JICFuS

Priority Issue 9
to be Tackled by Using Post K Computer
“Elucidation of the Fundamental Laws
and Evolution of the Universe”
KAKENHI grant 17K05433, 25870168

CNS Summer school 2019
2019/08/21-27, Hongo, The University of Tokyo

Nuclear shell model calculations – basics and practices –

2. shell model code “KSHELL”

CENTER for
NUCLEAR STUDY

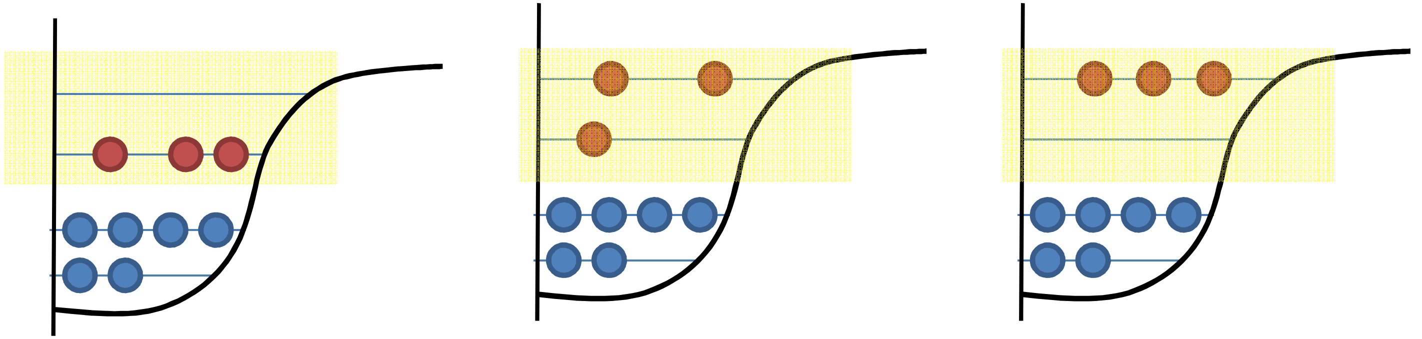
Noritaka Shimizu

東京大学
THE UNIVERSITY OF TOKYO

Center for Nuclear Study,
the University of Tokyo

Large-scale shell model calculation (LSSM)

- Consider the inert core and active particles in the valence shells (model space)
- Nuclear wave function is expressed as a linear combination of M-scheme basis states



$$|\Psi\rangle = \quad v_1|m_1\rangle \quad + \quad v_2|m_2\rangle \quad + \quad v_3|m_3\rangle \quad + \cdots$$

$$\left|\Psi\right\rangle = \sum_m v_m \left|m\right\rangle$$

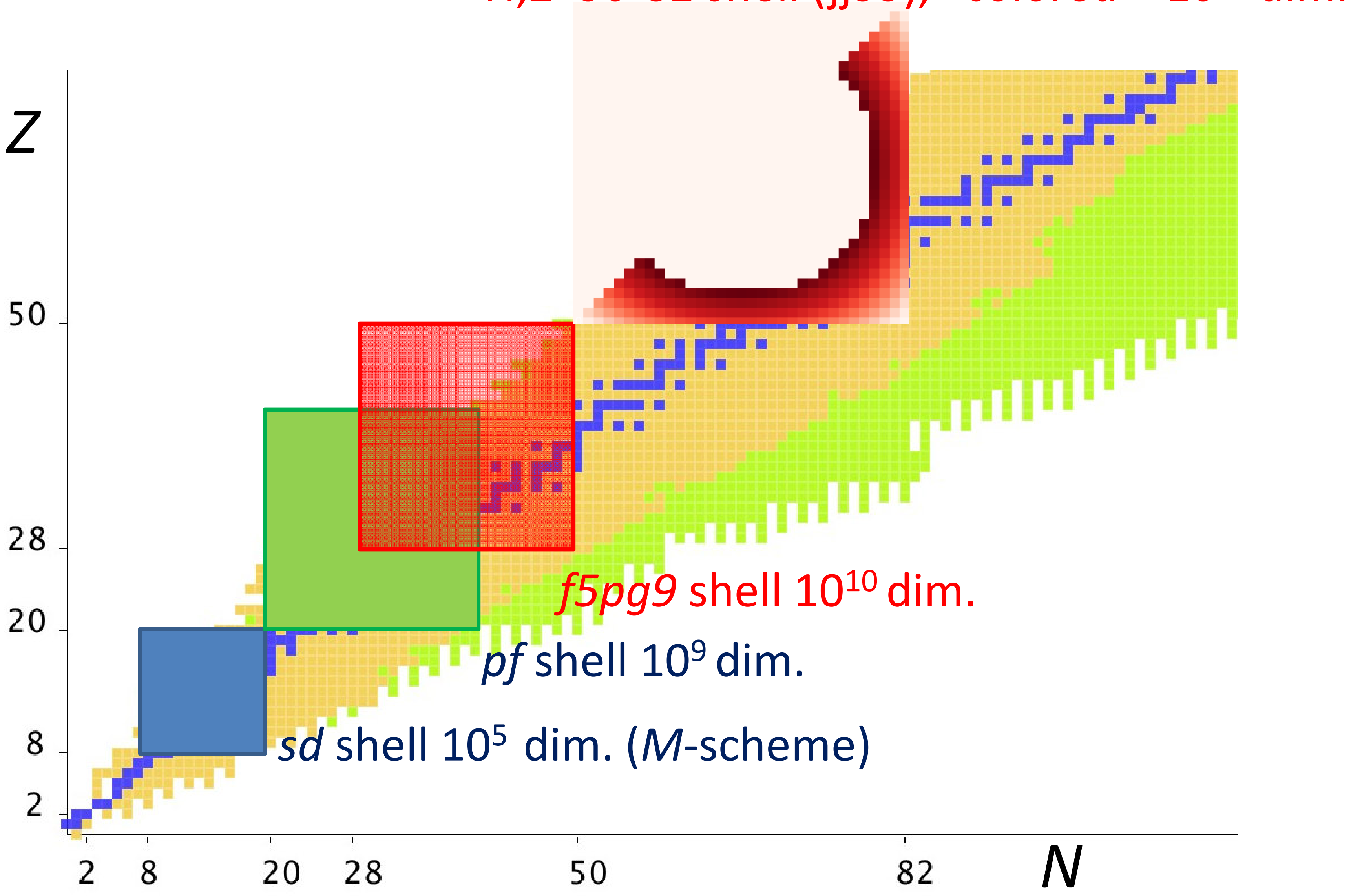
Schrodinger's equation

... Eigenvalue problem of huge sparse matrix

$$\boxed{\sum_k \left\langle m\middle|H\middle|k\right\rangle v_k = E v_m}$$

Solved to obtain low-lying eigenstates

Applicable region of shell-model calc.



N,Z=50-82 shell (jj55), colored < 10^{11} dim.

f5pg9 shell 10^{10} dim.

pf shell 10^{9} dim.

sd shell 10^{5} dim. (*M*-scheme)

Various shell-model codes

-------------- single node ----------------

- OXBASH/NuShell @MSU/Oxford
  - public, user interface, manual, OpenMP
  - *JT*-scheme
- ANTOINE / NATHAN @Strasbourg
  - public (ANTOINE only), highly tuned, single core
  - *M*-scheme / *J*-scheme
- MSHELL / MSHELL64 @Senshu T. Mizusaki et al.
  - *M*-scheme, unpublic
- Oslo code, CMichSM(CMU), EICODE(Jyvaskyla), jjSMQ(Kyusyu), ...

-------------- MPI Parallel ----------------

- BIGSTICK (San Diego), MFDn (Iowa, no core),....
  - supported by SciDAC UNEDF

M-scheme shell model code "KSHELL" can be used in a simple way.
From single PC to MPI+OpenMP supercomputer

shell-model code “KSHELL”

- *M*-scheme shell-model code

Ref. N. Shimizu, T. Mizusaki, T. Utsuno, and Y. Tsunoda, Comp. Phys. Comm. in press. https://doi.org/10.1016/j.cpc.2019.06.011

- MPI + OpenMP hybrid parallel, also useful for a PC
- Thick-restart block Lanczos method

- Awkward in no-core shell model calc., 3-body force is out of focus

Benchmark

^{46}Ti, pf-shell, KB3 interaction $D_M = 56{,}349$

Elapsed time to obtain 10 lowest *J*=4 states

(J=4, T=4 10 lowest states for OXBASH)

@ Xeon E5-2680v2 2.80GHz, 20 CPU cores

- OXBASH 227.3 sec.
- KSHELL 117.5 sec. 1 thread, block size=1
- KSHELL 8.6 sec. 20 threads, block size=1
- KSHELL 3.5 sec. 20 threads, block size=6

