Manual

TM Package for Exact Diagonalization

Takahiro Misawa

Department of Applied Physics, University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-0033

September 3, 2015

Contents

1
How to use TM Pack for multi-orbital Hubbard model

1.1 Definition files (概要)

TM Pacage を使うためには、いくつかの input file (*def) を用意する必要があります。どう用意するかを以下に説明します。Table 1.1 に用意する input file のリストを示します。input file は以下の 4 つの種類に分類されます。

(1) List:
EDnamelist.def に使用する input file の名前のリストを書きます。順番が重要なことに注意してください。

(2) Basic parameters:
EDmodpara.def, に対角化を実行する基本的なパラメーター (サイトの数、電子数、Lanczos ステップを何回やるかなど) を設定します。

(3) Hamiltonian:
zlocspn.def: 局在スピンの位置を指定しますが、ハバード模型では使用していません。mVMC との整合性を保つために存在しています。

ztransfer.def: ハミルトニアンの一体部分（トランスファー）を指定します。今のところ、スピンの S_z 成分を保存するトランスファーのみ対応しています。

zcoulombintra.def, **zcoulombinter.def**, **zhund.def**, **zpairhop.def**, **zexchange.def** は名前からわかると思いますが、それぞれ、On-site のクーロン相互作用、Off-site のクーロン相互作用、Hund coupling, Pair hopping, exchange coupling を指定します。それぞれの相互作用の定義はあとで述べます。

(4) Output:
zcisajs.def :出力する Green 関数を指定します。$\langle c^\dagger_{i\sigma} c_{j\sigma} \rangle$ が出力されます。

zcisajscktaltdc.def : 出力する 2 体項を指定します。$\langle c^\dagger_{i\sigma} c_{j\sigma} c^\dagger_{k\tau} c_{l\tau} \rangle$ が出力されます。

Name	Details
EDnamelist.def	List of all the definitions files.
EDmodpara.def	Setting: Parameter for Exact Diagonalization
zlocspn.def	Hamiltonian: Configurations of the local spins
ztransfer.def	Hamiltonian: Transfer and chemical potential
zcoulombintra.def	Hamiltonian: On-site Coulomb interactions
zcoulombinter.def	Hamiltonian: Off-site Coulomb interactions
zhund.def	Hamiltonian: Hund couplings
zpairhop.def	Hamiltonian: Pair-hopping terms
zexchange.def	Hamiltonian: Exchange interactions
zcisajs.def	Output : Green functions $\langle c^\dagger_{i\sigma} c_{j\sigma} \rangle$
zcisajscktaltdc.def	Output : Correlation functions $\langle c^\dagger_{i\sigma} c_{j\sigma} c^\dagger_{k\tau} c_{l\tau} \rangle$

Table 1.1: List of the definition files.

1.2 List file (EDnamelist.def)

1.2: **EDnamelist.def** の例。**EDnamelist.def** には使用する definition file を順番通りに並べる必要があります。ファイルは順番だけで識別しているので、ファイルの名前はとくに意味がないことに注意して下さい [*1]。また、使用するファイルは実行体と同じディレクトリにある必要があります。もし、あるファイルが存在しない場合はエラーメッセージとともに停止します。

```
1  EDmodpara.def
2  zlocspn.def
3  ztransfer.def
4  zcoulombintra.def
5  zcoulombinter.def
6  zhund.def
7  zpairhop.def
8  zexchange.def
9  zcisajs.def
10 zcisajscktaltdc.def
```

Table 1.2: EDnamelist.def の例。一番左の数字は行番号なので、実際は入力しないことに注意。

[*1] たとえば、zcoulombinter.def という名前のファイルを 3 番目に書くと transfer を指定するファイルとして処理されてしまう。

1.3 Setting files (EDmodpara.def)

この節では、対角化のパラメーターを指定する **EDmodpara.def** の詳細を説明します。まず、4.1 に **EDmodpara.def** の例を示します。これは,16 サイト、5up+5down 電子数のランチョス法による対角化で基底状態のエネルギーを求める場合のサンプルファイルです。

