ДЗ 5 Зименкова

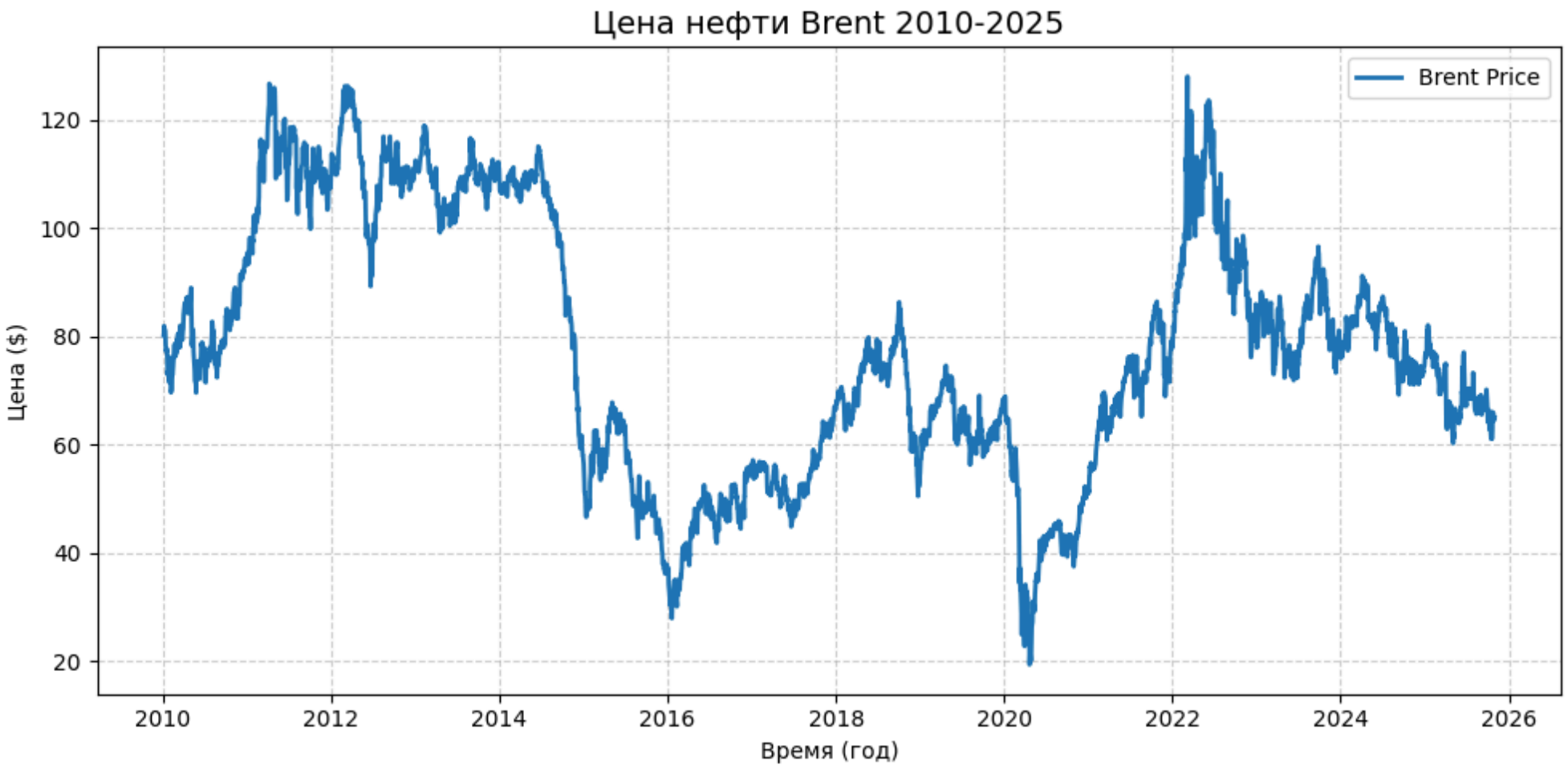
ноутбук colab: https://colab.research.google.com/drive/1boA62qomLSz1XdUGykTmv-Aj5mnSD40q?usp=sharing (https://colab.research.google.com/drive/1boA62qomLSz1XdUGykTmv-Aj5mnSD40q?usp=sharing)

