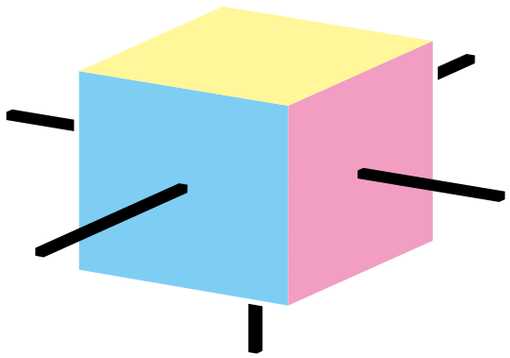


TeNeS

Tensor Network Solver for Quantum Lattice Systems

on MateriApps LIVE!

Yuichi Motoyama (ISSP)

2023-11-20
for TeNeS ver 2.0.0

TeNeS を最新版にする

- TeNeS を使う前にMALIVE! に入っているTeNeS を更新しておきます
  - パスワードを要求されたら `live`

```
$ sudo apt update          # パッケージ一覧を更新する
$ sudo apt install tenes   # TeNeS を更新する
```

（先頭の $ は入力待ち状態を示す記号なので入力しないでください）

（# 以降はコメントを示しています、入力しなくて大丈夫です）

```
$ tenes --version # バージョン確認
TeNeS v2.0.0      # OK!
```

サンプルをコピー

- TeNeS のサンプル集を手元にコピーしてきます

```
$ cd                                # HOME directory へ移動
$ cp -r /usr/share/tenes . # TeNeS のサンプルをコピー
$ cd tenes/sample
```

（ファイル名などは途中まで入力した後に Tab キーを押すと適宜補完可能です）

サンプルの説明はチュートリアルも参照してください

https://issp-center-dev.github.io/TeNeS/manual/v2.0.0/ja/html/tutorial/index.html

sample-1 横磁場イジング模型

- https://issp-center-dev.github.io/TeNeS/manual/v2.0.0/ja/html/tutorial/01_transverse_field_ising.html
- 最初の例として正方格子横磁場イジング模型を扱います
  - S=1/2 の演算子で書かれていることと符号に注意
  - hx ~ 1.5 あたりに量子相転移があります

$$\mathcal{H} = J_z \sum_{\langle ij \rangle} S_i^z S_j^z - h_x \sum_i S_i^x$$

```
$ cd 01_transverse_field_ising
$ cat simple.toml  # 入力ファイルの確認
 ... Skipped ...

[lattice]
type = "square lattice"
L = 2
W = 2
virtual_dim = 2
initial = "ferro"

[model]
type = "spin"
Jz = -1.0
Jx = 0.0
Jy = 0.0
hx = 0.0
```

正方格子

2x2 ユニットセル

ボンド次元 = 2

スピン模型

強磁性イジング Jz=-1, Jx=Jy=0

横磁場 hx = 0

sample-1 横磁場イジング模型

・ 入力ファイル simple.toml をinput.toml へと変換して tenes を実行

```
$ tenes_simple simple.toml  # convert to std.toml
$ tenes_std std.toml        # convert to input.toml
$ tenes input.toml          # main calculation

  # ... Skipped ...

Onesite observables per site:
  Sz                  = 0.5 0                          <Sz> の実部・虚部
  Sx                  = -1.28526262482e-13 0           <Sx> の実部・虚部
Twosite observables per site:
  bond hamiltonian = -0.5 0  エネルギー
  SzSz                = 0.5 0
  SxSx                = -1.7374919982e-18 0            最近接相関
  SySy                = 1.73749202733e-18 0

  # ... Skipped ...
```

sample-1 横磁場イジング模型

- 横磁場を入れてみましょう
  - 具体的には、 hx = 3 に変えてみます

```
$ mv output output_hx0  # 計算結果を退避しておく
$ mousepad simple.toml  # 入力ファイルを編集
```

```
[model]                                  [model]
type = "spin"                            type = "spin"
Jz = -1.0                                Jz = -1.0
Jx = 0.0                                 Jx = 0.0
Jy = 0.0                                 Jy = 0.0
hx = 0.0                                 hx = 3.0
```

(mousepad はメモ帳のような単純なエディタです)
(MALIVE! には他にも vim, emacs, nano が入っているのでお好きにどうぞ)
(MALIVE! on docker では vim と emacs のみ？)

sample-1 横磁場イジング模型

- 編集した simple.toml をinput.toml へと変換して tenes を実行

```
$ tenes_simple simple.toml  # convert to std.toml
$ tenes_std std.toml        # convert to input.toml
$ tenes input.toml          # main calculation
```

```
  # ... Skipped ...

Onesite observables per site:
  Sz                  = 8.05352165651e-09 0   <Sz> の実部・虚部
  Sx                  = 0.492509565167 0      <Sx> の実部・虚部
Twosite observables per site:
  bond hamiltonian = -1.52140952585 0   エネルギー
  SzSz                = 0.0438808303474 0
  SxSx                = 0.488706591605 0   最近接相関
  SySy                = -0.0407090179083 0

  # ... Skipped ...
```

