EMPIRE: A highly parallel semiempirical molecular orbital program: 1: Self-Consistent Field Calculations

Matthias Hennemann, and Timothy Clark*

Computer-Chemie-Centrum der Friedrich-Alexander-Universität Erlangen-Nürnberg, Nägelsbachstraße 25, 91052 Erlangen, Germany.

Abstract: EMPIRE is a massively parallel semiempirical (NDDO) molecular-orbital program designed to scale well both on single multi-core nodes (using open MP) and on large clusters (using a hybrid open MP/MPI model). The program design and performance are discussed for single self-consistent-field calculations on up to 76,800 atoms and on both single- and multi-node machines using either Windows 7 or Linux. EMPIRE currently carries out the full SCF calculation with no local approximations or other linear-scaling techniques. The single-node version is available free of charge to *bona fide* academic groups.

Introduction

We use the expression “semiempirical molecular-orbital theory” in the following only for MNDO-like NDDO-based molecular-orbital (MO) techniques [1] such as MNDO, [2] MNDO/c, [3] MNDO/d, [4-7] AM1, [8] AM1*, [9-14] RM1, [15] PM3 [16,17] and PM6. [18] These methods represent the currently accepted norm for semiempirical MO calculations, although more accurate methods that include orthogonalization corrections are also available. [19-21] The major advantage of such techniques is that they provide quite accurate one-electron properties [1] at a fraction (generally estimated to be at most 10^{-3}) of the computational cost of techniques such as density-functional theory (DFT) or *ab initio* calculations. Because of the dominant diagonalization of the Fock-matrix, semiempirical MO calculations are generally considered to scale with $O(N^3)$, where N is the number of atomic orbitals. Linear scaling can be approached quite easily by using either divide and conquer (D&C) [22-24] or localized molecular orbital (LMO) [25] techniques, although neither is suitable for very extensively conjugated systems such as those typically encountered in molecular electronic devices. [26] However, the current generation of linear scaling semiempirical MO programs is quite adequate for calculating wavefunctions for most protein-sized molecules. A practical upper limit on desktop hardware seems to lie around or slightly below 20,000 atoms. [27] Because the calculations are fast, relatively little attention has been paid to efficient parallel computation and most current programs are essentially scalar in nature. A parallel version of MNDO94 was described as long ago as 1995. [28] The D&C codes are in principle moderately parallel but parallel performance has not until now been an important issue for semiempirical MO calculations. More recently, an SCF approach for three-dimensional condensed-phase systems has been introduced that has the potential to be able to perform calculations on millions of atoms with NDDO. [29]

Current multi-core processors and modern supercomputers with tens-to-hundreds of thousands of cores now require that just about any compute-intensive software perform well in parallel. Potential modern applications of semiempirical MO-theory also require that very large systems can be calculated on appropriate hardware, for instance to simulate the electrical properties of molecular devices or aggregates.[30,31] Furthermore, many current programs such as our own VAMP [32] stem from the era

of vector computers and therefore pre-calculate many data in order to be able to process them in cpu-critical vector loops, so that memory requirements are large and memory access frequent. The former limits the sizes of the systems that can be calculated and the latter is rapidly becoming performance limiting because the speed of memory access has not kept pace with the Moore's Law increase in cpu-performance.

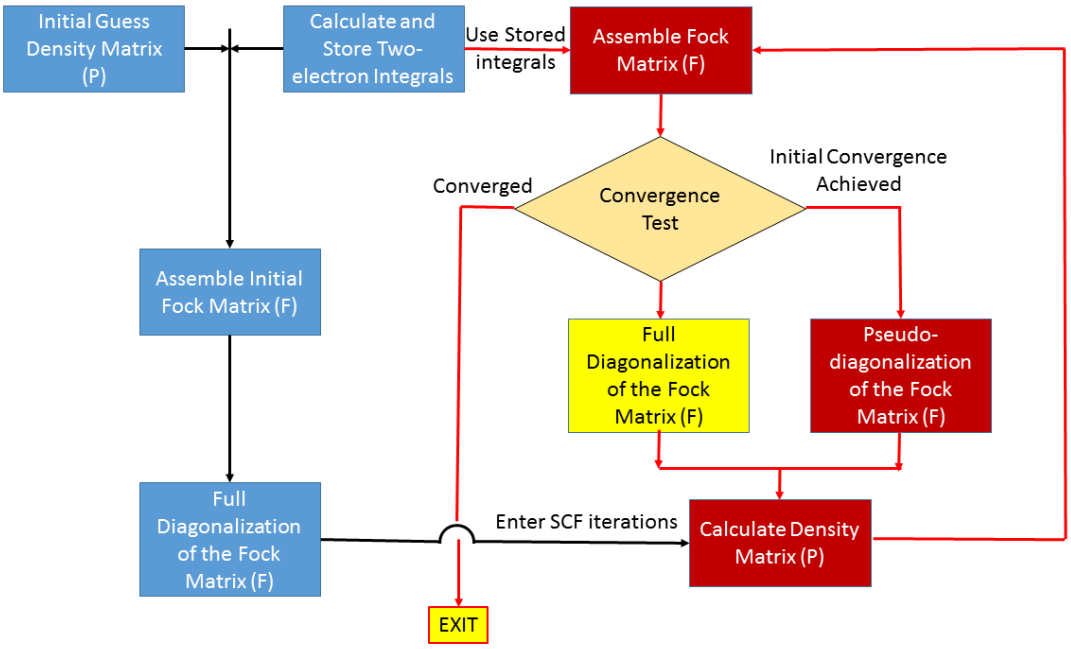
We therefore now report a new, massively parallel implementation of Hartree-Fock (RHF) NNDO-based self-consistent field (SCF) calculations in a specifically designed program, EMPIRE. Our purpose at this stage was not to write a linear scaling code, but rather to provide a reference program that performs full RHF calculations (i.e. with no approximations to improve scaling) on very large systems. Two aspects of parallel performance were important in designing the code; scalar performance equivalent to that of VAMP on one core combined with excellent scaling on single nodes (currently up to 16 cores) and effective scaling on larger numbers of processors (here up to 1,024) for very large calculations. In the second respect, our design goal was to be able to perform RHF calculations on up to 50,000 atoms on 1,000 cores. As described below, the current version of EMPIRE exceeds these specifications by a large margin.

The importance of the current reference version of the code is that it provides an excellent platform for evaluating and validating future linear scaling versions, both with respect to performance and accuracy. However, it also has the advantage that it is versatile and performs very well for small molecules of only a few hundred atoms, making it a suitable basis for the next generation of workhorse semiempirical MO programs.

Algorithms

The major bottleneck in most semiempirical MO calculations is the diagonalization of the Fock matrix in each SCF cycle. This step is only moderately parallel, even using optimized library routines, so that it must ideally be eliminated, or at least minimized if high parallel efficiency is to be achieved. The other $O(N^3)$ step in the calculation is to construct the density matrix, which is also necessary in every SCF cycle. The calculation of the core Hamiltonian (once) and the construction of the Fock matrix (every cycle) represent the remaining major calculational tasks. In a code designed for

very large systems, “infrastructure” routines, such as those designed to improve SCF convergence, can also become very important, so that these must also be considered. Scheme 1 shows the general flow chart of a conventional NDDO-SCF algorithm.



Scheme 1: Flow diagram for an SCF calculation in a conventional semiempirical molecular orbital program. The blue boxes indicate steps only carried out once at the beginning of the calculation.

