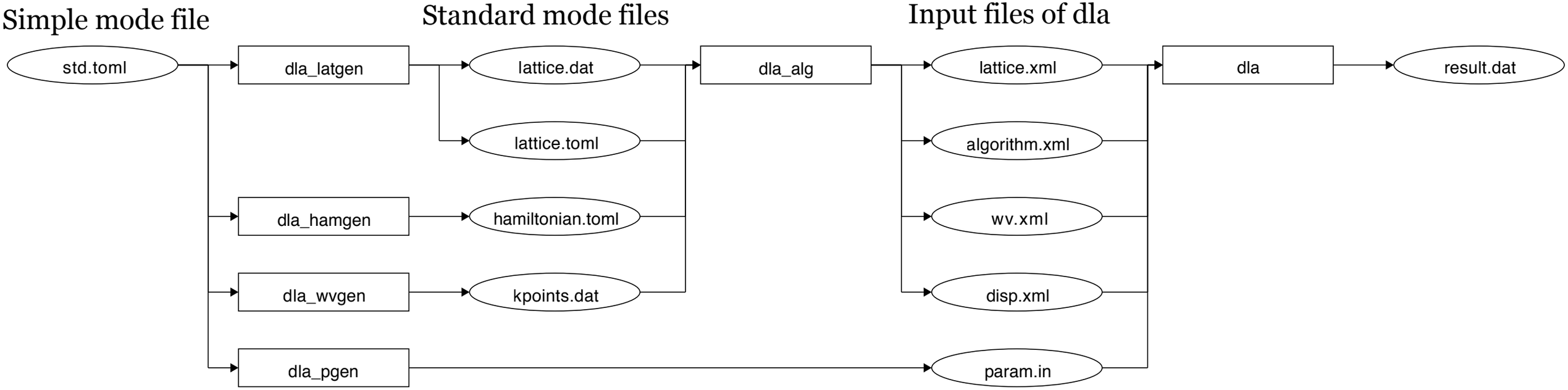
The standard mode of DSQSS/DLA



ISSP, UTokyo Yuichi Motoyama
2022-12-14 @ ISSP

The standard mode of DSQSS/dla

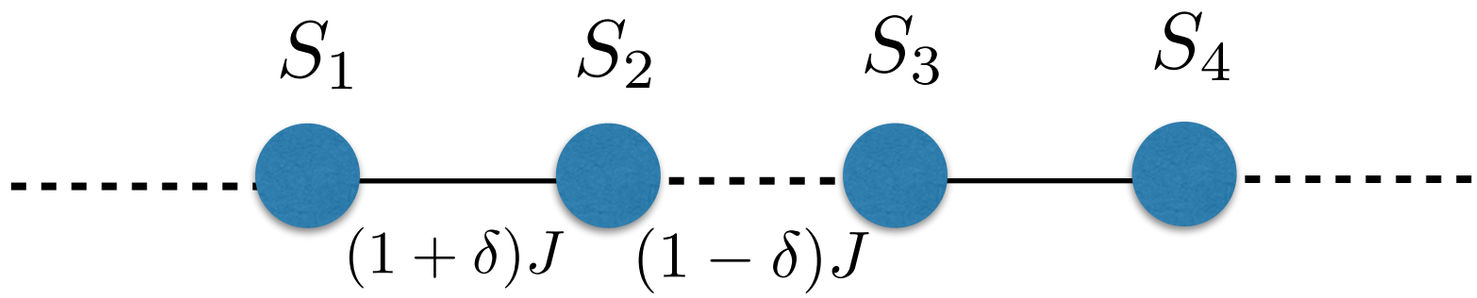
- Users can define own model and lattice (or graph) easily by the standard mode
- Standard mode files:
  - `hamiltonian.toml`
    - defines local hamiltonian (site hamiltonian, bond hamiltonian, ...)
  - `lattice.toml`
    - defines unitcell and lattice vector
  - `lattice.dat`
    - defines all the sites and all the bonds
    - can define a graph without translational symmetry
  - `kpoints.dat`
    - defines wave vectors

Use one of them

- Let's learn the standard mode with examples!

