

**Manual****MBPAC 2012-4A**

Hannah Leverentz, Erin E. Dahlke, Hai Lin, Bo Wang, Jeremy O. B. Tempkin,  
Helena Qi, and Donald G. Truhlar

*Department of Chemistry and Supercomputing Institute, University of Minnesota,  
Minneapolis, MN 55455-0431*

Program Version: 2012-4A

Program Version Date: September 27, 2012

Manual Version Date: July 31, 2021

Copyright 2007 – 2012

Abstract: MBPAC is a computer program for calculating electronic energies, gradients, and/or Hessians of molecular clusters by the electrostatically embedded many-body method (EE-MB). This program can carry out a two-body or three-body many-body expansion for the cluster of interest, it can perform the many-body expansion on the whole energy or just on the correlation energy, and it can use either point charges or screened charges for the electrostatic embedding. The program works in conjunction with either *Gaussian 09* (for all types of calculations) or *Molpro* (for single-point energy calculations), which is called by a script to generate the needed electronic structure data. The code is fully parallel in that all monomer, dimer, and trimer calculations (and the full Hartree–Fock calculation if the EE-MB-CE option has been selected) can be run simultaneously.

## Licensing

*MBPAC* - version 2012-4A is licensed under the [Apache License, Version 2.0](#).

The manual of *MBPAC* - version 2012-4A is licensed under [CC-BY-4.0](#).

Publications of results obtained with the *MBPAC* - version 2012-4A software should cite the program and/or the article describing the program.

No guarantee is made that this software is bug-free or suitable for specific applications, and no liability is accepted for any limitations in the mathematical methods and algorithms used within. No consulting or maintenance services are guaranteed or implied.

The use of the *MBPAC* - version 2012-4A implies acceptance of the terms of the licenses.

## Table of Contents

TITLE PAGE AND ABSTRACT.....	1
TABLE OF CONTENTS .....	3
1. INTRODUCTION .....	7
2. REFERENCES FOR MBPAC PROGRAM.....	9
3. GENERAL PROGRAM DESCRIPTION .....	10
4. THEORETICAL BACKGROUND.....	12
4.A. MANY-BODY EXPANSION.....	12
4.B. ELECTROSTATICALLY EMBEDDED MANY-BODY EXPANSION WITH POINT CHARGES.....	13
4.C. SCREENED CHARGE MODEL.....	16
4.D. ELECTROSTATICALLY EMBEDDED MANY-BODY EXPANSION OF THE CORRELATION ENERGY.....	18
5. INSTALLING MBPAC.....	20
5.A. INSTALLATION INSTRUCTIONS.....	20
6. DESCRIPTION OF FILES IN MBPAC.....	25
6.A. SOURCE CODE .....	25
6.A.1. SERIAL SOURCE CODE FILES .....	25
6.A.2. SUBPROGRAM LIST.....	27
6.A.3. PARALLEL SOURCE CODE FILES.....	38
6.A.4. PARALLEL SUBPROGRAM LIST .....	39
6.B. FILES REQUIRED TO RUN MBPAC .....	40
6.B.1. G09SHUTTLE.PL SCRIPT.....	40
6.B.2. MOLPSHUTTLE.PL SCRIPT.....	41
6.B.3. EX_SHUTTLE, G09-EX-SHUTTLE, AND GAU_EXTERNAL_2 SCRIPTS	41
6.C. FILES CREATED DURING ALL MBPAC RUNS .....	42

6.C.1. THE OUTPUT FILE.....	42
6.C.2 THE X_Y.EEMB FILES.....	43
6.C.3 THE FULLHF_1.TXT FILE.....	44
6.D. FILES CREATED ONLY DURING GAUSSIAN RUNS .....	44
6.D.1. THE N.G09 AND MMN.G09 FILES.....	44
6.D.2. THE N.OUT AND MMN.OUT FILES .....	45
6.D.3. TEST.FCHK and MMTEST.FCHK.....	45
6.D.4 ADDITIONAL FILES CREATED DURING AN EXTERNAL OPTIMIZATION .....	45
6.D.5 THE MMCORRN.OUT FILE .....	45
6.E. FILES CREATED ONLY DURING MOLPRO RUNS.....	46
6.E.1. THE N.INP AND N.LAT FILES .....	46
6.E.2. THE N.OUT AND N.XML FILES.....	46
7. USING MBPAC .....	47
7.A. THE PERL SCRIPT EEMB.PL.....	47
7.B. THE RESTART OPTION .....	48
8. INPUT FILE STRUCTURE AND EXPLANATION OF INPUT VARIABLES .....	49
8.A. GENERAL SECTION.....	50
8.B. FRAGMENT SECTION.....	52
8.C. JOB SECTION.....	54
9. THE MBPAC TEST SUITE.....	61
9.A. INTRODUCTION TO THE TEST SUITE.....	61
9.B. THE TEST RUNS.....	61
9.B.1. TEST 1 .....	61
9.B.2. TEST 2 .....	61
9.B.3. TEST 3 .....	62

9.B.4. TEST 4 .....	63
9.B.5. TEST 5 .....	63
9.B.6. TEST 6 .....	63
9.B.7. TEST 7 .....	63
9.B.8. TEST 8 .....	64
9.B.9. TEST 9 .....	64
9.B.10. TEST 10 .....	64
9.B.11. TEST 11 .....	66
9.B.12. TEST 12 .....	66
9.B.13. TEST 13 .....	66
9.B.14 TEST 14 .....	67
9.B.15 TEST 15 .....	67
9.B.16 TEST 16 .....	67
9.B.17 TEST 17 .....	68
9.B.18. TEST 18 .....	68
9.B.19. TEST 19 .....	68
9.B.20. TEST 20 .....	68
9.B.21 TEST 21 .....	68
9.B.22. TEST 22 .....	69
9.B.23. TEST 23 .....	69
9.B.24. TEST 24 .....	69
9.B.25. TEST 25 .....	70
9.B.26. TEST 26 .....	70
9.B.27. TEST 27 .....	70

10. COMPUTERS, OPERATING SYSTEMS, AND COMPILERS ON WHICH THE CODE WAS TESTED .....	71
---	----

11. CITED REFERENCES .....	77
12. BIBLIOGRAPHY .....	79
13. REVISION HISTORY .....	80
13.A. VERSION 1.0.....	80
13.B. Version 2007.....	80
13.C. Version 2007-2.....	80
13.D. Version 2009 (May 28, 2009) .....	80
13.E. Version 2009-2 (June 25, 2009) .....	81
13.F. Version 2011 (March 18, 2011) .....	82
13.G. Version 2011-2 (April 13, 2011) .....	83
13.H. Version 2011-3 (June 21, 2011).....	84
13.I. Version 2011-3A (July 7, 2011).....	85
13.J. Version 2011-4 (July 13, 2011).....	86
13.K. Version 2011-5 (November 22, 2011).....	86
13.L. Version 2012 (February 13, 2012) .....	87
13.M. Version 2012-2 (May 16, 2012).....	88
13.N. Version 2012-3 (May 23, 2012).....	89
13.O. Version 2012-4 (July 11, 2012) .....	89
13.P. Version 2012-4A (September 27, 2012).....	90

## Chapter One

### Introduction

MBPAC is a computer program for calculating single-point energies, gradients, Hessians, geometry optimizations, and/or population analyses of molecular clusters using the electrostatically embedded many-body (EE-MB) method.<sup>1</sup> The current version allows for energy and population analysis calculations using a two-body expansion (EE-PA) or a three-body expansion (EE-3B). The background charges used in the calculation are user defined, enabling one to utilize any charge model for the electrostatic embedding. With this version of the program, calculations can be carried out using any electronic structure method available in GAUSSIAN 09<sup>2</sup> that can be used in a two-layer ONIOM calculation (see the keyword *ONIOM* in the GAUSSIAN 09 manual) or any electronic structure method available in MOLPRO<sup>3</sup>. Any pre-defined or user-defined basis set that is compatible with GAUSSIAN 09 or MOLPRO may be used. MBPAC 2012-4A can treat clusters but does not yet include periodic boundary conditions.

Four kinds of calculations can be done with MBPAC 2012-4A using GAUSSIAN 09: energies, gradients, Hessians, and geometry optimizations (which use the GAUSSIAN 09 external optimizer). MOLPRO can be used only for energy calculations. The list below indicates the order in which each of the GAUSSIAN 09 calculations will be done within the program. If the user has not selected one of these calculations then the program will go on to the next calculation in the list.

1. *External optimization.* The MBPAC 2012-4A program has been interfaced with the external optimizer in the GAUSSIAN 09 software package. This is currently

the only means for geometry optimization. No restart option is currently implemented for geometry optimizations.

2. *Hessian calculation.* Frequencies and normal mode coordinates are also calculated in the Hessian calculation by mass-scaling and diagonalizing the Hessian matrix.
3. *Gradient calculation.* The gradient calculation is carried out only if a Hessian was not calculated since a Hessian calculation outputs both a gradient and an energy.
4. *Energy calculation.* An energy calculation is carried out only if a Hessian or gradient calculation was not done because both of these calculations output an energy as well.

Chapter 2 gives the references for the MBPAC program in three different styles, and Chapter 3 gives a general program description. Chapter 4 presents the theoretical background of the EE-PA and EE-3B methods, Chapter 5 describes the installation, and Chapter 6 provides a complete listing of all the files that comprise the MBPAC package. Chapter 7 discusses how to use the MBPAC program, Chapter 8 discusses the structure of the input file, Chapter 9 presents the test jobs, and Chapter 10 describes the computers, operating systems, and compilers the code has been tested on. Chapter 11 gives the cited references, Chapter 12 gives a bibliography of all published papers from our group pertaining to the EE-MB method, and Chapter 13 gives the revision history.

## Chapter Two

### References for MBPAC Program

The recommended reference for the current version of the code is given below in three styles, *J. Chem. Phys.* style, *J. Amer. Chem. Soc.* style, and *Chem. Phys. Lett.* style.

*J. Chem. Phys.* style:

H. Leverentz, E. E. Dahlke, H. Lin, B. Wang, J. O. B. Tempkin, H. W. Qi, and D. G. Truhlar, MBPAC 2012-4A (University of Minnesota, Minneapolis, 2012).

*J. Amer. Chem. Soc.* style:

Leverentz, H.; Dahlke, E. E.; Lin, H.; Wang, B; Tempkin, J. O. B.; Qi, H. W.; Truhlar, D. G. MBPAC 2012-4A, University of Minnesota, Minneapolis, 2012.

*Chem. Phys. Lett.* style:

H. Leverentz, E.E. Dahlke, H. Lin, B. Wang, J.O.B. Tempkin, H.W. Qi, D.G. Truhlar, MBPAC 2012-4A. University of Minnesota, Minneapolis, 2012.

The user should also reference the GAUSSIAN 09 electronic structure program (see Reference 2 in Chapter 11) or the MOLPRO 2010 electronic structure program (see Reference 3 in Chapter 11).

## Chapter Three

### General Program Description

MBPAC is written in FORTRAN77 and uses a PERL script to interface with GAUSSIAN 09 or MOLPRO 2010. The program is written using modular subprograms called “hooks”. There are four main hooks: energy hook (ehook), gradient hook (ghooks), Hessian hook (hhook), and optimization hook (ohook). Each one of these hooks is designed to be able to utilize a variety of electronic structure methods and basis sets. All of the hooks interface with the GAUSSIAN 09 software program and the energy hook also interfaces with the MOLPRO software program. The user must provide an input file (described in Chapter 8) that provides, in addition to other information, the geometry of the cluster, an array that identifies each atom for each fragment, and the background charges to be used in the electrostatic embedding.

For single point energies, gradients, and Hessians MBPAC uses the information provided in the input file to break the cluster up into all possible fragment combinations for the EE-MB calculation requested and write the corresponding GAUSSIAN 09 or MOLPRO input files. When the input file is complete the *g09shuttle* or *molpshuttle* script is called to run the GAUSSIAN 09 or MOLPRO calculation, parse the required data, and place it into a file. When all electronic structure calculations are finished, the MBPAC program extracts all of the data from the file and calculates the EE-PA or EE-3B energy. The overall control of the program is:

MBPAC ↔ *g09shuttle/molpshuttle* ↔ GAUSSIAN 09/ MOLPRO ↔ *g09shuttle/molpshuttle*

↔ MBPAC

Geometry optimizations are carried out using the external optimizer of GAUSSIAN 09. The overall control for this procedure is:

MBPAC ↔ *g09shuttle* ↔ GAUSSIAN 09 ↔ *Gau\_External\_2* ↔ *g09shuttle* ↔ MBPAC

If one chooses to do a geometry optimization the primary MBPAC calculations will call a GAUSSIAN 09 optimization with the *external* keyword. This GAUSSIAN 09 calculation calls an external PERL script *Gau\_External\_2*, which will provide the MBPAC energy and gradient needed for optimization. *Gau\_External\_2* will call a secondary MBPAC calculation and pass the results to GAUSSIAN 09. When GAUSSIAN 09 finishes the optimization, it will return the optimized geometry to the primary MBPAC calculations. The optimized geometry will be printed along with the energy and gradient.