1	==========	[xstring]	
2	Model_Parameters 0	[xstring]	[xint]
3	==========	[xstring]	
4	VMC_Cal_Parameters	[xstring]	
5	==========	[xstring]	
6	CDataFileHead zvo	[xstring]	[string01]
7	CParaFileHead zqp	[xstring]	[string02]
8	==========	[xstring]	
9	Nsite 16	[xstring]	[int01]
10	Nup 5	[xstring]	[int02]
11	Ndown 5	[xstring]	[int03]
12	Lanczos_max 1000	[xstring]	[int04]
13	initial_iv 202	[xstring]	[int05]
14	nvec 1	[xstring]	[int06]
15	exct 1	[xstring]	[int07]
16	LanczosEps 14	[xstring]	[int08]
17	LanczosTarget 2	[xstring]	[int09]
18	WRITE 0	[xstring]	[int10]
19	READ 0	[xstring]	[int11]

Table 1.3: EDmodpara.def の例。一番左の数字は行番号, 右の [xstring] などは後の説明ための変数なので、入力はしないことに注意。

以下が、**EDmodpara.def** の各パラメータの説明です。

- [xstring]: 任意の文字列。とくに使用しない。
- [xint]: 任意の整数。とくに使用しない。
- [string01]: アウトプットファイルのヘッダ。たとえば、一体の Green 関数は **string01_Lanczos_cisajs.dat.dat** といった名前のファイルに出力される。
- [string02]: とくに使用しない。mVMC との整合性を保つためにあるだけ。
- [int01]: サイト数を指定する整数。
- [int02]: アップスピンの電子数を指定する整数。
- [int03]: ダウンスピンの電子数を指定する整数。

- [int04]: ランチョスステップを行う回数。収束すればこれより短い回数で終わる。
- [int05]: 初期条件のベクトルでノンゼロの成分を指定する整数。
- [int06]: ランチョス法の段階で下からいくつの固有値を求めるかの数。
- [int07]: 1 なら基底状態のベクトルを求める。2 なら第一励起状態のベクトルを求める。[int06] >= [int07] とする必要があることに注意。
- [int08]: ランチョスの収束判定条件を指定する整数 (一つまえのステップの固有値との相対誤差が, $10^{-\text{int08}}$ 以下になったら収束したとする)。
- [int09]: 何番目の固有値でランチョスの収束判定を行うかを指定する整数。1 なら基底状態。2 なら第一励起状態。
- [int10]: list ベクトルを読み込むかどうか。1 なら読み込み、0 なら読み込まない。
- [int11]: list ベクトルを書き出すかかどうか。1 なら書き出し、0 なら書き出さない。

· WRITE と READ について説明しておきます[*2]。READ=0, WRITE=0 は普通に list ファイルを参照することなしに、計算が行われます。通常はこれを使ってください。

WRITE=1, READ=0 とすると list ファイルを出力して終了します (16site だと数 GB になるので注意して下さい)。そのあとで, READ=1, WRITE=0 とするとその list ファイルを読み込んで、計算が行われます。この際に参照する list ファイルは sz.c の中で直接指定します。READ=1,WRITE=1 は意味がないので使用しないでください。

[*2] あんまりスマートでないので要検討の部分。

1.4 Definition files for Hamiltonian

ここではハミルトニアンを指定するファイルについて説明します。

1.4.1 zlocspn.def

以下で、**zlocspn.def** の例を示します。最初の 5 行はヘッダで **xint** 以外はなにが書かれても問題ありません。**xint** が局在スピンの総数ですが、ハバード模型の場合は 0 です。**int01** でサイトの番号、**int02** で局在スピン (0) か遍歴電子 (1) かを指定します。ハバード模型の場合は当然、全て 1 です。

note: mVMC では局在スピン (0) を指定するとそのサイトの二重占有度が完全に排除されて、局在スピンとして取り扱われます。将来的には対角化もそうした方がよいかもしれません。対角化では局在スピンと遍歴電子が混在した実空間のリストを用意するのが面倒なので、ハバード（遍歴電子）模型、ハイゼンベルグ (局在スピン) 模型、近藤格子模型とそれぞれ独立にプログラムを作っています

