The ddCOSMO module's reference guide

Filippo Lipparini

October 1, 2015

In this brief document, the ddCOSMO implementation will be described and some directions will be provided on how to interface the module with an existing code for classical, quantum and hybrid levels of theory.

Before reading this guide, the user is encouraged to read the following references on the implementation:

- **on ddCOSMO itself**: F. Lipparini, B. Stamm, E. Cancès, Y. Maday, B. Mennucci, *J. Chem. Theory Comput* **9**, 3637−3648 (2013)
- **on interfacing ddCOSMO with various levels of theory**: F. Lipparini, G. Scalmani, L. Lagardère, B. Stamm, E. Cancès, Y. Maday, J.-P. Piquemal, M. J. Frisch, B. Mennucci, *J. Chem. Phys* **141**, 184108 (2014)

This guide provides only a superficial view of ddCOSMO, but provides complementary details regarding the coupling of ddCOSMO with another code.

1 Overview of ddCOSMO

ddCOSMO is an algorithm to solve the COSMO polarization equation in a strategy based on domain decomposition and to compute the electrostatic solvation energy and, if needed, its derivatives.

We will assume in the following that the solute is a molecule composed by M atoms and that it is accommodated into a van der Waals cavity $\Omega = \bigcup_{j=1}^{M} \Omega_j$, where Ω_j is the j-th sphere centered at the j-th atom. There are three important parameters controlling the ddCOSMO discretization:

- N: maximum degree of the spherical harmonics expansion of the local solutions
- N_g: number of points used for numerical integration (using a Lebedev grid $\{\mathbf{y}_n, w_n\}$)
- η: regularization parameter

The ddCOSMO linear system of equations reads:

$$LX = g, \tag{1}$$

where L is the ddCOSMO block-sparse matrix, made by M^2 blocks (the majority of which is zero) of size $(N+1)^2 \times (N+1)^2$, X is a vector that collects, for each local solution X_j, its $(N+1)^2$ expansion coefficients $[X_j]_l^m$ and g is the right-hand side, that depends linearly on the electrostatic potential produced by the solute at the cavity, given by the coefficients

$$[g_j]_l^m = \sum_{n=1}^{N_g} w_n Y_l^m(\mathbf{y}_n) U_n^j \Phi_n^j, \tag{2}$$

on the j-th sphere, where Φ_n^j is the solute's potential at the n-th grid point on the $j-th$ sphere. The ddCOSMO energy is then computed as

$$E_s = \frac{1}{2} f(\varepsilon) \sum_{j=1}^{M} \sum_{l=0}^{N} \sum_{m=-l}^{l} [\Psi_j]_l^m [X_j]_l^m = \frac{1}{2} f(\varepsilon) \langle \Psi, X \rangle, \tag{3}$$

where ε is the dielectric constant of the solvent, $f(\varepsilon) = \frac{\varepsilon - 1}{\varepsilon}$ is a scaling function and the Ψ vector is a quantity whose definition depends on the level of theory used to describe the solute and that will discussed fully in the following. We will write

$$\sum_{lm} := \sum_{l=0}^{N} \sum_{m=-l}^{l}, \quad \sum_{n} := \sum_{n=1}^{N_g}, \quad \sum_{j} := \sum_{j=1}^{M}$$

for brevity.

Notice that the L matrix is not symmetric, which implies that the ddCOSMO energy is not a variational quantity: in order to compute its derivatives the adjoint equation

$$L^* S = \Psi \tag{4}$$

has to be solved as well.

The ddCOSMO Fock matrix contribution can be computed as the derivative of the ddCOSMO energy with respect to the density matrix (in the atomic orbitals basis):

$$F_{\mu\nu}^{s} = \frac{\partial E_s}{\partial P_{\mu\nu}} = \frac{1}{2} f(\varepsilon) \left[\langle \frac{\partial \Psi}{\partial P_{\mu\nu}}, X \rangle + \langle \Psi, \frac{\partial X}{\partial P_{\mu\nu}} \rangle \right]. \tag{5}$$

The first term will be discussed later; the second term can be rearranged as follows:

$$F_{\mu\nu}^{s,2} := \langle \Psi, \frac{\partial X}{\partial P_{\mu\nu}} \rangle = \langle S, \frac{\partial g}{\partial P_{\mu\nu}} \rangle = \sum_{j} \sum_{lm} [S_j]_l^m \sum_{n} w_n Y_l^m(\mathbf{y}_n) U_n^j \frac{\partial \Phi_n^j}{\partial P_{\mu\nu}}.$$

