Symbolic Solution of Simple BVPs on the Operator Level: A New Approach

Markus Rosenkranz[1]

[1]Johannes Kepler University of Linz, Austria,
Research Institute for Symbolic Computation,
Markus.Rosenkranz@risc.uni-linz.ac.at

Abstract. We present a new method for solving simple BVPs, using noncommutative polynomials for modeling integral, differential and boundary operators. The method is based on right inversion of linear differential operators with constant coefficients and uses a fixed Gröbner basis for normalizing the resulting Green's operator. We have implemented the algorithm in *Theorema*. The paper concludes with presenting a sample computation carried out by our implementation.

This extended abstract is based on a poster presented at ISSAC'03, to appear in the SIGSAM bulletin. The work was supported by the Austrian Science Foundation FWF under the SFB grants F1302 and F1308.

AMS Subject Classification: 34B05.

Keywords and phrases: Symbolic Methods for ODEs, Linear BVPs.

1 Introduction

In [6, 8] we have presented a new approach for *solving simple BVPs* (throughout this extended abstract this means "BVPs for linear differential operators with constant coefficients such that there is a unique solution") via noncommutative Gröbner bases. In the course of the ongoing PhD thesis [7], we have found another new method, which has several advantages:

- It avoids the costly computation of a Gröbner bases by applying right inversion and one *fixed Gröbner basis* for reducing the Green's operator to standard form.
- Our new method is fully algorithmic and works for any simple BVP. It has been implemented in *Theorema* [2].
- It does *not need Gaussian elimination for symbolic expressions* as e.g. in the standard formula for the Green's function; see page 189 in [4].

The paper is *structured* as follows. We will specify in detail which type of problem we want to solve. Then we describe the main features of our solution algorithm, whose central datatype is the Green's polynomials. Finally, we present how a sample BVP is solved by our implementation.

Our topic is related but yet fundamentally different from the more common task of *symbolically solving differential equations*, as pursued e.g. by Bronstein, Singer, Ulmer, van der Put; see the ISSAC proceedings of 2003 and 2002 for a representative survey on the state of the art in this field. The essential difference between boundary problems and differential equations is that one searches for an operator rather than a function—as detailed in the next section.

2 Problem Specification

Let $[a, b]$ be a finite interval in $\mathbb{R}$, and let T be a linear differential operator with constant coefficients given by

$$T\,u = c_0\,u^{(n)} + \ldots + c_{n-1}\,u' + c_n\,u,$$

where c_0 is nonzero. We view T as a linear operator on the Banach space $(C[a, b], ||\cdot||_\infty)$ with dense domain of definition $\mathfrak{D}(T) = C^n[a, b]$. The boundary operators are defined on the same domain; for each $i = 1, \ldots, n$ we have

$$\begin{aligned} B_i\ u = {} & p_{i,0}\,u^{(n)}(a) + \ldots + p_{i,n-1}\,u'(a) + p_{i,n}\,u(a) \\ & + q_{i,0}\,u^{(n)}(b) + \ldots + q_{i,n-1}\,u'(b) + q_{i,n}\,u(b), \end{aligned}$$

where the coefficients $p_{i,j}, q_{i,j}$ are real numbers. Now the *boundary value problem induced by* T *and* $B_1, \ldots, B_n$ is to find for each right-hand side $f \in C[a, b]$ a function $u \in C^n[a, b]$ (we assume its existence and uniqueness throughout this note, restricting ourselves to regular BVPs) such that

$$\begin{aligned} & T\ u = f, \\ & B_1\,u = \ldots = B_n\,u = 0. \end{aligned} \qquad (1)$$

Now we *could* view this as a differential equation parametrized by the forcing function f, but this is a rather artifical interpretation in the sense of computer algebra: Solution algorithms for ODEs are typically specialized to a certain class of functions —like Liouvillian extensions—for exploiting all the structural information available. However, in the BVP (1), we view f as ranging over the nonalgorithmic domain $C[a, b]$. Therefore the natural interpretation of (1) is to search not for a *function* $u \in C^n[a, b]$ for the infinitely many instantiations of f but for an *operator* $G : C[a, b] \leftarrow C^n[a, b]$, mapping each $f \in C[a, b]$ to “its” solution $u \in C^n[a, b]$. The operator G is known as the *Green's operator* [9].

