JCTC Journal of Chemical Theory and Computation

Article

pubs.acs.org/JCTC

Efficient Calculations of Dispersion Energies for Nanoscale Systems from Coupled Density Response Functions

Rafał Podeszwa,*[†,‡] Wojciech Cencek,[‡] and Krzysztof Szalewicz[‡]

[†]Institute of Chemistry, University of Silesia, Szkolna 9, 40-006 Katowice, Poland

[‡]Department of Physics and Astronomy, University of Delaware, Newark, Delaware 19716, United States

S *Supporting Information*

ABSTRACT: Dispersion energies computed from coupled Kohn–Sham (CKS) dynamic density–density response functions are known to be highly accurate. At the same time, the computational algorithm is of only modest complexity compared to other accurate methods of dispersion energy calculation. We present a new implementation of this algorithm that removes several computational barriers present in current implementations and enables calculations of dispersion energies for systems with more than 200 atoms using more than 5000 basis functions. The improvements were mainly achieved by reorganizing the algorithm to minimize memory and disk usage. We present applications to two systems: the buckycatcher complex with fullerene and the vancomycin complex with a diacetyl-Lys-D-Ala-D-Ala bacterial wall precursor, both calculations performed with triple-ζ-quality basis sets. Our implementation makes it possible to use *ab initio* computed dispersion energies in popular "density functional theory plus dispersion" approaches.

I. INTRODUCTION

Symmetry-adapted perturbation theory (SAPT) based on a density functional theory (DFT) description of the interacting monomers [SAPT(DFT)] has been shown to provide excellent accuracy of potential energy surfaces for a wide range of interacting systems.[1−14] The method, based on an idea proposed by Williams and Chabalowski,[15] was developed by Misquitta et al.[16−19] and independently by Hesselmann and Jansen.[20−22] SAPT(DFT) utilizes asymptotically corrected Kohn–Sham (KS) functionals to describe monomers and, in the simplest version denoted as SAPT(KS),[18] the same expressions for calculating interaction energies as the wave function based SAPT0 (the SAPT approach that neglects intramonomer correlation effects).[23] It was also found[17,22] that to get accurate dispersion and induction contributions one should use the dynamic density–density response functions also called frequency-dependent density susceptibilities (FDDS) of the monomers obtained from the time-dependent DFT (TD-DFT). The latter approach is also known as the coupled Kohn–Sham (CKS) method, and SAPT(DFT) utilizes coupled dispersion and induction energies.

The efficiency of the original implementation of SAPT-(DFT) has been greatly improved by using density fitting[17,24−26] following development of density fitting for frequency-dependent response propagators by Görling et al.[27] Whereas the implementation of ref 24 was limited to nonhybrid Hessians in TD-DFT and to dimer-centered basis sets, the implementation of refs 25 and 26 was applicable to hybrid Hessians and arbitrarily centered basis sets. Implementations of refs 24−26 contain several computational bottlenecks restricting practical calculations to systems containing less than about 100 atoms.

More recently, Hohenstein et al.[28−31] developed a computationally efficient implementation of Hartree–Fock (HF) based SAPT, both at the SAPT0 level[28,31] and beyond.[29,30] The latest improvements of SAPT0[31] allow calculations for dimers with about 200 atoms using basis sets of about 3000 functions. However, SAPT0 gives only a qualitative description of interaction energies [for comparisons of the performance of SAPT0 with SAPT(KS) and SAPT(DFT), see refs 16−19] due to the neglect of intramonomer correlation effects. The techniques of ref 31 can be directly applied to all SAPT(KS) contributions but not to the CKS induction and dispersion energies, which follow completely different algorithms that require special treatments. Whereas the CKS dispersion energies are significantly more accurate than SAPT0 or SAPT(KS) ones, they are also more difficult to compute.

We present here a completely new computational implementation of the CKS dispersion energy algorithm that removes the bottlenecks and scales well up to at least about 200 atoms and 5000 basis functions. This implementation is presented in section II. To show the capabilities of the new approach, we performed calculations for the buckycatcher complex with fullerene and the vancomycin complex with diacetyl-Lys-D-Ala-D-Ala. The results are presented in section III and compared with literature values.

II. METHODS

The idea of density fitting, crucial for the efficiency of the SAPT(DFT) method, is to approximate the (generalized) density

$$\rho_{ij}(\mathbf{r}) = \phi_i(\mathbf{r})\,\phi_j(\mathbf{r}) \tag{1}$$

where ϕ_i and ϕ_j are molecular orbitals, by

Received: March 8, 2012
Published: April 13, 2012

$$\tilde{\rho}_{ij}(\mathbf{r}) = \sum_{K=1}^{N_{\text{aux}}} \bar{D}_{ij}^{K} \chi_K(\mathbf{r}) \tag{2}$$

where χ_K, $K = 1,...,N_{\text{aux}}$, are auxiliary basis functions. The coefficients $\bar{D}_{ij}^{K}$ can be obtained from electron-repulsion weighted fits in the form[32]

$$\bar{D}_{ij}^{K} = \sum_{L} \bar{L}_{ij}^{L} [\mathbf{J}^{-1}]_{LK} \text{ or } \bar{\mathbf{D}} = \bar{\mathbf{L}}\mathbf{J}^{-1} \tag{3}$$

where

$$\bar{L}_{ij}^{L} = (ij|L) = \int \mathrm{d}\mathbf{r}_1\, \mathrm{d}\mathbf{r}_2 \frac{\phi_i(\mathbf{r}_1)\, \phi_j(\mathbf{r}_1)\, \chi_L(\mathbf{r}_2)}{|\mathbf{r}_1 - \mathbf{r}_2|} \tag{4}$$

and

$$J_{LK} = \int \mathrm{d}\mathbf{r}_1\, \mathrm{d}\mathbf{r}_2 \frac{\chi_L(\mathbf{r}_1)\, \chi_K(\mathbf{r}_2)}{|\mathbf{r}_1 - \mathbf{r}_2|} \tag{5}$$

All the symbols with the overbar here and below refer to objects of size $ov \times N_{\text{aux}}$, where o and v are the numbers of occupied and virtual orbitals, respectively, whereas the other bold Roman letters refer to $N_{\text{aux}} \times N_{\text{aux}}$ objects.

