Large scale configuration interaction calculations for nuclear structure

Calvin Johnson, Department of Physics *and*
Computational Science Research Center
San Diego State University
cjohnson @ mail . sdsu . edu

Collaborators:
W. Erich Ormand, Lawrence Livermore
Plamen G. Krastev, SDSU/Harvard
Ken McElvain, UC Berkeley/LBNL

C. W. Johnson, W. E. Ormand, and P. G. Krastev,
Comp. Phys. Comm. **184**, 2761-2774 (2013)

THE BASIC PROBLEM

The basic *science question* is to model detailed quantum structure of many-body systems, such the electronic structure of an atom, or structure of an atomic nucleus.

To answer this, we attempt to solve *Schrödinger's equation*:

$$\left(\sum_i -\frac{\hbar^2}{2m}\nabla^2 + U(r_i) + \sum_{i<j} V(\vec{r}_i - \vec{r}_j)\right)\Psi(\vec{r}_1, \vec{r}_2, \vec{r}_3 \ldots) = E\Psi$$

The Basic Problem

The basic *science question* is to model detailed quantum structure of many-body systems, such the electronic structure of an atom, or structure of an atomic nucleus.

This differential equation is too difficult to solve directly

$$\left(\sum_i -\frac{\hbar^2}{2m}\nabla^2 + U(r_i) + \sum_{i<j} V(\vec{r}_i - \vec{r}_j)\right)\Psi(\vec{r}_1, \vec{r}_2, \vec{r}_3 \ldots) = E\Psi$$

so we use the matrix formalism

$$\hat{\mathbf{H}}|\Psi\rangle = E|\Psi\rangle$$

The Basic Problem

The basic *science question* is to model detailed quantum structure of many-body systems, such the electronic structure of an atom, or structure of an atomic nucleus.

This differential equation is too difficult to solve directly

$$\left(\sum_i -\frac{\hbar^2}{2m}\nabla^2 + U(r_i) + \sum_{i<j} V(\vec{r}_i - \vec{r}_j)\right)\Psi(\vec{r}_1,\vec{r}_2,\vec{r}_3\ldots) = E\Psi$$

so we use the matrix formalism

$$\hat{\mathbf{H}}\left|\Psi\right\rangle = E\left|\Psi\right\rangle$$

Now the dimensions can be large, up to 2 x 10^{10}!

THE GOAL OF THIS TALK…

…is to get "under the hood" of a shell-model configuration-interaction code to stimulate discussion of efficient algorithms….

THE GOAL OF THIS TALK…

In particular I will compare

"matrix storage" codes:

-- more straightforward

-- requires more memory

vs. "on-the-fly" or "factorization" algorithms

-- uses less memory

-- more complex algorithmically

ANATOMY OF SHELL MODEL CODES

Basis: trade off between

"correlated" bases which

contains more correlations (physics)

and thus need "fewer" basis states

but are more complicated to handle

and lead to slower algorithm

e.g. states with good J ("J-scheme")

or with known physics such as deformation

or

"uncorrelated" bases which

are easier to handle -> fast algorithms

but need more states to build up correlations

e.g. Slater determinants with good M ("M-scheme")

Anatomy of Shell Model Codes

Basis:
trade off between
"correlated" bases which
contains more cor
but are more com
and lead to slowe
e.g. states with good J
or with known phy

It's not our task here to pass judgement as to which is better but to delineate strengths and weaknesses

or

"uncorrelated" bases which
are easier to handle -> fast algorithms
but need more states to build up correlati
e.g. Slater determinants with good M ("M-sch

ANATOMY OF SHELL MODEL CODES

Hamiltonian:
trade off between

"matrix storage" codes:

-- more straightforward

-- requires more memory

The more correlated the basis, the more this is indicated (especially since basis dimension is smaller)

vs. "on-the-fly" or "factorization" algorithms

-- uses less memory

-- more complex algorithmically

Works more effectively with less correlated bases

Some Shell-Model Codes

Matrix storage:

Oak Ridge-Rochester (small matrices)

Glasgow-Los Alamos (M-scheme, stored on disk; introduced Lanczos)

OXBASH /Oxford-MSU (J-scheme, stored on disk)

MFDn/ Iowa State (M-scheme, stored in RAM; plans for J-scheme, SU(3)-scheme w/LSU)

MCSM/ Tokyo (J-scheme from selected states)

Importance Truncation SM/Darmstadt (M-scheme from selected states)

Sym Adapted SM / LSU (J-scheme + symplectic; see T. Dytrych's talk)

Factorization:

ANTOINE Strasbourg (M-scheme; originator of factorization)

NATHAN Strasbourg (J-scheme)

EICODE (J-scheme)

NuShell/NuShellX (J-scheme)

MSHELL64 / KSHELL Tokyo (M-scheme)

REDSTICK+BIGSTICK/ LSU-SDSU-Livermore

THE KEY IDEAS

Basic problem: find extremal eigenvalues of very large, very sparse Hermitian matrix

Lanczos algorithm

fundamental operation is *matrix-vector multiply*

Despite sparsity, nonzero matrix elements can require TB of storage

Only a fraction of matrix elements are unique; **most are reused.**
Reuse of matrix elements understood through *spectator* particles.

Reuse can be **exploited using exact factorization** enforced through *additive/multiplicative quantum numbers*

The algorithms described today are best applied to many body systems with
(a)two "species" (protons and neutrons, or +1/2 and -1/2 electrons)
(b)single-particle basis states with good rotational symmetry (j, m)

The Basic Problem

Find extremal eigenvalues of very large, very sparse Hermitian matrix

Lanczos algorithm

fundamental operation is *matrix-vector multiply*

we use the matrix formalism

$$\hat{\mathbf{H}}|\Psi\rangle = E|\Psi\rangle$$

$$|\Psi\rangle = \sum_{\alpha} c_{\alpha}|\alpha\rangle \qquad H_{\alpha\beta} = \langle\alpha|\hat{\mathbf{H}}|\beta\rangle$$

$$\sum_{\beta} H_{\alpha\beta} c_{\beta} = E c_{\alpha} \quad \text{if} \quad \langle\alpha|\beta\rangle = \delta_{\alpha\beta}$$

THE BASIC PROBLEM

Find extremal eigenvalues of very large, very sparse Hermitian matrix

→ Lanczos algorithm

fundamental operation is *matrix-vector multiply*

$$H_{\alpha\beta} = \langle \alpha | \hat{\mathbf{H}} | \beta \rangle$$

* **H** is generally a very large matrix – dimensions up to 10^{10} have been tackled.
* **H** is generally very sparse.
* We usually only want a few low-lying states

→ Lanczos algorithm!

The Basic Problem

Find extremal eigenvalues of very large, very sparse Hermitian matrix

Lanczos algorithm

fundamental operation is *matrix-vector multiply*

$$\mathbf{A}\vec{v}_1 = \alpha_1\vec{v}_1 + \beta_1\vec{v}_2$$

$$\mathbf{A}\vec{v}_2 = \beta_1\vec{v}_1 + \alpha_2\vec{v}_2 + \beta_2\vec{v}_3$$

$$\mathbf{A}\vec{v}_3 = \beta_2\vec{v}_2 + \alpha_3\vec{v}_3 + \beta_3\vec{v}_4$$

$$\mathbf{A}\vec{v}_4 = \beta_3\vec{v}_3 + \alpha_4\vec{v}_4 + \beta_4\vec{v}_5$$

Lanczos algorithm!

THE BASIC PROBLEM

Find extremal eigenvalues of very large, very sparse Hermitian matrix

→ Lanczos algorithm

fundamental operation is *matrix-vector multiply*

$$\mathbf{A}\vec{v}_1 = \alpha_1\vec{v}_1 + \beta_1\vec{v}_2$$

$$\mathbf{A}\vec{v}_2 = \beta_1\vec{v}_1 + \alpha_2\vec{v}_2 + \beta_2\vec{v}_3$$

$$\mathbf{A}\vec{v}_3 = \beta_2\vec{v}_2 + \alpha_3\vec{v}_3 + \beta_3\vec{v}_4$$

$$\mathbf{A}\vec{v}_4 = \beta_3\vec{v}_3 + \alpha_4\vec{v}_4 + \beta_4\vec{v}_5$$

matrix-vector multiply

Lanczos algorithm!

The Basic Problem

I need to quickly cover:

- How the basis states are represented
- How the Hamiltonian operator is represented
- Why most matrix elements are zero
- Typical dimensions and sparsity

How to Build an M-scheme basis

(You probably already know this, my apologies! We might still learn something.)

How to Build an M-scheme basis

- How the basis states are represented

This differential equation is too difficult to solve directly

$$\left(\sum_i -\frac{\hbar^2}{2m}\nabla^2 + U(r_i) + \sum_{i<j} V(\vec{r}_i - \vec{r}_j)\right)\Psi(\vec{r}_1,\vec{r}_2,\vec{r}_3\ldots) = E\Psi$$

Can only really solve 1D differential equation

$$\left(-\frac{\hbar^2}{2m}\frac{d^2}{dr^2} + U(r)\right)\phi_i(r) = \varepsilon_i\phi_i(r)$$

Usually assume spherical symmetry!

How to Build an M-scheme basis

- How the basis states are represented

Can only really solve 1D differential equation

$$\left(-\frac{\hbar^2}{2m}\frac{d^2}{dr^2}+U(r)\right)\phi_i(r)=\varepsilon_i\phi_i(r) \Rightarrow \left\{\phi_i(\vec{r})\right\}$$

Single-particle wave functions labeled by, *e.g.*, n, j, l, m

Atomic case: 1s, 2s, 2p, 3s, 3p, 3d *etc*

Nuclear: $0s_{1/2}$, $0p_{3/2}$, $0p_{1/2}$, $0d_{5/2}$, $1s_{1/2}$, $0d_{3/2}$, *etc*

How to Build an M-scheme basis

- How the basis states are represented

Can only really solve 1D differential equation

$$\left(-\frac{\hbar^2}{2m}\frac{d^2}{dr^2}+U(r)\right)\phi_i(r)=\varepsilon_i\phi_i(r) \Rightarrow \left\{\phi_i(\vec{r})\right\}$$

Product wavefunction ("Slater Determinant")

$$\Psi(\vec{r}_1,\vec{r}_2,\vec{r}_3\ldots)=\phi_{n_1}(\vec{r}_1)\phi_{n_2}(\vec{r}_2)\phi_{n_3}(\vec{r}_3)\ldots\phi_{n_N}(\vec{r}_N)$$

How to Build an M-scheme basis

- How the basis states are represented

Product wavefunction ("Slater Determinant")

$$\Psi(\vec{r}_1,\vec{r}_2,\vec{r}_3\ldots) = \phi_{n_1}(\vec{r}_1)\phi_{n_2}(\vec{r}_2)\phi_{n_3}(\vec{r}_3)\ldots\phi_{n_N}(\vec{r}_N)$$

Each many-body state can be *uniquely* determined by a list of "occupied" single-particle states = "occupation representation"

$$|\alpha\rangle = \hat{a}^+_{n_1}\hat{a}^+_{n_2}\hat{a}^+_{n_3}\ldots\hat{a}^+_{n_N}|0\rangle$$

How to Build an M-scheme basis

- How the basis is represented

"occupation representation" $$|\alpha\rangle = \hat{a}^{+}_{n_1}\hat{a}^{+}_{n_2}\hat{a}^{+}_{n_3}\ldots\hat{a}^{+}_{n_N}|0\rangle$$

n_i	1	2	3	4	5	6	7
α=1	1	0	0	1	1	0	1
α=2	1	0	1	0	0	1	1
α=3	0	1	1	1	0	1	0

Convenient for digital computers!

How to Build an M-scheme basis

- How the basis is represented

some technical details:
the "M-scheme"

$$\left|\alpha\right\rangle = \hat{a}^{+}_{n_1}\hat{a}^{+}_{n_2}\hat{a}^{+}_{n_3}\ldots\hat{a}^{+}_{n_N}\left|0\right\rangle$$

Because J_z commutes with H, we can use a basis with M fixed = "M-scheme"

For any Slater determinant, the total M = sum of the m_l's, making construction of an M-scheme basis *easy*.

(In general, any J-scheme basis state is a sum of M-scheme states – or a projection integral which is also a sum)

A Sparse Matrix

- How the Hamiltonian is represented

"occupation representation" $\left|\alpha\right\rangle = \hat{a}^{+}_{n_1}\hat{a}^{+}_{n_2}\hat{a}^{+}_{n_3}\ldots\hat{a}^{+}_{n_N}\left|0\right\rangle$

n_i	1	2	3	4	5	6	7
α=1	1	0	0	1	1	0	1
α=2	1	0	1	0	0	1	1
α=3	0	1	1	1	0	1	0

$$\hat{H} = \sum_{ij} T_{ij}\hat{a}^{+}_{i}\hat{a}_{j} + \tfrac{1}{4}\sum_{ijkl} V_{ijkl}\hat{a}^{+}_{i}\hat{a}^{+}_{j}\hat{a}_{l}\hat{a}_{k}$$

A Sparse Matrix

- How the Hamiltonian is represented

"occupation representation" $\left|\alpha\right\rangle = \hat{a}^{+}_{n_1}\hat{a}^{+}_{n_2}\hat{a}^{+}_{n_3}\ldots\hat{a}^{+}_{n_N}\left|0\right\rangle$

n_i	1	2	3	4	5	6	7
α=1	1	0	0	1	1	0	1
α=2	1	0	1	0	0	1	1
α=3	0	1	1	1	0	1	0

$$\hat{a}^{+}_{3}\hat{a}^{+}_{6}\hat{a}_{4}\hat{a}_{5}\left|\alpha=1\right\rangle = \left|\alpha=2\right\rangle$$

A Sparse Matrix

- How the Hamiltonian is represented

"occupation representation" $\left|\alpha\right\rangle = \hat{a}^{+}_{n_1}\hat{a}^{+}_{n_2}\hat{a}^{+}_{n_3}\ldots\hat{a}^{+}_{n_N}\left|0\right\rangle$

n_i	1	2	3	4	5	6	7
α=1	1	0	0	1	1	0	1
α=2	1	0	1	0	0	1	1
α=3	0	1	1	1	0	1	0

$$\hat{a}^{+}_{2}\hat{a}^{+}_{4}\hat{a}_{1}\hat{a}_{7}\left|\alpha=2\right\rangle = \left|\alpha=3\right\rangle$$

A Sparse Matrix

- Why most matrix elements are zero

"occupation representation" $\left|\alpha\right\rangle = \hat{a}^{+}_{n_1}\hat{a}^{+}_{n_2}\hat{a}^{+}_{n_3}\ldots\hat{a}^{+}_{n_N}\left|0\right\rangle$

n_i	1	2	3	4	5	6	7
α=1	1	0	0	1	1	0	1
α=2	1	0	1	0	0	1	1
α=3	0	1	1	1	0	1	0

$$\hat{a}^{+}_{2}\hat{a}^{+}_{4}\hat{a}^{+}_{6}\hat{a}_{1}\hat{a}_{5}\hat{a}_{7}\left|\alpha=1\right\rangle = \left|\alpha=3\right\rangle$$

need 3 particles to interact simultaneously!

A Sparse Matrix

- Typical dimensions and sparsity

Nuclide	valence space	valence Z	valence N	basis dim	sparsity (%)
^{20}Ne	"sd"	2	2	640	10
^{25}Mg	"sd"	4	5	44,133	0.5
^{49}Cr	"pf"	4	5	6M	0.01
^{56}Fe	"pf"	6	10	500M	2×10^{-4}

This corresponds to 2 Tb of data!

A Problem….

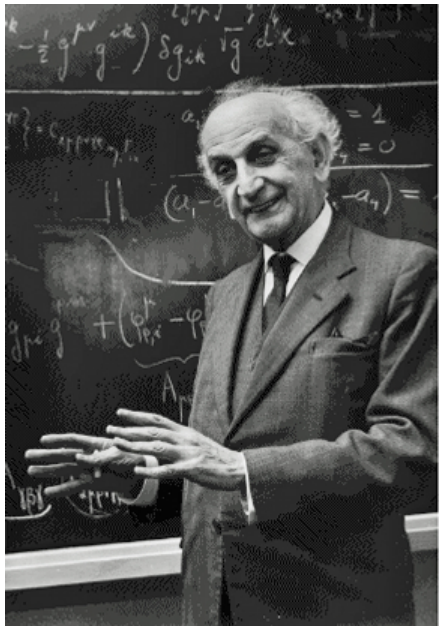
Despite sparsity, nonzero matrix elements can require TB of storage

Nuclide	Space	Basis dim	matrix store
^{56}Fe	*pf*	501 M	3.5 Tb
^{7}Li	N_{max}=12	252 M	3.6 Tb
^{7}Li	N_{max}=14	1200 M	23 Tb
^{12}C	N_{max}=6	32M	0.2 Tb
^{12}C	N_{max}=8	590M	5 Tb
^{12}C	N_{max}=10	7800M	111 Tb
^{16}O	N_{max}=6	26 M	0.14 Tb
^{16}O	N_{max}=8	990 M	9.7 Tb

Spread nonzero matrix elements over many MPI compute nodes:

A Sparse Matrix, but....

Despite sparsity, nonzero matrix elements can require TB of storage



(Cornelius Lanczos)

My algorithm is ideal as one can use sparse matrix-vector multiplication

That's true, but there is more to the story…

Recycled Matrix Elements

Only a fraction of matrix elements are unique; **most are reused.**

Reuse of matrix elements understood through *spectator* particles.

n_i	1	2	3	4	5	6	7	8
α=1	1	1	1	0	0	0	0	1
α=2	1	1	0	1	1	0	0	0
α=3	0	1	1	0	0	1	0	1
α=4	0	1	0	1	1	1	0	0
α=5	0	0	1	0	0	1	1	1
α=6	0	0	0	1	1	1	1	0

All of these have the same matrix element: V_{4538}

Recycled Matrix Elements

Only a fraction of matrix elements are unique; **most are reused.**

Reuse of matrix elements understood through *spectator* particles.

# of nonzero matrix elements vs. # unique matrix elements

Nuclide	valence space	valence Z	valence N	# nonzero	# unique
^{28}Si	"sd"	6	6	26 x 10^6	3600
^{52}Fe	"pf"	6	6	90 x 10^9	21,500

Atom	space	# nonzero	# unique
Be	CVB3	110x10^6	521,000
B	CVB2	1.4x10^9	379,000
C	CVB1	260x10^6	40,751

FACTORIZATION

Reuse can be **exploited using exact factorization**
enforced through *additive/multiplicative quantum numbers*

A quantum number is the eigenvalue of an operator

For composite systems, one can apply the operator to each component separately:

$$\hat{O}|\Psi\rangle = \left(\hat{O}_1 + \hat{O}_2 + \hat{O}_3 + \ldots\right)\left(|\Psi_1\rangle \otimes |\Psi_2\rangle \otimes |\Psi_3\rangle \otimes \ldots\right)$$

Sometimes the total quantum number is a simple sum/product as is the case for $\mathbf{J}_z$ or parity....

$$\hat{J}_z|\Psi\rangle = M|\Psi\rangle = (m_1 + m_2 + m_3 + \ldots)|\Psi\rangle$$

...but in other cases the addition is complicated (e.g. for $\mathbf{J}^2$)

FACTORIZATION

Reuse can be **exploited using exact factorization**
enforced through *additive/multiplicative quantum numbers*

I consider composite many-fermion systems,
in particular those with 2 major components
protons and *neutrons*
or
spin-up and *spin-down* electrons

$$\left|\Psi\right\rangle = \left|\Psi_1\right\rangle \otimes \left|\Psi_2\right\rangle$$

Each component itself is a Slater determinant which is composed of many particles

$$\hat{J}_z\left|\Psi\right\rangle = M\left|\Psi\right\rangle \qquad M = M_1 + M_2$$

$$M_1 = m_1^{(1)} + m_1^{(2)} + m_1^{(2)} + \ldots$$

FACTORIZATION

Reuse can be **exploited using exact factorization**
enforced through *additive/multiplicative quantum numbers*

Because the M values are discrete integers or half-integers (-3, -2, -1, 0, 1, 2, ... or -3/2, -1/2, +1/2, +3/2....) we can organize the basis states in discrete *sectors*

Example: 2 protons, 4 neutrons, total M = 0

$M_z(\pi) = -4$	$M_z(\nu) = +4$
$M_z(\pi) = -3$	$M_z(\nu) = +3$
$M_z(\pi) = -2$	$M_z(\nu) = +2$

FACTORIZATION

Reuse can be **exploited using exact factorization**
enforced through *additive/multiplicative quantum numbers*

In fact, we can see an example of factorization here because all proton Slater determinants in one M-sector *must* combine with all the conjugate neutron Slater determinants

Example: 2 protons, 4 neutrons, total M = 0

$M_z(\pi) = -4$: 2 SDs	$M_z(\nu) = +4$: 24 SDs	48 combined
$M_z(\pi) = -3$: 4 SDs	$M_z(\nu) = +3$: 39 SDs	156 combined
$M_z(\pi) = -2$: 9 SDs	$M_z(\nu) = +2$: 60 SDs	540 combined

FACTORIZATION

Reuse can be **exploited using exact factorization**
enforced through *additive/multiplicative quantum numbers*

In fact, we can see an example of factorization here because all proton Slater determinants in one M-sector *must* combine with all the conjugate neutron Slater determinants

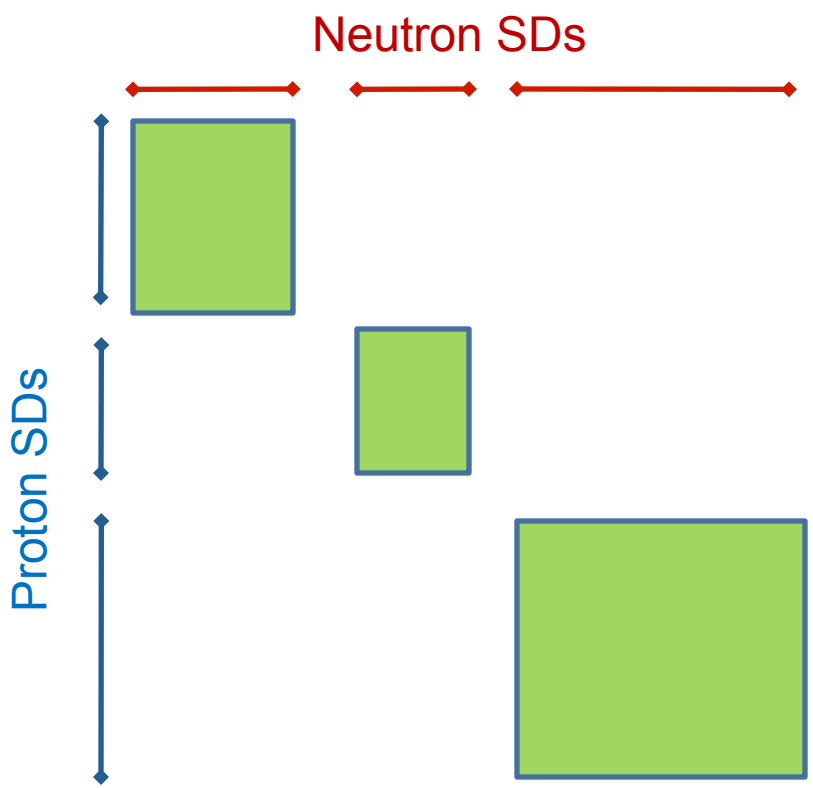
$M_z(\pi) = -4$: 2 SDs	$M_z(\nu) = +4$: 24 SDs	48 combined

$$\begin{matrix} |\pi_1\rangle \\ |\pi_2\rangle \end{matrix} \quad \times \quad \begin{matrix} |\nu_1\rangle \\ |\nu_2\rangle \\ |\nu_3\rangle \\ |\nu_4\rangle \\ \vdots \\ |\nu_{24}\rangle \end{matrix} \quad = \quad \begin{matrix} |\pi_1\rangle|\nu_1\rangle \\ |\pi_2\rangle|\nu_1\rangle \\ |\pi_1\rangle|\nu_2\rangle \\ |\pi_2\rangle|\nu_2\rangle \\ \vdots \\ |\pi_1\rangle|\nu_{24}\rangle \\ |\pi_2\rangle|\nu_{24}\rangle \end{matrix}$$

FACTORIZATION

Reuse can be **exploited using exact factorization** enforced through *additive/multiplicative quantum numbers*

$$|\alpha\rangle = |\alpha_p\rangle \times |\alpha_n\rangle$$



Example N = Z nuclei

Nuclide	Basis dim	# pSDs (=#nSDs)
^{20}Ne	640	66
^{24}Mg	28,503	495
^{28}Si	93,710	924
^{48}Cr	1,963,461	4895
^{52}Fe	109,954,620	38,760
^{56}Ni	1,087,455,228	125,970

FACTORIZATION

Reuse can be **exploited using exact factorization**
enforced through *additive/multiplicative quantum numbers*

Factorization allows us to keep track of all basis states without writing out every one explicitly
-- we only need to write down the proton/neutron components

The same trick can be applied to matrix-vector multiply

$$\hat{H} = \hat{H}_{pp} + \hat{H}_{nn} + \hat{H}_{pn}$$

Move 2 protons; *neutrons are spectators*

Move 2 neutrons; *protons are spectators*

Move 1 proton + 1 neutron; *rest are spectators*

FACTORIZATION

Reuse can be **exploited using exact factorization**
enforced through *additive/multiplicative quantum numbers*

$\hat{H}_{pp}$

Move 2 protons;
neutrons are spectators

Example: 2 protons, 4 neutrons, total M = 0

$M_z(\pi) = -4$: 2 SDs	$M_z(\nu) = +4$: 24 SDs	48 combined

There are potentially 48 × 48 matrix elements
But for H_{pp} at most 4 × 24 are nonzero
and we only have to look up 4 matrix elements

Advantage: **we can store 98 matrix elements as 4 matrix elements**
and avoid 2000+ zero matrix elements.

FACTORIZATION

Reuse can be **exploited using exact factorization** enforced through *additive/multiplicative quantum numbers*

$M_z(\pi) = -4$: 2 SDs

$M_z(\nu) = +4$: 24 SDs

48 combined

$$|\pi_1\rangle \quad |\pi_2\rangle \qquad H_{pp} = \begin{pmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{pmatrix} \qquad \begin{matrix} |\nu_1\rangle \\ |\nu_2\rangle \\ |\nu_3\rangle \\ |\nu_4\rangle \\ \vdots \\ |\nu_{24}\rangle \end{matrix}$$

Advantage: **we can store 98 matrix elements as 4 matrix elements** and avoid 2000+ zero matrix elements.

