Theoretical Methods

Introduction

During my honours project, I had used the following theoretical methods in developing my model:

- Electronic Structure Theory

- Power Series Expansion

- (Modified) Shepard Interpolation

This is a file which briefly describes these methods.

Electronic Structure Theory

Electronic Structure Theory calculations are methods used to solve the time independent Shrödinger equation:

$$\hat{\mathrm{H}}\psi = E\Psi, \tag{0.1}$$

where $\hat{\mathrm{H}}$ is the Hamiltonian operator, Ψ the wavefunction and E the total energy of the molecule being investigated.

This is done to get information on a molecule such as it's energy or dipole moment. I had used a supercomputer provided by NCI for these calculations so that I had data to build my model. There are many methods and basis sets which can be used for these calculations. High levels of theory exist for very accurate calculations however require large computation times, conversely, there are methods which are fast however are not as accurate.

For those interested, I had used the Hartree-Fock method, which is computationally cheap as it neglects explicit electron-electron correlation and is hence a mean-field theory. This method correctly determines $\sim 99\%$ of a molecules electronic energy, the remaining energy which is approximated is a result of the electron-electron repulsions in a molecule (electron correlation energy). The basis set used in my project was the 6-31+G(d,p) basis set. This is a split-valence double-zeta basis set which uses 6 Gaussian functions to make up each core orbital basis function, linear combinations of 3 and 1 Gaussians to make up the valence orbitals, has diffuse functions added for heavy atoms and polarisation functions for heavy and light atoms.

This is a relatively cheap level of theory and had been chosen for fast calculations so there would be more time to develop my model, as this was the focus of my project.

Power Series Expansion

The Power Series Expansion is a way of modelling a molecule's potential energy in an electric field:

$$V(\vec{E}) = V_0 - \left[\mu_i E_i + \frac{1}{2}\alpha_{ij}E_iE_j + \dots\right] - \left[\frac{1}{3}\Theta_{ij}E_{ij} + \frac{1}{6}C_{ij,kl}E_{ij}E_{jk} + \dots\right] - \dots, \quad (0.2)$$

Where E_{ij} are the Cartesian components of $\nabla\mathbf{E}$, and V_0, $\vec{\mu}$, α, $\mathbf{\Theta}$ and $\mathbf{C}$ are the zero-field potential energy, dipole moment, dipole polarisablity, quadrupole moment and hyperpolarisability, respectively. This equation however is only valid if converged rapidly, which is usually not the case. However, for a molecule with a large dipole (such as an amino acid), this power series converges rapidly, and molecule energies can be modelled using linear response theory:

$$V = V_0 - \mu_i E_i \quad (0.3)$$

Previously, electric field applications would be implemented in an electronic structure calculation, new calculations would need to be done all over again for different magnitudes and directions of fields which is computationally expensive. This model only requires one initial

electronic structure theory calculation and can be used to simulate the effects of any electric field on any charged molecule. All this requires is a surface of the molecule's potential energy and a dipole moment surfaces for each direction (x,y,z).

Shepard Interpolation

The Shepard interpolation method was definitely the most interesting theory method I had learned about this year, it can be used to interpolate or extrapolate off large volumes of data. This interpolation is a non-grid based method, meaning the calculations don't scale exponentially with the degrees of freedom a molecule has (opposed to something like a cubic spline interpolation). For a molecule like glycine which has 24 degrees of freedom, calculations would scale by (data points per dimension)^{24}. The Shepard interpolation is hence favoured as calculations aren't ridiculously expensive.

Why interpolate?

To be able to model the effects of any field on any charged molecule, you need an energy and dipole surface. These are constructed from an initial set of data points obtained using electronic structure theory and the rest of the data points are interpolated.

How does it work?

The Shepard Interpolation "guesses" an unknown value using a weighted sum of the surrounding data points, given by:

$$\mu_0(R) = \sum_{i=1}^{N_{data}} w_i \, \mu_i \tag{0.4}$$

Where $\mu_0(R)$ is the interpolated or "guessed" value at some configuration, R, w_i is the weight of each known data point and μ_i is the value of each known data point.

Known data points are weighted based on the inverse Euclidean distance:

$$w_i = \frac{1}{d^p} \tag{0.5}$$

Here, d is the distance between the known point and the interpolated value and p is an arbitrary scaling parameter which can be optimised.

