Tracking solution paths with phcpy

Konrad Kadzielawa
University of Illinois at Chicago

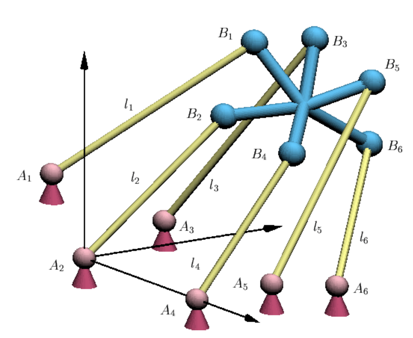
Mathematical Computing Laboratory

Introduction

Many polynomial systems arising in application from science and engineering involve several **parameters**.
It is of great interest to know **for which values of the parameters** do the solutions of the polynomial collide into multiple solutions or degenerate into positive dimensional solution sets (non isolated solutions).
The goal of the project was to develop Python scripts to heuristically explore the parameter space of polynomial systems using the *blackbox solver* feature of **PHCpack** -one specifically created to search for isolated solutions.

Motivation

The maximization of the real solutions of the systems of polynomial equations is of great interest in many practical areas, among them, design of mechanisms.
One of the motivating papers behind this project comes from P. Dietmaier from the Institut fur Mechanik, Technische Universitat Graz with the paper entitled: "The Stewart-Gough Platform of General Geometry can have 40 real postures" where he systematically shows, albeit through a different numerical scheme, that a Stewart Gough platforms (shown below) actually possesses 40 real (the only realizable) assembly modes or postures (real solutions).



Actually solving this problem involves finding all the possible numbers of real solutions with respect to the parameters' values, searching for their maximum.

Discriminant

To understand what it takes to find the maximum number of real solutions, we'll need to very briefly touch upon two concepts relevant here: the **discriminant** and the **discriminant variety**.
As an example, a simple quadratic,

$$f(x) = ax^2 + bx + c \quad (1)$$

has a discriminant of

$$\Delta = b^2 - 4ac \quad (2)$$

More generally, for a polynomial with indeterminate coefficients we may ask for a condition on the coefficients for which there are multiple roots, i.e.: where both the polynomial and its derivative vanish, implying that $\Delta(f)$ can be seen as the resultant of f and its derivative. Equivalently,

$$\Delta(f) = \begin{cases} f(x) = 0, \\ rank(J_f(x)) < n \end{cases}$$

where n = # of variables.
Lastly, given $\Delta(f)$, we define the **discriminant variety** as the solution set of the discriminant.

Two Circles Problem

Initially we considered a problem of interesting two circles in a plane - one being a unit circle, the other one being defined by its two parameters: its **radius** and **x-axis coordinate**.

$$f(x) = \begin{cases} x^2 + y^2 - 1 = 0 \\ (x-c)^2 + y^2 - r^2 = 0 \end{cases}$$

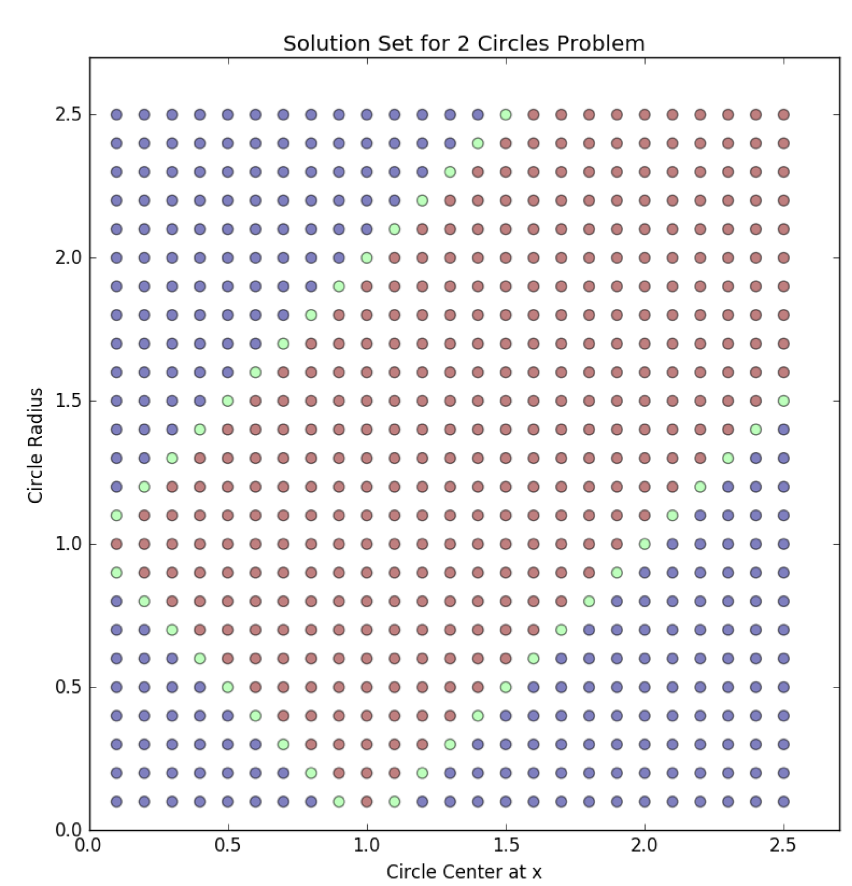
With initial and final value of radii and x-coordinates, we performed both a random walk and a systematic loop through all values of r and c within a given range, calculating the # of real solutions for each r, c combination.

