Fermi-orbital descriptor Monte-Carlo - `fodMC`

Kai Trepte, Central Michigan University,
trept1k@cmich.edu

1 General idea

It can be shown that Fermi-orbital descriptors (FODs) in FLO-SIC tend to arrange in symmetric patterns for atoms and molecules, considering different subshells. Accordingly, distributing points on spheres as far away from each other as possible (treating each point as a point charge interacting via Coulomb repulsion) might yield resonable FOD starting points. One can imagine two different species of points ($\uparrow$, $\downarrow$) to be treated (mainly) independently. Thus, the problem of FOD generation 'reduces' to an even distribution of points on spheres. This general idea has been employed using a Monte-Carlo (MC) algorithm to distribute points on spheres, which is mainly used for the core FODs. For the valence, a differentiation between bond and lone FODs is being made. This can be done in a rather deterministic fashion using the atomic coordinates as input. More details are given below.

The even distribution of points on a sphere can be very difficult once the number of points increases and symmetry between the points is either not obvious or not present. Accordingly, finding an analytic solution to any number of points on a sphere is a challenging problem. Thus, it would be desirable to describe the distribution of points by a quantity which can be described analytically (or very well numerically). Here, the Coulomb repulsion between all points was minimized assuming the points to be point charges. It should be considered that the distance is taken within three-dimensional euclidean space, and not as distances on the surface of the sphere. With that, the quantity to minimize is f_r given by

$$f_r = \min\left(\sum_{i=1}^{N}\sum_{j\neq i}^{N}\frac{1}{r_{ij}}\right), \tag{1}$$

where each distance is evaluated like

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

with x, y, z being the coordinates of points i and j.

2 Outline of Monte-Carlo simulation

To obtain the f_r, a Monte-Carlo (MC) simulation has been employed (written in FORTRAN, called `fodMC`). An example input will be provided later. Here, the basic concept for the distribution of points on a sphere will be briefly explained. First, the radii of the spheres r_k, the number of points on these spheres N_k, the number of MC steps n and the step size a_{step} are provided as input. Thus, everything is done in spherical coordinates (r, θ, ϕ). All N_k positions are initialized at (r_k,0.0,0.0), which represent the configuration with the largest possible Coulomb repulsion ($1/r = \infty$, global maximum). This was done to allow the MC to minimize the Coulomb repulsion without initializing the positions in some sort of unfavourable configuration (local minimum). Furthermore, by using the same initial starting positions the MC can be run several times to ensure consistent result when starting from the global maximum. After the initialization, the actual MC steps are carried out. For each position, two random numbers rand_1 and rand_2 are generated in the interval of [0,1]. These serve in the construction of a new position of point i ($i \in N_k$) at step n according to the ideas of a Metropolis algorithm

$$\theta_{i,n} = \theta_{i,n-1} + (2 \cdot \text{rand}_1 - 1) \cdot \pi \cdot a_{\text{step}} \tag{3}$$

$$\phi_{i,n} = \phi_{i,n-1} + (2 \cdot \text{rand}_2 - 1) \cdot 2\pi \cdot a_{\text{step}} \tag{4}$$

$$x_{i,n} = r \cdot \sin(\theta_{i,n}) \cdot \cos(\phi_{i,n}) \tag{5}$$

$$y_{i,n} = r \cdot \sin(\theta_{i,n}) \cdot \sin(\phi_{i,n}) \tag{6}$$

$$z_{i,n} = r \cdot \cos(\theta_{i,n}). \tag{7}$$

After all positions have been moved, the Coulomb repulsion according to equations (1) and (2) is evaluated. If $f_r^n < f_r^{n-1}$, the new positions are taken into the next MC step. If not, the old positions are taken again. The entire approach is summarized in figure 1. The step size is currently fixed to 0.001, which provides resonable results. The number of MC steps is defined by the number of electrons/FODs in the system. This number is multiplied by 1000 to get the number of steps. The minimum is set to 5000 and the maximum is set to 50000 to avoid insufficient or too many steps.

