Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computer Physics Communications 154 (2003) 89–97
Computer Physics Communications
www.elsevier.com/locate/cpc

A simple spectral algorithm for solving large-scale Poisson equation in 2D

X. Thibert-Plante [a], D.A. Yuen [b,*], A.P. Vincent [a,c]

[a] *Centre de Recherche en Calcul Appliqué, CERCA, 5160 Boulevard Décarie, Bureau 400, Montréal, QC, H3X 2H9 Canada*
[b] *Department of Geology and Geophysics and Minnesota Supercomputer Institute, University of Minnesota, Minneapolis, MN 55455-0219, USA*
[c] *Département de Physique, Université de Montréal, C.P. 6128, Succ. Centre-Ville, Montréal, QC, H3C 3J7 Canada*

Received 16 August 2002

Abstract

We show that it is possible with easy-to-program algorithms to reach spatial resolutions of the order of 10^8 grid points for computing the electric potential on 2D periodic lattices, such as the Si(111)7 × 7 surface. We have used a spectral Fourier technique and parallelized FFTs with OPEN_MP on SGI machines. This method can be easily extended to 3D.

Keywords: Poisson equation; FFT-parallelization; OPEN_MP; Crystal lattice

1. Introduction

The Poisson–Laplace–Boltzmann equation has universal applicability, ranging from gravity [20] and material physics [1,5] to biochemical studies [10,18,22–24].

The simulation or imaging of nano to meso-scale phenomena on crystal lattices requires very high resolution. An example is given by the surface potential of inorganic crystals. Surface topography can be mapped out by using Atomic Force Microscopy (AFM), Kelvin Probe Microscopy and related techniques. Strong correlation between the topography and potential has been noticed [12]. Si(111)7 × 7 is a very complex and fascinating crystal, although the actual surface actually consists of three layers [2]. Defects have been observed on surface maps of 60 nm × 60 nm [9] using AFM. Phonon propagation is another example of large scale patterns, having been recently observed on various surfaces such as LiF and TeO2 surfaces at least of the order of 150 × 150 μm^2 [21]. Starting from a charge distribution field, to compute surface potential associated with a map would require 8^2 grid points for each atom and thus of the order of 1 million grid points in each direction!

There are many numerical techniques for solving the Poisson–Laplace–Boltzmann equation: Monte Carlo technique [25], multigrid [26], multipole [7], Fourier spectral [3], splines [15], finite volumes [11].

* Corresponding author.
E-mail addresses: thibert@cerca.umontreal.ca (X. Thibert-Plante), davey@msi.umn.edu (D.A. Yuen), vincent@astro.umontreal.ca (A.P. Vincent).
URLs: http://www.cerca.umontreal.ca (X. Thibert-Plante), http://www.msi.umn.edu/~davey/ (D.A. Yuen), http://www.phys.umontreal.ca (A.P. Vincent).

0010-4655/$ – see front matter © 2003 Published by Elsevier B.V.
doi:10.1016/S0010-4655(03)00283-2

A review is given in [8]. In this paper we focus only on the Poisson equation with applicability in surface chemistry and nanosciences. Periodic boundary conditions on a 2D Cartesian plane have relevance in modelling a crystal lattice. In this case the Fourier spectral technique provides the fastest and most accurate solution. The Laplacian operator is diagonal and the algorithm is straightforward. Spectral filters [6] can be used for the treatment of local discontinuities (here defects or meso-scale faults in crystals) whereas the case of a globally non-periodic lattice (large inclusions) can be processed with global Fourier windows [16].

In the next section we describe the algorithm for parallelizing 2D FFTs loops and for solving the 2D periodic Poisson equation. Next we show how with a modest shared-memory computer, such as the SGI with 16 processors one can easily reach resolutions up to 20,000 grid points along each direction. We illustrate this capability with an example taken from the treatment of numerically constructed lattice of Si(111)7×7 with random defects. In the last section we discuss open problems such as generalization to elliptic equations with variable coefficients, extension to three dimensions or surfaces with internal boundaries.