By introducing the quantity

$$\zeta_n^j = \sum_{lm} w_n U_n^j Y_l^m(\mathbf{y}_n) [S_j]_l^m \tag{6}$$

we get:

$$F_{\mu\nu}^{s,2} = \sum_{j} \sum_{n} \zeta_n^j \frac{\partial \Phi_n^j}{\partial P_{\mu\nu}}. \tag{7}$$

Finally, the nuclear gradients can be computed as follows:

$$\frac{dE_s}{dx} = E^x = \frac{1}{2}f(\varepsilon)\langle\Psi, X\rangle^x = \frac{1}{2}f(\varepsilon)\left[\langle S, g^x\rangle - \langle S, (L^x)X\rangle + \langle\Psi^x, X\rangle\right] \tag{8}$$

The second term can be computed starting from the solution to eqs. 1 and 4 plus various ddCOSMO related quantities; the first involves the derivatives of the solute's potential:

$$\langle S, g^x\rangle = \sum_j \sum_{lm} \sum_n \left(w_n Y_l^m(\mathbf{y}_n)[S_j]_l^m (U_n^j)^x \Phi_n^j + w_n Y_l^m(\mathbf{y}_n)[S_j]_l^m U_n^j (\Phi_n^j)^x \right).$$

Again the first term requires one to compute ddCOSMO-related quantities, while the second requires the derivatives of the potential. By introducing again the vector ζ defined in eq. 6, the second term becomes:

$$\sum_j \sum_n \zeta_j^n (\Phi_n^j)^x. \tag{9}$$

In conclusion, to run a ddCOSMO calculation, the user needs to be able to provide the following quantities:

1. Coordinates of the solute and the associated van der Waals radii
2. Electrostatic potential of the solute at the grid points on each spheres (indeed, only at the external points, as it will be made clear in the following)
3. Ψ vector
4. For the Fock Matrix, the derivatives of the electrostatic potential and of Ψ with respect to the density matrix
5. For the forces, the derivatives of Ψ and of the electrostatic potential with respect to the nuclear coordinates.

All these terms are discussed in great detail in the second reference given at the beginning of the session, in the next session some operational detail will be provided.

2 Quantities to be provided by the user

2.1 Electrostatic potential

In order to compute the ddCOSMO electrostatic solvation energy, two quantities need to be assembled. The first one is the **electrostatic potential Φ**

```
real*8, dimension(ngrid,nsph) :: phi
```

on a grid of points:

$$\Phi_n^j = \Phi(\mathbf{R}_j + r_j \mathbf{y}_n) = \int_\Omega d\mathbf{x} \frac{\rho(\mathbf{x})}{|\mathbf{R}_j + r_j \mathbf{y}_n - \mathbf{x}|}, \qquad (10)$$

where $\mathbf{R}_j$ is the center of the j-th sphere. The electrostatic potential should be a standard one-electron integral in any quantum-mechanical code or is easily assembled for classical solutes. The Fast Multipole Method can be used to achieve linear scaling for such computation. The ddCOSMO initialization routine will provide an array of `ncav` points at which the potential is to be computed, which correspond to the list of grid points on the various spheres for which the function U_n^j is nonzero. This list of points

```
real*8, dimension(3,ncav) :: ccav
```

can be directly fed to the appropriate integral routine. We will refer to this set as "external cavity points".

2.2 Ψ vector

The second quantity is the Ψ **vector**

```
real*8, dimension((n+1)**2,nsph) :: psi
```

such a quantity is defined starting from the expression for the energy in the ddCOSMO paradigm:

$$E^s = \frac{1}{2} f(\varepsilon) \int_\Omega \rho(\mathbf{x}) W(\mathbf{x}) d\mathbf{x} \qquad (11)$$

The reaction potential W can be expressed as follows:

$$\forall \mathbf{x} \in \Omega_j; \quad W(\mathbf{x}) = \sum_{lm} \frac{4\pi}{2l+1} \frac{x_<^l}{x_>^{l+1}} [X_j]_l^m Y_l^m(\mathbf{s}), \qquad (12)$$

where $x = \|\mathbf{x}\|$ and $\mathbf{s} = \mathbf{x}/x$. Also, we introduced the notation $x_<$ and $x_>$, which refer to the smallest and largest among x and r_j, the van der Waals radius of the j-th sphere. Eq. 12 suggests to write the ddCOSMO energy as a sum of single sphere contributions, which can be done trivially for classical solutes as the density is given by a collection of atom-centered multipoles, but requires some caution for QM descrived solutes, as the density is a continuous function and double counting in the spheres intersections has to be avoided. This can be done with a multicentric numerical integration scheme such as the ones used in density function theory calculations: such schemes take into account the overlap between atoms by defining the proper weights. Let $\{\mathbf{x}_p^j, \tau_p^j\}_{p=1}^{N_g^b}$ be the points and weights associated to the j-th atom (and hence, to the j-th sphere) in such a quadrature scheme: we can rewrite the integral in eq. 11 as

