Санкт-Петербургский политехнический
университет им. Петра Великого
Институт прикладной математики и механики
Кафедра Прикладная математика

Лабораторная работа №4

Эмиссионная томография плазмы. Построение СЛАУ

по дисциплине: Вычислительные комплексы

Студент группы 3630102/60201 Митрофанова А.Г.

Преподаватель Баженов А.Н.

Санкт-Петербург
2020 год

Содержание

Список иллюстраций

1 Постановка задачи

Необходимо считать данные о сепаратрисе из g-файла. Построить кривую сепаратрисы на графике, вычислить и отметить магнитную ось. Построить геометрическую матрицу хорд. Поставить задачу о нахождении светимостей различных областей (переопределённая СЛАУ).

2 Теория

На сечении Токамак «Глобус-М» можно увидеть следующие компоненты: стенки вакуумной камеры, сепаратриса (граница плазмы, последняя замкнутая область магнитного потока) и магнитная ось (условный центр тока плазмы). [1]
Данные о геометрическом месте сепаратрисы содержатся в так называемых g-файлах в виде массивов **RBDRY**, **ZBDRY**. Функция MATLAB

$$[flux, RBDRY, ZBDRY, NBDRY, R, Z, time, rdim, zdim] =$$

$$= gfile\_extractor\_1t(shot\_number\_test, start\_efit\_time, MagMesh)$$

[2] возвращает

- **RBDRY**, **ZBDRY** – (R, Z)-координаты точек сепаратрисы
- **NDRY** – число точек сепаратрисы
- **rdim**, **zdim** – размер сетки по радиусу и вертикали
- **R**, **Z** – координаты точек сетки по радиусу и вертикали
- **flux** – двумерный массив магнитного потока

В окрестности минимума распределения **flux** находится «магнитная ось». Область «магнитной оси» можно считать «центром» плазмы.

2.1 Построение разбиения области

- Делим область на 2 сектора по экватору
- В каждом секторе выбираем точку с максимальным радиусом кривизны для нахождения опорного направления
- Находим в каждом секторе точку с минимальным радиусом кривизны
- Найдены 4 особые точки. Соединив их с центром получим 4 сектора
- Внутри каждого сектора проводим еще N дополнительных отрезков. Получено $4N$ отрезка
- Находим у каждого из отрезков середину и соединяем их между собой

2.2 Информация о детекторе

Данные о детекторе взяты из [3].

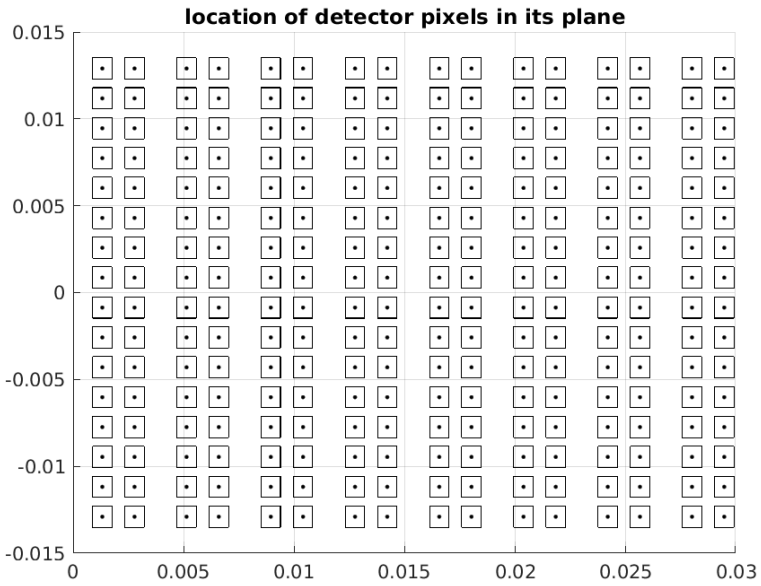

Рис. 1: Расположение пикселей детектора в его плоскости

Угол между направлением камеры-обскуры и направлением на центр (между 8 и 9 лучами)

$$ang = \arccos \frac{708^2 + 720^2 - 31^2}{2 \cdot 708 \cdot 720} \tag{1}$$

Положение края детектора (1-го столбца) в координатах XY

$$spd\_start = [0, -0.708] \tag{2}$$

Положение 16-го столбца в координатах XY

$$spd\_end = [0.72 \cdot \sin ang, 0.72 \cdot (-\cos ang)] \tag{3}$$

Вектор направления камеры-обскуры в экваториальной плоскости

$$spd\_vect = \frac{spd\_end - spd\_start}{||spd\_end - spd\_start||} \tag{4}$$

Шаг между столбцами в плоскости детектора

$$spd\_xy\_step = [2.3375 - 0.88, 3.81 - 2.3375 + 0.88] \cdot 10^{-3} \tag{5}$$

Центр детектора

$$pp = spd\_start + spd\_vect \cdot ((spd\_xy\_step(1) + spd\_xy\_step(2)) \cdot 8 + 0.52 \cdot 10^{-3}) \cdot \frac{1}{2} \tag{6}$$