FACTORIZATION

Reuse can be **exploited using exact factorization** enforced through *additive/multiplicative quantum numbers*

$M_z(\pi) = -4$: 2 SDs

$M_z(\nu) = +4$: 24 SDs

48 combined

$$|\pi_1\rangle \qquad |\pi_2\rangle \qquad H_{pp} = \begin{pmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{pmatrix}$$

$$|\nu_1\rangle \qquad |\nu_2\rangle \qquad |\nu_3\rangle \qquad |\nu_4\rangle \qquad \vdots \qquad |\nu_{24}\rangle$$

$$H_{pp}|\pi_1\rangle|\nu_1\rangle = H_{11}|\pi_1\rangle|\nu_1\rangle + H_{12}|\pi_2\rangle|\nu_1\rangle$$
$$H_{pp}|\pi_2\rangle|\nu_1\rangle = H_{12}|\pi_1\rangle|\nu_1\rangle + H_{22}|\pi_2\rangle|\nu_1\rangle$$
$$H_{pp}|\pi_1\rangle|\nu_2\rangle = H_{11}|\pi_1\rangle|\nu_2\rangle + H_{12}|\pi_2\rangle|\nu_2\rangle$$
$$H_{pp}|\pi_2\rangle|\nu_2\rangle = H_{12}|\pi_1\rangle|\nu_2\rangle + H_{22}|\pi_2\rangle|\nu_2\rangle$$
$$\vdots$$
$$H_{pp}|\pi_1\rangle|\nu_{24}\rangle = H_{11}|\pi_1\rangle|\nu_{24}\rangle + H_{12}|\pi_2\rangle|\nu_{24}\rangle$$
$$H_{pp}|\pi_2\rangle|\nu_{24}\rangle = H_{12}|\pi_1\rangle|\nu_{24}\rangle + H_{22}|\pi_2\rangle|\nu_{24}\rangle$$

Advantage: **we can store 98 matrix elements as 4 matrix elements** and avoid 2000+ zero matrix elements.

FACTORIZATION

Reuse can be **exploited using exact factorization** enforced through *additive/multiplicative quantum numbers*

Comparison of nonzero matrix storage with factorization

Nuclide	Space	Basis dim	matrix store	factorization
^{56}Fe	*pf*	501 M	3500 Gb	0.72 Gb
^{7}Li	N_{max}=12	252 M	3800 Gb	61 Gb
^{7}Li	N_{max}=14	1200 M	23 Tb	624 Gb
^{12}C	N_{max}=6	32M	196 Gb	3.3 Gb
^{12}C	N_{max}=8	590M	5000 Gb	65 Gb
^{12}C	N_{max}=10	7800M	111 Tb	1.4 Tb
^{16}O	N_{max}=6	26 M	142 Gb	3.0 Gb
^{16}O	N_{max}=8	990 M	9700 Gb	130 Gb

Comparison of nonzero matrix storage with factorization

^{7}Li

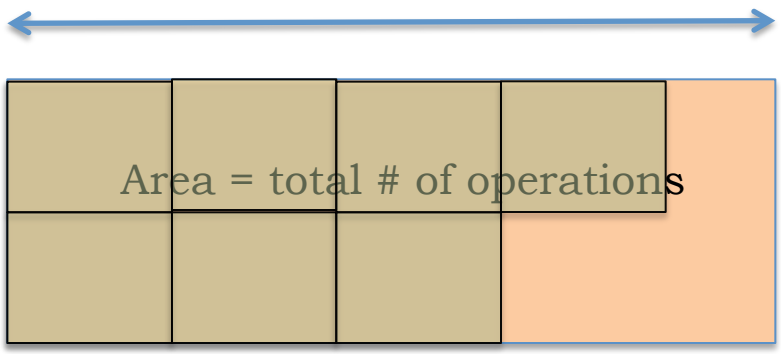
Space	Basis dim	matrix store (2-body)	factorization (2-body)	matrix store (3-body)	factorization (3-body)
N_{max}=8	6 M	36 Gb	1.5 Gb	1 Tb	26 Gb
N_{max}=10	43 M	430 Gb	10 Gb	170 Tb	250 Gb
N_{max}=12	250 M	4 Tb	60 Gb		

Space	Basis dim	matrix store (2-body)	factorization (2-body)	matrix store (3-body)	factorization (3-body)
N_{shell}=3	0.4 M	0.8 Gb	6 Mb	10 Gb	44 Mb
N_{shell}=4	45 M	330 Gb	0.3 Gb	9 Tb	4 Gb
N_{shell}=5	2 G	38 Tb	16 Gb	2 Pb	140 Gb
N_{shell}=6	50 G	2 Pb	87 Gb	170 Pb	3 Tb

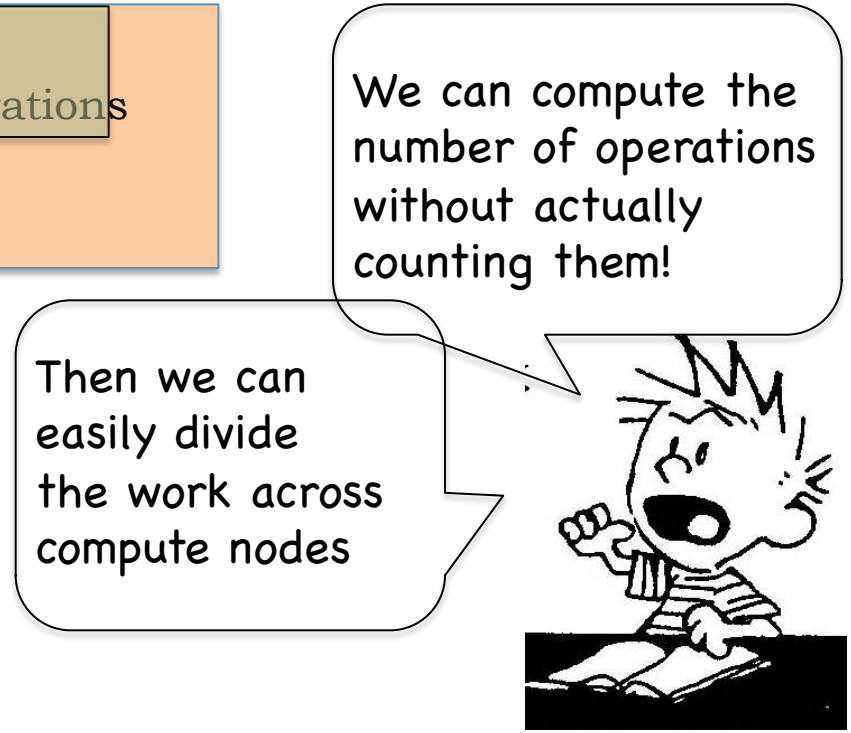
Parallel Implementation

Factorization makes it easier to compute workload and distribute across multiple nodes

length of sides = information to be stored



length of sides = information to be stored



EXECUTIVE SUMMARY ON THE BIGSTICK CODE

Many-fermion code: 2nd generation after REDSTICK code
(started in *Baton Rouge, La.*)

Uses "factorization" algorithm: Johnson, Ormand, and Krastev,
Comp. Phys. Comm. **184**, 2761(2013)

Arbitrary single-particle radial waveforms
Allows local or nonlocal two-body interaction
Three-body forces implemented and validated
Applies to both nuclear and atomic cases

Runs on both desktop and parallel machines
--can run at least dimension 300M+ on desktop
--has done *dimension 20 billion*+ on supercomputers

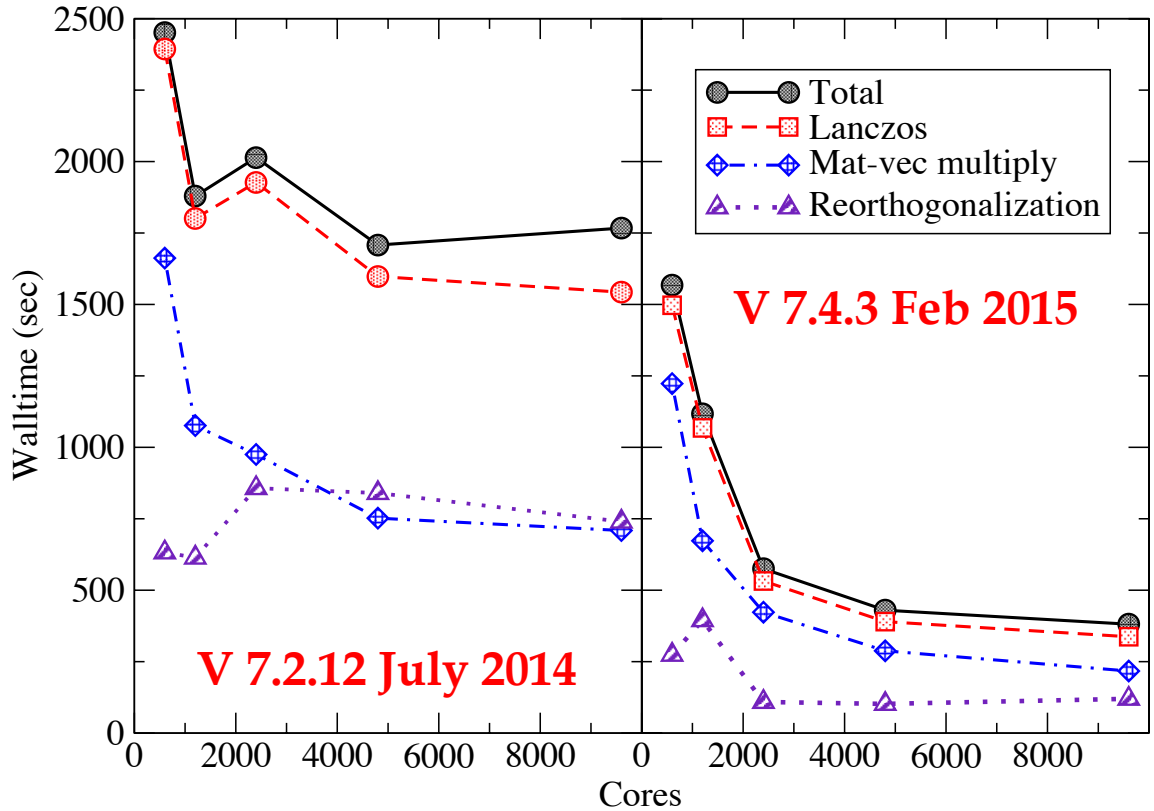
Inline calculations of one-body density matrices, single-particle occupations,
(+ options to compute strength functions via Lanczos trick, etc.)
Will add 2-body non-scalar transition operators later this year

45 kilolines of code Fortran 90 + MPI + OpenMP

Parallel Implementation – Latest developments

Over the past year we have dramatically improved our parallel performance (mostly through better use of MPI) due to Ken McElvain, UC Berkeley grad student

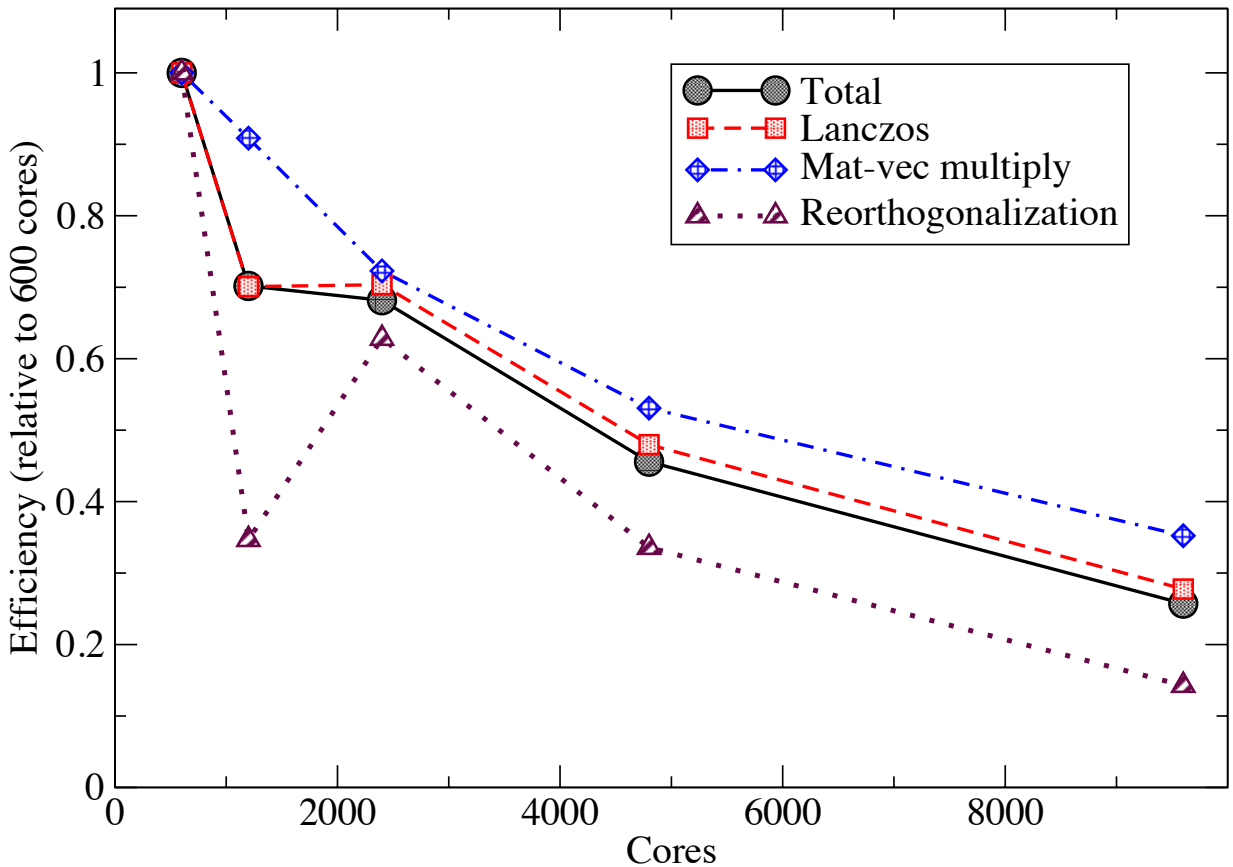
^{54}Mn in *pf* shell
(dim = 187 M)
200 iterations



LLNL Sierra

Parallel Implementation – Latest developments

Strong scaling

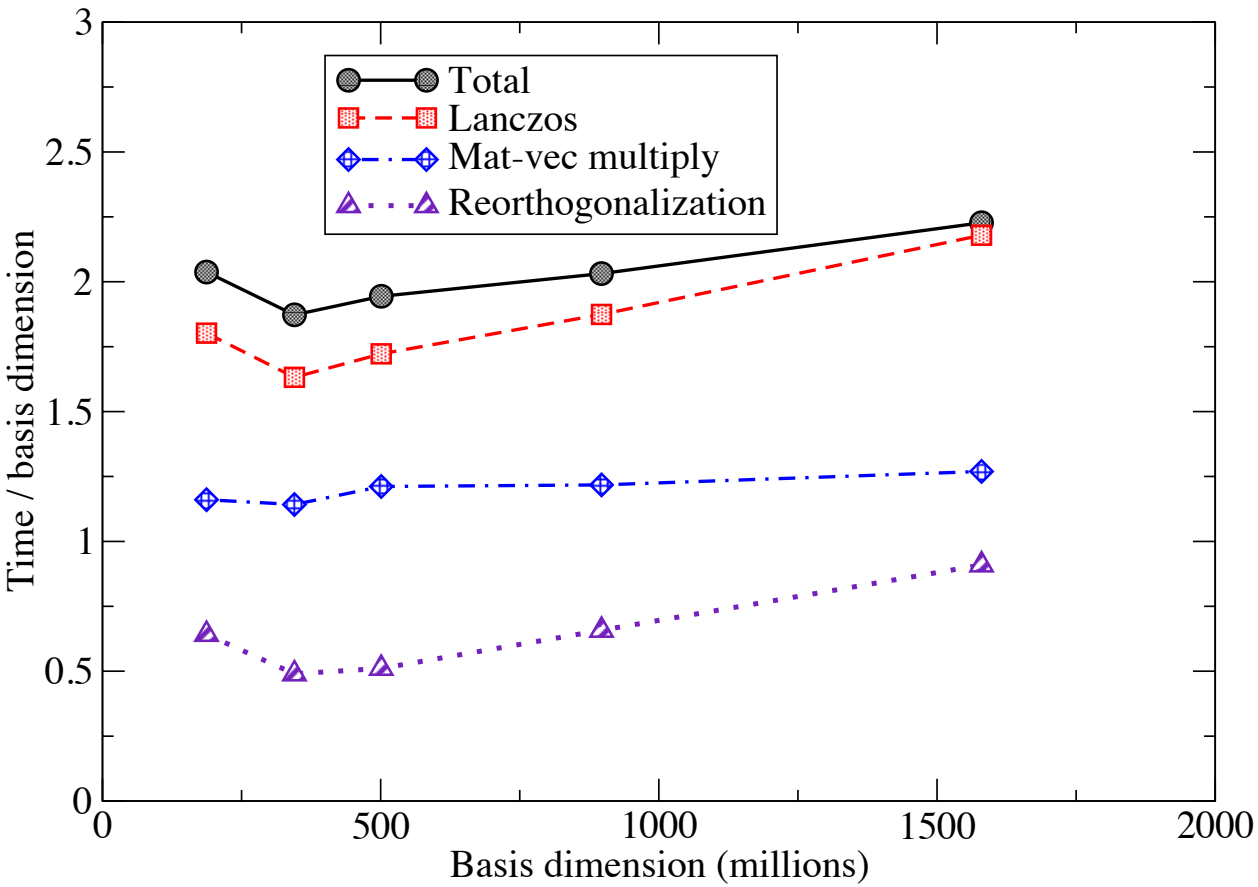


^{54}Mn in *pf* shell
(dim = 187 M)
200 iterations

LLNL Sierra

V 7.4.3 Feb 2015

Parallel Implementation – Latest developments

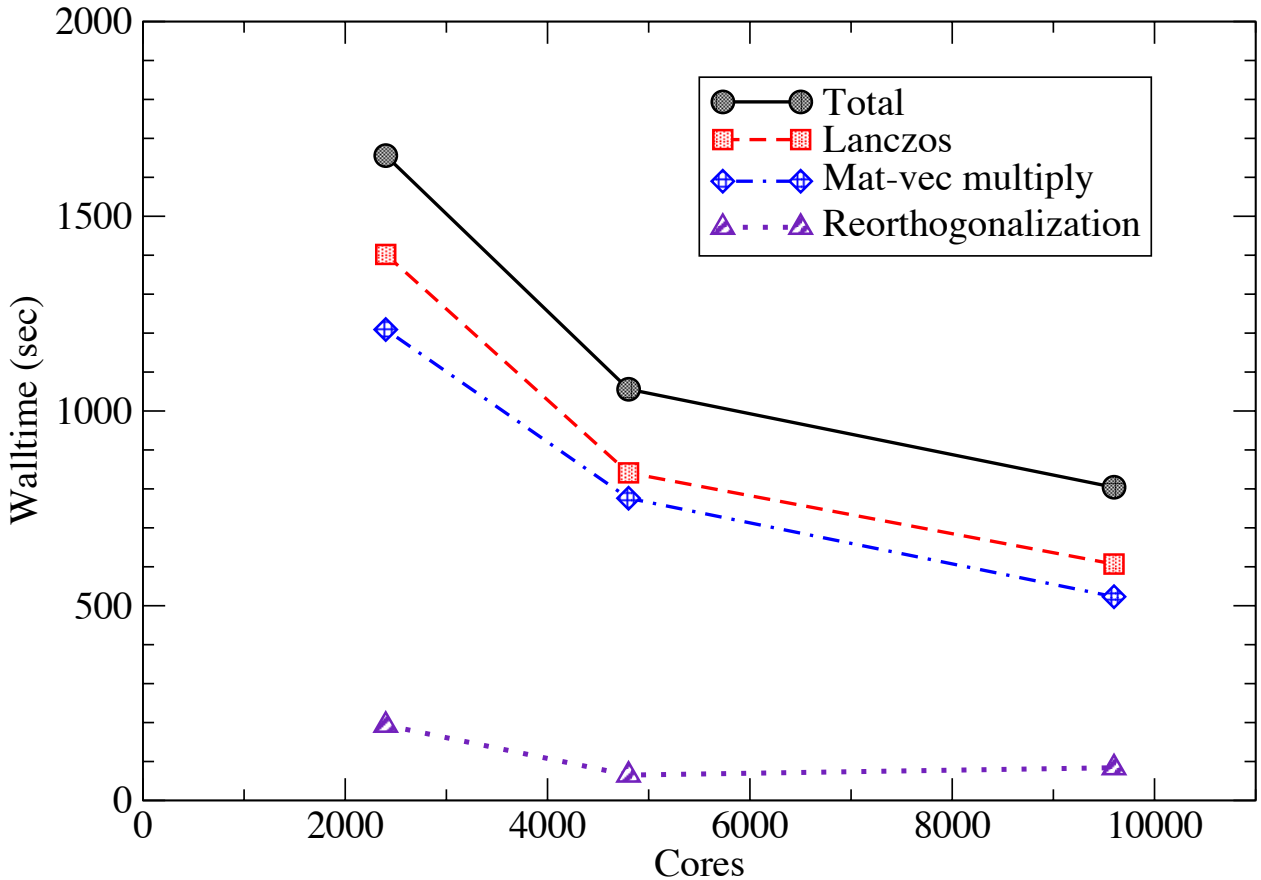


pf shell nuclides
(200 iterations)
800 MPI procs x 12 OpenMP threads

LLNL Sierra

V 7.4.3 Feb 2015

Parallel Implementation – Latest developments



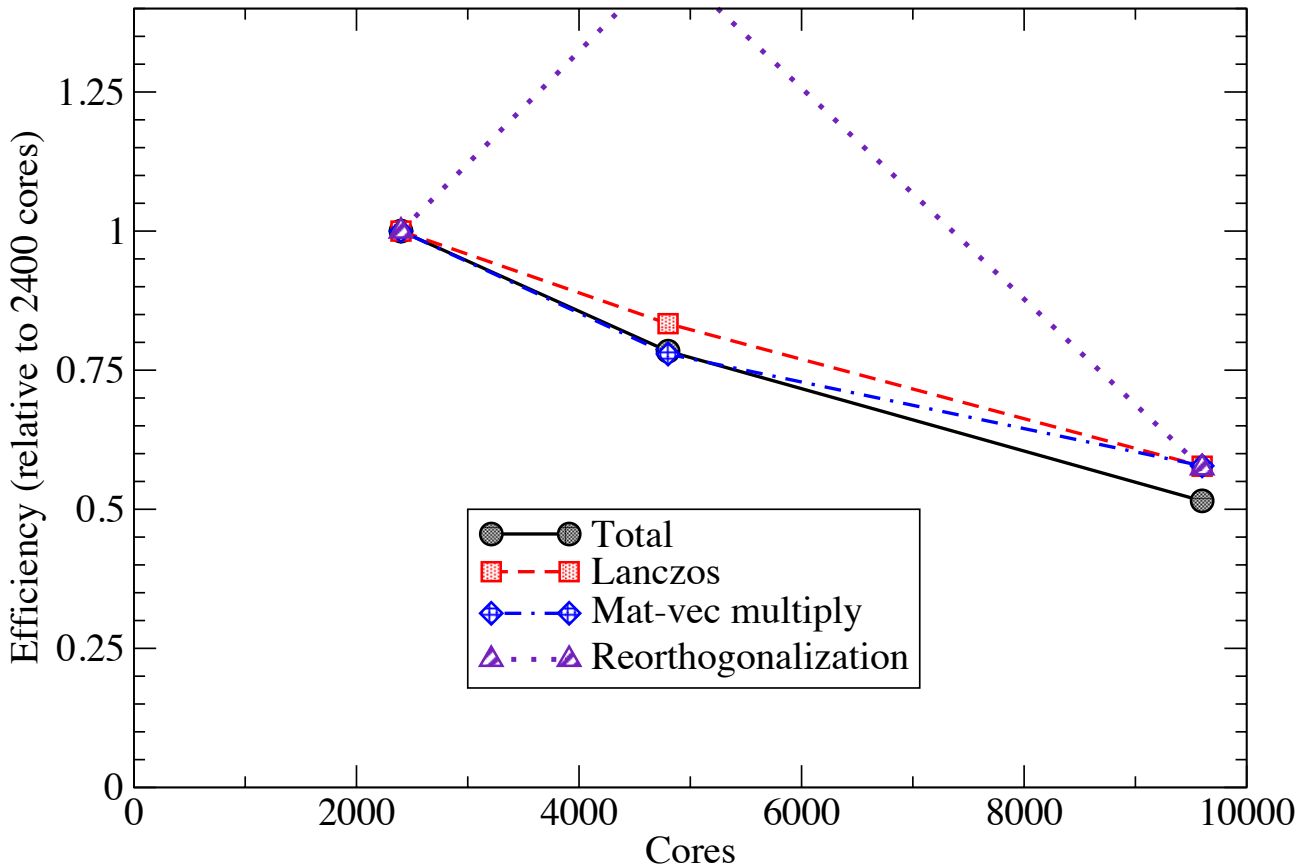
NCSM ^{10}B, N_{max} = 9 (dim = 547 M)
(100 iterations)
800 MPI procs x 12 OpenMP threads

LLNL Sierra

V 7.4.3 Feb 2015

Parallel Implementation – Latest developments

Strong scaling

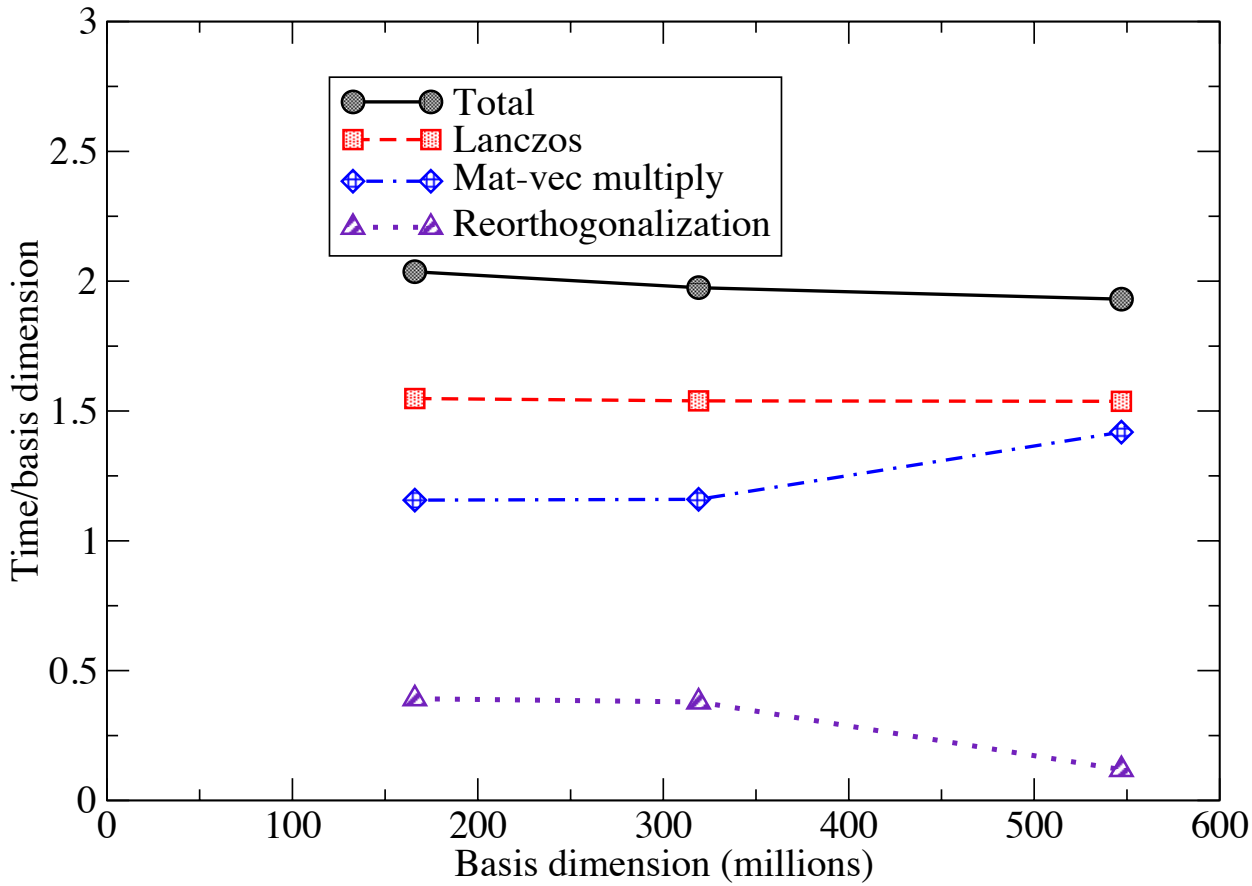


NCSM ^{10}B, N_{max} = 9 (dim = 547 M)
(100 iterations)
800 MPI procs x 12 OpenMP threads