Because of the memory-access limitations described above, EMPIRE is a traditional direct SCF code in which not only the two-electron integrals (always), but also (optionally) the core Hamiltonian are calculated on the fly. The core Hamiltonian can also be stored and reused for small enough systems, resulting in a moderate speedup compared with the on-the-fly calculations.

Design Considerations

The size of molecules/systems that can be calculated is limited by memory requirements. The first design consideration for EMPIRE was therefore to minimize the number of $N\times N$ (N is the number of basis functions) matrices stored. The minimum requirement for a restricted SCF calculation is to store the density and Fock matrices (both symmetrical) and MO Eigenvalues and Eigenvectors. This gives a minimum memory requirement of $2N^2+2N$ words. Perhaps paradoxically, however, EMPIRE works with full (square) versions of the symmetrical matrices, making its memory

requirement $3N^2 + N$ words, $N^2 - N$ more than if triangular storage were used for the symmetrical matrices. This apparent waste of storage is accepted as the price for being able to perform most matrix-vector and matrix-matrix operations using optimized routines form the *Intel Math Kernel Library* (MKL). [33] It also helps avoid communication in the multi-node (MPI) parallel version (see Fock matrix and Density matrix below). Using MKL gives us a speedup of approximately a factor of two compared to compiled code and contributes significantly to our goal of producing a parallel code that is also competitive with the best scalar (or vector) programs available. As a measure of the importance of this strategy, approximately 50% of the cpu-time needed by a standard EMPIRE single-point RHF calculation on 150 atoms is used in MKL library routines and approximately 60% for 750 atoms.

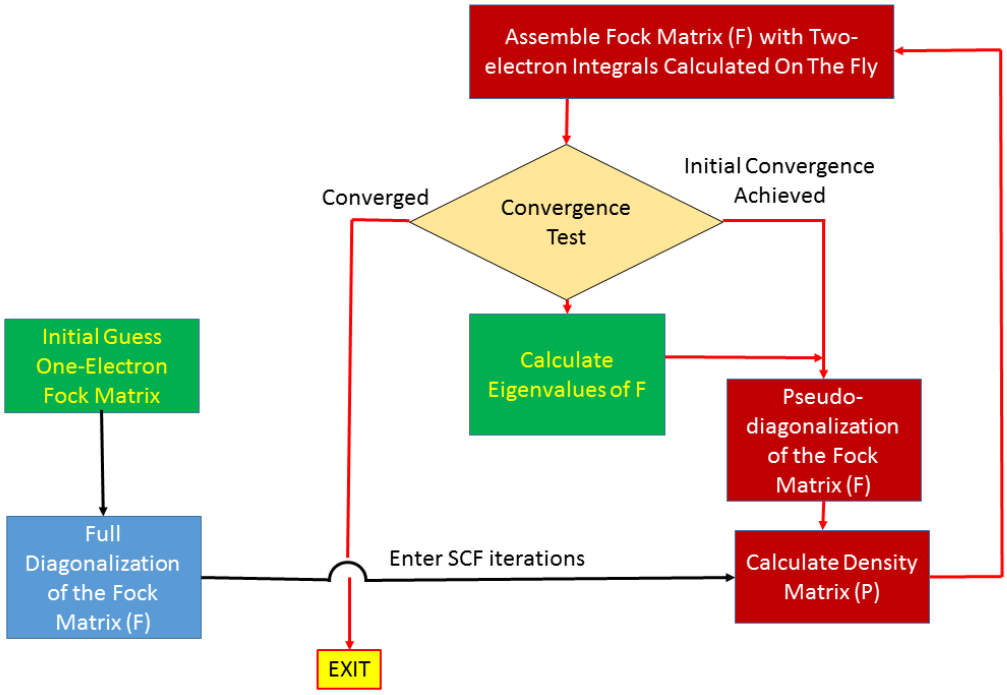
A further important factor is that many conventional convergence accelerators for SCF procedures, such as DIIS (known as Pulay's procedure in MOPAC), [34,35] require that additional copies of past density matrices be stored. Similarly, if the individual elements of the density matrix are not calculated strictly one at a time, a copy of the previous density matrix is required to calculate the convergence on the density matrix between consecutive SCF cycles, which is usually used together with the electronic energy to judge whether the SCF has converged or not. As will be described below, it was therefore necessary to find an alternative convergence criterion for the density in EMPIRE.

SCF convergence is usually ensured in standard semiempirical MO programs by one or more convergence accelerators. In VAMP and MOPAC, a two-point interpolation procedure that requires three copies of the density matrix is used as default. The IIS procedure [36] that we favor also requires three and DIIS [27,28] up to six as currently implemented in semiempirical MO programs. Other techniques such as Saunders and Hillier's level shifting [37] do not need extra copies of the density matrix, but are less effective than the interpolation/optimization schemes. However, if the initial guess is close enough to the final solution, no convergence technique is needed. We have used this approach in EMPIRE although it requires one additional full matrix diagonalization (see below).

A final design consideration for a program that performs very large calculations is how to present the results. Writing out, for instance, tens of thousands of net atomic charges or bond orders is clearly not sensible. Similarly, square matrices describing characteristics of the wavefunction for such systems are of very limited use. We have therefore resorted to lattice-based visual output, as described below.

Implementation

In the following, we describe the implementation of the individual calculational tasks in EMPIRE. The modified workflow for an SCF calculation in EMPIRE is shown in Scheme 2.



Scheme 2: Flow diagram for an SCF calculation in EMPIRE. The blue boxes indicate steps only carried out once at the beginning of the calculation and the green ones those that have been introduced for EMPIRE.

Core Hamiltonian

The core Hamiltonian can be calculated on the fly or stored, depending on the size of the system. The core Hamiltonian is generally only calculated on the fly for very large

systems, for which the extra N^2 storage requirement may become important. The core Hamiltonian is stored in horizontal stripes on the calculating nodes.

Two-electron integrals

The two-electron integrals are calculated on the fly as needed using the standard rotation method introduced for MNDO for the multipole-multipole interactions. [2]

Fock-matrix

The Fock-matrix is stored as horizontal stripes on the calculating nodes. On small single-node jobs, only half of the symmetric Fock matrix is calculated and then copied to the other half. On larger multi-node jobs, the complete stripes are calculated on each node. This doubles the amount of work done but avoids communication between the nodes. This approach is justified as the calculation of the Fock-matrix including the on-the-fly calculation of the two-electron integrals scales with N^2. Therefore, it only accounts for a minor portion of the computation time for larger calculations.

Density-matrix

The density matrix is calculated by a matrix multiplication of the occupied MO block of the Eigenvectors matrix with its transpose and by multiplying the result by a factor of two in the RHF case. By doing a full matrix multiplication we can use the MKL, which gives us a speedup of approximately a factor of two, but effectively doubles the amount of work that is done in this step. For larger calculations that are performed on multiple nodes smaller blocks of the density matrix are calculated at a time. The off-diagonal blocks are then transposed and copied to the position of their symmetric counterparts. By doing so, much of the doubling of the work is avoided.

Initial guess

The initial guess is critical for the performance of the program, especially as our options for convergence accelerators are very limited, as outlined above. The most usual initial guess used by semiempirical MO programs is to construct a diagonal density matrix in which the electrons are distributed evenly over the atomic orbitals. This is very primitive but also extremely fast to construct. Because we ideally want to rely on the standard SCF iteration scheme without an additional convergence accelerator, an initial guess

as close to the converged wavefunction as possible is required. Such guesses can be obtained from extended Hückel theory or from diagonalizing the core Hamiltonian matrix. This is not ideal for a massively parallel program as it introduces an extra full diagonalization step into the calculations. However, we have discovered that for most “normal” systems, the iterative SCF scheme converges unusually quickly if we diagonalize a matrix constructed from the two-atom blocks of the NDDO one-electron matrix combined with diagonal one-atom blocks that consist of the orbital ionization energies taken from INDO/S. [38,39] This matrix is essentially an extended Hückel matrix that uses the NDDO overlap terms and leads to MOs that are close to the final solution in almost all cases. We have therefore in the initial EMPIRE version accepted the disadvantage of an extra matrix diagonalization in order to obtain a robust and reliable SCF procedure.

