Путин Павел Александрович, группа 7-1

Лабораторная работа № 3

Вариант № 13

Распознавание образов, описываемых гауссовскими случайными векторами с разными матрицами ковариаций

Цель работы

Синтезировать алгоритмы распознавания образов, описываемых гауссовскими случайными векторами с разными матрицами ковариаций. Исследовать синтезированные алгоритмы распознавания с точки зрения ожидаемых потерь и ошибок.

Задание

Получить у преподавателя вариант задания и написать код, реализующий алгоритм распознавания образов, описываемых гауссовскими случайными векторами с заданными параметрами. Получить матрицы ошибок на основе аналитических выражений и вычислительного эксперимента. Провести анализ полученных результатов и представить его в виде выводов по проделанной работе.

m1=[1 2], m2=[1 -1], C1=[3 -1; -1 3], C2=[5 2; 2 6].

Построить график разности суммарной экспериментальной и теоретической ошибок первого рода (для первого класса) от числа испытаний (объема выборки.

Код программы (внесённые изменения в шаблон кода выделены)

```
%% Файл pr53_rec_gaus_uneq. Синтез и анализ алгоритмов распознавания ГСВ с
% различными матрицами ковариации
% Вариант 13
% m1=[1 2], m2=[1 -1], C1=[3 -1; -1 3], C2=[5 2; 2 6].
% Построить график разности суммарной экспериментальной и теоретической
% ошибок первого рода (для первого класса) от числа испытаний (объема
% выборки.
clear all; close all;
%% 1. Задание исходных данных
n = 2; M = 2; % размерность признакового пространства и число классов
K = 2000; % количество статистических испытаний
% Априорные вероятности, математические ожидания и матрицы ковариации классов
C = zeros(n, n, M);
C_ = C; % матрица ковариации вектора признаков различных классов
pw = [0.4 0.6];
pw = pw / sum(pw);
D = 3 * eye(2);
m = [1 2; 1 -1]';
C(:,:,1) = [3 -1; -1 3]; C(:,:,2) = [5 2; 2 6];
for k = 1 : M
    C_(:,:,k) = C(:, :, k) ^ -1;
end
np = sum(pw);
pw = pw / np; % исключение некорректного задания априорных вероятностей
%%  2. Расчет матриц вероятностей ошибок распознавания
PIJ = zeros(M);
PIJB = zeros(M);
mg = zeros(M);
Dg = zeros(M);
l0_ = zeros(M);
for i = 1 : M
    for j = i + 1 : M
            dmij = m(:, i) - m(:, j);
            l0_(i,j) = log(pw(j) / pw(i));
            dti = det(C(:, :, i));
            dtj = det(C(:, :, j));
            trij = trace(C_(:, :, j) * C(:, :, i) - eye(n));
            trji = trace(eye(n) - C_(:, :, i) * C(:, :, j));
            trij_2 = trace((C_(:, :, j) * C(:, :, i) - eye(n)) ^ 2);
            trji_2 = trace((eye(n) - C_(:, :, i) * C(:, :, j)) ^ 2);
            mg1 = 0.5 * (trij + dmij' * C_(:, :, i) * dmij - log(dti / dtj));
            Dg1 = 0.5 * trij_2 + dmij' * C_(:, :, j) * C(:, :, i) * C_(:, :, j) *
dmij;
            mg2 = 0.5 * (trji - dmij' * C_(:, :, j) * dmij + log(dtj / dti));
            Dg2 = 0.5 * trji_2 + dmij' * C_(:, :, i) * C(:, :, j) * C_(:, :, i) *
dmij;
            sD1 = sqrt(Dg1);
            sD2 = sqrt(Dg2);
            PIJ(i, j) = normcdf(l0_(i, j), mg1, sD1);
            PIJ(j, i) = 1 - normcdf(l0_(i, j), mg2, sD2);
            mu2 = (1 / 8) * dmij' * ((C(:, :, i) / 2 + C(:, :, j) / 2) ^ -1) * dmij
+ 0.5 * log((dti + dtj) / ( 2 * sqrt(dti * dtj))); % расстояние Бхатачария
            PIJB(i, j) = sqrt(pw(j) / pw(i)) * exp(-mu2);
```

```
            PIJB(j, i) = sqrt(pw(i) / pw(j)) * exp(-mu2); % границы Чернова
    end
    PIJB(i, i) = 1 - sum(PIJB(i, :));
    PIJ(i, i) = 1 - sum(PIJ(i, :)); % нижняя граница вероятности правильного
распознавания
end
%% 3. Тестирование алгоритма методом статистических испытаний
x = ones(n, 1);
u = zeros(M, 1);
Pc_ = zeros(M); % экспериментальная матрица вероятностей ошибок
for k = 1 : K % цикл по числу испытаний
    for i = 1 : M % цикл по классам
        [x, px] = randncor(n, 1, C(:, :, i));
