General Atomic and Molecular Electronic Structure System

Michael W. Schmidt,[1] Kim K. Baldridge,[1*] Jerry A. Boatz,[1†] Steven T. Elbert,[1] Mark S. Gordon,[1‡] Jan H. Jensen,[1] Shiro Koseki,[1§] Nikita Matsunaga,[1] Kiet A. Nguyen,[1] Shujun Su,[1] Theresa L. Windus,[1] Michel Dupuis,[2] and John A. Montgomery, Jr.[3]

[1]*Department of Chemistry, Iowa State University, Ames, Iowa 50011-0311,* [2]*IBM Corporation, Dept. MLMA/428, Neighborhood Road, Kingston, New York 12401, and* [3]*United Technologies Research Center, East Hartford, Connecticut 06108*

Received 15 December 1992; accepted 28 May 1993

A description of the *ab initio* quantum chemistry package GAMESS is presented. Chemical systems containing atoms through radon can be treated with wave functions ranging from the simplest closed-shell case up to a general MCSCF case, permitting calculations at the necessary level of sophistication. Emphasis is given to novel features of the program. The parallelization strategy used in the RHF, ROHF, UHF, and GVB sections of the program is described, and detailed speedup results are given. Parallel calculations can be run on ordinary workstations as well as dedicated parallel machines. © 1993 by John Wiley & Sons, Inc.

1. INTRODUCTION

The General Atomic and Molecular Electronic Structure System (GAMESS)[1] was formed from HONDO5 and other programs at the Department of Energy's National Resource for Computations in Chemistry in the late 1970s. The authors made the code available for use at other sites, and consequently this program has been in use in our group for the last decade. Many new capabilities and algorithmic improvements have been implemented during that time, including projects for conversion of the program to both vector and parallel computers. Several hundred copies of our version of GAMESS are now in use around the world. It seems appropriate to write a summary of the program's capabilities as of late 1992, especially in view of our initial positive experience with parallel computing.

GAMESS is certainly not the only *ab initio* program package, and therefore many of its functions are duplicated in other generally available packages. These include ACES II,[2] CADPAC,[3] COLUMBUS,[4] DISCO,[5] GAMESS-UK,[6] Gaussian 92,[7] GRADSCF,[8] HONDO8,[9] MELDF,[10] MESA,[11] MESSKIT,[12] MOLCAS,[13] MOLECULE-SWEDEN,[14] MOLPRO,[15] SPARTAN,[16] TX90,[17] and TURBOMOLE.[18] The existence of so many program systems is a measure of the vitality of modern *ab initio* quantum chemistry. Like each of these packages, GAMESS offers its own unique features.

The article is organized as follows: A description of the possible wave functions that GAMESS can calculate is followed by discussion of the types of runs that can be performed with each. Various methods of wave function analysis, from localized orbitals to graphics to properties, are decribed. Encouraging results from our efforts to run SCF-level calculations in parallel on systems ranging from ordinary workstations to the Touchstone Delta system at CalTech are presented.

2. OVERVIEW OF THE PROGRAM

The next two sections will describe the wave function types and run types that GAMESS can generate. GAMESS is primarily an *ab initio* program, but now includes semiempirical wave functions due to the inclusion of selected portions of MOPAC 6.[19] The *ab initio* calulations can treat all electrons explicitly or omit core orbitals through use of effective core potentials (ECPs).[20]

Specification of the molecular geometry can be made by Cartesian coordinates or by Z-matrix coordinates in two popular formats.[21] Molecules may contain any atoms through $Z = 103$. There are built in all electron or ECP basis sets for the elements H–Rn, except for the lanthanides. Of course, user input for any desired Gaussian basis set can be provided.

A user guide consisting of six chapters (about 120 pp.) is provided in machine-readable form with the source code. This manual provides directions on program installation, as well as tips on efficient use and

[*] Current address: San Diego Supercomputer Center, General Atomics, P.O. Box 85608, San Diego, CA 92138-5608.

[†] OLAC PL/RKFE, 9 Antares Road, Building 8451, Edwards AFB, CA 93523-5000.

[‡] Author to whom all correspondence should be addressed.

[§] Department of Education, Mie University, 1515 Kamihama-cho, Tsu 514, Japan.

CCC 0192-8651/93/111347-17

all possible input values. Input to GAMESS is gathered into namelist groups that contain closely related information, and values within groups are free format and keyword oriented.

Most input errors will be detected by the program so that a suitable error message may be printed. An important usability feature is the check job, which will step through the entire run. All time-consuming steps such as two-electron integral evaluation are skipped over during the check, but all input lines are read and checked for consistency. In addition, all memory needed for the real run is allocated. However, because the two-electron integrals are not evaluated the program is unable to be precise about how much disk storage for nonzero integrals will be needed.

Memory allocations are made dynamically as a GAMESS job executes, from a fixed-length pool of memory. On most machines, the size of this pool is determined by the user at job start. Larger calculations will switch to out-of-core methods when there is insufficient internal memory, and these methods will greatly outperform the page fault algorithms built into the operating system. GAMESS is designed to work well with 2,000,000 words (16 Mb) of memory over and above the memory required by the operating system and other processes running on the system. This is a modest requirement for modern workstations.

An important design goal in GAMESS is portability. There is only one source code version of GAMESS, for conventional scalar, vector, or parallel systems, or for 32- or 64-bit systems, from modern workstations such as the IBM RS/6000 or the DEC AXP, up to vector supercomputers such as those marketed by Cray or IBM. A typical workstation configuration would consist of 32 Mb of memory and 1 Gb of work disk (over and above that required for operating system and permanent user data). Computer systems with faster CPUs or larger disk and memory than these values naturally permit larger calculations. The hardware requirements for parallel computations will be given below.

3. WAVE FUNCTIONS

For the purposes of the present discussion, we define Hartree–Fock wave functions as those that are easily cast in terms of Fock operators to be calculated in the AO basis. It should be clear that some cases such as TCSCF share many common considerations with MCSCF from the user's point of view but from a computational point of view have more in common with these AO integral-driven methods.

There is a highly developed literature for quantum chemical wave function types, as well as many standard techniques for forcing convergence, etc. Here, we cite only key articles for standard methods implemented in GAMESS, with the discussion limited to methods or techniques unique to GAMESS, or where the original literature is lacking in detail.

3.1. Hartree–Fock

These wave functions are closed-shell (RHF), high- or low-spin-coupled restricted open-shell SCF (ROHF), spin-unrestricted SCF (UHF), and generalized valence bond (GVB). General topics such as direct SCF, convergence techniques such as DIIS, and a few specialized orbital options are discussed after the various wave functions.

RHF[22] remains the most commonly calculated wave function. Thus, four decades after Roothaan's work new techniques for RHF calculations continue to be proposed, particularly Pulay's direct inversion in the iterative subspace (DIIS) interpolation[23] and Almlof's direct SCF.[24] Both of these methods are implemented in GAMESS, which also provides a variant of the direct energy minimization (DEM) schemes.[25] The DEM idea is used together with DIIS interpolation,[26] but this option is seldom used as other convergence techniques normally force convergence.

For systems with unpaired electrons, GAMESS implements several open-shell SCF methods. The single determinant UHF[27] wave function is implemented, and a high-spin ROHF program is built from the UHF code. For ROHF, the SCF variational conditions for doubly occupied (c), singly occupied (o), and virtual (v) orbitals may be written[28e]

$$\langle v_i|F_c|c_j\rangle = \langle v_i|(F_\alpha + F_\beta)/2|c_j\rangle = 0$$

$$\langle v_i|F_o|o_j\rangle = \langle v_i|F_\alpha|o_j\rangle = 0$$

$$\langle o_i|2F_c - F_o|c_j\rangle = \langle o_i|F_\beta|c_j\rangle = 0$$

where $F_\alpha = F_o$ and $F_\beta = 2F_c - F_o$ are the usual alpha and beta Fock matrices. After transforming the separately calculated alpha and beta Fock operators to the current MO basis, a single Fock operator in the MO basis is constructed as follows:

$$F = \begin{array}{c|c|c|l} F_2 & F_\beta & (F_\alpha + F_\beta)/2 & \text{closed} \\ \hline F_\beta & F_1 & F_\alpha & \text{open} \\ \hline (F_\alpha + F_\beta)/2 & F_\alpha & F_0 & \text{virtual} \\ \text{closed} & \text{open} & \text{virtual} & \end{array}$$

Diagonalization of this single Fock operator necessarily keeps the alpha and beta orbitals restricted to equality.

As indicated above, the off-diagonal blocks are determined by the variational conditions, but the diagonal blocks are not specified. Their choice, however, does determine the convergence properties of the scheme and the numerical values of the individual orbitals and their eigenvalues (but not the total energy or density). As a result of this flexibility, a number of ROHF schemes have appeared in the literature with different choices of the diagonal blocks. Although many of these methods involve the construction and diagonalization of several Fock matrices during the iterative procedure, at convergence their orbital coefficients and energies may all be obtained from a single Fock matrix with suitable choices of F_2, F_1, and F_0. In our implementation, the diagonal blocks are

$$F_2 = A_{cc}F_\alpha + B_{cc}F_\beta$$

$$F_1 = A_{oo}F_\alpha + B_{oo}F_\beta$$

$$F_0 = A_{vv}F_\alpha + B_{vv}F_\beta$$

A number of choices for the canonicalization constants A, B are given in the GAMESS user guide, corresponding to various open-shell SCF procedures.[28] When DIIS is used, we find the canonicalization constants corresponding to Roothaan's single matrix approach[28a] converge well and that the application of damping factors[28e] to off-diagonal blocks serves only to hinder convergence.

Low-spin-coupled open-shell SCF calculations are accomplished in GAMESS by means of the GVB program. This section contains a general SCF capability, based on the energy formula[29]

$$E = 2\sum_i f_i h_{ii} + \sum_i \sum_j (\alpha_{ij}J_{ij} + \beta_{ij}K_{ij})$$

This formula is sufficiently general to contain many exotic SCF cases, with several open shells that may be degenerate. The literature contains many examples of the coupling coefficients f_i, α_{ij}, and β_{ij}.[30] Values for these constants are built into GAMESS for the most common cases and can be input by the user for other cases. DIIS is not implemented for low-spin open-shell SCF, so convergence must be obtained by standard procedures.

True GVB calculations[29] can, of course, be performed by the GVB code. Up to 10 pairs of electrons can be correlated by use of separated pair "geminals"[31]:

$$\Omega_i(1, 2) = C_{i1}u_i^2 + C_{i2}v_i^2, \text{ with } C_{i1}^2 + C_{i2}^2 = 1$$

with strong orthogonality enforced on all bonding and antibonding oritals u_i, v_i. Note that each geminal is written in natural form to eliminate the singly occupied configuration

$$u_i^1\ v_i^1$$

The GVB code has the option to include unpaired electrons along with these pair functions. After convergence is obtained, the nonorthogonal valence bond orbitals are generated from the natural orbitals and printed.

