Improved Grid-Based Algorithm for Bader Charge Allocation

EDWARD SANVILLE,[1]* STEVEN D. KENNY,[1] ROGER SMITH,[1] GRAEME HENKELMAN[2]
[1]*Department of Mathematical Sciences, Loughborough University, Loughborough LE11 3TU, United Kingdom*
[2]*Department of Chemistry and Biochemistry, University of Texas at Austin, Austin, Texas 78712*

Received 21 August 2006; Revised 29 September 2006; Accepted 2 October 2006
DOI 10.1002/jcc.20575
Published online 19 January 2007 in Wiley InterScience (www.interscience.wiley.com).

Abstract: An improvement to the grid-based algorithm of Henkelman et al. for the calculation of Bader volumes is suggested, which more accurately calculates atomic properties as predicted by the theory of Atoms in Molecules. The CPU time required by the improved algorithm to perform the Bader analysis scales linearly with the number of interatomic surfaces in the system. The new algorithm corrects systematic deviations from the true Bader surface, calculated by the original method and also does not require explicit representation of the interatomic surfaces, resulting in a more robust method of partitioning charge density among atoms in the system. Applications of the method to some small systems are given and it is further demonstrated how the method can be used to define an energy per atom in *ab initio* calculations.

 J Comput Chem 28: 899–908, 2007

Key words: Bader charge; Bader volumes; atomic energy; numerical integration

Introduction

According to Richard Bader's theory of Atoms in Molecules (AIM),[1] a molecule can be partitioned into atomic volumes, such that the flux of the gradient of the electron density through the interatomic surfaces vanishes at every point on the surfaces. Within such volumes, it can be shown that the atomic subsystems obey a local virial relation. Also, certain properties, such as an atomic charge, atomic dipole moment, and atomic kinetic and potential energies, are shown by the AIM theory to take on quantities that are well-defined by quantum mechanics, if they are integrated over these atomic subvolumes.[2] This leads to the useful ability to partition charge, energy, and other properties unambiguously over the atoms composing a chemical system. Unlike other charge partitioning schemes, such as the Mulliken population analysis, the AIM results are not highly sensitive to the basis set used in the calculation.[3,4] In particular, the AIM method does not require the assignment of each basis function to a specific atom. This makes AIM particularly advantageous in situations where this assignment would be difficult or impossible, such as with plane-wave basis sets. Additionally, the AIM analysis can be performed, using either theoretical or experimentally determined charge densities.

The principle obstacle to the integration of AIM properties has been the fast and accurate calculation of the atomic volumes, or equivalently their bounding interatomic surfaces. The first reported calculations involved a coordinate transformation to an atom-centered, distorted coordinate system, in which the integration becomes unbounded.[5] Despite the relative simplicity of this scheme, in practice it tends to be very computationally expensive. More common have been attempts to represent the interatomic surface by a function $r(\theta, \phi)$, where θ and ϕ are polar coordinates centered on each atomic nucleus and r represents the radial coordinate from the nucleus to the interatomic surface in question. An early method to approximate this function, used by the program PROAIM,[6] determines the interatomic surface with a set of points, which are obtained by following the electron density gradient downhill from bond critical points. Interpolations are performed between the point mesh. The point density within the mesh tends to be quite uneven, leading to interpolations of r that are also uneven in quality, with respect to θ and ϕ. In practice this can lead to "density leakage," which becomes more severe in molecules containing sharply curving interatomic surfaces. A later approach involves approximating the interatomic surfaces, by representing them as analytical functions of the angular coordinates of a spherical or curvilinear coordinate system in real space. In particular, an early algorithm used a Fourier–Chebyshev fit to a numerically obtained interatomic surface to obtain an analytical

***Correspondence to:** E. Sanville; e-mail: e.sanville@lboro.ac.uk

approximation to the surface.[7] Another algorithm uses a variational procedure to calculate an analytical solution, by varying a vector of parameters to eliminate the electronic density flux residual over a quadrature of points.[8] These and other algorithms have been critically examined by Popelier.[9]

Although much work has been published with respect to these methods of Bader volume calculation, they remain complicated to implement. In practice, they also tend to lack robustness across a wide range of electron density topologies. Attempts to represent the interatomic surfaces as functions of nucleus-centered angular coordinates are complicated by multiple ray-surface intersections. This essentially makes the interatomic surface a multivalued function. Complex algorithms have been developed to deal with this commonly encountered situation.[10] More recently, the "octal tree search algorithm" of computer graphics was used to analyze the topologically complicated Laplacian of the electron density.[11] Recently, an entirely different approach was suggested by Henkelman et al.[12] Instead of explicitly representing the interatomic surfaces, each volume element of the system was assigned to an atomic volume by tracing an approximate electron density gradient vector trajectory from the center of the volume element back to an atomic nucleus, and assigning the volume element to that atom. During the gradient vector trace, the position vector was constrained to an integral multiple of the grid spacing in all three-dimensions. Large speed increases could be achieved by this method, because volume elements could be assigned to atomic volumes early in the gradient tracing process, if the position vector corresponded to a volume element that had already been assigned. The original algorithm reported by Henkelman et al.,[12] demonstrated some conceptual and practical advantages over previously reported algorithms. Although the original algorithm was described in terms of a regular cubic mesh, it is generally applicable to any type of nonorthogonal regular mesh. The conceptual simplicity of the algorithm and its ease of implementation lead to very robust behavior, independent of the actual topology of the electronic charge density field. The algorithm can easily handle Bader volumes, which do not contain a nucleus. Additionally, multiple ray-surface intersections are not an issue. The original algorithm is summarized below.