Отступ апертуры от центра детектора

$$aperture\_xy\_offset = 0.0395 \tag{7}$$

Координаты апертуры

$$aperture\_xy = [pp(1) - spd\_vect(2) \cdot aperture\_xy\_offset, pp(2) + spd\_vect(1) \cdot aperture\_xy\_offset] \tag{8}$$

Устройство детектора в меридиональной плоскости

$$spd\_z\_start = \frac{27.52 - 0.49}{2} \cdot 10^{-3} \tag{9}$$

$$spd\_z\_step = -1.72 \cdot 10^{-3} \tag{10}$$

$$spd\_xy = spd\_start + spd\_vect \cdot \left(\frac{spd\_xy\_step(2)}{2} + 0.26 \cdot 10^{-3} \right) \tag{11}$$

2.3 Нахождение сечения плазмы плоскостью

Плазма представляется как фигура вращения, где роль образующей выполняет сетка разбиения сепаратрисы. Для каждого вертикального ряда пикселей детектора вычисляется коэффициенты (k и b) прямой, проходящей через этот пиксель апертуру детектора. После чего вычисляет расстояние H от центра токамака до прямой. Далее каждый элемент сетки представляется как фигура вращения, ось вращения которой совпадает с осью токамака, а образующая – текущий элемент сетки. Для этой фигуры рассчитывается сечение плоскостью $x = H$. В этом сечении для каждого пикселя в вертикальном ряду вычисляется прямая, проходящая через центр пикселя и апертуру детектора.

3 Реализация

Реализация осуществлена на языке программирования MATLAB в среде разработке MATLAB Online R2019b.
Для считывания информации о сепаратрисе из g-файла была использована функция $gfile\_extractor\_1t$ [2].
Необходимая информация для выполнения лабораторной была взята из [3].