sample-1 横磁場イジング模型

- output ディレクトリに更に詳細な結果が出力されます
- 例えば S_i^z などの1サイト演算子のサイト毎の期待値は onesite_obs.dat に出力

```
$ head -n 15 output/onesite_obs.dat
# The meaning of each column is the following:
# $1: op_group
# $2: site_index
# $3: real
# $4: imag
# The names of op_group are the following:
# 0: Sz
# 1: Sx
# -1: norm

0 0 8.0535578151827166 9e-09 0.0000000000000000e+00
0 1 8.0535568759964675 6e-09 0.0000000000000000e+00
0 2 8.0535586991227092 5e-09 0.0000000000000000e+00
0 3 8.0535569864889624 9e-09 0.0000000000000000e+00
1 0 4.9250956516679217 4e-018 0.0000000000000000e+00
```

sample-1 横磁場イジング模型

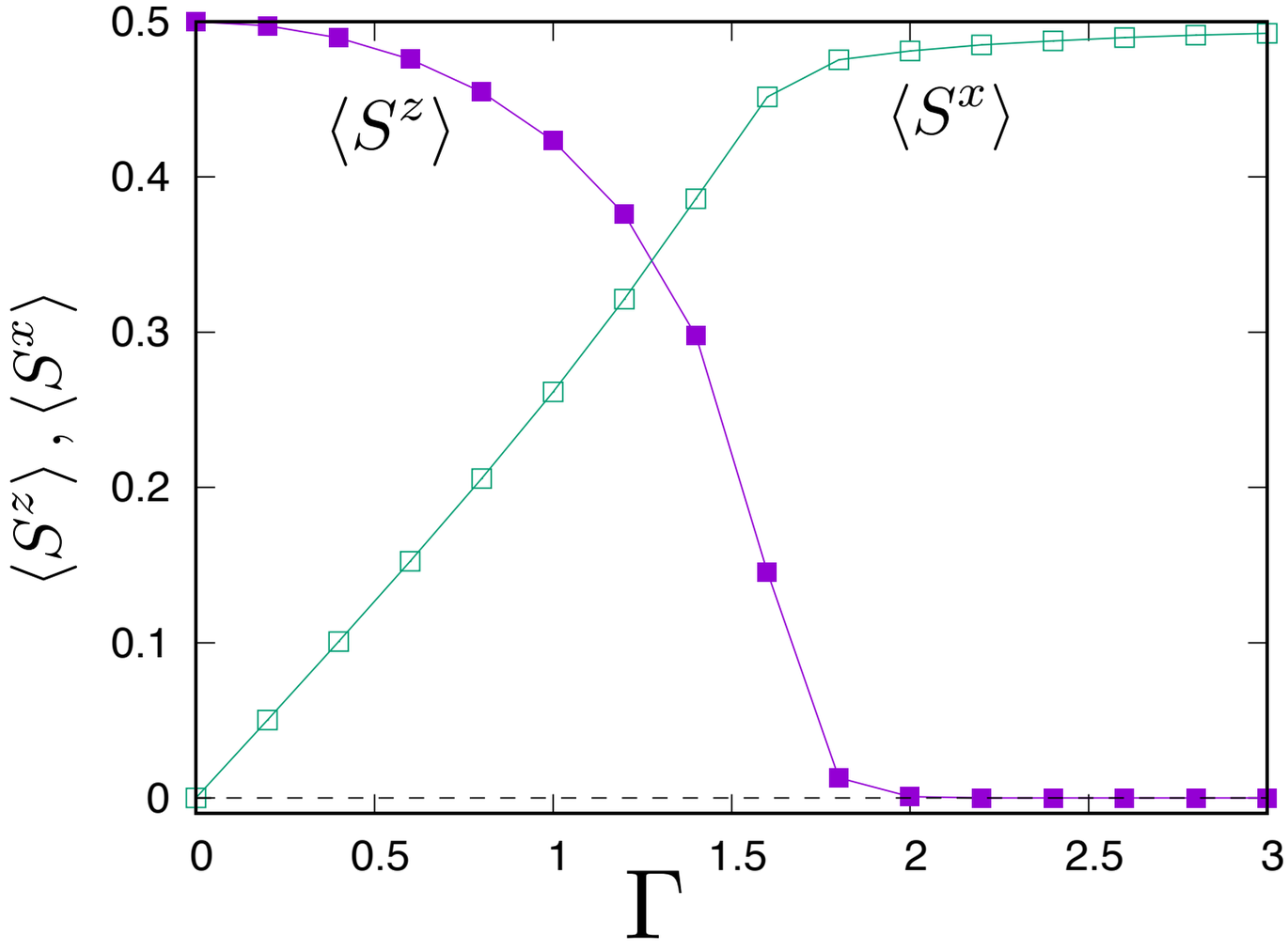
- 横磁場をすこしずつ変えながら計算していくと量子相転移を観測できます
  - 単純作業を手でやるのは不毛なのでプログラムを書くのが良いです
  - tutorial_example.py と tutorial_read.py が用意されています

```
$ mv output output_hx3          # 一応計算結果を退避しておく
$ python3 tutorial_example.py # ひたすら実行
$ python3 tutorial_read.py     # 計算結果をまとめる
0.0 -5.000000000000000e-01 5.000000000000000e-01
-1.2852626248178210 6e-13
0.2 -5.050041766921166 13e-01 4.974824892468133 20e-01
5.008365591343368 76e-02
0.4 -5.200673589117371 65e-01 4.897123560485541 76e-01
1.006771028363506 46e-01
   # ... Skipped ...

$ python3 tutorial_read.py > result.dat # ファイルへ保存
```

sample-1 横磁場イジング模型

- 出力された 4列の意味は順番に、
  - 横磁場 hx
  - サイトあたりのエネルギー
  - サイトあたりの縦磁化 Sz
  - サイトあたりの横磁化 Sx
- gnuplot でプロットしてみましょう