## Chapter Four

### Theoretical Background

This chapter describes the EE-MB method.

#### 4.A. MANY-BODY EXPANSION

For a system containing  $N$  particles the total energy of the system can be written, without any approximation, as:<sup>4</sup>

$$V = V_1 + V_2 + V_3 + \dots + V_N \quad (1)$$

where

$$V_1 = \sum_i^N E_i \quad (2)$$

$$V_2 = \sum_{i < j}^N (E_{ij} - E_i - E_j) \quad (3)$$

$$V_3 = \sum_{i < j < k}^N [(E_{ijk} - E_i - E_j - E_k) - (E_{ij} - E_i - E_j) - (E_{ik} - E_i - E_k) - (E_{jk} - E_j - E_k)] \quad (4)$$

and so on for higher order terms, where  $E_i$ ,  $E_{ij}$ ,  $E_{ijk}$ , ... are the energies of a monomer, dimer, trimer, and so forth. In general it is assumed that the series in equation 1 converges rapidly, and in general only the first few terms are kept. If only the first and second terms of equation 1 are kept, the total energy becomes

$$E_{PA} = \sum_{i < j}^N E_{ij} - (N - 2) \sum_i^N E_i \quad (5)$$

and one is said to have made the pairwise additive (PA) approximation. If one keeps the first three terms of equation 1, one is said to have made the three-body approximation (3B), and the total energy can be written as :

$$E_{3B} = \sum_{i<j<k}^N E_{ijk} - (N-3) \sum_{i<j}^N E_{ij} + \frac{(N-2)(N-3)}{2} \sum_i^N E_i \quad (6)$$

In general the pairwise additive approximation is accurate enough to determine qualitative trends, however, if one is interested in quantitative accuracy, inclusion of higher-order terms is necessary.<sup>4</sup> For a system in which the many-body terms are large one may need to include several of the higher-order terms in order to obtain the desired accuracy.

#### 4.B. ELECTROSTATICALLY EMBEDDED MANY-BODY EXPANSION WITH POINT CHARGES

In order to speed up the convergence of equation 1, one can embed each  $n$ -body cluster in a field representing the other  $N-n$  atoms. In the electrostatically embedded many-body expansion, with the point charge option (the screened charge option is discussed in Section 4.C), this is done by placing atom-centered point charges at the positions of the atoms in the other  $N-n$  fragments. When this is done equations 2 – 4 can be rewritten as

$$V_1 = \sum_i^N E'_i \quad (7)$$

$$V_2 = \sum_{i<j}^N (E'_{ij} - E'_i - E'_j) \quad (8)$$

and

$$V_3 = \sum_{i<j<k}^N [(E'_{ijk} - E'_i - E'_j - E'_k) - (E'_{ij} - E'_i - E'_j) - (E'_{ik} - E'_i - E'_k) - (E'_{jk} - E'_j - E'_k)] \quad (9)$$

where the prime denotes the energy of an embedded fragment. By an embedded fragment we mean the fragment embedded in the field of point charges as described above.

The electrostatically embedded pairwise additive (EE-PA) and electrostatically embedded three-body (EE-3B) energies are defined analogously to equations 5 and 6 as

$$E_{\text{EE-PA}} = \sum_{i < j}^N E'_{ij} - (N - 2) \sum_i^N E'_i \quad (10)$$

and

$$E_{\text{EE-3B}} = \sum_{i < j < k}^N E'_{ijk} - (N - 3) \sum_{i < j}^N E'_{ij} + \frac{(N - 2)(N - 3)}{2} \sum_i^N E'_i \quad (11)$$

where the primes in equations 10 – 11 have the same meaning as in 7 – 9. The term EE-MB denotes an electrostatically embedded many-body expansion of unspecified order.

If electrostatic embedding is carried out to  $N$ th order (i.e., EE-MB), where  $N$  is equal to the number of fragments in the whole cluster, one will get the exact result. This result will be independent of the choice of charge model used and will give the same results with or without embedding if equation 1 is used without truncation. However it has been shown<sup>5</sup> that the rate of the convergence for the series (i.e., the accuracy if one truncates after a given order of many-body terms) depends strongly on embedding.

One can also use a similar expansion to determine the MB or EE-MB approximation of the components of the full-system dipole moment:

$$m_n^{\text{EE-PA}} = \sum_{j \neq i}^N m_n^{ij} - (N - 2) m_n^i \quad (12)$$

$$m_n^{\text{EE-3B}} = \mathring{a}_{j^1i}^N \mathring{a}_{k^1i}^N m_n^{ijk} - (N-3) \mathring{a}_{j^1i}^N m_n^{ij} + \frac{(N-2)(N-3)}{2} m_n^i \quad (13)$$

In eqs 12 and 13,  $m_n^{\text{EE-MB}}$  is the EE-MB approximation to the  $\nu$ -component of the dipole moment, where  $\nu$  can be replaced by  $x$ ,  $y$ , or  $z$ . The expressions  $m_n^{ijk}$ ,  $m_n^{ij}$ , and  $m_n^i$  are the  $\nu$ -component of the dipole resulting from the wave functions or electron densities of trimer  $ijk$ , dimer  $ij$ , and monomer  $i$ , respectively.

Similarly, one may find the MB or EE-MB approximation of the full-system partial charge distribution that would result from various types of population or charge analysis, such as Mulliken<sup>6</sup> or CHelpG<sup>7</sup>:

$$q_A^{\text{EE-PA}} = \mathring{a}_{j^1i} q_A^{ij} - (N-2) q_A^i \quad (14)$$

$$q_A^{\text{EE-3B}} = \mathring{a}_{j^1i}^N \mathring{a}_{k^1i}^N q_A^{ijk} - (N-3) \mathring{a}_{j^1i}^N q_A^{ij} + \frac{(N-2)(N-3)}{2} q_A^i \quad (15)$$

In eqs 14 and 15,  $q_A^{\text{EE-MB}}$  is the EE-MB partial charge assigned to atom  $A$  belonging to monomer  $i$ , and  $q_A^{ijk}$ ,  $q_A^{ij}$ , and  $q_A^i$  are the partial charges assigned to atom  $A$  by the population or charge analyses of the wave functions or electron densities of trimer  $ijk$ , dimer  $ij$ , and monomer  $i$ , respectively.

There are several different ways in which the background point charges to be used in the embedding can be chosen. Three possible ways to determine the background point charges are:

- A. Determine a charge representation for the entire cluster; then, for each monomer, dimer, or trimer, represent the other  $N - 1$ ,  $N - 2$ , or  $N - 3$  fragments with the charges from this full-system charge calculation.
- B. For each monomer present in the cluster, determine a charge representation for the monomer in the same geometry that it has in the cluster, and then for each monomer, dimer, or trimer, represent the other  $N - 1$ ,  $N - 2$ , or  $N - 3$  fragments with the charges from these monomer calculations.
- C. For each type of molecule present in the cluster determine a charge representation for the geometrically relaxed gas-phase monomer, and then for each monomer, dimer, or trimer, represent the other  $N - 1$ ,  $N - 2$ , or  $N - 3$  fragments with the charges from these monomer calculations.

One can see that for a very large cluster the use of charge models A and B could prove to be expensive and time consuming, and it has been shown that for water clusters, which are known to have large many-body effects<sup>8</sup> (and thus slow convergence of equation 1), the use of strategy C gives sufficient accuracy to give good quantitative results.<sup>5</sup>

Another important choice is which charge representation to use (e.g., Mulliken,<sup>6</sup> Löwdin,<sup>9, 10</sup> redistributed Löwdin,<sup>11</sup> CM4,<sup>12</sup> or force field). The optimal set of charges may be different for each system, and it is up to the user to choose a set of charges that meets his or her desired accuracy level.

#### 4.C. SCREENED CHARGE MODEL

Background charges can also be represented by screened charges<sup>16</sup> rather than point charges. The screened charge scheme improves the point charge scheme by taking into account the penetration effects. The charge density of an atom is represented by two

components: (i) a smeared charge, of magnitude  $-n_{\text{screen}}$ , distributed like electrons in the orbital  $j_{(n)} = Ar^{n-1}\exp(-Zr)$ , which models the extension of the MM electron density and (ii) the rest of the charge, which is located at the nucleus. The comparison of a point charge model and a screened charge model is shown in Figure 1. Note that this screening procedure is sometimes called outer density screening (ODS) to distinguish it from other screening models.

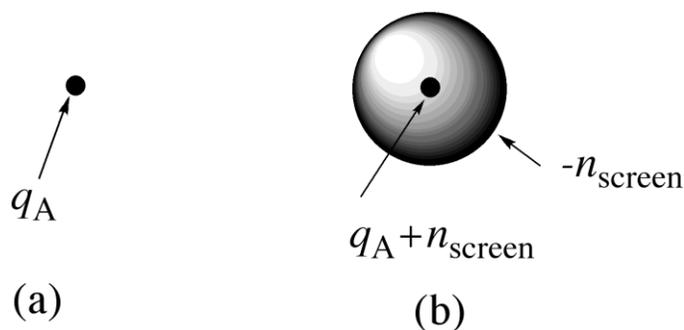


Figure 1. Comparison between (a) a point charge model and (b) a screened charge model of an MM atom A. The total smeared charge in model (b) is  $-n_{\text{screen}}$ , representing  $n_{\text{screen}}$  electrons.

Based on the new model, we can calculate the effective charge of the atom A as

$$q_A^* = q_A + n_{\text{screen}}f(Zr)\exp(-2Zr) \quad (16)$$

where the scaling factor  $f$  is

$$\begin{aligned} f(Zr) &= 1 + Zr & n = 1 \\ &= 1 + \frac{3}{2}Zr + (Zr)^2 + \frac{1}{3}(Zr)^3 & n = 2 \\ &= 1 + \frac{5}{3}Zr + \frac{4}{3}(Zr)^2 + \frac{2}{3}(Zr)^3 + \frac{2}{9}(Zr)^4 + \frac{2}{45}(Zr)^5 & n = 3 \\ &= 1 + \frac{7}{4}Zr + \frac{3}{2}(Zr)^2 + \frac{5}{6}(Zr)^3 + \frac{1}{3}(Zr)^4 + \frac{1}{10}(Zr)^5 + \frac{1}{45}(Zr)^6 + \frac{1}{315}(Zr)^7 & n = 4 \end{aligned} \quad (17)$$

In the screened charge model of eq. (16), the effective charge has two terms: the conventional point charge and an addition term to include the penetration effects. The latter term goes to zero when  $r$  approaches infinity and goes to  $q_A + n_{\text{screen}}$  when  $r$  approaches 0. The parameters of  $Z$  for common atoms (H, C, N, O, F, Si, P, S, Cl and Br) have been optimized and they have been listed in the Table 1.

**Table 1.**  $\zeta$  values used in the Slater-type orbital

Atom	H	B	C	N	O	F	Si
optimized parameters	1.32		0.92	0.92	1.20	1.16	0.73
MSB parameters <sup>a</sup>	1.32	0.72	0.87	1.01	1.12	1.24	0.74
Atom	P	S	Cl	Ge	As	Se	Br
optimized parameters	0.68	0.90	0.98				0.91
MSB parameters <sup>a</sup>	0.81	0.88	0.95	0.83	0.88	0.95	1.01

<sup>a</sup> modified Strand-Bonham (MSB) parameters (optimized parameter for H and half of the Strand-Bonham parameters for B through Br)

#### 4.D. ELECTROSTATICALLY EMBEDDED MANY-BODY EXPANSION OF THE CORRELATION ENERGY

If one is using a post-Hartree–Fock level of wave function theory such as MP2, QCISD(T), or CCSD(T), one can define the system’s correlation energy ( $\Delta V_{\text{corr}}$ ) as the energy at the higher level of electronic structure theory ( $V$ ) minus the Hartree–Fock energy ( $V_{\text{HF}}$ ) using the same basis set; that is,  $\Delta V_{\text{corr}} = V - V_{\text{HF}}$ . One can then expand the correlation energy in exactly the same way that one expands the total energy in the MB or EE-MB approximations in equations 1–4 or 7–9:

$$\Delta V_{\text{corr}} = \Delta V_{\text{corr}}^{(1)} + \Delta V_{\text{corr}}^{(2)} + \Delta V_{\text{corr}}^{(3)} + \dots + \Delta V_{\text{corr}}^{(N)} \quad (18)$$

where, for example,

$$\Delta V_{\text{corr}}^{(2)} = \sum_{i < j}^N \left( \Delta E_{ij}^{\text{corr}} - \Delta E_i^{\text{corr}} - \Delta E_j^{\text{corr}} \right) \quad (19)$$

and, for example,

$$\Delta E_{ij}^{\text{corr}} = E_{ij} - E_{ij}^{\text{HF}} \quad (20)$$

with  $E_{ij}$  being the energy of dimer  $ij$  at the post-Hartree–Fock level of wave function theory (electrostatically embedded in any chosen manner) and with  $E_{ij}^{\text{HF}}$  being the energy of dimer  $ij$  at the Hartree–Fock level of wave function theory.

