Interfacing with the Numerical Homotopy Algorithms in *PHCpack*[*]

Anton Leykin[1] and Jan Verschelde[2]

Department of Mathematics, Statistics, and Computer Science
University of Illinois at Chicago, 851 South Morgan (M/C 249)
Chicago, IL 60607-7045, USA.
[1]`leykin@math.uic.edu, http://www.math.uic.edu/~leykin`
[2]`jan@math.uic.edu, http://www.math.uic.edu/~jan`

Abstract. Our Maple package *PHCmaple* came to existence in 2004 when it provided a convenient interface to the basic functionality of `phc`, a program which is a part of *PHCpack* and implements numeric algorithms for solving polynomial systems using polynomial homotopy continuation. Following the recent development of *PHCpack* the package has been extended with functions that deal with singular polynomial systems, in particular, the deflation procedures that guarantee the ability to refine approximations to an isolated solution even if it is multiple. We see *PHCmaple* as a part of a larger project to integrate a numerical solver in a computer algebra system.

1 Introduction

In its current state *PHCmaple* is an interface to `phc` (**p**olynomial **h**omotopy **c**ontinuation), a program to solve polynomial systems, which is part of *PHCpack* [11].

Polynomial systems occur frequently in various models of science and engineering. There exist various symbolic methods for finding the exact solutions of such systems. However, these methods are of high complexity and, therefore, might not be a good choice, especially in the situation where one would be satisfied with numerical approximations to the solutions.

Homotopy continuation methods (see e.g. [1], [6], and [8]) constitute a class of efficient symbolic-numerical solvers that produce such approximations. The nature of continuation methods lies in a repeated application of Newton's method in order to follow continuation paths. A homotopy method creates a family of polynomial systems (i.e.: the homotopy), linking the system to be solved with a so-called start system whose solutions are easy to find. For efficiency, polynomial homotopy methods exploit the structure of the polynomials in the system considering the polynomials not merely as functions as other continuation methods do. It is this aspect that makes application of a homotopy method a

[*] This material is based upon work supported by the National Science Foundation under Grant No. 0134611 and Grant No. 0410036. *Date:* 2 June 2006.

symbolic-numerical operation and is a cornerstone in the foundation of *numerical algebraic geometry*. One good recent book on the subject is [10].

The package *PHCpack* implements various homotopy techniques to approximate all isolated complex roots of a polynomial system. The first release of the source code is documented by [11]. In [9], there is a description of recent extensions to *PHCpack* to deal with positive dimensional solution sets. *PHCpack* contains the source code to build the program `phc`, available for various platforms: Sun workstations running Solaris, MacOS X computers, PCs under Linux and Windows.

Maple itself does not need an introduction. This computer algebra system is instrumental in setting up mathematical models in order to derive and formulate the polynomial equations. Equally important is the analysis and visualization of the results obtained by a solver. We see the interaction with computer algebra not only as natural, but also vital in the solving process. A simple Maple procedure, operating as a shell around the blackbox solver of *PHCpack*, was first presented in [9]. In *PHCmaple* we implement a much more elaborate interface shaped as a Maple package. *PHCmaple* provides format conversions to bring systems and solutions from Maple into `phc` and from `phc` into Maple. The interface gives access to most of the functions implemented in *PHCpack*.

Besides a useful application of computer algebra system, an important research direction in recent years has been the application of numerical methods to the classical problems of symbolic computation. The numerical irreducible decomposition is one of the major highlights. With the recent implementation of the deflation algorithms, now an even more challenging task is at reach – numerical primary decomposition, which would recover embedded components of the solution set as well as the maximal ones and also compute the multiplicities of the components. The deflation methods became the most recent acquisition of *PHCmaple*.

2 Evolution of Interfaces to *PHCpack*

In this section we briefly describe the chronological evolution of the interfaces to the capabilities in *PHCpack*.