Algorithm

Data: $r_i, r_f, x_i, x_f, step$ variables
Result: set of tuples (r, x, # of sols)
initialization;
for r_i *to* r_f **do**
  for x_i *to* x_f **do**
    set up the polynomial system, plug into Blackbox solver, return # of real sols;
  end
end
if *# of real sols = 1* **then**
  export to *matplotlib*, color lime;
  discriminant variety
else if *# of real sols = 2* **then**
  export to *matplotlib*, color red;
  2 real solutions
else
  export to *matplotlib*, color blue;
  0 real solutions
end

Results

All points are solutions for the given combinations of the circle's radius and its center at the x-axis, with **red** being 2, **blue** being 0 and **lime** 1 real solution(s). Lime-colored points constitute the **discriminant variety** for this system.



Four Spheres Problem

As a more advanced example of investigating the parameter space of polynomial systems, we considered the following geometric problem: **Given four spheres, how many lines are tangent to all four spheres?**
The polynomial system was generated, containing **6** equations, **6** variables and **9** parameters defining the centers and radii of 3 spheres (1 was a unit sphere).
We performed a random walk through the parameter space using two approaches: **"wiggle"** approach where we varied each parameter individually around a "good" solution and **"box minimizing"** approach where we zoned in/minimized the range of parameters upon successive runs.

Procedural Outline

1. Generate 100 lists of random parameters.
2. Obtain # of real solutions for each, pick list with highest number of solutions (initial list).
   - 2.1 Generate 18 new lists, each with single up-down change to each parameter (wiggle approach).
   - 2.2 Generate 18 new lists using new range, where range = $max - min$ of the existing parameters (box approach).
3. Compare two solutions: if initial $\leq$ new, repeat steps 3,4.
4. Otherwise, rerun the program, generate new 100 lists.

Sample Solution

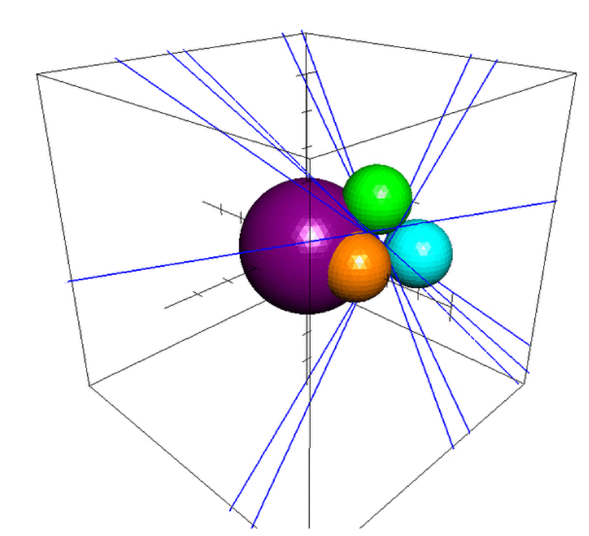


Figure: One of our 8 real tangent lines solutions.

