Plasma Densities and Satellite Potentials

A Study of EUV Interaction with the Cluster Spacecraft

Erik Winkler
Swedish Institute of Space Physics Uppsala

18th June 2007

Abstract

The satellite potential is dependent of both the plasma density and the solar EUV flux. Since the Cluster satellites have been operational for almost half a solar cycle, i.e. six years, the decrease in the EUV energy flux should be evident in the collected Cluster data. It is shown that the photoelectron current follows the solar cycle as expected.

Three major investigations have been made for this master thesis: *i*) The electronic components have been examined for continous time degredations. *ii*) The photoelectron yield function was investigated. *iii*) The relation between the three parameters: satellite potential, plasma density and the EUV energy flux, was investigated.

Using the bias voltage sweeps done by Cluster, the continous time degredation of the electronics were examined. It was shown that this time degradation is of such a small magnitude that it can safely be neglected.

We attempted to dervive a photoelectron yield function by studying how the photoemission saturation current varied with the solar UV spectrum shape and intensity. The yield functions derived in this way are not physically meaningful, but we can show that combining laboratory determined yield functions and UV measurements result in the probe current records reasonably consistent with observed values.

To improve the accuracy of the density determinations using the spacecraft potential, we introduced the UV flux variations into the conversion. As TIMED data are not available before 2003, we instead used two proxies: the probe photoemission saturation current and the $F_{10.7}$ index. Both give improved performance, though the photocurrent is shown to be best.

"För att vi har räknat det!!"
-Volodymyr Mazorchuk

Contents

Chapter 1

Introduction

Large parts of the magnetosphere are filled with a plasma of very low density, below one particle per cubic centimeter. In such plasmas, a sunlit spacecraft attains a positive electrostatic potential due to the photoelectric effect. The satellite potential, V_{sc}, depends on the plasma density (n). Since the satellite potential can be obtained from measurements done by double-probe instruments on the spacecraft, it is possible to measure n with such an instrument. When the potential is induced by the photons coming from the sun the potential should have a correlation with the solar cycle, i.e. the variation in the sun's outflow of photons should show a variation in the satellite's potential, as well. Thus V_{sc} should be dependendent of both n, and the solar flux.

To aquire these dependencies data have been collected from the four Cluster II satellites and the TIMED satellite. The former is used to get the spacecraft potentials, photo-saturation currents and the plasma densities, and the latter for information about the variations of the extreme ultraviolet (EUV) radiation from the sun. All of these satellites have been operational for almost half a solar cycle. The variation from the solar EUV flux should therefore , if there is such a relation, be evident in the Cluster data. The knowlege of this dependence should give an improved way to calculate n with V_{sc} and the solar flux.

Before anything else is done, we would like to evaluate the quality of the measuring components. Electronics that are operational for a longer period are bound to detoriate, wich is the case for most satellites around earth. The magnitude of this deterioration over time is examined in Chapter 4. The failing of components will have an affect on the reliability of the measurements and the introduced errors must be corrected.

In the EUV spectra some wavelengths might be of greater importance for the photo-electron currents, and thus for the satellite potential, than others. This yield function is investigated in in Chapter 5. If the yield function is out of reach for us then we may have to use other proxies for the EUV radiation from the sun. The photoelectron current is in a sence the power input on the satellite from the sun and may be used as an EUV radiation proxy.

The main purpose of this thesis is to investigate the relation between the three parameters V_{sc}, n and the solar EUV flux variations. This investigation is done in Chapter 6.

Chapter 2

The Satellites

Two satellite missions have been used to get the needed information about the enviroment around the earth, Cluster II and TIMED, where the first is actually four satellites. This chapter is an overview of these two missions.

2.1 The Satellites of Cluster II

The Cluster II mission consists of four identical spacecrafts that fly in a tetrahedral (triangular pyramid) formation. They fly in a polar elliptical orbit with a perigee of 4 R_E and a apogee of 19.6 R_E[1]. It has an orbital period of 57 hours. See the two Figures 2.1(a) and 2.1(b). These four satellites were named Rumba, Samba, Salsa and Tango.

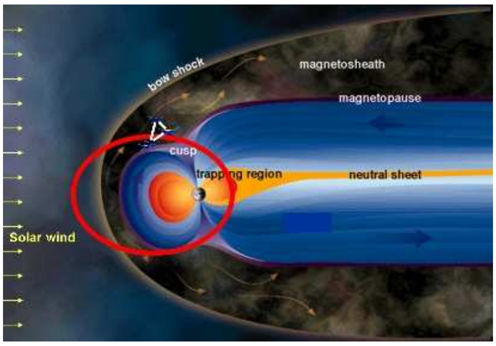
This mission's predecessor, Cluster I, met its premature end only 37 seconds after launch in 1996, when the automatic destruction sequence were executed due to aerodynamical problems. It was decided that four new spacecrafts would be built. These satellites would have the exact set of instruments as the initial four. They were completed in less than three years and were launched in the summer of 2000.

The Cluster II mission was originally funded for two years of operation but is still operational thanks to extensions of the mission.

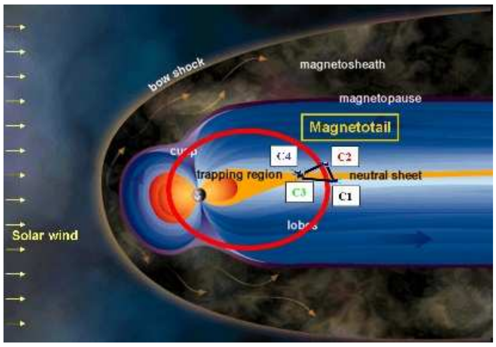
2.1.1 Mission Objectives

A main goal of the Cluster II mission is to study the small-scale structures in the plasmas surrounding the earth; both spatial and temporal. Some regions of intrest are *the solar wind and the bow shock*, *the magnetosphere*, *the polar caps*, *the magnetotail*, and *the auroral zone*.

[1] The used value of the radius of Earth (R_E) is 6371.2 km



(a)



(b)

Figure 2.1: The orbit of the Cluster II satellites in spring (a) and autumn (b). The spacecraft seperation in (b) is exaggerated to be more evident

2.1.2 Instruments

The Cluster satellites have a set of 11 instruments each. The composition of instruments are the same on all the four satellites. The two instruments mentioned hereafter are the two used in this report.

EFW, *Electric Field and Wave experiment*, (*Gustafsson* [8]) uses electrostatic probes on four 41 meters long wire booms to measure the electric field to study plasma convection and waves. It can take up to 36 000 samples per second of the electric field and fluctuations in the plasma density. The radius of the probes are 4 cm.

CIS, *Cluster Ion Spectrometry experiment*, (*Rème et al* [11]) analyses the composition, mass and distribution functions of the ions in the nearby plasmas and in the solar wind during each four second spin of the spacecraft. It consists of two different instruments, a Hot Ion Analyser (HIA) and a time-of-flight plasma Composition and Distribution Function analyser (CODIF), plus a sophisticated instrument control and data processing system, which permits extensive onboard data processing.

2.1.3 Data Sources

CIS data have been obtained from the Cluster Science Data System Prime Parameter database (CSDS PP) [1], containing CIS density data at spin resolution. For EFW, we have accessed the raw data using the MATLAB interface to the Isdat data system.

2.2 TIMED

TIMED, *Thermosphere, Ionosphere, Mesosphere, Energetics and Dynamics*, has a circular orbit 625 km above the earth's surface, Figure 2.2. It was launched in December 7, 2001 from Vandenberg Air Force base in California. It has an orbital period of 97 minutes.

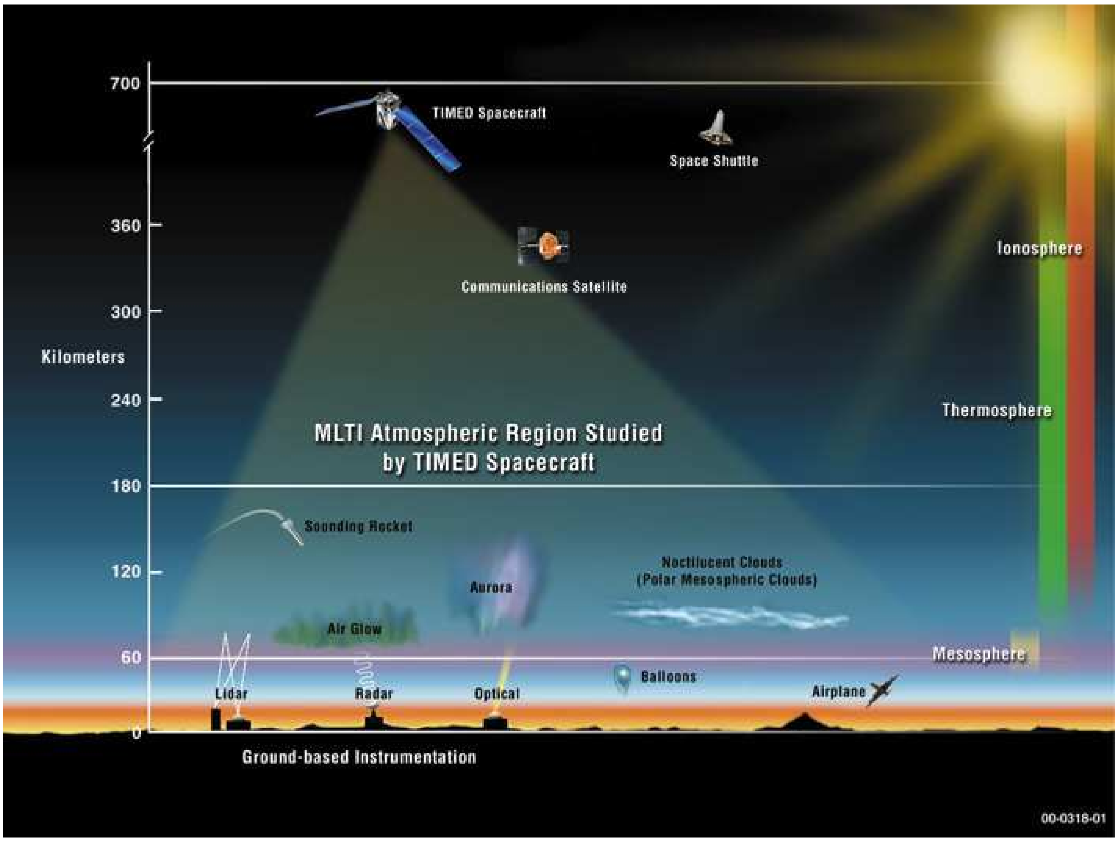


Figure 2.2: The location of the TIMED satellite and the regions it studies

2.2.1 Mission Objectives

In an effort to thoroughly examine the MLTI, *Mesosphere and Lower Thermosphere/ Ionosphere*, region the satellite TIMED was parked in orbit around earth. There has not been made a comprehensive study of this region earlier.

TIMED gives a complete picture of the MLTI region's basic pressure, temperature, wind structure and the spatial and temporal variations that result from the energy transfer into and out of this region.

2.2.2 Instruments

To be able to make a clear picture of the region's dynamics the satellite is equipped with four different instruments. The one that measures the EUV flux from the sun is used in this study.

SEE,*Solar Extreme ultraviolet Experiment*, measures the solar ultraviolet irradiance. The instrument is comprised of a spectrometer and a suite of photometers. The solar EUV flux is stored in 1 nm wavelength bins in the range 0.1 nm to 194 nm. It has a duty cycle of 3%, i.e. SEE measures 3 minutes of the almost 100 minutes orbit.

2.2.3 Data Sources

The data from the TIMED SEE mission have been aquired from NASA's Coordinated Data Analysis Web (CDAWeb). The datasets are available from 2003, thus our study starts with 2003.

Chapter 3

Measurements

To map the relation between n, V_{sc} and the EUV flux, the quantities of these parameters have to be known. All the measurents used in this report will be presented and explained in this chapter.

3.1 EUV Radiation from the Sun

From the ground on earth it is not possible to measure the EUV radiation from the sun with any greater accurracy due to the ionosphere. The ionosphere protects the earth surface from dangerous radiation at these wavelengths. Of course it also shields ground-based instruments from measuring this kind of radiation. To be able to directely measure EUV radiation we have to reach above the ionosphere.

In the absence of a direct measurement of the EUV radiation from the sun, it is common to use some proxy. The $F_{10.7}$ index is one wich has been measured since 1948 and is thus of good use for research that spans over greater time periods, particulary for research looking at times before the satellite era. Another proxy is the $E_{10.7}$ index. It a better approximation of the EUV flux than $F_{10.7}$ according to *Tobiska* [15] and is used in the SOLAR2000 model he is developing. Since none of these EUV proxies are going to be used they are only shortly mentioned.

TIMED is a mission resulting from NASA's ambition to measure the EUV flux from the sun dirrectly and thus these approximation indicies may be unnecessary for the years that TIMED has been in function.

3.1.1 $F_{10.7}$

$F_{10.7}$ is an index for the energy generated by the sun at a wavelength of 10.7 cm. This index has been used as a surrogate for the solar output in wavelengths that produce photoionization in the earth's ionospere (in the ultraviolet bands). This index is a meassurement of the integrated emission at 10.7 cm wavelength from all sources on the solar disc. These sources are almost completely thermal in their origin and are directly related to the amount of plasma trapped in the magnetic fields overlaying the active regions. This is in turn related to the amount of magnetic flux. A comparison made over several solar activity cycles show that there is indeed a linear correlation between the 10.7 cm EUV proxy and the total photosperic magnetic flux in active regions, see [4, 18, 14]. However correlation is only good over a 81 days mean. $F_{10.7}$ has the unit of [W m^{-2} Hz^{-1}] as the measurement is made for a specific wavelength, i.e. 10.7 cm which gives one frequency.

$F_{10.7}$ is produced from measurements by the *Dominion Radio Astrophysical Observatory* in Canada. The measurements are made three times every 24 hours, usually at 17.00, 21.00 and 23.00 UT (universal time).

3.1.2 $E_{10.7}$

A later introduced EUV proxy is the $E_{10.7}$ index, wich has the same unit as the $F_{10.7}$ index. It doesn't fluctuate as much as the $F_{10.7}$ index and gives a better representation of the EUV variation [16] [15]. However, $E_{10.7}$ is derived in a complicated and non-transparent way from a solar EUV model (SOLAR2000) and is less available. We have not used $E_{10.7}$ in this work.

3.1.3 EUV Measurement from TIMED

TIMED has a circular orbit 625 km above the surface of the earth. This is high up in the ionosphere, where the ionosphere is of little concequence for UV measurements due to the low plasma density integrated in a column above the spacecraft. In Figure 3.1 the total solar EUV intensity is shown as a function of alltidude above earth. It shows that the altitude of the TIMED satellite is sufficient. The EUV measurements made by TIMED should therefore not have much interference from the ionosphere. The instrument that measure the EUV flux on TIMED is named SEE. It measures the intensity of the wavelengths in steps of 1 nm for the wavelengths 0.1-194 nm with 0.5 nm centers. For the 1 nm bins that are below 27 nm this is not entirely true as the instrument that measure that (0.1 - 27 nm) interval has an resolution of 7 nm. Yet the data from those wavelengths are presented in 1 nm bins. The principal data product available is a 3 minutes mean of the intensity integrated over 1 nm for every bin. The flux is measured once every 97th minute. The intensity is given in the unit [Wcm^{-2}].

In the used TIMED data, called level L3A, the solar flares are not removed. It has been manipulated so that it has the value of the EUV intensity at a distance of 1 AU from the sun. To make it the EUV radiation at Earth's orbit, all the measurements of TIMED have been divided with the square of the earth's distance (in AU) from the sun at the time of the measurement.

The definition $E(t)$, (see Equation (3.1)), has been used to examine the measurements from TIMED. $E(t)$ has been acquired by adding the 105 first 1 nm bins.

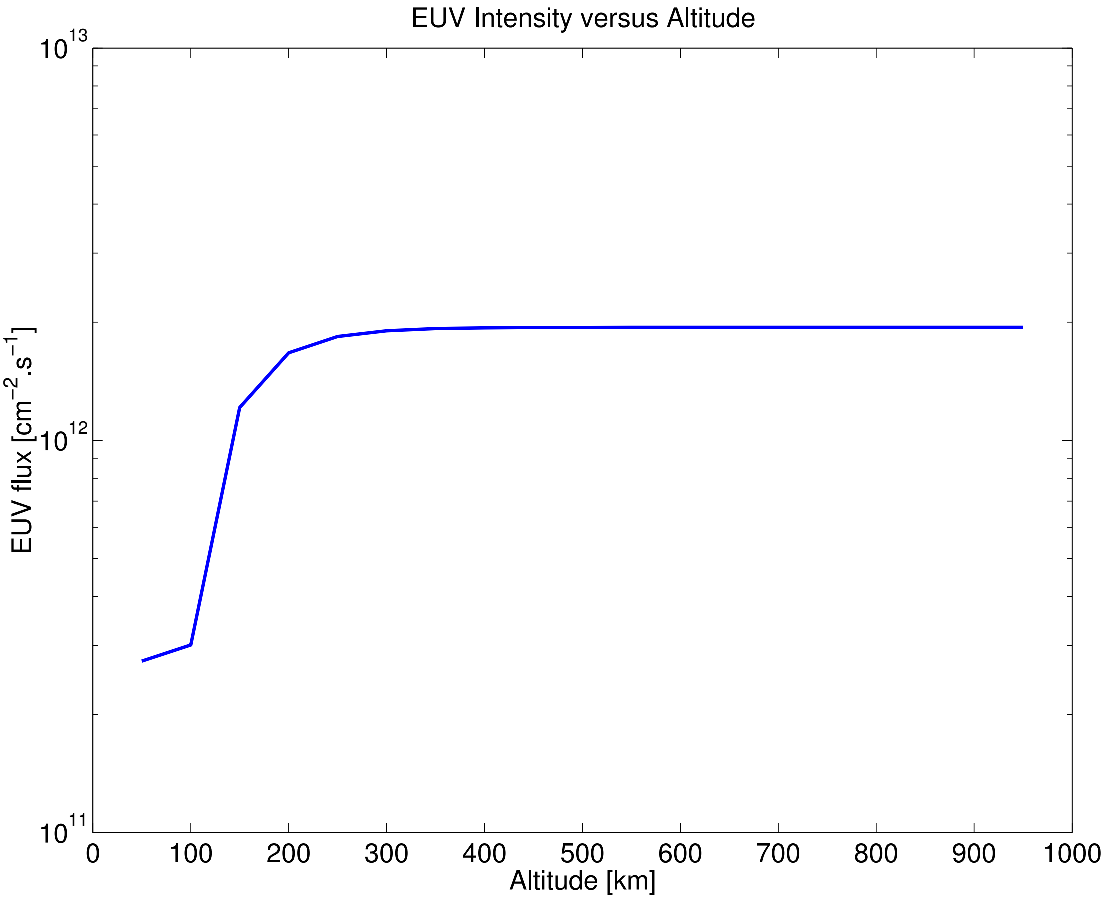


Figure 3.1: The intensity of the solar EUV flux as a function of the altitude above the Earth. This figure is with courtesy of *Ronan Modolo*, who has used a model for aeronomical calculations made by *Richards et al* [12]

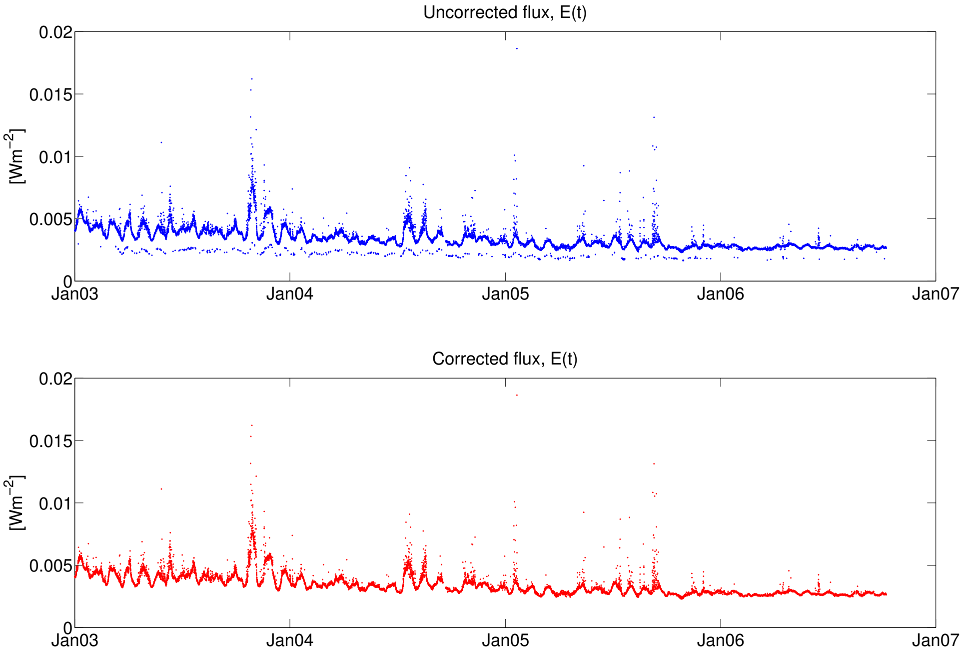


Figure 3.2: The uncorrected $E(t)$ in the top figure showing all the available points and the cleansed $E(t)$ is shown in the lower figure. This is otherwise raw data from TIMED e.i. no correction regarding distance from the sun nor any three days mean formation has been made.

$$E(t) = \sum_{\lambda=1}^{105} (flux(t, \lambda)) \tag{3.1}$$

Some points in Figure 3.2 positioned clearly below the $E(t)$ main curve. These ghost point are assumed wrong and thus have been discarded.

Every measurement point have been replaced by the mean value of all the data from 1.5 days before to 1.5 days after the measurement. This was done to make the data samples a little less noisy. Figure 3.3 shows the flux when all correction have been made.

3.2 Cluster EFW Measurements

The Cluster II satellites carry several instruments. The instruments used in this report are the EFW and the CIS instruments, see the Section 2.1.2. Two of the EFW probes have failed during the mission, so we have used the data from the remaining 14 probes.

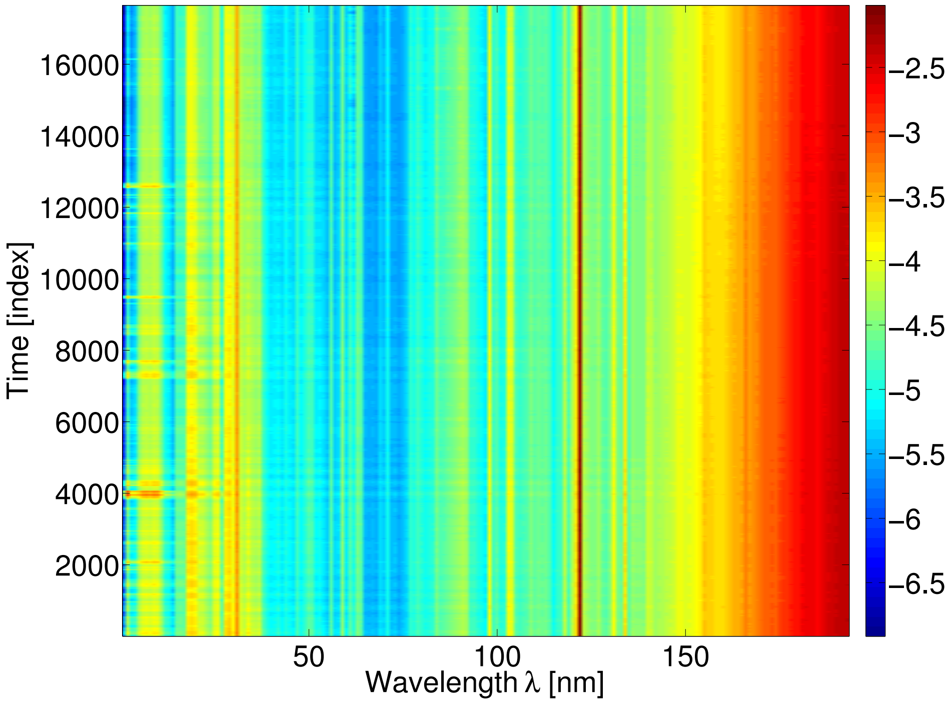


Figure 3.3: The flux intensities from TIMED after the three day mean formation.

3.2.1 The Bias Voltage Sweeps

To determine the photoemission current, we use the Cluster EFW probe bias sweeps. In this mode the *bias voltage* to the probe is varied and the total current is measured. This results in a current-voltage curve, see figure 3.4. Such sweeps are made every few hours. However, EFW cannot directly measure the probe current: instead, it measure the combination

$$V = V_B + RI$$

from wich I can be derived, knowing that R is an internal resistor of 5MΩ. If we use the digital units relevant for the ADCs (analouge to digital converters) doing the measurement, and the DACS (digital to analouge converters) setting the bias as units of V and V_B, this relation becomes

$$\beta V = \alpha V_B + RI$$

where α and β are the calibration factors of the DACs and the ADCs, respecivly. This relation is rewritten as

$$I = \frac{\beta V - \alpha V_B}{R} \tag{3.2}$$

where αV_B is the bias voltage that is varied from -36V up to 30V and then back down to -36V. βV is the measured quantity and I is the total current through the resistor R which has a known value, 5MΩ. It is obviously important to have as acurate calibration factors α and β as possible; a small error in these could propagate to a large error in I. We fine-tune these in Section 4.1.

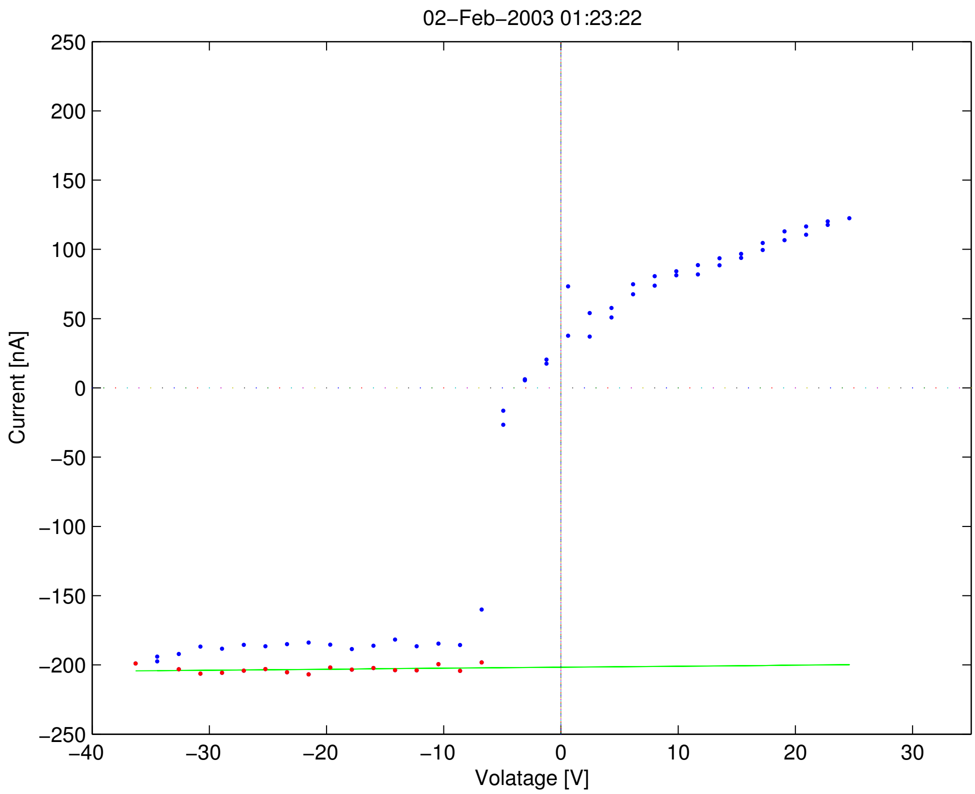


Figure 3.4: The points show one entire sweep. The red points are the points used for the least square fit to the straight line: the green line.

The Cluster satellites makes a bias voltage sweep every four hours and it takes about four seconds to do it.

3.2.2 The Satellite Potential

The satellite potential is the potential between the satellite and its surrounding plasma. It is a determined by the currents of ions and electrons hitting and escaping the spacecraft. These currents give a net charge and therefore result in a potential between the spacecraft and the ambient plasma, as is further explained by e.g. *Engwall* [5] and *Pedersen* [10].

In a quasi-neutral plasma the number density of electrons is equal to the number density of positively charged ions. Although the numbers are the same, more electrons than ions hit the spacecraft, the speed of the electrons are much higher than the speed of the ions due to the lower mass of the electrons. This will result in a negative charge contribution from the incident particles.

If the spacecraft is sunlit, electrons are emitted from its surface due to the photoelectric effect. This will result in a positive charge contribution. If the spacecraft has a positive potential this will force most of the electrons to return to the spacecraft. Only the electrons given sufficiently high energies from the EUV flux will manage to escape into the ambient plasma. The less dense plasma, the higher potential the spacecraft must attain in order not to emit more photoelectrons to infinity then it can attract plasma electrons. Hence, the spacecraft potential can be used as a proxy for the plasma density, as will be used in Chapter 6.

In equilibrium the currents will balance each other, i.e. the sum of the currents will be zero. The flows of the currents on the probe and the satellite can be seen in Figure 3.5.

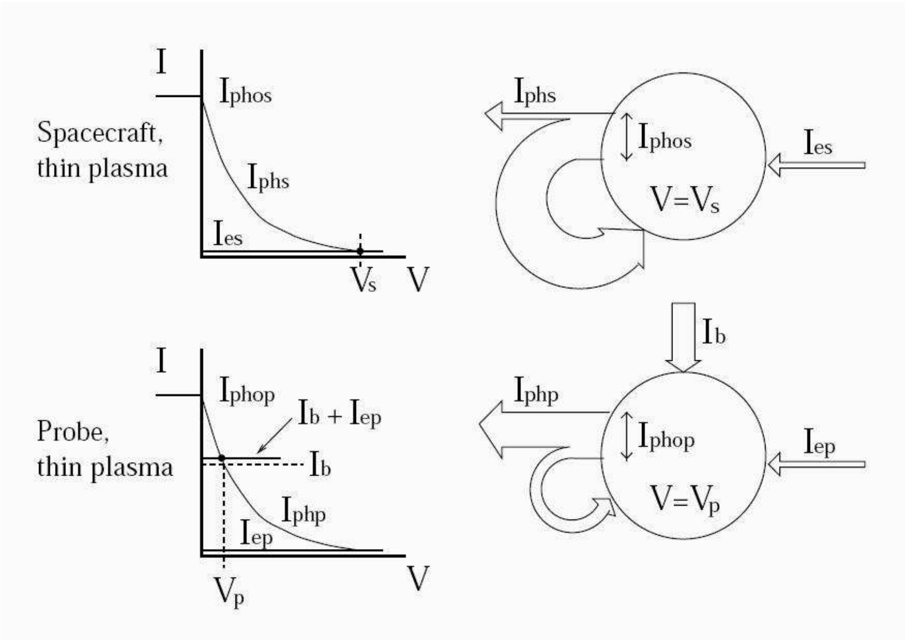


Figure 3.5: The currents on the satellite and the probe. I_{phos} and I_{phop} are the photoelectron saturation currents. I_{phs} and I_{php} are the currents of escaping photoelectrons. I_{es} and I_{ep} are the incident electrons from the ambient plasma. I_b is the bias current. (Adapted from *Pedersen et al.* [10])

The standard mode for EFW is to apply a bias current to each probe and measure the voltage between probe and spacecraft, U. This is not equal to the satellite potential, V_s, as there is a potential difference between the ambient plasma and the probes, (V_1 and

V_2 in Figure 3.6). From these potentials between the probes and the spacecraft, U_1 and U_2, the spacecraft potential, V_s may be determined.