1. **OpenXM [7] calls the blackbox solver.** The first interface is still available via OpenXM (Open message eXchange protocol for Mathematics) and only needs an executable file of the program. Just as one calls the blackbox solver of *PHCpack* as `phc -b input output`, a simple system call achieves the same effect.
2. **A simple Maple procedure calls the blackbox solver.** On [9, page 114], a simple Maple 7 procedure (less than 20 lines long) applies the experience of the first interface.
3. **A functional C interface to the Ada routines in *PHCpack*.** To process the output of the Pieri homotopies in *PHCpack* to compute feedback laws to control a linear system, a dedicated C interface was written, used in [12]

and described in an online appendix (available at the second author's web site). The main C program calls the Ada routines in *PHCpack* which then call another C function to process the output.

4. ***PHCmaple* gives access to the tools of `phc`.** The main executable program of *PHCpack* can be used as a blackbox or as a toolbox, calling the program with the appropriate options and selecting the desired actions from the menu. Via input redirections, it also just takes the executable version to gain access to the tools offered by *PHCpack*. In [3], we presented *PHCmaple*, a Maple interface to *PHCpack*.
5. **Using *PHCpack* as a state machine.** The dedicated C interface was not adequate for the parallel implementation of the path tracking routines in *PHCpack*. Therefore, a new interface was developed for use in [13], [4, 2] and [14].
   The Ada function `use_c2phc`

```
function use_c2phc ( job : integer;
                     a : C_intarrs.Pointer;
                     b : C_intarrs.Pointer;
                     c : C_dblarrs.Pointer ) return integer;
```

   is available to the C programmer as

```
extern void adainit( void );
extern int _ada_use_c2phc ( int job, int *a, int *b, double *c );
extern void adafinal( void );
```

   The first parameter `job` specifies the action requested from `phc`. The meaning of the other parameters `a`, `b`, and `c` depends on the `job`.

The chronological evolution pictures two distinct trends in interfacing with *PHCpack*: (1) using the executable originated with OpenXM; and (2) calling the compiled code.

3 Overview of *PHCmaple*

PHCmaple is available for download at `www.math.uic.edu/~leykin/PHCmaple/` and works with Maple version 8 or higher.

The goal of *PHCmaple* is to provide computer algebra users with a convenient interface to the

- blackbox solver of `phc`;
- homotopy path tracking facilities;
- deflation procedure;
- routines that create and manipulate witness sets for positive-dimensional components;
- factorization/decomposition capabilities of `phc`.

Using the following procedures one can deal with *isolated solutions* of square systems (with a finite number of solutions): to run the black-box solver execute **solve**, which returns approximations to all complex isolated roots of a square system; refines the solutions to any specified precision with **refine**, which also provides a way to set certain parameters in order to fine-tune the solver; **track** a subset of the solutions set of the start system to the corresponding solutions of the target system and visualize the results with **drawPaths**; for singular isolated solutions one might consider applying **deflationStep**, the implementation of the first-order *deflation* procedure [5], which given a polynomial system and an approximation to one of its multiple isolated solutions produces a new system of equations that has the same solution, but with lower multiplicity.

The positive-dimensional solution sets of general polynomial systems can be represented by means of *witness sets*, computing which reduces the problem to the isolated solution case.

The following functions of *PHCmaple* serve this purpose: construct an embedded system with **embed** in assumption that the dimension of its solution set is known; **cascade** runs the so-called *cascade of homotopies* for an embedded system, it computes the list of *witness sets* for the components of the solution set in every dimension; after producing the witness sets **filter** the points in lower-dimensional witness sets belonging to higher-dimensional components; to produce a *numeric irreducible decomposition* of a pure-dimensional solution component **decompose** its witness set; absolute factorization capability for multivariate polynomial is given by **factor**.

4 Development details and future prospects

At this moment *PHCmaple* operates by making system calls in order to launch a standalone process containing `phc`, the *PHCpack* executable. The current implementation was carefully tested on Windows versions up to Maple 10 and shows robust performance. However, the same experiments on Maple for Linux/Unix exhibited flaws in the mechanism of making external system calls.

