RICE UNIVERSITY

Analysis of Low Frequency Whistler Wave Occurrences in the Night-Side Venus Ionosphere.

by

Harold Glenn White

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE

Doctor of Philosophy

APPROVED, THESIS COMMITTEE:

[Signatures and names of committee members]

HOUSTON, TEXAS

MAY 2008
INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.
ABSTRACT

Analysis of Low Frequency Whistler Wave Occurrences
in the Night-Side Venus Ionosphere.

by

Harold Glenn White

This body of work deals with a detailed analysis of plasma, magnetic, and electric field
data from Pioneer Venus Orbiter (PVO) to determine if the data are consistent with the
possibility of lightning on Venus. This has been a strong topic of debate in the planetary
physics community. In a recent Nature article [Ingersoll, 2007], Ingersoll provides a
synopsis of the case against lightning on Venus. He states that there should not be
lightning on Venus, whose clouds, at roughly 55-60 km above the surface, are like
terrestrial smog clouds, which do not produce lightning. Ingersoll then goes on to
recount that no visible evidence (flashes) has been detected on the night or day-side of
of high frequency radio waves characteristic of terrestrial lightning (0.125 to 16MHz)
during Cassini’s two fly-bys of Venus, contrasted with the definite detection of RF waves
during Cassini’s later earth fly-by. However, the detection of low frequency whistler
waves by Venus Express has revived claims that the source of these whistler waves is
lightning in the lower atmosphere of Venus [Russell, et al., 2007]. Numerous other
papers have been published on different aspects of the debate, such as a paper addressing
telemetry interference being incorrectly interpreted as evidence for lightning [Taylor, et al., 1988], another paper suggesting the detected events are a local phenomenon [Taylor, et al., 1983], and a paper documenting some of the optical searches for Venusian lightning [Taylor, et al., 1994]. This work is a comprehensive reconsideration of 14 years of PVO plasma data on a season by season basis, as the spacecraft goes to low altitudes on the night-side of Venus. In this effort, intelligent software filters have been developed to find, sort, and analyze the frequency of occurrence of low frequency whistler waves. The results of this investigation show that the source cannot be in the lower atmosphere of Venus, since at the lowest altitudes (140-156km), the signals disappear. Therefore, an ionospheric source for these whistler waves must be considered.
ACKNOWLEDGEMENTS:

The author would like to thank Dr. Cloutier for his support and technical help along the way to ensure that this is a solid body of scientific work. The author would like to thank his wife for her eternal patience and frequent words of encouragement, his kids for being excellent sources of playful distractions, and the Good Lord for granting me wisdom, insight, and patience to see this effort through to conclusion.
# TABLE OF CONTENTS

1.0 BACKGROUND: PVO MISSION ................................................. 1

2.0 INTRODUCTION .................................................................. 5

2.1 VENUS IONOSPHERIC TROUGHS ......................................... 10

3.0 DATA DISCUSSION AND TOOLS DEVELOPMENTS ............ 18

3.1 TROUGH IDENTIFICATION FILTER ..................................... 21

3.2 EVENT IDENTIFICATION FILTER ........................................ 30

3.3 INVESTIGATION PLAN ....................................................... 34

4.0 DATA FINDINGS ............................................................... 40

5.0 CONCLUSIONS ................................................................ 53

6.0 REFERENCES .................................................................... 60

A.0 APPENDIX (CODE) ........................................................... 63
1.0 Background: PVO mission

The Pioneer Venus Orbiter (PVO) was launched on May 20, 1978 and reached orbit around Venus on December 4, 1978. PVO remained in orbit and gathering data for 14 years (5055 orbits, ~24 earth-hours each) before it exhausted its propellant and burned up in the atmosphere of Venus in October of 1992. The orbital spacecraft was equipped with a total of 17 experiments used to analyze conditions around Venus. Of these 17 experiments, several provided critical insight about the plasma conditions in the Venus ionosphere. The Orbiter Electric Field Detector (OEFD) used a self-contained balanced V-type antenna with a differential preamplifier to sample the electric field magnitude ~every 0.250 seconds at four 30% narrow band channels of 100Hz, 730Hz, 5400Hz, and 30000Hz. The Orbiter Magnetometer (OMAG) used a triaxial fluxgate magnetometer at the end of a long boom to sample the magnitude and orientation of the magnetic field around Venus. The Orbiter Electron Temperature Probe (OETP) used a pair of cylindrical Langmuir probes (two were required to ensure that one was always out of the wake of the spinning spacecraft) to measure the electron density and temperature of the ionosphere. The Orbiter Ion Mass Spectrometer (OIMS) used a Bennett radio-frequency mass spectrometer (retarding potential analyzer) to determine the composition and concentration of thermal positive ions in the ionosphere of Venus.
Figure 1-1: Artist's Impression of PVO around Venus (courtesy NASA)

Figure 1-2: Synopsis of PVO experiments and systems layout (courtesy NASA)
Figure 1-2 shows the configuration of the PVO spacecraft. The OMAG/boom is in the top left of the diagram, the OEFD V-type antenna can be seen just to the right, The OETP is further clockwise at roughly the four-o’clock position, and finally the OLMS can be seen at the nine-o’clock position.

![Pioneer Venus Orbiter Mission Diagram](image)

**Figure 1-3:** PVO elliptic orbit evolution (courtesy NASA)

PVO was established in a 24 earth-hour elliptical orbit around Venus. Initially, it had a periapsis in the 150km range, and its apoapsis in the 66,000 km range. From December of 1978 until July of 1980, the spacecraft’s periapsis was maintained in the 142 to 253 km range to accommodate radar and ionospheric measurements. After this time, the
periapsis was allowed to reach a maximum of 2290 km. In 1991, the Radar Mapper was reactivated to facilitate investigations of the southern region of the planet, and the spacecraft's periapsis was again held in the 150-250km range until the spacecraft exhausted its propellant and was burned up in the atmosphere. During this final low-altitude periapsis phase, the OMAG had stopped taking three-axis magnetic field data. Figure 1-3 shows a representation of the elliptic orbit.
2.0 Introduction

Shortly after arriving in orbit around Venus, the PVO’s OEFD sensor began detecting impulsive events. One category of events is where the event is broadband in nature and is detected in all four channels of the OEFD. The other category of events is where the event only registers in the 100Hz and/or the 730Hz channels. These latter events were typically detected while the orbiter was on the night-side of the planet, and at low altitudes. These latter impulsive events are viewed by some researchers as being the result of lightning occurring in the lower atmosphere [Scarf et al., 1980; Scarf and Russell, 1983; Singh and Russell, 1986; Russell et al., 1989]. Other researchers have adopted the view that the source of these events is some form of plasma instability [Taylor et al., 1985; Taylor and Cloutier, 1986; Taylor et al., 1987; Walker, 1992; Walker, 1993].

The idea of lightning on Venus actually predates the PVO mission dating back to as early as 1956 [Kraus, 1956a, b, c]. In this study, short duration, spiky radio signals in the 11 meter wavelength were observed from Venus, and Kraus reasoned that these events may have been generated by lightning. This was later refuted by Ksanfomaliti [1983] as being generated from the strongly-heated surface of the planet and lower layers of the tropopause.

In one of the first paper to attribute the PVO’s OEFD events to lightning [Scarf et al., 1980], the detection of the low frequency whistler waves in the form of short duration,
impulsive events in the 100Hz and 730Hz OEFD channels was likened to terrestrial whistlers triggered by lightning that then propagate up from the lower atmosphere along the magnetic field lines. The paper concluded that in the Venus case, this would only be possible on the night-side of Venus where the plasma densities were very low, and the magnetic field was both strong and radial in sense. It was noted that no signals were expected in the higher OEFD channels (5400Hz, 30000Hz), as these waves would be attenuated and not propagate upward the ionosphere.

_Borucki et al. [1981]_ documents an optical search of Venus for indications of lightning in the optical wavelength using the PVO’s star sensor. This study produced a null result in that it was unable to distinguish events that could be indications of lightning and events that were false alarms. This data was used to put a conservative upper limit on the flash rate and this was determined to not be greater than the event rate on earth. There was a subsequent optical search done using ground assets by _Borucki et al., [1981]_ that found no optical evidence of lightning.

In _Scarf and Russell [1983]_, the paper examined 100Hz events from the first 1185 orbits of PVO. The paper concluded that the detection of events could be correlated to PVO’s ground track over the highland regions of the planet and might be related to volcanic events (lightning generated by eruption plumes). This was countered by a series of papers by Taylor and collaborators [Taylor et al., 1985; Taylor et al., 1986; Taylor and Cloutier, 1986; Taylor et al., 1987] that examined the data and found no such correlation existed.
Singh and Russell [1986] reconsidered PVO data using a change in the definition of an event where it must be detected in all four OEDF channels. While the upper frequencies are not expected to be able to propagate, it was reasoned that there may be patchy areas of plasma density that would allow this. The paper considered the first, second, fourth, and fifth night-side seasons. The paper concluded that the number of detected events decreased with altitude suggesting that the source was atmospheric. Taylor and Cloutier [1987] released a study of the paper's claims and noted that the events were cataloged in altitude bins with respect to their time from periapsis. Since the periapsis changed greatly during the fourth and fifth night-side seasons, this introduced an error into the event history, precluding the absolute conclusion that the observed event rate dropped as altitude increased.

In a later paper by Taylor and Cloutier [1988], it was discovered that the distribution of broadband events reported in Singh and Russell [1986] was nearly identical to the distribution of telemetry data quality index flags recorded by NASA tracking stations and was completely different from the distribution of real data impulsive events. The conclusion was that telemetry errors had been mistakenly identified as impulsive broadband events due to Venus lightning.

This prompted a refined effort by Singh and Russell documented in [Russell and Singh, 1989] where they acknowledged the telemetry errors and also noted that the original data set no longer existed precluding any further study. They claimed to have corrected the
detection errors and the paper arrived at similar conclusions to the 1986 paper. Again, further consideration by Taylor and Cloutier [1991] found that there were events corresponding to telemetry spikes in the data set rendering the conclusions of the Russell and Singh paper suspect.

In Taylor et al. [1987], a more detailed analysis of the events reported by Scarf and Russell was performed to determine any dependency on the presence of troughs. This study concluded that nearly all of the events reported were associated with ion troughs. This paper also performed another consideration of the subsatellite position for each event to see if there was any dependency on ground topology. No correlation to the highland regions was found. An event rate for the events recorded was derived and provided the first statistical suggestion that the event rate increases in magnitude as the altitude decreases to ~200km, then the event rate drops sharply below this altitude. This was the first statistical indication that the event source may not be in the lower atmosphere.

Borucki et al. [1991] documents a second optical search for lightning using the same approach as was done for the first optical search. This search was performed using the PVO’s star sensor. The search had more viewing time and also covered the controversial highland regions conjectured to be sources of volcanic activity (and hence possibly sources of lightning). Again, this study produced a null result in that it was unable to distinguish events that could be indications of lightning and events that were false alarms.
This data was used to put a lower conservative upper limit on the flash rate and this was determined to not be greater than one-half the event rate on earth.

In an article published in Nature by Gurnett [Gurnett, D. A. et al. Nature 409, 313-315, 2001], Gurnett details an effort to detect impulsive high frequency radio waves (0.125 to 16MHz) using the Radio and Plasma Wave Science (RPWS) instrument on board the Cassini spacecraft during its two gravity-assisted fly-bys of Venus. On earth, these signals are characteristic of terrestrial lightning, and can be heard on AM band radios (envision the WWII radio tech hovering in front of his AM transceiver radio hearing the background whistling noise on the speaker). On Venus, spherics generated by lightning would have to have frequencies above the plasma frequency which is ~ 1MHz or less on the night-side and ~5MHz on the day-side [Brace et al., 1983]. The first fly-by covered the night-side, and the second fly-by covered the day-side. The two fly-bys covered effectively 61% of the planet surface. Neither of these fly-bys detected impulsive events above the background noise. In August of 1999, Cassini made a fly-by of Earth, and a study of the RPWS data clearly detected impulsive lightning-related events.

As noted in the abstract, the detection of low frequency whistler waves by Venus Express has provided new data that has started the debate again anew which provided the impetus to go back and fully reconsider the largest archive of low-altitude plasma data gathered during PVO's fourteen year mission to the planet.
2.1 Venus Ionospheric Troughs

Figure 2.1-1 shows a sketch of the Venus ionospheric interaction with the solar wind from Brace et al. [1983]. The orbit track of PVO is shown as it passes through its orbital periapsis on the night-side of Venus. There are two features labeled “Hole” on the night-side of Venus. These represent regions of plasma density depletion in the ionosphere, and are typically referred to as ion troughs. Grebowsk and Curtis [1981] proposed that these observed depletions in the nightside ionosphere are made possible due to electric fields acting parallel to the solar magnetic field imbedded in the Venus ionosphere. They do not provide specific details about this mechanism, rather they suggest, using a terrestrial analogy, that neutral sheet acceleration processes will result in a parallel electric field that is on the order of kilovolts. This electric field will accelerate ionospheric thermal-ions out to the magneto-tail, and accelerate energetic electrons in from the magneto-tail to serve as a night-time ionization and heat source.

Let us also discuss the mechanism whereby the solar magnetic field gets imbedded into the Venus ionosphere, and how the shape of the field lines evolve in this process. At Venus, the solar magnetic field is effectively azimuthal with respect to the orbit of Venus around the sun. Figure 2.1-2 depicts the convection of the solar magnetic field through the ionosphere of Venus. The mechanism behind this magnetic field draping and formation of a magnetic tail with a neutral body was first described by Alfvén [1957] as applied to the formation of comet tails. In the case for Venus, gravity keeps the neutrals at low altitudes thus maintaining an ionosphere. Figure 2.1-3 shows some interplanetary
flux tubes approaching the planet (panel a). As the flux tubes begin to penetrate the bow-shock up-stream of the planet, they are slowed down and compressed (panel b). The flux tubes are further slowed while interacting with the magnetosheath (ionosheath) and are eventually deflected by the ionosphere. This further enhances the draping effect as outside the ionosheath, the free ends of the flux tubes get carried downstream of the planet by the solar wind. The imbedded solar magnetic field provides a magnetic pressure which can then equalize with the surrounding plasma thermal pressure \([Brace, et al., 1982]\), \(B^2/2\mu_0 = n_e kT\), hence creating a local hole in the ionosphere that generally points upward from the planet surface (but not always purely radial). It is these troughs that are the locations of interest to look for low frequency whistler waves (impulsive events), be their source lightning, or plasma instabilities.
Figure 2.1-1: Sketch of ionosphere and PVO orbit track [Brace et al., 1983].
Figure 2.1-2: Progression of solar magnetic field through Venus Ionosphere [Brace et al., 1980].

Figure 2.1-3: Depiction of draping of flux tubes to form magnetic tail [Alfvén, 1957]
Figure 2.1-4 shows a local effect described in *Law and Cloutier* [1997] as weathervaning of the interplanetary magnetic field as it diffuses through the ionosphere. For each of the depicted solid field lines associated with regions A, B, and C, there is a dashed line that indicates where the line would be if it were not imbedded in the ionosphere. The three plots inset in the figure show the angle of rotation of the imbedded magnetic field and the axisymmetric nightward plasma flow. The angle of rotation is more pronounced at the noon meridian (regions A and B), and there is little rotation at the magnetic equator (region C).

**Figure 2.1-4**: Depiction of weathervaning [Law and Cloutier, 1997]
Figure 2.1-5 is a collation of data from the OEFD (100Hz channel), OMAG, OETP, and ephemeris data for orbit 530. This is the standard orbit typically picked in the literature to visually display the trough phenomenon as detected by PVO. In this figure, the PVO altitude can be seen, and the onset of the Venus ionosphere is apparent with the large jump in electron number density. The electron number density continues to increase as PVO gets lower in the ionosphere. Prior to the spacecraft reaching periapsis, there is a rapid drop in the electron number density, almost by an order of magnitude. Conversely, the magnetic field experiences a corresponding jump in magnitude at the leading edge of the trough. This is what is commonly identified as a trough, or hole, in the ionosphere. There is a partner to this trough on the outbound path of the spacecraft as its altitude increases. This outbound trough also shares the same characteristics as the inbound trough where the electron density decreases, and the magnetic field increases. Additionally, this outbound trough sees an increase in electron temperature which allows for a slightly larger relative electron number density drop in the trough.

This particular orbit has two well defined troughs, but detailed analysis of the condition of the plasma also indicates the vestiges of a trough (or perhaps just not well defined) just to the right of the spacecraft’s periapsis. This may not be apparent here, but conditional measures that were developed to help intelligent software to reliably recognize troughs clearly indicate there are issues with the stability of the plasma here as well. It is theorized that troughs should occur in north-south pairs [Brace et al., 1982], but since PVO’s orbit is not strictly polar (105 degrees) the spacecraft can experience 1, 2, or more. In the plot, the behavior of the 100Hz channel data while the spacecraft is inside the
ionospheric troughs (labelled) is indicative of the low frequency whistler waves being detected.
Figure 2.1.5: Synchronized PVO data from OEFD, OMAAG, OEIP, and ephemeris data. Ionospheric troughs are labelled.
3.0 Data Discussion and Tools Development

In order to do a comprehensive reconsideration of all 14 years of PVO data with the prime objective of detecting and cataloging events in troughs, some thought had to be given about which orbits should be considered and why. We developed a software tool to analyze the ephemeris data to gather and plot the periapsis history for the spacecraft over all 5055 orbits. These results are shown in Figure 3-1. We developed another software tool to determine for which orbits the spacecraft went below 1000km, and had a solar zenith angle greater than 110 degrees. Providing an altitude filter of 1000km ensures that the orbiter is in the ionosphere, and the high solar zenith angle ensures that it is on the night-side of Venus. These filtered orbits are shown in red overlaid atop the overall history. As can be seen, there are eight distinct collections of orbits where PVO's periapsis occurred while over the night-side of Venus. These eight seasons are labeled. Another technical point of note is that the OMAG stopped gathering three-axis data after orbit 3602.
Figure 3-1: PVO Periapsis History – night-side orbits are depicted in red.
As can be seen, there are eight distinct periods, or seasons, when the PVO’s periapsis occurs over the night-side of Venus.
Based on this analysis of PVO orbital characteristics, the first, second, and third nightside seasons were chosen for further analytical consideration. Season 4 was considered, but that season’s minimum periapsis was deemed to be too high to be useful in the process of considering the frequency of detected events to determine if a lower atmospheric source is plausible. Season 8 might seem to be another logical choice due to its low periapsis altitude, but it must be noted that the magnetometer had stopped taking three-axis data at this point and no magnetic field information is available for this season. As was determined during the development of the intelligent software algorithms, the magnetic field information is necessary to reliably detect the ionospheric troughs, as we will show in a moment.
3.1 Trough Identification Filter

While casual consideration of Figure 2.1-3 would suggest that detection of ionospheric troughs is easy owing to the visual appearance of the electron density curve, this is due to the fact that the human eye/brain were purpose-built to pick out patterns with ease. Creating a machine algorithm to reliably detect ionospheric troughs over a broad range of orbits and conditions is not so trivial a task.

