ang_mom.py

Angular Momentum Matrices

Cartesian Components J_x, J_y, J_z

ElecSus Library Documentation

Abstract

This document provides comprehensive documentation for `ang_mom.py`, which constructs the matrix representations of angular momentum operators in Cartesian coordinates. These matrices are essential for computing fine structure, hyperfine structure, and Zeeman interactions in atomic physics calculations.

Contents

1 Theoretical Foundation

1.1 Angular Momentum in Quantum Mechanics

Axiom 1 (Hermiticity of Observables)**.** *Physical observables correspond to Hermitian operators. The angular momentum components J_x, J_y, J_z satisfy:*

$$J_i^\dagger = J_i \tag{1}$$

Theorem 1 (Ladder Operator Decomposition)**.** *The Cartesian angular momentum components can be expressed in terms of ladder operators:*

$$J_x = \frac{1}{2}(J_+ + J_-) \tag{2}$$

$$J_y = \frac{1}{2i}(J_+ - J_-) = -\frac{i}{2}(J_+ - J_-) \tag{3}$$

$$J_z = \frac{1}{2}[J_+, J_-]_- = \frac{1}{2}(J_+J_- - J_-J_+) \tag{4}$$

where $J_\pm = J_x \pm iJ_y$.

Proof. From $J_+ = J_x + iJ_y$ and $J_- = J_x - iJ_y$:

$$J_+ + J_- = 2J_x \implies J_x = \frac{1}{2}(J_+ + J_-) \tag{5}$$

$$J_+ - J_- = 2iJ_y \implies J_y = \frac{1}{2i}(J_+ - J_-) \tag{6}$$

For J_z, use $[J_+, J_-] = 2\hbar J_z$ (in units $\hbar = 1$). □

Definition 1 (Adjoint Relationship)**.** *The lowering operator is the Hermitian adjoint of the raising operator:*

$$J_- = J_+^\dagger \tag{7}$$

For real matrix representations, this reduces to the transpose: $J_- = J_+^T$.

Corollary 1 (Hermiticity Verification)**.** *Using $J_- = J_+^\dagger$:*

$$J_x^\dagger = \frac{1}{2}(J_+^\dagger + J_-^\dagger) = \frac{1}{2}(J_- + J_+) = J_x \tag{8}$$

$$J_y^\dagger = \frac{i}{2}(J_+^\dagger - J_-^\dagger) = \frac{i}{2}(J_- - J_+) = J_y \tag{9}$$

confirming J_x and J_y are Hermitian.

2 Line-by-Line Code Analysis

2.1 Module Imports

```python
from numpy import transpose,dot
import ang_mom_p
```

Import NumPy's transpose and dot product, plus the ladder operator module.

2.2 The J_x Function

```
def jx(jj):
    jp=ang_mom_p.jp(jj)
    jm=transpose(jp)
    jx=0.5*(jp+jm)
    return jx
```

$$J_x = \frac{1}{2}(J_+ + J_-) \tag{10}$$

Construct J_x from the symmetric combination of ladder operators. The transpose gives $J_- = J_+^T$ since J_+ is real.

2.3 The J_y Function

```
def jy(jj):
    jp=ang_mom_p.jp(jj)
    jm=transpose(jp)
    jy=0.5j*(jm-jp)
    return jy
```

$$J_y = \frac{i}{2}(J_- - J_+) = -\frac{i}{2}(J_+ - J_-) \tag{11}$$

The antisymmetric combination with factor $i/2$ yields a Hermitian matrix. Note the sign convention: $J_y = 0.5j \cdot (J_- - J_+)$.

2.4 The J_z Function

```
def jz(jj):
    jp=ang_mom_p.jp(jj)
    jm=transpose(jp)
    jz=0.5*(dot(jp,jm)-dot(jm,jp))
    return jz
```

$$J_z = \frac{1}{2}(J_+J_- - J_-J_+) = \frac{1}{2}[J_+, J_-] \tag{12}$$

The commutator of ladder operators gives the z-component. This yields a diagonal matrix with eigenvalues $m = j, j-1, \ldots, -j$.

3 Mathematical Properties

3.1 Matrix Structure

Theorem 2 (Matrix Forms). *For angular momentum j:*

- *J_z: Diagonal with entries $\{j, j-1, \ldots, -j\}$*
- *J_x: Real symmetric, tri-diagonal*
- *J_y: Purely imaginary antisymmetric (Hermitian), tri-diagonal*

3.2 Spin-1/2 Example

For $j = 1/2$, the Pauli matrix representation (in units $\hbar = 1$):

$$J_x = \frac{1}{2}\sigma_x = \frac{1}{2}\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{13}$$

$$J_y = \frac{1}{2}\sigma_y = \frac{1}{2}\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \tag{14}$$

$$J_z = \frac{1}{2}\sigma_z = \frac{1}{2}\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{15}$$

3.3 Verification of Commutation Relations

The constructed matrices satisfy:

$$[J_x, J_y] = iJ_z, \quad [J_y, J_z] = iJ_x, \quad [J_z, J_x] = iJ_y \tag{16}$$

(in units where $\hbar = 1$).

4 Applications in ElecSus

4.1 Fine Structure Interaction

The spin-orbit coupling uses:

$$H_{FS} = A_{FS}\vec{L} \cdot \vec{S} = A_{FS}(L_x S_x + L_y S_y + L_z S_z) \tag{17}$$

4.2 Hyperfine Structure

The magnetic dipole hyperfine interaction:

$$H_{HFS} = A_{HFS}\vec{I} \cdot \vec{J} = A_{HFS}(I_x J_x + I_y J_y + I_z J_z) \tag{18}$$

4.3 Zeeman Interaction

The magnetic field coupling:

$$H_Z = \mu_B B_z (g_L L_z + g_S S_z + g_I I_z) \tag{19}$$

uses the z-components directly for fields along $\hat{z}$.

5 Numerical Verification

5.1 Total Angular Momentum Squared

The identity $J^2 = j(j+1)\mathbb{1}$ can verify correctness:

$$J^2 = J_x^2 + J_y^2 + J_z^2 = j(j+1)\mathbb{1}_{(2j+1)\times(2j+1)} \tag{20}$$

5.2 Eigenvalue Check

J_z should have eigenvalues $\{j, j-1, \ldots, -j\}$ appearing on the diagonal in descending order.

6 Summary

The `ang_mom.py` module provides:

1. `jx(j)`: Returns $(2j + 1) \times (2j + 1)$ matrix for J_x
2. `jy(j)`: Returns $(2j + 1) \times (2j + 1)$ matrix for J_y
3. `jz(j)`: Returns $(2j + 1) \times (2j + 1)$ matrix for J_z

Key features:

- Uses ladder operator construction for numerical stability
- Works for any half-integer or integer j
- Matrices satisfy SU(2) algebra
- Called by `fs_hfs.py` and `sz_lsi.py` for building atomic Hamiltonians