

**Михаил Шардин**

личный блог



11 марта 2025, 04:45

[+ Подписаться](#)

## Тестирование торговой стратегии с использованием нового индикатора Джона Ф. Элерса на Python для дневных данных Московской биржи

Торговля акциями требует гибкости, особенно когда речь идет о тестировании стратегий технического анализа на прошлых данных. Я выбрал Python и библиотеки `backtesting.py` и `aiomex`, потому что они позволяют анализировать рынок без сложных платформ и ограничений. Python дает свободу автоматизации, `backtesting.py` обеспечивает удобный и быстрый механизм тестирования стратегий, а `aiomex` позволяет скачивать данные напрямую с Московской биржи без привязки к брокеру.

Важно, что `backtesting.py` получил обновление после четырех лет без обновлений, что делает его актуальным инструментом. И в отличие от MetaTrader, StockSharp, TSLab и Quik, которые работают с Московской биржей, но требуют Windows, если брокер имеет API, то можно запускать скрипт на любом сервере, включая облачные решения и Raspberry Pi.

В этой статье я протестирую самую свежую стратегию теханализа Джона Ф. Элерса (John Ehlers), направленную на устранение запаздывания скользящей средней. Разберемся, как её адаптировать к акциям Московской биржи и протестировать с помощью Python.

### Новый индикатор Джона Элерса «устранение запаздывания скользящей средней»

Одна из главных проблем стандартных скользящих средних (SMA) — это запаздывание. Поскольку SMA рассчитывается как среднее за определенный период, её значение всегда отстает от реальной цены, что мешает своевременному входу в сделку.

Джон Элерс предложил решение — прогнозируемая скользящая средняя (PMA, Projected Moving Average). В отличие от обычных скользящих, PMA использует линейную регрессию для прогнозирования будущих значений, уменьшая лаг.

**БИЛЕТОВ  
ВСЕ МЕНЬШЕ**[УСПЕТЬ КУПИТЬ](#)

Введите текст комментария



SBER

Формула

PMA: `PMA = SMA + Slope * Length / 2,`

`<br />`где Slope — наклон линии регрессии.

Дополнительно Элерс предложил прогнозировать саму PMA: `<br /><code>PredictPMA = PMA + 0.5 (Slope - Slope[2]) Length`

и наклон: `<br /><code class="inline-code">PredictSlope = 1.5 Slope - 0.5 Slope[4].`

Пересечения PredictPMA и PMA помогают находить точки входа и выхода, делая стратегию более адаптивной к изменениям рынка.



S&P 500 E-Mini Futures

Стратегия на основе индикатора PMA Джона Ф. Элерса

**БИЛЕТОВ  
ВСЕ МЕНЬШЕ**



**УСПЕТЬ КУПИТЬ**

– 10-дневная РМА выше 50-дневной.

Риск-менеджмент:

- Первоначальный стоп-лосс устанавливается на 10% ниже цены входа.
- Выход из позиции осуществляется по скользящему стопу на основе ATR.

## Реализация бэктестинга через `backtesting.py`. Определение топ-20 акций по объему

Весь код [представлен на GitHub](#).

### Модуль `data_loader.py`

```

1  # Отвечает за загрузку данных с Мосбиржи
2
3  import pandas as pd
4  import aiohttp
5  from datetime import datetime, timedelta
6  import aiomoe
7  import asyncio
8
9  async def fetch_moex_data(ticker, timeframe='D', days=1825):
10     """
11     Асинхронная загрузка свечных данных с Московской биржи через aiomoe.
12     """
13     # Определяем конечную и начальную даты
14     end_date = datetime.now()
15     start_date = end_date - timedelta(days=days)
16
17     # Приводим к строковому формату YYYY-MM-DD
18     start_str = start_date.strftime('%Y-%m-%d')
19     end_str = end_date.strftime('%Y-%m-%d')
20
21     # Мappings интервалов
22     interval_map = {'D': 24, 'W': 7, 'H': 60, 'M': 31}
23     interval = interval_map.get(timeframe, 24) # По умолчанию дневные свечи
24
25     print(f"Загружаем данные для {ticker} с {start_str} по {end_str} (таймфрейм: {timeframe})")
26

```

Для тестирования стратегии необходимо загружать актуальные данные о торгах.

В этом помогает библиотека `aiomoe`, которая предоставляет API-доступ к Московской бирже. В модуле `data_loader.py` реализована функция `fetch_moex_data`, позволяющая асинхронно получать исторические данные по свечам.