1	==========	[xstring]
2	NLocalSpin 0	[xstring] [xint]
3	==========	[xstring]
4	==========	[xstring]
5	==========	[xstring]
6	0 1	[int01] [int02]
7	1 1	[int01] [int02]
8	2 1	[int01] [int02]
	(continue . . .)	

Table 1.4: zlocspn.def の例。最初の部分のみを表示している。

1.4.2 ztransfer.def

以下で、**ztransfer.def** の例を示します。

1	==========	[xstring]
2	NTransfer 128	[xstring] [xint]
3	==========	[xstring]
4	==========	[xstring]
5	==========	[xstring]
6	0 1 0 1.0	[int01] [int02] [int03] [double1]
7	0 3 0 1.0	[int01] [int02] [int03] [double1]
8	0 4 0 1.0	[int01] [int02] [int03] [double1]
	(continue . . .)	

Table 1.5: ztransfer.def の例。最初の部分のみを表示している。

ホッピングと一体部分をハミルトニアンに付け加えます。付け加える項は以下で与えられます。

$$H+ = -\sum_{i,j,\sigma} t_{ij} c^{\dagger}_{i\sigma} c_{j\sigma} \tag{1.1}$$

ここで、– が付いていることに注意してください [*3]。上の例でいえば、-1.0 のホッピングが付け加えられます。また、Hamiltonian がエルミートになるようにすること ($c^{\dagger}_{i\sigma}c_{j\sigma}$ と $c^{\dagger}_{j\sigma}c_{i\sigma}$ を両方いれる), アップスピンとダウンスピンの対称性を保つこと [*4][*5]($c^{\dagger}_{i\uparrow}c_{j\uparrow}$ と $c^{\dagger}_{i\downarrow}c_{j\downarrow}$ を両方いれる) 必要があることに注意。また、Sz を変える項 (たとえば、$c^{\dagger}_{i\uparrow}c_{j\downarrow}$) には対応していない [*6]。

最初の 5 行はヘッダで **xint** 以外はなにが書かれても問題ありません。**xint** がトランスファーの総数です。4×4 のハバード模型で最近接ホッピングのみの場合は 128 です。**int01** でサイトの番号 i、**int02** でサイトの番号 j、**int04** でスピンの番号、**double1** でホッピングの値（の符号が逆にしたもの）を指定します。

1.4.3 zcoulombintra.def

以下で、**zcoulombintra.def** の例を示します。

1	====================	[xstring]
2	NCoulombintra 16	[xstring] [xint]
3	====================	[xstring]
4	====================	[xstring]
5	====================	[xstring]
6	0 4.0	[int01] [double1]
7	1 4.0	[int01] [double1]
8	2 4.0	[int01] [double1]
	(continue . . .)	

Table 1.6: zcoulombintra.def の例。最初の部分のみを表示している。

オンサイトのクーロン相互作用の項をハミルトニアンに付け加えます。付け加える項は以下で与えられます。

$$H+ = \sum_{i} U_i c^{\dagger}_{i\uparrow} c_{i\uparrow} c^{\dagger}_{i\downarrow} c_{i\downarrow} \tag{1.2}$$

最初の 5 行はヘッダで **xint** 以外はなにが書かれても問題ありません。**xint** が On-site クーロン相互作用の総数です。**int01** でサイトの番号 i、**double1** でクーロン相互作用の値を指定します。