Within the first three seasons that were down-selected from the original list of eight, there are two sinusoidal variations in the periapsis altitude of the space craft. Figure 3.1-1 shows a close up view of these first three seasons. There is the longer period variation that has a period of roughly ~120 orbits. There is a shorter period variation that is on the order of 7 orbits. Both of these orbit periapsis altitude variations will affect the measured electron density when comparing one orbit to the next. Clearly the dynamics of the atmosphere/ionosphere will also greatly contribute to variations on a 24 hour orbit-basis. All this is to say that basing a filter on any absolute values of electron density to flag a trough will likely yield areas as troughs that are not troughs.
Figure 3.1-1: Close up view of PVO Periapsis History focusing on the first three night-side seasons of the spacecraft.
Another approach to developing a filter might be to create an algorithm to look for large relative changes in electron densities (say a half to full order of magnitude). The change in electron density associated with a trough for orbit 530 appears to decrease over a 100 second period. The return to normal also appears to occur over a 100 second period. The second trough appears to have a slightly quicker decrease and increase at ~70 seconds each. This could be due to the size of the trough, and if the spacecraft has only a “glancing blow” of the boundary that defines the trough. Also, within each trough event, the value of the electron density at the beginning of the trough is different from the value at the end of the trough since the spacecraft is at different altitudes.

Since this type of filter would be developed to declare a leading or trailing edge of a trough, a higher level logic would have to track if a leading edge flag has been declared prior to acknowledging a trailing edge flag. Aside from logic, this type of an algorithm would need to be tuned to evaluate sensitivity to ensure that physical parameters used to define the logic conditions were reliable. This is not impossible, but likely to result in a complicated filter with many conditions and cases to try and anticipate all eventualities.

Some thought was also given to making use of magnetic field and temperature data as well. However, by themselves, these parameters will have similar programming logic challenges as noted above. In reviewing Walker [1992] and Walker [1993], and subsequent discussions with Dr. Cloutier, events from Scarf and Russell [1983] considered by Walker showed some correlation to low-beta plasma conditions. In other words, the events occurred primarily in low-beta plasma regions. We decided to consider
the inverse of this calculated parameter as a possible conditional measure from which we could establish a logic flag. Figure 3.1-2 shows the relevant plasma data for orbit 530, along with the calculated parameter:

\[
\beta^{-1} = \frac{B^2}{2\mu_0 n_e kT}
\]

In the plot, beta-inverse shows a clear dependency with the troughs discussed earlier. A limit is drawn at roughly a value of beta-inverse = 5 as an example of how this calculated parameter can be established as a Boolean flag to automatically and reliably identify a trough. It should also be noted that this parameter also picks up some activity on the outbound leg of the orbit right after periapsis. This is not a strong instability, but a good data point to show how the parameter could be made to pick this up by only modifying an analog threshold.
Figure 3.1-2: Orbit 530 plasma data, and calculated parameter beta-inverse
Some additional plots for other orbits are provided to indicate the ability of the algorithm to identify a trough. Note in Figure 3.1-4 that two troughs occur after periapsis, but the beta-inverse flag clearly detects these. Figure 3.1-5 for orbit 527 shows that there was only one well-defined trough seen. Careful consideration of beta-inverse for Figure 3.1-3 (middle of well-defined troughs) and Figure 3.1-4 (right of well-defined troughs) show that there are additional regions that could be the remnants/beginnings of a trough, or just not a well defined trough.
**Figure 3.1-3:** Orbit 526 plasma data, and calculated parameter beta-inverse
Figure 3.1-4: Orbit 529 plasma data, and calculated parameter beta-inverse
Figure 3.1-5: Orbit 527 synchronized data, showing parameter beta-inversed.
3.2 Event Identification Filter

A study by Scarf et al., [1980], noted that low frequency whistler waves can only propagate through a magnetized plasma when the index of refraction is a real number. This can be determined from the following equation

$$n^2 = n^2(f, \theta) = 1 + \frac{f_p^2}{f(f_e \cos(\theta) - f)}$$

Here $f_p$ is plasma frequency, $f_e$ is the electron cyclotron frequency, and $\theta$ is the angle of propagation of the wave to the magnetic field. For $\theta=0^\circ$, the index of refraction is real for two scenarios. Either the wave frequency is greater than the plasma frequency, or the wave frequency is less than the electron cyclotron frequency. Conditions in the night-side ionosphere were noted to have plasma frequencies far above 30kHz, so PVO would not see any whistler waves in this category. The electron cyclotron frequencies on the night-side of Venus are such that whistlers would only be detected in the 100Hz and 730 Hz channels. (We confirmed the plasma frequency cutoff levels and electron cyclotron characteristics in our automated analysis of the PVO data. We will defer discussion of that to the data discussion section.)

Based on the above discussion, genuine whistler wave events are only expected to be seen in the 100Hz and 730Hz channels of the OEFD. With that, the algorithm was constructed with a mask filter that would mask the 100Hz and 730Hz channels for one second should an event be seen on either the 5400Hz or 30000Hz channels. As the events are expected to be impulsive in nature, the filter was written to look for rapid
changes in relative magnitude in a 0.250 second period. The magnitude was set at 1.5, 1.0, and 0.5 to evaluate sensitivity. In the end, the value of 0.50 was chosen as the preferred value for use with all three seasons. Figure 3.2-1 and Figure 3.2-2 show the results of the filter for orbit 530. This also includes the previous conditions of being below 1000km, and inside an ionospheric trough. Note in the latter figure that the filter properly does not flag the periodic events seen outside the troughs, rather only the impulsive events while the PVO is inside the troughs.
Figure 3.2-1: Orbit 530 events seen during inbound trough, or Trough 1
Figure 3.2-2: Orbit 530 events seen during outbound trough, or Trough 2
3.3 Investigation Plan

As part of the process of analyzing the three critical seasons of PVO historical data, a local archive was constructed with a logical folder structure better suited for automated analysis than currently exists on the National Space Science Data Center (NSSDC). The NSSDC archive for the PVO is maintained in the Planetary Data System: Planetary Plasma Interactions website that is maintained by University of California (UCLA). CDs are available by request, and the data is also maintained on the website in the same folder structure as is available on the numerous numbers of CDs. The manner in which experiment data was subdivided into folders was not standardized and as such not well suited for the task of automated analysis. This is not to say that the data was not being properly maintained, rather it seem to be a consequence of the computing tools and storage capacities available at the time the PVO was still in service. For example, subfolders of data within a specific instrument parent folder were not subdivided into logical groups of say 100 orbits, rather the number was inconsistent from folder to folder: 53 in one, 131 in another. Several Matlab scripts were written and tailored to this directory structure on the NSSDC web server to automatically download the relevant experiment files into a simplified directory structure on a local hard-drive more conducive to automated analysis. This served two purposes, first this helped with programming of the intelligent software, second, having the data on a local hard drive made for faster software execution times. The scripts that downloaded the data from the web server took considerably long execution times and were often left to run overnight.
The actual investigation effort (the part after downloading the data and creating the local
data archive structure) can be broken into three computationally intensive phases, and a
final manual post-processing phase that performs the final analysis of the automated
analysis data.

Figure 3.3-1 shows a flow diagram of the preliminary analysis phase of the investigation
effort. This phase is the initial analysis of the ephemeris data to establish what subsets of
orbits should be considered and why. For example, the full periapsis history of the
spacecraft was assembled by the ephem_alt_filter.m script analyzing all the ephemeris
data files which allowed us to construct Figure 3.1 and Figure 3.1-1. The data provided
by the ephem_filter.m script is the part that is overlaid in red on those figures. It is this
data that helped us determine which orbits to consider and why. As is already
documented, this led us to focus on the first three night-side seasons of the spacecraft.

Figure 3.3-1: Preliminary Analysis Phase of Investigation Plan
Figure 3.3-2: Computationally Intensive Phase of Investigation Plan

Figure 3.3-2 shows a flow diagram for the computationally intensive phase. The OETP_fullup.m script brings in all relevant experiment data for a selected orbit, synchronizes the data to a common time stamp (the spacecraft did not have a standardized delta-t for any of the experiments, and in some cases delta-t changes on the fly for a specific parameter), then calculates the beta-inverse parameter for the ionosphere, flagging troughs for the orbit.

This script’s secondary function was to gather good electron density data to determine three effective scale heights, one for the 140-190km range, one for the 190-250km range,
and one for the 250-400km range. Recall that scale height is the distance over which the pressure decreases by a factor of $e$. Consider the following equation for pressure as a function of height:

$$P(h) = P(h_0) e^{-(h-h_0)/H_p}$$

$P(h_0)$ is a reference pressure at a reference height $h_0$, $P(h)$ is the pressure at any height $h$, and $H_p$ is the scale height. We can consider two discrete points in the atmosphere and develop the solution for $H_p$

$$H_p = \frac{h_2 - h_1}{\ln\left(\frac{P(h_1)}{P(h_2)}\right)}$$

This is just the slope of the altitude versus the natural log of the pressure. The script enhances the approach by gathering multiple data points and fits a curve to the data population to derive a slope. This is done for each orbit, the slope is then multiplied by the square of the calculated correlation coefficient $R^2$, the sum of these products is then divided by the sum of the squares of the correlation coefficient to produce a statistical weighted average. The motivation to perform this statistical study of the pressure scale height is that the results can be used to develop the magnetic field as a function of altitude through the relationship: $B^2(h)/2\mu_0 = P(h)$. This can then be used to independently calculate the electron cyclotron frequency everywhere along the field line.

The OEFD.m filter takes the output from the OETP_fillup.m and analyzes the selected window for detection of events on the 100Hz, and 730Hz channels, masking as necessary
for noise on the 5400Hz and 30000Hz channels. The final script considers the windows of interest and integrates the time at altitude and time in trough for the spacecraft. This is the proper time to use to derive an event rate as the events tabulated were detected during these windows.

Figure 3.3-3 shows the flow diagram for the post-processing phase. This part principally enhances the event tabulation file by going back and gathering the altitude and magnetic field for the event. The altitude is needed to tabulate the events by bin so an event rate can be calculated. The magnetic field is needed to be able to calculate the electron cyclotron frequency for the given altitude and magnetic field where the event was detected. The average electron densities are calculated for both inside and outside troughs for a given season so that the plasma frequency can be calculated. After the post processing phase, the data was imported into Microsoft Excel for the final manual analysis phase.
Figure 3.3-3: Post Processing Phase of Investigation Plan
4.0 Data Findings

Figure 4.1 shows the event rate as a function of altitude for season one orbits. The altitude bins are 16km in height starting at 140 km. The labels in the chart are the midpoints of the bins. Each solid bar is the calculated event rate determined by taking the total number of events reported in that altitude bin (from count.txt file, see Figure 3.3-3), and dividing that number by the integral of the time the spacecraft spent in that altitude bin, while in a trough (from mod_season_bin_count.txt, see Figure 3.3-2). The time the spacecraft spent in any particular altitude bin, in a trough, is co-plotted on this chart in dashed bars. The event rate increases in magnitude as the spacecraft goes lower in altitude, until the lowest altitude bin where the rate rapidly drops to ~0. If these events are the result of an atmospheric source well below the spacecraft, then the anticipated minimum event rate for the lowest altitude bin (140-156km) would be no less than the event rate from the immediate adjacent upper bin (156-172km). This conservative minimum event-rate scenario for a lower atmospheric source can then be multiplied by the integrated time the orbiter spent in this lowest altitude bin while in a trough to get an expected event count commensurate with this conjecture source. The event rate in the 156-172km bin is 57 events per hour, and the orbiter spent 2069 seconds in troughs in the 140-156km bin meaning for a lower atmospheric source we expect to see a total event count in this bin of roughly 33 events. The automated analysis only detected 1 event in this bin. This indicates that the events detected are not likely to be coming from beneath the spacecraft, and hence not from a lower atmospheric source (such as lightning).
Figure 4-1: Season 1 Event Rate for half-order magnitude events
The events were also separated by channel, and the electron cyclotron frequency for the local plasma conditions was calculated for each event:

\[ f = \frac{1}{2\pi} \frac{q_e B}{m_e} \]

The cyclotron frequency was then plotted versus the magnitude of the detected event. Trend lines were added to the data, but the broadband nature of the data population makes any conclusions about the dependency of event magnitude to electron cyclotron frequency problematic. However, it can be noted that there are many more events in the 100Hz channel than the 730 Hz channel. Recall the discussion earlier that the waves can only propagate if their frequency is above the plasma frequency, or less than the cyclotron frequency. Since the population has an electron cyclotron frequency range from just above 200Hz up to the 1100Hz range, the plasma conditions during some events would have precluded the 730Hz (30% broadband) from detecting the wave as the wave would had to have been at a frequency too low for the 730Hz channel to detect.
Figure 4-2: Electron Cyclotron Frequency for all Season 1 events, categorized by channel, and plotted versus detected magnitude.
The data results from min_ne_seasonX.m script can be used to calculate the plasma frequency for the ionosphere, inside and outside troughs for altitudes < 156km, and altitudes < 190 km using:

\[ \omega_{pe} = \sqrt{\frac{n_e q_e^2}{m_e \varepsilon_0}} \]

Where \( n_e \) is the electron density, \( q_e \) is the electron charge, \( m_e \) is the electron mass, \( \varepsilon_0 \) is the permittivity of free space, and \( \omega_{pe} \) is the electron plasma frequency in radians/sec.

For Season 1, the average electron density inside troughs (\( \beta^I > 3 \)) for altitudes below 156 km is \( \langle n_e \rangle = 1234 \text{ #cm}^{-3} \), which gives a plasma frequency of 315 kHz. The average electron density inside troughs for altitudes below 190km is \( \langle n_e \rangle = 2014 \text{ #cm}^{-3} \), which gives a plasma frequency of 403 kHz. The same exercise is repeated for electron density outside troughs (\( \beta^I < 1 \) for altitudes below 156 km is \( \langle n_e \rangle = 32191 \text{ #cm}^{-3} \), which gives a plasma frequency of 1610 kHz. The average electron density outside troughs for altitudes below 190km is \( \langle n_e \rangle = 18987 \text{ #cm}^{-3} \), which gives a plasma frequency of 1200 kHz. Thus all electromagnetic waves above the plasma frequency, and hence permitted to propagate in this lower area of the ionosphere, are outside the PVO's OEFD range of detection.

The season 2 study produced a family of data in general agreement with the season 1 family of data. Figure 4-3 shows the derived event rate for season 2 for half-order magnitude events. Similar to the approach taken for season 1, an expected minimum event count for the lowest altitude bin (140-156km) can be calculated for the lower atmospheric source (lightning). The spacecraft spent 551 seconds in the lowest altitude
bin, in troughs. Using the event rate for the 156-172km bin of 61 events per hour, if the source were from the lower atmosphere, we would have expected to see an event count of roughly 9 events. There were none detected. These data results continue to suggest that the source of the detected events cannot be from the lower atmosphere.
Figure 4-3: Season 2 Event Rate for half-order magnitude events
Figure 4-4: Electron Cyclotron Frequency for all Season 2 events, categorized by channel, and plotted versus detected magnitude
Figure 4-4 shows the electron cyclotron frequency versus magnitude. The conditions of the plasma in the troughs yields similar data to season 1, supporting the fact that most of the data would be detected in the 100Hz channel. The plasma frequency can also be considered. For Season 2, the average electron density inside troughs ($\beta^I > 3$) for altitudes below 156 km is $<n_e> = 11700 \text{ #cm}^{-3}$, which gives a plasma frequency of 971 kHz. The average electron density inside troughs for altitudes below 190km is $<n_e> = 2002 \text{ #cm}^{-3}$, which gives a plasma frequency of 407.6 kHz. It should be noted that there was one data point inside a trough below 190km with an electron density of $<n_e> = 9 \text{ #cm}^{-3}$, which gives a plasma frequency of 9kHz. The same exercise is repeated for electron density outside troughs ($\beta^I < 1$) for altitudes below 156 km is $<n_e> = 24254 \text{ #cm}^{-3}$, which gives a plasma frequency of 1398 kHz. The average electron density outside troughs for altitudes below 190km is $<n_e> = 18486 \text{ #cm}^{-3}$, which gives a plasma frequency of 1220 kHz. Outside of the one data point, all electromagnetic waves above the plasma frequency in this lower area of the ionosphere are outside the PVO’s OEFD range of detection.

