A puzzling fact about numerical analysis

Jean-Paul Berrut

A typical problem in scientific computing is the approximation of the solution of a functional equation, which often is infinitely smooth at every point of its domain of definition, apart from the boundary and other subsets of measure zero, e.g., discontinuities. Most methods approximate such solutions with piecewise defined functions (splines, finite elements, finite volumes), thereby introducing derivative discontinuities at many locations where the true solution does not have any. These methods are efficient and perfectly suited for engineering needs, many of them even optimal in a sense or another.

My intention is in no way to criticize the fact that most scientists working on the solution of such problems concentrate on these methods. I just wish to convey my puzzlement at the fact that so many academic mathematicians do not seem to feel any unease at the introduction of such derivative discontinuities in the approximation of smooth functions, and that so few work on infinitely smooth interpolants in several variables. Admittedly, there are quite some, e.g., those who use and study radial basis functions or Fourier extensions, but not nearly as many as I would expect.

Chebfun impressively and beautifully demonstrates the effectiveness of smooth one-dimensional approximants. These may oscillate more than splines of comparable accuracy, but their convergence is faster and automatically adjusts to the smoothness of the underlying function. Chebfun2 effectively approximates two-dimensional functions, using, amongst other tools, efficient storage saving techniques; it seems however limited to relatively simple domains. Fundamental research on infinitely smooth two-dimensional interpolants may lead to interesting new approaches; yet the number of scientists working on them appears surprisingly small.

New directions in model order reduction for dynamical systems

Mark Embree

Discretization of a complicated partial differential equation often leads to a dynamical system with millions of spatial variables. In part this large dimension comes from accurate modeling of solution components that have little effect on the system's response to a specific physically relevant input. Model Order Reduction (MOR) is the craft of (automatically and drastically) reducing the dimension of the system while maintaining the accuracy of some specific input–output map, enabling rapid system simulation and design optimization.

MOR for linear systems developed to a high art over the past thirty-five years, starting with simple modal truncation (trim off the most heavily damped eigenspaces), then progressing to balanced truncation (elegant but expensive Gramian-based projection tailored to the model's input and output) and (standard and rational) Krylov subspace algorithms (interpolate the transfer function and its derivatives).

MOR for nonlinear systems has posed a much greater challenge, with fewer system-theoretic tools available to inspire algorithms. The Proper Orthogonal Decomposition method (sample the system in time, identify the dominant subspace via the SVD, and project) can effectively reduce the dimension, but still requires evaluation of the nonlinearity in the full-dimensional space and thus gives little computational advantage. Over the past decade, (Discrete) Empirical Interpolation Methods have largely addressed this challenge, using elegant ideas from oblique projection and approximation theory. The resulting algorithm is beautifully general, but as a consequence does not take advantage of the structure of the dynamical system.

Further progress hinges on the ability to work within the structure of various nonlinear dynamical systems, combining insight about dynamical behavior with algorithmic sophistication. This approach is now *de rigueur* for linear descriptor systems (differential-algebraic equations), where the reduced model must respect the subspace on which the solution evolves. More recent work by, e.g., Beattie, Benner, Damm, Gugercin, and colleagues extends this idea, tailoring ROMs for parameterized, bilinear, delay, and stochastic differential equations. Techniques are under development for the general class of port-Hamiltonian systems. *Interpolation* is often they key: design low-dimensional models that match a transfer function at optimal points in the complex plane.

The method—not the machine

Bengt Fornberg

It used to be said that improvements in scientific computing capabilities originate about equally from advances in algorithms and in hardware. However, during the last decade or two, much focus (and funding) has been devoted to building heroic-scale supercomputing facilities. Maybe this is partly because clock speed and processor numbers are easy to measure and to show off in lists that are widely published, with national and world records falling incessantly. However, the largest systems require inordinate amounts of both power and infrastructure, and they also become obsolete very quickly. For example, IBM's much celebrated Deep Blue, which won against world chess champion Garry Kasparov, had much less computing power than an iPhone. The power of today's large systems will soon be found in desktop units. Unless there is some genuinely revolutionary development (such as optics or quantum processes replacing electronics), increasing parallelism and minimizing communications will continue to be needed, but algorithmic opportunities are otherwise as widely open as ever.

