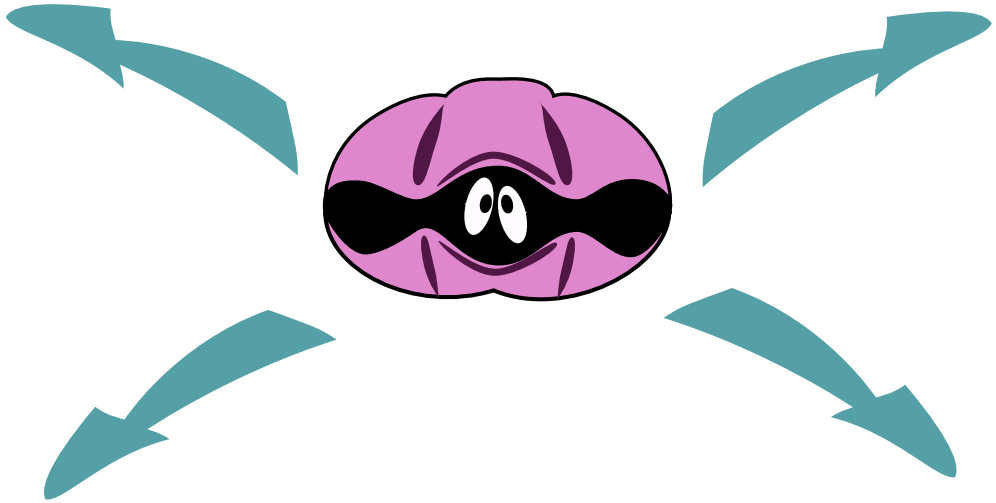
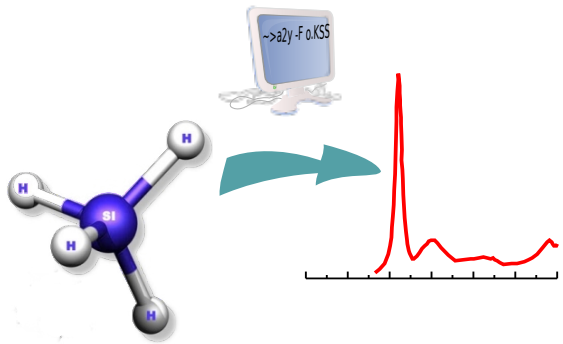
Step by step introduction to Yambo

the Yambo team



You will learn:

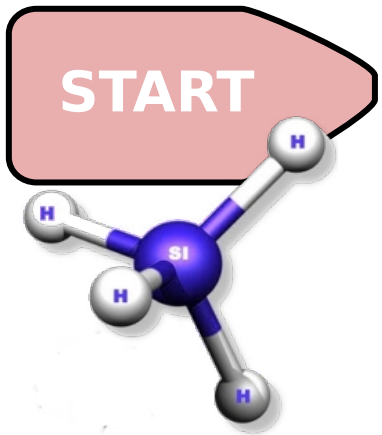


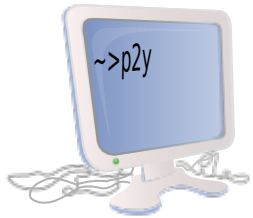
Flow of a Yambo calculation

Yambo command line options

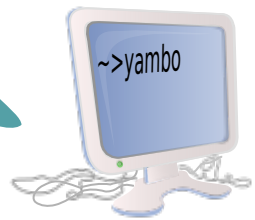

```
optics                     # [R OPT] Optics
chi                        # [R LR] Linear Response.
% QpntsRXd
  1 | 1 |                  # [Xd] Transferred momenta
%
% BndsRnXd
   1 |  10 |               # [Xd] Polarization function bands
%
NGsBlkXd= 1             RL  # [Xd] Response block size
% EnRngeXd
  7.50000 | 25.00000 |   eV  # [Xd] Energy range
%
% DmRngeXd
  0.10000 |  0.30000 |   eV  # [Xd] Damping range
%
ETStpsXd= 300               # [Xd] Total Energy steps
% LongDrXd
 1.000000 | 0.000000 | 0.000000 |      # [Xd] [cc] Electric Field
%
```