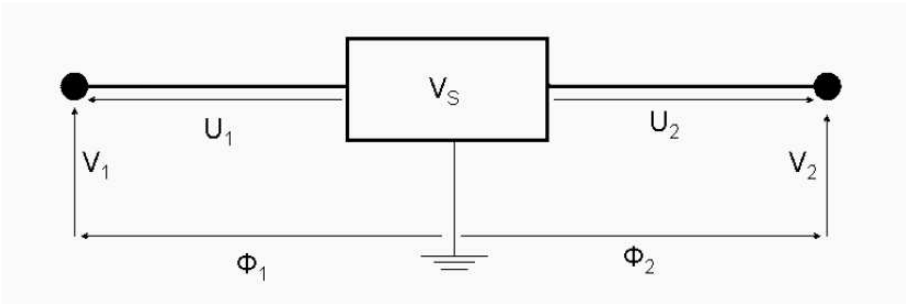


Figure 3.6: A shematic picture of the voltages around the satellite. U_1 and U_2 are the measured voltages. Φ_1 and Φ_2 are the voltages in the plasma. (Adapted from *Engwall* [5])

We use U for two purposes in this report. In Chapter 6, it is used as a density proxy. In Section 3.2.3, we use it for evaluation of the probe bias sweeps. In this case, the probe-satellite potential is an average over the 8 (2 spins) seconds directly before and the 8 seconds directly after every bias voltage sweep which takes 4 seconds to complete. Then the probe-satellite potentials measured during these 16 seconds are used to make an average of the potential, V_{psm}. V_{psm} is then used as the probe-satellite potential during the actual sweep. The probe-satellite potentials give an approximation of the photoelectron current, see Section 3.2.3. The probe-satellite potential is also used to make an estimation of the satellite potential which is used in Chapter 6 for plasma density calculations.

3.2.3 The Photoelectron Saturation Current

For a sunlit satellite at zero potential in a magnetospheric plasma the photoelectron current will be dominating. This gives the satellite a positive potential, as explained in Section 3.2.2. If the probe is brought to a sufficently high negative potential all the photoelectrons will escape from the probe. This is the photoelectron saturation current. In the *bias voltage sweep*, see Figure 3.4, the photoelectron saturation current can be measured where the current is constant regardless of how much the bias voltage is negatively increased, see the red points in the same figure.

The photoelectron saturation current is used in Chapter 5 to derive an photoelectron yield of the Cluster probes. If EUV measurements are lacking this current may be used as a proxy of the solar EUV flux [3].

To aquire the photoelectron saturation current we use the bias voltage sweeps (Section 3.2.1). Several steps are needed to analyse the bias voltage sweeps. There are 101 676 analysed bias voltage sweeps. For each step some of these sweeps are excluded.

1. First we had to check if V_{psm} is above -30V. If it is the case the sweep will be used since it has enough points to do a reliable least square approximation to the

straight line (the green line in Figure 3.4).

2. Then the points with current values above -100nA and $\alpha V_B > V_{psm} - 2$V are removed.

3. Only sweeps that have more than five points left after 1 and 2 are used.

4. Then a linear least square fit is made of the points that are left in the voltage sweep.

5. The points that are too far (8 nA) away from the fitted line are then removed.

6. Another control is made of how many points remain in the sweep. At the least five points are required.

7. A new linear least square fit is made if the number of points are enough.

8. The straight lines approximating the points, the red points in figure 3.4, which have slopes within the range of ± 2 nA/V are used to approximate the photosaturation currents. Any larger slopes are discarded as unphysical, as the probe current ought to be flat in the photoelectron saturation region.

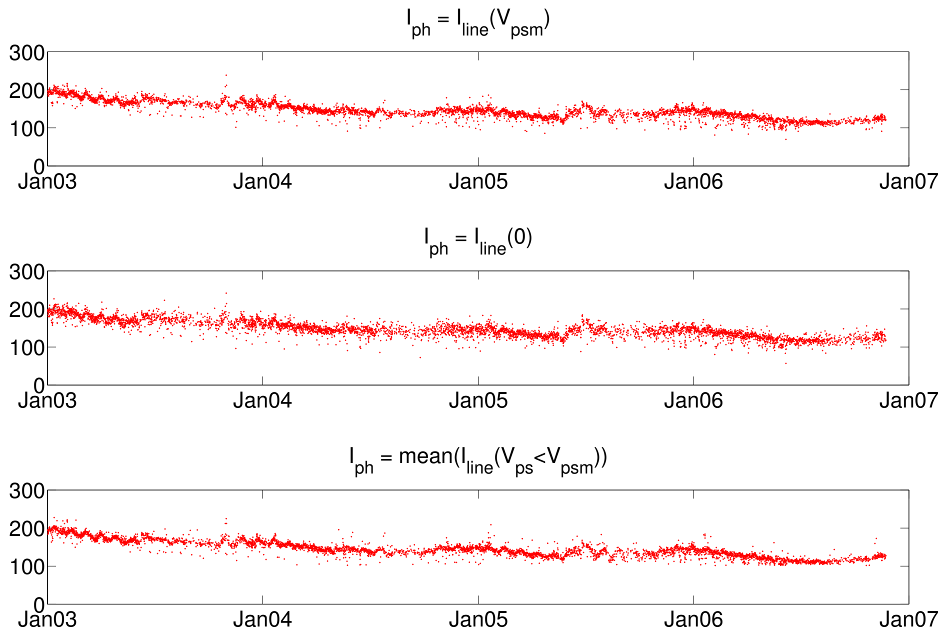


Figure 3.7: The three different approches to approximate the electron photo saturation current I_{ph}, using probe 3 on satellite 2

The photosaturation current should be any value anywhere on the straight line approximating the red points. As the line is never perfectly horizontal but has a slope different from zero this is not true. Three different methods for obtaining the photoelectron saturation currents have been used.

1. Take the value of the green line, I_{line}, in Figure 3.4 in the point where αV_B is equal to the probe-satellite potential V_{psm}, i.e $I_{ph} = I_{line}(V_{psm})$.
2. Take the mean of all the points used to make the green line, i.e. all the red points.
3. The value of the green line at $V_B = 0$ is seen as the photoelectron saturation current.

These approximations differs and thus the one that is the best has to be determined. The three different photoelectron saturation currents have been plotted in Figure 3.7. The top photoelectron saturation current approximation is tantamount to the bottom photoelectron saturation current. The middle approximation is noisier than the other two and is thus discarded. It was thus decided to use one of the two equivalent approximations and $I_{ph} = I_{line}(V_{psm})$ was chosen.

We have analysed all EFW probe bias sweeps between January 2003 and December 2006.

Chapter 4

Calibrating EFW Bias Sweps

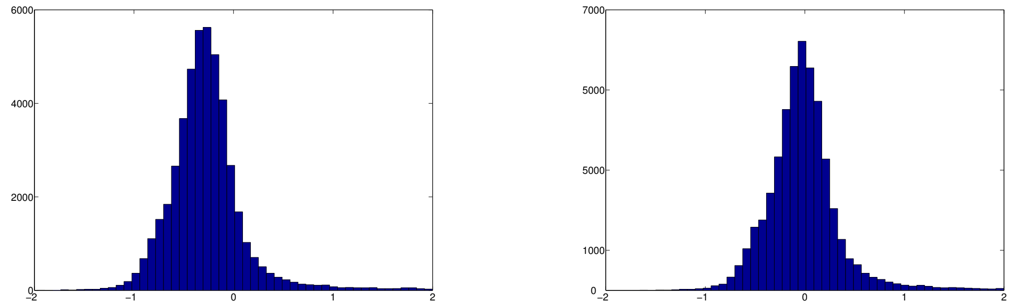
As remarked in Section 3.2.1, our analyses of the photoemission saturation current is sensitive to calibration errors. In this Chapter, we derive improved values for the calibration coefficients and control if they show any sign of change in time. Any electrical, or other, component will deteriorate sooner or later if they are left without maintenance. Deterioration can be of several kinds, we will here mainly look for signs of continous degradation. The Cluster II satellites have been in position above the earth for about six years. Are there any signs of aging components in the EFW data? Before any further analyses of the Cluster data are made this would be very useful to know. Our approach is to reveal any drift of the calibration factors α and β for the measured voltage and the bias voltage, connected by the equation

$$\beta V = \alpha V_B + IR \tag{4.1}$$

This equation is explained in chapter 3 regarding the sweep data recieved from the Cluster II satellites.

4.1 Fine-tuning of α

In the photosaturation current, the slope of the bias voltage sweep should be zero (left side of the Figure 3.4). This occur very rarely. The recieved data sweeps have a slope in this region that differs a little from zero due to varying sunlight exposure of the thin wire connecting the probe to the puck, noise in the plasma and the electronics, and the fact that the program doing the analysies of the sweeps is not perfect. However, we assume is that the distribution of all the magnitudes of all the slopes should be centered around zero. Using nominal (from ground tests) values of the calibration factors α and β, this distribution is, as can be seen in Figure 4.1(a), not centered around zero. The distance between the mean value and zero (which should be in the the vicinity of the peak) is due to some error. We cannot determine if this error is in α or β, but choose to put it all in α, assuming a constant β identical for all probes. We can now centre the



(a) With original α

(b) With calculated α_0

Figure 4.1: The two figures show the distribution of the magnitude of the slopes (including sign) for original α and the corrected α. These are for all the probes and all years.

distribution for each probe by introducing suitable values for α. To look for component detorioration, we have to ask the question: will these values change in time? With the error in α, α may be written as

$$\alpha = \varepsilon\alpha_0$$

where ε is the factor containing the error and α_0 is the correct value of the calibration coefficient for each indivdual probe. Wanted is the value of α_0 and α is the nominal calibrating coefficient. This gives

$$\alpha_0 = \frac{\alpha}{\varepsilon} \tag{4.2}$$

From equation 4.1 the estimated photosaturation current is written as

$$I_{est} = \frac{\beta V - \varepsilon\alpha_0 V_B}{R}$$

We want to eliminate β, and thus rewrite as

$$I_{est} = \frac{\beta V - [(\varepsilon - 1) + 1]\alpha_0 V_B}{R} \tag{4.3}$$

As the true photosaturation current is given by

$$I = \frac{\beta V - \alpha_0 V}{R}$$

the Equation (4.3) becomes

$$I_{est} = I - \frac{(\varepsilon - 1)\alpha_0}{R} V_B \tag{4.4}$$

The slope is said to be zero if the error ε is equal to zero. The true photosaturation current, I, is a constant (the slope is zero there). When Equation (4.4) is differentiated with respect to αV it becomes

$$\frac{dI_{est}}{d(\alpha V)} = -\frac{(\varepsilon - 1)}{R}\frac{\alpha_0}{\alpha} = k$$

where k is the observed slope in the saturation region. Solve for ε:

$$k = \frac{1-\varepsilon}{R}\frac{\alpha_0}{\alpha}$$

$$\frac{Rk\alpha}{\alpha_0} = 1 - \varepsilon$$

$$\varepsilon = 1 - \frac{Rk\alpha}{\alpha_0} \tag{4.5}$$

Use equation (4.2) in (4.5)

$$\varepsilon = 1 - \frac{Rk\varepsilon\alpha_0}{\alpha_0} = 1 - Rk\varepsilon$$

$$\varepsilon = \frac{1}{1+Rk} \tag{4.6}$$

then finally use (4.6) in (4.2) and the result is:

$$\alpha_0 = \alpha(1 + Rk) \tag{4.7}$$

When the corrected value α_0 for each individual probe is used, the slopes center around zero (as in Figure 4.1(b)). The usefulness of equation 4.7 depends on the number of included slopes in the mean. If they are to few the method described may increase the error in α rather than decrease it. The Figures 4.1(a) and 4.1(b) show the improved performance of α_0 in contrast to α. In these Figures data from *all* the satellites and *all* the probes were used. Some probes have irregular distributions of their slopes. Probe 4 on satellite 2 is the only one which does not have a nice distribution and therefore this calculation of α amounts to nothing. From this probe few values are used as well. As can be seen in Figure 4.2 the slope distributions from probe 4 on satellite 2, with and without correction, are not useful distributions. Therefore, we suspect some other errors on this probe.

The error is calculated for each individual probe. First recieve the distribution using the nominal value of α and then the distribution using the corrected α_0.

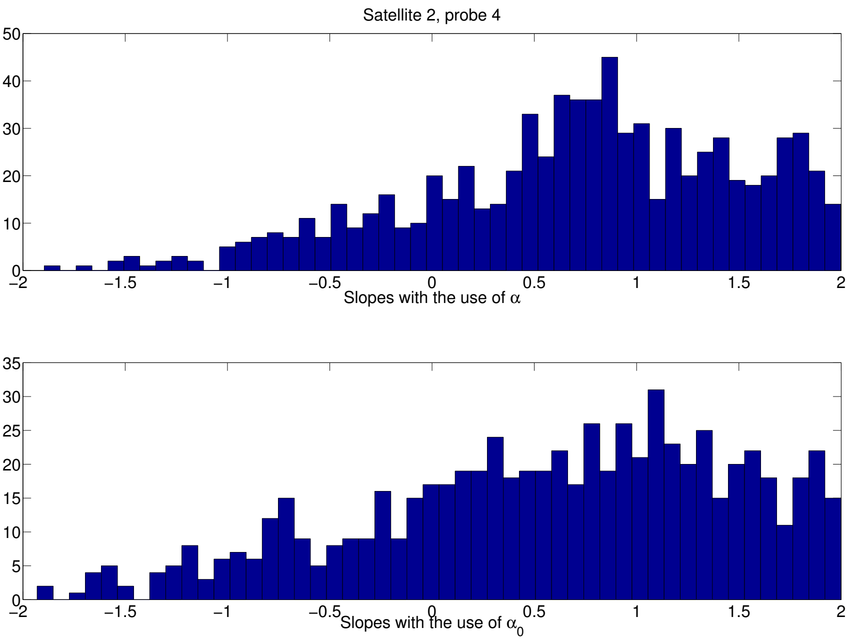


Figure 4.2: The distribution of slopes from satellite 2 probe 4.

4.2 Results

The improvment after using the corrected α_0 is shown in in figure 4.1(b). Equation 4.7 will give the α_0 for each individual probe. Table 4.1 shows all the different α_0.

	SATELLITE			
	1	2	3	4
1	-	0.30714	-	0.30665
2	0.30676	0.30748	0.30742	0.30743
3	0.30719	0.30723	0.30704	0.30717
4	0.30726	0.30879	0.30737	0.30752

Table 4.1: α_0 for each individual probe. Data taken for each probe during all the years 2003 through 2006

From this the difference in the α_0 in regard to α, in percent, can be calculated in the following way:

$$(\varepsilon - 1) * 100 = (\frac{\alpha}{\alpha_0} - 1) * 100 = 0.139\% \quad (4.8)$$

This is the error if *all* the satellites and *all* the probes from *all* the years are used. For the error for every individual probe see the Figures 4.3, 4.4, 4.5 and Figure 4.6.

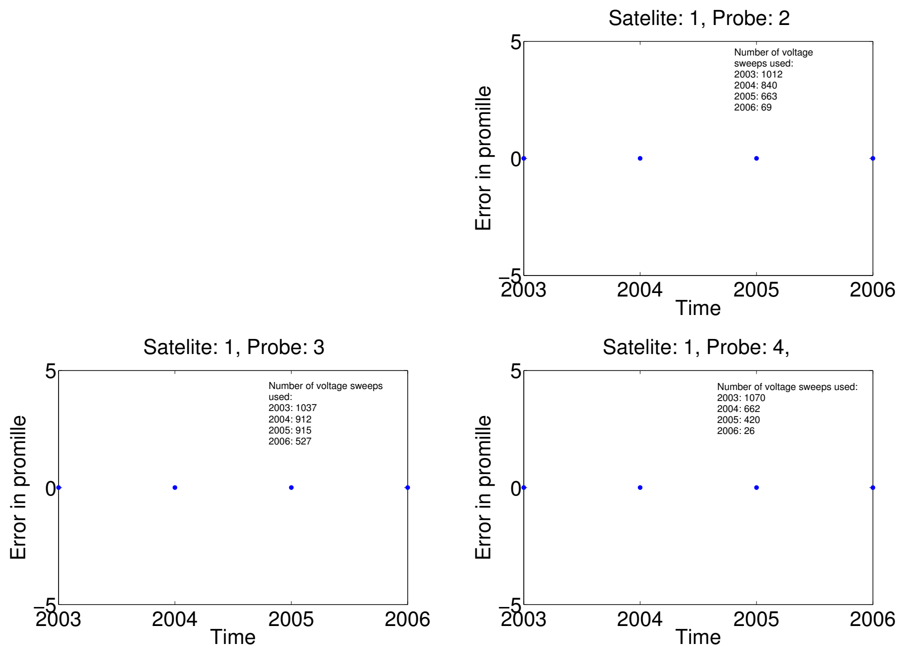


Figure 4.3: The error in α over the four years 2003 to 2006

4.3 Conclusion

Three assumptions have been made in this chapter. The first is that the slope in one sweep, the green line in figure 3.4, should be zero. Secondly is the assumption that all the values of the slopes should be in a distribution with a mean of zero. This means that nothing can be certain with few sweeps. The third assumption is that the error is exclusively in α.

The error in α is said to make itself visible in the distance between the mean and the wanted mean of zero. Here the correction of α has been to move the mean of slopes this distance towards zero. In the case of probe 4 on satellite 2 this was not possible. As seen in Figure 4.3 the distribution looks rather disorderly. This leads us to believe that something is wrong with this probe. All the other probes have distributions resembling (more or less) the one shown in Figures 4.1(a) and 4.1(b).

The error may not be only in α as assumed here, but also parly in β but either way the symptoms of the error has been dealt with.

When the error, regardless of where it comes from, is corrected with such a small change in α, as shown by the Figures 4.3, 4.4, 4.5 and Figure 4.6, the error may safely be neglected. The conclusion is therefore is that the EFW components on the Cluster II satellites are, at the current time, still in perfect condition.

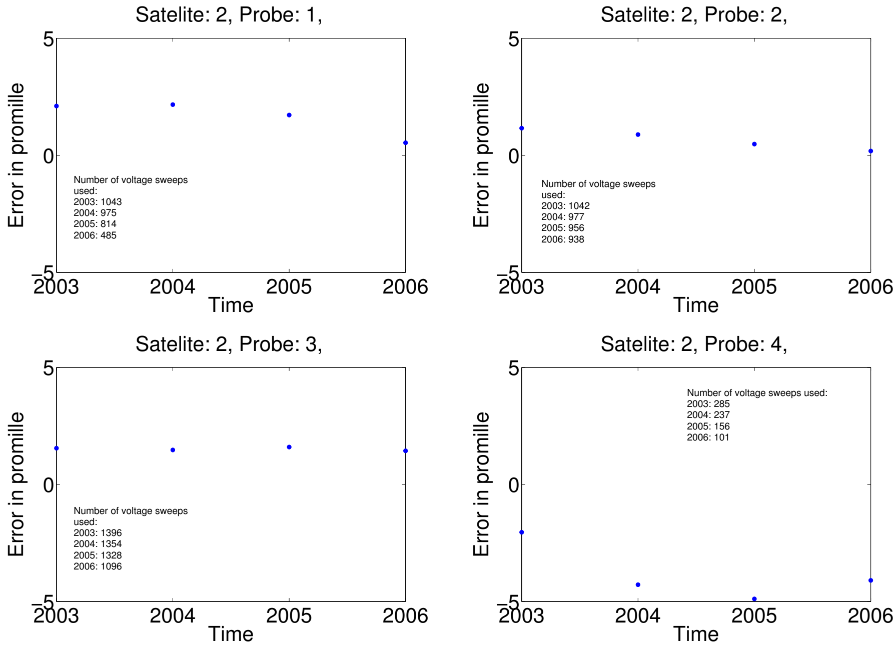


Figure 4.4: The error in α over the four years 2003 to 2006

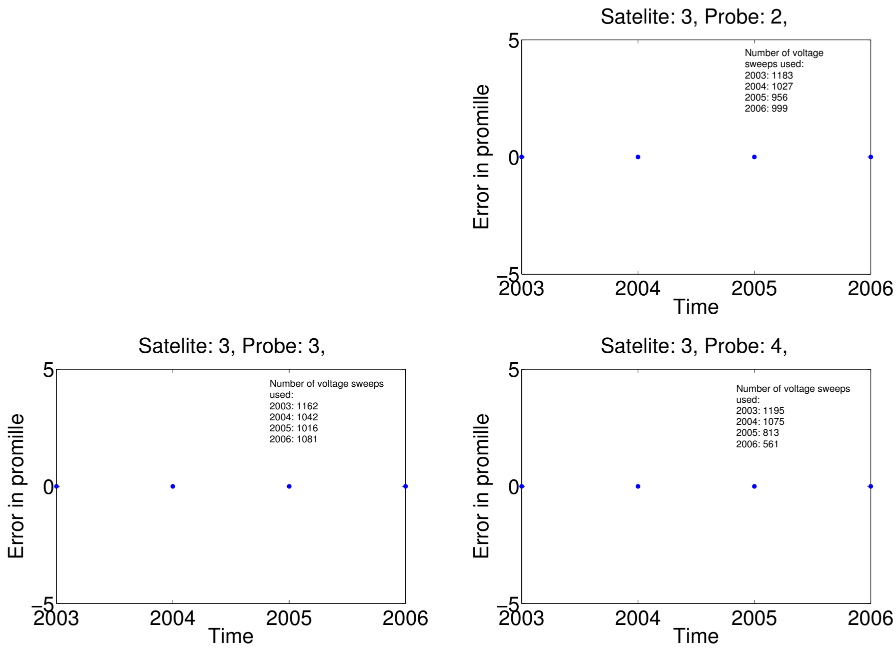


Figure 4.5: The error in α over the four years 2003 to 2006

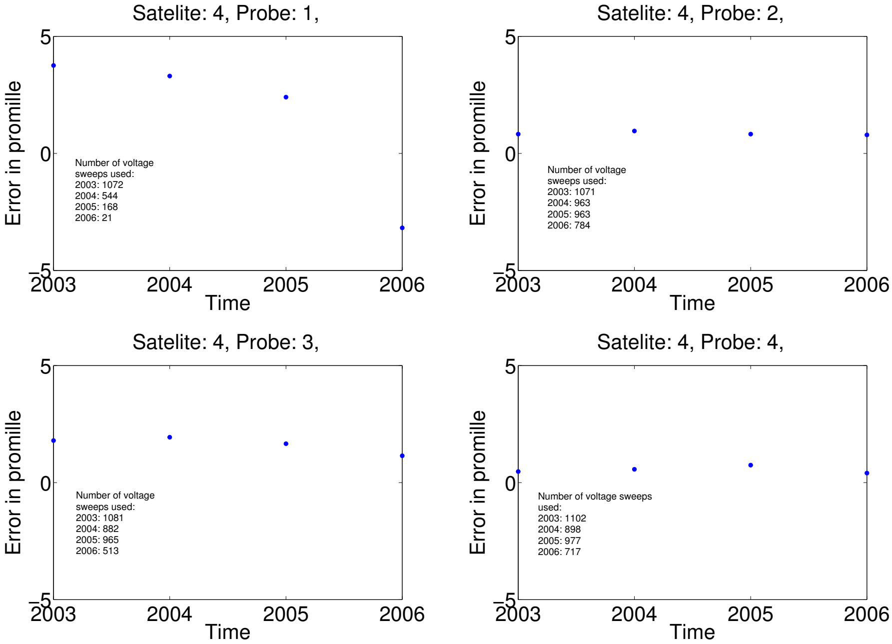


Figure 4.6: The error in α over the four years 2003 to 2006

Chapter 5

Photoelectron Current Yield of the Solar EUV Flux

The purpose is to find the photoelectron current yield from the Solar EUV energy flux as a function of the wavelength. The yield function should give a photoelectron current contribution for the energy flux of every wavelength. The sum over all these contributions give the total photoelectron current, I_{ph}, as in Equation (5.1).

$$I_{ph}(t) = \sum_{\lambda=1}^{193} flux(t,\lambda)A(\lambda) \tag{5.1}$$

Where λ is the wavelength bin (describeed in Section 3.1.3) and $flux$ is the EUV energy flux, [$\mathrm{Wm^{-2}}$], of that same wavelength λ. The A is the searched yield function and it has the unit $\mathrm{nAm^2W^{-1}}$ and t is the time of the measurement of the EUV energy flux.

Different approaches have been tried to find this $A(\lambda)$. *i*) wich wavelengths varies with the photoelectron current both in the short and long term periodic fluctuations and use these wavelengths to construct an yield function, *ii*) is there any yield functions already constructed by others gives a photoelectron current that conforms to the actual measured current. *iii*) solve a least square problem to find an yield function and see if any of the solutions are sufficently good estimations. Despite the clearly unphysical nature of this yield function, the current data are reasonably reproduced. This shows that reconstructing yield functions by deconvolution of probe current and UV flux measurements can result in unphysical results, presumably because of measurement errors. We will have reason to return to this issue in Section 5.3*iv*) we construct continous yield functions of the wavelength. All of these yield functions is compared with the photoelectron currents measured by the Cluster probes.

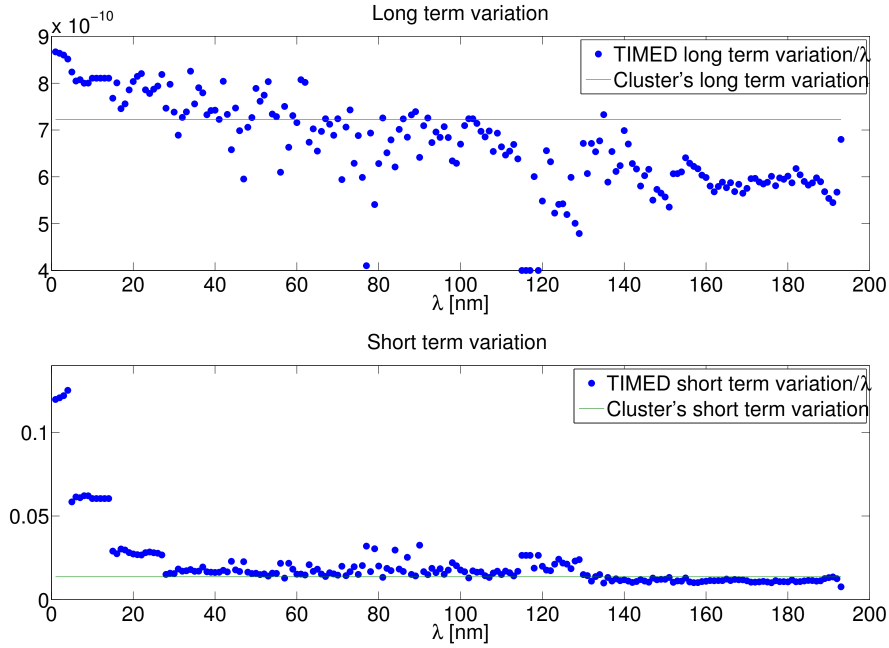


Figure 5.1: The short and long term variation of every wavelength (blue dots) compared with the variations of the photosaturationcurrent(greenline)

5.1 The Long and Short Term Variations in the TIMED Data

To acquire the wavelengths of importance for the Cluster photoelectron current the variation of every wavelength was compared with the variation in the photosaturation current.

These variations are computed in the following way: *i*) the current was approximated with a straight line over the entire time interval 2003 through 2006.*ii*) every measured value of the current was subtracted with its corresponding value on the straight line. *iii*) the standard deviation of this difference was normalised with the constant term of the straight line function that approximated the current. The same was then done to each wavelength of the EUV flux. This is the short term variation.

The long term variation is the time dependent term of the straight line approximations. See these variations compared in Figure 5.1. Few of the wavelengths have both the same long and short term variation. The TIMED data looks very peculiar at wavelengths shorter than 28 nm. The resolution of the TIMED data in this wavelength interval is only 7 nm, see Section 3.1.3.

The wavelengths where the short and long term variations in the flux coincided well with the photoelectron current's were used to build an yield function. It is observed, in

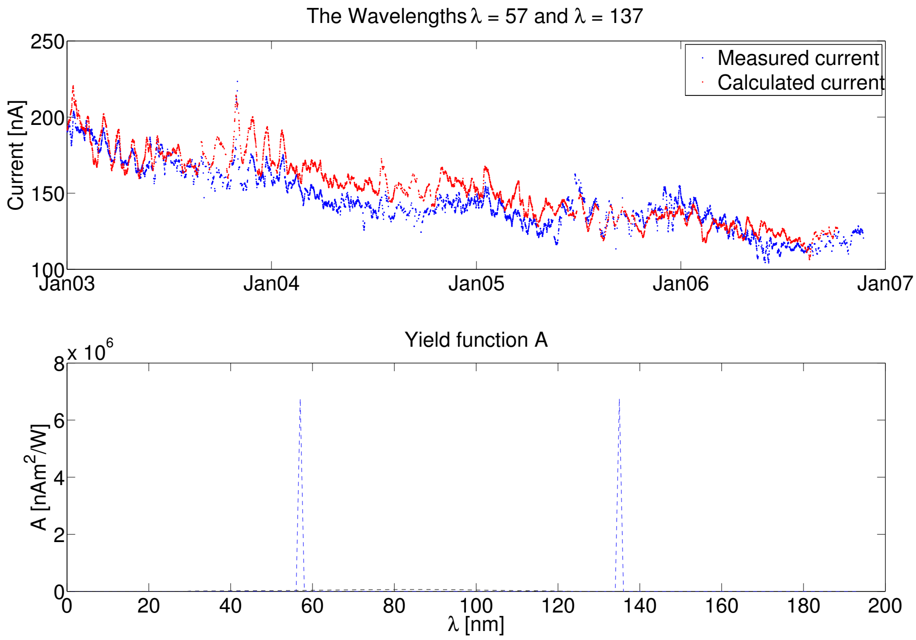


Figure 5.2: The photoelectron current given by an yield function taken notice of only the wavelengths of 57 nm and 135 nm.

Figure 5.1, that these points out a few specific wavelengths. Two wavelengths were used, namely $\lambda = 57$ nm and $\lambda = 135$ nm, both were adjusted to give the same contribution to the photoelectron current. The constructed yield function is then normalized so the first value of the calculated current coincide with the first value of the measured one. The result is shown in Figure 5.2.

5.2 Adapted Yield Functions

An a approach to find the yield function is to examine ones already constructed. Three such functions have been used to estimate a current and then this current is compared with the measured current from the Cluster probes. The three yield functions have been borrowed from *Brace et al* [3], *Samson and Cairns* [13], *Feuerbacher and Fitton* [7] and *Arends* [2].

All three functions have a yield of photoelectron per incident photon. They can thus be written on the form

$$Y = \frac{\Gamma_e}{\Gamma_\nu} \tag{5.2}$$

Γ_e is electrons per square metre and per second and Γ_ν is incident photons per square metre and per second.