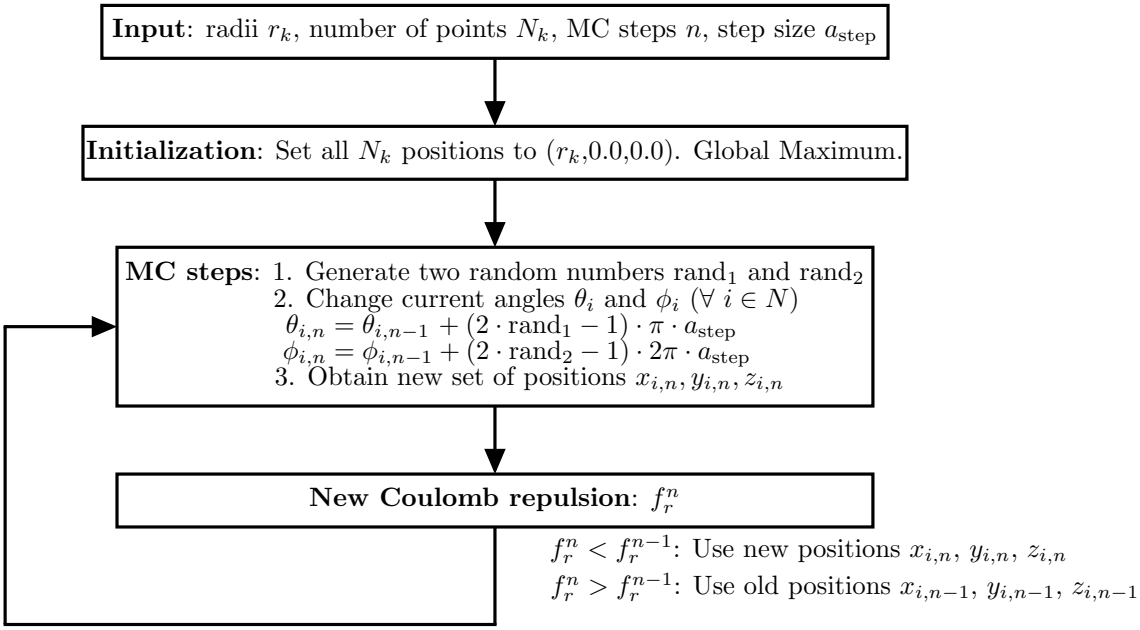

Figure 1: Monte-Carlo approach to evaluate the distribution of $N = \sum_k N_k$ points on multiple spheres according to a minimization of the Coulomb repulsion of all points.

3 Further steps

The procedure in figure 1 is a simplified version of the actual algorithm. Some further steps were implemented to give more resonable results. These steps are summarized in the following.

- **Spin** There is a strict differentiation between spin up (↑) and spin down (↓) points. This allows the treatment of spin-polarized systems and generally gives a more reasonable picture of the FODs in space. In addition, for atoms the ↑ and ↓ points are rotated against each other (after being individually optimized) to formally decrease their Coulomb repulsion (rotations are again based on MC). This leads to inverted tetrahedra of ↑ and ↓ in e.g. Ne.

- **Molecules** For molecules, the formation of bonds and lone pairs is essential. To form bonds, the center between atoms is taken as a starting point. The number of bond FODs are distributed either at this center (single bond) or are initialized perpendicular to the bond axis (multiple

bonds). This is in clear analogy to the procedure described by Luken [1]. For the latter, MC steps are carried out to distribute the number of bond FODs equally around the bond axis (via rotations). For lone FODs, a similar ansatz is used. First, the vectors describing all bonds to a given atom are added up to obtain one resulting 'bond' vector. In the opposite direction of this vector, the lone FODs have to reside. Thus, taking this vector one can define a point away from all bonds for a given atom at which the lone FODs are initialized. For multiple lone FODs, essentially the same procedure as for bond FODs is carried out. This gives an easy and intuitive formation of bonds and lone pairs.

In addition, there is the possibility to form e.g. 2-center-5-electron bonds by specifying the number of $\uparrow$ and $\downarrow$-FODs (which contribute for a given bond) individually. This is in line to the double quartett theory proposed by Linnett [2].

The more lone FODs there are for an atom, the closer they are generated to that atom. This gives a better description of the lone FODs in general.

- **Separate core and valence** The procedure mentioned above only deals with the valence FODs. The core FODs are introduced afterwards by symmetrically placing the core FODs on their respective spheres and optimize them with respect to the valence FODs, i.e. they minimze the $1/r$ globally. This gives somewhat resonable core FODs even for larger cores.

ATTENTION: A visual inspection of the generated guesses is essential!

