

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение  
высшего образования  
«Уральский федеральный университет имени  
первого Президента России Б. Н. Ельцина»

**ПОСТРОЕНИЕ ТИПОВЫХ МОДЕЛЕЙ АРПСС (ARIMA)**

**Методические указания к выполнению  
лабораторной работы № 5**

Екатеринбург

2020

## Содержание

|   |    |
|---|----|
| Введение.....                           | 3  |
| 1. Задание на лабораторную работу ..... | 3  |
| 2. Требования к оформлению отчета.....  | 13 |

## Введение

Напомним, что в общем виде модель авторегрессии – скользящего среднего порядка  $(p, q)$  АРПСС (ARIMA) выглядит как:

$$\tilde{z}_t = \phi_1 \tilde{z}_{t-1} + \phi_2 \tilde{z}_{t-2} + \dots + \phi_p \tilde{z}_{t-p} + a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2} - \dots - \theta_q a_{t-q}.$$

Эта модель временных рядов имеет целый ряд преимуществ в сравнении с другими моделями, одно из которых – это возможность их оперативного прогноза по построенной модели. В связи с этим ни одна методика анализа и изучения временных рядов не может обойтись без рассмотрения подобного класса задач.

### 1. Задание на лабораторную работу

Результатом выполнения лабораторной работы является оформленный отчет в виде *Jupyter*-тетради, в котором должны быть представлены и отражены все нижеперечисленные пункты:

- 1) Сначала импортируйте в свой код нужные библиотеки, функции и т.д.

```
import numpy as np  
import numpy.random as rand  
import matplotlib.pyplot as plt  
import h5py  
from statsmodels.tsa import api as tsa  
from statsmodels.graphics.tsaplots import plot_acf  
from statsmodels.tsa.arima_model import ARIMA  
%matplotlib inline
```

- 2) Для начала попробуем создать собственные АРПСС ряды первого и второго порядков и изучить их автокорреляционные функции.
- 3) Создадим два АР(1) процесса первого порядка:

$$z_t = 0.8z_{t-1} + a_t \quad \text{и} \quad z_t = -0.8z_{t-1} + a_t$$

где  $a_t$  – случайная нормально распределенная величина малой амплитуды (порядка 0.2),  $z_0 = 1$ .

```
z1 = np.zeros(100)  
z2 = np.zeros(100)  
z1[0] = 1  
z2[0] = 1  
for i in range(1,100):  
    z1[i] = 0.8 * z1[i - 1] + 0.2 * np.random.randn()  
    z2[i] = -0.8 * z2[i - 1] + 0.2 * np.random.randn()  
plt.figure(figsize = (10, 5))  
plt.plot(z1, 'b')  
plt.plot(z2, 'r')  
plt.show()
```

- 4) Постройте для этих рядов функции автокорреляции с помощью функции **plot\_acf**:

```
plt.figure(figsize = (10, 5))  
plot_acf(z1, lags=50)  
plot_acf(z2, lags=50)  
plt.show()
```

- 5) Сравните эти графики между собой: найдите их сходства и различия, а также характерные особенности, которые позволяют отнести их к модели АР первого порядка.

- 6) Оцените весовой параметр этих процессов (как если бы Вы не знали о них) с помощью формулы  $\phi = \rho_1$ , на основе функции автокорреляции. Также удостоверьтесь, что для модели  $AR(1)$  коэффициенты автокорреляции изменяются по степенному закону  $\rho(l) = \phi^l$ .

- 7) Аналогичным образом постройте два  $CC(1)$  процесса среднего-скользящего первого порядка:

$$z_t = a_t - 0.8a_{t-1} \quad \text{и} \quad z_t = a_t - (-0.8)a_{t-1}$$

где  $a_t$  – случайная нормально распределенная величина

```
z3 = np.zeros(100)
z4 = np.zeros(100)
ar = 0.2 * np.random.randn(100)
for i in range(1, 100):
    z3[i] = ar[i] - 0.8 * ar[i - 1]
    z4[i] = ar[i] + 0.8 * ar[i - 1]
plt.figure(figsize = (10, 5))
plt.plot(z3, 'b')
plt.plot(z4, 'r')
plt.show()
```

- 8) Постройте для этих рядов функции автокорреляции, достаточно взять 25 лагов (четверть от длины ряда).
- 9) Сравните эти графики между собой: найдите их сходства и различия, а также характерные особенности, которые позволяют отнести их к модели  $CC$  первого порядка.

