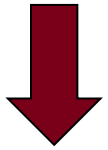
Running a PSCF Calculation: Commands, Inputs, and Outputs

Ryan P. Collanton

2023/08/24

University of Minnesota
Driven to Discover®

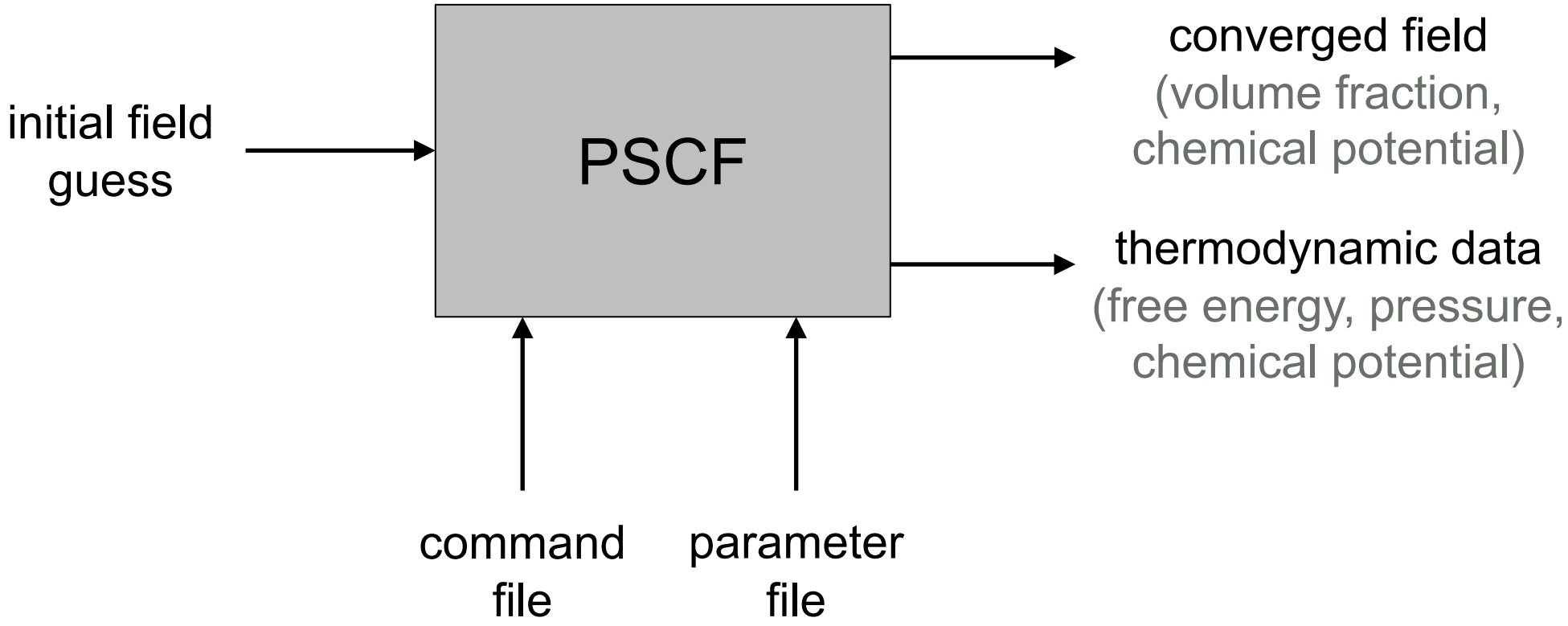
Our Example



`…/pscf_workshop/lam/`

A diblock lamellar phase with an initial chemical potential field guess generated using the level set method.

