

^{163}Lu - Excitation energies

New approach for TSD4 structure

Project which aims at computing the excitation energies for the wobbling spectrum of ^{163}Lu using a novel approach for the TSD4 band.

The nucleus ^{163}Lu has four wobbling bands (including the yrast TSD1) and the excitation spectrum has a rich structure. Each band (denoted by TSD1,2,3, and 4, respectively) has a certain number of *spin states*, with a definite energy for each state.

So far, the calculations involving *wobbling spectrum* of this nucleus were made with the fourth band considered as the coupling of an even-even core with an odd- j particle (the $h = \pi_{11/2}$ intruder proton). However, this current approach will consider the fourth band (*three-phonon wobbling band*) as a collection of states with definite energies, but the coupling an odd- j particle will be detached (no more coupling: **the spins in this band will be equivalent to the spin of the core itself**).

Main goals of the present project are:

- ☐ Obtain the analytic formulas for the four excited **wobbling bands** in ^{163}Lu .
- ☐ Write the experimental data set with the known spins and excitation energies for each band.
- ☐ **New formalism:** The spins for the fourth band (namely, TSD4) will represent the values of the *core*, and not the total particle-rotor model (no core+odd-particle coupling).
- ☐ Define the free parameter set which enters in the energy formulas.
- ☐ Compute the RMS of the excitation energies of all the bands w.r.t. to a parameter set P .
- ☐ Find the parameter set P_m which provides the best RMS value.

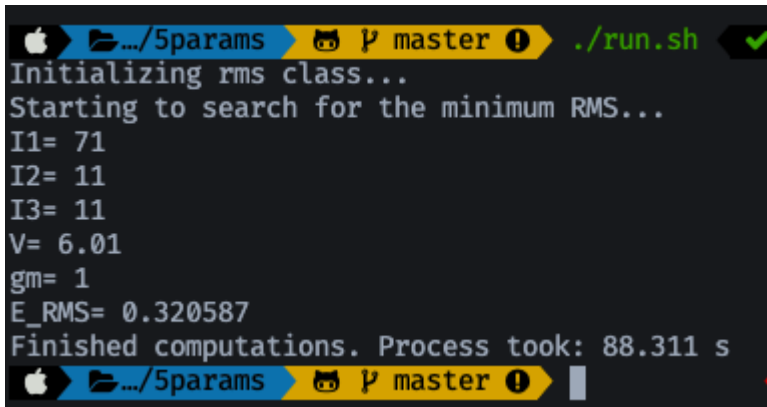
Initial test run

August 2020:

```
Initializing rms class...
Starting to search for the minimum RMS...
I1= 71
I2= 11
I3= 11
V= 1.01
gm= 6
E_RMS= 0.350248
Finished computations. Process took: 90.306 s
```

The algorithm is using large parameter steps, so accuracy is lost.

Update on the algorithm with optimized computations.

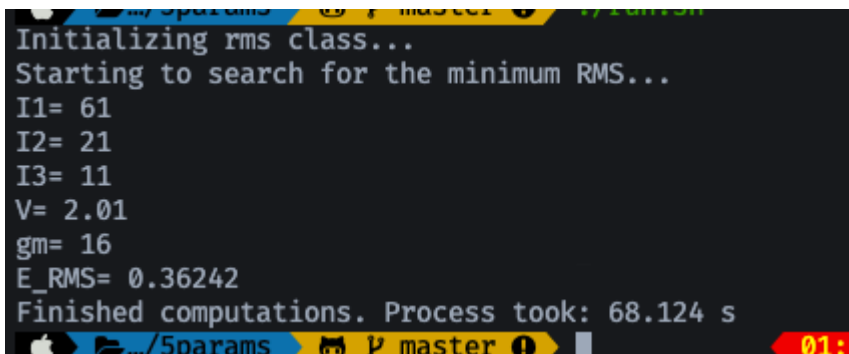


```

Apple > .../5params > P master ! > ./run.sh ✓
Initializing rms class...
Starting to search for the minimum RMS...
I1= 71
I2= 11
I3= 11
V= 6.01
gm= 1
E_RMS= 0.320587
Finished computations. Process took: 88.311 s
Apple > .../5params > P master ! >

```

Simulation implying the triaxiality conditions.

