OAK RIDGE National Laboratory

Nexus Monthly User and Developer Meeting

Meeting 3: Generating PWSCF Input

Jaron T. Krogel

18 January 2019

ORNL is managed by UT-Battelle, LLC for the US Department of Energy

U.S. DEPARTMENT OF ENERGY

Agenda for this Meeting

- Demos:
  - Difference between generate_pwscf and generate_pwscf_input
  - Keywords accepted by generate_pwscf_input and examples
  - Input generation method 1: direct composition w/ PwscfInput
  - Input generation method 2: read from a file, then manipulate
  - Input generation method 3: direct composition w/ generate_pwscf_input
  - Input generation method 4: composition assisted by generate_physical_system

Files located at: `nexus_training/monthly_meetings/03_190118_pwscf_input/`

Demo:
generate_pwscf and generate_pwscf_input

Difference between generate_pwscf and generate_pwscf_input

01_diamond_generate_write.py

```
#! /usr/bin/env python

from nexus import settings,job,run_project
from nexus import generate_physical_system
from nexus import generate_pwscf
from nexus import generate_pwscf_input

settings(
    pseudo_dir    = './pseudopotentials',
    status_only   = 0,
    generate_only = 0,
    sleep         = 3,
    machine       = 'ws16'
    )

dia16 = generate_physical_system(
    units  = 'A',
    axes   = [[ 1.785,  1.785,  0.   ],
              [ 0.   ,  1.785,  1.785],
              [ 1.785,  0.   ,  1.785]],
    elem   = ['C','C'],
    pos    = [[ 0.    ,  0.    ,  0.    ],
              [ 0.8925,  0.8925,  0.8925]],
    tiling = (2,2,2),
    kgrid  = (1,1,1),
    kshift = (0,0,0),
    C      = 4
    )

# most familiar
scf = generate_pwscf(
    # Nexus Simulation class inputs
    identifier   = 'scf',
    path         = 'diamond/scf',
    job          = job(cores=16,app='pw.x'),
    # PwscfInput class inputs
    input_type   = 'generic',
    calculation  = 'scf',
    input_dft    = 'lda',
    ecutwfc      = 200,
    conv_thr     = 1e-8,
    nosym        = True,
    wf_collect   = True,
    system       = dia16,         # shared by Simulation/PwscfInput
    pseudos      = ['C.BFD.upf'], # shared by Simulation/PwscfInput
    )
```

```
print
print '===================='
print 'overview of contents'
print '===================='
print repr(scf.input) # PwscfInput class


#equivalent to the above
scf_input = generate_pwscf_input(
    # PwscfInput class inputs
    selector     = 'generic',
    calculation  = 'scf',
    input_dft    = 'lda',
    ecutwfc      = 200,
    conv_thr     = 1e-8,
    nosym        = True,
    wf_collect   = True,
    system       = dia16,
    pseudos      = ['C.BFD.upf'],
    )

scf = generate_pwscf(
    # Nexus Simulation class inputs
    identifier   = 'scf',
    path         = 'diamond/scf2',
    job          = job(cores=16,app='pw.x'),
    pseudos      = ['C.BFD.upf'], # pseudos also here for file transfer
    system       = dia16,         # system info cached in sim object
    input        = scf_input,
    )

print
print '============================='
print 'actually write the input file'
print '============================='
text = scf_input.write() # write to string
print text
scf_input.write('./scf_01.in') # write to file



# no need to run, input demo
#run_project()
```

Accepts all PWSCF input keywords

Difference between generate_pwscf and generate_pwscf_input

Running the demo

```
./01_diamond_generate_write.py

====================
overview of contents
====================
  atomic_positions      atomic_positions
  atomic_species        atomic_species
  cell_parameters       cell_parameters
  control               control
  electrons             electrons
  k_points              k_points
  system                system
```

