PROBLEM SET #6

TO: PROF. DAVID MILLER, PROF. JOHN KEESEE, AND MS. MARILYN GOOD

FROM: NAMES WITHHELD

SUBJECT: PROBLEM SET #6 (STRUCTURE, THERMAL, AND COST)

DATE: 6/21/2004

MOTIVATION

Thermal radiation is an important consideration in spacecraft design. A spacecraft's skin must account for thermal effects, but it must also be inexpensive and reasonable to build. Trade-offs can be made between the thermal considerations, structural materials, and cost.

PROBLEM STATEMENT

Design a spacecraft structure to account for thermal radiation effects while minimizing cost. The spacecraft is assumed to be spherical with a set volume. Structural materials available include: Aluminum Alloy 2024-T4, Austenitic Stainless Steel 304, Titanium-Aluminum Alloy Ti-6Al-4V, Gold, and Graphite Epoxy. The skin may be either one or two layers thick and composed of different material.

APPROACH

The spacecraft shell is designed account for thermal effects while considering cost. If the volume of the spacecraft is known as well as information about the load it will have to withstand, then the different materials can be examined to determine three related factors: the thickness and mass of material that would be needed for the structure, the material that can compensate for thermal effects, and the overall cost. For the case of a single layer shell, the shell material will both have to provide structural support and have the right properties to account for thermal radiation. For a two layer shell, the inside shell will be sized to support the structure, and the outside shell will be a thin sheet that must account for thermal effects. The satellite is assumed to be spherical. The tool produces tables of different materials (or combinations of materials) that work given the input, as well as the structural mass and the estimated cost. These can be analyzed to see tradeoffs between the different subsystems.

SOLUTION

A tool using MATLAB evaluates the three subsystems. The user specifies various constraints on the satellite as inputs. These inputs are:

P: load on the satellite [N]
M: the largest moment applied to the satellite [Nm]
altitude: average height of the satellite above the earth's surface [m]
rad: radius of satellite (perhaps limited by launch vehicle choice) [m]
Q_w: electrical power dissipation [W]
T_user_max: desired maximum operating temperature of the satellite [Kelvin]
T_user_min: desired minimum operating temperature of the satellite [Kelvin]
ProductionNumber: number of satellites of this design needed

The outputs provide the results for total shell structure mass and estimated satellite cost (using two cost estimation methods) for all combinations of materials. The outputs are:

metalname: an array of the names of the different metals used, for reference
mass: a matrix of the structure mass for all combinations of materials
cost_rdte: a matrix of the estimated cost of the total number of satellites specified in *ProductionNumber*, using the Research Design Test and Evaluation Cost and Production Cost Model
cost_sscm: a matrix of the estimate cost of the total number of satellites specified in *ProductionNumber*, using the Small Satellite Cost Model

Examples of the outputs and how they should be used will be shown in the *Conclusions* section. The cost models will be described further in this section.

The tool is called by the top-level code *mat.m*. To run the tool, at the MATLAB prompt, type the following with the inputs and outputs filled in accordingly:

```
[metalname, mass, cost_rdte, cost_sscm]= …
        mat(P, M, altitude, rad, Q_w, T_user_max, T_user_min, ProductionNumber)
```

The properties of the different metals to be evaluated are defined inside the code. Emissivity and absorbtivity vary slightly with temperature; for metals where more than one value was found in references, an average value is used. The tool then loops through the different materials, performing all calculations for both a single layer shell (using the same material when considering structure and thermal) and double layer shell (using an inner shell material for structure and an outer for thermal). First, the thickness of the shell layer required to support the structure is determined; this is explained in the Structure section below. Next, the thermal effects with respect to the properties of the outer shell are considered, described in the Thermal section below. Finally, the cost of each possible design is determined, in the manner described in the Cost section below. The resulting data is output in the variables above and will be explained in more detail in the *Conclusions*.

Structure

The structure code, *strength.m,* calculates how thick a shell material must be in response to the applied loads and moments. To determine thickness, certain material properties are needed: the tensile strength, the yield strength, and Young's modulus. The radius of the satellite is also needed, and may vary based on the desired design. As mentioned, the satellite is modeled as a sphere. For a sphere, the axial load equals the lateral load and is therefore referred to in the code and in this section as simply the load.

To simplify the problem, rigidity due to axial and lateral frequencies is neglected. Note that this does result in the thickness found of the structure to be less than it might be otherwise. Also, stress due to internal pressure is ignored. Bending is assumed to occur at the furthest distance from the center of the satellite (the radius) so the worst case is calculated.

Safety factors also need to be considered. Safety factors for different types of testing options are given in Table 11-54 in SMAD. In this design, it will be assumed that option 2 in the table, 'proof test of all flight structures', will be used. So the yield safety factor *SF_yield* is set to 1.1 and the ultimate safety factor *SF_ultimate* is set to 1.25.