The yield should reeturn a current given in nA from the energy flux from the sun. This since measurements from Cluster is given in nA and the flux from the Sun is given in Wm^{-2}. The wanted form of the yield function is thus;

$$A = \frac{I_{ph}}{F} \tag{5.3}$$

Here F is the Solar EUV flux and I_{ph} is the photoelectron current given in ampere. Since the EUV flux is given in energy per square metre it can be written as:

$$F = \Gamma_\nu E_\nu = \Gamma_\nu h\nu = \Gamma_\nu \frac{hc}{\lambda} \tag{5.4}$$

E_ν is the energy of the photon at a certain frequency. The total photoelectron current can be expressed in a similar way:

$$I_{ph} = e\Gamma_e \pi a^2 \tag{5.5}$$

where πa^2 is the projected area of the probe towards the sun. e is the elementary charge. Now A in Equation (5.3) can be written with the use of Equations (5.4) and (5.5) as

$$A = \pi a^2 \frac{e\lambda}{hc} \frac{\Gamma_e}{\Gamma_\nu}$$

rewritten again using the Equation (5.2)

$$A = \pi a^2 \left(\frac{e\lambda}{hc} \right) Y \tag{5.6}$$

Given the yield function Y, in electrons per photon, we can now calculate its corresponding A yield function that gives current per energy flux.

As the yield function is a material characteristic, these adapted function are all slightly different since they are for different materials. The function from Brace is the yield for molybdenum and rhenium. The one from Samson and Cairns is for aluminium and Arends' function is for DAG-113 (wich is the same as Feuerbacher and Fitton's yield function) , a carbonaceous coating material. The yield functions and their calculated currents are shown in the Figures 5.3, 5.4 and 5.5.

5.3 Yield Functions from Cluster/TIMED Comparison

Another way to achive a yield function is to fit one to the measurements given by Cluster and TIMED. The arrangement of the EUV flux measurements from TIMED are presented in Section 3.1.3.

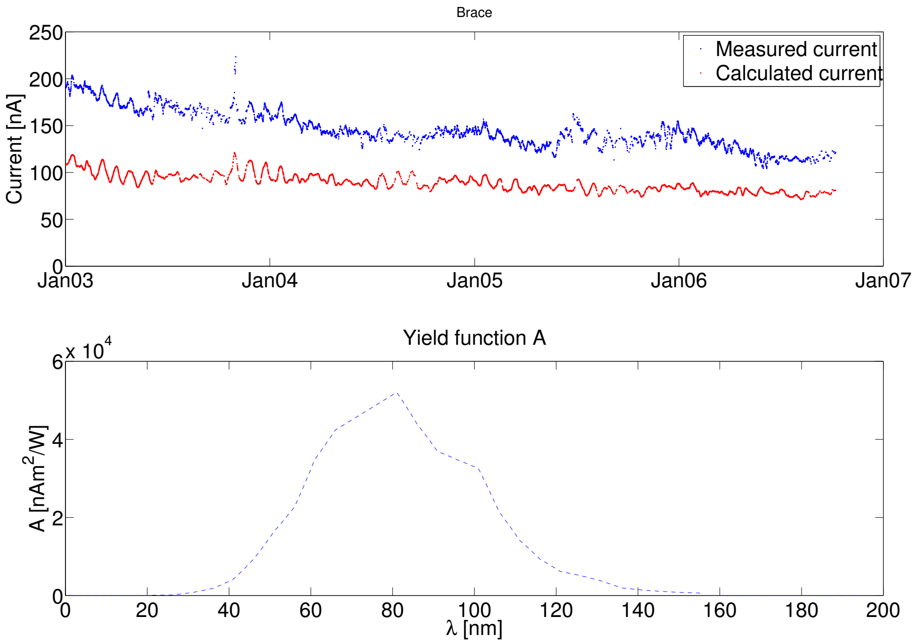


Figure 5.3: The measured current and the calculated current are shown in the top plot. Brace's yield function is for molybdenum and rhenium and is shown in the lower plot.

When both the photoelectron saturation current and the flux is known, a yield function can be calculated from the least square problem:

$$\mathbf{FA} = \mathbf{I}_{ph} \tag{5.7}$$

$\mathbf{F}$ is a $n \times 193$ matrix with the EUV flux measurements. $\mathbf{A}$ is a 193 elements long vector and $\mathbf{I}_{ph}$ is a n elements long vector with the photoelectron saturation currents. The n is the timeindex for the measurements.

As the measurements from Cluster and from TIMED are not equal in number and not are made at exactly the same time, one of them has to be interpolated to fit the other. Here the flux of every wavelength have been fitted to the times of the Cluster measurements, as there are more measurements from TIMED than from Cluster, see Section 3.2.3.

Each element in $\mathbf{A}$ represents the current yield for one wavelength, λ. All the values in $\mathbf{A}$ have to be positive or zero. Negative values of the current yield per photon are physically impossible. Thus the problem can't be solved by left division in MATLAB, when this gives negative solutions as well as positive. The solution has to be restricted to only posite values. Then the problem can be written as

$$Min\|\mathbf{FA} - \mathbf{I}_{ph}\| \qquad \text{subject to } \mathbf{A} \geqslant 0 \tag{5.8}$$

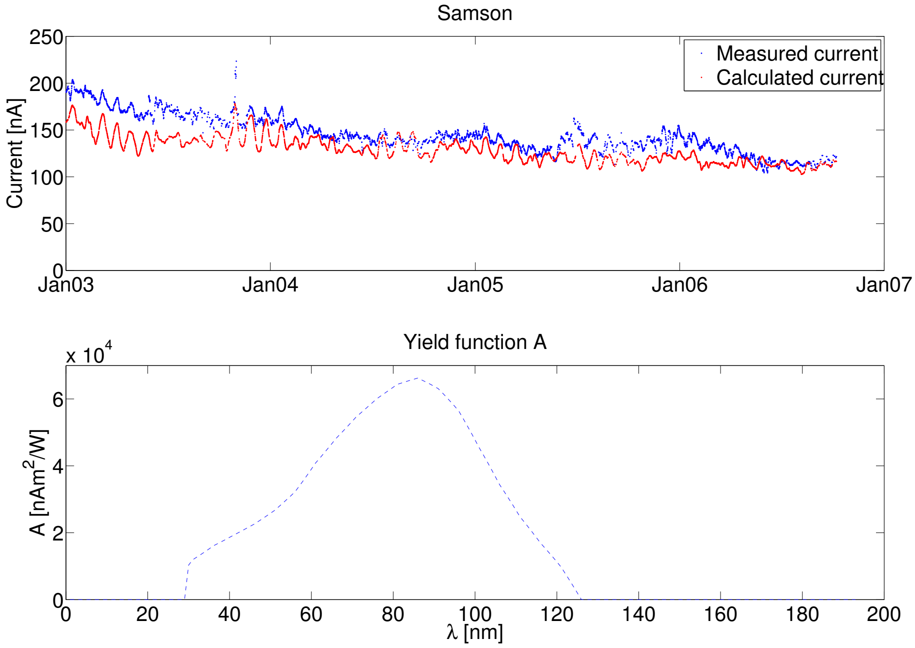


Figure 5.4: The measured current and the calculated current using Samson's yield function, are shown in the top plot. The yield function from Samson is for aluminium and shown in the lower plot

To make this minimization of this linear least square problem a MATLAB rutine named LSQNONNEG was used, based on an algorithm described by *Lawson and Hanson* [9]. This function put all the effort in a few of the wavelengths. The rest are set to zero. The non negative linear least square problem has always a solution, but it is only unique if the rank of **F** is larger than the length of **A**. All solutions calculated from a **F** with rank less than 193 are therefore discarded.

Some different ways to estimate an yield function **A** for the Cluster probes will be presented hereafter. All the estimated photoelectron currents calculated from these yield functions will be compared with the measured current from Cluster satellite 2, probe 3, this probe's measured current covers well the entire time interval 2003 to 2006.

In a first try to find a good solution, all the currents measured from all the operating probes (see Section 3.2) over all the four years are used. Then a three day mean value formation was made of the current. That means that every measured point is replaced by the mean of the values from 1.5 days before to 1.5 days after the measured point. The solution is shown in Figure 5.6.

Then we used only currents and EUV flux measurements from the beginning of the year 2003 (january through mars). From those currents and EUV flux measurements a yield function, **A**, was calculated. The yield function was then applied for the entire time interval 2003 to 2006. This solution is presented in Figure 5.7

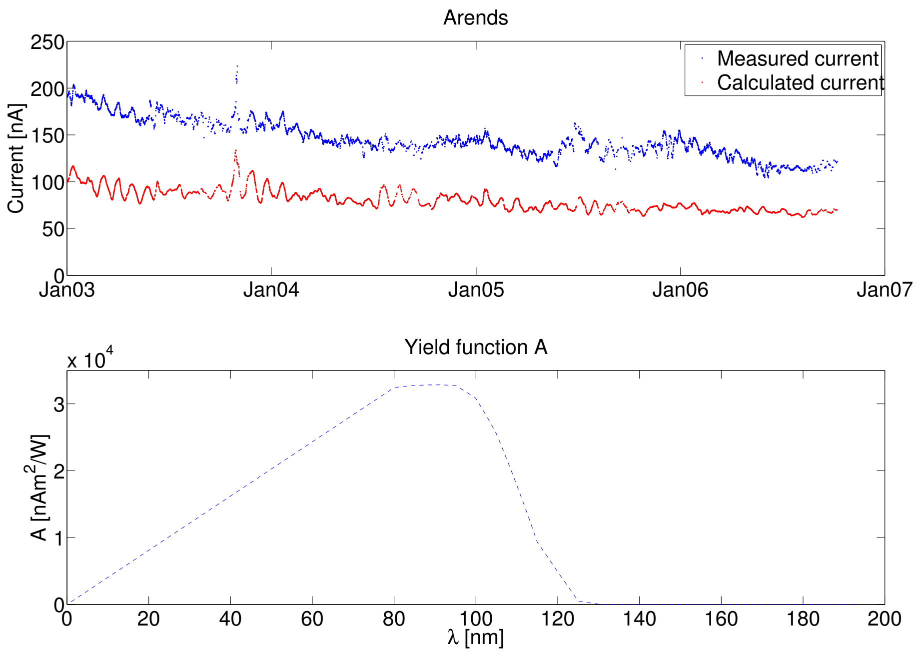
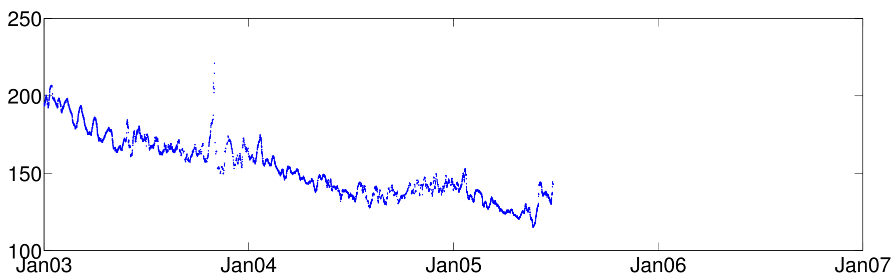


Figure 5.5: The yield function from Arends work, considering the material DAG113, a carbon-water paint.



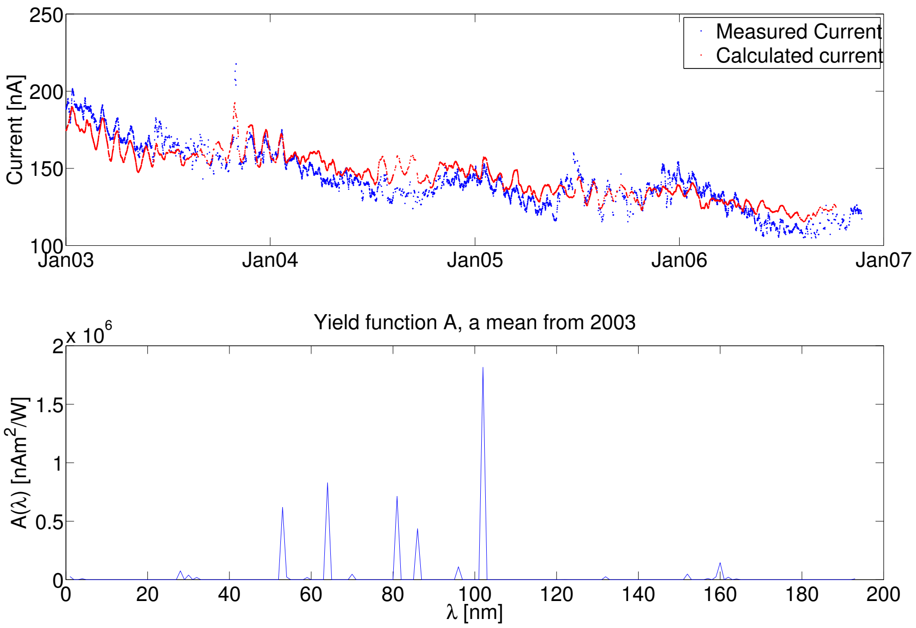


Figure 5.7: The yield function given by the measurements from the beginning of the year 2003. It is then used to calculate the photoelectron currents of all the years

The same was done again but with the measurements from the beginning of the year 2006 were used instead of the measurements from the beginning of the year 2003. The result is shown in Figure 5.8 result is shown. The reason for discarding the second half of the year is that Cluster then spends a large fraction of its time in the tenous magnetotail lobes, where V_{sc} is high and only a few data points remain in each sweep.

Then every year was divided in three parts. First part was january to mars, the second april to june and the last july to december. An yield function **A** was then calculated for every probe and period, i.e. 168 calculated yield functions. All the yield functions were than added together and divided by the number of added functions. This gives a mean of the yield for every wavelength (Figure 5.9).

The yield function from previous method gave a lot of yield from the wavelength 1 nm. That wavvelengt shouldn't give so much yield compared to the adapted yield functions from Arends and the others in Section 5.2. The next try is to set all the yield from wavelengths shorter than 27 nm to zero in the previous yield function. The value of 27 nm comes as a result of the fact that the TIMED data has an error of 20% in these wavelengths see *Woods et al* [17]. The Figure 5.10 shows the calculated currents from this slightly modified yield function.

Finally we tried to take measurements from the beginning of each of the four years and calculate a yield function from those values. The function was then applied to the flux

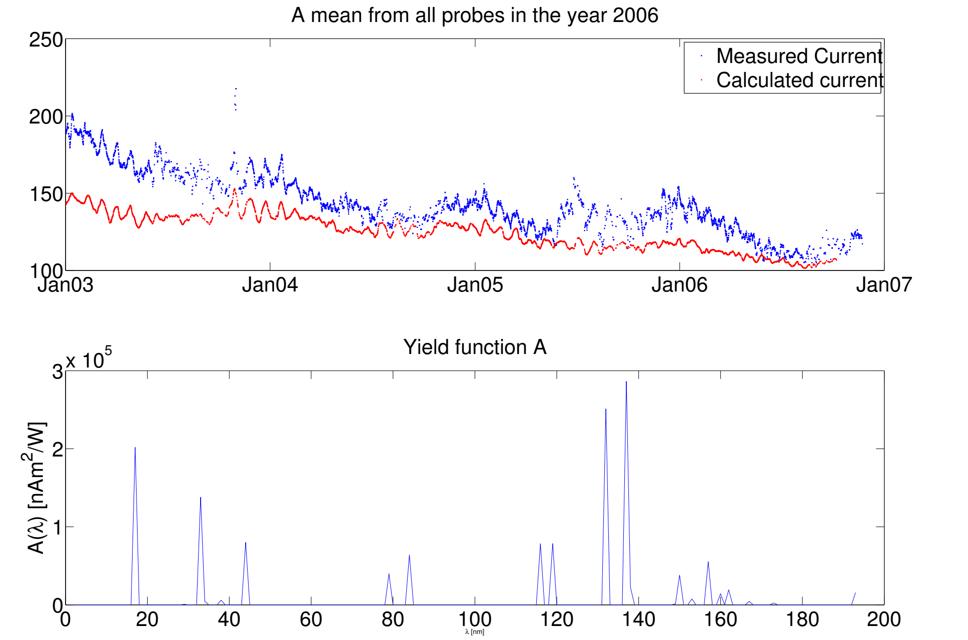


Figure 5.8: The yield function is calculated from the measurements from the beginning of the year 2006. This yield function is the used to calculate a photoelectron current on the probes

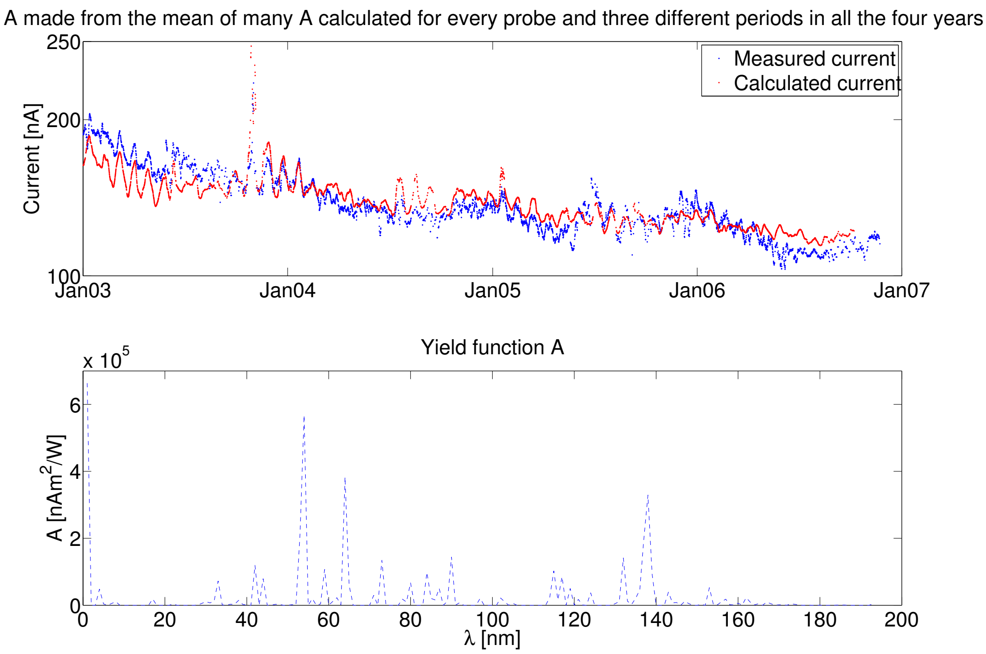


Figure 5.9: The yield function as a mean of many yield functions from many time intervals

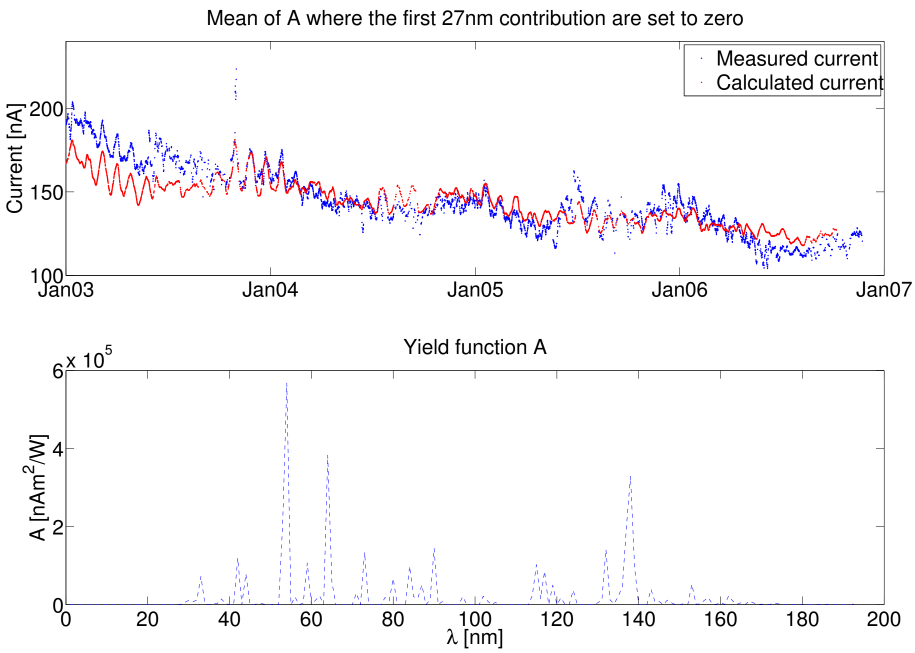


Figure 5.10: The yield function as a mean of many yield functions calculated from many time interval. The yield from wavelength shorter than 27 nm has been set to zero

for 2003 through 2006, Figure 5.11.

The last yield function has an huge yield in the 1 nm bin, see Figure 5.11. For the same reason as before every yield from the bins below 27 nm are set to zero. Then the photoelectron current is estimated from this yield. See Figure 5.12.

The minimization procedure often results in most of the contributions being attributed to a few wavelengths, with zeros inbetween. This is not physically realistic, but it is an inevitable consequence of the positive-**A** constraint on noisy data.

5.4 An Effort to make a more Continuous Looking Yield Function

The yield function is expected to be more like a continous function of the wavelength as those in Section 5.2. With increasing wavelength it should first rise for some wavelength, have a peak (maybe more than one peak), and then finally decrease towards zero.

In an effort to make such an yield function the following function was tested:

$$\mathbf{A} = \frac{(n_1 - n_2)}{n_2 \left(\frac{\mathbf{\Lambda}}{\Lambda_0}\right)^{-n_1} - n_1 \left(\frac{\mathbf{\Lambda}}{\Lambda_0}\right)^{-n_2}} \tag{5.9}$$

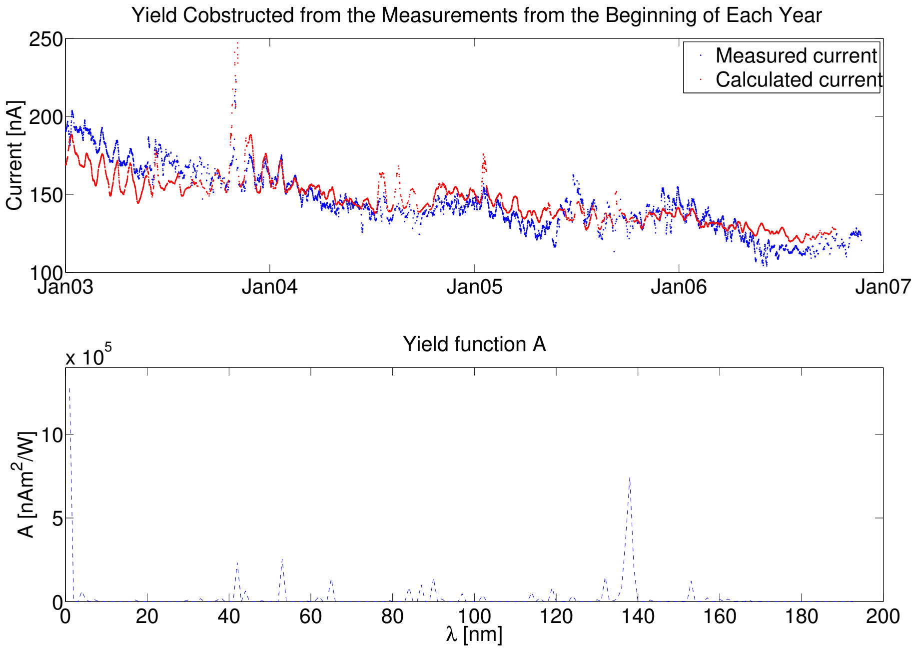


Figure 5.11: The yield from a function constructed from the measurements from the beginning of all the years (january through mars)

Λ is the entire EUV spectra, i.e. here 1 to 193 nm. The Λ_0 is the wavelength where the function is wanted to have its peak. The n_1 and n_2 parameters are values that determines how fast the function should rise and how fast it should decline. The parameters varied are n_1, n_2 and Λ_0. These parameter give the form of the curve, not its amplitude. The estimated photoelectron current is then multiplied with a factor making the first estimated current and the measured current coincide. This factor is used to make the amplitude of the yield function more accurate.

The first function constructed this way had the values: $n_1 = 40$, $n_2 = -40$ and $\Lambda_0 = 135$. This yield function with its estimated photoelectron current is shown in Figure 5.13

The parameters were chosen in many different ways. The useful functions were the ones that had few very sharp peaks at the wavelengths taken out by the MATLAB routine LSQNONNEG. This approach did not give anything new.

5.5 Exclude the Year 2003 in the Calculations

In almost all the currents calculated from different yield functions in the earlier Sections the short time variations are exaggerated. Particulary year 2003 exaggerate these variations. For this reason 2003 was excluded and some of the yield functions were used again.

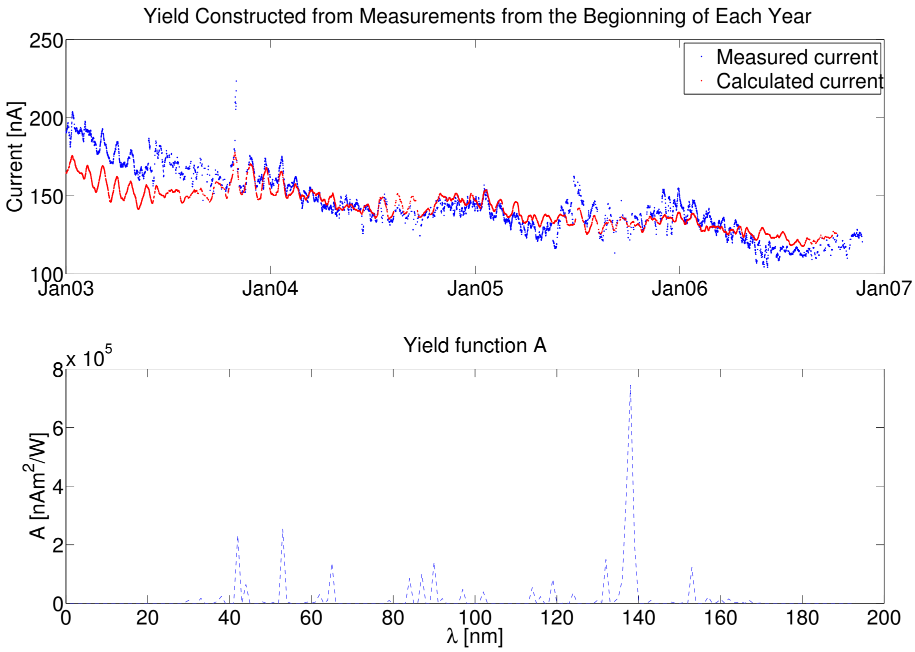


Figure 5.12: The yield from a function constructed from the measurements from the beginning of all the years. The yield from the wavelengths below 27 nm are set to zero

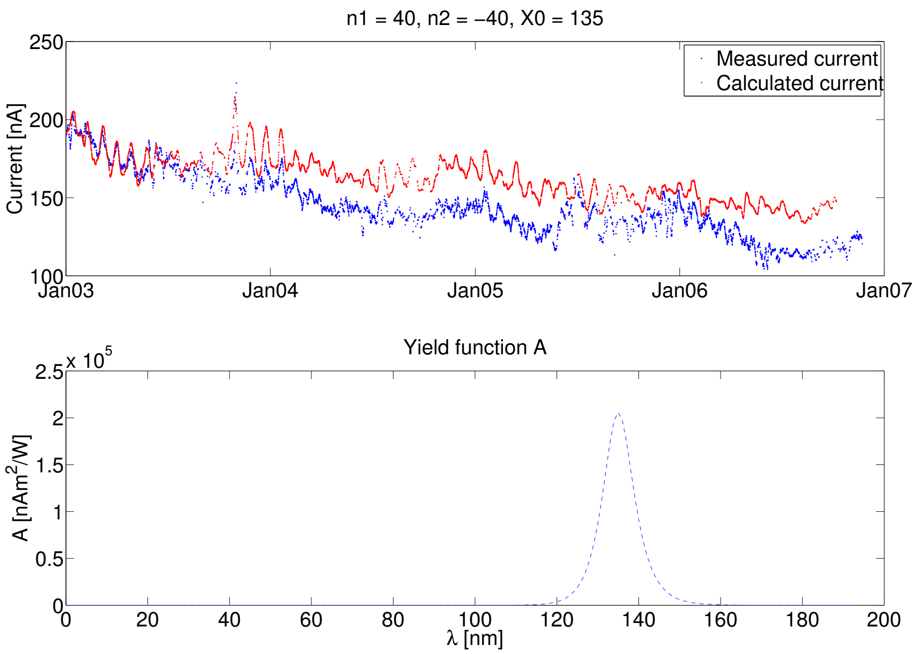


Figure 5.13: A continous yield function with its peak at 135 nm

The Cluster EFW probes had a carboneus coating at launch and thus the yield functiion for DAG-213 is of interest. Under the DAG-213 there is aluminium. For this reason these two yield functions were compared with 2003 data excluded. A new yield function using LSQNONNEG was also made to see if there would be any improvment.

In an effort to make an even better yield function the yield function for aluminium was multiplied with a factor. The factor is to make the average of the difference between calculated and measured current sufficently small. All these averages for the different yield function currents are displayed in Table 5.1. These averages are a measure of the error in the yeld function. The improved aluminium and DAG-213 yield functions are displayed with their currents in Figure 5.14. The calculated yield function for the probes using LSQNONEG is seen in Figure 5.15.

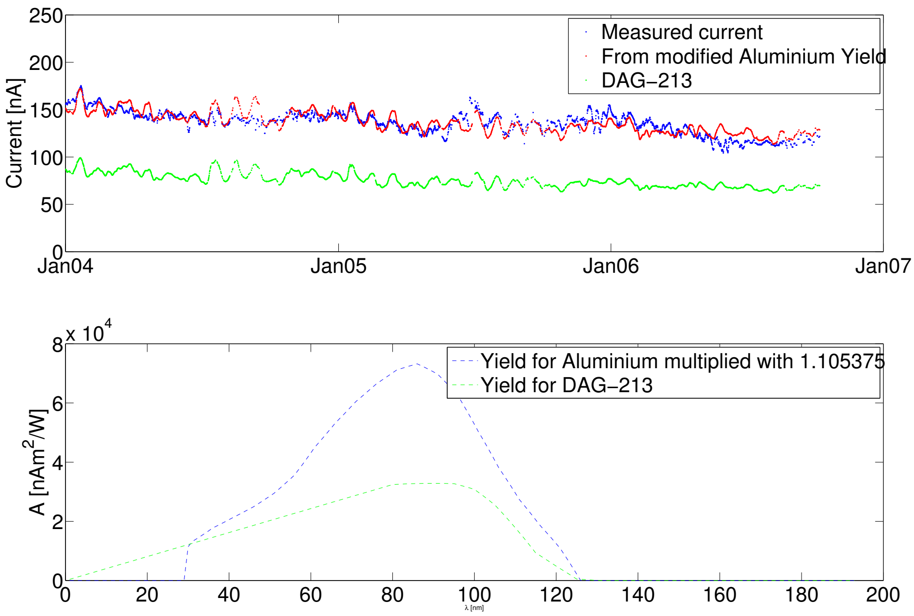


Figure 5.14: The yield functions for DAG-231 and the yield for aluminium multiplied with 1.105375 and their currents.

5.6 Results

The measured photoelectron current have three different types of periodic variatuions. The shortest time variation is 27 days. That is the same time period as the Sun's synodic[1] period. The one year period seen in the measured photoelectron current is induced by the Earth's elliptical orbit around the Sun. A peak is seen in the winter of

[1]Synodic period is the time it takes for the Sun to rotated ones aroud its axis relatively to the Earth.

