Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

MoebInv: C++ libraries for manipulations in non-Euclidean geometry

Vladimir V. Kisil

School of Mathematics, University of Leeds, Leeds LS2 9JT, United Kingdom of Great Britain and Northern Ireland

ARTICLE INFO

Article history:
Received 18 August 2019
Received in revised form 7 December 2019
Accepted 10 December 2019

MSC:
68U05
51B25
68W30
51N25
51B10
11E88

Keywords:
Lie spheres geometry
Möbius transformations
Fractional linear transformations
Symbolic computations

ABSTRACT

The introduced package MoebInv contains two C++ libraries for symbolic, numeric and graphical manipulations in non-Euclidean geometry. The first library `cycle` implements basic geometric operations on cycles, which are the zero sets of certain polynomials of degree two. The second library `figure` operates on ensembles of cycles interconnected by Moebius-invariant relations: orthogonality, tangency, etc. Both libraries work in spaces with any dimension and arbitrary signatures of their metrics. Their essential functionality is accessible in interactive modes from Python/Jupyter shells and a dedicated Graphical User Interface. The latter does not require any coding skills and can be used in education. The package is tested on (and supplied for) various Linux distributions, Windows 10, Mac OS X and several cloud services.

Code metadata

Current code version	v3.5.7
Permanent link to code/repository used of this code version	https://github.com/ElsevierSoftwareX/SOFTX_2019_248
Legal Code License	GPL v3.0
Code versioning system used	git
Software code languages, tools, and services used	C++, Python
Compilation requirements, operating environments & dependencies	Compiled for Linux, Win32, Mac OS X using GNUg++ or clang.
If available Link to developer documentation/manual	https://sourceforge.net/projects/moebinv/files/docs/figure.pdf
Support email for questions	kisilv@maths.leeds.ac.uk V.Kisil@leeds.ac.uk

1. Introduction

We present the Open Source package MoebInv [1,2]—a research and educational tool for various geometric setups. Its domain, design and functionality have some unique features which are not available elsewhere. The code is symbiotically growing together with the research in the extended Möbius–Lie geometry [3], both – the code and the theory – benefited from this interaction. Functionality of the package is accessible from a C++ code and can be interactively used through Python/Jupyter shells and a dedicated Graphical User Interface (GUI).

There is already a collection of well-established and reputable Open Source geometry software (GeoGebra [4], CaRMetal [5], Kig [6], Dr. Geo [7]) as well as commercial educational packages (The Geometer's Sketchpad, Cabri, Cinderella, NetPad). All of those are designed to work primary with the Euclidean geometry—the oldest archetypal mathematical theory. Nowadays it is the first from a large family of various geometries: affine, projective, conformal, Riemannian, etc. According to F. Klein's *Erlangen programme* (which was influenced by S. Lie), a geometry studies invariant properties under a certain transitive group action.

The package MoebInv works with invariants of fractional linear transformations (FLT) which contain Möbius maps as an important subset. The implementation admits spaces of any (including symbolic) dimensionality and arbitrary signatures of metric (covering the degenerate cases). The associated geometries span

E-mail address: kisilv@maths.leeds.ac.uk.

https://doi.org/10.1016/j.softx.2019.100385

a wide domain, which includes conformal [8, Ch. 9], hypercomplex [9] and Lie sphere geometries [10–12]. For these fields the package facilitates:

- Experiments and research.
- Automated theorem proving and symbolic calculation.
- Visualisation and interactive manipulations.
- Exact arithmetic and high precision numeric evaluation.
- Programming extensions of functionality.
- Educational usage (starting from school and college levels) which does not require any coding skills.

For sake of simplicity we illustrate this paper by the fundamental and most visual case of fractional linear transformations (FLT) of the complex plane defined by 2×2 complex matrices [8, Ch. 9]:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} : z \mapsto \frac{az+b}{cz+d}, \qquad \text{where } z \in \mathbb{C} \text{ and } \det \begin{pmatrix} a & b \\ c & d \end{pmatrix} \neq 0. \tag{1}$$

A FLT-invariant family of objects unifies circles, lines and points—all together are called *cycles* in this framework. A FLT-invariant relations between cycles include incidence, orthogonality, tangency, etc. The efficiency of the package MoebInv is based on the mathematical formalism [2,13] which encodes the variety of different FLT-invariant relations between cycles in a unified way.

