Цель работы: исследовать интенсивность отказов для невосстанавливаемых систем в разные периоды жизни.

Основные показатели надежности систем:

1) Функция надежности R(t) — это вероятность безотказной работы в течение заданного времени t. Практическое значение – количество система, которые работаю в момент t / количество всех систем. Теоретическая формула зависит от соединения.

2) Интенсивность отказа l(t) определяет условную плотность вероятности отказа объекта в момент времени, следующий непосредственно за моментом t.

   Практическая формула - $\frac{N_t+N_{t+del}}{Nt}$

   Теоретическая формула - $\frac{R'(t)*(-1)}{R(t)}$

Первый период жизни: период приработки.

Множество исследуемых экземпляров систем разбивается на k подмножеств. Попадание экземпляра во множество j характеризуется вероятностью pj. Подмножество j характеризуется своей интенсивностью отказов λj. Предполагается, что каждый экземпляр системы состоит из единственного элемента и в течение всего времени функционирования системы (вплоть до момента отказа), относящейся к подмножеству j, интенсивность отказов λj остается величиной постоянной. В данном периоде многие приборы отказывают из-за брака, интенсивность отказа подает, потому что с каждой единицей времени модели с браком отбрасываются.

Выбраны 2 группы:

float lambda1 = 0.8; float lambda2 = 1.5; float p1 = 0.5; float p2 = 0.5; int N = 50000;

Результат работы первого периода:

