

Juliaと量子多体計算

スパースモデリングからQuantics Tensor Trainまで

品岡寛

埼玉大学

- 学術変革領域研究B 「量子古典融合アルゴリズムが拓く計算物質科学」
- JSTさきがけ 「量子情報処理」
- 基盤研究B 「次元圧縮技術が拓く二粒子応答の第一原理計算」

自己紹介

物性の定量的予言がしたい → 計算物理 + 第一原理計算

Fortran, Ruby, C/C++, Python, Julia と使ってきて、今はJuliaがメイン

- Python的な気楽な書き方から、Fortran的な速度優先の書き方まで可能
- 学生が勉強しやすい

- ALPS/CT-HYB C++による量子モンテカルロ法実装 (C++)

<https://github.com/ALPSCore/CT-HYB>

- SparseIR.jl/sparse-ir 虚時間グリーン関数のスパースモデリング (Python/Julia)

<https://github.com/SpM-lab/SparseIR.jl>

<https://github.com/SpM-lab/sparse-ir>

- DCore 動的平均場近似計算 (Python)

<https://github.com/issp-center-dev/DCore>

- SpM 数値解析接続 (C++)

<https://github.com/SpM-lab/SpM>

量子古典融合アルゴリズムが拓く計算物質科学 (2023年4月～)

<https://qc-hybrid.github.io/>

領域代表: 品岡寛

The screenshot shows a web browser window with the URL <https://qc-hybrid.github.io/>. The page has a navigation bar with the following items: 量子古典融合 (Quantum-Classical Hybrid), 学術変革領域研究 (B) (Academic Transformation Area Research (B)), Overview (領域概要), Organization (研究組織), Events (イベント), Recruitment (公募情報), and Achievements and Outreach (研究成果・アウトリーチ). The main content area features a blue header with the title "Computational materials science based on quantum-classical hybrid algorithms" and the subtitle "量子古典融合アルゴリズムが拓く計算物質科学". Below this, there are three columns of research areas:

A01: Quantum embedding	A02: Wave function theory	A03: Quantum circuit
Hiroshi Shinaoka (Saitama Univ) Construction of effective models for matter using quantum embedding theory	Tsuyoshi Okubo (Univ. Tokyo) Development of wave function theory combined with quantum algorithms	Wararu Mizukami (Osaka Univ.) Pioneering quantum circuit for computational material science
A01: 量子埋め込み	A02: 波動関数理論	A03: 量子回路設計
品岡 寛 (埼玉大学) 量子埋め込み理論による物質の有効モデル構築	大久保 毅 (東京大学) 波動関数理論と量子アルゴリズムの融合	水上 渉 (大阪大学) 計算物質科学のための量子回路設計
Learn more	Learn more	Learn more

概要

計算物理は、桁違いに異なる長さスケールの共存を扱う

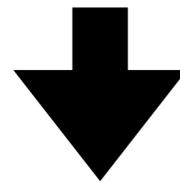
例 量子物性物理 (低温物理現象 vs 10^1 eV ~ 10^5 Kのバンド幅)

宇宙ひも (プランクスケールの細さ vs 宇宙スケールの長さ)

乱流

幅広い長さスケールを扱える表現の開発→試行錯誤 & 並列計算

Quantics tensor train

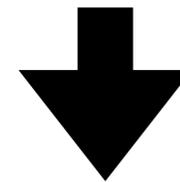


連続変数関数を離散化した「テンソルネットワーク」表現

MSSTA.jl (仮名)

2009年に応用数学で提案され、ここ数年で流体力学、量子力学等への適用が始まる

グリーン関数のスパースモデリング



悪条件の積分方程式の擬似四倍精度演算
虚時間の最適基底を構成

SparseIR.jl

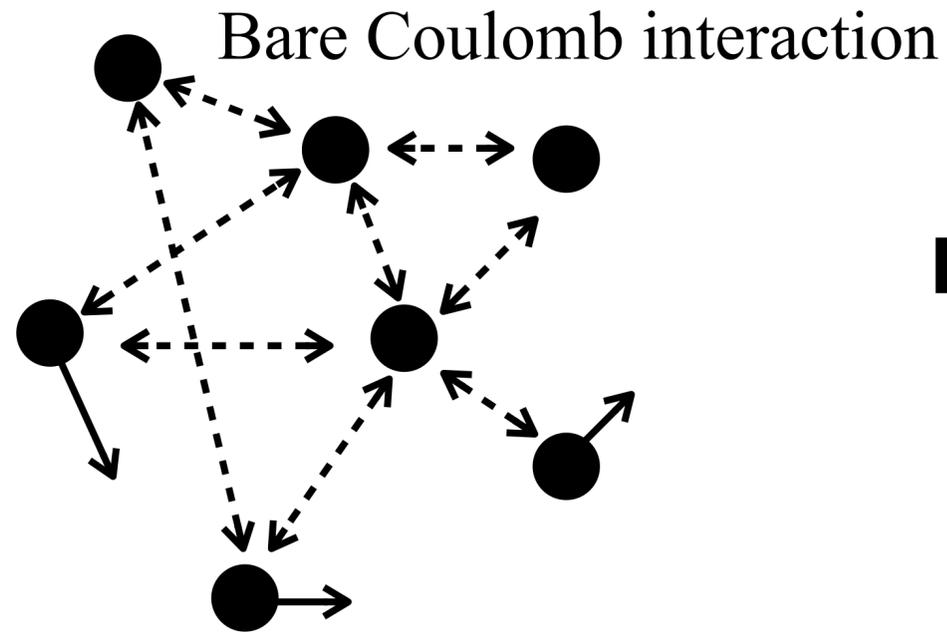
品岡等によって2017年に提案され、虚時間形式の場の量子論・第一原理計算に応用が進む

発表の構成

- 背景 時空データの圧縮
- Quantics tensor train 連続時空の離散化とテンソルネットワーク
- スパースモデリング 疑似多倍精度による積分方程式の解法
- 今後のJulia応用について

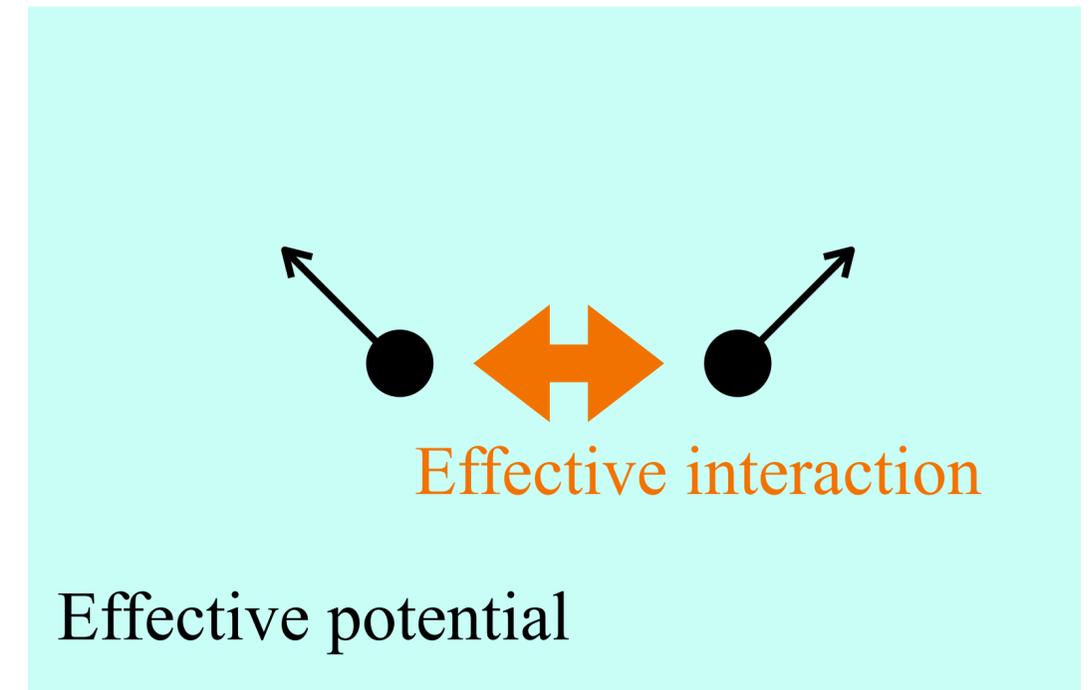
情報圧縮技術がなぜ必要？ (1/2ページ)

量子多体系



厳密な解法は指数的計算コスト

有効少数系



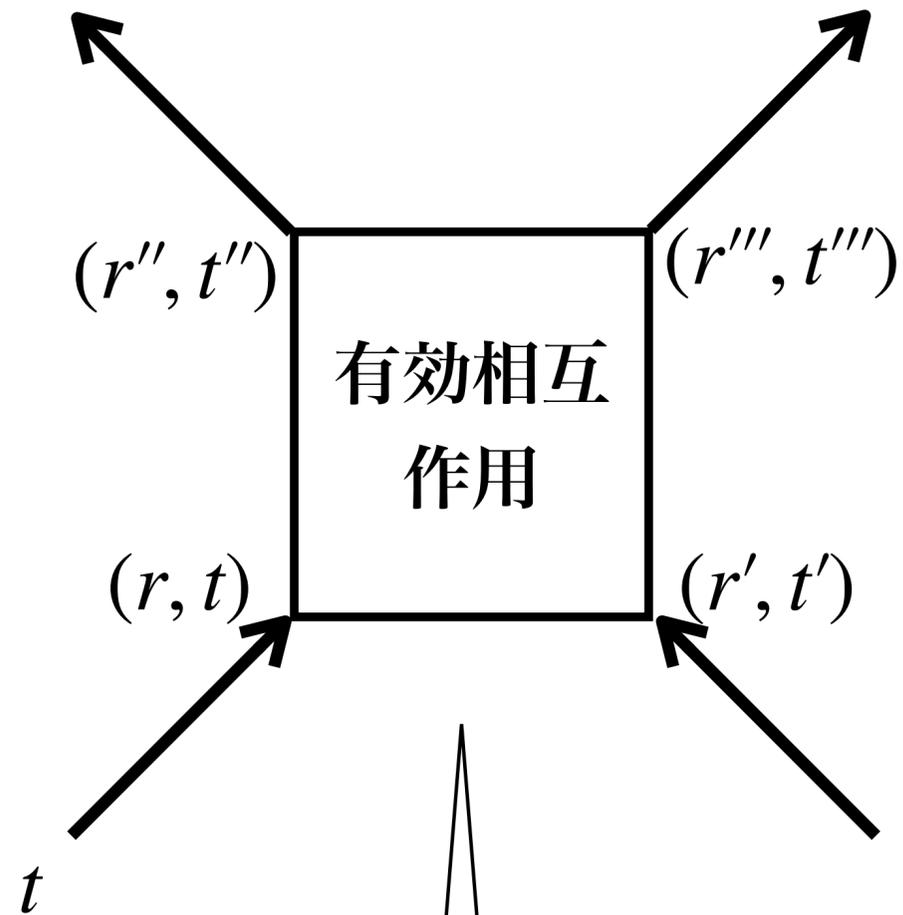
密度汎関数 (Kohn-Sham), 摂動論など

グランドチャレンジ

- より精度の良い近似を捜す (高次の摂動論、機械学習？など)
- 少数系ですら計算が難しい → 次ページ

情報圧縮技術がなぜ必要？ (2/2ページ)

2つの粒子が散乱される過程



$$N_t = 10^3, N_r = 100^3 \rightarrow (N_t N_r)^4 = 10^{36}$$

桁違いに異なるスケールの混在

→単純メッシュではデータ量が発散

→圧縮&演算する技術が必要

流体力学計算等でも同じような問題

テンソルネットワーク、スパースモデリング...

