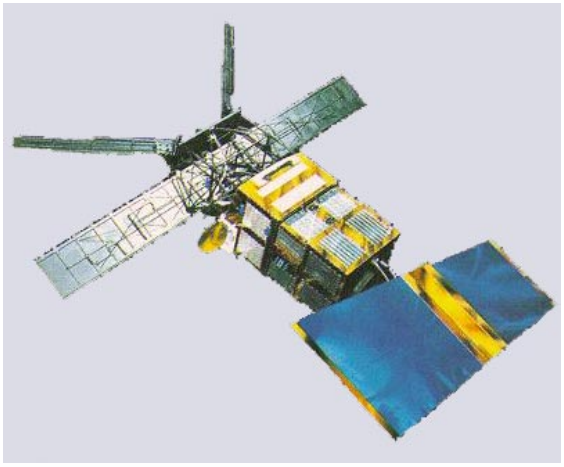
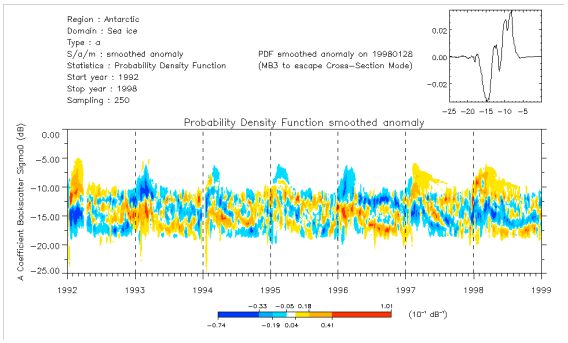
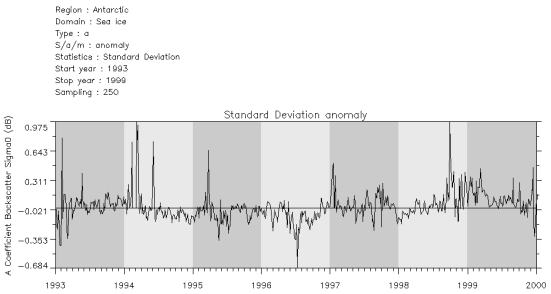
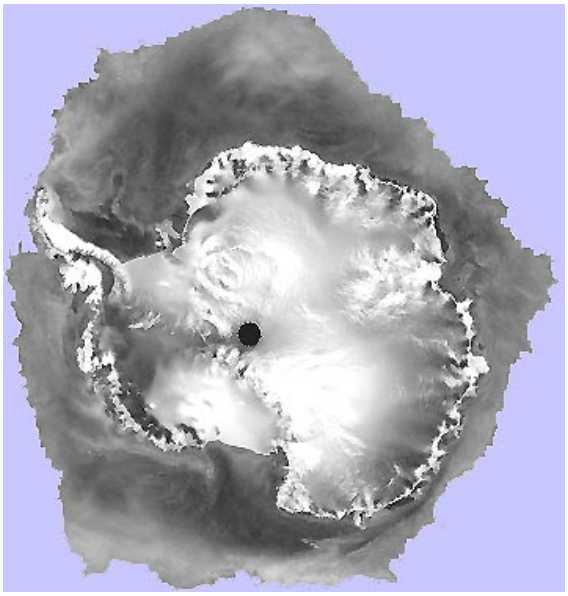




TOPICS

- Tool for Polar Ice Characteristics Study -





- Author -

Pierre Mercier

- Supervisors -

Dr. Mark Drinkwater and **Dr. Benjamin Holt**

FOREWORD

This report presents the results of a five month end of studies internship I made at the Jet Propulsion Laboratory in the Ocean Science Department from April to September 2000 as an engineering student from the Ecole Nationale Supérieure de l'Aéronautique et de l'Espace (ENSAE/SUPAERO), the French Aeronautics and Space Engineering School.

Dr Mark R. DRINKWATER initiated the project, and supervised my work in collaboration with Dr Benjamin HOLT.

Prepared by : Pierre Mercier

Date :

Supervised by : Dr. Mark Drinkwater
Dr. Benjamin Holt

Date :

Table of contents

INTRODUCTION

For the past 25 years, Earth observation satellites have provided scientists with ever more accurate data, allowing them to understand better the physical phenomena at stake in Earth climate. During the past decade the need for monitoring changes in our environment on short and mid terms has kept increasing as on-orbit remote sensing revealed the real extent of human activity impact on our planet's weather.

The aim of our project was to build a data analysis tool for investigating climate fingerprints in the longest available microwave data set: that of European Space Agency's ERS1 and ERS2 satellites. Here we are focusing on polar regions for two reasons: first polar ice melting or forming plays a major rule in global climate regulation, especially regarding heat transfer cycles. Then, Arctic and Antarctic regions are the most sensitive to changes; indeed, sea-ice and ice sheets are in a delicate balance with today's climate, and are constantly changing in characteristics in adjustment to short-term oscillations in atmospheric and oceanic conditions.

For the first time, we are able to analyze 8 years of continuous C-band radar data to study such changes. However, a tool is required which facilitates quick looks at the data and gives a useful perspective for change detection. TOPICS, which stands for TOol for Polar Ice Characteristics Study, has been developed to meet those expectations. This new data analysis tool computes statistics (among which the Probability Density Function) of the normalized cross section backscatter of polar ice (σ_0) from the ERS scatterometer data set (1992-2000). From those signals, TOPICS computes mean cycles and anomaly plots on which the "signatures", and thus the contributions of the different types of ice can be distinguished. Qualitative interpretations of climate trends or cycles are then derived in terms of changes in these contributions. The main asset of TOPICS is that custom regions of investigation can be defined, allowing precise localization of phenomena occurring in the evolution of polar ice characteristics.

