'Easy' projection of angular momentum

Calvin W. Johnson

I discuss how to efficiently extract angular-momentum projected matrix elements under certain circumstances.

I. INITIAL DISCUSSION

Angular momentum projection is computationally consuming. Sometimes, however, one can take a shortcut.

Suppose we have a state which has definite J_z-component (or L_z-component), which we label as $|\Psi_M\rangle$. We can always expand this in terms of its components with good J (or L):

$$|\Psi_M\rangle = \sum_{J=M}^{J_{max}} c_J |\Psi_{JM}\rangle \tag{1}$$

where J_{max} is the maximum J for the system (say for N particles).

Now let's rotate by an angle β about the y-axis; in terms of Euler angles this is letting angles α, γ about the z-axis be zero. We have

$$\hat{R}(0,\beta,0)|\Psi_M\rangle = \sum_J c_J \hat{R}(0,\beta,0)|\Psi_{JM}\rangle = \sum_J c_J \sum_{M'} \mathcal{D}^J_{M',M}(0,\beta,0)|\Psi_{JM'}\rangle = \sum_J c_J \sum_{M'} d^J_{M',M}(\beta)|\Psi_{JM'}\rangle \tag{2}$$

where $\mathcal{D}^J_{M',M}(\alpha,\beta,\gamma)$ is the Wigner D-function and $d^J_{M',M}(\beta)$ is the Wigner little-d function (see, e.g., Edmonds). Note: this only works because our original state had a unique value of M; otherwise the mixing would be much more complicated.

Now suppose we do this for several different angles β_i, where $i = M...J_{max}$. Then we have

$$\hat{R}(0,\beta_i,0)|\Psi_M\rangle = \sum_J c_J \sum_{M'} d^J_{M',M}(\beta)|\Psi_{JM'}\rangle. \tag{3}$$

Next, we define the matrix $A_{iJ} = d^J_{M,M}(\beta_i)$. Because both J and i run over the same dimensions, it is a square matrix. Assuming this is invertible (and that should be true if the angles β_i are chosen reasonably), then

$$\sum_i (A^{-1})_{Ji} \hat{R}(0,\beta_i,0)|\Psi_M\rangle = c_J|\Psi_{JM}\rangle \tag{4}$$
$$+ \sum_J c_J \sum_{M'\neq M} (A^{-1})_{Ji} d^J_{M',M}(\beta_i)|\Psi_{JM'}\rangle$$

The first term is the one we want. If we take any matrix element of an angular momentum scalar, we get, for example,

$$\sum_i (A^{-1})_{Ji} \langle\Phi'_M|\hat{R}(0,\beta_i,0)|\Psi_M\rangle = c'_J c_J \langle\Phi'_{JM}|\Psi_{JM}\rangle \tag{5}$$

and

$$\sum_i (A^{-1})_{Ji} \langle\Phi'_M|\hat{H}\hat{R}(0,\beta_i,0)|\Psi_M\rangle = c'_J c_J \langle\Phi'_{JM}|\hat{H}|\Psi_{JM}\rangle \tag{6}$$

because the matrix elements with all the other terms vanish.

Another advantage is one can get the matrix elements for all J almost simultaneously (although with a little planning it is also possible to do that as well with the 'standard' projection operator).

What if we want the matrix element of a nonscalar operator, such as a transition operator. Well, that's a lot harder, and may be the Achilles' heel of this technique, because, of course, there's more to life than just eigenvalues! To do this I think you have to do the following:

1. Do the projection, as above, on the right.
2. But also do the projection, as above, on the left.
3. You also have to insert a projection operator on M inbetween as well. It also is a finite sum, and in this case, because it's a Fourier series, can be inverted analytically.

This dramatically increases the number of points; if $M = 0$ (assuming J is an integer) then the number of points needed is $(J_{max}+1)^2(2J_{max}+1)$. On the other hand, it still beats traditional projection, because for a transition operator you need to do a 3-dimensional integral on both sides.