Data which is physically closer to the value being "guessed" will hence contribute more than data which is further away. Points with very large distances will have a weighting of 0, hence no contribution. To make sure interpolations don't take up a lot of computation time, a threshold can be defined for distances where weights will be zero such that only relevant neighbour points are considered.

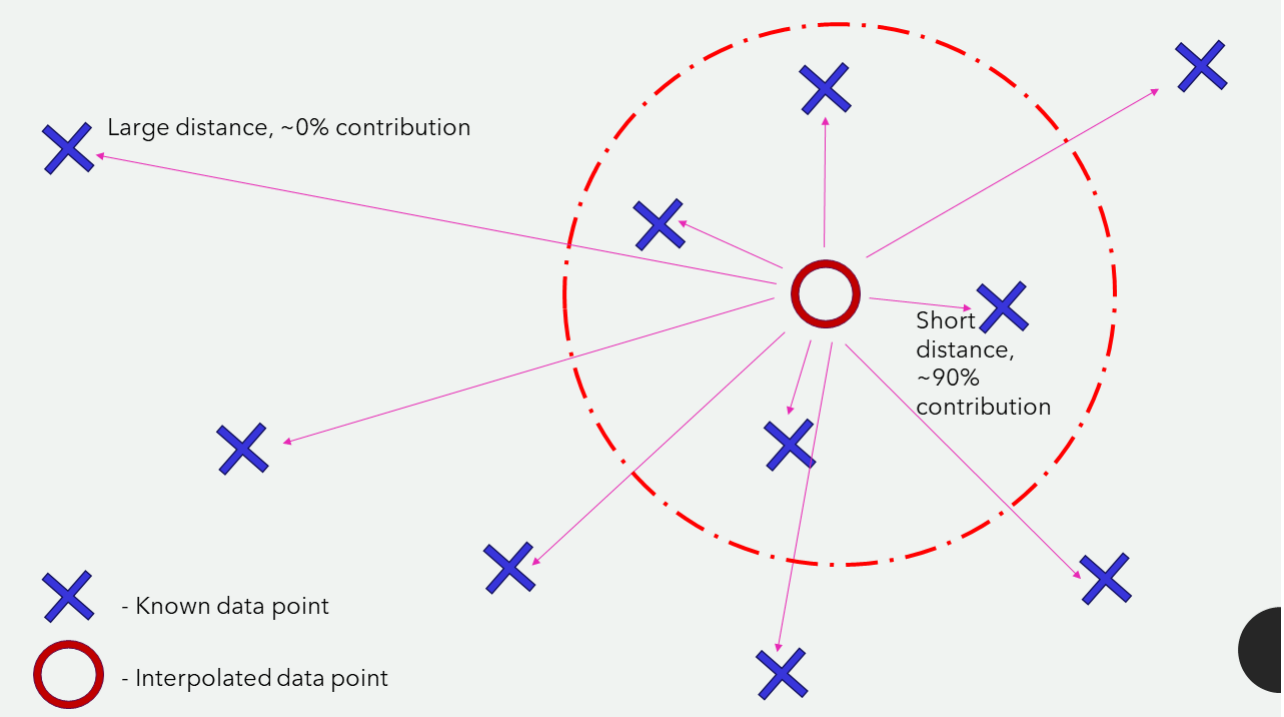

FIGURE 0.1: Schematic demonstrating the inverse distance weighting used in Shepard interpolation scheme, blue crosses represent known data points, red circle represents the value being interpolated and the dashed red circle represents the threshold.

Modified Shepard Interpolation

The modified Shepard interpolation is an extension of this method which incorporates first-order (and/or second-order) Taylor expansions to produce a more accurate interpolation:

$$T_i(R) = \mu(i) + \sum_j \frac{\partial \mu(i)}{\partial R_j} \Delta R_j(i) \tag{0.6}$$

Where $\Delta R_j(i) = R_j(i) - R_j$ for some coordinate, j. This is described visually in the figure 0.2

Throughout my project, I had found that for data calculated using a first-order modified Shepard interpolation resulted in a $\sim 5\%$ error on average (compared to true values) and an absolute worst error of 9% which is very low. Furthermore, rather than calculating analytic derivatives (via electronic structure theory) I had used approximate numerical derivatives calculated by central difference:

$$\left.\frac{d\mu_\alpha}{d\phi_1}\right|_i \approx \frac{\mu_\alpha(\phi_1(i+10^\circ), \phi_2(j)) - \mu_\alpha(\phi_1(i-10^\circ), \phi_2(j)}{20^\circ}, \qquad \alpha = x, y, z. \tag{0.7}$$

This is very computationally cheap and was found to improve the accuracy of interpolated values by $\sim 20\%$.

