On the generation of ab initio potential energy surfaces using machine learning techniques

Adam N. Hill

A thesis submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy

August 31, 2023

Faculty of Science
Department of Chemistry

The University Of Sheffield.

ABSTRACT

A potential energy surface is a major pre-requisite to carrying out quantum molecular dynamics studies on chemical systems. These studies allow theoreticians to explore the behaviour of modern-day chemistry in ways that are not feasible in a lab, enabling predictions to be made about experiments performed at extreme temperatures and pressures, and even helping to reveal reaction pathways. To achieve this, a PES associates nuclear position and energy, constructing an n-dimensional surface from high-level *ab initio* calculations that are fit using physically motivated functions. However, this fitting is a painstakingly slow process, and must be tailored to individual systems. This thesis will explore three ways of speeding up the generation of *ab initio* PESs: simplifying the fitting process, reducing the number of fitting points, and reducing the computational cost of the *ab initio* calculations themselves.

Machine learning (ML) algorithms offer a number of potential advantages for the construction of PESs: firstly, they represent more of a "black-box" approach to the fitting that promises an easier route to accurate surfaces; second, reducing the dimensionality of the problem holds the promise of constructing a surface from significantly fewer points. These algorithms also have access to active learning techniques that aim to reduce the size of machine learning datasets. As such, a particular subset of machine learning model, the neural network, will be used along side a novel application of a firefly inspired optimisation algorithm to speed up PES generation. While the development of new basis sets paired with correlation consistent effective core potentials will aim to speed up data generation.

To Kendra

ACKNOWLEDGMENTS

A lot has happened in four and a half years. I got married, witnessed a global pandemic, had a baby, got a job, and finally finished this PhD - in that order. It has been tough, and I would not have made it through without the incredible support of those around me.

First and foremost, I must thank both Grant and Anthony for your guidance and support throughout everything that has transpired over the last few years. You have been incredibly kind and patient with me, and for that I will forever be thankful. Harry, Jaz, thank you for keeping me sane and letting me distract you while we all definitely should have been working, and Heather, thank you for your friendship and support while I navigated postgraduate life.

To my parents, Dai and Lynda, you consistently support me in everything I do, and encourage me to be the best I can be. I could not ask for better role models (I'm just glad that I can say I submitted my thesis before you Mum). To my little sister and baby bro, you are the best family I could ask for. Keep being you, and don't let anyone tell you otherwise.

Finally, to my wonderful wife, Kendra, and my joy in the last twelve months, Teddy. I love you both so much, I'm really not sure I would have got through this without either of you. Kendra, you have supported and encouraged me, you have held and comforted me, and most importantly you are my best friend. Thank you, for everything.

Contents

1 Potential energy surfaces

Potential energy surfaces (PESs) are central to the field of computational chemistry, enabling theoreticians to explore the behaviour of modern-day chemistry in ways not feasible in a lab. Particularly within the field of quantum molecular dynamics (QMD), the study of chemical systems through the solving of the Schrödinger equation, these surfaces have made it possible to predict the results of an experiment performed at extreme temperatures and pressures, predict reaction pathways, and even calculate product ratios without the need for experiment.[1–7] To do so, much data pertaining to the chemical system to be explored is needed; specifically, it is important to have information about the energy of the system at a range of different geometries, and for QMD it is preferable to have energies spanning all possible geometries (the full range of bond lengths and angles (the degrees of freedom) is known as the configuration space). Generating this volume of data is computationally expensive for anything other than small molecules at lower levels of theory, as such, n-dimensional configuration spaces approximating energy from geometry (potential energy surfaces) are needed to minimise the number of expensive *ab initio* calculations.

1.1 The origin of the PES

The nuclei of atoms in molecules respond to the potential energy of the system. The value of this potential at any point in the configuration space of a molecule can be calculated through quantum mechanics and the time-independent Schrödinger equation,

$$\hat{H}\Psi = E\Psi.$$

This equation will be unpacked in detail in Chapter 3, but briefly this is an eigenvector/eigenvalue equation, where the state of a system, Ψ, is acted on by the Hamiltonian operator, $\hat{H}$, to give the energy, E, of that system in the given electronic state. The relation between molecular position and energy defines a potential energy surface. Therefore, by separating the nuclear and electronic motion, solving the Schrödinger equation for the electronic motion at a number of molecular geometries, and adding the potential energy of the nuclei separately, it is possible to build an n-dimensional surface for a molecule or reaction (where n is the number of atoms in the system). This separation of nuclear and electronic motion is known as the Born-Oppenheimer approximation[8] and its application will be justified in Chapter 3.

The minimum energy points in this potential are stable structures. For example, the relative energies of two separate configurations of a molecule will reveal information about the most stable structure, and the form it is likely to spend most of its time in at room temperature and pressure. However, dynamics calculations need more than just information about the stationary points on a surface, they require energies and gradient information of transitional structures as well. Pre-computing all conceivable energies of a system is not a particularly intelligent use of resources, and is not achievable for anything other than small molecules at low levels of theory. As an alternative, direct dynamics has been developed as a process by which the energy of the system is calculated 'on-the-fly' as the geometry is explored allowing for the directed evaluation of molecular behaviour,[9] and has been made possible due to the computational leaps made in recent decades. However, as the molecules of interest get larger, and the computational methods employed to explore their properties get more costly, direct dynamics is also prohibitively expensive for anything other than small molecules. Instead, it is often cheaper to calculate a set of energies for a number of pre-determined geometries and use a multidimensional function to fit the points. This function can be modified and adjusted based on fitting parameters, until the energy of the system is reproduced accurately over a large range of geometries. In theory this minimises the number of expensive *ab initio* calculations required for dynamics calculations, and if this fitted function is fast to evaluate then QMD calculations also become much faster to run.

This n-dimensional function is an approximation of the true potential energy surface defined by the Schrödinger equation, and an example of a 3D representation of a molecular PES is shown in Figure 1.1. Vari-

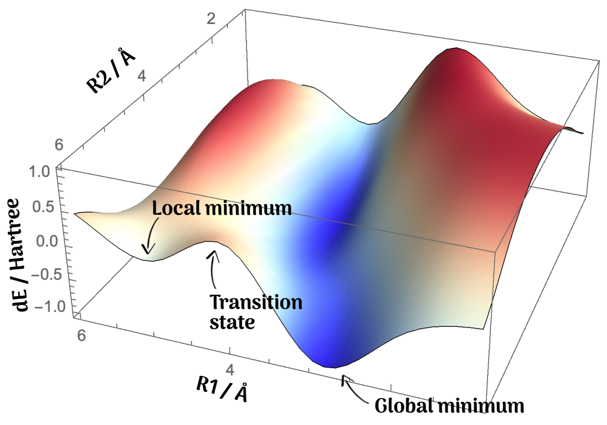


Figure 1.1. Dependence of a molecular PES on bond lengths R1 and R2 of a figurative molecule. Highlighted on the surface are a number of important points: the global minimum, the ground state equilibrium geometry of a molecule; a local minimum, another stable conformation of the molecules; and a maximum/saddle point, a transition state between conformations.

ous features of the surface correspond to important configurations of the molecule, with minima indicating stable structures and first order saddle points being transition states. A surface does not have to be purely uni-molecular however, and it could instead be a reaction PES, where the various minima would now be intermediates and products, and the saddle points would be the transition states between them. When developing a PES for use in quantum molecular dynamics, there are a few features that are important to the final form of the surface. It must be continuous, there should be no holes in the PES, and it must be smooth for both energy and the first and second derivatives of the energy.

1.2 METHODS OF SURFACE FITTING

As was highlighted in Section 1.1, one cannot generate all conceivable geometries of a system, and calculate their energies, for anything other than small molecules. Therefore a smaller number of points must be chosen and the function of the surface approximated.

The simplest, most general, method of fitting is to define a function with a number of adjustable parameters, and optimise the parameters so that the function correctly predicts the *ab initio* energies at any given geometry. This method was employed widely early in the field, and the reader is pointed towards a comprehensive review by Truhlar, Steckler, and Gordon, covering much of the early work surrounding this method[†] of function fitting.[10] For potentials with a small number of parameters this method appears to work well, but a PES developed by Kuhn *et al.*, covering the six degrees of freedom for hydrogen peroxide, has 76 adjustable parameters to define the full surface,[11] clearly making this kind of 'by hand' fitting tedious. Some more modern examples include a Ca_2 surface incorporating the long range behaviour of the system, with 16 fitted parameters[12], and an improvement to an earlier NH_3 surface using 184 parameters.[13,14]

There are two other disadvantages of this method to highlight. The first is that finding a suitable fitting function in the first place can be difficult, especially as systems reach higher degrees of freedom. Secondly this method requires a new function to be designed and fit for every new problem, that is, there is no transferability, which incidentally makes the complicated fitting process even more problematic. To deal with these issues many methods have been developed to semi-automate the fitting process using fitting methods and interpolation.

[†]This method is sometimes known as 'guessed function fitting' as the initial function is chosen as a 'best guess' of the true shape of the potential energy surface.

1.2.1 SPLINE INTERPOLATION

In mathematics, spline interpolation is a type of fitting algorithm that fits several low degree polynomials to pairs of points in the data, instead of fitting a single high degree polynomial to the whole set.[15,16] For example, a set of ten data points could be fit with a single polynomial of degree ten, or instead nine cubic or square polynomials could be fit between pairs of points. One of the main advantages of this method over simple polynomial interpolation is that it avoids Runge's phenomenon (shown in figure 1.2);[17] when using high-degree polynomials there is a chance that the fitted function oscillates between pairs of points, leading to a wildly 'wavy' fit that technically fits the data, but is clearly not a good interpolation of the points. The interpolated line would be said to be *overfit* to the data, reproducing the inputs exactly, but failing to capture the true nature of the curve.

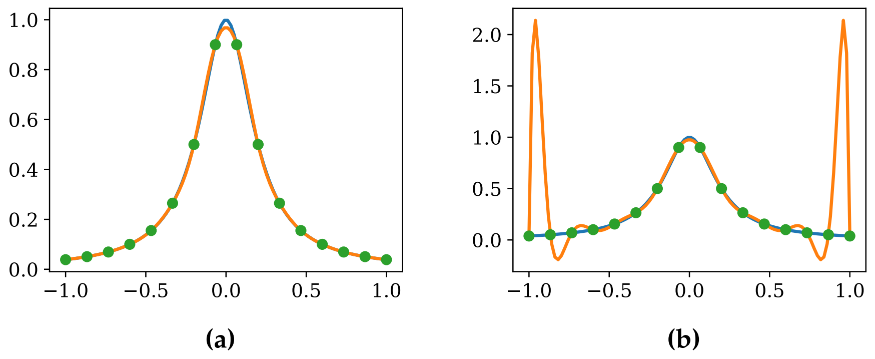


Figure 1.2. Points in the function $y = 1/(1 + 25x^2)$ interpolated by **(a)** cubic splines and **(b)** a 15 degree polynomial exhibiting Runge's phenomenon.

Examples of spline use for the purpose of PES fitting start with McLaughlin and Thompson,[18] who used cubic spline fitting to explore the reaction dynamics of $HeH^+ + H_2 \rightarrow He + H_3{}^+$. The use of splines removed the need to choose an interpolation function manually (and often arbitrarily), and ensured continuity across the whole surface. In an effort to simplify things the authors restricted the symmetry of the reaction pathway to C_{2v}, and report success with the method. However this does pose the question of how to deal with more complex, multidimensional surfaces. Sathyamurthy and Raff[19] aimed to solve this problem by developing a 3D cubic spline fitting routine. This method has been used to calculate both electronic and vibrational energies of small molecules,[20] but Varandas *et al.*[21] expressed concerns with the difficulty in obtaining a global fit of a PES using these methods, due to the problem of dimensionality. That is, the number of *ab initio* calculations that need to be solved for these methods scales with X^{3N-6}, where X is the minimum number of points required to fit the surface, and N is the number of atoms, which can lead to very large numbers of calculations being needed. Although Wu *et al.* were able to accurately fit a surface for H_3 using simple splines, they state that this was only possible due to a dense grid of 76,000 evenly spaced calculations making the fitting relatively simple.[22] As was expressed above, large numbers of high-accuracy *ab initio* calculation are impractical for larger systems, and methods that need less data are largely more desirable.

1.2.2 MODIFIED SHEPARD INTERPOLATION (MSI)

One way of cutting down the number of *ab initio* calculations is to start with a more sparse set of data points, then approximate the surrounding environment. When calculating the energy of a system through the Schrödinger equation it is further possible to calculate the derivative of the energy with respect to the nuclear coordinates. This reveals the shape of the potential around the configuration and the derivative can be used in a multivariate Taylor expansion to approximate nearby energies. The accurate range of these expansions is much smaller than the range of values one might be interested in when evaluating a PES, therefore interpolation between many of these points is required.

Shepard's method of interpolation[23] is as follows: for a set of points $\mathbf{x}$, the interpolated value of an unknown point in the same space is equal to a weighted mean of all the known values. The weighting here is the inverse distance from each point. To improve performance, a cut-off radius can be defined, and points beyond this radius are excluded from the treatment. In the context of PES fitting as described by Collins,[24] it is assumed that the PES can approximated as a Taylor expansion, $T_i(\xi)$, around a given configuration, $\xi(i)$:

$$\begin{aligned} E(\xi) \approx T_i(\xi) = E[\xi(i)] + \sum_{n=1}^{3N-6} [\xi_n - \xi_n(i)] \left.\frac{\partial E}{\partial \xi_n}\right|_{\xi(i)} \\ + \frac{1}{2} \sum_{n=1}^{3N-6} \sum_{m=1}^{3N-6} [\xi_n - \xi_n(i)] \\ \times [\xi_m - \xi_m(i)] \left.\frac{\partial^2 E}{\partial \xi_n \partial \xi_m}\right|_{\xi(i)} + \ldots, \end{aligned}$$

where n and m are internal coordinates. The modified Shepard method then expresses the PES as a weighted average of these Taylor expansions,

$$E(\mathbf{Z}) = \sum_{i=1}^{N_{\text{data}}} w_i(\mathbf{Z}) T_i(\mathbf{Z}), \tag{1.1}$$

where $\mathbf{Z}$ is a set of internal coordinates.† Following the MSI method, configurations that are closest to $\mathbf{Z}$ will have the largest weights, and vice versa.

In contrast to other methods of PES fitting, the MSI method starts initially with a mostly arbitrarily selected path of points on the surface, and the PES is fitted loosely to these points (as opposed to generating a grid of data over the whole configuration space). As the PES is probed in a trajectory calculation, extra *ab initio* points are calculated and added to the fit, in a semi-direct-dynamics manner, to gradually build a picture of the whole surface of interest. This way the number of calculations is kept to a minimum. However it is important to note that in addition to the energy calculation, gradients and Hessians have to be calculated for this method, which adds to the base computational cost. Moyano

†$\mathbf{Z}$ is actually defined as a set of reciprocal distances $Z_i = 1/R_i$, this is because the PES diverges to infinity if any atoms overlap. To make sure that this behaviour is seen, $Z_i = 1/R_i$ is defined so that when $R_i = 0$, $Z_i = \infty$[24]

and Collins proposed an improvement to the method, stating that by changing the selection criteria for new data points faster convergence could be reached.[25] They propose two new schemes, the first selects new points that are locally maxima in either the uncertainty or variance of the interpolated energy. This requires no additional *ab initio* calculations, in contrast to the second method that explores the nearby surface with a small number of additional calculations and adds new data points in regions of the largest interpolation error.

Modified Shepard interpolation has been used to generate surfaces for quantum reaction scattering,[26] to explore the catalysis of proton transport and proton-abstraction reactions,[27] and to build reactive potential energy surfaces for polyatomic systems.[28]

1.2.3 REPRODUCING KERNEL HILBERT SPACE (RKHS) INTERPOLATION

Another method of interpolating between points involves the use of Hilbert spaces, a space of vectors with a defined inner product that satisfies a set of constraints. A reproducing kernel Hilbert space (RKHS)[29,30] is a space of continuous real-valued functions, $f(\mathbf{x})$. If given a set of of $\mathbf{x}_i$, and the values of $f(\mathbf{x}_i)$ are known (f_i), then the representor theorem[31] states that the true function of $f(\mathbf{x})$ can be approximated through a linear combination of kernel products:

$$\tilde{f}(\mathbf{x}) = \sum_{i=1}^{N} c_i K(\mathbf{x}, \mathbf{x}_i). \tag{1.2}$$

Here, c_i are coefficients of the linear combination, and $K(\mathbf{x}, \mathbf{x}')$ is a reproducing kernel.† It is only a reproducing kernel if the inner product of the function with the kernel gives the evaluation functional:

$$f(\mathbf{x}) = \langle f(\mathbf{x}'), K(\mathbf{x}, \mathbf{x}') \rangle',$$

†The form of this kernel is chosen depending entirely on the surface being fit, but is generally a product of D, one-dimensional, kernels $k(x, x')$, such that $K(\mathbf{x}, \mathbf{x}') = \prod_{d=1}^{D} k^{(d)}(x^{(d)}, x'^{(d)})$, where $x^{(d)}$ is the dth component of a D-dimensional vector $\mathbf{x} = \{x^{(d)}\}$, and $k^{(d)}$ is a one-dimensional kernel of dimension d.

where the prime indicates that the inner product is performed over $\mathbf{x}'$. The kernel is also chosen so that it is both symmetric,

$$K(\mathbf{x}, \mathbf{x}') = K(\mathbf{x}', \mathbf{x}),$$

and positive definite,

$$\sum_{i,j=1}^{n} c_i K(\mathbf{x}_i, \mathbf{x}_j) c_j \geqslant 0.$$

such that the following equation can be solved through the Cholesky decomposition[32] when the coefficients c_i satisfy the linear relationship $f_j = \sum_{i=1}^{N} c_i K_{ij}$, where $K_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$:

$$\begin{vmatrix} K_{11} & K_{12} & \cdots & K_{1N} \\ K_{21} & K_{22} & \cdots & K_{11} \\ \vdots & \vdots & \ddots & \vdots \\ K_{N1} & K_{N2} & \cdots & K_{NN} \end{vmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix}. \tag{1.3}$$

Once the values of c_i have been calculated, then the function $\tilde{f}(\mathbf{x})$ from equation 1.2 can be evaluated for any value of $\mathbf{x}$, thus interpolating between the points of training data. The advantage of using RKHS interpolation over something like spline interpolation is that it can handle irregularly spaced points and missing values, making it much more robust.

This method has been used in the exploration of $N(^2D) + H_2$ by Hollebeek *et al.*[33] Two surfaces were produced for the 1A″ and 1A′ states of the reaction and surface crossing calculations carried out. They report that "the resulting surfaces are smooth, accurate, efficient to evaluate, exactly reproducing the input data upon which they are based. . . ".[33] Ho and Rabitz have also successfully produced a surface for the $C(^1D) + H_2$ reactive system, showing that the use of RKHS interpolation could produce an accurate and smooth PES using significantly fewer points of *ab initio* data, while also covering regions that are dynamically relevant and non rectangular.[34] There are, however, two major drawback to this method, occurring as the set of N training examples becomes large:[30]

1. The method used to solve equation 1.3 to determine the coefficients scales with $\mathcal{O}N^3$, which quickly gets prohibitive for large numbers of training examples ($N > 10^5$).

2. Equation 1.3 also reveals that the evaluation of $f(\mathbf{x})$ involves a sum over all training examples scaling as $\mathcal{O}N$. This is acceptable if only a few evaluations are required, but for QMD calculations many thousands of evaluations are needed and this results directly in a linear increase in simulation time.

1.2.4 INTERPOLATING MOVING LEAST-SQUARES (IMLS)

Computing both the derivative of the energy and the Hessian of a system, such as in modified Sheppard interpolation, adds significant computational cost to the generation of data and initial fitting points. One method that avoids both of these entirely has been outlined by Guo, called the interpolating moving least-squares (IMLS) method, and its application to PES fitting is as follows.[35] The fitted potential in this case is expressed as a linear combination of basis functions, $b_i(\mathbf{Z})$, dependent on the molecular configuration, $\mathbf{Z}$,

$$V_{\text{fitted}}(\mathbf{Z}) = \sum_{i=1}^{M} a_i(\mathbf{Z}) b_i(\mathbf{Z}). \tag{1.4}$$

M is the total number of basis functions, while $b_i(\mathbf{Z})$ are polynomials of a known form. The values of $a_i(\mathbf{Z})$ are determined by solving $\frac{\partial D}{\partial a_i} = 0$ where D is defined by

$$\begin{aligned} D[V_{\text{fitted}}(\mathbf{Z})] &= \sum_{j=1}^{N} w_j(\mathbf{Z})[V_{\text{fitted}}(\mathbf{Z}^{(j)}) - V(\mathbf{Z}^{(j)})]^2, \\ &= \sum_{j=1}^{N} w_j(\mathbf{Z}) \left[\sum_{i=1}^{M} a_i(\mathbf{Z}^{(j)}) b_i(\mathbf{Z}^{(j)}) - V(\mathbf{Z}^{(j)}) \right]^2, \end{aligned}$$

a sum of weighted squared deviations.[35] This is solved for N *ab initio* points at positions $\mathbf{Z}^{(ji)}$ with energies $V(\mathbf{Z}^{(j)})$. The parameter $w(\mathbf{Z})$ is a weight function, and similarly to the modified Shepard interpolation technique, this weight is larger the closer the unknown point is to the known points. Following this minimisation through leads to a matrix equation of the form,

$$\mathbf{B}^{T}\boldsymbol{W}\mathbf{B}\mathbf{a} = \mathbf{B}^{T}\boldsymbol{W}\boldsymbol{V}, \tag{1.5}$$

where $\mathbf{B}^{T}$ denoted the transpose of the $N \times M$ matrix $\mathbf{B}$,

$$\mathbf{B} = \begin{pmatrix} b_1(\mathbf{Z}^{(1)}) & b_2(\mathbf{Z}^{(1)}) & \cdots & b_M(\mathbf{Z}^{(1)}) \\ b_1(\mathbf{Z}^{(2)}) & b_2(\mathbf{Z}^{(2)}) & \cdots & b_M(\mathbf{Z}^{(2)}) \\ \vdots & \vdots & \ddots & \vdots \\ b_1(\mathbf{Z}^{(N)}) & b_2(\mathbf{Z}^{(N)}) & \cdots & b_M(\mathbf{Z}^{(N)}) \end{pmatrix},$$

$\boldsymbol{W}$ is a diagonal matrix of weights of size $N \times N$ and $\boldsymbol{V}$ is a vector of energies, length N,

$$\boldsymbol{W} = \mathrm{diag}[w_1(\mathbf{Z}), w_2(\mathbf{Z}), \ldots, w_N(\mathbf{Z})],$$
$$\boldsymbol{V} = [V(\mathbf{Z}^{(1)}), V(\mathbf{Z}^{(2)}), \ldots, V(\mathbf{Z}^{(N)})]^{T}.$$

Solving equation 1.5 leads to the vector $\mathbf{a}$ being obtained, which can then be used in equation 1.4 to calculated the potential at a given point $\mathbf{Z}$. The downside of this method is that $\mathbf{a}$ is a function of $\mathbf{Z}$ and thus must be re-evaluated at every single step of a trajectory propagation. Therefore the method cost scales with NM^2 and it can get prohibitively expensive if steps aren't taken to minimise both M and N.

The IMLS method has been directly compared to the MSI method by Ishida *et al.*[36] They compared the results from an IMLS fit of the $O(^1D) + H_2$ surface to that of a Shepard fit using second-order Taylor expansions, and report that the surface "reproduces the correct reactive cross-sections more accurately than the Shepard scheme, and with RMS [root mean squared] errors for energy and gradients that are significantly smaller than those from Shepard interpolation." They go on to clarify that "this occurs even though the present scheme does not utilize derivative and Hessian information, whereas the Shepard interpolation does", making this method far cheaper computationally. It has also been shown by Guo *et al.*, that a small number of points are able to fit an accurate PES for the system H_2 + CN $\rightarrow$ H + HCN. As few as 330 CCSD(T)/aug-cc-pvtz calculations were used to for a PES using the IMLS method, producing root mean squared errors of "few tenths of a kcal/mol."[37] The predictions of vibrational levels of $^1CH_2$ and HCN have also been made with errors as small as 0.1 cm^{-1} using less than 500 *ab initio* calculations of the energy and gradients.[38]

1.2.5 DOUBLE MANY BODY EXPANSION (DMBE)

The interpolation methods presented so far are very involved, and in the case of MSI and IMLS, probing the surface requires new calculations of gradients or Hessians. The double many body expansion (DMBE) method makes probing the surface a little easier by providing a single functional, that once fit, requires no extra *ab initio* data when it comes time to evaluate the surface. The method has been described in detail by Varandas[39] and the reader is pointed towards references for justifications of the forms of these equations and in depth review.[39–42] Here an overview of the formulation is presented for comparison to other PES fitting methods.

The potential energy surface is defined in two parts, an extended Hartree-Fock type energy, V_{EHF}, and a dynamic correlation term V_{dc},

$$V = V_{\text{EHF}}(\mathbf{R}) + V_{\text{dc}}(\mathbf{R}).$$

For an N-atom system this is generalised to

$$V(\mathbf{R}^N) = \sum_{n=2}^{N} [V_{\text{EHF}}^{(n)}(\mathbf{R}^n) + V_{\text{dc}}^{(n)}(\mathbf{R}^n)]. \tag{1.6}$$

where $\mathbf{R}^N$ is the full set of interatomic coordinates of N atoms, $\mathbf{R}^n$ is a set of $n(n-1)/2$ interatomic coordinates of fragments containing n atoms, and $V_x^{(n)}$ (where x = EHF, dc) is an n body term of the energy component. If $n = 2$, $\mathbf{R}^2$ represents a diatomic fragment of the full molecule, and $n = 3$ is a triatomic, and so forth. Equation 1.6 can be expanded and written as a 'many body expansion' of terms such that,

$$V_{\mathrm{EHF}}(\mathbf{R}^N) = \sum_{\alpha\beta} V^{(2)}_{\mathrm{EHF}}(\mathbf{R}^2) + \sum_{\alpha\beta\gamma} V^{(3)}_{\mathrm{EHF}}(\mathbf{R}^3) + \sum_{\alpha\beta\gamma\rho} V^{(4)}_{\mathrm{EHF}}(\mathbf{R}^4) + \ldots, \quad (1.7)$$

$$V_{\mathrm{dc}}(\mathbf{R}^N) = \sum_{\alpha\beta} V^{(2)}_{\mathrm{dc}}(\mathbf{R}^2) + \sum_{\alpha\beta\gamma} V^{(3)}_{\mathrm{dc}}(\mathbf{R}^3) + \sum_{\alpha\beta\gamma\rho} V^{(4)}_{\mathrm{dc}}(\mathbf{R}^4) + \ldots, \quad (1.8)$$

where the sums over $\alpha, \beta, \gamma, \rho$ indicate all combinations of diatomics, triatomics, and so on. The many body expansion terms of equations 1.7 and 1.8 are specific to the system, and must be formulated each time a new fragment is to be explored. The DMBE method has been used widely[39,43–47] and Ballester and Varandas have used it to develop a global potential energy surface for HSO_2,[41,42] which will specifically be explored in detail in Chapter 6. The following are examples of the two, three, and four body terms specific to the HSO_2 surface (their forms will differ for other systems). The two body terms are relatively straight forward; for the two-body EHF energy,

$$V^{(2)}_{\mathrm{EHF}} = DR^{-1}\left(1 + \sum_{i=1}^{3} a_i r^i\right) e^{-\gamma(r)r} + \chi_{\mathrm{exec}}(R) V^{\mathrm{asym}}_{\mathrm{exc}}(R). \quad (1.9)$$

$\chi_{\mathrm{exec}}(R)$ is a damping function to account for charge overlap effects, D is the well depth of the diatomic potential energy curve,[48] and γ is a range determining exponent, where

$$\gamma = \gamma_0[1 + \gamma_1 \tanh(\gamma_2 r)]. \quad (1.10)$$

Here $r = R - R_e$ is the equilibrium displacement of the diatomic, and γ_i are variable coefficients chosen based on the specific requirements of the system.[41] The asymptotic exchange energy $V^{\mathrm{asym}}_{\mathrm{exc}}(R)$ is defined as,

$$V^{\mathrm{asym}}_{\mathrm{exc}}(R) = -\tilde{A}R^{\tilde{\alpha}}(1 + \tilde{a}_1 R + \tilde{a}_2 R^2)e^{-\tilde{\gamma}R} \quad (1.11)$$

with $\tilde{A}, \tilde{a}_i, \tilde{\alpha}, \tilde{\gamma}$ are usually just *a priori* theoretical parameters.[41] The associated two-body dynamical correlation potential is,

$$V^{(2)}_{\mathrm{dc}} = \sum_n C_n \chi_n(R) R^{-n}, \quad (1.12)$$

where the damping function $\chi_n(R)$ takes the form,

$$\chi_n(R) = \left[1 - \exp\left(-A_n\frac{R}{\rho} - B_n\frac{R^2}{\rho^2}\right)\right]. \tag{1.13}$$

This function has a set of 'dimensionless universal parameters' that define $A_n = \alpha_0 n^{-\alpha_1}$ and $B_n = b\beta_0 e^{\beta_1 n}$ ($\alpha_0 = 16.36606$, $\alpha_1 = 0.70172$, $\beta_0 = 17.19338$, $\beta_1 = 0.09574$), and for an atom pair, XY, $\rho = 5.5 + 1.25(\langle r_X^2\rangle^{1/2} + \langle r_Y^2\rangle^{1/2})$.[41] All remaining coefficients appearing in the two-body equations above (1.9-1.13) are chosen so that they reproduce the experimental and theoretical data available for the diatomic fragment in question.[41] It is clear from the two body terms alone that the DMBE method is very involved, and requires a lot of prior knowledge about the system being explored. The addition of both three and four body terms increases this problem.

For the three body terms the extended Hartree-Fock energy takes the form,

$$V_{\text{EHF}}^{(3)}(\mathbf{R}^3) = \sum_i^n P_i^{(3)}(\mathbf{R}^3)T_i^{(3)}(\mathbf{R}^3),$$

where $T_i^{(3)}$ is a function determining the range, $P_i^{(3)}$ is a three-body polynomial, and n is the number of polynomials used in the fit. The value of n and the form of $P_i^{(3)}$ are determined by the fitting requirements of the three-body system they come from.

The three-body dynamical correlation is treated a little differently to the two body term, in that the following general form of the dynamical correlation and electrostatic energies is used,

$$V_{\text{ele}}^{(3)}(\mathbf{R}^3) = \sum_i \sum_n f_i(\mathbf{R}^3)C_n^{(i)}(R_i, \theta_i)\chi_n(r_i)r_i^{-n}, \tag{1.14}$$

where

$$f_i(\mathbf{R}^3) = \frac{1}{2}\{1 - \tanh[\xi(\eta s_i - s_j - s_k)]\}. \tag{1.15}$$

(r_i, R_i, θ_i) represent Jacobi coordinates. Equation 1.14 contains long range coefficients $C_n^{(i)}(R_i, \theta_i)$ that take the values $n = 4$ for the dipole-quadrupole and $n = 5$ for the quadrupole-quadrupole interactions.

Atom-diatom dispersion coefficients are represented by $C_n^{(i)}(R_i, \theta_i)$ for $n = 6$, 8, and 10. χ_n are the same damping functions that are defined in equation 1.13. In equation 1.15, the switching function $f_i(\mathbf{R}^3)$, has $s_i = R_i - R_i^{\text{ref}} \quad (i = 1 - 3)$, which is a displacement from the reference geometry.

The four body terms for HSO_2 initially seem very similar, taking the forms,

$$V_{\text{EHF}}^{(4)}(\mathbf{R}^4) = V_S^{(4)} + V_T^{(4)} + P^{(4)}T^{(4)}, \tag{1.16}$$

$$V_{\text{ele}}^{(4)}(\mathbf{R}^4) = \sum_i \sum_{n=3,4,5} f_i(\mathbf{R}^4) C_n^{(i)}(R_i, R_{i+3}, \theta_i, \theta_{i+3}, \phi_i)\chi_n(r_i) r_i^{-n}, \tag{1.17}$$

$$f_i(\mathbf{R}^4) = \frac{1}{2}\{1 - \tanh[\beta(\eta S_i - S_j - S_k)]\}, \tag{1.18}$$

however there are some important differences. In equation 1.17 values of $C_n^{(i)}$ are based upon well established forms,[49] e.g.,

$$C_3 = -\mu_{AB}(R_{AB})\mu_{CD}(R_{CD})(2\cos\theta_a\cos\theta_b - \sin\theta_a\sin\theta_b\cos\phi).$$

$\mathbf{R}^4$ contains six distances defining a four atom system, and the switching function (equation 1.18) uses generalised coordinates, $S_i = s_i + s_{i+3}$. Here, $s_i = R_i - R_i^{\text{ref}}$ is a displacement from an equilibrium distance. The two parameters β and η are defined so that $f_i(\mathbf{R}^4)$ vanishes when any of the four atoms is placed at infinite separation to the other three, and it must equal one at the diatom-diatom dissociation limit.

The EHF energy term was derived from exploring a version of the PES excluding it, and ensuring that any function added should alter the surface so as to fix issues such as too deep energy wells and the absence of formation barriers.[41] As a result of this exploration, equation 1.16 is made up of two four body Gaussian-type functions ($V_s^{(4)}$ and $V_T^{(4)}$), a second-order polynomial ($P^{(4)}$), and a Gaussian-type range determining factor ($T^{(4)}$).

It is clear that the fitting of a surface using the DMBE method is very involved, requires specific formulations for each system of interest, and most importantly, intimate knowledge of the surface behaviour is needed to fit it in the first place (which introduces significant problems surrounding chemical intuition). The major advantage of this method over methods such as spline interpolation or RKHS, is that there is defined a functional potential that can be evaluated in seconds for any geometry, without the need for further *ab initio* calculations, allowing for fast surface evaluation.

1.3 IMPROVING UPON CURRENT METHODS

It is clear that conventional methods of PES fitting are cumbersome and involved. Either a lot of work has to go into the functional form of the surface (in the case of DMBE) or many costly *ab initio* calculations are required to begin the fitting process. In addition, some methods require even more calculations to be carried out to accurately probe the full surface, such as derivatives being calculated at every step in the IMLS method, or every function evaluation requiring a sum over all training examples in RKHS.

For QMD to be more easily accessible, it would be helpful to speed up or simplify the fitting of potential energy surfaces so more molecules can be described by accurate potentials. There are three ways of speeding up PES generation that will be explored in this thesis:

1. Simplify the fitting process, either making it transferable or more easily applied to other systems.
2. Speed up the *ab initio* data generation itself.
3. Reduce the number of points of data needed to fit an accurate surface.

Simplifying the fitting process. At the core of the PES problem is a set of data points, representing geometries and their calculated energies, for which it is assumed that a function exists that maps the geometry and atoms onto an energy. However it is either unknown, incredibly sophisticated, or both. In recent years the field of machine learning (ML) has boomed in the literature with the number of ML papers being released year to year drastically increasing.[50] Machine learning algorithms specialise in fitting unknown functions to large datasets and certainly appear like they would be well equipped to assist with PES fitting. Chapter 2 will show that this is the case, and outline exactly how one can apply the concepts of machine learning to the generation of potential energy surfaces.

Speeding up data generation. Before a surface can be fit, a number of *ab initio* calculations must be carried out to attain the data needed for the fitting process, and this is still the case for ML fit PESs. Although computational power has increased massively over the last decade, potential energy surfaces can often require a spectroscopic level of accuracy (on the scale of a few cm^{-1}) and as such the level of theory that the surface needs to be fit at has grown and grown. For example, Garrido *et al.*[51] report that the initial level of theory that the DMBE-HSO_2 PES was fit at (complete active space self consistent field theory using a correlation consistent, polarised triple-ζ basis set augmented with additional diffuse functions [CASSCF/aug-cc-pVTZ]) is not sufficient to accurately represent all aspects of that system. As such they performed additional higher level calculations (complete active space with second order perturbation theory [CASPT2/aug-cc-pVTZ]) and used them to improve the fit of the surface. These kinds of high level calculation will be computationally expensive, especially as the systems of interest get larger.

Computational calculation times are directly related to the number of electrons in the system. This fact opens up a pathway to speed things up. Using an effective core potential (ECP) to model a number of the core electrons in an atom effectively 'removes' them from the calculation, directly speeding up the calculation (Chapter 3 contains further technical details). Sticking with the example of HSO_2, recently a set of correlation-consistent ECPs (ccECPs) for second row atoms were developed by Bennet *et al.*[52] and can be used to replace some of the core electrons in sulfur. However, to use these ccECPs, specifically optimised basis sets must be paired with them in the electronic structure calculation, so Chapter 5 will discuss the development of a new family of basis sets for second row atoms.[53]

This chapter will also cover the fitting of new core polarisation potential (CPP) parameters for second row atoms that approximate the effects of correlating core electrons in an electronic structure calculation.

Reducing the initial dataset size. Finally, it would be ideal to reduce the size of the dataset needed for an accurately fit surface, circumventing a number of *ab initio* calculations entirely. Active learning is a concept within the field of machine learning that aims to reduce the size of training datasets. It achieves this by first training on a minimal dataset, then selectively including additional, important, data points to improve the fit as quickly as possible with as few points as possible. How data points are defined as 'important' is different from method to method. Chapter 4 will introduce a new a new algorithm based on a firefly heuristic to be used with machine learning techniques to identify regions of 'poor' fit on potential energy surfaces and incrementally add new data points to the dataset.

Finally Chapter 6 will bring together both the new basis sets and the new firefly algorithm, to train a machine learning algorithm to fit a global PES for HSO_2, at a higher level of theory than the original DMBE-PES of Ballester and Varandas, using minimal data points in the fitting process.

2 | Machine learning

Machine learning is a statistical ‘tool’ that scientists and software developers are using to assist with the areas of research and development that prove difficult, or sometimes impossible, to navigate. Predominantly, ML is being applied to the analysis of large volumes of data and pattern recognition in very advanced relationships, but the methods can often be difficult to implement and as such, the human element has not been entirely eliminated. Some areas that machine learning algorithms are being applied to today include: image captioning[54], machine translation[55], playing complex computer games[56], and even helpful, human-like chat bots (GPT-4 and ChatGPT).[57] The following quote is the output of ChatGPT, a state of the art natural language processor, prompted with the following: “write me an example of an introduction to a thesis on machine learning in computational chemistry”.

> *The field of computational chemistry has seen a rapid growth in recent years, largely due to the development of machine learning algorithms that enable efficient and accurate predictions of molecular properties and behavior. The use of machine learning in computational chemistry has opened up new avenues for exploring complex chemical systems, from drug discovery to materials design. This thesis aims to investigate the potential of machine learning approaches in predicting various chemical properties, such as molecular energy, reactivity, and solubility, using large-scale datasets and state-of-the-art algorithms. By harnessing the power of machine learning, this research seeks to enhance our understanding of chemical processes*

> *and inform the design of new molecules and materials with desirable properties.*[57]

It covers the important topics, hitting on the concepts of property prediction and drug design, while also accurately bringing in ideas surrounding large datasets and complex algorithms. Most interestingly, and controversially, it reads much like something a human could have written themselves.

A concise, general definition for machine learning was first given by Samuel:

> [Machine learning is the field] of study that gives computers the ability to learn without being explicitly programmed. - *Arthur Samuel, 1959*

This statement captures the hypothesis behind this field of research - is it possible to get a computer program to improve itself and learn from mistakes, without having to write a long list of rules that have to be added to as time goes on? A more specific breakdown of what is meant by the idea of *learning* is given in the textbook by Mitchell:[58]

> A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E. - *Tom Mitchell, 1997*

When asking if a computer is learning it is possible to simply ask: does it get better at the task given to it over time?

2.1 THE TYPES OF MACHINE LEARNING

Machine learning is specifically a subset of the wider Artificial Intelligence (AI) umbrella. The distinction between the two is subtle, AI covers all aspects of computers exhibiting human like intelligence; whereas machine learning is simply an application of AI, using statistical models to learn without direct intervention. Learning is the process of taking a *dataset* (be it a collection of images, data rows in a table, or outputs from electronic structure calculations etc.) and feeding it into an algorithm that can extract some insights from the data. A computer is said to have learnt if its performance at a given task improves with experience of said task. Providing the algorithm with more experience is called *training*. There are a few different ways to achieve learning, and ML can be further broken down into broad categories of *supervised*, *unsupervised*, and *reinforcement* learning, the major differences lying in the data being used to train the algorithm. Ultimately the end goal is insight, either in the form of prediction, or in the form of categorisation.

In what is arguably the simplest of the three to understand, supervised learning makes use of labelled training data, providing the algorithm with both the input and the 'correct' output. The aim here is to discover the relationship between the input and the output, so as to predict the output from new datapoints. It is possible to further define regression and classification problems within the supervised learning framework. The output from regression tasks is a continuous variable, whereas a classification task would have a categorical output (such as identifying an image as either a cat or not a cat). On the other hand, unsupervised learning makes use of unlabelled data. The ML algorithm in this case aims to find patterns and relationships in the data, without a pre-defined end goal, known as association and clustering. This is useful for the analysis of very noisy data, or even data that looks, to the human eye, unrelated. Examples include media recommendations for TV shows or music; the machine learning algorithm clusters users into similar groups, assuming that likeminded people will enjoy the same kinds of media. In the scientific community, unsupervised learning is used in a number of different areas, for example: dimensionality reduction through principal component analysis,[59–65] and the approximation of molecular dynamics kinetics simulations.[66,67] For more information on the architecture of these unsupervised learning algorithms the interested reader is directed to a comprehensive review by Glielmo *et al.*[68]

Finally, reinforcement learning is quite different to both supervised and unsupervised learning in its approach to training. In these models an *agent* manipulates its *environment* to achieve an *end state*. The agent makes decisions and is *rewarded* if said decision is successful in moving towards the end state. If it is not then the agent is either not rewarded or *punished*. In the wider science community these kinds of learning algorithms have found use in self driving cars,[69] robotics,[70] and gaming.[56] Within computational chemistry they have been used to optimise chemical reactions,[71], design new molecules[72] and even predict the ground state energy of a system.[73] Similarly, the interested reader is pointed towards a recent review by Gow *et al.* for more information on the applications of reinforcement learning to computational chemistry.[74]

Table 2.1 shows a summary of the three types of learning, outlining the core differences between them. Supervised learning methods are

Table 2.1. Comparison of the different types of machine learning.

	Supervised	**Unsupervised**	**Reinforcement**
Data type	Labelled data	Unlabelled data	Punishment and reward
Problem type	Regression, classification	Association, clustering	Exploitation, exploration
Example models	Linear regression, Neural Networks	K-means	Q-learning
Applications	Property prediction, energy prediction	Dimensionality reduction for molecular dynamics	Reaction optimisation, molecular design

most common in the application of ML to computational chemistry, and will therefore be the only type of algorithm to be discussed in detail in this chapter.

2.1.1 LINEAR REGRESSION

A logical way to understand the various parts that make up a supervised learning algorithm is to look at the simplest example: linear regression. In statistics, linear regression is a way of modelling a linear relationship between an dependent and an independent variable. In machine learning the dependent variable would be the output or *labels*, while the independent variable would be the inputs or *features*. The aim in any supervised machine learning task is to find the function that transforms the features, x, into the labels, y. Oftentimes x is a vector of several features called a *feature vector*, and allows the label to depend on more than one variable. A collection of feature vectors and their true/calculated labels is known as a dataset.

A *hypothesis function* is a guess that should sufficiently represent the true function describing the system. For linear regression it takes the following form,

$$y = \omega \cdot x + b. \tag{2.1}$$

Here ω is a *weight* controlling the contributions of x, and b is a *bias* used to shift the values by a fixed amount. The 'best' values of ω and b will give rise to the line of best fit, and a function that can successfully predict new values of y for any value of x. By changing the hypothesis function to something more complicated, including powers and exponentials (known as nonlinear regression), it is possible to model much more in depth relationships between x and y.

To find the 'best' values of the weights and biases, the algorithm is first shown a selection of the dataset, called the training set. The weights and biases can be initialised randomly and then adjusted through an iterative process of minimising the error between all of the predicted labels, y_{pred}, and the true labels, y. This error is quantified through a *loss function*, J,

$$J = \frac{1}{n}\sum_{i=1}^{n}(y_i - y_{\text{pred},\,i})^2,$$

or substituting in the hypothesis function, equation 2.1,

$$J = \frac{1}{n}\sum_{i=1}^{n}(y_i - (\omega x_i + b))^2, \tag{2.2}$$

where n is the number of datapoints in the training set. For linear regression this is simply the Mean Squared Error (MSE) between y_{pred} and y_i. To minimise this value the model typically uses the *gradient descent* algorithm.[75] Once J is minimised, the algorithm uses the resulting, optimal values of ω and b to predict the labels for the remainder of the dataset, called the test set, to provide an estimate of the performance of the model and ensure that the algorithm can generalise to datapoints that it was not trained on.

The gradient descent algorithm. Imagine a hiker is at the top of a hill and wishes to reach the bottom. They take a look at their surroundings and head downhill in the direction that is steepest. They take large steps while the hill is very steep but as they move closer to their goal, and the terrain starts levelling out, they begin to take smaller and smaller steps. Eventually every direction to take a step will be up hill and they know they are at the bottom of the hill. This "hill walker" heuristic gives some insights into how the gradient descent algorithm works. Firstly the partial derivatives with respect to the weights and biases of the loss function ($\nabla_\theta J(\theta)$) are calculated to determine the gradients at the current step,

$$\begin{aligned}
\nabla_{\theta_\omega} J(\theta) &= \frac{1}{n}\sum_{i=1}^{n} 2(y_i - (\omega x_i + b))(-x_i), \\
&= \frac{-2}{n}\sum_{i=1}^{n} x_i(y_i - y_{\text{pred},i}), \\
\nabla_{\theta_b} J(\theta) &= \frac{-2}{n}\sum_{i=1}^{n}(y_i - y_{\text{pred},i}).
\end{aligned}$$

The weights and biases are then updated with respect to the gradients calculated,

$$\begin{aligned}\omega &\leftarrow \omega - \eta \cdot \nabla_{\theta_\omega} J(\theta),\\ b &\leftarrow b - \eta \cdot \nabla_{\theta_b} J(\theta).\end{aligned}$$

η here is the *learning rate* and controls the size of the update made at each step. A small value of η is common ($\eta = 0.0001$) to ensure reasonable accuracy.[†] This process is repeated until a threshold on the loss function is reached. Ideally this would be $J = 0$, but in reality it is set to a sensible value for the problem being solved. The loss function could be something other than MSE, such as root mean squared error (RMSE) or mean average error (MAE), and often the choice of loss function can be the difference between a ML model performing well or performing poorly.[76,77]

2.1.2 CONSIDERATIONS WHEN USING MACHINE LEARNING

In the application of machine learning there are two major pitfalls to consider, either the data used to train the network could be 'bad' or the algorithm chosen is unsuitable for the given task.

The data Firstly it is important that there is enough data. Small datasets are hard for ML algorithms to sufficiently learn from (although there is interesting research into low data training).[78] Secondly, and perhaps slightly obviously, it is important that the data being fed into the algorithm is accurate and does not contain errors, as an ML algorithm will not attempt to identify if data is erroneous and will train on anything passed to it.

Not only does the dataset need to be large enough to learn from, but the training set needs to be representative of the whole information space in order to cover all areas that predictions could potentially need to be

[†] η is an example of a *hyperparameter*, some value within the algorithm itself that can be adjusted to alter how the algorithm performs.

made for. For example, assume a model needs to predict house prices all around the world. If it has been trained on only house prices in the UK there is no way for the model to accurately predict house prices in other countries. There may be some similarities between size and local geography, but the housing market from country to country is different enough that it is unlikely the model will be able to extrapolate well. This is, in fact, an important point to make about machine learning in general, in that it can not necessarily extrapolate well, only interpolate.

The feature vector chosen as the representation of the dataset is very important, and must be sufficient enough to contain all the relevant information, but not too complex as to cause training issues. In the example of the house price prediction, if only location was given to the model as a feature it is not likely that the predicted house price would be accurate as there is too much variation in house price within similar locations. If instead the feature vector contained information about location, number of bedrooms, size of house, and number of bathrooms, it would be expected that the model would perform much better. The problem of representation is of particular interest in the application of machine learning to computational chemistry, and will be discussed in Section 2.3.

The algorithms Using a very complex model, such as an artificial neural network (NN), for a simple problem can lead to a problem known as overfitting. The essence of this problem is that the algorithm learns the training data too well, and has so much freedom in the 'line' that it can fit, that it cannot generalise to new datapoints/the test set. For example, Figure 2.1 shows two lines fit to the same dataset. In Figure 2.1a the predicted hypothesis function is a simple curve, with an RMSE of 0.2098 (dimensionless). To the human eye this seems like a good fit, and it likely is. However, Figure 2.1b has an RMSE of zero (the curve goes through every point), which would suggest that it is a better fit. This is not the case, and is obvious to the reader, but without plotting the data it is not immediately clear (this is similar to Runge's phenomenon seen in 1). This is where the test set mentioned earlier comes into play. Once the machine

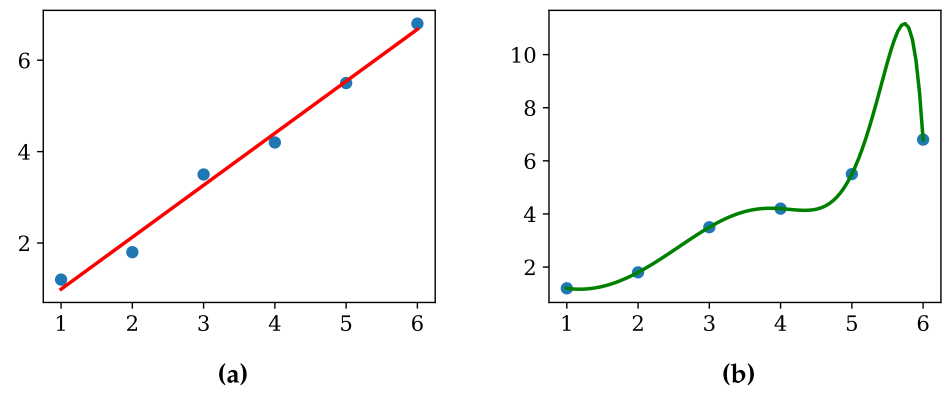


Figure 2.1. **(a)** Line of best fit. **(b)** 20-degree polynomial curve of best fit. They both have the similar RMSE values, in fact **(b)** is zero, but the polynomial curve clearly cannot generalise.

learning model has been trained, two loss values can be calculated, one for the training set, and the other for the test set. If the training loss value is drastically lower than the test set loss, then the model is said to be overfit. There are several methods of combatting overfitting, and they often come in the form of *hyperparameters*, but it is sometimes better to just use a simpler model.

A hyperparameter is a value within the model that can be adjusted in order to modify the behaviour of the algorithm, and improve the overall performance. Some examples include the learning rates of optimisers, the train/test split ratio, and the number of layers in a neural network. To determine the best values for the hyperparameters the test set is often split further into two sets: a validation set; and a test set. The validation set is used during training to check if the model can generalise. After one training cycle (a set of parameter updates, called an *epoch*), the model is used to predict values in the validation set, and much like before if the loss value on the training set is significantly less than the loss for the validation set, then the model is likely overfit. The hyperparameters can then be adjusted to reduce the error in the validation set, while also maintaining the performance on the training set. The test set is a final test of the algorithm to ensure that it has not just overfit to the validation set after hyperparameter optimisation. It is vital that the network is tested on data that it has never seen before, to ensure a fair assessment of performance.

Underfitting is the opposite problem and is commonly caused by the use of models that are too simple. For example, linear regression is unlikely to be able to model the housing market to any sufficient level as there are just too many factors at play. This can be fixed by using a more powerful model (such as a neural network) or even using a better selection of features. When considering the method to use for a new ML problem it is important to balance underfitting and overfitting (sometimes called bias and variance respectively) to achieve the best possible accuracy.

2.2 NEURAL NETWORKS

Artificial neural networks were given their name due to having been inspired by a biological neural network, the brain. In a drastic simplification, neurons in the brain work by triggering a signal when they receive enough inputs from other neurons, being similar to logic gates in this manner.

Neural networks are developed from an interconnected set of simple units (nodes or neurons) that resemble these logic gates, they take inputs from either the feature vectors, or a number of other nodes, and output a single value. One of these nodes is shown in Figure 2.2a and it can be seen that each node in the network is actually a single linear regressor such as in Section 2.1.1, with the output being transformed by an *activation function* (in this case bounding the output to between 0 and 1). The combination of several of these nodes is what makes up a neural network, and an example of one is shown in Figure 2.2b. The first set of nodes is called the *input layer*. These inputs then connect to each node in the next layer that performs a transformation on the input, with contributions of each input node being controlled by the weights, ω, and offset by some value of the bias, b. This layer is called the *hidden layer* as the outputs are not seen. These outputs could then either lead into another hidden later, or alternatively feed into a final layer, y, known as the *output layer*. The power of NNs lies in the fact that they are able to make multiple

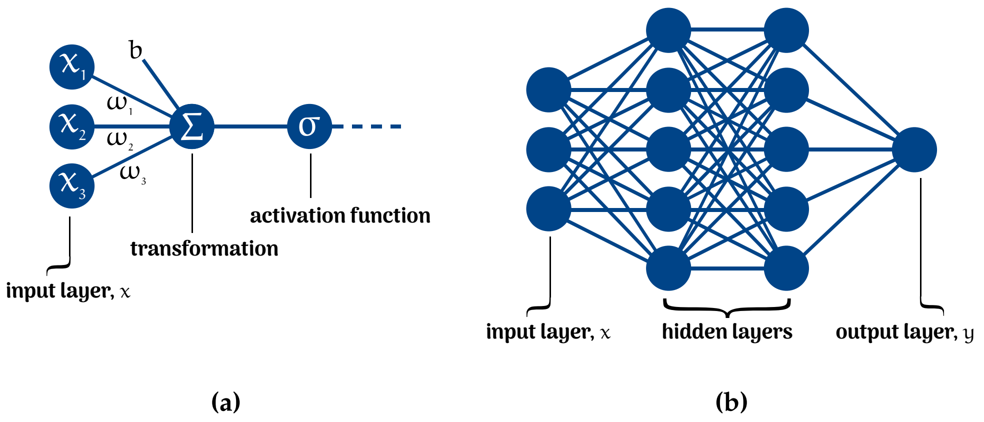


Figure 2.2. (a) The structure of an individual node within a network. **(b)** A deep neural network. The node consists of an input layer, a linear combination function node and an activation function, in this case the sigmoid function, σ.

non-linear transformations through many hidden layers, and by adding more hidden layers (constructing a deep neural network) it is possible to model increasingly complex and abstract features.

To update the weights and biases in linear regression a gradient descent algorithm can be used, however to update these values in an NN simple gradient descent cannot be applied. Updating the weights and biases in neural networks is done through a process called backpropagation.† Up until 1986 researchers struggled to train deep neural networks, but a paper by Hinton *et al.*[82] introduced this idea of backpropagation and opened up the field. In backpropagation, a network's output error is determined and the algorithm computes how much the previous hidden layers of nodes contributed to that error. It then calculates how much of these error contributions came from each node in the hidden layer before that, continuing in the same manner until it reaches the input layer. The

†This is only true for *feed-forward* neural networks. A feed-forward neural network is one in which the layers do not form, or contain, any loops. The NN that has been described so far is an example of such a feed-forward network. An example of an NN that does contain such a loop would be a *recurrent* neural network[79–81]

algorithm essentially measures the error gradient across all the weights in the network, and gradient descent is performed on those values.

2.2.1 VANISHING/EXPLODING GRADIENTS PROBLEM

The activation function within a neural network turns an unbounded input into a bounded output, and allows for a non-linear transformation. The first example of an activation function is the sigmoid function, $\sigma(x)$, that returns a value between 0 and 1 (Figure 2.3),

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \tag{2.3}$$

Large negative numbers will return 0 and large positive numbers will return 1. After the breakthrough with backpropagation, the sigmoid function was the most popular activation function in the implementation of neural networks. However, issues in training were discovered such that the gradient tended to get smaller and smaller as the algorithm moved to the earlier layers in the backpropagation algorithm, and the gradient descent algorithm ends up leaving these parameters virtually unchanged. This lead to instances of training never converging and is called the *vanishing gradients problem*. There was also the issue of the *exploding gradients problem* that, expectedly, is the opposite issue. The product of multiple large gradients leads to an exponentially growing gradient, which in turn leads to very large updates in the parameters and convergence never being reached.

