**ПРИЛОЖЕНИЯ**

**Приложение 1. Операторы и выражения языка VBA**

Dim – создание переменных или объектов различных типов.

Dim aa As Range – создание массива-диапазона ячеек aa

(объекта типа Range).

Set aa = Range("S30") – привязка аа к ячейкам; верхняя левая S30.

nn = Range("N10") – считывание переменной nn из ячейки N10 таблицы.

Эти три оператора чрезвычайно полезны: они позволяют связать ячейки Excel с массивами Basic и использовать обычные программы на Basic.

For N = 1 To nn – начало цикла: N изменяется от 1 до nn с шагом 1;

далее идёт тело цикла до Next N.

Rnd() – генератор случайных чисел, равномерно распределённых

в диапазоне 0 … 1.

gg(k, 2) – элемент массива gg, расположенный в строке k и столбце 2.

If gg(k, 2) >= q Then оператор сравнения gg(k, 2) и переменной q;

если соблюдается условие gg(k, 2) >= q,

срабатывают следующие операторы до End If.

Exit For – принудительный выход из цикла.

s =( gg(k) + gg(k-1))/2 – арифметическое выражение: вычисление среднего

значения двух соседних элементов массива.

Программный модуль (Private Sub) обычно запускается при наступлении события объекта, например, щелчок (Click) по кнопке CommandButton1.

Этапы создания кнопки и программного модуля:

1. Войдите в Разработчик Главного меню. Если Разработчика в меню нет, добавьте его: Файл – Параметры – Настройка ленты – флажок на Разработчик.
2. Войдите в Инструменты: щелчок по
3. Инициируйте создание кнопки: левый верхний элемент Active X.
4. Нажмите левую клавишу мыши и растяните кнопку на таблице.
5. Быстро щёлкните дважды по изображению кнопки (Double Click), и войдёте в режим программирования.
6. Напишите программный код, или скопируйте его с текстового документа.
7. Перейдите в таблицу. Сохраните файл как *Книга Excel с поддержкой макросов (xslm).*
8. Выйдите из режима конструктора: щелчок по
9. Щёлкните по кнопке.

Для тренировки создайте кнопку с программным модулем для сложения 6 элементов двух столбцов чисел: В4:В9 и С4:С9.

Private Sub CommandButton21\_Click()

Dim aa As Range

Set aa = Range("B4")

For i = 1 To 6 : aa(i, 4) = aa(i) + aa(i, 2) : Next i

End Sub

Часть таблицы с кнопкой показана на рисунке

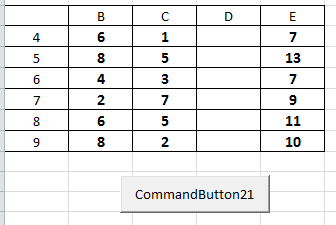


Рис. П1.1. Часть таблицы с кнопкой

**Приложение 2. Использование программ Python в среде Colab и Jupiter Notebook**

Для анализа данных на языке Python можно использовать такие среды разработки как Google Colab и Jupiter Notebook, сравним их. Google Colab предоставляет бесплатные вычислительные ресурсы и интегрированную среду разработки на платформе Google Cloud, в то время как Jupiter требует установки и настройки локально или на сервере. Таким образом, можно использовать любую из программ.

 Можно зайти на сайт <https://colab.research.google.com/?hl=ru>  со своего аккаунта Google (если Вы не вошли в него  на компьютере, то после открытия ссылки автоматически попросит  войти), всплывет окно и предложит открыть блокнот или создать новый.  
  Еще способ - зайти на Google диск и создать новый блокнот (на фото), это даже удобнее, так как можно создать новый блокнот в необходимо папке. Алгоритм создания нового файла в Google Colab:  
1) Зайти на Google Диск, используя свой аккаунт  
2) Выбрать или создать необходимую папку, в которой будет находиться блокнот  
3) Нажать кнопку "Создать" в левой части экрана, далее "Еще" и выбрать  
"Google Colaboratory"  
Для того, чтобы открыть существующий файл формата .ipynb, который  
находится на вашем компьютере, необходимо загрузить данный файл в  
папку на Google Диске. Далее нажать на него два раза, он автоматически  
откроется в Google Colab.

Перед началом работы установим с помощью pip необходимые библиотеки и модули:

!pip install numpy

!pip install pandas

!pip install matplotlib

!pip install sklearn

!pip install statsmodels

!pip install scipy

!pip install

!pip install gekko

Основные объекты, которые будут использоваться в работах: list, numpy.array, pandas.DataFrame. Про них подробнее можно узнать из документации Python.

Важный момент работы с данными – их считывание с устройства. Рассмотрим способы получения таблиц данных в зависимости от формата, для этого будем использовать pandas.

Для получения массива данных из csv-файла используется команда pd.read\_csv(“Путь к файлу.csv”), из excel-файла – pd.read\_excel(“Путь к файлу.xlsx”). Необходимо отметить, что если файл находится в одной папке со скриптом, то достаточно указать только название. В Google Colab для работы с файлами необходимо проделать следующие действия:

1. Написать в ячейке команду и запустить ее:

from google.colab import drive

drive.mount(«/content/drive»)

1. Далее в окне нажать «Подключиться к Google Диску»
2. Выбрать аккаунт и нажать кнопку «Разрешить»

Теперь можно работать с файлами, которые есть на диске. Еще один способ – поместить файлы в папку sample\_data, но после перезагрузки страницы файлы удалятся.

С notebook можно работать как с командной строкой, выполняя различные действия в ячейках. Например, с помощью команды git clone можно скопировать все файлы из репозитория GitHub, указав ссылку на него. Так, чтобы скачать все файлы формата ipunb, содержащие коды, и исходные данные для изучения материала, необходимо написать в ячейке Jupiter Notebook или Google Colab команду !git clone https://github.com/luckykittty/Analysis\_of\_Time\_Series\_2024 .

**Приложение 3. Программы на VBA и Python для обнаружения превышений над фоном**

**Программный код на VBA**

Private Sub CommandButton1\_Click()

Range("K5") = 0

Image1.Visible = False ‘ Рисунок, появление которого на экране сигнализирует об эффекте

Dim a, b As Range

Set a = Range("D6") ‘Data

Set b = Range("S15") ‘Results

N1 = Range("S5") : N2 = Range("T5") ‘ Count of events

N3 = Range("U5") : Xcp = 0

t = Range("K3") ; For k = t - 20 To t – 6 : Xcp = Xcp + a(k) : Next k

Xcp = Xcp / 15 ‘ Средний фон

S = Xcp ^ 0.5 ‘ СКО фона

k = t

If a(k) - Xcp > 4 \* S Then

Image1.Visible = True ‘ ALARM !!! Сигнал при обнаружении эффекта

Range("K5") = (a(k) - Xcp) / S \* 100: b(N1) = Range("K5") : Range("S5") = Range("S5") + 1

End If

If a(k) - Xcp > 2 \* S Then

If a(k - 1) - Xcp > 2 \* S Then

Image1.Visible = True

Range("K5") = (a(k) - Xcp) / S \* 100: b(N2, 2) = Range("K5"): Range("T5") = Range("T5") + 1

Else

If a(k - 2) - Xcp > 2 \* S Then

Image1.Visible = True

Range("K5") = (a(k) - Xcp) / S \* 100 : b(N2, 2) = Range("K5"): Range("T5") = Range("T5") + 1

End If : End If : End If

Range("K3") = Range("K3") + 1

End Sub

**Программный код на Python. В случае необходимости установить библиотеки c помощью pip.**

#Генерируем ряд с учетом эффектов:

#1) среднее отклонение от фона А(25) = 100, СКО = 5;

#2) А(35) = А(36) = А(37) = 30; СКО =5.

#3) А(50) = А(53) = А(54) = 20; СКО =5

n = 193 ; X = [] ; s = 10 ; mean = 100

mean\_a = [50, 30, 20]

s\_a = 3

for i in range(n):

if i == 24:

X.append(s\*sps.norm.ppf(np.random.random())+mean+mean\_a[0]+s\_a\*sps.norm.ppf(np.random.random()))

elif i in [34, 36, 37]:

X.append(s\*sps.norm.ppf(np.random.random())+mean+mean\_a[1]+s\_a\*sps.norm.ppf(np.random.random()))

elif i == 49:

X.append(s\*sps.norm.ppf(np.random.random())+mean+mean\_a[2]+s\_a\*sps.norm.ppf(np.random.random()))

else:

X.append(s\*sps.norm.ppf(np.random.random())+mean)

#Сигнал и вывод групп с аномалиями

for i in range(0, len(X)):

if X[i] - mean>4\*s:

print('ALARM !!!')

print(i, X[i], '\n')

if i-3<=len(X) and i-3>=0:

c = 0

for j in range(i-3, i):

if X[j] - mean>2\*s:

c += 1

if c >= 2:

print('ALARM !!!')

print(X[i-3:i], '\n')

if i-5<=len(X) and i-5>=0:

c = 0

for j in range(i-5, i):

if X[j] - mean>1.5\*s:

c += 1

if c >= 3:

print('ALARM !!!')

print(X[i-5:i], '\n')