This method can further be built up to incorporate second and third-order derivatives (and so on) however the improvement of accuracy achieved does not scale well with the computation expense of calculating these derivatives.

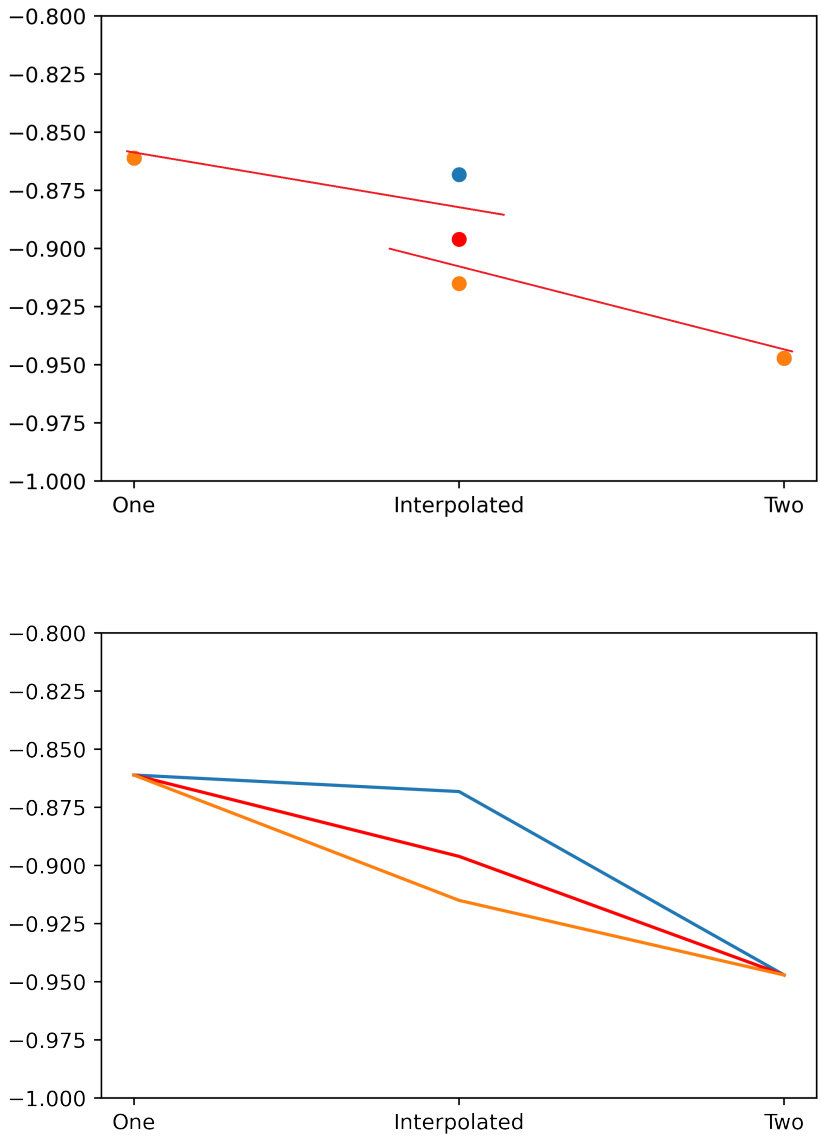

FIGURE 0.2: (Top) Schematic showing how derivatives are incorporated in interpolation of a point (red) between two known data points. Blue point represents interpolated value with no derivative, red point represented first-order interpolated point, orange points represent known (real) data and red lines represent numerical derivatives of known points. (Bottom) Schematic comparing Shepard interpolation, modified Shepard interpolation and real data curves.