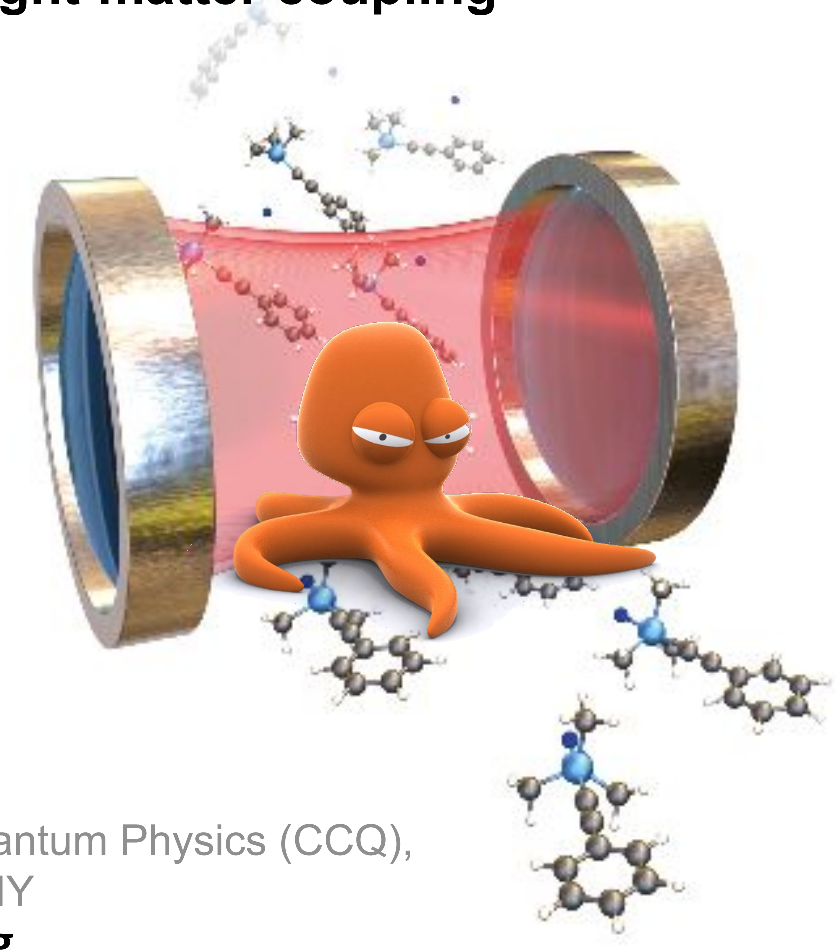
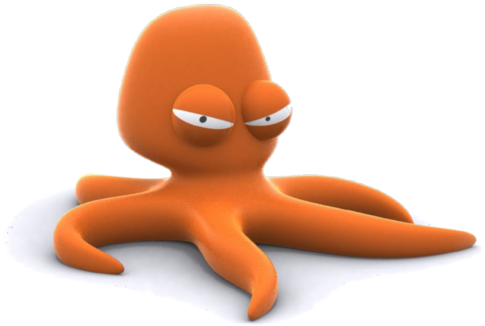
Octopus and strong light-matter coupling



Johannes Flick
Center for Computational Quantum Physics (CCQ),
Flatiron Institute, New York, NY
jflick@flatironinstitute.org

Overview of QEDFT implementations in octopus code



Definition of cavity,
i.e. frequency, coupling strength and cavity polarization

poisson/photon_mode.F90

Ground state:
One photon OEP **(merged)**
system/xc_oep.F90

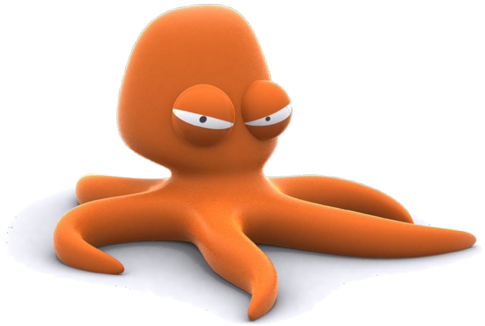
Photon exchange-correlation functionals
poisson/photon_mode_mf.F90

Excited states:
Mean-field time dependent implementation **(merged)**
poisson/photon_mode_mf.F90

Casida equation **(merged)**
main/casida.F90

Linear response for vibro-polaritons (John Bonini)
ions/vibrations.F90 (and others)

Overview of QEDFT implementations in octopus code



Definition of cavity,
i.e. frequency, coupling strength and cavity polarization

poisson/photon_mode.F90

Ground state:
One photon OEP **(merged)**
system/xc_oep.F90

Photon exchange-correlation functionals
poisson/photon_mode_mf.F90

Excited states:
Mean-field time dependent implementation **(merged)**
poisson/photon_mode_mf.F90

Casida equation **(merged)**
main/casida.F90

Linear response for vibro-polaritons (John Bonini)
ions/vibrations.F90 (and others)

Mean-field time dependent implementation: Equations

Dipole moment $$\boldsymbol{\mu} =< \text{multipoles} >= \sum_{i=1}^{n_e} -|e|r_i + \sum_{j=1}^{N_n} Z_j|e|R_j$$

Vks potential: $$v_{MF}(\mathbf{r}_i) = -\sum_{\alpha} \boldsymbol{\lambda}_\alpha \cdot \mathbf{r}_i \left[\omega_\alpha q_\alpha(t) + \boldsymbol{\lambda}_\alpha \cdot \boldsymbol{\mu}(t)\right]$$

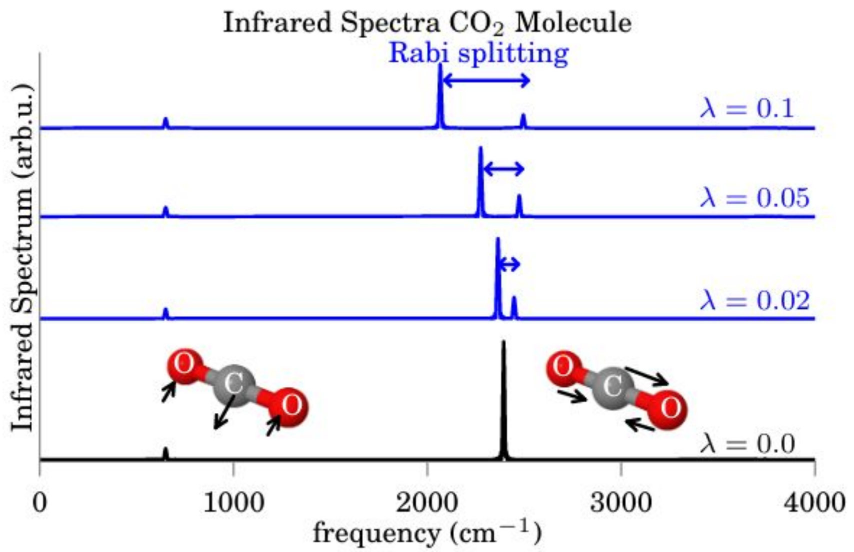
Photon forces on the nuclei $$\mathbf{F}_s^{(I,\beta)}(t) = \sum_{\alpha=1}^{\mathcal{N}} Z_I \omega_\alpha \boldsymbol{\lambda}_\alpha \left(q_\alpha(t) + \frac{\boldsymbol{\lambda}_\alpha}{\omega_\alpha} \cdot \boldsymbol{\mu}(t)\right)$$