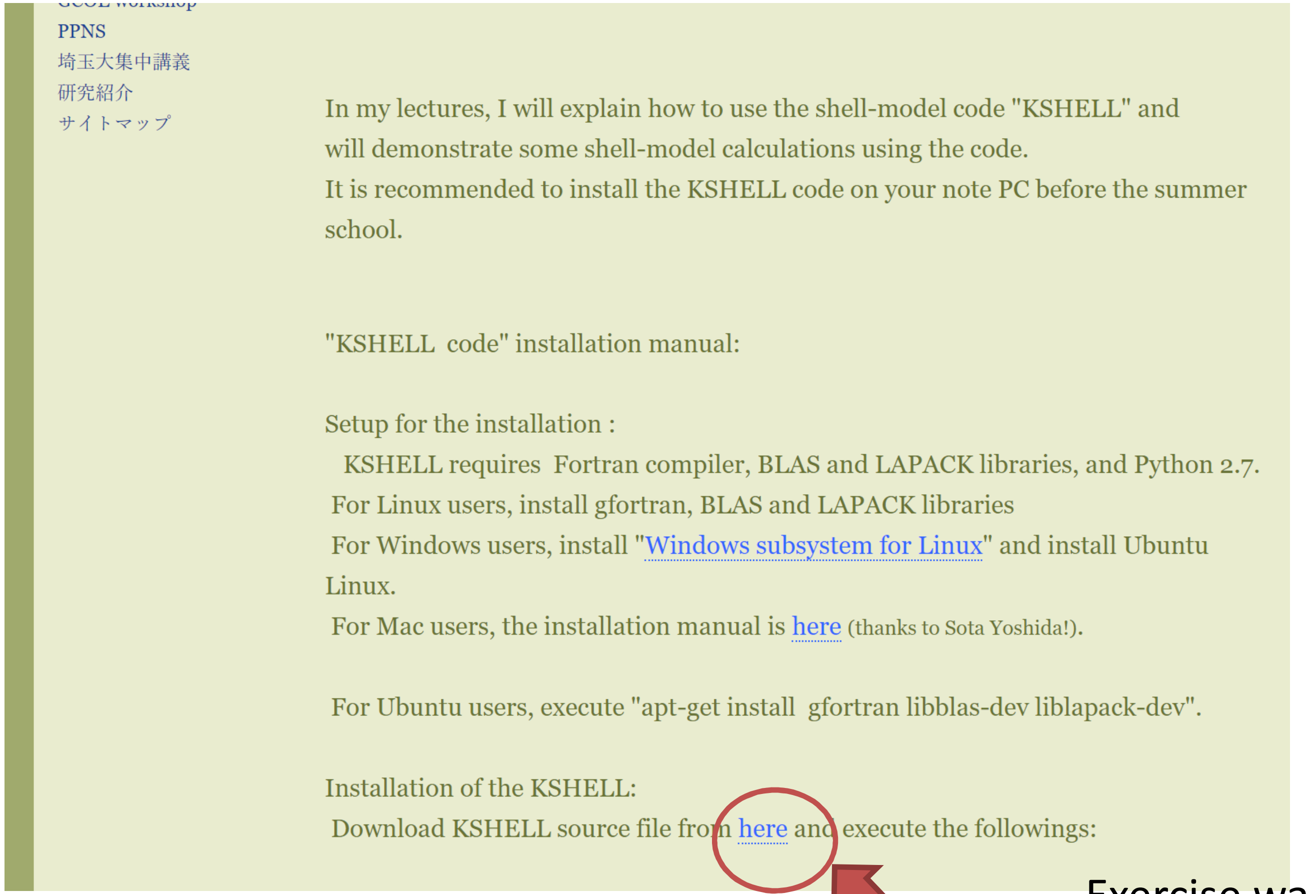
“The computer code OXBASH”, BA Brown, A Etchegoyen, WDM Rae, NS Godwin - MSU-NSCL Report, 1988

KSHELL how to

https://sites.google.com/a/cns.s.u-tokyo.ac.jp/shimizu/cns-summer-school-2019

Ref. N. Shimizu *et al.,* Comp. Phys. Comm. in press.

https://doi.org/10.1016/j.cpc.2019.06.011



In my lectures, I will explain how to use the shell-model code "KSHELL" and will demonstrate some shell-model calculations using the code.
It is recommended to install the KSHELL code on your note PC before the summer school.

"KSHELL code" installation manual:

Setup for the installation :
KSHELL requires Fortran compiler, BLAS and LAPACK libraries, and Python 2.7.
For Linux users, install gfortran, BLAS and LAPACK libraries
For Windows users, install "Windows subsystem for Linux" and install Ubuntu Linux.
For Mac users, the installation manual is here (thanks to Sota Yoshida!).

For Ubuntu users, execute "apt-get install gfortran libblas-dev liblapack-dev".

Installation of the KSHELL:
Download KSHELL source file from here and execute the followings:

Exercise was uploaded yesterday

Download here.

Preparation

- https://sites.google.com/a/cns.s.u-tokyo.ac.jp/shimizu/cns-summer-school-2019

Linux:

- Install gfortran, BLAS, LAPACK, and python (or Intel Fortran + MKL)

  (Ubuntu: apt-get install python gfortran liblapack-dev libblas-dev)

-
  ```
  tar xvzf kshell-cpc.tar.gz
  cd kshell-cpc/src
  make
  cd ../test
  ../bin/kshell_ui.py
  ```

- MS-Windows : install “Windows subsystem for Linux” or Cygwin
- Mac OS X : install Xcode

How to install

```
tar xvzf kshell-cpc.tar.gz

cd kshell-cpc/src

make

alias kshell_ui.py=(installed dir)/kshell-cpc/bin/kshell_ui.py
```

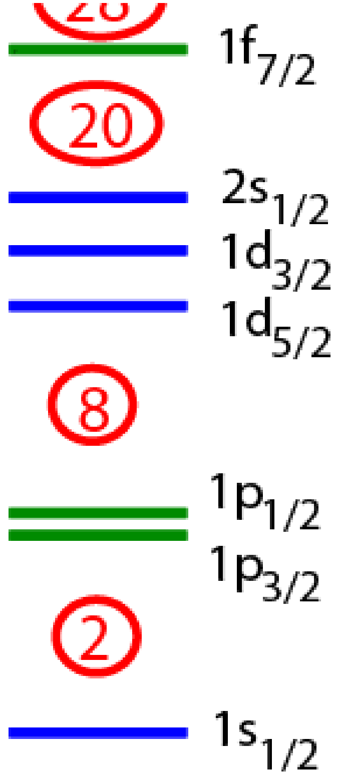
That is all, if you are lucky.

How to run

1. kshell_ui.py

... answer questions to generate a shell script

2. run the generated script



Example: ^{28}Si with USD interaction
6 protons, 6 neutrons with ^{16}O core
model space : sd-shell $(0d_{5/2}, 1s_{1/2}, 0d_{3/2})$
93,710 *M*-scheme dim.

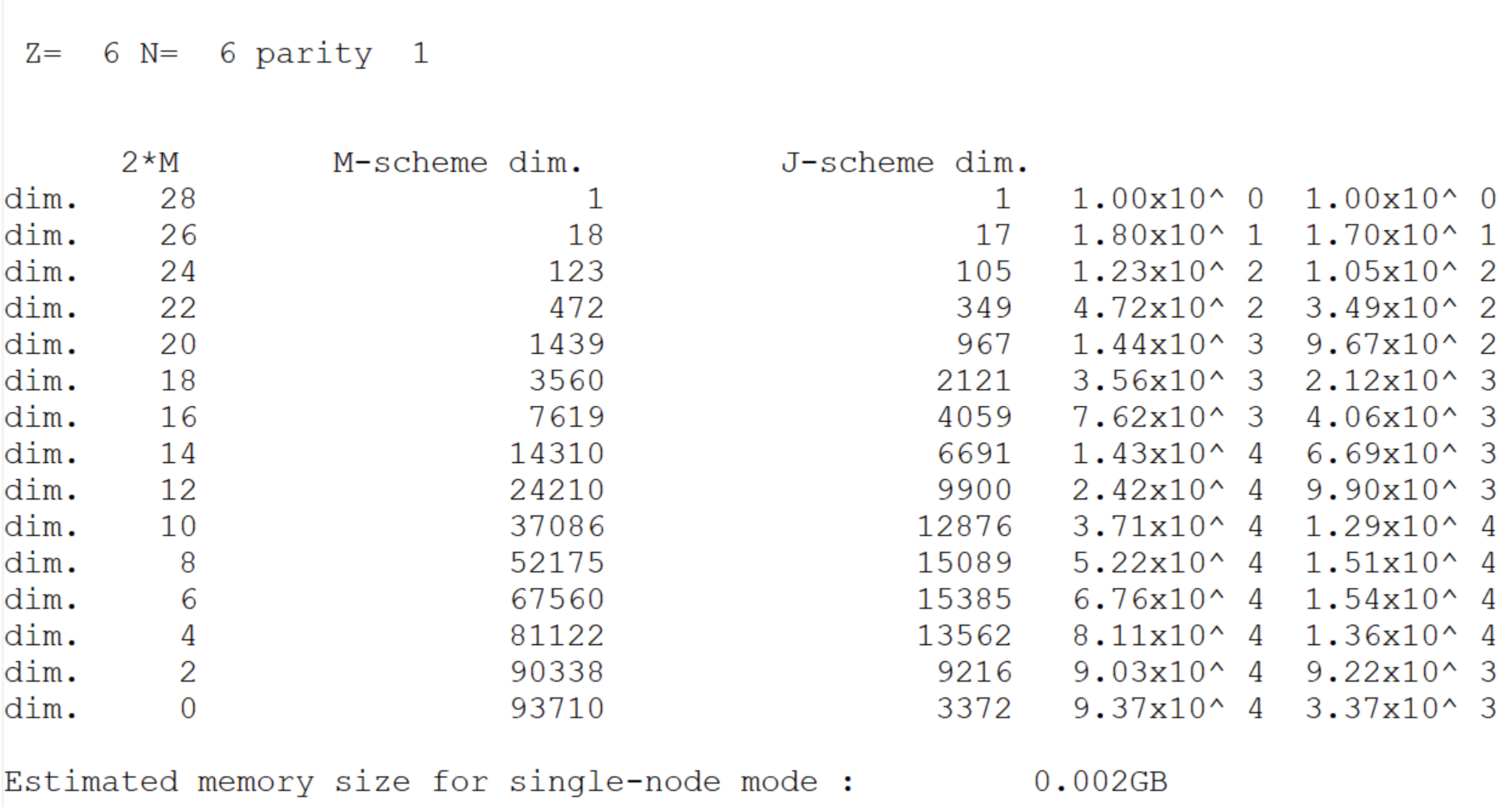
Demonstration

- 28Si in sd-shell

Count dimensions

- count M-scheme and J-scheme dimensions

../bin/count_dim.exe w.snt Si28_w_p.ptn



```
 Z=   6 N=   6 parity  1


      2*M         M-scheme dim.          J-scheme dim.
dim.    28                    1                      1    1.00x10^ 0   1.00x10^ 0
dim.    26                   18                     17    1.80x10^ 1   1.70x10^ 1
dim.    24                  123                    105    1.23x10^ 2   1.05x10^ 2
dim.    22                  472                    349    4.72x10^ 2   3.49x10^ 2
dim.    20                 1439                    967    1.44x10^ 3   9.67x10^ 2
dim.    18                 3560                   2121    3.56x10^ 3   2.12x10^ 3
dim.    16                 7619                   4059    7.62x10^ 3   4.06x10^ 3
dim.    14                14310                   6691    1.43x10^ 4   6.69x10^ 3
dim.    12                24210                   9900    2.42x10^ 4   9.90x10^ 3
dim.    10                37086                  12876    3.71x10^ 4   1.29x10^ 4
dim.     8                52175                  15089    5.22x10^ 4   1.51x10^ 4
dim.     6                67560                  15385    6.76x10^ 4   1.54x10^ 4
dim.     4                81122                  13562    8.11x10^ 4   1.36x10^ 4
dim.     2                90338                   9216    9.03x10^ 4   9.22x10^ 3
dim.     0                93710                   3372    9.37x10^ 4   3.37x10^ 3

Estimated memory size for single-node mode :          0.002GB
```

