**Казанский (Приволжский) Федеральный Университет**

**Институт вычислительной математики и информационных технологий**

**Направление:** **Прикладная математика и информатика**

Дисциплина: Прогнозирование временных рядов

Отчёт по учебной практике

студента гр. 09 – 812 Зарифянов Ю.З.

Казань – 2020

**Содержание**

**1**. Метод Ирвина

**2.** Критерии обнаружения тренда

**2.1** Критерий на основе средних

**2.2** Критерий на основе нисходящих и восходящих серий

**2.3** Критерий на основе медианы выборки

**3.** Сглаживание временных рядов

**3.1** Метод скользящей средней

**3.2** Экспоненциальный метод

**4.** Автокорреляция

**5.** Моделирование линейного тренда

**6.** Проверка адекватности модели

**7.** Моделирование сезонных и циклических колебаний

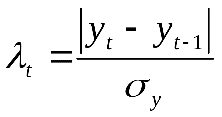
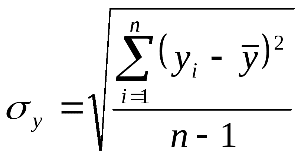
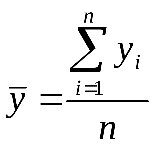
**1.Метод Ирвина**

Метод Ирвина используется для выявления аномальных значений уровней временного ряда. Под аномальным уровнем понимается отдельное значение уровней временного ряда, которое не отвечает потенциальным возможностям исследуемой экономической системы и которое, оставаясь в качестве уровня ряда, оказывает существенное влияние на значение основных характеристик временного ряда.

Причинами аномальных явлений могут быть ошибки технического порядка, или ошибки первого рода, они подлежат выявлению и устранению.

Кроме того, аномальные уровни во временных рядах могут возникать из-за воздействия факторов, имеющих объективный характер, но проявляющихся эпизодически. Их относят к ошибкам второго рода, которые не подлежат устранению.

Для выявления аномальных наблюдений может быть использован метод Ирвина. В этом случае вычисляется коэффициент λt, равный:

, , .

Расчетные значения λ2, λ3,... сравниваются с табличными значениями критерия Ирвина λα. Если оказывается, что расчетное значение λt больше табличного λα, то соответствующее значение yt уровня ряда считается аномальным.

После выявления аномальных значений уровней ряда обязательно определение причин их возникновения. Если точно установлено, что они вызваны ошибками первого рода, то они устраняются обычно заменой средней арифметической двух соседних уровней ряда, либо заменой значением соответствующей трендовой кривой.

Разрыв страницы

При проверке наличия аномальных колебаний с использованием метода Ирвина, получили следующие расчетные значения коэффициента λt:

Таблица

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| λ1 | λ2 | λ3 | λ4 | λ5 | λ6 | λ7 | λ8 | λ9 | λ10 |
| - | 1,443582 | 1,443582 | 1,443582 | 1,443582 | 1,443582 | 1,443582 | 1,443582 | 1,443582 | 1,443582 |
| λ11 | λ12 | λ13 | λ14 | λ15 | λ16 | λ17 | λ18 | λ19 | λ20 |
| 1,443582 | 1,443582 | 1,443582 | 1,443582 | 1,443582 | 1,443582 | 1,443582 | 1,443582 | 1,443582 | 1,443582 |

+Сравнивая найденные значения коэффициента λt с табличным значением λα, равным 1,3 для уровня значимости α = 0,05 и при n = 20 (число уровней временного ряда), получаем, что отдельные значения уровней ряда превосходят значение λα, следовательно делаем вывод о том, что в данной модели присутствуют аномальные колебания, вызванные ошибками второго рода, которые устранению не подлежат.

Данные:

X

110

111

118

125

126

121

126

142

154

161

Реализация метода Ирвина на языке Python(Входные данные в файле “arb.txt ”)

import numpy as np  
import pandas as pd  
import csv  
alfa = 0.05  
  
  
def Irvin(x, alpha):  
 mean = np.mean(x); dis = np.var(x, ddof=1)  
 mas = []; n = len(x); irwin=0  
 if (alpha == 0.05):  
 irwin = -229.21 \* n \*\* -3 + 422.39 \* n \*\* (-2.5) - 320.96 \* n \*\* (-2) + 124.594 \* n \*\* (-1.5) - 26.15 \* n \*\* (  
 -1) + 4.799 \* n \*\* (-0.5) + 0.7029  
 if (alpha == 0.01):  
 irwin = -205.06 \* n \*\* -3 + 424.26 \* n \*\* (-2.5) - 352.483 \* n \*\* (-2) + 143.747 \* n \*\* (-1.5) - 33.401 \* n \*\* (  
 -1) + 6, 381 \* n \*\* (-0.5) + 1.049  
 if (alpha == 0.1):  
 irwin = -132.78 \* n \*\* -3 + 224.24 \* n \*\* (-2.5) - 165.27 \* n \*\* (-2) + 68.614 \* n \*\* (-1.5) - 16.109 \* n \*\* (  
 -1) + 3.693 \* n \*\* (-0.5) + 0.549  
 mas.append(irwin)  
 for i in range(1, n):  
 x1 = abs(x[i] - x[i - 1]) / dis  
 mas.append(x1)  
 print("Критический уровень I:", irwin, "\n")  
 for i in range(n):  
 if (mas[i] > irwin):  
 print(i, x[i], mas[i], "Есть аномалия")  
 else:  
 print(i, x[i], mas[i], "Нет аномалии")  
  
f=open("arb.txt", "r")  
Y=[]  
for line in f:  
 print(line)  
 Y.append(int(line))  
  
lenx = len(Y)  
Y = [float(val) for val in Y]  
print(Y)  
  
