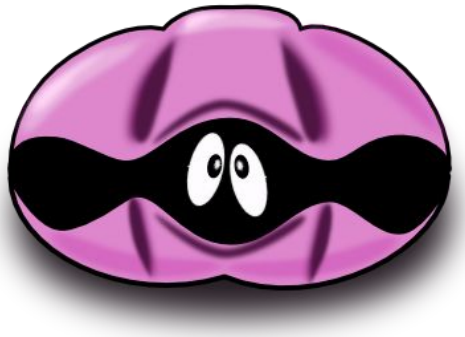
# Yambo©



# Cheatsheet

## For GPL v4.5

MAX

DRIVING THE EXASCALE TRANSITION

# Input file generation and command line interface (yambo -H)

## Lower case options are for input file generation

```
-i            :Initialization
-o <opt>      :Optics [opt=(c)hi is (G)-space
                      (b)se is (eh)-space ]
-k <opt>      :Kernel [opt=hartree/alda/lrc/hf/sex]
                      hf/sex only eh-space;
                      lrc only G-space
-y <opt>      :BSE solver [opt=h/d/s/(p/f)i]
                      (h)aydock   (s)lepc
                      (d)iagonalization  (i)nversion
-r            :Coulomb potential
-x            :Hartree-Fock Self-energy and local XC
-d            :Dynamical Inverse Dielectric Matrix
-b            :Static Inverse Dielectric Matrix
-p <opt>      :GW approximations
                      [opt=(p)PA/(c)HOSEX]
-g <opt>      :Dyson Equation solver
                       [opt=(n)ewton/(s)ecant/(g)reen]
-l            :GoWo Quasiparticle lifetimes
-a            :ACFDT Total Energy
```

## Upper case options are more general

```
-J <opt>      :Job string identifier
-V <opt>      :Input file verbosity
               [opt=RL,kpt,sc,qp,io,gen,resp,all,par]
-F <opt>      :Input file
-I <opt>      :Core I/O directory
-O <opt>      :Additional I/O directory
-C <opt>      :Communications I/O directory
-D            :DataBases properties
-W <opt>      :Wall Time limitation (1d2h30m format)
-Q            :Don't launch the text editor
-E <opt>      :Environment Parallel Variables file
-M            :Switch-off MPI support (serial run)
-N            :Switch-off OpenMP support (one thread)
```

## Combination of options

Examples: input file generation/runlevel selection:

```
$ yambo -o c -k hartree -V RL      Optics, LFE
$ yambo -x -g n -p p -V qp         GW with PPA
$ yambo -o b -k sex -y h -b        Optics, BSE
```

# Example of practical use

**Simple**

```
$ yambo -F RPA_optics.in -o c -k hartree
$ yambo -F RPA_optics.in
```

⇒ Generate RPA_optics.in input file (-F)

⇒ Run yambo. Parameters are loaded from RPA_optics.in (-F)

**Advanced**

```
$ yambo -F Screening.in -b -V RL
$ yambo -F Screening.in -J EM1S -C EM1S_out
```

⇒ Generate Screening.in input file

⇒ Run yambo to compute screening.

- Human readable files are stored inside EM1S_out folder (-C)
- To all human readable file-names is appended the EM1S string (-J)
- Binary files are stored inside EM1S folder (-J)

```
$ yambo -F BSE.in -o b -k sex -y h -b
$ yambo -F BSE.in -J "BSE,EM1S" -C BSE_out
```

⇒ Generate BSE.in input file

⇒ Do a BSE simulation loading the screening from EM1S folder. The parameters inside BSE.in for screening must match the Screening.in input file

## (1) Linear response (IP/RPA-NLF): `yambo -o c`

See (2)

$$\epsilon_{\alpha,\alpha}(\omega) = 1 + \frac{16\pi}{\Omega} \sum_{c,v} \sum_{\mathbf{k}} \frac{1}{E_{c\mathbf{k}} - E_{v\mathbf{k}}} \frac{|\langle v\mathbf{k}|\mathbf{p}_\alpha + i[V^{\mathrm{NL}}, \mathbf{r}_\alpha]|c\mathbf{k}\rangle|^2}{(E_{c\mathbf{k}} - E_{v\mathbf{k}})^2 - (\omega + i\gamma)^2}$$

```
% LongDrXd
 1.000 | 0.000 | 0.000 |
%
```
**E-field direction (for q=0)**
*Vector (cartesian coordinate)*
Refers to first q-point (`QpntsRXd`)

**DFT k-grid**

```
% EnRngeXd
 0.000 | 10.000 | eV
%
ETStpsXd = 100
```
**Energy grid in output**
*Range from 0 to 10 in 100 steps*

```
% BndsRnXd
 1 | 100 |
```
**Bands used (empty & filled)**
*Range from 1 to nbnd*
Reduce range to lower memory. In metals, includes partially filled bands. See also `EhEngyXd (-V all)`

```
% DmRngeXd
 0.1000 | 0.100 | eV
%
```
**Broadening of spectra**
*Either a fixed value, or linearly changing between 2 values*

## (2) FFT and oscillators: `yambo -o c -V RL (several runlevels)`

Dipole/momentum matrix elements ($q \to 0$)

$$\langle n\mathbf{k}|\mathbf{p} + i[V^{\mathrm{NL}}, \mathbf{r}]|m\mathbf{k}\rangle$$

Screening matrix elements (FFT)

$$\rho_{nm}(\boldsymbol{k},\boldsymbol{q},\boldsymbol{G}) = \langle n\boldsymbol{k}|e^{i(\boldsymbol{q}+\boldsymbol{G})\cdot\boldsymbol{r}}|m\boldsymbol{k}-\boldsymbol{q}\rangle$$

where the wavefunction is expanded over reciprocal lattice (G) vectors:

$$\phi_{n\mathbf{k}}(\mathbf{r}) = \frac{1}{\sqrt{V}}\sum_{\mathbf{G}} e^{i(\mathbf{k}+\mathbf{G})\cdot\mathbf{r}} c_{n\mathbf{k}}(\mathbf{G})$$

`FFTGvecs = 1 RL`
**Number of G-vectors (or energy cutoff) for expanding wavefunctions in transition matrix elements and FFT operations**
*Units: number of G-vectors (RL), or energy unit (Ry, mHa, eV)*
Determines size (memory) of calculation. Corresponds to cutoff in DFT calculation; can be much less than geometry cutoff

