# Monte Carlo Simulations with First-Principles Energies

José Luís Martins[*,‡] and J. M. Pacheco[†]

[*]*Departamento de Física, Instituto Superior Técnico*
*Av. Rovisco Pais, 1049-001 Lisboa, PORTUGAL*
[‡]*INESC, Rua Alves Redol, 9, 1000-029 Lisboa, Portugal*
[†]*Departamento de Física da Universidade, 3000 Coimbra, Portugal*

**Abstract.** We present a method to perform efficiently Monte Carlo simulations for molecules with total energies calculated from first principles density functional theory using a gaussian basis set. It relies on the strong coupling between the Monte Carlo algorithm and the structure of the first principles code, which has been parallelized. The feasibility of the method is illustrated by carrying out extensive computer simulations for a cluster with 21 atoms. Possible generalizations and extensions of the method are briefly discussed.

## INTRODUCTION

Two families of methods have dominated the field of statistical mechanics simulation of materials since the appearence of electronic computers: molecular dynamics and Monte Carlo. [1] In fact they are among the earliest simulations run on computers. The earlier simulations used classical interatomic forces and energies, that is, one had a parametrization or prescription for how the energy of the system $U(\vec{R}_1, \ldots, \vec{R}_n)$ depended on the positions $\vec{R}_i$ of the $n$ atoms in the system. For the case of a pair potential one assumes that $U$ takes the form,

$$U(\vec{R}_1, \ldots, \vec{R}_n) = \sum_{i<j} V(|\vec{R}_i - \vec{R}_j|) \tag{1}$$

and an example of a pair potential $V$ is the Lennard-Jones interaction, $V(R) = \epsilon((\sigma/R)^{12} - (\sigma/R)^6)$. Ideally one should calculate $U$ from first principles of quantum mechanics. Developments in electronic structure theory of solids and molecules showed that such an ideal would be feasible but with early computers and methods, the calculations could only be carried out for small molecules at great computational cost, and therefore could not be coupled with statistical mechanics methods that required the fast calculation of $U$ for a large number of atoms. That situation was changed when in a seminal paper Car and Parrinello [2] showed that one could do first principles molecular dynamics in an efficient way for systems with a few tens of atoms. (Currently we can deal with hundreds of atoms in a supercomputer or tens of atoms in a PC.) In the Car–Parrinello method energies and forces are calculated within the framework of density functional theory [3,4] with a plane-wave basis set (and a pseudopotential approximation) [5] and molecular dynamics simulated annealing was used to minimize the energy functional of the density. There are several technical details of the method that make it highly efficient, but make it non-trivial to extend to other basis sets or to Monte Carlo methods.

Here we present a method to perform efficiently Monte Carlo simulations for molecules using total energies calculated from first principles density functional theory using a gaussian basis set. In Chapter II we will review the characteristics of the Monte Carlo and molecular dynamics methods, as well as the key features of the first-principles molecular dynamics with a plane-wave basis set. In Chapter III we will present our implementation of the gaussian basis set Monte Carlo and illustrate the method with an example.

## BACKGROUND

### Molecular Dynamics

In a classical molecular dynamics (MD) simulation we are given the positions $\vec{R}_i(t)$ and velocities $\vec{v}_i(t) = d\vec{R}_i(t)/dt$ of the i-th atom at time $t$, and a prescription to calculate the potential energy of the system $U(\vec{R}_1(t), \ldots, \vec{R}_n(t))$. The forces on the atoms, $\vec{F}_i(\vec{R}_1(t), \ldots, \vec{R}_n(t)) = -\nabla_{\vec{R}_i} U(\vec{R}_1(t), \ldots, \vec{R}_n(t))$ are obtained from the gradient of the energy with respect to the atomic positions. One then integrates Newton's equations of motion:

$$\vec{R}_i(t+\delta t) = \vec{R}_i(t) + \delta t\, \vec{v}_i(t) + \frac{1}{2}\delta t^2 \, \frac{1}{M_i}\vec{F}_i(\vec{R}_1(t), \ldots, \vec{R}_n(t)) + \ldots \tag{2}$$

$$\vec{v}_i(t+\delta t) = \vec{v}_i(t) + \delta t\, \frac{1}{M_i}\vec{F}_i(\vec{R}_1(t), \ldots, \vec{R}_n(t)) + \ldots \tag{3}$$

Several algorithms are used in practice, corresponding to different combinations and truncations of the Taylor expansions of position and velocity. [1] Repeating the process allows one to produce a trajectory for the atoms.

