Encounter: A Cross-Platform Interaction Energy Calculator

Introduction

In quantum molecular modeling, atomic orbitals are represented using Basis Functions which, when combined as a linear combination, provide a description of the molecular system. The smallest number of functions to describe the system is equal to that which will just account for all the electrons within it. The larger the number of basis functions, the more accurate the description, with an infinite number for complete accuracy. Application of an infinite number of functions is not possible, and therefore a limited number are used thus leading to an incompleteness of the description.

When determining the interaction energy between two molecules in a system, a problem arises. For a two molecule system, for example, a simple subtraction as shown in Equation 1, is not possible.

$$\Delta E_{int} = E_{dimer} - (E_A + E_B)$$

Equation 1: Simple interaction energy calculation where A and B correspond to the two separate molecules in the dimer

This is due to an effect known as the Basis Set Superposition Error (BSSE). Basis functions from one component, being in close proximity to the other, can provide and compensate for the incompleteness of the other and vice-versa, thus leading to a “mixing” of the functions, or superposition. This leads to an overall lowering of the energy of the dimer, and therefore an overestimation of the interaction energy.

The solution is to use the Counterpoise Correction. For a dimer, five calculations are performed: one for the dimer using the full dimer basis, one calculation for each component with the geometry in the dimer using the component basis, and one calculation for each component using the full dimer basis. For the latter two calculations, only one component is present but the orbitals, as “ghost” orbitals, of the other component are also present. This leads to a lowering of the component energies as more basis functions are used for each calculation. An estimate of the interaction energy can be calculated from these values as shown in Equation 2.

$$\Delta E_{Int} = E(AB)_{Dimer} - [E(A)_{Dimer} + E(B)_{Dimer}]$$

Equation 2: Interaction energy calculation where the final two terms include the “ghost” orbitals of the other component

Encounter has been built to calculate this interaction energy from the output of a Counterpoise Correction calculation created using the GAUSSIAN modeling package. Using regular expressions, the five energy values are extracted as text, and then converted into numeric form. These values are then used to calculate the interaction energy, in both Hartree atomic units and kJmol^{-1}, and binding constant. These calculations are outlined in Equations 3 and 4.

$$E/\text{kJmol}^{-1} = E/\text{Ha} \times 2625.5$$

Equation 3: Conversion of energy from Hartree atomic units to kJmol^{-1}

$$K_{Bind} = exp\left(\frac{E_{int} \times 1000}{-8.314\ \mathrm{JK^{-1}mol^{-1}} \times 298\ \mathrm{K}}\right)$$

Equation 4: Calculation of the binding constant

Encounter is open-source software released under the Lesser General Public License (LGPL). It is available from CodePlex at http://encounter.codeplex.com. The provided CD-ROM contains the latest version of Encounter at the time of writing in both source and binary form.