First, the ultimate load and the thickness required to withstand it (*t_ultimate*) is calculated:

$$Ultimate\_load = (P + 2M/rad)*SF\_ultimate \ \ [\mathrm{N}] \tag{1}$$

$$t\_ultimate = Ultimate\_load/(2\pi*rad*Tensile\_strength) \ \ [\mathrm{m}] \tag{2}$$

where *P, M,* and *rad* are the inputs described above, and *Tensile_strength* is the tensile strength of the material being evaluated. Next, the yield load and the thickness required to withstand it (*t_yield*) is calculated:

$$Yield\_load = (P + 2M/rad)*SF\_yield \ \ [\mathrm{N}] \tag{3}$$

$$t\_yield = Yield\_load/(2\pi*rad*Yield\_strength) \ \ [\mathrm{m}] \tag{4}$$

where *Yield_strength* is the yield strength of the material being evaluated. The greater of these two thicknesses is used, but it then must be adjusted to account for compressive forces. The buckling stress *buckle* and the buckling load *P_buckle* are found by the equations:

$$\varphi = (1/16)*\mathrm{sqrt}(rad/t) \tag{5}$$

$$\gamma = 1 - 0.901*(1.0 - e^{-\varphi}) \tag{6}$$

$$buckle = 0.6\gamma*Young/rad \ \ [\mathrm{N/m^2}] \tag{7}$$

$$P\_buckle = buckle*2\pi*(rad^2 - (rad-t)^2) \ \ [\mathrm{N}] \tag{8}$$

where *t* is the greater of the yield and ultimate thicknesses, φ is a geometric parameter, γ is a reduction factor, and *Young* is the Young's modulus of the material. The value obtained for *P_buckle* must be less than that found for *Ultimate_load*. If this is not the case, the thickness is increased in small increments until this constraint is satisfied.

Thermal

Not all materials are reasonable to use when thermal effects are considered. Whether or not a material will satisfy the thermal constraints is determined using the code *thermal.m*. It returns a '1' if the design is feasible, and a '0' if it is not.

To determine whether a design is feasible, the absorbtivity and emissivity of the shell material is needed. If a one-layer shell is used, the absorbtivity (α) and emissivity (ε) are the properties of the material also used to support the structure requirements. If a two-layer shell is used, the emissivity and absorbtivity is assumed to be that of the material being considered for the outer layer.

The thermal calculations assume that the satellite is an isothermal sphere, and that there is uniform electrical generation on the sphere. The code does not involve maintenance of satellite battery limits; it could be assumed that they have a separate control system with thermostats.

The worst-case maximum temperature (*T_max_s*) and worst-case minimum temperature (*T_min_s*) of the satellite must be found. These equations use the following parameters:

G_S: solar constant
a: albedo
q_I: earth IR emission
σ: Stefan-Bolztmann's constant
R_E: radius of earth
ϱ: angular radius of the earth, $\text{asin}(R_E/(altitude+R_E))$
K_a: factor for reflection of collimated incoming solar energy off of earth, $0.664 + 0.521\varrho - 0.203\varrho^2$
D: diameter of satellite, including thickness of shell
A: surface area of satellite
A_C: cross section area of satellite
F: view factor of an infinitesimal sphere viewing a finite sphere, $(1 - \cos(\varrho))/2$

The equations are as follows:

$$T\_max\_s = ((A_C G_S \alpha + AFq_I\varepsilon + AFG_S a\alpha K_a + Q\_w)/(A\sigma\varepsilon))^{1/4} \text{ [Kelvin]} \quad (9)$$

$$T\_min\_s = ((AFq_I\varepsilon + Q\_w)/(A\sigma\varepsilon))^{1/4} \text{ [Kelvin]} \quad (10)$$

where *Q_w* is a system input described above.

The values for *T_max_s* must be less than the user specified value *T_user_max* and *T_min_s* must be greater than the user specified value *T_user_min*. This ensures that the spacecraft will never reach a temperature that it cannot operate at.

Additionally, the necessary area for a space radiator is found, as an additional check that the design is reasonable. The space radiator cools the system by removing excess heat. It must be able to do its job of cooling but be a reasonable size to fit on the satellite. It is assumed that the radiator will be placed at a 90 degree angle from the sun so it will not face the sun. Also, it is assumed that the radiator does not absorb IR emission from the earth or albedo from the earth. The area of the space radiator *A_r* is:

$$A\_r = Q\_w/(\sigma\varepsilon T^4) \text{ [m}^2\text{]} \quad (11)$$

where *T* is the average of *T_user_max* and *T_user_min*, which is assumed to be the approximate average operating temperature of the spacecraft. If the area of the space radiator is less than the surface area of the spacecraft, and the maximum and minimum temperatures are valid, then the design is feasible.

Cost

Satellites have a wide array of components that contribute to their cost. Fortunately there are Cost Estimating Relationships (CERs) that are mathematical expressions that relate a characteristic of a satellite component to either its cost or the cost of another component. CERs are used in the code *SATELLITE_COST.m* in order to calculate the cost of the small satellite with thermal characteristics calculated previously. The thickness of the shell was found in calculations of the other subsystems. From the previous calculation for the thickness

of the shell, the mass of the shell can be found using the density of the material(s) used. The mass of the structural components is then used in order to determine the cost of the entire satellite.

Two mass CERs are used to determine the cost of the satellite. The first CER is in two parts: the Research, Design, Test, and Evaluation cost (*RDT & E*) and the Production cost (*Production*). The second CER comes from the Aerospace Corp. Small Satellite Cost Model Ver 7.4. Both CERs take structural mass as input and output the cost of the satellite. Once the calculation is complete, one can compare the results of the two CERs and judge for themselves which is more accurate.

Along with the costing of the entire satellite, the cost of an entire program of N satellites is calculated in order to determine any economies of scale that result from choosing one design over another. As stated in previous chapters, the two designs are either a single layer structure, or a double layer structure. In order to characterize the difference in design from an economic point of view, a learning curve of 85% was applied to the single layer design and a learning curve of 90% was applied to the double layer design. Using the learning curve with each design, the average system costs along with the individual unit costs are calculated so that the results can be compared between designs.

