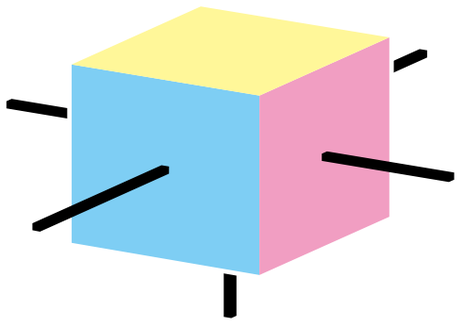
二次元量子格子系
テンソルネットワーク
ソルバーパッケージ



TeNeS

Tensor Network Solver for Quantum Lattice Systems

https://www.pasums.issp.u-tokyo.ac.jp/tenes

東大物性研　本山裕一

2020-07-29　物性アプリオープンフォーラム

目次

二次元量子格子模型（強相関）

- 物性物理として
  - 量子ゆらぎによる豊富な秩序（無秩序？）状態
    - 超流動、超伝導、スピンギャップ相、SPT 相、量子スピン液体、などなど
  - 豊富な格子 (see "Quantum Magnetism" chap.2)
  - 様々な現実物質が二次元量子格子模型で表現可能
    - 銅酸化物、鉄系超伝導体、有機塩、などなど
- 計算物理のフィールドとして
  - 平均場や単純な一体近似では解けないような相互作用領域
    - 数値計算がさらに重要になる
  - 一次元ほど特殊ではない
    - （特に強相関では）解析的な手法はほとんどない
  - 三次元よりは計算が（計算コスト的な意味で）楽

二次元量子格子模型の数値計算手法（一部）

- 厳密対角化・数値的対角化（ED）
  - 数値的に厳密な結果が得られる
  - システムサイズに強烈な制限がかかる（指数関数的なコスト増）（数十サイト程度）
- 経路積分モンテカルロ法(QMC, PIQMC)
  - 統計誤差の範囲内で厳密な結果が得られる
  - サイズや逆温度に対してもステップあたりの計算量が爆発しない（数万サイトの任意温度計算）
  - 負符号問題が発生すると統計誤差が指数関数的に増大する（要求精度に必要なステップ数が爆発する）
- 変分モンテカルロ法(VMC)
  - 変分原理に基づき、基底状態を高精度に探索する
  - 負符号問題が発生しない
  - そこそこ大規模な計算が可能（数百サイト）
- テンソルネットワーク（もっというとiTPS(iPEPS)）
  - 波動関数（の係数）を複数のテンソルの組み合わせで表現
  - 並進対称性を利用することで無限系の計算も可能
  - サイズの代わりに、テンソルの大きさ（要素数）に関する依存性が発生する

今日の話

二次元量子格子模型における
テンソルネットワーク法のアルゴリズム
（特に、TeNeS で用いられているもの）

TeNeS のマニュアルにおけるアルゴリズムの章に従います

https://issp-center-dev.github.io/TeNeS/manual/master/ja/html/algorithm/algorithms.html

(「TeNeS アルゴリズム」 でweb 検索できると思います)

もっと一般には

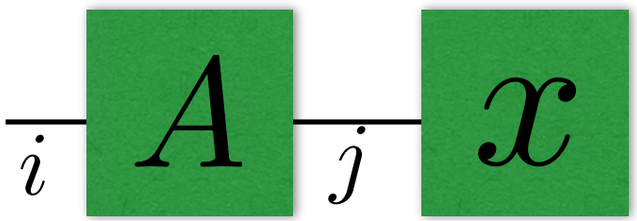
各種レビュー(例えば R. Orús Ann. of Phys. 349, 117 (2014))を参照

TNQMP2016 および CAQMP2019 でweb検索すると
過去に物性研で開かれたTN 法の国際滞在型ワークショップの
資料や講義ビデオ（公式）が見つかります
（川島研のwebpage の一番下にもリンクがあります）

テンソルとダイアグラム表記

- 非常にざっくりと、n 個の添字を持つ多次元配列のことをn階テンソルと呼ぶ
  - 添字の上付き・下付きにも特に意味を付けない
  - 0階テンソル＝スカラー
  - 1階テンソル＝ベクトル
  - 2階テンソル＝行列
- 複数のテンソルの関係を見やすくするためにダイアグラムで表す
  - 一つのテンソルは四角や丸などの図形
  - テンソルの添字（足）はそこから生えた線分
  - つながった足は縮約（添字に関する和）
    - 例：行列ベクトル積
    - 足の本数が計算量

$$\sum_j A_{ij} x_j =$$



$$O(D_i D_j)$$

テンソルネットワーク(TN)波動関数

- 波動関数の展開係数は巨大なテンソル

$$|\Psi\rangle = \sum_{\{s_i\}=1}^{d} \Psi_{s_1,s_2,\ldots,s_N} |s_1 s_2 \cdots s_N\rangle$$

N: サイト数

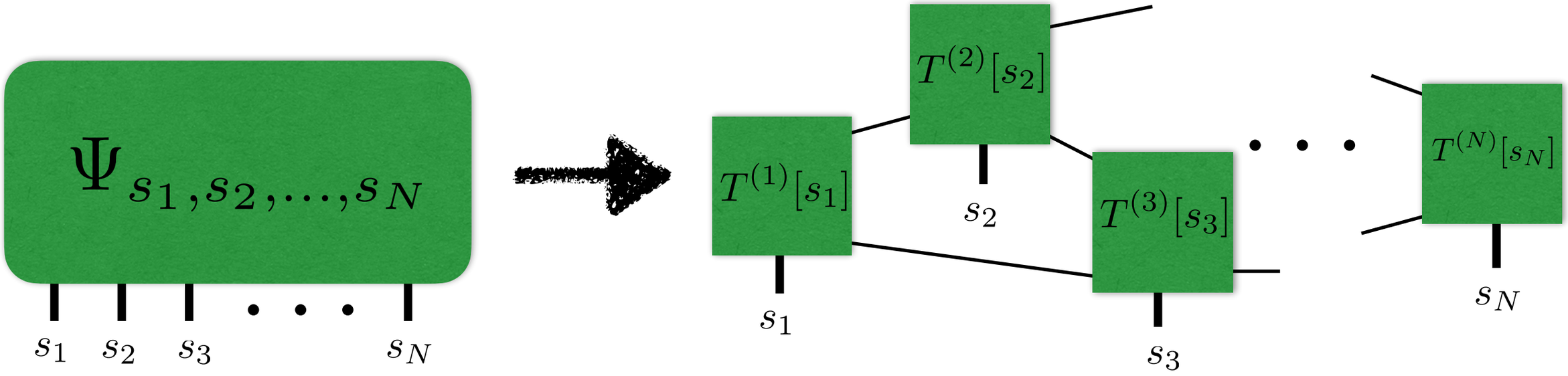
d: サイトあたりの状態数

  - テンソルの要素数は系のサイズと共に指数関数的に増える(d^N)

- 係数テンソルを小さなテンソルの積として表現する (tTr はTN の縮約)

$$\Psi_{s_1,s_2,\ldots,s_N} = \mathrm{tTr}\left[T^{(1)}[s_1]T^{(2)}[s_2]\cdots T^{(N)}[s_N]\right]$$

- テンソルの要素数は合計で O(Nd) になる
  - (更に複雑な、s を持たないテンソルを含むネットワークを考えても良い)

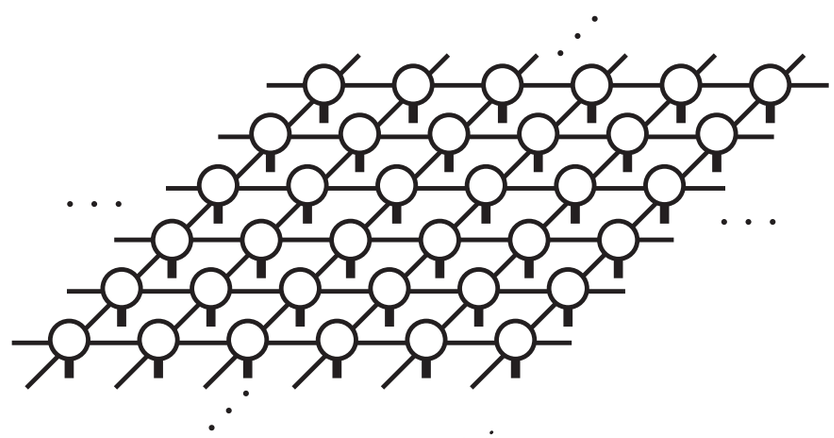


infinite Tensor Product State (iTPS)