4 Compilation

The provided folder structure is

- doc: Contains this document
- src: Contains the source code
- examples: Contains a number of examples. Run the code here

To compile the code, go into the src folder and run

```
bash compile.sh
```

to generate an executable with name fodMC. Afterwards, go to the examples folders, change the header of run.sh to include the correct path to your executable

```
export fodmc=[PATH TO THE FOLDER OF YOUR EXECUTABLE]$fodmc
```

Once the `system` file is set up, simply execute the program by

```
bash run.sh
```

5 Current status/version

The current status of the `fodMC` script is

- Atomic guesses are generated automatically. An example input is given under 'Example input'. There is no further input required.
- The molecular guess generation is not fully automatized. Input is needed for the number of bonds (or, to be more precise, the number of $\uparrow$ and $\downarrow$-FODs between atoms) and the number of lone FODs per atom. See 'Inputs/Outputs' for an example.
- Single bonds are placed automatically. Lone FODs for charge neutral systems are placed automatically as well.

!! ATTENTION !! The approach to generate FOD positions based on distributing points on spheres has the advantage of its speed (only seconds up to a few minutes to generate guesses for molecules). However, the guesses have to be carefully inspected by the user to make sure that the guesses make sense! It is based on random numbers after all. Thus, whenever you generate a guess you should always have a look at the resulting FOD geometry.

6 Inputs/Outputs

There are two input files: `system` and `xx_database_xx`. The file `system` contains all the information about the system for which FODs shall be generated (see Examples). The second file `xx_database_xx` contains the radii and number of FODs for all shells of all atoms. It is used to generate the atomic guesses as well as the core guesses in molecules automatically. Feel free to modify the parameters if you know what you are doing.
Three output files are generated: `CLUSTER`, `FRMORB` and `Nuc_FOD.xyz` (in periodic cases, an additional file called `Nuc_FOD.cif` is generated). The first two are standard input files for NRLMOL and can be used as given (they are always generated in Bohr). The file `Nuc_FOD.xyz` contains the last FOD geometry as well as the molecular geometry (in Å) and can be used as an input for PyFLOSIC. It is a simple xyz-file and can be visualized with a number of viewers (e.g. Xcrysden, ASE-gui, VMD, etc.). Please always review the generated guess before using it. In the last file, atoms are represented by their respective symbols in the periodic table, ↑-FODs are given as X (ghost) atoms and ↓-FODs are given by He atoms.

Example: Atoms As already mentioned, the atomic guess creation is automatized. The script will recognize to generate an atomic guess if the number of atoms equals 1. Below, an example input for the generation of a sodium guess is given. Each line will be explained below.

```
1 sodium atom
bohr
Na 0.0 0.0 0.0
```

The first line specifies the number of atoms in the system (here: 1). Only this number is read in, while everything after it is considered as a comment and will be ignored. The second line defines which units shall be used (`bohr` or `angstrom`). The next line specifies the element (here: Na) and the coordinates of this element (here: origin). The element needs to be specified as given in the `xx_database_xx`, thus always use the proper element description (first letter is always a capital letter). For transition metals (3d), the corresponding electronic configuration needs to be given, e.g. `Cu_4s1` or `Cu_4s2`.

All atoms are generated spin-unrestricted, even formally closed-shell systems like Ne. One can make them strictly closed-shell by deleting the second set of FODs (This is not recommended!).

Example: Molecules For molecules, the input becomes a little more complicated, because the bond pattern. An example is given below for SO_2.

```
3 SO2
bohr
S 0.00000  0.00000  0.6863
O 0.00000  2.34250 -0.6863
O 0.00000 -2.34250 -0.6863
bonds: (centerA-centerB)-(FODS_UP-FODS-DN), lone: centerA-(FODS_UP-FODS_DN)
(1-2)-(2-2) (1-3)-(2-2)
```

The first line specifies the number of atoms N_{atoms}, just like in the atomic case. The second line defines the units to be used. The next lines specifies the N_{atoms} lines for each element with its respective coordinates (here in bohr). The format is the same as a simple xyz-file. Next is another comment line, below which is the bond pattern and the number of UP and DN lone FODs. In this example, the sulphur forms a double bond with the first oxygen (1-2)-(2-2) and another double bond with the second oxygen (i.e. the third atom and accordingly (1-3)-(2-2)). The lone FODs (i.e. one lone pair on S and two lone pairs on each O are formed automatically). To summarize the input structure