Diagonalization/Pseudodiagonalization

As outlined above, the full diagonalization of the Fock matrix in every SCF-cycle represents the major bottleneck in conventional semiempirical MO calculations. Because this step scales with N^3 and because it accounts for more than half of the cpu-requirements in a conventional NDDO-SCF, it must be the major design target for a massively parallel NDDO-SCF program. Modern serial NDDO programs do not generally use full diagonalizations in every SCF-iteration, but rather switch to the so-called pseudodiagonalization procedure, in which only the Eigenvectors, but not the Eigenvalues, are updated. [40] The principles behind pseudodiagonalization have been described in detail recently, [41] but a brief description will be given here because pseudodiagonalization is essential for the EMPIRE SCF algorithm.

Pseudodiagonalization attempts to eliminate elements of the occupied-virtual block of the Fock matrix in the MO basis, $\Im$, which is given by:

$$\Im = c_o^+ F c_v \tag{1}$$

where c indicates the Eigenvector coefficients, the subscripts o and v the occupied and virtual MO blocks, respectively, and F the Fock matrix in the atomic-orbital (AO) basis. Rotation angles x_{ov} between occupied MO o and virtual one v can be estimated by simple perturbation theory to be:

$$x_{ov} = \frac{\Im_{ov}}{\left(\varepsilon_o - \varepsilon_v\right)} \tag{2}$$

where ε indicates the Eigenvalue of MO *o* or *v*. This expression reveals the reason for the fact that the pseudodiagonalization is usually turned on after a series iterations with full diagonalization of the Fock matrix: the Eigenvalues must be fairly converged because they are not updated in pseudodiagonalization iterations.

The rotation angles x_{ov} are then used for Givens (Jacobi) rotations between the occupied and virtual MO Eigenvectors:

$$\tilde{c}_o = x_{ov}c_o - \sqrt{\left(1 - x_{ov}^2\right)c_v}\,,\ \tilde{c}_v = x_{ov}c_v - \sqrt{\left(1 - x_{ov}^2\right)c_o} \tag{3}$$

The Givens rotation is a standard BLAS operation and is included in the MKL library. Although rotations over the entire MO Eigenvectors cannot be performed in parallel because of the recurrence of individual Eigenvectors in many rotations, the Eigenvectors themselves can be split into sections and these sections subjected to parallel rotations (i.e. each thread treats only a given subsection of the Eigenvectors). If the Eigenvectors are long enough, the rotations become efficiently parallel.

The potential bottleneck in this procedure if the “CFC” operation (Equation 1), which, however, can be implemented efficiently using stripes of the Fock matrix.

The missing component of a complete SCF-algorithm based on the pseudodiagonalization procedure is the calculation of the Eigenvalues ε (Equation 2), which must be carried out for those SCF iterations in which the conventional algorithm would use full diagonalization. The Eigenvalues can be calculated efficiently in parallel from the Fock matrix and the Eigenvectors. Timings for the above steps will be given below.

Convergence tests

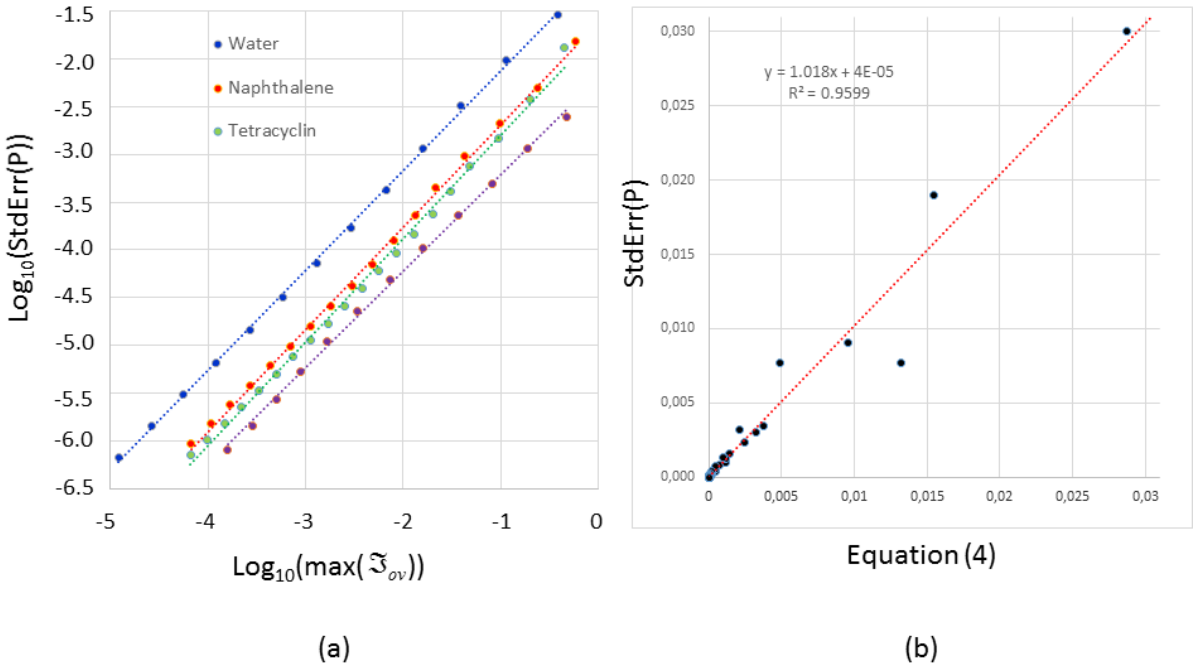
Conventional serial semiempirical MO programs test the electronic energy and the density matrix elements for convergence as the stopping criterion in the SCF-iteration process. The latter option becomes expensive inn terms of both storage requirements

and cpu-time as the size of the calculation increases. As outlined in the introduction, storing past density matrices can be every expensive for very large calculations, so that we need an alternative to the density-matrix test for standard EMPIRE calculations. The pseudodiagonalization procedure offers a useful alternative: the closer to convergence, the smaller the elements $\mathfrak{I}_{ov}$ of the occupied-virtual block of the Fock matrix in the MO basis (Equation 1). Figure 1(a) shows that the maximum element of the matrix $\mathfrak{I}_{ov}$ correlates linearly with that of the density matrix, P, so that it can be used as an equivalent measure of convergence. As $\mathfrak{I}$ is calculated in any case, it offers an excellent convergence measure in addition to the electronic energy. The relationship between $\max\left(\mathfrak{I}_{o-v}\right)$ and $StdErr\left(P_{ij}\right)$ depends on the number of orbitals in the species being calculated, as shown in Figure 1(b). Purely empirically, the RMSE of the density-matrix elements between cycles is related to the maximum element of $\mathfrak{I}_{ov}$ by:

$$StdErr\left(P\right) \approx \frac{\max\left(\mathfrak{I}_{ov}\right)}{4.5\sqrt{N_{orbs}}} \tag{4}$$

, as shown in Figure 1(b).

Figure1: (a) log/log plot of the standard error between density matrices in consecutive cycles against the maximum element of $\mathfrak{I}_{ov}$ for SCF calculations on four test molecules (one point per SCF cycle); (b) Comparison of the observed standard error between consecutive density matrices and those predicted by equation (4).



