# Hamiltonians and densities

The fundamental task of electronic structure theory is computation of the energy and other properties (dipole moment, etc) as a function of molecular geometry, E(**R**), where **R** are the 3Natom coordinates specifying the locations of the atoms. (R = x1,y1,z1, x2,y2,z2,x3,y3,z3… where x1,y1,z1 are the Cartesian coordinates of atom 1 etc.). Although the energy is the target, there are a few refinements:

* In practice, the absolute energy of a molecule is not measureable and not of any real interest. All uses of the computation will be examining energy differences, where the difference is typically a change in geometry but may also be a change due to an external environment.
* Most applications require derivatives of the energy with respect to atomic position, and finite difference for computation of such derivatives is usually computationally prohibitive. Instead, methods that compute the derivatives can be developed whose computational time is negligible compared to the computation of E(**r**). Computation of such derivatives is the sub-field of “analytic derivatives” in computational chemistry, and has consumed years of man effort. For a method to be practical, it must either harness past work on analytic derivatives (for instance, by using established quantum chemical algorithms) or tackle the analytic derivative issue on its own.
* Since the energy depends only on the relative orientation of the atoms (i.e. it does not change with overall translation or rotation), the geometry **R** is typically given in terms of internal coordinates (bond lengths and angles). The format of this internal coordinate geometry specification is the z-matrix.

Electronic structure theory is based on the time-independent Schroedinger equation, which is an eigenvalue problem:



where H is the Hamiltonian operator, which is typically written as a differential operator in the position and spin of the electrons. The eigenfunction, , is referred to as the many-electron wavefunction (since it captures the motion of all of the electrons in the system).  therefore has explicit dependence on the 3Nelec coordinates of the electrons in the system along with a dependence on the spin of the electrons (which can be either up or down). In addition, both H and  have a parametric dependence on the nuclear positions, **R**, i.e. for any given geometry, **R**, you can write down the Hamiltonian and solve it for .

There are many approaches to solving the above Schroedinger equation, which vary in philosophy, accuracy, and computational cost. The functional form of the wavefunction  varies widely across these methods and, in some common cases, the solution cannot even be written in terms of a wavefunction (e.g. perturbation theories and coupled-cluster theories). Instead, it is useful to express the solution of the Schroedinger equation in terms of densities:

* (1): captures the probability of an electron being at a particular position in the molecule
* (2): captures the joint probability that, if there is an electron at position 1, there is another electron at position 2.

These densities are sufficient to compute any property of the molecule, including the energy. There are a few reasons why the densities are sufficient to describe the molecule.

* Electrons are “indistinguishable” in the sense that you cannot design an experiment that would be able to distinguish the path of one electron from any other electron. So all we need to know is the probably of finding any electron a certain position.
* The forces are all 2-body forces. The attraction to the nuclei, and kinetic energy of the electrons, are obtained from (1). The electron-electron interactions are obtained from (2). These are all the forces in the system, so (1) and (2) determine all properties of interest.

The starting point for all practical techniques in quantum chemistry is the Hartree-Fock (HF), or self-consistent-field (SCF), approximation. HF theory is typically derived by assuming a particular form of the wavefunction (a single Slater determinant). In terms of densities, the Hartree-Fock approximation assumes the motion of electrons is uncorrelated, such that joint probability, (2), factors into a product of the probabilities, (1).

# Using a Basis set

A basis set is used to make the Schroedinger equation amenable to linear algebra techniques. The basis is used to describe the motion of a single electron.



Where (r) is a “molecular orbital”, meaning it describes the motion of an electron throughout the molecule. The basis functions (r) are centered on the individual atoms of the molecule and so are “atomic orbitals”. The above expression is often referred to as writing the molecular orbitals as a linear combination of atomic orbitals.

Given an atomic orbital basis, the first step in any quantum chemical approach is to compute the representation of the Hamiltonian operator in this basis. This is done by performing integrals over the atomic orbitals. The Hamiltonian is a sum of various terms:



The first three terms are the one-electron terms, H(1), since they depend only on the position of one-electron and so may be computed from (1). Each of these becomes a 2-index object in the basis, i.e. a matrix Mi,j where i and j span the basis set. The various terms in H(1) are:

 = kinetic energy of the electrons

 = interaction between the electrons and nucleus A

 = interaction between the electrons and charges in the environment

H(2) is the interaction between the electrons and becomes a 4-index object:

 = 2-electron matrix elements, often written in the notation (ij|kl)

The last term is a scalar describing the energy of interaction between the nuclei (Coulomb repulsion):



Where ZA and RA are the charge and location of the Ath nucleus.