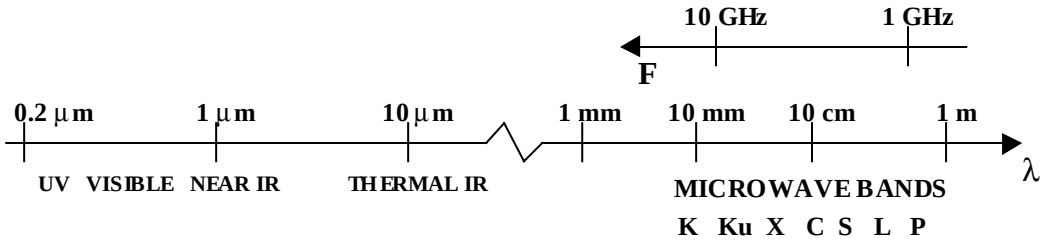
FIRST PART: General Considerations on Active Microwave Observation of Polar Ice

Interest of active microwave observations systems

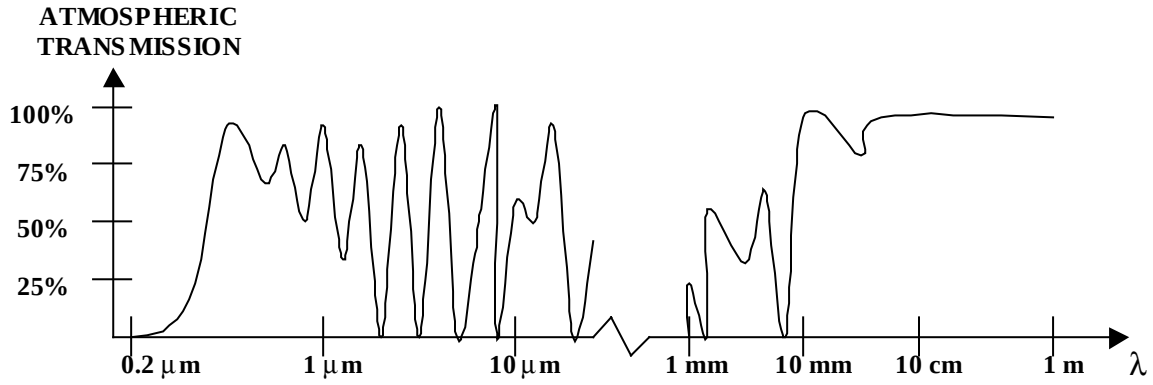
In order to improve our knowledge and understanding of global environment, space observation has been used for many years now, with more and more success. The main drawback of optic observation systems is to be hindered during the night or by cloudy meteorological conditions. On the other hand, active micromave remote sensing allows data collection whatever the weather or the sunlight conditions. Indeed, in some wavelengths, radar waves can penetrate clouds, and as an active device, radar does not need the sunlight to illuminate the observed scene. Polar regions being in total darkness six months a year and very cloudy the rest of the year, it comes as no surprise that active microwave remote sensing is the most efficient way to gather scientific data on this locations.

Atmospheric transmission

Radar waves have wavelengths greater than 10mm and lower than 1m, which corresponds to 10GHz and 1GHz frequencies, respectively.



Now, any electromagnetic wave propagating through the Earth's atmosphere loses power. However, the attenuation depends a lot on the signal wavelength, as illustrated below.



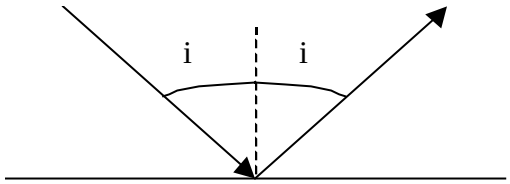
In order to be efficient, an active sensing instrument needs to maximize the power of the returned signal ; thus its wavelength must be inside a « transmission window », where atmospheric

transmission is the closest possible to 100%. As can be seen on the figure, radar waves naturally fall into such a window, which explains why most active (and other) remote sensors use them.

Backscatter

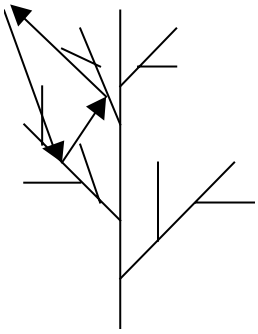
When a radar signal interacts with ground surfaces, it can either be reflected, scattered, absorbed, or transmitted (and refracted).

Reflection is often due to a material's high dielectric constant, usually meaning a high water content. Very smooth surfaces also encourage reflection. If the imaged surface is a smooth lake, for example, the incoming radar will be reflected off the lake according to Snell's law – at the same angle as the incidence angle. Such reflections return very little signal strength back to the satellite, resulting in a dark region on an image.



reflection on a smooth surface

Reflections can, however, bounce again off other objects and thereby be redirected back toward the spacecraft, resulting, in a stronger return signal. This process is called **volumetric scattering** (in contrast to surface scattering).

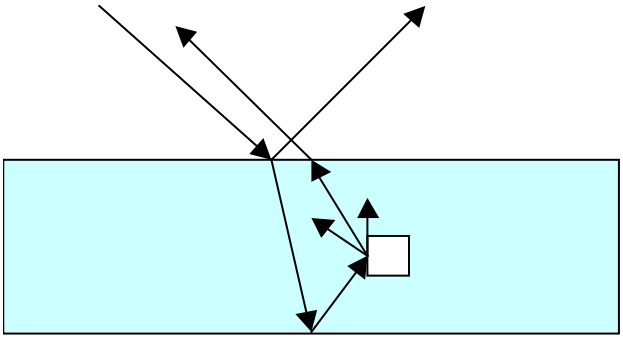


volumetric scattering

When a radar signal is transmitted through the surface, it will be refracted depending upon the density of the substance according to the index of refraction. The index of refraction equals the velocity of an electromagnetic wave in a vacuum divided by the velocity of an electromagnetic wave in the particular substance and is used to determine how a signal's characteristics will be altered when it passes into a different material. In our case, this means that the radar signal will travel more slowly in the surface material than they did in air, and therefore they will appear to refract toward the surface's

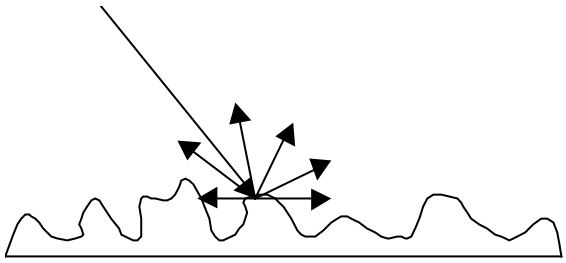
normal. The transmitted, refracted signal can either be absorbed by the material or have another surface interaction when hits a region with different properties.