Besides aesthetical value the conformal and Möbius–Lie geometries have countless applications ranging from quantum spin dynamics [14], cosmology [15] and integrable systems [16] to computer-aided design [17] and physiological models [18].

2. Problems and background

From the point of view of synthetic Euclidean geometry a cycle (2D sphere) is the locus of points on a distance R from a given point (x_0, y_0). Its analytic description is encoded in a quadratic equation:

$$k(x^2+y^2) - 2lx - 2ny + m = 0, \tag{2}$$

where $l = kx_0$, $n = ky_0$, and $m = l^2 + n^2 - R^2$ for an arbitrary $k \neq 0$. Although it is theoretically sufficient for analytic solutions of various geometrical problems, practically its non-linearity produces rapidly increasing complexity of expression in symbolic computations and growing rounding errors in numeric evaluations.

It was only relatively recently realised [19,20] that the space of all spheres possesses a FLT-invariant inner product, which naturally encodes many fundamental geometric relations—the orthogonality to be the most important among them. In addition there is significant set of FLT-invariant geometric relations, e.g. tangency, Steiner power, which are genially quadratic. Analytic solutions of several simultaneous quadratic relations may be very involved. Fortunately, it was observed [2,21] that there is a possibility to reduce such problem to a set of *only one* quadratic relation and several linear ones. Such a set admits an effective algorithmic solution even in symbolic setup, which is the backbone of the package MoebInv.

More specifically, the first library `cycle` [1,13,22] manipulates individual cycles within the GiNaC [23] computer algebra system. The mathematical formalism employed in the library `cycle` is based on Clifford algebras and the Fillmore–Springer–Cnops construction (FSCc), which has a long history, see [24, § 1.1], [20, § 4.1], [19], [25, § 4.2], [26], [13, § 4.2]. Compared to a plain analytical treatment [11,27], FSCc is much more efficient and conceptually coherent in dealing with FLT-invariant properties of cycles. Correspondingly, the computer code based on FSCc is easy to write and maintain. The second library `figure` manipulates ensembles of cycles (quadrics) interrelated by certain FLT-invariant geometric conditions. There are methods to add, modify and delete elements of the figure as described in the next section.

3. Software framework

The library `figure` implements the *functional programming* framework—in contrast to *procedural approach* used in popular software packages like GeoGebra [4], CaRMetal [5], Kig [6], Dr. Geo [7]. The later provides a fixed set of geometric construction procedures, e.g. "find the midpoint of an interval", "drop the perpendicular from a point to a line". In contrast, all new cycles in `class figure` are added through a list of defining relations, which links the new cycle to already existing ones.[1]

3.1. Software architecture

Thinking a cycle ensemble as a graph, one can say that the library `cycle` deals with individual vertices (cycles), while `figure` considers edges (relations between pairs of cycles) and the whole graph. Intuitively, an interaction with the library `figure` reminds compass-and-straightedge constructions, where new lines, points or circles are added to a drawing one-by-one through relations to already presented objects (e.g. the line through two points, the intersection point or the circle with given centre and a point). To avoid "chicken or the egg" dilemma all cycles are stored in a hierarchical structure of generations, numbered by integers. The basic principles are:

1. Any explicitly defined cycle (i.e., a cycle which is not related to any previously known cycle) is placed into generation-0;
2. Any new cycle defined by relations to *previous* cycles from generations $k_1, k_2, \ldots, k_n$ is placed to the generation k calculated as:

$$k = \max(k_1, k_2, \ldots, k_n) + 1. \tag{3}$$

   This rule has an exception that a cycle may have a relation to itself, e.g. isotropy (self-orthogonality) condition $\langle C, C \rangle = 0$, which specifies point-like cycles.

If the number or nature of conditions is not sufficient to define the cycle uniquely (up to natural quadratic multiplicity), then the cycle will depend on a number of free (symbolic) variables.