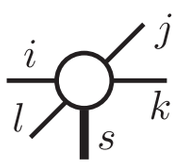
- テンソルネットワークの形（テンソルをどうつなげるか）は任意
  - 解きたい問題に応じてシステマティックに構築する
- ここでは（& TeNeS では）**正方格子状に整列したテンソルが互いに最近接のものと結合している**ようなネットワークを考える
  - さらに、系の（格子の）並進対称性を仮定・利用して、ユニットセル内の L×W 個のテンソルをコピペして並べることで無限系を表現する (**iTPS**)
    - テンソル同士をつなぐ足は virtual bond と呼ばれる
      - virtual bond の次元（取りうる状態の数）をD と表す
        - 次元が一様である必要はない

- (i)TPSが表現できるHilbert 空間の次元は NdD^4
- 相互作用が局所的なハミルトニアンの基底状態はこの中にある（可能性が高い）

$\Psi^{\mathrm{iTPS}} =$



$T_{ijkl}[s] =$

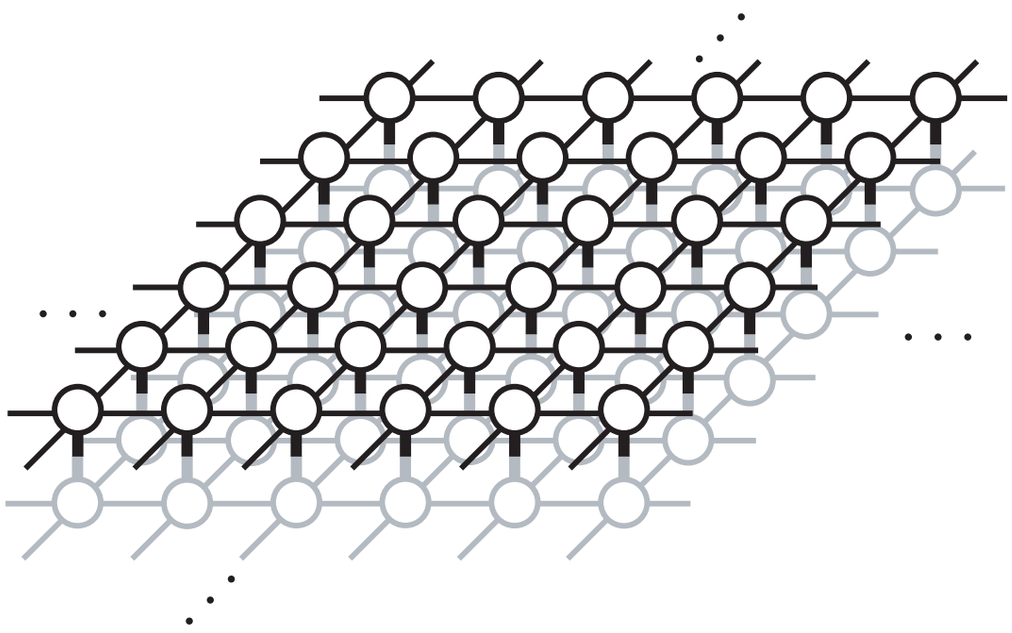


期待値の計算とTN の縮約

- 期待値計算

$$\left\langle \hat{O} \right\rangle_{\Psi} = \frac{\left\langle \Psi \middle| \hat{O} \middle| \Psi \right\rangle}{\left\langle \Psi \middle| \Psi \right\rangle}$$

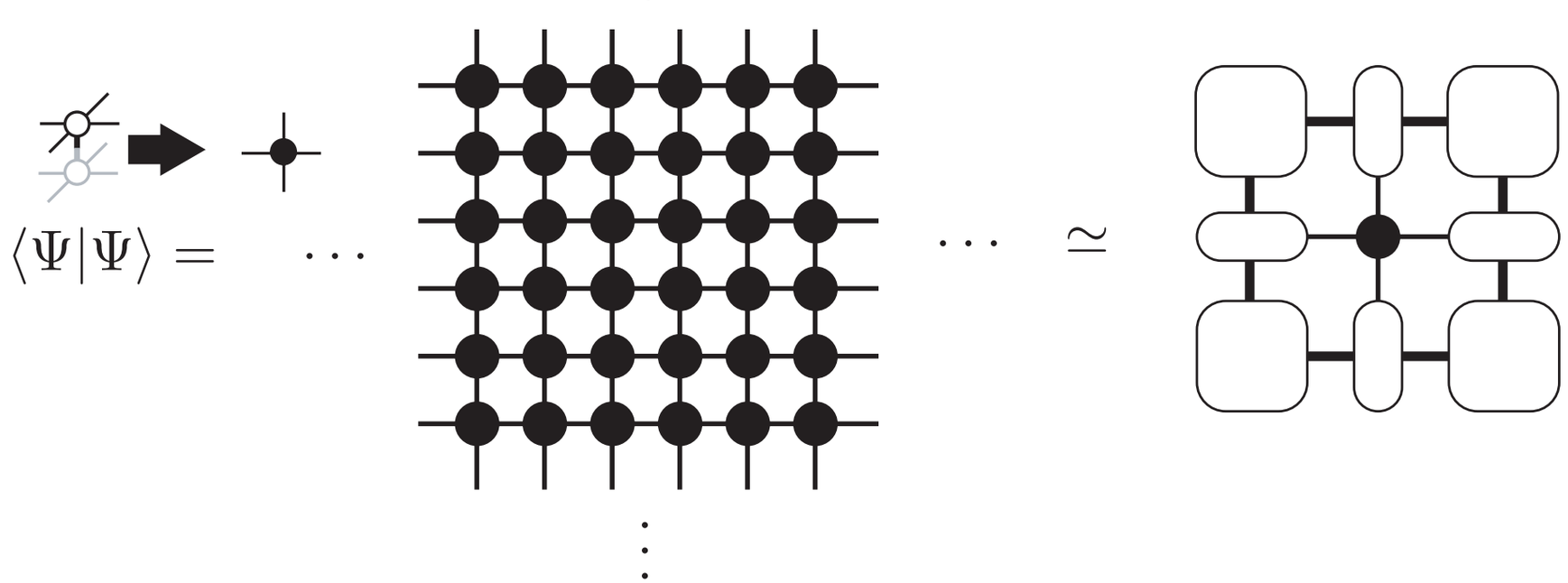
$\langle \Psi | \Psi \rangle =$



- まずはノルム<Ψ|Ψ> を考える
  - 基底として各サイトの完全直交系の直積状態を考えているので、そのまま各サイトのs に関して縮約を取れば良い
  - 表裏をひっくり返して重ねる（**double layered TN**, 右上図）
- iTPS ではこの縮約を厳密に取ることは一般にはほぼ不可能
  - なんらかの近似計算が必要
  - TeNeS では**角転送行列くりこみ群法(CTMRG)** を用いる
    - 半無限に広がった四隅と辺をそれぞれ一つのテンソルで近似する

角転送行列テンソルとエッジテンソル

- DLTN のサイトひとつの s に関する縮約を取って1枚に単純化
- あるサイトに着目して、それ以外を4つの角転送行列テンソルと4つのエッジテンソル（全部まとめて環境テンソル）で近似