So we could have a combination of transmission, reflection and scattering, for both surfaces and volumes.



combination of all properties of backscatter

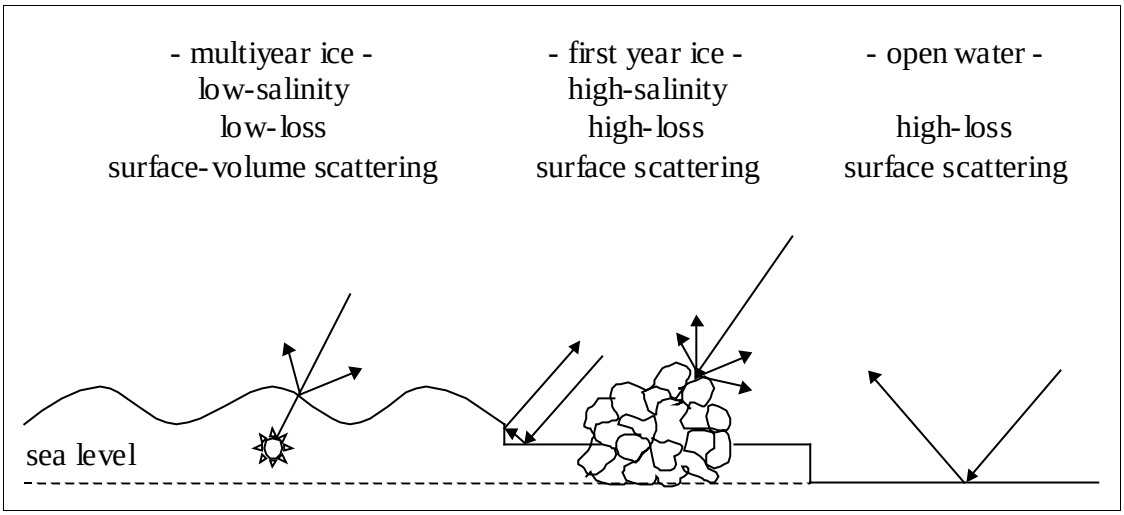
Surface scattering of incoming radar signal, determined by surface roughness, is often the main factor influencing radar return. Very smooth surfaces will reflect according to Snell's law, like the reflections on smooth water already discussed. Surface variations on the order magnitude of the radar wavelength will scatter radar as specified by the *Rayleigh criterion* : h.cos(look angle), the height of the surface variation in the direction of the incoming radar, must be less than $1/8^{th}$ the value of the radar's wavelength to be considered "smooth" (the Rayleigh parameter h represents the ground's height variation, while the look angle is measured from nadir to the radar's direction of travel). The rougher the surface, the stronger the radar backscatter return.



scattering on a rough surface

As far as sea ice is concerned, backscatter depends on many different parameters : it is influenced by the structure and composition of the ice (dielectric constant, ice thickness, temperature, salinity, thickness and moisture of the snow layer, brine pockets contents, air bubble contents, etc. – all these parameters depending on the ice type and age) and by the type of the electromagnetic wave (direction, polarization, wavelength, incidence angle, etc.).

The figure below shows how the different type of sea ice scatter electromagnetic wave back :



backscatter for multiyear ice, first year ice and smooth open water

Spaceborn scatterometry

Spaceborn scatterometry consists in measuring the backscatter signal when scanning the Earth's surface from a satellite, with an active (usually microwave) sensor called scatterometer.

High Frequency polarized energy pulses are generated inside the satellite and sent towards the Earth's surface thanks to an antenna. When the pulses hit the surface, it causes part of the incident signal to be returned to the source. An important fact is that the ratio of backscattered energy to incident energy depends on characteristics of the illuminated surface. After it is received by the satellite's antenna , the backscattered signal is converted into digital form for data processing. From what precedes, the key quantity to be derived is normalized radar cross-section backscatter σ^0; it is computed using the radar equation :

$$\mathbf{Pr=Pt\times[G^2\lambda^2/(4\pi)^3]\times\sigma^0 A/R^4}$$

where Pr=power of backscattered signal
Pt=power of emitted pulses
G=antenna gain at emission
λ=signal wavelength
A=antenna area
R=distance between source and targeted surface

Measuring σ^0 is useful because different types of surfaces usually have different σ^0 signatures ; roughly speaking, knowing σ^0 is often equivalent to knowing the nature of the surface. It turns out that the ranges of σ^0 are quite well known for the different types of ice (see for example table below) : it is thus possible to determine the type of ice by measuring σ^0. This is precisely what

scatterometers do, and that is why the raw data they collect are really useful to scientists who study polar ice.

Ice Type	Backscatter Range (dB)
Icebergs	$-6.0 \leq \sigma^0 \leq 0.0$
Multiyear/Pancakes	$-11.0 \leq \sigma^0 \leq -6.0$
Rough First Year	$-14.0 \leq \sigma^0 \leq -11.0$
Smooth First year	$-20.0 \leq \sigma^0 \leq -14.0$
Nilas	$-32.0 \leq \sigma^0 \leq -20.0$

summary of Winter C-band EScat backscatter ranges at 40º incidence of the main categories of sea ice in the Weddel Sea

As a last remark, $\sigma^0(\theta)$ strongly depends on surface wind over ocean (as it creates waves which change surface roughness). This fact is used by scatterometers to derive ocean near-surface wind speed and direction; such wind scatterometers have three antennae to measure σ^0 from different azimuth angles, in order to determine wind direction without ambiguity.

SECOND PART : The EScat Data Set

The ERS satellites

The European Remote Sensing Satellites (ERS) are two Earth observation satellites designed to study processes of the Earth's oceans and land. ERS-1 was successfully launched in July 1991, and was fully operated between the beginning of 1992 and the first third of 1996; ERS-2, launched in April 1995, took over at that time. It is still working now (September 2000). The two satellites are mostly identical (hardware/software/orbit), which provides continuous data sets.

artist view of ERS2

Both satellites are on a near-circular sun-synchronous polar orbit, which gives them visibility of all areas of the Earth as the planet rotates beneath their orbits. Moreover, the orbital plane will always maintain its position relative to the Sun, crossing the equator with the descending node at about 10:30 am local time. The main orbital parameters for the two satellites are specified in table 1.