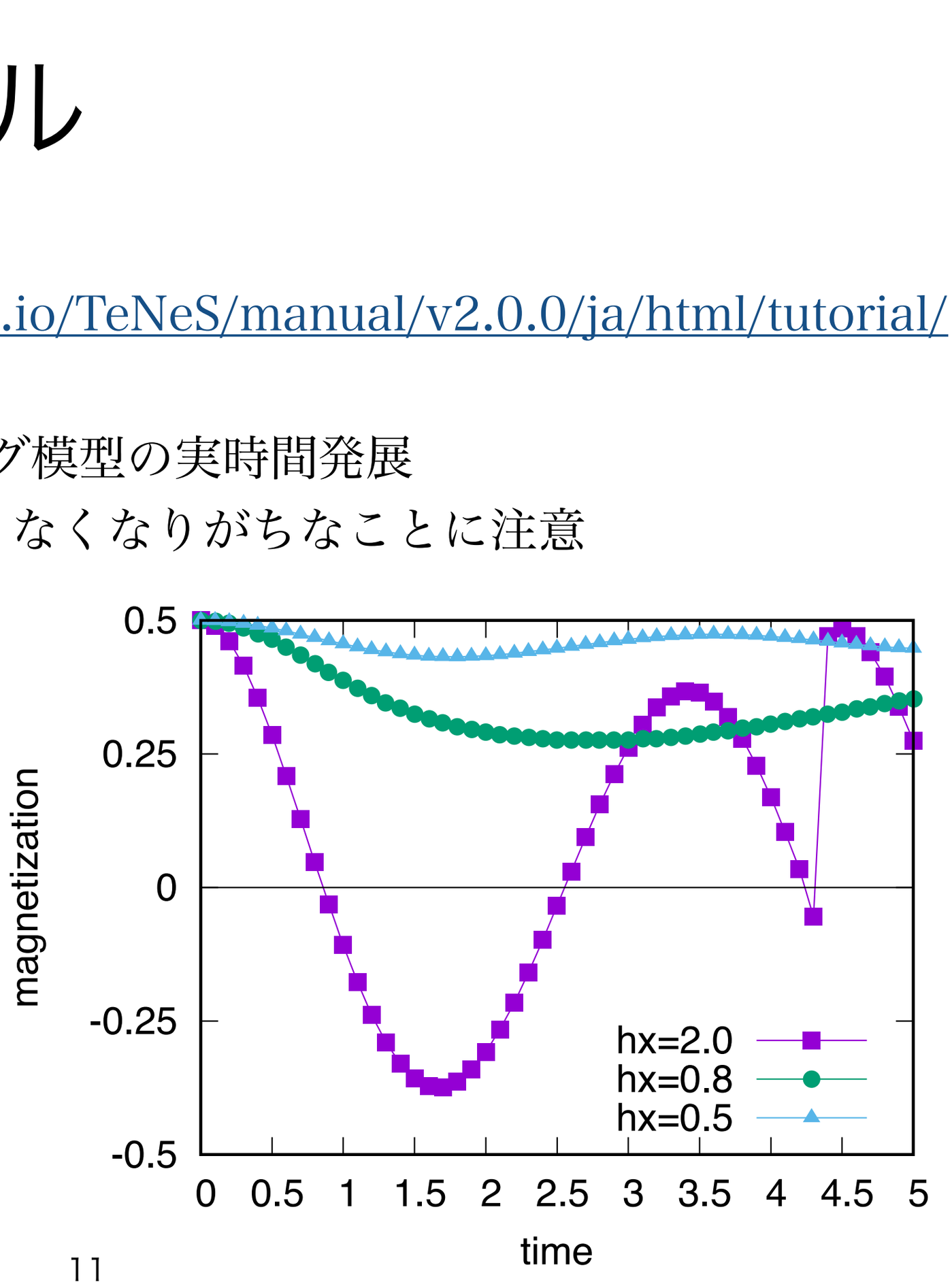



```
$ gnuplot
gnuplot> set style data lp
gnuplot> pl 'result.dat' u 1:3 t 'Sz', '' u 1:4 t 'Sx'
```

