Journal of Symbolic Computation 60 (2014) 62–77

Contents lists available at ScienceDirect

Journal of Symbolic Computation

www.elsevier.com/locate/jsc

Closed form solutions of linear difference equations in terms of symmetric products ☆

Yongjae Cha

Johannes Kepler University, 4040 Linz, Austria

ARTICLE INFO

Article history:
Received 27 December 2012
Accepted 24 July 2013
Available online 9 October 2013

Keywords:
Algorithm
Difference equations
Sum of squares

ABSTRACT

In this paper we show how to find a closed form solution for third order difference operators in terms of solutions of second order operators. This work is an extension of previous results on finding closed form solutions of recurrence equations and a counterpart to existing results on differential equations. As motivation and application for this work, we discuss the problem of proving positivity of sequences given merely in terms of their defining recurrence relation. The main advantage of the present approach to earlier methods attacking the same problem is that our algorithm provides human-readable and verifiable, i.e., certified proofs.
© 2013 Elsevier B.V. All rights reserved.

1. Introduction

This paper presents an extension of the algorithm *solver* (Cha, 2011; Cha and van Hoeij, 2009; Cha et al., 2010) that returns closed form solutions for second order linear difference equations to third order linear difference equations. The solutions that we are looking for are in terms of (finite) sums of squares. This is motivated by applying the algorithm for proving inequalities on special functions, i.e., on expressions that may be defined in terms of linear difference equations with polynomial coefficients. Conjectures about positivity of special functions arise in many applications in mathematics and science. Proving them usually requires profound knowledge on relations between these special functions. It is well known that there exist many algorithms for proving and finding special function identities (Zeilberger, 1990; Chyzak, 2000; Petkovšek et al., 1996; Koutschan, 2010). For automated proving of special functions inequalities only few approaches exist. Gerhold and Kauers (2005), Kauers (2006) introduced a method that is based on Cylindrical Algebraic Decomposition (CAD). This

☆ Supported by the Austrian Science Fund (FWF) under grant P22748-N18 and P20347-N18.
E-mail address: ycha@risc.jku.at.
URL: http://www.risc.jku.at/home/ycha.

0747-7171/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.jsc.2013.10.002

method has been proven to work well on many nontrivial examples (Gerhold and Kauers, 2006; Pillwein, 2008), but even though correctness is easy to be seen, termination cannot be guaranteed, hence it is not an algorithm in the strict sense. A first attempt to clarify the latter issue has been made in Kauers and Pillwein (2010). One of the features of proofs of special functions identities is that they usually come with a certificate, i.e., some easy to check identity that verifies the proof. The CAD-based approach cannot hope to have a similar certificate in the near future. The method presented here is a first step toward human readable proofs of special functions inequalities, although admittedly a representation in terms of sums of squares with positive coefficients is not expected to exist for any given input. Besides this application, the results presented are of independent interest as they provide difference case counterparts to results obtained for the differential case (Singer, 1985; van Hoeij, 2007).

First we review the available results in the differential case. Let k be a differential field and $L_d \in k[\partial]$, $\partial = d/dx$ be a linear homogeneous third order differential operator. Singer (1985) characterizes when solutions of L_d can be written in terms of solutions of a second order operator in $\bar{k}[\partial]$. Van Hoeij (2007) handles the similar problem when the coefficients of the second order operator are restricted to k and shows that it will be either of the following cases.

Case 1 L_d is the symmetric square of a second order operator $K_d \in k[\partial]$;
Case 2 L_d is gauge equivalent to a symmetric square of a second order operator $K_d \in k[\partial]$.

The definitions of symmetric products and gauge equivalence are recalled in Sections 2.3 and 2.4 below. The algorithm given in van Hoeij (2007) returns a second order differential operator, $K_d \in k[d/dx]$, and a gauge transformation in $k[\partial]$ that sends solutions of the symmetric squares of K_d to solutions of L_d for Case 2.

In the differential case, the symmetric square of L_d has order 5 if and only if we are in Case 1. In this case, there is a simple formula that gives K_d. Case 2 is equivalent to the symmetric square of L_d having order 6 and a first order right-hand side factor in $k[\partial]$ as well as a certain conic of L_d (Singer, 1985, Eq. 4.2.1) having a non-zero solution in k. Since for $k = \mathbb{C}(x)$ this conic is solvable over $\mathbb{C}(x)$, the last condition becomes trivial in this case. The algorithm given in van Hoeij (2007) in the first step checks the order of the symmetric square of L_d to distinguish the cases.

The difference case behaves differently; here we denote by $D = \mathbb{C}(x)[\tau]$ the ring of linear difference operators, where τ denotes the shift operator. Example 2.15 shows that the cases cannot be distinguished according to the order of the symmetric squares when the coefficients are in $\mathbb{C}(x)$. To set up a counterpart theorem for difference equations, this example shows that we need one more transformation than that in the differential case. Furthermore in Case 1, the algorithm for finding the second order operator is more complicated than in the differential case.

Summarizing, the ideas used in the differential case cannot be carried over immediately to the difference case. Furthermore our aim is to have a closed form solution of the given input. Hence, if a factorization is found that is not solvable, this fails to satisfy our goal. Thus we build on the ideas of the algorithm *solver* (Cha, 2011; Cha and van Hoeij, 2009; Cha et al., 2010). Here we say that a function is in closed form if it is a linear combination of elementary functions, special functions or hypergeometric functions over $\mathbb{C}(x)$. For instance the modified Bessel function of the first kind is a closed form solution of the second order operator $L_b := z\tau^2 - (2x+2)\tau + x + z$.

The algorithm *solver* returns closed form solutions for second order linear difference operators. The main idea of *solver* is to map the given operator L_1 to an operator L_2 of which a solution is known. This transformation is a bijective map, called GT-transformation, that sends solutions of L_1 to solutions of L_2. If a closed form solution to one of the operators is known, then by means of this transformation the solution of the second operator can be constructed. For this purpose a table with second order operators including parameters together with characteristic data (local data) has been constructed. This local data can be computed for the given operator, the corresponding equivalent operator is found by table look-up. Then by comparing parameters of the local data the GT-transformation can be constructed. The characteristic data is described in Section 3. To cover the extension described here the table has been extended so that we can give closed form solutions of certain third order linear difference operators.

Download English Version:

https://daneshyari.com/en/article/401187

Download Persian Version:

https://daneshyari.com/article/401187

Daneshyari.com