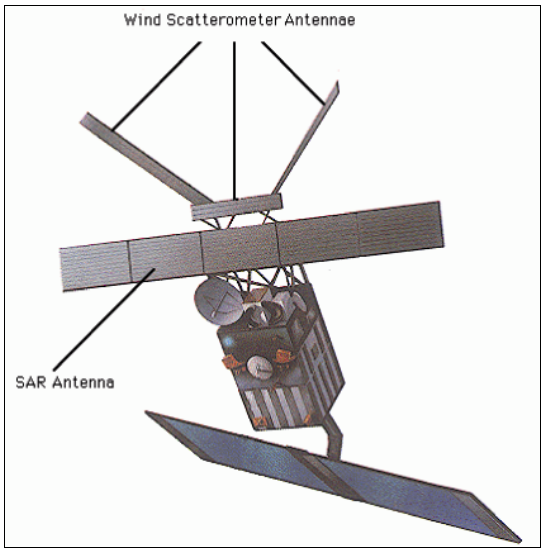
	ERS1/2
Inclination	98.516°
Altitude	785 km
Eccentricity	~ 0
Orbit repeat cycle	35 days
Period	1h 40min 23s

orbital parameters of ERS1/2

EScat Wind Scatterometer

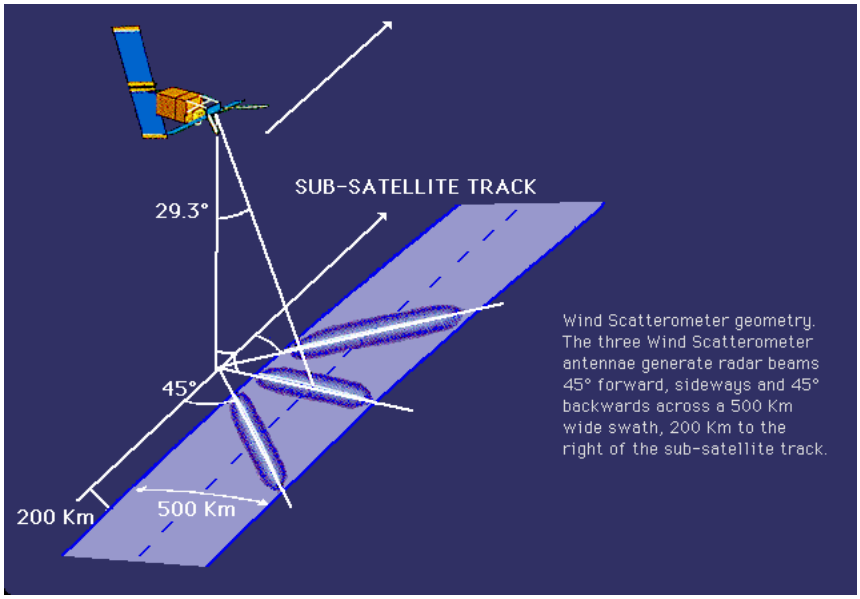
The main sensor onboard ERS satellites is the Active Microwave Instrument (AMI), which has a Synthetic Aperture Radar (SAR) than can operate in three different modes :

1- **SAR Image Mode**, for the acquisition of wide swath images over the oceans, polar ice caps and land areas.
2- **SAR Wave Mode**, yielding 5km×5km images at regular intervals along track for the derivation of length and direction of ocean waves.
3- **Wind Scatterometer Mode**, using three separate antennae for measurements of sea-surface wind speed and direction. When operated in this mode, the instrument is called the EScat instrument.



main sensors on ERS1/2

The Escat instrument measures VV-polarized σ^0 at 5.3 GHz (λ=5.66cm, C band) and at various azimuth and incidence angles along a 500 km-wide swath, as described in the figure below :



Wind Scatterometer Geometry

EScat Operation

The only measurement conflict is that the scatterometer subsystem of the AMI is periodically switched off during competing SAR-mode operation. This results in some scatterometer data gaps in the vicinity of operable Antarctic SAR receiving stations (most significantly over the tip of the Antarctic Peninsula and at locations within 200 km of Mc Murdo Station).

However, the key advantage of the ERS scatterometer is that it operates whenever the SAR is switched off, continuously retrieving information without the necessity of a local receiving station.. Its wider swath provides more frequent and broader incidence angle ($20° \leq \theta \leq 55°$) in a given location. This low-bit-rate data source is essential to fill in areas of sparse temporal and spatial SAR coverage of polar regions.

The SIRF algorithm

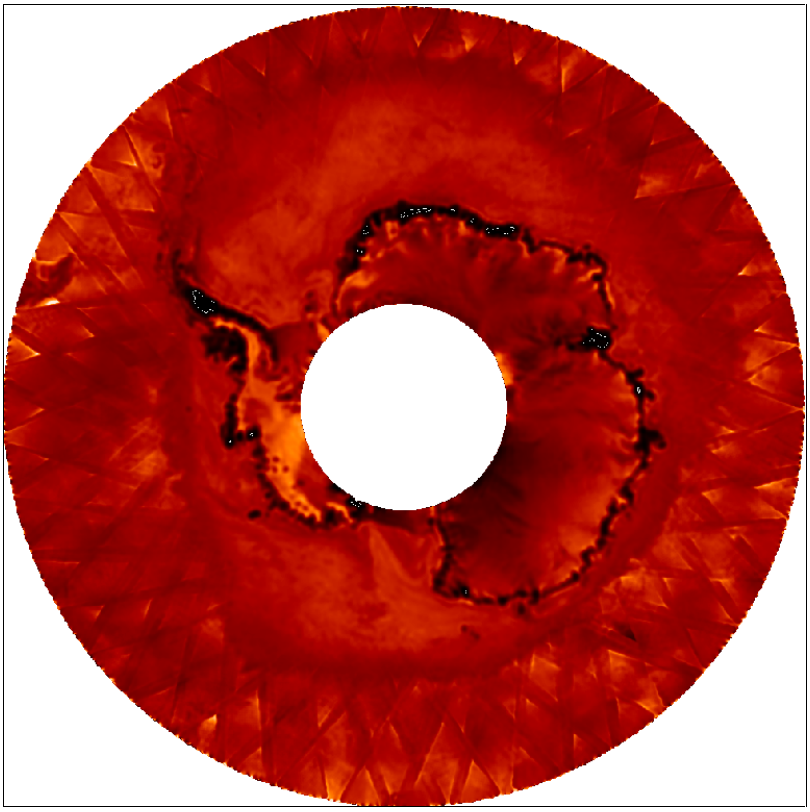
Until recently, the intrinsic low resolution of the raw measurement data was the main reason why scatterometer data have not been used to fill spatial or temporal gaps in SAR coverage of polar regions. However, recent developments in scatterometer image processing allow to produce weekly average backscatter maps of polar regions at a resolution higher than conventional alternatives such as SSM/I passive microwave radiometers.