One can then approximate the total energy of the system as the sum of the Hartree–Fock energy of the entire system with any desired number of terms in the expansion of the correlation energy. When one has done this, one has made the EE-MB-CE approximation to the total energy of the system. The EE-PA-CE energy is written as

$$V_{\text{EE-PA-CE}} = V_{\text{HF}} + \Delta V_{\text{corr}}^{(1)} + \Delta V_{\text{corr}}^{(2)} \quad (21)$$

and the EE-3B-CE energy is written as

$$V_{\text{EE-3B-CE}} = V_{\text{HF}} + \Delta V_{\text{corr}}^{(1)} + \Delta V_{\text{corr}}^{(2)} + \Delta V_{\text{corr}}^{(3)} \quad (22)$$

Although these methods scale roughly as  $N^4$  with system size due to the need for a full-system Hartree–Fock calculation, this is still a much more favorable scaling than any of the post-Hartree–Fock levels of theory, and it can yield results that are within 1 kcal/mol of the conventional calculations.<sup>17</sup>

## Chapter Five

### Installing MBPAC

A step-by-step procedure for installing MBPAC on a Unix computer is given here. Compilation of the MBPAC program can be accomplished with the PERL script *configure*. The test runs described in Chapter 8 illustrate the proper way in which to use the program.

#### 5.A. INSTALLATION INSTRUCTIONS

Step 1:

The MBPAC program should have been obtained in the tar format with the following name: *mbpac2012-4A.tar.gz*. This file should be placed in the directory in which the user wishes to install MBPAC, and then the following two commands should be executed:

```
gunzip mbpac2012-4A.tar.gz
```

```
tar -xvf mbpac2012-4A.tar
```

Once these two commands have been executed the directory structure in the next step should have been created. Please make sure that this is true.

Step 2:

Verify that the files have been placed into the directory structure as follows.

In the *mbpac2012-4A* directory

<i>basis/</i>	<i>configure</i>	<i>exe/</i>	<i>psrc/</i>
<i>script/</i>	<i>src/</i>	<i>testo/</i>	<i>testrun/</i>

In the *basis* directory:

<i>mg3s.gbs</i>	<i>b2.gbs</i>	<i>acTZ_noHe.gbs</i>
<i>631+gdp_noHe.gbs</i>		<i>JUN-cc-pVTdZ.mbs</i>

The *exe* directory should be empty.

In the *psrc* directory:

<i>common.inc</i>	<i>eehed.f</i>	<i>ehooks.f</i>	<i>frag.f</i>
<i>freq.f</i>	<i>ghooks.f</i>	<i>hhooks.f</i>	<i>main.f</i>
<i>ohooks.f</i>	<i>read.f</i>	<i>worker.f</i>	

In the *script* directory:

<i>checktestrun</i>	<i>check_all.pl</i>	<i>clean.pl</i>	<i>eemb.pl</i>
<i>ex_shuttle</i>	<i>g09shuttle.pl</i>	<i>g09-ex-shuttle.pl</i>	<i>Gau_External_2</i>
<i>mbcompile</i>	<i>molpshuttle.pl</i>	<i>testall.pl</i>	<i>updatetesto</i>
<i>updatetestrun</i>			

In the *src* directory:

<i>common.inc</i>	<i>eehed.f</i>	<i>ehooks.f</i>	<i>frag.f</i>
<i>freq.f</i>	<i>ghooks.f</i>	<i>hhooks.f</i>	<i>main.f</i>
<i>ohooks.f</i>	<i>read.f</i>		

In the *testrun* directory:

<i>test1/</i>	<i>test2/</i>	<i>test3/</i>
<i>test4/</i>	<i>test5/</i>	<i>test6/</i>
<i>test7/</i>	<i>test8/</i>	<i>test9/</i>
<i>test10/</i>	<i>test11/</i>	<i>test12/</i>
<i>test13/</i>	<i>test14/</i>	<i>test15/</i>

<i>test16/</i>	<i>test17/</i>	<i>test18/</i>
<i>test19/</i>	<i>test20/</i>	<i>test21/</i>
<i>test22/</i>	<i>test23/</i>	<i>test24/</i>
<i>test25/</i>	<i>test26/</i>	<i>test27/</i>

In each of the *testrun/testn* directories (where *n* is the number of the test run) :

*eemb.pl*            *testn.inp*

Test run 7 illustrates the restart option (see section 8.B.) and the *test7* directory contains the following additional files:

*x\_1.eemb*            *y\_2.eemb*

where *x* = 1, 2, 3, and 4 and where *y* = 1, 3, 4, 5, and 6. Please see section 8.A.7 for more information.

In the *testo* directory:

<i>test1/</i>	<i>test2/</i>	<i>test3/</i>
<i>test4/</i>	<i>test5/</i>	<i>test6/</i>
<i>test7/</i>	<i>test8/</i>	<i>test9/</i>
<i>test10/</i>	<i>test11/</i>	<i>test12/</i>
<i>test13/</i>	<i>test14/</i>	<i>test15/</i>
<i>test16/</i>	<i>test17/</i>	<i>test18/</i>
<i>test19/</i>	<i>test20/</i>	<i>test21/</i>
<i>test22/</i>	<i>test23/</i>	<i>test24/</i>
<i>test25/</i>	<i>test26/</i>	<i>test27/</i>

In each of the *testo/testn* directories (where *n* is the number of the test run):

*testn.out*

Step 3:

Change the working directory to the *mbpac2012-4A* directory, and run the script *configure* by typing:

```
./configure<Return>
```

This script will create a file in your home directory named *.mbpac\_path* stating where the MBPAC directory structure is located. This file is used by other scripts to locate MBPAC on the user's system. The *configure* script will look for one of the following serial compilers: *g77*, *xl*f, or *ifort*. It will also look for one of the following MPI compilers: *mpxlf*, *ifort*, or *mpif90*. If it successfully finds one of the MPI compilers it will ask if you would like to attempt to compile the parallel version of the code (found in the *psrc* directory). If you say no, or, if it is unable to find an MPI compiler, it will compile the serial version of the code (found in the *src* directory). Both the parallel and serial compilations are carried out by running the script *mbcompile* in the *script* directory. Successful compilation of the serial version of the code will place the executable *mbpac.exe* in the directory *exe*. Successful compilation of the parallel version of the code will place the executable *pmbac.exe* in the *exe* directory.

If the *configure* scripts is unable to find any MPI or any serial compilers, or if you tell it that it will ask you to manually input the name of your compiler into the first line of the *Makefile*. If the user would like to compile the serial version of the code they must edit the *Makefile* in the directory *src* as described above, and then compile by typing:

```
gmake mbpac.exe<Return>
```

and manually move the executable *mbpac.exe* to the *exe* subdirectory. If the user would like to compile the parallel version of the code they must edit the *Makefile* in the *psrc* directory and compile by typing:

*gmake pmbpac.exe*<Return>

and manually move the executable *pmbpac.exe* to the *exe* subdirectory.

After compilation please check to make sure that there are five object files (corresponding to the five source code files listed above) in the *src* or *psrc* directory, and that there is an executable file named *mbpac.exe* or *pmbpac.exe* in the *exe* directory. If any of these files are missing, something went wrong during the compilation.

Step 4:

While in the *script* directory, edit the variable *\$g09*, in the *g09shuttle.pl* and the *g09-ex-shuttle.pl* scripts, so that the path indicated for the GAUSSIAN 09 program is accurate for the computer system on which MBPAC has been installed. Also edit the variable *\$molp*, in the *molpshuttle.pl* script, so that the path indicated for the MOLPRO program is accurate for the computer system on which MBPAC has been installed. The user should also change the variable *\$scratchdir* variable that appears in both of these scripts to the directory on his/her computer where s/he would like the GAUSSIAN 09 or MOLPRO program to carry out its calculations and place the scratch files.

## Chapter Six

### Description of Files in MBPAC

This chapter describes the files involved in the compiling and running of MBPAC, in particular: the source code needed to compile MBPAC, files required to run MBPAC, files created during an MBPAC run, and a script supplied to simplify the running of MBPAC.

#### 6.A. SOURCE CODE

The serial MBPAC source code is composed of five FORTRAN77 files. Subsection 7.A.1 describes each file in the serial source code, and 7.A.2 lists each subprogram in the serial version of MBPAC, in alphabetical order, and describes it. Subsection 7.A.3 lists the FORTRAN77 files for the parallel version of MBPAC that are not included in, or are different than, the serial version, and subsection 7.A.4 lists the subprograms in these files that are not included in, or are different than, those listed in subsection 7.A.2.

##### 6.A.1. SERIAL SOURCE CODE FILES

###### **common.inc**

This file contains all the parameters that limit the system and fragment sizes in the MBPAC program. The parameters defined are

MAXAT	the maximum number of atoms allowed in the system
MAXAPF	the maximum number of atoms allowed in a fragment
MAXFRAG	the maximum number of fragments allowed in the system, defined as MAXAT/MAXAPF
MAXE	the maximum number of energies printed in the output file

- MAXLIST      the maximum number of fragment combinations that can be calculated, defined by specifying MAXAT
- MAXHESS      the maximum number of Hessian elements that can be stored, defined by specifying MAXAT

**eehed.f**

This file contains the subprograms to write the header information and input summary to the output file. It also contains the subprogram to set the default variables.

**ehooks.f**

This file contains all of the subprogram for carrying out the energy calculations. It also contains the subroutines for determining the restart information for energy calculations.

**frag.f**

This file contains the subprograms for generating all possible fragment combinations for a given cluster and calculation type, getting the geometry, charge, and multiplicity for the correct fragment, and getting the correct background charges.

**freq.f**

This file contains the subprograms for calculating the frequencies once the Hessian has been computed.

**ghooks.f**

This file contains the subprograms for carrying out gradient calculations. It also contains the subroutines for determining the restart information for gradient

calculations.

### **hhooks.f**

This file contains the subprograms for carrying out a Hessian calculation. It also contains the subroutines for determining the restart information for Hessian calculations.

### **main.f**

This file contains the driver for the MBPAC program.

### **read.f**

This file contains all the subprograms for parsing the input file *eemb.inp*.

## 6.A.2. SUBPROGRAM LIST

*In this section the subprogram names are in bold. On the same line is the type of subprogram and the file it is in. The following lines list what subprograms call it and a short description of the subprogram.*

<b>atomic</b>	function	<i>ehooks.f</i>
---------------	----------	-----------------

called by: eg09

Converts the symbol of the atom type to its atomic number.

<b>calcmbce</b>	subroutine	<i>ehooks.f</i>
-----------------	------------	-----------------

called by: wg09e

Calculates the requested EE-MB-CE approximation of the system's energy.

<b>case</b>	function	<i>read.f</i>
-------------	----------	---------------

called by: rline, chkln, g09outo

Converts a string to all lower case.

<b>cfloat</b>	function	<i>read.f</i>
called by: rgeom		
Converts an integer to a double precision number.		
<b>chgcorr</b>	subroutine	<i>ehook.f</i>
called by: eg09		
Calculates the energy of interaction between QM nuclei and screened charges.		
<b>chgmlt</b>	subroutine	<i>frag.f</i>
called by: eg09, gg09, hg09		
Calculates the charge and multiplicity of a cluster of fragments.		
<b>chkcompat</b>	subroutine	<i>read.f</i>
called by: readin		
Checks to make sure that the type of calculation and the keywords specified by the user are compatible with one another and generates an error message if they are not.		
<b>chkcpop</b>	subroutine	<i>read.f</i>
called by: eg09, rjob		
Checks whether the method of population analysis selected by the user is supported by MBPAC.		
<b>chkln</b>	subroutine	<i>read.f</i>
called by: rline, rtitl, rline2		
Checks a line to see if it's a special type.		
<b>daxpy</b>	subroutine	<i>freq.f</i>
called by: dgefa, dgedi		

	Computes a constant times a vector plus a vector.	
<b>default</b>	subroutine	<i>eehed.f</i>
	called by: main	
	Sets the default variables for the program.	
<b>dgefa</b>	subroutine	<i>freq.f</i>
	called by:	
	Factors a double precision matrix by Gaussian elimination	
<b>dgedi</b>	subroutine	<i>freq.f</i>
	called by: prjfc	
	Computes the determinant and inverse of a matrix.	
<b>dscal</b>	subroutine	<i>freq.f</i>
	called by: dgefa, dgedi	
	Scales a vector by a constant.	
<b>dswap</b>	subroutine	<i>freq.f</i>
	called by: dgedi	
	Interchanges two vectors.	
<b>eehed</b>	subroutine	<i>eehed.f</i>
	called by: main	
	Writes the program header to the output file.	
<b>eg09</b>	subroutine	<i>ehooks.f</i>
	called by: ehooks	
	Writes and runs the <i>N.g09</i> input files (see section 7.C.2.).	