        x = x + m(:, i); % генерация образа i-го класса
        for j = 1 : M % вычисление значения разделяющих функций
            u(j) = -0.5 * (x - m(:, j))' * C_(:, :, j) * (x - m(:, j)) - 0.5 *
log(det(C(:, :, j))) + log(pw(j));
        end
        [ui, iai] = max(u); % определение максимума
        Pc_(i, iai) = Pc_(i, iai) + 1; % фиксация результата распознавания
    end
end
Pc_ = Pc_ / K;
disp('Теоретическая матрица вероятностей ошибок');
disp(PIJ);
disp('Матрица вероятностей ошибок на основе границы Чернова');
disp(PIJB);
disp('Экспериментальная матрица вероятностей ошибок');
disp(Pc_);
%% 4. Визуализация разности суммарной экспериментальной и теоретической
% ошибок первого рода
a = PIJ(1, 2);
apc_ = zeros(K);
Pc_ = zeros(M); % экспериментальная матрица вероятностей ошибок
for ki = 1 : K
    for k = 1 : ki % цикл по числу испытаний
        i = 1;
        [x, px] = randncor(n, 1, C(:, :, i));
        x = x + m(:, i); % генерация образа 1-го класса
        for j = 1 : M % вычисление значения разделяющих функций
            u(j) = -0.5 * (x - m(:, j))' * C_(:, :, j) * (x - m(:, j)) - 0.5 *
log(det(C(:, :, j))) + log(pw(j));
        end
        [ui, iai] = max(u); % определение максимума
        Pc_(i, iai) = Pc_(i, iai) + 1; % фиксация результата распознавания
    end
    Pc_ = Pc_ / k;
    apc_(ki) = Pc_(1, 2);
end
apcDiff = apc_ - a;
figure;
plot(apcDiff);
xlim([1, K]);
ylim([-1, 1]);
axis auto;
```

```
title('Зависимость разности суммарной экспериментальной и теоретической ошибок
первого рода от числа испытаний', 'FontName', 'Courier');
xlabel('Число испытаний', 'FontName', 'Courier');
ylabel('Значение разности', 'FontName', 'Courier');
%% 5. Визуализация областей принятия решений для двумерного случая
if n == 2
    Es1 = pw(1) * PIJ(1, 2) + pw(2) * PIJ(2,1);
    Es2 = sqrt(pw(1) * pw(2)) * exp(-mu2); % граница Чернова для суммарной ошибки
    Es3 = pw(1) * Pc_(1, 2) + pw(2) * Pc_(2, 1);
    disp('Оценки суммарных ошибок');
    disp([Es1, Es2, Es3]); % отображение оценок суммарных ошибок
    xmin1 = -3 * sqrt(max(D(1, :))) + min(m(1, :));
    xmax1 = 3 * sqrt(max(D(1, :))) + max(m(1, :));
    xmin2 = -3* sqrt(max(D(2, :))) + min(m(2, :));
    xmax2 = 3* sqrt(max(D(2, :))) + max(m(2, :));
    x1 = xmin1 : 0.1 : xmax1;
    x2 = xmin2 : 0.1 : xmax2;
    axis([xmin1, xmax1, xmin2, xmax2]); % установка границ поля графика по осям
    figure;
    hold on;
    grid on;
    [X1, X2] = meshgrid(x1, x2); % матрицы значений координат случайного вектора
    x12 = [X1(:), X2(:)];
    for i = 1 : M
        f2 = mvnpdf(x12, m(:, i)', C(:, :, i)); % массив значений плотности
распределения
        f3 = reshape(f2, length(x2), length(x1)); % матрица значений плотности
распределения
        [Ch, h] = contour(x1, x2, f3, [0.01, 0.5 * max(f3(:))], 'Color', 'b',
'LineWidth', 0.75);
        clabel(Ch, h);
        for j = i + 1 : M % изображение разделяющих границ
            wij = C_(:, :, i) * m(:, i) - C_(:, :, j) * m(:, j);
            wij0 = -0.5 * (m(:, i)' * C_(:, :, i) * m(:, i) - m(:, j)' * C_(:, :,
j) * m(:, j));
            f4 = wij' * x12' + wij0 - 0.5 * log(det(C(:, :, i)) / det(C(:, :, j)));
            fd = -0.5 * (C_(:, :, i) - C_(:, :, j)) * x12';
            f4 = f4 + sum(x12' .* fd);
            f5 = reshape(f4, length(x2), length(x1));
            [Ch_, h_] = contour(x1, x2, f5, l0_(i, j), 'Color', 'k', 'LineWidth',
1.25);
        end
    end
    set(gca, 'FontSize', 13);
    title('Области локализации классов и разделяющие границы', 'FontName',
'Courier');
    xlabel('x1', 'FontName', 'Courier');
    ylabel('x2', 'FontName', 'Courier');
    strv1 = ' pw=';
    strv2 = num2str(pw, '% G');
    text(0, 0, [strv1, strv2], 'HorizontalAlignment', 'left', 'BackgroundColor',
[.8 .8 .8], 'FontSize', 12);
    legend('wi', 'gij(x)=0');
    hold off;
end
```

