Михаил Шардин
личный блог

19 марта 2025, 04:44

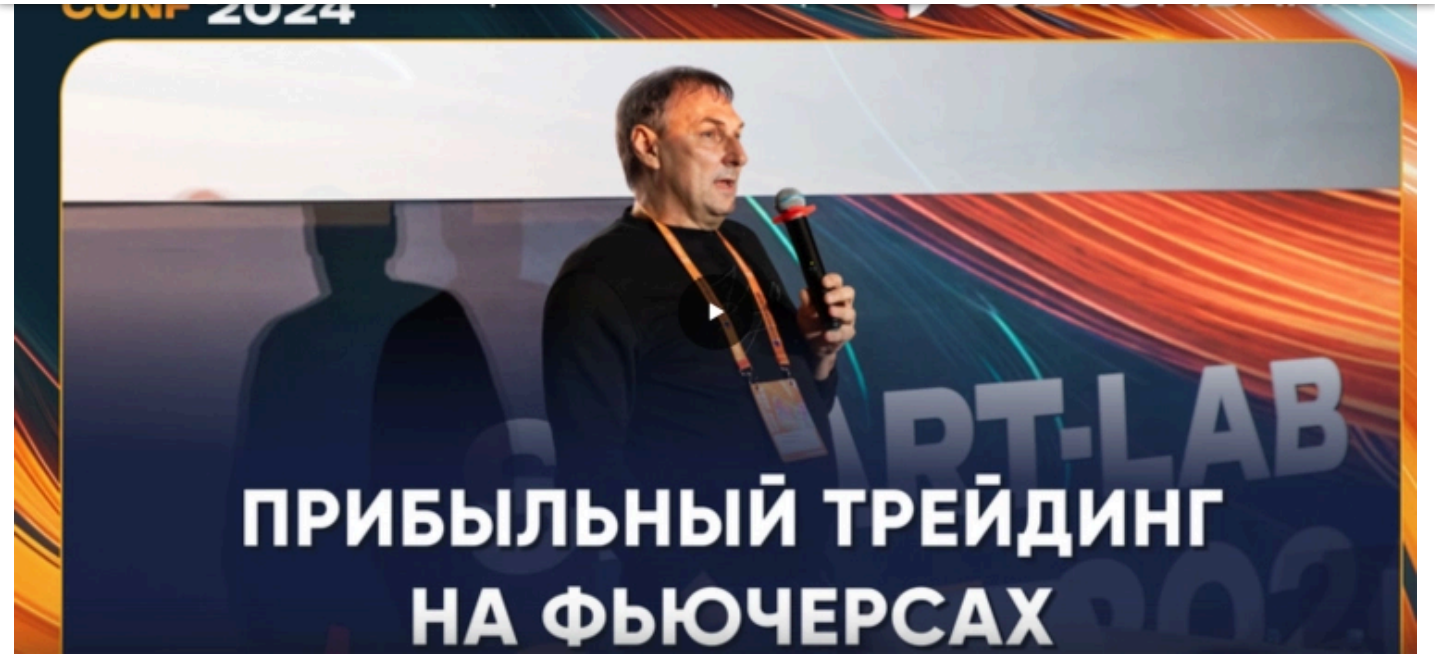
Тестировании торговой системы Александра Резвякова для фьючерсов Московской биржи с использованием Python

В этой статье расскажу о том, как воспроизвел и протестировал торговую систему для фьючерсов Московской биржи, основанную на идеях Александра Резвякова. Недавно, просматривая раздел алготрейдинга на Смартлабе, я наткнулся на видео с его выступления на конференции 2024 года под названием "5-6 идей для построения прибыльной торговой системы на фьючерсах". Меня привлекла четкость и понятность предложенных им правил торговли.

Поскольку я активно занимаюсь автоматизацией процессов и стремлюсь глубже изучить возможности Python библиотеки backtesting.py, мне показалось это хорошей идеей для практического применения.

Хотя я лично не знаком с Александром, полагаю, что публичное представление идеи предполагает возможность её независимого анализа и тестирования сообществом трейдеров и программистов.

Введите текст комментария



Обзор стратегии Александра Резвякова на фьючерсах

Основная идея — открывать сделки в строго определенное время и использовать структуру рынка последних дней для принятия решений.

