PRAMANA
— journal of
physics

© Indian Academy of Sciences

Vol. 53, No. 6
December 1999
pp. 1–10

Local Simulation Algorithms for Coulombic Interactions

L. Levrel[1], F. Alet[2] J. Rottler[3], A.C. Maggs[1],
[1] Laboratoire de Physico-Chimie Théorique, UMR CNRS-ESPCI 7083, 10 rue Vauquelin, F-75231 Paris Cedex 05, France.
[2]Theoretische Physik and Computational Laboratory, ETH Zürich, CH-8093 Zürich, Switzerland.
[3] PRISM, Bowen Hall, Princeton University, Princeton, NJ 08544, USA.

Abstract. We consider a problem in dynamically constrained Monte-Carlo dynamics and show that this leads to the generation of long ranged effective interactions. This allows us to construct a local algorithm for the simulation of charged systems without ever having to evaluate pair potentials or solve the Poisson equation. We discuss a simple implementation of a charged lattice gas as well as more elaborate off-lattice versions of the algorithm. There are analogies between our formulation of electrostatics and the bosonic Hubbard model in the phase approximation. Cluster methods developed for this model further improve the efficiency of the electrostatics algorithm.

Keywords. Coulomb, Algorithms, Maxwell's equations, Monte-Carlo

PACS Nos 2.0

1. Introduction

Computer modeling of charged systems is demanding due to the range of the Coulomb interaction [1]. The direct evaluation of the Coulomb sum for N particles, $U_c = \sum_{i<j} e_i e_j / 4\pi\epsilon_0 r_{ij}$, requires computation of the separations r_{ij} between all pairs of particles, which implies $O(N^2)$ operations are needed per sweep or time step. Much effort has been put into developing algorithms with more favorable scaling of the effort with system size. Most large scale codes in use at the moment solve Poisson's equation for the electrostatic potential using the fast Fourier transform [2] after interpolating charges to a grid. For an ensemble of charges interpolated to M nodes of a lattice this gives the electrostatic energy in a time which scales as $O(M \ln M)$.

The use of a *global* solver for the Poisson equation leads to a strong bias in the choice of simulation techniques. Since the solution of the Poisson equation is unique the motion of a single charge requires the global re-calculation of the electrostatic potential, so that Monte-Carlo methods must (apparently) lead to hopelessly inefficient codes: *All* efficient large scale codes are based at the moment on molecular dynamics rather than Monte-Carlo.

2. Electrostatics can be formulated locally

It is surprising that the Coulomb interaction poses such tremendous difficulty; after all the underlying Maxwell equations are *local.* The question then arises as to what part of Maxwell's equations give rise to electrostatic interactions. Is it possible to generate effective Coulomb interactions in a manner better adapted to computer simulation?

In electromagnetism Coulomb's law comes [3] from a local expression for the energy:

$$U = \int \frac{\epsilon_0 \mathbf{E}^2}{2}\, \mathrm{d}^3\mathbf{r} \tag{1}$$

where $\mathbf{E}$ is the electric field, and the imposition of Gauss' law

$$\mathrm{div}\ \mathbf{E} - \rho/\epsilon_0 = 0. \tag{2}$$

This motivates the use of the following partition function for the electric field

$$\mathcal{Z}(\{\mathbf{r}_i\}) = \int \mathcal{D}\mathbf{E} \prod_{\mathbf{r}} \delta\left(\mathrm{div}\ \mathbf{E} - \rho\left(\{\mathbf{r}_i\}\right)/\epsilon_0\right)\, e^{-U/k_BT}. \tag{3}$$

The charge density, $\rho(\mathbf{r}) = \sum_i e_i \delta(\mathbf{r} - \mathbf{r}_i)$ and the charge of the i'th particle is e_i. We change integration variables to $\mathbf{e} = \mathbf{E} + \nabla\phi_p$ with $\nabla^2\phi_p = -\rho(\{\mathbf{r}_i\})/\epsilon_0$ and find

$$\mathcal{Z}(\{\mathbf{r_i}\}) = \int \mathcal{D}\mathbf{e}\ \delta(\mathrm{div}\ \mathbf{e})\, e^{-\beta\frac{\epsilon_0}{2}\int(\mathbf{e}-\nabla\phi_p)^2\ \mathrm{d}^3\mathbf{r}}. \tag{4}$$