This is partially why deep neural networks had been abandoned until around 2010 when Glorot and Bengio[83] discovered that the outputs from layers further through the network were getting extremely large, eventually saturating the popular sigmoid activation function when combined with the most common initialisation technique for the weights and biases (random initialisation using a normal distribution with mean 0 and standard deviation 1). Figure 2.3 shows that very large or very small values return values close to zero and one, and the derivative at these points is close to zero. Thus backpropagation ends up working on too small of a gradient to start with, and has little to propagate back through the system. Glorot and Bengio proposed a few solutions to this problem. They

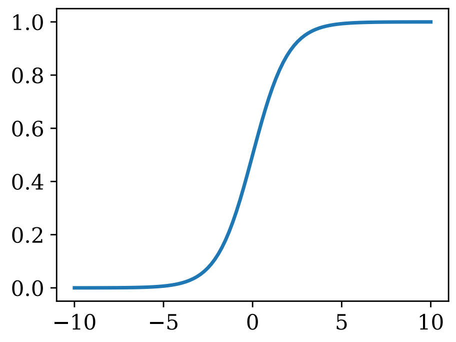


Figure 2.3. The sigmoid function, showing the small gradient of the curve as the values of x get larger, and the values of y get closer to zero and one.

state that if variance of the inputs was equal to the variance of the outputs, and the variance of the gradient was consistent throughout training then the vanishing gradients problem could be solved. These constraints are not possible to guarantee unless the number of nodes in the input and the output are the same, but they suggested a compromise. Instead of simply initialising the weights with a mean and standard deviation of 0 and 1, the weights should be selected from a normal distribution with a mean of 0 and a standard deviation equal to,

$$\sigma = \sqrt{\frac{2}{n_{\text{inputs}} + n_{\text{outputs}}}}$$

where n is the number of inputs and outputs to the layer of which the weights are being initialised. For situations where the activation function is logistic, this kind of initialisation can solve the problem of vanishing gradients.

As the activation function is partially to blame for this problem, other alternatives have also been suggested. Particularly, two of the most common activation functions used today are the Rectified Linear Unit[84] (ReLU - shown in figure 2.4a) and the hyperbolic tangent function[85]

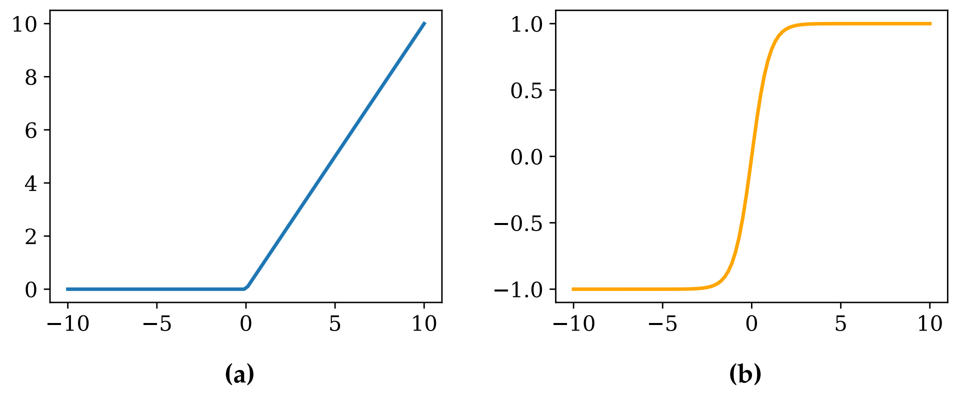


Figure 2.4. (a) The ReLU activation function, showing that it returns x when $x > 0$, and 0 when $x < 0$. **(b)** The tanh function, varying between -1 and 1.

(tanh, shown in figure 2.4b). ReLU is a function that returns x when $x > 0$, and 0 when $x \leqslant 0$. This does not saturate for positive values and is very fast to compute, thus making it a popular activation function. The tanh function is very similar in shape to the sigmoid function, but varies from -1 to 1 instead of 0 to 1. If ReLU or tanh are used, it is recommended that a different type of initialisation is used. For ReLU the normal distribution should have a standard deviation of

$$\sigma = \sqrt{2}\sqrt{\frac{2}{n_{\text{inputs}} + n_{\text{outputs}}}},$$

and for tanh the normal distribution should have a standard deviation of

$$\sigma = 4\sqrt{\frac{2}{n_{\text{inputs}} + n_{\text{outputs}}}}.$$

2.2.2 OPTIMISATION ALGORITHMS

Training deep neural networks with lots of hidden layers can be quite slow, and one area that can be adjusted to improve training speeds is the optimiser that minimises the loss function. By far the most popular optimiser is the adaptive moment estimation optimiser (or the Adam optimiser),[86] but it is worth exploring a few of the optimisers that came before it to fully understand the various aspects that it includes.

Momentum optimisation[87] If a ball was rolled down a hill it would be expected to slowly gain speed until it reached some terminal velocity determined by the friction and air resistance. This is the basic idea behind momentum optimisation, and contrasts the small regular steps that gradient decent takes.

In section 2.1.1, gradient descent was shown to update the parameters (θ) based on the gradient of the loss function calculated at each step [$\nabla_\theta J(\theta)$], then the gradient is discarded and the algorithm moves on. No information about previous gradients is retained, and if the current gradient is small the optimisation takes small steps. In contrast, momentum optimisation retains a portion of the gradient at each step in a momentum vector $\mathbf{m}$, and uses this vector to update the weights and biases instead.

$$\begin{aligned} &1.\quad \mathbf{m}_{i+1} \leftarrow \beta\mathbf{m}_i + \eta\nabla_\theta J(\theta), \\ &2.\quad \theta_{i+1} \leftarrow \theta_i - \mathbf{m}_{i+1}. \end{aligned}$$

A friction factor, β, is used to prevent the momentum from getting too large, and risking overshooting the minimum. This is typically set between 0 and 1 (low and high friction, respectively), and a common starting value is 0.9. In essence the gradient acts as an acceleration rather than a speed.

AdaGrad[88] Adaptive subgradient (AdaGrad) treats each parameter θ_i individually, updating the learning rates for each one.

$$\begin{aligned} &1.\quad \mathbf{s}_{i+1} \leftarrow \mathbf{s}_i + \nabla_\theta J(\theta) \otimes \nabla_\theta J(\theta), \\ &2.\quad \theta_{i+1} \leftarrow \theta_i - \eta\nabla_\theta J(\theta) \oslash \sqrt{\mathbf{s}_{i+1} + \varepsilon}. \end{aligned}$$

Step one involves calculating the square of the gradients for each parameter, $\mathbf{s}$, and step two is much like gradient descent. However, the gradient vector is scaled by a factor of $\sqrt{\mathbf{s} + \varepsilon}$, where ε is a term to avoid dividing by zero. This has the effect of decaying the learning rate for parameters with steeper gradients faster than those with shallower gradients, and better points the resulting vector towards the global minimum. Although this method works well for simple quadratic problems, it often stops too early when used with neural networks due to the learning rate getting very small very quickly.

RMSProp[89] To combat the problems highlighted with AdaGrad, Root Mean Square Propagation (RMSProp) only keeps the most recent gradients when updating $\mathbf{s}$ by adding an exponential decay, β, to step one.

$$\begin{aligned}
&1.\quad \mathbf{s}_{i+1} \leftarrow \beta \mathbf{s}_i + (1-\beta)\nabla_\theta J(\theta) \otimes \nabla_\theta J(\theta),\\
&2.\quad \theta_{i+1} \leftarrow \theta_i - \eta \nabla_\theta J(\theta) \oslash \sqrt{\mathbf{s}_{i+1} + \varepsilon}.
\end{aligned}$$

This method tends to outperform momentum optimisation techniques.

Adam[86] Returning to the start of this section, the adaptive moment estimation, or Adam optimiser combines the ideas of momentum optimisation with those of RMSProp, keeping track of an exponentially decaying average of both the past gradients and the past squared gradients.

$$\begin{aligned}
&1.\ \mathbf{m}_{i+1} \leftarrow \beta_1 \mathbf{m}_i + (1-\beta_1)\nabla_\theta J(\theta),\\
&2.\ \mathbf{s}_{i+1} \leftarrow \beta_2 \mathbf{s}_i + (1-\beta_2)\nabla_\theta J(\theta) \otimes \nabla_\theta J(\theta),\\
&3.\ \mathbf{m}_{i+1} \leftarrow \frac{\mathbf{m}_{i+1}}{1-\beta_1^T},\\
&4.\ \mathbf{s}_{i+1} \leftarrow \frac{\mathbf{s}_{i+1}}{1-\beta_2^T},\\
&5.\ \theta_{i+1} \leftarrow \theta_i - \eta \mathbf{m}_{i+1} \oslash \sqrt{\mathbf{s}_{i+1} + \varepsilon},
\end{aligned}$$

where T is the iteration number. Steps one and two are almost identical to momentum optimisation and RMSProp, as is step five. Steps three and four are there to boost $\mathbf{m}$ and $\mathbf{s}$ at the start of training, as they are initialised at 0 and therefore biased towards 0 to start with.

Where gradient descent takes small steps at every point, an Adam optimiser will take larger steps the steeper the gradient and smaller steps at shallower gradients - leading to much faster optimisation.

2.2.3 OTHER HYPERPARAMETERS

When it comes to improving the performance of neural networks, there are a few more methods available to reduce/avoid overfitting.

Regularisation Regularisation aims to address the bias vs variance trade-off previously mentioned by introducing parameters to the loss function to scale the learned parameters by a set amount. In general there are two types of regularisation that get implemented. The first is L1 regularisation (sometimes called lasso regression) which adds the absolute value of magnitude of the coefficients to the loss function. Taking equation 2.1 and adding this term, gives the following,

$$J = \frac{1}{n}\sum_{i=1}^{n}(y_i - (\sum_{j}^{p} \omega_j x_{ij} + b_j))^2 + \lambda \sum_{j}^{p} |\omega_j|.$$

Here, x is now a feature vector of p features, and λ is a scaling parameter that controls the strength of the regularisation. This has the effect of shrinking the coefficients of the least important features to be equal to zero, essentially performing a kind of feature selection, and reducing overfitting. The second type of regularisation is called L2 regularisation, or ridge regression. Instead of the absolute value of magnitude of the coefficients, L2 regression adds the squared magnitude of the coefficients to the loss function,

$$J = \frac{1}{n}\sum_{i=1}^{n}(y_i - (\sum_{j}^{p} \omega_j x_{ij} + b_j))^2 + \lambda \sum_{j}^{p} \omega_j^2.$$

This also shrinks the least important features, but they will never equal zero, and thus all features are used in predictions.

The type of regression, and the value of λ will need to be decided for each individual problem, and are thus 'adjustable parameters', or hyperparameters.

Early stopping If a neural network is not trained for enough epochs then it is likely to be underfit, and make poor predictions on the training *and* testing sets. However, if it is trained for too long, then it can become overfit, and *only* make accurate predictions on the training set, while underperforming on the test set. Early stopping halts training after the error on the validation set (usually half of the test set) begins to increase (shown in Figure 2.5), or if the error has not improved for a determined set of cycles. This stops the network becoming overfit to the training set, while allowing it to train for as long as possible.

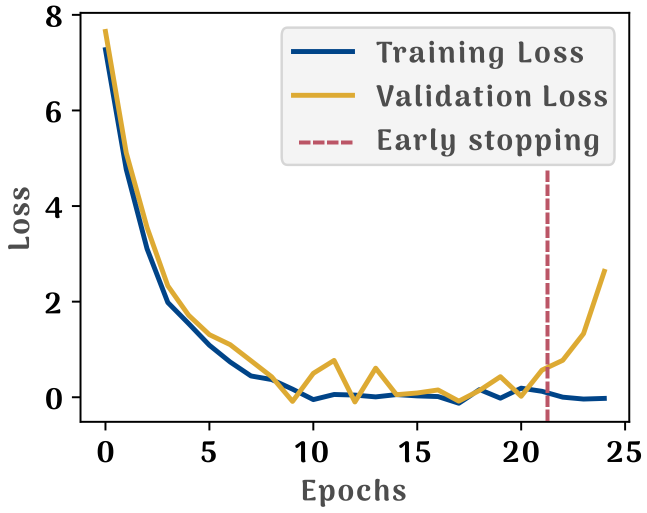


Figure 2.5. An example of a fake dataset, where the validation loss initially follows the training loss well, but after ~20 epochs the network begins to get over fit to the training data and the validation loss begins to increase rapidly. After early stopping, the network parameters as they were an epoch before stopping would usually be taken as the trained network parameters.

Dropout One way of improving the performance of a machine learning algorithm, reducing overfitting, and forcing it to generalise, is by averaging the output of many separately trained models, and if all the combined models are different then this performance gain can be quite large.[90] However, for larger neural networks this is impractical as 'different' NNs will either have varying architectures or be trained on entirely different datasets, both of which can be problematic. Hyperparameter optimisation of large networks can be difficult, and if the datasets are hard to generate in the first place then there may not be enough data to simply use subsets. Instead two datasets would need to be generated which is, of course, undesirable.

Dropout aims to solve both of these problems by temporarily disconnecting nodes[†] in the neural network, which changes the architecture for a single epoch at a time, allowing the final trained network to be an approximation of a combination of exponentially many neural networks.[90] A visualisation of this concept is shown in Figure 2.6, highlighting the

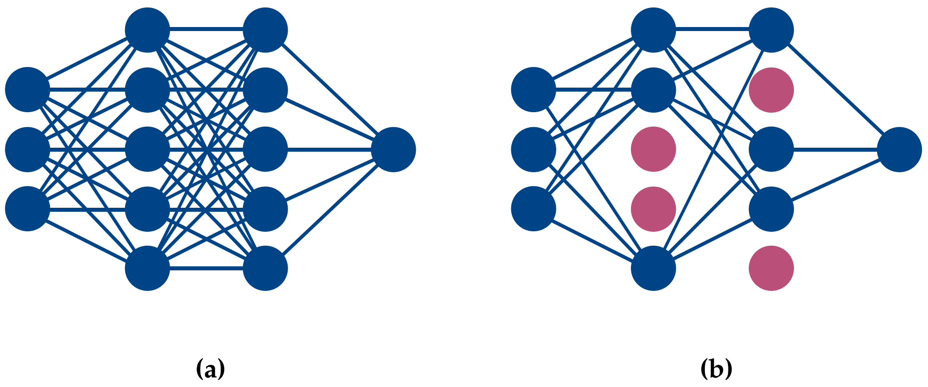


Figure 2.6. **(a)** A two-hidden-layer neural network **(b)** The same neural network with some of the nodes switched off (dropped).

different connectivity seen in the dropped out network. Srivastava *et al.* report that dropout "has been proved to be highly successful, with even the most state of the art models getting a 1 - 2 % accuracy boost."[90]

Dataset selection One of the main challenges in machine learning is acquiring a suitable set of training data. If not enough data exist then you have to generate more, and if you have enough data then you need

[†]The rate of dropout for each node is determined randomly, with the probability of a node turning off for any given training cycle being set by the hyperparameter p. Srivastava *et al.* state that $p = 0.5$ "seems to be close to optimal for a wide range of networks and tasks." The input layer should usually be left with p closer to one, but this is again a hyperparameter to adjust.[90]

to select a sufficiently representative subset of the data to train on. In computational chemistry, generating a large number of molecular structures is a task that can be easily automated for small systems at lower levels of theory, but as discussed in Chapter 1, the computational cost for larger systems at higher levels of theory gets prohibitively large, so fewer training data can be generated.

As machine learning algorithms cannot accurately extrapolate beyond what they have been taught, it is important to ensure that the training set encompasses the extremes of the data to be learned. For example, extreme bond elongations or contractions, or bond angle changes. On a related but separate note it is important to check for 'holes' in the training data. Figure 2.7 shows a dataset characterised by two features. For certain values of these features there is no training data, and a model predicting the label for a test point in this area may not provide any meaningful answers. One way to address both of these problems is

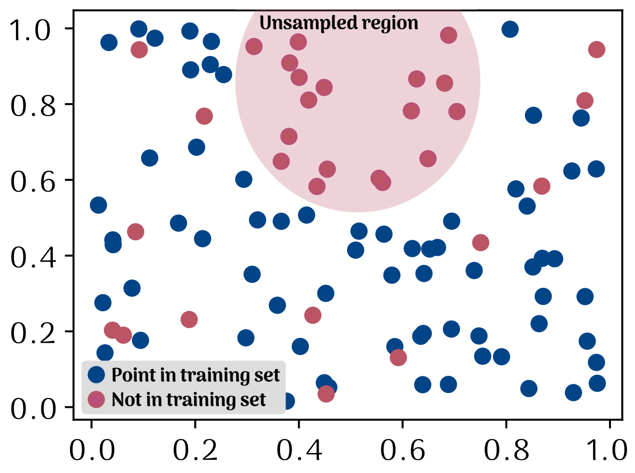


Figure 2.7. A dataset characterised by two features. Blue datapoints have been selected and placed in the training set. The red datapoints are missing from the training set. The red 'unsampled region' highlights an area of the dataset with no points in the dataset, and any new predictions made for these values of the features are likely to be incorrect.

through more intuitive configuration space sampling, which comes in the form of active sampling. Farthest point sampling is the method of choosing the point furthest away from a starting position, adding that to the training set, then selecting the next furthest point from this new position and repeating this process until some stopping criteria are met. While structure-based sampling aims to maximise the Euclidean distance between points in the training set.[91]

2.3 MACHINE LEARNING IN CHEMISTRY

The last few decades have seen the application of machine learning to a rising number of problems spanning numerous industries. These include day to day examples in machine translation,[55] video game AI,[56] self-driving cars,[92] and recommendations across social media and video streaming.[93] In the sciences, ML has been applied across many fields including: medical diagnostics,[94–98] bioinformatics,[99–102] particle physics,[103] nano-sciences,[104] and robotics.[70,105,106]

In chemistry, machine learning has found use across the whole field, from simple learning of molecular properties such as toxicity[107] and reaction rates,[108,109] to thermodynamic property prediction,[110–112] and drug and materials design.[113,114] For a thorough review of the application of machine learning to areas of chemistry separate to PESs, the interested reader is pointed towards the work of Keith *et al.*[115]

Machine learned potentials A large body of work surrounds so called 'machine learned potentials',[116–119] and it is important to briefly explain how these differ from 'potential energy surfaces', considering the similar names. These are specifically attempts at replacing the solution of the Schrodinger equation entirely with a potential trained on a large volume of *ab initio* data. A PES does not aim to circumvent the Schrödinger equation, in fact many thousands of *ab initio* points of data are needed to fit one. The aim of an ML potential is to be general enough to be transferable to a number of different systems.

A brief note on representation Although important in the wider applications of machine learning, the representation of the chemistry to the algorithm is crucial for successful application of ML techniques to computational chemistry. This might seem simple at first; a molecule is wholly characterised by the number of electrons and the positions and charges of the nuclei. This is, however, insufficient to describe the system for machine learning, as the swapping of two identical atoms within, or the translation and rotation of the molecule will not give rise to any changes to properties such as energy and this invariance needs to be accounted for in the feature vector provided to the machine learning algorithm. If an insufficient representation is used, then the predicted energy for two identical molecules, where one is slightly rotated compared to the other, could be wildly different. Even though it is clear that this should not be the case.

A simple set of bond lengths is insufficient as it contains no angle or connectivity information, and can lead to several different structures (a one to many relationship). The use of XYZ coordinates introduces enough information to give a wholly unique structure, but simply translating the molecular coordinates in space, or rotating them through space, numerically gives a completely new representation of a single geometry with a unique energy, with potentially infinite different representations of the same molecule (a many to one relationship).

Internal coordinates (or z-matrices) are the simplest form of representation that is invariant to these operations, characterising atoms relative to an internally consistent, central origin point, through (r, θ, ω). Where, r is a bond length, θ is a bond angle, and ω is a dihedral angle. In this case each set of internal coordinates characterises exactly one molecule (a one to one relationship). It is also important to think about the system being explored, as symmetries within the molecule may be able to lead to reductions in training sets by training for only one component of the symmetry, or even an increase of the size of the training set with very little effort by duplicating the dataset for each component of the symmetry.

A more complex representation has been described by Bartók *et al.*, predominantly for use with the mathematical potentials. They proposed a new way to represent chemical environments, which they named "Smooth Overlap of Atomic Positions" or, SOAP.[120] They showed that the similarity measure $K(q, q')$, where q is a parameter characterising the local atomic environment, is more important than the geometric descriptor itself. Therefore, SOAP bypasses the descriptor entirely and instead builds a similarity measure between neighbouring environments directly. However, for ML-PESs these are not useful, nor necessary, as they do not contain the granularity in geometry required to build a PES.

2.3.1 POTENTIAL ENERGY SURFACE GENERATION

As outlined in Chapter 1, conventional methods of PES fitting have a number of downsides. Methods such as RKHS, IMLS, and MSI require large numbers of expensive *ab inito* calculations, and to use the DMBE method an intimate knowledge of the form of the PES is already required. Chapter 1 also identified three main ways of speeding up PES generation,

1. speed up the *ab initio* data generation (basis sets),
2. speed up the fitting of the surface (ML algorithm),
3. reduce the amount of data needed to fit/train the surface itself,

and it is worth justifying the use of an ML algorithm to fit the surface, as per point two. This has been achieved in numerous examples through the use of machine learning techniques, and specifically, neural networks have found great success in their application to this field.

Some of the earliest examples of NN PES fitting date back 25 years. Blank *et al.* fit a neural network for CO on a nickel surface, introducing many important ideas surrounding NN PESs, including the choice of network size, considerations on overfitting, and the quality of the training data.[121] Brown *et al.* reported impressive mean errors of 25 cm^{-1} on an NN-PES for four atom van der Waals clusters.[122] However, the use of neural networks was limited, and seen mostly as a computational curiosity due to the perceived low accuracy of neural networks,[123] with Duch *et al.* going so far as to say "For many problems [. . .] an attempt to use neural networks [. . .] will lead to low accuracy and lengthy computations".[124] Many of the problems stemmed from the simplicity of the neural networks being trained, and the examples presented so far all used single layer neural networks. However, with the development of backpropagation (discussed in section 2.2), deep neural networks became more prevalent and the first multi layer NN-PESs were reported.[125,126]

Other ways to improve the accuracy of these surfaces are to ensure that the surface follows the symmetry of the system, meaning it is invariant to the permutation of identical atoms. Prudente *et al.* were the first to introduce this idea and implemented it through the use of a symmetrised hidden layer,[127] while Gassner *et al.* used symmetrised inputs for H_2O – Al^{3+} – H_2O three body interactions.[128] Lorenz *et al.* used symmetrised inputs to model the surface interaction of H_2 with the (2×2) potassium covered Pd(1 0 0) surface,[129] while Filho *et al.* chose to fit only the symmetrically unique sections of the surface for the H_3^+ molecule.[130]

A major breakthrough in the field came from Manzhos *et al.* where they showed that it was possible to fit an NN-PES for three and four atom molecules to levels of accuracy suitable for vibrational analysis.[131] They report accuracies on the order of 1 cm^{-1} on the global fit of the test set for H_2O, HOOH, and H_2CO. This was achieved by firstly fitting a rough surface using an NN, followed by a second surface fit by correcting the energy with high accuracy calculation on important fitting points. Manzhos has continued to work in this field and had recently published an in-depth review alongside Ihara and Carrington on "Machine learning for vibrational spectroscopy"[132] covering the use of machine learning techniques to directly calculate vibrational spectra, improve spectra calculated by other means, and aid with the interpretation of vibrational spectra.

2.3.2 SOFTWARE DEVELOPMENTS

Historically, one of the aspects holding ML back has been the complexity of implementation. However, as libraries such as TensorFlow[133] and Keras[134] have continued to be developed, and the breadth of information available in the form of easy to understand tutorials has grown, writing and applying machine learning algorithms to new problems has become significantly easier. Specifically, in the field of potential energy surfaces, Abbott *et al.* recently released 'PES Learn', a Python library capable of training neural networks and gaussian process regression algorithms to generate potential energy surfaces from datasets of molecular geometry and energies.[135] The program automatically performs hyperparameter searches, then fits a model using PyTorch[136] with the best possible hyperparameters searched over. It is then presented in an easy to interrogate model file that makes probing the trained surface very easy.

This ease of access, however, does not come without any downsides. In the introduction of their 2021 review titled "Combining Machine Learning and Computational Chemistry for Predictive Insights Into Chemical Systems", Keith *et al.* highlight four points from a poll they posed to the scientific community.[115]

1. ML methods are becoming less understood while they are also more regularly used as black box tools.

2. Many publications show inadequate technical expertise in ML (e.g., inappropriate splitting of training, testing, and validation sets).

3. It can be difficult to compare different ML methods and know which is the best for a particular application or whether ML should even be used at all.

4. Data quality and context are often missing from ML modelling, and data sets need to be made freely available and clearly explained.

It is clear that although ML is easier to implement, there is a need for more careful development and greater understanding.

The importance of good data In a real test of the field, during the global pandemic of COVID-19 hundreds of machine learning algorithms were developed in an attempt to help diagnose or treat the virus. Concerningly, not a single one of these models had any significant positive impact, as highlighted by two major studies released since the pandemic began. Wynants *et al.* explored 232 algorithms designed to diagnose patients, or predict the severity of the disease for individuals and found that none of them were fit for clinical use, while only two showed any real promise for future work.[137] Similarly, Driggs *et al.* studied 415 published tools for diagnosis of COVID-19 from medical imaging, and again found that none of them were viable for real world use.[138] This sentiment has also been echoed by The Alan Turing Institute in their 2021 report summarising workshops held in 2020, concluding that ML tools had very little impact on the fight against COVID-19.[139]

This apparent failing of the scientific community highlighted some of the areas of major problems surrounding real world application of the machine learned chemistry. In their review, Driggs *et al.* identify the datasets as some of the largest sources of error. They highlight problems with 'Frankenstein' datasets that have been spliced together, often containing several duplicates, and issues with feature selection/data cleaning.[138] For example, many studies unwittingly used lung scans of children without COVID-19 as examples of non-COVID cases, leading to algorithms that just identified children, rather than COVID. Some algorithms were found to be picking up on the fonts used in the scans, linking fonts from hospitals with larger volumes of serious cases to be a predictor for COVID.

All this highlights that although machine learning has been successfully applied to the field of computational chemistry, the advent of new technologies that make them easier to train and test[135] exacerbates the need for care to be taken when implementing ML algorithms, and curating the datasets for them.

3 | Electronic structure theory

In Chapter 1 it was established that the molecular geometry of a system can be used in conjunction with the Schrödinger equation to calculate the electronic energy of the system. The form of this equation, and the application of various approximations such as the Born-Oppenheimer approximation are all part of a wider 'electronic structure theory' that is essential to many modern methods of computational exploration of the chemical world. The resulting equations of electronic structure theory enable the calculation of energies of molecules based on the positions of the atoms within them. This in turn can lead to the predictions of molecular geometries, reactivities, and properties, to name a few. In the context of potential energy surface generation, it is essential that the basis for these predictions is understood, so that the limitations and drawbacks of the various theories are taken into account, and where appropriate, mitigated for.

3.1 THE SCHRÖDINGER EQUATION

The classical world can be described mathematically through Newton's laws of motion. Newton's second law describes the behaviour of a classical system through time,

$$\mathbf{F} = m\frac{d^2}{dt^2}\mathbf{s}$$

which states that the force acting upon the system, $\mathbf{F}$, is equal to the mass, m, times the second derivative of the position, $\mathbf{s}$, with respect to time (acceleration). However, these laws break down when looking at a quantum mechanical system, and instead the state of a such a system is described through the use of the *time-dependent* Schrödinger equation,

$$i\hbar\frac{\partial\Psi(\mathbf{r},t)}{\partial t} = \hat{H}\Psi(\mathbf{r},t). \tag{3.1}$$

This equation is a linear partial differential equation, dependent on time, that operates on the function Ψ. This function, $\Psi(\mathbf{r},t)$, is defined to be a wavefunction; it is a quantum mechanical descriptor of a system that depends on both position, $\mathbf{r}$, and time, t (the form of the wavefunction will be explored in detail in sections 3.1.2 and 3.3). The Hamiltonian, $\hat{H}$, is an operator that can be written more fully as,

$$i\hbar\frac{d\Psi(\mathbf{r},t)}{dt} = -\frac{\hbar^2}{2m}\nabla^2\Psi(\mathbf{r},t) + \hat{V}\Psi(\mathbf{r},t). \tag{3.2}$$

The term ∇^2 here is the Laplacian, a sum of the partial second derivatives with respect to each of the dimensions of $\mathbf{r}$. That is, for a vector of $\mathbf{r}^n = x_1, x_2, \ldots, x_n$,

$$\nabla^2 = \sum_{i=1}^{i}\frac{\partial^2}{\partial x_n^2},$$

where $\hbar$ is the reduced Planck constant ($h/2\pi$, $h = 6.62607015 \times 10^{-34}$ m^2 kg s^{-1}) and m is the mass of the system. The first term in equation 3.2 is a *kinetic* energy term, and the second a *potential* energy term, with $\hat{V}$ as the potential energy operator. It is possible to use the separation of variables to develop this equation, such that $\Psi(\mathbf{r},t)$ is the product of two single variable functions,

$$\Psi(\mathbf{r},t) = \Psi(\mathbf{r})\Phi(t),$$

where $\Phi(t)$ is a wavefunction dependent only on time. Substitution into equation 3.2 gives,

$$i\hbar\frac{d\Phi(t)}{dt}\Psi(\mathbf{r}) = -\frac{\hbar^2}{2m}\nabla^2\Psi(\mathbf{r})\Phi(t) + \hat{V}(\mathbf{r},t)\Psi(\mathbf{r})\Phi(t),$$

and when dividing by $\Psi(\mathbf{r})\Phi(t)$ this gives,

$$i\hbar\frac{d\Phi(t)}{dt}\frac{1}{\Phi(t)} = -\frac{\hbar^2}{2m}\nabla^2\Psi(\mathbf{r})\frac{1}{\Psi(\mathbf{r})} + \hat{V}(\mathbf{r},t).$$

For potentials that do not depend on time there are two sides to an equation that each contain only one variable. As it has been stated that position and time are independent, each side must be equal to a constant, and this gives two differential equations that can be solved:

$$i\hbar\frac{d\Phi(t)}{dt} = E\Phi(t), \tag{3.3}$$

$$-\frac{\hbar^2}{2m}\nabla^2\Psi(\mathbf{r}) + \hat{V}\Psi(\mathbf{r}) = E\Psi(\mathbf{r}). \tag{3.4}$$

The differential equation dependent only on time (equation 3.3), can be solved such that,

$$\Phi(t) = Ce^{\frac{-iEt}{\hbar}},$$

and incorporating the constant C into $\Psi(\mathbf{r})$ gives,

$$\Psi(\mathbf{r}, t) = e^{\frac{-iEt}{\hbar}}\Psi(\mathbf{r}),$$

showing that $\Psi(\mathbf{r}, t)$ is separable. Equation 3.4 is therefore a positional-dependent differential equation, and is called the *time-independent* Schrödinger equation, often written in its iconic form,

$$\hat{H}\Psi = E\Psi. \tag{3.5}$$

This is known as an eigenfunction/eigenvalue equation, where an operator ($\hat{H}$) acts on an eigenfunction (Ψ), to return the same eigenfunction multiplied by constant called an eigenvalue (E).[140]

3.1.1 SEPARATION OF THE NUCLEAR AND ELECTRONIC MOTION

As shown in equation 3.4, the Hamiltonian operator, $\hat{H}$, is the sum of both the kinetic energy ($\hat{T}$) and potential energy ($\hat{V}$) operators,

$$\hat{H} = \hat{T} + \hat{V}.$$

For a system of electrons, i, and nuclei, A, the kinetic energy operator is,

$$\hat{T} = -\sum_i \frac{\hbar^2}{2m_e}\nabla_i^2 - \sum_A \frac{\hbar^2}{2M_A}\nabla_A^2, \tag{3.6}$$

where m_e is the mass of an electron, and M_A is the mass of nucleus A. The potential energy operator is,

$$\hat{V} = \frac{e}{4\pi\varepsilon_0}\left(-\sum_{i,A}\frac{Z_A}{r_{iA}} + \sum_{i>j}\frac{1}{r_{ij}} + \sum_{A>B}\frac{Z_A Z_B}{R_{AB}}\right), \tag{3.7}$$

in which e is the charge of a proton, Z is nuclear charge, r is a distance involving electrons, R is a distance involving only nuclei, and ε_0 is the permittivity of free space. It is not possible to solve the Schrödinger equation exactly for a system with more than one electron, and several approximations must be made to simplify the problem for larger systems. The first of which is the Born-Oppenheimer approximation.[8]

The Born-Oppenheimer approximation It is possible to separate a Hamiltonian into two terms, where the total eigenfunction is a product of the individual eigenfunctions, and the total eigenvalue is a sum of the individual eigenvalues. If we separate the Hamiltonian into its electronic and nuclear parts, $\hat{H}_{el}$ and $\hat{H}_N$, the Schrödinger equation can be expressed as:

$$\hat{H}\Psi(\mathbf{r},\mathbf{R}) = (\hat{H}_{el} + \hat{H}_N)\Phi_{el}(\mathbf{r};\mathbf{R})\Phi_N(\mathbf{R}),$$

where $\Phi_N(\mathbf{R})$ is the nuclear wavefunction and $\Phi_{el}(\mathbf{r};\mathbf{R})$ is the electronic wavefunction parametrically dependent on the positions of the nuclei. That is, $\Phi_{el}(\mathbf{r};\mathbf{R})$ depends on the positions of the nuclei, but the nuclear positions do not appear explicitly in the wavefunction. It would be correct to notice that this separation should not be possible, due to the electron-nuclear potential term, $\hat{V}_{eN} = -\sum_{i,A}\frac{Z_A}{r_{iA}}$, in the potential energy operator (equation 3.7). However, the Born-Oppenheimer approximation states that due to the very large difference in mass between electrons and nuclei, and therefore the timescales of their motion, the movement of electrons can be de-coupled from the nuclear Hamiltonian, and two separate eigenfunctions can be used.

As $\hat{T}_N$ is so much smaller than $\hat{T}_e$ due to the large mass difference, $\hat{H}_N$ can be neglected entirely and the expression for the electronic Hamiltonian, or *clamped-nucleus* Hamiltonian, can be given as:

$$\hat{H}_{el} = \hat{T}_e(\mathbf{r}) + \hat{V}_{eN}(\mathbf{r};\mathbf{R}) + \hat{V}_{ee}(\mathbf{r}) + \hat{V}_{NN}(\mathbf{R}).$$

Which, using atomic units ($e \equiv 1$, $m_e \equiv 1$, $\hbar \equiv 1$, $4\pi\epsilon_0 = 1$), can be expanded to:

$$\hat{H}_{el} = -\frac{1}{2}\sum_i \nabla_i^2 - \sum_{i,A} \frac{Z_A}{r_{iA}} + \sum_{i>j} \frac{1}{r_{ij}} + \sum_{A>B} \frac{Z_A Z_B}{R_{AB}}. \tag{3.8}$$

From this point forward, all Hamiltonians are assumed to be electronic.

3.1.2 BUILDING A MANY ELECTRON WAVEFUNCTION

The electronic Schrödinger equation is not able to be solved for many body problems, and exact solutions only exist for one-electron wavefunctions. Therefore, to construct an n-electron wavefunction for a polyatomic molecule, it is assumed initially that electrons exist independent of those around them, and that they can be described by one particle wavefunctions called spinorbitals. For a single electron, a spinorbital is defined by the product of both a spatial wavefunction and a spin wavefunction:

$$\phi(i) = \phi(\mathbf{r}_i)\alpha(s_i),$$
$$\bar{\phi}(i) = \phi(\mathbf{r}_i)\beta(s_i),$$

where ϕ is a one particle wavefunction, α and β are spin up (spin magnetic quantum number, $m_s = +\frac{1}{2}$) and spin down ($m_s = -\frac{1}{2}$), respectively, and s_i is the spin quantum number ($\frac{1}{2}$ for an electron). It is not possible to simply express an n-electron wavefunction as a product of all one electron spinorbitals due to the Pauli antisymmetry principle,[141] which states that two wavefunctions must be antisymmetric to the exchange of the position of any two electrons.This is achieved by instead combining orbitals in a sum of all permutations of electrons in orbitals, and, to build an approximate wavefunction, $\widetilde{\Phi}$, for a system of n electrons,

n spinorbitals are combined in an antisymmetric Slater determinant:

$$\widetilde{\Phi}(1,2,...,n) = \frac{1}{\sqrt{n!}}\begin{vmatrix} \phi_1(1) & \phi_2(1) & \cdots & \phi_n(1) \\ \phi_1(2) & \phi_2(2) & \cdots & \phi_n(2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(n) & \phi_2(n) & \cdots & \phi_n(n) \end{vmatrix}.$$

Here, $\frac{1}{\sqrt{n!}}$ is a normalisation constant to ensure that,

$$\int |\widetilde{\Phi}(\mathbf{r})|^2 d\tau = 1, \tag{3.9}$$

where $|\Phi(r)|^2$ is the probability density of finding an electron at position $\mathbf{r}$. When integrating this over all space it is understood that the probability should be 1. This can be written in a shorthand as,

$$\widetilde{\Phi}(1,2,...,n) = |\phi_1(1)\phi_2(2)...\phi_n(n)\rangle, \tag{3.10}$$

where only the principal diagonal elements of the determinant are written out, the normalisation is assumed, and the spinorbitals are $\phi_i(i)$, which define electron i as being in orbital ϕ_i. This, therefore, implies the Pauli exclusion principle: no two electrons can exist in the same state at the same time.[141] If two electrons are in the same spin orbital, the resulting determinant will be zero and the wavefunction vanishes.

Orthonormality[142] For a given operator, $\hat{A}$, if its eigenvalues correspond to observables (that is, the eigenvalues of the Hamiltonian operator are orbital energies), then those eigenvalues must be real. As such the operator must be Hermitian, which means that the complex conjugate‡ of the operator ($\hat{A}^\dagger$) is equal to the operator itself,

$$\begin{aligned} \hat{A}^\dagger &= \hat{A}, \\ \int \psi^* \hat{A}\psi d\tau &= \int (\hat{A}^\dagger \psi^*)\psi d\tau, \end{aligned} \tag{3.11}$$

‡The complex conjugate of an imaginary number, $x + yi$, is $x - yi$. The complex conjugate transpose of a matrix reflects the matrix across the principal diagonal, and replaces each element with its own complex conjugate.

where here, ψ^* is the complex conjugate of ψ. To prove this, consider the following eigenvalue equation and its complex conjugate,

$$\hat{A}\psi = a\psi, \tag{3.12}$$

$$\hat{A}^\dagger\psi^* = a^*\psi^*. \tag{3.13}$$

In this scenario $a = a^*$ because they are real. Thus by left multiplying equation 3.12 by ψ^* and equation 3.13 by ψ, then integrating over all space gives,

$$\int \psi^*\hat{A}\psi d\tau = a \int \psi^*\psi d\tau,$$

$$\int \psi\hat{A}^\dagger\psi^* d\tau = a \int \psi\psi^* d\tau,$$

and if the eigenfunctions are normalised then $\int \psi^*\psi d\tau = 1$, which gives the following,

$$\int \psi^*\hat{A}\psi d\tau = \int \psi\hat{A}^\dagger\psi^* d\tau = a.$$

Since these functions commute,[‡] this can be re-written, and equation 3.11 can be seen to be true,

$$\int \psi^*\hat{A}\psi d\tau = \int (\hat{A}^\dagger\psi^*)\psi d\tau.$$

Returning to equations 3.12 and 3.13, if there are two different eigenfunctions, that is to say,

$$\hat{A}\psi_i = a_i\psi_i, \tag{3.14}$$

$$\hat{A}\psi_j = a_j\psi_j, \tag{3.15}$$

then by taking the complex conjugate of equation 3.14, left multiplying by ψ_j, left multiplying equation 3.15 by ψ_i^*, and integrating over all space, the following is shown,

$$\int (\hat{A}\psi_i)^*\psi_j d\tau = a_i \int \psi_i^*\psi_j d\tau,$$

$$\int \psi_i^*\hat{A}\psi_j d\tau = a_j \int \psi_i^*\psi_j d\tau.$$

[‡]functions commute if $xf(x) = f(x)x$

Recalling equation 3.11, the left hand sides of these equations are equal, which gives

$$(a_i - a_j)\int \psi_i^* \psi_j \, d\tau = 0.$$

If $i \neq j$ then the integral has to be zero, yielding,

$$\int \psi_i^* \psi_j \, d\tau = 0,$$

defining the wavefunctions to be orthogonal, and the two eigenfunctions are perpendicular to each other. If $i = j$, then

$$\int \psi_i^* \psi_j \, d\tau = 1,$$

as long as the wavefunctions are normalised. This defines them to be *orthonormal*, and can be written in terms of the Kronecker delta[143]

$$\int \psi_i^* \psi_j \, d\tau = \delta_{ij} = \begin{cases} 1, & \text{if } i = j, \\ 0, & \text{if } i \neq j. \end{cases} \tag{3.16}$$

A set of orthonormal wavefunctions forms an orthonormal basis, and in the context of computational chemistry all wavefunctions are assumed to form this orthonormal basis set.

Bra-ket notation It's useful here to introduce some mathematical notation called bra-ket notation.[144] Below is an example of a *ket*. This is a way of representing a column vector† such that,

$$|a\rangle = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}.$$

The corresponding *bra*, $\langle a|$, is the conjugate transpose of the ket,

$$\langle a^*| = [a_1^*, a_2^*, \ldots, a_n^*],$$

†The wavefunctions of a system are vectors and can be expressed in bra-ket notation.

and must follow the standard matrix rules for complex conjugates. The dot product of the two values is expressed as the following,

$$\langle a^*|a\rangle = [a_1^*, a_2^*, \ldots, a_n^*] \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}. \tag{3.17}$$

Equation 3.17 can also be represented in integral form:

$$\langle a^*|a\rangle = \int a^*(x)a(x)dx.$$

Evaluating the energy If the electronic energy of the system can be given by the expectation value of the electronic Hamiltonian,

$$E_{el} = \langle\Phi|\hat{H}_{el}|\Phi\rangle,$$

then for symmetric energy expressions the true ground state energy is given by the variational theorem,

$$E_{el} = \frac{\langle\widetilde{\Phi}|\hat{H}_{el}|\widetilde{\Phi}\rangle}{\langle\widetilde{\Phi}|\widetilde{\Phi}\rangle} \geqslant E_0. \tag{3.18}$$

By grouping one electron terms in the Hamiltonian together, we can define a core Hamiltonian, $\hat{h}_i$,

$$\hat{h}_i = -\frac{1}{2}\nabla_i^2 - \sum_A \frac{Z_A}{r_{iA}},$$

which, for a system at a fixed geometry (where $\hat{V}_{NN}$ is constant and hence implied moving forward), allows us to simplify equation 3.8 to,

$$\hat{H}_{el} = \sum_i \hat{h}_i + \sum_{i>j} \frac{1}{r_{ij}}. \tag{3.19}$$

For orbitals that are orthonormal, $\langle\widetilde{\Phi}|\widetilde{\Phi}\rangle = 1$ and we can substitute equation 3.19 into equation 3.18 to get,

$$E_{el} = \langle\widetilde{\Phi}|\sum_i \hat{h}_i + \sum_{i>j} \frac{1}{r_{ij}}|\widetilde{\Phi}\rangle \geqslant E_0.$$

By forming the Schrödinger equation this way, and remembering equation 3.10, we can now apply the Slater-Condon rules[145,146] to the n-dimensional system to express the integrals of n-dimensional wavefunctions, $\widetilde{\Phi}$, as sums over much smaller integrals of, at most, two molecular orbitals:

$$E_{el} = \sum_{i} \langle \phi_i | \hat{h}_i | \phi_i \rangle + \frac{1}{2} \sum_{ij} \left(\langle \phi_i \phi_j | \frac{1}{r_{ij}} | \phi_i \phi_j \rangle - \langle \phi_i \phi_j | \frac{1}{r_{ij}} | \phi_j \phi_i \rangle \right). \tag{3.20}$$

This means that instead of having to solve a $3n$-dimensional integral, we now just need to solve several integrals of both three and six dimensions. The above equation can be simplified by defining some terms that also help gain insight into the nature of the electronic energy:

$$\mathcal{J}_{ij} = \langle \phi_i \phi_j | \frac{1}{r_{ij}} | \phi_i \phi_j \rangle,$$
$$\mathcal{K}_{ij} = \langle \phi_i \phi_j | \frac{1}{r_{ij}} | \phi_j \phi_i \rangle.$$

Here, $\mathcal{J}_{ij}$ is the Coulomb matrix, which governs the repulsion of electrons as they interact and is an electrostatic interaction. $\mathcal{K}_{ij}$ is the exchange matrix that is an entirely quantum mechanical phenomenon. It can be interpreted as electrons with the same spin tending to avoid each other.

Using these definitions (and equation 3.20) we can write an equation that defines the Hartree-Fock energy,[147,148]

$$E_{HF} = \sum_{i} h_i + \frac{1}{2} \sum_{ij} (\mathcal{J}_{ij} - \mathcal{K}_{ij}) \geqslant E_0. \tag{3.21}$$

3.1.3 CALCULATING ORBITAL ENERGIES

As equation 3.21 is $\geqslant E_0$, the Hartree-Fock energy can be found by minimising E_{HF} using Lagrange's method of undetermined multipliers. This is a simple method to find the local minimum within a function that is subject to one or more constraints. We define the Lagrangian functional $\mathcal{L}$ as

$$\mathcal{L} = E_{HF} - \sum_{ij} \epsilon_{ij} \left(\langle \phi_i | \phi_j \rangle - \delta_{ij} \right),$$

where ϵ_{ij} are the undetermined Lagrange multipliers, $\langle\phi_i|\phi_j\rangle$ is the overlap matrix, and δ_{ij} is the Kronecker delta. To minimise this function we set $\partial\mathcal{L}/\partial\phi_i = 0$, and by doing so we arrive at the Hartree-Fock equations,

$$h_i\phi_i + \sum_{j\neq i}(\mathcal{J}_{ij} - \mathcal{K}_{ij})\phi_i = \epsilon_i\phi_i. \tag{3.22}$$

We can see that ϵ_i are the Hartree-Fock orbital energies, corresponding to the wavefunctions of electrons i. By introducing a new operator called the Fock operator,

$$\hat{f}_i = h_i + \sum_j \mathcal{J}_{ij} - \mathcal{K}_{ij},$$

we can simplify equation 3.22 to

$$\hat{f}_i\phi_i = \epsilon_i\phi_i \tag{3.23}$$

where the orbital energies can now clearly be seen as eigenvalues of the Fock operator.

3.1.4 THE HARTREE-FOCK-ROOTHAAN EQUATIONS

We have shown that by operating on a Slater determinant of molecular wavefunctions, we can arrive at the orbital energies of the electrons. For this to work in practice, we would need wavefunctions specific to the molecular system at hand. Instead of calculating wavefunctions for all molecules in chemical space, we can approximate the *molecular* wavefunction using a linear combination of *atomic* basis functions,

$$\phi_i = \sum_\mu c_{\mu i}\chi_\mu,$$

where χ_μ are atomic basis functions and $c_{\mu i}$ are variational parameters. The atomic basis functions are optimised for individual atoms, and we can combine them to approximate the molecular wavefunction. Introducing this basis set transformation to equation 3.23 gives,

$$f_i \sum_\nu c_{\nu i}\chi_\nu = \epsilon_i \sum_\nu c_{\nu i}\chi_\nu,$$

which if we left multiply by χ_μ^* and integrate over $\mathbf{r}$ we get,

$$\sum_\nu c_{\nu i} \langle \chi_\mu^* | f_i | \chi_\nu \rangle = \epsilon_i \sum_\nu c_{\nu i} \langle \chi_\mu^* | \chi_\nu \rangle. \tag{3.24}$$

Here we can define an overlap matrix $\mathcal{S}$ and a fock matrix $\mathcal{F}$,

$$\begin{aligned} \mathcal{S}_{\mu\nu} &= \langle \chi_\mu^* | \chi_\nu \rangle, \\ \mathcal{F}_{\mu\nu} &= \langle \chi_\mu^* | f_i | \chi_\nu \rangle, \end{aligned}$$

simplifying equation 3.24 to:

$$\begin{gathered} \sum_\nu \mathcal{F}_{\mu\nu} c_{\nu i} = \epsilon_i \sum_\nu \mathcal{S}_{\mu\nu} c_{\nu i}, \\ \boldsymbol{\mathcal{F}} \mathbf{c} = \boldsymbol{\mathcal{S}} \mathbf{c} \boldsymbol{\epsilon}. \end{gathered} \tag{3.25}$$

This gives us the Hartree-Fock-Roothaan equations,[149] where an iterative process of minimising the variational parameters c, and therefore ϕ, is applied to achieve the ground state orbital energies ϵ_i, of which the sum of these energies is equal to E_{HF}.

The computational effort of HF scales with $\sim N^4$ where N is related to the number of electrons. This scaling factor is important to think about in the context of potential energy surfaces. As systems get larger and larger, calculations get more expensive, and performing thousands of them for use in PES fitting becomes challenging.

3.2 CORRELATION ENERGY

The Hartree-Fock (HF) method gives us the best single determinant wavefunction (Φ_{HF}) for a given system. As the theory is variational, the ground state HF energy will always be greater than or equal to E_0, the true ground state energy (shown in figure 3.1). The difference between E_0 and E_{HF} is known as correlation energy and it arises due to the approximations made within HF theory, namely the fact that we assume there is no correlated motion of electrons. Distinction must be made between

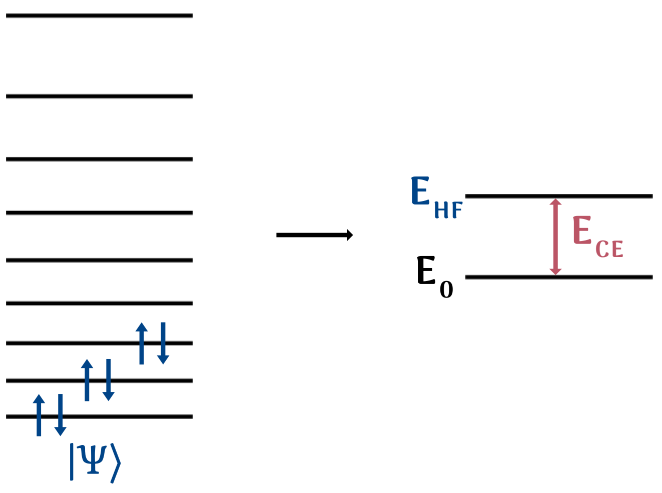


Figure 3.1. On the left hand side are optimised electron orbitals for a fictitious molecule with wavefunction $|\Psi\rangle$ calculated using Hartree-Fock (the unoccupied orbitals are virtual orbitals). The right hand side shows how the resulting energy, E_{HF} is higher in energy than the true ground state energy ,E_0, by a value known as correlation energy (E_{CE}).

static and dynamic correlation here.† Dynamic correlation describes the instantaneous motion of electrons in relation to each other, and is named due to its relation to the electron dynamics. Static correlation appears when a single determinant is not an accurate descriptor of a system, and multiple, near degenerate determinants are needed. It is called static (or non-dynamic) correlation because it has nothing to do with the motion of electrons, and most commonly appears when calculations involve bond breaking or formation, radicals, or transition metals. The systems in which static correlation is important are often called multi-reference systems.

The simplest way of recovering dynamic correlation energy is through the configuration interaction (CI) method.[150] HF theory gives rise to a set of virtual orbitals, which do not contain electrons in the ground state. Using these orbitals it is possible to construct a number of excited Slater

†It is important to note that there are no formal definitions of the two types of correlation energy and they tend to be hard to separate. However, they are useful to describe the different ways that methods aim to recover correlation effects.

determinants that combine to form the following CI wavefunction, Ψ_{CI},

$$\Psi_{CI} = \sum_{I} \kappa_I \Phi_I. \tag{3.26}$$

Here κ_I are CI expansion coefficients, and Φ_I are either Slater determinants or configuration state functions (CSFs). A configuration state function is a symmetry adapted linear combination of Slater determinants. In CI, Φ_I can be a number of excited states, such as single, Φ_i^a, or double, Φ_{ij}^{ab} excitations etc., which are visualised in figure 3.2. Here i, j are occupied orbitals, and a, b are virtual orbitals. The complete com-

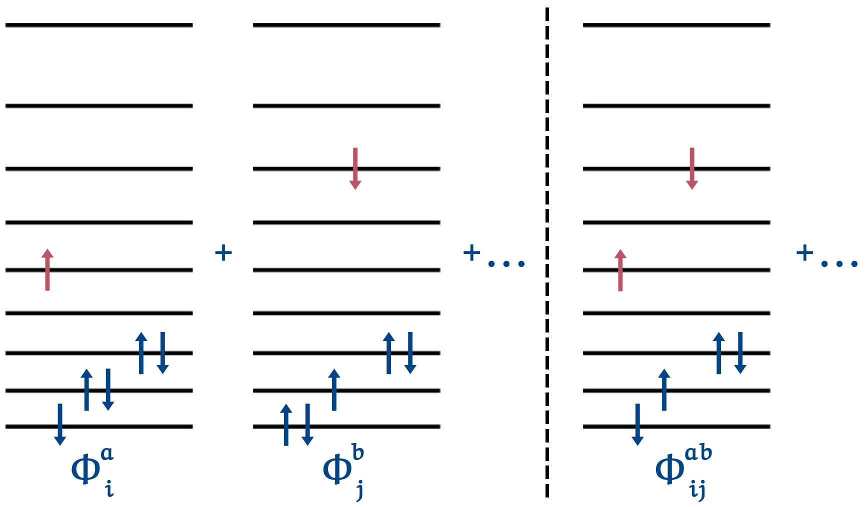


Figure 3.2. Visualisation of excited Slater determinants. Left, single excitations of electron i into orbital a and electron j into orbital b, and right, a double excitation of electrons i and j into orbitals a and b.

bination of every possible excitation is called Full CI, and would return the limit of the energy for a set of basis functions. To find the ground state energy, equation 3.26 is minimised with respect to the CI coefficients in much the same way as Hartree-Fock, using Lagrange's method of undetermined multipliers,[†]

$$\mathcal{L} = \sum_{ij} \kappa_i^* \kappa_j \langle \Phi_i | \hat{H} | \Phi_J \rangle - E \left(\sum_{ij} \kappa_i^* \kappa_j \langle \Phi_i | \Phi_j \rangle - 1 \right),$$

or utilising the notation for an overlap integral,

$$\mathcal{L} = \sum_{ij} \kappa_i^* \kappa_j \hat{H}_{ij} - E \left(\sum_{ij} \kappa_i^* \kappa_j S_{ij} - 1 \right).$$

This leads to a matrix equation of the form,

$$\mathbf{H}\boldsymbol{\kappa} = \mathbf{E}\mathbf{S}\boldsymbol{\kappa}$$

Which can be iterated through to minimise the CI energy.

CISD (including only single and double excitations) scales with N^6, while full CI scales with N factorial. Simply including just single and double excitations increases the computational cost compared to HF by a significant amount, and full CI is effectively unobtainable for large systems.

3.2.1 MCSCF

Related to CI, but focussed on recovering static correlation energy through the use of multiple determinants, is multiconfigurational self-consistent field theory (MCSCF).[151,152] The MCSCF wavefunction, Ψ_{MCSCF}, is defined in a very similar way to the CI wavefunction,

$$\Psi_{\text{MCSCF}} = \sum_{I} \kappa_I \Phi_I. \tag{3.27}$$

Here Φ_I can either be a set of chosen Slater determinants or configuration state functions. The use of CSFs makes sure that wavefunction is spin-correct, but the use of Slater determinants leads to simpler matrix elements within the MCSCF Hamiltonian and the spin can be dealt with

[†]Note that the Hartree-Fock orbitals that form the base of this are not re-optimised in the process.

later. Slater determinants are therefore the most commonly used in modern programs used to solve the MCSCF equations. In this section Φ_I will be taken to be Slater determinants, and κ_I are the CI coefficients used to weight the contributions of each determinant. The main difference to CI in MCSCF lies in the optimisation of the orbitals. The CI coefficients are calculated through the variational principle like in Hartree-Fock, but they are not optimised to minimise the energy of a single slater determinant, instead they are optimised to minimise the CI energy of the wavefunction.