<b>ehooks</b>	subroutine	<i>ehooks.f</i>
called by: main		
Calls the appropriate subroutines to run the single point energy calculations.		
<b>emolp</b>	subroutine	<i>ehooks.f</i>
called by: ehooks		
Writes and runs the <i>N.inp Molpro</i> input files.		
<b>expnd</b>	subroutine	<i>freq.f</i>
called by: hhooks		
expands a triangular matrix to a square matrix		
<b>fchar</b>	subroutine	<i>read.f</i>
called by: rline, rtitl, rword, cfloat, rline2, eg09, gg09, hg09		
Finds the first nonblank character in a string, starting at a position <i>istrt</i> .		
<b>fndstrt3</b>	subroutine	<i>ehooks.f</i>
called by: ehooks, ghooks, and hhooks		
Looks for a specific <i>x_y.emb</i> file to determine whether or not the energy of fragment number <i>x</i> of type <i>y</i> must be recalculated or not. (See section 7.B.)		
<b>freqcal</b>	subroutine	<i>freq.f</i>
called by: hhooks		
Performs normal mode analysis from the Hessian matrix		
<b>fspace</b>	subroutine	<i>read.f</i>
called by: rvar, rword, cfloat, eg09, gbgq, g09oute, ghooks, gg09, g09outg, fndstrt3, hhooks, hg09, g09outh		
Finds the first blank space in a string, starting at a position <i>istrt</i> .		





<b>gg09</b>	subroutine	<i>ghooks.f</i>
called by: ghooks		
Writes <i>Gaussian 09</i> input files for a gradient calculation.		
<b>ghooks</b>	subroutine	<i>ghooks.f</i>
called by: main		
Calls the appropriate subroutines to run a gradient calculation.		
<b>ggeom</b>	subroutine	<i>frag.f</i>
called by: eg09, gg09, hg09		
Gets the geometry of the appropriate fragment of the cluster.		
<b>gmpp</b>	subroutine	<i>frag.f</i>
called by: eg09		
Gets the appropriate pseudopotentials for the background charges.		
<b>hhooks</b>	subroutine	<i>hhooks.f</i>
called by: main		
Calls the appropriate subroutines to run a Hessian/frequency calculation.		
<b>hg09</b>	subroutine	<i>hhooks.f</i>
called by: hhooks		
Writes the <i>Gaussian 09</i> input files for a Hessian calculation.		
<b>idamax</b>	function	<i>freq.f</i>
called by: dgefa		
Find the index of the element having the maximum absolute value.		
<b>molphf</b>	subroutine	<i>ehooks.f</i>
called by: ehooks		

Sets up a *Molpro* input file for a full-system Hartree–Fock calculation if the EE-MB-CE approximation has been requested.

**og09** subroutine *ohooks.f*

called by: ohooks

Writes the G09 input files for external optimization.

**prjfc** subroutine *freq.f*

called by: freqcal

Calculates projected force constant matrix.

**ratoms** subroutine *freq.f*

called by: hhooks

Reads in the atom labels and assigns masses to them for calculating mass-scaled coordinates.

**rbasis** subroutine *read.f*

called by: rjob

Reads in the basis set information from the input file.

**rbgq** subroutine *read.f*

called by: rfrag

Reads in the background charge information from the input file.

**rcore** subroutine *read.f*

called by: rjob

Reads in the core potential information from the input file.

**rcm** subroutine *read.f*

called by: rfrag



<b>rline</b>	subroutine	<i>read.f</i>
called by: readin, rgen, rfrag, rjob, rgeom, rfid, rbgq, rcm		
Finds the first non-comment, non blank line of a file, and change all characters to lower case.		
<b>rline2</b>	subroutine	<i>read.f</i>
called by: rbasis		
Finds the first non-comment, non blank line of a file.		
<b>rsp</b>	subroutine	<i>freq.f</i>
called by: freqcal		
Finds the eigenvalues and eigenvectors of a real symmetric packed matrix.		
<b>rtitl</b>	subroutine	<i>read.f</i>
called by: rgen		
Reads in the title section of the input file.		
<b>rvar</b>	function	<i>read.f</i>
called by: rgen, rfrag, rjob		
Reads a variable following a keyword.		
<b>rword</b>	subroutine	<i>read.f</i>
called by: rvar, rgeom, rfid, rbgq, rcm		
Reads in the next word on a line after the word that <i>istrt</i> is in.		
<b>setuphf</b>	subroutine	<i>ehooks.f</i>
called by: ehooks		
Sets up a <i>Gaussian</i> input file for a full-system Hartree–Fock calculation if the EE-MB-CE approximation has been requested.		

**tred3** subroutine *freq.f*

called by: rsp

Reduces a real symmetric matrix, stored as a one-dimensional array, to a symmetric matrix using orthogonal similarity transformations.

**tqlrat** subroutine *freq.f*

called by: rsp

Finds the eigenvalues of a symmetric tridiagonal matrix by the rational ql method..

**tql2** subroutine *freq.f*

called by: rsp

Finds the eigenvalues and eigenvectors of a symmetric tridiagonal matrix by the

**trbak3** subroutine *freq.f*

called by: rsp

Forms the eigenvectors of a real symmetric matrix by back transforming those of the corresponding symmetric tridiagonal matrix.

**updtjobs** subroutine *ehooks.f*

called by: ehooks

Updates the array that relates the label of each fragment to the labels of its constituent monomers; also – if the restart option has been selected – updates the array that keeps track of which fragment calculations still must be run (see Section 7.B).

**wg09e** subroutine *ehooks.f*

called by: ehooks





Contains MPI statements for initiating an MPI run.

## 6.B. FILES REQUIRED TO RUN MBPAC

Aside from the executable, there are two files that are needed to run MBPAC energy, gradient and hessian calculations: the input file (*eemb.inp*) and the shuttle script (*g09shuttle.pl* or *molpshuttle.pl*). For geometry optimizations, three additional shuttle scripts are needed: *ex\_shuttle*, *g09-ex-shuttle*, and *Gau\_External\_2*. The input file will be described in Chapter 8. The next sections will describe the shuttle scripts.

### 6.B.1. G09SHUTTLE.PL SCRIPT

MBPAC must call the electronic structure package for each fragment calculation it carries out. A perl shuttle script is included, called *g09shuttle.pl*, to call GAUSSIAN 09 each time that it is needed.

Within the script the user must specify the path to the GAUSSIAN 09 executable for their system. This is done by modifying the variables *\$g09* and *\$scratchdir* in the *g09shuttle.pl* script. The directory you specify in this variable must exist or the program will not work. The shuttle script provided generates a new subdirectory, the location of which is specified by the *\$scratchdir* variable, moves the input file to the new subdirectory, runs the GAUSSIAN 09 calculation, parses the data out of the formatted checkpoint file, returns to the home directory and then deletes the subdirectory. The name of the subdirectory is *N\_M* where *N* is a number designating which fragment combination you are calculating, and *M* is the number of fragments in the calculation. For example, directory *5\_3* would be the directory holding the fifth 3-body energy calculation.

### 6.B.2. MOLPSHUTTLE.PL SCRIPT

MBPAC must call the electronic structure package for each fragment calculation it carries out. A perl shuttle script is included, called *molpshuttle.pl*, to call MOLPRO each time that it is needed.

Within the script the user must specify the path to the MOLPRO executable for their system. This is done by modifying the variables *\$molp* and *\$scratchdir* in the *molpshuttle.pl* script. The directory you specify in this variable must exist or the program will not work. The shuttle script provided generates a new subdirectory, the location of which is specified by the *\$scratchdir* variable, moves the input file to the new subdirectory, runs the MOLPRO calculation, parses the data out of the formatted checkpoint file, returns to the home directory and then deletes the subdirectory. The name of the subdirectory is *N\_M* where *N* is a number designating which fragment combination you are calculating, and *M* is the number of fragments in the calculation. For example, directory *5\_3* would be the directory holding the fifth 3-body energy calculation.

### 6.B.3. EX\_SHUTTLE, G09-EX-SHUTTLE, AND GAU\_EXTERNAL\_2 SCRIPTS

These three scripts are used only if an external optimization is carried out using the *Gaussian 09* external optimizer. The *ex\_shuttle* script sets up the calculations needed for an external optimization. The *g09-ex-shuttle.pl* script runs *Gaussian 09* during an external optimization. The *Gau\_External\_2* script is the interface between the *Gaussian 09* external optimizer and MBPAC 2012-4A.

The *ex\_shuttle* and *g09-ex-shuttle.pl* scripts are specific to the computer that the user is working on, and must be modified if the program is to run. Within the *ex\_shuttle*

script the user must specify the number of processors to be used in the optimization, by modifying the variable *\$nproc*. A value of one will automatically use the serial version of the code; any value greater than one will use the parallel version. As with the *eemb.pl* script the *ex\_shuttle* script will currently work in parallel only for operating systems that use an *mpirun* command of

```
mpirun -np nproc <path to parallel executable file>
```

(such as the Calhoun at the Minnesota Supercomputing Institute). Systems that use a different syntax will need to modify this script, or may use only one processor and carry out a serial optimization.

Within in the *g09-ex-shuttle.pl* script the user must specify the path to the *Gaussian 09* executable and the scratch directory they would like to use by modifying the variables *\$g09* and *\$scratchdir*. These variables are the same as those that are set in the *g09shuttle.pl* script.

## 6.C. FILES CREATED DURING ALL MBPAC RUNS

Several files are created during the execution of MBPAC. This section describes the files that are created during both GAUSSIAN and MOLPRO runs.

### 6.C.1. THE OUTPUT FILE

The output file (*eemb.out*) will contain a summary of the input file, followed by the EE-MB energy, dipole moment, population analysis, gradient, Hessian, frequency, and/or optimized coordinates, depending on what type(s) of calculation(s) you requested. If you have asked for an EE-3B calculation, the EE-PA energy and gradient will also be printed. The output file will list the EE-MB energy for the level you requested and also print out the EE-MB energies for any lower-level calculations done along the way. For

example, if you choose to do an MP2 calculation, GAUSSIAN09 will also calculate the Hartree–Fock (HF) energy, and so the EE-MB energies for both MP2 and HF will be reported. Gradients, Hessians, and optimized coordinates print only for highest level of theory chosen (i.e., will print only for MP2 and not for Hartree-Fock). The EE-MB dipole moment will also be printed to the output file regardless of which type of calculation is specified. However, the dipole moment resulting from correlated methods of wave function theory such as MP2 and CCSD(T) will print out as zero with a warning message in the output file, because the dipole moments from these levels of theory are not automatically performed by GAUSSIAN 09. Additionally, dipole moments are never printed when MOLPRO is used. If the user requests one of the supported methods of population or charge analysis, then the EE-MB partial charges based on that method of charge analysis will also be printed to the output file.

#### 6.C.2 THE X\_Y.EEMB FILES

During the course of an EE-MB calculation many electronic energies are calculated using the GAUSSIAN 09 or MOLPRO electronic structure package, via the *g09shuttle* or *molpshuttle* scripts. The shuttle script also collects all of the energies, gradients, dipole moments, and Cartesian force constants from the formatted checkpoint files (*Test.FChk*) for GAUSSIAN 09 or just the energies from the MOLPRO output files (*X\_Y.out*) and places them into intermediate files with names having the form *x\_y.eemb*, where  $y = 1$  denotes a monomer,  $y = 2$  denotes a dimer, and  $y = 3$  denotes a trimer, and where  $x$  labels the specific monomer, dimer, or trimer. For example, the file containing the energy, dipole moment, and other properties of trimer #2 is called *2\_3.eemb*, and the file containing the energy and other properties of monomer #4 is named *4\_1.eemb*. If a

calculation is interrupted, for any reason, before it is completed, the *x\_y.eemb* files remain in the working directory and can be used to restart the calculation. See subsection 8.B and test run 7 for more information.

In the GAUSSIAN 09 program the self-energy of the background point charges is not subtracted from the ONIOM total energy before it is printed to the output or checkpoint file, however, and it must be removed before computing the final EE-MB energy. Thus, the program automatically performs an additional MM calculation to get the self-energy. Both the self-energy of the charges and the electronic energy of the monomer, dimer, or trimer are printed to the *x\_y.eemb* files. This is also done for the dipoles, gradients, Hessians, and partial atomic charges if they are to be computed. When all electronic structure calculations have been finished, these files are read by the MBPAC program, and the EE-MB energy is calculated.