The method consists in processing the data with a scatterometer image reconstruction technique and a filtering algorithm called Scatterometer Image Reconstruction Filter (SIRF). Briefly, over glacial ice, σ^0 (in dB) can be approximated as a linear function of the incidence angle θ in the range $20° \leq \theta \leq 55°$:

$$\sigma^0(\theta) = A + B\,(\theta - 40)$$

where the coefficients A and B depend on surface characteristics, polarization and azimuth angle. A is expected value at 40°(mid_swath) incidence, and B describes the variation of σ^0 with θ. The SIRF algorithm uses multiple, overlapping measurements of σ^0 and postprocessing to create enhanced resolution, temporally averaged image data of A and B. As a tradeoff between resolution and temporal averaging, six-day averaged images are derived at three-day intervals, with an estimated resolution of ~25km, resampled onto a polar-stereographic grid.

The main characteristic of these relatively high incidence angle images is that sea-ice typically demonstrates smoothly varying isotropic returns in direct contrast to the highly variable returns of open ocean equatorward of the sea-ice margin. Variability in the backscatter from wind-generated waves is large on time scales of several days, and the advantage of the EScat is that it views the same location from a number of incidence and azimuthal angles during the period of image integration. This results in a very neat separation between highly anisotropic or azimuthally-dependent variable wind waves and relatively isotropic and azimuthally independent scattering from ice or land ; thus, ocean appears covered with diamond shaped boxes, corresponding to orbits tracks overlapping whereas land or ice-covered areas seem very smooth (see example below).



SIRF-enhanced resolution Escat image of Antarctica on 10/04/1995 (σ^0 A coefficient). Values vary between -32dB (very light red) and 0dB (dark red).

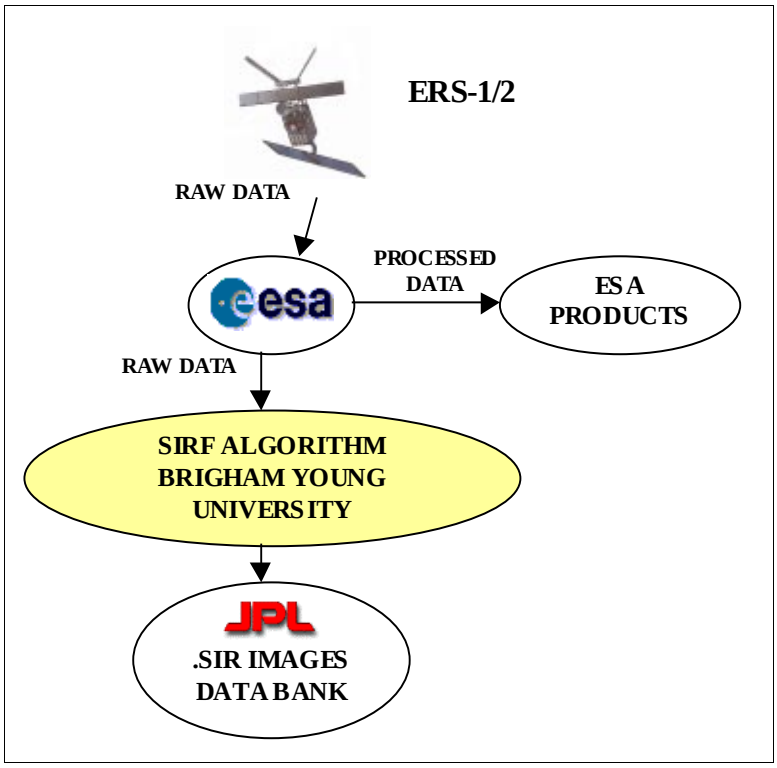
The hole that appear in the middle is caused by the satellite's orbit not passing over the poles; besides, it is smaller on Arctic images because the scatterometer looks on the right side of the orbit track, which diminish or increase the satellite's non-visibility area.

The polar regions mapped by ERS-1/2 have the following extent:

	Latitudes	Longitudes
Arctic	≥ 60°	-180° to 180°
Antarctic	≤ -52°	-180° to 180°

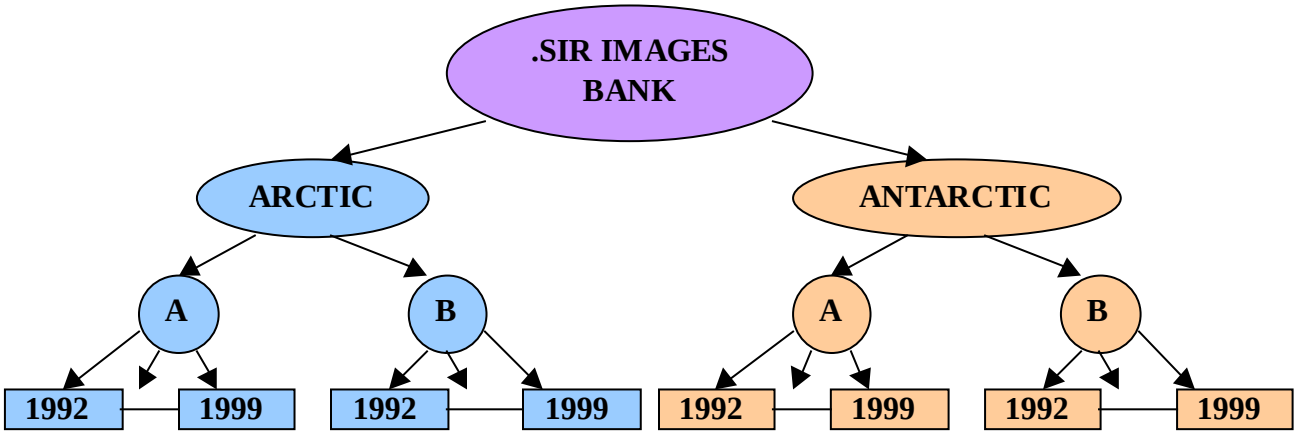
JPL .sir images bank

The Jet Propulsion Laboratory has collected so far 8 years of continuous SIRF-enhanced resolution EScat images, for both A and B σ^0 coefficients. Raw data are provided by ESA to the Brigham Young University which process them using SIRF algorithm. Produced .SIR images are then sent to JPL, which stores them in its data bank.