The energy can be written as,

$$E = \sum_{IJ} \kappa_I^* \kappa_J H_{IJ}, \tag{3.28}$$

where H_{IJ} is not a Hamiltonian, but the matrix elements between determinants Φ_I and Φ_J. Slaters rules state that these can be written in terms of just one and two electron integrals, and so the energy can be re-written as,

$$\hat{E}_{pq} = a^\dagger_{p\alpha} a_{q\alpha} + a^\dagger_{p\beta} a_{q\beta}. \tag{3.29}$$

This is the energy expressed in terms of unitary group generators.[153] Here $a^\dagger_{p\alpha}$ is a creation operator, that represents creating an electron in orbital p with spin α. Its counterpart $a_{q\alpha}$ is then an annihilation operator, for electron q with spin α. The second term is the same, but for β spin. The full Hamiltonian in terms of equation 3.29 is,

$$\hat{H} = \sum_{pq}^{K} (p|h|q)\hat{E}_{pq} + \frac{1}{2}\sum_{pqrs}^{K} (pq|rs)(\hat{E}_{pq}\hat{E}_{rs} - \delta_{qr}\hat{E}_{ps}),$$

which leads to $H_{IJ} = \langle \Phi_I | \hat{H} | \Phi_J \rangle$ being,

$$H_{IJ} = \sum_{pq} \gamma^{IJ}_{pq} h_{pq} + \frac{1}{2}\sum_{pqrs} \Gamma^{IJ}_{pqrs}(pq|rs). \tag{3.30}$$

The coupling coefficients, γ and Γ, are coefficients in front of one and two electron integrals, and specifically take the forms,

$$\begin{aligned} \gamma^{IJ}_{pq} &= \langle \Phi_I | \hat{E}_{pq} | \Phi_J \rangle, \\ \Gamma^{IJ}_{pqrs} &= \langle \Phi_I | \hat{E}_{pq}\hat{E}_{rs} - \delta_{qr}\hat{E}_{ps} | \Phi_J \rangle. \end{aligned}$$

Substitution of equation 3.30 into equation 3.28 leads to the following form of the energy,

$$E = \sum_{IJ} \kappa_I^* \kappa_J \left[\sum_{pq} \gamma_{pq}^{IJ} h_{pq} + \frac{1}{2} \sum_{pqrs} \Gamma_{pqrs}^{IJ} (pq|rs) \right],$$

or more simply,

$$\begin{aligned} E = \sum_{pq} h_{pq}\gamma_{pq} + \frac{1}{2} \sum_{pqrs} \Gamma_{pqrs}(pq|rs), \\ \gamma_{pq} = \sum_{IJ} \kappa_I^* \kappa_J \gamma_{pq}^{IJ}, \\ \Gamma_{pqrs} = \sum_{IJ} \kappa_I^* \kappa_J \Gamma_{pqrs}^{IJ}, \end{aligned} \tag{3.31}$$

Formulating the minimisation of the energy like this effectively combines Hartree-Fock and Configuration Interaction, by not only optimising the contribution of each excited determinant (as in CI), but also the molecular spinorbitals themselves (these are optimised in HF, but kept fixed in a CI calculation).

CASSCF It would follow then that the more CSFs/Slater determinants included in the description of the wavefunction the more static correlation energy can be recovered. However, due to the nature of combinations the number of determinants can quickly become thousands or even millions, and some limitations must be imposed to make these methods computationally feasible for larger systems.

This can be achieved by splitting the orbitals into three sets (shown in figure 3.3): the core orbitals, the active orbitals, and the virtual orbitals. Complete active space self-consistent field (CASSCF) theory[154] uses this orbital separation to reduce the number of CSFs in the calculation. In a CASSCF calculation, the electrons in the core orbitals are excluded from the correlation treatment, as they are deemed unlikely to participate in meaningful interactions with other electrons (for example, the 1s electrons in sulfur are very tightly bound). All electrons in the active orbitals are subject to a full CI expansion (a complete set of excitations into all of

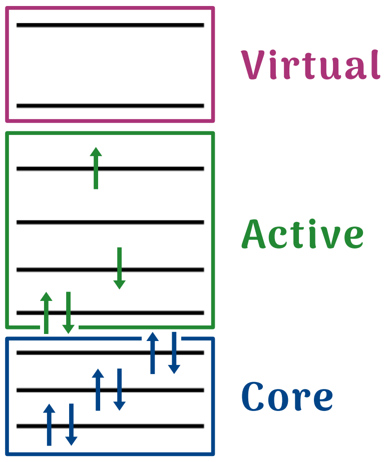


Figure 3.3. Visualisation of CASSCF orbitals showing the three spaces: core, active, and virtual. Electrons in the core orbitals are never included in the correlation treatment, while electrons in the active orbitals are subject to a complete set of excitations of all electrons into all active orbitals. The virtual orbitals outside of the active space are never occupied by any electrons in CASSCF.

the orbitals in the active space). Any virtual orbitals outside the active space are ignored in the correlation treatment as they are high in energy and never contain any electrons. This way the number of CSFs in the total wavefunction are limited, and the computational cost is reduced. The active space is chosen so that the orbitals and electrons most important in describing the multi-reference character of the system are included in the CI expansion of the wavefunction. This requires, therefore, some understanding and prior knowledge of the system and makes the method much more involved than a more 'black-box' CI calculation.

Choosing the active space Some work in recent years has made selecting the active space more automated. The Atomic Valence Active Space (AVAS) technique automatically generates an active space capable of describing all of the relevant electronic configurations that emerge from providing it with a set of target orbitals.[155] Sayfutyarova *et al.* state that, because it is known that a small set of identifiable atomic valence orbitals give rise to strong correlation effects, a set of active molecular orbitals can be determined by defining them in terms of the important atomic valence orbitals. They take a single-determinant reference function (such as one from a Hartree-Fock calculation) and use linear algebra to define simple mathematical rotations of the occupied and virtual orbitals, maximising their atomic valence character. It is then possible to automatically choose only the relevant molecular orbitals to include in the correlation treatment, removing the need to manually make this selection.

3.2.2 PERTURBATION THEORY

Perturbation theory is a way of finding an approximate solution to a problem through incremental improvements of a simpler, but related problem.[156] It can be used to approximate solutions to the Schrödinger equation, and is particularly useful in recovering the dynamic correlation of a system. The exact solution to the many-electron electronic Schrödinger equation is not known:

$$\hat{H}\Phi = E\Phi.$$

However, lets assume that the solution to a similar, but simpler, equation with Hamiltonian, $\hat{H}^{(0)}$, is known,

$$\hat{H}^{(0)}\Phi^{(0)} = E^{(0)}\Phi^{(0)}.$$

The difference then, between $\hat{H}$ and $\hat{H}^{(0)}$, is known as a perturbation. The full Hamiltonian can be expressed as a Taylor series expansion with λ representing the strength of the perturbation, while also representing the wavefunction and energy in a similar manner,

$$\begin{aligned}\hat{H} &= \hat{H}^{(0)} + \lambda\hat{H}^{(1)} + \lambda^2\hat{H}^{(2)} + \dots,\\ \Phi &= \Phi^{(0)} + \lambda\Phi^{(1)} + \lambda^2\Phi^{(2)} + \dots,\\ E &= E^{(0)} + \lambda E^{(1)} + \lambda^2 E^{(2)} + \dots\ .\end{aligned}$$

Here the superscript (1) represents a first order correction to a term, and (2) represents a second order term, etc. These can be grouped together to provide a full expression of the perturbed Schrödinger equation,

$$\begin{aligned}
&\hat{H}^{(0)}\Phi^{(0)} - E^{(0)}\Phi^{(0)} \\
&+ \lambda\{\hat{H}^{(0)}\Phi^{(1)} + \hat{H}^{(1)}\Phi^{(0)} - E^{(0)}\Phi^{(1)} - E^{(1)}\Phi^{(0)}\} \\
&+ \lambda^2\{\hat{H}^{(0)}\Phi^{(2)} + \hat{H}^{(1)}\Phi^{(1)} + \hat{H}^{(2)}\Phi^{(0)} - E^{(0)}\Phi^{(2)} - E^{(1)}\Phi^{(1)} - E^{(2)}\Phi^{(0)}\} \\
&+ \cdots = 0.
\end{aligned} \tag{3.32}$$

Note that the orders of each term combine to give the order of the perturbation that is represented by the combination of variables, e.g. $\hat{H}^{(1)}\Phi^{(1)}$ is second order. The only way to satisfy equation 3.32 for any arbitrary value of λ is for the coefficients of each power of λ to be zero. It is then possible to arrive at the following equations for the various orders of perturbation,

$$\begin{aligned}
\hat{H}^{(0)}\Phi^{(0)} &= E^{(0)}\Phi^{(0)}, \\
(\hat{H}^{(0)} - E^{(0)})\Phi^{(1)} &= (E^{(1)} - \hat{H}^{(1)})\Phi^{(0)}, \\
(\hat{H}^{(0)} - E^{(0)})\Phi^{(2)} &= (E^{(2)} - \hat{H}^{(2)})\Phi^{(0)} + (E^{(1)} - \hat{H}^{(1)})\Phi^{(1)}. \\
&\vdots
\end{aligned}$$

From the above, the expression for the first order correction to energy can be found to be,

$$E_i^{(1)} = \langle \Phi_i^{(0)} | \hat{H}^{(1)} | \Phi_i^{(0)} \rangle.$$

This is the expectation value of the first order Hamiltonian in the unperturbed system. Following a similar treatment, the expression for the second order correction to the energy can be found to be,

$$E_i^{(2)} = \hat{H}_{ii}^{(2)} + \sum_{i \neq j} \frac{\hat{H}_{ij}^{(1)}\hat{H}_{ji}^{(1)}}{E_i^{(0)} - E_j^{(0)}}, \tag{3.33}$$

where $\hat{H}_{ii}^{(2)} = \langle \Phi_i^{(0)} | \hat{H}^{(2)} | \Phi_i^{(0)} \rangle$, and $\hat{H}_{ij}^{(1)} = \langle \Phi_i^{(0)} | \hat{H}^{(1)} | \Phi_j^{(0)} \rangle$, etc.

CASPT2 CASSCF theory accounts well for static correlation effects, to improve upon this and include some of the dynamic correlation missing from the treatment, perturbation theory can be applied along side

CASSCF. Complete active space with second order perturbation theory (CASPT2)[157] takes a reference CASSCF calculation and applies a second order perturbation to the reference state, which could in principle be hundreds or even millions of CSFs.

The energy for this perturbation is given through equation 3.33,

$$E_i = E_i^{(0)} + \hat{H}_{ii}^{(2)} + \sum_{i \neq j} \frac{\hat{H}_{ij}^{(1)} \hat{H}_{ji}^{(1)}}{E_i^{(0)} - E_j^{(0)}}, \tag{3.34}$$

where $E_i^{(0)}$ is the CASSCF energy. In principal here, if the wavefunction is described in such a way that there is just a single CSF, then CASPT2 is a simple MP2 calculation. It is important to remember that CASPT2 is not variational and E_i could, in principle, go below E_0, and the method works best if the reference is close to the true solution.

3.2.3 COUPLED CLUSTER THEORY

Contrasting CASPT2, in systems where a single reference configuration is dominant, dynamic correlation can be accounted for in the 'gold standard' method of coupled cluster (CC) theory. Originally developed for nuclear physics,[158,159] it wasn't until much later that it was applied to the electron correlation problem of quantum mechanics.[160] In CC theory the ground state wavefunction Ψ_{CC} can be given by,

$$\Psi_{CC} = e^{\hat{T}} \Phi_0, \tag{3.35}$$

where Φ_0 is a single determinant reference wavefunction (most often a Hartree-Fock wavefunction) and $\hat{T}$ is a cluster operator of the form,

$$\hat{T} = \sum_i \hat{T}_i.$$

Here, $\hat{T}_i$ contains the i^{th} excited Slater determinants, generated from the HF reference,

$$\hat{T}_1\Phi_0 = \sum_{i,a} t_i^a \Phi_i^a$$
$$\hat{T}_2\Phi_0 = \sum_{\substack{i>j,\\ a>b}} t_{ij}^{ab} \Phi_{ij}^{ab}$$
$$\dots,$$

where t_i^a are expansion coefficients known as amplitudes, and i, j are occupied orbitals while a, b are virtual orbitals. Using equation 3.35, the Schrödinger equation becomes,

$$\hat{H}e^{\hat{T}}\Phi_0 = E_{CC}e^{\hat{T}}\Phi_0,$$

with the coupled cluster energy being expressed as,

$$E_{CC} = \langle\Phi_0|\hat{H}|e^{\hat{T}}\Phi_0\rangle,$$

and the exponential $e^{\hat{T}}$ can be expanded as a Taylor series to,

$$e^{\hat{T}} = 1 + \hat{T} + \frac{\hat{T}^2}{2} + \frac{\hat{T}^3}{3!} + \frac{\hat{T}^4}{4!}\cdots.$$

The major difference to CI (as CI aims to recover dynamic correlation as well), is size extensivity.[161] This means that the coupled cluster energy scales correctly, in a linear fashion, with the number of electrons.

3.3 BASIS SETS

The form of the molecular wavefunction (Φ) is clearly important, and although it is well approximated by a Slater determinant of molecular spinorbitals, ϕ,

$$\Phi(1, 2, ..., n) = |\phi_1\phi_2...\phi_n\rangle,$$

for this to work in practice ϕ_i would need to be a set of molecular spinorbitals that are specific to individual molecules (which would require molecular wavefunctions for all of chemical space). Instead the molecular spinorbitals can be further approximated through a linear combination of atomic basis functions, χ, typically optimised for individual atoms,

$$\phi_i = \sum_{\mu} c_{\mu i}\chi_\mu. \tag{3.36}$$

The atomic basis function χ is best described by a Slater type orbital of the form

$$\chi^{\text{STO}}_{abc}(x,y,z) = N x^a y^b z^c e^{-\zeta r}$$

where N is a normalisation constant, (a,b,c) control the angular momentum, and ζ - known as the exponent - controls the width of the orbital (a larger ζ leads to a tighter orbital, a smaller ζ leads to a more diffuse orbital). Figure 3.4 shows the shape of an STO, exhibiting a cusp at the nucleus and exponential decay. However, integrals over STOs are hard

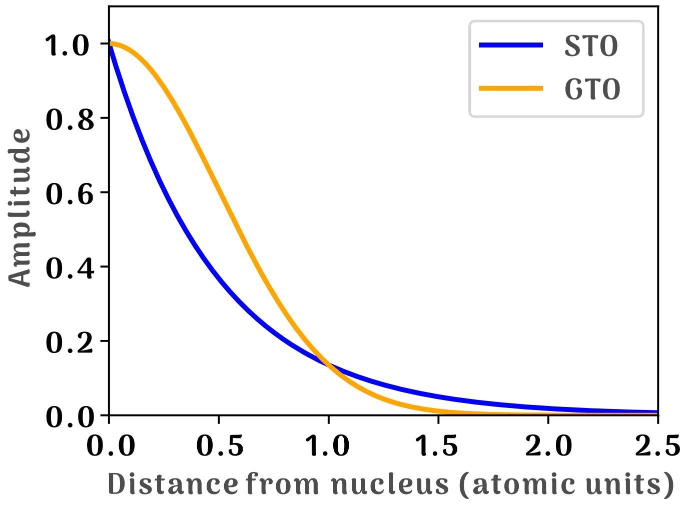


Figure 3.4. An STO plotted alongside a GTO showing the difference in orbital shape, specifically highlighting the lack of a cusp at the nucleus for the GTO.

to compute and require numerical integration to do so, leading to errors that are hard to control. Instead, we can use gaussian type orbitals, GTOs,

$$\chi^{\text{GTO}}_{abc}(x,y,z) = N x^a y^b z^c e^{-\zeta r^2}$$

where all parameters have the same meaning. These lose the cusp at the nucleus and decay too quickly (also shown in figure 3.4), but integrals over products of these functions are simpler to compute due to the Gaussian product theorem and are able to be solved analytically. It is also possible to take a linear combination of GTOs and approximate an STO while still having a computationally cheaper calculation.

$$\chi_{abc}^{CGTO}(x, y, z) = N \sum_{i} x^a y^b z^c \alpha_i e^{-\zeta_i r^2}$$

In this equation α is a contraction coefficient and controls the contribution from each GTO. A contracted gaussian type orbital (CGTO) of three GTOs would take the following form,

$$\chi_{abc}^{CGTO}(x, y, z) = N x^a y^b z^c (\alpha_1 e^{-\zeta_1 r^2} + \alpha_2 e^{-\zeta_2 r^2} + \alpha_3 e^{-\zeta_3 r^2})$$

Figure 3.5 shows how increasing the number of contracted GTOs leads to a functional form closer to the more accurate STO.

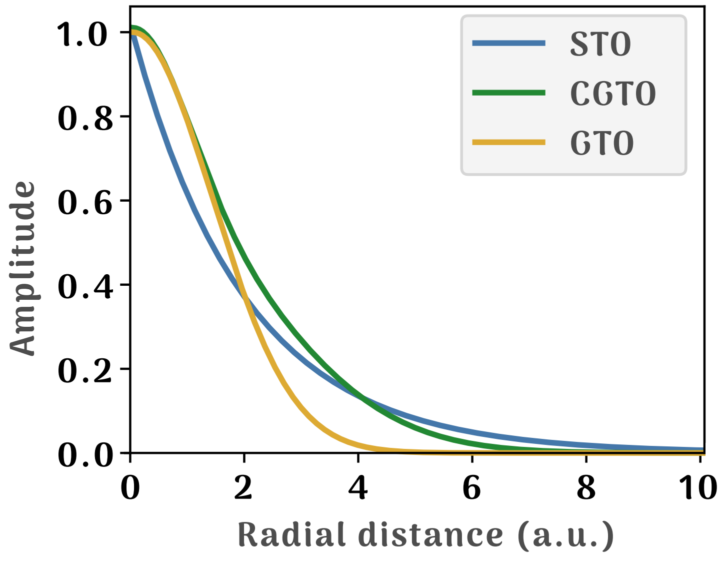


Figure 3.5. A 3 function CGTO plotted along side an STO and a single GTO, showing the improved shape of the 3-GTO.

Sets of these CGTOs are called basis sets, where the number of basis functions (χ), and values of the exponents (ζ), and contraction coefficients

(α), are typically optimised for individual atoms and obtained from a pre-optimised library of basis sets..

3.3.1 CLASSIFICATION OF BASIS SETS

At a bare minimum, an atom requires a single GTO for each atomic orbital, known as a minimal basis set. For example, carbon has two s orbitals (1s and 2s) and three p orbitals ($2p_x$, $2p_y$, and $2p_z$), and would have five basis functions of the form,

$$\begin{aligned}
\chi_{1s} &= N_{1s}e^{-\zeta r^2} \\
\chi_{2s} &= N_{2s}re^{-\zeta r^2} \\
\chi_{2p_x} &= N_{2p_x}xe^{-\zeta r^2} \\
\chi_{2p_y} &= N_{2p_y}ye^{-\zeta r^2} \\
\chi_{2p_z} &= N_{2p_z}ze^{-\zeta r^2}
\end{aligned}$$

In practice, minimal basis sets tend not to be used, and there are several extra functions that can be added to a basis set to better describe the orbitals. Recalling equation 3.36,

$$\phi_i = \sum_{\mu} c_{\mu i}\chi_{\mu},$$

atomic wavefunctions are represented as linear combinations of atomic basis functions. A simple combination of basis functions for this example would be two basis functions for each orbital, resulting in ten basis functions for carbon. This is known as a double-ζ basis set. However, each function added to the basis set adds computational cost to the calculation. As it tends to be the valence electrons that take part in the chemistry, it is usually only important for the valence orbitals to be described to high levels of accuracy. Therefore to keep computational cost to a minimum a 'split-valence' basis set can be used. In the example being used here, a double-ζ split-valence basis set for carbon would contain nine basis functions: one for the core 1s orbital, and two for each of the valence 2s and $2p_{x,y,z}$ orbitals.

In order to describe polarisation effects seen as atoms approach each other, extra basis functions can be added to the basis set that allow functions to deform by mixing with one another. An s orbital can polarise in the x direction if mixed with a p_x orbital, and p orbitals can polarise when mixed with d orbitals. In general to polarise a basis function with angular momentum l, it needs to be mixed with basis functions of angular momentum $l + 1$. In the carbon example, this adds the five d functions to the basis set.

Other important situations requiring extra functions are descriptions of anions, Rydberg states, and very electronegative atoms (such as fluorine) with a lot of electron density. In these situations a set of diffuse basis functions (those with small ζ exponents) are added to the basis sets, typically one basis function for each symmetry level. These are necessary for accurate polarizabilities or binding energies of van-der-Waals complexes (bound by dispersion), and calculations of these properties should not be done without extra diffuse functions.

3.3.2 CORRELATION CONSISTENT BASIS SETS

Correlation consistent basis sets are designed to converge towards the complete basis set (CBS) limit, and to recover a consistent amount of correlation energy with increases in basis set size. These were originally designed by Dunning[162] and have since become a gold standard in basis set design, with a large body of work resulting in correlation consistent basis sets being available for almost all of the elements. For the lighter elements, the s- and p- primitives are first optimised to converge towards a limit, with the values of ζ being calculated by minimising the energy while varying their values. Higher angular momentum correlating functions are optimised so as to recover a log-linear amount of correlation energy as basis set size increases and added to the basis sets in a modular fashion. This means they can easily be further added to in order to solve common problems, for example basis sets with extra diffuse functions have been developed to be used in anion calculations.[163] Typically these basis sets are called correlation consistent polarised valence n-zeta basis sets (cc-pVnZ), where $n =$ D, T, Q, 5, while the aug- prefix is added for the basis sets augmented with extra diffuse functions.

3.4 EFFECTIVE CORE POTENTIALS

With more advanced methods of solving the Schrödinger equation becoming the norm, and chemical systems of interest getting larger, the number of basis functions in large basis sets can become problematic. One way of combatting this problem is through the use of an effective core potential (ECP).

Separating the treatment of core and valence electrons is common throughout chemistry. The justification for this arises from the idea that only the valence electrons take part in any meaningful chemistry, while the core electrons contribute only indirectly by creating an effective potential in combination with the nucleus. Effective core potentials use these ideas in an attempt to considerably reduce the cost of a calculation by removing the explicit computation of core effects. A benefit of this is that an implicit treatment of relativistic effects can also be included for heavy atoms where relativity plays a large part in the behaviour of the core electrons, bypassing the solving of an expensive relativistic Schrödinger equation.

There are two main branches of ECPs: pseudopotentials (PPs) and model potentials (MPs). Model potentials aim to model the all electron Hartree-Fock potential for the valence electrons as accurately as possible, producing valence orbitals with the correct nodal structure. Pseudopotentials use a formal transformation of valence orbitals to pseudovalence orbitals with a simplified nodal structure, which leads to savings in the one electron basis set. Only pseudopotentials will be discussed in detail in this section as they are largely more popular within computational chemistry. The terms ECP and PP will be used interchangeably and the focus will be on the use of ECPs to remove electrons from the calculation rather than the relativistic benefits. However the interested reader is directed to an in depth review from Dolg and Cao[164] for information on how ECPs have been used to include relativistic effects, along with a description of the model potential approach.

Importantly, the use of an ECP requires a basis set that has optimised while using the ECP in question. This adds to the initial developmental effort, but in theory is offset by the computational cost recovery.

3.4.1 THE VALENCE-ONLY HAMILTONIAN

The first step in formulating ECPs is to define a valence-only Hamiltonian, where c and v refer to core and valence respectively,

$$\hat{H}_v = \sum_{i}^{n_v} \hat{h}_v(i) + \sum_{i<j}^{n_v} \hat{g}_v(i,j) + V_{cc} + \hat{V}_{CPP}$$

Here, $\hat{h}_v$ and $\hat{g}_v$ are one- and two- electron operators for n_v valence electrons, while V_{cc} is the repulsion of all nuclei and cores in the system. $\hat{V}_{CPP}$ is a core polarisation potential which will be discussed in section 3.4.4. For nonrelativistic ECPs, $\hat{h}_v$ and $\hat{g}_v$ take the forms,

$$\hat{h}_v(i) = -\frac{1}{2}\nabla_i^2 + \hat{V}_{cv}(i),$$
$$\hat{g}_v(i,j) = \frac{1}{r_{ij}},$$

where ∇_i^2 is the Laplacian, and $\hat{V}_{cv}(i)$ is a Coulomb electron-core operator. This term has to account for all of the interactions of a valence electron with the missing core electrons and the nucleus. For molecules, this term is assumed to be a superposition of N atomic contributions, where N is the number of cores,

$$\hat{V}_{cv}(i) = \sum_{\lambda}^{N} \left[-\frac{Q_\lambda}{r_{\lambda i}} + \Delta\hat{V}_{cv}^{\lambda}(i) \right],$$

and Q_λ is the core charge of the atoms. The form of the correction for orthogonality, $\Delta\hat{V}_{cv}^{\lambda}(i)$, is the focus of ECP development. The final term

V_{cc} is the Coulomb repulsion between the cores or nuclei and takes the form,

$$V_{cc} = \sum_{\lambda<\mu}^{N} \left[\frac{Q_\lambda Q_\mu}{r_{\lambda\mu}} + \Delta V_{cc}^{\lambda\mu}(r_{\lambda\mu}) \right].$$

The pairwise correction term, $\Delta V_{cc}^{\lambda\mu}(r_{\lambda\mu})$, accounts for orthogonality constraints and the Pauli-repulsion experienced by electron shells on separate cores. The valence only Hamiltonian can therefore be written fully as,

$$\hat{H}_\nu = -\frac{1}{2}\sum_{i}^{n_\nu} \hat{\nabla}_i^2 + \sum_{i<j}^{n_\nu} \frac{1}{r_{ij}} + \sum_{i}^{n_\nu}\sum_{\lambda}^{N} \left[-\frac{Q_\lambda}{r_{\lambda i}} + \Delta\hat{V}_{c\nu}^{\lambda}(i) \right] + V_{cc} + \hat{V}_{CPP} \quad (3.37)$$

and ECP development focuses on finding the form of $\Delta\hat{V}_{c\nu}$, V_{cc}, and $\hat{V}_{CPP}$. Four approximations are made in the formation of this Hamiltonian:

1. The core-valence separation is an appropriate separation to make. Where it isn't, it is accounted for by V_{CPP} where necessary.

2. Assumption that the atomic cores are inert. That is, ECPs are developed on atoms, we assume the core does not change significantly in the case of molecules.

3. The atomic one-electron Hamiltonian successfully replaces the core contributions.

4. Atomic cores do not interact with each other. For large cores this breaks down, and must be accounted for by additional parameters.

As highlighted by point four above, it is important to select an appropriate core size for the use-case in question. Large cores are attractive because they lead to largely reduced computation times, but smaller cores lead to more accurate results as more of the electrons are treated explicitly. Importantly for relativistic calculations, if the core is too small then the relativistic effects are not accounted for very well. Therefore, compromise between computational savings and accuracy should be made on a case by case basis.

3.4.2 ANALYTICAL FORM OF THE PSEUDOPOTENTIAL

From equation 3.37, the form of $\Delta\hat{V}^{\lambda}_{cv}(i)$ can be written for a given core λ, as a semi-local pseudopotential,[165,166]

$$\Delta\hat{V}^{\lambda}_{cv}(i) \approxeq \Delta\hat{V}^{\lambda}_{PP}(i) = \sum_{l=0}^{\infty} V^{\lambda}_{l}(r_{i\lambda})\hat{P}^{\lambda}_{l}(i).$$

$\hat{P}^{\lambda}_{l}(i)$, based on spherical harmonics, is a projection operator of the form,

$$\hat{P}^{\lambda}_{l}(i) = \sum_{m=-l}^{l} |lm,\lambda\rangle\langle lm,\lambda|,$$

By defining $L-1$ as the highest angular momentum in the core, it is possible to assume that $V^{\lambda}_{l}(r_{i\lambda}) = V^{\lambda}_{L}(r_{i\lambda})$ for $l \geqslant L$, and write,

$$\Delta\hat{V}^{\lambda}_{PP}(i) = V^{\lambda}_{L}(r_{i\lambda}) + \sum_{l=0}^{L-1}[V^{\lambda}_{l}(r_{i\lambda}) - V^{\lambda}_{L}(r_{i\lambda})]\hat{P}^{\lambda}_{l}(i). \tag{3.38}$$

This is semi-local under the understanding that equation 3.38 is a sum of local potentials for all angular momenta up to $L-1$, after which a common local potential acts upon all angular momenta.

3.4.3 PARAMETERISING ECPS

There are really two main ways of parameterising ECPs. The first method results in energy-consistent ECPs, where the parameters are adjusted to the all-electron valence energies of a number of many-electron states, simultaneously. The second adjusts parameters based on a specific reference state and the shape of valence orbitals in the spatial valence region of that state, along with orbital energies. This results in an ECP that is shape-consistent.

Correlation consistent ECPs In order to circumvent the poor performance in many-body theories seen with ECPs generated in an effective one-particle setting[167] Bennett *et. al.* have developed a new generation of correlation consistent effective core potentials (ccECPs) for the first and second row atoms.[52,168] These ccECPs are energy consistent and targeted specifically for use in correlated methods while retaining transferability between atoms and bonded molecules. More recently they have expanded this method to generate ccECPs for a number of transition metals.[169] They chose to use the well established form of the ECP given in equation 3.37 to ensure that these new ccECPs were able to be used in standard electronic structure packages.

3.4.4 CORE POLARISATION POTENTIALS

The separation of the core and valence electrons necessary for the application of ECPs also places limitations on the accuracy of correlated methods using basis sets employing them. This frozen-core approximation removes any correlation effects resulting from the interaction of core with valence electrons. It is necessary then, that these effects are negligible in the system of interest, but it has been shown that for alkali metal and alkaline earth compounds the core-valence correlation is almost as important as the valence correlation.[170] There have also been many studies highlighting the importance in core electron correlation in high-accuracy thermochemistry.[171–174] The physical origin of this core-valence correlation effect is principally the dynamic polarisation of the atomic cores by the valence electrons.[175]

In section 3.4.1 the $\hat{V}_{\text{CPP}}$ term in equation 3.37 was ignored, but it is this term that attempts to solve the problem of a lack of core-correlation. Briefly, this term can be defined as follows:

$$\hat{V}_{\text{CPP}} = -\frac{1}{2}\sum_{\lambda} \alpha_\lambda \mathbf{f}_\lambda^2,$$

where $\mathbf{f}_\lambda$ is the electric field generated at a core by all other cores and the valence electrons, i. Thus this equation describes the interaction between a valence electron and the core, λ, as being proportional to α_λ, the core dipole polarizability. The electric field, $\mathbf{f}_\lambda$, is given by:

$$\mathbf{f}_\lambda = \sum_{\lambda} \frac{\mathbf{r}_{\lambda i}}{r_{\lambda i}^3} g_\lambda(r_{\lambda i}) - \sum_{\mu(\neq\lambda)} \frac{Q_\mu \mathbf{r}_{\lambda\mu}}{r_{\lambda\mu}^3} g_\lambda(r_{\lambda\mu}),$$

which introduces a cutoff function, $g_\lambda(r)$, to limit the field inside the core region:

$$g_\lambda(r) = [1 - \exp\left(-\gamma_{\lambda n} r^2\right)]^n.$$

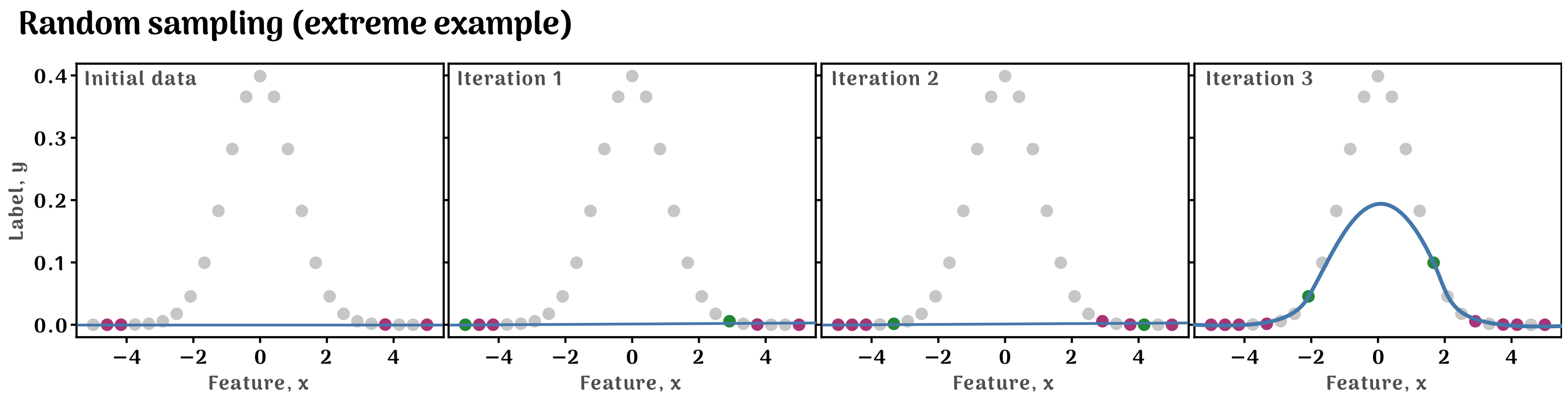
The parameter γ is fitted to suitable reference data, has two common forms, and is dependent on the functional form chosen. This first is the Fuentealba/Stoll form where $n = 1$,[176] and the second is the Müller/Meyer form where $n = 2$.[177] The development and history of the CPP approach has been reviewed by Dolg and Cao,[164] based on the pioneering work of Meyer and co- workers,[177–179] and Fuentealba and co-workers.[176]

4 | The firefly algorithm

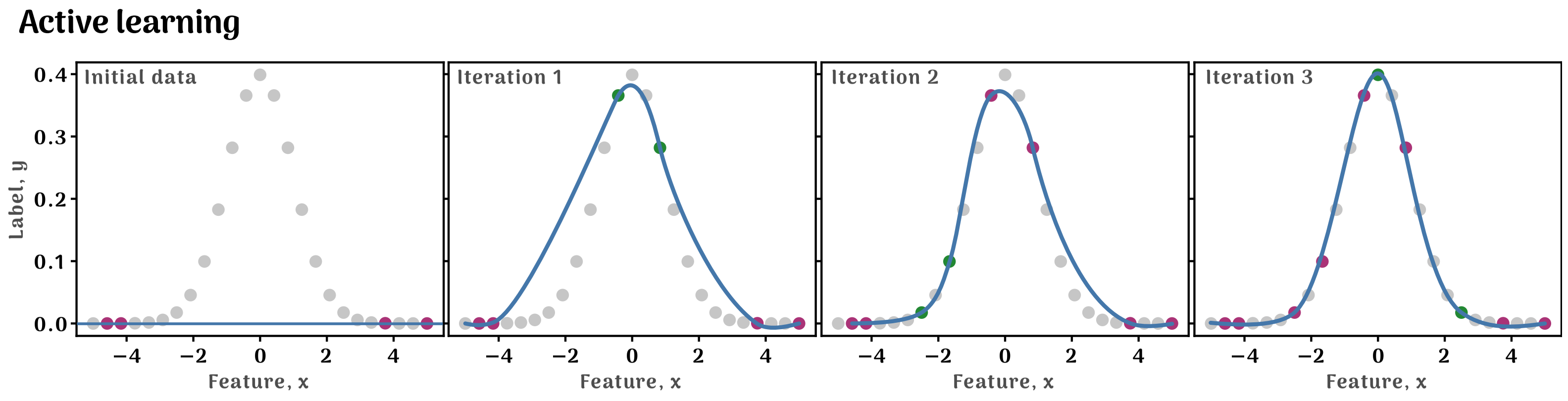
Chapter 2 highlighted the need for large datasets of high quality labelled data for supervised learning problems, particularly within the context of neural networks. Deep neural networks will often overfit if the training set is not large enough, and this becomes a problem in the application of these techniques to computational chemistry, principally due to the high computational cost of the data needed for the training set. As the desired computational accuracy has grown over the years, generating datasets of tens of thousands of *ab initio* calculations has become increasingly problematic. However, this has not just been a roadblock within the field of computational chemistry, even industry leaders in data science, who specialise in data collection, run into problems surrounding data cleaning and labelling. This process can be time consuming and costly, and has driven large volumes of research into areas such as unsupervised learning and active learning.

4.1 ACTIVE LEARNING TECHNIQUES

In the field of active learning, instead of providing a machine learning algorithm with large volumes of data the algorithm is instead presented with a smaller dataset, and is able to ask questions about the data. Specifically, it is able to ask questions about the labels of currently unlabelled data. By asking specific questions, directed by intelligent decision making, it is possible to achieve high values of generalisation, and improve the fit of an algorithm beyond just simply adding more random data to the training set. Imagining the situation where a regressor was required



(a)



(b)

Figure 4.1. An extreme example of random sampling vs. active learning. The grey points are the full dataset, the purple points are datapoints currently in the training set, and green points are new datapoints being added to the training data in that step. The blue curve/line is a best fit prediction of the points currently in the training set. In each step, new data is added to the training set and the algorithm re-trained. It can be seen that in random sampling, there is a chance for new data to be chosen in areas that are already well represented in the training set, and thus add very little information to the training. In contrast the active learning algorithm adds points to the dataset in areas that are most uncertain, capturing the nature of the full dataset much faster, and with far fewer training points.

to fit a line to a dataset (the grey points in Figure 4.1). Looking at the full dataset it is clear that some kind of polynomial curve is appropriate, but that would require knowledge of the whole configuration space, and, as highlighted by the purple points, only a small section of the dataset exists within the training set to begin with.

Instead, starting with only a few data points, the model fits a straight line. There are then two ways of adding more data to the training set. In the first example (albeit an extreme one), data is added randomly, and the extra points in the training set are very similar to the pre-existing data. As a result the algorithm does not perform any better, with the fit curve failing to capture the nature of the full data. Example two represents a fictitious active learning algorithm. In this case, the new datapoints are chosen in a way that samples an area of the configuration space not currently seen in the training data. By the third iteration the algorithm has managed to capture the nature of the curve, and continues to improve with successive iterations.

The end goal of active learning is to provide high accuracy predictions with as little data as possible. This can be achieved by adding the most important points of data to the training set, those that the algorithm is most 'uncertain' about. The various methods of active learning all formulate the definition of 'uncertainty' in their own way, and for an in depth review on a number of different types of active learning the reader is directed towards work by Settles[180] The following are a small number of examples of these types of algorithm.

Uncertainty sampling[181] This method can be used when the output of a machine learning model is a probability measure or metric. Once training is complete, a number of predictions on new datapoints can be made, and the confidence scores of each prediction can be sorted in order of most confident to least confident. In this case, the datapoint to be labelled would be the one which the algorithm has the least confidence in predicting, the point with the lowest probability score.

Query by committee[182] This method can be used if the training of the algorithm is not too computationally expensive. Here, several versions of the algorithm are trained on the same dataset, and due to the nature of randomisation within the training, each version will be slightly different. All the trained models are then asked to predict a set of values, and the value that is most disagreed upon (the one which is different between the most trained models) is the point to be labelled and added to the dataset.

The two active learning methods presented here require some way of quantifying and identifying areas of error in a trained machine learning algorithm. It could be said that locating areas of high error in a machine learning algorithm is an optimisation problem, and Chapter 2 covered a number of common optimisation algorithms. This chapter will introduce a different set of optimisation algorithms, and adapts one of them for use in the active learning of a potential energy surface.

4.2 NATURE-INSPIRED OPTIMISATION

Artificial neural networks were inspired in part by the human brain, and the function of the neurons within it. Scientists today often draw inspiration from nature, and there are a whole selection of optimisation algorithms inspired by various aspects of nature, specifically of interest to this thesis are those inspired by insect swarms.[183,184]

The optimisation algorithms in Chapter 2 are known as deterministic gradient based methods. They perform well when the search space is continuous, but fail when there are discontinuities. Methods know as stochastic algorithms have been developed to combat this, and broadly speaking fall into two similar, but distinct, categories: heuristic, and metaheuristic methods. The phrase 'heuristic' can be said to mean 'to find' or 'to discover by trial and error', and these methods work under the assumption that given enough time an optimum solution can be reached. The prefix 'meta-' can mean 'beyond', and is applied in the case of metaheuristics to imply an improvement to simple heuristic methods. Although there is no formal distinction between the two, recent trends tend to term all stochastic methods involving randomisation and locality, metaheuristics.[183]

There are two important parts to a metaheuristic algorithm: the effectiveness of the algorithm in searching the whole objective function, and the method by which the optimum solution is found (diversification and intensification, respectively). It is also useful to differentiate between two separate classes of metaheuristic algorithm; population based algorithms use several 'agents' to explore the space, while trajectory based algorithms use a single 'agent' that moves through the search space asking at each step what the 'best' move would be.

Some of the earliest examples of these algorithms date back to 1962 with the development of genetic algorithms by Holland and collaborators.[185] These are search methods that take the ideas presented in Darwinian evolution and represent them in a mathematical manner, bringing ideas of mutation, fitness, and survival of the fittest to the optimisation world. These algorithms, although some of the earliest to be developed, have formed the basis for a number of algorithms solving a number of modern problems.[186–190] Along the same lines, Rechenberg and Schwefel developed an evolutionary strategy for solving optimisation problems within aerospace engineering.[191] The work of both Holland and Rechenberg has formed the groundwork for a whole discipline called evolutionary algorithms[192–194]

The next major step in metaheuristics came with pioneering work on simulated annealing by Kirkpatrick *et al.*,[195] a trajectory based process that was inspired by the annealing process in metals. The system starts out hot and a new point is generated at a distance from the current point, proportional to a probability distribution that scales with temperature (essentially, the distance *can* be large when the temperature is high). If the new point lowers the objective function then it is accepted 100% of the time, however, if the point increases the objective function it is accepted based on a set probability. This allows the algorithm to escape local minima. Over time the temperature is lowered on a set schedule and the search space gets reduced, eventually converging to a minimum.

Dorigo introduced one of the first insect-based algorithms in his PhD thesis on optimisation and natural algorithms.[196–198] This technique was inspired by the swarming behaviour of ants resulting from pheromone communication. In a related step, Kennedy and Eberhart developed particle swarm optimisation (PSO), this time inspired by the swarming behaviour of birds, fish, and even humans.[199,200] In PSO multiple agents, or particles, are free to move throughout the search space, communicating the current and global 'best' solutions. A particle's movement is dictated by its own knowledge of the locally best solutions, and by the globally best solution, allowing the agents to locate both local and global minima. Many variants of PSO have since been developed,[201–205] and have been applied to the traveling salesman problem,[206] and task scheduling problems.[207,208]

Geem developed a harmony search algorithm that aims to mimic musicians and improvisation.[209–211] At each step in this algorithm three things can happen: a previously chosen *pitch* can be played by the musician (a previous value from memory is selected); a neighbouring pitch can be played (neighbouring value); or a random pitch is played that falls within the scope of the piece (a random value is chosen from within the scope of the problem).[211]

Returning to nature, Nakrani and Tovey aimed to optimise dynamic server allocation within internet hosting centres through the development of a honey bee algorithm.[212] Building upon these ideas, Pham *et al.* proposed the novel bee colony,[213] inspired by the natural swarming behaviour of honey bees. In the bee colony a number of bees are sent to 'scout' the objective function and they recruit nearby bees to explore the local neighbourhood. The 'best' bee is chosen for each neighbourhood, and acts as the scout for the second cycle of searches. Building on these ideas further, Karaboga developed the artificial bee colony (ABC).[214]

Reaching the focus of this chapter, Yang developed the firefly algorithm, based on the swarming behaviour of fireflies, which has been found to outperform genetic algorithms and even particle swarm optimisation in some cases.[183,215,216]

4.3 THE FIREFLY ALGORITHM

The firefly algorithm (FA) is a population based metaheuristic optimisation algorithm, inspired by the flashing and swarming behaviour of fireflies, developed by Yang.[183] Their rhythmic flashing is a process of bioluminescence used to attract mates and even prey. The rhythm, rate, and time between flashes all form the communication between sexes, bringing female fireflies towards males of the same species. Certain species of tropical fireflies even form awe-inspiring synchronised flashes within communities, exhibiting interesting biological self-organised behaviour.

The behaviour of fireflies can be simplified to develop a firefly-inspired algorithm. The following rules distil the core concepts of firefly behaviour:

1. All fireflies are attracted to all other fireflies, that is to say that sex is ignored.

2. For two fireflies of differing brightness, the less bright firefly will always move towards the brighter firefly (if there is no brighter firefly then the firefly in question will move randomly). This movement will be controlled by the attractiveness of a firefly, which will be proportional to both the brightness of said firefly and the distance between them.

3. The objective function will directly determine the brightness of a firefly.

These rules are realised as the pseudocode shown in Algorithm 1 below (which is adapted from Nature-inspired metaheuristic algorithms, Chapter 10),[183] outlining the general form of a firefly algorithm, finding the global optimum value $\mathbf{g}_*$.

Algorithm 1 The firefly algorithm (adapted from Nature-inspired metaheuristic algorithms, Chapter 10)[183]

```
1: Objective function $f(\mathbf{x})$,    $\mathbf{x} = (x_1, \dots, x_d)^T$
2: Generate initial population of fireflies $\mathbf{x}_i$ $(i = 1, 2, \dots, n)$
3: Light intensity $I_i$ at $\mathbf{x}_i$ is determined by $f(\mathbf{x}_i)$
4: Define light absorption coefficient $\gamma$
5: while (t < MaxGenerations) do
6:     for $i = 1 : n$ all $n$ fireflies do
7:         for $j = 1 : n$ all $n$ fireflies (inner loop) do
8:             if $(I_i < I_j)$ then
9:                 Move firefly $i$ towards $j$
10:            end if
11:            Vary attractiveness with distance $r$ via $\exp[-\gamma r]$
12:            Evaluate new solutions and update light intensity
13:        end for $j$
14:    end for $i$
15:    Rank the fireflies and find the current global best $\mathbf{g}_*$
16:    t++
17: end while
```

4.3.1 FORMULATING FIREFLY MOVEMENT

From Algorithm 1 there are two important parameters to define: the light intensity of each firefly; and its subsequent attractiveness to its neighbours. It is assumed that the intensity (or brightness) of each firefly is directly related to the objective function $[f(\mathbf{x})]$ being explored. Therefore, $f(\mathbf{x})$ gives I which in turn gives β, where I is the intensity and β the attractiveness. Let's assume that the light intensity $I(r)$ varies according to the inverse square law,

$$I(r) = \frac{I_s}{r^2}, \tag{4.1}$$

where I_s is the intensity at $r = 0$. By defining a light absorption coefficient, γ, the light intensity I varies with the distance r, shifted by a constant, γ,

$$I(r) = I_0 e^{-\gamma r}, \tag{4.2}$$

where I_0 is the light intensity at the position of the firefly. From equation 4.1 it is clear that there would be problems involving a singularity at

$r = 0$. To avoid this, the following Gaussian function is used as an approximation to equation 4.2,

$$I(r) = I_0 e^{-\gamma r^2}.$$

By defining attractiveness, β, as proportional to light intensity, the attractiveness of each firefly as seen by its neighbours is

$$\beta = \beta_0 e^{-\gamma r^2}, \tag{4.3}$$

where β_0 is the attractiveness at $r = 0$. Exponentials as in equation 4.3 can be slow to calculate, and a function of the form $1/(1 + r^2)$, that is faster to calculate, can be used instead to approximate the attractiveness as

$$\beta = \frac{\beta_0}{1 + \gamma r^2}. \tag{4.4}$$

If $\mathbf{x}_i$ is a vector of Cartesian coordinates for firefly i, then the Cartesian distance, r_{ij}, between two fireflies i and j is defined by

$$r_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\| = \sqrt{\sum_{k=1}^{d} (x_{i,k} - x_{j,k})^2},$$

here $x_{i,k}$ is a single component of the spatial coordinates $\mathbf{x}_i$ of firefly i. In the 3-D case this equation expands to

$$r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}.$$

By combining the above, an expression defining the movement of fireflies towards brighter (and therefore more attractive) fireflies can be written,

$$\mathbf{x}_i \leftarrow \mathbf{x}_i + \beta_0 e^{-\gamma r_{ij}^2} (\mathbf{x}_j - \mathbf{x}_i) + \alpha \epsilon_i. \tag{4.5}$$

The second term is the attraction due to brightness, controlled by the distance between the two fireflies, r_{ij}, and the light absorption coefficient, γ. The third term is a randomisation, with ϵ_i being a vector of numbers drawn from either a Gaussian distribution or a uniform distribution. The extent of randomisation is controlled by α. In general β_0, the attractiveness at $r = 0$, is set to 1 and $\alpha \in [0, 1]$.

4.3.2 USING A FIREFLY ALGORITHM AS AN ACTIVE LEARNING TECHNIQUE

In this work, the general firefly algorithm, Algorithm 1, is modified to be used as an active learning technique for neural network generation of potential energy surfaces. If the intensity is defined as the error in the prediction of the energy at the position of the firefly, the algorithm will cluster fireflies to points on the potential energy surface that are badly fit (a graphical example of this is shown in Figure 4.2). The ability

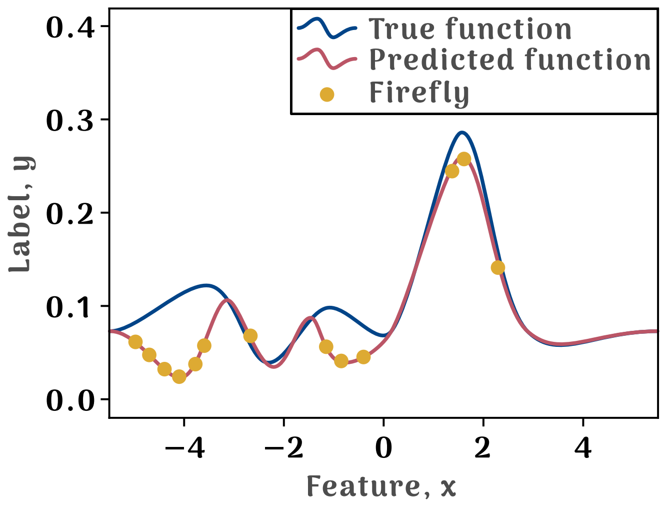


Figure 4.2. Example of firefly clustering behaviour on a machine learning predicted surface. The predicted function (red) can be seen to have a large fitting error for two major regions of the true function (blue). The fireflies (yellow) have clustered in these areas and lead to new training data being chosen in useful regions.

for the firefly algorithm to explore the global space, and hone in on a particularly badly fit section of the surface makes it a useful optimisation technique that can augment any machine learning problem. This way, a guided method of selecting data to add to the training data is applied, and by controlling the initial number of fireflies, the number of cycles of the algorithm, and the various hyperparameters within the algorithm, it is possible to maximise the accuracy of the surface while minimising the number of training points. In order to apply the firefly algorithm as an active learning technique for a machine learned potential energy surface it must be modified somewhat. Theses modifications are shown in Algorithm 2, followed by a diagram showing the general flow of the program (Figure 4.3), but the main differences are as follows:

1. Intensity, I, is defined by the difference between the neural network predicted energy and the *ab initio* calculated energy, which defines the error of the surface at a particular point.

2. The objective function is updated after a set number of generations, defined by the 'retrain counter' g. A generation is defined to be one loop of firefly intensity calculation and positional update. The value of g should be high enough that fireflies can cluster, but small enough to allow for a new network to be trained and the new fit to be evaluated. The act of retraining the neural network and defining a new objective function defines a cycle.

3. A cutoff value c is defined, so that if the error/intensity for all fireflies is less than the cutoff the algorithm stops. This minimises the number of cycles to further reduce the number of additional training points.

Figure 4.3 is a flow diagram outlining the general logic of the modified firefly algorithm. Starting with a trained neural network, the fireflies are initialised randomly on the PES and the NN is probed to acquire the predicted energy, $\tilde{E}$. The current position is sent to an electronic structure package input file (here this will be a Molpro input file) to calculate the true energy, E. Both of these energies are used to calculate the intensity $I = E - \tilde{E}$. The intensity is used to update the positions of all of the fireflies based on their attractiveness, through equation 4.5 from section 4.3.1,

$$\mathbf{x}_i \leftarrow \mathbf{x}_i + \beta_0 e^{-\gamma r_{ij}^2}(\mathbf{x}_j - \mathbf{x}_i) + \alpha \epsilon_i.$$

The updated position is again passed to Molpro to re-calculate E for this position, and I is updated. If the retrain limit has been reached, the NN is re-trained and the new objective function is used to calculate all new intensities. If at any point all values for I are less than c, the cutoff

Algorithm 2 Adapted firefly algorithm for active learning of PESs

```
 1: Objective function: $f(\mathbf{x}) = E(\mathbf{x}) - \tilde{E}(\mathbf{x}), \quad \mathbf{x} = (r, \theta, \phi)$
 2: Define a light absorption coefficient $\gamma$
 3: Define a cutoff value $c$
 4: Generate initial population of fireflies $\mathbf{x}_i$ $(i = 1, 2, \ldots, n)$
 5: Evaluate $\tilde{E}(\mathbf{x}_i)$ at each $\mathbf{x}_i$
 6: if $E(\mathbf{x}_i)$ does not exist then
 7:     Calculate $E(\mathbf{x}_i)$ using an electronic structure package
 8: end if
 9: Light intensity $I_i$ at $\mathbf{x}_i$ is determined by $E(\mathbf{x}_i) - \tilde{E}(\mathbf{x}_i)$
10: while $t <$ MaxGen, any $I_i > c$ do
11:     for $k = 1 : g$ do ▷ If $g = 2$ then $f(\mathbf{x})$ is updated after 2 generations
12:         for $i = 1 : n$ all $n$ fireflies do
13:             for $j = 1 : n$ all $n$ fireflies (inner loop) do
14:                 if $(I_i < I_j)$ then
15:                     Move firefly $i$ towards $j$ based on attractiveness $\beta$
16:                 end if
17:                 Evaluate $\tilde{E}(\mathbf{x}_i)$ at new positions
18:                 if $E(\mathbf{x}_i)$ does not exist then
19:                     Calculate $E(\mathbf{x}_i)$ using an electronic structure package
20:                 end if
21:                 Update $I_i$ for new position
22:             end for $j$
23:         end for $i$
24:         t++
25:     end for $g$
26:     Re-train the neural network and update the objective function $f(\mathbf{x})$
27:     Re-calculate $I_i$ for all fireflies using new $\tilde{E}(\mathbf{x}_i)$
28: end while
```

value, then the program stops, a NN is trained, and the final PES is presented. Otherwise, this loop continues until the maximum number of generations is reached, to avoid the algorithm running forever in the case that the PES never reaches the accuracy requested.

The fireflies will cluster in the areas of the surface that are not well fit as the error between the predicted energy and the true energy will be greatest at these points. As more data is added to the training set, and the network is retrained, these sections of the surface should improve in their fit, and the light intensities of the fireflies will drop. Eventually all fireflies should have $I < c$ and the PES will have reached the user-defined level of accuracy. In theory, this limit is reached using less data than might otherwise have been necessary by simply using a larger initial training set.

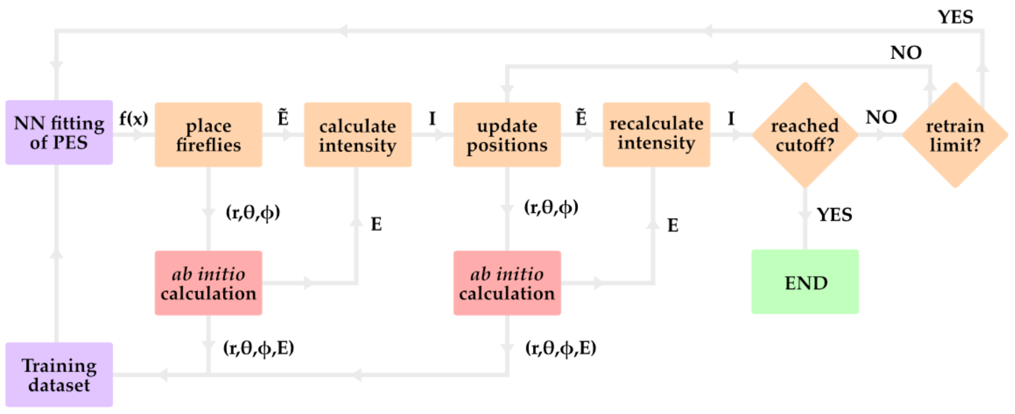


Figure 4.3. A flowchart outlining the general principle behind the application of a firefly algorithm as an active learning technique. Colours indicate the program/code that the step takes place in: purple is the neural network code, orange is the firefly algorithm, red is an *ab inito* electronic structure calculation carried out in Molpro in this case.