The CKS dispersion energy expressed in the auxiliary basis set is given by[17,25]

$$E_{\text{disp}}^{(2)}(\text{CKS}) = -\frac{1}{2\pi} \int_0^{\infty} \mathrm{d}\omega \sum_{KL}^{N_{\text{aux}}} \sum_{K'L'}^{N_{\text{aux}}} \tilde{C}_{KL}^{A}(i\omega) \times \tilde{C}_{K'L'}^{B}(i\omega)\, J_{KK'} J_{LL'} \tag{6}$$

where $\tilde{\mathbf{C}}^X(i\omega)$ is a coefficient matrix of the CKS FDDS at the imaginary frequency $i\omega$ (the propagator matrix) expressed in the auxiliary basis set for monomer X. The integration over ω is done by a quadrature, and as few as eight quadrature points ω_k yield a sufficient accuracy.[25] We have assumed in eq 6 that the auxiliary basis set is the same for both monomers. The calculations included in eq 6 require only $O(N_{\text{aux}}^3)$ operations and, therefore, are less expensive than the calculations of the propagator matrices, which scale as $O((ov)^2 N_{\text{aux}})$ for hybrid Hessians and $O(ovN_{\text{aux}}^2)$ for nonhybrid ones.[25] Since the two types of Hessians give similar dispersion energies (see a discussion below), the better scaling nonhybrid Hessian is a more appropriate choice for very large systems, and therefore, we have restricted the present project to this case only.

If one uses nonhybrid Hessians in calculations of FDDSs, the matrix $\tilde{\mathbf{C}}^X(i\omega)$ can be obtained from the following expression:[24,25]

$$\tilde{\mathbf{C}}(i\omega) = (\mathbf{J}^{-1})^t [\bar{\mathbf{L}}^t \mathbf{\Lambda} \bar{\mathbf{L}} + \bar{\mathbf{L}}^t \mathbf{\Lambda} \bar{\mathbf{F}} (\mathbf{J} - \bar{\mathbf{L}}^t \mathbf{\Lambda} \bar{\mathbf{F}})^{-1} \bar{\mathbf{L}}^t \mathbf{\Lambda} \bar{\mathbf{L}}] \mathbf{J}^{-1} \tag{7}$$

where

$$\bar{\mathbf{F}} = \bar{\mathbf{L}} + \bar{\mathbf{K}} \tag{8}$$

and $\bar{\mathbf{K}}$ is the so-called TD-DFT kernel matrix:[26]

$$\bar{K}_{ar}^{K} = \int \phi_a(\mathbf{r})\, \phi_r(\mathbf{r})\, \chi_K(\mathbf{r}) \frac{\delta v_{\text{xc}}}{\delta \rho}\, \mathrm{d}\mathbf{r} \tag{9}$$

In eq 9, a and r are occupied and virtual orbital indices, $\delta v_{\text{xc}}/\delta\rho$ is the exchange-correlation kernel, and $\mathbf{\Lambda}$ in eq 7 is a diagonal $ov \times ov$ matrix:

$$\Lambda_{ar,a'r'}(i\omega) = -4\delta_{ar,a'r'} \frac{(\epsilon_r - \epsilon_a)}{(\epsilon_r - \epsilon_a)^2 + \omega^2} \tag{10}$$

where ϵ_r (ϵ_a) are virtual (occupied) orbital energies (the matrix $\mathbf{\Lambda}$ corresponds to the $\boldsymbol{\lambda}\mathbf{d}$ product in ref 25). This matrix is the only one in eq 7 that depends on ω. Formula 7 is a simple modification of eq 26 in ref 25, obtained using the following identity:

$$(\mathbf{I} - \mathbf{J}^{-1} \bar{\mathbf{L}}^t \mathbf{\Lambda} \bar{\mathbf{F}})^{-1} = (\mathbf{J} - \bar{\mathbf{L}}^t \mathbf{\Lambda} \bar{\mathbf{F}})^{-1} \mathbf{J} \tag{11}$$

where $\mathbf{I}$ is the unit matrix of size $N_{\text{aux}} \times N_{\text{aux}}$.

As mentioned above, formula 7 scales as $O(ovN_{\text{aux}}^2)$, and the most expensive steps are the creation of $\bar{\mathbf{L}}^t \mathbf{\Lambda} \bar{\mathbf{L}}$ and $\bar{\mathbf{L}}^t \mathbf{\Lambda} \bar{\mathbf{F}}$ matrices. Such operations are best performed by optimized Basic Linear Algebra Subprograms (BLAS) subroutines.[33] A problem appearing for larger systems is the size of the $\bar{\mathbf{L}}$ and $\bar{\mathbf{F}}$ matrices equal to ovN_{aux}. Such matrices fitted into memory for the systems considered in ref 25; however, for dimers consisting of more than 100 atoms, these matrices may no longer fit into memory and have to be read from disk. The less memory is available, the more input/output (I/O) operations are necessary. In the implementation of the present paper, the disk I/O operations are minimized for a given memory buffer size. It is also important for the overall efficiency that the I/O operations are performed sequentially.

The elements $\bar{L}_{ij}^{K}$ are created in a two-step procedure from the three-center atomic orbital (AO) integrals $(\mu\nu|K)$,

$$(\mu j|K) = \sum_{\nu} c_{\nu j} (\mu\nu|K) \tag{12}$$

$$\bar{L}_{ij}^{K} = \sum_{\mu} c_{\mu i} (\mu j|K) \tag{13}$$

where $c_{\nu j}$ are Kohn–Sham molecular orbital coefficients (for monomer A or B) and $(\mu j|K)$ are partially transformed intermediate integrals. Although the expressions used by us for calculating $\bar{\mathbf{L}}$ are identical to those of ref 26, we have introduced several improvements in efficiency of the computer codes compared to the previous implementation. Both $(\mu\nu|K)$ and $(\mu j|K)$ are computed on-the-fly, which saves a significant number of I/O operations. Additionally, only a set of indices corresponding to a single basis set shell is stored in memory and transformed via matrix–matrix BLAS multiplication and stored on disk. This greatly reduces the memory requirements of the integral manipulations. We also changed the order of the auxiliary index in the stored matrices to be outermost, which allows for efficient sequential readings of matrices in chunks of arbitrary size in the propagator calculations.