EScat data end-to-end path

It is precisely this bank of images that TOPICS uses to derive statistical time series. Its structure (on 08/30/2000) is described by the figure below :



structure of JPL .sir images bank

THIRD PART : DATA PROCESSING

TOPICS' statement of purpose

TOPICS aim at providing a data analysis tool for investigating climate changes fingerprints in polar ice variability. Using 8 years of continuous scatterometer data from ERS-1/2, it computes statistical time series of the normalized cross-section backscatter σ^0 over polar regions, and produces graphical outputs which helps monitoring and interpreting the evolution of polar ice statistical characteristics.

Regions of investigations

There are two primary regions of investigation : those which correspond to ERS polar coverages :

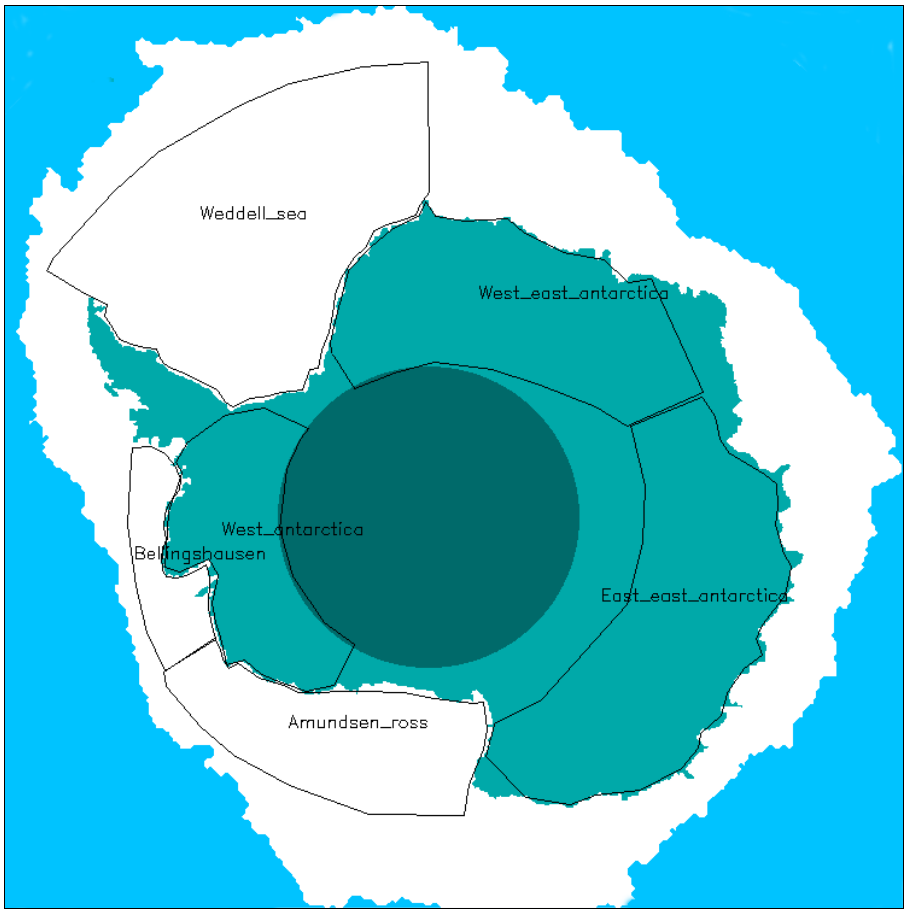
Arctic: locations with latitudes greater than 60°
Antarctic: locations with latitudes lower than -52°

In each of these two areas, custom regions can also be defined. TOPICS contains 16 such custom regions by default :

- In Arctic: Beaufort sea, Canada Alaska, Central Arctic, Chukchi sea, East Beaufort sea, Eastern Arctic, Greenland, Laptev sea, Russia, Siberia.

map of Arctic regions

- In Antarctic: Amundsen-Ross sea, Bellingshausen sea, East East Antarctica, Weddel sea, West Antarctic, West East Antarctic.



map of Antarctic regions

Each new custom region is defined by the user, which draws its contour on a map. Thanks to this contour, TOPICS makes a “region mask” which is used for statistics computation over the region.

Ice domains

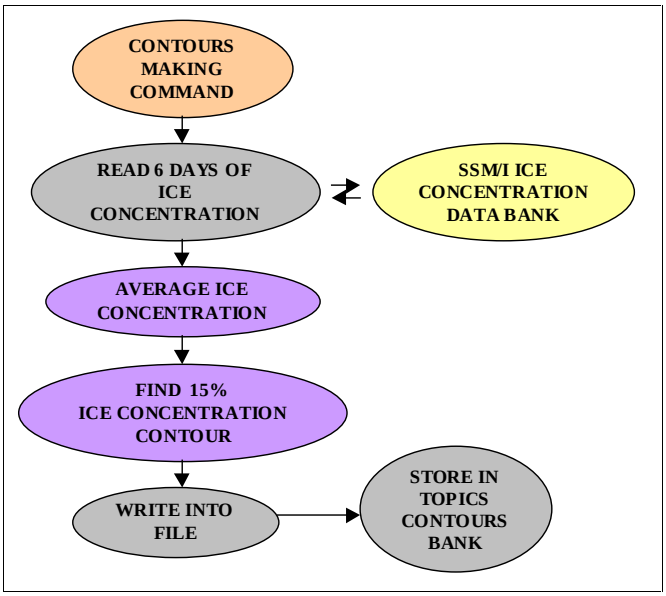
For a given region, TOPICS computes over one to three ice domains called “sea-ice”, “land(-ice)”,and “all ice”:

- “Sea-ice ” domain designates sea areas where ice concentration is greater than 15%. This latter value is commonly used to distinguish sea from sea-ice.
-
- “Land(-ice)” domain represents continental locations significantly or not covered with ice, depending on the region considered.
-
- “All ice” domain, when applicable, results from the union of “sea-ice” and “land-ice” domains.