№1

PYTHON

```python
import yfinance
import pandas
import matplotlib.pyplot as plt
from statsmodels.tsa.stattools import adfuller, kpss

# Цены
brent_price_data = yfinance.download('bz=f',start='2010-01-01',end='2025-10-31')
brent_price_data = pandas.DataFrame(brent_price_data)

# График
plt.figure(figsize=(10, 5))
plt.plot(brent_price_data.index, brent_price_data['Close'], label='Brent Price', linewidth=2)
plt.title('Цена нефти Brent 2010-2025', fontsize=14)
plt.xlabel('Время (год)')
plt.ylabel('Цена ($)')
plt.grid(True, linestyle='--', alpha=0.6)
plt.legend()
plt.tight_layout()
plt.show()

# Проверка стационарности

# 1. ADF тест
adf = adfuller(brent_price_data['Close'], autolag='AIC')
print()
print("ADF тест (H_0: нестационарен):")
print(f"Статистика: {adf[0]:.6f}")
print(f"p-value: {adf[1]:.6f}")
print("Вывод:")
if adf[1] < 0.05:
  print("Стационарен (alpha=0.05)")
else:
  print("Нестационарен (alpha=0.05)")

# 2. KPSS тест
kpss_result = kpss(brent_price_data['Close'], regression='c', nlags='auto')
print()
print("KPSS тест (H_0: стационарен):")
print(f"Статистика: {kpss_result[0]:.6f}")
print(f"p-value: {kpss_result[1]:.6f}")
print("Вывод:")
if kpss_result[1] > 0.05:
  print("Стационарен (alpha=0.05)")
else:
  print("Нестационарен (alpha=0.05)")
print()
```

Результат:




```
ADF тест (H_0: нестационарен):
Статистика: -1.910909
p-value: 0.326947
Вывод:
Нестационарен (alpha=0.05)

KPSS тест (H_0: стационарен):
Статистика: 2.148042
p-value: 0.010000
Вывод:
Нестационарен (alpha=0.05)

/tmp/ipython-input-18543015.py:37: InterpolationWarning: The test statistic is outside of the range of p-values available
in the
look-up table. The actual p-value is smaller than the p-value returned.

  kpss_result = kpss(brent_price_data['Close'], regression='c', nlags='auto')
```

Вывод

С помощью двух тестов (Дики-Фуллера и KPSS) было установлено, что гипотеза о том, что ряд стационарен, не подтверждена, и его можно считать нестационарным. Действительно - на графике видно, что цена на нефть случайно менялась с течением времени. Ее стоимость резко упала в 2015 и 2020 году и росла в 2011, 2017, 2022 году. Это свойственно для цен на нефть, логически связано с событиями в мире, а практически можно сделать вывод, что цены имеют стохастический тренд и не возвращаются к своему среднему значению без внешнего вмешательства.

№2

PYTHON

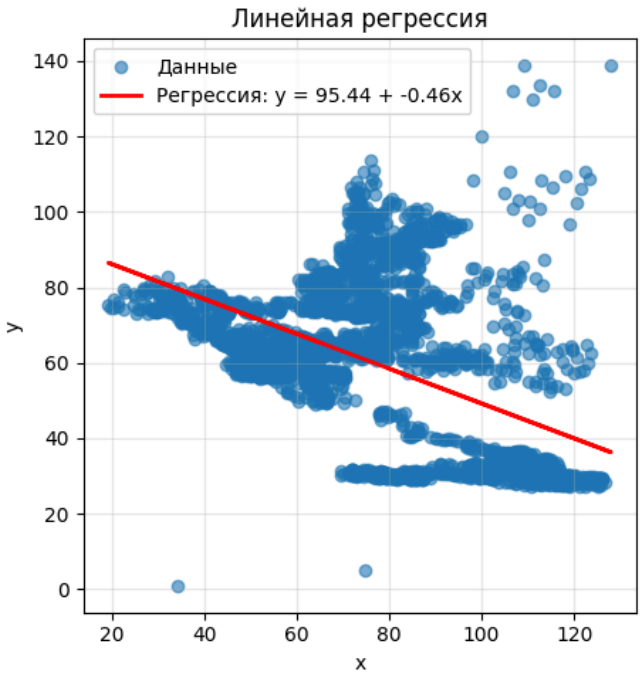
```python
import yfinance
import pandas
import matplotlib.pyplot as plt
import numpy as np
from statsmodels.tsa.stattools import adfuller, kpss
from scipy import stats
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.stats.diagnostic import acorr_ljungbox

# Цены
brent_price_data = yfinance.download('bz=f',start='2010-01-01',end='2025-10-31')
brent_price_data = pandas.DataFrame(brent_price_data)
rub_x = yfinance.download('rub=x',start='2010-01-01',end='2025-10-31')
rub_x = pandas.DataFrame(rub_x)

# Графики
plt.figure(figsize=(10, 5))
plt.plot(rub_x.index, rub_x['Close'], label='USD/RUB', linewidth=2)
plt.title('Курс доллара (USD) к рублю (RUB) 2010-2025', fontsize=14)
plt.xlabel('Время (год)')
plt.ylabel('Цена (RUB)')
plt.grid(True, linestyle='--', alpha=0.6)
plt.legend()
plt.tight_layout()
plt.show()
data = pandas.concat({ 'Brent': brent_price_data['Close'], 'USD_RUB': rub_x['Close']}, axis=1)
data = data.dropna()

x = data['Brent'].values
y = data['USD_RUB'].values
n = len(x)

# Оценки методом наименьших квадратов
x_mean = np.mean(x)
y_mean = np.mean(y)
beta1_eval = np.sum((x - x_mean) * (y - y_mean)) / np.sum((x - x_mean)**2)
beta0_eval = y_mean - beta1_eval * x_mean

# Предсказания и остатки
y_pred = beta0_eval + beta1_eval * x
resids = y - y_pred

# R^2
RSS = np.sum(resids**2)
TSS = np.sum((y - y_mean)**2)
R_sq = 1 - RSS/TSS

# Статистика
sigma_sq = RSS / (n - 2)
se_beta_1 = np.sqrt(sigma_sq / np.sum((x - x_mean)**2))
t_stat = beta1_eval / se_beta_1
p_value = 2 * (1 - stats.t.cdf(np.abs(t_stat), n - 2))
print()
print(f"Оценки МНК: ^beta_0 = {beta0_eval:.4f}, ^beta_1 = {beta1_eval:.4f}")
print(f"R^2 = {R_sq:.4f}")
print(f"t-статистика для beta_1: {t_stat:.4f}, p-value: {p_value:.6f}")
print()
```