The process that is used to determine the cost of all possible satellite designs is as follows:

The first CERs used to calculate the cost of the satellites are the following:

$$RDT\ \&\ E\ = 2460 + 416(Mass\{kg\})^{0.66}\ \ [\$K92] \tag{12}$$

$$Production = 86(Masss\{kg\})^{0.65}\ \ [\$K92] \tag{13}$$

The results are summed together to get the total cost of the satellite. The second CER used is from the Aerospace Corp. Small Satellite Cost Model Ver.7.4:

$$TotalBusCost = 1.47 + 0.7(Mass\{kg\})\ln(Mass\{kg\}^{0.66})\ \ [\$M94] \tag{14}$$

The mass for each CER is changed according to the design, resulting in a satellite cost for the single and double layer designs.

Once the cost of each design is calculated, the resulting value is set as the cost of the first unit of production. Next, the learning curve is applied in order to calculate the cost of the next N-1 units of production, along with the average system cost.

Note that this code itself outputs a structure with a variety of the cost data; not all of this information is output from the tool. If more detail is desired concerning the satellite cost, it can be found by running this code separately and examining the output structure SATELLITECOST.

CONCLUSIONS

Several examples cases were run to prove functionality and demonstrate trends. In all cases, the load *P* was set to 127500 N, the moment *M* to 294200 Nm, the altitude to 700000 m, the electrical dissipation *Q_w* to 170 Watts, the maximum temperature *T_user_max* to 350 Kelvin, and the minimum temperature *T_user_min* to 200 Kelvin. All of these values are similar to those used in examples in the textbook. The number of satellites to be produced was set arbitrarily to 10; so values output for cost represent the cost of 10 satellites.

To easily get the cost for one, the tool could be run with *ProductionNumber* set to 1. The radius of the satellite was varied to demonstrate how the design changes for different sized satellites.

Tables 1, 2, and 3 show the complete outputs for the tool run with the above inputs and a satellite radius of 1 meter. These tables demonstrate how the outputs are read. Table 1 shows the total structural mass for different combinations of materials. The diagonal of the table shows the case where only a single layer is used. For the other entries, the material listed at the top of the column is the material used for the inner layer, while the material listed at the beginning of the row is the material for the outer layer. The mass is the total mass of both materials. Tables 2 and 3 are set up the same way, except that they show the total estimated cost for 10 satellites for each material or combination of materials, using the two different CERs described above. An entry in any table of '-1' indicates that the design is not feasible; the thermal effects cannot be accounted for with the given materials with the specified inputs.

		Inner Layer				
		Aluminum Alloy	Steel	Titanium Alloy	Gold	Graphite Epoxy
Outer Layer	Aluminum Alloy	-1	-1	-1	-1	-1
	Steel	15.4533	60.6573	8.6411	214.5298	25.3174
	Titanium Alloy	14.6052	62.3437	7.9293	197.3102	23.7425
	Gold	-1	-1	-1	-1	-1
	Graphite Epoxy	-1	-1	-1	-1	-1

Table 1. Necessary Mass of Materials (kg)

		Inner Layer				
		Aluminum Alloy	Steel	Titanium Alloy	Gold	Graphite Epoxy
Outer Layer	Aluminum Alloy	-1	-1	-1	-1	-1
	Steel	0.5155e8	0.8295e8	0.4249e8	1.8417e8	0.6250e8
	Titanium Alloy	0.5051e8	0.9446e8	0.3687e8	1.7553e8	0.6087e8
	Gold	-1	-1	-1	-1	-1
	Graphite Epoxy	-1	-1	-1	-1	-1

Table 2. Estimated Cost (Dollars) to Produce 10 Satellites Satisfying Inputs, using RDT&E and Production Model

		Inner Layer				
Outer Layer		Aluminum Alloy	Steel	Titanium Alloy	Gold	Graphite Epoxy
	Aluminum Alloy	-1	-1	-1	-1	-1
	Steel	0.3217e8	0.9820e8	0.2228e8	4.2313e8	0.4784e8
	Titanium Alloy	0.3033e8	1.1346e8	0.1897e8	3.8634e8	0.4526e8
	Gold	-1	-1	-1	-1	-1
	Graphite Epoxy	-1	-1	-1	-1	-1

Table 3. Estimated Cost (Dollars) to Produce 10 Satellites Satisfying Inputs, using Small Satellite Cost Model

In this case, only designs with an outer layer of steel or the titanium alloy satisfied the constraints. Other materials could not keep the satellite within the desired temperature range. Other cases were tested with the radius sizes of 0.5 m and 2.0 m. Instead of showing all outputs for these, plots were created demonstrating the cost for the full production line of 10 satellites compared with the structural mass of one. The cost using the Small Satellite Cost Model is shown. Although the other CER is more commonly used, the Small Satellite Cost Model is newer and expected to be more accurate. In each plot, the materials corresponding to each point are shown in the legend. For the legend items that are pairs of materials, the first material listed is the material of the outer shell, the second the inner. Figure 1 shows the trending for a satellite with radius 0.5 m, Figure 2 for a satellite with radius 1.0 m (data taken from the tables above), and Figure 3 for a satellite with radius 2.0 m.

