

Extending GeoGebra/realgeom with QEPCAD B to obtain proofs on geometric inequalities

Zoltán Kovács¹, Róbert Vajda²

¹The Private University College of Education of the Diocese of Linz, Salesianumweg 3, Linz, 4020, Austria

²Bolyai Institute, University of Szeged, Aradi vértanúk tere 1., Szeged, 6720, Hungary 6720

Abstract

We report on our recent improvements of the GeoGebra/realgeom system that is able to symbolically compare geometric quantities of a given planar geometric figure. QEPCAD B is connected as an external tool now to solve the geometric problem that was translated into an algebraic form first, and by using real quantifier elimination, a generally true/false answer can be given. Our implementation allows free access to this toolset, not just for researchers but teachers and students too, on all major platforms including Windows, Mac and Linux.

Keywords

automated geometry proving, dynamic geometry software, Euclidean planar geometry, GeoGebra, geometric inequality, QEPCAD B, mathematical software, real quantifier elimination

1. Introduction

In this paper we present a free software system that is able to prove inequalities in planar Euclidean geometry by providing a graphical user interface. We introduce a combination of four software components: *GeoGebra* [1], *Giac* [2], *realgeom* [3] and *QEPCAD B* [4, 5]. The novelty we show in this paper is the insertion of the fourth component in the computation process. We already published a paper [6] on combining the three components with the proprietary software *Mathematica* [7]. Our system, called *GeoGebra Discovery* [8] is available online at <https://github.com/kovzol/geogebra-discovery> (we refer to its version 2021Mar13).

Fig. 1 shows the network layout of our system. Some components are optional: *Maple* (and its underlying subsystems), *Mathematica* and *Reduce* (with its underlying subsystem) can serve just benchmarking purposes and have no relevance in our current focus in this paper.

It is very important that all backends listed in Fig. 1 provide cylindrical algebraic decomposition (CAD) with real quantifier elimination (RQE) that are crucial to solve inequalities in our problem setting. Roughly speaking, given a set S of polynomials in \mathbb{R}^n , a CAD is a decomposition of \mathbb{R}^n into connected semialgebraic sets on which each polynomial of S has constant sign. CAD can provide effective RQE, that is, a simplification of a quantified formula to another one that does not contain any quantifiers. See [9] for further details.

CICM-WS 2021

✉ zoltan@geogebra.org (Z. Kovács); vajdar@math.u-szeged.hu (R. Vajda)

🌐 <https://matek.hu/zoltan> (Z. Kovács); <http://www.math.u-szeged.hu/~vajda/indexen.htm> (R. Vajda)

🆔 0000-0003-2512-5793 (Z. Kovács); 0000-0002-2439-6949 (R. Vajda)



© 2021 Copyright for this paper by its authors. Use permitted under Creative Commons Licence Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings (CEUR-WS.org)