Creates file: scf_01.in

```
============================
actually write the input file
============================
&CONTROL
   calculation     = 'scf'
   outdir          = 'pwscf_output'
   prefix          = 'pwscf'
   pseudo_dir      = './'
   wf_collect      = .true.
/

&SYSTEM
   celldm(1)       = 1.0
   ecutwfc         = 200
   ibrav           = 0
   input_dft       = 'lda'
   nat             = 2
   nosym           = .true.
   nspin           = 1
   ntyp            = 1
   tot_charge      = 0
/

&ELECTRONS
   conv_thr        = 1e-08
/

ATOMIC_SPECIES
   C  12.011 C.BFD.upf

ATOMIC_POSITIONS alat
   C        0.00000000       0.00000000       0.00000000
   C        1.68658058       1.68658058       1.68658057

K_POINTS crystal
   8
         0.00000000       0.00000000       0.00000000       0.12500000
         0.50000000      -0.00000000      -0.00000000       0.12500000
        -0.00000000       0.50000000       0.00000000       0.12500000
         0.50000000       0.50000000      -0.00000000       0.12500000
        -0.00000000      -0.00000000       0.50000000       0.12500000
         0.50000000      -0.00000000       0.50000000       0.12500000
        -0.00000000       0.50000000       0.50000000       0.12500000
         0.50000000       0.50000000       0.50000000       0.12500000

CELL_PARAMETERS cubic
         3.37316115       3.37316115       0.00000000
        -0.00000000       3.37316115       3.37316115
         3.37316115      -0.00000000       3.37316115
```

Demo:
generate_pwscf_input keywords

Keywords accepted by generate_pwscf_input and examples

Running the demo

```
./02_generate_input_keywords.py


pw = generate_pwscf_input(
    selector = 'generic',
    # what can go here?
    )


#most of the answer:


==================================
PWSCF namelist variables supported:
==================================
  a  adaptive_thr  angle1  angle2  assume_isolated  b  bfgs_ndim  block  block_1
  block_height  c  calculation  cell_dofree  cell_dynamics  cell_factor  celldm
  comp_thr  constrained_magnetization  conv_thr  conv_thr_init  conv_thr_multi
  cosac  cosbc  degauss  delta_t  dftd3_threebody  dftd3_version
  diago_david_ndim  diago_full_acc  diago_thr_init  diagonalization  dipfield
  do_ee  ds  dt  eamp  ecfixed  ecutcoarse  ecutfock  ecutrho  ecutvcut  ecutwfc
  efield  efield_cart  efield_phase  electron_maxstep  emaxpos  eopreg  esm_bc
  esm_nfit  esm_w  etot_conv_thr  exx_fraction  exxdiv_treatment  fcp_mu
  fe_step  first_last_opt  fixed_magnetization  forc_conv_thr  force_symmorphic
  gate  gdir  hubbard_alpha  hubbard_beta  hubbard_j  hubbard_j0  hubbard_u
  input_dft  ion_dynamics  ion_positions  ion_temperature  iprint  k_max  k_min
  lambda  lberry  lda_plus_u  lda_plus_u_kind  lelfield  lfcpopt  lforcet
  london  london_c6  london_rcut  london_rvdw  london_s6  lorbm  lspinorb
  mixing_beta  mixing_charge_compensation  mixing_fixed_ns  mixing_mode
  modenum  multiplicity  n_charge_compensation  nat  nberrycyc  nbnd  neldw
  nelup  nlev  no_t_rev  noinv  noncolin  nosym  nosym_evc  nppstr  nqx1  nqx2
  nr1  nr1s  nr2  nr2s  nr3  nr3s  nraise  nspin  nstep  ntyp  num_of_images
  one_atom_occupations  opt_scheme  origin_choice  ortho_para  outdir  path_thr
  pot_extrapolation  prefix  press  press_conv_thr  pseudo_dir  q2sigma  qcutz
  relaxz  remove_rigid_rot  report  restart_mode  rhombohedral  scf_must_converge
  smearing  space_group  starting_charge  starting_magnetization
  starting_spin_angle  startingpot  startingwfc  sw_nstep  tefield  temp_req
  title  tolp  tot_charge  tot_magnetization  tprnfor  tqr  trust_radius_ini
  trust_radius_min  ts_vdw_econv_thr  ts_vdw_isolated  tstress  u_projection_type
  upscale  use_all_frac  use_freezing  use_masses  vdw_corr  verbosity  w_1  w_2
  wfc_extrapolation  wfcdir  which_compensation  wmass  x_gamma_extrapolation
  xdm_a1  xdm_a2  xqq  zgate
```