LLNL Sierra

V 7.4.3 Feb 2015

Parallel Implementation – Latest developments



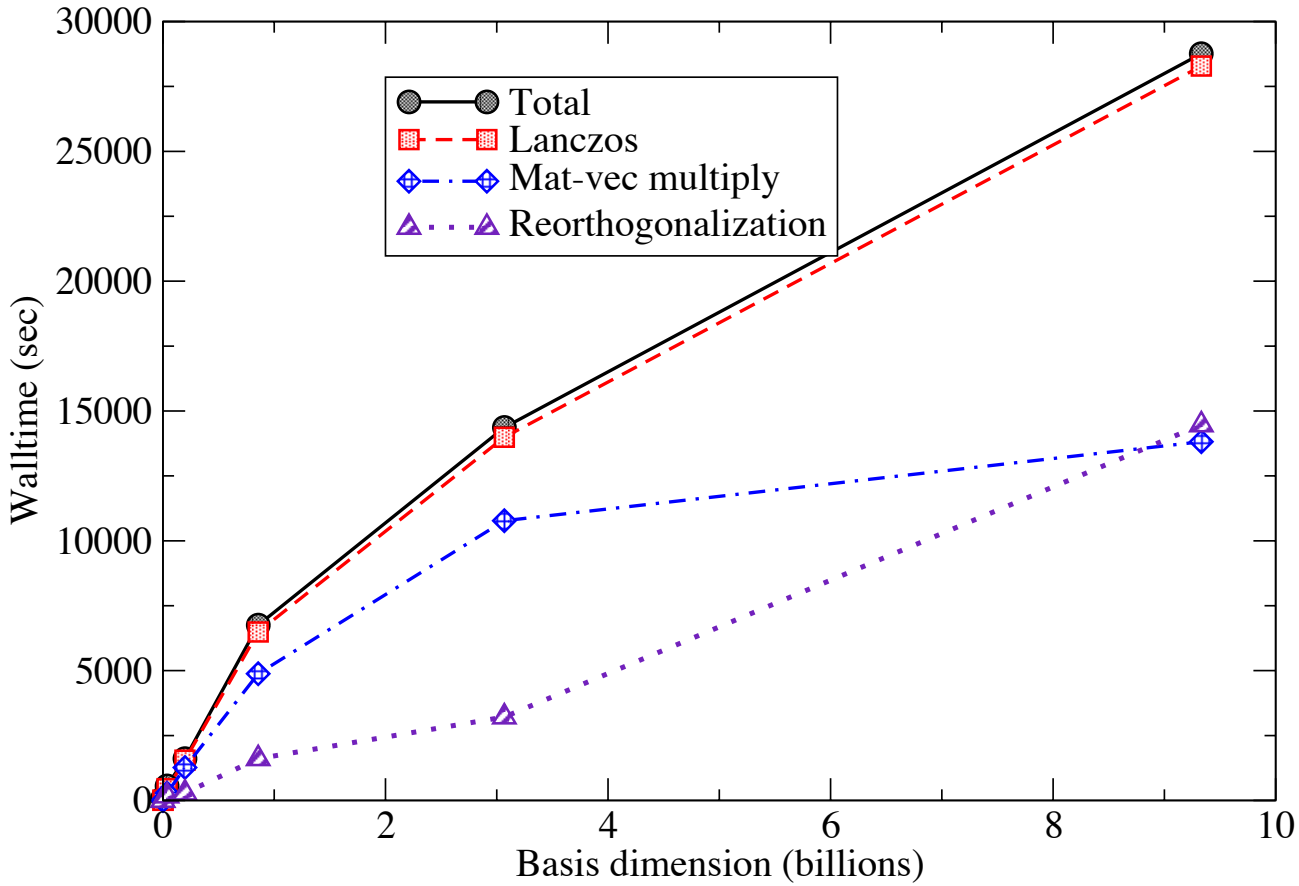
NCSM *p*-shell nuclides
(100 iterations)
800 MPI procs x 12 OpenMP threads

LLNL Sierra

V 7.4.3 Feb 2015

Parallel Implementation – Latest developments

Science runs! Dark matter scattering cross-sections



Xe isotopes with ^{100}Sn core
(140-250 iterations)
6000-12000 MPI procs x 4-6 OpenMP threads

LBL/NERSC Edison

V 7.4.3 Feb 2015

Recent Work

Pushing to larger cases

We have gone to dim 20 billion on 512 MPI nodes!

^{112}Ba with ^{100}Sn core: $2s_{1/2}$-$1d_{3/2}$-$1d_{5/2}$-$0g_{7/2}$-$0h_{11/2}$ valence space
LLNL Sierra 512 MPI processes with 24 Gb & 12 OpenMP threads/proc
2 Lanczos iterations took < 1 hr

Nonzero matrix elements require ~ 130 Tb = 5400 nodes

We plan (hope?) to go to dim ~ 100 billion in the next year

Advanced topic in factorization: Even *more* divide and conquer!

In general shell-model configuration interaction codes have three components:

- Set-up
- Matrix-vector multiply
- Linear algebra (reorthogonalization) (Lanczos algorithm)

The Lanczos part has fixed costs due to floating point operations and one can only distribute the work efficiently

The set-up can be expensive; in MFDn and related codes it takes a large fraction of the total time, as finding 10^{-6} nonzeros is nontrivial

What takes time are *sorts* and *searches*

Factorization speeds this up by reducing the lengths of lists to sorted and searched

Advanced topic in factorization: Even *more* divide and conquer!

Consider (proton) Slater determinants:

$n(l)_j$:			$0d_{5/2}$			
m	: 5/2	3/2	½	-1/2	-3/2	-5/2
	1	1	1	0	0	0
	1	1	0	1	0	0
	1	1	0	0	1	0
	1	0	1	1	0	0
	…..					

First, a Slater determinant composed of all single particle states with $m > 0$…

Next, a Slater determinant composed of all single particle states with $m < 0$…

Advanced topic in factorization: Even *more* divide and conquer!

We can combine these *half Slater determinants* into a "full" Slater determinant, in the same way that we combined proton and neutron Slater determinants into the final many-body basis.

Nuclide	Basis dim	# pSDs	# half Slater Determinants
^{20}Ne	640	66	22
^{24}Mg	28,503	495	57
^{28}Si	93,710	924	64
^{48}Cr	1,963,461	4895	386
^{52}Fe	109,954,620	38,760	848
^{56}Ni	1,087,455,228	125,970	1,013

Advanced topic in factorization:
Even *more* divide and conquer!

Sample numbers:

Nuclide	Basis dim	# pSDs	# half Slater Determinants
^{12}C (4hw)	1.1 M	33,475	5448
^{12}C (6hw)	32.6 M	381,159	40,247
^{12}C (8hw)	594 M	2.9 M	232,553
^{16}O (8hw)	996 M	5M	497,493

Advanced topic in factorization:
Even *more* divide and conquer!

Note that while all proton and neutron SDs have the same particle number, we build SDs from half Slaters with differing # of particles (but the sum is fixed—just another quantum number).

This leads to another innovation.

The fundamental operation on half-Slaters is not jumps but "hops" which are single-particle creation/annihilation.

This turns out to be natural, easy, and quick.

Half-Slaters are generated recursively:

$N_h = 0$:	000000				
$N_h = 1$:	100000	010000	001000	000100	….
$N_h = 2$:	110000	011000	001100	000110	….
	101000	010100	001010	000101	….
	100100	010010	001001		….
	100010	010001			….
	100001				

Each half-Slater has a fixed number of *destruction hops*:
it takes only a very short search to find the final half-Slater:

$N_h = 0$: 000000

$N_h = 1$: 100000 010000 001000 000100 ….

$N_h = 2$: 110000 011000 001100 000110 ….
101000 010100 001010 000101 ….
100100 010010 001001 ….
100010 010001 ….
100001

Finding all the *creation hops* is even easier,because
we just reverse the destruction hops:

Like the number of half-Slaters, the number of hops is small

^{28}Si: 192 hops

^{52}Fe: 3820 hops

^{12}C (6hw): 171,409 hops

^{12}C (8hw): 1,061,255 hops

Using hops we can build arbitrary operations :
1-body jumps, 2-body jumps, 3-body jumps,
spectroscopic factors, etc, all using the same underlying structure.

Using half-Slater determinants speeds up basis construction by 3x-4x, and jump construction by 10x

"It's not enough to just show up. You have to have a business plan."

Applications

ab initio Gamow-Teller transitions:

the search for quenching

Strength functions in the nuclear shell model

Part IIb: *ab initio* Gamow-Teller transitions

- Gamow-Teller important for weak physics, astrophysics
- Avoids dependence on radial wavefunctions (at lowest order); mostly SU(4) irreps; Ikeda sum rule strong constraint
- Consistent quenching of coupling—exchange currents, or what?
- **What about 0-neutrino double-beta decay?**

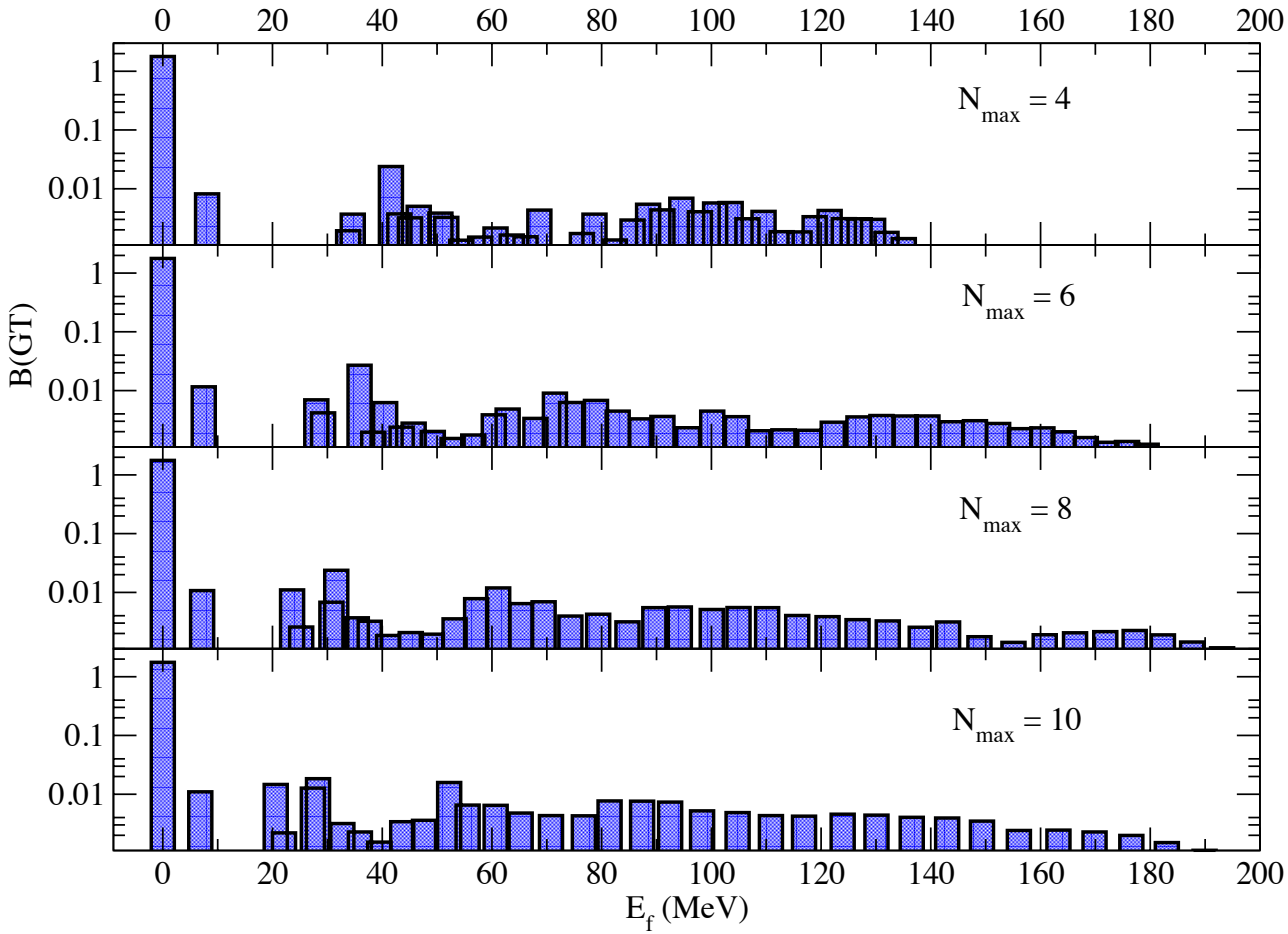
Two recent highlights:

Anomalously long ^{14}C half-life (Maris, Vary, Navratil, Ormand, Nam, Dean) Phys. Rev. Lett. 106, 202502 (2011): 'accidental' cancellation of matrix elements driven by 3-body force

Exchange current corrections from EFT (quenching of about 0.8):
S. Vaintraub, N. Barnea, and D. Gazit, Phys. Rev. C **79**, 065501 (2009);
J. Menendez, D. Gazit, and A. Schwenk, Phys. Rev. Lett **107**, 062501 (2011)

Strength functions in the nuclear shell model

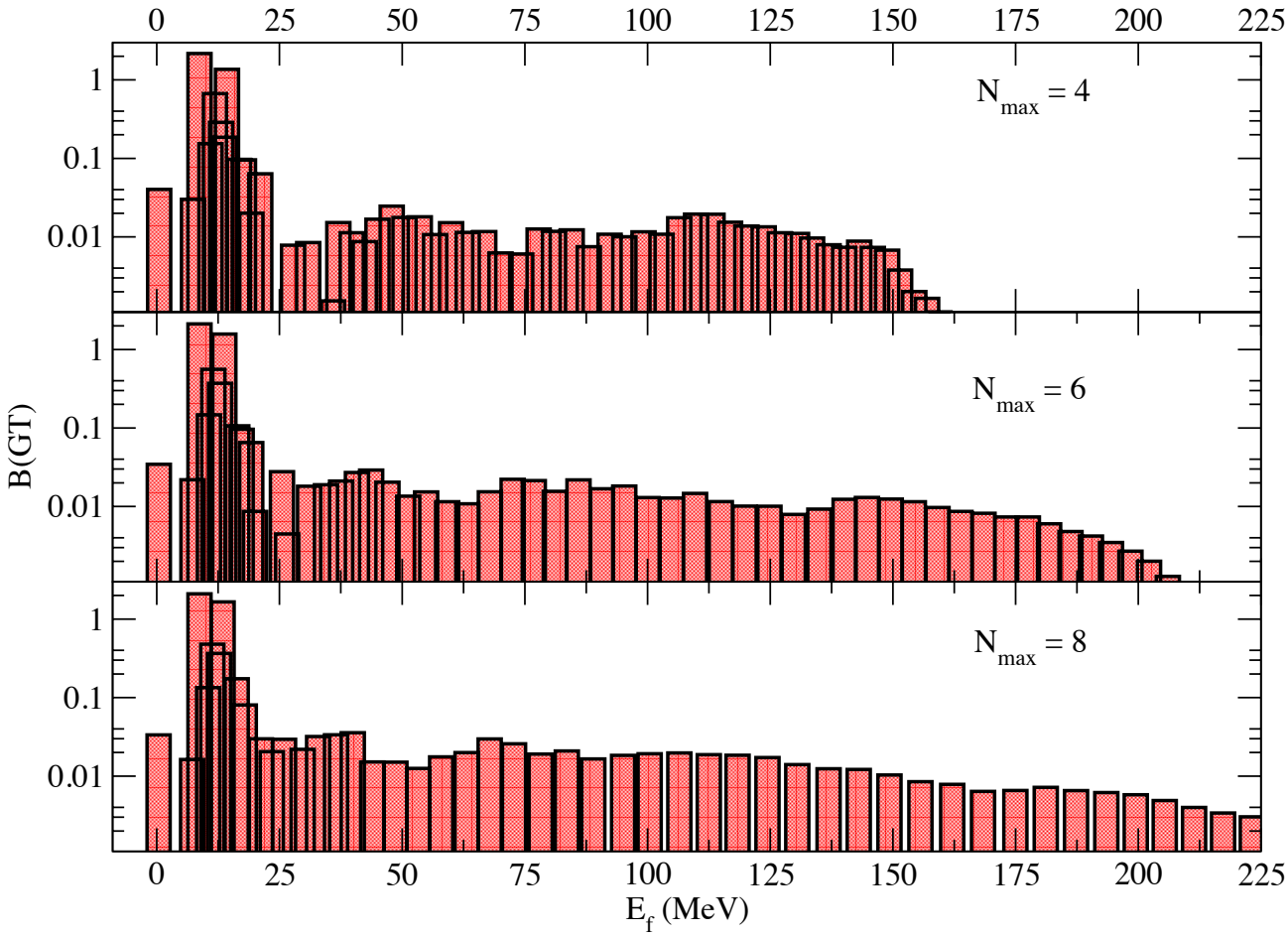
$^6He \rightarrow {}^6Li$



Preliminary!

Chiral 2-body forces SRG evolved to λ=2 f

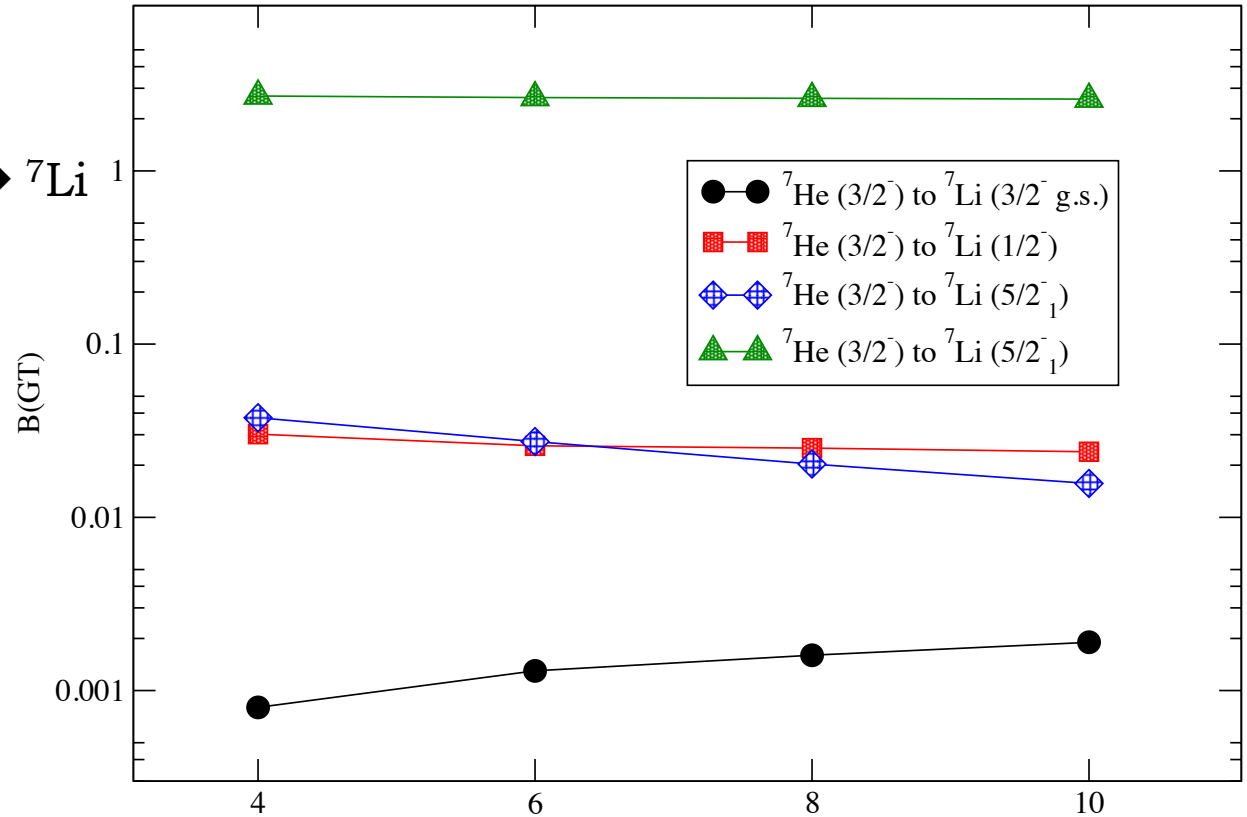
$^{7}He \rightarrow {}^{7}Li$



Preliminary!

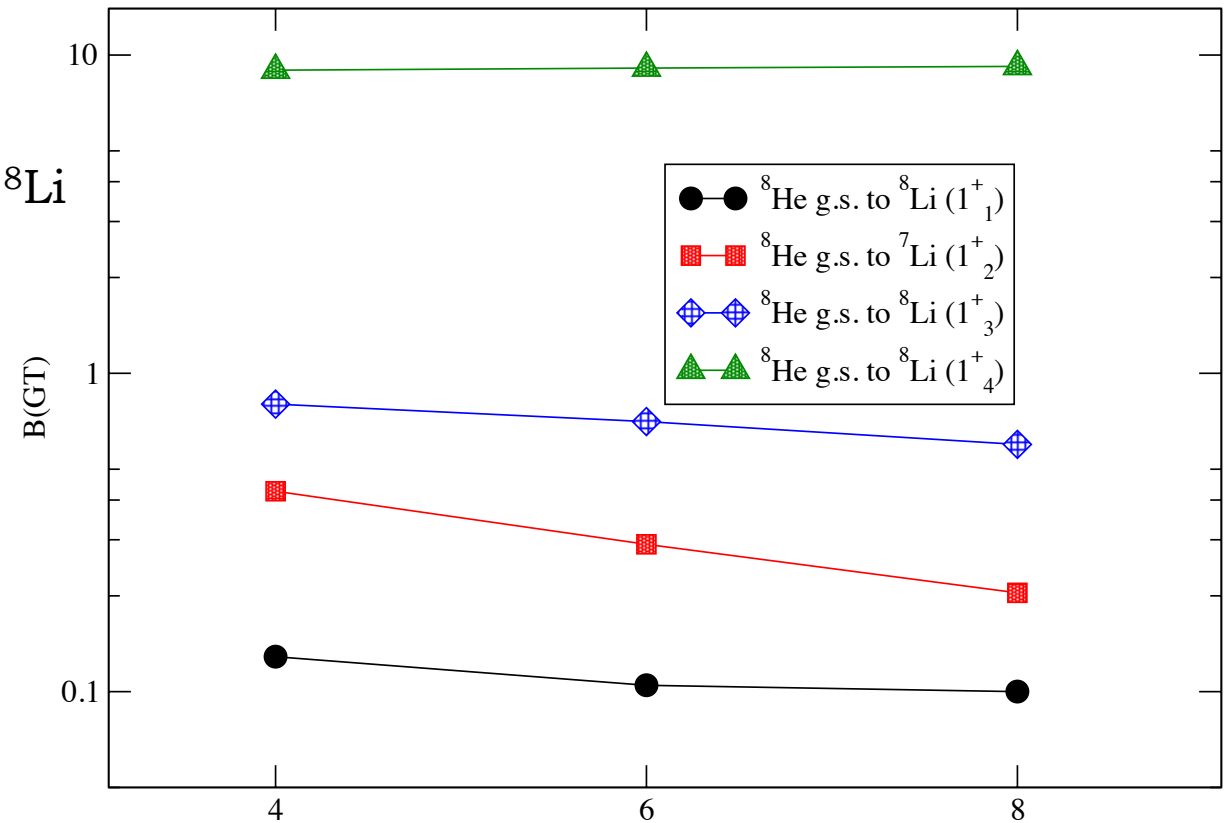
(Run on desktop machine with BIGSTICK)

$^7He \rightarrow {}^7Li$



Preliminary!

$^{8}He \rightarrow {}^{8}Li$



Preliminary!

Strength functions in the nuclear shell model

Need to run higher N_{max} (on supercomputers) but …

Despite being a "simple" operator, transition matrix elements of Gamow-Teller ($\sigma\tau$) do not have simple behavior:

- Some transitions quickly converge as we go up in N_{max}, others not
- Should be investigated by doing L-S/SU(4) decomposition
- Effect of 3-body forces likely important
- More work on chiral EFT exchange forces should be done
- Likely strong implications for $0\nu-\beta\beta$ matrix elements…

Applications

Ab initio E1 response

and

the Brink-Axel hypothesis

Transitions and the Brink-Axel hypothesis

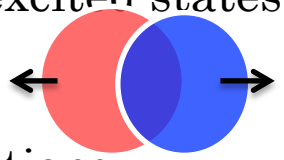
+ Michael K. G. Kruse (LLNL), W. Erich Ormand (LLNL), and Micah Schuster (SDSU)

Brink-Axel hypothesis (D. Brink, D. Phil. thesis, Oxford University (unpublished), 1955; P. Axel, Phys. Rev. **126**, 671 (1962)):

If the ground state has a giant dipole resonance (GDR), then excited states should have GDR

and

because the GDR is a collective proton-versus-neutrons oscillations, the GDR should be insensitive to the initial state.