→試行錯誤が迅速で、並列計算までカバーする

Juliaが最適

Quantics tensor train

- [HS](#) *et al.*, Physical Review X **13**, 021015 (2023)
- M. K. Ritter, Y. N. Fernández, M. Wallerberger, J. von Delft, [HS](#), X. Waintal, arXiv:2303.11819

<https://github.com/shinaoka/MSSTA.jl>

Quantics tensor train (QTT) とは？

I. V. Oseledets, Doklady Math. **80**, 653 (2009)

B. N. Khoromskij, Constr. Approx. **34**, 257 (2011)

桁違いに異なる長さスケールが混在する関数を圧縮可能

Image compression



José I. Latorre, arXiv:quant-ph/0510031v1

Vlasov-Poisson equations for collisionless plasmas

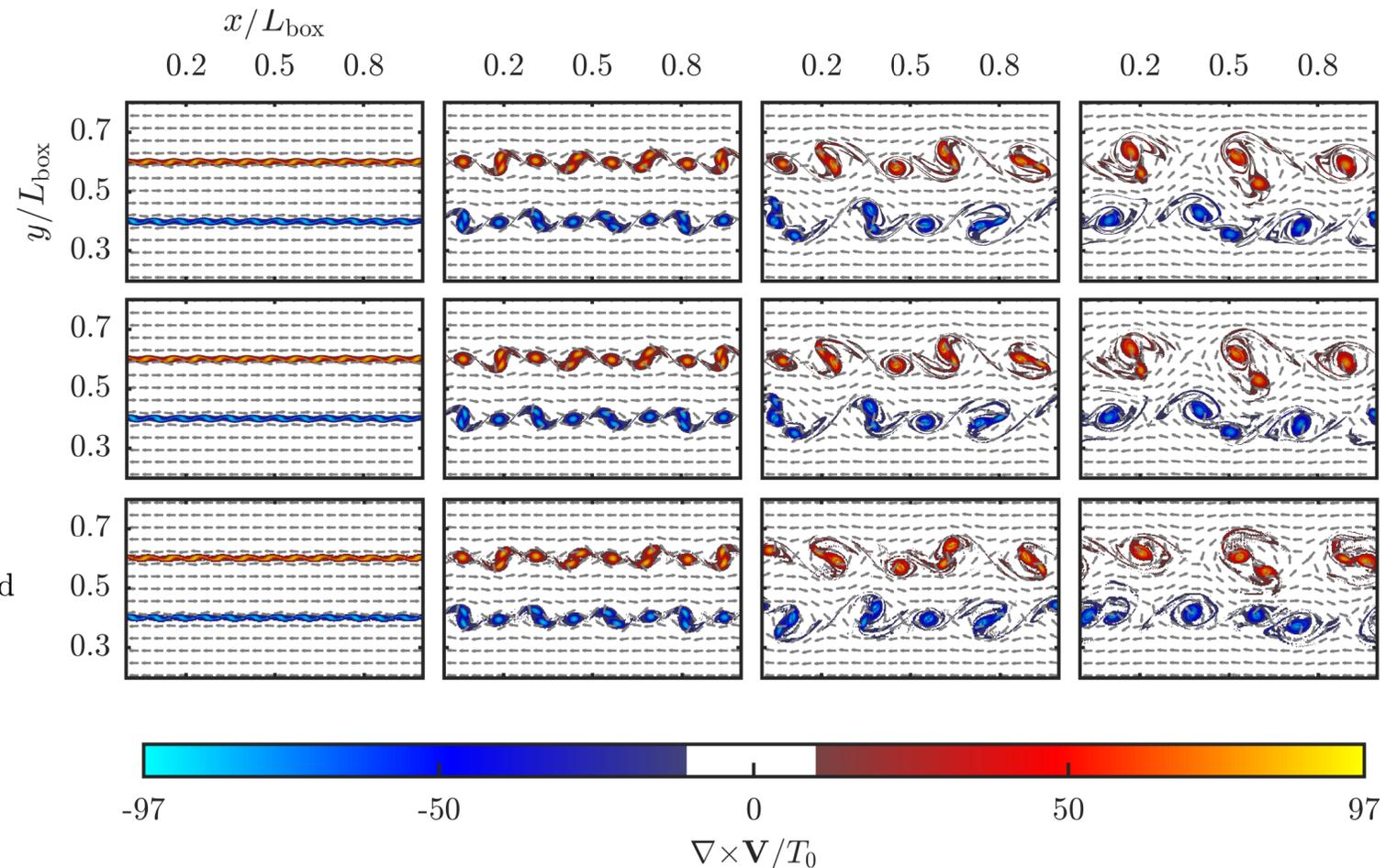
E. Ye and N. F. G. Loureiro, arXiv:2205.11990

Bond dimension

362^2 grid

DNS

$\chi=118$



N. Gourianov *et al.*, Nat. Comput. Sci. **2**, 30 (2022)

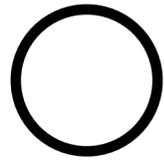
場の量子論の時空依存性を圧縮・演算できる！ HS *et al.*, PRX **13**, 021015 (2023)

2022年8月にアイデア思いつく、2022年10月に投稿 (Juliaだから可能だった)

テンソルネットワーク (1/3)

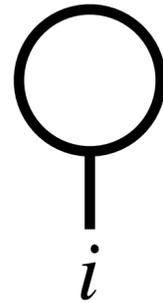
スカラー

a



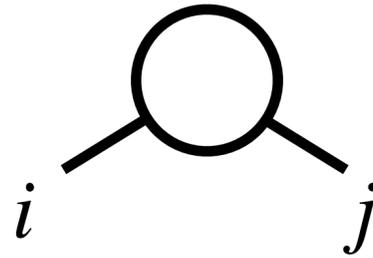
ベクトル

a_i



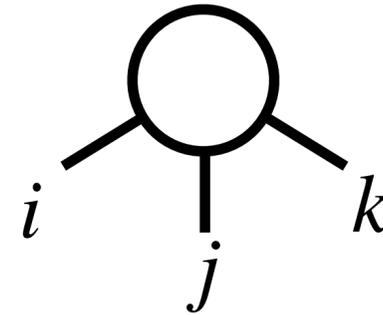
行列

a_{ij}



3階テンソル

a_{ijk}



突き出している足がindexに対応。突き出している向き、図形の形・色に意味はない。

テンソルネットワーク

$$i \text{---} [A] \text{---}^j [x] = \sum_j A_{ij} x_j$$

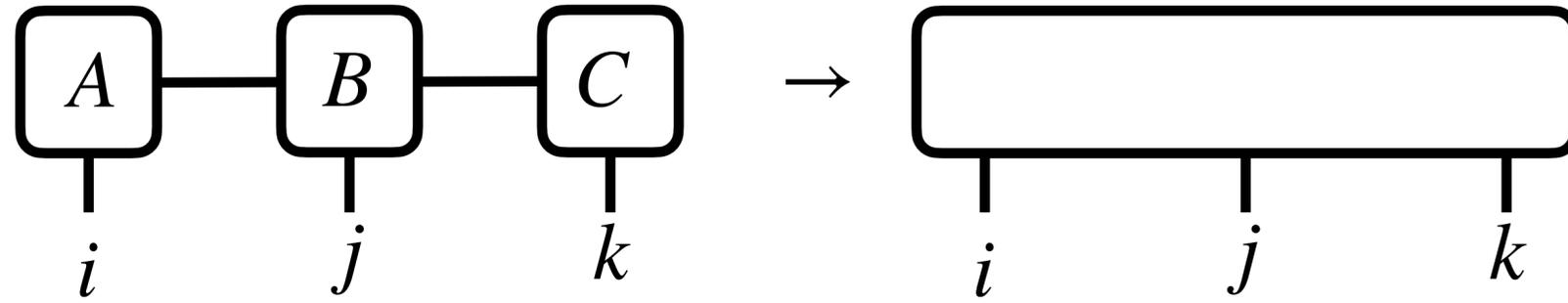
共通している足に対する和を取る。

$$\begin{matrix} [A] & \xrightarrow{\alpha} & [B] & \xrightarrow{\beta} & [C] \\ | & & | & & | \\ i & & j & & k \end{matrix} = \sum_{\alpha, \beta} A_i^\alpha B_j^{\alpha, \beta} C_k^\beta$$

上・下付きにも意味はない。

テンソルネットワーク (2/3)

Contraction

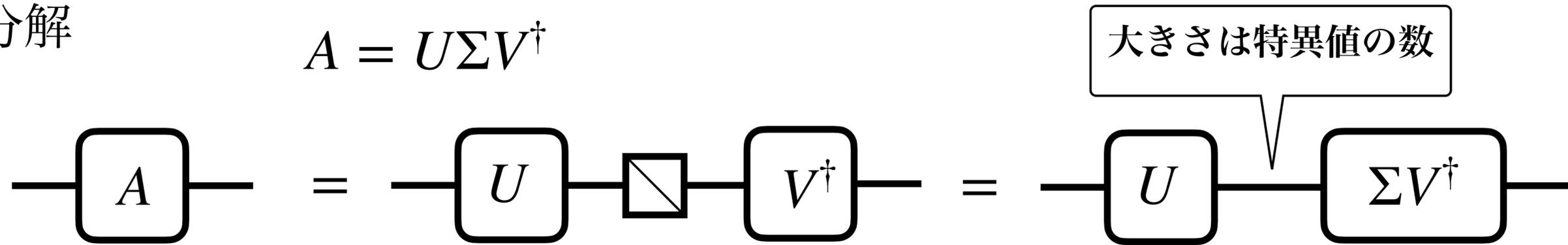


実際にvirtual bondに対する和を計算することをcontractionをとるといふ。行列積として実行可能

テンソルネットワーク (3/3)