**Non-local commutator term**
Can greatly increase CPU time. Remove by hiding the `SAVE/ns.kb_pp_pwscf` file.

## (3) Linear response with local fields (RPA-LFE): `yambo -o c -k hartree`

$$\chi_{\mathbf{G},\mathbf{G}'}(\mathbf{q},\omega) = \chi^0_{\mathbf{G},\mathbf{G}'}(\mathbf{q},\omega) + \sum_{\mathbf{G}_1,\mathbf{G}_2} \chi^0_{\mathbf{G},\mathbf{G}_1}(\mathbf{q},\omega)\left[v_{\mathbf{G}_1}(\mathbf{q})\delta_{\mathbf{G}_1,\mathbf{G}_2} + f^{xc}_{\mathbf{G}_1,\mathbf{G}_2}\right]\chi_{\mathbf{G}_2,\mathbf{G}'}(\mathbf{q},\omega)$$

Hartree ALDA/LRC

$$\chi^0_{\mathbf{G}\mathbf{G}'}(\mathbf{q},\omega) = 2\sum_{c,v}\int_{BZ}\frac{d\mathbf{k}}{(2\pi)^3}\rho^*_{cv\mathbf{k}}(\mathbf{q},\mathbf{G})\rho_{cv\mathbf{k}}(\mathbf{q},\mathbf{G}')\, f_{v\mathbf{k}-\mathbf{q}}(1-f_{c\mathbf{k}}) \times \left[\frac{1}{\omega + \epsilon_{v\mathbf{k}-\mathbf{q}} - \epsilon_{c\mathbf{k}} + i0^+} - \frac{1}{\omega + \epsilon_{c\mathbf{k}} - \epsilon_{v\mathbf{k}-\mathbf{q}} - i0^+}\right]$$

See (2)

```
% QpntsRXd
 1 | 6 |
%
```
**Transferred momenta**
*Range from 1:nqpt (1 = q→0)*

`NGsBlkXd=1`
**Response block size**
*Depends on inhomogeneity; needs convergence study, but much less than FFTGvecs*

`% EhEngyXd (-V resp)`
`-1.000 | -1.000 | eV`
**Electron-hole energy range** *An alternative way to BndsRnXd to restrict transitions, but will not reduce memory*

`Chimod="Hartree"`
**Kernel: set with -k <opt>**
*Hartree/ALDA/LRC*

`GrFnTpXd="T" (-V resp)`
**Green's function: T/R/r/Ta/Ra**
*"R" = resonant gives the causal response function*

## (4) Linear response within TDDFT: `yambo -o c -k ALDA/LRC`

$$\chi_{\mathbf{G},\mathbf{G}'}(\mathbf{q},\omega) = \chi^0_{\mathbf{G},\mathbf{G}'}(\mathbf{q},\omega) + \sum_{\mathbf{G}_1,\mathbf{G}_2} \chi^0_{\mathbf{G},\mathbf{G}_1}(\mathbf{q},\omega)\left[v_{\mathbf{G}_1}(\mathbf{q})\delta_{\mathbf{G}_1,\mathbf{G}_2} + f^{xc}_{\mathbf{G}_1,\mathbf{G}_2}\right]\chi_{\mathbf{G}_2,\mathbf{G}'}(\mathbf{q},\omega)$$

Hartree ALDA/LRC

$$\chi^0_{\mathbf{G}\mathbf{G}'}(\mathbf{q},\omega) = 2\sum_{c,v}\int_{BZ}\frac{d\mathbf{k}}{(2\pi)^3}\rho^*_{cv\mathbf{k}}(\mathbf{q},\mathbf{G})\rho_{cv\mathbf{k}}(\mathbf{q},\mathbf{G}')\, f_{v\mathbf{k}-\mathbf{q}}(1-f_{c\mathbf{k}})\times$$

See (2)

$$\times\left[\frac{1}{\omega+\epsilon_{v\mathbf{k}-\mathbf{q}}-\epsilon_{c\mathbf{k}}+i0^+}-\frac{1}{\omega+\epsilon_{c\mathbf{k}}-\epsilon_{v\mathbf{k}-\mathbf{q}}-i0^+}\right]$$

**`NGsBlkXd=1`**
**Response block size**
*Depends on inhomogeneity; needs convergence study, but much less than FFTGvecs*

**`FxcGRLc=1`**
**XC-kernel size**
*Needs convergence study. Much less than FFTGvecs*

**`LRC_alpha=1`**
**LRC fitting parameter**
*Long-range tail of the $f_{xc}$ kernel. Depends on the system: the larger the screening the smaller this parameter.*

**`Chimod="ALDA"`**
**Kernel: set with -k <opt>**
*<opt>=ALDA*
*<opt>=LRC: semi-empirical kernel with proper long-range behaviour. It needs a fitting parameter!*

## (5) Screening (RPA): `yambo -d`

See sheet (3): this runlevel computes the inverse dielectric matrix from X(G,G')

$$\epsilon^{-1}_{\mathbf{G},\mathbf{G}'}(\mathbf{q},\omega) = \delta_{\mathbf{G},\mathbf{G}'} + v_{\mathbf{G}}(\mathbf{q})\chi_{\mathbf{G},\mathbf{G}'}(\mathbf{q},\omega)$$

**Connection with experiment:**

$$\epsilon_M(\omega) = \lim_{\mathbf{q}\to 0} \frac{1}{\epsilon^{-1}_{\mathbf{G}=\mathbf{0},\mathbf{G}'=\mathbf{0}}(\mathbf{q},\omega)}$$

$$\mathrm{Abs}(\omega) = \mathrm{Im}\,\epsilon_M(\omega) \qquad \mathrm{EELS}(\omega) = -\mathrm{Im}\,\frac{1}{\epsilon_M(\omega)}$$

## (6a) Coulomb integrals RIM (Random Integration Method): `yambo -r`

$$v(\boldsymbol{q}+\boldsymbol{G}) = \frac{4\pi}{|\boldsymbol{q}+\boldsymbol{G}|^2}$$

$$\int_{Bz} \frac{d^3q}{2\pi^3} f(q,G)v(q+G) \approx \sum_{q_i} f(q_i,G)v(q_i+G)\Omega_{q_i}$$

Discretization of Bz for integrals

A better approximation is given by:

$$\int_{Bz} \frac{d^3q}{2\pi^3} f(q,G)v(q+G) \approx \sum_{q_i} f(q_i,G)I_{q_i}(G)$$

Monte Carlo integral of the Coulomb potential in each region the Bz has been dissected by the q point sampling

$$I_{q_i}(G) = \int_{R_\Gamma} \frac{d^3q'}{(2\pi)^3} v(q+q'+G)$$

Tip: 1. Needed for non 3D system to avoid divergences for small q
2. Needed to build cutoff potential with box shape

**Random Integration Method**
RandQpts=1000000
*Number of q points to perform Monte Carlo Integration,*
RandGvec= 1 RL
*Number of G vectors the RIM is calculated*
Tip: RandGvec=1 (gamma) is usually enough. 1 Million q points is usually accurate.