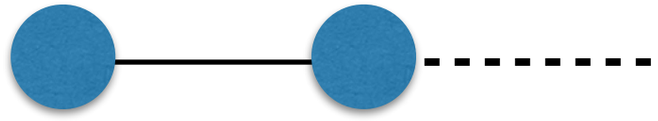
Example 1: Bond-alternating AFH chain

$$\mathcal{H} = -J \sum_{i=1}^{L} \left[1 + (-1)^{i-1}\delta\right] \vec{S}_i \cdot \vec{S}_{i+1} \qquad S = 1/2 \qquad J < 0$$



- If δ=0, it returns to a uniform AFH chain, and so the model is gapless
- If δ≠0, each spin pair on a strong bond forms a spin singlet, and so the model is gapful
- Users should define lattice.toml by hand
  - Hamiltonian, etc. are generated from the simple input
- Example files are available from THIS LINK(DropBox)
  - In future, they will be available from official repository

Lattice

unitcell = 

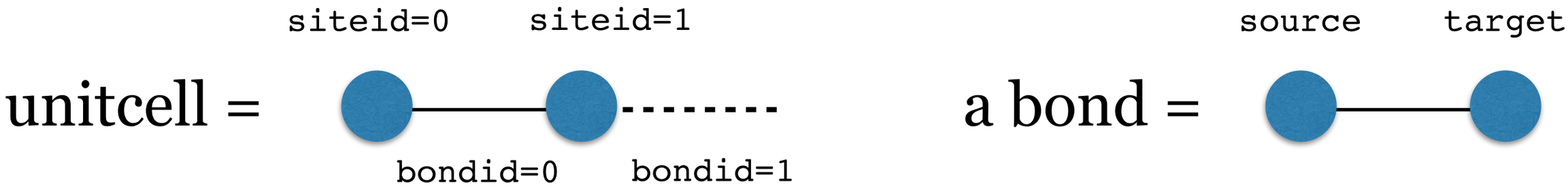
lattice vector =  with length of 2

e_1

lattice.toml

```toml
[parameter]
name = "bond-alternating chain"
dim = 1   # spatial dimension

# number of unitcells
# In this case, total number of sites is twice as L
L = 8

# boundary conditions
# true: periodic, false: open
bc = true

# lattice vectors (array of vector)
basis = [[2.0]]  # [e_1]
```

Lattice: unitcell



Site infomation

lattice.toml (cont.)

```
[unitcell]

[[unitcell.sites]]
siteid = 0
type = 0
coord = [0.0]

[[unitcell.sites]]
siteid = 1
type = 0
coord = [0.5] # fractional coord
```

"type" is used in Hamiltonian

Bond infomation

lattice.toml (cont.)

```
[[unitcell.bonds]]
bondid = 0
type = 0
[unitcell.bonds.source]
siteid = 0
[unitcell.bonds.target]
siteid = 1
offset = [0]  # the same cell

[[unitcell.bonds]]
bondid = 1
type = 1
[unitcell.bonds.source]
siteid = 1
[unitcell.bonds.target]
siteid = 0
offset = [1]  # the right neighbor cell
```

Other files

$$\mathcal{H} = -J \sum_{i=1}^{L} \left[1 + (-1)^{i-1}\delta\right] \vec{S}_i \cdot \vec{S}_{i+1}$$

std.toml

```
[parameter]
beta = 10.0
wvfile = wv.xml

[hamiltonian]

model = "spin"
M = 1                  # M=2S
Jz  = [-1.5, -0.5] # for bondtype=0 and 1
Jxy = [-1.5, -0.5] # for bondtype=0 and 1
```

$\delta = 0.5$

generate other files from std.toml

```
$ dla_hamgen std.toml
```
→ hamiltonian.toml