特異値分解

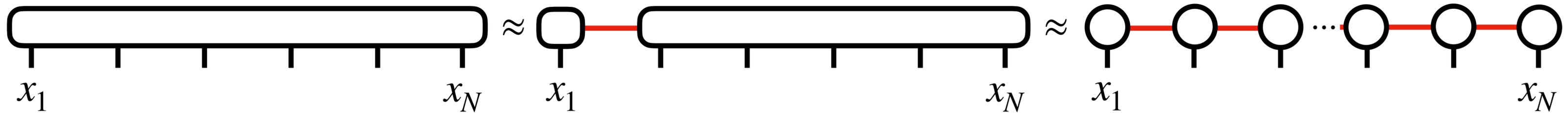
$$A = U\Sigma V^\dagger$$



$$(\Sigma)_{ij} = \delta_{ij}s_i$$

特異値は非負。急速に減衰する場合は、行列の低ランク近似が可能

行列積表現 (matrix product state) / tensor train



分解は特異値分解などで実行可能。最悪の場合、**virtual bond**の大きさは、指数的に大きい。

→ 多くの物理系、データにおいて、小さな次元ですむことがある。

例: 1次元量子系 離れた場所の間での「量子エンタングルエントロピー」が小さい

ITensors.jl

Flatiron instituteで開発が進められている

M. Fishman, S. R. White, E. Miles
Stoudenmire, SciPost Phys.
Codebases 4 (2022)

- MPSからより汎用的なテンソルネットワークまで
- 階層構造
 - バックエンド: NDTensors.jl (型安定性を保った高速なライブラリ)
 - MPSコード: 動的で柔軟なコード
- GPU対応 (multi GPU?)

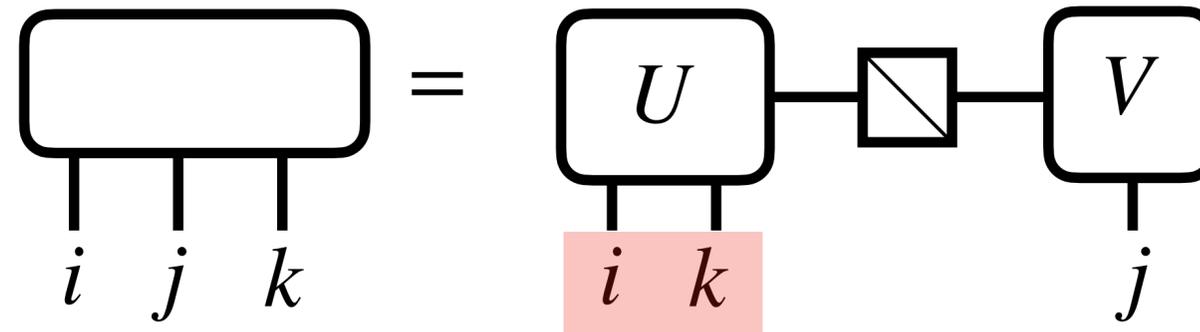
```
i = Index(3, "i")
j = Index(4, "j")
k = Index(5, "k")

T = randomITensor(i, j, k)

U, S, V = svd(T, (i, k))

@show norm(U*S*V-T)
```

$$A = U\Sigma V^\dagger$$



<https://itensor.github.io/ITensors.jl/stable/examples/ITensor.html>

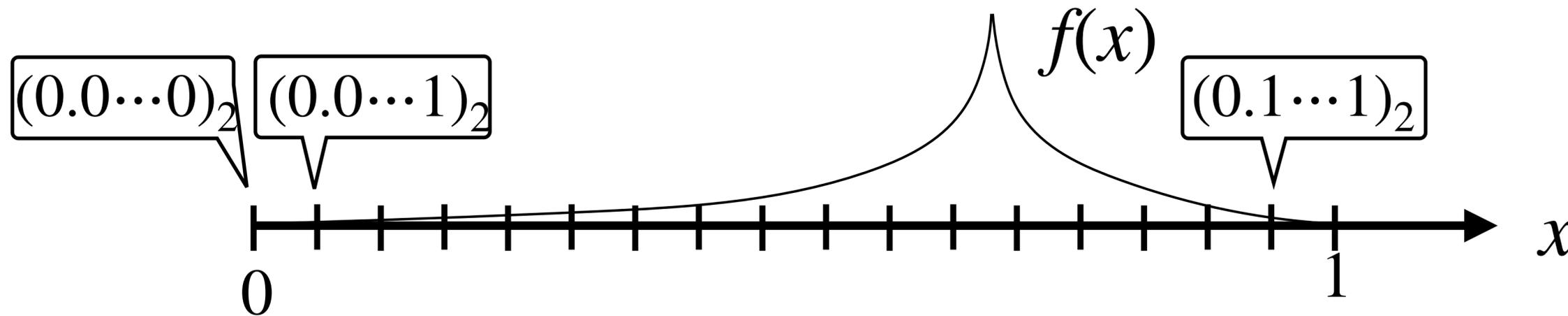
QTTのまとめ (1/3ページ)

I. V. Oseledets, Doklady Math. **80**, 653 (2009)

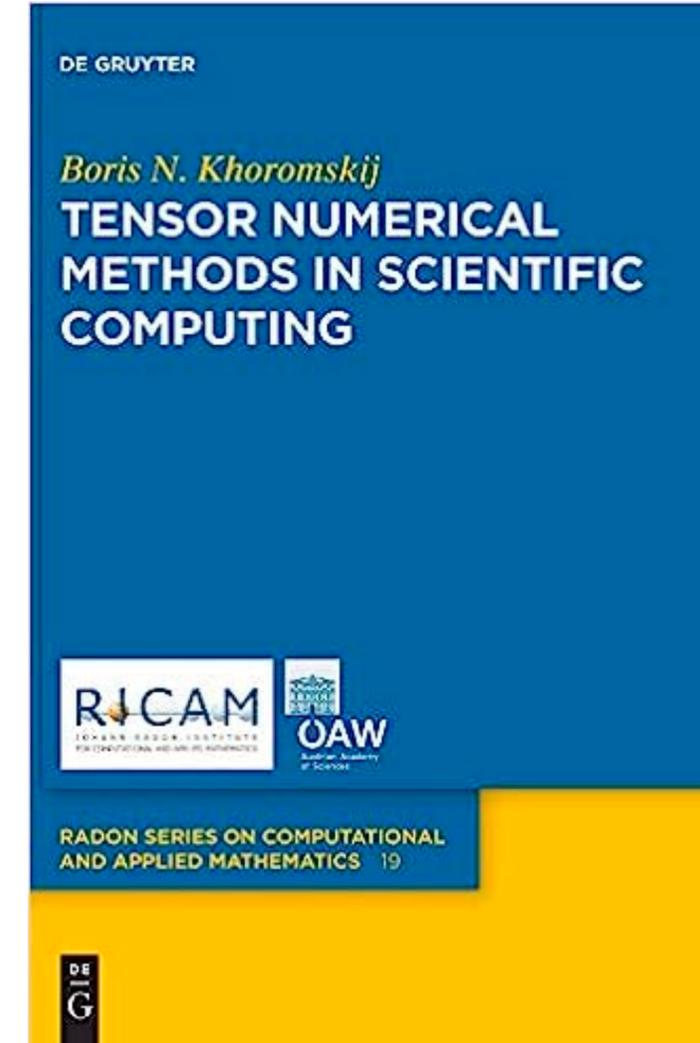
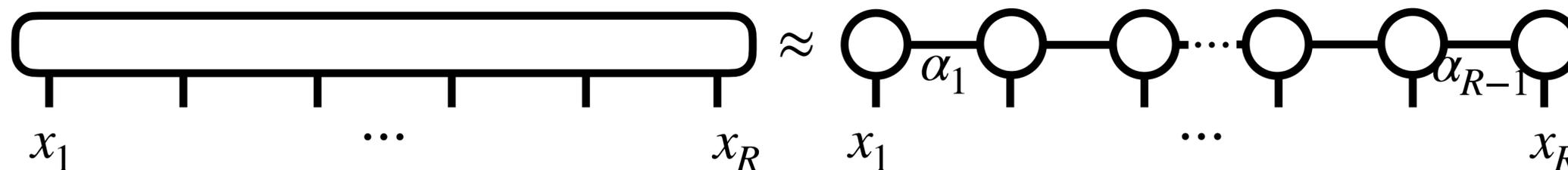
B. N. Khoromskij, Constr. Approx. **34**, 257 (2011)

Scalar-valued function $f(x)$ with $0 \leq x < 1$

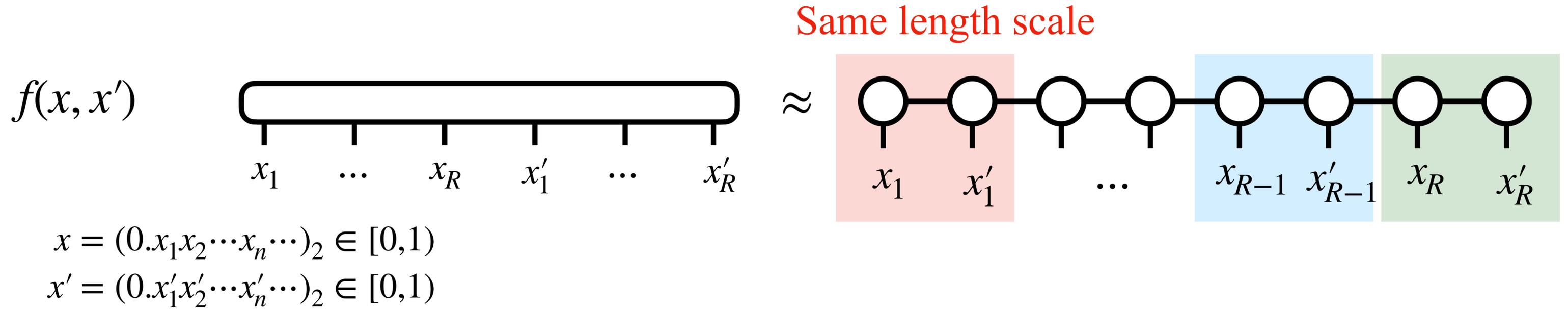
Binary coding: $x = (0.x_1x_2 \cdots x_R)_2 = x_1/2 + x_2/2^2 + \cdots + x_R/2^R$



$$f(x_1, \dots, x_R) \approx \sum_{\alpha_1=1}^{D_1} \cdots \sum_{\alpha_{R-1}=1}^{D_{R-1}} F_{x_1, 1\alpha_1}^{(1)} F_{x_1, \alpha_1\alpha_2}^{(2)} \cdots F_{x_R, \alpha_{R-1}1}^{(R)}$$



QTTのまとめ (2/3ページ)



同程度の長さスケールの変数が、強くエンタングルしている場合

QTTのまとめ (3/3ページ)

指数関数

$$f(x) = e^{-x} = e^{-x_1/2} e^{-x_2/2^2} \dots e^{-x_n/2^n} \dots \quad D = 1$$
$$x = (0.x_1x_2 \dots x_n \dots)_2 \in [0,1)$$

The Sum of N exponential functions can be represented as a QTT of rank at most N .
∴ Bond dimensions are added when MPSs are added.