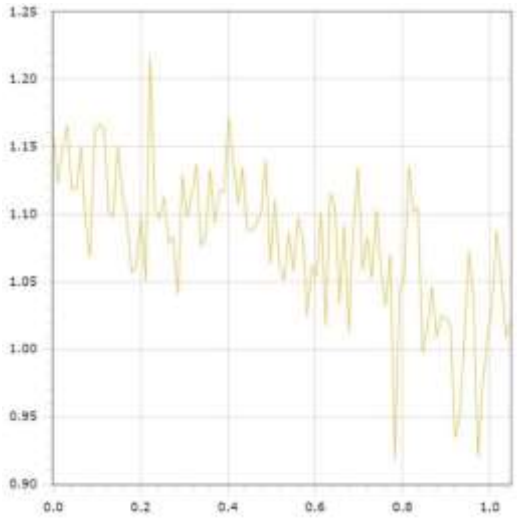

Рисунок 1 - интенсивность отказа

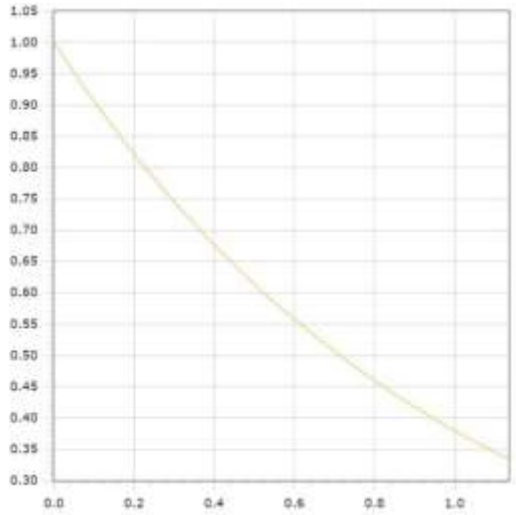

Рисунок 2 - функция надежности

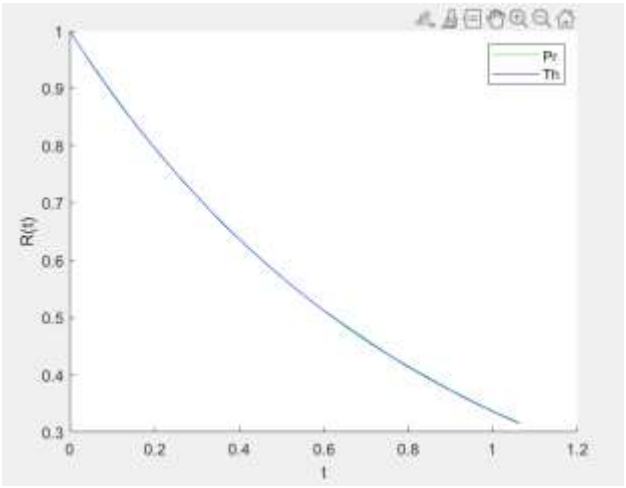

Рисунок 3 - сравнение функции надежности практической и теоритической

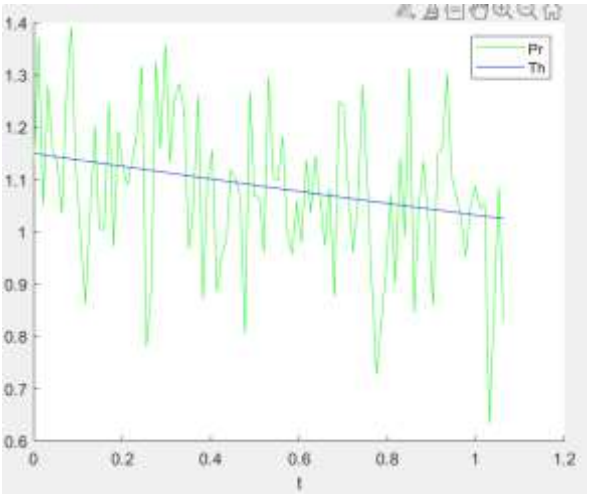

Рисунок 4 - сравнение функции интенсивности отказа теоретической и практической

Уравнение для теоретического вычисления функции надежности:

$$R(t) = e^{-l1t} * p1 + e^{-l2t} * p2$$

Второй период жизни: период нормального функционирования.

В качестве модели системы используется схема с последовательным соединением n элементов. Элемент i характеризуется интенсивностью отказов λi , которая является постоянной величиной. В данный период приборы работают без отказов, поэтому интенсивность отказа равна константе.

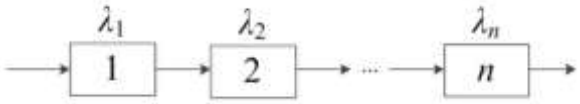

Рисунок 5 - Схема с последовательный соединением

Результаты работы второго периода:

