PESPIP: Software to Fit Complex Molecular and Many-body Potential Energy Surfaces with Permutationally Invariant Polynomials

Paul L. Houston*,[1, a)] Chen Qu,[2] Qi Yu,[3] Riccardo Conte*,[4, b)] Apurba Nandi,[5] Jeffrey K. Li,[5] and Joel M. Bowman*[5, c)]

[1)] *Department of Chemistry and Chemical Biology, Cornell University, Ithaca, New York 14853, U.S.A. and Department of Chemistry and Biochemistry, Georgia Institute of Technology, Atlanta, Georgia 30332, U.S.A*

[2)] *Independent researcher, Toronto, Ontario, Canada*[d)]

[3)] *Department of Chemistry Yale University, New Haven, Connecticut 06520, U.S.A.*[e)]

[4)] *Dipartimento di Chimica, Università degli Studi di Milano, via Golgi 19, 20133 Milano, Italy*

[5)] *Department of Chemistry and Cherry L. Emerson Center for Scientific Computation, Emory University, Atlanta, Georgia 30322, U.S.A.*

(Dated: 10 November 2022)

We wish to describe a potential energy surface by using a basis set of permutationally invariant polynomials (PIPs) whose coefficients will be determined by numerical regression so as to smoothly fit a data set of electronic energies as well as, perhaps, gradients. The polynomials will be powers of transformed internuclear distances, usually either Morse variables, $exp(-r_{i,j}/\lambda)$, where λ is a constant range hyperparameter, or reciprocals of the distances, $1/r_{i,j}$. The question we address is how to create the most efficient basis set, including a) which polynomials to keep or discard, b) how many polynomials will be needed, c) how to make sure the polynomials reproduce correctly the zero interaction at large distance, d) how to ensure special symmetries, and e) how to calculate gradients efficiently. This article discusses how these questions can be answered by using a set of programs to choose and manipulate the polynomials as well as to write efficient Fortran programs for calculation of energies and gradients. A user-friendly interface for access to monomial symmetrization approach (MSA) results is also described. The software for these programs is now publicly available.

I. INTRODUCTION

One goal of electronic structure calculations is to determine how the energy of a system of atoms changes with geometry. A list of energies and geometries is sometimes useful, but what theoretical chemists need most is a function, one that is fast to evaluate, reproduces precisely and smoothly the available electronic structure data, and capable of being used in classical and quantum molecular dynamics. How this function is derived from the data has received modern, innovative attention but is also based on old roots. Many recent perspectives and reviews have shown how new machine learning techniques, for examples, Gaussian processes and neural networks, can provide an answer.[1–7] Older methods, which use machine learning to fit a basis set to the data using coefficients determined by numerical regression, are still improving and currently remain the fastest performers for molecules up to about 15 or more atoms,[8] and for clusters of identical small molecules up to many hundreds of atoms, via a many-body expansion.

Two major challenges need to be solved for extension of these methods to even larger systems. The first challenge, which affects both newer and more established methods, is the need for larger and larger data sets. The number of high-accuracy energies and gradients required as a function of geometry becomes larger as the geometric description requires an increasing number of interatomic parameters. One recent advance, "Δ-machine learning", allows the use of fast, lower-level electronic structure calculations, e.g., Density Functional Theory (DFT), for the major fraction of the data set and then augmenting these with far fewer high-level calculations, e.g., at the CCSD(T) level, and using a correction fit to the difference.[9] Another advance is the use of "transfer learning", in which knowledge gained from solving one problem is used to solve a different but related problem.[10] The second challenge is that the basis set, typically a polynomial expansion, can become so large as to be unwieldy. It is this second challenge that we address here. Are there effective methods for achieving high accuracy and speed but using smaller, more efficient, large-molecule basis?

Setting aside the well-described development of permutational invariant polynomials (PIPs) themselves,[8,11–19] recently reviewed,[20] we briefly discuss the most recent work on new methods for using PIPs. We will limit our review here to using PIPs for determining global potential energy surfaces; local fitting will be considered in the Discussion section. The field has been supported since roughly 2005 by a library of PIPs for roughly 60

[a)] Electronic mail: plh2@cornell.edu
[b)] Electronic mail: riccardo.conte1@unimi.it
[c)] Electronic mail: jmbowma@emory.edu
[d)] Electronic mail: szquchen@gmail.com
[e)] Electronic mail: q.yu@yale.edu

molecule types up to 10 atoms.[11] This library has been distributed to a number of research groups who specifically requested PIP PES based on the PIPs in this library. There are some limitations of this library and we mention those when we discuss the new PIP software and the capability to also create a library of PIPs. That libary of PIPs was followed soon after by monomial symmetrization approach (MSA) software that determines PIPs given the permutational symmetry and the desired order.[21,22] First, we digress briefly to establish the notation for basis, because this notation is the input for the MSA software. We will describe in a later section a user-friendly interface for generating MSA basis set output and for fitting a potential energy surface to electronic structure data.

The permutational symmetry notation for a molecule is determined by enumerating which like atoms of the molecule permute with one another. A simple example might be acetic acid, CH_3COOH. The full symmetry designation would be 422, where the 4 refers to permutation of the identical hydrogens, one 2 refers to the permutation of the two carbons, and the other 2 refers to that of the two oxygens. In many cases, one might want to use a reduced symmetry, especially if the energy range of the potential will be low enough so that some permutations can be assumed not to happen. For example, it is unlikely at low energies either that the two carbons will exchange with one another or that the two oxygens will. The reduced symmetry for this situation would be designated 41111. It is also unlikely at low enough energies that the acidic hydrogen will exchange with the hydrogens on the CH_3 group, in which case the symmetry would be designated 311111. Reduced symmetries have the advantage of being usually faster to calculate, even though they have more polynomial terms, whereas full symmetries have fewer polynomials but are often slower because of the larger number of monomials that are symmetrized. In any case, the MSA software will use the user-selected symmetry designation, along with the desired maximum order of the polynomials, to provide polynomials that are permutationally invariant to the designated exchanges. The user-selected polynomial order is the maximum sum of the polynomial exponents; the number of polynomials increases very strongly with the order. The description of an MSA basis set is usually denoted by joining the permutational symmetry designation and the order with an underscore; e.g., The 311111 symmetry with maximum order 4 would be designated 311111_4.

Substantial new work using PIPs has appeared in the past several years. Paesani and co-workers have used PIPs in the the MB-pol potential[17,23] and many-body potentials of solvated molecules, e.g., CO_2/H_2O.[24] Győri and Czakó have introduced a "RoboSurfer" program, which uses PIPs, that automates the selection of new geometries, performs *ab initio* computations, and improves the PES iteratively. They applied it using PIPs to the reaction of $CH_3Br + F^-$.[25] Moberg et al. have described a method for using dictionary learning with a "greedy" selection to reduce the size of bases without appreciable loss of accuracy, and they have also shown that parallelization can automatically generate PIPs for complex systems with multiple reactive channels using software they call "PIPPy".[26,27] The Schaefer group has introduced PES-Learn, a package of software for automated generation of machine learning models for PESs.[28] Koner and Meuwly have reported a permutationally invariant method reproducing kernel-based PES for polyatomic molecules up to 10 atoms.[29] That method uses an "n-body" representation of the potential, which appears to be similar, at least qualitatively to the "n-mode"[30,31]. Truhlar and his group have studied several PESs using PIPs, including $N+N_2$, $N+O_2$, and N_2+N_2.[32–35] For many of these latter surfaces, as well as in the work of Moberg et al., a method is described for removing "disconnected" terms in the basis. It accomplishes what we call purification and compaction, as will be discussed below.[26,27,36]

To conclude these introductary remarks, it is reasonable to ask "why PIPS"? – that is, why do we recommend PIPs for PESs of molecules and for components of many-body expansions. First, we note that roughly fifty PIP PESs have already been developed and reviewed.[20] As of 2018, these for molecules with up to 10 atoms. As of 2022, PIP PESs have been reported for 15 atom molecules and for the 12-atom 4-body water interaction potential. So this is an indication that the approach can be used for many gas-phase molecules of potential interest. But another aspect of PIPs that recommends the approach is the efficiency of the method. Since the representation is simple and linear in the parameters it seems clear immediately that the evaluation of the PES would be fast. Recall that the generic expression for the potential V in terms of PIPs is given by

$$V(\boldsymbol{\tau}) = \sum_{i=1}^{n_p} c_i p_i(\boldsymbol{\tau}), \tag{1}$$

where c_i are linear coefficients, p_i are PIPs, n_p is the total number of polynomials for a given maximum polynomial order, and $\boldsymbol{\tau}$ are transformed internuclear distances, usually either Morse variables, $\exp(-r_{n,m}/\lambda)$, or inverse distances, $1/r_{n,m}$, where $r_{n,m}$ is the internuclear distance between atoms n and m. This simple expression has been used to fit energies of small molecules for many years where the polynomials were just monomials. There were two serious limitations though with that approach. First, the number of monomials grows highly non-linearly with the number of degrees of freedom, and also the expression is not manifestly invariant with respect to permutations of like atoms. PIPs obviously solve the second problem and also greatly mitigate the steep increase in the number of terms while keeping the "linearity" of the fitting approach and the evaluation of the sum. In fact the speed of evaluation of the PIPs representation was recently demonstrated for ethanol, where it greatly outperformed various Neural Network and Kernel Ridge methods.[37]

We now present a brief review of recently developed methods used by our group to make best use of the polynomials available with a several goals in mind. These are achieved by being able to selectively eliminate (or add) PIPs to the basis. This capability is especially important for application of PIP fits for molecules with more than 10 atoms. These methods are the focus of Mathematica[38] software described in detail below and made available in the Supplemental Material. The Theory and Programming section provides the theoretical background for each method and serves as a guide to the software. Examples are presented in the Results section and discussed subsequently. An overview of the molecular systems that have been studied in our group using these methods is provided in Table I. We start with an overview of four techniques that either select, from all the polynomials provided by the MSA software, those that are most important for specific cases or that provide simplification and efficient generation of gradients. Section II then provides a deeper description of the theory and programming of these techniques.

Fragmentation and Elimination of Duplicate Polynomials: Based loosely on ideas from groups trying to calculate the energies of large molecules based on fragmentation methods,[51–53] a fragmented PIP approach has been developed to simplify and prioritize PIP bases for large molecules. In this approach the total PIP basis is given by the union of PIP bases for fragments of the molecule. The method was tested for two H_3O^+ and CH_4 and applied to the 12-atom molecule, *N*-methyl acetamide (NMA).[43,54] The idea of using fragmentation for prioritizing the basis is to divide the molecule into (usually) overlapping fragments, each having its own permutational symmetry and total polynomial order. The fragments are chosen based on large internuclear distances (and thus small Morse variables) of non-common atoms in different fragments. For NMA these are the two sets of H atoms on each distant methyl rotors for which the corresponding Morse variables are small. However, as shown in ref. 43 those small interactions can be accounted for in a separate fragment involving just those nine Morse variables. Considering the whole molecule in the case of NMA, there were 66 internuclear distances to consider, with 8040 PIPs at third order. For a 2-fragment case, there were 57 internuclear distances to consider, with 6056 PIPs. For the 2-fragment basis set, the evaluation of the fitting basis was 20 times faster than for the basis from the full molecule, with RMS fitting errors of energies and gradients that were only about 20% larger. Fragmentation was also recently used in a study of tunneling splittings in acetylacetone.[49] In that case both a full symmetry basis for the entire molecule and a basis for four overlapping fragments of 9, 11, 11, and 8 atoms was considered. The full basis has 52 626 monomials and 6207 polynomials, whereas the fragmented basis had 3609 monomials and 24 030 polynomials. The fragmented basis was both twice as fast to evaluate as the full symmetry/full molecule basis and also produced RMS errors for the potential and gradients that were better by about a factor of two. In the first of these two examples, the bases were determined in distinct steps that required a hands-on approach. In the second case, the software described below was used both to automate the process and to make more efficient use of the polynomials.

