

Development of mathematical methods for Exascale and beyond.

Our approach is based on the basic observation that to design new methods, we have to think about limits, or more precisely about pushing this limit far ahead of us. This is exactly what happened for finite elements where significant progress was made after mathematical analysis allowed us to let the size of the meshes go to zero. This stimulated research on many aspects like mesh refinement techniques and, with the size of the discretization growing, groups working on the solution of linear and non-linear systems have been facing new problems. These new questions have stimulated progress in disciplines like the solution of linear systems with direct or iterative methods, and exploiting emerging preconditioning techniques. So the experience that we get from this slightly embellished example is that mathematics should explore the infinity to let practitioners be confident and efficient in developing new methods.

With such tremendous computing power now shortly at hand, what should be the mathematical techniques that we could foresee as being called to play an important role in future computations? With infinitely many processors, **Monte Carlo techniques** are of course appealing, because they often involve independent computations that scale ideally on massively parallel computers. The main problem with these techniques is related to their slow convergence that has prevented practitioners from using them more widely. But times are changing. In meteorology, in history matching, people are now considering the use of ensemble methods, and research is conducted in most simulation centres to combine these techniques with more traditional approaches, with the help of new algorithmic developments. So called embarrassingly parallel methods also occur in methods based on **Cauchy integral methods** for computing matrix functions, including eigensystem computations. These methods were discarded for a while, because their speed of convergence was considered sometimes disappointing, and also because they were very expensive in terms of computational effort. They certainly need to be reconsidered for future directions. Similarly, approaching **integrals by sampling** as done, in particular, by filters for parameter estimation is certainly appealing for the very large computers we are considering here. But for making all the algorithms we just named as easy to use and reliable as existing ones on large scale problems is a challenge. The obstacles are very important and require people that are well trained in traditional numerical analysis and in statistics, with a good knowledge of computer science. A challenge for education and research.

For our theoretical infinitely large computers having executions that present ideal scaling properties when the number of processors increases is a must. Why run on very large machines if not to better control execution time and possibly power consumption? The urgency for doing this is strengthened by the advent of modern computers. But there are directions emerging and therefore hopes. **Exploiting the structure** is an important idea ; here computing power can be used to explore parameter spaces and get useful **reduced models**, that should be as structured as possible, again to minimize the amount of data needed to represent them. Good choices of basis, and good mapping functions have to be found in this area. This is tightly related to recent work that has been done in the domain of direct solvers, where **block compression** (i.e. representing blocks by low rank matrices) is observed to significantly improve the performance of the solvers on academic problems. But much work is still required to get compression based direct solvers that would exploit dynamically these techniques whenever possible with well controlled accuracy.

Using **different levels of precision** is tempting, but controlling the computational errors is a challenge, and mathematical study of truncated algorithms in the presence of roundoff errors should be pursued. Generally speaking linear algebra is an important place to invest efforts, since many numerical methods for optimising and solving PDE's are based on it, and will immediately benefit from any progress.

Of course on very large computers, optimising the communication is also absolutely unavoidable. Communications are related to coupling that exist in the problems formulation. Good mathematical methods have to approximate intermediate quantities using less coupled computations. Future algorithms will consist in a hierarchy of methods composed of embedded iterations that will at every stage optimize the communications to get just the information needed from other computations. This is already what **hybrid direct iterative methods** are doing for some problems. But the investigation should be continued and, for instance, results already obtained by appropriately combining multigrid solvers and direct solvers for solving the Helmholtz equation are encouraging.

Having algorithms that make a parsimonious use of memory and communications is good, even necessary. But this does not always translate to high performance. In a homogeneous computer, we need to have an even work load among all computational units, so that the machine is fully busy. **Chaotic iterations**, that were popular several years ago are worth reconsidering. They reduce the time spent in synchronization among processors. But their analysis is far from obvious, and their provable range of applicability is still restricted to operators with strong mathematical properties. Getting a good work balance between processors will become even more challenging for large computers when the data layout will change in runtime, due to mesh refinement techniques, or due to a bifurcation of a time dependent dynamical system that dramatically changes the underlying physics to be considered (such as a change of phase, apparition of chemical species,...)

The list of mathematical methods that should be (re)considered in the future is very large, and we are here just proposing a subset, strongly biased by our own experience in applied mathematics at CERFACS. Another probably more exotic topic of investigation is **round-off error propagation** on infinitely many computers. Do round-off errors naturally behave nicely as for usual computations? Is the propagation the same when parabolic, hyperbolic or elliptic PDE's are solved with huge processor counts? Do we need to consider more accurate arithmetic than the standard 64 bit representation of floating point numbers for solving the coming large scale problems? Is it possible to get rid of some problem dependent constants in error bounds and get tight bounds for widely used numerical methods for large scale problems? A probably even more iconoclast question would be related to the use of **other algebraic structures** (such as Dickson algebra) than the set of floating-point numbers to get more efficient, and/or more accurate computations.

Finally it is also very clear that the advent of more powerful computers will itself stimulate the emergence of more difficult problems to solve. **Stochastic, non-Gaussian, nonconvex, huge-scale nonlinear problems** will have to be considered, and we must expect and develop mathematical methods for tackling them.

Iain Duff and Serge Gratton

7 May 2013