Оглавление

[Введение 2](#_Toc164004868)

[1. Базовый метод трансформации Шенкса 3](#_Toc164004869)

[2. Многошаговый алгоритм Шенкса и – алгоритм 5](#_Toc164004870)

[3. – алгоритм 16](#_Toc164004871)

[4. Сравнение алгоритмов 22](#_Toc164004872)

[Заключение 35](#_Toc164004873)

[Список литературы 36](#_Toc164004874)

[Приложение 1 38](#_Toc164004875)

# Введение

Метод Шенкса был разработан в 1955 году американским математиком Даниэлем Шенксом и с тех пор является важным инструментом в численном анализе и методах решения уравнений [5]. Он предназначен для ускорения сходимости последовательностей и суммируемых рядов. Скорость и порядок сходимости – это величины, которые представляют, насколько быстро последовательность приближается к своему пределу. Последовательность , сходящаяся к , имеет порядок сходимости и скорость сходимости μ, если . Во многих задачах, где требуется численное приближение, бывает необходимо вычислить сумму бесконечного ряда или оценить предел последовательности, однако, некоторые ряды сходятся медленно и требуют большого количества итераций для достижения точности.

Ускорение достигается путем изменения ряда с сохранением суммы. Это позволяет достичь лучшей сходимости и значительно уменьшить количество итераций.

# 1. Базовый метод трансформации Шенкса

Сущность метода Шенкса состоит в том, что по заданной последовательности строится новая. Нам необходимо определить – частичную сумму бесконечной последовательности (1), S – предел последовательности (2):

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
| . |  |

Трансформация нацелена на ускорение сходимости S. Для этого запишем предел в виде (3), теперь у нас три неизвестные: S, ɣ, ν, где <1, а

ν - параметр [3].

|  |  |
| --- | --- |
| . |  |

Их можно рассчитать на основе частичных сумм (4):

|  |  |
| --- | --- |
| . |  |

Предел S представлен в виде преобразования частичной суммы порядка n:

|  |  |
| --- | --- |
| . |  |

Важным свойством этого преобразования является то, что при повторном применении получается новая последовательность , имеющую на 2 члена меньше. Эту процедуру продолжаем пока в последовательности не останется 1 или 2 члена. (5) можно преобразовать в равенство (6), что является нашим первым шагом в процедуре трансформации:

|  |  |
| --- | --- |
| . |  |

# 2. Многошаговый алгоритм Шенкса и – алгоритм

Далее – последовательность, S – предел, – новая последовательность. Преобразование последовательности называется методом ускорения сходимости или методом экстраполяции.

Пусть последовательность, сходящаяся к пределу S, удовлетворяющему

|  |  |
| --- | --- |
|  |  |

Когда , последовательность сходится линейно[[1]](#footnote-1), когда , последовательность сходится логарифмически[[2]](#footnote-2), а когда , у нее сверхлинейная[[3]](#footnote-3)сходимость. Преобразование последовательности преобразует эту последовательность в новую последовательность , которая быстрее сходится к тому же пределу S при некоторых предположениях, то есть

|  |  |
| --- | --- |
|  |  |

Идея, лежащая в основе таких преобразований, состоит в предположении, что преобразуемая последовательность ведет себя как модельная последовательность[[4]](#footnote-4). Множество этих модельных последовательностей называется ядром преобразования. Оно характеризуется тем, что каждая из его последовательностей (последовательности интерполирующих значений, которые аппроксимируют поведение исходной последовательности ) удовлетворяет заданному алгебраическому соотношению в зависимости от своего неизвестного предела (или антипредела[[5]](#footnote-5), если он не сходится), и от *p* неизвестных параметров и имеет вид:

|  |  |
| --- | --- |
| . |  |

Это отношение называется неявной формой ядра. представляет собой действительную матрицу вращения элементов. Она вращает компоненты вектора в двумерном подпространстве, но не влияет на компоненты вектора в оставшемся () - мерном подпространстве. Рассматривая определенное количество последовательных членов подлежащей преобразованию , начиная с индекса n, то есть , мы ищем последовательность , удовлетворяющую следующим условиям:

|  |  |
| --- | --- |
|  |  |

что есть

|  |  |
| --- | --- |
| . |  |

Решая эту систему, мы получим *p + 1* неизвестных *a* и *u.* Неизвестное значение *u* является пределом последовательности и было получено путем экстраполяции. Поскольку оно зависит от *n*, то обозначим его через Следовательно, последовательность была преобразована в новую последовательность Если последовательность ( принадлежит ядру , то , её точный предел (или антипредел). Решение (7) дает замкнутую форму для , которая называется явной формой ядра. является обозначением для последовательности, полученной в результате применения преобразования T к исходной последовательности ().

*Способ 1.* Рассматриваем скалярную последовательность, удовлетворяющую соотношению:

|  |  |
| --- | --- |
|  |  |

где – неизвестные константы, сумма которых не равна 0 и (иначе k должно быть заменено на меньшее значение, так как порядок этого разностного уравнения уменьшается).

Последовательность (10) является обобщением последовательности вида

|  |  |
| --- | --- |
|  |  |

где известна

– условие нормализации. Набор последовательностей, удовлетворяющий этому разностному уравнению, называется ядром Шенкса. Опять же возникает проблема в вычислении неизвестного S. Когда , из (10) получается (10.1)

Поскольку (10) содержит *k+1* неизвестный элемент, и мы не знаем *S*, напишем еще соотношение для индексов *n,…,n+k*, которое представляет собой однородную линейную систему с ненулевым решением:

|  |  |
| --- | --- |
| . |  |

Таким образом, ее определитель должен быть равен нулю, иначе система будет иметь нулевое решение:

|  |  |
| --- | --- |
| . |  |

Вычтем из каждой строки предыдущую и запишем определитель, как разность двух определителей:

|  |  |
| --- | --- |
|  |  |

где , , .