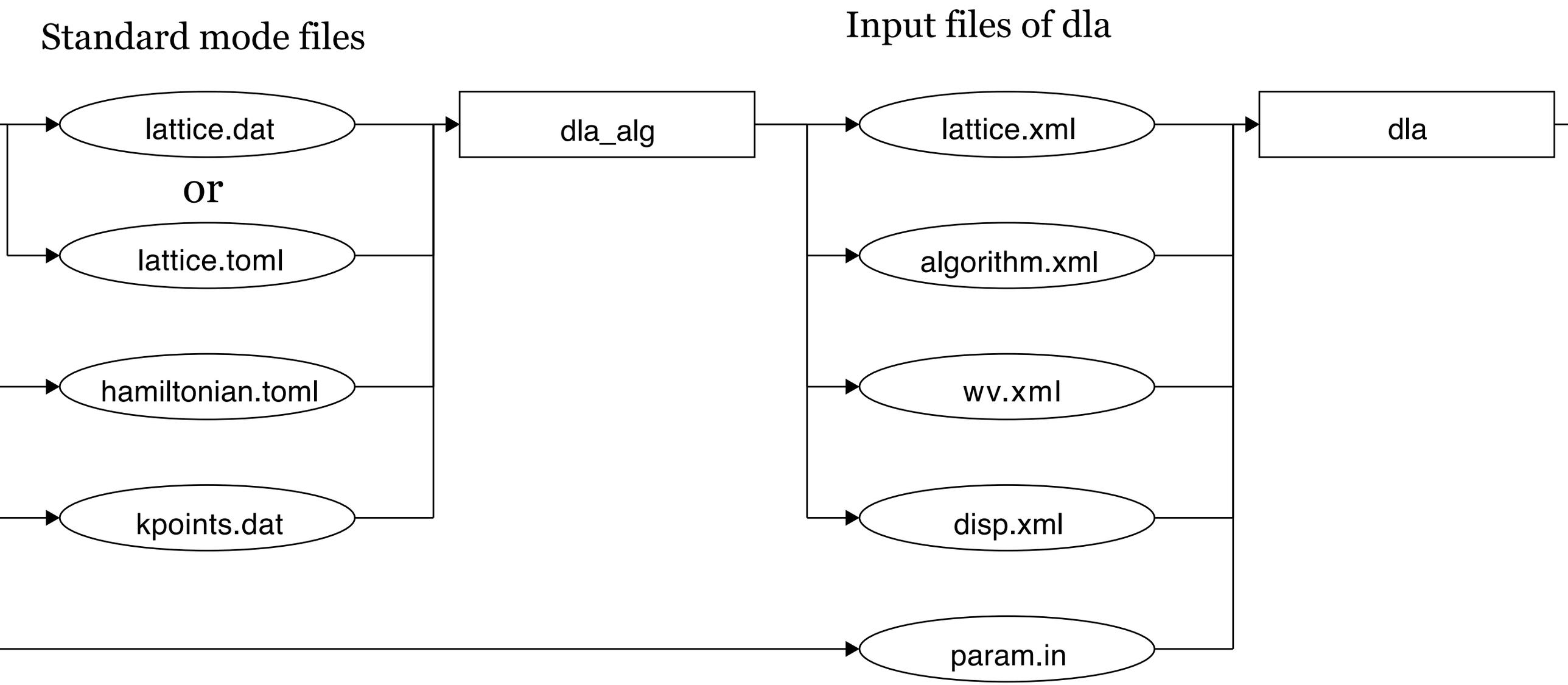
```
$ dla_pgen std.toml
```
→ param.in

```
$ dla_wvgen -s 16 std.toml
```
→ kpoints.dat

