Programming the Spin Hamiltonian

Deval Deliwala

The goal of this post is to program the spin Hamiltonian governing recombination in silicon carbide. Our system has two electrons and two nuclei.

We fix an ordered orthonormal basis. The electrons are in the coupled basis $|s, m\rangle$. The nuclei remain in the Zeeman basis $\left(|+\frac{1}{2}\rangle, |-\frac{1}{2}\rangle\right)$. Every state is written as

$$|s, m\rangle \; |m_{I1}\rangle \; |m_{I2}\rangle.$$

We will build $\hat{H}_Z$, $\hat{H}_{HF}$, $\hat{H}_{ZFS}$, and $\hat{H}_{EX}$ in this same ordering.

Zeeman

The Zeeman Hamiltonian in this basis is diagonal:

$$\hat{H}_Z \; |s, m\rangle \; | \; m_{I1}\rangle \; | \; m_{I2}\rangle = (m g_e \mu_B B_0 + \mu_N B_0 (m_{I1} g_{n1} + m_{I2} g_{n2}) \; |s, m\rangle \; |m_{I1}\rangle \; | \; m_{I2}\rangle.$$

Here g_e is the electron g-factor. g_{n1} and g_{n2} are the nuclear g-factors (silicon and carbon). μ_B and μ_N are the Bohr magneton and nuclear magneton.

Implementation

We encode the basis ordering explicitly. Electron states run slow. Nuclear states run fast.

```
import sympy as sp

# electron coupled basis |s, m>
electron_pairs = [
    (1, +1),
    (1,  0),
    (0,  0),
    (1, -1),
]

# nuclear Zeeman basis |mI1>|mI2>
nuclear_pairs = [
    (sp.Rational(+1, 2), sp.Rational(+1, 2)),
    (sp.Rational(+1, 2), sp.Rational(-1, 2)),
    (sp.Rational(-1, 2), sp.Rational(+1, 2)),
    (sp.Rational(-1, 2), sp.Rational(-1, 2)),
]
```

Then we apply the energy formula to every basis state in the same order.

```
g_e, g_n1, g_n2 = sp.symbols("g_e g_n1 g_n2")
mu_B, mu_N, B0  = sp.symbols("mu_B mu_N B0")

omega_e  = g_e  * mu_B * B0
omega_n1 = g_n1 * mu_N * B0
omega_n2 = g_n2 * mu_N * B0

energies = []
for s, m in electron_pairs:        # electrons run slow
    for mI1, mI2 in nuclear_pairs: # nuclei run fast
        energies.append(
            m   * omega_e +
            mI1 * omega_n1 +
            mI2 * omega_n2
        )

H_Z = sp.diag(*energies)
```

This nested-loop ordering is now fixed. Every other Hamiltonian must match it.

Hyperfine

The hyperfine Hamiltonian is simplest to construct in the Zeeman $(|\uparrow\rangle, |\downarrow\rangle)$ basis. We will build $\hat{H}_{HF}$ in the Zeeman basis for both electrons and both nuclei, then convert the electron subspace into the coupled basis using a Clebsch–Gordan transform.

Two electrons and two nuclei

For two electrons $k \in \{a, b\}$ and two nuclei $p \in \{1, 2\}$,

$$\hat{H}_{HF} = \sum_{k \in \{a,b\}} \sum_{p=1}^{2} \left(A_{kpx} S_{kx} I_{px} + A_{kpy} S_{ky} I_{py} + A_{kpz} S_{kz} I_{pz}\right).$$

We implement this using ladder-operator bookkeeping. In the Zeeman basis, the only actions we need are

$$S_+ \; |\downarrow\rangle = \hbar \; |\uparrow\rangle,$$
$$S_- \; |\uparrow\rangle = \hbar \; |\downarrow\rangle,$$
$$S_z \; |m\rangle = m\hbar \; |m\rangle,$$

and the same structure for I_+, I_-, I_z on nuclear states.