Irvin(Y, alfa)

Результат:

0 110.0 1.4423897379035169 Нет аномалии

1 111.0 0.0033083370092633437 Нет аномалии

2 118.0 0.023158359064843405 Нет аномалии

3 125.0 0.023158359064843405 Нет аномалии

4 126.0 0.0033083370092633437 Нет аномалии

5 121.0 0.016541685046316718 Нет аномалии

6 126.0 0.016541685046316718 Нет аномалии

7 142.0 0.0529333921482135 Нет аномалии

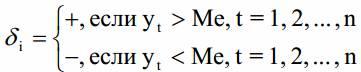
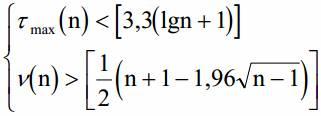
8 154.0 0.03970004411116013 Нет аномалии

9 161.0 0.023158359064843405 Нет аномалии

**2.Проверка существования тренда**

**2.1. Критерий серий основанный на медиане выборки**

Критерий серий, основанный на медиане выборки, реализуется в виде следующей последовательности шагов:

1. Из исходного ряда yt длиной *n* образуется ранжированный (вариационный) ряд yt
2. Определяется медиана ряда Me. В случае нечетного значения *n* (n=2m+1) Me=ym+1, в противном случае Me=(ym+ym+1)/2.
3. Образуется последовательность δi из плюсов и минусов по следующему правилу:  
     
   Если значение yt равно медиане, то это значение пропускается.
4. Подсчитывается v(n) - число серий в совокупности δi, где под серией понимается последовательность подряд идущих плюсов или минусов. Один плюс или один минус тоже будет считаться серией. Определяется t(n) - протяженность самой длинной серии.
5. Проверка гипотезы основывается на том, что при условии случайности ряда (при отсутствии систематической составляющей) протяженность самой длинной серии не должна быть слишком большой, а общее число серий - слишком маленьким. Поэтому для того, чтобы не была отвергнута гипотеза о случайности исходного ряда (об отсутствии систематической составляющей) должны выполняться следующие неравенства (для 5% уровня значимости uкр = 1.96):  
     
   Если хотя бы одно из неравенств нарушается, то гипотеза об отсутствии тренда отвергается.  
   Квадратные скобки в правой части неравенства означают целую часть числа.

Входные данные

38

33

29

16

40

45

21

30

Реализация метода на языке Python

from statistics import median

import numpy as np

f=open("test.txt")

s=[int(i.strip()) for i in f]; n=len(s)

mean=median(s); b=True; kol=1

if s[0]>mean: b=True

else: b=False

for i in range(1,n):

if s[i]>mean and b==False:

b=True; kol+=1

if s[i]<mean and b==True:

b=False; kol+=1

if (kol>(n+2-(1.96\*np.sqrt(n-1)))/2) and (kol<13.3\*int(np.log10(n)+1)):

print("NO trend")

else:

print("YES trend")

Выходные данные: NO trend.

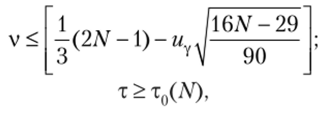
**2.2. Критерий нисходящих и восходящих серий.**

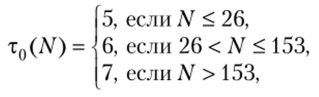
Данный критерий основан на использовании последовательности условных знаков. Правило формирование этой последовательности заключается в следующем:

● знак “+” ставят, если x(t+1)>x(t)

● знак “-” ставят, если x(t+1)<x(t)

Среди одинаковых подряд идущих элементов ряда учитывается только один из них, и серии знаков не прерывается. Полученные таким образов серии будут соответствовать участкам возрастания ( восходящие серии ) или убывание ( нисходящие серии ) значений элементов временного ряда. Этот критерий основан на том факте, что при непротиворечивости гипотезы отсутствия тренда длина полученных серий будет мала, а их общее количество велико. Гипотеза отвергается с вероятностью ошибки a, если выполняется хотя бы одно из неравенств





Uy – квантиль стандартного норм. Расп для доверительной вероятности.