s: size of lattice

dla_alg

- dla_alg is a tool to generate the input files of dla from standard mode files

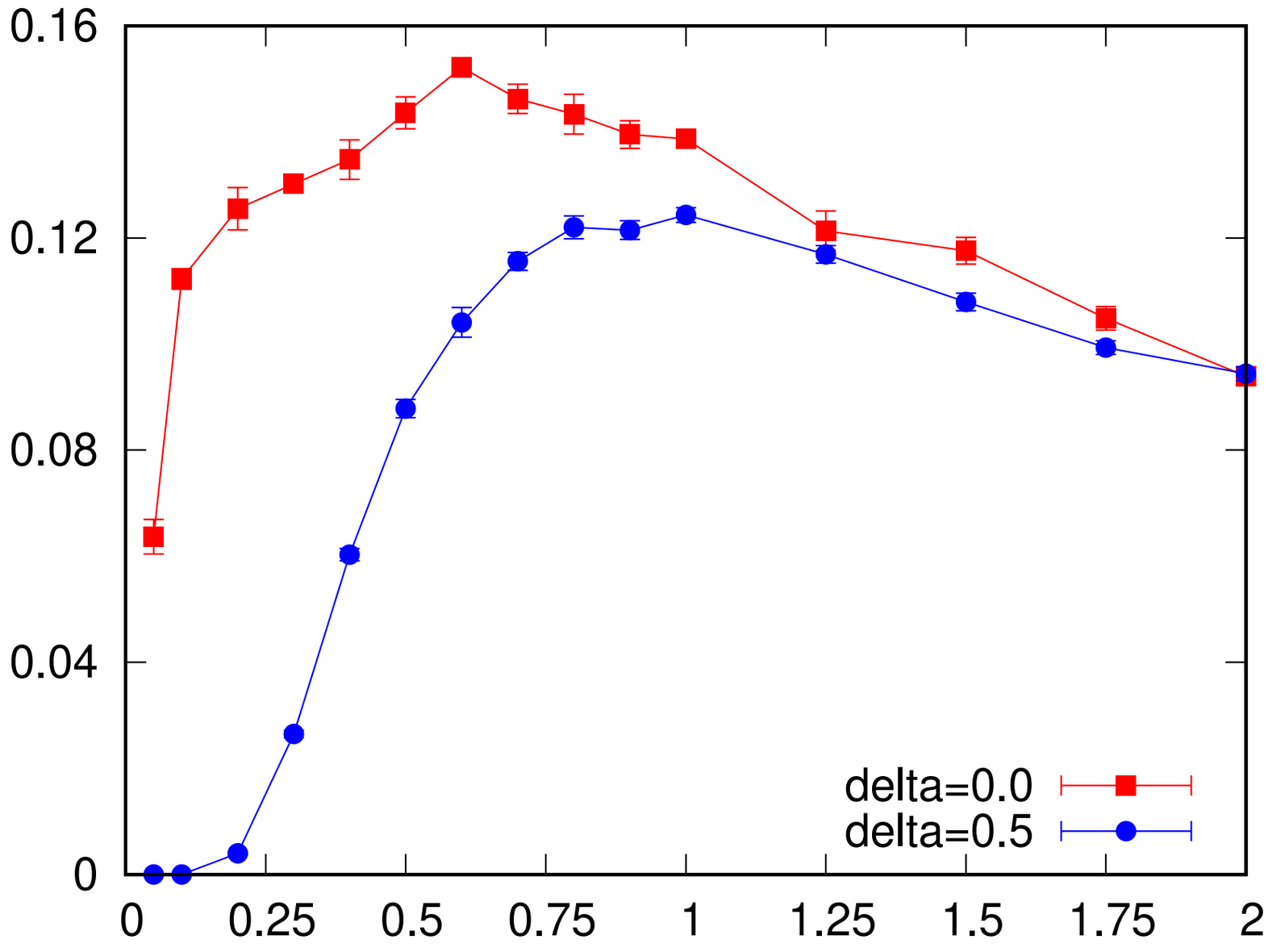



```
$ dla_alg -l lattice.toml -h hamiltonian.toml -k kpoints.dat
```

```
$ dla param.in
```

Example 1: Bond-alternating AFH chain

Staggered susceptibility vs. temperature

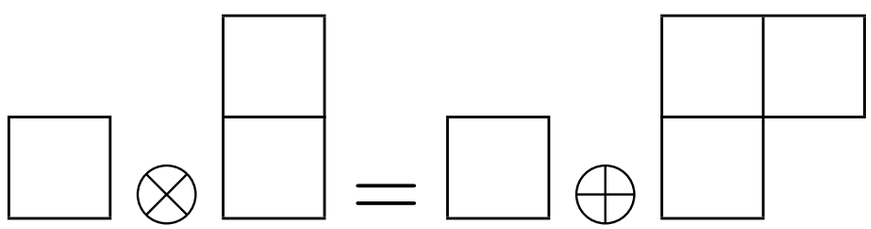


Bond alternating opens gap!