Electric dipole

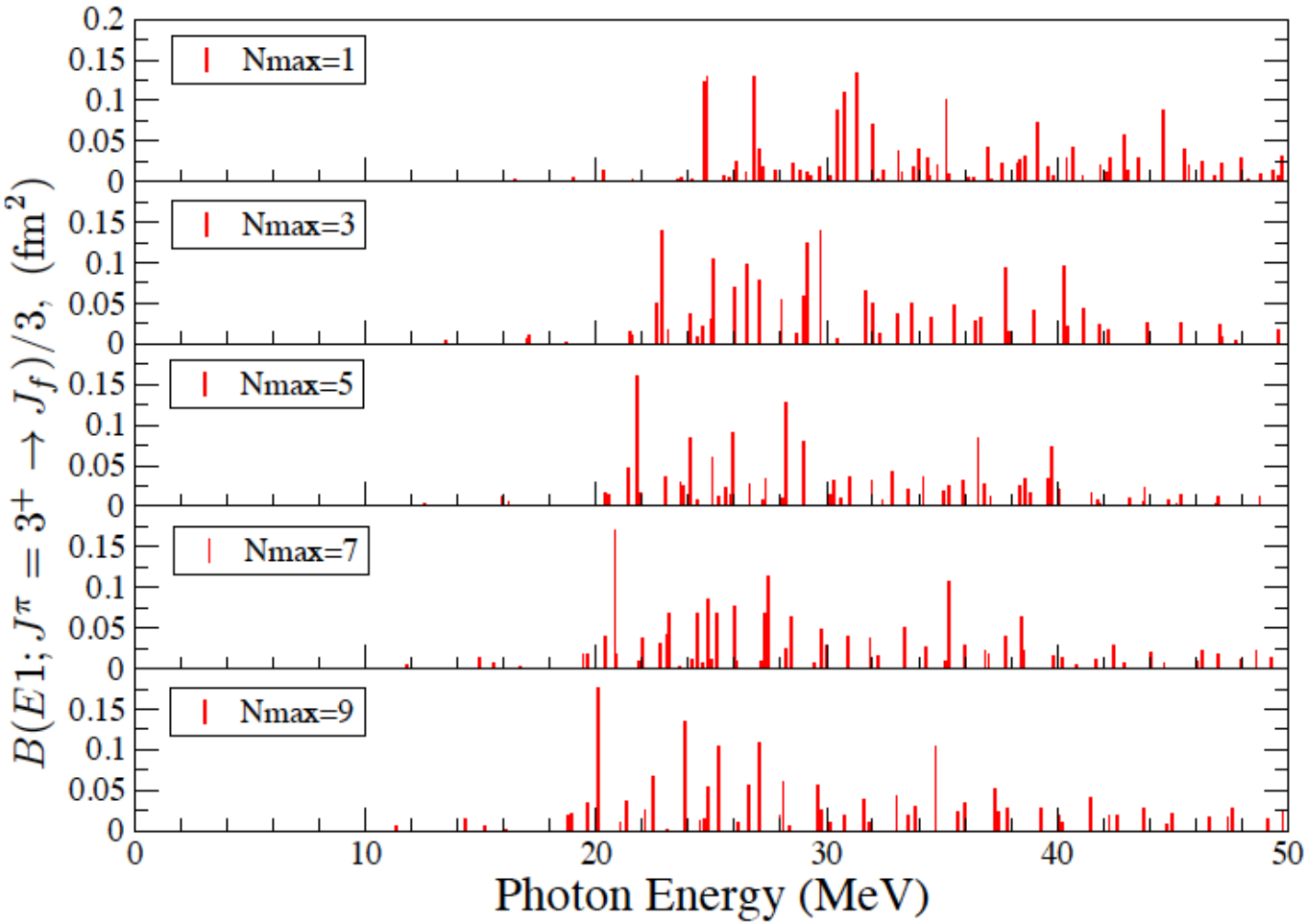
$$S(E_i, E_x) = \sum_f |\langle f|\hat{T}|i\rangle| \delta(E_x - E_f + E_i)$$

"Transition strength function"

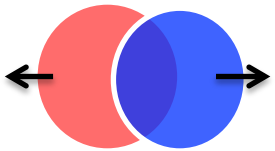
Brink-Axel: "S(E_i,E_x) independent of E_i"

Strength functions in the nuclear shell model

Kruse, Ormand, and Johnson: arXiv:1502:03464



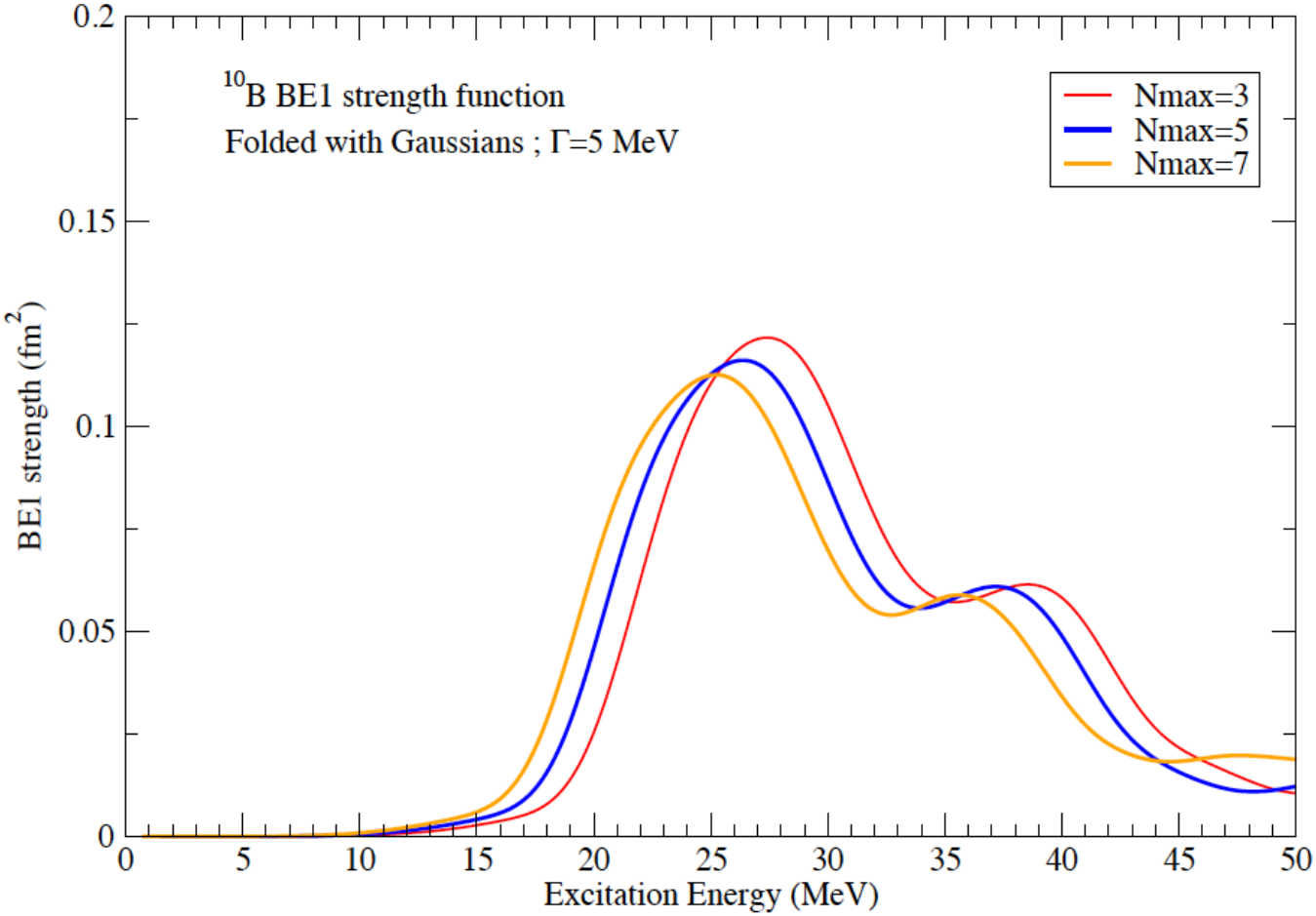
^{10}B E1 response



Electric dipole

Strength functions in the nuclear shell model

B(E1) strength with increasing basis size



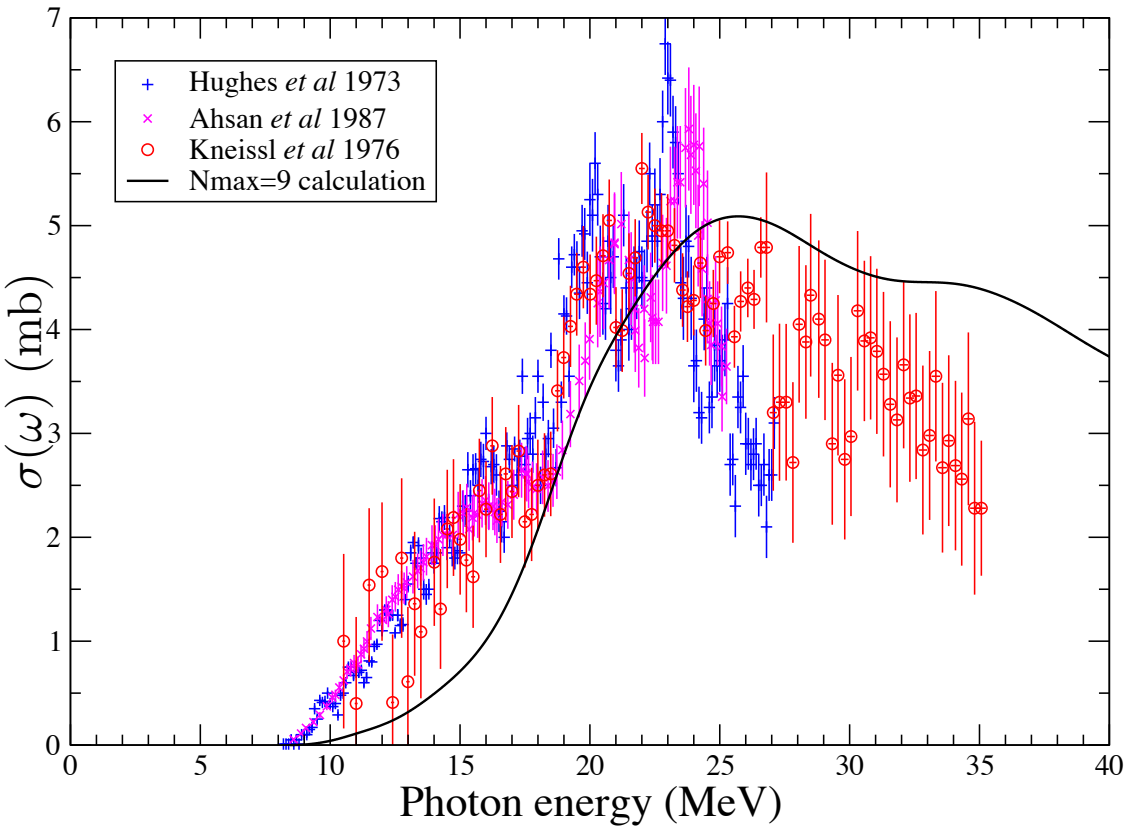
Strength distribution shape is robust in Nmax.

Slowly moves down in energy as a function of Nmax.

How to extrapolate this distribution?

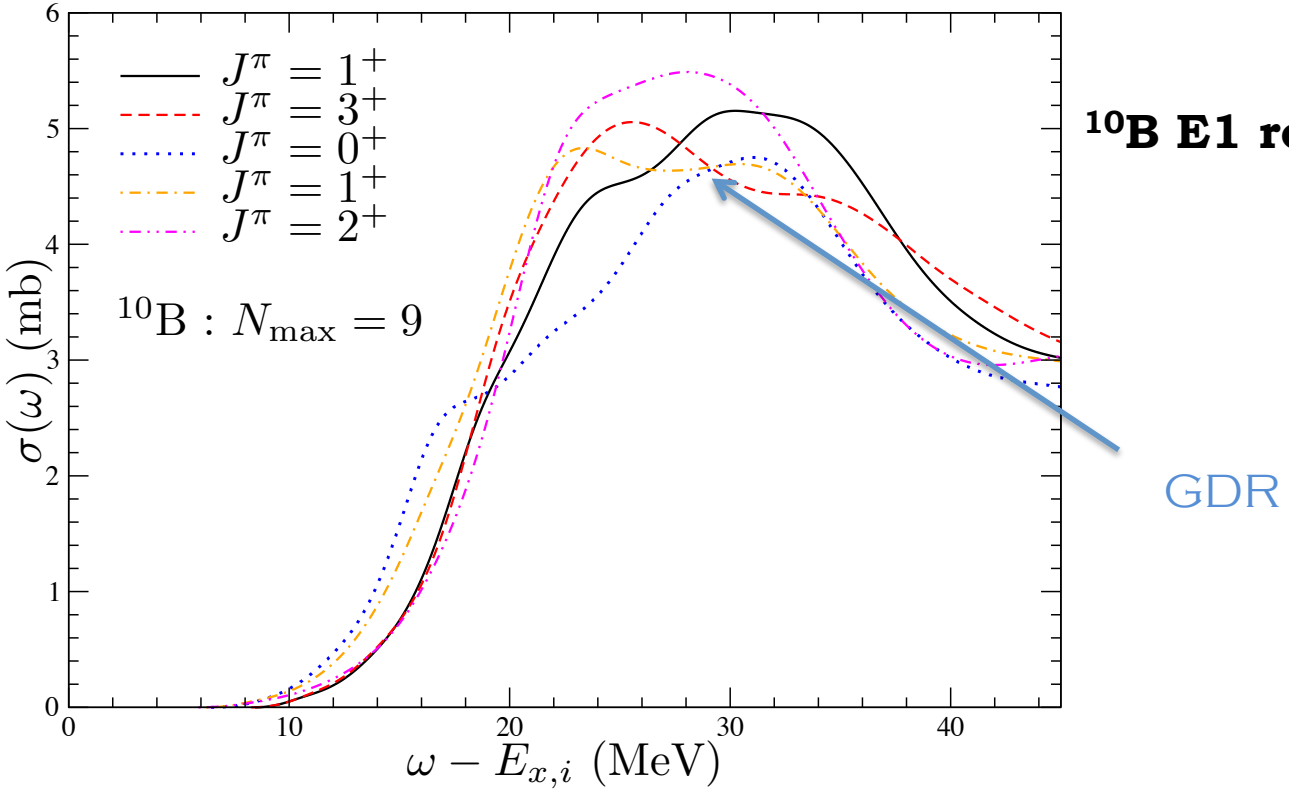
Perhaps it is best to extrapolate centroids?

Kruse, Ormand, and Johnson: arXiv:1502:03464



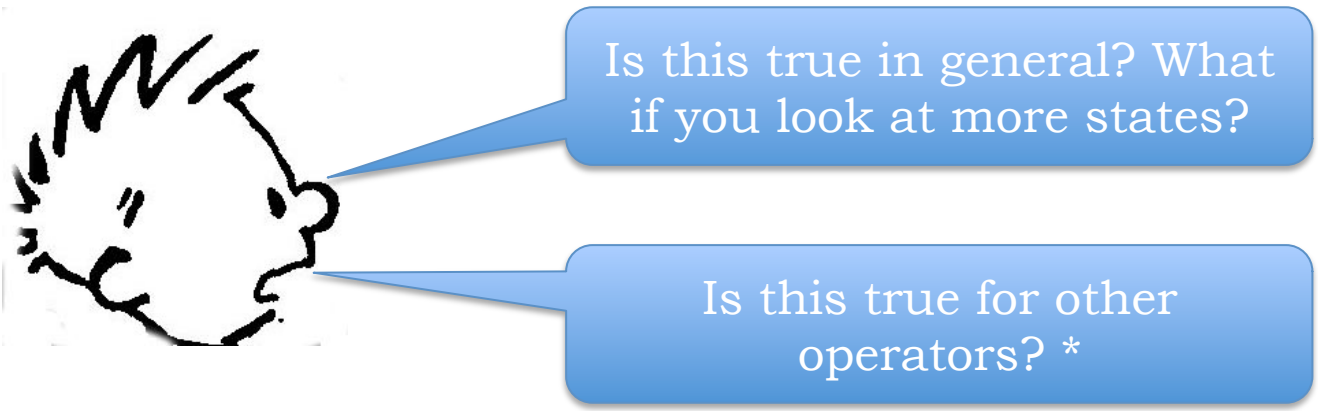
^{10}B E1 response

Kruse, Ormand, and Johnson: arXiv:1502:03464



Brink-Axel: "S(E_i,E_x) independent of E_i"

Strength functions in the nuclear shell model



* Some evidence to the contrary (with Gamow-Teller operator): Frazier, Brown, Millener, and Zelevinsky, Phys. Lett B **414**, 7 (1997); Misch, Fuller, and Brown, PRC 90, 065808 (2014)

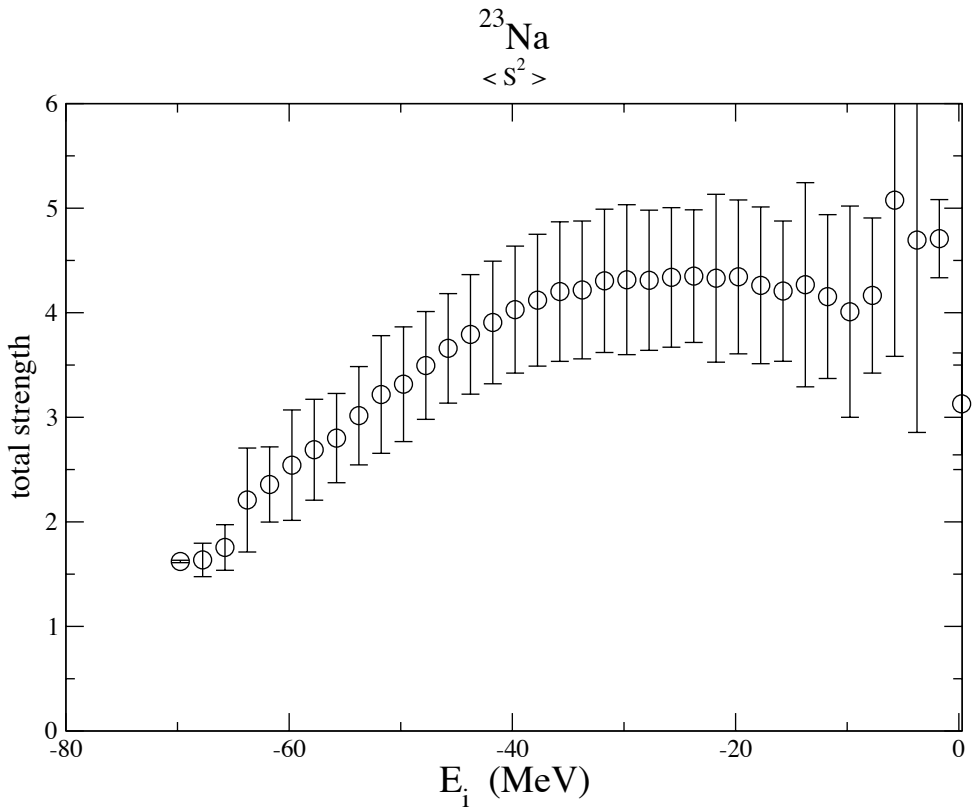
The total strength
(or *non-energy-weighted sum rule*)
can be computed as a simple expectation value

$$S_0(E_i) = \int S(E_i, E_x) dE_x = \sum_f |\langle f|\hat{T}|i\rangle| = \langle i|\hat{T}^+ T|i\rangle$$

Looks like large fluctuations about the average; can we characterize / quantify this?

The total strength (or *non-energy-weighted sum rule*)

$$\int S(E_i, E_x)dE_x = \sum \left|\langle f|\hat{T}|i\rangle\right| = \langle i|\hat{T}^{+}T|i\rangle$$



Furthermore, the smooth secular behavior is easily understood through *spectral distribution theory* of J. B. French *et al*

Average expectation value is just a trace!

$$\langle \hat{O} \rangle = \frac{1}{N} \sum_i \langle i | O | i \rangle = \frac{1}{N}\ tr\ (\hat{O})$$

Furthermore, the smooth secular behavior is easily understood through *spectral distribution theory* of J. B. French *et al*

Average expectation value is just a trace!

$$\langle \hat{O} \rangle = \frac{1}{N} \sum_i \langle i | O | i \rangle = \frac{1}{N}\ tr\ (\hat{O})$$

(Linear) energy dependence is *also* a trace!

$$\frac{1}{N} \sum_i E_i \langle i | O | i \rangle = \frac{1}{N} \sum_i \langle i | OH | i \rangle = \frac{1}{N}\ tr\ (\widehat{OH})$$

Slope is given by < O H > - < O > < H >

Furthermore, the smooth secular behavior is easily understood through *spectral distribution theory* of J. B. French *et al*

Average expectation value is just a trace!

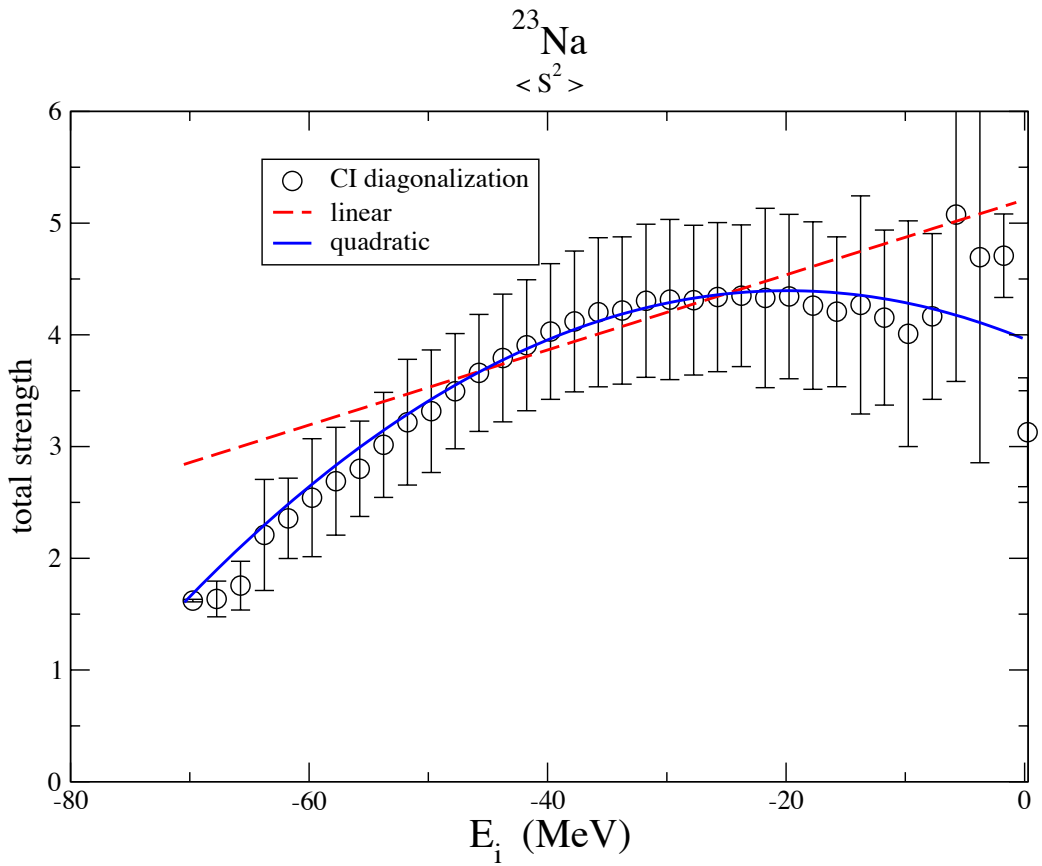
$$\langle \hat{O} \rangle = \frac{1}{N} \sum_i \langle i | O | i \rangle = \frac{1}{N}\ tr\ (\hat{O})$$

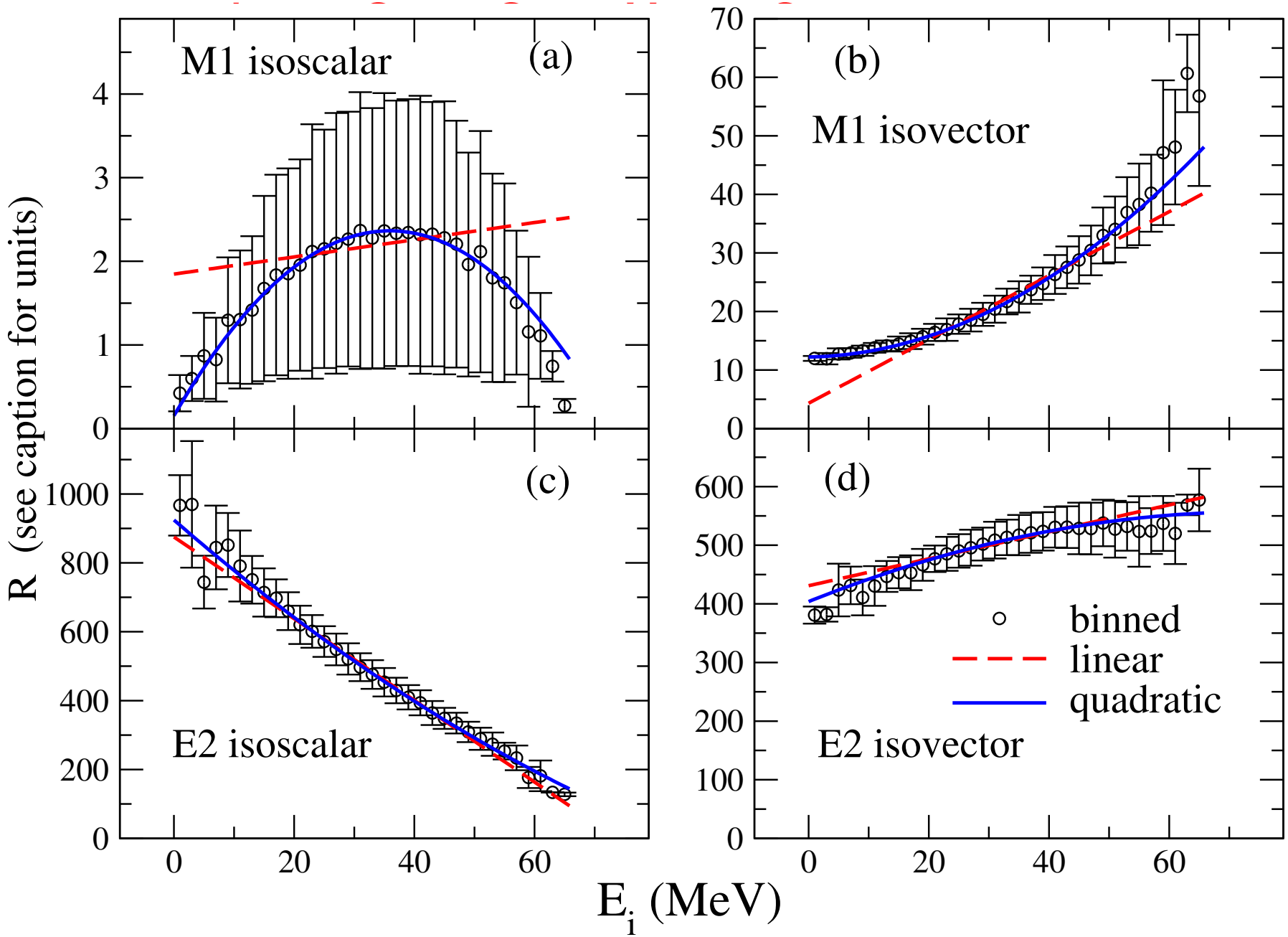
(Linear) energy dependence is *also* a trace!

$$\frac{1}{N} \sum_i E_i \langle i | O | i \rangle = \frac{1}{N} \sum_i \langle i | OH | i \rangle = \frac{1}{N}\ tr\ (\widehat{OH})$$