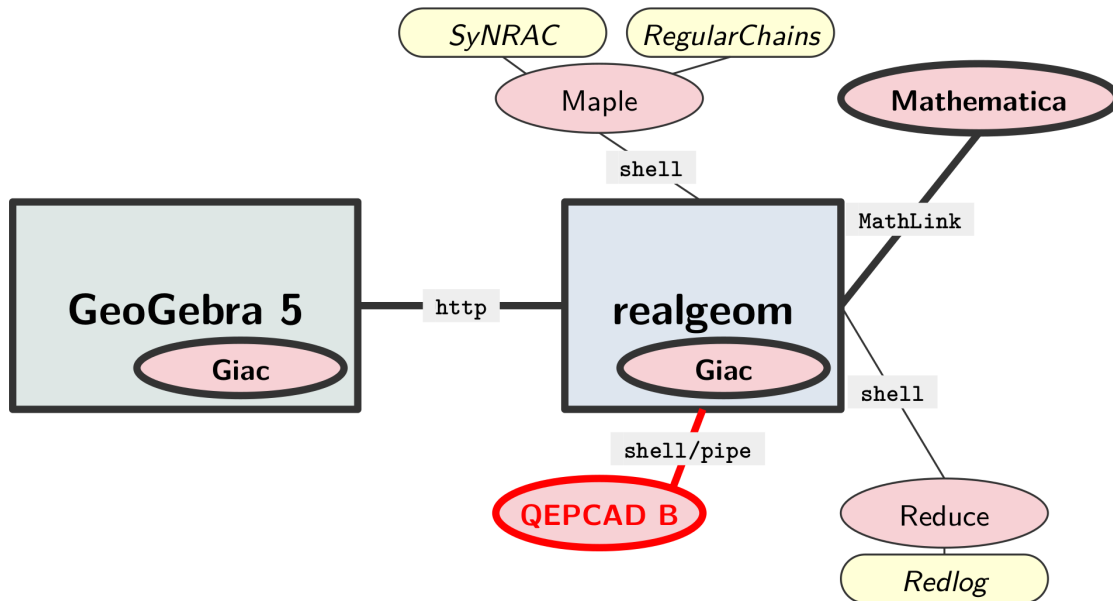



Figure 1: Computation hierarchy

2. An example: Generalizing Pythagoras' Theorem

Let a , b and c denote the lengths of sides in a right triangle, where c is the longest one. According to Pythagoras, it is well known that $a^2 + b^2 = c^2$. As a school exercise, one can ask the question what happens if we omit the assumption of having a right triangle, and no condition is given to the order of lengths a , b and c , either. By using our tool it can be quickly shown mechanically that in general the equality $a^2 + b^2 > \frac{c^2}{2}$ holds. We use only free software components in our current contribution to achieve this result. This allows a large number of users to make their own experiments on studying similar (but eventually more difficult) questions, including high school students who may have difficulties to access commercial software.

Fig. 2 shows how a typical screen layout of GeoGebra Discovery looks like, and, in particular, how the above example is solved for the user. Four windows can be seen: GeoGebra (Classic 5, a Java desktop application) on the right, a black one in the background shows the log messages of GeoGebra, the blue one corresponds to the component `realgeom` by showing its log information, and, in the foreground, GeoGebra communicates the result of comparison of the input quantities.

Here we explain in a nutshell how the user obtains the required output. First, the application suite is started by running a batch program. It starts the component `realgeom` that launches `QEPCAD B` in the background. After this step, GeoGebra's user interface is started. In GeoGebra's application window the user needs to draw a triangle by using the Polygon  tool. (This step can also be performed with other built-in geometric tools.) Next, in the Input Bar the command `Relation($a^2 + b^2, c^2$)` is to be typed. The `Relation` command looks for equality of