## (6b) Coulomb cutoff : `yambo -r`

$$v(\boldsymbol{q}+\boldsymbol{G}) = \frac{4\pi}{|\boldsymbol{q}+\boldsymbol{G}|^2}$$

Truncation of the Coulomb potential for non 3D system to speed up convergence with respect the vacuum

$$V_c(\mathbf{r}) = \begin{cases} \frac{1}{|\mathbf{r}|}, & \text{if } \mathbf{r} \in S \\ 0, & \text{otherwise.} \end{cases}$$

**S: interactting region:**

CUTGeo= "box Z"  X/Y/Z or XY/XZ/YZ or XYZ

- *Possible region:*
- *sphere (0D for molecules),*
- *cylinder (1D for polymers, tubes, etc),*
- *box (0D, 1D, 2D).*

*XYZ: cut in all directions*
*Box: XY: cut in XY only, etc…*
*Cylinder X/Y/Z indicates cylinder axis*

- Sphere XYZ: assign: CUTRadius= 10.0 a.u
- Cylinder Z: assign CUTRadius and CUTCylLen (CUTCylLen=0 indicates infinite cylinder)
- Box Z: assign CUTBox

```
% CUTBox
 0.00    | 0.00    | 32.00    |       # [CUT] [au]
Box sides
%
```

Box side=0 means do not cut in that direction

*Tip: When using Box shapes, the RIM is also needed to calculate the potential. In Box for large enough boxes assigns Box side slighlty smaller than the cell box*

## (7) Exchange self energy - Vxc : $\Sigma^x_{n\mathbf{k}} - V^{xc}_{n\mathbf{k}}$ yambo -x

$$\Sigma^x_{n\mathbf{k}} = \langle n\mathbf{k}|\Sigma^x(\mathbf{r}_1,\mathbf{r}_2)|n\mathbf{k}\rangle = -\sum_m \int_{BZ} \frac{d\mathbf{q}}{(2\pi)^3} \sum_{\mathbf{G}} v(\mathbf{q}+\mathbf{G}) \left|\rho_{nm}(\mathbf{k},\mathbf{q},\mathbf{G})\right|^2 f_{m(\mathbf{k}-\mathbf{q})}$$

See (6) — See (2)

occupied bands only

DFT k-grid
{q} = {k-k'}

**G-vectors in the exchange and xc-potential**
*Number of RL vectors, or energy in Ry / mHa / etc*
*Tip : to be converged, possibly leave the maximum value*

```
EXXRLvcs= 2487001 RL
VXCRLvcs= 2487001 RL
```

```
%QPkrange
  1|  5| 20|  59|
  4|  8| 60|  80|
%
%QPerange (-V qp)
  1| 32| 0.0|-1.0|
%
```

**nk, n'k' ranges where GW/$\Sigma_x$ elements are calculated**
*first k-point | last k-point | lower band | upper band*
*This can be split over several lines for multiple groups*
Tip: careful use of fewer k-points and bands reduces the calculation time; yambo will interpolate the rest
**nk,nk' ranges (alternative method)**
*first k-point | last k-point | lower energy | upper energy*

## (8) Correlation part of self energy: `yambo -g n`

$$\Sigma^c_{n\mathbf{k}}(\omega) = \langle n\mathbf{k}|\Sigma^c|n\mathbf{k}\rangle = i\sum_m \int_{BZ} \frac{d\mathbf{q}}{(2\pi)^3} \sum_{\mathbf{G},\mathbf{G}'} \frac{4\pi}{|\mathbf{q}+\mathbf{G}|^2} \rho_{nm}(\mathbf{k},\mathbf{q},\mathbf{G})\rho^*_{nm}(\mathbf{k},\mathbf{q},\mathbf{G}') \times \int d\omega' G^0_{m\mathbf{k}-\mathbf{q}}(\omega-\omega')\epsilon^{-1}_{\mathbf{G}\mathbf{G}'}(\mathbf{q},\omega')$$

```
%QPkrange
  1|  5| 20|  59|
  4|  8| 60|  80|
%
%QPerange (-V qp)
  1| 32| 0.0|-1.0|
%
```

```
% GbndRnge
  1 | 50 |
%
```

**Bands used in the GW summation**
*QP energies usually shows slow convergence*
Tip: If you are interested in gaps, energy differences converge faster

NGsBlkXp= 100 RL
**Response block size**
See (9)

**nk, n'k' ranges where GW/$\Sigma_c$ elements are calculated**
*first k-point | last k-point | lower band | upper band*
*This can be split over several lines for multiple groups*
Tip: careful use of fewer k-points and bands reduces the calculation time; yambo will interpolate the rest

## (8a) Dyson Solver: `yambo -g n/s`

$$E_{nk}^{QP} = \epsilon_{nk} + \langle \psi_{nk} | \Sigma(E_{nk}^{QP}) - V_{xc} | \psi_{nk} \rangle$$

DysSolver= "n" First order expansion around KS eigenvalue

$$E_{nk}^{QP} = \epsilon_{nk} + Z_{nk} \langle \psi_{nk} | \Sigma(\epsilon_{nk}) - V_{xc} | \psi_{nk} \rangle$$