Yambo I/O files

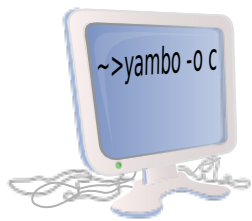




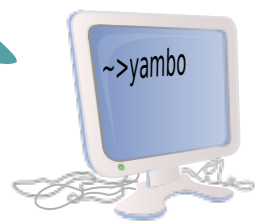
1. Generate the core databases



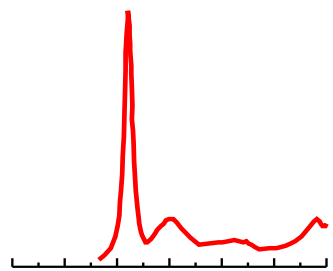
2. Run setup



3. Generate input



4. Run Yambo



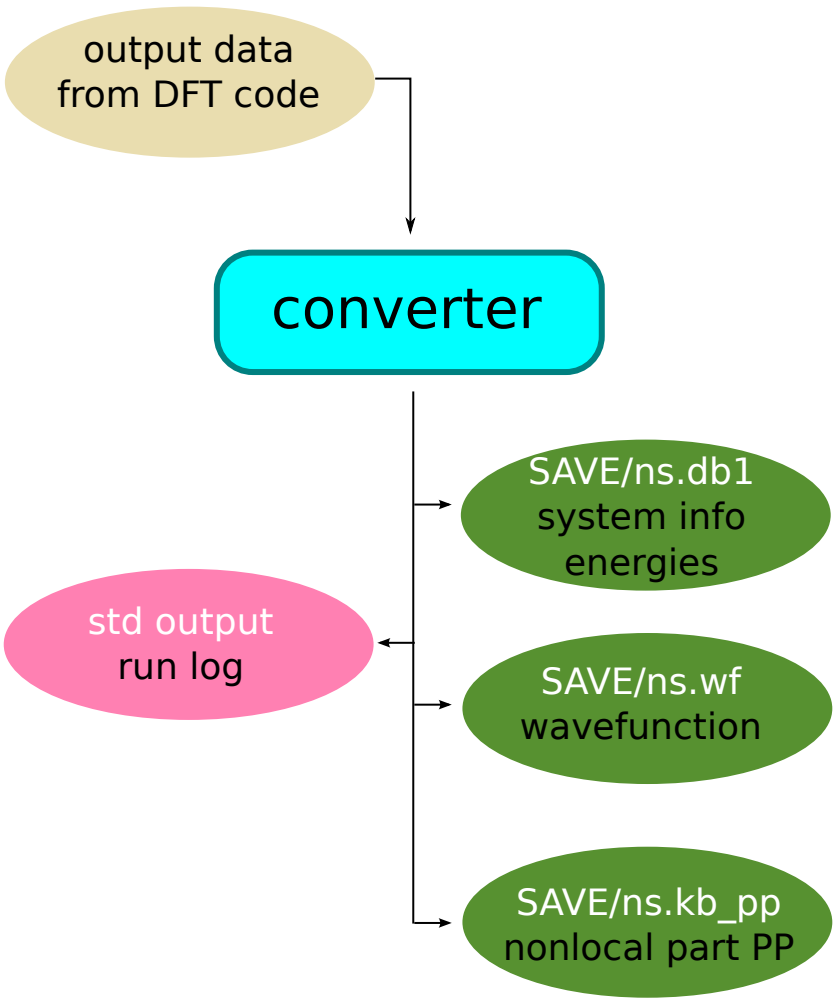
1. Generate the core databases

= convert data from standard ab initio DFT code (abinit and PWscf)

Use converters (a2y, p2y) from command line with options:

`-h`	Short Help
`-H`	Long Help
`-N`	Skip MPI initialization
`-F <opt>`	Input file name/prefix
`-O <opt>`	Output directory
`-S`	DataBases fragmentation
`-a <real>`	Lattice constants factor
`-t`	Force no TR symmetry
`-y`	Force no symmetries
`-w`	Force no wavefunctions