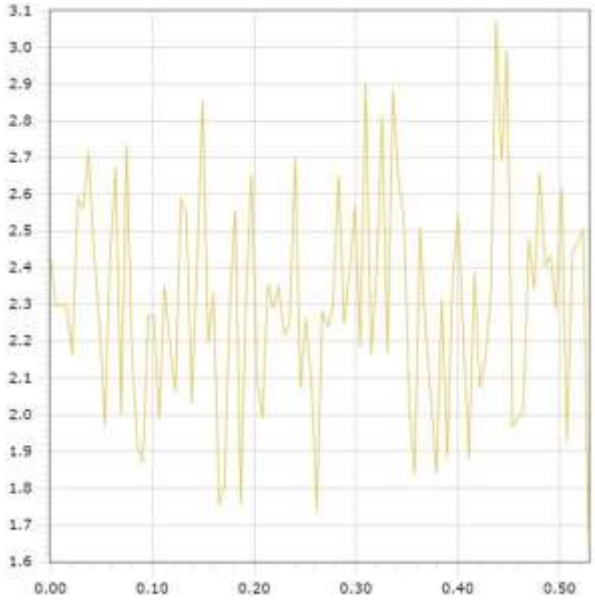

Рисунок 6 - интенсивность отказа

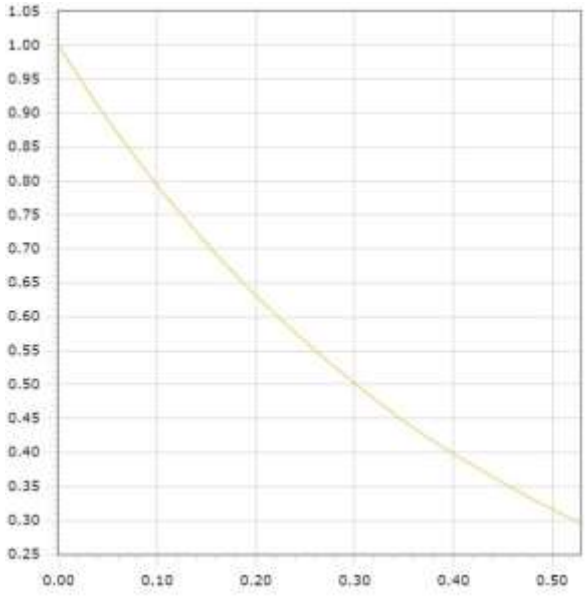

Рисунок 7 - функция надежности

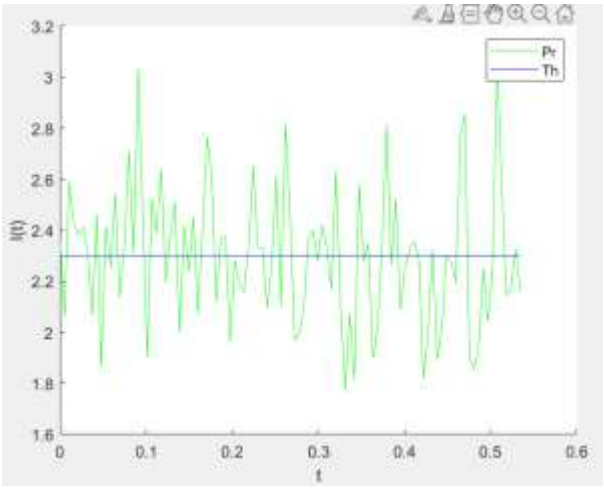

Рисунок 8 - сравнение интенсивности отказа практического и теоретического

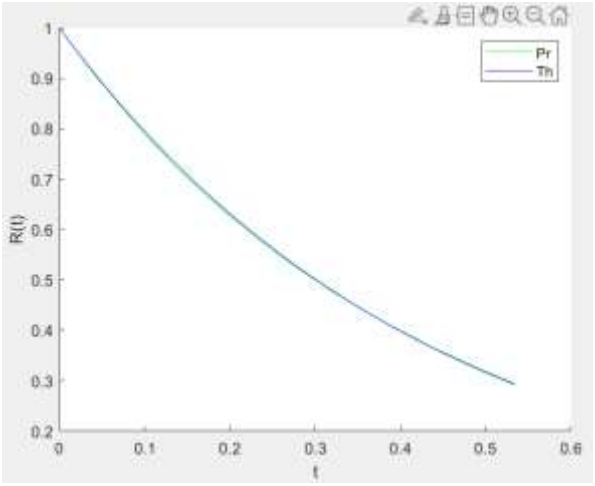

Рисунок 9 - сравнение функций надежности теоретической и практической

Уравнение для теоретического вычисления функции надежности:

$R(t) = e^{-(l1+l2)t}$

Уравнение для теоретического вычисления интенсивности отказа:

L(t) = l1 +l2

Третий период жизни: период старения.

Используется схема с параллельным соединением. В этом периоде интенсивность отказов растет, потому что приборы из-за старения ломаются.

Результаты работы третьего периода (элементы с параллельным соединением):