Правила входа

- Время входа: сделки совершаются только в 10:00.

Лонг:

- Должно быть не менее трех дневных свечей в наличии.
- За последние два дня минимумы (Low) и максимумы (High) должны повышаться.
- Вероятность входа — 50%.

Шорт:

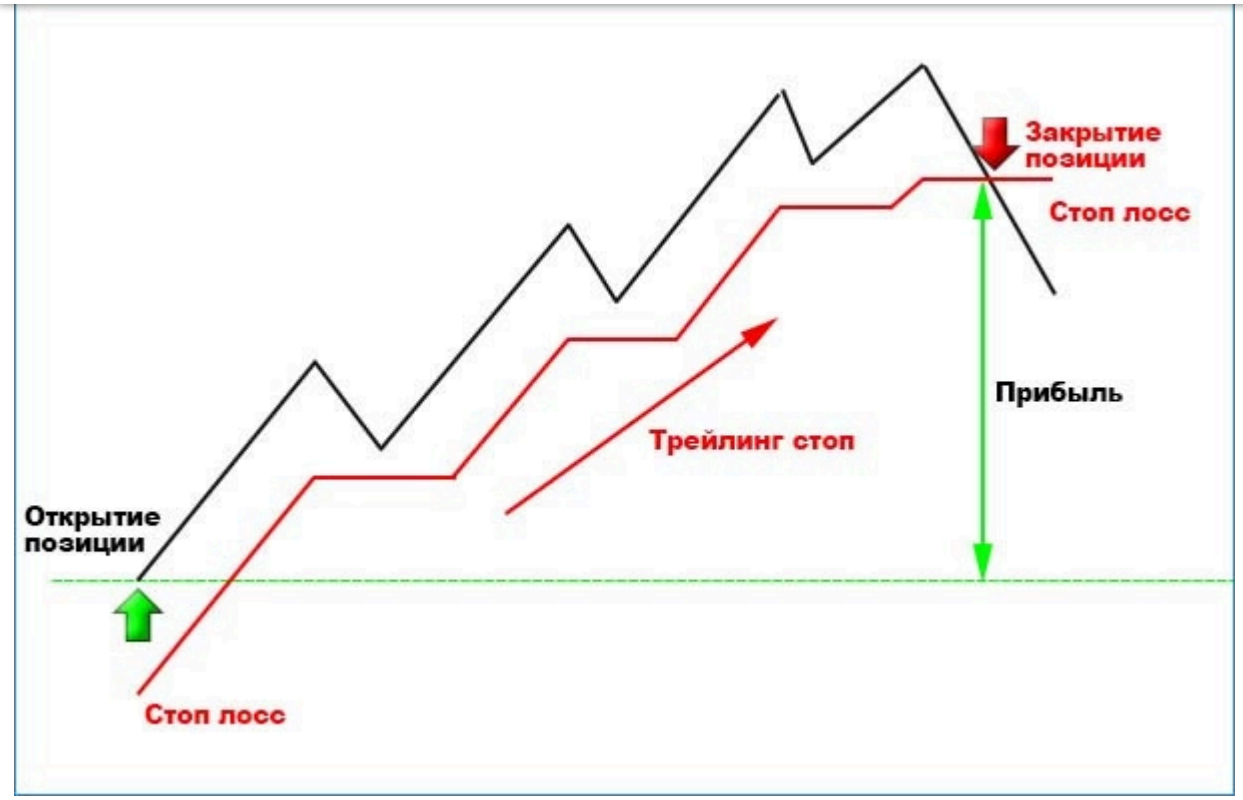
- Также минимум три дневных бара.
- За последние два дня максимумы и минимумы снижаются.
- Вероятность входа — 50%.

Правила выхода

- По времени: позиция закрывается в 20:40, независимо от результата.