4 Результаты

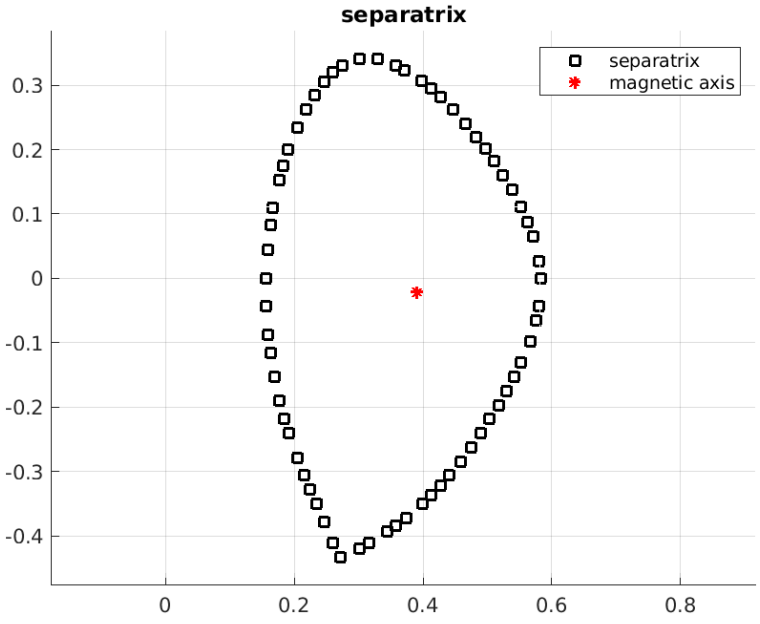

Рис. 2: График сепаратрисы и магнитной оси

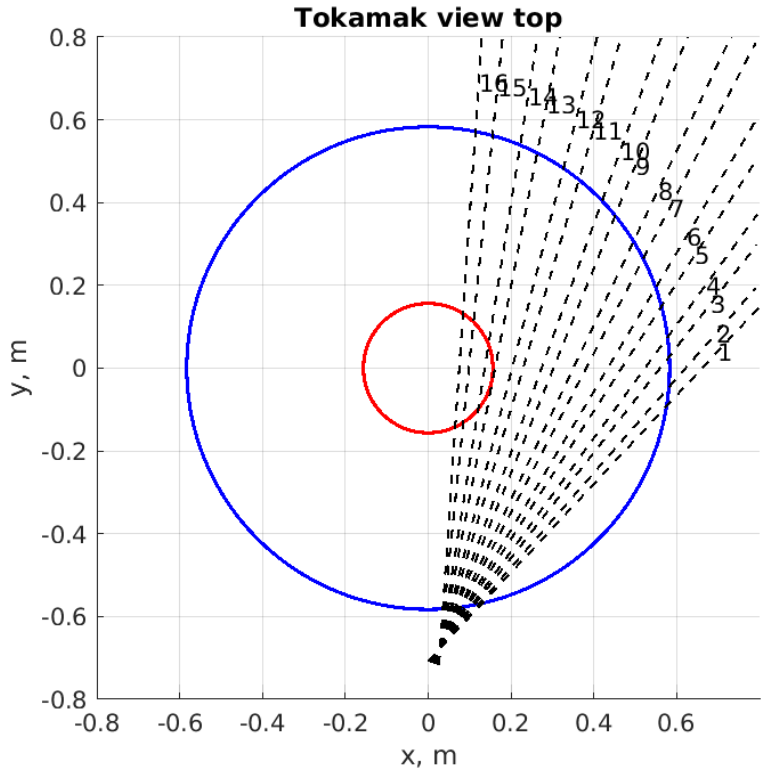

Рис. 3: График положения сечений

Далее приведены сечения для всех 16-ти столбцов, где

- Зелёным цветом – сетка
- Черные линии – лучи
- Красные круги – пересечения

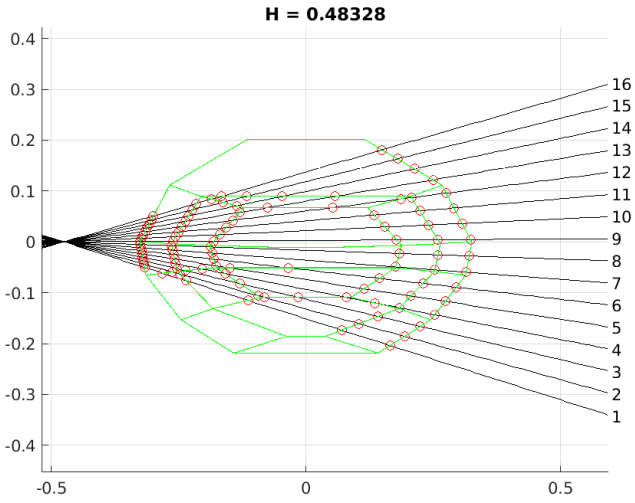

Рис. 4: Сечение 1

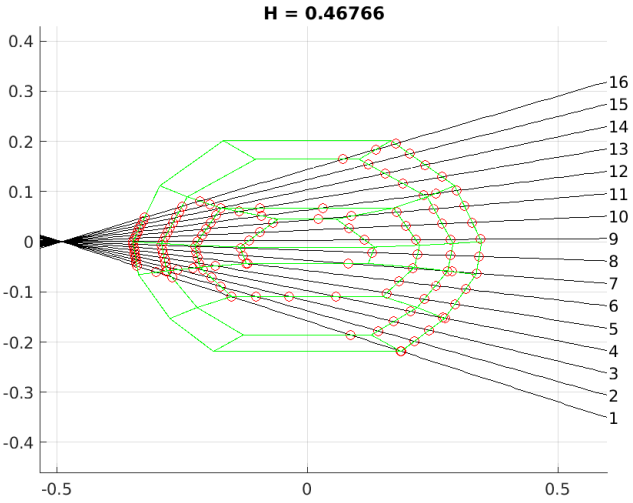

Рис. 5: Сечение 2

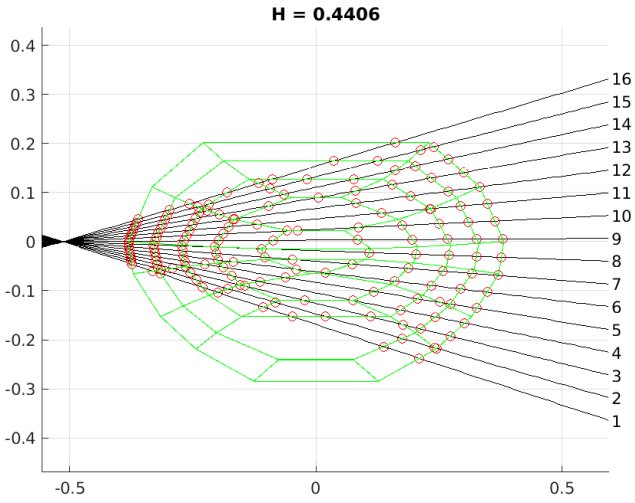

Рис. 6: Сечение 3

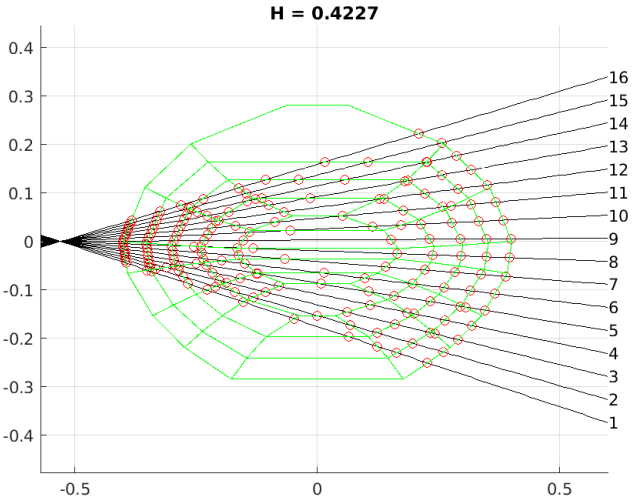

Рис. 7: Сечение 4

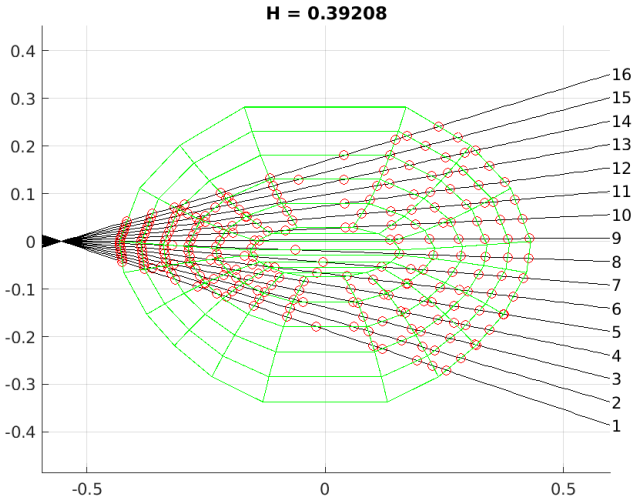

Рис. 8: Сечение 5

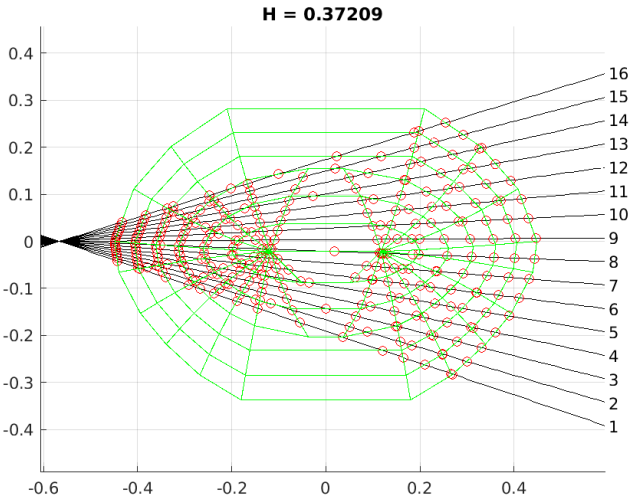