Example 2: SU(3) AFH dimer

- Files are available in `sample/dla/05_sun_dimer`
- Representation for site 0 is the fundamental (3) and that for site 1 is the conjugate of the fundamental ($\bar{3}$).
- The dimension is 3 for both representation.
- Nonzero elements of the bond Hamiltonian is $\left\langle \alpha\bar{\alpha} \right| \mathscr{H} \left| \beta\bar{\beta} \right\rangle = -1/3$
- The ground state $\langle \Psi \rangle = \frac{1}{\sqrt{3}} \sum_{\alpha} \langle \alpha\bar{\alpha} \rangle$ is unique and has the energy -1.

$$3 \otimes \bar{3} = 1 \oplus 8$$



Hamiltonian (site part)

Write nonzero elements of Hamiltonian

$$\left\langle \alpha\bar{\alpha} \middle| \mathcal{H} \middle| \beta\bar{\beta} \right\rangle = -1/3$$

hamiltonian.toml

```
name = "SU(3) AFH"
[[sites]]
type = 0  # corresponding to sitetype
N = 3     # local degree of freedom

# Elements of "local" order parameter
# (e.g. Sz)
values = [
   0.666666666666,
  -0.333333333333,
  -0.333333333333,
]

# one-site Hamiltonian (e.g. hSz)
elements = []

# wormhead source term ("Sx")
[[sites.sources]]
istate = [ 0,] # initial state
fstate = [ 1,] # final state
value = 0.5    # matrix element

[[sites.sources]]
istate = [ 1,]
fstate = [ 0,]
value = 0.5
```

```
[[sites.sources]]
istate = [ 1,]
fstate = [ 2,]
value = 0.5

[[sites.sources]]
istate = [ 2,]
fstate = [ 1,]
value = 0.5

[[sites.sources]]
istate = [ 0,]
fstate = [ 2,]
value = 0.5

[[sites.sources]]
istate = [ 2,]
fstate = [ 0,]
value = 0.5
```

Hamiltonian (interaction part)

Write nonzero elements of Hamiltonian

$$\left\langle \alpha\bar{\alpha} \middle| \mathcal{H} \middle| \beta\bar{\beta} \right\rangle = -1/3$$

hamiltonian.toml

```
[[interactions]]
type = 0       # corresponding to bondtype
nbody = 2      # number of sites
N = [ 3, 3,]  # local DoF of each site
[[interactions.elements]]
istate = [ 0, 0,]        # initial state
fstate = [ 0, 0,]        # finial state
value = -0.333333333333 # matrix element

[[interactions.elements]]
istate = [ 0, 0,]
fstate = [ 1, 1,]
value = -0.333333333333

[[interactions.elements]]
istate = [ 0, 0,]
fstate = [ 2, 2,]
value = -0.333333333333

[[interactions.elements]]
istate = [ 1, 1,]
fstate = [ 0, 0,]
value = -0.333333333333

... continue ...
```

$$\left\langle 1\bar{1} \middle| \mathcal{H} \middle| 0\bar{0} \right\rangle = -1/3$$

Other files

std.toml

```
[lattice]
lattice = "hypercubic" # hypercubic, periodic
dim = 1                # dimension
L = 2                  # number of sites along each direction
bc = false             # open boundary

[parameter]
beta = 100             # inverse temperature
nset = 5               # set of Monte Carlo sweeps
npre = 10              # MCSteps to estimate hyperparameter
ntherm = 10            # MCSweeps for thermalization
nmcs = 100             # MCSweeps for measurement
seed = 31415           # seed of RNG
```

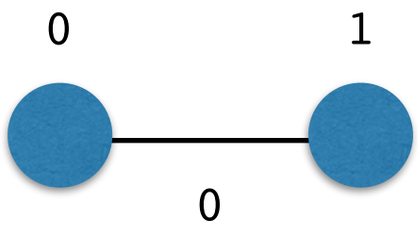
generate other files from std.toml

```
$ dla_latgen std.toml
```
→ lattice.dat

```
$ dla_pgen std.toml
```
→ param.in

lattice.dat

Information of all the sites and interactions (bonds)



lattice.dat

```
name
hypercubic

lattice
1 # dim
2 # size
0 # 0:open boundary, 1:periodic boundary
0 1.0 # latvec_0

directions
1 # ndirections
# id, coords...
0 1.0

sites
2 # nsites
# id, type, coord...
0 0 0.0
1 0 1.0

interactions
1 # nints
# id, type, nbody, sites..., edge_flag, direction
0 0 2 0 1 0 0
```

