A Comparison of Algorithms for Proving Positivity of Linearly Recurrent Sequences

Philipp Nuspl[1(✉)] and Veronika Pillwein[2]

[1] Doctoral Program Computational Mathematics, Johannes Kepler University, Linz, Austria
philipp.nuspl@jku.at
[2] Research Institute for Symbolic Computation, Johannes Kepler University, Linz, Austria
veronika.pillwein@risc.jku.at

Abstract. Deciding positivity for recursively defined sequences based on only the recursive description as input is usually a non-trivial task. Even in the case of C-finite sequences, i.e., sequences satisfying a linear recurrence with constant coefficients, this is only known to be decidable for orders up to five. In this paper, we discuss several methods for proving positivity of C-finite sequences and compare their effectiveness on input from the Online Encyclopedia of Integer Sequences (OEIS).

Keywords: Difference equations · Inequalities · Holonomic sequences

1 Introduction

A sequence is called D-finite (or P-recursive or holonomic), if it satisfies a linear recurrence with polynomial coefficients. These sequences appear in many applications, e.g., in combinatorics or as coefficient sequences of special functions [7,32]. They are interesting from the symbolic computation point of view, as they can be represented by a finite amount of data – the recurrence coefficients and sufficiently many initial values. Several closure properties hold for holonomic sequences and there exist summation algorithms that work with this representation for input and output. These methods are used to automatically prove and derive identities for holonomic sequences. When it comes to automatic proving of inequalities on holonomic sequences, there are not many algorithms available. Gerhold and Kauers [10] introduced a method in 2005 that can be used for sequences satisfying (a system of) recurrences including in particular holonomic sequences. This method (together with variations of it) has been applied successfully on several examples [17,29,31]. Still, a priori it is not known in general whether the procedure terminates [22].

The research was funded by the Austrian Science Fund (FWF) under the grant W1214-N15, project DK15.

In this paper, we restrict our study to C-finite sequences, i.e., holonomic sequences with constant coefficients, and the problem of proving positivity. This is known to be decidable for integer linear recurrences of order 2 [12], order 3 [23], order at most 5 and is related to difficult number theoretic problems for higher order [28]. We give an overview on some methods which can be used to prove the positivity of C-finite sequences, including the Gerhold–Kauers method and the most used variation (Algorithms 1 and 2 below). Other methods are based on theoretical results that, as far as we know, have not yet been implemented and tested on practical input on a larger scale. For testing the effectiveness of these different algorithms, we use input from the Online Encyclopedia of Integer Sequences (OEIS) [27] that are likely candidates for positive sequences. Our implementations are done both in SageMath and Mathematica and the source files as well as testing data are made available online (see links in Sect. 4).

2 Preliminaries

We introduce some notations and definitions that will be used throughout the paper. We always assume that $\mathbb{Q} \subseteq \mathbb{K} \subsetneq \mathbb{R}$ is some number field. We denote the field of algebraic numbers by $\overline{\mathbb{Q}}$ and the field of real algebraic numbers by $\mathbb{A} := \overline{\mathbb{Q}} \cap \mathbb{R}$. We denote the $\mathbb{K}$-vector space of sequences by $\mathbb{K}^{\mathbb{N}}$ and let σ denote the shift operator $\sigma((c(n))_{n\in\mathbb{N}}) := (c(n+1))_{n\in\mathbb{N}}$.

2.1 Linear Recurrence Sequences

We denote the Ore algebra of shift operators by $\mathbb{K}[x]\langle\sigma\rangle$. Let $\mathcal{A} = \sum_{i=0}^{r} p_i(x)\sigma^i \in \mathbb{K}[x]\langle\sigma\rangle$. If $p_r \neq 0$, then r is called the *order* of $\mathcal{A}$ and $\max_{i=0,\dots,r} \deg(p_i)$ is called the *degree* of $\mathcal{A}$. The operator $\mathcal{A}$ acts on a sequence $c \in \mathbb{K}^{\mathbb{N}}$ in the natural way as

$$\mathcal{A}c = (p_0(n)c(n) + \cdots + p_r(n)c(n+r))_{n\in\mathbb{N}}.$$

A sequence $c \in \mathbb{K}^{\mathbb{N}}$ is called *D-finite* (or P-recursive or holonomic) if there is a non-zero operator $\mathcal{A} \in \mathbb{K}[x]\langle\sigma\rangle$ with $\mathcal{A}c = 0$, i.e., the sequence satisfies a linear recurrence with polynomial coefficients. We call $\mathcal{A}$ an *annihilating operator* of c. It is well known that D-finite sequences form a computable difference ring [21]. The minimal possible order r of an annihilating operator is also called the *order* of the sequence c. The *degree* of c is then just defined as the degree of this operator.

A D-finite sequence c is called C-finite if it satisfies a linear recurrence with constant coefficients, i.e., if there are $\gamma_0, \dots, \gamma_r \in \mathbb{K}$ with $\gamma_r \neq 0$ such that

$$\gamma_0 c(n) + \cdots + \gamma_r c(n+r) = 0, \qquad \text{for all } n \in \mathbb{N}. \tag{1}$$

Again, the order of c as a C-finite sequence is the minimal r (note that the order of c considered as a C-finite sequence can be different from the order considered as a D-finite sequence, cf. Lemma 3). The set of C-finite sequences is again a computable difference ring. Every such sequence can be uniquely described by the coefficients of the recurrence $\gamma_0, \dots, \gamma_r$ and sufficiently many initial values $c(0), \dots, c(r-1)$.

2.2 Characteristic Polynomial

For an operator $\mathcal{A} = \sum_{i=0}^{r} p_i(x)\sigma^i \in \mathbb{K}[x]\langle\sigma\rangle$ the *characteristic polynomial* is defined as

$$\chi(\mathcal{A}) \coloneqq \operatorname{lc}_x\left(\sum_{i=0}^{r} p_i(x)y^i\right) \in \mathbb{K}[y].$$

The roots of $\chi(\mathcal{A})$ are called *eigenvalues* and usually govern the asymptotic behavior of sequences which are annihilated by $\mathcal{A}$ [22].

We can extend the notion of the characteristic polynomial to the left-Noetherian ring $\mathbb{K}(x)\langle\sigma\rangle$. For a univariate polynomial $p \in \mathbb{K}[x]$ we denote by $\operatorname{coeff}(p, i) \in \mathbb{K}$ the coefficient of x^i in p. For a rational function $\frac{p(x)}{q(x)}$ with coprime $p, q \in \mathbb{K}[x]$ we define the *degree* as $\deg(p/q) = \deg(p) - \deg(q)$ and call

$$\operatorname{lc}(p/q) \coloneqq \operatorname{coeff}(p/q, \deg(p/q)) \coloneqq \operatorname{lc}(p)/\operatorname{lc}(q)$$

the leading coefficient of p/q. Now, for an operator $\mathcal{A} = \sum_{i=0}^{r} \frac{p_i(x)}{q_i(x)}\sigma^i \in \mathbb{K}(x)\langle\sigma\rangle$ with $\deg(\mathcal{A}) \coloneqq \max_{i=0,\dots,r} \deg(p_i/q_i)$ we define the characteristic polynomial as

$$\chi(\mathcal{A}) \coloneqq \sum_{\substack{i=0 \\ \deg(p_i/q_i)=\deg(\mathcal{A})}}^{r} \operatorname{lc}(p_i/q_i)y^i \in \mathbb{K}[y].$$

Next, in Lemma 1 and Lemma 2, we state some basic properties of the characteristic polynomial. Since we could not find references for those, we add the proofs for sake of completeness.

