Coupling of Orbital and Spin Angular Momentum

The coupling of orbital and spin angular momentum ($\{L, S\}$ for atoms and $\{ L, S\}$ for molecules is often called Russel-Saunders coupling or *LS* coupling. The distribution of electrons in the available orbitals leads to one or more electronic configurations and, for each electronic configuration, one or more microstates. The set of microstates can be reduced to a set of electronic states with term symbols representative of the symmetry, multiplicity, and degeneracy of the underlying microstates. In some cases, the parity and orbital order (due to other interactions) are also determinable.

LS coupling for atoms is introduced at the third or forth year physical chemistry level. *LS* coupling for diatomics is reserved for graduate courses or is left to the student to learn, which is perfectly acceptable. General information on *LS* coupling can be found in, for example, Dykstra and Herzberg.

Once mastered, the process of determining the possible electronic states is repetitive and time consuming. Errors arise from the repetitive nature of the task. Consideration of multiple electronic configurations multiplies these issues. Consider the situation with three electrons in a π and σ orbital. The orbitals and electrons involved are called the *active space.*

electronic configuration	***possible electronic states***
$p^3 p^0 s^0$, $p^0 p^3 s^0$	$^2\Pi_i$
$p^2 p^1 s^0$, $p^1 p^2 s^0$	$^2\Phi_r$, $^4\Pi_r$, $^2\Pi_r$, $^2\Pi_i(2)$
$p^2 p^0 s^1$, $p^0 p^2 s^1$	$^2\Delta$,$^4\Sigma^-$, $^2\Sigma^+$, $^2\Sigma^-$
$p^1 p^0 s^2$, $p^0 p^1 s^2$	$^2\Pi_r$
$p^1 p^1 s^1$	$^4\Delta_r$, $^2\Delta(2)$, $^4\Sigma^+$, $^4\Sigma^-$, $^2\Sigma^+(2)$, $^2\Sigma^-(2)$

This process is readily automated.

In practice, only partially filled valence orbitals dictate the possible electronic states. Filled and empty orbitals lead to a $^1\Sigma^+$ configuration. Inclusion of [initially] filled and empty orbitals and allowing thier occupancy to vary allows for the calculation of the electronic states for all electronic configurations. For example, transition metal complexes have many low-energy orbitals; thier inclusion is important when considering the possible electronic transition within the molecules. This is not done in hand calculations because of the workload.

This program determines the possible electronic states for a user-defined set of orbitals and number of electrons. The procedure adopted is as follows:

① determine all possible single-electron quantum states
determine all allowed microstates from quantum states (combinatorial function)
organize the microstates based on electronic configuration and molecular quantum numbers: L and S
determine the electronic states (term symbols) for each electronic configuration

Data Input

data := (1 1 0 3)

In the form: $(\lambda_1\ \lambda_2\ \ldots.\ \lambda_n\ n_e)$

① *Single-Electon Quantum States*

orbs := cols(data) − 1 orbs = 3

Number of orbitals in active space

$n_e := data_{1,cols(data)}$ $n_e = 3$

Number of electrons in active space

Defines all possible quantum states

```
states := | x ← 1
          | for i ∈ 1..orbs                                  rates through the orbitals
          |   λ_i ← data_{1,i}
          |   if λ_i = 0                                     tine for σ state
          |     mλ ← 0                                       ines m_l
          |     for m_s ∈ −1/2 .. 1/2                        rates through the possible m_s states
          |       temp_x ← (i, λ_i, mλ, m_s)
          |       x ← x + 1
          |   otherwise                                      tine for non-σ states
          |     for mλ ∈ −λ_i, λ_i                           rates through the possible m_l states
          |       for m_s ∈ −1/2 .. 1/2                      rates through the possible m_s states
          |         temp_x ← (i, λ_i, mλ, m_s)
          |         x ← x + 1
          |     1
          | temp
```