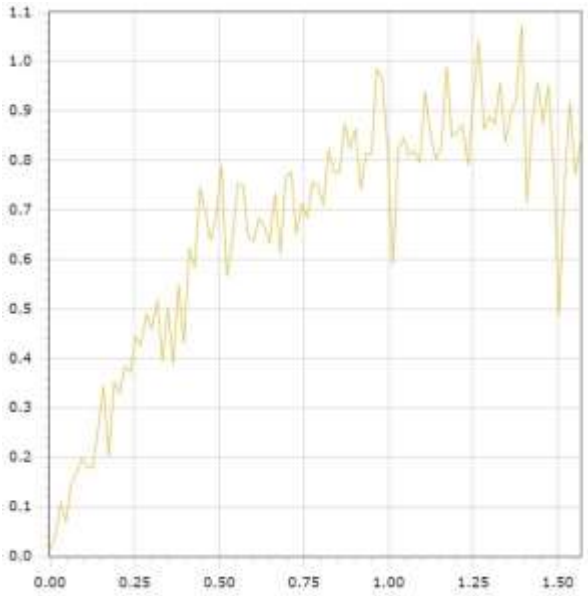

Рисунок 10 - интенсивность отказа

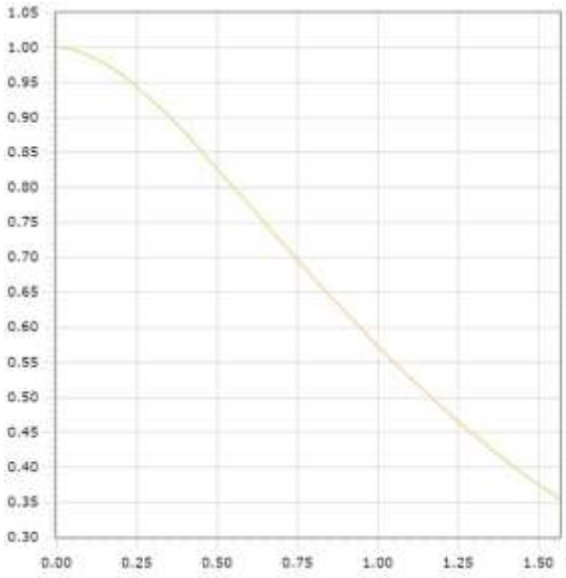

Рисунок 11 - функция надежности

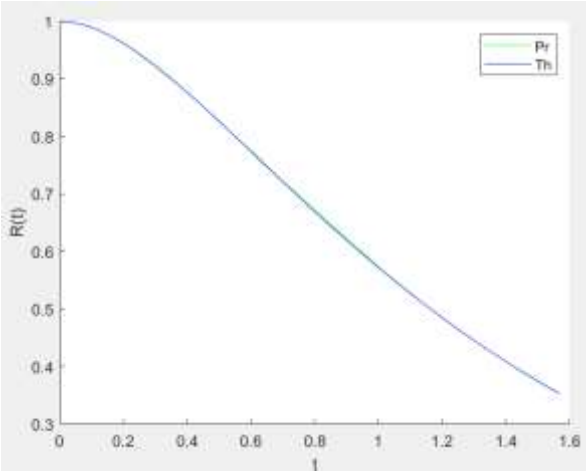

Рисунок 12 - сравнение функции надежности теоретической и практической

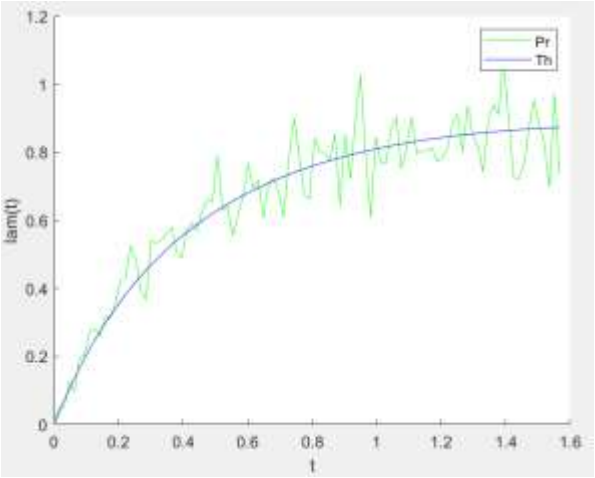

Рисунок 13 - сравнение интенсивности отказа теоретической и практической

Уравнение для теоретического вычисления функции надежности:

$$R(t) = R1(t) + R2(t) - R1(t) * R2(t)$$

Вывод: в ходе лабораторной работы были исследована интенсивность отказов для невосстанавливаемых систем в разные периоды жизни.