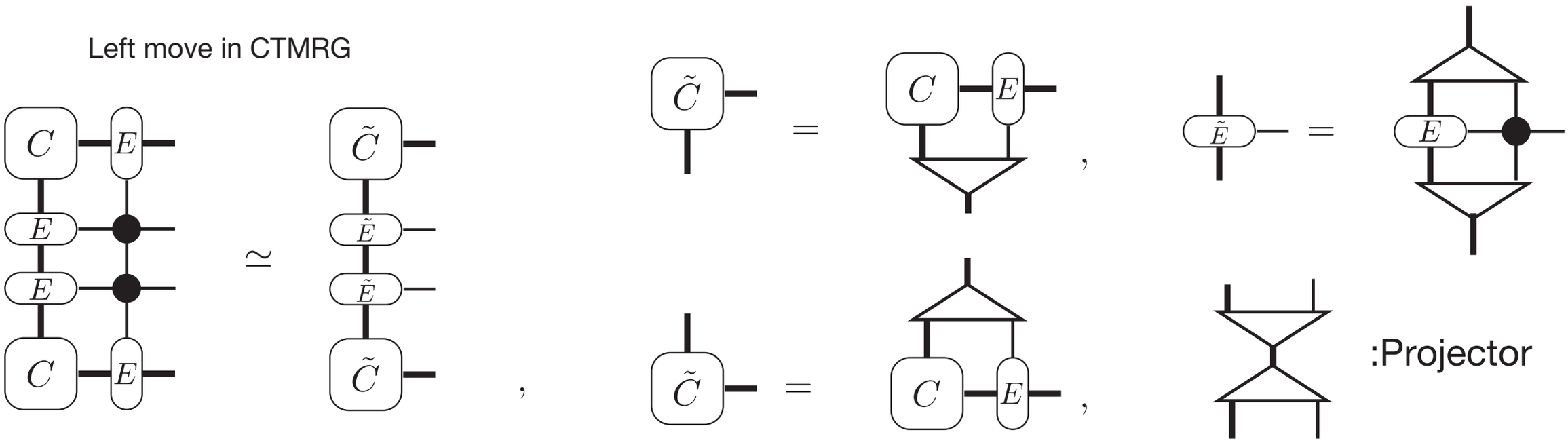


- 角行列テンソルが持つ足の次元を一般に χ で表す
  - 少なくともD 2本分が必要　$\chi \geq D^2$
- サイトテンソル（黒丸）から角転送行列テンソルとエッジテンソルを計算する
  - →　角転送行列くりこみ群法

角転送行列くりこみ群法

T. Nishino and K. Okunishi, JPSJ 65, 891 (1998)

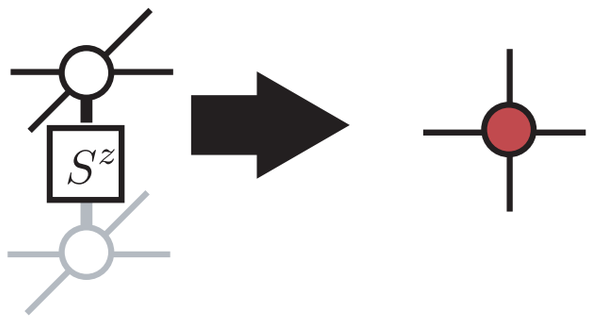
- 角転送行列にはエッジテンソルを、エッジテンソルにはサイトテンソルを吸収させ、自由度を繰り込んで元の形に戻す
- たとえば左側のテンソルに繰り込む left move は以下の通り



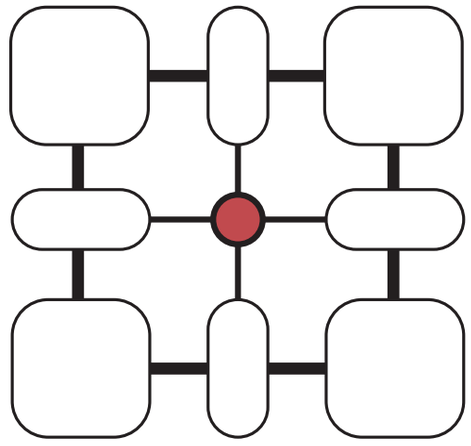
- 一回の吸収でボンド次元が χ から χD^2 に増えてしまうので、自由度を繰り込む操作として **Projector** （三角形のテンソル）を適用する
  - Projector の導出方法はいくつか提案されており、TeNeS では P. Corboz et al., PRL 113, 046402 (2014) （の簡略バージョン）を用いている
- $\chi \propto D^2$ としたとき、必要メモリ量は $O(D^8)$で計算量は $O(D^{10})$
  - Projector の導出に使う特異値分解(SVD) で密行列のfull SVD を行うと $O(D^{12})$

角転送行列くりこみ群法

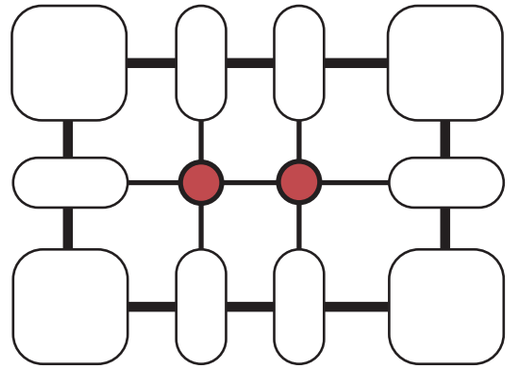
- CTMRG を適当な回数（もしくは収束するまで）行い環境テンソルを求める
- 環境テンソルを得られたらノルムおよび（局所）演算子の期待値が計算可能



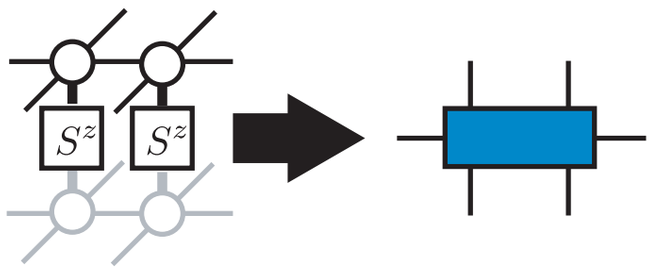
$$\langle\Psi|S^z|\Psi\rangle =$$

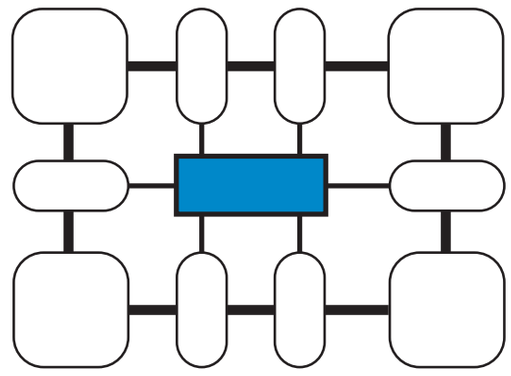


$$\langle\Psi|S^z_i S^z_{i+1}|\Psi\rangle =$$



or





遠距離になるほどコストが増大

TeNeS では 中心部分4x4 まで実装済み

iTPS の基底状態探索

- 虚時間発展 (ITE)

$$|\psi\rangle = e^{-\beta\mathcal{H}}\,|\phi\rangle \Big/ \sqrt{\langle\phi\,|\,e^{-2\beta\mathcal{H}}\,|\,\phi\rangle}$$

  - Simple update

  - Full update

← TeNeS はITEを採用

- 変分最適化

$$|\psi\rangle = \mathrm{minarg}_{\phi}\,\langle\phi\,|\,\mathcal{H}\,|\,\phi\rangle \Big/ \langle\phi\,|\,\phi\rangle$$

  - エネルギーの微分をCTMRG で表現（一般化固有値問題に帰着）

  - エネルギーを自動微分

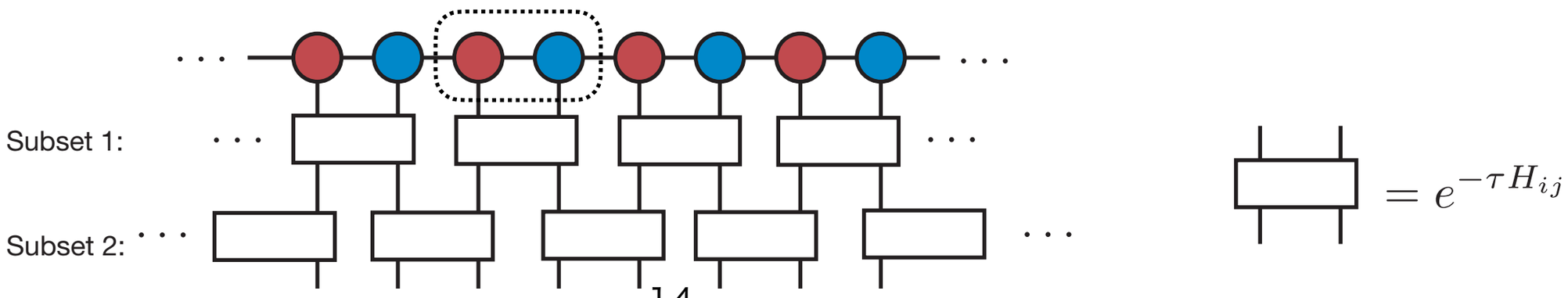
ITE の Suzuki-Trotter 分解

- 虚時間発展演算子を Suzuki-Trotter 分解する　　$\tau = \beta / N_{\mathrm{step}}$

$$e^{-\beta \mathcal{H}} = \left(e^{-\tau \mathcal{H}}\right)^{N_{\mathrm{step}}} = \left(e^{-\tau \sum_p \mathcal{H}_p}\right)^{N_{\mathrm{step}}} = \left(\prod_p e^{-\tau \mathcal{H}_p}\right)^{N_{\mathrm{step}}} + O(\tau)$$