Our most common use, by far, of the GVB code is the one-pair case, for this is equivalent to an MCSCF wave function with two active electrons in two orbitals. This three-configuration MCSCF function can be reduced, rigorously, to two configurations without change in the total wave function by writing it in natural form. This two-configuration SCF (TCSCF) is actually of vital importance in chemistry as it is the simplest qualitatively correct treatment of many singlet biradicals. Because of its great importance, we have singled out the TCSCF wave function for special treatment, including implementation of DIIS and analytic Hessian calculation. We frequently use the TCSCF option within the GVB code instead of the general MCSCF program because its formulation in the AO basis typically makes it much faster to generate in this way. Convergence difficulties in some cases necessitate the use of the MCSCF package to obtain TCSCF wave functions.

All of the SCF wave functions RHF, UHF, ROHF, and GVB can be calculated by means of direct SCF.[24] Our implementation does not store any integrals in memory to avoid their recomputation, but does use Schwarz screening.[32] In the case of RHF, UHF, or ROHF, we use the difference density to minimize the recomputation of integrals.[24c] Normally, we find that direct SCF computation is about two or three times slower than the conventional technique of storing integrals on disk. Direct SCF is best reserved for cases where the I/O capability of the computer is extraordinarily poor or sufficient disk storage is unavailable.

The simpler cases of RHF, UHF, and ROHF almost always show good convergence starting from extended Hückel orbitals obtained as described in Section 6.2. GVB convergence almost always requires the input of orbitals from some simpler wave function, such as RHF. If poor convergence is observed, it is usually far more fruitful to change starting orbitals than fuss with the convergence options described in the next paragraphs.

DIIS is implemented for RHF, UHF, ROHF, and TCSCF cases. The RHF routine works exactly as described by Pulay.[23b] As is alluded to briefly in that article, the convergence criterion for RHF is that the Fock elements in the MO basis that connect the occupied and virtual orbitals must vanish. If they do not, their asymmetry (orbital gradient) may be regarded as an "error," which the DIIS interpolation will reduce to zero. It can be shown that the MO error matrix $\varepsilon_{ia} = F_{ia} - F_{ai}$ (with diagonal blocks zero) is ε = FDS-SDF after transformation to the AO basis. This is conveniently evaluated with just two matrix multiplies. The AO error matrix must be transformed to any fixed orthonormal basis, for which we use Q rather than Pulay's suggestion of $S^{-1/2}$. The Q matrix is already generated in GAMESS as a transformation to a symmetry-adapted orthonormal basis[33] and satisfies $Q^tSQ = I$. This transformation represents two additional matrix multiplies. If DIIS is in operation, the SCF process therefore requires a total of seven matrix multiplies (three to solve FC = SCE) in addition to one diagonalization.

DIIS will sometimes converge to orbitals that satisfy the SCF equations but are not the global minimum (particularly in the open-shell cases UHF and ROHF). This can be avoided by initiating DIIS only after the largest component of the error matrix falls below a user-selectable threshold.

DIIS for UHF is implemented by calculating the error matrix for each spin separately:

$$\varepsilon_\sigma = F_\sigma D_\sigma S - S D_\sigma F_\sigma$$

where $\sigma = \alpha$ or β. The two error matrices are added, $\varepsilon = \varepsilon_\alpha + \varepsilon_\beta$, before applying the same DIIS routine that is used for the RHF case. For ROHF, the DIIS error matrix is calculated using the combined Fock operator described above, back transformed to the AO basis, and the total density $D = D_\alpha + D_\beta$, by $\varepsilon = \mathrm{FDS} - \mathrm{SDF}$. Because the UHF method is forced to use one set of interpolation coefficients for both α and β Fock matrices, while the ROHF method interpolates only the combined F, ROHF will converge substantially more quickly than UHF. It is fairly typical to see the ROHF wave function converge in 15 cycles whereas the same system run as UHF might take 25 iterations. DIIS for the TCSCF case is implemented as described by Pulay.[23c]

Prior to the onset of DIIS, GAMESS uses standard convergence accelerators, such as extrapolation[34] or damping.[28h] Both methods are applied directly to the Fock operators rather than the density matrices. Extrapolation is normally the most useful method (apart from DIIS) in forcing convergence. Typically, 0–2 iterations are required before the onset of DIIS, after which extrapolation and damping are turned off. Level shifting[35] is available, but is not ordinarily necessary to achieve convergence. A special option restricts orbital rotations by maximizing the overlap of the new orbitals with those of the previous iterations to prevent sudden orbital interchanges. This is the only convergence method that remains in effect after DIIS begins, and its use without caution can often lead to convergence to false SCF solutions.

Modified virtual orbitals (MVOs) can be obtained for RHF, ROHF, and GVB after convergence of the wave function.[36] MVOs are far more appropriate for use as starting orbitals in MCSCF (or GVB) calculations because these cationic virtual orbitals are normally valence antibonding MOs whereas the lowest virtuals of the neutral are normally more diffuse in nature. MVOs are constructed by forming a closed-shell Fock operator for a highly positive cation, using the density of the converged neutral orbitals after omitting about six electrons. Diagonalization of this operator within the virtual orbital space of the original neutral system yields the MVOs. The cost of MVOs is one RHF iteration, far less than the MCSCF iterations their use may save.

The natural orbitals for the UHF wave function can be obtained from the total UHF density, $D = D_\alpha + D_\beta$. These UHF-NOs have been proposed as a means of determining whether and what type of MCSCF calculation might be necessary.[37]

3.2. Post Hartree–Fock

Second-order perturbation theory is now known to be an affordable yet reasonably accurate approach to electron correlation.[38] A second-order Møller–Plesset program using symmetry during the integral transformation[39] is included in GAMESS. Originally limited to an RHF zeroth-order wave function, the MP2 code has now been extended to include the UHF case. In turn, we have used the UHF code to implement a second-order perturbation calculation starting from the ROHF wave function.[40] The code in GAMESS is fast due to its use of point group symmetry but is extravagant in its memory and disk usage. Caution should be taken to apply the MP2 methodology only when the single configuration reference is the dominant part of the wave function.

A configuration interaction (CI) program based on the graphical unitary group approach (GUGA) is included in GAMESS.[41] This is capable of handling up to 10 electronic states and is limited in size by available disk storage to store GUGA loops. The CI program permits use of any desired excitation level (singles, singles + doubles, . . .), and includes the option of generating a first- or second-order CI function. The latter is particularly useful with orbitals determined by an earlier MCSCF run. Full Abelian subgroup symmetry is exploited in generation of the configuration state functions (CSFs).

A second-order multiconfiguration SCF (MCSCF) program due to Dupuis[42] is the featured post Hartree–Fock method. This is based on the GUGA CI package, with a second-order Newton–Raphson orbital optimization step. The orbital optimization, of course, omits redundant rotations (core–core, and additionally active–active in the case of complete active spaces), and exploits Abelian subgroup symmetry. Convergence is normally good for the lowest state of each symmetry, provided the user has taken care to provide reasonable starting orbitals. State averaging makes it possible to obtain orbitals at least partially optimized for an excited state in cases where root flipping or poor convergence prevents direct excited-state optimization. An option to prevent optimization of core orbitals is provided to facilitate transition density calculation. Although we most commonly use complete active space wave functions of the full optimized reaction space (FORS) type,[43] it is possible to optimize lower excitation level MCSCF functions.

The purpose of MCSCF is, of course, to allow almost any sort of chemical reaction to be treated. Reactions involving bond breaking and comparison of states of different multiplicities are possible by appropriate choice of active space. While it is a powerful method, MCSCF has a number of pitfalls ranging from the design of the active space to selection of initial orbitals. Some preliminary reading before use of this section of GAMESS is advisable; see, for instance, ref. 43–45.

4. BASIC RUN TYPES

The three fundamental run types are calculation of the energy, gradient, or Hessian. Here, calculation of the energy Hessian (second derivative with respect to nuclear coordinates), of course, implies evaluation of the energy itself, along with its gradient (first derivative).

The energy can be calculated for any of the wave function types mentioned in the previous section. Analytic gradients are available for all cases where the energy can be calculated, apart from CI and MP2. Hessian calculations can be performed by finite differencing of any analytic gradient.[46] In addition, analytic calculation of the Hessian is implemented for RHF,[47] high- or low-spin ROHF,[48] and two-configuration SCF (TCSCF).[49] A summary of the program capabilities is provided in Table I.

The energy Hessian has many uses, including vibrational analysis of the molecule.[50] One can easily change isotopic masses and an analysis of the normal modes of vibration in terms of internal coordinats can be performed.[51] If the Hessian is obtained numerically, the IR intensities will be computed and printed. The Hessian is also useful in speeding the course of geometry optimizations or transition-state searches.

Contamination of the Hessian by translational and rotational (TR) degrees of freedom (particularly at nonstationary points) can be eliminated in two ways. Purification is accomplished by transforming the Hessian from Cartesian coordinates to internal coordinates and then back transforming to Cartesians once again. This effectively eliminates any TR contributions, while leaving the vibrational normal modes almost unaffected. Projection[52] of the Hessian also eliminates the TR contaminants, but also forces one "mode" to lie parallel to the gradient (if it is nonzero) while zeroing its frequency. Projection thus changes the remaining modes rather more than does "purification." Projected Hessians at various points along an intrinsic reaction coordinate (IRC) are useful quantities during the calculation of a reaction path Hamiltonian[52] in preparation for dynamics calculations.[53]

Table I. Summary of GAMESS capabilities.

Ψ	Energy	Gradient	Hessian	MP2	CI
RHF	CDPM	AP	NAP	y	y
ROHF	CDPM	AP	NAP	y	y
UHF	CDPM	AP	N P	y	n
GVB	CDP	AP	NAP	n	y
MCSCF	C	A	N	n	y

C, conventional; D, direct; P, parallel; M, MOPAC; A, analytic; N, numeric; y/n, yes/no.

5. OTHER RUN TYPES

Given the ability to calculate the energy, gradient, and Hessian for the desired quality of wave function, GAMESS can then perform a variety of chemically useful jobs. These are gradient- and non-gradient-based geometry optimizations and transition-state searches, reaction path following, and the calculation of polarizabilities, transition moments, or spin–orbit coupling.