Surprisingly, also private industry (where there is an economic incentive for cost-efficiency) often perpetuates 'legacy algorithms'. One example can be found in seismic exploration for oil and gas. Finite differences (FD) on regular grids for modeling wave propagation were updated from 2nd to 4th order in the 1980's. Now, 20th order FD are widely used for exploration production work, but material interfaces (providing essential reflected data) are still mostly treated to 1st order only. Computations rely on extreme refinement to keep errors at a tolerable level. This approach has been fine-tuned on massive systems to hold its own against (more up-to-date) FEM and DG. The room for algorithmic improvements is vast. Much the same holds for geoscientific simulations of the Earth's systems, such as climate, weather, etc.

Bottom line: What matters in the long run is: The Method—not the Machine.

Return to the days of Ptolemy

Anne Greenbaum

With the huge amounts of data now publicly available, privacy is a thing of the past. In the coming years we will learn more about what can and cannot be predicted through analysis of data, without necessarily formulating a realistic scientific model. Ptolemy was able to accurately predict planetary motions in this way; so far, no one has been able to accurately predict the stock market. There are attempts to predict the weather, our purchasing preferences, etc., etc., all through the analysis of data. The coming years will be interesting in determining which phenomena are and are not amenable to prediction through such pattern recognition techniques. This is good news for the numerical linear algebra community, as these computations require algorithms such as the singular value decomposition and they rely on theoretical analysis connected with, for example, matrix completion problems.

It will be fascinating to see what new things we learn about ourselves through data analysis. Professional athletes have long used films and statistics from previous games to hone their playing strategies, and soon such aids may be readily available to amateurs, as computers track our performance in many aspects of a sport, analyze the data, and offer suggestions for improvement. Will there come a day when Amazon sends me a book suggestion, I think, "no that doesn't look like something that I would be interested in," but then, instead of hitting the delete button, I think, "but Amazon has been right so often in the past...maybe they know more than I do about what I will like...I'll buy it."

This is not to say that data analysis will be the only means of understanding our world and ourselves. As soon as it is demonstrated that a physical phenomenon or a behavior pattern can be predicted based on past data, scientists will be looking for explanations in terms of models and equations. This reversal of the standard order of inquiry is already ubiquitous in mathematical proofs. Conjectures are often formed based on computational experiments, and possible steps in the proof are tested numerically before spending too much time trying to prove them.

While scientists and engineers will continue to use computational results as a guide, the general public will be relying on them completely. This is fine as long as nothing breaks but it will be an added source of stress in our lives when gadgets that we do not understand start doing things that we do not expect. It may spark a "back to nature" movement in which some decide to abandon all modern "conveniences" to live off the land and their own wits.

Mixed precision computations

Nicholas J. Higham

For the last 30 years, most floating point calculations in scientific computing have been carried out in IEEE double precision arithmetic, which provides the elementary operations of addition, subtraction, multiplication, and division at a relative accuracy of about 10^{-16}.

We are now seeing growing use of mixed precision, in which different floating point precisions are combined in order to deliver a result of the required accuracy at minimal cost. Single precision arithmetic is an attractive alternative to double precision because it halves the costs of storing and transferring data, and on Intel chips the SSE extensions allow single precision arithmetic to run twice as fast as double.

Quadruple precision arithmetic is supported by some compilers, and it can be implemented in terms of double precision arithmetic via "double-double arithmetic". Arbitrary precision floating point arithmetic is available through, for example, the GNU MPFR library, the mpmath library for Python, the core datatype BigFloat in the new language Julia, or the Advanpix Multiprecision Toolbox in MATLAB.

Here are some of the ways in which extra precision is currently being used. Iterative refinement, in its traditional form, improves the quality of an approximate solution to a linear system via updates obtained from residuals computed in extra precision. When an algorithm suffers instability it may be possible to overcome it by using extra precision in just a few, key places. This has been used recently in eigensolvers and for matrix orthonormalization.

Any iterative algorithm that accepts an arbitrary starting point can be run once at a given precision and the solution used to "warm start" a second run of the same algorithm at higher precision.

When one is developing error bounds or testing algorithms, one needs in principle the exact solution. In practice, a solution computed at high precision is usually adequate. As the relative costs and ease of computing at different precisions evolve, due to changing architectures and software, as well as the disruptive influence of accelerators such as GPUs, we will see an increasing development and use of mixed precision algorithms. In some ways this is analogous to the increasing interoperability of programming languages (illustrated by C++, Julia, and Python, for example): one uses the main tool (precision) one would like to work with and brings in other tools (precisions) as necessary in order to complete the task.