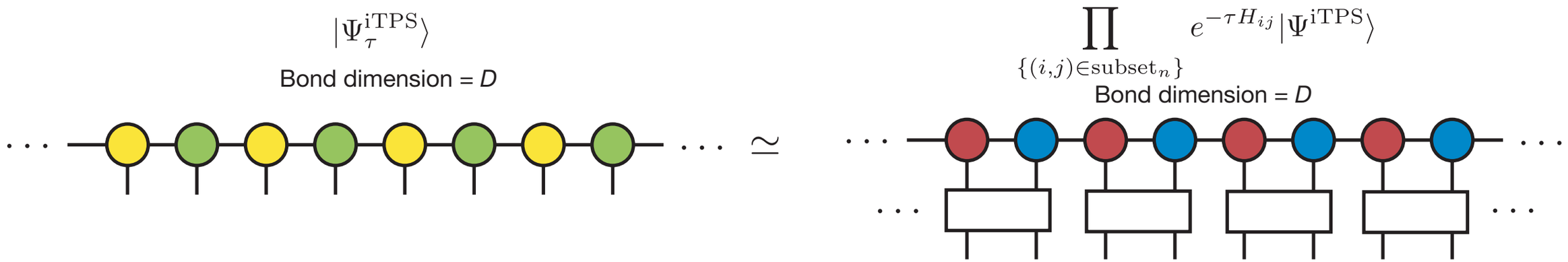
  - 全体のハミルトニアンは部分ハミルトニアン $\mathrm{H_p}$の和で書けるとした
  - この式は、「部分ハミルトニアンによる局所的な、短時間の虚時間発展を多数繰り返すことで所望の虚時間発展を得られる」ことを意味する

- iTPS ではユニットセルを並べることで無限系を表現していた
  - p の和の順番を取り直すことで、ユニットセル同士で等価な部分（e.g. ボンド）に対する短時間虚時間発展をすべて同時に行うようにする



ITE によるテンソル更新

- 局所ITE演算子を作用させると複数のサイトテンソルが合体する
  - 元のiTPS と同じ形（左図）に戻す必要がある（ボンド次元も含めて！）

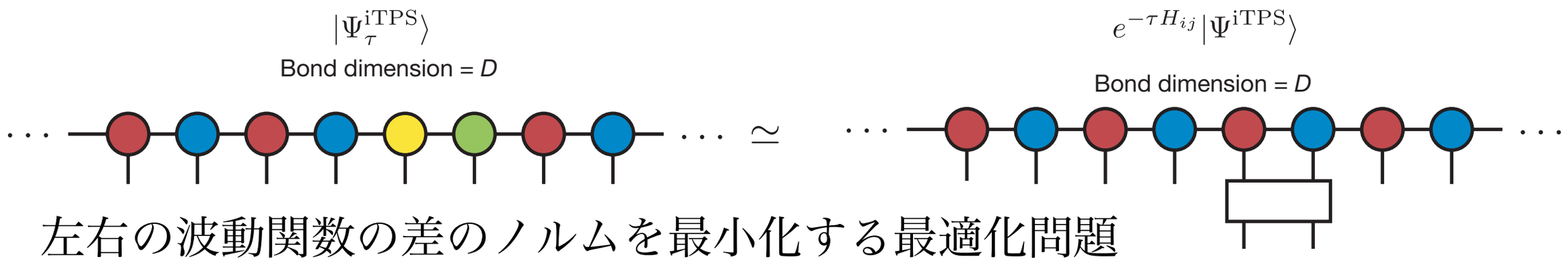


- 左右の波動関数で差のノルムを最小化するのが理想
  - 残念ながら、並進対称性のために非線形問題になっており、困難

Full update

J. Jordan *et al.*, PRL 101, 250602 (2008) など

- すべてを考える代わりに**局所ITEテンソルをひとつだけ作用**する
  - **作用したサイトのテンソルだけ変更する**



  - 左右の波動関数の差のノルムを最小化する最適化問題

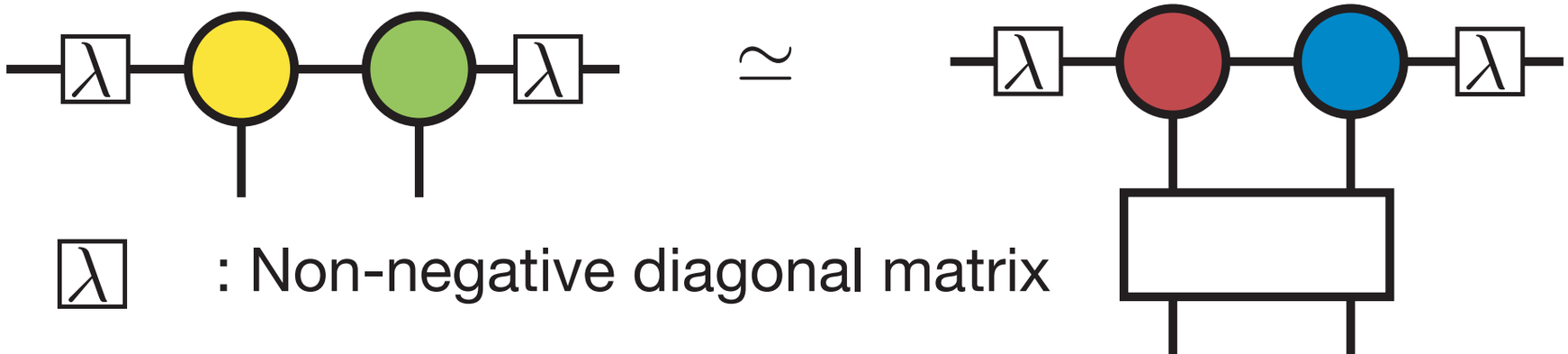
$$\left\| \left|\Psi_\tau^{\mathrm{iTPS}}\right\rangle - e^{-\tau \mathcal{H}_{ij}} \left|\Psi^{\mathrm{iTPS}}\right\rangle \right\|_2^2$$

  - このコスト関数はCTMRG で計算可能
    - 着目するテンソルに関する微分がゼロになるようにする
    - 計算量はCTMRG と同じ O(D^{10})
  - 得られたテンソルを他のユニットセルにコピペすることで全体のITE を行ったとみなす
- TeNeS は亜種である **Fast full update** を採用 (Ho N. Phien et al., PRB 92, 035142 (2015))

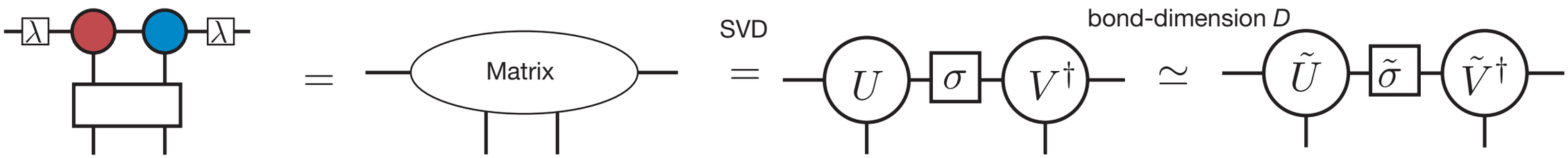
Simple update

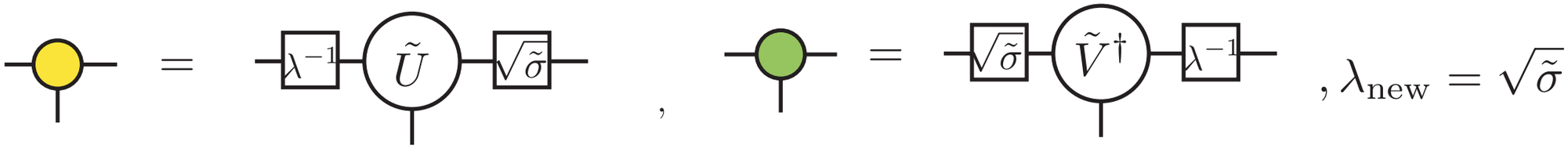
H. G. Jiang *et al.*, PRL 101, 090603 (2008)