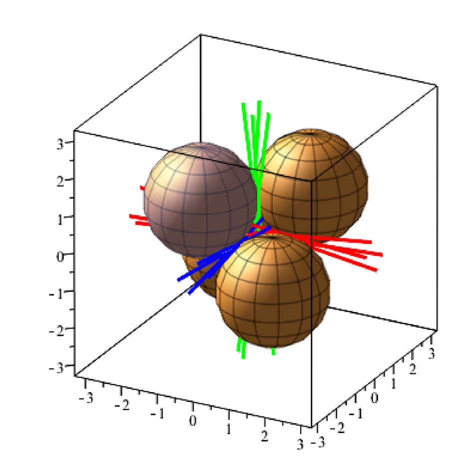
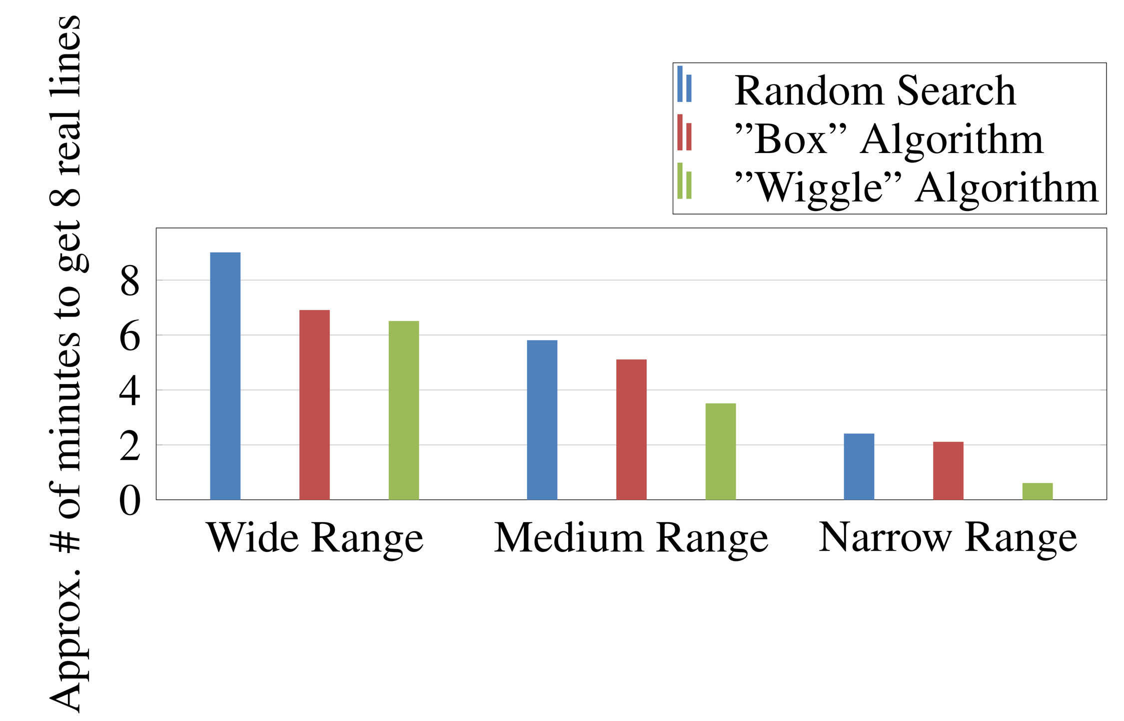


Figure: Fully real solution: 12 real tangent lines"

Results

Both of the algorithms were allowed to run for 2 hours.
The random walk through the parameter space, in both the "box minimizing" and "parameter wiggle" approaches yielded at most 8 real tangent lines, albeit the time required to find them was much smaller in the "wiggle" approach. In addition, the number of "hits" of 8 tangents lines parameter combination was overall much higher in the "parameter wiggle" approach.
Nevertheless, the heuristic approach seems unlikely to locate all 12 real tangent lines to 4 spheres in question.

Running Time Comparison



Based on 5 sets of simulations and averaging the time necessary to arrive at 8 real tangent lines to our system, we can conclude: given "wide" range of parameters, both of our algorithms fare slightly better than a purely random search.
The most interesting difference comes about at the "narrow" range for parameters around a well known solution - "wiggle" algorithm fares much better than "box" or a random search.

Conclusion

In this project, we examined the heuristic approach toward finding the maximum number of real solutions to polynomial systems using phcpy's Blackbox solver.
As two main examples we analyzed the "two circle" and "four spheres" problems. As expected, our two circle example was much easier to deal with and the parameter space has been solved. The four spheres example led to a maximum solution of 8 real tangent lines, where the complete solution has been shown to consist of 12. Nevertheless, different algorithmic approaches gave solutions in much different time frames, implying certain qualities of the parameter space.

References

D. Cox, J. Little, and D. OShea. *Ideals, Varieties and Algorithms. An Introduction to Computational Algebraic Geometry and Commutative Algebra* . Undergraduate Texts in Mathematics. SpringerVerlag, Fourth Edition, 2015.

J. Verschelde Lecture 2, Elimination Methods & Lecture 24, Newton's Method with Deflation in *MCS 563: Analytic Symbolic Computation Lecture Notes*

P. Dietmeier. "The Stewart-Gough platform of general geometry can have 40 real postures", in *Advances in Robot Kinematics: Analysis and Control*, Kluwer Academic Publishers, 1998, pp.1-10