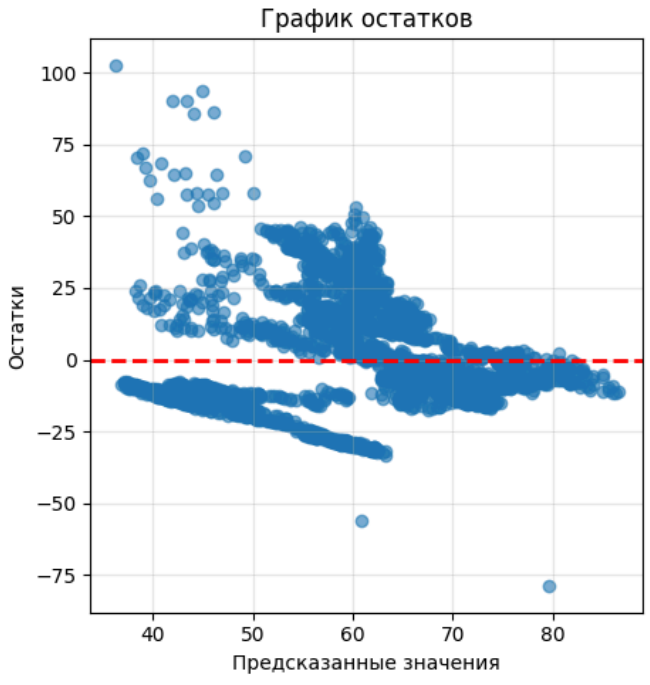
PYTHON

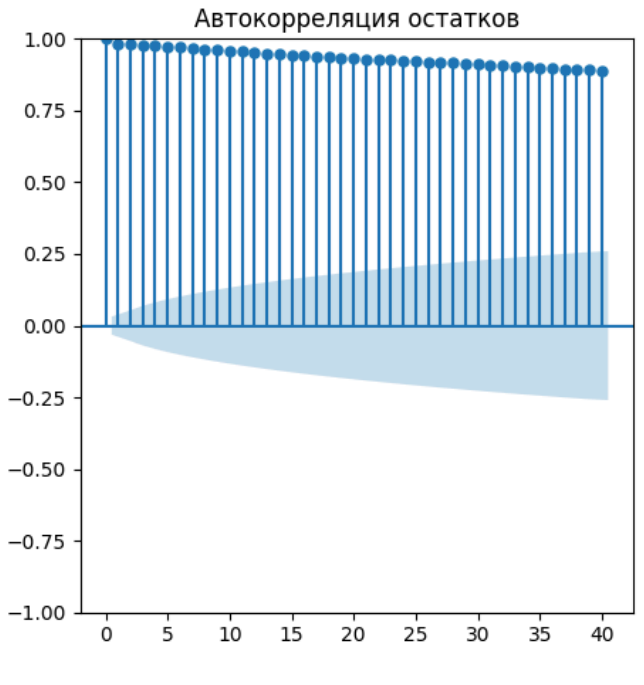
```python
# Визуализация
fig, axes = plt.subplots(1, 3, figsize=(14, 5))
axes[0].scatter(x, y, alpha=0.6, label='Данные')
axes[0].plot(x, y_pred, 'r-', linewidth=2, label=f'Регрессия: y = {beta0_eval:.2f} + {beta1_eval:.2f}x')
axes[0].set_xlabel('x')
axes[0].set_ylabel('y')
axes[0].set_title('Линейная регрессия')
axes[0].legend()
axes[0].grid(True, alpha=0.3)
axes[1].scatter(y_pred, resids, alpha=0.6)
axes[1].axhline(y=0, color='r', linestyle='--', linewidth=2)
axes[1].set_xlabel('Предсказанные значения')
axes[1].set_ylabel('Остатки')
axes[1].set_title('График остатков')
axes[1].grid(True, alpha=0.3)
plot_acf(resids, lags=40, alpha=0.05, ax=axes[2])
axes[2].set_title('Автокорреляция остатков')
plt.tight_layout()
plt.show()
plt.tight_layout()
plt.show()

# Тесты стационарности
print()
print("Тесты стационарности: остатки")

# 1. ADF тест
adf = adfuller(resids, autolag='AIC')
print()
print("ADF тест (H_0: нестационарен):")
print(f"Статистика: {adf[0]:.6f}")
print(f"p-value: {adf[1]:.6f}")
print("Вывод:")
if adf[1] < 0.05:
  print("Стационарен (alpha=0.05)")
else:
  print("Нестационарен (alpha=0.05)")

# 2. KPSS тест
kpss_result = kpss(resids, regression='c', nlags='auto')
print()
print("KPSS тест (H_0: стационарен):")
print(f"Статистика: {kpss_result[0]:.6f}")
print(f"p-value: {kpss_result[1]:.6f}")
print("Вывод:")
if kpss_result[1] > 0.05:
  print("Стационарен (alpha=0.05)")
else:
  print("Нестационарен (alpha=0.05)")

# Тест Льюнга-Бокса для всех процессов
max_lag = 20
print()
print("Тест Льюнга-Бокса: остатки")

lb = acorr_ljungbox(resids, lags=max_lag, return_df=True)
print()
print("Первые 10 лагов:")
print(lb.head(10).to_string())
significant = (lb['lb_pvalue'] < 0.05).sum()
print()
