

Portfolio
Satellite Engineering
16.851

Department of Aeronautics and Astronautics
Massachusetts Institute of Technology

Fall 2003

Matthew Richards
Professor David W. Miller
Col. John E. Keesee

Tables of Contents

1. Introduction.....	3
2. Launch Systems	5
3. Orbital Mechanics.....	11
4. Power Systems	18
5. Propulsion Systems.....	24
6. Space Environment	31
7. Optics	34
8. Attitude Control Systems.....	36
9. Electromagnetic Formation Flight.....	38
10. Spacecraft Computers and Software.....	40
11. Thermal System	41
12. Communications, Telemetry, Tracking, & Command.....	44
13. Human Factors	51
14. Cost Modeling.....	64
15. Structures	66

1. Introduction

This section describes the teaching goals of Satellite Engineering and my overall experience in the class.

Course Description

This course covered the fundamentals of satellite engineering design. We studied the orbital environment and analyzed problems of station keeping, attitude control, communications, power generation, structural design, thermal balance, and subsystem integration. In class and in our problem sets, we considered trade-offs among weight, efficiency, cost, and reliability. Many topics were discussed in the context of current satellite systems, including choice of design parameters, such as size, weight, power levels, temperature limits, frequency, and bandwidth.

Learning Objectives

Six primary learning objectives were identified in the syllabus:

- Understand the function of spacecraft subsystems.
- Apply orbital mechanics formula and tools to spacecraft mission design.
- Select appropriate launch systems and understand their affect on satellite and payload design and performance.
- Evaluate spacecraft subsystem performance and trades
- Estimate space system costs
- Trade subsystem performance requirements to optimize higher-level system performance, cost, or weight.

Personal Motivation for Taking 16.851

There were many reasons I chose to take 16.851. First and foremost, I wanted to increase my knowledge of space systems. Having worked at JPL on instrumentation for a rover point design and on a DARPA satellite servicing project as a systems engineer, I had some applied experience in the area but little academic grounding on the subject. Secondly, I am interested in space systems as an area of concentration in graduate school and recognized 16.851 as an opportunity to both explore this area of interest and potentially earn units which could be applied to a graduate degree.

Reflections on Learning Experience

This term, Satellite Engineering has been my most challenging, most frustrating, and most enjoyable course. Working on self-created problem sets offered a level of academic freedom I have not experienced ever before. As I learned the hard way (i.e. Problem Set Two, cryo-cooled optics and thermal subsystem), this freedom required well-scoped problem statements while satisfying trades across all subsystems. Additionally, it was

inspiring to work with master's and doctorate students who were often very accomplished in their fields.

Taking Satellite Engineering and 16.83 during the same term, I found the courses to be mutually complimentary and did not get tired of the subject material. After spending my junior year at the University of Cambridge in England where few aerospace subjects are offered in the Engineering Department, I was glad to immerse myself in space systems coursework.

General References

Space Mission Analysis and Design (SMAD), Wertz & Larson, Third edition, Microcosm Press, 1999.

Fundamentals of Space Systems, Pisacane and Moore, Oxford, 1994

Space Systems Engineering, Fortesque and Stark, John Wiley and Sons, 1995

Space Vehicle Design, Griffin and French, AIAA 1991

Communications Satellite Handbook, Morgan and Gordon, 1989

2. Launch Systems

The launch system includes a single or multi-stage launch vehicle and a launch site with associated ground support. Since the 1960's launch vehicle performance has increased by an order of magnitude while reliability of launch systems have increased from 85% to 95%. For a typical spacecraft, the propellant mass fraction of the launch system is 0.85, while the structure and payload mass fractions are 0.14 and 0.01, respectively. The following graph from my fifth problem set solution depicts the trade between payload capacity and achievable orbit altitude.

Learning Objectives

- Understand launch system characteristics, sizing and trade-offs
- Estimate launch system sizes, staging requirements
- Be able to select appropriate launch system for a given mission from available systems
- Be able to estimate spacecraft requirements driven by launch vehicle induced environments
- Determine costs of launch systems

Reflections on Learning Experience

Launch systems have become one of my strongest systems in 16.851. Launch site selection was the topic of the first problem set I worked on and it was the module I coded in MATLAB for the fifth problem set. I found launch vehicles to be a good choice for conducting trades given the readily-available metrics on various systems, especially those included in the 1999 AIAA Reference Guide to Space Launch Systems.

Prominent U.S. Launch Vehicles

Atlas

Lockheed Martin's Atlas family of launch vehicles is one of the largest commercial launch vehicle lines in the United States. Four Atlas launch vehicles are included in the launch vehicle database. The Atlas IIA has an upgraded RL 10 engine with optional nozzle extension for the Centaur stage. The Atlas IIAS adds solid strap-on boosters to this architecture. The relatively new Atlas IIIA (2000) is a re-engineered Atlas II, utilizing a high-performance RD-180 engine from Russia. The Atlas IIIB has a stretched Centaur that can use single or dual Centaur configurations.

Delta

Following the Challenger shuttle disaster in 1986, President Reagan announced that the Space Shuttle would carry no more commercial payloads. To fill this gap, the Delta II was built by Boeing (1989) to launch medium-sized NASA payloads and small commercial payloads. Delta III development began in 1995 to launch heavier commercial payloads. The Delta III includes a larger fairing, larger strap-on motors, and a cryogenic second stage. The Delta IV line of medium-to-heavy launch vehicles is Boeing's most advanced product in the Evolved Expendable Launch Vehicle (EELV) program. The Delta IV Heavy is capable of delivering a payload directly to geostationary orbit.

Pegasus

Orbital Sciences Corporation's Pegasus XL (1994) is designed to launch small commercial and government payloads. The Pegasus XL is air-launched from Orbital's L-1011 Stargazer aircraft at an altitude of approximately 12 km and a speed of Mach 0.8. The Pegasus XL consists of a solid-propellant booster with wings. Advantages to the air launch concept include added initial velocity, lower drag, and flexible inclination angles.

Space Shuttle

NASA's Space Shuttle (first launched in 1981) consists of two recoverable solid-propellant rocket boosters, an expendable liquid propellant tank, and a reusable delta-wing space plane. Primary Space Shuttle missions include launch of heavy payloads to LEO, and transfer of humans and cargo to the International Space Station.

Taurus

The Taurus launch vehicle (1994) is Orbital Sciences Corporation's ground-launched derivative of the Pegasus. The Taurus incorporates the motors and avionics of the Pegasus with a larger first stage (the government-provided Peacekeeper missile). Taurus was funded by DARPA to supporting a rapid-response launch capability.

Tools Developed

The launch vehicle module `launch_vehicle.m` functions in two capacities. When the function is called with no arguments, it returns the complete database of launch

vehicles. The second capacity takes the apogee radius of the parking orbit and the payload mass as inputs, and outputs the cheapest capable launch vehicle and ‘actual’ apogee and perigee radii. It is assumed that launch occurs at Cape Canaveral and that the launch vehicle only needs to get the payload to a parking orbit, which may be either circular or eccentric. Reliability, procurement lead time, and other factors that may influence the selection of a launch vehicle are not incorporated into this analysis.

Inputs to the launch vehicle module

Quantity	Units	Description
radius	km	Minimum required apogee radius for the parking orbit.
mass	kg	Spacecraft mass.

Outputs of the launch vehicle module

Quantity	Units	Description
lv_index		Index into the launch vehicle database of the optimum vehicle
apogee	km	Actual achievable parking orbit apogee.
perigee	km	Parking orbit perigee.

launch_vehicle.m

```
function varargout=launch_vehicle(varargin)
%LAUNCH_VEHICLE   Outputs appropriate launch vehicle for a mission
%
% [LV_DATABASE] = LAUNCH_VEHICLE
% Outputs a structure containing information on each launch vehicle.
%
% [LV_INDEX,APOGEE,PERIGEE] = LAUNCH_VEHICLE(RADIUS,MASS)
% Outputs the index into the LV_DATABASE of the launch vehicle that
% can most cost-effectively deliver a satellite of MASS kilograms
% into a circular parking orbit of RADIUS kilometers.  If no launch
% vehicle can meet the demands, LV_INDEX is set to zero.  APOGEE and
% PERIGEE are the actual values achievable for the specified MASS.

% Assumptions
% 1. Best launch vehicle is determined as a function of cost
% 2. Launch vehicle only needs to get payload to parking orbit
% 3. Reliability (i.e. 98% successful Space Shuttle) is not
%    incorporated into the analysis
% 4. Lead time is not factored into the module
% 5. Launch occurs from Cape Canaveral (28.5 inclination)

% Input launch vehicle characteristics, including Excel performance equations
% 1999 AIAA International Reference Guide to Space Launch Systems
% Inflation factor for 1999 launch vehicle estimates (source: SMAD, Chapter 20)
% min and max are range of validity for performance equation
% 0 perigee means circular orbit
i=0;
i=i+1;lvdata(i) = struct('name','Atlas IIA',...
                        'company','Lockheed Martin',...
                        'cost',91,...
                        'x2',NaN,'x',NaN,'intercept',NaN,...
```

```

        'c_log',-847.55,'c_int',12041,...
        'perigee',185,...
        'min',0,'max',36000);

i=i+1;lvdata(i) = struct('name','Atlas IIAS',...
        'company','Lockheed Martin',...
        'cost',112,...
        'x2',NaN,'x',NaN,'intercept',NaN,...
        'c_log',-968.15,'c_int',13978,...
        'perigee',185,...
        'min',0,'max',36000);

i=i+1;lvdata(i) = struct('name','Atlas IIIA',...
        'company','Lockheed Martin',...
        'cost',112,...
        'x2',NaN,'x',NaN,'intercept',NaN,...
        'c_log',-969.02,'c_int',14319,...
        'perigee',185,...
        'min',0,'max',36000);

i=i+1;lvdata(i) = struct('name','Atlas IIIB',...
        'company','Lockheed Martin',...
        'cost',112,...
        'x2',NaN,'x',NaN,'intercept',NaN,...
        'c_log',-1248.6,'c_int',17808,...
        'perigee',185,...
        'min',0,'max',36000);

i=i+1;lvdata(i) = struct('name','Delta II 7320',...
        'company','Boeing',...
        'cost',59,... )
        'x2',0,'x',-1.6,'intercept',4800,...
        'c_log',NaN,'c_int',NaN,...
        'perigee',0,...
        'min',0,'max',2000);

i=i+1;lvdata(i) = struct('name','Delta II 7420',...
        'company','Boeing',...
        'cost',59,...
        'x2',0,'x',-1.4815,'intercept',4888.9,...
        'c_log',NaN,'c_int',NaN,...
        'perigee',0,...
        'min',0,'max',2000);

i=i+1;lvdata(i) = struct('name','Delta II 7920',...
        'company','Boeing',...
        'cost',64,...
        'x2',0,'x',-1.0526,'intercept',5789.5,...
        'c_log',NaN,'c_int',NaN,...
        'perigee',0,...
        'min',0,'max',2000);

i=i+1;lvdata(i) = struct('name','Delta III',...
        'company','Boeing',...
        'cost',96,...
        'x2',NaN,'x',NaN,'intercept',NaN,...
        'c_log',-848.91,'c_int',12723,...
        'perigee',0,...
        'min',0,'max',36000);

i=i+1;lvdata(i) = struct('name','Pegasus XL',...
        'company','Orbital Sciences',...
        'cost',16,...

```



```

        'x2',0,'x',-4,'intercept',2000,...
        'c_log',NaN,'c_int',NaN,...
        'perigee',0,...
        'min',0,'max',2000);
i=i+1;lvdata(i) = struct('name','Space Shuttle',...
        'company','NASA',...
        'cost',320,...
        'x2',-.025,'x',-6,'intercept',25300,...
        'c_log',NaN,'c_int',NaN,...
        'perigee',0,...
        'min',0,'max',600);

i=i+1;lvdata(i) = struct('name','Taurus 2110',...
        'company','Orbital Sciences',...
        'cost',21,...
        'x2',0,'x',-2.1622,'intercept',3243.2,...
        'c_log',NaN,'c_int',NaN,...
        'perigee',0,...
        'min',0,'max',2000);

% determine which output type given the arguments
if (nargin==0 & nargout<=1)
    % return the LV database
    varargout{1} = lvdata;
    return
elseif (nargin==2 & nargout==3)
    % eventually return the LV index
    A_po = varargin{1}-6378.136; % change radius to altitude
    mass = varargin{2};
else
    % something is wrong
    error('Invalid number of input or output arguments');
end

% Get matrices
cost = cat(2,lvdata.cost);
x2 = cat(2,lvdata.x2);
x = cat(2,lvdata.x);
c_log = cat(2,lvdata.c_log);
c_int = cat(2,lvdata.c_int);
intercept = cat(2,lvdata.intercept);
perigee = cat(2,lvdata.perigee);
amin = cat(2,lvdata.min);
amax = cat(2,lvdata.max);

% Eliminate launch vehicles unable to launch to desired apogee radius
bound_indices = find(A_po<=amax & A_po>=amin);

% Compute maximum payload capability for a given altitude
pcap_a = x2.*A_po.^2 + x.*A_po + intercept;
pcap_b = c_log.*log(A_po)+c_int;
for i=1:length(lvdata)
    if ~isnan(pcap_a(i))
        pcap(i) = pcap_a(i);
    elseif ~isnan(pcap_b(i))
        pcap(i) = pcap_b(i);
    else
        pcap(i) = 0;
    end
end

% Eliminate launch vehicles unable to meet mass requirement
cap_indices = find(pcap>=mass);

```

```

% intersection of the sets
indices = intersect(cap_indices, bound_indices);

% find the minimum cost vehicle
[cost, index] = min(cost(indices));
index = indices(index);

if isempty(index)
    varargout{1} = NaN;
    varargout{2} = 0;
    varargout{3} = 0;
    return
end

if x2(index)==0
    apoapsis = (mass-intercept(index))/x(index);
elseif isnan(x2(index))
    apoapsis = min([exp((mass-c_int(index))/c_log(index)) amax(index)]);
else
    apoapsis = (-x2(index)-sqrt(x(index)^2-4*x2(index)*...
        (intercept(index)-mass)))/(2*x2(index));
end

if perigee==0
    periapsis=apoapsis;
else
    periapsis=perigee(index);
end

varargout{1} = index;
varargout{2} = apoapsis;
varargout{3} = periapsis;

```

Useful References

Atlas II Family. *International Launch Services*. <http://www.ilslaunch.com/atlas/atlasii/> (10 November 2003).

Atlas III Summary. *Space and Tech Database of Expendable Launch Vehicles*. http://www.spaceandtech.com/spacedata/elvs/atlas3_sum.shtml (13 November 2003).

Delta II Payload Planners Guide. *The Boeing Company*. January 2003. http://www.boeing.com/defense-space/space/delta/docs/delta_II_ppg_update_january_2003.pdf (10 November 2003).

Delta III Payload Planners Guide. *The Boeing Company*. October 2000. http://www.boeing.com/defense-space/space/delta/docs/DELTA_III_PPG_2000.PDF (10 November 2003).

Isakowitz, Steven J. *International Reference Guide to Space Launch Systems*. Reston: AIAA, September 1999.

Orbital Launch Vehicle Index. *Encyclopedia Astronautica*. <http://astronautix.com/lvs/orbindex.htm> (9 November 2003).

3. Orbital Mechanics

Orbital mechanics, also called flight mechanics, is the study of the motions of artificial satellites and space vehicles moving under the influence of forces such as gravity, atmospheric drag, thrust, etc. Orbital mechanics is a modern offshoot of celestial mechanics which is the study of the motions of natural celestial bodies such as the moon and planets. The root of orbital mechanics can be traced back to the 17th century when mathematician Isaac Newton (1642-1727) put forward his laws of motion and formulated his law of universal gravitation. The engineering applications of orbital mechanics include ascent trajectories, reentry and landing, rendezvous computations, and lunar and interplanetary trajectories. (Note: this introduction, the ‘Orbital Elements’ and ‘Types of Orbits’ sections are taken from <http://users.commkey.net/Braeunig/space/orbmech.htm#elements>)

Learning Objectives

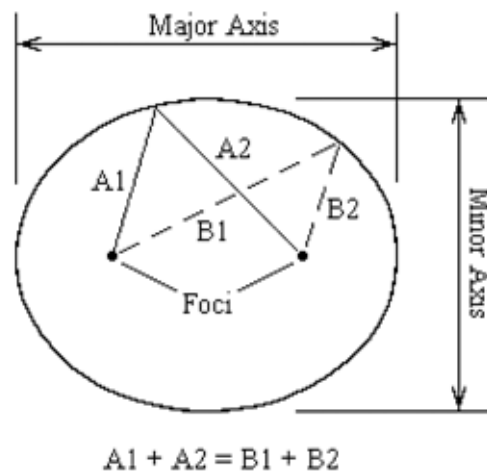
- Understand orbital elements, including: Semi-Major Axis, Eccentricity, Inclination, Argument of Periapsis, Time of Periapsis Passage, Longitude of Ascending Node
- Study different types of orbits
- Newton’s laws of motion and universal gravitation
- Uniform circular motion
- Motions of planets and satellites

Reflections on Learning Experience

Having had no instruction in orbits since touching on the basic laws in Unified, I had a lot of catching up to do for this section of the class. Sedwick’s lecture was fast-paced and I got lost in the math. However, two of my problem sets included orbits as a subsystem and these are where I was able to learn the basic principles and apply them in the modules. Learning how to integrate STK and MATLAB is a valuable skill I will likely use in the future.