Рис. 9: Сечение 6

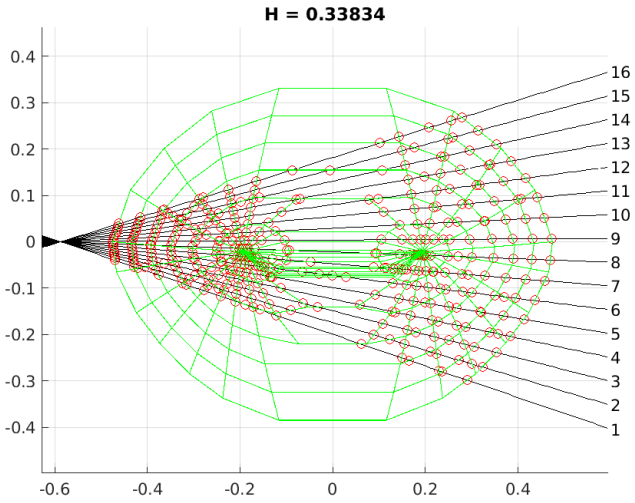

Рис. 10: Сечение 7

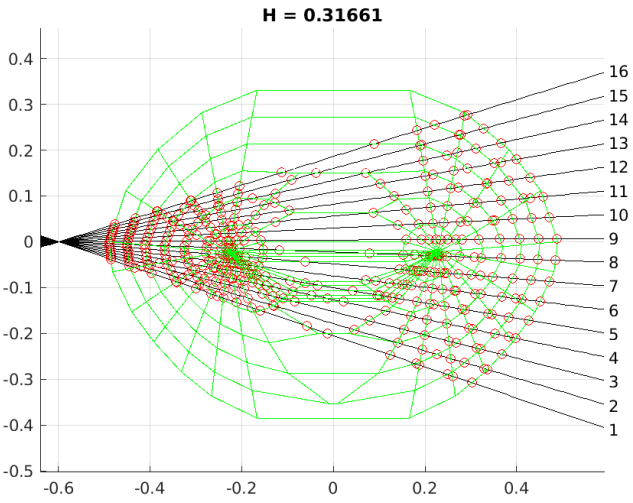

Рис. 11: Сечение 8

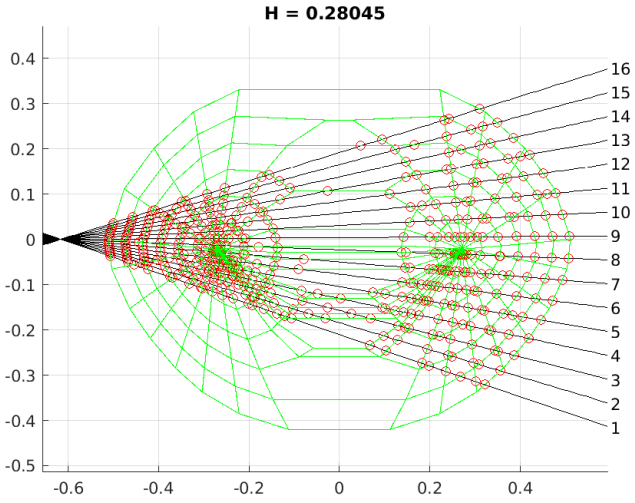

Рис. 12: Сечение 9

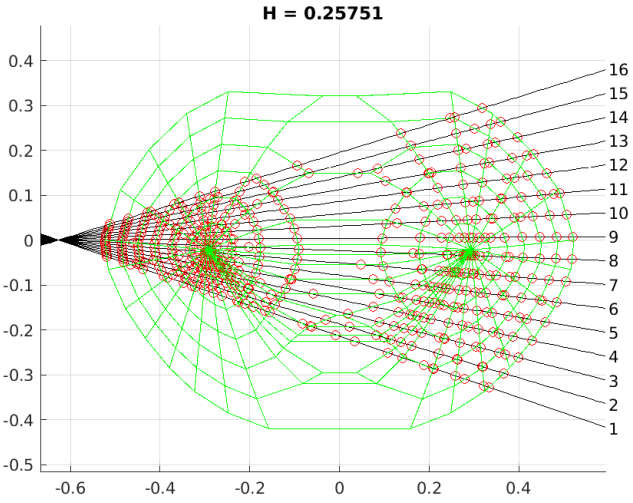

Рис. 13: Сечение 10

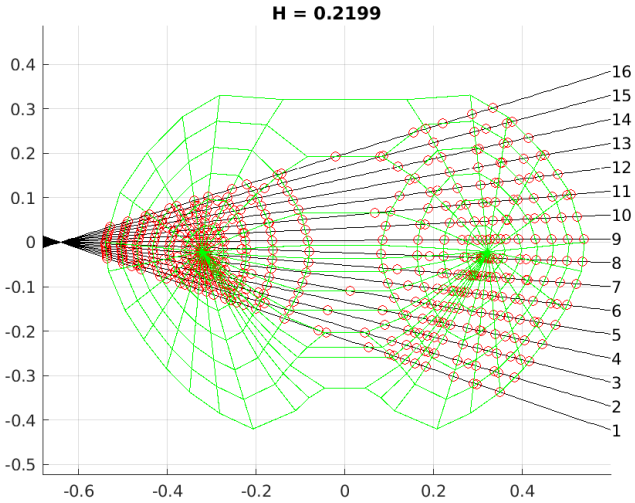

Рис. 14: Сечение 11

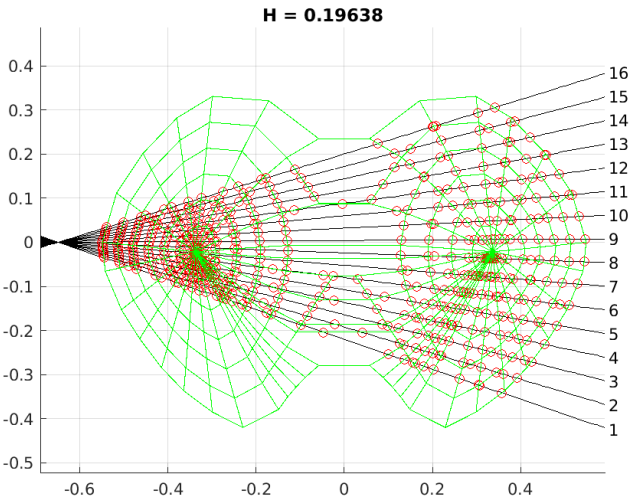

Рис. 15: Сечение 12

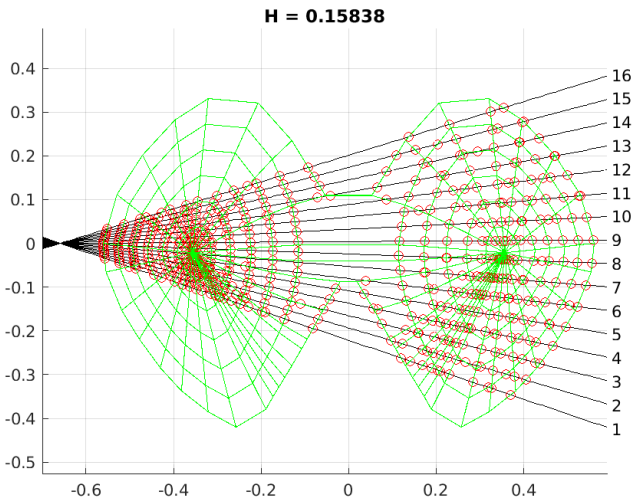

Рис. 16: Сечение 13

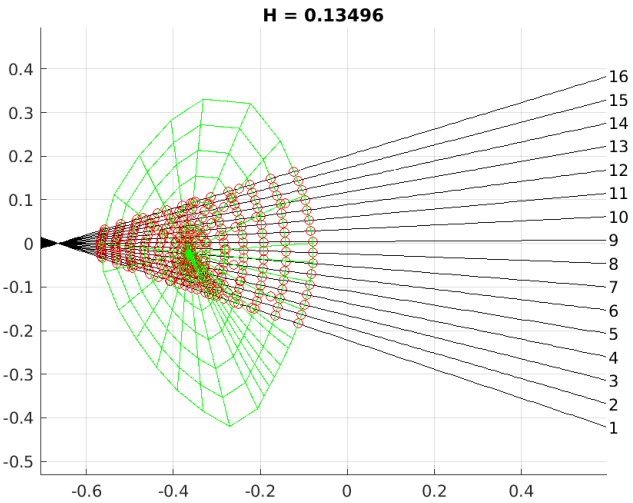

Рис. 17: Сечение 14

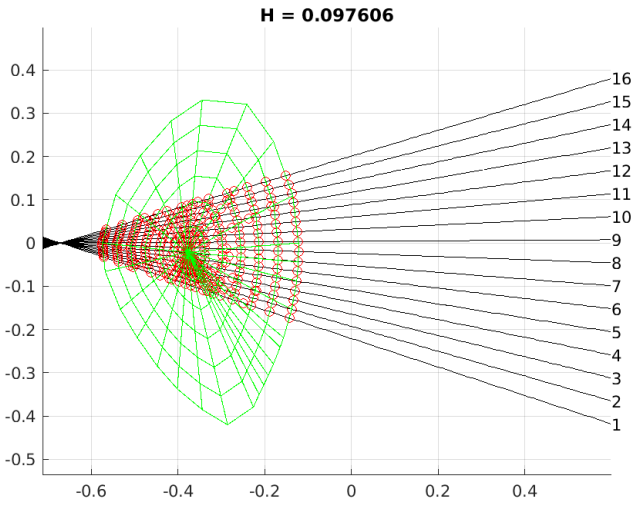

Рис. 18: Сечение 15

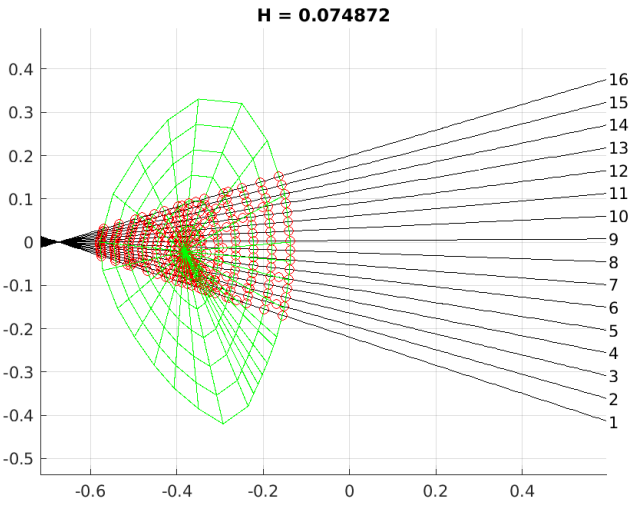

