Введение

Алгоритм Форда-Сиди был предложен в 1987 году математиками Уильямом Фордом и Авраамом Сиди для решения задачи ускорения сходимости рядов. Данный метод был предложен в [1] как реализация рекурсивного алгоритма генерализации процесса экстраполяции Ричардсона, представленного в [2], позволяющая решить задачу экстраполяции более быстро и эффективно по сравнению с непосредственным решением системы линейных уравнений, определяющих экстраполяцию. Также, стоит отметить, что данный алгоритм схож с -алгоритмом и, более того, математически эквивалентен -алгоритму, но при этом является более вычислительно эффективным.

Экстраполяция Ричардсона

Рассмотрим основную идею процесса экстраполяции Ричардсона [6].

Пусть – последовательность, а - известные вспомогательные последовательности. Предположим, что существуют неизвестные константы , такие что:

Если система линейных уравнений (1) является невырожденной, то по правилу Крамера можно выразить как отношение двух определителей:

При этом данная формулировка позволяет наиболее обобщенно определить процесс экстраполяции, что позволяет использовать различные известные преобразования (трансформации) рядов вида для ускорения сходимости изначального ряда [3].

-алгоритм, предложенный независимо Шнайдером [6],

Основная проблема заключается в необходимости вычисления (2), решаемая различными рекурсивными алгоритмами. При этом, большая часть алгоритмов рассчитана на использование конкретного преобразования ряда, что приводит к необходимости использования различных алгоритмов для различных трансформаций рядов. Известны два рекурсивных алгоритма для вычисления :

1. -алгоритм, предложенный независимо К. Брезински [3], Т. Хейви [4] и К. Шнайдером [5]. Шнайдер и Хёви вывели его методом Гауссова исключения, а Брезински – с использованием тождества Сильвестра для определителей.
2. Алгоритм Форда-Сиди [1], который требует меньшего числа арифметических операций, чем -алгоритм. Форд и Сиди вывели свой алгоритм, используя тождество Сильвестра.

Форд и Сиди [1] отметили основное различие в рекурсии -алгоритма и алгоритма Форда-Сиди. Брезински и Редиво Дзалья [7] вывели оба алгоритма, используя операторы аннигиляции разностей, и установили связи между ними.

Далее покажем, что -алгоритм и алгоритм Форда-Сиди математически эквивалентны в следующем смысле: оба алгоритма вычисляют одни и те же величины, но разными способами, и рекуррентные соотношения -алгоритма могут быть выведены из соотношений алгоритма Форда-Сиди, и наоборот. Также предложим эффективную реализацию алгоритма Форда-Сиди, которая немного экономичнее оригинальной.

-алгоритм

Пусть - любая последовательность, которую нужно преобразовать, а - любые вспомогательные последовательности. Обозначим 1 как постоянную последовательность с для Предположим, что все знаменатели ненулевые.

-алгоритм определяется следующим образом. Для величины и определяются как

Рекуррентные соотношения (3) и (4) называются основным правилом и вспомогательным правилом -алгоритма соответственно. Брезински [3] доказал следующую теорему с использованием тождества детерминантов Сильвестра.

*Теорема 1*: для, и и представляются как

соответственно.

Если положить , то (3) и (4) становятся

соответственно.

Для заданных вычисление , требует операций. Количество операций для -алгоритма, как упомянуто в [3], составляет , в то время как с использованием (5) и (6) оно становится . Более точно, последнее равно .

Отметим, что Форд и Сиди [1] реализовали -алгоритм, переписав его в формах

где , ​а ​.

Реализация с использованием (7) и (8) требует точно такого же количества арифметических операций, как и с использованием (5) и (6). Однако можно избежать потери значимых цифр, используя (5) и (6).

Алгоритм Форда-Сиди

Алгоритм Форда-Сиди определяется следующим образом. Пусть - одна из последовательностей или . Величины определяются как

Форд и Сиди [1] доказали следующую теорему с использованием тождества детерминантов Сильвестра следующим образом.