4.4 APPLICATION TO THE WATER POTENTIAL ENERGY SURFACE

The potential energy surface for water has been extensively studied,[217–223] and this section does not aim to improve upon the established literature. Instead the water surface is used because high level calculations on the system are fast, and this allows for iterations of the firefly algorithm for finetuning in the development process.

The general implementation of a FA shown in Algorithm 1 is modified to define the firefly brightness as a function of the predicted energy from a neural network and the true energy of an *ab initio* calculation. The full code for this algorithm can be found in Appendix A, and is available to download from GitHub (github.com/an-hill/ml-firefly-algorithm).

4.4.1 TECHNICAL DETAILS

The full dataset for this H_2O surface consists of 1331 *ab initio* single point calculations at the CCSD(T)/aug-cc-pVTZ level, calculated using the Molpro package of programs.[224,225] H_2O has three degrees of freedom: the angle, θ; and the two bond lengths, r_1 and r_2. To ensure sufficient coverage of the configuration space of the surface, the dataset consists of single point calculations of combinations of 11 points along each degree of freedom. The ranges for these 11 points are: 0.1 Å increments between 0.5 Å and 1.5 Å for the bond lengths; and 10° increments between 50° and 150° for the angle.

A deep neural network (DNN) was chosen to fit the PES for water, using internal co-ordinates as an input vector and calculated energies as the data labels. The Python library PES-Learn[135] was used to fit the DNN using the PyTorch machine learning library as a backend[136]. This library employs hyperparameter optimisation through the use of Hyperopt[226], employing both tree of parzen estimators (TPE)[227] and random search algorithms.

Before applying the firefly algorithm to the neural network three surfaces were fit to provide reference data for comparison purposes. The first was a surface fit using all 1331 points of data, to achieve a theoretical maximum performance threshold. Next a surface was fit using a randomly selected set of 300 points, leading to a less accurate surface with space for improvement. The third surface was based on the second, and used the 300 points of data already selected, while adding a second, randomly selected set of 300 points to the dataset. This resulted in 600 total points of data, chosen randomly. All training runs were initialised using the same set of parameters unless otherwise specified: the maximum number of hyperparameter search iterations was set to 20; the global minimum was forced into the training dataset (include global minimum in training dataset: True); and the trial hidden layers for the neural network were [32], [32,32], [64], [16,16,16], [64,64]. Here the number of entries in each set of square brackets indicates the number of layers, and the value of each entry indicates the number of nodes in that layer. For example, [64,64] denotes a neural network with two hidden layers, with each layer consisting of 64 nodes.

To ensure tests were fair the random seed for the hyperparameter search was set to a consistent value of 42 across all tests, and the trial hidden layer options were kept constant. To achieve the highest accuracy possible for the 'theoretical maximum performance', the network trained on all 1331 points of data used a structure based search function to select the training set. This means that datapoints are added so that the distance between them is maximised, theoretically adding a reasonable spread of data from all areas of the surface. However to assess how well the firefly algorithm performed at selectively improving a bad fit, random searching was used to select the training set for the low data surfaces to allow for the potential of sections being underrepresented.

For the firefly algorithm itself, 40 fireflies were used and allowed to move for four generations. This added 160 extra datapoints to the dataset, approximately 15% of the total training set. This meant that the dataset grew at a reasonable pace, while its minimal size was still maintained. The value of the light absorption coefficient, γ, was set to 3 as testing revealed that this allowed the fireflies to cluster in about four generations. The extent of random movement in the fireflies, α, was set to 0.2.

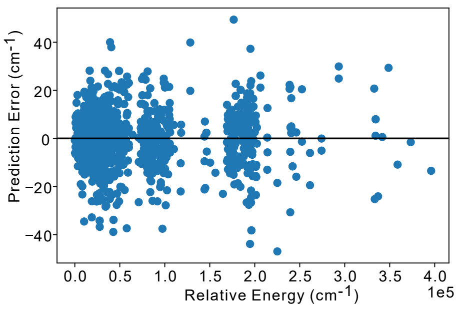
4.4.2 THEORETICAL MAXIMUM PERFORMANCE

To generate the best surface possible with the data already generated, all 1331 points were passed to PES-Learn, which chose the training/validation/testing sets in a 75 / 12.5 / 12.5 split, using structure based searching. After hyperparameter tuning the network had the following architecture:

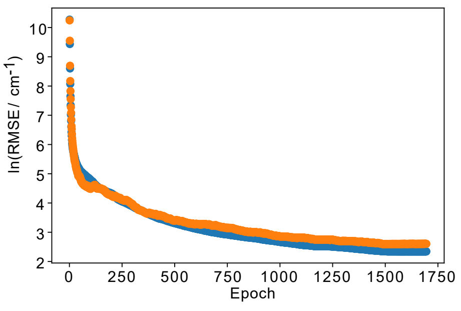
- **Layers** - [64,64],

- **Activation function** - tanh,
- **Learning rate** - 0.4.

The feature vectors ($[\theta, r_1, r_2]$) were standardised to have a mean of zero and standard deviation of one to improve training. The labels were also standardised. Training took 1695 epochs before early stopping was triggered. Figure 4.4a shows the error distribution and training/validation loss for the trained network. All errors on the dataset are



(a)



(b)

Figure 4.4. (a) Prediction errors and **(b)** the $\ln(\text{Loss})$ values of the neural network generated PES of H_2O, trained using the full 1331 point dataset. The blue line is training set loss, while the orange line is the validation set loss.

within ± 50 cm^{-1}, with a maximum absolute error of 49.39 cm^{-1}. The training/validation loss graph shows a smooth convergence towards the final RMSE values, with no evidence of overfitting. Training loss is 10.42 cm^{-1} while validation loss is 13.50 cm^{-1}. The final RMSE value on the test set is 12.69 cm^{-1}, with a RMSE across the full dataset (or a standard deviation) of 11.18 cm^{-1}. This is a reasonable accuracy, but as a comparison, a recent H_2O surface developed by Mizus *et al.* for *ab initio* spectroscopy[223] has an RMSE as low as 0.011 cm^{-1}, three orders of magnitude smaller.†

†Note, this surface was produced starting from an *ab initio* PES and refined using empirical rovibrational energy levels, thus is is unlikely that a purely computational surface will reach these levels of accuracy. However, it poses as a good example of the desired accuracy for spectroscopic calculations.

It is clear from Figure 4.4a that the dataset contains significantly less data at higher energies ($> 2 \times 10^5$ cm^{-1}, see horizontal axis) and as a result the average error for these high energy points is larger. Generally however the errors are equally distributed across all energies, suggesting that the surface is well fit across the whole space. Figure 4.5a shows a contour plot of a slice of the predicted PES for water. The plot shows a cut of the PES scanning the angle and keeping each bond the same length ($r_1 = r_2$). A plot of the same surface with slightly smaller bounds is also shown in Figure 4.5b, to capture the nature of the minimum a little better (as the full surface colour spread is dominated by the high energy, short bond length, structures). The predicted values for the global minimum

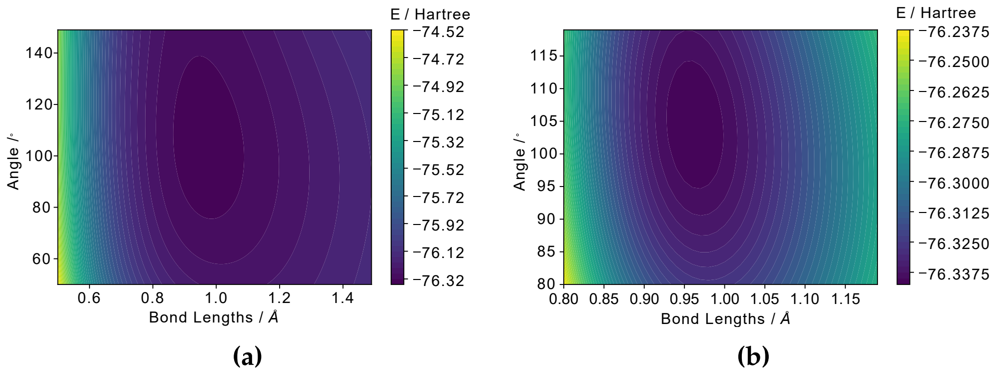


Figure 4.5. A predicted potential energy surface for H_2O trained on the full dataset training set of 998 datapoints, showing **(a)** a contour plot of the surface spanning the coordinates of the full dataset, and **(b)** a limited data range contour plot. The colour scale for the full data range is dominated by the high energy structures (where $\theta = 50°$ and $r_i = 0.5$ Å, $i = 1, 2$), so these have been removed in **(b)** to reveal the low energy contour.

ground state geometry and energy are shown in Table 4.1, along side the optimised CCSD(T)/aug-cc-pVTZ values. Prediction of the equilibrium geometry by the NN fit is reasonable, with bond lengths being under-predicted by 0.0025 Å and 0.0012 Å and the angle being over-predicted by 0.28°. However, as can be seen, the network predicts that the optimum geometry does not have equal bond lengths as calculated by CCSD(T), which is also chemically known to be the case. One downside to a

Table 4.1. Predicted ground state equilibrium geometry and energy of water resulting from training with 1331 points of data, along side the optimised CCSD(T)/aug-cc-pVTZ values

1331 points	Prediction from PES	CCSD(T) optimised geometry
R_1 / Å	0.9591	0.9616
R_2 / Å	0.9604	0.9616
θ / °	104.46	104.18
E / a.u.	−76.342331	−76.342326

machine learned surface is, that unless the neural network is trained with symmetry information, there will be no such considerations when determining the minima on the surface. The energy prediction of the global ground state is very good however, with an error of 0.000005 Hartree (1.10 cm^{-1}).

4.4.3 TRAINING ON A SMALLER DATASET

300 point surface. From the full 1331 point dataset, 300 points were randomly sampled and used as a starting point for a train-test split. The 300 point dataset was randomly split into training, validation, and testing sets in a 75 : 12.5 : 12.5 ratio. After hyperparameter optimisation the architecture of the network was as follows:

- **Layers** - [16,16,16],
- **Activation function** - tanh,
- **Learning rate** - 0.8.

The labels and features were scaled again using standardisation and training took 394 epochs before early stopping was triggered. Figure 4.6 shows the prediction errors and loss graph for the resulting 300 point surface. The maximum error for this surface is, as expected, much higher, reaching 1524.13 cm^{-1} for one high energy structure with a relative energy of around 2×10^5 cm^{-1}. However, excluding this, the errors all fall

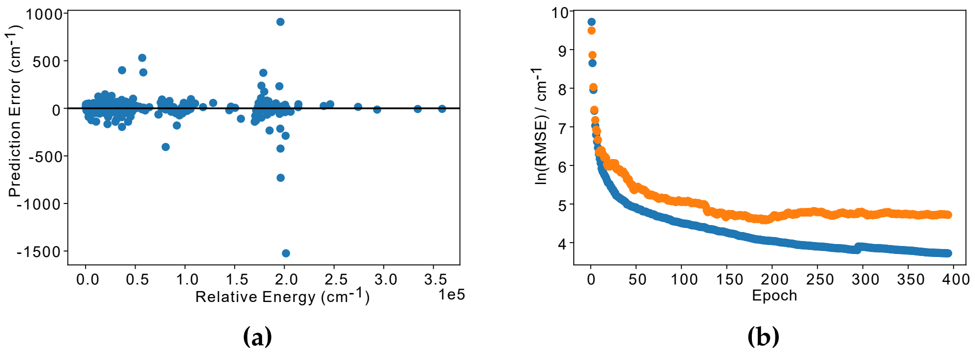


Figure 4.6. **(a)** Prediction errors and **(b)** the $\ln(\text{Loss})$ values of the neural network generated PES of H_2O, trained using a 300 point dataset. The blue line is training set loss, while the orange line is the validation set loss.

within ~500 cm^{-1}. Similarly to the full 1331 surface, the majority of the datapoints fall within the 2×10^5 cm^{-1} range, however Figure 5a suggests that this network predicts the energies of extremely high energy structures very well (although there are not very many of them), and most of the error comes from the mid-energy structures.

Figure 4.6b shows that the training loss and validation loss appear to be diverging as the training came to an end, suggesting that the surface may be over fit to the training data. This is reinforced by the particularly large RMSE on the test set, 363.70 cm^{-1}, compared to the training set, 41.40 cm^{-1}. The test set RMSE is also significantly larger than the validation set RMSE (111.68 cm^{-1}) which suggests that the training data has holes in specific areas of the configuration space, which the model has not learnt how to deal with, and the validation and test sets contain points in these areas. This explains why the very high energy structures were very well predicted. Looking at the test set, the maximum energy found in the dataset is −75.4217027 Hartree, the training set however has structures with energies of −74.7062416 Hartree. As the network appears to have been overfit, it is safe to assume that these high energy structures are only predicted well due to overfitting and their lack of appearance in the validation and testing datasets.

Table 4.2 shows that the ground state geometry is predicted similarly to the 1331 point surface (errors of 0.0007 Å, 0.0005 Å, 0.35°), however the error in energy prediction is 0.00011 Hartree, an factor of twenty larger than the full 1331 surface.

Table 4.2. Predicted ground state equilibrium geometry and energy of water resulting from training with 300 points of data, along side the optimised CCSD(T)/aug-cc-pVTZ values

300 points	Prediction from PES	CCSD(T) optimised geometry
R_1 / Å	0.9623	0.9616
R_2 / Å	0.9611	0.9616
θ / °	103.83	104.18
E / a.u.	−76.342216	−76.342326

600 point surface, randomly selected. Starting with the dataset from the 300 point surface, 300 additional points were randomly selected from the remaining pool of training data. These were added to the total dataset for a total of 600 datapoints, and randomly split into training/validation/testing sets at the same ratio as previous runs. After hyperparameter optimisation the neural network architecture for this surface was:

- **Layers** - [32],
- **Activation function** - tanh,
- **Learning rate** - 0.5,

with features and labels being standardised as before. Training concluded after 564 epochs.

The error and loss graphs immediately show improvements over the 300 point surface (Figure 4.7b). The absolute maximum error has been reduced to 462.98 cm^{-1} for a single high energy point ($> 3.5\times10^5$ cm^{-1}), and if ignored then the prediction errors on the dataset are all less than 400 cm^{-1}. Comparing the loss values between datasets shows significant im-

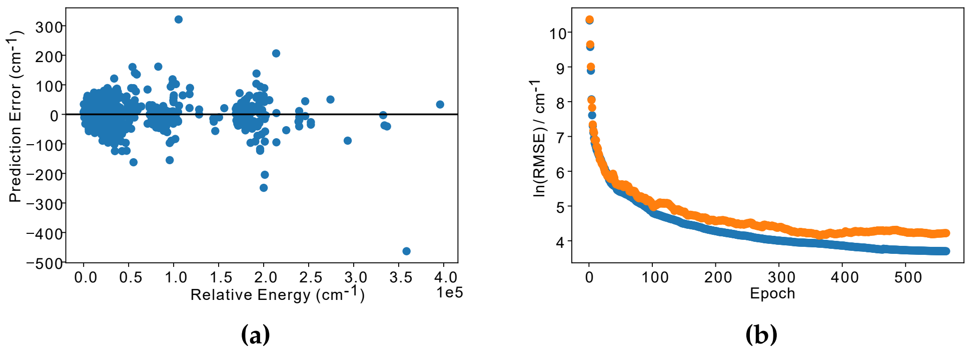


Figure 4.7. **(a)** Prediction errors and **(b)** the $\ln(\text{Loss})$ values of the neural network generated PES of H_2O, trained using a 600 point dataset, randomly improved from the 300 point dataset. The blue line is training set loss, while the orange line is the validation set loss.

provements over the 300 point surface, with the test set RMSE dropping to 80.19 cm^{-1}. However, the training set RMSE shows little to no improvement at 40.49 cm^{-1}. This suggests that the extra data predominantly helped combat the overfitting, and improved the train/validation/test split so that more of the structure space was covered in the training set. There is possibly still some overfitting present as the validation set RMSE is slightly higher at 68.39 cm^{-1}, but errors in the high energy region indicate that the particularly high energy structures are no longer overfit.

The global minimum energy and geometry predictions for this surface are shown in Table 4.3. This is a very similar performance to the 300 point surface, with slight improvement to the energy prediction (the error is down to 0.000063 Hartree, a factor $\sim \frac{1}{2}$) and minor geometry changes. This is expected however, as the accuracy of the low level regions of the surface, where the equilibrium geometry is found, did not improve with the additional datapoints.

Randomly selecting new data to improve the surface seems to have worked, so next the firefly algorithm was employed to assess what effect a more direct approach to new data selection would have on the accuracy

Table 4.3. Predicted ground state equilibrium geometry and energy of water resulting from training with 600 points of data, along side the optimised CCSD(T)/aug-cc-pVTZ values

600 points	Prediction from PES	CCSD(T) optimised geometry
R_1 / Å	0.9623	0.9616
R_2 / Å	0.9615	0.9616
θ / °	103.82	104.18
E / a.u.	−76.342263	−76.342326

of the machine learned surface.

4.4.4 APPLICATION OF THE FIREFLY ALGORITHM

By default the fireflies only have one set of geometric limits to keep them within the bounds of the trained surface, and they are free to move anywhere on said surface. For a surface with clearly identifiable minima/maxima this works and several clusters of fireflies are seen.[183,215,216] However, in the case of the water PES, there is only one clearly identifiable 'maximum' but there area several areas in the surface that could be better fit with more training data. Therefore the algorithm can be improved by splitting the fireflies into several groups. This is done by giving the fireflies minimum and maximum geometric limits that they are restricted to. By creating multiple groups with different limits, several distinct regions of the surface can be explored independently. This has the effect of forcing the fireflies to find several optima, and helps improve the spread of data added to the training set.

460 point surface, selected by the firefly algorithm. Starting with the trained 300 point surface, 40 fireflies were initialised into four groups, with four sets of geometric limits. The limits for each group are shown in Table 4.4. Ten fireflies were placed randomly within the limits for each group and their brightness calculated through $I = E_{\text{true}} - \tilde{E}$, where E_{true} is the calculated energy from a CCSD(T)/aug-cc-pVTZ calculation and $\tilde{E}$ is

Table 4.4. Geometric limits for each group of fireflies ($\theta/^\circ$, $r_1/\text{Å}$, $r_2/\text{Å}$)

	Minimum limits			Maximum limits		
Group	θ	r_1	r_2	θ	r_1	r_2
1	50	0.50	0.50	100	1.00	1.00
2	101	1.01	1.01	150	1.50	1.50
3	50	1.01	0.50	100	1.50	1.00
4	101	0.50	1.01	150	1.00	1.50

the energy predicted by the current PES. Every time a value of E_{true} was calculated, it was added to the 300 point dataset along with the current geometry. The fireflies were allowed to move according to their relative brightness and equation 4.5 for four iterations, at which point the neural network was re-trained with the new 460 point dataset.

Hyperparameters after optimisation were:

- **Layers** - [64,64],
- **Activation function** - tanh,
- **Learning rate** - 0.5,

and training concluded after 839 epochs. After only 160 extra datapoints the maximum absolute error is down to 280.32 cm^{-1}, less than the 462.98 cm^{-1} of the random 600 point surface. Most errors are less than 150 cm^{-1}, showing improvement over the 600 point surface (most errors less than 200 cm^{-1}). However, looking at the loss graph in figure 4.8 indicates overfitting on the training set with the divergence of the training and validation losses. The RMSE values also reveal this is likely to be true with the validation and testing RMSE values being approximately twice that of the training set loss (train = 37.87, test = 87.10, validation = 76.55 [all in cm^{-1}]). Generally however, the performance seems impressive, and the PES has reached the same levels of accuracy with just over half the extra datapoints as the 600 point surface.

The prediction of the global minimum is much improved from the 300 and 600 point surfaces (shown in Table 4.5), notably, the predicted

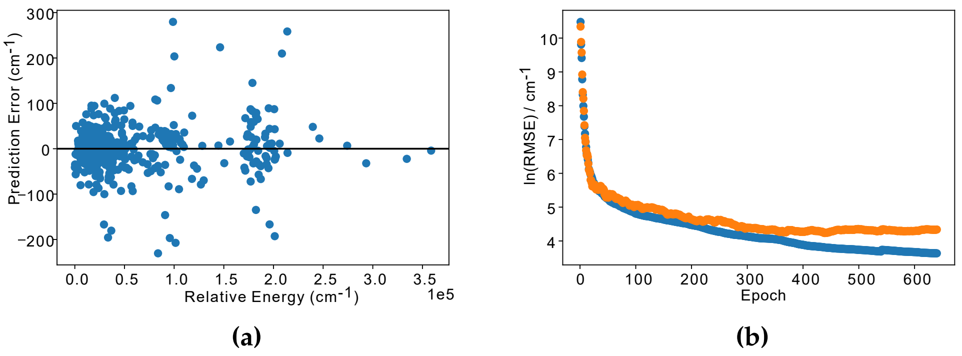


Figure 4.8. **(a)** Prediction errors and **(b)** the ln(Loss) values of the neural network generated PES of H_2O, trained using a 460 point dataset, improved from the 300 point dataset using the firefly algorithm. The blue line is training set loss, while the orange line is the validation set loss.

ground state equilibrium geometry for this surface has almost identical bond lengths (0.9626 Å and 0.9627 Å) which is predicted by CCSD(T), but missing from the previous surfaces. The ground state energy is also well predicted with an error of 0.000038 Hartree.

Table 4.5. Predicted ground state equilibrium geometry and energy of water resulting from training with 460 points of data (160 chosen by the firefly algorithm), along side the optimised CCSD(T)/aug-cc-pVTZ values

460 points	Prediction from PES	CCSD(T) optimised geometry
R_1 / Å	0.9626	0.9616
R_2 / Å	0.9627	0.9616
θ / °	104.07	104.18
E / a.u.	−76.342362	−76.342326

620 point surface, further selected by the firefly algorithm. In order to assess the performance that could be achieved with a similar number of points as the 600 point surface, another four firefly cycles were run using the 460 point dataset. The fireflies started from the same points they finished at in the 460 point run and were allowed to move for four more iterations, after which the network was re-trained. After hyperparameter tuning and training for 1323 epochs, the neural network architecture was as follows:

- **Layers** - [32],
- **Activation function** - tanh,
- **Learning rate** - 0.5,

and the errors and RMSE graphs are plotted in Figure 4.9. The RMSE

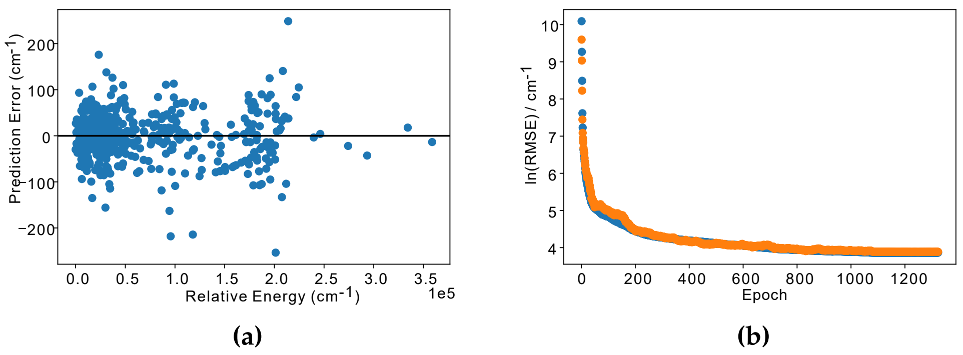


Figure 4.9. **(a)** Prediction errors and **(b)** the $\ln(\text{Loss})$ values of the neural network generated PES of H_2O, trained using a 620 point dataset, further improved from the 460 point dataset, generated by the firefly algorithm. The blue line is training set loss, while the orange line is the validation set loss.

on the training set has gone up slightly (to 47.73 cm^{-1}), however the validation and testing set RMSE values are 48.72 cm^{-1} and 46.01 cm^{-1} respectively, approximately half those in the 460 and 600 point runs. The similar values between the three datasets also suggests a lack of overfitting (also indicated by the shape of the graph). The maximum absolute error is 248.71 cm^{-1}, improved from the 460 point firefly surface a little, and about 70% of the maximum absolute error of the 600 point surface.

The equilibrium geometry prediction (Table 4.6) is slightly worse with the bond lengths no longer being equal, and the angle prediction being smaller. The energy prediction is equally impacted, having an error of 0.000053 Hartree (vs the 0.000032 Hartree of the previous surface). The implication here is that the network has sacrificed some of the accuracy at the lower energies to improve the overall fit of the surface. This is not unexpected as the minimum energy structure is forced into the training set in the train-test split that PES Learn employs. Therefore when the surface is less overfit to the training data, it would be expected that its prediction on points within that set are worse.

Table 4.6. Predicted ground state equilibrium geometry and energy of water resulting from training with 620 points of data (320 chosen by the firefly algorithm), along side the optimised CCSD(T)/aug-cc-pVTZ values

620 points	Prediction from PES	CCSD(T) optimised geometry
R_1 / Å	0.9610	0.9616
R_2 / Å	0.9629	0.9616
θ / °	103.75	104.18
E / a.u.	−76.342379	−76.342326

The clustering of the fireflies can be seen in Figure 4.10. It is clear that by four iterations (Figures 4.10a - 4.10d) of the algorithm (the first pass, and 460 datapoints), the fireflies have managed to cluster in the area of highest error for each of the four limits imposed on the algorithm. It is also clear that the largest source of error in this model is the region of the surface surrounding very short bond lengths. It is interesting that even though the second pass of the algorithm (Figures 4.10e - 4.10h) does not add much more in the way of varied data, the extra points in these high energy regions have reduced the overfitting and general performance of the surface.

Resetting firefly position. As the fireflies had clustered by four iterations any further points added to the dataset in the 620 point firefly algorithm surface had very similar geometries and did not add much in

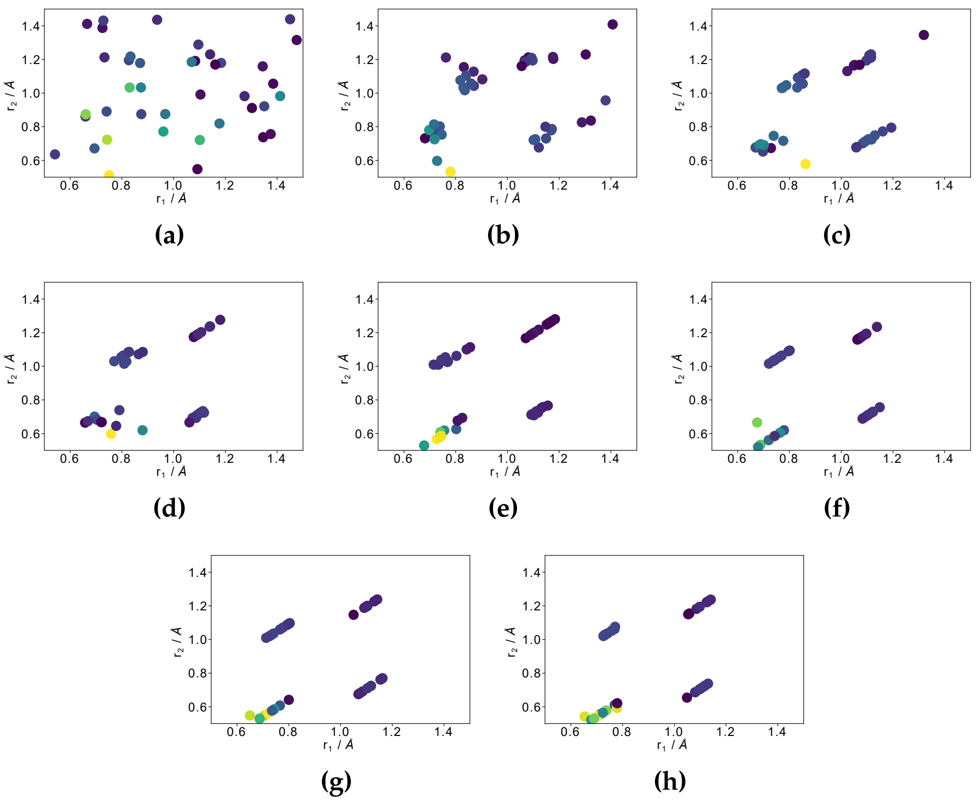


Figure 4.10. (a)-(h) Generations 0 - 7 of the fireflies showing the positional updates of r_1 and r_2 throughout the training. The brighter the firefly, the larger the prediction error. Generation 0 is the initial random placement of the fireflies. The clustering behaviour can be seen, showing four distinct clusters within the four defined regions.

the way of varied information to the training set. It would be expected that a greater improvement in the surface would be seen if the fireflies were able to reset their positions, and then re-optimise to new maxima appearing after the objective function has been updated. Therefore another firefly cycle was run, but with the positions of the fireflies re-set before starting. After training for 795 epochs the hyperparameters for this model were:

- **Layers** - [32,32],
- **Activation function** - tanh,

- **Learning rate** - 0.8.

Figure 4.11 shows that the fireflies have clustered in new regions of the surface, with the previous high error region surrounding 0.5 Å bond lengths being ignored entirely. The performance of the network is also improved (Figure 4.12). The RMSE on the training set is 11.06 cm^{-1},

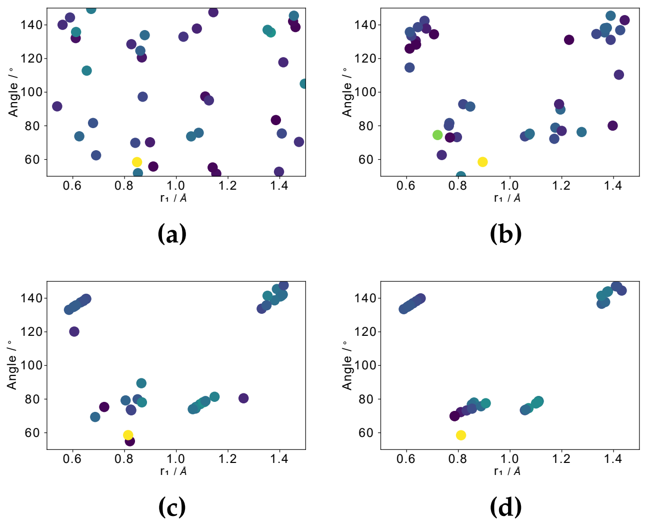
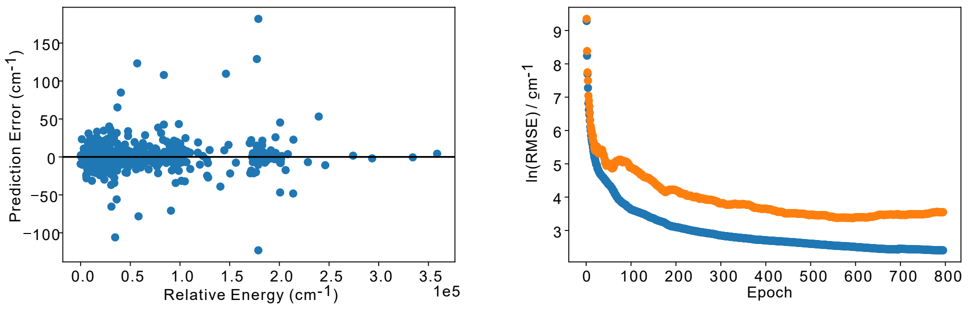


Figure 4.11. **(a)-(d)** Generations 0 - 3 of the fireflies after having their positions reset, showing the positional updates of r_1 and θ throughout the training.

almost reaching as low as the training RMSE of the full 1331 dataset (10.42 cm^{-1}) and the validation and test set RMSEs are the lowest of all the trained networks (34.77 cm^{-1} and 34.01 cm^{-1}, respectively). However, the maximum error is still approximately three times that of the full 1331 dataset model, being 181.89 cm^{-1} compared to 49.39 cm^{-1}.[†] The full dataset RMSE is 19.69 cm^{-1} suggesting that generally the model performs almost as well as the full 1331 dataset.

The predicted ground state equilibrium geometry and energy are reported in Table 4.7. The bond lengths are symmetric, with errors

[†]The structure of this point is quite strained with some very short bond lengths ($\theta = 150.0$ $r_1 = 0.8$ $r_2 = 0.5$), it also resides in a region of the surface that the fireflies had clustered around, so it is unsurprising that the prediction error is high.



(a) **(b)**

Figure 4.12. **(a)** Prediction errors and **(b)** the $\ln(\text{Loss})$ values of the neural network generated PES of H_2O, trained using a 620 point dataset, further improved from the 460 point dataset, generated by the firefly algorithm after resetting the firefly positions and updating the objective function. The blue line is training set loss, while the orange line is the validation set loss.

of 0.0011 Å, and a bond angle error of 0.46°, when compared to the calculated CCSD(T)/aug-cc-pVTZ geometry. The error in the energy prediction is 20% that of the 620 point surface, being 0.00001 Hartree. It is clear that by re-training the network (thus regenerating the objective function), and resetting the firefly positions, the firefly algorithm has performed better than simply running for eight generations, achieving almost the same level of accuracy as the 1331 surface, with half as many datapoints in the training set.

4.5 CONCLUSIONS

The firefly algorithm has been introduced as an optimisation algorithm, and has been adapted to work as an active learning technique for use in machine learning. A reference potential energy surface for water has been fit using a neural network trained on a dataset of 1331 different geometries of water and their associated CCSD(T)/aug-cc-pVTZ energies. To determine the efficacy of the the firefly algorithm a low data surface was trained on a random sample of 300 datapoints from the full dataset, and the firefly algorithm used to incrementally select new data to add to

Table 4.7. Predicted ground state equilibrium geometry and energy of water resulting from training with 620 points of data (320 chosen by the firefly algorithm), after having reset the firefly position after the first 160 points, along side the optimised CCSD(T)/aug-cc-pVTZ values

620 points	Prediction from PES	CCSD(T) optimised geometry
R_1 / Å	0.9605	0.9616
R_2 / Å	0.9605	0.9616
θ / °	103.72	104.18
E / a.u.	−76.342316	−76.342326

the dataset.

Table 4.8 shows a summary of all the trained networks in this chapter. There is a clear improvement in training/validation/testing loss values when using the firefly algorithm over a random data selection method. For example, the test set RMSE for the randomly selected 600 point surface is 80.18 cm^{-1}, whereas if data was selected by the firefly algorithm, with 620 points in the dataset the test set RMSE is 34.01 cm^{-1}, just a factor of three larger than the full 1331 surface. For this system, it has been shown that resetting the firefly positions after four generations helps the fireflies identify new areas on the objective function that could be improved, and results in an improved model (with full dataset RMSE values of 47.65 cm^{-1} for the first version of the algorithm, compared 19.69 cm^{-1} after having reset the firefly positions after the first cycle). This performance increase is seen due to the successful clustering behaviour of the fireflies seen in Section 4.4.4.

Table 4.8. Comparison of the different training runs, showing number of points trained on, the architecture after the hyperparameter optimisation, the final RMSE values in cm^{-1} for the training, validation, and testing sets, and the error in predicted ground state equilibrium geometry (r Å, θ°) and energies (Hartree) verses the optimised CCSD(T)/aug-cc-pVTZ geometry.

Dataset	1331 (Full)	300 (Rand)	600 (Rand)	460 (FA)	620 (FA)	620 (Reset)
Train dataset	998	225	450	345	465	465
Val/test datasets	166/166	38/37	75/75	58/57	78/77	78/77
Layers	[64,64]	[16,16,16]	[32]	[64,64]	[32]	[32,32]
Activation	tanh	tanh	tanh	tanh	tanh	tanh
Learning rate	0.4	0.8	0.5	0.5	0.5	0.8
Max |error|	49.39	3253.67	462.98	280.32	248.71	181.89
Train RMSE	10.42	41.40	40.49	37.87	47.73	11.06
Val. RMSE	13.50	111.68	68.39	76.55	48.72	34.77
Test RMSE	12.69	363.70	80.19	87.10	46.01	34.01
Full RMSE	11.16	138.48	51.08	52.48	47.65	19.69
Δr_1	0.0025	0.0007	0.0007	0.0010	0.0006	0.0011
Δr_2	0.0012	0.0005	0.0001	0.0011	0.0013	0.0011
$\Delta\theta$	0.28	0.35	0.36	0.11	0.43	0.46
ΔE	0.000005	0.000110	0.000063	0.000038	0.000053	0.000010

5 | Basis set development

Work from this chapter has been published as *Correlation Consistent Basis Sets and Core Polarization Potentials for Al–Ar with ccECP Pseudopotentials*, J. Phys. Chem. A 2022, 126, 34, 5853-5863.[53]

In Chapter 1 three methods of speeding up PES generation were highlighted: use a machine learning algorithm to reduce fitting complexity; use an active learning technique to reduce the number of datapoints needed in the training set; and speed up the *ab initio* data generation in some way. Chapter 4 outlined a new method of reducing the number of *ab initio* datapoints themselves through the use of a firefly algorithm, and Chapter 3 introduced effective core potentials as a way of removing core electrons from the calculation, inherently reducing calculation time. Bennett *et al.* have developed correlation consistent ECPs (ccECPs) for first and second row atoms,[52,168] and more recently for a selection of heavy elements (I, Te, Bi, Ag, Au, Pd, Ir, Mo, and W).[169] These ccECPs are designed to be specifically paired with the correlation consistent style of basis sets, to ensure smooth convergence towards the complete basis set limit. As such, new correlation consistent basis sets for the second row atoms (Al–Ar) to be used with the neon-core correlation consistent effective core potentials (ccECPs) have been developed in this work. The basis sets, denoted cc-pV(n+d)Z-ccECP (n = D, T, Q), include the "tight" d functions that are known to be important for second row elements. Sets augmented with additional diffuse functions are also reported. Also highlighted in Chapter 3 was the importance of core-valence correlation in certain systems. However, it can be prohibitively expensive to include, and usually requires the use of specifically optimised basis sets, such as core-valence basis sets (cc-pCVnZ), to properly account for these effects. An alternative method of including the effects of this correlation was highlighted in Chapter 3, called core polarisation potentials (CPPs). Parameters for these potentials have been adjusted for the same elements as the basis sets and two different forms of the CPP cut-off function have been analysed.

The accuracy of both the basis sets and the CPPs is assessed through benchmark calculations at the coupled-cluster level of theory for atomic and molecular properties. Agreement with all-electron results is much improved relative to the basis sets that originally accompanied the ccECPs, moreover, the combination of cc-pV(n+d)Z-ccECP and CPPs is found to be a computationally efficient and accurate alternative to including core electrons in the correlation treatment.

5.1 INTRODUCTION

The use of *ab initio* quantum chemistry methods to investigate the properties, thermochemistry and reactivity of molecules relies on the expansion of the wave function in products of one-electron orbitals, which are typically expressed in the basis of a linear combination of Gaussian-type functions. The choice of this basis set dictates both the accuracy and computational efficiency of quantum chemical calculations and has been the subject of a number of reviews.[228–230] The correlation consistent (cc) basis sets were originally developed by Dunning to systematically approach the complete basis set (CBS) limit.[162] A large body of work over the last three decades has resulted in cc basis sets available for almost all of the elements in the periodic table, with consistency in the exponents being energy optimised and using a general contraction scheme. They are typically denoted cc-pVnZ (correlation consistent polarized valence n-zeta) basis sets, where n = D, T, Q, 5,... and are designed in a modular fashion. This allows for the addition of functions to address common problems. For example, augmenting with diffuse functions (denoted aug-cc-pVnZ) gives a better description of anions, produces significantly better results for electron affinities of atoms, and is important in calculating molecular properties such as polarizabilities and intermolecular interactions.[163]

Correlation consistent basis sets for the second row elements Al–Ar were originally published in 1993.[231] However, a later investigation by Bauschlicher and Partridge reported that the aug-cc-pVnZ basis sets produced unacceptably large errors for the atomization energy of SO_2.[232] Careful evaluation of the performance of these basis sets showed that the addition of large exponent (tight) *d* functions led to major improvements in these benchmarks. Further analysis by Martin revealed that tight *d* functions can also have a large effect on the Hartree-Fock energies of molecules containing second-row elements in a high oxidation state, with the involvement of 3*d* functions in the bonding orbitals and a suggested term of "inner polarization functions" for basis functions of this type.[233,234] As a result of the highlighted basis set deficiencies, a new generation of cc-pV(n+d)Z basis sets were developed by Dunning *et al.*[235] It has been recommended that for second row p-block elements only these newer "plus d" sets should be used, since the minor increase in the total number of basis functions is typically offset by the increased accuracy. However, with the need for generating thousands of *ab initio* datapoints for machine learning training sets, and the desire to generate potential energy surfaces on ever larger molecules, there are substantial benefits to minimising the number of basis functions while still retaining acceptable computational accuracy.

As discussed in Chapter 3, effective core potentials (ECPs) reduce the computational effort of a given calculation relative to an equivalent all-electron treatment. It was shown that they achieve this by separating the core and valence electrons, an idea common throughout chemistry. The most popular ECPs within molecular quantum chemistry follow the pseudopotential (PP) approach which was described in detail in the same chapter, and there will continue to be no distinction made between ECPs and PPs herein. Instead, the interested reader is again directed toward the detailed review of Dolg and Cao.[164] Recent developments in the calculation of integrals over ECPs have further reduced computational cost, making their use even more attractive.[236–238] While they are often used to incorporate scalar-relativistic effects for heavier elements without the need for relativistic Hamiltonians, ECPs also offer a partial solution to the problem of large basis sets. Replacing the core electrons with a potential field removes the need for basis functions to describe these electrons. Hence, this reduces the overall size of the basis set. However, the accompanying basis set will need to be specifically paired to a given ECP, increasing the development work required. Within the cc family of basis sets, those paired to small-core ECPs are denoted cc-pVnZ-PP and have been developed for a number of heavier elements, including transition metals,[239,240] alkali metals and alkaline earths,[241] and some of the actinides.[242] Where lighter elements of the periodic table are concerned, correlation consistent basis sets for H and B–Ne, denoted cc-pVnZ-CDF, have been developed for use with the CASINO Dirac-Fock average relativistic pseudopotentials. However, these sets are intended for applications in quantum Monte Carlo calculations.[243]

Chapter 3 also introduced a new generation of correlation consistent effective core potentials (ccECPs), developed by Bennett *et al.* for first and second row atoms,[52,168] and more recently transition metals.[169] These are designed specifically for use in correlated electronic structure methods while retaining transferability between atoms and bonded molecules. They use a many-body approach to the construction of the ECPs to circumvent the poor performance for many-body theories seen with ECPs generated in an effective one-particle setting.[167] To ensure that these new ccECPs could be used in standard electronic structure packages, they chose a commonly used and well established form of the ECP:[244,245]

$$V^{I}_{\mathrm{ECP}}(i) = V^{I}_{\mathrm{loc}}(r_{Ii}) + \sum_{l=0}^{l_{\max}} V^{I}_{l}(r_{Ii})\hat{P}^{I}_{l} \tag{5.1}$$

where $V^{I}_{\mathrm{ECP}}(i)$ is the effective core potential that supplements the electronic Hamiltonian, with i indexing the electrons, I the nuclei, and r_{Ii} the radial distance of electron i from the origin of nucleus I. This potential is angular momentum dependent with $V^I_{loc}(r_{Ii})$ accounting for core-valence repulsion and $V^I_l(r_{Ii})\hat{P}^I_l$ accounting for core-valence orthogonality. Here, $\hat{P}^I_l$ is a projection operator defined as

$$\hat{P}^I_l = \sum_{m=-l}^{l} |Y_{lm}\rangle\langle Y_{lm}| \tag{5.2}$$

To ensure orthogonality l_{max} in equation 5.1 should be equal to the highest angular momentum present in the core.

To be used in quantum chemical calculations, these ECPs require specific basis sets that have been optimised for the valence electrons while using the ECP. Thus, basis sets, denoted ccECP-nZ (n = D–5) herein, were developed in the same work alongside the new ccECPs (large, neon-core) by minimizing the CCSD(T) ground state atomic energies using an even-tempered progression of exponents.[52] In addition to optimising all exponents at the CCSD(T) level, the ccECP-nZ basis sets include a number of design elements that differ from the established cc methodology. For example, the same set of *s*- and *p*-type primitives and contractions were used across all zeta-levels, which is not seen in the current generation of cc basis sets. More significantly, for the second row elements Al–Ar the additional tight-*d* functions demonstrated to be vital for accurate results are not included, and the sets fail to sufficiently capture the nature of the existing all-electron cc basis sets as the *s*- and *p*-type functions do not follow a systematic convergence towards the CBS limit. Despite this, the ccECP-nZ construction produces accurate results for excitation energies to low-lying electronic states and equilibrium bond lengths of several diatomic molecules. Short polar bonds, such as those in AlO or SiO, tend to be over-bound by ccECP-nZ. However, this appears to be relatively common across a number of Ne-core pseudopotentials.[52] As the ccECPs hold the promise of accurate results at a reduced computational cost, the need for new correlation consistent basis sets paired to the ccECPs for the elements Al–Ar is clear, with potential applications in the computation of extended potential energy surfaces, quantum Monte Carlo or *ab initio* molecular dynamics simulations. These ECPs will also then be used in Chapter 6 to help reduce the computational cost of calculations on a sulfur containing compound, HSO_2.

The same assumption that motivates the use of ECPs in quantum chemistry, namely the separation of core and valence electrons, also places limitations on the ultimate accuracy of correlated wave function methods. The so-called frozen-core approximation, where only the valence electrons enter the correlation treatment, neglects intershell correlation effects to reduce computational cost, but relies on said effects being negligible. The effect of core-valence correlation on molecular properties was first studied systematically by Meyer and Rosmus in 1975.[170] Their work showed that core-valence effects could be nearly as important as valence-correlation effects for alkali metal and alkaline earth compounds. Many subsequent investigations have demonstrated that even for main group elements the core electrons must be correlated in, for example, high-accuracy thermochemistry.[171–174]

In addition to including the correlation of more electron pairs, accurately capturing the core-valence effect requires larger basis sets that have been augmented with tight functions, such as the cc-pCVnZ correlation consistent sets.[246] Optimizing additional functions on the energy difference between correlating all electrons and only correlating valence electrons, addresses both the intrashell (core-core) and intershell (core-valence) correlation effects. Subsequent analysis and benchmarking has found that biasing the optimisation towards core-valence correlation, known as weighted core-valence or cc-pwCVnZ, results in basis sets that converge more rapidly towards the CBS limit for core correlation.[247] It is noted that for the second row elements it is common practice to exclude the low-energy 1s electrons from the correlation treatment, even in "core-valence" calculations. Indeed, the cc-pCVnZ and cc-pwCVnZ basis sets have been optimised under this assumption.

As mentioned in Chapter 3, it is principally the dynamic polarization of the atomic cores by the valence electrons that is the physical origin of this core-valence correlation effect.[175] This means that these effects, along with static polarization of the cores in the molecular environment, can be accounted for with a core polarization potential (CPP). The development and history of the CPP approach has been reviewed by Dolg and Cao,[164] based on the pioneering work of Meyer and co-workers,[177–179] and Fuentealba and co-workers.[176] Chapter 3 showed that the interaction between a valence electron and the core, λ, is proportional to α_λ, the core dipole polarizability, leading to:

$$V_{\text{CPP}} = -\frac{1}{2}\sum_{\lambda} \alpha_\lambda \mathbf{f}_\lambda^2 \tag{5.3}$$

where $\mathbf{f}_\lambda$ is the electric field generated at a core by all other cores and the valence electrons, i. This electric field is given by:

$$\mathbf{f}_\lambda = \sum_{\lambda} \frac{\mathbf{r}_{\lambda i}}{r_{\lambda i}^3} g_\lambda(r_{\lambda i}) - \sum_{\mu(\neq\lambda)} \frac{Q_\mu \mathbf{r}_{\lambda\mu}}{r_{\lambda\mu}^3} g_\lambda(r_{\lambda\mu}) \tag{5.4}$$

where a cutoff function, $g_\lambda(r)$, has been introduced to limit the field to the core region:

$$g_\lambda(r) = [1 - \exp(-\gamma_{\lambda n} r^2)]^n \tag{5.5}$$

The parameter γ is fitted to suitable reference data. Two common forms of the cutoff function are used, one is the Fuentealba/Stoll form where $n = 1$,[176] and the other the Müller/Meyer form where $n = 2$.[177] The value of γ is dependent on the functional form chosen.

There have been a small number of investigations where CPPs have been used in conjunction with an all-electron model.[248–251] Perhaps, the most notable work was by Nicklass and Peterson,[252] where it was demonstrated that the core-valence effect on the spectroscopic constants of first-row diatomic molecules can be accurately reproduced with a CPP. However, there has been considerably more interest in using CPPs alongside the ECP approximation, where the core electrons have been removed from the system. This combination promises the attractive proposition of accurate and efficient calculations that take into account core-valence correlation effects, without having to add large numbers of additional functions to the basis sets or significantly increasing the number of correlated electrons.

The goal of the present work is to develop new correlation-consistent basis sets for the second row elements Al–Ar specifically matched to the ccECPs of Bennett *et al.* The resulting basis sets, denoted (aug-)cc-pV(n+d)Z-ccECP (n = D, T, and Q), follow the established cc basis set design principles and include the tight-d functions required for accurate properties of molecules containing second row elements. It is noted that only basis sets up to quadruple-ζ have been developed, as the CPP code in Molpro does not support orbital angular momentum shells above g.[224,225,253] Benchmark calculations on several homonuclear and heteronuclear diatomic molecules are presented to validate the performance of these ECP-based basis sets relative to existing all-electron basis sets. New CPP parameters for Al–Ar have also been optimised and benchmark calculations carried out to demonstrate their efficacy in the computation of core-valence correlation effects.

5.2 COMPUTATIONAL DETAILS

All electronic structure calculations in this work were carried out in the Molpro[224,225,253] package of programs. The BFGS or simplex algorithms[75] were used for parameter optimisation during basis set development. For the primitive Hartree-Fock (HF) sets, exponents were optimised in symmetry-equivalenced HF calculations, whereby contraction coefficients were extracted from Molpro following the general contraction method of Raffenetti.[254] For all correlating exponents, optimizations were at the coupled-cluster with single and double excitations (CCSD) level. For open-shell species the Molpro implementation of UCCSD methods, which are spin-unrestricted in the CCSD calculations, but use restricted open-shell HF (ROHF) orbitals, was used.

All benchmarking calculations on the new ccECP basis sets, denoted (aug-)cc-pV(n+d)Z-ccECP, were carried out at the coupled-cluster with single, double, and perturbative triple excitations [CCSD(T)][255] level and were compared to equivalent all-electron calculations. All atomic correlated calculations used symmetry-equivalenced HF reference orbitals, and electron affinities were calculated using diffuse-augmented basis sets. Atomistic benchmarks of the ionisation energy were calculated by subtracting the total energy of the neutral atom from the total energy of the cation, while electron affinities were calculated by subtracting the total energy of the anion from the total energy of the neutral atom. For diatomic molecules the equilibrium bond length (R_e), harmonic frequency (ω_e), and dissociation energy (D_e) were calculated from a seven-point polynomial fit (Dunham analysis).[256]

5.3 METHODS

The initial basis set development of the present work closely follows that of the cc-pVnZ sets for Al–Ar by Woon and Dunning[231], albeit with the ccECP replacing the Ne core. Briefly, Hartree-Fock optimised primitives are developed for double-, triple-, and quadruple-ζ basis sets. Following this, correlation-consistent polarization functions are determined and added to the HF primitives. Additional tight-d correlating functions[235] were also optimised, and diffuse-augmented functions were optimised for the lowest energy states of the anions. Initially, a full family of (aug-)cc-pV(n+d)Z-ccECP basis sets (where n = D, T, and Q) were developed for sulfur, which were subsequently used as guidelines for the rest of the row.

5.3.1 HARTREE-FOCK PRIMITIVE SETS

The largest difference between the ECP-based basis sets of this work and the all-electron cc-pVnZ sets is a decrease in the number of primitive s and p functions due to the removal of the core electrons. As the ccECPs selected for this work define a large Ne-core, the primitive sets can be significantly reduced from the (12s8p), (15s9p) and (16s11p) of the analogous DZ, TZ and QZ all-electron sets. However, the principles dictating the choice of primitive set remain the same; a systematic decrease in the basis set incompleteness error in atomic HF calculations and maintaining the qualitative nature of the outermost exponents. The resulting primitive set sizes of (6s5p), (8s7p) and (9s8p) smoothly converge towards the HF/CBS limit. The HF primitive sets were then generally contracted to [1s1p] using atomic orbital coefficients from symmetry-equivalenced HF calculations on the electronic ground states of the atoms.

5.3.2 CORRELATING FUNCTIONS

The number of correlating functions to add to the contracted primitive sets was determined by following the familiar cc approach of using an even-tempered expansion to investigate the incremental lowering of the correlation energy. The resulting cc groupings of functions match those of the analogous all-electron sets and are depicted for sulfur in Figure 5.1. The even-tempered exponents were subsequently used as starting

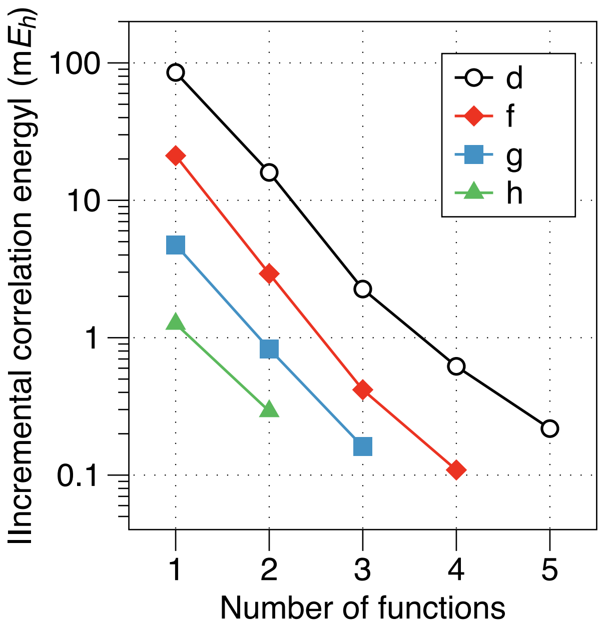


Figure 5.1. Contribution of d–h angular momentum functions to the UCCSD correlation energy for the electronic ground state of the S atom. A set of uncontracted (9s8p) functions were used as a base for these calculations.

points for unconstrained optimisation of a 1d function for the DZ basis set, 2d1f functions for the TZ basis set, and 3d2f1g functions for the QZ basis set.

The work of Blaudeau *et al*, Christiansen, and Peterson has indicated that single s-type primitives are poor correlating functions in basis sets designed for use with ECPs.[257–259] To establish whether this also applies to ccECPs, and whether it also affects p-type angular momentum functions for the second row elements, the correlation energy for sulfur obtained at the UCCSD level by adding successive s and p functions is shown in Figure 5.2. These functions were added to the contracted

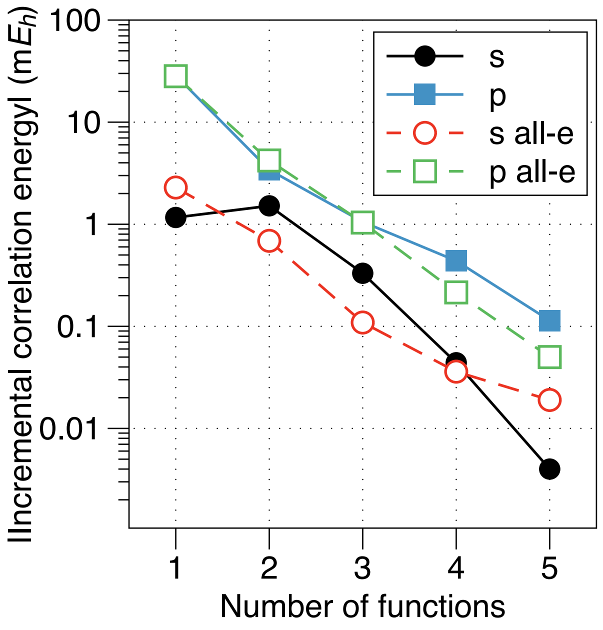


Figure 5.2. Contribution of s and p correlating functions to the UCCSD correlation energy for the electronic ground state of the S atom. All-electron (all-e) results use the [3s2p]+(3d2f1g) functions from the cc-pVQZ basis set as a base.