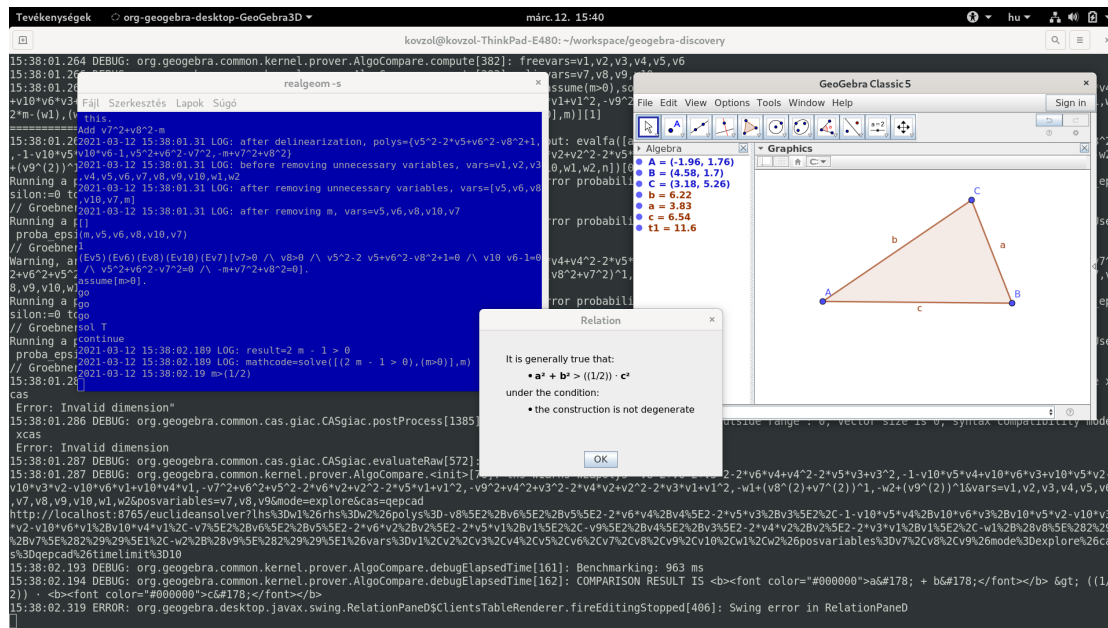


Figure 2: A typical screenshot of GeoGebra Discovery

the two expressions, but here we are also interested if they are eventually multiples in general, independently of the input points. Thus, GeoGebra informs the user that the quantities $a^2 + b^2$ and c^2 differ, but after clicking the button *More...* a symbolic preparation is started. After sending the outsourced data to realgeom/QEPCAD, the answer is obtained below 1 second, and communicated as shown in Fig. 2.

In fact, there are several mathematical and technical steps performed in the background. Most of them are completely invisible for the end user. First of all, the geometric setup is translated into a set of algebraic equations and inequalities: this step is performed inside GeoGebra. Then, another internal step is made: the Giac computer algebra system (CAS) checks whether there is a constant $m > 0$ such that

$$a^2 + b^2 = m \cdot c^2, \quad (1)$$

by using elimination via Gröbner bases (see, for example, [10] for more information on the elimination technique). In our case, there is no such constant, so the realgeom system is invoked via a call through `ht tp` as a next step.

The realgeom software first simplifies the equation system part of the input by using some sophisticated Giac code, including specialization of the input coordinates and delinearizing the system of equations. Then it uses a pipe to send the problem description to QEPCAD (as fallback a shell connection is also available), which, for the current setup, corresponds to the formula

$$\begin{aligned}
& (\exists v_5)(\exists v_6)(\exists v_8)(\exists v_{10})(\exists v_7) (v_7 > 0 \wedge v_8 > 0 \\
& \wedge v_5^2 - 2v_5 + v_6^2 - v_8^2 + 1 = 0 \wedge v_{10}v_6 - 1 = 0 \\
& \wedge v_5^2 + v_6^2 - v_7^2 = 0 \wedge -m + v_7^2 + v_8^2 = 0)
\end{aligned} \tag{2}$$

under the assumption $m > 0$. Here we quickly explain the quantified formula (2). The realgeom system assumes that points A and B are fixed to $(0, 0)$ and $(1, 0)$, respectively. This can be done without loss of generality. The variables v_5, v_6 stand for the coordinates of point C , while v_7 and v_8 describe the lengths of sides a and b . The remaining equations describe the geometric setup. They are obtained in an automated way in GeoGebra, by translating the user's input into an algebraic system. The existentially quantified formula is assembled finally by the realgeom system.

After retrieving the computed data, for this setup, namely the quantifier-free formula

$$2m - 1 > 0, \tag{3}$$

realgeom rewrites QEPCAD's output in a format that is usable by GeoGebra:

$$m > \frac{1}{2}, \tag{4}$$

here, again, Giac is used. Finally, GeoGebra is formatting the obtained data into a final output form and displays the inequality

$$a^2 + b^2 > \frac{c^2}{2}. \tag{5}$$

3. Results








We created a database of 113 test cases for various triangles and some other polygons, to be able to compare the various backends that are already available in realgeom. A major part of the test cases is based on [11]: they are comparisons of various expressions of quantities in a triangle.

