	52	41	78	113	80	104	75	52	42
0	0	12	12	15	46	31	48	40	55	35	0	0

Необходимо обработать матрицу с помехами двумя способами:

1) методом анизотропной фильтрации:

$$\frac{1}{22}\begin{pmatrix} 2 & 3 & 2 \\ 2 & 4 & 2 \\ 2 & 3 & 2 \end{pmatrix}$$

2) методом статистической пороговой фильтрации:
   m = 1.25, N = 3

В обоих способах используется нерекурентная фильтрация.
Крайние элементы отбрасываются (не обрабатываются).
Результаты фильтрации представить таблично и графически.

Текст программы

Программы для вычисления реакции системы написаны на языке C++ 11, в среде разработки Qt Creator 5.9.2. Программа для визуализации данных написана с использованием среды MatLab R2016a.

1) Применение фильтров к зашумленному сигналу

```
#include <stdio.h>
#include <math.h>
#define ROW 14 // n2 = ROW - i
#define COL 13 // n1 = j + 1
#define WINDOW_SIZE 3

void PrintMatrix(FILE* file, int* matrix[], const char* name)
{
	fprintf(file, "======= Matrix %s =======\n", name);
	for(int i = 0; i < ROW; i++)
	{
		for(int j = 0; j < COL; j++)
		{
			fprintf(file, "%d\t", matrix[i * COL + j]);
		}

		fprintf(file, "\n");
	}

	fprintf(file, "\n\n");
}

void main(void)
{
// [n1][n2] [n1][n2]
// in signal without noise: [5][4] = 162, [7][8] = 242
const int signal_noise[14][13] =
		{{ 0, 9, 18, 15, 27, 24, 16, 19, 11, 5, 10, 0, 0 },
		 { 12, 21, 26, 45, 50, 53, 79, 51, 39, 30, 8, 12, 8 },
		 { 19, 36, 57, 78, 100, 105, 124, 131, 82, 80, 57, 23, 14 },
		 { 28, 53, 91, 113, 107, 160, 160, 125, 137, 80, 53, 58, 16 },
		 { 23, 57, 102, 151, 133, 171, 220, 188, 162, 149, 85, 63, 44 },
		 { 29, 82, 109, 144, 400, 217, 261, 270, 195, 129, 146, 96, 30 },
		 { 13, 77, 119, 161, 175, 221, 323, 272, 290, 230, 152, 174, 107 },
		 { 6, 80, 116, 143, 194, 249, 357, 330, 450, 233, 177, 188, 97 },
		 { 6, 54, 99, 125, 147, 209, 293, 311, 264, 158, 143, 183, 113 },
		 { 0, 37, 75, 88, 133, 139, 268, 243, 142, 171, 101, 124, 81 },
		 { 0, 17, 47, 75, 95, 111, 137, 226, 153, 95, 108, 141, 121 },
		 { 0, 9, 37, 34, 60, 92, 126, 139, 151, 131, 95, 106, 77 },
		 { 0, 0, 15, 20, 52, 41, 78, 113, 80, 104, 75, 52, 42 },
		 { 0, 0, 12, 12, 15, 46, 31, 48, 40, 55, 35, 0, 0 }};

	int filter[3][3] = {{2, 3, 2},
	                    {2, 4, 2},
	                    {2, 3, 2}};

	const int amount = 22;
	const double m = 1.25;
	double nu = 0.0;
	int signal[14][13];
	int delta[14][13];

	FILE* file = fopen("../results/rash3.xls", "w");
	// anisotropic filter

	for(int i = 0; i < ROW; i++)
	{
		for(int j = 0; j < COL; j++)
		{
			signal[i][j] = 0;
			delta[i][j] = 0;
		}
	}

	for(int i = 0; i < ROW; i++)
	{
```

```
		for(int j = 0; j < COL; j++)
		{
			if(i == 0 || i == ROW - 1 || j == 0 || j == COL - 1)
			{
				signal[i][j] = signal_noise[i][j];
			}
			else
			{
				int sg = 0;

				for(int k1 = 0; k1 < WINDOW_SIZE; k1++)
				{
					for(int k2 = 0; k2 < WINDOW_SIZE; k2++)
					{
						sg += filter[k1][k2] * signal_noise[i - 1 + k1][j - 1 +
k2];
					}
				}

				signal[i][j] = sg / amount;
			}
		}
	}

	for(int i = 0; i < ROW; i++)
	{
		for(int j = 0; j < COL; j++)
		{
			delta[i][j] = signal_noise[i][j] - signal[i][j];
		}
	}

	fprintf(file, "Anisotropic filter\n\n");
	PrintMatrix(file, (int**)signal_noise, "Signal-noise");
	PrintMatrix(file, (int**)signal, "Signal");
	PrintMatrix(file, (int**)delta, "Delta (Signal-noise - Signal)");
	fprintf(file, "\n\n");
	// statistic filter (nu = m * sigma)

	for(int i = 0; i < ROW; i++)
	{
		for(int j = 0; j < COL; j++)
		{
			signal[i][j] = 0;
			delta[i][j] = 0;
		}
	}

	for(int i = 0; i < ROW; i++)
	{
		for(int j = 0; j < COL; j++)
		{
			int sum1 = 0;
			double sum2 = 0.0;

			for(int k1 = 0; k1 < WINDOW_SIZE; k1++)
			{
				for(int k2 = 0; k2 < WINDOW_SIZE; k2++)
				{
					sum1 += signal_noise[i - 1 + k1][j - 1 + k2];
				}
			}

			double G = (double)sum1 / pow((double)WINDOW_SIZE, 2);

			for(int k1 = 0; k1 < WINDOW_SIZE; k1++)
			{
				for(int k2 = 0; k2 < WINDOW_SIZE; k2++)
				{
					sum2 += pow((double)signal_noise[i - 1 + k1][j - 1 + k2] - G, 2);
				}
			}

			double D = sum2 / (pow((double)WINDOW_SIZE, 2) - 1);
			nu = m * sqrt(D);
			signal[i][j] = (signal_noise[i][j] - G) < nu ? signal_noise[i][j] : (int)G;
		}
	}
```

```
        for(int i = 0; i < ROW; i++)
        {
                for(int j = 0; j < COL; j++)
                {
                        delta[i][j] = signal_noise[i][j] - signal[i][j];
                }
        }

        fprintf(file, "Statistic filter\n\n");
        PrintMatrix(file, (int**)signal_noise, "Signal-noise");
        PrintMatrix(file, (int**)signal, "Signal");
        PrintMatrix(file, (int**)delta, "Delta (Signal-noise - Signal)");
        fprintf(file, "\n\n");

        fclose(file);
}
```

