Can Artificial Intelligence Serve Computer Algebra Systems?

Matthew England (Coventry University, UK)

Polish Conference on Artificial Intelligence (PP-RAI)
Łódź, Poland 24–26 April 2023

Author currently supported by EPSRC grant EP/T015748/1.

Coventry University

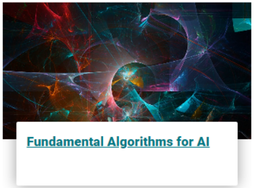
Based in Coventry (UK) but with campuses also in: London and Scarborough (UK); Wrocław (Poland); and in Egypt (as part of the Knowledge Hub).

The Wrocław campus opened in 2020 and is providing undergraduate education including in computing. In 2023 we will open *Coventry University Research Institute Europe* (CURIE) in Wrocław as the European outpost of the university's research, with a focus on clean growth and digital transformation.

Coventry University Research

Our research is organised into 16 research centres spanning engineering, arts, business and health. In the UK's 2021 research evaluation Coventry jumped into the top half of the Times Higher research power ranking.

The *Centre for Computational Science and Mathematical Modelling* is organised into three groups, all of whom are currently open to collaboration on new projects.



Outline

Symbolic Computation and Computer Algebra

Symbolic Computation refers to algorithms and software for manipulating exact mathematical expressions and objects.

This gives *exact answers*, as opposed to the more standard *Numerical Computation* which uses floating point arithmetic to give approximate answers.

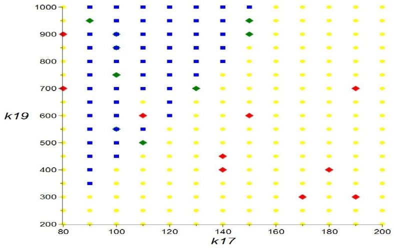
E.g. Compute with the symbols $\frac{1}{3}$, π and $\sqrt{2}$ rather than approximations 0.33333, 3.14159 and 1.41421.

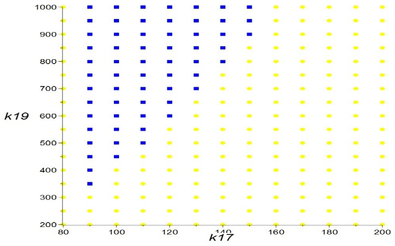
Traditionally, symbolic computation is implemented in specialist Computer Algebra Systems, e.g. Maple, Mathematica, SymPy, SageMath.

Why Use Symbolic Computation? 1/3

Symbolic Computation usually take lots more computer resources than numerical computation. So why use it?

- Problems not suited for numerical solution (e.g. chaotic behaviour; small inaccuracies lead to different answers).



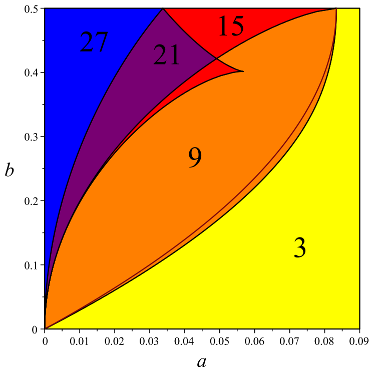


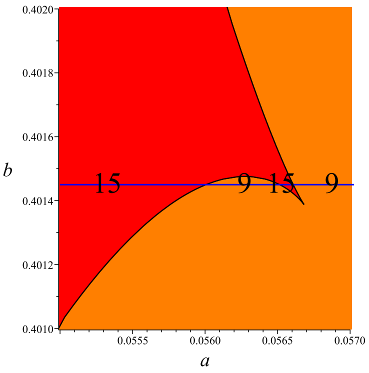
Why Use Symbolic Computation? 2/3

- Accuracy may be particularly important (e.g. safety critical systems; formal verification; large numerics).

Why Use Symbolic Computation? 3/3

- Because exact solutions offer fundamental insight that can be missing from numerical ones. Understand not just what the solution is but what it means; what properties it has.





Quantifier Elimination

Symbolic Computation is more than exact arithmetic. Encompasses polynomials, special functions, differential equations, etc.

My research speciality is algorithms to answer questions about polynomial systems. E.g. questions like **quantifier elimination**:

Input: $\exists x, x^2 + 3x + 1 > 0$
Output: `True`

Does there exist an x to satisfy this polynomial constraint? Yes, e.g. $x = 0$.

Input: $\forall x, x^2 + 3x + 1 > 0$
Output: `False`

Is this polynomial constraint satisfied by all x? No, e.g. $x = -1$.