print("Количество значимых лагов (p < 0.05): ", significant, " из ", max_lag)
if significant == 0:
  print("Вывод: Нет значимой автокорреляции")
else:
  print("Вывод: Есть значимая автокорреляция")
```

Результат:


```
Оценки МНК: ^beta_0 = 95.4384, ^beta_1 = -0.4622
R^2 = 0.2595
t-статистика для beta_1: -37.1756, p-value: 0.000000
```







```
Тесты стационарности: остатки

ADF тест (H_0: нестационарен):
Статистика: -2.176885
p-value: 0.214742
Вывод:
Нестационарен (alpha=0.05)

KPSS тест (H_0: стационарен):
Статистика: 7.820948
p-value: 0.010000
Вывод:
Нестационарен (alpha=0.05)

Тест Льюнга-Бокса: остатки

Первые 10 лагов:
         lb_stat  lb_pvalue
1    3823.670272        0.0
2    7626.498600        0.0
3   11404.149604        0.0
4   15165.009579        0.0
5   18908.263852        0.0
6   22627.510324        0.0
7   26325.020536        0.0
8   29997.713381        0.0
9   33654.640229        0.0
10  37285.891711        0.0

Количество значимых лагов (p < 0.05):  20  из  20
Вывод: Есть значимая автокорреляция

/tmp/ipython-input-3002898788.py:98: InterpolationWarning: The test statistic is outside of the range of p-values
available in the
look-up table. The actual p-value is smaller than the p-value returned.

  kpss_result = kpss(resids, regression='c', nlags='auto')
```

Вывод

Эта регрессия выглядит так, словно курс рубля значимо зависит от цены на нефть (коэффициент значим, R^2=25,9%), однако зависимости на самом деле нет. Оба ряда нестационарные, случайно меняются со временем и не возвращаются к среднему без внешнего воздействия. Остатки модели также нестационарные (по тесту Дики-Фуллера и KPSS) и автокоррелированы (на третьем графике "автокорреляция остатков" значения не входят в область, а также по тесту Льюнга-Бокса). Все 20 лагов значимы (p=0,0) и присутствует сильная автокорреляция.