**Приложение 4. Построение парной регрессии на языке Python. Аналог сервиса Регрессия.**

Импортируем библиотеки и их модули, которые будут использоваться в ходе решения задачи:

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import scipy.stats as sps

import statsmodels.api as sm

import statsmodels.stats.api as sms

import statsmodels.stats

#Для удобства данные из Таблицы 4.1 представим в виде pandas.DataFrame:

data = pd.DataFrame({'X': np.arange(21), 'Y': [11, 8, 9, 13, 6, 10, 11, 6, 7, 13, 12, 15, 18, 16, 24, 22, 27,28, 25, 32, 28]})

#Вычислим корреляцию величин:

data.corr()

По данным Таблицы 4.1 построим Модель 1 линейной регрессии для X от 10 до 21, используя метод OLS. С помощью методы summary() можно увидеть все характеристики полученной модели:

Y = data['Y'][10:]

X = data.drop('Y', axis = 1)[10:]

X2 = sm.add\_constant(X)

regression = sm.OLS(Y,X2).fit()

regression.summary()

#Построим график, сделав прогноз на 10 периодов вперед:

X\_pred = np.arange(11,31)

plt.scatter(data['X'][10:], data['Y'][10:], label = 'Y')

plt.plot(X\_pred, regression.predict(sm.add\_constant(X\_pred)), 'o-', c = 'r', label = f'Y^={regression.params["const"].round(4)}+{regression.params["X"].round(4)}\*X')

plt.xlabel('Время')

plt.ylabel('Продажи')

plt.title('Линейная модель от 11 до 21 градусa')

plt.legend()

plt.show()

Необходимо заметить, что табличные значения для критерия Стьюдента и Фишера можно вычислить на Python:

print('Табличные значения:')

print('t =', sps.t.isf(0.05/2,X.shape[0]-X.shape[1]-1))

print('F =', sps.f(X.shape[1], X.shape[0]-X.shape[1]-1).isf(0.05))

Модель 2:

Построим регрессию по всей выборке с аппроксимацией параболой:

Y = data['Y']

X = data.drop('Y', axis = 1)

X2 = pd.concat([X, X\*\*2], axis = 1)

X2.columns = ['X', 'X^2']

X2 = sm.add\_constant(X2)

regression = sm.OLS(Y,X2).fit()

regression.summary()

Аналогично построим график:

X\_pred = pd.DataFrame({'X': np.arange(31), 'X^2': np.arange(31)\*\*2})

data.plot('X', 'Y', kind = 'scatter', label = 'Y')

plt.plot(np.arange(31), regression.predict(sm.add\_constant(X\_pred)),

c = 'r', label = f'Y^ = {regression.params["const"].round(4)} + {regression.params["X"].round(4)}\*X + {regression.params["X^2"].round(4)}\*X^2')

plt.xlabel('Время')

plt.ylabel('Продажи')

plt.title('Параболическая модель')

plt.legend()

plt.show()

Далее можно увеличить степень полинома, как было сделано в Excel на Рис.4.3. Оставим 4 наблюдения в качестве контрольной выборки:

Y = data['Y'][:-4]

X = data.drop('Y', axis = 1)[:-4]

X2 = pd.concat([X, X\*\*2, X\*\*3], axis = 1) #полином степени 3

X2.columns = ['X', 'X^2', 'X^3']

X2 = sm.add\_constant(X2)

regression = sm.OLS(Y,X2).fit()

Для полинома четвертой степени:

Y = data['Y'][:-4]

X = data.drop('Y', axis = 1)[:-4]

X2 = pd.concat([X, X\*\*2, X\*\*3, X\*\*4], axis = 1) #полином степени 4

X2.columns = ['X', 'X^2', 'X^3', 'X^4']

X2 = sm.add\_constant(X2)

regression = sm.OLS(Y,X2).fit()

Можно также вызвать метод summary(), чтобы посмотреть результат.

**Приложение 5. Построение нелинейной регрессии на языке Python. Аналог сервиса Поиск решения.**

Сделаем импорт необходимых библиотек:

from gekko import GEKKO

import sklearn.metrics

from statsmodels.stats.stattools import durbin\_watson

import sklearn.metrics

import pandas as pd

import numpy as np

#Линейная аппроксимация по данным Таблицы 4.2. Начальные параметры

df= pd.DataFrame({'X': np.arange(0,21), 'Y': [11, 8, 9, 13, 6, 10, 11, 6, 7, 13, 12, 15, 18, 16, 24, 22, 27,28, 25, 32, 28]})

t = df['X'][10:].values

y = df['Y'][10:].values

n = 2 #кол-во переменных

m = GEKKO(remote=False)

m.options.SOLVER=1

#Изменяемые значения

ar = m.Array(m.Var, n, value = 0)

#Целевая функция = sum(e^2) -> min

obj = 0

for i in range(len(t)):

obj += (y[i]-(ar[0]+ar[1]\*t[i]))\*\*2

m.Minimize(obj) #целевая фунция

m.solve()

res = np.array([ar[0].value[0], ar[1].value[0]])

print(res) #полученные коэффициенты

y\_pred = res[0]+res[1]\*t

e = y-y\_pred

R\_2 = sklearn.metrics.r2\_score(y, y\_pred)

print(R\_2)

# Модель 2: Аппроксимация параболой по всем данным

#Начальные параметры

t = df['X'].values

y = df['Y'].values

n = 3 #кол-во переменных

m = GEKKO(remote=False) # create model

m.options.SOLVER=1

#Изменяемые значения

ar = m.Array(m.Var, n, value = 0)

#Целевая функция = sum(e^2) -> min

obj = 0

for i in range(len(t)):

obj += (y[i]-(ar[0]+ar[1]\*t[i]+ar[2]\*t[i]\*\*2))\*\*2

m.Minimize(obj) #целевая фунция

m.solve()

res = np.array([ar[0].value[0], ar[1].value[0], ar[2].value[0]])

print(res)

y\_pred = res[0]+res[1]\*t+res[2]\*t\*\*2

e = y-y\_pred

R\_2 = sklearn.metrics.r2\_score(y, y\_pred)

print(R\_2)

# Статистику Дарбина-Уотсона для моделей можно вычислить следующим образом:

from statsmodels.stats.stattools import durbin\_watson

dw = durbin\_watson(e)

# Статистика Голфелда-Квандта:

import statsmodels.stats.api as sms

X=np.column\_stack((np.ones(len(t)),t))

#для модели 2:

X=np.column\_stack((np.ones(len(t)), t,t\*\*2))

F\_gq = sms.het\_goldfeldquandt(e, X)

#Первое значение - статистика, второе - p-value

F\_gq

# Код на Python для настройки синусоиды по данным Табл. 4.8.

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import scipy.stats as sps

import statsmodels.api as sm

import statsmodels.stats.api as sms

import statsmodels.stats

import sklearn.metrics

x = np.array([4, 6, 9, 9, 11, 12, 19, 22, 27, 30, 33, 36, 39])

y = np.array([44, 66, 81, 99, 111, 122, 139, 103, 22, 17])

t = x[:-3] #Тренировочные данные

from gekko import GEKKO

n = 4 #кол-во переменных

m = GEKKO(remote=False) # create model

m.options.SOLVER=1 #solver for nonlinear tasks

#Изменяемые значения

H = m.Var(value = 100)

A = m.Var(value = 50)

W = m.Var(value = 6.28/30 )

f = m.Var(value = -2)

#Целевая функция = sum(e^2) -> min

obj = 0

for i in range(len(t)):

obj += (y[i]-(H+A\*m.sin(t[i]\*W+f)))\*\*2

m.Minimize(obj) #целевая фунция

m.solve()

res = np.array([H.value[0], A.value[0], W.value[0], f.value[0]])

print(res) # print solution

y\_pred = res[0]+res[1]\*np.sin(x\*res[2]+res[3])

e = y-y\_pred[:-3]

R\_2 = sklearn.metrics.r2\_score(y, y\_pred[:-3])

results = pd.DataFrame({'t': t, 'Y': y, 'Y^': y\_pred[:-3], 'e': e, 'e^2': e\*\*2})

#Прогноз для последних трех значений

print(x[-3:], y\_pred[-3:])

results #Результат

**Приложение 6. Использование метода Монте-Карло для исследования ошибок регрессионной модели.**

Создать на рабочем листе кнопку с программным модулем на языке Visual Basic for Applications (VBA) для сохранения получаемых параметров модели и создания псевдореального массива *Yимит*, значения которого будут распределены вокруг “идеальных” значений *Y* по закону нормального распределения (Гаусса) со стандартным отклонением *3СигмаОст.идеал*. В программном модуле используется генератор случайных чисел RND(), который выдает случайные числа, равномерно распределенные в диапазоне 0 – 1. Программа пересчитывает их в числа, распределенные по нормальному (Гауссову) закону в диапазоне от –3 до +3, используя заранее внесённую в одну из ячеек Excel функцию НОРМСТОБР. В данном примере функция внесена в ячейку N3, её аргумент формируется в предыдущей ячейке М3 функцией Бейсика Rnd(). Процедура пересчета представлена на рисунке. Случайное значение *Y* складывается из *Yидеал* и случайной величины, распределенной по нормальному закону в диапазоне от *–3СигмаОст.идеал* до *+3СигмаОст.идеал*. Технология создания на рабочем листе кнопки с программным модулем в среде Excel 2003:

* в Меню войти в Файл – Параметры, включить в Настройку ленты Разработчик, щёлкните Кнопка на панели Active X



* при нажатой левой клавиши мыши растяните на рабочем листе контур кнопки;
* после появления кнопки дважды быстро щелкните по ней мышью, и вы войдете в окно программного модуля;
* впишите в программный модуль текст программы, контролируя соответствие адресов переменных на рабочем листе и параметров объектов типа “Массив-диапазон ячеек” (Range);
* перейдите из режима конструктора в рабочий режим, щелкнув по .