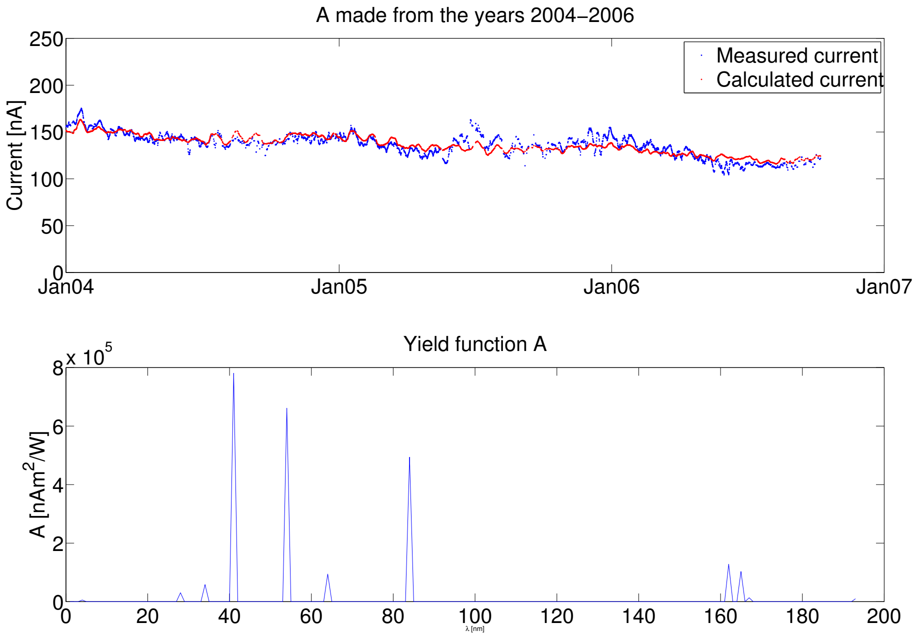


Figure 5.15: Calculated yield function without the data from 2003

each year. The third variation is the one following the Solar cycle, i.e. that the variation has a period of 11 years, see Section 3.1.

The estimated currents given by most of the yield functions exaggerate the short term variation. Even the photoelectron current estimated from the function constructed from two wavelengths that had almost the same short term and long term variation as the measured photoelectron current from the Cluster probes, Figure 5.2. In Figure 5.16 it can be seen that there is a correlation between the solar EUV flux and the current. The dispersion from a straight line in all the cases is probably due to the short term variation that is exaggerated in all these calculated currents except where 2003 is excluded.

None of the presented yield functions seems to estimate the photoelectron currents on the probes from the TIMED data with any greater accuracy. The sort term (27 days) variation appears to be attenuated by almost all the yield functions. The best estimated photoelectron currents are here made by yield functions that have discretely picked wavelengths. See for example Figure 5.6. The best is the modified Samson yield function for aluminium.

The estimated currents wiht the use of the EUV flux from TIMED and the yield funcction, has both a short and a long term variation with the same periods as the measured current has. The amplitudes of these estimated photoelectron currents are not equal to the one measured on the Cluster probes.

An approach to see the error in the calculated currents was to take an average of the

2003-2006	
DAG-213	66.372
Rhenium	57.712
Aluminium	17.123
Aluminium2	3.5410
LSQNONNEG	0.0103
2004-2006	
DAG-213	61.155
Rhenium	51.427
Aluminium	13.051
Aluminium2	0.0007
LSQNONNEG	0.0442

Table 5.1: Acverages of the differences of the calculated currents - measured current. Aluminium2 is the modified aluminium yield function described in Section 5.5.

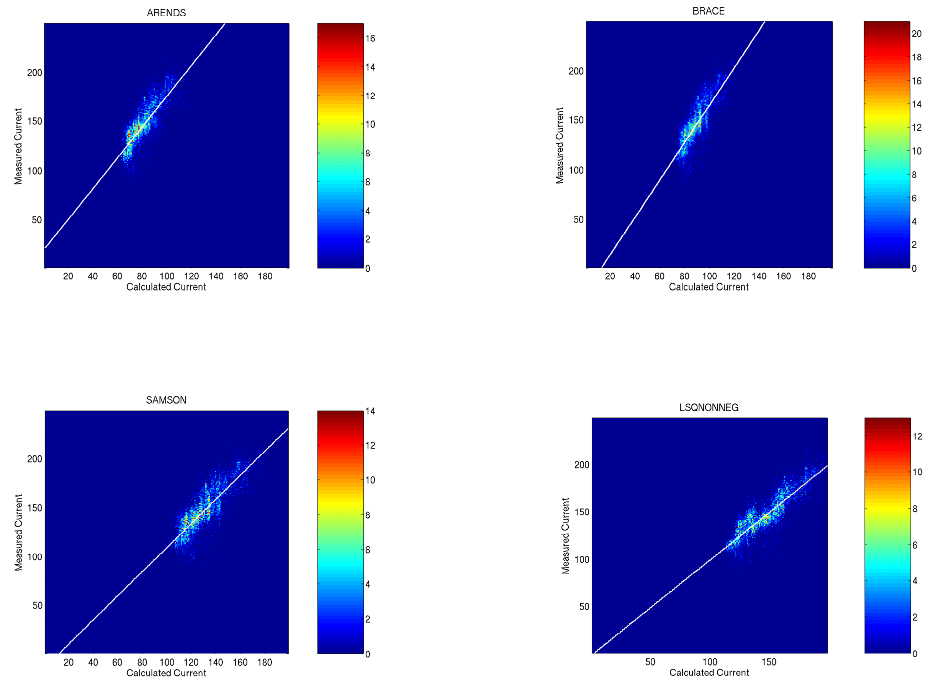


Figure 5.16: Four yield functions have been used to calculate a current. Solar EUV flux data was aquired from the TIMED data set.

differences between the calculated currents and the measured current. All these averages are shown in Table 5.1. A small average does not nececary imply a good yiled function. If half of the values are high above and the other half is equally far beneath the measured current, then the average woul be zero even though the calculated current is far from the measured one.

5.7 Conclusion

The yield function should be a continous function of the wavelength, λ. The interval should be greater than one wavelength where this function is lager than zero. Thus all calculated functions using the MATLAB routine LSQNONNEG are not plausible.

The probes on the Cluster satellites are made of aluminium and coated with a carboneous material much like the DAG-213 used in *Arends* study [2]. Since the yield function seized from Samson reproduces the photosaturation current much better than the one from Arends one can presume that there is not much left of the carboneous coating. Samson's yield function for aluminium is better during all the observed years. This implies that the coating was dissipated early in the Cluster mission. However, to see if the probe surfaces really have changed since launch, it would be necessary to study also the first years of the Cluster mission. This is outside the scope of the present study, wich concentrates on the period for wich UV data from TIMED are availably. By using som UV proxy, e.g. $F_{10.7}$, it would still be possible to evaluate any significant changes in the yield function even in absence of TIMED data, making a study reaching back to just after launch possible.

Most of the yield functions result in a higher amplitude of short-term (solar rotation) fluctuations than observed in the probe current, even though the long-term trends are reasonably modelled. Also, while long-term (annual, solar cycle) variations can be made to fit reasonably well for mmost times, there are time intervals where they do not. One may particulary note the year 2003, for wich the UV measurements predict a lower value than observed, compared to other years. We have no explanation for this behaviour. While our analysis certainly involves many steps, it is hard to see how such errors could be introduced. One should also note that all these differences are quite deterministic: the result is more or less the same for any spacecraft and any probe. Finally, it cannot be due to an erroneous yield function, as we showed that optimizing it led to unphysical forms of this function (Section 5.3. An interesting speculation is that our data set of photoemission currents could in some way be used to improve the calibrations of the TIMED/SEE instrument.

5.8 Outlook

If the yield function is known, the EFW instrument's measurement of the photoelectron current may be used to calculate the EUV flux. The resolution of the calculations will be the same as the resulotion of the yield function. This implies that thourough examinations has to be done for the materials used for the probes.

Chapter 6

The Density - Spacecraft Potential Relation

The purpose of this chapter is to see if the solar EUV radiation has any relation with the plasma density calculated from the satellite potential. The spacecraft potential's relation with the plasma density is described by *Escoubet et al.* [6].

The potential is, as is also described in Section 3.2.2, dependent on the intensity of the EUV radiation from the Sun. The amount of returning photoelectrons are also dependent on the density of the ambient plasma and thus it has an influence on the resulting satellite potential. The satellite potential may then be described as a function of the EUV flux and the density of the ambient plasma.

Since the EUV measurements from TIMED are from the upper ionospere and not above it, the measurements may be distorted and underestimated. Because of this, and the result from Chapter 5, the influence from the Sun will be evaluated by the photoelectron current on the probe (Section 3.2.3)and the EUV proxy $F_{10.7}$ (Section 3.1.1).

The plasma ion density, n_e has been plotted against the probe to satellite potential, V_{ps}. Then this plot has been compared with plots where the density has been normalized with the two different estimates of the solar EUV influence i.e the photoelectron current and the $F_{10.7}$ index. These are the three cases that have been compared.

6.1 Ion Density Calculated from Satellite Potential Measurements

The density of the ambient plasma has been calculated from a measured satellite potential by *Escoubet et al* [6]. The equation giving the plasma densities they present have the form

$$n = Aexp\left(\frac{-e(V_s - V_p)}{B}\right) + Cexp\left(\frac{-e(V_s - V_p)}{D}\right)$$

In this report, we investigate the possibility of using a power law relation,

$$n = G(V_s - V_p)^{\gamma}$$

as we found that the n-V_{sc} relation is very close to a line in a log-log diagram (Figures 6.2, 6.3 and 6.4).

6.1.1 Ion Densities Measured by CIS

The CIS instrument (Section 2.1.2) measures the ion flux in a wide range in energy and direction. From this, moments like density and velocity are computed by intergration. In some regions, high spacecraft potential and low ion energies may lead to many ions never reaching CIS, so that the density moment is underestimated. This is never a problem in the solar wind, where protons arrive at 1 keV energy and the spacecraft potential is lower than 15V. We thus use the solar wind as a good region for comparing CIS density to EFW V_{sc} measurements.

In Chapter 6, we compare CIS densities to EFW V_{sc} values obtained in the solar wind. To find solar wind periods, we have browsed data from January 2003 to March 2005 in Cluster Quicklook data set (Figure 6.1). The criterion has been a narrow energy spike in CIS spectogram (4th panel from top) aroun 1 keV. We also checked (in CSDS PP data, see Section 2.1.3) that CIS was in one of its solar wind modes (3 or 5) [1]. The timeintervals are listed in Table 6.1. Selected time intervals (Febuary and Mars) are framed in the same Table 6.1.

Table 6.1: Time intervals when the Cluster satellites are in the solar wind

Start time							End time					
Year	Month	Day	Hour	Min	Sec		Year	Month	Day	Hour	Min	Sec
2003	01	01	07	30	00	—	2003	01	01	11	30	00
2003	01	03	15	00	00	—	2003	01	03	21	00	00
2003	01	06	08	00	00	—	2003	01	06	18	00	00
2003	01	08	08	00	00	—	2003	01	08	24	00	00
2003	01	10	17	00	00	—	2003	01	10	24	00	00
2003	01	13	09	00	00	—	2003	01	13	18	00	00
2003	01	17	22	00	00	—	2003	01	18	17	00	00
2003	01	20	06	00	00	—	2003	01	20	18	00	00
2003	01	22	11	31	00	—	2003	01	23	12	00	00
2003	01	24	20	30	00	—	2003	01	25	21	00	00

Start time							End time					
Year	Month	Day	Hour	Min	Sec		Year	Month	Day	Hour	Min	Sec
2003	01	27	16	00	00	—	2003	01	28	06	29	00
2003	01	29	14	00	00	—	2003	01	30	19	00	00
2003	02	03	08	00	00	—	2003	02	04	09	00	00
2003	02	05	18	00	00	—	2003	02	06	19	00	00
2003	02	08	03	00	00	—	2003	02	09	04	00	00
2003	02	10	12	00	00	—	2003	02	11	13	00	00
2003	02	15	04	30	00	—	2003	02	16	07	30	00
2003	02	17	13	30	00	—	2003	02	18	15	00	00
2003	02	19	22	30	00	—	2003	02	21	01	30	00
2003	02	22	08	00	00	—	2003	02	23	10	30	00
2003	02	24	17	00	00	—	2003	02	25	19	00	00
2003	02	27	01	31	00	—	2003	02	28	05	00	00
2003	03	01	11	00	00	—	2003	03	02	13	30	00
2003	03	03	19	45	00	—	2003	03	04	23	00	00
2003	03	06	05	00	00	—	2003	03	07	08	00	00
2003	03	08	14	00	00	—	2003	03	09	17	10	00
2003	03	10	23	00	00	—	2003	03	12	02	00	00
2003	03	13	08	00	00	—	2003	03	14	11	10	00
2003	03	15	18	00	00	—	2003	03	16	20	10	00
2003	03	18	17	00	00	—	2003	03	19	05	10	00
2003	03	20	14	00	00	—	2003	03	21	14	10	00
2003	03	22	21	00	00	—	2003	03	23	23	00	00
2003	03	25	07	00	00	—	2003	03	26	08	00	00
2003	03	27	15	00	00	—	2003	03	28	17	00	00
2003	03	30	00	00	00	—	2003	03	31	01	50	00
2003	04	01	12	00	00	—	2003	04	02	10	50	00
2003	04	03	18	30	00	—	2003	04	04	19	45	00
2003	04	06	03	30	00	—	2003	04	07	04	30	00
2003	04	11	03	00	00	—	2003	04	11	22	20	00
2003	04	13	07	00	00	—	2003	04	14	07	10	00
2003	04	15	16	00	00	—	2003	04	16	05	30	00
2003	04	18	02	00	00	—	2003	04	19	00	30	00
2003	04	22	20	00	00	—	2003	04	23	16	00	00
2003	04	25	20	00	00	—	2003	04	26	02	00	00
2003	04	27	15	00	00	—	2003	04	28	09	30	00
2003	04	30	00	00	00	—	2003	04	30	19	59	50
2003	05	02	09	11	00	—	2003	05	03	04	25	00
2004	01	02	15	30	00	—	2004	01	03	11	30	00
2004	01	05	01	00	00	—	2004	01	05	20	00	00
2004	01	12	02	00	00	—	2004	01	12	23	30	00
2004	01	14	12	00	00	—	2004	01	14	23	30	00

Start time							End time					
Year	Month	Day	Hour	Min	Sec		Year	Month	Day	Hour	Min	Sec
2004	01	16	19	00	00	—	2004	01	17	18	00	00
2004	01	19	05	00	00	—	2004	01	20	02	00	00
2004	01	21	12	00	00	—	2004	01	21	24	00	00
2004	01	23	21	00	00	—	2004	01	24	23	00	00
2004	01	26	07	30	00	—	2004	01	27	06	30	00
2004	01	28	15	00	00	—	2004	01	29	17	10	00
2004	01	31	01	00	00	—	2004	02	01	02	00	00
2004	02	02	09	30	00	—	2004	02	03	06	00	00
2004	02	04	20	00	00	—	2004	02	05	19	00	00
2004	02	07	04	00	00	—	2004	02	08	04	00	00
2004	02	09	13	00	00	—	2004	02	10	14	55	00
2004	02	11	21	00	00	—	2004	02	12	22	30	00
2004	02	14	07	00	00	—	2004	02	15	04	00	00
2004	02	19	03	00	00	—	2004	02	19	24	00	00
2004	02	21	10	00	00	—	2004	02	22	12	00	00
2004	02	23	18	00	00	—	2004	02	24	20	00	00
2004	02	26	03	00	00	—	2004	02	27	06	00	00
2004	02	28	12	00	00	—	2004	02	29	13	00	00
2004	03	01	20	00	00	—	2004	03	02	24	00	00
2004	03	04	06	00	00	—	2004	03	05	09	30	00
2004	03	06	14	00	00	—	2004	03	07	17	00	00
2004	03	09	03	00	00	—	2004	03	10	03	30	00
2004	03	11	09	00	00	—	2004	03	12	12	00	00
2004	03	13	18	00	00	—	2004	03	14	19	00	00
2004	03	16	07	00	00	—	2004	03	17	01	00	00
2004	03	18	13	00	00	—	2004	03	19	12	00	00
2004	03	20	20	00	00	—	2004	03	21	24	00	00
2004	03	23	09	00	00	—	2004	03	24	08	00	00
2004	03	25	15	00	00	—	2004	03	26	18	00	00
2004	03	28	12	00	00	—	2004	03	29	01	00	00
2004	03	30	11	00	00	—	2004	03	31	10	00	00
2004	04	01	19	00	00	—	2004	04	02	18	00	00
2004	04	04	12	00	00	—	2004	04	05	02	00	00
2004	04	06	18	00	00	—	2004	04	07	12	00	00
2004	04	09	00	00	00	—	2004	04	09	23	00	00
2004	04	11	07	00	00	—	2004	04	12	07	00	00
2004	04	16	01	00	00	—	2004	04	17	01	00	00
2004	04	18	12	00	00	—	2004	04	19	10	00	00
2004	04	20	20	00	00	—	2004	04	21	18	00	00
2004	04	23	05	00	00	—	2004	04	24	03	00	00
2004	04	25	18	00	00	—	2004	04	26	05	00	00

Start time							End time					
Year	Month	Day	Hour	Min	Sec		Year	Month	Day	Hour	Min	Sec
2004	04	28	06	00	00	—	2004	04	28	20	00	00
2005	01	02	19	00	00	—	2005	01	02	23	00	00
2005	01	04	17	00	00	—	2005	01	04	23	00	00
2005	01	07	06	00	00	—	2005	01	07	22	00	00
2005	01	10	00	00	00	—	2005	01	07	07	00	00
2005	01	11	18	00	00	—	2005	01	12	17	00	00
2005	01	14	03	00	00	—	2005	01	15	01	00	00
2005	01	23	14	00	00	—	2005	01	24	14	00	00
2005	01	26	01	00	00	—	2005	01	26	23	00	00
2005	01	28	07	00	00	—	2005	01	29	10	00	00
2005	01	30	16	00	00	—	2005	01	31	17	00	00
2005	02	02	11	00	00	—	2005	02	03	03	00	00
2005	02	04	14	00	00	—	2005	02	05	12	00	00
2005	02	06	18	00	00	—	2005	02	07	15	00	00
2005	02	11	12	00	00	—	2005	02	12	14	00	00
2005	02	13	21	00	00	—	2005	02	14	24	00	00
2005	02	16	06	00	00	—	2005	02	17	10	00	00
2005	02	18	15	00	00	—	2005	02	19	18	00	00
2005	02	21	08	00	00	—	2005	02	22	04	00	00
2005	02	23	09	00	00	—	2005	02	24	12	00	00
2005	02	25	18	00	00	—	2005	02	26	20	00	00
2005	02	28	03	00	00	—	2005	03	01	07	00	00
2005	03	02	12	00	00	—	2005	03	03	15	00	00
2005	03	04	21	00	00	—	2005	03	05	24	00	00

6.2 Correlating the Three Quantities in the Solar Wind

We looked at the times when the Cluster satellites were in the solar wind, as there most ions reach the CIS instrument due to their high energy (Section 6.1.1). To have a high possibility that the satellites are in the solar wind and not in the magnetosphere, measurements from the months of febuary and mars were used. From the years of 2003, 2004 and 2005 CIS had accesible data. In Table 6.1 all the solar wind exposures are shown. To be sure that the satellites are in a solar wind mode, i.e. they measure the solar wind, the start and end time were checked for each interval. If the satellites were shown to be in a solar wind mode at both of these points, it was assumed that the satellites were in solar wind mode during the entire time interval.

The Cluster satellites has an ion cannon (ASPOC) that lowers the satellites potential by emitting these positive charges (as the satellites attain positive potentials in the teneous

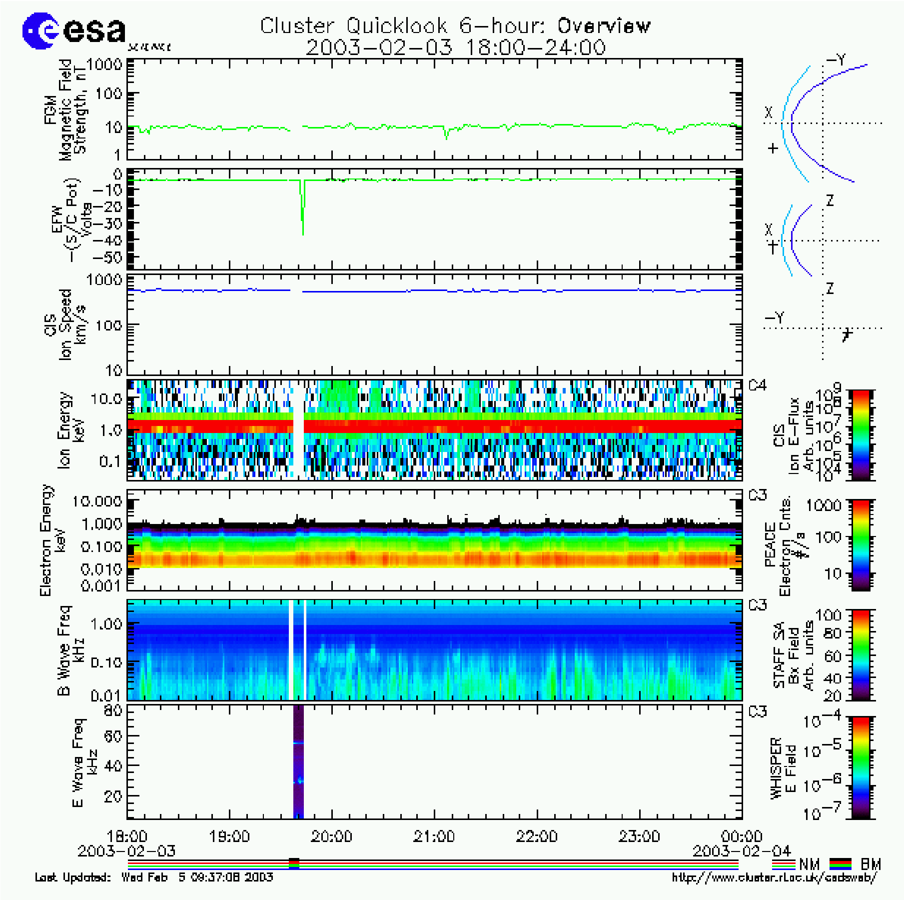


Figure 6.1: The middle plot shows the Ion energy and it is consentrated around 1keV (Adapted from http://www.cluster.rl.ac.uk/csdsweb/index.html)

plasma). ASPOCs interferense with the satellite potential could be devastatin for any conclusions. Therefore satellite 1 has been chosesen for its nonfunctional ion cannon.

To see a correlation between the three quantities, all of them have to be known. In order to correlate the them in time the following were done for every time interval. The density measurements from CIS and the satellite potential from the EFW instrument were aquired from the ISDAT server. If one of these quantites were absent during the interval for some reason, that interval was excluded and we moved on to the next. The interval was only kept if there was enough data points from both of these two quantities, in this case more than one point was required. We took the logarithm (base 10) of the density, n, and satellite potential, V_{sc}, and made of them a least squares fit to a straight line. To get an approximate density from the line we took $n_{approx} = 10^{line}$. The difference $dn = n - n_{approx}$ was then made. The standard deviation of this difference was a value of the dispersion in n. When this had been done the photoelectron current and measurements of the $F_{10.7}$ index was wanted for this time interval. The photoelectron current was available, at best, every fourth hour. This implies that we had less points of this current than we had the densities and potentials (which were measured every fourth second). The same was the case with the $F_{10.7}$ measurements as it was measured, usually at CET 17.00, 21.00 and 23.00. Since the photoelectron current and the $F_{10.7}$ measurements have a short time period of 27 days (Section 5.1) they can be estimated by the means in the current time interval, since the time intervals at the most are for 30 hours. If any of the EUV proxies were absent in one interval then that interval was discarded.

For a first verification if there was any improvement when the EUV is taken into account the density was first normalized with the $F_{10.7}$ index and then with the photoelectron current. When all the intervals were done for all these three cases (no EUV flux normalization, normalization with $F_{10.7}$ and normalization with the photoelectron current I_{ph}) a new least square fit to a straight line was made for all the three cases seperately. The differences between the estimated values of the fit and the actual values (normalized or otherwise) were taken for all three cases. As an estimation of how good the correlation is the standard deviation was taken for these three cases. The lower the value of this standard deviation the less dispersion of the pints, i.e. the lower value the better.

6.3 Results

First the logarithm of the plasma density n was plotted against the logarithm of the satellite potential V_{sc}. The result is shown in Figure 6.2. The standard deviation of the difference was 0.9911. The distribution of dn for this case is shown in Figure 6.5. We have almost 1 million points in the Figures 6.2, 6.3 and 6.4.

In Figure 6.3 the plasma densities are normalized by a factor $\frac{<F_{10.7}>}{F_{10.7}(t)}$. The distribution of dn for this case is shown in Figure 6.5. The standard deviation for this distribution has the value of 0.8744.

The next normalizing factor that was used was $\frac{<I_{ph}>}{I_{ph}(t)}$. The result is shown in Figure 6.4. The value of the standard deviation was 0.8097 for the distribution of dn shown in Figure 6.5.

density	n	$\frac{n<F_{10.7}>}{F_{10.7}}$	$\frac{n<I_{ph}>}{I_{ph}}$
σ	0.9911	0.8744	0.8097

Table 6.2: The standard deviation of the difference n for the three cases

Since we were not able to construct the photoelectron saturation current from TIMED data, this data has not been used to normalize the density.

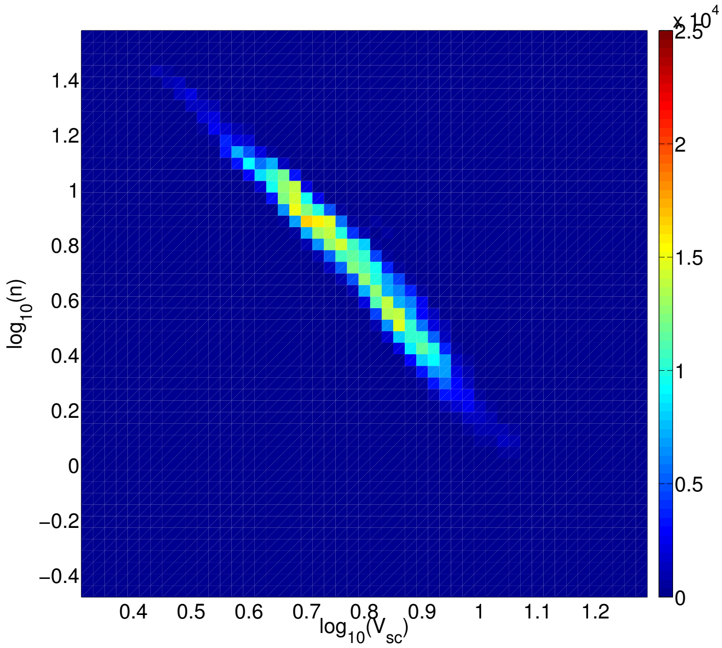


Figure 6.2: $\mathrm{Log}_{10}(n)$ versus $\mathrm{Log}_{10}(V_{sc})$. This is data uncorrelated with any EUV flux proxy.

6.4 Conclusion

As seen in Figures 6.2, 6.3 and 6.4, the unnormalized Figure, has a greater dispersion of the points than the latter two. We see that the density normalized with the photoelectron current has a little less dispersion than the one with $F_{10.7}$. It is more clearly shown by the values of the standard deviations also presented in Table 6.2. The standard deviation is of the distribution of the three different dn. These distributions are shown in Figure 6.5

This implies that the correlation between the plasma density and the satellite potential should become more accurate with the solar EUV flux as an additional input.

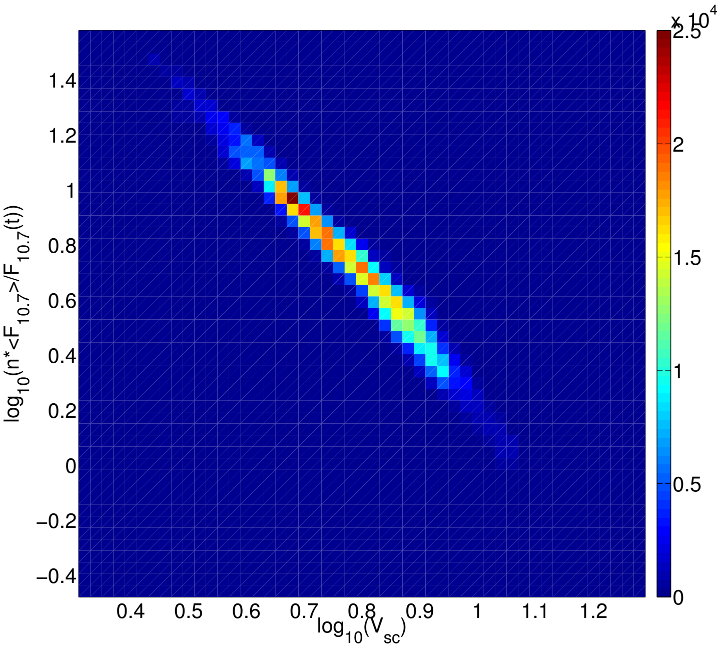


Figure 6.3: $\text{Log}_{10}(\frac{n<\text{F}_{10.7}>}{\text{F}_{10.7}})$ versus $\text{Log}_{10}(V_{sc})$. Normalized with the EUV proxy $\text{F}_{10.7}$.

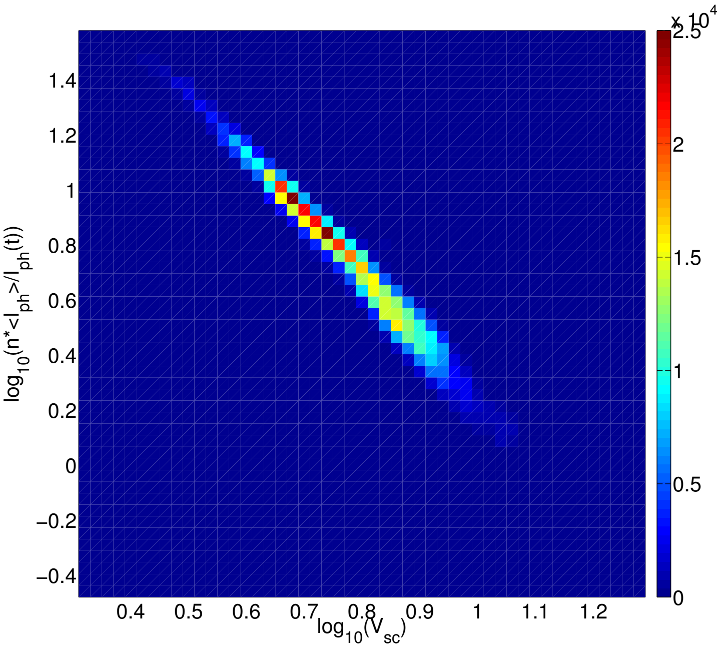


Figure 6.4: $\text{Log}_{10}(\frac{n<I_{ph}>}{I_{ph}})$ versus $\text{Log}_{10}(V_{sc})$. Tho photoelectron current is used as a EUV proxy

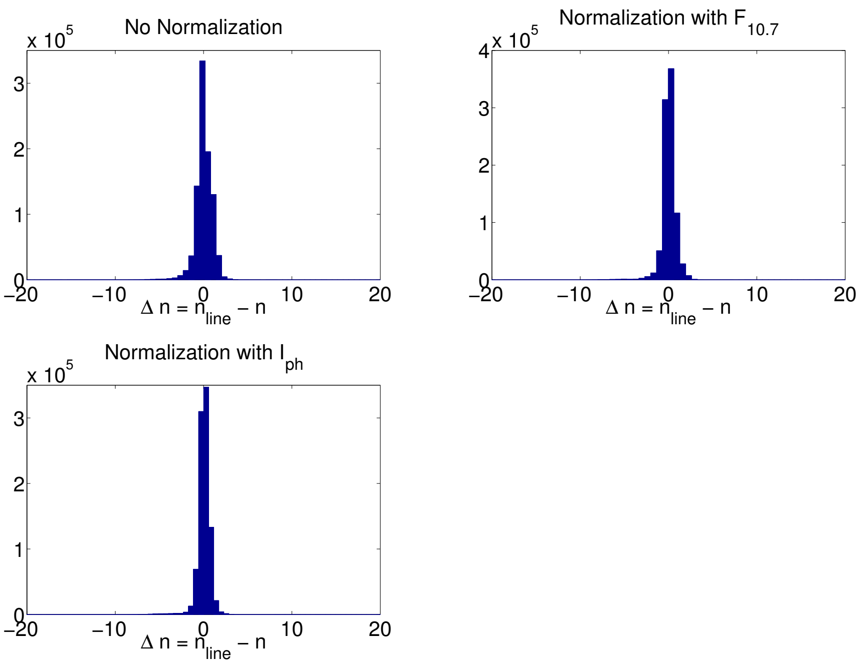


Figure 6.5: The distributions of the three different dn.