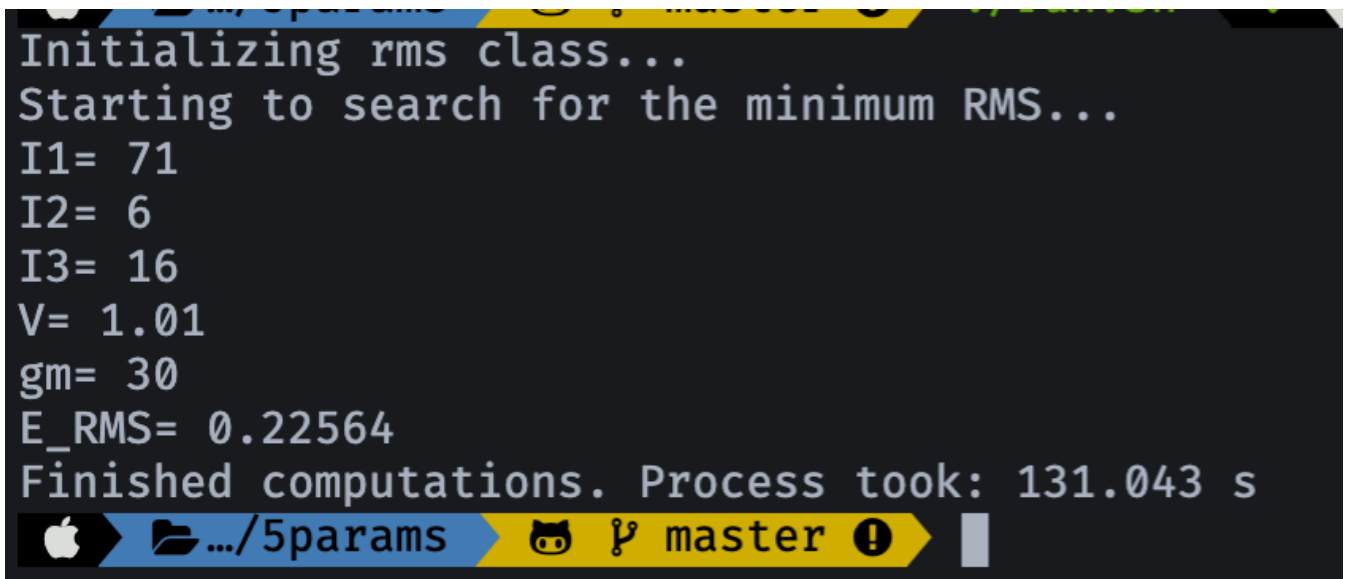


```

Apple > .../5params > P master ! > ./run.sh ✓
Initializing rms class...
Starting to search for the minimum RMS...
I1= 61
I2= 21
I3= 11
V= 2.01
gm= 16
E_RMS= 0.36242
Finished computations. Process took: 68.124 s
Apple > .../5params > P master ! > 01:

```

Optimized search with steps of size 5, 5, 0.5, respectively.



```

Apple > .../5params > P master ! > ./run.sh ✓
Initializing rms class...
Starting to search for the minimum RMS...
I1= 71
I2= 6
I3= 16
V= 1.01
gm= 30
E_RMS= 0.22564
Finished computations. Process took: 131.043 s
Apple > .../5params > P master ! >

```

Results in params.dat

```

Starting to search for the minimum RMS...
I1= 71
I2= 6
I3= 16
V= 1.01
gm= 30
E_RMS= 0.22564
0.241195,0.539159,0.89377,1.30494,1.77262,2.29675,2.87732,3.51429,4.20766,4.95
Finished computations after 777 valid parameter evaluations...
Total evaluations: 1861111

```

Computation ran on VM@elk

This calculations were done using the following sizes for the steps: $sl=2.5$ $gms=1$ $vs=0.25$

```
Initializing rms class...
Starting to search for the minimum RMS...
I1= 66
I2= 71
I3= 3.5
V= 9.76
gm= 28
E_RMS= 0.238275
Finished computations. Process took: 3315.19 s
```

Compilation using triaxiality and transverse regime.

