Luke Ali

Project Report: Hyperplane Index Finder

Description and Objective: To determine the N-1 dimensional hyperplane of an N dimensional grid. This problem comes from a note on the website for the workshop:

[[1]](#footnote-1)

**Motivation and Background:**

* Why is this project important:
  + Can be used to calculate the flux through the hyperplane.
* Method:
  + Deriving a mathematical relationship between point numbers and using that to write code that finds all the points through recursion.

Libra uses methods described [here](https://github.com/compchem-cybertraining/Tutorials_Libra/blob/master/6_dynamics/4_wavepackets/2_dvr_basics/tutorial.ipynb) to define an N dimensional grid. The problem asks us to “write a function that would return a subset of points with a fixed value of one projection.” I solved this by deriving the relevant mathematical relationship with the assumption that you’re only given the size of the grid, the projection axis, and the point along that axis.

Solution:   
Key Things:

* The grid index starts at 0 and follows the following pattern:  
  Index = 0 , Point = (0, 0, 0, … , 0)   
  Index = 1, Point = (0, 0, 0, … , 1)   
  Index = 2, Point = (0, 0, 0, … , 2)
* The max value on any axis depends on the size of the grid. For example:   
  A 3-Dimensional grid of size [4, 5, 7] will have a last or maximum index point of (3, 4, 6).   
  More generally, a 3-Dimensional grid of size will have a max index point of .

How I came up with the solution:

1. Using the key properties about the grids, I began deriving the mathematical equations.
2. First I began with the 3 dimensional case and then followed with the 4 dimensional case, immediately, the pattern became apparent when I rearranged the terms.
3. For the 4 Dimensional case, the expression is: Based on the slice that you’re trying to find, the coefficients are replaced by the value of that axes. For example, in the 4-Dimensional case, if we wanted to find the slice   
   (x, 5, x, x) then the equation would become: . In some sense, you can view the slices as a coefficient row vector , in which case, the example above would be .

Graphical illustrations:

3D Case

Grid =

Slice =

Chart

Description automatically generated

The blue plane is the hyperplane to this grid.

2D Case:  
Grid = [9, 8]   
Slice =

A picture containing chart

Description automatically generated

The 1D case is trivial and is the single point of the projection.

Other things I completed during the workshop:

Tutorials:

* Libra – NBRA Tutorial using
* OpenMolcas Tutorial
* COBRAMM Tutorial
* SHARC Tutorial

Tests I ran for other systems:

* NBRA for Lead Titanate – Step 1 & 2
* SHARC for Pyrrole (Results were not very promising, maybe too small active space)
* Optimization + Frequency for Pyrrole using OpenMolcas.

Where are my input/output files?:   
I believe they’re still on the UB servers. Unfortunately, in my infinite wisdom, I didn’t keep a local copy and have since lost access to the servers. EDIT: I found two files from the Lead Titanate calculations I ran in the NBRA tutorial for CP2K.

The python script containing the solution to the grid problem is included in the same directory as this file.

Personal Review:

Overall, the workshop was very detailed, though a bit disorganized at times; mostly due to technical issues. My personal progress was limited by my unfamiliarity with the content covered. My time at the workshop was not wasted however, as I was able to meet and interact with many people, and learned quite a number of new things. Most importantly, I was exposed to new concepts and theories. In the following months, I plan to read and indulge in the relevant literature to acquire the knowledge needed to truly understand the things discussed. Thank you for this experience and I hope I can do it again, next time with a developed and honed understanding of the underlying concepts.

1. https://compchem-cybertraining.github.io/Cyber\_Training\_Workshop\_2022/\_episodes/03-libra [↑](#footnote-ref-1)