Chapter 7

Discussion & Conclusion

In this report three major topics have been examined. The first was to see if there was any temporal induced error in the EFW components. This was not the case. The second was an attemt to find the yield function by combining time series of Cluster photoemission measurements with TIMED UV spectra. This attempt failed to give a physically reasonable yield function, but we could show that laboratory yield functions give reasonably agreement with measured current, at least at longer time scales (anual and solar cycle). For our continued work, the photoemission saturation current was used as the best proxy for the solar EUV flux. Finally the relation between the satellite potential and the plasma density was modified with the solar EUV flux. Here the solar EUV flux was presented as the measured photoelectron saturation current measured on the probes. It was clearly seen that the density measurements were improved when they where normalized with the photoelectron saturation current.

Acknowledgements

This master thesis has been done at the institute of space physics in Uppsala. I would like to thank my supervisor Anders Eriksson for his intrest in the progress of the work. Another person great help has been Mikael Lundberg. He helped me greatly with MATLAB routines. Ronan Modolo made sure that the TIMED satellite was at high enough altitude. For this he has my sincere thanks. Lastly I would want to thank my fellow master thersis workers at IRFU who has been great companions at the coffe table and to test ideas with.

Bibliography

[1] *Users Guide to the Cluster Science Data System.*

[2] H. Arends. Photoelectric Uniformity of DAG 213 Used as a Coating Material for a Satellite Electric Field Probe. Technical report, European Space Agency, 1981.

[3] L. H. Brace, W. R. Hoegy, and R. F. Theis. Solar EUV Measurement at Venus Based on Photoelectron Emission From the Pioneer Venus Langmuir Probe. *Journal of Geophysical Research (Space Physics)*, 93:7282–7296, July 1988.

[4] A. E. Covington. Internal Precision of the Daily Radio Flux Observations at 10.7 Centimetres. *R.A.S.C. Jour.*, 53:156–161, August 1959.

[5] E. Engwall. Cold Magnetospheric Plasma Flows: Properties and Interaction with Spacecraft, 2006.

[6] C. P. Escoubet, A. Pedersen, and R. Schmidt. Density in the Magnetosphere Infered from ISEE 1 Spacecraft Potential. *Journal of Geophysical Research*, 102:17595–17609, August 1997.

[7] B. Feuerbacher and B. Fitton. Experimental investigation of photoemission from satellite surface materials. *Journal of Applied Physics*, 43:1563–1572, April 1971.

[8] G. Gustafsson, R. Boström, B. Holback, G. Holmgren, A. Lundgren, K. Stasiewicz, L. Ahlén, F. S. Mozer, D. Pankow, P. Harvey, P. Berg, R. Ulrich, A. Pedersen, R. Schmidt, A. Butler, A. W. C. Fransen, D. Klinge, M. Thomsen, C.-G. Fälthammar, P.-A. Lindqvist, S. Christenson, J. Holtet, B. Lybekk, T. A. Sten, P. Tanskanen, K. Lappalainen, and J. Wygant. The electric field and wave experiment for the Cluster mission. *Space. Sci. Rev.*, 79:137–156, 1997.

[9] C. L Lawson and R. J Hanson. *Solving Least Squares Problems.* Siam, 1974.

[10] A. Pedersen, P. Decreau, C. P. Escoubet, G. Gustafsson, H. Laakso, P. A. Lindqvist, B. Lybekk, A. Masson, F. Mozer, and A. Vaivads. Four-point high time resolution information on electron densities by the electric field experiments (EFW) in Cluster. *Annales Geophysicae*, 19:1483–1489, June 2001.

[11] H. Reme, J. M. Bosqued, J. A. Sauvaud, A. Cros, J. Dandouras, C. Aoustin, J. Bouyssou, T. Camus, J. Cuvilo, C. Martz, J. L. Medale, H. Perrier, D. Romefort, J. Rouzaud, C. D'Uston, E. Mobius, K. Crocker, M. Granoff, L. M. Kistler, M. Popecki, D. Hovestadt, B. Klecker, G. Paschmann, M. Scholer, C. W. Carlson, D. W. Curtis, R. P. Lin, J. P. McFadden, V. Formisano, E. Amata, M. B. Bavassano-Cattaneo, P. Baldetti, G. Belluci, R. Bruno, G. Chionchio, A. di Lellis, E. G. Shelley, A. G. Ghielmetti, W. Lennartsson, A. Korth, H. Rosenbauer, R. Lundin, S. Olsen, G. K. Parks, M. McCarthy, and H. Balsiger. The Cluster Ion Spectrometry (CIS) Experiment. *Space Science Reviews*, 79:303–350, January 1997.

[12] P. G. Richards, J. A. Fennelly, and D. G. Torr. EUVAC: A solar EUV flux model for aeronomic calculations. *Journal of Geophysical Research*, 99:8981–8992, May 1994.

[13] J. A. R. Samson and R. B Cairns. Photoelectric Yield of Aluminium from 300 to 1300Å. *The Review of Scientific Instruments*, 36:19–21, January 1965.

[14] K. F. Tapping and D. P Charrois. Limits in the Accuracy of the 10.7 cm Flux. *Solar Physics*, 150:305–315, March 1993.

[15] W. K. Tobiska. Validating the solar EUV proxy, $E_{10.7}$. *Journal of Geophysical Research*, 106:29969–29978, December 2001.

[16] W. K. Tobiska, T. Woods, F. Eparvier, R. Viereck, L. Floyd, D. Bouwer, G. Rottman, and O. R. White. The SOLAR2000 empirical solar irradiance model and forcast tool. *Journal of Atmospheric and Solar-Terrestial Physics*, 62:1233–1250, April 2000.

[17] T. N. Woods, F. G. Eparvier, S. M. Bailey, P. C. Chamberlin, J. Lean, G. J. Rottman, S. C. Solomon, and D. L. Tobiska, W. K. Woodraska. Solar EUV Experiment (SEE): Mission overview and first results. *Journal of Geophysical Reasearch*, 110, January 2005.

[18] T. N. Woods, Tobiska W. K., G. J. Rottman, and Worden J. R. Improved solar Lyman α irradiance modeling from 1947 through 1999 based on UARS obsservations. *Journal of Geophysical Reasearch*, 105:27195–27215, December 2000.

Appendix A

MATLAB Routines

A.1 hunter.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%   hunter.m
%
%   Recieves and saves all the sweep data for the probe.
%
%   Erik Winkler, IRF-Uppsala, 2006
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%clear;
warning off

interval = 200;
db = Mat_DbOpen('db:10');

%-------------------------------------------------------------------------
%   Manually change the satellite number and probe number
%-------------------------------------------------------------------------
% satellite ='1';
% probe = 'p3';

%-------------------------------------------------------------------------
%   Take data from for all the four year
%-------------------------------------------------------------------------
for i = 2003:2006

   starttime= [i 01 01 01 00 00];
   endtime = [i 12 31 23 00 00];
   duration = toepoch(endtime)-toepoch(starttime);
   t = toepoch(starttime);
   position = [];
   data=[];
   while t < toepoch(endtime)

         time = fromepoch(t);
         [temp,postemp] = hunt(time,interval,satellite,probe,db);


          %Set returned data
          if (¬isempty(temp))

              ldata = length(temp(:,1));
              if isempty(data) || ldata==66
                  data = [data; temp];
                  position = [position; postemp];
                  %Only return the data if it is a whole sweep
               else
                  disp('Kastat svep')
               end
```

```
          end

          t = t+interval;
    end


           D = strcat('¬/Data/dataTM',num2str(starttime(1)),'s',satellite,probe,'.txt');
           P = strcat('¬/Data/positionTM',num2str(starttime(1)),'s',satellite,probe,'.txt')
           save(D, 'data', '-ASCII', '-double');
           save(P, 'position', '-ASCII', '-double');
end

Mat_DbClose(db);
```

A.2 hunt.m

```
function [data position]= hunt(starttime,interval,satellite,probe,db)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% HUNT.m
%%
%% Return time, in year, month, day, hour, min, sec and a column with sec
%% since 1-jan-00:00:00 1970. Then the voltage and current measurements
%% for that time. The voltage and currents are returned in TM units.
%%
%% data = [yyyy mm dd hh min sec epoch V I]
%% position = [ mean(epoch) x y z vpsmean]   x,y,z in geocentric
%% coordinates with x pointed in the direction of the sun, for every sweep
%%
%% Erik Winkler, IRF, Uppsala, 2006
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% warning off MATLAB:polyfit:RepeatedPointsOrRescale
warning off;

%--------------------------------------------------------------------------
%   Gets the currents and voltages for the time interval for required
%   satellite and probe.
%--------------------------------------------------------------------------
[tefw, TMi] = isGetTmData(db, starttime, interval, ...
        'Cluster', satellite, 'efw','N',probe,'sweep','current');

[tefw, TMV] = isGetTmData(db, starttime, interval, ...
        'Cluster', satellite, 'efw','N',probe,'sweep','Vbias');

%--------------------------------------------------------------------------
%   If there is data returned for the timeinterval
%--------------------------------------------------------------------------
if(¬isempty(TMV))

    %--------------------------
    %   Want to see the program's
    %   progress
    %--------------------------
    disp(starttime)

    switch satellite

        case {'1','3'}

            for p = 3:4

                pr = strcat('p',num2str(p));
                %----------------------------------------------------------
                %   Want the probe to satellite potential for the times
                %   around the bias voltage sweep. Leave out 4 second in
                %   total.
```

```
        %------------------------------------------------------------
        t0vps = fromepoch(mean(tefw)-10);
        [tvpstemp1, TMvpstemp1] = isGetTmData(db, t0vps, 8, ...
                'Cluster', satellite, 'efw','E',pr,'10Hz','lx');


        t0vps = fromepoch(mean(tefw)+2);
        [tvpstemp2, TMvpstemp2] = isGetTmData(db, t0vps, 8, ...
                'Cluster', satellite, 'efw','E',pr,'10Hz','lx');



        %------------------------------------------------------------
        %   Add the values from the two intervals around the sweep
        %   in the same vector. For the purpose to get a mean of
        %   the potential
        %------------------------------------------------------------
        TMvps = [TMvpstemp1;TMvpstemp2];

        %------------------------------------------------------------
        %   Makes the variables that this function will return
        %------------------------------------------------------------
        vpsmtm(p-2)=mean(TMvps);
    end

case {'2', '4'}

    for p = 1:4
        pr = strcat('p',num2str(p));
        %------------------------------------------------------------
        %   Want the probe to satellite potential for the times
        %   around the bias voltage sweep. Leave out 4 second in
        %   total.
        %------------------------------------------------------------
        t0vps = fromepoch(mean(tefw)-10);
        [tvpstemp1, TMvpstemp1] = isGetTmData(db, t0vps, 8, ...
                'Cluster', satellite, 'efw','E',pr,'10Hz','lx');


        t0vps = fromepoch(mean(tefw)+2);
        [tvpstemp2, TMvpstemp2] = isGetTmData(db, t0vps, 8, ...
                'Cluster', satellite, 'efw','E',pr,'10Hz','lx');



        %------------------------------------------------------------
        %   Add the values from the two intervals around the sweep
        %   in the same vector. For the purpose to get a mean of
        %   the potential
        %------------------------------------------------------------
        TMvps = [TMvpstemp1;TMvpstemp2];

        %------------------------------------------------------------
```

```
            %   Makes the variables that this function will return
            %--------------------------------------------------------------
            vpsmtm(p)=mean(TMvps);
        end
    end

    %------------------------------------------------------------------------
    %   Takes the mean of all the paired probes on the spacecraft as an
    %   approximation of the spacecraft's potential. It means that for
    %   satellite 1 & 3 it is a mean of only two probes and in the case
    %   of satellite 2 & 4 all four probes.
    %------------------------------------------------------------------------
    vpsTM = mean(vpsmtm);

    %------------------------------------------------------------------------
    %   Position of the satellite
    %------------------------------------------------------------------------
    [tpos, pos] = isGetTmData(db,starttime, interval, ...
            'Cluster', satellite, 'ephemeris', 'position','','','');



    %------------------------------------------------------------------------
    %   Return the current, voltage, position,
    position = [mean(tefw) pos(1,1) pos(2,1) pos(3,1) vpsTM];
    data = [fromepoch(tefw) tefw TMV TMi];

%----------------------------------------------------------------------------
%   If there is no data in the timeinterval requested the function returns
%   empty vectors
%----------------------------------------------------------------------------
else

    position = [];
    data = [];

end
```

A.3 CIS.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%   CIS.m
%
%   Have all the intervals that the Cluster satellites are in the solar
%   wind in the spring time (feb-mars). During these times, CIS aquire the
%   density measurements,photoelectron saturation currents and the probe to satellite
%   potential.
%
%   It then plots in a colorful way the log of the density (not normalizes,
%   normalized with the F_{10.7} and normalized with the photoelectron
%   saturation current) against the log of the satellite potential (-Vps)
%
%   Erik Winkler, IRF-Uppsala, 2007
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
t_efw    = [];
t_cis    = [];
n_cis    = [];
V_efw    = [];
I        = [];
F        = [];

%--------------------------------------------------------------------------
%   Load the improved alphas for each probe. Alpha is a matrix with the
%   satellite nr given by the row and probe number given by the column
%--------------------------------------------------------------------------
load Alpha_new.txt;
%load Flux_mean_anti_corrected.txt;
[t_F flux] = getflux([toepoch([2003 01 01 00 00 00]) ...
    toepoch([2006 04 01 00 00 00])],340);

%--------------------------------------------------------------------------
%   Want all the years from Read_allData
%--------------------------------------------------------------------------
bool_one_year_only = 0;

%--------------------------------------------------------------------------
%   Satellite and probe
%--------------------------------------------------------------------------
satellite = '1';
probe     = 'p3';

%--------------------------------------------------------------------------
%   Takes out the correct alpha
%--------------------------------------------------------------------------
alpha = Alpha_new(str2double(satellite),str2double(probe(2)));

%--------------------------------------------------------------------------
```

```
%   Retrieve the satellite potential of the satellite and the photoelectron
%   current on the probe
%----------------------------------------------------------------------------
Read_allData

%----------------------------------------------------------------------------
%   All the potentials in one vector
%----------------------------------------------------------------------------
Vp = [vpm1;vpm2;vpm3;vpm4];

%----------------------------------------------------------------------------
%   I_ph = [I_ph1;I_ph2;I_ph3;I_ph4] and
%   t_I  = [t_I1;t_I2;t_I3;t_I4] are already made in Read_allData.
%----------------------------------------------------------------------------



%----------------------------------------------------------------------------
%             Timeintervals when Cluster is in the Solar Wind. Taken from
%             CSDSWEB plots.
%----------------------------------------------------------------------------
t_interval = [
%-----------------------------------------2003-----------------------------------------%
%       toepoch([2003 01 01 07 30 00]) toepoch([2003 01 01 11 30 00]);    %OK
%       toepoch([2003 01 03 15 00 00]) toepoch([2003 01 03 21 00 00]);    %OK
%       toepoch([2003 01 06 08 00 00]) toepoch([2003 01 06 18 00 00]);    %OK
%       toepoch([2003 01 08 08 00 00]) toepoch([2003 01 08 24 00 00]);    %OK
%       toepoch([2003 01 10 17 00 00]) toepoch([2003 01 10 24 00 00]);    %OK
%       toepoch([2003 01 13 09 00 00]) toepoch([2003 01 13 18 00 00]);    %OK
%       toepoch([2003 01 17 22 00 00]) toepoch([2003 01 18 17 00 00]);    %OK
%       toepoch([2003 01 20 06 00 00]) toepoch([2003 01 20 18 00 00]);    %OK
%       toepoch([2003 01 22 11 31 00]) toepoch([2003 01 23 12 00 00]);    %OK
%       toepoch([2003 01 24 20 30 00]) toepoch([2003 01 25 21 00 00]);    %OK
%       toepoch([2003 01 27 16 00 00]) toepoch([2003 01 28 06 29 00]);    %OK
%       toepoch([2003 01 29 14 00 00]) toepoch([2003 01 30 19 00 00]);    %OK
      toepoch([2003 02 03 08 00 00]) toepoch([2003 02 04 09 00 00]);    %OK
      toepoch([2003 02 05 18 00 00]) toepoch([2003 02 06 19 00 00]);    %OK
      toepoch([2003 02 08 03 00 00]) toepoch([2003 02 09 04 00 00]);    %OK
      toepoch([2003 02 10 12 00 00]) toepoch([2003 02 11 13 00 00]);    %OK
      toepoch([2003 02 12 16 00 00]) toepoch([2003 02 13 23 00 00]);    %OK s3 & s4, s1 no data
      toepoch([2003 02 15 04 30 00]) toepoch([2003 02 16 07 30 00]);    %OK
      toepoch([2003 02 17 13 30 00]) toepoch([2003 02 18 15 00 00]);    %OK
      toepoch([2003 02 19 22 30 00]) toepoch([2003 02 21 01 30 00]);    %OK
      toepoch([2003 02 22 08 00 00]) toepoch([2003 02 23 10 30 00]);    %OK
      toepoch([2003 02 24 17 00 00]) toepoch([2003 02 25 19 00 00]);    %OK
      toepoch([2003 02 27 01 31 00]) toepoch([2003 02 28 05 00 00]);    %OK
      toepoch([2003 03 01 11 00 00]) toepoch([2003 03 02 13 30 00]);    %OK
      toepoch([2003 03 03 19 45 00]) toepoch([2003 03 04 23 00 00]);    %OK
      toepoch([2003 03 06 05 00 00]) toepoch([2003 03 07 08 00 00]);    %OK
      toepoch([2003 03 08 14 00 00]) toepoch([2003 03 09 17 10 00]);    %OK
      toepoch([2003 03 10 23 00 00]) toepoch([2003 03 12 02 00 00]);    %OK
      toepoch([2003 03 13 08 00 00]) toepoch([2003 03 14 11 10 00]);    %OK
      toepoch([2003 03 15 18 00 00]) toepoch([2003 03 16 20 10 00]);    %OK
      toepoch([2003 03 18 17 00 00]) toepoch([2003 03 19 05 10 00]);    %OK
```

```
     toepoch([2003 03 20 14 00 00]) toepoch([2003 03 21 14 10 00]);    %OK
     toepoch([2003 03 22 21 00 00]) toepoch([2003 03 23 23 00 00]);    %OK
     toepoch([2003 03 25 07 00 00]) toepoch([2003 03 26 08 00 00]);    %OK
     toepoch([2003 03 27 15 00 00]) toepoch([2003 03 28 17 00 00]);    %OK
     toepoch([2003 03 30 00 00 00]) toepoch([2003 03 31 01 50 00]);    %OK
     toepoch([2003 04 01 12 00 00]) toepoch([2003 04 02 10 50 00]);    %OK
%      toepoch([2003 04 03 18 30 00]) toepoch([2003 04 04 19 45 00]);    %OK
%      toepoch([2003 04 06 03 30 00]) toepoch([2003 04 07 04 30 00]);    %OK
%      toepoch([2003 04 11 03 00 00]) toepoch([2003 04 11 22 20 00]);    %OK
%      toepoch([2003 04 13 07 00 00]) toepoch([2003 04 14 07 10 00]);    %OK
%      toepoch([2003 04 15 16 00 00]) toepoch([2003 04 16 05 30 00]);    %OK
%      toepoch([2003 04 18 02 00 00]) toepoch([2003 04 19 00 30 00]);    %OK
%      toepoch([2003 04 22 20 00 00]) toepoch([2003 04 23 16 00 00]);    %OK
%      toepoch([2003 04 25 20 00 00]) toepoch([2003 04 26 02 00 00]);    %OK
%      toepoch([2003 04 27 15 00 00]) toepoch([2003 04 28 09 30 00]);    %OK
%      toepoch([2003 04 30 00 00 00]) toepoch([2003 04 30 19 59 50]);    %OK
%      toepoch([2003 05 02 09 11 00]) toepoch([2003 05 03 04 25 00]);    %OK
%-----------------------------------2004-----------------------------------%
%      toepoch([2004 01 02 15 30 00]) toepoch([2004 01 03 11 30 00]);    %OK
%      toepoch([2004 01 05 01 00 00]) toepoch([2004 01 05 20 00 00]);    %OK
%      toepoch([2004 01 12 02 00 00]) toepoch([2004 01 12 23 30 00]);    %OK
%      toepoch([2004 01 14 12 00 00]) toepoch([2004 01 14 23 30 00]);    %OK
%      toepoch([2004 01 16 19 00 00]) toepoch([2004 01 17 18 00 00]);    %OK
%      toepoch([2004 01 19 05 00 00]) toepoch([2004 01 20 02 00 00]);    %OK
%      toepoch([2004 01 21 12 00 00]) toepoch([2004 01 21 24 00 00]);    %OK
%      toepoch([2004 01 23 21 00 00]) toepoch([2004 01 24 23 00 00]);    %OK
%      toepoch([2004 01 26 07 30 00]) toepoch([2004 01 27 06 30 00]);    %OK
%      toepoch([2004 01 28 15 00 00]) toepoch([2004 01 29 17 10 00]);    %OK
%      toepoch([2004 01 31 01 00 00]) toepoch([2004 02 01 02 00 00]);    %OK
     toepoch([2004 02 02 09 30 00]) toepoch([2004 02 03 06 00 00]);    %OK
     toepoch([2004 02 04 20 00 00]) toepoch([2004 02 05 19 00 00]);    %OK
     toepoch([2004 02 07 04 00 00]) toepoch([2004 02 08 04 00 00]);    %OK
     toepoch([2004 02 09 13 00 00]) toepoch([2004 02 10 14 55 00]);    %OK
     toepoch([2004 02 11 21 00 00]) toepoch([2004 02 12 22 30 00]);    %OK
     toepoch([2004 02 14 07 00 00]) toepoch([2004 02 15 04 00 00]);    %OK
     toepoch([2004 02 19 03 00 00]) toepoch([2004 02 19 24 00 00]);    %OK
     toepoch([2004 02 21 10 00 00]) toepoch([2004 02 22 12 00 00]);    %OK
     toepoch([2004 02 23 18 00 00]) toepoch([2004 02 24 20 00 00]);    %OK
     toepoch([2004 02 26 03 00 00]) toepoch([2004 02 27 06 00 00]);    %OK
     toepoch([2004 02 28 12 00 00]) toepoch([2004 02 29 13 00 00]);    %OK
     toepoch([2004 03 01 20 00 00]) toepoch([2004 03 02 24 00 00]);    %OK
     toepoch([2004 03 04 06 00 00]) toepoch([2004 03 05 09 30 00]);    %OK
     toepoch([2004 03 06 14 00 00]) toepoch([2004 03 07 17 00 00]);    %OK
     toepoch([2004 03 09 03 00 00]) toepoch([2004 03 10 03 30 00]);    %OK
     toepoch([2004 03 11 09 00 00]) toepoch([2004 03 12 12 00 00]);    %OK
     toepoch([2004 03 13 18 00 00]) toepoch([2004 03 14 19 00 00]);    %OK
     toepoch([2004 03 16 07 00 00]) toepoch([2004 03 17 01 00 00]);    %OK
     toepoch([2004 03 18 13 00 00]) toepoch([2004 03 19 12 00 00]);    %OK
     toepoch([2004 03 20 20 00 00]) toepoch([2004 03 21 24 00 00]);    %OK
     toepoch([2004 03 23 09 00 00]) toepoch([2004 03 24 08 00 00]);    %OK
     toepoch([2004 03 25 15 00 00]) toepoch([2004 03 26 18 00 00]);    %OK
     toepoch([2004 03 28 12 00 00]) toepoch([2004 03 29 01 00 00]);    %OK
     toepoch([2004 03 30 11 00 00]) toepoch([2004 03 31 10 00 00]);    %OK
```

```
%       toepoch([2004 04 01 19 00 00]) toepoch([2004 04 02 18 00 00]);
%       toepoch([2004 04 04 12 00 00]) toepoch([2004 04 05 02 00 00]);
%       toepoch([2004 04 06 18 00 00]) toepoch([2004 04 07 12 00 00]);
%       toepoch([2004 04 09 00 00 00]) toepoch([2004 04 09 23 00 00]);
%       toepoch([2004 04 11 07 00 00]) toepoch([2004 04 12 07 00 00]);
%       toepoch([2004 04 16 01 00 00]) toepoch([2004 04 17 01 00 00]);
%       toepoch([2004 04 18 12 00 00]) toepoch([2004 04 19 10 00 00]);
%       toepoch([2004 04 20 20 00 00]) toepoch([2004 04 21 18 00 00]);
%       toepoch([2004 04 23 05 00 00]) toepoch([2004 04 24 03 00 00]);
%       toepoch([2004 04 25 18 00 00]) toepoch([2004 04 26 05 00 00]);
%       toepoch([2004 04 28 06 00 00]) toepoch([2004 04 28 20 00 00]);
%----------------------------------------2005----------------------------------------%
%       toepoch([2005 01 02 19 00 00]) toepoch([2005 01 02 23 00 00]);
%       toepoch([2005 01 04 17 00 00]) toepoch([2005 01 04 23 00 00]);
%       toepoch([2005 01 07 06 00 00]) toepoch([2005 01 07 22 00 00]);
%       toepoch([2005 01 10 00 00 00]) toepoch([2005 01 07 07 00 00]);
%       toepoch([2005 01 11 18 00 00]) toepoch([2005 01 12 17 00 00]);
%       toepoch([2005 01 14 03 00 00]) toepoch([2005 01 15 01 00 00]);
%       toepoch([2005 01 23 14 00 00]) toepoch([2005 01 24 14 00 00]);
%       toepoch([2005 01 26 01 00 00]) toepoch([2005 01 26 23 00 00]);
%       toepoch([2005 01 28 07 00 00]) toepoch([2005 01 29 10 00 00]);
%       toepoch([2005 01 30 16 00 00]) toepoch([2005 01 31 17 00 00]);
      toepoch([2005 02 02 11 00 00]) toepoch([2005 02 03 03 00 00]);    %OK
      toepoch([2005 02 04 14 00 00]) toepoch([2005 02 05 12 00 00]);    %OK
      toepoch([2005 02 06 18 00 00]) toepoch([2005 02 07 15 00 00]);    %OK
      toepoch([2005 02 11 12 00 00]) toepoch([2005 02 12 14 00 00]);    %OK
      toepoch([2005 02 13 21 00 00]) toepoch([2005 02 14 24 00 00]);    %OK
      toepoch([2005 02 16 06 00 00]) toepoch([2005 02 17 10 00 00]);    %OK
      toepoch([2005 02 18 15 00 00]) toepoch([2005 02 19 18 00 00]);    %OK
      toepoch([2005 02 21 08 00 00]) toepoch([2005 02 22 04 00 00]);    %OK
      toepoch([2005 02 23 09 00 00]) toepoch([2005 02 24 12 00 00]);    %OK
      toepoch([2005 02 25 18 00 00]) toepoch([2005 02 26 20 00 00]);    %OK
      toepoch([2005 02 28 03 00 00]) toepoch([2005 03 01 07 00 00]);    %OK
      toepoch([2005 03 02 12 00 00]) toepoch([2005 03 03 15 00 00]);    %OK
      toepoch([2005 03 04 21 00 00]) toepoch([2005 03 05 24 00 00]);    %OK
      toepoch([2005 03 07 06 00 00]) toepoch([2005 03 07 23 40 00]);    %OK
      toepoch([2005 03 09 15 00 00]) toepoch([2005 03 10 16 00 00])     %s1 no data in entire inte
      toepoch([2005 03 12 18 00 00]) toepoch([2005 03 12 24 00 00]);    %OK
%----------------------------------------2006----------------------------------------%
%       toepoch([2006 02 02 01 00 00]) toepoch([2006 02 02 20 00 00]);    %OK
];



%---------------------------------------------------------------------------
%           Get the measurements in the timentervals from TIMED and Cluster
%           photelectron current.
%---------------------------------------------------------------------------
for i = 1:length(t_interval(:,1))

    disp('once more')

    %----------------------------------------------------------------------
```

```
%   Here we get the ION density from CIS during the timeinterval in the
%   solar wind
%-----------------------------------------------------------------------
[t_c n_c] = get_n(t_interval(i,:),str2double(satellite));

%-----------------------------------------------------------------------
%   The same for the probe to satellite potentials.
%-----------------------------------------------------------------------
db = Mat_DbOpen('db:10');
[t_e V_e] = getPotential(t_interval(i,:),str2double(satellite),db);
Mat_DbClose(db);


%-----------------------------------------------------------------------
%   If there is no measured potential during the time interval skip
%   this annd move on to the next solar wind passage.
%-----------------------------------------------------------------------
if isempty(t_e)
    disp('No efw data')
    continue

%-----------------------------------------------------------------------
%   The same goes for the measurements from CIS. If there is none skip
%   to the next passage.
%-----------------------------------------------------------------------
elseif isempty(t_c)
    disp('No cis data')
    continue

%-----------------------------------------------------------------------
%   If either of the two are limited to less than two measurements then
%   skip to the next passage.
%-----------------------------------------------------------------------
elseif length(n_c)<2 || length(V_e)<2
    disp('Not enough points')
    continue

else

    %-------------------------------------------------------------------
    %   Find all the points without NaNs in CIS data
    %-------------------------------------------------------------------
    t_c = t_c((isnan(n_c)==0));
    n_c = n_c((isnan(n_c)==0));


    %-------------------------------------------------------------------
    %   Check the length ones again
    %-------------------------------------------------------------------
    if length(n_c)<2
        disp('Not enough points')
        continue
    end
```

```
%------------------------------------------------------------------------
%   Get the indecies for photosaturation currents and the F_{10.7}
%   for the time during this solar wind passage
%------------------------------------------------------------------------
ind_I    = find(t_I<(t_interval(i,2)) & t_I>(t_interval(i,1)));
ind_F    = find(t_F<(t_interval(i,2)) & t_F>(t_interval(i,1)));

%------------------------------------------------------------------------
%   Check if there are any currents in the interval
%------------------------------------------------------------------------
if isempty(ind_I)
    disp('No Photoelectron currents in the time interval!!!!!')
    continue

%------------------------------------------------------------------------
%   Check if there are any F_{10.7} measurements during the
%   interval
%------------------------------------------------------------------------
elseif isempty(ind_F)
    disp('No F_{10.7} in the time interval!!!!!!')
    continue

else

    %--------------------------------------------------------------------
    %   Want the potentials to fit the times of the CIS
    %   measurements
    %--------------------------------------------------------------------
    V = interp1(t_e,V_e,t_c,'linear');

    %--------------------------------------------------------------------
    %   Don't want any NaN
    %--------------------------------------------------------------------
    ind = find(isnan(V)==0);
    V = V(ind);
    t_c = t_c(ind);
    n_c = n_c(ind);

    %--------------------------------------------------------------------
    %   Want a polyfit for the loglog so...
    %--------------------------------------------------------------------
    V_log = log10(abs(V));
    n_log = log10(abs(n_c));

    %--------------------------------------------------------------------
    %   Makes the polyfit to the log values of the CIS and EFW data
    %--------------------------------------------------------------------
    p    = polyfit(V_log,n_log,1);
    ln_p = polyval(p,V_log);

    %--------------------------------------------------------------------
```

```
        %   Make non log data of the polyfitted value
        %--------------------------------------------------------------
        n_p = 10.^(ln_p);

        %--------------------------------------------------------------
        %   The different between the measured curve and that of the
        %   least square approximation of the curve.
        %--------------------------------------------------------------
        dn = n_c - n_p;

        %--------------------------------------------------------------
        %   The standard deviation of the difference
        %--------------------------------------------------------------
        s = std(dn)

        %--------------------------------------------------------------
        %   Takes away some of the points that's too far away from the
        %   approximated curve.
        %--------------------------------------------------------------
        ind = find(abs(dn)<5000*s);

        n_c = n_c(ind);
        V_e = 10.^(V_log);
        V_e = V_e(ind);

        %--------------------------------------------------------------
        %   Add all the values to the other timeintervals
        %--------------------------------------------------------------
        t_efw = [t_efw;t_e];
        V_efw = [V_efw;V_e];
        t_cis = [t_cis;t_c(ind)];
        n_cis = [n_cis;n_c];

        %--------------------------------------------------------------
        %   Take the values of the currents and the F_{10.7} during
        %   this passage and make the assumption that there is a little
        %   varriation during this short time. Thus the mean of those
        %   values will serve as the value over the entire passage.
        %--------------------------------------------------------------
        I_temp   = zeros(length(n_c),1);
        I_temp(:)   = mean(I_ph(ind_I));
        I = [I;I_temp];
        F_temp   = zeros(length(n_c),1);
        F_temp(:) = mean(flux(ind_F));
        F = [F;F_temp];
    end
  end
end


%----------------------------------------------------------------------
%   Want the standard deviation in in the three cases
%----------------------------------------------------------------------
```

```
%   No normalization   %
%--------------------%
V_log = log10(abs(V_efw));
            n_log = log10(abs(n_cis));

            p    = polyfit(V_log,n_log,1);
            ln_p = polyval(p,V_log);
            n_p = 10.^(ln_p);

            dn = n_cis - n_p;
            sigma = std(dn)

%   Normalized with the photoelectron current   %
%-----------------------------------------------%
V_log = log10(abs(V_efw));
            n_log = log10(abs(n_cis*mean(I)./I));

            p    = polyfit(V_log,n_log,1);
            ln_p = polyval(p,V_log);
            n_p = 10.^(ln_p);

            dn_I = n_cis*mean(I)./I - n_p;
            sigma_I = std(dn_I)

%   Normalized with F_{10.7}    %
%----------------------------%
V_log = log10(abs(V_efw));
            n_log = log10(abs(n_cis*mean(F)./F));

            p    = polyfit(V_log,n_log,1);
            ln_p = polyval(p,V_log);
            n_p = 10.^(ln_p);

            dn_F = n_cis*mean(F)./F - n_p;
            sigma_F = std(dn_F)



%---------------------------------------------------------------------------
%   Colorful plots
%---------------------------------------------------------------------------
hist2q([log10(V_efw) log10(n_cis)], 50,50,[0.3 1.3],[-0.5 1.6])
set(gca,'CLim',[0 25D3]);
h_bar = colorbar;
set(h_bar,'CLim',[0 25D3]);
ylabel('log_{10}(n)')
xlabel('log_{10}(V_{ps})')

figure;
hist2q([log10(V_efw) log10(n_cis*mean(I)./I)], 50,50,[0.3 1.3],[-0.5 1.6])
set(gca,'CLim',[0 25D3]);
h_bar = colorbar;
```

```
set(h_bar,'CLim',[0 25D3]);
ylabel('log_{10}(n*<I_{ph}>/I_{ph}(t))')
xlabel('log_{10}(V_{ps})')

figure;
hist2q([log10(V_efw) log10(n_cis*mean(F)./F)], 50,50,[0.3 1.3],[-0.5 1.6])
set(gca,'CLim',[0 25D3]);
h_bar = colorbar;
set(h_bar,'CLim',[0 25D3]);
ylabel('log_{10}(n*<F_{10.7}>/F_{10.7}(t))')
xlabel('log_{10}(V_{ps})')

figure;
subplot(2,2,1)
hist(dn,250)
xlim([-20 20])
xlabel('\Delta n = n_{line} - n','Fontsize',30)
title('No Normalization','Fontsize',30)

subplot(2,2,2)
hist(dn_F,250)
xlim([-20 20])
xlabel('\Delta n = n_{line} - n','Fontsize',30)
title('Normalization with F_{10.7}','Fontsize',30)

subplot(2,2,3)
hist(dn_I,250)
xlim([-20 20])
xlabel('\Delta n = n_{line} - n','Fontsize',30)
title('Normalization with I_{ph}','Fontsize',30)
```