3.2. Software functionalities

Both libraries are capable to work in spaces of any dimensionality and metrics with an arbitrary signatures: Euclidean, Minkowski and even degenerate. Parameters of objects can be symbolic or numeric, the latter admit calculations with exact or approximate arithmetic. Drawing routines work with any (elliptic, parabolic or hyperbolic) metric in two dimensions trough `Asymptote` [28] software.

There is a macro-like tool, which is called `subfigure`. Such a `subfigure` is a `figure` itself, such that its inner hierarchy of generations and relations is not visible from the current `figure`. Instead, some cycles (of any generations) of the current `figure` are used as predefined cycles of generation-0 of `subfigure`. Then only one dependent cycle of `subfigure`, which is known as `result`, is returned back to the current `figure`. The generation

[1] In fact, it is possible and useful to include relations of a new cycle to itself as well. For example, points are defined by the condition to be self-orthogonal.

of the `result` is calculated from generations of input cycles by the same formula (3).

There is a possibility to test certain conditions (e.g., "are two cycles orthogonal?") or measure certain quantities (e.g., "what is their intersection angle?") for already defined cycles. In particular, such methods can be used to prove geometrical statements according to the Cartesian programme, that is replacing the synthetic geometry by purely algebraic manipulations. Besides C++ libraries there is a Python wrapper, which can be used in interactive mode. It is accessible through cloud computing on several hosts [29,30] with sample Jupyter notebooks. There is a Graphical User Interface (GUI), which allows to create figures by mouse clicks. Python/Jupyterand GUI are bidirectionally integrated by data exchange routines. There are two – binary and human-readable/editable – formats provided for this purpose.

Metrics of the point space and the cycle spaces do not need to coincide. Similar bi-metric models was used recently to tackle the quantum Hall effect [31].

4. Illustrative examples

As an elementary demonstration, we an analytic proof of a geometrical statement from Jupyter notebooks [29,30]. Let P be a point of contact of a cycle a and its tangent l. We will show that a radius r of the cycle a to the point P is orthogonal to the line l. To simplify setup we assume that a is the unit circle.

First, we need to load libraries (assuming software installation is already done).

```
[1]: from figure import *
```

Then, we initialise a figure F with a default Euclidean metric.

```
[2]: F=figure()
```

We add the unit circle a to the figure by specifying the explicit coefficients $(1, 0, 0, -1)$ of its quadratic equations, cf. (2):

$$1 \cdot (x^2 + y^2) - 0 \cdot x - 0 \cdot y - 1 = 0.$$

```
[3]: a=F.add_cycle(cycle2D(1, [0, 0], -1), "a")
```

Then, we add the centre C of a as a specific point:

```
[4]: C=F.add_point(cycle2D(F.get_cycle(a)[0]).center(), "C")
```

Now, we want to add a line l tangent to a. A straight line is characterised among cycles by its orthogonality to infinity ("passes the infinity"). The line l is not uniquely defined by these two conditions (the tangency and orthogonality) because the point of its contact to a can be arbitrary. Therefore its coefficients contain a free variable like `t_l` as can be seen from the figure printout.

```
[5]: l=symbol("l")
     F.add_cycle_rel([is_tangent(a),is_orthogonal\
          (F.get_infinity()),\
          only_reals(l)],l)
     print(F.string())
```

```
C-(0): {'0, [[1,0]]~C-(0), 0', -3} --> (C); <-- ()
C-(1): {'0, [[0,1]]~C-(1), 0', -3} --> (C); <-- ()
infty: {'0, [[0,0]]~infty, 1', -2} --> (C,l); <-- ()
R: {'0, [[0,1]]~R, 0', -1} --> (); <-- ()
C: {'1, [[0,0]]~C, 0', 0} -->
(); <-- (C/o,infty|d,C-(0)|o,C-(1)|o)
a: {'1, [[0,0]]~a, -1', 0} --> (l); <-- ()
l:
{'0, [[-1/8*sqrt(8)*sqrt(2)*cos(t_l),-1/64*sqrt(2)
*sin(t_l)*sqrt(512)]]~l,1',
'0, [[1/8*sqrt(8)*sqrt(2)*cos(t_l),-1/64*sqrt(2)
*sin(t_l)*sqrt(512)]]~l,1',
'0, [[1/8*sqrt(8)*cos(t_l)*sqrt(2),1/64
*sin(t_l)*sqrt(2)*sqrt(512)]]~l,1',
'0, [[-1/8*sqrt(8)*cos(t_l)*sqrt(2),1/64
*sin(t_l)*sqrt(2)*sqrt(512)]]~l,1',1}
 --> (); <-- (a|t,infty|o,l|r)
Altogether 10 cycles in 7 cycle_nodes.
```