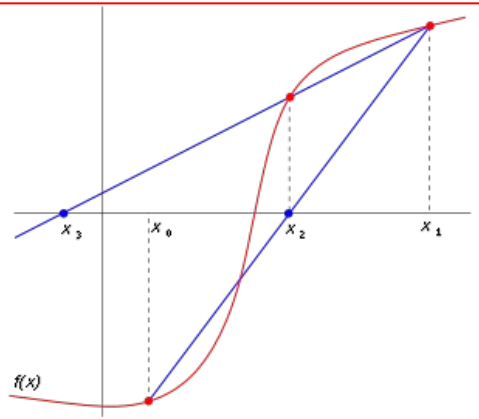
$$Z_{nk} = \left[ 1 - \frac{d\Sigma_{nk}(\omega)}{d\omega} \Big|_{\omega=\epsilon_{nk}} \right]^{-1}$$

dScStep= 0.10000 eV # [GW] Energy step to evaluate Z

DysSolver= "s" Secant iterative method

https://en.wikipedia.org/wiki/Secant_method

$$x_n = x_{n-1} - f(x_{n-1}) \frac{x_{n-1} - x_{n-2}}{f(x_{n-1}) - f(x_{n-2})} = \frac{x_{n-2} f(x_{n-1}) - x_{n-1} f(x_{n-2})}{f(x_{n-1}) - f(x_{n-2})}$$



## (9) Plasmon Pole approximation (PPA): `yambo -p p`

Components of the Dielectric matrix approximated has a single pole functions:

$$\epsilon^{-1}_{\mathbf{G},\mathbf{G}'}(\mathbf{q},\omega) \sim \delta_{\mathbf{G},\mathbf{G}'} + \mathbf{R}_{\mathbf{G},\mathbf{G}'}(\mathbf{q})[(\omega - \mathbf{\Omega}_{\mathbf{G},\mathbf{G}'}(\mathbf{q}) + i0^+)^{-1} - (\omega + \Omega_{\mathbf{G},\mathbf{G}'}(\mathbf{q}) - i0^+)^{-1}]$$

Residuals $R_{\mathbf{G},\mathbf{G}'}(\mathbf{q})$ and energies $\Omega_{\mathbf{G},\mathbf{G}'}(\mathbf{q})$ are found by imposing the PPA to reproduce the exact $\epsilon^{-1}$ function at ω = 0 and ω = $iE_{PPA}$ with $E_{PPA}$ being a suitable user-defined parameter.

$$R_{\mathbf{G},\mathbf{G}'}(\mathbf{q}) = -\frac{\epsilon^{-1}_{\mathbf{G},\mathbf{G}'}(\mathbf{q},\omega=0)\Omega_{\mathbf{G},\mathbf{G}'}}{2}$$

$$\Omega_{\mathbf{G},\mathbf{G}'} = E_{PPA}\sqrt{\frac{\epsilon^{-1}_{\mathbf{G},\mathbf{G}'}(\mathbf{q},\omega=\mathbf{E_{PPA}})}{\epsilon^{-1}_{\mathbf{G},\mathbf{G}'}(\mathbf{q},\omega=0) - \epsilon^{-1}_{\mathbf{G},\mathbf{G}'}(\mathbf{q},\omega=\mathbf{E_{PPA}})}}$$

The QP energies should not depend too much on the choice of imaginary plasmon frequency. **Tip:** Choose a value higher in energy than the plasmon peak (EELS spectrum)

```
% BndsRnXp
 1 | 100 |
```
**Bands used (empty & filled)**
*Range from 1 to nbnd*
Reduce range to lower memory.

NGsBlkXp= 100 RL
**Response block size**

PPAPntXp= 27.21138 eV
**PPA imaginary energy**

# (10a) Construction of the BSE Hamiltonian: `yambo -o b -k sex -b`

BSE is rewritten as an eigenvalue problem for the 2 particle Hamiltonian:
size of matrix [$N_v$ x $N_c$ x $K_{BZ}$ ] x [$N_v$ x $N_c$ x $K_{BZ}$]

$$H^{exc}_{\substack{vc\mathbf{k}\\ v'c'\mathbf{k}'}} = (\epsilon_{c\mathbf{k}} - \epsilon_{v\mathbf{k}})\delta_{c,c'}\delta_{v,v'}\delta_{\mathbf{k}\mathbf{k}'} + (f_{c\mathbf{k}} - f_{v\mathbf{k}})[2\bar{V}_{\substack{vc\mathbf{k}\\ v'c'\mathbf{k}'}} - W_{\substack{vc\mathbf{k}\\ v'c'\mathbf{k}'}}]$$

Difference of quasiparticle energies:
From DFT + QP corrections:

Kernel part: see next slide

**KfnQPdb= " E < ./SAVE/ndb.QP"**
**Location of QP corrections database**
*From previous GW calculation*

**OR**

**% KfnQP_E**
**1.4000 | 1.200 | 0.900 |**
**QP corrections parameters**
*scissor* | *stretch conduction* | *stretch valence*

**% BSEBands**
**2 | 8 |**
**Bands Range**
*lower band* | *upper band* |

## (10b) Construction of the BSE kernel: `yambo -o b -k sex -b`

Electron-hole exchange part (from Hartree potential - local field effects):

$$K^{x}_{vc\mathbf{k},v'c'\mathbf{k}'} = \bar{V}_{vc\mathbf{k},v'c'\mathbf{k}'} = \frac{1}{\Omega}\sum_{\mathbf{G}\neq\mathbf{0}} v(\mathbf{G})\langle c\mathbf{k}|e^{i\mathbf{G}\mathbf{r}}|v\mathbf{k}\rangle\langle v'\mathbf{k}'|e^{-i\mathbf{G}'\mathbf{r}}|c'\mathbf{k}'\rangle$$

**BSENGexx = 30 Ry**
**Components of Hartree potential**

Electron-hole attraction part (from screened exchange potential - excitonic effects):

$$K^{c}_{vc\mathbf{k},v'c'\mathbf{k}'} = W_{vc\mathbf{k},v'c'\mathbf{k}'} = \frac{1}{\Omega}\sum_{\mathbf{G},\mathbf{G}'} v(\mathbf{q}+\mathbf{G})\varepsilon^{-1}_{\mathbf{G},\mathbf{G}'}(\mathbf{q})\langle c\mathbf{k}|e^{i(\mathbf{q}+\mathbf{G})\mathbf{r}}|c'\mathbf{k}'\rangle\langle v'\mathbf{k}'|e^{-i(\mathbf{q}+\mathbf{G}')\mathbf{r}}|v\mathbf{k}\rangle\delta_{\mathbf{q}\mathbf{k}-\mathbf{k}'}$$