[*3] これは混乱のもとなのでやめたほうがいい？
[*4] これはじつは必要ない条件？要検討。
[*5] じつは内部 [EDTrans.c] では upspin (0) の場合のトランスファーのみを抜き出して、計算している。アップとダウンを等価に取り扱うの mVMC との整合性のため。
[*6] これは一般のスピン軌道相互作用を取り扱う場合には必要な項。将来的にはとりいれた方がよい。ただ Hilbert 空間に次元が大きくなってしまう。

1.4.4 zcoulombinter.def

以下で、**zcoulombinter.def** の例を示します。

1	====================	[xstring]
2	NCoulombinter 32	[xstring] [xint]
3	====================	[xstring]
4	====================	[xstring]
5	====================	[xstring]
6	0 1 1.5	[int01] [int02] [double1]
7	0 4 1.5	[int01] [int02] [double1]
8	1 2 1.5	[int01] [int02] [double1]
8	1 5 1.5	[int01] [int02] [double1]
	(continue . . .)	

Table 1.7: zcoulombinter.def の例。最初の部分のみを表示している。

オフサイトのクーロン相互作用の項をハミルトニアンに付け加えます。付け加える項は以下で与えられます。

$$H+ = \sum_{i,j} V_{ij}(n_{i\uparrow} + n_{i\downarrow})(n_{j\uparrow} + n_{j\downarrow}) \tag{1.3}$$

最初の 5 行はヘッダで **xint** 以外はなにが書かれても問題ありません。**xint** が Off-site クーロン相互作用の総数です。**int01** でサイトの番号 i、**int02** でサイトの番号 j、**double1** でクーロン相互作用 V_{ij} の値を指定します。

1.4.5 zhund.def

以下で、**zhund.def** の例を示します。

1	====================	[xstring]
2	NHund 32	[xstring] [xint]
3	====================	[xstring]
4	====================	[xstring]
5	====================	[xstring]
6	0 1 0.5	[int01] [int02] [double1]
7	0 4 0.5	[int01] [int02] [double1]
8	1 2 0.5	[int01] [int02] [double1]
8	1 5 0.5	[int01] [int02] [double1]
	(continue . . .)	

Table 1.8: zhund.def の例。最初の部分のみを表示している。

Hund 相互作用の項をハミルトニアンに付け加えます。付け加える項は以下で与えられます。

$$H+ = -\sum_{i,j} J_{ij}^{\mathrm{Hund}}(n_{i\uparrow}n_{j\uparrow} + n_{i\downarrow}n_{j\downarrow}) \tag{1.4}$$

最初の 5 行はヘッダで **xint** 以外はなにが書かれても問題ありません。**xint** が Hund 相互作用の総数です。**int01** でサイトの番号 i、**int02** でサイトの番号 j、**double1** で Hund 相互作用 J_{ij}^{Hund} の値を指定します。

1.4.6 zpairhop.def

以下で、**zpairhop.def** の例を示します。

1	==========				[xstring]	
2	NPairhop	64			[xstring]	[xint]
3	==========				[xstring]	
4	==========				[xstring]	
5	==========				[xstring]	
6	0	1	0.5		[int01]	[int02] [double1]
7	0	4	0.5		[int01]	[int02] [double1]
8	1	2	0.5		[int01]	[int02] [double1]
8	1	5	0.5		[int01]	[int02] [double1]
	(continue . . .)					

Table 1.9: zpairhop.def の例。最初の部分のみを表示している。

Pairhopping 相互作用の項をハミルトニアンに付け加えます。付け加える項は以下で与えられます。

$$H+ = \sum_{i,j} J_{ij}^{\mathrm{Pair}} c_{i\uparrow}^{\dagger} c_{j\uparrow} c_{i\downarrow}^{\dagger} c_{j\downarrow} \tag{1.5}$$

最初の 5 行はヘッダで **xint** 以外はなにが書かれても問題ありません。**xint** が Pairhopping 相互作用の総数です [*7]。**int01** でサイトの番号 i、**int02** でサイトの番号 j、**double1** で Pairhopping 相互作用 J_{ij}^{Pair} の値を指定します。