По трейлинг-стопу:

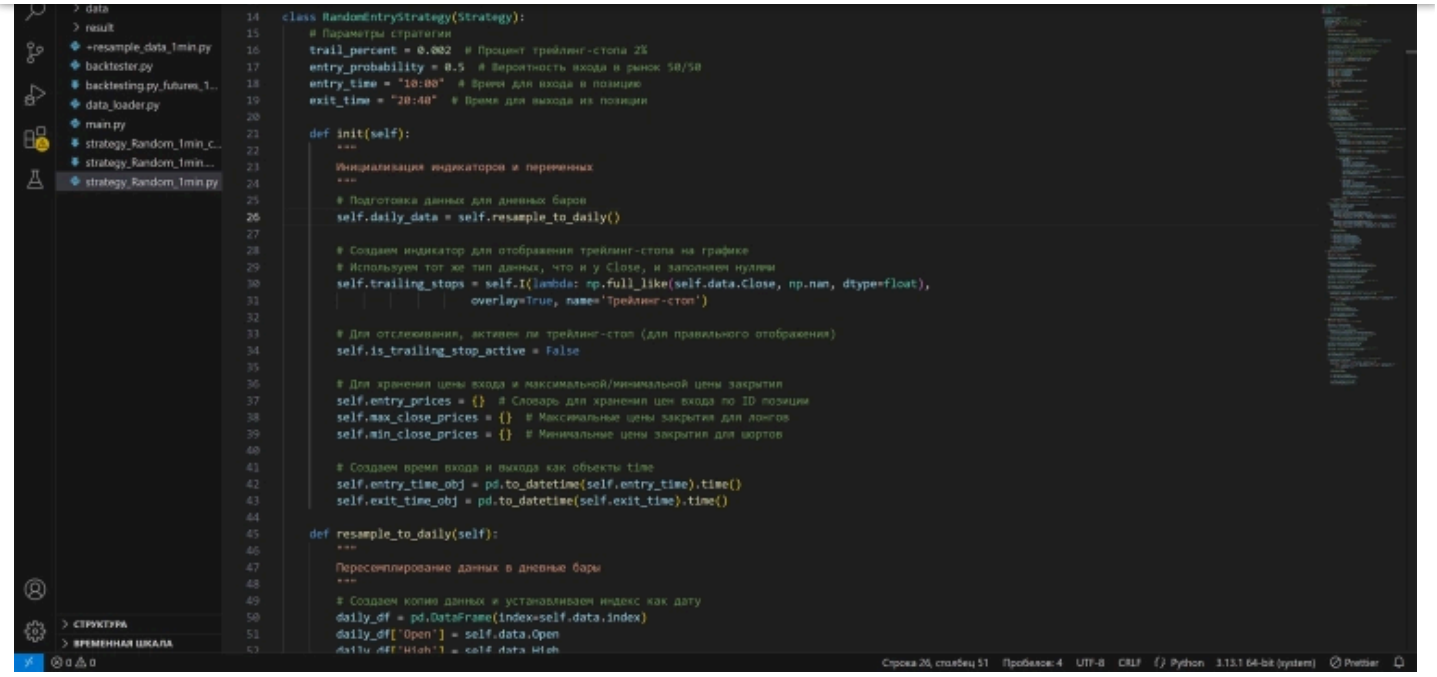
- Для ЛОНГА: если цена закрытия предыдущего бара падает больше чем на 2% от уровня покупки, то продажа. Если цена закрытия бара увеличивается, то этот уровень 2% убытка отчитывается уже от этой нового максимального уровня цены закрытия бара.
- Для ШОРТА: обратно лонгу — если цена растет на определенный процент от



Для лонга

Подготовка среды для тестирования

Для реализации и тестирования стратегии Александра Резвякова я выбрал Python и библиотеку backtesting.py. Этот выбор обусловлен несколькими факторами: Python предоставляет гибкие возможности для работы с данными, а backtesting.py позволяет эффективно моделировать торговые стратегии.



В своем проекте применил модульный подход, разделив функциональность на несколько основных компонентов:

- main.py — центральный модуль, запускающий весь процесс сканирования данных и бэктестирования стратегии
- resample_data_1min.py — отвечает за фильтрацию и преобразование данных, получаемых через API; выбирает только необходимый временной интервал котировок
- data_loader.py — преобразует данные из CSV-формата в структуру, пригодную для работы с backtesting.py
- backtester.py — содержит основную логику запуска и настройки бэктеста
- strategy_Random_1min.py — реализует саму торговую систему Резвякова с описанными правилами входа и выхода

Хотя в оригинальном видео Александр Резвяков рекомендовал работать на 5-минутном таймфрейме, я принял решение использовать 1-минутные свечи. Это позволяет более детально анализировать движение цены и точнее отслеживать срабатывание трейлинг-стопов. Благодаря более гранулярным данным, становится проще визуализировать и понять всю логику выхода из позиции, особенно в критических моментах, когда цена колеблется вблизи уровня стопа.