We now notice that the cross term in the energy is zero by integration by parts:

$$2U/\epsilon_0 = \int (\nabla\phi_p)^2\ \mathrm{d}^3\mathbf{r} + \int \mathbf{e}^2\ \mathrm{d}^3\mathbf{r} - \underbrace{2\int \nabla\phi \cdot \mathbf{e}\ \mathrm{d}^3\mathbf{r}}_{=0} \tag{5}$$

so that

$$\mathcal{Z} = e^{-\beta\frac{\epsilon_0}{2}\int(\nabla\phi_p)^2} \int \mathcal{D}\mathbf{e}\ \delta(\mathrm{div}\ \mathbf{e}) e^{-\beta\frac{\epsilon_0}{2}\int \mathbf{e}^2\ \mathrm{d}^3\mathbf{r}} \tag{6}$$

We conclude that $\mathcal{Z}(\{\mathbf{r}_i\}) = \mathcal{Z}_{Coulomb}(\{\mathbf{r}_i\}) \times \mathrm{const}$: Relative statistical weights are unchanged if we allow the transverse field to vary freely, rather than quenching it to zero. To sample the partition function Eq. (3) we *only* have to find solutions to Gauss' law. If we dispense with the condition $\mathrm{curl}\ \mathbf{E} = 0$ usually imposed in electrostatics, $\mathbf{E}$ can be *any* solution to the equation $\mathrm{div}\ \mathbf{E} - \rho/\epsilon_0 = 0$; the general solution is $\mathbf{E} = -\mathrm{grad}\ \phi_p + \mathrm{curl}\ \mathbf{Q}$ with $\mathbf{Q}$ arbitrary.

2.1 Monte-Carlo sampling

We have formulated a Monte-Carlo algorithm [4] that uses the Metropolis method together with the energy Eq. (1). In order to generate configurations according to the statistical weight of Eq. (3) we need to

- move particles without violating the constraint of Gauss' law to preserve the delta-function on $\mathrm{div}\ \mathbf{E} - \rho/\epsilon_0$;
- integrate over the transverse degrees of freedom $\mathbf{e}$, of the electric field.

Discretization

The system is discretized by placing charged particles on the $M = L^3$ vertices of a periodic cubic lattice, $\{i\}$. The components of the electric field $E_{i,j}$ are associated with the $3M$ links $\{i, j\}$ of the lattice. There are $3M$ plaquettes on the lattice each defined by four links. We use the notation $E_{1,2}$ to denote a local contribution to electric flux leaving 1 towards 2. If the link from 1 to 2 is in the positive x direction we consider that this is the local value of the x component of the field $\mathbf{E}$. The $3M$ variables $E_{i,j}$ are thus grouped into M three dimensional vectors.

Particle motion

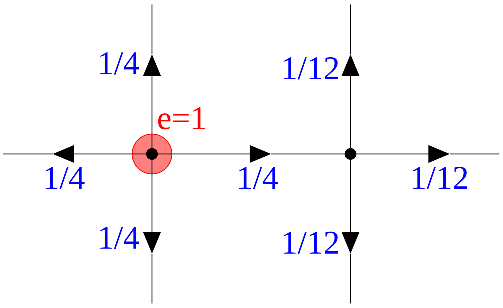

Figure 1. A local solution to Gauss' law. Numbers on each link correspond to the electric flux in the direction of the arrows. There is a charge $e/\epsilon_0 = 1$ on the leftmost node of the figure.

Two nodes of the lattice are shown in Fig. (1). Gauss' law is interpreted in the integral form $\int \mathbf{E} \cdot d\mathbf{S} = e/\epsilon_0$ where e is the charge enclosed by a surface enclosing each site. On our lattice we associate the fluxes with the links; if a charge of $e/\epsilon_0 = 1$ is at the leftmost site we can satisfy Gauss' law with a flux of $1/4$ on each outward link. On the rightmost node the charge is zero and the signed sum of the fluxes at this site is also zero.

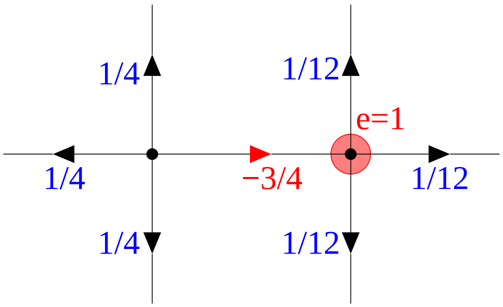

Figure 2. After moving the particle from the leftmost to the rightmost node we wish to find a new solution to the Gauss constraint. This is done by modifying the flux on a single link connecting the two sites from $1/4$ to $-3/4$.