[*7] 多軌道ハバード模型の場合 pairhopp の項の数が 2 倍になっていることに注意。これも混乱のもとなので、$H+ = \sum_{i,j} J_{ij}^{\mathrm{Pair}} (c_{i\uparrow}^{\dagger} c_{j\uparrow} c_{i\downarrow}^{\dagger} c_{j\downarrow} + c_{j\uparrow}^{\dagger} c_{i\uparrow} c_{j\downarrow}^{\dagger} c_{i\downarrow})$ にしたほうがよい？

1.4.7 zexchange.def

以下で、**zexchange.def** の例を示します。

1	====================	| [xstring]		
2	NExchange 32	| [xstring]	[xint]	
3	====================	| [xstring]		
4	====================	| [xstring]		
5	====================	| [xstring]		
6	0 1 0.5	| [int01]	[int02]	[double1]
7	0 4 0.5	| [int01]	[int02]	[double1]
8	1 2 0.5	| [int01]	[int02]	[double1]
8	1 5 0.5	| [int01]	[int02]	[double1]
	(continue . . .)			

Table 1.10: zexchange.def の例。最初の部分のみを表示している。

Exchange 相互作用の項をハミルトニアンに付け加えます。付け加える項は以下で与えられます。

$$H+ = \sum_{i,j} J_{ij}^{\mathrm{Ex}} (c_{i\uparrow}^{\dagger} c_{j\uparrow} c_{j\downarrow}^{\dagger} c_{i\downarrow} + c_{i\downarrow}^{\dagger} c_{j\downarrow} c_{j\uparrow}^{\dagger} c_{i\uparrow}) \tag{1.6}$$

最初の 5 行はヘッダで **xint** 以外はなにが書かれても問題ありません。**xint** が Hund 相互作用の総数です。**int01** でサイトの番号 i、**int02** でサイトの番号 j、**double1** で Exchange 相互作用 J_{ij}^{Ex} の値を指定します。

1.4.8 zinterall.def

今回のコードでは実装していないが、S_z 成分を保存する一般的な二体の相互作用

$$H+ = \sum_{i,j,\sigma,\tau} I_{ij\sigma\tau} c_{i\sigma}^{\dagger} c_{j\sigma} c_{k\tau}^{\dagger} c_{l\tau} \tag{1.7}$$

を実装したバージョンも存在する [*8]。計算がさぼれる実空間対角な相互作用以外は全てこれに統一したほうがよいかもしれない。

[*8] mVMC では標準実装済み。対角化の実装もたいして難しくない。

1.5 Definition files for output

1.5.1 one-body Green functions

以下で、一体の Green 関数を計算する **zcisajs.def** の例を示します。

1	==========				[xstring]
2	CisAjs	512			[xstring] [xint]
3	==========				[xstring]
4	==========				[xstring]
5	==========				[xstring]
6	0	0	0	0	[int01] [int02] [int03] [int04]
7	1	0	1	0	[int01] [int02] [int03] [int04]
8	2	1	0	0	[int01] [int02] [int03] [int04]
9	3	1	1	0	[int01] [int02] [int03] [int04]
	(continue . . .)				

Table 1.11: zcisajs.def の例。最初の部分のみを表示している。

これで、

$$\langle c^\dagger_{i\sigma} c_{j\sigma} \rangle = \langle \Phi | c^\dagger_{i\sigma} c_{j\sigma} | \Phi \rangle$$

計算します。

最初の 5 行はヘッダで **xint** 以外はなにが書かれても問題ありません。**xint** が計算する Green 関数の総数です。**int01** で計算する Green 関数の通し番号 n、**int02** でサイトの番号 i、**int03** でサイトの番号 j、**int04** でスピン σ を指定します。

出力は Lanczos 法、CG 法でもとめた波動関数による期待値を zvo_Lanczos_cisajs.dat, zvo_CG_cisajs.dat にそれぞれ出力します。

1.5.2 two-body Green functions

以下で、2 体の相関関数を計算する **zcisajscktaltdc.def** の例を示します。

1	==========						[xstring]
2	CisAjsCktAlt		1526				[xstring] [xint]
3	==========						[xstring]
4	==========						[xstring]
5	==========						[xstring]
6	0	0	0	0	0	0	[int01]. . . [int06]
7	0	0	0	0	0	1	[int01]. . . [int06]
8	0	0	1	0	0	0	[int01]. . . [int06]
9	0	0	1	0	0	1	[int01]. . . [int06]
	(continue . . .)						

