EPLF

Electron Pair Localization Function: User's Guide
May, 2010

Anthony Scemama

1 What is EPLF?

1.1 QMC Formulation

The EPLF[1] has been designed to describe local electron pairing in molecular systems. It is defined as a scalar function defined in the three-dimensional space and taking its values in the [-1,1] range. It is defined as follows:

$$\mathrm{EPLF}(\vec{r}) = \frac{d_{\sigma\sigma}(\vec{r}) - d_{\sigma\bar{\sigma}}(\vec{r})}{d_{\sigma\sigma}(\vec{r}) + d_{\sigma\bar{\sigma}}(\vec{r})}$$

where the quantity $d_{\sigma\sigma}(\vec{r})$ $[resp. d_{\sigma\bar{\sigma}}(\vec{r})$] denotes the quantum-mechanical average of the distance between an electron of spin σ located at $\vec{r}$ and the closest electron of same spin (resp., of opposite spin $\bar{\sigma}$).

The mathematical definition of these quantities can be written as

$$d_{\sigma\sigma}(\vec{r}) = \int \Psi^2(\vec{r}_1, \ldots, \vec{r_N}) \left[\sum_{i=1}^{N} \delta(\vec{r} - \vec{r}_i) \min_{j \neq i; \sigma_j = \sigma_i} |\vec{r}_i - \vec{r}_j| \right] d\vec{r}_1 \ldots d\vec{r}_N$$

$$d_{\sigma\bar{\sigma}}(\vec{r}) = \int \Psi^2(\vec{r}_1, \ldots, \vec{r}_N) \left[\sum_{i=1}^{N} \delta(\vec{r} - \vec{r}_i) \min_{j \neq i; \sigma_j \neq \sigma_i} |\vec{r}_i - \vec{r}_j| \right] d\vec{r}_1 \ldots d\vec{r}_N$$

where σ is the spin (\alpha or \beta), $\bar{\sigma}$ is the spin opposite to σ $\Psi(\vec{r}_1, \ldots, \vec{r}_N)$ is the wave function, and N is the number of electrons.

In a region of space, if the shortest distance separating anti-parallel electrons is smaller than the shortest distance separating electrons of same spin, the EPLF takes positive values and indicates pairing of anti-parallel electrons. In contrast, if the shortest distance separating anti-parallel electrons is larger than the shortest distance separating electrons of same spin, the EPLF takes negative values and indicates pairing of parallel electrons (which in practice never happens). If the shortest distance separating anti-parallel electrons is equivalent to the shortest distance separating electrons of same spin, the EPLF takes values close to zero and indicates no electron pairing.

The original formulation of EPLF is extremely easy to compute in the quantum Monte Carlo framework. However, it is not possible to compute it analytically due to the presence of the min function in the definitions of $d_{\sigma\sigma}$ and $d_{\sigma\bar{\sigma}}$.

1.2 Analytical Formulation

1.2.1 The EPLF operator

To evaluate exactly the average of the min function is not possible. We proposed[2] to introduce a representation of this function in terms of gaussians and to construct our d-functions in a slightly different way so that the gaussian contributions can be exactly integrated out. The representation of the min function we are considering here is written as

$$\min_{j \neq i} |\vec{r}_i - \vec{r}_j| = \lim_{\gamma \to \infty} \sqrt{-\frac{1}{\gamma} \ln\left(\sum_{j \neq i} e^{-\gamma |\vec{r}_i - \vec{r}_j|^2}\right)}$$

[1] "Electron pair localization function, a practical tool to visualize electron localization in molecules from quantum Monte Carlo data", A. Scemama, P. Chaquin and M. Caffarel, *J. Chem. Phys.* **121**, 1725-1735 (2004).

[2] "Structural and optical properties of a neutral Nickel bisdithiolene complex. Density Functional versus ab-initio methods.", F. Alary, J.-L. Heully, A. Scemama, B. Garreau-de-Bonneval, K. I. Chane-Ching and M. Caffarel, *Theoretical Chemistry Accounts*, DOI: 10.1007/s00214-009-0679-9

and we propose to define the new modified average-distances as

$$d_{\sigma\sigma}(\vec{r}) \sim \lim_{\gamma\to\infty} \sqrt{-\frac{1}{\gamma}\ln \int \Psi^2(\vec{r}_1,\ldots,\vec{r}_N) \sum_{i=1}^{N} \delta(\vec{r}-\vec{r}_i) \sum_{j\neq i;\sigma_i=\sigma_j}^{N} e^{-\gamma|\vec{r}_i-\vec{r}_j|^2} d\vec{r}_1 \ldots d\vec{r}_N}$$

$$d_{\sigma\bar{\sigma}}(\vec{r}) \sim \lim_{\gamma\to\infty} \sqrt{-\frac{1}{\gamma}\ln \int \Psi^2(\vec{r}_1,\ldots,\vec{r}_N) \sum_{i=1}^{N} \delta(\vec{r}-\vec{r}_i) \sum_{j\neq i;\sigma_i\neq\sigma_j}^{N} e^{-\gamma|\vec{r}_i-\vec{r}_j|^2} d\vec{r}_1 \ldots d\vec{r}_N}$$

