The libint-eigen interface

Laurent Lemmens

Thursday 17th August, 2017 17:02

Contents

1 Basis sets

In the libint basis set context, an *orbital* refers to a *GTO* (Gaussian-type (atomic) orbital), which is just a *basis function*. Orbitals/GTOs/basis functions are used to form *molecular orbitals*, in which Slater determinants are expanded. Finally, the final CI wave function is written as a linear combination of Slater determinants.

A (Cartesian) GTO, with angular momenta l_x, l_y and l_z and centered on $\mathbf{R} = (X, Y, Z)$, has the following mathematical form:

$$\phi_{\zeta,l_x,l_y,l_z,\mathbf{R}}(x, y, z) = N(x - X)^{l_z}(y - Y)^{l_y}(z - Z)^{l_z} \exp\left(-\zeta r^2\right), \tag{1}$$

in which

$$r^2 = x^2 + y^2 + z^2 . \tag{2}$$

The sum of the angular momenta

$$l = l_x + l_y + l_z \tag{3}$$

determines the type of AO: $l = 0$ refers to an s-type orbital, $l = 1$ to a p-type orbital, etc. (analogous to naming of the eigenfunctions of the hydrogen atom).

2 Are the contracted basis functions normalized?

When constructing a libint2 :: BasisSet object as in say

```
libint2::BasisSet obs ("STO-3G", atoms);
```

the corresponding file sto-3g.g94 is read (which is located at \$LIBINT_DATA_PATH), in which LibInt2 finds the exponents and contraction coefficients for the given basis for a given element.

In libint2 / basis .h, we can see the following (edited for brevity) code:

```
static ... read_g94_basis_library(...){
    ...
    ref_shells[Z].push_back(
        libint2::Shell{...}
        )
    ...
}
```

which calls a specific constructor of libint2 :: Shell:

```
Shell(...) {
    // embed normalization factors into contraction coefficients
    renorm();
```

that makes sure that the CGTO is normalized by including the normalization factor in the contraction coefficients. So, yes, LibInt2 internally works with normalized basis functions.