Lemma 1. *Let $\mathcal{A}, \mathcal{B} \in \mathbb{K}(x)\langle\sigma\rangle$. Then $\chi(\mathcal{A}\mathcal{B}) = \chi(\mathcal{A})\chi(\mathcal{B})$.*

Proof. Let $\mathcal{A} \coloneqq \sum_{i=0}^{r} p_i(x)\sigma^i \in \mathbb{K}(x)\langle\sigma\rangle$ and $\mathcal{B} \coloneqq \sum_{j=0}^{s} q_j(x)\sigma^j \in \mathbb{K}(x)\langle\sigma\rangle$ and $d_{\mathcal{A}} \coloneqq \max_{i=0,\dots,r} \deg p_i, d_{\mathcal{B}} \coloneqq \max_{j=0,\dots,s} \deg q_j \in \mathbb{Z}$ their respective degrees. We show that $\mathcal{A}\mathcal{B}$ has degree $d_{\mathcal{A}} + d_{\mathcal{B}}$. By the definition of multiplication in $\mathbb{K}(x)\langle\sigma\rangle$ and the properties of the degree of a rational function, the degree of $\mathcal{A}\mathcal{B}$ is certainly bounded by $d_{\mathcal{A}} + d_{\mathcal{B}}$. Let i', j' be maximal such that $\deg p_{i'} = d_{\mathcal{A}}$ and $\deg q_{j'} = d_{\mathcal{B}}$. We show that the coefficient of $\sigma^{i'+j'}$ of $\mathcal{A}\mathcal{B}$ has degree $d_{\mathcal{A}} + d_{\mathcal{B}}$. This coefficient is given by $\sum_{l=0}^{i'+j'} p_l(x)q_{i'+j'-l}(x+l)$. Because of the choice of i', j' we have

$$\deg(p_l(x)q_{i'+j'-l}(x)) = \deg(p_l(x)) + \deg(q_{i'+j'-l}(x+l)) < d_{\mathcal{A}} + d_{\mathcal{B}}$$

for all $l \neq i'$. For $l = i'$, we have $\deg(p_l(x)q_{i'+j'-l}(x)) = d_{\mathcal{A}} + d_{\mathcal{B}}$, so by the properties of the degree we have

$$\begin{aligned}\deg\left(\sum_{l=0}^{i'+j'} p_l(x)q_{i'+j'-l}(x+l)\right) &= \max_{l=0,\dots,i'+j'} \left(\deg\left(p_l(x)\right) + \deg\left(q_{i'+j'-l}(x+l)\right)\right) \\ &= d_{\mathcal{A}} + d_{\mathcal{B}}.\end{aligned}$$

Next, we show that all coefficients of $\chi(\mathcal{A})\chi(\mathcal{B})$ and $\chi(\mathcal{AB})$ agree. Let $i \in \{0, \dots, r+s\}$. Then,

$$\operatorname{coeff}(\chi(\mathcal{A}), i) = \operatorname{coeff}(p_i(x), d_{\mathcal{A}}), \quad \operatorname{coeff}(\chi(\mathcal{B}), i) = \operatorname{coeff}(q_i(x), d_{\mathcal{B}})$$

and therefore

$$\operatorname{coeff}(\chi(\mathcal{A})\chi(\mathcal{B}), i) = \sum_{j=0}^{i} \operatorname{coeff}(p_j(x), d_{\mathcal{A}}) \operatorname{coeff}(q_{i-j}(x), d_{\mathcal{B}}).$$

In the first part of the proof, we have shown that $\mathcal{AB}$ has degree $d_{\mathcal{A}} + d_{\mathcal{B}}$. Therefore,

$$\begin{aligned}\operatorname{coeff}(\chi(\mathcal{AB}), i) &= \operatorname{coeff}\left(\sum_{j=0}^{i} p_j(x) q_{i-j}(x+j), d_{\mathcal{A}} + d_{\mathcal{B}}\right) \\ &= \sum_{j=0}^{i} \operatorname{coeff}(p_j(x) q_{i-j}(x+j), d_{\mathcal{A}} + d_{\mathcal{B}}) \\ &= \sum_{j=0}^{i} \operatorname{coeff}(p_j(x), d_{\mathcal{A}}) \operatorname{coeff}(q_{i-j}(x+j), d_{\mathcal{B}}) \\ &= \sum_{j=0}^{i} \operatorname{coeff}(p_j(x), d_{\mathcal{A}}) \operatorname{coeff}(q_{i-j}(x), d_{\mathcal{B}}).\end{aligned}$$

□

Suppose $\mathcal{A}$ is an annihilator of a and $\mathcal{B}$ an annihilator of b. Then, the *least common left multiple* $\operatorname{lclm}(\mathcal{A}, \mathcal{B})$ is an annihilator of $a + b$ [19].

Lemma 2. *Let* $\mathcal{A}, \mathcal{B} \in \mathbb{K}[x]\langle\sigma\rangle$. *Then*

$$\chi(\mathcal{A}) \mid \chi(\operatorname{lclm}(\mathcal{A}, \mathcal{B})) \textit{ and } \chi(\mathcal{B}) \mid \chi(\operatorname{lclm}(\mathcal{A}, \mathcal{B})).$$

In particular, we have

$$\operatorname{lcm}(\chi(\mathcal{A}), \chi(\mathcal{B})) \mid \chi(\operatorname{lclm}(\mathcal{A}, \mathcal{B})).$$

Proof. Let $\mathcal{C} \in \mathbb{K}(x)\langle\sigma\rangle$ be such that $\mathcal{CA} = \operatorname{lclm}(\mathcal{A}, \mathcal{B})$. Then, with Lemma 1 we have

$$\chi(\operatorname{lclm}(\mathcal{A}, \mathcal{B})) = \chi(\mathcal{CA}) = \chi(\mathcal{C})\chi(\mathcal{A}).$$

□

Example 1. In Lemma 2, divisibility cannot be replaced with equality. Consider $\mathcal{A} := 1 + \sigma$ and $\mathcal{B} := x + (x+1)\sigma$. Then,

$$\chi(\mathcal{A}) = \chi(\mathcal{B}) = 1 + y,$$

but

$$\chi(\operatorname{lclm}(\mathcal{A}, \mathcal{B})) = \chi(x + (2x+2)\sigma + (x+2)\sigma^2) = 1 + 2y + y^2.$$

An operator $\mathcal{A} = \sum_{i=0}^{r} p_i \sigma^i \in \mathbb{K}[x]\langle\sigma\rangle$ is called *balanced* if

$$\deg p_0 = \deg p_r = \max_{i=0,\dots,r} \deg p_i.$$

Equivalently, $\mathcal{A}$ is balanced if and only if the degree of $\chi(\mathcal{A}) \in \mathbb{K}[y]$ equals the order of $\mathcal{A}$ and the trailing coefficient of $\chi(\mathcal{A})$ is non-zero, i.e., $y \nmid \chi(\mathcal{A})$.

2.3 Positivity

Suppose we are given a C-finite sequence c in terms of a recurrence and sufficiently many initial values. Our goal is to prove $c(n) > 0$ for all $n \in \mathbb{N}$ (i.e., show that c is positive) or to find an index $n_0 \in \mathbb{N}$ such that $c(n_0) \leq 0$. The very same methods can always be applied to show non-negativity instead of strict positivity of a sequence.