多項式

$$D \leq 1 + p$$

M. Ali and A. Nouy, “A. Approximation Theory of Tree Tensor Networks: Tensorized Univariate Functions”, Constr Approx (2023)

単位行列

$$f(x, y) = \delta_{x,y} = \delta_{x_1,y_1} \delta_{x_2,y_2} \dots \quad D = 1$$

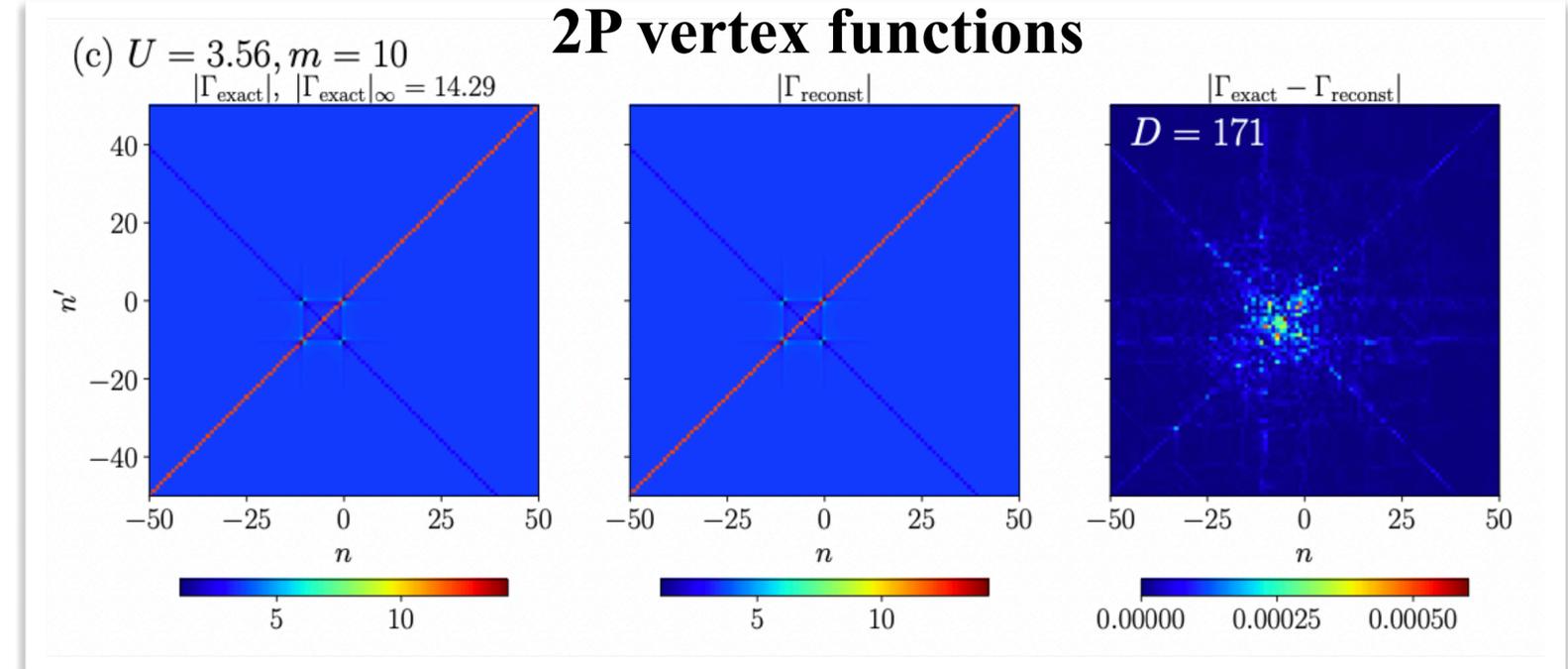
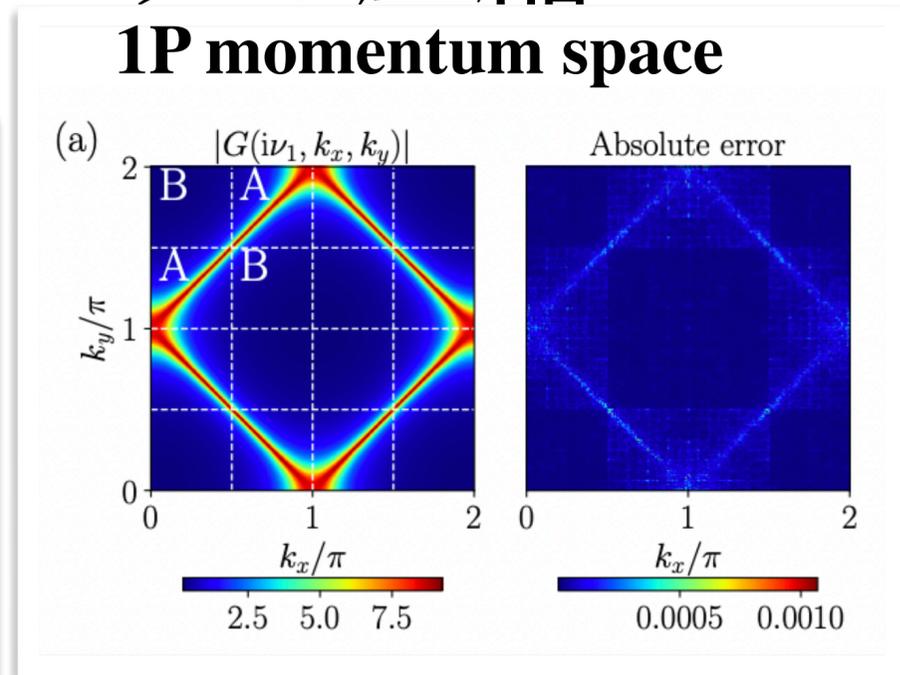
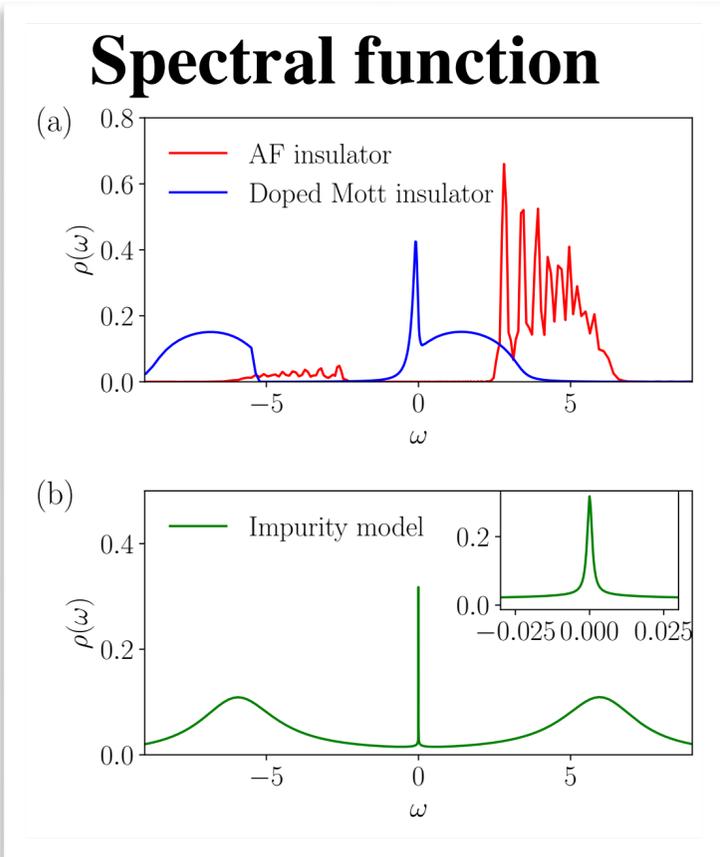
The diagram illustrates the Kronecker delta function $\delta_{x,y}$ as a tensor product of 1D Kronecker delta functions. On the left, a square matrix with a red diagonal line and '0' in the off-diagonal positions represents the 2D Kronecker delta. This is equated to a sequence of 1D Kronecker delta functions (represented by smaller squares with red diagonals) connected by tensor products (\otimes).