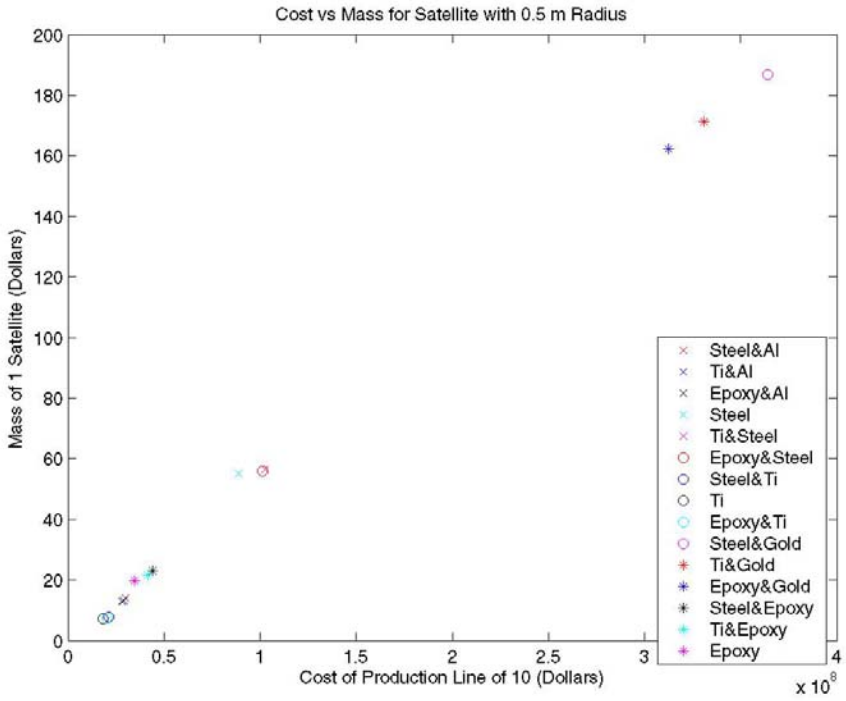

Figure 1. Cost of 10 Satellites vs. Mass of Structure of Each, 0.5m Radius

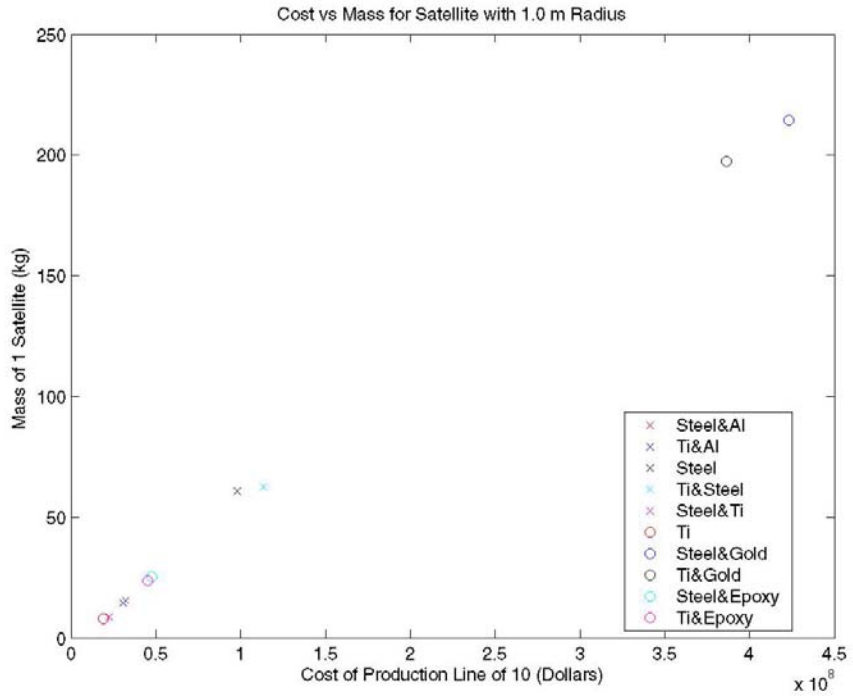

Figure 2. Cost of 10 Satellites vs. Mass of Structure of Each, 1.0m Radius

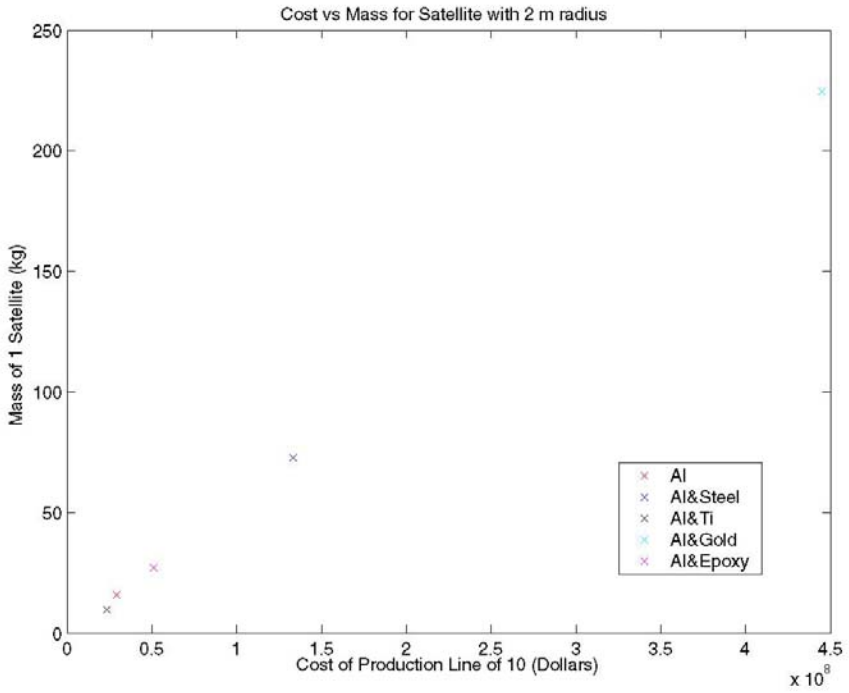

Figure 3. Cost of 10 Satellites vs. Mass of Structure of Each, 2.0m Radius

For different sized satellites, different combinations of materials satisfied the constraints. Titanium alloy is consistently the lightest and cheapest material. For satellites with 0.5m and 1.0m radii; the best choice is a single layer of titanium. For a 2.0m radius, an aluminum alloy outer shell with a titanium alloy inner shell is the best choice. These results are as expected, since both alloys are commonly used in satellite structure design. Any combinations with gold as the structural element are shown to be heavy and expensive, also as expected. The best results of the three test runs (lowest mass and lowest cost) are shown in Table 4.