Рис. 19: Сечение 16

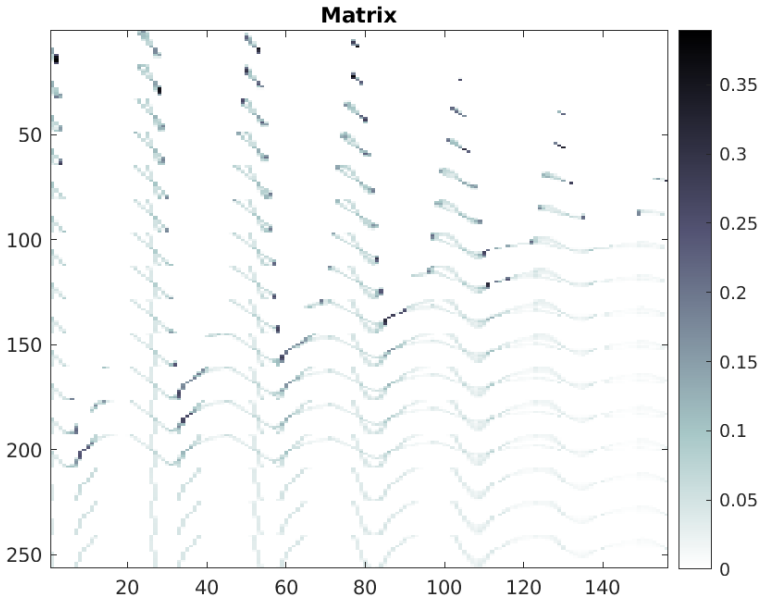

Рис. 20: График матрицы длин хорд

5 Обсуждение

На сечениях 14, 15, 16 плоскость сечения H меньше самой левой точки сепаратрисы, и область получается двусвязной. В случае двусвязной области считаем, что луч упирается в центральную ось токамака, и учитываем только левую область.

СЛАУ представляет собой матрицу $256xN$, где N – количество элементов разбиения. Каждая строка матрицы отвечает за свой луч, притом коэффициенты для каждого элемента разбиения – сумма длин хорд. Построенная матрица является сильно разреженной и плохо обусловленной.

6 Литература

Список литературы

[1] А.Н. Баженов, Лабораторный практикум. Методический материал. «Вычислительные комплексы» [Электронный ресурс, облачное хранилище]. Режим доступа: https://cloud.mail.ru/public/4ra6/5wwqBzMBC/LabPractics.pdf (дата обращения: февраль, 2019 г.)

[2] Код функции, извлекающей информацию о сепаратрисе [Электронный ресурс, облачное хранилище]. Режим доступа: https://cloud.mail.ru/public/5o3T/4G4dD71hL (дата обращения: февраль, 2019 г.)

[3] Необходимые сведения для выполнения лабораторной [Электронный ресурс]. Режим доступа: https://vk.com/doc211504187_521293842?hash=d0debf7b0ccd53ab93dl=eb351b97a099a08cd1 (дата обращения: февраль, 2019 г.)