Table 1.12: zcisajscktaltdc.def の例。最初の部分のみを表示している。

これで、

$$\langle c^\dagger_{i\sigma} c_{j\sigma} c^\dagger_{k\tau} c_{l\tau} \rangle = \langle \Phi | c^\dagger_{i\sigma} c_{j\sigma} c^\dagger_{k\tau} c_{l\tau} | \Phi \rangle$$

計算します。

最初の 5 行はヘッダで **xint** 以外はなにが書かれても問題ありません。**xint** が計算する相関関数の総数です。**int01** でサイトの番号 i、**int02** でサイトの番号 j、**int03** でスピン σ, **int04** でサイトの番号 k、**int05** でサイトの番号 l、**int06** でスピン τ, を指定します。

出力は Lanczos 法、CG 法でもとめた波動関数による期待値を zvo_Lanczos_cisajscktalt.dat, zvo_CG_cisajscktalt.dat にそれぞれ出力します。

2
計算の流れ

2.1 主な関数とその役割

関数の引数などは省略して概略をのべる。EDmain.c に出てくる関数を順に説明する。

ReadDefFileNInt() [readdef.c]: EDmodpara.def と各 def ファイルの 2 行目の xint を読み込む。

#include xsetmem_def.c：ReadDefFileNInt() で読み込んだ情報をもとに必要な配列を確保する [*1]。

ReadDefFileIdxPara() [readdef.c]: Hamiltonian などの情報を読み込む。

check() [check.c]: 組合せ ${}_nC_m$ を計算して、必要な次元などを求める [*2]。

#include xsetmem_large.c：chech で求めた次元をもとに波動関数を確保する。

EDTrans()：[EDTrans.c] トランスファーの情報を整理。

sgn()：[sgn.c] ホッピングでおきる符号の反転を計算するためのリスト list_3 を生成。

sz()：[sz.c] 計算に必要な実空間配列 list_1, list_2 を生成。TITPack と方法は同じ。

output_list()：[output_list.c] WRITE=1 なら list_1 を出力して終了。

dbln()：[dbln.c] 実空間対角な相互作用を計算をサボるために、list_Diagonal を作成している。これでメモリが少し増えてしまっている。

Lanczos_EigenValue()：[Lanczos_EigenValue.c] Lanczos 法で固有値を求めている。基本的には TITPack と同じ。実際に Hamitonian を波動関数に作用させているのは mltply_f と mltply の二つのルーチン。mltply_f は最初に作用させている場合で, 速度をあげるために transfer の部分が 0 になる場合を覚えている (sgn_u,sgn_d)。これはメモリを食うので 16site までの場合にしか使えない。

この内部で三重対角行列を対角化するのにデフォルトでは TITPack から移植し

[*1] この配列の確保は工夫の余地あり。とくに多次元配列。
[*2] あとあまり必要ないがトランスファーのチェックもしている。これはおそらく不要。

た bisection 法 (bisec.c) を使っている。lapack を使ったほうがよいかもしれない。いちおう lapcack を使うバージョンも ifdef 分岐でつくってはあるが [*3]、チェックはあまりしていない。

Lanczos_EigenVector()：[Lanczos_EigenVector.c] Lanczos 法で固有ベクトルを求めている。

expec_energy(),expec_cisajs(),expec_cisajscktaltdc: Lanczos 法で求めた固有ベクトルからエネルギー、グリーン関数、相関関数などの期待値を計算。一般に Lanczos 法の固有ベクトルの精度はあまりないので、この値はあまり信用できない。