Шорт фьючерса CRH5 (CNY-3.25)

Полный исходный код всех модулей доступен в моем репозитории на GitHub для тех, кто захочет воспроизвести эксперимент или улучшить предложенную реализацию.

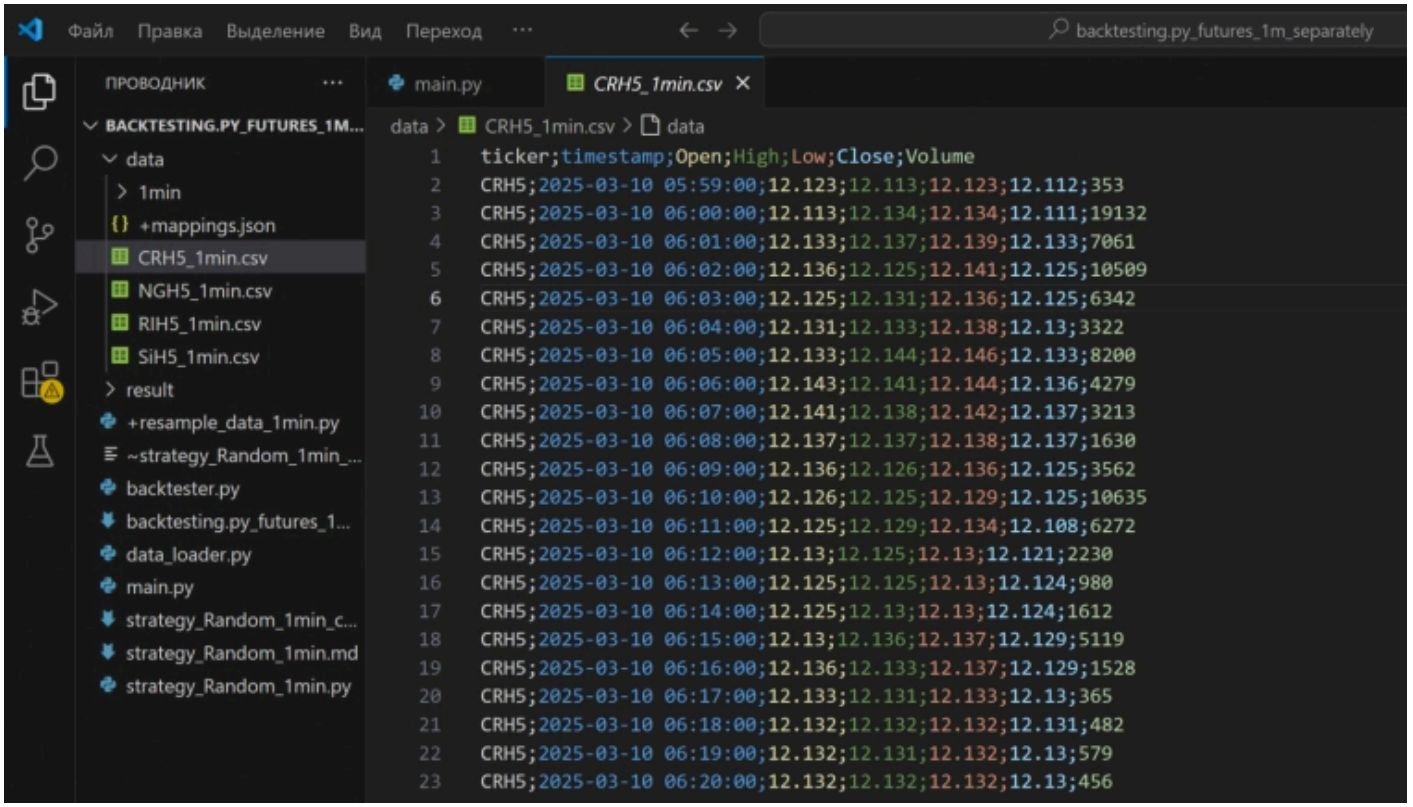
Реализация стратегии на Python с использованием backtesting.py

Вот подробное описание каждого шага с точки зрения программирования:

Подготовка данных:

Первым шагом является загрузка и предобработка исторических данных о фьючерсах. Модуль resample_data_1min.py отвечает за получение минутных данных с использованием API брокера (в данном случае, T-Invest API) и выборку необходимых временных интервалов. Он загружает данные из CSV-файлов, содержащих котировки, и фильтрует их по заданному диапазону дат. Ключевым моментом является функция filter_by_date, которая обеспечивает

готовые к дальнейшему использованию.