Radius (m)	Material	Cost for 10 (Million$)	Structural Mass of Each (kg)
0.5	Single-layer Titanium Alloy	1.813	7.2167
1.0	Single-layer Titanium	1.897	7.9293
2.0	Aluminum outside, Titanium inside	2.366	9.6393

Table 4. Lower cost and mass results for different sized satellites

REFERENCES

Wertz and Larson, Space Mission Analysis and Design, 3rd Edition, Space Technology Series, Space Technology Library, Microcosm Press, El Segundo CA, 1999

Touloukian and Ho, *Thermophysical Properties of Selected Aerospace Materials Part I: Thermal Radiative Properties*, Purdue Research, West Layfette IN, 1976

Boyer and Gall, *Metals Handbook*, American Society for Metals, Metals Park OH, 1985

Metals Reference Book, American Society for Metals 2nd Edition, 1983

Sala, *Radiative Properties of Materials*, Elsevier Science Publishers, New York, 1986

Metals Handbook: Properties and Selection of Nonferrous Alloys and Pure Metals, 9th Edition Vol. 2, American Society for Metals, 1979

Holman, Heat Transfer, McGraw-Hill, Inc, New York, 1997

Miller, *Lecture Notes: Cost Modeling*, Massachusetts Institute of Technology, 16.851, 2003

Wertz and Larson, Reducing Space Mission Cost, Space Technology Series, Space Technology Library, Microcosm Press, Torrance CA, 1996.