- 10) Оцените весовой параметр этих процессов (как если бы Вы не знали о них) с помощью формулы ниже, на основе функции автокорреляции.

$$\theta_1^2 + \theta_1 / \rho_1 + 1 = 0, \quad |\theta_1| < 1$$

- 11) Также удостоверьтесь, что для модели СС(1) коэффициенты автокорреляции соответствуют формуле

$$\rho_k = \begin{cases} \frac{-\theta_1}{1 + \theta_1^2}, & k = 1 \\ 0, & k \geq 2 \end{cases}$$

- 12) Наконец, создайте временной ряд процесса АРСС(1, 1):

$$z_t = 0.8z_{t-1} + a_t - 0.3a_{t-1} \quad \text{и} \quad z_t = -0.8z_{t-1} + a_t - 0.3a_{t-1}$$

где  $a_t$  – случайная нормально распределенная величина,  $z_0 = 1$ .

Напишите код *Python* самостоятельно на основе комбинации предыдущих примеров.

- 13) Постройте графики этих рядов и графики их автокорреляционных функций.
- 14) Есть и другой, более высокоуровневый способ генерации рядов АРПСС. Используем следующую функцию для создания АРСС (2, 2):

```
from statsmodels.tsa.arima_process import arma_generate_sample
ar = np.array([0.75, -0.25]) # задаем коэффициенты АР
ma = np.array([0.65, 0.35]) # задаем коэффициенты СС
y = arma_generate_sample(np.r_[1, -ar], np.r_[1, ma], 100)
# создаем ВР для АРСС (2, 2) = АРПСС (2, 0, 2) из 100 отсчетов
```

- 15) Теперь проведем анализ неизвестного ряда на типовом примере, а затем каждый из студентов проводит анализ собственного ВР по вариантам (номер варианта = последние две цифры студенческого билета).
- 16) Значения исходного ряда (всего их 24) приведены ниже:
- TEST = [0.00, 9.99, 12.89, 10.70, 5.12, -1.21, -6.50, -7.96, -4.30, 0.42, 3.41, 4.50, 3.57, 2.24, 1.78, 0.89, -1.20, -3.43, -2.35, -0.85, -0.21, -0.08, 0.95, 0.45]**
- 17) Постройте график ВР и его автокорреляционную функцию.
- 18) По ним можно судить, что ВР, в достаточной степени, **стационарен**, а, так как, эта функция является **знакопеременной**, то один из членов АР модели имеет отрицательный вес.
- 19) Создадим три пробные модели АРПСС для проверки ряда на  $AP(1) = AP(1, 0, 0)$ ,  $AP(2)$ ,  $AP(3)$ , без тренда ( $trend = 'nc'$ ):

```
arima1 = ARIMA(TEST, order = (1, 0, 0))           # создаем модель
model_fit1 = arima1.fit(dis = False, trend='nc') # подгоняем под ВР
print(model_fit1.summary())                       # выводим таблицу результатов
arima2 = ARIMA(TEST, order = (2, 0, 0))
model_fit2 = arima2.fit(dis = False, trend='nc')
print(model_fit2.summary())
arima3 = ARIMA(TEST, order = (3, 0, 0))
model_fit3 = arima3.fit(dis = False, trend='nc')
print(model_fit3.summary())
```