Загрузка данных из CSV:

Модуль data_loader.py отвечает за загрузку исторических данных из CSV-файлов и преобразование их в формат, пригодный для использования с библиотекой backtesting.py. Функция load_data_for_ticker считывает данные для указанного тикера, выполняет парсинг дат и устанавливает столбец timestamp в качестве индекса. Это позволяет backtesting.py правильно интерпретировать временные ряды.

Реализация условий для входа и выхода:

В модуле strategy_Random_1min.py реализована основная логика торговой стратегии. Класс RandomEntryStrategy наследуется от класса Strategy из библиотеки backtesting.py. В методе init происходит инициализация необходимых параметров и индикаторов.

Метод resample_to_daily пересчитывает минутные данные в дневные бары, необходимые для принятия решений о входе в позицию.

В методе next реализуются правила входа и выхода из позиции, описанные Александром Резвяковым. Время входа строго фиксировано (10:00), а решение о входе в лонг или шорт принимается на основе анализа предыдущих двух дневных баров. Вероятность входа в сделку устанавливается как 50%. Время выхода установлено на 20:40.

Интерактивный HTML

Настройка трейлинг-стопов:

Для реализации трейлинг-стопов в strategy_Random_1min.py используются методы update_long_position и update_short_position. В этих методах вычисляется уровень трейлинг-стопа на основе текущей цены и заданного процента (2%).

Трейлинг-стоп автоматически подтягивается вверх для длинных позиций и вниз для коротких, обеспечивая фиксацию прибыли и ограничение убытков. При достижении ценой уровня трейлинг-стопа позиция автоматически закрывается.

Тестирование стратегии

Библиотека backtesting.py позволяет не только моделировать сделки, но и визуализировать их в интерактивных HTML-отчетах. Это помогает детально анализировать точки входа и выхода, проверять корректность реализации логики и выявлять возможные ошибки в коде.

Чтобы избежать нагромождения данных в интерактивных HTML-отчетах решил загружать данные по неделям.

Кроме того, я задал комиссию в размере 0,2%, поскольку комиссии оказывают заметное влияние на итоговую доходность:

```
<code>
bt = Backtest(
    df,
    DynamicStrategyClass,
```

```
    hedging=False,
)
```

Так как стратегия использует случайный вход в сделки (вероятность 50% на покупку или продажу), результаты тестирования также носят случайный характер. Однако, уже на первых тестах удалось увидеть интересные закономерности.

Вот пример удачного теста на фьючерсе NGH5:

Шорт

```
<code>
==================================================
🚀 Запуск бэктеста для NGH5
==================================================


📋 Технические данные (для справки):
Колонки в данных: ['ticker', 'Open', 'High', 'Low', 'Close', 'Volume']
Первые 5 строк данных:
                    ticker   Open   High    Low  Close  Volume
timestamp
2025-03-10 05:59:00   NGH5  4.610  4.610  4.610  4.576    8743
2025-03-10 06:00:00   NGH5  4.610  4.605  4.629  4.605   10655
2025-03-10 06:01:00   NGH5  4.605  4.592  4.607  4.592    8855
2025-03-10 06:02:00   NGH5  4.592  4.597  4.598  4.592    5293
```

```
Open      float64
High      float64
Low       float64
Close     float64
Volume      int64
dtype: object


⏳ Запуск бэктеста...
ПРОДАЖА на 2025-03-12: Цена: 4.33, Стоп: 4.33
ВЫКУП (трейлинг-стоп) на 2025-03-12: Цена: 4.31, Стоп: 4.31, Вход: 4.33, Мин.зак
ПРОДАЖА на 2025-03-14: Цена: 4.08, Стоп: 4.09
ВЫКУП (трейлинг-стоп) на 2025-03-14: Цена: 4.02, Стоп: 4.02, Вход: 4.08, Мин.зак

📊 Результаты бэктеста:
⚙️ Стратегия: NGH5_RandomEntryStrategy
🗓️ Период тестирования: с 2025-03-10 05:59:00 по 2025-03-14 20:49:00
💰 Начальный капитал: 100,000 руб.
💵 Конечный капитал: 1104162.00 руб.
📈 Общая доходность: 10.42%
📊 Годовая доходность: 14652.07%
📈 Коэффициент Шарпа: 1.11
📉 Максимальная просадка: -6.51%
🔄 Количество сделок: 2
✅ Процент выигрышных сделок: 100.00%
💪 Лучшая сделка: +1.40%
😟 Худшая сделка: 0.44%
⏱️ Средняя продолжительность сделки: 0 days 00:40:00

📊 Построение графика результатов...
✅ График успешно построен!

==================================================
🏁 Бэктест для NGH5 завершен
==================================================
```