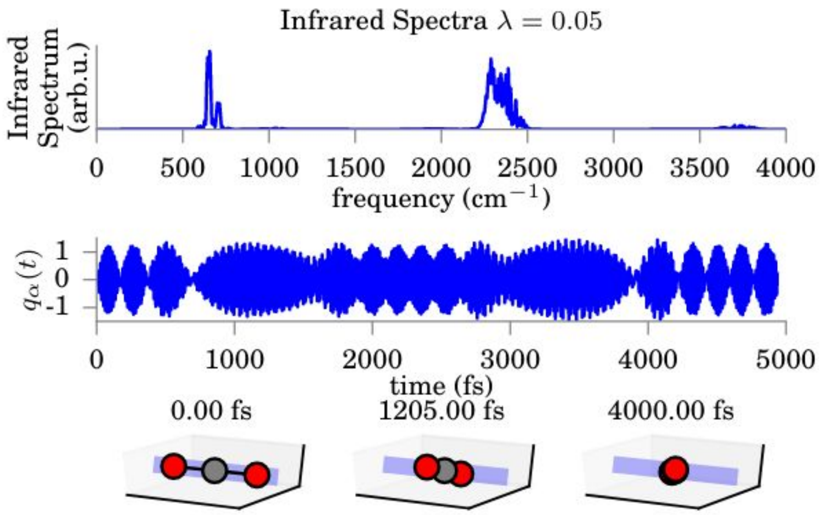
Wave equation $$\partial t^2 q_\alpha(t) + \omega_\alpha^2 q_\alpha(t) = -\omega_\alpha \boldsymbol{\lambda}_\alpha \cdot \boldsymbol{\mu}(t)$$

Explicit solution:
$$q_\alpha(t) = q_\alpha(t_0)\cos(\omega_\alpha t) + \frac{\dot{q}_\alpha(t_0)}{\omega_\alpha}\sin(\omega_\alpha t) - \mathcal{I}m\{e^{i\omega_\alpha t}\int_{t_0}^{t} dt' e^{-i\omega_\alpha t'} \boldsymbol{\lambda}_\alpha \cdot \boldsymbol{\mu}(t')\}$$

J. Flick, P. Narang, Phys. Rev. Lett. 121, 113002 (2018).

Mean-field time dependent implementation





Ehrenfest dynamics for nuclei using the forces due to the photons.
Long propagation times, few pico seconds using X. Andrade et al., JCTC 728-742 (2009).

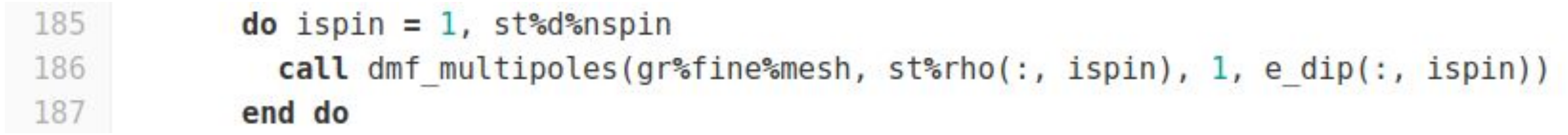
J. Flick, P. Narang, Phys. Rev. Lett. 121, 113002 (2018).

Mean-field time dependent implementation: Sign of dipole moment

$$\boldsymbol{\mu} =< \text{multipoles} >= \sum_{i=1}^{n_e} -|e| r_i + \sum_{j=1}^{N_n} Z_j |e| R_j$$

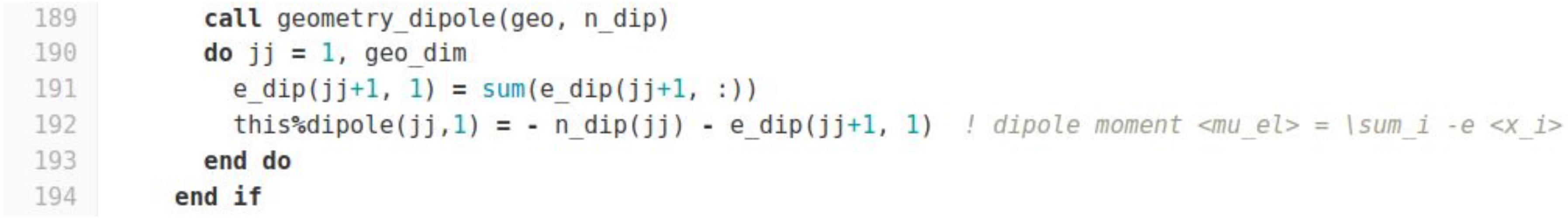
Internally octopus works with the ‘wrong’ sign. Electrons have positive charge.

Electronic part



```
  do ispin = 1, st%d%nspin
    call dmf_multipoles(gr%fine%mesh, st%rho(:, ispin), 1, e_dip(:, ispin))
  end do
```

Nuclear part



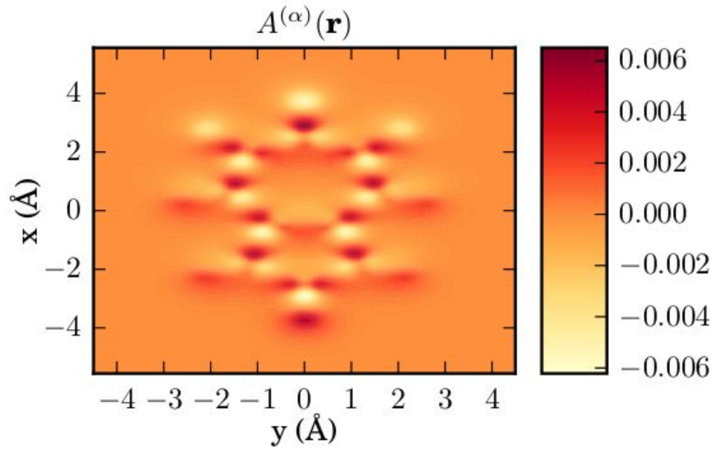
```
  call geometry_dipole(geo, n_dip)
  do jj = 1, geo_dim
    e_dip(jj+1, 1) = sum(e_dip(jj+1, :))
    this%dipole(jj,1) = - n_dip(jj) - e_dip(jj+1, 1)  ! dipole moment <mu_el> = \sum_i -e <x_i>
  end do
end if
```

Currently the mean-field and the OEP implementations work with the ‘correct’ sign.
-> Inconsistent notation with other parts of octopus? How should it be handled?