www.me.poly.edu/hk/subject/me262/notes/chapt9.pdf

MATLAB CODE

```
%This is the main file of the program.  It lists the material data and calls all functions
function [metalname, mass, cost_rdte, cost_sscm] =  mat(P, M, altitude, rad, Q_w, T_user_max, T_user_min, ProductionNumber)

%Inputs to the function:
%P is the load in Newtons, M is the moment on the structure in Nm.
%altitude of the orbit in meters
%radius of the total structure in meters
%Internal electrical power dissipation (Q_w) in Watts
%T_user_max is the largest operating temperature in Kelvin
%T_user_min is the lowest operating temperature in Kelvin
%Production number is the desired number of units...must be an integer

%Outputs to the function
%metal name is a cell array of the names of the metals.  they correspond to
%the other output arrays - each piece of data represents the metals in this
%order, along the side and top of the array.  in the arrays, the diagonal represents one metal.
%all other locations represent two metals, with the metal in the column representing the inner
%metal and the metal in the row representing the outer
%mass gives the total mass of the structure for each case
%cost_rdte gives the cost of the production line for each case using the
%research design test and evaluation model
%cost_sscm gives the cost of the production line for each case using the
%small satellite cost model
%a value in any array of -1 indicates the design is not feasible

%CONSTANTS
SF_yield    = 1.1;              %Can be taken from SMAD Table 11-54 for different options in testing
SF_ultimate = 1.25;             %Can be taken from SMAD Table 11-54 for different options in testing
YEAR        =   2000;
```

```
Inflation   =   .02;          % No Need to Change

%Calculate exterior volume from given radius
Volume = (4/3)*pi*rad^3;

%The metal (material) names and properties are listed below.  More can be added to this list and would be accounted for in the code
%The properties below are approximate and taken from the following refrences
%Reference 1:  Thermophysical Properties of Selected Aerospace Materials Part I:  Thermal Radiative Properties Y.S. Touloukian and C.Y. Ho
%ref1 contd: 1976 Purdue Research West Layfette Ind. 47907
%Reference 2:  American Society for Metals.  Metals Handbook, Metals Park, Ohio 44073 Edit by Howard E. Boyer, Timothy L. Gall, 1985
%Reference 3:  ASM metals Reference Book 2nd edition 1983
%Reference 4:  Radiative Properties of MAterials, Aleksander Sala, Elsevier Science Publishers NY,NY 10017, 1986
%Reference 5: Metals Handbook 9th edition vol. 2 Properties and selection of nonferrous alloys and pure metals,1979, ASM
%Reference 6: (for the graphite structural data) www.me.poly.edu/hk/subject/me262/notes/chapt9.pdf

metal.name={'Aluminum Alloy 2024 T4';'Austenitic Stainless Steel 304';'Titanium Alloy Ti-6Al-4V';'Gold';'Graphite Epoxy'};
metalname = metal.name;

metal.density(1,1)    =   2795; %kg/m^3
metal.density(2,1)    =   7900; %kg/m^3
metal.density(3,1)    =   4400; %kg/m^3
metal.density(4,1)    =   1800; %kg/m^3
metal.density(5,1)    =   2420; %kg/m^3


metal.emmisivity_value(1,1) =   (.030+.036+.040+.043+.046+.048)/6;  %Average value over temperature
metal.emmisivity_value(2,1) =   (.138+.154+.170+.186+.202+.212)/6;
metal.emmisivity_value(3,1) =   (.151+.162+.174+.187+.202+.210)/6;
metal.emmisivity_value(4,1) =   (.019+.021+.023+.026+.028+.029)/6;
metal.emmisivity_value(5,1) =   .888;  %Constant value, so no need to average

metal.absorb_value(1,1)     =   (.030+.036+.040+.043+.046+.048)/6;
metal.absorb_value(2,1)     =   (.138+.154+.170+.186+.202+.212)/6;
metal.absorb_value(3,1)     =   (.136+.138+.140+.141+.142+.142)/6;
metal.absorb_value(4,1)     =   .300;
metal.absorb_value(5,1)     =   .888;

metal.tensile_strength(1,1)  =   470e6;  %Pa
metal.tensile_strength(2,1)  =   515e6;  %Pa
metal.tensile_strength(3,1)  =   993e6;  %Pa
metal.tensile_strength(4,1)  =   130e6;  %Pa
metal.tensile_strength(5,1)  =   2000e6;  %Pa

metal.yield_strength(1,1)   =   325e6; %Pa
metal.yield_strength(2,1)   =   205e6; %Pa
metal.yield_strength(3,1)   =   924e6; %Pa
metal.yield_strength(4,1)   =   16e6; %Pa
metal.yield_strength(5,1)   =   1750e5; %Pa

metal.Young(1,1)    =   72.4e9; %Pa
metal.Young(2,1)    =   193e9;  %Pa
metal.Young(3,1)    =   113.8e9; %Pa
metal.Young(4,1)    =   79.9e9;  %Pa
metal.Young(5,1)    =   110e9; %Pa

%Determine the number of metal considered.
num =   length(metal.density);



%Assume that the inner layer provides the structure, but the outer layer provides the thermal protection
      for i = 1:num
         for(j=1:num)
            if(i==j)
               %If there is only one layer, it must provide the sturcutre and thermal protection
               thickness(i,j)    =   strength(P,M,rad,SF_ultimate,SF_yield,metal.tensile_strength(i),metal.yield_strength(i),metal.Young(i));
               %Feasible determines if the design determined by the strength codes is feasible from a thermal standpoint
                feasible = thermal(metal.absorb_value(i), metal.emmisivity_value(i), T_user_max, T_user_min, Volume, altitude, Q_w);
                if feasible == 1
```

```
            %Now we must compute the mass
            mass(i,j)    =    metal.density(i)*(Volume - ((4/3)*pi*(rad-thickness(i,j))^3));
            %Now compute the cost
            MATERIAL.NAME = metal.name(i);
            MATERIAL.MASS.SINGLE_LAYER = mass(i,j);
            MATERIAL.MASS.DOUBLE_LAYER = 1;
            SATELLITECOST = SATELLITE_COST(MATERIAL, YEAR, Inflation, ProductionNumber);
