Solution - Session 3 Report

Anirudh Agarwal

November 4, 2023

Abstract

First things first, the files to be uploaded to the google drive for submission are:

1. Overleaf source file in a separate folder
2. PDF report generated with overleaf
3. Mendeley references pdfs/ docs/ datasheets/ other in a separate references folder
4. Python codes, inputs, and outputs, in a separate folder

1 Objectives for this Session

The objectives of this session are:

1. Complete the code and search for session 2 based on the solutions provided.
2. Find Moment of Inertia (MOI) values for satellites of 1U to 16U.
3. Find typical values of mass for satellites of 1U to 16U.
4. Find coatings used for radiation protection in space based on session 2 solution.
5. Find suppliers for coating materials and their contact information.
6. Run the code from session 2 for each size from 1U to 16U for cells that are smaller than the smallest dimension.
7. Get lenopen, widopen, and mtot values for all to calculate MOI based on given equation.
8. Plot MOI vs size in U for each size for the different configurations.
9. Plot ptot/vtot and ptot/mtot vs size in U for different configurations.

2 Search

List of things searched:

- Mass and Moment of inertia values for crafts 1U to 16U.
- Coatings used for radiation protection and their suppliers

2.1 Mass and MOI for CubeSats

In the first paragraph of the Wikipedia page on CubeSats there is a link to a draft paper describing the mass and size of CubeSats [1]. From that, we get a mass for the size of 2kg per U such that the mass is 2 times the U. That is what we use in the excel sheet to make the craft sizes text file. The sizes of CubeSats were already covered previously.

For MOI, I found just 2 references, one Research Gate paper image showing MOI for 1U and one thesis project [2] for the MOI of a 3U craft. If you have found MOIs for more craft sizes from different sources, please make a table for MOI vs size. From my sources, I have MOI(1U) [0.006, 0.0013, 0.006] kg m^2 and MOI(3U) [0.037, 0.051, 0.021] kg m^2 for [Ix, Iy, Iz].

2.2 Coatings used for radiation protection and suppliers

This has been a major point of uncertainty for a while. So we started again from scratch and just see what is used in the industry instead of working it out for ourselves. I had detailed ChatGPT and Bard conversations to figure out what is traditionally used. It turns out that:

1. Transparent materials cannot really shield from any particle radiation. Only UV radiation and some solid debris. That is what the glass covers for solar panels protect against.

2. The rest of the particle radiation (electrons and protons) can be blocked using a 2mm Aluminium sheet for the body.

3. Neutrons are hard to block but are not a concern for low earth orbit.

So, the coating needed for the solar panels is a glass cover with a transparent adhesive. Both the glass and adhesive should let visible and some IR light through and block UV radiation. So, they should have high transparency (90 percent or more) between 300nm (UV cut off) and 2000nm (IR). The transparency should be close to 0 for light below 300nm in wavelength to block the UV and X-rays.

Borosilicate (borofloat 33) glass by SGP works well and has been used in space before. As for the adhesives, we select the ones mentioned in the Jaxa list of adhesives that are mentioned by ChatGPT and Bard. These were:

1. Araldite: https://www.go-araldite.com/en/contact-us

2. Loctite: https://www.loctiteproducts.com/en.html. However, the distributor for loctite is Henkel Adhesives: https://www.henkel-adhesives.com/sg/en/about/our-brands/loctite.html

These are the main sources with their contact-us pages from their websites. We are trying to reach out to these suppliers for adhesives now.

3 Code and Algorithms

The code is shared in separate files in the dedicated folders along with the text files for inputs, intermediate values, and the output values. The code is divided into separate dedicated files (supporting files) for functions that are separate from the main executable file. The 3 types of supporting files are:

1. Reader: reads the values from input/ intermediate files.

2. Writer: writes on to the intermediate and output files and to make the final plot.

3. Calculator: Calculates specific functions that are long and cumbersome

Just like last time. However, this time we have several different kinds of objects to store the information of one type for one size of the craft in one object. These objects are then passed into functions to do further calculations without having to pass several different variables. We also made arrays of the objects such that one array contains all the information for the inputs from one text file, one array for intermediate values, and one single array for the outputs.