**BSENGblk = 2 Ry**
**Screened interaction block size**

**%BandsRnXs**
**1 | 20 |**
**NGsBlkXs = 2 Ry**
**% LongDrXS**
**1.000 | 1.000 | 1.000|**

See (3)

## (11a) BSE solver (diagonalisation): `yambo -y d`

The macroscopic dielectric function is obtained as:

$$\epsilon_M(\omega) \equiv 1 - \lim_{\mathbf{q}\to 0} \frac{8\pi}{|\mathbf{q}|^2\Omega} \sum_{vc\mathbf{k}} \sum_{v'c'\mathbf{k}'} \langle v\mathbf{k}-\mathbf{q}|e^{-i\mathbf{q}\mathbf{r}}|c\mathbf{k}\rangle\langle c'\mathbf{k}'|e^{i\mathbf{q}\mathbf{r}}|v'\mathbf{k}'-\mathbf{q}\rangle \sum_{\lambda} \frac{A^{\lambda}_{cv\mathbf{k}}\left(A^{\lambda}_{c'v'\mathbf{k}'}\right)^*}{\omega - E_{\lambda}}$$

**% BEnRange**
**2.00000 | 8.00000 | eV**
**%**
**Energy range**
*min energy | max energy*

**BEnSteps= 200**
**Number of evenly spaced energy points**

**% BLongDir**
**1.000000 | 1.000000 | 0.000000 |**
**%**
**Direction of the longitudinal perturbation**

**% BDmRange**
**0.10000 | 0.10000 | eV**
**%**
**Lorentzian broadening changes linearly**
*broad@min energy| broad@max energy*

## (11b) BSE solver (Lanczos-Haydock): `yambo -y h`

The macroscopic dielectric function is obtained as:

$$\epsilon_M(\omega) \equiv 1 - \lim_{\mathbf{q}\to 0} \frac{8\pi}{|\mathbf{q}|^2 \Omega} \sum_{vc\mathbf{k}} \left| \langle v\mathbf{k}-\mathbf{q}| e^{-i\mathbf{q}\mathbf{r}} |c\mathbf{k}\rangle \right|^2 \cfrac{1}{(\omega - a_1) - \cfrac{b_2^2}{(\omega - a_2) - \cfrac{b_3^2}{\cdots}}}.$$

Where the a's and b's are obtained iteratively from Lanczos algorithm

**BSHayTrs= -0.02000**
**Threshold for accuracy of the iterative process**
*Negative sign: average difference, over the energy range, of two consecutive approximations to the spectrum*
*Positive sign: maximum difference, over the energy range, of two consecutive approximations to the spectrum*

In addition to input parameters defined in (11a)

# (13a) Parallelism: Linear Response

$$\chi^0_{\mathbf{G}\mathbf{G}'}(\mathbf{q},\omega) = 2\sum_{c,v}\int_{BZ}\frac{d\mathbf{k}}{(2\pi)^3}\rho^*_{cv\mathbf{k}}(\mathbf{q},\mathbf{G})\rho_{cv\mathbf{k}}(\mathbf{q},\mathbf{G}')f_{v\mathbf{k}-\mathbf{q}}(1-f_{c\mathbf{k}})\times\left[\frac{1}{\omega+\epsilon_{v\mathbf{k}-\mathbf{q}}-\epsilon_{c\mathbf{k}}+i0^+}-\frac{1}{\omega+\epsilon_{c\mathbf{k}}-\epsilon_{v\mathbf{k}-\mathbf{q}}-i0^+}\right]$$

Q momenta (MPI q)

Xo bands (MPI c,v)

k momenta (MPI k)

Space DoF

$$\chi(\mathbf{q},\omega) = \left[I-\chi^0(\mathbf{q},\omega)v\right]^{-1}\chi^0(\mathbf{q},\omega)$$

```
X_all_q_ROLEs= "q k c v"   # [PARALLEL] CPUs roles (q,k,c,v)
X_all_q_CPU= "1 2 4 2"     # [PARALLEL] CPUs for each role
X_Threads= 4               # [OPENMP/GW] Number of threads
                           # for  response functions
X_all_q_LinAlg_INV = 32    # [PARALLEL] CPUs for matrix inv
```

num MPI tasks = 1 * 2 * 4 * 2
num threads/MPI-tasks = 4
Total num threads = 4 * ( 1 * 2 *4 *2)
MPI-c,v  best memory distribution
MPI-k  efficient, some mem repl
MPI-q  may lead to load unbalance
OpenMP  efficient, need extra mem

# (13b) Parallelism: Correlation part of self energy

$$\Sigma^c_{n\mathbf{k}}(\omega) = \langle n\mathbf{k}|\Sigma^c|n\mathbf{k}\rangle = i\sum_m \int_{BZ} \frac{d\mathbf{q}}{(2\pi)^3} \sum_{\mathbf{G},\mathbf{G}'} \frac{4\pi}{|\mathbf{q}+\mathbf{G}|^2} \rho_{nm}(\mathbf{k},\mathbf{q},\mathbf{G})\rho^*_{nm}(\mathbf{k},\mathbf{q},\mathbf{G}') \times \int d\omega' G^0_{m\mathbf{k}-\mathbf{q}}(\omega-\omega')\epsilon^{-1}_{\mathbf{G}\mathbf{G}'}(\mathbf{q},\omega')$$

QP states (MPI qp)

G bands (MPI b)

Q transferred momenta (MPI q)

Space DoF (OMP SE_Threads)

```
SE_ROLEs= "q qp b"    # [PARALLEL] CPUs roles (q,qp,b)
SE_CPU= "1 2 8"       # [PARALLEL] CPUs for each role
SE_Threads= 4         # [OPENMP/GW] Number of threads
                      # for  self-energy
```

num MPI tasks = 1 x 2 x 8
num threads/MPI-tasks = 4
Total num threads = 4 x ( 1 x 2 x 8)
MPI-b best memory distribution
MPI-qp no communication
MPI-q leads to load unbalance
OpenMP very efficient

## (14) IO: `yambo -V io`

```
StdoHash=  40          # [IO] Live-timing Hashes
DBsIOoff= "none"       # [IO] Space-separated list of DB with NO I/O.
                        DB=(DIP,X,HF,COLLs,J,GF,CARRIERs,W,SC,BS,ALL)
DBsFRAGpm= "none"      # [IO] Space-separated list of +DB to be FRAG and -DB NOT to be FRAG.
                        DB=(DIP,X,W,HF,COLLS,K,BS,QINDX,
#WFbuffIO              # [IO] Wave-functions buffered I/O
```

Parts of the WFs are stored by the node. Nodes communicate when these elements are needed. Memory heavy.