Если не удовлетворяет (10), то определители всё ещё могут быть вычислены, но их отношение больше не будет равно *S* для всех *n*, а равно числу, зависящему от *k* и *n,* и обозначается . Последовательность может не удовлетворять уравнению (10) из-за несоответствия начальных условий или других факторов, которые могут влиять на значения коэффициентов или самой последовательности. Таким образом, последовательность была преобразована в набор последовательностей {}. Это и есть определение трансформации Шенкса.

Для получения формулы трансформации в числителе происходит замена каждой строки на её сумму с предыдущей, а в знаменателе каждого столбца – на его разность с предыдущим, что приводит к следующему выражению:

|  |  |
| --- | --- |
|  |  |

где обозначает определитель Ганкеля:

|  |  |
| --- | --- |
|  |  |

причем . Из-за своей особой формы такой определитель также называется симметричным в том смысле, что все его элементы на любой диагонали под прямым углом к главной диагонали одинаковы.

Выражения (14) и (15) могут быть рекурсивно вычислены с помощью тождества Сильвестра, которое в данном случае дает:

|  |  |
| --- | --- |
|  |  |

учитывая и .

Распишем (16) более подробно:

Тождество Сильвестра утверждает, что для произвольной матрицы выполнено:

где это подматрица , образованная пересечением подмножества строк u и столбцов w исходной матрицы, а определитель матрицы .

Тогда в формуле (16) подразумевается, что:

,

,

,

,

,

где подмножество первых строк, подмножество всех строк, кроме первой, подмножество первых столбцов, подмножество всех столбцов, кроме первого.

Поскольку , данное отношение можно вычислить, применяя отдельно предыдущее рекуррентное соотношение к его числителям и знаменателям. Шенкс действовал таким образом для рекурсивной реализации своего преобразования.

Формула (14) показывает, что представляет собой линейную комбинацию , коэффициенты которой обозначены через зависят от *n* (также и от *k*).

Заменяя каждую строку на её разность с предыдущей и повторяя эту операцию несколько раз и выполняя её также над столбцами, получим:

Заменяя в числителе и знаменателе (14) каждый столбец, начиная со второго, на его разность с предыдущим, получим:

Эта формула показывает, что благодаря детерминантной формуле Шура [10], является дополнением Шура, то есть

*Способ 2.* Трансформацию Шенкса можно получить другим путем. Из (10) уравнения, при раскрытии скобок и применении условия нормировки, получается:

|  |  |
| --- | --- |
|  |  |

где удовлетворяет системе линейных уравнений:

|  |  |
| --- | --- |
| . |  |

По правилу Крамера:

|  |  |
| --- | --- |
| . |  |

Преобразуем (19) для получения формулы преобразования Шенкса (14).

Разложим по его i-му столбцу . Добавим нулевую первую строку, заменяя в i-ом столбце ноль на единицу, . При таком преобразовании сохраняется размерность определителя и его значение, т.к. при вычислении, путем разложения по первой строке, получаем . Далее:

|  |  |
| --- | --- |
| . |  |

Умножая каждый на , а затем суммируя их и используя (17), получаем (14).

*Способ 3.* Этот способ получения преобразования Шенкса основывается на том, что (10) эквивалентно:

|  |  |
| --- | --- |
| . |  |

Запись этого соотношения для индексов *n, …, n + k* приводит к системе:

|  |  |
| --- | --- |
| . |  |

Покажем, что решение (22) для неизвестного *S* дает (14).

По правилу Крамера выразим , при транспонировании матрицы определитель не меняется, поэтому равенство можно переписать: , что равно (14).

*Способ 4.* Четвертая возможность получения преобразования Шенкса состоит в рассмотрении следующей линейной системы, где *с* – любая ненулевая константа:

|  |  |
| --- | --- |
|  |  |

где и содержит .

Система (23) получена из (11) уравнения, при раскрытии скобок и применении условия нормировки.

*Замечание.* Первые два способа получения преобразования Шенкса как отношения детерминантов могут быть обобщены на случай, когда являются элементами общего векторного пространства.

*Теорема 1*.

Достаточным условием того, что для всех n, является то, что последовательность удовлетворяет (7) вместе с , где – неизвестные константы, сумма которых не равна 0 (иначе k должно быть заменено на меньшее значение, так как порядок этого разностного уравнения уменьшается) и. Если , то условие тоже необходимо [1].

*Доказательство.*

Следуя (14), мы имеем , где является решением системы (18) и задается формулами (19).



Тогда условие упрощает . Используя (20), мы получаем , и легко выводим:

|  |  |
| --- | --- |
| . |  |

Аналогично

|  |  |
| --- | --- |
| . |  |

Из уравнений, соответствующих *i = 1, …, k* из этих двух групп получается

|  |  |
| --- | --- |
|  |  |

Поскольку коэффициенты и суммируются до 1, получается

|  |  |
| --- | --- |
| . |  |

Таким образом, мы получаем однородную систему *k +1* уравнений в *k +1* неизвестных

|  |  |
| --- | --- |
|  |  |

которое, как и предполагалось, отличалось от нуля для всех *n*. Следовательно, предыдущая однородная система неособая, и из этого следует, что ее решение равно нулю, что означает = , что не зависит от *n* при *i = 0, …, k.* Из (14) (или (19) и (20)) мы видим, что числитель коэффициента равен . К тому же, и . Условие больше не требуется для доказательства необходимого условия теоремы 1, если преобразование Шенкса реализовано с помощью - алгоритма.