Other things we need to cover are:

1. What the variables mean

2. What each function does

3. What are the inputs and outputs

4. A brief explanation of how the code works

3.1 Classes and Objects

class bodyinps: "given inputs about the body. Object array craft[]"

- U = 1
- L = 100
- W = 100
- H = 100
- mass = 2

class calcedinps: "calculated inputs from ses 2. Object array varinps[]"

- ptot = 109.2
- mtot = 12.7
- vtot = 15.5
- lenopen = 1.5
- widopen = 0.2

These are the 2 input classes, used by the reader file for setting up the values in the craft and varinps arrays which are used for further calculations. The variable names for the elements of the classes are the same as the ones defined in the problem statement.

class intervals: "Class for intermediate values. Object array intvars[]"

- mbody = 0
- dcom = 0
- moipanels = [0,0,0]
- moibody = [0,0,0]
- mpa = 5 "no need for mpa for folded panels"

This is the class used for storing the intermediate values.

class outvals: "Class for final putputs. Object array outvars[]"

- MOI = [0,0,0]
- powmass = 6
- powvol = 3

This is the class used for storing the final output values. Please note that the values for the variables used in the class initialization are just random and do not need to be there. Also note that the classes mentioned here are for the folded panels mechanism. The same classes are used for the outer cover panels, with some minor changes where lenopen and widopen are arrays instead.

3.2 Inputs

"craft inputs.txt"

- U L (mm) W (mm) H (mm) mass (kg)
- 1 113.5 100 100 2
- 2 227 100 100 4
- 3 340.5 100 100 6

- 6 340.5 226.3 100 12
- 8 454 226.3 100 16
- 12 340.5 226.3 213.5 24
- 16 454 226.3 213.5 32

Folded panels:
"output vals 6U.txt"

- 204 : (ncells) Total number of cells
- 244.8 : (ptot) Total available power in W
- 4495.36 : (vtot) Volume occupied by full assembly in cm^3
- 11.58 : (mtot) Total mass of panels in kg
- 4.03 : (lenopen) Length of deployed assembly in m
- 0.32 : (widopen) Width of deployed assembly in m

The output values for various sizes from running the session 2 code are stored in this format "output vals (i)U.txt" where i is the size in U.

Outer cover panels:
"output vals 6U.txt"

- 62 : (ncells) Total number of cells
- 74.4 : (ptot) Total power generated in W
- 1108.61 : (vtot) Total volume taken up in cm^3
- 3.49 : (mtot) Total mass in kg
- 1.386 : 1.259 : (lenopen[0,1]) Length of opened assembly in m
- 0.199 : 0.061 : (widopen[0,1]) Width of opened assembly in m

3.3 Intermediate Values

Folded panels:
"intermediate values.txt"

- mbody dcom moipanels[0 1 2] moibody[0 1 2]
- 0.51 74.5 0.3 0.0 0.3 0.0004 0.0005 0.0009
- 1.03 74.25 0.59 0.01 0.6 0.0009 0.0044 0.0053
- 1.54 74.33 0.89 0.04 0.93 0.0013 0.0149 0.0162
- 0.42 96.5 15.67 0.1 15.77 0.0018 0.0041 0.0059
- 0.57 96.44 20.88 0.24 21.12 0.0024 0.0098 0.0122
- 12.42 103.01 15.67 0.1 15.77 0.053 0.12 0.173
- 16.57 102.95 20.88 0.24 21.12 0.0707 0.2846 0.3553

Outer cover panels:
"intermediate values.txt"

- mbody dcom moipanels[0 1 2] moibody[0 1 2]
- 1.88 6.81 0.0 0.0 0.0 0.0016 0.0016 0.0032

- 3.05 53.91 0.032 0.032 0.064 0.0025 0.0025 0.005
- 4.46 87.39 0.105 0.105 0.21 0.0037 0.0037 0.0074
- 8.51 99.03 0.453 0.113 0.566 0.0071 0.0363 0.0434
- 9.51 184.15 1.641 0.32 1.961 0.0079 0.0406 0.0485
- 18.79 73.92 0.46 0.447 0.907 0.0714 0.0802 0.1516
- 23.48 120.88 1.645 0.944 2.589 0.0892 0.1002 0.1894

3.4 Output Values

Folded panels:
"output values.txt"

- U powmass powvol MOI[0 1 2]
- 1 20.94 49971.17 0.304 0.004 0.301
- 2 21.01 49971.571 0.599 0.022 0.605
- 3 20.987 49971.437 0.903 0.066 0.946
- 6 21.14 54456.15 15.676 0.108 15.776
- 8 21.154 54456.181 20.888 0.255 21.132
- 12 21.14 54456.15 15.996 0.493 15.943
- 16 21.154 54456.181 21.315 0.889 21.475

Outer cover panels:
"output values.txt"

- U powmass powvol MOI[0 1 2]
- 1 0.0 0.0 0.0 0.0 0.0
- 2 20.211 62579.447 0.072 0.072 0.069
- 3 18.701 56861.932 0.241 0.241 0.217
- 6 21.318 67111.067 0.747 0.436 0.609
- 8 16.641 51227.807 2.444 1.156 2.01
- 12 23.954 70607.005 1.004 1.0 1.059
- 16 18.028 53009.021 3.023 2.333 2.778

3.5 Functions

Simple self explanatory functions are used like *get_inter_vals*() and *get_outvals*() are used to get the intermediate and output values and store them in the calculation file. The equations implemented are the same as the problem statement. The names for the read and write functions are equally as self explanatory.