Keywords accepted by generate_pwscf_input and examples

Running the demo

```
==============================
Namelist variables by namelist
(compare to INPUT_PW.html)
==============================

control
-------------------
  calculation  dipfield  disk_io  dt  etot_conv_thr  forc_conv_thr  gate  gdir
  lberry  lelfield  lfcpopt  lkpoint_dir  lorbm  max_seconds  nberrycyc  nppstr
  outdir  prefix  pseudo_dir  restart_mode  tefield  title  tprnfor  tstress
  wf_collect  wfcdir

system
-------------------
  a  angle1  angle2  assume_isolated  b  block  block_1  block_2  block_height  c
  constrained_magnetization  cosab  cosac  cosbc  degauss  dftd3_threebody
  do_ee  eamp  ecfixed  ecutfock  ecutrho  ecutvcut  ecutwfc  edir  emaxpos
  esm_bc  esm_efield  esm_nfit  esm_w  exx_fraction  exxdiv_treatment  fcp_mu
  force_symmorphic  hubbard_alpha  hubbard_beta  hubbard_j  hubbard_j0  hubbard_u
  input_dft  la2f  lambda  lda_plus_u  lda_plus_u_kind  lforcet  london
  london_rcut  london_rvdw  london_s6  lspinorb  multiplicity  nat  nbnd  neldw
  nelup  no_t_rev  noinv  noncolin  nosym  nosym_evc  nqx1  nqx2  nqx3  nr1  nr1s
  nr2s  nr3  nr3s  nspin  ntyp  occupations  one_atom_occupations  origin_choice
  qcutz  relaxz  report  rhombohedral  screening_parameter  smearing  space_group
  starting_magnetization  starting_ns_eigenvalue  starting_spin_angle  tot_charge
  ts_vdw_econv_thr  ts_vdw_isolated  u_projection_type  uniqueb  use_all_frac
  x_gamma_extrapolation  xdm  xdm_a1  xdm_a2  zgate

electrons
-------------------
  adaptive_thr  conv_thr  conv_thr_init  conv_thr_multi  diago_cg_maxiter
  diago_full_acc  diago_thr_init  diagonalization  efield  efield_cart
  electron_maxstep  mixing_beta  mixing_fixed_ns  mixing_mode  mixing_ndim
  scf_must_converge  startingpot  startingwfc  tqr

ions
-------------------
  bfgs_ndim  ci_scheme  delta_t  ds  fe_nstep  fe_step  first_last_opt
  ion_dynamics  ion_positions  ion_temperature  k_max  k_min  nraise
  opt_scheme  path_thr  phase_space  pot_extrapolation  refold_pos
  sw_nstep  temp_req  tempw  tolp  trust_radius_ini  trust_radius_max
  upscale  use_freezing  use_masses  w_1  w_2  wfc_extrapolation
```

```
cell
-------------------
  cell_dofree  cell_dynamics  cell_factor  press
  press_conv_thr  wmass

phonon
-------------------
  modenum  xqq

ee
-------------------
  comp_thr  ecutcoarse  mixing_charge_compensation
  n_charge_compensation  nlev
```

Keywords accepted by generate_pwscf_input and examples

Running the demo

```
=================
Breakdown by type
=================

integers:
---------
  bfgs_ndim  dftd3_version  diago_cg_maxiter  diago_david_ndim  edir
  esm_nfit  fe_nstep  gdir  ibrav  iprint  lda_plus_u_kind  mixing_fixed_ns
  modenum  multiplicity  n_charge_compensation  nat  nberrycyc  nbnd  nlev
  nqx1  nqx2  nqx3  nr1  nr1s  nr2  nr2s  nr3  nr3s  nraise  nspin  nstep  ntyp
  origin_choice  ortho_para  report  space_group  sw_nstep

floats:
-------
  a  angle1  angle2  b  block_1  block_2  block_height  c  cell_factor  celldm
  conv_thr  conv_thr_init  conv_thr_multi  cosab  cosac  cosbc  degauss  delta_t
  ds  dt  eamp  ecfixed  ecutcoarse  ecutfock  ecutrho  ecutvcut  ecutwfc  efield
  emaxpos  eopreg  esm_efield  esm_w  etot_conv_thr  exx_fraction  fcp_mu
  fixed_magnetization  forc_conv_thr  g_amplitude  hubbard_alpha  hubbard_beta
  hubbard_j  hubbard_j0  hubbard_u  k_max  k_min  lambda  london_c6  london_rcut
  london_s6  max_seconds  mixing_beta  mixing_charge_compensation  neldw  nelec
  path_thr  press  press_conv_thr  q2sigma  qcutz  screening_parameter
  starting_magnetization  starting_ns_eigenvalue  temp_req  tempw  tolp
  tot_magnetization  trust_radius_ini  trust_radius_max  trust_radius_min
  upscale  w_1  w_2  wmass  xdm_a1  xdm_a2  xqq  zgate

strings:
--------
  assume_isolated  calculation  cell_dofree  cell_dynamics  ci_scheme
  diagonalization  disk_io  efield_phase  esm_bc  exxdiv_treatment  input_dft
  ion_positions  ion_temperature  mixing_mode  occupations  opt_scheme  outdir
  pot_extrapolation  prefix  pseudo_dir  restart_mode  smearing  startingpot
  title  u_projection_type  vdw_corr  verbosity  wfc_extrapolation  wfcdir
```

```
booleans:
---------
  adaptive_thr  block  dftd3_threebody  diago_full_acc  dipfield  do_ee
  force_symmorphic  gate  la2f  lberry  lda_plus_u  lelfield  lfcpopt  lforcet
  london  lorbm  lspinorb  no_t_rev  noinv  noncolin  nosym  nosym_evc
  refold_pos  relaxz  remove_rigid_rot  rhombohedral  scf_must_converge
  tefield  tprnfor  tqr  ts_vdw_isolated  tstress  uniqueb  use_all_frac
  use_masses  wf_collect  x_gamma_extrapolation  xdm

arrays:
------
  angle1  angle2  celldm  efield_cart  fe_step  fixed_magnetization
  hubbard_beta  hubbard_j  hubbard_j0  hubbard_u  london_c6  london_rvdw
  starting_ns_eigenvalue
```