$$\int_\Omega \rho(\mathbf{x}) W(\mathbf{x}) d\mathbf{x} \sim \sum_{j=1}^{M} \sum_{p=1}^{N_g^b} \tau_p^j W(\mathbf{x}_p^j) \rho(\mathbf{x}_p^j).$$

By substituting the reaction potential and collecting the various terms together we get:

$$E^s = \frac{1}{2} f(\varepsilon) \sum_j \sum_{lm} [X_j]_l^m \frac{4\pi}{2l+1} \sum_p \tau_p^j \rho(\mathbf{x}_p^j) \frac{x_<^l}{x_>^{l+1}} Y_l^m(\mathbf{s}_p^j), \tag{13}$$

where we used the same notation introduced before replacing $\mathbf{x}$ with $\mathbf{x}_p^j$. Notice that the grid points might be outside the sphere Ω_j which is in practice not a problem. We refer to the second reference for a more detailed discussion. By comparing eq. 13 with eq. 3, one gets:

$$[\Psi_j]_l^m = \frac{4\pi}{2l+1} \sum_p \tau_p^j \rho(\mathbf{x}_p^j) \frac{x_<^l}{x_>^{l+1}} Y_l^m(\mathbf{s}_p^j). \tag{14}$$

Therefore, the Ψ vector can be obtained via a numerical integration similar to the ones used to evaluate exchange-correlation functionals.

As already mentioned, the expression for Ψ is greatly simplified for classical solutes. For a classical force field where the electrostatics is model by point charges, one simply gets:

$$[\Psi_j]_l^m = \sqrt{4\pi} q_j \delta_{l0} \delta_{m0}. \tag{15}$$

2.3 Fock Matrix

In order to compute the Fock matrix contributions, after solving the ddCOSMO equations 1 and their adjoint 4, the user needs to be able to compute the derivatives of the electrostatic potential at the external cavity points and contract it with the vector ζ (which is nonzero only at the external cavity points, as it contains the U function). The electronic density can be expanded in eq. 10 in terms of the atomic orbitals (we omit the obvious nuclear contribution):

$$\Phi_n^j = -\sum_{\mu\nu} P_{\mu\nu} \int_\Omega d\mathbf{x} \frac{\chi_\mu(\mathbf{x})\chi_\nu(\mathbf{x})}{|\mathbf{R}_j + r_j \mathbf{y}_n - \mathbf{x}|} =: -\sum_{\mu\nu} P_{\mu\nu} V_{\mu\nu}^{j,n};$$

and the second contribution to the Fock matrix, eq. 7, is therefore:

$$F_{\mu\nu}^{s,2} = \sum_j \sum_n \zeta_j^n V_{\mu\nu}^{j,n}. \tag{16}$$

The $V_{\mu\nu}$ integrals can be generated and directly contracted with the ζ elements, which can be fed to the integral routine, together with the position of the external cavity points.

The first contribution to the fock matrix (see eq. 5) contains the derivatives of Ψ with respect to the density matrix and requires a second numerical integration. It can be assembled by expanding again the density in terms of the atomic orbitals in eq. 14 (we omit the nuclear contributions, which can be treated as for classical solutes) and differentiating:

$$\frac{\partial [\Psi_j]_l^m}{\partial P_{\mu\nu}} = \frac{4\pi}{2l+1} \sum_p \tau_p^j \chi_\mu(\mathbf{x}_p^j)\chi_\nu(\mathbf{x}_p^j) \frac{x_<^l}{x_>^{l+1}} Y_l^m(\mathbf{s}_p^j).$$

The overall contribution to the Fock matrix is obtained by contracting the Ψ derivative with the X coefficients:

$$F_{\mu\nu}^{s,1} = \sum_j \sum_p \tau_p^j \chi_\mu(\mathbf{x}_p^j)\chi_\nu(\mathbf{x}_p^j) \sum_{lm} \frac{4\pi}{2l+1} \frac{x_<^l}{x_>^{l+1}} Y_l^m(\mathbf{s}_p^j)[X_j]_l^m, \tag{17}$$

where of course the sum over l, m can be precomputed, for each atom, before performing the numerical integration.