Calculation of the $(\mu j|K)$ matrix according to eq 12 is an $O(oN_{\text{basis}}^2 N_{\text{aux}})$ process, where N_{basis} is the size of the main basis and the time needed for calculating the three-center integrals $(\mu\nu|K)$, nominally a lower-order step, is also non-negligible since efficient BLAS routines cannot be used here. The $(\mu\nu|K)$ and J_{KL} integrals are calculated using the implementation of ref 26 adopted from the GAMESS–US code.[34] Integrals corresponding to the dimer-centered basis set are calculated first. Although not used in the present study, the code can utilize monomer-centered basis sets, where the required integrals are obtained by selecting integrals corresponding to monomer A or B from the dimer basis. The savings from using monomer-centered main basis sets are modest for the CKS dispersion calculations, because the scaling depends only linearly on the size of the main basis set in the most expensive steps of eqs 7 and 9. The only step where the dependence is quadratic or higher is that of eq 12 and in the monomer DFT calculations, but these steps are relatively fast in practical calculations (see section III). Monomer-centered basis sets may become economical in calculations of exchange-dispersion corrections, where the basis set dependence is quadratic. More significant savings are possible for monomer-centered auxiliary basis sets. However, the use of such basis sets would require substantial changes in our implementation.

The kernel matrix $\bar{\mathbf{K}}$ is obtained from a numerical integration of eq 9, where, following refs 19 and 25, the exchange-correlation kernel $\delta v_{xc}/\delta\rho$ is calculated in the adiabatic local-density approximation (ALDA). The cost of calculating the matrix of eq 9 scales as $O(ovN_{aux}g)$, where g is the number of quadrature grid points and can be a substantial part of the whole calculation (see section III) due to g being a very large number.

For large systems, the ovN_{aux} size of the kernel matrix in eq 9 can become larger than the size of the available memory. Therefore, its calculation is performed in n_K batches, each batch encompassing a subrange of values of the auxiliary basis index, K, and followed by storing the calculated data on disk. Within each K-batch, the calculation of the given portion is, in turn, performed in n_i batches encompassing different subranges of the grid point index, i. For each i-batch, the values of occupied and virtual orbitals, and auxiliary basis functions at the grid points, $\phi_a(\mathbf{r}_i)$, $\phi_r(\mathbf{r}_i)$, and $\chi_K(\mathbf{r}_i)$, are calculated first, followed by the densities $\rho(\mathbf{r}_i)$ and the derivatives $\delta v_{xc}/\delta\rho(\mathbf{r}_i)$. Then, matrix $\mathbf{Z}$ of the size $goN_{aux}/(n_in_K)$ is created according to

$$z_{ia}^{K} = \phi_a(\mathbf{r}_i)\, \chi_K(\mathbf{r}_i)\, w_i \frac{\delta v_{xc}}{\delta\rho}(\mathbf{r}_i) \tag{14}$$

where w_i are the grid integration weights. Finally, the currently calculated portion of the kernel matrix is updated with the values resulting from this i-batch:

$$\bar{K}_{ar}^{K} \leftarrow \bar{K}_{ar}^{K} + \sum_i \phi_r(\mathbf{r}_i)\, z_{ia}^{K} \tag{15}$$

To minimize the memory requirements of the code, only the values of $\phi_a(\mathbf{r}_i)$ and $\phi_r(\mathbf{r}_i)$ corresponding to the current i-batch are kept in memory, which means that their calculation must be repeated for each K-batch. Unless the calculations are performed in a very limited amount of memory (which increases the number of K-batches), this does not significantly influence the efficiency, because the evaluation of the grid orbital values is much faster than the step of eq 15. The latter step scales as $O(ovN_{aux}g)$ and in practice determines the total time needed to evaluate the matrix of eq 9. It is implemented as a call to a multithreaded BLAS library, and a very satisfactory parallel scaling of the whole kernel matrix calculation has been achieved up to 16 cores tested (see below) without any parallelization of the remaining part of the kernel code. The i-batch or K-batch structure of the code can be trivially exploited to achieve good scaling on a larger number of cores.

For the kernel matrix calculations, the grid in eq 9 can be much sparser than that for the DFT energy calculations, without any significant loss of accuracy.[22] Table 1 contains results of CKS dispersion calculations for the sandwich configuration of the benzene dimer (with monomers separated by 3.85 Å), using different values of radial integration and angular integration points (N_{rad} and N_{ang}, respectively) in the kernel integral evaluation (total number of grid points for each atom is $N_{rad}N_{ang}$). Even the sparsest grid listed, with N_{rad} = 20 and N_{ang} = 26, leads to just 1.4% of error in the dispersion energy. In all of the calculations presented in this paper, we used N_{rad} = 30 and N_{ang} = 26, which for the benzene dimer gave an error of just below 1%.

Table 1. Error (in percent) of the CKS Dispersion Energy for the Benzene Dimer, Calculated Using the PBE Functional and the Dimer-Centered aug-cc-pVDZ Basis Set, As a Function of the Number of Radial (N_{rad}) and Angular (N_{ang}) Grid Points Per Atom in the Kernel Integral Calculation[a]

N_{rad}/N_{ang}	26	50	146	590
20	−1.361	− 0.681	−0.694	−0.687
30	−0.966	0.196	0.056	0.102
60	−0.828	0.049	−0.026	0.000
100	−0.812	0.042	−0.018	0.000

[a] The value of the dispersion energy converged in the number of grid points is −5.339 kcal/mol.