The input data for the algorithm is a three-dimensional regular Cartesian mesh of values of the electron density ρ defined over a rectangular volume of space V. This volume should be large enough so that the value of ρ is negligible outside this region. The volume V is subdivided into M smaller volumes δV_i, $i = 1 \ldots M$ centered around each mesh point. Each point at the center of δV_i is given by $r_i = l\Delta x\hat{\mathbf{i}} + m\Delta y\hat{\mathbf{j}} + n\Delta z\hat{\mathbf{k}}$, where l, m, and n, are integers that indicate the discrete numbering of the mesh points and Δx, Δy, Δz are the mesh spacings along each Cartesian direction. The idea is to assign each volume element δV_i to a Bader volume associated with a particular atomic nucleus or in rare cases to a Bader volume associated with a local maximum in ρ, which is not a nucleus. The following algorithm is performed to assign the mesh points to a Bader volume:

1. Start to scan through the set of volume elements δV_i.
2. Initialize a list of volume elements A to be empty. Let $\mathbf{r}_i$ be the mesh point at the center of the next volume element to be assigned. If there are no more volume elements to be assigned, end program.
3. If the value of ρ at $\mathbf{r}_i$ is a local maximum then terminate the current search and assign the volume element δV_i to the Bader volume associated with that maximum. If δV_i has already been assigned to a Bader volume associated with a local maximum, then assign all the volume elements stored in the list A to that Bader volume and go to step 2 at the next mesh point to be chosen. This can be done by increasing l, m, and n sequentially ignoring points that have already been assigned.
   There are 26 grid points immediately adjacent to each grid point. This is because there are 26 nonzero values of $\Delta\mathbf{r} = l\Delta x\hat{\mathbf{i}} + m\Delta y\hat{\mathbf{j}} + n\Delta z\hat{\mathbf{k}}$ such that l, m, $n \in \{-1,0,1\}$. For all 26 adjacent grid points $\mathbf{r}_i + \Delta\mathbf{r}$, define an approximation to the component of the electron density gradient in the direction of $\Delta\mathbf{r}$ by
$$\rho_{\Delta\mathbf{r}} \equiv \frac{\rho(\mathbf{r}_i + \Delta\mathbf{r}) - \rho(\mathbf{r}_i)}{|\Delta\mathbf{r}|}. \quad (1)$$
4. The location of the current mesh point is stored in memory in the list A. Let $\mathbf{r}_i = \mathbf{r}_i + \Delta\mathbf{r}$ for the $\Delta\mathbf{r}$ with the highest computed value of $\rho_{\Delta\mathbf{r}}$. Go to step 3.

The earlier algorithm, however, does not always converge to the correct Bader volumes in the limit of an infinitely fine mesh. In particular, it does not approximate the true Bader surface in areas where the electron density gradient is constant or slowly changing. In these areas, which tend to lie farther from the nuclei, the algorithm tends to "snap" the electron density gradient trajectories to one of the 26 directions of $\Delta\mathbf{r}$, irrespective of the mesh density. The cause of this snapping is illustrated in Figure 1a. The results of a Bader volume calculation using this grid-based algorithm, performed on a two-dimensional scalar field is shown in Figure 1a. The scalar field was composed of the sum of four different two-dimensional Gaussians centered at the four vertices of a square.

This dependence of the calculated Bader volume on the orientation of the mesh axes can be demonstrated by calculating the Bader charge of an atom in a molecule at various orientations of the mesh axes and at various mesh densities. The results of these calculations for the water molecule are shown in Figure 2a. In these calculations, the water molecule is rotated by between 0° and 90° with respect to the grid axes, and the Bader charge of the oxygen atom in the water molecule is calculated at mesh spacings of 0.16, 0.18, and $0.20a_0$. From this graph, it can be seen that the original grid-based algorithm gives results, which vary by up to $0.12e$, depending upon the orientation of the water molecule with respect to the axes.

Improved Algorithm

The intention of the improved algorithm is to correct the dependence of the original method upon the orientation of the system with respect to the mesh axes, and to converge to the correct Bader volumes in the limit of an infinitely fine mesh. To eliminate

