Tutorial: Exact Numerical Computation in Algebra and Geometry

Chee K. Yap*
Courant Institute of Mathematical Sciences
New York University, New York, NY 10012, USA
and
Korea Institute of Advanced Study, Seoul, Korea
yap@cs.nyu.edu

ABSTRACT

Many problems in Computational Science & Engineering (CS&E) are defined on the continuum. Standard algorithms for these problems are numerical and approximate. Their computational techniques include iteration, subdivision, and approximation. Such techniques are rarely seen in exact or algebraic algorithms. In this tutorial, we discuss a mode of computation called **exact numerical computation** (ENC) that achieves exactness through numerical approximation. Through ENC, we can naturally incorporate iteration, subdivision and approximation into the design of exact algorithms for computer algebra and computational geometry. Such algorithms are both novel and practical. This tutorial on ENC is divided into three equal parts:
(a) Zero Problems
(b) Explicitization Problems
(c) Techniques and Complexity Analysis of Adaptivity

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems —*Geometrical Problems and Computations*; I.1.2 [**Symbolic and Algebraic Manipulation**]: Algorithms—*Algebraic Algorithms*

General Terms

Algorithms, Theory

Keywords

Exact Numerical Computation, Numerical Computational Geometry, Numerical Algebraic Computation, Zero Bounds, Explicitization Problems, Adaptive Complexity Analysis

1. INTRODUCTION

The algorithms of theoretical computer science are strikingly discrete, combinatorial and exact. This fact is often celebrated by computer scientists, and rightly so. This exact view is also resonant with computer algebra. There is also an impulse to extend this view into a universal truth about all computation (e.g., Zeilberger [12]). The attitude is that, since all computation is discrete, the notion of continuous computation is an unnecessary abstract construct. But there is a *very* big world of continuous computation out there. Computational Science and Engineering (CS&E) is a convenient label for this world. The problems of CS&E are largely set in the continuum and are solved numerically and approximately. The continuum often refers to the real numbers $\mathbb{R}$, but for our purposes, we may expand its reference to any locally compact topological space such as $\mathbb{C}$ or $\mathbb{R}^n$.

The continuous world has a similar impulse to universalize its view point. In contrast to the exact and finitistic view, the experience of numerical computation suggests the opposite view: "*most problems of continuous mathematics cannot be solved in finite time*" [11, p. 323, Appendix]. But this is not a problem, says the algebraist/computer scientist: we can compute in $\mathbb{R}$ just as we do in any ring or field. Use the Real RAM model, and we are back in the world of discrete exact computation. But this model, though useful for other purposes, does not suffice if you want to solve practical numerical problems in CS&E. One difference between computing in $\mathbb{R}$ and fields such as GF_p or $\mathbb{Q}$ is the awkward matter of uncountability of $\mathbb{R}$: only a vanishing fraction of the elements in $\mathbb{R}$ are even representable. Despite this, we can build profound mathematical theories on the continuum. Computing in the continuum is another matter: we can only do this very delicately, by making careful choices and circumscriptions. We must impose conditions on our input in order to assure solvability – continuity, Lipschitz, smoothness, Morseness, non-singularity, well-posed, well-conditioned, etc. Such conditions are well-studied in applied mathematics. In discrete computation, we can always invoke the "brute force search algorithm" when all else fails. No such super-algorithm exists in continuous computation.

2. EXACT NUMERICAL COMPUTATION

This tutorial is about a synthesis of the above two views of computation. This synthesis might be called **exact numerical computation** or ENC for short. Through ENC, we can produce algorithms that are practical and relevant to the problems of CS&E. New forms of exact algorithms will arise in computer algebra and computational geometry.

This talk is divided into three 45-minutes parts, briefly introduced in the following subsections.

*This work is supported by NSF Grants CCF-043086, CCF-0728977, and by Korea Institute of Advanced Studies.

ISSAC'09, July 28–31, 2009, Seoul, Republic of Korea.
ACM 978-1-60558-609-0/09/07.

2.1 Zero Problems

This work originated with computational geometers who wanted to solve the numerical non-robustness issues that plague geometric algorithms [10]. Computational Geometry gave us unique insights into what it means to compute exactly in the continuum. This can be precisely located in the so-called Zero Problems. Based on this, a mode of numerical computation was invented that amounts to exact computation with algebraic numbers. This is encoded in libraries such as `LEDA` or `Core Library`. Surprisingly, there is no analogue in computer algebra, which has many other techniques for computing with algebraic numbers (e.g., [2] or [5, Chap. 4]).

The key difference is that in geometric applications, we are less interested in the algebraic properties embedded in $\mathbb{R}$ or $\mathbb{C}$ than in their analytic properties. This remark applies also to most applications of CS&E. But we do need to exploit the algebraic properties of numbers, but only for the purposes of deciding zero. Thus we use a "stripped-down" form of algebraic computation, which is ultimately more efficient than a purely algebraic approach.