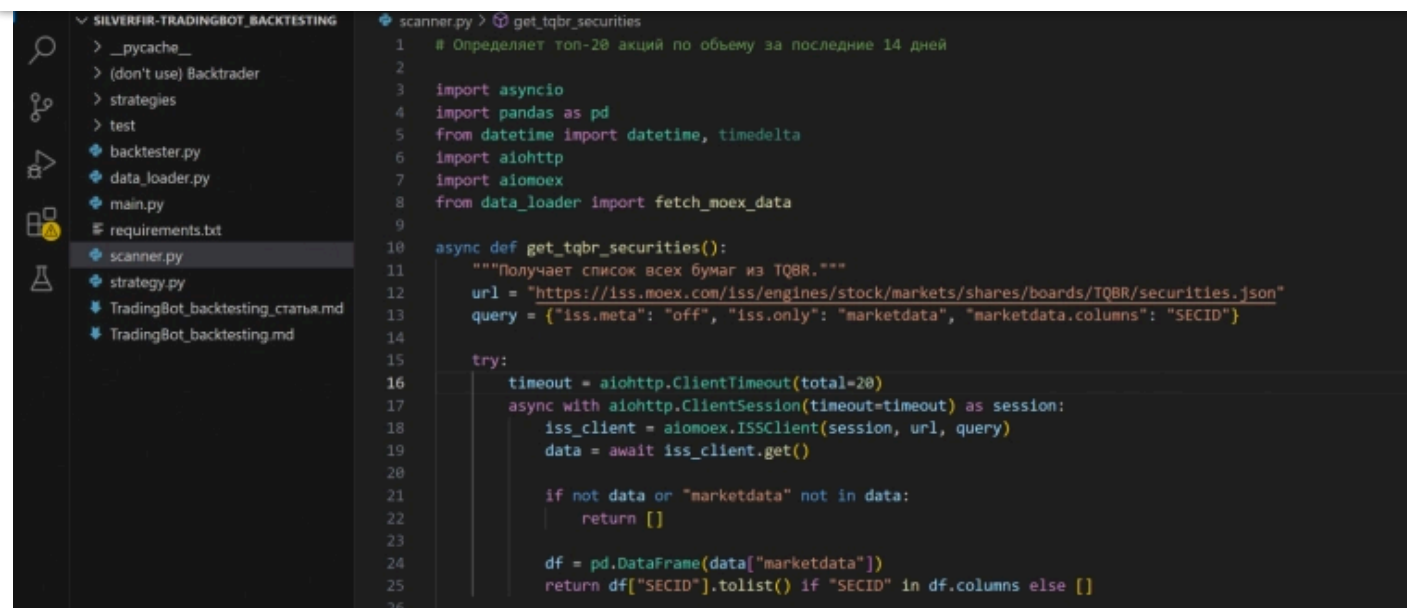
Функция запрашивает данные за последние 1825 дней (примерно 5 лет) и конвертирует их в формат Pandas DataFrame. Особенность реализации — использование асинхронного HTTP-клиента `aiohttp`, что ускоряет загрузку. Данные приводятся к удобному формату: преобразуются даты, устанавливается индекс, а названия колонок заменяются на стандартные для анализа.

Фильтрация ликвидных ордера для тестирования. Модуль `scanner.py`

**БИЛЕТОВ  
ВСЕ МЕНЬШЕ**



**УСПЕТЬ КУПИТЬ**



```

1  # Определяет топ-20 акций по объему за последние 14 дней
2
3  import asyncio
4  import pandas as pd
5  from datetime import datetime, timedelta
6  import aiohttp
7  import aiomoex
8  from data_loader import fetch_moex_data
9
10 async def get_top_20_stocks():
11     """Получает список всех бумаг из TQBR."""
12     url = "https://iss.moex.com/iss/engines/stock/markets/shares/boards/TQBR/securities.json"
13     query = {"iss.meta": "off", "iss.only": "marketdata", "marketdata.columns": "SECID"}
14
15     try:
16         timeout = aiohttp.ClientTimeout(total=20)
17         async with aiohttp.ClientSession(timeout=timeout) as session:
18             iss_client = aiomoex.ISSClient(session, url, query)
19             data = await iss_client.get()
20
21             if not data or "marketdata" not in data:
22                 return []
23
24             df = pd.DataFrame(data["marketdata"])
25             return df["SECID"].tolist() if "SECID" in df.columns else []
26

```

После загрузки данных важно отобрать ликвидные бумаги. Для этого в модуле `scanner.py` реализована функция `get_top_20_stocks`, которая анализирует объем торгов за последние 14 дней и выделяет 20 наиболее ликвидных акций.