Adding or Pruning: One can easily imagine either adding polynomials to a smaller basis or "pruning" polynomials from a larger one so as to get a reasonable compromise between basis accuracy and speed. The idea is similar to that in fragmentation: we want to keep the polynomials that have large values (these have small distances) and discard those that have small values (these have long distances). Instead of using fragmentation to decide which ones should be kept or discarded, we use the data set itself, the set of energies and, perhaps, gradients to which the polynomial expansion will be fit. Details will be provided in the Methods section.

Pruning or adding polynomials to form new bases, has been employed recently in several studies, In glycine,[44,46] we reported six fits to an extensive data set containing 70 099 geometries spanning an energy range up to more than 60 000 cm^{-1} above the global minimum. The fits were to the full molecule, to a 2-fragment version and to a 3-fragment version, each constructed for both third or fourth order. The number of polynomials ranged from 1022 to 46 654. As expected, the fourth order fits were the most accurate, whereas the third order fit with two fragments was the fastest. The RMS errors for the energy were between 7.5 and 132 cm^{-1}, and the mean absolute errors for the vibrational frequencies of the global minimum structure were between 7.2 and 31.2 cm^{-1}. The PESs determined with these bases were suitable for diffusion Monte Carlo (DMC) calculations of the wavefunctions for several conformers as well as their zero-point energies (ZPEs). Results for several fourth-order fits that were pruned to provide smaller basis sets or which had polynomials added to form larger ones were also obtained.[44]

For tropolone,[47] we investigated bases for the full molecule using total polynomial order of three for a two-fragment and a four-fragment version. Pruning and adding of polynomials produced further bases, as described in some detail, but these pruned or augmented bases were not used in the final analysis of tunneling splitting. Although the fragmented fits had weighted RMS errors that were not so different from the full fit, what we were most interested in was the hydrogen exchange between the two oxygens. There are four low-energy stationary points on the surface, and the fit to the full molecule gave better values for these, particularly for the global minimum and the transition state to hydrogen transfer.

For ethanol,[8] pruning was used to obtain a data set of 8895 polynomials that was somewhat smaller than the order-four 321111 set with 14 752 polynomials.

Purification/Compaction/Grouping: In many applications to non-covalent systems, we used the well-

TABLE I. Potential energy surfaces with bases that used the methods described in the text.

Molecular System	Method	Year(s)	References
Ar-HOCO	Purification	2015	39
CH_4-H_2O	Purification	2015	40
$H_2(H_2O)$	Purification	2015	41
$H_2(H_2O)_2$	Purification	2015	41
$CO_2@(H_2O)_{20}$	Purification	2017	42
N-methyl acetamide	Fragmentation	2019-2021	9,43,44
CH_4	Derivatives	2019	45
Glycine	Fragmentation, Pruning	2020	44,46
Tropolone	Fragmentation, Pruning	2020	47
Acetylacetone	Fragmentation	2021	48,49
H_2O, 3-body, 4-body	Purification	2021	50
CH_3CH_2OH	Pruning, Reverse Derivatives	2022	8

known many-body expansion where we focus on each intrinsic n-body interaction. The "particles" may be identical, in which case we would refer to each as a monomer, or they may be different, in which case we would refer to each as a fragment. The many-body expansion equation in terms of the n-body contributions is for N monomers

$$V(1,\cdots,N) = \sum_{i=1}^{N} V_{1b}(i) + \sum_{i>j}^{N} V_{2b}(i,j) + \sum_{i>j>k}^{N} V_{3b}(i,j,k) + \sum_{i>j>k>l}^{N} V_{4b}(i,j,k,l) + \cdots . \quad (2)$$

A simple example is the interaction of Ar with the HOCO radical, studied previously.[39] In this case, the first term in the expansion, $\sum_{i=1}^{N} V_{1b}(i)$, would correspond to the sum of the potential energies of the Ar and the HOCO, whereas the second term would be due purely to the intrinsic interaction of the Ar and HOCO. (If there is a second Ar atom, the third term would describe the intrinic 3-b Ar-Ar-HOCO interaction.) The polynomial basis for this system was at first taken as 2111_5, with 1632 polynomials. Note, however, that for the 2-body interaction potential, the potential energy must go to zero if the Ar and HOCO are far away from one another. Most of the polynomials in the 2111_5 basis do have this property; however, some do not. Those polynomials contain only Morse variables that depend on the internal fragments variables, which do not go to zero in the separated fragment limit. This was noted in the 2009 review by Braams and Bowman.[11] The non-separablity is however small numerically, provided that sufficient data is contained in the dataset. The non-separabilty and a simple solution to eliminate it was reported by Truhlar and co-workers.[36] Clearly, these polynomials should be eliminated so as to ensure the correct zero interaction at large distance and that indeed was the solution suggested. The process of identifying these polynomials is what we call "purification" of the basis set. The principle is simple: one takes a random geometry and compares the values of each polynomial for this geometry to those for a geometry in which the components, here Ar and HOCO, are removed from one another by a large distance. If the polynomial value goes to zero, then it is kept. If it does not, it is identified for later removal. In the case of the Ar-HOCO system, of the original 1632 polynomials, 1376 (84 percent) had the correct zero interaction at large distance. The process of eliminating the ones with the wrong behavior is what we call "compaction"; it involves renaming the polynomials with the wrong behavior, seeing which of the monomials and renamed polynomials are required to calculate the values of those polynomials with the correct zero limit, discarding the rest, and renumbering all remaining monomials, renamed polynomials, and regular polynomials. Other examples of the use of purification in our group may be found elsewhere;[39,40,42,50,55] in some of these the purification has been done "by hand", whereas in others the software to be described in the Methods section was used.

In examples where there are identical molecules, there is one other issue that must be confronted. Consider a cluster of three water molecules. One might consider the full symmetry 63, but as discussed above many of the permutations are unfeasible, and this fact and computational efficiency lead us to consider 222111 symmetry, for which the hydrogens on each water permute, but no atoms permute *between* water molecules. In addition, we wish to describe the invariance of the potential with permutation of intact water molecules. For three waters, there are six such permutations. Unfortunately, while MSA software provides results for any permutation of individual atoms, it does not provide results for the permutations of whole molecules. There are two ways to correct this issue. One is to replicate the energies for all relevant permutations. In the case of the three water molecules, this is a factor of 6 and this enlarges the ultimate datasize by this factor. A better method is to identify those groups of polynomials that have permu-

tational symmetry with respect to the water exchange and to then form "superpolynomials" that are the sum of the polynomial members of each group. The method for identifying such groups will be described below. It has been used successfully for the the 3- and 4-body potentials for water,[50] results for which will be presented below.

Analytical Gradients: The orginal PIPs software, namely the large library of primary and secondary invariants[11] and the MSA software,[22] did not have analytical gradients. And so gradients were obtained using finite-difference methods, which have a minimum time cost for the complete set of gradients about equal to $6Nt_e$, where N is the total number of atoms and t_e is the time for calculating the energy. Recently, a fairly straight-forward method for analytical derivatives was introduced and used for CH_4.[12,45] While better than the finite-difference method in accuracy and speed, it still requires the calculation of $3N$ derivatives to obtain all the gradients, at a time cost of approximately $3Nt_e$. This cost can be decreased substantially at the price of increased storage by using a "fast-forward" technique described elsewhere.[8] This same reference also shows, however, that it is possible to use a reverse, or backwards, derivative approach[56,57] that, for molecules up to at least 15 atoms, scales in time as 3 to 4 times t_e, independent of the size of the molecule. Details of this method and the software that has been developed to calculate reverse gradients for PIPs will be discussed and illustrated below.

II. THE THEORY AND PROGRAMMING OF THESE METHODS

A. Fragmentation and Elimination of Duplicate Polynomials

Recall that the monomials[22] used to make up the polynomial basis have the form of Eq. (3):

$$\tau_1^{n_1}\tau_2^{n_3}\cdots\tau_l^{n_l}, \quad l = N(N-1)/2, \tag{3}$$

where the τ_i are transformed internuclear distances, typically Morse variables, n_i are exponents, and N is the total number of atoms. For large molecules, because many internuclear distances are large, the corresponding Morse variables will be nearly zero, so that the monomials including them will also be small and the basis functions containing the monomials can be discarded. Fragmentation of the basis set is a method for prioritizing which polynomials are most important. The molecule is divided into overlapping fragments, each having its own permutational symmetry and order, and chosen so that long internuclear distances, where interactions are small, are neglected. In the first examples, the total basis was simply generated from the union of those for the individual fragments and the assembly was performed by hand. We have now developed automated methods that easily provide bases for any number and type of fragments and, additionally, delete any duplicate polynomials caused by the desired overlap between the fragments.[44] Most importantly, the fragmentation method provides a path for extending the PIP approach for PES fitting to much larger molecules than were hitherto possible.

The two main goals of the software for fragmentation are to automate the process and to delete any duplicate polynomials that may be produced by combining fragments that have common Morse variables. Two strategies for deleting duplicate polynomials were successful, a pairwise method and a sequential method. In order to determine which values of the Morse variables, $\boldsymbol{\tau}$, the monomials, $\mathbf{m}$, and the polynomials, $\mathbf{p}$, are duplicates, we use the following method in both approaches: a) assign a 16-digit number between 0.1 and 1.0 to each Morse variable; b) calculate all monomials and polynomials; and c) multiply the values by 10^8 and compare the integer parts to identify duplicates. The probability of incorrectly identifying a duplicate with this method is no larger than about 0.01

In the sequential method we take the output of the MSA software for each individual fragment and convert it to Mathematica format. The ($\boldsymbol{\tau}$,**m,p**) are taken as given for the first fragment, whereas for each additional fragment the following steps are taken: the numberings of the new fragments are augmented by the maximum values of those for the previous fragment, the new ($\boldsymbol{\tau}$,**m,p**) are compared with those of previous fragments and marked for deletion, and the lists are joined. After all fragments are added, the surviving ($\boldsymbol{\tau}$,**m,p**) are then consistently and consecutively renumbered, and the Fortran version of the program, similar in format to the output of the MSA program, is written. The pairwise method uses the same first step as the sequential method, and then the second fragment is added, augmenting the numberings of the ($\boldsymbol{\tau}$,**m,p**). The pair of fragments is tested for common ($\boldsymbol{\tau}$,**m,p**), which are marked for deletion. The next fragment is then added, with its ($\boldsymbol{\tau}$,**m,p**) compared in a pairwise fashion with each of the previous fragments, marking duplicates for deletion. Once all fragments have been added, the deletions are made and renumbering of the surviving ($\boldsymbol{\tau}$,**m,p**) is performed. The two methods give identical results and are about equally efficient.

A note is in order to alert the reader that the requirements of symmetry and those of fragmentation might occasionally be at odds with one another. For example, in our study of tropolone,[47] we found that it was undesirable to put two of the distant atoms, the hydrogens on carbons adjacent to the oxygenated carbons, in different fragments, even though doing so would correctly eliminate polynomials based on the long internuclear distance between them. However, putting them in different fragments would mean that they could not permute symmetrically, and this would destroy the symmetry of the hydrogen transfer barrier that we wanted to study. More details, with diagrams can be found in Ref. 47.

The Mathematica program that automates and accomplishes fragmentation and the removal of duplicates is part of the DeleteDuplicatesVXX.x.wl pack-

age, where XX.x denotes the version number. The inputs are placed in a "template" supplied with the software, and the function DeleteDuplicatesAndProvideDerivativesIfDesired[] is executed. Further details are provided in ref. 44 as well as in the Supporting Information for that reference.

B. Polynomial Pruning or Adding

The MSA software is generated from just two inputs, the permutational symmetry of the molecule, using the notation given above, and the total order desired; i.e., the maximum sum of the polynomial powers of the transformed internuclear distances. The permutational symmetry need not be the complete symmetry of the system, but should include the relevant exchange possibilities at the maximum energy of the data set. For the given permutational symmetry, the total number of PIPs will be determined by the maximum polynomial order. In general and up to a limit, the more polynomials, the more accurate will be the fit, but the longer will be the time for calculating the energy and/or gradients. The limit is determined by the need to have far fewer coefficients than one has energy/gradient constraints so as to avoid "overfitting", in which all the points are well-fit but the region between them has strong oscillations in energy. In many cases, one would like to have a number of polynomials/coefficients that is somewhere in between the numbers arrived at for two different choices of order. In such cases, we "prune" the number of polynomials provided by the larger order to the number desired for a reasonable compromise between accuracy and speed, or we add polynomials to the basis for the smaller order by considering multiples of the existing basis functions. We use the following method to decide which of the polynomials should be kept.