These domains are determined applying two masks on the .sir image (see figure in "Statistics Computation"): first the region mask, which specifies where the investigation takes place (Ex: Weddel sea mask for Antarctic .sir images); then the ice domains mask, which splits the region into the ice domains defined above (some regions may be composed of only one ice domain : for example, the Beaufort sea in the Arctic Region simply contains a sea-ice domain. In that case, statistics are performed just on that particular domain).

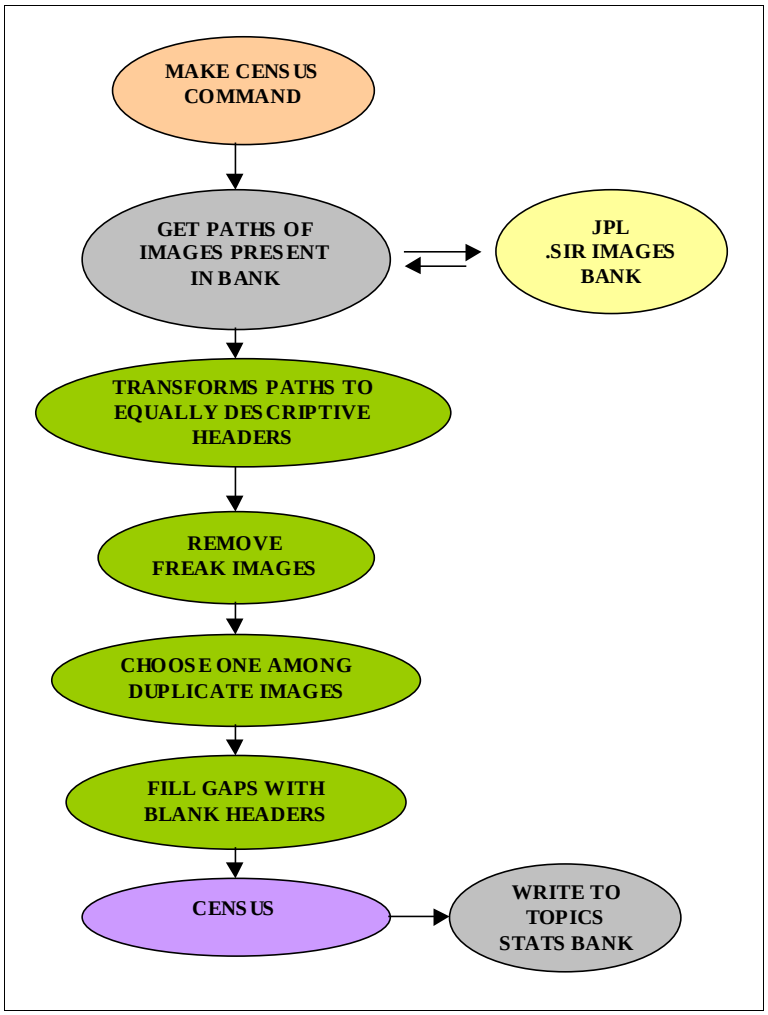
Ice contours bank

As sea-ice and thus land(-ice) domains change along the year, it is necessary to have an ice-contours bank. In order to match the .sir images bank, the (ice) contours bank contains 6-day averaged sea-ice contours at three day intervals for both Arctic and Antarctic regions (locations with latitude ≥60° and ≤-52° respectively). The contours are derived by averaging daily SSM/I ice-concentration grids over 6 days and finding the 15% iso-contours (the SSM/I is a passive microwave radiometric system mounted on the Defense Meteorological Satellite Program (DMSP). It has operated on 4 satellites since 1987. After data processing, maps are produced which give the percentage of ice content in a given location).



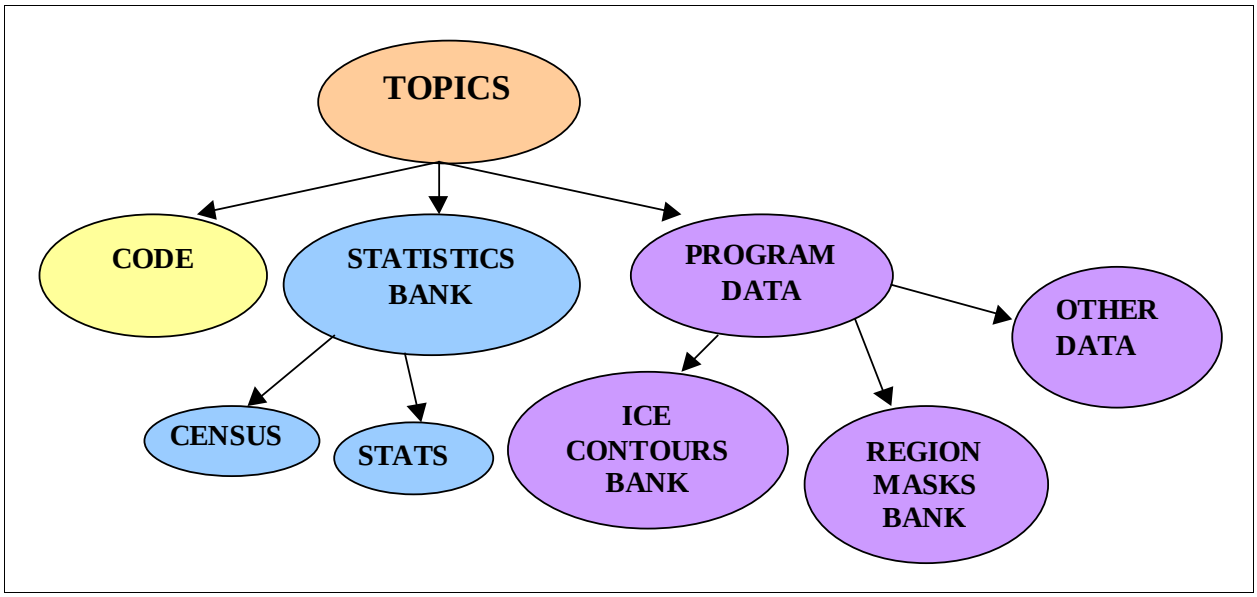
Census

Because JPL's .sir images bank has gaps, contains duplicate or weird images, TOPICS has its own internal representation of it; this representation is a census of .sir images which are useful to TOPICS; the figure below describes how TOPICS takes such a census.