However, we have a new function this time to make the graphs called *make_graphs*() which plots all the graphs for the current configuration. This gives us the plots that help us visualize the situation.

3.6 Output plots

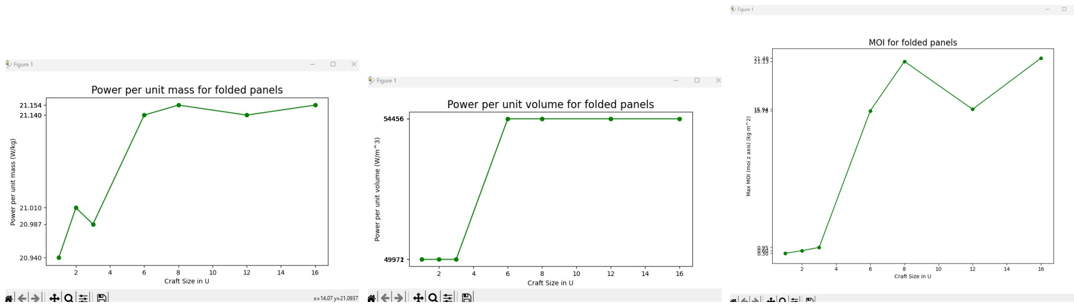

Figure 1: Plots for folded panels configuration (a) Power per unit mass (W/kg) (b) Power per unit volume (W/m^3) (c) MOI along z axis for folded panels (kgm^2)

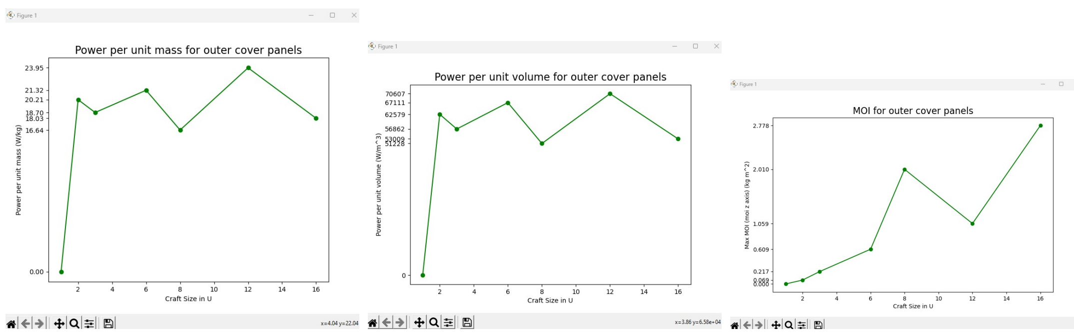

Figure 2: Plots for outer cover panels configuration (a) Power per unit mass (W/kg) (b) Power per unit volume (W/m^3) (c) MOI along z axis for outer cover panels (kgm^2)

4 Learning and Insights

By running the codes from session 2, we realize that the size of solar panels we have does not fit on the 1U craft. So it should not be included in the calculations.

Based on the plots made for the 2 craft types, we see that the outer cover panels consistently have more power per unit mass and per unit volume while at the same time, having lower overall MOI for the craft. This means that for most cases, the outer cover mechanism is a superior choice unless a large amount of power is needed. Only the total amount of power available from the folded panels is higher so only if power is a priority over everything else, do we use the folded mechanism.

Because of this clear and obvious difference, there is no need to plot the folded and outer cover outputs in the same plot to compare. However, this means that the subsequent lessons will need to change and be more straightforward and less uncertain.

Here are some other things I learned while compiling this:

- Python data structures and implementation for classes, arrays and variables is different from the C++ implementations I was familiar with. Thus, implementing classes, passing arrays of objects, and dynamic variables in loops were hard for me to adapt to. This delayed the submission of the Session 3 solution.
- There are some unique insights revealed about the 2 different configurations from this exercise as mentioned above. In addition, internal discussions within Galamad have changed the design of the ADCS subsystem set for the upcoming sessions. Thus, the search and calculations can be significantly different from my original plans.
- Subsequent sessions shall get progressively more streamlined as I learn more about Python and the design constraints and relationships come into clearer focus with the results of these sessions.

This brings home the fact that this is a live project that changes with time even more so than before.

References

[1] . O. L. n. Poly San, C. Cubesat design specification rev. 14. *The CubeSat Program, Cal Poly SLO CubeSat Design Specification.*

[2] J. n. Zhou. Attitude determination and control of the cubesat mist space technology.