Trends in Scientific Computing

Olivier Pironneau
University of Paris VI, INRIA and IUF*

September 29, 2000

1 Introduction

I have been asked to speak about the future of Scientific Computing. I will restrict the talk to the numerical solution of partial differential equations but we must remember that it is only a part of scientific computing which for instance leaves aside ordinary differential equations.
The field is 30 years old more or less and it is indeed an interesting period for changes. The military lobby was a major contractor of scientific computing but since the end of the cold war this is no longer true. Even complex problems like the simulation of plasma by laser are not steering waves of enthusiasm (and money) outside their home institutions. It is also necessary to acknowledge the fact that many of those who witnessed the birth of the field, and I am one of them, are over fifty years of age and perhaps rusty when it comes to the learning of new tools. But most of all, it seems to me, is the fact that scientific computing does not rely on super-computing as much as it did before and that 95% of the problems can be solved on Linux or Windows or other cheap machines. This has helped the software companies to distributed good PDE solvers but it has also caused many industries to close down their computing lab and relocate the numerical analyst inside the engineering departments.

2 Old Problems

There are still very many unsolved problems. The famous Navier-Stokes equations and its turbulence is still very hard to reproduce numerically and when it is done, it is by models which are mathematically unsound. Perhaps Large Eddy Simulations will improve the situation.
By in large problems with multiple scales are very hard. Propagation in random media is one and homogenization has done some break-through, but percolation, enhanced diffusion... are still open numerical problems. In chemistry there are many phenomena which happen at the pico-second time scale and yet produce tangible results at the level of the minute; these would require very long computations if nothing is done theoretically!
Multi-phase flows are also very complex still. Part of it is due to the modeling though, and, like turbulence, the problem is unsolved at the theoretical level.

*pironneau@ann.jussieu.fr

But even linear problems like the Maxwell equations can be very difficult when the aspect ratio between the wave length and the objects are large. Yet it is a pleasure to see that improvement are made at a growth rate comparable to the one of computing power thanks for instance to algorithm like the fast multi-pole method.
But if on one side some problems are still hard, on another many have become routine work and it is possible to improve the design of the system they simulate by optimization. We have witnessed lately the birth of non-deterministic methods in optimization and I believe that this is just the beginning because global and multi-criteria optimizations are necessary in practice.

3 Environmental Sciences

At present much energy goes into the simulation of environmental problems, Meteorology and climatology which requires a coupled ocean-atmosphere modeling are very challenging and still lacking computer resources.
Urban pollution is an interesting problem which requires much chemistry in addition to meteorology. They are currently being coupled with road traffic models and I believe that they will soon be reliable.
It may not be the case of waste management yet. In particular safety assessment of nuclear waste repository vaults by simulation is a major issue. It is a multi-physics simulation with thermo-hydrology, mechanics and chemistry plus some nuclear engineering. The radioactive elements have to be followed for hundreds of thousands of years underground in factured layers.
A new field is emerging: Risk monitoring with the help of satellite images and other sensors with applications to earth-quake, land-slide, floods, Tsunami, forests, fish...
There is a difficulty in all these fields which is related to data acquisition. Many of them require data assimilation. Consequently reliable data are expensive and collaboration between research and institutions have been slowed down by this point.

4 New Tools and New Applications

Scientific Computing is a very lively field still. On the theoretical side there are many new ideas such as wavelets, hierarchical basis, sparse grids, fictitious domain methods...
On the computing side, we have new architectures with emphasis on shared memory multiprocessors at the moment but also new software developments line Matlab, C++,Java, Blitz, Pooma, Aldol-C, Odyssee...

The military-industrial complex is no longer the leading contractor but fortunately there are new customers such as mobile phone companies (optimal location of relays) the Medical industry with Molecular biology, Neuro-sciences and nano-technologies, banks with mathematical finance, and even the world wide web with java interfaces and distributed computing.

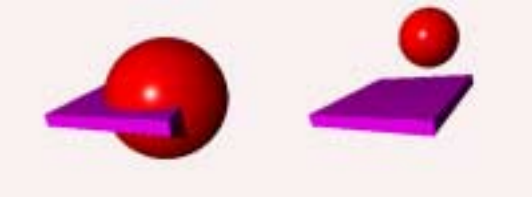

Figure 1: In this rendering there is no intersection computed, it is due to the z-buffer algorithm.

Looking at the rate at which the communications increase some people say that computing power will soon be managed like power plugs, namely not a problem for the user but for the provider. If this is so, distributed computing over the web has a good future.
But most interesting to me is the rise of Virtual Reality with applications to simulators (medical in particular), films and computer games. At INRIA-Grenoble there is a project for simulation tools for VR; for instance a simulator for surgeons operating on the liver requires the simulation of a 3D elastic (as a first approximation) structure with large displacement to render realistically the reaction of the liver to the scalpel.