$$D_J = D_{M=J} - D_{M=J+1}$$

output : summary_Si28_w.txt

Energy relative to ^{16}O core

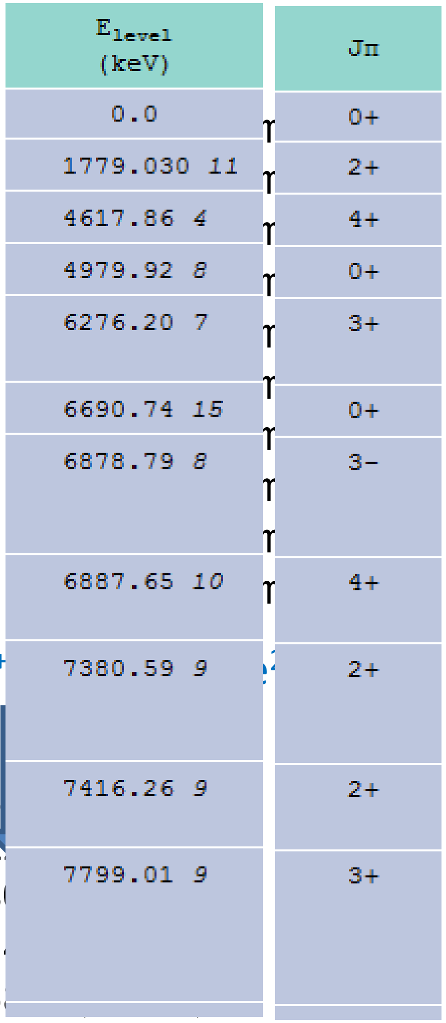
Excitation energy

Experiment (Nudat2)

```
 Energy levels

       N     J prty N_Jp      T      E(MeV)    Ex(MeV)

0+_1   1      0 +     1       0    -135.938     0.000
  2+_1 2      2 +     1       0    -133.950     1.987
4+_1   3      4 +     1       0    -131.279     4.659
  0+_2 4      0 +     2       0    -130.927     5.011
3+_1   5      3 +     1       0    -129.771     6.167
  4+_2 6      4 +     2       0    -128.901     7.037
0+_3   7      0 +     3       0    -128.699     7.239
  2+_2 8      2 +     2       0    -128.415     7.522
2+_3   9      2 +     3       0    -128.032     7.906
  1+_1 10     1 +     1       0    -127.998     7.940
```

B(E2; 2+

```
B(E2)  ( > -0.0 W.u.)  mass = 28    1 W.u. = 5
                                           e^2

  J_i     Ex_i     J_f     Ex_f    dE          B(E
  2+( 1)   1.988    0+( 1)  0.000   1.987       8                80.1)
  4+( 1)   4.659    2+( 1)  1.988   2.671      11                40.8)
  0+( 2)   5.011    2+( 1)  1.988   3.024       6                 2.7)
```



E_{level} (keV)	Jπ
0.0	0+
1779.030 11	2+
4617.86 4	4+
4979.92 8	0+
6276.20 7	3+
6690.74 15	0+
6878.79 8	3-
6887.65 10	4+
7380.59 9	2+
7416.26 9	2+
7799.01 9	3+

output : log_Si28_w_m0p.txt

```
total # of partitions                   1679  = 10** 3.23
total m-scheme dimension               93710  = 10** 4.97
 max. # dim. / a partition                    1156
 max local dim. / proc, average                   93710                  93710

Memory for one global Lanczos vector:     0.000 GB
Memory / process is:     0.000 GB x     10 =     0.003 GB
Total Memory for Lanczos vectors:     0.003 GB
...
...
...
```

Energy

0^+_1

```
-------------------------------------------------
   1  <H>:  -135.93772  <JJ>:     0.00000  J:  0/2  prty  1
                        <TT>:     0.00000  T:  0/2
 <p Nj>  0.673  4.623  0.704
 <n Nj>  0.673  4.623  0.704
-------------------------------------------------
```

Occupation number of each orbit ($d_{3/2}, d_{5/2,} s_{1/2}$)

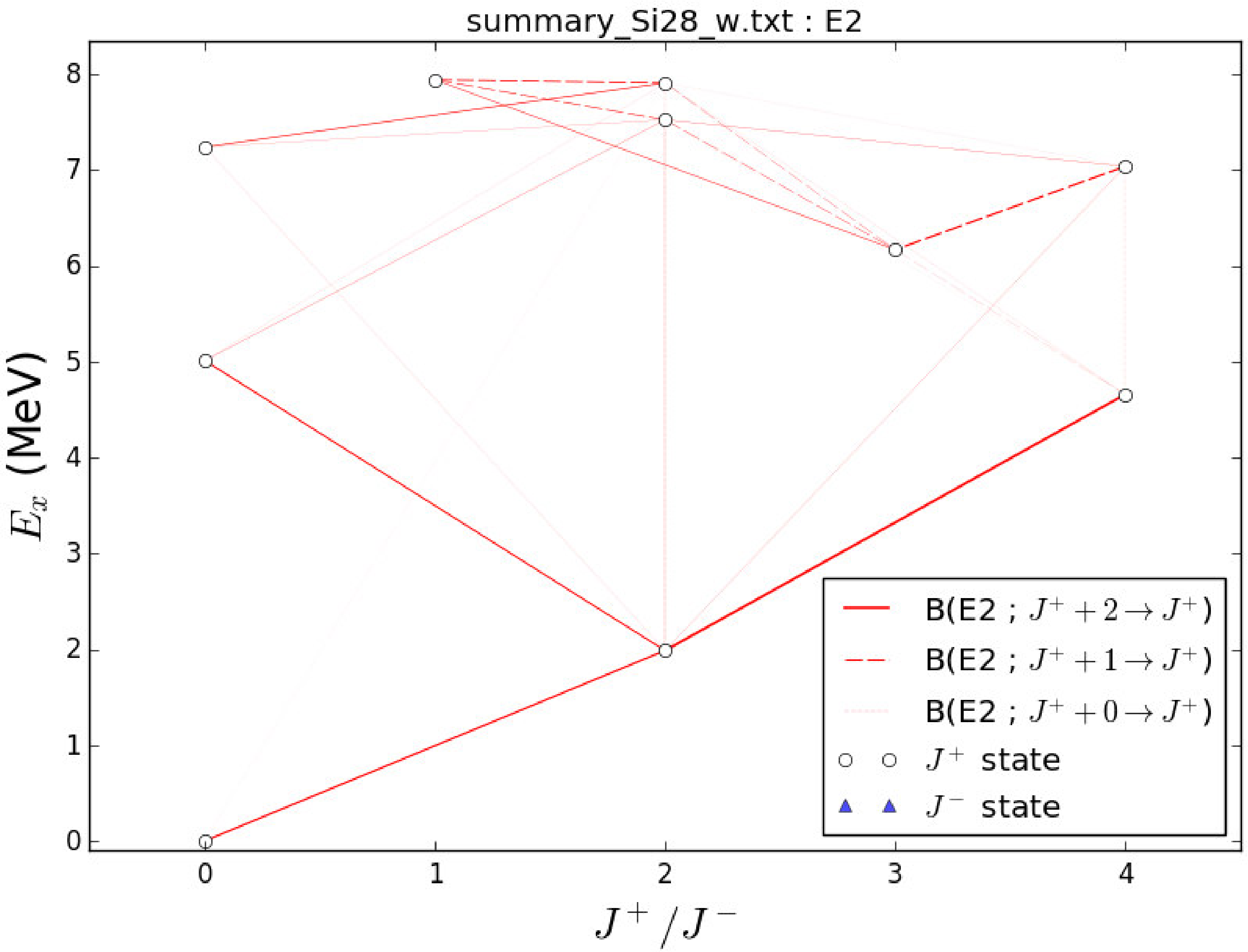
2^+_1

```
   2  <H>:  -133.95030  <JJ>:     6.00000  J:  4/2  prty  1
                        <TT>:     0.00000  T:  0/2
 <p Nj>  0.771  4.252  0.977
 <n Nj>  0.771  4.252  0.977
   <Qp>    10.375   <Qn>    10.375   <eQ>     20.751
-------------------------------------------------

....
```