The final season 3 study also produced a family of data in general agreement with the other two seasons of data. Figure 4-5 shows the derived event rate for season 3 for half-order magnitude events. Continuing with the approach taken for seasons 1 and 2, an expected minimum event count for the lowest altitude bin (140-156km) can be calculated for the lower atmospheric source (lightning). The spacecraft spent 1500 seconds in the lowest altitude bin, in troughs. Using the event rate for the 156-172km bin of 35 events
per hour, if the source were from the lower atmosphere, we would have expected to see
an event count of roughly 14 events, yet there were none detected.
Figure 4-5: Season 3 Event Rate for half-order magnitude events
Figure 4-6: Electron Cyclotron Frequency for all Season 3 events, categorized by channel, and plotted versus detected magnitude.
Figure 4-6 shows the electron cyclotron frequency versus magnitude. The conditions of the plasma in the troughs yields similar data to the first two seasons, supporting the fact that most of the data would be detected in the 100Hz channel. The plasma frequency can also be considered. For Season 3, the average electron density inside troughs ($\beta^1>3$) for altitudes below 156 km is $<n_e> = 9869 \text{ cm}^{-3}$, which gives a plasma frequency of 892 kHz. The average electron density inside troughs for altitudes below 190km is $<n_e> = 3116 \text{ cm}^{-3}$, which gives a plasma frequency of 501 kHz. It should be noted that there was one data point inside a trough below 190km with an electron density of $<n_e> = 8 \text{ cm}^{-3}$, which gives a plasma frequency of 25.7 kHz. The same exercise is repeated for electron density outside troughs ($\beta^1<1$) for altitudes below 156 km is $<n_e> = 29583 \text{ cm}^{-3}$, which gives a plasma frequency of 1544 kHz. The average electron density outside troughs for altitudes below 190km is $<n_e> = 16385 \text{ cm}^{-3}$, which gives a plasma frequency of 1149 kHz. Outside of the one data point, all electromagnetic waves above the plasma frequency in this lower area of the ionosphere are outside the PVO’s OEFD range of detection.
5.0 Conclusions

From the preponderance of data provided by this analysis, some reasonably firm conclusions can be drawn. First, the data confirms the characteristics of the ionosphere are such that transverse electromagnetic waves above the plasma frequency cannot (barring a few isolated data points) be detected by the PVO’s OEFD experiment. Second, the data confirms that the electron cyclotron frequency for detected events in the 140 to 400km altitude bin ranges from \(~200\)Hz to \(~1100\)Hz, meaning that the preponderance of whistler waves are expected to be seen (and are) in the 100Hz channel over the 730Hz channel.

The most significant conclusion that can be drawn is that the calculated event rate derived from PVO’s plasma data for seasons 1, 2, and 3 shows that the event rate goes to zero rapidly below the 156km altitude level. This is shown to be true in each season. We also calculated an expected minimum event count (for an atmospheric source) for that lowest altitude bin (140-156km) based on the time the spacecraft spent in that altitude bin while inside troughs. This is motivated by the historical position that these events are linked to a source in the lower atmosphere such as lightning. If this is the case, then the event rate in the 156-172km bin should be equal to or less than the expected event rate in the 140-156km bin. To that end, the event rate in the 156-172km bin can be multiplied by the PVO dwell time in the lower altitude bin (140-156km) while in a trough to calculate a minimum expected count of events. This was shown in all three cases to be much larger
than the observed rate that was almost zero for the entire period over all three seasons (only one event detected).

As a matter of a sanity check, we decided to reconsider season 3 with a broader plasma filter. In effect, we turned off the beta flag and simply cataloged the total number of events detected, using the same event filter logic of masking during 5400Hz or 30000Hz noise. This was done to consider whistler waves that may occur due to off-nominal scenarios such as 1) trough is ill-defined and event occurs at edge, 2) trough is tilted and energetic solar particle from magnetic tail tunnels off axis to trough’s magnetic field vector and triggers event outside of trough, and 3) ensure that more events can be detected in lowest altitude bin as shown in Taylor et al., [1987]. This last point is more to address the robustness of the algorithm as compared to a historical data source. The downside to this historical comparison is that the exact parameters that were used to detect and catalog the “lightning wave” events used therein are lost to history and no longer available. As such, we have at the very least normalize our modified event rate to the historical maximum rate shown in the noted reference. Figure 5-1 shows the comparison of the two data sets. They both show the characteristic drop off of event rate at the lowest altitude bin of 140-156 km, and a similar normalized magnitude. This also supports the conclusion we drew above that these events are a local event, and not from a source in the lower atmosphere such as lightning.
Figure 5-1: Season 3 modified plasma filter event rate normalized to the Harry Taylor event rate [Taylor, et al. 1987] calculated using the historical Russell "lightning wave" events
It should be noted that in order for whistler mode waves to propagate, there must be a collective response of the electrons in response to the $v \times B$ of the Lorentz force. When the collision frequency of electrons with ions (or neutrals) becomes higher than the electron cyclotron frequency, this inhibits the collective response of the electrons (i.e. the electrons become demagnetized). Once this happens, the only possible electromagnetic waves that can propagate in the medium are transverse electromagnetic waves above the plasma frequency. Recall that the plasma frequency is all but beyond the PVO's OEFD frequency range. However, as noted in the abstract, in an article published in Nature by Gurnett [Gurnett, et al., 2001], Gurnett details the non-detection of high frequency radio waves characteristic of terrestrial lightning (0.125 to 16MHz) during Cassini's two fly-bys of Venus, contrasted with the definite detection of these RF waves during Cassini's later earth fly-by.

If these events are not from a source in the lower atmosphere, then a local ionospheric source should be considered. In Walker [1993], it was shown that there were not enough currents within the troughs to support a two-stream instability. She also showed that the ion acoustic instability might account for a few events, the variability of trough conditions were not universally stable to produce 100Hz waves. She showed that the gentle-bump instability cannot produce 100Hz waves, and that all Alfven waves would not propagate. Finally, she showed that for the lower-hybrid drift instability, the most likely candidate reviewed in her effort, the frequencies generated by this instability would be too low to be detected by the OEFD.
Another local ionospheric plasma source could be a Kennel-Petschek electron cyclotron wave instability associated with a loss-cone that triggers scattering of energetic particles. Consider the scenario of energetic electrons propagating down the magnetic field lines of the ionospheric troughs towards the planet coming in from the magneto-tail as proposed by Grebowsk and Curtis [1981], and discussed in the section detailing the characteristics of the ion troughs. Each particle will be rotating around the field lines with a given pitch angle, where the pitch angle is defined as the angle between the particle’s velocity and the magnetic field vector. At or near the exobase at ~140km where the electron and ion temperatures converge for the night-side, some of the electrons will mirror producing a flux of upward bound electrons that have an initial pitch angle distribution of only 90 degrees. All other pitch angles for this upward bound flux of electrons are empty at this point. In this scenario, particles with parallel velocities which put them in cyclotron resonance with electron cyclotron waves propagating parallel to B cause instability and growth of these cyclotron waves, which in turn scatter particles into the loss cone by changing the particles’ perpendicular energy. The resulting scattering fills the loss cone, removing the anisotropy. That said, just because there is a pitch angle anisotropy, this does not guarantee that the cyclotron waves will go unstable. A detailed analysis of this has been done in Kennel and Petschek [1966], yielding the following equation for the anisotropy parameter:
\[
A_e = \frac{\int_v v_0 \, dv_0 \left( v_{0l} \frac{\partial F_0}{\partial v_{0l}} - v_{0l} \frac{\partial F_0}{\partial v_{0l}} \right) v_{0l}}{2 \int_{0}^{\infty} F_0 v_{0l} \, dv_{0l}} \bigg|_{v_{0l} = \frac{-\omega_{ce} - w}{k_{ll}}} 
\]

Where the criterion for instability is such that:

\[
A_e > \frac{1}{\frac{\omega_{ce}}{\omega} - 1} 
\]

Note that this does not mean that the instability process will always be present if there is a loss cone. Just because there is a loss cone does not always mean that this mechanism will be in effect. The denominator comes from the resonance condition:

\[
\omega - k_{ll} v_{ll} - |\omega_{ce}| = 0 \Rightarrow k_{ll} v_{ll} = \frac{|\omega_{ce}|}{\omega} - 1 
\]

In the Venus case, the data suggests that the generation of waves does not occur at all altitudes. If it did, then the event rate as a function of altitude would increase with altitude as the events detected in upper bins would be the local events \textit{and} the events that have propagated up from the lower bin. This is not seen in the data. The data suggests that the generation of these waves occurs within a slab of ionosphere ranging from 150-180km, and then the events that are detected in upper bins are the remnants of these lower events after some have been absorbed. This should manifest itself as an exponential decay in the observation rate dependent upon the scale height. In Figure 5-2, several exponential curves for different scale heights are shown compared to the normalized season 1 observed event rate. As a matter of future work, a numeric analysis
approach for wave generation and wave growth is being undertaken to derive a higher
fidelity predicted event rate as a function of altitude.

Figure 5-2: Event Detection Decay Rate predictions for several scale heights compared
to a normalized event rate from season 1
6.0 References

Alfvén, H., On the Theory of Comet Tails, Tellus, 9, 92, 1957


Kraus, J. D., Impulsive Radio Signals from the Planet Venus, Nature, 178,33,1956a

Krauss, J. D., Radio Observations of the Planet Venus at a Wave-length of 11m, Nature, 178, 103, 1956b

Krauss, J. D., Class II Radio Signals from Venus at a Wave-length of 11 Meters, Nature, 178, 159, 1956c

Krauss, J. D., Recent Observations of Radio Signals from Venus at 11 Meters Wave Length, Astron J., 21, 1957


Walker, S., Plasma instabilities in the Venus nightside ionosphere, Thesis, Rice University, 1992

Walker, S., Source of Impulsive 100Hz Electric Field Signals Detected in the Nightside Ionosphere of Venus, Dissertation, Rice University, 1993
Appendix

The software used to download and analyze PVO data is presented in this appendix for documentation purposes. The individual scripts are taken and inserted in the form that they existed during compilation of this report. Any subsequent efforts by others to use this software should ensure that all data paths are corrected for any new data archive configurations and data files and formats used. The investigation plan shows the interdependency of the individual software files and the order of execution. As a general note, most of the scripts were written to parse the historical data in a seasonal manner, as is evident when reviewing the code.
OMAG Download Script

OMAG_download.m
% download CMAG data

\%s1 = sprintf('d:\\PVO\\ephem_filter.txt');
s1 = sprintf('d:\\PVO\\season 7 orbits.txt');

% open file for input of data
fid0 = fopen(s1,'r');

orbit = fscanf(fid0, '%d %f %f %f', [4,inf]);
orbit = orbit';
[m,n] = size(orbit)

for i = 1:m;
    bg = orbit(i,1)
    if (bg>=1 & bg<=9) \%0001
        s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB00%d.ffd',bg);
        s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0001/DATA/OMAG/24S/ORB00%d.FFD',bg);
    end
    if (bg>=10 & bg<=40) \%0001
        s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB00%d.ffd',bg);
        s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0001/DATA/OMAG/24S/ORB00%d.FFD',bg);
    end
    if (bg>=41 & bg<=80) \%0002
        s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB00%d.ffd',bg);
        s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0002/DATA/OMAG/24S/ORB00%d.FFD',bg);
    end
    if (bg>=81 & bg<=99) \%0003
        s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB00%d.ffd',bg);
        s2 = sprintf('http://www.igpp.ucla.edu/cache2/PV01_0003/DATA/OMAG/24S/ORB00%d.FFD',bg);
    end
    if (bg>=100 & bg<=110) \%0003
        s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB0%d.ffd',bg);
        s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0003/DATA/OMAG/24S/ORB0%d.FFD',bg);
    end
    if (bg>=111 & bg<=150) \%0004
        s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB0%d.ffd',bg);
        s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0004/DATA/OMAG/24S/ORB0%d.FFD',bg);
end

if (bg>=151 & bg<=190) %0005
    s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB0%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0005/DATA/OMAG/24S/ORB0%d.FFD',bg);
end

if (bg>=191 & bg<=251) %0006
    s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB0%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0006/DATA/OMAG/24S/ORB0%d.FFD',bg);
end

if (bg>=282 & bg<=350) %0007
    s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB0%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0007/DATA/OMAG/24S/ORB0%d.FFD',bg);
end

if (bg>=351 & bg<=390) %0008
    s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB0%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0008/DATA/OMAG/24S/ORB0%d.FFD',bg);
end

if (bg>=391 & bg<=430) %0009
    s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB0%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0009/DATA/OMAG/24S/ORB0%d.FFD',bg);
end

if (bg>=431 & bg<=460) %0010
    s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB0%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0010/DATA/OMAG/24S/ORB0%d.FFD',bg);
end

if (bg>=461 & bg<=500) %0011
    s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB0%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0011/DATA/OMAG/24S/ORB0%d.FFD',bg);
end

if (bg>=501 & bg<=540) %0012
    s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB0%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0012/DATA/OMAG/24S/ORB0%d.FFD',bg);
end
if (bg>=541 & bg<=570)%0013
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB0%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0013/DATA/OMAG/24S/ORB0%d.ffd',bg);
end

if (bg>=571 & bg<=590)%0014
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB0%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0014/DATA/OMAG/24S/ORB0%d.ffd',bg);
end

if (bg>=591 & bg<=620)%0015
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB0%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0015/DATA/OMAG/24S/ORB0%d.ffd',bg);
end

if (bg>=621 & bg<=660)%0016
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB0%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0016/DATA/OMAG/24S/ORB0%d.ffd',bg);
end

if (bg>=661 & bg<=740)%0017
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB0%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0017/DATA/OMAG/24S/ORB0%d.ffd',bg);
end

if (bg>=741 & bg<=800)%0018
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB0%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0018/DATA/OMAG/24S/ORB0%d.ffd',bg);
end

if (bg>=801 & bg<=900)%0019
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB0%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0019/DATA/OMAG/24S/ORB0%d.ffd',bg);
end

if (bg>=901 & bg<=990)%0020
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB0%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0020/DATA/OMAG/24S/ORB0%d.ffd',bg);
end
if (bg>=991 & bg<=999) %0021
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cachel/PV01_0021/DATA/OMAG/24S/ORB%d.'
.ffd',bg);
end
if (bg>=1000 & bg<=1050) %0022
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cachel/PV01_0021/DATA/OMAG/24S/ORB%d.'
.FFD',bg);
end
if (bg>=1051 & bg<=1080) %0023
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cachel/PV01_0022/DATA/OMAG/24S/ORB%d.'
.FFD',bg);
end
if (bg>=1081 & bg<=1120) %0024
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cachel/PV01_0023/DATA/OMAG/24S/ORB%d.'
.FFD',bg);
end
if (bg>=1121 & bg<=1160) %0025
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cachel/PV01_0024/DATA/OMAG/24S/ORB%d.'
.FFD',bg);
end
if (bg>=1161 & bg<=1200) %0026
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cachel/PV01_0025/DATA/OMAG/24S/ORB%d.'
.FFD',bg);
end
if (bg>=1201 & bg<=1240) %0027
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cachel/PV01_0026/DATA/OMAG/24S/ORB%d.'
.FFD',bg);
end
if (bg>=1241 & bg<=1280) %0028
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cachel/PV01_0027/DATA/OMAG/24S/ORB%d.'
.FFD',bg);
end
if (bg>=1281 & bg<=1330) %0029
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cachel/PV01_0028/DATA/OMAG/24S/ORB%d.'
.FFD',bg);
end
s2 = sprintf('http://www.igpp.ucla.edu/cache1/TV01_0028/DATA/OMAG/24S/ORB%d.FFD', bg);
end

if (bg>=1331 & bg<=1400) %0029
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB%d.ffd', bg);
s2 =
    sprintf('http://www.igpp.ucla.edu/cache1/TV01_0029/DATA/OMAG/24S/ORB%d. FFD', bg);
end

if (bg>=1401 & bg<=1510) %0030
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB%d.ffd', bg);
s2 =
    sprintf('http://www.igpp.ucla.edu/cache1/TV01_0030/DATA/OMAG/24S/ORB%d. FFD', bg);
end

if (bg>=1511 & bg<=1580) %0031
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB%d.ffd', bg);
s2 =
    sprintf('http://www.igpp.ucla.edu/cache1/TV01_0031/DATA/OMAG/24S/ORB%d. FFD', bg);
end

if (bg>=1581 & bg<=1620) %0032
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB%d.ffd', bg);
s2 =
    sprintf('http://www.igpp.ucla.edu/cache1/TV01_0032/DATA/OMAG/24S/ORB%d. FFD', bg);
end

if (bg>=1621 & bg<=1680) %0033
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB%d.ffd', bg);
s2 =
    sprintf('http://www.igpp.ucla.edu/cache1/TV01_0033/DATA/OMAG/24S/ORB%d. FFD', bg);
end

if (bg>=1681 & bg<=1730) %0034
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB%d.ffd', bg);
s2 =
    sprintf('http://www.igpp.ucla.edu/cache1/TV01_0034/DATA/OMAG/24S/ORB%d. FFD', bg);
end

if (bg>=1731 & bg<=1770) %0035
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB%d.ffd', bg);
s2 =
    sprintf('http://www.igpp.ucla.edu/cache1/TV01_0035/DATA/OMAG/24S/ORB%d. FFD', bg);
end

if (bg>=1771 & bg<=1810) %0036
    s1 = sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB%d.ffd', bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0036/DATA/OMAG/24S/ORB%d.FFD', bg);
end