When analytic gradients are available, GAMESS can efficiently locate molecular structures [relative minima on the potential energy surface (PES)] or transition states. Both geometry searches use quasi-Newton optimizations and can be performed in Cartesian or internal coordinates. A more reasonable guess for the initial Hessian can be provided by Badger's rules if internals are in use[54] (in Cartesian space, the guess Hessian is $\frac{1}{3}\mathbf{I}$). A calculated Hessian (possibly obtained at the semiempirical level) may be useful in speeding the search for minima, and is required for saddle point searches. There are two quasi-Newton geometry search procedures implemented in GAMESS, due to Schlegel[55] and Baker.[56] For minima, these differ primarily in the Hessian update formula used, and we find they function equally well. For saddle points, the Baker method seems to work best, due to its handling of the negative curvature search as a separate problem, and therefore this is the default geometry search procedure.

Often, a change in the choice of internals can lead to dramatic improvement in the number of search steps. It is useful to note that the internal coordinates used during the geometry optimization can be different from those used to define the initial molecular geometry, for example, permitting ring closure. In addition to the usual distances, bends, and torsions, the internal coordinate options include four novel types: linear bends, bond–plane bends, five atom torsions, and "ghost" torsions.

Nongradient geometry optimizations (but not transition-state searches) can also be performed using a Fletcher–Powell-based minimization scheme.[57] Ordinarily, this is used only for MP2 or CI optimizations for which gradients are not available. The same routine can optimize the exponents of Gaussians such as diffuse or polarization functions.

The IRC is a trajectory calculated in mass-weighted Cartesian coordinates, connecting reactants through the transition state to products.[58] This run type has several options for the IRC integration, including corrected Euler,[59] fourth-order Runge–Kutta,[60] and Schlegel's second-order[61] steps. IRC calculations have several uses, including identification of the products and reactants associated with a particular transition state, which is not always obvious from examination of the normal mode associated with the imaginary frequency. A particularly important use of IRCs is "direct dynamics," in which reaction rates are obtained directly from PES information (including Hessians) calculated along an IRC.[53]

Polarizabilities and hyperpolarizabilities can be calculated by a numerical method involving calculations in the presence of various applied electric fields.[62] Polarizabilities use computed energy differences as well as dipole differences, so any of the wave functions in GAMESS can be used, including CI excited states. Comparison of the two values gives some idea of the accuracy with which they are calculated.

The radiative transition moment and Einstein coefficients connecting two states of the same multiplicity can be calculated. The method used to calculate the transition density[63] permits separate optimization of the orbitals by MCSCF calculation on both states provided that the doubly occupied orbitals are kept identical in the two states. The transition moment is calculated from both dipole length and dipole velocity forms, which would give identical numerical results if exact wave functions were available. Differences in the two computed values therefore give some idea of the quality of the wave functions employed.

An approximate one-electron spin–orbit coupling operator using an effective nuclear charge[64] is available. As for transition moments, the user can choose to optimize the active orbitals for the two different spin states separately, keeping only the core orbitals common. Preliminary applications indicate that the effective operator is reasonably accurate compared to full spin–orbit calculations, and we have presented values for the effective charges to be used for elements Li–Cl.[64]

6. USEFUL OPTIONS

6.1. Localized Orbitals

Three different schemes for localizing orbitals are built into GAMESS. Two of these involve the use of one-electron integrals: the dipole-based Boys localization[65] and the overlap-based population localization.[66] The third is Ruedenberg's energy localization,[67] requiring the electron repulsion integrals. Recent work has shown that this method is actually not as expensive as commonly thought.[68] The three methods do not always yield the same results, even qualitatively,[69] so it may be useful to compare each type.

By default, GAMESS localizes orbitals for RHF, UHF, ROHF, and MCSCF to achieve the maximum localization possible that still leaves the wave function invariant. For example, an MCSCF localization will localize doubly occupied valence orbitals and the active orbitals but will not mix these two spaces. Chemical core orbitals are normally excluded from the localization, but this may be overridden by the user.

An additional option to localize under the constraint that orbitals of different symmetries do not mix is also provided. This is intended to allow these orbitals to be used as starting orbitals for MCSCF. These partially localized orbitals allow one to easily identify the bonds that should be correlated in the active space while preserving the orbital symmetry crucial to speed MCSCF runs.

6.2. Hückel Guess

It is clearly desirable to begin SCF calculations with orbitals that converge within a small number of iterations, particularly if the SCF is run in direct mode. Perhaps less obviously, it is also important that the electron configuration be correctly filled from the beginning. If orbitals of the wrong symmetry are occupied at the beginning, at best the convergence will be poor and at worst one may arrive at an incorrect wave function if the electrons do not flip into the correct orbitals during the iterations.

We have found that orbitals generated by a Hückel guess almost always satisfy the latter criteria and also converge satisfactorily. In the year that we have been using the procedure outlined below, we find the majority of RHF calculations converge in 11–15 iterations, 15–20 iterations for ROHF, and about 25 iterations for UHF. The Hückel guess seldom gives an incorrect HOMO/LUMO order, although it (or indeed any other method) should not be expected to be foolproof in this regard.

Our extended Hückel orbitals are generated by diagonalization of the Wolfsberg–Helmholz operator[70]

$$H_{ij} = K \cdot S_{ij} \cdot (H_{ii} + H_{jj})/2$$

where S_{ij} is the overlap integral, H_{ii} and H_{jj} are energy parameters for a given AO, and K is empirically found to be 1.75.[71] This formula (or its modern generalization[72]) has been employed extensively in qualitative MO theory to predict approximate orbital shapes together with their energy orderings.[73] This is exactly what we desire for starting orbitals.

As written, the extended Hückel formula applies to valence minimal basis sets. In *ab initio* calculations, we normally have inner shell orbitals present, and of course we often use split valence or polarized basis sets. The second problem can be eliminated by calculating the Hückel operator in a minimal basis set and then projecting the results onto the current basis. The presence of inner shells is actually only a minor complication.

We use the Huzinaga group's MINI basis[74] (each shell containing three contracted Gaussians) to set up the extended Hückel operator. This basis exists for all atoms from hydrogen to radon and is of a uniform quality for all these atoms. We use large STO basis SCF or numerical SCF eigenvalues[75] for the atomic energy parameters H_{ii}. Because these energies can be large for inner shell orbitals, all core–core and core–valence H_{ij} elements are reduced by an additional factor of 0.05. This decreases the mixing between valence and core orbitals to an appropriately small amount.

Projection of the resulting Hückel orbitals from the MINI basis onto the actual basis is done by corresponding orbitals.[76] This requires the overlap integrals between the two basis sets. The projection is made onto symmetry-adapted linear combinations of the current AO basis,[33] in two or three steps. First, all doubly occupied Hückel orbitals are projected. Then, any partially occupied orbitals are projected onto the orthogonal complement space of the first projection. Finally, we project the first five unoccupied orbitals of the Hückel calculation in the rare circumstances that the user wishes to reorder the orbitals to occupy some of these before the SCF is initiated. If a single projection step were attempted, diagonalization of the symmetric quantity D^tD that arises during the projection[76] would scramble these spaces. The remaining virtuals are generated by a Schmidt orthogonalization step.

Each projection is accomplished using standard formulae, which are not repeated here.[76] The entire Hückel process is expensive, requiring several matrix diagonalizations as well as various matrix multiplies. Although considerably more time consuming than the common alternative of diagonalizing the bare nucleus Hamiltonian h (which is also implemented in GAMESS), it is far faster than a single SCF cycle. Besides saving many SCF cycles, we find the Hückel guess usually generates the correct electron configuration across the periodic table, from main group to transition metal compounds.

6.3. Semiempirical Wave Functions

Semiempirical wave functions can be of RHF and UHF type, as in MOPAC 6, and, uniquely to GAMESS, can be of the spin-restricted ROHF type using the mechanism described in Section 3.1. The MNDO, AM1, and PM3 parameter sets are all available.[19]

Little of MOPAC apart from its integral evaluation, Fock formation, and gradient routines was incorporated into GAMESS. All SCF drivers, geometry steppers, and so on remain those already present in GAMESS. This means all run types using the gradient are enabled, from geometry searches to reaction path following. The various numerical convergence criteria are the same as those used for *ab initio* runs and consequently are generally tighter than those used in MOPAC. Thus, you will not be able to exactly duplicate MOPAC 6 numerical results, and not all MOPAC options are available. However, a number of the usual GAMESS features work automatically, including Hückel guess starting orbitals, symmetry labels for semiempirical orbitals, orbital plotting, and displacement of only the symmetry-unique atoms during numerical Hessians.

Those MOPAC routines that were incorporated were modified to use the dynamic memory management of GAMESS so that calculations on large molecules use far less memory than in MOPAC itself. This has enabled us to run a molecule such as taxol ($C_{47}H_{51}NO_{14}$, 113 atoms, 299 AOs) with a modest memory demand of 900,000 words. The AM1 energy and gradient of this molecule can be computed in just 4 min on an IBM RS/6000 Model 350.

6.4. Molecular Properties

GAMESS calculates a number of standard properties for any of the wave functions (except MP2). These include the standard multipole moments: dipole, quadrupole, and octupole. Total electron density or spin density at the nuclei can be found. The necessary energy contributions (T, V_{ne}, V_{ee}) needed to compute the virial ratio are computed.

Atomic charges may be computed by Mulliken's method[77] of assigning populations to AOs. Additionally, the "Lowdin population" is obtained by use of Mulliken's formula after transformation of the wave function to a symmetrically orthogonal basis.[78] Of course, these atomic charges are basis set dependent, so GAMESS can also compute the molecular electrostatic potential (MEP). Unlike the two atomic charges, the MEP is in principal physically observable. One application of the MEP is to predict reactivity at different sites in the molecule. As mentioned below, the MEP may be displayed in the form of a picture. The first and second derivatives of the potential, which are the electric field and the electric field gradient, may also be computed.

The distributed multipole analysis (DMA) of Stone is included.[79] The DMA is a decomposition of the electrostatic potential in terms of multipoles located at various points in the molecule.

6.5. Graphics

Four graphics programs are included in GAMESS, to be used to display results of calculations. These are all 2-D graphics programs, with modest hardware requirements. Interactive use requires the X Window System, which is included on nearly every workstation, and a 2-D graphics screen. The programs can also produce PostScript files for hardcopy.

These programs and their features are: MOLPLT draws ball and stick figures and optionally includes a normal mode of vibration. PLTORB draws contour plots of molecular orbitals, which can be expressed in either a Gaussian or Slater orbital basis. These orbital plots can in turn be used by DENDIF to draw total electron densities or density differences. Finally, MEPMAP displays the electrostatic potential around a molecule.

While not a graphics program itself, the auxiliary program MBLDR (model builder)[21] will interactively convert a *Z*-matrix file into Cartesian coordinates. After previewing the Cartesian coordinates with MOLPLT, the *Z*-matrix may be inserted directly into a GAMESS input file in confidence that the molecule is the desired one.