Cохранить рабочую книгу как файл Excel c поддержкой модулей VBA.

Для расчёта коэффициентов, *R2* и *F* следует применять функцию ЛИНЕЙН, которая автоматически срабатывает при каждой новой имитации. Все вычисленные параметры модели следует разместить в одну строку, используя копирование по формуле: так проще запрограммировать их сохранение. Целесообразно создать две кнопки: вторая для многократного вызова программного модуля главной кнопки.

Метод Монте-Карло можно применять только с использованием программных модулей, но его принципы можно усвоить в Excel в ручном режиме. Для этого надо заменить нормальное распределение возмущений в диапазоне от *–3СигмаОст.идеал* до *+3СигмаОст.идеал* на равномерное распределение в диапазоне от *–1,5СигмаОст.идеал* до *+1,5СигмаОст.идеал*, как показано на рисунке П1.2. и реализуется по формуле

*Yимит = Yидеал* +НОРМ.СТ.ОБР(СЛЧИС())\**СигмаОст.идеал*

Функция НОРМ.СТ.ОБР(СЛЧИС()) преобразует равномерное распределение в интервале 0…1 в гауссово распределение с m=0 ϭ=1.

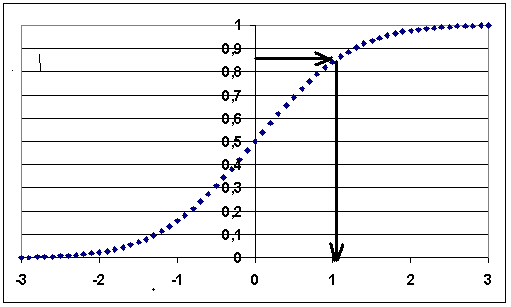


Рис.П4.1.Преобразование равномерного случайного распределения в нормальное

4. Программный модуль:

Private Sub CommandButton1\_Click()

Dim A, В, Gauss, Y As Range ‘Создание объектов типа массив-диапазон

‘ Настройка массивов-диапазонов (Range) на

соответствующие ячейки рабочего листа:

Set A = Range("H4") ‘ a, b, Yпрогноз, GQ, DW и др.

Set В = Range("H20") ‘Подготовка сохранения результатов

‘начиная с ячейки Н20

Set Gauss = Range("M3") ‘ в ячейку N3 внести функцию

НОРМСТОБР(“M3”)

Set Y = Range("C4") ‘Yideal с ячейки С4, Yимит с D4

Randomize

For i = 1 To 20 ‘ Создание массива из 20 чисел *Yидеал*+RND

Gauss(1)= Rnd()

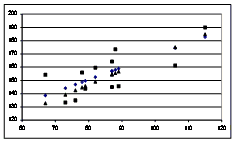
M=Gauss(1,2)

Sigma = Range("G19") ‘σ остатков в "G19"

Y(i,2) = Y(i) + M \* Sigma ‘ Yимит

Next i

N=Range(“N1”) ‘ счётчик строк массива сохраняемых данных

For i=1 to 5

B(N,i) = A(N,1) ‘ сохранение *a, b, Yпрогноз, DW, GQ*

Next i

Range(“N1”) = Range(“N1”) + 1

End Sub

5. Выделите столбцы *X, Yideal,* Yrandom*, Y^.* Постройте диаграмму типа *Точечная*. На рисунке ромбики *Yideal,* квадраты *Yимит*, треугольники *Y^*.

6. Последовательно нажимайте кнопку и cледите за изменениями на диаграмме и за накоплением результатов.

Для нажатия кнопки CommandButton1 10000 раз создать вторую кнопку с программным модулем

For k= 1 to 10000 : Call CommandButton1\_Click() : Next k

**Оценка погрешностей параметров эконометрической модели в R**

# Monte Carlo 1

k1=c() ; k2=c() ; y3=c()

sdvig = rep(0, 20 ) ; v =rep(4, 20)

sdvig ; v

for(n in 1: 10) {

x=seq(1,20) ; y=5+x

y

s=rnorm(20, 0, 1) ; ys =y+s \*v + sdvig ;

ys

plot(ys, type='p',col='red', lwd=5)

lines(y, type='l', lwd=5, col='blue')

coef = c( lm(ys ~ x)$coefficients)

coef

yf = c( lm(ys ~ x)$fitted.values)

lines(yf, type='l', lwd=5, col='black')

k1=cbind(k1,coef[1]) ; k2=cbind(k2,coef[2])

y3=cbind(y3,coef[1]+coef[2]\*20)

k1 ; k2 ; y3

}

y3

k1=as(k1, "vector") ; k2=as(k2, "vector") ; y4=as(y3, "vector")

k1 ; k2 ; y4

cat('mean(k1)= ', mean(k1), 'mean(k2) = ', mean(k2), 'mean(k4) = ', mean(y4))

cat('sd(k1) = ', sd(k1), 'sd(k2) = ', sd(k2), 'sd(k4) = ', sd(y4))

cat(' % sd(k1) = ', sd(k1)/mean(k1)\*100, ' % sd(k2) = ', sd(k2)/mean(k2)\*100, ' % sd(k4) = ', sd(y4)/mean(y4)\*100 )

cat('cor(k1,k2) = ', cor(k1,k2))

# Monte Carlo 2

k1=c() ; k2=c() ; y3=c()

sdvig = rep(0, 20 ) ; v =rep(4, 20)

# v = c(2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6)

# v = c(6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2)

# v = c(3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 10, 10, 9, 3, 3, 3)

# v= c(3, 3, 10, 10, 9, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3)

# sdvig = c(4, -4, -4, 4, 4, -4, -4, 4, 4, -4, -4, 4, 4, -4, -4, 4, 4, -4, -4, 4)

# sdvig = c(4, 4, 4, -4, -4, -4, -4, -4, 4, 4, 4, 4, 4, -4, -4, -4, -4, -4, 4, 4)

# sdvig = c(4, -4, 4, -4, 4, -4, 4, -4, 4, -4, 4, -4, 4, -4, 4, -4, 4, -4, 4, -4)

for(n in 1: 10000) {

x=seq(1,20) ; y= 5+x ;

s=rnorm(20, 0, 1) ; ys =y+s \*v + sdvig ;

coef = c( lm(ys ~ x)$coefficients)

yf = c( lm(ys ~ x)$fitted.values)

k1=cbind(k1,coef[1]) ; k2=cbind(k2,coef[2])

y3=cbind(y3,coef[1]+coef[2]\*20)

}

k1=as(k1, "vector") ; k2=as(k2, "vector") ; y4=as(y3, "vector")

cat('mean(k1)= ', mean(k1), 'mean(k2) = ', mean(k2), 'mean(k4) = ', mean(y4))

cat('sd(k1) = ', sd(k1), 'sd(k2) = ', sd(k2), 'sd(k4) = ', sd(y4))

cat(' % sd(k1) = ', sd(k1)/mean(k1)\*100, ' % sd(k2) = ', sd(k2)/mean(k2)\*100, ' % sd(k4) = ', sd(y4)/mean(y4)\*100 )

cat('cor(k1,k2) = ', cor(k1,k2))

**Код на Python, представленный для случая v = 4:**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import scipy.stats as sps

import statsmodels.api as sm

import statsmodels.stats.api as sms

import statsmodels.stats

from statsmodels.stats.stattools import durbin\_watson

import statsmodels.stats.api as sms

#Функция для расчета Y\_imit: v, b – параметры, которые можно изменить

def Y\_imit(X, Yideal, v, n = 20, b = None):

if b == None:

b = [0]\*len(X)

#b - сдвиг

Yimit = []

for i in range(n):

Yimit.append(Yideal[i]+v[i]\*sps.norm.ppf(np.random.random())+b[i])

return Yimit

X = np.arange(1,31) #последовательность 1, 2, ..., 30

Yideal = []

a = 5

for i in range(30):

Yideal.append(X[i] + a) #X+a

a\_coef, b\_coef, pred\_30, r2, F, dw, GQ = [], [], [], [], [], [], []

v = [4]\*20

for i in range(10000):

Yimit = Y\_imit(X, Yideal, v)

df = pd.DataFrame({'X': X[:20],

'Yideal':Yideal[:20],

'Yimit': Yimit[:20],

'v': v})

Y = df['Yimit']

X1 = df['X']

X2 = sm.add\_constant(X1)

regression = sm.OLS(Y,X2).fit()

y\_pred = regression.predict(sm.add\_constant(X))

e = Yimit - y\_pred[:20]

a\_coef.append(regression.params['const'])

b\_coef.append(regression.params['X'])

pred\_30.append(y\_pred[29])

r2.append(regression.rsquared)