3. – алгоритм

Теперь, поскольку детерминанты долго вычислять, необходим другой алгоритм реализации преобразования Шенкса. – алгоритм является рекурсивным алгоритмом, разработанным для реализации Шенкса без вычисления определителей Ганкеля, фигурирующих в (14) [1].

Основное правило [11]:

Если

Тогда

|  |  |
| --- | --- |
| , |  |

учитывая и

Связь между – алгоритмом и преобразованием Шенкса определяется выражением:

|  |  |
| --- | --- |
|  |  |

Таким образом, являются промежуточным результатом:

|  |  |
| --- | --- |
| . |  |

Величины обычно отображаются в виде двумерного массива ( -массив), где нижний индекс *k* остается неизменным в столбце таблицы, а верхний индекс *n* остается неизменным по нисходящей диагонали:

|  |  |
| --- | --- |
| *.* |  |

Правило (29) – алгоритма связывает величины, расположенные в четырех вершинах ромба, в трёх столбцах и двух нисходящих диагоналях:

|  |  |
| --- | --- |
|  |  |

Для эффектной реализации – алгоритма, наилучший метод, по мнению Винна [6,7], состоит в сохранении последней возрастающей диагонали -массива (в этой диагонали сумма нижнего и верхнего индексов постоянна, например, и складывая по одному, то есть ) члены последовательности, подлежащей преобразованию. Это позволяет уйти от хранения величин в двумерном массиве к хранению в одномерном. Данные, которые теперь не будут запоминаться могут быть восстановлены с помощью соотношений, представленных ранее.

*Теорема 2.*

|  |  |
| --- | --- |
| . |  |

Этот результат показывает, что значения являются промежуточными и необходимы для вычисления величин с четными индексами [4].

У нас так же есть следующее уравнение:

|  |  |
| --- | --- |
| . |  |

Поскольку в рекурсивной формуле есть возможность идти с шагом 2, то члены с нечетным нижним индексом могут быть исключены.

Из правила (35) – алгоритма имеем:

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
| . |  |

Вычитая первое соотношение из второго, левая часть становится

|  |  |
| --- | --- |
|  |  |

и мы получаем перекрестное правило:

|  |  |
| --- | --- |
| . |  |

при начальных условиях для *n* = 0, 1, ….

Это правило касается таблицы, где сохранены только величины с меньшим индексом того же соотношения, пять величин, обозначаемых сторонами света и отображаемых следующим образом:

|  |  |
| --- | --- |
|  |  |

и перекрестное правило записывается (где *С* – центр):

|  |  |
| --- | --- |
| . |  |

Когда две величины в знаменателе равны (или близки), происходит деление на ноль (или на малую величину). Основное правило, как показал Винн, позволяет избежать такую сингулярность[[6]](#footnote-6) и продолжить вычисление [6,12]. А именно:

или мы можем это переписать, как , где a – числитель дроби, написанной ранее. Отсюда видно, что если и близки по значению, то является неопределенным, но , и вычисляются из основного правила и тождества, представленного выше. А если и равны, то перекрестное правило для алгоритма становится проще: [13]. Для данного метода в статье [13] представлен код, его блок-схема представлена в Приложении 1.

Предположим, что условие не выполняется. Тогда это означает, что существует, по крайней мере, один индекс *n* такой, что .

Теперь реализуем преобразование Шенкса, используя – алгоритм без его основного правила (31).

Поскольку , мы имеем неопределенным.

Далее предположим, что . Таким образом, является неопределенным. То же самое с . Такое обоснование применимо и к +1.



Значит, индекс изолированный, такой, что , то есть не может быть одновременно и (или ).

Поскольку или и, аналогично, .

Теперь так как изначальное предположение состояло в том, что (что, очевидно, означает, что эти величины могут быть вычислены), оно содержит Из-за опредленных условий алгоритма, описанных выше, при вычислении следующих столбцов происходит деление на ноль, и алгоритм должен быть остановлен. Мы переходим к *теореме 3*.

*Теорема 3.*

Необходимым и достаточным условием того, что для всех n, является то, что последовательность удовлетворяет (7) вместе с и[2].

Разница между *теоремой 1* и *теоремой 3* заключается в том, что *теорема 1* касается преобразования Шенкса, когда как *теорема 3* связана с рекурсивным алгоритмом, используемым для его реализации. Поскольку преобразование Шенкса может быть реализовано и другими алгоритмами, для их правильной работы, возможно, придется повторно ввести условие , отсутствующее в *теореме 3*, и тогда, *теорема 3* справедлива только для – алгоритма.



4. Сравнение алгоритмов

Сравним многошаговое преобразование Шенкса и – алгоритм. Оба алгоритма хорошо подходят для осциллирующих[[7]](#footnote-7) последовательностей и плохо подходят для монотонных[[8]](#footnote-8), т.к. происходит накопление ошибки [2]. Также преобразование Шенкса подходит для ускорения рядов с факториалами и тригонометрическими функциями, что показал Винн, изучая сходимость рядов Ньютона и Дирихле [2]. Но главным недостатком преобразования является затрата сил на вычисления определителей, чего нет у – алгоритма. – алгоритм используется для неявных и устойчивых методов интегрирования Рунге-Кутта, а также для решения краевых задач системы обыкновенных дифференциальных уравнений [8]. Но точно сказать, что последовательность, преобразованная с помощью –алгоритма, будет намного лучше сходится относительно искомой нельзя, ускорение может быть малозаметным [14].