Метод наименьших квадратов показывает зависимость из-за того, что оба ряда меняются согласно тренду, а не потому что между ними есть реальная экономическая зависимость.

№3

PYTHON

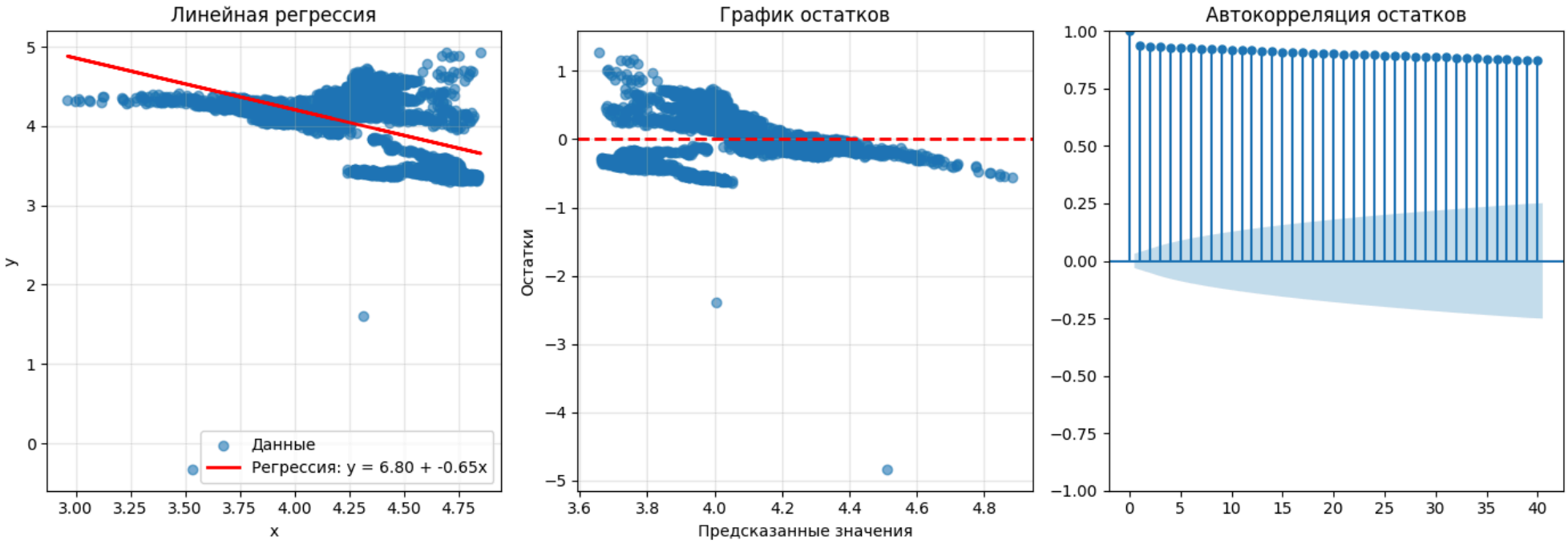
```python
import yfinance
import pandas
import matplotlib.pyplot as plt
import numpy as np
from statsmodels.tsa.stattools import adfuller, kpss
from scipy import stats
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.stats.diagnostic import acorr_ljungbox

# Цены
brent_price_data = yfinance.download('bz=f',start='2010-01-01',end='2025-10-31')
brent_price_data = pandas.DataFrame(brent_price_data)
rub_x = yfinance.download('rub=x',start='2010-01-01',end='2025-10-31')
rub_x = pandas.DataFrame(rub_x)

# Графики
plt.figure(figsize=(10, 5))
plt.plot(rub_x.index, rub_x['Close'], label='USD/RUB', linewidth=2)
plt.title('Курс доллара (USD) к рублю (RUB) 2010-2025', fontsize=14)
plt.xlabel('Время (год)')
plt.ylabel('Цена (RUB)')
plt.grid(True, linestyle='--', alpha=0.6)
plt.legend()
plt.tight_layout()

plt.show()
data = pandas.concat({ 'Brent': brent_price_data['Close'], 'USD_RUB': rub_x['Close']}, axis=1)
data = data.dropna()

# Логарифмирование
data['log_Brent'] = np.log(data['Brent'])
data['log_USD_RUB'] = np.log(data['USD_RUB'])

x = data['log_Brent'].values
y = data['log_USD_RUB'].values
n = len(x)

# МНК-оценки
x_mean = np.mean(x)
y_mean = np.mean(y)
beta1_eval = np.sum((x - x_mean) * (y - y_mean)) / np.sum((x - x_mean)**2)
beta0_eval = y_mean - beta1_eval * x_mean

# Предсказания и остатки
y_pred = beta0_eval + beta1_eval * x
resids = y - y_pred

# R^2
RSS = np.sum(resids**2)
TSS = np.sum((y - y_mean)**2)
R_sq = 1 - RSS/TSS

# Статистика
sigma_sq = RSS / (n - 2)
se_beta_1 = np.sqrt(sigma_sq / np.sum((x - x_mean)**2))
t_stat = beta1_eval / se_beta_1
p_value = 2 * (1 - stats.t.cdf(np.abs(t_stat), n - 2))
print()
print(f"Оценки МНК: ^beta_0 = {beta0_eval:.4f}, ^beta_1 = {beta1_eval:.4f}")
print(f"R^2 = {R_sq:.4f}")
print(f"t-статистика для beta_1: {t_stat:.4f}, p-value: {p_value:.6f}")
print()
```