Since the explicit creation of $\bar{\mathbf{F}} = \bar{\mathbf{L}} + \bar{\mathbf{K}}$ would require reading and writing of large, ovN_{aux}-sized matrices, this matrix is never computed. Instead, the matrices $\bar{\mathbf{L}}^t\mathbf{\Lambda}\bar{\mathbf{L}}$ and $\bar{\mathbf{L}}^t\mathbf{\Lambda}\bar{\mathbf{K}}$ are computed and added. These matrices are only $N_{aux} \times N_{aux}$, so that $O(ovN_{aux})$ I/O operations are avoided. The algorithm for the time-limiting steps in the creation of $\bar{\mathbf{L}}^t\mathbf{\Lambda}\bar{\mathbf{L}}$ and $\bar{\mathbf{L}}^t\mathbf{\Lambda}\bar{\mathbf{K}}$ is detailed below. [The other steps in eq 7, including inversion of matrices, are fairly straightforward $N_{aux} \times N_{aux}$ matrix operations that scale as $O(N_{aux}^3)$ and require $O(N_{aux}^2)$ I/O operations.]

1. Create the $\bar{\mathbf{K}}$ and $\bar{\mathbf{L}}$ matrices as described above and write them to disk in records of size ov for each auxiliary index (i.e., the auxiliary index is the outermost one).
2. Allocate a buffer $\mathbf{W}$ of size $ov \times n_{buf}$ occupying a large fraction of the available memory. The size of the buffer will determine the I/O count of the calculations. This buffer will contain a submatrix of $\bar{\mathbf{L}}^t$ (in a nontransposed form) needed for the $\bar{\mathbf{L}}^t\mathbf{\Lambda}\bar{\mathbf{L}}$ product.
3. Read an $ov \times n_{buf}$ stripe of $\bar{\mathbf{L}}$ into $\mathbf{W}$.
4. Allocate a buffer $\mathbf{v}$ of size $ov \times n_{slice}$, which should use most of the remaining available memory. The buffer will store a submatrix of $\bar{\mathbf{L}}$ used in the $\mathbf{\Lambda}\bar{\mathbf{L}}$ product.
5. Read an $ov \times n_{slice}$ stripe of $\bar{\mathbf{L}}$ into $\mathbf{v}$.
6. For each row of $\mathbf{v}$, scale it by $\Lambda(i\omega_k) = -4(\epsilon_r - \epsilon_a)/[(\epsilon_r - \epsilon_a)^2 + \omega_k{}^2]$ (cf. eq 10) for k = 1 and by $\Lambda(i\omega_k)/\Lambda(i\omega_{k-1})$ for $k > 1$, where k = 1, ..., N_{quad} and N_{quad} is the number of quadrature points in the integration over ω. This step corresponds to the multiplication by $\mathbf{\Lambda}$.
7. Perform the $\mathbf{W}^t\mathbf{v}$ multiplication with BLAS and write the resulting $n_{buf} \times n_{slice}$ submatrix of $\bar{\mathbf{L}}^t\mathbf{\Lambda}\bar{\mathbf{L}}$ to disk.
8. Repeat steps 6 and 7 for all ω_k.
9. Repeat steps 5−8 until all stripes $\mathbf{\Lambda}\bar{\mathbf{L}}$ are processed.
10. Perform steps 5−9 replacing $\mathbf{\Lambda}\bar{\mathbf{L}}$ with $\mathbf{\Lambda}\bar{\mathbf{K}}$. This process will create the $\bar{\mathbf{L}}^t\mathbf{\Lambda}\bar{\mathbf{K}}$ submatrices.
11. If $\bar{\mathbf{L}}^t$ was not completely processed, go to step 3.

After performing the steps listed above, matrices $\bar{\mathbf{L}}^t\mathbf{\Lambda}\bar{\mathbf{L}}$ and $\bar{\mathbf{L}}^t\mathbf{\Lambda}\bar{\mathbf{K}}$ will be ready on disk. The value of n_{slice} determines only the size of the matrix multiplication slice. For the best BLAS efficiency, this value should be as large as possible, but the size of n_{buf} is more important since $\lceil(2N_{aux}/n_{buf})\rceil ovN_{aux}$ words (where $\lceil x\rceil$ denotes the ceiling function of x) must be read from disk and step 6 is repeated $\lceil(N_{aux}/n_{buf})\rceil$ times for each of the matrices $\bar{\mathbf{L}}$ and $\bar{\mathbf{K}}$. For $n_{buf} = N_{aux}$, all matrices will be read only once and step 6 performed only once for each element of $\bar{\mathbf{L}}$ and $\bar{\mathbf{K}}$. Multiplication by $\Lambda(i\omega_k)/\Lambda(\omega_{k-1})$ in step 6 enables reutilization of **v**. Otherwise, **v** would have to be read from disk for each ω_k. The indirect scaling in this step raises some potential issues of numerical round-off error accumulating in the **v** matrix. However, with only a few ω_k values in the quadrature, the round-off errors are completely negligible, and this approach saves significant I/O time.

The major performance improvement of our implementation of the CKS dispersion energy calculations that enables its use for nanosized systems stems from our efficient use of memory and the minimization of the I/O operations. The old implementation stored some $O(N^3)$ objects in memory and some $O(N^4)$ on disk. For the systems calculated in the present paper, it would require hundreds of GB of memory and tens of TB of disk space. In contrast, the current implementation stores only $O(N^2)$ objects in memory and $O(N^3)$ on disk. However, several other improvements in various parts of the code contributed to the overall increase of performance. In fact, virtually all the steps in the algorithm have been significantly improved, including monomer calculations, kernel integrals, calculations of three-center integrals, and integral transformation. The present code is therefore faster also for systems that can be calculated with the code of ref 26.