2) Суммирование взвешенных и сдвинутых импульсных откликов

2) Построение 3D графиков реакции системы.

```
function [] = filter_signals()
%% FILTER_SIGNALS
% Summary of this function goes here.
%
% * Syntax
%
% [OUTPUTARGS] = FILTER_SIGNALS(INPUTARGS)
%
% * Input
%
% -- INPUTARGS -
%
% * Output
%
% -- OUTPUTARGS -
%
% * Examples:
%
% Provide sample usage code here
%
% * See also:
%
% List related files here
%
% * Author: Dmitrii Leliuhin
% * Email: dleliuhin@mail.ru
% * Date: 04/04/2019 22:55:53
% * Version: 1.0 $
% * Requirements: PCWIN64, MatLab R2016a
%
% * Warning:
%
% # Warnings list.
%
% * TODO:
%
% # TODO list.
%

%% Code
clc;
clear all;
close all;

y.rows = 14;
y.cols = 13;

file_name_1 = '../results/rash3.xls';
file_name_2 = '../results/method_1.xls';

mat_origin = zeros(y.rows, y.cols);
mat_noise = zeros(y.rows, y.cols);

mat_anis_filt = zeros(y.rows, y.cols);
mat_diff_anis = zeros(y.rows, y.cols);

mat_stat_filt = zeros(y.rows, y.cols);
mat_diff_stat = zeros(y.rows, y.cols);
```

```
xls_range_origin = 'A23:M36';
xls_range_noise = 'A4:M17';

xls_range_1 = 'A21:M34';

xls_range_3 = 'A76:M89';

mat_origin = xlsread(file_name_2, xls_range_origin);
mat_noise = xlsread(file_name_1, xls_range_noise);

mat_anis_filt = xlsread(file_name_1, xls_range_1);

mat_stat_filt = xlsread(file_name_1, xls_range_3);

diff_anis = mat_origin - mat_anis_filt;

diff_stat = mat_origin - mat_stat_filt;

%=========================================================================

figure;
title('Noised signal 2D.', 'FontSize', 18);
surf(mat_noise);
view(2);
snapnow;
saveas(gcf, '../results/2D-view-noised', 'jpg');

figure;
title('Noised signal.', 'FontSize', 18);
surf(mat_noise)
saveas(gcf, '../results/noised', 'jpg');

%=========================================================================

%=========================================================================

figure;
title('Anisotropic filtered signal 2D.', 'FontSize', 18);
surf(mat_anis_filt);
view(2)
snapnow;
saveas(gcf, '../results/2D-view-anisotr-filter', 'jpg');

figure;
title('Anisotropic filtered signal.', 'FontSize', 18);
surf(mat_anis_filt)
saveas(gcf, '../results/anisotr-filter', 'jpg');

%=========================================================================

figure;
title('Anisotropic difference 2D.', 'FontSize', 18);
surf(diff_anis);
view(2)
snapnow;
saveas(gcf, '../results/2D-view-anisotr-diff', 'jpg');

figure;
title('Anisotropic difference.', 'FontSize', 18);
surf(diff_anis)
saveas(gcf, '../results/anisotr-diff', 'jpg');

%=========================================================================

%=========================================================================

figure;
title('Statistic filter 2D.', 'FontSize', 18);
surf(mat_stat_filt);
view(2)
snapnow;
saveas(gcf, '../results/2D-view-stat-filter', 'jpg');

figure;
title('Statistic filter.', 'FontSize', 18);
surf(mat_stat_filt)
saveas(gcf, '../results/stat-filter', 'jpg');

%=========================================================================

figure;
```

```
title('Statistic difference 2D.', 'FontSize', 18);
surf(diff_stat);
view(2)
snapnow;
saveas(gcf, '../results/2D-view-stat-diff', 'jpg');

figure;
title('Statistic difference.', 'FontSize', 18);
surf(diff_stat)
saveas(gcf, '../results/stat-diff', 'jpg');

%======================================================================

save('../results/workspace.mat');

close all;
end
```