Sharing the code

Randy LeVeque

A positive development in numerical analysis and scientific computing is the increasing interest in archiving and sharing computer programs that are an integral part of research publications. In mathematics it is unthinkable to publish a theorem without its proof, and mathematicians recognize the need to spend some time cleaning up the proof (and perhaps fixing errors in it) in the course of preparing a manuscript. Unfortunately we have not had analogous expectations in relation to code written to test algorithms.

The "reproducible research" movement is attempting to address this and is an important trend, particularly since an increasing number of journals and funding agencies are starting to require it. And it is not only much easier than in the past but also much more fun, due to improvements in technology for sharing code, including for example open source repositories such as GitHub, virtualization and cloud computing platforms, and browser-based notebooks for exposition.

Of course some people have been doing this for decades, and I think the career of Nick Trefethen serves as a good exemplar. He has always enjoyed polishing his code to make it accessible and informative, in research paper and exposition as well as in his software packages from SCPACK to Chebfun. To choose just one example from his oeuvre, try to imagine "Spectral Methods in Matlab" without the Matlab. It would still be a valuable book, clearly presenting some of the mathematical theory with nice figures illustrating the mathematics and numerical algorithms, but without the code to learn from and experiment with, I believe it would have had far less impact. We should all take such pride and pleasure in sharing our code!

A new modeling, simulation and optimization paradigm

Volker Mehrmann

Modern key technologies require Modeling, Simulation, and Optimization (MSO) of complex dynamical systems. Most real world systems are multi-physics systems, which form a network of components with different accuracies and scales. The models are often generated automatically (via modeling languages such as modelica) and coupling is often done on the solver level via simulator coupling. These developments, which will continue to increase and become the standard in Industrial and Applied Mathematics as well as Scientific Computing, lead to extreme challenges for the numerical methods in approximation, simulation, optimization, and control. Furthermore, the analysis and quantification of errors and uncertainties is very much lagging behind. This dilemma can only be resolved via a stronger interdisciplinary cooperation, where modeling, analysis, as well as numerical, control and optimization techniques go as much as possible hand in hand. A new holistic and integrated MSO paradigm is needed, where models for technological processes and products are modeled in a network of components which themselves present a hierarchy of different approximation levels so that the errors in different components can be adapted to achieve the required accuracies on the global level.

This requires good error and stability (condition number) bounds or estimates for all components, which in many cases, where these are hard to determine, may have to be created via statistical methods. While such detailed error bounds and estimates are well established for many (linear) algebra problems and classes of ordinary differential equations, in the context of partial differential equations or coupled systems of (partial and ordinary) differential and algebraic equations such analysis is only in the beginning. The natural approach seems to be the extension of classical backward error analysis which allows to decouple perturbations in the model and the data from those in the numerical methods and the implementation on the computer implementation. To incorporate this into adequate software implementations, again requires intensive interdisciplinary cooperation.

High dimensionality—a new direction for numerical analysis

Ian H. Sloan

I contend that high dimensional problems will be an important part of numerical analysis in the future.

If you want to be difficult, you might say that high dimensional problems are not a new direction, because they were already initiated by Norbert Wiener back in 1938, through his inscrutably titled paper “The homogeneous chaos”. From a modern point of view that paper has nothing to do with chaos. Rather, it is devoted to a multivariate polynomial method for approximating some physical quantity as a function of many independent stochastic variables. The numerical analysis of high dimensionality was further advanced in the 1950s and 1960s through the work of the number theorists Sobol, Hlawka and Korobov, who (without any help from numerical analysts) initiated the subject of Quasi-Monte Carlo (or equal-weight) integration in many dimensions. So the topic is in some sense not new, and not a new direction.

But high dimensional problems have until now been something of a minor theme in the world of numerical analysis. I think that this is destined to change.

A major driver of change is the current “hot topic” of uncertainty quantification. There are now active journals in this area, such as the new Journal of Uncertainty Quantification (a joint project of SIAM and the American Statistical Association) and the International Journal of Uncertainty Quantification. My observation is that a good proportion of papers in these journals are currently devoted to problems with moderate to high dimensionality. The prototype of all such problems seems to be that of Darcy flow through a porous medium. A random field, modelling perhaps the permeability of a oil field, is in a very natural way described by an infinite number of scalar random variables – that is, it gives rise to an infinite-dimensional problem, and it may well need a very large finite number of random variables to obtain a good approximation. Some buzzwords for the current crop of numerical methods are polynomial chaos and generalized polynomial chaos, stochastic Galerkin, stochastic collocation, sparse grids, multi-level Monte Carlo, Quasi Monte Carlo, multi-level Quasi-Monte Carlo, and so on.