The above matrices express the Hamiltonian, a differential operator, in the basis. Solution of the Schroedinger equation, at a particular level of approximation, generates the densities (1) and (2), which are also expressed as 2 and 4 index operators respectively:  and . The energy of the system is obtained as traces over products of the Hamiltonian and the densities:





The total energy is E = E(1)+E(2).

# Decomposition of the energy

Since H(1) is a sum of various contributions, the one-particle energy, E(1), can be decomposed into each of these contributions. For instance, the kinetic energy can be written



We can also decompose the energy spatially, for instance, in terms of atomic energies and energies between atoms. This is done by considering the atoms on which the atomic orbitals, , reside. The one-electron contributions to the energy, such as the kinetic energy, can then be decomposed as,



Where A, B refer to atom numbers, and i,j are atomic orbital indices. The diagonal elements EA,A are the atomic energies, and the off-diagonal elements EA,B are interaction energies. For atoms that are bonded, EA,B, can be thought of as the bond contribution to the energy. For atoms that are not bonded, EA,B, can be thought of as an interaction energy.

The same can be done for the two-electron energy:



The diagonal terms, EA,A,A,A, are the atomic energies (the unfavorable energy associated with electron-electron interactions on an atomic site). The Coulomb interaction between electrons on atom A and those on atom B is probably best viewed as . (Another possibility is to consider the sum of all  in which I,J,K, and L are: (i) not all equal and (ii) contain only A and B.) Current semi-empirical quantum chemical models assume is zero unless i,j are on the same atom and k,l are on the same atom. In this manner, these methods assume  is the only two-electron interaction between atoms. The existence of the other two-electron terms is something we will tackle when analyzing the two-electron energies.

For our purposes, the decomposition into atomic energies is particularly convenient because it allows us to compare across basis sets. Different basis sets have different numbers of basis functions, meaning that H and  have different sizes. However, EA,B and EA,B,C,D refer to atoms and so have the same size and meaning in all basis sets. This makes them directly comparable.

Decomposition into atomic energies is also important if the resulting models are to be applied to different molecules. If a model can reproduce the energy of a carbon atom in different environments (i.e. bound to different atoms and in different Henv), then the model is useful for making predictions regarding arbitrary molecules.

# General structure of the models

The energy of a molecule is a sum of various contributions that are physically quite distinct.

Current semi-empirical models invoke approximations by modifying the Hamiltonian matrix elements of a minimally-sized basis. If we apply the intuition underlying these approximations into energy decompositions, we get something like the following:

*  + : the one-electron energy of an electron on a particular atom, which includes both the kinetic energy of the electron and the attraction to that particular nucleus. The KE and nuclear interaction are often combined, as shown here, because together they specify the 1-electron energy of the electrons in that particular atom.
*  : the energy of interaction between the electrons on atom A and the nucleus of a remote electron.
* : the one-electron contribution to the bonding energy between atoms A and B.
* : the electron-electron coulomb energy associated with the electrons on atom A.
* : the two-electron contribution to the bonding energy between atoms A and B.

If we could develop a way to predict these individual energies in a general way, we could combine them to get the total energy of the molecule.

The data against which we can train is predictions of a high-level (HL) model for the energies decomposed into atomic contributions.

The input to the model is information generated from low-level (LL) models. The information from LL models may include, for instance,

* EA,B and EA,B,C,D: the energy decomposed into atomic contribution.
* (1) and (2): the densities from the LL model. We can create various combinations of the densities that capture potentially relevant features, such as the asymmetry of the charge distribution on an atom (i.e. the dipole, quadrupole, and higher moments of the atomic charge distribution).
* Projection on principal components of (1) and (2): feature extraction on (1) and (2) may reveal components of the density that dominate the variations of (1) and (2) across the molecular data set. The projection onto these components may then provide useful inputs for a model.