In principle the process should conserve the total energy of the system, $E = U(\vec{R}_1(t), \ldots, \vec{R}_n(t)) + \sum_i (1/2M_i)\vec{v}_i(t)^2$ and the best algorithms are those that have a good energy conservation. One can couple the system with a thermostat, allowing for energy to flow between the system being simulated and the thermostat, so that the simulations may be performed at constant temperature instead of constant energy. If one lowers slowly the temperature of the thermostat we will lower also the energy of the system, a minimization procedure called simulated annealing.

## Monte Carlo

In a typical Monte Carlo (MC) simulation, [1] given the positions $\vec{R}_i^{(m)}$ of the i-th atom at step $m$ one calculates the corresponding potential energy $V^{(m)} = U(\vec{R}_1^{(m)}, \ldots, \vec{R}_n^{(m)})$. One then generates a trial configuration with atomic positions $\vec{R}_i^{\text{try}}$, calculates its corresponding potential energy $V^{\text{try}}$ and decides either to accept or reject that configuration according to a well defined algorithm. Acceptance means taking $\vec{R}_i^{(m+1)} = \vec{R}_i^{\text{try}}$, while rejection implies generating new trial configurations until one is accepted. The popular Metropolis algorithm for accepting/rejecting configurations is the following: If $V^{\text{try}} < V^{(m)}$ the configuration is accepted; If $V^{\text{try}} > V^{(m)}$ the configuration is accepted with probability $0 < p = \exp(-(V^{\text{try}} - V^{(m)})/k_B T) < 1$ where $T$ is the simulation temperature, and $k_B$ the Boltzmann constant. A chain of configurations $\vec{R}_i^{(m)}$ is generated by repeating the procedure. Again if we decrease slowly the simulation temperature, we will be minimizing the energy of the system by simulated annealing.

The efficiency of a Monte Carlo simulation depends on the choice of the trial configurations. If they are close to the original configuration the acceptance rate will be high but one will move slowly in configuration space. If they are very different, we find out that $V^{\text{try}} - V^{(m)} >> k_B T$ most of the times (remember, we are supposed to be near a minimum of energy) and most trial moves will be rejected. A popular choice of Monte Carlo trial step is to move one atom at a time, $\vec{R}_i^{\text{try}} = \vec{R}_i^{(m)}$ for $i \neq k$, and $\vec{R}_k^{\text{try}} = \vec{R}_k^{(m)} + \delta\vec{R}$, where $\delta\vec{R}$ is a randomly chosen vector with an average length chosen to optimize the acceptance/rejection rate. At each Monte Carlo step one tries to move a different atom.

## Comparative efficiency of classical MD and MC

We can now compare the computational cost of the Monte Carlo and Molecular Dynamics methods. To give a chance to all atomic coordinates to move in an uncorrelated way, we must try $3n$ Monte Carlo moves. In Molecular Dynamics we update all the coordinates in a single step, in the “right” direction dictated by the interplay of the forces and inertia. However for each step of Molecular Dynamics we must calculate $3n$ force components. For a classical pair potential, each force component has about the same computational cost of calculating just the energy. We can roughly say that a Molecular dynamics step costs $\approx 3n$ times more than a Monte Carlo step, but we need $\approx 3n$ times less steps to sample a representative part of the phase space. That is the computational efficiency of both methods is comparable, and choosing one or another depends on many factors. For example, it may be easier to extract dynamical quantities from Molecular Dynamics and equilibrium properties from Monte Carlo.

## First Principles Molecular Dynamics

In a first principles Density Functional simulation the ground state electronic energy of the system is calculated by finding an electronic density $\rho_{\min}(\vec{r})$ which minimizes a density functional,

$$E_{\{Z_j,\vec{R}_j\}}[\rho] = \int \rho(\vec{r}) v(\vec{r})\, d^3r + F[\rho] + E_{\text{Ewald}} \tag{4}$$

where $v(\vec{r}) = \sum_j -\frac{Z_j}{|\vec{r}-\vec{R}_j|}$ is the coulomb potential of nuclei with charge $Z_j$ located at positions $\vec{R}_j$, $\rho(\vec{r})$ is the total electron density, $F[\rho]$ is a universal energy functional (in particular it is independent of $\vec{R}_j$) and the Ewald energy is $E_{\text{Ewald}} = \sum_{i<j} \frac{Z_j Z_i}{|\vec{R}_i-\vec{R}_j|}$. The minimization is done under the constraint of fixed number $N$ of electrons, that is $\int \rho(\vec{r})\, d^3r = N$.