- 20) Будут выведены три таблицы со всевозможной информацией, например, как ниже:

| ARMA Model Results |                  |                     |         |           |        |        |
|--------------------|------------------|---------------------|---------|-----------|--------|--------|
| Dep. Variable:     | y                | No. Observations:   | 24      |           |        |        |
| Model:             | ARMA(2, 0)       | Log Likelihood      | -41.543 |           |        |        |
| Method:            | css-mle          | S.D. of innovations | 1.201   |           |        |        |
| Date:              | Sun, 10 Mar 2019 | AIC                 | 89.086  |           |        |        |
| Time:              | 23:50:38         | BIC                 | 92.620  |           |        |        |
| Sample:            | 0                | HQIC                | 90.024  |           |        |        |
|                    | coef             | std err             | z       | P> z      | [0.025 | 0.975] |
| ar.L1.y            | 1.5108           | 0.056               | 27.117  | 0.000     | 1.402  | 1.620  |
| ar.L2.y            | -0.9641          | 0.035               | -27.509 | 0.000     | -1.033 | -0.895 |
| Roots              |                  |                     |         |           |        |        |
|                    | Real             | Imaginary           | Modulus | Frequency |        |        |
| AR.1               | 0.7836           | -0.6506j            | 1.0185  | -0.1103   |        |        |
| AR.2               | 0.7836           | +0.6506j            | 1.0185  | 0.1103    |        |        |

- 21) В этой таблице значения коэффициентов модели авторегрессии AP(2) написаны в столбце **coef**. СКВО их расчета – в следующем столбце.
- 22) **Как по этим таблицам выбрать наилучшую модель?** Во-первых, стоит обратить внимание на значение **AIC** – информационный критерий Акаике, который показывает максимальное правдоподобие модели при штрафовании за избыточные параметры системы. Считается, что наилучшей будет модель с **наименьшим** значением критерия AIC.
- 23) Аналогично есть **BIC** – Байесовский информационный критерий, модификация AIC. Данный критерий налагает больший штраф на увеличение количества параметров по сравнению с AIC.
- 24) Аналогично есть **HQIC** – информационный критерий Ханнана-Куинна (Hannan-Quinn), который асимптотически более точный метод чем BIC для дискретных параметров.



25) В любом случае, лучшей моделью будет та, что имеет наименьшее значение информационного критерия среди множества других. Рекомендуется, в первую очередь, выбирать по критерию **BIC**, так как он сильнее штрафует за переобучение модели и увеличение числа параметров по сравнению с другими. В нашем случае для тестового ВР, для любых информационных критериев, это модель  $AP(2)$ .

26) Другим методом выбора модели может служить построение моделей АРСС выбранного порядка и с найденными коэффициентами на графиках совмещенно.

**`plt.plot(model_fit.fittedvalues)`**

Например, для приведенного примера модель  $AP(1)$  совсем слабо подходит к ВР,  $AP(2)$  и  $AP(3)$  близки,  $AP(3)$  почти не отличается от  $AP(2)$ , но избыточен по числу параметров ( $3 > 2$ ), а значит  $AP(2)$  является наиболее оптимальной моделью ВР.

27) Теперь попробуйте найти весовые коэффициенты для АР моделей только **1 и 2 порядка** самостоятельно. Для этого Вам потребуется построить автокорреляционную функцию этого ряда.

28) Для нахождения весового коэффициента  $AP(1)$  используйте следующую формулу:

$$\phi = \rho_1.$$

где  $\rho_1 - r(1)$  оценка автокорреляционной функции.

29) Для  $AP(2)$  используйте следующие формулы:

$$\phi_1 = \frac{\rho_1(1 - \rho_2)}{1 - \rho_1^2}, \quad \phi_2 = \frac{\rho_2 - \rho_1^2}{1 - \rho_1^2}.$$

30) Убедитесь, что полученные веса будут близки к тем, что были получены с помощью функций *Python*.

- 31) Теперь в зависимости от своего варианта, который определяется по последним двум цифрам студ. билета, выберите из выданных преподавателей **mat-файлов** тот, который имеет номер Вашего варианта и загрузите из него временной ряд **Z**, например:

```
file = h5py.File('12.mat', 'r')
```

```
data = file.get('z12')
```

```
Z = np.array(data)
```