A simulation: basic components and workflow



The command and its arguments

CPU: `pscf_pcX -p param -c command`

GPU: `pscf_pgX -p param -c command`

`X`	1, 2, 3 (number of dimensions)
`param`	path to param file
`command`	path to command file

Command File

command

```
RGRID_TO_BASIS        in/c.rf    in/c.bf
ESTIMATE_W_FROM_C     in/c.bf
ITERATE
WRITE_W_BASIS         out/w.bf
WRITE_C_RGRID         out/c.rf
WRITE_PARAM           out/data
WRITE_THERMO          out/data
FINISH
```

Common commands for performing simulations

Commands used to read an input field guess, run a calculation (or many), and write outputs.

```
READ_W_(BASIS/RGRID)
ESTIMATE_W_FROM_C
COMPUTE
ITERATE
SWEEP
WRITE_(W/C)_(BASIS/RGRID)
WRITE_PARAM
WRITE_THERMO
```

Common commands for performing workflow operations

Commands used to convert field formats, compare fields, get more detailed output, or get crystallographic information.

```
COMPARE_(BASIS/RGRID)
BASIS_TO_RGRID
RGRID_TO_BASIS

WRITE_C_BLOCK_RGRID
WRITE_Q
WRITE_Q_SLICE

CHECK_RGRID_SYMMETRY
WRITE_WAVES
WRITE_STARS
WRITE_GROUPS
```

Parameter File

param

```
System{
  Mixture{
    nMonomer        2
    monomers[
                    1.0
                    1.0
    ]
    nPolymer        1
    Polymer{
      type            linear
      nBlock          2
      blocks[
                      0  0.5
                      1  0.5
      ]
      phi             1.0
    }
    ds              0.01
  }
  Interaction{
    chi(
           1   0   20.0
    )
  }
  Domain{
    mesh            32
    lattice         lamellar
    groupName       P_-1
  }
  AmIterator{
    epsilon         1e-8
    maxItr          200
    maxHist         20
    isFlexible      1
  }
}
```

Syntax and Organization

Nested blocks

```
System{

Mixture{
        …
        }
        …
}
```

- Delineated with curly brackets
- Not sensitive to whitespace, including new lines and indentation
- *Order of blocks matters!* (corresponds directly with the underlying C++ code)
- Some blocks required, some optional

Arrays and Matrices

$$v = [a \quad b]$$

```
v[
        0
a
        1
b
]
```

$$A = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix}$$

```
A(
        0  0
a00
        0  1
a01
        1  0
a10
        1  1
)
```

param

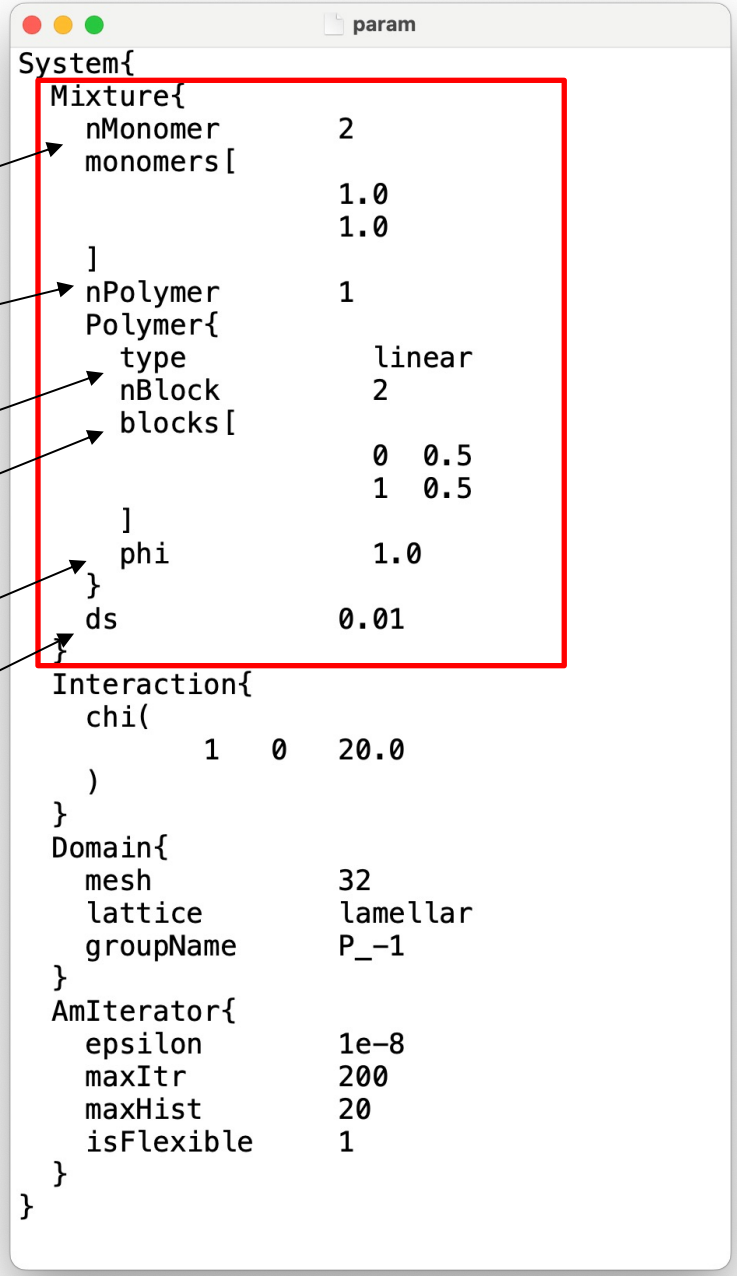
```
System{
  Mixture{
    nMonomer        2
    monomers[
                    1.0
                    1.0
    ]
    nPolymer        1
    Polymer{
      type            linear
      nBlock          2
      blocks[
                      0  0.5
                      1  0.5
      ]
      phi             1.0
    }
    ds              0.01
  }
  Interaction{
    chi(
            1   0   20.0
    )
  }
  Domain{
    mesh            32
    lattice         lamellar
    groupName       P_-1
  }
  AmIterator{
    epsilon         1e-8
    maxItr          200
    maxHist         20
    isFlexible      1
  }
}
```