Currently 5 different computation methods can be tested:

1. **Classic5-q**: GeoGebra (Java) first tries to use elimination to solve a problem. If it is not successful or the answer is ambiguous, realgeom is called with the QEPCAD backend.
2. **Classic5-m**: GeoGebra (Java) first tries to use elimination to solve a problem. If it is not successful or the answer is ambiguous, realgeom is called with the Mathematica backend.
3. **Classic5-rg-q**: GeoGebra (Java) immediately calls realgeom with the QEPCAD backend.
4. **Classic5-rg-m**: GeoGebra (Java) immediately calls realgeom with the Mathematica backend.
5. **Classic6**: GeoGebra (JavaScript) uses elimination to solve a problem.

By default, the first method is used. We note that the web version of GeoGebra (Classic 6) has no support to solve inequality systems with quantifier elimination.

Table 1
Steps to take to get the result

No.	Name	Toolbar Icon	Description	Value
1	Point C			$C = (0, 2)$
2	Point B			$B = (6, 1)$
3	Segment a		Segment C, B	$a = 6.08$
4	Line g		Line through C perpendicular to a	$g : -6x + y = 2$
5	Point A		Point on g	$A = (0.33, 3.99)$
6	Segment b		Segment C, A	$b = 2.02$
7	Segment c		Segment A, B	$c = 6.41$
8	Text $text1$	ABC		“Comparison of the Sides of a Right Triangle via realgeom”
9	Text $text2$	ABC	Compare($a + b, c$)	“ $c < (a + b) \leq ((\sqrt{2}) \cdot c)$ ”

In our benchmarking database we set a time limit of 30 seconds for each test. According to our recent results at https://prover-test.geogebra.org/job/GeoGebra_Discovery-comparetest/73/artifact/fork/geogebra/test/scripts/benchmark/compare/html/all.html (as of 15 March 2021) Mathematica *far* outperforms QEPCAD. From 113 test cases Mathematica successfully solves 107, the rest time out (except one that crashes in GeoGebra due to exhaustion of resources, before continuing with any external computations). QEPCAD shows a lower performance, but still 41 tests are successfully processed: 40 of them below 5 seconds and 31 below 1 second. (By raising the time limit to 20 minutes we cannot get significantly better results for QEPCAD, either.)

Acceptance of a test result is currently decided in the following way: If *any* answer is given by the backend, then the result is accepted. For such a small amount of test cases a quick human verification can actually double-check if the outputs are indeed correct. However, some outputs can indeed differ significantly, even if they are equivalent and correct. For example, in the test case `IsoTriangle-Perimeter_CircumRadius_a` (it is about comparing the perimeter p and the circumradius R of an arbitrary but isosceles triangle) Mathematica reports the inequality $R \geq (1/(3 \cdot \sqrt{3})) \cdot p$, while QEPCAD B communicates $R \geq ((\sqrt{3}/9)) \cdot p$.

We show one of these tests in detail, namely, the test case `RightTriangle-Bottema11.20` which is taken from [11].

Tab. 1 shows which steps in GeoGebra need to be made to achieve the result as shown in Fig. 3. This result differs a bit from the one in Fig. 2: here we used the `Compare` low level command to immediately get the two inequalities, instead of using the high level interface of the `Relation` command. Also, we learn that the sum $a + b$ is actually bounded by c and $\sqrt{2} \cdot c$.

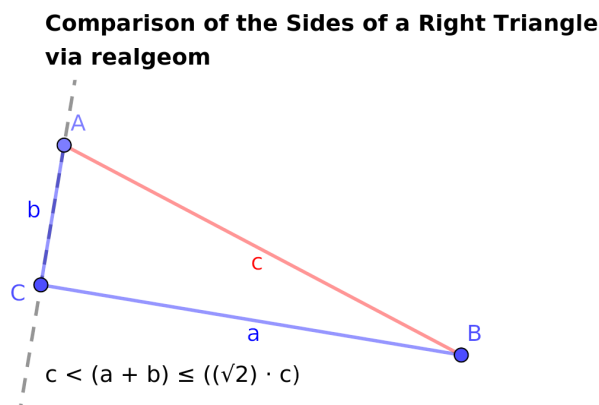


Figure 3: Screenshot of benchmark RightTriangle-Bottema11.20

In fact, our system always expects to find sharp positive constants m and M such that

$$m \cdot Q_1 \underset{(<)}{=} Q_2 \underset{(<)}{=} M \cdot Q_1 \tag{6}$$

where Q_1 and Q_2 can be arbitrary homogeneous expressions of quantities of the same degree. In our implementation these quantities must always be distances between two points of the planar construction. In Fig. 3 the constants $m = 1$ and $M = \sqrt{2}$ are obtained, while the first inequation is sharp, but the second one allows equality as well.