Note, that the parametrisation of the line l uses trigonometric functions in order to avoid square roots appearing in the solutions of the quadratic tangency relation. With such substitution automatic simplifications of algebraic expressions are much more efficient.

At the next step we add the point P of contact of the circle a and the line l. A point belongs to a cycle if the point is orthogonal the cycle. Also a point is characterised among all cycle by orthogonality to itself. To define the latter reflexive condition we need to "pre-cook" its symbol in advance.

```
[6]: P=symbol("P")
     P=F.add_cycle_rel([is_orthogonal(P), is_orthogonal(a),\
         is_orthogonal(l), only_reals(P)], P)
```

Finally we add the radius r passing P: it is a straight line (is orthogonal to the infinity) and passes both P and C (is orthogonal to each of them).

```
[7]: r=F.add_cycle_rel([is_orthogonal(P), is_orthogonal(C),\
          is_orthogonal(F.get_infinity())], "r")
```

Recall, that r depends on the same free parameters as l. Now, we check the orthogonality relation between r and l. Because two of the above conditions were quadratic (i.e., tangency and self-orthogonality), each of them doubled the number of solutions. Therefore, there are four instances of the cycle r, and each of them is checked separately.

```
[8]: Res=F.check_rel(l,r,"orthogonal")
     for i in range(len(Res)):
        print("Tangent and radius are orthogonal: %s"%\
        bool(Res[i].subs(pow(cos(wild(0)),2)==1-pow\
           (sin(wild(0)),2))\
           .normal()))
```

```
Tangent and radius are orthogonal: True
Tangent and radius are orthogonal: True
Tangent and radius are orthogonal: True
Tangent and radius are orthogonal: True
```

Note, that we had uses the Pythagoras substitution $\cos^2 t = 1 - \sin^2 t$ to assist the CAS with algebraic simplifications. Another interesting moment is that the entire construction is based on the single relation of orthogonality (apart from the tangency required to define l).

Many more examples of computer-assisted proofs were presented in [13] and are freely available for evaluation at [29, 30].

5. Software impact

The first library cycle was initially reported in [1] and since then was employed as the main research tool in several papers [22,26] and monograph [13]. New results, e.g. elliptic and hyperbolic conformal version of the nine-point theorem, discovered with the second library figure were published in [2,3], cf. Fig. 1. The package has several different uses:

- It is easy to produce high-quality illustrations, which are fully-accurate with automatic evaluation of cycles' parameters, cf. Fig. 2