Результаты выполнения задания

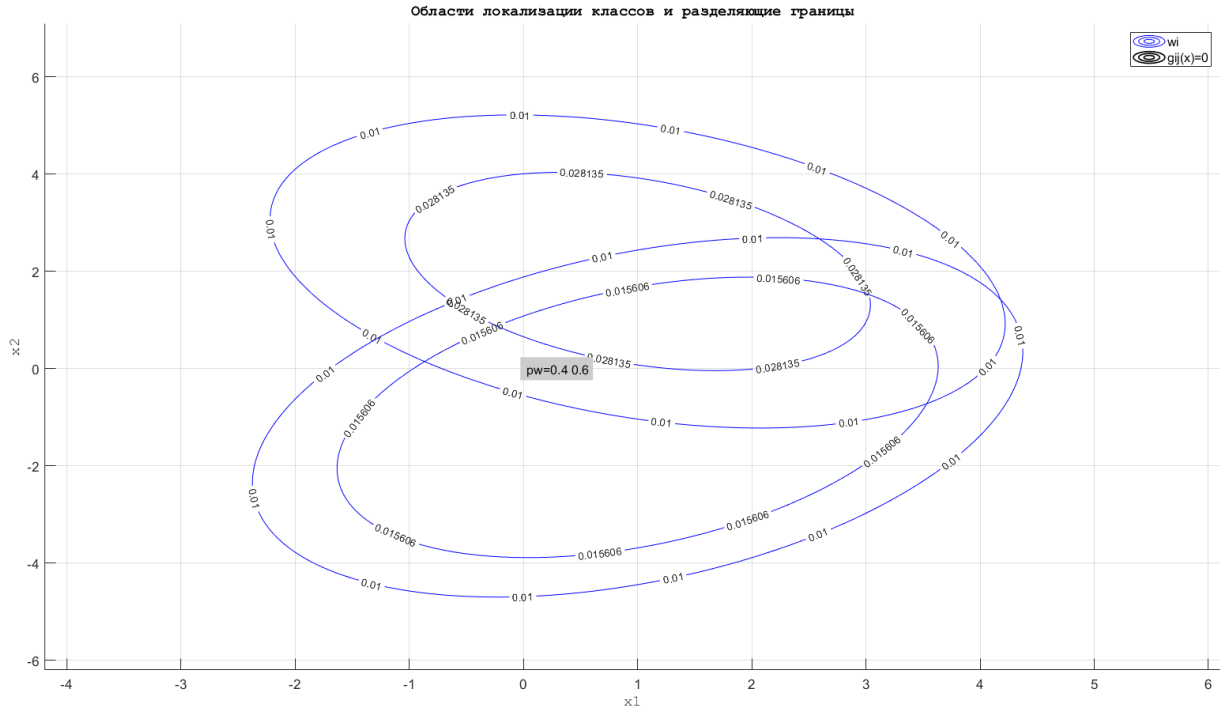

Рисунок 1 - Области локализации классов и разделяющие границы

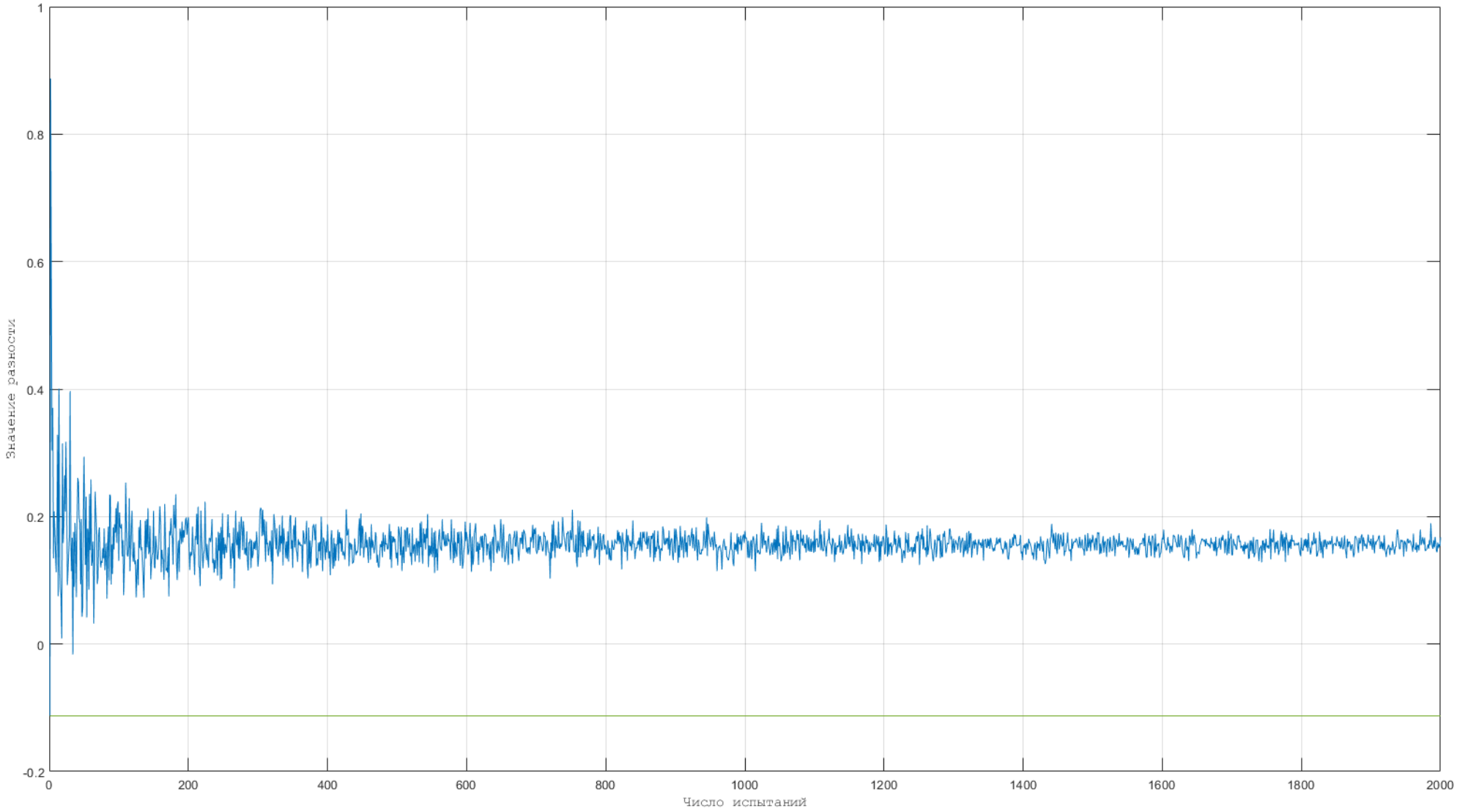

Рисунок 2 - Зависимость разности суммарной экспериментальной и теоретической ошибок первого рода от числа испытаний

Выводы

1. При увеличении числа испытаний разность ошибок начинает сходится к определённому значению. В данном случае к значению примерно 0,15.
2. Элементы главной диагонали матрицы ошибок показывают вероятность принятия правильного решения при классификации объекта в данный класс.
3. Элементы побочной диагонали характеризуют вероятность ошибки отнесения объекта к неправильному классу: объекта первого класса ко второму классу (ошибка первого рода), а объекта второго класса к первому (ошибка первого рода).
4. Формы кластеров объектов в пространстве используемых признаков определяются матрицей ковариации.