### 6.C.3 THE FULLHF\_1.TXT FILE

The *fullhf\_1.txt* file is only created when an EE-MB-CE calculation is requested (see Section 4.D). This file is formatted in the same manner as the *x\_y.eemb* files except that it contains the Hartree–Fock energy of the entire system rather than the energy of a fragment of the system.

## 6.D. FILES CREATED ONLY DURING GAUSSIAN RUNS

### 6.D.1. THE N.G09 AND MMN.G09 FILES

When an ONIOM input file is made, it is given the name *N.g09* where *N* is a number designating the current fragment combination. These files are deleted once the energetic information has been extracted from the formatted checkpoint file (*Test.FChk*). When an MM input file is made, it is given the name *mmN.g09* where *N* is a number

designating the current fragment combination. These files are deleted once the energetic information has been extracted from the formatted checkpoint file (*mmTest.FChk*).

#### 6.D.2. THE N.OUT AND MMN.OUT FILES

These are the output files resulting from each GAUSSIAN 09 ONIOM runs of *N.g09* and MM runs of *mmN.g09*. These files are deleted once the energetic information has been extracted from the formatted checkpoint files (*Test.FChk* and *mmTest.FChk*).

#### 6.D.3. TEST.FCHK and MMTTEST.FCHK

These are the formatted checkpoint file created by GAUSSIAN 09. They are deleted once the energetic information has been extracted from it.

#### 6.D.4 ADDITIONAL FILES CREATED DURING AN EXTERNAL OPTIMIZATION

When *Gaussian 09*'s external optimizer is used for a geometry optimization two new files and one new directory are formed. The *extopt.inp* file is the *Gaussian 09* input file used for the geometry optimization. The *extopt.out* file is the *Gaussian 09* output file created during the optimization. During the course of the geometry optimization a new directory *ext/* is created. It is in this directory that all EE-MB gradient calculations are carried out. It is deleted at the end of a successful geometry optimization.

#### 6.D.5 THE MMCORRN.OUT FILE

The *mmcorrN.out* file is only created when screened embedding charges are used (see Section 4.C), where *N* is a number designating the current fragment combination. This file contains the interaction energy between the nuclei of the current fragment with the external screened point charges. This is a correction that must be added on to the

total energy that is given in the *N.out* and *Test.FChk* files (which are described in Sections 6.C.1 and 6.D.3, respectively).

## 6.E. FILES CREATED ONLY DURING MOLPRO RUNS

### 6.E.1. THE N.INP AND N.LAT FILES

When a MOLPRO input file is made, it is given the name *N.inp* where *N* is a number designating the current fragment combination. The associated lattice file, which contains the values and Cartesian coordinates of the embedding charges, is given the name *N.lat*.

### 6.E.2. THE N.OUT AND N.XML FILES

These are the output files resulting from each MOLPRO run of *N.inp*.

## Chapter Seven

### Using MBPAC

#### 7.A. THE PERL SCRIPT EEMB.PL

The input and output files described in Chapter 8 and Section 6.C.1 must have the names *eemb.inp* and *eemb.out*, which may be impractical as they will be written over every time a new MBPAC calculation is run. Additionally, the shuttle script *g09shuttle.pl* or *molpshuttle.pl* must be placed in the current working directory. For those users who would prefer not to have to copy the shuttle script every time they do a new calculation, and who would like to use more descriptive files names than *eemb.inp* and *eemb.out* a perl script called *eemb.pl* has been provided. The usage is:

*eemb.pl file\_name.inp nproc*

where *file\_name.inp* is the name of the input file you would like to run and *nproc* is the number of processors you would like to use in a parallel calculation. If you are using the serial version of the code *nproc* must be set to one. The file given by *file\_name.inp* will be copied to *eemb.inp*, and the necessary shuttle script will be copied to the current working directory. The user must edit the beginning of the *eemb.pl* file to point to the scratch directory also used in the shuttle scripts. A subdirectory of the scratch space, *file\_name*, will be created, to allow the user to run multiple MBPAC calculations in separate working directories without accidentally deleting or overwriting valuable files in the scratch space. At the end of the calculation *eemb.out* will be moved to *file\_name.out*, and *eemb.inp* will be deleted. Currently, the *eemb.pl* script is set up to work with any serial executable, however, for parallel execution the script is set up to work only for those systems that run a parallel executable via

*mpirun -np nproc <path to parallel executable file>*

If the operating system you are using uses a different command to run parallel executables (e.g., the IBM BladeCenter Linux Cluster at the University of Minnesota) the user must either not use the *eemb.pl* script, or, must edit the script to make it work with their operating system.

#### 7.B. THE RESTART OPTION

The restart option allows the user to finish partially completed EE-MB calculations. When an EE-MB calculation is carried out, the program begins by calculating all of the monomer, dimer, and trimer energies and placing the energies into files named *x\_y.eemb*, where *y* = 1 for monomers, 2 for dimers, and 3 for trimers, and where *x* is replaced by an integer that labels each specific monomer, dimer, or trimer (see section 6.C.5).

If the calculation stops, for any reason, before all monomer, dimer, or trimer calculations have been completed, the *x\_y.eemb* files corresponding to any fragment calculations that were completed successfully will remain in the working directory. These files can be read into the program to restart the calculation where it left off by setting RESTART equal to 1 (or to any number other than 0). The restart option is available for energy, gradient, and Hessian calculations only; there is no restart option available for geometry optimizations in this version.

## Chapter Eight

### Input File Structure and Explanation of Input Variables

The input file (eemb.inp) is divided into three sections namely, the \*GENERAL, \*FRAGMENT, and \*JOB sections. The \*GENERAL section must be first in the input file, and each section *must* be preceded by an asterisk, as shown above. A description of each of the three sections is given below.

There are three types of keywords used in the input file: switches, variables and lists. All keywords are case insensitive.

A switch keyword has the syntax

*Switch*

A variable keyword has the syntax

*Variable Value*

where *Variable* is the name of the keyword, and *Value* is the value you would like it to take.

A list keyword has the syntax

*List Name*

*List Items*

.  
.  
.

*End*

All lists must end with the word *end*, where capitalization of *End* is optional. It is possible for a keyword to have both a value (like a variable keyword) and a list associated with it. The syntax for such a case would be





















for helium-centered basis functions are not included in the general basis set specifications.

For expert users, we now explain the reason for the above two restrictions: The Gaussian input file written by MBPAC uses He atoms with effective core potentials to represent the screened charges, and the screened-charge centers do not have basis sets. Therefore, we cannot include He atoms in the system. Furthermore, if the user uses the basis set library (the predefined basis sets in Gaussian), then the basis set for all He atoms (including the ones that represent screened charges) is automatically assigned, which interferes with the correct operation of the screened charges option. Therefore, the users must provide the basis sets themselves and not include a basis set for He atoms. To accomplish this in MBPAC, the user needs to choose GEN for the keyword BASIS, and provide the basis sets of atoms in the system.

## Chapter Nine

### The MBPAC Test Suite

#### 9.A. INTRODUCTION TO THE TEST SUITE

The test suite has been designed to give the user a sample of the MBPAC capabilities, as well as to provide examples of input and output files. Each test run is designed to demonstrate a feature of the program and to help the user to become familiar with the MBPAC program.

In order to use the test suite, change the working directory to the *testrun* directory in the *mbpac2012-4A* directory. Within the *testrun* directory there are twenty-four subdirectories –*test1*, ..., *test24*. Each subdirectory contains all of the files necessary to complete the test run, and the corresponding subdirectory of the *testo* directory contains the test run output. If the code is installed properly one should be able to reproduce this output.

#### 9.B. THE TEST RUNS

##### 9.B.1. TEST 1

Test run 1 calls GAUSSIAN to perform a PBE/aug-cc-pVTZ EE-PA calculation on a water trimer. This test run illustrates how to run a simple EE-PA calculation with a pre-defined GAUSSIAN 09 basis set.

##### 9.B.2. TEST 2

Test run 2 calls GAUSSIAN to perform a BLYP/6-31+G(d,p) EE-3B calculation on an  $[\text{H}_3\text{O}(\text{H}_2\text{O})_3]^+$  cluster. This test run illustrates how to run an EE-3B calculation using different charges for different atoms of the same element. In

this test run the oxygen and hydrogen of the hydronium ion have different charges than the oxygen and hydrogen in the water molecules.

### 9.B.3. TEST 3

Test run 3 calls GAUSSIAN to perform a HF/MG3S EE-PA calculation on a methanol trimer. This test run illustrates how to read in a basis set from a file. Note that for methanol the MG3S basis set is identical to 6-311+G(2df,2p). Although this is a built in basis set in GAUSSIAN 09 we read it from a file to illustrate how this is done. In order for this test run to work correctly, the user will need to modify the input file to reflect the correct path to the basis set file. The basis set file used in this calculation, *mg3s.gbs*, is provided in the subdirectory *basis*. To run this test run, replace the line

```
@/home/alta/erind/mbpac2007/basis/mg3s.gbs
```

in *test3.inp* with the correct path to the *mg3s* file on your computer. To find the path go to the *basis* directory and type *pwd*:

```
erind@altix [~/mbpac2007/basis]% pwd
```

After typing *enter* the output should look like

```
/home/alta/erind/mbpac2007/basis
```

The complete path to the basis set that should be listed in your input file is *@/home/alta/erind/mbpac2007/basis/mg3s.gbs*. The @ symbol is necessary in the GAUSSIAN 09 input. See the *gen* keyword in the GAUSSIAN 09 manual for more information on how to specify a user-defined basis set.

#### 9.B.4. TEST 4

Test run 4 calls GAUSSIAN to perform a PBE1PBE EE-PA calculation on an ammonia trimer. It uses the 6-311+G(2df,2p) basis set on nitrogen and the 6-31+G(d,p) basis set on hydrogen. This test run illustrates how to use different pre-defined basis sets for different types of elements. See the *gen* keyword in the GAUSSIAN 09 manual for more information on how to specify a user-defined basis set.

#### 9.B.5. TEST 5

Test run 5 calls GAUSSIAN to perform an MP2/MIDI! EE-3B calculation on a hydrogen fluoride tetramer, where the MIDI! basis set is read in from the input file, rather than used as a pre-defined basis set. (As a pre-defined basis set it would be called MIDIX.) See the *gen* keyword in the GAUSSIAN 09 manual for more information on how to specify a user-defined basis set.

#### 9.B.6. TEST 6

Test run 6 calls GAUSSIAN to perform a PBE/aug-cc-pVTZ pairwise additive calculation on the same water trimer as in test run 1. This test run illustrates how to run a many-body calculation without the use of embedded charges, and allows you to compare the result obtained to the EE-PA calculation in test run 1.

#### 9.B.7. TEST 7

Test run 7 calls GAUSSIAN to perform a restart of test run 2. This test run illustrates how to restart a calculation that has partially completed. For this reason, some of the *x\_y.emb* files are already present in this directory, just as they would

be if test run 2 had died or been terminated before all of the monomer, dimer, or trimer calculations had been completed, and the RESTART value has been set to 1 in the *test7.inp* file.

#### 9.B.8. TEST 8

Test run calls GAUSSIAN to perform an EE-PA *mPW1PW91/aug-cc-pVDZ* single-point gradient on an  $[\text{OH}(\text{H}_2\text{O})_2]^-$  cluster. This test run illustrates how to carry out a single-point gradient calculation.

#### 9.B.9. TEST 9

Test run 9 calls GAUSSIAN to perform an EE-PA BLYP/6-31+G(d,p) frequency (Hessian) calculation on a  $\text{NH}_3(\text{H}_2\text{O})_2$  cluster. The frequencies are obtained by mass-scaling and diagonalizing the Hessian. This test run illustrates how to carry out a single-point Hessian calculation.

#### 9.B.10. TEST 10

Test run 10 calls GAUSSIAN to perform an EE-PA HF/STO-3G geometry optimization of an  $\text{HF}(\text{H}_2\text{O})_2$  cluster. This test run illustrates how to carry out an optimization using the *Gaussian 09* external optimizer. Note, if when running this test run you receive the following error “sh: line 1: Gau\_External\_2: command not found” add the current directory to the Unix/Linux environmental “PATH” variable and try rerunning the test run. You can do this by adding a command to the shell startup configuration file and then executing that file. For example, if you use the bash shell, add the line “export PATH=.:\$PATH” to the *.bashrc* file stored in your home directory and then type “source ~/.bashrc” to

make the change effective for the current session (it will automatically be effective for subsequent sessions because the shell startup configuration file executes when you start a new shell). Or, if you use the C shell, add a similar change (e.g., “set path = (\$path ./)”) to your `.cshrc` file, and type “source ~/.cshrc”.

