fs_hfs.py

Fine and Hyperfine Structure Matrices

Energy Splitting in Alkali Atoms

ElecSus Library Documentation

Abstract

This document provides comprehensive documentation for `fs_hfs.py`, which calculates the fine structure ($\vec{L}\cdot\vec{S}$) and hyperfine structure ($\vec{I}\cdot\vec{J}$) interaction matrices. These matrices are essential for computing atomic energy levels and transition frequencies in alkali atoms.

Contents

1 Theoretical Foundation

1.1 Fine Structure Interaction

Axiom 1 (Spin-Orbit Coupling)**.** *The fine structure arises from the interaction between the electron's orbital angular momentum and its spin:*

$$H_{FS} = \xi(r)\vec{L} \cdot \vec{S} \tag{1}$$

where $\xi(r)$ is the spin-orbit coupling function.

Theorem 1 (Fine Structure Energy)**.** *Using $\vec{J} = \vec{L} + \vec{S}$:*

$$\vec{L} \cdot \vec{S} = \frac{1}{2}(J^2 - L^2 - S^2) = \frac{1}{2}[J(J+1) - L(L+1) - S(S+1)]\hbar^2 \tag{2}$$

Definition 1 (Total Electronic Angular Momentum)**.** *For alkali atoms with one valence electron:*

$$\vec{J} = \vec{L} + \vec{S} \tag{3}$$

For S-states ($L = 0$): $J = S = 1/2$. For P-states ($L = 1$): $J = 1/2$ or $3/2$.

1.2 Hyperfine Structure Interaction

Axiom 2 (Nuclear Magnetic Dipole Interaction)**.** *The magnetic dipole hyperfine interaction couples nuclear and electronic angular momenta:*

$$H^{(1)}_{HFS} = A_{HFS}\vec{I} \cdot \vec{J} \tag{4}$$

where A_{HFS} is the magnetic dipole hyperfine constant.

Theorem 2 (Hyperfine Energy Shift)**.** *Using $\vec{F} = \vec{I} + \vec{J}$:*

$$E^{(1)}_{HFS} = \frac{A_{HFS}}{2}[F(F+1) - I(I+1) - J(J+1)] \tag{5}$$

Definition 2 (Electric Quadrupole Interaction)**.** *For $I \geq 1$ and $J \geq 1$, the nuclear electric quadrupole interaction:*

$$H^{(2)}_{HFS} = B_{HFS}\frac{3(\vec{I} \cdot \vec{J})^2 + \frac{3}{2}(\vec{I} \cdot \vec{J}) - I(I+1)J(J+1)}{2I(2I-1)J(2J-1)} \tag{6}$$

2 Basis State Formalism

2.1 Uncoupled Basis

Definition 3 (Product Basis)**.** *The uncoupled basis states are:*

$$|L, m_L\rangle \otimes |S, m_S\rangle \otimes |I, m_I\rangle \tag{7}$$

with dimensions $(2L+1)(2S+1)(2I+1)$.

Theorem 3 (Kronecker Product Structure)**.** *Operators on the full Hilbert space are constructed via tensor products:*

$$J_x = L_x \otimes \mathbb{1}_S \otimes \mathbb{1}_I + \mathbb{1}_L \otimes S_x \otimes \mathbb{1}_I \tag{8}$$

3 Line-by-Line Code Analysis

3.1 Module Imports

```
from numpy import identity,dot
from scipy.linalg import kron
from ang_mom import jx,jy,jz
```

Import identity matrix, matrix multiplication, Kronecker product, and angular momentum matrices.

3.2 Fine Structure Function: Hfs(L,S,I)

```
def Hfs(L,S,I):
    """Provides the L dot S matrix (fine structure)"""
    gS=int(2*S+1) #number of mS values
    Sx=jx(S)
    Sy=jy(S)
    Sz=jz(S)
    Si=identity(gS)
```

Create spin matrices and identity in the $(2S+1)$-dimensional spin space.

```
    gL=int(2*L+1)
    Lx=jx(L)
    Ly=jy(L)
    Lz=jz(L)
    Li=identity(gL)
```

Create orbital angular momentum matrices in $(2L+1)$-dimensional space.

```
    gJ=gL*gS
    Jx=kron(Lx,Si)+kron(Li,Sx)
    Jy=kron(Ly,Si)+kron(Li,Sy)
    Jz=kron(Lz,Si)+kron(Li,Sz)
    J2=dot(Jx,Jx)+dot(Jy,Jy)+dot(Jz,Jz)
```

$$\vec{J} = \vec{L} \otimes \mathbb{1}_S + \mathbb{1}_L \otimes \vec{S} \tag{9}$$

$$J^2 = J_x^2 + J_y^2 + J_z^2 \tag{10}$$

Construct total electronic angular momentum in the product space.

```
    gI=int(2*I+1)
    Ii=identity(gI)
    gF=gJ*gI
    Fi=identity(gF)
    Hfs=0.5*(kron(J2,Ii)-L*(L+1)*Fi-S*(S+1)*Fi)
    return Hfs
```

$$H_{FS} = \frac{1}{2}(J^2 \otimes \mathbb{1}_I - L(L+1)\mathbb{1} - S(S+1)\mathbb{1}) \tag{11}$$