If b, c are C-finite sequences, then the inequality $b > c$ (or $b \geq c$) can easily be reduced to the positivity problem. The sequence $b-c$ is again C-finite. Hence, proving the equivalent positivity problem $b - c > 0$ (or $b - c \geq 0$) shows the original inequality.

Suppose c is C-finite satisfying the recurrence (1). Let $k \in \mathbb{N}$ be minimal such that $\gamma_k \neq 0$. Now, define $\tilde{c} := \sigma^k c$. Then, $\tilde{c}$ is again C-finite satisfying the recurrence

$$\gamma_k \tilde{c}(n) + \cdots + \gamma_r \tilde{c}(n + r - k) = 0, \qquad \text{for all } n \in \mathbb{N}.$$

The sequence c is positive if and only if the sequence $\tilde{c}$ and the initial values $c(0), \dots, c(k-1)$ are positive. Therefore, we can (and will) always assume that a C-finite sequence c is given by a recurrence with coefficients $\gamma_0, \dots, \gamma_r$ with $\gamma_0, \gamma_r \neq 0$. Such a sequence c can then always be written as a polynomial-linear combination of exponential sequences. One can compute polynomials $p_1, \dots, p_m \in \overline{\mathbb{Q}}[x]$ and pairwise distinct non-zero constants $\lambda_1, \dots, \lambda_m \in \overline{\mathbb{Q}}$ such that

$$c(n) = \sum_{i=1}^{m} p_i(n) \lambda_i^n, \qquad \text{for all } n \in \mathbb{N}. \tag{2}$$

These λ_i are called the *eigenvalues* of c and they are the roots of the characteristic polynomial $\sum_{i=0}^{r} \gamma_i y^i \in \mathbb{K}[y]$ of the minimal order recurrence of c. More precisely, if λ_i is a root of multiplicity d_i, then $\deg(p_i) = d_i - 1$. Hence, $r = \sum_{i=1}^{m} d_i$ [21].

Two sequences b, c which are non-zero from some term on are called asymptotically equivalent if $\lim_{n\to\infty} \frac{b(n)}{c(n)} = 1$. In this case, we write $b \sim c$. The asymptotic behavior of c is governed by the k eigenvalues of maximal modulus, we call them the *dominant eigenvalues*. We assume $|\lambda_1| = \cdots = |\lambda_k| > |\lambda_{k+1}| \geq \cdots \geq |\lambda_m|$. Let $d := \max_{i=1,\dots,k} \deg p_i$. Then, $c(n) \sim n^d \sum_{i=1}^{k} \operatorname{coeff}(p_i, d)\, \lambda_i^n$ [21].

In the special case that we have a unique dominant eigenvalue (i.e., $k = 1$) we have $c(n) \sim \gamma n^d \lambda_1^n$ for some γ [21]. Hence, c can only be a positive sequence if $\gamma, \lambda_1 \in \mathbb{A}$ and $\gamma > 0, \lambda_1 > 0$. Then, c is positive if and only if $c(n)/\lambda_1^n$ is positive. Therefore, it is sufficient to show positivity of a sequence

$$p(n) + \sum_{i=1}^{s} \left(o_i(n)\xi_i^n + \overline{o_i}(n)\overline{\xi_i}^n \right) + \sum_{i=1}^{l} q_i(n)\rho_i^n \tag{3}$$

with $p \in \mathbb{A}[x], o_1, \ldots, o_s \in \overline{\mathbb{Q}}[x], q_1, \ldots, q_l \in \mathbb{A}[x]$ and constants $\xi_1, \ldots, \xi_s \in \overline{\mathbb{Q}}, \rho_1, \ldots, \rho_l \in \mathbb{A}$ where the leading coefficient of p is positive [28].

3 Algorithms

In this section we give an overview over some methods which can be used to prove positivity of a C-finite sequence. Algorithms 1 and 2 introduced below in Sects. 3.1, 3.2 can be applied to D-finite sequences. As such they can be used to prove positivity of C-finite sequences. However, sometimes C-finite sequences satisfy a D-finite recurrence of lower order, which is better suited as input for these methods. In Sect. 3.3, we discuss when such a D-finite recurrence exists. A method based on the combination of Algorithms 1 and 2 as well as on the closed form of a C-finite sequence is introduced in Sect. 3.5. The methods described in Sects. 3.4 and 3.6 also make use of the closed form of C-finite sequences. They are based on known results, but we believe that they had not been implemented so far.

3.1 Algorithm 1

In 2008 [10], a method based on cylindrical algebraic decomposition [1,3,5,6] (CAD) was introduced which can be used to show positivity of sequences that can be defined recursively along some discrete parameter. This procedure, however, is not guaranteed to terminate. For D-finite sequences of small order conditions which guarantee the termination of the algorithm were found [22,30].

We give a short description of Algorithm 1 from [22]. For a D-finite sequence c of order r, the $\mathbb{Q}(x)$-vector space which is generated by the shifts of c is finitely generated [21]. In fact, it is generated by $c, \ldots, \sigma^{r-1}c$, i.e.,

$$\langle \sigma^i c \mid i \in \mathbb{N} \rangle_{\mathbb{Q}(x)} = \langle c, \ldots, \sigma^{r-1} c \rangle_{\mathbb{Q}(x)}.$$

Hence, for all $\rho \in \mathbb{N}$ there are rational functions $q_{\rho,0}(x), \ldots, q_{\rho,r-1}(x) \in \mathbb{K}(x)$ with $c(n+\rho) = \sum_{i=0}^{r-1} q_{\rho,i}(n)c(n+i)$ for all $n \in \mathbb{N}$. The idea now is to check with CAD whether $c(n), \ldots, c(n+r-1) > 0$ implies $c(n+r) > 0$ where $c(n+r)$ can be written in terms of the $c(n), \ldots, c(n+r-1)$. If this is true, then by induction it would be sufficient to check finitely many initial values to deduce positivity of the entire sequence. If, however, this cannot be shown, then we can add $c(n+r) > 0$ to the hypothesis and show $c(n+r+1) > 0$. This process is iterated. In the iteration step $\rho \geq r$ we try to show positivity of the formula

$$\Phi(\rho, c) := \forall y_0, \ldots, y_{r-1}, x \in \mathbb{R}\colon \left(x \geq 0 \wedge \bigwedge_{j=0}^{\rho-1} \sum_{i=0}^{r-1} q_{j,i}(x) y_i > 0 \right) \implies \sum_{i=0}^{r-1} q_{\rho,i}(x) y_i > 0.$$

Formula $\Phi(\rho, c)$ is a generalized induction formula over the reals. It is certainly sufficient to prove the initial induction step and has the advantage of being a valid input for CAD. Here, we give a slightly adjusted version which searches for an index n_0 such that the sequence $\sigma^{n_0} c$ is positive, i.e., it checks whether the sequence is *eventually* positive (hence, we denote the algorithm by Algorithm 1e). If such an n_0 can be found by the algorithm, then it is sufficient to check the initial values $c(0), \ldots, c(n_0 - 1)$ of the sequence to prove positivity of c.