（実際の計算ではボンド次元などのハイパーパラメータを変えて結果の収束のしかたを確認します）

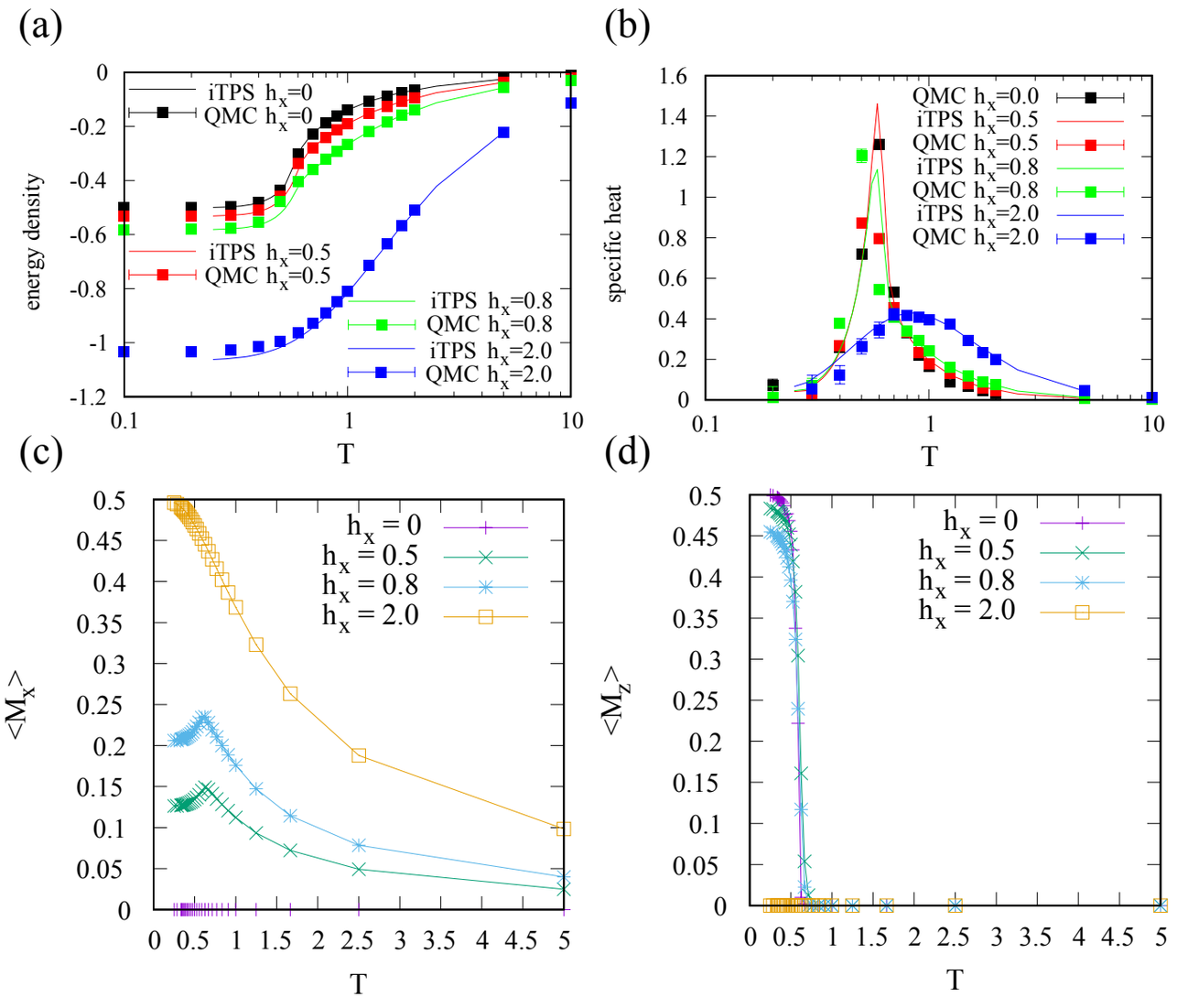
その他のサンプル

- `02_time_evolution`
  - https://issp-center-dev.github.io/TeNeS/manual/v2.0.0/ja/html/tutorial/02_time_evolution.html
  - 正方格子 S=1/2 横磁場イジング模型の実時間発展
  - 実時間発展はボンド次元が足りなくなりがちなことに注意