1. Generate the core databases



example: convert abinit KSS file with a2y

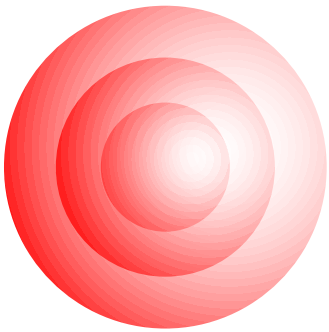
```
% ls
3D_LiF_o_DS2_KSS
```

```
% a2y -F 3D_LiF_o_DS2_KSS
 <---> [01] A(binit) 2 Y(ambo)
 <---> Checking input file ...3D_LiF_o_DS2_KSS
 <---> DBs path set to :.
 <---> KSS Header...abinit version 5.8.4
 <---> :: Atoms positions...done
 <---> Symmetries...[ID yes]...[INV no]...[TR yes]...[SI yes]
 <---> RL vectors...done
 <---> Energies...done
 <---> Report:
 <--->  :: Electrons            : 8.000000
 <--->  :: Temperature       [ev]: 0.000000
 <--->  :: Lattice factors [a.u.]: 7.703475  7.703475  7.703475
 <--->  :: K-points             : 10
 <--->  :: Bands                : 10
 <--->  :: Spinor components    : 1
 <--->  :: Spin polarizations   : 1
 <--->  :: Spin orbit coupling  : no
 <--->  :: Symmetries   [spatial]: 24
 <--->  ::              [T-rev]: yes
 <--->  :: Max WF components    : 1885
 <--->  :: RL vectors       (WF): 1885
 <--->  :: RL vectors    (CHARGE): 1885
 <--->  :: XC potential         : Perdew & Wang (xc)
 <--->  :: Atomic species       : 2
 <--->  :: Max atoms/species    : 1
 <---> == DB1 ...done ==
 <---> == DB2 + nlPP ...done ==
```

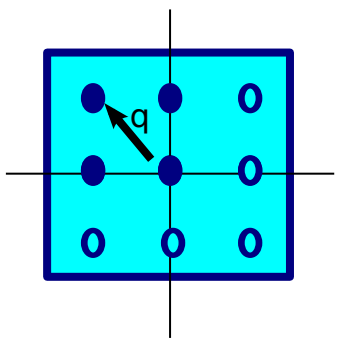
```
% ls SAVE
ns.db1         ns.kb_pp       ns.wf
```

2. Run setup

= prepare general purpose databases for later use

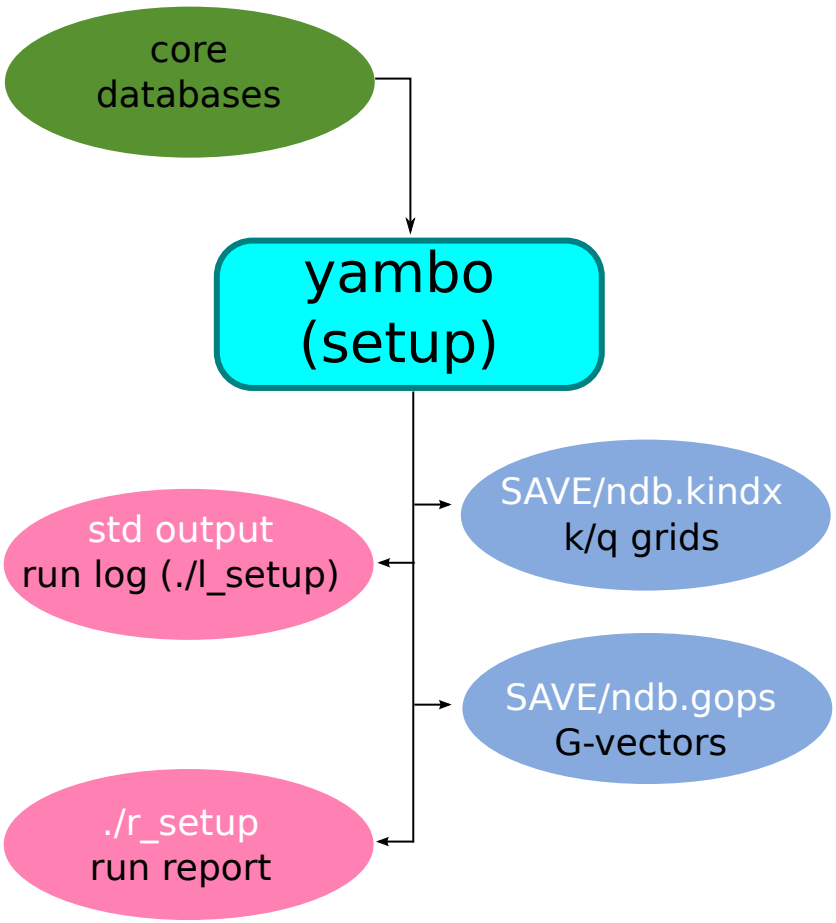


* **Data initialization:**
reorders G-vectors into spherical shells
calculates Fermi level and electronic occupations
sets up energy grids