Входные данные

38

33

29

16

40

45

21

30

Реализация метода на языке Python

Import math

with open("test.txt") as f:

a=[int(i) for i in f]; n=len(s)

c=[]

for i in range(n-1):

if a[i+1]>a[i]:

c.append('+')

elif a[i+1]<a[i]:

c.append('-')

rmax=1;r=1; v=1

for i in range(len(c)-1):

if c[i]!=c[i+1]:

rmax=max(r,rmax) ; r=1; v+=1

else:

r+=1

if v>(((2\*n-1)/3-1.96\*((16\*n-29)/90))\*\*(0.5))//1 and ((n<=26 and rmax<5) or (n>26 and n<=153 and rmax<6) or (n>153 and n<=1170 and rmax<7)):

print("No trend")

else:

print("Yes trend")

Выходные данные

No trend

**2.3. Сравнение средних уровней ряда.**

Проверка тренда у временного ряда сводится к проверке гипотезе о равенстве средних 2х выборок.

Выборка разбивается на 2 равные части. Находятся средние и дисперсии. После, сравниваются дисперсии по F-критерию Фишера, для дальнейшего использования двухвыборочного t-критерия Стьюдента. Выборки имеют нормальное распределение.

Входные данные

38

33

29

16

40

45

21

30

Реализация метода на языке Python

import numpy as np

import pandas as pd

from scipy.stats import f

from scipy.stats import t

data=pd.read\_csv("data.csv")

y=data['X']

print(y)

#разбиение выборки на 2 примерно равные

def trend\_existence(y):

    if (len(y)/2%2==0):

        n1=len(y)/2

        n2=len(y)/2

    else:

        n1=(len(y)-1)/2

        n2=len(y)-n1

    y1=[]

    y2=[]

    for item in range(len(y)):

        if item%2==0:

            y1.append(y[item])

        else:

            y2.append(y[item])

    #Обе получившиеся выборки должны иметь нормальное распределение

    #Проверка равенства дисперсий критерием Фишера

    var1 = np.var(y1,ddof=1)

    var2 = np.var(y2,ddof=1)

    S=var1/var2

    alpha=0.05#input("Введите уровень значимости")

    print("F расчётное: "+str(S))

    dfn=len(y1)-1

    dfd=len(y2)-1

    x = f.ppf(1-alpha, dfn, dfd)

    print("F табличное: "+str(x))

    if S>=x:

        print("Дисперсии различаются\nКритерий не подходит")

    else:

        print("Дисперсии равны")

        #Проверка равенства средних критерием Стьюдента

        t1=(abs(np.mean(y1)-np.mean(y2))/np.sqrt(dfn\*var1+dfd\*var2))\*np.sqrt(len(y1)\*len(y2)\*(dfn+dfd)/(len(y1)+len(y2)))

        print("t расчётное: "+str(t1))

        tt=t.ppf(1-alpha,dfn+dfd)

        print("t:",tt)

        if tt>t1:

            print("Тренда нет")

        else:

            print("Тренд возможен")

trend\_existence(y)

Выходные данные

Тренда нет

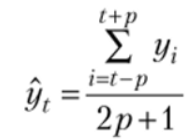
**3.Сглаживание временного ряда**

**3.1. Метод простой скользящей средней**

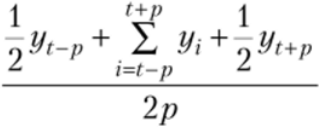
Для временного ряда y1,y2,...,yn определяется интервал сглаживания m (m < n). Если необходимо сгладить мелкие беспорядочные колебания, то интервал сглаживания берут по возможности большим; интервал сглаживания уменьшают, если нужно сохранить более мелкие колебания. При прочих равны условиях интервал сглаживания рекомендуют брать нечетным.

Для первых m уровней временного ряда вычисляется их средняя арифметическая; это будет сглаженное значение уровня ряда, находящегося в середине интервала сглаживания. Затем интервал сглаживания сдвигается на один уровень вправо, повторяется вычисление средней арифметической и т.д. Для вычисления сглаженных уровней ряда у применяется формула:

Если m-чётное, p=(m-1)/2



Если m-нечётное, p=m/2



В результате такой процедуры получаются n - m + 1 сглаженных значений уровня ряда.

## **Недостатки метода**

1. Первые и последние уровни ряда теряются (не сглаживаются).
2. Метод применим лишь для рядов, имеющих линейную тенденцию.

Реализация на языке Python:

#Метод простой скользящей средней

import pandas as pd

import numpy as np  
  
m=int(input("Введите значение m:"))  
data=pd.read\_csv("data.csv")  
y=data['X']  
def smooth1(y):  
 x=[]  
 for i in range(0,len(y)):  
 x.append(y[i])  
  
 if m%2==1:  
 p=int((m-1)/2)  
 for i in range(p,len(y)-p):  
 s=0  
 for j in range(i-p,i+p+1):  
 s=s+y[j]  
 s=s/m  
 x[i]=s  
  
 else:  
 p=int(m/2)  
 for i in range(p,len(y)-p):  
 s=y[i-p]/2+y[i+p]/2  
 for j in range(i-p+1,i+p-1+1):  
 s=s+y[j]  
 s=s/m  
 x[i]=s  
 return x  
x=smooth1(y)  
print(x)