Algorithm 1e. Adjusted version of Algorithm 1 from [22]

```
Input  : $D$-finite sequence $c$ of order $r$
output: $n_0$ such that $\sigma^{n_0} c$ is positive
$n, n_0 \leftarrow 0$
$d \leftarrow c$
while $n < r$ or $\neg\Phi(n, d)$ do
    if $d(n) > 0$ then
        $n \leftarrow n + 1$
    else
        $n_0 \leftarrow n_0 + n + 1$
        $d \leftarrow \sigma^{n+1} d$
        $n \leftarrow 0$
return $n_0$
```

Clearly, Algorithm 1e is not guaranteed to terminate. E.g., if the input sequence c is negative, then the algorithm never terminates. Suppose the sequence c is eventually positive, i.e., there exists an $n_0 \in \mathbb{N}$ such that $\sigma^{n_0} c$ is positive. Since $\chi(c) = \chi(\sigma^{n_0} c)$, the same termination conditions for Algorithm 1 in [22] now also apply to Algorithm 1e.

Example 2. The sequence A001903 is C-finite of order 3 satisfying

$$c(n) - c(n+1) + c(n+2) - c(n+3) = 0$$

with initial values $c(0) = 1, c(1) = 7, c(2) = 9$. Algorithm 1e terminates for this sequence for $n = 4$ showing that c is positive.

3.2 Algorithm 2

Algorithm 2 in [22] again uses CAD to prove positivity of a D-finite sequence. The idea is to check whether there is a $\mu > 0$ such that $c(n+1) \geq \mu c(n)$ for all $n \in \mathbb{N}$. By induction, if there is a $\mu > 0$ such that $c(n+1) \geq \mu c(n), \dots, c(n+r-1) \geq \mu c(n+r-2)$ implies $c(n+r) \geq \mu c(n+r-1)$, then it is again sufficient to check finitely many initial values to prove positivity of c. Hence, the important step in the algorithm is to use CAD to check whether there exists a $\mu > 0$ such that the formula

$$\begin{aligned}\Psi(\xi, \mu, c) \coloneqq \forall y_0, \dots, y_{r-1} \in \mathbb{R}\, \forall x \in \mathbb{R}_{\geq \xi}\colon \left(y_0 > 0 \wedge \bigwedge_{i=0}^{r-2} y_{i+1} \geq \mu y_i \right) \\ \implies \sum_{i=0}^{r-1} q_i(x) y_i \geq \mu y_{r-1}\end{aligned}$$

is valid where $q_i \in \mathbb{K}(x)$ are such that $c(n+r) = \sum_{i=0}^{r-1} q_i(n) c(n+i)$ for all $n \in \mathbb{N}$.

Again, we give a slightly adjusted version which searches for an index n_0 such that the sequence $\sigma^{n_0} c$ is positive. If the input sequence c is eventually positive, then the same termination conditions as for Algorithm 2 in [22] apply in this adjusted version.

Algorithm 2e. Adjusted version of Algorithm 2 from [22]

```
Input  : $D$-finite sequence $c$ of order $r$
output: $n_0$ such that $\sigma^{n_0} c$ is positive
$n, n_0 \leftarrow 0$
$d \leftarrow c$
$\Psi(\xi, \mu) \leftarrow$ quantifier free formula equivalent to $\Psi(\xi, \mu, d)$
for $n = 0, 1, \dots$ do
    if $d(n) \leq 0$ then
        $n_0 \leftarrow n_0 + n + 1$
        $d \leftarrow \sigma^{n+1} d$
        $\Psi(\xi, \mu) \leftarrow$ quantifier free formula equivalent to $\Psi(\xi, \mu, d)$
        $n \leftarrow 0$
    else if $\exists \mu > 0\colon \bigwedge_{i=0}^{r-2} d(n+i+1) \geq \mu d(n+i) \wedge \Psi(n, \mu)$ then
        return $n_0$
```

Example 3. The sequence A005682 is C-finite of order 6 satisfying

$$c(n) + c(n+2) - 2c(n+5) + c(n+6) = 0$$

with initial values $c = \langle 1, 2, 4, 8, 15, 28, \dots \rangle$. Algorithm 2e terminates for this sequence at $n = 0$ showing that c is positive. Algorithm 1e cannot show positivity of c in 60 s.

3.3 *D*-finite Reduction

Clearly, every C-finite sequence is also D-finite. Sometimes, C-finite sequences satisfy shorter D-finite recurrences. In these cases, it can be helpful to use this shorter D-finite recurrence as the next example shows.

Example 4. Let c be the sequence defined by $c(n) = n^2 + 1$ for all $n \in \mathbb{N}$ (A002522). If c is considered as a C-finite sequence of order 3, then neither Algorithm 1e nor Algorithm 2e terminate in 60 s. If c is, however, considered as a D-finite sequence of order 1 and degree 2, then both algorithms terminate and show that c is indeed positive.

The next lemma shows that we can find a shorter D-finite recurrence of a C-finite sequence c if and only if c has eigenvalues of higher multiplicities or equivalently the characteristic polynomial of c is not squarefree.

Lemma 3. *Let c be a C-finite sequence of order r with $y \nmid \chi(c)$. Then, c is D-finite of order $m < r$ if and only if $\chi(c)$ is not squarefree.*

Proof. Suppose c is given as in (2).

$\Longleftarrow$: The sequences $p_i(n)\lambda_i^n$ are D-finite of order 1 and degree d_i over $\overline{\mathbb{Q}}$. Hence, by the bounds for closure properties of D-finite sequences, $c(n)$ is D-finite of order at most m over $\overline{\mathbb{Q}}$ [21]. [9, Lemma 2] shows that the sequence is then also D-finite over $\mathbb{K}$ with the same order and degree. In particular, if $\chi(c)$ is not squarefree, then $r = \sum_{i=1}^m d_i > m$.

$\Longrightarrow$: Suppose c satisfies a D-finite recurrence of order $m < r$ and degree d

$$\sum_{i=0}^{m} p_i(n)c(n+i) = 0 \quad \text{for all } n \in \mathbb{N} \tag{4}$$

with $p_i(n) = \sum_{k=0}^d p_{i,k} n^k$ where not all $p_{i,k}$ are zero. Furthermore, suppose that c is C-finite of order r with pairwise distinct eigenvalues $\lambda_1, \dots, \lambda_r \in \overline{\mathbb{Q}}$, i.e., $c(n)$ can be written as $c(n) = \sum_{j=1}^r \gamma_j \lambda_j^n$ for some $\gamma_j \in \overline{\mathbb{Q}}$. Using this closed form in (4) yields

$$\sum_{k=0}^{d} \left(\sum_{i=0}^{m} \sum_{j=1}^{r} p_{i,k} \gamma_j \lambda_j^{n+i} \right) n^k = 0. \tag{5}$$

Let $\gamma_{k,j} := \sum_{i=0}^m p_{i,k}\gamma_j\lambda_j^i$, then (5) is equivalent to $\sum_{k=0}^d \left(\sum_{j=1}^r \gamma_{k,j}\lambda_j^n\right) n^k = 0$. For $n = 0, \dots, r(d+1)-1$ we get a homogeneous linear system for the $\gamma_{k,j}$. The corresponding matrix is regular [24, Theorem 2.2.1],[13, Proposition 2.11], so $\gamma_{k,j} = 0$ for all k, j. Let k be such that $p_{i,k} \neq 0$ for some i. Then,

$$0 = \sum_{j=1}^{r} \lambda_j^n \sum_{i=0}^{m} p_{i,k}\gamma_j\lambda_j^i = \sum_{i=0}^{m}\sum_{j=1}^{r} p_{i,k}\gamma_j\lambda_j^{n+i} = \sum_{i=0}^{m} p_{i,k} c(n+i).$$

Hence, c satisfies a C-finite recurrence of order $m < r$, a contradiction to c being C-finite of order r. □

The proof of Lemma 3 shows that precisely the polynomial factors can be reduced in the D-finite recurrence, i.e., the m in the statement of Lemma 3 is the number of distinct eigenvalues of the sequence, which is also denoted by m in Eq. (2). The degree of the D-finite recurrence can be bounded by

$$(m(m+1)-m) \max_{i=1,\dots,m} d_i = m^2 \max_{i=1,\dots,m} d_i \leq r^3$$

using [18, Theorem 2].