2. Algorithms

2.1. A simple 2D parallel FFT

The two-dimensional real-complex symmetric (RCS) Fourier transform is:

$$f(x,y) = \sum_{k_x} \sum_{k_y} \hat{f}(k_x,k_y) \mathrm{e}^{\mathrm{i}k_x x} \mathrm{e}^{\mathrm{i}k_y y}, \tag{1}$$

where $\mathrm{i} = \sqrt{-1}$, the summation is taken over $-N_x/2+1 < k_x < N_x/2$ and $-N_y/2+1 < k_y < N_y/2$ with N_x and N_y being the number of grid points in the x and y directions, $\vec{k} = (k_x, k_y)$ is the Fourier wave vector. $\hat{f}^*(\vec{k}) = \hat{f}(-\vec{k})$ is the complex conjugate of $\hat{f}(\vec{k})$. This last Hermitian condition ensures that the sum is real.

We started from a 1D complex-complex Fast Fourier Transform **fttpack** written in Fortran 77 by Paul Swartztrauber of the National Center for Atmospheric Research at Boulder Colorado [19]. At the moment FFT pack may be perhaps one of the fastest FFTs available in the public domain. The algorithm that we developed for computing 2D fully periodic real-complex FFT, while using Swartztrauber's 1D complex-complex FFT as a black box, is shown in Algorithms 1 and 2. Parallelization makes use of the OPEN_MP package (http://www.openmp.org/). We did not parallelize the 1D FFT but put in parallelization only in the loops of our 2D FFT. There are several parallel FFTs available currently in the public domain (e.g., FFTW from MIT: http://www.fftw.org/). Our point is to show that we can use any 1D non parallel FFT without changes and build a parallel 2D or 3D FFTs with extremely simple algorithms, while having surprisingly good performances.

The output arrays of Fourier modes $\hat{f}(\vec{k})$ is shown in Fig. 1 in the space of Fourier wavevectors (k_x, k_y). The Hermitian property

$$\hat{f}^*(k_x,k_y) = \hat{f}(-k_x,-k_y)$$

acts on the first indice k_x here.

The machine we have used is a Silicon Graphics SGI Origin 3800 with ninety six 500 MHz MIPS R14000 Processors with each CPU having a 8 MB secondary cache and a total shared memory size

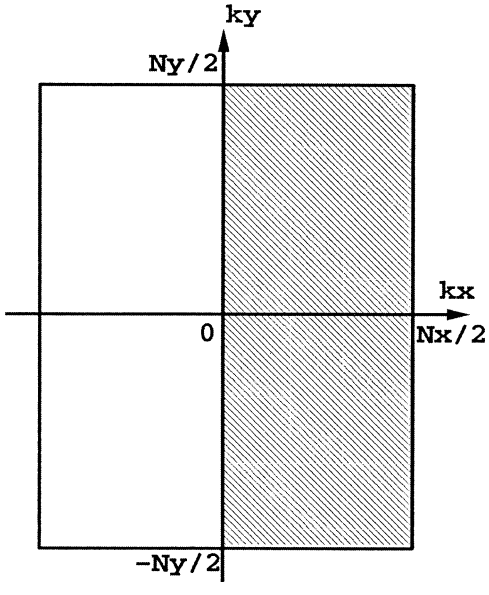

Fig. 1. Output of the real complex symmetric (RCS) FFT in the space of Fourier wavevectors $\vec{k} = (k_x, k_y)$. Hermicity plays on the first component k_x here.

```
Require: real x[N_x][N_y]
  integer i, j, N_x, N_y
  complex Xcomplex[N_x][N_y], tmpH[N_y], tmpV[N_x]
Ensure: x[N_x][N_y] = Real-Complex-Symmetric Fourier transform of x[N_x][N_y]
  OMP PARALLEL DEFAULT(NONE) PRIVATE(i, j, tmpH[ ])
  OMP SHARED(x[ ][ ], Xcomplex[ ][ ], N_x, N_y)
  OMP DO
  for 1 ⩽ i ⩽ N_x do
    for 1 ⩽ j ⩽ N_y do
      tmpH[j] = cmplx(x[i][j], 0)
    end for
    1D Swartztrauber (FFTpack) Fourier transform of tmpH with is own
    copy of the working array
    for 1 ⩽ j ⩽ N_y do
      Xcomplex[i][j] = tmpH[j]
    end for
  end for
  OMP ENDDO
  OMP END PARALLEL
  OMP PARALLEL DEFAULT(NONE) PRIVATE(i, j, tmpV[ ])
  OMP SHARED(x[ ][ ], Xcomplex[ ][ ], N_x, N_y)
  OMP DO
  for 1 ⩽ j ⩽ N_y do
    for 1 ⩽ i ⩽ N_x do
      tmpV[i] = Xcomplex[i][j]
    end for
    1D Swartztrauber (FFTpack) Fourier transform of tmpV with is own
    copy of the working array
    for 1 ⩽ i ⩽ N_x/2 do
      x[2i − 1][j] = real(tmpV[i])
      x[2i][j] = imag(tmpV[i])
    end for
  end for
  OMP ENDDO
OMP END PARALLEL
```

