Санкт-Петербургский государственный университет

Кафедра Управления Медико-Биологическими Системами

Жданов Карим Искандерович

Выпускная квалификационная работа бакалавра

Исследование бутстрап-методов и их применение в статистике

Направление 01.03.02

Прикладная математика и информатика

Научный руководитель,

доктор технических наук,

профессор

Буре В.М.

Санкт-Петербург

2018

Оглавление

Введение 3
Постановка задачи 4
Обзор литературы 5
Обзор по методам бутстрапа 6
Метод простой перестановки (resampling) 6
Блочный бутстрап (Block bootstrap) 6
Глава 1. Применение бутстрап-метода для временных рядов 8
1.1. Описание метода 8
1.2. Моделирование исследования 10
1.2.1. MFB для независимой и одинаково распределенной выборки 10
1.2.2. MFB для временного ряда 12
1.3. Применение метода к реальным данным 15
1.4. Подведение итогов 18
Глава 2. Использование бутстрап-метода для построения доверительных интервалов линейной регрессии 20
2.1. Введение 20
2.2. Оценка бутстрапа 21
2.2.1 Стандартный бутстраповский доверительный интервал (SB) 21
2.2.2 Процентный бутстраповский доверительный интервал (PB) 22
2.2.3 Бутстраповский доверительный интервал с коррекцией смещения и ускорением (BCa) 22
2.3. Моделирование исследования 23
2.4. Применение методов к реальным данным 24
2.5. Подведение итогов 26
Заключение 27
Список литературы 28
Приложение 30

Введение

До компьютерной эпохи существовала малая часть методов для работы со статистическими данными. Большинство из них были или чрезмерно сложными, или требовали строгого выполнения некоторых условий (например, что распределение является нормальным). Это существенно сужало количество данных, которые можно было изучать, так как при невыполнении этих условий значительно понижалась надежность исследования.

В настоящее время появились эмпирические методы, основанные на интенсивном использовании компьютера, которые позволяют обходить это ограничение ранних методов.

Одним из них является бутстрап-метод, представленный Эфроном в 1976 году [1]. Он позволяет оценивать основные статистические характеристики, такие как среднее, медиана, стандартное отклонение и другие, а также построение доверительных интервалов, без предположений о распределении для малой выборки. Основная идея этого метода состоит в многократном извлечении выборки того же размера, что и исходная путем «вытягивания» случайного элемента с возвращением [2]. Таким образом, можно сгенерировать большое множество выборок, для каждой из которых после рассчитываем значение необходимой характеристики. На основе полученного множества можно построить гистограмму значений тестируемого показателя, отражающую закономерности его вариации, что дает возможность оценить доверительные интервалы и другие выборочные характеристики анализируемой величины.

Постановка задачи

Целью данной работы является изучение эмпирического метода бутстрап, его применение в анализе временных рядов и в задачах линейной регрессии.

Для достижения этой цели требуется выполнить следующие задачи:

1. Изучение методик и алгоритмов.
2. Симуляция метода на искусственных данных.
3. Применение методов на реальных данных и оценка их эффективности.

Обзор литературы

Для изучения теоретической основы методики бутстрапа были использованы книги Эфрона Б. [1] и Шитикова В.К., Розенберга Г.С. [2]. В них даны определение бутстрапа и общий принцип его работы.

У Бюльмана [3] и Анатольева С.А. [4] описаны основные типы бутстрапа, их методика, различия и сильные стороны и сферы их применения.

На основе статьи "A Simple Bootstrap Method for Time Series" [5] было подробно исследовано применение бутстрапа во временных рядах. С помощью статьи А.А. Молчанова [6] была изучена модель GARCH, рассматриваемая в первой главе. Для подтверждения эффективности изученных методик использовались данные индекса DJI с сайта http://investfunds.kz [8].

Далее было рассмотрено применение бутстрапа в задачах линейной регрессии. Для этого была использована статья К. Самарта, Н. Джансакула и М. Чонгваучхамнана [9], а также учебник Дж. Фокса [10]. Для оценки изученных методов были использованы данные с сайта http://vincentarelbundock.github.io/Rdatasets [11].

Для реализации рассмотренных выше алгоритмов на языке python, была изучена документация библиотек arch, numpy, pandas и matplotlib на сайтах https://github.com/bashtage/arch [7] и https://www.python.org/doc [12].

Обзор по методам бутстрапа

Метод простой перестановки (resampling)

Применяется для оценки основных статистик распределения (к примеру, среднего, медианы, дисперсии) без использования параметрических допущений [4]. Можно выделить два способа выполнения алгоритма:

Используется стандартный алгоритм Монте-Карло для передискретизации определенное количество раз, пока бутстраповская оценка распределения статистики не будет достаточно точной.

Второй же способ отличается от первого лишь тем, что проводится полный “перебор” всех возможных вариантов сочетания данных с возвращением.

Блочный бутстрап (Block bootstrap)

Рассматривается, когда данные скоррелированы (например, временные ряды). В этом случае метод простой перестановки не сработает, так как он не рассчитан на сохранение корреляции в данных [3]. Блочный бутстрап же использует для этого перестановку отдельных блоков данных. Также этот метод делится на несколько по способу составления блоков:

Простой блочный бутстрап – данные разбиваются на неперекрывающиеся блоки.

Стационарный бутстрап – задается вероятность окончания блока p. Первый элемент бутстраповской выборки выбирается произвольным образом. Далее, с вероятностью 1-p в данный блок включается следующий элемент исходной выборки, а с вероятностью p идет переход к новому блоку, первый элемент которого снова выбирается случайно из исходной выборки. Данный алгоритм продолжается до тех пор, пока в бутстраповскую выборку не наберется необходимое количество элементов (равной исходной выборке).

Выбор оптимальной длины блока является одним из основных вопросов. Чаще всего для ее определения используется графический способ.

Глава 1. Применение бутстрап-метода для временных рядов

В этой главе будет рассмотрен простой бутстрап-метод для временных рядов (далее MFB метод). Предлагаемый метод не зависит от модели и, следовательно, позволяет избежать определенных ситуаций, когда образцы бутстрапа могут содержать невозможные значения из-за повторной дискретизации из остатков. Метод прост в применении и может применяться к стационарным и нестационарным временным рядам (подробное описание метода рассматривается в [5]).

1.1. Описание метода

Для начала определяется количество порогов, которые будут определять группки наблюдений разной длины. (m-пороги).

Пусть имеется временной ряд длины n: $x_1, x_2, x_3, \ldots, x_n$.

Определим T_i $(i = 1, \ldots, m)$ как пороги, такие что:

$-\infty < T_1 < T_2 < \ldots < T_{m-1} < T_m = +\infty$.

Значения порогов определим следующим образом. Пусть $h = \frac{1}{m}$, тогда T_i равен (ih)-му квантилю ряда $x_1, x_2, x_3, \ldots, x_n$ при $i = 1, \ldots, m-1$ и $T_m = +\infty$.

Далее, каждому элемент x_t сопоставим элемент $y_t = i$, если i – это наименьшее значение, такое что $x_t < T_i$. Тогда наш MFB метод будет состоять из следующих шагов:

Шаг 1. Сопоставим каждому элементу x_t элемент y_t, получив ряд

$y_i, i = 1, \ldots, n$.

Для примера, рассмотрим ряд:

$$x_t: \ 7, 7, 3, 8, 7, 8, 8, 9, 7, 5, 4, 4, 8, 1, 2, 8, 9, 4, 5, 3.$$

Пусть $m = 3$, тогда $T_1 = 1,\ T_2 = 5, T_3 = \infty$. Следовательно, ряд y_i пусть выглядеть:

$$y_t:\ 3, 3, 2, 3, 3, 3, 3, 3, 3, 2, 2, 2, 3, 1, 2, 3, 3, 2, 2, 2.$$

Шаг 2. Разобьем ряды x_t и y_i на последовательности 1-порогов,..., m-порогов.

$$x_t:\ (7, 7)(3)(8, 7, 8, 8, 9, 7)(5, 4, 4)(8)(1)(2)(8, 9)(4, 5, 3)$$

$$y_t:\ (3, 3)(2)(3, 3, 3, 3, 3, 3)(2, 2, 2)(3)(1)(2)(3, 3)(2, 2, 2).$$

Шаг 3. Создаем бутстрап-выборку из y_t путем "вытягивания" с возвращением групп 1-порогов,..., m-порогов. Продолжать этот процесс до тех пор, пока размер бутстрап-выборки будет не меньше размера исходной выборки.

В нашем примере образовалось всего три таких группы:

1-порог: $(1)_1$

2-порог: $(2)_1\ (2, 2, 2)_2\ (2)_3\ (2, 2, 2)_4$

3-порог: $(3, 3)_1\ (3, 3, 3, 3, 3, 3)_2\ (3)_3\ (3, 3)_4$.

И в соответствии с ними собирается новая бутстрап-выборка, в итоге чего получается

$$y_t^*: (3)_3(2)_3(3, 3)_1(2, 2, 2)_4(3, 3)_4(1)_1(2, 2, 2)_4(3, 3, 3, 3, 3, 3)_2(2, 2, 2)_2.$$

Можно обратить внимание на длину получившейся выборки, пришлось добавить последнюю группу, так как без нее не хватало одного элемента до размера исходной выборки.