In practice, we can easily check whether $\chi(c)$ is squarefree by checking whether $\chi(c)$ and its derivative are coprime. The shorter D-finite recurrence can then be either found by guessing or by computing it explicitly from the closed form of c.

3.4 Classical Algorithm for Sequences with Unique Dominant Eigenvalue

If a C-finite sequence has a unique dominant eigenvalue, checking positivity of the sequence is known to be decidable [28]. In this section, we give a full description of such an algorithm based on that result.

As discussed in Sect. 2.3 we can assume that a C-finite sequence c is given in its closed form representation, i.e., as

$$c(n) = p(n) + r(n), \tag{6}$$

where $p \in \mathbb{A}[x]$ with $\operatorname{lc}(p) > 0$ and $r(n) = \sum_{i=1}^{m} p_i(n)\lambda_i^n$ with $p_i \in \overline{\mathbb{Q}}[x], \lambda_i \in \overline{\mathbb{Q}}$ and $1 > |\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_m|$. The idea is now to compute an $\varepsilon \in (0,1)$ and $n_0, n_1 \in \mathbb{N}$ such that $|r(n)| < (1-\varepsilon)^n$ for $n \geq n_0$ and $p(n) \geq (1-\varepsilon)^n$ for $n \geq n_1$. Then, clearly $c(n)$ is positive from $\max(n_0, n_1)$ on. The initial values can be checked separately again.

Algorithm C. Positivity for sequences with dominant eigenvalues [28]

```
Input : C-finite sequence $c$ of the form (6)
output: true if $c(n) > 0$ for all $n \in \mathbb{N}$ and false otherwise
$\varepsilon \leftarrow \frac{1-|\lambda_1|}{2}$
compute $n_0$ such that $|r(n)| < (1-\varepsilon)^n$ for all $n \geq n_0$
compute $n_1$ such that $p(n) \geq (1-\varepsilon)^n$ for all $n \geq n_1$
if $c(n) > 0$ for $n = 0, \dots, \max(n_0, n_1)$ then
    return true
else
    return false
```

For a polynomial $p_i(x) = \sum_{j=0}^{d_i} \gamma_{i,j} x^j \in \mathbb{A}$ of degree d_i we can easily compute a constant $c_i \in \mathbb{A}$ such that $|p_i(n)| \leq c_i n^{d_i}$ for all $n \geq 1$. For example, we can choose $c_i := \sum_{i=0}^{d_i} |\gamma_{i,j}|$. Let $c := \sum_{i=1}^{m} c_i$ and $d := \max(d_1, \dots, d_m)$, i.e., the maximal multiplicity of the eigenvalues $\lambda_1, \dots, \lambda_m$. Furthermore, let $\varepsilon := \frac{1-|\lambda_1|}{2}$. Then, $1-\varepsilon = |\lambda_1| + \varepsilon$.

First, we show how n_0 can be found such that $|r(n)| < (1-\varepsilon)^n$ for $n \geq n_0$. Let $\mu := \frac{|\lambda_1|+\varepsilon}{|\lambda_1|}$. If $d = 0$, then

$$|r(n)| \leq c|\lambda_1|^n < (1-\varepsilon)^n \iff \tfrac{\log(c)}{\log(\mu)} < n.$$

Hence, we can choose $n_0 := \lceil \frac{\log(c)}{\log(\mu)} \rceil$ in this case. If $d > 0$, then

$$|r(n)| \leq c\, n^d |\lambda_1|^n < (1-\varepsilon)^n \iff \log(c^{1/d}) < \tfrac{n}{d} \log(\mu) - \log(n).$$

The derivative of the right-hand side of this inequality is positive if $n > \frac{d}{\log(\mu)}$, i.e., from $\lceil \frac{d}{\log(\mu)} \rceil$ on the sequence on the right-hand side is monotonously increasing. Hence, if the inequality is true for some $n_0 \geq \lceil \frac{d}{\log(\mu)} \rceil$, then it is true for all $n \geq n_0$. Checking these values one by one, we will find a suitable n_0 eventually.

If the polynomial $p(x) = p_0$ is just constant, then $p(n) \geq (1-\varepsilon)^n$ if and only if $n \geq \frac{\log(p_0)}{\log(1-\varepsilon)}$. Otherwise, we can compute the largest real root x_1 of the derivative of $p(x)$. If $p(n_1) \geq (1-\varepsilon)^{n_1}$ for any $n_1 \geq \lceil x_1 \rceil$, then the inequality holds for all $n \geq n_1$.

Example 5. The sequence A000126 is C-finite of order 4 satisfying

$$c(n) - c(n+1) - 2c(n+2) + 3c(n+3) - c(n+4) = 0$$

with initial values $c = \langle 1, 2, 4, 8, \dots \rangle$. The sequence has the unique dominant root $\frac{1+\sqrt{5}}{2}$. Algorithm 1e and Algorithm 2e do not terminate in 60 s whereas Algorithm C terminates after checking the first 14 terms.

3.5 Combination of Algorithm 1 and Algorithm 2

In the case that the C-finite sequence has a unique dominant eigenvalue, we can combine the closed form representation of the sequence together with Algorithm 1e and Algorithm 2e. As we know that the polynomial term $p(n)$ in (3) certainly dominates the exponential terms, we can find indices n_i using Algorithm 1e and Algorithm 2e from which on the exponential sequences are dominated by the polynomial term. These input sequences have very low order (maximum order 3). Therefore, the termination criteria in [22] show that these algorithms terminate in most instances.

As Algorithm 2e terminates for essentially all sequences of order 2, the real algebraic part of Algorithm P certainly terminates.

Theorem 1. *Algorithm P terminates if $s = 0$, i.e., if all eigenvalues of c are real algebraic.*

Algorithm P. Positivity for sequences with dominant eigenvalues

```
Input  : $C$-finite sequence $c$ of the form (3)
output: true if $c(n) > 0$ for all $n \in \mathbb{N}$ and false otherwise
for $i \leftarrow 1$ to $s$ do
    $n_{i,\overline{\mathbb{Q}}} \leftarrow$ Algorithm 1e applied to $\frac{p(n)}{s+l} + o_i(n)\xi_i^n + \overline{o_i}(n)\overline{\xi_i}^n$
for $i \leftarrow 1$ to $l$ do
    $n_{i,\mathbb{A}} \leftarrow$ Algorithm 2e applied to $\frac{p(n)}{s+l} + q_i(n)\rho_i^n$
$n_0 \leftarrow \max(n_{1,\overline{\mathbb{Q}}}, \dots, n_{s,\overline{\mathbb{Q}}}, n_{1,\mathbb{A}}, \dots, n_{l,\mathbb{A}})$
if $c(n) > 0$ for $n = 0, \dots, n_0$ then
    return true
else
    return false
```