Quadrupole moment

E2 map

./map_transit.py summary_Si28_w.txt



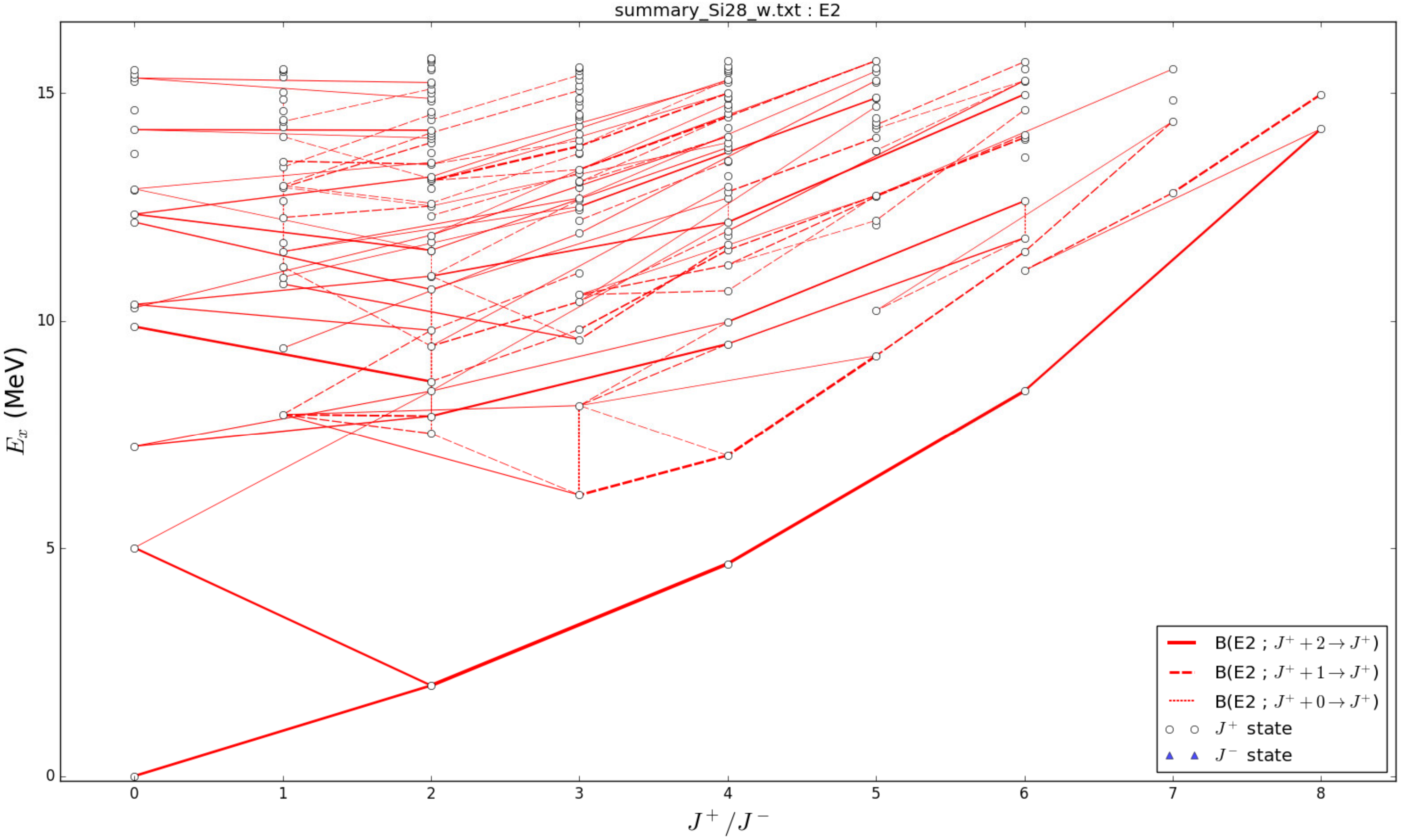
x-axis: J

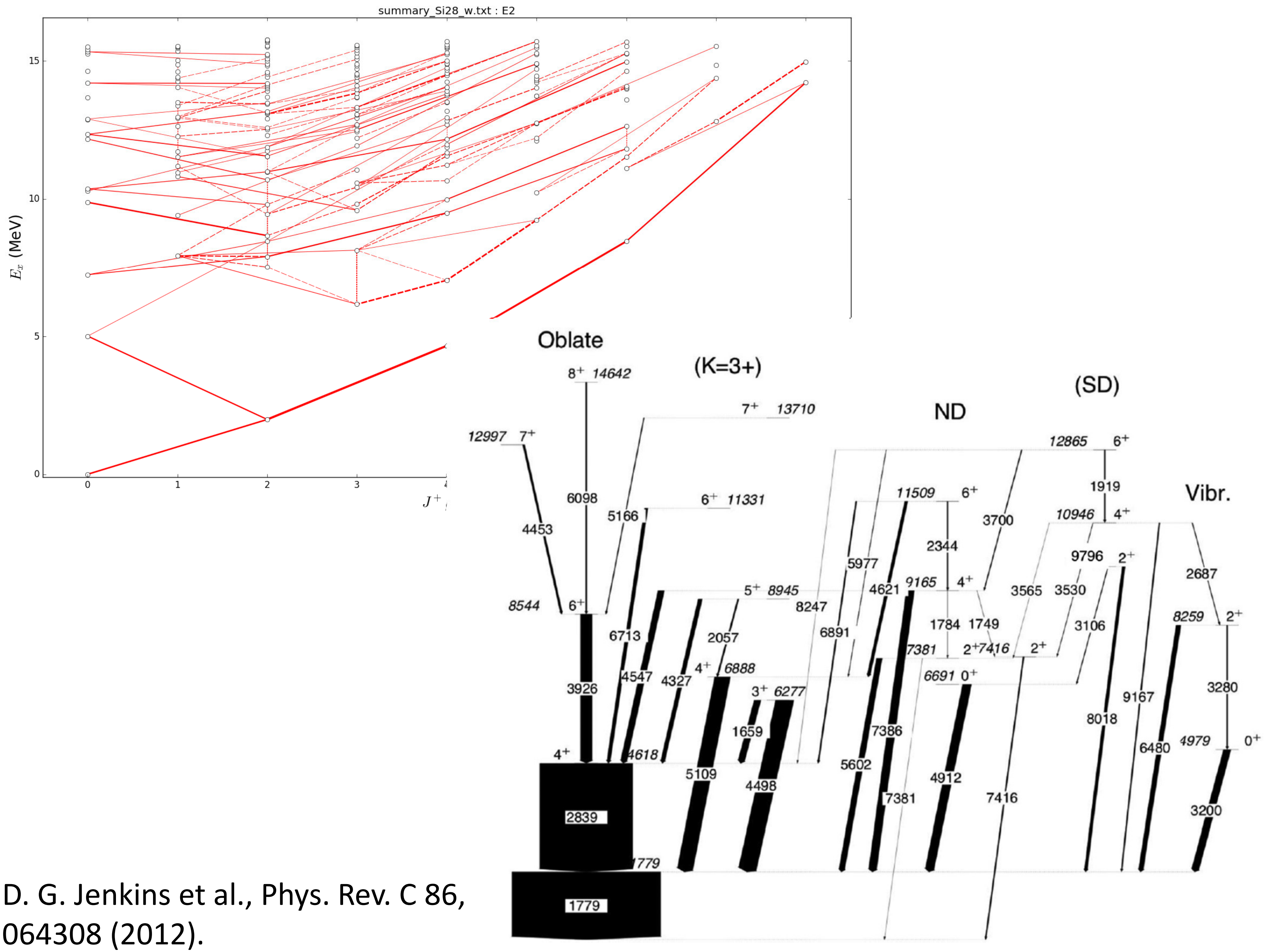
y-axis : Ex. (MeV)

The width of the connected red line is proportional to the B(E2) values.

This figure would help you to find band structures.

E2 map: ^{28}Si, 200 states





D. G. Jenkins et al., Phys. Rev. C 86, 064308 (2012).

interaction and model space

kshell/snt

- w.snt ... Wildenthal's USD interaction (for sd shell)
- gxpf1a.snt ... GXPF1A interaction (for pf shell)
- jun45.snt ... JUN45 interaction (for f5pg9 shell Z,N=28-50)
- sdpf-mu.snt … SDPF-MU interaction (for sdpf shell)

GXPF1A, JUN45	Courtesy of Michio Honma (Aizu)
SDPF-M, SDPF-MU	Courtesy of Yutaka Utsuno (JAEA)

"snt" file defines model space and its interaction.

You can make by your own snt file.

Some famous interaction files are equipped.

Interaction file in NuShell/OXBASH package can be transformed with "nushell2snt.py"

w.snt (USD interaction)

```
! Wildenthal's USD interaction for sd-shell
! B. A. Brown and B. H. Wildenthal, Annu. Rev. Nucl. Part. Sci. 38, 29 (1988
! proton-orbit, neutron-orbit, proton core, neutron core
3 3 8 8
! n  l j  tz
! model space
  #       n,   l,   2j   2tz
  1       0    2    3    -1     !  1 = p 0d_3/2
  2       0    2    5    -1     !  2 = p 0d_5/2
  3       1    0    1    -1     !  3 = p 1s_1/2
  4       0    2    3     1     !  4 = n 0d_3/2
  5       0    2    5     1     !  5 = n 0d_5/2
  6       1    0    1     1     !  6 = n 1s_1/2
! one-body interaction
! number of lines, method1
 6   0
  1    1       1.64658
  2    2      -3.94780
  3    3      -3.16354
  4    4       1.64658
  5    5      -3.94780
  6    6      -3.16354
```

Model space
(*sd* shell)

Single-particle energy of each orbit
(one-body interaction)

$\mathrm{SPE}(d_{5/2})$ = -3.9478 MeV

w.snt (USD interaction) cont’d

Two-body matrix elements (TBME)

```
! two-body interaction (TBME)
! # of lines, method2 A mass dependence factor
158   1  18  -0.30000
!  i   j   k   l     J    <i,j| V | k,l>_J
   1   1   1   1     0      -2.18450
   1   1   1   1     2      -0.06650
   1   1   1   2     2       0.61490
   1   1   1   3     2       0.51540
   1   1   2   2     0      -3.18560
   1   1   2   2     2      -1.62210
   1   1   2   3     2      -0.40410
   1   1   3   3     0      -1.08350
   1   2   1   2     1       1.03340
   1   2   1   2     2      -0.32480
   1   2   1   2     3       0.58940
   1   2   1   2     4      -1.44970
...
   5   5   5   5     0      -2.81970
... 158 lines continued
```

$\mathrm{SPE}(d_{5/2})$ = -3.9478 MeV

$$\left\langle \nu 0d_{5/2}, \nu 0d_{5/2} \middle| V \middle| \nu 0d_{5/2}, \nu 0d_{5/2} \right\rangle_{J=0}$$