Keywords accepted by generate_pwscf_input and examples

Running the demo

```
==============
Valid examples
==============

integers:
---------
generate_pwscf_input(
    nspin = 2,
    nbnd  = 63,
    )


floats:
---------
generate_pwscf_input(
    ecutwfc     = 350.0,
    mixing_beta = 0.5,
    )


strings:
--------
generate_pwscf_input(
    calculation = 'scf',
    input_dft   = 'lda',
    )
```

```
booleans:
---------
generate_pwscf_input(
    lda_plus_u = True,
    wf_collect = False,
    )


arrays:
-------
generate_pwscf_input(
    hubbard_u              = {1 : 3.1},
    starting_magnetization = {1 : 0.9},
    starting_ns_eigenvalue = {(1,2,1) : 0.0,
                              (2,2,1) : 0.0476060,
                              (3,2,1) : 0.0476060,
                              (4,2,1) : 0.9654373,
                              (5,2,1) : 0.9954307},
    )

generate_pwscf_input(
    hubbard_u              = obj(V1=3.5,V2=3.5),
    starting_magnetization = obj(V1=1.0,V2=-1.0),
    )
```

Demo:
Input generation method 1:
Direct composition with PwscfInput

Pattern: Compose-Manipulate-Write

Input generation method 1: direct composition w/ PwscfInput

Native PwscfInput class closely mimicks PWSCF input structure

Card data (e.g. ATOMIC_SPECIES) requires added (keyword) structure

03_compose_input.py

```
#! /usr/bin/env python

from numpy import array
from generic import obj
from pwscf_input import PwscfInput

# directly compose input in Nexus internal representation

pw = PwscfInput()
pw.control.set(
    calculation   = 'scf',
    restart_mode  = 'from_scratch',
    wf_collect    = True,
    outdir        = './output',
    pseudo_dir    = '../pseudo/',
    prefix        = 'fe',
    etot_conv_thr = 1.0e-9,
    forc_conv_thr = 1.0e-6,
    tstress       = True,
    tprnfor       = True,
    )
pw.system.set(
    ibrav            = 1,
    celldm           = { 1 : 15 },
    nat              = 2,
    ntyp             = 1,
    ecutwfc          = 100,
    ecutrho          = 300,
    nbnd             = 18,
    occupations      = 'smearing',
    degauss          = 0.0005,
    smearing         = 'methfessel-paxton',
    nspin            = 2,
    assume_isolated = 'martyna-tuckerman',
    lda_plus_u       = True,
    hubbard_u        = { 1 : 3.1 },
    starting_magnetization = { 1 : 0.9 },
    starting_ns_eigenvalue = { (1,2,1) : 0.0,
                               (2,2,1) : 0.0476060,
                               (3,2,1) : 0.0476060,
                               (4,2,1) : 0.9654373,
                               (5,2,1) : 0.9954307},
    )
```

```
pw.electrons.set(
    conv_thr        = 1.0e-9,
    mixing_beta     = 0.7,
    diagonalization = 'david',
    mixing_fixed_ns = 500,
    )
pw.atomic_species.set(
    atoms            = ['Fe'],
    masses           = obj(Fe=58.69000),
    pseudopotentials = obj(Fe='Fe.pbe-nd-rrkjus.UPF'),
    )
pw.atomic_positions.set(
    specifier = 'angstrom',
    atoms     = ['Fe','Fe'],
    positions = array([
        [2.070000000,   0.000000000,   0.000000000],
        [0.000000000,   0.000000000,   0.000000000],
        ]),
    )
pw.k_points.set(
    specifier = 'automatic',
    grid      = (1,1,1),
    shift     = (1,1,1),
    )


# print internal representation
print
print 'internal representation'
print pw  # this works for all PwscfInput's, however obtained

# manipulate and write
for ecut in [100,120,140,160]:
    pw.system.ecutwfc = ecut
    pw.write('scf_03_ecut_{0}.in'.format(ecut))
#end for
```