Orbital Elements

An orbiting satellite follows an oval shaped path known as an ellipse with the body being orbited, called the primary, located at one of two points called foci. An ellipse is defined to be a curve with the following property: for each point on an ellipse, the sum of its distances from two fixed points, called foci, is constant (see figure to right). The longest and shortest lines that can be drawn through the center of an ellipse are called the major axis and minor axis, respectively. The semi-major axis is one-half of the major axis and

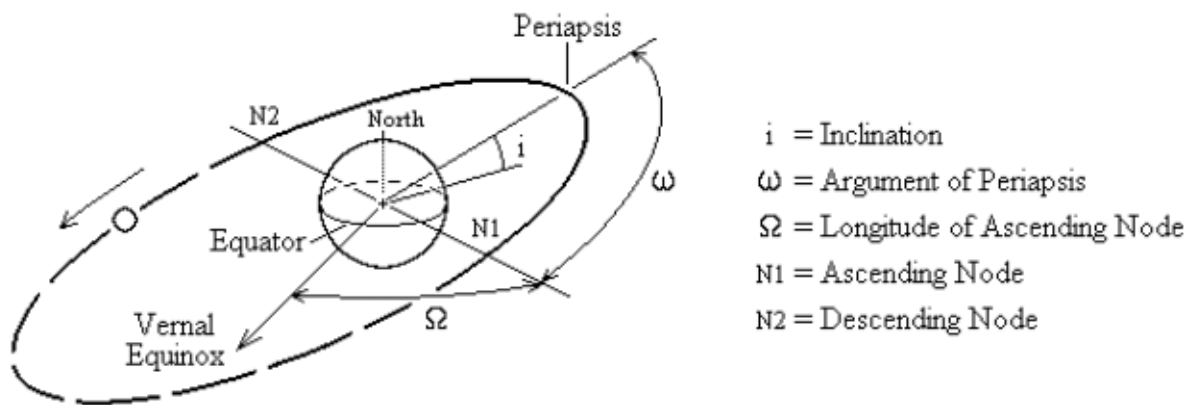


represents a satellite's mean distance from its primary. Eccentricity is the distance between the foci divided by the length of the major axis and is a number between zero and one. An eccentricity of zero indicates a circle.

Inclination is the angular distance between a satellite's orbital plane and the equator of its primary (or the ecliptic plane in the case of heliocentric, or sun centered, orbits). An inclination of zero degrees indicates an orbit about the primary's equator in the same direction as the primary's rotation, a direction called prograde (or direct). An inclination of 90 degrees indicates a polar orbit. An inclination of 180 degrees indicates a retrograde equatorial orbit. A retrograde orbit is one in which a satellite moves in a direction opposite to the rotation of its primary.

Periapsis is the point in an orbit closest to the primary. The opposite of periapsis, the farthest point in an orbit, is called apoapsis. Periapsis and apoapsis are usually modified to apply to the body being orbited, such as perihelion and aphelion for the Sun, perigee and apogee for Earth, perijove and apojoove for Jupiter, perilune and apolune for the Moon, etc. The argument of periapsis is the angular distance between the ascending node and the point of periapsis (see figure below). The time of periapsis passage is the time in which a satellite moves through its point of periapsis.

Nodes are the points where an orbit crosses a plane, such as a satellite crossing the Earth's equatorial plane. If the satellite crosses the plane going from south to north, the node is the ascending node; if moving from north to south, it is the descending node. The longitude of the ascending node is the node's celestial longitude. Celestial longitude is analogous to longitude on Earth and is measured in degrees counter-clockwise from zero with zero longitude being in the direction of the vernal equinox.



In general, three observations of an object in orbit are required to calculate the six orbital elements. Two other quantities often used to describe orbits are period and true anomaly. Period is the length of time required for a satellite to complete one orbit. True anomaly is the angular distance of a point in an orbit past the point of periapsis, measured in degrees.

Types of Orbits

For a spacecraft to achieve earth orbit, it must be launched to an elevation above the Earth's atmosphere and accelerated to orbital velocity. The most energy efficient orbit, that is one that requires the least amount of propellant, is a direct low inclination orbit. To achieve such an orbit, a spacecraft is launched in an eastward direction from a site near the Earth's equator. The advantage being that the rotational speed of the Earth contributes to the spacecraft's final orbital speed. At the United States' launch site in Cape Canaveral (28.5 degrees north latitude) a due east launch results in a "free ride" of 915 mph (1,470 kph). Launching a spacecraft in a direction other than east, or from a site far from the equator, results in an orbit of higher inclination. High inclination orbits are less able to take advantage of the initial speed provided by the Earth's rotation, thus the launch vehicle must provide a greater part, or all, of the energy required to attain orbital velocity. Although high inclination orbits are less energy efficient, they do have advantages over equatorial orbits for certain applications. Below we describe several types of orbits and the advantages of each:

Geosynchronous orbits, also called *geostationary orbits* (GEO), are circular, low inclination orbits around the Earth having a period of 24 hours. A spacecraft in a geosynchronous orbit appears to hang motionless above one position on the Earth's surface. For this reason, they are ideal for some types of communication and meteorological satellites. To attain geosynchronous orbit, a spacecraft is first launched into an elliptical orbit with an apogee of 22,240 miles (35,790 km) called a *geostationary transfer orbit* (GTO). The orbit is then circularized by firing the spacecraft's engine at apogee.

Polar orbits (PO) are orbits with an inclination of 90 degrees. Polar orbits are useful for satellites that carry out mapping and/or surveillance operations because as the planet rotates the spacecraft has access to virtually every point on the planet's surface.

Walking orbits: An orbiting satellite is subjected to a great many gravitational influences. First, planets are not perfectly spherical and they have slightly uneven mass distribution. These fluctuations have an effect on a spacecraft's trajectory. Also, the sun, moon, and planets contribute a gravitational influence on an orbiting satellite. With proper planning it is possible to design an orbit which takes advantage of these influences to induce a precession in the satellite's orbital plane. The resulting orbit is called a *walking orbit*, or precessing orbit.

Sun synchronous orbits (SSO) are walking orbits whose orbital plane precesses with the same period as the planet's solar orbit period. In such an orbit, a satellite crosses periapsis at about the same local time every orbit. This is useful if a satellite is carrying instruments which depend on a certain angle of solar illumination on the planet's surface. In order to maintain an exact synchronous timing, it may be necessary to conduct occasional propulsive maneuvers to adjust the orbit.

Hohmann transfer orbits are interplanetary trajectories whose advantage is that they consume the least possible amount of propellant. A Hohmann transfer orbit to an outer planet, such as Mars, is achieved by launching a spacecraft and accelerating it in the

direction of Earth's revolution around the sun until it breaks free of the Earth's gravity and reaches a velocity which places it in a sun orbit with an aphelion equal to the orbit of the outer planet. Upon reaching its destination, the spacecraft must decelerate so that the planet's gravity can capture it into a planetary orbit.

To send a spacecraft to an inner planet, such as Venus, the spacecraft is launched and accelerated in the direction opposite of Earth's revolution around the sun (i.e. decelerated) until it achieves a sun orbit with a perihelion equal to the orbit of the inner planet. It should be noted that the spacecraft continues to move in the same direction as Earth, only more slowly.

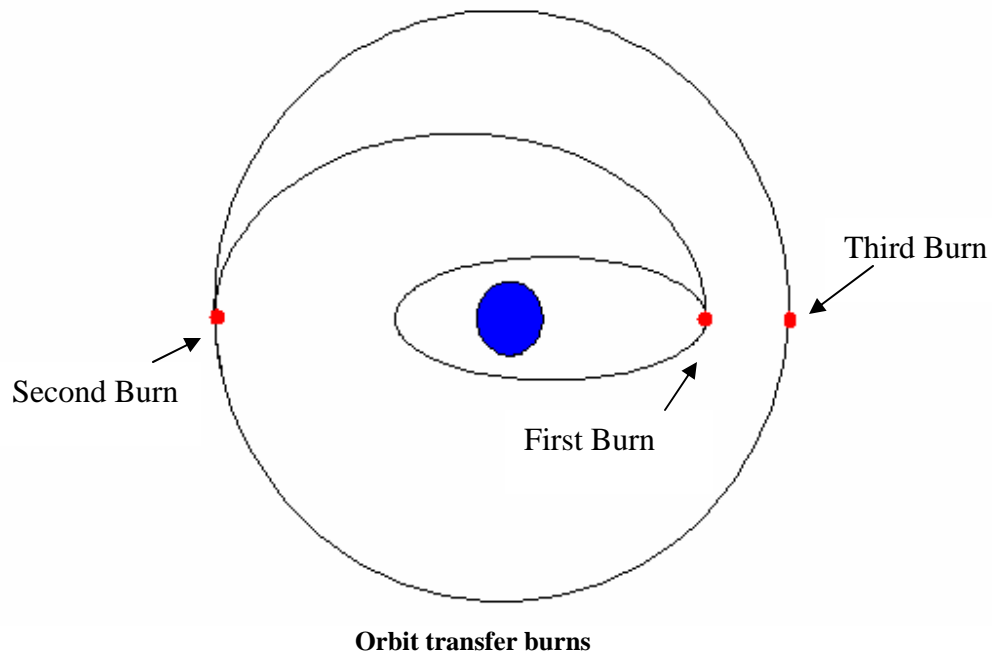
To reach a planet requires that the spacecraft be inserted into an interplanetary trajectory at the correct time so that the spacecraft arrives at the planet's orbit when the planet will be at the point where the spacecraft will intercept it. This task is comparable to a quarterback "leading" his receiver so that the football and receiver arrive at the same point at the same time. The interval of time in which a spacecraft must be launched in order to complete its mission is called a *launch window*.

Tools Developed

The parking orbit into which a spacecraft is initially placed by a launch vehicle is often elliptical. The orbit is circularized to the desired radius, and plane changes are performed, using the onboard propulsion system. The orbit module `calculate_delta_v.m` takes as inputs the parking orbit apogee, perigee, and inclination, and the destination orbit radius and inclination, and determines the change in velocity needed to achieve the final orbit.

The first burn takes place at the apogee of the parking orbit, and is used to raise the perigee. The required change in velocity for this first burn is determined by the following equations, where r_a and r_p are the apogee and perigee radii, respectively, a is the semi-major axis, e is the eccentricity, p is the semi-latus rectum and h is the angular momentum. The constant $\mu=GM_{Earth}$ is the gravitational parameter of Earth.

$$a = \frac{r_a + r_p}{2}$$
$$e = (r_a / a) - 1$$
$$p = a(1 - e^2)$$
$$h = \sqrt{p\mu}$$



The specific energy ε is calculated for both the initial and final orbits. This allows for the calculation of initial and final velocities at each orbit. The change in velocity is equal to the difference between the final and initial velocities. These calculations are then repeated for the second burn, which brings the orbit from its new altitude at perigee to its new altitude at apogee, thereby circularizing the orbit.

$$\varepsilon = \frac{h^2}{2r_a^2} - \frac{\mu}{r}$$

$$v = \sqrt{2\left(\varepsilon + \frac{\mu}{r}\right)}$$

The final burn changes the plane, if needed, through a simple plane change using the following equations, where i_f and i_i are the final and initial inclinations, respectively.

$$\theta = i_f - i_i$$

$$\Delta v = 2v_i \sin(\theta/2)$$

After the magnitudes of the three changes in velocity are determined, the orbit module calculates the period of each orbit using the following equation.

$$P = 2\pi\sqrt{(a^3 / \mu)}$$

The orbit module outputs the changes in velocity required for each of the three burns, and the orbital periods for each of the three orbits.

Inputs to the orbit module.

Quantity	Units	Description
apo_i	km	Parking orbit (initial) apogee.
peri_i	km	Parking orbit (initial) perigee.
inc_i	rad	Parking orbit (initial) inclination.
rad_f	km	Destination orbit (final) radius.
inc_f	rad	Destination orbit (final) inclination.

Outputs from the orbit module.

Quantity	Description
dvinfos	A data structure containing information on the required ΔV s and the orbital periods of the parking, transfer, and final orbits.

calculate_delta_v.m

```
function [output] = calculate_delta_v(parking_apogee, parking_perigee,...
    parking_inclination, final_radius, final_inclination)

%CALCULATE_DELTA_V determines the delta V required for an orbit transfer
%
% [DVINFO] = CALCULATE_DELTA_V(APO_I, PERI_I, INC_I, RAD_F, INC_F)
% Given the apogee, perigee, and inclination of the initial orbit
% (APO_I, PERI_I, INC_I) and the circular radius RAD_F and inclination
% INC_F of the final orbit, determines the delta V required for each
% of three burns and the orbital periods of the initial, transfer, and
% final orbits. These data are contained in the structure DVINFO.

% Date: November 10, 2003

% This module returns the change in velocity needed to perform an orbit
% transfer from an initial parking orbit to an intermediate orbit and a
% circular final orbit, including, if needed, a plane change.
% The time of flight is also returned for the three burns is also output.

mu = 3.98600e5; % Earth's gravitational constant in: km^3 / s^2
third_burn_delta_v = 0;

semimajor_axis = (parking_apogee + parking_perigee) / 2;
eccentricity = (parking_apogee / semimajor_axis) - 1;
semilatus_rectum = semimajor_axis * (1 - eccentricity ^ 2);
angular_momentum = sqrt(semilatus_rectum * mu);

intermediate_semimajor_axis = (final_radius + parking_apogee) / 2;
intermediate_eccentricity = (final_radius / intermediate_semimajor_axis) - 1;
intermediate_semilatus_rectum = intermediate_semimajor_axis *...
    (1 - intermediate_eccentricity ^ 2);
intermediate_angular_momentum = sqrt(intermediate_semilatus_rectum * mu);

specific_energy_parking_orbit = ((angular_momentum^2) / (2 * parking_apogee^2)) ...
    - (mu / parking_apogee);
```



```

specific_energy_intermediate_orbit = ((intermediate_angular_momentum ^ 2) / ...
(2 * parking_apogee ^ 2)) - (mu / parking_apogee);
velocity_parking_orbit = sqrt(2 * (specific_energy_parking_orbit + ...
(mu / parking_apogee)));
velocity_intermediate_orbit = sqrt(2 * (specific_energy_intermediate_orbit + ...
(mu / parking_apogee)));

first_burn_delta_v = velocity_intermediate_orbit - velocity_parking_orbit;

specific_energy_intermediate_orbit = ((intermediate_angular_momentum ^ 2) / ...
(2 * final_radius ^ 2)) - (mu / final_radius);
specific_energy_final_orbit = -(mu / (2 * final_radius));
velocity_intermediate_orbit = sqrt(2 * (specific_energy_intermediate_orbit + ...
(mu / final_radius)));
velocity_final_orbit = sqrt(2 * (specific_energy_final_orbit + (mu / final_radius)));

second_burn_delta_v = velocity_final_orbit - velocity_intermediate_orbit;

if(~(parking_inclination == final_inclination))
    inclination_angle_change = (final_inclination - parking_inclination) * (pi / 180);
    third_burn_delta_v = 2 * velocity_final_orbit * sin(inclination_angle_change / 2);
end

period_parking_orbit = 2 * pi * sqrt((semimajor_axis ^ 3) / mu);
period_intermediate_orbit = 2 * pi * sqrt((intermediate_semimajor_axis ^ 3) / mu);
period_final_orbit = 2 * pi * sqrt((final_radius ^ 3) / mu);

output.deltaV = [first_burn_delta_v second_burn_delta_v third_burn_delta_v];
output.period_parking = period_parking_orbit;
output.period_intermediate = period_intermediate_orbit;
output.period_final = period_final_orbit;

```

Useful References

Bate, Roger R., Mueller, Donald D., and White, Jerry E., Fundamentals of Astrodynamics, New York, Dover Publications, Inc., 1971.

Braeunig, David A. Rocket and Space Technology.
<http://users.commkey.net/Braeunig/space/orbmech.htm#types> (21 November 2003).

Orbital Mechanics Educational Network. <http://www.omenet.co.uk/omen16.htm>

4. Power Systems

The power system is responsible for providing, storing, distributing, and regulating electricity throughout the spacecraft. Mission life, orbital parameters, spacecraft configuration, and average and peak electrical power requirements all effect design. The traditional source of power for Earth-orbiting spacecraft is solar photovoltaics. Solar thermal dynamics are also a potential source of power for satellites in LEO. Sources of power for interplanetary spacecraft include radioisotopes, nuclear reactors, and fuel cells.

Learning Objectives

Study of the following issues are critical to understanding the power system:

- Strengths and weaknesses of different solar cell types
- Relationship between the spacecraft's orbital parameters (i.e. altitude) and the allowable Depth-of-Discharge of solar arrays
- Characteristics of fuel cells
- Issues associated with RTG's
- Solar cell operating characteristics and temperature effects

Reflections on Learning Experience

SMAD has a well-documented approach (section 11.4) to designing solar arrays and the battery storage system. Given the readily-available data in this section and the dependency of the power subsystem on other subsystems of the spacecraft, it is a good selection for conducting trades in the problem sets. In the second problem set I dealt with the degradation of solar arrays due to radiation in the space environment. In the fourth problem set I coded modules which sized the solar array and battery storage system.