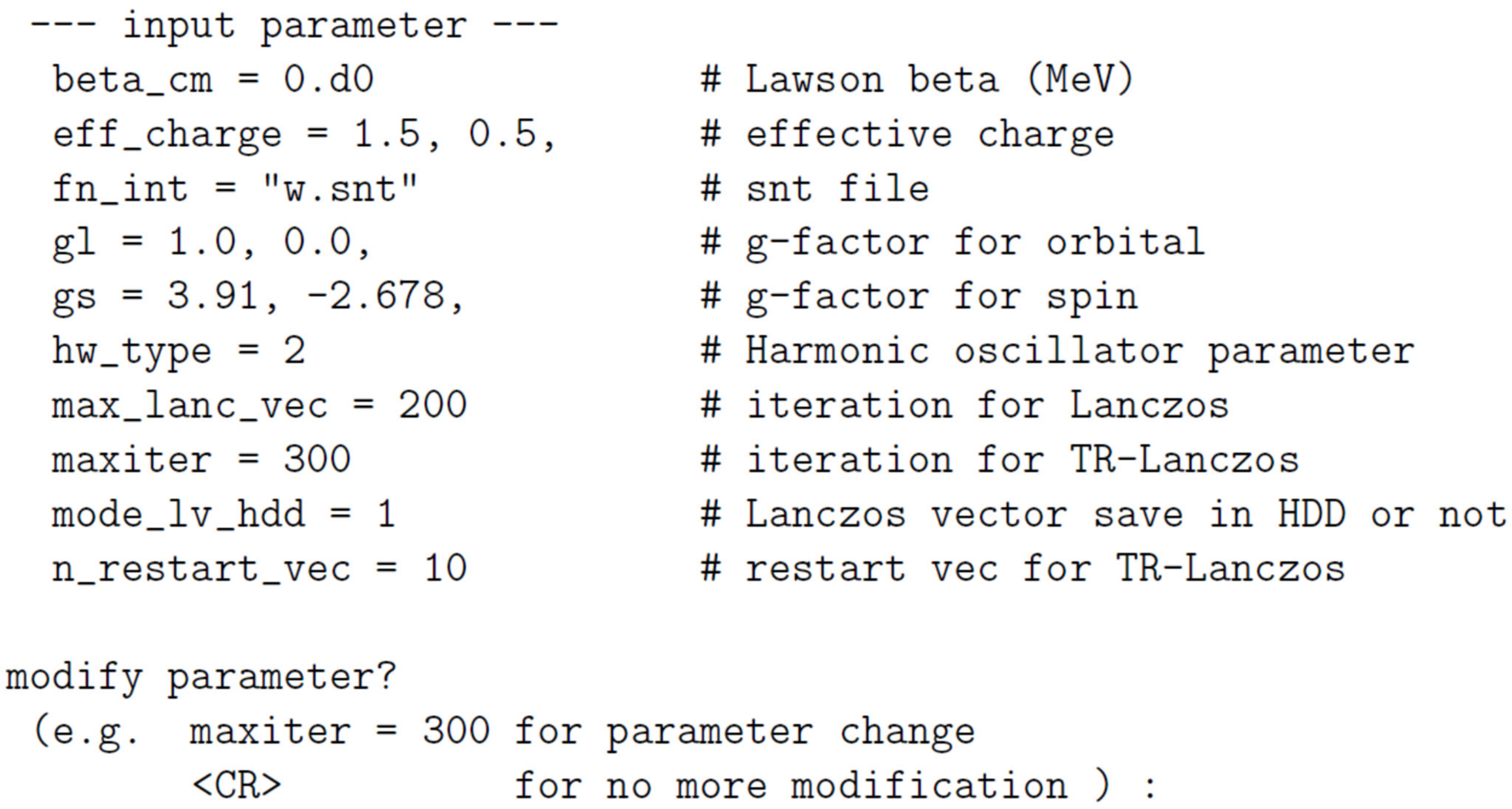
$$= -2.8197\ \mathrm{MeV}$$

Shell-model energy of ^{18}O in d5/2 orbit is

$-3.947 * 2 - 2.189 = -10.713$ MeV

Shell-model energy in full *sd* shell is

$E\ =\ -12.171$ MeV

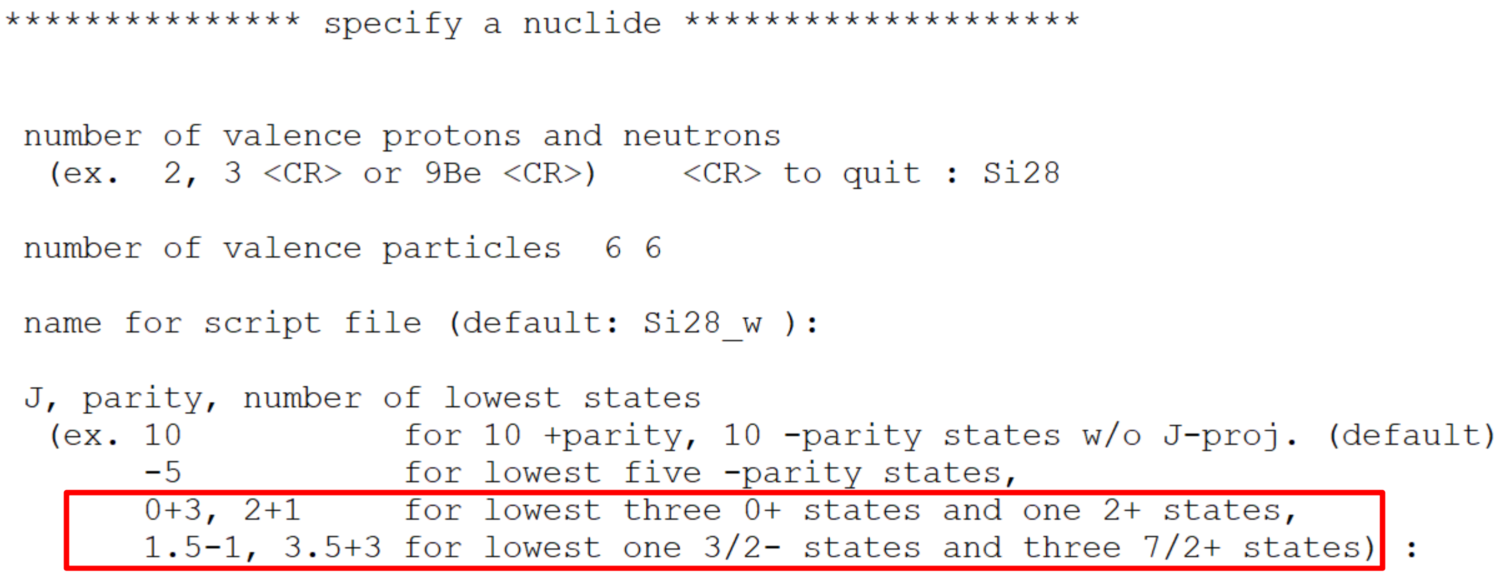


```
 --- input parameter ---
  beta_cm = 0.d0                # Lawson beta (MeV)
  eff_charge = 1.5, 0.5,        # effective charge
  fn_int = "w.snt"              # snt file
  gl = 1.0, 0.0,                # g-factor for orbital
  gs = 3.91, -2.678,            # g-factor for spin
  hw_type = 2                   # Harmonic oscillator parameter
  max_lanc_vec = 200            # iteration for Lanczos
  maxiter = 300                 # iteration for TR-Lanczos
  mode_lv_hdd = 1               # Lanczos vector save in HDD or not
  n_restart_vec = 10            # restart vec for TR-Lanczos

modify parameter?
 (e.g.  maxiter = 300 for parameter change
        <CR>          for no more modification ) :
```

n_block = 4 … block size for block Lanczos method for acceleration

J-projection

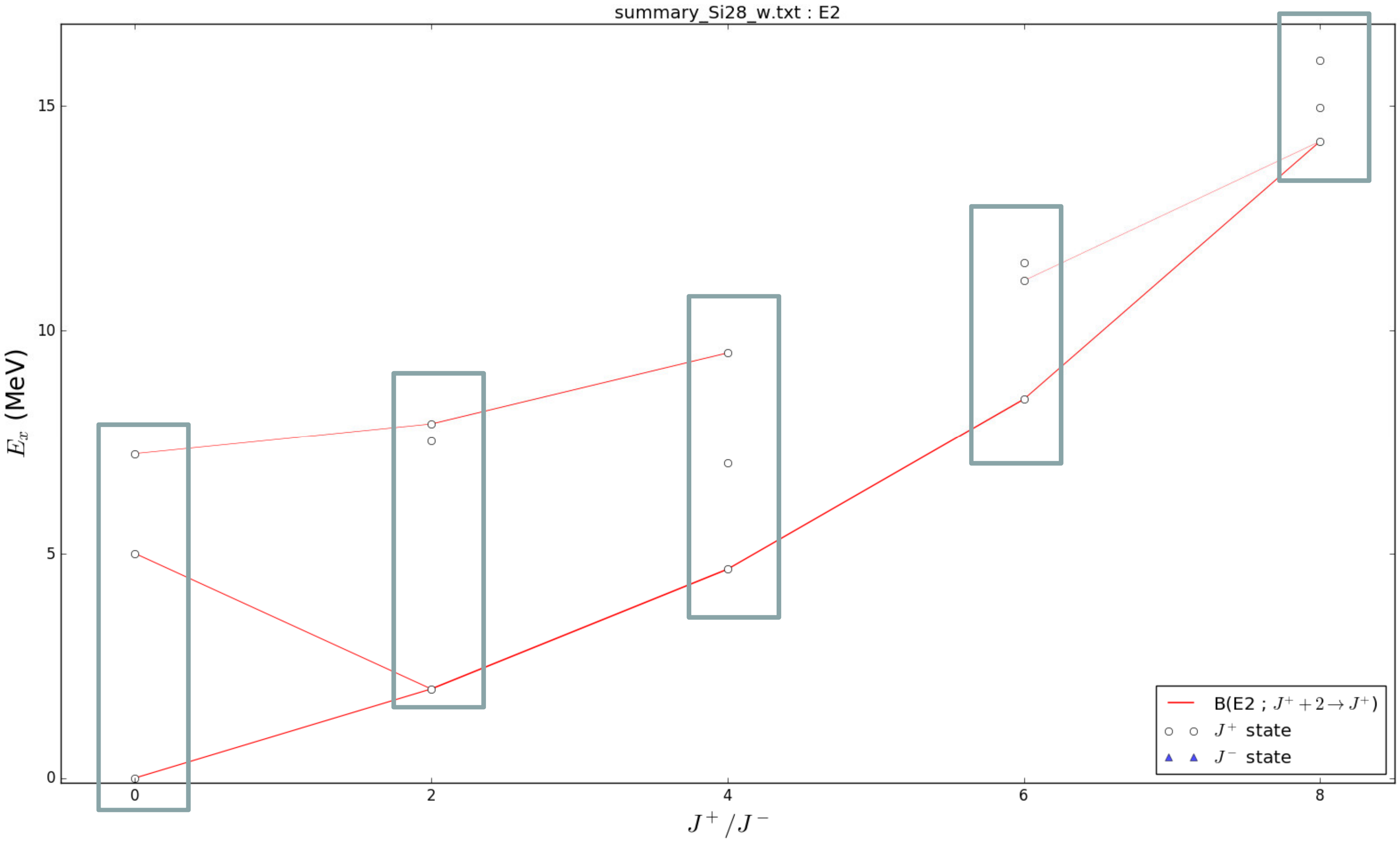
- By default, KSHELL diagonalizes the Hamiltonian in $M = 0$ subspace ($M = \frac{1}{2}$ for odd nuclei).
- It can also obtain only specified-*J* states by projecting the Lanczos vectors to good *J* states at every Lanczos iteration.



```
*************** specify a nuclide ********************


 number of valence protons and neutrons
  (ex.  2, 3 <CR> or 9Be <CR>)    <CR> to quit : Si28

 number of valence particles   6 6

 name for script file (default: Si28_w ):

 J, parity, number of lowest states
  (ex. 10            for 10 +parity, 10 -parity states w/o J-proj. (default)
       -5            for lowest five -parity states,
       0+3, 2+1      for lowest three 0+ states and one 2+ states,
       1.5-1, 3.5+3 for lowest one 3/2- states and three 7/2+ states) :
```