5 Computing for VR

There are hundreds of applications in architecture also such as the sizing and design of air conditioning in a building which exists only in VR. In fact the first problem of this kind that I saw was solved by B. Lohner when he was asked to simulate the explosion in the parking lot of the world trade center. Treating 450 parked car is easy in VR but very hard in engineering CAD systems.
What is different in VR systems is that solids are not intersected and so there is nowhere a data structure for the surfaces in the scene. Figure 5 shows a sphere and a box which look like the intersection has been computed but it is not so. The illusion is produced by the z-buffer algorithm which check that a voxel (point in 3D) is in front of previously displayed ones before changing the screen pixel. Therefore to describe the scene requires little effort; in VRML [1] it is done by

```
Separator {
    DEF sphere1 Separator {
             Translation { translation 1 0 0 }
        Sphere { radius 2 }
    }
    DEF Cube1 Separator {
          Transform {  rotation -20 30 0}
          Translation { translation -15 30 0}
        Cube { width 4  height 0.4 depth 4}
    }
 }
```

POV-ray [6] is a ray-tracing program which is older than VRML but also more

Figure 2: This object, produced with POV-Ray will be used as a reflector in an electromagnetic simulator

sophisticated . It runs on PCs Macs and Linux. It is one of the first language using CSG (Constructive Solid Geometry). For example Figure 5 is produced by

```
#include "colors.inc" #include "metals.inc"
 camera { location <-1, 0, -3>
   look_at  <1, 1,  2>  }
 light_source { <2, 4, -3>
   color White}
 #declare altere = union{
     cylinder {<-1.5,0,0>
           <1.5,0,0>, .35}
     sphere {<-1.5,0,0>, 0.5}
     sphere {<1.5,0,0>, .5}}
 union {
     object { altere rotate z*90 }
     object {altere scale 1}
     object { altere rotate y*90 }
     sphere{<0,0,0>, 0.6 }
 }
```

To compute with VR data we have suggested in Del Pino et al[8] to proceed as follows
Step 1. Parse the VR language source code to create a function **in(x,y,z**) which returns the index of the object to which **(x,y,z)** belongs.
Step 2. Triangulate each elementary object.
Step .3 - Build a proper triangulation of the surfaces and call a 3d Mesh Generator or use the Fictitious Domain Method.

6 The freefem3D Project

As the input of data in 3D is a real problem we will use POV-Ray and or VRML. At present automatic mesh generators are not free so we will use the Fictitious Domain Method. Then High Tech linear iterative solvers are needed for speed. The results will be displayed by POV-Ray thanks to a patch [7].

Figure 3: Solution of the exterior problem for Maxwell equations with perfect reflection on the object

Parallelism is also needed for speed. It can be implemented via CSG. For instance if the domain of the PDE is

$$\Omega = \sum \Omega_i \text{ we use } u = \sum u^i$$

so that

$$-\Delta u = f \quad \Leftrightarrow \quad -\Delta u^i + \lambda^i = f, \quad \sum \lambda^i = 0, \quad u^i \in H_0^2(\Omega_i)$$

This sort of ideas have been tested in 2D in J.L. Lions et al [3][4][5] and the references therein.
Currently a 2D version is on the web at http://www.ann.jussieu.fr/∼pironneau; it uses a language like

```
border Gamma(t=0,2*pi){ x = cos(t); y = sin(t);  };
mesh   Omega = buildmesh(Gamma(60));
femp1  f =(10/pi)*sin(sin(x*y));
varsolve(Omega) a(u,v)  with {
   a = int(Omega)(dx(u)*dx(v)+dy(u)*dy(v)-f*v)
       + on(Gamma)(v)(u=0);
} plot(u);
```

The 3D version in under development but a 3D test computation was done in an afternoon at the University of Jyvaskyla (Finland) by coupling with a Fictitious Domain code [2] for Maxwell equations (figure 6)

References

[1] J. Hartman and J. Wernecke, "The VRML 2.0 Handbook" Addison-Wesley 1996.

[2] Y. Kuznetsov, and K. Lipnikov, "3D Helmholtz wave equation by fictitious domain method", *Russian J. Numer. Anal. Math. Modelling* **13** (1998) 371–387.

[3] J.L. Lions, O. Pironneau, Algorithmes parallèles pour la solution de problèmes aux limites, C.R.A.S., 327, pp 947-352, Paris 1998.

[4] J.L. Lions, O. Pironneau, Sur le contrôle des sytèmes distribués. C.R.A.S., 327, pp 993-998, Paris 1998.

[5] J.L. Lions, O. Pironneau, Domain decomposition methods for CAD. C.R.A.S., 328, pp 73-80, Paris 1999.

[6] A. Wardley, Persistence of Vision, `POV-Ray` in `http://www.povray.org/`.

[7] R. Suzuki, A patch to `POV-Ray` for iso-surfaces. In `http://www.public.usit.net/rsuzuki/e/povray/iso/index.html`.

[8] S. Del Pino, E. Heikkola, O. Pironneau and J. Toivanen : A Finite Element Method for Virtual Reality Data. C.R.A.S. May 2000.