# Sympy - Solvers: Extending `solveset`

Jogi Miglani

March 19, 2019

**Abstract**

Sympy currently is in the process of replacing `solve` with `solveset`. Many places are left to replace `solve` with `solveset`. I plan to do the following over the summer.

- Extending `transolve`: Completing lambert and handling modular equations.
- Integerating helper solvers with `solveset`.
- Enhancing the set infrastructure.
- Handle trigonometric/transcendental equations in `nonlinsolve`.

## Contents

**1 About Me** **2**

**2 The Project** **4**
2.1 Brief Overview . . . . . 4
2.2 The Plan . . . . . 5
2.2.1 Completing transolve . . . . . 5
2.2.2 Defining ImageSet Union for Trigonometric Equation Solver . . . . . 6
2.2.3 Integrating helper solvers with solveset . . . . . 7

**3 Timeline** **7**
3.1 Community Bonding Period . . . . . 8
3.2 Coding Period . . . . . 8
3.3 Post GSoC . . . . . 9

# 1 About Me

## The Student

**Name** Jogi Miglani

**e-mail** jmig5776@gmail.com

**Github** https://github.com/jmig5776

## The Institute

**University** Indian Institute of Technology (BHU), Varanasi

**Major** Mathematics and Computing

**Year** Sophomore

**Degree** Integrated Masters Degree

## Programming Experience and Mathematical Background

I have been programming from last two years, and from the past one year in python. Apart from python I have experience in C, C++, Java, Javascript. For version control, I have been using git. I work on Ubuntu 18.04.1 LTS with *vim* as my primary editor. I have background knowledge in Differential Calculus, Linear Algebra and Abstract Algebra along with Probability and Statistics. I have good experience in python as well as git version control.

Solving equations is very important in mathematics. Solving transcedental equations using Sympy is a favourite feature for me.

```
>>> from sympy.solvers.solvers import _tsolve as tsolve
>>> from sympy.abc import x
>>> tsolve(3**(2*x + 5) - 4, x)
  [-5/2 + log(2)/log(3), (-5*log(3)/2 + log(2) + I*pi)/log(3)]
```

## My Motivation

I have always been fascinated with solving algebraic equations fast and with accuracy. Then I searched for libraries which solves equations and my interests matched with Sympy. So I decided to explore this library more.

Solvers module of sympy is very powerful and no doubt it fascinates me with its techniques and algorithms. So I went through the solvers module to get deep taste of its features.

## Contributions

I started using Sympy in November and made my first contribution in beginning of December. I have been constantly contributing and learning new skills from the community.

### Merged PR's

- 15607 - Fixed `sympy.plot` sampling issue with logarithmic axis
- 15712 - Fixed matrix power failure error, it raised `ValueError` error when called with variable not known to be negative.
- 15735 - Implemented support for equations of type *f(x)**g(x) = c* to include all the solutions in the result. Earlier following types of equation does not returned all solutions :

  ```
  >>> eq = Eq((x**2 - 7*x + 11)**(x**2 - 13*x + 42), 1)
  >>> solve(eq)
  [2, 5, 6, 7]
  ```

  It did not considered the case *(-1)**(2*k)* k being an positive integer. But it does now :

  ```
  >>> eq = Eq((x**2 - 7*x + 11)**(x**2 - 13*x + 42), 1)
  >>> solve(eq)
  [2, 3, 4, 5, 6, 7]
  ```

  And many more cases are also included in real and complex range.
- 15742 - Changed CSS in docs to prefer DejaVu Sans Mono.
- 15786 - Fixed `Piecewise` to choose appropriate backend, it raised `TypeError` earlier.
- 15809 - Fixed Min/Max when no args are given to return `oo`/`-oo`.
- 15855 - Fixed `summation` to prevent infinite loop.
- 15861 - Fixed docstring of `logcombine` method.
- 15957 - Fixed `checkodesol` method and made more robust when checking solutions involving Integrals.
- 15961 - Fixed simplification of an evaluated *tan(3*pi/8)*.

**Unmerged PR's**

- 15692 - Fixed `Mul.flatten` method such that it returns the correct return type in case of matrices addition/multiplication.
- 15708 - Fixed `powsimp` to simplify further.
- 15826 - Fixed *integrate(1/(2**(2*x/3)+1), (x,0,oo))*.
- 15935 - Fixed `evalf` when expression involving symbols.