Результат:

Введите значение m: 3

[141.1, 147.20000000000002, 159.03333333333333, 176.76666666666665, 170.13333333333333, 164.96666666666667, 146.06666666666666, 151.93333333333334, 155.36666666666667, 166.83333333333334, 166.53333333333333, 163.93333333333337, 169.36666666666665, 176.13333333333333, 181.13333333333333, 178.66666666666666, 169.79999999999998, 163.9, 169.4, 190.53333333333333, 196.20000000000002, 191.83333333333334, 175.76666666666665, 178.4333333333333, 173.03333333333333, 177.96666666666667, 173.13333333333333, 181.16666666666666, 179.1, 164.96666666666667, 162.79999999999998, 147.43333333333334, 167.49999999999997, 157.06666666666666, 171.76666666666665, 158.86666666666667, 161.63333333333333, 156.6, 155.13333333333333, 155.13333333333333, 155.13333333333333, 161.36666666666665, 157.70000000000002, 156.76666666666665, 144.03333333333333, 159.43333333333334, 157.56666666666666, 166.16666666666666, 163.60000000000002, 168.23333333333332, 167.23333333333332, 180.70000000000002, 183.1, 189.0, 171.03333333333333, 180.1, 171.70000000000002, 170.16666666666666, 176.6, 182.36666666666665, 186.80000000000004, 176.4, 174.0666666666667, 174.46666666666667, 167.03333333333333, 176.4, 161.70000000000002, 164.33333333333334, 168.4, 177.9333333333333, 187.20000000000002, 199.1]

**3.2. Метод экспоненциального сглаживани**я

Важным методом стохастических прогнозов является метод экспоненциального сглаживания. Этот метод заключается в том, что ряд динамики сглаживается с помощью скользящей средней, в которой веса подчиняются экспоненциальному закону.

Эту среднюю называют экспоненциальной средней и обозначают St.

Она является характеристикой последних значений ряда динамики, которым присваивается наибольший вес.

Экспоненциальная средняя вычисляется по рекуррентной формуле:

St = α\*Yt + (1- α)St-1

где St - значение экспоненциальной средней в момент t;

St-1 - значение экспоненциальной средней в момент (t = 1);

Что касается начального параметра S0, то в задачах его берут или равным значению первого уровня ряда у1, или равным средней арифметической нескольких первых членов ряда

Yt - значение экспоненциального процесса в момент t;

α - вес t-ого значения ряда динамики (или параметр сглаживания).

Последовательное применение формулы дает возможность вычислить экспоненциальную среднюю через значения всех уровней данного ряда динамики.

Реализация метода на языке Python:

#Метод экспоненциального сглаживания

import numpy as np  
import pandas as pd  
  
data=pd.read\_csv("data.csv")  
y=data['X']  
def smooth2(y):  
 x=[]  
 for i in range(0,len(y)+1):  
 x.append("")  
 x[0]=y[0] #можно использовать среднее выбоки y вместо y[0]  
 alpha=2/(len(y)+1)  
 print("alpha: "+str(alpha))  
 for i in range(0,len(y)):  
 x[i+1]=alpha\*y[i]+(1-alpha)\*x[i]  
 return x  
x=smooth2(y)  
forecast=x.pop(len(x)-1)  
print(x)  
print("Прогноз: "+str(forecast))

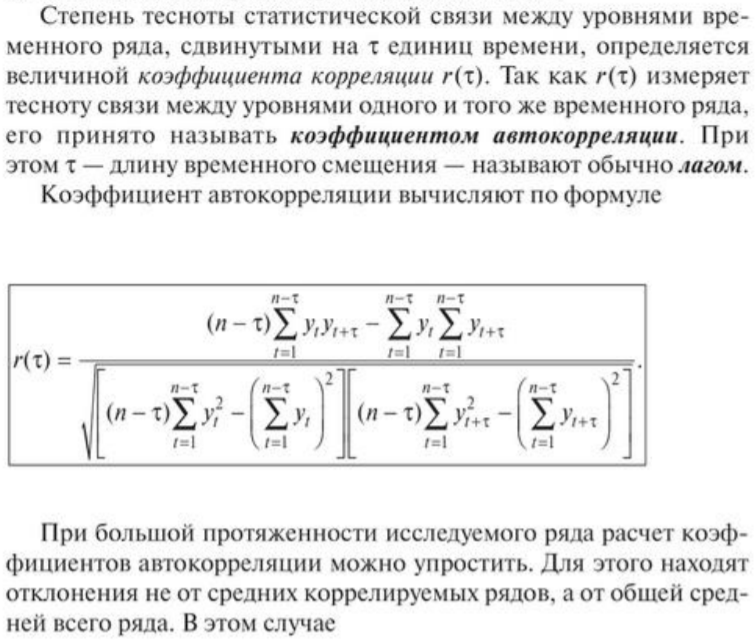
Результат:

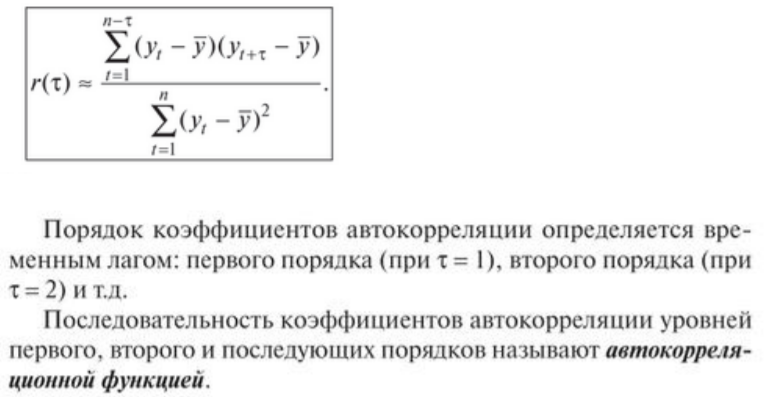
alpha: 0.0273972602739726

[141.1, 141.1, 141.06986301369864, 141.60219553387125, 142.56103949184737, 143.94840827289264, 143.85667105993667, 144.32909103089733, 144.11459538621523, 144.5004968824833, 145.23746957063443, 145.9405525960965, 146.25177033318977, 146.72706429666403, 147.83591185018008, 148.65136632003816, 149.47187683181792, 150.30278431587772, 150.3218861154427, 150.61169745474564, 151.863431771054, 153.57676240746346, 154.24315247849185, 155.03649076675237, 155.34234033478654, 156.17953648999787, 156.47598754506643, 157.1478782972564, 157.5383473850028, 158.45784471692053, 158.90557499864875, 158.08624417676796, 158.8126484458976, 157.9876443788867, 158.84277740960212, 158.69092049427056, 159.0774706177152, 158.84247142270928, 158.91802015085423, 158.88464973576234, 158.53438535944008, 158.6183748016472, 158.5932138481774, 158.7632627838438, 158.53961174866998, 158.43989635829547, 157.56757043067094, 158.64243151476217, 158.38647448696045, 158.2224614873177, 159.0684214465693, 159.18161537953998, 158.91362591708685, 160.8474991796324, 161.10921153087534, 161.27334272181025, 161.66585388011683, 162.65035103408624, 162.08184826602908, 162.32618119024747, 163.8213543083229, 163.69474186151953, 164.25926948175186, 164.84668675622441, 164.50020218756075, 165.07553911392895, 165.02963393272543, 165.4480001263494, 164.78915080781925, 164.967530237742, 165.72184447780387, 165.83905421813802]

Прогноз: 166.75031300668218

**4.Автокорреляция**





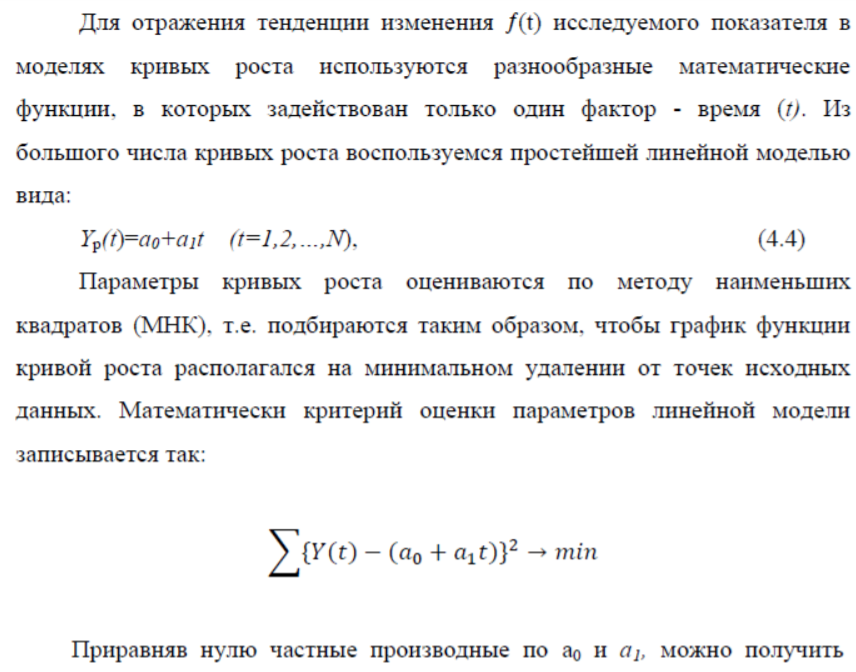
Реализация нахождения автокорреляционной функции на языке Python:

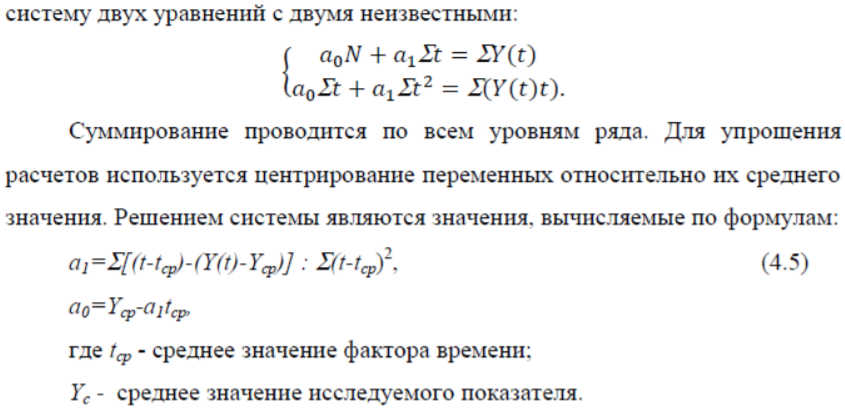
import pandas as pd  
import numpy as np  
  
data=pd.read\_csv("data.csv")  
y=data['X']  
  
def mean1(y):  
 n=len(y)  
 y1=[]  
 for i in range(1,n):  
 a=0  
 for j in range (i,n):  
 a=a+y[j]  
 a=a/(n-i)  
 y1.append(a)  
 return y1  
  
def mean2(y):  
 n=len(y)  
 y2=[]  
 for i in range(1,n):  
 b=0  
 for j in range (i,n):  
 b=b+y[j-i]  
 b=b/(n-i)  
 y2.append(b)  
 return y2  
y1=mean1(y)  
y2=mean2(y)  
  
def autocorr\_coef(y,y1,y2):  
 r=[]  
 for i in range(1,len(y1)):  
 sum1=0  
 sum2=0  
 sum3=0  
 for j in range(i,len(y)):  
 sum1=sum1+(y[j]-y1[i-1])\*(y[j-i]-y2[i-1])  
 sum2=sum2+(y[j]-y1[i-1])\*\*2  
 sum3=sum3+(y[j-i]-y2[i-1])\*\*2  
 sum1=float('{:.0f}'.format(sum1))  
 sum2=float('{:.0f}'.format(sum2))  
 sum3=float('{:.0f}'.format(sum3))  
 b=sum2\*sum3  
 x=(np.sqrt(b))  
 r1=sum1/x  
 r1 = float('{:.3f}'.format(r1))  
 r.append(r1)  
 return r  
  
r=autocorr\_coef(y,y1,y2)  
print(r)

Результат:

[-0.088, 0.105, 0.133, 0.052, 0.03, 0.234, 0.231, -0.202, 0.114, 0.099, -0.166, -0.039, 0.026, -0.315, -0.04, -0.118, -0.135, -0.072, -0.154, -0.248, -0.173, -0.139, -0.133, -0.128, 0.014, -0.217, -0.035, -0.039, -0.088, 0.058, 0.142, 0.011, 0.297, -0.019, 0.012, 0.239, 0.134, 0.209, 0.055, 0.365, -0.217, 0.19, 0.233, 0.002, 0.001, 0.16, -0.203, 0.014, 0.174, 0.207, -0.096, -0.088, -0.052, -0.464, 0.238, 0.281, 0.077, 0.132, -0.261, 0.358, -0.154, 0.244, -0.249, -0.589, 0.311, 0.102, 0.69, 0.384, 0.71, -0.778]