Шаг 4. Воссоздаем новую бутстрап-выборку x_t^* по y_t^*, сопоставляя каждой группе соответствующую ей из исходной выборки, после чего, если размер новой бутстрап-выборки получился больше исходной, отбрасываем лишние элементы.

Возвращаясь к нашему примеру:

1-порог: $(1)_1$

2-порог: $(3)_1\ (5,4,4)_2\ (2)_3\ (4,5,3)_4$

3-порог: $(7,7)_1\ (8,7,8,8,9,7)_2\ (8)_3\ (8,9)_4$.

Поэтому новая бутстрап-выборка будет выглядеть так:

$$y_t^*: (8)_3(2)_3(7,7)_1(4,5,3)_4(8,9)_4(1)_1(4,5,3)_4(8,7,8,8,9,7)_2(5,4,4)_2.$$

И, отбросив лишние элементы, получаем итоговую бутстрап-выборку

$$x_t: 8,2,7,7,4,5,3,8,9,1,4,5,3,8,7,8,8,9,7,5.$$

Шаг 5. Повторять действия Шага 3, пока не наберем M бутстрап-выборок.

Следует обратить внимание на то, что MFB метод гарантирует сохранение локальной автокорреляционной структуры бутстрап-выборок подобно структуре исходной выборки, потому что работа с порогами сохранит их порядок таким же, как и в исходной выборке. Также, благодаря тому, что группы порогов заменяются на группы из того же диапазона квантиля, бутстрап-выборки будут иметь количество элементов в каждом квантиле, пропорционально исходной выборке.

По этой причине стоить ожидать от MFB метода надежное средство для оценки распределения интересующих сложных статистик. Далее же проведем различные симуляционные исследования с целью определения эффективности MFB метода, а также проведем сравнение с методом двигающегося блока.

1.2. Моделирование исследования

1.2.1. MFB для независимой и одинаково распределенной выборки

По формулам, $X = \frac{1}{n}\sum_{i=1}^{n} X_i$, $\sigma_x^2 = \frac{\sigma^2}{n}$, где $X_1, \ldots, X_n$ – это нормальная и одинаково распределенная (дальше Н.О.Р.), со случайным математическим ожиданием и дисперсией σ^2. Будем рассматривать Н.О.Р., а именно,

сгенерируем 50 независимых случайных выборок $x_1^{(j)},\ldots,x_{200}^{(j)}$ $(j=1,\ldots,50)$ каждая из которых принадлежит нормальному распределению $N(0,16)$. Тогда при $n=200$ и $\sigma_x^2=\frac{16}{200}=0.08$.

Для каждой выборки j рассматриваем последовательность числа порогов m, $m=m_l$ $(l=1,2,\ldots,L)$, где в нашем случае m будет пробегать значения {2,4,6,8,10,12,14,16,18,20,30,40,50,60,70,80,90},

$(l=1,2,\ldots,17)$.

И для каждого j и l, соберем по 500 бутстрап-выборок размера 200, после чего оценим σ_x^2 по формуле $\hat{\sigma}^2_{x_{(jlk)}}=\frac{\hat{\sigma}^2_{x_{(jlk)}}}{n}$, где $\hat{\sigma}^2_{(jlk)}$ есть выборочная дисперсия k-й бутстрап-выборки, $k=1,\ldots,500$. Тогда $\hat{\sigma}^2_{x_{(jlk)}}(k=1,\ldots,500)$ будет представлять собой эмпирическое бутстрап-распределение σ_x^2.

Для каждого j и l:

$$\hat{\sigma}^2_{x_{(jl\cdot)}}=\frac{1}{500}\sum_{k=1}^{500}\frac{\hat{\sigma}^2_{(jlk)}}{n}, j=1,\ldots,50, l=1,\ldots,15.$$

И пусть $q_{jl}^{0.025}$ and $q_{jl}^{0.975}$ – это 2.5% и 97.5% квантили $\hat{\sigma}^2_{x_{(jlk)}}$ $(k=1,..,500)$, соответственно. Тогда $\hat{\sigma}^2_{x_{(jlk)}}$ даст среднее значение оценки бутстрапа величины σ_x^2 для j-й выборки с m_l числом порогов, и $(q_{jl}^{0.025},q_{jl}^{0.975})$ соответствует 95%-му бутстрап-квантилю.

Далее

$$\hat{\sigma}^2_{x_{(\cdot l\cdot)}}=\frac{1}{50}\sum_{j=1}^{50}\hat{\sigma}^2_{x_{(jl\cdot)}},\qquad \hat{q}_l^{0.025}=\frac{1}{50}\sum_{j=1}^{50}q_{jl}^{0.025},\hat{q}_l^{0.975}=\frac{1}{50}\sum_{j=1}^{50}q_{jl}^{0.975},$$

можно рассмотреть средний показатель оценки бутстрапа величины $\hat{\sigma}_x^2$ как функцию, зависящую от числа порогов, учитывающихся в MFB методе. На графике Рис. 1 можно увидеть $\hat{\sigma}^2_{x_{(\cdot l\cdot)}}$, зависящую от числа порогов, вместе с соответствующим 95%-м бутстрап-квантилем. Можно заметить, что ширина

95%-го бутстрап-квантиля стабилизируется при увеличении числа l. Но средняя оценка бутстрапа σ_x^2 не меняется в зависимости от изменения числа порогов, что легко объясняется использованием независимых выборок в симуляции. Средняя квадратичная ошибка между $\hat{\sigma}_{\cdot l}^2$ и значением величины $\sigma_x^2 = 0.08$ на всем диапазоне числа порогов не более 7×10^{-6}, что очень мало. Рис. 1 также показывает, что в среднем 40 порогов хватает для стабильного 95%-го бутстрап-квантиля.

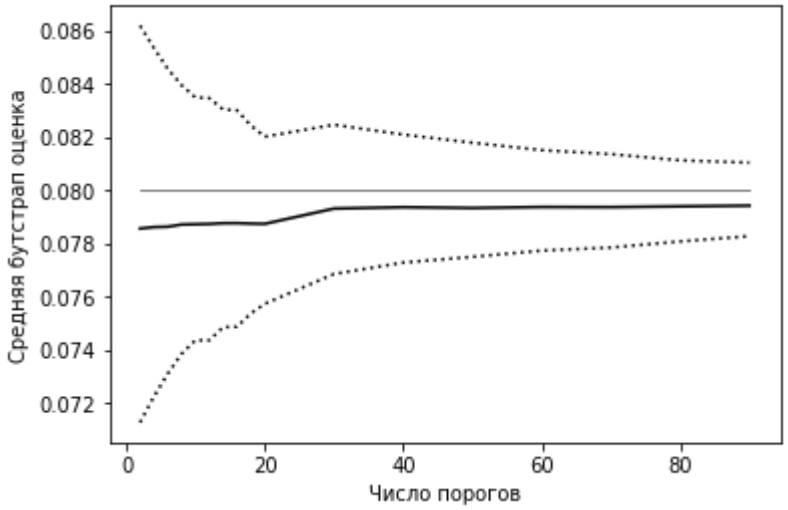

Рис. 1 Тонкая серая линия - σ_x^2 в зависимости от числа порогов, черная кривая линия - ее оценка, а две пунктирные кривые - границы 95%-го доверительного интервала.

1.2.2. MFB для временного ряда

Рассмотрим модель временного ряда ARMA(2,1)

$$x_t = 0.8x_{t-1} - 0.6x_{t-2} + \varepsilon_t + 0.3\varepsilon_{t-1}, \quad (1)$$

где ε_t это Н.О.Р., случайная величина со средним значением 0 и дисперсией 2. Тогда имеется $\varphi_1 = 0.8, \varphi_2 = -0.6, \varphi_3 = 0.3$. В этой симуляции исследуется средний показатель оценки бутстрапа и соответствующий ему квантиль бутстрап для модели с параметрами.

Как в прошлый раз, сгенерируем 50 независимых временных рядов $x_1^{(j)}, \ldots, x_{200}^{(j)}$ $(j = 1, \ldots, 50)$ по формуле (1).

Для каждого j рассматриваем последовательность чисел порогов. Также пусть m_l состоит из чисел

$\{2,4,6,8,10,12,14,16,18,20,30,40,50,60,70,80,90\}$.

Для каждого j и l соберем 500 бутстрап-выборок размерности 200, используя MFB метод, а затем построим модель ARMA(2,1) для каждой бутстрап-выборки, применив метод максимального правдоподобия (дальше ММП). Пусть $\varphi_{jlk}^{(i)}$ есть ММП оценка величины φ_i, где $i = 1,2,3,\ j = 1,\dots,5, l = 1,\dots,17$ и $k = 1,\dots,500$. Тогда средняя оценка бутстрапа величины φ_i будет рассчитываться по формуле

$$\varphi_{jl\cdot}^{(i)} = \frac{1}{500}\sum_{k=1}^{500}\varphi_{jlk}^{(i)}, i = 1,2,3, j = 1,\dots,50, l = 1,\dots,17.$$

Пусть $q_{jl}^{(i)0.025}$ и $q_{jl}^{(i)0.975}$ – это 2.5% и 97.5% квантили $\varphi_{jlk}^{(i)}$ $(k = 1,..,500)$, соответственно. Тогда $(q_{jl}^{(i)0.025}, q_{jl}^{(i)0.975})$ соответствует 95%-му бутстрап-квантилю величины φ_i для каждого j и l.