if (bg>=1811 & bg<=1910) %037
s1 = sprintf('d:\PV0\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0037/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=1911 & bg<=2100) %038
s1 = sprintf('d:\PV0\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0038/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=2101 & bg<=2230) %039
s1 = sprintf('d:\PV0\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0039/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=2231 & bg<=2290) %040
s1 = sprintf('d:\PV0\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0040/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=2291 & bg<=2330) %041
s1 = sprintf('d:\PV0\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0041/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=2331 & bg<=2390) %042
s1 = sprintf('d:\PV0\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0042/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=2391 & bg<=2570) %043
s1 = sprintf('d:\PV0\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0043/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=2571 & bg<=2760) %044
s1 = sprintf('d:\PV0\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0044/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=2761 & bg<=2830) \%045
  s1 = sprintf('d:\\PV0\DATA\\OMAG\\24S\\ORB%d.ffd',bg);
s2 =
  sprintf('http://www.igpp.ucla.edu/cache1/PV01_0045/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=2831 & bg<=2870) \%046
  s1 = sprintf('d:\\PV0\DATA\\OMAG\\24S\\ORB%d.ffd',bg);
s2 =
  sprintf('http://www.igpp.ucla.edu/cache1/PV01_0046/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=2871 & bg<=2900) \%047
  s1 = sprintf('d:\\PV0\DATA\\OMAG\\24S\\ORB%d.ffd',bg);
s2 =
  sprintf('http://www.igpp.ucla.edu/cache1/PV01_0047/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=2901 & bg<=2920) \%048
  s1 = sprintf('d:\\PV0\DATA\\OMAG\\24S\\ORB%d.ffd',bg);
s2 =
  sprintf('http://www.igpp.ucla.edu/cache1/PV01_0048/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=2921 & bg<=2950) \%049
  s1 = sprintf('d:\\PV0\DATA\\OMAG\\24S\\ORB%d.ffd',bg);
s2 =
  sprintf('http://www.igpp.ucla.edu/cache1/PV01_0049/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=2951 & bg<=3020) \%050
  s1 = sprintf('d:\\PV0\DATA\\OMAG\\24S\\ORB%d.ffd',bg);
s2 =
  sprintf('http://www.igpp.ucla.edu/cache1/PV01_0050/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=3021 & bg<=3140) \%051
  s1 = sprintf('d:\\PV0\DATA\\OMAG\\24S\\ORB%d.ffd',bg);
s2 =
  sprintf('http://www.igpp.ucla.edu/cache1/PV01_0051/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=3141 & bg<=3320) \%052
  s1 = sprintf('d:\\PV0\DATA\\OMAG\\24S\\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PVOL_0052 DATA/OMAG/24S/ORB%d.
FFD',bg);
end

if (bg>=3321 & bg<=3390) %0053
s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PVOL_0053 DATA/OMAG/24S/ORB%d.
FFD',bg);
end

if (bg>=3391 & bg<=3450) %0054
s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PVOL_0054 DATA/OMAG/24S/ORB%d.
FFD',bg);
end

if (bg>=3451 & bg<=3500) %0055
s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PVOL_0055 DATA/OMAG/24S/ORB%d.
FFD',bg);
end

if (bg>=3501 & bg<=3540) %0056
s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PVOL_0056 DATA/OMAG/24S/ORB%d.
FFD',bg);
end

if (bg>=3541 & bg<=3580) %0057
s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PVOL_0057 DATA/OMAG/24S/ORB%d.
FFD',bg);
end

if (bg>=3581 & bg<=3601) %0058
s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PVOL_0058 DATA/OMAG/24S/ORB%d.
FFD',bg);
end

if (bg>=3602 & bg<=3800) %0059
s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PVOL_0059 DATA/OMAG/24S/ORB%d.
FFD',bg);
end

if (bg>=3801 & bg<=3970) %0060
s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0060/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=3974 & bg<=4020) %0061
  s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0061/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=4021 & bg<=4050) %0062
  s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0062/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=4051 & bg<=4100) %0063
  s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0063/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=4101 & bg<=4150) %0064
  s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0064/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=4151 & bg<=4300) %0065
  s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0065/DATA/OMAG/24S/ORB%d.FFD',bg);
end

if (bg>=4301 & bg<=4600) %0066
  if (bg<4400)
    s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0066/DATA/OMAG/24S/ORB4300/ORB%d.FFD',bg);
  end
  if (bg>=4400 & bg<4500)
    s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0066/DATA/OMAG/24S/ORB4400/ORB%d.FFD',bg);
  end
  if (bg>=4500)
    s1 = sprintf('d:\PVO\DATA\OMAG\24S\ORB%d.ffd',bg);
s2 = sprintf('http://www.igpp.ucla.edu/cache1/POV0_0067/DATA/OMAG/24S/ORB4600/ORB%d.FFD',bg);
end

if (bg>=4601 & bg<=4750)%0067
  if (bg<4700)
    s1 = sprintf('d:\\POV0\DATA\OMAG\24S\ORB%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/POV0_0067/DATA/OMAG/24S/ORB4600/ORB%d.FFD',bg);
  end
  if (bg>=4700)
    s1 = sprintf('d:\\POV0\DATA\OMAG\24S\ORB%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/POV0_0067/DATA/OMAG/24S/ORB4700/ORB%d.FFD',bg);
  end
end

if (bg>=4751 & bg<=5055)%0068
  if (bg<4800)
    s1 = sprintf('d:\\POV0\DATA\OMAG\24S\ORB%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/POV0_0068/DATA/OMAG/24S/ORB4700/ORB%d.FFD',bg);
  end
  if (bg>=4800 & bg<4900)
    s1 = sprintf('d:\\POV0\DATA\OMAG\24S\ORB%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/POV0_0068/DATA/OMAG/24S/ORB4800/ORB%d.FFD',bg);
  end
  if (bg>=4900 & bg<5000)
    s1 = sprintf('d:\\POV0\DATA\OMAG\24S\ORB%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/POV0_0068/DATA/OMAG/24S/ORB4900/ORB%d.FFD',bg);
  end
  if (bg>=5000)
    s1 = sprintf('d:\\POV0\DATA\OMAG\24S\ORB%d.ffd',bg);
    s2 = sprintf('http://www.igpp.ucla.edu/cache1/POV0_0068/DATA/OMAG/24S/ORB5000/ORB%d.FFD',bg);
  end
end

% open input file
urlwrite(s2, s1);
end

fclose(fid0);
Ephemeris Download Script

ephem_download.m
% this matlab script will download PVO ephemeris data to local drive

% initialize the starting orbit and finishing orbit
ORBIT_MIN = 4741;
ORBIT_MAX = 5055;

s1 = 'test';
s2 = 'test2';

mkdir('d:\BVO');
bg = ORBIT_MIN;
while (bg <= ORBIT_MAX)
    if (bg>=252 & bg<=261), bg = 282;, end
    if (bg>=836 & bg<=865), bg = 866;, end
    if (bg>=1415 & bg<=1446), bg = 1447;, end
    if (bg>=1454 & bg<=1456), bg = 1457;, end
    if (bg>=1480 & bg<=1481), bg = 1482;, end
    if (bg==1546), bg = 1547;, end
    if (bg==1717), bg = 1718;, end
    if (bg==1852), bg = 1853;, end
    if (bg==1980), bg = 1981;, end
    if (bg>=2010 & bg<=2032), bg = 2033;, end
    if (bg>=2162 & bg<=2167), bg = 2168;, end
    if (bg>=2470 & bg<=2474), bg = 2475;, end
    if (bg==2484), bg = 2485;, end
    if (bg>=2502 & bg<=2503), bg = 2504;, end
    if (bg==2521), bg = 2522;, end
    if (bg==2580), bg = 2581;, end
    if (bg>=2590 & bg<=2613), bg = 2614;, end
    if (bg==2615), bg = 2616;, end
    if (bg==3166), bg = 3167;, end
    if (bg==3169 & bg<=3195), bg = 3196;, end
    if (bg==3306), bg = 3307;, end
    if (bg>=3761 & bg <= 3787), bg = 3788; end
    if (bg==3971 & bg <= 3973), bg = 3974; end
    if (bg>=4333 & bg <= 4365), bg = 4366; end
    if (bg==4367), bg = 4368;, end
    if (bg==4379), bg = 4380;, end
    if (bg==4402), bg = 4403;, end
    if (bg==4427), bg = 4428;, end
    if (bg==4449), bg = 4450;, end
    if (bg==4487), bg = 4488;, end
    if (bg==4531), bg = 4532;, end
    if (bg>=4902 & bg <= 4908), bg = 4909; end
    if (bg==4927 & bg <= 4960), bg = 4961; end
    if (bg>=4678 & bg <= 4682), bg = 4683; end
    if (bg==4695), bg = 4696;, end
    if (bg==4741), bg = 4742;, end
    if (bg==4780), bg = 4781;, end
    if (bg==4860), bg = 4861;, end
if (bg==4924), bg = 4925;, end
if (bg>=4981 & bg <= 4982), bg = 4983; end
if (bg==4993), bg = 4994;, end

bg

if (bg>=1 & bg<9) %0001
  s1 = sprintf('d:\PVO\DATA\EPHEM\ORB000%d.ffd',bg);
  s2 = sprintf('http://www.igpp.ucla.edu/cache1/1V01_0001/DATA/EPHEM/ORB000%d.FFD',bg);
end
if (bg>=10 & bg<=40) %0001
  s1 = sprintf('d:\PVO\DATA\EPHEM\ORB00%d.ffd',bg);
  s2 = sprintf('http://www.igpp.ucla.edu/cache1/1V01_0001/DATA/EPHEM/ORB00%d.FFD',bg);
end

if (bg>=41 & bg<=80) %0002
  s1 = sprintf('d:\PVO\DATA\EPHEM\ORB00%d.ffd',bg);
  s2 = sprintf('http://www.igpp.ucla.edu/cache1/1V01_0002/DATA/EPHEM/ORB00%d.FFD',bg);
end

if (bg>=81 & bg<=99) %0003
  s1 = sprintf('d:\PVO\DATA\EPHEM\ORB00%d.ffd',bg);
  s2 = sprintf('http://www.igpp.ucla.edu/cache1/1V01_0003/DATA/EPHEM/ORB00%d.FFD',bg);
end
if (bg>=100 & bg<=110) %0003
  s1 = sprintf('d:\PVO\DATA\EPHEM\ORB0%d.ffd',bg);
  s2 = sprintf('http://www.igpp.ucla.edu/cache1/1V01_0003/DATA/EPHEM/ORB0%d.FFD',bg);
end

if (bg>=111 & bg<=150) %0004
  s1 = sprintf('d:\PVO\DATA\EPHEM\ORB0%d.ffd',bg);
  s2 = sprintf('http://www.igpp.ucla.edu/cache1/1V01_0004/DATA/EPHEM/ORB0%d.FFD',bg);
end
if (bg>=151 & bg<=190) %0005
  s1 = sprintf('d:\PVO\DATA\EPHEM\ORB0%d.ffd',bg);
  s2 = sprintf('http://www.igpp.ucla.edu/cache1/1V01_0005/DATA/EPHEM/ORB0%d.FFD',bg);
end
if (bg>=191 & bg<=251) %0006
  s1 = sprintf('d:\PVO\DATA\EPHEM\ORB0%d.ffd',bg);
s2 =
    sprintf('http://www.igpp.ucla.edu/cache1/PV01_0006/DATA/EPHEM/ORB0%d.FFD',bg);
end

if (bg>=282 & bg<=350)%0007
    s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB0%d.ffd',bg);
    s2 =
    sprintf('http://www.igpp.ucla.edu/cache1/PV01_0007/DATA/EPHEM/ORB0%d.FFD
D',bg);
end

if (bg>=351 & bg<=390)%0008
    s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB0%d.ffd',bg);
    s2 =
    sprintf('http://www.igpp.ucla.edu/cache1/PV01_0008/DATA/EPHEM/ORB0%d.FFD
D',bg);
end

if (bg>=391 & bg<=430)%0009
    s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB0%d.ffd',bg);
    s2 =
    sprintf('http://www.igpp.ucla.edu/cache1/PV01_0009/DATA/EPHEM/ORB0%d.FFD
D',bg);
end

if (bg>=431 & bg<=460)%0010
    s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB0%d.ffd',bg);
    s2 =
    sprintf('http://www.igpp.ucla.edu/cache1/PV01_0010/DATA/EPHEM/ORB0%d.FFD
D',bg);
end

if (bg>=461 & bg<=500)%0011
    s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB0%d.ffd',bg);
    s2 =
    sprintf('http://www.igpp.ucla.edu/cache1/PV01_0011/DATA/EPHEM/ORB0%d.FFD
D',bg);
end

if (bg>=501 & bg<=540)%0012
    s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB0%d.ffd',bg);
    s2 =
    sprintf('http://www.igpp.ucla.edu/cache1/PV01_0012/DATA/EPHEM/ORB0%d.FFD
D',bg);
end

if (bg>=541 & bg<=570)%0013
    s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB0%d.ffd',bg);
    s2 =
    sprintf('http://www.igpp.ucla.edu/cache1/PV01_0013/DATA/EPHEM/ORB0%d.FFD
D',bg);
end

if (bg>=571 & bg<=590)%0014
    s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB0%d.ffd',bg);
s2 = sprintf('http://www.igpp.ucla.edu/cache1/PVOL_0014/DATA/EPHEM/ORBO%d.FFD',bg);
end

if (bg>=591 & bg<=620) %0015
   s1 = sprintf('d:\\PVO\\DATA\\EPHEM\\ORB0%d.ffd',bg);
   s2 = sprintf('http://www.igpp.ucla.edu/cache1/PVOL_0015/DATA/EPHEM/ORBO%d.FFD',bg);
end

if (bg>=621 & bg<=660) %0016
   s1 = sprintf('d:\\PVO\\DATA\\EPHEM\\ORB0%d.ffd',bg);
   s2 = sprintf('http://www.igpp.ucla.edu/cache1/PVOL_0016/DATA/EPHEM/ORBO%d.FFD',bg);
end

if (bg>=661 & bg<=740) %0017
   s1 = sprintf('d:\\PVO\\DATA\\EPHEM\\ORB0%d.ffd',bg);
   s2 = sprintf('http://www.igpp.ucla.edu/cache1/PVOL_0017/DATA/EPHEM/ORBO%d.FFD',bg);
end

if (bg>=741 & bg<=800) %0018
   s1 = sprintf('d:\\PVO\\DATA\\EPHEM\\ORB0%d.ffd',bg);
   s2 = sprintf('http://www.igpp.ucla.edu/cache1/PVOL_0018/DATA/EPHEM/ORBO%d.FFD',bg);
end

if (bg>=801 & bg<=900) %0019
   s1 = sprintf('d:\\PVO\\DATA\\EPHEM\\ORB0%d.ffd',bg);
   s2 = sprintf('http://www.igpp.ucla.edu/cache1/PVOL_0019/DATA/EPHEM/ORBO%d.FFD',bg);
end

if (bg>=901 & bg<=990) %0020
   s1 = sprintf('d:\\PVO\\DATA\\EPHEM\\ORB0%d.ffd',bg);
   s2 = sprintf('http://www.igpp.ucla.edu/cache1/PVOL_0020/DATA/EPHEM/ORBO%d.FFD',bg);
end

if (bg>=991 & bg<=999) %0021
   s1 = sprintf('d:\\PVO\\DATA\\EPHEM\\ORB0%d.ffd',bg);
   s2 = sprintf('http://www.igpp.ucla.edu/cache1/PVOL_0021/DATA/EPHEM/ORBO%d.FFD',bg);
end

if (bg>=1000 & bg<=1050) %0021
   s1 = sprintf('d:\\PVO\\DATA\\EPHEM\\ORB0%d.ffd',bg);
s2 = 
sprintf('http://www.igpp.ucla.edu/cachel/PV01_0021/DATA/EPHEM/ORB%d.FFD ',bg);
end

if (bg>=1051 & bg<=1080) %0022
  s1 = sprintf('d:\\PVO\DATA\EPHEM\ORB%d.ffd',bg);
  s2 = 
sprintf('http://www.igpp.ucla.edu/cachel/PV01_0022/DATA/EPHEM/ORB%d.FFD ',bg);
end

if (bg>=1081 & bg<=1120) %0023
  s1 = sprintf('d:\\PVO\DATA\EPHEM\ORB%d.ffd',bg);
  s2 = 
sprintf('http://www.igpp.ucla.edu/cachel/PV01_0023/DATA/EPHEM/ORB%d.FFD ',bg);
end

if (bg>=1121 & bg<=1160) %0024
  s1 = sprintf('d:\\PVO\DATA\EPHEM\ORB%d.ffd',bg);
  s2 = 
sprintf('http://www.igpp.ucla.edu/cachel/PV01_0024/DATA/EPHEM/ORB%d.FFD ',bg);
end

if (bg>=1161 & bg<=1200) %0025
  s1 = sprintf('d:\\PVO\DATA\EPHEM\ORB%d.ffd',bg);
  s2 = 
sprintf('http://www.igpp.ucla.edu/cachel/PV01_0025/DATA/EPHEM/ORB%d.FFD ',bg);
end

if (bg>=1201 & bg<=1240) %0026
  s1 = sprintf('d:\\PVO\DATA\EPHEM\ORB%d.ffd',bg);
  s2 = 
sprintf('http://www.igpp.ucla.edu/cachel/PV01_0026/DATA/EPHEM/ORB%d.FFD ',bg);
end

if (bg>=1241 & bg<=1280) %0027
  s1 = sprintf('d:\\PVO\DATA\EPHEM\ORB%d.ffd',bg);
  s2 = 
sprintf('http://www.igpp.ucla.edu/cachel/PV01_0027/DATA/EPHEM/ORB%d.FFD ',bg);
end

if (bg>=1281 & bg<=1330) %0028
  s1 = sprintf('d:\\PVO\DATA\EPHEM\ORB%d.ffd',bg);
  s2 = 
sprintf('http://www.igpp.ucla.edu/cachel/PV01_0028/DATA/EPHEM/ORB%d.FFD ',bg);
end

if (bg>=1331 & bg<=1400) %0029
  s1 = sprintf('d:\\PVO\DATA\EPHEM\ORB%d.ffd',bg);

s2 = 
    printf('http://www.igpp.ucla.edu/cache1/PV01_0029/DATA/EPHEM/ORB%d.FFD',bg);
    end

if (bg>=1401 & bg<=1510) %0030
    s1 = printf('d:\PV0\DATA\EPHEM\ORB%d.ffd',bg);
    s2 =
    printf('http://www.igpp.ucla.edu/cache1/PV01_0030/DATA/EPHEM/ORB%d.FFD',bg);
    end

if (bg>=1511 & bg<=1580) %0031
    s1 = printf('d:\PV0\DATA\EPHEM\ORB%d.ffd',bg);
    s2 =
    printf('http://www.igpp.ucla.edu/cache1/PV01_0031/DATA/EPHEM/ORB%d.FFD',bg);
    end

if (bg>=1581 & bg<=1620) %0032
    s1 = printf('d:\PV0\DATA\EPHEM\ORB%d.ffd',bg);
    s2 =
    printf('http://www.igpp.ucla.edu/cache1/PV01_0032/DATA/EPHEM/ORB%d.FFD',bg);
    end