6.6. Program Interfaces

GAMESS can communicate with several other packages to provide for the computation of additional molecular properties not included in GAMESS or the graphics programs just mentioned.

The Atoms In Molecules package (AIMPAC) of Prof. Richard Bader's group can be used with any of the wave functions (except MP2). AIMPAC analyzes the shape of the total electron density and computes a well-defined atomic charge.[80] The interface consists of ASCII data punched by GAMESS upon the user's request. If desired, the wave function may be computed on a large computer and the analysis (which is largely graphical) performed on a desktop workstation. For information about the availability of AIMPAC, contact Prof. Bader.[81]

The random-phase approximation package (RPAC) of the late Prof. Tom Boumann and Prof. Aage Hansen can be used with RHF wave functions only. RPAC computes nuclear magnetic resonance (NMR) shieldings and the vertical excitation spectrum of a molecule.[82] The interface consists of passing the integral file and one other binary file to RPAC, so the analysis must be performed on the same machine used to generate the wave function. For information about the availability of RPAC, contact Prof. Hanson.[83]

The natural bond orbital package (NBO) of Prof. Frank Weinhold's group can be used with any of the wave functions except MP2. NBO analyzes the populations and hybridizations by means of localized electron pairs.[84] The interface consists of additional subroutines to be linked directly into GAMESS, after uncommenting a call within subroutine HFPROP. For information about the availability of NBO, contact Prof. Weinhold.[85]

By using the FRIEND keyword, GAMESS can generate most of the input needed by HONDO8, MELDF, GAMESS-UK, or Gaussian 92. Some user massaging of these skeleton input files may be needed, however. Similarly, GAMESS output of energies, gradients, and Hessians along an IRC can be converted (by a utility program named IRCED) to input data for the variational transition-state theory program POLYRATE.[86]

6.7. Use of Symmetry

GAMESS makes use of molecular symmetry throughout most stages of a calculation. This is largely automatic, with the user responsible primarily for naming the point group in use. Some familiarity with irrep names and term symbols is required when using the more sophisticated wave functions, which partially occupy certain orbitals.

Symmetry is handled by the petite list formalism during the evaluation of repulsion integrals[87] or their first[88] or second[89] derivatives. Fock matrices are placed in block diagonal form,[33] preventing mixing of degenerate levels. The CI and MCSCF programs work in the highest available Abelian subgroup of the group requested by the user. Only symmetry-unique responses are solved for during the coupled-Hartree–Fock (CPHF) stage of an analytic Hessian calculation.

As already stated, the use of symmetry is almost entirely automatic. However, some caution on the part of the user may be required. The use of petite integral lists followed by the symmetrization of some skeleton quantity such as the gradient or the Fock matrix "is derived assuming that the electronic charge density transforms according to the completely symmetric representation of the point group."[88] This quote is placed here as a reminder that the MCSCF and GVB portions of the program can generate wave functions that are spatially degenerate. Some specific examples are ^2E or $^1\Delta$ electronic states arising from e^1 or π^2 configurations. These types of runs require use of symmetry be turned off.

7. HIGH-PERFORMANCE IMPLEMENTATION

7.1. Vector Capability

Limited vectorization of GAMESS has been achieved by writing clean enough loops for automatic compiler vectorization to occur rather than by redesigning algorithms with vector processing in mind. The use of compiler directives is minimized so that many vector machines can be used. In some cases, special replacement routines were written either to call a vectorized system library or in the few instances in which vectorized code slowed scalar execution. The compiling procedure automatically selects these vectorized routines instead of their scalar counterparts.

Because existing algorithms have been used, the results of the vectorization project were meager for integrals, SCF, or gradients, with a total job speedup of about 20% with vectorization. Certain other sections of the program will vectorize nicely, particularly the second-derivative integrals, integral transformations, and the CPHF solver. For example, speedups in the integral transformation are about 5 and in the CPHF solution about 7 compared to the scalar time on an IBM 3090VF machine. The latter result is near maximum machine performance. However, these vectorizable steps do not represent the only compute-intensive portions of these jobs, and thus an entire MCSCF or analytic Hessian job will normally run in about half of the scalar time.

7.2. Parallel Capability

Some work on parallelization of *ab initio* codes has been done in the past years. Investigators have parallelized the two-electron repulsion and gradient integrals[6,90,91] (many in conjunction with direct SCF), integral transformation,[92] MP2 energy,[93] configuration interaction energy,[94] and coupled-cluster energy.[95] While this article was being reviewed, an entire issue of *Theoretica Chimica Acta* was devoted to this subject.[96] We have drawn on some of these ideas to parallelize GAMESS. While many of the ideas may not be new, the breadth of parallelization in GAMESS is.

There are presently many different models for parallel computing, ranging from shared-memory designs to small or large distributed memories, and single or multiple instruction on multiple data (SIMD vs. MIMD). From a practical viewpoint, hardware designs with MIMD capability and large memory for each processor seem the most accessible for quantum chemistry applications. There are a number of vendors such as Intel and nCUBE marketing this type of hardware, and such systems have become common at the National Supercomputer Centers.

The pandemonium in hardware designs is reflected in (or is perhaps caused by) a similar chaos in programming models for parallel computation. For instance, many vendors have chosen to implement specialized parallel directives for use in FORTRAN DO loops while other vendors provide messaging system calls used to run entire programs in parallel.

Thus, at the present time the programmer is faced with the dilemma of perhaps having to choose a particular (and probably nonportable) vendor FORTRAN extension or waiting for the inevitable shake out in parallel hardware/software. Fortunately, there are alternatives. A number of packages exist to permit more portable parallel coding, including among others PVM, P4, PCN, Linda, or Express. We have chosen to use a similar tool kit known as TCGMSG.

TCGMSG is a message-passing library developed by the theoretical chemistry group at Argonne National Laboratory.[97] This package implements a distributed-memory MIMD programming model on hardware of this type, as well as on some shared-memory multiprocessor systems. TCGMSG also implements this model on ordinary workstations attached by means of Ethernet. TCGMSG uses the best communications available, ranging from direct use of the interconnect in parallel machines to TCP/IP sockets on Ethernet. An efficient global sum (reduction) routine is available, which is optimized for each type of communication. The TCGMSG package can be installed and used with only ordinary user-level privileges. Many common UNIX workstations are supported by TCGMSG, bringing parallel quantum chemistry within the reach of individual research groups.

TCGMSG is available by anonymous FTP. This is the only software requirement (in addition to UNIX and the usual compilers) that is needed to run GAMESS in parallel. In case the person installing GAMESS prefers to run only in sequential mode, there is a file of "do nothing" stub routines that satisfy the external references that may be linked instead of the TCGMSG library.

GAMESS will currently evaluate the energy and gradient of any of the Hartree–Fock wave functions (RHF through GVB) in parallel. Representative timing results for realistic molecules will be presented below. To parallelize these SCF wave functions, the following sections of the program were modified to run in parallel: $1e^-$ integrals, $1e^-$ ECP integrals, $2e^-$ integrals, matrix manipulations to set up the SCF equations, "semiparallelization" of the matrix diagonalization (to be explained below), $1e^-$ gradient, $1e^-$ ECP gradient, and the $2e^-$ gradient. We found it advantageous to make some of these sections that ordinarily seem trivial (on a single CPU) run in parallel. The time for these small steps can actually become significant once the longer steps run in parallel on many nodes. Note that once the energy and gradient run in parallel many other run types are effectively parallelized, for example, geometry searches or numerical Hessians.

The parallel SCF energy and gradient can be computed using conventional disk storage of integrals if the individual nodes have sufficient storage for their subset of the two-electron integrals. If the nodes are poor at I/O (e.g., the Intel iPSC/860) or if they simply do not have sufficient disk storage, then the direct SCF option may be employed.

Parallel programs are normally implemented by either a master–slave model or a peer model. In the former case, the master is the original sequential program, doing all sequential steps, but using slave programs to carry out the compute-intensive parts of the calculation. Typically, the slave codes are much simpler and therefore result in smaller programs. In the peer model, each node runs the same program and therefore the same amount of memory is required on each node.

GAMESS is implemented as a peer parallel program because this satisfies our basic philosophy that there shall be only one version of GAMESS regardless of the hardware platform. However, we found it convenient to adopt the master–slave model to handle the basic I/O. All input cards are read and output lines written by the "master" program running on node zero, which also maintains a single copy of the direct access file, containing one-electron integrals, current orbitals, and so on. The other "slave" nodes skip over all output statements and receive input lines or direct access records that have been broadcast to them by the master.

An effective strategy for parallelization of SCF wave function evaluation has been known for some time.[90] Each of the processors (p in number) executes the same program, entering the loops over sets of shells for which integrals must be computed. Each processor skips most integral blocks, taking only those that it determines have been assigned to it. A scheme must be devised so that each block of integrals is evaluated once. The calculation is completed by adding together the partial Fock matrices evaluated by each node from its partial integral list.

The simplest load balance strategy is to divide the integral list evenly among the various processors (static load balancing). In what we term LOOP balancing, each node takes regular turns, evaluating every *p*th block of integrals, while skipping past the other blocks.[90] This scheme of sending equal numbers of integral blocks to each processor works well when all processors are of equal speed. Although each block of integrals could be different, for example an [ss|ss] vs. [dd|dd] set, there are so many more integral blocks than processors that these local imbalances average out.

However, if the processors are unequal in performance or if they possess unequal background workloads the LOOP balance scheme will not extract the maximum work. A strategy we refer to as NXTVAL will work better in this circumstance. A counter (shared by all nodes) running from 0, 1, 2, ... is returned by the TCGMSG function NXTVAL. It is up to the application program to map this list of integers onto the complete list of integral blocks that must be calculated. Pseudocode is given in Figure 1 to show the differences in the two types of load balancing. Unlike LOOP, in which each processor is given an equal amount of work, NXTVAL balancing assigns the next item on the list of work to be done to each processor as it finishes its previous task. This dynamic load balancing allows the faster processors to take relatively more of the work.