Proof. Each sequence $h(n) := \frac{p(n)}{s+l} + q_i(n)\rho_i^n$ is the sum of two balanced D-finite sequences g, f over $\mathbb{A}$ satisfying the recurrences

$$-p(n+1)g(n) + p(n)g(n+1) = 0, \quad -q_i(n+1)\rho_i f(n) + q_i(n)f(n+1) = 0$$

with characteristic polynomials

$$\chi(\mathcal{G}) = \operatorname{lc}(p)(y-1), \quad \chi(\mathcal{F}) = \operatorname{lc}(q_i)(y-\rho_i),$$

where $\mathcal{G}, \mathcal{F}$ denote the annihilating operators of g, f, respectively. As these characteristic polynomials are coprime, Lemma 2 yields

$$\chi(\mathcal{H}) = \chi(\mathcal{G})\chi(\mathcal{F}) = \gamma(y-1)(y-\rho_i)$$

for some constant γ where $\mathcal{H}$ denotes the annihilating operator of h. In particular, $\mathcal{H}$ is balanced. Furthermore, $h \sim p(n)$ by construction. With [22, Theorem 3], Algorithm 2e terminates with input h. □

It is conjectured that Algorithm 1e terminates for sequences of order 3 if the eigenvalues are complex. This is the case if we apply Algorithm 1e. Hence, if the conjecture is true, Algorithm P terminates for all C-finite sequences with a unique dominant eigenvalue.

Theorem 2. *Assume Conjecture 1 from [22] is true. Then, Algorithm P terminates.*

Proof. The proof of Theorem 1 already shows that the algorithm terminates for the real algebraic eigenvalues. Analogously, in the complex case, the sequences $h(n) := \frac{p(n)}{s+l} + o_i(n)\xi_i^n + \overline{o_i}(n)\overline{\xi_i}^n$ are D-finite of order 3 with a balanced annihilating operator $\mathcal{H}$ with characteristic polynomial

$$\chi(\mathcal{H}) = \gamma(y-1)(y-\xi_i)(y-\overline{\xi_i})$$

for some constant γ. With Conjecture 1, Algorithm 1e terminates on this input. □

Example 6. The sequence A002248 is C-finite of order 4 satisfying the recurrence

$$4c(n) - 8c(n+1) + 7c(n+2) - 4c(n+3) + c(n+4) = 0$$

with initial values $c = \langle 2, 8, 14, 16, \dots \rangle$. The sequence has the unique dominant eigenvalue 2. Neither Algorithm 1e nor Algorithm 2e terminate in 60 s. However, both Algorithm C and Algorithm P terminate in negligible time.

3.6 Decomposition into Non-degenerate Sequences

A C-finite sequence c is called *degenerate* if the ratio $\frac{\lambda_i}{\lambda_j}$ of two distinct eigenvalues λ_i, λ_j is a root of unity. Every C-finite sequence c can be written as the interlacing of non-degenerate and zero-sequences $c_1, \dots, c_k$ [8, Theorem 1.2]. For proving inequalities for C-finite sequences this decomposition often turned out useful [26,28,35]. For proving positivity of c we can compute this decomposition and prove positivity for every subsequence $c_1, \dots, c_k$.

One can explicitly compute the eigenvalues of a C-finite sequence and check whether the ratio of two eigenvalues is a root of unity [4]. Hence, a naive algorithm can decompose a sequence c into k subsequences

$$c_1(n) = c(kn), \dots, c_k(n) = c(kn + k - 1)$$

and check whether all these subsequences are either zero or non-degenerate. Eventually, for large enough k, this is the case. This already works well in practice as we see in Sect. 4. A more efficient algorithm is given in [36].

If decomposition into subsequences is used together with Algorithm C or Algorithm P, then it is more efficient to check whether every subsequence has a unique dominant root (which can be done numerically with arbitrary-precision arithmetic) instead for checking degeneracy. The main bottleneck (cf. Example 8) is usually the computations of the subsequences. Hence, an efficient implementation should certainly aim to minimize the computations of these subsequences.

Example 7. The sequence A000115 is C-finite of order 8 and satisfies the recurrence

$$\begin{aligned} c(n) - c(n+1) - c(n+2) &+ c(n+3) \\ -c(n+5) + c(n+6) + c(n+7) &- c(n+8) = 0. \end{aligned}$$

with initial values $c = \langle 1, 1, 2, 2, 3, 4, 5, 6, \dots \rangle$. It has 6 dominant eigenvalues and is degenerate. It can be decomposed into 10 non-degenerate sequences with unique dominant eigenvalues. For these subsequences, Algorithm C and Algorithm P both have no problem showing positivity.

4 Comparison

As far as we are aware the only implementations of the algorithms presented in Sect. 3 are implementations of the Gerhold–Kauers method for Mathematica in the package `SumCracker` [16] and for SageMath [34]. We have implemented the presented algorithms in SageMath (using QEPCAD-B) and in Mathematica and tested them on C-finite sequences which could be obtained from the OEIS by guessing.

4.1 Test Set

We used guessing on the terms given in the OEIS to check for each sequence whether it is C-finite. To have reasonable certainty that the guessed recurrence is indeed correct we make sure that the corresponding linear systems are overdetermined with at least 15 more equations than variables. We take the first 1000 of these sequences for which the first 500 terms are strictly positive and are therefore highly likely to be positive altogether[1].

The maximal order of these sequences is 42. The following table shows the number of sequences of each given order:

order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	> 15
	73	134	117	139	120	80	87	36	47	27	31	14	17	10	10	58

More than half of these sequences, 567, have a unique dominant eigenvalue. There are 102, 40, 70, 32 sequences with $2, 3, 4, 5$ distinct dominant eigenvalues, respectively. Hence, there are 139 sequences with more than 6 distinct dominant eigenvalues.

About half of the sequences, 513, have a characteristic polynomial which is not squarefree. By Lemma 3 these are the sequences which have a shorter D-finite recurrence.

4.2 SageMath Implementation

The methods for proving inequalities for C-finite sequences (and in a limited way for D-finite sequences) are part of the `rec_sequences` package which is itself based on the `ore_algebra` package [20]. SageMath provides an interface to QEPCAD-B which allows CAD computations [2,33]. This is used in the implementations of Algorithm 1 and Algorithm 2. For Algorithm C, we rely on fast arbitrary-precision arithmetic using the library Arb which is included in SageMath [15]. To decompose a sequence into subsequences with a unique dominant eigenvalue, we decompose the sequence into k subsequences and check, using arbitrary-precision arithmetic, whether all of these have a unique dominant eigenvalue. If they do not have a unique dominant eigenvalue, we increase k by one. The main bottleneck when decomposing is by far the computation of the subsequences. Checking whether a subsequence has a unique dominant eigenvalue or proving positivity of a sequence with a unique dominant eigenvalue using Algorithm C only takes negligible time in our examples.

[1] A table with these sequences and additional information is given on the website https://www3.risc.jku.at/people/pnuspl/PositivityCFinite. It also contains the detailed results of the SageMath and Mathematica tests.

The package is publicly available[2]. We give a list of the methods that can be used on C-finite sequences to show positivity. Every method has a parameter `strict` which is `True` by default and indicates whether strict positivity or non-negativity should be shown. The additional parameter `time` can be used to give an upper bound (in seconds) after which the algorithms should be terminated, the default value is -1, indicating that they should not stop prematurely.