Algorithm 1. Forward Fourier transform.

of 144 GB for the system, which is located at the University of Minnesota Supercomputing Institute in Minneapolis, Minnesota.

2.2. Spectral Fourier Poisson solver

We want to solve:

$$\Delta P = q, \tag{2}$$

where Δ is the Laplacian operator, $P = P(x, y)$ is the potential and $q = q(x, y)$ is the charge density field. In the spectral Fourier space, the solution of the Poisson equation in spectral space takes the form:

$$\widehat{P}(k_x, k_y) = -\hat{q}(k_x, k_y)/k^2. \tag{3}$$

If there is no mean zero mode $\vec{k} = (0, 0)$, a simple division by k^2 is sufficient (Algorithm 3). The entire algorithm consists first in a direct FFT from physical space to spectral space. Then this is followed by a division by k^2 and then an inverse FFT from the spectral space back to the physical space.

Require: real $x[N_x][N_y]$
integer i, j, N_x, N_y
complex Xcomplex$[N_x][N_y]$, tmpH$[N_y]$, tmpV$[N_x]$
Ensure: $x[N_x][N_y]$ = Real-Complex-Symmetric Fourier transform of $x[N_x][N_y]$
$j \Leftarrow 1$
$tmpV[1] = cmplx(x[1][j], x[2][j])$
for $2 \leqslant i \leqslant N_x/2$ **do**
$tmpV[i] = cmplx(x[2i-1][j], x[2i][j])$
$tmpV[N_x - i + 2] = cmplx(x[2i-1][j], -x[2i][j])$
end for
$tmpV[N_x/2 + 1 = cmplx(0, 0)$
1D Swartztrauber (FFTpack) Fourier transform of *tmpV*
for $1 \leqslant i \leqslant N_x$ **do**
$Xcomplex[i][j] = tmpV[i]$
end for
OMP PARALLEL DEFAULT(NONE) PRIVATE(i, j, tmpV[ ])
OMP SHARED(x[ ][ ], Xcomplex[ ][ ], N_x, N_y)
OMP DO
for $2 \leqslant j \leqslant N_y$ **do**
$tmpV[1] = cmplx(x[1][j], x[2][j])$
for $2 \leqslant i \leqslant N_x/2$ **do**
$tmpV[i] = cmplx(x[2i-1][j], x[2i][j])$
$tmpV[N_x - i + 2] = cmplx(x[2i-1][N_y - j + 2], -x[2i][N_y - j + 2])$
end for
$tmpV[N_x/2 + 1] = cmplx(0, 0)$
1D Swartztrauber (FFTpack) Fourier transform of *tmpV* with is own
copy of the working array
for $1 \leqslant i \leqslant N_x$ **do**
$Xcomplex[i][j] = tmpV[i]$
end for
end for
OMP ENDDO
OMP END PARALLEL
OMP PARALLEL DEFAULT(NONE) PRIVATE(i, j, tmpV[ ])
OMP SHARED(x[ ][ ], Xcomplex[ ][ ], N_x, N_y)
OMP DO
for $1 \leqslant i \leqslant N_x$ **do**
for $1 \leqslant j \leqslant N_y$ **do**
$tmpH[j] = Xcomplex[i][j]$
end for
1D Swartztrauber (FFTpack) Fourier transform of *tmpH* with is own
copy of the working array
for $1 \leqslant j \leqslant N_y$ **do**
$x[i][j] = real(tmpH[j])$
end for
end for
OMP ENDDO
OMP END PARALLEL

Algorithm 2. Backward Fourier transform.

```
Require: real p[N_x][N_y], q[N_x][N_y]
  integer i, j, N_x, N_y
Ensure: Poisson solution
  for 1 ⩽ i ⩽ N_x/2 do
    for 1 ⩽ j ⩽ N_y do
      k^2 = k_x^2(i) + k_y^2(j)
      if k^2 > 0 then
        p[2i − 1][j] = −q[2i − 1][j]/k^2
        p[2i][j] = −q[2i][j]/k^2
      end if
    end for
  end for
```

Algorithm 3. Poisson solver.