No ndb.* file is written. Example: DBsIOoff= “DIP” - ndb.dip_iR_and_P_fragment_* is not written, but stored in memory if Yambo needs it.

Fragments the database. Smaller files (e.g. ndb.em1s_fragment_*) are created instead of a large one (e.g. ndb.em1s).

Faster read/write operations in parallel runs

# Input file generation and command line interface (ypp -H)

**Lower case options are for input file generation** **(same upper case options as yambo)**

| | |
|---|---|
| -k <opt> | :BZ Grid generator [(k)pt,(q)pt,(s)hifted,(h)igh symmetry,(r)andom] |
| -q <opt> | :(g)enerate-modify/(m)erge quasi-particle databases |
| -i | :Wannier 90 interface |
| -b | :Read BXSF output generated by Wannier90 |
| -s <opt> | :Electrons [(w)ave,(d)ensity,(m)ag,do(s),(b)ands] |
| -e <opt> <opt> | :Excitons [(s)ort,(sp)in,(a)mplitude,(w)ave;qindx] |
| -f | :Free hole position [excitons plot] |
| -m | :BZ map fine grid to coarse |
| -w <opt> | :WFs:(p)erturbative SOC mapping or (c)onversion to new I/O format |
| -y | :Remove symmetries not consistent with an external perturbation |

**ypp_rt / ypp_nl specific**

| | |
|---|---|
| -t <opt> | :TD-polarization [(X)response] |

**ypp_nl specific**

| | |
|---|---|
| -u | :Non-linear response analysis |

**Use (ypp)**

```
$ ypp -F excitons.in -e w  qindx  -J BSE
$ ypp -F excitons.in -J “EXC_WFs,BSE” -C BSE_out
```

**Use (ypp_rt)**

```
$ ypp_rt -F abs.in -t X -J TD-SEX
$ ypp_rt -F abs.in -J “ABS,TD-SEX” -C TD-SEX_out
```

## (YPP 1) Postprocessing - exciton plot: `ypp -e w qindx`

```
excitons                # [R] Excitons
wavefunction            # [R] Wavefunction
Format= "x"             # Output format [(c)ube/(g)nuplot/(x)crysden]
Direction= "123"   # [rlu] [1/2/3] for 1d or [12/13/23] for 2d [123] for 3D
FFTGvecs= 30 Ry         # [FFT] Plane-waves
States= "1 - 3"         # Index of the BS state(s)
Degen_Step=   0.0100   eV   # Maximum energy separation of two degenerate states
% Cells
 5 |  5 |  1 |           # Number of cell repetitions in each direction (odd or 1)
%
% Hole
 0.00 | 3.44        | 0.00 |       # [cc] Hole position in unit cell
```

*Excitonic wavefunction does not have the periodicity of the e and h wavefunctions but is generally more extended, with a fictitious periodicity due to the k-points sampling*

# Input file generation and command line interface (yambo_rt / yambo_nl -H)

**Common to yambo_rt and yambo_nl**

-i :Initialization
-r :Coulomb potential
-b :Static Inverse Dielectric Matrix
-e :Evaluate Collisions
-v <opt> :Self-Consistent Potential
opt=(h)artree,(f)ock,(coh),(sex),(cohsex),(d)ef,(ip)
(h)artree can be combined with others, i.e. hf, hsex, ...

**Specific to yambo_rt**

-n <opt> :Real-time dynamics [opt=(p)ump or probe,(pp)ump & probe, (pn) n external fields]

**Specific to yambo_nl**

-u :Non-linear spectroscopy

**Example of practical use**

```
$ yambo -F Collisions.in -e -v hsex                    ⇒ Generate Collisions.in input file
$ yambo -F Collisions.in -J “COLL,EM1S” -C COLL_out    ⇒ Run yambo to compute Collisions (use EM1S db)
$ yambo -F TD-SEX.in -n p -v hsex                          ⇒ Generate TD-SEX.in input file
$ yambo -F TD-SEX.in -J “TD-SEX,COLL” -C TD-SEX_out        ⇒ Do a TD-SEX simulation loading the
                                                           collisions from COLL folder.
```

## (RT 0) TD-IP propagation with density matrix: `yambo_rt -n p`

$$i\hbar\partial_t \underline{\rho}_{\mathbf{k}}(t) = \left[\underline{h}^{\mathrm{rt}}_{\mathbf{k}}[\underline{\rho}] + \underline{U}^{\mathrm{ext}}_{\mathbf{k}}(t), \underline{\rho}_{\mathbf{k}}(t)\right] - i\Gamma_{\mathbf{k}}\underline{\rho}_{\mathbf{k}}$$

*All underlined quantities are matrices of size n^2 with n=RTbands(2) - RTbands(1)*
*See next slides for definitio of RTbands*

*The number of k-points is defined by the SAVE*

*Inverse dephasing time*

*The external field defined in the input file*

$$\underline{h}^{rt}_{\mathbf{k}}[\underline{\rho}] = \underline{h}^{eq}_{\mathbf{k}} + \Delta\underline{h}^{qp}_{\mathbf{k}} + \Delta\underline{\Sigma}^{Hxc}[\underline{\rho}]$$

*DFT Hamiltonian*

*QP corrections*

*Hxc_potential*

*This is exact for Sigma linear in rho*

**Collisions**

$$\Delta\underline{\Sigma}^{\mathrm{Hxc}}[\underline{\rho}] = \underline{\underline{K}}^{\mathrm{Hxc}} \cdot \underline{\rho}$$

# (RT 1) TD-IP propagation with density matrix: `yambo_rt -n p`

## (basic parameters)

```
negf                                  # [R] Real-Time dynamics
HXC_Potential= "NONE"                 # [SC] SC HXC Potential
% RTBands
 1 | 20 |                             # [RT] Bands
%
Integrator= "RK2"                     # [RT] Integrator.
PhLifeTime= 0.000000   fs             # [RT] Dephasing Time
RTstep=10.000000            as        # [RT] Real Time step length
NETime= 100.0000            fs        # [RT] Simulation Time
```