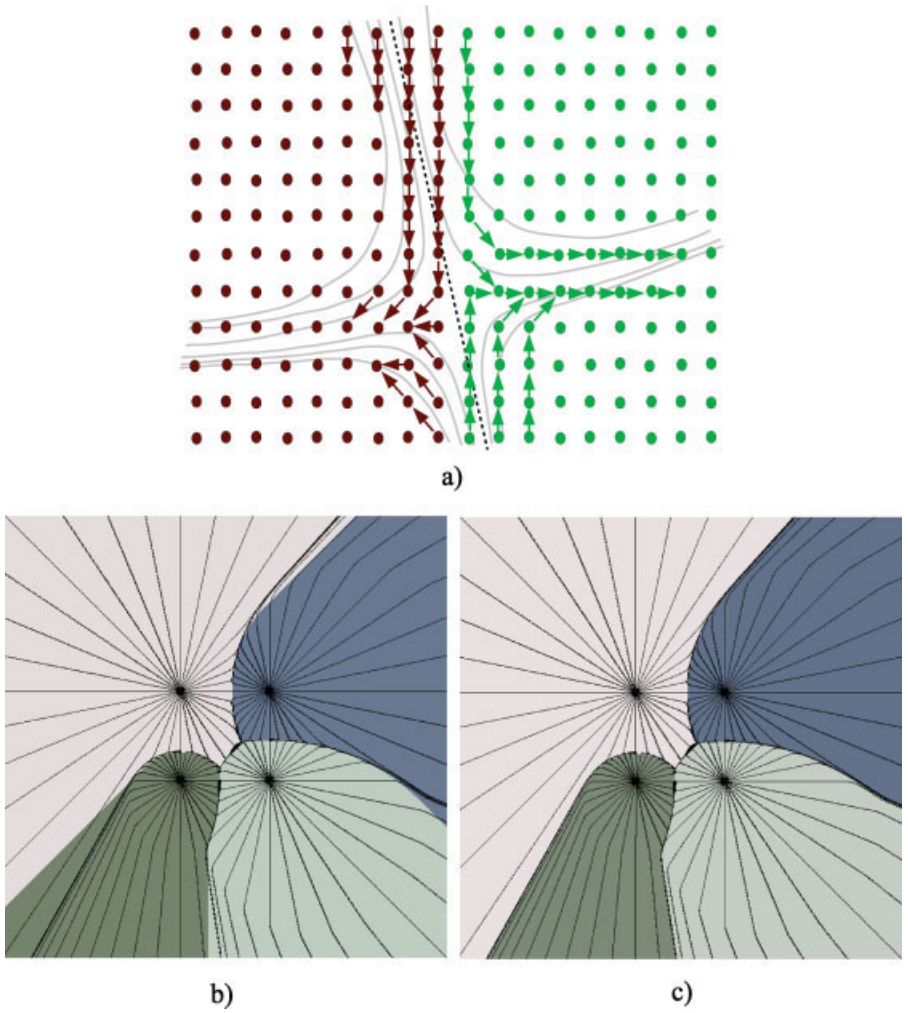

Figure 1. (a) A schematic illustrating deviation in the original grid-based algorithm near an interatomic surface. The mesh points are illustrated as circles, with fill colors representing the Bader volume assignment. The solid lines represent the gradient vector trajectories, and the dotted line illustrates the true Bader interatomic surface. The arrows represent the $\Delta\mathbf{r}$ vectors. Bader volume assignments are shown in (b) and (c), for the original grid-based algorithm, and the algorithm reported in this article, respectively. These calculations were performed in two-dimensions, using a density field composed of the sum of four Gaussian functions arranged in a square.

dependence of the computed charges on the orientation of the system, the gradient trajectory is followed freely in space, with a step size equal to the mesh interval, without restricting its direction to one of the 26 discrete values of $\Delta\mathbf{r}$. This allows a perfect trace of the gradient trajectory in the limit of an infinitely fine mesh, and therefore an accurate assignment of every mesh point to the proper Bader volume. The new (and the original) algorithm can be applied to non-Cartesian grid with different spacing along three nonplanar directions but for the sake of simplicity we restrict the description again to Cartesian meshes with a regular spacing.

As the vector $\mathbf{r}$ moves through space, tracing the gradient vector trajectory, it enters and leaves cubes of volume that correspond to the closest mesh point. When the gradient vector trajectory is followed to a point where it is located in a mesh cube that is completely surrounded by mesh cubes that have already been assigned to a particular Bader volume, it is assumed that the currently followed gradient trajectory eventually leads to the same maximum in the charge density. At this point, each mesh cube that has been passed through on the current gradient trajectory is also assumed to be a part of the same Bader volume, and assigned accordingly.

A second improvement to the original algorithm is made, in which the calculation is performed on a coarse mesh initially, with the mesh becoming finer at each subsequent step of the calculation. This allows larger volumes of space to be assigned to a particular Bader volume in early steps, with finer details being worked out at later steps in the calculation.