Точность каждого метода зависит от множества факторов, таких как исходный ряд, метод ускорения и параметры алгоритма. В общем случае, чем больше количество членов ряда, тем точнее результат.

Рассмотрим работу двух алгоритмов на конкретных примерах. Ниже предоставлены графики, демонстрирующие работу алгоритмов ускорения сходимости рядов, используемых в проекте, для первых 25 членов каждого из рассматриваемых ниже рядов. В случае, если какая-то из функций расчета частичной суммы ускоренного ряда выкидывает исключение (деление на ноль), график будет обрываться.

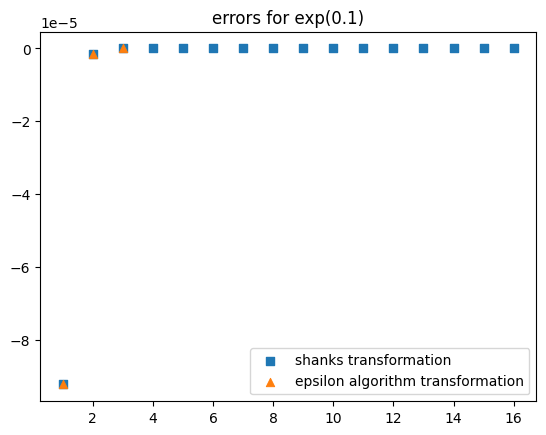


Рисунок 1. График ошибок для преобразования Шенкса и 𝜀-алгоритма.

Таблица 1. Сравнение преобразования Шенкса и 𝜀-алгоритма на последней итерации.

|  |  |  |
| --- | --- | --- |
| номер итерации n | преобразования Шенкса | 𝜀-алгоритм |
| 3 | 0 | 0 |

Время работы алгоритмов:

Алгоритм Шенкса – 0.0163594 секунд

𝜀-алгоритм – 0.0016803 секунд

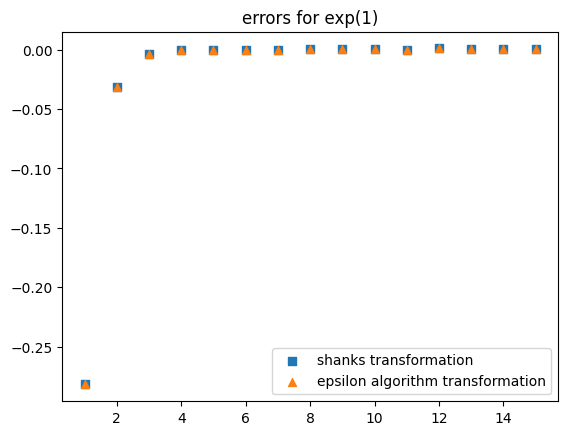


Рисунок 2. График ошибок для преобразования Шенкса и 𝜀-алгоритма.

Таблица 2. Сравнение преобразования Шенкса и 𝜀-алгоритма на последней итерации.

|  |  |  |
| --- | --- | --- |
| номер итерации n | преобразования Шенкса | 𝜀-алгоритм |
| 15 | 0.000800848 | 0.000800848 |

Время работы алгоритмов:

Алгоритм Шенкса – 0.0015706 секунд

𝜀-алгоритм – 0.0017533 секунд

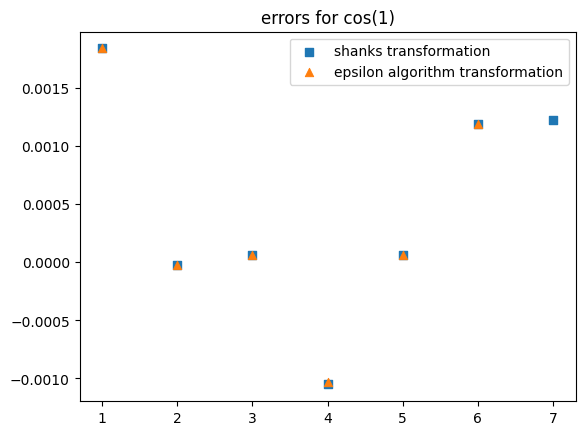


Рисунок 3. График ошибок для преобразования Шенкса и 𝜀-алгоритма.

Таблица 3. Сравнение преобразования Шенкса и 𝜀-алгоритма на последней итерации.

|  |  |  |
| --- | --- | --- |
| номер итерации n | преобразования Шенкса | 𝜀-алгоритм |
| 6 | 0.0011903 | 0.0011903 |

Время работы алгоритмов:

Алгоритм Шенкса – 0.0121402 секунд

𝜀-алгоритм – 0.0121285 секунд

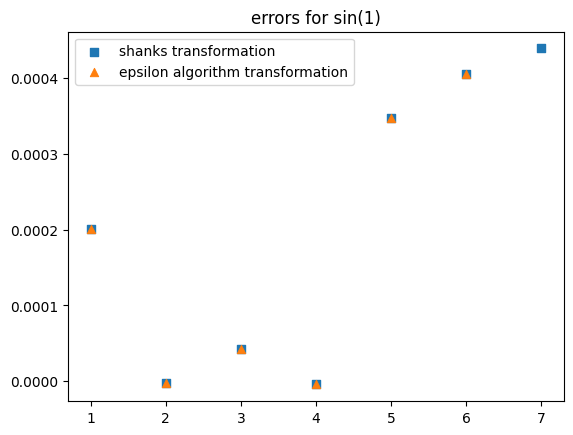


Рисунок 4. График ошибок для преобразования Шенкса и 𝜀-алгоритма.