QZ HF primitives developed above, along with the QZ higher angular momentum correlating functions to form a [1s1p]+(3d2f1g) base. The results from analogous all-electron calculations, using the [3s2p]+(3d2f1g) taken from the cc-pVQZ set, are also shown.

Focusing initially on p-type functions, it can be seen that both the ECP-based and all-electron functions produce a smooth decrease in incremental correlation energy as successive functions are added, and that the correlation energy recovered is similar for both cases. In contrast, for *s*-type functions the second ECP-based function recovers a larger amount of correlation energy than the first. After which, subsequent functions proceed to smoothly decrease incremental correlation energy. This is even more striking when comparing to the all-electron case as the first *s*-type all-electron function recovers roughly twice as much correlation energy as the ECP-based equivalent. An analysis of the exponents indicates that the first ECP-based function has a relatively diffuse exponent, confirming the work of Christiansen.[258] Given Figure 5.2 it would appear logical to include (2*s*1p) correlating functions for a DZ basis set, (3*s*2p) for TZ and (4*s*3p) for QZ. However, initial testing, shown in Table 5.1, demonstrated that inclusion of the additional *s*-type correlating functions tends to cancel for relative energies (making negligible difference to the resulting spectroscopic constants). For example, the dissociation energy for S_2 at the QZ level is 99.91 kcal mol^{-1} and 99.75 kcal mol^{-1} for the 4*s*4p and 5*s*4p ECP basis sets, respectively, and for Al_2 the analogous values are identical (32.65 kcal mol^{-1}). It seems that these extra functions are most important in smaller basis sets, as the ionisation potential and electron affinity for S differs by closer to ± 1 kcal mol^{-1} at the DZ level; the IP for DZ with 2*s*2p functions is 222.49 kcal mol^{-1}, and for DZ with 3*s*2p functions it is 221.57 kcal mol^{-1}. While the EA for DZ is 39.78 kcal mol^{-1} and 40.70 kcal mol^{-1} with 2*s*2p and 3*s*2p functions, respectively. However, in general the values benchmarked are not greatly affected by the addition of an extra s function, and thus, the decision was taken to retain the standard cc groupings of s and p correlating functions of (1*s*1p), (2*s*2p) and (3*s*3p) for DZ–QZ, respectively. This keeps the number of contracted functions as small as possible. As is common practice for cc basis sets, the final s and p correlating functions were uncontracted from the HF sets.

Table 5.1. Spectroscopic constants (D_E / kcal mol^{-1}, R_e / Å, ω_e / cm^{-1}), ionisation energies (kcal mol^{-1}), and electron affinities (kcal mol^{-1}) of sulfur and aluminium comparing three families of basis sets: the ccECP basis sets developed here; those same sets with an extra correlating *s*-function; and the equivalent all electron basis set.

s & p Functions		Basis set	D_E	R_e	ω_e	IP	EA
S/S$_2$	2s2p	cc-pV(D+d)Z-ccECP	85.21	1.9169	712.69	222.49	39.78
	3s3p	cc-pV(T+d)Z-ccECP	96.46	1.8983	725.66	231.47	44.33
	4s4p	cc-pV(Q+d)Z-ccECP	99.91	1.8929	728.16	235.13	46.56
	3s2p	cc-pV(D+d)Z-ccECP	85.74	1.9158	709.91	221.57	40.70
	4s3p	cc-pV(T+d)Z-ccECP	95.29	1.9011	718.86	231.79	44.48
	5s4p	cc-pV(Q+d)Z-ccECP	99.75	1.8932	727.00	235.18	46.56
		cc-pV(D+d)Z	85.33	1.9189	709.80	222.27	41.04
		cc-pV(T+d)Z	95.47	1.9057	718.80	232.81	44.97
		cc-pV(Q+d)Z	100.13	1.8969	728.69	236.14	47.02
Al/Al$_2$	2s2p	cc-pV(D+d)Z-ccECP	27.92	2.7476	277.56	137.99	7.71
	3s3p	cc-pV(T+d)Z-ccECP	31.89	2.7056	285.66	137.43	9.64
	4s4p	cc-pV(Q+d)Z-ccECP	32.65	2.7041	285.04	137.64	9.94
	3s2p	cc-pV(D+d)Z-ccECP	28.19	2.7484	277.02	135.42	8.30
	4s3p	cc-pV(T+d)Z-ccECP	31.61	2.7065	285.34	137.31	9.68
	5s4p	cc-pV(Q+d)Z-ccECP	32.65	2.7043	285.48	137.64	9.94
		cc-pV(D+d)Z	28.33	2.7472	279.61	134.82	8.42
		cc-pV(T+d)Z	31.75	2.7220	285.01	137.00	9.84
		cc-pV(Q+d)Z	32.67	2.7139	284.96	137.54	10.09

5.3.3 ADDITIONAL TIGHT-D FUNCTIONS

An additional tight-d function was added to each of the DZ, TZ, and QZ basis sets to avoid the problems previously noted in studies of molecules containing second-row elements.[232,234,260] For TZ and QZ the exponents of the d functions were fixed, and a tighter exponent was optimised at the (U)CCSD level. The tight-*d* function was then fixed and the remaining d-type exponents allowed to relax in a subsequent (U)CCSD optimisation. For the DZ set the new tight-*d* exponent was determined through a scaling of the TZ exponent by a ratio of $\zeta_2(\mathrm{TZ})/\zeta_3(\mathrm{QZ})$, following Ref. [235] (where ζ_2 is the second most diffuse function in the TZ basis, and ζ_3 is the third most diffuse function in the QZ basis).

The resulting composition of the cc-pV(*n*+d)Z-ccECP sets are shown in Table 5.2, along with the analogous ccECP-*n*Z and all-electron cc-pV(*n*+d)Z sets. It can be seen that the basis sets developed in this work have significantly fewer primitive functions than either of the alternatives, and, as one would expect, fewer contracted functions than the all-electron cc-pV(*n*+d)Z. This comparison also highlights the lack of tight-*d* functions in the ccECP-*n*Z sets. The exponents for all of the basis sets developed for Al–Ar can be found in Appendix B.

5.3.4 DIFFUSE AUGMENTING FUNCTIONS

To improve the results for calculations on anions, electron affinities and polarizabilities, additional diffuse functions were optimised for each basis set to produce aug-cc-pV(*n*+d)Z-ccECP sets. An additional function was added to each angular momentum shell present in the standard basis, and the exponents were energy-optimised for the anion of the respective element, such as the $^2\mathrm{P_u}$ state of the sulfur anion. The tight-*d* functions were excluded from the basis set for this optimisation in keeping with standard methods. Diffuse s and p functions were optimised at the Hartree-Fock level, while higher angular momentum polarization functions were optimised at the (U)CCSD level.

Table 5.2. Composition of the valence correlating ccECP based correlation consistent basis sets developed in this work for Al–Ar. The ccECP-*n*Z[52] and all-electron cc-pV(*n*+d)Z[235] sets are shown for comparison.

Basis set	Composition
cc-pV(D+d)Z-ccECP	(6s5p2d)/[2s2p2d]
cc-pV(T+d)Z-ccECP	(8s7p3d1f)/[3s3p3d1f]
cc-pV(Q+d)Z-ccECP	(9s8p4d2f1g)/[4s4p4d2f1g]
ccECP-DZ	(11s11p1d)/[2s2p1d]
ccECP-TZ	(12s12p2d1f)/[3s3p2d1f]
ccECP-QZ	(13s13p3d2f1g)/[4s4p3d2f1g]
cc-pV(D+d)Z	(12s8p2d)/[4s3p2d]
cc-pV(T+d)Z	(15s9p3d1f)/[5s4p3d1f]
cc-pV(Q+d)Z	(16s11p4d2f1g)/[6s5p4d2f1g]

5.4 CORE POLARIZATION POTENTIALS

The adjustment of the CPPs follows the method outlined by Nicklass and Peterson,[252] although a 1s2s2p core was chosen during the adjustment to reflect the heavier elements of the current work. A second notable deviation from the earlier work is that while Nicklass and Peterson adjusted their B–F CPPs to reproduce experimental ionization energies, the decision was made to reproduce CCSD/CBS limit estimates of the first ionization energy. More specifically, the aim was to reproduce the core-valence effect on the ionization energy ($\Delta\mathrm{IE}^{\mathrm{CBS}}_{\mathrm{core}}$). This is computed as $\Delta\mathrm{IE}^{\mathrm{CBS}}_{\mathrm{core}} = \mathrm{IE}^{\mathrm{CBS}}_{\mathrm{CV}} - \mathrm{IE}^{\mathrm{CBS}}_{\mathrm{Val}}$, where $\mathrm{IE}^{\mathrm{CBS}}_{\mathrm{CV}}$ is the ionization energy assembled from the HF energies, the core-valence correlation energies, and the valence-valence correlation energy (that is, excluding the core-core correlation energy that cannot be recovered using a CPP). The $\mathrm{IE}^{\mathrm{CBS}}_{\mathrm{Val}}$ term is constructed from the HF energies and the valence-valence correlation energy in the usual way. Each of these energetic terms are extrapolated to the CBS limit using the Karton-Martin extrapolation formula for HF energies,[261] and the formula of Helgaker and co-workers for correlation energies.[262,263] All extrapolations used cc-pCV5Z and cc-pCV6Z basis set results.[247,264]

The γ parameters (see equation 5.5) were subsequently adjusted to reproduce $\Delta\text{IE}^{\text{CBS}}_{\text{core}}$ for Al–Cl based on the (U)CCSD/cc-pV(Q+d)Z valence-only ionization energies. Separate γ parameters were determined for both the $n = 1$ and $n = 2$ forms of the cutoff function. The values of these parameters are given in Table 5.3, along with the core dipole polarizabilities, α, reproduced from Johnson *et al.*[265] It can be seen that the γ values for the $n = 2$ cutoff function are significantly larger than $n = 1$, which is consistent with the observations of Nicklass and Peterson for B–F.[252]

Table 5.3. Core polarization potential cutoff parameters (γ, see equation 5.5) for the atoms Al–Cl, adjusted for both the Fuentealba/Stoll ($n = 1$) and the Müller/Meyer ($n = 2$) forms of the cutoff function. Also presented are the core dipole polarizabilities (α), with respect to the neon isoelectronic series, obtained from Ref. [265].

	$\gamma_{(n=1)}$ (a_0^{-2})	$\gamma_{(n=2)}$ (a_0^{-2})	α
Al	1.5324	4.7998	0.2649
Si	1.7544	5.4614	0.1624
P	1.9926	6.1635	0.1057
S	2.5742	7.8453	0.07205
Cl	2.7965	8.5214	0.05093

To identify which form of the cutoff function to use, the sensitivity of the (U)CCSD/cc-pV(Q+d)Z/CPP first ionization energy of sulfur to the values of γ is quantified through the absolute change in ionization energy ($|\Delta\text{IE}|$) as γ is varied from $\gamma_{\text{opt}} - 1$ to $\gamma_{\text{opt}} + 1$, where γ_{opt} are the optimised values of Table 5.3. The resulting plot in Figure 5.3 clearly demonstrates that the Müller/Meyer form ($n = 2$) of the cutoff function is much less sensitive to the value of γ, thus it is preferred herein.

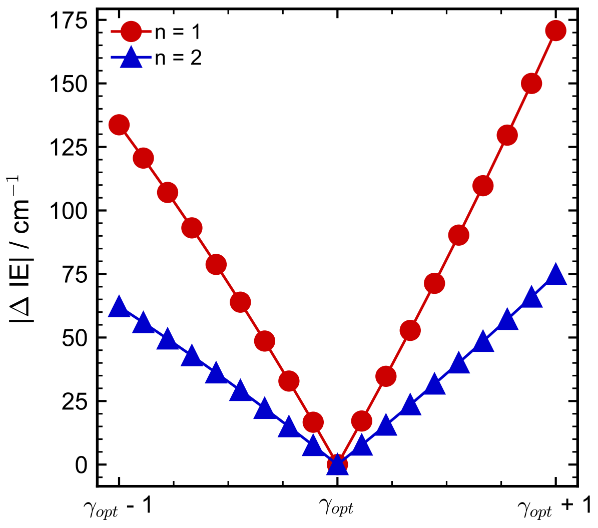


Figure 5.3. The absolute change in ionization energy (|ΔIE|) for the first ionization energy of sulfur at the (U)CCSD/cc-pV(Q+d)Z/CPP level, where the value of the cutoff parameter (γ) is varied from $\gamma_{\text{opt}} - 1$ to $\gamma_{\text{opt}} + 1$. |ΔIE| is plotted for both the Fuentealba/Stoll ($n = 1$) and Müller/Meyer ($n = 2$) forms of the cutoff function.

5.5 RESULTS & DISCUSSION

5.5.1 BASIS SET BENCHMARKS

Atomistic benchmarks

For the elements Al–Cl, ionization energies and electron affinities have been calculated using the basis sets developed in this work, and subsequently compared with those calculated using the cc-pV(n+d)Z sets of Dunning and co-workers and the PP-based ccECP-nZ basis sets of Bennett *et al.*[52] The calculated ionization energies are presented in Table 5.4, where it can be seen that all three basis set families perform approximately equally. The newly developed basis sets slightly outperform the ccECP-nZ basis sets at all levels with mean average deviations of −1.84 kcal mol^{-1} at the DZ level, 0.22 kcal mol^{-1} at TZ, and 0.15 kcal mol^{-1} at QZ (compared to −4.14, −0.63, and −0.34 kcal mol^{-1} at the DZ, TZ, and QZ level for the ccECP-nZ basis sets). For the all electron calculations,

Table 5.4. Ionisation energies (kcal mol^{-1}) at the CCSD(T) level of theory for the atoms Al–Cl.

Basis	Al	Si	P	S	Cl
cc-pV(D+d)Z-ccECP	137.99	186.13	238.85	222.49	285.31
cc-pV(T+d)Z-ccECP	137.43	187.21	241.58	231.47	292.45
cc-pV(Q+d)Z-ccECP	137.64	187.79	242.51	235.13	296.43
ccECP-DZ	138.67	187.37	240.81	225.69	289.72
ccECP-TZ	137.74	187.78	242.26	232.63	293.94
ccECP-QZ	137.65	188.00	242.92	235.85	297.56
cc-pV(D+d)Z	134.82	183.42	236.23	222.27	284.82
cc-pV(T+d)Z	137.00	187.01	241.24	232.81	293.16
cc-pV(Q+d)Z	137.54	187.59	242.13	236.14	296.87
Experiment[266–270]	138.04	187.99	241.83	238.91	299.05

a convergence with basis set towards a maximum value is seen for all atoms. This is mostly seen for both of the ECP basis sets, with the exception of Al, which has a much higher DZ value than would be expected in both cases (Al with ccECP-*n*Z actually trends downwards towards a minimum). The DZ to TZ increment is largest for sulfur for all three families of basis sets (+6.94 kcal mol^{-1} for ccECP-*n*Z, +8.98 kcal mol^{-1} for cc-pV(*n*+d)Z-ccECP, and +10.54 kcal mol^{-1} for cc-pV(*n*+d)Z). Overall, the agreement with experiment is good and the error introduced by the use of an ECP appears to be minimal.

Table 5.5 shows the electron affinities for Al-Cl. These have all been calculated using basis sets augmented with additional diffuse functions as it is well known that such functions are necessary for the correct description of anions.[163] From Table 5.5, it is clear that the aug-cc-pV(*n*+d)Z-ccECP basis sets lead to smaller values of the electron affinities than the corresponding aug-cc-pV(*n*+d)Z basis sets, although there is relatively good agreement throughout. This effect becomes smaller as the basis set size is increased, with a mean average deviation of −1.04 kcal

Table 5.5. Electron affinities (kcal mol^{-1}) at the CCSD(T) level of theory for the atoms Al–Cl.

Family	nZ	Al	Si	P	S	Cl
aug-cc-pV(n+d)Z-ccECP	DZ	7.71	28.95	6.63	39.78	78.07
	TZ	9.64	31.84	13.43	44.33	80.51
	QZ	9.94	32.34	15.30	46.56	83.03
aug-ccECP-nZ	DZ	4.09	24.11	−6.12	27.55	65.47
	TZ	9.63	31.88	13.39	44.36	80.51
	QZ	9.03	32.32	12.89	43.89	80.44
aug-cc-pV(n+d)Z	DZ	8.42	29.72	8.15	41.04	79.01
	TZ	9.84	32.07	14.17	44.97	80.92
	QZ	10.09	32.45	15.90	47.02	83.29
Experiment[271–274]		9.98	32.04	17.22	47.90	83.31

mol^{-1} at the DZ level, −0.44 kcal mol^{-1} at the TZ level, and −0.32 kcal mol^{-1} at the QZ level. The convergence of the aug-cc-pV(n+d)Z-ccECP results with basis set is generally smooth. However, there is a more significant DZ to TZ increment for P than for any of the other elements, similar to what is observed with the all-electron aug-cc-pV(n+d)Z sets. Experimental electron affinities are presented in Table 5.5 to provide context for the CCSD(T) values calculated in this work. However, it is note there are significant post-CCSD(T) effects,[275] including scalar relativistic and spin-orbit splitting, that we do not include in calculated values of Table 5.5. The all-electron aug-cc-pV(n+d)Z results should instead be considered as the "ground-truth" values that the ECP-based sets aim to reproduce.

Comparing the newly-developed aug-cc-pV(n+d)Z-ccECP with the aug-ccECP-nZ sets that use the same ECPs, it can be seen that the new sets offer a significant improvement at both the DZ and QZ levels, relative to the all-electron results. Indeed, the mean average deviation between aug-ccECP-nZ and aug-cc-pV(n+d)Z of −10.24 kcal mol^{-1} at the DZ level,

-0.44 kcal mol^{-1} at the TZ level, and -2.04 kcal mol^{-1} at the QZ level highlights significant convergence problems with the aug-ccECP-*n*Z sets as the TZ results are closer to the limiting value than the QZ results. It should also be noted that the electron affinity of P with CCSD(T)/aug-ccECP-DZ is -6.12 kcal mol^{-1}, with the neutral atom predicted to be lower in energy than the anion. Further analysis indicates this is due to the augmenting "diffuse" *p* exponent of Bennett *et al.* being too tight; replacing this with the analogous diffuse exponent from aug-cc-pVDZ results in an electron affinity of the correct sign (+6.55 kcal mol^{-1}).

Diatomic molecule benchmarks

Dissociation energies, equilibrium bond lengths, and harmonic frequencies were calculated for the homonuclear diatomic molecules Al_2–Cl_2 as well as for sulfur oxide (SO) using the basis sets developed in this work, as well as with the cc-pV(*n*+d)Z and ccECP-*n*Z sets for comparison purposes. These spectroscopic constants were calculated through a seven-point polynomial fit (Dunham analysis[256]) for the ground state of each dimer. The ground state of Al_2 is $^3\Pi_u$ so the CCSD(T) calculation used symmetry-equivalenced reference orbitals.[276]

Table 5.6 shows the dissociation energies for all six molecules, where it can be seen that the new ECP-based sets lead to smaller values of the dissociation energy, compared to the equivalent all-electron cc-pV*n*Z basis, at the DZ and QZ level, but a higher value at the TZ level. However, the absolute difference is typically within 1 kcal mol^{-1} (except for P_2/cc-pV(D+d)Z-ccECP, where it is 1.24 kcal mol^{-1}), and the mean average deviations are -0.55, $+0.48$ and -0.27 kcal mol^{-1} at the DZ, TZ and QZ level, respectively. The convergence with basis set is smooth and follows the general trend of the all-electron sets. In contrast, the mean average deviation between ccECP-*n*Z and cc-pV(*n*+d)Z sets are 10.45 kcal mol^{-1} at the DZ level, 2.28 kcal mol^{-1} at the TZ level, and 0.91 kcal mol^{-1} at QZ level. Comparing the mean average deviations of the new cc-pV(*n*+d)Z-ccECP sets with the ccECP-*n*Z sets shows that there

Table 5.6. Dissociation energies (kcal mol^{-1}) at the CCSD(T) level of theory for the diatomic molecules Al_2–Cl_2 and SO.

Family	*n*Z	Al_2	Si_2	P_2	S_2	Cl_2	SO
cc-pV(*n*+d)Z-ccECP	DZ	27.92	61.58	91.15	85.21	43.69	100.05
	TZ	31.89	71.28	105.63	96.46	53.70	118.35
	QZ	32.65	73.92	111.09	99.91	56.51	122.39
ccECP-*n*Z	DZ	24.65	52.29	75.81	71.32	36.38	89.73
	TZ	31.08	69.16	102.47	92.53	50.84	114.68
	QZ	32.48	73.36	110.41	98.91	55.55	121.53
cc-pV(*n*+d)Z	DZ	28.33	62.33	92.39	85.33	43.76	100.74
	TZ	31.75	70.98	105.37	95.47	53.51	117.35
	QZ	32.67	74.00	111.67	100.13	57.04	122.20
Experiment[276–278]		31.70	75.60	117.20	102.90	59.70	126±1

are significant improvements for all basis set qualities.

Calculated equilibrium bond lengths for all six diatomic molecules are presented in Table 5.7. Comparing the newly-developed sets to the all-electron results again shows that the use of an ECP leads to shorter bond lengths. This effect is greatest for the TZ basis sets, as shown by the mean average deviation across all six dimers: −0.0030 Å at the DZ level, −0.0102 Å at the TZ level, and −0.0054 Å at the QZ level. Comparing the dimers themselves, the deviation is larger for the two lighter atom pairs, Al_2 and Si_2. If the mean average deviation is calculated for only these two diatomic molecules then the results are −0.0057 Å for the DZ level, −0.0162 Å for the TZ level, and −0.0088 Å for the QZ level. The best agreement occurs for SO, with values of −0.0000, −0.0016, and −0.0024 Å for DZ, TZ, and QZ calculations, respectively. Generally, the new basis sets converge in a similar manner to the all-electron sets.

As mentioned previously, Ne-core ECPs are known to produce shorter bond lengths than all-electron calculations,[52] which is consistent with the results of Table 5.7. The SO equilibrium bond length having the

Table 5.7. Equilibrium bond lengths (Å) at the CCSD(T) level of theory for the diatomic molecules Al_2–Cl_2 and SO.

Family	*n*Z	Al_2	Si_2	P_2	S_2	Cl_2	SO
cc-pV(*n*+d)Z-ccECP	DZ	2.7464	2.2787	1.9222	1.9169	2.0266	1.5150
	TZ	2.7028	2.2494	1.9021	1.8983	1.9959	1.4900
	QZ	2.7043	2.2462	1.8968	1.8929	1.9934	1.4838
ccECP-*n*Z	DZ	2.8410	2.3644	1.9833	1.9887	2.0999	1.5527
	TZ	2.7223	2.2640	1.9115	1.9117	2.0157	1.4976
	QZ	2.7087	2.2497	1.8980	1.8957	1.9964	1.4855
cc-pV(*n*+d)Z	DZ	2.7472	2.2831	1.9247	1.9189	2.0350	1.5150
	TZ	2.7220	2.2625	1.9097	1.9057	2.0079	1.4916
	QZ	2.7139	2.2542	1.9019	1.8969	1.9966	1.4862
Experiment[276–278]		2.701	2.246	1.8934	1.8892	1.9879	1.4811

smallest deviation from all-electron results adds extra evidence to this; the oxygen atom does not use an ECP, thus the over-binding effect is smaller. This may also explain why the lighter second-row elements have the greatest over-binding, as a larger proportion of the electrons are replaced by an ECP. Table 5.7 also shows that the ccECP-*n*Z sets produce bond lengths that are systematically too short as the size of the basis tends towards the limit. Conversely, at the DZ and TZ level the bonds lengths are too long, relative to the all electron calculations, leading to poor convergence with basis set size.

Table 5.8 shows the calculated harmonic frequencies for all six molecules across the three basis set families. Comparing the cc-pV(*n*+d)Z-ccECP sets with cc-pV(*n*+d)Z sets reveals good agreement throughout, with better agreement as the zeta-level increases. The absolute mean average deviation is 3.7 cm^{-1} at the DZ level, 3.3 cm^{-1} at the TZ level, and 0.8 cm^{-1} at the QZ level. Convergence with the basis set size is generally smooth, except for Al_2 which has a larger harmonic frequency at the TZ level than the QZ level. However, this is consistent with the all-

Table 5.8. Harmonic frequencies (cm^{-1}) at the CCSD(T) level of theory for the diatomic molecules Al_2–Cl_2 and SO.

Family	*n*Z	Al_2	Si_2	P_2	S_2	Cl_2	SO
cc-pV(*n*+d)Z-ccECP	DZ	279.3	498.0	762.3	712.7	519.1	1077.5
	TZ	287.1	513.6	775.0	725.7	552.3	1153.0
	QZ	285.6	515.4	782.3	728.1	555.6	1154.2
ccECP-*n*Z	DZ	263.3	463.4	707.1	659.6	492.6	1023.9
	TZ	281.2	505.2	767.9	711.4	541.9	1138.8
	QZ	284.5	513.1	780.8	725.9	555.0	1151.6
cc-pV(*n*+d)Z	DZ	279.6	497.6	764.6	709.8	512.9	1087.8
	TZ	285.0	511.4	773.5	718.8	549.1	1149.3
	QZ	285.0	514.8	783.2	728.7	557.4	1154.3
Experiment[276–278]		285.8	510.98	780.77	725.65	559.70	1149.22

electron calculations. Comparing the new cc-pV(*n*+d)Z-ccECP sets with the ccECP-*n*Z sets, it can be seen that the new basis sets offer significant improvements across the board, with large improvements observed with small basis sets. This is particularly striking for ccECP-DZ, which has a mean average deviation, relative to the all-electron calculation, of −40.4 cm^{-1}.

Overall, it can be seen that the (aug-)cc-pV(*n*+d)Z-ccECP basis sets developed in this work produce results that are significantly closer to those from the all-electron (aug-)cc-pV(*n*+d)Z sets than the (aug-)cECP-*n*Z sets that use the same ECPs. This is particularly evident at the DZ level, but remains significant even at QZ. The new basis sets converge smoothly with basis set size and in the vast majority of cases agree well with (aug-)cc-pV(*n*+d)Z results at the QZ level, implying that any error introduced by the use of the ECPs is small when combined with appropriate basis sets. However, for the lighter elements equilibrium bond lengths are underestimated. As cc-pV(*n*+d)Z-ccECP and ccECP-*n*Z appear to tend towards the same limits for bond lengths, this indicates that the error is likely to be related to the use of a large-core ECP.

5.5.2 CPP BENCHMARKS

As CPPs account for both core-valence effects and the static polarization of atomic cores by the molecular environment, the comparison of the effect of core-valence correlation (that is, the difference between a given property in valence-only and core-valence correlating calculations) between all-electron and ECP with CPP-based calculations, as usually done with correlation consistent basis sets for core-valence correlation, is not a fair one. Instead, to validate the performance of the combination of the basis sets and CPPs developed in this work, we compared the computed spectroscopic properties of the homonuclear diatomic molecules Al_2–Cl_2 computed with the all-electron cc-pCV*n*Z basis sets with those from cc-pV(*n*+d)Z-ccECP/CPP. All calculations were carried out with the CCSD(T) method on the electronic ground state of the molecule and all-electron calculations used a 1*s* frozen core. In the cc-pV(*n*+d)Z-ccECP/CPP calculations, all electrons not replaced by the ECP were correlated and the Müller/Meyer form of the CPP cutoff function was used. In keeping with methods employed by Peterson and Dunning,[247] atomic calculations carried out for dissociation energies were not performed using symmetry-equivalenced reference orbitals.

Tables 5.9, 5.10 and 5.11 display the calculated dissociation energies, equilibrium bond lengths and harmonic frequencies, respectively, of the five homonuclear diatomic molecules. Focusing initially on the dissociation energies in Table 5.9, it can be seen that the combination of the cc-pV(*n*+d)Z-ccECP basis and CPP reproduces the all-electron cc-pCV*n*Z results well. The agreement between the two approaches increases with basis set size and, fortuitously, the cc-pV(*n*+d)Z-ccECP/CPP results tend to be slightly closer to the basis set limit than the equivalent cc-pCV*n*Z. The dissociation energies computed with the CPP approach also converge smoothly towards the basis set limit. However, as with the all-electron results, there is a large difference between DZ and TZ results.

Table 5.9. Dissociation energies (kcal mol^{-1}) at the CCSD(T) level of theory for the homonuclear diatomic molecules Al_2–Cl_2, including core-valence correlation effects.

Family	*n*Z	Al_2	Si_2	P_2	S_2	Cl_2
cc-pV(*n*+d)Z-ccECP/CPP	DZ	28.61	62.77	92.65	86.53	44.47
	TZ	32.21	72.08	106.92	97.54	54.23
	QZ	32.74	74.49	112.15	100.87	57.01
cc-pCV*n*Z	DZ	28.37	61.82	90.82	83.19	42.76
	TZ	31.66	71.03	105.95	95.76	53.65
	QZ	32.60	74.24	112.54	100.67	57.13
	5Z	32.90	75.22	114.71	102.39	58.53
	6Z	33.02	75.65	115.66	103.14	59.10
Experiment[276–278]		31.70	75.60	117.20	102.90	59.70

Table 5.10. Equilibrium bond lengths (Å) at the CCSD(T) level of theory for the homonuclear diatomic molecules Al_2–Cl_2, including core-valence correlation effects.

Family	*n*Z	Al_2	Si_2	P_2	S_2	Cl_2
cc-pV(*n*+d)Z-ccECP/CPP	DZ	2.7177	2.2596	1.9095	1.9063	2.0157
	TZ	2.6795	2.2338	1.8909	1.8885	1.9871
	QZ	2.6823	2.2313	1.8862	1.8836	1.9853
cc-pCV*n*Z	DZ	2.7525	2.2909	1.9329	1.9331	2.0465
	TZ	2.7162	2.2580	1.9054	1.9027	2.0045
	QZ	2.7018	2.2458	1.8952	1.8913	1.9914
	5Z	2.6992	2.2431	1.8922	1.8880	1.9874
	6Z	2.6979	2.2422	1.8913	1.8868	1.9859
Experiment[276–278]		2.701	2.246	1.8934	1.8892	1.9879

Table 5.11. Harmonic frequencies (cm^{-1}) at the CCSD(T) level of theory for the homonuclear diatomic molecules Al_2–Cl_2, including core-valence correlation effects.

Family	*n*Z	Al_2	Si_2	P_2	S_2	Cl_2
cc-pV(*n*+d)Z-ccECP/CPP	DZ	281.9	502.3	769.3	719.3	526.8
	TZ	289.4	517.4	781.2	732.1	556.3
	QZ	286.0	518.8	788.4	733.7	558.7
cc-pCV*n*Z	DZ	278.7	494.5	759.3	703.4	512.5
	TZ	283.7	511.2	777.0	720.5	551.8
	QZ	286.8	516.9	788.4	732.3	560.0
	5Z	286.0	517.5	790.6	734.6	564.1
	6Z	286.4	518.1	791.9	736.0	565.1
Experiment[276–278]		285.8	511.0	780.8	725.7	559.7

The equilibrium bond lengths presented in Table 5.10 are an interesting set of results. The agreement between the two approaches for any given basis set zeta-level is less than ideal, with the CPP-based approach underestimating the all-electron bond length in all cases. The agreement does increase with zeta-level and typically the agreement is also better for the heavier elements under consideration, the latter of which is consistent with the trend for valence-only results in Table 5.7. The consistently too-short bond lengths does lead to some fortuitous error cancellation, with cc-pV(D+d)Z-ccECP/CPP producing bond lengths roughly equivalent to the considerably more expensive cc-pCVTZ results. The cc-pV(T+dZ-ccECP/CPP) lengths are also similar to those of cc-pCV5Z, although by this point the Al_2 and Si_2 bonds are already shorter than the cc-pCV6Z results and potentially beyond the all-electron basis set limit. This acts as another reminder that large-core ECPs for these elements can lead to bond lengths that are too short.

Trends similar to those observed for the dissociation energies are also seen for the harmonic frequencies in Table 5.11. The agreement between the approaches is relatively good and improves with basis set size. Again, the cc-pV(D+d)Z-ccECP/CPP result is better than the cc-pCVDZ result and approaches that of the cc-pCVTZ result. This extends to other zeta-levels, such that cc-pV(*n*+d)Z-ccECP/CPP gives results roughly equivalent to cc-pCV(*n*+1)Z. Thus, due to the use of ECP, the lack of core-correlating basis functions, and because fewer electrons are entering the correlation treatment, a considerable saving of computational effort is achieved.

5.5.3 TIMING BENCHMARKS

The savings in computational cost are demonstrated in Tables 5.12 and 5.13, which present the CPU times for single point CCSD(T) energy evaluations on pentathiolane (S_5). The timings are broken down into the components of the calculation, such as time spent in integral evaluation, HF etc. All of the calculations were carried out in C_1 symmetry and the timings are taken as the mean average of three individual calculations that were all performed on a single core of an Intel i7-8700 CPU with 16 GB of RAM. The exception to this is the cc-pCVQZ calculation in Table 5.13, where the value is for a single run due to the very-long runtime.

Table 5.12 compares the time taken for valence-only calculations with the new cc-pV(*n*+d)Z-ccECP sets against cc-pV(*n*+d)Z, hence it shows any reduction in computational cost from the use of an ECP and the basis sets developed in this work. As percentages relative to the total time taken for the analogous cc-pV(*n*+d)Z calculation, cc-pV(*n*+d)Z-ccECP takes 68% of the time at the DZ level, 83% at TZ, and 85% at QZ. Individually, the largest percentage time reductions are seen for the Hartree-Fock step, with the cc-pV(*n*+d)Z-ccECP calculation taking 33% of the time at the DZ level, 51% at TZ, and 68% at QZ. The perturbative-triples calculation is effectively the same at the TZ and QZ levels (97% at DZ, 100% at TZ, and 101% at QZ), which is unsurprising as the same number of electrons are being correlated. The largest magnitude of time reduction is seen for the CCSD step, with 9.2 s (60%) at the DZ level, 45.7 s (72%) at TZ, and 334.5 s (68%) at QZ. Although, this is only due to the relative length of time

Table 5.12. CPU timing breakdown (s) for a valence-only single point CCSD(T) energy evaluation on pentathiolane. Total time is broken down into integrals (Int.), Hartree-Fock (HF), integral transformation (Trans.), coupled cluster with single and double excitations (CCSD), and perturbative triples [(T)].

Family	nZ	Int.	HF	Trans.	CCSD	(T)	Total
nZ-ccECP	DZ	0.6	0.6	0.3	14.2	10.8	26.7
	TZ	5.4	7.7	3.6	115.8	155.9	288.6
	QZ	57.3	75.9	32.9	722.7	1363.1	2252.0
cc-pV(n+d)Z	DZ	1.6	1.8	0.7	23.8	11.1	39.2
	TZ	10.6	15.0	5.6	161.5	155.4	348.3
	QZ	88.5	112.2	44.4	1057.2	1349.4	2651.8

these take compared to the integrals, HF, and integral transformation calculations.

Table 5.13 shows that significantly more impressive gains in computational efficiency are observed when a CPP is used for core-valence correlation rather than the conventional approach using a cc-pCVnZ basis. Again, as percentages relative to the time taken for the analogous cc-pCVnZ calculation (with a 1s frozen core), cc-pV(n+d)Z-ccECP/CPP takes 4% at the DZ level, 2% at the TZ level and 1% at QZ. Crucially, cc-pV(T+d)Z-ccECP/CPP takes less than half the CPU time of the cc-pCVDZ calculation, and cc-pV(Q+d)Z-ccECP/CPP is almost an order of magnitude faster than the lower zeta-level cc-pCVTZ. Breaking these timings down shows the opposite trend in terms of where the largest gains are found, with the perturbative-triples taking 3%, 1%, and 1% of the time for DZ, TZ, and QZ respectively (reflecting the number of electrons correlated), while the HF calculations take 9%, 5%, and 4% of the time. Comparing Table 5.13 with Table 5.12 indicates that the calculation using the CPP can be marginally faster than the analogous calculation without it. The timing breakdowns indicate that this is primarily due to a reduction in the time taken for HF self-consistent field convergence.

Table 5.13. CPU timing breakdown (s) for a core-valence single point CCSD(T) energy evaluation on pentathiolane. Total time is broken down into integrals (Int.), Hartree-Fock (HF), integral transformation (Trans.), coupled cluster with single and double excitations (CCSD), and perturbative triples [(T)].

Family	*n*Z	Int.	CPP	HF	Trans.	CCSD	(T)	Total
*n*Z-ccECP/CPP	DZ	0.6	0.2	0.3	0.3	14.0	10.9	26.3
	TZ	4.9	1.3	3.6	3.3	111.7	155.9	280.7
	QZ	57.8	16.6	35.4	29.9	681.7	1358.2	2179.6
cc-pCV*n*Z	DZ	2.5	—	3.3	3.6	387.8	343.8	741.0
	TZ	41.4	—	70.5	72.1	5227.2	11977.2	17388.6
	QZ	548.3	—	999.4	1303.4	47384.5	150229.0	200464.9

5.6 CONCLUSIONS

Correlation consistent basis sets for the second row elements Al–Ar, denoted cc-pV(*n*+d)Z-ccECP, have been developed for use with the large-core correlation consistent ECPs of Bennett *et al.*[52] The new basis sets are designed as a replacement for the sets that were provided with these ECPs, ensuring that the resulting basis sets follow the established correlation consistent design philosophy and include the tight-d functions that are known to be important for the second row. The basis sets are accompanied by newly-adjusted CPPs to recover the effects of core-valence correlation, moreover it is found that the $n = 2$ (Müller/Meyer) form of the CPP cutoff function is least sensitive to the value of the cutoff parameter.

Benchmarking of the new basis sets at the CCSD(T) level on atomic ionization energies and electron affinities, and on the dissociation energy and harmonic frequencies of diatomic molecules, demonstrates that the new cc-pV(*n*+d)Z-ccECP sets produce results significantly closer to those from the all-electron cc-pV(*n*+d)Z, when compared to the ccECP-*n*Z provided with the ECPs. In fact, the new basis sets reproduce the all-electron benchmark well and any deviations decrease with basis set size. It also follows that the new cc-pV(*n*+d)Z-ccECP sets converge smoothly towards the basis set limit, as expected for a correlation consistent basis. Analysis of computed equilibrium bond lengths for homonuclear diatomic molecules reveals that they are underestimated with both the cc-pV(*n*+d)Z-ccECP and ccECP-*n*Z sets, and that the underestimation increases for the lighter elements. As both of the ECP-based basis sets appear to be converging towards the same limit, it appears that this overbinding is likely due to the ECP, rather than the basis set. However, it is unclear whether this could be addressed by further adjustment of the ECPs, or that it is unavoidable when using a large-core ECP for the second row elements.

Although the time gains are not particularly large for the use of ECPs (approximately 15% faster), over the course of thousands/tens of thousands of calculations, even small decreases in computation time add up. For systems containing second row atoms, the use of these ECP basis sets could be used to reduce *ab initio* calculation time and speed up dataset generation for machine learning fit potential energy surfaces.

The CPPs are benchmarked on the spectroscopic properties of homonuclear diatomic molecules and calibrated against the all-electron core-valence cc-pCV*n*Z results. For dissociation energies and harmonic frequencies, the cc-pV(*n*+d)Z-ccECP/CPP approach produces accurate values with minimal computational expense compared to extensive calculations with large numbers of correlated electrons. As with the valence-only cc-pV(*n*+d)Z-ccECP calculations, equilibrium bond lengths are underestimated for the lighter elements. For all of the spectroscopic properties, at a given zeta-level the CPP-based results are slightly closer to the basis set limit than the equivalent cc-pCV*n*Z. This is most pronounced for DZ, where cc-pV(D+d)Z-ccECP/CPP results are roughly comparable to cc-pCVTZ. In general, the accuracy and low computational expense of the CPP approach here is a continuation of what was observed by Nicklass and Peterson for B–F[252] and is an under-explored tool for large molecular systems or high-throughput computation/benchmarking.

6 | The HSO_2 potential energy surface

To further test the capabilities of machine learned potential energy surfaces it is helpful to explore a system with a more varied surface than H_2O, in particular the four atom system HSO_2 will be considered. The addition of an extra atom gives six internal degrees of freedom, which is a considerable jump in complexity for the potential energy surface and its fitting. As a result, the double many body expansion (DMBE) method becomes significantly more complex, and as a consequence machine learning seems to be a better alternative. However, ML is not a 'free-lunch' in this regard, in particular, a larger structural space means more data is needed in the training set to sufficiently cover the important regions. To do this as efficiently as possible, data was chosen in a smarter way, first running a selection of cheaper CASSCF/aug-cc-pV(T+d)Z-ccECP calculations to identify areas of the surface that are very high in energy (greater than 1000 kJ mol^{-1} above the lowest energy structure), then excluding these from the higher level CASPT2/aug-cc-pV(T+d)Z-ccECP dataset generation. The sulfur aug-cc-pV(n+d)Z-ccECP basis sets[53] developed in Chapter 5 were also used instead of all-electron equivalent sets, in an effort to speed up the generation of thousands of *ab initio* data points. Finally the firefly algorithm developed in Chapter 4 was used in conjunction with PES Learn[135] to attempt to minimise the number of datapoints needed to fit an accurate surface.

6.1 THE HISTORY OF THE HSO_2 SURFACE

HSO_2 is known to play an important part in both atmospheric and combustion chemistry.[279,280] It was first reported experimentally by McDowell *et al.*, who detected the molecule using electron paramagnetic resonance spectroscopy and reported the structure as a symmetric σ radical of HSO_2.[281] It has since been subject to a number of experimental studies,[280,282–287] with most of the focus on calculating kinetic information about the system under various conditions. It has been shown to inhibit CO oxidation, where SO_2 reacts with O to form SO_3 under the presence of NO,[279] moreover, it has been shown by Alzueta *et al.* to both inhibit *and* promote the oxidation of fuels, depending on the specific stoichiometric ratios of SO_2.[280]

It is therefore clear that the specific behaviour of this molecule is complex, and a better understanding of its electronic structure could lead to experimental insights. As a consequence, the HSO_2 system has been the focus of many theoretical explorations of its properties and behaviour through the use of *ab initio* calculations. Initially, Boyd *et al.* performed HF/STO-3G* calculations (where the * indicates that this basis set includes d-type polarisation functions on second row atoms) on six sulfonyl radicals attempting to predict geometries, describe the behaviour of the radical, and calculate bond energies.[288] They report a strong preference for a staggered HOSO conformation with a dihedral angle of 81.9° (an example of this structure calculated at the CASPT2/aug-cc-pV(5+d)Z level of theory is shown in Figure 6.1). This structure was reported to be 170 kJ mol^{-1} lower in energy than the HSO_2 structure (which is also shown in Figure 6.1). Impressively, the qualitative nature of these conclusions holds true today, with only the values of specific bond lengths, angles, and energies having been improved with higher levels of theory. The use of potential energy surfaces has been a major contributor to these improvements, and there have been several efforts to study the kinetics, reaction mechanisms, and dynamics of the system through surfaces with varying levels of theory and coverage.[41,42,289–307]

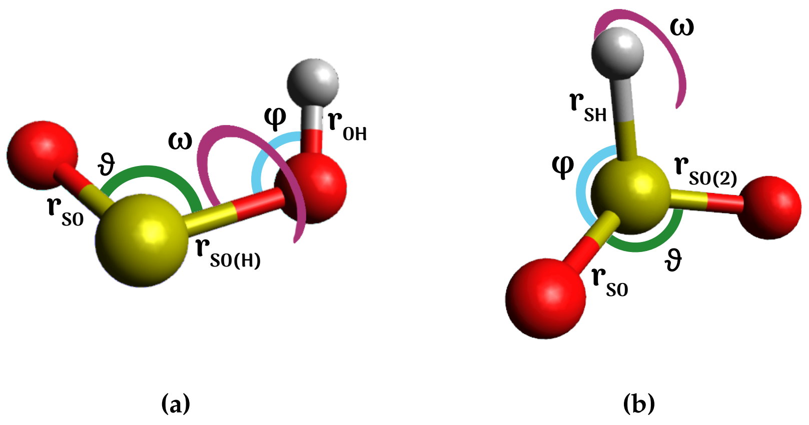


Figure 6.1. The structures of **(a)** HOSO and **(b)** HSO_2, showing the staggered nature of the equilibrium ground state minimum in HOSO ($\omega = 44.79°$). Calculated at the CASPT2/aug-cc-pV(5+d)Z level of theory.

Binns and Marshall reported a potential energy surface for $H+SO_2$,[290] using HF/3-21G$^{(*)}$ and MP2/3-21G$^{(*)}$ calculations to find stationary points on the surface,[†] and MP4/6-31G* to calculate optimised geometry energies of HOSO and HSO_2. They report that a similarly skewed HOSO structure (dihedral angle 59.5°) is more stable than HSO_2 by 145 kJ mol^{-1}, in qualitative agreement with Boyd *et al.*. Some efforts have been made to characterise the system using DFT optimised structures, but at the B3LYP/6-31+G(d) level HOSO presents a planar *cis-* geometry, rather than a skewed structure,[296] in contrast to much of the previous work.

A study on the kinetics of SH + O_2 + Ar by Goumri *et al.* reports MP2(full)/6-31G(d) geometry optimisations for new configurations of this system, HSOO and SOO. They also used the QCISD(T) method (quadratic configuration interaction with single, double, and perturbative triple excitations, which can be described as an approximation to a

[†]The 3-21G$^{(*)}$ basis sets are "3-21G basis sets with the addition of a complete set of six second order Gaussian primitives". These basis sets "should not be viewed as a full-polarized basis set; rather it is best seen as a representation that, in as simple manner as possible, is able to account for the participation of d-symmetry functions in the bonding about second-row atoms."[308]

higher cost CCSD(T) calculation), along with the 6-311+G(3df,3p) basis sets to carry out single point energy calculations for the optimised geometries.[293] These structures start to identify a different, yet important, section of the potential energy surface that is much higher in energy, and Zhou *et al.* reported a "Computational Study of the Reaction SH + O2"[300] that strongly highlights that there are two distinct regions of this potential energy surface (shown in Figure 6.2). A higher energy region described by the HS + O_2 and HSO + O limits, containing isomers of HOOS. The second is a low energy region, which contains the ground state geometry, HOSO, and describes the OH + SO and H + SO_2 limits. This is shown in Figure 6.2.

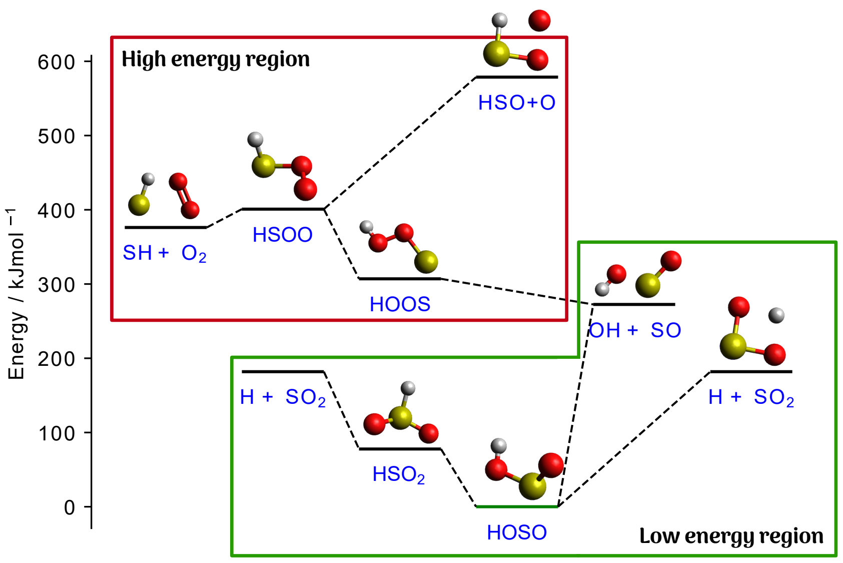


Figure 6.2. Visualisation of the high and low energy regions of the HSO_2 PES. Calculated at the CASPT2/aug-cc-pV(Q+d)Z level.

The low energy region of the surface has been described at a high level of theory, with CCSD(T)/aug-cc-pV(T+d)Z results being reported by Rodrígues-Linares *et al.*[304] A van der Waals structure, H···SO_2 is shown as part of the surface, confirming work by Varandas,[41,42] but due to the nature of the CCSD(T) method employed, the full dissociation limits of the surface were not explored as the method cannot fully describe bond dissociation[309]. They do report the skewed HOSO structure as the ground state equilibrium geometry with a dihedral angle of 24.57°. Unfortunately, but perhaps not surprisingly, the high energy region has received significantly less attention, with only a few studies covering solely this section of the surface.[293,297,300] The high energy nature of the structures in this region results in them containing more multi-reference character than the lower energy region, which requires the use of computational methods capable of accounting for this (CASPT2 for example). Resende *et al.* show that $SH + O_2 \longrightarrow HSOO$ is thermodynamically unfavourable by 25.40 kJ mol^{-1} through multi-reference MP2 calculations, and Zhou *et al.* use MRCI calculations to identify a number of stationary points between the $SH + O_2$ and $SO + OH$ limits.

There have been several efforts to characterise pathways between the two regions, with Laakso *et al.* identifying HOOS as an intermediate connecting the two regions,[291] and Gourmi *et al.* outlining a full pathway from $H + SO_2 \rightarrow \text{cis}-HOSO \rightarrow HSO_2 \rightarrow HSOO \rightarrow SH + O_2$.[295] Varandas *et al.* have been successful in using the double many-body expansion (DMBE) method to build a *global* potential energy surface (DMBE-PES) using CASSCF/aug-cc-pVTZ data.[41,42] A concern here is that the original correlation consistent basis sets for second row atoms (in this case the sulfur atom) require additional tight d-type functions in order to accurately describe the electronic structure,[260] and a few reports have highlighted the specific sensitivity of the geometry of this molecule to basis set size and method.[299,310] Wheeler and Schaefer showed specifically that the dihedral angle in HOSO is very sensitive to basis set size, and changing the basis set from cc-pV(D+d)Z to cc-pV(T+d)Z changed the ground state geometry from a planar-*cis* configuration to a skewed-*cis* configuration with a dihedral angle of 24.2°. As well as the issue of basis set sensitivity, it has been seen that the high energy region of the surface has significant multi-reference character[293,297,300] and CASSCF calculations are not sufficient for this task, failing to treat the dynamic correlation needed. In support of this, Zhou *et al.* report a failure to identify the HSO···O van der Waals structure identified by Varandas *et al.* at the CASSCF level.[300]

In light of this, Garrido *et al.* carried out a study on the major stationary points of the HSO_2 DMBE-PES, at the CASPT2/aug-cc-pV(T+d)Z level,[51] reporting an updated electronic ground state geometry and identifying three new stationary points on the surface that help define new reaction pathways for $SH + O_2 \longrightarrow H + SO_2$ and $SH + O_2 \longrightarrow OH + SO$. Importantly, this study improves on the DMBE-PES by using a multi-reference method capable of accounting for dynamic correlation and a basis set of sufficient size including tight-d functions for sulfur. Freitas *et al.* have also identified a transition state between HSOO on the high energy region and HSO_2 on the low energy region at the CCSD(T)/aug-cc-pV(T+d)Z and CASPT2/aug-cc-pV(T+d)Z levels, further highlighting the need for an improved *global* PES.[302]

Some of the most recent work on this system, and particularly pertinent to this thesis, is by Qin *et al.*[306,307] They report a CCSD(T)-F12a/aug-cc-pVTZ potential energy surface for the lower energy region of the system ($OH + SO \longrightarrow H + SO_2$) fit using permutation invariant polynomials (PIPs) as inputs for a deep neural network (referred to as PES-2019 herein). Their work was motivated in part by the shortcomings of the global DMBE-PES outlined by Pires *et al.*[303] Comparing the computed rate coefficients of $OH + SO \longrightarrow H + SO_2$ at 500 K to the experimental values of Blitz *et al.*[286] shows two different outcomes. Where experiment shows a sudden drop in rate constant as temperature is increased above 500 K the DMBE-PES predicts no such change.

PES-2019 was fit using 39,200 CCSD(T)-F12a/aug-cc-pVTZ calculations, alarmingly not employing the use of the aug-cc-pV(*n*+d)Z basis sets that have been shown to be of vital importance for second row atoms.[260] The inputs to the hidden layer are low order permutation invariant polynomials (up to a maximum order of 3), "symmetrised monomials of Morse-like variables of internuclear distances,"[306]

$$G = \hat{S} \prod_{i<j}^{4} p_{ij}^{l_{ij}},$$

where $p_{ij} = e^{\frac{-r_{ij}}{\alpha}}$ ($\alpha = 1.0$ Å and r is internuclear distance), and $\hat{S}$ is the symmetrisation operator. The neural network architecture itself is two hidden layers of between 40 and 100 neurons, with two different activation functions between the layers, however the form of these functions is not specified by the authors. Weights and biases were updated to minimise the root mean squared error (RMSE) between the predicted energy and the calculated energy. Training, validating, and testing set ratios were 90:5:5 and the final RMSEs reported were 0.72, 0.98, 1.13, and 0.76 kJ mol^{-1} for the training, validation, testing, and total sets, respectively, with a maximum deviation of 13.51 kJ mol^{-1}. Analysis of the dynamic mechanisms resulting from this surface showed significant differences between PES-2019 and DMBE-PES, with major differences in trajectories of $H + SO_2$ collisions. PES-2019 showed that there were four possible mechanisms for collision: direct, involving just the HOSO well, involving just HSO_2 well, and involving both. Meanwhile the DMBE-PES showed only two possible pathways involving the HOSO well and the HSO_2 well (overwhelmingly dominated by the HOSO pathway at 86%). These differences are largely thought to be caused by the DMBE-PES overestimating the energy barrier of the transition state between HOSO and HSO_2 (299.15 kJ mol^{-1} for DMBE-PES, 219.91 kJ mol^{-1} for PES-2019).

Further to this, Qin *et al.* report an updated UCCSD(T)-F12a/aug-cc-pVTZ PES (herein referred to as PES-2020)[307] attempting to address issues surrounding the higher energy OH + SO entrance channel of the PES-2019 surface. Although technically lying on the 'low-energy' region of the global PES, this species is high enough in energy that multi-reference methods are important for its accurate description. Qin *et al.* report that the UCCSD(T)-F12a/aug-cc-pVTZ successfully describes this section of the surface, but one wonders if this is simply coincidence, considering UCCSD(T)-F12a is not a multi-reference method and the plus-d basis set still has not been used. Regardless, PES-2020 is a newly fit PIP-NN surface in which care has been taken to attempt to properly describe the electronic structure of the entrance channel. Starting with the PES-2019 surface, an iterative process of adding training points was employed to improve the fit in badly described regions of the surface. Eventually

44,700 points of *ab initio* UCCSD(T)-F12a/aug-cc-pVTZ are used to train a PIP-NN in an identical way to PES-2019. Concerningly, the reported ground state equilibrium geometry of HOSO has a dihedral angle of 0° for the level of theory employed, contradicting all current understanding of the global minimum.

PES-2020, while being reasonably high level and appearing to describe the reaction of $OH + SO \longrightarrow H + SO_2$ fully, still only covers the low energy region of the PES. If a new, high level, *global* potential energy surface (analogous to DMBE-PES) is to be developed, then the higher energy region must be included. In this region, multi-reference methods such as CASPT2 must be employed to properly describe the electronic structure of the geometries (shown by Garrido *et al.*).[51] Unfortunately, CASPT2/aug-cc-pV(T+d)Z calculations are expensive, and thousands of *ab initio* points of data are required to build an accurate PES at this level (PES-2019 used 39,200). It is therefore important to find a way of reducing the cost of this surface generation through both reducing the number of calculations, and the level of theory they are run at, while still being able to sufficiently describe the important chemistry. The ccECPs developed in Chapter 5 offer a means of reducing computational cost, and as machine learning has been shown to successfully generate potential energy surfaces of many different systems,[115,121–132] specifically being shown to be successful within the HSO_2 system already,[306,307] it offers a way of fitting a global surface that should, in theory, need less *ab initio* data.

This chapter is split into two distinct sections: the benchmarking of methods and basis sets to establish a minimum level of theory to build a potential energy surface; and the training of a deep neural network using the firefly algorithm (from Chapter 4) to build these surfaces.