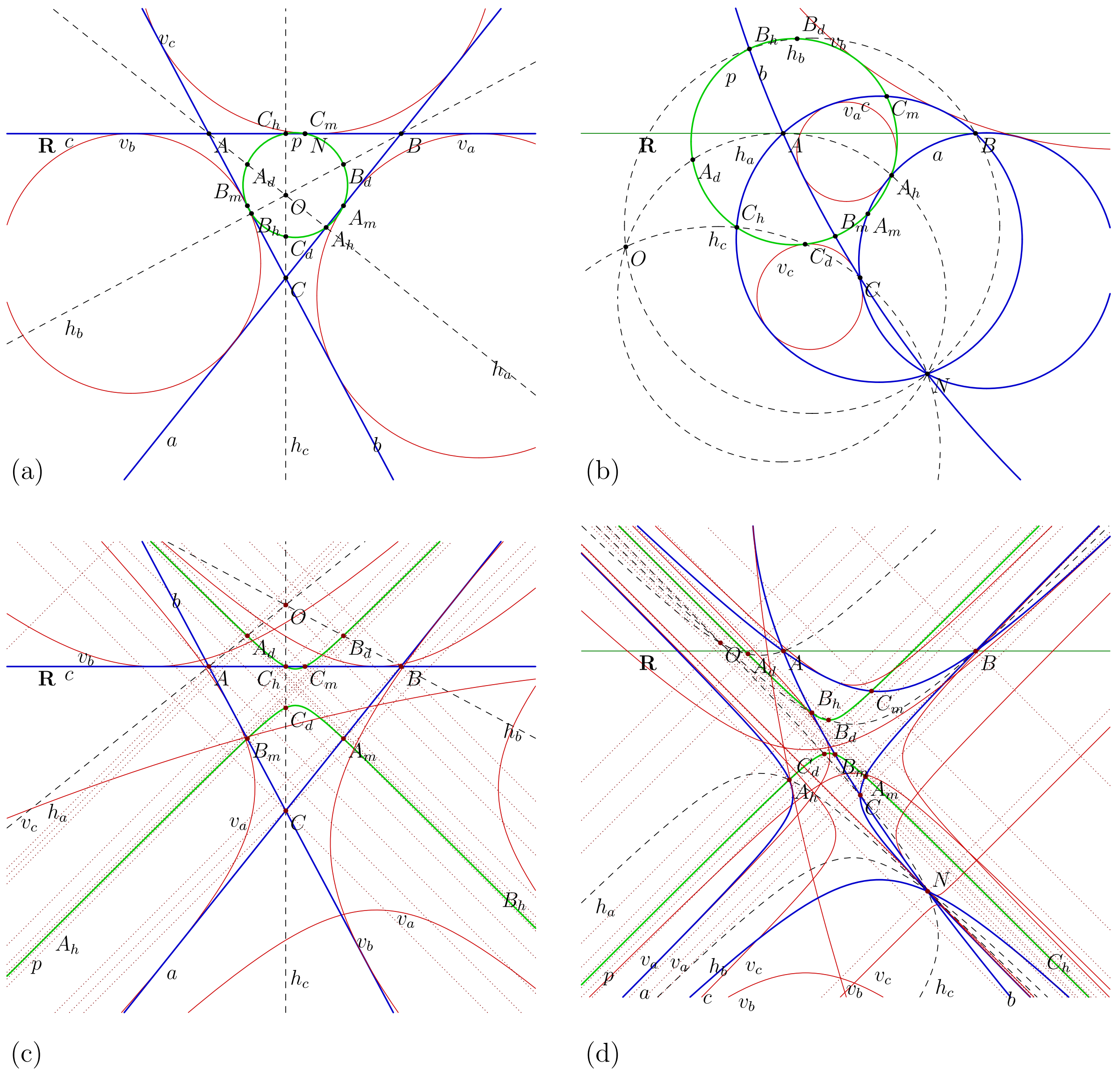

Fig. 1. The illustration of the conformal nine-points theorem discovered with MoebInv package. The left column is the statement for a triangle with straight sides, the right column is its conformal version. The first row show the elliptic point space, the second row—the hyperbolic point space.

- The package can be used for computer experiments in Möbius–Lie geometry.
- Since the library is based on the GiNaC system, which provides a symbolic computation engine, there is a possibility to make fully automatic proofs of various statements in Möbius–Lie geometry, see the previous Section.
- Last but not least, the combination of classical beauty of Lie sphere geometry and modern computer technologies is a useful pedagogical tool to widen interest in mathematics through visual and hands-on experience. A dedicated GUI makes it accessible for all schoolchildren. Several videos created with the package are uploaded to the popular video hosting [32] to disseminate aesthetical attraction of mathematics to wider public.

As a non-trivial example of automated proof accomplished by the `figure` library for the first time, we present a FLT-invariant version of the classical nine-point theorem [2,3], which is illustrated in the attached video.