In the future, the authors plan to part with this interfacing strategy in favor of a more efficient approach: calling *PHCpack* functions directly from a dynamical library. The current work on the C bindings for *PHCpack* (originally coded in Ada) makes the first step towards creation of such a library.

Apart from Maple our future plan is to look into creating an interfaces with the non-commercial computer algebra systems like Axiom and Macaulay 2 as well as tools for numerical computation like Octave and Scilab. We also find interesting the work of Joris Van der Hoeven on Mathemagix that promises to be "a bridge between symbolic computation and numerical analysis or symbolic computation".

References

1. E.L. Allgower and K. Georg. *Introduction to Numerical Continuation Methods*, volume 45 of *Classics in Applied Mathematics*. SIAM, 2003.

2. A. Leykin and J. Verschelde. Decomposing solution sets of polynomial systems: a new parallel monodromy breakup algorithm. Accepted for publication in *The International Journal of Computational Science and Engineering.*
3. A. Leykin and J. Verschelde. PHCmaple: A Maple interface to the numerical homotopy algorithms in PHCpack. In Quoc-Nam Tran, editor, *Proceedings of the Tenth International Conference on Applications of Computer Algebra (ACA'2004)*, pages 139–147, 2004.
4. A. Leykin and J. Verschelde. Factoring solution sets of polynomial systems in parallel. In Tor Skeie and Chu-Sing Yang, editors, *Proceedings of the 2005 International Conference on Parallel Processing Workshops. 14-17 June 2005. Oslo, Norway. High Performance Scientific and Engineering Computing*, pages 173–180. IEEE Computer Society, 2005.
5. A Leykin, J. Verschelde, and A. Zhao. Newton's method with deflation for isolated singularities of polynomial systems. To appear in *Theoretical Computer Science.*
6. T.Y. Li. Numerical solution of polynomial systems by homotopy continuation methods. In F. Cucker, editor, *Handbook of Numerical Analysis. Volume XI. Special Volume: Foundations of Computational Mathematics*, pages 209–304. North-Holland, 2003.
7. M. Maekawa, M. Noro, K. Ohara, Y. Okutani, N. Takayama, and Tamura Y. Openxm – an open system to integrate mathematical softwares. Available at `http://www.OpenXM.org/`.
8. A. Morgan. *Solving polynomial systems using continuation for engineering and scientific problems.* Prentice-Hall, 1987.
9. A.J. Sommese, J. Verschelde, and C.W. Wampler. Numerical irreducible decomposition using PHCpack. In M. Joswig and N. Takayama, editors, *Algebra, Geometry, and Software Systems*, pages 109–130. Springer–Verlag, 2003.
10. A.J. Sommese and C.W. Wampler. *The Numerical solution of systems of polynomials arising in engineering and science.* World Scientific Press, Singapore, 2005.
11. J. Verschelde. Algorithm 795: PHCpack: A general-purpose solver for polynomial systems by homotopy continuation. *ACM Trans. Math. Softw.*, 25(2):251–276, 1999. Software available at `http://www.math.uic.edu/~jan`.
12. J. Verschelde and Y. Wang. Computing dynamic output feedback laws. *IEEE Transactions on Automatic Control*, 49(8):1393–1397, 2004.
13. J. Verschelde and Y. Wang. Computing feedback laws for linear systems with a parallel Pieri homotopy. In Y. Yang, editor, *Proceedings of the 2004 International Conference on Parallel Processing Workshops, 15-18 August 2004, Montreal, Quebec, Canada. High Performance Scientific and Engineering Computing*, pages 222–229. IEEE Computer Society, 2004.
14. J. Verschelde and Y. Zhuang. Parallel implementation of the polyhedral homotopy method. Accepted for publication in the proceedings of *The 8th Workshop on High Performance Scientific and Engineering Computing (HPSEC-06)*, Columbus, Ohio, USA, August 18, 2006.