J. Flick, P. Narang, Phys. Rev. Lett. 121, 113002 (2018).

Simple electron-photon functionals for density-functional theory (QEDFT)

One photon OEP-functional
Connection DFT - MBPT: Sham-Schlueter equation

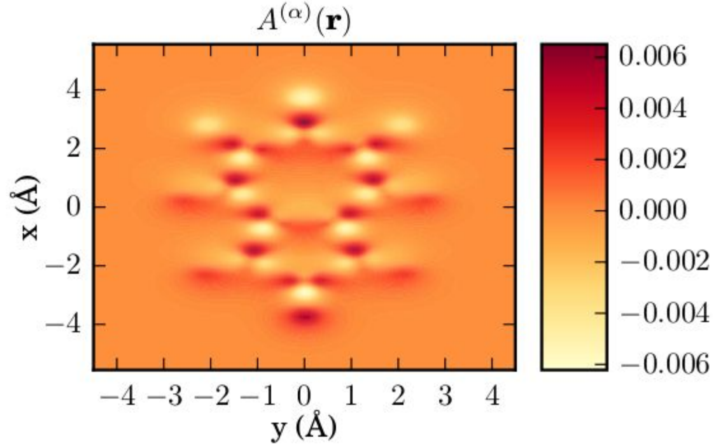


J. Flick, C. Schäfer, M. Ruggenthaler, H. Appel, A. Rubio, ACS Photonics (2018).

J. Flick, arXiv:2104.06980 (2021)

Simple electron-photon functionals for density-functional theory (QEDFT)

One photon OEP-functional
Connection DFT - MBPT: Sham-Schlueter equation



J. Flick, C. Schäfer, M. Ruggenthaler, H. Appel, A. Rubio, ACS Photonics (2018).

Reformulate energy expression in terms of polarizabilities using **fluctuation-dissipation theorem** of QEDFT

$$E_{\mathrm{xc}}^{(2)} = -\frac{1}{\pi}\int_0^\infty d\omega \sum_\alpha \left[\frac{\omega_\alpha^2}{\omega^2+\omega_\alpha^2}+2\right] \boldsymbol{\lambda}_\alpha \cdot \boldsymbol{\alpha}(i\omega)\cdot \boldsymbol{\lambda}_\alpha \tag{12}$$

$$\boldsymbol{\alpha}(i\omega) = -2\int d\mathbf{r}\int d\mathbf{r}' \sum_{ia} \frac{(\epsilon_a-\epsilon_i)\,\varphi_a(\mathbf{r})\mathbf{r}\varphi_i(\mathbf{r})\varphi_i(\mathbf{r}')\mathbf{r}'\varphi_a(\mathbf{r}')}{(\epsilon_a-\epsilon_i)^2+\omega^2} \tag{13}$$

Approximate polarizability using density functionals (dispersion functionals, e.g.)

$$\alpha(\mathbf{r}, iu) = \frac{1}{4\pi}\frac{\omega_p^2(\mathbf{r})}{\omega_p^2(\mathbf{r})/3+\omega_g^2(\mathbf{r})+u^2},$$

O.A. Vydrov, T. Van Voorhis, Phys. Rev. A 81, 062708 (2010)

plasmon frequency $\omega_p^2(\mathbf{r}) = 4\pi n(\mathbf{r})e^2/m$

gap frequency $\omega_g^2(\mathbf{r}) = C\,\frac{\hbar^2}{m^2}\left|\frac{\nabla n(\mathbf{r})}{n(\mathbf{r})}\right|^4$

J. Flick, arXiv:2104.06980 (2021)

Simple electron-photon functionals for density-functional theory (QEDFT)

Through explicit integration over frequencies:

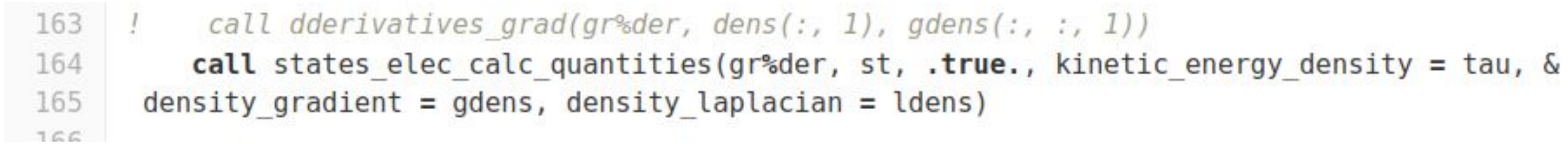
Simple energy expression:

$$E_{\mathrm{x}}^{(\mathrm{GA})}[n, \nabla n] = \frac{1}{16\pi} \sum_{\alpha=1}^{N_p} |\boldsymbol{\lambda}_\alpha|^2 \int d\mathbf{r} \frac{\omega_p^2(\mathbf{r})}{\sqrt{\omega_p^2(\mathbf{r})/3 + \omega_g^2(\mathbf{r})} + \omega_\alpha}. \tag{10}$$

plasmon frequency $\omega_p^2(\mathbf{r}) = 4\pi n(\mathbf{r}) e^2/m$

gap frequency $\omega_g^2(\mathbf{r}) = C \frac{\hbar^2}{m^2} \left| \frac{\nabla n(\mathbf{r})}{n(\mathbf{r})} \right|^4$

Only density and gradient of density is necessary (similar as GGA)