Required Blocks – Mixture Block

Specifies the polymer system!

- Number of monomers and their statistical segment lengths
- Number of polymers
- Polymer architecture (linear or general branched)
- Blocks and block lengths
- Volume fraction (must sum to unity)
- Chain discretization size

Number of polymer blocks set by nPolymer. Can also add solvent. See documentation



Required Blocks – Interaction Block

Specifies interactions between monomers in terms of the Flory-Huggins segregation strength, χN

- Definition in case of multiple polymers and block lengths available in notes
- Assumes self-interactions (diagonal) are zero unless otherwise specified
- Defined to be symmetric, such that $(\chi N)_{ij} = (\chi N)_{ji}$. Only need to enter one.

param

```
System{
  Mixture{
    nMonomer        2
    monomers[
                    1.0
                    1.0
    ]
    nPolymer        1
    Polymer{
      type            linear
      nBlock          2
      blocks[
                      0  0.5
                      1  0.5
      ]
      phi             1.0
    }
    ds              0.01
  }
  Interaction{
    chi(
          1   0   20.0
    )
  }
  Domain{
    mesh            32
    lattice         lamellar
    groupName       P_-1
  }
  AmIterator{
    epsilon         1e-8
    maxItr          200
    maxHist         20
    isFlexible      1
  }
}
```

Required Blocks – Domain Block

Specifies crystal structure and unit cell details.

- Spatial discretization
- Lattice type (lamellar, square, rhombic, cubic, tetragonal, triclinic, etc)
- Space group name

Unit cell parameters (a, b, …) are specified in the inputted field file!

param

```
System{
  Mixture{
    nMonomer        2
    monomers[
                    1.0
                    1.0
    ]
    nPolymer        1
    Polymer{
      type            linear
      nBlock          2
      blocks[
                      0  0.5
                      1  0.5
      ]
      phi             1.0
    }
    ds              0.01
  }
  Interaction{
    chi(
            1   0   20.0
    )
  }
  Domain{
    mesh            32
    lattice         lamellar
    groupName       P_-1
  }
  AmIterator{
    epsilon         1e-8
    maxItr          200
    maxHist         20
    isFlexible      1
  }
}
```

Optional Blocks – Iterator Block

Adds an iterator to the program. Multiple options available, and in continuing development.

- Iterator{ … } defaults to an Anderson-Mixing (AM) iterator, also available with AmIterator{ … }
- Only necessary if an ITERATE command is present in the command file

Parameters differ between iterators, but generally…

- Error tolerance
- Maximum number of iterations
- Maximum number of histories to use when generating the next guess (AM specific)
- Whether to optimize unit cell parameters!

param

```
System{
  Mixture{
    nMonomer        2
    monomers[
                    1.0
                    1.0
    ]
    nPolymer        1
    Polymer{
      type            linear
      nBlock          2
      blocks[
                      0  0.5
                      1  0.5
      ]
      phi             1.0
    }
    ds              0.01
  }
  Interaction{
    chi(
            1   0   20.0
    )
  }
  Domain{
    mesh            32
    lattice         lamellar
    groupName       P_-1
  }
  AmIterator{
    epsilon         1e-8
    maxItr          200
    maxHist         20
    isFlexible      1
  }
}
```

Optional Blocks – Sweep Block

New example! Multicomponent system with a "parameter sweep".