- Full update は最適化問題を解くのに系全体を見に行くので重い
- **遠くの情報を平均場として簡略化**し、最適化問題も局所化することにより計算量を大幅に削減する



- 平均場 λ は途中に出てくる特異値 σ から求める

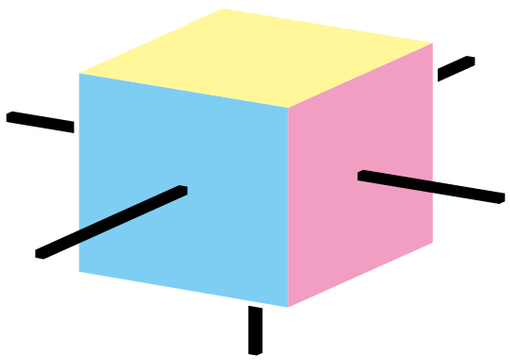




- QR 分解を併用することで $O(D^5)$ の計算量で実行可能
- 初期状態依存性が強く、また秩序変数を過大評価しがち

プログラミング

- テンソルの保持・演算は基本的に既存のライブラリを使うべき
  - 添字アクセス、縮約、テンソル分解
- テンソルライブラリの不完全なリスト
  - C++
    - mptensor, ITensors, Uni10, Cyclops Tensor Framework, TBLIS, ...
      - TeNeS はmptensor を利用
  - Julia
    - テンソル＝多次元配列をnative support
    - TensorOperations.jl, OMEinsum.jl, ITensors.jl, ...
  - Python
    - NumPy (tensordot, einsum), Tensorflow/TensorNetwork, Uni10, ...



TeNeS

Tensor Network Solver for Quantum Lattice Systems

https://www.pasums.issp.u-tokyo.ac.jp/tenes

https://github.com/issp-center-dev/TeNeS

開発チーム

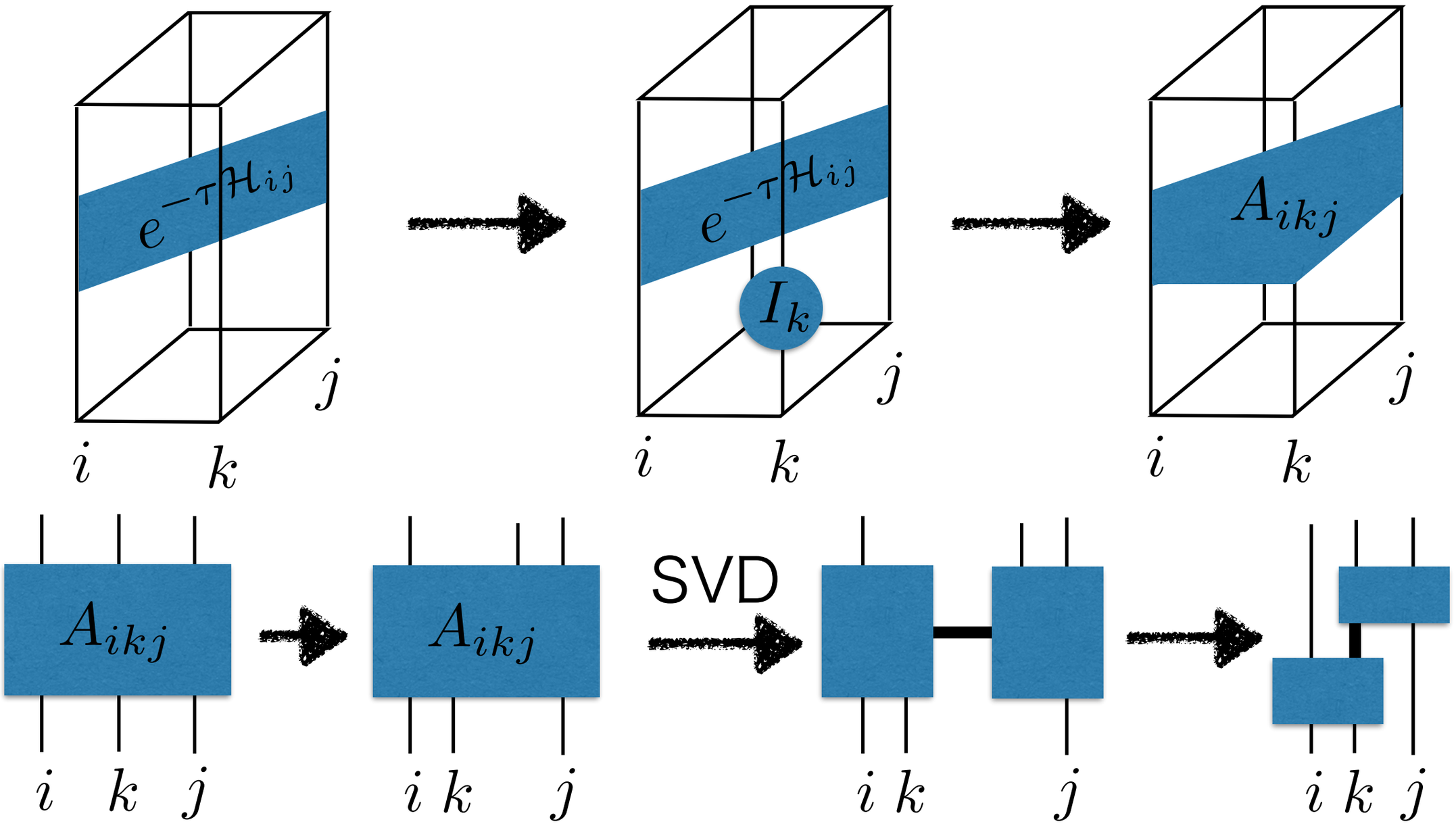
- **大久保毅**（東大院理）
  - アルゴリズム部分の実装
- **森田悟史**（東大物性研）
  - 開発用ライブラリ・ツール作成（mptensor, tensordot）
- **本山裕一**（東大物性研）
  - メインプログラム・入力ファイル作成ツールの設計・実装
- **吉見一慶**（東大物性研）
  - ユーザテスト、サンプル（自動実行スクリプトなど）の作成、Pj マネージメント
- **加藤岳生**（東大物性研）
  - ユーザテスト、サンプル作成
- **川島直輝**（東大物性研）
  - プロジェクトリーダー

TeNeS の特徴

- **二次元量子格子模型の基底状態探索**を行うソフトウェア
  - **正方格子iTPS** 波動関数を **ITE (SU/FU)** で最適化する
  - 無限系の縮約には**CTMRG** を利用する
  - 現状、フェルミオン系は扱えない（スピンとボソンがメイン）
- 正方格子上の任意の短距離2サイト相互作用を計算可能
  - 現状はx, y 方向にそれぞれ3サイト先まで
  - 相互作用の形を工夫することで正方格子以外も計算可能
    - 例：正方格子の斜め方向に入れて三角格子
- 1サイト演算子・2サイト演算子の期待値が計算可能
  - 多サイト演算子や相関長の計算は導入予定
- テンソル演算には **mptensor** を利用
  - MPI 実行するだけで ScaLAPACK を利用した分散メモリ並列がなされる

TeNeS の特徴

- TeNeS のITE は簡単のために正方格子最近接ボンドのみ扱う
  - 長距離ITE テンソルは正方格子最近接 ITE テンソルの積に変換しておく



TeNeS の特徴

- よく使われそうな模型・格子は定義済み
  - 模型
    - 量子スピン模型（スピンの大きさ Sは任意）

$$\mathcal{H} = \sum_{i<j} \left[\left(\sum_{\alpha}^{x,y,z} J_{ij}^{\alpha} S_i^{\alpha} S_j^{\alpha} \right) + B_{ij} \left(\vec{S}_i \cdot \vec{S}_j \right)^2 \right] - \sum_i \left[\sum_{\alpha}^{x,y,z} h^{\alpha} S_i^{\alpha} + D \left(S_i^z \right)^2 \right]$$