NXTVAL requires some communication between nodes to distribute the list of integers, in contrast to the completely communications-free LOOP balancing scheme. When equal processors are in use, this NXTVAL overhead adds 2–5% to the run's wall clock time compared to LOOP balancing. The added communications makes it advantageous to include more blocks of integrals in the work packets. Each GAMESS processor skips blocks in the innermost of the four (petite list) shell loops if LOOP balance type[90] is in use but skips sets of blocks in the second shell loop for NXTVAL. Similar load balance strategies are employed in the two-electron gradient section and in the various types of one-electron integrals.

```
      SUBROUTINE TWOEI
C
C     ME = this processor's ID number
C     NPROC = number of processors
C
C     initialize parallel work
C
      IPCOUNT = ME-1
      NEXT = -1
      MINE = -1
C                 begin the 4 loops over shell sets [IJ|KL]
      DO 100 I =
         .
         .
         DO 200 J =
            IF (NXTVAL balancing AND PARALLEL) THEN
               MINE = MINE + 1
               IF (MINE.GT.NEXT) NEXT=NXTVAL()
               IF (NEXT.NE.MINE) GO TO 200
            END IF
            .
            .
            DO 300 K =
               .
               .
               DO 400 L =
                  IF (LOOP balancing AND PARALLEL) THEN
                     IPCOUNT = IPCOUNT + 1
                     IF (MOD(IPCOUNT,NPROC).NE.0) GO TO 400
                  END IF
                  .
C      Generate integral block [IJ|KL]
C      Write to disk or place directly in Fock matrix
                  .
 400           CONTINUE
 300        CONTINUE
 200     CONTINUE
 100  CONTINUE
      END
```

Figure 1. Pseudocode for integral parallelization.

It is possible to perform some of the work for the solution of the Hartree–Fock equations in parallel. In particular, the seven matrix multiplies that are required to calculate the DIIS error matrix and solve the closed-shell SCF equations can be run in parallel. The Fock matrix is in block diagonal form, so we use an ordinary sequential diagonalization routine, handing different blocks off to different processors. Diagonalization requires roughly the same time as one matrix multiply and is difficult to run in parallel, so this "semiparallelization" strategy was adopted. We find it best to perform the DIIS interpolation itself on a single node, broadcasting the result to the other nodes. It is the inefficiencies in the SCF solution process that cause deterioration in the efficiency for the overall SCF step. The parallelization in the repulsion integral generation and Fock assembly is actually as high as the gradient integrals.

The parallel version of GAMESS changes the bottleneck for large SCF calculations from CPU time to memory. The symmetric density and Fock matrices must be stored on each node while the partial Fock contributions are evaluated. Other matrices of order N^2 are replicated on each node during the SCF process, as we have chosen not to implement small slave programs that only carry out Fock evaluation. For RHF, the total memory demand for D, F, S, h, V, Q, and the various working arrays totals $7N^2$. If storage is tight, some of these can be overlapped to reduce total storage needs to $5N^2$. Unfortunately, it is common for MIMD computers to be installed with rather small memories on each node. Thus, for RHF wave functions the maximum permissible basis set size N will be about 185 on an 8-Mb real memory system such as the 64-node iPSC/860 installed at the San Diego Computer Center. The majority of the 8 Mb is consumed by operating system, message buffers, and GAMESS object code, leaving just 2 Mb for working storage. Problems of this size can be handled (albeit more slowly) on a conventional machine. The Touchstone Delta system installed at Cal Tech contains 512 nodes, each having 16 Mb. This permits calculations up to about 400 AOs to be performed at high speed. This amount of memory permits us to undertake projects that could not otherwise be contemplated, but 400 AOs still represents a limit imposed by memory rather than by the available machine time on equipment such as the Delta. We estimate that 5–10 workstations with a 64-Mb memory (and with virtual memory to page some of the arrays, which are not all in simultaneous use) would permit efficient parallel SCF with basis sets on the order of 1000.

Our first timing example is phosphinoaluminate ion, $(CH_3)_3AlPH_2^-$, which has C_s symmetry. The basis used is 6-31 + +G(*d*, *p*), which has a total of 169 AOs. The run generates a 550-Mb integral file (734 Mb on the Cray) when run conventionally. The timings for all steps needed to evaluate the wave function's energy and its gradient are included. The data in Table II indicate that conventional SCF utilizing local disks for integral storage outperforms direct SCF on multiple IBM RS/6000 hosts (42-MHz clock) when LOOP load balancing is used. This is due to a factor of 3 change in the SCF CPU time, 5773 s in direct mode vs. 774 + 1130 = 1904 using disk storage. This is clearly true for the CPU time alone and is even true of the wall clock time (which is elongated by I/O delays in the case of conventional SCF). Further, it is possible to equal or even outperform a single node of a Cray Y-MP with two workstations of this type. In all fairness, though, GAMESS is not fully optimized to use the Cray architecture as the integrals were not rewritten to take full advantage of the vector processor. Essentially, we are seeing the scalar time comparison. Because the Cray processor was not dedicated to the benchmark, its wall clock timing is meaningless.

Table II. RHF/6-31G + +(*d*, *p*) $(CH_3)_3AlPH_2^-$, timings in seconds.

	No. of Processors			
	1	2	4	Cray Y-MP
Conventional SCF: RS/6000 Model 350				
Job setup	0.9	0.9	1.1	0.3
$1e^-$ integrals	3.0	2.1	1.8	3.7
Initial guess	49.0	49.4	51.6	29.4
$2e^-$ integrals	733.8	369.0	186.5	482.1
RHF iterations	1129.9	593.9	336.5	248.6
Properties	5.1	6.7	8.0	1.5
$1e^-$ gradient	20.8	10.9	6.1	25.9
$2e^-$ gradient	2995.7	1502.0	755.1	1687.0
Total CPU	4938.2	2535.0	1346.8	2478.4
Total wall	7401	3996	2401	n/a
Direct SCF: RS/6000 Model 350				
Job setup	0.9	1.0	1.0	0.3
$1e^-$ integrals	3.0	2.1	1.7	3.7
Initial guess	49.0	49.5	51.2	29.5
$2e^-$ integrals + RHF	5773.4	2939.6	1521.6	4731.0
Properties	5.1	6.9	8.0	1.5
$1e^-$ gradient	20.6	10.9	6.2	25.8
$2e^-$ gradient	2994.4	1501.3	754.8	1690.6
Total CPU	8846.5	4511.4	2344.6	6482.3
Total wall	8875	4615	2855	n/a

This calculation requires 12 iterations using the disk and 11 iterations in direct mode. All timing tables in this article use LOOP balancing.

We chose the rather small aluminate example because it is about the largest calculation we are able to perform on a single node of the Touchstone Delta. Table III gives wall clock times for each step of the same calculation, as well as the speedup ratio s(p) = t(p)/t(1) and the efficiency of the parallelization, s(p)/p times 100%, where p is the number of processors.

These data indicate the value of paying attention to seemingly minor steps such as the $1e^-$ gradient. On 512 nodes, the $2e^-$ gradient can be computed more rapidly than the $1e^-$ contribution to this term on 1 node! While the efficiencies of these small steps will never be high, they do contribute to the overall job efficiencies.

The data in Table III clearly indicate that future efforts to run the Hückel initial guess in parallel would be useful. In the context of a geometry optimization, it is important to realize that the Hückel guess is made only at the initial geometry. The most important Amdahl's law bottleneck remaining is the solution of the SCF equations. From timing checks during preliminary development stages of the parallel GAMESS, we know that the computation of integrals and their assembly into the Fock operator scales exactly like the two-electron gradient. As may be seen in our two larger examples discussed below, this becomes efficient for large molecules. It is the solution of the Fock equations that causes the decline of efficiency in the "ints + RHF" line of Table III. The SCF solution requires about 300–400 s on

Table III. RHF/6-31G++(*d*, *p*) $(CH_3)_3AlPH_2^-$, direct SCF on the Delta.

	No. of Processors									
	1	2	4	8	16	32	64	128	256	512
Wall clock timings (s)										
Job setup	3	3	3	3	3	3	3	5	5	5
1e^- integrals	21	11	6	4	3	2	2	2	2	2
Initial guess	128	124	122	123	123	123	125	121	122	123
2e^- integrals + RHF	21,893	11,130	5667	3005	1594	890	540	424	292	262
Properties	17	17	18	18	19	20	21	22	22	22
1e^- gradient	95	49	26	13	8	5	4	3	3	3
2e^- gradient	11,153	5607	2811	1480	748	381	196	96	53	32
Total job	33,309	16,942	8653	4647	2497	1424	892	675	499	449
Speedup ratio										
2e^- integrals + RHF	1.00	1.97	3.86	7.28	13.7	24.6	40.5	51.6	75.1	83.7
1e^- gradient	1.00	1.94	3.71	7.38	12.4	18.4	23.1	28.9	31.6	33.1
2e^- gradient	1.00	1.99	3.97	7.53	14.9	29.3	57.0	115.0	209.5	352.3
Total job	1.00	1.97	3.85	7.17	13.3	23.4	37.3	49.4	66.8	74.3
Parallel efficiency										
2e^- integrals + RHF	100	98	97	91	86	77	63	40	29	16
1e^- gradient	100	97	93	92	78	57	36	23	12	6
2e^- gradient	100	99	99	94	93	92	89	91	82	69
Total job	100	98	96	90	83	73	58	39	26	15

one node of the Delta. Improvements to the parallel matrix multiplication routines would be necessary to increase the efficiency on 256 or 512 nodes, where almost all the time is going into Fock solution. Note that our ability to spread out the diagonalization work is limited to two processors in this example as there are only two symmetry blocks in the C_s group.

A more realistic test of the performance of GAMESS on large numbers of nodes comes from the cyclophane HSi $[(CH_2)_3]_3C_6H_3$, shown in Figure 2. It possesses C_3 symmetry, but the test data were deliberately obtained by running in C_1 to simulate a large asymmetric case. For the 6-31G(*d*) basis, there are 292 AOs. Due to the administrative policy for time block allocations on the Delta, we are unable to run this test on fewer than 16 nodes. In effect, we have reached our goal of enabling calculations on molecules that are otherwise too large to handle sequentially. Table IV presents the timing data needed to evaluate the RHF gradient.

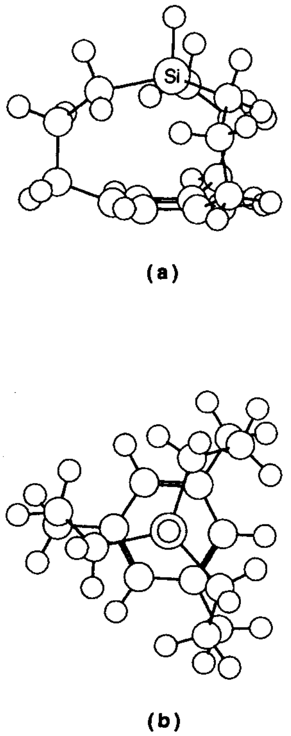

Figure 2. $SiH[(CH_2)_3]_3C_6H_3$. (a). Side view. (b) Top view.

Because the one-node time is unavailable, we cannot present the usual table of speedups and efficiencies. As a substitute, the ratio of time decrease when doubling the number of processors is reported as "consecutive speedup." This stays near two for the 2e^- gradient portion of the calculation and shows that for this total job as many as 256 proces-

Table IV. RHF/6-31G(*d*) $SiH[(CH_2)_3]_3C_6H_3$, direct SCF on Delta.