Кроме того, пусть

$$\varphi_{\cdot l\cdot}^{(i)} = \frac{1}{50}\sum_{j=1}^{50}\varphi_{jl\cdot}^{(i)},\ q_l^{(i)0.025} = \frac{1}{50}\sum_{j=1}^{50} q_{jl}^{(i)0.025}, q_l^{(i)0.975} = \frac{1}{50}\sum_{j=1}^{500} q_{jl}^{(i)0.975}.$$

Тогда $\varphi_{\cdot l\cdot}^{(i)}$ и $(q_{jl}^{(i)0.025}, q_{jl}^{(i)0.975})$ обеспечивает среднюю оценку бутстрапа величины φ_i и средний 95%-й бутстрап-квантиль для 50 независимых временных рядов модели (1).

На Рис. 2 показаны $\varphi_{\cdot l\cdot}^{(i)}$, зависящие от числа порогов, вместе с соответствующим 95%-м бутстрап-квантилем.

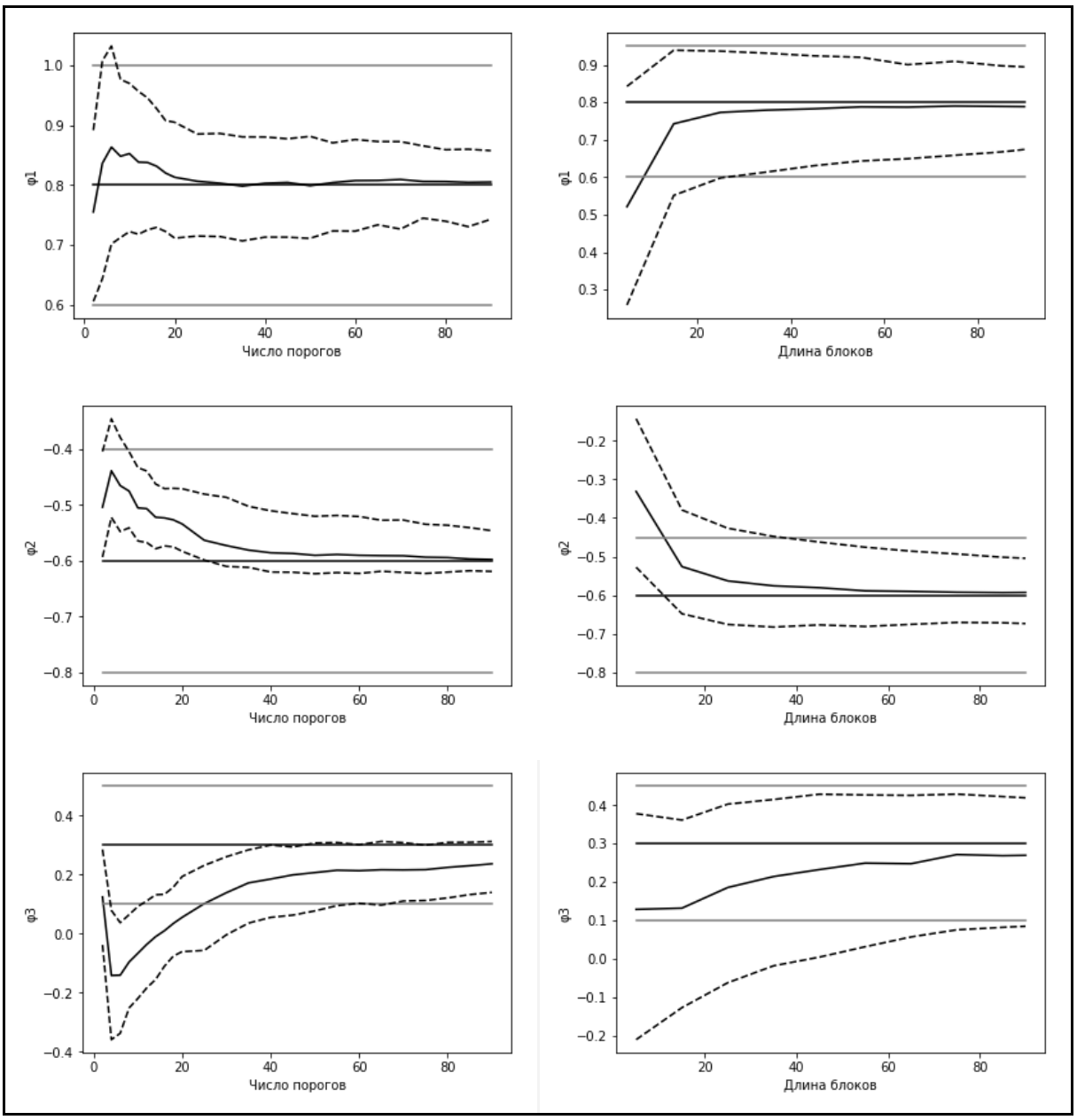

Рис. 2. На графиках изображены истинные значения параметров (черные горизонтальные прямые), средние оценки бутстрап (черные кривые), 95%-е диапазоны бутстрап-квантилей (пунктирные кривые) и 95%-е доверительные интервалы (серые прямые), полученные ММП методом для 50 независимо сгенерированных временных рядов модели (2).

Также был использован метод бутстрап двигающегося блока для создания 500 бутстрап-выборок для каждого j и J, где J это длина блоков (описание и алгоритм этого метода были взяты из статьи [3] и библиотеки python arch из источника [10]), $J \in \{5,15,25,35,45,55,65,75,85,90\}$.

Результаты этого отображены в правой колонке Рис.2

По левой колонке Рис. 2 видно, что $\varphi_{.l.}^{(i)}$ сходится к истинному значению φ_i. Диапазон бутстрап-квантилей для $i = 1,2,3$ также содержит φ_i и при увеличении числа порогов приближается к нему верхней и нижней границами. Стоит также отметить, что по мере роста числа порогов, диапазон бутстрап-квантилей начинает лучше описывать истинное значение φ_i, нежели 95%-й доверительный интервал, полученный ММП методом.

Обратив внимание на правую колонку Рис. 2, можно заметить, что средняя эффективность метода бутстрап двигающегося блока в целом схожа с MFB методом, но границы 95%-го бутстрап-квантиля несколько шире, по сравнению с MFB методом.

1.3. Применение метода к реальным данным

В этой части применяется исследуемая методология к Индексу Доу Джонса (DJI) на отрезке от 03.01.2011 до 18.05.2018 (данные были взяты с информационного портала [8]). Длина ряда 1864. График временного ряда наблюдаемого индекса и функция “log returns” (в процентах), обозначенная y_t $(t = 1, \dots, n = 1864)$, которую можно найти по формуле

$$R = ln(\frac{V_f}{V_t}), V_f \ - \text{конечное значение}, V_t \ - \ \text{начальное значение},$$

показаны на Рис. 3(а) – (б). Как можно было ожидать, наблюдаемый индекс показывает наличие экстремумов и волатильность кластеров, которые можно подробнее рассмотреть на графике автокорреляционной функции от y_t(дальше АКФ) и АКФ от y_t^2 на Рис. 3(в) – (г). Для их изучения можно использовать GARCH(1,1) модель к y_t при использовании МНК метода [6].

