$C$-finite and $C^2$-finite Sequences in SageMath

P. Nuspl

June 2022

RISC Report Series No. 22-06

ISSN: 2791-4267 (online)

Available at https://doi.org/10.35011/risc.22-06

This work is licensed under a CC BY 4.0 license.
C-finite and $C^2$-finite Sequences in SageMath*

Philipp Nuspl

philipp.nuspl@jku.at

1Johannes Kepler University Linz, Doctoral Program Computational Mathematics

Abstract

We present the SageMath package rec_sequences which provides methods to compute with sequences satisfying linear recurrences. The package can be used to show inequalities of C-finite sequences, i.e., sequences satisfying a linear recurrence relation with constant coefficients. Furthermore, it provides functionality to compute in the $C^2$-finite sequence ring, i.e., to compute closure properties of sequences satisfying a linear recurrence with $C^2$-finite coefficients.

1 Introduction

Sequences satisfying a linear recurrence with constant or polynomial coefficients arise in many combinatorial examples and as coefficient sequences of many special functions [5]. These sequences are called C-finite or D-finite, respectively. Recently, a generalization of these has been investigated: A sequence is called $C^2$-finite if it satisfies a linear recurrence with $C^2$-finite coefficients whose leading coefficient has no zero terms. As in the classical case of C-finite and D-finite sequences, these $C^2$-finite sequences form a difference ring. They satisfy additional closure properties such as taking subsequences or partial sums [3, 2].

The package rec_sequences provides an implementation of these closure properties for $C^2$-finite sequences. The computations for $C^2$-finite sequences are reduced to solving linear systems over the C-finite sequence ring. Arithmetic operations of C-finite sequences can then be efficiently performed using the ore_algebra package [4]. The main difference of $C^2$-finite sequences compared to the classically studied sequences is the presence of zero divisors in the coefficients of the recurrence and therefore in the linear systems. To compute with these zero divisors, the zero terms of these sequences have to be known. We implement algorithms presented in [6, 9] which can find these zeros in many cases. In general, however, it is not known whether one can decide if a C-finite sequence contains a zero term [8]. The implemented algorithms can, however, in many cases be used to show inequalities, non-negativity or positivity of C-finite sequences. To our knowledge our package provides the most powerful implementation for showing C-finite inequalities and the only one which provides methods to work with $C^2$-finite sequences. The only other packages known to us for proving inequalities are Mathematica and SageMath implementations of the general Gerhold-Kauers method. These implementations, however, do

*The research was funded by the Austrian Science Fund (FWF) under the grant W1214-N15, project DK15.
not provide any special procedures for $C$-finite sequences and are therefore much more limited regarding $C$-finite inequalities (see [7] and references therein).

The source code, more detailed instructions for the installation and extensive documentation can be obtained from GitHub\(^1\). For a standard SageMath installation, the package can be installed using

```
sage --pip install git+https://github.com/PhilippNuspl/rec_sequences.git
```

This will also automatically install the `ore_algebra` package. For proving inequalities, the optional package `QEPCAD` is needed [1]. It can be installed using `sage -i qepcad`.

## 2 $C$-finite sequences

After the installation of the package it can be loaded in any SageMath session. A $C$-finite sequence ring over a field of characteristic zero $K$ can be created by `CFiniteSequenceRing(K)`. A sequence in this ring $C$ can now be defined by a list of the coefficients of the recurrence and initial values:

\[
C([\gamma_0, \ldots, \gamma_r], [c_0, \ldots, c_{r-1}]) \leftrightarrow \begin{cases} 
\gamma_0 c(n) + \cdots + \gamma_r c(n + r) = 0, \\
c(0) = c_0, \ldots, c(r - 1) = c_{r-1}.
\end{cases}
\]

Alternatively, a symbolic expression in one variable or a list of initial terms can be used to define a $C$-finite sequence. In both cases guessing is used to find a recurrence.

```
sage: from rec_sequences.CFiniteSequenceRing import *
sage: C = CFiniteSequenceRing(QQ)
sage: fib = C([1,1,-1], [0,1], name="f") # Fibonacci numbers
sage: var("n")
sage: exp2 = C(2^n)
sage: alt = C(10*[1,-1])
sage: alt
C-finite sequence a(n): (1)*a(n) + (1)*a(n+1) = 0 and a(0)=1
```

Terms of a $C$-finite sequence can be obtained in the same way that elements of lists in Python are obtained.

```
sage: exp2[3], fib[:10]
(8, [0, 1, 1, 2, 3, 5, 8, 13, 21, 34])
```

Closure properties of $C$-finite sequences are computed using the `ore_algebra` package [4]. These include difference ring operations (using `+`, `*` and `shift`), partial sums (using `sum`), Cauchy product (using `cauchy`), interlacing (using `interlace`) and subsequences (using `subsequence`). The equality of two $C$-finite sequences is proven by checking enough initial values.

```
sage: fib.sum()==fib.shift(2)-1
True
```

Furthermore, one can obtain the recurrence coefficients and the initial values of a $C$-finite sequence (using `coefficients` and `initial_values`, respectively) or compute the closed form:

```
sage: (fib^2-fib.shift()^2).closed_form() # Cassini identity
(-(-1)^n)
```

Several algorithms to show inequalities of sequences are implemented. These include methods from [6, 9].

\(^1\)https://github.com/PhilippNuspl/rec_sequences
More information on any of the methods can be obtained using ?, e.g. fib.interlace?.

3 C²-finite sequences

Analogous to C-finite sequences, C²-finite sequences can again be defined by the coefficients of the recurrence and initial values. It is assumed, but not checked, that the leading coefficient does not contain any zero terms. In many cases, the method has_no_zeros can be used to verify whether a sequence has a zero.

If a sequence \(c(n)\) is C-finite, then the sparse subsequence \(c(n^2)\) is C²-finite and we can compute the C-finite coefficients of the recurrence [2]:

Other closure properties can again be computed using the same syntax as for C-finite sequences. For instance, we can compute a recurrence for \(\sum_{k=0}^{\lfloor n/3 \rfloor} f((2k+1)^2)\) [2, Example 5.1]:

A different recurrence with much shorter coefficients can be attained by computing the C²-finite subsequence \(f(4n^2 + 4n + 1)\) directly instead of computing a subsequence \(f((2n+1)^2)\) of the C²-finite sequence \(f(n^2)\):
sage: h2 = fib.sparse_subsequence(C2, 4, 4, 1).sum().multiple(3)
sage: h2.order(), h2.degree(), h2.leading_coefficient().order()
(9, 12, 6)

Acknowledgments

The author likes to thank Antonio Jiménez Pastor for providing generous help especially concerning details of implementations and Veronika Pillwein for many helpful discussions.

References