$(\text{atom}_i\text{-atom}_j)\text{-}(\text{FOD}_{\text{UP}}\text{-FOD}_{\text{DN}})$ $\forall$ bonds between i and j
$\text{atom}_i\text{-}(\text{loneFOD}_{\text{UP}}\text{-loneFOD}_{\text{DN}})$ $\forall$ lone FODs on atom i

Only bonds which differ from single bonds need to be specified. Single bonds are initialized automatically. This initial setting can be overwritten as explained above. Lone FODs only need to be excplicitely specified when the number of lone FODs differs from what the fodMC generates. Otherwise, lone FODs are placed automatically. Accordingly, one specifies the bond partners as well as the numbers of UP and DN FODs in that bond. With that, any bond pattern can be initialized. There can be more than one input line for the bond pattern. It has to be made sure that the entries of such lines start with a '('. Any other line is interpreted as input for lone FODs. The output on the screen is

```
!!! MOLECULAR GUESS WILL BE CREATED FOR     SO2
charge = 0.000   spin = 0.000   Point charge dipole -0.00381    0.00216    2.69386
Total CPU time was   0.93999999999999995      s
```

summarizing the simulation details, giving information about the generated guess (charge, spin, point charge dipole (excluding 1s core-FODs)) and tells how long it took to generate the guess.

In addition, the `Nuc_FOD.xyz` file in this case looks like

```
  35
 angstrom
S        0.00000000     0.00000000     0.36317418
O        0.00000000     1.23959715    -0.36317418
O        0.00000000    -1.23959715    -0.36317418
X        0.00034265    -0.00036675     0.36334185
X       -0.10614229    -0.19874292     0.45942622
X       -0.03285114     0.00073056     0.12037868
X        0.23984455    -0.00222819     0.41316432
X       -0.10159076     0.20117828     0.45927833
X       -0.42799888     0.64554456     0.04393850
X        0.42798899     0.64558655     0.04401015
X       -0.42799210    -0.64557336     0.04398764
X        0.42799236    -0.64557223     0.04398572
X        0.00000000     0.00000000     1.08153743
X        0.00000594     1.24101723    -0.36388382
X        0.00017694     1.33159193    -0.91664040
X       -0.00018754     1.76740093    -0.17288215
X        0.00000635    -1.24101725    -0.36388377
X        0.00009866    -1.33159192    -0.91664042
X       -0.00015883    -1.76740094    -0.17288214
He       0.00043578    -0.00008266     0.36288558
He       0.24241827    -0.00046560     0.41383075
He      -0.10533232    -0.20174488     0.46083188
He      -0.03384378     0.00016500     0.11784279
He      -0.10435560     0.20227982     0.46077388
He       0.00000000     0.00000000     1.08153743
He       0.00010294     1.26144132    -0.37391481
He       0.42798899     0.64558655     0.04401015
He      -0.42799888     0.64554456     0.04393850
He       0.00017694     1.33159193    -0.91664040
He      -0.00018754     1.76740093    -0.17288215
He       0.00013933    -1.26143649    -0.37392422
He       0.42799236    -0.64557223     0.04398572
He      -0.42799210    -0.64557336     0.04398764
He       0.00009866    -1.33159192    -0.91664042
He      -0.00015883    -1.76740094    -0.17288214
```

Example: Solids/extended systems For periodic systems, the input changes slightly with respect to molecules as described in the last paragraph.
First, in the second line of the `system` file right after the specification of the used units (angstrom, bohr), one needs to put the keyword 'pbc'. This will tell the code that a periodic system will be initialized. If such a keyword is not present, a molecular guess will be created. Second, the cell parameters (in the same units as used for the atomic coordinates) need to be specified after the atomic coordinates, see example input (neither all atomic coordinates not the connectivity pattern are shown for simplicity). Besides that, all other inputs are the same as in the previous molecular example.

```
114 UiO-66
angstrom pbc
  H    -10.6245106790     5.8739520000     7.6455778800
  H    -10.3538893190    15.1044479990     7.6455778800
  H    -10.6245106790    15.1044479990    13.3328221190
  H    -10.3538893190     5.8739520000    13.3328221190
  H    -13.3328221190    10.3538893190     5.8739520000
  H    -15.1044479990     7.6455778800    10.3538893190
  .
  .
  .
cell
-10.4891999990  0.0000000000 10.4891999990
  0.0000000000 10.4891999990 10.4891999990
-10.4891999990 10.4891999990  0.0000000000
con_mat
.
.
.
```

As an additional output, the file `Nuc_FOD.cif` will be generated, which is a typical .cif file to visualize the periodic structure.