Why now? That is, why has this come to the forefront only in recent years? Of course the answer is that only now is the computing power available to tackle realistic problems with moderately high dimensionality in a meaningful way.

So why do I say that such problems will be of increasing interest in the future? Because such problems are inherently truly hard, and suffer from the famous “curse of dimensionality”. The present reality is that in practice we are tackling only simple versions of such problems – for example, allowing only random fields with “finite-dimensional noise”, small variance and long correlation lengths. As computers become more powerful some areas of numerical analysis may be effectively tamed, but high dimensional problems never will be fully tamed: there are always more difficult versions waiting in the wings. High dimensional problems are here to stay.

Master equations

Shev MacNamara and Gil Strang

Master equations are blessed with an impressive name. They are linear differential equations

$$\frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}t} = \boldsymbol{A}\boldsymbol{p}$$

for a probability vector $\boldsymbol{p}(t)$ (with nonnegative components that sum to 1). The matrix $\boldsymbol{A}$ has nonnegative off-diagonals, and zero column sum. The Master Equation governs the continuous time evolution of the probability distribution of a Markov process with discrete states. The probability of being in state j is $\boldsymbol{p}_j$, and $a_{ij}\mathrm{d}t$ is the probability for the state to change from j to i in a small time interval $\mathrm{d}t$. The solution (formally) comes from a matrix exponential, $\boldsymbol{p}(t) = e^{t\boldsymbol{A}}\boldsymbol{p}(0)$.

An example is the two-way reaction $A+B \rightleftarrows C$. Starting with 4 molecules of A and of B, the five possible states have $4, 3, 2, 1, 0$ molecules of A and B, and $0, 1, 2, 3, 4$ molecules of C. The master equation matrix is

$$\boldsymbol{A} = \begin{pmatrix} -16 & 1 & 0 & 0 & 0 \\ 16 & -10 & 2 & 0 & 0 \\ 0 & 9 & -6 & 3 & 0 \\ 0 & 0 & 4 & -4 & 4 \\ 0 & 0 & 0 & 1 & -4 \end{pmatrix}.$$

This matrix appears as a block in the matrix for the *Michaelis–Menten* reactions: $S+E \rightleftarrows C \rightarrow P+E$. An enzyme E converts a substrate S to a product P, via an intermediate complex C (Gunawardena, *FEBS J*, 2014). This block matrix has interesting spectral and *pseudospectral* properties, which are closely related to those of a finite difference matrix for convection-diffusion. Eigenvalues describe time scales and they are *all real*: $\boldsymbol{A}$ can be symmetrized by a diagonal matrix in $\boldsymbol{DAD}^{-1}$. The Michaelis-Menten matrix also has all real eigenvalues, because it is constructed from bimolecular blocks. However, naïve use of `eig`($\boldsymbol{A}$) returns complex eigenvalues!

This example illustrates key features that come from (modern) chemistry and physics. Numerical mathematics—*when it is driven by real applications*—will have to deal with these features:

1. Probabilities rather than certainties
2. Trajectories and sample paths rather than one path
3. Simulations (like Monte Carlo) from which a histogram recovers $\boldsymbol{p}(T)$.

It is not the nice world we might want. But it is the real world. Single-molecule technologies such as the Dimple Machine ("Mass action at the single-molecule level," *J. Am. Chem. Soc.*, 2012) make it possible to watch molecular reactions *one at a time*, and a Master Equation describes those experiments extremely well.

Mathematically, we are *sampling* instead of solving. The dimensions are huge, partly because $S + E \rightleftarrows C \rightarrow P + E$ can go backward from C arbitrarily often before it goes forward to P and stops. Ease of Monte Carlo simulation is a killer advantage in high dimensions.

The alternative to Monte Carlo is to compute a matrix exponential. Perhaps it is surprising to learn that this is usually too difficult. Moler and Van Loan's SIAM Review classic "Nineteen dubious ways to compute the exponential of a matrix" comes to mind. For N species with possibly $M-1$ molecules each, the matrix can have size M^N. Still, there is hope. The matrix is sparse, and its structure is simple. A successful computational strategy would be worth the effort because Master Equations have important applications.