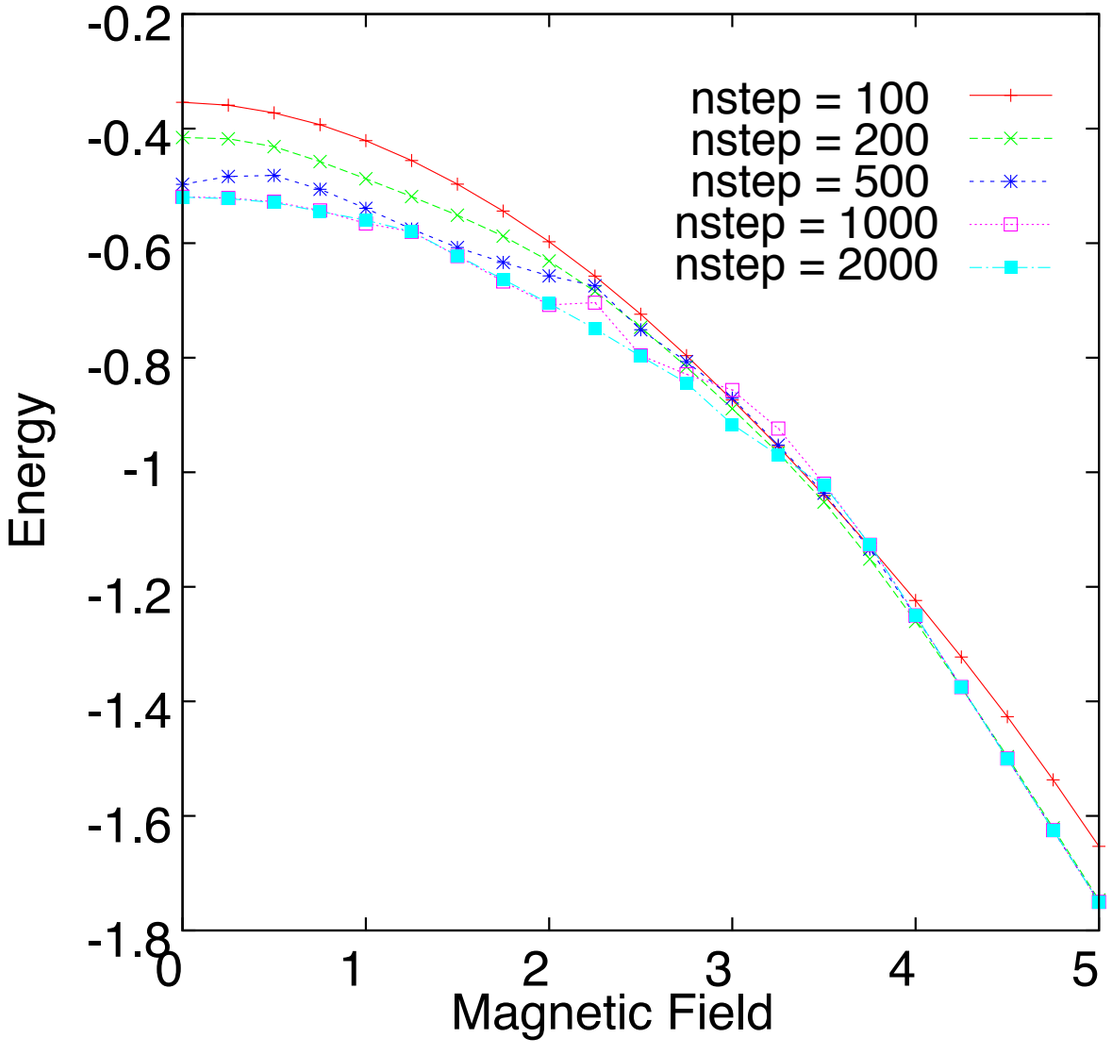
その他のサンプル

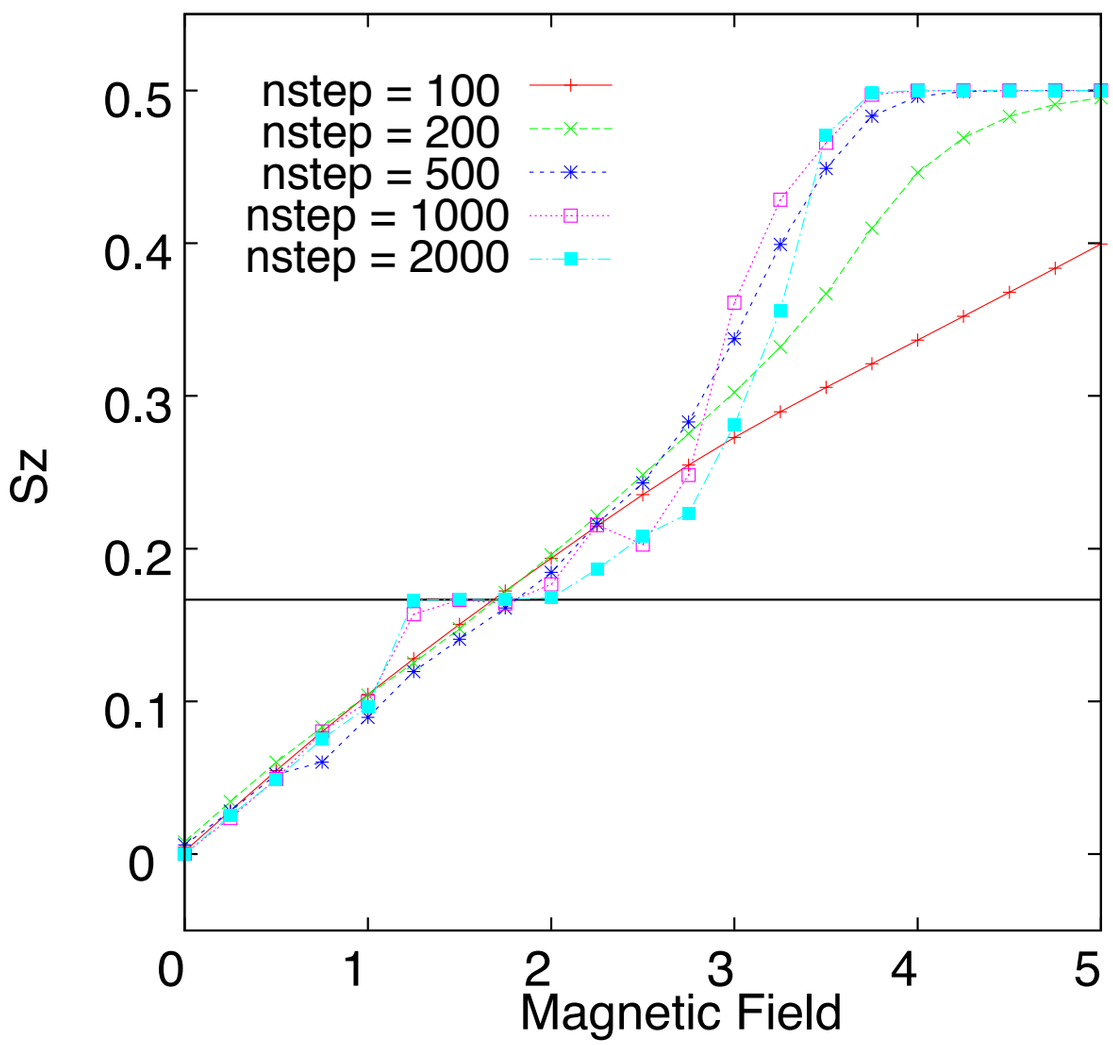
- `03_finite_temperature`
  - https://issp-center-dev.github.io/TeNeS/manual/v2.0.0/ja/html/tutorial/03_finite_temperature.html
  - 正方格子 S=1/2 横磁場イジング模型の有限温度計算
  - v2.0.0 の plot_mz.plt, plot_mx.plt が MALIVE! でうまく動かないので注意
    - 158行目の
    - `set locale "ja_JP.UTF-8"`
    - を消すとよい
- v2.0.0 の calcspec.py は横磁場のない系の比熱を計算しない
  - 16行目の for loop に "zero" を追加すれば良い
  - MALIVE! では直してある