3. Results

3.1. Performance

The number of arithmetic operations in Fast Fourier Transform is proportional to $N\log_2(N)$. The elapsed wallclock time in seconds is displayed in Fig. 2. As expected, the time scales like $N\log_2(N)$. At a given resolution, the speed-up factor is defined as:

$$SU = \left(\frac{\text{Elapsed Time}(N\ \text{Proc})}{\text{Elapsed Time}(1\ \text{Proc})}\right) \tag{4}$$

and is shown in Fig. 3. For 8 processors, it is about 6 and this represents an appreciable improvement. However, the speed-up saturates above about 16 processors. This is due to an increase of the bus traffic with the number of processors and is particular to each machine.

Memory use is another important consideration to take into account. From $N_x = N_y = 8$ to $N_x = N_y = 2048$ memory increases like N^3, above this value it is proportional to $N_x \times N_y$ (the size of an array). Above $N_x = 2048$, the RAM memory is increased in our case (Fig. 4) because OPEN_MP duplicates the arrays in each processor. The value 2048 is in 32 bits words, which can be fit within the capacity of the secondary cache whose size is 8 Mbytes in the case of the SGI machine. Another limitation was the RAM memory of the machine. It is possible to go to resolutions of

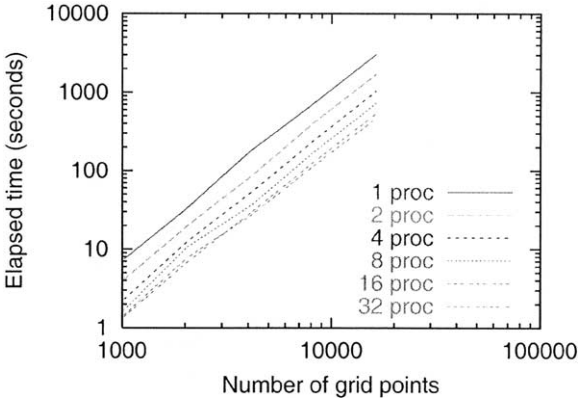

Fig. 2. Elapsed wallclock time (second) versus the number of grid points $N_x = N_y$ is shown here from 1 to 32 processors. The time is proportional to $N\log_2(N)$, as expected with the Fast Fourier Transform.

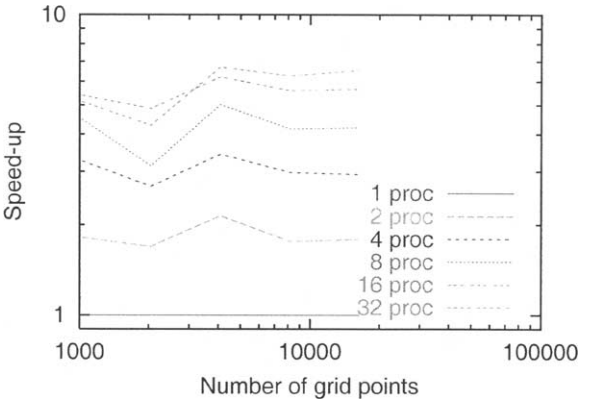

Fig. 3. At a given grid resolution, the speed-up (SU) is improved with the number of processors Nb_{Proc} but saturates around $Nb_{\text{Proc}} \approx 16$. Here, $SU =$ (Elapsed Time(N Proc)/Elapsed Time(1 Proc)).