* **Brillouin-zone sampling:**
expands k-points to full BZ
generates q-point meshes
checks on uniformity of grids

2. Run setup

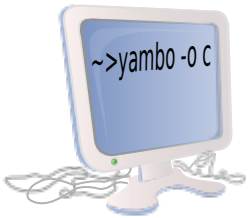


example: setup run

```
% ls SAVE/
ns.db1  ns.kb_pp  ns.wf
```

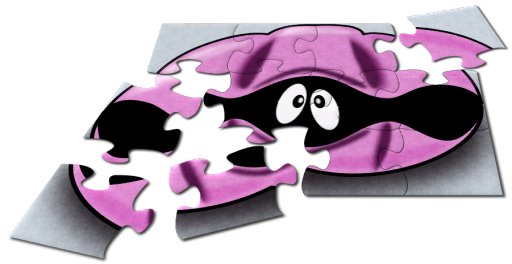
```
% yambo

 <---> [01] Job Setup
 <---> [02] Input variables setup
 <---> [02.01] Unit cells
 <---> [02.02] Symmetries
 <---> [02.03] RL shells
 <---> Shells finder |####################| [100%] --(E) --(X)
 <---> [02.04] K-grid lattice
 <---> [02.05] Energies [ev] & Occupations
 <---> [03] Transferred momenta grid
 <---> X indexes |####################| [100%] --(E) --(X)
 <---> SE indexes |####################| [100%] --(E) --(X)
 <---> [04] Game Over & Game summary
```

```
% ls
r_setup  SAVE
% ls SAVE
ndb.gops  ndb.kindx  ns.db1  ns.kb_pp  ns.wf
```



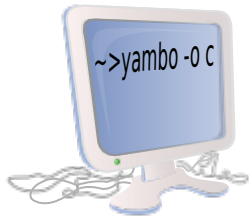
3. Generate input file

= select runlevel(s) and choose parameter values



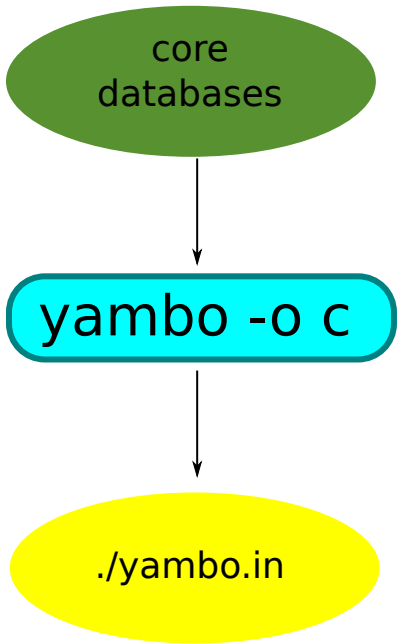
yambo acts as input file generator
when we add command line options

Calculation type	**Command line**
Setup	yambo -i
Optics in RPA	yambo -o c
TDDFT optics (ALDA in G-space)	yambo -t a -o c
TDDFT optics (ALDA in R-space)	yambo -t a -o b
Static screened interaction	yambo -b
BSE Kernel + solvers	yambo -o b -y hd
Exchange self-energy	yambo -x
Plasmon Pole GW	yambo -g n -p p



3. Generate input file

example: linear response (rpa level)