Note that these new definitions of the average-distances are different from the previous ones essentially because $\ln\langle X\rangle \neq \langle \ln X\rangle$ where X denotes the sum $\sum_j e^{-\gamma|\vec{r}-\vec{r}_j|^2}$ and the symbol $\langle\ldots\rangle$ denotes the quantum average. Remark that the equality is almost reached when the fluctuations of X are small. In our applications it seems that we are not too far from this regime and we have thus systematically observed that both definitions of the d-functions lead to similar EPLF landscapes.

Now, introducing

$$f^{\gamma}_{\sigma\sigma}(\vec{r}) = \sum_{i=1}^{N} \delta(\vec{r}-\vec{r}_i) \sum_{j\neq i;\sigma_i=\sigma_j}^{N} e^{-\gamma|\vec{r}_i-\vec{r}_j|^2}$$

$$f^{\gamma}_{\sigma\bar{\sigma}}(\vec{r}) = \sum_{i=1}^{N} \delta(\vec{r}-\vec{r}_i) \sum_{j;\sigma_i\neq\sigma_j}^{N} e^{-\gamma|\vec{r}_i-\vec{r}_j|^2}$$

we obtain

$$\mathrm{EPLF}(\vec{r}) = \frac{\sqrt{-\frac{1}{\gamma}\ln\langle\Psi|f^{\gamma}_{\sigma\sigma}(\vec{r})|\Psi\rangle} - \sqrt{-\frac{1}{\gamma}\ln\langle\Psi|f^{\gamma}_{\sigma\bar{\sigma}}(\vec{r})|\Psi\rangle}}{\sqrt{-\frac{1}{\gamma}\ln\langle\Psi|f^{\gamma}_{\sigma\sigma}(\vec{r})|\Psi\rangle} + \sqrt{-\frac{1}{\gamma}\ln\langle\Psi|f^{\gamma}_{\sigma\bar{\sigma}}(\vec{r})|\Psi\rangle}}$$

1.2.2 Single determinant wave functions

$$\Psi(r_1,\ldots,r_N) = D_0 = |\phi_1,\ldots,\phi_N|$$

$$\langle\Psi|f^{\gamma}_{\sigma\sigma}(\vec{r})|\Psi\rangle = \sum_{\sigma=\alpha,\beta} \sum_{i=1}^{N_\sigma} \sum_{j\neq i}^{N_\sigma} \int \phi_i(\vec{r})\phi_j(\vec{r'}) e^{-\gamma|\vec{r'}-\vec{r}|^2} \phi_i(\vec{r})\phi_j(\vec{r'}) d\vec{r'}$$

$$- \int \phi_i(\vec{r})\phi_j(\vec{r'}) e^{-\gamma|\vec{r'}-\vec{r}|^2} \phi_j(\vec{r})\phi_i(\vec{r'}) d\vec{r'}$$

$$\langle\Psi|f^{\gamma}_{\sigma\bar{\sigma}}(\vec{r})|\Psi\rangle = \sum_{\sigma=\alpha,\beta} \sum_{i=1}^{N_\sigma} \sum_{j=1}^{N_{\bar{\sigma}}} \int \phi_i(\vec{r})\phi_j(\vec{r'}) e^{-\gamma|\vec{r'}-\vec{r}|^2} \phi_i(\vec{r})\phi_j(\vec{r'}) d\vec{r'}$$