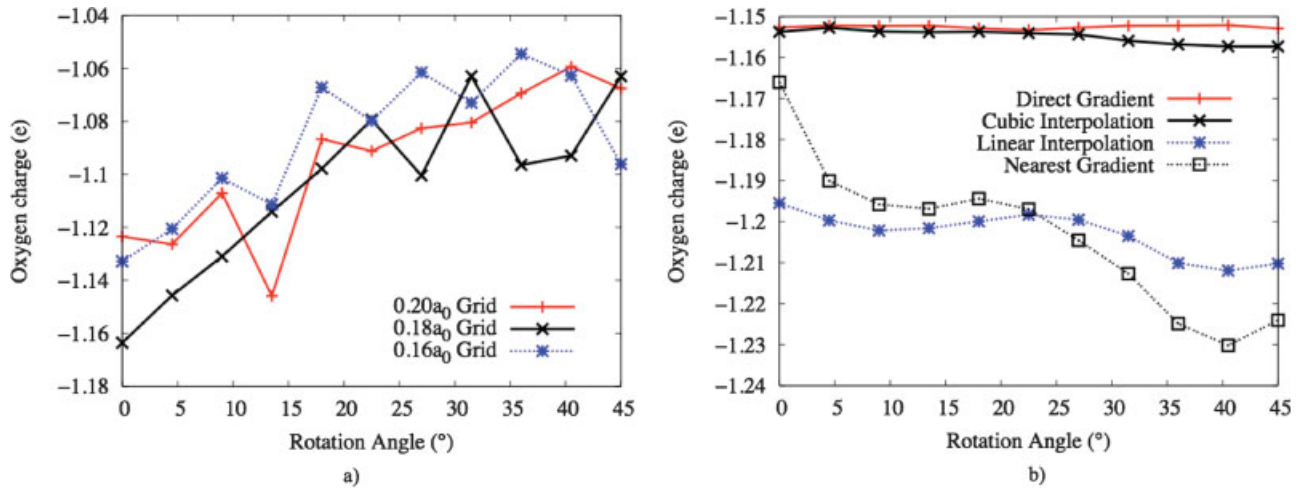

Figure 2. Bader charge on the oxygen atom in a water molecule, calculated by integration over the Bader volume (a) using the original grid-based algorithm, and (b) the new algorithm with various methods of gradient calculation. All calculations use an electron density obtained from calculations, using the LDA functional with PLATO.[13] The calculated oxygen Bader charges are plotted with respect to the rotation of the water molecule around the z-axis of the coordinate system. The series in (a) correspond to regular meshes with spacings of 0.20, 0.18, and 0.16 Bohr radii. The series in (b) correspond to direct gradient evaluation, and cubic and linear interpolation of the gradient, as well as the nearest gradient scheme. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

The third improvement is performed during the final integration, where a random sampling technique is used within all volume elements that are still not completely surrounded by elements assigned to the same Bader volume. The dividing surface between atoms passes through these cubes. To obtain a more accurate charge assignment, points in the volume element are sampled randomly. Each randomly selected point is then gradient-traced until the point is unambiguously assigned to an atomic volume as described earlier. The contribution of the volume element to the atomic properties is then divided up proportionally among the atoms to which the random samples have been assigned.

In the following pseudocode, the same variables are used as in the previous section. However, for simplicity we define

$$\delta w \equiv \Delta x = \Delta y = \Delta z \tag{2}$$

as a scalar equal to the mesh spacing in the Cartesian directions (x,y,z). A small constant δg is chosen as the gradient trajectory step size for the duration of the procedure, typically one half of the mesh spacing of the original data. A second constant δQ is defined as a precision parameter used during the random sampling integration procedure. It corresponds to the amount of subcharge in a volume element that we want to be assigned by the charge allocation method. Assuming a typical desired Bader charge precision of around +/− 0.001 electrons, the optimal value of δQ determined heuristically for this precision turns out to be about 1% of this value. However, as δQ is used in a random sampling process, it would be expected that the error would decrease with the inverse of the square root of the number of sampled points. Since the number of sampled points is inversely proportional to δQ, the error should increase as the square root of δQ. In addition to these precision parameters a set B is defined, which has the same number of elements as there are distinct maxima in ρ. When all volume elements are allocated, each element of B is the set of volume elements associated with each distinct maxima of ρ and is thus the Bader volume associated with that maximum point. The full improved algorithm is given below:

1. From the supplied data points on the grid, set up a coarse mesh using a large value of δw. Initialize B to be empty. Define a mesh length value δw_0, which defines the allowable accuracy of the spatial variation in the Bader volumes. This should be greater than or equal to the mesh spacing given in the input data set.
2. Initialize a set of volume elements A to be empty. Let $\mathbf{r}_i$ be the mesh point at the centre of the next element δV_i to be assigned. This is chosen as in the case of the original algorithm. If there are no volume elements to be assigned, go to step 3.
3. If all 26 cubes surrounding δV_i are already assigned to the same Bader volume (i.e., element of B), then assign δV_i to that Bader volume. If $\mathbf{r}_i$ is located within a distance δw of an atomic nucleus, then assign δV_i to the Bader volume associated with that atomic nucleus. If ρ has local maximum at $\mathbf{r}_i$ but $\mathbf{r}_i$ is not located within a distance δw of any atomic nucleus, assign the δV_i to a Bader volume associated with a non-nuclear attractor. Assign all the volume elements stored in the list A to the same Bader volume as δV_i and go to step 2. Otherwise, add δV_i to the list A.

4. Let

$$\Delta \mathbf{r} \equiv \frac{\delta g \nabla \rho(\mathbf{r})}{|\nabla \rho(\mathbf{r})|}. \quad (3)$$

   Let $\mathbf{r} = \mathbf{r}_i + \Delta \mathbf{r}$. Determine the new volume element δV_i, which contains $\mathbf{r}$. Go to step 3.
5. If $\delta w \leq \delta w_0$, go to 5.
6. Let $\delta w = \delta w/2$. From each old volume element, eight new cubes are produced initially allocated to one of the maxima in ρ. Those new cubes which are completely surrounded by cubes allocated to the same maximum are kept in the appropriate element of B. Those that are not so surrounded are erased from association with a maximum (i.e., removed from that element of B) and considered as unallocated for further investigation. Go to 2.
7. Integrate the electron density over all mesh volumes in the mesh. Let δq_i be the total charge contained within the ith volume element. For each volume element that is completely surrounded by volume elements with the same Bader volume assignment, assign δq_i to that Bader volume. Otherwise, define an integer N by

$$N = \text{int}\left\lceil \frac{\delta q_i}{\delta Q} \right\rceil. \quad (4)$$

Choose N random points in δV_i and trace the gradient trajectory for each point as in step 3, until reaching a volume element that is completely surrounded by volume elements with the same Bader volume assignment. Once this occurs, assign $\delta q_i/N$ to that Bader volume. End the program.