	No. of Processors					
	16	32	64	128	256	512
Wall clock timings (s)						
Startup	4	4	4	8	7	7
1e^-	12	9	7	7	6	6
Guess	973	974	976	1028	964	968
Integrals + SCF	14,392	7771	4425	2833	1999	1599
Prop	95	96	98	108	112	103
1e^- grad	45	27	19	15	13	10
2e^- grad	11,639	5926	2982	1471	770	422
Total	27,160	14,808	8512	5470	3872	3116
Consecutive speedups						
Integrals + SCF	—	1.85	1.76	1.56	1.42	1.25
2e^- grad	—	1.96	1.99	2.03	1.91	1.82
Total job	—	1.83	1.74	1.56	1.41	1.24

This calculation requires 12 iterations.

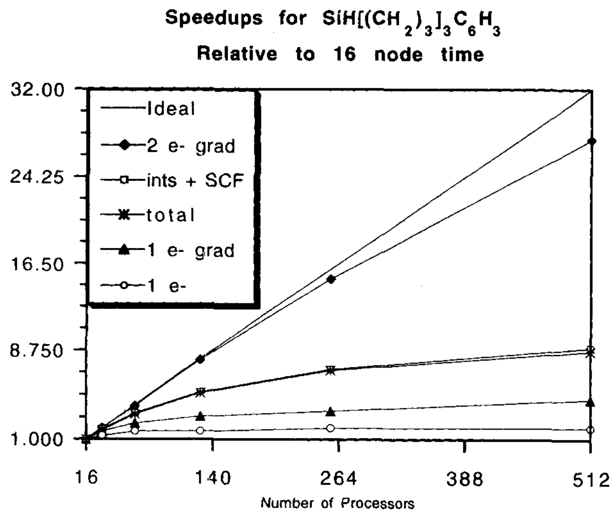

Figure 3. Speedups for $SiH[(CH_2)_3]_3C_6H_3$ relative to 16-node time.

Table V. RHF/6-31G(*d*, *p*) cAMP, direct SCF on Delta.

	No. of Processors					
	16	32	64	128	256	512
Wall clock timings (s)						
Startup	4	5	4	4	4	4
1e$^-$	16	13	11	12	11	11
Guess	2281	2357	2300	2276	2277	2278
Integrals + SCF	20,625	11,768	7167	4928	3841	3286
Prop	250	245	262	261	264	263
1e$^-$ grad	51	33	27	22	20	19
2e$^-$ grad	16,466	8331	4198	2162	1146	631
Total	39,693	22,754	13,971	9666	7565	6494
Consecutive speedups						
Integrals + SCF	—	1.75	1.64	1.45	1.28	1.17
2e$^-$ grad	—	1.98	1.98	1.94	1.87	1.82
Total job	—	1.74	1.63	1.45	1.28	1.16

This calculation requires 14 iterations.

sors can be effectively employed. Another method of plotting this data is given in Figure 3, which gives the speedup relative to the 16-node timings. This plot also confirms that up to 256 processors can be effectively employed.

As a final example of the parallel capabilities of GAMESS, the results of an RHF/6-31G(*d*, *p*) gradient for cyclic adenosine monophosphate ($C_{10}O_6N_5PH_{11}$; cAMP) are presented. The PM3 structure of this molecule is shown in Figure 4. There are 389 AOs for this basis set, which is near the limit of the number of basis functions that can be used in the Delta's 16 Mb of memory. Table V presents the timing data and consecutive speedups for cAMP. The speedup relative to the 16-node timing is plotted in Figure 5.

Usually, one expects speedups to improve if the parallel sections (integrals and Fock formation) of a program become increasingly dominant, as they would if the number of integrals remained N^4. However, the speedups for cAMP are essentially the same as for the cyclophane. cAMP is spatially large enough for many integrals to go to zero because of the large distances between atoms,[32] and therefore the number of integrals is no longer scaling as N^4. The direct SCF on cAMP evaluates 581 million integrals on the first iteration, compared to $N^4/8 = 2862$ million. Differential Fock formation reduces the number of integrals evaluated on the final (14th) cycle to just 214 million. The SCF solution process scales as N^3, and this step detracts from the efficiency because the number of integrals is not in fact growing as N^4. However, it should be noted that the absolute times are still getting quicker as more nodes are added.

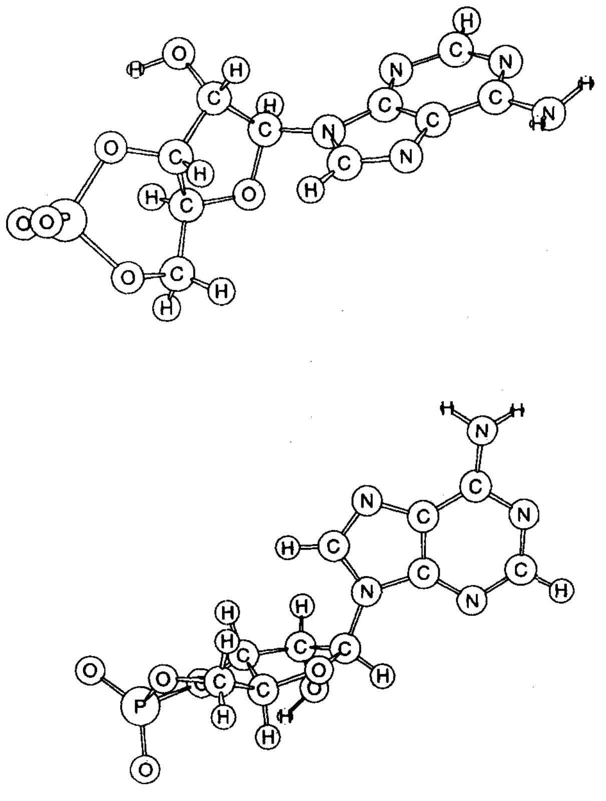

Figure 4. Two views of cAMP.

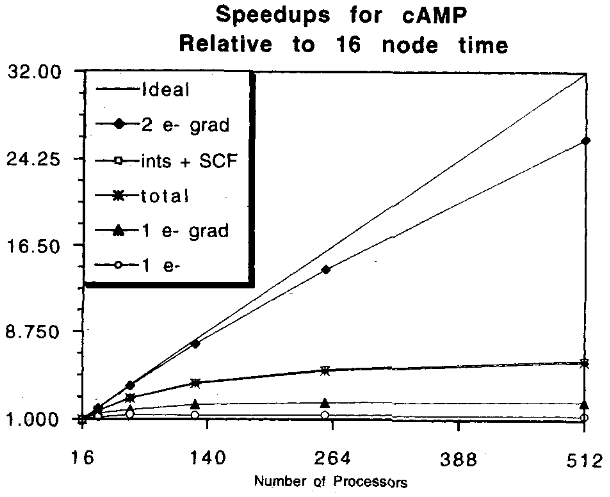

Figure 5. Speedups for cAMP relative to 16-node time.

In practice, one would not run a small problem such as the first example (aluminate ion) on more than about 16 nodes, so the efficiencies for this example on more than 32 nodes are somewhat misleading. To take full advantage of the parallel capabilities of GAMESS, one should tailor the number of nodes to the size of the job. With the current code, we find that a reasonable compromise between short run times and maximum system throughput is to use whatever number of nodes makes the two-electron gradient time, which is highly parallelized, approximate the sequential initial orbital guess time.

Factors such as the amount of symmetry and the spatial separation of the atoms combine to reduce the number of integrals below the formal count of $N^4/8$. Thus, other portions of an SCF calculation remain significant and contribute to a decline in the efficiency of the computation as more processors are added. We suggest the above examples be used as a guideline to the number of nodes that you use for a given problem size. If symmetry or large separation of atoms are involved, you should use fewer nodes (128 or 256) if you want to have "reasonable" efficiencies. Conversely, a large basis on a small molecule may effectively use more nodes. However, as noted before, if you simply want an answer quicker the absolute times are shortest when all 512 nodes of the Delta are brought to bear on a problem.

8. CONCLUSIONS

The GAMESS package offers a wide range of quantum mechanical wave functions, capable of treating systems ranging from closed-shell molecules through bond-breaking reactions. These wave functions may be combined with various run types to perform chemically important tasks, ranging from geometry optimization to transition-state location to reaction path following. These calculations may be performed on the entire periodic table down to radon, apart from the lanthanides. The use of relativistic ECPs permits accurate and affordable calculations even on these heavy atom systems.

Some examples of the types of chemical applications that can be performed with GAMESS can be found in the recent literature. RHF followed by MP2 has been used to explore acid reactivity with aluminate ions.[98] RHF and MP2 were also used to examine the gas-phase formation of the peptide bond.[99] ROHF and TCSCF analytic Hessians have been used to show the biradical nature of some cyclic Si—O compounds.[100] MCSCF calculations have been used to find π bond strengths from rotational barriers.[101] Localized orbitals were used to analyze bonding in benzene analogs.[102] Similar calculations involving ECPs, MCSCF, localized orbitals, and CI have been used to understand ionic vs. covalent bonding in transition metal compounds.[103] The combination of MCSCF wave functions, Hessians, and reaction path following was crucial in unraveling the unexpected manner in which methane inversion occurs.[104] Parallel RHF and electrostatic potentials have been used to explain the regiospecific Diels–Alder cycloaddition reactions of certain benzoquinones.[105]

The SCF portions of GAMESS have been shown to run effectively on systems with many processors. TCGMSG appears to be an excellent tool for the development of efficient, yet portable parallel FORTRAN programs. Scaling past 200 processors will require additional programming attention to the remaining sequential bottlenecks. We have now crossed the barrier from code development to using parallelized GAMESS for production work. As an example (to be published later), we are currently exploring approximately.50 cage molecules similar to the second example for several different properties. These calculations would not be possible if not for the parallel implementation. We intend to continue the development of GAMESS on parallel systems, as well as various other hardware platforms.

9. AVAILABILITY

It is our intention to continue to license usage of GAMESS at no cost to academic or commercial users. We will ask anyone receiving GAMESS to sign a simple letter regarding our copyright to the code, which asks you not to distribute second-generation copies.

If you wish to obtain GAMESS, the preferred method would be to send an E-mail request to the Internet address, mike@si.fi.ameslab.gov, that includes your address (regular and electronic) and the type of hardware on which you wish to run. Before sending a request, please make sure your system has about 10 Mb of available disk storage to spool the incoming mail files.