</code>

некоторых тестах убытки накапливались быстрее, чем прибыльные сделки перекрывали потери. В будущем стоит рассмотреть добавление дополнительных фильтров для входа.

Рекомендации по улучшению стратегии

Наращивание позиций в выигрышных сделках

Автор рекомендует увеличивать объем позиции в прибыльных сделках. Однако в текущей реализации теста этот момент не был учтен. Основная сложность в том, как правильно идентифицировать момент для добавления к позиции и реализовать это в backtesting.py.

Если у кого-то из читателей есть идеи или готовые решения, буду рад обсудить их.

Возможные улучшения стратегии

- Оптимизация трейлинг-стопа

  В текущей версии трейлинг-стоп фиксирован на уровне 2%, однако более гибкий подход, основанный на волатильности актива, может улучшить результаты. Например, можно использовать ATR (Average True Range) для динамического определения уровня стопа.

- Добавление фильтров для входа

  Вход в 10:00 производится без дополнительных условий, что делает стратегию чувствительной к случайным колебаниям рынка. Можно добавить индикаторы тренда (например, SMA, VWAP) или фильтры объема, чтобы подтверждать сигналы на вход.

- Добавление адаптивного модуля управления рисками

  Вместо фиксированного размера позиции можно вводить динамическое управление рисками на основе максимальной просадки (drawdown) или волатильности актива.

Все эти идеи требуют детального тестирования, но их внедрение может значительно повысить эффективность стратегии.

Заключение

В рамках этой статьи я воспроизвел и протестировал торговую систему Александра Резвякова на фьючерсах Московской биржи, используя Python и библиотеку backtesting.py.

Несмотря на кажущуюся простоту (случайные входы в 10:00 с вероятностью 50%), стратегия показала интересные результаты благодаря четким правилам выхода из позиций: трейлинг-стопы и закрытие по времени в 20:40.

Данный опыт может быть особенно полезен начинающим алгоритмическим трейдерам.

- Во-первых, он наглядно демонстрирует, что даже базовая стратегия со случайными входами может приносить прибыль при грамотном управлении рисками.
- Во-вторых, проект служит отличным примером структурирования кода для торговых

Что касается перспектив развития, система обладает значительным потенциалом для улучшения. Добавление динамических трейлинг-стопов на основе ATR, интеграция дополнительных фильтров для входа и реализация механизма наращивания позиций в прибыльных сделках могут существенно повысить эффективность стратегии. Python и backtesting.py предоставляют все необходимые инструменты для таких экспериментов, открывая широкие возможности для дальнейших исследований и оптимизации.