Representation of Continuous Functions: <https://tensornetwork.org/functions/>

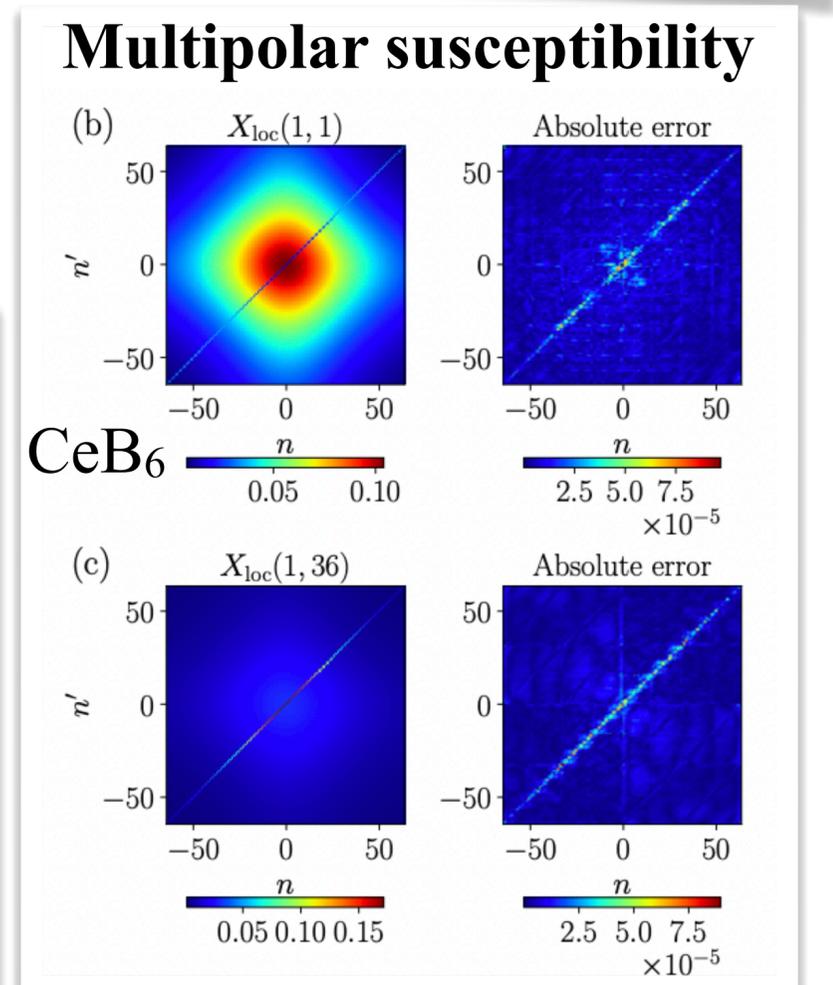
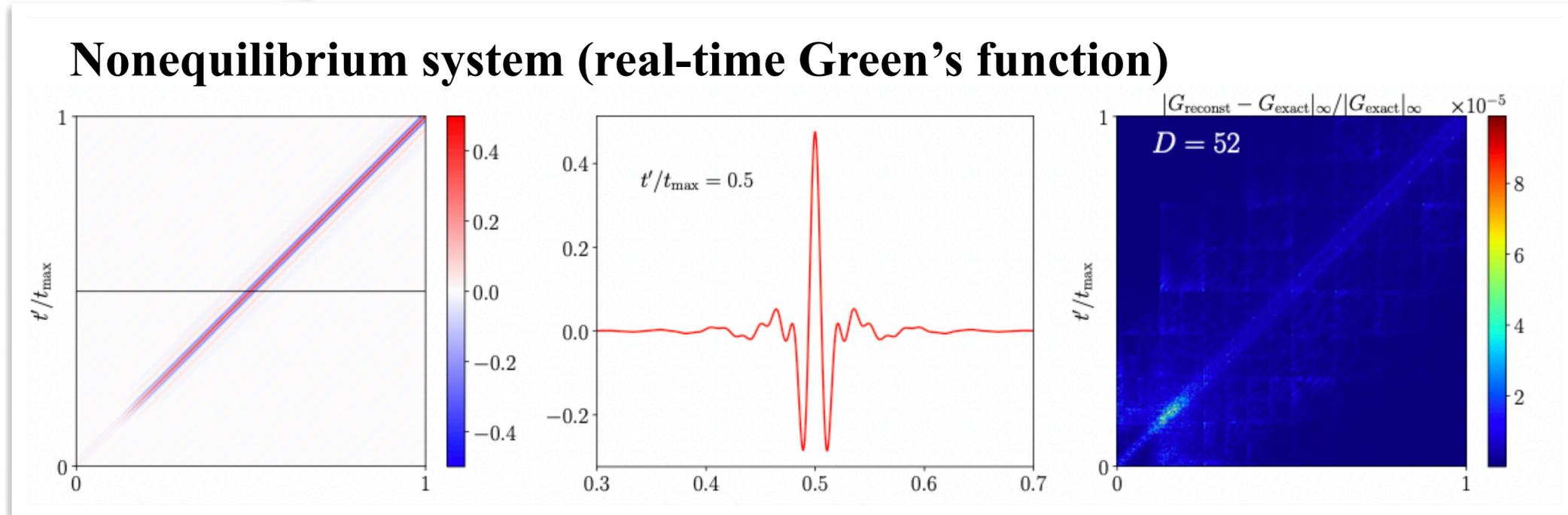
Written by Miles Stoudenmire

場の量子論データの圧縮

HS *et al.*, Physical Review X **13**, 021015 (2023)



既存のデータを特異値分解で圧縮してみた
多くの時空データが数桁に渡り圧縮可能



圧縮したまま(場の量子論)計算ができるか？

R を増やすことで、指数的な解像度向上 (計算量は R に線形)

フーリエ変換

$$F(r) = \int dk \hat{F}(k) e^{ikr}$$

量子フーリエ変換回路 = 古典コンピュータで効率的に実行可能

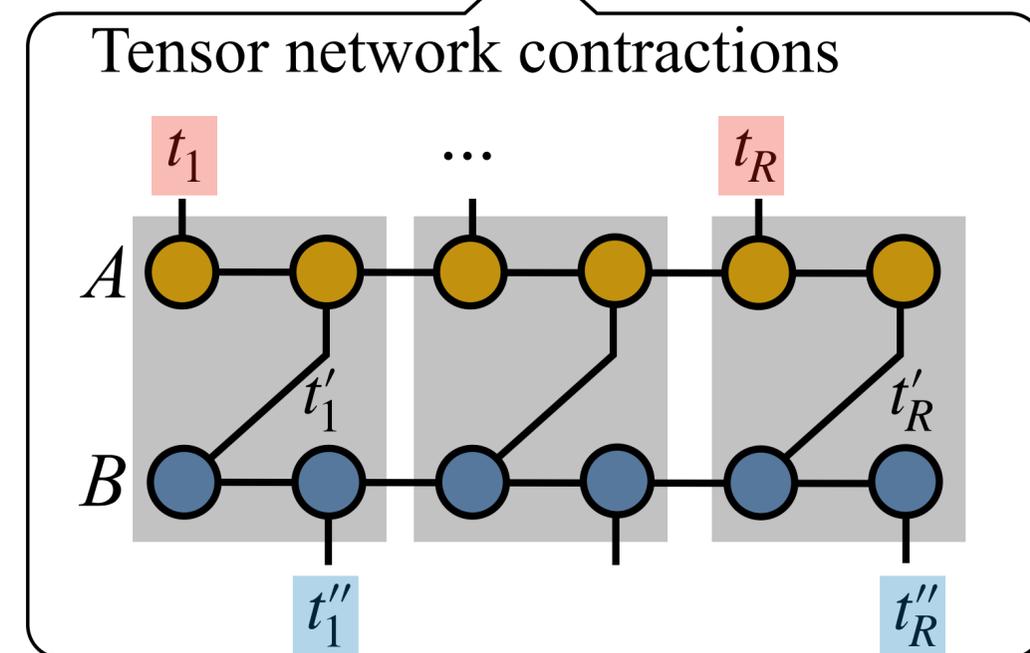
K. J. Woolfe *et al.*, Quantum Inf. Comput. **17**, 1 (2017), J. Chen *et al.*, arXiv:2210.08468v1

行列積

$$C(t, t'') = \int dt' A(t, t') B(t', t'')$$

$$C(t_1, t_1'', \dots, t_R, t_R'') = \sum_{t'_1, \dots, t'_R} A(t_1, t'_1, \dots, t_R, t'_R) B(t'_1, t_1'', \dots, t'_R, t_R'')$$

ほぼ任意のダイアグラム計算が実行可能



Juliaライブラリの実装 MSSTA.jl (multiscale space-time ansatz)

まだ、実験的なライブラリです (○ユニットテスト、△ドキュメンテーション)

<https://github.com/shinaoka/MSSTA.jl>

- フーリエ変換などの演算を簡単に実装可能なライブラリ
- バックエンドとして、ITensors.jlを利用

Juliaをなぜ選んだのか？

- forループの使用が避けられない
- 戦略的にJuliaで場の量子論計算のエコシステムを作る
- (ITensors.jlの開発者と交流があった)

VS Codeを使ってデモしてみます！

QTTの展望

- 分散並列計算、GPU対応
- より複雑なテンソルネットワークへの拡張
- 皆さんの周りに、桁違いの長さスケールが共存する難しい問題ありませんか？
QTTで非線型常微分方程式も解けます Cahn-Hilliard方程式?
- Quantics Tensor Cross Interpolation (TCI): 特異値分解の代替
M. K. Ritter, Y. N. Fernández, M. Wallerberger, J. von Delft, **HS**, X. Waintal, arXiv:2303.11819

宣伝: ICIAM2023



Quantics表現の宣伝予定！

[00789] Algorithmic advances in computational quantum mechanics

- **Session Date & Time :**
 - 00789 (1/3) : 3C (Aug.23, 13:20-15:00)
 - 00789 (2/3) : 3D (Aug.23, 15:30-17:10)
 - 00789 (3/3) : 3E (Aug.23, 17:40-19:20)
- **Type :** Proposal of Minisymposium
- **Abstract :** Chemistry, physics, and materials science have benefited tremendously from advances in algorithmic tools for the simulation of quantum systems. In recent years, ideas developed in collaboration with the applied mathematics community have played an increasingly prominent role. This minisymposium will focus on recent algorithmic advances in computational quantum mechanics driven by numerical linear algebra, numerical methods for partial differential equations and integral equations, fast algorithms for the manipulation of structured operators, convex optimization, tensor networks, randomized algorithms, and machine learning methods.
- **Organizer(s) :** Jason Kaye, Michael Lindsey
- **Classification :** 81-08, 65Z05, Computational Quantum Physics
- **Speakers Info :**
 - Jason Kaye (Flatiron Institute, Simons Foundation)
 - Chao Yang (Lawrence Berkeley National Laboratory)
 - Hiroshi Shinaoka (Saitama University)
 - Fabian Faulstich (University of California, Berkeley)
 - Yuehaw Khoo (The University of Chicago)
 - Eloise Letournel (INRIA Paris)
 - Kangbo Li (Cornell University)
 - Miles Stoudenmire (Flatiron Institute, Simons Foundation)