A.4 Fluxuncorrection.m

```
clear;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%%  Take the flux and fix it to be the flux at Earth's distance from the
%%  Sun
%%
%%  Erik Winkler, IRF-Uppsala, 2007
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%--------------------------------------------------------------------------
%       Load all the SEE data
%--------------------------------------------------------------------------
[t1, flux1] = getflux([toepoch([2003 01 01 00 00 00]) ...
    toepoch([2003 12 31 24 00 00])],360);
[t2, flux2] = getflux([toepoch([2004 01 01 00 00 00]) ...
    toepoch([2004 12 31 24 00 00])],360);
[t3, flux3] = getflux([toepoch([2005 01 01 00 00 00]) ...
    toepoch([2005 12 31 24 00 00])],360);
[t4, flux4] = getflux([toepoch([2006 01 01 00 00 00]) ...
    toepoch([2006 12 31 24 00 00])],360);

t_flux = [t1;t2;t3;t4];
flux   = [flux1; flux2; flux3; flux4];

%--------------------------------------------------------------------------
%       Load the indecies for corrected data. That so E_{10.7} gets rid of
%       all the ghost points.
%--------------------------------------------------------------------------
load ind1.txt;
load ind2.txt;
load ind3.txt;
load ind4.txt;

indfix = [ind1;
          (ind2+length(t1));
          (ind3+length([t1;t2]));
          (ind4+length([t1;t2;t3]))];

%--------------------------------------------------------------------------
%       Correct the data (et rid of the ghost points)
%--------------------------------------------------------------------------
flux = flux(indfix,:);
t_flux = t_flux(indfix);

%--------------------------------------------------------------------------
%       Difference between Julian day and ISDAT epoch in days
%--------------------------------------------------------------------------
diff = 2440587.5;   %Julian day number for 1-jan-1970 the ISDAT epoch
```

```matlab
%------------------------------------------------------------------------
%       Load the distancies to the Sun for all the years
%------------------------------------------------------------------------
d = 'Earthpos.txt';         % makes a string of the filepath

dist = fopen(d,'r');          % opens the path to the file
D = textscan(dist,'%f %f %*f', 'delimiter', ' ','multipledelimsasone',1);
fclose(dist);                 % close the path to the file

[Julian pos] = deal(D{:});   % Deals D to two vectors

%------------------------------------------------------------------------
%       Change the Julian day to ISDAT epoch times
%------------------------------------------------------------------------
t_pos = (Julian - diff)*24*3600;

%------------------------------------------------------------------------
%       Interpolate the position in the time for the flux from the Sun
%------------------------------------------------------------------------
pos = interp1(t_pos,pos,t_flux);

%------------------------------------------------------------------------
%       Uncorrection of the flux data
%------------------------------------------------------------------------
for i = 1:193
    flux_uncorrected(:,i) = flux(:,i)./(pos.^2);
end

%------------------------------------------------------------------------
%       Saves the flux and its times in a text file
%------------------------------------------------------------------------
% flux_uncorrected = [t_flux flux_uncorrected];
% save Flux_uncorrected.txt flux_uncorrected -ASCII -double
```

A.5 getflux.m

```
function [t,flux] = getflux(time_interval,lamda)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% getflux.m                                                              %
%%                                                                        %
%% [t,flux] = getflux(time_interval,lamda)                                %
%%                                                                        %
%% lamda = 0.5-192.5, returns that particular wavelength                  %
%% lamda = 340 returns F_{10.7}                                           %
%% lamda = 350 returns E_{10.7}                                           %
%% lamda = 360 returns a matrix with all wavelengths                      %
%%                                                                        %
%% time_interval is a vector of length 2, with the times                  %
%% at start and end, given in epochtime, i.e. sek since                   %
%% 1-Jan 1970. The time interval has to be within a year.                 %
%% If the time interval of intrest spans over several years               %
%% the function has to be called several times                            %
%%                                                                        %
%% t is time returned in epoch time.                                      %
%%                                                                        %
%% Erik Winkler, IRF, Uppsala, 2006                                       %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%---------------------------------------------------------------------------
%   F_{10.7} taken directly from file. With some modification to the time
%   columns that, for some reason, are in a strange order
%---------------------------------------------------------------------------
if lamda == 340

    F107 = load('~/Data/F10720020122-06.txt');

    Zero = zeros(length(F107), 1);
    hh = F107(:,6)./100;
    TIME = [F107(:,3) F107(:,4) F107(:,5) hh Zero Zero];
    time = toepoch(TIME);
    tindex = find(time>time_interval(1) & time<time_interval(2));
    t = time(tindex);

    flux = F107(tindex,8);

%---------------------------------------------------------------------------
%   If it's not F_{10.7}, then it must be data from TIMED that's required
%---------------------------------------------------------------------------
else

    year = fromepoch(time_interval(1));

    %-----------------------------------------------------------------------
    % To get the right TIMED data as they are stored in files containing
    % one year only.
```

```
%-----------------------------------------------------------------------

data_str = sprintf('¬/Data/TIMED_L3A_SEE_%d.txt',year(1));
TIMED = load(data_str);


% Rearranging the time collumns (as usal)
TIME = [TIMED(:,3) TIMED(:,2) TIMED(:,1) TIMED(:,4:6)];
time = toepoch(TIME);
tindex = find(time>time_interval(1) & time<time_interval(2));
t = time(tindex);

%-----------------------------------------------------------------------
%    E_{10.7} is the sum of the wavelengths from 0.5nm to 104.5nm
%-----------------------------------------------------------------------
if lamda == 350

    flux = sum(TIMED(tindex,7:111),2);

%-----------------------------------------------------------------------
%    Returns a matrix with all the times and all the fluxies for all
%    wavelengths
%-----------------------------------------------------------------------
elseif lamda == 360

    flux = TIMED(tindex,7:199);

%-----------------------------------------------------------------------
%    Returns the flux for the specified wavelength only.
%-----------------------------------------------------------------------
else

    flux = TIMED(tindex,(lamda+6.5));

end
end
```

A.6 getData.m

```
function [data,vpsm] = getData(time_interval,s,p)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%% Reads the data files containing the data points in the sweeps, i.e the
%% time, the voltage and the current. It also reads the the file containing
%% the position and the satellite potential.
%%   time_interval can be given as
%%                   = [year]
%%                   = [year month]
%%                   = [year startmonth endmonth]
%%                 s = the number of the satellite as a string
%%                 p = probe number as a string
%%  Erik Winkler, IRF-Uppsala, 2006
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Need the satellite nr an probe nr in strings
%as well the time
time = num2str(time_interval(1),4);
%--------------------------------------------------------------------------
%   Make the path to the data file into a string
%--------------------------------------------------------------------------
d = (strcat('¬/Data/dataTM',time,'s',s,p,'.txt'));
pos = (strcat('¬/Data/positionTM',time,'s',s,p,'.txt'));

%--------------------------------------------------------------------------
%   Open the data and position files in read only mode
%--------------------------------------------------------------------------
data = fopen(d,'r');
position = fopen(pos,'r');

%--------------------------------------------------------------------------
%   Error check
%--------------------------------------------------------------------------
if data == -1 || position == -1
    error('Data or Position failed to be found')
end
%--------------------------------------------------------------------------
%   Read all the information needed in the data file and position files
%--------------------------------------------------------------------------
D = textscan(data,'%*f %d %*f %*f %*f %*f %f %f %f', 'delimiter', ' ','multipledelimsasone',1);
P = textscan(position,'%*f %*f %*f %*f %f','delimiter',' ','multipledelimsasone',1);

%--------------------------------------------------------------------------
%   Close the paths
%--------------------------------------------------------------------------
fclose(data);
fclose(position);

%--------------------------------------------------------------------------
```

```
%   Store all the information in correct variables
%------------------------------------------------------------------------
[month epoch_time voltage current] = deal(D{:});
vpsm = deal(P{:});

%------------------------------------------------------------------------
%   Check how long the tiime_interval vector is. Depending on length
%   different things are returned.
%------------------------------------------------------------------------
L = length(time_interval);

switch L
    case 1
        indsweeps = find((voltage(:)==(-118)));
    case 2
        indsweeps = find((voltage(:)==(-118)) & (month(:)==time_interval(2)));
    case 3
        indsweeps = find((voltage(:)==(-118)) & (month(:)≥time_interval(2)) & (month(:)≤tim
    otherwise
        disp('No can do')
end

n = indsweeps(1)/66;  %number of the first sweep in the wanted month
m = length(indsweeps);%the number of sweeps in the wanted month

%------------------------------------------------------------------------
%   Take out the data in the interval and return only that.
%------------------------------------------------------------------------
voltage = voltage((indsweeps(1)-65):(indsweeps(end)),:);
current = current((indsweeps(1)-65):(indsweeps(end)),:);
epoch_time = epoch_time((indsweeps(1)-65):(indsweeps(end)),:);
vpsm = vpsm(n:n+m-1,:);
data = [epoch_time voltage current];
```

A.7 Read_allData.m

```
if bool_one_year_only == 1

    [data2006,vpsm2006] = getData([2006],satellite,probe);
    k2006 = find(data2006(:,2)==(-118));
    [t_I4,I_ph4,I_ph42,I_ph43,coeff4,vpm4,k6,l6,m6,n6] = ...
        Sweeps(data2006,k2006,vpsm2006,alpha,1);
    time_interval = [t_I(1); t_I(end)];

else

    %------------------------------------------------------------------------
    %       Get the sweep data. All bias volatage sweep data.
    %------------------------------------------------------------------------
    [data2003,vpsm2003] = getData([2003],satellite,probe);
    [data2004,vpsm2004] = getData([2004],satellite,probe);
    [data2005,vpsm2005] = getData([2005],satellite,probe);
    [data2006,vpsm2006] = getData([2006],satellite,probe);

    %------------------------------------------------------------------------
    %       Find the indecies between the bias voltage sweeps
    %------------------------------------------------------------------------
    k2003 = find(data2003(:,2)==(-118));
    k2004 = find(data2004(:,2)==(-118));
    k2005 = find(data2005(:,2)==(-118));
    k2006 = find(data2006(:,2)==(-118));

    %---------------------
    %   New corrected alpha
    %---------------------
    %alpha = 0.307692;

    %------------------------------------------------------------------------
    %       Get the current data with Sweeps directly. There is no correction
    %       needed.
    %------------------------------------------------------------------------
    [t_I1,I_ph1,I_ph12,I_ph13,coeff1,vpm1,k3,l3,m3,n3] = ...
        Sweeps(data2003,k2003,vpsm2003,alpha(s,p),0);
    [t_I2,I_ph2,I_ph22,I_ph23,coeff2,vpm2,k4,l4,m4,n4] = ...
        Sweeps(data2004,k2004,vpsm2004,alpha(s,p),0);
    [t_I3,I_ph3,I_ph32,I_ph33,coeff3,vpm3,k5,l5,m5,n5] = ...
        Sweeps(data2005,k2005,vpsm2005,alpha(s,p),0);
    [t_I4,I_ph4,I_ph42,I_ph43,coeff4,vpm4,k6,l6,m6,n6] = ...
        Sweeps(data2006,k2006,vpsm2006,alpha(s,p),0);

    %
     t_I = [t_I1; t_I2; t_I3; t_I4];
     I_ph = [I_ph1; I_ph2; I_ph3; I_ph4];

    %------------------------------------------------------------------------
```

```
    %       Nr of cases where the sweeps have been discarded
    %--------------------------------------------------------------------
    k = k3+k4+k5+k6;
    l = l3+l4+l5+l6;
    m = m3+m4+m5+m6;
    n = n3+n4+n5+n6;

    coeff_all = [coeff1;coeff2;coeff3;coeff4];
%    alpha_all = [alpha1;alpha2;alpha3;alpha3];
    vpsm = [vpm1;vpm2;vpm3;vpm4];
%     c = [c1 c2 c3 c4]';
%     er = (c-1)*1000;
%    k_mean = [k_mean1;k_mean2;k_mean3;k_mean4];
    time_interval1 = [t_I1(1); t_I1(end)];
    time_interval2 = [t_I2(1); t_I2(end)];
    time_interval3 = [t_I3(1); t_I3(end)];
    time_interval4 = [t_I4(1); t_I4(end)];
end
```

A.8 Sweeps.m

```
function [t,I_ph1,I_ph2,I_ph3,coeff,vpmean,kc,lc,mc,nc] = Sweeps(data,k,vpsm,alpha,bool_plot)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%% Returns an approximated photosaturation current and the slope of the
%% same current where it should be zero. The time and the satellte
%% potential (probe to satellite potential) are also returned.
%%
%% Usage:
%%      [t,I_ph,coeff,vpmean
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

t      = [];
ind    = [];
I_ph1  = [];
I_ph2  = [];
I_ph3  = [];
coeff  = [];
vpmean = [];


%--------------------------------------------------------------------------
%   Precreation of variables of the right size but filled with zeros to
%   increase the speed
%--------------------------------------------------------------------------
[V,I]=deal(zeros(66,length(k)-1));

%--------------------------------------------------------------------------
%   One of calibration factors, the other is an input variable, due it is
%   sometimes changed.
%--------------------------------------------------------------------------
beta = 2.12D-3;

%--------------------------------------------------------------------------
%   Calibrate teh probe-satellite potential
%--------------------------------------------------------------------------
vpm = beta*vpsm;

%--------------------------------------------------------------------------
%   Counters for how many times a criterium has failed
%--------------------------------------------------------------------------
kc=0;
lc=0;
mc=0;
nc=0;

for i = 1:(length(k)-1)

    %----------------------------------------------------------------------
    %   Only when the probe to satellite potential is above -30V
```

```
    %--------------------------------------------------------------------------
    if vpm(i) > -30

        V(:,i) = alpha*(data((k(i)-65):(k(i)),2));
        I(:,i) = ((data((k(i)-65):(k(i)),3)*beta-V(:,i))/5D6)*1D9;

        %----------------------------------------------------------------------
        %   All the indecies that have a satellite potential above -30V are
        %   kept
        %----------------------------------------------------------------------
        ind = [ind;i];

    %--------------------------------------------------------------------------
    %   If the satellite potential is less -30V the counter kc is added with
    %   one
    %--------------------------------------------------------------------------
    else
       kc = kc+1;
    end
end

%------------------------------------------------------------------------------
%   Divide the sweep in one uppgoing and one downgoing (with respect
%   the the Vbias)
%------------------------------------------------------------------------------
%V_uppsvep = V(1:floor(end/2));
V_nersvep = V(floor(end/2)+1:end,:);
%I_uppsvep = I(1:floor(end/2));
I_nersvep = I(floor(end/2)+1:end,:);

num_sweeps = length(ind);
vpm = vpm(ind);
for i = 1:num_sweeps

    %--------------------------------------------------------------------------
    %   Finds the indecies where the voltage is lower than probe-satellite
    %   potential minu 2 volt and bellow -100nA
    %--------------------------------------------------------------------------
    undermean_nersvep = find((V_nersvep(:,i)<vpm(i)-2) & (I_nersvep(:,i)< (-100)));

    %--------------------------------------------------------------------------
    %    Goes on only if the number of points left after the previous
    %    weeding
    %--------------------------------------------------------------------------
    if  (length(undermean_nersvep)>5)

        %----------------------------------------------------------------------
        %   Saves only the points that passt the weeding
        %----------------------------------------------------------------------
        V_p_nersvep = V_nersvep(undermean_nersvep,i);
        I_p_nersvep = I_nersvep(undermean_nersvep,i);

        %----------------------------------------------------------------------
```

```
%   Makes a straight line approximation of the ponts left.
%------------------------------------------------------------
I_coeff_nersvep = polyfit(V_p_nersvep,I_p_nersvep,1);
I_eval_nersvep = polyval(I_coeff_nersvep,V_p_nersvep);

%------------------------------------------------------------
%   The point to far away from this straight line are descarded
%------------------------------------------------------------
ind2 = find(abs(I_p_nersvep-I_eval_nersvep)<8);

V_m_nersvep = V_p_nersvep(ind2);
I_m_nersvep = I_p_nersvep(ind2);


%------------------------------------------------------------
%   Hopefully enough points are left after these exclusions
%------------------------------------------------------------
if  length(I_m_nersvep) > 5

    %--------------------------------------------------------
    %   Makes a new straight line approximation of the poits that
    %   are left
    %--------------------------------------------------------
    I_coeffm_nersvep = polyfit(V_m_nersvep,I_m_nersvep,1);
    I_evalm_nersvep = polyval(I_coeffm_nersvep,V(:,i));

    %--------------------------------------------------------
    %   The new lines slope has to have a ablsolute value of less
    %   than 2
    %--------------------------------------------------------
    if  (abs(I_coeffm_nersvep(1))) < 2

        %----------------------------------------------------
        %   Returns all the variables
        %----------------------------------------------------
        t_mean=mean(data(((k(i)+1):(k(i+1)-1)),1));
        t = [t;t_mean];
        coeff = [coeff;I_coeffm_nersvep(1)];
        vpmean = [vpmean;vpm(i)];

        %----------------------------------------------------
        %   The photosaturation current is approximated by the
        %   value of the straight line at V_bias equal to
        %   probe-satellite potential
        %----------------------------------------------------
        I_ph1 = [I_ph1; polyval(I_coeffm_nersvep,vpm(i))];

        %----------------------------------------------------
        %   The photosaturation current is approximated by the
        %   value of the straight lina at V_bias equal to zero
        %----------------------------------------------------
        I_ph2 = [I_ph2; I_coeff_nersvep(2)];
```

```
                %-----------------------------------------------------------
                %   The photosaturation current is approximated by the mean
                %   of the points approximated by the straight line
                %-----------------------------------------------------------
                I_ph3 =[I_ph3; mean(I_m_nersvep)];

            %---------------------------------------------------------------
            %   If the slope value is larger than 2, the counter nc is
            %   added by one
            %---------------------------------------------------------------
            else
                nc = nc+1;
            end

            %---------------------------------------------------------------
            %   Plot every sweep if the bool variable is one
            %---------------------------------------------------------------
            if bool_plot == 1
                plot(V(:,i),I(:,i),'b.',V(:,i),I_evalm_nersvep,'g', ...
                V_m_nersvep,I_m_nersvep,'r.');
                line(0,-250:250);
                line(-40:35,0);
                set(gca,'xlim',[-40 35],'ylim',[-250 250]);
                title(datestr(fromepoch(data(k(i),1))));
                xlabel('Voltage [V]');
                ylabel('Current [nA]');
                pause
            end

        %-------------------------------------------------------------------
        %   If there are less than 5 points left after the points, farther
        %   away from the first straight line approximaton than 8nA, are
        %   deleted the counter adds one
        %-------------------------------------------------------------------
        else
            mc = mc+1;
        end

    %-----------------------------------------------------------------------
    %   If there are less than 5 points left when the points above
    %   -100nA and above V_bias = (vpsm-2)V, the counter adds one
    %-----------------------------------------------------------------------
    else
        lc=lc+1;
    end
end
```

A.9 Mean_ days.m

```
function y = Mean_days(t,F)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   Returns for every time t(i) the mean of the measurements in F
%   within 1.5 days prior and after that time. Done seperately for
%   every column of F.
%
%   Usage:
%           y = Mean_days(t,F)
%
%   where t is the time culumn for the measurements in F. F can be a vector
%   or a matrix.
%
%   Erik Winkler, IRF-Uppsala, 2007
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%--------------------------------------------------------------------------
%   The time before and after the given point should be taken into the mean
%--------------------------------------------------------------------------
dur = 129600; %1.5 dagar in seconds. Gives a 3 day mean

%--------------------------------------------------------------------------
%   Makes a empty vector where the meanvalues will be stored and then
%   returned.
%--------------------------------------------------------------------------
y =[];

%--------------------------------------------------------------------------
%   Makes a mean for every measured point in time
%--------------------------------------------------------------------------
for i = 1:length(t)

    %----------------------------------------------------------------------
    %   If the time of the measurement is at the beginning of the interval
    %----------------------------------------------------------------------
    if t(i) <= (t(1)+dur);
        ind = find(t<=t(i)+dur);

    %----------------------------------------------------------------------
    %   If the time of the measurement is at the end of the interval
    %----------------------------------------------------------------------
    elseif t(i) >= (t(end)-dur)
        ind = find(t>=t(i)-dur);

    %----------------------------------------------------------------------
    %   Take out the indices for all the measurements from 1.5 days prior the
    %   current t(i) to 1.5 days after
    %----------------------------------------------------------------------
    else
        ind = find(t>=(t(i)-dur) & t<=(t(i)+dur));
```

```
    end

    %------------------------------------------------------------------------
    %   Make a mean of the points taken in the interval of 3 days
    %------------------------------------------------------------------------
    temp = sum(F(ind,:))/(length(ind));

    %------------------------------------------------------------------------
    %   Add the mean to the output vector
    %------------------------------------------------------------------------
    y = [y;temp];

    %disp(i)
end
```

A.10 Arends.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%   Arends.m
%
%   Returns Arends' yield function [%] in the unit of [Wm 2/W]. It is for
%   DAG-213
%
%   Erik Winkler, IRF-Uppsala, 2007
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%-----------------------------------------------------------------------
%   Needed constants
%-----------------------------------------------------------------------
e = 1.60217733E-19; % Elementary charge
h = 6.6260755E-34;  % Planck's constant
c = 299792458;      % Speed of light

%-----------------------------------------------------------------------
%   Make it in 5 bin first. 1 bin would be hard to do.
%-----------------------------------------------------------------------
Yield_5bin = 1E-2*[10 10 10 10 10 10 10 9.5 9 8.5 7.6 6 4 2 1 0.1 0.01];

%-----------------------------------------------------------------------
%   The wavelengths of the Yield_5bin and the wanted Yield_1bin
%-----------------------------------------------------------------------
x_5bin = (50:5:130)';
x_1bin = (50:1:130)';

%-----------------------------------------------------------------------
%   Interpolate the 5_bin to 1_bin.
%-----------------------------------------------------------------------
Yield_1bin = interp1(x_5bin,Yield_5bin,x_1bin);



%-----------------------------------------------------------------------
%   Add the rest of the wavelengths with a value of zero
%-----------------------------------------------------------------------
Yield_1bin = [zeros(49,1); Yield_1bin; zeros(63,1)];


%-----------------------------------------------------------------------
%   All the wavelength, 1-193 nm
%-----------------------------------------------------------------------
x_1bin = (1:193)';

%-----------------------------------------------------------------------
%   Calculates a yieldfunction A from Yield_1bin
```

```
%------------------------------------------------------------------------------
A_Arends = (pi*0.04*0.04)*(e/(h*c))*(1E-9*x_1bin).*Yield_1bin; %#ok<NASGU> % Everything in
A_Arends = 1E9*A_Arends(1:193); %want the first 193 elements only and want it in nAm^2/W

%------------------------------------------------------------------------------
%   Want a straight line from zero to the first value in Yield_bin1
%------------------------------------------------------------------------------
k = A_Arends(50)/50;    % the slope of the line
y = k*[1:49]';          % the line

%------------------------------------------------------------------------------
%   Makes the new A_Arends with its new straight line in its beginning
%------------------------------------------------------------------------------
A_Arends = [y;A_Arends(50:end)];
```

A.11 Brace.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%   Brace.m
%
%   Calculates the tranfer function A from Brace's Yield function for Rhenium
%   and/or Molybdenum (not sure, Brace do not say)
%
%   Erik Winkler, IRF-Uppsala, 2007
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Needed constants
e = 1.60217733E-19;
h = 6.6260755E-34;
c = 299792458;

a = zeros(4,1)-9;
b = zeros(8,1)-9;

%By hand taken Yield funtion [electron/photon]
Yield = 1E-6*10.^...
    [a;3.05;3.2;3.8;4.1;4.4;4.7;4.9;5;5.15;5.2;5.2;5.2;5.2;5.1;5;4.95;4.9;4.7;4.5;4.3;4.1;4;3.85;3
x40 = [1:5:200]';
x200 = [1:200]';

Y = interp1(x40,Yield,x200);
Y(1:20)=zeros(20,1);
Y(156:200) = zeros(45,1);

A_Brace = (pi*0.04*0.04)*(e/(h*c))*(1E-9*x200).*Y;
A_Brace = 1E9*A_Brace(1:193);
```

A.12 Samson.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%   Samson.m
%
%   Returns Samson's yield function [%] in the unit of [Wm2/W]. It isa for
%   Aluminium.
%
%   Erik Winkler, IRF-Uppsala, 2007
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Needed constants
e = 1.60217733E-19; % Elementary charge
h = 6.6260755E-34;  % Planck's constant
c = 299792458;      % Speed of light

%Set the beginning and the end in the long vektor to zeros
%Make the thing first in five bins.
a = zeros(length([1:5:25]),1);
b = zeros(length([130:5:200]),1);

%Make it in 5 bin first. 1 bin would be hard to do (not to say tedious).
Yield_5bin = 1E-2*...
    [a; 4.5; 9.5; 11; 11.6; 12.1; 12.9; 14.2; 16.5; 18; 19; 19.5; 19.6; 19; 17.1; 14.5; 11.

x_5bin = [1:5:200]';
x_1bin = [1:1:200]';

%Interpolate the 5_bin to 1_bin.
Yield_1bin = interp1(x_5bin,Yield_5bin,x_1bin);

Yield_1bin(1:29)=zeros(29,1);
Yield_1bin(126:200) = zeros(75,1);

%Calculates a yieldfunction A from Yield_1bin
A_Samson = (pi*0.04*0.04)*(e/(h*c))*(1E-9*x_1bin).*Yield_1bin;
A_Samson = 1E9*A_Samson(1:193); %want the first 193 elements only and want it in nAm^2/W
```

A.13 A_buster.m

```
clear;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%%  Makes a yield function for period and probe (168 of them) and makes a
%%  mean  of all these yield functions
%%
%%  Erik Winkler, IRF-Uppsala 2007
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%---------------------------------------------------------------------------
%   Loading the three days mean and anti-corrected flux [W m_{-2}]
%---------------------------------------------------------------------------
Flux = load('Flux_mean_anti_corrected.txt');

%---------------------------------------------------------------------------
%                               Counter
%---------------------------------------------------------------------------
i_A = 1

%---------------------------------------------------------------------------
%   The corrected alpha
%---------------------------------------------------------------------------
alpha = 0.307266;