Input: $\forall x, x^2 + bx + 1 > 0$
Output: $(-2 < b) \wedge (b < 2)$

For what values of b is the polynomial constraint satisfied by all x? When b is between -2 and 2.

Computer Algebra VS Machine Learning

Machine Learning (ML) can use statistics and big data to learn how to perform tasks that have not been explicitly programmed.

(Q) So can ML replace symbolic computation?

There is a growing body of research on the use of ML in place of expensive symbolic computation. E.g. for symbolic integration and the solution of differential equations [Lample and Charton, 2020].

However, like many ML applications, there are issues of over-fitting: where the ML does very well on the data used to train it but poorly on different data. Unlike other ML applications it is not always obvious which data is similar to the training data.

The examples in [Lample and Charton, 2020] work well because it is cheap to symbolically check the correctness of the answer. This is not the case for most symbolic computation.

Computer Algebra WITH Machine Learning

ML can only offer probabilistic guidance, but symbolic computation prizes exact results. 99% accuracy is great for image recognition but would not be acceptable for a mathematical proof.

However: ML can be applied to symbolic computation and still ensure exact results; by having it guide existing algorithms rather than replace them entirely.

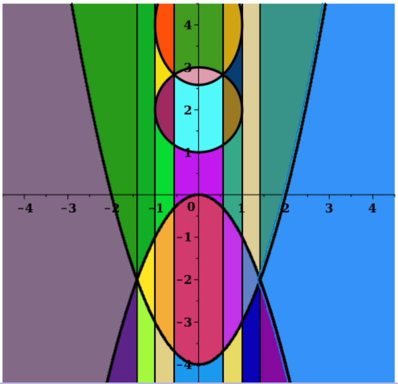
Computer Algebra algorithms will often come with choices that need to be made but which do not effect the mathematical correctness of the final result; but do effect the resources required to find that result, and how the result is presented.

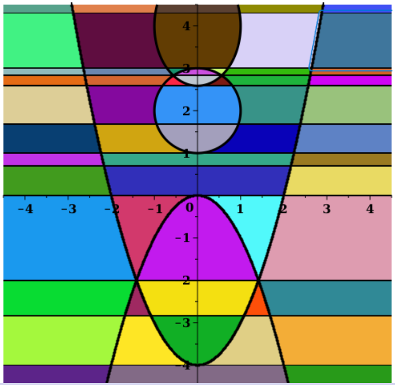
Such choices are often either left to the user, hard coded by the developer, or made based on a simple heuristic. ML can offer a superior choice.

Example: Variable Ordering for Quantifier Elimination

Quantifier Elimination may be tackled via a cylindrical algebraic decomposition (CAD) of the space [Collins, 1975].

That algorithm requires a variable ordering: there can be multiple valid orderings which lead to an acceptable decomposition, but some lead to smaller decompositions via less computation.





Survey: ML to Optimise Computer Algebra

- Huang et al. [2014] was the first use of ML for computer algebra: used a support vector machine to select the CAD variable ordering (prior slide).
- Kuipers et al. [2015] used a Monte-Carlo tree search to find the representation of polynomials that are most efficient to evaluate numerically.
- Simpson et al. [2016] used ML to choose which algorithm to compute the resultant with for a given problem instance.
- Brown and Daves [2020] used a neural network to select the order of polynomial constraints to process in their solver.
- Peifer et al. [2020] applied reinforcement learning to select which S-Pair to process next when building a Gröbner Basis.

Fuller survey in [Pickering et al., 2023].

This is INTERESTING Machine Learning

So ML has great potential for Symbolic Computation. But note this is also a particularly challenging / interesting ML domain:

- No a priori limit on the input space.
- Supervised learning hard: because labelling dataset needs lots of expensive symbolic computation.
- Unsupervised learning is hard: because it is unclear if a particular outcome is good or bad without seeing the competition!
- What constitutes a meaningful and representative data set?
- Insufficient quantities of real world data for deep learning.
- How to perform data augmentation / synthetic data generation to allow for good generalisability on problems of interest?

Beyond Efficiency Gains

Peifer et al. [2020] applied ML to choose the order in which to process S-pairs in Buchberger's algorithm for a Gröbner Basis.

A human analysis of their model revealed a simple, "*human level*" strategy that explained most of the benefit of the ML model's choices. Simple, but very different to the other human designed heuristics in use.

Suggests that ML can reveal new mathematical truths about the algorithm, and direct future non-ML algorithm development. But how to automate the analysis?