Таблица 4. Сравнение преобразования Шенкса и 𝜀-алгоритма на последней итерации.

|  |  |  |
| --- | --- | --- |
| номер итерации n | преобразования Шенкса | 𝜀-алгоритм |
| 6 | 0.000405312 | 0.00040549 |

Время работы алгоритмов:

Алгоритм Шенкса – 0.0105809 секунд

𝜀-алгоритм – 0.0112633 секунд

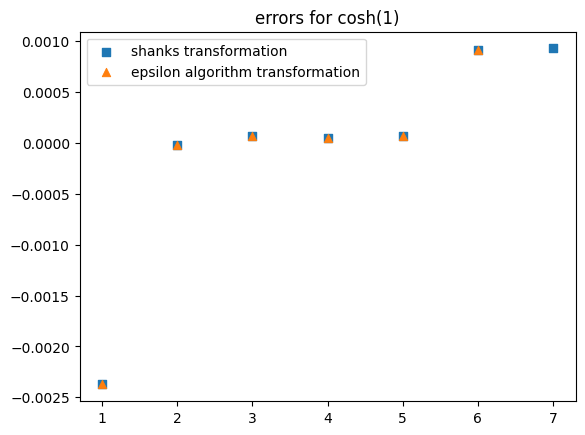


Рисунок 5. График ошибок для преобразования Шенкса и 𝜀-алгоритма.

Таблица 5. Сравнение преобразования Шенкса и 𝜀-алгоритма на последней итерации.

|  |  |  |
| --- | --- | --- |
| номер итерации n | преобразования Шенкса | 𝜀-алгоритм |
| 6 | 0.000911474 | 0.000911474 |

Время работы алгоритмов:

Алгоритм Шенкса – 0.0121832 секунд

𝜀-алгоритм – 0.0120484 секунд

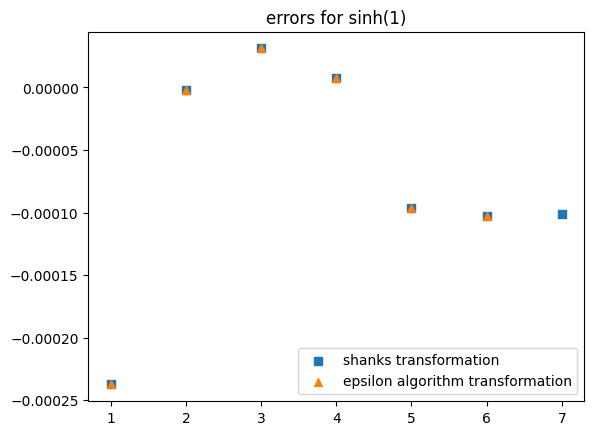


Рисунок 6. График ошибок для преобразования Шенкса и 𝜀-алгоритма.

Таблица 6. Сравнение преобразования Шенкса и 𝜀-алгоритма на последней итерации.

|  |  |  |
| --- | --- | --- |
| номер итерации n | преобразования Шенкса | 𝜀-алгоритм |
| 6 | -0.000102878 | -0.000102878 |

Время работы алгоритмов:

Алгоритм Шенкса – 0.0123199 секунд

𝜀-алгоритм – 0.0114201 секунд

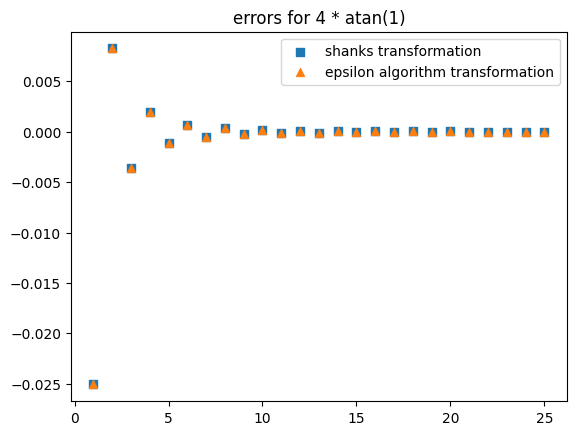


Рисунок 7. График ошибок для преобразования Шенкса и 𝜀-алгоритма.

Таблица 7. Сравнение преобразования Шенкса и 𝜀-алгоритма на последней итерации.

|  |  |  |
| --- | --- | --- |
| номер итерации n | преобразования Шенкса | 𝜀-алгоритм |
| 25 | -1.40667e-05 | -1.40667e-05 |

Время работы алгоритмов:

Алгоритм Шенкса – 0.0015034 секунд

𝜀-алгоритм – 0.00016972 секунд

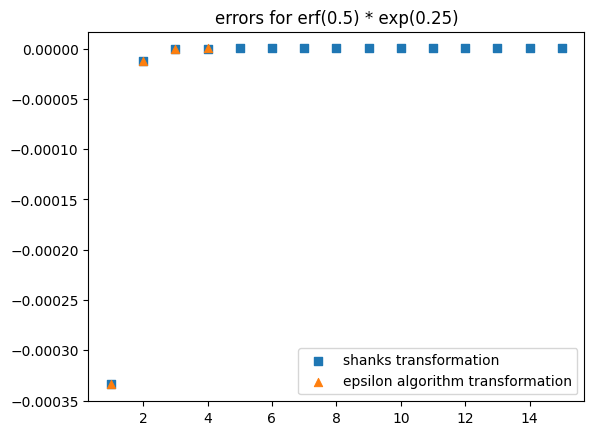


Рисунок 8. График ошибок для преобразования Шенкса и 𝜀-алгоритма.