II. REVISED DISCUSSION (JULY 2015)

Now let's get down to the task in earnest.

In general, we want to compute the general norm and Hamiltonian matrices

$$N^J_{MK}(ab) = \langle \Psi_a | \hat{P}^J_{MK} | \Psi_b \rangle, \tag{7}$$

$$H^J_{MK}(ab) = \langle \Psi_a | \hat{H} \hat{P}^J_{MK} | \Psi_b \rangle, \tag{8}$$

where I allow for different initial and final Slater determinants. Note: it is important that they not be orthogonal to each other, or else our methods for computing the overlaps won't work. The projection operator is the usual one:

$$\hat{P}^J_{MK} = \frac{2J+1}{8\pi^2} \int d\Omega \mathcal{D}^{J*}_{MK}(\Omega) \hat{R}(\Omega) \tag{9}$$

where Ω represents the Euler rotation angles α, β, γ, $d\Omega = d\alpha \sin\beta d\beta d\gamma$,

$$\hat{R}(\Omega) = \exp(i\alpha \hat{J}_z) \exp(i\beta \hat{J}_y) \exp(i\gamma \hat{J}_z), \tag{10}$$

and we have the Wigner D-function

$$\mathcal{D}^J_{MK}(\Omega) = e^{i\alpha M} d^J_{MK}(\beta) e^{i\gamma K}. \tag{11}$$

Here $d^J_{MK}(\beta)$ is of course the Wigner little-d function.

The projection operator works because, for example,

$$\langle \Psi_a | \exp(i\alpha \hat{J}_z) \exp(i\beta \hat{J}_y) \exp(i\gamma \hat{J}_z) | \Psi_b \rangle = \sum_{JMK} e^{i\alpha M} d^J_{MK}(\beta) e^{i\gamma K} N^J_{KM}. \tag{12}$$

Normally one extracts the projected norm and Hamiltonian matrices by integration, using orthogonality of the functions. But because there are a finite set of values of J, M, K, we instead treat this as a linear algebra problem, namely, we evaluate

$$N_{ijk}(ab) = \langle \Psi_a | \exp(i\alpha_i \hat{J}_z) \exp(i\beta_j \hat{J}_y) \exp(i\gamma_k \hat{J}_z) | \Psi_b \rangle \tag{13}$$

at some set of points $\{\alpha_i, \beta_j, \gamma_k\}$, and as long as the matrix

$$G_{ijk,JMK} = e^{i\alpha_i M} d^J_{MK}(\beta_j) e^{i\gamma_k K} \tag{14}$$

is nonsingular, we can solve the linear algebra problem

$$N_{ijk} = \sum_{JMK} G_{ijk,JMK} N^J_{MK}. \tag{15}$$

For our investigation, we first will solve the straightforward way, and then later will try to solve using the minimal number of points.

A. Numerical solution I

No matter how we solve, we must start by choosing a set of points $\{\Omega\} = \{\alpha_i, \beta_j, \gamma_k\}$. Because for all the cases we will consider this will at most a few tens of thousands of points, we can simply modify our routines to (1) generate the set of points $\{\Omega\}$ and then (2) to generate and store N_{ijk} (and also H_{ijk}). Let's assume we have done this.

We start off by inverting on each of the indices separately. This is not the most efficient, but it is the quickest and most likely to work. To do this, we chose the angles $\{\alpha_i\}, \{\beta_j\}, \{\gamma_k\}$ independent of each other. Choosing some $J_{\max}$, we choose $2J_{\max}+1$ values of $\{\alpha_i\}$ and of $\{\gamma_k\}$ and invert separately $\exp(i\alpha_i M)$ and $\exp(i\gamma_k K)$. This can be done analytically by using the identity, for some integer N

$$\frac{1}{N} \sum_{k=1}^{N} \exp\left(i \frac{2\pi M k}{N} \right) = \delta_{M,0}. \tag{16}$$