In the first five problem sets my group always dealt with LEO satellites, offering little chance to explore radioisotopes, nuclear reactors, and fuel cells. I am hoping to explore these alternative power system in our sixth problem set which deals with a manned mission to Mars

Tools Developed

The power production module coded on the fourth problem set outputs the mass of the solar array, the area the solar array, and the solar cell type (either Silicon, Thin Sheet Amorphous Si, Gallium Arsenide, Indium Phosphide, or Multijunction GaInP/GaAs) that will minimize the solar array area.. (See MATLAB function `Power_Optimizing` to follow) The inputs to the module are mission life, power required (not including data transmission needs), power required for data transmission needs, time to downlink data, average time of eclipse, average time of sunlight, and inclination of the orbit.

Many assumptions were made in coding the power production module. Energy transfer efficiencies ($X_e = .65$ during eclipse and $X_d = .85$ during sunlight) were taken as constants from *SMAD*. The input power required for data transmission was assumed to include energy transfer efficiency losses ($X_{dl} = 1$). Also, the non-transmitting power required was assumed to be constant in sunlight and eclipse ($P_e = P_d$). Calculating a dynamic value of inherent degradation—entailing analysis of the solar cell design and assembly process, temperature of array, and shadowing of cells— falls outside of the scope of our problem so a constant value of inherent degradation (.77) was taken from *SMAD*.

The first step in the power production module is to compute the power the solar array is required to collect during sunlight, P_{sa} . T_e , T_d , and T_{dl} are time in eclipse, time in sunlight, and time to downlink respectively.

$$P_{sa} = \frac{\left(\frac{P_e T_e}{X_e} + \frac{P_d T_d}{X_d} + \frac{P_{dl} T_{dl}}{X_{dl}} \right)}{T_d}$$

The second step is to compute beginning-of-life power, P_{bol} , per unit area. The solar illumination intensity, S_i , is reduced by solar cell efficiency, degradation, and angle-of-incidence factors. Each solar cell efficiency is fed into the equation using array X_{ss} . The orbit's inclination, inc , is taken as an input. The worst-case sun angle between equatorial and ecliptic planes is taken as 23.5 degrees.

$$P_{bol} = S_i \cdot X_{ss} \cdot I_d \cdot \cos\left(\frac{23.5\pi}{180} + inc\right)$$

The third step is to compute end-of-life power collection capability, P_{eol} , of the solar array. For the duration of the mission each solar cell experiences life degradation, D_{ss} , due to thermal cycling in and out of eclipses and other factors. Mission life, L_m , is an obvious input.

$$P_{eol} = P_{bol}(1 - D_{ss})^{L_m}$$

The fourth and final step of the power production module is to determine the solar array area, A_{sa} , that meets the power collection requirements, P_{sa} . We must design for the end-of-life power collection capability, P_{eol} .

$$A_{sa} = \frac{P_{sa}}{P_{eol}}$$

A_{sa} and P_{eol} vary for each solar cell variety. The module selects a solar cell type that minimizes array area and proceeds to compute mass of the power subsystem using equation 10-13 in *SMAD*.

Power_optimizing.m

```

% Matthew Richards
function [cell_type, M_sa, area] = Power_Optimizing(L_m, P_av, P_dl, T_dl, T_e, T_d, inc)
% This function selects the optimal solar cell type as a function of area

% INPUTS
% L_m: mission life (years)
% P_av: power requirement, not including data transmission needs (W)
% P_dl: power required for data transmission needs (W)
% T_dl: time to downlink data (minutes)
% T_e: average time of eclipse (minutes)
% T_d: average time not in eclipse (minutes)
% inc: inclination of the orbit (rad)

% OUTPUTS
% cell_type: name of solar cell
% M_sa: mass of solar array (kg)
% area: area of solar array (m^2)

disp('Finding Solar Array...');

% Defining problem variables (source: SMAD)
X_e = .65;           %energy transfer efficiency during eclipse, direct energy transfer
X_d = .85;           %energy transfer efficiency during daylight, direct energy transfer
S_i = 1367;         %solar illumination intensity, W/m^2
X_si = .148;        %solar cell production efficiency, Silicon
X_th = .050;        %solar cell production efficiency, Thin Sheet Amorphous Si
X_ga = .185;        %solar cell production efficiency, Gallium Arsenide
X_ip = .180;        %solar cell production efficiency, Indium Phosphide
X_mj = .220;        %solar cell production efficiency, Multijunction GaInP/GaAs
I_d = .77;          %inherent degradation
theta = 23.5 * (pi/180) + inc; % worst-case sun angle, radians

% Compute power collected by solar array during daylight
P_sa = (P_av * T_e / X_e + P_av * T_d / X_d + P_dl * T_dl) / T_d;

% Vector of solar cell efficiencies
X = [X_si X_th X_ga X_ip X_mj];

% Compute power output for various solar cell types (W/m^2)
P_o = S_i .* X;

% Compute beginning-of-life power (W) per unit area for various solar cell types
P_bol = (cos(theta) * I_d) .* P_o;

% Compute actual lifetime degradation for various solar cell types
A_d = [.0375 .0375 .0275 .0375 .0375];
L_d = (1 - A_d) .^ L_m;

% Compute end-of-life power (W) per unit area for various solar cell types
P_eol = P_bol .* L_d;

% Compute solar array area (m^2) required to support input power requirement
A_sa = P_sa ./ P_eol;

% List best solar cell type (smallest cell area)
fprintf('\n');
disp('Best solar cell type, minimizing for area:');
Min = min(A_sa);
if Min == A_sa(1);

```

```

    cell_type = 'Silicon'
elseif Min == A_sa(2);
    cell_type = 'Thin Sheet Amorphous Si'
elseif Min == A_sa(3);
    cell_type = 'Gallium Arsenide'
elseif Min == A_sa(4);
    cell_type = 'Indium Phosphide'
elseif Min == A_sa(5);
    cell_type = 'Multijunction GaInP/GaAs'
end

% Compute mass (kg) of solar array
M_sa = .04 * P_sa;

% Area of chosen solar array
area = Min;

disp('Solar array mass (kg): ');
disp(M_sa);
disp('Solar array area (m^2): ');
disp(area);

```

The power storage module (also coded on the fourth problem set) outputs the optimal secondary battery type (choosing among Nickel-Cadium and three Nickel-Hydrogen varieties) as a function of mass. (See MATLAB function `Storage_Optimizing` to follow.) Inputs to the function include the mission lifetime, number of times the satellite goes in and out of eclipse, power required (not including data transmission needs), power required for data transmission needs, time to downlink data, and average time of eclipse.

One assumption made in the module is that transmission efficiency between the battery and the load, n , is 90%. Also, in our module we are designing to meet a battery capacity requirement. Whether or not to have additional batteries to provide redundancy is a trade which may become the subject of future work.

The depth-of-discharge (DOD), or battery capacity removed during a discharge period such as eclipse, is the first calculation performed in the power storage module. The DOD is inversely related to the number of charge and discharge cycles undergone by a battery throughout its life. The number of cycles is determined by our spacecraft's orbital parameters (i.e. altitude). Figure 11-11 from *SMAD* is used in our function to determine DOD for our Nickel-Cadmium and Nickel Hydrogen battery types. To simplify the calculation, cycle life is broken into seven groups for each secondary battery (less than 1000 cycles, between 1000 and 2000 cycles, etc.). The DOD for a given range of values is taken as the average of the maximum DOD and minimum DOD of that range.

The second step of the power storage module determines power required during eclipse, P_e . Designing for the batteries for the worst-case scenario, we assume the data transmission down to Earth occurs during eclipse.

$$P_e = \frac{(P_{av}T_e + P_{dt}T_{dt})}{T_e}$$

The third step determines the battery capacity, C_r , required for the spacecraft. Depth-of-discharge, X_{dod} , is input to this calculation as an array representing the four battery types.

$$C_r = \frac{P_e T_e}{60nX_{dod}}$$

The fourth and final step of the function selects among the battery types to minimize mass, M . Specific energy density, X_{ed} , is fed into this function as an array.

$$M = \frac{C_r}{X_{ed}}$$

Storage_optimizing.m

```
% Matthew Richards
function [battery_name, mass] = Storage_Optimizing(L_m, cyc, P_av, P_dl, T_dl, T_e)
% This function selects the optimal battery as a function of mass

% INPUTS
% L_m: life of mission (years)
% cyc: cycle life (number of times satellite goes in and out of sun)
% P_av: average power requirement, not including for data transmission (W)
% P_dl: power for data transmission (W)
% T_dl: time of each data transmission (minutes)
% T_e: time of eclipse (minutes)

% OUTPUTS
% battery_name
% mass: mass of battery (kg)

disp('Finding Battery...');
% Defining problem variables (source: SMAD)
X_nc = 30; %Nickel-Cadium specefic energy (W*hr/kg)
X_nhi = 43; %Nickel-Hydrogen (individual pressure) specefic energy (W*hr/kg)
X_nhc = 56; %Nickel-Hydrogen (common pressure) specefic energy (W*hr/kg)
X_nhs = 57; %Nickel-Hydrogen (single pressure) specefic energy (W*hr/kg)
X_li = 110; %Lithium-Ion specefic energy (W*hr/kg)
X_ss = 210; %Sodium-Sulfur specefic energy (W*hr/kg)
n = 0.90; %Transmission effeciency between the battery and the load

% Vector of specific energy densities
X = [X_nc X_nhi X_nhc X_nhs];

% Determine Depth-of-Discharge
if cyc < 1000
    DoD_nc = .65;
    DoD_nhi = .90;
    DoD_nhc = .90;
    DoD_nhs = .90;
elseif cyc >= 1000 & cyc < 2000
    DoD_nc = .55;
    DoD_nhi = .75;
    DoD_nhc = .75;
    DoD_nhs = .75;
elseif cyc >= 2000 & cyc < 4000
    DoD_nc = .45;
    DoD_nhi = .65;
```

```

    DoD_nhc = .65;
    DoD_nhs = .65;
elseif cyc >= 4000 & cyc < 10000
    DoD_nc = .35;
    DoD_nhi = .55;
    DoD_nhc = .55;
    DoD_nhs = .55;
elseif cyc >= 10000 & cyc < 20000
    DoD_nc = .30;
    DoD_nhi = .50;
    DoD_nhc = .50;
    DoD_nhs = .50;
elseif cyc >= 20000 & cyc < 40000
    DoD_nc = .20;
    DoD_nhi = .40;
    DoD_nhc = .40;
    DoD_nhs = .40;
elseif cyc > 40000
    DoD_nc = .15;
    DoD_nhi = .35;
    DoD_nhc = .35;
    DoD_nhs = .35;
end

% Determine Power Required During Eclipse, design for worst-case: assume downlink during eclipse
P_e = (P_av * T_e + P_dl * T_dl) / T_e;

% Determine Battery Capacity (W*hr)
DoD = [DoD_nc DoD_nhi DoD_nhc DoD_nhs];
C_r = (P_e * T_e/60) .* ((n * DoD).^(-1));

% Determine Mass of Storage System
M = C_r ./ X;

% Display Lightest Power Storage System
disp('Best secondary battery type (lightest):');
Min = min(M);
if Min == M(1);
    battery_name = 'Nickel-Cadium'
elseif Min == M(2);
    battery_name = 'Nickel-Hydrogen (individual pressure vessel)'
elseif Min == M(3);
    battery_name = 'Nickel-Hydrogen (common pressure vessel)'
elseif Min == M(4);
    battery_name = 'Nickel-Hydrogen (single pressure vessel)'
end

% Display Mass of Lightest Power Storage System
disp('Battery Mass (kg):');
Min = min(M);
disp(Min)
mass = Min;

```

Useful References

Space Mission Analysis and Design (SMAD), Wertz & Larson, Third edition, Microcosm Press, 1999.

Fundamentals of Space Systems, Pisacane and Moore, Oxford, 1994.

5. Propulsion Systems

Space propulsion systems are typically responsible for three tasks: lifting the launch vehicle and payload into a parking orbit in LEO, orbital transfer maneuvers, and thrust for attitude control and orbit corrections.

Learning Objectives

SMAD identifies five areas of study for space propulsion systems:

- Propulsion subsystem selection and sizing
- Basics of rocket propulsion
- Types of rockets
- Component selection and sizing
- Staging

Reflections on Learning Experience

I enjoyed this lecture and found the chapter in *SMAD* (17) to be strong in both covering the fundamentals of rocket propulsion and in providing tables with metrics on current propulsion systems.

Rocket Propellants

This section taken from: <http://users.comkey.net/Braeunig/space/propel.htm>

Propellant is the chemical mixture burned to produce thrust in rockets and consists of a fuel and an oxidizer. A *fuel* is a substance which burns when combined with oxygen producing gas for propulsion. An *oxidizer* is an agent that releases oxygen for combination with a fuel. Propellants are classified according to their state - liquid, solid, or hybrid.

The gauge for rating the efficiency of rocket propellants is *specific impulse*, stated in seconds. Specific impulse indicates how many pounds (or kilograms) of thrust are obtained by the consumption of one pound (or kilogram) of propellant in one second. Specific impulse is characteristic of the type of propellant, however, its exact value will vary to some extent with the operating conditions and design of the rocket engine.

Liquid Propellants

In a liquid propellant rocket, the fuel and oxidizer are stored in separate tanks, and are fed through a system of pipes, valves, and turbopumps to a combustion chamber where they are combined and burned to produce thrust. Liquid propellant engines are more complex than their solid propellant counterparts, however, they offer several advantages. By controlling the flow of propellant to the combustion chamber, the engine can be throttled, stopped, or restarted.

A good liquid propellant is one with a high specific impulse or, stated another way, one with a high speed of exhaust gas ejection. This implies a high combustion temperature and exhaust gases with small molecular weights. However, there is another important factor which must be taken into consideration: the density of the propellant. Using low density propellants means that larger storage tanks will be required, thus increasing the mass of the launch vehicle. Storage temperature is also important. A propellant with a low storage temperature, i.e. a cryogenic, will require thermal insulation, thus further increasing the mass of the launcher. The toxicity of the propellant is likewise important. Safety hazards exist when handling, transporting, and storing highly toxic compounds. Also, some propellants are very corrosive, however, materials that are resistant to certain propellants have been identified for use in rocket construction.

Liquid propellants used by NASA and in commercial launch vehicles can be classified into three types: petroleum, cryogenics, and hypergolics.

Petroleum fuels are those refined from crude oil and are a mixture of complex hydrocarbons, i.e. organic compounds containing only carbon and hydrogen. The petroleum used as rocket fuel is kerosene, or a type of highly refined kerosene called RP-1 (refined petroleum). It is used in combination with liquid oxygen as the oxidizer.

RP-1 and liquid oxygen are used as the propellant in the first-stage boosters of the Atlas/Centaur and Delta launch vehicles. It also powered the first-stages of the Saturn 1B and Saturn V rockets. RP-1 delivers a specific impulse considerably less than cryogenic fuels.

Cryogenic propellants are liquefied gases stored at very low temperatures, namely liquid hydrogen (LH₂) as the fuel and liquid oxygen (LO₂) as the oxidizer. LH₂ remains liquid at temperatures of -423 degrees F (-253 degrees C) and LO₂ remains in a liquid state at temperatures of -298 degrees F (-183 degrees C).

Because of the low temperatures of cryogenic propellants, they are difficult to store over long periods of time. For this reason, they are less desirable for use in military rockets which must be kept launch ready for months at a time. Also, liquid hydrogen has a very low density (0.59 pounds per gallon) and, therefore, requires a storage volume many times greater than other fuels. Despite these drawbacks, the high efficiency of liquid hydrogen/liquid oxygen makes these problems worth coping with when reaction time and storability are not too critical. Liquid hydrogen delivers a specific impulse about 40% higher than other rocket fuels.

Liquid hydrogen and liquid oxygen are used as the propellant in the high efficiency main engines of the space shuttle. LH₂/LO₂ also powered the upper stages of the Saturn V and Saturn IB rockets as well as the second stage of the Atlas/Centaur launch vehicle, the United States' first LH₂/LO₂ rocket (1962).

Hypergolic propellants are fuels and oxidizers which ignite spontaneously on contact with each other and require no ignition source. The easy start and restart capability of hypergolics make them ideal for spacecraft maneuvering systems. Also, since hypergolics

remain liquid at normal temperatures, they do not pose the storage problems of cryogenic propellants. Hypergolics are highly toxic and must be handled with extreme care.

Hypergolic fuels commonly include hydrazine, monomethyl hydrazine (MMH) and unsymmetrical dimethyl hydrazine (UDMH). The oxidizer is typically nitrogen tetroxide (N_2O_4) or nitric acid (HNO_3). UDMH is used in many Russian, European, and Chinese rockets while MMH is used in the orbital maneuvering system (OMS) and reaction control system (RCS) of the Space Shuttle orbiter. The Titan family of launch vehicles and the second stage of the Delta use a fuel called Aerozine 50, a mixture of 50% UDMH and 50% hydrazine.

Hydrazine is also frequently used as a mono-propellant in *catalytic decomposition engines*. In these engines, a liquid fuel decomposes into hot gas in the presence of a catalyst. The decomposition of hydrazine produces temperatures of about 1700 degrees F and a specific impulse of about 230 or 240 seconds.

Solid Propellants

Solid propellant motors are the simplest of all rocket designs. They consist of a casing, usually steel, filled with a mixture of solid compounds (fuel and oxidizer) which burn at a rapid rate, expelling hot gases from a nozzle to produce thrust. When ignited, a solid propellant burns from the center out towards the sides of the casing. The shape of the center channel determines the rate and pattern of the burn, thus providing a means to control thrust. Unlike liquid propellant engines, solid propellant motors can not be shut down. Once ignited, they will burn until all the propellant is exhausted.