6.2 METHOD AND BASIS SET EXPLORATION

The global surface developed by Varandas *et al.* was computed at the CASSCF/aug-cc-pVTZ level of theory, and has been shown to be insufficient to fully describe the behaviour of HSO_2, being further improved by a number of CASPT2/aug-cc-pV(T+d)Z calculations by Garrido *et al.*[51] The aim of this section is to show the importance of the tight-d functions in the aug-cc-pV(n+d)Z basis sets, and the effect that using CASPT2 has, by benchmarking a number of stationary points identified by Garrido *et al.*[51] (shown in Figure 6.3). The geometries of some of these points were re-optimised using a number of different basis sets and methods and compared to best estimate structures to assess what effect the method and basis set has on these structures. The performance of each method/basis set pair was established through relative energies to HOSO as it has been well established as the global minimum,[41,42,289–307] and the root mean squared deviations (RMSDs) to best estimate geometries.

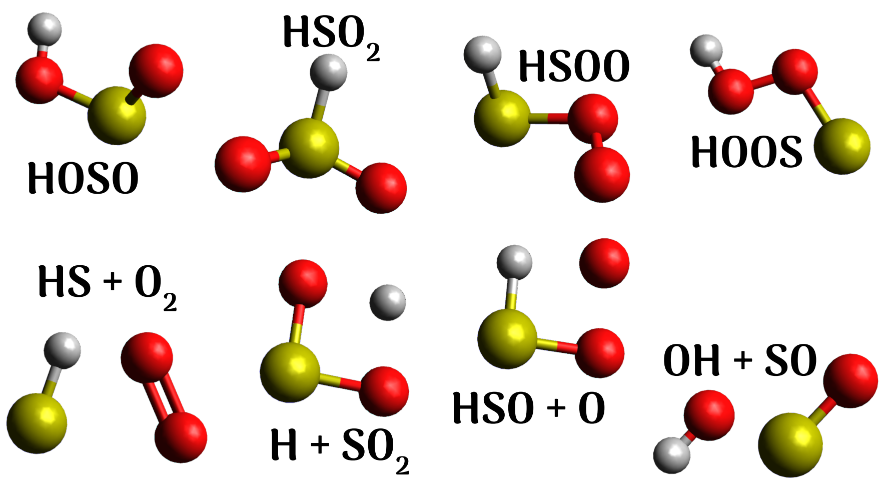


Figure 6.3. Structures of stationary points identified by Garrido *et al.* on the HSO_2 potential energy surface, chosen to benchmark different methods and basis sets. The top row are a number of important minima on the surface and the bottom row are the dissociative limits.

An aspect of this surface that has been identified as particularly important is the electronic ground state equilibrium geometry of HOSO, specifically surrounding the dihedral angle.[299] Therefore, frozen scans of the dihedral angle in HOSO were carried out to determine the effects of basis set and method on this value.

6.2.1 TECHNICAL DETAILS

All calculations were carried out in the Molpro package of programs[224,253] (version 2021.2). Both the CASSCF and CASPT2 methods were used along with the following basis sets: aug-cc-pV*n*Z[231] (aV*n*Z), aug-cc-pV(*n*+d)Z[235] (aV*n*Z+d), and aug-cc-pV(*n*+d)Z-ccECP[53] (aV*n*Z-ccECP, developed in Chapter 5), where n = D, T, and Q. For the low energy region of the surface, extra aV*n*Z+d calculations were carried out at the quintuple-ζ level to act as a 'best estimate' for geometry optimisations. For each of these calculations the active space was selected through the use of the atomic valence active space (AVAS)[155] method within Molpro, where the target orbitals for all-electron calculations were: $3s$, $3p$ for sulfur, $2s$, $2p$ for both oxygens, and $1s$ for hydrogen. The resulting active space was 19 electrons in 13 active orbitals, with 7 closed orbitals for all-electron calculations and 2 closed orbitals for ccECP calculations (as 5 of the core orbitals have been replaced by the ECP).

Relative energies were calculated by subtracting the total energy of a geometry from the total energy of the equivalent HOSO equilibrium geometry calculation at the same computational level. The high energy region of this system contains a number of structures surrounding the stationary points shown in figures 6.2 and 6.3 that are similar in energy, and make large regions of the surface flat.[51] As such it is very difficult to optimise the geometry of these points using the routines found within Molpro, in fact these points were identified by Garrido *et al.* through the analysis of the final fitted surface, and were not calculated beforehand.[51] As such, the relative energy calculations in this benchmarking section are carried out using single point energy calculations on the geometries reported by Garrido *et al.* Optimised geometries of the lower region however, were compared to best estimates calculated at the CASPT2/aV5Z+d level. To establish how well a method/basis pair performed, the root mean squared deviation (RMSD) of the cartesian coordinates, compared to the best estimate geometry, was calculated for each geometry. This was done using the RMSD program by Charnley.[311] Firstly the two molecules are translated to the origin, then a Kabsch algorithm[312] is used to find the best rotation of the two molecules. This minimizes the RMSD between the two structures, providing the best possible comparison.

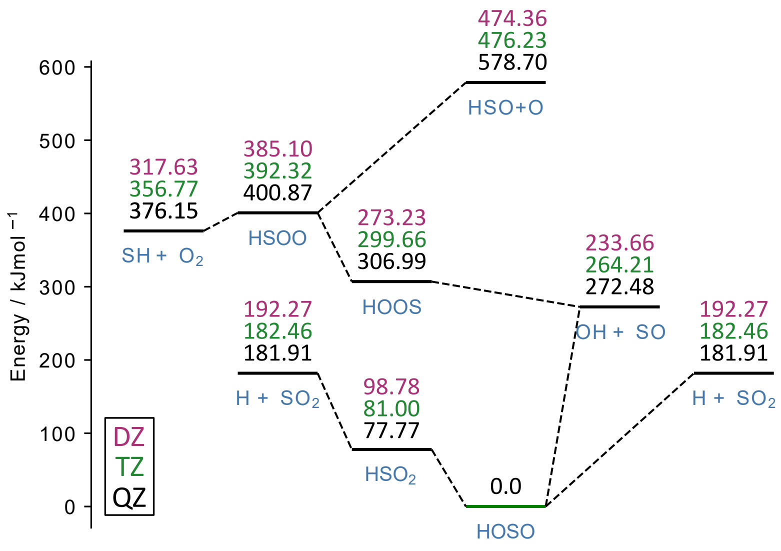
Dihedral angle (ω) scans were also calculated for HOSO at the same levels of theory as the geometry optimisations. The molecule was frozen at its optimised geometry and the dihedral angle incrementally changed from 0° to 180°, 10° at a time, with a CAS calculation carried out at each step.

6.2.2 SINGLE POINT RELATIVE ENERGIES

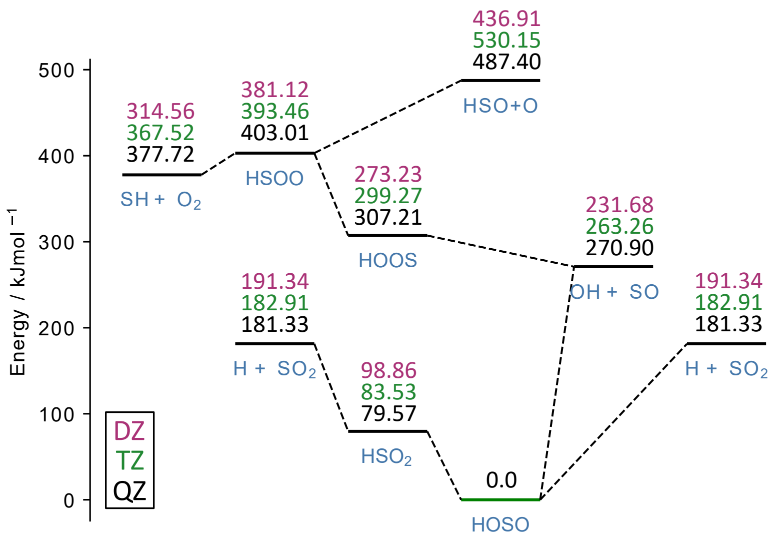
Figure 6.4 is an energy level diagram showing the relative energies of the stationary points shown in Figure 6.3 for both the all electron basis sets with tight-d functions, and the ccECP basis sets. The diagram is plotted at the quadruple-ζ level for both sets of values, while the relative energies of the triple- and double-ζ basis sets are shown in green and purple respectively. In both cases HOSO is set as the global minimum as it has been shown to be the most stable structure for this system.[41,42,289–307]

The aVnZ-ccECP basis sets recreate the all-electron relative energies to within the widely accepted 'chemical accuracy' of 4 kJ mol^{-1} across all stationary points with the largest discrepancies appearing for the dissociation limits of the high-energy region of the surface. The aVnZ-ccECP basis sets perform similarly as basis set size increases, with the average deviation across all points being 1.67, 2.70, and 1.32 kJ mol^{-1} for the DZ, TZ, and QZ basis sets, respectively. The triple-ζ basis set appears to perform the worst in comparison to the all-electron energies, but actually maintains the 4 kJ mol^{-1} accuracy. In general, the lower energy region of the surface is recreated more accurately by the aVnZ-ccECP basis sets than the upper energy region, with the average deviation of the upper region being 2.35 kJ mol^{-1}, while the lower region is 1.00 kJ mol^{-1} at the QZ level, leading to an average deviation of 1.67 kJ mol^{-1}, as noted above.

Most importantly, the overall ordering of stationary points is main-



(a)



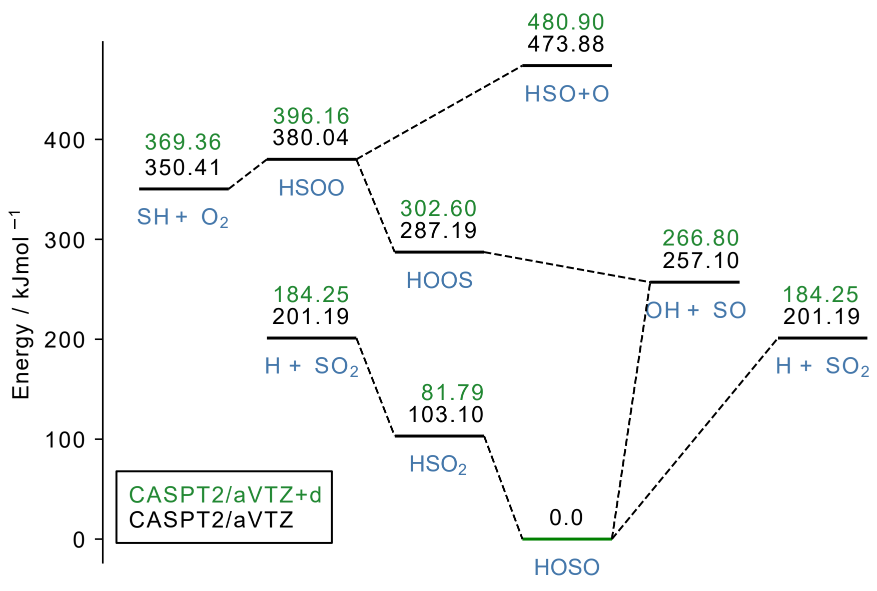
(b)

Figure 6.4. Stationary points on the CASPT2 surface for **(a)** the aV*n*Z+d basis set and **(b)** the aV*n*Z-ccECP basis sets, at the DZ (purple), TZ (green), and QZ (black) level.

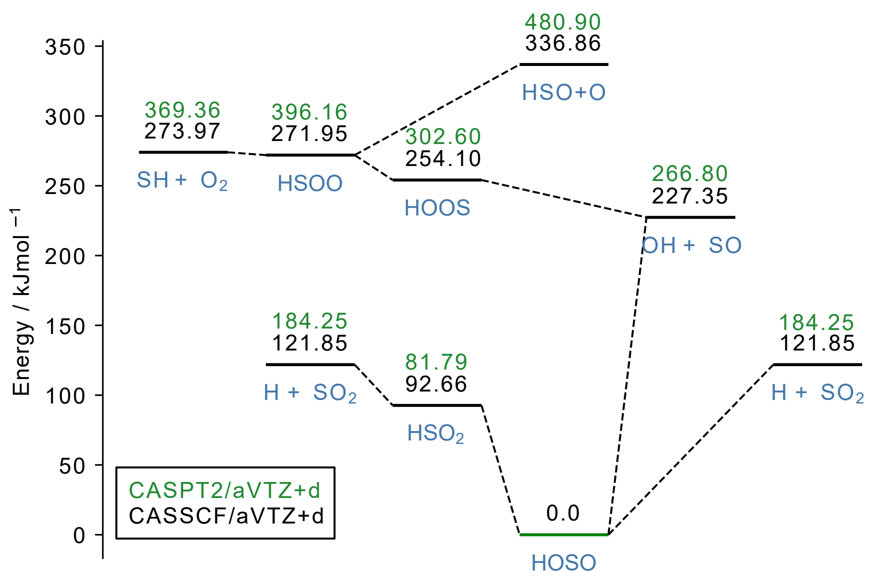
tained when employing the use of an ECP, and the same diverging behaviour of the upper and lower regions of the surface is seen as basis set size increases; that is, increasing the basis set size decreases the relative energies of the lower energy region, and increases the relative energies of the higher energy region. It is imperative that the *n*Z-ccECP basis sets exhibit the same behaviour as the all-electron equivalents if an accurate PES is to be developed using these new basis sets.

The importance of tight-d functions and dynamic correlation. In Chapter 5 the vital importance of extra tight-d functions in the correlation consistent basis sets was highlighted,[232–235] and subsequently the basis sets developed in that chapter included such functions.[53] The lack of the use of these basis sets in the DMBE-PES is reason enough for a second look at a global surface, ignoring the need for a multi-reference method such as CASPT2. Furthermore, the absence of these basis sets in PES-2019 and PES-2020 is both concerning and unexplained, calling into question the accuracy of these surfaces.

Figure 6.5a shows the single point relative energies of CASPT2 calculations with both aVTZ+d and aVTZ basis sets. It is clear that the tight-d functions have a significant effect on all values of the relative energies, for example the relative energy of HSO_2 is over estimated by 21.32 kJ mol^{-1}. The general ordering of energies is the same, however. Conversely, Figure 6.5b shows the effect of using CASSCF over CASPT2. Here it can be seen that the lack of dynamic correlation drastically affects the relative energies of the system, particularly for the high-energy region and the dissociation limits. The relative energy of HSOO is underestimated by 124.21 kJ mol^{-1} and is actually lower in energy than the SH + O_2, switching the order of their relative energy in the surface. It is clear that both plus-d basis sets, and a method capable of recovering dynamic correlation are important for an accurate PES.



(a)



(b)

Figure 6.5. Stationary points on the HSO_2 surface for **(a)** the CASPT2 method with both the aVTZ+d basis sets (green) and the aVTZ basis sets (black), highlighting the need for tight-d functions, and **(b)** the CASPT2 (green) and CASSCF(black) methods, both with the aVTZ+d basis sets, highlighting the importance of dynamic correlation.

6.2.3 OPTIMISED GEOMETRIES

As well as recreating the general order of stationary points, any method used to generate the PES needs to recreate the geometry of this system accurately, particularly the much debated ground state equilibrium geometry.

HOSO The RMSD values in Figure 6.6 are calculated as the difference to the optimised CASPT2/aV5Z+d geometry of HOSO. For all three fam-

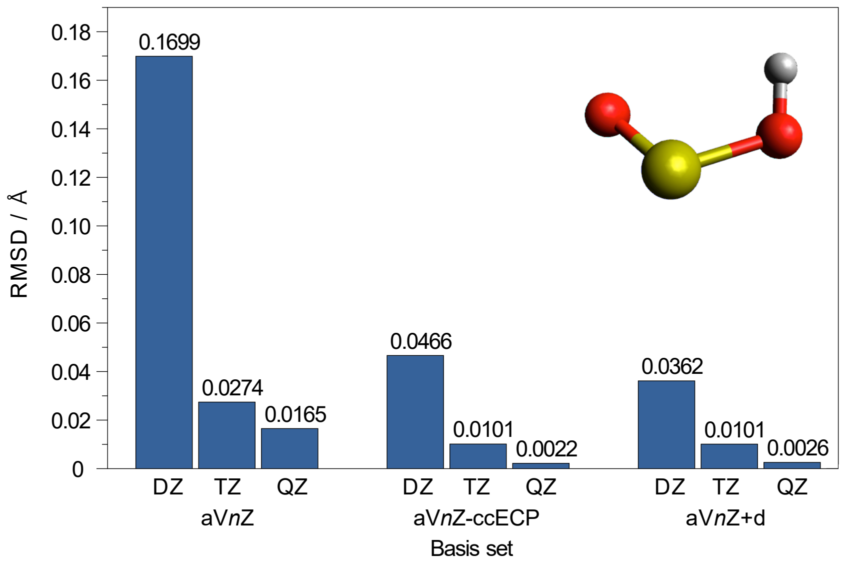


Figure 6.6. CASPT2 RMSD relative to the CASPT2/aV5Z+d ground state optimised geometry of HOSO.

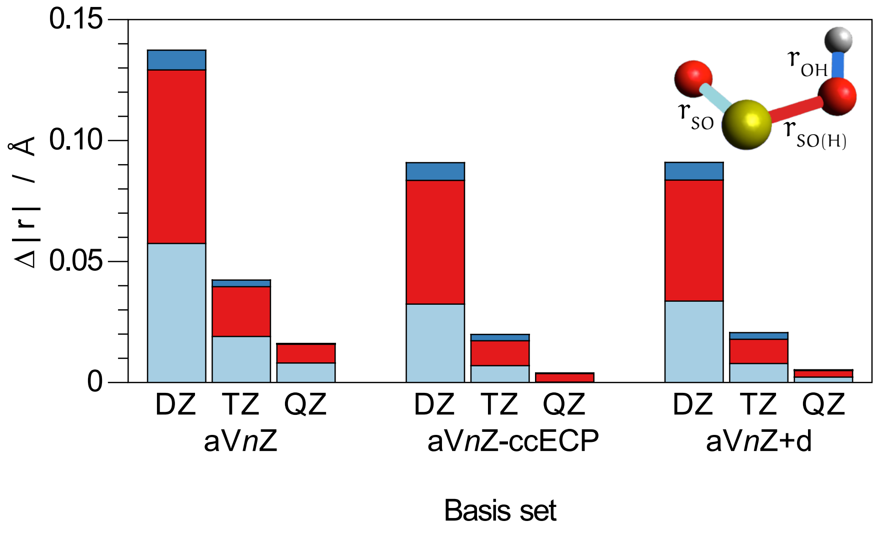
ilies of basis sets, the deviation decreases as basis set size increases. This is entirely expected when the concepts introduced in Chapter 3 (Electronic structure theory) are considered. The larger the number of basis functions, the more accurately the electronic orbitals are modelled and the more accurate the calculations. The aV*n*Z-ccECP calculations perform very similarly to the aV*n*Z+d calculations, with the RMSD values of the triple-ζ basis sets being identical (0.0101 Å), and narrowly outperforming the all electron tight-d sets by 0.0004 Å at the quadruple-ζ level. In contrast, the aVQZ optimisations perform worse than the triple-ζ calculations of the other two sets (0.0165 Å vs. 0.0101 Å), and the aVTZ

RMSD is over twice as large, further highlighting the need for tight-d functions. The double-ζ RMSDs are larger for all three basis set families, but the aVDZ RMSD is significantly worse, at approximately four times more than the other two families (0.0362 Å, 0.0466 Å, and 0.1699 Å for aVDZ+d, aVDZ+d-ccECP, and aVDZ respectively).

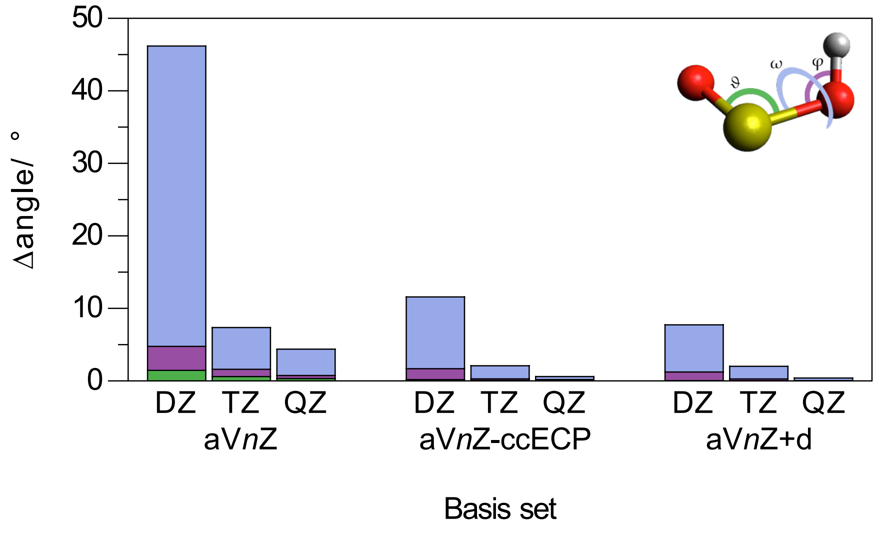
Figure 6.7 breaks down the error in the six degrees of freedom to show which aspects contribute to the RMSD the most. Generally, the errors decrease as basis set size increases, which is not unexpected when considering Figure 6.6. The errors are also larger for the aV*n*Z basis sets than the aV*n*Z-ccECP basis sets.

For bond lengths (Figure 6.7a), the aV*n*Z+d and aV*n*Z+d-ccECP basis sets have very similar magnitudes of error, while the aV*n*Z basis sets have larger bond errors, though this is to be expected based on the RMSD values in Figure 6.6. At the triple-ζ level, the bond errors for the aVTZ+d and aVTZ+d-ccECP basis sets are $\Delta R_{SO} = 0.0078$ Å, $\Delta R_{SO(H)} = 0.0100$ Å, $\Delta R_{OH} = 0.0027$ Å, and $\Delta R_{SO} = 0.0069$ Å, $\Delta R_{SO(H)} = 0.0103$ Å, $\Delta R_{OH} = 0.0026$ Å, respectively, showing very good agreement with each other. The SO(H) bond length consistently has the largest error across all basis set sizes and families (except for aVQZ where the SO bond error is larger at 0.0081 Å verses 0.0077 Å for aVTZ), accounting for approximately 50% of the total error in all cases bar one. In the case of the the aV*n*Z+d-ccECP basis sets the SO(H) bond length error grows from 0.0035 Å (90%) at the QZ level to 0.0103 Å(52%) at the TZ level.

The errors in the bond angles (Figure 6.7b) show similar trends in terms of the errors between the aV*n*Z+d and aV*n*Z+d-ccECP basis sets, having almost identical QZ and TZ errors. At the QZ level all bond angle errors are less than 0.5°, with essentially no difference to the aV5Z+d geometry. At the TZ level the dihedral angle error ($\Delta\omega$) is 1.74° and 1.80° for the two basis sets respectively. In fact, generally, across all basis sets, the dihedral angle is the largest source of error for the bond angles alone, while the error in the SOH bond angle (ϕ) is only significant at the DZ level. That said, the bond angle errors for the TZ and QZ basis sets of the aV*n*Z+d and aV*n*Z+d-ccECP families are very small, and most of the



(a)



(b)

Figure 6.7. **(a)** Bond length errors and **(b)** bond angle errors of HOSO compared to the CASPT2/aV5Z+d ground state optimised geometry. For the bond lengths the separate errors are differenciated by three colours, teal for the r_{SO} bond, red for the $r_{SO(H)}$ bond, and blue for the r_{OH} bond, and the full bar represents to total bond error. Similarly, the separate bond angle errors are represented by blue for the dihedral angle, ω, purple for the SOH angle, ϕ, and green for the OSO angle, θ.

RMSD error comes from the bond lengths.

It is promising that for both the triple-ζ and quadruple-ζ basis sets that the ccECP basis sets predict the geometry in very similar ways to the all-electron tight-d basis sets.

Dihedral angle scans of HOSO The relative energy of four dihedral scans of HOSO at both the CASSCF and CASPT2 levels of theory for the aVTZ+d and aVTZ+d-ccECP basis sets are shown in Figure 6.8. The lowest energy structure is set to 0 kJ mol^{-1} and all other energies are plotted relative to that.

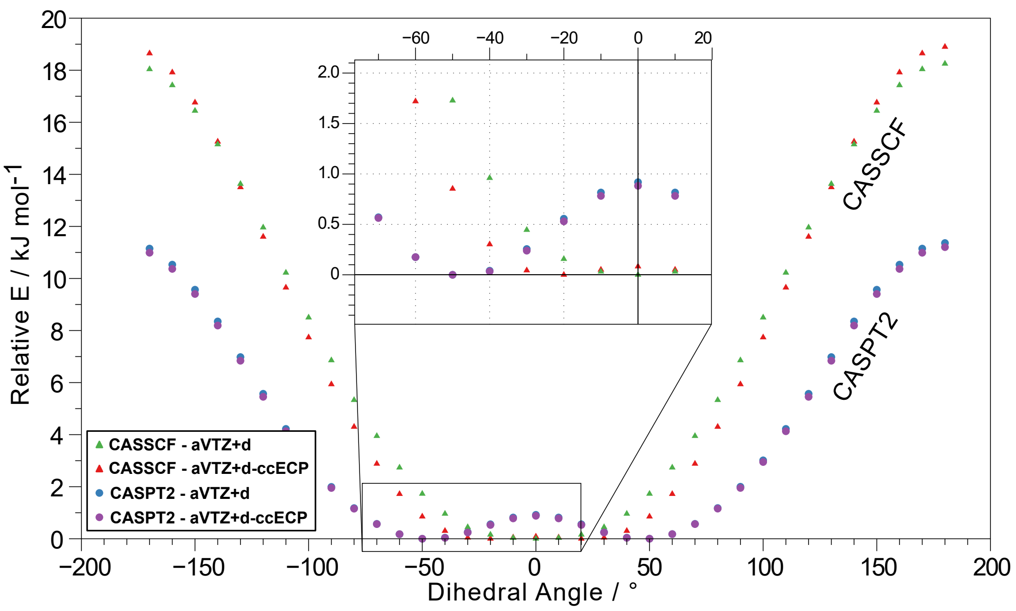


Figure 6.8. Dihedral angle scans of HOSO using the CASSCF and CASPT2 methods with the aVTZ+d and aVTZ-ccECP basis sets. This shows the skewed nature of the CASPT2/aVTZ+d and CASPT2/aVTZ-ccECP structures, while also revealing the fact that the use of an ECP breaks the symmetry of the CASSCF structure.

It can be seen that at the triple-ζ level, both of the CASPT2 calculations show preference for the skewed HOSO structure, with minima at around ±50° compared to the relative energies of the flat structure; 0.88 kJ mol^{-1} and 0.92 kJ mol^{-1} for the all electron and ccECP basis sets, respectively (for comparison the CASPT2/aV5Z+d ground state dihedral angle is

$\pm 44.79^\circ$). The all-electron CASSCF calculations predicts a flat structure with a minimum at 0°, while interestingly, the CASSCF/aVTZ-ccECP calculation predicts a *skewed* structure with minima at around $\pm 20^\circ$ and the flat structure having a relative energy of 0.8 kJ mol^{-1}. The use of an ECP appears to break the symmetry of the CASSCF calculation.

HSO_2 Figure 6.9 shows the RMSD values for the HSO_2 structure. Gen-

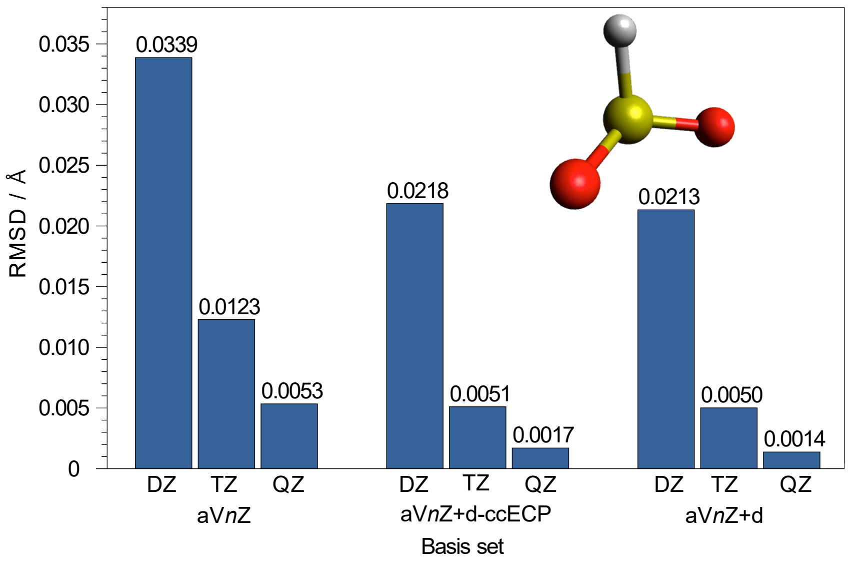


Figure 6.9. CASPT2 RMSD relative to the CASPT2/aV5Z+d ground state optimised geometry of HOSO.

erally the RMSD values are about a factor of two smaller than for HOSO, suggesting that there is less ambiguity in the structure of HSO_2. The same trends between basis set families are seen, with the aV*n*Z basis sets performing worst, while the aV*n*Z+d and aV*n*Z+d-ccECP perform very similarly. Again, the CASPT2/aV*n*Z+d-ccECP calculations predict the geometry well enough to replace the all-electron equivalents.

The accuracy/cost trade off. The geometries of the system are important to get correct, and it is clear from the RMSD values, the bond length and angle errors, and the dihedral angle scans, that using the CASPT2 method with the aV*n*Z+d-ccECP basis sets successfully recreates the all-electron geometries. As the aVTZ-ccECP basis sets perform well, they appear to be a reasonable accuracy-to-cost trade off for use in generating a new global PES for the HSO_2 system.

6.3 BUILDING A POTENTIAL ENERGY SURFACE

It is clear that CASSCF/aug-cc-pVTZ is not a sufficient level of theory to accurately describe this system. A new global potential energy surface at the CASPT2/aug-cc-pV(T+d)Z level, to replace the DMBE-PES, may help to reveal new transition states and new pathways between the upper and lower regions of the system. Chapter 4, as well as work such as that by Qin *et al.*,[306,307] has shown that a machine learning algorithm is a suitable replacement for surface fitting methods such as DMBE, but their use does not come without drawbacks. With six degrees of freedom in a four atom system, the structural space that needs to be covered is still very large, and if a machine learning algorithm is to be useful it needs a reasonably large dataset (~10,000 points) of high level *ab initio* calculations spanning the full space. There are, however a few ways to keep this training set size as small as possible, and minimise the computational effort needed to generate it.

To reduce the computational cost of the *ab initio* calculations themselves, they can be run using the aug-cc-pV(*n*+d)Z-ccECP basis sets benchmarked in Section 6.2. This section showed that the ccECP basis sets can accurately reproduce the features of the all-electron data, and timing benchmarks in Chapter 5 show a 15% increase in speed. Due to the nature of the data generation (described in Section 6.3.1) a large number of extreme, or unusual geometries could be initially chosen for the *ab initio* data generation. In order to reduce the number of failed calculations, and therefore wasted computational effort, a set of lower level (CASSCF) calculations will be run. Then only the successful calculations that are less than 1000 kJ mol^{-1} above the ground state HOSO geometry will be chosen for the CASPT2 calculations. This way a number of expensive CASPT2 calculations that are either not important in the dynamics (too high in energy) or unlikely to converge will be skipped over, saving considerable amounts of computational time. Further to this, using active learning techniques, such as the firefly algorithm in Chapter 4, can reduce the overall size of the dataset for the machine learning algorithm. The initial training dataset can be much more sparse, and the fireflies will fill the sections with the largest fitting error with new *ab initio* data, rather than calculating lots of values in areas of the surface that are not important. The ability for the firefly algorithm to explore the global space, and then hone in on a particularly badly fit section of the surface makes it a useful addition to any machine learning problem.

6.3.1 TRAINING DATA GENERATION

The HSO_2 molecule has six degrees of freedom, three bond lengths and three angles. To generate a starting dataset spanning the whole configuration space, ten different points along each degree of freedom were chosen. The bond lengths spanned from from 0.5 Å to 3.5 Å, and the bond angles were varied from 0° to 180°. This resulted in 1,000,000 geometries generated, spanning the whole configuration space, which were reduced down to 50,000 points through filtering method based on the Euclidean distances between structures. The method takes a grid of geometries (the 1,000,000 points in this case) and creates an evenly spaced subgrid containing a user specified number of points (50,000 here) and generates these points instead. This was achieved through the use of the PES Learn python library,[135] which employs a scheme similar to the structure-based sampling of Dral *et al.*.[91] To further reduce the dataset, any structure that had *any* internuclear distance of less than 0.5 Å after being generated was removed, as these are likely to be very high in energy and not especially interesting in a quantum molecular dynamics calculation. This resulted in a total of 49,260 structures.

CASSCF/aug-cc-pV(T+d)Z-ccECP calculations were run on every structure in this dataset and any calculation that failed to converge, or had a $\Delta E > 0.4$ Hartree (1050 kJ mol^{-1}) above the ground state energy of HOSO (−160.964757 Hartree) was removed from the dataset. This resulted in 20,601 different geometries, for which a CASPT2/aVTZ-ccECP calculation was run for each one. Any calculation that failed to converge was removed from the dataset, and the final dataset consisted of 6,815 CASPT2/ aug-cc-pV(T+d)Z-ccECP energies with their respective geometries. This large drop in dataset size seems to be due to the CASPT2 calculation failing to converge, even with an increased number of iterations. Finally, the stationary points and optimised geometries from Section 6.2 were added into the dataset to seed the NN with important stationary points. Due to the nature of the data generation, the dihedral angle for all calculations lay between 0 ° and 180 °; as the geometry/energy of HSO_2 is symmetric around the dihedral angle, the whole dataset was duplicated and dihedral angle for this second set was multiplied by -1. This has the effect of doubling the size of the training dataset for free, resulting in 13,640 datapoints. Once the full dataset had been generated, it was separated into training, validation, and testing sets at a ratio of 7.5:1.25:1.25 through PES learn's smart random sampling method, where datapoints are chosen to produce training and testing sets that match the energy distribution of the original dataset as close as possible.[135]

6.3.2 MODEL TRAINING

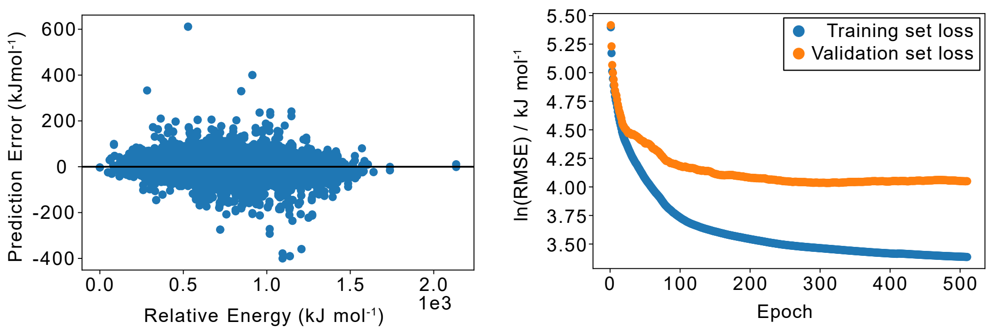
Similarly to the potential energy surface for water in Chapter 4, a deep neural network was chosen to fit a PES for this system. The six degrees of freedom were used as the input vector for the network and the calculated *ab initio* energies were the labels. To fit the surface the same python library, PES Learn,[135] was used to perform a hyperparameter search and train the neural network. The maximum number of hyperparameter search iterations was set to 50 to allow for a larger number of combinations of hyperparameters to be tested. This gives a higher probability that the 'best' hyperparameters are found. The number of points in the training set was chosen as 75% of the full dataset, resulting in 10,230 datapoints. The global minimum energy was forced into the training set, and the set of trial layers was defined as: [16,16,16], [32], [32,32], [64], [64,64], [64,64,64] (similarly to Chapter 4 the number of entries in each set of square brackets indicates the number of hidden layers, and the value of each entry indicates the number of nodes in that layer). There was no random seed set for the hyperparameter search.

6.3.3 INITIAL SURFACE

After PES learn had performed the hyperparameter search, the trained network had the following architecture:

- **Layers** - [32,32,32],
- **Activation function** - tanh,
- **Learning rate** - 0.8,
- **Feature scaling** - standardised to have a mean and standard deviation of 1.
- **Label scaling** - standardised to have a mean and standard deviation of 1.

Training took 510 epochs, at which point early stopping was triggered. Figure 6.10 shows the error distribution and training/validation loss for the trained network. The maximum prediction error is 611.20 kJ mol^{-1} (Figure 6.10a), and the total loss on the whole dataset (also the standard deviation of the dataset) is 38.41 kJ mol^{-1}. This is obviously far from the accuracy required of a PES for use in quantum molecular dynamics calculations (chemical accuracy is 4 kJ mol^{-1}) and Figure 6.10b shows clear overfitting to the training data, with loss values of 29.57 kJ mol^{-1} and 57.40 kJ mol^{-1} on the training and validation sets, respectively. The dataset does show good coverage of all relative energies, with only one distinct hole in the data at a relative energy of 1500–2000 kJ mol^{-1}. Figure 6.10a reveals that although high energy structures were removed from the dataset after a CASSCF calculation, further energy calculations



(a) **(b)**

Figure 6.10. **(a)** Prediction errors of the full dataset calculated as $E_{pred} - E_{calc}$, and **(b)** the $\ln(\text{Loss})$ of the model as it trained, the training set loss is in blue and the validation loss is in orange.

at the CASPT2 level have re-introduced high energy structures to the dataset (~2000 structures), highlighting that further data curation could have potentially been employed. However, with an already small dataset, removing more datapoints from the training set is perhaps undesirable (especially since the expensive CASPT2 calculations has already been carried out). Extracting a number of examples of the high error structures ($\geqslant \pm 250$ kJ mol^{-1}) reveals that the OH + SO dissociation limit energy is being over predicted, while the energy of the particularly strained structure OHSO is underpredicted. This is not unsurprising as the strained OHSO has a relative energy of $+1017.44$ kJ mol^{-1}, meaning similar structures were likely removed from the dataset at the CASSCF step of data generation. As a result the training set likely contains very few examples of these high energy strained structures.

Impressively, the model can evaluate 270,000 points in approximately four minutes, allowing for a simple brute force algorithm (implemented by the Python library SciPy[313]) to determine minima in the model. This algorithm will be used to find local minima throughout the surface. Figure 6.13a shows the predicted energies of the stationary points benchmarked in Section 6.2 (relative to HOSO, as before). Although this model performs poorly from a loss perspective, the qualitative nature of the of the predicted stationary points matches that as expected from Section 6.2,

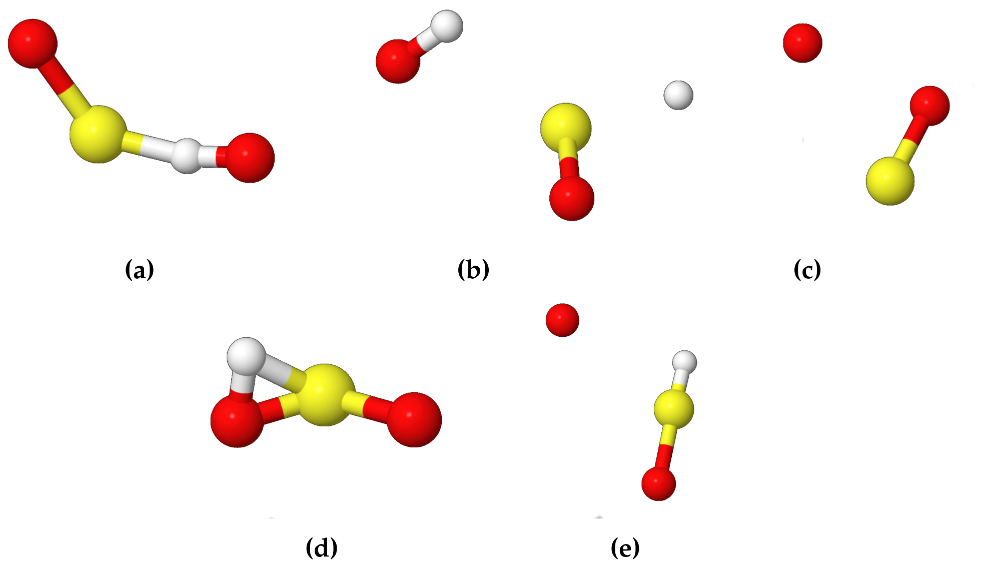
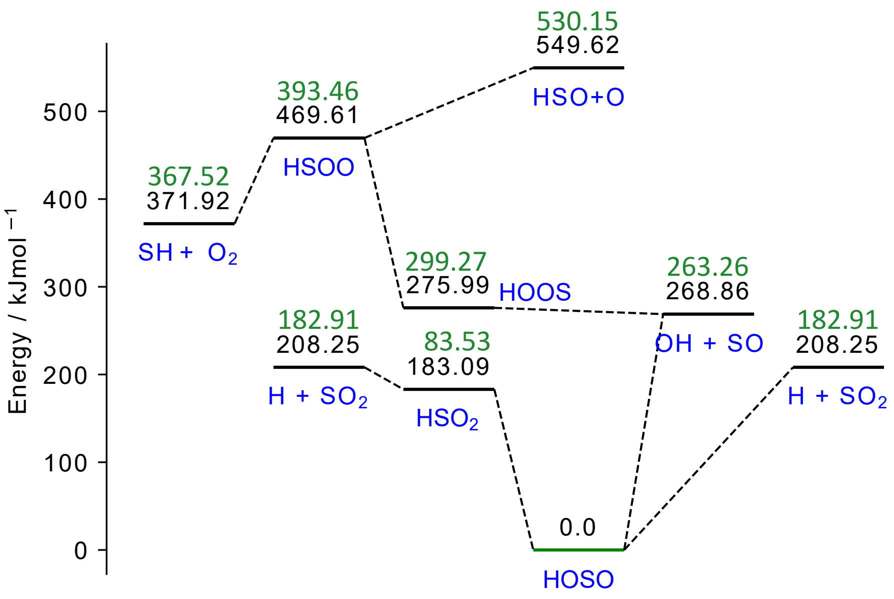


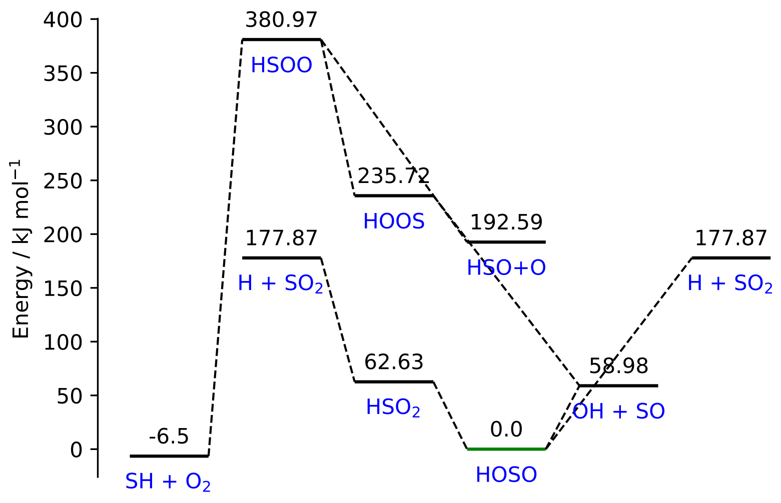
Figure 6.11. **(a)-(e)** Geometries of calculations with a prediction error of $\geqslant \pm 250$ kJ mol^{-1}

with the HOSO structure correctly being predicted as the lowest energy structure, and the average absolute error across all eight stationary points being 36.26 kJ mol^{-1}. The relative energy of HSO_2 is predicted the worst, with an error of +99.56 kJ mol^{-1}. This is because the pre-calculated stationary point was not present in the training set, and instead was present in only the test set. This could be avoided by forcing the pre-calculated stationary points into the *training* dataset, rather than just adding them dataset as a whole before the train/test split is performed. Figure 6.13b shows the stationary points predicted by the model. Here there is also moderate agreement with qualitative nature of the surface, and the network predicts that the minima around the HSO_2 structure as having a relative energy of 62.63 kJ mol^{-1}. Importantly again, HOSO is predicted as the minimum energy structure of the system.

However, this performance improvement is only true within the limits of the surface. As the molecule reaches the higher energy dissociation limits such as $SH + O_2$, $OH + SO$, and $HSO + O$ the relative energies drop, instead of reaching a plateau. It would be important for QMD to ensure that these limits are correctly identified as plateaus. Another problem



(a)



(b)

Figure 6.13. Stationary points on the HSO_2 PES. **(a)** Neural network predicted energies of benchmark stationary points , and **(b)** Minima predicted by the NN itself.

with the current surface is that it isn't completely symmetric around the dihedral, despite being seeded as such. The energy profile of the ground state HOSO structure was shown to be completely symmetric in Section 6.2, but Figure 6.12 reveals that this is not the case in the predicted surface (E at $-58°$ = -160.97065623 Hartree, E at $58°$ = -160.97367159 Hartree). There are four major problems with the dihedral scan: the location of the

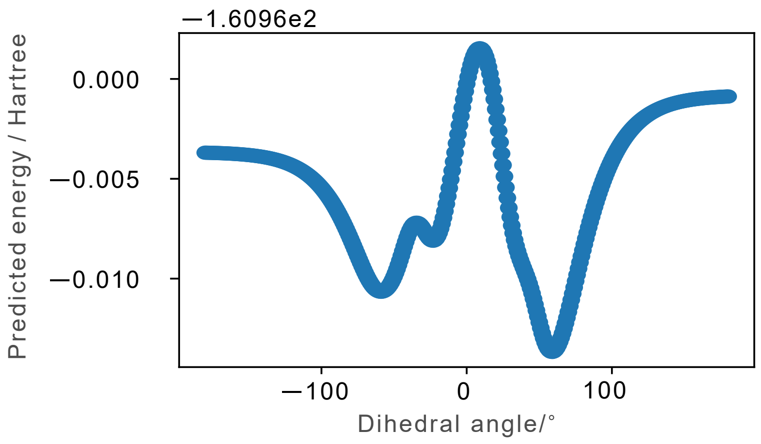


Figure 6.12. A dihedral angle scan of HOSO using the NN fit PES. The predicted ground state equilibrium geometry was used as a base, and the dihedral angle was varied from -180 ° to $180°$ while the rest of the molecule remained frozen.

minima is symmetric in terms of the angle it appears at, but predicted energies are different, there is a local minima around $-20°$ which is not present in the calculated dihedral scan, the maxima at $0°$ is much too high, and the limits are not equal in energy.

It is clear that there is room for improvement in the predicted surface, and this model was subsequently used as a starting point for the firefly algorithm developed in Chapter 4

6.3.4 THE APPLICATION OF THE FIREFLY ALGORITHM

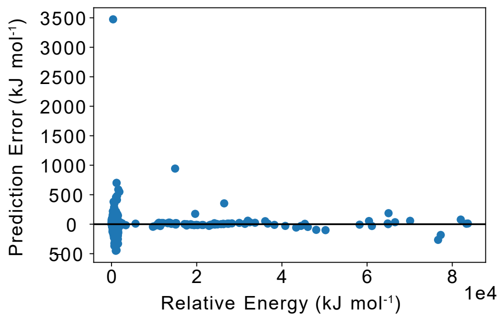
Starting with the trained model, 100 fireflies were initialised into a single group with minimum limits of [1.2, 1.2, 0.9, 0, 0, -180] and maximum limits of [4.2, 4.6, 3.0, 130, 130, 0] for $[r_{SO}, r_{SO(H)}, r_{OH}, \theta, \phi, \omega]$, respectively. These limits were chosen based on the minimum and maximum values of each degree of freedom in the benchmarking data. The value of γ was set to 3 and $\alpha = 0.2$. To assess the performance of the network while keeping computational costs low, the algorithm was run for eight generations and the neural network was retrained, updating the objective function, after 4 generations (defining one cycle = four generations).

Cycle one. After four generations of the FA, PES learn[135] was used to retrain the neural network using the new dataset (keeping all the same options for PES Learn). Training took 640 iterations before early stopping was triggered, and the hyperparameters after training were:

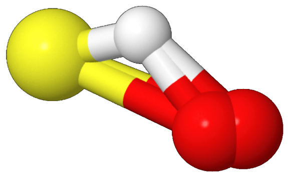
- **Layers** - [64,64],
- **Activation function** - tanh,
- **Learning rate** - 1.0,
- **Feature scaling** - standardised to have a mean and standard deviation of 1.
- **Label scaling** - standardised to have a mean and standard deviation of 1.

Figure 6.14a shows the prediction errors across the whole dataset and immediately reveals that the firefly algorithm has successfully clustered the fireflies, but they have clustered around a very high energy structure (pictured in Figure 6.14b) with a relative energy of $> 80,000$ kJ mol^{-1}. In fact, all of the datapoints with a relative energy of $> 20,000$ kJ mol^{-1} have a very similar structure. In the initial model, the predicted energy of this structure is -418426.45 kJ mol^{-1}, while the calculated value during the firefly algorithm is -338964.31 kJ mol^{-1}, giving a prediction error of -79462.14 kJ mol^{-1} (it is a little surprising that this calculation managed to converge). An analysis of the points added to the dataset by the firefly algorithm reveals that almost all of the points identified by the fireflies are these high energy structures, and there were very few datapoints added to the useful region of the PES beyond the first generation of fireflies (the random initialisation of positions). This analysis also revealed that only 129 new datapoints were added to the dataset, out of a possible 400. This is due to a large number of the CASPT2/auc-cc-pV(T+d)Z-ccECP calculations failing to converge, thus not adding any new data to the model. As well as the very high relative energy energy datapoints, Figure 6.14a also shows a singe very high prediction error datapoint. This is in the validation set, is the SH + O_2 structure seeded by the benchmarking, and it is hard to see why the model predicts this geometry so badly.

On a positive note, the fireflies have successfully clustered around a high error point, but it is unfortunately not a particularly useful one. This highlights an important improvement that needs to be made to the firefly algorithm. It is imperative that a 'sanity' check is added to the firefly positions to ensure that only valid chemical structures are added to the dataset.



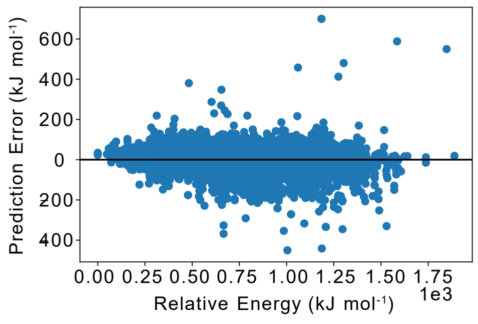
(a)



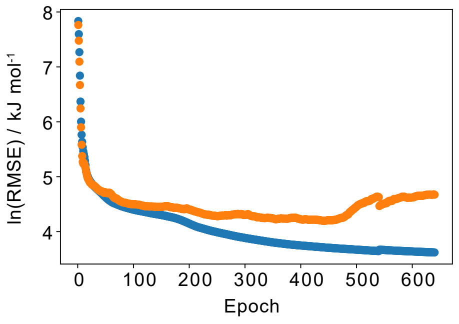
(b)

Figure 6.14. **(a)** Prediction errors of the full dataset calculated as $E_{pred} - E_{calc}$, and **(b)** the average geometry of the high energy datapoints added to the dataset by the firefly algorithm.

For completeness, Figure 6.15 shows the loss graph of the trained network, as well as a slice of the error plot, showing datapoints with relative energies of $< 2,000$ kJ mol^{-1} and predictions errors of $< 2,000$ kJ mol^{-1}. The loss graph (Figure 6.15b) looks very similar to the original models loss graph, with the same overfitting up until 439 epochs. At which point the validation error begins to increase drastically while the training error has mostly plateaued. The final values for the train, validation, and test set losses are 37.34 kJ mol^{-1}, 107.26 kJ mol^{-1}, and 66.51 kJ mol^{-1}, respectively, with a full dataset RMSE of 55.11 kJ mol^{-1}. The unusually high validation set loss is due to the large prediction error of the OH + SO datapoint. It seems that early stopping has not triggered, and this highlights another area for improvement. Although PES Learn is a very helpful tool, allowing for very fast data generation and model training, it is partially a 'black-box', and there is little room to develop the machine learning model itself. Chapter 2 highlighted that the choice of model, and the architecture of said model, can be just as important as the data provided for a machine learning problem. For example, as the model trained is a neural network, applying dropout[90] could solve some of the overfitting problems that have plagued this surface.



(a)



(b)

Figure 6.15. **(a)** Prediction errors of datapoints with relative energies of $< 2,000$ kJ mol^{-1} and predictions errors of $< 2,000$ kJ mol^{-1}, calculated as $E_{pred} - E_{calc}$, and **(b)** $\ln(\text{Loss})$ of the model as it trained, the training set loss is in blue and the validation loss is in orange.

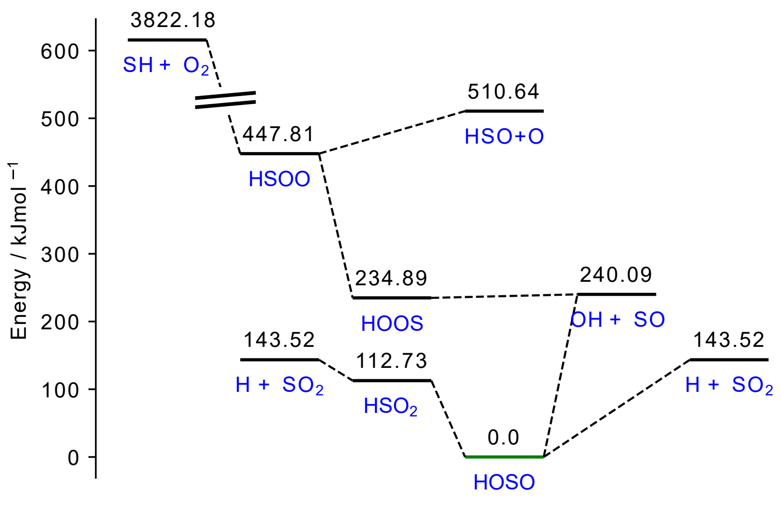
If the model had stopped training after 439 epochs the training set loss would be 41.0 kJ mol^{-1}, and the validation loss, 66.51 kJ mol^{-1} (it can be seen that the training set loss did not change much with further training cycles, another argument for the training to have stopped). However, if these values are compared to the original surface RSME values (29.57 kJ

mol^{-1} and 57.40 kJ mol^{-1}, training and validation loss, respectively), it is clear that the model actually performs worse after having had more data included in its dataset. This is likely due to all of the data being added in these extremely high energy regions, leading to the network sacrificing accuracy in the low energy regions to compensate for the new datapoints. This is reinforced by comparing the RMSE of the datapoints with relative energies < 2000 kJ mol^{-1} (mostly the just the original dataset), with the RMSE of the total dataset in the original model: 41.48 kJ mol^{-1} vs 38.41 kJ mol^{-1}.

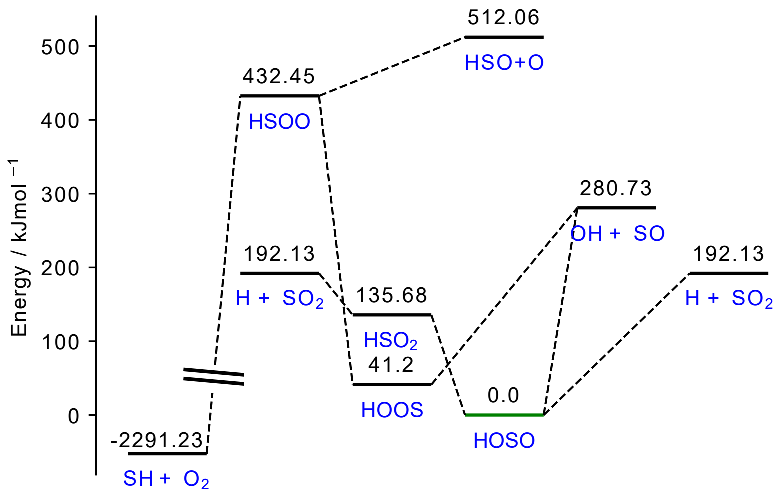
Figure 6.16 shows both the predicted relative energies of the benchmark stationary points, and the predicted relative energies of the minima identified by the model. For the benchmarked stationary points the average error has slightly risen to 38.33 kJ mol^{-1} (excluding the prediction error of 3454.66 kJ mol^{-1} for SH + O_2), showing much the same performance as the initial model. However the relative energy of HOOS has dropped to 234.89 kJ mol^{-1}, now lower in energy than the OH + SO limit at 240.09 kJ mol^{-1}. The benchmarked stationary points are predicted better for the lower energy regions than those that lie on the high energy region. The drop in relative energy for HOOS within the stationary points predicted by the model itself, continues, dropping below the relative energy of HSO_2 (41 kJ mol^{-1} vs 135.68 kJ mol^{-1}, respectively), further reinforcing the idea that the lower energy region is better fit. Finally, as was seen in the original surface, the dissociation limits continue to massively drop in energy, instead of plateauing.