By comparing the performance of Mathematica and QEPCAD here we see that both systems are very fast. They deliver the solution far below 1 second. This also confirms that realgeom/QEPCAD can be used to obtain non-trivial results, by allowing a large scale of users to study inequalities in a planar Euclidean geometry construction.

We provide another figure that shows the density estimate of benchmark time output visualized with statistics software R's sm package [12] on those 41 tests that uniformly work on both systems (see Fig. 4). The x -axis shows the time in milliseconds. According to this diagram, Mathematica's vantage is clear because of its smaller deviation, however, QEPCAD's performance is also remarkable.

4. Technical considerations and further work

Our aim was to reach as much potential users as possible. So we created a version of GeoGebra Discovery for each major platform, including Windows 10 (32-bit), Mac OS and Linux (64-bit). We used the repositories [13] and [14] to compile QEPCAD B 1.74 on Mac and Linux, and for Windows we used an own build of Petter Strandmark's QEPCAD B 1.69 fork from GitHub. For the Mac and Linux versions, for better portability, we compiled QEPCAD statically and released all components as a single .zip bundle. (On Mac we also provided a Java Runtime Environment in the package.) The bundles can be downloaded at <https://github.com/kovzol/geogebra/releases/tag/v5.0.591.0-2021Mar13>.

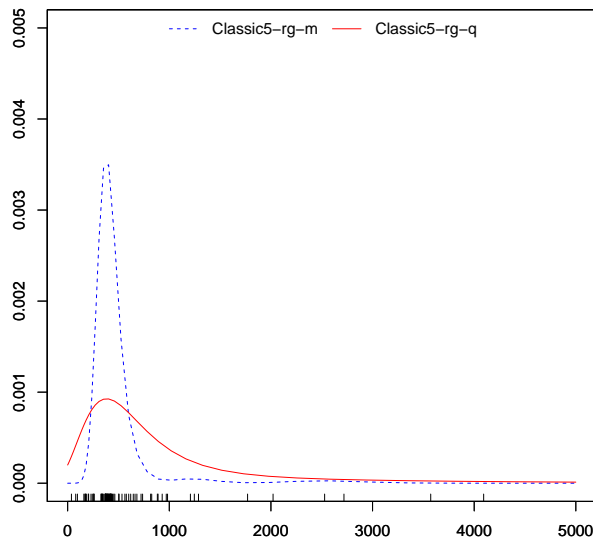


Figure 4: Comparison of Mathematica and QEPCAD

Our final aim is to embed our experimental network of the four systems into one tool, namely, into the *mainstream version* of GeoGebra, developed at University of Linz, Austria. It runs on several platforms, including desktop computers with various operating systems (Windows, Mac and Linux), and the web platform, and also smartphones. This high variety of technological background requires heavy simplification of the underlying technologies and it calls for minimizing the applied external tools.

That is, for the long term our aim is to embed all external parts of our machinery into GeoGebra. For the realgeom part, this seems to be achievable since it is a single Java application with no graphical user interface: only some sophisticated algorithms are included that can be copied to GeoGebra.

On the other hand, copying QEPCAD B into GeoGebra seems to be somewhat more difficult. QEPCAD is based on Saclib [13], a C library that offers CAS computations. QEPCAD is written in C++. Thus, it seems a viable way to compile both systems as a Java Native Interface (JNI) and include them as a dynamically loadable object on all desktop platforms. The GeoGebra Team has already quite a lot of experience on this technological step because Giac, also written in C++, is embedded in the same way in GeoGebra since 2013 [2]. In that case SWIG (<http://www.swig.org/>) was used to perform the step from C++ to Java.