There are two families of solids propellants: homogeneous and composite. Both types are dense, stable at ordinary temperatures, and easily storable.

Homogeneous propellants are either simple base or double base. A simple base propellant consists of a single compound, usually nitrocellulose, which has both an oxidation capacity and a reduction capacity. Double base propellants usually consist of nitrocellulose and nitroglycerine, to which a plasticiser is added. Homogeneous propellants do not usually have specific impulses greater than about 210 seconds under normal conditions. Their main asset is that they do not produce traceable fumes and are, therefore, commonly used in tactical weapons. They are also often used to perform subsidiary functions such as jettisoning spent parts or separating one stage from another.

Modern composite propellants are heterogeneous powders (mixtures) which use a crystallized or finely ground mineral salt as an oxidizer, often ammonium perchlorate, which constitutes between 60% and 90% of the mass of the propellant. The fuel itself is aluminum. The propellant is held together by a polymeric binder, usually polyurethane or polybutadienes. Additional compounds are sometimes included, such as a catalyst to help increase the burning rate, or other agents to make the powder easier to manufacture. The final product is rubberlike substance with the consistency of a hard rubber eraser.

Solid propellant motors have a variety of uses. Small solids often power the final stage of a launch vehicle, or attach to payloads to boost them to higher orbits. Medium solids such

as the Payload Assist Module (PAM) and the Inertial Upper Stage (IUS) provide the added boost to place satellites into geosynchronous orbit or on planetary trajectories.

The Titan, Delta, and Space Shuttle launch vehicles use strap-on solid propellant rockets to provide added thrust at liftoff. The Space Shuttle uses the largest solid rocket motors ever built and flown. Each booster contains 1,100,000 pounds (499,000 kg) of propellant and can produce up to 3,300,000 pounds (14,680,000 Newtons) of thrust.

Hybrid Propellants

Hybrid propellant engines represent an intermediate group between solid and liquid propellant engines. One of the substances is solid, usually the fuel, while the other, usually the oxidizer, is liquid. The liquid is injected into the solid, whose fuel reservoir also serves as the combustion chamber. The main advantage of such engines is that they have high performance, similar to that of solid propellants, but the combustion can be moderated, stopped, or even restarted. It is difficult to make use of this concept for very large thrusts, and thus, hybrid propellant engines are rarely built.

Tools Developed

The purpose of the propulsion system module `propulsion.m` is to determine the mass of the propulsion system required to support three orbital maneuvers, characterized by their ΔV s. The module uses the rocket equation to determine the propellant mass required for each maneuver.

$$m_p = m_o \left(1 - e^{-\Delta V / I_{sp} g} \right)$$

A check is performed to ensure that each candidate propulsion system is capable of performing each of the maneuvers in a reasonable amount of time; the argument values used here specify that the burn time must be less than the time it takes to travel 2° through the current orbit. This check is necessary because the orbital transfer equations were derived under the assumption of instantaneous application of ΔV .

Inputs to the propulsion module

Quantity	Units	Description
<code>sc_mass</code>	kg	The mass of the spacecraft.
<code>dV1</code>	m/s	The magnitude of the first ΔV .
<code>dT1</code>	s	The time in which <code>dV1</code> must be completed.
<code>dV2</code>	m/s	The magnitude of the second ΔV .
<code>dT2</code>	s	The time in which <code>dV2</code> must be completed.
<code>dV3</code>	m/s	The magnitude of the third ΔV .
<code>dT4</code>	s	The time in which <code>dV3</code> must be completed.

Outputs from the propulsion module

Quantity	Description
----------	-------------

`p_data` A data structure containing information on the propulsion system such as mass and index into the propulsion database.

`propulsion.m`

```
function [varargout] = propulsion(varargin)
%PROPULSION Sizes the propulsion system to provide a particular delta V
%
% [P_DATA] = PROPULSION
% Returns the propulsion system database.
%
% [PROP] = PROPULSION(SC_MASS, DV1,DT1, DV2,DT2, DV3,DT3)
% [PROP] = PROPULSION(SC_MASS, DV1,DT1, DV2,DT2, DV3,DT3, P_INDEX)
% Determines the mass of the propulsion system, given a spacecraft initial
% mass SC_MASS and required changes in velocity DVn in times DTn. Selects
% the optimal propulsion system from among the choices in P_DATA, unless
% a specific P_INDEX is specified as an argument. PROP is a structure
% with the following fields:
%
% PROP
% .P_INDEX      index into the P_DATA structure
% .MASS_TOTAL   total propulsion system mass
% .MASS_PROPELLANT  propellant mass
% .MASS_DRY     propulsion system dry mass (thrusters, tanks, etc)
% .FIRE_TIMES   firing times required to reach each DVn

g = 9.806; % [m/s^2] gravitational acceleration at sea level
pIndex = 0;

% load in all the propulsion data
pdata = propulsion_data;

% check for invalid number of outputs
if (nargout > 1)
    error('Invalid number of output arguments');
    varargout{1} = [];
    return
end

% determine which output type given the arguments
switch nargin
case 0
    % return the P databse
    varargout{1} = pdata;
    return
case {7,8}
    sc_mass = varargin{1};
    dV(1) = varargin{2};
    dT(1) = varargin{3};
    dV(2) = varargin{4};
    dT(2) = varargin{5};
    dV(3) = varargin{6};
    dT(3) = varargin{7};
    if nargin==8
        pIndex = varargin{8};
    end
otherwise
    error('Invalid number of input arguments');
    varargout{1} = [];
```

```

    return
end

% find the propulsion system mass
Isps = cat(2,pdata.Isps);
m_prop1 = sc_mass*(1-exp(-dV(1)./(Isps*g)));
m_prop2 = sc_mass*(1-exp(-dV(2)./(Isps*g)));
m_prop3 = sc_mass*(1-exp(-dV(3)./(Isps*g)));
m_dry = cat(2,pdata.mass);
m_total= m_prop1+m_prop2+m_prop3+m_dry;

% make sure the maneuver can complete in the specified time
indices = 1:length(pdata);
for i=1:3
    times = sc_mass*dV(i)./cat(2,pdata.thrust);
    indices = intersect(indices, find(times<=dT(i)));
    fire_times(i,:) = times;
end

% if the user did not specify a system, select the mass-optimal one
if ~pIndex
    [opt_mass, pIndex] = min(m_total(indices));
    if isempty(pIndex)
        varargout{ 1 } = [];
        return
    end
    pIndex = indices(pIndex);
end

prop.p_index = pIndex;
prop.mass_total = m_total(pIndex);
prop.mass_dry = m_dry(pIndex);
prop.mass_propellant = [m_prop1(pIndex) m_prop2(pIndex) m_prop3(pIndex)];
prop.fire_times = fire_times(:,pIndex);

varargout{ 1 } = prop;

%% -----%%

function [pdata] = propulsion_data
%PROPULSION_DATA
%
% [P_DATA] = PROPULSION_DATA
% Returns a data structure containing information on several
% propulsion systems.
%
% Data are from http://www.asi.org/adb/04/03/09/01/ and SMAD III page 694.

i=0;
i=i+1;pdata(i)=struct('name','MR-103','company','Rocket
Research','thrust',0.89,'Isp',216,'fuel','hydrazine','ox','hydrazine','life',NaN,'mass',0.332);
i=i+1;pdata(i)=struct('name','MR-104','company','Rocket
Research','thrust',445,'Isp',228,'fuel','hydrazine','ox','hydrazine','life',NaN,'mass',1.86);
i=i+1;pdata(i)=struct('name','MR-106','company','Rocket
Research','thrust',26.7,'Isp',225,'fuel','hydrazine','ox','hydrazine','life',NaN,'mass',0.476);
i=i+1;pdata(i)=struct('name','MR-107','company','Rocket
Research','thrust',178,'Isp',226,'fuel','hydrazine','ox','hydrazine','life',NaN,'mass',0.885);
i=i+1;pdata(i)=struct('name','Dual mode liquid apogee
engine','company','TRW','thrust',454,'Isp',314.5,'fuel','hydrazine','ox','MON','life',NaN,'mass',4.8);
i=i+1;pdata(i)=struct('name','S400','company','DASA','thrust',9.7,'Isp',287,'fuel','MMH','ox','MON','life',NaN,'mass',0.35
);

```

```

i=i+1;pdata(i)=struct('name','Ariane
L9','company','DASA','thrust',27500,'Isp',324,'fuel','MMH','ox','N2O4','life',NaN,'mass',111);
i=i+1;pdata(i)=struct('name','XLR-
132','company','Rocketdyne','thrust',16700,'Isp',340,'fuel','N2O4','ox','MMH','life',5000,'mass',51.26);
i=i+1;pdata(i)=struct('name','Transtar','company','Aerojet','thrust',16700,'Isp',334,'fuel','N2O4','ox','MMH','life',5400,'m
ass',57.15);
i=i+1;pdata(i)=struct('name','Transtage','company','Aerojet','thrust',35600,'Isp',315,'fuel','N2O4','ox','A-
50','life',1000,'mass',107.95);
i=i+1;pdata(i)=struct('name','Delta-
II','company','Aerojet','thrust',43600,'Isp',320,'fuel','N2O4','ox','MMH','life',1200,'mass',99.79);
i=i+1;pdata(i)=struct('name','R-
4D','company','Marquardt','thrust',4000,'Isp',309,'fuel','N2O4','ox','MMH','life',25000,'mass',7.26);
i=i+1;pdata(i)=struct('name','OME/UR','company','Aerojet','thrust',26700,'Isp',340,'fuel','N2O4','ox','MMH','life',1200,'
mass',90.72);
i=i+1;pdata(i)=struct('name','RL10-A','company','Pratt &
Whitney','thrust',73400,'Isp',446,'fuel','LO2','ox','LH2','life',400,'mass',138.35);
i=i+1;pdata(i)=struct('name','DM/LAE','company','TRW','thrust',445,'Isp',302,'fuel','N2O4','ox','MMH','life',20000,'mas
s',5.22);
i=i+1;pdata(i)=struct('name','R-
4D','company','Marquardt','thrust',489,'Isp',310,'fuel','N2O4','ox','MMH','life',20000,'mass',3.76);
i=i+1;pdata(i)=struct('name','R-42','company','Marquardt','thrust',890,'Isp',305,'fuel','MON-
3','ox','MMH','life',20000,'mass',4.54);
i=i+1;pdata(i)=struct('name','MMBPS','company','TRW','thrust',445,'Isp',302,'fuel','N2O4','ox','MMH','life',20000,'mass'
,5.22);
i=i+1;pdata(i)=struct('name','RS-
41','company','Rocketdyne','thrust',11100,'Isp',312,'fuel','N2O4','ox','MMH','life',2000,'mass',113.40);
i=i+1;pdata(i)=struct('name','ADLAE','company','TRW','thrust',445,'Isp',330,'fuel','N2O4','ox','MMH','life',28000,'mass'
,4.5);
i=i+1;pdata(i)=struct('name','HS601','company','ARC/LPG','thrust',489,'Isp',312,'fuel','N2O4','ox','MMH','life',10000,'m
ass',4.08);
i=i+1;pdata(i)=struct('name','R-
40A','company','Marquardt','thrust',4000,'Isp',309,'fuel','N2O4','ox','MMH','life',25000,'mass',7.26);
i=i+1;pdata(i)=struct('name','HPLAM','company','TRW','thrust',445,'Isp',325,'fuel','N2O4','ox','MMH','life',30000,'mass'
,4.6);

% there are also electric propulsion systems, but these are beyond the scope of this project
% i=i+1;pdata(i)=struct('name','MR-508 Arcjet','company','Rocket
Research','thrust',0.21,'Isp',502,'fuel','hydrazine','ox','', 'life',NaN,'mass',1.338);

```

Useful References

Space Mission Analysis and Design (SMAD), Wertz & Larson, Third edition, Microcosm Press, 1999.

Braeunig, David A. Rocket and Space Technology.
<http://users.commkey.net/Braeunig/space/propel.htm> (21 November 2003).

The Artemis Project, <http://www.asi.org/adb/04/03/09/01/>, 13 Nov 2003.

6. Space Environment

A spacecraft around Earth orbit is irradiated by many types of radiations, including trapped radiation, solar particle events, and Galactic cosmic rays. In particular, the trapped radiation of the Van Allen radiation belts is problematic to Earth orbiting satellites' solar arrays performance.

Learning Objectives

- Overview of effects: Outgassing in near vacuum, Atmospheric drag, Chemical reactions, Plasma-induced charging, Radiation damage of microcircuits, solar arrays, and sensors, Single event upsets in digital devices, Hyper-velocity impacts
- Solar Cycle
- Gravity
- Neutral Atmosphere
- Ionosphere
- Geomagnetic Field
- Plasma
- Radiation

Reflections on Learning Experience

My third problem set on characterizing the degradation of a solar array given orbit position was a good opportunity to study the near-Earth space environment. I found lecture fascinating on this subject and am glad it was broken into two parts. I did not delve into the hardness and survivability issues as much as I would have liked—particularly the issue of reducing satellite vulnerability to nuclear weapon effects.

Tools Developed

```
% Predicting Solar Cell Degradation as a Function of Radiation
% 16.851 Problem Set #3
```

```
% This module performs two tasks
% 1. Model power loss of GaAs/Ge cells following proton irradiation
% 2. Model power loss of n/p Si cells following electron irradiation
```

```
% Define Constants
P_o = 1
C = .1295
D_x = 1.295*10^9
```

```
% Input Dose of Protons
D = [10^7 10^7.5 10^8 10^8.5 10^9 10^9.5 10^10 10^10.5 10^11 10^11.5 10^12] % MeV/g
```

```
% Calculate Characteristic Curve For Proton Damage of GaAs/Ge cells
P_max = (1-C*log(1+D./D_x)).*P_o
```

```
% Graph P_max / P_o, the normalized maximum power of GaAs/Ge cells as a function of absorbed proton dose
figure
semilogx(D, P_max, '-');
```

```

title('Power loss of GaAs/Ge cells following proton irradiation as a function of absorbed dose')
xlabel('Absorbed dose (MeV/g)')
ylabel('P_max / P_o')
AXIS([10^7 10^12 .4 1])
hold

% Electron induced damage
% Overall equation: D_eff = D * (S_e(E) / S_e(1.0))^(n-1)

% Defining variables
D_eff = 1.0; % Measured in MeV

% Experimentally determined exponents
% n=1 for n-type silicon
% n=2 for p-type silicon, GaAs, and InP

% Calculate Characteristic Curve For Electron Damage of n/p Si cells
P_o = 1
C = 0.06593
D_x = 1.841*10^8
P_max = (1-C*log(1+D./D_x)).*P_o

% Input Dose of Electrons
D = [10^7 10^7.5 10^8 10^8.5 10^9 10^9.5 10^10 10^10.5 10^11 10^11.5 10^12]
% MeV/g

% Graph P_max / P_o, the normalized maximum power of n/p Si cells as a function of absorbed electron dose
figure
semilogx(D, P_max, '-');
title('Power loss of n/p Si cells following electron irradiation as a function of absorbed dose')
xlabel('Absorbed dose (MeV/g)')
ylabel('P_max / P_o')
AXIS([10^7 10^12 .4 1])
Hold

```

Useful References

N. A. Tsyganenko, A. V. Usmanov, Determination of the Magnetospheric Current System Parameters and Development of Experimental Geomagnetic Field Models Based on Data from IMP and HEOS Satellites, *Planet. Space Sci.* **30**, 985-998, 1982.

<http://nssdc.gsfc.nasa.gov/space/model/>

Allen, Douglas M. "A survey of next generation solar arrays (for spacecraft electric power)." *W. J. Schafer Associates*. AIAA, 1997.

Ansbaugh, Bruce E. "Proton and Electron Damage Coefficients for GaAs/Ge Solar Cells." *Jet Propulsion Laboratory*. IEEE, 1991.

Gaddy, Edward M. "Relative cost effectiveness of multi-junction, gallium arsenide, and silicon solar cells." *NASA Goddard Space Flight Center*. AIAA, 1996.

Gardner, B. M. "A Tool for Assessing Photovoltaic Array End-of-Life Power Performance." *NASA Lewis Research Center*. AIAA, 1997.

Kerslake, Thomas W. "Solar Power System Options for the Radiation and Technology Demonstration Spacecraft." *NASA Glenn Research Center*. AIAA, 2000.

Landis, Geoffrey A. "Photovoltaic Power for Future NASA Missions." *NASA John Glenn Research Center*. AIAA, 2002.

Ralph, E. "Solar Cell Array System Trades – Present and Future." *TECSTAR/ASD*. AIAA, 1999.

"Research on Radiation Damage Mechanisms of Si Solar Cell." *NASDA Office of Research and Development*. No. 73, July 1998. http://www.nasda.go.jp/lib/nasda-news/1998/07/solar_e.html (5 October 2003).

Sharps, P. R. "High Efficiency Advanced Triple-Junction Space Solar Cell Production at Emcore Photovoltaics." *Emcore Photovoltaics*. AIAA, 2003.

"Solar Radiation Handbook." *Jet Propulsion Laboratory*. Third Edition, 1982.

Summers, Geoffrey P. "A General Method for Predicting Radiation Damage to Solar Cells in the Space Environment." *Naval Research Laboratory*. IEEE, 1991.

Summers, Geoffrey P. "A New Approach To Damage Prediction For Solar Cells Exposed To Different Radiations." *Naval Research Laboratory*. IEEE, 1994.