Difference between generate_pwscf and generate_pwscf_input

Running the demo

```
./03_compose_input.py

internal representation
  atomic_positions
    atoms           = ['Fe', 'Fe']
    positions       = [[2.07 0.   0.  ]
                       [0.   0.   0.  ]]
    specifier       = angstrom
  end atomic_positions
  atomic_species
    atoms           = ['Fe']
    specifier       =
    masses
      Fe              = 58.69
    end masses
    pseudopotentials
      Fe              = Fe.pbe-nd-rrkjus.UPF
    end pseudopotentials
  end atomic_species
  control
    calculation     = scf
    etot_conv_thr   = 1e-09
    forc_conv_thr   = 1e-06
    outdir          = ./output
    prefix          = fe
    pseudo_dir      = ../pseudo/
    restart_mode    = from_scratch
    tprnfor         = True
    tstress         = True
    wf_collect      = True
  end control
  electrons
    conv_thr        = 1e-09
    diagonalization = david
    mixing_beta     = 0.7
    mixing_fixed_ns = 500
  end electrons
  k_points
    grid            = (1, 1, 1)
    shift           = (1, 1, 1)
    specifier       = automatic
  end k_points
  system
    assume_isolated = martyna-tuckerman
    celldm          = {1: 15}
    degauss         = 0.0005
    ecutrho         = 300
    ecutwfc         = 100
    hubbard_u       = {1: 3.1}
    ibrav           = 1
    lda_plus_u      = True
    nat             = 2
    nbnd            = 18
    nspin           = 2
    ntyp            = 1
    occupations     = smearing
    smearing        = methfessel-paxton
    starting_magnetization = {1: 0.9}
    starting_ns_eigenvalue = {(2, 2, 1): 0.047606,
                              (1, 2, 1): 0.0,
                              (5, 2, 1): 0.9954307,
                              (4, 2, 1): 0.9654373,
                              (3, 2, 1): 0.047606}
  end system
```

Creates files:

- scf_03_ecut_100.in
- scf_03_ecut_120.in
- scf_03_ecut_140.in
- scf_03_ecut_160.in

Demo:
Input generation method 2:
Read from a file, then manipulate

Pattern: Read-Manipulate-Write

Input generation method 2: read from a file, then manipulate

04_read_input.py

```
#! /usr/bin/env python

from pwscf_input import PwscfInput
from nexus import read_input

# read the input from a file
pw = PwscfInput('./Fe_start_ns_eig.in')


# reading this way also works
pw = read_input('./Fe_start_ns_eig.in',format='pwscf')


# manipulate and write
for ecut in [100,120,140,160]:
    pw.system.ecutwfc = ecut
    pw.write('scf_04_ecut_{0}.in'.format(ecut))
#end for
```

Creates files:

- scf_04_ecut_100.in
- scf_04_ecut_120.in
- scf_04_ecut_140.in
- scf_04_ecut_160.in