1.3 Multi-determinant wave functions

$$\Psi = \sum_k c_k D_k$$

$$\langle \Psi | f(\vec{r}) | \Psi \rangle = \sum_k \sum_l c_k c_l \langle D_k | f(\vec{r}) | D_l \rangle$$

$$\langle D_k | f^{\gamma}_{\sigma\sigma}(\vec{r}) | D_l \rangle = \sum_{\sigma=\alpha,\beta} \sum_{i=1}^{N_\sigma} \sum_{j\neq i}^{N_\sigma} \int \phi_i^k(\vec{r}) \phi_j^k(\vec{r'}) e^{-\gamma|\vec{r'}-\vec{r}|^2} \phi_i^l(\vec{r}) \phi_j^l(\vec{r'}) d\vec{r'}$$

$$- \int \phi_i^k(\vec{r}) \phi_j^k(\vec{r'}) e^{-\gamma|\vec{r'}-\vec{r}|^2} \phi_j^l(\vec{r}) \phi_i^l(\vec{r'}) d\vec{r'}$$

$$\langle D_k | f^{\gamma}_{\sigma\bar{\sigma}}(\vec{r}) | D_l \rangle = \sum_{\sigma=\alpha,\beta} \sum_{i=1}^{N_\sigma} \sum_{j=1}^{N_{\bar{\sigma}}} \int \phi_i^k(\vec{r}) \phi_j^k(\vec{r'}) e^{-\gamma|\vec{r'}-\vec{r}|^2} \phi_i^l(\vec{r}) \phi_j^l(\vec{r'}) d\vec{r'}$$

1.3.1 Expression of the gamma parameter

In our definition of the modified EPLF the parameter γ is supposed to be (very) large. In practice, if it is chosen too small, the integrals $\langle \Psi | f^{\gamma}_{\sigma\bar{\sigma}}(\vec{r}) | \Psi \rangle$ can take values larger than 1 in regions of high density, giving a negative value for the shortest average distance. If it is chosen too large numerical instabilities appear since the values of the integrals become extremely small. We propose to use a value of γ which depends on the electron density $\rho(\vec{r})$ as:

$$\gamma(\vec{r}) = (-\ln \epsilon) \left(\frac{1}{N} \frac{4}{3} \pi \rho(\vec{r}) \right)^{2/3}$$

where ϵ is the smallest possible floating point number that can be represented with 64 bits ($\sim 2.10^{-308}$). The reason for this choice is the following. The largest possible value of the approximate shortest distance can be computed as

$$d_{\max} = \sqrt{-\frac{1}{\gamma} \ln \epsilon}$$

If the density is conidered constant in a sphere of radius $d_{\max}$, the number N of electrons contained in the sphere is

$$N = \frac{4}{3} \pi d_{\max}^3 \rho(r)$$

The maximum possible distance can be parametrized by a target constant number N of electrons in the sphere

$$d_{\max}(r) = \left(\frac{1}{N} \frac{4}{3} \pi \rho(r) \right)^{-1/3}$$

If N is chosen small enough (0.01 electron), the value of $\langle \Psi | f(r) | \Psi \rangle$ is very unlikely to exceed 1, and the value of γ will also be sufficiently large in the valence regions.

2 Structure of the EPLF code

The EPLF code is written using the IRPF90 environment[1], which needs Python>2.3. IRPF90 can be downloaded from http://irpf90.ups-tlse.fr

The input/output data are handled by the Eazy Fortran I/O library generator (EZFIO), which can be downloaded from http://ezfio.sourceforge.net. In this way, the data needed by the Fortran code is accessible both by Fortran an Python using the same API. The computational part of EPLF is written in IRPF90 and the user interfaces are written in Python.

[1] "IRPF90: a programming environment for high performance computing" A. Scemama, arXiv:0909.5012v1 [cs.SE] (2009)

3 Using the EPLF program

In theory, the EPLF can be computed from any kind of wave function (but note that the present code is written in the Restricted Hartree-Fock framework). The first step is to calculate a wave function using a quantum chemistry code. Then, the parameters of the wave function need to be saved in the EZFIO database in order to be communicated to the EPLF fortran program.

3.1 Wave function preparation

Wave functions calculated using Gaussian, Molpro and GAMESS can be read from the output files. A major constraint is to realize a *single point* calculation, all the following quantities appearing in the output file:

- The basis set
- The full set of MOs
- The coefficients of the Slater determinant expansion for CI wave functions.

3.1.1 Using Gaussian

In the Gaussian input file, use the keywords `GFPRINT` and `pop=Full`. In the case of CAS-SCF wave functions, use the `#p` keyword and the `SlaterDet` attribute of the `CAS` keyword. When doing a CAS-SCF with Gaussian, first do the Hartree-Fock calculation saving the checkpoint file and then do the CAS-SCF in a second step.