F.append(regression.fvalue)

dw.append(durbin\_watson(e))

GQ.append(sms.het\_goldfeldquandt(e, X2.values)[0])

#Результат

pd.DataFrame({'b': [np.mean(b\_coef), np.std(b\_coef)],

'a': [np.mean(a\_coef), np.std(a\_coef)],

'Y(30)^': [np.mean(pred\_30), np.std(pred\_30)],

'R2': [np.mean(r2), np.std(r2)],

'F': [np.mean(F), np.std(F)],

'GQ': [np.mean(GQ), np.std(GQ)],

'DW': [np.mean(dw), np.std(dw)]}, index = ['mean', 'std'])

#Гистограмма DW

plt.hist(dw, bins = np.linspace(np.min(dw), np.max(dw), 10))

**Приложение 7. Прогнозирование продаж бройлеров.**

Исходные данные – в текстовом файле D://RR/Chicks.txt. Первый столбец – время. На всякий случай предусмотрено чтение данных из Буфера обмена (clipboard), здесь заблокировано. Реализована регрессия по исходным данным и по логарифмам. Оценённые значения *Yfit* вычислены непосредственно функцией *lm* и по коэффициентам (*Yfit2*).

f = "D://RR/Chicks.txt"

vv = read.table(f, head=TRUE, sep='', dec=',')

# vv=read.table("clipboard", head=TRUE, dec=',')

vv ; t =vv[,1] ; Y = vv[,6]

coef = lm( Y[1:14] ~ vv[,2][1:14] + vv[,3][1:14]+ vv[,4][1:14] +

vv[,5][1:14] )$coefficients

Yfit = lm( Y[1:14] ~ vv[,2][1:14] + vv[,3][1:14]+ vv[,4][1:14] +

vv[,5][1:14] )$fitted.values

Y ; Yfit

i = c(1,18) ; j=c(30,60) ; plot(i,j, col='white')

lines(Y, col='blue', lwd=5)

lines(Yfit, type='l',col='red', lwd=5)

coef

Yfit2 = coef[1] + coef[2]\*vv[2] + coef[3]\*vv[3] + coef[4]\*vv[4] +coef[5]\*vv[5]

lines(Yfit2, type='l',col='red', lwd=5)

vv2 = log(vv) ; vv

Y2 = vv2[,6]

coef = lm( Y2[1:14] ~ vv2[,2][1:14] + vv2[,3][1:14]+ vv2[,4][1:14] +

vv2[,5][1:14] )$coefficients

Yfit2 = lm( Y2[1:14] ~ vv2[,2][1:14] + vv2[,3][1:14]+ vv2[,4][1:14] +

vv2[,5][1:14] )$fitted.values

Y2 ; Yfit2

i = c(1,18) ; j=c(3.4, 4) ; plot(i,j, col='white')

lines(Y2, col='blue', lwd=5)

lines(Yfit2, type='l',col='red', lwd=5)

coef

Yfit3 = coef[1] + coef[2]\*vv2[2] + coef[3]\*vv2[3] + coef[4]\*vv2[4] +coef[5]\*vv2[5]

lines(Yfit3, type='l',col='red', lwd=5)

Yfit4 = exp(Yfit3)

plot(Y, type='l',col='blue', lwd=5)

lines(Yfit4, type='l',col='red', lwd=5)

**Приложение 8. Исходные данные для настройки**

**макроэкономических моделей.**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Год*** | ***Y*** | ***C*** | ***I*** | ***G*** | ***Год*** | ***Y*** | ***C*** | ***I*** | ***G*** |
| 1946 | 211,10 | 146,90 | 28,70 | 30,90 | 1977 | 1887,20 | 1206,50 | 297,80 | 394,00 |
| 1947 | 233,30 | 165,60 | 30,20 | 28,60 | 1978 | 2249,70 | 1403,50 | 416,80 | 425,20 |
| 1948 | 259,00 | 177,90 | 42,70 | 36,60 | 1979 | 2508,20 | 1566,80 | 454,80 | 467,80 |
| 1949 | 258,20 | 180,60 | 33,50 | 43,60 | 1980 | 2732,00 | 1732,60 | 437,00 | 530,30 |
| 1950 | 286,80 | 194,60 | 52,50 | 42,00 | 1981 | 3052,60 | 1915,10 | 515,50 | 588,10 |
| 1951 | 329,80 | 208,10 | 58,60 | 62,90 | 1982 | 3166,00 | 2050,70 | 447,30 | 641,70 |
| 1952 | 348,00 | 218,10 | 52,50 | 77,50 | 1983 | 3405,70 | 2234,50 | 502,30 | 675,00 |
| 1953 | 365,40 | 232,60 | 50,30 | 82,80 | 1984 | 3765,00 | 2428,20 | 662,10 | 733,40 |
| 1954 | 363,10 | 238,00 | 48,90 | 75,30 | 1985 | 3998,10 | 2600,50 | 662,10 | 815,40 |
| 1955 | 397,50 | 256,90 | 63,80 | 75,60 | 1986 | 4268,60 | 2850,60 | 717,60 | 833,00 |
| 1956 | 419,20 | 269,90 | 67,40 | 79,00 | 1987 | 4539,90 | 3052,20 | 749,30 | 881,50 |
| 1957 | 441,10 | 281,40 | 67,80 | 86,10 | 1988 | 4900,40 | 3296,10 | 793,60 | 918,70 |
| 1958 | 447,30 | 290,10 | 60,90 | 94,20 | 1989 | 5250,80 | 3523,10 | 832,30 | 975,20 |
| 1959 | 483,70 | 311,20 | 75,30 | 97,00 | 1990 | 5546,10 | 3761,20 | 808,90 | 1047,40 |
| 1960 | 503,70 | 325,20 | 74,80 | 99,60 | 1991 | 5724,80 | 3902,40 | 744,80 | 1097,40 |
| 1961 | 520,10 | 335,20 | 71,70 | 107,60 | 1992 | 6020,20 | 4136,90 | 788,30 | 1125,30 |
| 1962 | 560,30 | 355,10 | 83,00 | 117,10 | 1993 | 6657,40 | 4477,90 | 953,40 | 1291,20 |
| 1963 | 590,50 | 375,00 | 87,10 | 122,50 | 1994 | 7072,20 | 4743,30 | 1097,10 | 1325,50 |
| 1964 | 632,40 | 401,20 | 94,00 | 128,70 | 1995 | 7397,70 | 4975,80 | 1144,00 | 1369,20 |
| 1965 | 684,90 | 432,80 | 108,10 | 137,00 | 1996 | 7816,90 | 5256,80 | 1240,30 | 1416,00 |
| 1966 | 747,60 | 465,50 | 120,80 | 156,20 | 1997 | 8304,30 | 5547,40 | 1389,80 | 1468,70 |
| 1967 | 796,30 | 490,40 | 120,80 | 180,20 | 1998 | 8747,00 | 5879,50 | 1509,10 | 1518,30 |
| 1968 | 868,50 | 535,90 | 131,50 | 198,70 | 1999 | 9268,40 | 6282,50 | 1625,70 | 1620,80 |
| 1969 | 935,50 | 579,70 | 146,20 | 207,90 | 2000 | 9817,00 | 6739,40 | 1735,50 | 1721,60 |
| 1970 | 982,40 | 618,80 | 140,80 | 218,90 | 2001 | 10128,00 | 7045,40 | 1614,30 | 1825,60 |
| 1971 | 1063,40 | 668,20 | 160,00 | 233,70 | 2002 | 10469,60 | 7385,30 | 1582,10 | 1961,10 |
| 1972 | 1171,10 | 733,00 | 188,30 | 253,10 | 2003 | 10960,80 | 7703,60 | 1664,10 | 2092,50 |
| 1973 | 1306,60 | 809,90 | 220,00 | 269,50 | 2004 | 11685,90 | 8195,90 | 1888,60 | 2216,80 |
| 1974 | 1412,90 | 889,60 | 214,60 | 302,70 | 2005 | 12421,90 | 8694,10 | 2086,10 | 2355,30 |
| 1975 | 1528,80 | 979,10 | 190,90 | 338,40 | 2006 | 13178,40 | 9207,20 | 2220,40 | 2508,10 |
| 1976 | 1605,50 | 1030,60 | 179,10 | 368,20 | 2007 | 13807,50 | 9710,20 | 2130,40 | 2674,80 |