Fe_start_ns_eig.in

```
&CONTROL
  calculation   = 'scf' ,
  restart_mode  = 'from_scratch' ,
  wf_collect    = .true. ,
  outdir        = './output' ,
  pseudo_dir    = '../pseudo/' ,
  prefix        = 'fe' ,
  etot_conv_thr = 1.0D-9 ,
  forc_conv_thr = 1.0D-6 ,
  tstress       = .true. ,
  tprnfor       = .true. ,
/

&SYSTEM
  ibrav           = 1,
  celldm(1)       = 15,
  nat             = 2,
  ntyp            = 1,
  ecutwfc         = 100 ,
  ecutrho         = 300 ,
  nbnd            = 18,
  occupations     = 'smearing' ,
  degauss         = 0.0005 ,
  smearing        = 'methfessel-paxton' ,
  nspin           = 2 ,
  assume_isolated = 'martyna-tuckerman'
  lda_plus_u      = .true. ,
  Hubbard_U(1)    = 3.1,
  starting_magnetization(1) = 0.9,
  starting_ns_eigenvalue(1,2,1) = 0.0
  starting_ns_eigenvalue(2,2,1) = 0.0476060
  starting_ns_eigenvalue(3,2,1) = 0.0476060
  starting_ns_eigenvalue(4,2,1) = 0.9654373
  starting_ns_eigenvalue(5,2,1) = 0.9954307
/

 &ELECTRONS
   conv_thr        = 1.0e-9 ,
   mixing_beta     = 0.7 ,
   diagonalization = 'david' ,
   mixing_fixed_ns = 500,
 /

ATOMIC_SPECIES
   Fe   58.69000  Fe.pbe-nd-rrkjus.UPF

ATOMIC_POSITIONS angstrom
   Fe      2.070000000    0.000000000    0.000000000
   Fe      0.000000000    0.000000000    0.000000000

K_POINTS automatic
  1 1 1   1 1 1
```

Demo:
Input generation method 3:
Direct composition w/ generate_pwscf_input

Pattern: Generate-Manipulate-Write

Input generation method 3: direct composition w/ generate_pwscf_input

05_generate_input.py

```
#! /usr/bin/env python

from nexus import obj
from nexus import generate_pwscf_input

# generate using only keywords

pw = generate_pwscf_input(
    selector        = 'generic',
    # control inputs
    calculation     = 'scf',
    restart_mode    = 'from_scratch',
    wf_collect      = True,
    outdir          = './output',
    pseudo_dir      = '../pseudo/',
    prefix          = 'fe',
    etot_conv_thr   = 1.0e-9,
    forc_conv_thr   = 1.0e-6,
    tstress         = True,
    tprnfor         = True,
    # system inputs
    ibrav           = 1,
    celldm          = { 1 : 15 },
    nat             = 2,
    ntyp            = 1,
    ecutwfc         = 100,
    ecutrho         = 300,
    nbnd            = 18,
    occupations     = 'smearing',
    degauss         = 0.0005,
    smearing        = 'methfessel-paxton',
    nspin           = 2,
    assume_isolated = 'martyna-tuckerman',
    lda_plus_u      = True,
    hubbard_u       = { 1 : 3.1 },
    starting_magnetization = { 1 : 0.9 },
    starting_ns_eigenvalue = {(1,2,1) : 0.0,
                              (2,2,1) : 0.0476060,
                              (3,2,1) : 0.0476060,
                              (4,2,1) : 0.9654373,
                              (5,2,1) : 0.9954307},
    # electrons inputs
    conv_thr        = 1.0e-9,
    mixing_beta     = 0.7,
    diagonalization = 'david',
    mixing_fixed_ns = 500,
    # atomic_species inputs
    mass            = obj(Fe=58.69000),
    pseudos         = ['Fe.pbe-nd-rrkjus.UPF'],
    # atomic_positions inputs
    elem            = ['Fe','Fe'],
    pos             = [[2.070000000, 0.000000000, 0.000000000],
                       [0.000000000, 0.000000000, 0.000000000]],
    pos_specifier   = 'angstrom',
    # k_points inputs
    kgrid           = (1,1,1),
    kshift          = (1,1,1),
    )


# manipulate and write
for ecut in [100,120,140,160]:
    pw.system.ecutwfc = ecut
    pw.write('scf_05_ecut_{0}.in'.format(ecut))
#end for
```

Creates files:

- scf_05_ecut_100.in
- scf_05_ecut_120.in
- scf_05_ecut_140.in
- scf_05_ecut_160.in

Demo:
Input generation method 4:
Composition assisted by generate_physical_system