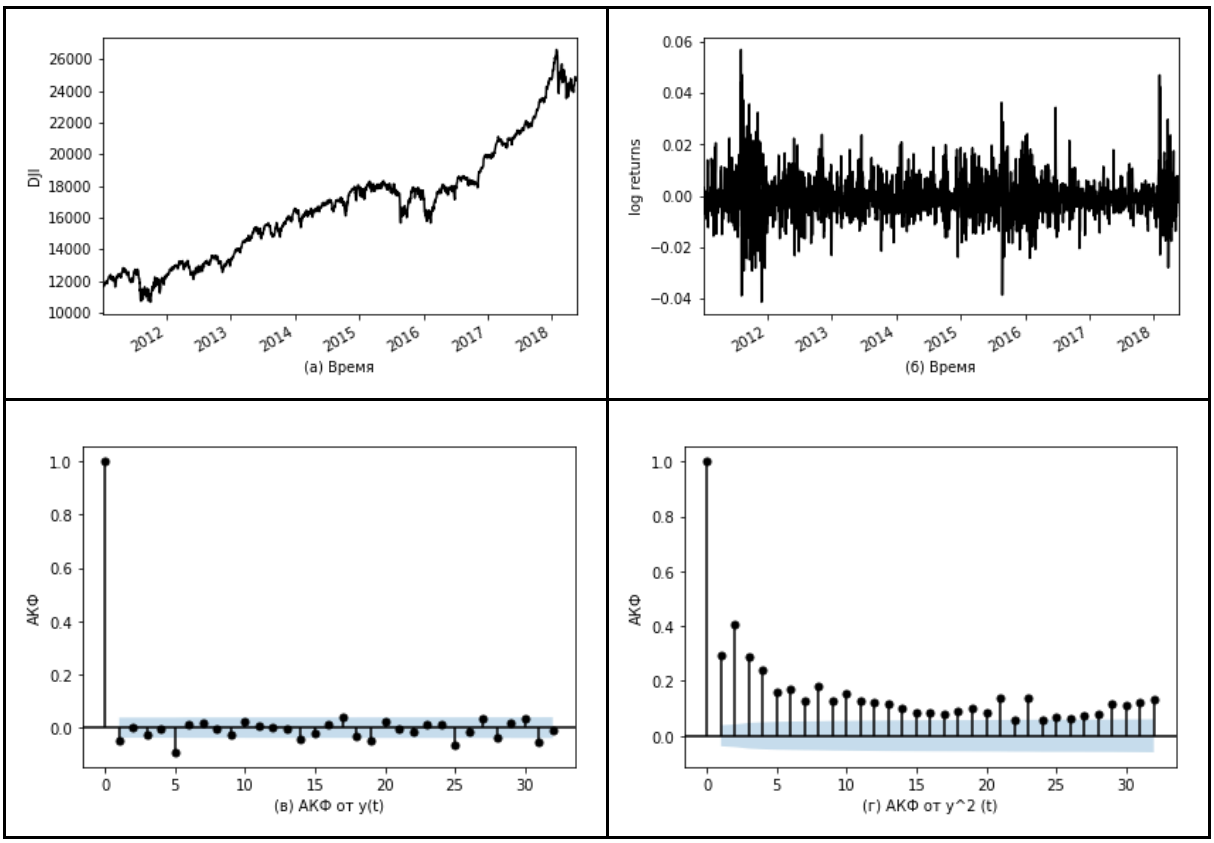

Рис. 3. Графики оценок бутстрапа и границы бутстрап-квантиля вместе с доверительным интервалом, полученным ММП методом для временного ряда индекса DJI.

Так как цель нашего исследования состоит не в поиске наилучшей модели временного ряда, будем использовать модель GARCH(1,1) для изучения эффективности MFB метода. Модель GARCH(1,1) выражается

$$y_t = \sigma_t \varepsilon_t, \sigma_t^2 = \alpha_0 + \alpha_1 y_{t-1}^2 + \alpha_2 \sigma_{t-1}^2,$$

где ε_t- НОР $N(0,1)$, оценка параметров и их стандартная ошибка представлены в Таблице 1.

Вновь будем рассматривать последовательность чисел порогов $m_l = 2,4,6,8,10,12,14,16,18,20,30,40,50,60,70,80,90,100,150$, то есть $l = 1,\dots,19$. Для каждого l строится 500 бутстрап-выборок по рассмотренному выше ряду y_t. Значения оцениваемых параметров, основывающихся на бутстрапе выборок, будем обозначать как $\alpha_{ilk} (i = 0,1,2, l = 1,\dots,19$ и $k = 1,\dots,500)$. Пусть $\alpha_{il.} = \frac{1}{500}\sum_{k=1}^{500} \alpha_{ilk}$, и $(q_{jl}^{0.025}, q_{jl}^{0.975})$ – средняя оценка бутстрапа 95%-го бутстрап-квантиля для α_i, соответственно, где $q_{jl}^{0.025}$ и $q_{jl}^{0.975}$ – это 0.25%-й и

97.5%-й квантиль величины α_{ilk} $(k = 1, \ldots, 500)$, соответственно (вычислительная работа была проведена на языке python при использовании библиотеки [7]). Реализация алгоритма представление в Приложении 1.

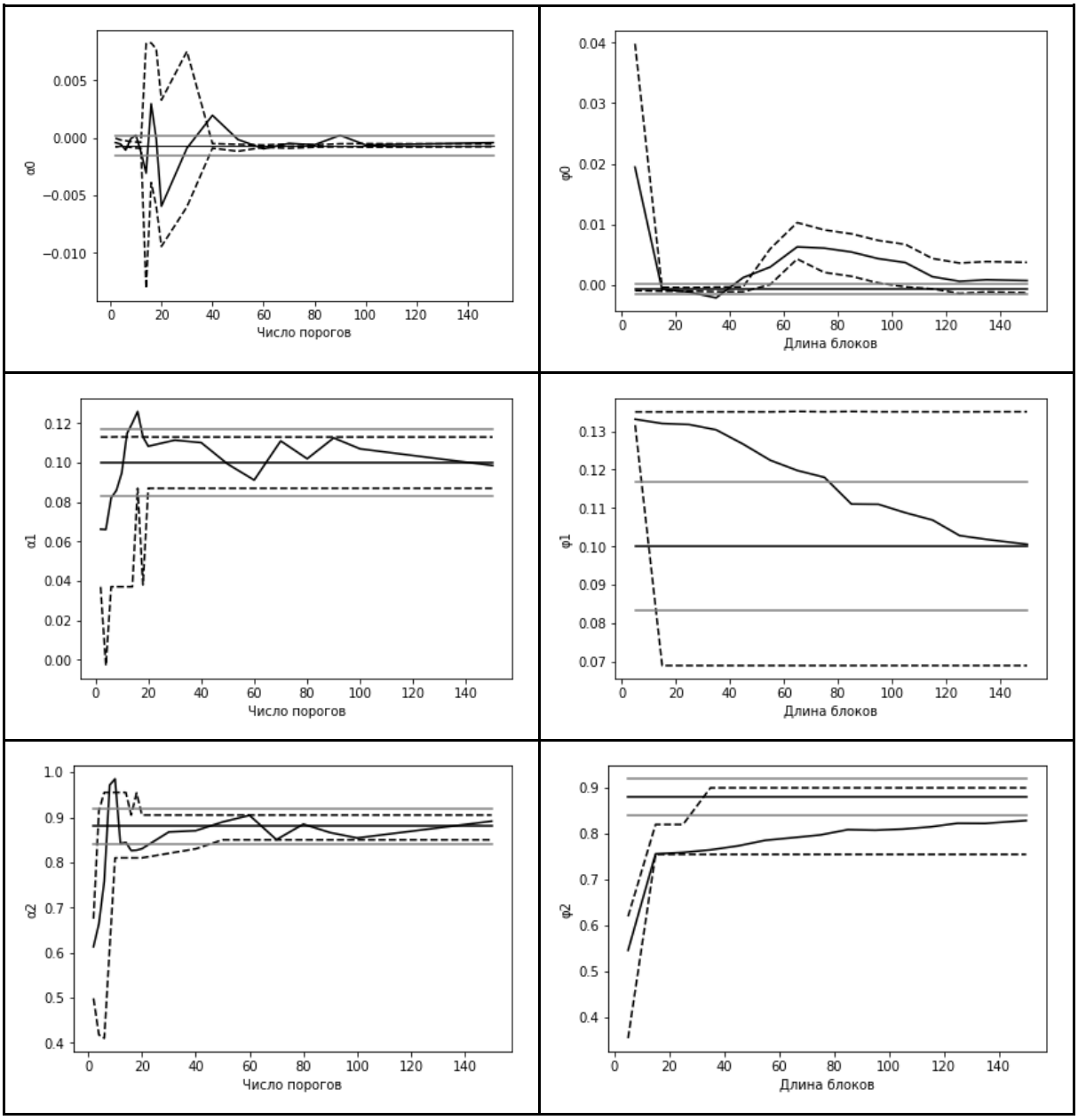

Рис. 4. Графики оценки бутстрапа и квантиль бутстрапа вместе с параметрами модели ММП и доверительными интервалами временного ряда индекса DJI

Левая колонка Рис. 4 показывает графики оценки параметров бутстрапа (черная прямая) и соответствующие 95%-ые бутстрап-квантили (пунктирные кривые) для α_i $(i = 0,1,2)$, полученные с помощью MFB метода. Также на

графиках отображены параметры ММП (темные горизонтальные линии) и соответствующие 95%-ые доверительные интервалы (светлые горизонтальные линии). Как можно увидеть, в первой половине значений числа порогов оценка бутстрапа нестабильна, но при его увеличении оценка бутстрапа и её бутстрап-квантиль начинают сходиться в пределах доверительного интервала ММП метода. Также стоит отметить, что после значения $m_0 = 80$ оценки бутстрапа почти совпадают со своим 95%-м бутстрап-квантилем. Поэтому эти оценки можно применять в статистическом анализе.

В правой колонке Рис. 4 отображены подобные результаты, полученные уже методом бутстрапа двигающегося блока. И вновь, общая эффективность последнего метода схожа с MFB методом. То есть, оценка бутстрапа двигающегося блока с увеличением длины блока сходится, но соответствующий ей 95%-й квантиль более широкий, нежели у MFB метода.

1.4. Подведение итогов

Был рассмотрен MFB метод, его алгоритм, а также изучили его эффективность на большом количестве моделирований исследований. Результаты показывают, что при увеличении числа порогов оценки параметров бутстрапа начинают сходиться к истинным значениям. Кроме того, 95%-й квантиль бутстрапа параметров модели в среднем уже, чем 95%-е доверительные интервалы, полученные при помощи ММП метода. Исследование временного ряда по индексу DJI показывает, что метод хорош и на практике.