CG_EigenVector: [CG_EigenVector.c] Lanczos 法で求めた固有ベクトルを初期ベクトルとして CG 法を行う。CG 法自体は TITPack と全く同じ。

expec_energy(),expec_cisajs(),expec_cisajscktaltdc: CG 法で求めた固有ベクトルからエネルギー、グリーン関数、相関関数などの期待値を計算。これは信頼のできる値になるはず。

2.2 使い方

1. コンパイル：sh com.sh とするだけ。実行体 ED が作成される。

2. Sec. 1 に従って def ファイルを作成。

3. ./ED EDnamelist.def　で実行。OPENMP を使うなら、適切に環境変数を設定。

4. 途中経過: zvo_Lanczos_Step.dat に Lanczos ステップの途中経過が出力される。zvo_TimeKeeper.dat に時間が出力される。

5. 最終結果: 物理量の期待値は zvo_Lanczos_energy.dat, zvo_CG_energy.dat などに出力される。

[*3] コンパイル時に -Dlapcak をつけるとそのバージョンになる。

3
Samples

3.1 Two-dimensional Hubbard model

2次元Hubbardモデルののdefファイルを作る一連のperlファイルはMakeDefHubbard以下に入っている。MakeDefHubbardのdef_ED.shを「sh def_ED.sh」とすればinput.txtを読み込んで、EDnamelist.defとEDmodpara.def**以外**のファイルは作成される。input.txtのLx,Lyはx方向の格子点、y方向の格子点の数。Uはon-siteクーロン。lambdaやorb_numは使用しないのでとりあえずいじる必要はない。

ED_Aft_Sq.plはzvo_CG_cisajscktalt.datからスピン相関関数、電荷相関関数を計算するスクリプト。Result_ED_Sq.dat, Result_ED_Nq.datがそれぞれスピン相関関数と電荷相関関数。

4
熱的純粋量子状態 (TPQ) の計算への適用

4.1 概要

杉浦・清水によって、少数個（サイズが大きい場合はほぼ一つ）の波動関数から有限温度の物理量を計算する方法が提案された。その状態は熱的純粋量子状態 (TPQ) と呼ばれている。TPQ はハミルトニアンを波動関数に順次作用させて得られるので、Lanczos 法の技術がそのまま使える。ここでは、とくに計算が簡単な, micro canonical TPQ(mTPQ) の概要を述べる。

$|\psi_0\rangle$ をあるランダムベクトルとする。これに $(l - \hat{H}/N_s)$(l はある定数、N_s はサイト数) を k 回作用させた（規格化された）ベクトルは次のように与えられる。

$$|\psi_k\rangle \equiv \frac{(l - \hat{H}/N_s)|\psi_{k-1}\rangle}{|(l - \hat{H}/N_s)|\psi_{k-1}\rangle|}. \tag{4.1}$$

この $|\psi_k\rangle$ が mTPQ 状態であり、この mTPQ 状態に対応する逆温度 β_k は以下のように内部エネルギー u_k から求めることができる。

$$\beta_k \sim \frac{2k/N_s}{l - u_k} \tag{4.2}$$

$$u_k = \langle\psi_k|\hat{H}|\psi_k\rangle/N_s. \tag{4.3}$$

そして、任意 [*1] の物理量 $\hat{A}$ の β_k での平均値は

$$\langle\hat{A}\rangle_{\beta_k} = \langle\psi_k|\hat{A}|\psi_k\rangle/N_s \tag{4.4}$$

となる。

[*1] 一応局所的という条件がつく。

4.2 defファイルの変更点

基本的には対角化とまったく一緒の def ファイルで動くが、EDmodpara.def だけすこし変更している。**EDmodpara.def** の例を以下に示します。