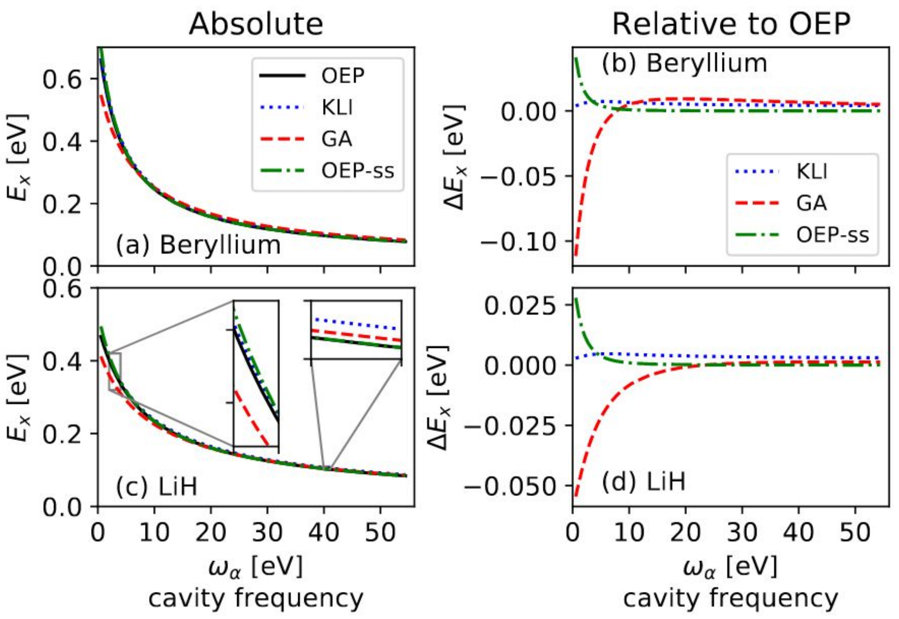


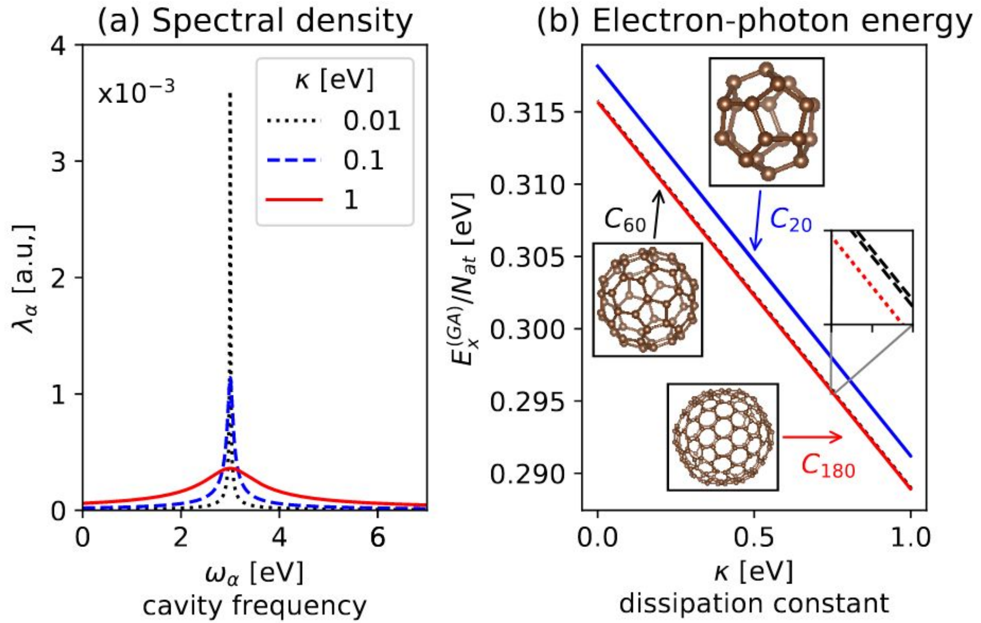
```
!    call dderivatives_grad(gr%der, dens(:, 1), gdens(:, :, 1))
       call states_elec_calc_quantities(gr%der, st, .true., kinetic_energy_density = tau, &
 density_gradient = gdens, density_laplacian = ldens)
```

Computationally inexpensive!

J. Flick, arXiv:2104.06980 (2021)

Simple electron-photon functionals for density-functional theory (QEDFT)





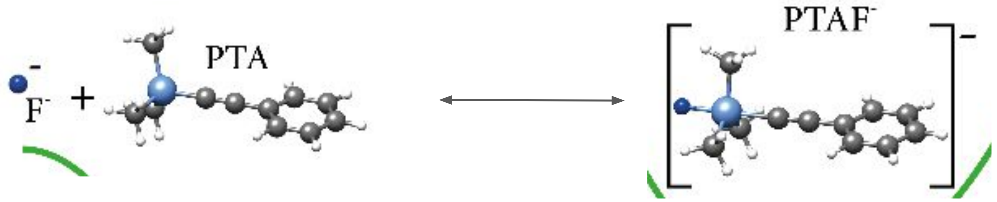
Accurate for small benchmark systems, easily scalable to 100.000s of photon modes

J. Flick, arXiv:2104.06980 (2021)

Geometry relaxation with constraints

For a time-dependent simulation:
We want a initial configuration that is different from the relaxed geometry.

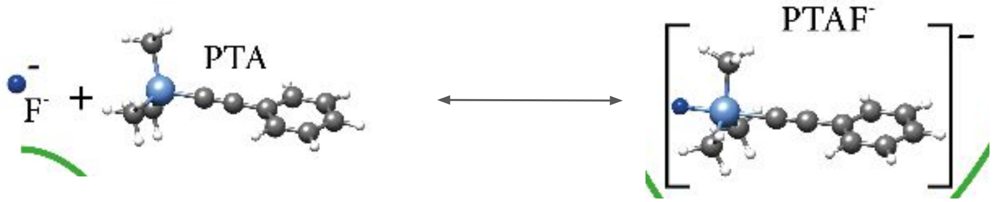
Example:

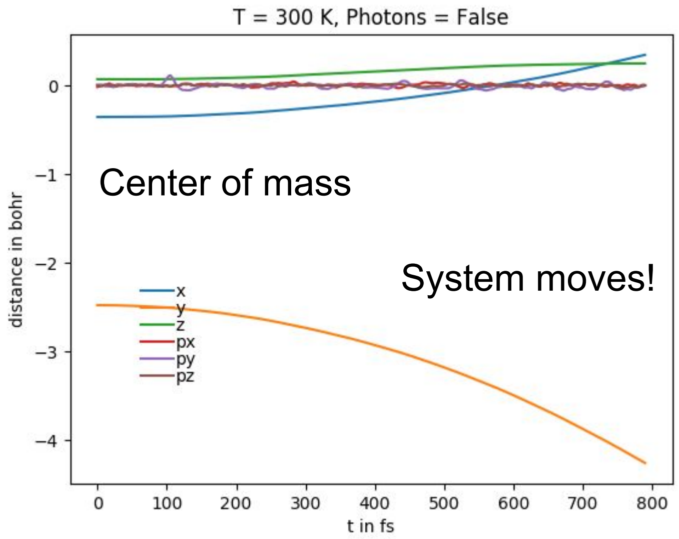


Geometry relaxation with constraints

For a time-dependent simulation:
We want a initial configuration that is different from the relaxed geometry.

Example:





Center of mass

System moves!

Workaround:
Relax the molecule without the F atom and then add the F atom to the box

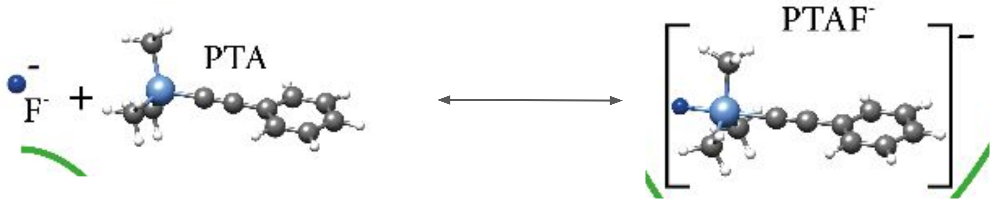
This of course somewhat works, however in this project we wanted to perform very long time propagations > few picoseconds.

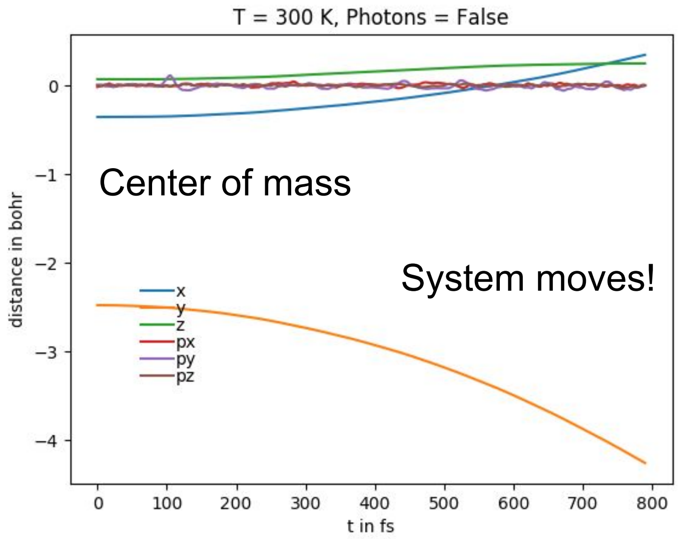
If the system is not relaxed properly then finite forces will exist that will move (accelerate!) the system. This will become visible for long times

Geometry relaxation with constraints

For a time-dependent simulation:
We want a initial configuration that is different from the relaxed geometry.

Example:





Workaround:
Relax the molecule without the F atom and then add the F atom to the box

This of course somewhat works, however in this project we wanted to perform very long time propagations > few picoseconds.