スパースモデリング 疑似多倍精度による積分方程式の解法

「虚時間」に特化した最適な基底の構成

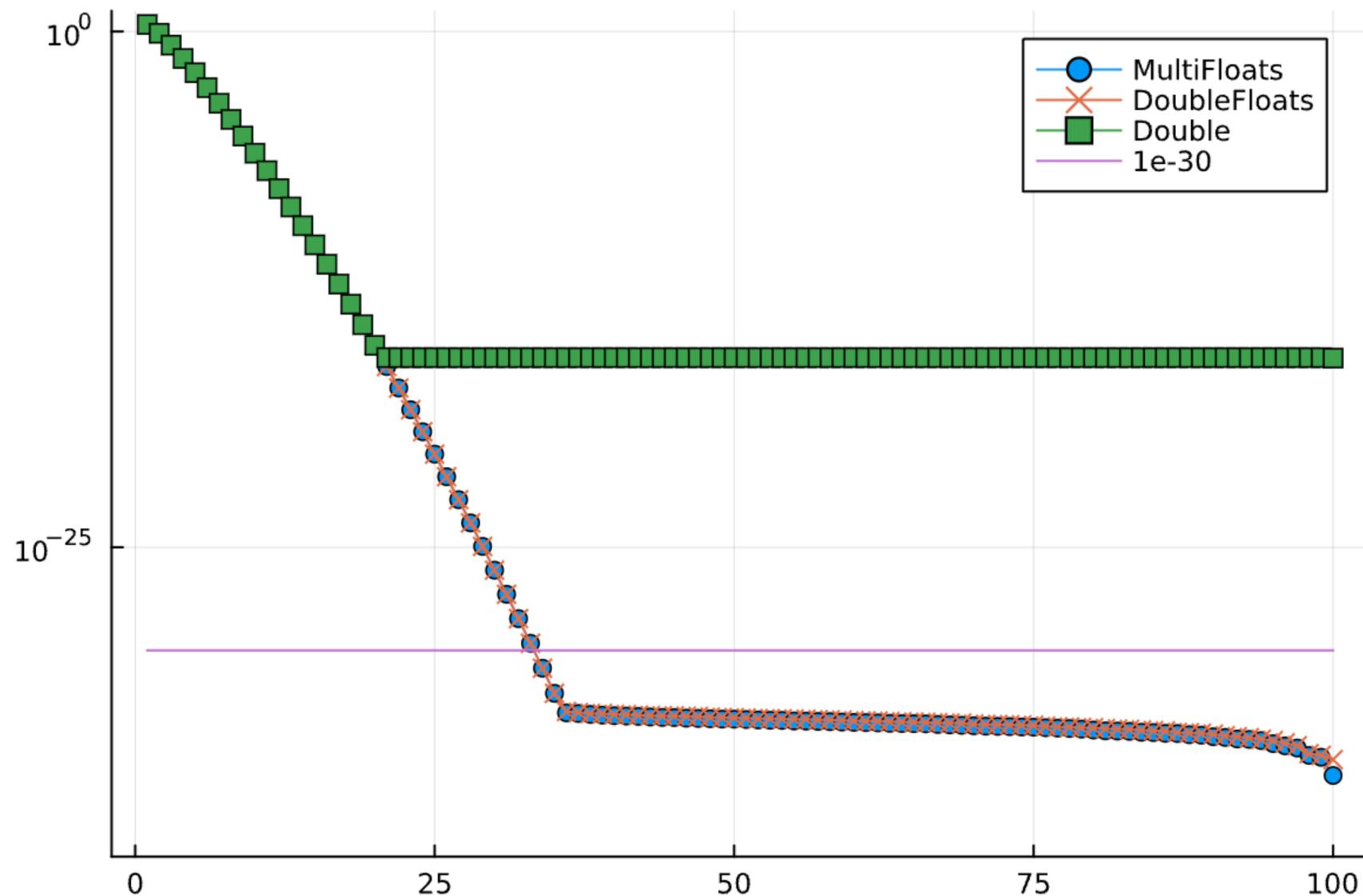
MultiFloats.jl vs DoubleFloats.jl

ヒルベルト行列

$$H_{ij} = 1/(i + j - 1) \quad \begin{bmatrix} 1 & 1/2 & 1/3 \\ 1/2 & 1/3 & 1/4 \\ 1/3 & 1/4 & 1/5 \end{bmatrix}$$

共に疑似四倍精度浮動小数点演算を実装

- SVDは、MultiFloats.jlの方が数割速い
- MultiFloats.jlは、四倍精度より高次の実装を含む



MultiFloats.jl 39 ms

```
n = 100
Hmf = [1/Float64x2(i+j-1) for i=1:n, j=1:n] # MultiFloats
u_mf, s_mf, vt_mf = svd(Hmf)
```

DoubleFloats.jl 62 ms

```
Hdd = [1/Double64(i+j-1) for i=1:n, j=1:n] # DoubleFloats
u_dd, s_dd, vt_dd = svd(Hdd)
```

cf. Float64 0.5 ms 二桁の速度差

Julia v1.91, MultiFloats v1.0.3, DoubleFloats v1.2.4
on Apple M1 Pro (1 thread)

虚時間形式の相関関数とは？

相関関数

$\langle A(r, t)B(r', t') \rangle$ 違う時空点間での物理量の相関を測る。
久保公式によって、物理系の外場応答と対応。

長時間極限 $|t - t'|$ で速く減衰しない。→数値的に圧縮しづらい

虚時間形式 (松原形式とも呼ぶ) 数値的なダイアグラム計算でよく用いられる。

$t = i\tau$ 時間を仮想的に純虚数に取る

長時間極限 $|\tau - \tau'|$ で指数的に減衰する。→数値的に圧縮し易い (が、情報も一部失われている)

“スパースモデリング” = 虚時間での最適基底の構成

虚時間形式の相関関数における情報損失

$$\underline{G(\tau)} = - \langle T_\tau A(\tau) B(0) \rangle = \int_{-\infty}^{\infty} d\omega K(\tau, \omega) \underline{\rho(\omega)}$$

虚時間の量

実周波数(時間)の量

系に寄らない積分核

$$K(\tau, \omega) = \frac{e^{-\tau\omega}}{1 + e^{-\beta\omega}}$$

高周波数の情報を τ に伝搬しにくい

フェルミオンの場合 $\beta = 1/T$

虚時間の相関関数は、実時間の相関関数に比べて、高周波数の情報が落ちている。

→ 積分核を使って、虚時間の最適な正規直交基底を作れるか？

積分核の特異分解による基底の構成

HS, J. Otsuki, M. Ohzeki, K. Yoshimi, PRB **96**, 035147 (2017)

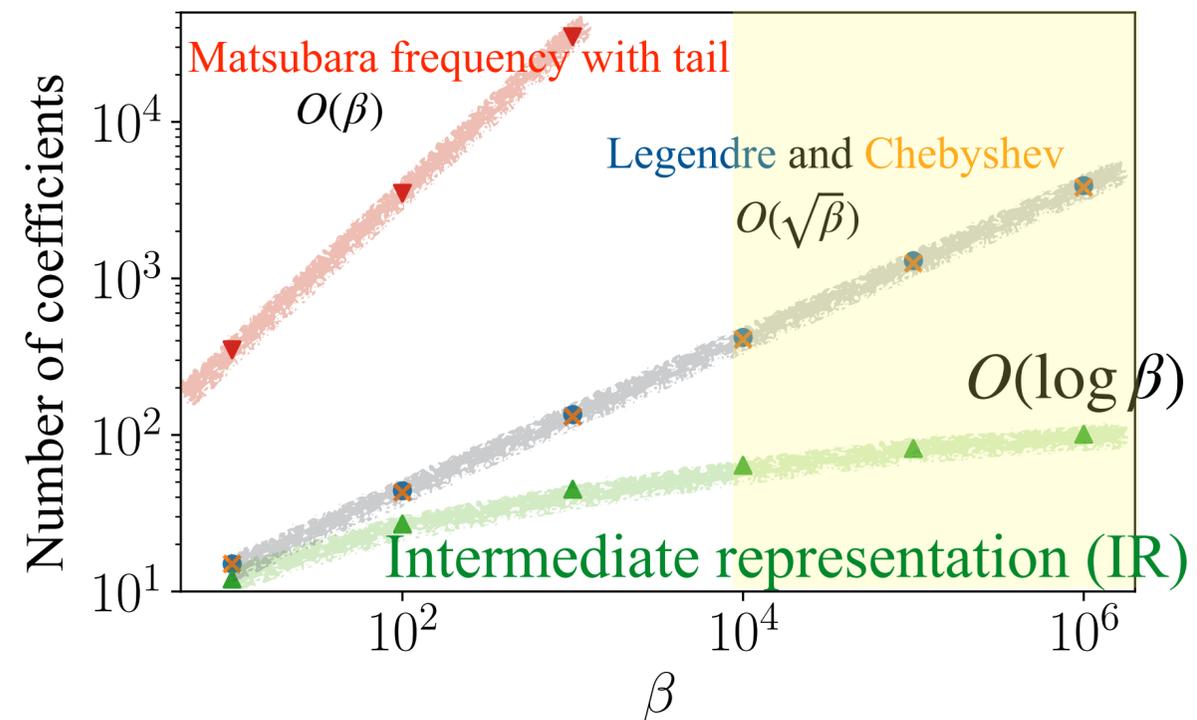
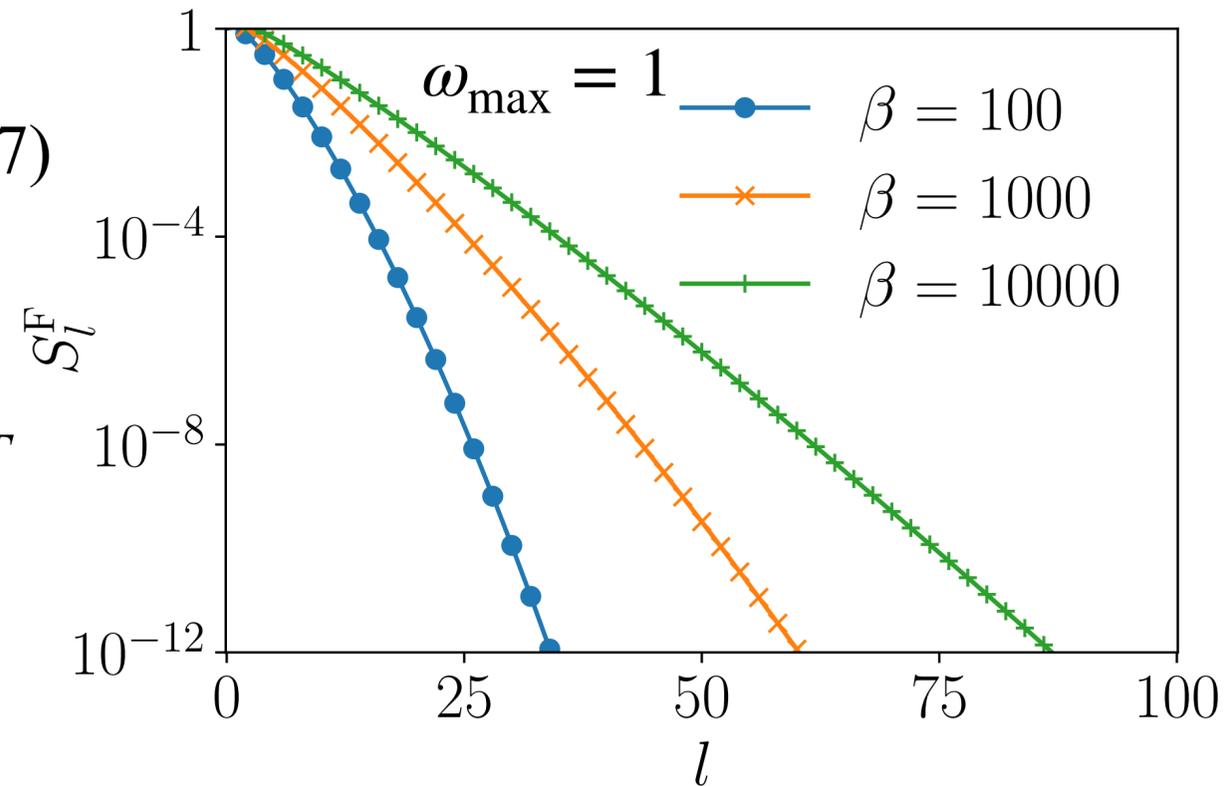
$$K(\tau, \omega) = \sum_{l=0}^{\infty} S_l \underline{U_l(\tau) V_l(\omega)} \quad \text{for} \quad |\omega| < \omega_{\max}, 0 < \tau < \beta = 1/T$$