The forces on nuclei are given by

$$\vec{F}_i = -\vec{\nabla}_{\vec{R}_i} E_{\{Z_j,\vec{R}_j\}}|_{\rho_{\min}} = \int \rho_{\min}(\vec{r}) \frac{\vec{r}-\vec{R}_i}{|\vec{r}-\vec{R}_i|^3}\, d^3r + \sum_{j\neq i} Z_j Z_i \frac{\vec{R}_i-\vec{R}_j}{|\vec{R}_i-\vec{R}_j|^3} \tag{5}$$

where we used the fact that

$$\frac{\delta E_{\{Z_j,\vec{R}_j\}}[\rho]}{\delta\rho}|_{\rho_{\min}} = 0 \ . \tag{6}$$

This result is the Hellman-Feynman theorem applied to Density Functional Theory. Within such a scheme, the minimization of the the density functional, that is the calculation of $\rho_{\min}(\vec{r})$, is extremely expensive when compared with the calculation of $\vec{F}_i$. If one compares the efficiency of a Monte Carlo algorithm with a molecular dynamics algorithm for this case of "almost cost free" computation of the forces compared with the calculation of the energy, one reaches the conclusion that Molecular Dynamics is much more efficient than Monte Carlo, by a factor that is proportional to the number of atoms of the system. This is the case for a plane-wave basis-set. However, whenever the minimization of the energy functional in Eq. 4 is done under some constraints — for example a finite atomic basis set — extra terms, called Pulay forces, may appear in the expression for the force, rendering its calculation more expensive. [6]

The balance between the energy minimization and force calculation can be seen easily in the original Car-Parrinello framework, where the charge density is given by a sum of the squares of one particle wavefunctions, $\rho(\vec{r}) = \sum_k |\psi_k(\vec{r})|^2$, and the wavefunctions are expanded in a sum of plane-waves, $\psi_k(\vec{r}) = \sum_l c_{kl} \frac{1}{V} \exp(i\vec{G}_l \cdot \vec{r})$. The minimization of the energy functional is now a minimization with respect to the coeficients $c_{kl}$ (with appropriate constraints of orthonormality of the $\psi_k$). As the scale of the electronic energies is the atomic unit, of the order of $3 \times 10^5$ K, if one does a molecular dynamics simulation with the $c_{kl}$ as fictitious variables at a temperature of a few hundred Kelvin, one will always be very close to the minimum of the electronic energy, that is we are effectively minimizing the energy functional by simulated annealing. The number of coefficients $c_{kl}$ is quadratic in the number of atoms, $N_{\rm at}$, and is of the order of $100N_{\rm at}^2$. So in a molecular dynamics simulation the calculation of the 3n derivative components $\partial E/\partial R_k$ is cheap compared with the calculation of $\sim 100 \times N_{\rm at}^2$ values of $\partial E/\partial c_{kl}$.

There are two aspects of the Car-Parrinello algorithm that should be noticed. The first is that they use a method using Fast Fourier Transforms (FFT) to calculate $\partial E/\partial c_{kl}$ in an efficient way. This method is specific to the plane-wave basis set. The FFT method can be used with minimization strategies of $E_{\{Z_j,\vec{R}_j\}}[\rho]$ other than simulated annealing. [7] The second is that in any minimization a good initial guess is always extremely helpful. With only small modifications of $\vec{R}_j$ from one step to the next, the $\rho_{\rm min}(\vec{r})$ of one step is a good guess for the next step. The first aspect is sufficiently important to ensure that the large majority of first principles molecular dynamics simulations are done with a plane-wave basis set.