A decision on which polynomials are most important may be made by examining the data set. This is a logical choice, because the PES fit will be made to the data set. The first step is to evaluate the maximum values of the each transformed internuclear distance compared to the range of values found among the geometries of the data set. Recall that the transformed internuclear distances are usually Morse or reciprocal values. Thus, long distances have small transformed values (and are less important) than short distances, which have transformed values nearer to unity (for the Morse transformation). Taking the maximal values of the transformed internuclear distances, we then evaluate all the polynomials available from the MSA output. For pruning, we start with a larger order and then eliminate those polynomials with the smaller values until we arrive at the desired number of final polynomials. The method works both for regular polynomials generated by the MSA software and for "superpolynomials" generated as described in the next section. For adding, we generate new polynomials by multiplying together those of the smaller basis, and we keep those that have the largest values until the desired number is reached.

The Mathematica program that adds or prunes polynomials is part of the PolynomialPruningAddingVXX.x.wl package, where XX.x is the current version number. The programs for adding or pruning are PolynomialAdding[] and PolynomialPruning[], respectively. Inputs to these programs are in templates that are part of the Mathematica software.

C. Purification/Compaction/Grouping

Consider a group of identical particles whose potential energy surface we wish to describe using a many-body expansion (MBE, see Eq. (2)) . This potential energy surface will be determined by using a basis of permutationally invariant polynomials (PIPs). We will determine the coefficients in the expansion by numerical regression so as to fit a dataset of electronic energies and, perhaps, gradients for different geometries as smoothly as possible. The PIPs used as the basis will be products or sums of monomials (see Eq. (3)) where the transformed internuclear distances are either Morse ($exp(-r_{i,j}/\lambda)$) or reciprocal variables. However, there are two potential issues for application to the MBE problem. First, not all of the PIPs generated by the MSA software will have the correct zero interaction at large distance in the many-body expansion. Second, the MSA software is not designed to consider the exchange of identical *groups* of atoms, so we need to ensure this type of permutational symmetry by another method.

We first examine the limiting behavior. We use "purification" to refer to the process of identifying and setting aside permutationally invariant polynomials (PIPs) that do not have the correct zero interaction when individual monomers or groups of monomers are removed from the others to a great distance. The fits we seek are to the n-body contributions to the potential energy. When any of the group members are separated by large distances, thus eliminating the n-body nature of the interaction, this n-body contribution must go to zero. We ensure the correct limits for all polynomials by the following method: We take an arbitrary geometry, then remove n-mers from one another to large distances, and finally calculate the values of the PIPs to see whether they go to zero. For example, for water clusters, in the case of the 4-body interactions, we remove each of the 4 water monomers from the other three and each of the six pairs of water dimers from one another. In the case of the 3-body interactions, we remove each water monomer from the remaining pair. In the case of the 2-body interactions, we move the two monomers to a large distance from one another. In our calculations, the distances are augmented by 100 Å, and we accept the polynomial as having the correct behavior if its Morse value is below 10^{-6}.

However, polynomials that do not have the correct zero interaction at large distance cannot immediately be re-

moved from consideration, since there may be other polynomials that, for example, are composed of products between one with an incorrect limit and one with a correct limit. Instead, we rename the ones with an incorrect limit. When the process of evaluating the polynomials is finished, we then look at the definitions of all those with the correct limits and determine which of the monomials and which of the renamed polynomials with incorrect limits are required to calculate them. We then remove those that are not required and renumber those that remain, keeping the order of calculation so that no partial calculation that contributes to any polynomial needs to be performed twice. We call this step in the process "compaction". The result is a set of polynomials that all have the correct zero interaction at large distance and that can be calculated efficiently. Note that the purification procedures described above can only guarantee that the interaction is zero at large distance, but the fit may not have the correct limiting behavior, as it may approach zero too rapidly or slowly. The long-range behavior of the PES will be discussed in detail in Section IV.

We now turn to the issue of permutational symmetry for exchange of the identical monomers. As mentioned previously, this exchange is not taken into account by the MSA software. Thus, the polynomials that we create by purification will not, in general, have permutational symmetry with respect to exchange of identical monomers. In fact, when renamed polynomials (with the wrong limiting behavior) are deleted, it is easy to produce polynomials that do not have permutational symmetry of identical monomers, even if the original basis did have it. As mentioned above, one method for dealing with this symmetry is to augment the dataset by adding all relevant permutations of the Cartesian coordinates and assigning them the same energy. If there are n monomers, this requires a set of $n!$ geometries for each energy. A better method is to identify those groups of polynomials that have permutational symmetry with respect to monomer exchange and to then form "superpolynomials" that are the sum of the polynomial members of each group. Each group will have $n!$, not necessarily unique, members whose sum is independent of the monomer permutations and which can then receive a single coefficient in the energy or gradient determination. This identification of coefficients follows directly from the numerical solution of the least-square equation, and this has been verified in the replication procedure.

We identify the permutationally invariant groups of polynomials by taking a single set of $n!$ permutationally related geometries and calculating the value of each polynomial. Taken together, the groups of polynomials for each permutation will have the same group of values, but the the values of individual polynomials will vary from permutation to permutation. For each permutation, one can form pairs of the polynomial identities and their values, and then sort the pairs by their values. Looking at all pairs that have the same value component in all permutations gives the identities of the polynomials, some of which may be repeated, that make up a permutationally invariant group. In general, there will be as many groups as there were original polynomials. These groups, each with $n!$ (not necessarily unique) polynomial contributions, are then summed to form "superpolynomials" having permutational symmetry with respect to exchange of identical molecules.

The Mathematica programs for accomplishing these goals are found in the PolynomialPruningAddingVXX.x.wl package, where XX.x is the current version number. The programs for adding or pruning are PolynomialAdding[] and PolynomialPruning[], respectively. Inputs to these programs are in templates provided with the software.

D. Analytical Gradients

It is often very desirable to have analytical gradients of the potential so that forces between atoms can be calculated, either for use in the fitting process or, for example, in molecular dynamics simulations. There are several methods for accomplishing this goal, as outlined in the Introduction and described in detail elsewhere.[8] Here, we focus on the reverse (or backward) derivative technique.[56,57]

The method is perhaps easiest to understand by working through a very simple example, the case of a homonuclear diatomic molecule, presented previously[8] and here in Section S-I of the Supplementary Material. Of course, this example has only one internuclear distance and four polynomials (counting p(0)). The situation gets much more complicated for larger molecules. Taking the 22221111_4 basis for the water 4-body interaction as an example, more than 150 lines of Fortran code are required to define some reverse derivatives, even when the number of superpolynomials is pruned to 1000. Fortunately, there is a recursion relationship in the equations that makes the programming manageable.

As seen in the diatom example, the forward differentiation uses a series of steps to define the differentials of the internuclear distances, monomials, and polynomials, in the order $\boldsymbol{\tau}$, $\mathbf{m}$, and $\mathbf{p}$. The derivative of the potential is then given by $dV = \boldsymbol{c} \cdot d\boldsymbol{p}$. Let the number of the step in this forward direction be designated by i. We define the adjoint, $a(i)$, of a particular step in the reverse direction as the derivative of V with respect to the "conjugate" variable defined in step i of the forward calculation, where the conjugate variable is, for example, a τ, m or p variable. Consider a general case in which we have s steps in the forward direction defining the variables $z_i,\ i = 1...s$. These variables are conjugate to the adjoints $a_i,\ i = 1...s$, where $a_i = dV/dz_i$. At any particular step j, the potential calculated in the forward direction depends on the variables $z_1...z_j$, and the adjoint a_j depends on the variables $z_j...z_s$, so that $a_j = dV(z_j, ..., z_s)/dz_j$. We start with the differential for

$dV(z_j, ..., z_s)$ expressed as a sum of partial derivatives:

$$dV(z_j, ..., z_s) = \sum_{i=j}^{s} \frac{\partial V}{\partial z_i}\, dz_i. \tag{4}$$

Applying this equation to the definition of a_j we have

$$\begin{aligned} a_j = \frac{dV(z_j, ..., z_s)}{dz_j} &= \sum_{i=j}^{s} \frac{\partial V}{\partial z_i}\, \frac{dz_i}{dz_j} \\ &= \frac{\partial V}{\partial z_j} + \sum_{i=j+1}^{s} \frac{\partial V}{\partial z_i}\, \frac{dz_i}{dz_j} \\ &= \frac{\partial V}{\partial z_j} + \sum_{i=j+1}^{s} a_i\, \frac{dz_i}{dz_j}. \end{aligned} \tag{5}$$

Thus, each adjoint can be seen to be the sum of the partial derivative of V with respect to the conjugate variable to the adjoint plus the sum of all previous adjoints times the derivatives of their conjugate variables with respect to the conjugate variable of the current adjoint. At first, it may seem that we have pulled of a slight of hand in going to the last line in Eq. 5 from the one before it because a_i is defined as a total derivative rather than a partial one. However, when defined, a_i *was* a total derivative, but at this point we have added more variables, so it is correct to identify it here as a partial derivative. Note also that the first term in the last line of Eq. 5 is non-zero only if V depends directly on z_j. Because $dV = \boldsymbol{c}\cdot\boldsymbol{p}$, this term will exist only if z_j is a p polynomial, in which case $\frac{\partial V}{\partial z_j}$ will be equal to the corresponding coefficient c. The key Mathematica function that calculates the adjoints is provided in the Supplementary Material to ref. 8.

The Mathematica program for appending the reverse derivatives and for calculating the gradients is located in the FastDerivativesVXX.x.wl package, where XX.x is the current version number. Inputs to these programs are in a template provided with the software.

E. Program Details

More details concerning the software programs are contained in Section S-I of the Supplementary Material, including *a)* the diatomic example, *b)* a description of the MSA interface, *c)* answers to the question "Why Mathematica?", and *d)* a description of the Mathematica programs. A flowchart for the Mathematica Programs is shown in Figure 1.

III. RESULTS

A. Fragmentation: N-methyl acetamide (NMA) and Acetyacetone (AcAc)

The examples provided with the Mathematica software cover the fragmentation of N-methyl acetamine to either 2 or 3 fragments and the fragmentation of acetylacetone to 3 or 4 fragments. Table II shows some results for full (unfragmented), and 2-, 3-, and 5-fragment NMA as well as full (unfragmented) and 4-fragment AcAc.[44,49,54] The time listed in the Table is for performing 5000 energy calculations or $5000 \times 3N$ gradient component calculations on an i7 (2.7 GHz) processor. From the NMA example, we see that increasing the fragments from the full (unfragmented) case to 2, 3, or 5 fragments decreases substantially the number of polynomials and also the times for calculation, but at some cost to the accuracy; the cost is fairly minor for 2 fragments. For the AcAc case, using 4 fragments in place of the full (unfragmented) molecule actually increases the number of polynomials, but both decreases the time and increases the accuracy. This is because the full molecule used also the full symmetry (62222111), for which the polynomials become much more complicated to calculate. Note as well, that for all gradients the "normal" derivative method was used. These could be sped up substantially by using the reverse derivative method (See Reverse Derivatives subsection below).

TABLE II. Fragmentation Results for N-methyl acetamide (NMA) and acetylacetone (AcAc). All RMS energies are in cm^{-1} and gradients are in cm^{-1} $bohr^{-1}$. RMSE and RMSG are the root-mean-squared errors for the energies and gradients, respectively. The Time is for evaluating the energy or gradients for 5000 geometries.