- `is_positive_algo1` implements Algorithm 1 from [22]. As an additional parameter `bound` can be specified which gives an upper bound on the number of iterations.
- `is_positive_algo2` implements Algorithm 2 from [22]. Again, `bound` can be specified. This method is also implemented for general D-finite sequences and can be called using `is_positive` on D-finite sequences.
- `is_positive_dominant_root` implements Algorithm C for sequences with a unique dominant eigenvalue.
- `is_positive_dominant_root_decompose` first tries to decompose the sequence into sequences with a unique dominant eigenvalue and zero sequences and calls Algorithm C on each of those.
- `is_positive` is a combination of all these algorithms which additionally uses a reduction to D-finite sequences if possible. This method is also applied if the comparison operators >, <, >=, <= are used.

The following example session shows how the methods can be used.

```
sage: from rec_sequences.CFiniteSequenceRing import *
sage: C = CFiniteSequenceRing(QQ)
sage: f = C([1,1,-1], [0,1]) # Fibonacci numbers
sage: f.is_positive(strict=False)
True
sage: var("n")
sage: c1 = C(n^2+1) # A002522
sage: c1 >= 0 # use is_positive implicitly
True
sage: c2 = C([1, -1, -1, 1, 0, -1, 1, 1, -1],
sage:         [1, 1, 2, 2, 3, 4, 5, 6]) # A000115
sage: c2.is_positive_dominant_root_decompose()
True
sage: c = C(1/100 * (-3)^n + 100 * 2^n)
sage: c > 0
False
```

Using the above mentioned methods, 987 out of the 1000 sequences from the test set could be proven to be positive where each method was given 60 s. The following table gives an overview on the number of sequences which could be

[2] The package can be obtained from https://github.com/PhilippNuspl/rec_sequences. Extensive documentation and instructions for the installation can be found under the same link. The version used to run the experiments is available at https://github.com/PhilippNuspl/rec_sequences/tree/v0.1-exp.

proven to be positive by each method ("Comb." stands for a combination of the algorithms and a "D" indicates that decomposition of the sequence is used):

Algo. 1	Algo. 2	Algo. C	D, Algo. C	Comb.
384	327	566	984	986

It is clear that decomposing the sequences and using Algorithm C is the most powerful method. The implementation of Algorithm C is very fast and takes at most 0.3 s for every example we considered.

Example 8. The sequence A008628 is C-finite of order 13 satisfying

$$\begin{aligned}&c(n) - c(n+1) - 2c(n+2) + c(n+3) + 2c(n+4) - c(n+6)\\ &+c(n+7) - 2c(n+9) - c(n+10) + 2c(n+11) + c(n+12) - c(n+13) = 0\end{aligned}$$

with initial values $= c = \langle 1, 1, 2, 3, 5, 7, 10, 13, 18, 23, 31, 38, 49 \rangle$. If the sequence is decomposed into 30 subsequences, then all of the subsequences have a unique dominant root and positivity of these subsequences can be shown easily with Algorithm C. It takes about 2 min to show positivity of the sequence c and 98% of the time is used to compute the subsequences in the decomposition.

Allowing more than 60 s for each sequence, all 1000 sequences can be shown to be positive using decomposition into subsequences with a unique dominant eigenvalue and Algorithm C for these subsequences.

4.3 Mathematica Implementation

The Mathematica package `Positivity` encompasses several of the algorithms described in Sect. 3. It is part of `RISCErgoSum` which is a collection of Mathematica packages developed at RISC[3]. The package `GeneratingFunctions` is used to compute closure properties of C-finite sequences [25]. Our package, therefore, uses the same syntax as Mallinger's package for defining sequences. For the quantifier elimination steps in Algorithm 1e and Algorithm 2e, we use the Mathematica method `Resolve`. It might be interesting to compare different quantifier elimination procedures for our concrete examples. Following, we give a list of the methods contained in the `Positivity` package. All methods can be used in a strict version to show strict positivity of a sequence (this is the default) or a non-strict version to show non-negativity of a sequence using the parameter `Strict` set to `False`. If the parameter `Verbose` is set to `True`, then more information about the different computation steps are printed.

[3] It can be obtained from https://www3.risc.jku.at/research/combinat/software/ergosum/RISC/PositiveSequence.html. A demo notebook can be found on the same webpage. The source code is available on GitHub. The version used to run the experiments is available at https://github.com/PhilippNuspl/PositiveSequence/tree/v0.1-exp.

- `KPAlgorithm1` implements Algorithm 1e, i.e., for a C-finite or D-finite sequence an index n_0 is returned from which the sequence is guaranteed to be positive. If the parameter `Eventual` is set to `False`, then the traditional Algorithm 1 from [22] is executed which returns `True` if the sequence is positive or `False` if the sequence is not positive.
- `KPAlgorithm2` implements Algorithm 2e and Algorithm 2 from [22], analogous to `KPAlgorithm1`.
- `AlgorithmDominantRootClassic` is an implementation of Algorithm C.
- `AlgorithmDominantRootCAD` provides an implementation of Algorithm P.
- `AlgorithmClassic` and `AlgorithmCAD` first decompose the sequence into non-degenerate and zero sequences and check positivity of these subsequences with `AlgorithmDominantRootClassic` and `AlgorithmDominantRootCAD`, respectively.
- `PositiveSequence` combines some of the previous algorithms.

The methods can be used in the following way:

```
In[1]:= << RISC`Positivity`
In[2]:= f = RE[{{0, 1, 1, -1}, {0, 1}}, c[n]];
In[3]:= PositiveSequence[f, Strict -> False] (*Fibonacci*)

Out[3]= True

In[4]:= c1 = SeqFromExpr[n^2 + 1, c[n]];
In[5]:= PositiveSequence[c1] (*A002522*)

Out[5]= True

In[6]:= c2 = RE[{{0, 1, -1, -1, 1, 0, -1, 1, 1, -1}, {1, 1, 2, 2, 3, 4, 5, 6}}, c[n]];
In[7]:= AlgoClassic[c2] (*A000115*)

Out[7]= True

In[8]:= c3 = SeqFromExpr[1/100 * (-3)^n + 100 * 2^n, c[n]];
In[9]:= PositiveSequence[c3]

Out[9]= False
```

Comparing the different algorithms on the test set we see similar results as in the SageMath implementation. Every method was again aborted after 60 s. 980 out of the 1000 sequences could be shown to be positive by at least one of the methods. The following table shows the number of sequences which could be proven positive by each method:

Algo. 1	Algo. 2	Algo. C	Algo. P	D, Algo. C	D, Algo. P	Comb.
387	325	526	528	940	942	980

A more precise comparison of Algorithm C and Algorithm P shows that the two methods are not only equally powerful on the test set, but their runtime for the individual examples is also very similar. One can, however, expect that this is due to the specific implementation as the next example indicates. Hence, if provided by the computer algebra system, implementations based on numerical arbitrary-precision computations should be prefered over implementations based on algebraic number computations or quantifier elimination methods.

Example 9. The C-finite sequence A003520 is C-finite of order 5 satisfying

$$c(n) + c(n+4) - c(n+5) = 0$$

with initial values $c(0) = \cdots = c(4) = 1$. The sequence has a unique dominant root. The Mathematica implementations of Algorithm C and Algorithm P both take several seconds. The SageMath implementation based on arbitrary-precision ball arithmetic instead of computations with algebraic numbers takes less than 0.1 s.

Increasing the time shows that the combined algorithm can show the positivity of 996 sequences with a time limit of 12 hours per sequence.