## Solving Kohn-Sham equations with a finite basis set

In Kohn-Sham DFT the universal density functional is written as

$$F[\rho] = T_s[\rho] + \frac{1}{2}\iint \frac{\rho(\vec{r})\rho(\vec{r}')}{|\vec{r}-\vec{r}'|}\, d^3r\, d^3r' + E_{\rm xc}[\rho], \tag{7}$$

where $T_s[\rho]$ is the ground state kinetic energy of a non-interacting electron system which has a ground state charge density $\rho(\vec{r})$. The exchange and correlation energy functional $E_{\rm xc}[\rho]$ is defined by that equation. If we write the charge density as a sum of the squares of one particle wavefunctions, $\rho(\vec{r}) = \sum_k |\psi_k(\vec{r})|^2$ and minimize the energy functional with respect to the wavefunctions under the constraint that they remain orthonormal,

$$\int \bar{\psi}_i(\vec{r})\psi_j(\vec{r})\, d^3r = \delta_{ij}, \tag{8}$$

we obtain an Euler-Lagrange equation that is similar to a one particle Schrödinger equation

$$-\frac{\hbar^2}{2m}\nabla^2\psi_i(\vec{r}) + v_{\rm eff}(\vec{r};[\rho])\psi_i(\vec{r}) = \epsilon_i\psi_i(\vec{r}). \tag{9}$$

However, the effective potential $v_{\rm eff}$ has a non-linear dependence on the electron density,

$$v_{\rm eff}(\vec{r}) = v_{\rm ext}(\vec{r}) + \int \frac{\rho(\vec{r})\rho(\vec{r}')}{|\vec{r}-\vec{r}'|}\, d^3r' + \frac{\delta E_{\rm xc}}{\delta\rho(\vec{r})} = v_{\rm ext}(\vec{r}) + v_{\rm H}(\vec{r}) + v_{\rm xc}(\vec{r}) \tag{10}$$

where the three terms are the external, Hartree and exchange and correlation potentials. This equation is solved iteratively: we assume an input electron density $\rho^{\text{in}}$ and calculate the input effective potential $v_{\text{eff}}^{\text{in}}$; this potential is used to solve for the now linearized Eq. (9) and obtain the output orbitals. These are used to construct a new input electron density and the process is iterated until self-consistency is achieved. The reader can find the details on how to calculate $v_{\text{eff}}$ on other chapters of this volume.

If one expands the orbitals in a finite basis-set,

$$\psi_i(\vec{r}) = \sum_j^M c_{ij}\phi_j(\vec{r}) \tag{11}$$

then the Euler-Lagrange equation becomes a matrix equation of the form

$$\sum_j c_{ij}[H_{kj} - \epsilon_i S_{kj}] = 0 \tag{12}$$

where

$$H_{kj} = \int \bar{\phi}_k(\vec{r})\Big(-\frac{\hbar^2}{2m}\nabla^2\psi_i(\vec{r}) + v_{\text{eff}}(\vec{r};[\rho])\Big)\phi_j(\vec{r})\, d^3r \tag{13}$$

and

$$S_{kj} = \int \bar{\phi}_k(\vec{r})\phi_j(\vec{r})\, d^3r \tag{14}$$

## Gaussian Orbitals

One of the most popular basis-sets uses Gaussian basis-functions

$$\phi_i(\vec{r}) = N_i \exp(-\alpha_i(\vec{r} - \vec{R}_{s(i)})^2) Z_{l(i)}^{m(i)}(\vec{r} - \vec{R}_{s(i)}) \tag{15}$$

where the angular funtions $Z_l^m$ are chosen to be real solid harmonics (real polynomials of order $l$), $N_i$ are normalization factors, and $\alpha_i$ is an exponent that controls the size of the function. These functions are centered at $\vec{R}_{s(i)}$, where $s(i)$ is the mapping between the index of the basis function and the atomic site where it is centered. The reason for the popularity of gaussian basis sets is that they are sufficiently flexible to describe atomic orbitals with a few functions, and the integrals with gaussian functions can often be written in an explicit analytical form.

We can also use gaussian basis functions to represent the components of the effective potential. [8–10] These sets are not the same as those for the orbitals, but here for the sake of simplicity of notation we will assume they are the same. They are used to represent the input charge density

$$\rho^{\mathrm{in}}(\vec{r}) = \sum_i a_i \phi_i(\vec{r}) \tag{16}$$

and the input exchange and correlation potential

$$v_{\mathrm{xc}}^{\mathrm{in}}(\vec{r}) = \sum_i b_i \phi_i(\vec{r}) \; . \tag{17}$$