Orthonormal *intermediate-representation* bases

$$G(\tau) = \sum_{l=0} G_l U_l(\tau) \quad \rho(\omega) = \sum_{l=0} \rho_l V_l(\omega)$$

$$G_l = -S_l \rho_l$$

- Dimensionless parameter $\Lambda \equiv \beta \omega_{\max}$
- $|G_l|$ decay as fast as S_l ($\because |\rho_l|$ is bounded).
- Logarithmic growth of $S_l > \epsilon$ w.r.t Λ



小さな特異値に対応する特異ベクトルを倍精度で数値的に決めたい

応用の拡がり

量子多体計算から第一原理計算まで

Analytic continuation

Sparse modeling (SpM)

- J. Otsuki, M. Ohzeki, **HS**, K. Yoshimi, PRE **95**, 061302(R) (2017)
- [Application to lattice QCD] E. Itou and Y. Nagai, JHEP **07**, 007 (2020)

Nevalinna/Carathéodory formalism

- J. Fei, C.-N. Yeh, E. Gull, PRL **126**, 056402 (2021)
- J. Fei, C.-N. Yeh, D. Zgid, and E. Gull, PRB **104**, 165111 (2021)

All electron GW calculations

- K. Haule, S. Mandal, arXiv:2008.07727

FLEX

- N. Witt, E. G. C. P. van Loon, T. Nomoto, R. Arita, T. O. Wehling, PRB **103**, 205148 (2021)
- N. Witt, J. M. Pizarro, T. Nomoto, R. Arita, T. O. Wehling, arXiv:2108.01121v1

Estimation of magnetic interactions

- T. Nomoto, T. Koretsune and R. Arita, PRB **102**, 014444 (2020)
- T. Nomoto, T. Koretsune and R. Arita, PRL **125**, 117204 (2020)
- Y. Nomura, T. Nomoto, M. Hirayama and R. Arita, PRR **2**, 043144 (2020)

Migdal-Eliashberg theory

- T. Wang, T. Nomoto, Y. Nomura, **HS**, J. Otsuki, T. Koretsune, and R. Arita, PRB **102**, 134503 (2020)

Self-energy embedding calculations

- S. Isakov, C.-N. Yeh, E. Gull, and D. Zgid, PRB **102**, 085105 (2020)
- C. N. Yeh, S. Isakov, D. Zgid, and E. Gull, PRB **103**, 195149 (2021)

Extension to the two-particle level

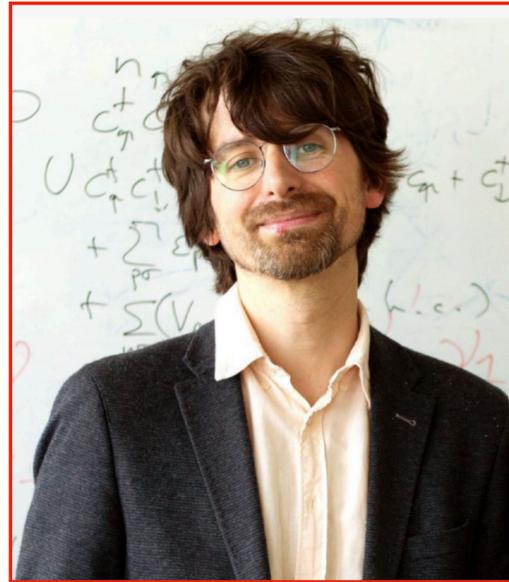
- **HS**, J. Otsuki, M. Ohzeki, K. Yoshimi, K. Haule, M. Wallerberger, E. Gull, PRB **97**, 205111 (2018)
- **HS**, D. Geffroy, M. Wallerberger, J. Otsuki, K. Yoshimi, E. Gull, J. Kuneš, SciPost Phys. **8**, 012 (2020)
- M. Wallerberger*, **HS***, A. Kauch, PRR **3**, 033168 (2021)

独自実装や、Pythonライブラリの使用も含みます

SparseIR.jl

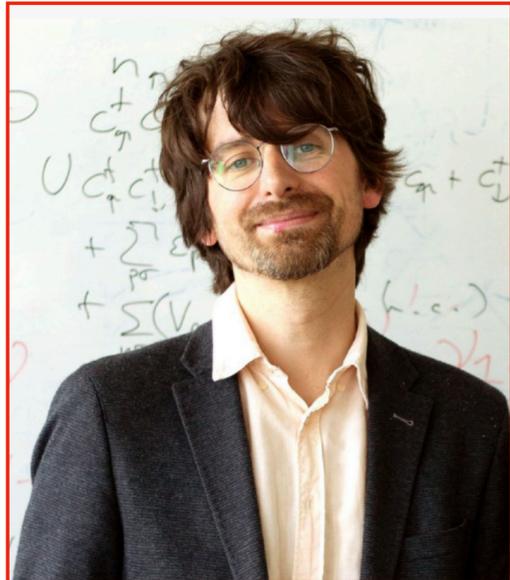
M. Wallerberger, S. Badr, S. Hoshino, S. Huber, F. Kakizawa, T. Koretsune, Y. Nagai, K. Nogaki, T. Nomoto, H. Mori, J. Otsuki, S. Ozaki, T. Plaikner, R. Sakurai, C. Vogel, N. Witt, K. Yoshimi, H. Shinaoka, SoftwareX **21**, 101266 (2023)

<https://github.com/SpM-lab/SparseIR.jl>



- 基底関数の区分多項式 (Legendre)による展開: 特異分解→特異値分解
- 疑似四倍精度演算による特異値分解 MultiFloats.jl + GenericLinearAlgebra.jl
- 基底の構成が終われば、後は倍精度演算で十分

Python実装 (sparse-ir) との比較



Pure Python 実装: <https://github.com/SpM-lab/sparse-ir>

numpyの疑似四倍精度浮動小数点拡張 <https://github.com/tuwien-cms/xprec>

C言語 + Python

M. Wallerberger, S. Badr, S. Hoshino, S. Huber, F. Kakizawa, T. Koretsune, Y. Nagai, K. Nogaki, T. Nomoto, H. Mori, J. Otsuki, S. Ozaki, T. Plaikner, R. Sakurai, C. Vogel, N. Witt, K. Yoshimi, H. Shinaoka, SoftwareX **21**, 101266 (2023)

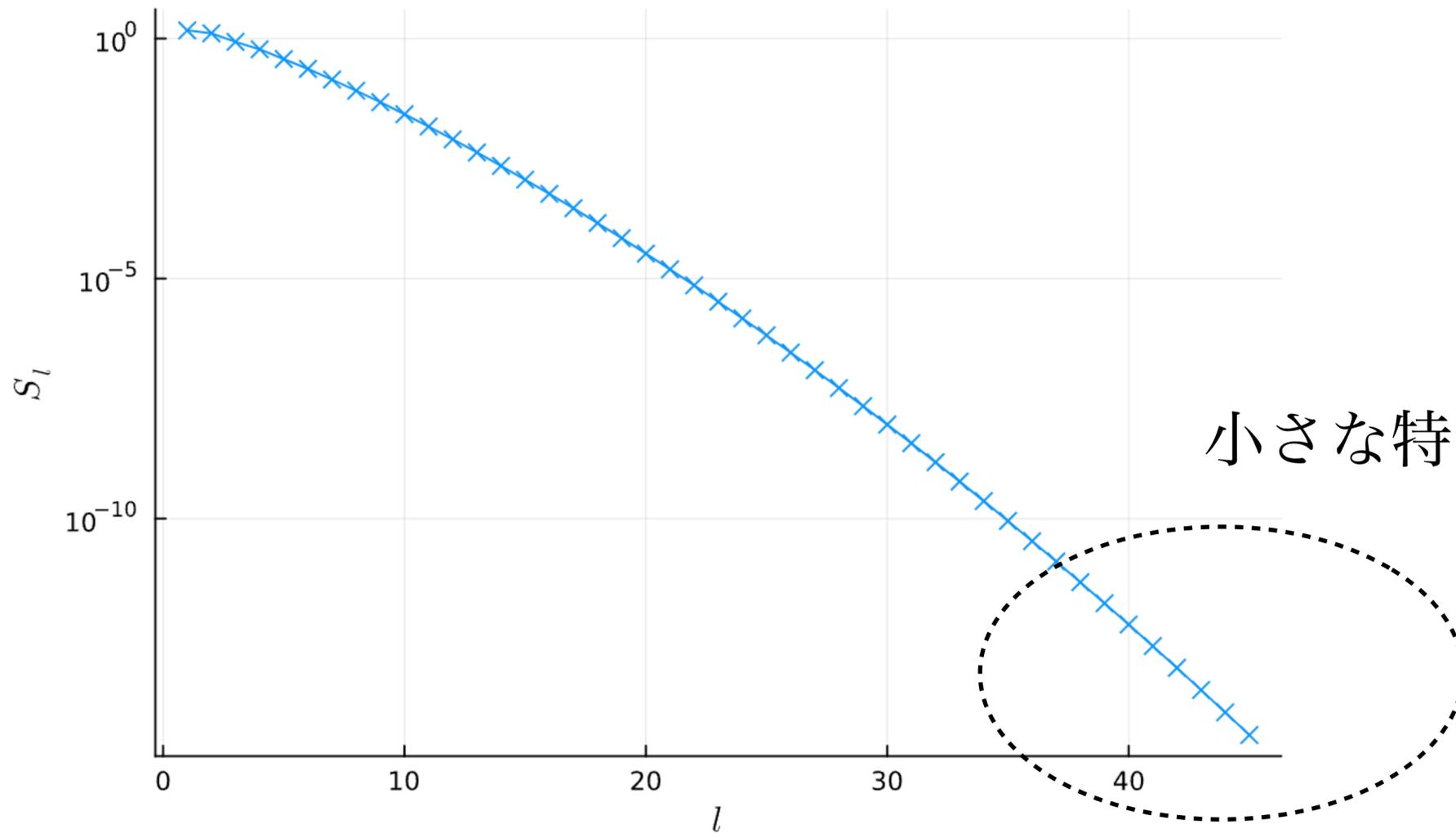
パッケージのリリース時に、バイナリーの自動ビルド+登録

- condaとPyPIを両方メンテナンスする必要
- ビルドに使うGitHub Actionsのランナーとcondaの互換性 (glibcのバージョンなど)
- 普通の物理学者、学生にはメンテナンスが難しい...