Although the current implementation of the CKS dispersion energy is limited to nonhybrid Hessians, the accuracy of the dispersion energy for nonhybrid functionals is only slightly worse than for the hybrid ones. First, note that many TD-DFT calculations use the ALDA kernel; i.e., even if a generalized gradient approximation (GGA) exchange-correlation potential is used to obtain the KS orbitals and orbital energies, the potential v_{xc} in eq 9 is the LDA one (computed with GGA orbitals). It was shown in ref 19 that such an approximation leads to a small error, below 1%, in calculations of dispersion energies. This indicates that the dispersion energy is not very sensitive to the exact form of the Hessian. Indeed, for the helium dimer, where we know the exact dispersion energy in a given basis set, a nonhybrid HCTH407 functional gives 3.4% error, whereas two hybrid functionals, B97-2 and PBE0, give 1.5% and 2.9% errors, respectively, all results in the ALDA approximation and with consistent use of hybrid/nonhybrid orbitals, orbital energies, and Hessians (ref 19). For other investigated systems, exact dispersion energies are unknown, but differences between the predictions of hybrid and nonhybrid functionals are small. For example, for the water dimer at equilibrium, the HCTH407 dispersion energy is −2.42 kcal/mol, whereas both hybrid functionals give −2.40 kcal/mol (ref 19). One can also use hybrid orbitals and orbital energies in nonhybrid Hessians, but we found such strategy not to improve the dispersion energies. For some other SAPT corrections, the differences between hybrid and nonhybrid functionals are larger, and the former values are usually better, so these terms should preferrably be computed using the hybrid functionals.

III. APPLICATIONS

We chose two fairly large systems for demonstrating the performance of our implementation. The first one is a complex of the so-called buckycatcher, $C_{60}H_{24}$, and fullerene C_{60} (ref 35). The buckycatcher consists of two corannulene pincers connected by a small molecular tether. Due to its shape, it can enclose the fullerene molecule and form a van der Waals complex. Published calculations of the interaction energy for this dimer have led to rather discrepant results. Interaction energies of −21 and −26 kcal/mol, calculated by Zhao and Truhlar[36] with M05-2X[37] and M06-2X[38] functionals, respectively, are much smaller than the value of −43 kcal/mol reported by Muck-Lichtenfeld et al.[39] using the B97-D method.[40] The dimer geometry used in the present study was taken from ref 39 (complex C60@2). The structure is presented in Figure 1.

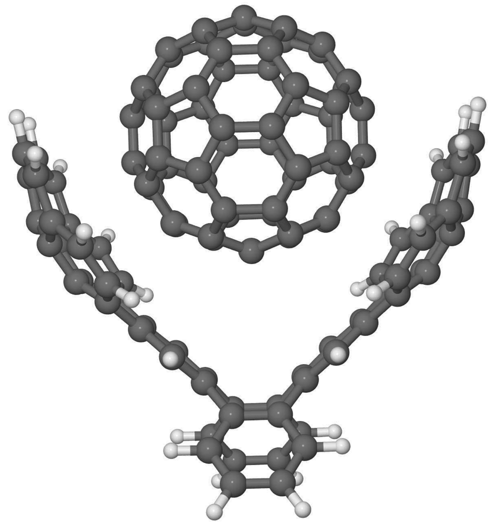

Figure 1. Buckycatcher⋯fullerene dimer.

The other considered system is a complex of the antibiotic vancomycin, $C_{66}H_{75}Cl_2N_9O_{24}$, and the diacetyl-Lys-D-Ala-D-Ala peptide, $C_{16}H_{28}N_4O_6$ (ref 41). This system is a model of binding of the antibiotic with a bacterial wall precursor analogue. A model of vancomycin interacting with several ligands was the subject of the SAPT study of ref 42. The dimer consists of 230 atoms, a challenging test of the efficiency and robustness of our method. The experimental crystal geometry of ref 41 does not contain hydrogen positions. We have added idealized hydrogen positions with the PyMOL program.[43] We have also manually added, in idealized positions, hydrogen atoms to the carboxyl groups making both the vancomycin and the peptide neutral (it is charged in aqueous solution). Neutral molecules are more appropriate for interactions of such systems in a vacuum. The resulting dimer geometry is given in the Supporting Information, and the complex is presented in Figure 2.

For monomer calculations, we used the release 2.9.0 of Orca.[44] The PBE functional[45] was used with an asymptotic correction (AC) of ref 46. The ionization potentials (IP) needed for the AC correction were obtained in a separate B3LYP[47,48] calculation as 6.783 eV for the buckycatcher, 7.112 eV for the vancomycin, and 7.813 eV for the diacetyl-Lys-D-Ala-D-Ala. For the fullerene, we used the experimental IP value of 7.433 eV.[49]

We used two basis sets: def2-SVP[50] and def2-TZVPP.[51] The former is too small to provide accurate dispersion energies but allows for timing comparisons with the larger def2-TZVPP. The latter basis set has been shown to work relatively well in SAPT(DFT) calculations.[11] Although, due to the lack of very diffuse functions, the accuracy of the dispersion energy computed in this basis is not better[12] than that computed using a smaller aug-cc-pVDZ basis set,[52] the latter basis set leads to convergence problems in DFT calculations. For auxiliary basis sets, we used appropriate ones from refs 53 and

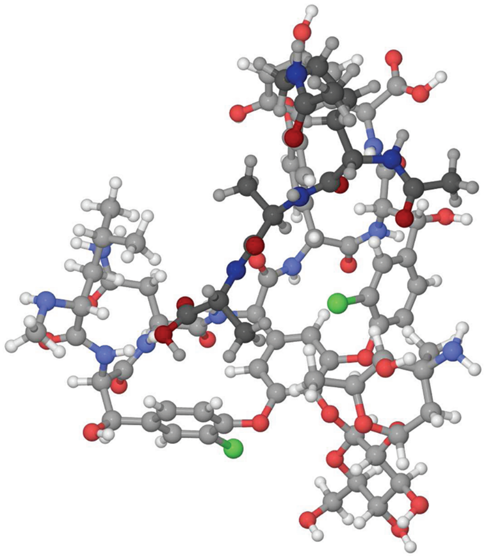

Figure 2. Vancomycin···diacetyl-Lys-D-Ala-D-Ala complex. Vancomycin and the peptide are rendered in lighter and darker shades, respectively.

54 in the monomer calculations and from ref 55 in the kernel and propagator calculations.