%---------------------------------------------------------------------------
%                     Vector with the parts of the year.
%---------------------------------------------------------------------------
Part = [01 03;          % January to Mars
        04 07;          % April to July
        08 12];         % August to December


for y = 2003:2006                 % For all years

    for p = 1                     % For all the time periods in the year

        for s = 1:4               % All the satellites

            if s == 1 | s == 3    % Satellites 1 and 3 have only three probes

                for pp = 2:4      % The three probes

                    %-----------------------------------------------------
                    %       The satellite and probe nr have to be strings
                    %-----------------------------------------------------
                    satellite = num2str(s);
                    probe     = strcat('p',num2str(pp));

                    %-----------------------------------------------------
```

```
%       Load the sweeps in the timeperiod
%--------------------------------------------------------
[data vpsm] = getData([y Part(p,:)],satellite,probe); %

%--------------------------------------------------------
%       Finds the sweeps startpoints
%--------------------------------------------------------
k = find(data(:,2)==-118);

%--------------------------------------------------------
%       Get the photosaturation current
%--------------------------------------------------------
[t_I,I_ph,I_ph2,I_ph3,coeff,vpm,k,l,m,n] = Sweeps(data,k,vpsm,alpha,0);

%--------------------------------------------------------
%       Make a threeday mean of the current
%--------------------------------------------------------
I_meandays = Mean_days(t_I,I_ph);

%--------------------------------------------------------
%       Find the Flux in this time period
%--------------------------------------------------------
ind = find(Flux(:,1)≥t_I(1) & Flux(:,1)≤t_I(end));
F   = Flux(ind,2:194);  % Flux: 1 to 193nm
t   = Flux(ind,1);      % Its time

%--------------------------------------------------------
%       Get the time for the Current within the
%       flux's timeinterval
%--------------------------------------------------------
ind = find(t_I>min(t) & t_I<max(t));
t_I_meandays = t_I(ind);

%--------------------------------------------------------
%       Need at least one value for t_I
%--------------------------------------------------------
L = length(ind)
if L>193
    %----------------------------------------------------
    %       Interpolate the flux to the time of the
    %       current
    %----------------------------------------------------
    F_meandays = interp1(t,F,t_I_meandays);
    I_meandays = -I_meandays(ind);

    %----------------------------------------------------
    %       Calculate the yield function A
    %----------------------------------------------------
    A(:,i_A) = lsqnonneg(F_meandays,I_meandays);

    %----------------------------------------------------
    %       Adds a number to i. The couter of As.
    %----------------------------------------------------
```

```
                i_A = i_A + 1

            %--------------------------------------------------------
            %       If ind is 0, I want to know.
            %--------------------------------------------------------
            else

                disp('Not enough!');

            end
        end

    else    % The other two satellites have a full set of probes

        for pp = 1:4


            %--------------------------------------------------------
            %       The satellite and probe nr have to be strings
            %--------------------------------------------------------
            satellite = num2str(s);
            probe     = strcat('p',num2str(pp));

            %--------------------------------------------------------
            %       Load the sweeps in the timeperiod
            %--------------------------------------------------------
            [data vpsm] = getData([y Part(p,:)],satellite,probe);

            %--------------------------------------------------------
            %       Finds the sweeps startpoints
            %--------------------------------------------------------
            k = find(data(:,2)==-118);

            %--------------------------------------------------------
            %       Get the photosaturation current
            %--------------------------------------------------------
            [t_I,I_ph,I_ph2,I_ph3,coeff,vpm,k,l,m,n] = Sweeps(data,k,vpsm,alpha,0);

            %--------------------------------------------------------
            %       Make a threeday mean of the current
            %--------------------------------------------------------
            I_meandays = Mean_days(t_I,I_ph);

            %--------------------------------------------------------
            %       Find the Flux in this time period
            %--------------------------------------------------------
            ind = find(Flux(:,1)≥t_I(1) & Flux(:,1)≤t_I(end));
            F   = Flux(ind,2:194);  % Flux: 1 to 193nm
            t   = Flux(ind,1);      % Its time

            %--------------------------------------------------------
            %       Get the time for the Current within the
            %       flux's timeinterval
```

```
%-----------------------------------------------------------------
ind = find(t_I>min(t) & t_I<max(t));
t_I_meandays = t_I(ind);

%-----------------------------------------------------------------
%       Need at least one value for t_I
%-----------------------------------------------------------------
L = length(ind)
if L>193
    %-----------------------------------------------------------------
    %       Interpolate the flux to the time of the
    %       current
    %-----------------------------------------------------------------
    F_meandays = interp1(t,F,t_I_meandays);
    I_meandays = -I_meandays(ind);

    %-----------------------------------------------------------------
    %       Calculate the yield function A
    %-----------------------------------------------------------------
    A(:,i_A) = lsqnonneg(F_meandays,I_meandays);

    %-----------------------------------------------------------------
    %       Adds a number to i. The couter of As.
    %-----------------------------------------------------------------
    i_A = i_A + 1

%-----------------------------------------------------------------
%       If ind is less than 193, I want to know.
%-----------------------------------------------------------------
else

    disp('Not enough!');

                    end % if L>0
                end % for pp = 1:4 --Probes
            end % if s == 1 | s == 3 satellite check
        end % for s = 1:4  --satellites
    end % for p = 1:3 --for the periods in the year
end % for y = 2003:2004 --for all the years

%-----------------------------------------------------------------
%       Makes a mean for every wavelength of A, and that becomes the
%       new A function.
%-----------------------------------------------------------------
A_mean = sum(A,2)/i_A;

%-----------------------------------------------------------------
%       Get rid of all contributions from wavelengths shorter than 27nm
%-----------------------------------------------------------------
A_mean_fixed = [zeros(27,1); A_mean(28:193)];

%-----------------------------------------------------------------
%       Calculated current with the use of the Flux and A_mean
```

```
%--------------------------------------------------------------------------
I_calc = Flux(:,2:194)*A_mean;              % Use untempered A_mean
I_calc_fixed = Flux(:,2:194)*A_mean_fixed;  % Use tempered A_mean

%--------------------------------------------------------------------------
%       Get the currents from Cluster with Sweeps
%--------------------------------------------------------------------------
satellite = '2';         % satelite nr for Sweep data
probe     = 'p3';        % probe nr for sweep data
bool_one_year_only = 0; % want the currents from all years
Read_allData;            % Reads all the data for the current probe

%--------------------------------------------------------------------------
%       Three days mean formation of the total current
%--------------------------------------------------------------------------
I_meandays = Mean_days(t_I,I_ph);

%--------------------------------------------------------------------------
%       Interpolate the calculated current to the time of the measured
%       current
%--------------------------------------------------------------------------
I_calc = interp1(Flux(:,1),I_calc,t_I);             % For untempered
I_calc_fixed = interp1(Flux(:,1),I_calc_fixed,t_I); % For tempered

%--------------------------------------------------------------------------
%       Plot the calculated currents and the measures current in two
%       subplots
%--------------------------------------------------------------------------
Plot_MC_CC_A(t_I,-I_meandays,I_calc,A_mean,...
    'A made from the mean of many A caclculated for every probe and three different periods in al

figure(2)
Plot_MC_CC_A(t_I,-I_meandays,I_calc_fixed,A_mean_fixed,...
    'Mean of A where the first 27nm contribution are set to zero')
```

A.14 A_mega.m

```
clear;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%%  Makes a yield function by the use of all currents on the Cluster
%%  probes, flux data from TIMED and the LSQNONNEG MATLAB toutine.
%%
%%  Erik Winkler, IRF-Uppsala,2007
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%--------------------------------------------------------------------------
%   Load the flux already anti corrected and with a three days mean value
%   formation.
%--------------------------------------------------------------------------
load Flux_mean_anti_corrected.txt;

%--------------------------------------------------------------------------
%   Load all currents from all probes
%--------------------------------------------------------------------------
get_allCurrents;
t_I = Currents(:,1);
I_ph = Currents(:,2);

%--------------------------------------------------------------------------
%   Make a three day mean formation of the currents (all currents)
%--------------------------------------------------------------------------
bool_EUVflux = 0;
bool_current = 1;
Mean_days;

t_flux = Flux_mean_anti_corrected(:,1);
flux = Flux_mean_anti_corrected(:,2:194);

%--------------------------------------------------------------------------
%   The Currents and the flux need the same amount of elements, so
%   interpolate the flux to the currents time
%--------------------------------------------------------------------------
ind = find(t_I>min(t_flux) & t_I<max(t_flux));
t_I_meandays = t_I(ind);
F = interp1(t_flux,flux,t_I_meandays);
I = -I_meandays(ind);

%--------------------------------------------------------------------------
%   Make the yield function A
%--------------------------------------------------------------------------
A = lsqnonneg(F,I);

%--------------------------------------------------------------------------
%   Calculates a current from the aquired A
```

A.14 A_mega.m

```
%------------------------------------------------------------------------------
I_approx = F*A;

%------------------------------------------------------------------------------
%   Plot all parameters
%------------------------------------------------------------------------------
Plot_MC_CC_A(t_I_meandays,I,I_approx,A,'all','all')
```

A.15 IF.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%   Takes the photoelectron saturation current and the a calculated current
%   and plot them. Four different yield functions are used to calculate
%   the current. Arends, Brace, Samson and one derived by LSQNONNEG.
%
%   (The same photoelectron saturation current is then plotted against the
%   F_{10.7} index.)
%
%   Erik Winkler, IRF-Uppsala, 2007
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%--------------------------------------------------------------------------
%   Satellite and probe:
%--------------------------------------------------------------------------
satellite = '2';
probe     = 'p3';

%--------------------------------------------------------------------------
%   Need the fine-tuned alpha for this probe
%--------------------------------------------------------------------------
load Alpha_new.txt;
alpha = Alpha_new(str2double(satellite),str2double(probe(2)));
clear Alpha_new;

%--------------------------------------------------------------------------
%   All the years are wanted
%--------------------------------------------------------------------------
bool_one_year_only = 0;     % used in Read_allData.m

%--------------------------------------------------------------------------
%   Read_allData gets the currents from the selected probe
%--------------------------------------------------------------------------
Read_allData;

%--------------------------------------------------------------------------
%   Need the flux of every 1 nm bin and during these 4 years
%--------------------------------------------------------------------------
FF = load('Flux_mean_anti_corrected.txt');

%--------------------------------------------------------------------------
%   The four yield functions:
%--------------------------------------------------------------------------

%   Arends
%%%%%%%%%%
Arends;
A_Arends = A_Arends(1:193);
```

```
%   Brace
%%%%%%%%%
Brace;
A_Brace = A_Brace(1:193);

%   Samson
%%%%%%%%%%
Samson;
A_Samson = A_Samson(1:193);

%   LSQNONNEG
%%%%%%%%%%%%%%%
t_I = t_I(t_I>min(FF(:,1)) & t_I<max(FF(:,1)));
F = interp1(FF(:,1),FF(:,2:194),t_I);
I_ph = -I_ph(t_I>min(FF(:,1)) & t_I<max(FF(:,1)));
A_LSQ = lsqnonneg(F,I_ph);

%--------------------------------------------------------------------------
%   The four calculated currents
%--------------------------------------------------------------------------

%   Arends
%%%%%%%%%%
I_Arends = F*A_Arends;

P_Arends = polyfit(I_Arends,I_ph,1)
I_A = polyval(P_Arends,0:200);

%   Brace
%%%%%%%%%%
I_Brace = F*A_Brace;

P_Brace = polyfit(I_Brace,I_ph,1)
I_B = polyval(P_Brace,0:200);

%   Samson
%%%%%%%%%%
I_Samson = F*A_Samson;

P_Samson = polyfit(I_Samson,I_ph,1)
I_S = polyval(P_Samson,0:200);

%   LSQNONNEG
%%%%%%%%%%
I_LSQ = F*A_LSQ;

P_LSQ = polyfit(I_LSQ,I_ph,1)
I_L = polyval(P_LSQ,0:200);

%--------------------------------------------------------------------------
%   Plot the Measured current I_ph against the four calculated currents
%--------------------------------------------------------------------------
```

```
% %   Arends
% %%%%%%%%%%
subplot(2,2,1)
hist2q([I_Arends I_ph],200,250,[0 200],[0 250])
hold on
plot(I_A,'g-')
% plot(I_Arends,I_ph,'r.')
% xlabel('Calculated Current from TIMED data and Arends'' Yield Function')
% ylabel('Measured Current on Satellite 2, Probe 3')
title('ARENDS')
% ylim([0 250])
% xlim([60 200])

% %   Brace
% %%%%%%%%%%
subplot(2,2,2)
hist2q([I_Brace I_ph],200,250,[0 200],[0 250])
hold on
plot(I_B,'g-')
% plot(I_Brace,I_ph,'g.')
% xlabel('Calculated Current from TIMED data and Brace''s Yield Function')
% ylabel('Measured Current on Satellite 2, Probe 3')
title('BRACE')
% ylim([0 250])
% xlim([60 200])

% %   Samson
% %%%%%%%%%%
subplot(2,2,3)
hist2q([I_Samson I_ph],200,250,[0 200],[0 250])
hold on
plot(I_S,'g-')
% plot(I_Samson,I_ph,'b.')
% xlabel('Calculated Current from TIMED data and Samson''s Yield Function')
% ylabel('Measured Current on Satellite 2, Probe 3')
title('SAMSON')
% ylim([0 250])
% xlim([60 200])

% %   LSQNONNEG
% %%%%%%%%%%
subplot(2,2,4)
hist2q([I_LSQ I_ph],200,250,[0 200],[0 250])
hold on
plot(I_L,'g-')
% plot(I_LSQ,I_ph,'c.')
% xlabel('Calculated Current from TIMED data and Yield Function')
% ylabel('Measured Current on Satellite 2, Probe 3')
title('LSQNONNEG')
% ylim([0 250])
% xlim([60 200])
```

A.16 A_NO2003.m

```
clear;

%% Loading the flux for all the years (anti_corrected flux)%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Flux = load('Flux_mean_anti_corrected.txt');  %Three days mean and anti-corrected flux [W m_{-2}]

%------------------------------------------------------------------------------
%   The corrected alpha
%------------------------------------------------------------------------------
alpha = load('Alpha.txt');

s = 2;
pp = 3;

%-----------------------------------------------------------------
%       The satellite and probe nr have to be strings
%-----------------------------------------------------------------
satellite = num2str(s);
probe     = strcat('p',num2str(pp));

%------------------------------------------------------------------------------
%
%------------------------------------------------------------------------------
I_ph = [];
t = [];

for y = 2003:2006                % For all years

    %-----------------------------------------------------------
    %       Load the sweeps in the timeperiod
    %-----------------------------------------------------------
    [data vpsm] = getData(y,satellite,probe); %

    %-----------------------------------------------------------
    %       Finds the sweeps startpoints
    %-----------------------------------------------------------
    k = find(data(:,2)==-118);

    %-----------------------------------------------------------
    %       Get the photosaturation current
    %-----------------------------------------------------------
    [t_I,I,I_ph2,I_ph3,coeff,vpm,k,l,m,n] = Sweeps(data,k,vpsm,alpha(s,pp),0);

    t = [t;t_I];
    I_ph = [I_ph;I];

end % for y = 2004:2006 --for all the years
```

```
%-------------------------------------------------------------------------
%       Make a threeday mean of the current
%-------------------------------------------------------------------------
I_meandays = Mean_days(t,I_ph);

%-------------------------------------------------------------------------
%       Find the Flux in this time period
%-------------------------------------------------------------------------
t = t(t>min(Flux(:,1)) & t<max(Flux(:,1)));
F = interp1(Flux(:,1),Flux(:,2:194),t);
I_ph = -I_meandays(t>min(Flux(:,1)) & t<max(Flux(:,1)));

%-------------------------------------------------------------------------
%   Calculate the yield function
%-------------------------------------------------------------------------
A = lsqnonneg(F,I_ph);

%-------------------------------------------------------------------------
%   Calculate the Current with the yield function
%-------------------------------------------------------------------------
I_calc = F*A;

%-------------------------------------------------------------------------
%   The mean of the difference between calculated and measured current
%-------------------------------------------------------------------------
mean_LSQ = mean(I_calc-I_ph)

%-------------------------------------------------------------------------
%       Plot the calculated currents and the measures current in two
%       subplots
%-------------------------------------------------------------------------
Plot_MC_CC_A(t,I_ph,I_calc,A,'A made from the years 2004-2006')
```

A.17 Brace_Samson.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%%  Plot the yield functions from Arends, Brace and Samson. In addition it
%%  also plots the calculated currents from these yields and compare them
%%  with the measured current from the Cluster satellite nr 2.
%%
%%  Erik Winkler, IRF-Uppsala, 2007
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%--------------------------------------------------------------------------
%   Load the flux from the Sun
%--------------------------------------------------------------------------
load Flux_mean_anti_corrected.txt;
flux = Flux_mean_anti_corrected(:,2:194);
t_flux = Flux_mean_anti_corrected(:,1);

clear Flux_mean_anti_corrected

%--------------------------------------------------------------------------
%   Load two of the Yield functions given from Brace et al. and
%   Samson. Arends is the new member here
%--------------------------------------------------------------------------
Arends;
Brace;
Samson;

%--------------------------------------------------------------------------
%   Calculate the currents predicted from the two yield function from Brace
%   and Samson
%--------------------------------------------------------------------------
I_A = flux*A_Arends;
I_B = flux*A_Brace;
I_S = flux*1.105375*A_Samson;

%--------------------------------------------------------------------------
%   Loads the currents for satellite 2 probe 3 when it has almost measured
%   currents all teh times
%--------------------------------------------------------------------------
satellite = '2';
probe     = 'p3';
s =2;
p =3;

%------------------Load the Data and for all the years----------------------
bool_one_year_only = 0;

%--------------------The "corrected" alpha is used--------------------------
alpha = load('Alpha.txt');
```

```
%---------------------Read the Data for the satellite---------------------------
Read_allData;

%------------------------------------------------------------------------------
%   3 days mean formation of the current
%------------------------------------------------------------------------------
I_meandays = Mean_days(t_I,I_ph);

%------------------------------------------------------------------------------
%   Interpolate the calculated currents and the flux to the times of the
%   actual measured phot saturation currents on the probe
%------------------------------------------------------------------------------
ind      = find(t_I > min(t_flux) & t_I < max(t_flux));
Flux     = interp1(t_flux,flux,t_I(ind));
I_Brace  = interp1(t_flux,I_B,t_I(ind));
I_Samson = interp1(t_flux,I_S,t_I(ind));
I_Arends = interp1(t_flux,I_A,t_I(ind));

%------------------------------------------------------------------------------
%
mean_Arends = mean(I_Arends+I_meandays(ind));
mean_Brace  = mean(I_Brace +I_meandays(ind));
mean_Samson = mean(I_Samson+I_meandays(ind));
%------------------------------------------------------------------------------
%   Plotting it all
%------------------------------------------------------------------------------
Plot_MC_CC_A(t_I(ind),-I_meandays(ind),I_Brace,A_Brace,'Brace')
figure(2)
Plot_MC_CC_A(t_I(ind),-I_meandays(ind),I_Samson,A_Samson,'Samson')
figure(3)
Plot_MC_CC_A(t_I(ind),-I_meandays(ind),I_Arends,A_Arends,'Arends')
```

A.18 Plot_ MC_ CC_ A.m

```
function Plot_MC_CC_A(t_I_meandays,I,I_approx,A,satellite,probe)

if nargin == 7
    TITLE = strcat(['Satellite nr ',satellite,' and probe nr ',probe(2)]);
else
    TITLE = satellite;
end

subplot(2,1,1)
plot(datenum(fromepoch(t_I_meandays)),I,'b.',datenum(fromepoch(t_I_meandays)),I_approx,'r.')
datetick('x','mmmyy')
ylim([0 250])
title(TITLE)
legend('Measured current','Calculated current')
ylabel('Current [nA]')

subplot(2,1,2)
plot(1:193,A,'b--')
title('Yield function A')
xlabel('\lambda [nm]')
ylabel('A [nAm^2/W]')
```

A.19 Variations.m

```
%I_meandays   Measured currents
%
%   THe sizes of the vectors that will contain all the variations for the
%   flux are created.
%
Flux_pol = zeros(193,2);
s_F = zeros(193,1);

%
%   The current is approximated with a straight line. THen the variation is
%   the difference between the actual measrements and the value on the
%   straight line.
%
I_pol = polyfit(t_I,I_meandays,1);
I_var = I_meandays - polyval(I_pol,t_I);

%
%   The short time variation of teh current as the satandard deviation of
%   the difference I_var divided by the constant term of the straigh line
%   function that approximated the current
%
s_I = std(I_var)/I_pol(2);


%
%   For every wavelength the same is done as for the current.
%
for i = 1:193

    %
    %   Every wavelength's enrgy flux is approximated with a straight line
    %   and then the difference between the measured flux and the straight
    %   line is taken
    %
    Flux_pol(i,:) = polyfit(Flux_mean_anti_corrected(:,1),Flux_mean_anti_corrected(:,i+1),1
    Flux_var = Flux_mean_anti_corrected(:,i+1) - polyval(Flux_pol(i,:),Flux_mean_anti_corre

    %
    %   THe short term variation of every wavelength is then the standard
    %   deviation normalized by the constant term in the straight line
    %   approximation
    %
    s_F(i) = std(Flux_var)/Flux_pol(i,2);

end

%
%   The long term variation is ploted. It is just the slope pf the straight
%   line approximating the current/wavelengths. Normalized with the
```

```
%   constant term of the straight line approximation
%---------------------------------------------------------------------------
subplot(2,1,1)
plot(1:193,-Flux_pol(:,1)./Flux_pol(:,2),'.',[1 193],-I_pol(1)/I_pol(2)*[1 1])

%---------------------------------------------------------------------------
%   Short term variations plotted
%---------------------------------------------------------------------------
subplot(2,1,2)
plot(1:193,s_F, '.', [1 193], -s_I*[1 1])
```

A.20 fil2.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%   Ploting different approximations of the photosaturation currents
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

satellite = '2';
probe     = 'p3';

%-----------------------------------------------------------------------
%   Want all the years
%-----------------------------------------------------------------------
bool_one_year_only = 0;

%-----------------------------------------------------------------------
%   Use the "corrected" calibration factor (nominal=0.307692)
%-----------------------------------------------------------------------
alpha = 0.307266;

%-----------------------------------------------------------------------
%   Reads all the currents. All three photoelectron currents are given in
%   this script
%-----------------------------------------------------------------------
Read_allData;

%-----------------------------------------------------------------------
%   All photoelectron current approximations have the same time vector.
%   Here the total time vector is constructed from the yearly vectors
%-----------------------------------------------------------------------
time_I =   [t_I1; t_I2; t_I3; t_I4];

%-----------------------------------------------------------------------
%   The three different approximations
%-----------------------------------------------------------------------
I_tot1 = [I_ph1;I_ph2;I_ph3;I_ph4];
I_tot2 = [I_ph12;I_ph22;I_ph32;I_ph42];
I_tot3 = [I_ph13;I_ph23;I_ph33;I_ph43];

%-----------------------------------------------------------------------
%   Load the three days mean formation of the flux
%-----------------------------------------------------------------------
%Flux_mean_anti_corrected = load('Flux_uncorrected.txt');

%-----------------------------------------------------------------------
%   Interpolate the flux to have the same time values as teh currents
%-----------------------------------------------------------------------
%ind = find(time_I>min(Flux_mean_anti_corrected(:,1)) & time_I<max(Flux_mean_anti_corrected
%time_I = time_I(ind);
%F = interp1(Flux_mean_anti_corrected(:,1),Flux_mean_anti_corrected(:,2:194),time_I);
```

```
%------------------------------------------------------------------------
%   Take the indecies that we have flux for in the curertns. Multiply with
%   -1 to have a positive current
%------------------------------------------------------------------------
I_tot1 = -I_tot1; %(ind);
I_tot2 = -I_tot2; %(ind);
I_tot3 = -I_tot3; %(ind);

%------------------------------------------------------------------------
%   Plot
%------------------------------------------------------------------------
subplot(3,1,1)
plot(datenum(fromepoch(time_I)),I_tot1,'r.')
ylim([0 300]);
datetick('x','mmmyy')
title('I_{ph} = I_{line}(V_{ps})','Fontsize',30)

subplot(3,1,2)
plot(datenum(fromepoch(time_I)),I_tot2,'r.')
% plot(datenum(fromepoch(t)),flux,'.')
%ylim([0 0.015]);
ylim([0 300]);
datetick('x','mmmyy')
title('I_{ph} = I_{line}(0)','Fontsize',30)

subplot(3,1,3)
plot(datenum(fromepoch(time_I)),I_tot3,'r.')
ylim([0 300]);
datetick('x','mmmyy')
title('I_{ph} = mean(I_{line})','Fontsize',30)

% subplot(3,1,1)
% plot(tmp,-I_1tot,'k.')
% subplot(2,2,2)
% plot(tmp,I_2tot,'r.')
% ylim([-300 0]);
% datetick('x','mm')
% subplot(2,2,3)
% plot(tmp,I_3tot,'g.')
% ylim([-300 0]);
% datetick('x','mm')
% subplot(2,2,4)
% plot(tmp,I_3tot,'g.',tmp,I_2tot,'r.',tmp,I_1tot,'k.')
% ylim([-300 0]);
% datetick('x','mm')


%datenum(fromepoch(time_I)),20000*F,'k.',
```

A.21 E_corre.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%%  Cleaning up the E(t) data from TIMED. Then plots the results
%%
%%  Erik Winkler, IRF Uppsala, 2007
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%-----------------------------------------------------------------------------
%       Load all the SEE data
%-----------------------------------------------------------------------------
[t1, flux1] = getflux([toepoch([2003 01 01 00 00 00]) ...
    toepoch([2003 12 31 24 00 00])],350);
[t2, flux2] = getflux([toepoch([2004 01 01 00 00 00]) ...
    toepoch([2004 12 31 24 00 00])],350);
[t3, flux3] = getflux([toepoch([2005 01 01 00 00 00]) ...
    toepoch([2005 12 31 24 00 00])],350);
[t4, flux4] = getflux([toepoch([2006 01 01 00 00 00]) ...
    toepoch([2006 12 31 24 00 00])],350);

t = [t1;t2;t3;t4];
flux   = [flux1; flux2; flux3; flux4];

%-----------------------------------------------------------------------------
%       Load the indecies for corrected data. That so E_{10.7} gets rid of
%       all the ghost points. The indecies are recieved from the scripts
%       fil2003.m, fil2004.m, fil2005.m and fil2006.m
%-----------------------------------------------------------------------------
load ind1.txt;
load ind2.txt;
load ind3.txt;
load ind4.txt;

indfix = [ind1;
          (ind2+length(t1));
          (ind3+length([t1;t2]));
          (ind4+length([t1;t2;t3]))];

%-----------------------------------------------------------------------------
%   Makes a "cleann" E(t)
%-----------------------------------------------------------------------------
flux_fixed = flux(indfix);
t_fixed = t(indfix);

%-----------------------------------------------------------------------------
%   Plot the results
%-----------------------------------------------------------------------------
subplot(2,1,1)
plot(datenum(fromepoch(t)),flux,'b.')
```

```
datetick('x','mmmyy');
ylim([0 0.02]);
title('Uncorrected flux, E(t)','Fontsize',16)
ylabel('[Wm^{-2}]','Fontsize',16);

subplot(2,1,2)
plot(datenum(fromepoch(t_fixed)),flux_fixed,'r.')
datetick('x','mmmyy');
ylim([0 0.02]);
title('Corrected flux, E(t)','Fontsize',16)
ylabel('[Wm^{-2}]','Fontsize',16);
```

A.22 fil2003.m

```
time_interval = [toepoch([2003 01 01 00 00 00]) toepoch([2003 12 31 24 00 00])];

[t1 flux1] = getflux(time_interval,350);

tmp = fromepoch(t1);

ind1 = [find(tmp(:,2)<12 & flux1>3.1E-3);find(tmp(:,2)==12 & flux1>2.8E-3)];


% subplot(2,1,1)
% plot(datenum(fromepoch(t1(ind1))),flux1(ind1),'b.')
% title('Corrected E_{10.7}')
% ylabel('E_{10.7} flux [Wcm^{-1}\lambda ]')
% datetick('x','mmm')
%
% subplot(2,1,2)
% plot(datenum(fromepoch(t1)),flux1,'r.')
% title('Uncorrected E_{10.7}')
% ylabel('E_{10.7} flux [Wcm^{-1}\lambda ]')
% datetick('x','mmm')
```

A.23 fil2004.m

```
time_interval = [toepoch([2004 01 01 00 00 00]) toepoch([2004 12 31 24 00 00])];

[t2 flux2] = getflux(time_interval,350);

ind2 = find(flux2>2.7E-3);

% subplot(2,1,1)
% plot(datenum(fromepoch(t2(ind2))),flux2(ind2),'b.')
% ylim([1E-3 10E-3])
% title('Corrected E_{10.7}')
% ylabel('E_{10.7} flux [Wcm^{-1}\lambda ]')
% datetick('x','mmm')
%
% subplot(2,1,2)
% plot(datenum(fromepoch(t2)),flux2,'r.')
% ylim([1E-3 10E-3])
% title('Uncorrected E_{10.7}')
% ylabel('E_{10.7} flux [Wcm^{-1}\lambda ]')
% datetick('x','mmm')
```

A.24 fil2005.m

```
time_interval = [toepoch([2005 01 01 00 00 00]) toepoch([2005 12 31 24 00 00])];

[t3 flux3] = getflux(time_interval,350);

tmp = fromepoch(t3);

ind3 = [find(tmp(:,2)<10 & flux3>2.41E-3); find(tmp(:,2)==10 & flux3>2E-3); find(tmp(:,2)>1

% subplot(2,1,1)
% plot(datenum(fromepoch(t3(ind3))),flux3(ind3),'b.')
% ylim([0 0.02])
% title('Corrected E_{10.7}')
% ylabel('E_{10.7} flux [Wcm^{-1}\lambda ]')
% datetick('x','mmm')
%
% subplot(2,1,2)
% plot(datenum(fromepoch(t3)),flux3,'r.')
% ylim([0 0.02])
% title('Uncorrected E_{10.7}')
% ylabel('E_{10.7} flux [Wcm^{-1}\lambda ]')
% datetick('x','mmm')
```

A.25 fil2006.m

```
time_interval = [toepoch([2006 01 01 00 00 00]) toepoch([2006 12 31 24 00 00])];

[t4 flux4] = getflux(time_interval,350);

tmp = fromepoch(t4);

ind4 = [find(tmp(:,2)==1 & flux4>2.58E-3);
        find(tmp(:,2)==2 & flux4>2.48E-3);
        find(tmp(:,2)==3 & flux4>2.496E-3);
        find(tmp(:,2)==4 & flux4>2.625E-3);
        find(tmp(:,2)>4 & tmp(:,2)<8 & flux4>2.5E-3);
        find(tmp(:,2)>7 & flux4>2.44E-3)];