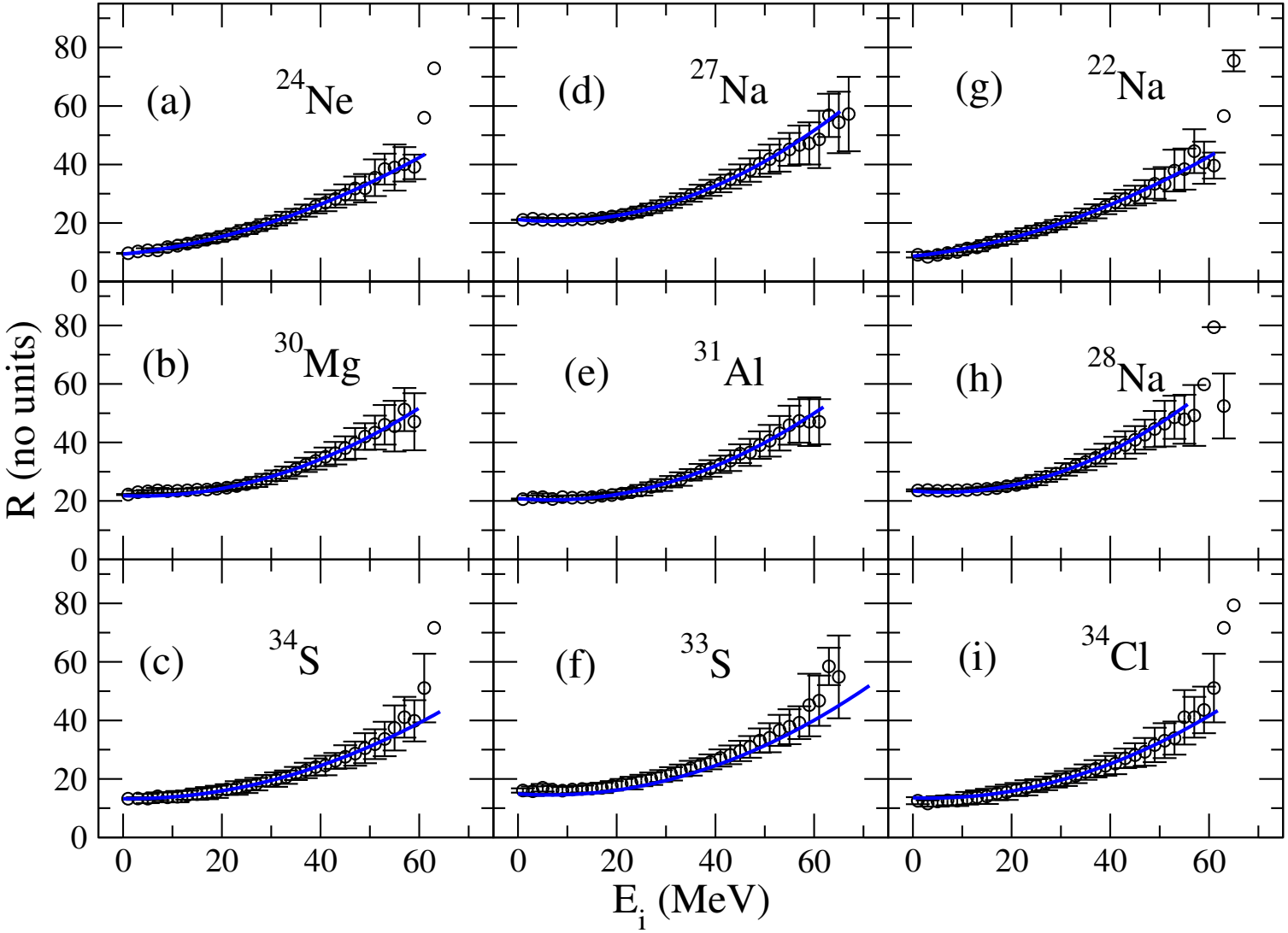
From this we can derive the secular behavior of expectation values

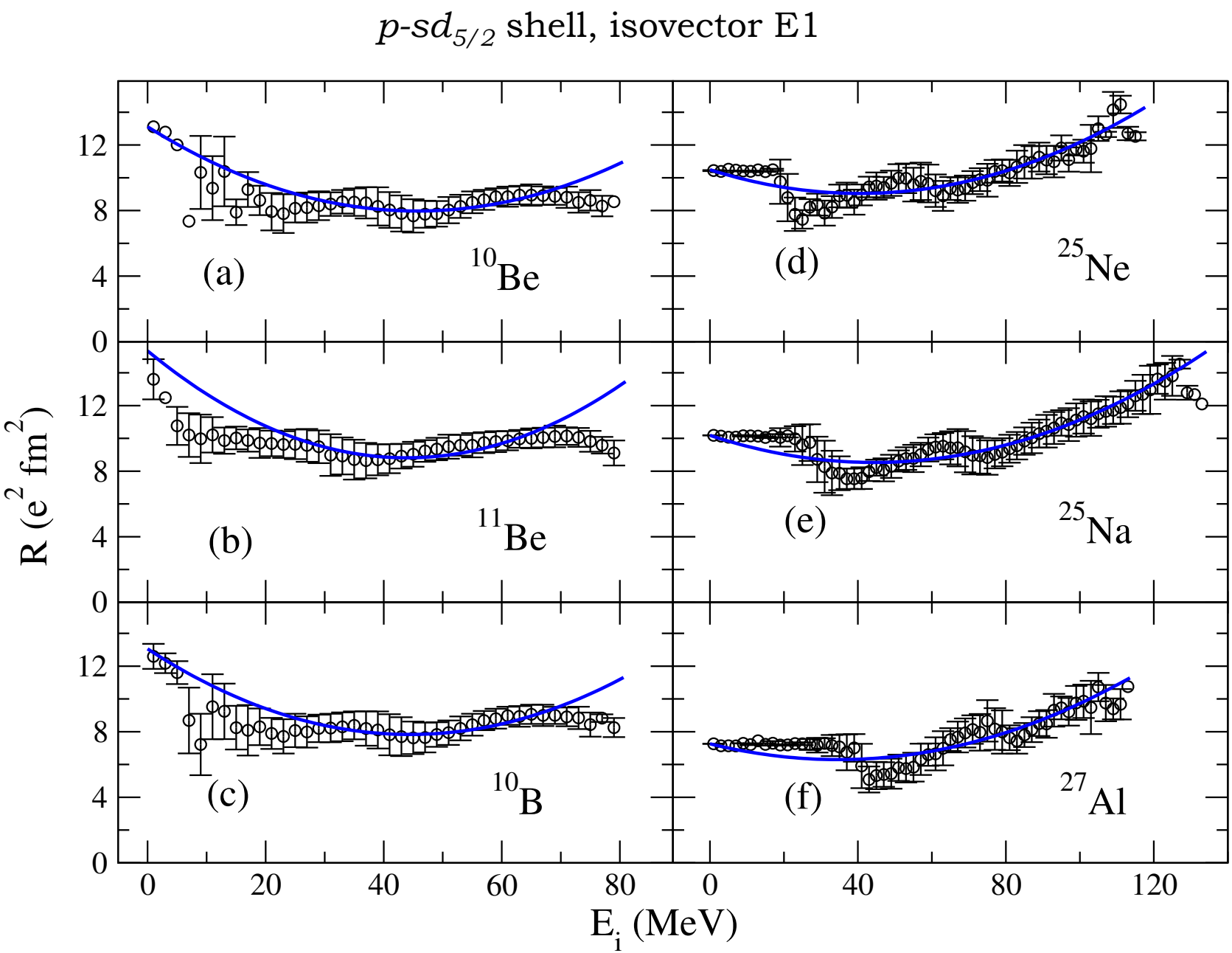
Furthermore, the smooth secular behavior is easily understood through *spectral distribution theory* of J. B. French *et al*




M1 isoscalar
(a)
(b)
M1 isovector
(c)
E2 isoscalar
(d)
E2 isovector
binned
linear
quadratic
R (see caption for units)
E_i (MeV)
0
1
2
3
4
10
20
30
40
50
60
70
200
400
600
800
1000
100
300
500

sd shell, Gamow-Teller

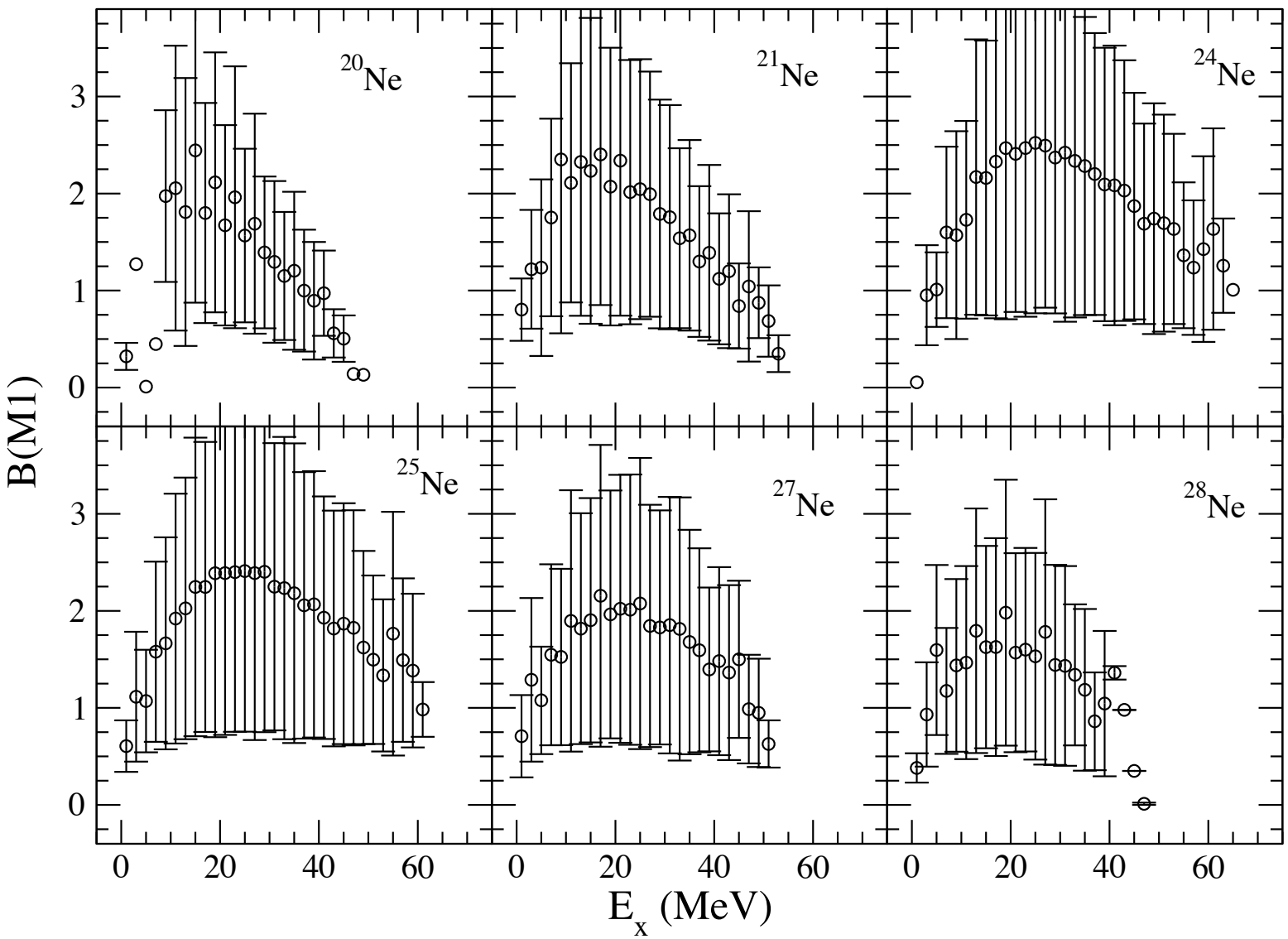



p-$sd_{5/2}$ shell, isovector E1
(a)
^{10}Be
(b)
^{11}Be
(c)
^{10}B
(d)
^{25}Ne
(e)
^{25}Na
(f)
^{27}Al
R (e^2 fm^2)
E_i (MeV)
0
4
8
12
0
20
40
60
80
120

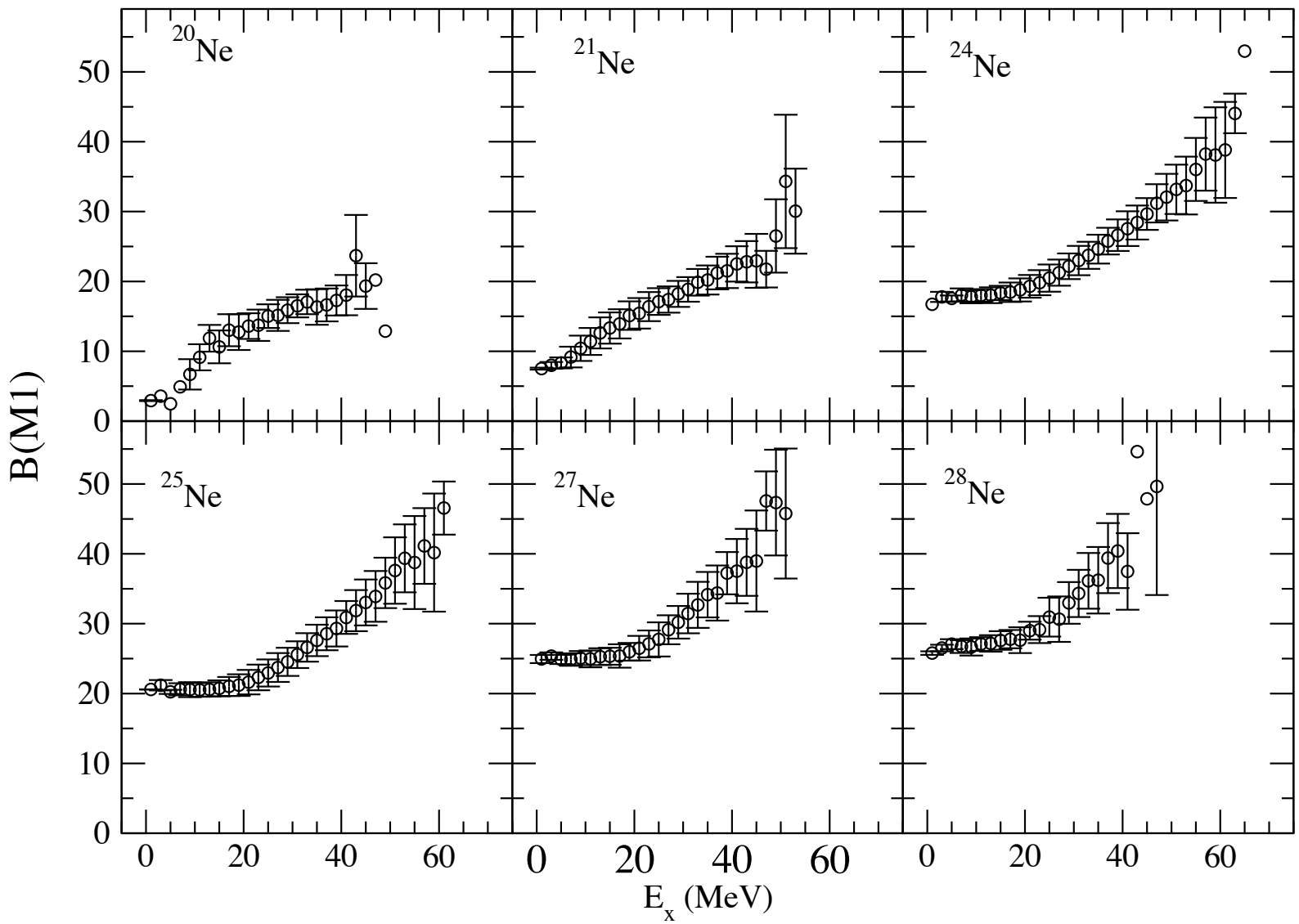
Strength functions in the nuclear shell model

What about as we go to extreme isospin?

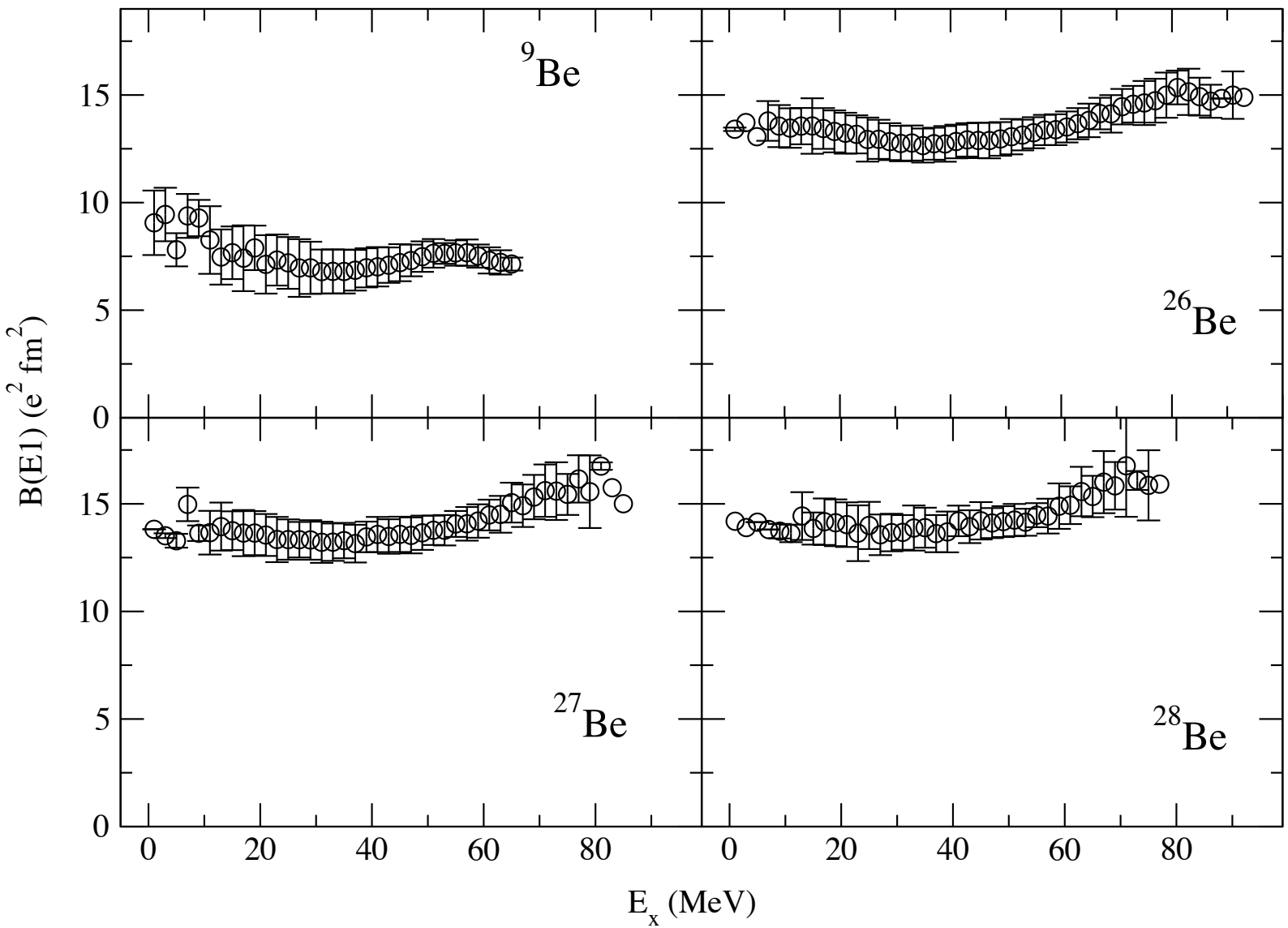
sd shell, isoscalar M1



sd shell, isovector M1



p-$sd_{5/2}$ shell, isovector E1



Strength functions in the nuclear shell model

What we do learn from this?

The generalized Brink-Axel hypothesis
(for arbitrary operators) is *wrong*!
-- total strength evolves with initial (parent) energy
-- significant fluctuations even for nearby parent states

We can understand this through *spectral distribution theory,*
that is,
traces of operators (weighted by the energy);

A lack of energy dependence can occur *only* if

$$\langle O H \rangle - \langle O \rangle \langle H \rangle = 0$$

Strength functions in the nuclear shell model

Also (unsurprisingly) isovector transitions show more evolution as we go to extreme isospin

The generalized Brink-Axel hypothesis
(for arbitrary operators) is *wrong*!
-- total strength evolves with initial (parent) energy
-- significant fluctuations even for nearby parent states

We can understand this through *spectral distribution theory,*
that is,
traces of operators (weighted by the energy);

A lack of energy dependence can occur *only* if

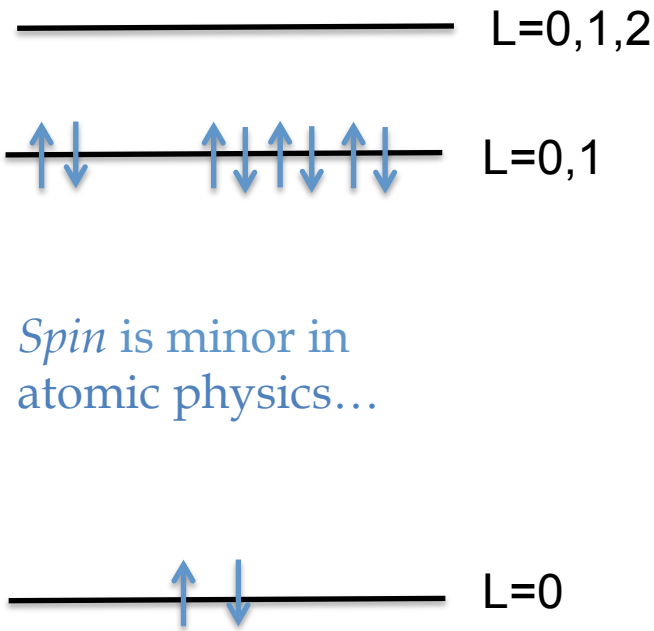
$$\langle O H \rangle - \langle O \rangle \langle H \rangle = 0$$

Applications

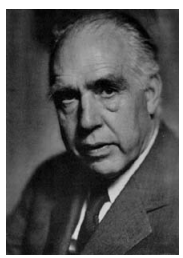
Spin-orbit decomposition of *ab initio* nuclides

C. W. J, Phys. Rev. C **91**, 034313 (2015).

Atoms :



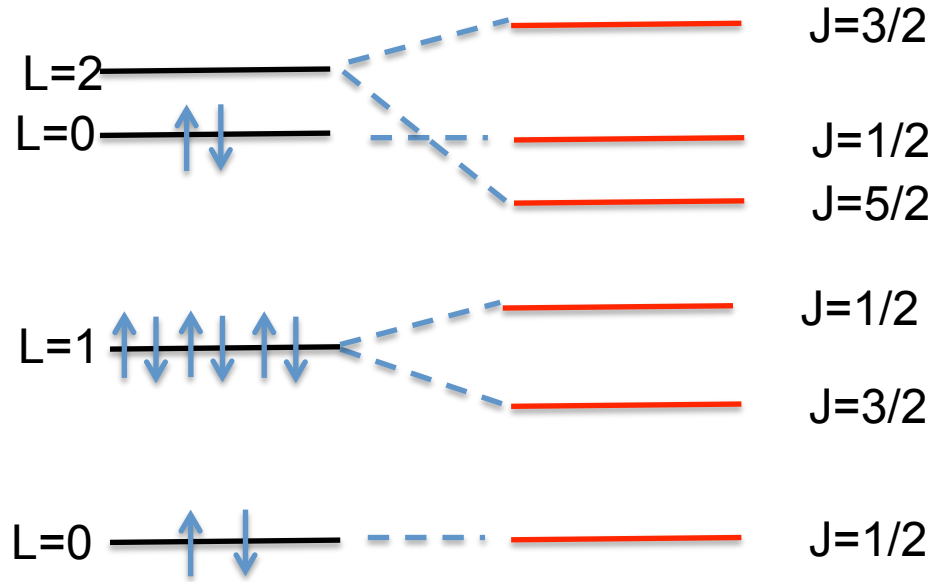
Spin is minor in atomic physics…



(Niels Bohr)

(E. Schrodinger)

Nuclei:



…but crucial in nuclear physics…

(Maria Goeppert-Mayer)

j-j versus L-S

$$\boxed{l_1 + s_1} \quad \boxed{l_2 + s_2} \quad \boxed{l_3 + s_3} \quad \boxed{l_4 + s_4} \quad \boxed{\ldots}$$

$$\boxed{\begin{array}{ccccccccc} = & & = & & = & & = & & \\ j_1 & + & j_2 & + & j_3 & + & j_4 & + & \ldots \end{array}} = J$$

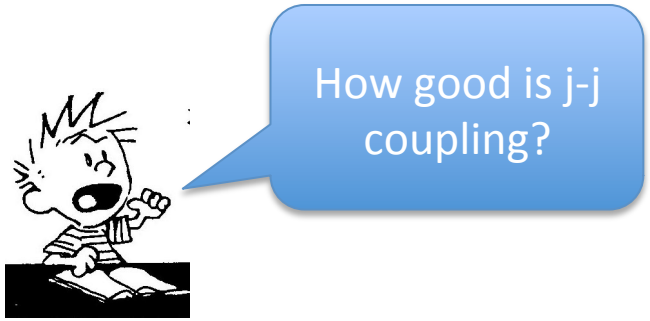
"j-j coupling"

$$\boxed{l_1 + l_2 + l_3 + l_4 + \ldots}$$

$$\boxed{s_1 + s_2 + s_3 + s_4 + \ldots}$$

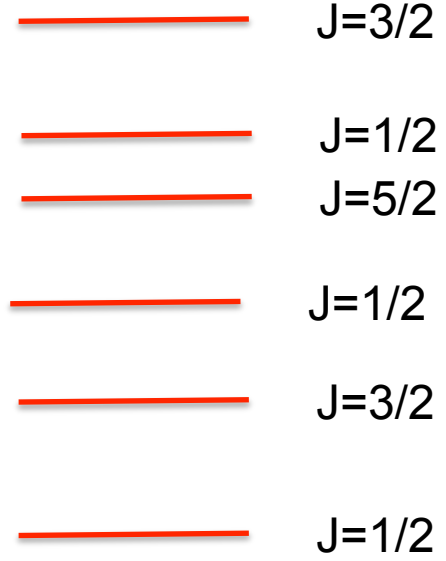
$$\boxed{\begin{array}{c} = L \\ + \\ = S \end{array}} = J$$

"L-S coupling"



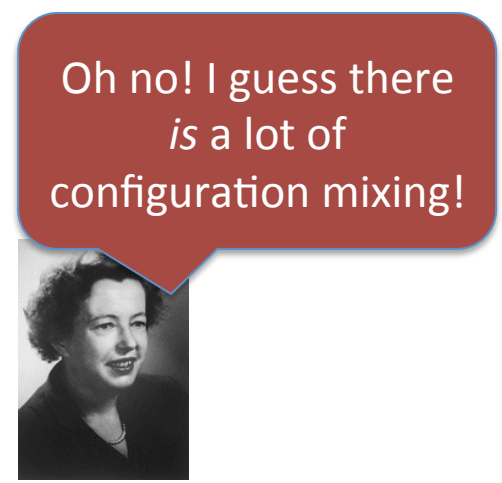
(Calculations are standard configuration-mixing: diagonalization of Hamiltonian in *m*-scheme Slater determinants, in single major harmonic oscillator shell)

Nuclei:

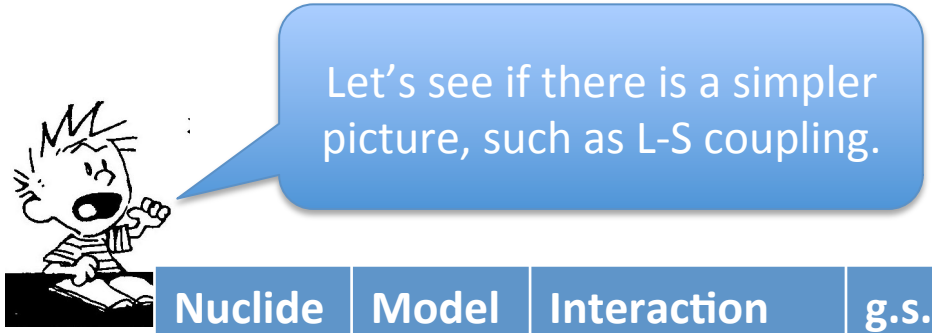


Nuclide	Model space	Interaction	g.s. =
^{48}Ca	pf	KB3G	90 % $(0f_{7/2})^8$
^{24}O	sd	USDB	91% $(0d_{5/2})^6 (1s_{1/2})^2$
^{22}O	sd	USDB	75% $(0d_{5/2})^6$
^{8}He	p	Cohen-Kurath	53 % $(0p_{3/2})^4$

Nuclide	Model space	Interaction	g.s. =
^{32}S	sd	USDB	29 % $(0d_{5/2})^{12} (1s_{1/2})^4$
^{28}Si	sd	USDB	21% $(0d_{5/2})^{12}$
^{12}C	p	Cohen-Kurath	37% $(0p_{3/2})^8$

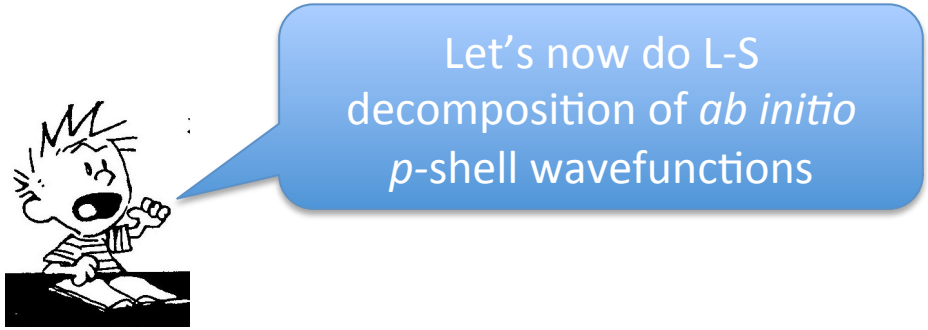


(Maria Goeppert-Mayer)



Nuclide	Model space	Interaction	g.s. =	g.s. =
^{48}Ca	pf	KB3G	90 % $(0f_{7/2})^8$	20% L = 0
^{24}O	sd	USDB	91% $(0d_{5/2})^6 (1s_{½})^2$	34% L = 0
^{22}O	sd	USDB	75% $(0d_{5/2})^6$	38% L = 0
^{8}He	p	Cohen-Kurath	53 % $(0p_{3/2})^4$	96% L = 0
^{32}S	sd	USDB	29 % $(0d_{5/2})^{12} (1s_{½})^4$	34% L = 0
^{28}Si	sd	USDB	21% $(0d_{5/2})^{12}$	36% L = 0
^{12}C	p	Cohen-Kurath	37% $(0p_{3/2})^8$	82% L = 0

This illustrates a (once) well-known fact: that L-S coupling is a better approximation in the *p*-shell than *j*-*j* coupling.