2.4 Forces

The computation of the forces require the user to provide the derivatives of the electrostatic potential. By differentiating eq. 10 one gets (always neglecting the obvious nuclear contributions):

$$(\Phi_n^j)^x = -\sum_{\mu\nu} P_{\mu\nu} \int_\Omega d\mathbf{x} \left\{ \frac{(\chi_\mu(\mathbf{x})\chi_\nu(\mathbf{x}))^x}{|\mathbf{R}_j + r_j\mathbf{y}_n - \mathbf{x}|} - \frac{\chi_\mu(\mathbf{x})\chi_\nu(\mathbf{x})}{|\mathbf{R}_j + r_j\mathbf{y}_n - \mathbf{x}|^3} (\mathbf{R}_j + r_j\mathbf{y}_n - \mathbf{x})^x \right\} \tag{18}$$

The first term involves the derivatives of the basis functions, and vanishes unless such functions are centered at the atom which the coordinate x belongs to. The second term involves the derivative of the Kernel, and is therefore the electric field produced by the electronic density. Both quantities should be standard in QM codes. The two terms need to be contracted with the ζ vector, as for the Fock matrix, in order to obtain the forces.

For classical solutes, it is easily proved that the two terms, contracted with the ζ vector, can be rewritten as the electric field produced by the MM charges at the external cavity points multiplied by ζ plus the electric field produced by ζ at the nuclei multiplied by the MM charges:

$$\begin{aligned} E_s^x = & \sum_j \sum_n \zeta_n^j \left(\sum_k \frac{q_k}{|\mathbf{R}_j + r_j\mathbf{y}_n - \mathbf{R}_k|} \right) (\mathbf{R}_j + r_j\mathbf{y}_n - \mathbf{R}_k|)^x \\ & + \sum_j q_j \left(\sum_k \sum_n \frac{\zeta_n^j}{|\mathbf{R}_j - (\mathbf{R}_k + r_k\mathbf{y}_n)|^3} [\mathbf{R}_j - (\mathbf{R}_k + r_k\mathbf{y}_n)]^x \right). \end{aligned} \tag{19}$$

For QM solutes, the derivatives of the Ψ vector are also needed. By differentiating eq. 14, one gets two contributions:

$$([\Psi_j]_l^m)^x = \frac{4\pi}{2l+1} \sum_p [(\tau_p^j)^x \rho(\mathbf{x}_p^j) + \tau_p^j \sum_{\mu\nu} P_{\mu\nu} (\chi_\mu(\mathbf{x}_p^j)\chi_\nu((\mathbf{x}_p^j))^x] \frac{x_<^l}{x_>^{l+1}} Y_l^m(\mathbf{s}_p^j).$$

The first contribution involves the derivatives of the integration weights, while the second involves the derivatives of the atomic orbitals. Such quantities should be available in the numerical quadrature routines used for DFT if analytical derivatives are implemented in the code.

3 Content of the ddCOSMO distribution and description of the examples

Six Fortran files are included in the current release:

- ddcosmo.f90: the main ddCOSMO module. Contains all the ddCOSMO specific routines.
- llgrid.f: the Lebedev-Laikov grid routine by D. Laikov and C. van Wuellen, as publicly available on CCL. Please, notice that this work needs to be acknowledge by citing the following reference: V.I. Lebedev, and D.N. Laikov "A quadrature formula for the sphere of the 131st algebraic order of accuracy" *Doklady Mathematics*, **59**, 477-481 (1999)
- main.f90: a commented example of what the interface of ddCOSMO with an existing code should look like for the computation of energy and forces.
- forces.f90: a commented example of what the interface of ddCOSMO with an existing code should look like. This routine is specific for the forces.
- mkrhs.f90: silly routine that assembles the potential at gridpoints and the Ψ vector
- efld.f90: silly routine that assembles the electric field produced by some sources at some (different) target points, used for the computation of the forces.

The last four routines are accessory to the ddCOSMO module and mimic what an user should do in order to interface the module with his/her favorite code. In particular, the routines mkrhs.f90 and efld.f90 should be replaced by the proper routines to compute molecular integrals or electrostatic properties for classical force fields. The main.f90 file contains all the main steps required to set up a ddCOSMO calculations:

1. read or define the ddCOSMO parameters, i.e.:
   - iprint (integer): printing flag that regulates how verbose the output should be
   - nproc (integer): number of openmp threads
   - lmax (integer): angular momentum of the spherical harmonics basis (here called N)
   - ngrid (integer): number of lebedev points (here called N_g)
   - iconv (integer): $10^{-\text{iconv}}$ is the convergence threshold for the iterative solver
   - igrad (integer): flag that is used to determine whether to compute the forces (1) or not (0)
   - eps (real): dielectric constant of the solvent
   - eta (real): regularization parameter (we suggest to use 0.1)

   In the example, such parameters are read from a text file Input.txt.