(a) (b)

However, we have neglected the weak dependence on $\sqrt{N_{orbs}}$ in the EMPIRE code in order to ensure that large molecules are well converged. Quite generally, the SCF-convergence criteria in EMPIRE are somewhat stricter than those in comparable programs.

Results and Discussion

SCF convergence

EMPIRE differs from traditional NDDO programs in its use of the extended-Hückel-like initial guess without convergence accelerators. Its only convergence accelerator is a dynamic level-shifting scheme, [30] which turns on automatically as required. We have therefore tested its SCF-convergence behavior extensively in order to assess the effectiveness of the initial guess. Quite generally, EMPIRE converges for “normal”

molecules within 50 SCF-cycles or less. This includes extensive cluster models for semiconductor particles, which might be expected to converge slowly. Difficulties are encountered as the band gap approaches zero, although compounds with a band gap down to 0.5 eV converge in less than 1,000 cycles. The model graphene compound (1800 carbon atoms, 7,326 orbitals and electrons, approximately 10×6 nm) shown in Figure 2, for instance, converges in 3,079 cycles with AM1 without dynamic level shifting. The convergence of such low band-gap materials depends, however, very strongly on the geometry and topology of the molecules.

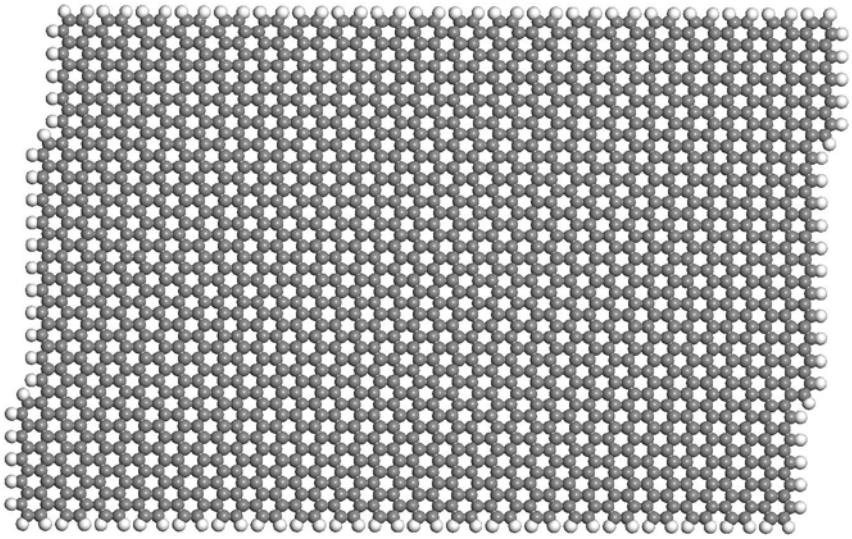


Figure 2: Model planar graphene sheet consisting of 1,800 carbon atoms saturated with terminal hydrogens.

As outlined in a separate study, EMPIRE converges very slowly for gas-phase proteins because of very slow inductive charge transfer across the molecule during the SCF iterations. [42]

Scalar performance and accuracy

EMPIRE has been implemented with consistent and up-to-date physical constants and conversion factors throughout. The exception are those constants used for the original parameterizations of the Hamiltonians, which are used as parameterized. However, the marginally different constants used in EMPIRE and the fact that other semiempirical programs may use cutoffs for some variables can lead to very small differences in calculated energies. Note also that in very specific circumstances, the localized MO SCF technique [18] used in Mopac [43] may not converge to the

variational wavefunction, so that EMPIRE also (correctly) gives lower total energies (Heats of Formation) in such cases. [35]

Figure 3: The melanine oligomers used form the benchmark calculations reported in Tables 1 and 2.

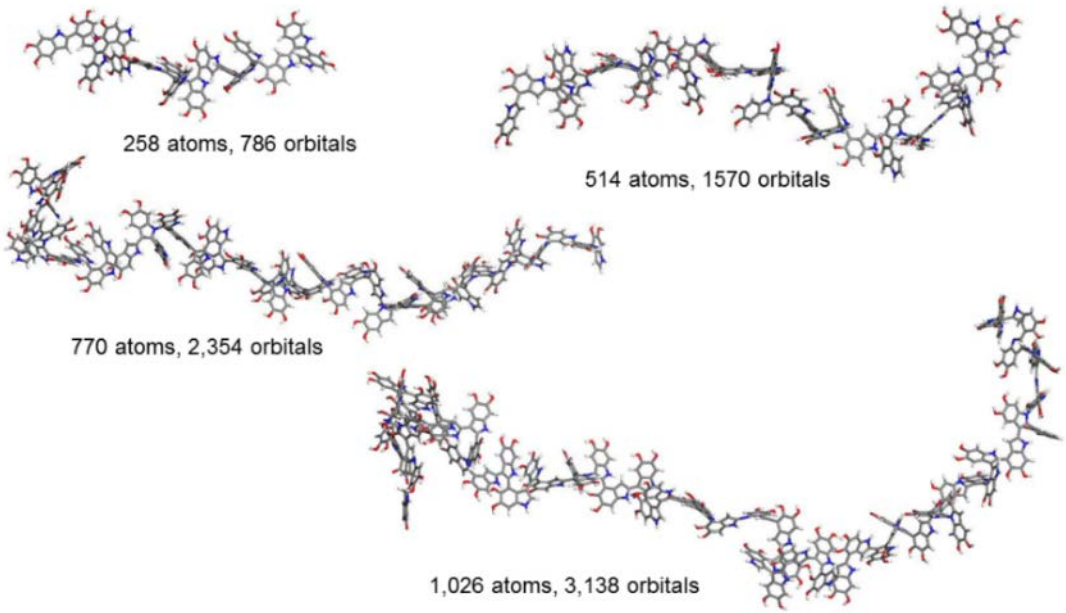


Table 1 shows the results of single-point AM1 calculations on the series of linear melanine oligomers shown in Figure 3. The results are summarized graphically in Figure 4.

Table 1: Results and cpu times for the test melanine oligomers using strictly serial versions of EMPIRE and VAMP on a single core (2.9 GHz Intel Xeon E5-2690, 64-bit Windows 7 Enterprise). The test molecules are shown in Figure 3. The timings are for single-point calculations with standard convergence criteria using the AM1 Hamiltonian. No convergence aids were used with EMPIRE. VAMP used the Badziag and Solms converger. [36] The default initial guesses for each program were used. The VAMP version used is the internal development version, which differs considerably from the commercially available version.