PYTHON

```python
# Визуализация
fig, axes = plt.subplots(1, 3, figsize=(14, 5))
axes[0].scatter(x, y, alpha=0.6, label='Данные')
axes[0].plot(x, y_pred, 'r-', linewidth=2, label=f'Регрессия: y = {beta0_eval:.2f} + {beta1_eval:.2f}x')
axes[0].set_xlabel('x')
axes[0].set_ylabel('y')
axes[0].set_title('Линейная регрессия')
axes[0].legend()
axes[0].grid(True, alpha=0.3)
axes[1].scatter(y_pred, resids, alpha=0.6)
axes[1].axhline(y=0, color='r', linestyle='--', linewidth=2)
axes[1].set_xlabel('Предсказанные значения')
axes[1].set_ylabel('Остатки')
axes[1].set_title('График остатков')
axes[1].grid(True, alpha=0.3)
plot_acf(resids, lags=40, alpha=0.05, ax=axes[2])
axes[2].set_title('Автокорреляция остатков')
plt.tight_layout()
plt.show()
plt.tight_layout()
plt.show()

# Тесты стационарности
print()
print("Тесты стационарности: остатки")

# 1. ADF тест
adf = adfuller(resids, autolag='AIC')
print()
print("ADF тест (H_0: нестационарен):")
print(f"Статистика: {adf[0]:.6f}")
print(f"p-value: {adf[1]:.6f}")
print("Вывод:")
if adf[1] < 0.05:
  print("Стационарен (alpha=0.05)")
else:
  print("Нестационарен (alpha=0.05)")

# 2. KPSS тест
kpss_result = kpss(resids, regression='c', nlags='auto')
print()
print("KPSS тест (H_0: стационарен):")
print(f"Статистика: {kpss_result[0]:.6f}")
print(f"p-value: {kpss_result[1]:.6f}")
print("Вывод:")
if kpss_result[1] > 0.05:
  print("Стационарен (alpha=0.05)")
else:
  print("Нестационарен (alpha=0.05)")

# Тест Льюнга-Бокса для всех процессов
max_lag = 20
print()
print("Тест Льюнга-Бокса: остатки")
lb = acorr_ljungbox(resids, lags=max_lag, return_df=True)
print()
print("Первые 10 лагов:")
print(lb.head(10).to_string())
significant = (lb['lb_pvalue'] < 0.05).sum()
print()
print("Количество значимых лагов (p < 0.05): ", significant, " из ", max_lag)
if significant == 0:
  print("Вывод: Нет значимой автокорреляции")
else:
  print("Вывод: Есть значимая автокорреляция")
```

