Electronic Structure Calculation using Plane Wave Basis Set

Fadjar Fathurrahman

Contents

1 Introduction

`ffr-ElectronicStructure.jl` using plane wave basis set.

2 PWGrid_xx.jl

`xx` stands for `01`, `02`, and `03`.

2.1 PWGrid_01.jl

In this file, a type `PWGrid` is defined:

```
type PWGrid
  Ns::Array{Int64}
  LatVecs::Array{Float64,2}
  RecVecs::Array{Float64,2}
  Npoints::Int
  Ω::Float64
  r::Array{Float64,2}
  G::Array{Float64,2}
  G2::Array{Float64}
end
```

The fields of this type are:

- `Ns` is an integer array which defines number of sampling points in each lattice vectors.
- `LatVecs` is 3×3 matrix which defines lattice vectors of unit cell in real space.
- `RecVecs` is 3×3 matrix which defines lattice vectors of unit cell in reciprocal space. It is calculated according to (2).
- `Npoints` Total number of sampling points
- `Ω` Unit cell volume in real space
- `r` Real space grid points

- `G` **G**-vectors
- `G2` Magnitude of **G**-vectors

Constructor for `PWGrid` is defined as follow.

```
function PWGrid( Ns::Array{Int,1},LatVecs::Array{Float64,2} )
  Npoints = prod(Ns)
  RecVecs = 2*pi*inv(LatVecs')
  Ω = det(LatVecs)
  R,G,G2 = init_grids( Ns, LatVecs, RecVecs )
  return PWGrid( Ns, LatVecs, RecVecs, Npoints, Ω, R, G, G2 )
end
```

The function `init_grid()` is defined as follow. It takes `Ns`, `LatVecs`, and `RecVecs` as the arguments.

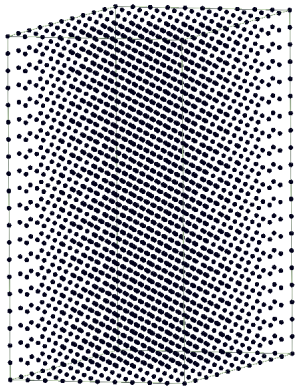
```
function init_grids( Ns, LatVecs, RecVecs )
```

First, grid points in real space are initialized:

```
  #
  Npoints = prod(Ns)
  #
  r = Array(Float64,3,Npoints)
  ip = 0
  for k in 0:Ns[3]-1
  for j in 0:Ns[2]-1
  for i in 0:Ns[1]-1
    ip = ip + 1
    r[1,ip] = LatVecs[1,1]*i/Ns[1] + LatVecs[2,1]*j/Ns[2]
              + LatVecs[3,1]*k/Ns[3]
    r[2,ip] = LatVecs[1,2]*i/Ns[1] + LatVecs[2,2]*j/Ns[2]
              + LatVecs[3,2]*k/Ns[3]
    r[3,ip] = LatVecs[1,3]*i/Ns[1] + LatVecs[2,3]*j/Ns[2]
              + LatVecs[3,3]*k/Ns[3]
  end
  end
  end
```

In the next step, grid points in reciprocal space, or **G**-vectors and also their squared values are initialized

```
  G  = Array(Float64,3,Npoints)
  G2 = Array(Float64,Npoints)
  ip    = 0
  for k in 0:Ns[3]-1
  for j in 0:Ns[2]-1
  for i in 0:Ns[1]-1
    gi = mm_to_nn( i, Ns[1] )
    gj = mm_to_nn( j, Ns[2] )
    gk = mm_to_nn( k, Ns[3] )
    ip = ip + 1
    G[1,ip] = RecVecs[1,1]*gi + RecVecs[2,1]*gj + RecVecs[3,1]*gk
    G[2,ip] = RecVecs[1,2]*gi + RecVecs[2,2]*gj + RecVecs[3,2]*gk
    G[3,ip] = RecVecs[1,3]*gi + RecVecs[2,3]*gj + RecVecs[3,3]*gk
    G2[ip] = G[1,ip]^2 + G[2,ip]^2 + G[3,ip]^2
  end
  end
  end
```



The function `mm_to_nn` defines mapping from real space to Fourier space:

```
function mm_to_nn(mm::Int,S::Int)
  if mm > S/2
    return mm - S
  else
    return mm
  end
end
```

Finally, the variables `r`, `G`, and `G2` are returned.

```
  return r,G,G2
```

We give an example of creating a `PWGrid` object:

```
Ns = [40, 40, 40]
LatVecs = 10*diagm(ones(3))
pw = PWGrid( Ns, LatVecs )
```

3 Visualizing real-space grid points

In the directory `pwgrid_01`, we visualize grid points in real space using Xcrysden program. Originally Xcrysden, is meant to visualize crystalline structure, however, we also can use it to visualize grid points, taking periodic boundary conditions into consideration.

4 Formulae

Plane wave basis $b_\alpha(\mathbf{r})$:

$$b_\alpha(\mathbf{r}) = \frac{1}{\sqrt{\Omega}} e^{\mathbf{G}_\alpha \cdot \mathbf{r}} \tag{1}$$

Lattice vectors of unit cell in reciprocal space:

$$\mathbf{b} = 2\pi \left(a^T\right)^{-1} \tag{2}$$

G-vectors:

$$\mathbf{G} = i\mathbf{b}_1 + j\mathbf{b}_2 + k\mathbf{b}_3 \tag{3}$$

Structure factor:

$$S_I(\mathbf{G}) = \sum_{\mathbf{G}} e^{-\mathbf{G}\cdot\mathbf{X}_I} \tag{4}$$