Cycle two. Although cycle one shows some serious problems, the algorithm was run for eight generations, resulting in two cycles, and two newly trained models. For cycle two the same training options were used, and after the hyperparameter search the network had the following architecture:

- **Layers** - [16,16,16],
- **Activation function** - tanh,



(a)



(b)

Figure 6.16. Stationary points on the HSO_2 PES. **(a)** The neural network predicted energies of the structures used in the benchmarking of of methods and basis sets, and **(b)** the relative energies of the minimum energy structures predicted by the neural network, for each of the structures in the benchmarking set.

- **Learning rate** - 0.6,
- **Feature scaling** - standardised to have a mean and standard deviation of 1.
- **Label scaling** - standardised to have a mean and standard deviation of 1.

Figure 6.17 shows both the error graph and the loss graph for this model, and it is immediately obvious that the model performs worse than cycle one. From Figure 6.17a it is clear that the firefly algorithm has again

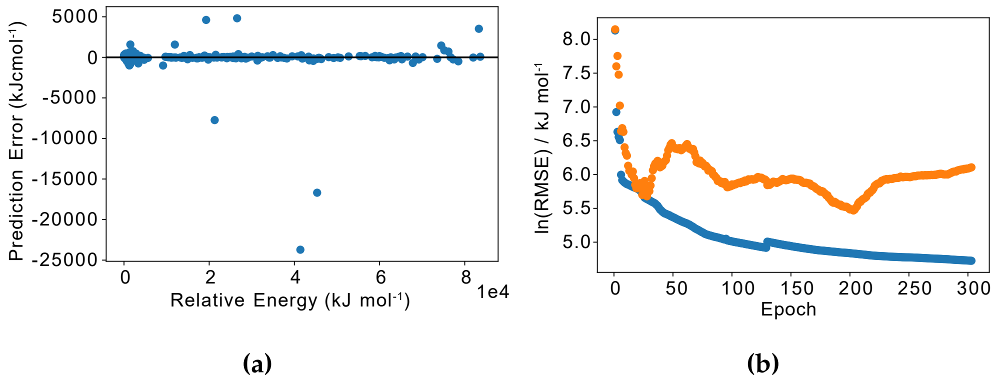


Figure 6.17. **(a)** Prediction errors of datapoints with relative energies of $< 2,000$ kJ mol^{-1} and predictions errors of $< 2,000$ kJ mol^{-1}, calculated as $E_{pred} - E_{calc}$, and **(b)** $\ln(\text{Loss})$ of the model as it trained, the training set loss is in blue and the validation loss is in orange.

clustered around extremely high energy structures, interestingly finding the same geometry as before, even after the firefly positions had been reset. The validation loss also oscillates wildly, never stabilising, and eventually diverging from the training loss as the model gets more overfit. By calculating the loss on the datapoints with relative energies of < 2000 kJ mol^{-1} (the original dataset, like for cycle one), it can be seen that the model now performs much worse than the original PES, with an RMSE just under three times higher (92.93 kJ mol^{-1}). There is obviously improvement to be had here, as the firefly algorithm has actively reduced the accuracy of the PES.

6.4 CONCLUSIONS

The HSO_2 system has been benchmarked for both the CASSCF and CASPT2 methods, using the aug-cc-pV*n*Z, aug-cc-pV(*n*+d)Z, and aug-cc-pV(*n*+d)Z-ccECP basis sets. It was shown that the aug-cc-pV(*n*+d)Z-ccECP basis sets developed in Chapter 5 successfully recreate the qualitative nature of the stationary points in the HSO_2 system, with the average deviation across all points being 1.67 kJ mol^{-1}, 2.70 kJ mol^{-1}, and 1.32 kJ mol^{-1} for the DZ, TZ, and QZ basis sets, respectively. It was also shown that generally the lower region of the surface is better recreated than the upper, higher energy, region, with the average deviation of the upper region being 2.35 kJ mol^{-1}, while the lower region is 1.00 kJ mol^{-1} at the QZ level. It was also shown that both not using the so called 'plus-d' basis sets, or a method capable of recovering dynamic correlation, such as CASPT2, leads to significantly less accurate potential energy surfaces when compared to CASPT2/aVTZ+d calculations. In fact, the use of CASSCF over CASPT2 led to a slight re-ordering of the stationary points, with HSOO dropping lower in energy than SH + O_2 (271.95 kJ mol^{-1} vs 273.97 kJ mol^{-1} respectively). This benchmarking lead to the conclusion that neither the PES developed by Varandas *et al.* (DMBE-PES)[41] or the NN fitted potential energy surfaces of Qin *et al.* (PES-2019 and PES-2020)[306,307] use a sufficient method/basis pair, and there is space for a new CASPT2/aVTZ-ccECP *global* PES for HSO_2.

An initial dataset of ~13,000 CASPT2/aVTZ-ccECP calculations was generated, and used as the dataset in the training of a neural network through PES Learn. Analysis of this initial surface revealed moderate performance, though considering how few datapoints it was trained on, the performance was higher than expected. The qualitative nature of stationary points was mostly recovered, however the total RMSE across the whole dataset was 38.41 kJ mol^{-1}, much higher than the desired 'chemical accuracy' of 4 kJ mol^{-1}.

The firefly algorithm developed in Chapter 4 was then applied to the model for two cycles, or eight generations. The network trained at the end of cycle one was analysed and revealed that the fireflies had successfully clustered around a point with very high error (-79462.14 kJ mol^{-1}), however this structure was not chemically valid (containing overlapping oxygen atoms) and had an incredibly high relative energy compared to the ground state HOSO ($> 80,000$ kJ mol^{-1}). This highlighted the need for a sanity check surrounding the data added to the model through the firefly algorithm. As the majority of the fireflies clustered around this high energy structure there was minimal change in the RMSE of the surface for the data surrounding the benchmarked stationary points. The RMSE for the initial surface was 38.41 kJ mol^{-1}, while the surface after cycle one of the firefly algorithm had an RMSE of 41.48 kJ mol^{-1}. This shows that the model now performs worse having had more data added to the dataset, which is obviously undesirable. Finally, the model trained at the end of cycle two is briefly analysed, revealing many of the same conclusions as cycle one. The firefly algorithm successfully clusters the fireflies, but they are outside the useful bounds of the PES, and thus do not improve the surface quality at all. In fact, in the case of the second FA cycle the surface accuracy is actively reduced by the addition of extra data to the dataset (92.93 kJ mol^{-1} verses 38.41 kJ mol^{-1}).

It is clear that the firefly algorithm can be improved, and Chapter 7 will highlight some ways in which the FA could be modified to avoid some of the issues highlighted in this chapter.

7 | Conclusions & future work

In this thesis I investigated three ways of reducing the computational cost of generating potential energy surfaces through modern machine learning techniques and *ab initio* quantum chemistry. Chapter 1 introduced the concept of potential energy surfaces and highlighted the great computational cost of calculating all energies within a configuration space, and how even direct dynamics is prohibitively expensive for anything other than small molecules. Interpolation between a set of predefined geometries and their energies is proposed as an alternative route to PES fitting and a number of interpolation methods were introduced. Ultimately however, they were found to be very time-consuming, required a lot of *ab initio* calculation in their evaluation, or demanded prior, intimate knowledge about the system and its functional forms (particularly DMBE). At this point, three core questions were asked, highlighting areas for improvement, and formed the basis for the rest of the thesis:

1. Can the fitting process be simplified or streamlined?
2. Can the volume of required *ab initio* data be reduced?
3. Can the data generation be made computationally cheaper?

To answer question one, I looked towards machine learning and the methods within as a way of making the fitting process more 'black-box'.

Chapter 2 explored the field of machine learning, first introducing core concepts such as the loss function, optimisation methods, and hyperparameters, then highlighting the success of many ML models within chemistry and the wider STEM community. Finally the focus narrowed in on the applications of machine learning to potential energy fitting, importantly showing that it was possible to fit a PES using a neural network, and achieve the levels of accuracy needed for vibrational analysis.[131] This chapter also highlighted the importance of good quality training data, through a discussion on the impact of machine learning on the COVID-19 pandemic.[137–139]

To address question number three above, it is important to have a good understanding of electronic structure theory, and the methods and limitations within. Working through the derivation of the energy of the Schrödinger equation in Chapter 3 ultimately led to a discussion on the importance of static and dynamic correlation and the ways to recover them. While an introduction to basis sets and the representation of the wavefunction gave rise to a method of removing core electrons from a system to avoid treating them explicitly (in the form of effective core potentials), opening an avenue towards reducing the computational cost of *ab initio* calculations. Core polarisation potentials were also introduced as another way of approximating important interactions within atoms.

The use of a machine learning algorithm to fit a PES already addresses question number two to an extent, but a concept called active learning pushes the point even further. Chapter 4 introduced the concept of active learning as well as highlighting a number of nature inspired optimisation algorithms. Within this discussion, the firefly algorithm was specifically called out, and chosen to be adapted into a novel active learning technique specifically for machine learned potential energy surfaces. Subsequently, a firefly algorithm, specifically formulated to work with a potential energy surface generated by PES Learn[135] was developed in Python and tested on a very simple system: water.

A dataset of 1331 datapoints was generated at the CCSD(T)/aug-cc-pVTZ level of theory and used as the training data for a neural network. The RMSE of the fitted surface was 0.13 kJ mol^{-1}, which is impressive for the little effort it took to generate data and train the model, and almost certainly lends credence to machine learning being able to simplify the surface fitting process. However, this could perhaps be considered a large error when compared to a surface generated for use in *ab initio* spectroscopy. Mizus *et al.*[223] report fitting errors as low as 0.00013 kJ mol^{-1} for their H_2O surface, albeit for a higher level of theory and with relativistic effects taken into account.

In order to test the firefly algorithm, another model was trained, this time on 300 points of data randomly sampled from the full set. This was expectedly less accurate, having a full dataset RMSE of 1.66 kJ mol^{-1} which allowed room for improvement. To have something to compare to, first another 300 points of data were randomly sampled from the remaining dataset, and a model was trained on the 600 datapoints. Then, the firefly algorithm was run for two cycles (eight generations). The first cycle added 160 datapoints to the dataset and improved the model fit from 1.66 kJ mol^{-1} to 0.63 kJ mol^{-1}, compared to the extra 300 points of randomly sampled data, the firefly algorithm had an almost identical performance (RMSE for the 600 point surface was 0.61 kJ mol^{-1}). This was already promising, and after cycle two the network had improved slightly to an error of 0.57 kJ mol^{-1}. Analysis of the firefly positions revealed that by generation four, all of the fireflies had clustered together and did not explore the surface much in further generations. Therefore, the firefly positions were re-set and the algorithm run one more time. After resetting their positions, the fireflies clustered in new regions of the 460 point surface, and the final RMSE on the 620 point surface was 0.24 kJ mol^{-1}, almost reaching the accuracy of the 1331 point surface with half as much data.

An important aspect of the firefly algorithm are the hyperparameters within that can be adjusted to alter the behaviour. In this case the values of γ and α were chosen through trial and error so that the fireflies clustered in about four generations. This decision was mostly arbitrary and was only possible due to the short computation times of water. For a more complicated system, trial and error will not be possible, and it would be worth assessing if these values of γ and α are transferable, or if they need to be determined system to system. Training a number of different complexities of system with various values of γ and α would hopefully give a good idea as to weather or not they need to be system specific.

These results were promising, but it is important to note that this is a very simple PES, and water is not a particularly complex molecule. It was important to test the FA on a more complicated PES landscape, and in a situation where the *ab initio* data is not so easy to generate. A molecule that meets these criteria is HSO_2, having a history of discussion surrounding the exact shape and form of its PES. However, calculations on HSO_2 are significantly more expensive (it has 33 electrons compared to water's 10), and its high energy structures contain a lot of multireference character, requiring a method capable of recovering dynamic correlation (CASPT2 in this case). As a result, question three from above became pertinent to think about.

Chapter 3 introduced an option for reducing the computational cost of a calculation, through the use of an ECP. Recently, Bennet *et al.*[52,169] developed a new generation of correlation consistent ECPs (ccECPs) for second row atoms (sulfur etc.) for use with the correlation consistent family of basis sets. They provided basis sets to use with these ccECPs, however benchmarking revealed that the provided basis sets do not perform particularly well, performing especially badly for calculations on anions. Therefore, in Chapter 5 I developed a new set of correlation consistent basis sets for use with the ccECPs, using methods consistent with previous work in the field, for the atoms Al–Ar [(aug)-cc-pV(*n*+d)Z-ccECP].[53]

One of the contributing factors to the poor performance of the original ccECP basis sets was the lack of tight-d correlating functions that have been shown to be of vital importance to second row atoms;[232–235]this will become important to remember as the HSO_2 system is explored in Chapter 6.

These new basis sets perform very well, with mean average deviations of dissociation energy, from the all-electron equivalent basis sets, of -2.30, $+2.01$ and -1.13 kJ mol^{-1}, at the DZ, TZ, and QZ levels, respectively. The performance on bond length is a little worse, with ECPs leading to shorter bond lengths on average (-0.0030 Å at the DZ level,

−0.0102 Å at the TZ level, and −0.0054 Å at the QZ level). However, the error in their benchmarking is made up for in the 15% cost reduction seen in the timing benchmarks for these basis sets. Although 15% is not a large time saving, over the many thousands of calculations required for a PES, this 15% adds up.

Core polarisation potentials are relatively easy to pair with ECPs, therefore I also parameterised a set of CPPs for the second row atoms Al–Ar. In combination with the ccECPs these CPPs can accurately recreate the effects of very large and expensive cc-pCV*n*Z calculations. They also lead to very large time savings when compared to doing a full core-valence calculation (taking 4% of the time at the DZ level, 2% at the TZ level and 1% at QZ level). Crucially, cc-pV(T+d)Z-ccECP/CPP takes less than half the CPU time of the cc-pCVDZ calculation, and cc-pV(Q+d)Z-ccECP/CPP is almost an order of magnitude faster than the lower zeta-level cc-pCVTZ.

Having now developed a way of reducing the computational cost for a sulfur containing molecule through the use of a ccECP and its associated basis set the HSO_2 system was explored in depth. The system was benchmarked for both the CASSCF and CASPT2 methods, using the aug-cc-pV*n*Z, aug-cc-pV(*n*+d)Z, and aug-cc-pV(*n*+d)Z-ccECP basis sets, showing that the basis sets developed in Chapter 5 successfully recreate the qualitative nature of the stationary points in the HSO_2 system,

At the time of writing the only *global* potential energy surface for HSO_2 was the double many body expansion surface described in Chapter 1, developed by Varandas *et al.* This surface was fit using CASSCF/aug-cc-pVTZ data and neither uses a tight-d basis set nor a method capable of recovering dynamic correlation. It was improved by Garrido *et al.* using CASPT2/aVTZ+d calculations, but the underlying surface is still too low of a level of theory. More recently, and of particular interest to this thesis, Qin *et al.* fit a CCSD(T)-F12a/aug-cc-pVTZ PES for the low energy region of the surface using a neural network, and representing the geometries as permutational invariant polynomials. Concerningly for such a modern work, this surface does not use the plus-d basis sets. Qin *et al.* further attempted to improve their surface using UCCSD(T)-F12a/aug-cc-pVTZ calculations to address issues surrounding the high energy OH + SO limit due to its multi-reference character, however they still fail to use the plus-d basis sets, and UCCSD(T)-F12a is not a multi-reference method, so one wonders if any performance gains seen were merely coincidence. Therefore it is clear that a new, *global*, CASPT2/aug-cc-pV(T+d)Z-ccECP could reveal new information about the dynamics of this system.

A set of 13,000 CASPT2/cc-pCV(T+d)Z-ccECP calculations was successfully generated for this dataset, however this was the result of artificially doubling the data through the symmetry around the dihedral angle. Only 6815 of the original 50,000 generated geometries managed to converge and return an energy at the CASPT2/cc-pCV(T+d)Z-ccECP level. This is due to the way the data was generated (selecting 10 points equally along each degree of freedom). Had the original dataset curation kept molecular geometries within the regions of the reactive parts of the PES, then one would expect that more of the calculations would be successful. This would also simply aid training, by having more data in the regions of the surface where the interesting chemistry happens. There was an attempt to perform a little bit of data curation, by first performing a CASSCF calculation, and only continuing to the more expensive CASPT2 calculation if the first succeeded. But is is clear from the error graphs in Chapter 6 that after the high energy CASSCF calculations were removed, a number of the resulting CASPT2 calculations have similarly large relative energies. The problem with needing a more directed data generation step is that it makes some knowledge of the surface/system necessary before a PES is able to be fit, which reduces the transferability of the method, and contradicts the idea that ML PESs are useful because they are more hands off.

Using PES Learn [135] Before applying the firefly algorithm, a NN was fit to the data, and analysed. Qualitatively the model performs reasonably well, recreating the order of stationary points from the benchmarking section, but the RMSE on the whole dataset is 38 kJ mol^{-1}, far from a desired accuracy.† This is where the use of PES Learn as a tool has some downsides. Although the the training of a neural network through PES Learn is very easy, there are currently limited options when it comes to dealing with overfitting. Dropout[90] has been shown to combat overfitting well, and one wonders how well the model might perform if it were applied (although it does seem possible to modify the code to include it). The loss graph in Section 6.3 also highlights a downfall of using such a program, in this case early stopping seems to have failed, as the validation set reaches a minimum RMSE and then begins to climb. The model trains for 200 more epochs before it stops. These kinds of problems are hard to troubleshoot and explain with a more commercial, black-box program. Finally the dataset should have manually been split into a training and testing set, as one of the important seeded stationary points was placed in the validation set for the first cycles model. This meant that one of the points expected to be reasonably well predicted had the largest error in the whole dataset. This would also stop the model from choosing a different train/test split every time the firefly algorithm finished a cycle, which in theory should not be a problem as the split should be representative, but with so few datapoints it can be hard to generate a representative sample.

Ultimately, developing a custom-built machine learning architecture specifically for use with a firefly algorithm could lead to more accurate models, as it would also be possible to *update* a model, rather than entirely retrain it with a whole new set of hyperparameters, every time the dataset is updated.

The firefly algorithm The performance of the firefly algorithm on the HSO_2 surface is discussed in detail in Chapter 6, but briefly, its inclusion as an active learning technique seems to have only made the model perform worse, particularly by the end of cycle two, where the full dataset

†Once this process has been improved, it would be best to assess the performance of a machine learned model through the running of a quantum molecular dynamics calculation using the model to probe the surface. It is important that the surface performs well for QMD as well as recreating the stationary points.

loss had risen to 92.93 kJ mol^{-1} from 38.41 kJ mol^{-1}. A large part of this performance drop is due to the fireflies clustering around exceedingly high energy points of the surface that correspond to chemically impossible structures. One method of combatting this would be to add a weighting to the brightness of a firefly that reduces the brightness of a firefly proportionally to its energy difference from the global ground state minimum energy. This would have the effect of punishing a firefly that is too high in energy, and keeping the fireflies within a more chemically interesting region.

It would be useful to test the performance of the FA against simply training two neural networks and generating new data points in the regions where they disagree the most. The hyperparameters of the firefly algorithm (γ and α) are hard to optimise in the HSO_2 model because the algorithm requires expensive CASPT2 calculations in its execution. The time investment compared to just training two networks is significantly more, and unless the performance of the firefly algorithm is significantly higher, one wonders if the cost is worth it.

It is also possible that the algorithm would perform better if it took longer for the fireflies to cluster. In both cycles of the HSO_2 surface there seems to be only one cluster point. Too large of a step size might lead to the fireflies moving significantly in the first generation, putting them very close to the current brightest firefly. This would have the effect of almost immediately locking all fireflies into one cluster point.

Conclusion Returning to the three core questions that were asked at the start this chapter: PES Learn, and machine learning in general, seems to successfully streamline the fitting of a potential energy surface. Although hard to test, the model prediction errors were likely due to the initial dataset quality, rather than a downfall of the ML. The new basis sets, although not returning a lot of time, show that the *ab initio* data generation certainly can be made cheaper without a loss of computational accuracy. Finally, Chapter 4 suggests that the firefly algorithm has the potential to succeed as an active learning technique. However, in light of the performance on the HSO_2 surface, in its current state more work is needed to improve its performance before it can be said that the use of a firefly algorithm resulted in a model accurately trained on a minimum volume of data.

If an accurate global PES could be generated for HSO_2 the next steps would be to run some quantum molecular dynamics simulations on the system to determine reaction pathways and to compare predicted reaction observables to experiment. If successful, an attempt to carry out the whole process – data generation to the application of the firefly algorithm – on a new, less well known system would reveal how transferable this method is, and whether or not it has the potential to reveal meaningful insight into the chemistry of important molecules.

A | Python code for the firefly algorithm

A.1 The Firefly class

```python
class Firefly:
    def __init__(self, ffnumber):
        self.group = None
        self.generation = 0
        self.ffnumber = ffnumber
        self.limits = None
        self.position = None
        self.pred_e = 0
        self.true_e = 0

        self.alpha = 0.2
        self.beta0 = 1


        self.gamma = 3

    # ID
    @property
    def id(self):
        return f'{self.group}-{self.generation}-{self.ffnumber}'
```

```python
def generate_calc_string(self, template):
    calc_str = template.format(*self.position)
    return calc_str

# BRIGHTNESS
@property
def brightness(self):
    return abs(self.pred_e-self.true_e)

# POSITION UPDATE
def update_position(self, brighter_ff_position):
    starting_position = self.position.copy()

    normed_position = (np.array(self.position)
                         - np.array(self.limits['min_limits'])) \
                      / (np.array(self.limits['max_limits'])
                         - np.array(self.limits['min_limits']))

    normed_brighter = (np.array(brighter_ff_position)
                         - np.array(self.limits['min_limits'])) \
                      / (np.array(self.limits['max_limits'])
                         - np.array(self.limits['min_limits']))

    # r = sqrt[(x_i-x_j)^2+(y_i-y_j)^2+(z_i-z_j)^2]
    distance = np.sum(np.square(normed_position - normed_brighter))
    beta = self.beta0 * np.exp(-self.gamma * distance)
    randomness = self.alpha*(random.uniform(0, 1)-0.5)
    print(f'r: {distance}, B: {beta}, aE: {randomness}')

    normed_position += (beta * (normed_brighter-normed_position))
                           + randomness

    self.position = (
        normed_position * (np.array(self.limits['max_limits'])
            - np.array(self.limits['min_limits']))
```

```
                + np.array(self.limits['min_limits']))

        # Limit position to within self.limits
        for i in range(len(self.position)):
            if self.position[i] > self.limits['max_limits'][i]:
                self.position[i] = self.limits['max_limits'][i]
            if self.position[i] < self.limits['min_limits'][i]:
                self.position[i] = self.limits['min_limits'][i]

        print("Updated ", starting_position, " to ", self.position)
        print("----------------------")

    def random_walk(self, area):
        self.position = np.array(
            [random.uniform(cord-area, cord+area)
                for cord in self.position])

        print(np.array([random.uniform(cord-area, cord+area)
                for cord in self.position]))

        for i in range(len(self.position)):
            if self.position[i] > self.limits['max_limits'][i]:
                self.position[i] = self.limits['max_limits'][i]
            if self.position[i] < self.limits['min_limits'][i]:
                self.position[i] = self.limits['min_limits'][i]
        print(f'Brightest firefly is {self.id}:{self.brightness},
                random walk to {self.position}')
        print("----------------------")
```

A.2 THE VISUALISER CLASS

```
class Visualiser:
    def __init__(self, **kwargs):
        self.generation = kwargs.get("generation", [])
        self.positions = kwargs.get("positions", [])
        self.brightnesses = kwargs.get("brightnesses", [])

    def add_firefly(self, firefly):
        self.generation.append(firefly.generation)
        self.positions.append(firefly.position)
        self.brightnesses.append(firefly.brightness)
        print("Adding to plot...")
        print("Generation" , firefly.generation)
        print("Position" , firefly.position)
        print("Brightness" , firefly.brightness)
        print("----------------------")

    @property
    def num_generations(self):
        return set(self.generation)
```

A.3 GENERAL FUNCTIONS

```
def gen_ff(i, j, limits):
    ff = Firefly(ffnumber=i)
    ff.group = j
    ff.limits = limits
    ff.position = [random.uniform(i,j) for i, j in
        zip(ff.limits['min_limits'],ff.limits['max_limits'])]
    return ff

def generate_input_file(template, ff):
    input_str = ff.generate_calc_string(template)
    filename = ff.id
    with open(f'{filename}.inp', 'w') as inp_file:
        inp_file.write(input_str)
    return filename

def read_output_file(ff,energy_line_pattern):
    filename = ff.id
    with open(f'{filename}.out', 'r') as outfile:
        lines = outfile.readlines()
    for line in lines:
        if energy_line_pattern in line:
            true_energy = float(line.strip().split(' ')[-1])
    return true_energy

from compute_energy import pes
def probe_surface(ff):
    predicted_energy = pes(ff.position, cartesian=False)
    return predicted_energy
```

A.4 THE FIREFLY ALGORITHM

```
def firefly_algorithm(num_fireflies, max_generations,
    limits, calc_template, qsub_template, energy_line_pattern,
    ml_input_string, retrain_limit=2):

    # Initialise the fireflies

    # The number of max/min limits you give the FA
    # defines the number of groups

    # group-generation-firefly is a unique ID.

    num_groups = int(len(limits['max_limits']))
    fireflies = []
    for i in range(num_groups):
        lims = [list_of_values[i] for list_of_values in limits.values()]
        group_limits = {'min_limits': lims[0], 'max_limits': lims[1]}
        for j in range(int(num_fireflies/num_groups)):
            fireflies.append(gen_ff(j, i, group_limits))

    # Initialise the visualiser
    visualiser = Visualiser()
```

```
cutoff = False
c = 0.000015 # Hartree

retrain_limit = retrain_limit
retrain_counter = 0
number_of_retrains = 0
additional_data = []

current_dataset_path = 'path/to/PES.dat'

G = 0
# if the cutoff not reached, run FA
while cutoff == False and G < max_generations:
    print(f'Current genreration: {G}')
    # Generate the input files
    print('----- GENERATING INPUT FILES -----')
    file_list = f'gen-{fireflies[0].generation}-filelist.txt'
    for f in fireflies:
        filename = generate_input_file(calc_template,f)
        with open(f'{file_list}', 'a') as list_file:
            print(f'{filename}.inp', file=list_file)

    # Generate qsub script
    print('----- GENERATING QSUB SCRIPT -----')
    with open(f'submit_generation_{fireflies[0].generation}.sh',
              'w') as queue_file:
        print(qsub_template.format(len(fireflies),file_list),
              file=queue_file)

    # Run the input files (on sol, via SGE)
    # This SHOULD wait for the qsub command to end
    print('----- RUNNING SOL SCRIPT -----')
    os.system(f"ssh $USER@$SERVER 'cd ~/path/to/directory &&
        qsub -sync y
        submit_generation_{fireflies[0].generation}.sh' && exit")

    # Read the input files and probe the neural network
    print('----- READING AND PREDICTING ENERGIES -----')
    for f in fireflies:
        f.true_e = read_output_file(f, energy_line_pattern)
        f.pred_e = probe_surface(f)

    # Add current generation of fireflies to visualiser here
    print('----- ADDING CURRENT GEN TO VISUALISER -----')
    for i in range(len(fireflies)):
        visualiser.add_firefly(fireflies[i])

    # Generate a csv of new data,
    # to be added to the training dataset later
    print('----- CREATEING CSV FROM CURRENT DATA -----')
    generation_dataset = [[*f.position,f.true_e] for f in fireflies]
    generation_dataframe = pd.DataFrame(generation_dataset)
    generation_dataframe.to_csv(
        f'generation-{fireflies[0].generation}.csv',
        index=False, header=False)

    # Update the positions WITHIN GROUPS based on brightness
    print('----- UPDATING POSITIONS -----')
    for k in range(num_groups):
        for i in fireflies:
            if i.group == k:
                for j in fireflies:
                    if j.group == k:
                        if i == j:
                            print("Same firefly, nothing to do.")
                            print("----------------------")
                        elif i.brightness < j.brightness:
                            i.update_position(j.position)
                        else:
                            print("Firefly is dimmer, not moving.")
                            print("----------------------")
    for i in fireflies:
        i.generation += 1 # THE GENERATION IS INCREMENTED IN THIS STEP

    # Random walk the brightest firefly
    print('----- RANDOM WALKING BRIGHTEST FIREFLY -----')
    current_brightest = max(fireflies, key=attrgetter('brightness'))
    current_brightest.random_walk(0.1)

    # Check brightness of all fireflies
    print('----- CHECKING BRIGHTNESS FOR CUTOFF -----')
    all_b = [f.brightness for f in fireflies]
    cutoff = all(i <= c for i in all_b) # True if all all_b < c
    if cutoff == True:
        print('Energy cutoff reached, exiting after next train')

    # Retrain the network on the new dataset,
    # if the re-train limit is reached
    print('----- CHECKING RETRAIN LIMIT -----')
    retrain_counter += 1
    print(f'Retrain counter = {retrain_counter}')

    print('----- ADDING DATA TO ADDITIONAL DATA DF -----')
    for row in generation_dataset:
        additional_data.append(row) # add new data to a growing list

    if retrain_counter == retrain_limit:
        print('----- RETRAINING ML ALGORITHM -----')
        current_dataset = pd.read_csv(current_dataset_path)
        columns = current_dataset.columns
        new_dataset = current_dataset.append(pd.DataFrame(additional_data,
                        columns=columns),
                        ignore_index = True).copy()
        new_dataset.to_csv('modified_dataset.csv', index=False)
```

```
        current_dataset_path = 'modified_dataset.csv'

        len_training_set = (0.75 * len(new_dataset))
        input_string = ml_input_string.format(
            len_training_set=int(len_training_set))

        input_obj = peslearn.InputProcessor(input_string)
        nn = peslearn.ml.NeuralNetwork(current_dataset_path, input_obj)
        nn.optimize_model()

        number_of_retrains += 1

        os.system(f'mv train_val_loss.csv
            train_val_loss_model{number_of_retrains}.csv')

        # for this to work there must be NO previous PES Learn models in
        # the current directory, at least not named 'modeln_data'

        print(f'Getting model{number_of_retrains}_data files.')
        os.system(f'cp model{number_of_retrains}_data/PES.dat .')
        os.system(f'cp model{number_of_retrains}_data/model.pt .')
        os.system(f'cp model{number_of_retrains}_data/compute_energy.py .')

        retrain_counter = 0 # reset the retrain counter
        additional_data = [] # reset additional data list

    G += 1
print('Firefly algorithm exited successfully? Nice.')
return fireflies,visualiser
```

B | Basis Sets in Molpro Format

All of the following basis sets must be used in conjunction with the ccECPs of Bennett *et al., J. Chem. Phys.* **149**, 104108 (2018).

B.1 CC-PV(D+D)Z-CCECP

```
s,Al, 9.467632E+00, 5.626780E+00, 2.011750E+00, 1.153557E+00, 1.680911E-01, 6.127412E-02
c,1.6, 2.106000E-03, -5.525000E-03, 6.608500E-02, -2.884150E-01, 6.652750E-01, 4.728490E-01
c,6.6, 1.0
p,Al, 5.086488E+00, 3.178563E+00, 1.986886E+00, 1.961017E-01, 5.681130E-02
c,1.5, -5.667000E-03, 1.819200E-02, -2.883900E-02, 4.918900E-01, 6.154460E-01
c,5.5, 1.0
d,Al, 1.308217E+00, 1.832422E-01
s,Si, 1.108664E+01, 6.688456E+00, 2.621138E+00, 1.512618E+00, 2.341472E-01, 8.553995E-02
c,1.6, 2.178000E-03, -5.499000E-03, 8.033500E-02, -3.226790E-01, 6.862160E-01, 4.647370E-01
c,6.6, 1.0
p,Si, 6.598755E+00, 4.126387E+00, 2.581579E+00, 2.886592E-01, 8.597700E-02
c,1.5, -5.591000E-03, 2.022600E-02, -3.687200E-02, 5.193070E-01, 5.869270E-01
c,5.5, 1.0
d,Si, 1.738470E+00, 2.675427E-01
s,P,1.238236E+01,7.485122E+00,3.082440E+00,1.926247E+00,3.056385E-01,1.119803E-01
c,1.6,2.195000E-03,-5.501000E-03,1.094400E-01,-3.638660E-01,6.998080E-01,4.593350E-01
c,6.6,1.0
p,P,8.332862E+00,5.213576E+00,3.265010E+00,3.917816E-01,1.186369E-01
c,1.5,-5.202000E-03,1.989200E-02,-4.200200E-02,5.369650E-01,5.687700E-01
c,5.5,1.0
d,P,2.161064E+00,3.604973E-01
s,S,1.371939E+01,8.361204E+00,3.780862E+00,2.363702E+00,3.851312E-01,1.408929E-01
c,1.6,2.159000E-03,-6.249000E-03,1.213550E-01,-3.861230E-01,7.155840E-01,4.512290E-01
c,6.6,1.0
p,S,1.075098E+01,6.729523E+00,4.218423E+00,5.045883E-01,1.473090E-01
c,1.5,-4.307000E-03,1.663900E-02,-3.974800E-02,5.441890E-01,5.668270E-01
c,5.5,1.0
d,S,2.624970E+00,4.600632E-01
s,Cl,1.468532E+01,9.081676E+00,4.517516E+00,2.823726E+00,4.726501E-01,1.727413E-01
```

```
 c,1.6,3.796000E-03,-1.095200E-02,1.279250E-01,-3.953880E-01,7.245830E-01,4.471770E-01
 c,6.6,1.0
 p,Cl,1.328317E+01,8.318690E+00,5.220721E+00,6.312944E-01,1.821076E-01
 c,1.5,-3.718000E-03,1.493300E-02,-3.997300E-02,5.513830E-01,5.617710E-01
 c,5.5,1.0
 d,Cl,3.195709E+00,5.780413E-01
 s,Ar,1.561888E+01,9.399571E+00,5.294684E+00,3.311215E+00,5.648221E-01,2.065069E-01
 c,1.6,1.705000E-03,-2.509000E-03,1.186860E-01,-4.024230E-01,7.324230E-01,4.447530E-01
 c,6.6,1.0
 p,Ar,1.648115E+01,1.031973E+01,6.477514E+00,7.679845E-01,2.212524E-01
 c,1.5,-2.576000E-03,1.244000E-02,-3.908600E-02,5.569660E-01,5.563190E-01
 c,5.5,1.0
 d,Ar,3.640558E+00,7.022656E-01
```

B.2 CC-PV(T+D)Z-CCECP

```
s,Al,8.212835E+00,4.555659E+00,2.241051E+00,1.298574E+00,6.706170E-01,2.208360E-01,
        9.829919E-02,4.356435E-02
c,1.8,2.969000E-03,-1.335600E-02,7.908900E-02,-2.317920E-01,-9.787600E-02,4.101350E-01,
        5.783300E-01,1.869940E-01
c,7.7,1.0
c,8.8,1.0
p,Al,8.744620E+00,5.193068E+00,2.982244E+00,1.387771E+00,2.586180E-01,9.758792E-02,
        3.679630E-02
c,1.7,9.110000E-04,-5.434000E-03,9.921000E-03,-3.063100E-02,2.871910E-01,5.444390E-01,
        3.013670E-01
c,6.6,1.0
c,7.7,1.0
d,Al,1.814977E+00,3.198831E-01,1.079784E-01
f,Al,2.899838E-01

s,Si,9.665766E+00,5.482739E+00,2.775639E+00,1.643690E+00,8.488039E-01,3.097429E-01,
        1.385535E-01,6.089588E-02
c,1.8,2.722000E-03,-1.138100E-02,9.174100E-02,-2.748170E-01,-9.523000E-02,4.248530E-01,
        5.844440E-01,1.850260E-01
c,7.7,1.0
c,8.8,1.0
p,Si,9.981886E+00,6.178782E+00,3.676271E+00,1.953541E+00,3.720789E-01,1.446960E-01,
        5.503535E-02
c,1.7,1.540000E-03,-7.908000E-03,1.617000E-02,-4.200400E-02,3.158970E-01,5.424010E-01,
        2.731420E-01
c,6.6,1.0
c,7.7,1.0
d,Si,2.391269E+00,4.580241E-01,1.552119E-01
f,Si,3.692667E-01
```

```
s,P,1.122502E+01,6.369909E+00,3.337544E+00,1.973366E+00,9.581721E-01,4.051031E-01,
        1.809439E-01,7.953374E-02
c,1.8,2.402000E-03,-9.789000E-03,9.596100E-02,-3.140620E-01,-8.159000E-02,4.443610E-01,
        5.855890E-01,1.805490E-01
c,7.7,1.0
c,8.8,1.0
p,P,1.181951E+01,6.805271E+00,4.141216E+00,2.587294E+00,4.999773E-01,1.978731E-01,
        7.557219E-02
c,1.7,1.329000E-03,-9.122000E-03,2.398700E-02,-5.448600E-02,3.332890E-01,5.401530E-01,
        2.575540E-01
c,6.6,1.0
c,7.7,1.0
d,P,3.026100E+00,6.204295E-01,2.107737E-01
f,P,4.771697E-01

s,S,1.271740E+01,7.211320E+00,4.108071E+00,2.297170E+00,1.100207E+00,4.758254E-01,
        2.126538E-01,9.413496E-02
c,1.8,1.735000E-03,-8.089000E-03,8.814300E-02,-3.418750E-01,-4.562200E-02,5.034010E-01,
        5.565790E-01,1.457610E-01
c,7.7,1.0
c,8.8,1.0
p,S,1.390646E+01,8.407031E+00,5.255629E+00,3.286285E+00,6.403901E-01,2.485725E-01,
        9.175393E-02
c,1.7,1.974000E-03,-1.109000E-02,2.356400E-02,-5.335700E-02,3.470450E-01,5.322760E-01,
        2.592810E-01
c,6.6,1.0
c,7.7,1.0
d,S,3.607259E+00,7.800180E-01,2.638650E-01
f,S,5.634499E-01

s,Cl,1.390166E+01,8.099804E+00,4.560530E+00,2.737296E+00,1.194114E+00,5.845004E-01,
        2.595187E-01,1.134861E-01
c,1.8,2.569000E-03,-7.665000E-03,1.012690E-01,-3.692210E-01,-4.401400E-02,5.185750E-01,
        5.588800E-01,1.388690E-01
c,7.7,1.0
c,8.8,1.0
p,Cl,1.638099E+01,1.024131E+01,6.404853E+00,4.006446E+00,7.985489E-01,3.084952E-01,
        1.121782E-01
c,1.7,2.486000E-03,-1.178300E-02,2.182600E-02,-5.337800E-02,3.565500E-01,5.285790E-01,
        2.568250E-01
c,6.6,1.0
c,7.7,1.0
d,Cl,4.382079E+00,9.824805E-01,3.297425E-01
f,Cl,7.101264E-01

s,Ar,1.489270E+01,8.737185E+00,4.922859E+00,3.075375E+00,1.497732E+00,6.819138E-01,
        3.112818E-01,1.381357E-01
c,1.8,-9.380000E-04,1.646700E-02,6.215900E-02,-3.759580E-01,-1.241600E-02,5.160160E-01,
        5.453520E-01,1.439860E-01
c,7.7,1.0
c,8.8,1.0
p,Ar,1.937247E+01,1.210980E+01,7.575727E+00,4.739667E+00,9.657799E-01,3.737167E-01,
        1.352415E-01
c,1.7,2.327000E-03,-9.304000E-03,1.331000E-02,-4.765200E-02,3.643430E-01,5.258920E-01,
        2.519590E-01
c,6.6,1.0
c,7.7,1.0
d,Ar,4.947375E+00,1.198435E+00,4.003559E-01
f,Ar,8.956061E-01
```

B.3 CC-PV(Q+D)Z-CCECP

```
s,Al,7.949914E+00,4.529029E+00,2.432129E+00,1.498216E+00,8.078694E-01,4.166727E-01,
        1.939421E-01,8.995064E-02,4.109861E-02
c,1.9,4.006000E-03,-1.946100E-02,8.612900E-02,-1.601450E-01,-1.811080E-01,5.199000E-02,
        4.505290E-01,5.295550E-01,1.531160E-01
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
p,Al,9.876031E+00,5.669519E+00,3.471873E+00,1.009537E+00,3.966336E-01,1.728821E-01,
        7.330786E-02,3.065228E-02
c,1.8,3.550000E-04,-2.650000E-03,2.661000E-03,-4.195600E-02,1.123410E-01,3.727120E-01,
        4.744470E-01,1.917390E-01
c,6.6,1.0
c,7.7,1.0
c,8.8,1.0
d,Al,1.808743E+00,4.437953E-01,1.953366E-01,7.965274E-02
f,Al,4.429014E-01,1.741183E-01
g,Al,3.991464E-01

s,Si,1.000301E+01,5.840344E+00,3.100974E+00,1.915116E+00,1.108635E+00,4.398912E-01,
        2.578032E-01,1.248379E-01,5.743548E-02
c,1.9,3.791000E-03,-1.650600E-02,8.843000E-02,-1.625180E-01,-1.976540E-01,8.836700E-02,
        4.405530E-01,5.132750E-01,1.497040E-01
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
p,Si,1.098641E+01,6.385523E+00,3.886146E+00,1.600543E+00,4.555836E-01,2.091416E-01,
        9.417660E-02,4.141521E-02
c,1.8,6.460000E-04,-4.084000E-03,6.136000E-03,-4.316700E-02,1.934680E-01,4.180540E-01,
        4.034270E-01,1.267590E-01
c,6.6,1.0
```

```
c,7.7,1.0
c,8.8,1.0
d,Si,2.496123E+00,6.300131E-01,2.729389E-01,1.126100E-01
f,Si,5.618072E-01,2.150865E-01
g,Si,4.981687E-01

s,P,1.144594E+01,6.702436E+00,3.643314E+00,2.230657E+00,1.278947E+00,4.990009E-01,
        2.941025E-01,1.542838E-01,7.338672E-02
c,1.9,3.636000E-03,-1.631400E-02,1.016420E-01,-2.236860E-01,-1.711240E-01,1.998080E-01,
        4.185340E-01,4.581190E-01,1.328930E-01
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
p,P,1.238219E+01,7.179106E+00,4.380560E+00,2.051358E+00,6.334161E-01,3.006345E-01,
        1.361396E-01,5.897863E-02
c,1.8,6.610000E-04,-4.157000E-03,5.708000E-03,-5.113500E-02,1.859600E-01,4.147800E-01,
        4.097020E-01,1.329160E-01
c,6.6,1.0
c,7.7,1.0
c,8.8,1.0
d,P,3.273202E+00,8.687763E-01,3.732263E-01,1.540900E-01
f,P,7.321927E-01,2.818867E-01
g,P,6.303479E-01

s,S,1.304422E+01,7.527028E+00,4.217795E+00,2.548020E+00,1.440352E+00,6.091360E-01,
        3.388140E-01,1.790031E-01,8.751556E-02
c,1.9,2.456000E-03,-1.244500E-02,1.062420E-01,-2.807790E-01,-1.395360E-01,2.534480E-01,
        4.626390E-01,4.018840E-01,1.061460E-01
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
p,S,1.371753E+01,8.019559E+00,4.884424E+00,2.754758E+00,7.559207E-01,3.478529E-01,
        1.550598E-01,6.663373E-02
c,1.8,1.109000E-03,-6.752000E-03,1.034900E-02,-5.156400E-02,2.331750E-01,4.287970E-01,
        3.736230E-01,1.132400E-01
c,6.6,1.0
c,7.7,1.0
c,8.8,1.0
d,S,3.776298E+00,1.071908E+00,4.629455E-01,1.902392E-01
f,S,8.914082E-01,3.265297E-01
g,S,7.092373E-01

s,Cl,1.407461E+01,8.237384E+00,4.734507E+00,2.858994E+00,1.509973E+00,6.958753E-01,
        3.820075E-01,2.194034E-01,1.095167E-01
c,1.9,2.929000E-03,-1.088200E-02,1.051570E-01,-3.325820E-01,-9.824000E-02,3.335220E-01,
        4.255470E-01,3.602330E-01,1.135250E-01
c,7.7,1.0
```

```
c,8.8,1.0
c,9.9,1.0
p,Cl,1.475695E+01,8.897799E+00,5.560378E+00,3.474530E+00,9.358483E-01,4.292111E-01,
        1.895415E-01,8.042236E-02
c,1.8,1.752000E-03,-1.041900E-02,1.703400E-02,-5.707600E-02,2.443960E-01,4.313190E-01,
        3.658380E-01,1.107200E-01
c,6.6,1.0
c,7.7,1.0
c,8.8,1.0
d,Cl,4.531908E+00,1.347215E+00,5.809417E-01,2.378593E-01
f,Cl,1.121156E+00,4.158898E-01
g,Cl,8.543884E-01

s,Ar,1.499224E+01,8.819630E+00,5.336511E+00,3.329262E+00,1.474522E+00,8.804411E-01,
        4.500511E-01,2.669738E-01,1.336571E-01
c,1.9,5.600000E-04,4.352000E-03,9.420400E-02,-3.560670E-01,-1.407840E-01,3.887130E-01,
        4.359850E-01,3.452360E-01,1.224840E-01
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
p,Ar,1.646851E+01,1.026891E+01,6.415053E+00,4.009512E+00,1.132865E+00,5.202681E-01,
        2.293103E-01,9.671475E-02
c,1.8,2.358000E-03,-1.051600E-02,9.491000E-03,-5.125700E-02,2.507430E-01,4.332380E-01,
        3.605570E-01,1.087320E-01
c,6.6,1.0
c,7.7,1.0
c,8.8,1.0
d,Ar,4.977088E+00,1.628626E+00,7.053144E-01,2.878835E-01
f,Ar,1.408786E+00,5.409325E-01
g,Ar,1.023726E+00
```

B.4 AUG-CC-PV(D+D)Z-CCECP

```
s,Al,9.467632E+00,5.626780E+00,2.011750E+00,1.153557E+00,1.680911E-01,6.127412E-02,
        2.088086E-02
c,1.6,2.106000E-03,-5.525000E-03,6.608500E-02,-2.884150E-01,6.652750E-01,4.728490E-01
c,6.6,1.0
c,7.7,1.0
p,Al,5.086488E+00,3.178563E+00,1.986886E+00,1.961017E-01,5.681130E-02,1.477338E-02
c,1.5,-5.667000E-03,1.819200E-02,-2.883900E-02,4.918900E-01,6.154460E-01
c,5.5,1.0
c,6.6,1.0
d,Al,1.308217E+00,1.832422E-01,5.296992E-02

s,Si,1.108664E+01,6.688456E+00,2.621138E+00,1.512618E+00,2.341472E-01,8.553995E-02,
        3.051215E-02
```

```
c,1.6,2.178000E-03,-5.499000E-03,8.033500E-02,-3.226790E-01,6.862160E-01,4.647370E-01
c,6.6,1.0
c,7.7,1.0
p,Si,6.598755E+00,4.126387E+00,2.581579E+00,2.886592E-01,8.597700E-02,2.459099E-02
c,1.5,-5.591000E-03,2.022600E-02,-3.687200E-02,5.193070E-01,5.869270E-01
c,5.5,1.0
c,6.6,1.0
d,Si,1.738470E+00,2.675427E-01,8.004663E-02

s,P,1.238236E+01,7.485122E+00,3.082440E+00,1.926247E+00,3.056385E-01,1.119803E-01,
        3.879193E-02
c,1.6,2.195000E-03,-5.501000E-03,1.094400E-01,-3.638660E-01,6.998080E-01,4.593350E-01
c,6.6,1.0
c,7.7,1.0
p,P,8.332862E+00,5.213576E+00,3.265010E+00,3.917816E-01,1.186369E-01,3.398252E-02
c,1.5,-5.202000E-03,1.989200E-02,-4.200200E-02,5.369650E-01,5.687700E-01
c,5.5,1.0
c,6.6,1.0
d,P,2.161064E+00,3.604973E-01,1.116762E-01

s,S,1.371939E+01,8.361204E+00,3.780862E+00,2.363702E+00,3.851312E-01,1.408929E-01,
        4.709953E-02
c,1.6,2.159000E-03,-6.249000E-03,1.213550E-01,-3.861230E-01,7.155840E-01,4.512290E-01
c,6.6,1.0
c,7.7,1.0
p,S,1.075098E+01,6.729523E+00,4.218423E+00,5.045883E-01,1.473090E-01,4.041793E-02
c,1.5,-4.307000E-03,1.663900E-02,-3.974800E-02,5.441890E-01,5.668270E-01
c,5.5,1.0
c,6.6,1.0
d,S,2.624970E+00,4.600632E-01,1.449989E-01

s,Cl,1.468532E+01,9.081676E+00,4.517516E+00,2.823726E+00,4.726501E-01,1.727413E-01,
        5.666529E-02
c,1.6,3.796000E-03,-1.095200E-02,1.279250E-01,-3.953880E-01,7.245830E-01,4.471770E-01
c,6.6,1.0
c,7.7,1.0
p,Cl,1.328317E+01,8.318690E+00,5.220721E+00,6.312944E-01,1.821076E-01,4.898700E-02
c,1.5,-3.718000E-03,1.493300E-02,-3.997300E-02,5.513830E-01,5.617710E-01
c,5.5,1.0
c,6.6,1.0
d,Cl,3.195709E+00,5.780413E-01,1.901646E-01

s,Ar,1.561888E+01,9.399571E+00,5.294684E+00,3.311215E+00,5.648221E-01,2.065069E-01,
        6.623105E-02
c,1.6,1.705000E-03,-2.509000E-03,1.186860E-01,-4.024230E-01,7.324230E-01,4.447530E-01
c,6.6,1.0
c,7.7,1.0
```

```
p,Ar,1.648115E+01,1.031973E+01,6.477514E+00,7.679845E-01,2.212524E-01,5.755607E-02
c,1.5,-2.576000E-03,1.244000E-02,-3.908600E-02,5.569660E-01,5.563190E-01
c,5.5,1.0
c,6.6,1.0
d,Ar,3.640558E+00,7.022656E-01,2.353302E-01
```

B.5 AUG-CC-PV(T+D)Z-CCECP

```
s,Al,8.212835E+00,4.555659E+00,2.241051E+00,1.298574E+00,6.706170E-01,2.208360E-01,
        9.829919E-02,4.356435E-02,1.670994E-02
c,1.8,2.969000E-03,-1.335600E-02,7.908900E-02,-2.317920E-01,-9.787600E-02,4.101350E-01,
        5.783300E-01,1.869940E-01
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
p,Al,8.744620E+00,5.193068E+00,2.982244E+00,1.387771E+00,2.586180E-01,9.758792E-02,
        3.679630E-02,1.155016E-02
c,1.7,9.110000E-04,-5.434000E-03,9.921000E-03,-3.063100E-02,2.871910E-01,5.444390E-01,
        3.013670E-01
c,6.6,1.0
c,7.7,1.0
c,8.8,1.0
d,Al,1.814977E+00,3.198831E-01,1.079784E-01,3.499576E-02
f,Al,2.899838E-01,9.958119E-02

s,Si,9.665766E+00,5.482739E+00,2.775639E+00,1.643690E+00,8.488039E-01,3.097429E-01,
        1.385535E-01,6.089588E-02,2.487397E-02
c,1.8,2.722000E-03,-1.138100E-02,9.174100E-02,-2.748170E-01,-9.523000E-02,4.248530E-01,
        5.844440E-01,1.850260E-01
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
p,Si,9.981886E+00,6.178782E+00,3.676271E+00,1.953541E+00,3.720789E-01,1.446960E-01,
        5.503535E-02,1.962433E-02
c,1.7,1.540000E-03,-7.908000E-03,1.617000E-02,-4.200400E-02,3.158970E-01,5.424010E-01,
        2.731420E-01
c,6.6,1.0
c,7.7,1.0
c,8.8,1.0
d,Si,2.391269E+00,4.580241E-01,1.552119E-01,5.475022E-02
f,Si,3.692667E-01,1.367250E-01

s,P,1.122502E+01,6.369909E+00,3.337544E+00,1.973366E+00,9.581721E-01,4.051031E-01,
        1.809439E-01,7.953374E-02,3.594531E-02
c,1.8,2.402000E-03,-9.789000E-03,9.596100E-02,-3.140620E-01,-8.159000E-02,4.443610E-01,
        5.855890E-01,1.805490E-01
```

```
c,7.7,1.0
c,8.8,1.0
p,P,1.181951E+01,6.805271E+00,4.141216E+00,2.587294E+00,4.999773E-01,1.978731E-01,
        7.557219E-02,2.629990E-02
c,1.7,1.329000E-03,-9.122000E-03,2.398700E-02,-5.448600E-02,3.332890E-01,5.401530E-01,
        2.575540E-01
c,6.6,1.0
c,7.7,1.0
c,8.8,1.0
d,P,3.026100E+00,6.204295E-01,2.107737E-01,7.798817E-02
f,P,4.771697E-01,1.716984E-01

s,S,1.271740E+01,7.211320E+00,4.108071E+00,2.297170E+00,1.100207E+00,4.758254E-01,
        2.126538E-01,9.413496E-02,3.685113E-02
c,1.8,1.735000E-03,-8.089000E-03,8.814300E-02,-3.418750E-01,-4.562200E-02,5.034010E-01,
        5.565790E-01,1.457610E-01
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
p,S,1.390646E+01,8.407031E+00,5.255629E+00,3.286285E+00,6.403901E-01,2.485725E-01,
        9.175393E-02,3.191429E-02
c,1.7,1.974000E-03,-1.109000E-02,2.356400E-02,-5.335700E-02,3.470450E-01,5.322760E-01,
        2.592810E-01
c,6.6,1.0
c,7.7,1.0
c,8.8,1.0
d,S,3.607259E+00,7.800180E-01,2.638650E-01,1.045166E-01
f,S,5.634499E-01,2.180021E-01

s,Cl,1.390166E+01,8.099804E+00,4.560530E+00,2.737296E+00,1.194114E+00,5.845004E-01,
        2.595187E-01,1.134861E-01,4.329746E-02
c,1.8,2.569000E-03,-7.665000E-03,1.012690E-01,-3.692210E-01,-4.401400E-02,5.185750E-01,
        5.588800E-01,1.388690E-01
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
p,Cl,1.638099E+01,1.024131E+01,6.404853E+00,4.006446E+00,7.985489E-01,3.084952E-01,
        1.121782E-01,3.891738E-02
c,1.7,2.486000E-03,-1.178300E-02,2.182600E-02,-5.337800E-02,3.565500E-01,5.285790E-01,
        2.568250E-01
c,6.6,1.0
c,7.7,1.0
c,8.8,1.0
d,Cl,4.382079E+00,9.824805E-01,3.297425E-01,1.371921E-01
f,Cl,7.101264E-01,3.135908E-01

s,Ar,1.489270E+01,8.737185E+00,4.922859E+00,3.075375E+00,1.497732E+00,6.819138E-01,
        3.112818E-01,1.381357E-01,4.594268E-02
c,1.8,-9.380000E-04,1.646700E-02,6.215900E-02,-3.759580E-01,-1.241600E-02,5.160160E-01,
        5.453520E-01,1.439860E-01
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
p,Ar,1.937247E+01,1.210980E+01,7.575727E+00,4.739667E+00,9.657799E-01,3.737167E-01,
        1.352415E-01,4.591860E-02
c,1.7,2.327000E-03,-9.304000E-03,1.331000E-02,-4.765200E-02,3.643430E-01,5.258920E-01,
        2.519590E-01
c,6.6,1.0
c,7.7,1.0
c,8.8,1.0
d,Ar,4.947375E+00,1.198435E+00,4.003559E-01,1.698676E-01
f,Ar,8.956061E-01,4.091794E-01
```

B.6 AUG-CC-PV(Q+D)Z-CCECP

```
s,Al,7.949914E+00,4.529029E+00,2.432129E+00,1.498216E+00,8.078694E-01,4.166727E-01,
        1.939421E-01,8.995064E-02,4.109861E-02,1.578828E-02
c,1.9,4.006000E-03,-1.946100E-02,8.612900E-02,-1.601450E-01,-1.811080E-01,5.199000E-02,
        4.505290E-01,5.295550E-01,1.531160E-01
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
c,10.10,1.0
p,Al,9.876031E+00,5.669519E+00,3.471873E+00,1.009537E+00,3.966336E-01,1.728821E-01,
        7.330786E-02,3.065228E-02,1.017416E-02
c,1.8,3.550000E-04,-2.650000E-03,2.661000E-03,-4.195600E-02,1.123410E-01,3.727120E-01,
        4.744470E-01,1.917390E-01
c,6.6,1.0
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
d,Al,1.808743E+00,4.437953E-01,1.953366E-01,7.965274E-02,2.922625E-02
f,Al,4.429014E-01,1.741183E-01,6.441994E-02
g,Al,3.991464E-01,1.661437E-01

s,Si,1.000301E+01,5.840344E+00,3.100974E+00,1.915116E+00,1.108635E+00,4.398912E-01,
        2.578032E-01,1.248379E-01,5.743548E-02,2.365888E-02
c,1.9,3.791000E-03,-1.650600E-02,8.843000E-02,-1.625180E-01,-1.976540E-01,8.836700E-02,
        4.405530E-01,5.132750E-01,1.497040E-01
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
c,10.10,1.0
```

```
p,Si,1.098641E+01,6.385523E+00,3.886146E+00,1.600543E+00,4.555836E-01,2.091416E-01,
        9.417660E-02,4.141521E-02,1.635956E-02
c,1.8,6.460000E-04,-4.084000E-03,6.136000E-03,-4.316700E-02,1.934680E-01,4.180540E-01,
        4.034270E-01,1.267590E-01
c,6.6,1.0
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
d,Si,2.496123E+00,6.300131E-01,2.729389E-01,1.126100E-01,4.818720E-02
f,Si,5.618072E-01,2.150865E-01,8.743436E-02
g,Si,4.981687E-01,2.263078E-01

s,P,1.144594E+01,6.702436E+00,3.643314E+00,2.230657E+00,1.278947E+00,4.990009E-01,
        2.941025E-01,1.542838E-01,7.338672E-02,3.293226E-02
c,1.9,3.636000E-03,-1.631400E-02,1.016420E-01,-2.236860E-01,-1.711240E-01,1.998080E-01,
        4.185340E-01,4.581190E-01,1.328930E-01
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
c,10.10,1.0
p,P,1.238219E+01,7.179106E+00,4.380560E+00,2.051358E+00,6.334161E-01,3.006345E-01,
        1.361396E-01,5.897863E-02,2.262816E-02
c,1.8,6.610000E-04,-4.157000E-03,5.708000E-03,-5.113500E-02,1.859600E-01,4.147800E-01,
        4.097020E-01,1.329160E-01
c,6.6,1.0
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
d,P,3.273202E+00,8.687763E-01,3.732263E-01,1.540900E-01,6.591598E-02
f,P,7.321927E-01,2.818867E-01,1.080602E-01
g,P,6.303479E-01,2.562073E-01

s,S,1.304422E+01,7.527028E+00,4.217795E+00,2.548020E+00,1.440352E+00,6.091360E-01,
        3.388140E-01,1.790031E-01,8.751556E-02,3.551585E-02
c,1.9,2.456000E-03,-1.244500E-02,1.062420E-01,-2.807790E-01,-1.395360E-01,2.534480E-01,
        4.626390E-01,4.018840E-01,1.061460E-01
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
c,10.10,1.0
p,S,1.371753E+01,8.019559E+00,4.884424E+00,2.754758E+00,7.559207E-01,3.478529E-01,
        1.550598E-01,6.663373E-02,2.636426E-02
c,1.8,1.109000E-03,-6.752000E-03,1.034900E-02,-5.156400E-02,2.331750E-01,4.287970E-01,
        3.736230E-01,1.132400E-01
c,6.6,1.0
c,7.7,1.0
c,8.8,1.0
```

```
c,9.9,1.0
d,S,3.776298E+00,1.071908E+00,4.629455E-01,1.902392E-01,8.418609E-02
f,S,8.914082E-01,3.265297E-01,1.370203E-01
g,S,7.092373E-01,3.057057E-01

s,Cl,1.407461E+01,8.237384E+00,4.734507E+00,2.858994E+00,1.509973E+00,6.958753E-01,
        3.820075E-01,2.194034E-01,1.095167E-01,4.335384E-02
c,1.9,2.929000E-03,-1.088200E-02,1.051570E-01,-3.325820E-01,-9.824000E-02,3.335220E-01,
        4.255470E-01,3.602330E-01,1.135250E-01
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
c,10.10,1.0
p,Cl,1.475695E+01,8.897799E+00,5.560378E+00,3.474530E+00,9.358483E-01,4.292111E-01,
        1.895415E-01,8.042236E-02,3.189509E-02
c,1.8,1.752000E-03,-1.041900E-02,1.703400E-02,-5.707600E-02,2.443960E-01,4.313190E-01,
        3.658380E-01,1.107200E-01
c,6.6,1.0
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
d,Cl,4.531908E+00,1.347215E+00,5.809417E-01,2.378593E-01,1.063254E-01
f,Cl,1.121156E+00,4.158898E-01,2.159381E-01
g,Cl,8.543884E-01,3.787343E-01

s,Ar,1.499224E+01,8.819630E+00,5.336511E+00,3.329262E+00,1.474522E+00,8.804411E-01,
        4.500511E-01,2.669738E-01,1.336571E-01,5.071317E-02
c,1.9,5.600000E-04,4.352000E-03,9.420400E-02,-3.560670E-01,-1.407840E-01,3.887130E-01,
        4.359850E-01,3.452360E-01,1.224840E-01
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
c,10.10,1.0
p,Ar,1.646851E+01,1.026891E+01,6.415053E+00,4.009512E+00,1.132865E+00,5.202681E-01,
        2.293103E-01,9.671475E-02,3.742817E-02
c,1.8,2.358000E-03,-1.051600E-02,9.491000E-03,-5.125700E-02,2.507430E-01,4.332380E-01,
        3.605570E-01,1.087320E-01
c,6.6,1.0
c,7.7,1.0
c,8.8,1.0
c,9.9,1.0
d,Ar,4.977088E+00,1.628626E+00,7.053144E-01,2.878835E-01,1.284648E-01
f,Ar,1.408786E+00,5.409325E-01,2.948560E-01
g,Ar,1.023726E+00,4.517628E-01
```

Bibliography

[1] A. Kietzmann, B. Holst, R. Redmer, M. P. Desjarlais and T. R. Mattsson, *Phys. Rev. Lett.*, 2007, **98**, 190602.