Property	NMA *N=12*				AcAc *N=15*	
Fit Order	3	3	3	3	3	3
Frags	full	2	3	5	full	4
Morse vars	66	57	45	57	105	105
Polys	8040	5240	1806	1936	6207	24030
RMSE	26.8	34	149	93	49	22
RMSG	54.7	67	172	142	29	16
Time/s energy	0.298	0.110	0.037	0.041	0.95	0.43
Time/s gradients	10.72	3.97	1.346	1.482	80.91	26.61

B. Purification/Compaction/Grouping: 2-, 3-, and 4-Body Interaction Potentials for Water Clusters

The n-body bases for water clusters form an interesting of examples and tests of the purification, compaction, and grouping steps. Table III indicates the effect of the three steps on the number of monomials and polynomials for 2-, 3-, and 4-body potentials for water clusters, as well as the starting symmetry, order, and numbers. The letter $\boldsymbol{m}$ indicates the number of monomials, $\boldsymbol{p}$ indicates the number of polynomials with coefficients, and $\boldsymbol{q}$ indicates the number of other polynomials which, while needed for the calculation, are not assigned coefficients (i.e., they are needed only in order to calculate the p polynomials). We start with the number of m, p, q given

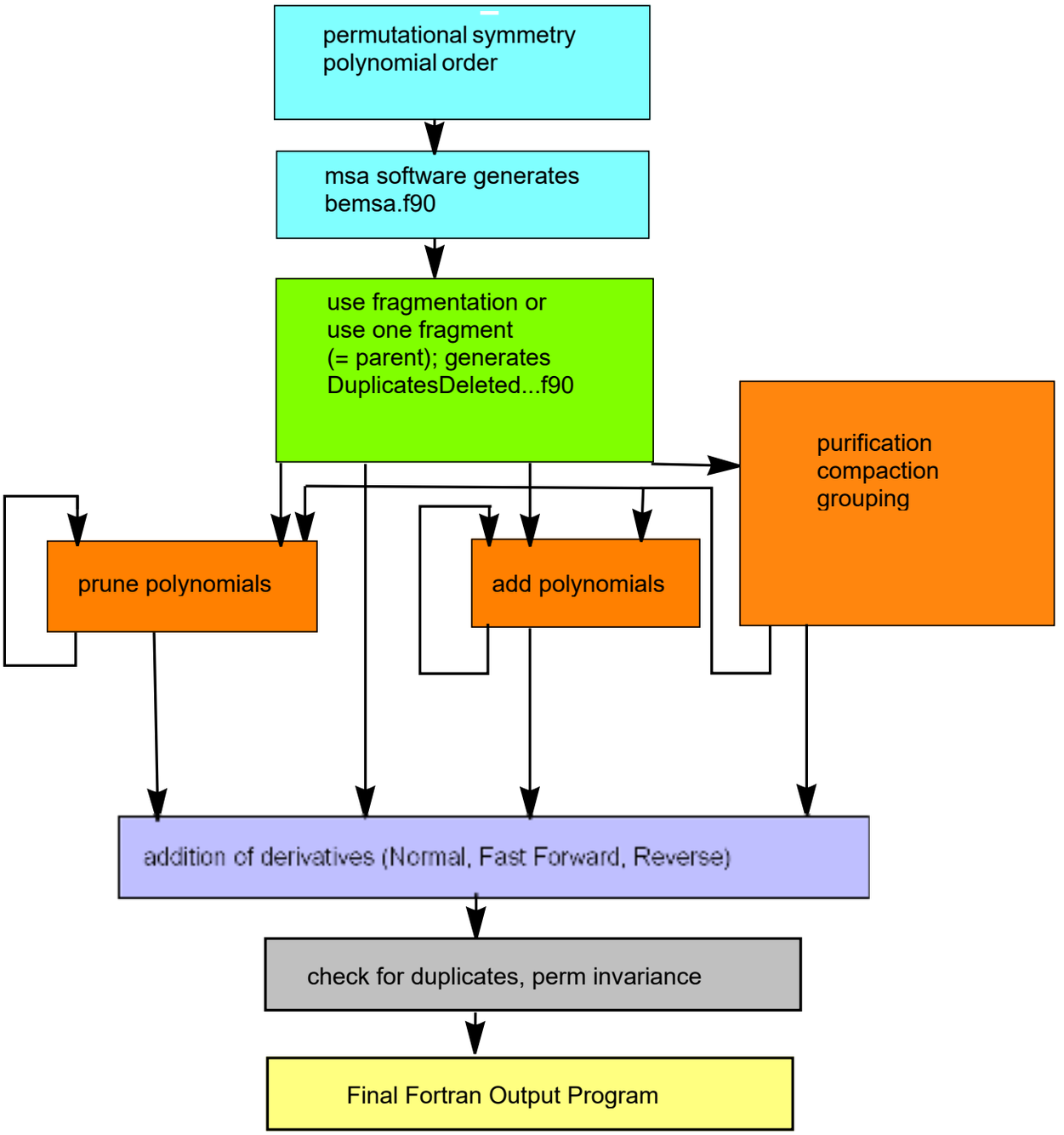


FIG. 1. Flow Diagram to Create PIPs Using MSA and Mathematica Programs

by the MSA software for the indicated symmetry and order. The purification step examines the polynomials and decides which ones do not have the correct zero interaction at large distance. These polynomials, now renamed q, might still needed to calculate the p polynomials. The compaction step determines which monomials and q polynomials actually are necessary and drops the rest. The grouping step decides which of the p polynomials are in permutationally invariant groups with respect to monomer exchange, makes new superpolynomials by summing those polynomials in each group, and moves the polynomials which were summed to the q column. The final number of polynomials are very efficient at fitting the data base, and the data base does not need to

be replicated for permutations – the monomer permutational symmetry is built into the basis set. Determination of all three of the water-cluster bases are examples in the TemplatesVXX.x.nb notebook. These bases are neither the most accurate nor the fastest that we found, but they serve as good examples and are calculated rapidly. We have also fit these bases to the appropriate CCSD(T) data sets. Figure 2 shows the results, where the time is that for Mathematica to evaluate the basis set; the Fortran code that is produced is orders of magnitude faster.

TABLE III. Number of monomials and polynomials as a function of Purification, Compaction, and Grouping for 2-, 3-, and 4-body water clusters starting with indicated symmetries and orders. Letter $\boldsymbol{m}$ indicates monomials, $\boldsymbol{p}$ indicates polynomials with coefficients, and $\boldsymbol{q}$ indicates polynomials needed for final calculation but not assigned coefficients.

Step \ number	$\boldsymbol{m}$	$\boldsymbol{p}$	$\boldsymbol{q}$
2-body 2211_5			
Start	194	4616	0
Purify	194	4406	210
Compact	194	4406	102
Grouped	194	2347	4508
3-body 222111_4			
Start	2732	18 979	0
Purify	2732	13,229	5750
Compact	2573	13,229	983
Grouped	2573	2292	14212
4-body 22221111_3			
Start	2910	10 736	0
Purify	2910	1648	9088
Compact	930	1648	279
Grouped	830	872	1927

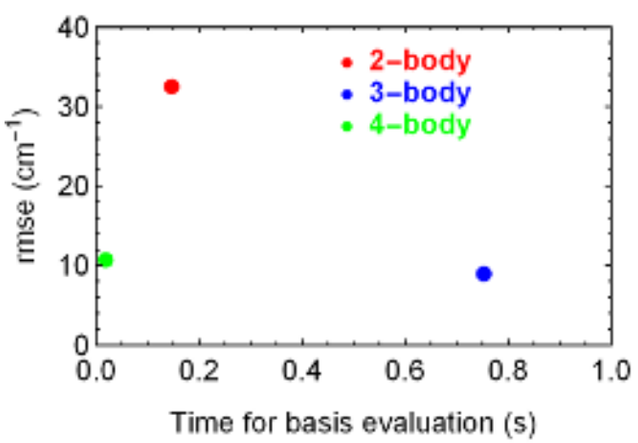


FIG. 2. Root mean squared error and time for Mathematica to evaluate the basis for fits to the 2-, 3-, and 4-body data sets shown in Table III.

C. Pruning and Adding: Glycine, Water Clusters

The introductory remarks mentioned that we pruned and added polynomials to fragmented bases for glycine (N=10). Figure 3 shows the results as a function of the number of polynomials, including the rmse values for energies and gradients and the time for evaluation of 3000 energies or 3000 $\times(3N = 30)$ gradients (using forward differentiation). The starting point was the 4th-order glycine basis with 9337 polynomials. The data set to which the basis was fit had 3100 energies and $3100 \times 30 = 93\,000$ gradient components. Bases to the right of 9337 on the abscissa were produced by addition, whereas those to the left were produced by pruning.

It is interesting to note that the addition of ca. 700-1700 polynomials actually improves the rmse for both energies and gradients substantially, more than one might expect from the trend of values for lower numbers of polynomials. The reason for this is that the added polynomials can sometimes be more important than the original ones, even though they are generally of higher order. The method could probably be improved even further by swapping out the old polynomials with smaller polynomial values for the new ones that have large polynomial values, and then adding from either group until the desired number is attained.

The water 4-body data sets, including the 3rd-order one described in the previous section along with a 4th-order one, provide another good example of how one can prune polynomials so as to get any desired trade-off between accuracy and speed. The 4th-order data set has 2342 polynomials that are permutationally invariant to water exchange. The 3rd-order data set has 87. By pruning the 4th-order set to 1000, 500, 200, 100, 30, and 15 polynomials and the 3rd order data set to 70, we were able to span a large range of rmse values and times. Figure 4 shows a large fraction of the results. (The full, 4th-order result with an rmse of 0.7 cm^{-1} and a time of 4 s, is off-scale to the right and not shown). The rmse for the energy is plotted in red against the left axis as a function of the number of superpolynomials, whereas the time in seconds for Mathematica to evaluate the basis is plotted in blue against the right axis, also as a function of the number of superpolynomials. The trade-off between speed and accuracy is readily apparent. Fortunately, one has many options to choose from.

D. Reverse Derivatives: Water 3-body and Ethanol

The examples given in the accompanying Mathematica software cover the addition of the reverse derivatives for gradient calculations for both ethanol[58,59] and for the water 3-body interaction, both nine-atom systems. The reverse method may be compared to others by normalizing the time to calculate the $3N$ gradients by the time to calculate the energy. Figure 5 compares the reverse method with three other methods, the 2-point finite difference

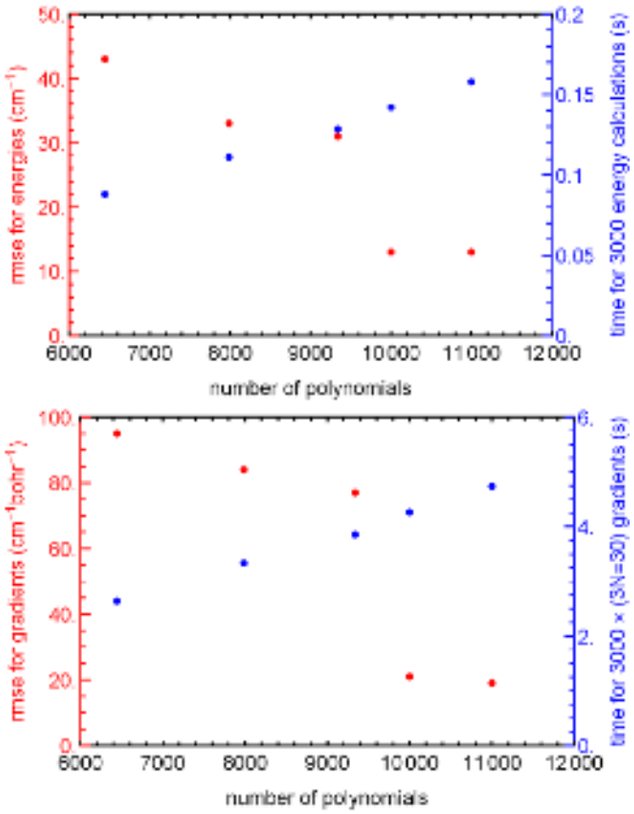


FIG. 3. Root mean squared error (red, left axis) for energies (top panel) and gradients (bottom panel) and time (blue, right axis) for 3000 evaluations on an i7 (2.7GHz) processor as a function of the number of polynomials for a three-fragment 4th-order Glycine basis with pruning and adding. The original basis had 9337 polynomials.