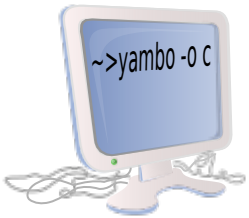


directly open in editor

```
optics                    # [R OPT] Optics
chi                       # [R CHI] Dyson equation for Chi.
% QpntsRXd
  1 | 19 |                # [Xd] Transferred momenta
%
% BndsRnXd
  1 | 10 |                # [Xd] Polarization function bands
%
NGsBlkXd= 1             RL  # [Xd] Response block size
% EnRngeXd
  0.00000 | 10.00000 |   eV  # [Xd] Energy range
%
% DmRngeXd
  0.10000 |  0.10000 |   eV  # [Xd] Damping range
%
ETStpsXd= 100               # [Xd] Total Energy steps
% LongDrXd
 1.000000 | 0.000000 | 0.000000 |      # [Xd] [cc] Electric Field
%
```

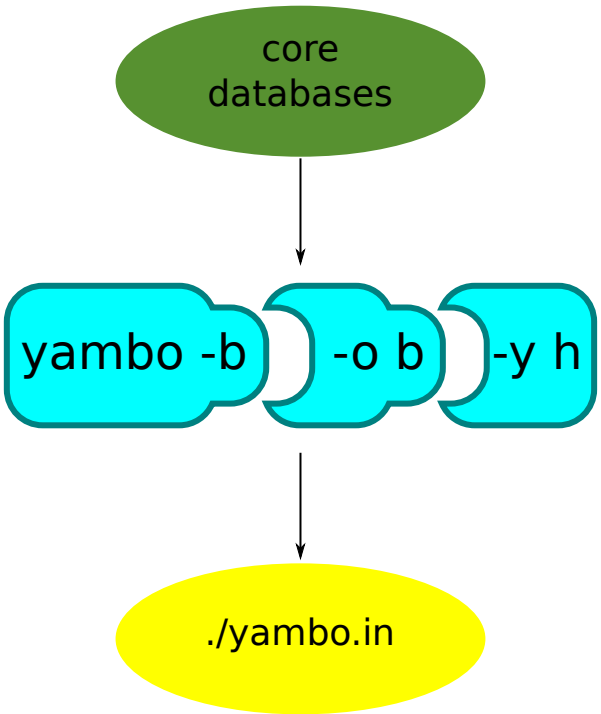
runlevels

default values from existing databases (compatibility)



3. Generate input file

example: linear response (BSE level)