We now displace the particle to the rightmost site: Fig. (2). We must find a new solution to Gauss' law. *One* solution is found by taking the field on the link between the sites, E and modifying it with the law $E \rightarrow E - e/\epsilon_0$ or from $1/4$ to $-3/4$. This is the essence of the method: Gauss' law allows an entirely local modification of the fields, whereas Poisson's equation implies a global update of the potential at all sites of the lattice. The energy change, $\Delta U = \epsilon_0 \left((-3/4)^2 - (1/4)^2\right)/2$, associated with the displacement is used in a standard Metropolis algorithm to decide whether to accept the update.

Transverse field

The update of the field, slaved to the particle motion is not in itself sufficient to fully sample the partition function, Eq. (3); we must also integrate over the transverse field $\mathbf{e}$. One can group four links into plaquettes and modify the four links by the same increment, Δ so as to conserve Gauss' law: Fig. (3). In our code particle motion and plaquette updates are mixed stochastically with probabilities p and $(1-p)$. We chose Δ uniformly distributed in $[-\Delta_0$,$\Delta_0]$, where Δ_0 is chosen so that the Monte-Carlo acceptance probability of this move is close to $1/2$.

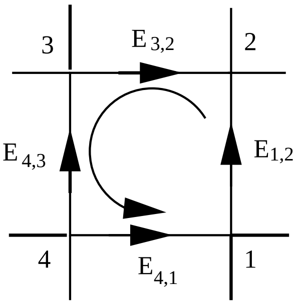

Figure 3. The circulation of a plaquette is incremented by 4Δ by modifying the field of each link: $E_{4,3} \rightarrow E_{4,3}-\Delta$, $E_{4,1} \rightarrow E_{4,1}+\Delta$, $E_{3,2} \rightarrow E_{3,2}-\Delta$, $E_{1,2} \rightarrow E_{1,2}+\Delta$

2.2 Large scale dynamics

The above algorithm is based on local, stochastic updates for both the particle positions and the electric field. It is thus rather natural that both the particles and the electric field exhibit diffusive dynamics. It can be shown [6] that the field dynamics are governed by a Langevin equation:

$$\frac{\partial \mathbf{E}}{\partial t} = D_E \left(\nabla^2 \mathbf{E} - \mathrm{grad}\ \rho/\epsilon_0\right) - \mathbf{J}/\epsilon_0 + \mathrm{curl}\ \zeta(t, \mathbf{r}) \tag{7}$$

which *replace* Maxwell's equations in our Monte-Carlo scheme. The diffusion coefficient of the transverse electric field D_E is related to p and Δ_0. The analytic structure of Eq. (7) is particularly interesting when there are free, mobile charges so that the electric current is linked to the electric field via the equation $\mathbf{J} = \sigma \mathbf{E}$. In this case the dispersion law of the electric field develops a *gap* so that

$$i\omega\,\hat{\mathbf{q}}\wedge\mathbf{E} = (\sigma/\epsilon_0 + D_E\,\mathbf{q}^2)\,\hat{\mathbf{q}}\wedge\mathbf{E}. \tag{8}$$

The relative diffusion rates of the electric and charge degrees of freedom are adjusted by modifying p. We initially believed that the algorithm should be run in a regime in which the effective diffusion coefficient of the electric field is comparable to or somewhat larger than that of the particles: The particles are then always interacting via a field that is close to equilibrium and are not slowed down by the "drag" from the electric degrees of freedom. In dilute systems this choice would mean that much more CPU time must be spent updating plaquette degrees of freedom than those corresponding to the particles and the algorithm would become inefficient.

Simulations show that one can use a much lower rate of update for the plaquettes. There are several ways of understanding the efficiency of the code [7]:

- The gap in the electric dispersion law Eq. (8) gives fast relaxation of the electric degrees of freedom *even at long wavelengths* and when D_E is small.
- Motion of particles already integrates over the electric field: Motion of a particle around a single plaquette has *exactly* the same effect as an update of a plaquette by $\Delta = e/\epsilon_0$.