Two final notes:

When concentrations are high, the reactions are not usually modeled by a master equation. Instead, three ODEs form a mass-action kinetics model: $\mathrm{d}[A]/\mathrm{d}t = -[A][B] + [C]$, $\mathrm{d}[B]/\mathrm{d}t = -[A][B] + [C]$ and $\mathrm{d}[C]/\mathrm{d}t = +[A][B] - [C]$. Linear algebra has a role to play in a large class of these mass action models, both in the deterministic and stochastic versions ("Product form stationary distributions for deficiency zero chemical reaction networks," *Bull. Math. Biol.*, 2010). The solution to this deterministic ODE is approximately the mean of the stochastic master equation.

Stochastic *spatial* reaction diffusion models give rise to even bigger computational challenges (`www.StochSS.org`). A temporal master equation governs the reactions inside each subvolume, and diffusion between subvolumes is another reaction. Only Monte Carlo is used because the state space has high dimension. The Elf lab (*SCIENCE* 336, p.595, 2012) shows single molecules binding to DNA to turn a gene on and off, in a living cell. The molecule combines 3D diffusion in the volume of the cell and 1D diffusion on a DNA strand to find its target gene. Spatial models are increasingly important to understand stochastic events.

And stochastic events have become an essential way to understand our world.

John Francis and the impact of numerical algorithms

Andy Wathen

A few weeks ago I was able to don the academic dress of my PhD—gown, hood and 'beefeater' hat—for only the second time in my life; the first since I was awarded the degree thirty years ago. This second occasion was far more special: I had been invited to The University of Sussex for the degree ceremony at which John Francis was awarded an Honorary doctorate for his invention in the early 1960s of the QR algorithm—the standard algorithm for the computation of eigenvalues of a matrix.

This is one of the top 10 algorithms of the 20th century, and most in the know, consider it one of the few truly 20th century algorithms amongst the core algorithms of linear algebra—neither Euler nor Gauss got a look-in. Over the past 50 years it has become a cornerstone of so many applications it is just remarkable how it took so long for us to get to the point of this celebration—but that is another (fascinating) story. The symmetric QR algorithm is the most perfect and beautiful mathematics. We were acknowledging the work of a young Englishman, now in his eighties, for this, his gift to all of us.

Sussex did it well: a bright yellow gown so that John would stand out amongst the assembled academics and graduands in the packed Brighton Dome, speeches of explanation for the lay person from colleagues and from John Francis himself about what it was all about. Some of the graduands jostled to shake John's hand as he left the stage, to acknowledge a pioneer of the computational age. It was a perfect day on the south coast, and this being England, it was followed, of course, by lunch.

I was expecting some mild interrogation from family and friends as one of the outside 'experts' as we tucked into fresh seafood and quaffed a crisp wine wine, but I was unprepared for their enquiry. The one question that nobody could understand it seemed was not about how John came to invent the algorithm, nor, not surprisingly, how it worked, but only how on earth did John fail to become a millionaire! Surely this seemed such a fundamental contribution that with even the tiniest royalty on each commercial application of the algorithm, he would have amassed great riches! I could not deny it, but no such…

I reflected on the way home that it seems likely there will always be this tension between the artistic ingenuity and inventiveness that many of us would like to achieve in our research—to any small part of the degree achieved by John Francis—and the utility that others will make of our efforts.

New directions in the teaching of numerical computing

André Weideman

I coded my first Gaussian elimination using a thick stack of punch cards. Today, that involves exactly three characters in MATLAB. Back then, finding a research paper involved a trip to the library. Today, I lift exactly one finger. As an instructor of numerical analysis and scientific computing, I keep asking myself what to take and what to leave from this veritable buffet of technological advances.

Teaching my subject in a computer laboratory has been my mode of instruction for more than two decades. Quadratic convergence, I feel, is much better experienced firsthand than second. Symbolic software has been with us for a while, but these days WolframAlpha understands natural language (most of the time). While I firmly believe students should be able to integrate by parts by hand, symbolic software can do that six term Taylor expansion without jeopardizing understanding. With Google and Wikipedia available, do we even need that expensive text book (the one where only the exercise numbers have been shuffled in the new edition?) Come to think of it, with a lecture from that famous MIT professor only a video link away, have I myself become redundant?

Along with these opportunities for enhanced instruction come hardware and software advances that touch the very heart of our subject. What will the new algorithms look like and what should we teach? Should processor technology be part of the course? I hear a flop is practically free these days, and memory access is what's important. Should those beautiful flop-counting diagrams in Trefethen & Bau be replaced in the next edition of the book? If so, with what?

Questions are many and answers are few. What is clear is that the classroom (or future equivalent) as well as the course content will look very different two decades from now.