If the system is not relaxed properly then finite forces will exist that will move (accelerate!) the system. This will become visible for long times

Solution: Constraints (bond length/bond angles) during the relaxation, while minimizing forces

Geometry relaxation with constraints

Solution: Constraints (bond length/bond angles) during the relaxation, while minimizing forces

This is standard feature in many codes! However octopus can not do that. Octopus can only completely freeze certain nuclei. However then finite forces remain in that case.

Geometry relaxation with constraints

Solution: Constraints (bond length/bond angles) during the relaxation, while minimizing forces

This is standard feature in many codes! However octopus can not do that. Octopus can only completely freeze certain nuclei. However then finite forces remain in that case.

Implemented a simple version directly by modifying the relaxation routine:

Only works for 1, or 2 constraints. Constraint is bond angle and/or bond distance

Idea: convert in subspace into internal coordinates and keep e.g. bond length fixed.

Example: System with x1, x2, x3:

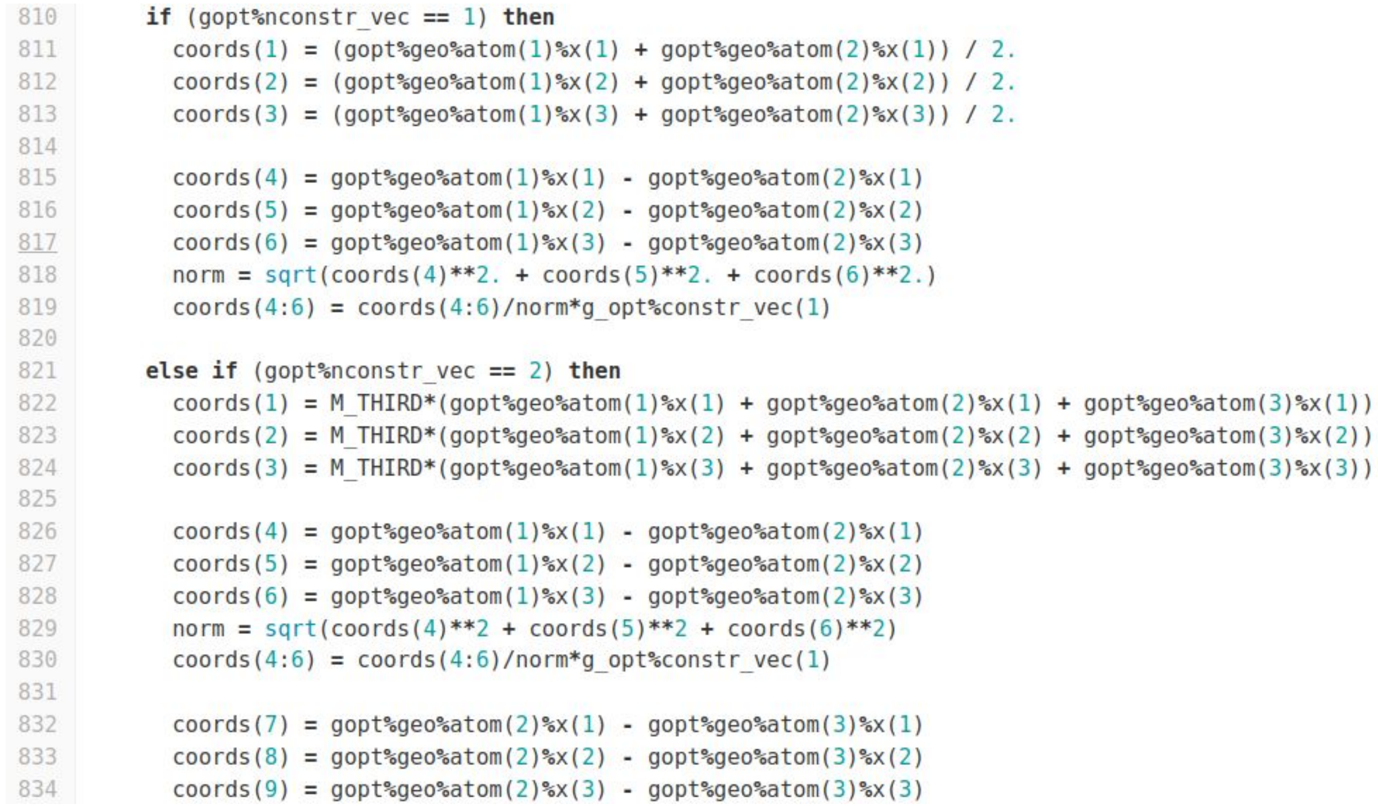
r = x1-x2
R = (x1 + x2)/2
x3 = x3
relax R, and x3

Geometry relaxation with constraints

Everything hard coded:
main/geom_opt.F90

This is limited:
More general framework necessary potentially using Lagrangian?

Or external library?



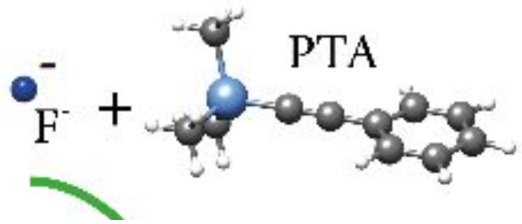
```
    if (gopt%nconstr_vec == 1) then
      coords(1) = (gopt%geo%atom(1)%x(1) + gopt%geo%atom(2)%x(1)) / 2.
      coords(2) = (gopt%geo%atom(1)%x(2) + gopt%geo%atom(2)%x(2)) / 2.
      coords(3) = (gopt%geo%atom(1)%x(3) + gopt%geo%atom(2)%x(3)) / 2.

      coords(4) = gopt%geo%atom(1)%x(1) - gopt%geo%atom(2)%x(1)
      coords(5) = gopt%geo%atom(1)%x(2) - gopt%geo%atom(2)%x(2)
      coords(6) = gopt%geo%atom(1)%x(3) - gopt%geo%atom(2)%x(3)
      norm = sqrt(coords(4)**2. + coords(5)**2. + coords(6)**2.)
      coords(4:6) = coords(4:6)/norm*g_opt%constr_vec(1)

    else if (gopt%nconstr_vec == 2) then
      coords(1) = M_THIRD*(gopt%geo%atom(1)%x(1) + gopt%geo%atom(2)%x(1) + gopt%geo%atom(3)%x(1))
      coords(2) = M_THIRD*(gopt%geo%atom(1)%x(2) + gopt%geo%atom(2)%x(2) + gopt%geo%atom(3)%x(2))
      coords(3) = M_THIRD*(gopt%geo%atom(1)%x(3) + gopt%geo%atom(2)%x(3) + gopt%geo%atom(3)%x(3))

      coords(4) = gopt%geo%atom(1)%x(1) - gopt%geo%atom(2)%x(1)
      coords(5) = gopt%geo%atom(1)%x(2) - gopt%geo%atom(2)%x(2)
      coords(6) = gopt%geo%atom(1)%x(3) - gopt%geo%atom(2)%x(3)
      norm = sqrt(coords(4)**2 + coords(5)**2 + coords(6)**2)
      coords(4:6) = coords(4:6)/norm*g_opt%constr_vec(1)

      coords(7) = gopt%geo%atom(2)%x(1) - gopt%geo%atom(3)%x(1)
      coords(8) = gopt%geo%atom(2)%x(2) - gopt%geo%atom(3)%x(2)
      coords(9) = gopt%geo%atom(2)%x(3) - gopt%geo%atom(3)%x(3)
```