# atoms	# orbitals	# electrons	EMPIRE			VAMP		
			cpu seconds	SCF cycles	$\Delta H°_f$ (kcal mol−1)	cpu seconds	SCF cycles	$\Delta H°_f$ (kcal mol−1)
258	786	866	4.4	24	-442.215	3.9	20	-442.219
514	1570	1720	26.1	25	-873.743	22.1	22	-873.751
770	2354	2594	76.1	24	-1302.547	65.8	23	-1302.558
1026	3138	3458	215.6	34	-1439.638	674.4	79	-1439.640

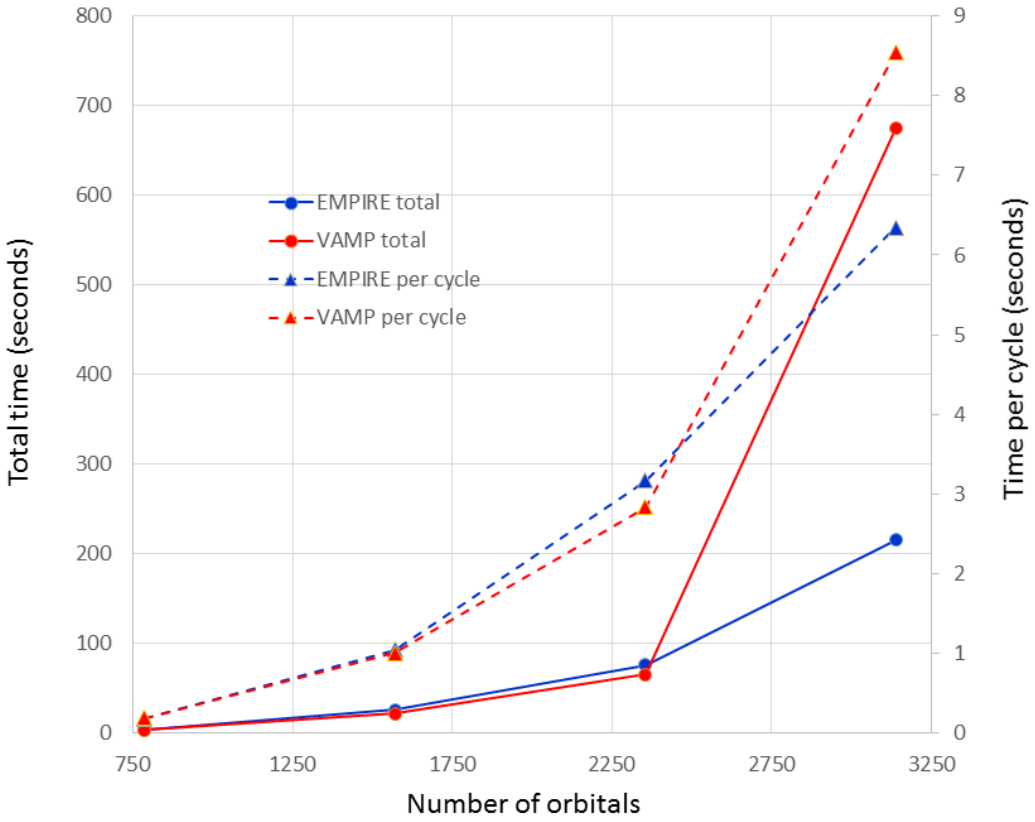


Figure 4: Total (taken from Table 1) cpu times and times per SCF cycle for the test single-point calculations defined in Table 1.

The scalar performance of the new program is surprisingly good. This is largely the result of the consequent use of MKL wherever possible and the heavily reduced memory requirement compared to VAMP, which is a classical vector program. We can conclude from the results that EMPIRE is only 10 to 20% slower than VAMP. The poor performance of VAMP for the largest test job is due to the large number of SCF cycles required with full diagonalization in order to achieve initial convergence. The more sophisticated initial guess used in EMPIRE eliminates this difficulty, so that the program performs correspondingly better. The large time per cycle found for this test with VAMP is the result of the large number of full diagonalizations.

The memory requirements are modest. Under Windows 7, EMPIRE uses a maximum of 430 MB memory for the largest melanine oligomer (1,026 atoms, 3,138 orbitals, 3,458 electrons).

Scaling

Single-node open MP

Table 2 shows the performance on a single node for the melanine test molecules shown in Figure 3. The benchmarks were conducted twice, once with Turbo Boost activated and once using a constant frequency for all runs. Turbo Boost causes runs with fewer cores to run at a higher frequency than those with more, so that it results in apparently worse scaling.

Table 2: Results and cpu times (2.8 GHz Intel Xeon E5-2680) for the test melanine oligomers using the single-node OMP version of EMPIRE as a function of the number of cores used. The test molecules are shown in Figure 3. The timings are for single-point calculations with standard convergence criteria using the AM1 Hamiltonian. No convergence aids were used.

# atoms	# AOs	# electrons	System[a]	time (seconds) for N cores						
				N=1	N=2	N=3	N=4	N=6	N=8	N=10
Without Turbo Boost										
258	786	866	Windows 7	5.4	2.9	2.1	1.7	1.3	1.1	1.0
			Linux	5.1	2.7	1.9	1.5	1.2	1.0	1.0
514	1570	1720	Windows 7	33.7	18.0	12.4	10.3	7.0	6.5	5.5
			Linux	33.2	17.8	12.3	9.3	7.0	5.4	5.0
770	2354	2594	Windows 7	96.9	50.5	35.1	27.0	19.4	15.5	14.0
			Linux	95.9	49.5	34.2	27.1	19.5	15.8	13.5
1026	3138	3458	Windows 7	272.1	139.7	96.6	74.0	52.4	41.8	37.2
			Linux	268.0	137.6	93.9	72.9	51.1	40.6	35.3
With Turbo Boost										
258	786	866	Windows 7	4.2	2.4	1.8	1.5	1.2	1.0	1.0
			Linux	4.0	2.2	1.6	1.4	1.1	0.9	0.8
514	1570	1720	Windows 7	26.9	14.5	10.4	8.3	6.4	5.3	5.1
			Linux	25.9	14.6	10.7	8.7	6.9	5.4	4.6
770	2354	2594	Windows 7	76.8	41.3	29.4	23.4	17.6	14.1	12.7
			Linux	74.3	39.8	29.3	23.1	17.1	13.7	11.7
1026	3138	3458	Windows 7	216.2	115.4	81.1	64.2	47.6	37.9	33.9
			Linux	209.3	113.9	78.7	62.1	46.1	37.1	32.0

[a] Windows 7: 64-bit Windows 7 Enterprise
Linux: openSUSE 12.3 (x86_64)

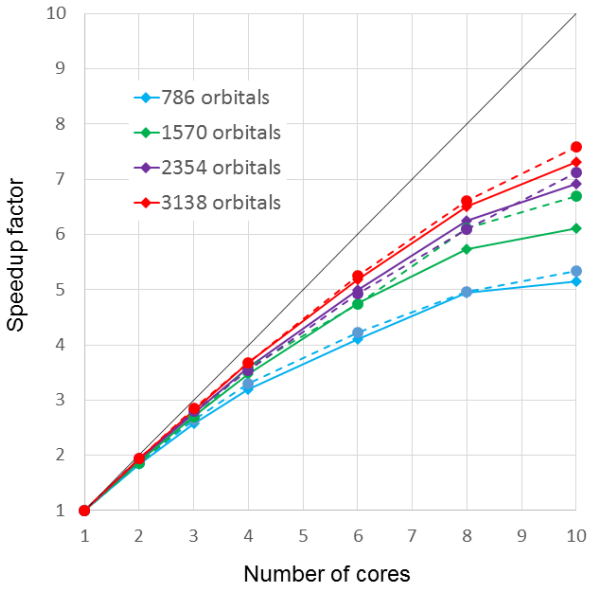


Figure 5: Scaling factors for the calculations defined in Table 2 (without Turbo Boost). The solid lines and diamonds refer to the Windows 7 calculations, circles and dashed lines to Linux.

Figures 5 and 6 show the scaling graphically, Figure 5 without Turbo boost to indicate the true scaling of the calculations, and Figure 6 with Turbo Boost, which corresponds to the normal operating conditions. The performance of the program under Windows 7 and Linux is essentially the same except that small jobs fall off faster for ten cores under Windows than Linux. The smallest test job runs at approximately 52% parallel efficiency on ten cores. This increases to 71-76% for the two largest jobs. Using Turbo Boost, the parallel efficiency on ten cores for the smallest and largest melanine oligomers is 44-48% and 60-66%, respectively. Thus, even on machines that feature Turbo Boost (which favors single-core calculations), calculation of more than 2,000 orbitals or more are 6-7× faster on ten cores than on one. As expected, the scaling for small calculations falls of gradually, so that lower numbers of cores are optimal. Even for the smallest calculation, however, the scaling up to four cores is impressive.