The contribution of the effective potential to the hamiltonian $H_{ij}$ is

$$\begin{aligned} V_{ij} &= \int \phi_i(\vec{r}) v_{\mathrm{eff}}(\vec{r}, [\rho^{rmin}]) \phi_j(\vec{r}) \, d^3r \qquad (18) \\ &= \sum_k \int \phi_i(\vec{r}) \Big(-\frac{Z_j}{|\vec{r}-\vec{R}_j|}\Big) \phi_j(\vec{r}) \, d^3r + \sum_k a_k \iint \frac{\phi_i(\vec{r})\phi_k(\vec{r}')\phi_j(\vec{r})}{|\vec{r}-\vec{r}'|} d^3r \, d^3r' \\ &+ \sum_k b_k \int \phi_i(\vec{r})\phi_k(\vec{r})\phi_j(\vec{r}) \, d^3r \\ &= \sum_k V_{ijk}^{\mathrm{ext}} + \sum_k A_{ijk}^{\mathrm{H}} a_k + \sum_k B_{ijk}^{\mathrm{xc}} b_k \end{aligned}$$

In the course of the self-consistent iterations, the coefficients $a_k$ and $b_k$ change their value, but $V_{ijk}^{\mathrm{ext}}$, $A_{ijk}^{\mathrm{H}}$ and $B_{ijk}^{\mathrm{xc}}$ remain the same. They can be evaluated only once at the beginning of the calculation. As a result calculating the contribution of the effective potential to the hamiltonian requires only array multiplications. Moreover, the diagonalization of $H_{ij}$ is also a question of linear algebra. As the indexes $i, j$ and $k$ can reach to several hundred, the size of the three-index arrays $X_{ikj}$ ($X = V^{\mathrm{ext}}, A^{\mathrm{H}}, B^{\mathrm{xc}}$) requires a huge amount of memory. Although analytical, the calculation of each of the $X_{ikj}$ is non-trivial and requires a reasonable number of floating point operations. Compared with the calculation of these tables, and the access to the tables to construct the hamiltonian $H_{ij}$, the diagonalization of $H_{ij}$ is fast, requiring only a few percent of the total computing time. This is just the opposite of the situation for a typical plane-wave code where setting up the hamiltonian takes a few percent of the time and diagonalization takes most of the computing time. Furthermore the calculation of forces with gaussian basis sets requires the implementation of Pulay corrections [6] for basis sets incompleteness, which is not a trivial task.

As a result, in a finite basis set, the Monte Carlo methods can become competitive with Molecular Dynamics methods. On the following we present our implementation of such a Monte Carlo scheme.

## FIRST PRINCIPLES MONTE CARLO

### Implementation

To obtain an efficient First Principles Monte Carlo algorithm it is not enough to pass the energies calculated with a first principles method to an existing Monte Carlo program. Indeed it is crucial to find a synergy between the two methods.

Once $E_{GS}(\vec{R}_1, \ldots, \vec{R}_{N_{at}})$ is obtained for a given molecular configuration, the Monte Carlo Simulated Annealing algorithm "decides" upon the next move. This procedure will be repeated many thousands of times before an annealed struture is obtained, hopefully corresponding to the global minimum of $E_{GS}$. When moving from one Monte Carlo iteration to the next, the Simulated Annealing algorithms typically change the coordinates of one single atom, $\vec{R}_i \rightarrow \vec{R}_i + \delta\vec{R}$. As the basis set functions are centered on the atomic sites (see Eq. 15), only those functions that are centered on atom $i$, that is, those that satisfy $s(j) = i$, change with the Monte Carlo move. The coefficient $X_{jkl}$ will not change with the Monte Carlo step if $s(j) \neq i$ and $s(k) \neq i$ and $s(l) \neq i$, and does not need to be recalculated. In this way, only a fraction of the order of $1/N_{\mathrm{at}}$ of the total number of integrals $X_{ijk}$ needs to be recalculated, leading to a substantial saving in computer time. A crucial point is that as the system size increases a smaller fraction of integrals have to be recalculated. In Fig. 1 we show schematically the three dimensional arrangement of a three index array $X_{ijk}$. Each index is associated with an atom in the molecule. The shaded region indicates the elements of the array where at least one of the indices is associated with a given atom, and therefore only those elements have to be recalculated when that atom moves.