Besides the native platform, a web application and a smartphone version should also be supported. Compilation of C/C++ applications for the web is already well supported by various tools. Here we highlight Emscripten [15] that provides multiple targets including JavaScript or WebAssembly. For GeoGebra's former developments this tool is already known: it is continu-

ously used to compile Giac as an embedded system for the web/smartphone platforms. On the other hand, Giac's "online" version has some limitations at the moment: it does not support allocating a high amount of memory, and its speed significantly underperforms the JNI version. There is, however, still hope to make substantial improvements in this direction by using the most recent versions of Emscripten.

We remark that Saclib has a built-in garbage collector (GC) that needs to be initialized in the beginning of a QEPCAD session, by setting the maximal value of the used cells. This number cannot be increased during a QEPCAD session, and this issue results in some limitations in the current version of GeoGebra Discovery. As a possible future improvement, dynamic increase of the GC space could be a solution for this problem.

A more serious problem is that QEPCAD does not optimize the input. This results in well-known speed issues, even on very simple inputs. To solve this problem, we could use the *Tarski* system [16, 17]: it can preprocess the input planned to be processed with QEPCAD, and it usually suggests a better input form for the user. By following this concept, we also consider adding Tarski as a preprocessor between realgeom and QEPCAD. We expect a substantial speedup after this step.

In fact, the Saclib-QEPCAD machinery is more a piece of research software than an industrial application. It was evolutionary software during the last decades, but as of today, its technological background is somewhat outdated. Therefore, we also consider trying other software as underlying backends. As of writing of this paper, the SMT-RAT [18] system seems to be a more up-to-date framework that could be a potential long-term backend to support symbolic proofs on geometric inequalities in GeoGebra.

Even if the above mentioned technical difficulties can be solved, speed remains an issue for more complicated inputs. This also means that studying inequalities in planar Euclidean geometry by automatically obtained proofs will be still a challenging research field in the following years as well.

5. Conclusion

We presented our recent improvements on GeoGebra Discovery that combine various software tools to automatically obtain sharp constants in inequalities like (6). Our first impressions with its new backend QEPCAD B are very positive and we find this system to be suitable in many levels of studying Euclidean geometry. There is, however, still much room for further development.

One issue is the complexity of our system: it prevents its easy embedding into the mainstream version of GeoGebra. The second issue is speed: as it is well known, CAD has doubly exponential complexity in the number of variables (see [19, 20]), so in general there is no hope to cover all problems in feasible time. But the questions we raise are quite specific, so there seems to be much room for remarkable speedups.

For some possible optimization ideas we refer to our recent paper [6], but here we also highlight that as future work we plan to measure the difficulty of the input problems. According to the number of free parameters, equations and dimension (that is, degree of freedom), for example. By having a detailed set of data we may be able to observe how the geometric problem

setting relates to the types of algebraic objects.

In the class of isosceles and right triangles we observed that the related translated algebraic problems are one-parameter problems, whose solutions can be obtained by adapted parametric real root counting/finding methods as well. This shows a potential speedup for these type of problems.

Also, we would like to extend the benchmark suite with some problems for quadrilaterals of special types, among other inputs.

Acknowledgments

First author was partially supported by the grant PID2020-113192GB-I00 from the Spanish MICINN. The second author was supported by the EU-funded Hungarian grant EFOP-3.6.2-16-2017-00015.

We are thankful to Christopher W. Brown for his very useful advices on improving our QEPCAD program codes, and to have a well-optimized version of the QEPCAD binary.

Also, we thank the three anonymous reviewers for recommending several changes on a preliminary version of this paper.