**Issues/Bugs Caught**

- 15940 - `trigsimp` or `simplify` fails to simplify this inverse trigonometric identity:

```
>>> from sympy import Symbol, acos, sqrt, atan
>>> D = Symbol('D', positive=True)
>>> e = acos(D) + 2*atan(D/(sqrt(-D + 1)*sqrt(D + 1))
- 1/sqrt(-D**2 + 1))
>>> trigsimp(e)
acos(D) + 2*atan(D/(sqrt(-D + 1)*sqrt(D + 1))
- 1/sqrt(-D**2 + 1))
>>> simplify(e)
acos(D) + 2*atan(D/(sqrt(-D + 1)*sqrt(D + 1))
- 1/sqrt(-D**2 + 1))
```

# 2 The Project

In this section I will explain my project details and its vision. I expect this structure to be considerably enhanced under the guidance of my mentor.

## 2.1 Brief Overview

One of the pretty powerful module in Sympy is Solvers. It has `solve` which is very powerful but also has some major issues as follows:

- It doesn't have a consistent output for various types of solutions It needs to return a lot of types of solutions.
- The input API is also a mess, there are a lot of parameter. Many of them are not needed and they makes it hard for the user and the developers to work on solvers.

- There are cases like finding the maxima and minima of function using critical points where it is important to know if it has returned all the solutions. `solve` does not guarantee this.

To fix the above issues `solveset` module was created and is under development. We had to replace `solve` with `solveset` completely. My main concern of this project are:

- Completing `transolve`
- Integrating helper solvers with `solveset`
- Build the set infrastructure.
- Handle system having trigonometric/transcendental equations correctly at a time by `nonlinsolve`.

## 2.2 The Plan

### 2.2.1 Completing transolve

As work done by @Yathartha22[1] in the summer of 2018 results that now transolve is fully designed and is able to handle logarithmic and exponential equations for solveset. I want to continue his work and want to complete the following :

- Equations solvable by LambertW function

  Lambert solver is almost complete by @Yathartha22[1] in 14972. Some leftovers are to pass some tests of bivariate solvers and to replace `solve` with `solveset` in `_lambert` in `bivariate`. I had opened a 16324 to complete the Lambert solver.

- Handling Modular Equations

  In 13178 one could possibly define `_invert_modular` to come up with a general fix of the issue.
  Issue:

```
>>> n = symbols('n', integer=True)
>>> a = 742938285
>>> z = 1898888478
>>> m = 2**31 - 1
>>> x = 20170816
>>> solveset(x | Mod(a**n*z, m), n, S.Integers)
```

Possible fix: Defining `_invert_modular` that will be inverting modular equations. One approach is here 14284 but it needs to be generalised and worked upon.

- Nested Trigonometric Equation Solver

  `solveset` uses `_solve_trig` to solve such trigonometric equations and focuses on returning general solutions. But still `_solve_trig` is unable to solve a variety of problems. Following are the issues that it faces and I would like to fix this during my GSoC term.

  Examples to be worked upon:

  The problem is inability to solve nested trignometric expressions 10217

```
>>> eq=cos((sin(x)+1)) - 1
>>> solveset(eq,domain=S.Reals)
ConditionSet(x, Eq(cos(sin(x) + 1) - 1, 0), (-oo, oo))
```

  Possible fix: such problems are the ones that can be converted to polynomial equations(by the principle of Decomposition and Rewriting) and thus solved. For these type of problems we can have an internal call to `solve_decomposition` which solves the problem