Geometry relaxation with constraints

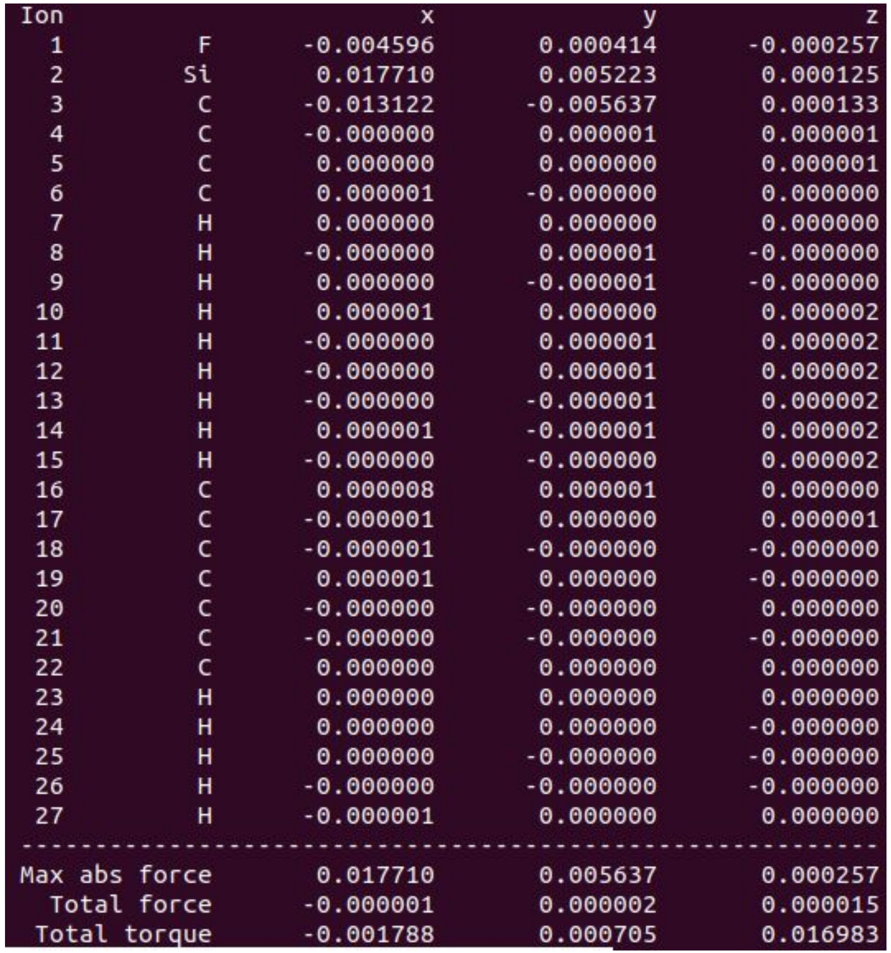
Example:

2 constraints:

1. Fixed bond length between F and Si
2. Fixed bond angle between F-C and Si-C



Max abs force is large
But vanishing total force!



```
Ion                           x               y               z
  1          F        -0.004596        0.000414       -0.000257
  2         Si         0.017710        0.005223        0.000125
  3          C        -0.013122       -0.005637        0.000133
  4          C        -0.000000        0.000001        0.000001
  5          C         0.000000        0.000000        0.000001
  6          C         0.000001       -0.000000        0.000000
  7          H         0.000000        0.000000        0.000000
  8          H        -0.000000        0.000001       -0.000000
  9          H         0.000000       -0.000001       -0.000000
 10          H         0.000001        0.000000        0.000002
 11          H        -0.000000        0.000001        0.000002
 12          H        -0.000000        0.000001        0.000002
 13          H        -0.000000       -0.000001        0.000002
 14          H         0.000001       -0.000001        0.000002
 15          H        -0.000000       -0.000000        0.000002
 16          C         0.000008        0.000001        0.000000
 17          C        -0.000001        0.000000        0.000001
 18          C        -0.000001       -0.000000       -0.000000
 19          C         0.000001        0.000000       -0.000000
 20          C        -0.000000       -0.000000        0.000000
 21          C        -0.000000       -0.000000       -0.000000
 22          C         0.000000        0.000000        0.000000
 23          H         0.000000        0.000000        0.000000
 24          H         0.000000        0.000000       -0.000000
 25          H         0.000000       -0.000000       -0.000000
 26          H        -0.000000       -0.000000       -0.000000
 27          H        -0.000001        0.000000        0.000000
----------------------------------------------------------------
Max abs force          0.017710        0.005637        0.000257
  Total force         -0.000001        0.000002        0.000015
 Total torque         -0.001788        0.000705        0.016983
```

Casida-Salahub (AC-LDA) functional implementation

Good excitation energies from linear response calculations?
Currently in octopus only the LDA kernel is implemented.
While LDA can be somewhat accurate for low-lying excited states. However due to the wrong asymptotics it becomes quite unreliable for higher lying/Rydberg excitations.

M. Casida, K.C. Casida, D.R. Salahub, Int. Journ. of Quant. Chem. 70, 933-941 (1998).

Casida-Salahub (AC-LDA) functional implementation

Good excitation energies from linear response calculations?
Currently in octopus only the LDA kernel is implemented.
While LDA can be somewhat accurate for low-lying excited states. However due to the wrong asymptotics it becomes quite unreliable for higher lying/Rydberg excitations.
A simple correction has been introduced by Casida and Salahub:

Combine the asymptotically correction potential of van Leeuwen and Baerends (LB94) in the asymptotic region with the LDA in the bulk region

$$v_{\mathrm{xc}}^{\text{AC-LDA}}(\mathbf{r}) = \mathrm{Max}\left[v_{\mathrm{xc}}^{\mathrm{LDA}}(\mathbf{r}) - \Delta, v_{\mathrm{xc}}^{\mathrm{LB94}}(\mathbf{r}) \right], \quad (1.5)$$

where

Highest occupied orbital

$$\Delta = I + \epsilon_{\mathrm{HOMO}} \quad (1.6)$$

ionization potential

M. Casida, K.C. Casida, D.R. Salahub, Int. Journ. of Quant. Chem. 70, 933-941 (1998).

Casida-Salahub (AC-LDA) functional implementation

$$v_{\mathrm{xc}}^{\mathrm{AC\text{-}LDA}}(\mathbf{r}) = \mathrm{Max}\left[v_{\mathrm{xc}}^{\mathrm{LDA}}(\mathbf{r}) - \Delta, v_{\mathrm{xc}}^{\mathrm{LB94}}(\mathbf{r})\right], \quad (1.5)$$

where

Highest occupied orbital

$$\Delta = I + \epsilon_{\mathrm{HOMO}} \qquad (1.6)$$

ionization potential

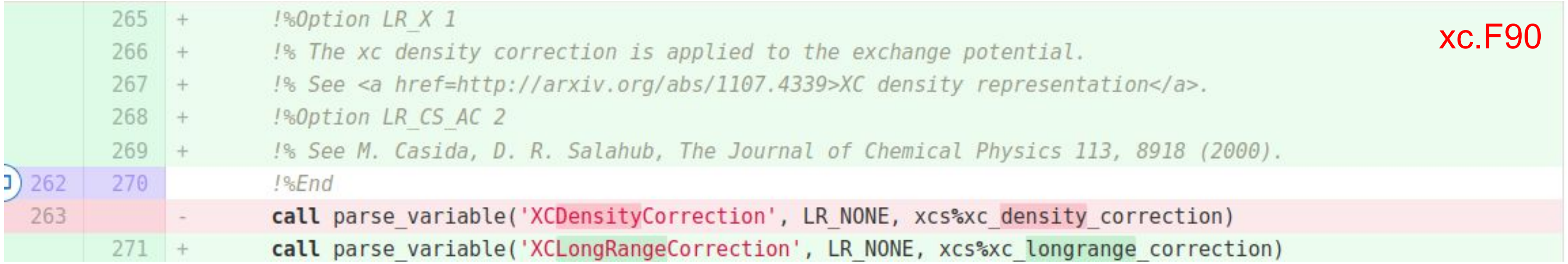
Workflow:

1. Two ground-state calculations to get the ionization potential with N-1, and N+1 electrons
2. AC-LDA ground-state run with N elections and I as input
3. Casida run for excited states with LDA kernel

Problems for the octopus implementation:
To calculate the potential one has to access both LDA and GGA type objects.
However the potential specifies the family of xc, e.g. either LDA or GGA
Workaround: overload the kernel. Put the LB94 into the kernel

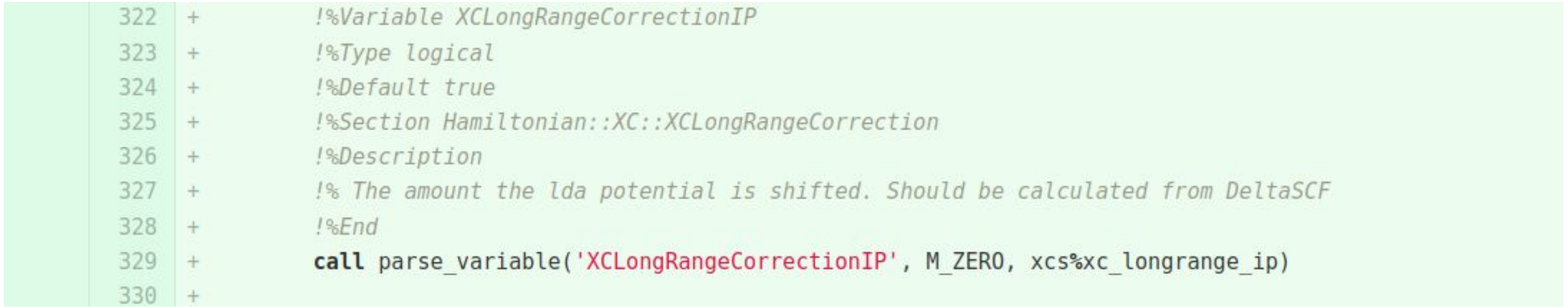
M. Casida, K.C. Casida, D.R. Salahub, Int. Journ. of Quant. Chem. 70, 933-941 (1998).

Casida-Salahub (AC-LDA) functional implementation

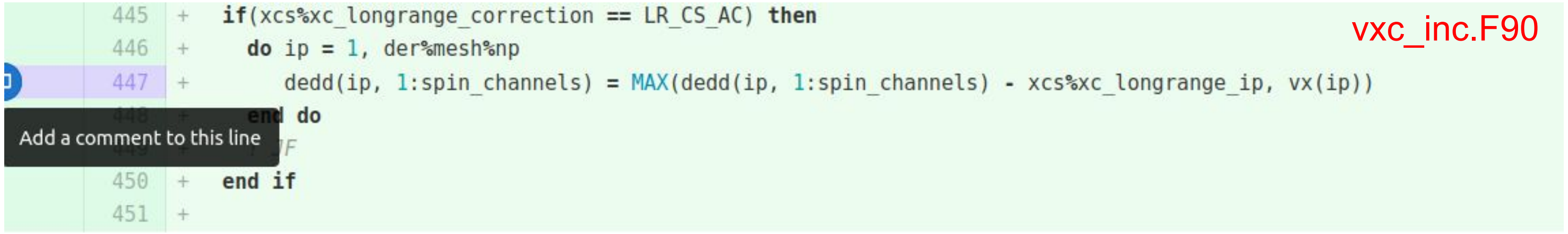


xc.F90

```
+        !%Option LR_X 1
+        !% The xc density correction is applied to the exchange potential.
+        !% See <a href=http://arxiv.org/abs/1107.4339>XC density representation</a>.
+        !%Option LR_CS_AC 2
+        !% See M. Casida, D. R. Salahub, The Journal of Chemical Physics 113, 8918 (2000).
         !%End
-        call parse_variable('XCDensityCorrection', LR_NONE, xcs%xc_density_correction)
+        call parse_variable('XCLongRangeCorrection', LR_NONE, xcs%xc_longrange_correction)
```



```
+          !%Variable XCLongRangeCorrectionIP
+          !%Type logical
+          !%Default true
+          !%Section Hamiltonian::XC::XCLongRangeCorrection
+          !%Description
+          !% The amount the lda potential is shifted. Should be calculated from DeltaSCF
+          !%End
+          call parse_variable('XCLongRangeCorrectionIP', M_ZERO, xcs%xc_longrange_ip)
+
```



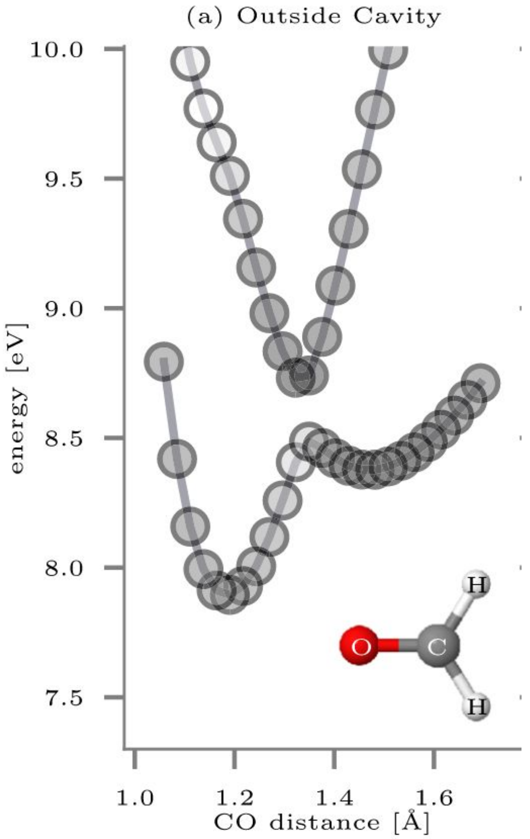
vxc_inc.F90

```
+ if(xcs%xc_longrange_correction == LR_CS_AC) then
+   do ip = 1, der%mesh%np
+      dedd(ip, 1:spin_channels) = MAX(dedd(ip, 1:spin_channels) - xcs%xc_longrange_ip, vx(ip))
+   end do
+   JF
+ end if
+
```