Locality is an important aspect of the modeling effort. Ideally, the energies associated with a particular atom, EA,A and EA,A,A,A, would be functions only of the LL model predictions for properties of that atom. Likewise, energies associated with bonded atoms, EA,B and EA,A,B,B would be functions only of LL model predictions for terms involving the bonded atoms. It may well be, however, that inclusion of more remote information may be necessary, e.g. EA,A and EA,A,A,A may depend on LL model predications for atom A and the atoms to which A is bonded. Locality is a central aspect of chemical systems, so we expect that there will be a length scale beyond which inputs from the LL model will be irrelevant for the predictions. Identifying this length scale is a useful target for early studies.

# An initial model

The LL and HL models differ both in the size of the basis set, and the inclusion of electron correlation. Our past work has suggested the more difficult LL🡪HL maps are those where the LL model uses a smaller basis than the HL model. The extrapolation from LL Hartree-Fock models to HL models that include electron correlation, (i.e. HL models for which (2) is not a product of (1)’s) seem to be less challenging. In addition, generating training data for HL models that include electron correlation is much more computationally expensive than HL Hartree-Fock models. So, both from the perspective of computational expediency and from the perspective of taking on the more difficult aspect of the machine learning problem, it is useful to consider mapping from Hartree-Fock in a small basis to Hartree-Fock in a large basis.

## Training data

Each row of the training data corresponds to a different molecular system. The different rows can vary in three different ways:

* **External electrostatic environment** This is likely the weakest variation. The external environment is used to push the electron density in various directions, and so yield information on how the energies relate to the density in a particular molecule
* **Molecular geometry** For a given molecule, we can change the geometry by altering bond lengths and bond angles. This is likely a stronger perturbation than simply pushing electron density around with external charges.
* **Different molecules** We can generate different molecules that contain the same atoms. The goal is develop a model that can describe, for instance, any molecule consisting of C, H, N, and O. (Modern *ab initio* theory claims success, although at high computational cost, on the atoms H, C, N, O, F, Cl and perhaps some other atoms in particular cases).

We will refer to each row of the table as a different system, where system takes together all of the above features (molecule, molecular geometry, and external electrostatic environment).

For a given collection of molecules, we would ideally like to generate the HL (large basis set) result for the decomposed energies ,, . Our initial attempts at this, which assume that the orbitals centered on an atom provide a good definition of the atom, suggest that we need to do develop a better decomposition scheme to be able to decompose the large basis set results into atomic components. The decomposition of the energy into components corresponding to different operators is, however, rigorous:  , , and .

## Model 1

This model takes linear combinations of predictions from a set of LL models performed on the same molecular system (i.e. LL calculations on the same molecule, with the same geometry and the same external environment, as the HL model). An initial set of LL models are Hartree-Fock solutions in a modified STO-3G basis, where the modifications are obtained by scaling the width of the atomic basis functions i. The data table therefore has the structure:



A proposed notation for this model is: (LL1 LL2)/(P1 P2 ..PNp) 🡪 HL/P. For instance, consider using modified STO-3G with narrow (S=0.8) and broad (S=1.2) scaling, and considered using the KE of carbon atom 1 to predict the KE of Carbon from 6-31G\*\*. This would be written as:

(STOs=0.8, STOS=1.2)/() 🡪 6-31G\*\*/

To address locality, we can add LL predictions that include properties from adjacent atoms:

(STOs=0.8, STOS=1.2)/(, , ) 🡪 6-31G\*\*/

Where C1-B means the set of all atoms that are bonded to C1.  is therefore the KE associated with all of the bonds to atom C1 (4 values if there are 4 atoms bonded to C1). Similarlycorresponds to the KE on atoms bonded to C1. (For 4 atoms the notation refers to diagonal elements and so corresponds to 4 values, while the notation  corresponds to diagonal and off-diagonal elements, or 10 values given that the matrix is symmetric.)

It is also useful to consider maps that use inputs with different properties,

(STOs=0.8, STOS=1.2)/(, ) 🡪 6-31G\*\*/

# Future directions

As the training data expands to include more diverse classes of molecules, it may become important to use different model parameters for different situations. For instance, it is likely that the model parameters required for a carbon bonded to four other atoms (4-coordinate carbon) differ from those required for 3-coordinate atoms. The model would then use a classification stage to select a regression. The discovery and use of these classes is an area where this type of model is likely to be substantially better than other models.

The above model finds the Hartree-Fock solution for two different STO basis sets, and then combines these to make a prediction:



An alternative approach, that is much closer to current approaches to semi-empirical quantum chemistry, is to instead combine the integrals as part of the self-consistent iterations of the Hartree-Fock procedure:



The advantage here is that this model can allow the combination of integrals from the LL models to be functions of the density matrix. Such a model has a greater chance of being sufficiently flexible to handle a large class of molecules.