Таблица 8. Сравнение преобразования Шенкса и 𝜀-алгоритма на последней итерации.

|  |  |  |
| --- | --- | --- |
| номер итерации n | преобразования Шенкса | 𝜀-алгоритм |
| 4 | 1.19209e-07 | 1.19209e-07 |

Время работы алгоритмов:

Алгоритм Шенкса – 0.0122545 секунд

𝜀-алгоритм – 0.0148476 секунд

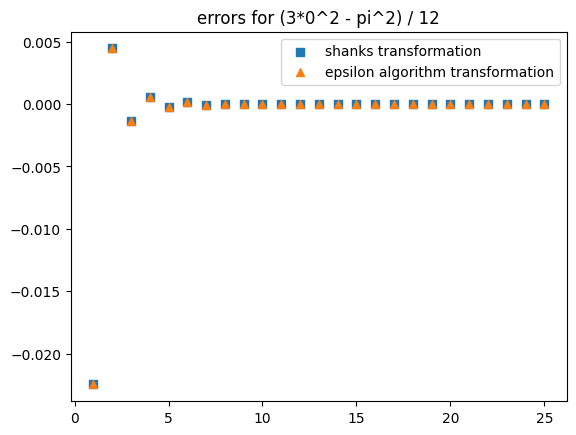


Рисунок 9. График ошибок в точке для преобразования Шенкса и 𝜀-алгоритма.

Таблица 9. Сравнение преобразования Шенкса и 𝜀-алгоритма на последней итерации.

|  |  |  |
| --- | --- | --- |
| номер итерации n | преобразования Шенкса | 𝜀-алгоритм |
| 25 | -5.96046e-07 | -5.96046e-07 |

Время работы алгоритмов:

Алгоритм Шенкса – 0.0015097 секунд

𝜀-алгоритм – 0.0016325 секунд

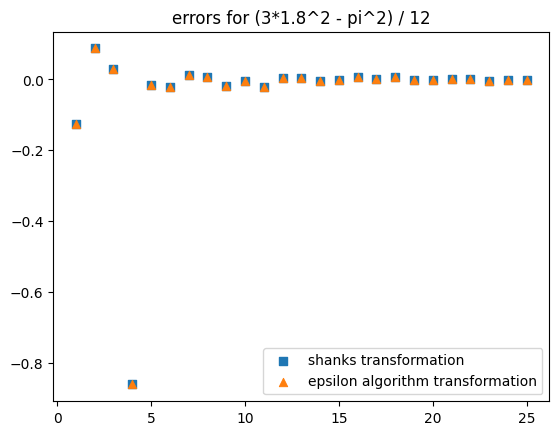


Рисунок 10. График ошибок в точке для преобразования Шенкса и 𝜀-алгоритма.

Таблица 10. Сравнение преобразования Шенкса и 𝜀-алгоритма на последней итерации.

|  |  |  |
| --- | --- | --- |
| номер итерации n | преобразования Шенкса | 𝜀-алгоритм |
| 25 | -0.00257336 | -0.00257338 |

Время работы алгоритмов:

Алгоритм Шенкса – 0.0019258 секунд

𝜀-алгоритм – 0.0020634 секунд

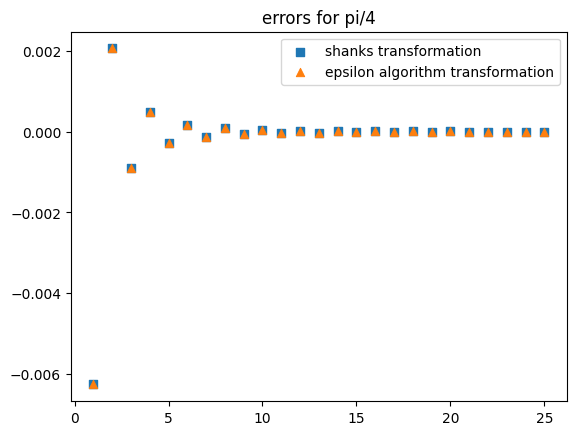


Рисунок 11. График ошибок

Таблица 11. Сравнение преобразования Шенкса и 𝜀-алгоритма на последней итерации.

|  |  |  |
| --- | --- | --- |
| номер итерации n | преобразования Шенкса | 𝜀-алгоритм |
| 25 | -3.51667e-06 | -3.51667e-06 |

Время работы алгоритмов:

Алгоритм Шенкса – 0.0014423 секунд

𝜀-алгоритм – 0.0009331 секунд

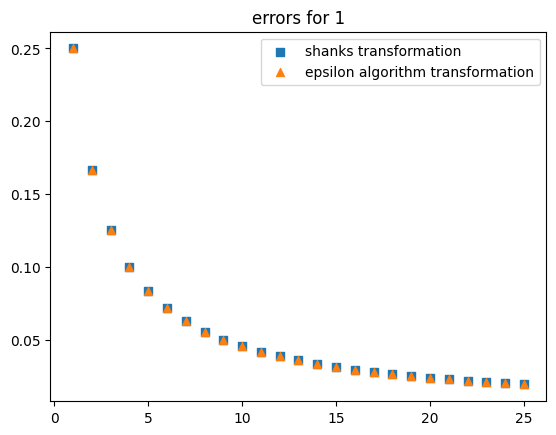


Рисунок 12. График ошибок

Таблица 12. Сравнение преобразования Шенкса и 𝜀-алгоритма на последней итерации.

|  |  |  |
| --- | --- | --- |
| номер итерации n | преобразования Шенкса | 𝜀-алгоритм |
| 25 | 0.0192305 | 0.0192314 |

Время работы алгоритмов:

Алгоритм Шенкса – 0.0015767 секунд

𝜀-алгоритм – 0.0014957 секунд

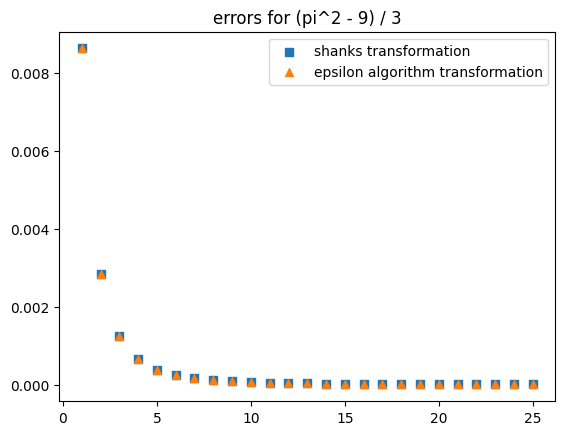


Рисунок 13. График ошибок

Таблица 13. Сравнение преобразования Шенкса и 𝜀-алгоритма на последней итерации.

|  |  |  |
| --- | --- | --- |
| номер итерации n | преобразования Шенкса | 𝜀-алгоритм |
| 25 | 4.70877e-06 | 4.38094e-06 |

Время работы алгоритмов:

Алгоритм Шенкса – 0.0024177 секунд

𝜀-алгоритм – 0.0017156 секунд

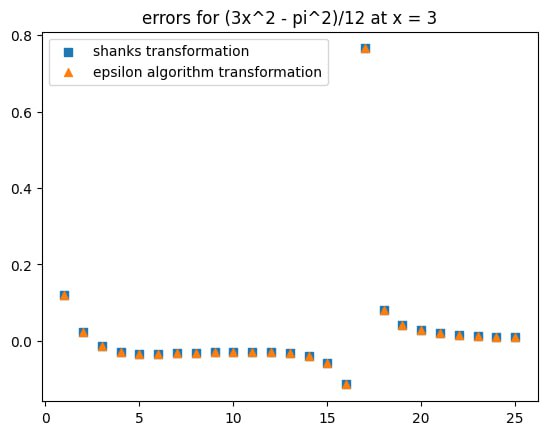


Рисунок 14. График ошибок в точке 3

Таблица 14. Сравнение преобразования Шенкса и 𝜀-алгоритма на последней итерации.

|  |  |  |
| --- | --- | --- |
| номер итерации n | преобразования Шенкса | 𝜀-алгоритм |
| 25 | 0.00949327 | 0.00949327 |

Время работы алгоритмов:

Алгоритм Шенкса – 0.0015664 секунд

𝜀-алгоритм – 0.0015756 секунд

Примерная оценка сложности алгоритмов при входных параметрах n и k:

Алгоритм Шенкса: (n+5)\*add + 2\*mult+div+3\*fma+(n+2)\*operator+[5\*add + mult + div + 3\*fma]\*[2k-2+k\*(k-1)/2].

𝜀-алгоритм: (n+3)\*add + mult + (2k+n)(swap+erase+operator) + [2\*add + div]\*[2k+n+k\*(2k+1)].

Где n – число членов ряда для аппроксимацииж k – порядок аппроксимации (количество шагов), add – операция сложения, mult – операция умножения, div – операция деления, fma - операция умножения–сложения с плавающей запятой, выполняемая за один шаг с одинарным округлением, operator – возвращает член ряда, swap – меняет указатель, erase – удаляет элемент вектора.

На рисунке 14 происходит скачек, так как порядок знаменателя и числителя совпадают, из-за чего происходит прибавление большого числа. Значения разницы между настоящей суммой и аппроксимацией для двух методов показывают, что результаты идентичны, но алгоритм преобразования Шенкса часто может посчитать больше преобразования для бо́льшего n, прежде чем программа выдаст исключение *division by zero*. Скорее всего, это связано с тем, что в алгоритме преобразования Шенкса в знаменателе стоит разность членов ряда , которая проходит проверку на отличие от нуля для бо́льших n, чем знаменатель разности промежуточных приближенных величин, полученных 𝜀-алгоритмом. На степенных и тригонометрических рядах алгоритм преобразования Шенкса работает немного быстрее, а на чисто числовых рядах - эпсилон алгоритм. Из оценки сложности алгоритмов нельзя при данной реализации сделать точный вывод о том, ккой из алгоритмов медленнее.

