

## 0. Advantages and Design of PHClab

- PHCpack [2] offers no scripting language;
- Automatic input/output format conversions for systems and solutions.

PHClab is a collection of m-files which call `phc`, the executable built with PHCpack. It applies the idea of OpenXM [1], needs only executable program.

## 1. Calling the Blackbox Solver `phc -b`

Consider for example the system

$$\begin{cases} 1.3x_1^2 + 4.7x_2^2 - 3.1 + 2.3i = 0 \\ 2.1x_2^2 - 1.9x_1 = 0 \end{cases} \quad \text{with } i = \sqrt{-1}.$$

Representing the system in matrix format, we solve it via

```
t = [1.3 2 0; 4.7 0 2; -3.1 + 2.3*i 0 0; 0 0 0;
     2.1 0 2; -1.9 1 0; 0 0 0];
s = solve_system(t); % call the blackbox solver
ns = size(s,2)      % check number of solutions
s3 = s(3)           % look at the 3rd solution
```

On the screen we see:

```
ns =
     4
s3 =
      time: 1
multiplicity: 1
      err: 5.9040e-017  ——— ||Δx|| correction
      rco: 0.2770  ————— inverse condition number
      res: 1.1100e-016  ——— residual: ||f(x)||
      x1: 0.6470- 0.3876i
      x2: -0.7961+ 0.2202i
```

A solution is a structure with diagnostics and the coordinates of the solution.

## 2. Download and Installation

PHClab was tested on MATLAB 6.5 and Octave 2.1.64 on Windows and Linux machines. On an Apple laptop running Mac OS X version 10.3.7, we executed PHClab in Octave 2.1.57. PHCpack and PHClab are available at <http://www.math.uic.edu/~jan/download.html>

- download and install `phc` executable;
- download and unpack files in `PHClab.tar.gz`;
- add the name of the PHClab directory to MATLAB/Octave's search path;

The first command of PHClab one executes is `set_phcpath`.

## References

- [1] M. Maekawa, M. Noro, K. Ohara, Y. Okutani, N. Takayama, and Y. Tamura. OpenXM – an open system to integrate mathematical softwares. Available at <http://www.OpenXM.org/>.
- [2] J. Verschelde. Algorithm 795: PHCpack: A general-purpose solver for polynomial systems by homotopy continuation. *ACM Trans. Math. Softw.*, 25(2):251–276, 1999. <http://www.math.uic.edu/~jan/download.html>.

## 3. Automatic Testing and Benchmarking

The function `read_system` reads a system from file. The script

```
f = {'/tmp/Demo/ku10'
    '/tmp/Demo/cyclic5'
    '/tmp/Demo/fbrfive4'
    '/tmp/Demo/game4two'};
for k = 1:size(f,1)
    p = read_system(f{k});
    t0 = clock;
    s = solve_system(p);
    et = etime(clock(),t0);
    n = size(s,2);
    fprintf('Found %d sols for %s in %f sec.\n',n,f{k},et);
end;
```

} systems from demo database of PHCpack

produces the following statistics:

```
Found 2 sols for /tmp/Demo/ku10 in 1.819892 sec.
Found 70 sols for /tmp/Demo/cyclic5 in 11.094403 sec.
Found 36 sols for /tmp/Demo/fbrfive4 in 18.750158 sec.
Found 9 sols for /tmp/Demo/game4two in 1.630962 sec.
```

## 4. An Application: the Griffis-Duffy platform

The Griffis-Duffy platform [3] is a special Stewart-Gough platform, first analyzed in [4], it is “architecturally singular”: the figure below shows its motion.