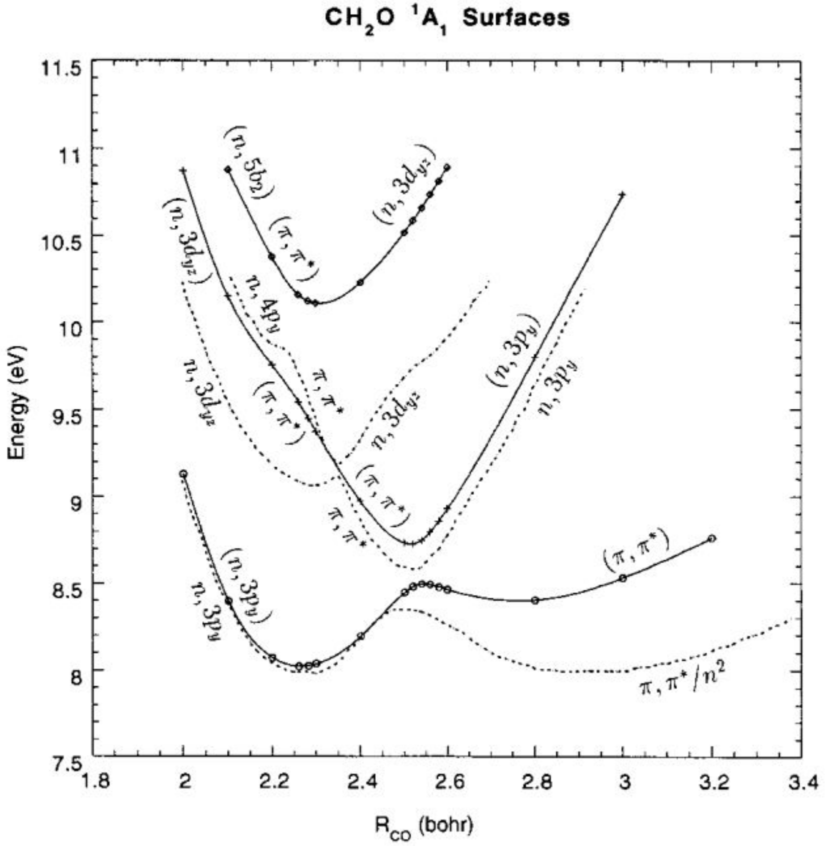
Add a comment to this line

M. Casida, K.C. Casida, D.R. Salahub, Int. Journ. of Quant. Chem. 70, 933-941 (1998).

Casida-Salahub (AC-LDA) functional implementation



J. Flick, P. Narang, J. Chem. Phys. **153**, 094116 (2020).

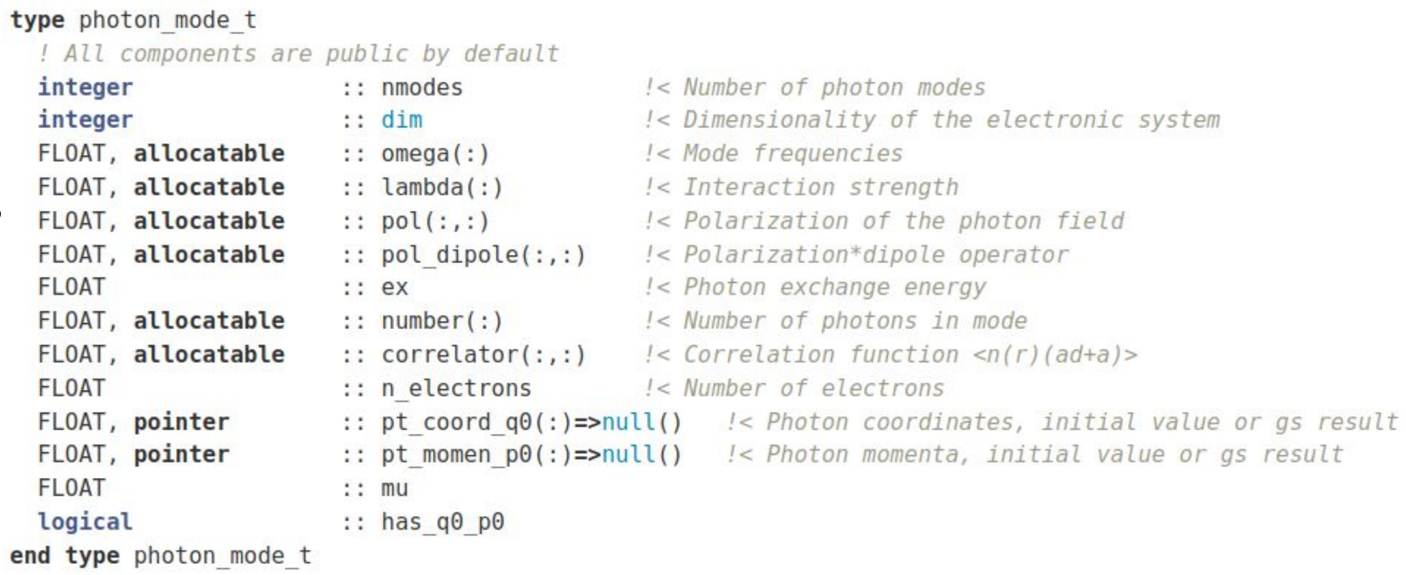


M. Casida, K.C. Casida, D.R. Salahub, Int. Journ. of Quant. Chem. 70, 933-941 (1998).

Coordination Photon Implementations

-Some coordination would be good about common files: e.g. syntax photon_modes.F90
photon_mode.F90 is (still) in the poisson folder?

Move to subroutines?



```
type photon_mode_t
  ! All components are public by default
  integer                   :: nmodes                 !< Number of photon modes
  integer                   :: dim                    !< Dimensionality of the electronic system
  FLOAT, allocatable        :: omega(:)               !< Mode frequencies
  FLOAT, allocatable        :: lambda(:)              !< Interaction strength
  FLOAT, allocatable        :: pol(:,:)               !< Polarization of the photon field
  FLOAT, allocatable        :: pol_dipole(:,:)        !< Polarization*dipole operator
  FLOAT                     :: ex                     !< Photon exchange energy
  FLOAT, allocatable        :: number(:)              !< Number of photons in mode
  FLOAT, allocatable        :: correlator(:,:)        !< Correlation function <n(r)(ad+a)>
  FLOAT                     :: n_electrons            !< Number of electrons
  FLOAT, pointer            :: pt_coord_q0(:)=>null()   !< Photon coordinates, initial value or gs result
  FLOAT, pointer            :: pt_momen_p0(:)=>null()   !< Photon momenta, initial value or gs result
  FLOAT                     :: mu
  logical                   :: has_q0_p0
end type photon_mode_t
```

Syntax?
Sufficient?
1D/2D/3D?

%PhotonModes
 omega1 | lambda1| PolX1 | PolY1 | PolZ1 | q0 | p0
...
%

Discussion and Conclusion

Discussion points:

- Photon part under the multi-system architecture?
- Some coordination would be good for common files, photon_mode.F90, forces ...
- Connection to the Maxwell implementation?
- Consistency of definition of the sign of the dipole moment with other parts of the code.

- What to do with “old” implementations?
- Casida-salahub functional and the 1/2 geometry constraint relaxation
- Is there interest to get it merged? Help with merging?

Other features:

- Beyond LDA kernels?

Discussion and Conclusion

Discussion points:

- Photon part under the multi-system architecture?
- Some coordination would be good for common files, photon_mode.F90, forces ...
- Connection to the Maxwell implementation?
- Consistency of definition of the sign of the dipole moment with other parts of the code.

- What to do with “old” implementations?
- Casida-salahub functional and the 1/2 geometry constraint relaxation
- Is there interest to get it merged? Help with merging?

Other features:

- Beyond LDA kernels?

I want to thank the octopus developer and maintainer!