It is important to carry out the charge allocation given in step 5 for elements with trajectories that can be traced back to different Bader volumes. If this is not done, significant systematic errors can occur in the Bader charge integration in certain circumstances. For example, if the true Bader surface is planar or nearly planar, and runs parallel to the xy-plane of the mesh coordinate system, significant errors would result unless ρ was defined on a very fine grid. This is because all of the mesh cubes bisected by this plane would be assigned to either one or the other of the adjacent Bader volumes, depending upon which Bader volume contained the mesh points in this plane.

To obtain the gradient vectors, several methods can be used. The "Direct" method, calculates the gradient vector, using the electronic orbital coefficients. The second method involves interpolation between points in a regular mesh, to obtain each of the three gradient vector components. In this article, interpolations between a mesh of gradient vector components were performed, using a three-dimensional cubic polynomial interpolation algorithm.[13] This cubic interpolation algorithm is referred to as "Cubic interpolation." For comparison, calculations are also reported by using two simpler algorithms to calculate the gradient vector components. In the first method, the gradient vector is simply taken as the gradient vector of the closest grid point. This algorithm is referred to as "Nearest gradient." In the final method, a simple trilinear algorithm was used to calculate the gradient vector components. This algorithm is referred to as "Linear interpolation."

In rare cases where Bader charge has been allocated to a maximum ρ value, which is not an atom, there are two different courses of action. The choice of this method to use is dependent upon the specific application. In the first method, each of the volume elements making up the corresponding element of B are assigned individually to another element of B, which corresponds to the nearest atom to that volume element.

Results

The results of a Bader volume calculation performed, using the algorithm reported in this article is shown graphically in Figure 1b, using the same scalar field as shown in Figure 1a. A comparison of Figures 1b and 1c demonstrates that the new algorithm succeeds in assigning all points to their respective Bader volumes, even those which lie close to an interatomic surface.

A series of Bader charge calculations was performed on the oxygen atom in a water molecule at a mesh spacing of 0.2 Bohr. As shown in Figure 2b, the calculated oxygen Bader charges vary by approximately $0.001e$ with respect to rotation of the integral mesh axes, when the new algorithm is used with the Direct method of gradient vector calculation. The algorithm varies by less than $0.005e$, when the cubic interpolation scheme is used. The Nearest Gradient and Linear Interpolation schemes varied by approximately $0.08e$ with respect to rotation of the grid axes. The mean Bader analysis times for calculations performed on a water molecule with a mesh spacing of 0.2 Bohr radii, using all four gradient interpolation schemes are summarized in Table1. The cubic interpolation scheme requires approximately 20% of the CPU time that the direct gradient scheme requires.

Several molecules were analyzed, using the current integration algorithm. In each case, the molecule was fully geometry-optimized, using the density-functional theory Package for Linear Combination of Atomic Orbitals (PLATO),[14] before the Bader analysis was performed. For a subset of smaller molecules, the calculations were performed, using the cubic interpolation scheme, as well as with a slower direct evaluation of the gradient vector, for comparison. The results of these calculations are summarized in Table2. Further atomic charge calculations involving only the cubic interpolation scheme are reported in Table3.

The electron densities were calculated, using the full LCAO treatment and the LDA density functional. The core electrons were represented by the relativistic electronic core potentials of Hartwigsen et al.[15] for the elements H, C, O, and Si. For the remaining elements, the nonrelativistic pseudopotentials of Goedecker et al[16] were used. For all of the elements except hydro-

Table 1. Mean Bader Analysis Calculation Time for a Water Molecule, Using the Four Different Gradient Calculation Methods.

Gradient scheme	Bader analysis time (s)
Direct	354.2
Cubic interpolation	61.0
Trilinear interpolation	25.0
Nearest gradient	23.8

All calculations were performed on a 2.5 Ghz PowerPC G5 processor.

Table 2. Bader Charges on Various Atoms in Small Molecules, Analyzed with the Current Charge Integration Algorithm.