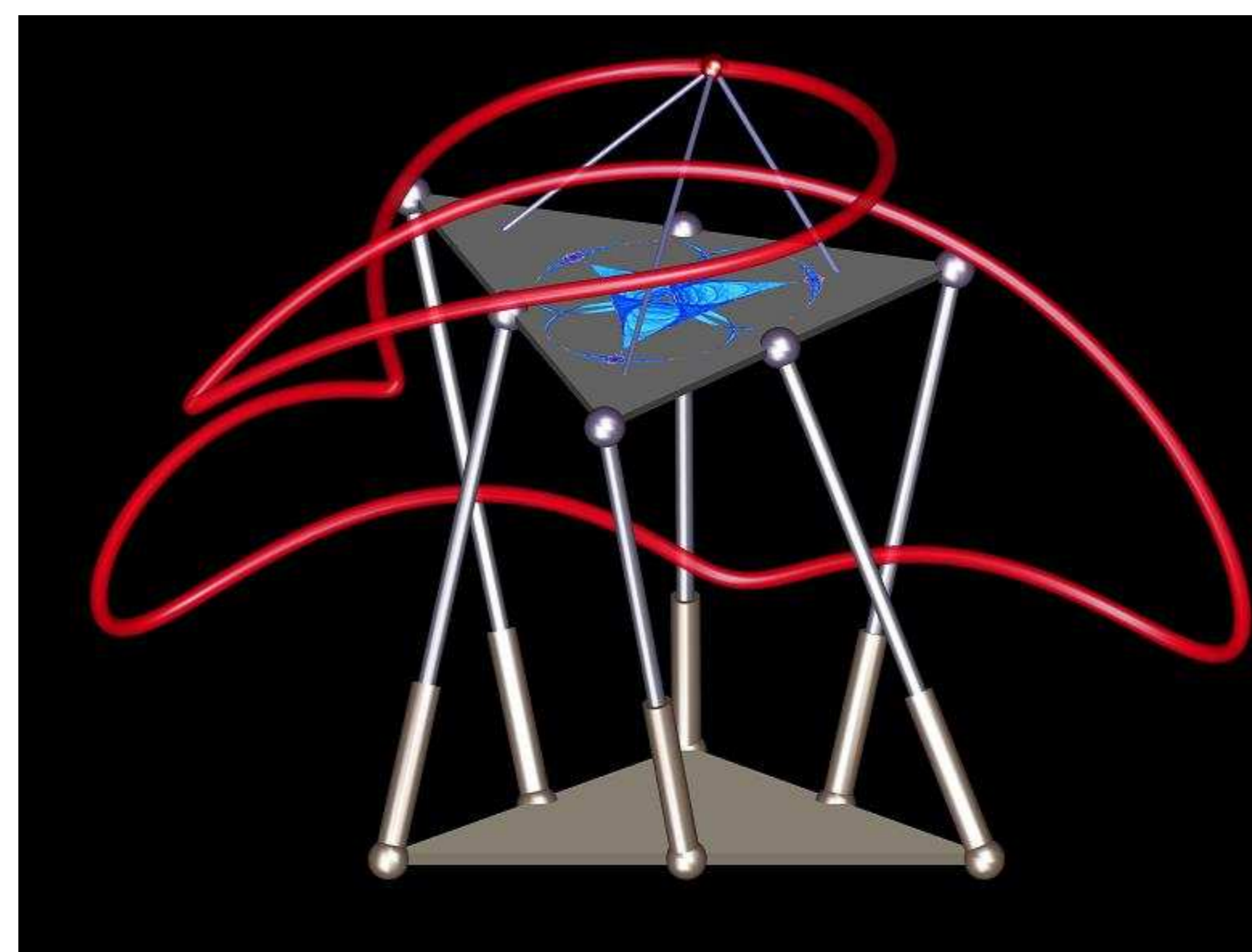


Figure 1: Image of Griffis-Duffy platform by Charles W. Wampler and Douglas N. Arnold

## References

- [3] M. Griffis and J. Duffy. Method and apparatus for controlling geometrically simple parallel mechanisms with distinctive connections. US Patent 5,179,525, 1993.
- [4] M.L. Husty and A. Karger. Self-motions of Griffis-Duffy type parallel manipulators. *Proc. 2000 IEEE Int. Conf. Robotics and Automation*, CDROM, San Francisco, CA, April 24–28, 2000.

## 5. Computing a Numerical Irreducible Decomposition

A *witness set* representing a  $k$ -dimensional solution set  $Z \subset f^{-1}(0)$  consists of

1. the system  $f$  augmented with  $k$  random hyperplanes; and
2. solutions satisfying the augmented polynomial system.

### top down computation with cascade

First we compute a numerical representation of the curve:

```
S = read_system('gdplatB');
E = embed(S,1); % embed with 1 plane
solutions = solve_system(E);
[SW,R] = cascade(E,solutions);
A witness set for the curve is in R{2} and SW{2,1}.
```

### bottom up computation: equation-by-equation

- + requires no top dimension as with cascade;
- performance depends on the order of equations.

```
p = read_system('gdplatBa'); % easy equations first
[SW,R] = eqnbyeqn(p); % solve equation by equation
returns a witness set of a curve of degree 40
```

### decomposition into irreducible factors

Taking output of either the `cascade` or `eqnbyeqn`:

```
decom = decompose(R{2},SW{2,1});
```

On return we receive 13 irreducible factors, see [5], [6], [7] for more.

### finding real witness points

If we take the slicing hyperplane to be real, we may find real witness points and use these for graphing. The instructions below use `track`:

```
start = E; % start system
E{size(E,1)} = modify_poly(E{size(E,1)});
factor = find_factor(decom) % interesting factor
for k=1:size(factor,2)
    factor(k).time = 0;
end
L = track(E,start,factor); % track paths
Among all the witness points, two of them are real.
```

## References

- [5] A.J. Sommese, J. Verschelde, and C.W. Wampler. Advances in polynomial continuation for solving problems in kinematics. *ASME Journal of Mechanical Design* 126(2):262-268, 2004.
- [6] A.J. Sommese, J. Verschelde, and C.W. Wampler. Using monodromy to decompose solution sets of polynomial systems into irreducible components. In C. Ciliberto, F. Hirzebruch, R. Miranda, and M. Teicher, editors, *Application of Algebraic Geometry to Coding Theory, Physics and Computation*, pages 297–315. Kluwer Academic Publishers, 2001. Proceedings of a NATO Conference, February 25 - March 1, 2001, Eilat, Israel.
- [7] A.J. Sommese and C.W. Wampler. *The Numerical solution of systems of polynomials arising in engineering and science*. World Scientific, 2005.