Theoretical Computer Science 681 (2017) 3–26

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

A contour integral approach to the computation of invariant pairs

Moulay Barkatou, Paola Boito, Esteban Segura Ugalde *

Université de Limoges, CNRS, XLIM UMR 7252, 123 avenue Albert Thomas, 87060 Limoges Cedex, France

ARTICLE INFO

Article history:
Available online 30 March 2017

Keywords:
Matrix polynomials
Eigenvalues
Invariant pairs
Contour integral
Moments
Solvents
Triangularization

ABSTRACT

We study some aspects of the invariant pair problem for matrix polynomials, as introduced by Betcke and Kressner [3] and by Beyn and Thümmler [6]. Invariant pairs extend the notion of eigenvalue–eigenvector pairs, providing a counterpart of invariant subspaces for the nonlinear case. We compute formulations for the condition numbers and the backward error for invariant pairs and solvents. We then adapt the Sakurai–Sugiura moment method [1] to the computation of invariant pairs, including some classes of problems that have multiple eigenvalues. Numerical refinement via a variant of Newton's method is also studied. Furthermore, we investigate the relation between the matrix solvent problem and the triangularization of matrix polynomials.

1. Introduction

Invariant pairs, introduced and analyzed in [3,6] and [34], are a generalization of eigenpairs for matrix polynomials. Let $P(\lambda) = \sum_{j=0}^{\ell} A_j \lambda^j$ be an $n \times n$ matrix polynomial, and choose a positive integer k. Then the matrices X, S of sizes $n \times k$ and $k \times k$, respectively, form an invariant pair of size k for $P(\lambda)$ if

$$P(X, S) := \sum_{j=0}^{\ell} A_j X S^j = 0.$$

Invariant pairs offer a unified theoretical perspective on the problem of computing several eigenvalue–eigenvector pairs for a given matrix polynomial. From a numerical point of view, moreover, the computation of an invariant pair tends to be more stable than the computation of single eigenpairs, particularly in the case of multiple or tightly clustered eigenvalues. Observe that the notion of invariant pairs can also be applied to more general nonlinear problems, although here we will limit our presentation to matrix polynomials.

How to compute invariant pairs? Beyn and Thümmler [6] adopt a continuation method of predictor–corrector type. Betcke and Kressner [3], on the other hand, establish a correspondence between invariant pairs of a given polynomial and of its linearizations. Invariant pairs for $P(\lambda)$ are extracted from invariant pairs of a linearized form and then refined via Newton's method.

The approach we take in this paper to compute invariant pairs is based on contour integrals. Being able to specify the contour Γ allows us to select invariant pairs that have eigenvalues in a prescribed part of the complex plane. Contour in-

* Corresponding author.
E-mail addresses: moulay.barkatou@unilim.fr (M. Barkatou), paola.boito@unilim.fr (P. Boito), esteban.segura@etu.unilim.fr (E. Segura Ugalde).

http://dx.doi.org/10.1016/j.tcs.2017.03.024
0304-3975/

tegrals play an important role in the definition and computation of moments, which form a Hankel matrix pencil yielding the eigenvalues of the given matrix polynomial that belong to the prescribed contour. The use of Hankel pencils of moment matrices is widespread in several applications such as control theory, signal processing or shape reconstruction, but non-linear eigenvalue–eigenvector problems can also be tackled through this approach, as suggested for instance in [1] and [4]. E. Polizzi's FEAST algorithm [31] is also an interesting example of contour-integral based eigensolver applied to large-scale electronic structure computations.

Here we adapt such methods to the computation of invariant pairs. This work studies in particular the scalar moment method and its relation with the multiplicity structure of the eigenvalues, but it also explores the behavior of the block version. We seek to compute invariant pairs while avoiding the linearization of $P(\lambda)$, which explains the choice of moment methods. We are motivated, in part, by the problem of better understanding the invariant pair problem in a symbolic or symbolic–numeric framework, that is, computing invariant pairs exactly, or with arbitrary accuracy via an effective combination of symbolic and numerical techniques: this is one of the reasons why we are interested in the issue of eigenvalue multiplicity.