Алгоритм работы следующий:

1. Получение списка всех торгуемых акций на основном рынке (TQBR) через API Московской биржи.
2. Асинхронная загрузка дневных данных по каждому инструменту с помощью `fetch_moex_data`.
3. Расчет суммарного объема торгов за 14 дней.
4. Формирование списка из 20 акций с наибольшим объемом.

Таким образом, отбираются бумаги с высоким оборотом, что повышает надежность тестирования стратегии и снижает риск торговли неликвидными активами.

## Реализация бэктестинга через `backtesting.py`. Тестирование стратегии на исторических данных

### Зачем всё разделил на модули?

Разделение кода на модули делает его более удобным для сопровождения, масштабирования и переиспользования. В нашем случае:

- `data_loader.py` отвечает за загрузку данных с Московской биржи.
- `scanner.py` фильтрует ликвидные бумаги.
- `backtester.py` выполняет бэктестинг.
- `strategy.py` содержит описание стратегии.

**БИЛЕТОВ  
ВСЕ МЕНЬШЕ**



**УСПЕТЬ КУПИТЬ**

## Пример кода для бэктестинга с использованием backtesting.py

```
<code>import asyncio
import pandas as pd
from backtesting import Backtest
from data_loader import fetch_moex_data
from strategy import LongOnlyPMAMultiTimeframeATRTrailingStop

async def run_backtest(ticker):
    print(f"\n{'='*50}")
    print(f"🚀 Запуск бэктеста для {ticker}")
    print(f"{'='*50}\n")

    # Получаем данные
    df, start_str, end_str = await fetch_moex_data(ticker) # Получаем start_str

    # Запускаем бэктест
    print("⌚ Запуск бэктеста...")
    strategy_class = LongOnlyPMAMultiTimeframeATRTrailingStop # Класс стратегии
    strategy_name = f"{ticker}_{start_str}_{end_str}_LongOnlyPMAMultiTimeframeAT
    DynamicStrategyClass = type(strategy_name, (strategy_class,), {}) # Создаем
    bt = Backtest(df, DynamicStrategyClass, cash=100_000, commission=0.002) # Ис
    stats = bt.run()

    # Вывод результатов
    print("\n📊 Результаты бэктеста:")
    print(f"⚙️ Стратегия: {strategy_name}") # Выводим динамическое имя стратегии
    print(f"📅 Период тестирования: с {stats['Start']} по {stats['End']}")
    print(f"💰 Начальный капитал: 100,000 руб.")
    print(f"💵 Конечный капитал: {stats['Equity Final ($)']:.2f} руб.")
    print(f"📈 Общая доходность: {stats['Return [%]']:.2f}%")
    print(f"📊 Годовая доходность: {stats['Return (Ann.) [%]']:.2f}%")
    print(f"📉 Коэффициент Шарпа: {stats['Sharpe Ratio']:.2f}")
    print(f"📉 Максимальная просадка: {stats['Max. Drawdown [%]']:.2f}%")
    print(f"🔄 Количество сделок: {stats['# Trades']}")
    print(f"✅ Процент выигрышных сделок: {stats['Win Rate [%]']:.2f}%")
    print(f"📉 Лучшее соотношение (stats['Best Trade [%]']:.2f)%")
```

**БИЛЕТОВ  
ВСЕ МЕНЬШЕ**



**УСПЕТЬ КУПИТЬ**

```
print("\n 📊 Построение графика результатов...")
try:
    bt.plot()
    print(" ✅ График успешно построен!")
except ValueError as e:
    print(f" ❌ Ошибка при построении графика: {e}")
print(f"\n{'='*50}")
print(f" 🏁 Бэктест для {ticker} завершен")
print(f"\n{'='*50}")
return stats
```

### Основные метрики оценки

Для оценки стратегии используются ключевые метрики:

- Общая доходность (%) — показывает, сколько стратегия заработала за весь период тестирования.
- Годовая доходность (%) — усреднённый годовой прирост капитала.
- Коэффициент Шарпа — измеряет соотношение доходности к риску.
- Максимальная просадка (%) — определяет максимальную потерю капитала.
- Процент выигрышных сделок.
- Средняя продолжительность сделки.

Эти показатели позволяют оценить эффективность стратегии и принять решение о её использовании в реальной торговле.

### Результаты тестирования на акциях Московской биржи

Как положительные, так и отрицательные.

Примеры в [html файлах на GitHub'e](#).