Why?

-- To see if this pattern holds for *ab initio* interactions

-- How well do phenomenological interactions match *ab initio*?

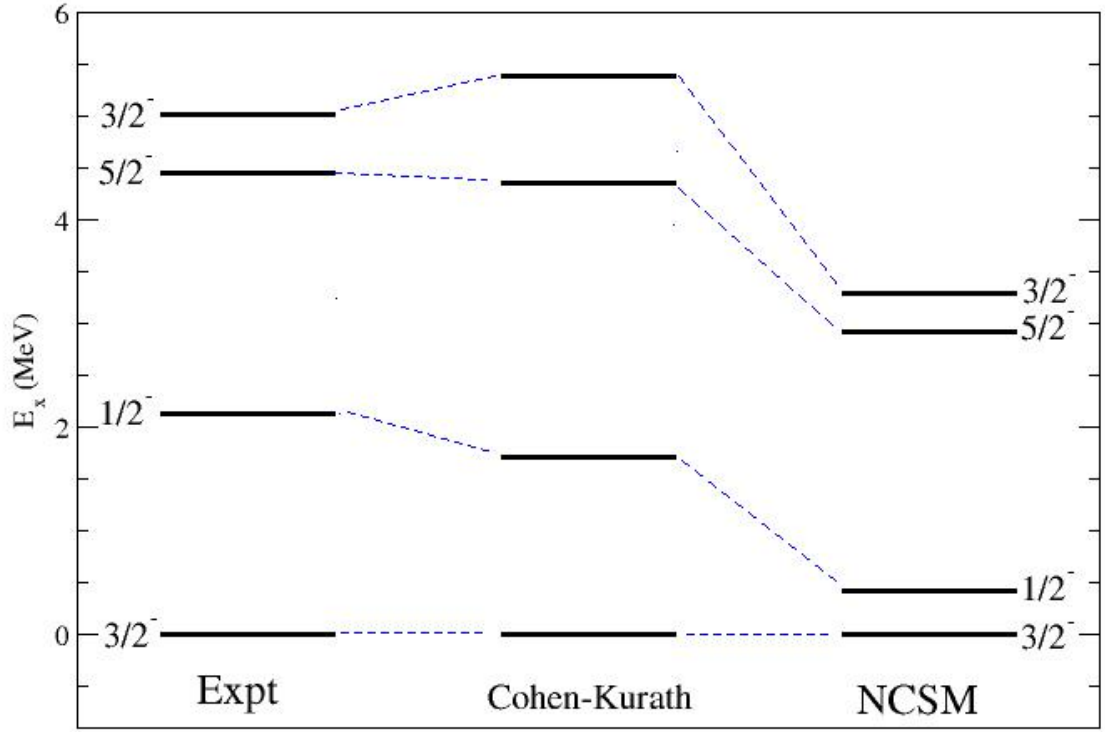
-- Crucially, we know the 3-body forces strongly affects the spin-orbit force. Can we see this happen directly?

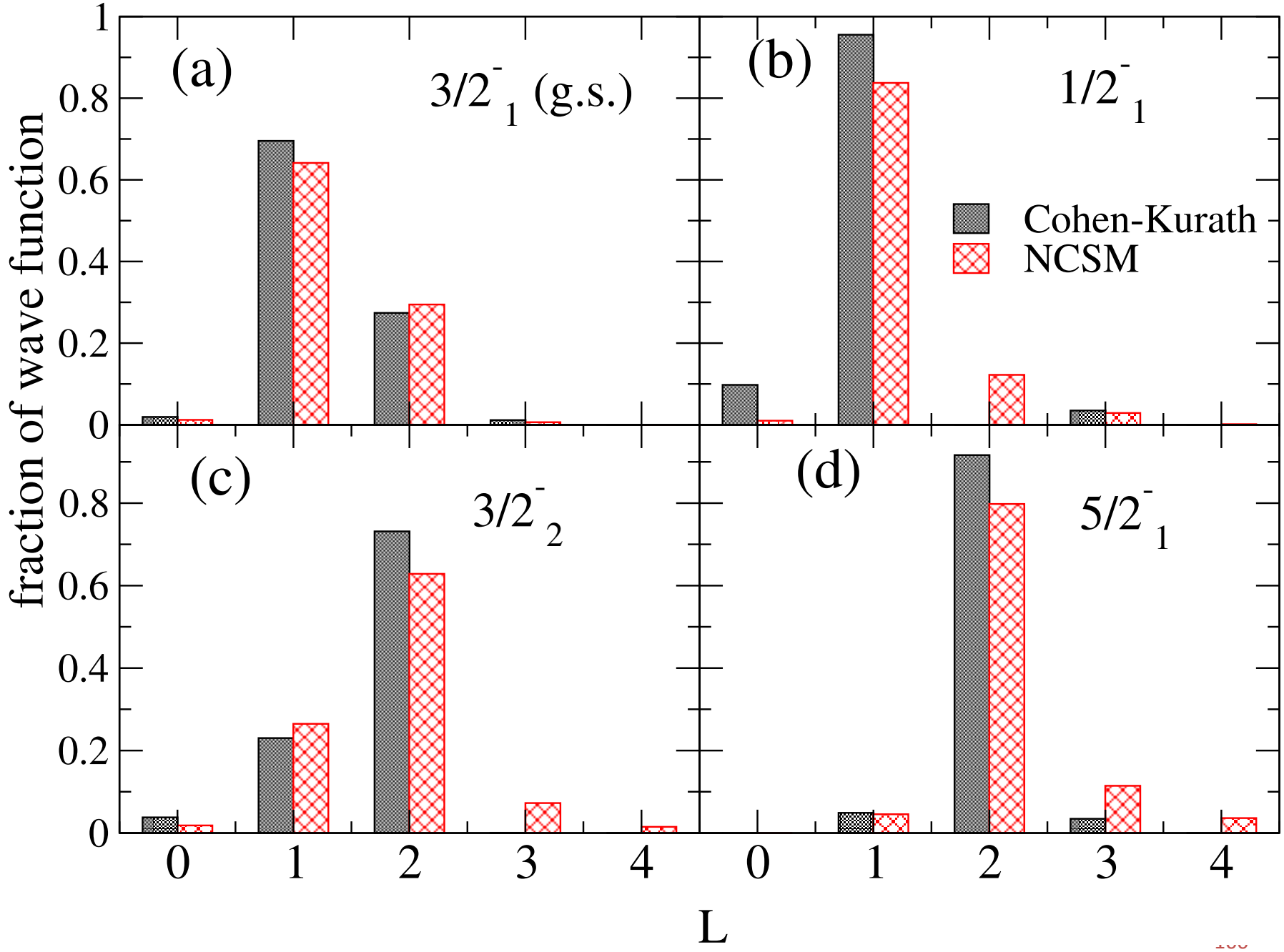
Note: In this talk I only give 2-body results. 3-body forces later…

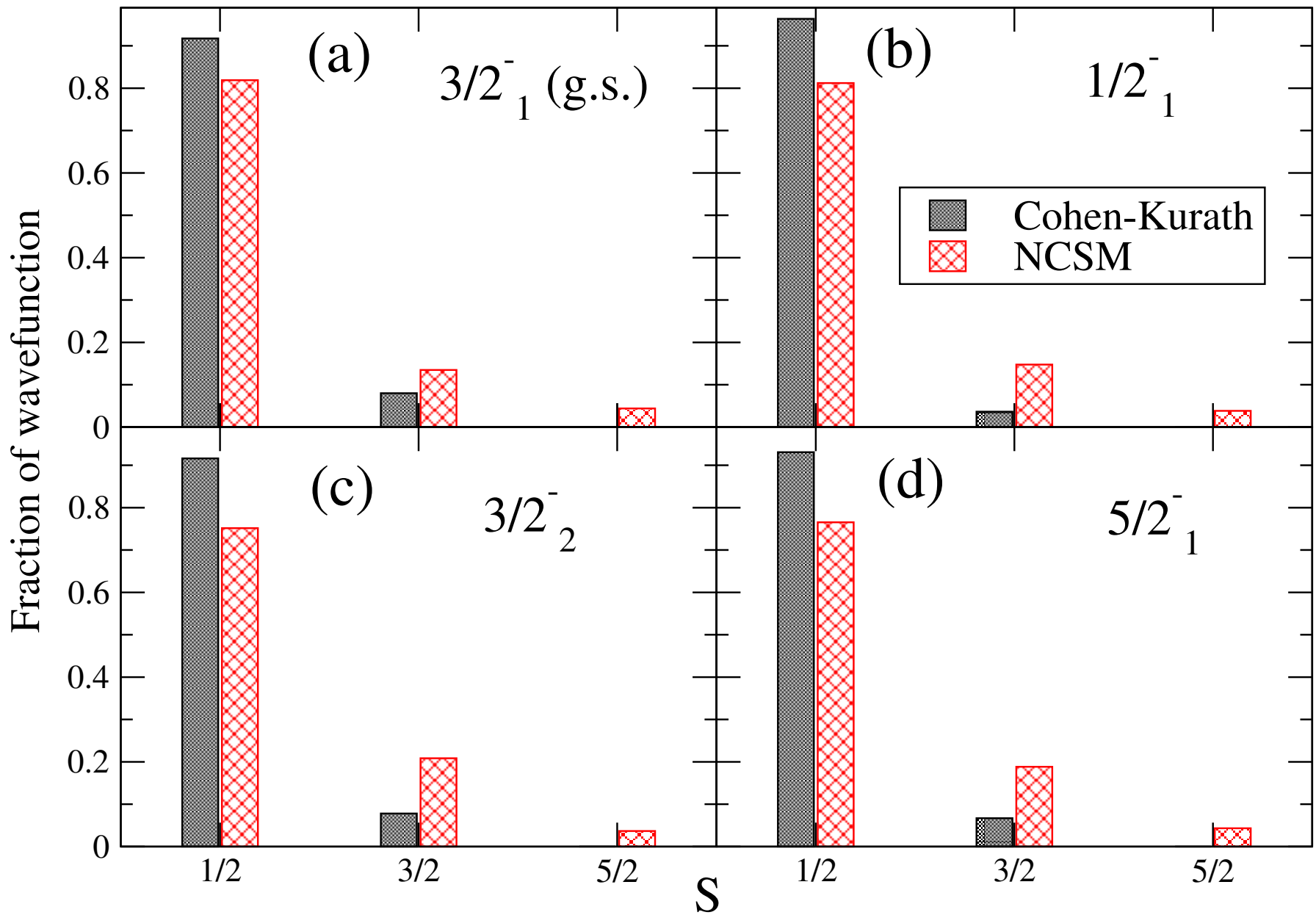
^{11}B

Phenomenological Cohen-Kurath *m*-scheme dimension: 62

NCSM: N3LO chiral 2-body force SRG evolved to λ = 2.0 fm^{-1}, N_{max} = 6, $\hbar\omega$=22 MeV
m-scheme dimension: 20 million



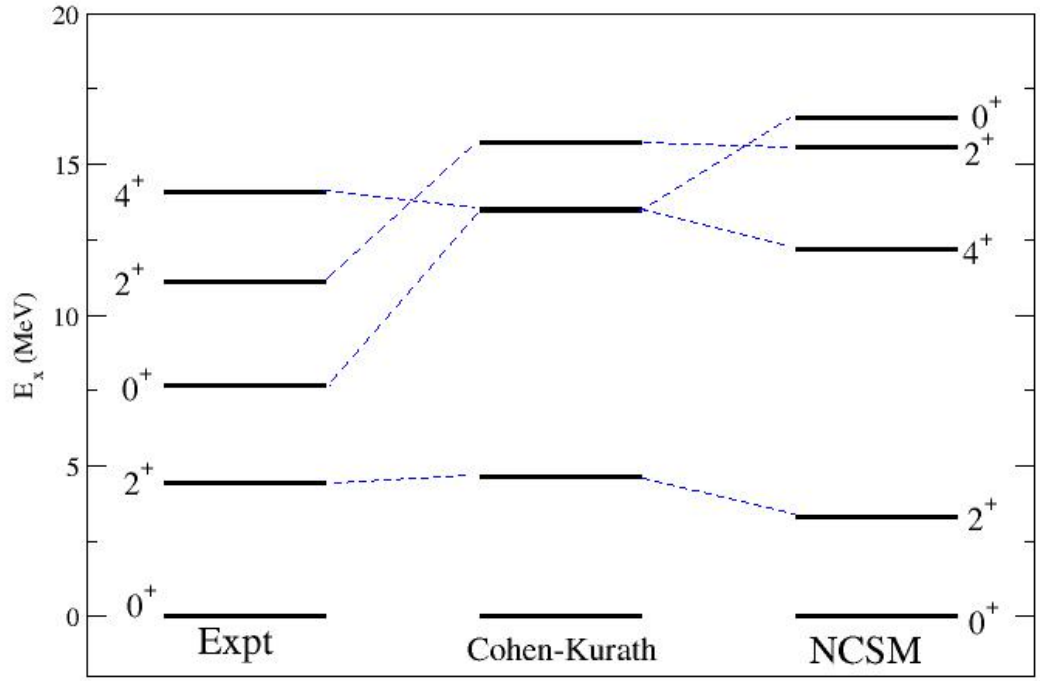

(a)
3/2⁻ 1 (g.s.)
(b)
1/2⁻ 1
Cohen-Kurath
NCSM
(c)
3/2⁻ 2
(d)
5/2⁻ 1
fraction of wave function
L
0
0.2
0.4
0.6
0.8
1
0
1
2
3
4

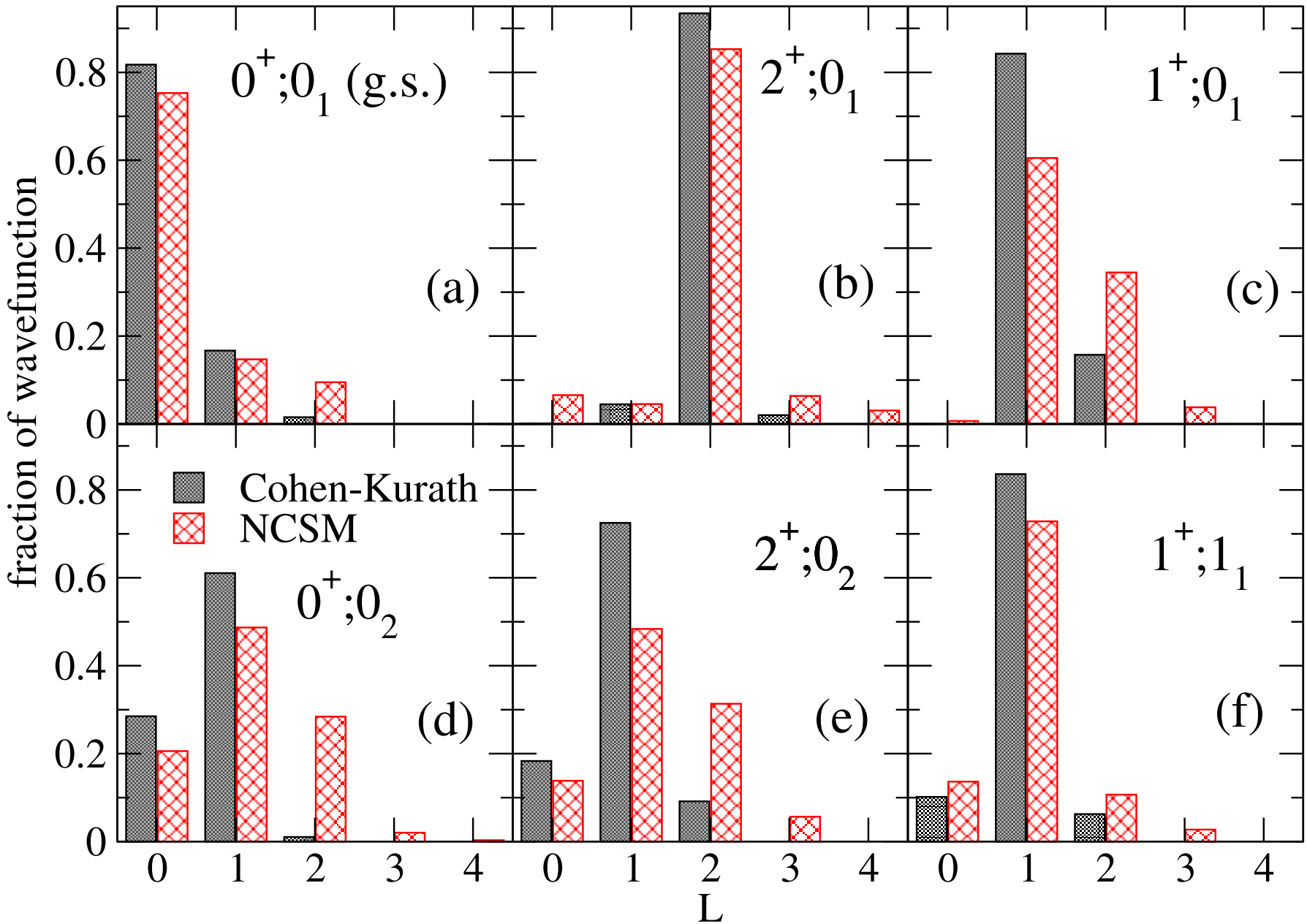

(a)
3/2⁻ 1 (g.s.)
(b)
1/2⁻ 1
(c)
3/2⁻ 2
(d)
5/2⁻ 1
Cohen-Kurath
NCSM
Fraction of wavefunction
0
0.2
0.4
0.6
0.8
1/2
3/2
5/2
S

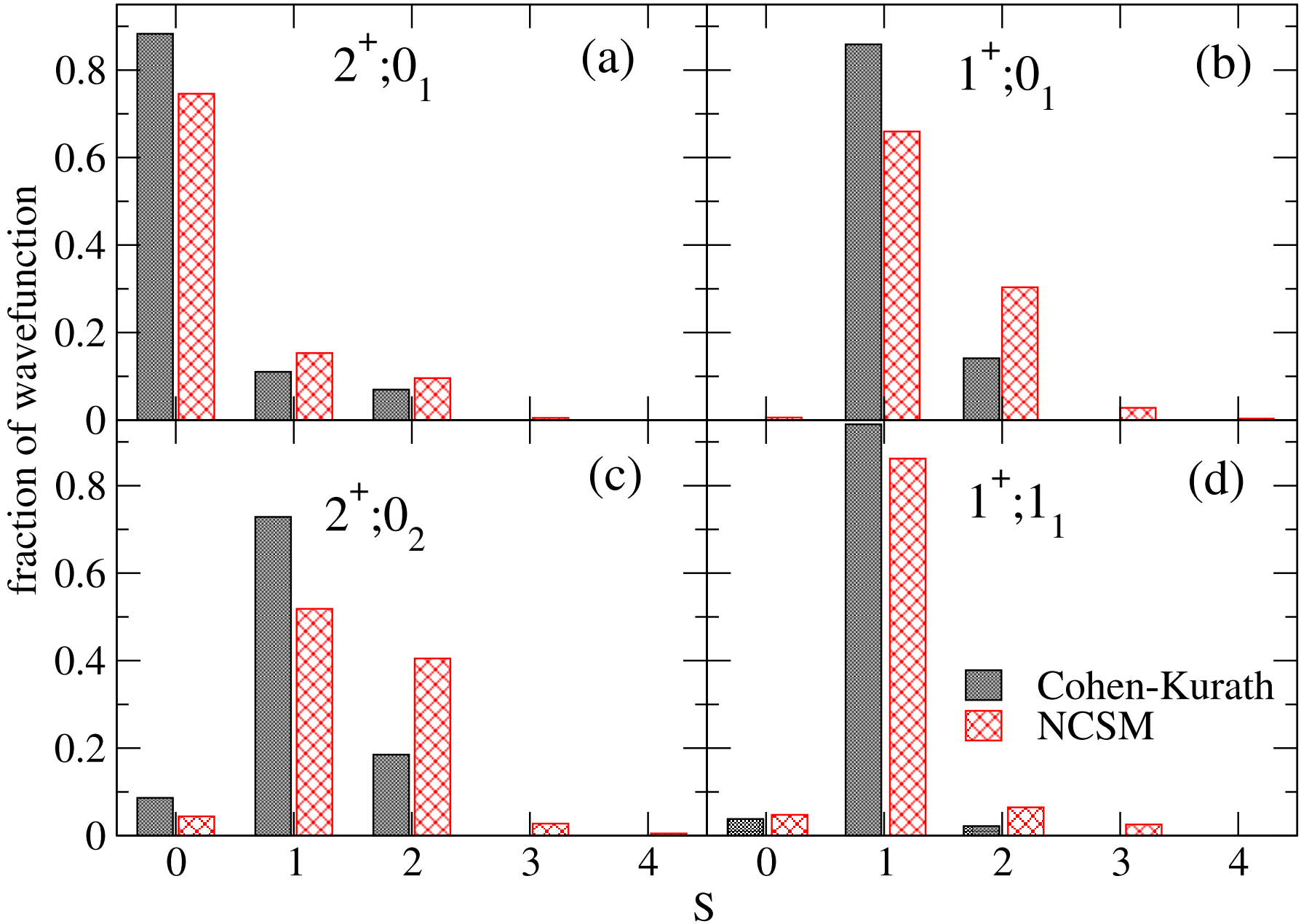
^{12}C

Phenomenological Cohen-Kurath force (1965) in 0*p* shell
m-scheme dimension: 51

NCSM: N3LO chiral 2-body force SRG evolved* to λ = 2.0 fm^{-1}, N_{max} = 6, $\hbar\omega$=22 MeV
m-scheme dimension: 35 million



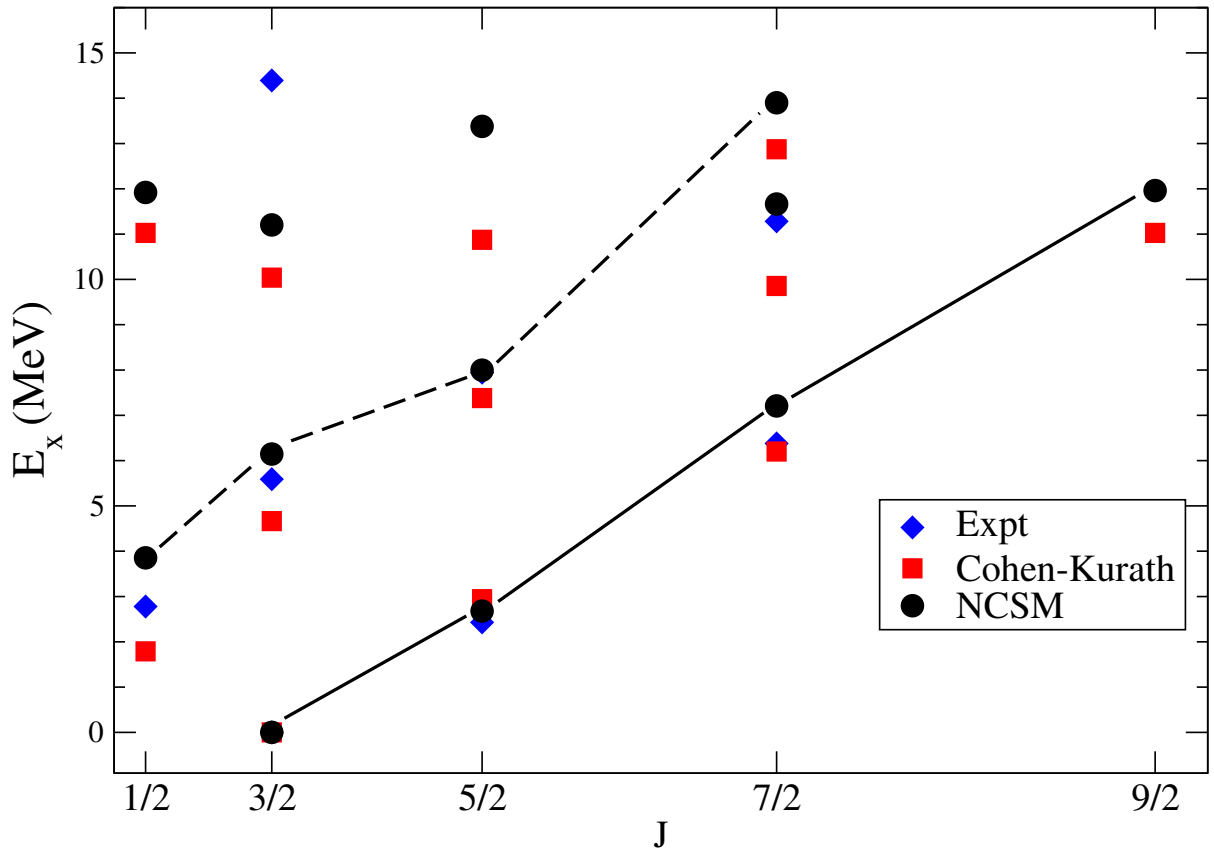

fraction of wavefunction
0.8
0.6
0.4
0.2
0
$0^+;0_1$ (g.s.)
(a)
$2^+;0_1$
(b)
$1^+;0_1$
(c)
Cohen-Kurath
NCSM
$0^+;0_2$
(d)
$2^+;0_2$
(e)
$1^+;1_1$
(f)
0 1 2 3 4
L


fraction of wavefunction
0.8
0.6
0.4
0.2
0
(a)
$2^+;0_1$
(b)
$1^+;0_1$
(c)
$2^+;0_2$
(d)
$1^+;1_1$
Cohen-Kurath
NCSM
0
1
2
3
4
S