Результат:


```
Оценки МНК: ^beta_0 = 6.8024, ^beta_1 = -0.6487
R^2 = 0.2718
t-статистика для beta_1: -38.3616, p-value: 0.000000
```



```
Тесты стационарности: остатки

ADF тест (H_0: нестационарен):
Статистика: -2.242773
p-value: 0.191089
Вывод:
Нестационарен (alpha=0.05)

KPSS тест (H_0: стационарен):
Статистика: 8.442784
p-value: 0.010000
Вывод:
Нестационарен (alpha=0.05)

Тест Льюнга-Бокса: остатки

Первые 10 лагов:
    lb_stat         lb_pvalue
1   3450.891100     0.0
2   6887.183585     0.0
3   10310.904756    0.0
4   13725.467955    0.0
5   17128.844306    0.0
6   20521.761468    0.0
7   23900.572581    0.0
8   27265.933967    0.0
9   30625.115303    0.0
10  33966.750648    0.0

Количество значимых лагов (p < 0.05):  20  из  20
Вывод: Есть значимая автокорреляция

/tmp/ipython-input-108762488.py:107: InterpolationWarning: The test statistic is outside of the range of p-values
available in the
look-up table. The actual p-value is smaller than the p-value returned.

  kpss_result = kpss(resids, regression='c', nlags='auto')
```

Вывод

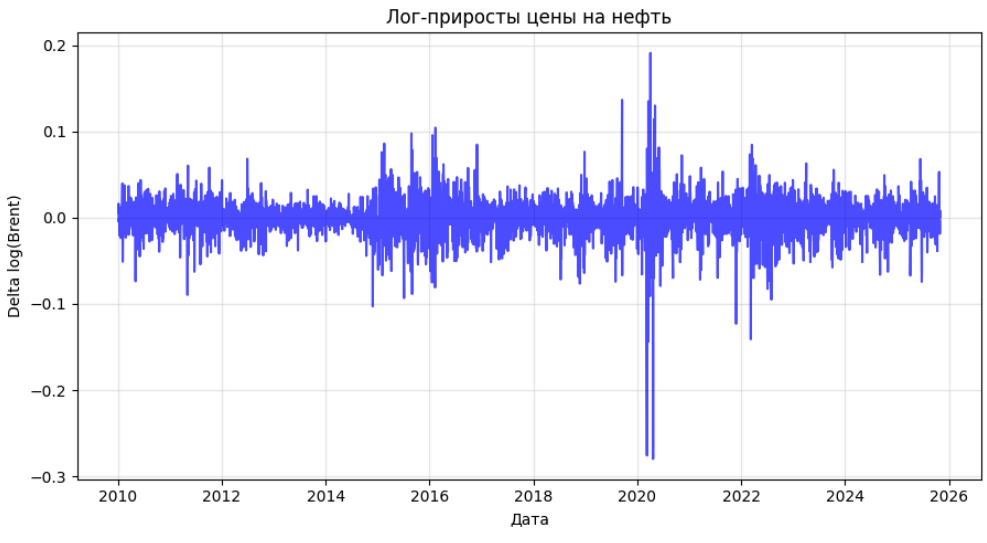
Регрессии (как в уровнях, так и в логарифмах) показывают значимую зависимость рубля от цены на нефть (чем выше цена на нефть, тем выше курс доллара и слабее рубль), однако зависимости опять же нет. Остатки моделей снова получились нестационарными (по тесту Дики-Фуллера и KPSS) и сильно автокоррелированными (20 из 20 лагов значимы по тесту Льюнга-Бокса и по третьему графику). Долгосрочного равновесия между курсом и ценой на нефть нет.