Example 2: SU(3) AFH dimer

```
$ dla_alg -l lattice.dat -h hamiltonian.toml
```

```
$ dla param.in
```

```
$ grep ene sample.log
R ene = -5.08066667e-01 5.70157873e-03
```

$$E = -1.01(1)$$

$$E_{\mathrm{exact}} = -1$$

Summary

- In this tutorial, we have learned how to define user's own model and/or lattice by using two simple examples:
  - Ex.1: How to define lattice (bond-alternating AFH chain)
  - Ex.2: How to define model (SU(3) AFH dimer)
- DSQSS/DLA offers some tools for generating input files of standard mode from the simple mode input file (std.toml)
- To modify generated files is easy way to define model and/or lattice
- To check generated files representing well-known model/lattice is good for education
  - Appendix: AFH model on square lattice

appendix

lattice.toml

- `lattice.toml` can be generated from the simple mode input TOML file by using `dla_latgen`

std.toml

```
[lattice]
lattice = ‘hypercubic'
dim = 2
L = [4,4]
```

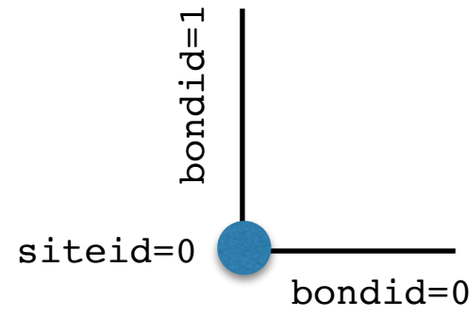
```
$ dla_latgen -t lattice.toml std.toml
```

lattice.toml

```
[parameter]
name = "hypercubic"
dim = 2         # spatial
dimension
L = [ 4, 4,]  # number of
unitcells

# boundary conditions
bc = [ true, true,]

## omit the rest...
```