**Настройка макроэкономической модели с двумя эндогенными переменными МНК с Поиском решения**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| t | Year | Qrt | X1 | X2 | X3 | Y1 | Y2 | Ŷ1 | Ŷ2 | e1 | e2 |
| 1 | 2003 | 1 | 11639 | 1810 | 1091 | 1676 | 7646 | 1676 | 7646 | 0 | 0 |
| 2 | 2003 | 2 | 11738 | 1822 | 1087 | 1700 | 7727 | 1665 | 7874 | 34 | -147 |
| 3 | 2003 | 3 | 11931 | 1888 | 1117 | 1720 | 7882 | 1761 | 7942 | -42 | -60 |
| 4 | 2003 | 4 | 12039 | 1960 | 1169 | 1784 | 7961 | 1816 | 8069 | -32 | -108 |
| 5 | 2004 | 1 | 12118 | 1970 | 1197 | 1829 | 8098 | 1841 | 8238 | -11 | -140 |
| 6 | 2004 | 2 | 12196 | 2056 | 1216 | 1903 | 8194 | 1872 | 8288 | 30 | -94 |
| 7 | 2004 | 3 | 12287 | 2082 | 1225 | 1930 | 8317 | 1907 | 8399 | 23 | -83 |
| 8 | 2004 | 4 | 12387 | 2125 | 1253 | 1979 | 8474 | 1941 | 8489 | 38 | -15 |
| 9 | 2005 | 1 | 12515 | 2170 | 1276 | 1991 | 8592 | 2001 | 8592 | -10 | 0 |
| 10 | 2005 | 2 | 12571 | 2131 | 1304 | 2013 | 8730 | 2010 | 8724 | 3 | 6 |
| 11 | 2005 | 3 | 12671 | 2155 | 1304 | 2025 | 8896 | 2041 | 8737 | -16 | 159 |
| 12 | 2005 | 4 | 12736 | 2233 | 1337 | 2082 | 8996 | 2064 | 8789 | 18 | 208 |
| 13 | 2006 | 1 | 12896 | 2266 | 1389 | 2122 | 9126 | 2143 | 9000 | -22 | 126 |
| 14 | 2006 | 2 | 12949 | 2263 | 1412 | 2145 | 9249 | 2153 | 9130 | -7 | 119 |
| 15 | 2006 | 3 | 12950 | 2231 | 1414 | 2171 | 9372 | 2129 | 9197 | 42 | 174 |
| 16 | 2006 | 4 | 13038 | 2167 | 1474 | 2168 | 9457 | 2169 | 9467 | -1 | -11 |
| 17 | 2007 | 1 | 13056 | 2145 | 1497 | 2200 | 9602 | 2151 | 9551 | 49 | 51 |
| 18 | 2007 | 2 | 13174 | 2193 | 1521 | 2211 | 9721 | 2204 | 9632 | 7 | 89 |
| 19 | 2007 | 3 | 13270 | 2176 | 1578 | 2216 | 9818 | 2248 | 9833 | -32 | -15 |
| 20 | 2007 | 4 | 13326 | 2124 | 1622 | 2187 | 9949 | 2258 | 10017 | -71 | -68 |
| 21 | 2008 | 1 | 13267 | 2056 | 1644 | 2194 | 10019 | 2204 | 10098 | -10 | -79 |
| 22 | 2008 | 2 | 13311 | 2024 | 1694 | 2180 | 10127 | 2211 | 10286 | -31 | -160 |
| 23 | 2008 | 3 | 13187 | 1935 | 1679 | 2143 | 10136 | 2128 | 10295 | 16 | -159 |
| 24 | 2008 | 4 | 12884 | 1745 | 1581 | 2059 | 9861 | 1984 | 10104 | 75 | -242 |
| 25 | 2009 | 1 | 12711 | 1516 | 1453 | 1856 | 9768 | 1889 | 9785 | -33 | -17 |
| 26 | 2009 | 2 | 12701 | 1401 | 1455 | 1777 | 9764 | 1883 | 9596 | -106 | 168 |
| 27 | 2009 | 3 | 12747 | 1395 | 1502 | 1849 | 9889 | 1887 | 9631 | -38 | 257 |
| 28 | 2009 | 4 | 12873 | 1521 | 1585 | 1933 | 9963 | 1956 | 9880 | -23 | 82 |
| 29 | 2010 | 1 | 12948 | 1591 | 1608 | 1981 | 10069 | 1983 | 10011 | -2 | 58 |
| 30 | 2010 | 2 | 13020 | 1646 | 1645 | 2074 | 10148 | 2014 | 10147 | 60 | 1 |
| 31 | 2010 | 3 | 13104 | 1710 | 1684 | 2143 | 10244 | 2050 | 10345 | 93 | -101 |
| 32 | 2010 | 4 | 13181 | 1684 | 1725 | 2143 | 10402 | 2069 | 10589 | 74 | -187 |
| 33 | 2011 | 1 | 13184 | 1662 | 1749 | 2165 | 10566 | 2037 | 10681 | 129 | -115 |
| 34 | 2011 | 2 | 13265 | 1711 | 1766 | 2166 | 10685 | 2066 | 10725 | 100 | -40 |
| 35 | 2011 | 3 | 13307 | 1736 | 1793 | 2191 | 10791 | 2072 | 10785 | 119 | 6 |
| 36 | 2011 | 4 | 13441 | 1867 | 1799 | 2217 | 10874 | 2144 | 10731 | 74 | 143 |
| 37 | 2012 | 1 | 13506 | 1895 | 1819 | 2234 | 11007 | 2159 | 10804 | 75 | 203 |
| 38 | 2012 | 2 | 13549 | 1898 | 1842 | 2250 | 11067 | 2175 | 10897 | 75 | 170 |
| 39 | 2012 | 3 | 13653 | 1929 | 1851 | 2246 | 11154 | 2226 | 10920 | 21 | 234 |

Здесь *Х1* = Gross Domestic Product (ВВП), *X2* = Gross private domestic investment, *X3* = Export, *Y1* = Import, *Y2* = Nominal Consumption.

**Приложение 9. Прогноз по временному ряду с периодическими колебаниями в R**.

Для обработки в R исходные данные Таблицы 7.2 должны быть скопированы в текстовый файл, в данном случае предусмотрено обращение к файлу "D://RR/TimeSeries.txt".

Таблица 7.2.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Год | 2010 | 2011 | 2012 | 2013 | t |
| январ | 0 | 64 | 72 | 83 | 1 |
| февр | 0 | 72 | 80 | 93 | 2 |
| март | 0 | 80 | 83 | 98 | 3 |
| апрель | 0 | 81 | 94 | 102 | 4 |
| май | 0 | 76 | 80 | 97 | 5 |
| июнь | 0 | 84 | 90 | 100 | 6 |
| июль | 0 | 82 | 93 | 102 | 7 |
| август | 0 | 84 | 93 | 102 | 8 |
| сентяб. | 0 | 86 | 91 | 102 | 9 |
| октяб. | 0 | 86 | 96 | 110 | 10 |
| нояб | 0 | 87 | 96 | 110 | 11 |
| дек. | 100 | 108 | 118 | 140 | 12 |

# Временной ряд с сезонными колебаниями из Блокнота

a=read.table("D://RR/TimeSeries.txt", head=TRUE, sep='') ; a

# Чтение файла в вектор а # или clipboard

i=c(1,12); j=c(60,180) ; i ; j # Подготовка осей

plot(i, j, col='white' ) # False line

lines(a[,6],a[,3],type='b', col='blue',lwd=4 )

lines(a[,6],a[,4],type='b', col='blue',lwd=4 )

lines(a[,6],a[,5],type='b', col='blue',lwd=4 )

s1 = sum(a[,3]) ; s1

s2 = sum(a[,4]) ; s2

s3 = sum(a[,5]) ; s3

t = c(1, 2, 3) ; ss = c(s1,s2,s3) ; ss ; t

coef = c( lm(ss ~ t)$coefficients) # Регрессия

coef[1] ; coef[2]

s4 = coef[1] + coef[2]\*4

s5 = coef[1] + coef[2]\*5 ; s4 ; s5

pers = a[,3]+a[,4]+a[,5] ; pers

pers = pers/(s1+s2+s3) ; round(pers , 3) # Проценты по месяцам

Y2014 = pers\*s4; round(Y2014, 1) #Прогноз на 2014 год по месяцам

Y2015 = pers\*s5; round(Y2015,1) #Прогноз на 2015 год по месяцам

lines(a[,6],Y2014,type='b', col='red',lwd=4 )

lines(a[,6],Y2015,type='b', col='red',lwd=4 )

# Автокорреляция

s = c(a[2,12],a[,3],a[,4],a[,5]); s #

aut = c(1:12)

for(i in 1:12) {

x = s[1:24]; m = 1+i; n = 24+i;

y = s[m:n]; aut[i] = cor(x,y)}

round(aut, digits= 3)

plot( aut, lty=1, lwd=2, pch=15,col= 'red', type="b",

main="Autocorrelation",cex.axis=1.7,

cex.lab=1.7, xlab="order")

# plot( s, lty=1, lwd=2, pch=15,col=1, type="b", main="Earnings",

cex.axis=1.7,cex.lab=1.7, xlab="Month")

**Приложение 10. Построение эконометрических трендов**

**в рядах цен фондового рынка**

Private Sub CommandButton1\_Click()

Dim a As Range

Set a = Range("C11")

K1 = Range("K1"): K2 = Range("K2"): dK2 = Range("K2"): Flat = Range("K3")

Ltrend = Range("K4"): Shift1 = Range("H1"): Shift2 = Range("H2")

Shift3 = Range("H3"): mTrend = Range("H4")

m1: b = B1(K1, K2)

a0 = AAA(K1, K2, b)

ckoRes = EEE(K1, K2, a0, b)

m3:

K2 = K2 + 1:

Range("K6") = K2: If K2 > 3000 Then Stop

If a(K2, 2) = Empty Then GoTo m5

If Abs(a(K2 + Shift1, 2) - a0 - b \* a(K2)) > 3 \* ckoRes Then

If Abs(a(K2 + Shift2, 2) - a0 - b \* a(K2)) > 3 \* ckoRes Then

GoTo m4

Else

K2 = K2 + Shift2

End If

Else

If (K2 - K1) > mTrend Then

q = 1: GoTo m4: End If

b = B1(K1, K2) : a0 = AAA(K1, K2, b) : ckoRes = EEE(K1, K2, a0, b)

a(K2, 6) = a(K2, 2) - a0 - b \* a(K2) : GoTo m3

End If

m4:

a(K1, 4) = a0: a(K1, 5) = b: a(K1, 7) = ckoRes

For j = K1 To K2:

a(j, 3) = a0 + b \* a(j): a(j, 6) = a(j, 2) - a0 - b \* a(j):

If Abs(b / a(K1, 2) \* 10000) < Flat Then a(j, 8) = "flat" Else a(j, 8) = "trend"

If (K2 - K1) < Ltrend Then a(j, 8) = ""

Next j

a(K2, 8) = "Bifurcation"

If q = 1 Then

a(K2, 8) = "Long Trend": q = 0

End If

K1 = K2 + Shift3: K2 = K1 + dK2

If K2 > 3000 Then Stop

If a(K2, 2) = Empty Then GoTo m5

GoTo m1

m5: End Sub

Function B1(K1, K2)

Dim a As Range : Set a = Range("C11") : st = 0: Sa = 0

For i = K1 To K2 : st = st + a(i): Sa = Sa + a(i, 2) : Next i

tcp = st / (K2 - K1) : acp = Sa / (K2 - K1) : qt = 0: qa = 0: qc = 0

For i = K1 To K2

qt = qt + (a(i) - tcp) ^ 2 : qa = qa + (a(i, 2) - acp) ^ 2

qc = qc + (a(i) - tcp) \* (a(i, 2) - acp) : Next i

ckot = qt / (K2 - K1 - 1) : Cov = qc / (K2 - K1 - 1) : B1 = Cov / ckot

End Function

Function AAA(K1, K2, b)

Dim a As Range : Set a = Range("C11") : st = 0: Sa = 0

For i = K1 To K2 : st = st + a(i) : Sa = Sa + a(i, 2) : Next i

tcp = st / (K2 - K1) : acp = Sa / (K2 - K1) : AAA = acp - b \* tcp

End Function

Function EEE(K1, K2, a0, b)