Листинг кода C++ (вычисление значений):

```
#include <iostream>
#include <time.h>
#include <cmath>
#include <vector>
#include <fstream>
#include <random>
using namespace std;

float lambda1 = 0.8;
float lambda2 = 1.5;
float p1 = 0.5;
float p2 = 0.6;
int N = 50000;

vector<double> generation(int code)
{
    vector<double> res;
    std::random_device rd;
    std::mt19937_64 generator = std::mt19937_64(rd());
    std::bernoulli_distribution random = std::bernoulli_distribution(p1);
    std::exponential_distribution<> tau1 = std::exponential_distribution<>(lambda1);
    std::exponential_distribution<> tau2 = std::exponential_distribution<>(lambda2);
    for (int i = 0; i < N; ++i) {
        if (code == 1)
        {
            if (random(generator) <= p1) {
                res.push_back(tau1(generator));
            }
            else {
                res.push_back(tau2(generator));
            }
        }
        else if (code == 2)
        {
            res.push_back(min(tau1(generator), tau2(generator)));
        }
        else if (code == 3)
        {
            res.push_back(max(tau1(generator), tau2(generator)));
        }
    }
    return res;
}

void solve(vector<double> T, int code)
{
    double ttt = 0;
    for (double t : T) {
        ttt += t;
    }
    ttt /= N;
    ttt += 0.1;
    double step = ttt / 100;
    double delta_step = step / 5;
    std::vector<double> rt;
    std::vector<double> lambdat;
    std::vector<double> time;
    vector<int> Nt;
    vector<int> DelS;
    for (double i = 0; i < ttt; i = i + step)
    {
        int nt = 0;
        time.push_back(i);
```

```
    for (double t : T)
    {
        if (t >= i)
        {
            ++nt;
        }
    }
    Nt.push_back(nt);
    rt.push_back((double)nt / N);
}

for (double i = delta_step; i < ttt + delta_step; i = i + step)
{
    int count = 0;
    for (double t : T)
    {
        if (t >= i)
        {
            count++;
        }
    }
    DelS.push_back(count);
}

for (int i = 0; i < Nt.size(); i++)
{
    int tmp1 = Nt[i] - DelS[i];
    double tmp2 = (double)tmp1;
    double tmp3 = (double)(tmp2 / double(Nt[i]));
    double tmp4 = tmp3 * (double)(1 / delta_step);
    lambdat.push_back(tmp4);
}

auto iterTime = time.begin();
std::cout << step << std::endl;
std::cout << ttt << std::endl;
for (double v : rt) {
    std::cout << *iterTime << " " << v << endl;
    iterTime++;
}
std::cout << std::endl;
iterTime = time.begin();
for (double v : lambdat) {
    std::cout << *iterTime << " " << v << endl;
    iterTime++;
}
std::cout << std::endl;

std::ofstream out0;
out0.open("time.txt");
if (out0.is_open())
{
    auto iter1 = time.begin();
    for (; iter1 != time.end(); iter1++, )
    {
        out0 << *iter1  << endl;
    }
}

std::ofstream out1;
if (code == 1)
{
    out1.open("Rt1.txt");
}
else if (code == 2)
```

```
    {
        out1.open("Rt2.txt");
    }
    else if (code == 3)
    {
        out1.open("Rt3.txt");
    }
    if (out1.is_open())
    {
        auto iter1 = time.begin();
        auto iter2 = rt.begin();
        for (; iter1 != time.end(); iter1++, iter2++)
        {
            out1 << *iter1 << " " << *iter2 << endl;
        }
    }

    std::ofstream out2;
    if (code == 1)
    {
        out2.open("Lam1.txt");
    }
    else if (code == 2)
    {
        out2.open("Lam2.txt");
    }
    else if (code == 3)
    {
        out2.open("Lam3.txt");
    }
    if (out2.is_open())
    {
        auto iter1 = time.begin();
        auto iter2 = lambdat.begin();
        for (; iter1 != time.end(); iter1++, iter2++)
        {
            out2 << *iter1 << " " << *iter2 << endl;
        }
    }
}

void period1()
{
    vector<double> T = generation(1);
    solve(T, 1);
}

void period2()
{
    vector<double> T = generation(2);
    solve(T, 2);
}

void period3()
{
    vector<double> T = generation(3);
    solve(T, 3);
}

int main()
{
    period1();
    period2();
    period3();
    return 0;
```

```
}
```