SparseIR.jlのデモンストレーション

```
▷  
1 using SparseIR  
2 using Plots  
3 using LaTeXStrings  
4  
5 # 基底を作成  
6 beta = 15.0  
7 wmax = 10.0  
8 basis = FiniteTempBasis(Fermionic(), beta, wmax, 1e-15)  
9  
10 plot(basis.s, title="", xlabel=L"l", ylabel=L"S_l", label="", yaxis=:log, marker=:x)  
[1] ✓ 5.3s
```

...

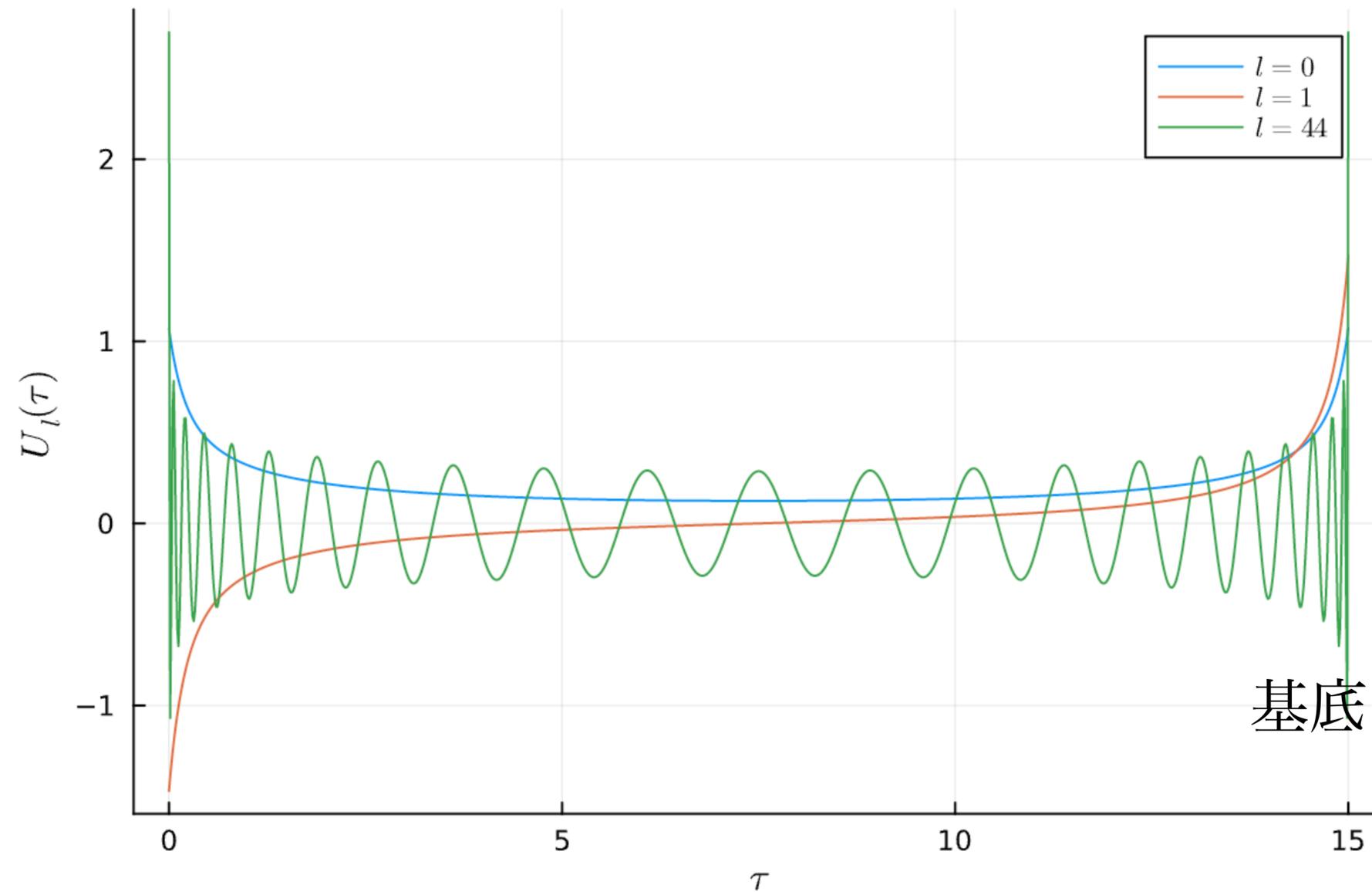


小さな特異値まで精度良く計算

SparseIR.jlのデモンストレーション

```
1 p = plot(title="", xlabel=L"\tau", ylabel=L"U_l(\tau)", label="")
2 ts = collect(LinRange(0, beta, 1000))
3 for l in [0, 1, length(basis.u)-1]
4     plot!(p, ts, basis.u[l+1](ts), label=L"l=%$(l)")
5 end
6 p
```

✓ 0.4s



基底関数の精度は大体10桁

チュートリアル集

- 典型的な計算のサンプルコード
- Jupyter Book
- 手元でのビルド環境は、VS Code + Dockerで用意

<https://spm-lab.github.io/sparse-ir-tutorial/>

sparse-ir

Basic theory

- Intermediate representation (IR)
- Sparse sampling
- Discrete Lehmann representation

Reference

- API reference
- Additional material

Sample codes

- Transformation from/to IR
- Sparse sampling
- Discrete Lehmann Representation
- Second-order perturbation
- The GF2 & GW method
- [DMFT with IPT solver](#)
- FLEX approximation
- TPSC approximation
- Eliashberg theory for Holstein-Hubbard model
- Orbital magnetic susceptibility
- Estimation of exchange interactions

Welcome to sparse-ir tutorial

This online document provides comprehensive tutorials of the Python library [sparse-ir](#) library as well as its port to [Julia](#) and [Fortran](#).

You can run jupyter notebooks online on [mybinder!](#)

Please free to make a contact with the authors of each tutorial page if you have any questions!

Contributors

- Markus Wallerberger (TU Wien)
- Samuel Badr (TU Wien)
- Shintaro Hoshino (Saitama Univ.)
- Fumiya Kakizawa (Saitama Univ.)
- Takashi Koretsune (Tohoku Univ.)
- Yuki Nagai (JAEA, Riken)
- Kosuke Nogaki (Kyoto Univ.)
- Takuya Nomoto (Univ. Tokyo)
- Hitoshi Mori (Riken)
- Junya Otsuki (Okayama Univ.)
- Soshun Ozaki (Univ. Tokyo)
- Rihito Sakurai (Saitama Univ.)
- Constanze Vogel (TU Wien)
- Niklas Witt (Univ. Hamburg)
- Kazuyoshi Yoshimi (ISSP)
- Hiroshi Shinaoka (Saitama Univ.)

解説 + コード例
by 永井さん

The screenshot shows a web browser window displaying a JupyterLab notebook. The browser's address bar shows 'spm-lab.github.io'. The notebook's sidebar on the left contains a table of contents for a project named 'sparse-ir'. The main content area displays the title 'Bogoliubov-de Gennes equations in real space' by Yuki Nagai. A warning message states: 'This notebook is expensive. Running this notebook takes a few minutes.' The text begins with 'We consider a superconductivity in a mean-field level.' and then introduces the 'Basic theory of superconductivity in real space'. It defines a Hamiltonian $H = \psi^\dagger \hat{\mathcal{H}} \psi / 2$ and describes the column vector ψ and row vector ψ^\dagger in terms of fermionic annihilation and creation operators. The 'Hamiltonian' matrix $\hat{\mathcal{H}}$ is defined as a $2N \times 2N$ Hermitian matrix.

sparse-ir

- Basic theory
- Intermediate representation (IR)
- Sparse sampling
- Discrete Lehmann representation
- Reference
 - API reference
 - Additional material
- Sample codes
 - Transformation from/to IR
 - Sparse sampling
 - Discrete Lehmann Representation
 - Second-order perturbation
 - The GF2 & GW method
 - DMFT with IPT solver
 - FLEX approximation
 - TPSC approximation
 - Eliashberg theory for Holstein-Hubbard model
 - Orbital magnetic susceptibility
 - Estimation of exchange interactions

Bogoliubov-de Gennes equations in real space

Author: [Yuki Nagai](#)

This notebook is expensive. Running this notebook takes a few minutes.

We consider a superconductivity in a mean-field level.

Basic theory of superconductivity in real space

Let us consider a Hamiltonian for a fermion system given as $H = \psi^\dagger \hat{\mathcal{H}} \psi / 2$. The column vector ψ is composed of N fermionic annihilation c_i and creation operators c_i^\dagger ($i = 1, 2, \dots, N$), $\psi = (\{c_i\}^T, \{c_i^\dagger\}^T)$, where $\{c_i\} = (c_1, c_2, \dots, c_N)^T$ and $\{c_i^\dagger\} = (c_1^\dagger, c_2^\dagger, \dots, c_N^\dagger)$. The row vector ψ^\dagger is also defined as $\psi^\dagger = (\{c_i^\dagger\}^T, \{c_i\}^T)$. The subscription i in c_i or c_i^\dagger indicates a quantum index depending on spatial site, spin, orbital, etc. The "Hamiltonian" matrix $\hat{\mathcal{H}}$ is a $2N \times 2N$ Hermitian matrix given as

$$\hat{\mathcal{H}} = \begin{pmatrix} \hat{H} & \hat{\Delta} \\ \hat{\Delta}^\dagger & -\hat{H}^* \end{pmatrix},$$

where \hat{H} is a normal state Hamiltonian and $\hat{\Delta}$ is a superconducting order parameter.

The BdG equations are regarded as the eigenvalue equation with respect to $\hat{\mathcal{H}}$ expressed as

$$\hat{\mathcal{H}} \vec{f}_\gamma = \epsilon_\gamma \vec{f}_\gamma$$
$$\vec{f}_\gamma = \begin{pmatrix} \vec{u}_\gamma \\ \vec{v}_\gamma \end{pmatrix}$$

最後に: Juliaによる計算物理の展望

Juliaを使うことで、試行錯誤のスピードを上げることが出来た！

今後挑戦したい

- テンソルネットワークの分散並列計算 Distributed vs MPI
- Multi GPUs

研究員募集中!

- 学術変革領域研究B「量子古典融合アルゴリズムが拓く計算物質科学」(領域代表品岡: 2023~2025年)
- JST創発「2粒子レベルの量子埋め込み理論に基づく新規第一原理計算手法の開発と実証」(2024~2026年: フェーズ1、2027~2030年: フェーズ2)
- 3年任期 **Juliaを使った時空圧縮技術の開発と応用、物性物理外からの挑戦も歓迎!**
- 特任助教の名称の付与可
- 助教級の待遇 (1年目: 480~530万円、2年目: 530~580万円)
- 別途、家賃補助、通勤手当あり