Пусть

При этом (2) можно представить в виде:

Определим:

Следовательно, можно выразить (13) в виде

Заметим, что и с точки зрения определителя отличаются лишь последним столбцом, поэтому будет естественно вывести соотношение между ними. Для этого воспользуемся теоремой Сильвестра.

*Теорема 2 (теорема Сильвестра)*: пусть – матрица, - матрица, получаемая из путем удаления строки и столбца , - матрица, получаемая из путем удаления строк и и столбцов и, ,

Тогда

При этом, если – матрица , то .

Если применить данную теорему к размером при то получим:

Тогда можем получить следующее соотношение:

Пусть:

Тогда из (18) и (19) можно выразить

Из (16) и (20) можем видеть, что и могут быть вычислены рекурсивно. Следовательно, задача сводится к эффективному вычислению . При этом учитывая отсутствия конкретных данных о , учитывая, что согласно (4) и (5) можем сделать вывод, что:

Из этого можем выразить:

Из чего следует, что

То есть (15) и (23) позволяет рекурсивно решить поставленную задачу экстраполяции.

Рассмотрим непосредственно сам алгоритм:

1. Для вычислим:
2. Для вычислим  
   А также вычислим

.

И, соответственно, вычислим:

Процесс вычисления можно представить следующим образом:

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

Таблица 1. Представление массива

Замечание. Несмотря на кажущуюся необходимость в вычислении для вычисления каждого , на практике, это нужно только для , чтобы затем вычислить , и, соответственно, . Однако, как было отмечено в [6], на практике этих вычислений можно избежать, вычислив по формуле:

*Теорема 3*: для и представляются как

Следовательно, по теоремам 1 и 3, ​ можно вычислить как

Следуя реализации Форда и Сиди [1], вычисление требует вычитаний и арифметических операций. Псевдокод для алгоритма Форда-Сиди представлен на *Рисунке 1*, а пример его применения представлен на *Рисунке 2*.

**Вход**: последовательность (1), вспомогательные последовательности , максимальный порядок экстраполяции , количество используемых членов последовательности

**Выход**: (13)

Получить

#Инициализация (аналогично (9)):

**for** от до :

,

**for** :

#Рекурсивное вычисление (аналогично (20))

**for**  от до:

**for**  от до :

Вычислить #**(22)**

Для всех последовательностей :

Вычислить (аналогично (20)) #**Теорема 2**

**for** от до :

**for** от до :

Вычислить #**15**

**return**

*Рисунок 1*. Псевдокод для алгоритма Форда-Сиди.

**Вход**:

**Выход**:

*Рисунок 2*. Пример применения алгоритма Форда-Сиди.

Замечание. (1) Алгоритм Форда-Сиди требует ​ для вычисления ​. Однако для вычисления ​ алгоритм Форда-Сиди не требует ​. Причина кроется в следующем: пусть ​ - любая последовательность, такая что

Пусть

Тогда из (25) следует

Используя этот прием, когда заданы ​, можно определить все , с помощью алгоритма Форда-Сиди.

(2) Более того, ни значение ​, ни последовательность не требуются для вычисления ​, когда используется

что выводится из (10) и (25). Реализация этого факта и есть эффективный алгоритм Форда-Сиди, которая будет описана позже.

Вывод алгоритма Форда-Сиди из -алгоритма

Пусть – любая последовательность. Предположим, что преобразованная последовательность : лиопределены как

Предположим, что преобразования последовательностей =() определены как

*Теорема 4*: величины , определенные (28), удовлетворяют (9) и (10).

Доказательство. Из (26) и (28) следует, что . Используя математическую индукцию по , легко доказать, что . Таким образом, имеем

Из (28) следует

Из (27) и (30) получаем

Разделив обе части (13) на , получаем

следовательно, из (14) и (17) получаем

Отметим, что Брезински и Редиво Залья [7] вывели соотношения (28)-(30) из своих определений и .