Atom	Net Bader charge (cubic interpolation)	Atomic energy (cubic interpolation)	Net Bader charge (direct gradient)	Atomic energy (direct gradient)
Ammonia				
N	−1.111	−20.829615	−1.112	−20.829704
H	0.370	−0.84956	0.370	−0.849559
H	0.370	−0.849073	0.371	−0.848998
H	0.370	−0.849067	0.370	−0.849053
Total	0.000	−23.377315	0.000	−23.377314
Carbon dioxide				
C	1.967	−10.310103	1.967	−10.310097
O	−0.984	−32.56825	−0.984	−32.568174
O	−0.983	−32.56829	−0.983	−32.568372
Total	0.000	−75.446643	0.000	−75.446643
Carbon disulfide				
C	−0.945	−11.463289	−0.941	−11.463491
S	0.472	−20.354888	0.471	−20.354857
S	0.473	−20.354921	0.470	−20.35475
Total	0.000	−52.173098	0.000	−52.173098
Epoxyethane				
O	−0.845	−32.487569	−0.844	−32.487565
C	0.296	−11.377862	0.296	−11.377894
C	0.297	−11.377806	0.297	−11.377798
H	0.063	−1.026462	0.063	−1.026466
H	0.063	−1.026431	0.063	−1.0264
H	0.063	−1.026472	0.063	−1.026438
H	0.063	−1.026479	0.062	−1.02652
Total	0.000	−59.349081	0.000	−59.349081
Formaldehyde				
O	−0.990	−32.382007	−0.987	−32.382071
C	0.869	−11.407462	0.866	−11.407419
H	0.061	−0.926921	0.061	−0.926863
H	0.061	−0.926868	0.061	−0.926905
Total	0.000	−45.643258	0.000	−45.643258
Hydrazine				
N	−0.709	−20.397229	−0.708	−20.397173
N	−0.710	−20.397202	−0.710	−20.397171
H	0.343	−0.913877	0.342	−0.913894
H	0.343	−0.913889	0.343	−0.91392
H	0.367	−0.866139	0.366	−0.86619
H	0.367	−0.866168	0.367	−0.866156
Total	0.000	−44.354504	0.000	−44.354504
Hydrogen peroxide				
O	−0.573	−32.303759	−0.575	−32.303605
O	−0.576	−32.303448	−0.576	−32.303428
H	0.574	−0.780503	0.575	−0.780635
H	0.575	−0.780638	0.575	−0.780681
Total	0.000	−66.168348	0.000	−66.168349
Methane				
C	−0.200	−12.24834	−0.202	−12.248537
H	0.050	−0.949933	0.050	−0.949881
H	0.050	−0.949776	0.051	−0.949718
H	0.050	−0.949792	0.051	−0.949753
H	0.050	−0.949829	0.051	−0.94978
Total	0.000	−16.04767	0.000	−16.047669
Methanol				
O	−1.048	−32.540578	−1.049	−32.540597
C	0.370	−11.754068	0.372	−11.753924

(*continued*)

Table 2. (*Continued*)

Atom	Net Bader charge (cubic interpolation)	Atomic energy (cubic interpolation)	Net Bader charge (direct gradient)	Atomic energy (direct gradient)
Alcohol H	0.559	−0.841434	0.559	−0.841445
Methyl H	0.060	−0.972826	0.060	−0.972893
Methyl H	0.029	−0.967279	0.029	−0.967348
Methyl H	0.029	−0.967332	0.029	−0.96731
Total	0.000	−48.043517	0.000	−48.043517
Methylamine				
N	−1.011	−20.539741	−1.011	−20.539699
C	0.230	−11.77167	0.230	−11.771758
Amine H	0.351	−0.909831	0.351	−0.90982
Amine H	0.351	−0.909834	0.351	−0.909851
Methyl H	0.006	−0.985429	0.005	−0.985525
Methyl H	0.036	−0.985956	0.037	−0.985842
Methyl H	0.037	−0.985824	0.037	−0.985791
Total	0.000	−37.088285	0.000	−37.088286
Nitrogen dioxide				
O	−0.314	−32.114714	−0.313	−32.11493
O	−0.312	−32.115132	−0.313	−32.115007
N	0.626	−19.398378	0.625	−19.398287
Total	0.000	−83.628224	0.000	−83.628224
Silane				
Si	2.330	−7.849469	2.330	−7.849485
H	−0.582	−1.153241	−0.582	−1.153225
H	−0.583	−1.15324	−0.583	−1.153257
H	−0.582	−1.153244	−0.582	−1.153248
H	−0.583	−1.153248	−0.583	−1.153228
Total	0.000	−12.462442	0.000	−12.462442
Sulfur dioxide				
S	2.182	−19.60388	2.181	−19.603563
O	−1.091	−32.405994	−1.090	−32.406387
O	−1.090	−32.406382	−1.090	−32.406307
Total	0.000	−84.416256	0.000	−84.416257
Water				
O	−1.152	−32.772041	−1.152	−32.77202
H	0.576	−0.787338	0.576	−0.78735
H	0.576	−0.787326	0.576	−0.787334
Total	0.000	−34.346705	0.000	−34.346704

The electron densities were calculated at the optimized geometries, using full SCF and the LDA functional. Gradient vectors were calculated by a cubic interpolation scheme, and directly for comparison. The atomic energies are also reported (in Rydbergs).

gen, a triple-numeric basis set was used with added polarization functions. For hydrogen, a double numeric polarized basis set was taken. Because the SCF included only valence electrons, the core electrons were assumed to remain completely localized for the purposes of the Bader charge calculations. A fixed, spherically symmetric core electron density function was used at each nucleus for the purposes of the gradient vector calculations. This core electron density was calculated as the difference between an all-electron density atomic calculation, and the corresponding atomic calculation, using the appropriate pseudopotential.

When the cubic interpolation scheme was used, the cubic interpolation was performed on a grid containing valence electron gradient components only. In these cases, the core electron density gradients were added on after the cubic interpolation was completed.

Bader charge analysis is important for a number of reasons. One application is that it allows the total energy of an atomic system to be apportioned uniquely to each atom in the system, defining a so-called "energy per atom." This is a concept that is quite useful when fitting empirical potentials to quantum mechanical calculations. This apportioning can be accomplished as follows.

Within the local density approximation, the total energy of a system is given by the equation

$$E_{\text{tot}} = \sum_n f_n \varepsilon_n + E_{\text{Ion-Ion}} + E_{\text{d.c.}} \tag{5}$$

Table 3. Bader Charges on Various Atoms in Molecules, Analyzed with the Current Charge Integration Algorithm.