0+3, 2+3, 4+3, 6+3, 8+3

E2 map: ^{28}Si, 3 states for each J



Input parameters for shell-model calculations

- Model space and Hamiltonian
  - ask shell-model people!
- effective charges for *Q*-moment, B(E2)
  - $(e_\pi, e_\nu) = (1.5, 0.5)e$ is typical value, caused by the core polarization
- *g*-factor for *M*-moment, B(M1)
  - $g_{l\pi} = 1,\ \mathrm{g}_{\mathrm{l}\nu} = 0,\ \mathrm{g}_{s\pi} = 5.586,\ \mathrm{g}_{s\nu} = -3.826$ for free particles
  - spin *g*-factor is typically quenched by 0.7, caused by the core polarization and meson exchange current
- $\hbar\omega$: Energy of the harmonic oscillator quanta
  - $\hbar\omega = 41\mathrm{A}^{-1/3}$ or $\hbar\omega = 45A^{-1/3} - 25A^{-2/3}$
- Lawson's beta for removing contamination of center-of-mass motion (beyond $0\hbar\omega$ model space)
  - $\frac{\beta_{CM}\hbar\omega}{A} = 10.$ $\quad H' = H + \beta_{\mathrm{CM}} H_{\mathrm{CM}}$

Spectroscopic factor

- Spectroscopic factor of $A + n \leftrightarrows B$ reaction is defined as

$$C^2 S_j = \frac{\left| \langle \Psi_B || a_j^\dagger || \Psi_A \rangle \right|^2}{2J_B + 1}$$

j : single-particle orbit

- It is well-defined in shell-model calculations, while the value determined by experiments is model dependent (reaction models).

Spectroscopic factor in shell-model calc.

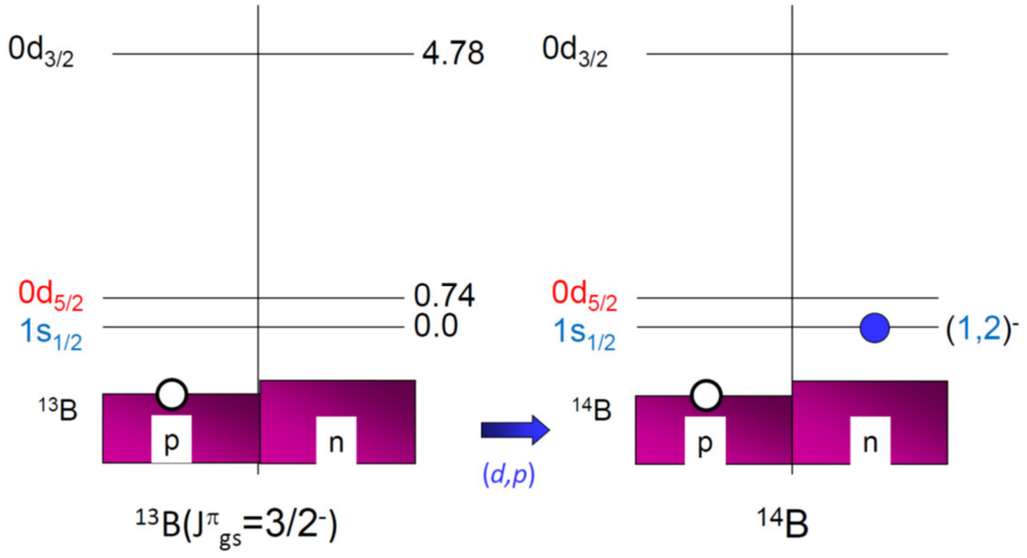
- In case of ^{13}B(d,p)^{14}B reaction,

$$\mathrm{C^2S} = \frac{\langle {}^{14}\mathrm{B}; J^{\pi} || c_j^{\dagger} || {}^{13}\mathrm{B}; \frac{3}{2}{}_1^{-} \rangle}{2J+1}$$

$$j = \nu 1s_{1/2}, \nu 0d_{5/2}$$

$$J^{\pi} = 2^-, 1^-, 3^-, 4^-$$

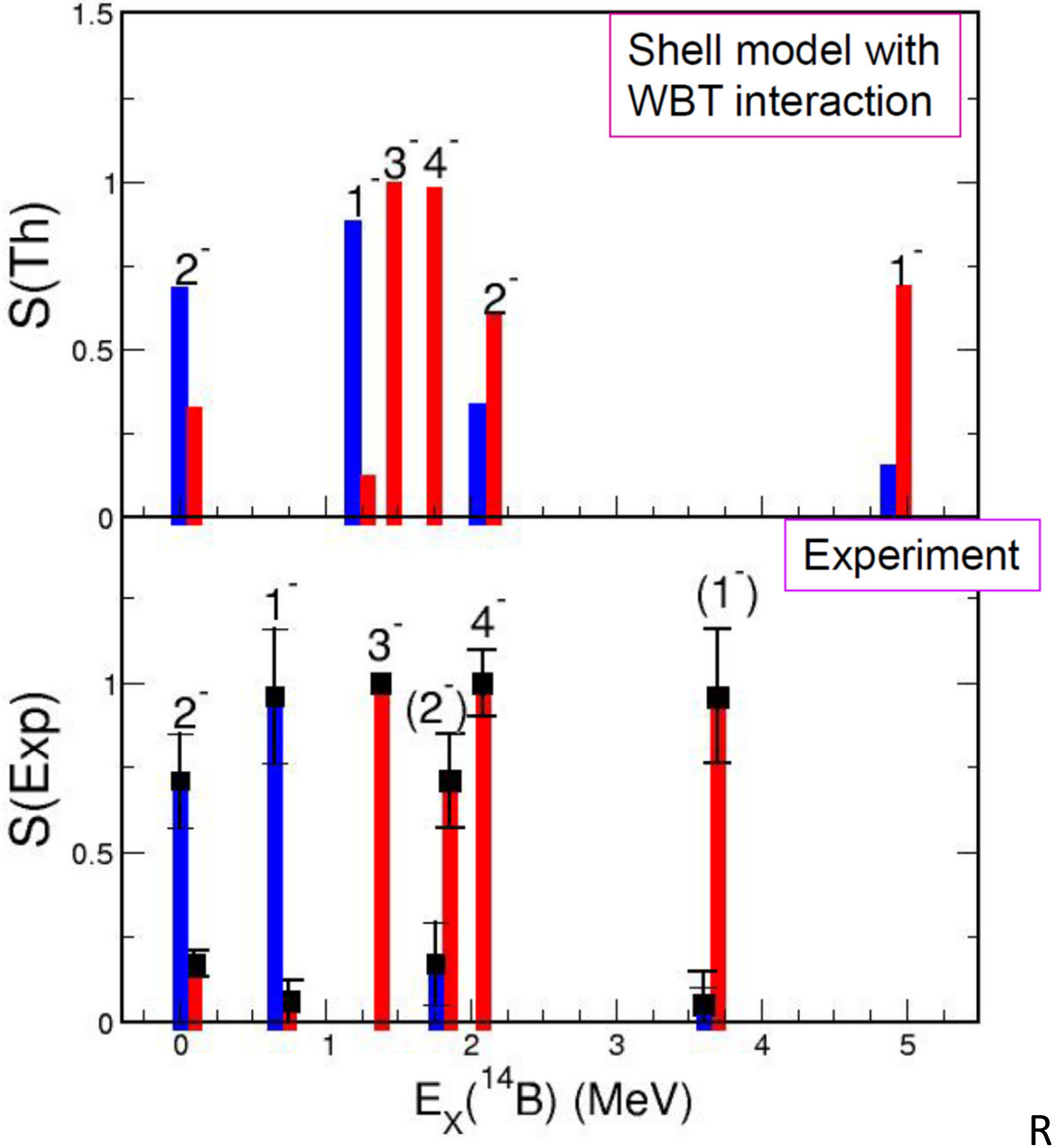
Simple picture for ^{13}B(*d*,*p*)^{14}B



(*d*,*p*) populates single-neutron states in ^{14}B
(schematic picture)

From 2nd lecture by Prof. Wuosmaa

^{13}B(*d*,*p*)^{14}B Spectroscopic factors



Excitation energies and relative spectroscopic factors from the shell model

Blue: L=0, 1$s_{1/2}$

Red: L=2, 0$d_{5/2}$

Ref. PRC 88, 011304(R) (2013)

From 2nd lecture by Prof. Wuosmaa

Exercise 1

Backbending in ^{48}Cr

Exp. Nudat 2.6
http://www.nndc.bnl.gov/nudat2/

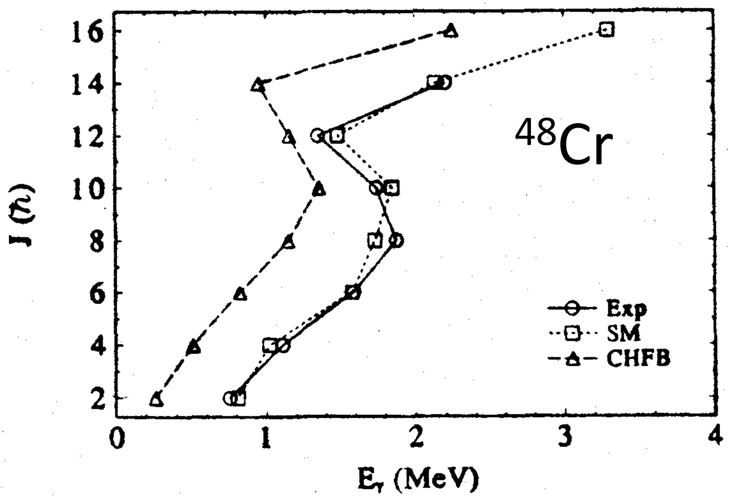
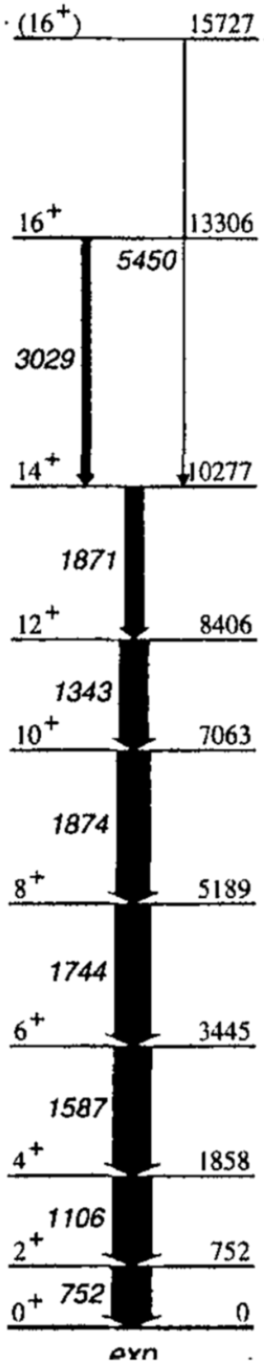


FIG. 1. Yrast energies $E_\gamma = E(J) - E(J-2)$.