1	==========	[xstring]	
2	Model_Parameters 0	[xstring]	[xint]
3	==========	[xstring]	
4	VMC_Cal_Parameters	[xstring]	
5	==========	[xstring]	
6	CDataFileHead zvo	[xstring]	[string01]
7	CParaFileHead zqp	[xstring]	[string02]
8	==========	[xstring]	
9	Nsite 16	[xstring]	[int01]
10	Nup 5	[xstring]	[int02]
11	Ndown 5	[xstring]	[int03]
12	Lanczos_max 1000	[xstring]	[int04]
13	initial_iv 202	[xstring]	[int05]
14	nvec 1	[xstring]	[int06]
15	exct 1	[xstring]	[int07]
16	LanczosEps 14	[xstring]	[int08]
17	LanczosTarget 2	[xstring]	[int09]
18	WRITE 0	[xstring]	[int10]
19	READ 0	[xstring]	[int11]
19	Largevalue 10	[xstring]	[int12]
19	NumAve 20	[xstring]	[int13]
19	ExpecInterval 20	[xstring]	[int14]

Table 4.1: EDmodpara.def の例。一番左の数字は行番号, 右の [xstring] などは後の説明ための変数なので、入力はしないことに注意。

int01... int11 までは対角化と同じ。ただ、int05 は最初のランダムベクトルを与えるシードになっている。以下は、**EDmodpara.def** に追加した各パラメータの説明です。

- [int12]: $l - \hat{H}/N_s$ の l
- [int13]: 独立な run を何回行うかを指定する整数
- [int14]: zcisajs.def,zcisajsckaltdc.def で指定される物理量計算を何回おきに行うか。一回おきとかにしてしまうと計算時間が莫大になってしまうため。

4.3 output ファイル

SS_rand0.dat に 0 番目の run の物理量が出力されます。順番は β_k, $\langle\hat{H}\rangle$, $\langle\hat{H}^2\rangle$,$\langle\hat{D}\rangle$, k に出力される。ここで $\hat{D}$ は二重占有度。相関関数などは、zvo_cisajscktalt_set4step620.dat の形で出力される。set の後の数字は run の回数、step の後の数字は k 回目のステップという意味。step が多いと出力されるファイルの数が多くなるので注意。

5
Misc

いくつか、他の派生コードや適用可能性について。

5.1 その他の模型の適用例と適用可能性

S_z が保存する相互作用に対しては, スピン 1/2 の局在スピンの模型 (Heisenberg 模型) とスピン 1/2 の局在スピンと遍歴電子が結合した模型 (近藤格子模型) を取り扱うコードはある。基本的には def ファイルなどはほとんど一緒。コードとしては主にリストを生成する sz.c が変更されているだけ。この sz.c を変更すれば [*1]、Kitaev 模型などを取り扱えるコードになる。メモリ使用量は増えるが、コードとしては S_z 非保存のほうが簡単になる。

t-J 模型も軽微な修正で対応可能だが、いままで需要がなかったので作成していない。スピン 1 の Heisenberg もそこまで変更は必要ないはずだが、これも需要がなかったので作成していない。

5.2 実時間発展への適用

波動関数の実時間発展は原理的には以下の式で与えられる。

$$|\psi(t)\rangle = e^{i\hat{H}t}|\psi(0)\rangle. \tag{5.1}$$

これから、微小時間 Δt の時間発展は

$$|\psi(t+\Delta t)\rangle = e^{i\hat{H}\Delta t}|\psi(t)\rangle \tag{5.2}$$

$$\sim (1+i\hat{H}\Delta t)|\psi(t)\rangle \tag{5.3}$$

となる。要するにこれもハミルトニアンを順次作用させていくだけなので、Lanczos 法のアルゴリズムがそのまま使える。変更点としては、波動関数が複素数になることだけである。実時間発展を行うコードの作成も容易であり、実際に新手法開発のベンチマークに使っているコードがある。

[*1] ようするに $S_z = 0$ の制限をとるだけ。あとは波動関数を複素数にする。