            cost_rdte(i,j) = sum(SATELLITECOST.UnitCosts.SINGLE);
            cost_sscm(i,j) = sum(SATELLITECOST.UnitCosts_SSCM7_4.SINGLE);
         else
            mass(i,j) = -1;
            cost_rdte(i,j) = -1;
            cost_sscm(i,j) = -1;
         end
      end
      if(j~=i)
         %If there are two layers, the inner holds the structure, while the outer provides the thermal protection
         %The thickness of the outer layer is determined to be 5 percent of the inner layer
         tinner(i,j)    =    strength(P,M,rad,SF_ultimate,SF_yield,metal.tensile_strength(j),metal.yield_strength(j),metal.Young(j));
         touter(i,j)    =     .05*tinner(i,j);
         thickness(i,j)   =    tinner(i,j)+touter(i,j);

         %The emmisivity and the absorbtivity are just the values of the outer sheel
         feasible = thermal(metal.absorb_value(i), metal.emmisivity_value(i), T_user_max, T_user_min, Volume, altitude, Q_w);

         if feasible == 1
            mouter(i,j)   =    metal.density(i)*(Volume - (4/3)*pi*(rad-touter(i,j))^3);
            minner(i,j) =    metal.density(j)*((4/3)*pi*(rad-touter(i,j))^3 - (4/3)*pi*(rad-touter(i,j)-tinner(i,j))^3);
            mass(i,j)    =    mouter(i,j)+minner(i,j);
            %Now compute the cost
            MATERIAL.NAME = {[metal.name(i), metal.name(j)]};
            MATERIAL.MASS.SINGLE_LAYER = 1;
            MATERIAL.MASS.DOUBLE_LAYER = mass(i,j);
            SATELLITECOST = SATELLITE_COST(MATERIAL, YEAR, Inflation, ProductionNumber);
            cost_rdte(i,j) = sum(SATELLITECOST.UnitCosts.DOUBLE);
            cost_sscm(i,j) = sum(SATELLITECOST.UnitCosts_SSCM7_4.DOUBLE);
         else
            mass(i,j) = -1;
            cost_rdte(i,j) = -1;
            cost_sscm(i,j) = -1;
         end

      end
   end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [t]    =    strength(P,M,rad,SF_ultimate,SF_yield,Tensile_strength,Yield_strength,Young)

%This function inputs P, which is the load applied in Newtons, and M, the Moment applied in N-m.  In addition, the radius, rad of the sphere
%must be defined.  %Also, the safety factor for the structure needs to be given.  For help choosing a safety factor, SMAD pg 468, table 11-54
%gives a guide.  Both the ultimate safety factor and the yield safety factor should be provided

%Rigity due to both axial and lateral frequencies is neglected
%Ignoring stress due to internal pressure
%For a sphere, all is symmetric and the axial load equals the lateral load, which equals the load and is therefore referred to only as a load
%The bending will be assume to act at the furthest distance for the worst case (radii)

%The equations to determine the thickness are taken from SMAD pg 488
%We first size for tensile strength and then check the thickness agains a compressive load
%In order to size for strength the effective load, including safety factor must be determined equation 11-51

Load_limit  =    P + 2*M/rad;

%Determine the ultimate load
Ultimate_load    =    Load_limit*SF_ultimate;
```

```
%Size the thickness of the material to withstand the ultimate load
%Approximating cross-sectional area as 2*pi*rad*thickness
t_ultimate  =   Ultimate_load/(2*pi*rad*Tensile_strength);

%Determine the yield load
Yield_load  =   Load_limit*SF_yield;

%Determine the thickness of the material to withstand the yield load

t_yield =   Yield_load/(2*pi*rad*Yield_strength);

%The maximum value of the thickness, is the thickness of the material
t   =   max(t_ultimate,t_yield);

%Now, we determine if this thickness can withstand the necessary compressive forces SMAD pg480 eq. 11-52-11-54
phi =   (1/16)*sqrt(rad/t);
gamma   =   1-.901*(1-exp(-phi));

%Determine buckling stress and buckling load
%The area here is the actual cross-sectional area of the center of the sphere.
%This is the worst case scenario since the area is greatest and therefore the buckling force is greatest
buckle  =   .6*gamma*Young/rad;
P_buckle    =   buckle*2*pi*(rad^2 - (rad-t)^2);


%Determine if the thickness is enough.  Loop through, increasing the thickness until the buckling load (the load it can withstand) is greater than
%the ultimate load with safety factor included (the load it needs to withstand...within a factor of safety)
while(P_buckle/Ultimate_load < 1)
   t = 1.025*t;
   %Now, we determine if this thickness can withstand the necessary compressive forces SMAD pg480 eq. 11-52-11-54
   phi =   (1/16)*sqrt(rad/t);
   gamma   =   1-.901*(1-exp(-phi));

   %Determine buckling stress and buckling load
   %The area here is the actual cross-sectional area of the center of the sphere.
   %This is the worst case scenario since the area is greatest and therefore the buckling force is greatest
   buckle  =   .6*gamma*Young/rad;
   P_buckle    =   buckle*2*pi*(rad^2 - (rad-t)^2);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%references:
%SMAD
% Holman, J.P. Heat Transfer.  New York:  McGraw-Hill, Inc. 1997.

function feasible = thermal(alpha, epsilon, T_user_max, T_user_min, exterior_vol, altitude, Q_w)

% thermal - checks to see if spacecraft design is feasible from a thermal
% standpoint for a first-order approximation

% Assumptions:
% - isothermal sphere
% - uniform electrical generation on sphere
% - material may be one, two, or more layers thick, but usually
% still use epsilon to distinguish multiple layer insulation since radiation is extremely predominant
% over conduction in the space environment (SMAD pg451)
% - takes into account one half always in sun and view factor from Earth
% - no attempt to mainain spacecraft battery limits, possibly separate