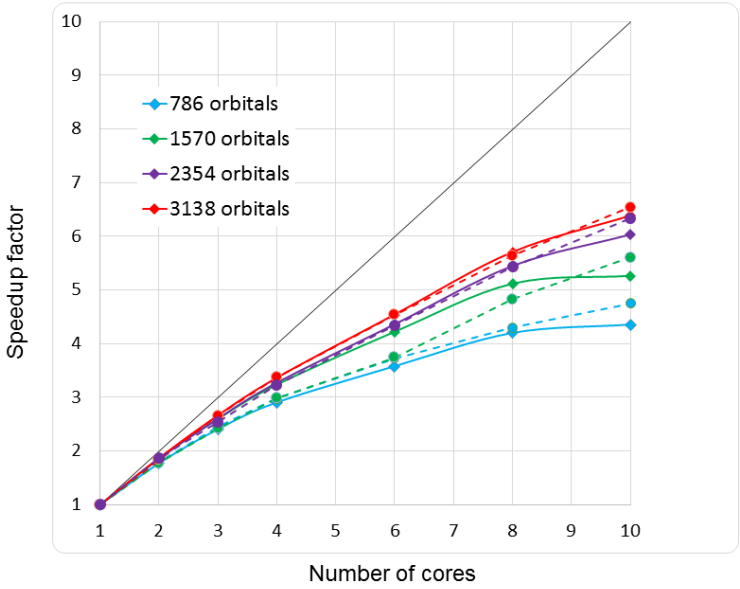


Figure 6: Scaling factors for the same calculations as shown in Figure 5, but with Turbo Boost. The solid lines and diamonds refer to the Windows 7 calculations, circles and dashed lines to Linux.

Multi node hybrid OMP/MPI

In order to test the scaling for single-point calculations on a multi-node cluster, adamantane crystals of different sizes were built with Materials Studio 6.1 [44] and used for single-point AM1 calculations on the LiMa cluster of the *Regionales Rechenzentrum Erlangen*. [45] The results are shown in Table 3. The scaling behavior is analogous to that found for the single-node version. Smaller jobs scale up to a critical number of nodes, after which the performance tends towards a plateau. If we use 75% parallel efficiency as a lower limit, the smallest calculation (11,232 atoms) can be carried out efficiently on up to 8 nodes (192 cores), increasing to 64 nodes (1,536 cores) for 37,908 atoms. The trend is approximately quadratic and reflects the fact that the calculation load per node must be adequate to offset the communication overhead.

Larger calculations than those reported in Table 3 (up to 100,000) atoms have been performed successfully with development versions of the program on larger clusters. One pleasing aspect of the calculations is that the number of SCF cycles required for convergence only rises from 23 for 11,232 atoms to 34 for 37,908, so that the initial guess is clearly also effective for large calculations.

Table 3: Results and lapsed times (2.66 GHz Intel Xeon 5650, 12 MB shared cash per chip, 24 GB RAM, Infiniband interconnect with 40 GBit/s bandwith) for adamantane crystals using the multi-node OMP/MPI version of EMPIRE as a function of the number of nodes used. The timings are for single-point calculations with standard convergence criteria using the AM1 Hamiltonian. No convergence aids were used. The crystals were built with Materials Studio 6.1 [44] and the geometries used unchanged.

# atoms	# electrons	time (seconds) for N nodes@2×12 cores							# Nodes for >75% efficiency
		N=1	N=2	N=4	N=8	N=16	N=32	N=64	
11,232	24,192	15,052	8,760	4,707	2,658	1,537	1,044	989	8
17,836	38,416				11,385	5,835	3,640	2,930	16
26,624	57,344					19,113	11,182	8,298	32
37,908	81,648						29,490	19,658	64
52,000	112,000							47,300	>64

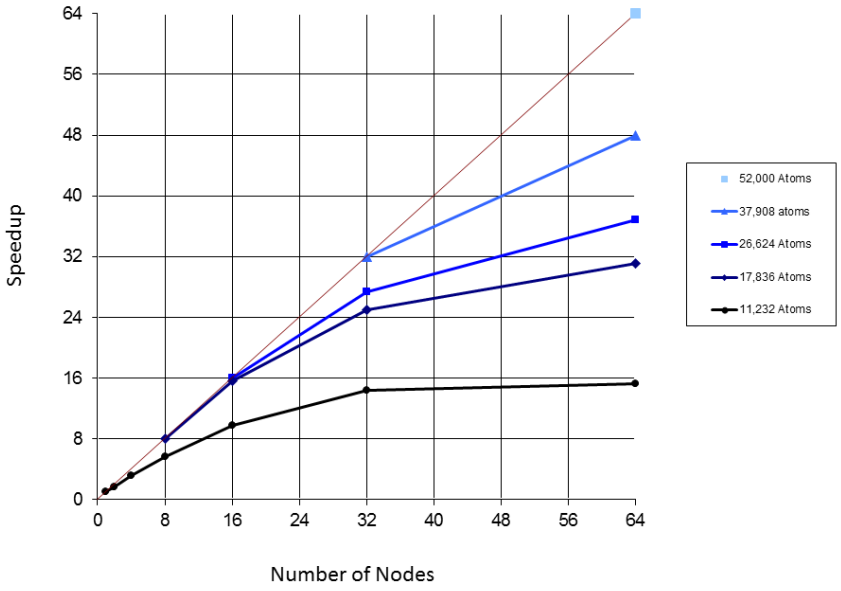


Figure 7: Scaling for adamantane crystals on the cluster defined in Table 3 using the hybrid OMP/MPI version of EMPIRE. The calculations are those described in Table 3.

Output considerations

Very large semiempirical MO calculations are now possible, but traditional output formats for quantum mechanical calculations (MOs, net atomic charges, bond orders, etc.) rapidly become too complex for molecules consisting of tens of thousands of atoms. The standard output from EMPIRE is therefore kept short and concise and emphasis is placed on extracting relevant properties from a binary file in HDF5 format

[46] that contains all the details necessary to analyze the results of the calculation *a posteori* as required. The most effective way to analyze the results for very large systems is to visualize volumetric data for local properties such as the electron density, molecular orbitals, local ionization energy, [47] local electron affinity, polarizability, electronegativity or hardness. [48,49] Such analyses are particularly useful, for instance, for characterizing the electronic properties of cluster models for crystals [50] or self-assembled monolayers (SAMs). [51] As the size of 3D property maps scales with the molecular volume, rather than the number of atoms, and the resolution can be varied to suit the application, volumetric data is usually more compact for very large systems. Apart from diagnostic information about the course of the SCF convergence, only data such as the Koopmans theorem ionization potential and electron affinity and the molecular dipole moment are provided in the output file; all other relevant properties (population analyses, electron-density maps, molecular orbitals, local property maps etc.) can be derived by post-processing the output HDF5 file, which is compact compared to a normal ASCII output file.

Figure 8, for instance, shows the molecular electrostatic potential mapped onto the 0.001 a.u. isodensity surface of the hydrogen-terminated graphene shown in Figure 2. The dominant effect of the edge structures can clearly be seen. Such maps combined with those for the local ionization energy and electron affinity provide valuable information in a compact and understandable form.