Also, in order to run test 10 (or any geometry optimization) in parallel, you may need do the following: (1) Make sure that you have compiled both the serial and the parallel versions of the code by checking to make sure that the `mbpac2012-4A/exe` directory contains two files: `mbpac.exe` and `pmbpac.exe`. If `mbpac.exe` is missing, you must run the `configure` script in the `mbpac2012-4A` directory and type “no” in response to the prompt “Do you want to try to use the MPI version of MBPAC?”. If `pmbpac.exe` is missing, then you must run the `configure` script and answer “yes” to the prompt about the MPI version. (2) Go into the `mbpac2012-4A/script` directory and modify the `ex_shuttle` script so that the `$nproc` variable is set equal to the desired number of processors. (3) You must initiate the geometry optimization using the serial version of the code; if the `$nproc` variable in the `ex_shuttle` script is greater than 1, the script will call the parallel version of the code for the individual energy and gradient calculations to be performed on each geometry generated by the *Gaussian* optimizer. That is, even though you want to run the geometry optimization in parallel, you must type “perl eemb.pl test10.inp 1” at the command line (or in a batch script) in order for the parallel calculation to work. The fact that the `$nproc` variable is greater than 1 in the `ex_shuttle` script will ensure that parallel calculations will still be performed for the geometry optimization. This procedure should be followed if an error

message like “A message is attempting to be sent to a process whose contact information is unknown” is generated and/or if the calculation seems to hang “in limbo” for a while without generating any subdirectories in the designated scratch space.

#### 9.B.11. TEST 11

Test run 11 calls GAUSSIAN to perform an EE-3B HF/MIDI! single-point energy calculation on a 3-fragment  $[\text{OH}(\text{H}_2\text{O})_2]^-$  cluster. This test run shall give exactly the same energy (within numerical precision) with the full-QM calculation on the system. This can be used to test if the program is correct.

#### 9.B.12. TEST 12

Test run 12 calls GAUSSIAN to perform an EE-3B HF/MIDI! single-point gradient calculation on a 3-fragment  $[\text{OH}(\text{H}_2\text{O})_2]^-$  cluster. This test run shall give exactly the same result (within numerical precision) with the full-QM calculation on the system. This can be used to test if the program is correct.

#### 9.B.13. TEST 13

Test run 13 calls GAUSSIAN to perform an EE-3B HF/MIDI! single-point Hessian calculation on a 3-fragment  $[\text{OH}(\text{H}_2\text{O})_2]^-$  cluster. This test run shall give exactly the same result (within numerical precision) with the full-QM calculation on the system. This can be used to test if the program is correct.

#### 9.B.14 TEST 14

Test run 14 calls GAUSSIAN to perform a single-point EE-PA energy calculation with the PBE density functional on a  $[\text{Zn}(\text{NH}_3)_2]^{2+}$  cluster using the MG3S basis set on N and H, the B2 basis set<sup>15</sup> on Zn, and the SDD effective core potential on Zn. This test is meant to illustrate how you would request a pre-defined effective core potential for an atom and how to use the KEYWORD option. Note that when the effective core potential is defined there is a '-' in front of the Zn. This is necessary so that fragment calculations that do not include the Zn atom will run properly.

#### 9.B.15 TEST 15

Test run 15 calls GAUSSIAN to perform a PA calculation with the BLYP density functional on a  $[\text{Ca}(\text{NH}_3)_2]^{2+}$  cluster using the LANL2DZ basis set and effective core potential on  $\text{Ca}^{2+}$  and the 6-31G basis set on N and H. This test run illustrates how to use an effective core potential by listing the information for the ECP in the input file. We note that the LANL2DZ ECP is available in Gaussian 09, but we read it from the input file to illustrate how it is done.

#### 9.B.16 TEST 16

Test run 16 calls GAUSSIAN to perform an EE-3B energy calculation and an EE-3B Merz-Singh-Kollman<sup>13,14</sup> charge analysis with the M06-2X density functional on an  $(\text{H}_2\text{O})_4$  cluster using the 6-31+G basis. This test run illustrates how to use the POPULATION keyword in order to obtain EE-MB estimates of partial charges for the entire system.

## 9.B.17 TEST 17

Test run 17 calls GAUSSIAN to perform an EE-PA energy calculation and an EE-PA Mulliken<sup>6</sup> charge analysis at the Hartree-Fock level of electronic structure theory with the STO-3G basis set on an HF(H<sub>2</sub>O)<sub>4</sub> cluster. This test run illustrates that MBPAC can be run on a system containing five monomers.

## 9.B.18. TEST 18

Test run 18 calls GAUSSIAN to perform a PBE/aug-cc-pVTZ EE-PA calculation on a water trimer. This test run illustrates how to do an EE-PA calculation with point charges represented by pseudopotentials. The  $\zeta$  values of all atom types are set to  $-1.0$ . The basis set should be read in from a file. Results can be compared with test run 1 with int=grid=ultrafine.

## 9.B.19. TEST 19

Test run 19 calls GAUSSIAN to perform a PBE/aug-cc-pVTZ EE-PA calculation on a water trimer. This test run illustrates how to run an EE-PA calculation using screened charges.

## 9.B.20. TEST 20

Test run 20 calls GAUSSIAN to perform a BLYP/6-31+G(d,p) EE-3B calculation on an [(H<sub>3</sub>O)(H<sub>2</sub>O)<sub>3</sub>]<sup>+</sup> cluster. This test run illustrates how to run an EE-3B calculation using screened charges.

## 9.B.21 TEST 21

Test run 21 calls GAUSSIAN to perform a single-point EE-PA energy calculation with the PBE density functional on a [Zn(NH<sub>3</sub>)<sub>2</sub>]<sup>2+</sup> cluster using the MG3S basis

set on N and H, the B2 basis set<sup>15</sup> on Zn, and the MDF10 effective core potential<sup>18</sup> (using SDD keyword in GAUSSIAN 09) on Zn. An SDF28 effective core potential<sup>19</sup> (named ‘Stuttgart RLC ECP’ in EMSL Basis Set Exchange Library) is added to the Zn background charge in the monomer, dimer, and trimer calculations. This test run illustrates how to add effective core potentials on the background charges in calculations using the MMPSEUDO option. When the MMPSEUDO option is used, the SCRCHG option must also be used.

#### 9.B.22. TEST 22

Test run 22 calls GAUSSIAN to perform a CCSD(T)/6-31G EE-PA-CE calculation on a water trimer. This test run illustrates how to run an EE-PA-CE calculation and provides an example of the output from a CCSD(T) calculation.

#### 9.B.23. TEST 23

Test run 23 calls GAUSSIAN to perform an MP2/6-31G EE-3B-CE calculation on an  $[(\text{H}_3\text{O})(\text{H}_2\text{O})_3]^+$  cluster. This test run illustrates how to run an EE-3B-CE calculation on a positively charged system.

#### 9.B.24. TEST 24

Test run 24 calls GAUSSIAN to perform an MP2/B2 EE-PA-CE calculation on a  $[\text{Zn}(\text{NH}_3)_2]^{2+}$  cluster. This test run illustrates that the EEMBCE keyword is compatible with the CORE, MMPSEUDO, and SCRCHG keywords.

#### 9.B.25. TEST 25

Test run 25 calls MOLPRO to perform a CCSD(T)/aug-cc-pVTZ EE-PA calculation on a water trimer. This test run illustrates how to run a simple EE-PA calculation with a pre-defined MOLPRO basis set.

#### 9.B.26. TEST 26

Test run 26 calls MOLPRO to perform a CCSD(T)/jun-cc-pVTdZ EE-PA-CE calculation on a water trimer. This test run illustrates how to run an EE-PA-CE calculation with a non-pre-defined MOLPRO basis set.

#### 9.B.27. TEST 27

Test run 27 calls MOLPRO to perform a PBE/ 6-311+G(d,p) EE-3B calculation on a water tetramer. This test run illustrates how to run an EE-3B calculation with a pre-defined MOLPRO basis set.

## Chapter Ten

### Computers, Operating Systems, and Compilers on Which the Code was Tested

In each case we give the computer, operating system, and compiler (with version) on which MBPAC was tested.

#### MBPAC1.0

Computer	Operating System	FORTRAN compiler (fortran versions supported)
SGI Altix with Itanium 2 processors	SuSe Linux Enterprise Server 9.3	G77 from gcc version 3.2.3 (FORTRAN77)  Intel ifort version 8.1 (FORTRAN77/90/95)
IBM Regatta with Power 4 processors	AIX 5.2	G77 from gcc version 3.1.4 (FORTRAN77)  XL Fortran version 8.1.1 (FORTRAN77/90/95)

#### MBPAC 2007

Computer	Operating System	FORTRAN compiler (fortran versions supported)
SGI Altix with Itanium 2 processors	SuSe Linux Enterprise Server 9.3	G77 from gcc version 3.3.3 (FORTRAN77)  Intel ifort version 8.1 (FORTRAN77/90/95)
IBM Regatta with Power 4 processors	AIX 5.2	G77 from gcc version 3.1.4 (FORTRAN77)  XL Fortran version 8.1.1 (FORTRAN77/90)
IBM BladeCenter H with 2.6 GHz Opteron processors	SuSe Linux Enterprise Server 9	G77 from gcc version 3.3.3 (FORTRAN77) Mpif90 from PathScale version 2.5 (FORTRAN77/90)

## MBPAC 2007-2

Computer	Operating System	FORTRAN compiler (fortran versions supported)
SGI Altix with Itanium 2 processors	SuSe Linux Enterprise Server 9.3	G77 from gcc version 3.3.3 (FORTRAN77)  Intel ifort version 8.1 (FORTRAN77/90/95)
IBM BladeCenter H with 2.6 GHz Opteron processors	SuSe Linux Enterprise Server 9	G77 from gcc version 3.3.3 (FORTRAN77)  Mpif90 with Intel ifort version 9.1 (FORTRAN77/90)

## MBPAC 2009

Computer	Operating System	FORTRAN compiler (fortran versions supported)
SGI Calhoun with Intel Xeon processors	SuSe Linux Enterprise Server 10	Intel ifort version 11.0 (FORTRAN77/90/95)
IBM Blade Center with AMD Opteron processors	SuSe Linux Enterprise Server 9	Intel ifort version 11.0 (FORTRAN77/90/95)
SGI Altix with Itanium 2 processors	SuSe Linux Enterprise Server 9.3	Intel ifort version 8.1 (FORTRAN77/90/95)

## MBPAC 2009-2

Computer	Operating System	FORTRAN compiler (fortran versions supported)
SGI Calhoun with Intel Xeon processors	SuSe Linux Enterprise Server 10	Intel ifort version 11.0 (FORTRAN77/90/95 and FORTRAN77)  Intel ifort version 11.0 with OpenMPI (FORTRAN77/90/95 and FORTRAN77)
IBM Blade Center with AMD Opteron processors	SuSe Linux Enterprise Server 9	Intel ifort version 11.0 (FORTRAN77/90/95)
SGI Altix with Itanium 2 processors	SuSe Linux Enterprise Server 9.3	Intel ifort version 8.1 (serial and parallel) (FORTRAN77/90/95)

## MBPAC 2011

Computer	Operating System	FORTRAN compiler (fortran versions supported)
HP Linux Cluster with Intel Xeon X5560 “Nehalem-EP”-class processors	SuSe Linux Enterprise Server 11	Intel ifort version 11.1 (FORTRAN77/90/95)  Intel ifort version 11.1 with OpenMPI version 1.4.3 (FORTRAN77/90/95)
Sun Fire X4600 Linux Cluster with AMD Opteron processors (Models 8356 and 8222)	SuSe Linux Enterprise Server 10, Patchlevel 3	Intel ifort version 11.1 (FORTRAN77/90/95)

## MBPAC 2011-2

Computer	Operating System	FORTRAN compiler (fortran versions supported)
HP Linux Cluster with Intel Xeon X5560 “Nehalem-EP”-class processors	SuSe Linux Enterprise Server 11	Intel ifort version 11.1 (FORTRAN77/90/95)  Intel ifort version 11.1 with OpenMPI version 1.4.3 (FORTRAN77/90/95)
SGI Altix with Intel Xeon processors (Calhoun)	SuSe Linux Enterprise Server 10	Intel ifort version 11.0 (FORTRAN77/90/95)
SGI Altix UV 1000 with Intel Xeon X7542 “Westmere” processors (Koronis)	SuSe Linux Enterprise Server 11	Intel ifort version 11.1 with SGI MPT (Message Passing Toolkit) MPI version 2.03 (FORTRAN77/90/95)

## MBPAC 2011-3

Computer	Operating System	FORTRAN compiler (fortran versions supported)
SGI Altix with Intel Xeon processors (Calhoun)	SuSe Linux Enterprise Server 10	Intel ifort version 11.0 (FORTRAN77/90/95)
Sun Fire X4600 Linux Cluster with AMD Opteron processors (Models 8356 and 8222)	SuSe Linux Enterprise Server 10, Patchlevel 3	Intel ifort version 11.1 (FORTRAN77/90/95)