Pattern: Generate-Manipulate-Write

Input generation method 4: composition assisted by generate_physical_system

06_generate_input_with_system.py

```
#! /usr/bin/env python

from nexus import obj
from nexus import read_structure
from nexus import generate_physical_system
from nexus import generate_pwscf_input

# generate using keywords + system

# read structure from file
s = read_structure('VO2_M1_afm.xsf')
s.elem[0] = 'V1' # set AFM pattern
s.elem[1] = 'V2'
s.elem[2] = 'V1'
s.elem[3] = 'V2'

# create physical system from structure
vo2 = generate_physical_system(
    structure = s,
    V1        = 13,
    V2        = 13,
    O         =  6,
    )
```

Creates files:

- scf_06_ecut_100.in
- scf_06_ecut_120.in
- scf_06_ecut_140.in
- scf_06_ecut_160.in

```
# generate pwscf input
pw = generate_pwscf_input(
    selector          = 'generic',
    calculation       = 'scf',
    disk_io           = 'low',
    verbosity         = 'high',
    wf_collect        = True,
    input_dft         = 'lda',
    hubbard_u         = obj(V1=3.5,V2=3.5),
    ecutwfc           = 350,
    bandfac           = 1.3,
    nosym             = True,
    occupations       = 'smearing',
    smearing          = 'fermi-dirac',
    degauss           = 0.0001,
    nspin             = 2,
    start_mag         = obj(V1=1.0,V2=-1.0),
    diagonalization   = 'david',
    conv_thr          = 1e-8,
    mixing_beta       = 0.2,
    electron_maxstep  = 1000,
    system            = vo2,
    pseudos           = ['V.opt.upf','O.opt.upf'],
    kgrid             = (6,6,6),
    kshift            = (0,0,0),
    )

# manipulate and write
for ecut in [100,120,140,160]:
    pw.system.ecutwfc = ecut
    pw.write('scf_06_ecut_{0}.in'.format(ecut))
#end for
```

Input generation method 4: composition assisted by generate_physical_system

scf_06_ecut_100.in

```
&CONTROL
   calculation     = 'scf'
   disk_io         = 'low'
   outdir          = 'pwscf_output'
   prefix          = 'pwscf'
   pseudo_dir      = './'
   verbosity       = 'high'
   wf_collect      = .true.
/

&SYSTEM
   celldm(1)       = 1.0
   degauss         = 0.0001
   ecutwfc         = 100
   Hubbard_U(2)    = 3.5
   Hubbard_U(3)    = 3.5
   ibrav           = 0
   input_dft       = 'lda'
   lda_plus_u      = .true.
   nat             = 12
   nbnd            = 65
   nosym           = .true.
   nspin           = 2
   ntyp            = 3
   occupations     = 'smearing'
   smearing        = 'fermi-dirac'
   starting_magnetization(2) = 1.0
   starting_magnetization(3) = -1.0
   tot_charge      = 0
/

&ELECTRONS
   conv_thr        = 1e-08
   diagonalization = 'david'
   electron_maxstep = 1000
   mixing_beta     = 0.2
/
```

```
ATOMIC_SPECIES
   O  15.999 O.opt.upf
   V1 50.942 V.opt.upf
   V2 50.942 V.opt.upf

ATOMIC_POSITIONS alat
   V1       2.45778327       8.39460555       0.22661828
   V2       2.92496303       0.18059370       8.33794247
   V1      -0.28569594       4.46819332       4.50889865
   V2       5.66844224       4.10700593       4.05566210
   O        0.00978311       1.81708472       1.78656736
   O        5.37296317       6.75811453       6.77799337
   O       -2.73369610       2.47051490       6.06884775
   O        8.11644240       6.10468435       2.49571300
   O        2.71381355       6.02493499       2.55909075
   O        2.66893273       2.55026426       6.00547000
   O       -0.02966566       6.83786388       6.84137114
   O        5.41241194       1.73733537       1.72318961

K_POINTS automatic
   6 6 6  0 0 0

CELL_PARAMETERS cubic
        10.86970472       0.00000000       0.00000000
         0.00000000       8.57519925       0.00000000
        -5.48695844       0.00000000       8.56456075
```

Question & Answer + Updates

- Q&A for Nexus features, workflow how to's, code details, etc
- Updates on current development

Next Monthly Meeting

- Date/Time: Friday Feb. 15th 1-2pm EST
- BlueJeans link: https://bluejeans.com/190169126
- Tentative topic: Excited state calculations with QMCPACK
- Feature demo and/or topic requests welcome!