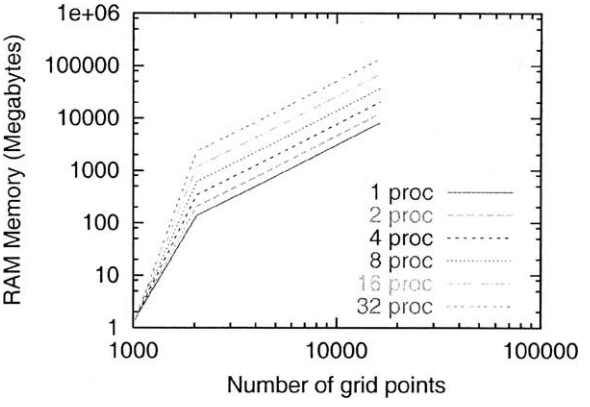

Fig. 4. RAM Memory versus the number of grid points. Up to $N_x = N_y = 2048$, memory increases like N^3, above this value this is proportional to N^2 (the size of an array). The value $N = 2048$ in 32 bit words gives 8 Mbytes, the size of the cache on the SGI.

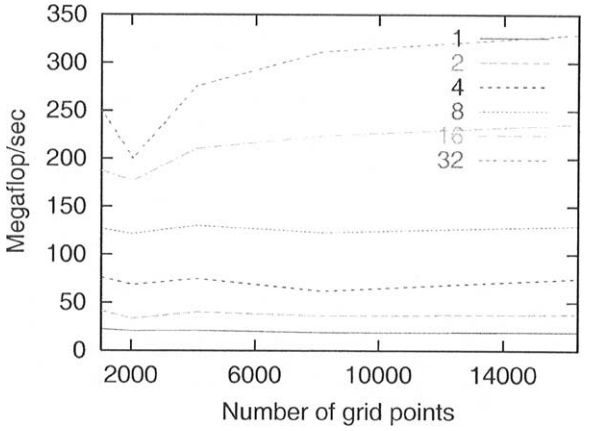

Fig. 5. Performance in Megaflops/sec as calculated with the software "perfex" does not change with the number of grid points. Labels refer to the number of processors.

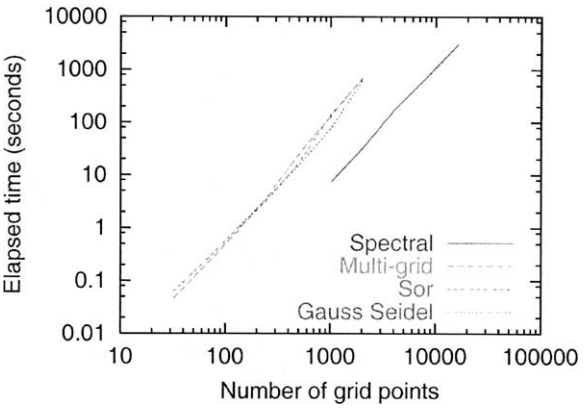

Fig. 6. Comparison between Gauss–Seidel Multi-Grid, SOR and Spectral Fourier method on a single processor of SGI. Elapsed time is shown versus the number of grid points in each direction $N_x = N_y$. The direct spectral Fourier method is about 100 times faster.

the order of the $100{,}000^2$ grid points, provided we do not store the full arrays $P(x, y)$ or $q(x, y)$ in the core but only as one single column or line of the array at a time.

We found that performance has been determined to be about 300 Mflops for 32 processors (Fig. 5) and remains almost unchanged with the resolution. In our simulations, the machine precision we use is 32 bits. This is enough for most applications. As long as only 7 digits of accuracy is required, the spectral method would not introduce additional errors as would finite differences, for instance.

The spectral solver developed here is much faster than the Gauss–Seidel (GS) method, which needs $N_x \times N_y$ iterations to obtain a digit more in improvement of accuracy [16]. We display the comparison on a single processor in Fig. 6. The elapsed time is shown here versus the number of grid points $N_x = N_y$ in each direction. The Gauss–Seidel method is found to be about 100 times slower than the spectral method. The methods Successive Over Relaxation *SOR* and multigrid are not any faster (Fig. 6). Comparison with the multi-grid method has been done with the free software written by David Pruett from James Madison University. Only with the use of a massively parallel distributed memory machine would the red-black Gauss–Seidel become competitive. Although the Gauss–Seidel method has been proposed [26] and would be better adapted to complex geometries and boundary conditions, no known comparison has been made with the spectral algorithm.