2. gather geometrical information on the solute (in the provided example, such information is also read from Input.txt). For a classical solute, the coordinates, van der Waals radii and point charge of each atom are needed. For QM solutes, the information needed to call the required integral routines are to be used.

3. call the initialization routine ddinit, which sets up various quantities, allocates memory and assembles the cavity information, including the vector `ccav` of coordinates of the external cavity points and the number of such points, `ncav`.

4. allocate memory for the potential (`ncav`) and for the Ψ vector (`(n+1)**2,nsph`)

5. **compute the potential Φ and Ψ.** This is one of the points that require to be modified by the **user**

6. allocate the vector of coefficients X (`(n+1)**2,nsph`)

7. call the ddCOSMO solver `itsolv`

If only the energy is needed, this concludes the routine. The memory is deallocated (including by calling memfree, which deallocates the ddCOSMO arrays). If the forces are required, the vector of the S coefficients is allocated (`(n+1)**2,nsph`) and the ddCOSMO iterative solver is called again with the "adjoint" flag set to `.true.` . The forces routine is then called. The main.f90 file is largely commented and all the operations performed are described in detail.

The forces.f90 file contains a prototypical driver to compute the various contributions to the ddCOSMO forces and requires to be modified by the user, as the forces require one to assemble various solute-dependent quantities. The following operations need to be performed:

1. Allocate memory for various scratch quantities (see the routine)

2. Assemble an intermediate (vector `xi` in the routine)

3. Compute the ddCOSMO contributions to the forces (terms involving the derivatives of the ddCOSMO matrix and of the U function)

4. Free the memory

5. Allocate memory for the ζ vector and assemble it

6. Allocate memory for the electric field at the external cavity points

7. **Compute the solute's electric field at the external cavity points and contract it with ζ.** This is one of the points that need to be modified by the **user**

8. **Compute the "electric field" produced by ζ at the nuclei and contract it with the nuclear charges**. This is the last point that needs to be modified by the **user**.

9. Free the memory and return.

For QM solutes, the additional computations described in the previous section need to be carried out, too. The forces routine is largely commented and should be used as a template.

4 Bibliography

Please, read and acknowledge the following articles:

- Domain decomposition for implicit solvation models.
  E. Cancès, Y. Maday, B. Stamm
  J. Chem. Phys., **139**, 054111 (2013)
  `http://dx.doi.org/10.1063/1.4816767`
- Fast Domain Decomposition Algorithm for Continuum Solvation Models: Energy and First Derivatives.
  F. Lipparini, B. Stamm, E. Cancès, Y. Maday, B. Mennucci
  J. Chem. Theory Comput., **9**, 3637-3648 (2013)
  `http://dx.doi.org/10.1021/ct400280b`
- Quantum, classical, and hybrid QM/MM calculations in solution: General implementation of the ddCOSMO linear scaling strategy.
  F. Lipparini, G. Scalmani, L. Lagardère, B. Stamm, E. Cancès, Y. Maday, J.-P. Piquemal, M. J. Frisch, B. Mennucci
  J. Chem. Phys., **141**, 184108 (2014)
  http://dx.doi.org/10.1063/1.4901304

Please also acknowledge the following references if you use ddCOSMO in conjunction with a classical (including polarizable) force field or a semi-empirical model or if you want to provide a reference for timings:

- Polarizable Molecular Dynamics in a Polarizable Continuun Solvent
  F. Lipparini, L. Lagardère, C. Raynaud, B. Stamm, E. Cancès, B. Mennucci, M. Schnieders, P. Ren, Y. Maday, J.-P. Piquemal *J. Chem. Theory Comput.*, **11**, 623-634 (2015)
  `http://dx.doi.org/10.1021/ct500998q`
  for classical solutes
- Quantum Calculations in Solution for Large to Very Large Molecules: A New Linear Scaling QM/Continuum Approach.
  F. Lipparini, L. Lagardère, G. Scalmani, B. Stamm, E. Cancès, Y. Maday, J.-P. Piquemal, M. J. Frisch, B. Mennucci
  J. Phys. Chem. Lett. **5**, 953-958 (2014)
  `http://dx.doi.org/10.1021/jz5002506`
  for semi-empirical solutes and timings