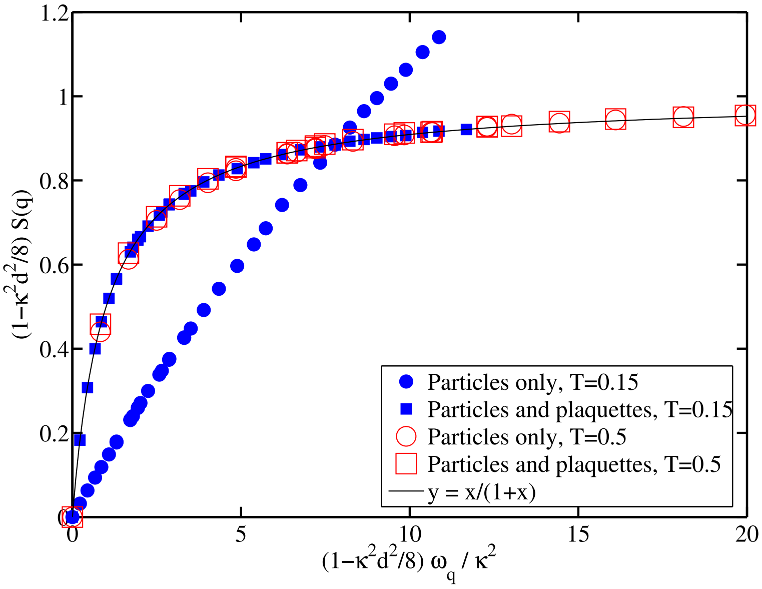

Figure 4. Charge structure factor as a function of ω_q. At high temperatures the same structure factor Eq. (9) is found with and without plaquette updates. At lower temperatures this law is still observed when the plaquettes are updated; without plaquette updates one finds $S \sim \omega_q$. Fit to $d = 1.29$. $L = 15$, $N = 336$.

We simulated the above lattice gas and measured the charge-charge structure factor. Debye-H¨uckel theory applied to a lattice gas gives the following expression for $S(q) = \omega_q/(\kappa^2 + \omega_q)$ with $\omega_q = \sum_{i=1}^{3} 2(1-\cos q_i)$ and $\kappa^2 = c\,e^2/k_B T\epsilon_0$, with c the charge density. When $\mathbf{q} \to 0$ then $\omega_q \to \mathbf{q}^2$. When the finite size of particles is also taken into account [8] the long wavelength structure factor must be slightly modified. We thus fit our numerical structure factors to the expression

$$S(q) = \frac{\omega_q}{\kappa^2 + \omega_q(1 - \kappa^2 d^2/8)} \tag{9}$$

where d is the particle diameter. In Fig. (4) we show results of simulations at $k_B T \epsilon_0/e^2 = 0.5$ and $k_B T \epsilon_0/e^2 = 0.15$ which are in very good agreement with this expression.

Numerical experimentation leads to a rather remarkable result: If one *drops the plaquette updates entirely* and works at a sufficiently high temperature the charge-charge correlation functions are almost indistinguishable, Fig. (4), from those in which the system is simulated with plaquette updates. The system is clearly *not fully ergodic*, since only values of the plaquette circulation which are integer multiples of e/ϵ_0 are possible. Yet charge correlation functions appear to be very similar to those of a fully equilibrated system.

It is only at lower temperatures that there is a qualitative change in behaviour and the system has very different behaviors depending on whether the plaquettes are independently excited or not. One possible explanation is that without plaquette updates the charges are condensing into tightly bound, neutral pairs. Indeed, if one studies a system with just two charges at low temperatures one finds a finite, constant line tension between charges, and thus a potential between two charges which increases linearly with separation.

3. Improved discretization

The lattice gas described above is not yet useful for simulations in condensed matter physics: The algorithm becomes *very* inefficient at low temperatures. Motion of a particle between two nodes leads to a finite modification of the field on the connecting link, and thus a finite energy barrier; the Monte-Carlo acceptance rate falls off exponentially at low temperatures. In practice this leads to hopelessly inefficient simulations for temperatures such that $k_B T \epsilon_0/e^2 < 0.15$. In condensed matter physics it is usual to work in terms of a normalized Bjerrum length, ℓ_B: the length at which the electrostatic interaction between two particles is equal to $k_B T$ when measured in units of particle size: $\ell_B/a = e^2/4\pi\epsilon_0 k_B T$. Typically we are interested in $\ell_B \sim 5 - 20$. $k_B T \epsilon_0/e^2 = 0.15$ corresponds to only $\ell_B \sim 1/2$, we need to gain at least a factor of ten in temperature to have a useful code.