Dim a As Range : Set a = Range("C11") : Se = 0

For i = K1 To K2 : Se = Se + (a(i, 2) - a0 - b \* a(i)) ^ 2 : Next i

EEE = (Se / (K2 - K1 - 2)) ^ 0.5

End Function

**Приложение 11. Временные ряды цен на фондовом рынке**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C | D | E | F | G | H | I | J |
| 3,75 | 14,07 | 20,22 | 27,75 | 49,02 | 16,3 | 29,2 | 1,57 | 10,65 | 10,27 |
| 3,9 | 14,5 | 20,45 | 29,58 | 49,68 | 17,02 | 29,68 | 1,58 | 10,95 | 10,4 |
| 4 | 14,1 | 20,77 | 29,65 | 49,04 | 17,08 | 30,02 | 1,57 | 11,01 | 10,5 |
| 4,7 | 14,2 | 20,97 | 29,18 | 49,27 | 17 | 29,04 | 1,52 | 10,96 | 10,5 |
| 4,78 | 14,1 | 21,07 | 29,86 | 48,87 | 17,32 | 29,15 | 1,54 | 11,15 | 10,62 |
| 4,55 | 14,05 | 21,01 | 27,37 | 48,67 | 17,25 | 28,95 | 1,47 | 11,22 | 10,51 |
| 4,46 | 13,98 | 20,86 | 27,38 | 48,19 | 17,21 | 28,55 | 1,35 | 11,04 | 10,31 |
| 4,35 | 13,81 | 20,77 | 27,57 | 48,06 | 17,18 | 28,63 | 1,37 | 11,1 | 10,67 |
| 4,29 | 13,75 | 20,52 | 26,92 | 48,27 | 16,98 | 28,87 | 1,39 | 11 | 10,95 |
| 4,44 | 14,04 | 20,81 | 27,94 | 48,48 | 16,05 | 29,51 | 1,31 | 11,06 | 10,89 |
| 4,45 | 14,25 | 20,95 | 28,31 | 48,35 | 16,05 | 29,75 | 1,32 | 11,29 | 10,9 |
| 4,41 | 14,37 | 20,66 | 30,26 | 47,6 | 15,98 | 28,99 | 1,44 | 10,89 | 10,82 |
| 4,46 | 14,71 | 20,84 | 31,32 | 47,76 | 16,03 | 28,69 | 1,65 | 11,11 | 11,04 |
| 4,62 | 14,9 | 21,21 | 31,5 | 47,72 | 16,08 | 28,11 | 1,71 | 11,15 | 11,14 |
| 5 | 15,57 | 22,26 | 32,73 | 48,26 | 16,48 | 27,73 | 1,66 | 11,06 | 11,77 |
| 4,96 | 15,43 | 22,14 | 33,3 | 48,76 | 17 | 28,05 | 1,78 | 10,95 | 12,11 |
| 5,38 | 15 | 21,85 | 32,99 | 48,46 | 17,46 | 27,98 | 1,7 | 10,97 | 11,82 |
| 5,45 | 14,82 | 21,7 | 31,32 | 48 | 17,47 | 27,98 | 1,58 | 10,94 | 11,95 |
| 5,66 | 15,41 | 22,25 | 32,76 | 48,8 | 17,94 | 28,75 | 1,68 | 11,06 | 12 |
| 5,99 | 15,68 | 22,33 | 33,82 | 49,06 | 17,76 | 29,07 | 1,82 | 11,25 | 11,84 |
| 6,01 | 15,3 | 22,18 | 33,12 | 49,28 | 18,21 | 29,12 | 1,79 | 11,31 | 12,05 |
| 5,96 | 15,75 | 22,99 | 33,16 | 48,7 | 18 | 28,73 | 1,91 | 11,38 | 12,35 |
| 6,89 | 16,24 | 23,76 | 33,25 | 49,3 | 18,4 | 29,25 | 2,2 | 11,48 | 12,63 |
| 6,47 | 16,15 | 23,58 | 33,76 | 49,44 | 18,19 | 31,1 | 2,12 | 11,52 | 12,55 |
| 6,17 | 16,37 | 23,3 | 34,18 | 49,27 | 18,6 | 31,77 | 2,15 | 11,63 | 12,4 |
| 6,25 | 16,62 | 22,81 | 33,58 | 50,25 | 18,8 | 31,11 | 2,12 | 11,69 | 12,36 |
| 6,24 | 16,53 | 22,42 | 32,92 | 50,12 | 18,8 | 30,94 | 2,05 | 11,8 | 12,53 |
| 6,55 | 16,33 | 23,07 | 34,25 | 50,5 | 19,08 | 31,25 | 2,3 | 12,1 | 12,83 |
| 6,95 | 16,98 | 23,42 | 35,21 | 50,4 | 18,99 | 31,26 | 2,43 | 12,17 | 13,49 |
| 6,7 | 17,12 | 23,2 | 34,6 | 50,17 | 18,19 | 31,73 | 2,37 | 12,1 | 13,68 |
| 6,79 | 17,27 | 23,62 | 34 | 50,08 | 18 | 32,31 | 2,38 | 11,86 | 13,67 |
| 6,93 | 17,27 | 23,6 | 35,24 | 50,83 | 18,7 | 32,91 | 2,35 | 11,92 | 13,99 |
| 7,39 | 17,47 | 22,9 | 34,93 | 50,84 | 18,29 | 32,78 | 2,38 | 12,21 | 12,75 |
| 7,29 | 17,01 | 22,63 | 33,28 | 50,55 | 18,16 | 32,83 | 2,27 | 12 | 12,93 |
| 7,38 | 16,79 | 22,97 | 33,63 | 50,33 | 18,06 | 33,34 | 2,23 | 12,02 | 12,75 |
| 6,9 | 16,47 | 22,2 | 34,25 | 50,25 | 17,7 | 33,95 | 2,28 | 11,75 | 12,63 |
| 7,04 | 16,57 | 22,4 | 34,92 | 50,82 | 17,7 | 33,7 | 2,43 | 11,81 | 12,63 |
| 7,32 | 16,33 | 22,51 | 34,9 | 50,38 | 17,9 | 33,91 | 2,43 | 11,73 | 12,87 |
| 7,93 | 16,53 | 21,9 | 36,7 | 50,59 | 17,97 | 34,43 | 2,54 | 11,88 | 13,42 |
| 7,65 | 16,9 | 21,86 | 37,05 | 50,83 | 18,31 | 34,36 | 2,6 | 11,91 | 13,4 |
| 7,93 | 16,79 | 21,98 | 38,33 | 50,34 | 17,63 | 34,56 | 2,59 | 11,86 | 13,5 |
| 7,92 | 17,51 | 22,11 | 38,55 | 51,1 | 17,88 | 34,92 | 2,44 | 11,9 | 13,61 |
| 8,1 | 17,74 | 22,32 | 37,38 | 50,76 | 17,85 | 34,46 | 2,59 | 11,79 | 13,45 |
| 8,1 | 17,6 | 22,16 | 37,85 | 51,12 | 17,56 | 33,5 | 2,53 | 10,95 | 13,49 |
| 8,3 | 18,53 | 22,34 | 39,5 | 50,78 | 18,15 | 34,42 | 2,67 | 10,98 | 13,46 |
| 8,37 | 18,01 | 22,5 | 38,81 | 50,92 | 18,87 | 34,6 | 2,61 | 10,99 | 13,4 |
| 7,83 | 17,76 | 22,36 | 38,53 | 50,44 | 19,15 | 35,23 | 2,63 | 10,69 | 13,28 |
| 8,05 | 16,95 | 22,1 | 36,7 | 50,55 | 19,03 | 35,32 | 2,5 | 10,7 | 13,09 |
| 7,67 | 17,1 | 22,24 | 37,28 | 49,65 | 18,88 | 36,25 | 2,64 | 10,6 | 13,9 |
| 7,94 | 17,05 | 22,14 | 36,44 | 49,95 | 19,15 | 37,6 | 2,61 | 10,57 | 14,23 |
| 8,05 | 17,43 | 22,39 | 35,3 | 50,33 | 19,65 | 37,25 | 2,59 | 10,67 | 14,48 |
| 7,97 | 17,25 | 22,3 | 34,44 | 50,33 | 19,49 | 37,15 | 2,5 | 10,66 | 14,12 |
| 8,2 | 17,75 | 22,82 | 35 | 51,53 | 21 | 38,3 | 2,35 | 11,05 | 14,65 |
| 7,39 | 17,71 | 22,92 | 33,9 | 51,32 | 22,22 | 37,95 | 2,41 | 11,17 | 14,79 |
| 7,15 | 17,87 | 22,94 | 35,17 | 51,7 | 20,69 | 38,15 | 2,53 | 11,14 | 14,77 |
| 7,2 | 17,06 | 22,37 | 35,17 | 51,2 | 20,3 | 36,75 | 2,46 | 11,17 | 13,75 |
| 6,93 | 17,63 | 22,61 | 33,99 | 51,54 | 20,33 | 36,69 | 2,46 | 11,12 | 14,3 |
| 6,48 | 17,25 | 22,29 | 33,05 | 51,2 | 20,08 | 35,99 | 2,33 | 10,99 | 13,56 |
| 6,35 | 17,71 | 22,42 | 32,75 | 51,77 | 19,52 | 35,81 | 2,23 | 10,82 | 13,5 |
| 6,6 | 17,6 | 22,4 | 33,04 | 50,8 | 19,62 | 35,69 | 2,26 | 10,84 | 12,96 |
| 6,59 | 17,78 | 22,65 | 34,56 | 50,6 | 19,51 | 36,2 | 2,34 | 10,84 | 12,75 |
| 6,38 | 17,95 | 22,65 | 34,05 | 50,52 | 16,93 | 36,74 | 2,28 | 10,9 | 13 |
| 6,35 | 17,99 | 22,8 | 34,82 | 51,1 | 17,08 | 37,2 | 2,33 | 10,96 | 13,4 |
| 6,28 | 17,74 | 22,41 | 35,25 | 51,5 | 16,44 | 37,2 | 2,41 | 11 | 12,98 |
| 6,95 | 18,84 | 22,47 | 35,82 | 52,16 | 15,89 | 38,79 | 2,6 | 11,03 | 13,2 |
| 7,14 | 19,15 | 22,6 | 36 | 52,07 | 16,2 | 38,52 | 2,52 | 11,06 | 13,1 |
| 7,28 | 19,98 | 22,82 | 36,71 | 52,43 | 16,36 | 39,22 | 2,7 | 11,15 | 13,75 |
| 7,17 | 19,9 | 23,56 | 37,29 | 52,28 | 16,46 | 40,19 | 2,98 | 11,1 | 13,67 |
| 7,25 | 19,98 | 23,74 | 37,44 | 52,25 | 16,33 | 40,44 | 3,1 | 11,05 | 14,2 |
| 7 | 19,77 | 22,89 | 36,25 | 52,25 | 16,57 | 39,96 | 2,99 | 10,73 | 14,02 |
| 7,05 | 19,65 | 22,96 | 36,7 | 53,33 | 17,47 | 40,26 | 2,99 | 10,62 | 14,32 |
| 7,1 | 20,1 | 23,16 | 38,16 | 53,55 | 17,44 | 39,72 | 3,01 | 10,75 | 14,3 |
| 7,05 | 20,4 | 23,52 | 38,04 | 53,65 | 17,23 | 39,04 | 3 | 10,79 | 14,57 |
| 6,97 | 21,1 | 23,33 | 37,91 | 53,69 | 17,17 | 38,34 | 2,93 | 10,87 | 14,65 |
| 6,92 | 20,62 | 23,25 | 36,13 | 52,99 | 17,49 | 37,5 | 2,82 | 10,78 | 14,49 |
| 6,9 | 21,03 | 23,47 | 35,7 | 54,5 | 17,64 | 38,02 | 2,82 | 10,8 | 14,42 |
| 6,51 | 20,47 | 23,17 | 35,76 | 54,54 | 17,62 | 38,68 | 2,68 | 10,72 | 14,04 |
| 6,84 | 20,35 | 23,14 | 37,48 | 55,04 | 18,1 | 38,68 | 2,83 | 10,75 | 14,42 |
| 6,68 | 20,08 | 23,06 | 37,5 | 54,98 | 18,5 | 38,7 | 2,67 | 10,88 | 14,4 |
| 6,91 | 20,59 | 22,94 | 37,13 | 54,68 | 18,9 | 38,26 | 2,88 | 10,74 | 14,48 |
| 6,85 | 21,02 | 23,03 | 38,18 | 55,98 | 19,27 | 39,45 | 2,75 | 11,21 | 14,24 |
| 6,73 | 20,86 | 23,13 | 38,75 | 56,75 | 18,8 | 41,1 | 2,72 | 11,41 | 14,63 |
| 6,59 | 23,31 | 22,92 | 38 | 56,32 | 18,7 | 42,49 | 2,7 | 11,3 | 15,34 |
| 6,3 | 22,21 | 23,04 | 37,29 | 55,93 | 19,05 | 41,95 | 2,82 | 11,17 | 15,35 |
| 6,29 | 22,34 | 22,9 | 37,95 | 55,63 | 19,18 | 41,22 | 2,81 | 11,2 | 15,27 |
| 6,08 | 22,07 | 22,65 | 39,51 | 55,2 | 19,13 | 40,2 | 2,74 | 11,02 | 14,62 |
| 5,98 | 21,6 | 21,8 | 39,73 | 55,48 | 18,78 | 41,75 | 2,7 | 10,99 | 14,91 |
| 6,01 | 20,8 | 21,82 | 38,19 | 55,41 | 18,53 | 40 | 2,65 | 10,92 | 14,84 |
| 6 | 21,4 | 21,84 | 38,25 | 55,82 | 18,27 | 40,55 | 2,58 | 10,69 | 14,8 |
| 5,91 | 21,15 | 21,69 | 37,27 | 55,61 | 18,33 | 41,75 | 2,52 | 10,97 | 14,93 |
| 6,04 | 21,37 | 21,93 | 36,03 | 55,87 | 18,62 | 42 | 2,46 | 10,91 | 14,94 |
| 6,04 | 21,53 | 21,37 | 36,77 | 55,84 | 18,14 | 40,82 | 2,36 | 11,48 | 15 |
| 6,25 | 22,02 | 21,49 | 36,48 | 56,24 | 19,13 | 41,48 | 2,41 | 11,35 | 15,32 |
| 6,33 | 22,35 | 21,35 | 37,06 | 56,26 | 20,23 | 41,99 | 2,6 | 11,36 | 15,11 |
| 6,34 | 22,68 | 21,66 | 37,09 | 56,53 | 20,67 | 41,51 | 2,71 | 11,33 | 15,02 |
| 6,26 | 22,17 | 21,62 | 36,35 | 56,76 | 20,46 | 42 | 2,82 | 11,32 | 15,1 |
| 6,12 | 23,05 | 21,71 | 38,4 | 56,65 | 21,03 | 42 | 2,85 | 11,62 | 15,08 |
| 6,13 | 23,41 | 21,96 | 38,8 | 56,21 | 21,89 | 42,05 | 2,8 | 11,82 | 15,12 |
| 6,42 | 23,31 | 21,93 | 38,84 | 56,31 | 21,98 | 41,5 | 2,83 | 12 | 15,2 |
| 6,61 | 23,56 | 21,93 | 40,5 | 56,4 | 22,63 | 42,1 | 2,84 | 11,92 | 15,31 |
| 6,53 | 23,01 | 21,42 | 41,51 | 55,83 | 22,08 | 42,28 | 2,88 | 11,97 | 15,13 |
| 6,29 | 22,82 | 21,6 | 40,7 | 56,09 | 21,68 | 41,7 | 2,76 | 11,95 | 14,97 |
| 6,22 | 23,46 | 21,67 | 39,6 | 55,99 | 21,34 | 41,4 | 2,76 | 11,87 | 14,77 |
| 6,33 | 23,37 | 21,52 | 40,97 | 56,19 | 21,58 | 41,55 | 2,73 | 12,12 | 14,88 |
| 6,33 | 23,58 | 21,74 | 40,96 | 56,87 | 21,38 | 41 | 2,95 | 12,13 | 15,04 |
| 6,48 | 23,67 | 21,88 | 41 | 57,08 | 22,08 | 40,72 | 3,02 | 12,15 | 15,16 |
| 6,63 | 23,86 | 22 | 41,7 | 57,43 | 21,25 | 41,2 | 3,15 | 12,31 | 15,19 |
| 6,69 | 24,15 | 22,2 | 40,98 | 57,72 | 20,5 | 41,9 | 3,25 | 12,73 | 15,76 |
| 6,97 | 24,6 | 22,2 | 41,15 | 58,45 | 20,44 | 41,58 | 3,19 | 12,75 | 15,83 |
| 7,1 | 24,26 | 22,23 | 41,78 | 57,75 | 19,8 | 41,2 | 3,59 | 12,72 | 15,76 |
| 7,19 | 24,39 | 22,5 | 42,74 | 57,79 | 20,23 | 41,9 | 3,7 | 12,75 | 15,76 |
| 7,05 | 24,4 | 22,5 | 41,88 | 57,41 | 19,88 | 42,53 | 3,53 | 12,8 | 16,06 |