Following this bookkeeping, the action on a Zeeman-basis state $|m_a, m_b, m_{I1}, m_{I2}\rangle$ can be written compactly as:

$$\begin{aligned}
\hat{H}_{HF} \; |m_a, m_b, m_{I1}, m_{I2}\rangle = \hbar^2 \sum_{p=1}^{2} \left(A_{apz} m_a m_{Ip} + A_{bpz} m_b m_{Ip}\right) \; |m_a, m_b, m_{I1}, m_{I2}\rangle \\
+ \frac{\hbar^2}{4} \left((A_{a1x} - A_{a1y}) \delta_{m_a, m_{I1}} + (A_{a1x} + A_{a1y}) \delta_{m_a, -m_{I1}} \right) \; | - m_a, m_b, -m_{I1}, m_{I2}\rangle \\
+ \frac{\hbar^2}{4} \left((A_{a2x} - A_{a2y}) \delta_{m_a, m_{I2}} + (A_{a2x} + A_{a2y}) \delta_{m_a, -m_{I2}} \right) \; | - m_a, m_b, m_{I1}, -m_{I2}\rangle \\
+ \frac{\hbar^2}{4} \left((A_{b1x} - A_{b1y}) \delta_{m_b, m_{I1}} + (A_{b1x} + A_{b1y}) \delta_{m_b, -m_{I1}} \right) \; |m_a, -m_b, -m_{I1}, m_{I2}\rangle \\
+ \frac{\hbar^2}{4} \left((A_{b2x} - A_{b2y}) \delta_{m_b, m_{I2}} + (A_{b2x} + A_{b2y}) \delta_{m_b, -m_{I2}} \right) \; |m_a, -m_b, m_{I1}, -m_{I2}\rangle.
\end{aligned}$$

This builds $\hat{H}_{HF}$ in the Zeeman basis for the electrons. We now convert the electron subspace to the coupled basis.

Clebsch–Gordan transform

The mapping between electron Zeeman states and electron coupled states is

$$\begin{pmatrix} |\uparrow\uparrow\rangle \\ |\uparrow\downarrow\rangle \\ |\downarrow\uparrow\rangle \\ |\downarrow\downarrow\rangle \end{pmatrix} = U \begin{pmatrix} |1 \quad 1\rangle \\ |1 \quad 0\rangle \\ |0 \quad 0\rangle \\ |1 \quad -1\rangle \end{pmatrix},$$

with

$$U = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

We lift this to the full Hilbert space by leaving nuclei unchanged:

$$W = U \otimes I_4.$$

If $H_{HF,\text{ Zeeman}}$ is built in the ordered Zeeman basis, then the coupled basis matrix is

$$H_{HF,\text{ coupled}} = W H_{HF,\text{ Zeeman}} W^{\dagger}.$$

Implementation

We generate the 16 Zeeman-basis kets in a fixed order.

```
import sympy as sp

def generate_zeeman_basis():
    half = sp.Rational(1, 2)
    return [
        (m_a, m_b, m_I1, m_I2)
        for m_a in (half, -half)      # electrons run slow
        for m_b in (half, -half)
        for m_I1 in (half, -half)     # nuclei run fast
        for m_I2 in (half, -half)
    ]
```

We implement the delta logic and build the sparse action map `{new_ket: coeff}`. In the code below we set `planck = 1` and restore factors later.

```
def delta_parallel(m_e, m_I):
    return m_e == m_I

def delta_antiparallel(m_e, m_I):
    return m_e == -m_I

class HyperfineBuilder:
    def __init__(self):
        self.A = {}
        for e in ("a", "b"):
            for n in ("1", "2"):
                for comp in ("x", "y", "z"):
                    key = f"A{e}{n}{comp}".upper()
                    self.A[key] = sp.symbols(key)

    def action_on_ket(self, ket):
        m_a, m_b, m_I1, m_I2 = ket
        psi = {}

        # diagonal Sz Iz terms
        diag = (
            self.A["AA1Z"] * m_a * m_I1 +
            self.A["AB1Z"] * m_b * m_I1 +
            self.A["AA2Z"] * m_a * m_I2 +
            self.A["AB2Z"] * m_b * m_I2
        )
        psi[ket] = diag

        def add_flip(m_e, m_I, x_key, y_key, new_ket):
            if delta_parallel(m_e, m_I):
                psi[new_ket] = (self.A[x_key] - self.A[y_key]) / 4
            elif delta_antiparallel(m_e, m_I):
                psi[new_ket] = (self.A[x_key] + self.A[y_key]) / 4

        # off-diagonal flip-flops
        add_flip(m_a, m_I1, "AA1X", "AA1Y", (-m_a,  m_b, -m_I1,  m_I2))
        add_flip(m_a, m_I2, "AA2X", "AA2Y", (-m_a,  m_b,  m_I1, -m_I2))
        add_flip(m_b, m_I1, "AB1X", "AB1Y", ( m_a, -m_b, -m_I1,  m_I2))
        add_flip(m_b, m_I2, "AB2X", "AB2Y", ( m_a, -m_b,  m_I1, -m_I2))

        return psi
```