[2] R. Q. Hood, L. H. Yang and J. A. Moriarty, *Phys. Rev. B*, 2008, **78**, 024116.

[3] C. L. Zhou, Y. G. Ma, D. Q. Fang and G. Q. Zhang, *Phys. Rev. C*, 2013, **88**, 024604.

[4] M. Yang, J. Zou, G. Wang and S. Li, *J. Phys. Chem. A*, 2017, **121**, 1351–1361.

[5] H. Kwon, B. D. Etz, M. J. Montgomery, R. Messerly, S. Shabnam, S. Vyas, A. C. T. van Duin, C. S. McEnally, L. D. Pfefferle, S. Kim and Y. Xuan, *J. Phys. Chem. A*, 2020, **124**, 4290–4304.

[6] Y. Sumiya, Y. Harabuchi, Y. Nagata and S. Maeda, *JACS Au*, 2022, **2**, 1181–1188.

[7] T. Matsubara, *J. Phys. Chem. A*, 2023, **127**, 4801–4814.

[8] M. Born and R. Oppenheimer, *Annalen der Physik*, 1927, **389**, 457–484.

[9] H. B. Schlegel, *Acc. Chem. Res.*, 2021, **54**, 3749–3759.

[10] D. G. Truhlar, Rozeanne. Steckler and M. S. Gordon, *Chem. Rev.*, 1987, **87**, 217–236.

[11] B. Kuhn, T. R. Rizzo, D. Luckhaus, M. Quack and M. A. Suhm, *J. Chem. Phys.*, 1999, **111**, 2565–2587.

[12] R. J. L. Roy and R. D. E. Henderson, *Mol. Phys.*, 2007, **105**, 663–677.

[13] R. Marquardt, K. Sagui, W. Klopper and M. Quack, *J. Phys. Chem. B*, 2005, **109**, 8439–8451.

[14] R. Marquardt, K. Sagui, J. Zheng, W. Thiel, D. Luckhaus, S. Yurchenko, F. Mariotti and M. Quack, *J. Phys. Chem. A*, 2013, **117**, 7502–7522.

[15] C. A. Hall and W. W. Meyer, *Journal of Approximation Theory*, 1976, **16**, 105–122.

[16] G. Wolberg, *Columbia University Computer Science Technical Reports*, 1988, **CUCS**, 389–88.

[17] C. Maes, *Runge's Phenomenon*, http://demonstrations.wolfram.com/RungesPhenom 2007.

[18] D. R. McLaughlin and D. L. Thompson, *J. Chem. Phys.*, 1973, **59**, 4393–4405.

[19] N. Sathyamurthy and L. M. Raff, *J. Chem. Phys.*, 1975, **63**, 464–473.

[20] J. M. Bowman, J. S. Bittman and L. B. Harding, *J. Chem. Phys.*, 1986, **85**, 911–921.

[21] M. Patrício, J. Santos, F. Patrício and A. Varandas, *J. of Math. Chem.*, 2013, **51**, 1729–1746.

[22] Y.-S. M. Wu, A. Kuppermann and J. B. Anderson, *Phys. Chem. Chem. Phys.*, 1999, **1**, 929–937.

[23] D. Shepard, in Proceedings of the 1968 23rd ACM National Conference, New York, NY, USA, 1968, pp. 517–524.

[24] M. A. Collins, *Theor Chem Acc*, 2002, **108**, 313–324.

[25] G. E. Moyano and M. A. Collins, *J. Chem. Phys.*, 2004, **121**, 9769–9775.

[26] M. A. Collins and D. H. Zhang, *J. Chem. Phys.*, 1999, **111**, 9924–9931.

[27] M. A. Collins, S. Petrie, A. J. Chalk and L. Radom, *J. Chem. Phys.*, 2000, **112**, 6625–6634.

[28] O. Tishchenko and D. G. Truhlar, *J. Chem. Phys.*, 2010, **132**, 084109.

[29] T. Hollebeek, T.-S. Ho and H. Rabitz, *Annu. Rev. Phys. Chem.*, 1999, **50**, 537–570.

[30] O. T. Unke and M. Meuwly, *J. Chem. Inf. Model.*, 2017, **57**, 1923–1931.

[31] B. Schölkopf, R. Herbrich and A. J. Smola, Computational Learning Theory, Berlin, Heidelberg, 2001, pp. 416–426.

[32] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, 3rd edn., 1996.

[33] T. Hollebeek, T.-S. Ho, H. Rabitz and L. B. Harding, *J. Chem. Phys.*, 2001, **114**, 3945–3948.

[34] T.-S. Ho and H. Rabitz, *J. Chem. Phys.*, 2003, **119**, 6433–6442.

[35] Y. Guo, A. Kawano, D. L. Thompson, A. F. Wagner and M. Minkoff, *J. Chem. Phys.*, 2004, **121**, 5091–5097.

[36] T. Ishida and G. C. Schatz, *J. Comp. Chem.*, 2003, **24**, 1077–1086.

[37] Y. Guo, L. B. Harding, A. F. Wagner, M. Minkoff and D. L. Thompson, *J. Chem. Phys.*, 2007, **126**, 104105.

[38] R. Dawes, A. F. Wagner and D. L. Thompson, *J. Phys. Chem. A*, 2009, **113**, 4709–4721.

[39] A. J. C. Varandas and H. G. Yu, *Mol. Phys.*, 1997, **91**, 301–318.

[40] A. J. C. Varandas, *Journal of Molecular Structure: THEOCHEM*, 1985, **120**, 401–424.

[41] M. Y. Ballester and A. J. C. Varandas, *Phys. Chem. Chem. Phys.*, 2005, **7**, 2305.

[42] M. Ballester and A. Varandas, *Chemical Physics Letters*, 2007, **433**, 279–285.

[43] A. J. C. Varandas, *Int. J. Quantum Chem.*, 1987, **32**, 563–574.

[44] A. J. C. Varandas, *Journal of Molecular Structure: THEOCHEM*, 1988, **166**, 59–74.

[45] L. P. Viegas, M. Cernei, A. Alijah and A. J. C. Varandas, *J. Chem. Phys.*, 2003, **120**, 253–259.

[46] B. R. L. Galvão and A. J. C. Varandas, *J. Phys. Chem. A*, 2009, **113**, 14424–14430.

[47] Y. Z. Song and A. J. C. Varandas, *J Phys Chem A*, 2011, **115**, 5274–5283.

[48] A. J. C. Varandas and j. D. da Silva, *J. Chem. Soc., Faraday Trans. 2*, 1986, **82**, 593–608.

[49] A. D. Buckingham, in *Advances in Chemical Physics*, John Wiley & Sons, Ltd, 1967, pp. 107–142.

[50] J. Behler, *Angew. Chem. Int. Ed.*, 2017, **56**, 12828–12840.

[51] J. D. Garrido, M. Y. Ballester, Y. Orozco-González and S. Canuto, *J. Phys. Chem. A*, 2011, **115**, 1453–1461.

[52] M. C. Bennett, G. Wang, A. Annaberdiyev, C. A. Melton, L. Shulenburger and L. Mitas, *J. Chem. Phys.*, 2018, **149**, 104108.

[53] A. N. Hill, A. J. H. M. Meijer and J. G. Hill, *J. Phys. Chem. A*, 2022, **126**, 5853–5863.

[54] J. Mao, W. Xu, Y. Yang, J. Wang, Z. Huang and A. Yuille, *arXiv:1412.6632 [cs]*, 2014.

[55] I. Sutskever, O. Vinyals and Q. V. Le, *arXiv:1409.3215 [cs]*, 2014.

[56] OpenAI, *OpenAI Five*, https://blog.openai.com/openai-five/, 2018.

[57] OpenAI, *http://arxiv.org/abs/2303.08774*, 2023.

[58] T. M. Mitchell, *Machine Learning*, McGraw-Hill, New York, 1997.

[59] K. Pearson, *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 1901, **2**, 559–572.

[60] I. T. Jolliffe and J. Cadima, *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 2016, **374**, 20150202.

[61] T. Ichiye and M. Karplus, *Proteins: Structure, Function, and Bioinformatics*, 1991, **11**, 205–217.

[62] A. E. García, *Phys. Rev. Lett.*, 1992, **68**, 2696–2699.

[63] A. Amadei, A. B. M. Linssen and H. J. C. Berendsen, *Proteins: Structure, Function, and Bioinformatics*, 1993, **17**, 412–425.

[64] A. L. Tournier and J. C. Smith, *Phys. Rev. Lett.*, 2003, **91**, 208106.

[65] V. Spiwok, P. Lipovová and B. Králová, *J. Phys. Chem. B*, 2007, **111**, 3073–3076.

[66] B. E. Husic and F. Noé, *J. Chem. Phys.*, 2019, **151**, 054103.

[67] F. Paul, F. Noé and T. R. Weikl, *J. Phys. Chem. B*, 2018, **122**, 5649–5656.

[68] A. Glielmo, B. E. Husic, A. Rodriguez, C. Clementi, F. Noé and A. Laio, *Chem. Rev.*, 2021, **121**, 9722–9758.

[69] M. Martínez-Díaz and F. Soriguera, *Transportation Research Procedia*, 2018, **33**, 275–282.

[70] *Atlas™*, https://www.bostondynamics.com/atlas.

[71] Z. Zhou, X. Li and R. N. Zare, *ACS Cent. Sci.*, 2017, **3**, 1337–1344.

[72] J. Horwood and E. Noutahi, *ACS Omega*, 2020, **5**, 32984–32994.

[73] M. Ostaszewski, L. M. Trenkwalder, W. Masarczyk, E. Scerri and V. Dunjko, Advances in Neural Information Processing Systems, 2021, pp. 18182–18194.

[74] S. Gow, M. Niranjan, S. Kanza and J. G. Frey, *Digital Discovery*, 2022, **1**, 551–567.

[75] R. Fletcher, *Practical Methods of Optimization*, John Wiley & Sons, Ltd, 2000.

[76] I. Goodfellow, Y. Bengio and A. Courville, *Deep Learning*, The MIT Press, Cambridge, Massachusetts, 2016.

[77] A. Géron, *Hands-on Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, O'Reilly Media, Beijing ; Boston, First edition edn., 2017.

[78] T. Wang, J.-Y. Zhu, A. Torralba and A. A. Efros, *Dataset Distillation*, 2020.

[79] F. Grisoni, M. Moret, R. Lingwood and G. Schneider, *J. Chem. Inf. Model.*, 2020, **60**, 1175–1183.

[80] X. Li, Y. Xu, H. Yao and K. Lin, *J Cheminform*, 2020, **12**, 42.

[81] Y. Murakami and A. Shono, *Chemical Engineering Journal Advances*, 2022, **9**, 100219.

[82] G. E. Hinton, S. Osindero and Y. W. Teh, *Neural Comput.*, 2006, **18**, 1527–54.

[83] X. Glorot and Y. Bengio, *Proc. 13th Int. Conf. Artif. Intell. Stat.*, 2010, **9**, 249–256.

[84] A. F. Agarap, *Deep Learning Using Rectified Linear Units (ReLU)*, 2019.

[85] T. De Ryck, S. Lanthaler and S. Mishra, *Neural Networks*, 2021, **143**, 732–750.

[86] D. P. Kingma and J. Ba, *Adam: A Method for Stochastic Optimization*, 2017.

[87] N. Qian, *Neural Networks*, 1999, **12**, 145–151.

[88] J. Duchi, E. Hazan and Y. Singer, *J. Mach. Learn. Res.*, 2011, **12**, 2121–2159.

[89] G. Hinton, *Neural Networks for Machine Learning*, 2012.

[90] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, *J. Mach. Learn. Res.*, 2014, **15**, 1929–1958.

[91] P. O. Dral, A. Owens, S. N. Yurchenko and W. Thiel, *J. Chem. Phys.*, 2017, **146**, 244108.

[92] D. Pomerleau, *Ph.D. thesis*, Carnegie Mellon University, 1992.

[93] C. A. Gomez-Uribe and N. Hunt, *ACM Trans. Manage. Inf. Syst.*, 2016, **6**, 13:1–13:19.

[94] D. Capper, D. T. W. Jones, M. Sill, V. Hovestadt, D. Schrimpf, D. Sturm, C. Koelsche, F. Sahm, L. Chavez, D. E. Reuss, A. Kratz, A. K. Wefers, K. Huang, K. W. Pajtler, L. Schweizer, D. Stichel, A. Olar, N. W. Engel, K. Lindenberg, P. N. Harter, A. K. Braczynski, K. H. Plate, H. Dohmen, B. K. Garvalov, R. Coras, A. Hölsken, E. Hewer, M. Bewerunge-Hudler, M. Schick, R. Fischer, R. Beschorner, J. Schittenhelm, O. Staszewski, K. Wani, P. Varlet, M. Pages, P. Temming, D. Lohmann, F. Selt, H. Witt, T. Milde,

O. Witt, E. Aronica, F. Giangaspero, E. Rushing, W. Scheurlen, C. Geisenberger, F. J. Rodriguez, A. Becker, M. Preusser, C. Haberler, R. Bjerkvig, J. Cryan, M. Farrell, M. Deckert, J. Hench, S. Frank, J. Serrano, K. Kannan, A. Tsirigos, W. Brück, S. Hofer, S. Brehmer, M. Seiz-Rosenhagen, D. Hänggi, V. Hans, S. Rozsnoki, J. R. Hansford, P. Kohlhof, B. W. Kristensen, M. Lechner, B. Lopes, C. Mawrin, R. Ketter, A. Kulozik, Z. Khatib, F. Heppner, A. Koch, A. Jouvet, C. Keohane, H. Mühleisen, W. Mueller, U. Pohl, M. Prinz, A. Benner, M. Zapatka, N. G. Gottardo, P. H. Driever, C. M. Kramm, H. L. Müller, S. Rutkowski, K. von Hoff, M. C. Frühwald, A. Gnekow, G. Fleischhack, S. Tippelt, G. Calaminus, C.-M. Monoranu, A. Perry, C. Jones, T. S. Jacques, B. Radlwimmer, M. Gessi, T. Pietsch, J. Schramm, G. Schackert, M. Westphal, G. Reifenberger, P. Wesseling, M. Weller, V. P. Collins, I. Blümcke, M. Bendszus, J. Debus, A. Huang, N. Jabado, P. A. Northcott, W. Paulus, A. Gajjar, G. W. Robinson, M. D. Taylor, Z. Jaunmuktane, M. Ryzhova, M. Platten, A. Unterberg, W. Wick, M. A. Karajannis, M. Mittelbronn, T. Acker, C. Hartmann, K. Aldape, U. Schüller, R. Buslei, P. Lichter, M. Kool, C. Herold-Mende, D. W. Ellison, M. Hasselblatt, M. Snuderl, S. Brandner, A. Korshunov, A. von Deimling and S. M. Pfister, *Nature*, 2018, **555**, 469–474.

[95] F. Klauschen, K. R. Müller, A. Binder, M. Bockmayr, M. Hägele, P. Seegerer, S. Wienert, G. Pruneri, S. de Maria, S. Badve, S. Michiels, T. O. Nielsen, S. Adams, P. Savas, F. Symmans, S. Willis, T. Gruosso, M. Park, B. Haibe-Kains, B. Gallas, A. M. Thompson, I. Cree, C. Sotiriou, C. Solinas, M. Preusser, S. M. Hewitt, D. Rimm, G. Viale, S. Loi, S. Loibl, R. Salgado and C. Denkert, *Seminars in Cancer Biology*, 2018, **52**, 151–157.

[96] P. Jurmeister, M. Bockmayr, P. Seegerer, T. Bockmayr, D. Treue, G. Montavon, C. Vollbrecht, A. Arnold, D. Teichmann, K. Bressem, U. Schüller, M. von Laffert, K.-R. Müller, D. Capper and F. Klauschen, *Science Translational Medicine*, 2019, **11**, eaaw8513.

[97] D. Ardila, A. P. Kiraly, S. Bharadwaj, B. Choi, J. J. Reicher, L. Peng, D. Tse, M. Etemadi, W. Ye, G. Corrado, D. P. Naidich and S. Shetty, *Nat. Med.*, 2019, **25**, 954–961.

[98] A. Binder, M. Bockmayr, M. Hägele, S. Wienert, D. Heim, K. Hellweg, M. Ishii, A. Stenzinger, A. Hocke, C. Denkert, K.-R. Müller and F. Klauschen, *Nat Mach Intell*, 2021, **3**, 355–366.

[99] T. Lengauer, O. Sander, S. Sierra, A. Thielen and R. Kaiser, *Nat Biotechnol*, 2007, **25**, 1407–1410.

[100] A. W. Senior, R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, T. Green, C. Qin, A. Žídek, A. W. R. Nelson, A. Bridgland, H. Penedones, S. Petersen, K. Simonyan, S. Crossan, P. Kohli, D. T. Jones, D. Silver, K. Kavukcuoglu and D. Hassabis, *Nature*, 2020, **577**, 706–710.

[101] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, A. Bridgland, C. Meyer, S. A. A. Kohl, A. J. Ballard, A. Cowie, B. Romera-Paredes, S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. Reiman, E. Clancy, M. Zielinski, M. Steinegger, M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli and D. Hassabis, *Nature*, 2021, **596**, 583–589.

[102] M. Varadi, S. Anyango, M. Deshpande, S. Nair, C. Natassia, G. Yordanova, D. Yuan, O. Stroe, G. Wood, A. Laydon, A. Žídek, T. Green, K. Tunyasuvunakool, S. Petersen, J. Jumper, E. Clancy, R. Green, A. Vora, M. Lutfi, M. Figurnov, A. Cowie, N. Hobbs, P. Kohli, G. Kleywegt, E. Birney, D. Hassabis and S. Velankar, *Nucleic Acids Research*, 2022, **50**, D439–D444.

[103] P. Baldi, P. Sadowski and D. Whiteson, *Nat Commun*, 2014, **5**, 4308.

[104] P. Leinen, M. Esders, K. T. Schütt, C. Wagner, K.-R. Müller and F. S. Tautz, *Science Advances*, 2020, **6**, eabb6987.

[105] D.-O. Won, K.-R. Müller and S.-W. Lee, *Science Robotics*, 2020, **5**, eabb9764.

[106] E. Guizzo, *IEEE Spectrum*, 2019, **56**, 34–39.

[107] J. Zhang, U. Norinder and F. Svensson, *J. Chem. Inf. Model.*, 2021, **61**, 2648–2657.

[108] E. Komp, N. Janulaitis and S. Valleau, *Phys. Chem. Chem. Phys.*, 2022, **24**, 2692–2705.

[109] M. Liu, B. Kwon and P. K. Kang, *Sci Rep*, 2022, **12**, 5486.

[110] T. Morawietz, A. Singraber, C. Dellago and J. Behler, *Proceedings of the National Academy of Sciences*, 2016, **113**, 8368–8373.

[111] B. Cheng, E. A. Engel, J. Behler, C. Dellago and M. Ceriotti, *Proceedings of the National Academy of Sciences*, 2019, **116**, 1110–1115.

[112] A. Reinhardt and B. Cheng, *Nat Commun*, 2021, **12**, 588.

[113] J. Vamathevan, D. Clark, P. Czodrowski, I. Dunham, E. Ferran, G. Lee, B. Li, A. Madabhushi, P. Shah, M. Spitzer and S. Zhao, *Nat Rev Drug Discov*, 2019, **18**, 463–477.

[114] S. Dara, S. Dhamercherla, S. S. Jadav, C. M. Babu and M. J. Ahsan, *Artif Intell Rev*, 2022, **55**, 1947–1999.

[115] J. A. Keith, V. Vassilev-Galindo, B. Cheng, S. Chmiela, M. Gastegger, K.-R. Müller and A. Tkatchenko, *Chem. Rev.*, 2021, **121**, 9816–9872.

[116] J. Tersoff, *Phys. Rev. Lett.*, 1986, **56**, 632–635.

[117] J. F. Justo, M. Z. Bazant, E. Kaxiras, V. V. Bulatov and S. Yip, *Phys. Rev. B*, 1998, **58**, 2539–2550.

[118] A. C. T. van Duin, S. Dasgupta, F. Lorant and W. A. Goddard, *J. Phys. Chem. A*, 2001, **105**, 9396–9409.

[119] V. Babin, C. Leforestier and F. Paesani, *J. Chem. Theory Comput.*, 2013, **9**, 5395–5403.

[120] A. P. Bartók, R. Kondor and G. Csányi, *Phys. Rev. B*, 2013, **87**, 184115.

[121] T. B. Blank, S. D. Brown, A. W. Calhoun and D. J. Doren, *The Journal of Chemical Physics*, 1995, **103**, 4129–4137.

[122] D. F. R. Brown, M. N. Gibbs and D. C. Clary, *J. Chem. Phys.*, 1996, **105**, 7597–7604.

[123] J. Zupan and J. Gasteiger, *Analytica Chimica Acta*, 1991, **248**, 1–30.

[124] W. Duch and G. H. F. Diercksen, *Computer Physics Communications*, 1994, **82**, 91–103.

[125] E. Tafeit, W. Estelberger, R. Horejsi, R. Moeller, K. Oettl, K. Vrecko and G. Reibnegger, *Journal of Molecular Graphics*, 1996, **14**, 12–18.

[126] K. Tai No, B. Ha Chang, S. Yeon Kim, M. Shik Jhon and H. A. Scheraga, *Chemical Physics Letters*, 1997, **271**, 152–156.

[127] F. V. Prudente and J. Soares Neto, *Chemical Physics Letters*, 1998, **287**, 585–589.

[128] H. Gassner, M. Probst, A. Lauenstein and K. Hermansson, *J. Phys. Chem. A*, 1998, **102**, 4596–4605.

[129] S. Lorenz, A. Groß and M. Scheffler, *Chemical Physics Letters*, 2004, **395**, 210–215.

[130] T. M. Rocha Filho, Z. T. Oliveira Jr., L. a. C. Malbouisson, R. Gargano and J. J. Soares Neto, *International Journal of Quantum Chemistry*, 2003, **95**, 281–288.

[131] S. Manzhos, X. Wang, R. Dawes and T. Carrington, *J. Phys. Chem. A*, 2006, **110**, 5295–5304.

[132] S. Manzhos, M. Ihara and T. Carrington, in *Quantum Chemistry in the Age of Machine Learning*, ed. P. O. Dral, Elsevier, 2023, pp. 355–390.

[133] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Y. Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu and Xiaoqiang Zheng, *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*, 2015.

[134] F. Chollet *et al.*, *Keras*, 2015.

[135] A. S. Abbott, J. M. Turney, B. Zhang, D. G. A. Smith, D. Altarawy and H. F. I. Schaefer, *J. Chem. Theory Comput.*, 2019, **15**, 4386–4398.

[136] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai and S. Chintala.

[137] L. Wynants, B. V. Calster, G. S. Collins, R. D. Riley, G. Heinze, E. Schuit, E. Albu, B. Arshi, V. Bellou, M. M. J. Bonten, D. L. Dahly, J. A. Damen, T. P. A. Debray, V. M. T. de Jong, M. D. Vos, P. Dhiman, J. Ensor, S. Gao, M. C. Haller, M. O. Harhay, L. Henckaerts, P. Heus, J. Hoogland, M. Hudda, K. Jenniskens, M. Kammer, N. Kreuzberger, A. Lohmann, B. Levis, K. Luijken, J. Ma, G. P. Martin, D. J. McLernon, C. L. A. Navarro, J. B. Reitsma, J. C. Sergeant, C. Shi, N. Skoetz, L. J. M. Smits, K. I. E. Snell, M. Sperrin, R. Spijker, E. W. Steyerberg, T. Takada, I. Tzoulaki, S. M. J. van Kuijk, B. C. T.

van Bussel, I. C. C. van der Horst, K. Reeve, F. S. van Royen, J. Y. Verbakel, C. Wallisch, J. Wilkinson, R. Wolff, L. Hooft, K. G. M. Moons and M. van Smeden, *BMJ*, 2020, **369**, m1328.

[138] M. Roberts, D. Driggs, M. Thorpe, J. Gilbey, M. Yeung, S. Ursprung, A. I. Aviles-Rivero, C. Etmann, C. McCague, L. Beer, J. R. Weir-McCall, Z. Teng, E. Gkrania-Klotsas, J. H. F. Rudd, E. Sala and C.-B. Schönlieb, *Nat Mach Intell*, 2021, **3**, 199–217.

[139] I. von Borzyskowski, A. Mazumder, B. Mateen and M. Wooldridge, *Data Science and AI in the Age of COVID-19: Reflections on the Response of the UK's Data Science and AI Community to the COVID-19 Pandemic*, The Alan Turing Institute technical report, 2021.

[140] E. Schrödinger, *Phys. Rev.*, 1926, **28**, 1049–1070.

[141] W. Pauli, *Z. Physik*, 1925, **31**, 765–783.

[142] *4.5: Eigenfunctions of Operators Are Orthogonal*, https://chem.libretexts.org/Courses/Pacific_Union_College/Quantum_Chemistry/04%3
2020.

[143] C. Clapham and J. Nicholson, in *The Concise Oxford Dictionary of Mathematics*, Oxford University Press, 2009.

[144] P. a. M. Dirac, *Mathematical Proceedings of the Cambridge Philosophical Society*, 1939, **35**, 416–418.

[145] J. C. Slater, *Phys. Rev.*, 1929, **34**, 1293–1322.

[146] E. U. Condon, *Phys. Rev.*, 1930, **36**, 1121–1133.

[147] D. R. Hartree, *Mathematical Proceedings of the Cambridge Philosophical Society*, 1928, **24**, 89–110.

[148] V. Fock, *Z. Physik*, 1930, **61**, 126–148.

[149] C. C. J. Roothaan, *Rev. Mod. Phys.*, 1951, **23**, 69–89.

[150] C. David Sherrill and H. F. Schaefer, in *Advances in Quantum Chemistry*, ed. P.-O. Löwdin, J. R. Sabin, M. C. Zerner and E. Brändas, Academic Press, 1999, vol. 34, pp. 143–269.

[151] A. C. Wahl and G. Das, in *Methods of Electronic Structure Theory*, ed. H. F. Schaefer, Springer US, Boston, MA, 1977, pp. 51–78.

[152] P. G. Szalay, T. Müller, G. Gidofalvi, H. Lischka and R. Shepard, *Chem. Rev.*, 2012, **112**, 108–181.

[153] I. Shavitt, *International Journal of Quantum Chemistry*, 1978, **14**, 5–32.

[154] B. O. Roos, P. R. Taylor and P. E. M. Sigbahn, *Chemical Physics*, 1980, **48**, 157–173.

[155] E. R. Sayfutyarova, Q. Sun, G. K.-L. Chan and G. Knizia, *J. Chem. Theory Comput.*, 2017, **13**, 4063–4078.

[156] F. M. Fernandez, *Introduction to Perturbation Theory in Quantum Mechanics*, CRC Press, Boca Raton, 2000.

[157] B. O. Roos, P. Linse, P. E. M. Siegbahn and M. R. A. Blomberg, *Chemical Physics*, 1982, **66**, 197–207.

[158] F. Coester, *Nuclear Physics*, 1958, **7**, 421–424.

[159] R. F. Bishop and H. G. Kümmel, *Physics Today*, 1987, **40**, 52–60.

[160] J. Čížek, *Journal of Chemical Physics*, 1966, **45**, 4256–4266.

[161] R. J. Bartlett, *Annu. Rev. Phys. Chem.*, 1981, **32**, 359–401.

[162] T. H. Dunning, *J. Chem. Phys.*, 1989, **90**, 1007–1023.

[163] R. A. Kendall, T. H. Dunning and R. J. Harrison, *J. Chem. Phys.*, 1992, **96**, 6796–6806.

[164] M. Dolg and X. Cao, *Chem. Rev.*, 2012, **112**, 403–480.

[165] W. H. E. Schwarz, *Theoret. Chim. Acta*, 1968, **11**, 307–324.

[166] L. R. Kahn and W. A. Goddard, *Chemical Physics Letters*, 1968, **2**, 667–670.

[167] L. Shulenburger and T. R. Mattsson, *Phys. Rev. B*, 2013, **88**, 245117.

[168] M. C. Bennett, C. A. Melton, A. Annaberdiyev, G. Wang, L. Shulenburger and L. Mitas, *J. Chem. Phys.*, 2017, **147**, 224106.

[169] G. Wang, B. Kincaid, H. Zhou, A. Annaberdiyev, M. C. Bennett, J. T. Krogel and L. Mitas, *J. Chem. Phys.*, 2022, **157**, 054101.

[170] W. Meyer and P. Rosmus, *J. Chem. Phys.*, 1975, **63**, 2356–2375.

[171] A. Karton, E. Rabinovich, J. M. L. Martin and B. Ruscic, *J. Chem. Phys.*, 2006, **125**, 144108.

[172] D. Feller, K. A. Peterson and D. A. Dixon, *Mol. Phys.*, 2012, **110**, 2381–2399.

[173] A. Tajti, P. Szalay, A. G. Császár, M. Kállay, J. Gauss, E. F. Valeev, B. A. Flowers, J. Vázquez and J. F. Stanton, *Mol. Phys.*, 2012, **110**, 2381–2399.

[174] A. G. Császár, W. D. Allen and H. F. Schaefer, *J. Chem. Phys.*, 1998, **108**, 9751–9764.

[175] C. Bottcher and A. Dalgarno, *Proc. R. Soc. Lond. A*, 1974, **340**, 187–198.

[176] P. Fuentealba, H. Preuss, H. Stoll and L. Von Szentpály, *Chem. Phys. Lett.*, 1982, **89**, 418–422.

[177] W. Müller, J. Flesch and W. Meyer, *J. Chem. Phys.*, 1984, **80**, 3297–3310.

[178] W. Müller and W. Meyer, *J. Chem. Phys.*, 1984, **80**, 3311–3320.

[179] I. Schmidt-Mink, W. Müller and W. Meyer, *Chem. Phys.*, 1985, **92**, 263–285.

[180] B. Settles, *Active Learning Literature Survey*, University of Wisconsin-Madison Department of Computer Sciences technical Report, 2009.

[181] D. D. Lewis and W. A. Gale, Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Berlin, Heidelberg, 1994, pp. 3–12.

[182] H. S. Seung, M. Opper and H. Sompolinsky, Proceedings of the Fifth Annual Workshop on Computational Learning Theory, New York, NY, USA, 1992, pp. 287–294.

[183] X.-S. Yang, *Nature-Inspired Metaheuristic Algorithms*, Luniver Press, Frome, 2nd edn., 2010.

[184] X.-S. Yang, *Journal of Computational Science*, 2020, **46**, 101104.

[185] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*, 1992.

[186] J. Porta, J. Parapar, R. Doallo, F. F. Rivera, I. Santé and R. Crecente, *Computers, Environment and Urban Systems*, 2013, **37**, 45–58.

[187] W.-C. Hong, Y. Dong, L.-Y. Chen and S.-Y. Wei, *Applied Soft Computing*, 2011, **11**, 1881–1890.

[188] A. Hussain, Y. S. Muhammad, M. Nauman Sajid, I. Hussain, A. Mohamd Shoukry and S. Gani, *Computational Intelligence and Neuroscience*, 2017, **2017**, e7430125.

[189] M. Kaur and V. Kumar, *Mod. Phys. Lett. B*, 2018, **32**, 1850115.

[190] D. Datta, A. R. S. Amaral and J. R. Figueira, *European Journal of Operational Research*, 2011, **213**, 388–394.

[191] H.-P. Schwefel, *G. Winter, J. Perieaux, M. Gala, P. Cuesta (Eds.), Proceedings of Genetic Algorithms in Engineering and Computer Science, John Wiley & Sons*, 1995.

[192] I. Rechenberg, Optimization: Methods and Applications, Possibilities and Limitations, Berlin, Heidelberg, 1989, pp. 106–126.

[193] T. Back and H.-P. Schwefel, Proceedings of IEEE International Conference on Evolutionary Computation, 1996, pp. 20–29.

[194] H.-G. Beyer and H.-P. Schwefel, *Natural Computing*, 2002, **1**, 3–52.

[195] S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi, *Science*, 1983, **220**, 671–680.

[196] D. M, *Ph.D. Thesis, Politecnico di Milano*, 1992.

[197] M. Dorigo, V. Maniezzo and A. Colorni, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 1996, **26**, 29–41.

[198] M. Dorigo, M. Birattari and T. Stutzle, *IEEE Computational Intelligence Magazine*, 2006, **1**, 28–39.

[199] J. Kennedy and R. Eberhart, Proceedings of ICNN'95 - International Conference on Neural Networks, 1995, pp. 1942–1948 vol.4.

[200] R. Eberhart and J. Kennedy, MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, 1995, pp. 39–43.

[201] D. Freitas, L. G. Lopes and F. Morgado-Dias, *Entropy*, 2020, **22**, 362.

[202] E. Peer, F. van den Bergh and A. Engelbrecht, Proceedings of the 2003 IEEE Swarm Intelligence Symposium. SIS'03 (Cat. No.03EX706), 2003, pp. 235–242.

[203] F. van den Bergh and A. Engelbrecht, IEEE International Conference on Systems, Man and Cybernetics, 2002, pp. 6 pp. vol.3–.

[204] F. van den Bergh and A. Engelbrecht, *IEEE Transactions on Evolutionary Computation*, 2004, **8**, 225–239.

[205] Z.-H. Zhan, J. Zhang, Y. Li and H. S.-H. Chung, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 2009, **39**, 1362–1381.

[206] M. Rosendo and A. Pozo, IEEE Congress on Evolutionary Computation, 2010, pp. 1–8.

[207] M. Rosendo and A. Pozo, 2010 Eleventh Brazilian Symposium on Neural Networks, 2010, pp. 235–240.

[208] L. Junliang, H. Wei, S. Huan, L. Yaxin and L. Jing, 2017 12th IEEE Conference on Industrial Electronics and Applications (ICIEA), 2017, pp. 1860–1864.

[209] Z. W. Geem, J. H. Kim and G. Loganathan, *SIMULATION*, 2001, **76**, 60–68.

[210] X. Z. Gao, V. Govindasamy, H. Xu, X. Wang and K. Zenger, *Computational Intelligence and Neuroscience*, 2015, **2015**, e258491.

[211] M. Dubey, V. Kumar, M. Kaur and T.-P. Dao, *Mathematical Problems in Engineering*, 2021, **2021**, e5594267.

[212] S. Nakrani and C. Tovey, *Adaptive Behavior*, 2004, **12**, 223–240.

[213] D. Pham, A. Ghanbarzadeh, E. Koç, S. Otri, S. Rahim and M. Zaidi, *Manufacturing Engineering Centre, Cardiff University, UK*, 2005, 1–57.

[214] D. Karaboga, *Technical Report, Erciyes University*, 2005.

[215] X.-S. Yang, Stochastic Algorithms: Foundations and Applications, Berlin, Heidelberg, 2009, pp. 169–178.

[216] X.-S. Yang, Research and Development in Intelligent Systems XXVI, London, 2010, pp. 209–218.

[217] O. L. Polyansky, P. Jensen and J. Tennyson, *J. Chem. Phys.*, 1994, **101**, 7651–7657.

[218] O. L. Polyansky, P. Jensen and J. Tennyson, *J. Chem. Phys.*, 1996, **105**, 6490–6497.

[219] H. Partridge and D. W. Schwenke, *J. Chem. Phys.*, 1997, **106**, 4618–4639.

[220] J. Tennyson, N. F. Zobov, R. Williamson, O. L. Polyansky and P. F. Bernath, *Journal of Physical and Chemical Reference Data*, 2001, **30**, 735–831.

[221] I. I. Bubukina, N. F. Zobov, O. L. Polyansky, S. V. Shirin and S. N. Yurchenko, *Opt. Spectrosc.*, 2011, **110**, 160–166.

[222] J. Tennyson, P. F. Bernath, L. R. Brown, A. Campargue, A. G. Császár, L. Daumont, R. R. Gamache, J. T. Hodges, O. V. Naumenko, O. L. Polyansky, L. S. Rothman, A. C. Vandaele, N. F. Zobov, A. R. Al Derzi, C. Fábri, A. Z. Fazliev, T. Furtenbacher, I. E. Gordon, L. Lodi and I. I. Mizus, *Journal of Quantitative Spectroscopy and Radiative Transfer*, 2013, **117**, 29–58.

[223] I. I. Mizus, A. A. Kyuberis, N. F. Zobov, V. Y. Makhnev, O. L. Polyansky and J. Tennyson, *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 2018, **376**, 20170149.

[224] H.-J. Werner, P. J. Knowles, G. Knizia, F. R. Manby and M. Schütz, *WIREs Comput Mol Sci*, 2012, **2**, 242–253.

[225] H.-J. Werner, P. J. Knowles, G. Knizia, F. R. Manby, M. Schütz, P. Celani, W. Györffy, D. Kats, T. Korona, R. Lindh, A. Mitrushenkov, G. Rauhut, K. R. Shamasundar, T. B. Adler, R. D. Amos, S. J. Bennie, A. Bernhardsson, A. Berning, D. L. Cooper, M. J. O. Deegan, A. J. Dobbyn, F. Eckert, E. Goll, C. Hampel, A. Hesselmann, G. Hetzer, T. Hrenar, G. Jansen, C. Köppl, S. J. R. Lee, Y. Liu, A. W. Lloyd, Q. Ma, R. A. Mata, A. J. May, S. J. McNicholas, W. Meyer, T. F. Miller III, M. E. Mura, A. Nicklass, D. P.

O'Neill, P. Palmieri, D. Peng, K. Pflüger, R. Pitzer, M. Reiher, T. Shiozaki, H. Stoll, A. J. Stone, R. Tarroni, T. Thorsteinsson, M. Wang and M. Welborn, *MOLPRO, Version 2019.2, a Package of Ab Initio Programs*, 2019.

[226] J. Bergstra, D. Yamins and D. Cox, Proceedings of the 30th International Conference on Machine Learning, 2013, pp. 115–123.

[227] J. Bergstra, R. Bardenet, Y. Bengio and B. Kégl, *Advances in Neural Information Processing Systems*, 2011, **24**, 2546–2554.

[228] F. Jensen, *Wiley Interdiscip. Rev. Comput. Mol. Sci.*, 2013, **3**, 273–295.

[229] J. G. Hill, *Int. J. Quantum Chem.*, 2013, **113**, 21–34.

[230] B. Nagy and F. Jensen, in *Reviews in Computational Chemistry*, ed. A. L. Parrill and K. B. Lipkowitz, John Wiley & Sons, 2017, vol. 124, ch. 3.

[231] D. E. Woon and T. H. Dunning, *J. Chem. Phys.*, 1993, **98**, 1358–1371.

[232] C. W. Bauschlicher and H. Partridge, *Chem. Phys. Lett.*, 1995, **240**, 533–540.

[233] J. M. L. Martin, *J. Chem. Phys.*, 1998, **108**, 2791–2800.

[234] J. M. L. Martin, *J. Mol. Struc.-THEOCHEM*, 2006, **771**, 19–26.

[235] T. H. Dunning, K. A. Peterson and A. K. Wilson, *J. Chem. Phys.*, 2001, **114**, 9244–9253.

[236] R. A. Shaw and J. G. Hill, *J. Chem. Phys.*, 2017, **147**, 074108.

[237] S. C. McKenzie, E. Epifanovsky, G. M. J. Barca, A. T. B. Gilbert and P. M. W. Gill, *J. Phys. Chem. A*, 2018, **122**, 3066–3075.

[238] R. A. Shaw and J. G. Hill, *J. Open Source Softw.*, 2021, **6**, 3039.

[239] K. A. Peterson, D. Figgen, M. Dolg and H. Stoll, *J. Chem. Phys.*, 2007, **126**, 124101.

[240] D. Figgen, K. A. Peterson, M. Dolg and H. Stoll, *J. Chem. Phys.*, 2009, **130**, 164108.

[241] J. G. Hill and K. A. Peterson, *J. Chem. Phys.*, 2017, **147**, 244106.

[242] K. A. Peterson, *J. Chem. Phys.*, 2015, **142**, 074105.

[243] J. Xu, M. J. Deible, K. A. Peterson and K. D. Jordan, *J. Chem. Theory Comput.*, 2013, **9**, 2170–2178.

[244] M. Burkatzki, C. Filippi and M. Dolg, *J. Chem. Phys.*, 2007, **126**, 234105.

[245] I. Ovcharenko, A. Aspuru-Guzik and W. A. Lester, *J. Chem. Phys.*, 2001, **114**, 7790–7794.

[246] D. E. Woon and T. H. Dunning, *J. Chem. Phys.*, 1995, **103**, 4572–4585.

[247] K. A. Peterson and T. H. Dunning, *J. Chem. Phys.*, 2002, **117**, 10548–10560.

[248] H. Partridge, C. W. Bauschlicher, L. G. M. Pettersson, A. D. McLean, B. Liu, M. Yoshimine and A. Komornicki, *J. Chem. Phys.*, 1990, **92**, 5377–5383.

[249] L. G. M. Pettersson and H. Åkeby, *J. Chem. Phys.*, 1991, **94**, 2968–2976.

[250] H. Åkeby, L. G. M. Pettersson and P. E. M. Siegbahn, *J. Chem. Phys.*, 1992, **97**, 1850–1857.

[251] L. Pettersson and B. Persson, *Chem. Phys.*, 1993, **170**, 149–159.

[252] A. Nicklass and K. A. Peterson, *Theor Chem Acc*, 1998, **100**, 103–111.

[253] H.-J. Werner, P. J. Knowles, F. R. Manby, J. A. Black, K. Doll, A. Hesselmann, D. Kats, A. Köhn, T. Korona, D. A. Kreplin, Q. Ma, T. F. Miller III, A. Mitrushenkov, K. A. Peterson, I. Polyak, G. Rauhut and M. Sibaev, *J. Chem. Phys.*, 2020, **152**, 144107.

[254] R. C. Raffenetti, *J. Chem. Phys.*, 1973, **58**, 4452–4458.

[255] K. Raghavachari, G. W. Trucks, J. A. Pople and M. Head-Gordon, *Chem. Phys. Lett.*, 1989, **157**, 479–483.

[256] J. L. Dunham, *Phys. Rev.*, 1932, **41**, 721–731.

[257] J.-P. Blaudeau, S. R. Brozell, S. Matsika, Z. Zhang and R. M. Pitzer, *Int. J. Quantum Chem.*, 2000, **77**, 516–520.

[258] P. A. Christiansen, *J. Chem. Phys.*, 2000, **112**, 10070–10074.

[259] K. A. Peterson, D. Figgen, E. Goll, H. Stoll and M. Dolg, *J. Chem. Phys.*, 2003, **119**, 11113–11123.

[260] J. M. Martin and O. Uzan, *Chemical Physics Letters*, 1998, **282**, 16–24.

[261] A. Karton and J. M. L. Martin, *Theor. Chem. Acc.*, 2006, **115**, 330–333.

[262] T. Helgaker, W. Klopper, H. Koch and J. Noga, *J. Chem. Phys.*, 1997, **106**, 9639–9646.

[263] A. Halkier, T. Helgaker, P. Jørgensen, W. Klopper, H. Koch, J. Olsen and A. K. Wilson, *Chem. Phys. Lett.*, 1998, **286**, 243–252.

[264] J. G. Hill, S. Mazumder and K. A. Peterson, *J. Chem. Phys.*, 2010, **132**, 054108.

[265] W. R. Johnson, D. Kolb and K. N. Huang, *Atomic Data and Nuclear Data Tables*, 1983, **28**, 333–340.

[266] V. Kaufman and W. C. Martin, *Journal of Physical and Chemical Reference Data*, 1991, **20**, 775–858.

[267] W. C. Martin and R. Zalubas, *Journal of Physical and Chemical Reference Data*, 1983, **12**, 323–380.

[268] W. C. Martin, R. Zalubas and A. Musgrove, *Journal of Physical and Chemical Reference Data*, 1985, **14**, 751–802.

[269] W. C. Martin, R. Zalubas and A. Musgrove, *Journal of Physical and Chemical Reference Data*, 1990, **19**, 821–880.

[270] L. J. Radziemski and V. Kaufman, *J. Opt. Soc. Am.*, 1969, **59**, 424.

[271] M. Scheer, R. C. Bilodeau, J. Thøgersen and H. K. Haugen, *Phys. Rev. A*, 1998, **57**, R1493–R1496.

[272] W. Chaibi, R. J. Peláez, C. Blondel, C. Drag and C. Delsart, *Eur. Phys. J. D*, 2010, **58**, 29–37.

[273] R. J. Peláez, C. Blondel, M. Vandevraye, C. Drag and C. Delsart, *J. Phys. B: At. Mol. Opt. Phys.*, 2011, **44**, 195009.

[274] U. Berzinsh, M. Gustafsson, D. Hanstorp, A. Klinkmüller, U. Ljungblad and A.-M. Mårtensson-Pendrill, *Phys. Rev. A*, 1995, **51**, 231–238.

[275] N. Sylvetsky, M. K. Kesharwani and J. M. L. Martin, *J. Chem. Phys.*, 2017, **147**, 134106.

[276] M. Cai, T. Dzugan and V. Bondybey, *Chem. Phys. Lett.*, 1989, **155**, 430–436.

[277] Z. Fu, G. W. Lemire, G. A. Bishea and M. D. Morse, *J. Chem. Phys.*, 1990, **93**, 8420–8441.

[278] K. Huber and G. Herzberg, *Molecular Spectra and Molecular Structure: IV. Constants of Diatomic Molecules.*, Springer US, New York, 1979.

[279] P. Glarborg, *Proceedings of the Combustion Institute*, 2007, **31**, 77–98.

[280] M. U. Alzueta, R. Bilbao and P. Glarborg, *Combustion and Flame*, 2001, **127**, 2234–2251.

[281] C. A. McDowell, F. G. Herring and J. C. Tait, *J. Chem. Phys.*, 1975, **63**, 3278–3283.

[282] R. W. Fair and B. A. Thrush, *Trans. Faraday Soc.*, 1969, **65**, 1557–1570.

[283] J. L. Jourdain, G. L. Bras and J. Combourieu, *International Journal of Chemical Kinetics*, 1979, **11**, 569–577.

[284] V. R. Morris, K.-L. Han and W. M. Jackson, *J. Phys. Chem.*, 1995, **99**, 10086–10091.

[285] A. J. Frank, M. Sadílek, J. G. Ferrier and F. Tureček, *J. Am. Chem. Soc.*, 1997, **119**, 12343–12353.

[286] M. A. Blitz, K. W. McKee and M. J. Pilling, *Proceedings of the Combustion Institute*, 2000, **28**, 2491–2497.

[287] J. Ma, M. J. Wilhelm, J. M. Smith and H.-L. Dai, *Phys. Rev. A*, 2016, **93**, 040702.

[288] R. J. Boyd, A. Gupta, R. F. Langler, S. P. Lownie and J. A. Pincock, *Can. J. Chem.*, 1980, **58**, 331–338.

[289] A. Hinchliffe, *Journal of Molecular Structure*, 1981, **71**, 349–352.

[290] D. Binns and P. Marshall, *J. Chem. Phys.*, 1991, **95**, 4940–4947.

[291] D. Laakso, C. E. Smith, A. Goumri, J.-D. R. Rocha and P. Marshall, *Chemical Physics Letters*, 1994, **227**, 377–383.

[292] V. R. Morris and W. M. Jackson, *Chemical Physics Letters*, 1994, **223**, 445–451.

[293] A. Goumri, J.-D. R. Rocha and P. Marshall, *J. Phys. Chem.*, 1995, **99**, 10834–10836.

[294] J.-X. Qi, W.-Q. Deng, K.-L. Han and G.-Z. He, *J. Chem. Soc., Faraday Trans.*, 1997, **93**, 25–28.

[295] A. Goumri, J.-D. R. Rocha, D. Laakso, C. E. Smith and P. Marshall, *J. Phys. Chem. A*, 1999, **103**, 11328–11335.

[296] M. L. McKee and P. H. Wine, *J. Am. Chem. Soc.*, 2001, **123**, 2344–2353.

[297] S. M. Resende and F. R. Ornellas, *Physical Chemistry Chemical Physics*, 2003, **5**, 4617–4621.

[298] M. Y. Ballester, P. J. S. B. Caridade and A. J. C. Varandas, *Chem. Phys. Lett.*, 2007, **439**, 301–307.

[299] S. E. Wheeler and H. F. Schaefer, *J. Phys. Chem. A*, 2009, **113**, 6779–6788.

[300] C. R. Zhou, K. Sendt and B. S. Haynes, *J. Phys. Chem. A*, 2009, **113**, 2975–2981.

[301] M. Y. Ballester, Y. Orozco-Gonzalez, J. D. Garrido and H. F. Dos Santos, *J. Chem. Phys.*, 2010, **132**, 044310.

[302] G. N. Freitas, J. D. Garrido, M. Y. Ballester and M. A. C. Nascimento, *J. Phys. Chem. A*, 2012, **116**, 7677–7685.

[303] W. a. D. Pires, J. D. Garrido, M. a. C. Nascimento and M. Y. Ballester, *Phys. Chem. Chem. Phys.*, 2014, **16**, 12793–12801.

[304] D. Rodríguez-Linares, G. N. Freitas, M. Y. Ballester, M. A. C. Nascimento and J. D. Garrido, *J. Phys. Chem. A*, 2015, **119**, 8734–8743.

[305] R. S. da Silva, J. D. Garrido and M. Y. Ballester, *J. Chem. Phys.*, 2017, **147**, 084308.

[306] J. Qin, Y. Liu, D. Lu and J. Li, *J. Phys. Chem. A*, 2019, **123**, 7218–7227.

[307] J. Qin and J. Li, *Phys. Chem. Chem. Phys.*, 2021, **23**, 487–497.

[308] W. J. Pietro, M. M. Francl, W. J. Hehre, D. J. DeFrees, J. A. Pople and J. S. Binkley, *J. Am. Chem. Soc.*, 1982, **104**, 5039–5048.

[309] D. Feller and J. Sordo, *The Journal of Chemical Physics*, 2000, **113**, 485–493.

[310] B. Napolion and J. D. Watts, *Chemical Physics Letters*, 2006, **421**, 562–565.

[311] Charnley, *Rmsd*, 2019.

[312] W. Kabsch, *Acta Cryst A*, 1976, **32**, 922–923.

[313] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa and P. van Mulbregt, *Nat Methods*, 2020, **17**, 261–272.