HP Linux Cluster with Intel Xeon X5560 “Nehalem-EP”-class processors	SuSe Linux Enterprise Server 11	Intel ifort version 11.1 (FORTRAN77/90/95)
--	---------------------------------	--

---

## MBPAC 2011-3A

Computer	Operating System	FORTRAN compiler (fortran versions supported)
HP Linux Cluster with Intel Xeon X5560 “Nehalem-EP”-class processors	SuSe Linux Enterprise Server 11	Intel ifort version 11.1 (FORTRAN77/90/95)

---

## MBPAC 2011-4

Computer	Operating System	FORTRAN compiler (fortran versions supported and number of processors used)
SGI Altix with Intel Xeon processors (Calhoun)	SuSe Linux Enterprise Server 10	Intel ifort version 11.0 (FORTRAN77/90/95) on 1 processor
HP Linux Cluster with Intel Xeon X5560 “Nehalem-EP”-class processors	SuSe Linux Enterprise Server 11	Intel ifort version 11.0 with OpenMPI version 1.2.8 (FORTRAN77/90/95) on 8 processors Intel ifort version 11.1 (FORTRAN77/90/95) with Platform MPI version 8.0 on 8 processors

---

## MBPAC 2011-5

Computer	Operating System	FORTRAN compiler (fortran versions supported and number of processors used)
SGI Altix with Intel Xeon processors (Calhoun)	SuSe Linux Enterprise Server 10	Intel ifort version 10.0 (FORTRAN77/90/95) on 1 processor

SGI Altix UV 1000 with Intel Xeon X7542 “Westmere” processors (Koronis)	SuSe Linux Enterprise Server 11	Intel ifort version 11.1 (FORTRAN77/90/95) on 1 processor
		Intel ifort version 11.1 (FORTRAN77/90/95) with SGI Message Passing Toolkit on 6 processors

## MBPAC 2012

Computer	Operating System	FORTRAN compiler (fortran versions supported and number of processors used)
HP Linux cluster with Intel Xeon processors (Itasca)	SuSe Linux Enterprise Server 11	Intel ifort version 11.1 (FORTRAN77/90/95) with Platform MPI on 8 processors
SGI Altix UV 1000 with Intel Xeon X7542 “Westmere” processors (Koronis)	SuSe Linux Enterprise Server 11	Intel ifort version 11.1 (FORTRAN77/90/95) on 1 processor

## MBPAC 2012-2

Computer	Operating System	FORTRAN compiler (fortran versions supported and number of processors used)
SGI Altix UV 1000 with Intel Xeon X7542 “Westmere” processors (Koronis)	SuSe Linux Enterprise Server 11	Intel ifort version 12.1 (FORTRAN77/90/95) on 1 processor
		Intel ifort version 12.1 (FORTRAN77/90/95) with SGI Message Passing Toolkit on 12 processors

## MBPAC 2012-3

Computer	Operating System	FORTRAN compiler (fortran versions supported and number of processors used)
SGI Altix UV 1000 with Intel Xeon X7542 “Westmere” processors (Koronis)	SuSe Linux Enterprise Server 11	Intel ifort version 12.1 (FORTRAN77/90/95) on 1 processor  Intel ifort version 12.1 (FORTRAN77/90/95) with SGI Message Passing Toolkit on 12 processors

## MBPAC 2012-4

Computer	Operating System	FORTRAN compiler (fortran versions supported and number of processors used)
SGI Altix with Intel Xeon processors (Calhoun)	CentOS 6.2	Intel ifort version 12.1 (FORTRAN77/90/95) on 1 processor  Intel ifort version 12.1 (FORTRAN77/90/95) with OpenMPI on 12 processors

## MBPAC 2012-4A

Computer	Operating System	FORTRAN compiler (fortran versions supported and number of processors used)
SGI Altix with Intel Xeon processors (Calhoun)	CentOS 6.2	Intel ifort version 12.1 (FORTRAN77/90/95) with OpenMPI on 12 processors
SGI Altix UV 1000 with Intel Xeon X7542 “Westmere” processors (Koronis)	SuSe Linux Enterprise Server 11	Intel ifort version 12.1 (FORTRAN77/90/95) with SGI Message Passing Toolkit on 12 processors

## Chapter Eleven

### Cited References

- (1) Dahlke, E. E.; Truhlar, D. G. *J. Phys. Chem. B* **2006**, *109*, 15677.
- (2) Frisch, M. J.; Trucks, G. W.; Schlegel, H. B.; Scuseria, G. E.; Robb, M. A.; Cheeseman, J. R.; Scalmani, G.; Barone, V.; Mennucci, B.; Petersson, G. A.; Nakatsuji, H.; Caricato, M.; Li, X.; Hratchian, H. P.; Izmaylov, A. F.; Bloino, J.; Zheng, G.; Sonnenberg, J. L.; Hada, M.; Ehara, M.; Toyota, K.; Fukuda, R.; Hasegawa, J.; Ishida, M.; Nakajima, T.; Honda, Y.; Kitao, O.; Nakai, H.; Vreven, T.; Montgomery, Jr., J. A.; Peralta, J. E.; Ogliaro, F.; Bearpark, M.; Heyd, J. J.; Brothers, E.; Kudin, K. N.; Staroverov, V. N.; Kobayashi, R.; Normand, J.; Raghavachari, K.; Rendell, A.; Burant, J. C.; Iyengar, S. S.; Tomasi, J.; Cossi, M.; Rega, N.; Millam, N. J.; Klene, M.; Knox, J. E.; Cross, J. B.; Bakken, V.; Adamo, C.; Jaramillo, J.; Gomperts, R.; Stratmann, R. E.; Yazyev, O.; Austin, A. J.; Cammi, R.; Pomelli, C.; Ochterski, J. W.; Martin, R. L.; Morokuma, K.; Zakrzewski, V. G.; Voth, G. A.; Salvador, P.; Dannenberg, J. J.; Dapprich, S.; Daniels, A. D.; Farkas, Ö.; Foresman, J. B.; Ortiz, J. V.; Cioslowski, J.; Fox, D. J. *Gaussian 09*—versions a01 and a02; Gaussian Inc.: Wallingford, CT, **2009**.
- (3) Werner, H.-J.; Knowles, P. J.; Knizia, G.; Manby, F. R.; Schütz, M.; Celani, P.; Korona, T.; Lindh, R.; Mitrushenkov, A.; Rauhut, G.; Shamasundar, K. R.; Adler, T. B.; Amos, R. D.; Bernhardsson, A.; Berning, A.; Cooper, D. L.; Deegan, M. J. O.; Dobbyn, A. J.; Eckert, F.; Goll, E.; Hampel, C.; Hesselmann, A.; Hetzer, G.; Hrenar, T.; Jansen, G.; Köppl, C.; Liu, Y.; Lloyd, A. W.; Mata, R. A.; May, A. J.; McNicholas, S. J.; Meyer, W.; Mura, M. E.; Nicklass, A.; O'Neill, D. P.; Palmieri, P.; Pflüger, K.; Pitzer, R.; Reiher, M.; Shiozaki, T.; Stoll, H.; Stone, A. J.; Tarroni, R.; Thorsteinsson, T.; Wang, M.; Wolf, A. MOLPRO, version 2010.1, a package of *ab initio* programs.
- (4) Elrod, M. J.; Saykally, R. J. *Chem. Rev.* **1994**, *94*, 1975.
- (5) Dahlke, E. E.; Truhlar, D. G. *J. Chem. Theory. Comp.* **2007**, *3*, 46.
- (6) Mulliken, R. S. *J. Chem. Phys.* **1955**, *23*, 1833.
- (7) Breneman, C. M.; Wiberg, K. B. *J. Comp. Chem.* **1990**, *11*, 361.
- (8) Xantheas, S. S. *J. Chem. Phys.* **1994**, *100*, 7523.
- (9) Löwdin, P. O. *J. Chem. Phys.* **1950**, *18*, 3654.
- (10) Baker, J. *Theor. Chim. Acta* **1985**, *68*, 221.
- (11) Thompson, J. D.; Xidos, J. D.; Sonbuchner, T. M.; Cramer, C. J.; Truhlar, D. G. *PhysChemComm* **2002**, *5*, 1170.

- (12) Storer, J. W.; Giesen, D. J.; Cramer, C. J.; Truhlar, D. G. *J. Comput.-Aided Mol. Design* **1995**, *9*, 872.
- (13) Singh, U. C.; Kollman, P. A. *J. Comp. Chem.* **1984**, *5*, 129.
- (14) Besler, B. H.; Merz Jr., K. M.; Kollman, P. A. *J. Comp. Chem.* **1990**, *11*, 431.
- (15) Amin, E. A.; Truhlar, D. G. *J. Chem. Theory. Comput.* **2008**, *4*, 75.
- (16) Wang, B.; Truhlar, D. G. *J. Chem. Theory Comput.* **2010**, *6*, 3330.
- (17) Dahlke, E. E.; Leverentz, H. R.; Truhlar, D. G. *J. Chem. Theory Comput.* **2008**, *4*, 33.
- (18) Dolg, M.; Wedig, U.; Stoll, H.; Preuss, H. *J. Chem. Phys.* **1987**, *86*, 866.
- (19) Igel-Mann, G. Ph.D. Thesis, University of Stuttgart, Stuttgart, **1987**.

## Chapter Twelve

### Bibliography

- (1) Dahlke, E. E.; Truhlar, D. G. "Electrostatically Embedded Many-Body Expansion for Large Systems, with Applications to Water Clusters" *J. Chem. Theory Comput.* **2007**, *3*, 46–53.
- (2) Dahlke, E. E.; Truhlar, D. G. "Electrostatically Embedded Many-Body Correlation Energy, with Applications to the Calculation of Accurate Second-Order Møller-Plesset Theory Energies for Large Water Clusters" *J. Chem. Theory Comput.* **2007**, *3*, 1342–1348.
- (3) Dahlke, E. E.; Truhlar, D. G. "Electrostatically Embedded Many-Body Expansion for Simulations" *J. Chem. Theory Comput.* **2008**, *4*, 1–6.
- (4) Dahlke, E. E.; Leverentz, H. R., Truhlar, D. G. "Evaluation of the Electrostatically Embedded Many-Body Expansion and the Electrostatically Embedded Many-Body Expansion of the Correlation Energy by Application to Low-Lying Water Hexamers" *J. Chem. Theory Comput.* **2008**, *4*, 33–41.
- (5) Sorkin, A.; Dahlke, E. E.; Truhlar, D. G. "Application of the electrostatically embedded many-body expansion to microsolvation of ammonia in water clusters" *J. Chem. Theory Comput.* **2008**, *4*, 683–688.
- (6) Leverentz, H. R.; Truhlar, D. G. "Electrostatically Embedded Many-Body Approximation for Systems of Water, Ammonia, and Sulfuric Acid and the Dependence of Its Performance on Embedding Charges" *J. Chem. Theory Comput.* **2009**, *5*, 1573–1584.
- (7) Hua, D.; Leverentz, H. R.; Amin, E. A.; Truhlar, D. G. "Assessment and Validation of the Electrostatically Embedded Many-Body Expansion for Metal-Ligand Bonding" *J. Chem. Theory Comput.* **2011**, *7*, 251–255.
- (8) Tempkin, J. O. B.; Leverentz, H. R.; Wang, B.; Truhlar, D. G. "The Screened Electrostatically Embedded Many-Body Method" *J. Phys. Chem. Lett.* **2011**, *2*, 2141–2144.
- (9) Kurbanov, E. K.; Leverentz, H. R.; Truhlar, D. G.; Amin, E. A. "Electrostatically Embedded Many-Body Expansion for Neutral and Charged Metalloenzyme Model Systems" *J. Chem. Theory Comput.* **2012-3**, *8*, 1–5.

## Chapter Thirteen

### Revision History

#### 13.A. VERSION 1.0

First distributed version

#### 13.B. Version 2007

1. The restart option has been added.
2. The code has been parallelized using MPI.
3. The output has been modified to print the  $V_1$ ,  $V_2$ , and  $V_3$  contributions to the energy.

#### 13.C. Version 2007-2

1. Gradient calculations have been added.
2. Hessian/frequency calculations have been added.
3. Geometry optimizations with the Gaussian 03 external optimizer have been implemented.

#### 13.D. Version 2009 (May 28, 2009)

1. In the previous version, the gradient calculations were not correct because the gradients at the background point charges were left out. Now this has been corrected in the current version. The new implementation uses the `ONIOM` method instead of the previous Gaussian keyword `charge`. Two new subroutines `getbqap` and `getbqhp` have been added, and modifications have been made to subroutines: `eg09`, `ehooks`, `g09oute`, `g09outg`, `g09outh`,

ghooks, hhooks, main, og09, wg09g, and wg09h as well as scripts g09shuttle and Gau\_External\_2. The common.inc file was also revised.