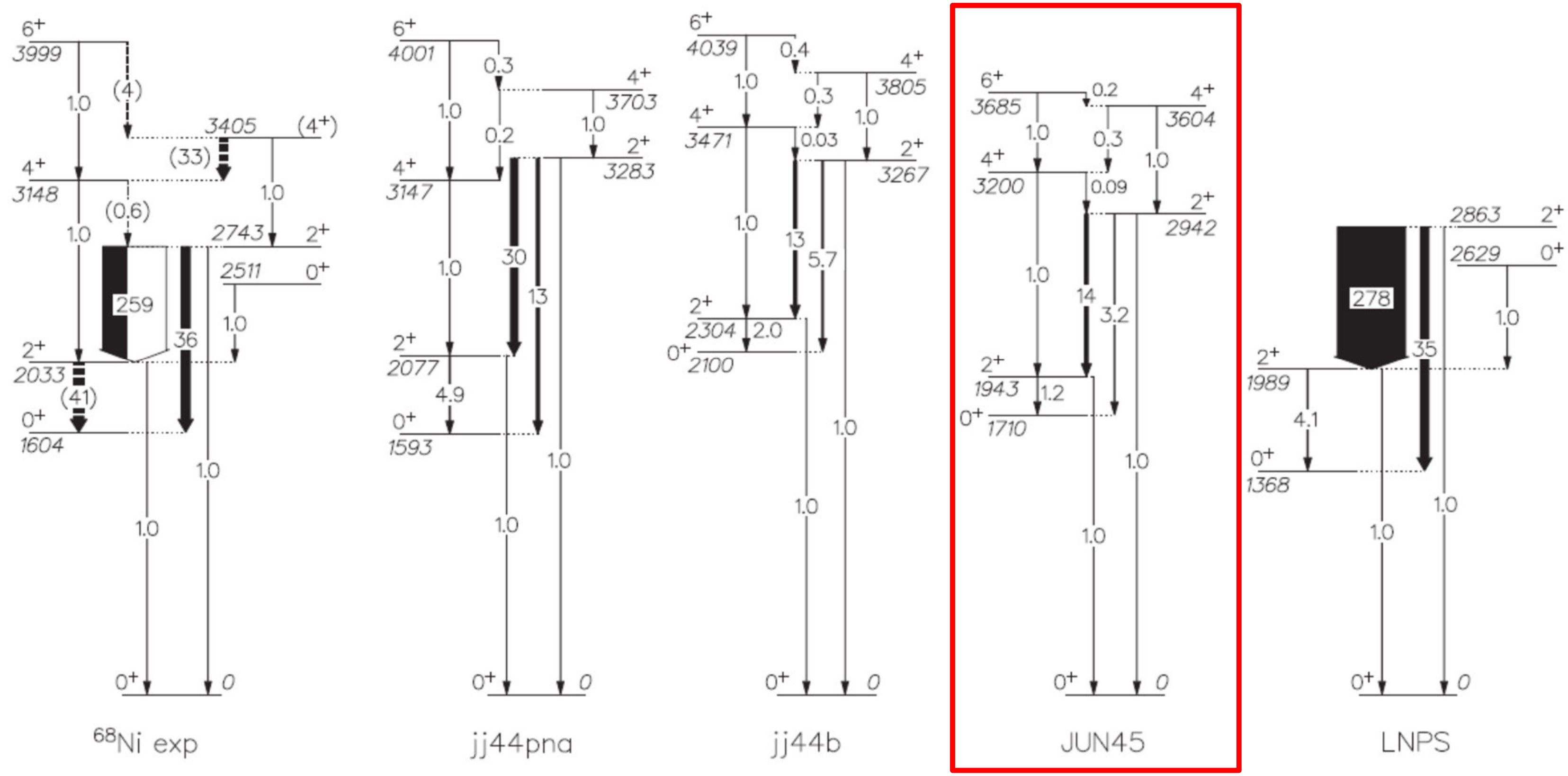
E. Caurier et al., PRL75 (1995) 2466



- make backbending plot of ^{48}Cr with GXPF1A
- Try speedup of computation
  - ... J-projection, particle-hole truncation

Exercise 2

^{68}Ni



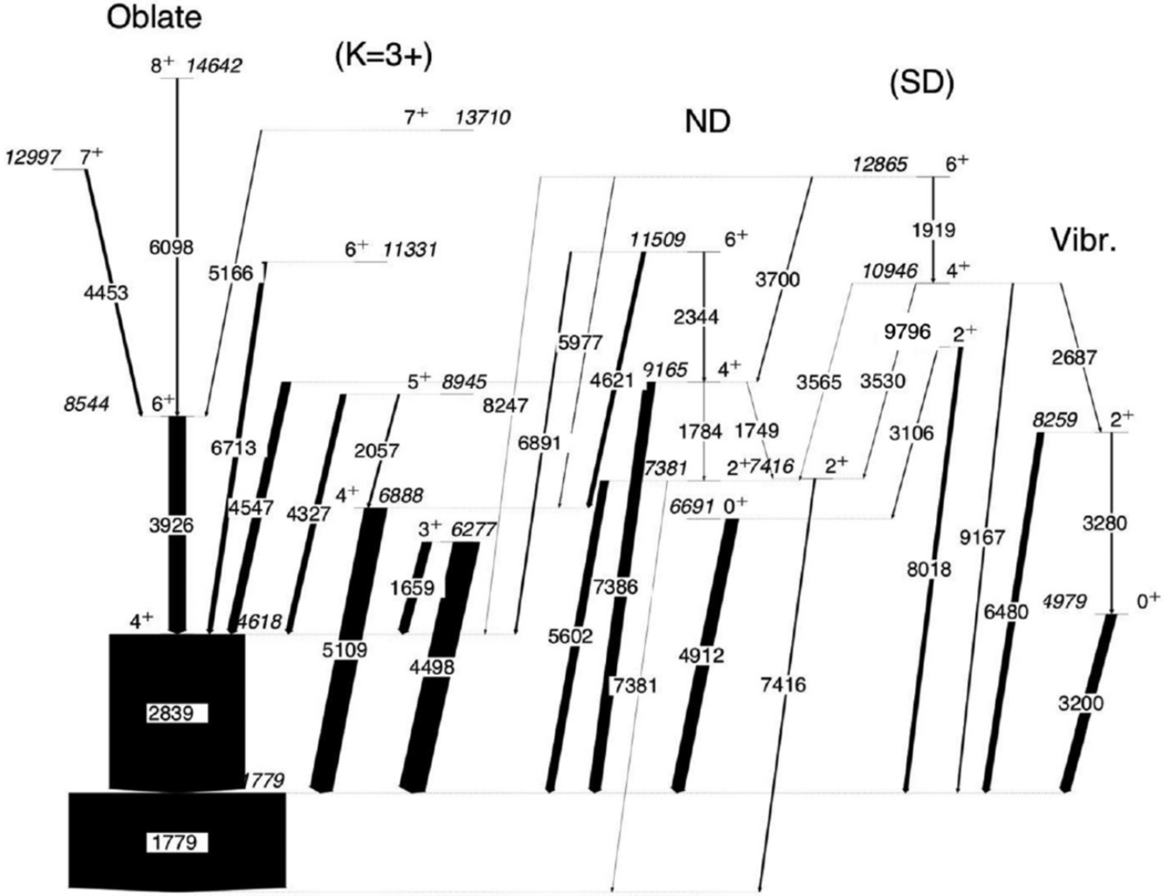
Calculate the energy levels of ^{68}Ni

B(E2) transition probabilities

Ref. F. Recchia *et al.,* Phys. Rev. C **88**, 041302(R) (2013)

Exercise 3

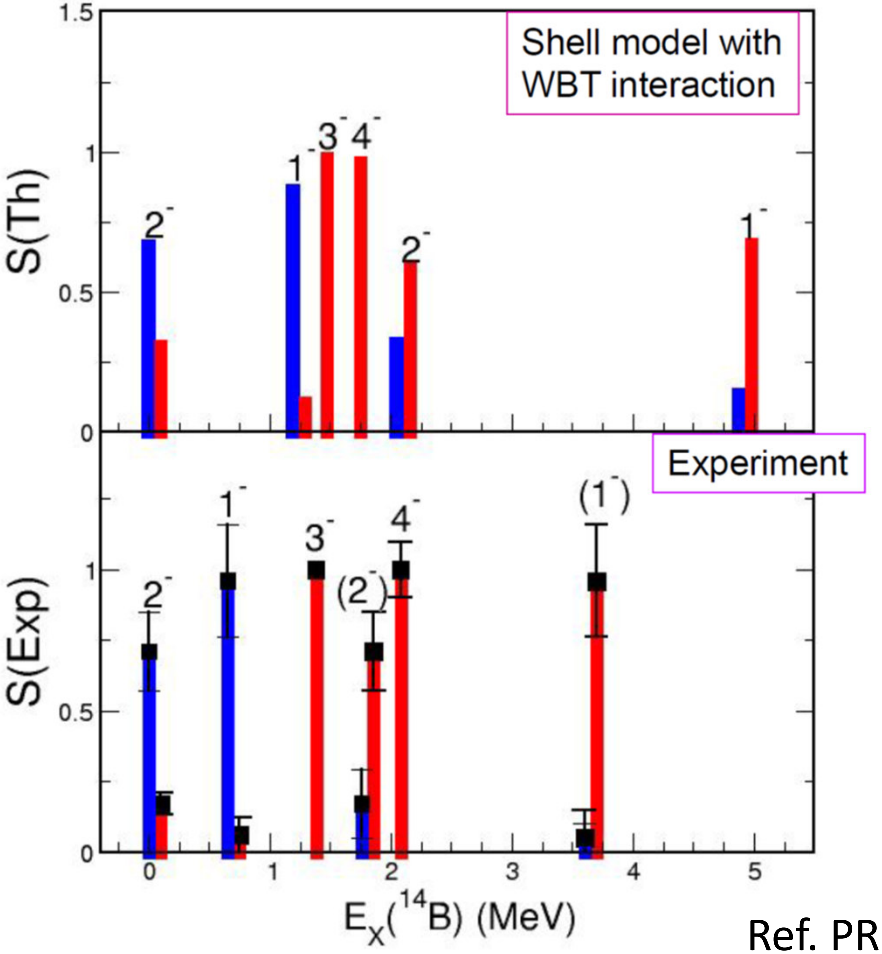
Perform LSSM calculations of 28Si and draw the E2 map. Discuss the comparison with the experimental levels.