Thus we choose, e.g. $\gamma_k = (k-1)\frac{2\pi}{2J_{\max}+1}, k = 1, 2J_{\max}+1$, and we can define the inverse solution

$$Z_{Kk} = \frac{1}{2J_{\max}+1}\exp(-iK\gamma_k). \tag{17}$$

We make the identical choice for $\{\alpha_i\}$ and introduce Z_{Mi}. We can then define the intermediate quantity

$$N_{j,MK} = \sum_{ik} Z_{Mi}Z_{Kk}N_{ijk} = \sum_J d^J_{MK}(\beta_j)N^J_{MK}. \tag{18}$$

Before we do that, of course, we have to choose the set of $\{\beta_j\}$. They have to be such that (18) is solvable for all choices of M, K. This might be a little tricky, as we have the selection rule that $J \geq |M|, |K|$, so that for most cases the system is technically overdetermined, which means that there are more equations than there are variables. However, if our decomposition is correct, then any subset of equations should give us the same, or nearly the same, solution. One way to tackle this is then through singular value decomposition (SVD), but keep in mind that we have to pay attention as we do this.

We don't have to actually call an SVD routine, but instead can simply construct the object we would use in SVD, namely

$$\Delta^{J'J}_{MK} = \sum_j d^{J'}_{MK}(\beta_j)d^J_{MK}(\beta_j), \tag{19}$$

with $J, J' \geq |M|, |K|$. The matrix $\Delta^{J'J}$ is real and symmetric. **It is a matrix with fixed M, K and it's indices are J' and J.** It should have only nonzero (and nonnegative) eigenvalues. Given a set of $\{\beta_i\}$, construct Δ for all possible M, K and check the eigevalues. The real condition we want is the dynamic range of the eigenvalues, that is the ratio of the smallest to the largest eigenvalues. As long as this is larger than, I presume, 0.01 or 0.001 we should be okay. We will have to experiment on this. The eigensolver to use for this is the LAPACK routine `dsyev`.

The number of J values we choose is $N = J_{\max}+1$ if an even system and $= J_{\max}+1/2$ if an odd number of nucleons. The most obvious choice of values is

$$\beta_j = (j-1)\frac{\pi}{N}, \tag{20}$$

though we will have to experiment with this as well to see if the Δ matrices are all invertible.

Now to solve we construct another intermediate matrix,

$$\tilde{N}_{J',MK} = \sum_j d^{J'}_{MK}(\beta_j)N_{j,MK}. \tag{21}$$

Then we simply solve

$$\sum_J \Delta^{J'J}_{MK}N^J_{MK} = \tilde{N}_{J',MK} \tag{22}$$

for N^J_{MK} using a suitable routine, probably the LAPACK routine `zgesv`.

To summarize:

For a given $J_{\max}$ (see below), choose $\gamma_k = (k-1)\frac{2\pi}{2J_{\max}+1}, k = 1, 2J_{\max}+1$ and similarly for α_i, and construct the matrix (17).

Construct the β_j as above; construct, for all allowed values of M, K the $\Delta^{J'J}_{MK}$, and find their eigenvalues; check that for a given M, K the dynamic range is not too large (subject ot experimentation).

With the chosen set of $\{\alpha_i, \beta_j, \gamma_k\}$, obtain N_{ijk} and H_{ijk}.

Construct $N_{j,MK}$ and $H_{j,MK}$ as in the first part of (18) using the **Z** matrices.

Finally construct $\tilde{N}_{J',MK}$ and $\tilde{H}_{J',MK}$ according to (21) and solve by (22) using `zgesv`.

As a test, we will see if the matrices coming out here look like the matrices from integration.

B. Look aheads

Eventually we will want to create some fast routines to find out what are the values of, say, $J_{\max}$. The first step is to find the minimum and maximum values of M, K. This can be done by computing

$$N_M = \sum_i Z_{mi}\langle\Psi|\exp(i\alpha_i\hat{J}_z)|\Psi\rangle \tag{23}$$

which should go quite fast. With those limits we can then find the full N^J_{MM} which will give us min and max J. But this can wait until later.