6. Conclusions

The package accumulates more than 15 years of work and is still under active development. The design of the library `figure` shaped the general theoretical approach to the extension of Möbius–Lie geometry [2,3], which leaded to specific realisations in [33–35]. Furthermore, it shall be helpful for computer experiments in Lie sphere geometry of indefinite or nilpotent metrics since our intuition is not elaborated there in contrast to the Euclidean space [22,26,36]. Some advances in the two-dimensional space were achieved recently [13,37], however further developments in higher dimensions are still awaiting their researchers.

The current version (v3.5.7) of the package was tested on major desktop platforms: Linux (Debian, Ubuntu and CentOS distributions), Windows 10, Mac OS X. Pre-compiled binaries for these OSes are provided on the project Web page [2]. The interactive Python/Jupyter shells can be used from cloud services CoLab [30] and CodeOcean [29].

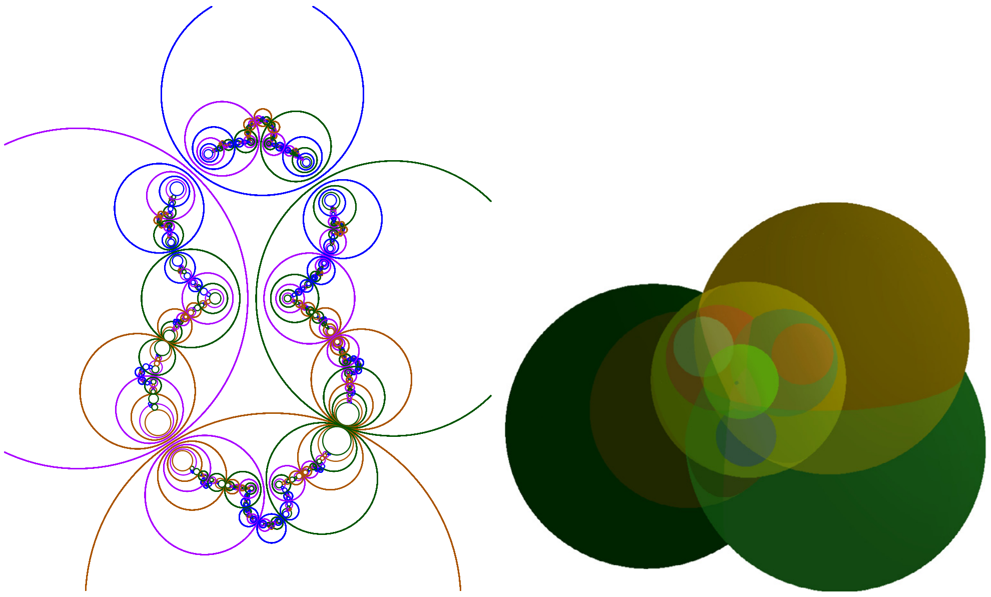

Fig. 2. Left: a Kleinian group generated by four cycle reflections. Right: an example of Apollonius problem in three dimensions.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

I am grateful to Prof. Jay P. Fillmore for stimulating discussion, which enriched the package. The University of Leeds provided supports for several student projects, which helped to develop the package. L. Hutton wrote the initial version of the GUI and helped to port the package to Mac OS X.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.softx.2019.100385.

References

[1] Kisil VV. Fillmore-Springer-Cnops construction implemented in `GiNaC`. Adv Appl Clifford Algebr 2007;17(1):59–70, On-line. A more recent version: E-print: arXiv:cs.MS/0512073. The latest documentation, source files, and live ISO image are at the project page: http://moebinv.sourceforge.net/. Zbl05134765.

[2] Kisil VV. An extension of Mobius–Lie geometry with conformal ensembles of cycles and its implementation in a GiNaC library. Proc Int Geom Cent 2018;11(3):45–67. http://dx.doi.org/10.15673/tmgc.v11i3.1203, E-print: arXiv:1512.02960. Project page: http://moebinv.sourceforge.net/.

[3] Kisil VV. Möbius–Lie geometry and its extension. In: Mladenov IM, Meng G, Yoshioka A, editors. Geometry, integrability and quantization XX. Sofia: Bulgar. Acad. Sci.; 2019, p. 13–61. http://dx.doi.org/10.7546/giq-20-2019-13-61, E-print: arXiv:1811.10499.