Visentine, J. "MIR Solar Array Return Experiment." *Boeing Space Station Program Office*. AIAA, 1999.

7. Optics

Lightweight infrared optical systems technologies will enable future NASA missions including earth observation and remote sensing systems, satellite-to-satellite communications systems, aircraft-borne LIDAR (light detection and ranging) systems for detection of wind shear and military surveillance and early warning systems.

Learning Objectives

The goal of the optics lecture was to provide the necessary optics background to tackle a space mission, which includes an optical payload. Topics covered in the lecture notes include:

- Light
- Interaction of Light w/ environment
- Optical design fundamentals
- Optical performance considerations
- Telescope types and CCD design
- Interferometer types
- Sparse aperture array
- Beam combining and Control
- ARGOS overview

Reflections on Learning Experience

SMAD offers a strong introduction to basic telescope optics. Our lecture on this subject was a fast-paced overview of the fundamentals, proposed future missions, and ARGOS. I was intrigued by the Michelson and Fizeau interferometer missions.

Thermal-Optical Interaction

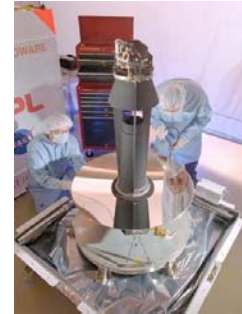
One of the key performance drivers of infrared systems is temperature. CCD (charge-coupled devices) detectors perform best when they are cool. At ordinary temperatures, the signal output of a photon detector is swamped by the background noise. This is due to random generation and recombination of carriers in the semiconductor. Cooling is essential for maintaining a signal-to-noise ratio that permits useful observations for infrared instruments. We need to avoid any contribution to instrumental noise by the telescope.

There are a variety of methods to cool detectors in flight. The method of cooling chosen for a space system depends on the operating temperature and the system's logistical requirements. Passive-cooling methods such as mounting detectors to a thermally conductive metal part that thermally connects to a radiator facing deep space is an easy way to maintain detector temperatures around 50 K. Thermal shields are also used to protect instruments from sunlight, and even earthlight and moonlight. Active cooling

techniques, such as cryogenic coolers or Adiabatic Demagnetization Refrigerators, are often necessary to deal with higher heat loads.

Sample Infrared Imagers

The Space Infrared Telescope Facility (**SIRTF**)—which was recently launched from Cape Canaveral, Florida on 25 August 2003—is designed to obtain IR images and spectra by detecting wavelengths of 3 and 180 microns during its 2.5 year mission. SIRTF is the largest infrared telescope ever launched into space and it needs to be cooled to near absolute zero (5.5K). In addition to protection from its own heat, the telescope also must be protected from the heat of the Sun and the infrared radiation put out by the Earth.



The Near Infrared Camera (**NIRCam**) is to be the primary James Webb Space Telescope (JWST) imager operating over the wavelength range of 0.6 to 5 microns at a temperature of 50K. The NIRCam is designed for detection of the early phases of star and galaxy formation such as detection of distant supernovae and mapping dark matter through gravitational lensing. The camera consists of two broad-band and intermediate-band imaging modules and two tunable filter imaging modules, each with a 2.3 x 2.3 arcmin field of view. The short wavelength channels and the long wavelength channels are to be sampled at 4096 x 4096 pixels and 2048 x 2048 pixels, respectively. One tunable filter module is optimized for wavelengths from about 1.2 to 2.5 microns, the other from 2.5 to 4.5 microns.

Hubble's Near Infrared Camera and Multi-Object Spectrometer (**NICMOS**) provides imaging capabilities in the broad, medium, and narrow band filters, broad-band imaging polarimetry, coronagraphic imaging, and slitless grism spectroscopy. It covers the wavelength range of 0.8-2.5 microns and has a current temperature requirement of 77.1K. NICMOS has three adjacent cameras designed to operate independently. Each has a dedicated array at a different magnification scale.

Useful References

Larson, W.J., Wertz, J.R., "Space Mission Analysis and Design", Second Edition, 9.5 Designing Visual and IR Payloads, pp.. 249-274, Microcosm, Inc,1992

Born Max, Wolf Emil, "Principles of Optics", Electromagnetic Theory of propagation interference and diffraction of light, Sixth Edition, Cambridge University Press, 1998

Hecht E. "Optics", Addison-Wesley, 1987

Günter Diethmar Roth, "Compendium of Practical Astronomy", Volume 1, Instrumentation and Reduction Techniques, Springer Verlag, Berlin, New York, ISBN 3-540-56273-7, 1994

8. Attitude Control Systems

The purpose of the attitude determination and control system is to stabilize the spacecraft and orient it during mission operations. It compensates for external torques acting on the vehicle. Attitude is determined using sensors (GPS, star trackers, limb sensors, rate gyros, inertial measurement units) and controlled using actuators (Reaction Wheel Assemblies, Control Moment Gyros, magnetic torque rods, thrusters).

Learning Objectives


As defined in the lecture notes, the learning objectives are to understand:

- Coordinate Systems and Mathematical Attitude Representations
- Rigid Body Dynamics
- Disturbance Torques in Space
- Passive Attitude Control Schemes
- Actuators
- Sensors
- Active Attitude Control Concepts
- ADCS Performance and Stability Measures
- Estimation and Filtering in Attitude Determination
- Maneuvers
- Other System Consideration, Control/ Structure interaction
- Technological Trends and Advanced Concepts


Reflections on Learning Experience

I did not trade this subsystem in any of the problem sets so my learning experience is restricted to reading section 11.1 in *SMAD* and the material covered in class. I understand the concepts behind the passive and active attitude control schemes and the differences in their performance, but did not absorb some of the math theory covered in the fast-paced lecture. In the future, I would like to explore the implications of satellite servicing on the attitude control systems of two interacting satellites.

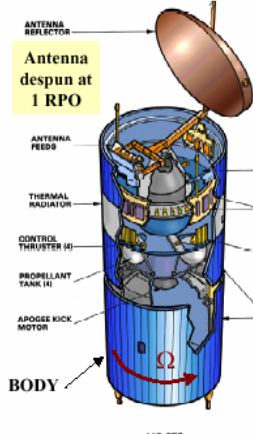
Excerpts from Lecture



Spin Stabilized Spacecraft



UTILIZED TO STABILIZE SPINNERS

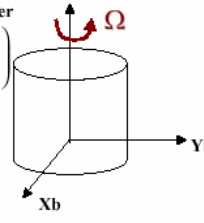


Antenna despun at 1 RPO

HS 378 SPACECRAFT CONFIGURATION


Perfect Cylinder

$$I_{xx} = I_{yy} = \frac{m}{4} \left(\frac{L^2}{3} + R^2 \right)$$


$$I_{zz} = \frac{mR^2}{2}$$


DUAL SPIN

- Two bodies rotating at different rates about a common axis
- Behaves like simple spinner, but part is despun (antennas, sensors)
- requires torquers (jets, magnets) for momentum control and nutation dampers for stability
- allows relaxation of major axis rule



Disturbance Torques



Assessment of expected disturbance torques is an essential part of rigorous spacecraft attitude control design

Typical Disturbances

- Gravity Gradient: “Tidal” Force due to 1/r² gravitational field variation for long, extended bodies (e.g. Space Shuttle, Tethered vehicles)
- Aerodynamic Drag: “Weathervane” Effect due to an offset between the CM and the drag center of Pressure (CP). Only a factor in LEO.
- Magnetic Torques: Induced by residual magnetic moment. Model the spacecraft as a magnetic dipole. Only within magnetosphere.
- Solar Radiation: Torques induced by CM and solar CP offset. Can compensate with differential reflectivity or reaction wheels.
- Mass Expulsion: Torques induced by leaks or jettisoned objects
- Internal: On-board Equipment (machinery, wheels, cryocoolers, pumps etc...). No net effect, but internal momentum exchange affects attitude.

Useful References

Space Mission Analysis and Design (SMAD), Wertz & Larson, Third edition, Microcosm Press, 1999.

Fundamentals of Space Systems, Pisacane and Moore, Oxford, 1994.

Smith, J. L. Low-Cost Attitude Determination and Control for Small Satellites.
<http://www.sdl.usu.edu/conferences/smallsat/proceedings/10/sess10/alowcost.pdf>

Space Systems Engineering, Fortesque and Stark, John Wiley and Sons, 1995.

9. Electromagnetic Formation Flight

The traditional use of propellant as a reaction mass has the advantages of a mobile spacecraft center of gravity and independent control of each spacecraft. However, disadvantages include the fact that propellant is a limited resource and that propellant can contaminate precision optics. Electromagnetic Formation Flight provides the opportunity for propellant-less control (excluding absolute position of the center of mass of a cluster of spacecraft) for image construction, Earth coverage, disturbance rejection, and docking missions.


Learning Objectives

- Fundamental Principles
 - Governing Equations
 - Trajectory Mechanics
 - Stability and Control
- Mission Applicability
 - Sparse Arrays
 - Filled Apertures
 - Other Proximity Operations
- Mission Analyses
 - Sparse Arrays
 - Filled Apertures
 - Other Proximity Operations
- MIT EMFFORCE Testbed
 - Design
 - Calibration
 - Movie
- Space Hardware Design Issues
 - Thermal Control
 - Power System Design
 - High B- Field Effects


Reflections on Learning Experience

This was a fascinating subject, especially given the ongoing research at MIT SSL in this area. Prior to coverage of EMFF in lecture, I was not aware of its potential to provide longer satellite mission lifetime and to reduce contamination and degradation.

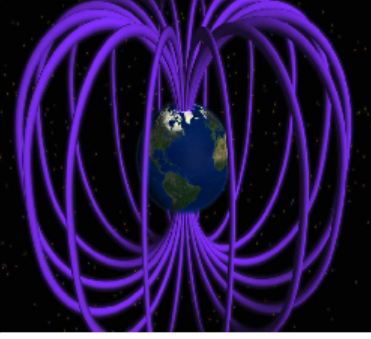
Excerpts from Lecture




Earth's Magnetic Field




- Many formation flight missions will operate in LEO.
- The electromagnets will interact with the Earth's magnetic field producing unwanted forces and torques on the formation
- The Earth's magnetic field can be approximated by a large bar magnet with a magnetic dipole strength of $8 \cdot 10^{22}$
- (EMFF Testbed $\sim 2 \cdot 10^4$)





EMFF Applications in 10-20 Years



Sparse Apertures


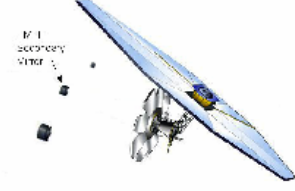
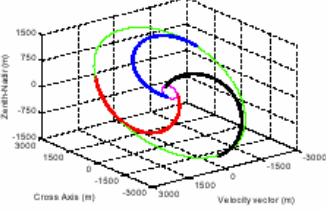


Image from 1999 TFF Book

EMFF Secondary Mirrors

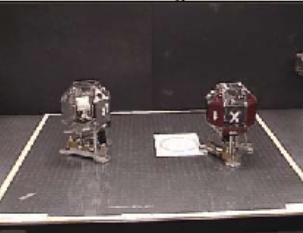


Cluster Reconfiguring



Z-axis (m) 1500 750 0 -750 -1500 -3000
Cross Axis (m) -1500 0 1500 -3000
Velocity vector (m)

Docking



Useful References

CDIO Project: <http://cdio-prime.mit.edu/CDIO3/>

CDIO Progress Report:

http://cdio-prime.mit.edu/CDIO3/NRO_REPORTS/NRO3_200206.pdf

10. Spacecraft Computers and Software

As stated in the lecture notes, “spacecraft systems and their software provide unprecedented capability on orbit, but drive system cost and complexity.” Computer systems in space are similar to desktop applications, but require low power, low volume, low mass, and high reliability and fault tolerance.

Learning Objectives

Topics covered in lecture on spacecraft computers include:

- Definitions
- Computer system specification
- Estimating throughput and processor speed requirements
- Computer selection
- Memory
- Mass storage
- Input/output
- Radiation hardness
- Fault tolerance
- Error detection and correction
- Integration and test

Topics on space systems software:

- Types of software
- Computer resource estimation
- Functional partitioning
- Software maintenance and life-cycle support
- Costing (I think we decided that this was impossible)

Reflections on Learning Experience

SMAD and lecture were both good introductions to these topics. Coming in to the class, the only course I had taken on computers and software was 16.070, and I appreciated the coverage of the fundamentals. I did not develop any MATLAB modules nor did I review any problems sets that dealt with these subsystems.

Useful References

History of Computers on NASA Space Missions:

<http://www.hq.nasa.gov/office/pao/History/computers/Part1.html>

Space Mission Analysis and Design (SMAD), Wertz & Larson, Third edition, Microcosm Press, 1999.

Fundamentals of Space Systems, Pisacane and Moore, Oxford, 1994

11. Thermal System

The thermal subsystem maintains operational or survival temperatures for all parts of the spacecraft during all mission phases. Temperature control is critical for batteries, hydrazine propulsion equipment, pointing structures (to minimize differential thermal expansion), and electronic components. Typically, the thermal control subsystem accounts for 2-5% of the spacecraft cost and dry weight.

Learning Objectives

- Principles of Heat Transfer, including convection, conduction, radiation
- Thermal Control Components (materials and components, optical solar reflectors, silver-coated Teflon, multiplayer insulation, electrical heaters, thermostats...)
- Thermal Subsystem Design
- Thermal Analysis Components (equilibrium temperature, temperature of insulated surfaces in space, solar array/flat plate analysis...)
- Development of new thermal control materials having resistance to extreme temperatures and controlled outgassing behavior

Reflections on Learning Experience

I found thermal to be one of the more difficult subsystems covered, especially in the development of modules trading parameters with other subsystems. *SMAD* Chapter 11.5 on the thermal subsystem offered a good introduction to the principles of thermal control but little in the way of tables or design approaches useful for module development. In problem set two we traded optical temperature parameters with various passive and active cooling techniques.

Tools Developed

The list of inputs to our thermal model includes the following:

- emissivity and absorptivity of all entity surfaces
- area of these surfaces
- geometrical factors: view factors and area fractions
- temperature of the sunshield and the satellite core
- cooling power and minimum reachable temperature of the cryocooler
- thermal resistance of the different conductive paths.

The outputs are the different temperatures of the entities, and the radiative and conductive fluxes from and to each spacecraft entity. Excerpts from the Excel thermal model:

Radiative Characteristics

eps2Emissivity	(Black paint)	0.874
alpha2Absorptivity		0.975
Absorptivity/Emissivity ratio	alpha2/eps2	1.115560641
rho2Reflectivity	1-alpha2	0.025

Geometry Factors

A2Area (m2)		4
<i>View Factors (VIEW FROM SHIELD)</i>		
f20Uncoated optical device	f20	0.114
f21Coated optical device	f21	0.1425
f22Thermal shield		0.5
f24Radiator		0
f25Other parts of the spacecraft (including cryocooler)		0
<i>Special Factors</i>		
xA27Fraction of device in the sunlight		0
xA26Fraction of device in view of outer space		1
xA28Fraction of device in view of Earth		0
<i>View Factors (VIEW FROM OUT THERE)</i>		
f02Uncoated optical device	f20*A2/A0	0.8
f12Coated optical device	f21*A2/A1	1
f22Thermal shield	f22	0.5
f42Radiator	f24*A2/A4	0
f52Other parts of the spacecraft (including cryocooler)	f25*A2/A5	0

*Some Incident Radiative Fluxes
Per Unit Area Of Shield (W/m2)*

phi62From outer space	phi60	4.5927E-06
phi72From Sun	phi70	1418
phi82From Earth	phi80	754.3

Incident Radiative Fluxes (W)

PHI02From uncoated optical device	f02*PHI0x	0.014918897
PHI12From coated optical device	f12*PHI1x	0.023307504
PHI22From thermal shield	f22*PHI2x	0.081780715
PHI42Radiator	f42*PHI4x	0
PHI52From other parts of the spacecraft (including cryocooler)	f52*PHI5x	0
PHI62From outer space	phi62*A2*xA26	1.83708E-05
PHI72From Sun	phi72*A2*xA27	0
PHI82From Earth	phi82*A2*xA28	0
PHIx2Sum of the incoming radiative fluxes		0.120025486

Temperature (°C)	306.15
T2Temperature (K)	30

Outgoing Fluxes (W)

<i>Outgoing Radiative Fluxes</i>		
PHI_E2xEmitted radiation flux (W)	$A2 * \epsilon^2 * \sigma * T2^4$	0.160560792
PHI_R2xReflected flux (W)	$\rho^2 * \Phi x2$	0.003000637
PHI 2xSum of outgoing radiative fluxes		0.16356143

Useful References

<http://cryowwwwebber.gsfc.nasa.gov/ADR/ADR.html>

<http://eo1.gsfc.nasa.gov/miscPages/TechForumReports/CCR-2.pdf>

<http://ircamera.as.arizona.edu/nircam/features.html>

<http://sirtf.caltech.edu/about/index.shtml>

<http://techreports.larc.nasa.gov/ltrs/PDF/1998/mtg/NASA-98-13asc-wlv.pdf>

<http://www.astrium-space.com/corp/prod/index.htm>

<http://www.cme.rl.ac.uk/cryo/cooler.htm>

<http://www.emsstg.com/pdf/sunshield.pdf>

<http://www.ena.umd.edu/ASC/abstracts/abs146.pdf>

http://www.engin.umich.edu/class/ae483/Lander/Power_Thermal/FINALPowerThermalReport.doc

<http://www.estec.esa.nl/thermal/cryo2a.html>

<http://www.estec.esa.nl/thermal/cryo2b.html>

<http://www.ilcdover.com/SpaceInf/SunShields/ngst.htm>

<http://www.ilcdover.com/SpaceInf/SunShields/skylab.htm>

<http://www.ilcdover.com/WebDocs/isis.pdf>

http://www.jpl.nasa.gov/adv_tech/coolers/Cool_ppr/C11-tes.pdf

http://www.jpl.nasa.gov/adv_tech/coolers/Cool_ppr/C12-ovr.pdf

<http://www.lgarde.com/people/papers/highpower.pdf>

<http://www.mssl.ucl.ac.uk/~lbcw/epsrccadr/epsrccadr2.htm>

http://www.ngst.nasa.gov/public/unconfigured/doc_1008/rev_01/NGST-ARTL-001944.PDF

http://www.space.com/business/technology/technology/webb_ngst_030108.html

http://www.starsys.com/catalog/pdf/src_103.pdf

http://www.starsys.com/catalog/pdf/src_105.pdf

<http://www.sunpower.com/products/index.html>

http://www.sunpower.com/products/m77_gs.html

<http://www.swift.psu.edu/xrt/Documents/XRT-GSFC-004.pdf>

<http://www.tsgc.utexas.edu/archive/subsystems/thermal.pdf>

[http://www-glast.slac.stanford.edu/MechanicalSystems/Meetings&Talks/I-PDR/PublishedIPDRDocs/LAT-TD-00330-2\(RadPDRDesignReport\).pdf](http://www-glast.slac.stanford.edu/MechanicalSystems/Meetings&Talks/I-PDR/PublishedIPDRDocs/LAT-TD-00330-2(RadPDRDesignReport).pdf)

12. Communications, Telemetry, Tracking, & Command

The telemetry, tracking, and command subsystem provides the interface between the spacecraft and the ground systems, including carrier tracking, command reception and detection, telemetry modulation and transmission, ranging, and subsystem operations. A communications architecture is the arrangement of satellites and ground stations and the communications links that transfer information between them.