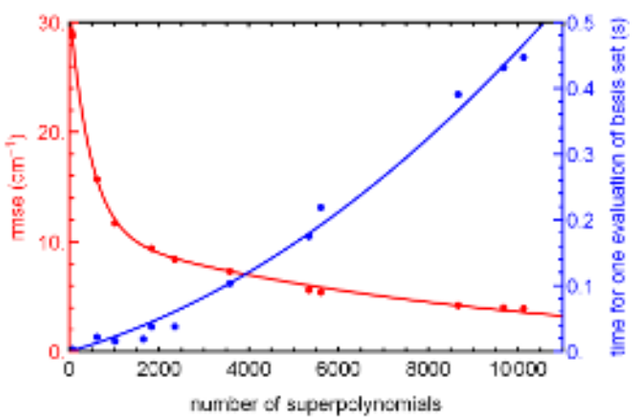


FIG. 4. Root mean squared error (red, left axis) and time for Mathematica to evaluate the basis (blue, right axis) as a function of the number of superpolynomials for pruned 22221111 3rd and 4th order bases.

method, the "normal" forward derivative method, and a "fast forward" derivative method. The "normal" method simply takes the differential of each step in the calculation of the energy and then uses the equation $dV = \boldsymbol{c} \cdot d\boldsymbol{p}$ to calculate the gradients. The time for this method can be estimated as about $3N$ times that for the energy because the forward direction must be computed once for each derivative with respect to a Cartesian coordinate; it actually scales a bit worse than that, about $3.7N$, as shown in the figure. The 2-point finite difference method should scale as about $6N$ because two calculations of the energy are needed for each gradient component; the figure shows that it actually scales as about $5.8N$. The "fast forward" method has been described in detail;[8] it scales between the "normal" method and the reverse method at about $1.7N$. The reverse method scales as about 3.6 times the time for the energy and is *independent* of N, at least in the range shown. The molecules used for these calculations are the water 2-body interaction (6 atoms), ethanol (9 atoms), the water 4-body interaction (12 atoms), and tropolone (15 atoms).

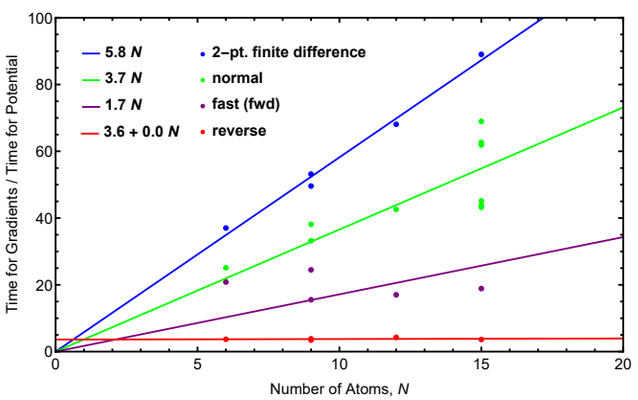


FIG. 5. The time for calculating $3N$ gradients relative to the time for calculating the energy for various methods as a function of the number of atoms N. (Reproduced from Houston et al., J. Chem. Phys. **156**, 044120 (2022), with the permission of AIP Publishing.)

IV. DISCUSSION

We now list and discuss some of the common themes that run through the methods presented above.

1. *The potential depends most on local atoms and data points.*

Although he did not originate the phrase, the American politician and former Speaker of the U. S. House of Representatives, Tip O'Neill, was closely associated with saying, "All politics is local." We might paraphrase him by saying, "All potentials are local." They are local in two senses of the word.

One theme that emerges from the methods used above is that the potential depends most on the atoms pairs

that are closest to one another; this is no surprise because all atomic interactions fall off at long distances. Fragmentation works by recognizing this aspect of the potential and preferentially neglecting internuclear distances that are long and also neglecting the polynomials that have small values because they depend on these distances. The methods of polynomial addition and pruning are also based on this idea – if internuclear distances are never close among all the geometries in the data set, then the polynomials that depend on them will be small and can be discarded, at least relative to those polynomials that have larger values. So the potential is local in the sense that it depends on which atoms are local.

It is local in a different sense as well. While the fit itself might be global, the fitted potential near any point is most influenced by the potentials at points closest to it, as opposed to those far away. For most of this article we have assumed that the goal is to produce a global potential energy surface. A global surface is one in which the addition of another geometry with energy and/or gradients is treated mathematically in the calculation of the surface in a way that is not specific to the location of addition.[60] For a local surface, by contrast, the mathematical influence of an additional geometry typically varies with location, with nearby points structured by the mathematical fitting to have more influence on the fitting of the new point than distant ones. There are many advantages to having a global surface: its evaluation is computationally efficient, it is easy to improve it with additional points, it is relatively easy to include analytic gradients, and once a global surface is found, the hard work (except for evaluating it at an arbitrary point) is over; it needs to be done only once. On the other hand, there are disadvantages. A global surface is less accurate for specific locations than a local fit; it takes far more parameters and basis functions to accomplish a fit; the fitting program must deal with much larger matrices, because of the needs for both a larger data set and a larger basis set; and, at least without including weighting, global fits to very high energy points have a disproportionate influence on the fits to points at low energy. Nor is the scaling very favorable:[27] in the absence of reduction methods such as permutation invariance and those described herein, the cost scales approximately as the total number of terms, $N_t = (d+o)!/d!/o!$, where d is the number of internuclear distances, $N(N-1)/2$, with N the number of atoms, and o is the maximum polynomial order. This results is an initial exponential rise in N_t with N. Two questions thus arise. As N becomes large, is a global surface affordable? And, given that several of our methods for reducing the complexity are based on neglecting polynomials that depend on large distances, might not a local potential approach be as good or better? A thoughtful article is an excellent starting place for the discussion.

Bender and colleagues[60] have developed a local potential method with a long descriptive name: statistically localized, permutationally invariant, local interpolating moving least squares method for fitting the many-body problem (or SL-PI-L-IMLS-MP). It is heavily based on previous work by Dawes and colleagues,[61–63] by Guo et al.[64], by Ischtwan and Collins,[65] and by Maisuradze et al.[66,67] Incorporation of gradients for moving least squares methods has been considered by Tokmakov et al.[68] Much of the fundamental insight for the moving least squares methods is from previous work in the field of mathematics.[69,70]

A brief description of the method is as follows. For each point in the data set, a weighted local fit of that point and points in the neighboring vicinity is performed in a "construction step". The weighting, with weights $\boldsymbol{\omega}$, defines the local neighborhood. In the "evaluation step", when the potential at a new point is desired, the new potential is calculated as a weighted average, with potentially different weights $\boldsymbol{w}$, of the local potentials previously calculated in the construction step. The weights are chosen such that they go rapidly to zero at a radius, R, describing a hypersphere around any chosen point; this ensures that only a small number of local fits need to be considered for any calculation of the potential. The radius R is chosen to vary as a function of the local density of data points.

The paper by Bender et al. then applies this method to the construction of a six-dimensional potential for N_2-N_2 interactions. The fit has high fidelity, typically less than a few tenths of a percent error, which is quite good considering that the number of polynomials for the local fits is just 83. The data set contained 13 148 geometries. Less than 5% of the 13 148 available fits were typically used in any potential evaluation. The implementation of the method brings out two complications, neither insurmountable. First, there is a strong reason to use PIPs as the basis set, but the method requires that the weights, as well as the polynomials, have permutational symmetry. Second, the method makes it more complicated to calculate analytic gradients because they involve derivatives of the weight functions as well as of the polynomials. What seems lacking at this stage is an application to a much larger molecular system. Whether or not the method of Bender et al.[60] is the best direction in which to move for large molecules, it does appear that more thought will need to be given to local fitting methods.

2. *There is a close connection between the basis and the data set; ignoring it leads to inefficiencies.*

The connection is two-way. The data set must be dense enough and broad enough to be fit smoothly and accurately by the basis set; importantly, the data set must provide substantially more constraints than coefficients needed in the basis so as to avoid overfitting. On the other hand, the polynomials used in the basis should be prioritized by examining the data set. If certain internuclear distances are always large in the data set, it makes sense to omit those polynomials which depend strongly on them and which consequently have small values. Finally, both the data set and the basis should be chosen with an eye to the chemical problem being addressed.

In the case of large molecules, many data sets are too small and span too narrow an energy range. We have argued elsewhere that for general use, especially for quantum calculations, the data set should have energies at least 1.5 times as large as the anticipated harmonic zero-point energy.[37] In addition, the density of points should be increased around all stationary points. Because it is unlikely that these points can be specified before a preliminary PES has been calculated, it is clear that creating a good PES is an iterative process, often calling for new data or a different basis as it is developed.

Finally, as we attempt to calculate PESs for larger and larger molecules, the sizes of both the data set and the basis must increase if we wish to maintain chemical accuracy. On the other hand, it may be that the chemical process of interest, for example, an isomerization, conformer change, or hydrogen transfer, depends mostly on a fairly local barrier. Even if one wanted a global PES, it might not be necessary to maintain the same accuracy in other parts of the surface as one might need near the barrier.

3. *Usefulness of gradients.*

A third theme that runs through the methods described above is the usefulness of gradients, particularly if they can be calculated at small cost. Analytical gradients are very useful in fitting the PES when electronic structure calculations can produce *ab initio* values because they reduce the number of data set geometries needed by increasing the number of constraints for each point. They are certainly valuable if the goal of the PES is to allow classical or quantum molecular dynamics calculations. Furthermore, they allow for rapid location on the surface of stationary points, which are often of the highest chemical interest.

One question is how gradients may be useful in determining the fit, beyond providing the further constraints just mentioned. Tokmakov et al. have considered their use in moving least-squares methods for local fitting of potential energy surfaces.[68] As might be expected, their inclusion reduces the number of data set points required, often by as much as 40%. Formulae for incorporating weighted fitting of energies and gradients are provided. Another direction to investigate might be whether using local analytical gradients might improve the accuracy of the calculation of the potential at an arbitrary geometry, even in a global fit.

We additionally note that a combination of reverse analytical gradients with finite difference numerical methods can provide the Hessian for any point on the surface by using two calls to the gradient as a function of one variable for symmetrically displaced values of the second variable. The finite difference between the two results gives a full row of the Hessian at a cost of less than 8 times the cost of an energy calculation. When repeated for all of the second variables, the full Hessian costs less than $24N$ times the cost of an energy calculation, whereas a fully numerical finite difference method would require $(2\times3N)^2 = 36N^2$ times the cost of an energy calculation. Thus the speed-up factor gained by using the reverse gradient method is about 1.5 N. The program supplied in the Supplementary Material automatically adds a Hessian subroutine based on this algorithm to the Fortran output whenever the reverse derivative gradients are requested.

4. *Long-range behavior*

As discussed in detail above, purification enforces rigorous separability of fragments in the asymptotic region. This is necessary but perhaps not a sufficient condition to describe the long-range behavior of any ML PES. The details of the long-range behavior need to be considered and ideally incorporated precisely into the PES. The long-range behavior of two molecules is formally known from the multipole expansion, e.g., dipole-dipole, dipole-quadrupole, etc. So, it would seem advantageous to make use of these analytical interactions in a global PES. For example, this could be done using a switching function that switches to the analytical long-range interactions at some suitable separation distance. Of course for this to be practical, the dipole moment, polarizability, etc. of the separated fragments must be known, and known as a function of the internal coordinates of each fragment. One example from our group where this was done is the water dimer PES.[71] In this case an accurate, *ab initio* dipole moment surface for the flexible water monomer was available[72] and used to describe the dipole-dipole interaction in the long range via a switching function. In general though, monomer properties such as dipole and polarizability are not known and not computationally easy to obtain. Therefore, we don't follow the analytical approach; instead we perform a separate, precise fit to CCSD(T) energies in the long range and then switch to this fit. The fit in the long-range uses a larger Morse range parameter, e.g., 6-8 bohr,[73–75] and the RMS fitting error is typcially less than 0.1 cm^{-1}.