combine runlevels

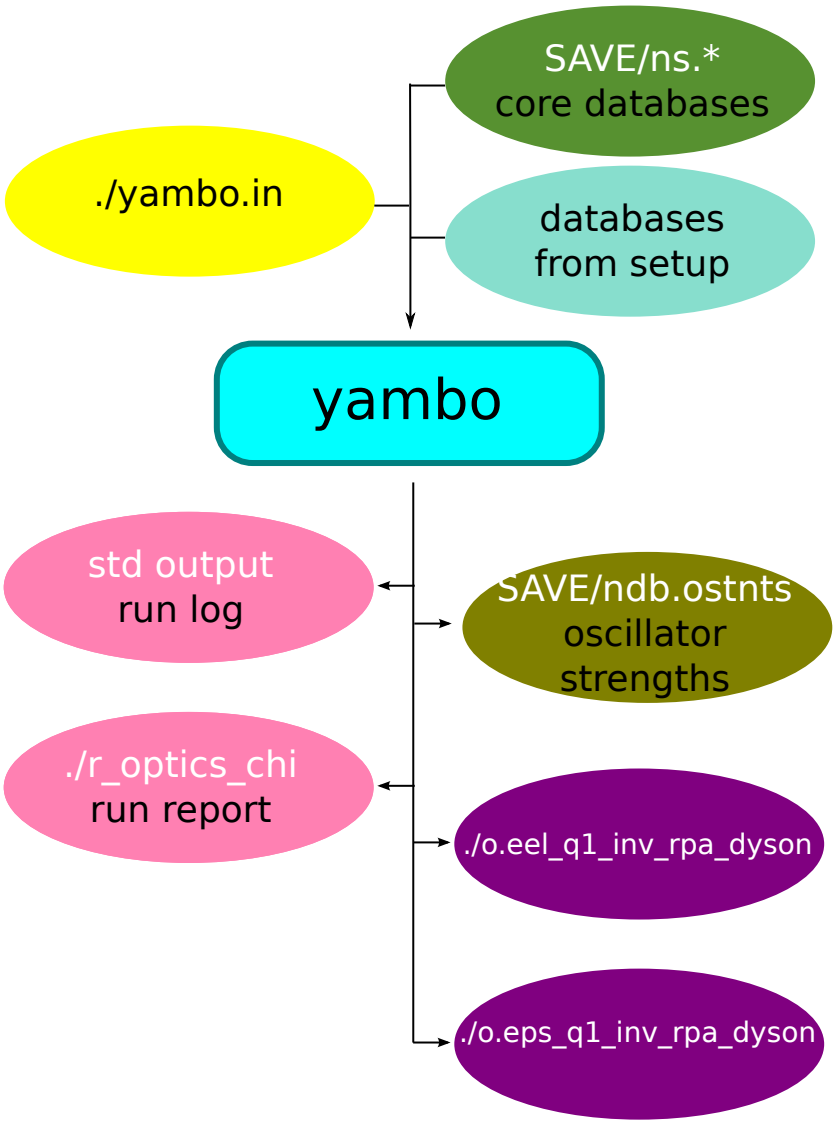
-b : static inverse dielectric matrix

-o b : BSE kernel

-y h : BSE solver (Haydock)

```
em1s                     # [R Xs] Static Inverse Dielectric Matrix
optics                   # [R OPT] Optics
bse                      # [R BSK] Bethe Salpeter Equation.
bss                      # [R BSS] Bethe Salpeter Equation solver
BSresKmod= "xc"            # [BSK] Resonant Kernel mode. (`x`;`c`;`d`)
% BSEBands
  1 | 10 |               # [BSK] Bands range
%
BSENGBlk= 1            RL  # [BSK] Screened interaction block size
BSENGexx= 1885         RL  # [BSK] Exchange components
% QpntsRXs
  1 | 19 |               # [Xs] Transferred momenta
%
% BndsRnXs
  1 | 10 |               # [Xs] Polarization function bands
%
NGsBlkXs= 1            RL  # [Xs] Response block size
% LongDrXs
 1.000000 | 0.000000 | 0.000000 |      # [Xs] [cc] Electric Field
%
BSSmod= "h"                # [BSS] Solvers `h/d/i/t`
% BEnRange
  0.00000 | 10.00000 |   eV  # [BSS] Energy range
%
% BDmRange
  0.10000 |  0.10000 |   eV  # [BSS] Damping range
%
BEnSteps= 100              # [BSS] Energy steps
% BLongDir
 1.000000 | 0.000000 | 0.000000 |      # [BSS] [cc] Electric Field
%
```

4. Run yambo



example: linear response (RPA)

```
% ls
r_setup  SAVE  yambo.in
```

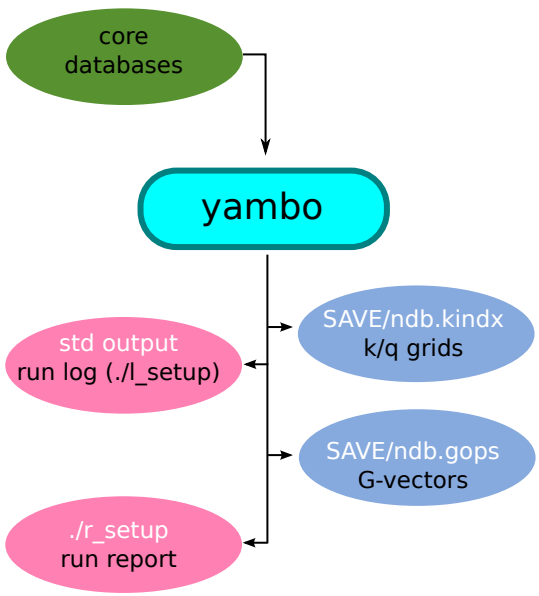
```
% yambo
 <---> [01] Job Setup
 <---> [02] CORE Variables Setup
 <---> [02.01] Unit cells
 <---> [02.02] Symmetries
 <---> [02.03] RL shells
 <---> [02.04] K-grid lattice
 <---> [02.05] Energies [ev] & Occupations
 <---> [03] Transferred momenta grid
 <---> [04] Optics
 <---> [WF-Oscillators/G space loader] Wfs (re)loading | [...]
 <---> Dipole (T): |####################| [100%] --(E) --(X)
 <---> [FFT-X] Mesh size: 18  18  18
 <---> [WF-X loader] Wfs (re)loading |#####| [100%] --(E) --(X)
 <---> [X-CG] R(p) Tot o/o(of R)  :   222   6144    100
 <---> [X] Upper matrix triangle filled
 <---> Xo@q[1] 1-100 |####################| [100%] --(E) --(X)
 <---> X @q[1] 1-100 |####################| [100%] --(E) --(X)
 <---> [05] Game Over & Game summary
```

```
% ls
o.eel_q1_inv_rpa_dyson  o.eps_q1_inv_rpa_dyson
r_optics_chi  r_setup  SAVE  yambo.in
```