if (bg>=1621 & bg<=1680) %0033
    s1 = printf('d:\PV0\DATA\EPHEM\ORB%d.ffd',bg);
    s2 =
    printf('http://www.igpp.ucla.edu/cache1/PV01_0033/DATA/EPHEM/ORB%d.FFD',bg);
    end

if (bg>=1681 & bg<=1730) %0034
    s1 = printf('d:\PV0\DATA\EPHEM\ORB%d.ffd',bg);
    s2 =
    printf('http://www.igpp.ucla.edu/cache1/PV01_0034/DATA/EPHEM/ORB%d.FFD',bg);
    end

if (bg>=1731 & bg<=1770) %0035
    s1 = printf('d:\PV0\DATA\EPHEM\ORB%d.ffd',bg);
    s2 =
    printf('http://www.igpp.ucla.edu/cache1/PV01_0035/DATA/EPHEM/ORB%d.FFD',bg);
    end

if (bg>=1771 & bg<=1810) %0036
    s1 = printf('d:\PV0\DATA\EPHEM\ORB%d.ffd',bg);
    s2 =
    printf('http://www.igpp.ucla.edu/cache1/PV01_0036/DATA/EPHEM/ORB%d.FFD',bg);
    end

if (bg>=1811 & bg<=1910) %0037
    s1 = printf('d:\PV0\DATA\EPHEM\ORB%d.ffd',bg);
s2 = 
    sprintf('http://www.igpp.ucla.edu/cach1/PV01_0037/DATA/EPHEM/ORB%d.FFD 
',bg);
    end

    if (bg>=1911 & bg<=2100) %0038
        s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB%d.ffd',bg);
        s2 = 
    sprintf('http://www.igpp.ucla.edu/cach1/PV01_0038/DATA/EPHEM/ORB%d.FFD 
',bg);
    end

    if (bg>=2101 & bg<=2230) %0039
        s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB%d.ffd',bg);
        s2 = 
    sprintf('http://www.igpp.ucla.edu/cach1/PV01_0039/DATA/EPHEM/ORB%d.FFD 
',bg);
    end

    if (bg>=2231 & bg<=2290) %0040
        s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB%d.ffd',bg);
        s2 = 
    sprintf('http://www.igpp.ucla.edu/cach1/PV01_0040/DATA/EPHEM/ORB%d.FFD 
',bg);
    end

    if (bg>=2291 & bg<=2330) %0041
        s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB%d.ffd',bg);
        s2 = 
    sprintf('http://www.igpp.ucla.edu/cach1/PV01_0041/DATA/EPHEM/ORB%d.FFD 
',bg);
    end

    if (bg>=2331 & bg<=2390) %0042
        s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB%d.ffd',bg);
        s2 = 
    sprintf('http://www.igpp.ucla.edu/cach1/PV01_0042/DATA/EPHEM/ORB%d.FFD 
',bg);
    end

    if (bg>=2391 & bg<=2570) %0043
        s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB%d.ffd',bg);
        s2 = 
    sprintf('http://www.igpp.ucla.edu/cach1/PV01_0043/DATA/EPHEM/ORB%d.FFD 
',bg);
    end

    if (bg>=2571 & bg<=2760) %0044
        s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB%d.ffd',bg);
        s2 = 
    sprintf('http://www.igpp.ucla.edu/cach1/PV01_0044/DATA/EPHEM/ORB%d.FFD 
',bg);
    end

    if (bg>=2761 & bg<=2830) %0045
        s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB%d.ffd',bg);
    
s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0045/DATA/EPHEM/ORB%d.FFD', bg);
end

if (bg>=2831 & bg<=2870)%0046
  s1 = sprintf('d:\PVO\DATA\EPHEM\ORB%d.ffd',bg);
s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0046/DATA/EPHEM/ORB%d.FFD',bg);
end

if (bg>=2871 & bg<=2900)%0047
  s1 = sprintf('d:\PVO\DATA\EPHEM\ORB%d.ffd',bg);
s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0047/DATA/EPHEM/ORB%d.FFD',bg);
end

if (bg>=2901 & bg<=2920)%0048
  s1 = sprintf('d:\PVO\DATA\EPHEM\ORB%d.ffd',bg);
s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0048/DATA/EPHEM/ORB%d.FFD',bg);
end

if (bg>=2921 & bg<=2950)%0049
  s1 = sprintf('d:\PVO\DATA\EPHEM\ORB%d.ffd',bg);
s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0049/DATA/EPHEM/ORB%d.FFD',bg);
end

if (bg>=2951 & bg<=3020)%0050
  s1 = sprintf('d:\PVO\DATA\EPHEM\ORB%d.ffd',bg);
s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0050/DATA/EPHEM/ORB%d.FFD',bg);
end

if (bg>=3021 & bg<=3140)%0051
  s1 = sprintf('d:\PVO\DATA\EPHEM\ORB%d.ffd',bg);
s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0051/DATA/EPHEM/ORB%d.FFD',bg);
end

if (bg>=3141 & bg<=3320)%0052
  s1 = sprintf('d:\PVO\DATA\EPHEM\ORB%d.ffd',bg);
s2 = sprintf('http://www.igpp.ucla.edu/cache1/PV01_0052/DATA/EPHEM/ORB%d.FFD',bg);
end

if (bg>=3321 & bg<=3390)%0053
  s1 = sprintf('d:\PVO\DATA\EPHEM\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0053/DATA/EPHEM/ORB%d.FFD',bg);
end

if (bg>=3391 & bg<=3450) %0054
  s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB%d.ffd',bg);
  s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0054/DATA/EPHEM/ORB%d.FFD',bg);
end

if (bg>=3451 & bg<=3500) %0055
  s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB%d.ffd',bg);
  s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0055/DATA/EPHEM/ORB%d.FFD',bg);
end

if (bg>=3501 & bg<=3540) %0056
  s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB%d.ffd',bg);
  s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0056/DATA/EPHEM/ORB%d.FFD',bg);
end

if (bg>=3541 & bg<=3580) %0057
  s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB%d.ffd',bg);
  s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0057/DATA/EPHEM/ORB%d.FFD',bg);
end

if (bg>=3581 & bg<=3601) %0058
  s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB%d.ffd',bg);
  s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0058/DATA/EPHEM/ORB%d.FFD',bg);
end

if (bg>=3602 & bg<=3800) %0059
  s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB%d.ffd',bg);
  s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0059/DATA/EPHEM/ORB%d.FFD',bg);
end

if (bg>=3801 & bg<=3970) %0060
  s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB%d.ffd',bg);
  s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0060/DATA/EPHEM/ORB%d.FFD',bg);
end

if (bg>=3974 & bg<=4020) %0061
  s1 = sprintf('d:\\PV0\\DATA\\EPHEM\\ORB%d.ffd',bg);
s2 =
sprintf('http://www.igpp.ucla.edu/cache1/PV01_0061/DATA/EPHEM/ORB%d.FFD '
',bg);
end

if (bg>=4021 & bg<=4050)%0062
  s1 = sprintf('d:\\PV01\DATA\\EPHEM\\ORB%d.ffd',bg);
  s2 =
  sprintf('http://www.igpp.ucla.edu/cache1/PV01_0062/DATA/EPHEM/ORB%d.FFD '
',bg);
end

if (bg>=4051 & bg<=4100)%0063
  s1 = sprintf('d:\\PV01\DATA\\EPHEM\\ORB%d.ffd',bg);
  s2 =
  sprintf('http://www.igpp.ucla.edu/cache1/PV01_0063/DATA/EPHEM/ORB%d.FFD '
',bg);
end

if (bg>=4101 & bg<=4150)%0064
  s1 = sprintf('d:\\PV01\DATA\\EPHEM\\ORB%d.ffd',bg);
  s2 =
  sprintf('http://www.igpp.ucla.edu/cache1/PV01_0064/DATA/EPHEM/ORB%d.FFD '
',bg);
end

if (bg>=4151 & bg<=4300)%0065
  s1 = sprintf('d:\\PV01\DATA\\EPHEM\\ORB%d.ffd',bg);
  s2 =
  sprintf('http://www.igpp.ucla.edu/cache1/PV01_0065/DATA/EPHEM/ORB%d.FFD '
',bg);
end

if (bg>=4301 & bg<=4600)%0066
  s1 = sprintf('d:\\PV01\DATA\\EPHEM\\ORB%d.ffd',bg);
  s2 =
  sprintf('http://www.igpp.ucla.edu/cache1/PV01_0066/DATA/EPHEM/ORB%d.FFD '
',bg);
end

if (bg>=4601 & bg<=4750)%0067
  s1 = sprintf('d:\\PV01\DATA\\EPHEM\\ORB%d.ffd',bg);
  s2 =
  sprintf('http://www.igpp.ucla.edu/cache1/PV01_0067/DATA/EPHEM/ORB%d.FFD '
',bg);
end

if (bg>=4751 & bg<=5055)%0068
  s1 = sprintf('d:\\PV01\DATA\\EPHEM\\ORB%d.ffd',bg);
  s2 =
  sprintf('http://www.igpp.ucla.edu/cache1/PV01_0068/DATA/EPHEM/ORB%d.FFD '
',bg);
end

% open input file
uriwrite(s2, s1);
bg = bg + 1;
endwhile
Time At Altitude (not in trough) Script

ephem_alt_bin_filter.m
% this is the first matlab script and it will parse the ephemeris files for:
% - solar zenith angles >120 degrees, and < 180 degrees
% - altitudes that are less than 1000 km
% this script will output a file with three columns of numbers, the first will be
% an integer of the orbit number that passed the above two filters, the second and
% third will be the start and stop times as double floats of when the spacecraft
% satisfied these filters during the noted orbit.

% initialize the starting orbit and finishing orbit
ORBIT_MIN = 1;
ORBIT_MAX = 5055;

% initialize altitude bins
delta_alt = 16;
alct = (400 - 140)/delta_alt + 1;

for ia=1:alct
    alt_bin(ia) = 0;
    alt_lim(ia) = 140 + delta_alt*(ia-1);
end

s1 = sprintf('d:\\PVO\\season 8 orbits.txt');

% open file for input of data
fid0 = fopen(s1,'r');

orbit = fscanf(fid0, '%d %f %f %f', [4,inf]);
orbit = orbit';
[m,n] = size(orbit)

for bg = 1:m;
    if (orbit(bg,1)<=9), s2 = sprintf('d:\\PVO\\DATA\\EPHEM\\ORB000%0d.FFD',orbit(bg,1));, end
    if (orbit(bg,1)>=10 & orbit(bg,1)<=99), s2 = sprintf('d:\\PVO\\DATA\\EPHEM\\ORB00%d.FFD',orbit(bg,1));, end
    if (orbit(bg,1)>=100 & orbit(bg,1)<=999), s2 = sprintf('d:\\PVO\\DATA\\EPHEM\\ORB0%d.FFD',orbit(bg,1));, end
    if (orbit(bg,1)>=1000), s2 = sprintf('d:\\PVO\\DATA\\EPHEM\\ORB%d.FFD',orbit(bg,1));, end

    %open ephemeris data file
    fid1 = fopen(s2,'r');
    if fid1 == -1
        bg
        disp('arg!');
        return
end if fid1 == -1

% begin process of reading in ephemeris data for the given orbit
count = 1;

eofstat = feof(fid1);
double x;

while(~eofstat)
    x = fread(fid1,1,'float64','b');
eofstat = feof(fid1);
    if(eofstat),break, end if
    UT(count) = x;

    x = fread(fid1,1,'float32','s');
    X_VSO(count) = x;

    x = fread(fid1,1,'float32','s');
    Y_VSO(count) = x;

    x = fread(fid1,1,'float32','s');
    Z_VSO(count) = x;

    x = fread(fid1,1,'float32','s');
    ALT(count) = x;

    x = fread(fid1,1,'float32','s');
    SZA(count) = x;

    x = fread(fid1,1,'float32','s');
    PLONG(count) = x;

    x = fread(fid1,1,'float32','s');
    PLAT(count) = x;

    x = fread(fid1,1,'float32','s');
    SPX_VSO(count) = x;

    x = fread(fid1,1,'float32','s');
    SPY_VSO(count) = x;

    x = fread(fid1,1,'float32','s');
    SPZ_VSO(count) = x;

    x = fread(fid1,1,'float32','s');
    CLAT(count) = x;

    x = fread(fid1,1,'float32','s');
    CLONG(count) = x;

    x = fread(fid1,1,'float32','s');
    ELONG(count) = x;

    x = fread(fid1,1,'float32','s');
    RSUN(count) = x;

    eofstat = feof(fid1);
count = count + 1;
    if(count > 10000), break, end\if
end\while(~eofstat)

%since it is at EOF, we can close this
fclose(fid1);

%get proper max count
count = count - 1;
limit = count;

%initialize solar zenith and altitude flags
SZA_FLAG = 0;
ALT_FLAG = 0;
EDGE_UP = 0;
EDGE_DOWN = 0;
BOTH = 0;
double start;
start = 0;
double stop;
stop = 0;
min_alt = 10001;

for ia = 1:alct
    ALT_FLAG(ia) = 0;
    ALT_start(ia) = 0;
    ALT_stop(ia) = 0;
end

%% now time to test bins
for nt = 2:limit;
    for ia = 1:alct-1;
        i+1 to i -- e.g. 156 to 140,

        % see if PVO is just going below upper level of bin (ia+1)
        if (ALT(nt)<alt_lim(ia+1) & ALT(nt-1)>alt_lim(ia+1) &
~ALT_FLAG(ia+1))
            ALT_FLAG(ia+1) = 1;
            ALT_start(ia+1) = abs(alt_lim(ia+1)-ALT(nt))/abs(ALT(nt-1)-ALT(nt)) * abs(UT(nt)-UT(nt-1)) + UT(nt-1);
        end%if

        % see if PVO is going below lower level if bin (ia)
        if (ALT(nt)<alt_lim(ia) & ALT(nt-1)>alt_lim(ia) &
ALT_FLAG(ia+1))
            ALT_FLAG(ia+1) = 0;
            ALT_stop(ia+1) = abs(alt_lim(ia)-ALT(nt))/abs(ALT(nt-1)-ALT(nt)) * abs(UT(nt)-UT(nt-1)) + UT(nt-1);
            alt_bin(ia+1) = alt_bin(ia+1) + (ALT_stop(ia+1) - ALT_start(ia+1));
        end%if
% see if PVO went below upper limit, but never breaks lower
limit since periapse is above lower limit
    if (ALT(nt)>=alt_lim(ia+1) & ALT(nt-1)<alt_lim(ia+1) &
ALT_FLAG(ia+1))
        ALT_FLAG(ia+1) = 0;
        ALT_stop(ia+1) = abs(alt_lim(ia+1)-ALT(nt-1))/abs(ALT(nt-1)-ALT(nt)) * abs(UT(nt)-UT(nt-1)) + UT(nt-1);
        alt_bin(ia+1) = alt_bin(ia+1) + (ALT_stop(ia+1) -
ALT_start(ia+1));
    end

% coming from below bin level...
    if (ALT(nt)>=alt_lim(ia) & ALT(nt-1)<alt_lim(ia) &
~ALT_FLAG(ia+1))
        ALT_FLAG(ia+1) = 1;
        ALT_stop(ia+1) = abs(alt_lim(ia)-ALT(nt-1))/abs(ALT(nt-1)-
ALT(nt)) * abs(UT(nt)-UT(nt-1)) + UT(nt-1);
        alt_bin(ia+1) = alt_bin(ia+1) + (ALT_stop(ia+1) -
ALT_start(ia+1));
    end
end
end

end for nt = 1:limit;

% see if we caught us a PVO fishey
fprintf(fid0,'%d ', bg);
fprintf(fid0,'%f ', min_alt);
fprintf(fid0,'
');

%bg = bg + 1;

end for bg = ORBIT_MIN:ORBIT_MAX;
fclose(fid0);

fid2 = fopen('d:\PVO\season_bin_count.txt', 'a');
fprintf(fid2,'%d &d', orbit[1,1], orbit(m,1));
fprintf(fid2,'
');
for ia=1:alct;
    fprintf(fid2,'%d ', alt_lim(ia));
    fprintf(fid2,'%f ', alt_bin(ia));
    fprintf(fid2,'
');
end
fclose(fid2);
Periapsis History Of PVO Script

ephem_alt_filter.m
% this is a matlab script that will parse the ephemeris files for:
% - solar zenith angles >120 degrees, and < 180 degrees
% - altitudes that are less than 1000 km
% this script will output a file with three columns of numbers, the
% first will be
% an integer of the orbit number that passed the above two filters, the
% second and
% third will be the start and stop times as double floats of when the
% spacecraft
% satisfied these filters during the noted orbit.