Compute $\vec{L}\cdot\vec{S}$ using the identity $\vec{L}\cdot\vec{S} = \frac{1}{2}(J^2 - L^2 - S^2)$, extended to include nuclear spin space.

3.3 Hyperfine Structure Function: Hhfs(L,S,I)

```
def Hhfs(L,S,I):
    """Provides the I dot J matrix (hyperfine structure interaction)"""
```

Calculate the magnetic dipole hyperfine matrix $\vec{I} \cdot \vec{J}$.

```
    # [Similar construction of L, S matrices...]

    Fx=kron(Jx,Ii)+kron(Ji,Ix)
    Fy=kron(Jy,Ii)+kron(Ji,Iy)
    Fz=kron(Jz,Ii)+kron(Ji,Iz)
    Fi=identity(gF)
    F2=dot(Fx,Fx)+dot(Fy,Fy)+dot(Fz,Fz)
    Hhfs=0.5*(F2-I*(I+1)*Fi-kron(J2,Ii))
    return Hhfs
```

$$\vec{F} = \vec{J} \otimes \mathbb{1}_I + \mathbb{1}_J \otimes \vec{I} \tag{12}$$

$$H_{HFS} = \frac{1}{2}(F^2 - I(I+1)\mathbb{1} - J^2 \otimes \mathbb{1}_I) = \vec{I} \cdot \vec{J} \tag{13}$$

Use the identity $\vec{I} \cdot \vec{J} = \frac{1}{2}(F^2 - I^2 - J^2)$.

3.4 Electric Quadrupole Function: Bbhfs(L,S,I)

```
def Bbhfs(L,S,I):
    """Calculates electric quadrupole matrix."""
    # [Angular momentum construction...]

    IdotJ=kron(Jx,Ix)+kron(Jy,Iy)+kron(Jz,Iz)
    IdotJ2=dot(IdotJ,IdotJ)
```

$$(\vec{I} \cdot \vec{J})^2 = (I_x J_x + I_y J_y + I_z J_z)^2 \tag{14}$$

Square of the dot product, needed for quadrupole interaction.

```
    if I != 0:
        Bbhfs=1./(6*I*(2*I-1))*(3*IdotJ2+3./2*IdotJ-I*(I+1)*15./4*Fi)
    else:
        Bbhfs = 0
    return Bbhfs
```

$$H_Q = \frac{1}{6I(2I-1)}\left[3(\vec{I} \cdot \vec{J})^2 + \frac{3}{2}(\vec{I} \cdot \vec{J}) - \frac{15}{4}I(I+1)\mathbb{1}\right] \tag{15}$$

Electric quadrupole term. Returns zero for $I = 0$ *(no nuclear quadrupole moment).*

4 Physical Examples

4.1 Rubidium-87 Ground State

For ^{87}Rb ground state ($5S_{1/2}$): $L = 0$, $S = 1/2$, $I = 3/2$, $J = 1/2$.

Hyperfine levels: $F = 1, 2$ with splitting:

$$\Delta E_{HFS} = A_{HFS}(F=2) - A_{HFS}(F=1) = A_{HFS} \cdot \frac{3}{2} \cdot 2 = 3A_{HFS} \tag{16}$$

4.2 Cesium D2 Line

For ^{133}Cs $6P_{3/2}$: $L = 1$, $S = 1/2$, $I = 7/2$, $J = 3/2$.

The quadrupole interaction ($B_{HFS} \neq 0$) causes additional level shifts.

5 Summary

The `fs_hfs.py` module provides three essential functions:

1. `Hfs(L,S,I)`: Fine structure matrix $\vec{L} \cdot \vec{S}$
2. `Hhfs(L,S,I)`: Magnetic dipole hyperfine matrix $\vec{I} \cdot \vec{J}$
3. `Bbhfs(L,S,I)`: Electric quadrupole hyperfine matrix

Key features:

- Works in the uncoupled $|m_L, m_S, m_I\rangle$ basis
- Uses Kronecker products for tensor space construction
- Called by `EigenSystem.py` to build the full atomic Hamiltonian
- Matrix dimensions: $(2L+1)(2S+1)(2I+1) \times (2L+1)(2S+1)(2I+1)$