The last part of the paper shows an application of our results on invariant pairs to the particular case of matrix solvents, that is, to the matrix equation

$$P(S) := \sum_{j=0}^{\ell} A_j S^j = 0.$$

The matrix solvent problem has received remarkable attention in the literature, since Sylvester's work [33] in the 1880s. The relation between the Riccati and the quadratic matrix equation is highlighted in [7], whereas a study on the existence of solvents can be found in [13]. Several works address the problem of computing a numerical approximation for the solution of the quadratic matrix equation: an approach to compute, when possible, the dominant solvent is proposed in [12]. Newton's method and some variations are also used to approximate solvents numerically: see for example [11,22,21, 27]. The work in [19] uses interval arithmetic to compute an interval matrix containing the exact solution to the quadratic matrix equation. For the case of the general matrix solvent problem, we can also cite [9,30] and [23]. On the other hand, the question of designing symbolic algorithms for computing solvents remains relatively unexplored. Attempts have been made to formulate the problem as a system of polynomial equations which can be solved via standard methods. However, this approach becomes cumbersome for problems of large size (see [20]).

Here we exhibit computable formulations for the condition number and backward error of the general matrix solvent problem, generalizing existing works on the quadratic matrix equation. Moreover, we propose an adaptation of the moment method to the computation of solvents. Finally, we build on existing work on triangularization of matrix polynomials (see [38] and [35]) and explore the relationship between solvents of matrix polynomials in general and in triangularized form.

The paper is organized as follows. Section 2 introduces preliminary notions, definitions and notation. The backward error and condition number for the invariant pair problem are computed in Section 2.1.

Section 3 is devoted to the computation of eigenvalues and invariant pairs through moments and Hankel pencils. Our main results here consist in Theorem 5, Corollary 1 and Theorem 7. A comparison of different techniques for numerical refinement of invariant pairs is presented in Section 3.4.

Finally, Sections 4, 5 and 6 are devoted to the applications to matrix solvents.

The methods presented in the paper have been implemented both in a symbolic (exact) and in a numerical version. The Maple and Matlab codes are available online at the URL http://www.unilim.fr/pages_perso/esteban.segura/software.html.

2. Matrix polynomials and invariant pairs

A complex $n \times n$ *matrix polynomial* $P(\lambda)$ of degree ℓ takes the form:

$$P(\lambda) = A_0 + A_1\lambda + A_2\lambda^2 + \cdots + A_\ell\lambda^\ell \tag{1}$$

where $A_\ell \neq 0$ and $A_i \in \mathbb{C}^{n\times n}$, for $i = 0, \ldots, \ell$.

In this work, we assume that $P(\lambda)$ is regular, i.e., $\det P(\lambda)$ does not vanish identically.

A crucial property of matrix polynomials is the existence of the Smith form (see, e.g., [17]):

Theorem 1. *[Thm. S1.1, [17]] Every $n \times n$ regular matrix polynomial $P(\lambda)$ admits the representation*

$$D(\lambda) = E(\lambda)P(\lambda)F(\lambda), \tag{2}$$

where $D(\lambda) = \operatorname{diag}(d_1(\lambda), \ldots, d_n(\lambda))$ is a diagonal polynomial matrix with monic scalar polynomials $d_i(\lambda)$ such that $d_i(\lambda)$ is divisible by $d_{i-1}(\lambda)$; $E(\lambda)$ and $F(\lambda)$ are matrix polynomials of size $n \times n$ with constant nonzero determinants.

The *polynomial eigenvalue problem* (PEP) consists in determining right eigenvalue–eigenvector pairs $(\lambda, x) \in \mathbb{C} \times \mathbb{C}^n$, with $x \neq 0$, such that

$$P(\lambda)x = 0,$$