% initialize the starting orbit and finishing orbit
ORBIT_MIN = 1;
ORBIT_MAX = 5055;
s1 = sprintf('d:\\PVO\\ephem_alt_filter.txt');

% open file for output of data
fid0 = fopen(s1,'a');

bg = ORBIT_MIN;
% for bg = ORBIT_MIN:ORBIT_MAX;
while (bg <= ORBIT_MAX)
    if (bg>=252 & bg<=281), bg = 282;, end
    if (bg>=836 & bg<=865), bg = 866;, end
    if (bg>=1415 & bg<=1446), bg = 1447;, end
    if (bg>=1454 & bg<=1456), bg = 1457;, end
    if (bg>=1480 & bg<=1481), bg = 1482;, end
    if (bg>=1546), bg = 1547;, end
    if (bg==1717), bg = 1718;, end
    if (bg==1852), bg = 1853;, end
    if (bg==1980), bg = 1981;, end
    if (bg>=2010 & bg<=2032), bg = 2033;, end
    if (bg>=2162 & bg<=2167), bg = 2168;, end
    if (bg>=2470 & bg<=2474), bg = 2475;, end
    if (bg==2484), bg = 2485;, end
    if (bg>=2502 & bg<=2503), bg = 2504;, end
    if (bg=2521), bg = 2522;, end
    if (bg=2580), bg = 2581;, end
    if (bg=2590 & bg=2613), bg = 2614;, end
    if (bg=2615), bg = 2616;, end
    if (bg==3166), bg = 3167;, end
    if (bg>=3169 & bg=3195), bg = 3196;, end
    if (bg==3306), bg = 3307;, end
    if (bg>=3761 & bg <= 3787), bg = 3788; end
    if (bg>=3971 & bg <= 3973), bg = 3974; end
    if (bg=4333 & bg = 4365), bg = 4366; end
    if (bg=4367), bg = 4368; end
    if (bg=4379), bg = 4380; end
    if (bg=4402), bg = 4403; end
if (bg==4427), bg = 4428; end
if (bg==4449), bg = 4450; end
if (bg==4487), bg = 4488; end
if (bg==4531), bg = 4532; end
if (bg>=4902 & bg <= 4908), bg = 4909; end
if (bg>=4927 & bg <= 4960), bg = 4961; end

if (bg>=4678 & bg <= 4682), bg = 4683; end
if (bg==4695), bg = 4696; end
if (bg==4741), bg = 4742; end
if (bg==4780), bg = 4781; end
if (bg==4860), bg = 4861; end
if (bg==4924), bg = 4925; end
if (bg>=4981 & bg <= 4982), bg = 4983; end
if (bg==4993), bg = 4994; end

if (rem(bg,100)==0),bg, end

if (bg<=9), s2 = sprintf('d:\PVO\DATA\EPHEM\ORB00%d.FFD',bg); end
if (bg>=10 & bg<=99), s2 = sprintf('d:\PVO\DATA\EPHEM\ORB00%d.FFD',bg); end
if (bg>=100 & bg<=999), s2 = sprintf('d:\PVO\DATA\EPHEM\ORB0%d.FFD',bg); end
if (bg>=1000), s2 = sprintf('d:\PVO\DATA\EPHEM\ORB%d.FFD',bg); end

%open ephemeris data file
fid1 = fopen(s2,'r');

if fid1 == -1
    bg
    disp('arg!');
    return
end

fid1 = -1

% begin process of reading in ephemeris data for the given orbit
count = 1;

eofstat = feof(fid1);
double x;

while(~eofstat)
    x = fread(fid1,1,'float64','b');
eofstat = feof(fid1);
    if(eofstat),break, end
    UT(count) = x;
    x = fread(fid1,1,'float32','s');
    X_VSO(count) = x;
    x = fread(fid1,1,'float32','s');
    Y_VSO(count) = x;
    x = fread(fid1,1,'float32','s');
    Z_VSO(count) = x;
x = fread(fid1,1,'float32','s');
ALT(count) = x;

x = fread(fid1,1,'float32','s');
SZA(count) = x;

x = fread(fid1,1,'float32','s');
FLONG(count) = x;

x = fread(fid1,1,'float32','s');
FLAT(count) = x;

x = fread(fid1,1,'float32','s');
SPX_VSO(count) = x;

x = fread(fid1,1,'float32','s');
SPY_VSO(count) = x;

x = fread(fid1,1,'float32','s');
SPZ_VSO(count) = x;

x = fread(fid1,1,'float32','s');
CLAT(count) = x;

x = fread(fid1,1,'float32','s');
CLONG(count) = x;

x = fread(fid1,1,'float32','s');
ELONG(count) = x;

x = fread(fid1,1,'float32','s');
RSUN(count) = x;

eofstat =feof(fid1);

    count = count + 1;
    if(count > 10000), break, end
end

%since it is at EOF, we can close this
fclose(fid1);

%get proper max count
count = count - 1;
limit = count;

%initialize solar zenith and altitude flags
SZA_FLAG = 0;
ALT_FLAG = 0;
EDGE_UP = 0;
EDGE_DOWN = 0;
BOTH = 0;
double start;
start = 0;
double stop;
stop = 0;
min_alt = 10001;

%%% now time to test solar zenith and altitude
for nt = 1:limit;

    if (SZA(nt))<110
        SZA_FLAG=0;
    end\%if

    if(SZA(nt)>=110)
        SZA_FLAG=1;
    end\%if

    if (ALT(nt)>1000)
        ALT_FLAG = 0;
    end\%if

    if (ALT(nt)<=1000)
        ALT_FLAG = 1;
    end\%if

    if (ALT(nt)<min_alt)
        min_alt = ALT(nt);
    end\%if

    if (SZA_FLAG & ALT_FLAG)
        BOTH = 1;
    else
        BOTH = 0;
    end\%if

    if (BOTH & ~EDGE_UP)
        EDGE_UP = 1;
        start = UT(nt);
    end\%if

    if (EDGE_UP & ~BOTH & ~EDGE_DOWN)
        EDGE_DOWN = 1;
        stop = UT(nt);
    end\%if

end\%for nt = 1:limit;

% see if we caught us a PVO fishey
fprintf(fid0,'%d ', bg);
fprintf(fid0,'%f ', min_alt);
fprintf(fid0,'%\n');

bg = bg + 1;

end\%for bg = ORBIT_MIN:ORBIT_MAX;

fclose(fid0);
Season Window Generator Script

ephem_filter.m

(similar to last script, except conditional statement added to fprintf section to only pick

low, nightside orbits.)
% this is the first matlab script and it will parse the ephemeris files for:  
% - solar zenith angles >120 degrees, and < 180 degrees  
% - altitudes that are less than 1000 km  
% this script will output a file with three columns of numbers, the first will be  
% an integer of the orbit number that passed the above two filters, the second and  
% third will be the start and stop times as double floats of when the spacecraft  
% satisfied these filters during the noted orbit.  

% initialize the starting orbit and finishing orbit  
ORBIT_MIN = 1;  
ORBIT_MAX = 5055;  

s1 = sprintf('d:\EVO\ephem_filter.txt');  

% open file for output of data  
fid0 = fopen(s1,'a');  

bg = ORBIT_MIN;  
for bg = ORBIT_MIN:ORBIT_MAX;  
while (bg <= ORBIT_MAX)  
  if (bg==252 & bg<=281), bg = 282;, end  
  if (bg>=836 & bg<=865), bg = 866;, end  
  if (bg>=1415 & bg<=1446), bg = 1447;, end  
  if (bg>=1454 & bg<=1456), bg = 1457;, end  
  if (bg>=1480 & bg<=1481), bg = 1482;, end  
  if (bg==1546), bg = 1547;, end  
  if (bg==1717), bg = 1718;, end  
  if (bg==1852), bg = 1853;, end  
  if (bg==1980), bg = 1981;, end  
  if (bg==2010 & bg<=2032), bg = 2033;, end  
  if (bg==2162 & bg<=2167), bg = 2168;, end  
  if (bg==2470 & bg<=2474), bg = 2475;, end  
  if (bg==2484), bg = 2485;, end  
  if (bg==2502 & bg<=2503), bg = 2504;, end  
  if (bg==2521), bg = 2522;, end  
  if (bg==2580), bg = 2581;, end  
  if (bg==2590 & bg<=2613), bg = 2614;, end  
  if (bg==2615), bg = 2616;, end  
  if (bg==3166), bg = 3167;, end  
  if (bg==3169 & bg<=3195), bg = 3196;, end  
  if (bg==3306), bg = 3307;, end  
  if (bg==3761 & bg <= 3787), bg = 3788; end  
  if (bg==3971 & bg <= 3973), bg = 3974; end  
  if (bg==4333 & bg <= 4365), bg = 4366; end  
  if (bg==4367), bg = 4368; end  
  if (bg==4379), bg = 4380; end
if (bg==4402), bg = 4403; end
if (bg==4427), bg = 4428; end
if (bg==4449), bg = 4450; end
if (bg==4487), bg = 4488; end
if (bg==4531), bg = 4532; end
if (bg>=4902 & bg <= 4908), bg = 4909; end
if (bg>=4927 & bg <= 4960), bg = 4961; end

if (bg>=4678 & bg <= 4682), bg = 4683; end
if (bg==4695), bg = 4696; end
if (bg==4741), bg = 4742; end
if (bg==4780), bg = 4781; end
if (bg==4860), bg = 4861; end
if (bg==4924), bg = 4925; end
if (bg>=4981 & bg <= 4982), bg = 4983; end
if (bg>=4993), bg = 4994; end

if (rem(bg,100)==0),bg, end

if (bg<=9), s2 = sprintf('d:\PVO\DATA\EPHEM\ORB000%d.FFD',bg); end
if (bg>=10 & bg<=99), s2 = sprintf('d:\PVO\DATA\EPHEM\ORB00%d.FFD',bg); end
if (bg>=100 & bg<=999), s2 = sprintf('d:\PVO\DATA\EPHEM\ORB0%d.FFD',bg); end
if (bg>=1000), s2 = sprintf('d:\PVO\DATA\EPHEM\ORB%d.FFD',bg); end

%open ephemeris data file
fid1 = fopen(s2,'r');

if fid1 == -1
    bg
    disp('arg!');
    return
end%if fid1 == -1

% begin process of reading in ephemeris data for the given orbit
count = 1;

eofstat = feof(fid1);
double x;

while(~eofstat)
    x = fread(fid1,1,'float64','b');
eofstat = feof(fid1);
if(eofstat),break, end%if
UT(count) = x;
    x = fread(fid1,1,'float32','s');
X_VSO(count) = x;
    x = fread(fid1,1,'float32','s');
Y_VSO(count) = x;
    x = fread(fid1,1,'float32','s');
Z_VSO(count) = x;
x = fread(fid1,1,'float32','s');
ALT(count) = x;

x = fread(fid1,1,'float32','s');
SZA(count) = x;

x = fread(fid1,1,'float32','s');
PLONG(count) = x;

x = fread(fid1,1,'float32','s');
PLAT(count) = x;

x = fread(fid1,1,'float32','s');
SPX_VSO(count) = x;

x = fread(fid1,1,'float32','s');
SPY_VSO(count) = x;

x = fread(fid1,1,'float32','s');
SPZ_VSO(count) = x;

x = fread(fid1,1,'float32','s');
CLAT(count) = x;

x = fread(fid1,1,'float32','s');
CLONG(count) = x;

x = fread(fid1,1,'float32','s');
ELONG(count) = x;

x = fread(fid1,1,'float32','s');
RSUN(count) = x;

eofstat = feof(fid1);

count = count + 1;
if(count > 10000), break, end@if
end%while(~eofstat)

%since it is at EOF, we can close this fclose(fid1);

%get proper max count
count = count - 1;
limit = count;

%initialize solar zenith and altitude flags
SZA_FLAG = 0;
ALT_FLAG = 0;
EDGE_UP = 0;
EDGE_DOWN = 0;
BOTH = 0;
double start;
start = 0;
double stop;
stop = 0;
min_alt = 1001;

%%% now time to test solar zenith and altitude
for nt = 1:limit;

if (SZA(nt)<110
   SZA_FLAG=0;
end\%if

if(SZA(nt)>=110)
   SZA_FLAG=1;
end\%if

if (ALT(nt)>1000)
   ALT_FLAG = 0;
end\%if

if (ALT(nt)<=1000)
   ALT_FLAG = 1;
end\%if

if (ALT(nt)<=min_alt)
   min_alt = ALT(nt);
end\%if

if (SZA_FLAG & ALT_FLAG)
   BOTH = 1;
else
   BOTH = 0;
end\%if

if (BOTH & ~EDGE_UP)
   EDGE_UP = 1;
   start = UT(nt);
end\%if

if (EDGE_UP & ~BOTH & ~EDGE_DOWN)
   EDGE_DOWN = 1;
   stop = UT(nt);
end\%if

end\%for nt = 1:limit;

%%% see if we caught us a PVO fishey
if (EDGE_UP & EDGE_DOWN);
   \%disp('Got One!');
   fprintf(fid0,'%d ', bg);
   fprintf(fid0,'%f ', start);
   fprintf(fid0,'%f ', stop);
   fprintf(fid0,'%f ', min_alt);
   fprintf(fid0,'\n');
end \%if

bg = bg + 1;
end\%for bg = ORBIT_MIN:ORBIT_MAX;
fclose(fid0);
OETP Intelligent Monitoring Agent

OETP_fullup.m
% input the season data

fid1 = fopen('D:\PVO\season 1 orbits.txt');
fid1 = fopen('D:\PVO\season 2 orbits.txt');
fid1 = fopen('D:\PVO\season 3 orbits.txt');
fid1 = fopen('D:\PVO\season 4 orbits.txt');

scan_data = fscanf(fid1, '%d %f %f %f', [4, inf]);
scan_data = scan_data';
[m1,n1]=size(scan_data);

for i=1:m1;
    target_orbit(i) = scan_data(i,1);
    start_time(i) = scan_data(i,2) - 24;
    stop_time(i) = scan_data(i,3) + 24;
end
fclose(fid1);

fid2 = fopen('D:\PVO\windows.txt', 'a');
fid3 = fopen('D:\PVO\scales.txt', 'a');

USH_SUM = 0;
MSH_SUM = 0;
LSH_SUM = 0;
USH_RSQ_SUM = 0;
MSH_RSQ_SUM = 0;
LSH_RSQ_SUM = 0;

%% !!!!don't forget to change upper limit, and target orbit!!!!
for il=1:m1
    bg = target_orbit(il);

    % input the QETP data

    REFYR = 1966;
    MN = [31,28,31,30,31,30,31,31,30,31,30,31];
    UT_S = 0;
    days = 0;

    if (bg>=26 & bg<=126), s2 = sprintf('d:\PVO\DATA\OETP\LOWRES\season1.tab',bg);, end
    if (bg>=251 & bg<=342), s2 = sprintf('d:\PVO\DATA\OETP\LOWRES\season2.tab',bg);, end
    if (bg>=475 & bg<=567), s2 = sprintf('d:\PVO\DATA\OETP\LOWRES\season3.tab',bg);, end
    if (bg>=699 & bg<=791), s2 = sprintf('d:\PVO\DATA\OETP\LOWRES\season4.tab',bg);, end
    fid1 = fopen(s2,'r');

file_data = fscanf(fid1, '%d %d %f %f %f %f', [7,inf]);
file_data = file_data';
[m2,n2]=size(file_data);
last_good_ne = 0;
last_good_Te = 0;
for i=1:m2;
    yr = int16(file_data(i,1)/1000);
    day = int16(file_data(i,1) - double(yr)*1000);
    sec = file_data(i,2)/1000;
    for iyr = REFYR:double(yr)-1;
        days = days +365;
        if (rem(iyr,4)==0), days = days +1;, end % leap year
    end
    days = days + double(day) - 1;
    UT_S = days * 86400;
    UT_S = UT_S + sec;
    UT_ETP(i) = UT_S;
    orbit(i) = file_data(i,3);
    if (file_data(i,5)<9999999)
        temp(i) = file_data(i,5);
        last_good_Te = temp(i);
    else
        temp(i) = last_good_ne;
    end
    if (file_data(i,6)<9999999)
        ne(i) = file_data(i,6);
        last_good_ne = ne(i);
    else
        ne(i) = last_good_ne;
    end
    Pe(i) = ne(i)*1.38*10^-23*temp(i)*1000000;
    UT_S = 0;
    days = 0;
end
OETP_limit = 1;
for i=1:m2;
    if (orbit(i) == target_orbit(i1));
        UT_ETP_local(OETP_limit) = UT_ETP(i);
        temp_local(OETP_limit) = temp(i);
        ne_local(OETP_limit) = ne(i);
        Pe_local(OETP_limit) = Pe(i);
        OETP_limit = OETP_limit + 1;
    end
end
OETP_limit = OETP_limit - 1;
fclose(fid1);
if (bg<=9), s2 =
sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB000%d.FFD\',bg));, end
if (bg>=10 & bg<=99), s2 =
sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB00%d.FFD\',bg));, end
if (bg>=100 & bg<=999), s2 =
sprintf('d:\\PVO\\DATA\\OMAG\\24S\\ORB%d.FFD\',bg));, end
if (bg>=1000), s2 =
fid1 = fopen(s2,'r');
\% begin process of reading in ephemeris data for the given orbit
count = 1;
eofstat = feof(fid1);
double x;

while(~eofstat)
    x = fread(fid1,1,'float64','b');
eofstat = feof(fid1);
    if(eofstat), break, end
    UT_MAG(count) = x;

    x = fread(fid1,1,'float32','s');
    B_X(count) = x;

    x = fread(fid1,1,'float32','s');
    B_Y(count) = x;

    x = fread(fid1,1,'float32','s');
    B_Z(count) = x;

    x = fread(fid1,1,'float32','s');
    B_T(count) = x;

\% data we do not need...
    x = fread(fid1,1,'float32','s');
    x = fread(fid1,1,'float32','s');
    x = fread(fid1,1,'float32','s');
    x = fread(fid1,1,'float32','s');
    x = fread(fid1,1,'float32','s');
    x = fread(fid1,1,'float32','s');
    x = fread(fid1,1,'float32','s');
    x = fread(fid1,1,'float32','s');
    x = fread(fid1,1,'float32','s');

    eofstat = feof(fid1);

    count = count + 1;
    if(count > 10000), break, end
end while(~eofstat)

\% since it is at EOF, we can close this
fclose(fid1);
% get proper max count
count = count - 1;
OMAG_limit = count;

% input the EPH data
% if (bg<=9), s2 = sprintf('d:\PVO\DATA\EPHEM\ORB000%d.FFD',bg);, end
if (bg>=10 & bg<=99), s2 = sprintf('d:\PVO\DATA\EPHEM\ORB00%d.FFD',bg);, end
if (bg>=100 & bg<=999), s2 = sprintf('d:\PVO\DATA\EPHEM\ORB0%d.FFD',bg);, end
if (bg>=1000), s2 = sprintf('d:\PVO\DATA\EPHEM\ORB%d.FFD',bg);, end

% open ephemeris data file
fid1 = fopen(s2,'r');

if fid1 == -1
    bg
    disp('arg!');
    return
endif fid1 == -1

% begin process of reading in ephemeris data for the given orbit
% count = 1;

eofstat = feof(fid1);
double x;

while(~eofstat)
    x = fread(fid1,1,'float64','b');
eofstat = feof(fid1);
if(eofstat),break, end