A parameter sweep *varies system parameters* and *solves the SCFT equations* at each state point. The solution at the previous state point is used to construct the guess for the current state point.

- Number of sweep steps to take
- Where to save sweep output
- Number of parameters that will be varied
- Names and indices to specify which parameters to vary
- Total amount to vary the parameter, spaced linearly in the number of steps specified above

param

```
System{
  Mixture{
     nMonomer  3
     monomers[
               6.07
               6.07
               6.07
     ]
     nPolymer  1
     nSolvent  1
     Polymer{
        type    linear
        nBlock  2
        blocks[
                0   0.35
                1   0.65
        ]
        phi     0.5
     }
     Solvent{
        monomerId  2
        size       0.02
        phi        0.5
     }
     ds   0.01
  }
  Interaction{
    chi(
          1   0   30.0
          2   0   30.0
          2   1   100.0
    )
  }
  Domain{
    mesh           100
    lattice        lamellar
    groupName      P_-1
  }
  AmIterator{
    epsilon      1.0e-11
    maxItr       200
    maxHist      30
    isFlexible   1
  }
  LinearSweep{
    ns             4
    baseFileName   out/
    nParameter     2
    parameters[
          phi_polymer  0  +0.20
          phi_solvent  0  -0.20
    ]
  }
}
```

Field Files

Field files describe any spatial field.

- Concentration/volume fraction fields
- Chemical potential fields

Can be either inputted (initial guess, fixed external potential) or outputted (converged results)!

Three Equivalent Formats

*rgrid – *field values stored on a real space grid*

kgrid – *field values stored on a frequency space grid*

*basis – *field values stored as coefficients of symmetry-adapted basis functions*

* generally most useful!

Iteration and Thermodynamic Details

```
ITERATE


 Iteration     0,  error  =   1.4122308e-01
 Iteration     1,  error  =   1.3482938e-01
 Iteration     2,  error  =   9.7931173e-02
 Iteration     3,  error  =   6.4262320e-02
 Iteration     4,  error  =   7.5150859e-02
 Iteration     5,  error  =   1.6385768e-02
 Iteration     6,  error  =   1.7624596e-02
 Iteration     7,  error  =   2.6099101e-02
 Iteration     8,  error  =   1.6438435e-02
 Iteration     9,  error  =   2.1455000e-02
 Iteration    10,  error  =   1.4755512e-02
```

```
fHelmholtz      2.98467885480e+00
pressure        2.88047397940e+00

Free energy components:
fIdeal          1.10420488804e+00
fInter          1.88047396676e+00

Polymers:
               phi                    mu
    0   1.00000000000e+00   5.86515283420e+00

Lattice parameters:
        1.65130644948e+00
```

Outputting fields and data to a file

command

```
RGRID_TO_BASIS       in/c.rf    in/c.bf
ESTIMATE_W_FROM_C    in/c.bf
ITERATE
WRITE_W_BASIS        out/w.bf
WRITE_C_RGRID        out/c.rf
WRITE_PARAM          out/data
WRITE_THERMO         out/data
FINISH
```

param

```
    maxHist 10
    isFlexible   1
  }
  LinearSweep{
    ns             4
    baseFileName   out/
    writeCRGrid    1
    writeCBasis    1
    writeWRGrid    1
    nParameter     2
    parameters[
           block   0 0 -0.08
           block   0 1 +0.08
    ]
  }
}
```

You tell the program what you want out!

- Specify general output in the command file!
- Output of a sweep specified in the param file (quirky)

Processing and analyzing output…

- We have some tools to help with post-processing and analysis!
- Purpose-built scripts are usually the best option

Let’s run the example!

University of Minnesota

Driven to Discover®

Crookston Duluth Morris Rochester Twin Cities