## Aggregation based on atom types

For the model to have broad utility, it should be applicable to different molecules. This can be done by defining parameters that are associated with atom types, as opposed to specific atoms. The first pass on typing can simply be element types (C, H, etc). For aggregation based on types, we can write:

## Implementation

Description of the classes for implementing the above. Properties as shown in italics and methods are shown in bold italics

## Environment class

A set of *ncharge* external charges with magnitudes *rho(1,i)* and cartesian coordinates *r(3,i)*, where the first index is x,y,z.

***newCube(size,mag)*** [static] creates a cube with corners (size(1),size(2),size(3)),   
(-size(1),size(2),size(3)), … with magnitudes randomly generated between –mag and mag. The charges are therefore at the corners of a rectangle with lengths 2\*size(i) centered at (0,0,0).

***gaussianText()*** returns a string that describes the charges in Gaussian format.

***compare(environment)*** returns 1 if the environments are equal. **Note that the charges must be identical and in the same order.**

***plotFig(nfig)*** puts a representation of the charge field in figure number nfig. Note that it shows red circles for negative charge and blue circles for positive charge, but does not indicate the charge magnitude.

## Fragment

Each instance handles a particular molecule in a particular geometry and described with a particular basis set. The class also handles a set of external charge environments. The reason for this data partitioning is that all elements of the Hamiltonian will change when a molecule’s geometry is changed or when the basis set is changed. However, a change in environment changes only Henv.

***defaultConfig()***[static] returns a default structure specifying the configuration of the calculation. This config object has the properties:

* *template:* name of a file containing the z-matrix that specifies the molecular geometry. The template may contain strings (par1 par2 par3) that will be replaced the parameters discussed below.
* *basisSet:* string specifying the basis-set, in Gaussian format
* *charge:* overall charge on the molecule
* *spin:* total spin state of the molecule (will typically be 1)
* *par:* vector holding the values that will be put in place of par1, par2,.. in the template file.

***Fragment(datapath,config)***[constructor] Does computations of the molecule specified in config, in the absence of external charges. The results are saved in datapath so that if the same calculation is requested again, the data is simply read from the already computed files. (Note that “same calculation” means the config structures are identical.)

***setEnvSize(nenv)*** Predimensions the object to handle storage of this number of environments. This is to prevent the arrays from needing to grow as environments are added. (Note that a call to this method destroys all environment data, and so should only be called before addition of any environments.)

***addEnv(env)*** Perform calculation in the environment env, and add the results to the object. If the calculation has already been performed, the data is loaded instead of being computed.

***loadAllEnv()*** Look through the datapath directory and load results for all environments that have already been computed.

***density(ienv)***  in environment number ienv. If ienv=0 (default), then the result for the isolated molecule (no external charges) is returned.

***density2p(ienv),*** with ienv treated as in density(ienv).

***printBasis()*** displays information on the atomic basis set (using matlab disp() function).

***partitionE1(env,…)*** computes sums of the type . The input is flexible to allow summation over different sets of orbitals, and for different one-electron operators (see matlab documentation).

***PartitionE2(env,…)*** computes sums of the type . The input is flexible to allow summation over different sets of orbitals (see matlab documentation).

All matrix elements H and results from the Hartree-Fock solution are stored as public properties in the object, as documented in the matlab file.

## Intended use

The workflow for generation of data for an experiment is the following:

% First create some environments

size = [3,3,3];

mag = 1.0;

nenv = 10;

for ienv = 1:nenv

env{ienv} = Environment.newCube(size,mag);

end

% Then generate HL data

c1 = Fragment.defaultConfig();

c1.template = 'h2';

c1.basisSet = '6-31G\*\*';

c1.par = 1.0;

fragHL = Fragment('data',c1);

fragHL.setEnvSize(nenv);

for ienv=1:nenv

fragHL.addEnv(env{ienv})

end

% generate LL data

c1 = Fragment.defaultConfig();

c1.template = 'h2';

c1.basisSet = 'STO-3G';

c1.par = 1.0;

fragLL = Fragment('data',c1);

fragLL.setEnvSize(nenv);

for ienv=1:nenv

fragLL.addEnv(env{ienv})

end