In Tables 2 and 3, we show timings of the dispersion energy calculations. We tested the implementation on two machines, a modest two-processor quad core (eight cores in total) Opteron 2.4 GHz with 32 GB RAM and a four-processor quad core (16 cores in total) Opteron 2.4 GHz with 128 GB RAM. In almost all cases, the most time-consuming step is the construction of the kernel matrix, but the propagator part takes comparable time. The Orca code and the three-center integral generation and transformation of eqs 12 and 13 utilized Message Passing Interface (MPI), and all other parts of the code used multithreaded BLAS subroutines for parallelization. Although further work will be needed to achieve massive parallelization of our program, the current implementation should scale reasonably well on machines with up to a few dozen cores. The worst scaling is exhibited by the monomer DFT calculations with the Orca code, but this step takes only a few percent of the total time. For most of the steps, the parallel scaling is worse than linear, but for the propagator calculations with large basis sets, the 16-core calculations are superlinear. This is caused by more memory available on the 16-core machine. The propagator code is fairly I/O intensive, and larger memory buffers improve performance, as discussed in section II. However, even in the largest vancomycin···peptide/def2-TZVPP run, we were able to fit the calculations in 24 GB of memory and perform the propagator calculations using only a factor of 2.3 more time on the eight-core machine. We estimate that calculations for 300-atom dimers in the def2-TZVPP basis set should be doable with the current implementation.

In Tables 4 and 5, the dispersion energies computed for the two studied systems are presented. To obtain the complete interaction energies, we have computed dispersionless interaction energies using the dispersionless density functional (dlDF).[56] Together with dlDF, a site−site dispersion function had been developed in ref 56. This function has been subsequently improved in ref 57, and we present the results given by the latter function (labeled D_{as}) in Tables 4 and 5. The dlDF energies were calculated with the def2-SVP basis set including the counterpoise correction. For the buckycatcher complex, we also compare with the literature results.[36,39]

The CKS dispersion energies computed in the def2-TZVPP basis set are quite close to those given by the dispersion function D_{as}: the discrepancies are only 6.4% and 5.5% for the buckycatcher···fullerene and vancomycin···peptide complexes, respectively. This provides, on one hand, a good check of the correctness of our codes for these large systems and, on the other hand, a confirmation of the high quality of the dispersion function used here way beyond the systems on which it was tested before. One can analyze this agreement in more detail by taking into account that D_{as} was fitted on a training set containing the sums of SAPT(DFT)/aug-cc-pVTZ dispersion and exchange-dispersion energies. The aug-cc-pVTZ basis gives dispersion energies saturated better in basis set size than the def2-TZVPP basis. On the basis of the calculations for the coronene dimer[12] in aug-cc-pVDZ and aug-cc-pVTZ bases and of our unpublished calculations for the $H_2@C_{60}$ complex in aug-cc-pVDZ and def2-TZVPP bases, we can estimate that aug-cc-pVTZ should give dispersion energies about 5% larger in magnitude than def2-TZVPP. This factor alone would near perfectly account for the discrepancy in the case of the buckycatcher···fullerene complex but increase the discrepancy for the vancomycin···peptide complex. However, if one includes the other factor, the exchange-dispersion energies which are always positive and range from 5% to 10% of the magnitude of the dispersion energy [for example, 8% for the coronene dimer[12]], the agreement becomes excellent for vancomycin···peptide and deteriorates for buckycatcher···fullerene. In any case, discrepancies below 10% between the predictions of D_{as} and our *ab initio* calculations should be considered as fairly small for such large and diverse systems; however, for systems with positive dispersionless energy, the relative error will be larger for the total interaction energies.

After adding the dispersionless interaction energy, the total interaction energy for buckycatcher···fullerene can be compared

Table 2. Wall Times (in hours) of the Buckycatcher···Fullerene Complex Calculations[a]

level/basis/cores	def2-SVP/8	def2-SVP/16	def2-TZVPP/8	def2-TZVPP/16
DFT	3.5	2.6	5.7	4.6
kernel	6.0	3.5	23.4	13.1
transformation	0.5	0.3	3.1	1.9
dispersion	3.8	2.1	24.4	12.0
total	13.8	8.5	56.6	31.6

[a] In the symbol "basis/*n*", *n* is the number of cores used. "DFT" corresponds to monomer calculations, "kernel" to calculations of the $\bar{\mathbf{K}}$ matrix, "transformation" to computation of the $\bar{\mathbf{L}}$ matrix and three- and two-center integrals, "dispersion" to CKS propagator and dispersion energy calculations. The complex size is 148 atoms, the number of occupied orbitals in each monomer is 194 and 180, respectively. For def2-SVP, N_{basis} = 1820 and N_{aux} = 6152, and for def2-TZVPP, N_{basis} = 4112 and N_{aux} = 9960. Calculations were performed in dimer-centered basis sets.

Table 3. Wall Times (in hours) of the Vancomycin···Di-acetyl-Lys-D-Ala-D-Ala Dimer[a]

level/basis/cores	def2-SVP/8	def2-SVP/16	def2-TZVPP/8	def2-TZVPP/16
DFT	1.8	1.4	10.1	9.4
kernel	20.6	12.1	91.1	50.1
transformation	1.1	0.7	7.0	4.7
dispersion	9.6	4.9	81.4	35.3
total	33.1	19.1	189.6	99.5

[a]For the explanation of symbols, see Table 2. The complex size is 230 atoms, the numbers of occupied orbitals in each monomer are 380 and 100, respectively. For def2-SVP, N_{basis} = 2301 and N_{aux} = 7566, and for def2-TZVPP, N_{basis} = 5401 and N_{aux} = 12790. Calculations were performed in dimer-centered basis sets.

Table 4. Dispersion, Dispersionless, And Total Interaction Energies (in kcal/mol) of the Buckycatcher···Fullerene Complex[a]

dlDF	51.7
D_{as}	−87.8
$E^{(2)}_{disp}$(CKS)/def2-SVP	−68.9
$E^{(2)}_{disp}$(CKS)/def2-TZVPP	−82.6
dlDF+D_{as}	−36.1
dlDF+$E^{(2)}_{disp}$(CKS)/def2-TZVPP	−30.9
B97-D[39]	−43.1
M06-2X[36]	−26.4
M05-2X[36]	−20.7

[a]The PBE functional was used in the calculations of CKS dispersion energies. The dimer geometry was taken from ref 39.