**СПБ Биржа (тикер SPBE):**

**БИЛЕТОВ  
ВСЕ МЕНЬШЕ**



**УСПЕТЬ КУПИТЬ**





Результаты бэктеста:

Стратегия: SPBE\_2020-03-03\_2025-03-02\_LongOnlyPMAMultiTimeframeATRTrailingStop

Период тестирования: с 2021-11-19 00:00:00 по 2025-03-01 00:00:00

Начальный капитал: 100,000 руб.

Конечный капитал: 349138.97 руб.

Общая доходность: 249.14%

Годовая доходность: 47.34%

Коэффициент Шарпа: 0.80

Максимальная просадка: -27.41%

Количество сделок: 11

Процент выигрышных сделок: 54.55%

Лучшая сделка: +36.56%

Худшая сделка: -8.31%

Средняя продолжительность сделки: 25 days 00:00:00

Новатэк ао (тикер NVTK):

**БИЛЕТОВ  
ВСЕ МЕНЬШЕ**



**УСПЕТЬ КУПИТЬ**



Результаты бэктеста:

Стратегия: NVTK\_2020-03-03\_2025-03-02\_LongOnlyPMAMultiTimeframeATRTrailingStop

Период тестирования: с 2020-03-03 00:00:00 по 2025-03-01 00:00:00

Начальный капитал: 100,000 руб.

Конечный капитал: 94443.56 руб.

Общая доходность: -5.56%

Годовая доходность: -1.15%

Коэффициент Шарпа: -0.07

Максимальная просадка: -38.70%

Количество сделок: 22

Процент выигрышных сделок: 27.27%

Лучшая сделка: +18.55%

Худшая сделка: -10.78%

Средняя продолжительность сделки: 23 days 00:00:00

## Проблема учета смены лидеров по объему

Мой код отдельно тестирует каждую акцию из топ-20 на момент отбора (на сегодня).

Однако он не учитывает смену лидеров по объему и не позволяет работать с единой

корзиной акций, где позиции могут удерживаться даже после выпадения бумаги из топ-20

**БИЛЕТОВ  
ВСЕ МЕНЬШЕ**



**УСПЕТЬ КУПИТЬ**



при котором состав портфеля регулярно пересматривается и обновляется на основе новых данных.

## Следующие шаги

Фиксированный список топ-акций по объему устаревает. Использование динамического реестра позволит оперативно учитывать смену лидеров, корректируя состав активных позиций в стратегии.

Библиотека `ta-lib` мне не очень понравилась из-за сложностей с установкой — проще переписать индикатор вручную в будущем.

Получится ли реализовать это через `backtesting.py`? Скорее всего вряд ли.

Скорее всего придётся вернуться к `Backtrader`.

## Заключение

Тестирование стратегии на акциях Мосбиржи показало её стабильную эффективность при использовании индикатора РМА на дневных свечах.

Python доказал свою ценность в алгоритмической торговле, обеспечивая гибкость и автоматизацию. Однако `backtesting.py` имеет ограничения.

Автор: Михаил Шардин

 [Моя онлайн-визитка](#)

 [Telegram «Умный Дом Инвестора»](#)

11 марта 2025 г.

Python

торговые роботы

4K 

☆ 15

💬 29

❤️ 24



**Михаил Шардин**

📍 Пермь

👤 161 📊 1 414

📅 с 23 января 2019

🔗 +HreHDn1F5CZjN...

+ Подписаться

**БИЛЕТОВ  
ВСЕ МЕНЬШЕ**



**УСПЕТЬ КУПИТЬ**

Не работают  
backtesting.py => backtesting.ru?

— Показать 1 ответ



Михаил Шардин

11 марта 2025, 08:35



Сергей Сергаев, например в каком направлении? Одним словом



Noname

11 марта 2025, 08:57



Решение запаздывания скользящей средней решается уменьшением таймфрейма, у Элиота в то время этой возможности не было, у нас есть.

15та на часовом таймфрейме равна 900та на минутном таймфрейме. Проблема решена)

— Показать 3 ответа



Михаил Шардин

11 марта 2025, 09:11



Сергей Сергаев, вы имеете ввиду что если стратегия показывала хорошие результаты на исторических данных, это не означает, что она будет работать в будущем.

Но я-то думал что единственное слово будет что-то вроде «фундаментал».



Ещё 11 комментариев


Напишите комментарий...

БИЛЕТОВ  
ВСЕ МЕНЬШЕ



УСПЕТЬ КУПИТЬ

Установите приложение Смартлаба:

 RuStore

 AppGallery

 App Store



[О смартлабе](#)

[Реклама](#)

[Полная версия](#)



Московская Биржа является спонсором ресурса smart-lab.ru

**БИЛЕТОВ  
ВСЕ МЕНЬШЕ**



**УСПЕТЬ КУПИТЬ**