Charge spreading

One can avoid the fall off in efficiency at low temperatures by spreading the charge over several lattice sites. In Fig. (5) we have plotted the Monte-Carlo acceptance rate for particle motion as a function of temperature for a system of dimensions $L = 15$ with two free charges. We associate the charge of each particle to n^3 sites with $n = 1, 2, 3$. We find that the temperature at which the code becomes inefficient varies as $T \sim 1/n^3$. When we measure the Bjerrum length in terms of the size of the spread-out particle we find that we can simulate at a Bjerrum length $\ell_B < n^2/2$ without the acceptance rate falling too low.

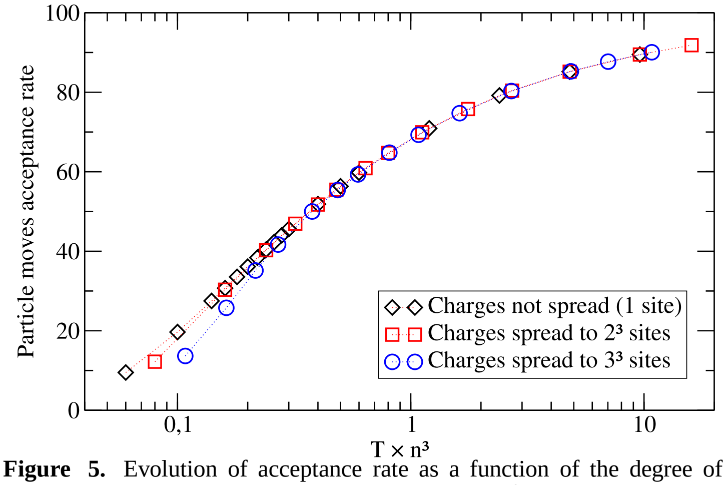

Figure 5. Evolution of acceptance rate as a function of the degree of spreading. Dashed lines: guide to eye. Temperature in units of $e^2/\epsilon_0 k_B$.

Off-lattice methods

By allowing the particles to move in the continuum and then interpolating the charges to the network (as is usual in Fourier codes [2]) we can continuously tune the Monte-Carlo step size so that the acceptance rate does not fall off with temperature. Many choices of interpolation scheme are possible, in our codes [5] we have used a scheme based on piecewise quadratic splines, leading to interpolation to a cube of 27 sites around each particle.

When a particle moves in the continuum we need to generalize the update rule $E \to E - e/\epsilon_0$ that generates the local updates to the electric field. We must generate an electric current $\mathbf{J}$ which is a solution to the continuity equation $\mathrm{div}\, \mathbf{J} = -\Delta\rho$, where $\Delta\rho$ is the finite variation of the charges on the nodes calculated from the spline interpolation scheme. We then update the electric field with $\mathbf{E} \to \mathbf{E} - \mathbf{J}/\epsilon_0$. The equation for $\mathbf{J}$ is clearly *under-determined.* We use this fact to our advantage to find *one* solution which is simple. This is illustrated in Fig. (6). We construct a Hamiltonian path which visits each node where the interpolated charge has changed. The current is then calculated on each link as the sum of the charge variations on all previous nodes on the path.

4. Cluster sampling for the electric field

In many physical situations the algorithm as described above is remarkably efficient: The presence of a finite concentration of free charges always gives rise to a finite σ and thus fast field relaxation: the slowest modes in the system are then modes of density relaxation. However, one is sometimes interested in highly heterogeneous systems where charges are either not mobile (perhaps they are attached to a polymer or a surface) or are excluded from

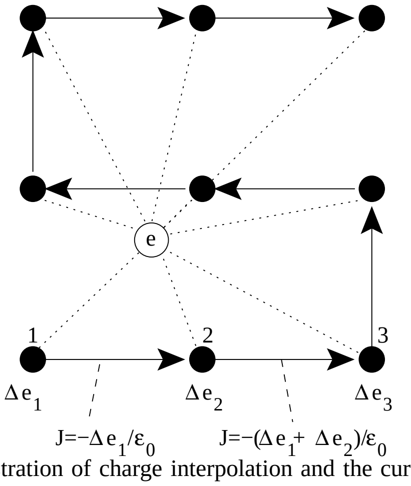

Figure 6. Illustration of charge interpolation and the current update using a Hamiltonian path in two dimensions. A charge (open circle) interpolates onto the $3^2 = 9$ lattice sites (solid). The partial charges on these sites change by an amount Δe_i when the particle moves. The current **J** flows along the Hamiltonian path and terminates at the last site, since $\sum_i \Delta e_i = 0$.