Листинг кода MatLab (построение графиков):

```
clear;
lam1 = 0.8;
lam2 = 1.5;
p = 0.5;
infig = 1;

% График функции надежности в первом периоде ==============================

A1 = importdata("Rt1.txt");
time1 = zeros(1, length(A1));
practika1 = zeros(1, length(A1));
for k = 1:length(A1)
    time1(k) = A1(k, 1);
    practika1(k) = A1(k, 2);
end
Rt1 = zeros(1, length(A1));
for k = 1:length(A1)
    tmp1 = exp((-1)*lam1*time1(k))*p;
    tmp2 = exp((-1)*lam2*time1(k))*p;
    Rt1(k) = tmp1+tmp2;
end

% График интенсивности отказа в первом периоде ============================

zw1 = importdata("Lam1.txt");
time6 = zeros(1, length(zw1));
practika6 = zeros(1, length(zw1));
for k = 1:length(zw1)
    time6(k) = zw1(k, 1);
    practika6(k) = zw1(k, 2);
end
th6 = zeros(1, length(zw1));
for k = 1:length(time6)
    y = -(2/5*exp(-4/5*time6(k)))-(3/4*exp(-3/2*time6(k)));
    th6(k) = y/Rt1(k)*(-1);
end
figure(infig);
infig = infig + 1;
hold on;
plot (time6, practika6, 'Color', 'g');
plot (time6, th6, 'Color', 'b');
xlabel('t');
ylabel('R(t)');
legend('Pr', 'Th');
hold off;

figure(infig);
infig = infig + 1;
hold on;
plot (time1, practika1, 'Color', 'g');
plot (time1, Rt1, 'Color', 'b');
xlabel('t');
ylabel('R(t)');
legend('Pr', 'Th');
hold off;

% График функции надежности во втором периоде ==============================
```

```
A2 = importdata("Rt2.txt");
time2 = zeros(1, length(A2));
practika2 = zeros(1, length(A2));
for k = 1:length(A2)
    time2(k) = A2(k, 1);
    practika2(k) = A2(k, 2);
end
Rt2 = zeros(1, length(A2));
for k = 1:length(A2)
    tmp1 = exp((-1)*(lam1+lam2)*time2(k));
    Rt2(k) = tmp1;
end
figure(infig);
infig = infig + 1;
hold on;
plot(time2, practika2, 'Color', 'g');
plot(time2, Rt2, 'Color', 'b');
xlabel('t');
ylabel('R(t)');
legend('Pr', 'Th');
hold off;

% График интенстивности отказа во втором периоде ==========================

A4 = importdata("Lam2.txt");
time4 = zeros(1, length(A4));
practika4 = zeros(1, length(A4));
for k = 1:length(A4)
    time4(k) = A4(k, 1);
    practika4(k) = A4(k, 2);
end
Lam2 = zeros(1, length(A4));
for k = 1:length(A4)
    Lam2(k) = lam1 + lam2;
end
figure(infig);
infig = infig + 1;
hold on;
plot(time4, practika4, 'Color', 'g');
plot(time4, Lam2, 'Color', 'b');
xlabel('t');
ylabel('l(t)');
legend('Pr', 'Th');
hold off;

% График функции надежности в третьем периоде ============================

A5 = importdata("Rt3.txt");
time5 = zeros(1, length(A5));
practika5 = zeros(1, length(A5));
for k = 1:length(A5)
    time5(k) = A5(k, 1);
    practika5(k) = A5(k, 2);
end
Rt3 = zeros(1, length(A5));
for k = 1:length(A5)
    tmp1 = exp((-1)*lam1*time5(k));
    tmp2 = exp((-1)*lam2*time5(k));
    Rt3(k) = tmp1 + tmp2 - tmp1*tmp2;
end
figure(infig);
infig = infig + 1;
```

```
hold on;
plot (time5, practika5, 'Color', 'g');
plot (time5, Rt3, 'Color', 'b');
xlabel('t');
ylabel('R(t)');
legend('Pr', 'Th');
hold off;

% График интенсивности отказа в третьем периоде

zw2 = importdata("Lam3.txt");
time7 = zeros(1, length(zw2));
practika7 = zeros(1, length(zw2));
for k = 1:length(zw2)
    time7(k) = zw2(k, 1);
    practika7(k) = zw2(k, 2);
end
th7 = zeros(1, length(zw2));
for k = 1:length(time7)
    tmp1 = exp(-31/10*time7(k));
    tmp2 = 23*exp(4/5*time7(k));
    tmp3 = 8*exp(23/10*time7(k));
    tmp4 = -(3/2)*exp(-3/2*time7(k));
    y = (tmp1*(tmp2-tmp3))/10 + tmp4;
    th7(k) = y/Rt3(k)*(-1);
end
figure(infig);
infig = infig + 1;
hold on;
plot (time7, practika7, 'Color', 'g');
plot (time7, th7, 'Color', 'b');
xlabel('t');
ylabel('lam(t)');
legend('Pr', 'Th');
hold off;
```