A small library for solving real quadratic equations symbolically

Group N

Abstract

We present `quadratic`, a small Python library for solving the real quadratic equation $ax^2 + bx + c = 0$. The library exposes four short functions: the discriminant, the set of real solutions, the number of real roots, and the coordinates of the vertex of the corresponding parabola. The roots are returned exactly using `sympy.solveset` [2]; the documentation follows the Diataxis framework [3] and the conventions of the open-source textbook *Python for Mathematics* [1].

1 Introduction

The real quadratic equation $ax^2 + bx + c = 0$ is the first non-trivial equation that mathematics students meet at school and one of the few for which a closed-form solution is taught directly. The solutions are given by the quadratic formula

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a},$$

and the nature of the roots depends on the sign of the discriminant $\Delta = b^2 - 4ac$. When $\Delta > 0$ there are two distinct real roots; when $\Delta = 0$ the two roots coincide; and when $\Delta < 0$ there are no real roots.

Several Python libraries can solve quadratics. The `numpy.roots` function works with numeric inputs and returns floating-point approximations, which is convenient but loses the exact form of the answer. The standard `math` library does not handle quadratics at all. We take a different approach. By calling `sympy.solveset` [2] with the domain restricted to `sympy.S.Reals` we get an exact `sympy.Set` of real roots, which is also the recommended interface in the `sympy` documentation.

We present a four-function library, `quadratic`, that returns exact real solutions, the discriminant, the number of real roots, and the vertex of the parabola. The library is small but covers the three discriminant cases in its tests.

2 The library

The library lives in the single module `quadratic.py` and follows the conventions of the algebra chapter of *Python for Mathematics* [1]: every input is wrapped in `sympy.S(...)` before any arithmetic, so that an integer input is automatically promoted to `sympy.Integer` and an expression input passes through unchanged.

`discriminant(quadratic_coefficient, linear_coefficient, constant_coefficient)` returns $b^2 - 4ac$.

`solutions(quadratic_coefficient, linear_coefficient, constant_coefficient, variable)` returns the set of real roots. We pass the constructed expression to `sympy.solveset` with `domain=sympy.S.Reals`; this returns an empty set when $\Delta < 0$, a one-element set when $\Delta = 0$ and a two-element set otherwise.

`number_of_real_roots(quadratic_coefficient, linear_coefficient, constant_coefficient)` returns 0, 1, or 2 according to the sign of the discriminant.

`vertex(quadratic_coefficient, linear_coefficient, constant_coefficient)` returns the coordinates $(-b/(2a), c - b^2/(4a))$ of the vertex of the parabola $y = ax^2 + bx + c$.

3 Worked examples

For $x^2 - 3x + 2 = 0$ the discriminant is $1 > 0$, so we expect two real roots. The library returns $\{1, 2\}$, which we verified by hand.

For $x^2 - 2x + 1 = 0$ the discriminant is zero. The library returns the singleton $\{1\}$, and the vertex of $y = (x - 1)^2$ is at $(1, 0)$, as expected.

For $x^2 + 1 = 0$ the discriminant is $-4 < 0$. The library returns the empty set; the vertex of $y = x^2 + 1$ is at $(0, 1)$, above the x-axis, which is consistent with the parabola not crossing it.

4 Discussion

The `sympy` documentation recommends `solveset` over the older `solve` because it returns a `Set`, which gives a uniform way to talk about the empty case (no real roots) and the singleton case (repeated root). Working with sets is also easier when the caller wants to combine the result with set operations such as union or membership.

The library handles only quadratics. Cubic and higher-degree polynomials would need a different formula (cubics) or no closed form at all (degree five and above by Abel-Ruffini). A natural extension would be to fall back to `sympy.solveset` for higher-degree polynomials and expose the same interface.

5 Conclusion

We have presented `quadratic`, a small library for solving real quadratic equations symbolically. The library returns exact roots, the number of real roots, the discriminant, and the vertex of the parabola.

References

[1] Vincent Knight. Python for Mathematics, 2024.

[2] Aaron Meurer, Christopher P. Smith, Mateusz Paprocki, Ondřej Čertík, Sergey B. Kirpichev, Matthew Rocklin, AmiT Kumar, Sergiu Ivanov, Jason K. Moore, Sartaj Singh, et al. SymPy: symbolic computing in Python. *PeerJ Computer Science*, 3:e103, 2017.

[3] Daniele Procida. Diátaxis: a systematic framework for technical documentation authoring, 2017.