5 Conclusions

Summarizing, we have investigated some well known and new methods for showing positivity of C-finite sequences. To our knowledge, most of these algorithms were never implemented and it was not clear how well they perform on practical examples. It turned out that the methods are already powerful enough to prove the positivity of most C-finite sequences from the OEIS in reasonable time.

The given algorithms already cover most of the sequences appearing in combinatorial examples. One can, however, construct examples of non-degenerate sequences which have multiple dominant eigenvalues. For sequences with up to 5 dominant eigenvalues, positivity is still known to be decidable [28]. Other algorithms for showing positivity are given for instance in [11] and [14]. It would certainly be interesting to check whether and how these methods can be applied and implemented in practice and how their runtime compares to the algorithms presented here.

Acknowledgments. We like to thank Ralf Hemmecke for providing helpful feedback on the Mathematica implementation. We thank the referees for their careful reading and their valuable suggestions that helped improve the quality of the paper. In particular, the suggestion of one of the reviewers to use arbitrary-precision arithmetic greatly improved the implementation of the classical method for showing positivity.

References

1. Basu, S., Roy, S., Pollack, R., Roy, M.F.: Algorithms in Real Algebraic Geometry. Algorithms and Computation in Mathematics, Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-33099-2
2. Brown, C.W.: QEPCAD B: a program for computing with semi-algebraic sets using CADs. SIGSAM Bull. **37**(4), 97–108 (2003)
3. Caviness, B.F., Johnson, J.R.: Quantifier Elimination and Cylindrical Algebraic Decomposition. Texts & Monographs in Symbolic Computation, Springer, Vienna (1998). https://doi.org/10.1007/978-3-7091-9459-1
4. Cohen, H.: A Course in Computational Algebraic Number Theory. Graduate Texts in Mathematics, Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-662-02945-9
5. Collins, G.E.: Quantifier elimination for real closed fields by cylindrical algebraic decompostion. In: Brakhage, H. (ed.) GI-Fachtagung 1975. LNCS, vol. 33, pp. 134–183. Springer, Heidelberg (1975). https://doi.org/10.1007/3-540-07407-4_17
6. Collins, G.E., Hong, H.: Partial cylindrical algebraic decomposition for quantifier elimination. J. Symb. Comput. **12**(3), 299–328 (1991)
7. Olver, F.W.J., et al.: NIST Digital Library of Mathematical Functions. Release 1.1.0 of 2020–12-15 (2021). http://dlmf.nist.gov
8. Everest, G., van der Poorten, A., Shparlinski, I., Ward, T.: Recurrence Sequences. Mathematical Surveys and Monographs, American Mathematical Society, Providence, USA (2015)
9. Gerhold, S.: Combinatorial Sequences: Non-Holonomicity and Inequalities. Ph.D. Thesis, Johannes Kepler University Linz (2005)
10. Gerhold, S., Kauers, M.: A Procedure for proving special function inequalities involving a discrete parameter. In: Proceedings of ISSAC 2005, Beijing, China, 24–27 July 2005. pp. 156–162 (2005)
11. Gourdon, X., Salvy, B.: Effective asymptotics of linear recurrences with rational coefficients. Discrete Math. **153**(1–3), 145–163 (1996)
12. Halava, V., Harju, T., Hirvensalo, M.: Positivity of second order linear recurrent sequences. Discrete Appl. Math. **154**(3), 447–451 (2006)
13. Halava, V., Harju, T., Hirvensalo, M., Karhumäki, J.: Skolem's Problem: On the Border Between Decidability and Undecidability. Technical Report (2005)
14. van der Hoeven, J.: Fuchsian holonomic sequences (2021). https://hal.archives-ouvertes.fr/hal-03291372/
15. Johansson, F.: Arb: efficient arbitrary-precision midpoint-radius interval arithmetic. IEEE Trans. Comput. **66**, 1281–1292 (2017)
16. Kauers, M.: SumCracker: a package for manipulating symbolic sums and related objects. J. Symb. Comput. **41**(9), 1039–1057 (2006)
17. Kauers, M.: Computer algebra and power series with positive coefficients. In: Proceedings of FPSAC 2007, pp. 1–7 (2007)
18. Kauers, M.: Bounds for D-finite closure properties. In: Proceedings of ISSAC 2014, Kobe, Japan, pp. 288–295. Association for Computing Machinery, New York, NY, USA (2014)
19. Kauers, M.: Algorithms for *D*-finite functions. In: JNCF 2015, Cluny, France (2015)
20. Kauers, M., Jaroschek, M., Johansson, F.: Ore polynomials in Sage. In: Computer Algebra and Polynomials: Applications of Algebra and Number Theory, pp. 105–125. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-15081-9_6

21. Kauers, M., Paule, P.: The Concrete Tetrahedron. Texts and Monographs in Symbolic Computation, Springer, Vienna (2011). https://doi.org/10.1007/978-3-7091-0445-3
22. Kauers, M., Pillwein, V.: When can we detect that a P-finite sequence is positive? In: Proceedings of ISSAC 2010, Munich, Germany, pp. 195–201. Association for Computing Machinery, New York, NY, USA (2010)
23. Laohakosol, V., Tangsupphathawat, P.: Positivity of third order linear recurrence sequences. Discrete Appl. Math. **157**(15), 3239–3248 (2009)
24. Li, H.C.: Studies on Generalized Vandermonde Matrices: Their Determinants, Inverses, Explicit LU Factorizations, with Applications. Ph.D. Thesis, National Chengchi University (2006)
25. Mallinger, C.: Algorithmic Manipulations and Transformations of Univariate Holonomic Functions and Sequences. Diplomarbeit, Johannes Kepler University Linz (1996)
26. Mignotte, M., Shorey, T.N., Tijdeman, R.: The distance between terms of an algebraic recurrence sequence. J. für die reine und angewandte Mathematik **349**, 63–76 (1984)
27. OEIS Foundation Inc.: The On-Line Encyclopedia of Integer Sequences (2022). http://www.oeis.org
28. Ouaknine, J., Worrell, J.: Positivity problems for low-order linear recurrence sequences. In: SODA 2014: Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 366–379 (2014)
29. Pillwein, V.: Positivity of certain sums over Jacobi kernel polynomials. Adv. Appl. Math. **41**(3), 365–377 (2008)
30. Pillwein, V.: Termination conditions for positivity proving procedures. In: Proceedings of ISSAC 2013, Boston, USA, 26–29 June 2013. pp. 315–322 (2013)
31. Pillwein, V.: On the positivity of the Gillis-Reznick-Zeilberger rational function. Adv. Appl. Math. **104**, 75–84 (2019)
32. Stanley, R.P.: Enumerative Combinatorics: Vol. 2, Cambridge Studies in Advanced Mathematics, Cambridge University Press (1999)
33. The Sage Developers: SageMath, the Sage Mathematics Software System (Version 9.4) (2022). https://www.sagemath.org
34. Uray, M.J.: On proving inequalities by cylindrical algebraic decomposition. Annales Univ. Sci. Budapest. Sect. Comp, pp. 231–252 (2020)
35. Vereshchagin, N.K.: Occurrence of zero in a linear recursive sequence. Mat. Zametki **38**(2), 609–615 (1985)
36. Yokoyama, K., Li, Z., Nemes, I.: Finding roots of unity among quotients of the roots of an integral polynomial. In: Proceedings of ISSAC 1995, Montreal, Quebec, Canada, 10–12 July 1995. pp. 85–89 (1995)