$num_{states} := rows(states)$ $num_{states} = 10$

$num_{microstates} := combin(num_{states}, n_e)$ $num_{microstates} = 120$

Structure of output: ("original orbital", "angular momentum", "angular projection", "spin projection")

Allowed Microstates

```
states := | temp ← states
          | for i ∈ rows(states) + 1 .. 2 · rows(states)
          |   temp_i ← 0
          | temp
```

Augments the states matrix because the microstates algorithm is terrible — any ideas on optimizing it?

```
microstates := | x ← 1
               | for i ∈ 1 .. | num_states − n_e + 1   if n_e ≥ 1
               |              | 1   otherwise
               |   for j ∈ i + 1 .. | num_states − n_e + 2   if n_e ≥ 2
               |                    | i + 1   otherwise
               |     for k ∈ j + 1 .. | num_states − n_e + 3   if n_e ≥ 3
               |                      | j + 1   otherwise
               |       for l ∈ k + 1 .. | num_states − n_e + 4   if n_e ≥ 4
               |                        | k + 1   otherwise
               |         for m ∈ l + 1 .. | num_states − n_e + 5   if n_e ≥ 5
               |                          | l + 1   otherwise
               |           for n ∈ m + 1 .. | num_states − n_e + 6   if n_e ≥ 6
               |                            | m + 1   otherwise
               |             for o ∈ n + 1 .. | num_states − n_e + 7   if n_e ≥ 7
               |                              | n + 1   otherwise
               |               for p ∈ o + 1 .. | num_states − n_e + 8   if n_e ≥ 8
               |                                | o + 1   otherwise
               |                 temp_x,1 ← states_i
               |                 temp_x,2 ← states_j
               |                 temp_x,3 ← states_k
               |                 temp_x,4 ← states_l
               |                 temp_x,5 ← states_m
               |                 temp_x,6 ← states_n
               |                 temp_x,7 ← states_o
               |                 temp_x,8 ← states_p
               |                 x ← x + 1
               | temp ← submatrix(temp, 1, rows(temp), 1, n_e)
```

Defines all possible combinations of microstates

Organized Microstates

$elec_{config} :=$

$x \leftarrow 1$

for $i \in 1 .. num_{microstates}$

$m_\Lambda \leftarrow \sum_{j=1}^{n_e} \left(microstates_{i,j}\right)_3$ determines the total angular projection

continue if $m_\Lambda < 0$

$m_S \leftarrow \sum_{j=1}^{n_e} \left(microstates_{i,j}\right)_4$ determines the total spin projection

continue if $m_S < 0$

$temp_{x,1} \leftarrow$ for $j \in 1 .. orbs$ determines the occupancy of each orbital

$occupancy_j \leftarrow 0$

for $j \in 1 .. n_e$

$occupancy_{\left(microstates_{i,j}\right)_1} \leftarrow occupancy_{\left(microstates_{i,j}\right)_1} + 1$

occupancy

$temp_{x,2} \leftarrow$ $temp2 \leftarrow 0$ generates a sorting parameter

for $j \in 1 .. orbs$

$temp2 \leftarrow temp2 + 10^j \cdot occupancy_j$

$temp_{x,3} \leftarrow m_\Lambda$

$temp_{x,4} \leftarrow m_S$

$x \leftarrow x + 1$

$temp2 \leftarrow csort(temp, 2)$

$temp2_{rows(temp2)+1,2} \leftarrow 0$

$x \leftarrow 1$

$y \leftarrow 1$

$y_i \leftarrow 1$

for $i \in 1 .. rows(temp2) - 1$

$y \leftarrow y + 1$ if $temp2_{y,2} = temp2_{y+1,2}$

otherwise

$temp3_{x,1} \leftarrow temp2_{y,1}$

$temp3_{x,2} \leftarrow submatrix\left(temp2, y_i, y, 3, 4\right)$

$x \leftarrow x + 1$

$y \leftarrow y + 1$

$y_i \leftarrow y$

$temp3$

Extracts the minimal microstates necessary to resolve state: $m_L \geq 0$; $m_S \geq 0$.