% subplot(2,1,1)
% plot(datenum(fromepoch(t4(ind4))),flux4(ind4),'b.')
% ylim([0 5E-3])
% title('Corrected E_{10.7}')
% ylabel('E_{10.7} flux [Wcm^{-1}\lambda ]')
% datetick('x','mmm')
%
% subplot(2,1,2)
% plot(datenum(fromepoch(t4)),flux4,'r.')
% ylim([0 5E-3])
% title('Uncorrected E_{10.7}')
% ylabel('E_{10.7} flux [Wcm^{-1}\lambda ]')
% datetick('x','mmm')
```

A.26 Alpha_allyears_allprobes.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%%  Get the error in  alpha for every probe and every year
%%
%%  Erik Winkler, IRF-Uppsala, 2007
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

years = 2003:2006;
e = zeros(4,4,4);

%--------------------------------------------------------------------------
%   All satellites
%--------------------------------------------------------------------------
for i = 1:4

    %----------------------------------------------------------------------
    %   All the years
    %----------------------------------------------------------------------
    for y = years
        satellite = num2str(i);

        if i == 1 || i == 3

            %--------------------------------------------------------------
            %   Satellites 1 and 3 have only probes 2-4 operational
            %--------------------------------------------------------------
            for j = 2:4

                probe = strcat('p',num2str(j));

                d = sprintf('Year: %d Satellite: %d Probe: %d',y,i,j);
                disp(d)

                %----------------------
                %   The original alpha
                %----------------------
                alpha = 0.307692;

                %----------------------------------------------------------
                %   Read all the relevant data for the current probe and
                %   current year.
                %----------------------------------------------------------
                [data,vpsm] = getData(y,satellite,probe);

                %----------------------------------------------------------
                %   Find indecies for the bias voltage sweeps
                %----------------------------------------------------------
                k = [find(data(:,2)==(-118))];
```

```
        %--------------------------------------------------------------
        %   Get the photo saturation current and slopes for the
        %   wanted to be flat part of the sweep.
        %--------------------------------------------------------------
        [t_I,I_ph1,I_ph2,I_ph3,coeff,vpm1,k3,l3,m3,n3] = Sweeps(data,k,vpsm,alpha,0);

        %--------------------------------------------------------------
        %   Want to store the number of coeff for every probe
        %--------------------------------------------------------------
        L_coeff1(j,(y-2002)) = length(coeff);

        %--------------------------------------------------------------
        %   Calculates a new "corrected" alpha
        %--------------------------------------------------------------
        alpha = alpha + 5D6*(mean(coeff))*1D-9*alpha;

        %--------------------------------------------------------------
        %   Get the photo saturation current and slopes for the
        %   wanted to be flat part of the sweep with a corrrected alpha.
        %--------------------------------------------------------------
        [t_I,I_ph1,I_ph2,I_ph3,coeff,vpm,k,l,m,n] = Sweeps(data,k,vpsm,alpha,0);

        %--------------------------------------------------------------
        %   Want the new number of coeff
        %--------------------------------------------------------------
        L_coeff2(j,(y-2002)) = length(coeff);

        %--------------------------------------------------------------
        %   Stores the "corrected" alpha
        %--------------------------------------------------------------
        alpha_all(i,j,(y-2002)) = alpha;

        %--------------------------------------------------------------
        %   The error
        %--------------------------------------------------------------
        error(j,(y-2002)) = (0.307692/alpha - 1)*1000;


    end % Probes

else

    %------------------------------------------------------------------
    %   The other 2 satellites have all their probes operational
    %------------------------------------------------------------------
    for j = 1:4

        probe = strcat('p',num2str(j));

        d = sprintf('Year: %d Satellite: %d Probe: %d',y,i,j);
        disp(d)
```

```
%---------------------------------
%   The original alpha
%---------------------------------
alpha = 0.307692;

%------------------------------------------------------------------
%   Read all the relevant data for the current probe and
%   current year.
%------------------------------------------------------------------
[data,vpsm] = getData(y,satellite,probe);

%------------------------------------------------------------------
%   Find indecies for the bias voltage sweeps
%------------------------------------------------------------------
k = [find(data(:,2)==(-118))];

%------------------------------------------------------------------
%   Get the photo saturation current and slopes for the
%   wanted to be flat part of the sweep.
%------------------------------------------------------------------
[t_I,I_ph1,I_ph2,I_ph3,coeff,vpm,k,l,m,n] = Sweeps(data,k,vpsm,alpha,0);

%------------------------------------------------------------------
%   Want to store the number of coeff for every probe
%------------------------------------------------------------------
L_coeff1(j,(y-2002)) = length(coeff);

%------------------------------------------------------------------
%   Calculates a new "corrected" alpha
%------------------------------------------------------------------
alpha = alpha + 5D6*(mean(coeff))*1D-9*alpha;

%------------------------------------------------------------------
%   Get the photo saturation current and slopes for the
%   wanted to be flat part of the sweep with a corrrected alpha.
%------------------------------------------------------------------
[t_I,I_ph1,I_ph2,I_ph3,coeff,vpm1,k3,l3,m3,n3] = Sweeps(data,k,vpsm,alpha,0

%------------------------------------------------------------------
%   Want the new number of coeff
%------------------------------------------------------------------
L_coeff2(j,(y-2002)) = length(coeff);

%------------------------------------------------------------------
%   Stores the "corrected" alpha
%------------------------------------------------------------------
alpha_all(i,j,(y-2002)) = alpha;

%------------------------------------------------------------------
%   The error
%------------------------------------------------------------------
error(j,(y-2002)) = (0.307692/alpha - 1)*1000;
```

```
            end % Probes
        end % Satellites 1 & 3 vs 2 & 4
    end % Year

    disp(error)
    figure

    if i == 2 || i ==4
        subplot(2,2,1)
        plot(years,error(1,:),'bo')
        title(sprintf('Satelite: %d, Probe: %d',i,1))
        axis([2003 2006 -5 5])
        xlabel('Time')
        ylabel('Error in promille')
        t = sprintf('Number of values from the different years: 2003: %d 2004: %d 2005: %d 2006: 
        text(0.1,0.1,t)
    end

    subplot(2,2,2)
    plot(years,error(2,:),'bo')
    title(sprintf('Satelite: %d, Probe: %d',i,2))
    axis([2003 2006 -5 5])
    xlabel('Time')
    ylabel('Error in promille')
    t = sprintf('Number of values from the different years: 2003: %d 2004: %d 2005: %d 2006: %d',
    text(0.1,0.1,t)

    subplot(2,2,3)
    plot(years,error(3,:),'bo')
    title(sprintf('Satelite: %d, Probe: %d',i,3))
    axis([2003 2006 -5 5])
    xlabel('Time')
    ylabel('Error in promille')
    t = sprintf('Number of values from the different years: 2003: %d 2004: %d 2005: %d 2006: %d',
    text(0.1,0.1,t)

    subplot(2,2,4)
    plot(years,error(4,:),'bo')
    title(sprintf('Satelite: %d, Probe: %d',i,4))
    axis([2003 2006 -5 5])
    xlabel('Time')
    ylabel('Error in promille')
    t = sprintf('Number of values from the different years: 2003: %d 2004: %d 2005: %d 2006: %d',
    text(0.1,0.1,t)

end % Satellite
```

A.27 Alpha_probes.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%%  Calculates new calibration factors for the Cluster probes
%%  individually.
%%
%%  Erik Winkler, IRF-Uppsala,2007
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
alpha_probe = zeros(4);

%-----------------------------------------------------------------------------
%   Read_allData is wanted to read data from all years
%-----------------------------------------------------------------------------
bool_one_year_only = 0;


for s = 1:4
    satellite = num2str(s);

    if s == 1 || s == 3
        for p = 2:4

            %-----------------------------------------------------------------
            %   Original alpha
            %-----------------------------------------------------------------
            alpha = 0.307692;

            probe = strcat('p',num2str(p));
            Read_allData;
            k_old = [coeff1;coeff2;coeff3;coeff4];
            k_mean_old(s,p) = mean(k_old);
            alpha = alpha + 5D6*k_mean_old(s,p)*1D-9*alpha;

            alpha_probe(s,p) = alpha;

            Read_allData;
            k_new = [coeff1;coeff2;coeff3;coeff4];
            k_mean_new(s,p) = mean(k_new);

            TITLE = sprintf('Satellite %d, probe %d',s, p);
            figure
            subplot(2,1,1)
            hist(k_old,50)
            title(TITLE)
            subplot(2,1,2)
            hist(k_new,50)

        end
    else
```

```
        for p = 1:4

            %--------------------------------------------------------------
            %   Original alpha
            %-------------------------------------------------------------
            alpha = 0.307692;

            probe = strcat('p',num2str(p));
            Read_allData;
            k_old = [coeff1;coeff2;coeff3;coeff4];
            k_mean_old(s,p) = mean(k_old);
            alpha = alpha + 5D6*k_mean_old(s,p)*1D-9*alpha;

            alpha_probe(s,p) = alpha;

            Read_allData;
            k_new = [coeff1;coeff2;coeff3;coeff4];
            k_mean_new(s,p) = mean(k_new);

            TITLE = sprintf('Satellite %d, probe %d',s, p);
            figure
            subplot(2,1,1)
            hist(k_old,50)
            title(TITLE)
            subplot(2,1,2)
            hist(k_new,50)
        end
    end
end
```

Appendix B

Yield Functions

B.1 Yield for DAG-213 and Rhenium

Table B.1: The Yield functions for two different materials. The yield is presented in two different units, Y is in electrons per incident photon, A is in nA per Wm^{-2}.

Yield Functions				
	DAG-213		Rhenium	
nm	Y	A	Y	A
1	0.00000	0.00000	0.00000	0.00000
2	0.00000	0.00000	0.00000	0.00000
3	0.00000	0.00000	0.00000	0.00000
4	0.00000	0.00000	0.00000	0.00000
5	0.00000	0.00000	0.00000	0.00000
6	0.00000	0.00000	0.00000	0.00000
7	0.00000	0.00000	0.00000	0.00000
8	0.00000	0.00000	0.00000	0.00000
9	0.00000	0.00000	0.00000	0.00000
10	0.00000	0.00000	0.00000	0.00000
11	0.00000	0.00000	0.00000	0.00000
12	0.00000	0.00000	0.00000	0.00000
13	0.00000	0.00000	0.00000	0.00000
14	0.00000	0.00000	0.00000	0.00000
15	0.00000	0.00000	0.00000	0.00000
16	0.00000	0.00000	0.00000	0.00000
17	0.00000	0.00000	0.00000	0.00000
18	0.00000	0.00000	0.00000	0.00000
19	0.00000	0.00000	0.00000	0.00000
20	0.00000	0.00000	0.00000	0.00000
21	0.00000	0.00000	0.00112	95.52623
22	0.00000	0.00000	0.00121	108.33205
23	0.00000	0.00000	0.00131	121.88849
24	0.00000	0.00000	0.00140	136.19557
25	0.00000	0.00000	0.00149	151.25328
26	0.00000	0.00000	0.00158	167.06163
27	0.00000	0.00000	0.00253	276.92255
28	0.00000	0.00000	0.00347	394.44537
29	0.00000	0.00000	0.00442	519.63007
30	0.00000	0.00000	0.00536	652.47666
31	0.00000	0.00000	0.00631	792.98514
32	0.00000	0.00000	0.00757	981.50275
33	0.00000	0.00000	0.00882	1180.20394
34	0.00000	0.00000	0.01008	1389.08873

Yield Functions				
	DAG-213		Rhenium	
nm	Y	A	Y	A
35	0.00000	0.00000	0.01133	1608.15710
36	0.00000	0.00000	0.01259	1837.40907
37	0.00000	0.00000	0.01510	2264.34848
38	0.00000	0.00000	0.01760	2711.60682
39	0.00000	0.00000	0.02011	3179.18409
40	0.00000	0.00000	0.02261	3667.08030
41	0.00000	0.00000	0.02512	4175.29544
42	0.00000	0.00000	0.03012	5128.50556
43	0.00000	0.00000	0.03512	6122.25728
44	0.00000	0.00000	0.04012	7156.55060
45	0.00000	0.00000	0.04512	8231.38552
46	0.00000	0.00000	0.05012	9346.76205
47	0.00000	0.00000	0.05598	10667.09297
48	0.00000	0.00000	0.06184	12034.96179
49	0.00000	0.00000	0.06771	13450.36849
50	0.10000	20270.91530	0.07357	14913.31309
51	0.10000	20676.33361	0.07943	16423.79558
52	0.10000	21081.75192	0.08355	17613.01501
53	0.10000	21487.17022	0.08766	18835.58768
54	0.10000	21892.58853	0.09177	20091.51360
55	0.10000	22298.00683	0.09589	21380.79275
56	0.10000	22703.42514	0.10000	22703.42514
57	0.10000	23108.84345	0.10825	25015.49655
58	0.10000	23514.26175	0.11650	27394.46807
59	0.10000	23919.68006	0.12475	29840.33971
60	0.10000	24325.09836	0.13300	32353.11145
61	0.10000	24730.51667	0.14125	34932.78329
62	0.10000	25135.93498	0.14470	36371.91594
63	0.10000	25541.35328	0.14815	37838.99905
64	0.10000	25946.77159	0.15160	39334.03261
65	0.10000	26352.18989	0.15504	40857.01662
66	0.10000	26757.60820	0.15849	42407.95108
67	0.10000	27163.02651	0.15849	43050.49580
68	0.10000	27568.44481	0.15849	43693.04051
69	0.10000	27973.86312	0.15849	44335.58522
70	0.10000	28379.28142	0.15849	44978.12994
71	0.10000	28784.69973	0.15849	45620.67465
72	0.10000	29190.11804	0.15849	46263.21936
73	0.10000	29595.53634	0.15849	46905.76408
74	0.10000	30000.95465	0.15849	47548.30879

Yield Functions				
	DAG-213		Rhenium	
nm	Y	A	Y	A
75	0.10000	30406.37296	0.15849	48190.85350
76	0.10000	30811.79126	0.15849	48833.39822
77	0.10000	31217.20957	0.15849	49475.94293
78	0.10000	31622.62787	0.15849	50118.48764
79	0.10000	32028.04618	0.15849	50761.03236
80	0.10000	32433.46449	0.15849	51403.57707
81	0.09900	32510.49396	0.15849	52046.12178
82	0.09800	32579.41508	0.15197	50521.35229
83	0.09700	32640.22782	0.14545	48943.72147
84	0.09600	32692.93220	0.13893	47313.22933
85	0.09500	32737.52822	0.13241	45629.87587
86	0.09400	32774.01586	0.12589	43893.66108
87	0.09300	32802.39514	0.12071	42577.52052
88	0.09200	32822.66606	0.11554	41219.39071
89	0.09100	32834.82861	0.11036	39819.27167
90	0.09000	32838.88279	0.10518	38377.16338
91	0.08900	32834.82861	0.10000	36893.06585
92	0.08800	32822.66606	0.09783	36487.24913
93	0.08700	32802.39514	0.09565	36063.79686
94	0.08600	32774.01586	0.09348	35622.70904
95	0.08500	32737.52822	0.09130	35163.98568
96	0.08320	32381.57094	0.08913	34687.62678
97	0.08140	32011.01861	0.08719	34286.64800
98	0.07960	31625.87122	0.08525	33869.95153
99	0.07780	31226.12877	0.08331	33437.53737
100	0.07600	30811.79126	0.08137	32989.40550
101	0.07280	29809.59721	0.07943	32525.55595
102	0.06960	28781.45638	0.07357	30423.15870
103	0.06640	27727.36879	0.06771	28273.22356
104	0.06320	26647.33442	0.06184	26075.75053
105	0.06000	25541.35328	0.05598	23830.73961
106	0.05600	24065.63065	0.05012	21538.19080
107	0.05200	22557.47455	0.04642	20136.68187
108	0.04800	21016.88499	0.04272	18705.17855
109	0.04400	19443.86196	0.03902	17243.68085
110	0.04000	17838.40547	0.03532	15752.18876
111	0.03600	16200.51551	0.03162	14230.70230
112	0.03200	14530.19209	0.02929	13299.09700
113	0.02800	12827.43520	0.02695	12348.56653
114	0.02400	11092.24485	0.02462	11379.11089

Yield Functions				
	DAG-213		Rhenium	
nm	Y	A	Y	A
115	0.02000	9324.62104	0.02229	10390.73006
116	0.01800	8465.13423	0.01995	9383.42407
117	0.01600	7589.43069	0.01848	8765.76842
118	0.01400	6697.51042	0.01701	8136.17179
119	0.01200	5789.37341	0.01553	7494.63418
120	0.01000	4865.01967	0.01406	6841.15560
121	0.00820	4022.56043	0.01259	6175.73604
122	0.00640	3165.50613	0.01207	5970.64081
123	0.00460	2293.85678	0.01155	5761.34666
124	0.00280	1407.61236	0.01104	5547.85358
125	0.00100	506.77288	0.01052	5330.16158
126	0.00082	418.87819	0.01000	5108.27066
127	0.00064	329.52400	0.00942	4848.06601
128	0.00046	238.71030	0.00883	4583.12520
129	0.00028	146.43709	0.00825	4313.44822
130	0.00010	52.70438	0.00766	4039.03507
131	0.00000	0.00000	0.00708	3759.88577
132	0.00000	0.00000	0.00637	3410.62805
133	0.00000	0.00000	0.00567	3055.64369
134	0.00000	0.00000	0.00496	2694.93267
135	0.00000	0.00000	0.00425	2328.49499
136	0.00000	0.00000	0.00355	1956.33067
137	0.00000	0.00000	0.00334	1855.60430
138	0.00000	0.00000	0.00313	1753.19748
139	0.00000	0.00000	0.00293	1649.11020
140	0.00000	0.00000	0.00272	1543.34247
141	0.00000	0.00000	0.00251	1435.89429
142	0.00000	0.00000	0.00237	1361.61130
143	0.00000	0.00000	0.00222	1286.13865
144	0.00000	0.00000	0.00207	1209.47632
145	0.00000	0.00000	0.00193	1131.62433
146	0.00000	0.00000	0.00178	1052.58266
147	0.00000	0.00000	0.00167	997.88878
148	0.00000	0.00000	0.00157	942.35267
149	0.00000	0.00000	0.00147	885.97435
150	0.00000	0.00000	0.00136	828.75380
151	0.00000	0.00000	0.00126	770.69103
152	0.00000	0.00000	0.00121	743.88312
153	0.00000	0.00000	0.00116	716.65532
154	0.00000	0.00000	0.00110	689.00762

Yield Functions				
	DAG-213		Rhenium	
nm	Y	A	Y	A
155	0.00000	0.00000	0.00105	660.94004
156	0.00000	0.00000	0.00000	0.00000
157	0.00000	0.00000	0.00000	0.00000
158	0.00000	0.00000	0.00000	0.00000
159	0.00000	0.00000	0.00000	0.00000
160	0.00000	0.00000	0.00000	0.00000
161	0.00000	0.00000	0.00000	0.00000
162	0.00000	0.00000	0.00000	0.00000
163	0.00000	0.00000	0.00000	0.00000
164	0.00000	0.00000	0.00000	0.00000
165	0.00000	0.00000	0.00000	0.00000
166	0.00000	0.00000	0.00000	0.00000
167	0.00000	0.00000	0.00000	0.00000
168	0.00000	0.00000	0.00000	0.00000
169	0.00000	0.00000	0.00000	0.00000
170	0.00000	0.00000	0.00000	0.00000
171	0.00000	0.00000	0.00000	0.00000
172	0.00000	0.00000	0.00000	0.00000
173	0.00000	0.00000	0.00000	0.00000
174	0.00000	0.00000	0.00000	0.00000
175	0.00000	0.00000	0.00000	0.00000
176	0.00000	0.00000	0.00000	0.00000
177	0.00000	0.00000	0.00000	0.00000
178	0.00000	0.00000	0.00000	0.00000
179	0.00000	0.00000	0.00000	0.00000
180	0.00000	0.00000	0.00000	0.00000
181	0.00000	0.00000	0.00000	0.00000
182	0.00000	0.00000	0.00000	0.00000
183	0.00000	0.00000	0.00000	0.00000
184	0.00000	0.00000	0.00000	0.00000
185	0.00000	0.00000	0.00000	0.00000
186	0.00000	0.00000	0.00000	0.00000
187	0.00000	0.00000	0.00000	0.00000
188	0.00000	0.00000	0.00000	0.00000
189	0.00000	0.00000	0.00000	0.00000
190	0.00000	0.00000	0.00000	0.00000
191	0.00000	0.00000	0.00000	0.00000
192	0.00000	0.00000	0.00000	0.00000
193	0.00000	0.00000	0.00000	0.00000

B.2 Yield for Aluminium and 1.10573*Aluminium

Table B.2: The Yield functions for two different materials. The yield is presented in two different units, Y is in electrons per incident photon, A is in nA per Wm^{-2}.

Yield Functions				
	Aluminium		1.105375*Aluminium	
nm	Y	A	Y	A
1	0.00000	0.00000	0.00000	0.00000
2	0.00000	0.00000	0.00000	0.00000
3	0.00000	0.00000	0.00000	0.00000
4	0.00000	0.00000	0.00000	0.00000
5	0.00000	0.00000	0.00000	0.00000
6	0.00000	0.00000	0.00000	0.00000
7	0.00000	0.00000	0.00000	0.00000
8	0.00000	0.00000	0.00000	0.00000
9	0.00000	0.00000	0.00000	0.00000
10	0.00000	0.00000	0.00000	0.00000
11	0.00000	0.00000	0.00000	0.00000
12	0.00000	0.00000	0.00000	0.00000
13	0.00000	0.00000	0.00000	0.00000
14	0.00000	0.00000	0.00000	0.00000
15	0.00000	0.00000	0.00000	0.00000
16	0.00000	0.00000	0.00000	0.00000
17	0.00000	0.00000	0.00000	0.00000
18	0.00000	0.00000	0.00000	0.00000
19	0.00000	0.00000	0.00000	0.00000
20	0.00000	0.00000	0.00000	0.00000
21	0.00000	0.00000	0.00000	0.00000
22	0.00000	0.00000	0.00000	0.00000
23	0.00000	0.00000	0.00000	0.00000
24	0.00000	0.00000	0.00000	0.00000
25	0.00000	0.00000	0.00000	0.00000
26	0.00000	0.00000	0.00000	0.00000
27	0.00000	0.00000	0.00000	0.00000
28	0.00000	0.00000	0.00000	0.00000
29	0.00000	0.00000	0.00000	0.00000
30	0.08500	10338.16680	0.09396	11427.55113
31	0.09500	11939.56911	0.10501	13197.70121

Yield Functions				
	Aluminium		1.105375*Aluminium	
nm	Y	A	Y	A
32	0.09800	12713.91808	0.10833	14053.64720
33	0.10100	13512.59214	0.11164	14936.48154
34	0.10400	14335.59130	0.11496	15846.20424
35	0.10700	15182.91556	0.11828	16782.81529
36	0.11000	16054.56492	0.12159	17746.31470
37	0.11120	16680.53078	0.12292	18438.24172
38	0.11240	17316.22669	0.12424	19140.92408
39	0.11360	17961.65263	0.12557	19854.36178
40	0.11480	18616.80861	0.12690	20578.55482
41	0.11600	19281.69464	0.12822	21313.50321
42	0.11700	19922.25556	0.12933	22021.56324
43	0.11800	20570.92485	0.13043	22738.58606
44	0.11900	21227.70251	0.13154	23464.57166
45	0.12000	21892.58853	0.13264	24199.52004
46	0.12100	22565.58292	0.13375	24943.43122
47	0.12260	23361.01363	0.13552	25822.68044
48	0.12420	24169.41773	0.13729	26716.27013
49	0.12580	24990.79522	0.13906	27624.20027
50	0.12740	25825.14610	0.14082	28546.47087
51	0.12900	26672.47036	0.14259	29483.08192
52	0.13160	27743.58552	0.14547	30667.06585
53	0.13420	28835.78244	0.14834	31874.35301
54	0.13680	29949.06111	0.15122	33104.94342
55	0.13940	31083.42153	0.15409	34358.83707
56	0.14200	32238.86370	0.15696	35636.03396
57	0.14660	33877.56449	0.16205	37447.41285
58	0.15120	35553.56377	0.16713	39300.02055
59	0.15580	37266.86153	0.17222	41193.85706
60	0.16040	39017.45778	0.17730	43128.92239
61	0.16500	40805.35251	0.18239	45105.21653
62	0.16800	42228.37076	0.18570	46678.18533
63	0.17100	43675.71411	0.18902	48278.04249
64	0.17400	45147.38256	0.19234	49904.78800
65	0.17700	46643.37611	0.19565	51558.42187
66	0.18000	48163.69476	0.19897	53238.94410
67	0.18200	49436.70824	0.20118	54646.10137
68	0.18400	50725.93846	0.20339	56071.18422
69	0.18600	52031.38540	0.20560	57514.19264
70	0.18800	53353.04908	0.20781	58975.12663
71	0.19000	54690.92949	0.21002	60453.98618

Yield Functions				
	Aluminium		1.105375*Aluminium	
nm	Y	A	Y	A
72	0.19100	55753.12545	0.21113	61628.11104
73	0.19200	56823.42978	0.21223	62811.19869
74	0.19300	57901.84247	0.21334	64003.24912
75	0.19400	58988.36353	0.21444	65204.26234
76	0.19500	60082.99296	0.21555	66414.23834
77	0.19520	60935.99308	0.21577	67357.12335
78	0.19540	61790.61486	0.21599	68301.80091
79	0.19560	62646.85833	0.21621	69248.27102
80	0.19580	63504.72346	0.21643	70196.53370
81	0.19600	64364.21027	0.21665	71146.58893
82	0.19480	64759.89854	0.21533	71583.97285
83	0.19360	65145.85677	0.21400	72010.60142
84	0.19240	65522.08495	0.21267	72426.47466
85	0.19120	65888.58310	0.21135	72831.59255
86	0.19000	66245.35121	0.21002	73225.95510
87	0.18620	65675.33307	0.20582	72595.87130
88	0.18240	65074.50314	0.20162	71931.72891
89	0.17860	64442.86142	0.19742	71233.52795
90	0.17480	63780.40791	0.19322	70501.26839
91	0.17100	63087.14261	0.18902	69734.95026
92	0.16580	61840.88673	0.18327	68357.37017
93	0.16060	60552.46736	0.17752	66933.18361
94	0.15540	59221.88448	0.17178	65462.39055
95	0.15020	57849.13809	0.16603	63944.99102
96	0.14500	56434.22820	0.16028	62380.98500
97	0.13820	54347.94560	0.15276	60074.86037
98	0.13140	52206.52611	0.14525	57707.78880
99	0.12460	50009.96973	0.13773	55279.77029
100	0.11780	47758.27645	0.13021	52790.80484
101	0.11100	45451.44629	0.12270	50240.89245
102	0.10480	43337.59525	0.11584	47904.29434
103	0.09860	41173.47233	0.10899	45512.12697
104	0.09240	38959.07754	0.10214	43064.39034
105	0.08620	36694.41088	0.09528	40561.08443
106	0.08000	34379.47235	0.08843	38002.20925
107	0.07500	32534.81906	0.08290	35963.17562
108	0.07000	30649.62394	0.07738	33879.32806
109	0.06500	28723.88698	0.07185	31750.66658
110	0.06000	26757.60820	0.06632	29577.19116
111	0.05500	24750.78759	0.06080	27358.90183

Yield Functions				
	Aluminium		1.105375*Aluminium	
nm	Y	A	Y	A
112	0.05120	23248.30734	0.05660	25698.09773
113	0.04740	21715.01531	0.05239	24003.23505
114	0.04360	20150.91148	0.04819	22274.31378
115	0.03980	18555.99587	0.04399	20511.33393
116	0.03600	16930.26846	0.03979	18714.29550
117	0.03280	15558.33291	0.03626	17197.79224
118	0.02960	14160.45059	0.03272	15652.60808
119	0.02640	12736.62150	0.02918	14078.74299
120	0.02320	11286.84564	0.02564	12476.19700
121	0.02000	9811.12301	0.02211	10844.97009
122	0.01600	7913.76533	0.01769	8747.67836
123	0.01200	5983.97420	0.01326	6614.53548
124	0.00800	4021.74960	0.00884	4445.54146
125	0.00400	2027.09153	0.00442	2240.69630
126	0.00000	0.00000	0.00000	0.00000
127	0.00000	0.00000	0.00000	0.00000
128	0.00000	0.00000	0.00000	0.00000
129	0.00000	0.00000	0.00000	0.00000
130	0.00000	0.00000	0.00000	0.00000
131	0.00000	0.00000	0.00000	0.00000
132	0.00000	0.00000	0.00000	0.00000
133	0.00000	0.00000	0.00000	0.00000
134	0.00000	0.00000	0.00000	0.00000
135	0.00000	0.00000	0.00000	0.00000
136	0.00000	0.00000	0.00000	0.00000
137	0.00000	0.00000	0.00000	0.00000
138	0.00000	0.00000	0.00000	0.00000
139	0.00000	0.00000	0.00000	0.00000
140	0.00000	0.00000	0.00000	0.00000
141	0.00000	0.00000	0.00000	0.00000
142	0.00000	0.00000	0.00000	0.00000
143	0.00000	0.00000	0.00000	0.00000
144	0.00000	0.00000	0.00000	0.00000
145	0.00000	0.00000	0.00000	0.00000
146	0.00000	0.00000	0.00000	0.00000
147	0.00000	0.00000	0.00000	0.00000
148	0.00000	0.00000	0.00000	0.00000
149	0.00000	0.00000	0.00000	0.00000
150	0.00000	0.00000	0.00000	0.00000
151	0.00000	0.00000	0.00000	0.00000

Yield Functions				
	Aluminium		1.105375*Aluminium	
nm	Y	A	Y	A
152	0.00000	0.00000	0.00000	0.00000
153	0.00000	0.00000	0.00000	0.00000
154	0.00000	0.00000	0.00000	0.00000
155	0.00000	0.00000	0.00000	0.00000
156	0.00000	0.00000	0.00000	0.00000
157	0.00000	0.00000	0.00000	0.00000
158	0.00000	0.00000	0.00000	0.00000
159	0.00000	0.00000	0.00000	0.00000
160	0.00000	0.00000	0.00000	0.00000
161	0.00000	0.00000	0.00000	0.00000
162	0.00000	0.00000	0.00000	0.00000
163	0.00000	0.00000	0.00000	0.00000
164	0.00000	0.00000	0.00000	0.00000
165	0.00000	0.00000	0.00000	0.00000
166	0.00000	0.00000	0.00000	0.00000
167	0.00000	0.00000	0.00000	0.00000
168	0.00000	0.00000	0.00000	0.00000
169	0.00000	0.00000	0.00000	0.00000
170	0.00000	0.00000	0.00000	0.00000
171	0.00000	0.00000	0.00000	0.00000
172	0.00000	0.00000	0.00000	0.00000
173	0.00000	0.00000	0.00000	0.00000
174	0.00000	0.00000	0.00000	0.00000
175	0.00000	0.00000	0.00000	0.00000
176	0.00000	0.00000	0.00000	0.00000
177	0.00000	0.00000	0.00000	0.00000
178	0.00000	0.00000	0.00000	0.00000
179	0.00000	0.00000	0.00000	0.00000
180	0.00000	0.00000	0.00000	0.00000
181	0.00000	0.00000	0.00000	0.00000
182	0.00000	0.00000	0.00000	0.00000
183	0.00000	0.00000	0.00000	0.00000
184	0.00000	0.00000	0.00000	0.00000
185	0.00000	0.00000	0.00000	0.00000
186	0.00000	0.00000	0.00000	0.00000
187	0.00000	0.00000	0.00000	0.00000
188	0.00000	0.00000	0.00000	0.00000
189	0.00000	0.00000	0.00000	0.00000
190	0.00000	0.00000	0.00000	0.00000
191	0.00000	0.00000	0.00000	0.00000

Yield Functions				
	Aluminium		1.105375*Aluminium	
nm	Y	A	Y	A
192	0.00000	0.00000	0.00000	0.00000
193	0.00000	0.00000	0.00000	0.00000