Также становится понятно, что эффективность MFB метода схожа эффективности метода бутстрапа двигающегося блока. А для наиболее точного результата выбор наилучшего числа порогов m_0 можно воспользоваться графическим методом, как было показано на Рис. 2 и 4.

После чего продолжить статистический анализ, основываясь на оценке бутстрапа и оценки квантиль бутстрапа для данного m_0.

Кроме того, для улучшения метода можно увеличить число рассматриваемых различных бутстрап-выборок, но по мере их роста будет повышаться и время работы. Данные о времени работы представлены в Таблице 2. По ней можно судить, что время выполнения данных методов для подобной точности в пределах нормы.

Глава 2. Использование бутстрап-метода для построения доверительных интервалов линейной регрессии

В этой главе основное внимание уделяется использованию точного бутстрапа для построения доверительных интервалов для параметров регрессии в небольших образцах (подробное описание методологии рассматривается в [9]). Сравнение точного метода бутстрапа с простым бутстрап-методом будет проведено на искусственной взятой из базы данных выборах.

2.1. Введение

Регрессионный анализ используется повсеместно для изучения взаимосвязей между переменными, прогнозирования и предсказания значений независимых переменных. Также необходимо упомянуть о четырех важных предположениях: линейность, независимость, гомоскедастичность и нормальность, выполнение которых обусловливает достоверность результатов. Для точного прогнозирования необходимо большое количество наблюдений, получение которых в некоторых случаях не представляется возможным. По этой причине размер интересующей выборки может быть в пределах десяти значений. Это может вызвать нарушение одного из четырех предположений, а следовательно, дальнейшие результаты могут оказаться ненадежными.

Для обхода этой проблемы разработан точный бутстрап-метод. В данной главе будет проведено сравнение его с простым бутстрап-методом.

2.2. Оценка бутстрапа

Бутстрап – статистический подход, основанный на случайном построении выборок, созданных из оригинальной выборки с сохранением распределения. Пусть $X = (X_1, X_2, \ldots, X_n)$ – случайная выборка неизвестного распределения F, где $\hat{\theta}$ - оценка его параметра θ. Тогда бутстрап-выборка будет выглядеть как $X^* = X_1^*, X_2^*, \ldots, X_n^*$, а оценка этой бутстрап-выборки будет обозначаться $\widehat{\theta^*}$.

Аппроксимация выборочного распределения с параметром $\hat{\theta}$ проводится при использовании бутстрап-оценки $\widehat{\theta^*}$. Обычно, для построения распределения $\widehat{\theta^*}$ использовался бутстрап-метод на основе метода приближения Монте Карло и количество созданных бутстрап-выборок ограничивалось некоторым числом (500, 1000), но с улучшением компьютерных технологий этот метод не является единственным, если размер выборке не слишком велик.

Пусть исходная выборка состоит из n элементов, тогда могут быть созданы всевозможные образцы общим количеством n^n, то есть все реализации могут быть вычислены. Этот процесс называется точным методом начальной загрузки. Его невозможно применять в больших образцах, так как количество всех возможных образцов будет экспоненциально быстро возрастать с размером выборки.

Также были разработаны три бутстраповских доверительных интервалов, описанные следующим образом.

2.2.1 Стандартный бутстраповский доверительный интервал (SB)

Поскольку большинство статистик асимптотически нормально распределены, можно получить SB доверительный интервал $100\,(1-\alpha)\%$ для θ на основе оценки $\hat{\theta}$ в больших выборках следующим образом:

$$\left(\hat{\theta} - z_{\alpha/2} SE^*(\hat{\theta}^*), \hat{\theta} + z_{\alpha/2} SE^*(\hat{\theta}^*)\right),$$

где $z_{\alpha/2}$ это стандартная нормальная величина с вероятностью $\alpha/2$ и

для простого бутстрапа: $SE^*(\hat{\theta}) = \sqrt{\frac{\sum_{b=1}^{B}(\hat{\theta}_b^* - \theta^*)^2}{B-1}}$

для точного бутстрапа: $SE^*(\hat{\theta}) = \sqrt{\frac{\sum_{b=1}^{B}(\hat{\theta}_b^* - \theta^*)^2}{B}}$

где $\theta^* \equiv \frac{\sum_{b=1}^{B}\hat{\theta}_b^*}{B}$.

2.2.2 Процентный бутстраповский доверительный интервал (PB)

Непараметрический доверительный интервал для величины θ может быть построен с помощью квантилей распределения бутстрап образца $\hat{\theta}^*$. Пусть $\hat{\theta}_b^*$ – упорядоченный массив оценок бутстрапа. PB доверительный интервал $100\ (1 - \alpha)\%$ для θ будет выглядеть как

$$\left(\hat{\theta}^*_{(lower)}, \hat{\theta}^*_{(upper)}\right),$$

где $lower = 0.025 \times B$ и $upper = 0.975 \times B$.

2.2.3 Бутстраповский доверительный интервал с коррекцией смещения и ускорением (BCa)

Бутстрап распределение величины $\hat{\theta}^*$ может быть смещено. Поэтому предложен другой подход, корректирующий потенциальное смещение и ускоряющий сходимость бутстрап распределения. Доверительный интервал $100\ (1 - \alpha)\%$ для θ, с двумя корректирующими факторами, Z и A, определяются следующим образом:

$$Z = \Phi^{-1}\left[\frac{\#_{b=1}^{B}(\hat{\theta}_b^* < \hat{\theta})}{B}\right],$$

где Φ^{-1} обозначает обратную функцию стандартного нормального распределения Φ и $\#_{b=1}^{B}(\hat{\theta}_b^* < \hat{\theta})$ представляет собой долю бутстрап образцов ниже оценки $\hat{\theta}$. Пусть $\hat{\theta}_{(-i)}$ представляет значение $\hat{\theta}$, образованное

путем исключения i-го наблюдения из оригинальной выборки, θ есть среднее $\hat{\theta}_{(-i)}$. Тогда имеется:

$$A \equiv \frac{\sum_{i=1}^{n}(\hat{\theta}_{(-i)} - \theta)^3}{6\left[\sum_{i=1}^{n}(\hat{\theta}_{(-i)} - \theta)^2\right]^{3/2}},$$

и вычислим

$$A_1 \equiv \Phi\left[Z + \frac{Z - z_{\alpha/2}}{1 - A(Z - z_{\alpha/2})}\right],$$

$$A_2 \equiv \Phi\left[Z + \frac{Z + z_{\alpha/2}}{1 - A(Z + z_{\alpha/2})}\right].$$

Следовательно, BCa доверительный интервал $100\,(1 - \alpha)\%$ для θ будет получен из

$$\left(\hat{\theta}^*_{(lower^*)}, \hat{\theta}^*_{(upper^*)}\right),$$

где $lower^* = BA_1$ и $upper^* = BA_2$.

2.3. Моделирование исследования

Эта часть содержит результаты симуляции, которая иллюстрирует сравнение эффективности оценки доверительных интервалов для параметров регрессии (более подробно работа с параметрами линейной регрессии описана в [10]). Проведено симуляционное исследование для сравнения доверительных интервалов, построенных на основе точного бутстрап-метода с учетом простого метода бутстрапа. Оценка трех доверительных интервалов была основана на их точности охвата и средней длине.

Генерируем две случайные выборки X размера $n = 5$ и $n = 8$, где X принадлежит равномерному распределению [0,1], и соответствующие им значения Y, полученные по формуле:

$$Y_i = 2 + 4x_i + e_i,$$

где e_i – это набор независимых значений следующих распределений:

1. Стандартное нормальное распределение,

2. Экспоненциальное распределение со средним значением 1,

3. Распределение Лапласа со средним значением 0 и дисперсией 1

4. Распределение Пуассона со средним значением 3.

Для каждого заданного распределения и размера выборки будем оценивать коэффициенты линейной регрессии a и b. Найдем их по формулам

$$b = \frac{Cov_n(x,y)}{Var_n(x)},$$

$$a = y - bx.$$

Далее, построим бутстрап-выборки для простого ($B = 1000$) и точного ($B = n^n$) метода, по которым вычислим оценки коэффициентов $\widehat{a_i^*}$ и $\widehat{b_i^*}$, $i = 1, \ldots, B$, после чего можно будет построить для каждого случая доверительные интервалы, рассмотренные ранее.

Для оценивания доверительных интервалов рассмотрим такие величины как вероятность покрытия и средняя длина (вычислительная работа была проведена на языке python [12]). Полученные результаты представлены в приложении в Таблицах 3, 4.

Обратив внимание на Таблицу 3, можно заметить, что в целом вероятность покрытия точного метода более полная по сравнению с простым методом. Причем, значения для доверительных интервалов SB несколько выше значений PB и Bca. Из Таблицы 4 же становится видно, что средняя длина интервалов практически одинаковая для простого и точного методов.

2.4. Применение методов к реальным данным

В этой части применяются ранее изученные в этой главе методики для проведения регрессионного анализа на данных “Text Prices”. Данные были взяты с сайта [11]. Была построена линейная регрессия между двумя переменными “Pages” и “Price” (количество страниц и цена книг), оценки

коэффициентов и их доверительные интервалы. Реализация алгоритма представлена в Приложении 2. Результат показан в Таблице 5.