lattice.toml

unitcell =



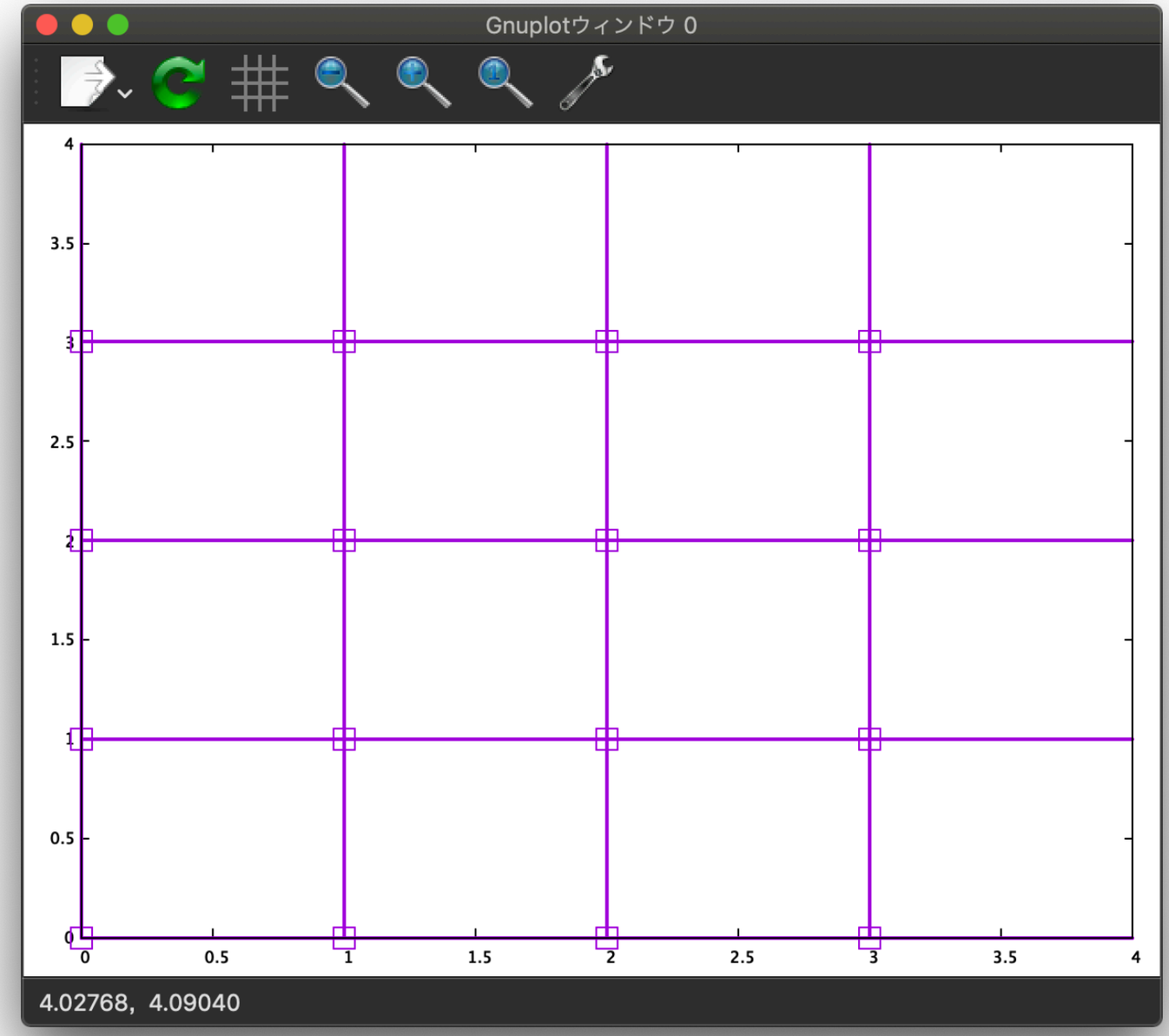

```toml
[parameter]
name = "hypercubic"
dim = 2         # spatial dimension
L = [ 4, 4,]  # number of unitcells

# boundary conditions
bc = [ true, true,]

# lattice vectors
basis = [[1. 0.]    # e_1
         [0. 1.]]   # e_2

[unitcell]
# sites in the unitcell
[[unitcell.sites]]
siteid = 0
type = 0    # sitetype

# fractional coordinate in the cell
coord = [ 0.0, 0.0,]

## continued to the right
```

```toml
## continued from the left

# horizontal bond in the cell
[[unitcell.bonds]]
bondid = 0
type = 0

# an end site of the bond
[unitcell.bonds.source]
siteid = 0

# the opposite end site
[unitcell.bonds.target]
siteid = 0
# relative coord of the target cell
offset = [ 1, 0,]

# vertical bond
[[unitcell.bonds]]
bondid = 1
type = 0

[unitcell.bonds.source]
siteid = 0
[unitcell.bonds.target]
siteid = 0
offset = [ 0, 1,]
```

lattice preview by gnuplot

- `dla_latgen` can generate a gnuplot file for viewing the 2D lattice

```
$ dla_latgen -g lattice.plt lattice.toml
$ gnuplot lattice.plt
```



lattice.dat

- `lattice.dat` can be generated from the simple mode input TOML file by using `dla_latgen`

std.toml

```
[lattice]
lattice = ‘hypercubic'
dim = 2
L = [4,4]
```

```
$ dla_latgen -o lattice.dat std.toml
```

lattice.dat

```
name
hypercubic

lattice
2   # dim
4 4 # size (Lx, Ly)

## omit the rest...
```

lattice.dat

4x4 square lattice

```
name     # tag
hypercubic

lattice  # tag
2    # dim
4 4 # size (Lx, Ly)

# boundary conditions
# 0: open, 1:periodic
1 1 # x, y

# lattice vectors
0 1.0 0.0 # latvec_0
1 0.0 1.0 # latvec_1

# direction of bonds
directions  # tag
2 # ndirections

# id, coords...
0 1.0 0.0
1 0.0 1.0

# continued to the right
```

```
# continued from the left

# each sites
sites  # tag
16 # nsites
# id, type, coord...
0 0 0.0 0.0
1 0 1.0 0.0
2 0 2.0 0.0
  … skip …
15 0 3.0 3.0

# each N-body interaction
interactions
32 # nints
# id, type, nbody, sites..., edge_flag, direction
   0      0       2        0 1            0            0
   1      0       2        0 4            0            1
   2      0       2        1 2            0            0
  … skip …
31 0 2 15 3 1 1
```