7 Приложение

7.1 main.m

```
%%
clear all
point_ = get_point_interface();
element_ = get_element_interface();
detector_ = get_detector_interface();
%%
[flux, RBDRY, ZBDRY, NBDRY, R, Z, time, rdim, zdim] = gfile_extractor_1t(34363, 000162, 65);
%%
% change the start of the crawl and change the detour of the separatrix to counter-hourly
% since the last point is equal to the first, let's take this into account
%a new beginning - the 32nd element
[RBDRY,ZBDRY] = circle_spin_and_reverse(RBDRY,ZBDRY,NBDRY, 32);

[arr, ind_arr] = min(flux);
[flux_min, min_j] = min (arr);
min_i = ind_arr(min_j);
magnet_axis = [R(min_j), Z(min_i)];
%% plot separatrix and magnetic axis
figure()
grid on
hold on
plot(RBDRY, ZBDRY, "ks", 'LineWidth', 1.5);
plot(magnet_axis(1), magnet_axis(2), "r*", 'LineWidth', 1.5);
[x_lim, y_lim] = get_plot_lim(RBDRY, ZBDRY);
xlim(x_lim)
ylim(y_lim)
axis equal
title("separatrix")
legend("separatrix", "magnetic axis")
%%
num_sectors = 8;
num_circle = 6;
cur_cut_y = RBDRY;
cur_cut_z = ZBDRY;
cur_magnet_axis = magnet_axis;

[R_segments_arr, Z_segments_arr, lines_start, lines_end] = get_web_grid(cur_cut_y, cur_cut_z
    , cur_magnet_axis, num_sectors, num_circle);
elements = create_element_from_grid(R_segments_arr, Z_segments_arr, lines_start, lines_end);
%% plot grid of splitting of separatrix
figure()
grid on
hold on
for i = 1 : length(elements)
    element_.draw(elements(i), "b", i);
```

```
end
axis equal
title("grid of splitting separatrix")
legend("grid")
%% calculate partition grids of the separatrix in the cross section plane x = H
% H - distance from the center of the tokamak to the section plane
min_x_sep = min(RBDRY);
max_x_sep = max(RBDRY);
H = 0;
h = length(elements);
cut_elements = [];
left_cut_elem = [];
right_cut_elem = [];
for i = 1 : h
    % the result of the intersection of the sector rotation shape and the x = H plane
    [elem, count] = element_.get_cut(elements(i), H);
    if(count == 2)
        left_cut_elem = [left_cut_elem, elem(1)];
        right_cut_elem = [right_cut_elem, elem(2)];
    else
        left_cut_elem = [left_cut_elem, elem];
    end
end

cut_elements = [left_cut_elem, right_cut_elem];
%% magic constants
% angle between the pinhole camera direction and the center direction (between 8 and 9 beams
    )
ang = acos((708^2 + 720^2 - 31^2) / (2 * 708 * 720));
% position of the detector edge (1st column)
spd_start = [0, -0.708];
% position of the 16th column
spd_end = [0.72 * sin(ang), 0.72 * -cos(ang)];
% direction vector of the pinhole camera in the Equatorial plane
spd_vect = (spd_end - spd_start) / norm(spd_end - spd_start);
% step between columns in the detector plane, 2 numbers
spd_xy_step = [2.3375 - 0.88 , 3.81 - 2.3375 + 0.88 ] * 1e-03;
% the center of the detector
pp = spd_start + spd_vect * ((spd_xy_step(1) + spd_xy_step(2)) * 8 + 0.52 * 1e-03) / 2;
% aperture offset from the center of the detector
aperture_xy_offset = 0.0395;
% coordinate of the aperture
aperture_xy = [pp(1) - spd_vect(2) * aperture_xy_offset, pp(2) + spd_vect(1) *
    aperture_xy_offset];
spd_z_start = (27.52 - 0.49) / 2 * 1e-03;
spd_z_step = -1.72 * 1e-03;
spd_xy = spd_start + spd_vect * (spd_xy_step(2) / 2 + 0.26 * 1e-03);
%% detectors parametres
detector = detector_.create();
detector.start = point_.create(spd_start(1), spd_start(2));
detector.end = point_.create(spd_end(1), spd_end(2));
detector.step = spd_xy_step;
detector.direction = point_.create(spd_vect(1), spd_vect(2));
detector.center = point_.create(pp(1), pp(2));
detector.aperture_offset = 0.0395;
detector.aperture_pos = point_.create(aperture_xy(1), aperture_xy(2));
detector.z_start = spd_z_start;
detector.z_step = spd_z_step;
detector.horizontal_step = point_.create((detector.end.x - detector.start.x) / 16 , (
    detector.end.y - detector.start.y) / 16 );
%% plot Tokamak view top
figure()
grid on
hold on
axis equal
phi = -pi : pi / 360 : pi;
R = max(RBDRY);
r = min(RBDRY);
plot(R * cos(phi), R * sin(phi), "b", 'LineWidth', 1.5);
plot(r * cos(phi), r * sin(phi), "r", 'LineWidth', 1.5);
```

```
112
113 x = detector.start.x;
114 y = detector.start.y;
115
116 for i = 1 : 16
117     A = detector_.get_col_pos(detector, i);
118     x = A.x : 0.01 : 0.8;
119     B = detector.aperture_pos;
120     [k, b] = get_line(A, B);
121     plot(x, k*x + b, "k--", 'LineWidth', 1)
122     text_R = R * 1.2;
123     text_x = (-2 * b * k + sqrt(4 * b ^ 2 * k ^ 2 - 4 * (k ^ 2 + 1) * (b ^ 2 - text_R ^ 2)))