A final editorial comment on this is that the long-range behavior of general MLPs has not received as much attention in the literature as it should.

V. CONCLUSIONS

The results presented above as well as those discussed above from the literature show several ways to make PIP bases more efficient.

Fragmentation has already been applied to molecules with 15 atoms, and shows promise for extending the number of atoms further. The related methods of adding or pruning polynomials so as to keep the most important ones also holds similar promise, as does the "greedy" method of Moberg and colleagues.[27]

The method we call purification, which is equivalent to deletion of unconnected terms,[26,36] appear to be have the same purpose, namely to remove spurious interfragment coupling in the separated fragment limit.

Reverse derivatives[8,56,57] make calculation of gradients much faster, particularly for molecular dynamics calcu-

lations and the calculation of Hessians.

Common themes from this article suggest paths for future research. These include expanding the fragmented basis approach to describe larger molecules and asymptotic dissociation, further optimization of the Δ-ML process especially for larger molecules for which standard CCSD(T) calculations become infeasible, exploration of local fitting techniques for more sizeable molecules, more extensive use of fast (reverse) gradients and Hessians, and better fitting of the long-range potential behavior.

VI. AUTHOR'S CONTRIBUTIONS

The Mathematica software was written by PLH with development support from RC and CQ. The MSA interface was written by JKL and CQ. AN, CQ, RC, PLH, QY, and JMB applied these techniques to develop PESs for the molecules listed in Table I. JMB designed and led the research. All authors contributed to writing the manuscript.

VII. ACKNOWLEDGEMENTS

JMB thanks the ARO, DURIP grant (W911NF-14-1-0471), for funding a computer cluster where most of the calculations were performed and current financial support from NASA (80NSSC20K0360). QY thanks Professor Sharon Hammes-Schiffer and National Science Foundation (Grant No. CHE-1954348) for support. RC thanks Università degli Studi di Milano ("PSR, Azione A Linea 2 - Fondi Giovani Ricercatori") for support.

VIII. DATA AVAILABILITY

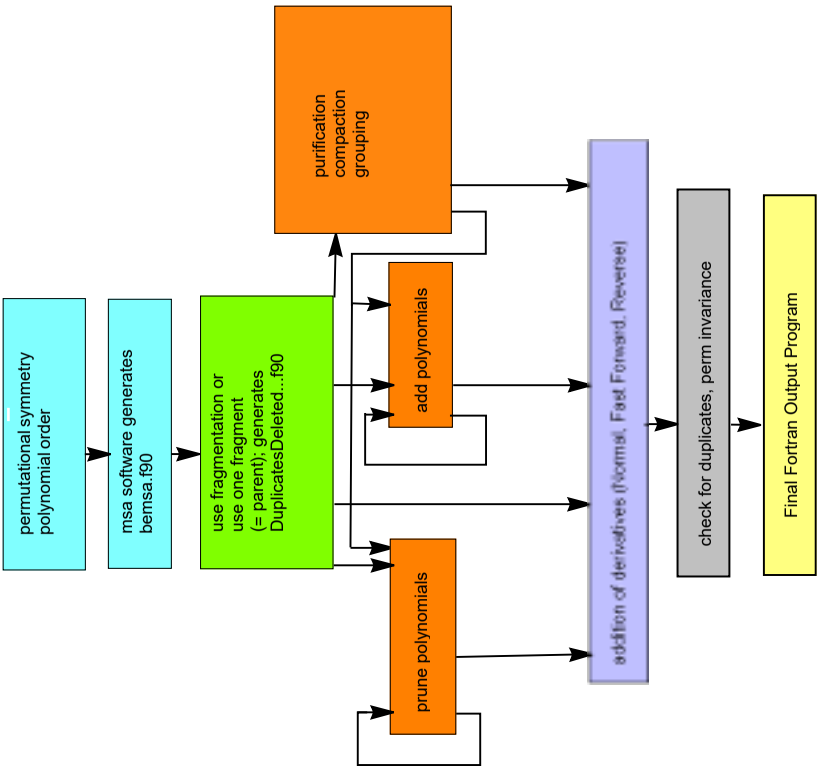
Databases of electronic energies and gradients when available for the molecules considered here, as well as many others, are provided at https://github.com/jmbowma/QM-22.

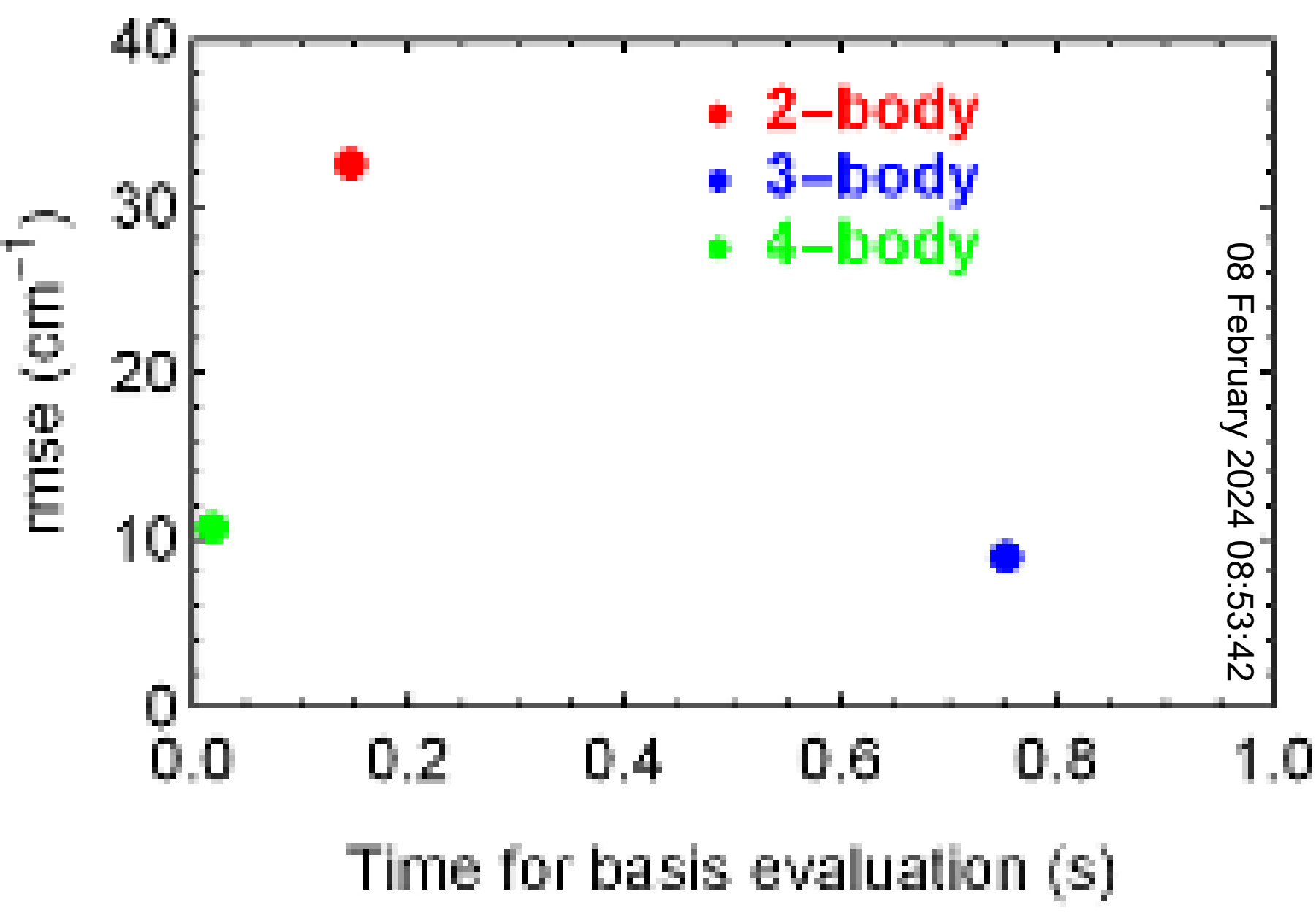
IX. SUPPLEMENTARY MATERIAL

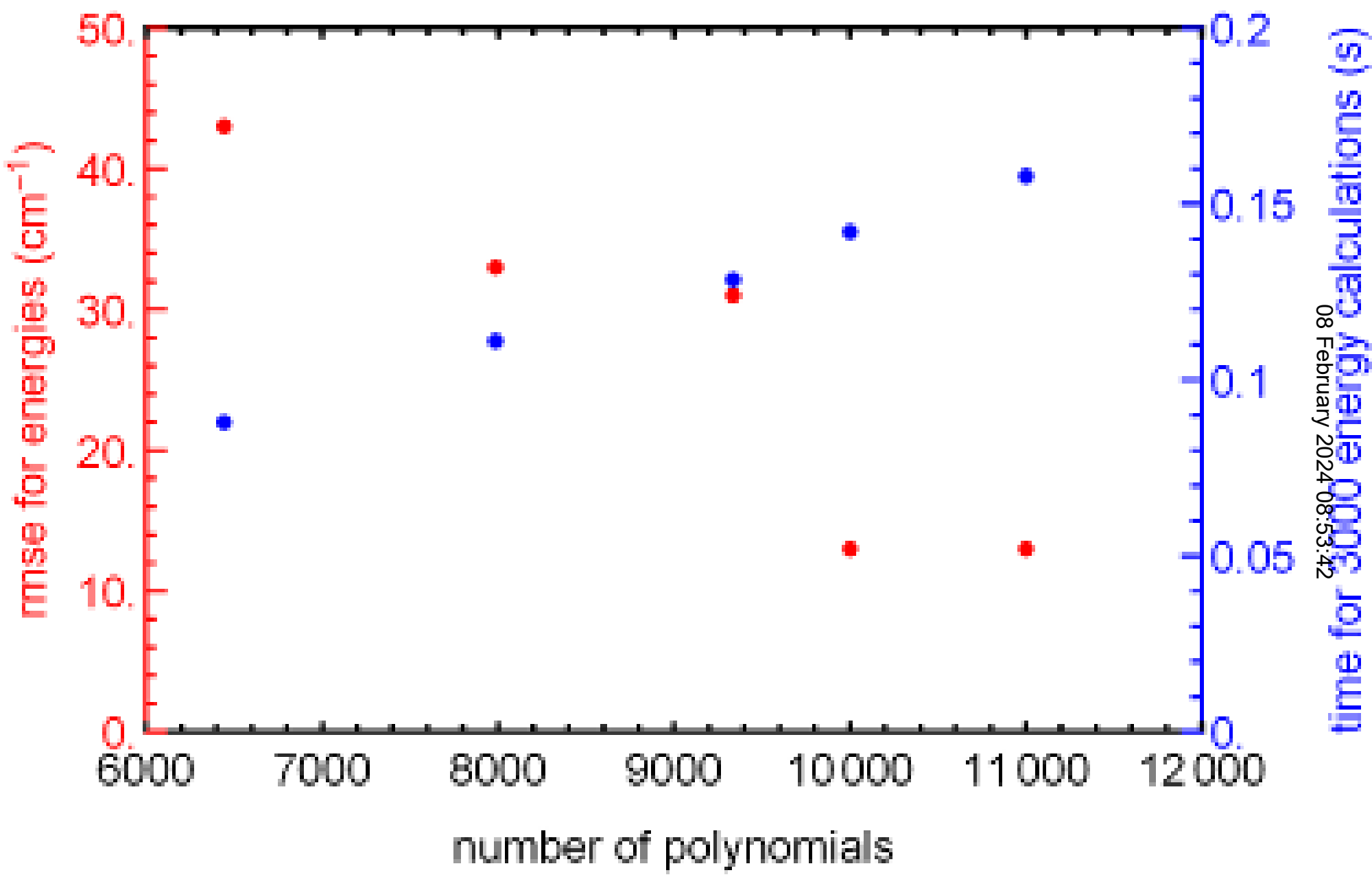
The SM contains a section with descriptions of the programs. Also provided are compressed folders containing 1) the MSA GUI interface for generating MSA PIP output and using it to fit electronic structure results, and 2) the Mathematica Notebooks and functions, with examples, for the techniques described above.