[4] Hohenwarter M, et al. GeoGebra, https://www.geogebra.org/, an Interactive Geometry, Algebra, Staistics and Calculus Application (since 2001).

[5] Hakenholz E, et al. CaRMetal, http://carmetal.org/, free Software on Dynamical Geometry (since 2006).

[6] Devriese D, et al. Kig, https://edu.kde.org/kig/, free and Open-Source Interactive Geometry Software (since 2006).

[7] Fernandes H, et al. GNU Dr. Geo, http://www.drgeo.eu/, interactive Geometry Software (since 1996).

[8] Simon B. Szegő's theorem and its descendants. In: Spectral theory for L^2 perturbations of orthogonal polynomials. M. B. Porter lectures, Princeton, NJ: Princeton University Press; 2011.

[9] Yaglom IM. A simple non-Euclidean geometry and its physical basis. Heidelberg science library, New York: Springer-Verlag; 1979, translated from the Russian by Abe Shenitzer, with the editorial assistance of Basil Gordon.

[10] Cecil TE. Lie sphere geometry: With applications to submanifolds. Universitext, 2nd ed. New York: Springer; 2008.

[11] Benz W. Classical geometries in modern contexts. Geometry of real inner product spaces. 2nd ed.. Basel: Birkhäuser Verlag; 2007, http://dx.doi.org/10.1007/978-3-0348-0420-2.

[12] Juhász ML. A universal linear algebraic model for conformal geometries. J Geom 2018;109(3). 48. http://dx.doi.org/10.1007/s00022-018-0451-1, 18, E-print: arXiv:1603.06863.

[13] Kisil VV. Geometry of Möbius transformations: Elliptic, parabolic and hyperbolic actions of $SL_2(\mathbf{R})$. London: Imperial College Press; 2012, includes a live DVD. Zbl1254.30001.

[14] Jaćimović V. Magnetization dynamics and geometry: Coupled Möbius transformations. Rep Math Phys 2018;81(3):347–57. http://dx.doi.org/10.1016/S0034-4877(18)30052-1.

[15] Cartas-Fuentevilla R, Escalante-Hernandez A, Herrera-Aguilar A, Gonzalez-Quaglia R. Hyperbolic symmetries, inflaton-phantom cosmology, and inflation. 2019, arXiv e-prints arXiv:1907.03551.

[16] Bobenko AI, Schief WK. Circle complexes and the discrete CKP equation. Int Math Res Not IMRN 2017;(5):1504–61. http://dx.doi.org/10.1093/imrn/rnw021.

[17] Jüttler B, Maroscheck S, Kim M-S, Hong QY. Arc fibrations of planar domains. Comput Aided Geom Design 2019;71:105–18. http://dx.doi.org/10.1016/j.cagd.2019.04.010, URL http://www.sciencedirect.com/science/article/pii/S016783961930024X.

[18] Zhang J. Characterizing projective geometry of binocular visual space by Möbius transformation. J Math Psych 2019;88:15–26. http://dx.doi.org/10.1016/j.jmp.2018.10.007.

[19] Fillmore JP, Springer A. Möbius groups over general fields using Clifford algebras associated with spheres. Internat J Theoret Phys 1990;29(3):225–46. http://dx.doi.org/10.1007/BF00673627.

[20] Cnops J. An introduction to dirac operators on manifolds. Progress in mathematical physics, vol. 24, Boston, MA: Birkhäuser Boston Inc.; 2002.

[21] Fillmore JP, Springer A. Determining circles and spheres satisfying conditions which generalize tangencyPreprint, http://www.math.ucsd.edu/fillmore/papers/2000LGalgorithm.pdf.

[22] Kisil VV. Erlangen program at large–0: Starting with the group $SL_2(\mathbf{R})$. Notices Amer Math Soc 2007;54(11):1458–65, E-print: arXiv:math/0607387, On-line. Zbl1137.22006.