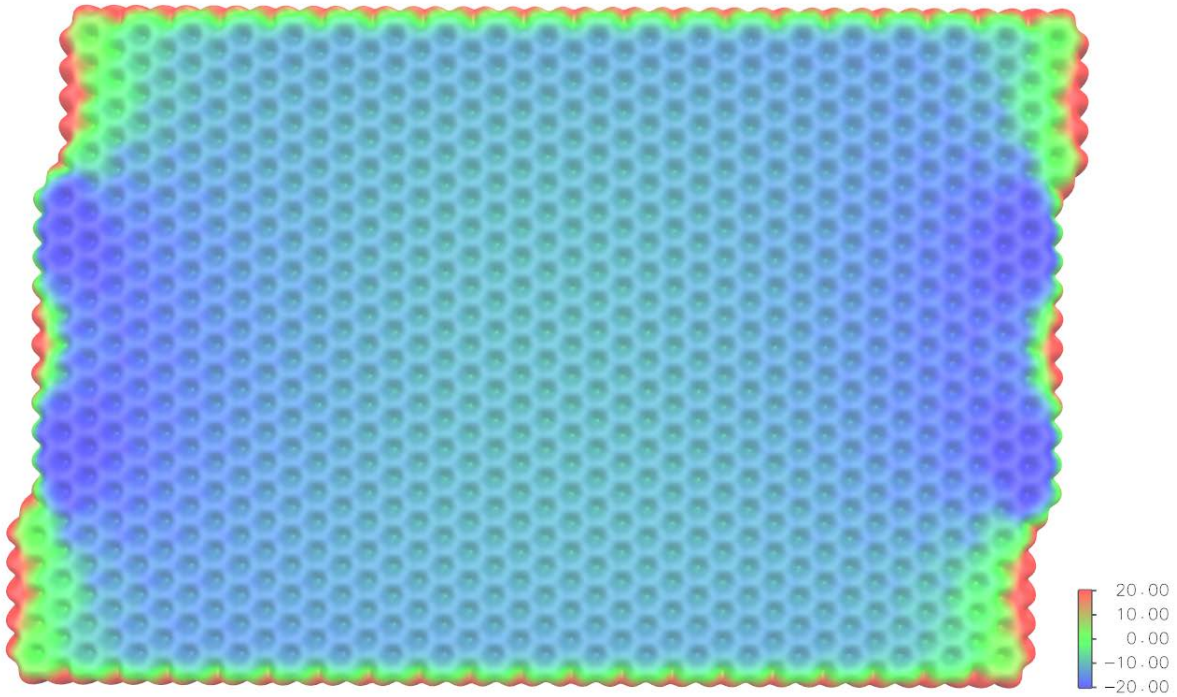


Figure 8: Molecular electrostatic potential projected onto the 0.001 a.u. isodensity surface of the hydrogen-terminated graphene shown in Figure 2. The color scale (in kcal mol^{-1}) is given on the right. The structure was built with Materials Studio 6.1 [44] and used unchanged.

Summary and conclusions

EMPIRE has proven to be very effective on both multi-core desktop machines and highly parallel clusters. The largest calculation performed so far was for 100,000 atoms on 1,024 processors (calculation not described here), but the size of systems to be calculated is only limited by the available hardware, the software is unlimited. If the core Hamiltonian is calculated on the fly, EMPIRE only requires three permanent N^2 matrices and therefore makes very economical use of memory. The program is competitive with conventional highly optimized serial programs, even on a single core, and scales well both on single nodes and on clusters of many nodes.

The single-node version of the program is available free to bona fide academic groups [52] and the program manual is available online. [53]

Acknowledgements

This work was supported by the Bavarian State Government as part of the KONWIHR II initiative and by the *Bundesministerium für Bildung und Forschung* as part of the

hpCADD project, by the *Deutche Forschungsgemeinschaft* as part of the Excellence Cluster *Engineering of Advanced Materials* and by the *Solar Technologies go Hybrid* (SolTech) initiative of the Bavarian State Government.

Literature and references

[1] MNDO-like Semiempirical Molecular Orbital Theory and its Application to Large Systems, T. Clark and J. J. P. Stewart in, *Computational Methods for Large Systems, J. J. Reimers (ed.), Wiley, Chichester, **2011**, Chapter 8 (ISBN: 978-0-470-48788-4).*

[2] Ground states of molecules. 38. The MNDO method. Approximations and parameters, Michael J. S. Dewar, Walter Thiel, *J. Am. Chem. Soc.*, 1977, *99*, 4899–4907.

[3] The MNDOC method, a correlated version of the MNDO model, W. Thiel, J. Am. Chem. Soc., 103, 1413-1420 (1981).

[4] Extension of the MNDO formalism to d-orbitals: Integral approximations and preliminary numerical results, Thiel, W.; Voityuk, A. A. *Theoret. Chim. Acta*, **1992**, *81*, 391-404, Erratum **1996**, *93*, 315.

[5] Extension of MNDO to d orbitals: Parameters and results for the halogens, Thiel, W.; Voityuk, A. A. *Int. J. Quant. Chem.*, **1994**, *44*, 807-829.

[6] Extension of MNDO to d orbitals: parameters and results for silicon, Thiel, W.; Voityuk, A. A. *J. Mol. Struct*. THEOCHEM, **1994**, *313*, 141-154.

[7] Extension of MNDO to d Orbitals: Parameters and Results for the Second-Row Elements and for the Zinc Group, Thiel, W.; Voityuk, A. A. *J. Phys. Chem.,* **1996**, *100*, 616-626.

[8] Dewar, M. J. S.; Zoebisch, E. G.; Healy, E. F.; Stewart, J. J. P. *J. Am. Chem. Soc.*, **1985**, *107*, 3902.

[9] AM1* parameters for phosphorus, sulfur and chlorine, Winget, P.; Horn, A. H. C.; Selçuki, C.; Martin, B.; Clark, T. *J. Mol. Model.*, **2004**, *9*, 408-414.

[10] AM1* parameters for aluminum, silicon, titanium and zirconium, Winget, P.; Clark, T. *J. Mol. Model.*, **2005**, *11*, 439-456.

[11] AM1* parameters for copper and zinc, Kayi, H.; Clark, T. *J. Mol. Model.*, **2007**, *13*, 965-979.

[12] AM1* parameters for bromine and iodine, Kayi, H.; Clark, T. *J. Mol. Model.*, **2009**, *15*, 295-308.

[13] AM1* parameters for vanadium and chromium, Kayi, H.; Clark, T. *J. Mol. Model.,* **2009**, *15*, 1253-1269.

[14] AM1* parameters for cobalt and nickel, Kayi, H.; Clark, T. *J. Mol. Model.,* **2010**, *16*, 29-47.

[15] RM1: A reparameterization of AM1 for H, C, N, O, P, S, F, Cl, Br, and I , Rocha, G. B.; Freire, R. O.; Simas, A. M.; Stewart, J. J. P. *J. Comput. Chem.*, **2006**, *27*, 1101-1111.

[16] Optimization of parameters for semiempirical methods I. Method, Stewart, J. J. P. *J. Comp. Chem.*, **1989**, *10*, 209-220.

[17] Optimization of parameters for semiempirical methods II. Applications, Stewart, J. J. P. J. Comp. Chem., 1989, 10, 221-264.

[18] Optimization of parameters for semiempirical methods V: modification of NDDO approximations and application to 70 elements, Stewart, J. J. P. *J. Mol. Model.*, **2007**, *13*, 1173-213.

[19] Beyond the MNDO model: Methodical considerations and numerical results, Kolb, M.; Thiel, W. *J. Comput. Chem.,* **1993**, *14*, 775-789.