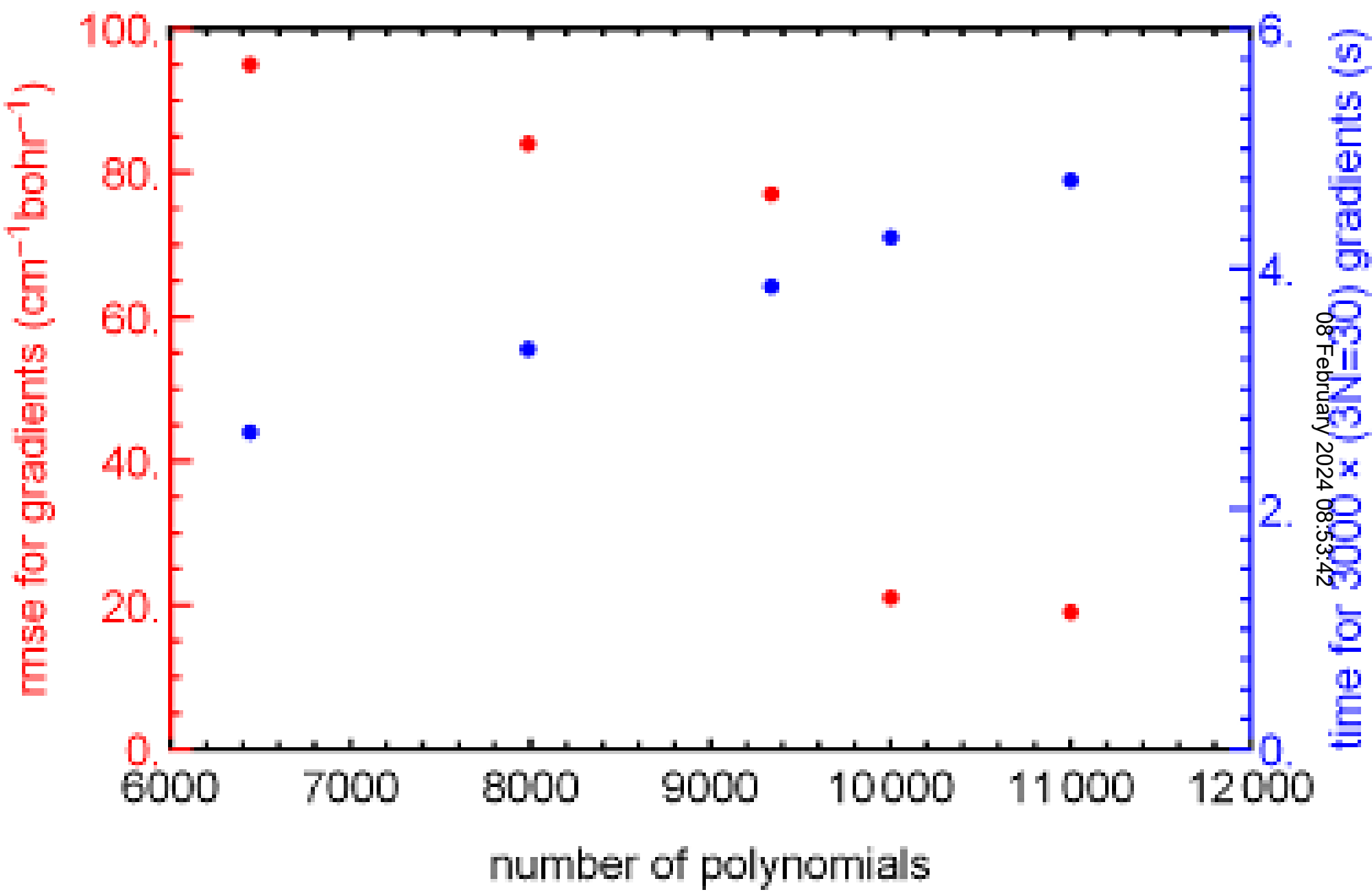
[1] J. Behler, The J. Chem. Phys. **145**, 170901 (2016).
[2] J. Westermayr, M. Gastegger, K. T. Schütt, and R. J. Maurer, J. Chem. Phys. **154**, 230903 (2021).
[3] D. Koner, S. M. Salehi, P. Mondal, and M. Meuwly, J. Chem. Phys. **153**, 010901 (2020).
[4] T. Fröhlking, M. Bernetti, N. Calonaci, and G. Bussi, J. Chem. Phys. **152**, 230902 (2020).
[5] T. Mueller, A. Hernandez, and C. Wang, J. Chem. Phys. **152**, 050902 (2020).
[6] C. Qu, Q. Yu, B. L. Van Hoozen, J. M. Bowman, and R. A. Vargas-Hernández, J. Chem. Theory Comput. **14**, 3381 (2018).
[7] J. Li, B. Jiang, and H. Guo, J. Chem. Phys. **139**, 204103 (2013).
[8] P. L. Houston, C. Qu, A. Nandi, R. Conte, Q. Yu, and J. M. Bowman, J. Chem. Phys. **156**, 044120 (2022).
[9] A. Nandi, C. Qu, P. L. Houston, R. Conte, and J. M. Bowman, J. Chem. Phys. **154**, 051102 (2021).
[10] J. S. Smith, B. T. Nebgen, R. Zubatyuk, N. Lubbers, C. Devereux, K. Barros, S. Tretiak, O. Isayev, and A. E. Roitberg, Nat. Commun. **10**, 2903 (2019).
[11] B. J. Braams and J. M. Bowman, Int. Rev. Phys. Chem. **28**, 577 (2009).
[12] "Msa software with gradients," `https://github.com/szquchen/MSA-2.0` (2019), accessed: 2019-01-20.
[13] B. Jiang, J. Li, and H. Guo, J. Phys. Chem. Lett. **11**, 5120 (2020).
[14] B. Jiang, J. Li, and H. Guo, Int. Rev. Phys. Chem. **35**, 479 (2016).
[15] R. Dawes and S. A. Ndengué, Int. Rev. Phys. Chem. **35**, 441 (2016).
[16] V. Babin, C. Leforestier, and F. Paesani, J. Chem. Theory Comput. **9**, 5395 (2013).
[17] S. K. Reddy, S. C. Straight, P. Bajaj, C. Huy Pham, M. Riera, D. R. Moberg, M. A. Morales, C. Knight, A. W. Götz, and F. Paesani, J. Chem. Phys. **145**, 194504 (2016).
[18] N. M. Kidwell, H. Li, X. Wang, J. M. Bowman, and M. I. Lester, Nat. Chem. **8**, 509 (2016).
[19] A. W. Jasper, L. B. Harding, C. Knight, and Y. Georgievskii, J. Phys. Chem. A **123**, 6210 (2019).
[20] C. Qu, Q. Yu, and J. M. Bowman, Annu. Rev. Phys. Chem. **69**, 6.1 (2018).
[21] "Original msa software," `https://www.mcs.anl.gov/research/projects/msa/` (2019), accessed: 2019-12-20.
[22] Z. Xie and J. M. Bowman, J. Chem. Theory Comput. **6**, 26 (2010).
[23] V. Babin, G. R. Medders, and F. Paesani, J. Chem. Theory Comput. **10**, 1599 (2014).
[24] M. Riera, E. P. Yeh, and F. Paesani, J. Chem. Theory Comput. **16**, 2246 (2020).
[25] T. Győri and G. Czakó, J. Comput. Theory Chem. **16**, 51 (2020), pMID: 31851508.
[26] D. R. Moberg and A. W. Jasper, J. Chem. Theory Comput. **17**, 5440 (2021).
[27] D. R. Moberg, A. W. Jasper, and M. J. Davis, J. Phys. Chem. Lett. **12**, 9169 (2021), pMID: 34525799.
[28] A. S. Abbott, J. M. Turney, B. Zhang, D. G. A. Smith, D. Altarawy, and H. F. Schaefer, J. Chem. Theory Comput. **15**, 4386 (2019).
[29] D. Koner and M. Meuwly, J. Chem. Theory Comput. **16**, 5474 (2020).
[30] S. Carter, S. J. Culik, and J. M. Bowman, J. Chem. Phys. **107**, 10458 (1997).
[31] J. M. Bowman, S. Carter, and X. Huang, Int. Rev. Phys. Chem. **22**, 533 (2003).
[32] Y. Shu, J. Kryven, A. G. Sampaio de Oliveira-Filho, L. Zhang, G.-L. Song, S. L. Li, R. Meana-Pañeda, B. Fu, J. M. Bowman, and D. G. Truhlar, J. Chem. Phys. **151**, 104311 (2019).
[33] J. Li, Z. Varga, D. G. Truhlar, and H. Guo, J. Chem. Theory Comput. **16**, 4822 (2020).
[34] Z. Varga and D. G. Truhlar, Phys. Chem. Chem. Phys. **23**, 26273 (2021).
[35] Z. Varga, Y. Liu, J. Li, Y. Paukku, H. Guo, and D. G. Truhlar, J. Chem. Phys. **154**, 084304 (2021).
[36] Y. Paukku, K. R. Yang, Z. Varga, and D. G. Truhlar, J. Chem. Phys. **139**, 044309 (2013).
[37] J. M. Bowman, C. Qu, R. Conte, A. Nandi, P. L. Houston, and Q. Yu, J. Chem. Phys. **156**, 240901 (2022).

