|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ «Информатика и системы управления»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Лабораторная работа № 2*

*По предмету: «Математическая статистика»*

**Тема: Интервальные оценки**

*Вариант 25*

Студент: Юмаев Артур Русланович

Группа: ИУ7-65Б

Оглавление

[Цель и содержание работы 3](#_Toc40751531)

[1. Определение – доверительного интервала для значения параметра распределения случайной величины 4](#_Toc40751532)

[2. Формулы для вычисления – доверительного интервала для математического ожидания и дисперсии нормальной случайно величины 4](#_Toc40751533)

[2.1 Доверительная оценка для математического ожидания при неизвестной дисперсии 5](#_Toc40751534)

[2.2 Доверительная оценка для дисперсии при неизвестном математическом ожидании 5](#_Toc40751535)

[3. Текст программы 6](#_Toc40751536)

[4. Результат работы программы и графики 8](#_Toc40751537)

# Цель и содержание работы

**Цель работы:** построение доверительных интервалов для математического ожидания и дисперсии нормальной случайной величины.

**Содержание работы:**

1. Для выборки объема из генеральной совокупности реализовать в виде программы на ЭВМ
   1. вычисление точечных оценок и математического ожидания и дисперсии соответственно;
   2. вычисление нижней и верхней границ , для – доверительного интервала для математического ожидания ;
   3. вычисление нижней и верхней границ , для – доверительного интервала для математического ожидания ;
2. вычислить и для выборки из индивидуального варианта;
3. для заданного пользователем уровня доверия и *N* – объема выборки из индивидуального варианта:
   1. на координатной плоскости *Oyn* построить прямую , также графики функций , и как функций объема *n* выборки, где *n* изменяется от 1 до *N*.
   2. на другой координатной плоскости *Ozn* построить прямую , также графики функций , и как функций объема *n* выборки, где *n* изменяется от 1 до *N*.

# 1. Определение – доверительного интервала для значения параметра распределения случайной величины

Пусть — случайная выборка объема *n* из генеральной совокупности *X* с функцией распределения , зависящей от параметра *θ*, значение которого неизвестно. Предположим, что для параметра *θ* построен интервал границ , где и являются функциями случайной выборки , такими, что выполняется равенство

*.*

В этом случае интервал называют ***интервальной оценкой*** для параметра *θ* с коэффициентом доверия . А соответственно ***нижней*** и ***верхней*** ***границами*** интервальной оценки.

Вероятностной характеристикой точности оценивания параметра *θ* является случайная величина

которая для любой реализации случайной выборки есть длина интервала . Интервал называют ***доверительным интервалом*** для параметра *θ* с коэффициентом доверия или ***доверительным интервалом***.

# 2. Формулы для вычисления – доверительного интервала для математического ожидания и дисперсии нормальной случайно величины

Оценка математического ожидания:

Несмещенная оценка дисперсии:

Пусть – случайная выборка объема *n* из генеральной совокупности *X*, распределенной по нормальному закону с параметрами и .

## 2.1 Доверительная оценка для математического ожидания при неизвестной дисперсии

При неизвестной дисперсии статистика

имеет *распределение Стьюдента* с *n – 1* степенями свободы, поэтому

где *n* – объем выборки, – квантиль уровня *q* *распределения Стьюдента* с

*n – 1* степенями свободы, .

## 2.2 Доверительная оценка для дисперсии при неизвестном математическом ожидании

Статистика

имеет – распределение с *n – 1* степенями свободы. Поэтому

где – квантиль уровня *q* – распределения с *n – 1* степенями свободы, , .