We have been experimenting with *Explainable ML* (XAI) for this process. Started by looking at explanations for the performance of `sklearn` classifiers previously developed (by Florescu and England [2020] with an algorithmic feature engineering approach).

Recent work on XAI for QE variable ordering

Pickering et al. [2023] applied the SHAP XAI tool to analyse the features: some of those identified as the most impactful had been known before but others were new.

Next constructed non-ML heuristics from trio's of these features, in a similar design to prior human-designed heuristics. The best of these outperform's the state-of-the-art on standard benchmarks.

Demonstrates the potential for XAI as an exploratory tool in human optimisation of symbolic computation algorithms.

Contact Details

Contact Details

Matthew.England@coventry.ac.uk

https://matthewengland.coventry.domains/

Thanks for Listening

Bibliography I

C. W. Brown and G. C. Daves. Applying machine learning to heuristics for real polynomial constraint solving. In A. Bigatti, J. Carette, J. H. Davenport, M. Joswig, and T. de Wolff, editors, *Mathematical Software – ICMS 2020*, volume 12097 of *Lecture Notes in Computer Science*, pages 292–301. Springer International Publishing, 2020. URL https://doi.org/10.1007/978-3-030-52200-1_29.

B. Caviness and J. Johnson. *Quantifier Elimination and Cylindrical Algebraic Decomposition*. Texts & Monographs in Symbolic Computation. Springer-Verlag, 1998. URL https://doi.org/10.1007/978-3-7091-9459-1.

Bibliography II

G. E. Collins. Quantifier elimination for real closed fields by cylindrical algebraic decomposition. In *Proceedings of the 2nd GI Conference on Automata Theory and Formal Languages*, pages 134–183. Springer-Verlag (reprinted in the collection Caviness and Johnson [1998]), 1975. URL https://doi.org/10.1007/3-540-07407-4_17.

D. Florescu and M. England. A machine learning based software pipeline to pick the variable ordering for algorithms with polynomial inputs. In A. Bigatti, J. Carette, J. H. Davenport, M. Joswig, and T. de Wolff, editors, *Mathematical Software – ICMS 2020*, volume 12097 of *Lecture Notes in Computer Science*, pages 302–322. Springer International Publishing, 2020. URL https://doi.org/10.1007/978-3-030-52200-1_30.

Bibliography III

Z. Huang, M. England, D. Wilson, J. H. Davenport, L. Paulson, and J. Bridge. Applying machine learning to the problem of choosing a heuristic to select the variable ordering for cylindrical algebraic decomposition. In S. M. Watt, J. H. Davenport, A. P. Sexton, P. Sojka, and J. Urban, editors, *Intelligent Computer Mathematics*, volume 8543 of *Lecture Notes in Artificial Intelligence*, pages 92–107. Springer International, 2014. URL http://dx.doi.org/10.1007/978-3-319-08434-3_8.

J. Kuipers, T. Ueda, and J. A. M. Vermaseren. Code optimization in FORM. *Computer Physics Communications*, 189:1–19, 2015. URL https://doi.org/10.1016/j.cpc.2014.08.008.

Bibliography IV

G. Lample and D. Charton. Deep learning for symbolic mathematics. In S. Mohamed, M. White, K. Cho, and D. Song, editors, *Eighth International Conference on Learning Representations (ICLR 2020)*, 2020. URL https://iclr.cc/virtual_2020/poster_S1eZYeHFDS.html.

D. Peifer, M. Stillman, and D. Halpern-Leistner. Learning selection strategies in Buchberger's algorithm. In H. Daumé III and A. Singh, editors, *Proceedings of the 37th International Conference on Machine Learning (ICML 2020)*, volume 119 of *Proceedings of Machine Learning Research*, pages 7575–7585. PMLR, 2020. URL https://proceedings.mlr.press/v119/peifer20a.html.

Bibliography V

Lynn Pickering, Tereso Del Rio Almajano, Matthew England, and Kelly Cohen. Explainable AI insights for symbolic computation: A case study on selecting the variable ordering for cylindrical algebraic decomposition. *Submitted*, 2023. URL https://arxiv.org/abs/2304.12154.

Matthew C. Simpson, Qing Yi, and Jugal Kalita. Automatic algorithm selection in computational software using machine learning. In *15th IEEE International Conference on Machine Learning and Applications (ICMLA 2016)*, pages 355–360, 2016. URL https://doi.org/10.1109/ICMLA.2016.0064.