Atom	Net Bader charge	Plato Bader atomic energy
Acetone		
O	−1.038	−32.69231
Carbonyl C	0.920	−10.42030
Methyl C	−0.131	−11.95030
Methyl C	−0.130	−11.95030
H	0.074	−1.03217
H	0.074	−1.03222
H	0.058	−1.02723
H	0.058	−1.02719
H	0.058	−1.02722
H	0.058	−1.02718
Total	0.000	−73.18642
Cyclobutane		
C	−0.033	−11.57325
C	−0.032	−11.57317
C	−0.032	−11.57305
C	−0.033	−11.57325
H	0.018	−1.03349
H	0.014	−1.10579
H	0.018	−1.03358
H	0.014	−1.10571
H	0.019	−1.03347
H	0.014	−1.10578
H	0.019	−1.03339
H	0.014	−1.10568
Total	0.000	−54.84960
Cyclopropane		
C	−0.085	−11.56238
C	−0.083	−11.56192
C	−0.086	−11.56225
H	0.043	−1.06900
H	0.042	−1.06947
H	0.042	−1.06935
H	0.043	−1.06908
H	0.042	−1.06950
H	0.043	−1.06928
Total	0.000	−41.10224
Benzene		
C	−0.049	−11.44412
C	−0.048	−11.44256
C	−0.048	−11.44256
C	−0.049	−11.44419
C	−0.048	−11.44246
C	−0.049	−11.44271
H	0.049	−1.10652
H	0.048	−1.11600
H	0.048	−1.11599
H	0.049	−1.10644
H	0.048	−1.11606
H	0.048	−1.11609
Total	0.000	−75.33570
Ethane		
C	−0.071	−11.89605
C	−0.071	−11.89599
H	0.024	−0.99729
H	0.024	−0.99923
H	0.024	−0.99926
H	0.024	−0.99739
H	0.024	−0.99914
H	0.023	−0.99931
Total	0.000	−29.78367
Dimethyl ether		
O	−0.993	−32.21170
C	0.386	−11.73560
C	0.386	−11.73553
H	0.058	−1.01175
H	0.058	−1.01176
H	0.026	−1.01232
H	0.026	−1.01236
H	0.025	−1.01238
H	0.026	−1.01235
Total	0.000	−61.75576
Ethanol		
O	−1.050	−32.69142
Alcohol C	0.421	−11.13652
Methyl C	−0.075	−11.99770
Alcohol H	0.554	−0.96628
Methylene H	0.017	−0.92801
Methylene H	0.017	−0.92793
Methyl H	0.032	−1.05746
Methyl H	0.041	−1.04450
Methyl H	0.041	−1.04451
Total	0.000	−61.79433
Formamide		
O	−1.063	−32.62141
C	1.278	−10.85011
N	−1.099	−20.44242
Aldehyde H	0.053	−0.90935
Amine H	0.423	−0.92662
Amine H	0.410	−1.05355
Total	0.000	−66.80346
Formic acid		
Carbonyl O	−1.059	−32.55783
Alcohol O	−1.047	−32.56055
Carbonyl C	1.408	−10.96165
Aldehyde H	0.109	−0.85759
Alcohol H	0.589	−0.82373
Total	0.000	−77.76134
Nitric Oxide		
O	−0.329	−32.07318
N	0.329	−19.56455
Total	0.000	−51.63773
Propane		
Methylene C	0.004	−11.26301
Methyl C	−0.076	−12.01459
Methyl C	−0.076	−12.01458
Methylene H	0.012	−0.94514
Methylene H	0.013	−0.94508
Methyl H	0.024	−1.05797
Methyl H	0.023	−1.05804
Methyl H	0.019	−1.05643
Methyl H	0.019	−1.05642
Methyl H	0.020	−1.05639
Methyl H	0.019	−1.05645
Total	0.000	−43.52409

The electron densities were calculated at the optimized geometries, using full SCF and the LDA functional. Gradient vectors were calculated by a cubic interpolation scheme. The atomic energies are also reported (in Rydbergs).

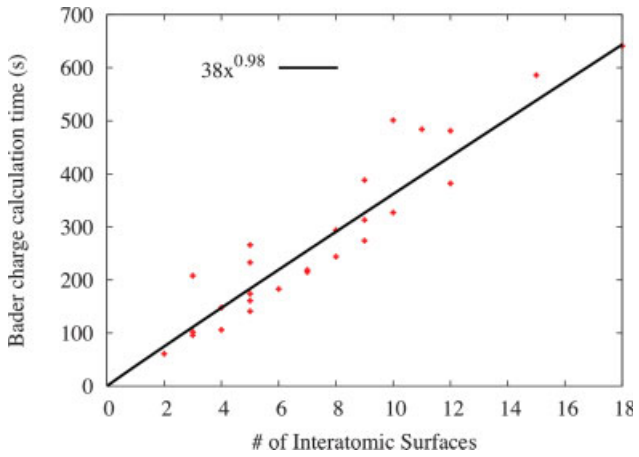

Figure 3. CPU time plotted against the total number of interatomic surfaces for each of the molecules studied in this article, (red crosses). The CPU time reported represents the time required by the Bader volume calculation and integration algorithms and does not include the CPU time required to produce the density and gradient arrays. Also shown is the best-fit power curve corresponding to the timing data, (black line). [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

where f_n is the occupancy of the n-th Kohn–Sham orbital, and e_n is its eigenvalue. $E_{\text{Ion–Ion}}$ is a sum of the ion–ion interactions and the final term is the double-counting energy, defined as

$$E_{\text{d.c.}} \equiv \int_{\rho} \left(-\frac{1}{2} V_{\text{Ha}}(\rho) - V_{\text{XC}}(\rho) + E_{\text{XC}}(\rho) \right) d\mathbf{r}. \tag{6}$$