- 32) Постройте график ВР и его автокорреляционную функцию.
- 33) Оцените порядок АРСС модели с помощью класса ARIMA. Для упрощения задачи выбора модели используйте только чистые АР или СС модели, то есть класс ARIMA с  $\text{order} = (p, 0, 0)$  или  $\text{order} = (0, 0, q)$ .
- 34) Выберите модель с наиболее подходящей структурой и вычислите для нее коэффициенты. Поясните в отчете выбор модели.
- 35) В дальнейшем попробуйте подобрать такую модель АРСС (ARIMA) со всевозможными параметрами  $\text{order} = (p, d, q)$ , которая будет наилучшей для данного ВР среди всех других по одному из информационных критериев.
- 36) Теперь обратимся к **прогнозированию** на основе АРСС моделей. Загрузите из mat-файла **Fort.mat** ряд, содержащий отсчеты некоторого реального ВР, всего 174 отсчета в вектор-строке.

```
file = h5py.File('Fort.mat', 'r')
```

```
data = file.get('Fort')
```

```
Fort = np.array(data)
```

```
plt.figure(figsize = (10, 5))
```

```
plt.plot(Fort, 'k')
```

```
plt.show()
```

- 37) Мы будем производить **ретроспективный прогноз**, аналогично предыдущей лабораторной работе. Для этогоотрежем от данного ряда последние 24 точки (которые мы и будем прогнозировать):

```
Z = Fort[:len(Fort)-24+1] # отрезаем последние 24 точки
t=np.arange(0, len(Z), 1) # временная шкала для регрессии
t=t.reshape(-1,1)
plt.figure(figsize = (10, 5))
plt.plot(Fort, 'k') # исходный ВР
plt.plot(t, Z, 'b') # урезанный ряд
plt.show()
```

- 38) Прежде, чем строить модель АРПСС, **обратите внимание:** модели АРПСС строятся для рядов с около-нулевым средним, что неверно для заданного временного ряда. Поэтому – **сначала постройте линейный тренд прогнозируемого ряда** (см. линейную регрессию первого порядка из предыдущей лаб. работы), **а затем вычтите его из исходного ряда**, приведя его к нулевому среднему значению (к так называемой **тренд-стационарной форме**).

- 39) Подберите для данного приведенного к нулю ВР, у которого к тому же отрезали последние 24 точки, модель АРПСС ( $p, d, q$ ) некоторого порядка (все параметры целиком и полностью определяются самим студентом) по таблицам и информационным критериям. Например, была найдена некоторая наилучшая модель:

```
arimaz = ARIMA(Z_minus_trend, order = (p, d, q))
model_fit = arimaz.fit(dispatch = False) # подгоняем под ВР
print(model_fit.summary())
```

- 40) Тогда график прогноза по данной модели вместе с доверительными интервалами строится очень легко:

```
model_fit.plot_predict(0, len(Fort))
```

- 41) Но хотелось бы все же увидеть – как же этот прогноз по АРПСС модели соотносится с исходными известными 24 прогнозными точками (ведь прогноз все-таки ретроспективный). Для этого нужно из исходного ряда *Fort* тоже вычесть линейный тренд и соотнести их на одном изображении:

```
plt.figure(figsize = (10, 5))
```

```
model_fit.plot_predict(0, len(Fort)) # прогноз по АРПСС
```

```
plt.plot(t0, Fort-(trend_as_func_of_t0), 'r') # исходный ВР минус тренд
```

```
plt.show()
```

- 42) Сами прогнозные значения по модели АРПСС можно получить с помощью функции **predict**:

```
model_fit.predict(len(Z), len(Fort))
```

- 43) Используйте эти значения для оценки точности прогноза на основе оценок из предыдущей лабораторной работы (все, кроме коэффициентов несоответствия).

- 44) Также попробуйте построить АРПСС модель для прогнозирования данного ряда, **но без исходного вычитания из него линейного тренда**. Отметьте получившиеся отличия в работе функций *Python* и точности конечных результатов.

- 45) Аналогично, для данной модели постройте графики прогноза с доверительными интервалами относительно оригинального ряда *Fort*, а также оцените точность прогноза на основе оценок из пункта 11 выше.

- 46) Не забудьте в отчет-тетрадь добавить необходимые рисунки и таблицы результатов.

## **2. Требования к оформлению отчета**

Отчет в Jupyter-тетради должен обязательно содержать: номер лабораторной работы, ФИО студента, номер варианта (либо студенческий номер), номер группы, результаты выполнения работы с комментариями студента (комментарии пишутся после #) и изображениями.