3.1.2 Using Molpro

Use the following options in the Molpro input file:

- `print,basis;`
- `gprint,civector;`
- `gprint,orbital;`
- `gthresh,printci=0.;` for MCSCF calculations

An RHF calculation is mandatory before any MCSCF calculation, since some information is printed only the RHF part. Be sure to print *all* molecular orbitals using the `orbprint` keyword, and to use the same spin multiplicity and charge between the RHF and the CAS-SCF .

3.1.3 Using GAMESS

For MCSCF calculations, first compute the MCSCF single-point wave function with the GUGA algorithm. Then, put the the MCSCF orbitals (of the `.dat` file) in the GAMESS input file, and run a single-point GUGA CI calculation with the following keywords:

- `PRTTOL=0.0001` in the `$GUGDIA` group to use a threshold of 10^{-4} on the CI coefficients
- `NPRT=2` in the `$CIDRT` group to print the CSF expansions in terms of Slater determinants
- `PRTMO=.T.` in the `$GUESS` group to print the molecular orbitals

3.1.4 Using your own code

Any other code producing a wave function can be used, as long as you are able to gather all the needed data.

3.2 Building the EZFIO database

An EZFIO database is a directory which contains all the needed input/output data needed by the code.

3.2.1 Using the provided wrapper

The output file from Molpro, GAMESS or Gaussian has to be transformed into an EZFIO database in order to be used by EPLF . This step is realized by calling the `to_ezfio.exe` command, followed by the name of the file containing the wave function:

```
to_ezfio.exe test.out
```

An EZFIO database is then produced, named '`test.out.ezfio`'.

3.2.2 Using your own code

Refer to the EZFIO web site[1] to learn how to use the EZFIO API to collect your data into an EZFIO database. The EZFIO definition file is the '`eplf.config`' file, located at the root of the EPLF package.

The EPLF code is able to compute the following functions on a regular three-dimensional grid:

- The electron pair localization function (EPLF)
- The electron localization function (ELF)
- The electron density

The gradients, the norm of the gradient and the laplacian of any of these three functions can also be requested. Note that the EPLF can only be calculated for a single Slater determinant.

3.3 Using the provided interface

The '`eplf_edit.py`' script allows to modify easily the conditions of the calculation: the choice of the functions to compute, the dimension of the grid, etc. Running the following command

```
eplf_edit.py test.out.ezfio
```

opens an input file in your default editor (defined by the `$EDITOR` environment variable). Lines starting with a `#` character are commented. The input file is separated in three sections.

3.3.1 The Computation section

If you are in a parallel environment, the first line of this section is

```
nproc = 0
```

which defines the number of processors to use. If nproc is chosen equal to 0, then the batch queuing system may choose itself the proper number of processors. Otherwise, specify the number of processors you want to use.

Then, all the computable functions appear, preceded by empty parentheses. Put an `X` between the parentheses for the functions you want to compute, for example

```
(X) elf
( ) density
(X) eplf
```

3.3.2 The Edit section

You can edit the nuclear coordinates and/or the parameters of the grid by un-commenting the `edit(geometry)` or `edit(grid_parameters)` statements.

When you save and quit the current input file, a new file will be open containing the parameters to edit. In the case of the grid parameters, as there is some redundance between the possible inputs quantities, the `Default` keyword can be used to replace the (x,y,z) specification.

[1] http://ezfio.sourceforge.net

```
##########################
# Grid : test.out.ezfio
##########################

# Number of points along x, y, z
40 40 40

# Coordinates of the origin (x,y,z)
-3.000000 -4.744648 -5.322027

# Coordinates of the opposite point (x,y,z)
Default

# Step sizes (x,y,z)
Default
```

3.3.3 The Clear section

All the previously calculated grids are saved in the EZFIO database. If the grid parameters are changed, these grids are inconsistent. Therefore, it may be necessary to clear the previously calculated grids. This can be realized by un-commenting the `clear(*)` statements. The `clear(all)` statement clears all the grids, and also the grid parameters.

3.3.4 Saving the input data

When the editor is closed, after saving the file, all the input data is saved into the EZFIO database. It can be modified later by running again the '`eplf_edit.py`' script. A shell script '`run_test.out.ezfio.sh`' is automatically created to launch the calculation.

3.4 Running the calculation

To run the calculation, run the `./run_test.out.ezfio.sh` command, or place it in a submission script of a batch queuing system.

Once the calculation is done, the grid is saved into the EZFIO database. It can be converted to a cube file using the `to_cube` tool, specifying as a first argument the EZFIO database, and which grid to convert as a second argument. For example:

```
to_cube test.out.ezfio eplf
```

produces the file '`test.out.ezfio_eplf.cube`'