[20] Orthogonalization corrections for semiempirical methods, Weber, W.; Thiel, W. *Theor. Chem. Acc.*, **2000**, *103*, 495-506.

[21] Semiempirische Verfahren mit Orthogonalisierungskorrekturen: Die OM3 Methode, Scholten, M. PhD thesis, Heinrich-Heine-Universität Düsseldorf, **2003**.

[22] Direct calculation of electron density in density-functional theory, Yang, W. *Phys. Rev. Lett*., **1991**, *66*, 1438-1441.

[23] Fast, accurate semiempirical molecular orbital calculations for macromolecules, Dixon, S. L.; Merz, K. M. Jr. *J. Chem. Phys.*, 1997, 107, 879-893.

[24] Interaction energy decomposition in protein–protein association: A quantum mechanical study of barnase–barstar complex, Ababoua, A; van der Vaart, A.; Gogonea, V.; Merz, K. M. Jr. *Biophys. Chem.*, **2007**, *125*, 221-236.

[25] Application of localized molecular orbitals to the solution of semiempirical self-consistent field equations, Stewart, J. J. P. *Int. J. Quant. Chem.,* **1996**, *58*, 133-146.

[26] J. C. Cuevas and E. Scheer, Molecular Electronics: An Introduction to Theory and Experiment (World Scientific Series in Nanotechnology and Nanoscience), World Scientific, Singapore, 2010.

[27] Application of the PM6 method to modeling proteins, Stewart, J. J. P. *J. Mol. Model.,* **2009**, *15*, 765-805.

[28] The MNDO94 Code: Parallelization of a Semiempirical Quantum-chemical Program, W. Thiel and D. G. Green, in Methods and Techniques in Computational Chemistry: METECC-95, E. Clementi and G. Corongiu, (Eds), STEF, Cagliari, **1995**, pp. 141-165.

[29] Linear Scaling Self-Consistent Field Calculations with Millions of Atoms in the Condensed Phase, Vandevondele, J., Borštnik, U., Hutter, J., J. Chem. Theor. Comput., **2012**, 8, 3565–3573.

[30] Phosphonate- and carboxylate-based self-assembled monolayers for organic devices: A theoretical study of surface binding on aluminum oxide with experimental support, T. Bauer, T. Schmaltz, T. Lenz, M. Halik, B. Meyer, and T. Clark, ACS Applied Materials & Interfaces, 2013, 5, 6073-6080 (DOI: 10.1021/am4008374).

[31] An unsymmetrical pentacene derivative with ambipolar behavior in organic thin-film transistors, R. R. Tykwinski, S. Etschel, A. Waterloo, J. T. Margraf, A. Y. Amin, F. Hampel, C. Jäger, T. Clark and M. Halik, Chem. Commun., **2013**, accepted manuscript (DOI: 10.1039/C3CC43270J).

[32] Clark, T.; Alex, A.; Beck, B.; Burckhardt, F.; Chandrasekhar, J.; Gedeck, P.; Horn, A.; Hutter, M.; Martin, B.; Rauhut, G.; Sauer, W.; Schindler, T.; Steinke, T. *VAMP 10.0*, available from Accelrys Inc., San Diego, CA , USA: Erlangen, Germany, 2007.

[33] Intel® Math Kernel Library 10.3, Intel Corporation, Santa Clara, CA, 2011; http://software.intel.com/en-us/articles/intel-mkl/ (accessed 19th April 2012)

[34] P. Pulay, Convergence acceleration of iterative sequences. The case of SCF iteration, Chem.Phys.Lett. 73 (1980) 393-398.

[35] P. Pulay, Improved SCF convergence acceleration, J. Comp. Chem. 3, 556 (1982) 556-560

[36] P. Badziag and F. Solms, An improved SCF iteration scheme, Computers & Chemistry 12 (1988) 233-236.

[37] A “Level–Shifting” method for converging closed shell Hartree–Fock wave functions, V R. Saunders and I. H. Hillier. Intern J Quantum Chem., 7 (1973) 699-705.

[38] An intermediate neglect of differential overlap technique for spectroscopy: Pyrrole and the azines, J. Ridley, M. Zerner, Theor. Chim. Acta **1973**, *32*, 111 – 134.

[39] Semiempirical Molecular Orbital Methods, M. Zerner, Reviews in Computational Chemistry, Volume 2, Eds. K. B. Lipkowitz and D. B. Boyd, VCH, New York, 313-365, (1991)

[40] Fast semiempirical calculations, Stewart, J. J. P.; Császár, P.; Pulay, P. *J. Comput. Chem.*, **1982**, *3*, 227-228.

[41] MNDO-like Semiempirical Molecular Orbital Theory and its Application to Large Systems, T. Clark and J. J. P. Stewart in, Computational Methods for Large Systems, J. J. Reimers(ed.), Wiley, Chichester, **2011**, Chapter 8 (ISBN: 978-0-470-48788-4).

[42] Self-Consistent Field Convergence for Proteins: A Comparison of Full and Localized-Molecular-Orbital Schemes, C. R. Wick, M. Hennemann, J. J. P. Stewart

and T. Clark, J. Mol. Model., 2014, 20, 2159 (6 pages) (DOI: 10.1007/s00894-014-2159-y).

[43] MOPAC2009, J. J. P. Stewart, Stewart Computational Chemistry, Colorado Springs, CO, USA, (http://www.openmopac.net/MOPAC2009brochure.pdf, accessed Mai 12th 2014).

[44] Materials Studio 6.1, Accelrys Inc., San Diego, 2013 (http://accelrys.com/products/materials-studio/index.html), accessed May 12th 2014.

[45] http://www.rrze.uni-erlangen.de/dienste/arbeiten-rechnen/hpc/systeme/lima-cluster.shtml, accessed May 12th 2014.

[46] HDF5 File Format Specification Version 2.0, (http://www.hdfgroup.org/HDF5/doc/H5.format.html, accessed May 12th 2014)

[47] Sjoberg, P.; Murray, J. S.; Brinck, T.; Politzer, P. Average Local Ionization Energies on the Molecular-Surfaces of Aromatic Systems as Guides to Chemical-Reactivity. Can. J. Chem. 1990, 68 (8), 1440−1443.

[48] Local molecular properties and their use in predicting reactivity , B. Ehresmann, B. Martin, A. H. C. Horn and T. Clark, J. Mol. Model. 2003, 9, 342-347.

[49] The Local Electron Affinity for Non-Minimal Basis Sets, T. Clark, J. Mol. Model., 2010, 16, 1231-1238 (DOI: 10.1007/s00894-009-0607-x).

[50] An unsymmetrical pentacene derivative with ambipolar behavior in organic thin-film transistors, R. R. Tykwinski, S. Etschel, A. Waterloo, J. T. Margraf, A. Y. Amin, F. Hampel, C. Jäger, T. Clark and M. Halik, Chem. Commun., 2013, 49, 6725-6727 (DOI: 10.1039/C3CC43270J).

[51] Improving the Charge Transport in Self-assembled Monolayer Field-effect Transistors - From Theory to Devices, C. M. Jäger, T. Schmaltz, M. Novak, A. Khassanov, A. Vorobiev, M. Hennemann, A. Krause, H. Dietrich, D. Zahn, A. Hirsch, M. Halik and T. Clark, J. Am. Chem. Soc., 2013, 135, 4893-4900 (DOI: 10.1021/ja401320n).

[52] http://www.ceposinsilico.de/products/empire.htm, accessed May 12th 2014.

[53] http://www.ceposinsilico.de/pdf/Empire.pdf, accessed May 12th 2014.