% control system with thermostats

%Inputs:

% alpha - solar absorptivity of the sphere
% epsilon - IR emissivity of the sphere
% T_user_max [K] - worst-case hot temperature for equipment
% T_user_min [K] - worst-case cold temparture for equipment
% exterior_vol [m^3] - interior volume of satellite
% altitude [m] - height of spacecraft from earth's surface
```

```
% Q_w [W] - electrical power disappation (if not known use orbital average
% power

%Outputs
% feasible - 0 if not feasible, 1 if feasible

%Constants:
G_s = 1367; %[W.m^-2] solar constant
a = 0.3; %albedo 30% of direct solar
q_i = 237; %[W.m^-2] earth IR emission
sigma = 5.67051e-8; %[W.m^-2.K-4] Stefan-Boltzmann's constant
R_e = 6378e3; %[m] - radius of earth
rho = asin(R_e/(altitude+R_e));
K_a = 0.664 + 0.521*rho - 0.203*rho^2; % a factor which accounts for the reflection of collimated incoming solar energy off a spherical Earth
D = (6*exterior_vol/pi)^(1/3); %[m] calculation of outer diameter of satellite
A = pi*D^2; %[m^2] surface area of satellite, m^2
A_c = A/4; %[m^2] cross section area of the spherical satellite.
F = (1 - cos(rho))/2;

% Calculations for temperatures from SMAD pg447

T_max_s = ((A_c*G_s*alpha + A*F*q_i*epsilon + A*F*G_s*a*alpha*K_a + Q_w)/(A*sigma*epsilon))^(0.25);
T_min_s = ((A*F*q_i*epsilon + Q_w)/(A*sigma*epsilon))^(0.25);

% Calculations for space radiator
% Assume theta = 90 degrees since we will place space radiator not facing
% sun
% Assume radiator does not absorb IR emission from earth or albedo from
% earth.

T_ave = (T_user_max + T_user_min)/2;  % use average case temperature
A_r = Q_w/(sigma*epsilon*(T_ave)^4);

% Check for feasiblity
if T_user_max > T_max_s & T_user_min < T_min_s & A_r < A
   feasible = 1;
else
   feasible = 0;
end


T_max_s & T_user_min > T_min_s & A_r < A
   feasible = 1;
else
   feasible = 0;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function SATELLITECOST = SATELLITE_COST(MATERIAL, YEAR, Inflation, ProductionNumber)

%*********Asumptions

LearningCurveSlope.SingleLayer=.85;
LearningCurveSlope.DoubleLayer=.9;
MATERIAL.MASS.PercentTotalMass = 0.3;

for MATERIAL_INDEX=1:size(MATERIAL.NAME,2)

SATELLITECOST.RTDE.SINGLE(MATERIAL_INDEX)=((2460+416*(MATERIAL.MASS.SINGLE_LAYER(MATERIAL_INDEX))^0.66)*10
^3)*(1+Inflation)^(YEAR-1992);      %FY92->FY-YEAR

   SATELLITECOST.RTDE_ERROR.SINGLE(MATERIAL_INDEX)=(4773*10^3)*(1+Inflation)^(YEAR-1992) ;
%FY92->FY-YEAR


SATELLITECOST.Production.SINGLE(MATERIAL_INDEX)=((86*(MATERIAL.MASS.SINGLE_LAYER(MATERIAL_INDEX))^0.65)*10^3)
*(1+Inflation)^(YEAR-1992);      %FY92->FY-YEAR
```

```
   SATELLITECOST.Production_ERROR.SINGLE(MATERIAL_INDEX)=(1247*10^3)*(1+Inflation)^(YEAR-1992);
%FY92->FY-YEAR


SATELLITECOST.SSCM7_4_TotalBusCost.SINGLE(MATERIAL_INDEX)=((1.47+.07*(MATERIAL.MASS.SINGLE_LAYER(MATERIAL_IN
DEX))*...
      log(MATERIAL.MASS.SINGLE_LAYER(MATERIAL_INDEX))^0.66)*10^6)*(1+Inflation)^(YEAR-1994) ;
%FY94->FY-YEAR

   SATELLITECOST.SSCM7_4_ERROR.SINGLE(MATERIAL_INDEX)=(5.4*10^6)*(1+Inflation)^(YEAR-1994);
%FY94->FY-YEAR

   MATERIAL.MASS.TotalSatelliteMassSingle=MATERIAL.MASS.SINGLE_LAYER/MATERIAL.MASS.PercentTotalMass;

   MATERIAL.MASS.TotalSatelliteMassDouble=MATERIAL.MASS.DOUBLE_LAYER/MATERIAL.MASS.PercentTotalMass;


SATELLITECOST.RTDE.DOUBLE(MATERIAL_INDEX)=((2460+416*(MATERIAL.MASS.DOUBLE_LAYER(MATERIAL_INDEX))^0.66)*
10^3)*(1+Inflation)^(YEAR-1992);      %FY92->FY-YEAR

   SATELLITECOST.RTDE_ERROR.DOUBLE(MATERIAL_INDEX)=(4773*10^3)*(1+Inflation)^(YEAR-1992);
%FY92->FY-YEAR


SATELLITECOST.Production.DOUBLE(MATERIAL_INDEX)=((86*(MATERIAL.MASS.DOUBLE_LAYER(MATERIAL_INDEX))^0.65)*10^
3)*(1+Inflation)^(YEAR-1992);                %FY92->FY-YEAR

   SATELLITECOST.Production_ERROR.DOUBLE(MATERIAL_INDEX)=(1247*10^3)*(1+Inflation)^(YEAR-1992);
%FY92->FY-YEAR


SATELLITECOST.SSCM7_4_TotalBusCost.DOUBLE(MATERIAL_INDEX)=((1.47+.07*(MATERIAL.MASS.DOUBLE_LAYER(MATERIAL_
INDEX))*...
      log(MATERIAL.MASS.DOUBLE_LAYER(MATERIAL_INDEX))^0.66)*10^6)*(1+Inflation)^(YEAR-1994);
%FY94->FY-YEAR

   SATELLITECOST.SSCM7_4_ERROR.DOUBLE(MATERIAL_INDEX)=(5.4*10^6)*(1+Inflation)^(YEAR-1994);
%FY94->FY-YEAR



   for i=1:ProductionNumber
      %Single Layer
      p=(log10(LearningCurveSlope.SingleLayer)/log10(2));

      SATELLITECOST.AvgSystemCost.Single(MATERIAL_INDEX)=(SATELLITECOST.RTDE.SINGLE(MATERIAL_INDEX)+...
         SATELLITECOST.Production.SINGLE(MATERIAL_INDEX))*ProductionNumber^p/(1+p);

      SATELLITECOST.AvgSystemCost_SSCM7_4.Single(MATERIAL_INDEX)=...
         SATELLITECOST.SSCM7_4_TotalBusCost.SINGLE(MATERIAL_INDEX)*ProductionNumber^p/(1+p);


SATELLITECOST.UnitCosts.SINGLE(MATERIAL_INDEX,i)=(SATELLITECOST.RTDE.SINGLE(MATERIAL_INDEX)+SATELLITECOS
T.Production.SINGLE(MATERIAL_INDEX))*i^p;


SATELLITECOST.UnitCosts_SSCM7_4.SINGLE(MATERIAL_INDEX,i)=SATELLITECOST.SSCM7_4_TotalBusCost.SINGLE(MATERIAL_I
NDEX)*i^p;

      %Double Layer
      p=(log10(LearningCurveSlope.DoubleLayer)/log10(2));

      SATELLITECOST.AvgSystemCost.Double(MATERIAL_INDEX)=...

(SATELLITECOST.RTDE.DOUBLE(MATERIAL_INDEX)+SATELLITECOST.Production.DOUBLE(MATERIAL_INDEX))*ProductionNum
ber^p/(1+p);

      SATELLITECOST.AvgSystemCost_SSCM7_4.Double(MATERIAL_INDEX)=...
         SATELLITECOST.SSCM7_4_TotalBusCost.DOUBLE(MATERIAL_INDEX)*ProductionNumber^p/(1+p);
```

```
SATELLITECOST.UnitCosts.DOUBLE(MATERIAL_INDEX,i)=(SATELLITECOST.RTDE.DOUBLE(MATERIAL_INDEX)+SATELLITECO
ST.Production.DOUBLE(MATERIAL_INDEX))*i^p;


SATELLITECOST.UnitCosts_SSCM7_4.DOUBLE(MATERIAL_INDEX,i)=SATELLITECOST.SSCM7_4_TotalBusCost.DOUBLE(MATERIAL
_INDEX)*i^p;
   end

end
```