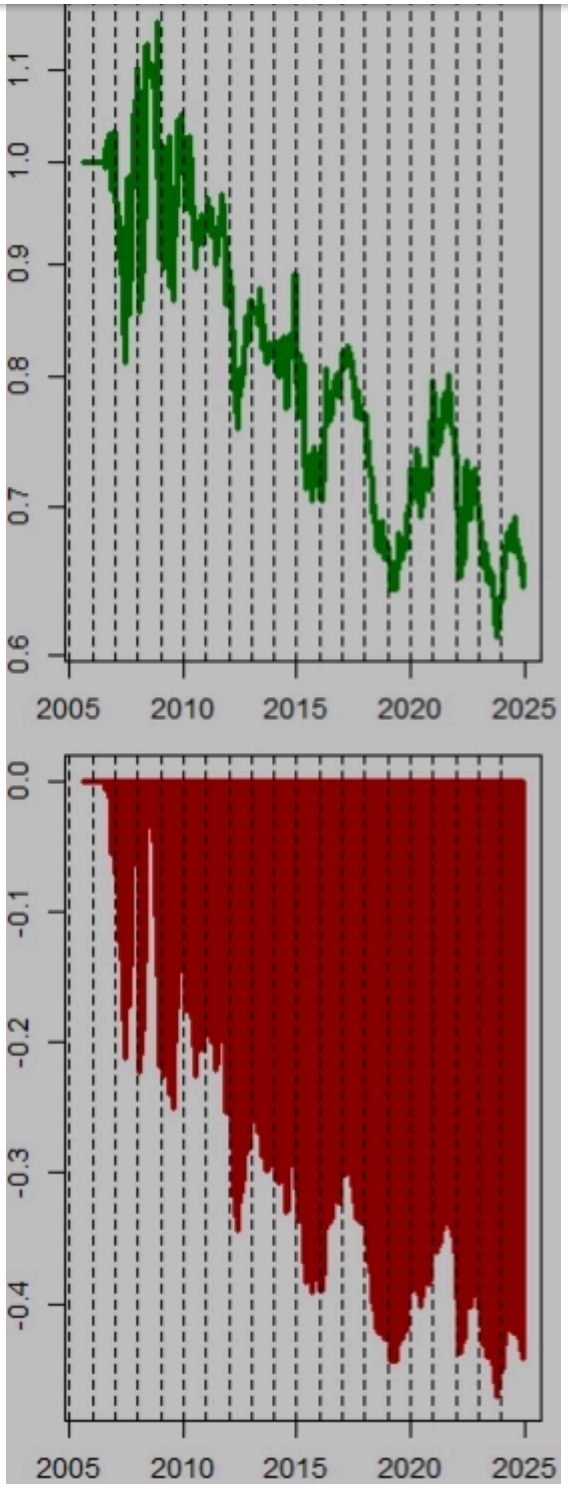
Автор: Михаил Шардин
Моя онлайн-визитка
Telegram «Умный Дом Инвестора»

19 марта 2025 г.