The Green's operator can be defined analogously for many other types of BVPs for ODEs and PDEs, and it can often be described as an integral operator having a so-called *Green's function* g as its kernel. In the case of (1), this is indeed possible [3], leading to the Green's operator

$$G\,f(x) = \int_a^b g(x, \xi)\,f(\xi)\,d\xi. \qquad (2)$$

Thus one *can* reduce the search for the operator G to the search of the (bivariate!) function g, and there is a solution method going along these lines [4].

However, working directly on the *operator level* seems more natural to us, and we will now outline a new method for computing G in a suitable polynomial setting [6, 7]. The Green's function g can be extracted from G in a trivial post-processing step, if this is desired. Apart from the conceptual advantages, our method may also be superior to [4] on efficiency grounds, but this should be analyzed in detail later.

3 The Green's Polynomials

Our approach models the involved differential, integral and boundary operators as *noncommutative polynomials*: The differentiation $u \mapsto u'$ is represented by the indeterminate D, the antiderivative operator $u \mapsto (x \mapsto \int_a^x u(\xi)\, d\xi)$ by A, its dual $u \mapsto (x \mapsto \int_x^b u(\xi)\, d\xi)$ by B, the left boundary operator $u \mapsto (x \mapsto u(a))$ by L, and the right counterpart $u \mapsto (x \mapsto u(b))$ by R. Moreover, we have a parametrized family of multiplication operators M_f representing $u \mapsto (x \mapsto f(x)\, u(x))$.

The functions f are assumed to range over some algebra $\mathfrak{F}$ of functions that should fulfill suitable closure axioms coming out as a natural extension of the well-known axioms for a differential algebra. We call such an $\mathfrak{F}$ an *analytic algebra*; see [7] for details. For the results presented here, it is sufficient to take $\{x \mapsto x^k\, e^{\lambda x} \mid k \in \mathbb{N} \wedge \lambda \in \mathbb{C}\}$ for $\mathfrak{F}$. The noncommutative polynomial ring

$$\mathfrak{An}(\mathfrak{F}) = \mathbb{C}\langle A, B, D, L, R, M_f \mid f \in \mathfrak{F}\rangle$$

will be called *analytic polynomials*.

The *algorithm for solving* a BVP of the type (1) proceeds in three phases. First we compute a projector $P \in \mathfrak{An}(\mathfrak{F})$ onto the nullspace of T by using some trivial linear algebra on the fundamental system of T (the latter is typically presupposed when solving a BVP). Second we employ the Moore-Penrose theory [5] for reducing (1) to the right-inversion problem $G\,T = 1 - P$, which can be solved immediately by factoring the characteristic polynomial of T. Third we reduce the resulting expression $(1 - P)\, T^\blacklozenge$ (with $T^\blacklozenge \in \mathfrak{An}$ being the right inverse) with respect to a carefully selected system of 36 polynomial equations. The resulting polynomial is in a normal form that allows to read off the Green's function (2) immediately.

The polynomial equation system is the core idea for the algebraization employed in our approach. It captures the essential interactions between the indeterminates of $\mathfrak{An}(\mathfrak{F})$. For example, we have the equation $D\,A = 1$, expressing the Fundamental Theorem of Calculus, the product rule, and integration by parts. We have proved that the rewrite system induced by these equations is noetherian and confluent; thus one can regard it as a noncommutative Gröbner basis [1]. The ideal generated by them identifies all those operators that should be equal on analytic grounds; we call the corresponding factor algebra the *Green's polynomials*. The algorithm computes the Green's polyomial associated with the Green's operator by providing its canonical representative.