# Заключение

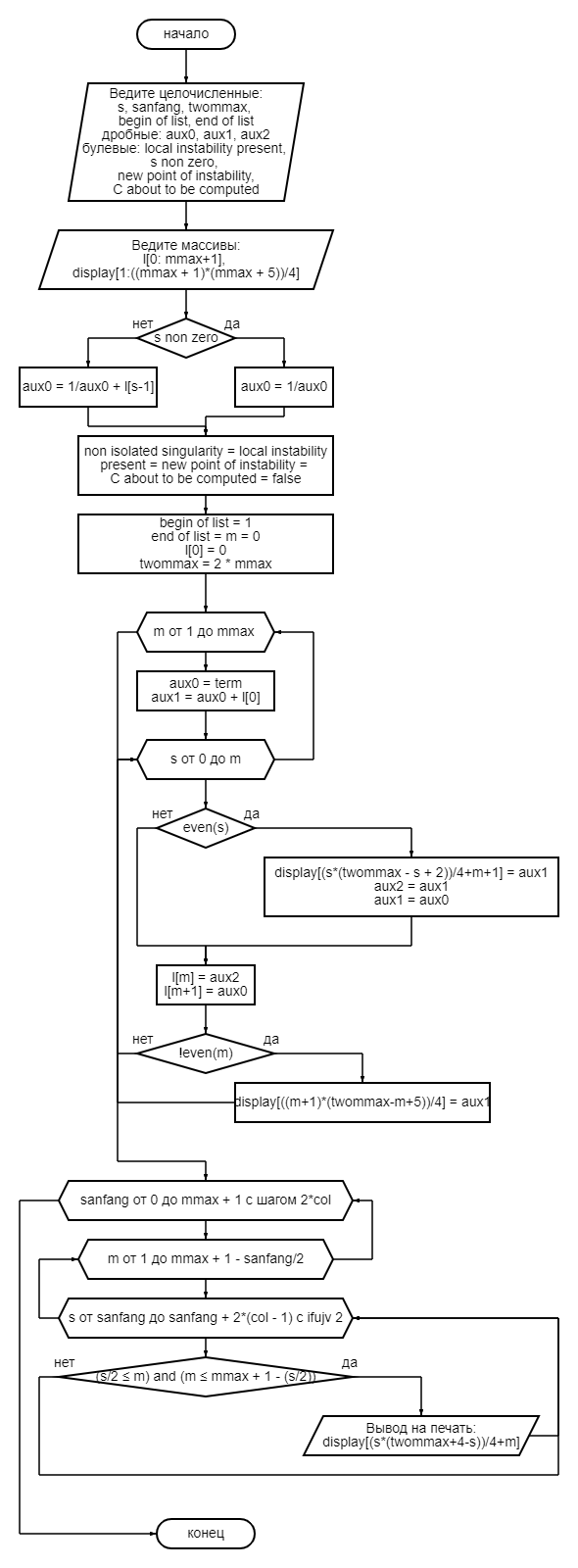
Метод Шенкса является мощным инструментом в численном анализе и численных методах решения уравнений. Он позволяет улучшить сходимость суммируемых рядов и повысить точность численных приближений. Из всех рассмотренных подходов к решению задачи нельзя сделать однозначный вывод о том, какой алгоритм лучше. У каждого есть свои преимущества.

# Список литературы

1. The genesis and early developments of Aitken’s process, Shanks’ transformation, the ε–algorithm, and related fixed point methods // Claude Brezinski & Michela Redivo-Zaglia - 2018. – P. 11 - 69.
2. On condition numbers of the Shanks transformation // M.N. Senhadji  - 1999. – P. 5 - 21.
3. Efficient Capacitance Extraction for Periodic Structures by Shanks Transformation // Ye Liu, Mei Xue, Zheng-Fan Li, Rui-Feng Xue  - 2004. – P. 1 - 5.
4. An Extended Multistep Shanks Transformation and Convergence Acceleration Algorithm with Their Convergence and Stability Analysis // Jian-Qing Sun, Xiang-Ke Chang, Xing-Biao Hu, Yi He - 2013. – P. 8 - 25.
5. Non-linear transformations of divergent and slowly convergent sequences // D. Shanks – 1955. – P. 1–42.
6. Acceleration techniques in numerical analysis, with particular references to problems in one independent variable // P. Wynn – 1962. – P. 149–156.
7. An arsenal of Algol procedures for the evaluation of continued fractions and for effecting the epsilon algorithm // P. Wynn - 1966 – P. 327–362.
8. Generalisations de la transformatiost de Shanks, de la table de pade et de L’ ε-algorithme // C. Brezinski – 1975. – P. 350-358.
9. Reanalysis of linear and nonlinear structures using iterated Shanks // Jorge Hurtado – 2002. – P. 4220 - 4222.
10. The Schur Complement and its Applications // Zhang - 2005. – P. 4 – 12.
11. On a device for computing the transformation // P. Wynn – 1956. – P. 91-96.
12. Upon Systems of Recursions which Obtain Among the Quotients of the Pade Table // P. Wynn – 1965. – P. 3- 6.
13. Singular rules for certain non-linear algorithms // P. Wynn – 1963. – P. 175 – 195.
14. On the convergence and stability of the epsilon algorithm // P. Wynn – 1966. – P. 100 – 110.

# Приложение 1

Блок-схема для реализации теоремы 2



1. Последовательность , сходящаяся к , обладает линейной сходимостью если существует такая постоянная , что для всех . [↑](#footnote-ref-1)
2. Положительная последовательность обладает логарифмической сходимостью если существует такое α, что для всех . [↑](#footnote-ref-2)
3. Последовательность , сходящаяся к , обладает сверхлинейной сходимостью если существует такая положительная последовательность , что и для всех . [↑](#footnote-ref-3)
4. Модельная последовательность – это последовательность, предел которой может быть точно вычислен с помощью алгебраического процесса. [↑](#footnote-ref-4)
5. Антипредел – это математический термин, обозначающий обратный процесс к операции нахождения предела; является эквивалентом предела для расходящихся рядов. Вычисляется с помощью формулы для предела параметризованного ряда, применяемой к любым значениям параметров, в которых ряд не сходится. [↑](#footnote-ref-5)
6. Сингулярность - точка, в которой математическая функция стремится к бесконечности или имеет какие-либо иные нерегулярности поведения. [↑](#footnote-ref-6)
7. Осциллирующая последовательность — это последовательность, у которой значения оказываются близкими к какому-то числу через один номер, а через другой — к другому числу, при этом значения не стремятся к какому-то определенному числу. [↑](#footnote-ref-7)
8. Монотонная последовательность — это последовательность, элементы которой с увеличением номера не возрастают, или, наоборот, не убывают. [↑](#footnote-ref-8)