We build the Zeeman-basis matrix by iterating through basis states and inserting coefficients.

```
def build_hyperfine_zeeman_matrix():
    builder = HyperfineBuilder()

    basis = generate_zeeman_basis()
    index = {ket: i for i, ket in enumerate(basis)}

    n = len(basis)
    H = sp.MutableDenseMatrix(n, n, lambda *_: 0)

    for j, ket in enumerate(basis):
        action = builder.action_on_ket(ket)
        for new_ket, coeff in action.items():
            i = index[new_ket]
            H[i, j] = coeff

    return H.as_immutable()
```

Now we build $W = U \otimes I_4$ and transform into the coupled basis.

```
def build_cg_unitary():
    half = sp.sqrt(sp.Rational(1, 2))
    U = sp.Matrix([
        [1,    0,     0,    0],
        [0,  half,  half,   0],
        [0,  half, -half,   0],
        [0,    0,     0,    1],
    ])
    I4 = sp.eye(4)
    return sp.kronecker_product(U, I4)

H_HF_ze = build_hyperfine_zeeman_matrix()
W = build_cg_unitary()

H_HF = sp.simplify(W * H_HF_ze * W.H)
```

Zero-Field Splitting

From the previous derivation, the action on a general $|s, m\rangle$ is

$$\begin{aligned}
\hat{H}_{ZFS} \; |s, m\rangle = (Dm^2) \; |s, m\rangle - \frac{D}{3}(s(s+1)) \; |s, m\rangle \\
+ \frac{E}{2}(s(s+1) - m(m+1)) \; |s, m+2\rangle \\
+ \frac{E}{2}(s(s+1) - m(m-1)) \; |s, m-2\rangle.
\end{aligned}$$

This term acts only on the electron subspace, so we lift it with a tensor product by I_4 on the nuclei.

Implementation

```
import sympy as sp

electron_pairs = [
    (1, +1),
    (1,  0),
    (0,  0),
    (1, -1),
]

D, E = sp.symbols("D E")

# 4x4 electron ZFS matrix (coupled basis)
H_elec = sp.zeros(4)

for i, (s, m) in enumerate(electron_pairs):
    H_elec[i, i] = D * m**2 - (D / 3) * s * (s + 1)

    for dm, expr in [
        ( 2, (E / 2) * (s * (s + 1) - m * (m + 1))),
        (-2, (E / 2) * (s * (s + 1) - m * (m - 1))),
    ]:
        m2 = m + dm
        if (s, m2) in electron_pairs:
            j = electron_pairs.index((s, m2))
            H_elec[i, j] = expr
            H_elec[j, i] = expr

I_nuc = sp.eye(4)
H_ZFS = sp.kronecker_product(H_elec, I_nuc)
```

Exchange Interaction

From the previous derivation,

$$\hat{H}_{EX} = -J\left(\hat{S}_a \cdot \hat{S}_b\right).$$

Using $\hat{S} = \hat{S}_a + \hat{S}_b$,

$$\hat{S}_a \cdot \hat{S}_b = \frac{1}{2}\left(\hat{S}^2 - \hat{S}_a^2 - \hat{S}_b^2\right).$$

For spin-1/2 electrons in the coupled basis,

$$\hat{S}^2 \; |s, m\rangle = \hbar^2 s(s+1) \; |s, m\rangle,$$
$$\hat{S}_a^2 \; |s, m\rangle = \frac{3\hbar^2}{4} \; |s, m\rangle,$$
$$\hat{S}_b^2 \; |s, m\rangle = \frac{3\hbar^2}{4} \; |s, m\rangle.$$

Therefore,

$$\hat{H}_{EX} \; |s, m\rangle = -J\frac{\hbar^2}{2}\left(s(s+1) - \frac{3}{2}\right) \; |s, m\rangle.$$

This is diagonal in the coupled basis and lifts by $\otimes I_4$ on nuclei.