その他のサンプル

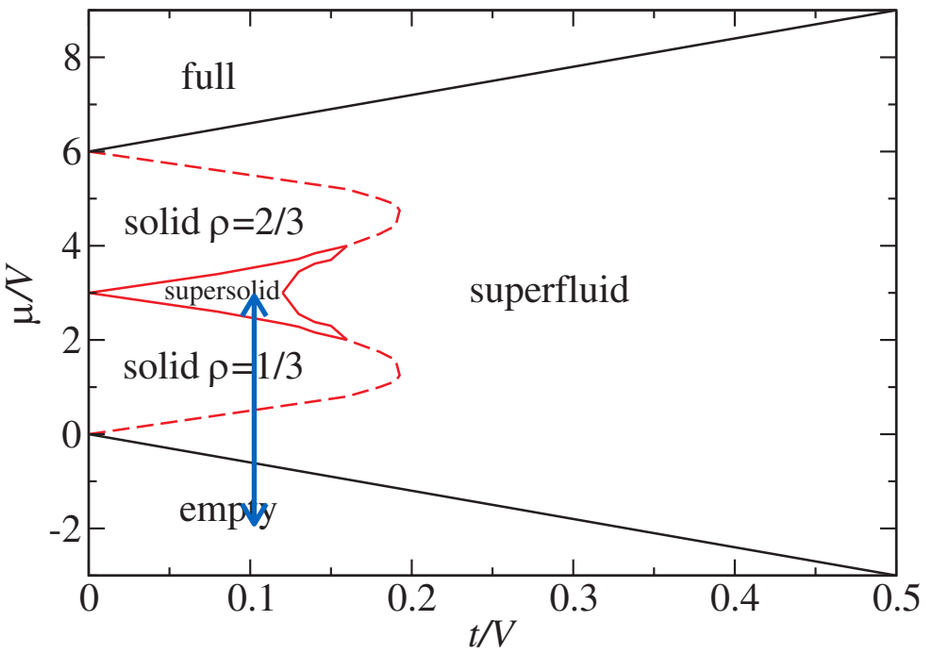
- 04_magnetization
  - https://issp-center-dev.github.io/TeNeS/manual/v2.0.0/ja/html/tutorial/04_magnetization.html
  - S=1/2 反強磁性ハイゼンベルグ模型の、正方格子・三角格子での磁化曲線