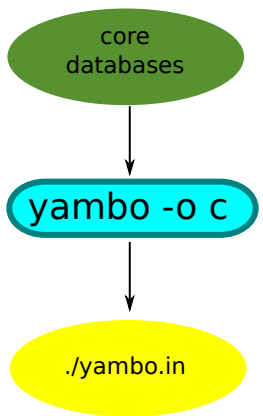
```
% ls SAVE
ndb.gops  ndb.kindx  ndb.ostnts  ns.db1  ns.kb_pp  ns.wf
```

To remember: one source many execs

run setup

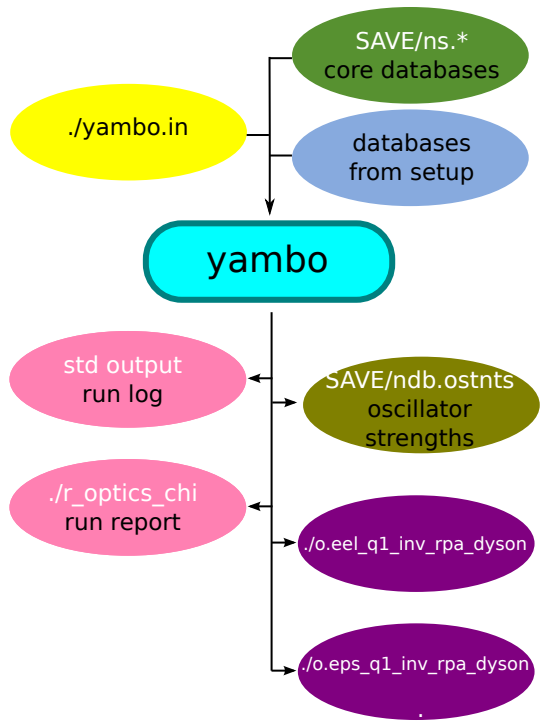


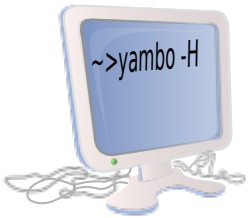
generate input



select runlevel(s)

run specific task(s)





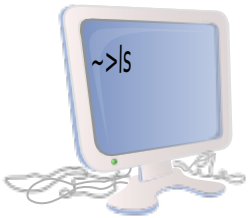
To remember: command line options

`-h`	Short Help
`-H`	Long Help
`-J <opt>`	Job string identifier
`-V <int>`	Input file verbosity
`-F <opt>`	Input file
`-I <opt>`	Core I/O directory
`-O <opt>`	Additional I/O directory
`-C <opt>`	Communications I/O directory
`-N`	Skip MPI initialization
`-D`	DataBases properties
`-S`	DataBases fragmentation
`-i`	Initialization
`-o <opt>`	Optics [opt=(c)hi/(b)se]
`-t <opt>`	The TDDFTs [opt=(a)LDA/(l)RC]
`-c`	Coulomb interaction
`-x`	Hartree-Fock Self-energy and Vxc
`-b`	Static Inverse Dielectric Matrix
`-p <opt>`	GW approximations [opt=(p)PA]
`-y <opt>`	BS equation solver [opt=h/d]

display list of options

uppercase options added at run-time to specify I/O, MPI initialization, etc.