The first term in eq. (5) is the band energy, and can be expressed in terms of the density and Hamiltonian matrix,

$$\sum_{n} f_n \varepsilon_n = \sum_{i\alpha, j\beta} \rho_{i\alpha j\beta} H_{i\alpha j\beta}. \tag{7}$$

The terms on the right side can then be partitioned among the atoms in the system, by assigning on-site and bonding contributions to each atom,

$$E_{\text{band},i} = \sum_{\alpha,\beta} \rho_{i\alpha i\beta} H_{i\alpha i\beta} + \sum_{\substack{\alpha,j\beta \\ i \neq j}} \rho_{i\alpha j\beta} H_{i\alpha j\beta}. \tag{8}$$

The ion–ion interactions can be partitioned among the atoms of the system by assigning half of each ion–ion term to each of the two ions involved. The double counting energy can then be partitioned among the atoms in the system by integrating the right side of eq. (6) over each atomic basin.

The energies reported in Tables 2 and 3 were calculated, using this energy partitioning scheme.

Scaling

The total amount of CPU time required by the new algorithm to perform the Bader analysis is plotted against the total number of interatomic surfaces present in all of the molecules studied (Fig. 3). The calculations were performed on a 2.5 GHz PowerPC G5 processor, using a mesh resolution of 0.2 Bohr radii. The CPU time required to calculate the density and gradient arrays were not included in this data. These timing values were fit to a power curve, using a least-squares fitting procedure. The best-fit equation is

$$t = 38\, x^{0.98}. \tag{9}$$

Hence, the algorithm scales linearly with respect to the total number of interatomic surfaces in the system, and requires approximately 38 s of CPU time per interatomic surface on the computer system used. Due to the fact that the algorithm scales linearly with respect to the number of interatomic surfaces, for larger systems it can also be expected to scale linearly with respect to the total number of atoms in the system. This does not include the time required to perform DFT simulations.

Summary

An improved version of the grid-based algorithm for Bader atomic property integration has been written. This algorithm is relatively simple to implement, and robust, as well as insensitive to the topology of the electronic charge density. This algorithm can also be applied to a mesh of precalculated gradient vectors, using a suitable interpolation scheme. A series of molecular calculations was presented, comparing the performance of the algorithm, using both direct gradient vector evaluation, and a cubic interpolation scheme. These calculations indicated excellent agreement between the interpolated and directly-evaluated gradient vector methods, validating the interpolation scheme. Compared with the original grid-based algorithm, the reported algorithm was also shown to be relatively insensitive to the rotation of the coordinate system of the mesh.

It was shown that a cubic gradient interpolation scheme yields nearly the same results as a direct gradient evaluation scheme, but requires only approximately 20% of the CPU time. It was also demonstrated that the algorithm scales linearly with respect to the total number of interatomic surfaces in the system. This implies that, for larger systems, the algorithm scales linearly with respect to the total number of atoms.

References

1. Bader, R. Atoms in Molecules: A Quantum Theory; Oxford University Press: New York, 1990.
2. Bader, R. F. W. Phys Rev B 1994, 49, 13348.
3. Wiberg, K. B.; Rablen, P. R. J Comput Chem 1993, 14, 1504.
4. Angyan, J. G.; Jansen, G.; Loos, M.; Hattig, C.; Hess, B. A. Chem Phys Lett 1994, 219, 267.

5. Biegler-Konig, F. W.; Nguyen-Dang, T. T.; Tal, Y.; Bader, R. F. W.; Duke, A. J. J Phys Chem 1981, 14, 2739.
6. Biegler-Konig, F. W.; Bader, R. F. W.; Tang, T. H. J Comput Chem 1982, 3, 317.
7. Popelier, P. L. A. Theor Chim Acta 1994, 87, 465.
8. Cioslowski, J.; Stefanov, B. B. Mol Phys 1995, 84, 707.
9. Popelier, P. L. A. Mol Phys 1996, 87, 1169.
10. Popelier, P. L. A. Comput Phys Comm 1998, 108, 180.
11. Malcolmn, N. O. J.; Popelier, P. L. A. J Comput Chem 2003, 24, 1276.
12. Henkelman, G.; Arnalsson, A.; Jonsson, H. Comput Mater Sci 2006, 36, 354.
13. Press, W. H.; Teukolsky, S. A.; Vetterling, W. T.; Flannery, B. P. Numerical Recipes in Fortran, 2nd ed.; Cambridge University Press, New York, 1992; p 118.
14. Kenny, S. D.; Horsfield, A. P.; Fujitani, H. Phys Rev B 2000, 62, 4899.
15. Hartwigsen, C.; Goedecker, S.; Hutter, J. Phys Rev B 1998, 58, 3641.
16. Goedecker, S.; Teter, M.; Hutter, J. Phys Rev B 1996, 54, 1703.