*Potential. “HARTREE” and “DFT” can be computed on-the-fly. For non local potentials, i.e. “HF”, “HSEX” a preliminary calculation of the collisions is needed (see next)*

*bands see [RT 0]*

*Integrator the propagation:*
*- step kind: EULER, EXP, INV*
*- number of steps: SIMPLE, RK2, HEUN*
*- more: RWA (analytical integration of IP term)*

*Dephasing time*

## (field parameters)

```
% Field1_Freq
 0.00   | 0.00  | eV                 # [RT Field1] Frequency
%
Field1_Int= 1.E4   kWLm2              # [RT Field1] Intensity
Field1_Width= 0.000000 fs             # [RT Field1] Width
Field1_kind= "DELTA"                  # [RT Field1] Kind
Field1_pol= "linear"                  # [RT Field1] Polarization
% Field1_Dir
 0.000000 | 1.000000 | 0.000000 |     # [RT Field1] Versor
%
```

*Field kind. Options are:*
*- DELTA: Delta pulse in time  - SIN: sin function*
*- QSSIN: sin convoluted with a Gaussian*
*- PULSE: sin convoluted with a time limited function*

*Polarization of the electric field*

*Frequency of the sin function for some pulses*

## (RT 2) Generation of collisions: `yambo_rt -e -v hsex`

**Collisions**

$\underline{\underline{K}}^{\mathrm{Hxc}}$

*Generate the BSE kernel in the "symmetry broken IBZ" later used to efficiently update the self-energy during the real-time propagation*

```
collisions                # [R] Eval the extended Collisions
% COLLBands
  1 | 20 |                # [COLL] Bands for the collisions
%
HXC_Potential= "HARTREE+SEX" # [SC] SC HXC Potential
HARRLvcs= 18475      RL   # [HA] Hartree     RL components
EXXRLvcs= 18475      RL   # [XX] Exchange RL components
CORRLvcs= 18475      RL   # [GW] Correlation RL components
…
Other variables related to the screening are present as for BSE
```

*compare with BSEbands (see 10a) Must be consistent with RTBands*

*Approximation chosen for the self-energy*

*Energy cutoff for Hartree, Exchange and Correlation part of the Self-energy*

- **HARRLvcs** *corresponds to* **BSENGexx** *since the Hartree interaction of two electrons corresponds to the exchange interaction of one electron and one hole*
- **EXXRLvcs** *and* **CORRLvcs** *corresponds to* **BSENGblk** *: the two in BSE cannot be defined independently*

## (RT 3) TD-SEX propagation with density matrix: `yambo_rt -n p -v hsex`

*The input file for the case with the collisions has the same input variables of the TD-IP run plus same the keywords for the HXC potential used in the file for the collisions*

```
HXC_Potential= "HARTREE+SEX"   # [SC] SC HXC Potential
HARRLvcs= 18475      RL    # [HA] Hartree     RL components
EXXRLvcs= 18475      RL    # [XX] Exchange RL components
CORRLvcs= 18475      RL    # [GW] Correlation RL components
```

## (RT 4) Post-processing: `ypp_rt -t X`

```
RealTime                    # [R] Real-Time Post-Processing
 RT_X                       # [R] Response functions Post-Processing
 Xorder=1                   # Max order of the response/exc
functions
 % EnRngeRt
  0.00     | 10.00     | eV      # Energy range
 %
 ETStpsRt=1001                  # Total Energy steps
 % TimeRange
  0.00     | 0.00     | fs      # Time-window where processing is done
 %
DampMode= "LORENTZIAN"          # Damping type ( NONE |
LORENTZIAN | GAUSSIAN )
DampFactor=  0.10000   eV    # Damping parameter
```

$$\hat{\chi}(\omega) = \frac{\mathbf{P}(\omega)}{\epsilon_0 \mathbf{E}(\omega)}$$

## (NL 0) Non-linear response: `yambo -u`

For the non-linear response, Yambo solves a Time-Dependent Schrodinger Equation for each k-point:

$$\mathrm{i}\hbar\partial_t \left|v_{m\mathbf{k}}\right\rangle = \left(h^{\mathrm{rt}}_{\mathbf{k}}[\rho] + \mathrm{i}\boldsymbol{\mathcal{E}}\cdot\tilde{\partial}_{\mathbf{k}}\right)\left|v_{m\mathbf{k}}\right\rangle$$

This equation is numerically integrated with the algorithm **NLintegrator** and time step **NLStep**.

The time-dependent valence bands: $\left|v_{i\mathbf{k}}(t)\right\rangle$

are expanded in the Kohn-Sham basis set as: $\left|v_{i\mathbf{k}}(t)\right\rangle = \sum_j c^j_{i,\mathbf{k}}(t)\left|u_{j\mathbf{k}}\right\rangle$

where the j-sum run over the **NLBands** range chosen in the calculation.

# (NL 1) TD-IP propagation with wave-function: `yambo_nl -u -V nl`

## (basic parameters)

```
nloptics                # [R NL] Non-linear optics
% NLBands
  4 | 5 |               # [NL] Bands
%
NLverbosity= "low"      # [NL] Verbosity level (low | high)
NLtime=55.000000        fs      # [NL] Simulation Time
NLintegrator= "CRANKNIC"        # [NL] Integrator
NLCorrelation= "IPA"    # [NL] Correlation
NLDamping= 0.100000     eV      # [NL] Damping
```

*Damping term*

*Simulation length. If you put -1 Yambo automatically sets the optical length for SHG*

## (field parameters)

```
% NLEnRange
 0.200000 | 8.000000 | eV          # [NL] Energy range
%
NLEnSteps=  1                # [NL] Energy steps
% ExtF_Dir
 0.000000 | 1.000000 | 0.000000 |         # [NL ExtF] Versor
%
ExtF_kind= "DELTA"           # [NL ExtF]
Kind(SIN|SOFTSIN|RES|ANTIRES|GAUSS|DELTA|QSSIN)
```

*Field kind. Options are:*
*- DELTA: Delta pulse in time (for linear resp.)*
*- SOFTSIN: sinusoidal field (for non-linear resp.)*