Finally, the question of reaching one million of grid points in each direction is an unavoidable step in the near future in order to simulate complex large-scale phenomena in surface chemistry, such as wave propagation involving surface phonons. Here is the outline of an algorithm that would handle a grid of $N_x = 2^{20} \times N_y = 2^{20}$ points. Such a large matrix cannot be stored any longer in core of most systems; this would demand at least a terabyte of core memory. The solution is to decompose the matrix by $Nb_{\text{proc}} \times Nb_{\text{proc}}$ blocks where Nb_{proc} is the number of processors and a power of two. To read a complete line, we need N_x/Nb_{proc} real or complex I/O "operations". To read a complete column, we need N_y/Nb_{proc} operations. The procedure becomes now:

- read a real line that is N_x/Nb_{proc} I/O operations,
- write a complex line that is $2 \times N_x/Nb_{\text{proc}}$ I/O operations,
- read a complex column that is $2 \times N_y/Nb_{\text{proc}}$ I/O operations,
- write a real column that is N_y/Nb_{proc} I/O operations.

Altogether we require $4 \times N_x/Nb_{\text{proc}}$ I/O operations to carry out forward or backward FFTs. By including

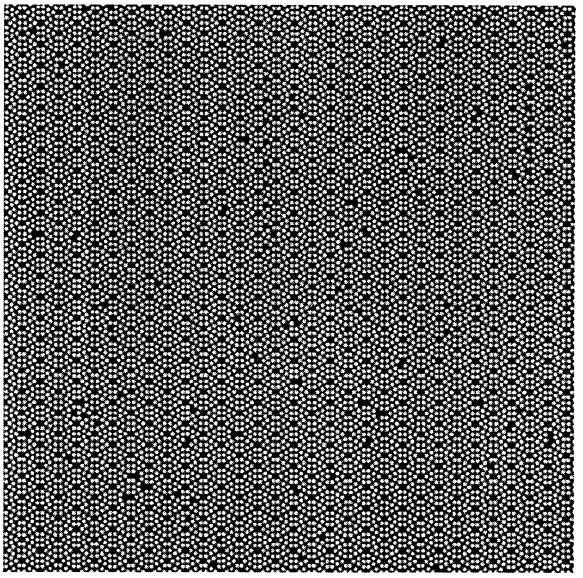

Fig. 7. Simulated charges distributions on a Si(111)7 × 7 lattice. Holes have been randomly added by taking off an atom at this site. Local charge distribution is a Gaussian function with a center at each atom location. Spatial resolution here is: $N_x = N_y = 1024$.

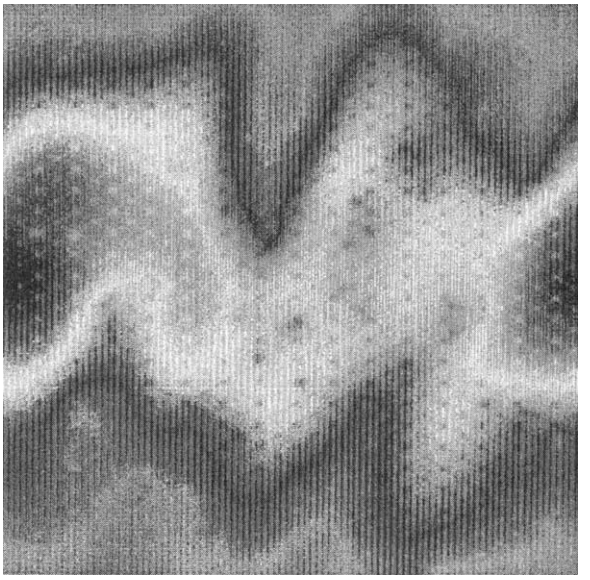

Fig. 8. Potential computed from the charge distribution of Fig. 7. Large scale structure is due to the particular configuration of the defects in the lattice.

the transfer rate TR, the total I/O time is now:

$$T_{\text{tot}} \approx 4 * \big((N_x/\text{nbproc}) + (N_y/\text{nbproc})\big) * AAT + 6 * (N_x * N_y)/\text{TR}. \quad (5)$$

Depending on the number of processors, the total time devoted to I/O is displayed in Fig. 9. The three key parameters are the transfert rate (TR), the average access time (AAT) and the number of processors. Since 1990, improvements are now down to $AAT \approx 2$–3 ms and TR = 200 Mbit/sec. We would expect these numbers to improve in the near future with the greater memory capacity from a denser packing, thus making out-of-core operations more competitive.