The authors acknowledge equipment grants or price reductions for computing equipment from IBM Corp., Digital Equipment Corp., and Ardent Corp. during the past decade. A Joint Study Agreement with IBM from 1987 to 1989 enabled GAMESS on the IBM Vector Facility. Digital Equipment provided support for development of the graphics codes. Mariusz Klobukowski adapted the TRUDGE code and provided the FRIEND interface, and has also provided much useful advice on open-shell calculations. Various sections of the code are due to collaborations with Frank Jensen, Henry Kurtz, Dave Garmer, and Walt Stevens. Francis Muguet provided the PostScript interface. Major improvements in the transition metal capability are the result of the persuasive skills of Thomas Cundari. It is a pleasure to acknowledge interactions with the other members of the Gordon Gang in the past decade. Funding for the continued development of GAMESS was provided by the Air Force Office of Scientific Research. Access to the Touchstone Delta machine is provided by an ongoing project with DARPA.

References

1. M. Dupuis, D. Spangler, and J.J. Wendoloski, National Resource for Computations in Chemistry Software Catalog, University of California, Berkeley, CA, Program QG01, 1980.
2. J.F. Stanton, J. Gauss, J.D. Watts, W.J. Lauderdale, and R.J. Bartlett, Quantum Theory Project, University of Florida, Gainesville, FL, 1992.
3. R.D. Amos and J.E. Rice, CADPAC, the Cambridge Analytical Derivatives Package, issue 4, Cambridge University, Cambridge, UK, 1987.
4. R. Shepard, I. Shavitt, R.M. Pitzer, D.C. Comeau, M. Pepper, H. Lischka, P.G. Szalay, R. Ahlrichs, F.B. Brown, and J.-G. Zhao, *Int. J. Quantum Chem. Symp.*, **22,** 149 (1988).
5. J. Almlof, K. Faegri, M. Feyereisen, T.H. Fischer, K. Korsell, and H.P. Luthi, University of Minnesota.
6. M.F. Guest, R.J. Harrison, J.H. van Lenthe, and L.C.H. van Corler, *Theor. Chim. Acta,* **71,** 117 (1987).
7. M.J. Frisch, G.W. Trucks, M. Head-Gordon, P.M.W. Gill, M.W. Wong, J.B. Foresman, B.G. Johnson, H.B. Schlegel, M.A. Robb, E.S. Replogle, R. Gomperts, J.L. Andres, K. Ragavachari, J.S. Binkley, C. Gonzalez, R.L. Martin, D.J. Fox, D.J. Defrees, J. Baker, J.J.P. Stewart, and J.A. Pople, Gaussian 92, Gaussian, Inc., Pittsburgh, PA, 1992.
8. A. Komornicki and H.F. King, Polyatomics Research Institute, Mountain View, CA.
9. (a) M. Dupuis, A. Farazdel, S.P. Karna, and S.A. Maluendes, In *Modern Techniques in Computational Chemistry,* E. Clementi, Ed., Escom, Leiden, The Netherlands, 1990, p. 277; (b) M. Dupuis, S. Chin, and A. Marquez, *Proceedings of the 1992 Vancouver NATO Workshop,* submitted.
10. MELDF was originally written by L. McMurchie, S. Elbert, S. Langhoff, and E.R. Davidson. It has been substantially modified by D. Feller, R. Cave, D. Rawlings, R. Frey, R. Daasch, L. Nitchie, P. Phillips, K. Iberle, and E.R. Davidson, University of Indiana.
11. P.W. Saxe, R.L. Martin, B.H. Lengsfield, and M. Page, University of California, 1990.
12. R.A. Kendall, E. Earl, R. Hernandez, H.L. Taylor, D. O'Neal, J. Nichols, M. Hoffmann, M. Gutowski, X. Wang, J.A. Boatz, J. Anchell, K. Bak, and M. Feyereisen, University of Utah, 1990.
13. K. Andersson, M.P. Fulscher, R. Lindh, P.-A. Mamqvist, J. Olsen, B.O. Roos, A.J. Sadlej, and P.-O. Widmark, MOLCAS v. 2, *User's Guide,* University of Lund, Lund, Sweden, 1991.
14. P.E.M. Sieghbahn, C.W. Bauschlicher, B.O. Roos, P.R. Taylor, A. Heidberg, J. Almlof, S.R. Langhoff, and D.P. Chong.
15. H.-J. Werner and P.J. Knowles, *User's Manual for MOLPRO,* University of Sussex, Sussex, UK, 1991.
16. Wavefunction, Inc., Irvine, CA.
17. P. Pulay, G. Fogarasi, X. Zhou, and P.W. Taylor, *Vib. Spectrosc.,* **1,** 159 (1990).
18. R. Ahlrichs, M. Bar, M. Haser, H. Horn, and C. Kolmel, *Chem. Phys. Lett.,* **162,** 165 (1989).
19. (a) J.J.P. Stewart, MOPAC 6.0, Program 455, Quantum Chemistry Program Exchange, Bloomington, IN; (b) J.J.P. Stewart, *J. Comp.-Aided Mol. Design,* **4,** 1 (1990).
20. L.R. Kahn, P. Baybutt, and D.G. Truhlar, *J. Chem. Phys.,* **65,** 3826 (1976).
21. (a) Program 135, Quantum Chemistry Program Exchange, Bloomington, IN; (b) J.A. Pople and M.S. Gordon, *J. Am. Chem. Soc.,* **89,** 4253 (1967).
22. C.C.J. Roothaan, *Rev. Mod. Phys.,* **23,** 69 (1951).
23. (a) P. Pulay, *Chem. Phys. Lett.,* **73,** 393 (1980); (b) P. Pulay, *J. Comp. Chem.,* **3,** 556 (1982); (c) T.P. Hamilton and P. Pulay, *J. Chem. Phys.,* **84,** 5728 (1986).
24. (a) J. Almlof, K. Faegri, and K. Korsell, *J. Comp. Chem.,* **3,** 285 (1982); (b) D. Cremer and J. Gauss, *J. Comp. Chem.,* **7,** 274 (1986); (c) M. Haser and J. Ahlrichs, *J. Comp. Chem.,* **10,** 104 (1989).
25. (a) R. Seeger and J.A. Pople, *J. Chem. Phys.,* **65,** 265 (1976); (b) R.N. Camp and H.F. King, *J. Chem. Phys.,* **75,** 268 (1981).
26. H. Sellers, *Chem. Phys. Lett.,* **180,** 461 (1991).
27. J.A. Pople and R.K. Nesbet, *J. Chem. Phys.,* **22,** 571 (1954).
28. (a) C.C.J. Roothann, *Rev. Mod. Phys.,* **32,** 179 (1960); (b) S. Huzinaga, *Phys. Rev.,* **120,** 214 (1960); (c) R. McWeeny and G. Diercksen, *J. Chem. Phys.,* **49,** 4852 (1968); (d) E.R. Davidson, *Chem. Phys. Lett.,* **21,** 565 (1973); (e) M.F. Guest and V.R. Saunders, *Mol. Phys.,* **28,** 819 (1974); (f) J.S. Binkley, J.A. Pople, and P.A. Dobosh, *Mol. Phys.,* **28,** 1423 (1974); (g) K. Faegri and R. Manne, *Mol. Phys.,* **31,** 1037 (1976); (h) H. Hsu, E.R. Davidson, and R.M. Pitzer, *J. Chem. Phys.,* **65,** 609 (1976).
29. F.W. Bobrowicz and W.A. Goddard, In *Modern Theoretical Chemistry,* H.F. Schaefer, Ed., Plenum Press, New York, 1977, p. 79, and references therein.
30. (a) R. Carbo and J.M. Riera, *A General SCF Theory,* Springer-Verlag, Berlin, 1978, appendix A; (b) R. Poirier, R. Kari, and I.G. Csizmadia, *Handbook of Gaussian Basis Sets,* Elsevier, Amsterdam, 1985; (c) W.D. Edwards and M.C. Zerner, *Theor. Chim. Acta,* **72,** 347 (1987).
31. D.M. Silver, E.L. Mehler, and K. Ruedenberg, *J. Chem. Phys.,* **52,** 1174 (1970).
32. (a) see appendix I of ref. 22; (b) J.L. Whitten, *J. Chem. Phys.,* **58,** 4496 (1973); (c) see also ref. 24c.
33. E. Hollauer and M. Dupuis, *J. Chem. Phys.,* **96,** 5220 (1992).
34. (a) C.C.J. Roothaan and P.S. Bagus, In *Methods in Computational Physics,* B. Adler, S. Fernbach, and M. Rotenberg, Eds., Academic Press, New York, 1963, p. 47; (b) W.B. Nielsen, *Chem. Phys. Lett.,* **18,** 225 (1973).
35. V.R. Saunders and I.H. Hillier, *Int. J. Quantum Chem.,* **7,** 699 (1973).
36. C.W. Bauschlicher, *J. Chem. Phys.,* **72,** 880 (1980).
37. (a) P. Pulay and T.P. Hamilton, *J. Chem. Phys.,* **88,** 4926 (1988); (b) J.M. Bofill and P. Pulay, *J. Chem. Phys.,* **90,** 3637 (1989).
38. J.A. Pople, J.S. Binkley, and R. Seeger, *Int. J. Quantum Chem.,* **10,** 1 (1976).
39. P. Carsky, B.A. Hess, and L.J. Schaad, *J. Comp. Chem.,* **5,** 280 (1984).
40. P.J. Knowles, J.S. Andrews, R.D. Amos, N.C. Handy, and J.A. Pople, *Chem. Phys. Lett.,* **186,** 130 (1991).
41. (a) B. Brooks and H.F. Schaefer, *J. Chem. Phys.,* **70,** 5092 (1979); (b) B. Brooks, W. Laidig, P. Saxe, N. Handy, and H.F. Schaefer, *Phys. Scripta,* **21,** 312 (1980).
42. M. Dupuis, J.D. Watts, H.O. Villar, and G.J.B. Hurst, *Comp. Phys. Comm.,* **52,** 415 (1989).
43. (a) K. Ruedenberg, M.W. Schmidt, M.M. Gilbert, and S.T. Elbert, *Chem. Phys.,* **71,** 41 (1982); (b) K. Ruedenberg, M.W. Schmidt, and M.M. Gilbert, *Chem. Phys.,* **71,** 51 (1982); (c) K. Ruedenberg, M.W. Schmidt, M.M. Gilbert, and S.T. Elbert, *Chem. Phys.,* **71,** 65 (1982).
44. K. Ruedenberg and K.R. Sundberg, In *Quantum Science,* J.-L. Calais, O. Goscinski, J. Linderberg, and Y. Ohrn, Eds., Plenum, New York, 1976, p. 505.

