SI2-SSE: Solving Polynomial Systems with PHCpack and phcpy

Jan Verschelde, University of Illinois at Chicago

the mathematical problem

Solve $f(\mathbf{x}) = \mathbf{0}$, a polynomial system in several variables $\mathbf{x}$.
Some examples of algorithms and computed objects:
- evaluation and differentiation of polynomials,
- trajectories of solutions defined by parameters.

Example of an input:

```
2
  x**2 + 4*y**2 - 4;
  2*y**2 - x;
```

The polynomials define an ellipse and a parabola.
The solutions of the system are points of intersection.
How? Deform systems, starting at the solutions of

```
2
  x**2 - 1;
  y**2 - 1;
```

what kind of mathematics?

Solving polynomial systems requires many methods.
- computational algebraic geometry

What does solving a system of polynomials mean?
- discrete and computational geometry

The exponents of the monomials define a Newton polytope.
- symbolic-numeric computation

Polynomials are algebraic objects (symbolic algebra),
that we evaluate and deform (numerical analysis).
- parallel and distributed computing

Parallelism includes message passing, shared memory multithreading, and acceleration with graphics processor units.

PHCpack

PHCpack is a package for Polynomial Homotopy Continuation.
ACM Transactions on Mathematical Software achived version 1.0 (Ada 83) as Algorithm 795, vol. 25, no. 2, pages 251–276, 1999.
- most popular seems to be the *blackbox solver:*

`phc -b` computes all isolated solutions of a polynomial system.
Version 2.0 was rewritten using concepts of Ada 95
and extended with arbitrary multiprecision arithmetic.
Versions 2.1, 2.2, and 2.3 added a *numerical irreducible decomposition* for positive dimensional solution sets. Since summer of 2005, smaller and more frequent releases enable evolutionary improvements.
- Distributed under the GNU General Public License.

Public repository under version control
at `https://github.com/janverschelde/PHCpack`.
Documentation is written with the aid of Sphinx.

users and applications

Google scholar counts the citations to over four hundred.
More than eighty documented users in the scientific literature.
Who counts as a user?
- Published scientific results obtained with the software,
- obtained without assistance of the software developer (no co-author!).

Some sampling of the application areas:
- real algebraic geometry: disprove Kouchnirenko's conjecture [Haas 2002],
- mechanism design: constrained robots [Perez-McCarthy 2004],
- geomagnetism: stable states in chains of crystals [Newell 2009].

The full list is at `http://www.math.uic.edu/~jan/users.html`.

goals of the development

PHCpack prototyped polyhedral homotopies for isolated solutions
and a numerical irreducible decomposition for sets of solutions.
Some current projects:
- Littlewood-Richardson homotopies

In collaboration with Abraham Martin del Campo, Anton Leykin, Frank Sottile, Ravi Vakil, we compute linear planes that meet a given set of planes in specific ways. Implemented in a Macaulay2 package.
- methods from tropical algebraic geometry

As a continuation of joint work with Danko Adrovic,
we develop Puiseux power series to represent solution sets.
Power series are a hybrid form of symbolic-numeric computation.
- GPU acceleration of polynomial homotopies

In collaboration with Xiangcheng Yu, GPU acceleration compensates the cost of using the QD library [Hida, Li, and Bailey 2001].
- development of a web interface, we have a beta version running

at `https://kepler.math.uic.edu` (Joint with Xiangcheng Yu).

using `phcpy`

```
>>> from phcpy.solver import solve
>>> f = ['x**2*y**2 + x + y;','x*y + x + y + 1;']
>>> s = solve(f,silent=True)
>>> len(s)
4
>>> print s[0]
t :  1.00000000000000E+00 0.00000000000000E+00
m :  1
the solution for t :
x :  -1.00000000000000E+00 0.00000000000000E+00
y :  -1.61803398874989E+00 0.00000000000000E+00
== err :  9.930E-17 = rco :  4.775E-02 = res :  2.220E-16 =
```

Indicators for the quality of the solution:
- `err`: the norm of the last update made by Newton's method (forward error),
- `rco`: estimate for the inverse condition number of the Jacobian matrix,
- `res`: norm of the evaluated solution (backward error).