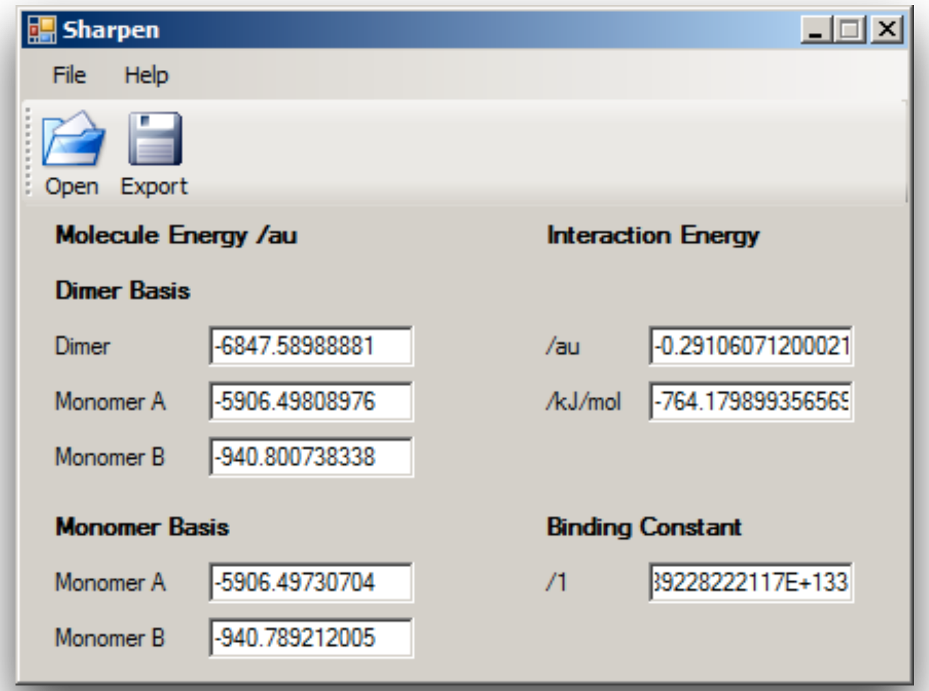
Language Implementations

Four different implementations in different languages have been built of which three have a general user interface (GUI), the other having a command-line interface, as indicated in Table 1.

Language	Program Name
C++ (Qt)	Encounter
C# (.NET)	Sharpen
Java	JCounterpoise (command-line interface)
Objective-C	Xen

Table 1: Implementations and program names of Encounter

Despite the differences between the languages, each implementation has been written to work in as similar a fashion as possible to the others. Implementations with a GUI share a common design as shown in Figure 1.



Molecule Energy /au

Dimer Basis

Dimer -6847.58988881

Monomer A -5906.49808976

Monomer B -940.800738338

Monomer Basis

Monomer A -5906.49730704

Monomer B -940.789212005

Interaction Energy

/au -0.29106071200021

/kJ/mol -764.17989935656

Binding Constant

/1 39228222117E+133

Figure 1: Common GUI Design, demonstrated by the .NET implementation, Sharpen

Building from Source

The CD-ROM provided contains the source code of all implementations in the *src* directory, and binaries of the C# and Java versions in the *bin* directory. System requirements are outlined below.

C++ (Qt): Qt Framework 4.7 or newer

C# (.NET): Microsoft .NET Frameowrk 2.0 or newer

Equivalent alternative implementations of the Microsoft .NET Framework, such as Mono or DotGNU, can be used although Sharpen functionality and stability status under these alternatives is not known at the time of writing.

Java: Java 6 Standard Edition

Objective-C: Mac OS X 10.7 "Lion" or newer

Source code for all implementations of Encounter is also available via the Mercurial source code repository provided by CodePlex. To clone the repository run the following commands on the command-line:

```
hg clone https://hg01.codeplex.com/encounter
cd encounter
```

Building Implementations using an Integrated Development Environment (IDE)

All implementations can be built using an IDE, and is the recommended building method for the C++, C#, and Objective-C versions. To build, simply load the project file for the implementation and build using the standard tools within the IDE. IDEs supported for the implementations are outlined below:

- **C++ (Qt):** Qt Creator
- **C# (.NET):** Microsoft Visual Studio 2010 (2008 if project file is accessed directly)
- **Java:** Any Java-supported IDE, e.g. Oracle NetBeans
- **Objective-C:** Xcode 4

To build the Java implementation within an IDE, the source code files (*.java) should be imported and built using the IDE build tools.

Building Implementations using Command-Line Tools

This is the recommended method for installing the Java implementation and a simple alternative for the C++ version. For UNIX-based systems, a configuration script is available which generates a *Makefile* to control the build, installation, and uninstallation of the Java implementation. Direct compilation using a Java compiler is also possible.

To use the configuration script, load the included *configure* file in a text editor and follow the instructions to edit the necessary variables at the top of the file. Do not edit beyond this section as marked in the file. When complete, save and close the file, and run the following commands on the command-line:

```
./configure
make
```

If no errors occurred during configuration or building, the binaries and a shell script will be built in the directories specified in the configuration script.

To build using the Java compiler, run the following command on the command-line from the directory containing the Java sources. The *-d* option can be used to specify an output directory for the binaries.

```
javac *.java
...or...
javac *.java -d ~/bin
```

The binaries will be built in the same directory as the sources unless an alternative is specified.