45. B.O. Roos, In *Advances in Chemical Physics*, K.P. Lawley, Ed., John Wiley, New York, 1987, p. 69.
46. J.W. McIver and A. Komornicki, *J. Am. Chem. Soc.*, **94,** 2025 (1972), Appendix B.
47. H.F. King and A. Komornicki, In *Geometrical Derivatives of Energy Surfaces and Molecular Properties*, P. Jorgensen and J. Simons, Eds., D. Reidel, Dordrecht, 1986, p. 167.
48. Y. Osamura, Y. Yamaguchi, D.J. Fox, M.A. Vincent, and H.F. Schaefer, *J. Mol. Struct.*, **103,** 183 (1983).
49. M. Duran, Y. Yamaguchi, and H.F. Schaefer, *J. Phys. Chem.*, **92,** 3070 (1988).
50. W.D. Gwinn, *J. Chem. Phys.*, **55,** 477 (1971).
51. J.A. Boatz and M.S. Gordon, *J. Phys. Chem.*, **93,** 1819 (1989).
52. W.H. Miller, N.C. Handy, and J.E. Adams, *J. Chem. Phys.*, **72,** 99 (1980).
53. D.G. Truhlar and M.S. Gordon, *Science*, **249,** 491 (1990).
54. (a) D.R. Herschbach and V.W. Laurie, *J. Chem. Phys.*, **35,** 458 (1961); (b) H.B. Schlegel, *Theor. Chim. Acta*, **66,** 333 (1984).
55. H.B. Schlegel, *J. Comp. Chem.*, **3,** 214 (1982).
56. J. Baker, *J. Comp. Chem.*, **7,** 385 (1986).
57. H.F. King, unpublished, adapted from HONDO7.
58. K. Fukui, *Acc. Chem. Res.*, **14,** 363 (1981).
59. (a) K. Ishida, K. Morokuma, and A. Komornicki, *J. Chem. Phys.*, **66,** 2153 (1977); (b) M.W. Schmidt, M.S. Gordon, and M. Dupuis, *J. Am. Chem. Soc.*, **107,** 2585 (1985).
60. K.K. Baldridge, M.S. Gordon, R. Steckler, and D.G. Truhlar, *J. Phys. Chem.*, **93,** 5107 (1989).
61. (a) C. Gonzalez and H.B. Schlegel, *J. Phys. Chem.*, **94,** 5523 (1990); (b) C. Gonzalez and H.B. Schlegel, *J. Chem. Phys.*, **95,** 5853 (1991).
62. H.A. Kurtz, J.J.P. Stewart, and K.M. Dieter, *J. Comp. Chem.*, **11,** 82 (1990).
63. B.H. Lengsfield, J.A. Jafri, D.H. Phillips, and C.W. Bauschlicher, *J. Chem. Phys.*, **74,** 6849 (1981).
64. S. Koseki, M.W. Schmidt, and M.S. Gordon, *J. Phys. Chem.*, **96,** 10768 (1992).
65. (a) see p. 461 of ref. 67; (b) S.F. Boys, In *Quantum Theory of Atoms, Molecules, and Solids*, P.O. Lowdin, Ed., Academic Press, New York, 1966, p. 253; (c) D.A. Kleier, T.A. Halgren, J.H. Hall, and W.N. Lipscomb, *J. Chem. Phys.*, **61,** 3905 (1974).
66. J. Pipek and P.G. Mezey, *J. Chem. Phys.*, **90,** 4916 (1989).
67. C. Edmiston and K. Ruedenberg, *Rev. Mod. Phys.*, **35,** 457 (1963).
68. R.C. Raffenetti, K. Ruedenberg, C.L. Janssen, and H.F. Schaefer, *Theor. Chim. Acta*, to appear.
69. M.W. Schmidt, S. Yabushita, and M.S. Gordon, *J. Phys. Chem.*, **88,** 382 (1984).
70. M. Wolfsberg and L. Helmholz, *J. Chem. Phys.*, **20,** 837 (1952).
71. R. Hoffmann, *J. Chem. Phys.*, **39,** 1397 (1963).
72. J.H. Ammeter, H.-B. Burgi, J.C. Thibeault, and R. Hoffmann, *J. Am. Chem. Soc.*, **100,** 3686 (1978).
73. T.A. Albright, J.K. Burdett, and M.H. Whangbo, *Orbital Interactions in Chemistry*, John Wiley, New York, 1985.
74. S. Huzinaga, J. Andzelm, M. Klobukowski, E. Radzio-Andzelm, Y. Sakai, and H. Tatewaki, *Gaussian Basis Sets for Molecular Calculations*, Elsevier, Amsterdam, 1984.
75. (a) E. Clementi and C. Roetti, *Atom. Data Nucl. Data Tables*, **14,** 177 (1974); (b) J.B. Mann, Los Alamos Technical Reports LA-3690 and LA-3691, 1967.
76. H.F. King, R.E. Stanton, H. Kim, R.E. Wyatt, and R.G. Parr, *J. Chem. Phys.*, **47,** 1936 (1967).
77. (a) R.S. Mulliken, *J. Chem. Phys.*, **23,** 1833 (1955); (b) R.S. Mulliken, *J. Chem. Phys.*, **23,** 1841 (1955).
78. P.O. Lowdin, *J. Chem. Phys.*, **18,** 365 (1950).
79. (a) A.J. Stone, *Chem. Phys. Lett.*, **83,** 233 (1981); (b) S.L. Price and A.J. Stone, *Chem. Phys. Lett.*, **98,** 419 (1983); (c) A.D. Buckingham and P.W. Fowler, *J. Chem. Phys.*, **79,** 6426 (1983).
80. R.F.W. Bader, *Atoms in Molecules—A Quantum Theory*, Oxford University Press, Oxford, UK, 1990.
81. Richard F.W. Bader, bader@sscvax.cis.mcmaster.ca, or Department of Chemistry, McMaster University, Hamilton, Ontario, Canada.
82. A.E. Hansen and T.D. Bouman, *J. Chem. Phys.*, **82,** 5035 (1985).
83. Aage Hansen, naheah@vm.uni-c.dk, or Department of Physical Chemistry, H.C. Orsted Institute, Copenhagen, Denmark.
84. (a) A.E. Reed, L.A. Curtiss, and F. Weinhold, *Chem. Rev.*, **88,** 899 (1988); (b) F. Weinhold and J.E. Carpenter, *The Structures of Small Molecules*, Plenum, New York, 1988, p. 227.
85. Frank Weinhold, weinhold@chem.wisc.edu, or Theoretical Chemistry Institute, Department of Chemistry, University of Wisconsin, Madison, WI.
86. A.D. Isaacson, D.G. Truhlar, S.N. Rai, R. Steckler, G.C. Hancock, B.C. Garrett, and M.J. Redmon, *Comp. Phys. Comm.*, **47,** 91 (1987).
87. (a) M. Dupuis, J. Rys, and H.F. King, *J. Chem. Phys.*, **65,** 111 (1976); (b) M. Dupuis and H.F. King, *Int. J. Quantum Chem.*, **11,** 613 (1977).
88. M. Dupuis and H.F. King, *J. Chem. Phys.*, **68,** 3998 (1978).
89. (a) T. Takada, M. Dupuis, and H.F. King, *J. Chem. Phys.*, **75,** 332 (1981); (b) M. Dupuis and H.F. King, In *Geometrical Derivatives of Energy Surfaces and Molecular Properties*, P. Jorgensen and J. Simons, Eds., D. Reidel, Dordrecht, 1986, p. 167.
90. M. Dupuis and J.D. Watts, *Theor. Chim. Acta*, **71,** 91 (1987).
91. (a) R. Ernenwein, M.M. Rohmer, and M. Bénard, *Comp. Phys. Comm.*, **58,** 305 (1990); (b) M.M. Rohmer, J. Demuynck, M. Bénard, R. Wiest, C. Bachmann, C. Henriet, and R. Ernenwein, *Comp. Phys. Comm.*, **60,** 127 (1990); (c) R.J. Harrison and R.A. Kendall, *Theor. Chim. Acta*, **79,** 337 (1991); (d) M.D. Cooper and I.H. Hillier, *J. Comp.-Aided Mol. Design*, **5,** 171 (1991); (e) S. Kindermann, E. Michel, and P. Otto, *J. Comp. Chem.*, **13,** 414 (1992); (f) H.P. Luthi, J.E. Mertz, M.W. Feyereisen, and J.E. Almlöf, *J. Comp. Chem.*, **13,** 160 (1992); (g) M. Feyereisen and R.A. Kendall, *Theor. Chem. Acta*, **84,** 289 (1993).
92. (a) R.A. Whiteside, J.S. Binkley, M.E. Colvin, and H.F. Schaefer III, *J. Chem. Phys.*, **86,** 2185 (1987); (b) L.A. Covick and K.M. Sando, *J. Comp. Chem.*, **11,** 1151 (1990); (c) R. Wiest, J. Demuynck, M. Bénard, M.M. Rohmer, and R. Ernenwein, *Comp. Phys. Comm.*, **62,** 107 (1991).
93. J.D. Watts and M. Dupuis, *J. Comp. Chem.*, **9,** 158 (1988).
94. (a) R.J. Harrison, *J. Chem. Phys.*, **94,** 5021 (1991); (b) R.J. Harrison and E. Stahlberg, *CSCC Update*, **13,** 5 (1992); (c) M. Schuller, T. Kovar, H. Lischka, R. Shepard, and R.J. Harrison, *Theor. Chim. Acta*, **84,** 489 (1993).
95. A.P. Rendell, T.J. Lee, and R. Lindh, *Chem. Phys. Lett.*, **194,** 84 (1992).
96. *Theoret. Chim. Acta*, **84** (1993).

97. R.J. Harrison, now at Pacific Northwest Laboratory, v. 4.03, available in director/pub/tcgmsg from host ftp.tcg.anl.gov.

98. R. Damrauer, M.M. Krempp, M.W. Schmidt, and M.S. Gordon, *J. Am. Chem. Soc.*, **115,** 5218 (1993).

99. J.H. Jensen, K.K. Baldridge, and M.S. Gordon, *J. Phys. Chem.*, **96,** 8340 (1992).

100. M.W. Schmidt, K.A. Nguyen, M.S. Gordon, and J.A. Montgomery, *J. Am. Chem. Soc.*, **113,** 5998 (1991).

101. T.L. Windus and M.S. Gordon, *J. Am. Chem. Soc.*, **114,** 9559 (1992).

102. N. Matsunaga, T.R. Cundari, M.W. Schmidt, and M.S. Gordon, *Theor. Chim. Acta*, **83,** 57 (1992).

103. T.R. Cundari, *J. Am. Chem. Soc.*, **114,** 7879 (1992).

104. M.S. Gordon and M.W. Schmidt, *J. Am. Chem. Soc.*, to appear.

105. G.A. Kraus, J. Li, M. Gordon, and J. Jensen, *J. Am. Chem. Soc.*, **115,** 5859 (1993).