Sorts the output via electronic configuration.

Structure of output: ("orbital occupancy" "LS values")

The output can be used to construct the typical *L,S* diagram.

Term Symbols

```
parser(data, start) := | temp ← submatrix(data, 1, start, 1, 3)
                       | x ← start
                       | while data_{x,2} > 0.5
                       |     if data_{x+1,2} = data_{x,2}
                       |         x ← x + 1
                       |         temp ← stack(temp, submatrix(data, x, x, 1, 3))
                       |     x ← x + 1   otherwise
                       | temp ← stack(temp, submatrix(data, x + 1, rows(data), 1, 3))
```

Subroutine to remove redundant LS values (similar to the cancelation procedure in hand-calculations)

```
Term_symbols := | for i ∈ 1 .. rows(elec_config)
                |     temp_{i,1} ← ""
                |     temp_{i,2} ← ""
                |     for j ∈ 1 .. orbs
                |         temp_{i,1} ← concat[ temp_{i,1}, Symbol_orbital(data_{1,j+1}), "(", num2str[ (elec_config_{i,1})_j ], ")" ]
                |     temp2 ← elec_config_{i,2}
                |     for j ∈ 1 .. rows(temp2)
                |         temp2_{j,3} ← 10 · temp2_{j,1} + temp2_{j,2}
                |     temp2 ← reverse(csort(temp2, 3))
                |     k ← 1
                |     while k < rows(temp2)
                |         temp2 ← parser(temp2, k)   if temp2_{k,2} > 0
                |         k ← k + 1
                |     temp2 ← submatrix(temp2, 1, rows(temp2), 1, 2)
                |     for j ∈ 1 .. rows(temp2)
                |         spec ← ""
                |         temp_{i,2} ← concat(temp_{i,2}, num2str(2 · temp2_{j,2} + 1), Symbol_Orbital_{temp2_{j,1}+1}, spec, ", ")
                |     temp_{i,2} ← substr(temp_{i,2}, 0, strlen(temp_{i,2}) − 2)
                | temp
```

Determines the Term Symbols for each electronic configuration.

prepares the electronic configuration

a sorting parameter to do a multicolumn sort: *L* then *S*

removes redundant LS values

prepares the Term Symbol

Still to come: assignment of $\Sigma^{\pm}$ states and to identify regular and inverted degenerate levels.

$$\mathrm{Term}_{symbols} = \begin{pmatrix} \text{"pi(3)pi(0)sigma(0)"} & \text{"2-Pi"} \\ \text{"pi(2)pi(1)sigma(0)"} & \text{"2-Phi, 4-Pi, 2-Pi, 2-Pi, 2-Pi"} \\ \text{"pi(1)pi(2)sigma(0)"} & \text{"2-Phi, 4-Pi, 2-Pi, 2-Pi, 2-Pi"} \\ \text{"pi(0)pi(3)sigma(0)"} & \text{"2-Pi"} \\ \text{"pi(2)pi(0)sigma(1)"} & \text{"2-Delta, 4-Sigma, 2-Sigma, 2-Sigma"} \\ \text{"pi(1)pi(1)sigma(1)"} & \text{"4-Delta, 2-Delta, 2-Delta, 4-Sigma, 4-Sigma, 2-Sigma, 2-Sigma, 2-Sigma, 2-Si} \\ \text{"pi(0)pi(2)sigma(1)"} & \text{"2-Delta, 4-Sigma, 2-Sigma, 2-Sigma"} \\ \text{"pi(1)pi(0)sigma(2)"} & \text{"2-Pi"} \\ \text{"pi(0)pi(1)sigma(2)"} & \text{"2-Pi"} \end{pmatrix}$$