When one starts a self-consistent calculation, an "educated guess" for the starting potential or charge density is required. In our case we use for Monte Carlo iteration $n+1$ the self-consistent density obtained from iteration $n$. In this way, in *all* but the start-up Monte Carlo iteration, a small number of iterations is required to attain

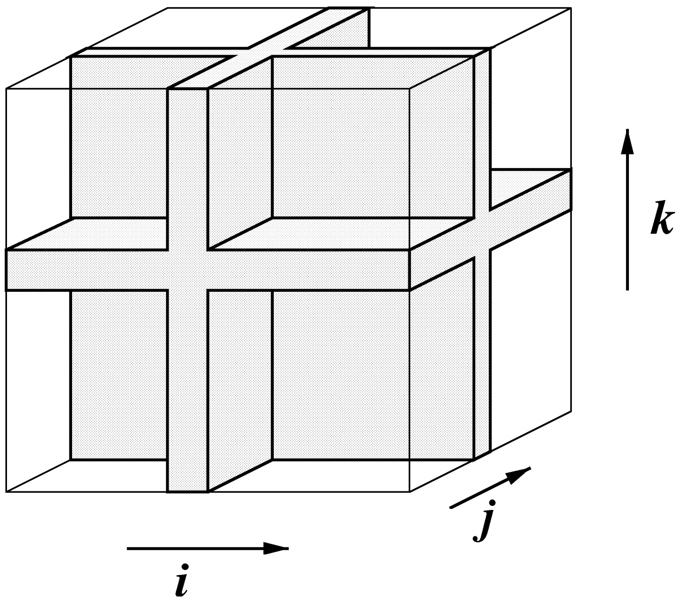

**FIGURE 1.** The arrangement of elements of a three index array $X_{ijk}$ is shown schematically. The shaded area shows the array elements that have at least one index that is associated with one given atom.

self-consistency, resulting in further savings in computer time. It is this coupling between the Monte Carlo and DFT parts of the code that allow us to have a highly efficient code which enables us to run simulations in which the self-consistent energy of a large cluster needs to be computed many thounsands of times.

## Parallelism

Since each of the $X_{jkl}$ is independent from the others, we can use the modular structure of the program in order to parallelize the code and distribute both the tasks and the storage among the available processors. We recast each matrix-element $V_{ij}$ in the form

$$V_{ij} = \sum_{\lambda=1}^{N_{\rm proc}} V_{ij}[\lambda] \tag{19}$$

where the indexed $V_{ij}[\lambda]$ will be evenly distributed among the $N_{\rm proc}$ processes executing the program, so that processor $\lambda$ contains only the non-zero elements of $V_{ij}[\lambda]$ and no other elements of $V_{ij}$. Similarly, the three-index array $X_{ikj}$ is distributed as

$$X_{ikj} = \sum_{\lambda=1}^{N_{\rm proc}} X_{ikj}[\lambda] \tag{20}$$

in such a way that $X_{ikj}[\lambda]$ is null if $V_{ij}[\lambda]$ is null. Of course, the null elements are not stored so the large array is distributed among all the processes, which makes the code optimal for parallel architectures based on distributed memory. As

$$V_{ij}[\lambda] = \sum_{k=1}^{L} a_k(\{c_{ij}\}) A_{ikj}[\lambda] + \ldots \tag{21}$$

there is no need to exchange the values of $X_{ikj}$ among processes, but only those of $a_k$ before summation, and $V_{ij}[\lambda]$ after the summation. As a result, the calculation of $X_{ikj}$ is split among the processes, the storage is also distributed, and $X_{ikj}$ never appears in the communications. A detailed analysis of the parallelization issues will be published elsewhere. [11]

## Example

We have applied the method described above to the optimization of Na clusters. Fig. 2 shows the total energy of the cluster as a function of MC step for a $\mathrm{Na}_{21}^{+}$ cluster (see inset). This is a "magic number" cluster, for which the optical response has been measured. At first the MC temperature is high to randomize the cluster. Then the temperature is reduced as can be seen from the decreasing fluctuations of the cluster energy. A J-walk algorithm [12,13] has been used to increase the frequency of barrier jumping processes. The more than 60000 total energies were efficiently calculated in a departmental workstation farm.