    - Bose-Hubbard 模型（粒子数カットオフ付き）

$$\mathcal{H} = \sum_{i<j} \left[-t_{ij} \left(b_i^{\dagger} b_j + \mathrm{h.c} \right) + V_{ij} n_i n_j \right] + \sum_i \left[U \frac{n_i (n_i - 1)}{2} - \mu n_i \right]$$

  - 格子
    - 格子回転異方性や最近接・2次近接・3次近接相互作用を導入可能
    - 正方格子　　詳しくはマニュアル参照
    - 三角格子
    - 蜂の巣格子
    - カゴメ格子

TeNeS の目標

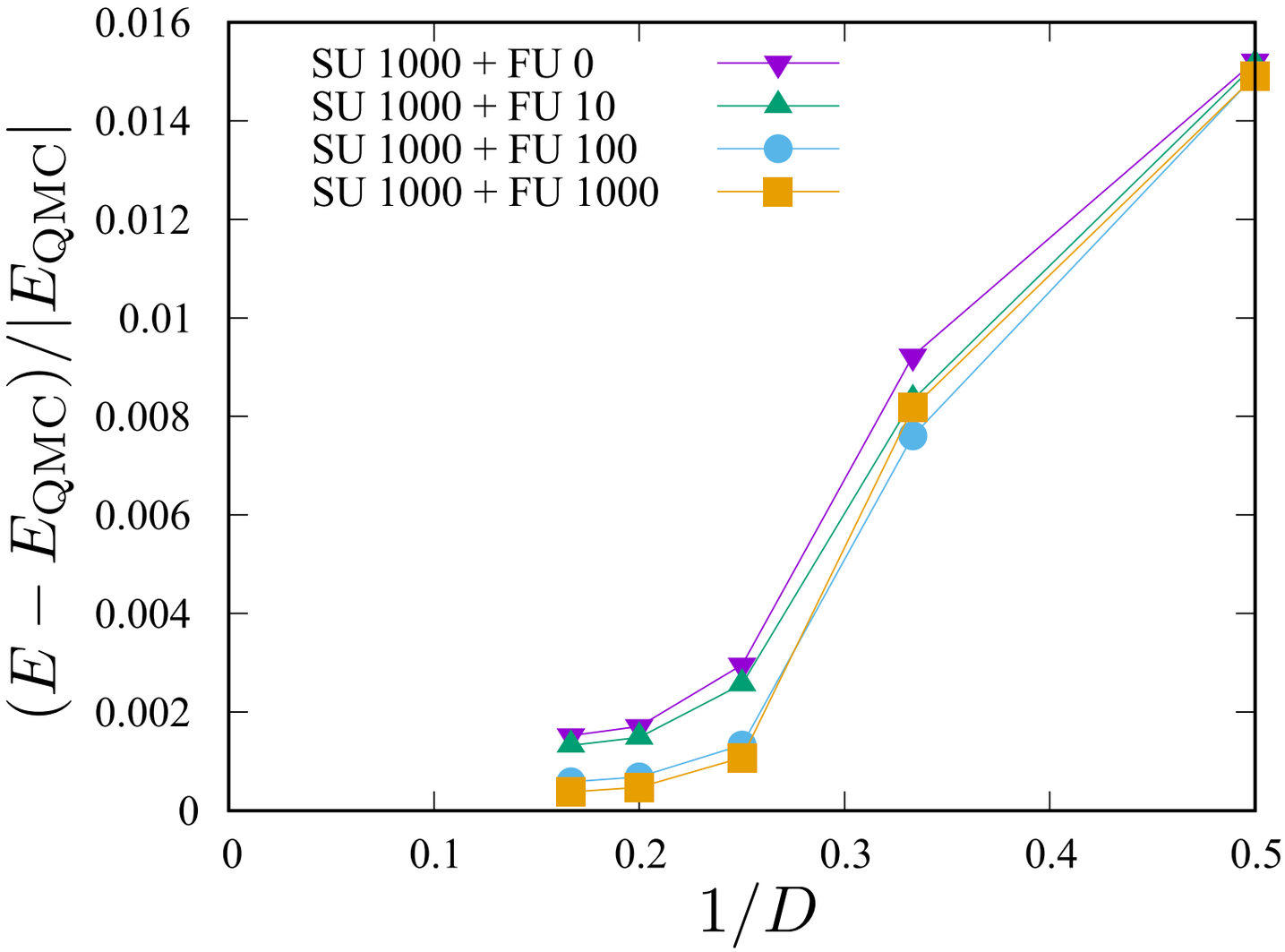
- 量子格子模型におけるテンソルネットワーク法研究への参入障壁を下げる
  - TN 法のアルゴリズム・プログラム開発
    - テンソル演算のライブラリはそれぞれの言語で多数存在
    - アルゴリズムも（ダイアグラム表記のおかげで）割とわかりやすい
    - 実際に1から組み上げるのは結構難しい（つらい）
      - テンソルの添字の順番など、システマティックに構築しないとすぐに混乱する
      - 複数のテンソルを縮約する場合、順番によって計算量オーダーが変わる
- 計算結果をどう評価するか？
  - ボンド次元など、ハイパーパラメータが結構多い
- **TeNeS をベンチマークとして用いる！**
  - 計算結果の比較
  - ソースコードを読む・TeNeS をベースに新アルゴリズムを入れる
    - GNU GPL v3 であることに留意

TeNeS の目標

- 量子格子模型におけるテンソルネットワーク法研究への参入障壁を下げる
  - 別の分野の研究者
    - 実験家
      - 実験データのフィッティングなど
    - TN 以外の手法開発者
      - TN 法との比較のために使う
- 使いやすさや汎用性を重視
  - 特定の模型や格子に強く依存した最適化はしない
  - 常に正方格子iTPS として扱う
- **「とりあえずTN 計算を試してみる」ができるようにする**