References

- [1] M. Hohenwarter, M. Borchers, G. Ancsin, B. Bencze, M. Blossier, A. Delobelle, C. Denizet, J. Éliás, A. Fekete, L. Gál, Z. Konečný, Z. Kovács, S. Lizelfelner, B. Parisse, G. Sturr, GeoGebra 5, 2014. <http://www.geogebra.org>.
- [2] Z. Kovács, B. Parisse, Giac and GeoGebra – improved Gröbner basis computations, in: J. Gutierrez, J. Schicho, M. Weimann (Eds.), Computer Algebra and Polynomials, Lecture Notes in Computer Science, Springer, 2015, pp. 126–138. URL: http://dx.doi.org/10.1007/978-3-319-15081-9_7.
- [3] R. Vajda, Z. Kovács, *realgeom*, a tool to solve problems in real geometry, A GitHub project, 2018. <https://github.com/kovzol/realgeom>.
- [4] G. E. Collins, H. Hong, Partial cylindrical algebraic decomposition for quantifier elimination, *Journal of Symbolic Computation* 12 (1991) 299–328.
- [5] C. W. Brown, An overview of QEPCAD B: a tool for real quantifier elimination and formula simplification, *Journal of Japan Society for Symbolic and Algebraic Computation* 10 (2003) 13–22.
- [6] R. Vajda, Z. Kovács, GeoGebra and the *realgeom* reasoning tool, CEUR Workshop Proceedings (2020) 204–219. URL: <https://doi.org/urn:nbn:de:0074-2752-0>. arXiv:<http://ceur-ws.org/Vol-2752/paper15.pdf>.
- [7] Wolfram Research, Inc., Mathematica, version 12.1, 2020. Champaign, IL.
- [8] Z. Kovács, GeoGebra Discovery, A GitHub project, 2020. <https://github.com/kovzol/geogebra-discovery>.
- [9] T. Sturm, V. Weispfenning, Computational geometry problems in REDLOG, in: International Workshop on Automated Deduction in Geometry. LNCS, vol. 1360, Springer, Berlin, Heidelberg, 1996.

- [10] T. Recio, M. P. Vélez, Automatic discovery of theorems in elementary geometry, *Journal of Automated Reasoning* 23 (1999) 63–82.
- [11] O. Bottema, R. Djordjevic, R. Janic, D. Mitrović, P. Vasić, *Geometric Inequalities*, Wolters-Noordhoff Publishing, Groningen, 1969.
- [12] A. W. Bowman, A. Azzalini, Package `sm`: non-parametric smoothing methods (version 2.2-5), 2013. URL: <http://www.stats.gla.ac.uk/~adrian/sm>, http://azzalini.stat.unipd.it/Book_sm/index.html.
- [13] C. W. Brown, `SacliB`, A GitHub project, 2021. <https://github.com/chriswestbrown/sacliB>.
- [14] C. W. Brown, `QEPCAD`, A GitHub project, 2021. <https://github.com/chriswestbrown/qepcad>.
- [15] A. Zakai, `Emscripten`: An LLVM-to-JavaScript compiler (2013). <https://github.com/kripken/emscripten/blob/master/docs/paper.pdf?raw=true>.
- [16] F. Vale-Enriquez, C. Brown, Polynomial constraints and unsat cores in `TARSKI`, in: *Mathematical Software – ICMS 2018*. LNCS, vol. 10931, Springer, Cham, 2018, pp. 466–474.
- [17] C. W. Brown, F. Vale-Enriquez, `Tarski`, A GitHub project, 2018. <https://github.com/chriswestbrown/tarski>.
- [18] F. Corzilius, G. Kremer, S. Junges, S. Schupp, E. Abraham, `SMT-RAT`: an open source C++ toolbox for strategic and parallel SMT solving, in: *International Conference on Theory and Applications of Satisfiability Testing*, LNCS, vol. 9340, Springer, 2015, pp. 360–368.
- [19] C. Brown, J. Davenport, The complexity of quantifier elimination and cylindrical algebraic decomposition, in: *Proceedings of ISSAC '07*, ACM, 2007, pp. 54–60.
- [20] J. Davenport, J. Heintz, Real quantifier elimination is doubly exponential, *Journal of Symbolic Computation* 5 (1988) 29–35.