In this section, we will review this mode of exact numerical computation, touching on the topics of precision-driven evaluation and filter technology. The focus is on constructive zero bounds.

2.2 Explicitization Problems

In CAD/CAM, a surface S is said to be **implicit** if it is given by a defining equation $f(x, y, z) = 0$. In many applications, we desire some mesh or triangulation T that is a simplicial approximation of S. Various properties can be imposed on T: the most common ones being (1) T is isotopic to S, and (2) T is ε-close to S. This is then called the **mesh generation problem**. Mesh generation in 1-dimension is well-known in computer algebra: if S is the zero set of a univariate polynomial $f(x)$, then (1) amounts to root isolation and (2) amounts to root refinement. Meshing is a special case of the general **explicitization problem**, that of computing an explicit approximation T from some implicit representation S. The representation "$f(x, y, z) = 0$" encodes S exactly, but the analytic properties of S are not easily accessible. A simple example is where S is given by the expression $\sqrt{25 - \sqrt{624}}$, and T might be 0.1414. Explicitization is the critical first-step in many CS&E applications: subsequent processes (from visualization to computing algebraic invariants) rely on the combinatorial representation T.

There are many approaches to meshing problems as surveyed in [1]. In this section, we describe an approach originating in Plantinga and Vegter's work on meshing of nonsingular surfaces [8]. It serves as a paradigm for numerical approaches to geometric problems. We will discuss the computational basis for ENC (computational rings, box functions) and extensions of Plantinga-Vegter: non-algebraic functions, singularities [3], improved techniques [7], and the arbitrary dimensional case from Galehouse's thesis [6].

2.3 Complexity Analysis and Techniques: Adaptivity Issues

A main characteristic of numerical computation is its **adaptivity**, namely its complexity can vary greatly for different input instances of the same input size. Adaptive algorithms are usually favored by practitioners because, in a certain measure-theoretic sense, the difficult inputs are rare. This is in sharp contrast to algebraic algorithms that are largely non-adaptive. One of the biggest challenge in exact numerical algorithms is to develop complexity analysis techniques that could account for adaptivity. Until recently, all adaptive complexity analysis relies on some probabilistic assumption. We introduce the technique of **integral analysis** that makes no probabilistic assumption [4]. We also touch on another important theme, computational approaches to estimating zero bounds, drawing on recent work of Sagraloff, Kerber and Hemmer [9].

3. REFERENCES

[1] J.-D. Boissonnat, D. Cohen-Steiner, B. Mourrain, G. Rote, and G. Vegter. Meshing of surfaces. In J.-D. Boissonnat and M. Teillaud, editors, Effective Computational Geometry for Curves and Surfaces. Springer, 2006. Chapter 5.

[2] B. Buchberger, G. E. Collins, and R. Loos, editors. Computer Algebra. Springer-Verlag, Berlin, 2nd edition, 1983.

[3] M. Burr, S. Choi, B. Galehouse, and C. Yap. Complete subdivision algorithms, II: Isotopic meshing of singular algebraic curves. In Proc. Int'l Symp. Symbolic and Algebraic Computation (ISSAC'08), pages 87–94, 2008. Hagenberg, Austria. Jul 20-23, 2008.

[4] M. Burr, F. Krahmer, and C. Yap. Integral analysis of evaluation-based real root isolation, Feb. 2009. Submitted, 2009.

[5] H. Cohen. A Course in Computational Algebraic Number Theory. Springer, 1993.

[6] B. Galehouse. Topologically Accurate Meshing Using Spatial Subdivision Techniques. Ph.D. thesis, New York University, Department of Mathematics, Courant Institute, May 2009. From `http://cs.nyu.edu/exact/doc/`.

[7] L. Lin and C. Yap. Adaptive isotopic approximation of nonsingular curves: the parametrizability and non-local isotopy approach. In Proc. 25th ACM Symp. on Comp. Geometry, page to appear, June 2009. Aarhus, Denmark, Jun 8-10, 2009.

[8] S. Plantinga and G. Vegter. Isotopic approximation of implicit curves and surfaces. In Proc. Eurographics Symposium on Geometry Processing, pages 245–254, New York, 2004. ACM Press.

[9] M. Sagraloff, M. Kerber, and M. Hemmer. Certified complex root isolation by subdivision and modular computation, 2009. Preprint.

[10] S. Schirra. Robustness and precision issues in geometric computation. In J. Sack and J. Urrutia, editors, Handbook of Computational Geometry. Elsevier Science Publishers, B.V. North-Holland, Amsterdam, 1999.

[11] L. N. Trefethen and D. Bau. Numerical linear algebra. Society for Industrial and Applied Mathematics, Philadelphia, 1997.

[12] D. Zeilberger. "real" analysis is a degenerate case of discrete analysis. In B. Aulbach, S. Elaydi, and G. Ladas, editors, New Progress in Difference Equations, London, 2001. Taylor and Frances. Proc. ICDEA 2001.