2. For the convenience of users and developers, three scripts have been added: checktestrun for comparing test run results with those distributed with the package, updatetestrun for preparing a new testrun directory (newtestrun), and updatetesto for preparing a new testo directory (newtesto) that containing the latest test run results.
3. Three test runs (testrun11, testrun12, and testrun13) have been added, which perform EE-3B energy, gradient, and Hessian calculations on the system used in testrun8; testrun8 carries out an EE-PA gradient calculation. Since EE-3B calculations for 3-fragment systems are exact, those newly added test runs can compared directly with full-QM calculations on the same system, and can be used to check if the program implementation is correct (it will not reveal all bugs, but it can be a good test to reveal many bugs).
4. The testall.pl script is revised, and a testall.pbs script is provided for submitting the test runs to the queue.
5. Many of the additions to MBPAC2009 were implemented using FORTRAN90. As a result the G77 compiler and other FORTRAN77 compilers are no longer supported.

#### 13.E. Version 2009-2 (June 25, 2009)

1. Two new keywords have been added: CORE and KEYWORDS. CORE allows the user to use effective core potentials, and KEYWORDS allows the user to specify

Gaussian 03 keywords for use in the calculation. Changes have been made to the following files: `main.f`, `eehed.f`, `ehooks.f`, `ghooks.f`, `hhooks.f`, `ohooks.f`, `read.f`, and `worker.f`.

2. Two new test runs (test runs 14 and 15) have been included to illustrate the new keywords.
3. A new script `check_all.pl` has been provided to check the results of the test runs to the output in `testo/`

### 13.F. Version 2011 (March 18, 2011)

Authors of the revisions in this version: Hannah Leverentz and Donald G. Truhlar

Complete author list of this version: Erin Dahlke, Hai Lin, Hannah Leverentz, and

Donald G. Truhlar

1. MBPAC now contains calls to *Gaussian 09* instead of *Gaussian 03*. Changes were made to source files `common.inc`, `eehed.f`, `ehooks.f`, `freq.f`, `ghooks.f`, `hhooks.f`, `main.f`, `ohooks.f`, `worker.f`, and to scripts `eemb.pl`, `ex_shuttle`, `g03-ex-shuttle.pl` (now `g09-ex-shuttle.pl`), `g03shuttle.pl` (now `g09shuttle.pl`), and `Gau_External_2`.
2. One new keyword has been added: POPULATION. This keyword allows the user to specify a method of population analysis that can be used to obtain EE-MB estimates of partial charges for the entire system. The methods of population analysis available in this version are Mulliken,<sup>6</sup> Merz-Singh-Kollman,<sup>13,14</sup> and CHelpG.<sup>7</sup> Changes have been made to the following files: `main.f`, `eehed.f`, `ehooks.f`, `ghooks.f`, `hhooks.f`, `ohooks.f`, `read.f`, `worker.f`, and

`g09shuttle.pl`. Two new subroutines, `readqgm` and `chkcpop`, were added to files `ehooks.f` and `read.f`, respectively.

3. EE-MB estimates of the wave function dipole moment are now printed at the bottom of the output file for every type of calculation, not just for gradient and hessian calculations. Changes were made to `ehooks.f` and `g09shuttle.pl`.
4. One new test run (test run 16) has been included to illustrate the new keyword.
5. A bug has been fixed that formerly prevented geometry optimizations from being performed on systems containing more than 10 atoms. Changes were made to source files `ghooks.f` and `hhooks.f` and to the script `ex_shuttle`.

### 13.G. Version 2011-2 (April 13, 2011)

Authors of the revisions in this version: Hannah Leverentz and Donald G. Truhlar

Complete author list of this version: Erin Dahlke, Hai Lin, Hannah Leverentz, and

Donald G. Truhlar

1. The default geometry optimization algorithm in *Gaussian 09* is different from that of *Gaussian 03*, and, as a result, geometry optimizations in MBPAC 2011 are much slower to converge than they are in earlier versions of MBPAC. In MBPAC 2011-2, the geometry optimization by the external *Gaussian* optimizer is forced to use the same algorithm that was used in *Gaussian 03*, thus enabling MBPAC geometry optimizations to converge more efficiently. Changes were made to source file `ohooks.f`.

2. In previous versions of MBPAC, the EE-MB optimized geometries were printed in bohr, whereas in this version the coordinates are printed in Angstroms. Changes were made to source files `common.inc` and `ohooks.f`.
3. A bug that had been introduced in version 2011 and which had prevented MBPAC from being run on systems containing five or more fragments was corrected: two new variables were added to subroutine `eg09` in `ehooks.f`.
4. Test run 17 was added to allow checking that MBPAC runs successfully on a system containing five fragments.
5. In version 2011, dipole moments were being printed in atomic units even though the output file incorrectly stated that the units were debye. Now the dipole moments are genuinely being printed to the output file in debye. Changes were made to `common.inc`, `ehooks.f`, `ghooks.f`, and `hhooks.f`.

#### 13.H. Version 2011-3 (June 21, 2011)

Authors of the revisions in this version: Bo Wang, Hannah Leverentz, and  
Donald G. Truhlar

Complete author list of this version: Erin Dahlke, Hai Lin, Hannah Leverentz, Bo  
Wang, and Donald G. Truhlar

1. The screened charge model has been added into the serial version of the code (see keyword `SCRCHG` in Section 8.C). In this version only energy calculations are available when the screened charge model is requested; gradients, Hessians, and geometry optimizations cannot be performed if the screened charge model is being used. If unscreened embedding charges are

being used, then energies, gradients, and Hessians are still available in both the serial and parallel versions of the program.

2. Test runs 18, 19, and 20 were added as examples of the screened charge model.
3. Major changes were made to subroutine `eg09` in `ehooks.f` and `g09shuttle.pl`.
4. Three new subroutines were added: subroutines `atomic` and `chgcorr` in `ehooks.f` and subroutine `gbgqs` in `frag.f`
5. In `ehooks.f`, added `nosym` keyword to the setup of normal calculations so that the full-system calculation of the energy of a trimer will be consistent with the individual monomer and dimer calculations within the trimer.
6. Increased `MAXAPF` (the maximum number of atoms allowed per fragment) from 10 to 20 in `common.inc`.

### 13.I. Version 2011-3A (July 7, 2011)

Authors of the revisions in this version: Hannah Leverentz

Complete author list of this version: Erin Dahlke, Hai Lin, Hannah Leverentz, Bo Wang, and Donald G. Truhlar

1. Increased parameter `MAXAT` (the maximum total number of atoms) in `common.inc` from 100 to 200 atoms.
2. Changed dimensions of `bgq` array in subroutine `wsum` in file `eehed.f` from `(maxat,6)` to `(maxat,5)`.

### 13.J. Version 2011-4 (July 13, 2011)

Authors of the revisions in this version: Jeremy Tempkin, Hannah Leverentz, and  
Donald G. Truhlar

Complete author list of this version: Erin Dahlke, Hai Lin, Hannah Leverentz, Bo  
Wang, Jeremy Tempkin, and Donald G. Truhlar

1. The screened charge model has been added into the parallel version of the code (see keyword SCRCHG in Section 8.C). In this version only energy calculations are available when the screened charge model is requested; gradients, Hessians, and geometry optimizations cannot be performed if the screened charge model is being used. If unscreened embedding charges are being used, then energies, gradients, Hessians, and geometry optimizations are still available in both the serial and parallel versions of the program.
2. Changes were made to subroutines eg09 and chgcorr in `ehooks.f` in both the serial and parallel versions of the code, and changes were also made to `g09shuttle.pl`. The main purpose of these changes is to allow the calculations with screened charges to be carried out in parallel.

### 13.K. Version 2011-5 (November 22, 2011)

Authors of the revisions in this version: Bo Wang, Hannah Leverentz, and  
Donald G. Truhlar

Complete author list of this version: Hannah Leverentz, Erin Dahlke, Hai Lin, Bo  
Wang, Jeremy Tempkin, and Donald G. Truhlar

1. An option to use pseudopotentials on background charges has been added into the serial and parallel versions of the code.

2. A bug is fixed to allow using SCRCHG and CORE options together.
3. The 1-body approximations to dipoles and, when appropriate, the 1-body approximations to atomic partial charges are now printed to the output file (previously only the 2- and 3-body approximations to these properties were being printed).
4. Increased parameter MAXE in `common.inc` from 10 to 15.
5. Added the EEMBCE keyword to enable the EE-MB-CE approximation to be used. In this version of the program the EEMBCE keyword can only be used with single-point energy calculations. Using the EEMBCE keyword with a gradient, hessian, geometry optimization, or population analysis is not supported; attempting to use one of these unsupported combinations of keywords will cause the program to print an error message and stop.

### 13.L. Version 2012 (February 13, 2012)

Authors of the revisions in this version: Hannah Leverentz and Donald G. Truhlar  
(the authors are grateful to Shuxia Zhang and Bo Wang for advice on the creation of this version)

Complete author list of this version: Hannah Leverentz, Erin Dahlke, Hai Lin, Bo Wang, Jeremy Tempkin, and Donald G. Truhlar

1. Rather than printing the essential information of the fragment calculations of a certain type (i.e., monomer, dimer, or trimer) to a single file called `1.eemb`, `2.eemb`, or `3.eemb`, the information pertaining to each fragment is sent to an individual file with a name having the form `x_1.eemb`, `x_2.eemb`, or `x_3.eemb`, where  $x$  labels the individual monomer, dimer, or trimer. This

change makes the parallel version of the code more efficient and also prevents problems that were arising when more than one processor needed to write large amounts of information to the same file. In order to accommodate this change, the serial and parallel versions of subroutines `fndstrt`, `fndstrtg`, and `fndstrth` have been replaced with subroutines `fndstrt2s` (for the serial version) and `fndstrt2` (for the parallel version).

2. Frequency calculations have been made more stable by using the

`opt=(maxcyc=1,nomicro, cartesian, calcfc)` keyword and options rather than the `freq` keyword in the *Gaussian* input files set up by MBPAC.

This change prevents problems that were arising when *Gaussian* was unable to convert Cartesian coordinates to internal coordinates for certain configurations of various systems.

### 13.M. Version 2012-2 (May 16, 2012)

Authors of the revisions in this version: Hannah Leverentz and Donald G. Truhlar

Complete author list of this version: Hannah Leverentz, Erin Dahlke, Hai Lin, Bo

Wang, Jeremy Tempkin, and Donald G. Truhlar

1. The parallel version has been made more efficient: all monomer, dimer, and trimer calculations (and the full Hartree–Fock calculation if the EE-MB-CE option has been selected) can now be run simultaneously during single-point energy calculations, whereas in previous versions the full Hartree–Fock portion of an EE-MB-CE calculation would have been done first, then all monomer calculations would be run in parallel, then all dimer calculations in parallel, and finally all trimer calculations in parallel. In this version, the

gradient and hessian calculations are still done using the older, less efficient algorithm.

2. Changes have been made to the serial version of the code so that the serial version uses the same data structures as the parallel version.

### 13.N. Version 2012-3 (May 23, 2012)

Authors of the revisions in this version: Hannah Leverentz and Donald G. Truhlar

Complete author list of this version: Hannah Leverentz, Erin Dahlke, Hai Lin, Bo Wang, Jeremy Tempkin, and Donald G. Truhlar

1. The parallel version has been made more efficient for gradient and hessian calculations.
2. Changes have been made to the serial version of the code so that the serial version uses the same data structures as the parallel version in the gradient and hessian calculations.
3. In previous versions, EE-MB-CE calculations could be restarted but the full-system Hartree–Fock calculation would always be rerun; in this version the EE-MB-CE calculations can be restarted and the Hartree–Fock calculation will not be rerun if it has already been done.

### 13.O. Version 2012-4 (July 11, 2012)

Authors of the revisions in this version: Helena Qi, Hannah Leverentz, and Donald G. Truhlar

Complete author list of this version: Hannah Leverentz, Erin Dahlke, Hai Lin, Bo Wang, Jeremy Tempkin, Helena Qi, and Donald G. Truhlar

1. The option to use MOLPRO instead of GAUSSIAN for energy calculations, both EE-MB and EE-MB-CE, has been added.
2. The parameter MAXAT in common.inc was increased from 200 to 400.

13.P. Version 2012-4A (September 27, 2012)

Authors of the revisions in this version: Hannah Leverentz and Donald G. Truhlar  
(the authors are grateful to Shuxia Zhang for advice on the creation of this version)

Complete author list of this version: Hannah Leverentz, Erin Dahlke, Hai Lin, Bo Wang, Jeremy Tempkin, Helena Qi, and Donald G. Truhlar

1. Different tag numbers have been assigned to the variables in “MPI\_Send” and “MPI\_Recv” calls in the parallel version of the code in order to prevent problems when large calculations are performed, and many messages must be passed between the master and worker processors.

End of manual