lowercase options drive input file generator



To remember: yambo I/O files

generated by the converters
geometry, basis set,
energies, wavefunctions

generated during setup run
k/q grids, G-vectors

generated at run-time
info specific to a runlevel

SAVE/ns.*
core databases

SAVE/ndb.*
stable databases

SAVE/ndb.*
job-dependent
databases

yambo

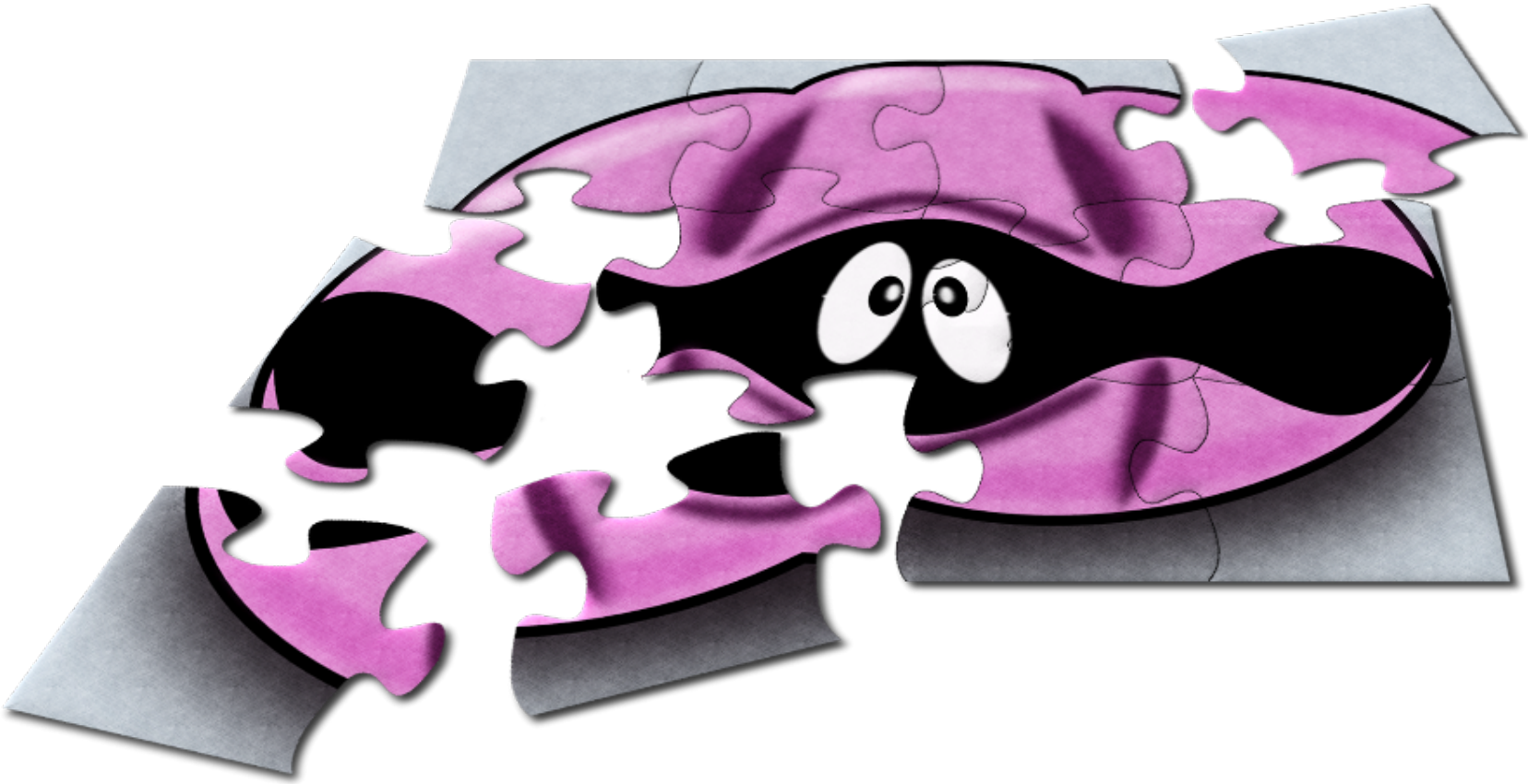
./yambo.in

./o.*
output files

./l_* ./r_*
run-logs and
reports

generated by yambo+command line options
select runlevel(s)
parameter values

generated at the end of a runlevel
for human use:
e.g. plot or inspection



the Yambo team

1. Many-body perturbation theory calculations using the yambo code
   Journal of Physics: Condensed Matter 31, 325902 (2019)
2. Yambo: an ab initio tool for excited state calculations
   Comp. Phys. Comm. 144, 180 (2009)