certain parts of the simulation volume (perhaps due to the presence of a large colloidal inclusion). Similarly in dipolar fluids such as water in the absence of free charges $\sigma = 0$. In this case we have to rely on the slower diffusive motion of the electric field in Eq. (7) driven by the plaquette updates. We will now show how to generate updates which give rise to fast field equilibration even for the case $\sigma = 0$ using a cluster algorithm for the electric field variables.

There is a remarkable analogy between the constrained partition function Eq. (3) and the bosonic 2+1 dimensional Hubbard model at zero temperature in the phase approximation [9]. Its partition function is written in terms of a constrained vector field discretized on a lattice:

$$\sum_{\mathbf{E}\,\mathrm{integer}} \exp\left(-\frac{1}{2K}\int \mathbf{E}^2\, \mathrm{d}^3\mathbf{r}\right) \prod_{\mathbf{r}} \delta(\mathrm{div}\ \mathbf{E}). \tag{10}$$

The major difference compared with the partition function Eq. (3) is that the field variables on each link are now integers rather than reals; the field variables represent the real bosonic currents in the system. K is the ratio between kinetic energy and the repulsive energy and is interpreted as an effective temperature. The partition function Eq. (10) has been extensively studied [9] and has a phase diagram with a small K (large repulsion) insulating phase and a superfluid phase at large K (large kinetic energy).There is a phase transition between these phases at $K = 0.33305(5)$. We have already seen an example of a similar partition function when we suppressed the plaquette updates in the electrostatic algorithm. The very different structure factors observed in Fig.(4) are presumably linked to the two phases of the Hubbard model.

Early numerical studies of Eq. (10) were based on local updates to plaquette degrees of freedom, in a manner very similar to that described above for the electrostatic algorithm. More recently *much* more efficient algorithms have been found based on cluster or worm sampling [10]. The algorithm works by nucleating a pair of positive and negative pseudo-particles on the same site of the system. One of the pseudo-particle is then moved on the lattice so that $\mathrm{div}\ \mathbf{E} \neq 0$. The motion of this pseudo-particle is a random walk biased by the energy in Eq. (10) [10]. The pseudo-particle eventually returns to its stationary partner and the two particles then annihilate each other. When the pseudo-particle moves, the electric field is updated exactly as described above for the electrostatic algorithm, $E \rightarrow E - e/\epsilon_0$. At the moment of annihilation the total weight of the generated path or cluster is compared with its time reversed version and the update is either globally accepted or refused [10]. The algorithm is very reminiscent of the creation of virtual pairs of particles in quantum mechanics. In the original version of the algorithm [10] the pair of pseudo-particles had a charge $e/\epsilon_0 = \pm 1$ in order to generate fields with integer flux on the links.