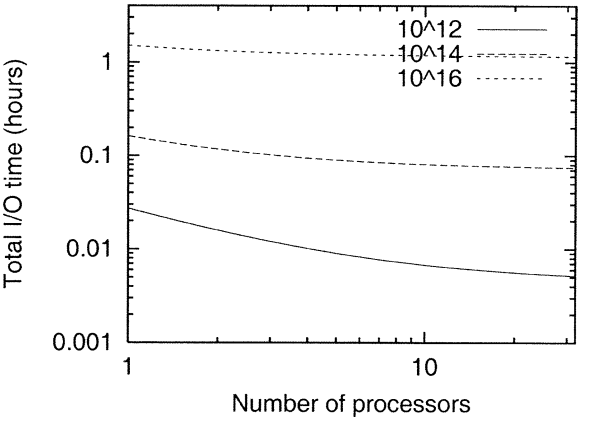

Fig. 9. Estimates of the time spent out-of-core for the computation of 2D periodic electric potential. We have based our estimates on $AAT = 2.5 \times 10^{-3}$ seconds and TR = 200 Mbit/sec. Labels refer to the size of the lattice: $2^{12} \times 2^{12}$ grid points (solid line), $2^{14} \times 2^{14}$ grid points (dashed line) and $2^{16} \times 2^{16}$ grid points (doted line).

3.2. Maps of the electric surface potential

Our example is given from solid state physics with the Si(111)7 × 7 lattice [2,9]. We simulated the surface of Si(111)7 × 7 with Gaussian Coulomb-like electric charges and random adatoms (holes) with a probability of 0.01. This is shown in Fig. 7. The computed electric potential is displayed Fig. 8. Here the resolution is $N_x = N_y = 1024$. Large-scale variations of the potential are due to the random locations of the vacancies. Visualization of large grids involving 10^8 grid points requires the use of large display device, such as the Power Wall at the University of Minnesota (http://www.lcse.umn.edu/).

4. Concluding remarks and work in progress

We have shown that one can reach resolutions of the order of 10,000 grid points along each direction in solving the 2D Poisson–Laplace equation with the spectral Fourier technique. Today it is possible to go to still higher resolutions of up to 100,000 grid points in each direction, provided we do not set the full arrays $P(x, y)$ but only deal with the lines of the matrix. We have used a 2D Fourier periodic spectral technique with high quality public domain FFTs and OPEN_MP on SGI parallel machine. The obvious example in such a case is a crystal lattice but many other applications are possible like, for instance, in marine geophysics [20]. In 3D and for Poisson–Laplace equation, spectral techniques have been already used [3,14] but with the need to take into account complex boundary conditions. In actually observed situations, there are discontinuities. In these cases, Fourier smoothing [3,26] has been used. We did not find in our example any Gibbs oscillations that would be due to local discontinuities from to the presence of large-scale faults. Perhaps surface vacancies do not create localized discontinuities. Nevertheless, this also raises the interesting issue of spectral method being able to resolve problems involving large contrasts in material properties, such as viscosity, permeability or elastic parameters. We will address these questions with spectral Fourier filters [6] in a future work. The accuracy of the spectral technique cannot be challenged by any other techniques, since it has an order equal to the number of Fourier modes [6]. However, for most applications one may not need such a high accuracy. The distinct advantage held by the Fourier spectral technique in a simple case such as a lattice is the raw speed because of the diagonal nature Laplacian operator. Moreover, even the FFT, already a fast algorithm, can be made faster by parallelization and in this case the speed-up factor is proportional to the number of processors. We are currently working on the out-of-core version of the code in order to handle very large grids, which are needed for large surface area of the order of $100\ \mu m \times 100\ \mu m$. Thus bigger versions of FFTs are required which will outstrip the largest memory of computers and will call for out-of-core usage (e.g., [17]). These issues will become more poignant, when three-dimensional Poisson equations are explored.

This kind of spectral Fourier technique could also be used for the Helmholtz equation and Darcian flow in heterogeneous porous media [4]. Generalization to variable dielectric constant [13] or more general elliptic problems is also possible but in these cases operators become more complex. Finally, Fourier treatment of complex geometries or internal boundaries remains an open question for future work.