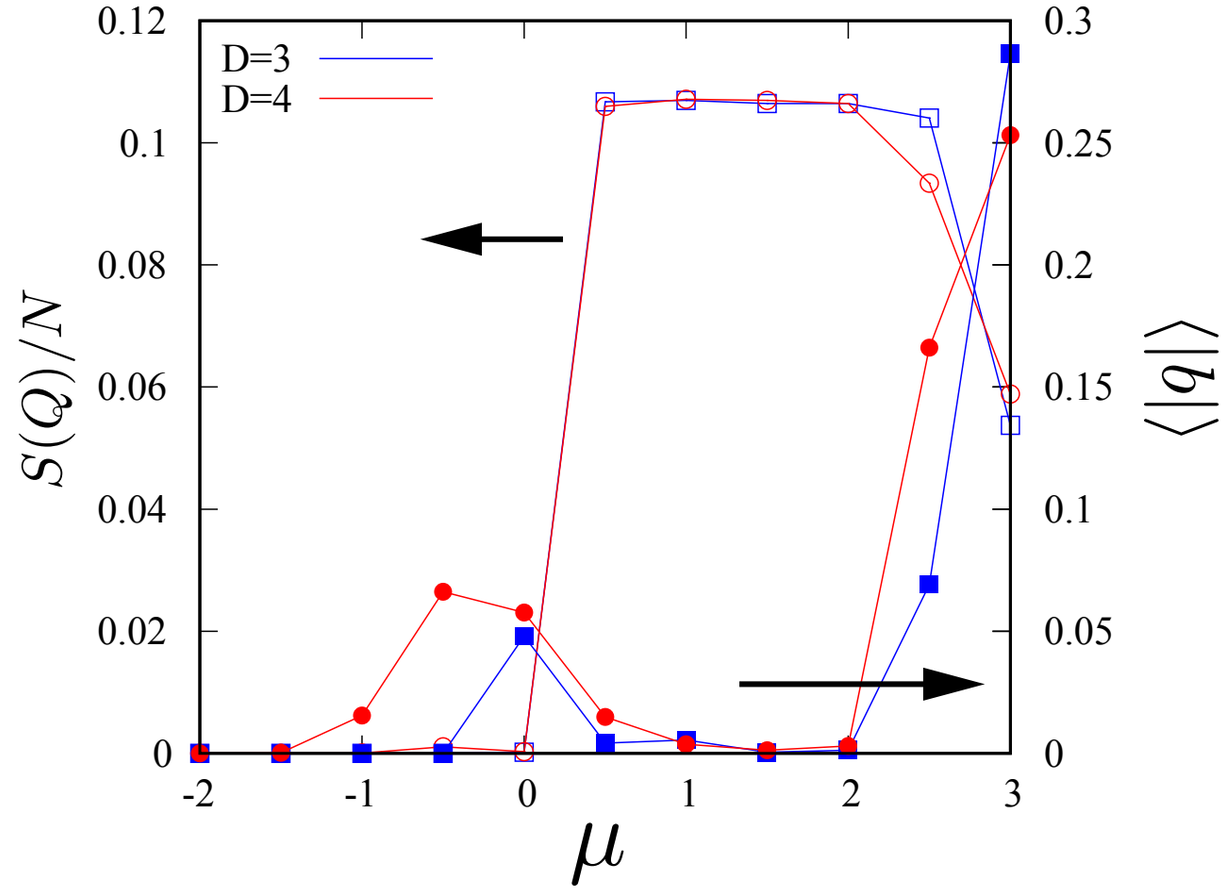
計算例：正方格子S=1/2反強磁性ハイゼンベルグ模型のエネルギー

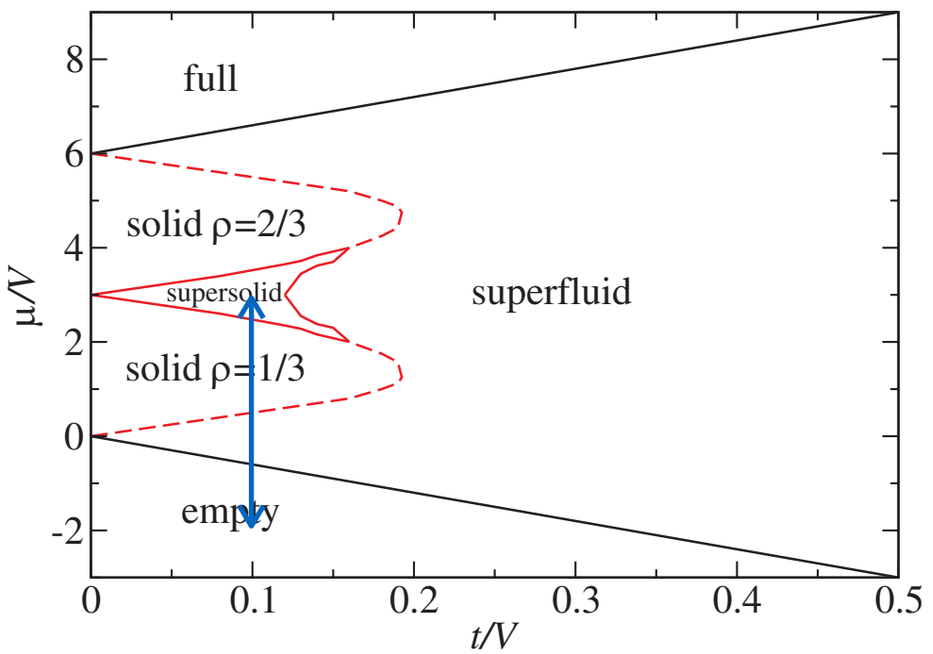
- 正方格子S=1/2反強磁性ハイゼンベルグ模型の基底エネルギー
- 横軸はボンド次元の逆数
- 縦軸はQMC による見積もりからの相対誤差
  - A.W. Sandvik, AIP Conf. Proc. **1297**, 135 (2010)
- ユニットセルの大きさは2×2
- χ =D²
- 24CPU の計算機で数時間程度



計算例：三角格子ハードコアボースハバード模型

- 1/3 相と 2/3 相の間に超固体相がある
  - 超流動性と固体性が同時に存在
- t=0.1, V=1.0
- D=3 および D=4
- $\chi = D^2$
- ユニットセルは 3×3
- SU 2000 step, FU なし
- 構造因子S(Q)（左軸） と超流動オーダー|b|（右軸）
  - Q は√3×√3 オーダーの波数
- 先行研究(QMC) と矛盾しない結果





Wessel and Troyer, PRL 95, 127205 (2005)

TeNeS の使い方 -- 前準備

- TeNeS に必要なもの
  - C++11 に対応したコンパイラが必要
    - よほど古い計算機でなければ問題ないはず
  - BLAS/LAPACK が必要
    - 分散並列をしたい場合は MPI, ScaLAPACK も必要
  - インストールには CMake3.6 が必要
    - 内部で依存ライブラリをダウンロードするために Git も必要
- 入力ファイル生成ツール (tenes_simple, tenes_std) に必要なもの
  - Python3
    - モジュールとして numpy, scipy, toml が必要

TeNeS の使い方 -- インストール

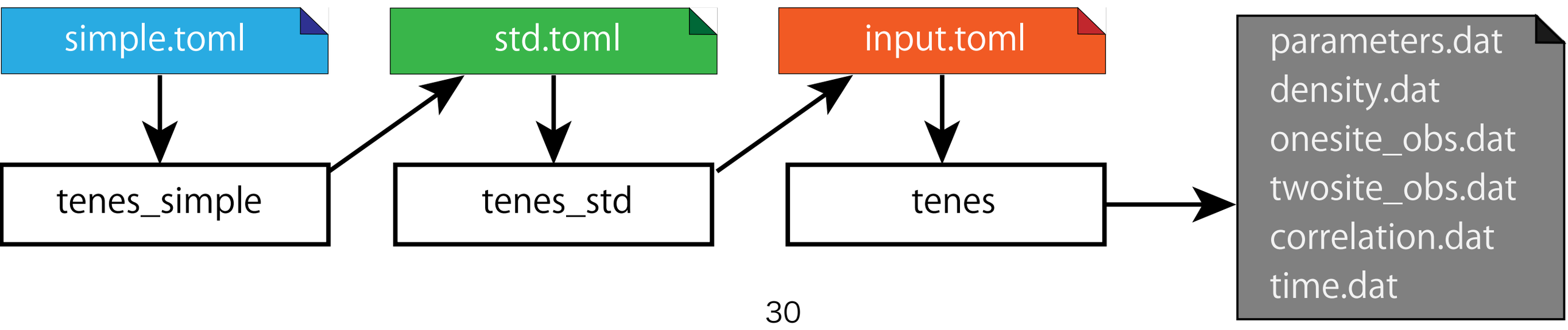
- TeNeS のダウンロード
  - https://github.com/issp-center-dev/TeNeS
    - リリースページから適当なバージョンのtar をダウンロード・展開
    - もしくは git clone
- TeNeS のインストール

```
$ cd TeNeS                    作業ディレクトリを作成・移動
$ mkdir build && cd build
$ cmake -DCMAKE_CXX_COMPILER=`which icpc` \⏎          コンパイラの指定
        -DCMAKE_INSTALL_PREFIX=$HOME/opt/tenes \⏎
        ../                                  インストール先の指定
$ make install
```