Результаты

1) обработка на основе анизотропного фильтра

Матрица до применения фильтра:

0	9	18	15	27	24	16	19	11	5	10	0	0
12	21	26	45	50	53	79	51	39	30	8	12	8
19	36	57	78	100	105	124	131	82	80	57	23	14
28	53	91	113	107	160	160	125	137	80	53	58	16
23	57	102	151	133	171	220	188	162	149	85	63	44
29	82	109	144	**400**	217	261	270	195	129	146	96	30
13	77	119	161	175	221	323	272	290	230	152	174	107
6	80	116	143	194	249	357	330	**450**	233	177	188	97
6	54	99	125	147	209	293	311	264	158	143	183	113
0	37	75	88	133	139	268	243	142	171	101	124	81
0	17	47	75	95	111	137	226	153	95	108	141	121
0	9	37	34	60	92	126	139	151	131	95	106	77
0	0	15	20	52	41	78	113	80	104	75	52	42
0	0	12	12	15	46	31	48	40	55	35	0	0

Матрица после применения фильтра:

0	9	18	15	27	24	16	19	11	5	10	0	0
12	21	33	46	55	63	68	61	48	35	24	14	8
19	37	57	74	89	104	111	104	84	63	44	27	14
28	51	82	105	121	143	156	146	126	98	70	46	16
23	63	100	148	180	198	200	191	160	126	94	66	44
29	69	111	163	211	230	243	243	209	168	135	101	30
13	71	114	169	216	259	284	301	273	219	167	134	107
6	65	109	142	179	239	293	319	296	229	178	154	97
6	53	91	123	157	217	273	295	260	199	160	140	113
0	37	69	97	125	166	220	231	192	148	131	127	81
0	23	47	70	92	126	165	182	159	126	116	110	121
0	12	29	46	64	87	117	137	132	111	99	92	77
0	7	16	27	41	59	79	92	94	87	72	53	42
0	0	12	12	15	46	31	48	40	55	35	0	0

Разница матриц до и после применения фильтра:

0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	-7	-1	-5	-10	11	-10	-9	-5	-16	-2	0
0	-1	0	4	11	1	13	27	-2	17	13	-4	0
0	2	9	8	-14	17	4	-21	11	-18	-17	12	0
0	-6	2	3	-47	-27	20	-3	2	23	-9	-3	0
0	13	-2	-19	-49	-13	18	27	-14	-39	11	-5	0
0	6	5	-8	-41	-38	39	-29	17	11	-15	40	0
0	15	7	1	15	10	64	11	-54	4	-1	34	0
0	1	8	2	-10	-8	20	16	4	-41	-17	43	0
0	0	6	-9	8	-27	48	12	-50	23	-30	-3	0
0	-6	0	5	3	-15	-28	44	-6	-31	-8	31	0
0	-3	8	-12	-4	5	9	2	19	20	-4	14	0
0	-7	-1	-7	11	-18	-1	21	-14	17	3	-1	0
0	0	0	0	0	0	0	0	0	0	0	0	0

2) обработка на основе анизотропного фильтра

Матрица до применения фильтра:

0	9	18	15	27	24	16	19	11	5	10	0	0
12	21	26	45	50	53	79	51	39	30	8	12	8
19	36	57	78	100	105	124	131	82	80	57	23	14
28	53	91	113	107	160	160	125	137	80	53	58	16
23	57	102	151	133	171	220	188	162	149	85	63	44
29	82	109	144	**400**	217	261	270	195	129	146	96	30
13	77	119	161	175	221	323	272	290	230	152	174	107
6	80	116	143	194	249	357	330	**450**	233	177	188	97
6	54	99	125	147	209	293	311	264	158	143	183	113
0	37	75	88	133	139	268	243	142	171	101	124	81
0	17	47	75	95	111	137	226	153	95	108	141	121
0	9	37	34	60	92	126	139	151	131	95	106	77
0	0	15	20	52	41	78	113	80	104	75	52	42
0	0	12	12	15	46	31	48	40	55	35	0	0

Матрица после применения фильтра:

0	9	18	15	27	24	16	19	11	5	10	0	0
12	21	26	45	50	53	79	51	39	30	8	12	8
19	36	57	78	100	105	124	131	82	80	57	23	14
28	53	91	113	107	160	160	125	137	80	53	58	16
23	57	102	151	133	171	220	188	162	149	85	63	44
29	82	109	144	197	217	261	270	195	129	146	96	30
13	77	119	161	175	221	323	272	290	230	152	174	107
6	80	116	143	194	249	285	330	282	233	177	188	97
6	54	99	125	147	209	293	311	264	158	143	183	113
0	37	75	88	133	139	268	243	142	171	101	124	81
0	17	47	75	95	111	137	226	153	95	108	106	121
0	9	37	34	60	92	126	139	151	131	95	106	77
0	0	15	20	52	41	78	113	80	104	75	52	42
0	0	12	12	15	46	31	48	40	55	35	0	0

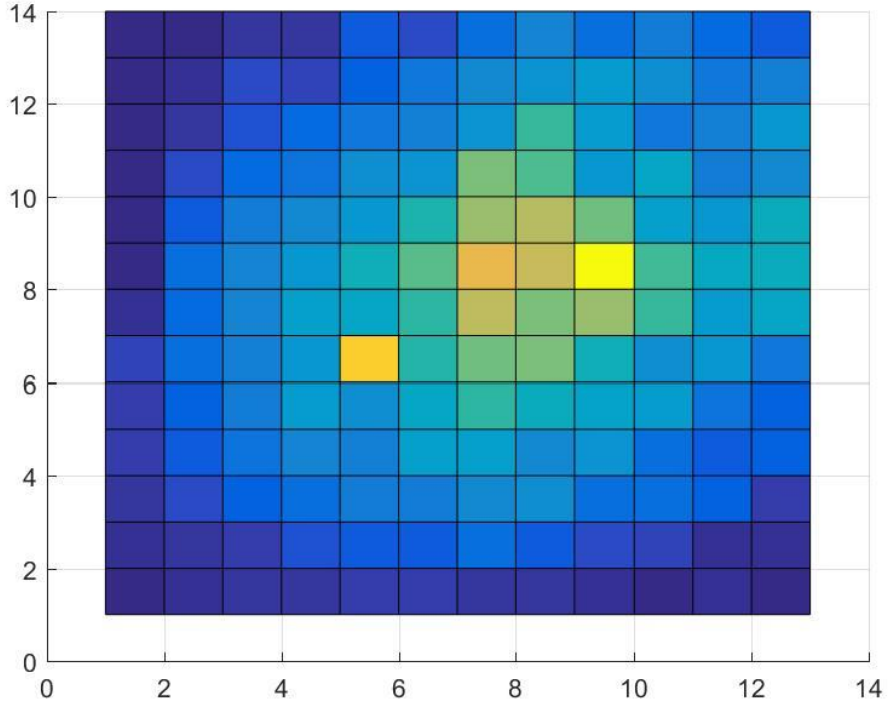
Разница матриц до и после применения фильтра:

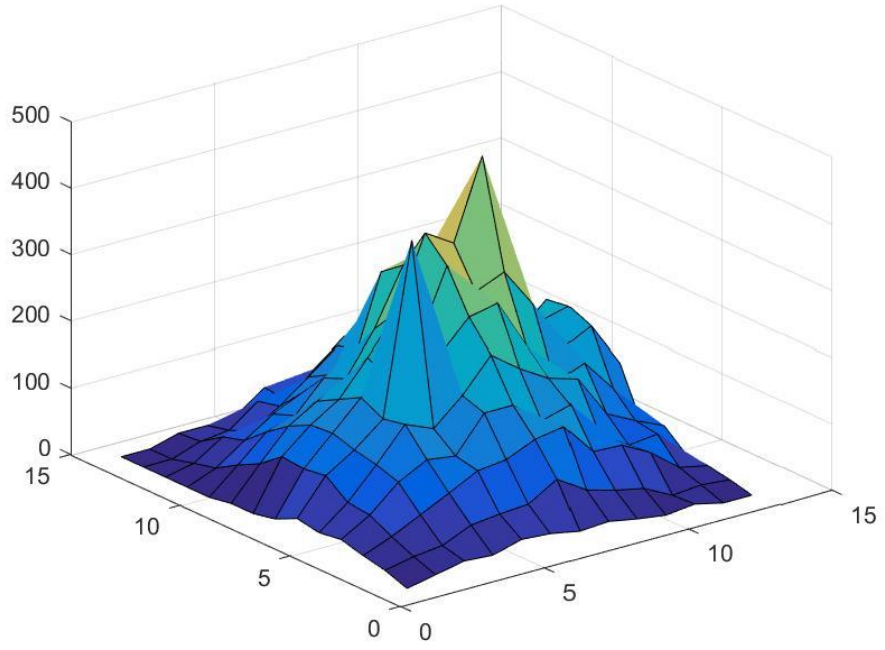
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	-35	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	72	0	-40	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	35	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0

Графическое представление выходного сигнала

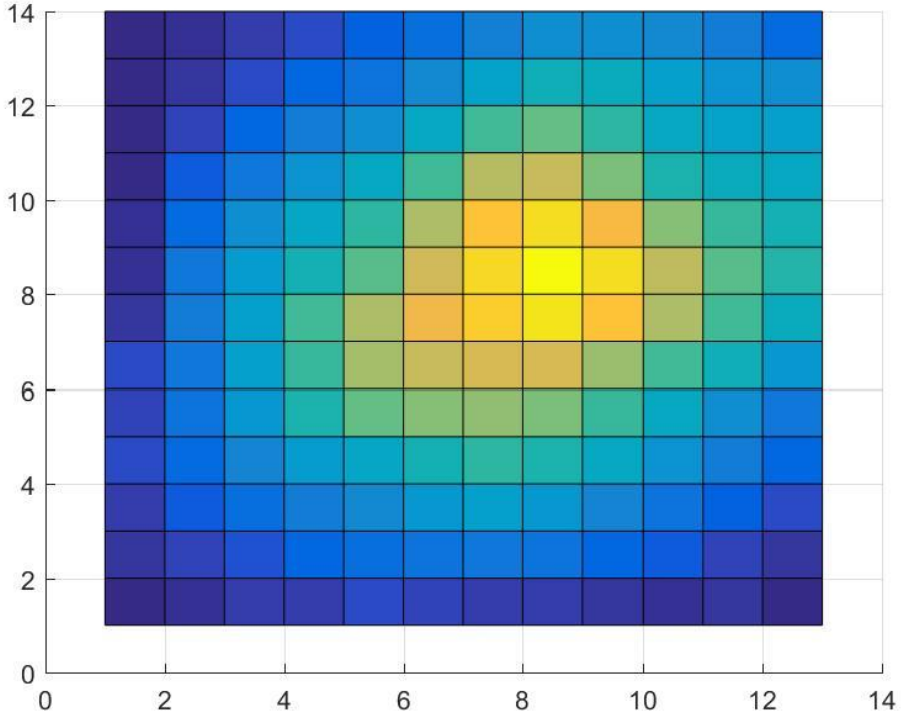
Графики построены в среде MatLab R2016a.

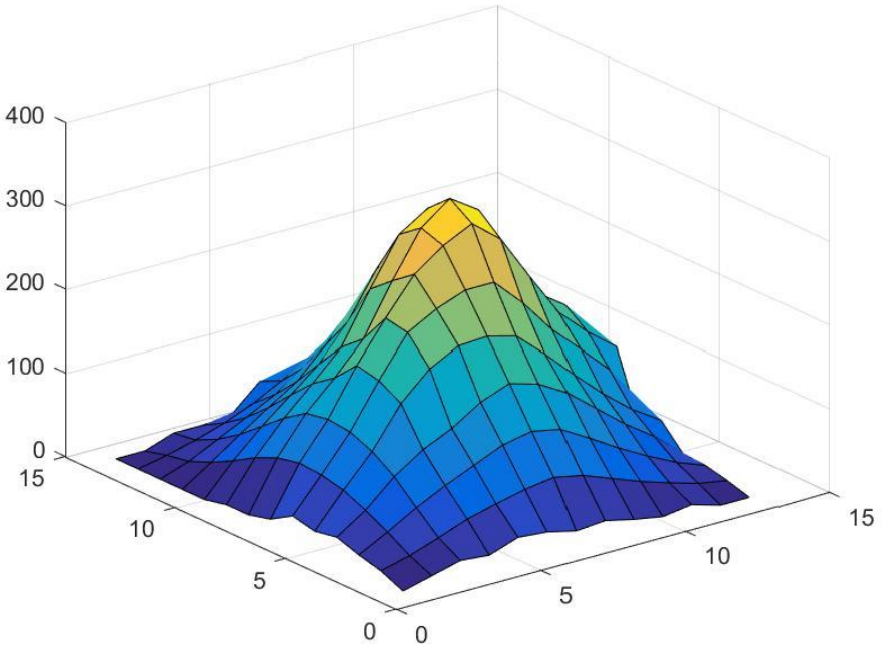
Вид сигнала до применения фильтров (сигнал с шумом):