Table 5. Dispersion, Dispersionless, and Total Interaction Energies (in kcal/mol) of the Vancomycin···Di-acetyl-Lys-D-Ala-D-Ala Complex[a]

dlDF	18.6
D_{as}	−67.0
$E^{(2)}_{disp}$(CKS)/def2-SVP	−52.6
$E^{(2)}_{disp}$(CKS)/def2-TZVPP	−70.9
dlDF+D_{as}	−48.4
dlDF+$E^{(2)}_{disp}$(CKS)/def2-TZVPP	−52.3

[a]The PBE functional was used in the calculations of the CKS dispersion energies. The geometry of the dimer is given in the Supporting Information.

with the literature data. Our value lies between the M06-2X and B97-D results. We believe that the M05-2X and M06-2X class of functionals cannot be reliable for such large systems because these functionals perform poorly in the long-range regime[56] and significant fragments of the dimers considered here are very far apart. Since the B97-D values overestimated the binding energy for the coronene dimer and graphene,[12] it is reasonable to expect this to be also the case for the similar buckycatcher···fullerene complex. Thus, there are reasons to believe that our results provide the best estimate of the interaction energy for this complex.

For the vancomycin···peptide complex, the dispersionless energy is also positive, and it is still the dispersion energy that binds the system. This may seem surprising for a system of this type since most of the biochemical models rely on the electrostatic interactions and hydrogen bonds only. The fact that for larger molecules, including strongly polar ones, the dispersion energy becomes the dominating component was shown in ref 58. It was also pointed out there that approaches to interactions of biomolecules that use only electrostatic arguments cannot be correct.

IV. CONCLUSIONS

We have shown that the dispersion energies can be computed from CKS propagators for systems with about 200 atoms and 5000 basis functions using modest computational resources. Since such dispersion energies have been shown to be very accurate, much more accurate than those given by SAPT0 or SAPT(KS) approaches, our work enables an accurate first-principles insight into intermolecular interactions of nanosized systems. By combining our dispersion energies with the remaining components from dlDF or SAPT(KS), one obtains accurate total interaction energies. This approach opens possibilities of studies in the area of supramolecular chemistry, interactions of biomolecules, and nanotechnology.

■ ASSOCIATED CONTENT

Ⓢ Supporting Information

Geometry of the vancomycin···diacetyl-Lys-D-Ala-D-Ala complex. This material is available free of charge via the Internet at http://pubs.acs.org.

■ AUTHOR INFORMATION

Corresponding Author

*E-mail: rafal.podeszwa@us.edu.pl.

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

This work was supported by an ARO grant and by the NSF grant CHE-0848589. R.P. acknowledges the support by the Polish Ministry of Science and Higher Education grant no. N N204 123337.

■ REFERENCES

(1) Hesselmann, A.; Jansen, G. *Phys. Chem. Chem. Phys.* **2003**, *5*, 5010.

(2) Podeszwa, R.; Szalewicz, K. *Chem. Phys. Lett.* **2005**, *412*, 488−493.

(3) Vissers, G. W. M.; Hesselmann, A.; Jansen, G.; Wormer, P. E. S.; van der Avoird, A. *J. Chem. Phys.* **2005**, *122*, 054306.

(4) Bukowski, R.; Szalewicz, K.; Groenenboom, G. C.; van der Avoird, A. *J. Chem. Phys.* **2006**, *125*, 044301.

(5) Podeszwa, R.; Bukowski, R.; Szalewicz, K. *J. Phys. Chem. A* **2006**, *110*, 10345−10354.

(6) Hesselmann, A.; Jansen, G.; Schütz, M. *J. Am. Chem. Soc.* **2006**, *128*, 11730−11731.

(7) Podeszwa, R.; Bukowski, R.; Rice, B. M.; Szalewicz, K. *Phys. Chem. Chem. Phys.* **2007**, *9*, 5561−5569.

(8) Podeszwa, R.; Szalewicz, K. *Phys. Chem. Chem. Phys.* **2008**, *10*, 2735−2746.

(9) Misquitta, A. J.; Welch, G. W. A.; Stone, A. J.; Price, S. L. *Chem. Phys. Lett.* **2008**, *456*, 105−109.

(10) Podeszwa, R.; Rice, B. M.; Szalewicz, K. *Phys. Rev. Lett.* **2008**, *101*, 115503.

(11) Korona, T.; Hesselmann, A.; Dodziuk, H. *J. Chem. Theory Comput.* **2009**, *5*, 1585−1596.

(12) Podeszwa, R. *J. Chem. Phys.* **2010**, *132*, 044704.

(13) van der Avoird, A.; Podeszwa, R.; Szalewicz, K.; Leforestier, C.; van Harrevelt, R.; Bunker, P. R.; Schnell, M.; von Helden, G.; Meijer, G. *Phys. Chem. Chem. Phys.* **2010**, *12*, 8219−8240.

(14) Taylor, D. E.; Rob, F.; Rice, B. M.; Podeszwa, R.; Szalewicz, K. *Phys. Chem. Chem. Phys.* **2011**, *13*, 16629−16636.

(15) Williams, H. L.; Chabalowski, C. F. *J. Phys. Chem. A* **2001**, *105*, 646−659.

(16) Misquitta, A. J.; Szalewicz, K. *Chem. Phys. Lett.* **2002**, *357*, 301−306.

(17) Misquitta, A. J.; Jeziorski, B.; Szalewicz, K. *Phys. Rev. Lett.* **2003**, *91*, 033201.

(18) Misquitta, A. J.; Szalewicz, K. *J. Chem. Phys.* **2005**, *122*, 214109.

(19) Misquitta, A. J.; Podeszwa, R.; Jeziorski, B.; Szalewicz, K. *J. Chem. Phys.* **2005**, *123*, 214103.

(20) Hesselmann, A.; Jansen, G. *Chem. Phys. Lett.* **2002**, *357*, 464−470.

(21) Hesselmann, A.; Jansen, G. *Chem. Phys. Lett.* **2002**, *362*, 319−325.