The step size for $l=5$

```
Initializing rms class...
Starting to search for the minimum RMS...
I1= 56
I2= 61
I3= 6
V= 9.76
gm= 50
E_RMS= 0.475362
Finished computations. Process took: 498.02 s
```

Compilation using triaxiality and transverse regime.

The step size for $l=2.5$

```
Initializing rms class...
Starting to search for the minimum RMS...
I1= 66
I2= 71
I3= 3.5
V= 9.76
gm= 28
E_RMS= 0.238275
Finished computations. Process took: 4546.12 s
```

Compilation using fixed size array and small step

```
Parameter set determination using the fixed size arrays, with no memory re-all
I1= 76
I2= 77.5
I3= 2.5
V= 3.01
gm= 59
E_RMS= 0.2066
Finished computations. Process took: 16662.5 s
```

Using the VM@elk

I_step=1

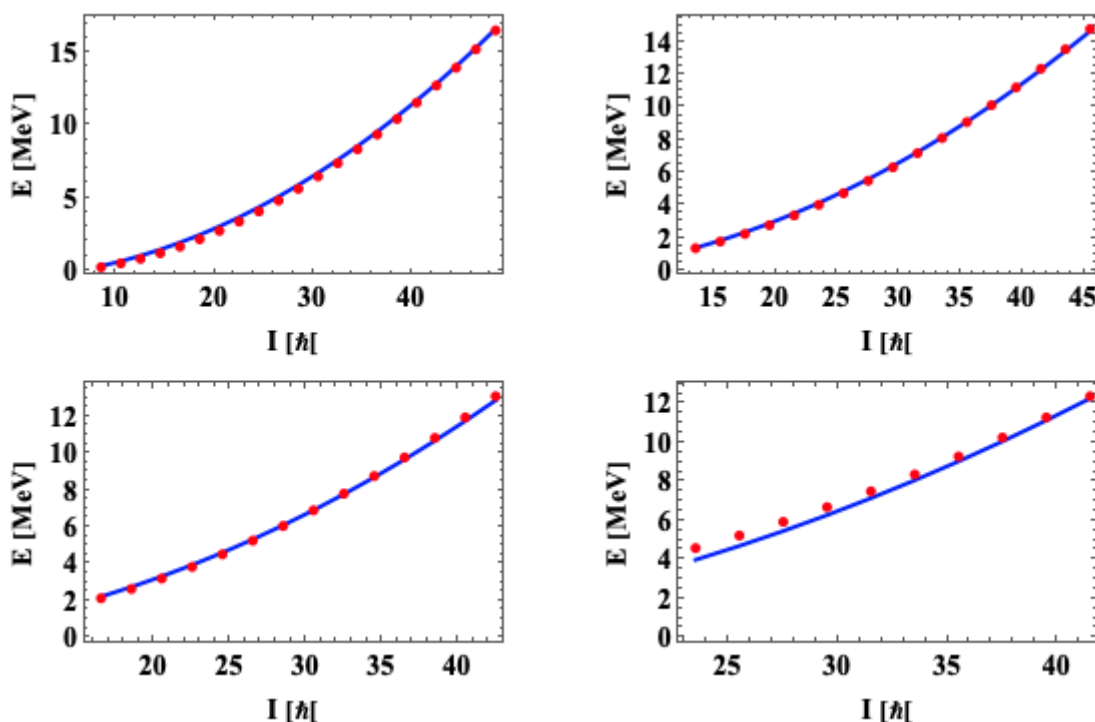
```
Initializing rms class...
Starting to search for the minimum RMS...
```

```
I1= 71
I2= 72
I3= 3
V= 5.01
gm= 19
E_RMS= 0.209134
```

Finished computations. Process took: 51392.6 s

Preliminary results

(small step - fixed array computation)



Using a fixed (δ) least-difference between the three moments of inertia

This method only chooses the moments of inertia that have a relative absolute difference between each other, bigger than a fixed quantity (e.g. `double MOI_AcceptedDifference = 5;` in code).

```
Searching for the minimal RMS value using the fixed array procedure.
Initializing rms class...
Starting to search for the minimum RMS...
Parameter set determination using the fixed size arrays, with no memory re-all
I1= 83
I2= 93
I3= 2
V= 9.1
```

```
gm= 17  
E_RMS= 0.325772  
Finished computations. Process took: 13.866 s
```