To build the C++ implementation from the command-line in the source directory, run the following commands to generate a *Makefile* and build the binary:

```
qmake
make
```

The binary will be built in the same directory as the sources.

Program Usage

A sample GAUSSIAN Counterpoise Correction calculation file is included on the CD-ROM in the *samples* directory which is an example of a crown ether, [18]crown-6, with a bound potassium ion. A structure is shown in Figure 2.

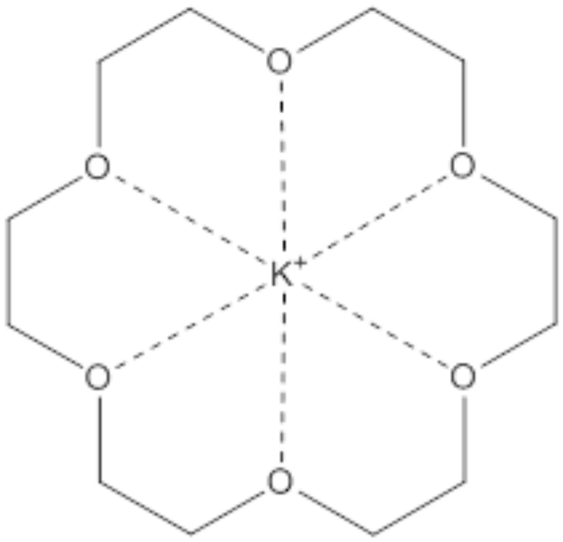


Figure 2: Crown ether [18]crown-6 with bound potassium cation

Implementations with a GUI (Encounter, Sharpen, and Xen)

As already outlined, implementations with a GUI operate in an identical fashion. The main difference lies in the keyboard shortcuts required for operations within the program.

Opening a File

To open a file, simply use the File->Open menu command or the Open toolbar button. Alternatively, type Ctrl+O in Encounter and Sharpen, or ⌘O in Xen. Using the Open dialog box, browse to the directory containing a Counterpoise calculation file and open it. If no errors occur, all text fields in the main application window will be populated with the 5 values from the calculation file, and the calculated interaction energy, in both Hartree atomic units and kJ/mol, and the binding constant. If an error does occur, a dialog box or sheet will be displayed describing the situation.

Exporting a File

Currently, exporting of data to comma separated value (CSV) format is supported. CSV files can be opened with common spreadsheet applications, including Microsoft Excel, OpenOffice.org Calc, and Apple Numbers.

To export the data, use the File->Export menu command or the Export toolbar button. Alternatively, type Ctrl+S in Encounter and Sharpen, or ⌘S in Xen. Using the Export dialog box, choose a filename in the necessary directory and export.

Any unavailable values, resulting from incomplete datasets or no energy values being present in the calculation file, will not be displayed in the CSV file, although it can still be opened in a spreadsheet application. The cells which should contain those missing values are simply empty.

Implementation without a GUI (JCounterpoise)

Running via the Java Runtime

Similar functionality to the implementations with a GUI is provided via command-line flags and options.

To run JCounterpoise, simply run the following command on the command line in the directory containing the binaries. This will offer a prompt requesting the filename of a calculation.

```
java JCounterpoise
```

Alternatively, the filename can be provided as an argument, e.g. *calc.log*:

```
java JCounterpoise calc.log
```

To export the data to comma-separated values format (*.csv), use the *-o* or *--output=* option, which can be placed before or after the filename:

```
java JCounterpoise calc.log -o calc.csv
```
...or...
```
java JCounterpoise calc.log --output=calc.csv
```

Information about JCounterpoise can be accessed using the *-v* or *--version* flags:

```
java JCounterpoise -v
```
...or...
```
java JCounterpoise --version
```

If an error occurs on loading the calculation file, an error will be displayed describing the situation.

Running via the Shell Script

UNIX-based systems can utilise a shell script to run JCounterpoise from the directory it was built using shorter commands. By default, the command is:

```
jcp
```

This will offer the standard prompt for a filename. Alternatively, a filename can be provided:

```
jcp calc.log
```

To export data in CSV format, the *-o* option can be used but can only be placed after the calculation filename in the command line:

```
jcp calc.log -o calc.csv
```