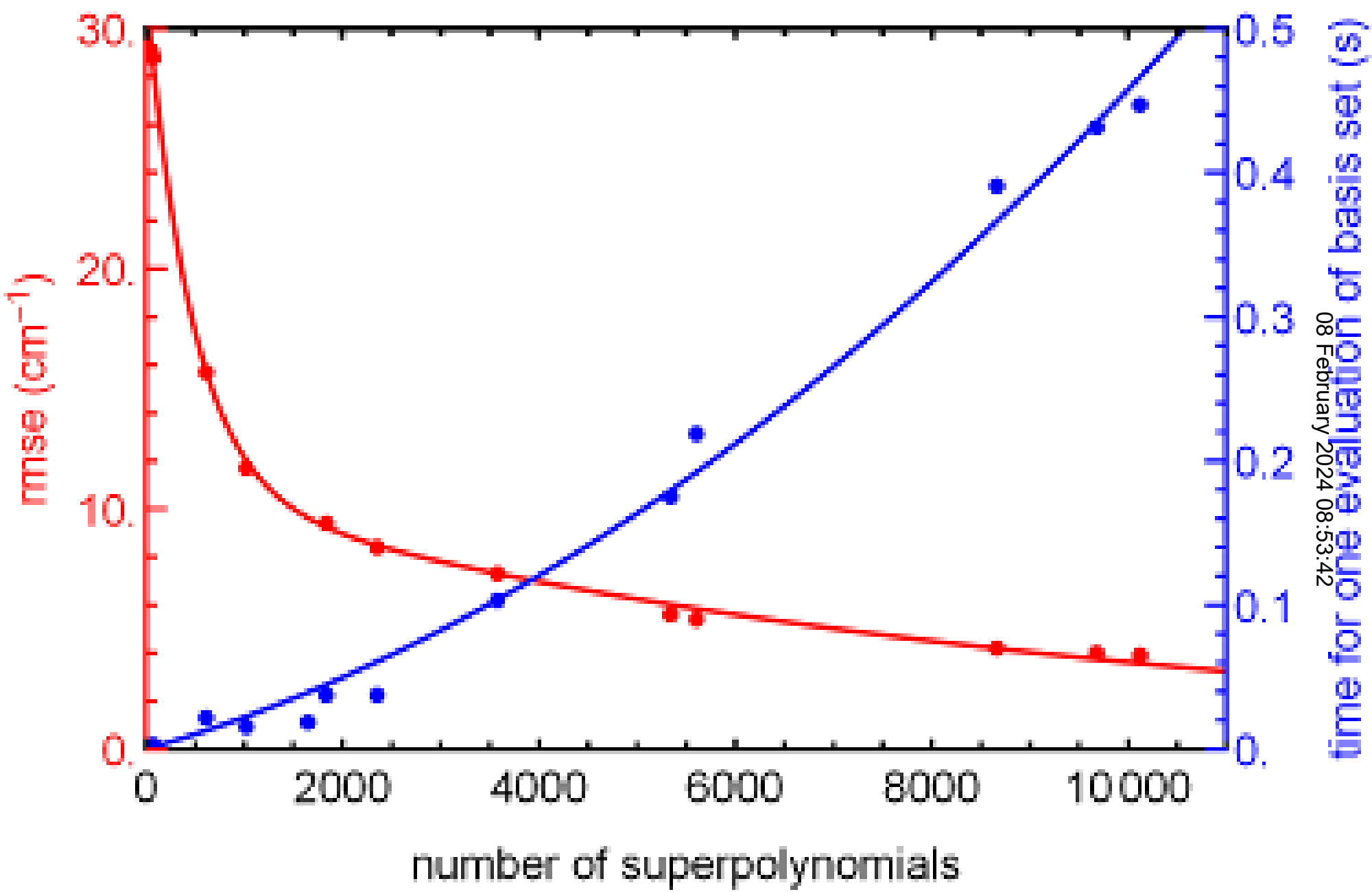
[38]Wolfram Research Inc., "Mathematica, Version 12.0," (2019), champaign, IL, 2019.
[39]R. Conte, P. L. Houston, and J. M. Bowman, J. Chem. Phys. **140**, 151101 (2014).
[40]C. Qu, R. Conte, P. L. Houston, and J. M. Bowman, Phys. Chem. Chem. Phys. **17**, 8172 (2015).
[41]Z. Homayoon, R. Conte, C. Qu, and J. M. Bowman, J. Chem. Phys. **143**, 084302 (2015).
[42]Q. Wang and J. M. Bowman, J. Chem. Phys. **147**, 161714 (2017).
[43]A. Nandi, C. Qu, and J. M. Bowman, J. Chem. Phys. **151**, 084306 (2019).
[44]R. Conte, C. Qu, P. L. Houston, and J. M. Bowman, J. Chem. Theory Comput. **16**, 3264 (2020).
[45]A. Nandi, C. Qu, and J. M. Bowman, J. Chem. Theory Comp. **15**, 2826 (2019).
[46]R. Conte, P. L. Houston, C. Qu, J. Li, and J. M. Bowman, J. Chem. Phys. **153**, 244301:1 (2020).
[47]P. L. Houston, R. Conte, C. Qu, and J. M. Bowman, J. Chem. Phys. **153**, 024107 (2020).
[48]C. Qu, P. L. Houston, R. Conte, A. Nandi, and J. M. Bowman, J. Phys. Chem. Lett. **12**, 4902 (2021).
[49]C. Qu, R. Conte, P. L. Houston, and J. M. Bowman, Phys. Chem. Chem. Phys. **23**, 7758 (2021).
[50]A. Nandi, C. Qu, P. L. Houston, R. Conte, Q. Yu, and J. M. Bowman, J. Phys. Chem. Lett. **12**, 10318 (2021).
[51]S. Li, W. Li, and J. Ma, Acc. Chem. Res. **47**, 2712 (2014).
[52]X. He, T. Zhu, X. Wang, J. Liu, and J. Z. H. Zhang, Acc. Chem. Res. **47**, 2748 (2014).
[53]M. A. Collins, M. W. Cvitkovic, and R. P. A. Bettens, Acc. Chem. Res. **47**, 2776 (2014).
[54]C. Qu and J. M. Bowman, J. Chem. Phys. **150**, 141101 (2019).
[55]R. Conte, C. Qu, and J. M. Bowman, J. Chem. Theory Comput. **11**, 1631 (2015).
[56]A. G. Baydin and B. A. Pearlmutter, JMLR: Workshop and Conference Proceedings, ICML 2014 AutoML Workshop , 1 (2014).
[57]A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, J. Mach. Learn. Res. **18**, 5595–5637 (2017).
[58]A. Nandi, R. Conte, C. Qu, P. L. Houston, Q. Yu, and J. M. Bowman, J. Chem. Theory Comput. **18**, 5527 (2022).
[59]R. Conte, A. Nandi, C. Qu, Q. Yu, P. L. Houston, and J. M. Bowman, J. Phys. Chem. A **126**, 7709 (2022).
[60]J. D. Bender, S. Doraiswamy, D. G. Truhlar, and G. V. Candler, J. Chem. Phys. **140**, 054302 (2014).
[61]R. Dawes, D. L. Thompson, Y. Guo, A. F. Wagner, and M. Minkoff, J. Chem. Phys. **126**, 184108 (2007).
[62]R. Dawes, D. L. Thompson, A. F. Wagner, and M. Minkoff, J. Chem. Phys. **128**, 084107 (2008).
[63]R. Dawes, A. F. Wagner, and D. L. Thompson, J. Phys. Chem A **113**, 4709 (2009).
[64]Y. Guo, I. Tokmakov, D. L. Thompson, A. F. Wagner, and M. Minkoff, J. Chem. Phys. **127**, 214106 (2007).
[65]J. Ischtwan and M. A. Collins, J. Chem. Phys. **100**, 8080 (1994).
[66]G. G. Maisuradze and D. L. Thompson, J. Phys. Chem A **107**, 7118 (2003).
[67]G. G. Maisuradze, D. L. Thompson, A. F. Wagner, and M. Minkoff, J. Chem. Phys. **119**, 10002 (2003).
[68]I. V. Tokmakov, A. F. Wagner, M. Minkoff, and D. L. Thompson, Theor. Chem. Acc. **118**, 755 (2007).
[69]P. Lancaster and K. Šalkauskas, Math. Comput. **37**, 141 (1981).
[70]P. Lancaster and K. Šalkauskas, *Curve and Surface Fitting: An Introduction* (Academic Press, London, 1986).
[71]Q. Yu, C. Qu, P. L. Houston, R. Conte, A. Nandi, and J. M. Bowman, J. Phys. Chem. Lett. **13**, 5068 (2022).
[72]L. Lodi, J. Tennyson, and O. L. Polyansky, J. Chem. Phys. **135**, 034113 (2011).
[73]B. Yang, P. Zhang, C. Qu, X. H. Wang, P. C. Stancil, J. M. Bowman, N. Balakrishnan, B. M. McLaughlin, and R. C. Forrey, J. Phys. Chem. A **122**, 1511 (2018).
[74]B. Yang, P. Zhang, C. Qu, P. C. Stancil, J. M. Bowman, N. Balakrishnan, and R. C. Forrey, Phys. Chem. Chem. Phys. **20**, 28425 (2018).
[75]B. Yang, P. Zhang, C. Qu, P. Stancil, J. Bowman, N. Balakrishnan, and R. Forrey, Chem. Phys. **532**, 110695 (2020).

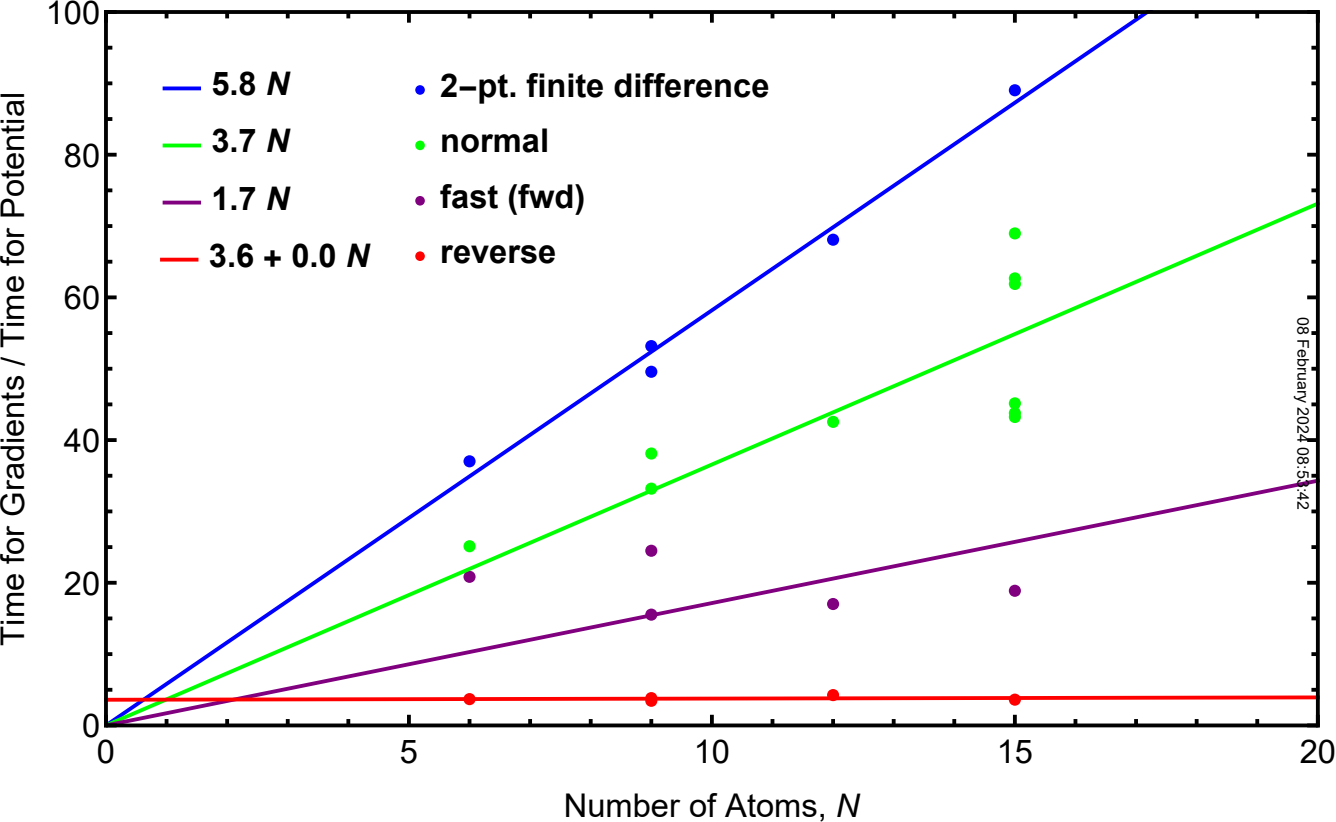

permutational symmetry
polynomial order
msa software generates
bemsa.f90
use fragmentation or
use one fragment
(= parent); generates
DuplicatesDeleted...f90
purification
compaction
grouping
add polynomials
prune polynomials
addition of derivatives (Normal, Fast Forward, Reverse)
check for duplicates, perm invariance
Final Fortran Output Program


2-body
3-body
4-body
rmse (cm^{-1})
Time for basis evaluation (s)
0
10
20
30
40
0.0
0.2
0.4
0.6
0.8
1.0
08 February 2024 08:53:42


50.
40.
30.
20.
10.
0.
rmse for energies (cm^{-1})
0.2
0.15
0.1
0.05
0.
time for 3000 energy calculations (s)
6000
7000
8000
9000
10 000
11 000
12 000
number of polynomials
08 February 2024 08:53:42


rmse for gradients ($cm^{-1}bohr^{-1}$)
time for 300 × (3N−6) gradients (s)
number of polynomials
100.
80.
60.
40.
20.
0.
6.
4.
2.
0.
6000
7000
8000
9000
10000
11000
12000
08 February 2024 08:53:42


mse (cm^{-1})
time for one evaluation of basis set (s)
number of superpolynomials
0.
10.
20.
30.
0
2000
4000
6000
8000
10000
0.1
0.2
0.3
0.4
0.5
08 February 2024 08:53:42


5.8 N
3.7 N
1.7 N
3.6 + 0.0 N
2–pt. finite difference
normal
fast (fwd)
reverse
Time for Gradients / Time for Potential
Number of Atoms, N
0
5
10
15
20
0
20
40
60
80
100
08 February 2024 08:53:42