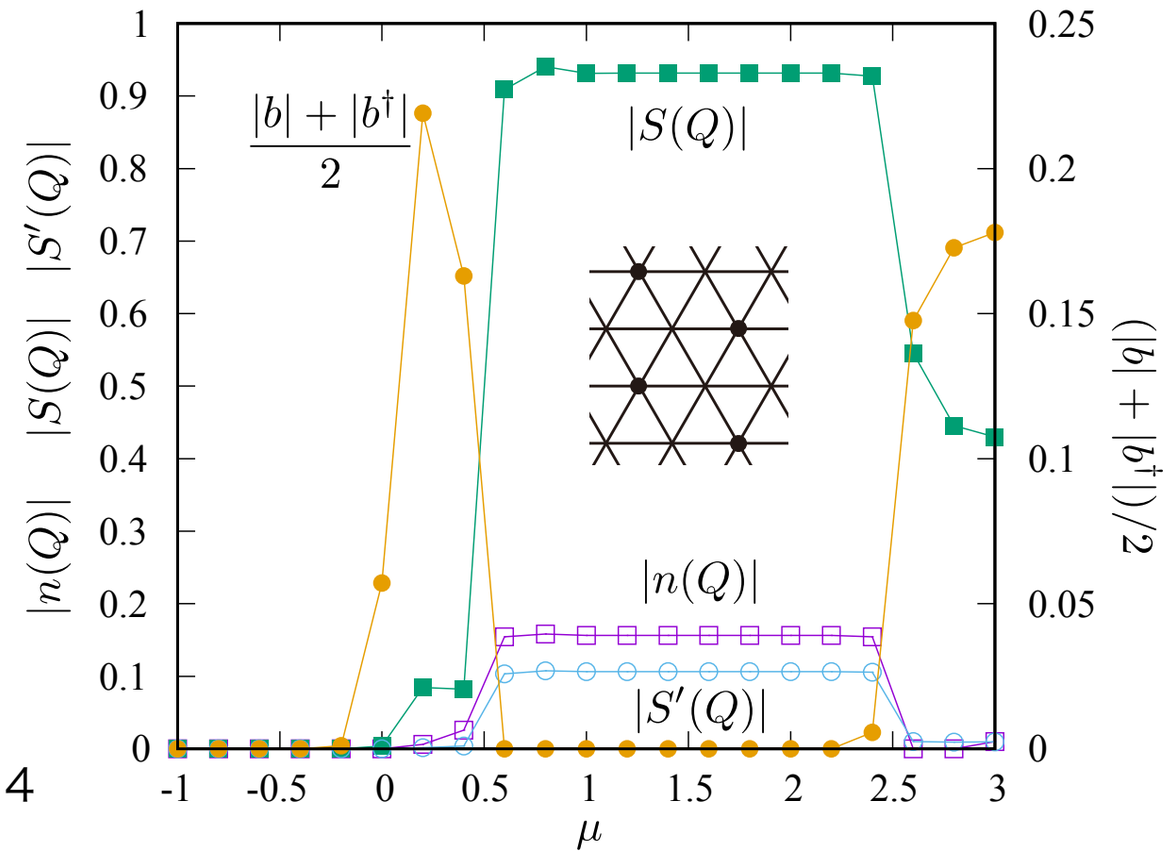


その他のサンプル

- `05_hardcore_boson_triangular`
  - https://issp-center-dev.github.io/TeNeS/manual/v2.0.0/ja/html/tutorial/05_hardcore_boson_triangular.html
  - 三角格子ハードコアボースハバード模型
  - 固体相・超流動相・超固体相の計算
    - 超流動性は非対角秩序変数 $\langle b_i \rangle$ で判断する
  - 固体相のために構造因子 $S(Q)$ を計算する例を含む（やや高度）



Wessel and Troyer, PRL 95, 127205 (2005)

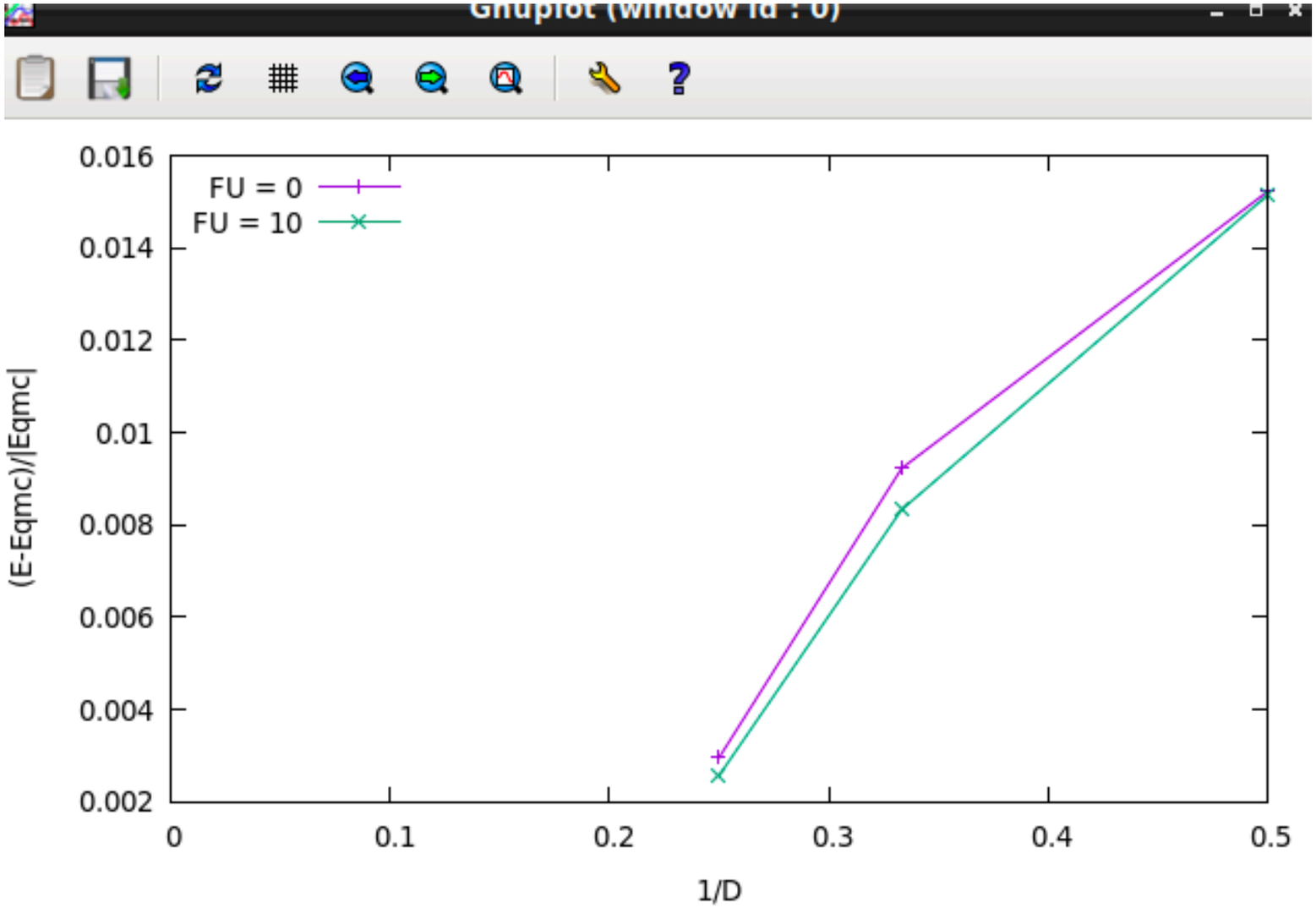


その他のサンプル

- 06_std_model
  - https://issp-center-dev.github.io/TeNeS/manual/v2.0.0/ja/html/tutorial/06_std_model.html
  - std.toml を編集して模型を定義するサンプル
  - 交代磁場下のスピン模型を計算してNeel 状態の iTNSを得る
    - tensor_save = "ディレクトリ名" で得られた状態 を保存可能
    - tensor_load = "ディレクトリ名" で初期状態として利用可能