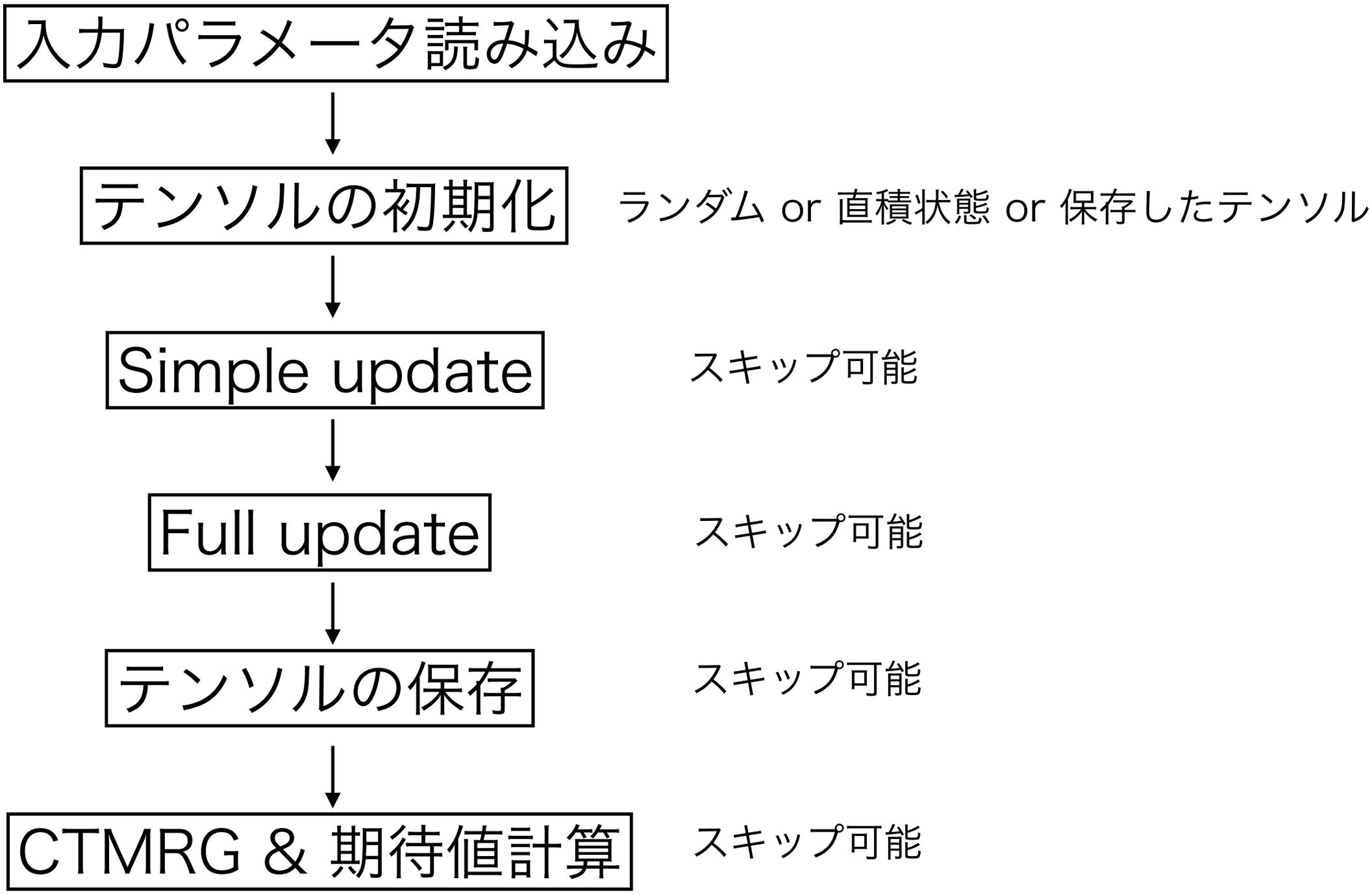
- $HOME/opt/tenes/bin に実行ファイル tenes, tenes_std, tenes_simple
- $HOME/opt/tenes/share 以下にサンプルファイル

TeNeS の使い方 -- 実行フロー

- tenes_simple
  - 定義済み模型・格子のパラメータからハミルトニアンやユニットセル情報を生成するツール
- tenes_std
  - ハミルトニアンやユニットセル情報からITE テンソルを生成するツール
  - 長距離ITE テンソルの分解も行われる
- tenes
  - 実際の計算を行うメインプログラム



TeNeS の使い方 -- 内部フロー



例:S=1/2 正方格子反強磁性ハイゼンベルグ模型

SU はτ=0.01 を1000ステップ

FU は行わない

角転送行列のボンド次元は16

正方格子
ユニットセルは2x2
ボンド次元は4

スピン系
S=1/2 （デフォルト）
反強磁性ハイゼンベルグ

```
$ cat simple.toml
[parameter]
[parameter.simple_update]
tau = 0.01        # SU の虚時間刻み
num_step = 1000   # SU のステップ数

[parameter.full_update]
tau = 0.01        # FU の虚時間刻み
num_step = 0      # FU のステップ数

[parameter.ctm]
dimension = 16    # ボンド次元 χ

[lattice]
type = "square lattice" # 正方格子
L = 2             # ユニットセルの長さ
W = 2             # ユニットセルの幅
initial = "antiferro" # 初期状態
virtual_dim = 4   # ボンド次元 D

[model]
type = "spin"     # スピン模型
J = 1.0           # 交換相互作用
```

例:S=1/2 正方格子反強磁性ハイゼンベルグ模型

```
$ $HOME/opt/tenes/bin/tenes_simple simple.toml  # convert to std.toml
$ $HOME/opt/tenes/bin/tenes_std std.toml        # convert to input.toml
$ $HOME/opt/tenes/bin/tenes input.toml          # perform calculation
    ... 進捗報告 （省略） ...

Onesite observables per site:
  Sz           = 5.70170299863e-12 8.24461048345e-19
  Sx           = -1.93698629873e-08 -8.53148278222e-18
  Sy           = 3.12611410373e-08 -4.29087879573e-18
Twosite observables per site:
  hamiltonian = -0.66746306716 2.74670113479e-17
  SzSz         = -0.345269357812 -1.31655470664e-17
  SxSx         = -0.161096823276 -4.72116684558e-18
  SySy         = -0.161096825628 9.13209512715e-18
    Save elapsed times to output/time.dat
Wall times [sec.]:
  all            = 28.396518775
  simple update = 7.993189097
  full update   = 0
  environmnent  = 18.26470468
  observable    = 2.068691087

Done.
```

$E_{\mathrm{QMC}}/N = -0.6694422$

output ディレクトリには
サイトごとの物理量など、
もう少し詳細な出力ファイルが生成される

まとめ

- 二次元量子格子模型テンソルネットワークソルバーTeNeS の紹介をした
  - TeNeS で用いられる TN 法のアルゴリズムをレビューした
    - iTPS, 角転送行列くりこみ群法, 虚時間発展 (simple/full update)
  - TeNeS の特徴と目標を紹介した
  - TeNeS を用いた計算例と、使い方を紹介した
- 今後の目標（下ほど難易度が高そう）
  - 出力の増強
    - 多サイト演算子、相関長
    - 後処理ツール (フーリエ変換や構造因子など)
  - 基底状態以外の計算
    - 実時間発展、有限温度計算
  - 変分最適化
  - フェルミオン系

質問・要望は
GitHub のissue か
メール (tenes-dev@issp.u-tokyo.ac.jp)へ！

付録

二次元量子格子模型のアルゴリズム

- 厳密対角化（数値的対角化, ED）
  - ハミルトニアンを Lanczos, LOBPCG などの手法で「対角化」して基底状態や低エネルギー励起状態の固有エネルギー・固有ベクトルを得る
  - 関連技術として有限温度のカノニカル平均も計算可能(e.g., TPQ state)
  - 出てくる結果は厳密で、系の詳細（性質）にはよらずに適用可能
- システムサイズに非常に強烈な制限がかかる（指数関数的なコスト）
  - S=1/2 ではたかだか 30-40 site
  - 国策レベルのスパコン上で、スピンや運動量に対する部分対角化など、チューニングしたプログラムを用いて 50 site 行くかどうか
- プログラムパッケージ
  - HΦ, TITPACK, SpinPACK など

二次元量子格子模型のアルゴリズム

- 経路積分モンテカルロ法（世界線モンテカルロ法, PIQMC）
  - 分配関数を虚時間経路積分展開ないしはテイラー展開することで、D+1 次元上の世界線表示に変換し、（マルコフ連鎖）モンテカルロ法で物理量の有限温度期待値を計算する手法
  - 模型・格子・次元によらずに統計誤差の範囲内で厳密な結果が得られる
    - 数千、数万サイトの計算が任意温度で可能
- 符号問題
  - モンテカルロ法で使う重みとして負や複素数が出てくる事がある
    - フラストレートスピン系やフェルミオン系など
  - 統計誤差がサイト数および逆温度に対して指数関数的に増大する
    - → 一定の精度を得るのに必要なコストが指数関数的に増大
- プログラムパッケージ
  - ALPS/Looper, DSQSS など

二次元量子格子模型のアルゴリズム

- 変分モンテカルロ法 (VMC)
  - 量子力学の変分原理を利用して基底状態波動関数を探索する手法
  - 手持ちの波動関数のエネルギー（ハミルトニアンの期待値）を計算して、エネルギーが小さくなるように波動関数を更新していく
  - 系の状態そのものをサンプリングするのではなく、物理量の期待値計算に現れる完全系($\Sigma |x><x|$)をサンプリングする
    - 符号問題が発生しない
- そこそこ大きな系の計算が可能(数百サイト)
- 波動関数をどう表現するかが重要となる（変分波動関数）
  - 最近はボルツマンマシンのような機械学習的手法や量子ゲートのような量子計算機を利用した変分波動関数が提案されており、結構hot
- プログラムパッケージ
  - mVMC など