**5.Моделирование линейного тренда**

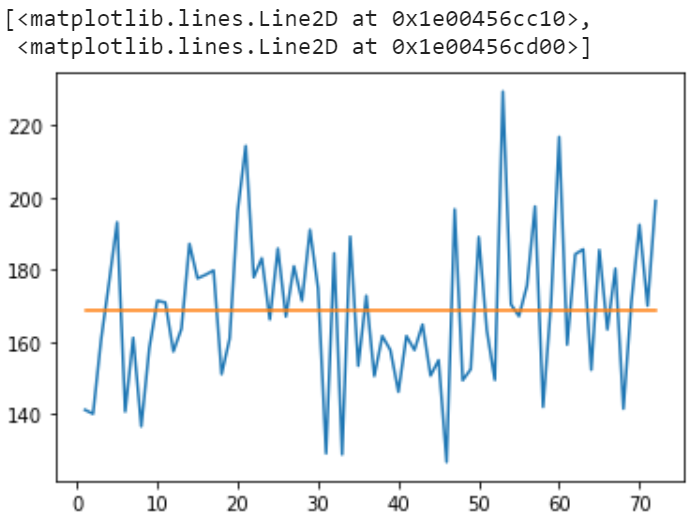




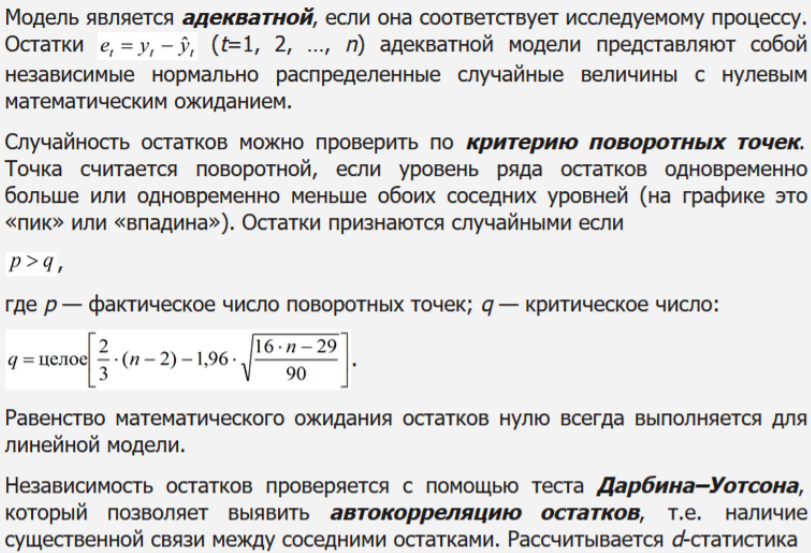
Реализация нахождения линейного тренда на языке Python:

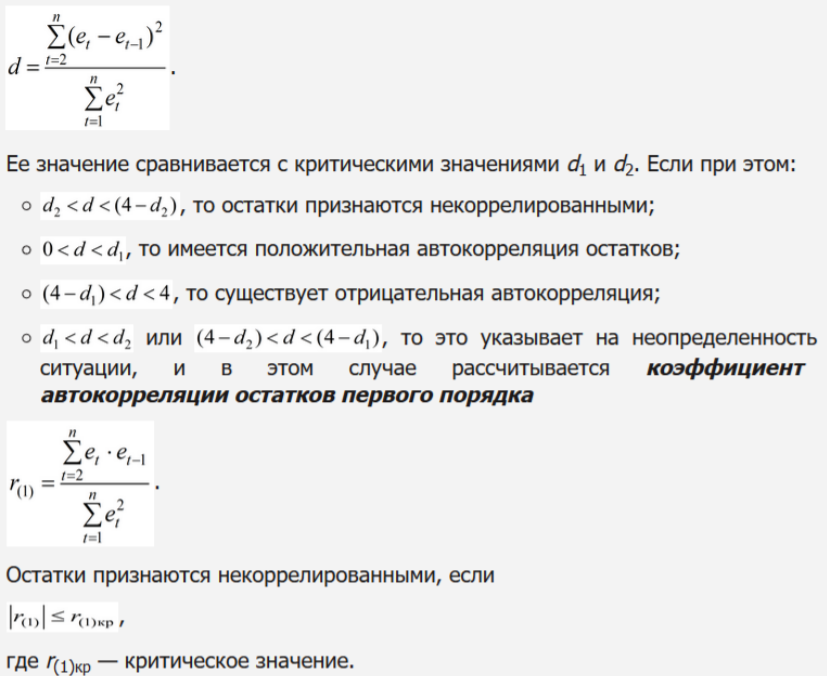
import numpy as np  
  
data = pd.read\_csv("data.csv")  
y = data['X']  
y\_mean = np.mean(y)  
t\_mean = (len(y) - 1) / 2 + 1  
t = []  
for i in range(1, len(y) + 1):  
 t.append(i)  
  
sum1 = 0  
sum2 = 0  
for i in range(0, len(y)):  
 sum1 += ((i + 1 - t\_mean) - (y[i] - y\_mean))  
 sum2 += (i + 1 - t\_mean) \*\* 2  
  
a1 = sum1 / sum2  
a0 = y\_mean - a1 \* t\_mean  
print(a1, a0)  
  
yp = []  
for i in range(0, len(y)):  
 yp.append(a0 + a1 \* t[i])  
  
import matplotlib.pyplot as plt  
  
plt.plot(t, y, t, yp)

Результат:

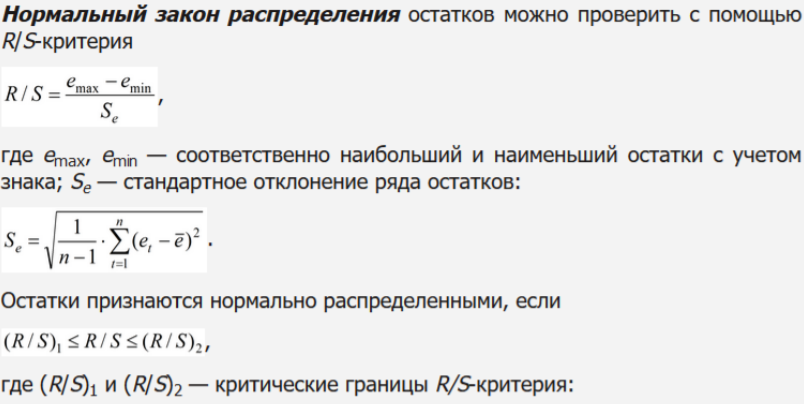


**6.Проверка адекватности модели**











Реализация проверки адекватности модели на языке Python:

import pandas as pd  
import numpy as np  
  
data = pd.read\_csv("data.csv")  
y = data['X']  
y\_mean = np.mean(y)  
t\_mean = len(y) / 2  
t = []  
for i in range(1, len(y) + 1):  
 t.append(i)  
  
sum1 = 0  
sum2 = 0  
for i in range(0, len(y)):  
 sum1 += ((i + 1 - t\_mean) - (y[i] - y\_mean))  
 sum2 += (i + 1 - t\_mean) \*\* 2  
  
a1 = sum1 / sum2  
a0 = y\_mean - a1 \* t\_mean  
print(a1, a0)  
  
y1 = []  
for i in range(0, len(y)):  
 y1.append(a0 + a1 \* t[i])  
  
e = []  
for i in range(0, len(y)):  
 e.append(y[i] - y1[i])  
  
n = len(y)  
q = int(2 \* (n - 2) / 3 - 1.96 \* np.sqrt((16 \* n - 29) / 90))  
p = 0  
for i in range(1, n - 1):  
 if (y[i] > y[i - 1] and y[i] > y[i + 1]) or (y[i] < y[i - 1] and y[i] < y[i + 1]):  
 p += 1  
print("q=", q, "p=", p)  
if p > q:  
 print("Остатки случайны")  
else:  
 print("Остатки неслучайны")  
  
# Статистика Дарбина-Уотсона  
s1 = 0  
s2 = 0  
for i in range(1, n):  
 s1 += (e[i] - e[i - 1]) \*\* 2  
 s2 += (e[i]) \*\* 2  
d = s1 / s2  
print("d=", d)  
  
# Для объёма выборки 70 и alpha=0.05  
d1 = 1.58  
d2 = 1.64  
  
if d > d2 and d < 4 - d2:  
 print("Остатки некоррелированы")  
if d > 0 and d < d1:  
 print("Положительная автокорреляция остатков")  
if d < 4 and d > 4 - d1:  
 print("Отрицательная автокорреляция остатков")  
if d > d1 and d < d2:  
 print("Неопределённость ситуации\nСчитаем r1...")  
 s1 = 0  
 for i in range(1, n):  
 s2 += e[i]  
 r1 = s1 / s2  
 print("r1=", r1)  
 # Для объёма выборки 70 и alpha=0.05  
 r\_critical = 0.235  
 if abs(r1) <= r\_critical:  
 print("Остатки некоррелированы")  
 if r1 > 0:  
 print("Положительная автокорреляция остатков")  
 if r1 < 0:  
 print("Отрицательная автокорреляция остатков")  
# Для объёма выборки 70 и alpha=0.05   
rs\_u = 5.63  
rs\_l = 4.06  
rs = (max(e) - min(e)) / np.std(e)  
print("rs=", rs)  
if rs > rs\_l and rs < rs\_u:  
 print("Остатки имеют нормальное распределение")  
else:  
 print("Остатки не имеют нормальное распределение")

Результат:

0.0011569610489781082 168.8486271800146

q= 39 p= 53

Остатки случайны

d= 2.171375538353481

Остатки некоррелированы

rs= 4.98375130774926

Остатки имеют нормальное распределение

**7.Моделирование сезонных и циклических колебаний**

Существует несколько подходов к анализу структуры временных рядов, содержащих сезонные или циклические колебания. Простейший подход - расчет значений сезонной компоненты методом скользящей средней и построение аддитивной или мультипликативной модели временного ряда. Общий вид аддитивной модели следующий:



Эта модель предполагает, что каждый уровень временного ряда может быть представлен как сумма трендовой , сезонной  и случайной  компонент. Общий вид мультипликативный модели выглядит так:



Эта модель предполагает, что каждый уровень временного ряда может быть представлен как произведение трендовой , сезонной  и случайной  компонент. Выбор одной из двух моделей осуществляется на основе анализа структуры сезонных колебаний. Если амплитуда колебаний приблизительно постоянна, строят аддитивную модель временного ряда, в которой значения сезонной компоненты предполагаются постоянными для различных циклов. Если амплитуда сезонных колебаний возрастает или уменьшается, строят мультипликативную модель временного ряда, которая ставит уровни ряда в зависимость от значений сезонной компоненты.

Построение аддитивной и мультипликативной моделей сводится к расчету значений  и  для каждого уровня ряда.

Процесс построения модели включает в себя следующие шаги.

1. Выравнивание исходного ряда методом скользящей средней.

2. Расчет значений сезонной компоненты .

3. Устранение сезонной компоненты из исходных уровней ряда и получение выравненных данных  в аддитивной или  в мультипликативной модели.

4. Аналитическое выравнивание уровней  или  и расчет значений  с использованием полученного уравнения тренда.

5. Расчет полученных по модели значений  или .

6. Расчет абсолютных и/или относительных ошибок.

Если полученные значения ошибок не содержат автокорреляции, ими можно заменить исходные уровни ряда и в дальнейшем использовать временной ряд ошибок  для анализа взаимосвязи исходного ряда и других временных рядов.

Реализация аддитивной модели на языке Python