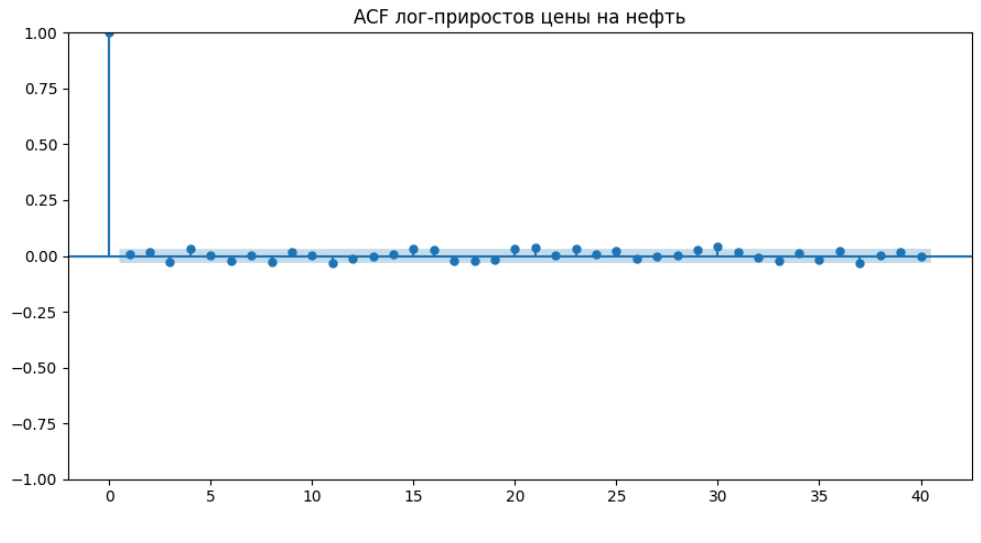
№4

PYTHON

```python
import yfinance
import pandas
import matplotlib.pyplot as plt
import numpy as np
from statsmodels.tsa.stattools import adfuller, kpss
from scipy import stats
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.stats.diagnostic import acorr_ljungbox

# Цены
brent_price_data = yfinance.download('bz=f',start='2010-01-01',end='2025-10-31')
brent_price_data = pandas.DataFrame(brent_price_data)

# Лог-приросты
brent_price_data['log_Close'] = np.log(brent_price_data['Close'])
brent_price_data['log_return'] = brent_price_data['log_Close'].diff()
log_returns = brent_price_data['log_return'].dropna()
fig, axes = plt.subplots(1, 2, figsize=(18,5))

# График лог-приростов
axes[0].plot(log_returns.index, log_returns.values, color='blue', alpha=0.7)
axes[0].set_title('Лог-приросты цены на нефть')
axes[0].set_xlabel('Дата')
axes[0].set_ylabel('Delta log(Brent)')
axes[0].grid(True, alpha=0.3)

# График автокорреляции (ACF)
plot_acf(log_returns, lags=40, alpha=0.05, ax=axes[1])
axes[1].set_title('ACF лог-приростов цены на нефть')
plt.tight_layout()
plt.show()

# Тест Льюнга-Бокса для всех процессов
max_lag = 20
print()
print("Тест Льюнга-Бокса: лог-прирост")
lb = acorr_ljungbox(log_returns, lags=max_lag, return_df=True)
print()
print("Первые 10 лагов:")
print(lb.head(10).to_string())
significant = (lb['lb_pvalue'] < 0.05).sum()
print()
print("Количество значимых лагов (p < 0.05): ", significant, " из ", max_lag)
if significant == 0:
  print("Вывод: Нет значимой автокорреляции")
else:
  print("Вывод: Есть значимая автокорреляция")
```

Результат:





```
Тест Льюнга-Бокса: лог-прирост
Первые 10 лагов:
    lb_stat     lb_pvalue
1   0.117814    0.731418
2   1.042361    0.593819
3   3.838204    0.279474
4   7.889677    0.095704
5   7.902603    0.161686
6   10.181262   0.117223
7   10.196720   0.177697
8   13.454535   0.097138
9   14.610741   0.102198
10  14.626982   0.146265

Количество значимых лагов (p < 0.05): 5 из 20
Вывод: Есть значимая автокорреляция
```

Вывод

Первые 15 лагов незначимы при уровне значимости 0,05, а 16-20 лаги значимые. Всего получается 5 из 20 значимых лагов - значимая автокорреляция присутствует, однако она слабая и отложенная. Такая автокорреляция на больших лагах может вызываться случайно, особенно на большой выборке, поэтому не указывает на серьезную зависимость - лог-приросты цены на нефть близки к белому шуму, и предыдущие изменения цены не помогают предсказать следующие. В принципе, это типичное поведение для финансовых временных рядов.

В целом лог-приросты цены на нефть можно считать приближенно стационарным рядом, пригодным для регрессионного анализа. Для самих цен этого сказать нельзя.

Рефлексия

в ходе выполнения основной сложностью было разобраться в библиотеках. я никогда раньше не использовала эти библиотеки python, да и в целом редко пишу код, поэтому я наелась ошибок в процессе применения библиотек. и еще была трудность в том, чтобы выгрузить сами данные, но одногруппники подсказали, что в питоне есть библиотека yfinance, которая позволяет это сделать, и поделились материалами, которые прислал семинарист 5-й группы.
еще я сначала пыталась делать это локально на своем ноутбуке, но поставить библиотеки python для матстата или нейронок на мак это всегда пытка. пришлось перейти в colab.
с кодом очень помог chat gpt - он отвечал на мои вопросы по применению библиотек и помогал искать ошибки. сам он код генерировал очень перегруженный и анализировал кучу показателей одновременно, и вникать в это было долго и сложно - он явно написал больше, чем требовали в задании, поэтому я решила не пытаться просить его сделать мне примеры и решила сочетать свои усилия с гуглингом, точечными вопросами и рассуждениями одногруппников. вроде получилось.