4 An Example

The algorithm outlined above has been implemented in Mathematica, using the framework of the Theorema system, an integrated working environment for proving, solving and computing in various domains [2]. The following call solves the boundary value problem for the differential operator $T = D^2 + 2D + 1$ for the boundary conditions $Lu = 0$ and $Ru = 0$. We give the input and output verbatim:

$$\begin{aligned}
&\text{1In[13]:=} \quad \text{Compute[Green}[D^{2} + 2D + 1, \langle L, R\rangle, by \to GreenEvaluator]\\
&\text{Out[13]=} \quad (1-\pi^{-1})\lceil e^{-x}x\rceil A\lceil e^{x}\rceil - \lceil e^{-x}\rceil A\lceil e^{x}x\rceil + \pi^{-1}\lceil e^{-x}x\rceil A\lceil e^{x}x\rceil\\
&\qquad\qquad - \pi^{-1}\lceil e^{-x}x\rceil B\lceil e^{x}\rceil + \pi^{-1}\lceil e^{-x}x\rceil B\lceil e^{x}x\rceil
\end{aligned}$$

The multiplication operators M_f are denoted by $\lceil f\rceil$ for the sake of readability. Note that one can immediately read off the corresponding term $g(x, \xi)$ for the Green's function (2), which is typically defined by a case distiction on $\xi < x$ and $\xi > x$: The summands with A go into the first case, those with B into the second; the multiplication operators before A and B yield terms in x, those after yield terms in ξ.

The *computing time* for the example above is below one minute on a Pentium i686, which is rather short in the light of Mathematica's interpretation strategy: It is known that self-made functions are slower by a factor of 100 to 1000 when compared to C (using many built-in library functions, reduced the factor down to almost 2). Thus Mathematica should only be used as a convenient platform for early prototyping, which is precisely the intention of our current research phase.

5 Conclusion

The method we have presented in this extended abstract can solve any regular BVP for linear differential operators with constant coefficients. It is clear, however, that the essential ideas contained in our approach can be transferred to *various more general settings*, if one manages to make the appropriate adaptions. For example, the computation of the nullspace projector and the reduction with respect to interaction equalities do not presuppose constant coefficients in the given differential operator; only the right inversion must be slightly generalized.

Besides this, one may use very similar techniques for approaching certain easy partial differential equations (using D_x and D_y instead of just D, etc); the crucial point is to find an appropriate notion of boundary operators. These and similar questions will be addressed by the author in the course of further research after the PhD thesis [7].

References

1. B. Buchberger. *Introduction to Gröbner Bases.* In B. Buchberger, F. Winkler, *Gröbner Bases and Applications*, Cambridge University Press, Cambridge (UK), 1998. Pages 3–31.

2. B. Buchberger, W. Windsteiger. *The Theorema Language: Implementing Object- and Meta-Level Usage of Symbols.* In: Proceedings of Calculemus 98 (Eindhoven, Netherlands), 1998.
3. E.A. Coddington, N. Levinson. *Theory of Ordinary Differential Equations*, McGraw-Hill Book Company, New York, 1955.
4. E. Kamke. *Differentialgleichungen und Lösungsmethoden (Volume 1)*, Teubner, Stuttgart, tenth edition, 1983. Pages 190–193.
5. M.Z. Nashed (Ed.). *Generalized Inverses and Applications*, Proceedings of an Advanced Seminar Sponsored by the Mathematics Research Center, Academic Press, New York, 1976. Pages 57–61.
6. M. Rosenkranz, B. Buchberger, H.W. Engl. Solving Linear BVP via Non-Commutative Gröbner Bases, accepted to *Applicable Analysis*, 2003.
7. M. Rosenkranz. *A Polynomial Approach to Linear Boundary Value Problems.* PhD thesis, to be submitted in 2003.
8. M. Rosenkranz. *Solving Linear Boundary Value Problems via Non-Commutative Gröbner Bases.* Extended abstract for SYNASC'02, Timisoara, Romania, October 2002.
9. I. Stakgold. *Green's Functions and Boundary Value Problems*, John Wiley & Sons, New York, 1979.