[23] Bauer C, Frink A, Kreckel R. Introduction to the GiNaC framework for symbolic computation within the C++ programming language. J Symbolic Comput 2002;33(1):1–12. http://dx.doi.org/10.1006/jsco.2001.0494, web: http://www.ginac.de/. E-print: arXiv:cs/0004015. URL http://www.sciencedirect.com/science/article/pii/S0747717101904948.

[24] Schwerdtfeger H. Geometry of complex numbers: Circle geometry, Moebius transformation, non-Euclidean geometry. Dover books on advanced mathematics, New York: Dover Publications Inc.; 1979, a corrected reprinting of the 1962 edition.

[25] Kirillov AA. A tale of two fractals. New York: Springer; 2013, http://dx.doi.org/10.1007/978-0-8176-8382-5, draft: http://www.math.upenn.edu/kirillov/MATH480-F07/tf.pdf.

[26] Kisil VV. Erlangen program at large–1: Geometry of invariants. SIGMA Symmetry Integr Geom Methods Appl 2010;6(076):45. http://dx.doi.org/10.3842/SIGMA.2010.076, E-print: arXiv:math.CV/0512416. MR2011i:30044. Zbl1218.30136.

[27] Pedoe D. Circles: A mathematical view. In: MAA spectrum. Washington, DC: Mathematical Association of America; 1995, revised reprint of the 1979 edition, With a biographical appendix on Karl Feuerbach by Laura Guggenbuhl.

[28] Hammerlindl A, Bowman J, Prince T. Asymptote—powerful descriptive vector graphics language for technical drawing, inspired by MetaPost. Tech. rep., 2004–2019, uRL: http://asymptote.sourceforge.net/.

[29] Kisil VV. MoebInv library: Demo (v5), http://dx.doi.org/10.24433/CO.9934595.v5. URL https://codeocean.com/capsule/7952650/tree.

[30] Kisil VV. MoebInv library: Jupyter notebooks (v0.3), https://github.com/vvkisil/MoebInv-notebooks. http://dx.doi.org/10.5281/zenodo.3558468.

[31] Lapa MF, Gromov A, Hughes TL. Geometric quench in the fractional quantum Hall effect: Exact solution in quantum Hall matrix models and comparison with bimetric theory. Phys Rev B 2019;99. 075115. http://dx.doi.org/10.1103/PhysRevB.99.075115, URL https://link.aps.org/doi/10.1103/PhysRevB.99.075115.

[32] Kisil VV. MoebInv illustrations, YouTube playlist: https://goo.gl/Z9GUL0 (2015–19). URL https://goo.gl/Z9GUL0.

[33] Kisil VV. Poincaré extension of Möbius transformations. Complex Var Elliptic Equ 2017;62(9):1221–36. http://dx.doi.org/10.1080/17476933.2016.1250399, E-print: arXiv:1507.02257.

[34] Kisil VV. Remark on continued fractions, Möbius transformations and cycles. Известия Коми научного центра УрО РАН [Izv. Komi Nauchnogo cent. UrO RAN] 2016;25(1):11–7, E-print: arXiv:1412.1457, on-line. URL http://www.izvestia.komisc.ru/Archive/i25_ann.files/kisil.pdf.

[35] Kisil VV, Reid J. Conformal parametrisation of loxodromes by triples of circles. In: Bernstein S, editor. Topics in Clifford analysis: Special volume in honor of Wolfgang Sprößig. Cham: Birkhäuser; 2019, p. 313–30. http://dx.doi.org/10.1007/978-3-030-23854-4_15, E-print: arXiv:1802.01864.

[36] Kisil VV. Erlangen program at large—2: Inventing a wheel. The parabolic one. Zb Pr Inst Mat NAN Ukr (Proc Math Inst Ukr Ac Sci) 2010;7(2):89–98, E-print: arXiv:0707.4024.

[37] Mustafa KA. The groups of two by two matrices in double and dual numbers, and associated Möbius transformations. Adv Appl Clifford Algebr 2018;28(5). 92. http://dx.doi.org/10.1007/s00006-018-0910-7, 25, E-print: arXiv:1707.01349.