*Frequency range and steps of the sin function*

## (NL 2) Results Analysis: `ypp_nl -u`

### (linear optics)

```
nonlinear                    # [R] NonLinear Optics
Post-Processing
Xorder= 1                    # Max order of the response
% TimeRange
-1.000000 |-1.000000 | fs    # Time-window where
processing is done
%
ETStpsRt= 200                # Total Energy steps
% EnRngeRt
  0.00000 | 10.00000 | eV    # Energy range
%
DampMode= "LORENTZIAN"
DampFactor= 0.100000   eV        # Damping parameter
```

*Post-processing damping (only for linear optics)*

### (non linear optics)

```
nonlinear                    # [R] NonLinear Optics
Post-Processing
Xorder= 5                    # Max order of the response
% TimeRange
-1.000000 |-1.000000 | fs    # Time-window where
processing is done
%
ETStpsRt= 200                # Total Energy steps
% EnRngeRt
  0.00000 | 10.00000 | eV    # Energy range
%
DampMode= "NONE"
DampFactor= 0.100000   eV        # Damping parameter
```

*Response order (1 for linear optics, 4 or 5 for SHG)*

# (YP 1) Yambo-python

`code structure`

## yambopy - python module

### io

**YamboIn:** read, write and manipulate yambo input files
**YamboOut:** read yambo output files and save in .json

### analyse

**analyse:** read .json files generated with yamboout and plot them together
**recipes:** user contributed scripts

### bse

read exciton info from ypp runs and plotting tools

### dbs

**YamboLatticeDB:** read lattice parameters, symmetries and k-points from ns.db1
**YamboElectronsDB:** electronic states from ns.db1
**YamboDipolesDB:** dipole matrix elements from ndb.dip*
**YamboStaticScreeningDB:** static dielectric screening from ndb.em1s*
**YamboQPDB:** read the quasiparticle energies db ndb.QP
**YamboExcitonDB**: Read exciton info from yambo db ndb.BS
**YamboElectronPhononDB:** Read the electron-phonon matrix elements from ndb.gkkp
**YamboRTDB:** read the real time dbs ndb.RT
**(...)**

## qepy

**PwIn, PhIn, DynmatIn, ProjwfcIn**
read, write and manipulate Quantum espresso input files (pw.x, ph.x, dynmat.x, projwfc.x respectively)

**PwXML, ProjwfcXMl**
read output files (datafile.xml, datafile-schema.xml, projwfc.xml)

**Auxiliary tools for QE**

## schedulerpy

### scheduler

submit and run codes with the same interface for:
bash, pbs, oar

## yamboparser

parsing of files, databases, etc

## yambopy - scripts

analysebse, plotem1s, analysegw, mergeqp, test, plotexciton

# (YP 2) Yambo-python

quick-reference

## qepy

```
from qepy import *

#create input file from scratch
qe = PwIn()

#input structure
qe.atoms = [['Si',[0.125,0.125,0.125]],
            ['Si',[-.125,-.125,-.125]]]
qe.atypes = {'Si': [28.086,"Si.pbe-mt_fhi.UPF"]}

#control variables
qe.control['prefix'] = "'si'" #strings need
double "'"
qe.control['wf_collect'] = '.true.' #logicals

#system
qe.system['celldm(1)'] = 10.3
qe.system['ecutwfc'] = 30
qe.system['occupations'] = "'fixed'"
qe.system['nat'] = 2
qe.system['ntyp'] = 1
qe.system['ibrav'] = 2

#electrons
qe.electrons['conv_thr'] = 1e-8

#write file
qe.write('si.scf')
```

## yambopy

```
from yambopy import *

#create input file in 'bse' folder with SAVE
y = YamboIn.from_runlevel('yambo -b -o b -k sex -y d -V
all',folder='bse')

# define variables
y['FFTGvecs'] = [30,'Ry'] # scalar + units
y['BndsRnXs'] = [1,30] # array with integers
y['BSEBands'] = [3,6] # array with integers
y['BEnRange'] = [[0,8],'eV'] # array + units
y['BEnSteps'] = 500 # numbers
y['KfnQPdb'] = 'E < yambo/ndb.QP' #strings

#write the file
y.write('bse/yambo_run.in')

#create ypp input file
y = YamboIn.from_runlevel('ypp -e -a -V
all',filename='ypp.in')

#read local file
y = YamboIn.from_file(filename='bse/yambo_run.in')

#analyse data in the bse folder
ya = YamboAnalyser('bse')
print(ya)

# plot eel and eps from BSE
ya.plot_bse('eel')
ya.plot_bse('eps')
```

## schedulerpy

```
from schedulerpy import *

# scheduler 1 node and 4 cores
shell = Scheduler.factory(nodes=1,cores=4)

# scheduler of pbs type
shell = Scheduler.factory(scheduler='pbs')

#add commands to the shell
shell.add_command("echo 'hello world'")

#view commands on the screen
print( shell )

#write to a file
shell.write("commands.sh")

#submit or run the job
shell.run()

#clean scheduler commands
shell.clean()
```

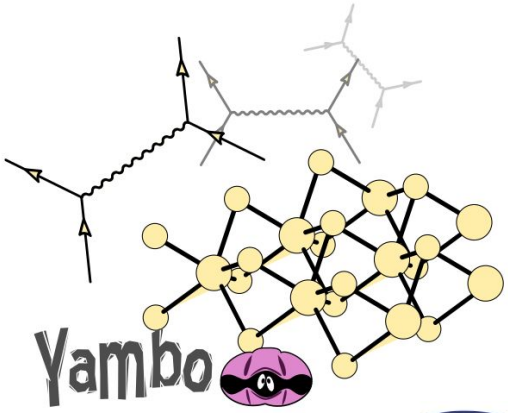
## yambopy (bash)

```
$ yambopy #lists all possible commands
$ yambopy plotem1s #help about this command
```

# Happy Yamboing!

**Computational School on Electronic Excitations in Novel Materials Using the Yambo Code**

ICTP Trieste, Italy *January 27, 2020 - January 31, 2020*




MAX

DRIVING THE EXASCALE TRANSITION

ICTP

The Abdus Salam
International Centre
for Theoretical Physics

UNESCO
United Nations
Educational, Scientific and
Cultural Organization

IAEA
International Atomic Energy Agency