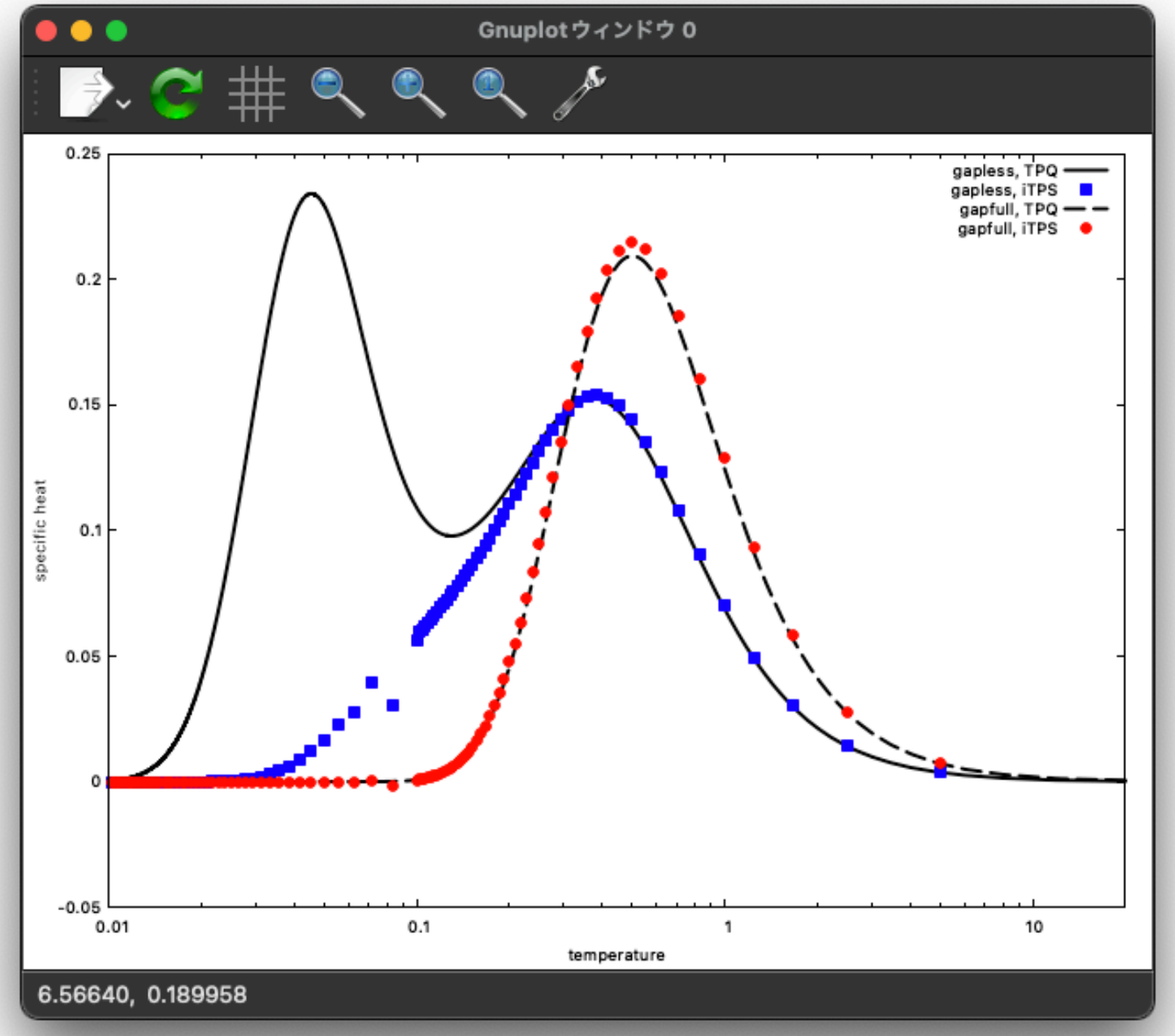
その他のサンプル

- AFH_square
  - 正方格子 S=1/2 反強磁性ハイゼンベルグ模型
  - ボンド次元や full update のステップ数を変えたときに得られる基底状態エネルギーの振る舞い



その他のサンプル

- FT_Kitaev
  - 蜂の巣格子 S=1/2 Kitaev 模型の有限温度計算
  - gapless (blue symbols)
    - Jx = Jy = Jz = 1
  - gapfull (red symbols)
    - Jx = 0.3
    - Jy = 0.3
    - Jz = 2.4
- 実線と破線はTPQ計算



MALIVE! 以外での注意

- sample 以下にあるPython スクリプトについて
  - `tenes` や `tenes_simple` などを修飾なしに実行します
    - これらの実行ファイルへの PATH が通っている必要があります
  - `tenes` を MPI 並列実行したい場合は、スクリプト中の最初の方にある MPI_cmd を修正してください
    - 例： `MPI_cmd = "mpiexec -np 2"`
  - OpenMP の並列数は環境変数 `OMP_NUM_THREADS` を設定してください
    - ボンド次元が小さい場合は並列数を減らしたほうが速いです
- `tenes_simple` / `tenes_std` ではいくつかのPython パッケージが必要です
  - NumPy, SciPy, toml
  - 実行時に見つからなかった場合は適宜入れましょう
    - `pip3 install --user numpy scipy toml`
    - `--user` を使うと自分のホームディレクトリ以下にインストールできます