UT_EPH(count) = x;

x = fread(fid1,1,'float32','s');
X_VSO(count) = x;

x = fread(fid1,1,'float32','s');
Y_VSO(count) = x;

x = fread(fid1,1,'float32','s');
Z_VSO(count) = x;

x = fread(fid1,1,'float32','s');
ALT(count) = x;

x = fread(fid1,1,'float32','s');
S2A(count) = x;

x = fread(fid1,1,'float32','s');
PLONG(count) = x;
x = fread(fid1,1,'float32','s');
PLAT(count) = x;

x = fread(fid1,1,'float32','s');
SPX_VSO(count) = x;

x = fread(fid1,1,'float32','s');
SPY_VSO(count) = x;

x = fread(fid1,1,'float32','s');
SPZ_VSO(count) = x;

x = fread(fid1,1,'float32','s');
CLAT(count) = x;

x = fread(fid1,1,'float32','s');
CLONG(count) = x;

x = fread(fid1,1,'float32','s');
ELONG(count) = x;

x = fread(fid1,1,'float32','s');
RSUN(count) = x;

eofstat = feof(fid1);

count = count + 1;
if(count > 10000), break, end
end
while (~eofstat)

% since it is at EOF, we can close this
fclose(fid1);

% get proper max count
count = count - 1;
EPHEM_limit = count;

% Serialization
time = start_time(i1);

MAG_ct = 1;
ETP_ct = 1;
EPH_ct = 1;

c_t_s = 1;

while time < stop_time(i1)
    for i=2:OETP_limit
        if (UT_ETP_local(i) > time) & UT_ETP_local(i-1) <= time
            ETP_ct = i;
            break;
        end
    end
    for i=2:OMAG_limit
        if (UT_MAG(i) > time) & UT_MAG(i-1) <= time
            MAG_ct = i;
            break;
        end
    end
end
end
end
for i=2:EPHEM_limit
    if (UT_EPH(i) > time) & UT_MAG(i-1) <= time
        EPH_c = i;
        break;
    end
end

UT_ser(ct_s) = time;
BT_s(ct_s) = B_T(MAG_ct);
ne_s(ct_s) = ne_local(ETP_ct);
P_s(ct_s) = Pe_local(ETP_ct);
temp_s(ct_s) = temp_local(ETP_ct);
ALT_s(ct_s) = ALT(EPH_ct);

    time = time + 12; % min([UT_ETP_local(ETP_ct+1), UT_MAG(MAG_ct+1), UT_EPH(EPH_ct+1)])
    ct_s = ct_s + 1;
    if ct_s > 1000, break, end
end

ct_s = ct_s-1;

ALT_USSH = 0;
ALT_MSSH = 0;
ALT_LSSH = 0;
lneossipct = 0;
lne_s = 0;
lne_lsh = 0;
T USSH = 0;
T MSSH = 0;
T LSSH = 0;

USSH_CT = 1;
MSH_CT = 1;
LSSH_CT = 1;

for i=1:ct_s
    beta(i) = P_s(i) / (BT_s(i) * 10^-9)^2 / (2*4*pi*10^-7);
    beta_inv(i) = beta(i)^-1;
    log beta(i) = log(beta_inv(i));
    if (ALT_s(i) <= 400 & ALT_s(i) > 250 & beta_inv(i) < 1 & beta_inv(i) > 0) % 250 to 400 originally
        ALT_USSH(USSH_CT) = ALT_s(i) * 1000; % in meters
        ln e_s(USSH_CT) = log(ne_s(i));
        ln e USSH(USSH_CT) = log(Pe_s(i));
        beta_inv USSH(USSH_CT) = beta_inv(i);
        T USSH(USSH_CT) = temp_s(i);
        USH CT = USH CT + 1;
    end

    if (ALT s(i) <= 250 & ALT s(i) > 190 & beta_inv(i) < 1 & beta_inv(i) > 0) % 190 to 250 originally
        ALT_MSSH(MSSH_CT) = ALT_s(i) * 1000; % in meters
        ln e_MSSH(MSSH_CT) = log(ne_s(i));
        ln e MSSH(MSSH_CT) = log(Pe_s(i));
        beta_inv MSSH(MSSH_CT) = beta_inv(i);
        T MSSH(MSSH_CT) = temp_s(i);
        MSH CT = MSH CT + 1;
\texttt{ln_{\text{MSH}}(\text{MSH\_CT}) = \log(Pe\_s(i));}
\texttt{beta\_inv\_MSH(\text{MSH\_CT}) = beta\_inv(i);} 
\texttt{T\_MSH(\text{MSH\_CT}) = temp\_s(i);} 
\texttt{MSH\_CT = MSH\_CT + 1;}
\texttt{end}

\texttt{if (ALT\_s(i) <= 190 \& ALT\_s(i) >= 140 \& beta\_inv(i) < 1 \& beta\_inv(i) > 0) to 190 originally}
\texttt{ALT\_LSH(\text{LSH\_CT}) = ALT\_s(i)*1000; \% in meters}
\texttt{\%ln_{\text{LSH}}(\text{LSH\_CT}) = \log(ne\_s(i));}
\texttt{ln_{\text{LSH}}(\text{LSH\_CT}) = \log(Pe\_s(i));}
\texttt{beta\_inv\_LSH(\text{LSH\_CT}) = beta\_inv(i);} 
\texttt{T\_LSH(\text{LSH\_CT}) = temp\_s(i);} 
\texttt{LSH\_CT = LSH\_CT + 1;}
\texttt{end}
\texttt{end}

\texttt{USH\_CT = USH\_CT - 1;}
\texttt{MSH\_CT = MSH\_CT - 1;}
\texttt{LSH\_CT = LSH\_CT - 1;}

\texttt{fprintf(fid3,'\%d \n', bg);}
\texttt{if USH\_CT >= 4}
\texttt{USH = polyfit(ln_{\text{USH}}, ALT\_USH, 1);}
\texttt{\%\% quick algorithm found on the web to calc \text{R}^2}
\texttt{x = ln_{\text{USH}};}
\texttt{y = ALT\_USH;}
\texttt{ypred = polyval(USH,x);}
\texttt{dev = y - mean(y);}
\texttt{SST = sum(dev.^2);}
\texttt{resid = y - ypred;}
\texttt{SSE = sum(resid.^2);}
\texttt{normr = sqrt(SSE);}
\texttt{Rsq = abs(1 - SSE/SST);}
\texttt{if (USH(1)<0 \& USH(1)>=-1000000)}
\texttt{USH\_SUM = USH\_SUM + USH(1)*Rsq;}
\texttt{USH\_RSQ\_SUM = USH\_RSQ\_SUM + Rsq;}
\texttt{end}
\texttt{fprintf(fid3,\'\%f \%f \%d \n', USH(1), Rsq, USH\_CT);}
\texttt{for i3=1:USH\_CT}
\texttt{fprintf(fid3,\'\%f \%f \%f \%f \n', ln_{\text{USH}}(i3), ALT\_USH(i3),
T\_USH(i3), beta\_inv\_USH(i3));}
\texttt{end}
\texttt{else}
\texttt{fprintf(fid3,'-9999999 \n');}
\texttt{end}
\texttt{if MSH\_CT >= 4}
\texttt{MSH = polyfit(ln_{\text{MSH}}, ALT\_MSH, 1);}
\texttt{\%\% quick algorithm found on the web to calc \text{R}^2}
\texttt{x = ln_{\text{MSH}};}
\texttt{y = ALT\_MSH;}
\texttt{ypred = polyval(MSH,x);}
\texttt{dev = y - mean(y);}
SST = sum(dev.^2);
resid = y - ypred;
SSE = sum(resid.^2);
normr = sqrt(SSE);
Rsq = abs(1 - SSE/SST);

if (MSH(1)<0 & MSH(1)>=-1000000)
    MSH_SUM = MSH_SUM + MSH(1)*Rsq;
    MSH_RSQ_SUM = MSH_RSQ_SUM + Rsq;
end

fprintf(fid3,'%f %f %d \n', MSH(1), Rsq, MSH_CT);
for i3=1:MSH_CT
    fprintf(fid3,'%f %f %f %f \n', lne_MSH(i3), ALT_MSH(i3),
    T_MSH(i3), beta_inv_MSH(i3));
end
else
    fprintf(fid3,'-9999999 \n');
end

if LSH_CT >= 4
    LSH = polyfit(lne_LSH, ALT_LSH,1);
    x = lne_LSH;
    y = ALT_LSH;
    ypred = polyval(LSH,x);
    dev = y - mean(y);
    SST = sum(dev.^2);
    resid = y - ypred;
    SSE = sum(resid.^2);
    normr = sqrt(SSE);
    Rsq = abs(1 - SSE/SST);

if (LSH(1)<0 & LSH(1)>=-1000000)
    LSH_SUM = LSH_SUM + LSH(1)*Rsq;
    LSH_RSQ_SUM = LSH_RSQ_SUM + Rsq;
end

fprintf(fid3,'%f %f %d \n', LSH(1), Rsq, LSH_CT);
for i3=1:LSH_CT
    fprintf(fid3,'%f %f %f %f \n', lne_LSH(i3), ALT_LSH(i3),
    T_LSH(i3), beta_inv_LSH(i3));
end
else
    fprintf(fid3,'-9999999 \n');
end
fprintf(fid3,'\n');

BETA_FLAG = 0;
for i=1:ct_s
    if (beta_inv(i) >= 5 & ~BETA_FLAG)
        BETA_FLAG = 1;
        fprintf(fid2, '%d %f ', bg, UT_ser(i));
    end
    if (beta_inv(i) < 4.7 & BETA_FLAG)
        BETA_FLAG = 0;
        fprintf(fid2, '%f ', UT_ser(i));
    end
end
fprintf(fid2,'\n');
end
end
if BETA_FLAG
fprintf(fid2, '%f ', UT_ser(ct_s));
fprintf(fid2,'\n');
BETA_FLAG = 0;
end

% create synchronized orbital data file
if (bg<=9), s2 = sprintf('d:\FVO\SYNC\SORB00%d.txt',bg);, end
if (bg>=10 & bg<=99), s2 = sprintf('d:\FVO\SYNC\SORB00%d.txt',bg);, end
if (bg>=100 & bg<=999), s2 = sprintf('d:\FVO\SYNC\SORB0%d.txt',bg);, end
if (bg>=1000), s2 = sprintf('d:\FVO\SYNC\SORB%d.txt',bg);, end
fid4 = fopen(s2,'a');
if fid4 == -1
bg
disp('arg!');
return
eendif fid1 == -1
for i=1:ct_s
fprintf(fid4,'%f %f %f %f %f %f \n', UT_ser(i), BT_s(i),
ne_s(i), Pe_s(i), temp_s(i), beta_inv(i), ALT_s(i));
end
fclose(fid4);
end % orbit loop check
if (USH_RSQ_SUM ~=0), Upper_Scale = USH_SUM/USH_RSQ_SUM, else
Upper_Scale = 0, end
if (MSH_RSQ_SUM ~=0), Middle_Scale = MSH_SUM/MSH_RSQ_SUM, else
Middle_Scale = 0, end
if (LSH_RSQ_SUM ~=0), Lower_Scale = LSH_SUM/LSH_RSQ_SUM, else
Lower_Scale = 0, end
fclose(fid2);
fclose(fid3);

fprintf(fid0,'%d ', bg);
fprintf(fid0,'%f ', start);
fprintf(fid0,'%f ', stop);
fprintf(fid0,'%f ', min_alt);
fprintf(fid0,'\n');
Time At Altitude, Time In Trough Script

mod_ephem_alt_bin_filter.m
%open target orbit files
%fid0 = fopen('d:\PVO\SEASON 1\windows.txt');
fid0 = fopen('d:\PVO\SEASON 2\windows.txt');
%fid0 = fopen('d:\PVO\SEASON 3\windows.txt');
%f1 = fopen('d:\PVO\SEASON 4\windows.txt');
orbit = fscanf(fid0, '%d %g %g', [3, inf]);
orbit = orbit';
[m,n]=size(orbit)
close(fid0);
%
% initialize altitude bins
delta_alt = 16;
alct = (400 - 140)/delta_alt + 1;
for ia=1:alct
    alt_bin(ia) = 0;
    alt_lim(ia) = 140 + delta_alt*(ia-1);
end
for bg = 1:m;
    if (orbit(bg,1)<=9), s2 = sprintf('d:\PVO\DATA\EPHEM\ORB000%d.FFD',orbit(bg,1));, end
    if (orbit(bg,1)>=10 & orbit(bg,1)<=99), s2 = sprintf('d:\PVO\DATA\EPHEM\ORB00%d.FFD',orbit(bg,1));, end
    if (orbit(bg,1)>=100 & orbit(bg,1)<=999), s2 = sprintf('d:\PVO\DATA\EPHEM\ORB0%d.FFD',orbit(bg,1));, end
    if (orbit(bg,1)>=1000), s2 = sprintf('d:\PVO\DATA\EPHEM\ORB%d.FFD',orbit(bg,1));, end

    %open ephemeris data file
    fid1 = fopen(s2,'r');
    if fid1 == -1
        bg
        disp('arg!');
        return
    end
    if fid1 == -1

    % begin process of reading in ephemeris data for the given orbit
    count = 1;
    eofstat =feof(fid1);
    double x;
    while(~eofstat)
        x = fread(fid1,1,'float64','b');
        eofstat = feof(fid1);
        if(eofstat),break, end
        UT(count) = x;
        x = fread(fid1,1,'float32','s');
        X_VSO(count) = x;
        x = fread(fid1,1,'float32','s');
        Y_VSO(count) = x;
    end
end

end

x = fread(fid1,1,'float32','s');
Z_VSO(count) = x;

x = fread(fid1,1,'float32','s');
ALT(count) = x;

x = fread(fid1,1,'float32','s');
SZA(count) = x;

x = fread(fid1,1,'float32','s');
PLONG(count) = x;

x = fread(fid1,1,'float32','s');
PLAT(count) = x;

x = fread(fid1,1,'float32','s');
SPX_VSO(count) = x;

x = fread(fid1,1,'float32','s');
SPY_VSO(count) = x;

x = fread(fid1,1,'float32','s');
SPZ_VSO(count) = x;

x = fread(fid1,1,'float32','s');
CLAT(count) = x;

x = fread(fid1,1,'float32','s');
CLONG(count) = x;

x = fread(fid1,1,'float32','s');
ELONG(count) = x;

x = fread(fid1,1,'float32','s');
RSUN(count) = x;

eofstat = feof(fid1);

count = count + 1;
if(count > 10000), break, end$if
eendwhile(~eofstat)

%since it is at EOF, we can close this
fclose(fid1);

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

%get proper max count
count = count - 1;
limit = count;

%initialize solar zenith and altitude flags
SZA_FLAG = 0;
ALT_FLAG = 0;
EDGE_UP = 0;
EDGE_DOWN = 0;
BOTH = 0;
double start;
start = 0;
double stop;
stop = 0;
min_alt = 10001;

for ia = 1:alct
    ALT_FLAG(ia) = 0;
    ALT_start(ia) = 0;
    ALT_stop(ia) = 0;
end

%% now time to test bins
for nt = 2:limit;
    if (orbit(bg,2)>=UT(nt) & UT(nt)<=orbit(bg,3))
        for ia = 1:alct-1;
            %i+1 to i -- e.g. 156 to 140,
            % see if PVO is just going below upper level of bin (ia+1)
            if (ALT(nt)<=alt_lim(ia+1) & ALT(nt-1)>alt_lim(ia) &
               ~ALT_FLAG(ia+1))
                ALT_FLAG(ia+1) = 1;
                ALT_start(ia+1) = abs(alt_lim(ia+1)-ALT(nt)) / abs(ALT(nt-1)-ALT(nt)) * abs(UT(nt)-UT(nt-1)) + UT(nt-1);
            end
        end
        % see if PVO is going below lower level if bin (ia)
        if (ALT(nt)<=alt_lim(ia) & ALT(nt-1)>alt_lim(ia) &
            ALT_FLAG(ia+1))
            ALT_FLAG(ia+1) = 0;
            ALT_stop(ia+1) = abs(alt_lim(ia)-ALT(nt)) / abs(ALT(nt-1)-ALT(nt)) * abs(UT(nt)-UT(nt-1)) + UT(nt-1);
            alt_bin(ia+1) = alt_bin(ia+1) + (ALT_stop(ia+1) - ALT_start(ia+1));
        end
    end
    %see if PVO went below upper limit, but never breaks lower limit
    if (ALT(nt)>=alt_lim(ia+1) & ALT(nt-1)<alt_lim(ia+1) &
        ALT_FLAG(ia+1))
        ALT_FLAG(ia+1) = 0;
        ALT_stop(ia+1) = abs(alt_lim(ia+1)-ALT(nt-1)) / abs(ALT(nt-1)-ALT(nt)) * abs(UT(nt)-UT(nt-1)) + UT(nt-1);
        alt_bin(ia+1) = alt_bin(ia+1) + (ALT_stop(ia+1) - ALT_start(ia+1));
    end
    % coming from below bin level...
    if (ALT(nt)>=alt_lim(ia) & ALT(nt-1)<alt_lim(ia) &
        ~ALT_FLAG(ia+1))
        ALT_FLAG(ia+1) = 1;
        ALT_stop(ia+1) = abs(alt_lim(ia)-ALT(nt-1)) / abs(ALT(nt-1)-ALT(nt)) * abs(UT(nt)-UT(nt-1)) + UT(nt-1);
        alt_bin(ia+1) = alt_bin(ia+1) + (ALT_stop(ia+1) - ALT_start(ia+1));
    end
end % for ia
end% if orbit
    end%for nt = 1:limit;
end%for bg

fid2 = fopen('d:\PVO\mod_season_bin_count.txt', 'a');
fprintf(fid2, '%d %d', orbit(1,1), orbit(m,1));
fprintf(fid2, '
');
for ia=1:alct;
    fprintf(fid2, '%d ', alt_lim(ia));
    fprintf(fid2, '%f ', alt_bin(ia));
    fprintf(fid2, '
');
end
fclose(fid2);
Event Detection Script