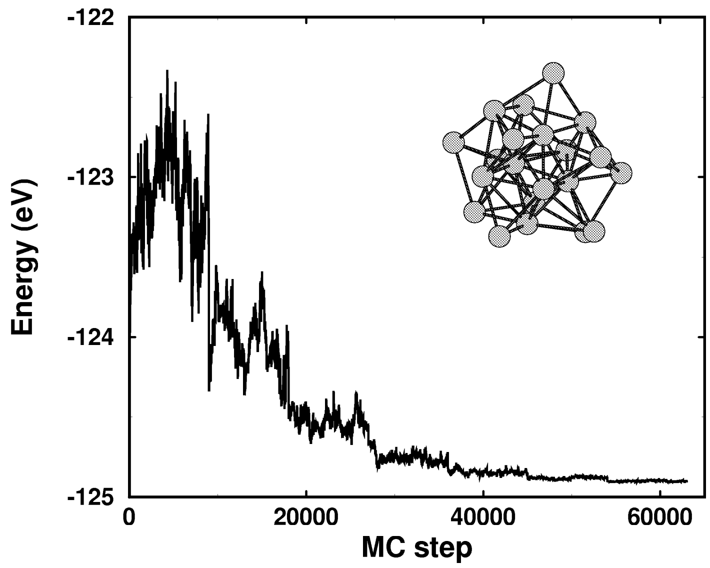

**FIGURE 2.** Evolution of the total energy of $Na_{21}^{+}$ as a function of MC step during a simulated annealing run

## CONCLUSIONS

We have successfully implemented a first principles Monte Carlo scheme for the calculation of the equilibrium structures of molecules and clusters. Combining the fact that a typical Monte Carlo step moves only one atom at a time with the use of atom centered functions to represent both wave-functions and potentials, we were able to reuse most of the molecular integrals calculated in a previous iteration, with important savings in computer time. Moreover, the modular structure of the calculation of the molecular integrals allowed an easy parallelization of that part of the computation.

We use a gaussian basis set, but the method can be used with any atom-centered basis set, in as much as the potentials are also expanded in atomic centered functions. The current bottleneck is the large amount of molecular integrals that are calculated and stored. As the size of the cluster increases it is possible to decide a priori if a whole class of integrals needs to be calculated. We have implemented this strategy for gaussians and found modest gains for the example discussed. However for larger clusters the savings from such a procedure will be more important. Notice that if we had a finite range atomic basis set the number of integrals would ultimately scale linearly in the number of atoms.

## ACKNOWLEDGEMENTS

This work was support by the Portuguese Ministry of Science and Technology under contracts PRAXIS/C/FIS/10019/98 (JMP) and PRAXIS/PCEX/FIS/11213/98 (JLM).

## REFERENCES

1. Allen, M.P., and Tildesley, D.J., *Computer Simulation of Liquids*, Oxford: Clarendon Press, 1987. Ciccoti, G., Frenkel, D., and McDonald, I.R., eds., *Simulation of Liquids and Solids. Molecular Dynamics and Monte Carlo Simulations in Statistical mechanics*, Amsterdam: North Holland, 1987. Binder, K., and Heermann, D.W., *Monte carlo Simulation in Statistical Physics*, Berlin: Springer-Verlag, 1988. Rappaport, D.C., *The Art of Molecular Dynamics Simulations*, Cambridge: Cambridge University Press, 1995.
2. Car, R., and Parrinello, M., *Phys. Rev. Lett.* **55**, 2471 (1985).
3. Hohenberg, P., and Kohn, W., *Phys. Rev.* **136** , B864 (1964).
4. Kohn, W., and Sham, L.J., *Phys. Rev.* **136** , A1133 (1965).
5. Pickett, W.E., *Computer Physics Reports* **9**, 115 (1989).
6. Pulay, P., *Applications of Electronic Structure Theory, Modern Theorethical Chemistry*, edited by Schaeffer, H.F., Vol. 4, p. 153, New York: Plenum, 1977.
7. Martins, J.L., and Cohen, M.L., *Phys. Rev. B* **37**, 6134 (1988).
8. Sambe, H., and Felton, R.F., *J. Chem. Phys.* **62**, 1122, (1975).
9. Dunlap, B.I., Connolly, J.W.D., and Sabin, J.R., *J. Chem. Phys.* **71**, 3396, (1979).
10. Martins, J.L., Buttet, J., and Car, R., *Phys. Rev. B* **31**, 1804 (1985). Car, R. and Martins, J.L., *Surface Sci.* **106**, 280 (1981).
11. Pacheco, J.M., and Martins, J.L., *Lecture Notes in Computer Science*, Springer (in press)
12. Franz, D.D., Freeman, D.L., and Doll, J.D., *J. Chem. Phys.* **93**, 2769, (1990).
13. Pacheco, J.M. (unpublished).