`edge_flag`: whether to cross the boundary (1) or not (0)

hamiltonian.toml

- `hamiltonian.toml` can be generated from the simple mode input TOML file by using `dla_hamgen`
  - If you want to define your model, it is recommended that you first generate some hamiltonian.toml files of some different simple models to learn.

std.toml

```
[hamiltonian]
model = 'spin'
M = 1
Jz = -1
Jxy= -1
h = 0.5
```

→ `$ dla_hamgen std.toml` →

hamiltonian.toml

```
name = "S=1/2 XXZ model"
[[sites]]
type = 0
N = 2
values = [ -0.5, 0.5,]

## omit the rest...
```

hamiltonian.toml

```
name = "S=1/2 XXZ model" # Name of model

# One Site
[[sites]]
type = 0  # Sitetype (sublattice)
N = 2     # The number of local degree of freedom (e.g., N=2 if S=1/2)
values = [ -0.5, 0.5,]  # Value of local basis (e.g., Sz)
                        # <0|Sz_0|0> = -0.5
                        # <1|Sz_0|1> =  0.5

# Site term Hsite_0 (e.g., Zeeman term)
[[sites.elements]]
istate = [ 0,]      # initial state
fstate = [ 0,]      # final state
value = 0.25        # <0|Hsite_0|0> = 0.25

[[sites.elements]]
istate = [ 1,]
fstate = [ 1,]
value = -0.25       # <1|Hsite_0|1> = -0.25

# Worm source term
[[sites.sources]]
istate = [ 0,]
fstate = [ 1,]
value = 0.5         # <1|G_0|0> = 0.5

[[sites.sources]]
istate = [ 1,]
fstate = [ 0,]
value = 0.5         # <0|G_0|1> = 0.5

## continued to the next page
```

hamiltonian.toml

```
## continued from the previous page

# manybody interactions Hint_0
[[interactions]]
type = 0        # interaction type (id)
nbody = 2       # the number of sites
N = [ 2, 2,]    # the number of local d.o.f. of sites

# matrix elements
[[interactions.elements]]
istate = [ 0, 0,]  # initial state
fstate = [ 0, 0,]  # final state
value = 0.25       # <0,0|Hint_0|0,0> = 0.25

[[interactions.elements]]
istate = [ 0, 1,]
fstate = [ 0, 1,]
value = -0.25      # <0,1|Hint_0|0,1> = -0.25

[[interactions.elements]]
istate = [ 0, 1,]
fstate = [ 1, 0,]
value = 0.5        # <1,0|Hint_0|0,1> = 0.5

[[interactions.elements]]
istate = [ 1, 0,]
fstate = [ 1, 0,]
value = -0.25      # <1,0|Hint_0|1,0> = -0.25

## omit the rest
```

$$H_0^{\mathrm{int}} = \begin{pmatrix} 1/4 & 0 & 0 & 0 \\ 0 & -1/4 & 1/2 & 0 \\ 0 & 1/2 & -1/4 & 0 \\ 0 & 0 & 0 & 1/4 \end{pmatrix}$$

kpoints.dat

coordinate of site $\vec{r} = \sum_{d=1}^{D} r_d \vec{e}_d$

lattice basic vector e_d defined
in `lattice.toml` or `lattice.dat`

wavevector $\vec{k} = \sum_{d=1}^{D} k_d \vec{g}_d$

reciprocal basic vector g_d defined such that

$$\vec{g}_d \cdot \vec{e}_{d'} = \frac{2\pi}{L_d} \delta_{d,d'}$$

`std.toml`

```
[kpoints]
kstep = [4,4]
```

→ `$ dla_wvgen -s "8 8" std.toml`

↓ `kpoints.dat`

```
dim
2

kpoints
# id, k_1, k_2, …
0  0  0
1  4  0
2  0  4
3  4  4
```

$$\vec{k}_0 = \vec{0}$$
$$\vec{k}_1 = \pi \vec{g}_1$$
$$\vec{k}_2 = \pi \vec{g}_2$$
$$\vec{k}_3 = \pi \vec{g}_1 + \pi \vec{g}_2$$