generators for the path trackers

What *more* can you do with phcpy scripting?

```
>>> from phcpy.solver import total_degree_start_system
>>> p = ['x**2 + 4*x**2 - 4;', '2*y**2 - x;']
>>> (q,s) = total_degree_start_system(p)
>>> from phcpy.trackers import initialize_standard_tracker
>>> from phcpy.trackers import initialize_standard_solution
>>> from phcpy.trackers import next_standard_solution
>>> initialize_standard_tracker(p,q)
>>> initialize_standard_solution(len(p),s[0])
>>> s1 = next_standard_solution()
```

The user requests the next solution point on the path and controls the pace of the path tracker. Also useful for plotting paths, for example:
`>>> points = [next_standard_solution() for i in range(11)]`
and then use `matplotlib` on the list of `points`.

parallelism and performance

Most computers are multiprocessor and multicore.
Graphic cards have surpassed ordinary CPUs in computing power.
- message passing (with Yusong Wang, Yun Guan, Anton Leykin)

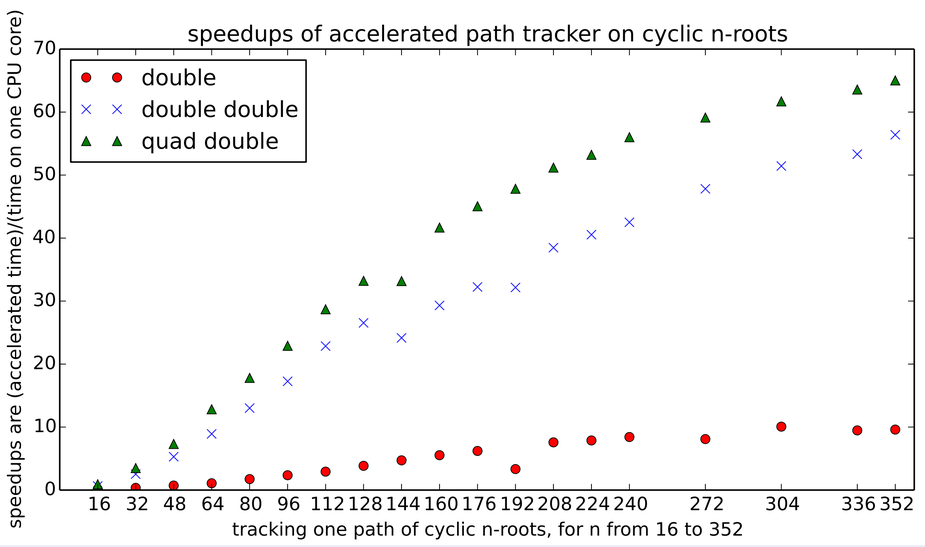
By design, the main program is written in C, responsible for the job scheduling, with the aid of MPI. The jobs execute Ada procedures.
- multicore shared memory programming (with Genady Yoffe)

The goal of this project is to offset the cost of double double and quad double arithmetic with multithreaded code.
`phc -b -t4` runs path trackers in the blackbox solver with 4 threads, using the tasking mechanism provided by Ada (multitasking can be called in phcpy).
- acceleration with graphics processors (with Xiangcheng Yu).

The code is a mix of Ada, C, and C++ CUDA.

accelerating polynomial homotopy continuation

Results on tracking one path of the cyclic n-roots benchmark problem, accelerated on the NVIDIA K20C, using GQD [Lu, He, and Luo 2010]:



Double digit speedups allow to compensate for the overhead caused by complex double double and quad double arithmetic.
Joint with Xiangcheng Yu, preprint `arXiv:1501.06625 [cs.MS]`.

interfaces

Interfaces translate inputs (polynomials) and outputs (solutions).
- with computer algebra and mathematical software systems:

Maple (with Anton Leykin), MATLAB & Octave (with Yun Guan), Macaulay2 (with Elizabeth Gross and Sonja Petrović).
- PHCpack is an optional component of Sage, the current `phc.py` is by Marshall Hampton and Alex Jokela, based on efforts of Kathy Piret and William Stein.

This material is based upon work supported by the National Science Foundation under Grant 1440534.