        / (2 *(k ^ 2 + 1));
124     text_y = k * text_x + b;
125     text(text_x, text_y, num2str(i));
126 end
127
128 xlim([-0.8, 0.8])
129 ylim([-0.8, 0.8])
130
131 title('Tokamak view top')
132 xlabel('x, m')
133 ylabel('y, m')
134 %%
135 for cut_ind = 1 : 16
136     figure()
137     grid on
138     hold on
139     axis equal
140     H = detector_.get_plane(detector, cut_ind);
141     title(strcat("H = ", num2str(H)));
142     element_num = length(elements);
143     cut_elements = [];
144     for i = 1 : element_num
145         [elem, count] = element_.get_cut(elements(i), H);
146         if(H < min_x_sep)
147             if(count == 2)
148                 cut_elements = [cut_elements, elem(2)];
149             else
150                 cut_elements = [cut_elements, elem];
151             end
152         else
153             cut_elements = [cut_elements, elem];
154         end
155     end
156
157     for ray_ind = 1 : 16
158         [k, b, det_pos, apper_pos] = detector_.get_ray(detector, cut_ind, ray_ind);
159         x = det_pos.x : 0.01 : 0.6;
160         plot(x, k * x + b, "k")
161         for t = 1 : length(cut_elements)
162             element_.draw(cut_elements(t), "g")
163             [hord, intersection] = element_.get_hord(cut_elements(t), k, b);
164             for g = 1 : length(intersection)
165                 plot(intersection(g).x, intersection(g).y, "or");
166             end
167         end
168         text_x = 0.6;
169         text_y = text_x * k + b;
170         text(text_x, text_y, num2str(ray_ind));
171     end
172 end
173 %% plot location of detector pixels in its plane
174 figure()
175 grid on
176 hold on
177 %axis equal
178 spd_sizes = [0.88, 1.23] * 1e-3;
179
180 y_square_up = [0, spd_sizes(1), spd_sizes(1), 0, 0];
```

```
y_square_down = [0, 0, spd_sizes(1), spd_sizes(1), 0];
z_square_up = [0, 0, spd_sizes(2), spd_sizes(2), 0];
z_square_down = [0, - spd_sizes(2), - spd_sizes(2), 0, 0];
color = "k";
point_color = ".k";
for j = 0 : 7
    for i = 0 : 7
        x = (spd_sizes(1) + j * sum(spd_xy_step) + y_square_up);
        y = (- spd_z_step - spd_sizes(2)) / 2 + i * (- spd_z_step) + z_square_up;
        plot(x, y, color);
        x = x(1 : 4);
        x = sum(x) / 4;
        y = y(1 : 4);
        y = sum(y) / 4;
        plot(x, y, point_color);
        x = spd_sizes(1) + spd_xy_step(1) + j * sum(spd_xy_step) + y_square_up;
        y = (- spd_z_step - spd_sizes(2)) / 2 + i * (- spd_z_step) + z_square_up;
        plot(x, y, color);
        x = x(1 : 4);
        x = sum(x) / 4;
        y = y(1 : 4);
        y = sum(y) / 4;
        plot(x, y, point_color);
        x = spd_sizes(1) + j * sum(spd_xy_step) + y_square_down;
        y = (spd_z_step + spd_sizes(2)) / 2 + i * spd_z_step + z_square_down;
        plot(x, y, color);
        x = x(1 : 4);
        x = sum(x) / 4;
        y = y(1 : 4);
        y = sum(y) / 4;
        plot(x, y, point_color);
        x = spd_sizes(1) + spd_xy_step(1) + j * sum(spd_xy_step) + y_square_down;
        y = (spd_z_step + spd_sizes(2)) / 2 + i * spd_z_step + z_square_down;
        plot(x, y, color);
        x = x(1 : 4);
        x = sum(x) / 4;
        y = y(1 : 4);
        y = sum(y) / 4;
        plot(x, y, point_color);
    end
end

title('location of detector pixels in its plane');
%%
element_num = length(elements);
hord_matrix = zeros(256, element_num);
cur_row = 1;
for cut_ind = 1 : 16
    H = detector_.get_plane(detector, cut_ind);
    cut_elements = [];
    for i = 1 : element_num
        [elem, count] = element_.get_cut(elements(i), H);
        if(H < min_x_sep)
            if(count == 2)
                cut_elements = [cut_elements, elem(2)];
            else
                cut_elements = [cut_elements, elem];
            end
        else
            cut_elements = [cut_elements, elem];
        end
    end
    N = length(cut_elements);
    for ray_ind=1 : 16
        [k, b, det_pos, apper_pos] = detector_.get_ray(detector, 16, ray_ind);
        for t = 1 : N
            [hord, intersection] = element_.get_hord(cut_elements(t), k, b);
            hord_matrix(cur_row, cut_elements(t).index) = hord_matrix(cur_row, cut_elements(
t).index) + hord;
        end
```

```
        cur_row = cur_row + 1;
    end
end
figure
imagesc(hord_matrix);
title('Matrix');
colormap('Bone');
cm = colormap;
colormap(flipud(cm));
colorbar;
```