Acknowledgements

We thank Paul Woodward for access to the Power Wall at LCSE, Renata Wentzcovitch for encouraging remarks, Shuxia, Zhang for technical assistance at MSI, Don Truhlar for helpful comments and Michel Beland from RQCHP for advice on OPEN_MP. Simulations have been made at the Minnesota Supercomputing Institute at Minneapolis and at CERCA at Montreal. This research has been financed by CRSNG Canada (Alain Vincent) and NSF and DOE (David Yuen). Xavier Thibert-Plante is a fellow of CRSNG Canada.

References

[1] N.A. Baker, D. Sept, S. Joseph, M.J. Holst, J.A. McCammon, Proc. Nat. Acad. Sci. 18 (2001) 10 037.
[2] E. Bengu, R. Plass, L.D. Marks, T. Ichihashi, P.M. Ajayan, S. Iijima, Phys. Rev. Lett. 20 (1996) 4226.
[3] E. Braverman, M. Israeli, A. Averbuch, L. Vozovoi, J. Comput. Phys. 144 (1998) 109.
[4] M.L. Bresseau, Rev. Geophys. 32 (1994) 285.
[5] T. Cagin, J. Che, Y. Qi, E. Demiralp, G. Gao, W.A. Goddard III, J. Nanoparticle Res. 1 (1999) 51.
[6] Canuto, M. Hussaini, A. Quarteroni, T. Zang, Spectral Methods in Fluid Dynamics, Springer, Berlin, 1988.
[7] H. Cheng, L. Greengard, V. Rokhlin, J. Comput. Phys. 155 (1999) 468.
[8] R.D. Coalson, T.L. Beck, http://bessie.che.uc.edu/tlb/rctb6/rctb6.html.
[9] M. Guggisberg, O. Pfeiffer, S. Schr, V. Barvich, M. Bammerlin, Ch. Loppacher, R. Bennewitz, E. Meyer, Appl. Phys. 19 (2001) A72.
[10] M. Holst, R.E. Kozack, F. Saied, S. Subramaniam, J. Biomol. Struct. Dyn. 6 (1994) 1437.
[11] H. Johansen, P. Colella, J. Comput. Phys. 147 (1998) 60.
[12] H.-Y. Nie, K. Horiuchi, H. Yamauchi, J. Masai, Nanotechnology 8 (1997) A24.
[13] I.M.B. Nielsen, C.L. Janssen, Comput. Phys. Comm. 136 (1–2) (2001) 29.
[14] Ch. Peter, W.F. van Gunsteren, Ph.H. Hunenberger, J. Chem. Phys. 17 (2002) 7434.

[15] L. Plagne, J.-Y. Berthou, J. Comput. Phys. 157 (2000) 419.
[16] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, Numerical Recipes, Cambridge University Press, 1992.
[17] D.N. Rockmore, Comput. Sci. Engrg. (2000) 60–64.
[18] T. Schlick, R.D. Skeel, A.T. Brunger, L.V. Kalé, J.D. Board, J. Hermans, K. Schulten, J. Comput. Phys. 151 (1999) 9.
[19] P. Swartztrauber, Fast Fourier Transform **fttpack**, http://www.netlib.org/fftpack/index.html, National Center for Atmospheric Research, Boulder, CO, 2002.
[20] W.H.F. Smith, Ann. Rev. Earth Planet. Sci. 26 (1998) 697.
[21] Y. Sugawara, O.B. Wright, O. Matsuda, M. Takigahira, Y. Tanaka, S. Tamura, V.E. Gusev, Phys. Rev. Lett. 18 (2002) 185504-1.
[22] V. Vlachy, Ann. Rev. Phys. Chem. 50 (1999) 145.
[23] E. Wasserman, A.R. Felmy, Appl. Env. Microbiology 6 (1998) 2295.
[24] E. Wasserman, A.R. Felmy, A. Chilakapati, Colloids and Surfaces B: Biointerfaces (2000) 19.
[25] V. Zaloj, N. Agmon, J. Chem. Phys. 3 (1998) 1216.
[26] J. Zhang, J. Comput. Phys. 143 (1998) 449.