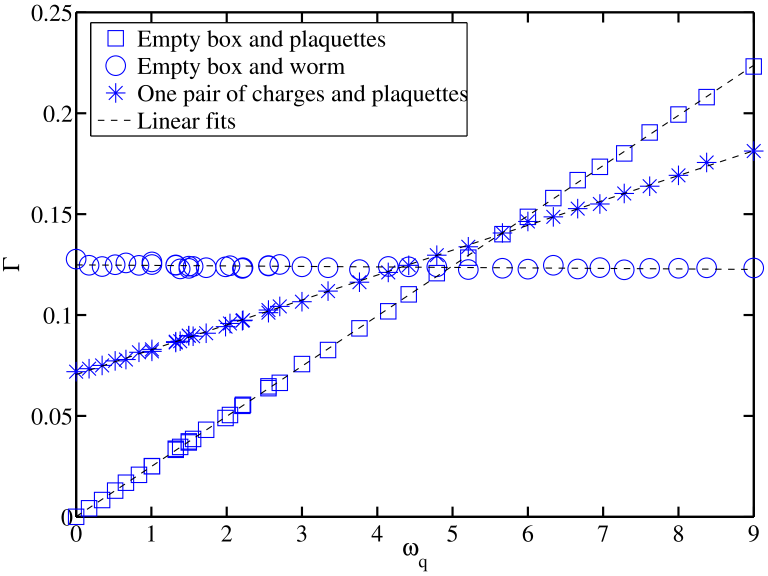

Figure 7. Decay rate, Γ of the transverse electric field as a function of ω_q. □: empty box, plaquette updates only illustrating diffusive field motion $\Gamma = D_E \omega_q$; time scale: L^3 Monte-Carlo tries. ✳: two particles in box with plaquette updates $\Gamma = \sigma/\epsilon + D_E \omega_q$; time scale: L^3 Monte-Carlo trials, 1:1 split in Monte-Carlo attempts between particles and plaquettes. ○: cluster algorithm showing uniform relaxation across all modes; time scale: a single cluster update. $L = 15$, $k_B T \epsilon_0 / e^2 = 0.5$.

Given the efficiency of this algorithm in sampling Eq. (10) we have explored the interest of the method for sampling the electrostatic partition function Eq. (3). In the electrostatic problem we are interested in integrating over all values of the electric field. Thus we create the pair of virtual particles with a random charge distributed according to a uniform distribution between $-e_m$ and e_m. If e_m is small we find that the typical cluster is very large containing about $0.7 \times L^3$ links. In order to maximize the rate of change in the field on the links we should use a large value for e_m. However, the use of a too large value for e_m prevents the propagation of the virtual particle, which backtracks too readily in its own path and only visits $O(1)$ sites before the annihilation. This is again understood in the boson analogy as being due to the different behaviour of a particle in a superfluid or insulating phase. In practice we use $e_m^2 \sim k_B T \epsilon_0$.

In Fig. (7) we plot the decay rate for the Fourier components of the transverse electric field for three different simulations. The first simulation is a box without free charges simulated using simple plaquette updates. As predicted by Eq. (7) the dynamics of the field is diffusive, $\Gamma = D_E q^2$ for small q. We then add two free charges to the system and measure the same relaxation rate. We see that a finite gap is introduced into the dispersion law as predicted by Eq. (8). Finally we simulate the system without charges using the cluster algorithm. We find that the relaxation rate is weakly dependent on the mode and we have a method which equilibrates the transverse electric field in about two cluster moves per link, rather than the $O(L^2)$ sweeps needed for the diffusion of the electric field with local plaquette updates. Mixing particle motion with occasional cluster updates in a heterogeneous system now allows one to ensure fast relaxation of the electric degrees of freedom even in situations where σ is zero.

We understand the efficiency of the cluster algorithm in both the Hubbard model and in the electrostatic algorithm as being due to the dispersion law Eq. (7) in the presence of a current $\mathbf{J} = \sigma \mathbf{E}$. We have seen that if σ is non-zero then we develop a gap in the electric spectrum and the system is described by a dynamic exponent $z = 0$ rather than $z = 2$ characteristic of diffusion. In the cluster algorithm the finite conductivity comes from the virtual particles rather than from true free ions but their effect is very similar on the relaxation spectrum for the electric field.

References

[1] T. Schlick et al. *J. Comp. Phys.* 151, 9 (1998).

[2] Ulrich Essmann, Lalith Perera, Max L. Berkowitz, Tom Darden, Hsing Lee, Lee G. Pedersen *J. Chem. Phys.* 103, 8577-8593, (1995).

[3] *Classical Electrodynamics*, J.S. Schwinger et al. (Westview Press, 1998).

[4] A. C. Maggs and V. Rossetto, *Phys. Rev. Lett.* 88, 196402 (2002).

[5] Joerg Rottler and A.C. Maggs, *J. Chem. Phys.* 120, 3119 (2004).

[6] A.C. Maggs, *J. Chem. Phys.* 117, 1975 (2002).

[7] A.C. Maggs, *J. Chem. Phys.* 120, 3108 (2004).

[8] B.P. Lee, M.E. Fisher, *Europhys. Lett.* 39, 611 (1997).

[9] Erik S. Sørensen, Mats Wallin, S. M. Girvin and A. Peter Young, *Phys. Rev. Lett.* 69, 828 (1992).

[10] F. Alet and E. Sørensen, *Phys. Rev.* E 67, 015701(R) (2003).