### 2.2.2 Defining ImageSet Union for Trigonometric Equation Solver

This issue in Sympy still prevails due to the lack of a proper algorithm for unifying two ImageSet objects such as `2*n*pi` and `2*n*pi + pi`. And many much of its use case is in trigonometric equation solving.

```
>>> solveset(sin(2*x) + sin(4*x) + sin(6*x) , x, S.Reals)
Union(ImageSet(Lambda(_n, 2*_n*pi), Integers),
ImageSet(Lambda(_n, 2*_n*pi + pi), Integers),
ImageSet(Lambda(_n, 2*_n*pi + pi/2), Integers),
ImageSet(Lambda(_n, 2*_n*pi + 3*pi/2), Integers),
ImageSet(Lambda(_n, 2*_n*pi + 4*pi/3), Integers),
ImageSet(Lambda(_n, 2*_n*pi + 2*pi/3), Integers),
ImageSet(Lambda(_n, 2*_n*pi + 5*pi/3), Integers),
ImageSet(Lambda(_n, 2*_n*pi + pi/3), Integers),
ImageSet(Lambda(_n, 2*_n*pi + 5*pi/4), Integers),
ImageSet(Lambda(_n, 2*_n*pi + 3*pi/4), Integers),
ImageSet(Lambda(_n, 2*_n*pi + 7*pi/4), Integers),
ImageSet(Lambda(_n, 2*_n*pi + pi/4), Integers))
```

```
# this should be
Union(ImageSet(Lambda(_n, _n*pi/4), Integers),
ImageSet(Lambda(_n, _n*pi + pi/6), Integers),
ImageSet(Lambda(_n, _n*pi - pi/6), Integers))

>>> solveset(sin(x), x)
Union(ImageSet(Lambda(_n, 2*_n*pi), Integers),
ImageSet(Lambda(_n, 2*_n*pi + pi), Integers))

# This should be
ImageSet(Lambda(_n, _n*pi), Integers)
```

The above problem can be solved by defining Image set Union. Various approaches are tried by contributors, some are :
@Shekharrajak[2] tried to solve this problem in these PRs 10733 10898 10713. The approach is good but it needs to be implemented in a general way which will solve the problem.

### 2.2.3 Integrating helper solvers with solveset

Here are three solvers which I will be going to integrate in solveset.

- `linsolve`: Solves a system of linear equations
- `nonlinsolve`: Solves a system of non-linear equations and improving it to handle transcedental/trigonometric equations at a time.
- `solve_decomposition`: Solves a varied class of equations using the concept of Rewriting and Decomposition(It will be already implemented while covering trigonometric equation solver.)

# 3 Timeline

This project will make `solveset` as strong as `solve` and I assure that I will devote all my time in this project and try to finish what I proposed. I will be having my semester exams till end of May, but even during this time I will try to give 2-3 hours daily and thereafter I will devote all of my time(about 40-50 hrs a week or even more).I might not be available for 1-2 days in July(travelling). College will start in August and not enough work happens during the first month so I can work with full devotion.

## 3.1 Community Bonding Period

- Discuss the project with my mentor in further detail.
- Get more acquainted with Sympy's `solvers` codebase
- Also I will have discussions and meetings with the mentors and we will come up with the efficient way

## 3.2 Coding Period

- **Week 1,2,3 (May 30 - June 19)**

  Completing `lambert`
  Implementing modular equation handler
  Implementind nested Trigonometric equation solver

- **Week 4,5 (June 20 - July 3)**

  Writing test cases
  Fixing bugs and issues that may have raised during previous implementation

- **Week 6 (July 4 - July 10)**

  Improvement of ImageSet starts
  Analyzing how ImageSet works and discussing with mentors and arriving at a plan.

- **Week 7,8(July 11 - July 24)**

  Work on ImageSet Union, make it efficient and try to merge it.
  Adding test cases and minor bug fixes

- **Week 9,10,11(July 25 - Aug 14)**

  Integrating `linsolve` with solveset.
  Integrating `nonlinsolve` with solveset
  Improving `nonlinsolve` to handle transcedental/trigonometric equations at a time.
  Writing tests and fixing minor bugs/issues.

- **Week 12,13(Aug 15 - Aug 29)**

  Finishing above work if it is remaining.
  Replace all internal calls from `solve` to `solveset`
  Analyzing what all things that can be to improve trigonometric equation solver to make it more better.

### 3.3 Post GSoC

If there are things that are left unimplemented I will try to complete post GSoC and even I will also continue my contributions to sympy and help the new contributors.Sympy provides me with a good platform to hone my programming skills. I would love to contribute more to Sympy and its solvers module to make it more powerful and robust.

## References


[1] Yathartha Joshi
https://github.com/Yathartha22

[2] Shekhar Prasad Rajak
https://github.com/Shekharrajak