Вывод -алгоритма из алгоритма Форда-Сиди

Предположим, что преобразования последовательностей удовлетворяют (9) и (10) для любой последовательности . Пусть преобразования последовательностей ​ определены как

*Теорема 5*: , определенные (32), удовлетворяют (26) и (27).

Доказательство. Из (10), имеем

Следовательно, по определению (32), получаем

Из (32) и (10) следует

Следовательно, по теоремам 4 и 5, считаем, что -алгоритм и алгоритм Форда-Сиди математически эквивалентны.

Эффективная реализация для алгоритма Форда-Сиди

Пусть определены (9) и (10). Из (25) и (10) следует, что в уравнении (2) представляется как

Вычисление с использованием эффективной реализации для алгоритма Форда-Сиди требует вычитаний и делений, всего арифметических операций. Хотя количество операций предложенного метода и алгоритма Форда-Сиди асимптотически равны, предложенный метод немного более экономичен, чем алгоритм Форда-Сиди.

Предположим, что ускорение сходимость обычной последовательности происходит с помощью подходящего метода, такого как -преобразование Левина в двойной точности, и что - оптимальное экстраполированное значение. Тогда обычно . (см., например, [8, 9].) Таким образом, предложенный метод на практике на 6−11% более экономичен, чем алгоритм Форда-Сиди. Псевдокод для эффективной реализации алгоритма Форда-Сиди представлен на *Рисунке 3*, а пример её применения представлен на *Рисунке 4*.

**Вход**: (аналогично канонической форме алгоритма)

**Выход**: (34)

Получить

#Инициализация (аналогично канонической форме алгоритма):

#Прямое вычисление (34)

**for**  от до:

**for**  от до :

Вычислить #**34**

Вычислить #**34**

Вычислить = числитель / знаменатель

**return**

*Рисунок 3*. Псевдокод для эффективной реализации алгоритма Форда-Сиди.

**Вход**:

**Выход**:

*Рисунок 4*. Пример применения эффективной реализации алгоритма Форда-Сиди.

Заключение

В данной работе были рассмотрены -алгоритм и алгоритм Форда-Сиди, выведена его рекуррентная формула, а также была доказана математическая эквивалентность алгоритма Форда-Сиди и -алгоритма. Также была предложена эффективная реализация алгоритма Форда-Сиди, которая немного экономичнее оригинальной.

Список литературы

1. An Algorithm for a Generalization of the Richardson Extrapolation Process // SIAM J. Numer. Anal. 24 (5) // W. F. Ford, A. Sidi – 1987. – P. 1212–1232.
2. Some Properties of a Generalization of the Richardson Extrapolation Process // Journal of Computational and Applied Mathematics // A. Sidi. – 1979. P. 327-346.
3. A general extrapolation algorithm // Numer. Math. 35 // C. Brezinski. – 1980. – P. 175–187.
4. Generalized neville type extrapolation schemes // BIT 19 // T. Håvie. – 1979. – P. 204–213.
5. Vereinfachte Rekursionen zur Richardson-Extrapolation in Spezialfällen // Numer. Math. 24 // C. Schneider. – 1975. – P. 177– 184.
6. The E-algorithm and the Ford–Sidi algorithm // Journal of computational and applied mathematics 122 (1) // N. Osada. – 2000. P. 223-230.
7. A general extrapolation procedure revisited // Adv. Comput. Math. 2 // C. Brezinski, M. Redivo Zaglia. – 1994. – P. 461-477.
8. Acceleration of linear and logarithmic sequence // SIAM J. Numer. Anal, 16 (2) // D.A. Smith, W.F. Ford. – 1979. - P. 223−240.
9. Extrapolation Methods, Theory and Practice // // C. Brezinski, M. Redivo Zaglia. - 1991.