(22) Hesselmann, A.; Jansen, G. *Chem. Phys. Lett.* **2003**, *367*, 778−784.

(23) Jeziorski, B.; Moszyński, R.; Szalewicz, K. *Chem. Rev.* **1994**, *94*, 1887−1930.

(24) Hesselmann, A.; Jansen, G.; Schütz, M. *J. Chem. Phys.* **2005**, *122*, 014103.

(25) Bukowski, R.; Podeszwa, R.; Szalewicz, K. *Chem. Phys. Lett.* **2005**, *414*, 111−116.

(26) Podeszwa, R.; Bukowski, R.; Szalewicz, K. *J. Chem. Theory Comput.* **2006**, *2*, 400−412.

(27) Görling, A.; Heinze, H. H.; Ruzankin, S. P.; Staufer, M.; Rösch, N. *J. Chem. Phys.* **1999**, *110*, 2785−2799.

(28) Hohenstein, E. G.; Sherrill, C. D. *J. Chem. Phys.* **2010**, *132*, 184111.

(29) Hohenstein, E. G.; Sherrill, C. D. *J. Chem. Phys.* **2010**, *133*, 014101.

(30) Hohenstein, E. G.; Sherrill, C. D. *J. Chem. Phys.* **2010**, *133*, 104107.

(31) Hohenstein, E. G.; Parrish, R. M.; Sherrill, C. D.; Turney, J. M.; Schaefer, H. F. *J. Chem. Phys.* **2011**, *135*, 174107.

(32) Dunlap, B. I.; Connolly, J. W. D.; Sabin, J. R. *J. Chem. Phys.* **1979**, *71*, 4993−4999.

(33) Goto, K.; van de Geijn, R. A. *ACM Trans. Math. Soft* **2008**, *34*, 12.

(34) Schmidt, M. W.; Baldridge, K. K.; Boatz, J. A.; Elbert, S. T.; Gordon, M. S.; Jensen, J. H.; Koseki, S.; Matsunaga, N.; Nguyen, K. A.; Su, S. J.; Windus, T. L.; Dupuis, M.; Montgomery, J. A. *J. Comput. Chem.* **1993**, *14*, 1347−1363.

(35) Sygula, A.; Fronczek, F. R.; Sygula, R.; Rabideau, P. W.; Olmstead, M. M. *J. Am. Chem. Soc.* **2007**, *129*, 3842−3843.

(36) Zhao, Y.; Truhlar, D. G. *Phys. Chem. Chem. Phys.* **2008**, *10*, 2813−2818.

(37) Zhao, Y.; Schultz, N. E.; Truhlar, D. G. *J. Chem. Theory Comput.* **2006**, *2*, 364−382.

(38) Zhao, Y.; Truhlar, D. G. *Theor. Chem. Acc.* **2008**, *120*, 215−241.

(39) Muck-Lichtenfeld, C.; Grimme, S.; Kobryn, L.; Sygula, A. *Phys. Chem. Chem. Phys.* **2010**, *12*, 7091−7097.

(40) Grimme, S. *J. Comput. Chem.* **2006**, *27*, 1787−1799.

(41) Nitanai, Y.; Kikuchi, T.; Kakoi, K.; Hanamaki, S.; Fujisawa, I.; Aoki, K. *J. Mol. Biol.* **2009**, *385*, 1422−1432.

(42) Li, X.; Volkov, A.; Coppens, P.; Szalewicz, K. *Acta Crystallogr., Sect. . D* **2006**, *62*, 639−647.

(43) The PyMOL molecular graphics system, version 1.2r1; Schrödinger, LLC: New York, 2009.

(44) Neese, F.; Becker, U.; Ganyushin, D.; Hansen, A.; Liakos, D.; Kollmar, C.; Kossmann, S.; Petrenko, T.; Reimann, C.; Riplinger, C.; Sivalingam, K.; Wezisla, B.; Wennmohs, F. *ORCA*; Universität Bonn: Bonn, Germany.

(45) Perdew, J. P.; Burke, K.; Ernzerhof, M. *Phys. Rev. Lett.* **1996**, *77*, 3865−3868.

(46) Grüning, M.; Gritsenko, O. V.; van Gisbergen, S. J. A.; Baerends, E. J. *J. Chem. Phys.* **2001**, *114*, 652.

(47) Becke, A. D. *J. Chem. Phys.* **1993**, *98*, 5648−5652.

(48) Stephens, P. J.; Devlin, F. J.; Chabalowski, C. F.; Frisch, M. J. *J. Chem. Phys.* **1994**, *98*, 11623−11627.

(49) Lias, S. G. Ionization Energy Evaluation. In NIST Chemistry WebBook, NIST Standard Reference Database Number 69. http://webbook.nist.gov (accessed April 2012).

(50) Schäfer, A.; Horn, H.; Ahlrichs, R. *J. Chem. Phys.* **1992**, *97*, 2572.

(51) Weigend, F.; Ahlrichs, R. *Phys. Chem. Chem. Phys.* **2005**, *7*, 3297.

(52) Kendall, R. A.; Dunning, T. H., Jr.; Harrison, R. J. *J. Chem. Phys.* **1992**, *96*, 6796−6806.

(53) Eichkorn, K.; Treutler, O.; Öhm, H.; Häser, M.; Ahlrichs, R. *Chem. Phys. Lett.* **1995**, *242*, 652.

(54) Eichkorn, K.; Weigend, F.; Treutler, O.; Ahlrichs, R. *Theor. Chem. Acc.* **1997**, *97*, 119.

(55) Weigend, F.; Köhn, A.; Hättig, C. *J. Chem. Phys.* **2002**, *116*, 3175−3183.

(56) Pernal, K.; Podeszwa, R.; Patkowski, K.; Szalewicz, K. *Phys. Rev. Lett.* **2009**, *103*, 263201.

(57) Podeszwa, R.; Pernal, K.; Patkowski, K.; Szalewicz, K. *J. Phys. Chem. Lett.* **2010**, *1*, 550−555.

(58) Bukowski, R.; Szalewicz, K.; Chabalowski, C. *J. Phys. Chem. A* **1999**, *103*, 7322−7340.