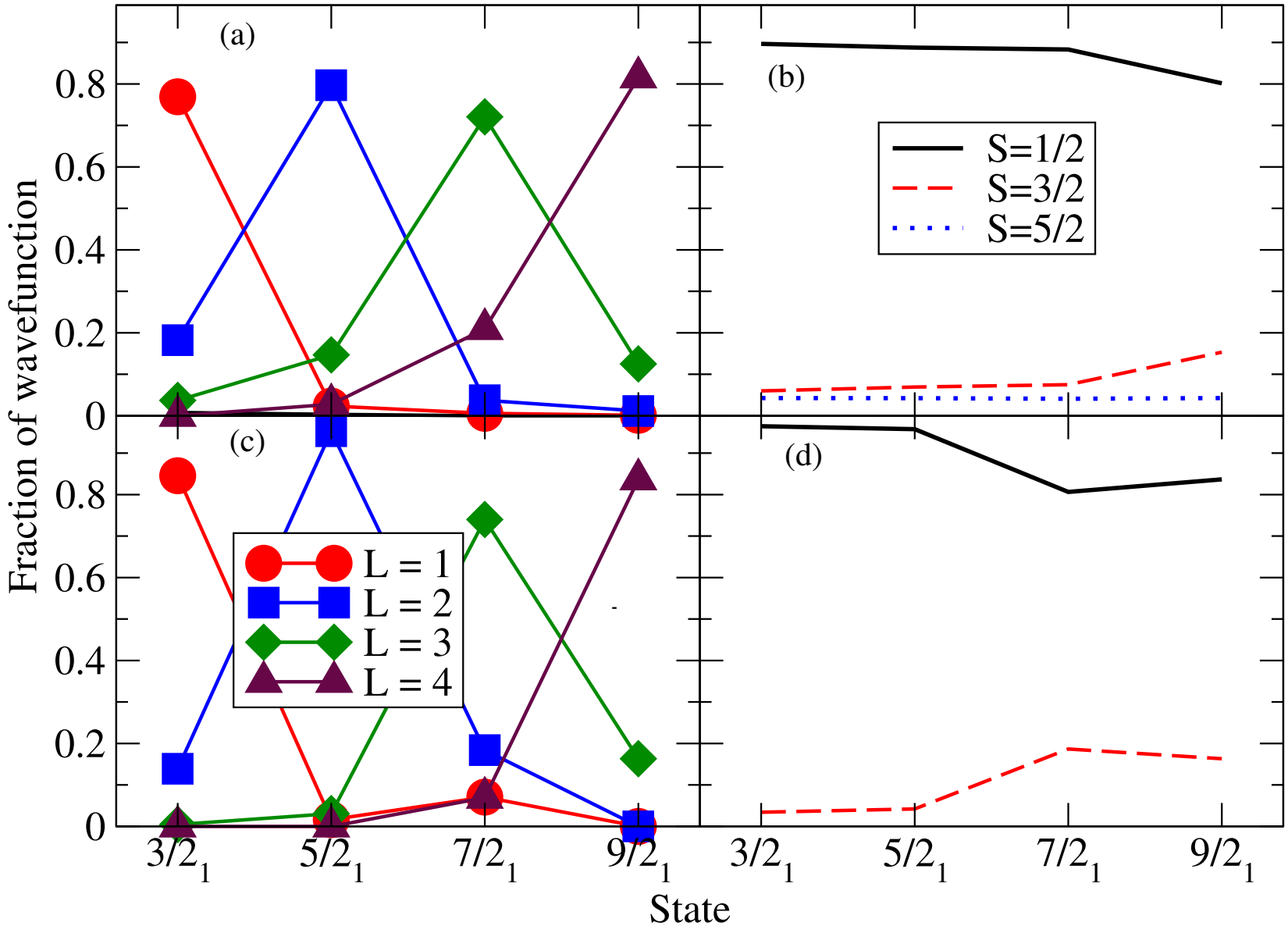
^{9}Be

Phenomenological Cohen-Kurath *m*-scheme dimension: 62

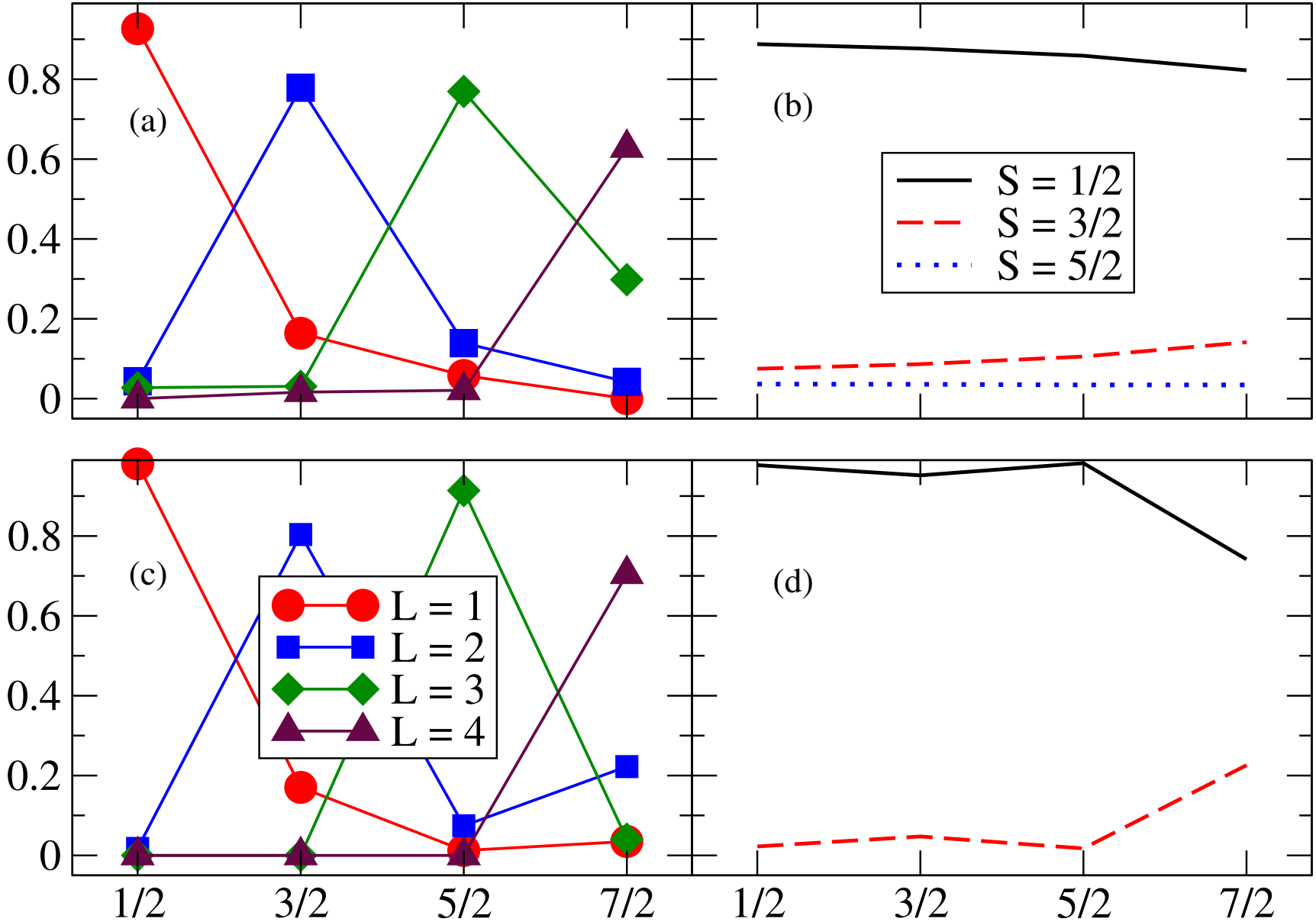
NCSM: N3LO chiral 2-body force SRG evolved to λ = 2.0 fm^{-1}, N_{max} = 6, $\hbar\omega$=22 MeV
m-scheme dimension: 5.2 million

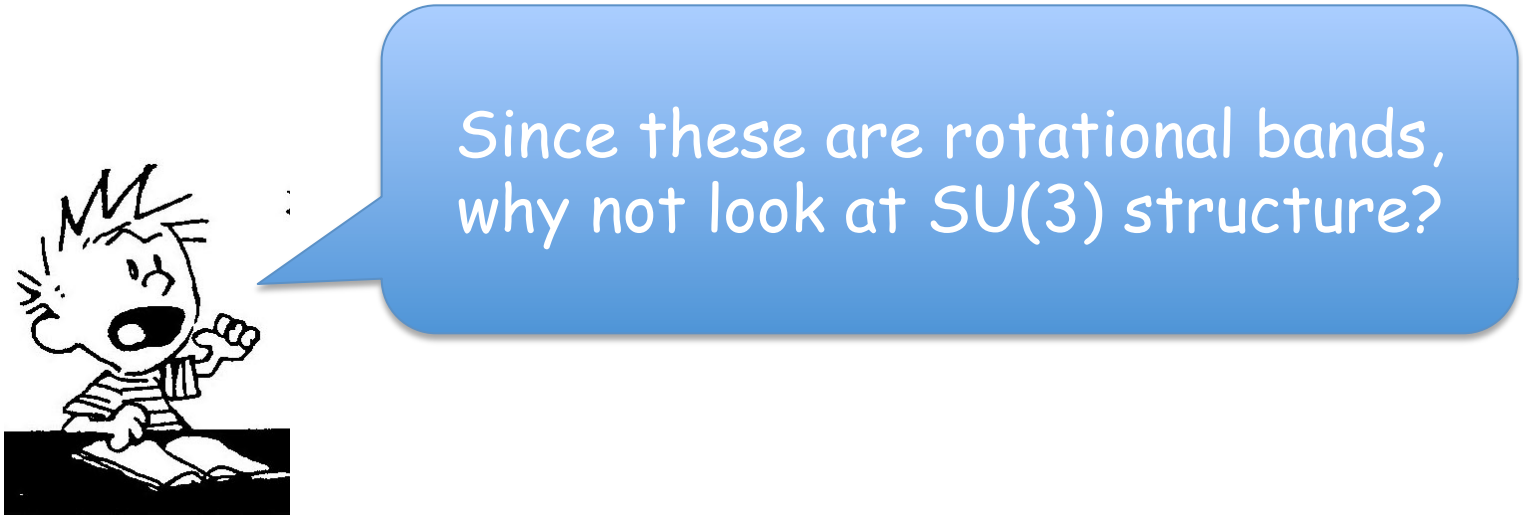


9Be ground state band



9Be excited state band



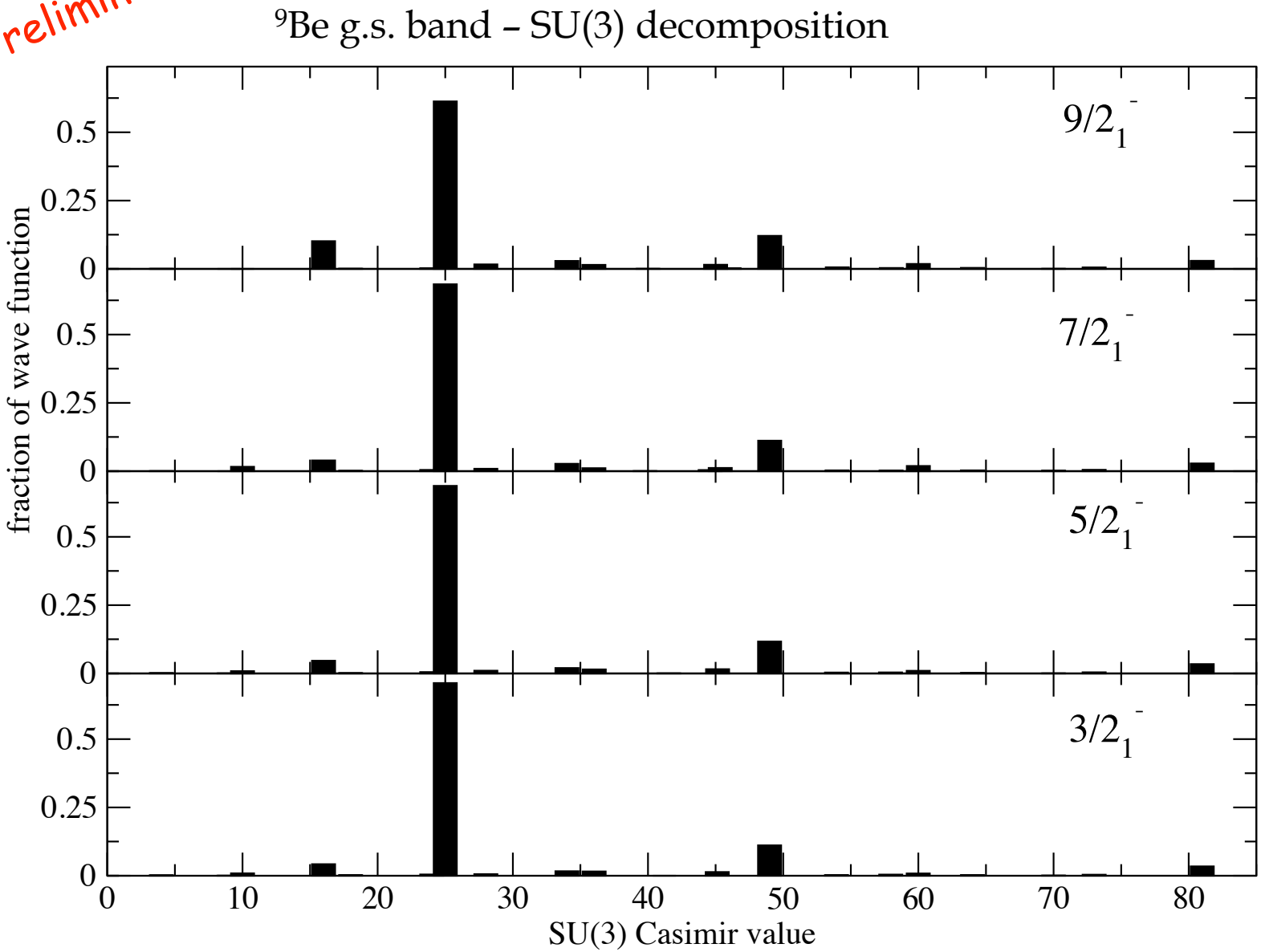


SU(3) Casimir = ¼ (Q_{Ell} .Q_{Ell} + 3 L^2)

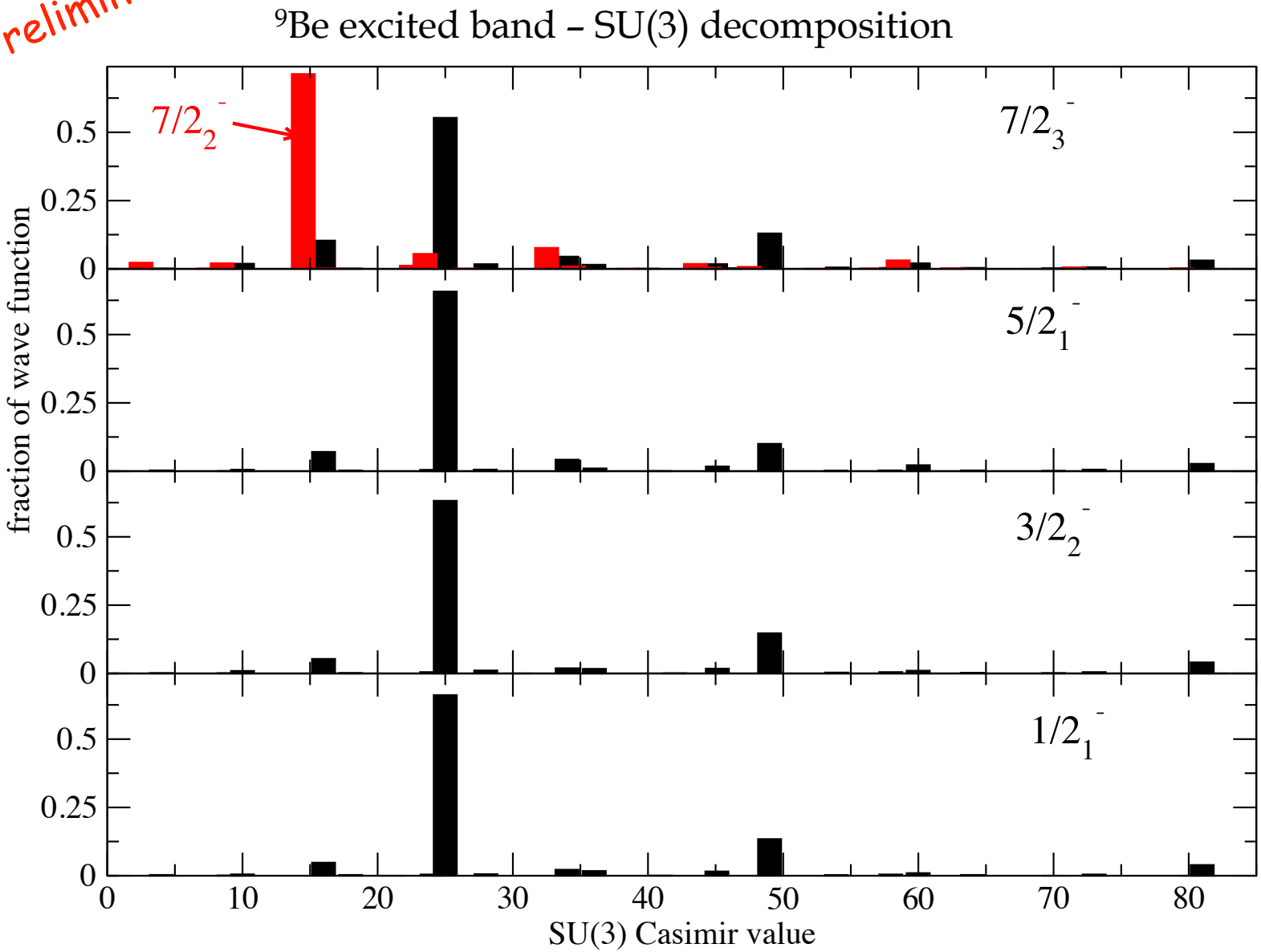
Q_{Ell}= Elliott quadrupole = $(r^2+p^2)Y_2$;
does not contain cross-shell matrix elements

(symplectic operators couple across h.o. shells;
will address in future work)

Preliminary!

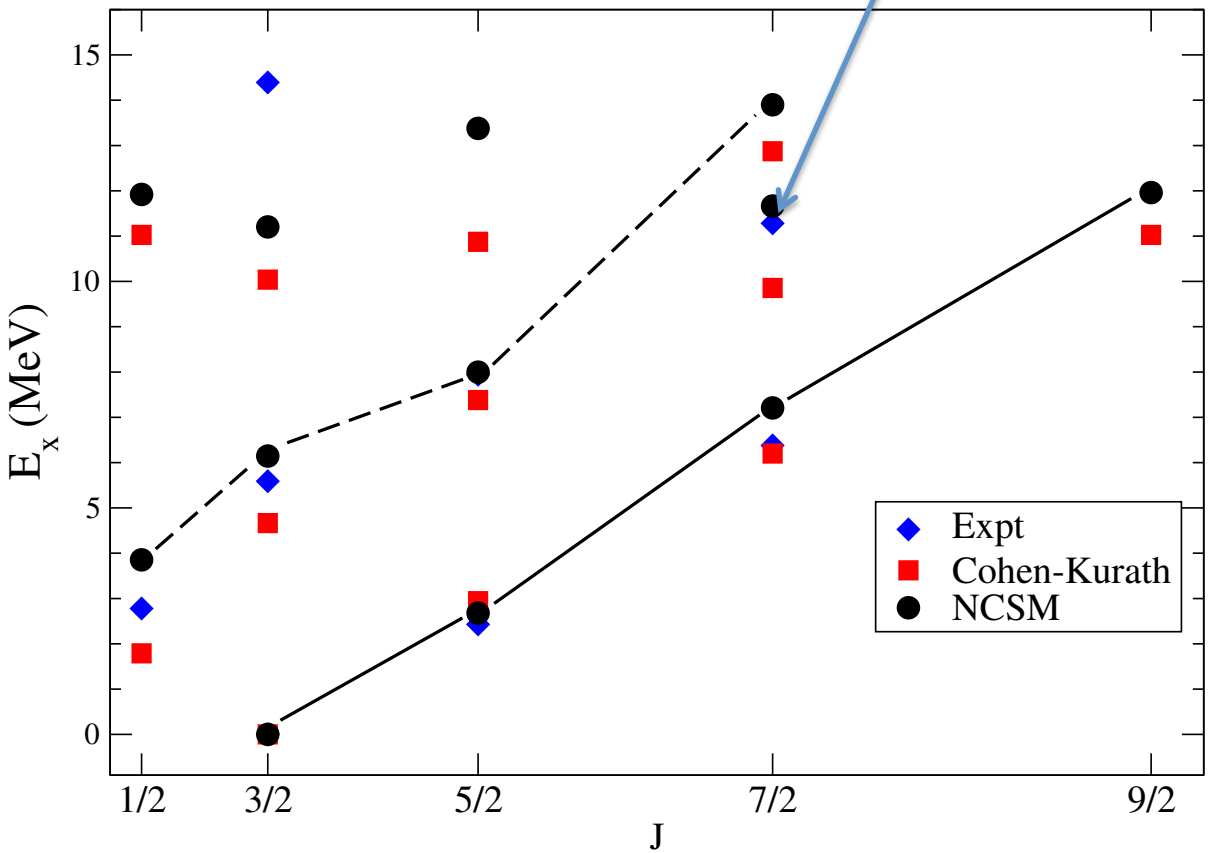


Preliminary!



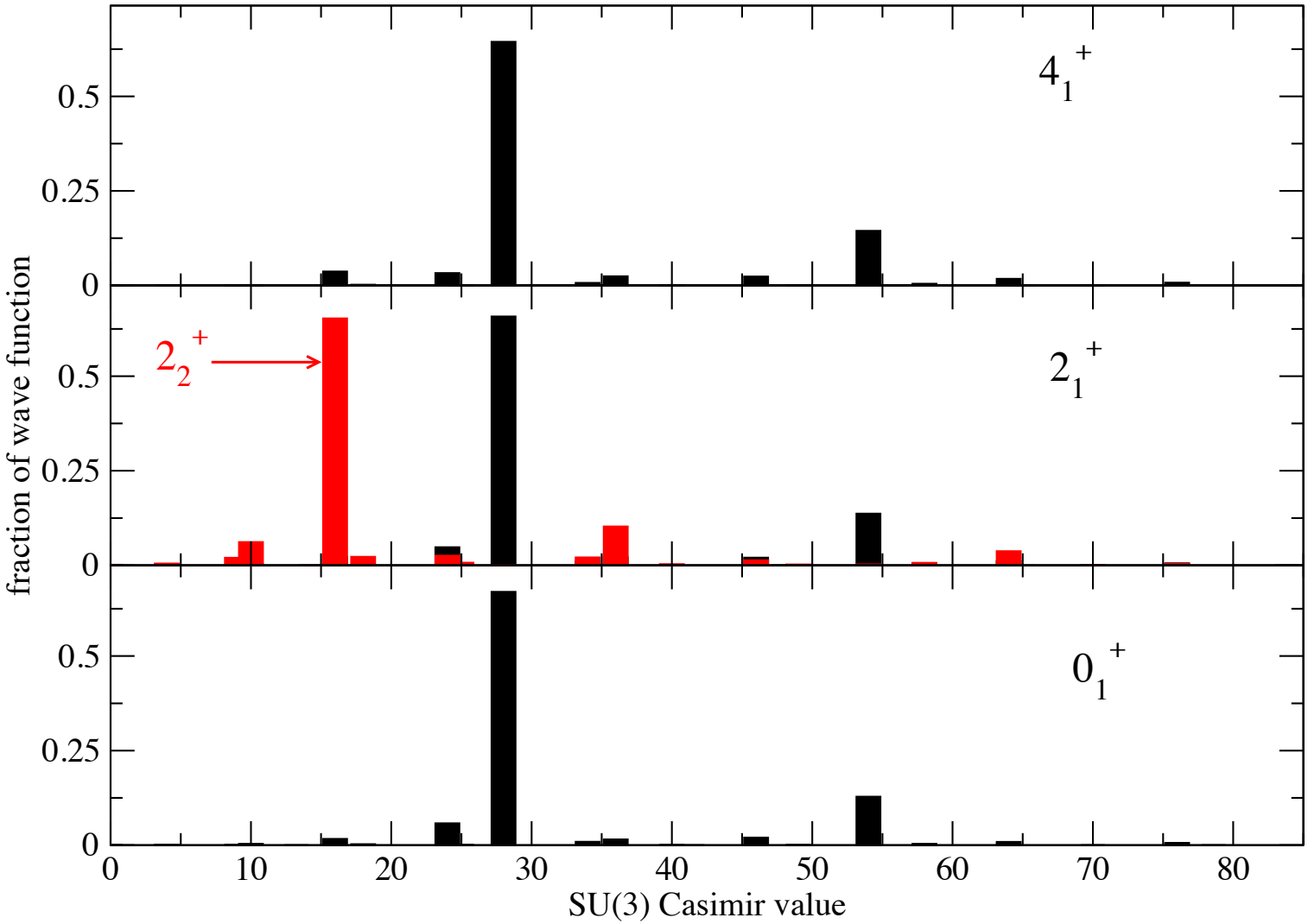
^{9}Be

The "wrong" $7/2^-$ state…



Preliminary!