По таблице можно определить, что, несмотря на невысокую разницу между оценками коэффициентов точного и простого метода, стандартная ошибка бутстрапа будет меньше в пользу точного метода. Что касается доверительных интервалов, у SB интервала при использовании простого метода длина несколько больше, а у BCa интервала между длинами обоих методов практически нет никакой разницы.

Таблица 5. Оценка бутстрапа, стандартная ошибка и 95%-й доверительный интервал коэффициентов линейной регрессии для “Paperback Books”.

	Тип бутстрапа	Коэффициенты	
		a	b
Средняя оценка бутстрапа	Точный	0.110438	-1.83475
	Простой	0.110476	-1.85728
Стандартная ошибка бутстрапа	Точный	0.024	5.834289
	Простой	0.029	6.42246
SB интервал	Точный	(0.05089, 0.159326)	(-11.484, 8.75763)
	Простой	(0.05492, 0.155298)	(-16.538, 13.8113)
PB интервал	Точный	(0.07061, 0.14979)	(-11.87, 8.2261)
	Простой	(0.072847, 0.14979)	(-11.9709, 7.9764)
BCa интервал	Точный	(0.07402, 0.15796)	(-12.4064, 7.7691)
	Простой	(0.07488, 0.15509)	(-12.138, 7.71578)

2.5. Подведение итогов

Был рассмотрен точный бутстрап и были построены три доверительных интервала SB, PB и BCa для параметров регрессии в небольших образцах, после чего проверили их эффективность в моделировании и на реальных данных. Была проведена оценка трех доверительных интервалов на основе их вероятности покрытия и средней длине интервалов. Результаты исследования показали, что точный метод оказался более эффективный, так как его вероятность покрытия оказалась более высокой по сравнению с обычным методом, а средняя длина интервала – более короткой.

Кроме того, наблюдая за данными, представленными в Таблице 6, можно обратить внимания на разницу по времени, затраченному на выполнение обоих методов. Из чего следует, что на небольшом размере выборки ($n < 10$) лучше использовать точный метод бутстрапа, Но при большем размере рекомендуется прибегать к простому методу с ограничением числа бутстрап образцов ($B \approx 10000$).

Заключение

В результате работы были изучены методики статистического анализа: бутстрап-метод для временных рядов, точный бутстрап-метод для регрессионного анализа и способы построения доверительных интервалов для оценки его эффективности.

Также был проведен сравнительный анализ на искусственных данных и на данных, полученных из базы данных (индекс Доу Джонса в бутстрап-методе для временных рядов и данные параметров книг в точном бутстрап-методе для регрессионного анализа). По сравнительному анализу была доказана эффективность рассмотренных в работе методов.

Бутстрап-метод является хорошим средством работы со статистическими данными, характер распределения которых не может быть однозначно определен, а также выборками, содержащими небольшое количество данных, без снижения надежности полученных результатов.

Список литературы

[1] Эфрон Б. – Нетрадиционные методы многомерного статистического анализа. Год: 1988 264 с.

[2] Шитиков В.К., Розенберг Г.С. Рандомизация и бутстрап: статистический анализ в биологии и экологии с использованием R. – Тольятти: Кассандра, 2013. 314 с.

[3] Peter Bühlmann. Bootstraps for Time Series. Statistical Science.

2002, Vol. 17, No. 1, 52–72

[4] Анатольев С.А. Экономический ликбез: бутстрап. Основы бутстрапирования. Квантиль, №3, сентябрь 2007.

[5] Yuzhi Cai & Neville Davies. A Simple Bootstrap Method for Time Series. Communications in Statistics-Simulation and Computation®, 41: 621–631, 2012.

[6] А.А. Молчанов. Использование GARCH модели для исследования динамики курса валют. 2006.

[7] Авторегрессионная Условная Гетероскедастичность (ARCH) и другие инструменты финансовой эконометрики, написанные на Python. https://github.com/bashtage/arch

[8] Информационный портал о личных инвестициях и финансах http://investfunds.kz/world/indicators/indeks-dow-jones/

[9] K. Samart, N.Jansakul & M. Chongcheawchamnan.

Exact bootstrap confidence intervals for regression coefficients in small samples. Communications in Statistics-Simulation and Computation®, VOL.0, NO. 0, 1–7, 2017

[10] John Fox. Applied regression analysis and generalized linear models. 2016. 817 с.

[11] Rdasets. Архив наборов данных, распределенных с R. http://vincentarelbundock.github.io/Rdatasets/

[12] Документация библиотек языка python/

https://www.python.org/doc/

Приложение

Таблица 1. Оценки параметров и соответствующие им стандартные ошибки GARCH(1,1) модели для временного ряда индекса DJI.

	α_0	α_1	α_2
Оценка	-0.000692	0.1000	0.8800
Стандартная ошибка	0.844e-03	1.673373e-02	0.0395300
95%-й доверительный интервал	[-7.282e-02, -6.563e-02]	[6.730e-02, 0.133]	[0.853, 0.907]

Таблица 2. Время выполнения методов для вычисления оценок.

Бутстрап-метод	Время работы метода (сек.)
MFB для Н.О.Р. выборки	6830.5145
MFB для временного ряда	112443.42992
Бутстрап движущихся блоков для временного ряда	49221.9722
MFB для временного ряда индекса DJI	21023.5527
Бутстрап движущихся блоков для временного ряда индекса DJI	11334.5743

Таблица 3. Вероятность покрытия 95%-го доверительного интервала

			Доверительные интервалы бутстрапа					
			SB		PB		BCa	
Коэфф-т	Распределение	Тип бутстрапа	n=5	n=8	n=5	n=8	n=5	n=8
a	Нормальное	Точный	0.701	0.83	0.69	0.79	0.661	0.777
		Простой	0.695	0.812	0.678	0.781	0.654	0.767
	Экспоненциальное	Точный	0.491	0.487	0.441	0.436	0.462	0.382
		Простой	0.486	0.486	0.437	0.428	0.444	0.391
	Лапласа	Точный	0.70	0.825	0.671	0.767	0.665	0.785
		Простой	0.704	0.814	0.66	0.756	0.644	0.774
	Пуассона	Точный	0.52	0.572	0.56	0.511	0.51	0.523
		Простой	0.496	0.567	054	0.519	0.509	0.499
b	Нормальное	Точный	0.63	0.775	0.629	0.76	0.618	0.753
		Простой	0.632	0.781	0.621	0.768	0.620	0.748
	Экспоненциальное	Точный	0.704	0.765	0.641	0.747	0.655	0.742
		Простой	0.699	0.762	0.637	0.736	0.644	0.747
	Лапласа	Точный	0.648	0.775	0.641	0.759	0.604	0.753
		Простой	0.639	0.776	0.628	0.754	0.599	0.755
	Пуассона	Точный	0.69	0.777	0.651	0.771	0.648	0.766
		Простой	0.686	0.772	0.644	0.758	0.641	0,756

Таблица 4. Средняя длина 95%-го доверительного интервала

			Доверительные интервалы бутстрапа					
			SB		PB		BCa	
Коэфф-т	Распределение	Тип бутстрапа	n=5	n=8	n=5	n=8	n=5	n=8
a	Нормальное	Точный	2,5	2,2	2,5	2,2	2,3	2,1
		Простой	2,5	2,3	2,5	2,2	2,4	2,1
	Экспоненциальное	Точный	2,5	2,1	2,4	2,1	2,3	2,1
		Простой	2,5	2,1	2,4	2,1	2,3	2,1
	Лапласа	Точный	3,3	3,1	3,3	2,9	3,1	3,0
		Простой	3,3	3,1	3,3	2,9	3,1	3,1
	Пуассона	Точный	2,6	2,1	2,4	2,1	2,4	2,2
		Простой	2,6	2,1	2,4	2,1	2,4	2,2
b	Нормальное	Точный	4,6	3,7	4,4	3,8	4,0	3,9
		Простой	4,6	3,7	4,5	3,8	4,1	3,9
	Экспоненциальное	Точный	4,2	3,7	4,1	3,6	3,9	3,5
		Простой	4,2	3,7	4,1	3,6	3,9	3,5
	Лапласа	Точный	5,9	5,2	5,7	5,2	5,5	5,0
		Простой	5,9	5,2	5,7	5,2	5,5	5,0
	Пуассона	Точный	4,6	3,8	4,4	3,7	4,3	3,7
		Простой	4,6	3,8	4,4	3,7	4,3	3,7

Таблица 6. Время выполнения методов для вычисления оценок доверительных интервалов.