Examples: Some additional information It should be mentioned that you can store any number of inputs as described above in one and the same `system` file. The MC script will only read the very first entry of that file. Make sure that every entry ends with an empty line. In addition, make sure that the input is on top of the file.

7 Additional features

1. The file `xx_database_xx` contains information such that FODs for any atom can be initialized with small and large-core pseudopotentials (PPs). For large-core PPs, use the tag `_ECP_LC` after the element name, e.g. `N_ECP_LC` for nitrogen (large core PPs are currently unavailable for the 3d metals). For small-core PPs, the tag is currently `_ECP_SC`. For transition metals, the configuration (i.e. for 3d metals whether the $4s^1$ or the $4s^2$ configuration shall be used) needs to be additionally added, e.g. `Ni_ECP_SC_4s2` for a Ni with a $3d^8 4s^2$ electronic configuration.
2. Input units can be angstrom (Å) or bohr. The output files are adjusted accordingly (`CLUSTER` and `FRMORB` are always given in Bohr, while the `Nuc_FOD.xyz` is always in angstrom).
3. Inputs require the specification of the bond centers and number of bond-FODs for all bonds. In addition, the number of lone FODs for all atoms (except H) are needed.
4. NO input need for single bonds. All bonds within a molecule are initialized as single bonds. Thus, only bonds which are different from single bonds need to be specified.
5. Lone FODs are placed automatically, asssuming charge neutrality of the atoms.

Versions

Version 1 (November 8, 2018) Remove the need for input lines regarding H atoms

Version 2 (January 21, 2019) Input pattern changed. Now, the actual bond pattern, i.e. information per bond, needs to be provided in addition to the lone FODs. This makes the input more intuitive and simpler (less input needed). Lone FODs are now placed automatically as well assingle bonds.

(February 11, 2019) Heaps of smaller updates regarding input handling, placing of single bonds and placing of lone FODs.

Version 3 (March 18, 2019) Implementation of periodic boundary conditions

Contact information For any comments, remarks, question etc., please contact Kai Trepte from Central Michigan University (group of Prof. Koblar Alan Jackson) via trept1k@cmich.edu .

References

[1] W. L. Luken. Properties of the Fermi Hole in Molecules. *Croatica Chemica Acta*, **57**:1283–1294, 1984.

[2] J. W. Linnett. A Modification of the Lewis-Langmuir Octet Rule. *Journal of the American Chemical Society*, **83**:2643–2653, 1961.

Fractional coordinates

Given a cartesian coordinate $\mathbf{r}_{\text{cart}} = (x,y,z)^{\text{T}}$ of any atom/FOD in a unit cell described by the vectors **a**, **b**, **c**, getting to the fractional coordinates is rather straightforward. One can define a matrix containing the unit cell vectors as

$$\mathbf{A}_{ij} = \begin{pmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \end{pmatrix}.$$

To obtain the fractional coordinates from the cartesian ones, one can use

$$\mathbf{r}_{\text{frac}} = \mathbf{r}_{\text{cart}} \cdot \mathbf{A}_{ij}^{-1}, \tag{8}$$

where $\mathbf{A}_{ij}^{-1}$ is the inverse of the introduced matrix. This inverse matrix can be written as

$$\mathbf{A}_{ij}^{-1} = \frac{1}{\det(\mathbf{A}_{ij})} \begin{pmatrix} b_y c_z - b_z c_y & a_z c_y - a_y c_z & a_y b_z - a_z b_y \\ b_z c_x - b_x c_z & a_x c_z - a_z c_x & a_z b_x - a_x b_z \\ b_x c_y - b_y c_x & a_y c_x - a_x c_y & a_x b_y - a_y b_x \end{pmatrix}.$$

Here, the determinant of the matrix $\mathbf{A}_{ij}$ is

$$\det(\mathbf{A}_{ij}) = a_x b_y c_z + a_z b_x c_y + a_y b_z c_x - a_z b_y c_x - a_x b_z c_y - a_y b_x c_z. \tag{9}$$

This formalism can be used for any unit cell. To transfer the fractional coordinates to the cartesian ones, one can simply invert the process by

$$\mathbf{r}_{\text{cart}} = \mathbf{r}_{\text{frac}} \cdot \mathbf{A}_{ij}. \tag{10}$$