D. G. Jenkins et al., Phys. Rev. C 86, 064308 (2012).

Exercise 4 (advanced)

Spectroscopic factors of $^{13}B(d,p)^{14}B$ reaction using “YSOX” interaction.



$^{13}B(d,p)^{14}B$ Spectroscopic factors

Excitation energies and relative spectroscopic factors from the shell model

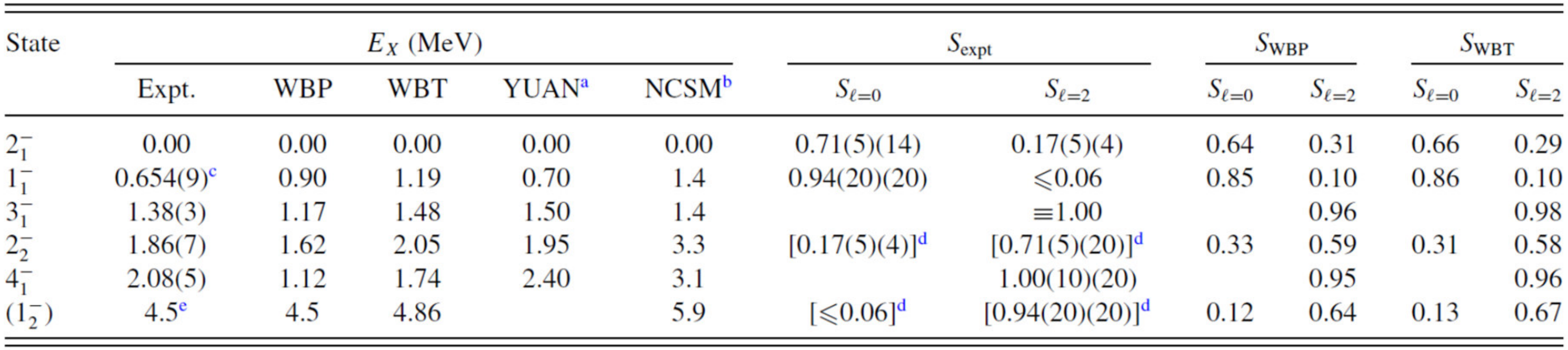
Blue: L=0, $1s_{1/2}$
Red: L=2, $0d_{5/2}$

Ref. PRC 88, 011304(R) (2013)

From 2nd lecture by Prof. Wuosmaa

Hints of Exercise 4

- Use “ysox.snt” as an interaction file.
- If computation time is too long, you can apply truncation scheme, e.g. up to $3\hbar\omega$ excitation
- At “kshell_ui.py” input to compute 13B 3/2- and 14B in a sequence. Then, “kshell_ui.py” will ask you “one-particle spectroscopic factor ? y/n”.



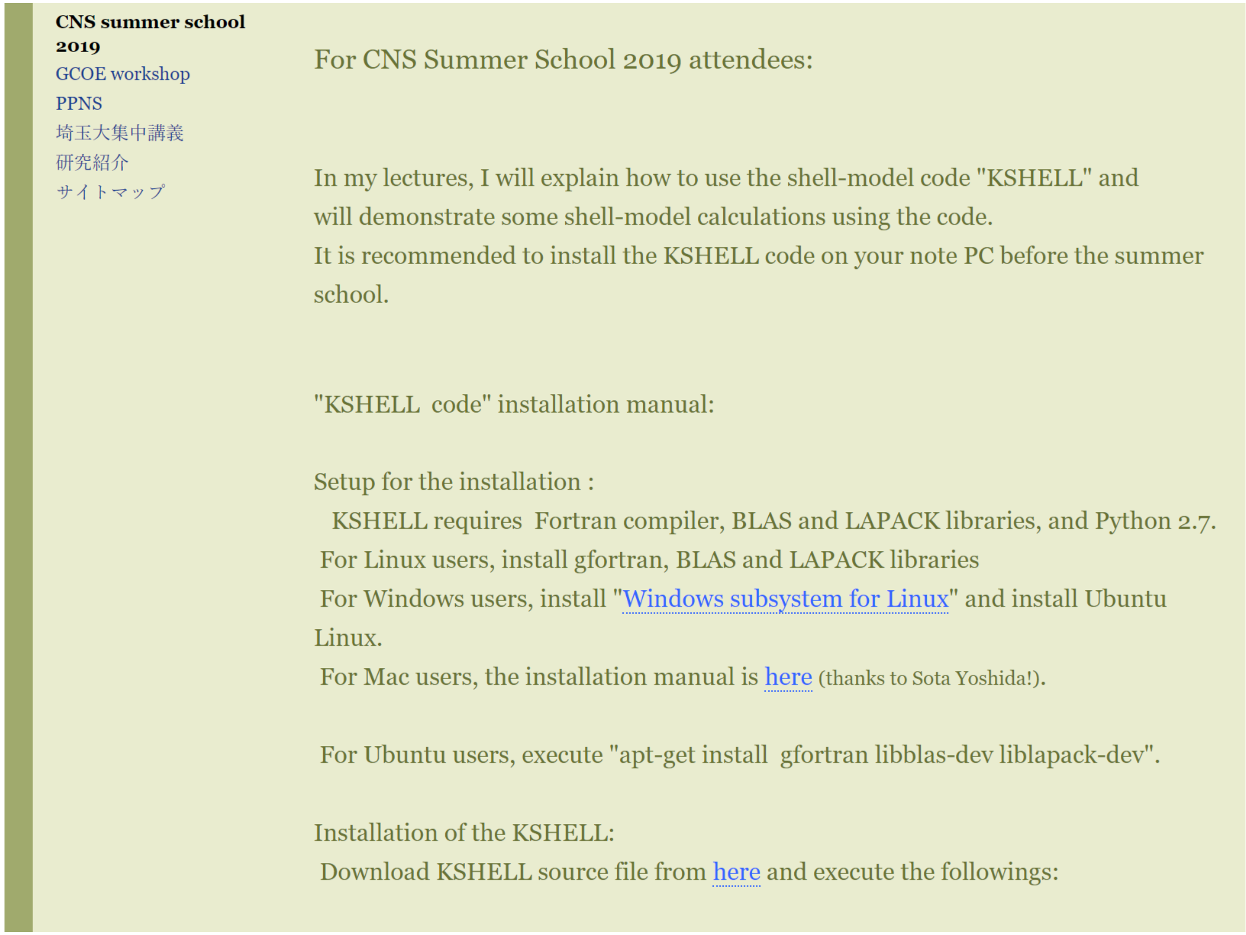
State	E_X (MeV)					S_{expt}		S_{WBP}		S_{WBT}	
	Expt.	WBP	WBT	YUAN[a]	NCSM[b]	$S_{\ell=0}$	$S_{\ell=2}$	$S_{\ell=0}$	$S_{\ell=2}$	$S_{\ell=0}$	$S_{\ell=2}$
2_1^-	0.00	0.00	0.00	0.00	0.00	0.71(5)(14)	0.17(5)(4)	0.64	0.31	0.66	0.29
1_1^-	0.654(9)[c]	0.90	1.19	0.70	1.4	0.94(20)(20)	$\leqslant$0.06	0.85	0.10	0.86	0.10
3_1^-	1.38(3)	1.17	1.48	1.50	1.4		$\equiv$1.00		0.96		0.98
2_2^-	1.86(7)	1.62	2.05	1.95	3.3	[0.17(5)(4)][d]	[0.71(5)(20)][d]	0.33	0.59	0.31	0.58
4_1^-	2.08(5)	1.12	1.74	2.40	3.1		1.00(10)(20)		0.95		0.96
(1_2^-)	4.5[e]	4.5	4.86		5.9	[$\leqslant$0.06][d]	[0.94(20)(20)][d]	0.12	0.64	0.13	0.67

S. Bedoor et al., Phys. Rev. C 88, 011304(R) (2013)

Try and have fun !

https://sites.google.com/a/cns.s.u-tokyo.ac.jp/shimizu/cns-summer-school-2019

Any questions and comments are welcome.
If you have problems, ask me or Yusuke Tsunoda-san during the school.



CNS summer school 2019
GCOE workshop
PPNS
埼玉大集中講義
研究紹介
サイトマップ

For CNS Summer School 2019 attendees:

In my lectures, I will explain how to use the shell-model code "KSHELL" and will demonstrate some shell-model calculations using the code.
It is recommended to install the KSHELL code on your note PC before the summer school.

"KSHELL code" installation manual:

Setup for the installation :
KSHELL requires Fortran compiler, BLAS and LAPACK libraries, and Python 2.7.
For Linux users, install gfortran, BLAS and LAPACK libraries
For Windows users, install "Windows subsystem for Linux" and install Ubuntu Linux.
For Mac users, the installation manual is here (thanks to Sota Yoshida!).

For Ubuntu users, execute "apt-get install gfortran libblas-dev liblapack-dev".

Installation of the KSHELL:
Download KSHELL source file from here and execute the followings:

shimizu@cns.s.u-tokyo.ac.jp

ytsunoda@cns.s.u-tokyo.ac.jp