Александр Резвяков | торговые системы | торговые роботы | трейдинг

69 КОММЕНТАРИЕВ Сначала старые

Sergey Pavlov
19 марта 2025, 05:48

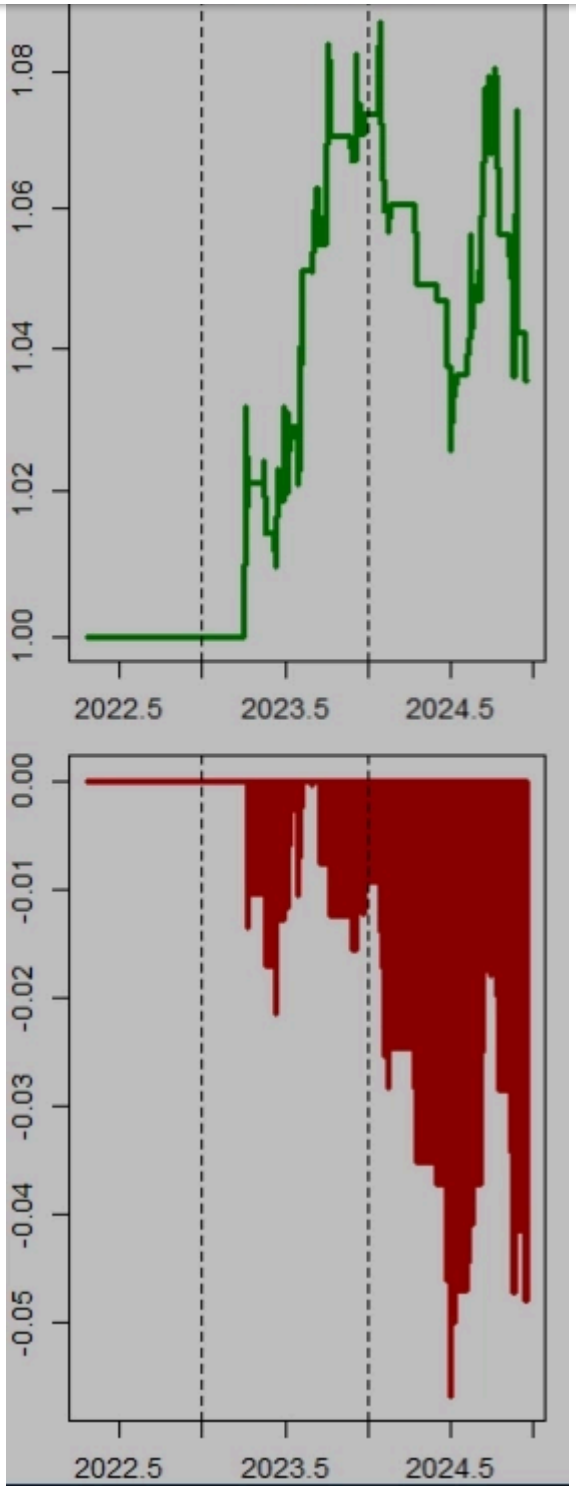


Показать 2 ответа

+7

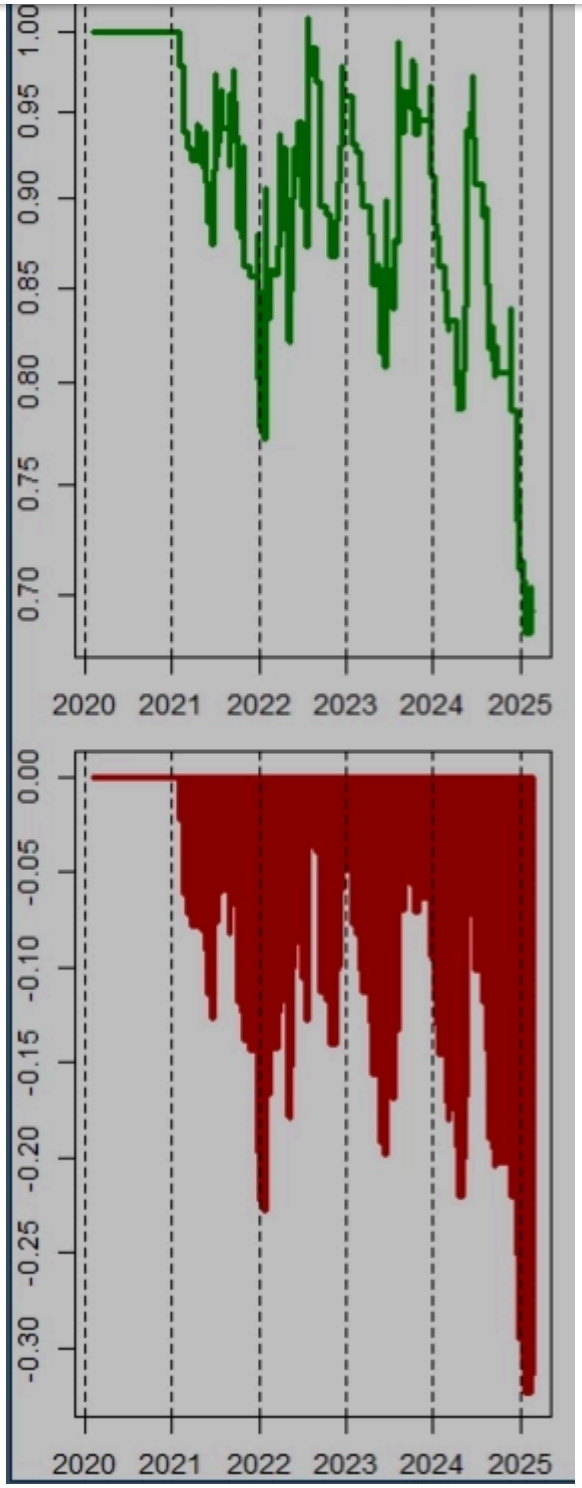
Sergey Pavlov

19 марта 2025, 05:49



+5

Sergey Pavlov
19 марта 2025, 05:50



Показать 9 ответов

+5

Sergey Pavlov

19 марта 2025, 05:51

откуда вот это всё взялось?

— Показать 1 ответ

0

Ещё 13 комментариев

Напишите комментарий...

ОТПРАВИТЬ