Implementation

```
import sympy as sp

J = sp.symbols("J")

H_ex_elec = sp.zeros(4)

for i, (s, m) in enumerate(electron_pairs):
    H_ex_elec[i, i] = -J * (s * (s + 1) - sp.Rational(3, 2)) / 2

I_nuc = sp.eye(4)
H_EX = sp.kronecker_product(H_ex_elec, I_nuc)
```

Full spin Hamiltonian

Once all four sub-Hamiltonians are constructed in the same ordered basis, the full Hamiltonian is

```
H_SPIN = H_Z + H_HF + H_ZFS + H_EX
```

Basis ordering

Indices 0–15 refer to the row/column ordering of $\mathscr{H}$.

$$\begin{aligned}
&0 \quad |1, 1\rangle \quad \otimes |+\tfrac{1}{2}, +\tfrac{1}{2}\rangle \\
&1 \quad |1, 1\rangle \quad \otimes |+\tfrac{1}{2}, -\tfrac{1}{2}\rangle \\
&2 \quad |1, 1\rangle \quad \otimes |-\tfrac{1}{2}, +\tfrac{1}{2}\rangle \\
&3 \quad |1, 1\rangle \quad \otimes |-\tfrac{1}{2}, -\tfrac{1}{2}\rangle \\
&4 \quad |1, 0\rangle \quad \otimes |+\tfrac{1}{2}, +\tfrac{1}{2}\rangle \\
&5 \quad |1, 0\rangle \quad \otimes |+\tfrac{1}{2}, -\tfrac{1}{2}\rangle \\
&6 \quad |1, 0\rangle \quad \otimes |-\tfrac{1}{2}, +\tfrac{1}{2}\rangle \\
&7 \quad |1, 0\rangle \quad \otimes |-\tfrac{1}{2}, -\tfrac{1}{2}\rangle \\
&8 \quad |0, 0\rangle \quad \otimes |+\tfrac{1}{2}, +\tfrac{1}{2}\rangle \\
&9 \quad |0, 0\rangle \quad \otimes |+\tfrac{1}{2}, -\tfrac{1}{2}\rangle \\
&10 \quad |0, 0\rangle \quad \otimes |-\tfrac{1}{2}, +\tfrac{1}{2}\rangle \\
&11 \quad |0, 0\rangle \quad \otimes |-\tfrac{1}{2}, -\tfrac{1}{2}\rangle \\
&12 \quad |1, -1\rangle \otimes |+\tfrac{1}{2}, +\tfrac{1}{2}\rangle \\
&13 \quad |1, -1\rangle \otimes |+\tfrac{1}{2}, -\tfrac{1}{2}\rangle \\
&14 \quad |1, -1\rangle \otimes |-\tfrac{1}{2}, +\tfrac{1}{2}\rangle \\
&15 \quad |1, -1\rangle \otimes |-\tfrac{1}{2}, -\tfrac{1}{2}\rangle
\end{aligned}$$

(For nuclei only: $|+\frac{1}{2}\rangle = |\uparrow\rangle$ and $|-\frac{1}{2}\rangle = |\downarrow\rangle$.)

Next step

At this point $\mathscr{H}$ contains unknown constants:

g_e	electron g-factor
g_{n1}, g_{n2}	nuclear g-factors (Si, C)
μ_B	Bohr magneton
μ_N	nuclear magneton
B_0	external magnetic field
$A_{kpx}, A_{kpy}, A_{kpz}$	hyperfine tensor components
D	axial zero-field
E	transverse zero-field
J	exchange coupling.

The next step is eigenenergy simulation. We substitute values for these constants and sweep over B_0 in $\hat{H}_Z$. At each field point we compute the eigenvalues of $\mathscr{H}$, producing B_0 vs. energy curves for all 16 states. We will use LAPACK solves.