Тип доверительного интервала	Время работы метода (сек.)			
	Простой		Точный	
	n=5	n=8	n=5	n=8
SB	0.2445354	49.61516	9.144111	3890.288
PB	0.19812345	49.60917	9.037314	4131.755
BCa	0.25734109	49.66207	9.047794	3272.9112

Приложение 1. Программная реализация MFB метода на python

```
import random
import numpy
import statistics
import pandas as pd
import numpy as np
import random
import time
import pylab
from arch import arch_model

k1 = 500 #количество повторов метода бутстрапа для каждого значения
m1 = [2,4,6,8,10,12,14,16,18,20,30,40,50,60,70,80,90,100,150]

def generate():
    dji = pd.read_csv('dow_jones.csv',
                      sep=',', encoding='latin1',
                      parse_dates=['Data'], dayfirst=True,
                      index_col=['Data'])
    x = np.array([np.log(dji['Count'][i]) - np.log(dji['Count'][i-1])for i in
range(1, 1864)])
    return(x)
def n_0(x, m):
    """
    Высчитывает квантили T
    """
    T = [np.sort(x)[(len(x)//m)*(i)] for i in range(1, m)]
    T.append(1000)
    return T



def n_1(x, T):
    """
    Выводит y для определенного числа порогов m по заданному выше T
    """
    y = []
    for i in x:
        for e,t in enumerate(T):
            if i <= t:
                y.append(e+1)
                break
    return y
```

```
def n_2(x, y, m):
    """
    Группирует x и y в xx и yy по их принадлежности в определнные квантили
    Создает словарь, в котором для каждого квантиля (значения T) записываются
все различные
    группы
    b - сохраняет первоначальный вид x по этим квантилям
    """
    d = {i+1:[] for i in range(m)}
    z, q = y[0], x[0]
    b = [(y[0])]
    xx, yy = [], []
    z1, q1 = [y[0]], [x[0]]
    for i in range(len(y)-1):
        if y[i+1] == z:
            z1.append(y[i+1])
            q1.append(x[i+1])
        else:
            d[z].append([z1, q1])
            xx.append(z1)
            yy.append(q1)
            z = y[i+1]
            b.append(y[i+1])
            q = x[i+1]
            z1, q1 = [(y[i+1])], [(x[i+1])]
    d[z].append([z1, q1])
    xx.append(z1)
    yy.append(q1)
    return(xx, yy, d, b)


def n_3(d, b, lenght):
    """
    создается новый бутстрап y3 и по нему x3
    """
    i = 0
    x3 = []
    y3 = []
    while len(y3) <= lenght:
        k = random.randrange(0, len(d[b[i]]))
        y3.append((d[b[i]][k][0]))
        x3.append((d[b[i]][k][1]))
        i += 1
        i = i%len(b)
    return(x3, y3)

def n_4(x3, y3):
```

```
    """
    Приводят x3 и y3 к нормальному виду
    """
    xall=[]
    yall=[]
    for lst in x3:
        xall.extend(lst)
    for lst in y3:
        yall.extend(lst)
    return(xall[:len(x3)], yall[:len(x3)])

def simple_boot(x, m, k=k1):
    h1, h2, h3 = [], [], []
    T = n_0(x, m)
    y = n_1(x, T)
    xx, yy, d, b = n_2(x, y, m)
    for i in range(k):
        x3, y3 = n_3(d, b, len(x))
        x4, y4 = n_4(x3, y3)
        res = arch_model(x4).fit(update_freq=5).params
        h1.append(res[0])
        h2.append(res[2])
        h3.append(res[3])
    return(sorted(h1), sorted(h2), sorted(h3))

def estimate(m=m1, k=k1):
    beta = [[], [], []]
    beta_25 = [[], [], []]
    beta_75 = [[], [], []]
    x = generate()
    for l in m:
        qqq = simple_boot(x, l, k)
        for step in range(3):
            beta[step].append((np.mean(qqq[step])))
            beta_25[step].append(qqq[step][round((k-1)*0.025)])
            beta_75[step].append(qqq[step][round((k-1)*0.975)])
    return(beta, beta_25, beta_75)

def plot_graph(m=m1, k=k1):
    start_time = time.time()
    sigma_l, q_l_025, q_l_975 = estimate(m, k)
    print("--- %s seconds ---" % (time.time() - start_time))
    print(sigma_l, q_l_025, q_l_975)
    xlist = m
    ylist = sigma_l[0]
    ylist1 = [-4.3740584650247047 for i in range(len(m))]
    ylist_25 = q_l_025[0]
```

```
    ylist_75 = q_l_975[0]
    pylab.plot (xlist, ylist, color = "k")
    pylab.plot (xlist, ylist1, color = "k")
    pylab.plot (xlist, ylist_25, linestyle = "--", color = "k")
    pylab.plot (xlist, ylist_75, linestyle = "--", color = "k")
    pylab.xlabel('Число порогов')   # подпись оси OX
    pylab.ylabel(u"\u03C6%s" % 0)   # подпись оси OY
    pylab.show()
    ylist = sigma_l[1]
    ylist1 = [-2.5319982573218507 for i in range(len(m))]
    ylist_25 = q_l_025[1]
    ylist_75 = q_l_975[1]
    pylab.plot (xlist, ylist, color = "k")
    pylab.plot (xlist, ylist1, color = "k")
    pylab.plot (xlist, ylist_25, linestyle = "--", color = "k")
    pylab.plot (xlist, ylist_75, linestyle = "--", color = "k")
    pylab.xlabel('Число порогов')   # подпись оси OX
    pylab.ylabel(u"\u03C6%s" % 1)   # подпись оси OY
    pylab.show()

    ylist = sigma_l[2]
    ylist1 = [0.9089 for i in range(len(m))]
    ylist_25 = q_l_025[2]
    ylist_75 = q_l_975[2]
    pylab.plot (xlist, ylist, color = "k")
    pylab.plot (xlist, ylist1, color = "k")
    pylab.plot (xlist, ylist_25, linestyle = "--", color = "k")
    pylab.plot (xlist, ylist_75, linestyle = "--", color = "k")
    pylab.xlabel('Число порогов')   # подпись оси OX
    pylab.ylabel("\u03C6%s" % 2)   # подпись оси OY
    pylab.show()
    print("--- %s seconds ---" % (time.time() - start_time))
    return(0)

plot_graph(m=m1, k=k1)
```