**Приложение 12. Исследование нелинейного взаимодействия**

**в R и Python**

# d2x/dx2 = - k\*(exp(a\*x)-1)

x = c(1:10) ; v = x ; w = x ; x[1] =10 ; v[1] = 10 ; w[1] = 10

dt = 1

# k = 0.000081

# for ( n in 2 : 3000) { w[n] = -k\*x[n-1]^3

a=0.093 ; k=0.99

for ( n in 2 : 6000) {if (x[n-1]>0) b=1 else b=-1

w[n] = -k\*exp(a\*abs(b\*x[n-1])-1)\*b

v[n] = v[n-1] + w[n]\*dt

x[n] = x[n-1] + v[n]\*dt

}

plot( x, type = 'l', col = 'blue', lwd =4)

lines( v, type = 'l', col = 'red', lwd =4)

lines( w, type = 'l', col = 'green', lwd =4)

h=c(1:2); nh=1

for(k in 1:1000) {

x = c(1:10) ; v = x ; w = x ; x[1] =10 ; v[1] = 10 ; w[1] = 10

dt = 1 ; s=0.1

a=0.093; k=0.9

for ( n in 2 : 6000) {if (x[n-1]>0) b=1 else b=-1

if (abs(x[n-1]) > 100) { h[nh]= n ; nh=nh+1; break }

w[n] = -k\*exp(a\*abs(b\*x[n-1]-1))\*b

v[n] = v[n-1] + w[n]\*dt

x[n] = x[n-1] + v[n]\*dt +rnorm(1,0,s)

if (abs(x[n]) > 100) { h[nh]= n ; nh=nh+1; break }

}

plot( x, type = 'l', col = 'blue', lwd =4)

}

h

cases=as(h/600,'integer')

casefactor=factor(cases)

plot(casefactor, col="blue")

Аналогичный код на Python:

import numpy as np

import matplotlib.pyplot as plt

x = list([1, 2, 3, 4, 5, 6, 7, 8, 9, 10]) ; x.extend([None]\*5990)

v = list([1, 2, 3, 4, 5, 6, 7, 8, 9, 10]) ; v.extend([None]\*5990)

w = list([1, 2, 3, 4, 5, 6, 7, 8, 9, 10]) ; w.extend([None]\*5990)

x[0] = 10 ; v[0] = 10 ; w[0] = 10 ; dt = 1 ; a = 0.093 ; k = 0.99

for n in range(1, 6000):

if x[n - 1] > 0:

b = 1

else:

b = -1

w[n] = -k \* np.exp(a \* abs(b \* x[n - 1]) - 1) \* b

v[n] = v[n - 1] + w[n] \* dt

x[n] = x[n - 1] + v[n] \* dt

plt.plot(np.arange(1, 6001), x, color='blue', linewidth=4)

plt.plot(np.arange(1, 6001),v, color='red', linewidth=4)

plt.plot(np.arange(1, 6001), w, color='green', linewidth=4)

h = list([1, 2, 3])

h.extend([None]\*5997)

nh = 1

for k in range(1, 1001):

x = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

x.extend([None]\*5990)

v = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

v.extend([None]\*5990)

w = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

w.extend([None]\*5990)

x[0] = 10 ; v[0] = 10 ; w[0] = 10 ; dt = 1 ; s = 0.1 ; a = 0.093 ; k = 0.9

for n in range(1, 6000):

if x[n - 1] > 0:

b = 1

else:

b = -1

if abs(x[n - 1]) > 100:

h[nh] = n

nh = nh + 1

break

w[n] = -k \* np.exp(a \* abs(b \* x[n - 1] - 1)) \* b

v[n] = v[n - 1] + w[n] \* dt

x[n] = x[n - 1] + v[n] \* dt + np.random.normal(0, s)

if abs(x[n]) > 100:

h[nh] = n

nh = nh + 1

break

plt.plot(x, color='blue', linewidth=4)

plt.show()

h = np.array(h)

h = h[h!=None]

print(h)

plt.hist(h/600, bins = 10, color="blue")

**Приложение 13. Инструкция получения данных  
с различных сайтов: kaggle, МосБиржа, яхо, Всемирный банк, датасеты из эконометрической библиотеки.**

С помощью Python можно загрузить данные для анализа, используя API. API (application programming interface) позволяет взаимодействовать с другими сервисами, получать и отправлять данные. Для работы с API понадобится библиотека requests. Установим ее:

pip install requests

На примере МосБиржи рассмотрим, как можно получить данные. Перед этим установим библиотеку:

pip install apimoex

Пример кода для тикера 'VTBR'

import apimoex

from datetime import date, timedelta

import requests

import pandas as pd

ticker = 'VTBR'

n\_days=100

start = str(date.today() - timedelta(days=n\_days))

end = str(date.today())

with requests.Session() as session:

data = apimoex.get\_market\_candles(session, ticker, start=start, end=end)

df = pd.DataFrame(data)

Также скачивать данные с помощью API можно с известного сайта kaggle.com, который содержит множество датасетов на различные темы:

pip install kaggle

Для корректной работы пакета необходим файл json, который хранил бы ваши данные с портала. Его можно скачать на сайте или создать самим. Для этого необходимо перейти в настройки и выполнить действия: Settings->API->Create New Token. Далее автоматически будет скачан файл kaggle.json. Его необходимо поместить в директорию C:\Users\<Windows-username>\.kaggle\kaggle.json (Windows) или /Users/mac\_username/.kaggle/kaggle.json (Mac). Создать файл можно следующим образом:

import json

api\_token = {'username': 'your\_ username’, 'key':" your\_key'"}

with open('/Users/ mac\_username/.kaggle/kaggle.json', 'w') as file:

json.dump(api\_token, file)

Найдем наборы данных, которые содержат в названии ВВП:

datasets = kaggle.api.dataset\_list(search="gdp")

И выберем первый из списка и скачаем его:

ds = datasets[0]

kaggle.api.dataset\_download\_files(ds.ref)

Файл появился в папке с ipynb, но можно выбрать другую, указав в аргументе path полный путь.

Далее необходимо произвести разархивацию, чтобы извлечь файлы:

path\_to\_zip\_file = 'world-gdpgdp-gdp-per-capita-and-annual-growths.zip'

import zipfile

with zipfile.ZipFile(path\_to\_zip\_file, 'r') as zip\_ref:

zip\_ref.extractall()

data = pd.read\_csv('gdp.csv', index\_col = 0)

Возьмем данные только по России:

data\_ru = data.loc['Russian Federation']

Котировки акций зарубежных компаний можно получить с портала yfinance, используя его API. Оно имеет простые функции и методы:

pip install yfinance

import yfinance as yf

msft = yf.Ticker("MSFT") #Выбираем тикер

hist = msft.history(period="1mo") #Выбираем период

Результат – pandas.DataFrame, что очень удобно для дальнейшей работы с данными.

Макроэкономические показатели можно найти на сайте Всемирного банка, для которого создан API:

pip install world-bank-data

import world\_bank\_data as wb

Получим данные по ВВП США:

gdp = pd.DataFrame(wb.get\_series('NY.GDP.MKTP.CD', mrv=100))

Есть удобная функция для поиска:

wb.search\_indicators('gdp') #поиск по индикаторам

Небольшое количество датасетов, которые отлично подойдут для учебных задач, можно получить из библиотеки stetdmodels:

import statsmodels.api as sm

import pandas as pd

data = sm.datasets.anes96.load\_pandas()

data.data

help(sm.datasets) #все датасеты

Описание наборов данных: <https://www.statsmodels.org/dev/datasets/index.html>

Если необходимо скачать данные с GitHub, то можно использовать команду git clone name\_repo. Данная строка копирует все файлы из указанного репозитория.

Таким образом, API – очень удобный и быстрый способ получения данных в Python. Многие крупные порталы и сайты создают такой интерфейс для упрощенной загрузки данных.