OEFD.m
%open target orbit files
%fid0 = fopen('d:\PVO\SEASON 1\windows.txt');
fid0 = fopen('d:\PVO\SEASON 2\windows.txt');
%fid0 = fopen('d:\PVO\SEASON 3\windows.txt');
%fid0 = fopen('d:\PVO\SEASON 4\windows.txt');
file_data = fscanf(fid0, '%d %g %g', [3,inf]);
file_data = file_data';
[m2,n2]=size(file_data)
close(fid0);

for i=1:m2

    bg = file_data(i,1)%target orbit
    start_time = file_data(i,2);
    stop_time = file_data(i,3);
    %530 453721161.074000 453721245.074000

    % open file for input
    if (bg>0 & bg<=9), s2 = sprintf('d:\PVO\OEFD\ORB00%d.tab',bg);,
    end
    if (bg>9 & bg<=99), s2 = sprintf('d:\PVO\OEFD\ORB0%d.tab',bg);,
    end
    if (bg>99 & bg<=999), s2 = sprintf('d:\PVO\OEFD\ORB%d.tab',bg);,
    end
    if (bg>999), s2 = sprintf('d:\PVO\OEFD\ORB%d.tab',bg);,
end

    fid1 = fopen(s2,'r')

    REFYR = 1966;
    MN = [31,28,31,30,31,30,31,30,31,30,31,30,31,30,31];

    eofstat = feof(fid1);
count = 1;
watch_count = 1;
last_good_E100H = 0;
last_good_E730H = 0;
last_good_E5400H = 0;
last_good_E30000H = 0;

while(~eofstat)

    UT_S = 0;
days = 0;
    yr = str2num(fscanf(fid1, '%c ', [1,4]));
junk = fscanf(fid1, '%c', [1,1]);
mon = str2num(fscanf(fid1, '%c ', [1,2]));
junk = fscanf(fid1, '%c', [1,1]);
day = str2num(fscanf(fid1, '%c ', [1,2]));
junk = fscanf(fid1, '%c', [1,1]);
hour = str2num(fscanf(fid1, '%c ', [1,2]));
junk = fscanf(fid1, '%c', [1,1]);
min = str2num(fscanf(fid1, '%c ', [1,2]));
junk = fscanf(fid1, '%c', [1,1]);
sec = str2num(fscanf(fid1, '%c ', [1,2]));
junk = fscanf(fid1, '%c', [1,1]);
msec = str2num(fscanf(fid1, '%c ', [1,3]));
msec = msec/1000;
for iyr = REFYR:yr-1;
    days = days +365;
    if (rem(iyr,4)==0), days = days + 1;, end % leap year
end$for
for imn = 1:mos-1;
    days = days + MN(imn);
    if (rem(yr,4)==0 & imn == 2), days = days + 1;, end % local
    leap year
end
    days = days + day - 1;
UT_S = days * 86400;
UT_S = UT_S + hr * 3600 + min * 60 + sec + msec;

if(UT_S > stop_time), break, end

UT(count) = UT_S;

junk = fscanf(fid1, '%c', [1,1]);

E100H(count) = fscanf(fid1, '%g', [1,1]);
if (E100H(count)<99999999)
    last_good_E100H = E100H(count);
else
    E100H(count) = last_good_E100H;
end

junk = fscanf(fid1, '%c', [1,1]);

E730H(count) = fscanf(fid1, '%g', [1,1]);
if (E730H(count)<99999999)
    last_good_E730H = E730H(count);
else
    E730H(count) = last_good_E730H;
end

junk = fscanf(fid1, '%c', [1,1]);

E5400H(count) = fscanf(fid1, '%g', [1,1]);
if (E5400H(count)<99999999)
    last_good_E5400H = E5400H(count);
else
    E5400H(count) = last_good_E5400H;
end

junk = fscanf(fid1, '%c', [1,1]);

E30000H(count) = fscanf(fid1, '%g', [1,1]);
if (E30000H(count)<99999999)
    last_good_E30000H = E30000H(count);
else
    E30000H(count) = last_good_E30000H;
end
junk = fscanf(fid1, '%g', [1,1]);
junk = fscanf(fid1, '%c', [1,1]);
junk = fscanf(fid1, '%g', [1,1]);
junk = fscanf(fid1, '%c', [1,1]);
junk = fscanf(fid1, '%g', [1,1]);
junk = fscanf(fid1, '%c', [1,1]);
%yr = str2num(fscanf(fid1, '%c ', [1,4]))
eofstat =feof(fid1);
if (UT_S>=start_time & UT_S <=stop_time)
    count = count + 1;
end

watch_count = watch_count + 1;
if(rem(watch_count,50000)==0),watch_count,end
if(watch_count > 200000), break, end
end%while

%done with file
fclose(fid1);

%get proper max count
count = count - 1
limit = count;

count = 1;

fid2 = fopen('d:\PVO\EVENTS\events.txt','a');

E100H_flag = 0;
E730H_flag = 0;
E5400H_flag = 0;
E30000H_flag = 0;
E5400Hpause_filter = 1;
E30000Hpause_filter = 1;
MAG = 0.5;%1.0;%1.01.5;

for i=2:limit
    if (E5400H(i) > 0 & E5400H(i-1) > 0 & E5400H(i) > E5400H(i-1))
        if (log10(E5400H(i)/E5400H(i-1))>MAG)
            E5400H_flag = 1
        end
    end
    if (E30000H(i) > 0 & E30000H(i-1) > 0 & E30000H(i) > E30000H(i-1))
        if (log10(E30000H(i)/E30000H(i-1))>MAG)
            E30000H_flag = 1;
        end
    end
end

if(~E5400H_flag & ~E30000H_flag)
    if (E100H(i) > 0 & E100H(i-1) > 0 & E100H(i) > E100H(i-1))
        if (~E100H_flag)
            E100H_flag = 1
        end
    end
end
if (log10(E100H(i)/E100H(i-1)) > MAG)
    E = 100
    UT(i)
    fprintf(fid2, '%d 100 %f %g %g', bg, UT(i), E100H(i),
    E100H(i-1));
    fprintf(fid2, '\n');
end
end
if (E730H(i) > 0 & E730H(i-1) > 0 & E730H(i) > E730H(i-1))
    if (log10(E730H(i)/E730H(i-1)) > MAG)
        E = 730
        UT(i)
        fprintf(fid2, '%d 730 %f %g %g', bg, UT(i), E730H(i),
        E730H(i-1));
        fprintf(fid2, '\n');
    end
end
if (E5400H_flag)
    E5400Hpause_filter = E5400Hpause_filter + 1;
    if (E5400H_pause_filter >= 3)
        E5400H_flag = 0;
        E5400H_pause_filter = 1;
    end
end
if (E30000H_flag)
    E30000Hpause_filter = E30000Hpause_filter + 1;
    if (E30000H_pause_filter >= 3)
        E30000H_flag = 0;
        E30000H_pause_filter = 1;
    end
end
end
fclose(fid2);
end %for
Event Ephemeris Lookup Script

event_ephem_filter.m
% modified ephemeris script to look up altitudes for given events

fid0 = fopen('d:\PVO\SEASON 2\S2_005events.txt','r');
file_data = fscanf(fid0, '%d %d %g %g %g', [5,inf]);
fclose(fid0);

% open file for output of data
fid2 = fopen('d:\PVO\SEASON 2\S2_005events_altst.txt','a');

for i=1:m2
  bg = file_data(i,1)
  if (bg<9), s2 = sprintf('d:\PVO\DATA\EPHEM\ORB00%d.FFD',bg);
  end
  if (bg>=10 & bg<99), s2 = sprintf('d:\PVO\DATA\EPHEM\ORB00%d.FFD',bg);
  end
  if (bg>=100 & bg<=999), s2 = sprintf('d:\PVO\DATA\EPHEM\ORB0%d.FFD',bg);
  end
  if (bg>=1000), s2 = sprintf('d:\PVO\DATA\EPHEM\ORB%d.FFD',bg);
  end

% open ephemeris data file
fid1 = fopen(s2,'r');

if fid1 == -1
  bg
  disp('arg!');
  return
end if fid1 == -1

% begin process of reading in ephemeris data for the given orbit
count = 1;

eofstat = feof(fid1);
double x;

while(~eofstat)
  x = fread(fid1,1,'float64','b');
  eofstat = feof(fid1);
  if(eofstat),break, end if
  UT(count) = x;
  X_VSO(count) = x;
  Y_VSO(count) = x;
  Z_VSO(count) = x;

end while

x = fread(fid1,1,'float32','s');
ALT(count) = x;

x = fread(fid1,1,'float32','s');
SZA(count) = x;

x = fread(fid1,1,'float32','s');
PLONG(count) = x;

x = fread(fid1,1,'float32','s');
PLAT(count) = x;

x = fread(fid1,1,'float32','s');
SPX_VSO(count) = x;

x = fread(fid1,1,'float32','s');
SPY_VSO(count) = x;

x = fread(fid1,1,'float32','s');
SPZ_VSO(count) = x;

x = fread(fid1,1,'float32','s');
CLAT(count) = x;

x = fread(fid1,1,'float32','s');
CLONG(count) = x;

x = fread(fid1,1,'float32','s');
ELONG(count) = x;

x = fread(fid1,1,'float32','s');
RSCN(count) = x;

eofstat = feof(fid1);

count = count + 1;
if(count > 10000), break, end
end
while(~eofstat)

%since it is at EOF, we can close this
fclose(fid1);

%get proper max count
count = count - 1;
limit = count;

% now time to find altitude for event
for nt = 2:limit;
    if (UT(nt)>file_data(i,3) & UT(nt-1)<file_data(i,3))
        %linear interpolation
        E_ALT = (file_data(i,3)-UT(nt-1))/(UT(nt)-UT(nt-1))*(ALT(nt)-
        ALT(nt-1))+ALT(nt-1);
        fprintf(fid2, '%d %d %f %g %g %f', bg, file_data(i,2),
        file_data(i,3), file_data(i,4), file_data(i,5), E_ALT);
        fprintf(fid2, '\n');
        break
end\%if
end\%for nt = 1:limit;

end\%for

fclose(fid2);
Event OMAG Lookup Script

event_OMAG_filter.m
% modified ephemeris script to look up B field for given events

% open events files

fid0 = fopen('d:\PVO\SEASON 2\S2_05events.txt','r');
file_data = fscanf(fid0, '%d %d %g %g %g', [5, inf]);
fclose(fid0);

% open file for output of data
fid2 = fopen('d:\PVO\SEASON 2\S2_05events_B.txt','a');

for i=1:m2
    bg = file_data(i,1)

    % input the OMAG data

    if (bg<=9), s2 = sprintf('d:\PVO\DATA\OMAG\24S\ORB000%d.FFD',bg), end
    if (bg>=10 & bg<=99), s2 = sprintf('d:\PVO\DATA\OMAG\24S\ORB00%d.FFD',bg), end
    if (bg>=1000), s2 = sprintf('d:\PVO\DATA\OMAG\24S\ORB%d.FFD',bg), end
    fid1 = fopen(s2,'r');
    % begin process of reading in ephemeris data for the given orbit
    count = 1;
eofstat = feof(fid1);
double x;

    while(~eofstat)
        x = fread(fid1,1,'float64','b');
eofstat = feof(fid1);
        if(eofstat), break, end
        UT(count) = x;

        x = fread(fid1,1,'float32','s');
        B_X(count) = x;

        x = fread(fid1,1,'float32','s');
        B_Y(count) = x;

        x = fread(fid1,1,'float32','s');
        B_Z(count) = x;

        x = fread(fid1,1,'float32','s');
        B_T(count) = x;
    end
end
%data we do not need...
x = fread(fid1,1,'float32','s');
x = fread(fid1,1,'float32','s');
x = fread(fid1,1,'float32','s');
x = fread(fid1,1,'float32','s');
x = fread(fid1,1,'float32','s');
x = fread(fid1,1,'float32','s');
ex = fread(fid1,1,'float32','s');
ex = fread(fid1,1,'float32','s');
ex = fread(fid1,1,'float32','s');
ex = fread(fid1,1,'float32','s');
ectstat = feof(fid1);

    count = count + 1;
    if(count > 10000), break, endif
endwhile(~ectstat)

%since it is at EOF, we can close this
fclose(fid1);

%get proper max count
count = count - 1;
limit = count;

% now time to find altitude for event
for nt = 2:limit;
    if (UT(nt) > file_data(i,3) & UT(nt-1) <= file_data(i,3))
        %linear interpolation
        E_B = (file_data(i,3)-UT(nt-1))/(UT(nt)-UT(nt-1))*B_T(nt)-
             B_T(nt-1));
        fprintf(fid2, '%d %d %f %g %g %f', bg, file_data(i,2),
                file_data(i,3), file_data(i,4), file_data(i,5), E_B);
        fprintf(fid2, '
');
        break
    endif
endfor
fclose(fid2);
Event Counter (in altitude bins) Script

event_count.m
%open events files
%input
fid0 = fopen('d:\PVO\SEASON 2\S2_005events_alts.txt','r');
%output
fid1 = fopen('d:\PVO\SEASON 2\S2_005_count.txt','a');

file_data = fscanf(fid0, '%d %d %g %g %g %f', [6,inf]);
% orbit, channel, time, final E, initial E, alt
file_data = file_data';
[m2,n2]=size(file_data);
close(fid0);

for i=1:m2
    CH(i) = file_data(i,2);
    UT(i) = file_data(i,3);
    E_f(i) = file_data(i,4);
    E_i(i) = file_data(i,5);
    ALT(i) = file_data(i,6);
end

% initialize altitude bins
delta_alt = 16;
alct = (400 - 140)/delta_alt + 1;

for ia=1:alct
    alt_bin(ia) = 0;
    alt_lim(ia) = 140 + delta_alt*(ia-1);
end

wave_ct = 0;
pack1_ct = 0;
pack2_ct = 0;
pack3_ct = 0;
pack4_ct = 0;

for i=1:m2
for ia=1:alct-1
    if (ALT(i)<alt_lim(ia+1) & ALT(i)>=alt_lim(ia))
        alt_bin(ia) = alt_bin(ia) + 1;
        pack1_ct = pack1_ct + 1;
    end
end

% if (ALT(i)<300) & alt_lim(int16(alct)))
if i>1
    if ((UT(i)<=(UT(i-1)+0.25)) & CH(i)==100 & CH(i-1)==100)
        wave_ct = wave_ct + 1;
        if wave_ct == 1
            pack2_ct = pack2_ct + 1;
        end
        if wave_ct == 2
            pack3_ct = pack3_ct + 1;
            pack2_ct = pack2_ct - 1;
        end
        if wave_ct == 3
            pack4_ct = pack4_ct + 1;
    end
end
pack3_ct = pack3_ct - 1;
end
else
    wave_ct = 0;
end
end
end

for ia=1:alct
    fprintf(fid1, '%d %d', alt_bin(ia), alt_lim(ia));
    fprintf(fid1, '
');
end
fclose(fid1);

pack1_ct
pack2_ct
pack3_ct
pack4_ct
Electron Density Analysis Script

min_ne_seasonX.m
fid1 = fopen('D:\PVO\season 2 orbits.txt');

scan_data = fscanf(fid1, '%d %f %f %f', [4,inf]);
scan_data = scan_data';
[m1,n1]=size(scan_data);

for i=1:m1;
    target_orbit(i) = scan_data(i,1);
end
fclose(fid1);

counters for electron densities
O_min_ne = 9999999;
O_avg_ne = 0;
O_avg_ne_sum = 0;
O_avg_ne_CT = 0;

I_min_ne = 9999999;
I_avg_ne = 0;
I_avg_ne_sum = 0;
I_avg_ne_CT = 0;

%% !!!! don't forget to change upper limit, and target orbit!!!!
for ii=1:m1
    bg = target_orbit(ii);
    
    % input complete sync'd data for orbit
    if (bg<=9), s2 = sprintf('d:\PVO\SYNC\SORB000%d.txt',bg), end
    if (bg>10 & bg<=99), s2 = sprintf('d:\PVO\SYNC\SORB000%d.txt',bg), end
    if (bg>100 & bg<=999), s2 = sprintf('d:\PVO\SYNC\SORB00%d.txt',bg), end
    if (bg>1000), s2 = sprintf('d:\PVO\SYNC\SORB%d.txt',bg), end
    fid1 = fopen(s2,'r');
    file_data = fscanf(fid1, '%f %f %f %f %f %f %f', [7,inf]);
    file_data = file_data';
    [m2,n2]=size(file_data);
    for i=1:m2;
        UT_ser(i) = file_data(i,1);
        BT_s(i) = file_data(i,2);
        ne_s(i) = file_data(i,3);
        Pe_s(i) = file_data(i,4);
        temp_s(i) = file_data(i,5);
        beta_inv(i) = file_data(i,6);
        ALT_s(i) = file_data(i,7);
    end
    fclose(fid1);

    for i=1:m2
        if (ALT_s(i)<=190 & beta_inv(i) < 1) %outside trough

            % processing here

        end
    end
end
if ne_s(i) < O_min_ne, O_min_ne = ne_s(i);, end;
    O_avg_ne_sum = O_avg_ne_sum + ne_s(i);
    O_avg_ne_CT = O_avg_ne_CT + 1;
end

    if (ALT_s(i)<=190 & beta_inv(i) > 3) inside trough
        if ne_s(i) < I_min_ne, I_min_ne = ne_s(i);, end;
        I_avg_ne_sum = I_avg_ne_sum + ne_s(i);
        I_avg_ne_CT = I_avg_ne_CT + 1;
    end
end

end %orbit loop check

if O_avg_ne_CT > 0
   O_avg_ne = O_avg_ne_sum/O_avg_ne_CT
   O_min_ne
end

if I_avg_ne_CT > 0
   I_avg_ne = I_avg_ne_sum/I_avg_ne_CT
   I_min_ne
end