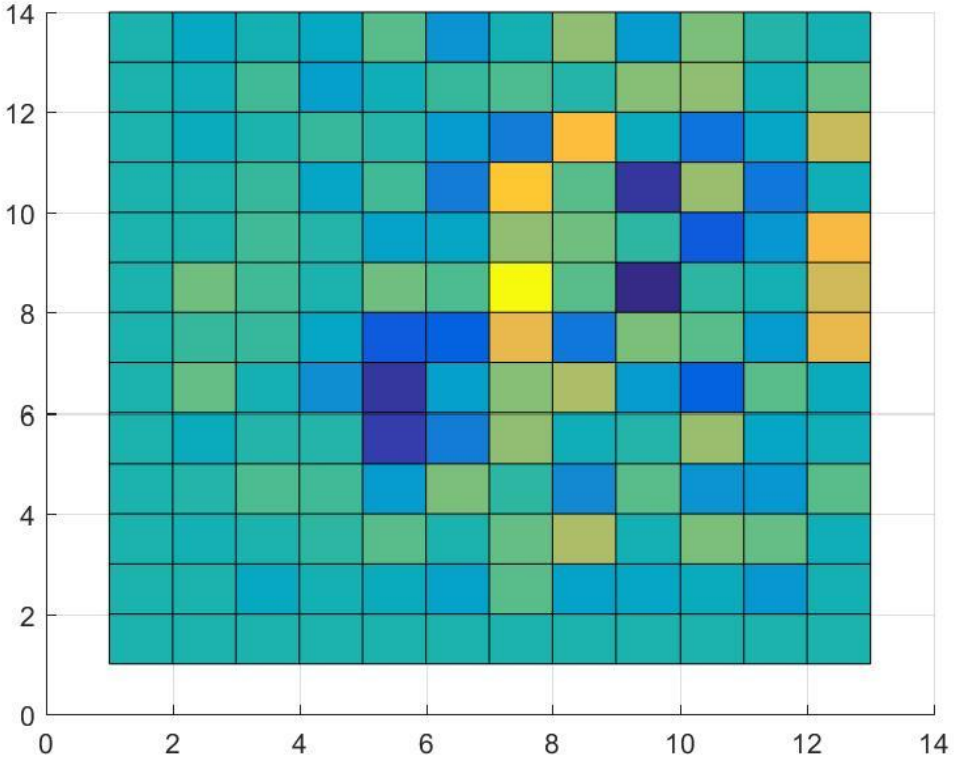


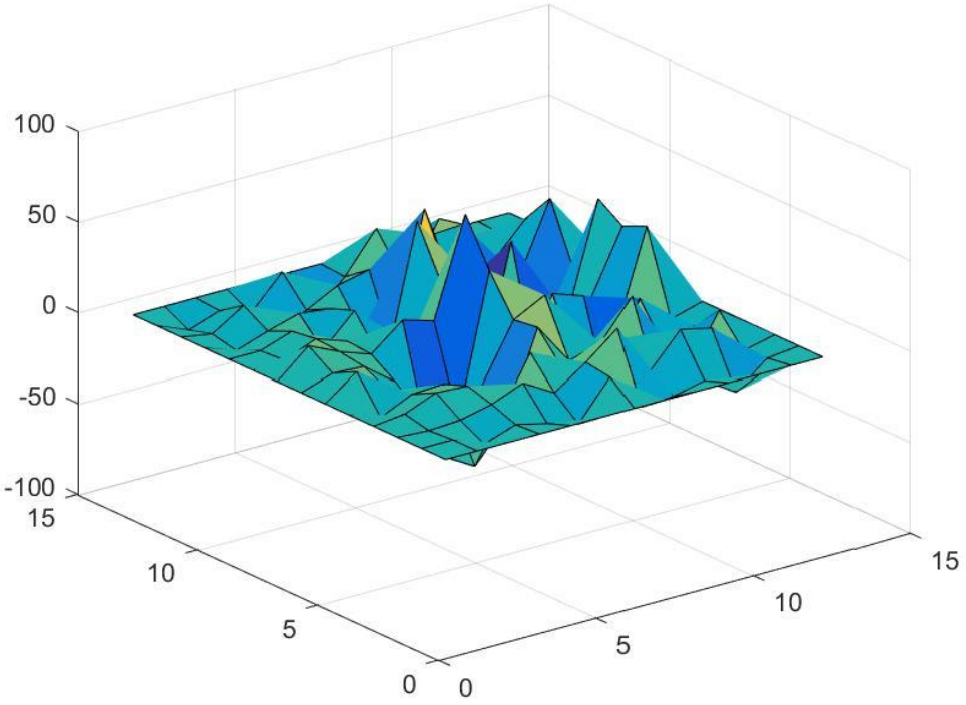
Вид сигнала после применения анизотропного фильтра:



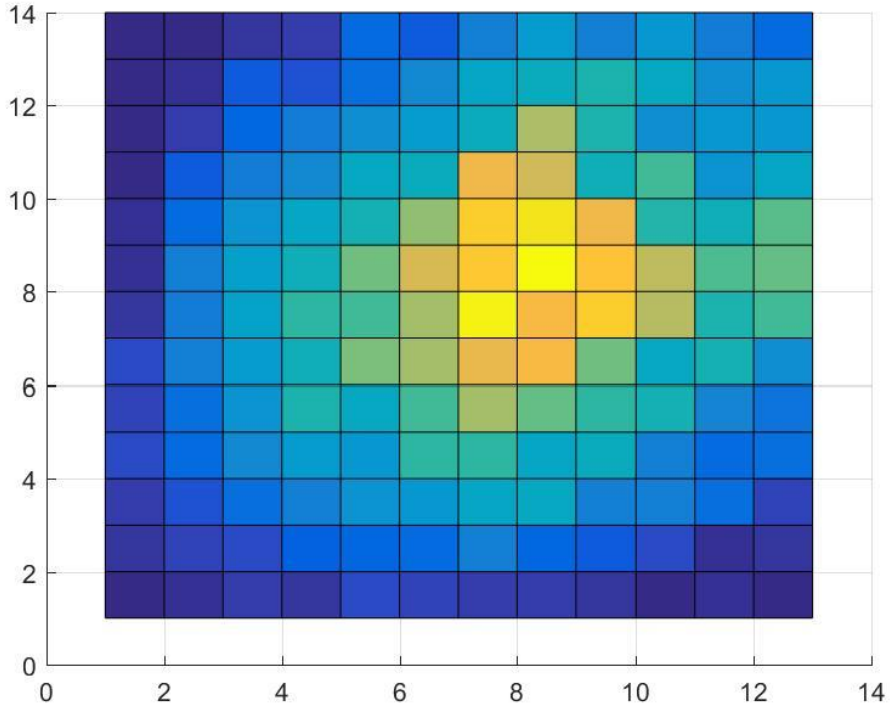


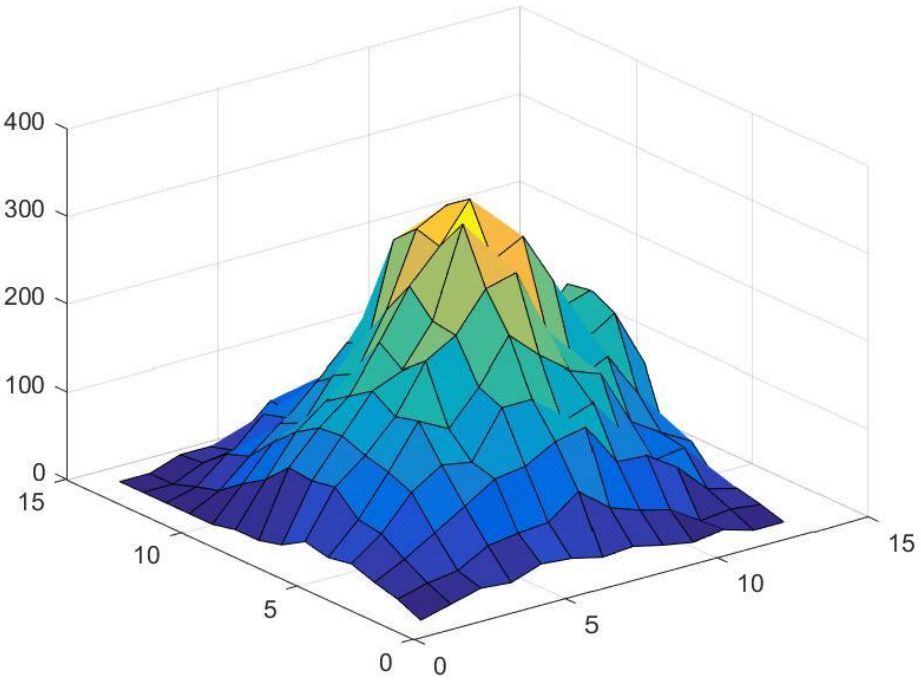
Разница между сигналом без шума и сигналом после применения анизотропного фильтра:



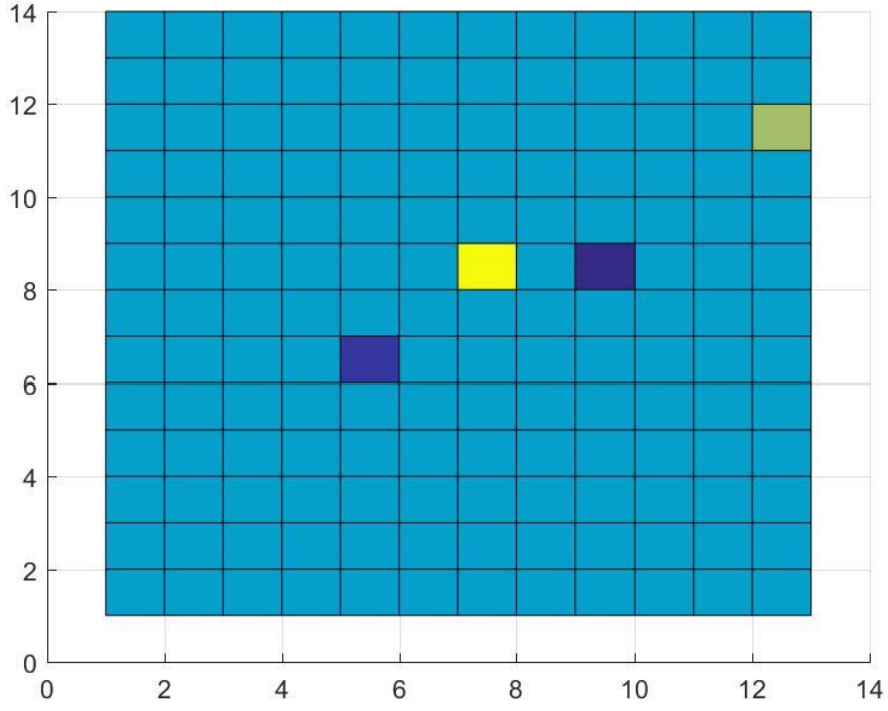


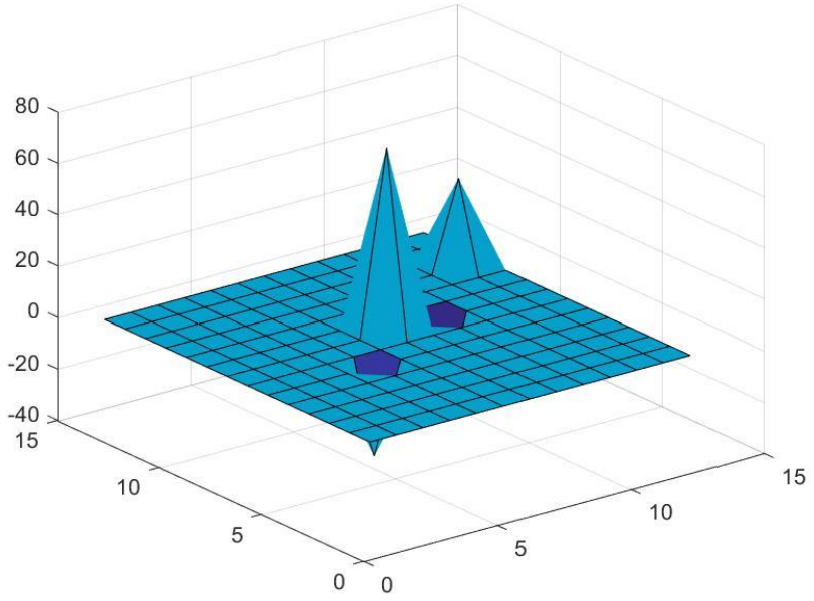
Вид сигнала после применения статистического порогового фильтра:





Разница между сигналом без шума и сигналом после применения статистического фильтра:





Выводы

Как видно из графиков и таблиц применение статистической пороговой фильтрации дало лучшие результаты, чем анизотропный фильтр для данного сигнала и характера внесенных помех.