Learning Objectives

As outlined in lecture, the learning objectives of this section include understanding:

- Communication terminology (i.e. uplink, downlink, crosslink, relays)
- Communication architecture as defined by satellite-ground geometry, including store & forward, geostationary, and Molniya
- Communication selection criteria
 - Orbit
 - RF spectrum
 - Data rate
 - Duty factor
 - Link availability
 - Link access time
 - Threat
- Link design process
- Modulation, bit error rate, and coding

Reflections on Learning Experience

Chapter 13 in *SMAD* on communications offers a lengthy introduction to this subsystem with plenty of equations for implementation in MATLAB functions. I thought that this lecture had a good balance between breadth and detail, with interesting case studies of SATCOM and Milstar. I found communications an easy subsystem to pass parameters to and from—but not necessarily a good choice for the problem sets. In the one problem I worked on this term involving communications, it did not seem to have a significant impact on the other subsystems.

Tools Developed

A MATLAB program has been designed to solve the proposed problem. The tool flow is as follows:

- The orbit is found so that the satellite passes through perigee over the ground site once each sidereal day.
- STK is opened, and a scenario is set up with the ground site and satellite. STK returns data indicating how long each pass used for ground communication is, and how long the satellite is in eclipse each orbit.

- The amount of data that can be transmitted to the ground each pass is determined, as well as the power needed for each transmission.
- The solar arrays are designed and a battery is chosen. Each of these parts of the tool will be described in detail in this section.

Orbit Model

The orbit is determined using the ground site latitude and longitude, the altitude of the satellite at perigee, and the time the satellite is first at perigee. The following assumptions and decisions are made:

- The earth is circular with radius = 6378.137 km
- Site elevation is not significant enough to use
- A nearly circular orbit is desirable, so eccentricity will be minimized
- Inclination will be set to 5 degrees above the value of the site latitude; for site latitude over 85 degrees, it will be set to 90

First the radius of perigee rp is found by adding the radius of the earth to the altitude of perigee. Next, the initial eccentricity ei is set to 0.01, for a nearly circular orbit. Using eccentricity and the radius of perigee, an initial value for the orbit's semi-major axis ai and period pi can be found (μ is the earth's gravitational constant in the correct units):

$$ai = rp / (1.0 - ei) \text{ [km]}$$

$$pi = 2 \pi \sqrt{ai^3 / \mu} \text{ [minutes]}$$

However, the orbit must be a repeating ground track, which requires that the period p be equal to an integer number of sidereal days divided by an integer number of revolutions. The number of revolutions using the initial period is found:

$$nrevs = (1436.068167 \text{ minutes}) / pi$$

The number of revolutions is rounded down to an integer, and the final period, semi-major axis, and eccentricity is found. The eccentricity will always be larger than 0.01, but it will be small. For the satellite to pass over the ground site, inclination i can be in the range:

$$site_latitude \leq i \leq 180 \text{ deg} - site_latitude$$

As mentioned, inclination is arbitrarily chosen to be $site_latitude + 5$ degrees. The argument of perigee ω and longitude of ascending node Ω are found so that the satellite will be at perigee over the site at the desired starting time. Argument of perigee is found by the equation:

$$\omega = \text{asin}(\sin(site_latitude) / \sin(i)) \text{ [deg]}$$

Longitude of ascending node is found by the following series of equations:

$$temp1 = \text{asin}(\cos(i)/\cos(\text{site_latitude})) \text{ [deg]}$$

$$temp2 = \text{acos}(\cos(temp1)/\sin(i)) \text{ [deg]}$$

$$\hat{U} = \text{gst} + \text{site_longitude} - temp2; \text{ [deg]}$$

Where *gst* is Greenwich sidereal time, found from the starting time (see MATLAB function *find_gst*).

STK

The orbit found is entered into STK by connecting through MATLAB and setting up a new scenario (see MATLAB function *run_stk*). The ground site is also added to the STK scenario as a facility. STK can return reports detailing scenario behavior for a given time interval. STK is used in this tool to return a report indicating the length of time the satellite is in eclipse each orbit, and to find all of the time periods the satellite has line-of-sight access to the ground site.

Ideally, the STK scenario should be run for the entire lifetime of the satellite, and averages taken over this time period. However, connecting to STK through MATLAB is slow, and STK is slow in doing calculations needed for the reports. Averaging over a 5 or even 1 year life time would make each individual run very slow, and repeated runs for trending would not have been possible. To solve this problem, averages are taken over a much smaller period. While it is acknowledged that this will not provide the most accurate answer, particularly with eclipse times since those may vary over the course of a year, it was decided that this method would be sufficient for the scope of this project.

The scenario is set to run for 12 days. For ground access times, the first and last days are eliminated, since the passes on each of those days will be partial (since the satellite is at perigee at the beginning of the scenario). Note that a sidereal day is nearly 4 minutes shorter than a calendar day. This will cause the pass over the ground site to occur approximately 4 minutes earlier each day. STK reports the eclipse time for each orbit over the 12 days. The first and last eclipse times are thrown out since they may reflect partial eclipses. The remaining values are used and an average is taken.

Run_stk.m

```
function [eclipse, pass_time] = run_stk(orbit, site_lat, site_lon, date)
% opens stk, sets up the orbit and the ground site
% runs scenario
% finds average time craft is in eclipse each orbit
% finds average time of each pass over ground site (average of closest pass
% each day)
% INPUTS
% orbit: [semi-major axis (km), eccentricity, i (degrees), w (degrees), ascending node (degrees)]
% site_lat: site latitude in degrees
% site_lon: site longitude in degrees
% date: start date of orbit (time of first perigee)
% [year, month, day, hour, minute]
% OUTPUTS
% eclipse: average time each orbit craft is in eclipse, in minutes
% pass_time: average time of the longest pass each day, in minutes
disp('Connecting to STK...');
```

```

year = date(1);
day = date(3);
hour = date(4);
min = date(5);
% initialize connection to STK, if this has not yet been done
stkinit;
% use the default connection data for connecting to STK
% port should be 5001, hostname 'localhost': this setup is what STK
% recommends and guides you to do when STK and Matlab are on the same
% machine
remMachine = stkDefaultHost;
%open the connection to STK
conid=stkOpen(remMachine);
% first check to see if a scenario is open
% if there is, close it
scen_open = stkValidScen;
if scen_open == 1
stkUnload('/')
end
% set up scenario
cmd = 'New / Scenario ground_comm';
stkExec(conid, cmd);
% put the satellite in the scenario
cmd = 'New /*/Satellite sat1';
stkExec(conid, cmd);
% put the ground site in the scenario
cmd = 'New /*/Facility ground_site';
stkExec(conid, cmd);
if(date(2) == 1)
month = 'Jan';
elseif(date(2) == 2)
month = 'Feb';
elseif(date(2) == 3)
month = 'Mar';
elseif(date(2) == 4)
month = 'Apr';
elseif(date(2) == 5)
month = 'May';
elseif(date(2) == 6)
month = 'June';
elseif(date(2) == 7)
month = 'July';
elseif(date(2) == 8)
month = 'Aug';
elseif(date(2) == 9)
month = 'Sept';
elseif(date(2) == 10)
month = 'Oct';
elseif(date(2) == 11)
month = 'Nov';
elseif(date(2) == 12)
month = 'Dec';
else
disp('invalid date, month must be 1 through 12');
end
startDate = ['"', num2str(day), ' ', month, ' ', num2str(year), ' ', num2str(hour), ':', num2str(min), ':00.0"'];
stopDay = day + 12;
stopDate = ['"', num2str(stopDay), ' ', month, ' ', num2str(year), ' ', num2str(hour), ':', num2str(min), ':00.0"'];
epochDate = startDate;
% set the scenario epoch
cmd = 'SetUnits / km sec GregUTC';
stkExec(conid, cmd);

```

```

cmd = ['SetEpoch * ' epochDate];
stkExec(conid, cmd);
stkSyncEpoch;
% set the time period for the scenario
stkSetTimePeriod(startDate, stopDate, 'GREGUTC');
% set the animation parameters
tmp = ['SetValues ', startDate, ' 60 0.1'];
rtn = stkConnect(conid,'Animate','Scenario/ground_comm',tmp);
rtn = stkConnect(conid,'Animate','Scenario/ground_comm','Reset');
% set orbit elements
cmd = ['SetState */Satellite/sat1 Classical TwoBody ' startDate, '', stopDate, ' 60 J2000 ', epochDate, '',
num2str(orbit(1)), '', ...
num2str(orbit(2)), '', num2str(orbit(3)), '', num2str(orbit(4)), '', num2str(orbit(5)), ' 0.0'];
stkExec(conid, cmd);
% set ground site position
cmd = ['SetPosition */Facility/ground_site Geocentric ', num2str(site_lat), '', num2str(site_lon), ' 0.0'];
stkExec(conid, cmd);
% get access data, satellite to ground site
intervals = stkAccess('*/Satellite/sat1', '*/Facility/ground_site');
[n,m] = size(intervals);
% loop through access data, eliminating first partial day and last partial day,
% find the pass overhead - at perigee, the repeating pass once a sidereal
% day - that we want to use
% length of sidereal day in minutes
sid_day = 1436.068167;
perigee_pass = zeros(1,10);
for(j=1:10)
for(i=1:n)
start_min = intervals(i).start/60;
stop_min = intervals(i).stop/60;
if(start_min < j*sid_day && stop_min > j*sid_day)
perigee_pass(1,j) = stop_min - start_min;
end
end
end
% get average pass length
pass_time = mean(perigee_pass);
% find average eclipse time
[data, names] = stkReport('*/Satellite/sat1', 'Eclipse Times');
all_times = stkFindData(data{1}, 'Total Duration');
ecl_numbers = stkFindData(data{1}, 'Start Pass Number');
% for some reason ecl_numbers is not numeric, and won't convert right as an
% array, so convert one at a time
for(i=1:length(ecl_numbers))
ecl_n(i) = str2num(ecl_numbers(i,:));
end
% the eclipse report lists several lines of data for each eclipse; the
% total duration values are repeated, but we don't want duplicates - find
% uniques; i are the indices we want
[b, i, j] = unique(ecl_n);
unique_times = all_times(i);
% remove first and last times, in case they are partial eclipse values
% average the rest
len = length(unique_times);
eclipse = mean(unique_times(2:len-1))/60;
stkClose(conid);

```


Telecommunications Model

Next the power needed for each transmission can be found, as well as the amount of data that can be transmitted each pass (see MATLAB function *transmitter*). The following assumptions were made:

- The ground station's antenna has a diameter of 5 meters
- Each pass that should result in a transmission will be successful
- The time needed to initialize a transmission is two minutes

The power needed was found from figure 13-5 in *SMAD*. Since the tool is only designed for low-earth orbits, the line indicating 'Earth coverage case' with a 5 meter ground antenna was used. From visual examination of this graph, the power needed is approximately equal to 10 times the data rate in Mbps:

$$P_{trans} = 10 * (data_rate \text{ [Mbps]}) \text{ [W]}$$

For finding total data sent each pass, equation X.X from *SMAD* was used:

$$Data_Sent = data_rate * (F * Topass - Tinit) / M \text{ [bits]}$$

M represents how often transmissions are successful. Because of assumption (2) above, and because the amount of data sent on each successful transmission is what we want to calculate, M can be set to 1. $Tinit$ is the time needed to initialize each transmission, and has been set to 2 minutes, which is a typical value according to *SMAD*. $Topass$ is the time a pass directly over the ground site has line-of-sight access, and F alters this to account for passes that are not directly overhead. Since this tool is only using passes that are directly overhead, and STK calculates the line-of-sight time taking any slight variation into account, the pass time found by the STK module ($Tpass$) replaces this term. So the equation reduces to:

$$Data_Sent = data_rate * (Tpass - Tinit) \text{ [bits]}$$

```
transmitter.m
```

```
function [data_sent, power] = transmitter(dr, pass_time)
% INPUTS
% dr: data rate of data transmitted from spacecraft to groundsite, in bps
% pass_time: average time of each spacecraft pass where downlink is made (minutes)
% OUTPUTS
% data_sent: average amount of data that can be sent each pass (bits)
% power: power needed for each transmission (W)
% From figure 13-5 from SMAD
% power for different data rates
% for the ground station diameter of 5 meters, the plot indicates that
% power is equal to approximately 10x the data rate, in Mbps
power = 10*(dr/10e5);
% total data sent on each pass (on average)
data_sent = dr*(pass_time*60 - 2*60);
```

```
disp('Power needed for data transmission (W): ');  
disp(power);  
disp('Bits of data sent each transmission: ');  
disp(data_sent);
```

Useful References

Wertz and Larson, Space Mission Analysis and Design, Third Edition, El Segundo: Microcosm Press, 1999.

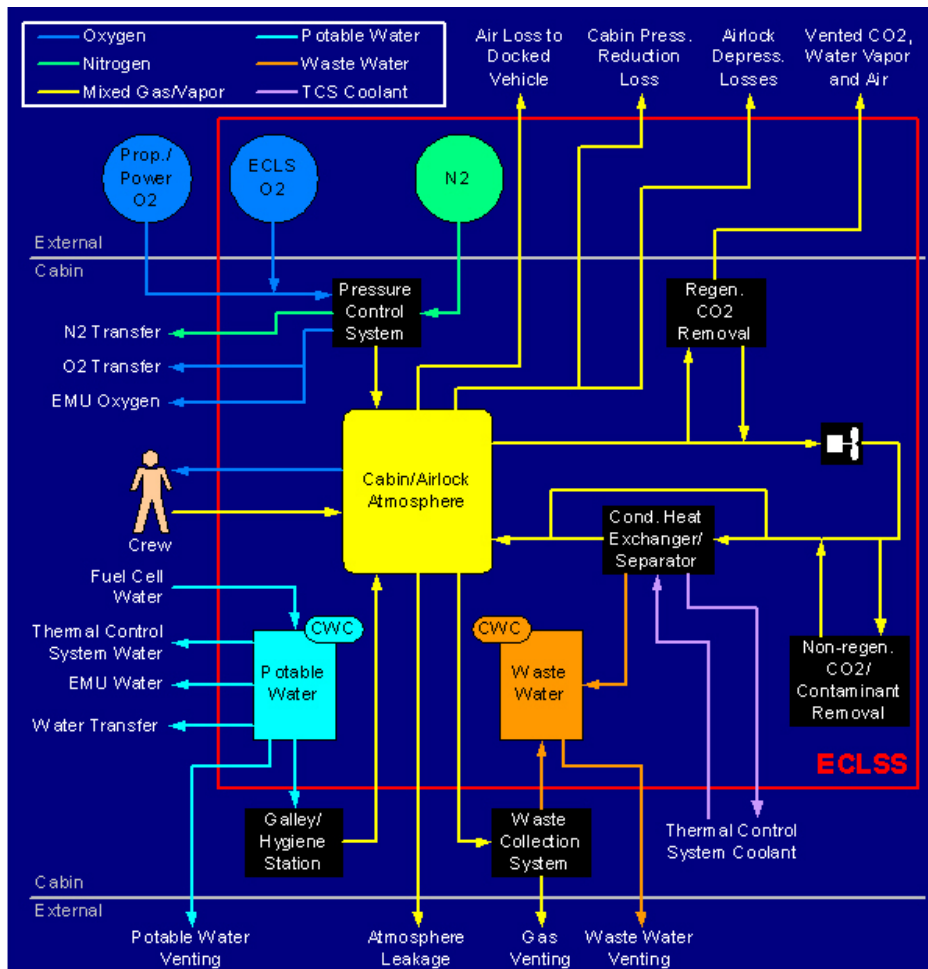
Vallado, Fundamentals of Astrodynamics and Applications, New York: McGraw-Hill Co, 1997.