Приложение 2. Программная реализация простого и точного бутстрап методов, а также доверительных интервалов SB. PB, BCa на python

```
from itertools import combinations_with_replacement, permutations
import numpy as np
import pandas as pd
import numpy as np
from scipy.stats import norm

alpha = 0.05

def estimate_coef(y, x):
    a = (np.mean(y*x)-np.mean(x)*np.mean(y))/(np.mean(x**2)-np.mean(x)**2)
    b = np.mean(y)-a*np.mean(x)
    return(a, b)

def sample_bootstrap(y, x, replicates, generate_list):
    estimate_B = []
    Col = 0
    for index, value in enumerate([replicates[i] for i in generate_list]):
        if len(set(value)) == 1:
            Col += 1
            continue
        Y_B_estimate, X_B_estimate = [], []
        for j in value:
            Y_B_estimate.append(y[j-1])
            X_B_estimate.append(x[j-1])
        if len(set(X_B_estimate)) == 1:
            Col += 1
            continue
        estimate_B.append(estimate_coef(np.array(Y_B_estimate),
np.array(X_B_estimate)))
    return(list(zip(*estimate_B)), Col)

def sample_bootstrap_exact(y, x, replicates):
    estimate_B = []
    Col = 0
    for value in replicates:
        if len(set(value)) == 1:
            Col += 1
            continue
        Y_B_estimate, X_B_estimate = [], []
        for j in value:
            Y_B_estimate.append(y[j-1])
            X_B_estimate.append(x[j-1])
```

```
        if len(set(X_B_estimate)) == 1:
            Col += 1
            continue
        estimate_B.append(estimate_coef(np.array(Y_B_estimate),
np.array(X_B_estimate)))
    return(list(zip(*estimate_B)), Col)

def sample_bootstrap_without_1(y, x, size_without1, N):
    estimate_B = []
    Col = 0
    for value in size_without1:
        if len(set(value)) == 1:
            Col += 1
            continue
        Y_B_estimate, X_B_estimate = [], []
        for k in value:
            Y_B_estimate.append(y[k-1])
            X_B_estimate.append(x[k-1])
        if len(set(X_B_estimate)) == 1:
            Col += 1
            continue
        estimate_B.append(estimate_coef(np.array(Y_B_estimate),
np.array(X_B_estimate)))
    return(list(zip(*estimate_B)), Col)



def SB_basic(y, x, replicates, generate_list, B):
    theta_cap = estimate_coef(y, x)
    theta_cap_asterisk, Col = sample_bootstrap(y, x, replicates,
generate_list)
    theta_cap_asterisk_mean = np.mean(theta_cap_asterisk[0]),
np.mean(theta_cap_asterisk[1])
    SE_asterisk_basic = (np.sqrt(np.sum((theta_cap_asterisk[0]-
theta_cap_asterisk_mean[0])**2)/(B-1)),
                         np.sqrt(np.sum((theta_cap_asterisk[1]-
theta_cap_asterisk_mean[1])**2)/(B-1)))
    a, b = ((theta_cap[0] + norm.ppf(alpha/2)*SE_asterisk_basic[0],
             theta_cap[0] + norm.ppf(1 - alpha/2)*SE_asterisk_basic[0]),
            (theta_cap[1] + norm.ppf(alpha/2)*SE_asterisk_basic[1],
             theta_cap[1] + norm.ppf(1 - alpha/2)*SE_asterisk_basic[1]))
    return(a, b)

def SB_exact(y, x, replicates, N):
    theta_cap = estimate_coef(y, x)
    theta_cap_asterisk_exact, Col = sample_bootstrap_exact(y, x, replicates)
    theta_cap_asterisk_exact_mean = np.mean(theta_cap_asterisk_exact[0]),
np.mean(theta_cap_asterisk_exact[1])
    SE_asterisk_exact = (np.sqrt(np.sum((theta_cap_asterisk_exact[0]-
theta_cap_asterisk_exact_mean[0])**2)/(N**N)),
```

```
                        np.sqrt(np.sum((theta_cap_asterisk_exact[1]-
theta_cap_asterisk_exact_mean[1])**2)/(N**N)))
    a, b = ((theta_cap[0] + norm.ppf(alpha/2)*SE_asterisk_exact[0],
             theta_cap[0] + norm.ppf(1 - alpha/2)*SE_asterisk_exact[0]),
            (theta_cap[1] + norm.ppf(alpha/2)*SE_asterisk_exact[1],
             theta_cap[1] + norm.ppf(1 - alpha/2)*SE_asterisk_exact[1]))
    return(a, b)



def PB_basic(y, x, replicates, generate_list, B):
    theta_cap_asterisk, Col = sample_bootstrap(y, x, replicates,
generate_list)
    a, b = ((np.sort(theta_cap_asterisk[0])[int(np.round(B*0.05/2))],
np.sort(theta_cap_asterisk[0])[int(np.round(B*(1-0.05/2)))]),
            (np.sort(theta_cap_asterisk[1])[int(np.round(B*0.05/2))],
np.sort(theta_cap_asterisk[1])[int(np.round(B*(1-0.05/2)))]))
    return(a, b)


def PB_exact(y, x, replicates, N):
    theta_cap_asterisk_exact, Col = sample_bootstrap_exact(y, x, replicates)
    a, b =
((np.sort(theta_cap_asterisk_exact[0])[int(np.round(N**N*0.05/2))],
             np.sort(theta_cap_asterisk_exact[0])[int(np.round(N**N*(1-
0.05/2)))]),

(np.sort(theta_cap_asterisk_exact[1])[int(np.round(N**N*0.05/2))],
             np.sort(theta_cap_asterisk_exact[1])[int(np.round(N**N*(1-
0.05/2)))]))
    return(a, b)



def BCa_basic(y, x, B, N, replicates, generate_list, size_without1):
    theta_cap_asterisk, Col = sample_bootstrap(y, x, replicates,
generate_list)
    B -= Col
    theta_cap_asterisk_mean = np.mean(theta_cap_asterisk[0]),
np.mean(theta_cap_asterisk[1])
    theta_cap_without_1, Col_1 = sample_bootstrap_without_1(y, x,
size_without1, N)
    theta_cap_without_1_mean = np.mean(theta_cap_without_1[0]),
np.mean(theta_cap_without_1[1])
    Z = (norm.ppf(np.sum(list([1 for i in range(B) if
theta_cap_asterisk[0][i] < theta_cap_asterisk_mean[0]]))/B),
        norm.ppf(np.sum(list([1 for i in range(B) if theta_cap_asterisk[1][i]
< theta_cap_asterisk_mean[1]]))/B))
    A = (np.sum((theta_cap_without_1[0] - theta_cap_without_1_mean[0])**3 / \
                (6*(np.sum((theta_cap_without_1[0] -
theta_cap_without_1_mean[0])**2))**(3/2))),
        np.sum((theta_cap_without_1[1] - theta_cap_without_1_mean[1])**3 / \
                (6*(np.sum((theta_cap_without_1[1] -
theta_cap_without_1_mean[1])**2))**(3/2))))
```

```
    A_1 = (norm.cdf(Z[0] + (Z[0] + norm.ppf(alpha/2))/(1-A[0]*(Z[0] +
norm.ppf(alpha/2)))),
          norm.cdf(Z[1] + (Z[1] + norm.ppf(alpha/2))/(1-A[1]*(Z[1] +
norm.ppf(alpha/2)))))
    A_2 = (norm.cdf(Z[0] + (Z[0] + norm.ppf(1 - alpha/2))/(1-A[0]*(Z[0] +
norm.ppf(1 - alpha/2)))),
          norm.cdf(Z[1] + (Z[1] + norm.ppf(1 - alpha/2))/(1-A[1]*(Z[1] +
norm.ppf(1 - alpha/2)))))
    a, b = ((np.sort(theta_cap_asterisk[0])[int(np.round(B*A_1[0]))-1],
np.sort(theta_cap_asterisk[0])[int(np.round(B*A_2[0]))-1])
        ,(np.sort(theta_cap_asterisk[1])[int(np.round(B*A_1[1]))-1],
np.sort(theta_cap_asterisk[1])[int(np.round(B*A_2[1]))-1]))
    return(a, b)


def BCa_exact(y, x, B, N, replicates, generate_list, size_without1):
    theta_cap_asterisk, Col = sample_bootstrap_exact(y, x, replicates)
    NN = N**N-Col
    theta_cap_asterisk_mean = np.mean(theta_cap_asterisk[0]),
np.mean(theta_cap_asterisk[1])
    theta_cap_without_1, Col_1 = sample_bootstrap_without_1(y, x,
size_without1, N)
    theta_cap_without_1_mean = np.mean(theta_cap_without_1[0]),
np.mean(theta_cap_without_1[1])
    Z = (norm.ppf(np.sum(list([1 for i in range(NN) if
theta_cap_asterisk[0][i] < theta_cap_asterisk_mean[0]]))/(N**N)),
        norm.ppf(np.sum(list([1 for i in range(NN) if
theta_cap_asterisk[1][i] < theta_cap_asterisk_mean[1]]))/(N**N)))
    A = (np.sum((theta_cap_without_1[0] - theta_cap_without_1_mean[0])**3 / \
              (6*(np.sum((theta_cap_without_1[0] -
theta_cap_without_1_mean[0])**2))**(3/2))),
        np.sum((theta_cap_without_1[1] - theta_cap_without_1_mean[1])**3 / \
              (6*(np.sum((theta_cap_without_1[1] -
theta_cap_without_1_mean[1])**2))**(3/2))))
    A_1 = (norm.cdf(Z[0] + (Z[0] + norm.ppf(alpha/2))/(1-A[0]*(Z[0] +
norm.ppf(alpha/2)))),
          norm.cdf(Z[1] + (Z[1] + norm.ppf(alpha/2))/(1-A[1]*(Z[1] +
norm.ppf(alpha/2)))))
    A_2 = (norm.cdf(Z[0] + (Z[0] + norm.ppf(1 - alpha/2))/(1-A[0]*(Z[0] +
norm.ppf(1 - alpha/2)))),
          norm.cdf(Z[1] + (Z[1] + norm.ppf(1 - alpha/2))/(1-A[1]*(Z[1] +
norm.ppf(1 - alpha/2)))))
    a, b = ((np.sort(theta_cap_asterisk[0])[int(np.round(NN*A_1[0]))-1],
np.sort(theta_cap_asterisk[0])[int(np.round(NN*A_2[0]))-1])
        ,(np.sort(theta_cap_asterisk[1])[int(np.round(NN*A_1[1]))-1],
np.sort(theta_cap_asterisk[1])[int(np.round(NN*A_2[1]))-1]))
    return(a, b)
np.random.seed(2)
B = 1000
fixed_df = pd.read_csv('city.csv', sep=',', encoding='latin1')
y = np.array(fixed_df['u'])
x = np.array(fixed_df['x'])
```

```
def build_x(size, a=4, b=2):
    eps = np.random.uniform(low=0, high=1, size=size)
    normal = np.random.normal(loc=0, scale=1, size = size)
    exponential = np.random.exponential(scale=1, size=size)
    laplace = np.random.laplace(loc=0, scale=1, size = size)
    poisson = np.random.poisson(lam=3, size=size)
    y_normal = b + a*normal + eps
    y_exponential = b + a*exponential + eps
    y_laplace = b + a*laplace + eps
    y_poisson = b + a*poisson + eps
    return((normal, exponential, laplace, poisson),
           (y_normal, y_exponential, y_laplace, y_poisson))

def replacement(size):
    """
    generation different combinators
    """
    x = list(set(permutations(i, size)) for i in
combinations_with_replacement(range(1,size+1), size))
    replicates = set()
    for i in x:
        replicates.update(i)
    return(list(replicates))

def generate_bootstrap(size, B):
    generate_list = np.random.choice(size**size, size=B, replace=False)
    return(list(generate_list))

def size_without_1(size):
    list_without_1 = []
    for index in range(size):
        list_without_1.append(list([i+1 for i in range(size) if index != i]))
    return(np.array(list_without_1))

def replacement_without_1(size):
    replicates_without_1 = list()
    size_without1 = size_without_1(size)
    for i in range(size):
        x = list(set(permutations(i, size-1)) for i in
combinations_with_replacement(size_without1[i], size-1))
        replicates = set()
        for i in x:
            replicates.update(i)
        replicates_without_1.append(list(replicates))
    return(replicates_without_1)
```