# 3. Текст программы

Листинг 1. Программа на Matlab к лабораторной работе №2

|  |
| --- |
| function lab2  clear  X = [-17.04,-18.29,-17.38,-18.11,-18.96,-17.65,-17.02,-17.22,-16.25,-17.44,-17.69,-17.61,-17.09,-17.19,-16.02,-17.56,-16.94,-17.29,-16.93,-16.61,-19.38,-17.53,-16.39,-17.89,-17.98,-17.04,-16.22,-19.09,-18.91,-17.77,-18.30,-17.44,-18.84,-16.39,-16.13,-18.37,-16.37,-16.70,-17.78,-17.03,-17.76,-17.87,-17.20,-18.44,-17.19,-17.75,-16.81,-17.97,-18.03,-16.87,-16.10,-19.16,-16.51,-18.39,-16.48,-18.08,-17.49,-18.89,-19.09,-17.96,-18.40,-16.96,-18.15,-18.71,-17.81,-17.86,-19.47,-17.86,-17.60,-17.30,-17.60,-17.71,-18.42,-16.88,-16.76,-18.00,-17.97,-16.83,-18.00,-18.08,-17.61,-17.02,-16.73,-17.64,-18.76,-17.68,-18.04,-16.45,-18.79,-18.03,-17.38,-15.27,-15.97,-17.41,-18.61,-18.00,-17.42,-17.77,-19.05,-16.16,-16.27,-18.00,-18.90,-17.05,-17.46,-17.49,-18.20,-17.59,-15.78,-18.88,-18.53,-17.39,-17.83,-18.17,-16.15,-17.66,-17.76,-18.32,-17.70,-17.56];  N = length(X);  gamma = 0.9;  alpha = (1 - gamma)/2;  mu = findMu(X);  S2 = findS2(X);  muArray = getMuArray(X, N);  varArray = getVarArray(X, N);   muHigh = findMuHigh(muArray, varArray, alpha, N);  muLow = findMuLow(muArray, varArray, alpha, N);   sigma2High = getsigma2High(varArray, alpha, N);  sigma2Low = getsigma2Low(varArray, alpha, N);   figure  hold on;  plot([1,N], [mu, mu], 'g');  plot((1:N), muArray, 'r');  plot((1:N), muLow, 'b');  plot((1:N), muHigh, 'm');  legend('mu(x\_N)','mu(x\_n)','muLow(x\_n)','muHigh(x\_n)');  grid on;  hold off;   figure  hold on;  plot([1,N], [S2, S2], 'g');  plot((1:N), varArray, 'r');  plot((1:N), sigma2Low, 'b');  plot((4:N), sigma2High(4:length(sigma2High)), 'm');  legend('S^2(x\_N)','S^2(x\_n)','(sigma^2)Low(x\_n)','(sigma^2)High(x\_n)');  grid on;  hold off;   N = size(X,2);  muArray = getMuArray(X, N);  varArray = getVarArray(X, N);  muHigh = findMuHigh(muArray, varArray, alpha, N);  muLow = findMuLow(muArray, varArray, alpha, N);  sigma2High = getsigma2High(varArray, alpha, N);  sigma2Low = getsigma2Low(varArray, alpha, N);   fprintf('mu = %.2f\n', mu);  fprintf('S^2 = %.2f\n', S2);  fprintf('mu\_low = %.2f\n', muLow(end));  fprintf('mu\_high = %.2f\n', muHigh(end));  fprintf('sigma^2\_low = %.2f\n', sigma2Low(end));  fprintf('sigma^2\_high = %.2f\n', sigma2High(end));   function mu = findMu(X)  mu = sum(X)/size(X,2);  end   function sigma = getSigmaSqr(X)  tempMu = findMu(X);  sigma = sum((X - tempMu) .\* (X - tempMu))/size(X,2);  end   function S2 = findS2(X)  n = size(X,2);  S2 = n/ (n - 1) \* getSigmaSqr(X);  end   function muArray = getMuArray(X, N)  muArray = zeros(1,N);  for i = 1:N  muArray(i) = findMu(X(1:i));  end  end   function varArray = getVarArray(X, N)  varArray = zeros(1,N);  for i = 1:N  varArray(i) = findS2(X(1:i));  end  end   function muHigh = findMuHigh(muArray, varArray, alpha, N)  muHigh = zeros(1, N);  for i = 1:N  muHigh(i) = muArray(i) + sqrt(varArray(i)./ i) .\* tinv(1 - alpha, i - 1);  end  end   function muLow = findMuLow(muArray, varArray, alpha, N)  muLow = zeros(1, N);  for i = 1:N  muLow(i) = muArray(i) + sqrt(varArray(i)./ i) .\* tinv(alpha, i - 1);  end  end   function sigma2High = getsigma2High(varArray, alpha, N)  sigma2High = zeros(1, N);  for i = 1:N  sigma2High(i) = varArray(i) .\* (i - 1) ./ chi2inv(alpha, i - 1);  end  end   function sigma2Low = getsigma2Low(varArray, alpha, N)  sigma2Low = zeros(1, N);  for i = 1:N  sigma2Low(i) = varArray(i) .\* (i - 1) ./ chi2inv(1 - alpha, i - 1);  end  end end |

# 4. Результат работы программы и графики

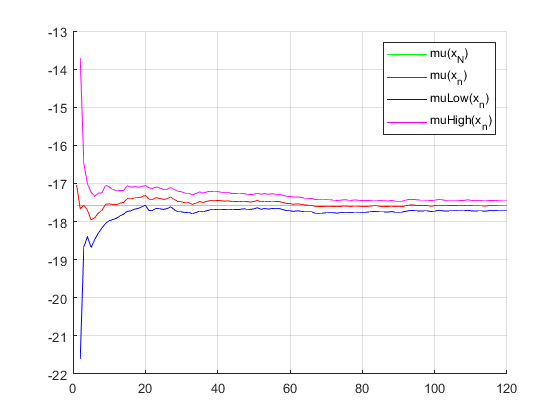


Рисунок 1. Оценка для математического ожидания

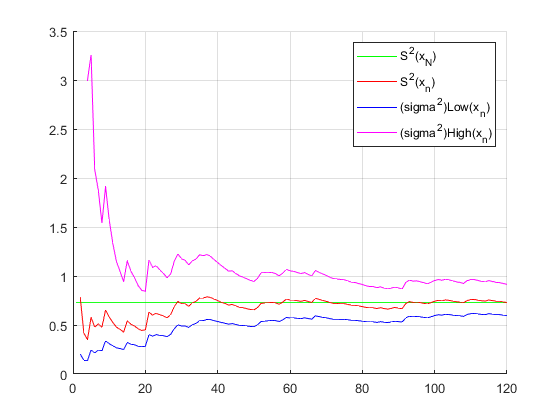


Рисунок 2. Оценка для дисперсии