13. Human Factors

Human assets in space provide an unparalleled situational awareness capability while greatly increasing mission cost and political risk. The Environment Control and Life Support System (ECLSS) is a complex subsystem composed of various components. Two important design metrics are power and system mass, but reliability, and safety are also vital factors that are difficult to enumerate. The ECLSS also has interdependencies on the power, thermal and structure subsystems.

Learning Objectives

Mission design for manned missions requires an understanding of the various life support subsystems.



Schematic diagram of Environment Control and Life Support System [Lawson, 2003]

The *Air Revitalization Subsystem* maintains the atmospheric environment, including pressure control, composition maintenance, and trace elements. In addition to interfacing with all of the other life support systems, it also is responsible for detecting and responding to fires.

The *Biomass Subsystem* produces and stores agricultural products for the Food Subsystem. The Biomass Subsystem also serves to regenerate air and water.

The *Food Subsystem* stores ready-to-eat, prepackaged foods, beverages, and ingredients included on the spacecraft at launch. It also receives agricultural products from the Biomass Subsystem and stores them for consumption as necessary.

The *Thermal Subsystem* maintains temperature and humidity ranges for the crew and rejects waste heat to the space environment as needed. It is assumed that the rejection of heat to the environment provides an adequate amount of air circulation.

The *Waste Subsystem* collects and processes solid waste material, including human waste, inedible biomass, and food packaging materials.

The *Water Subsystem* stores and distributes water as necessary for consumption, hygiene, and other purposes. The Water Subsystem is also responsible for collecting, transporting, and processing wastewater.

Reflections on Learning Experience

I enjoyed Professor Hoffman's lectures a great deal. His stories on the Hubble repair mission made for a truly memorable class. He also touched upon practical design issues for manned missions. These included more restrictive pressure, thermal, and radiation requirements, in addition to providing for bodily functions and a safe return to Earth. It wasn't until the sixth problem set when I worked on a life support module for a Mars mission that I implemented some of these principles in code.

ECLSS Design Principles

Two conflicting design principles exist. One approach is to minimize power use and technology development costs by using pre-existing flight-tested technologies, for example, technologies currently in use on the ISS, and the second approach is to implement what is sometimes referred to as an ALS, or advanced life support system. The idea behind using an ALS is to close the mass loop through the use of regenerative systems and hence to recover as much mass as possible over the mission duration. This can help to minimize initial launch mass and put less stress on the propulsion system. The development of ALS systems is especially important if current propulsion technology – i.e. Chemical rockets – are to be used for a Mars mission. For example, laundering clothing during the mission as opposed to launching enough clothing required for the entire mission would help to reduce launch mass.

In this problem set six we looked at possible ECLSS designs for a Mars-Transit vehicle. From an ECLSS design point of view, and indeed from a human mission point of view, the optimal solution is to minimize transfer time from Earth to Mars as much as possible. This would help not only from a mass point of view, but also to limit the adverse affects

of radiation (Mars transit involves possibly high levels of galactic cosmic rays and solar protons) and microgravity on the crew members. However, decreasing transfer time is only feasible up to a certain point after which the propellant mass becomes prohibitive.

Various methods can be used to compare technologies when designing an ECLSS system. The NASA/MSFC and McDonnell Douglas Method uses eight trade study parameters to compare the available systems: mass, power consumption, volume, resupply (for shorter missions), development potential, emergency operation, reliability and safety. The method we implemented in the sixth problem set is based on NASA's Equivalent System Mass (ESM) metric that expresses mass, volume, power consumption, cooling requirement and crew-time requirement in terms of a so-called "equivalent" mass. Development potential, emergency operation, reliability and safety are considered in selecting the technologies to compare via the ESM metric and are not expressed quantitatively. ESM is often used as a transportation cost measure in studies comparing Advanced Life Support Systems (ALS) since the cost to transport a payload is proportional to its mass [Levri, 2003].

In this analysis the power subsystem is being studied in detail, so the ESM equation has been modified to remove the power equivalency factor. Instead, estimated power consumption is determined for each subsystem and used as an input into the power module. Further analysis follows the discussion of ESM, although power values are included in description of ECLSS options for comparison purposes. ESM* refers to the version of ESM without power equivalency.

The equation used for the ESM is that developed in [Levri, 2003]. It is a sum of the ESM values over the ECLSS subsystems being considered. The subscript i indicates the index of the subsystem.

$$ESM^* = \sum_{i=1}^n \left[(M_i \cdot SF_i) + (V_i \cdot V_{eqi}) + (C_i \cdot C_{eqi}) + (CT_i \cdot D \cdot CT_{eqi}) + (M_{TDi} \cdot D \cdot SF_{TDi}) + (V_{TDi} \cdot D \cdot V_{eqi}) \right]$$

Mass equivalency factors are used to express the non-mass parameters volume, cooling requirement and crewtime in terms of mass units. For example, the volume equivalency is computed as the total empty mass of the spacecraft divided by the total volume capacity. Mass is considered to be a resource and each unit of volume (power, cooling, crewtime) is expressed in terms of the amount of mass it uses.

The rationale for using crewtime as an equivalency factor is explained in detail in [Levri, 2003]. Briefly, crewtime used in maintenance or operation of the life support system takes away from crewtime available to carry out mission objectives. If the life support system requires too much crewtime, the crew size would have to be increased to achieve scientific objectives and hence the life support system would be incrementally larger.

The following table describes the parameters involved in the ESM calculation for each subsystem of the ECLSS onboard the spacecraft.

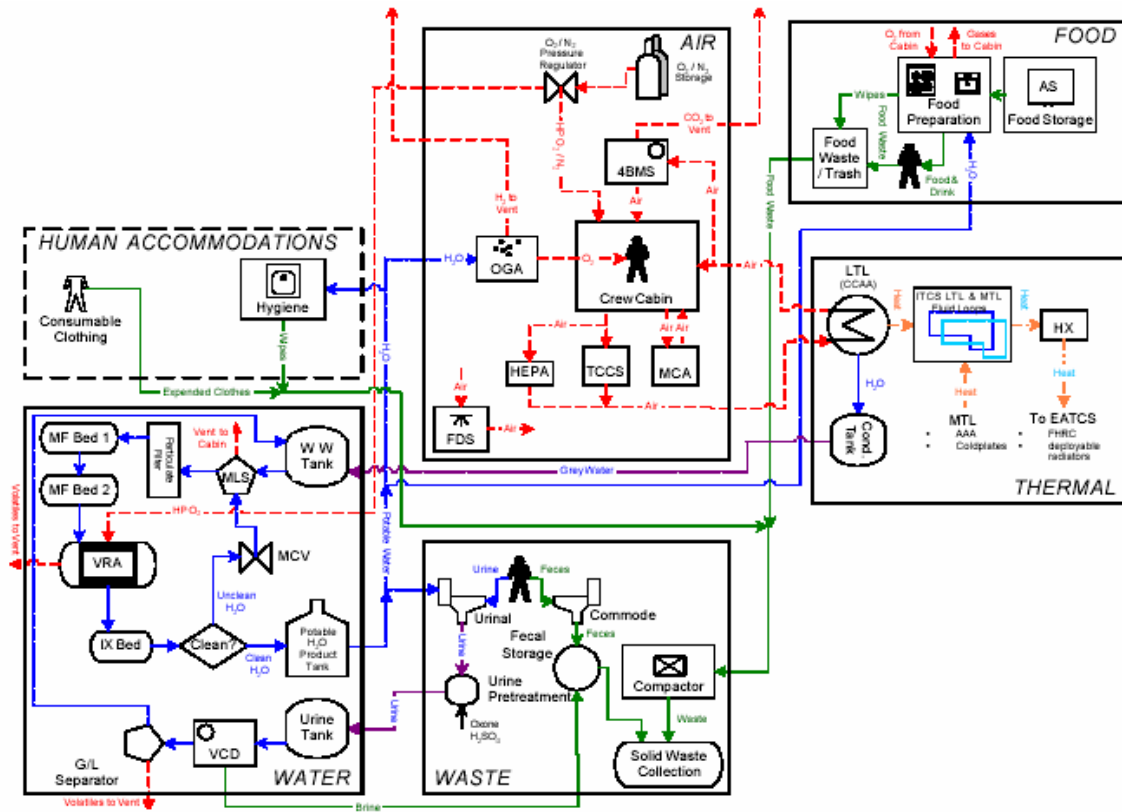
Parameters involved in ESM* equation.

Variable	Description	Units	Comments
M_{li}	Initial mass	Kg	
SF_{li}	Initial mass stowage factor	kg/kg	Accounts for additional hardware required to fasten and contain equipment
V_{li}	Initial volume	m^3	Any pressurized volume necessary for life support hardware
V_{eqi}	Mass equivalency factor for the pressurized volume support infrastructure	kg/m^3	
C_i	Cooling requirement	kW	Typically equivalent to subsystem power needs
C_{eqi}	Mass equivalency factor for the cooling infrastructure	kg/kW	
CT_i	Crewtime requirement	CM-h/y	Time spent by the crew in operation/maintenance
CT_{eqi}	Mass equivalency factor for the crewtime	kg/CM-h	
D	Duration of the mission segment of interest	y	Calculated in trajectory generation module (dependent on orbit)
M_{TDi}	Time/Event-dependent mass	kg/y	Neglected in this analysis
SF_{TDi}	Time/Event-dependent stowage factor	kg/kg	Neglected in this analysis
V_{TDi}	Time/Event-dependent volume	m^3	Neglected in this analysis

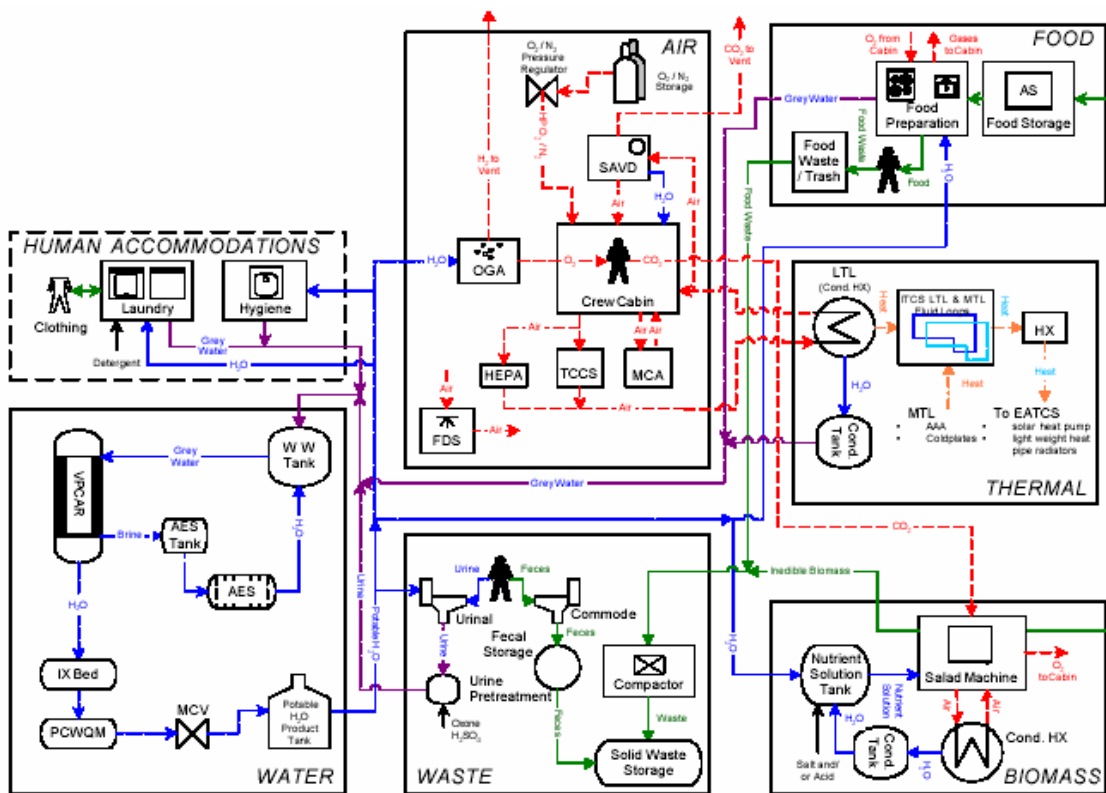
General notes and assumptions:

- Subsystem-specific equivalencies are not used [Levri, 2003]. The equivalency factors are instead constant over all subsystems for the mission segment under consideration. Since the volume equivalency factor depends on the amount of radiation shielding provided, it may vary according to subsystem if for example the plant-growth area has less shielding. A more detailed model of the desired vehicle design would be required to approximate this parameter. Similarly, if different power systems are used to power the various subsystems, the power equivalency should be variable.
- Time dependent mass, stowage factor and volume values are not used in the code. Incorporating these variables could be an extension to the current work.
- A crew size of 6 is assumed for equipment sizing.
- Assumed food consumption is 1.82 kg/CM-d plus 0.23 kg/CM-d of disposable packaging for a total packaged food mass of 2.05 kg/CM-d.
- The expected oxygen generation and carbon dioxide removal from the biomass chamber are not currently incorporated into the ALS air revitalization design.
- Plants grown in the biomass chamber could also be used for water regeneration through a transpiration process however this is not taken into consideration in this study.

Two types of systems are compared in the following analysis from our sixth problem set to develop a life support system for a Mars transit vehicle. The first is model of the ISS-Baseline approach to life support, and the second is a model of an Advanced Life Support system, which makes use of as many regenerative technologies as possible. This section of the report builds on previous work and thus detailed descriptions of the function of subsystem components will not be developed in detail here. The following two figures show schematics of the ISS-Baseline and ALS Mars Transfer vehicle ECLSS designs respectively.



Schematic diagram of Environment Control and Life Support System for a Mars Transfer Vehicle using ISS-Baseline Technology [Stafford, 2001]



Schematic diagram of Environment Control and Life Support System for a Mars Transfer Vehicle using ALS Technology [Stafford, 2001]

Air Revitalization Subsystem –ISS

The ISS air revitalization subsystem can be broken down into three main components: a CO₂ removal system, an O₂ generation system and a trace contaminant control system (TCCS). The function and operation of various implementations of these subsystems are discussed in detail in previous work as well as in [Lawson, 2003] and will not be explained in detail here. CO₂ removal on the ISS is accomplished via a 4-Bed Molecular Sieve (4BMS), electrolysis is used to generate molecular oxygen from oxygen-containing compounds available in the spacecraft (Solid Polymer Water Electrolysis SPWE), and the ISS Baseline TCCS system uses activated carbon to remove non-combustible gases and bacteria filters to remove particulate [Hanford, 2003].

ESM value for ISS Air Regeneration System

Option	Mass [kg]	Volume [m ³]	Power [kW]	Crew Time [ch/y]	ESM* [kg]
ISS	715	3.63	2.35	8.1	1643.65

Air Revitalization Subsystem –ALS

ALS options for an air revitalization subsystem were introduced in previous work and are as follows [Drysdale, 1999]:

1. 4-Bed Molecular Sieve (4BMS) + Sabatier Carbon Dioxide Removal System (CRS) + Solid Polymer Water Electrolysis (SWPE) Oxygen Generation System (OGS) + Node 3 Advanced TCCS (regenerable sorbent bed)
2. Sabatier CGS + SWPE OGS + Node 3 TCCS
3. Improved Carbon Dioxide Removal Assembly (CDRA) + Sabatier CGS + SPWE OGS + Node 3 TCCS
4. 4BMS + Bosch CRS + SPWE OGS + ISS Baseline TCCS

Comparison of ESM values for Air Regeneration System Options

Mission Duration 0.49y

Option	Mass [kg]	Volume [m3]	Power [kW]	Crew Time [ch/y]	ESM* [kg]
1	891	3.00	3.17	8.1	1732.2
2	893	1.60	2.39	8.1	1385.7
3	812	1.90	2.53	8.1	1377.8
4	783	2.99	3.33	8.1	1631.7

Mission Duration 0.73y (8.67months – Hohmann Transfer)

Option	Mass [kg]	Volume [m3]	Power [kW]	Crew Time [ch/y]	ESM* [kg]
1	891	3.00	3.17	8.1	1734.4

Using the ESM metric, various air-regeneration systems can be constructed and compared. With increased mission duration, there is minimal increase in equivalent mass since the only part of the ESM equation where this comes into play is the crewtime component. To minimize mass and power required, Option 2 with a reduced mission duration of 0.49y is the optimal choice.

Food and Biomass Subsystem-ISS

The food subsystem calculations assume that the majority of food is provided from Earth in the form of prepackaged dehydrated entrees. The food storage unit is approximated using values from an ISS freezer/refrigerator component [Hanford, 2002]. The ESM for food is proportional to mission duration since it is a consumable resource.