^{8}Be g.s. band – SU(3) decomposition



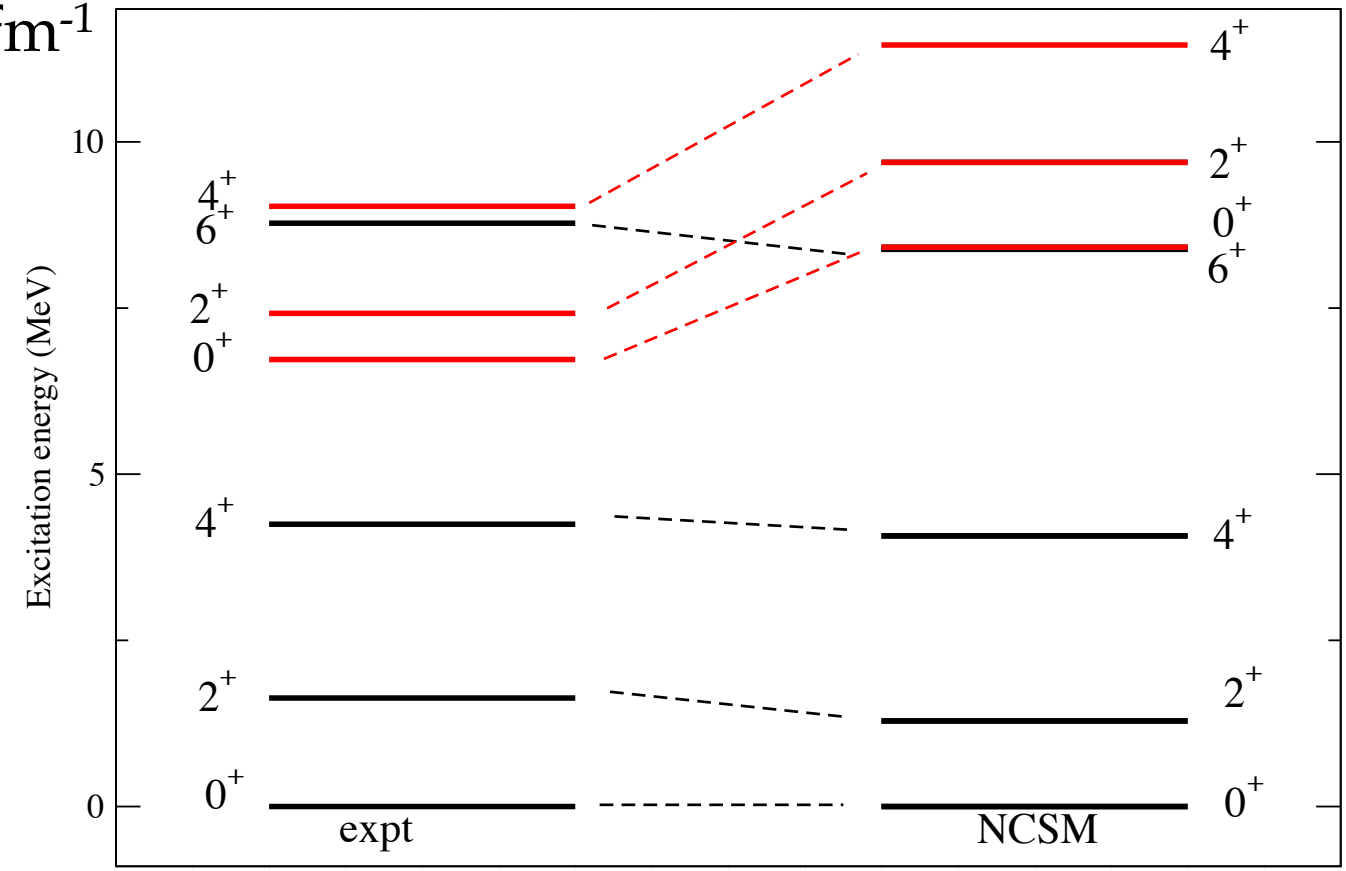
sd-shell nuclei: ^{20}Ne and ^{24}Mg

N_{max}=2
hw= 16 MeV
λ_{SRG}= 2.0 fm^{-1}

Preliminary!

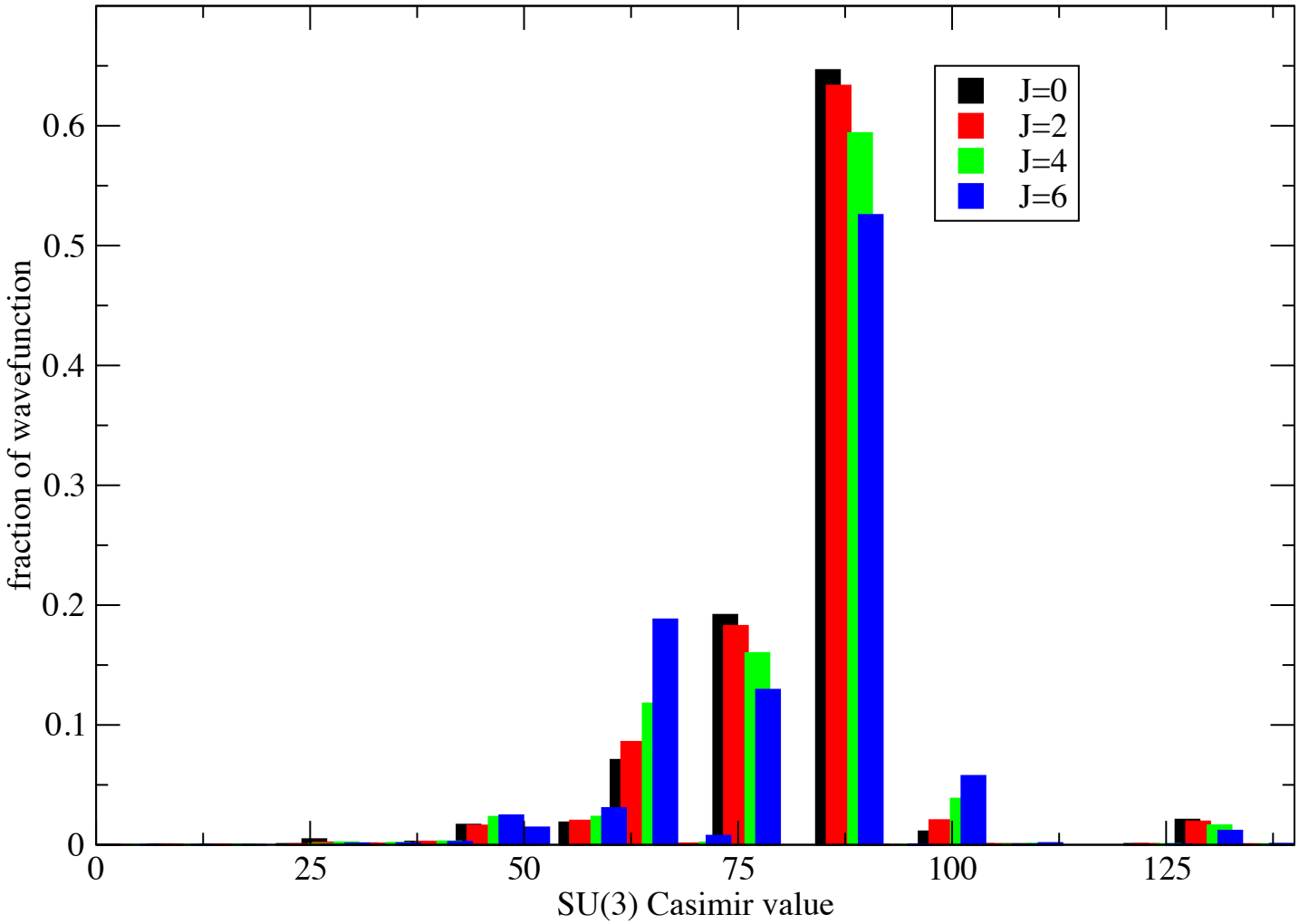
sd-shell nuclei: ^{20}Ne

N_{max}=2
hw= 16 MeV
λ_{SRG}= 2.0 fm^{-1}



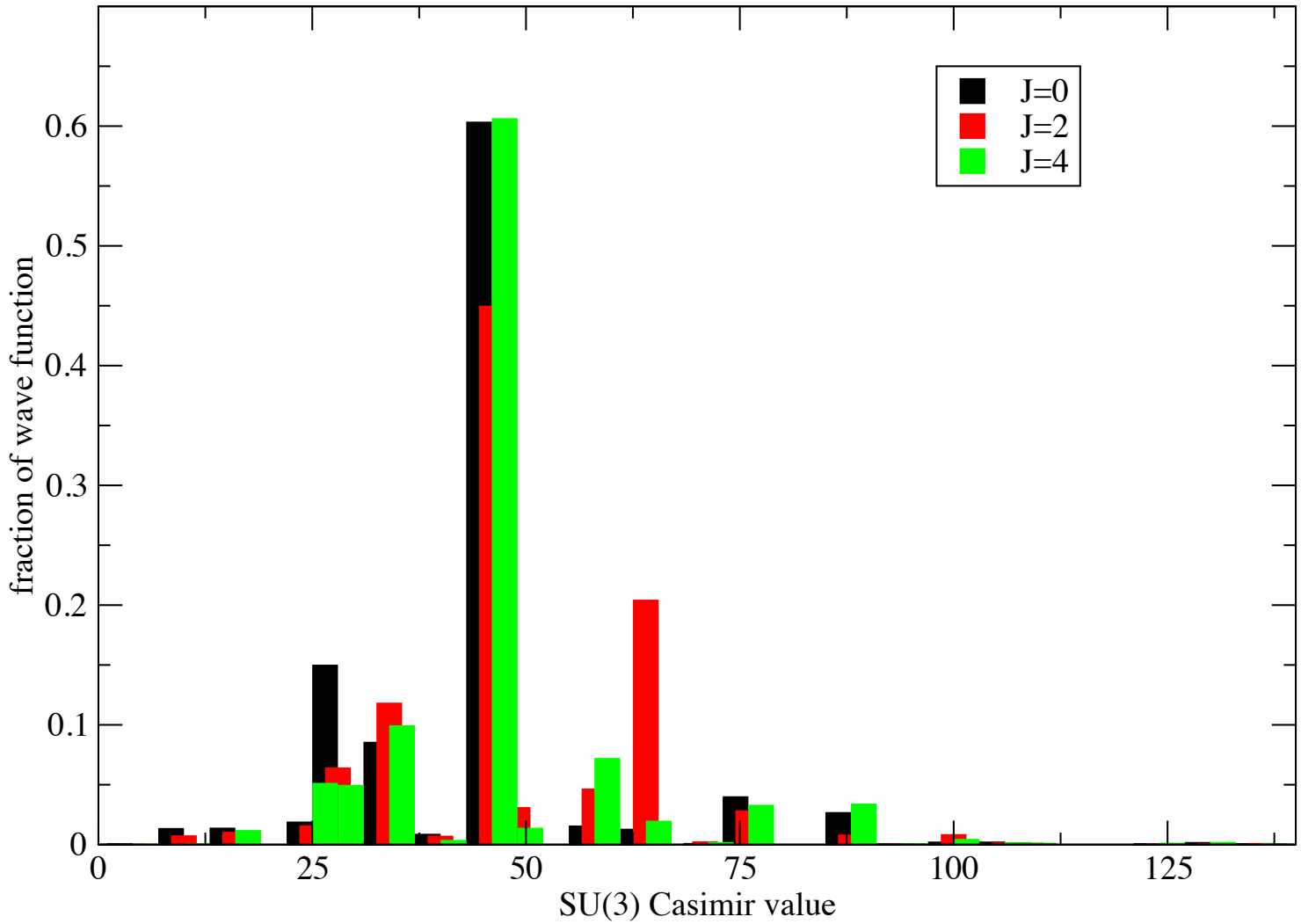
Preliminary!

^{20}Ne g.s. band – SU(3) decomposition



Preliminary!

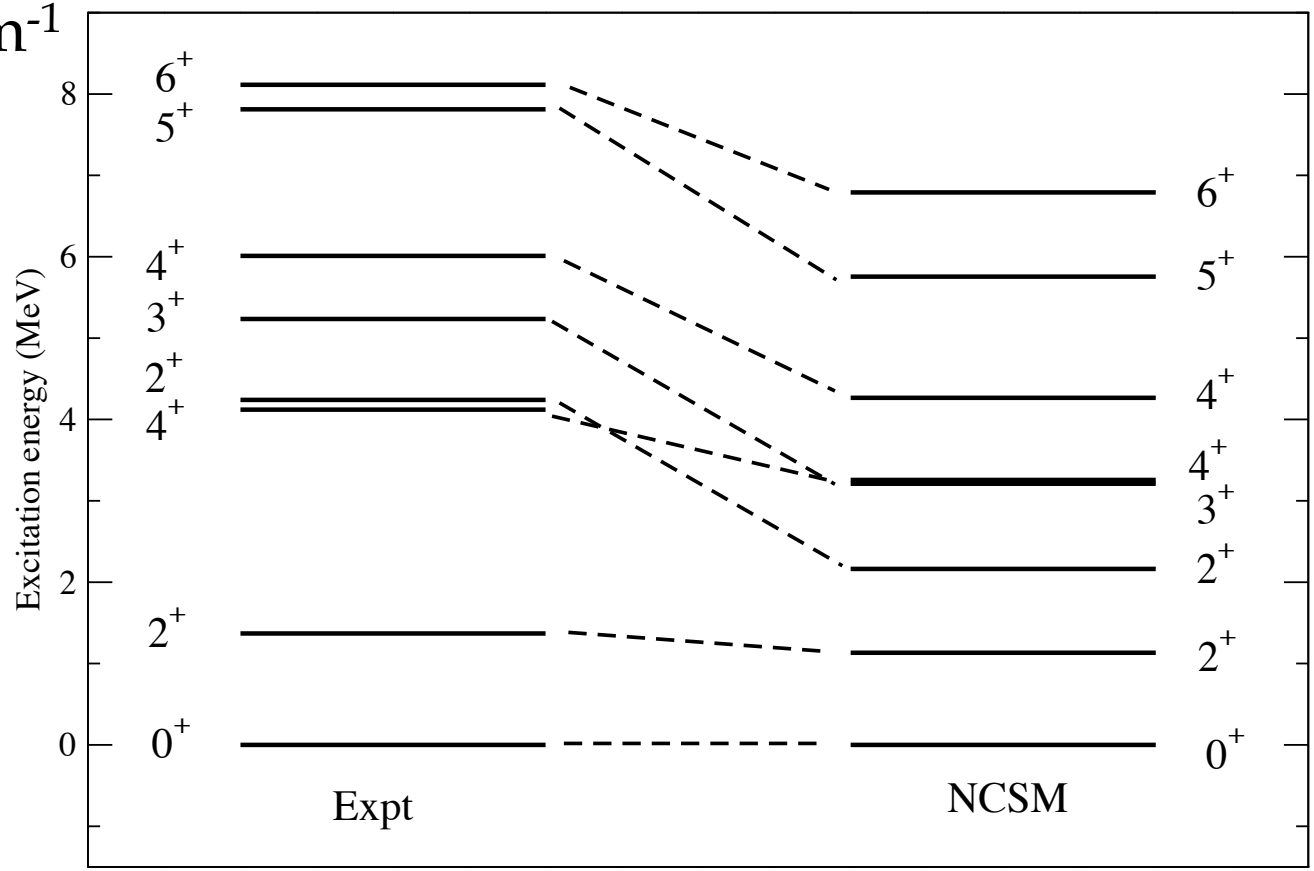
^{20}Ne excited band – SU(3) decomposition



Preliminary!

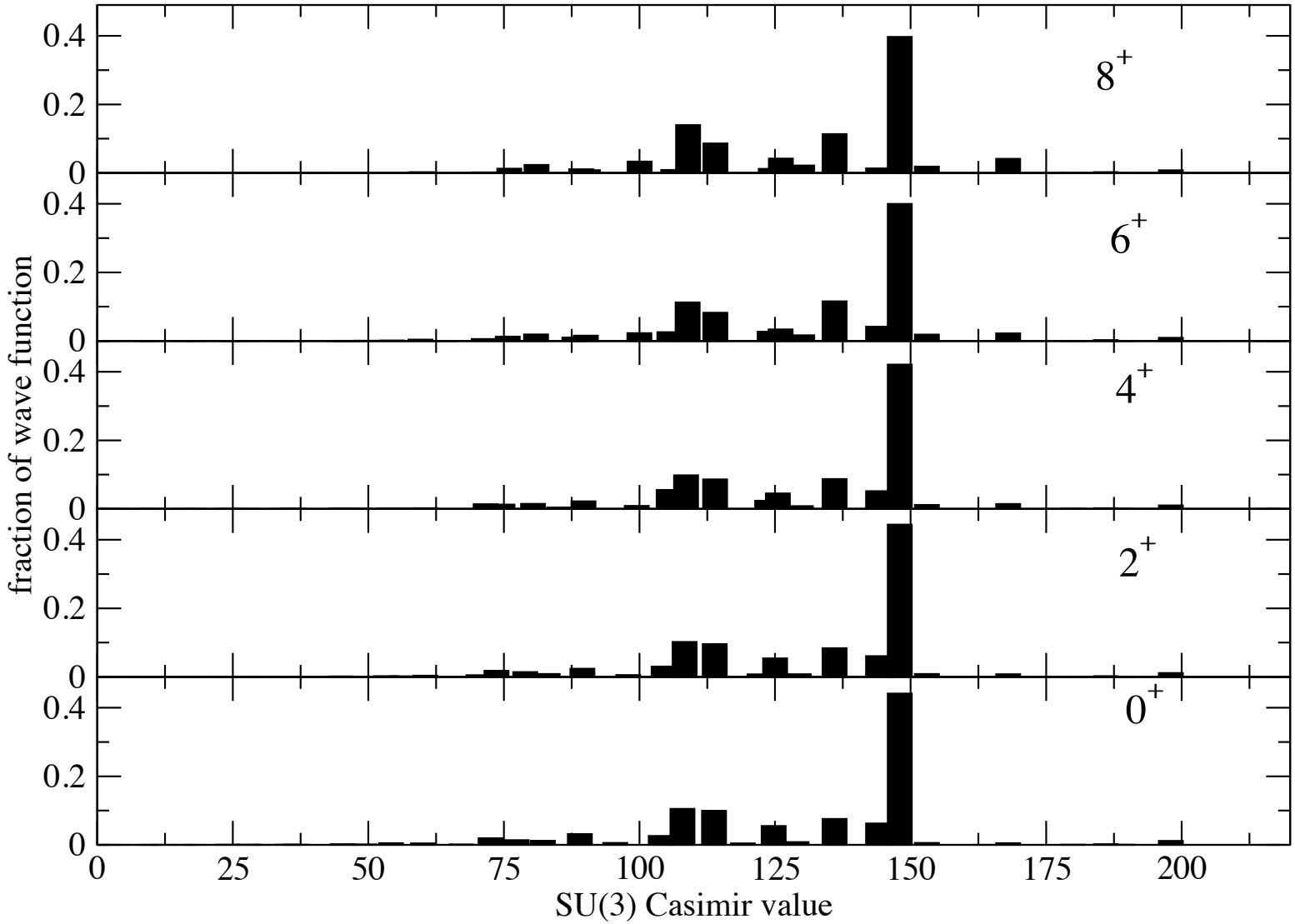
sd-shell nuclei: ^{24}Mg

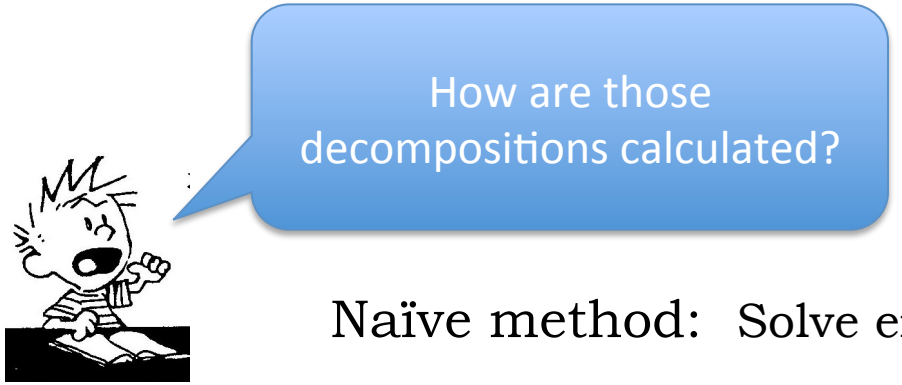
N_{max}=2
hw= 16 MeV
λ_{SRG}= 2.0 fm^{-1}



Preliminary!

^{24}Mg g.s. band – SU(3) decomposition





Naïve method: Solve eigenpair problems, e.g.

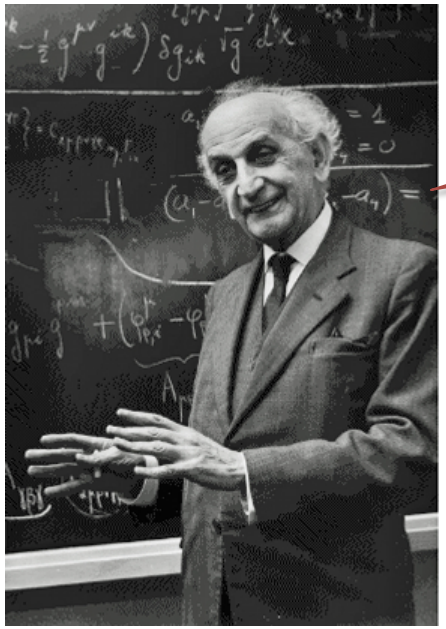
$$\mathbf{H} \mid \Psi_n > = E_n \mid \Psi_n >$$

and

$$\mathbf{L}^2 \mid l; a > = l(l+1) \mid l; a >$$

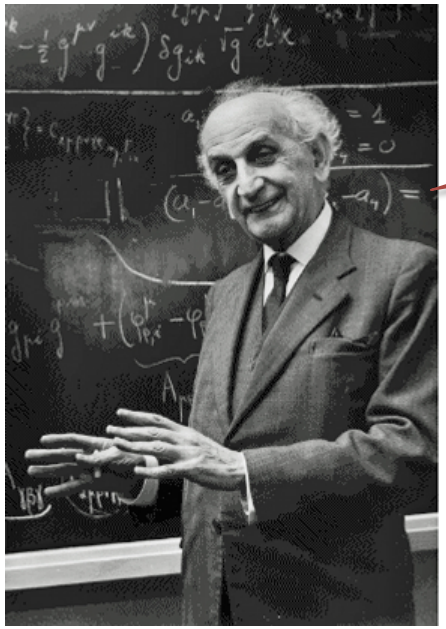
...and then take overlaps, $|< l; a \mid \Psi_n >|^2$

PROBLEM: the spectrum of $\mathbf{L}^2$ is highly degenerate (labeled by a);
Need to sum over all a not orthogonal to $\mid \Psi_n >$!



(Cornelius Lanczos)

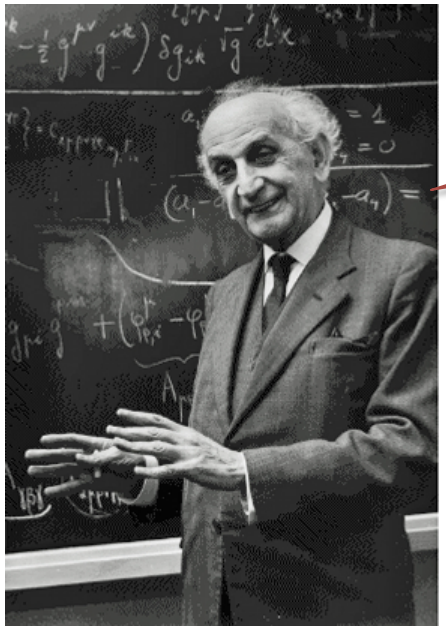
There is another way



(Cornelius Lanczos)

There is another way

The Lanczos Algorithm!



(Cornelius Lanczos)

There is another way

$$\mathbf{A}\vec{v}_1 = \alpha_1\vec{v}_1 + \beta_1\vec{v}_2$$

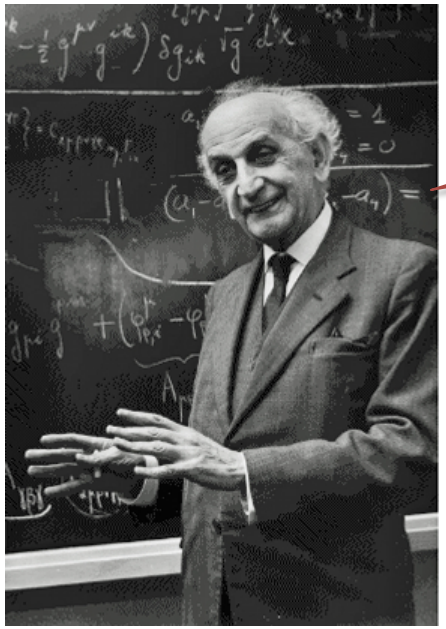
$$\mathbf{A}\vec{v}_2 = \beta_1\vec{v}_1 + \alpha_2\vec{v}_2 + \beta_2\vec{v}_3$$

$$\mathbf{A}\vec{v}_3 = \beta_2\vec{v}_2 + \alpha_3\vec{v}_3 + \beta_3\vec{v}_4$$

$$\mathbf{A}\vec{v}_4 = \beta_3\vec{v}_3 + \alpha_4\vec{v}_4 + \beta_4\vec{v}_5$$

Starting from some initial vector (the "pivot") v_1 , the Lanczos algorithm iteratively creates a new basis (a "Krylov space") in which to diagonalize the matrix **A**.

Eigenvectors are then expressed as a linear combination of the "Lanczos vectors": $|\psi\rangle = c_1 |v_1\rangle + c_2 |v_2\rangle + c_3 |v_3\rangle + \ldots$



(Cornelius Lanczos)

There is another way

Eigenvectors are expressed as a linear combination of the "Lanczos vectors":

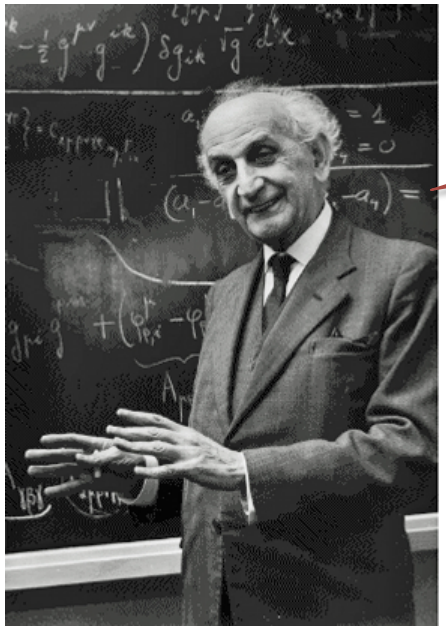
$$|\psi\rangle = c_1 \, |v_1\rangle + c_2 \, |v_2\rangle + c_3 \, |v_3\rangle + \ldots$$

It is easy to read off the overlap of an eigenstate with the "pivot" :

$$|\langle v_1 \, | \psi \rangle|^2 = c_1^2$$

Furthermore, the only eigenvectors (of **A**) that are contained in the Krylov space are those with nonzero overlap with the pivot $|v_1\rangle$.

If **A** is say $\mathbf{L}^2$ then we can efficiently expand any state $|v_1\rangle$ into its components with good L.



(Cornelius Lanczos)

There is another way

This trick has been applied before

Computing strength functions

Caurier, Poves, and Zuker, Phys. Lett. B252, 13 (1990);
PRL **74**, 1517 (1995)
Caurier *et al*, PRC **59**, 2033 (1999)
Haxton, Nollett, and Zurek, PRC **72**, 065501 (2005)

Decomposition of wavefunction into SU(3) components, looking at effect of spin-orbit force:
V. Gueorguiev, J. P Draayer, and C. W. J., PRC **63**, 014318 (2000).

Present calculations carried out using BIGSTICK shell-model code:
Johnson, Ormand, and Krastev, Comp. Phys. Comm. **184**, 2761 (2013).

Large scale configuration interaction calculations for nuclear structure

Summary and looking forward

Bigstick is a powerful configuration-interaction shell model code coming into maturity. We can now reach the largest dimensions of other CI codes, using significantly less computational resources. (Still work to be done to fully optimize for N_{max} calculations and three-body forces.) We hope to make the code publically available in the near future.

As a sample application, we can decompose wave functions using operators, usually Casimirs of groups. This gives us an "x-ray" into the wavefunctions and illustrate (a) overall similarity with phenomenological calculations and (b) clearly show the fingerprint of "intrinsic states."

"More work to be done!"