Comparison of ESM Values for Food System

System Component	ESM* [kg]
Refrigerator/Freezer	905
Prepackaged Food from Earth (D=0.49)	3719.2
Prepackaged Food from Earth (D=0.73)	5540.8

Food and Biomass Subsystem-ALS

For a long duration mission, salad greens and possibly other fresh vegetables such as potatoes could supplement packaged food if a small biomass chamber was included on board the transit vehicle. The equivalent system mass for this component takes into account grow-lamps and ballasts as well as the crops and the required growth space. Since this technology is still under development, a small unit of approximately 5m² is included in the ALS system. This growth space factor multiplies the ESM* baseline value for 1m² of space and also multiplies the power required. The addition of fresh produce to the astronauts' diet would probably be more of a psychological boost than a packaged meal replacement. However, with further study and testing it is possible that a more advanced biomass chamber could be added to a transit vehicle to increase the variety of foods available to the crew.

ESM Values for Biomass System

System Component	ESM* [kg]
Biochamber (5m ² potato, salad greens)	2433.9

Temperature and Humidity Control (THC) System-ISS/ALS

Temperature and humidity must be kept within a nominal range to ensure comfortable working conditions for the crew. Human metabolism as well as cabin equipment must be taken into account when calculating the amount of heat produced and hence the amount of cooling required. Temperature control is usually accomplished by removing heat from the atmosphere using a heat exchange with the excess heat eventually vented to space. Humidity control can be accomplished using a desiccant or phase change process. Ventilation and air circulation are also important concerns when designing the THC System.

The ISS baseline thermal control system details were difficult to find at the component level, so the entire THC system is considered as a unit. This system includes avionic air assemblies to cool equipment, cabin air assemblies for cooling and dehumidification of the crew quarters, and condensate storage/water flow loops for heat transport. Heat exchangers are considered to be part of the external Thermal Control System as opposed to part of the ECLSS.

ESM for Thermal Control System

Subsystem	Mass [kg]	Volume [m3]	Power [kW]	Crew Time [ch/y]	ESM* [kg]
Thermal	793	2.31	3.02	0	1734.4

Water Recovery System-ISS

The water recovery system must achieve loop closure approaching 100% to alleviate the need for resupply. No single system for water recovery has been designed to remove all contaminants or treat all types of wastewater that need to be processed. A trade study must therefore compare combinations of technologies that serve to fulfill all treatment requirements including storage, filtration or phase change processes, urine processing, water quality monitoring and disinfection. The ISS Baseline system uses Vapor Compression Distillation in combination with Multifiltration and Volatile Removal Assembly.

ESM for ISS Water Recovery System

Option	Mass [kg]	Volume [m3]	Power [kW]	Crew Time [ch/y]	ESM* [kg]
ISS	638	0.5	0.99	8.0	811.8

Water Recovery System-ALS

The ALS water treatment schemes offer a significant improvement in recovery over the ISS baseline model. For the systems outlined below, loop closure approaches 100% and hence the amount of water launched can be significantly reduced. The five combinations available for comparison according to ESM are as follows:

1. Vapor Compression Distillation (VCD) + Ultrafiltration/Reverse Osmosis (UF/RO) + Aqueous Phase Catalytic Oxidation Subsystem (APCOS)
2. VCD + UF/RO + Milli-Q Post Processor (MilliQ)
3. Biological Water Processor (BWP) + RO + MilliQ
4. BWP + RO + Air Evaporation Subsystem (AES) + MilliQ

Comparison of ESM values for ALS Water Recovery System Options

Option	Mass [kg]	Volume [m3]	Power [kW]	Crew Time [ch/y]	ESM* [kg]
1	263	0.5	1.56	8.0	471.0
2	229	0.5	0.72	8.0	386.6
3	141	0.5	0.68	8.0	296.2
4	186	0.5	1.68	8.0	401.2

All simulations are for nominal mission duration of 0.49 years. Varying the mission duration will not have a strong effect on the water recovery system since most of the associated costs are infrastructure related rather than consumable. From the above chart we can see that option 3 is currently the best from an ESM perspective. These values are approximate however since CrewTime and Volume data are not yet available for the technologies under study and therefore are set to be constant across all options.

Waste Processing System-ISS

The waste processing system involves the collection and storage of waste material. For shorter duration missions, storage is usually preferred to in-situ treatment, and hence the ISS waste processing system consists of toilet facilities and a storage tank. Again, since a detailed breakdown of power and mass values for individual components was unavailable, the system is treated and analyzed as a whole.

ESM for ISS-Baseline Waste Processing System

Subsystem	Mass [kg]	Volume [m³]	Power [kW]	Crew Time [ch/y]	ESM* [kg]
Waste	149	0.09	0.17	90	228.9

Waste Processing System-ALS

For a Mars transit mission, waste treatment may become an issue due to the long mission duration and the desire to recover as much material as possible from waste. Techniques to consider include incineration, dessication, freezing, heat sterilization and chemical treatment. The ALS system can either use the ISS-Baseline components for storage alone or add to this a treatment component. The only option considered for treatment is super-critical wet oxidization (SCWO), which is a physiochemical process used to treat waste. The process is able to breakdown waste products quite completely, with the resulting products being water vapor, carbon dioxide, nitrogen and salts. Other processes are likely to produce toxins and hence are not as viable for a Mars transit mission where safety and reliability are critical.

ESM for ALS Waste Processing System

Subsystem	Mass [kg]	Volume [m³]	Power [kW]	Crew Time [ch/y]	ESM* [kg]
Waste-ISS	149	0.09	0.17	90	228.9
SCWO	200	104	0.5	0.2	820.7

Human Accommodations-ISS

All clothing is assumed to be brought from Earth. The relationship used to calculate clothing mass is $1.5 * CrewSize * Mission\ Duration\ (days)$ [Hanford, 2003].

Human Accommodations-ALS

A smaller amount of clothing may be brought from Earth if there is an aqueous laundry available to the astronauts. Since water recovery is increasingly possible with an ALS system, it can be used to wash clothing and thus reduce the vehicle launch mass. With a laundry system available, clothing mass at launch can be estimated as $0.267 * CrewSize *$

Mission Duration (days) [Hanford, 2003]. Assume the crew do a load of laundry once a week, so the Crew Time value is multiplied by a factor of 52.

ESM for ALS Waste Processing System

Subsystem	Mass [kg]	Volume [m3]	Power [kW]	Crew Time [ch/y]	ESM* [kg]
Laundry	118	0.66	0.31	0.33*52	288.4

Logistics ISS/ALS

Because the ALS systems tend to have greater recycling capabilities, the amount of air, water and clothing that must be launched is lower for the ALS systems. This comparison is shown in Table 12 [Hanford, 2003] with values calculated for a mission duration of 0.49y.

Comparison of Logistics for ISS and ALS Systems

ECLSS	O ₂ and N ₂ [kg]	Water [kg]	Clothing [kg]
ISS-Baseline	294.9	863.87	1609.7
ALS	99.9	11.76	286.5

ECLSS Summary and Comparison

The ISS-Baseline approach to ECLSS design has the advantage of being technologically proven over a long-term mission duration, however, the mass of this system tends to be higher than the mass of an ECLSS that incorporates advanced regenerative technologies.

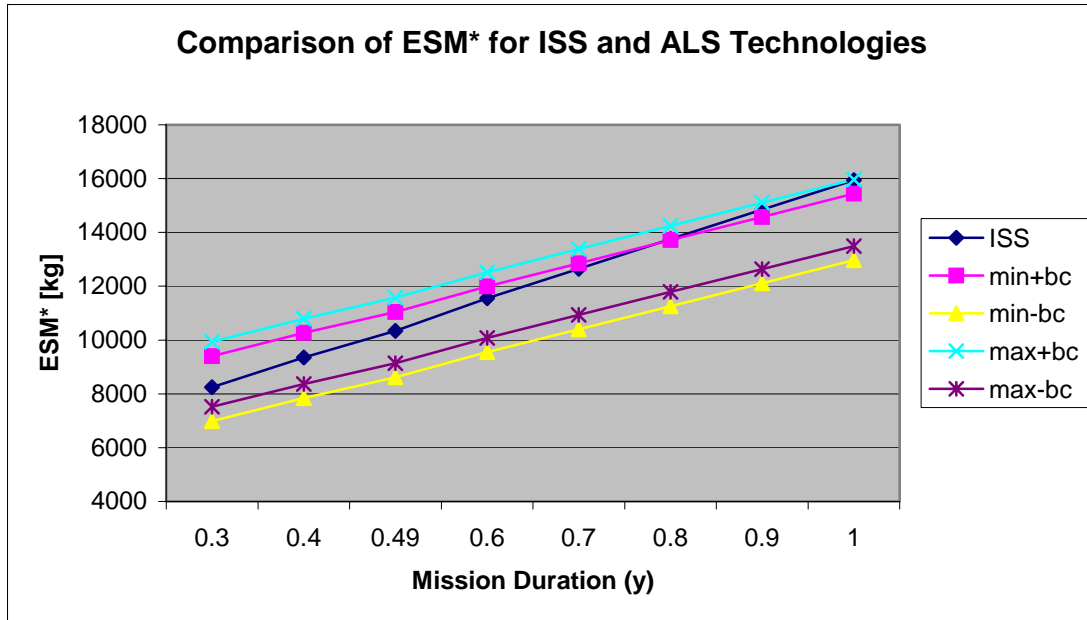
The biomass chamber is the single highest contributor to mass for the ALS system, and so the ALS is examined with and without a biomass chamber to show that in general, the ALS mass is less than that of the ISS (less-regenerative) system.

Comparison of Logistics for ISS and ALS Systems

ECLSS	ESM*[kg] (min)	ESM*[kg] (nominal)	ESM*[kg] (max)
ISS-Baseline		10339.66	
ALS	8609.258		9138.508
ALS+Biomass	11043.17		11572.42

Comparison cannot be done on the basis of mass alone however, as the ESM in this case does not include power. Power use of the different approaches must also be compared, as must technological development cost and reliability/safety. For a Mars mission scheduled to leave in the next two or three years, the ISS technologies would probably be preferable for the added safety relative to the minor mass improvement in using ALS.

The figure below shows a plot of the ISS and ALS ESM* as mission duration varies. The four ALS plots correspond to minimum ALS value with and without the biomass chamber and maximum ALS value with and without the biochamber respectively. As expected, the tendency to higher masses with longer mission durations is close to linear, and the ECLSS system design favours low mission duration in terms of mass.



Comparison of ESM* for ISS and ALS Systems

Challenges for a Manned Mars Mission

A manned flight to Mars also embodies a host of new safety challenges. For one, given the long duration of a mission to Mars, there is an increased chance of failure of dynamic systems such as pumps, valves, and electronics. These failures may be caused by the increases in operating times, on/off cycles, structural flexing, corrosion, and abrasion. Secondly, there is no opportunity to receive equipment or supplies from Earth, underscoring the need for reusable, robust systems. A third challenge is the increasing communications lag between the transportation vehicle and Mission Control as the vehicle approaches Mars. Once in a Mars orbit, this lag will vary from eight to forty minutes depending upon orbital positions. Data rates will also decrease dramatically, on the order of 10^{-6} compared to Earth orbiters. One obvious challenge facing the power system is the reduction in available solar energy as the vehicle approaches Mars. In fact, the power available to solar array will decrease by 50% from the start of the mission.

Useful References

Drake, Bret G. *Technologies for Human Space Exploration: Earth's Neighborhood and Beyond.* AIAA, 2001.

Drysdale, A. E. *Advanced Life Support Systems Modeling and Analysis Project Baseline Values and Assumptions Document.* NASA CTSD-ADV-371. Houston TX, 1999.

- Hanford, A. J. *Advanced Life Support Baseline Values and Assumptions Document*. NASA CTSD-ADV-484. Houston TX, 2002.
- Hanford, A. J. *Advanced Life Support Research and Technology Development Metric – Fiscal Year 2003*. Lockheed Martin Space Operations. Houston TX, 2003.
- Kennedy, Kriss J. “*The Vernacular of Space Architecture.*” AIAA, 2002.
- Lawson, Mike. *Advanced Life Support*. NASA Johnson Space Center. Updated August 2003. <http://advlifesupport.jsc.nasa.gov>
- Levri, Julie. *Advanced Life Support Equivalent System Mass Guidelines Document*. NASA/TM-2003-2112278. Washington DC , 2003.
- Oliver, Fernando Abilleira. “*Mars Generations Mission, a Manned Mission to Mars.*” AIAA, 2002.
- Peterson, Donald H. “*Safety Concepts for a Crewed Mars Mission.*” AIAA, 2000.
- Stafford, Kristin W. “*Advanced Life Support, Systems Integration, Modeling, and Analysis Reference Missions Document.*” Lockheed Martin Space Operations—Crew and Thermal Systems Division. 5 November 2001.
http://peer1.nasaprs.com/peer_review/prog/RMD.pdf (30 November 2003).
- Wertz, James R. and Larson, Willey J., *Space Mission Analysis and Design*, Third Edition, Microcosm Press, 1999.
- Zubrin, Robert. *The Case for Mars*. Touchstone, New York, 1997.

14. Cost Modeling

Detailed bottom-up estimating is accurate for established programs but it is a weak method for immature designs with a high number of technical uncertainties. Analogy-based estimating is the application of an existing cost model to a similar system, adjusted for size and complexity differences. When known physical, technical, and performance parameters can be related to cost, the parametric costing method is best for conducting conceptual designs under time constraints, and is the preferred method of the Department of Defense. In the real world, Professor Miller points out that cost is the “size of the standing army times the length of the program.”

Learning Objectives

Understand the following concepts covered in lecture:

- Cost Estimating Methods
- Work Breakdown Structure
- Software Cost Estimation
- Technology Readiness Levels
- Management Reserves
- Production Costs
- Annual Funding Profiles
- Inflation
- Cost Analysis Requirements Document

Reflections on Learning Experience

Chapter 20 on cost in *SMAD* is an excellent introduction to cost estimating for space systems. I enjoyed lecture and the coverage of the material in the context of the ‘Resource Plan’ for the Stellar Interferometer Tracking Experiment.

Software Cost Estimation

SMAD estimates that flight software and ground software RDTE&E cost \$435 and \$220, respectively. These estimates are then multiplied by the language factor:

Language Factor	
Ada	1.00
UNIX-C	1.67
PASCAL	1.25
FORTTRAN	0.91

Technology Readiness Levels

Technology Readiness Level	Definition	Risk	Std Dev (%)
1	Basic principles observed	High	>25
2	Conceptual design formulated	High	>25
3	Conceptual design tested analytically or experimentally	Moderate	20-25
4	Critical function/characteristic demonstrated	Moderate	15-20
5	Component or breadboard tested in relevant environment	Moderate	10-15
6	Prototype/engineering model tested in relevant environment	Low	<10
7	Engineering model tested in space	Low	<10
8	Full operational capability	Low	<10

Useful References

Space Mission Analysis and Design (SMAD), Wertz & Larson, Third edition, Microcosm Press, 1999.

Reducing Space Mission Cost , Wertz and Larson

International Reference Guide to Space Launch Systems , Isakowitz, AIAA

Jane's Space Directory

Cost Models (Aerospace Corporation Small Satellite Cost Model (SSCM), Air Force

Unmanned Spacecraft Cost Model (USCM), NASA Goddard Multivariable Instrument

Cost Model (MICM), NASA World Wide Web sites)

<http://www.jsc.nasa.gov/bu2/guidelines.html>

15. Structures

The structures subsystem mechanically supports all spacecraft subsystems, mates the spacecraft to the launch vehicle, and is responsible for ordnance-activated separation. The structures subsystem must meet strength and stiffness requirements of all primary loads and secondary loads such as wire bundles and propellant lines.

Learning Objectives

The learning objectives identified in lecture include understanding:

- The role of the thermal subsystem in shielding the payload from extreme thermal and radiation environments, maintaining system geometry, and carrying loads
- Power and thermal management applications
- Issues associated with light-weight structures and thermal distortion
- Multifunctional structural technologies
- Deployment and geometry maintenance of deployable booms, mesh antennas, membrane structures, inflatables, and tethers

Reflections on Learning Experience

SMAD (11.6) and lecture were both clear on this subject. I especially enjoyed the discussion of multifunctional structures and advanced materials and composites. The structures subsystem is difficult to trade and hence was not a popular subsystem in the problem set modules.

Thermal Issues with Structures

(Taken from lecture notes)

- Sunshields
 - To observe in the thermal infrared requires cold optics and detectors
 - Sunshields are used to block sunlight from heating these elements
 - Need to be large and lightweight
- Thermal Snap
 - The heat load into a structure can change due to Earth eclipse in LEO or due to a slew of the S/C
 - Nonzero or differential coefficient of thermal expansion (CTE) can cause stresses to build
 - Friction joints in deployment mechanisms can eventually slip causing an impulsive input
 - This high frequency vibration can disturb precision instruments
- Thermal Flutter
 - Differential thermal expansion can cause a portion of the structure to curve and reduce its exposure to a heat source
 - The structure then curves back thereby increasing its heat load

- This can lead to a low frequency instability (flutter)
- Thermal Distortions
Differential thermal expansion in optics and optical mounts can dramatically degrade performance
Kinematic mounts ensure that only 6-DOF loads are applied thereby holding the optic's 6-DOF in place without applying bending and shearing loads

Useful References

Space Mission Analysis and Design (SMAD), Wertz & Larson, Third edition, Microcosm Press, 1999.

Fundamentals of Space Systems, Pisacane and Moore, Oxford, 1994

Space Systems Engineering, Fortesque and Stark, John Wiley and Sons, 1995

Space Vehicle Design, Griffin and French, AIAA 1991

MEMS for Inflatable Space Structures article:

<http://www.spacedaily.com/news/nanotech-01h.html>