TOPICS structure

TOPICS is naturally divided into three major parts : first, the code of the routines. Then a statistics bank which contains .stats files computed as explained above and the .sir images bank census. Finally, program data among which an ice contours bank needed to create ice masks, a regions masks bank which contains the contours of custom-defined regions, and other data the routines load or create when they run.

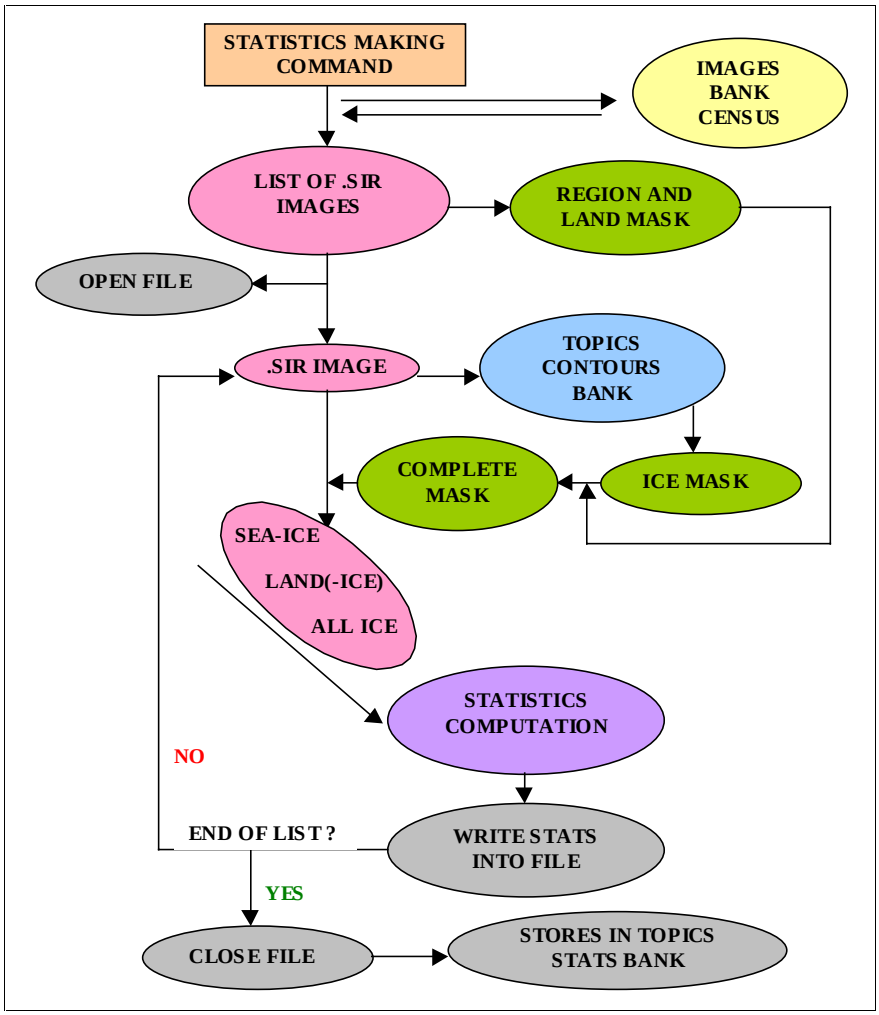


TOPICS' structure

Statistics computation

For each available ice domain, TOPICS computes the probability density function (PDF), the mean, the standard deviation (STD), and the median of the normalized cross-section backscatter σ^0. (For readers unfamiliar with statistics, if x is a random variable, the Probability Density Function f associated with x is so that the probability that x falls into $[x_0,x_0+dx]$ is equal to $f(x_0)dx$.)

The list of images to be processed depends on the time range entered by the user and is determined using the images bank census defined previously.



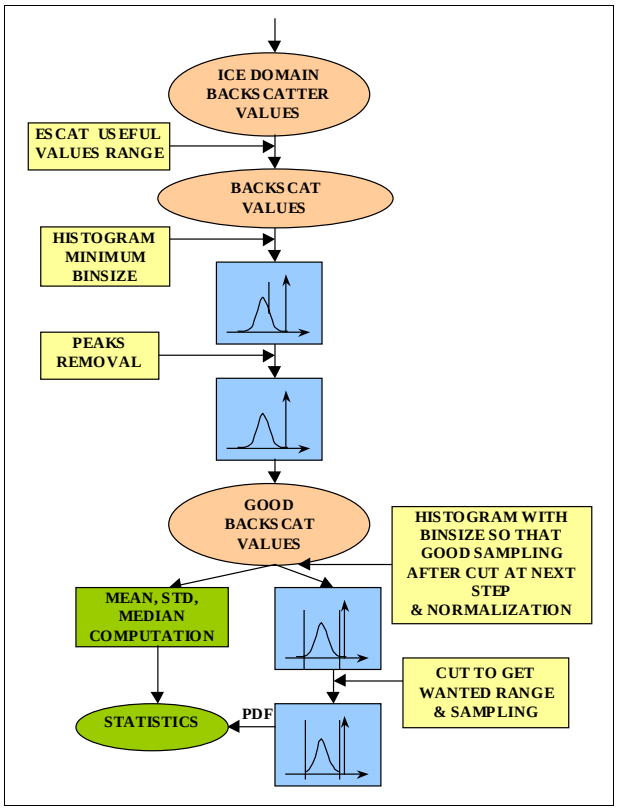
The pdf is computed assuming that backscatter values vary in the range [minimum useful value, maximum useful value] of the ERS data. However, because the pdf is almost always very small in the lower part of this interval, TOPICS only takes the significant part of the pdf computed this way. TOPICS display ranges are indicated below.

	A		B	
	MIN (dB)	MAX (dB)	MIN (dB)	MAX (dB)
ERS DATA USEFUL RANGE	-30.000	-0.001	-1.0000	-0.0001
TOPICS RANGE	-25.000	-0.001	-0.5000	-0.0001

The PDF computation has several steps :

1- A mask is applied to the .sir image to determine the collection of backscatter values inside a given region and a given ice domain.
2- Values outside the EScat useful range are cut.

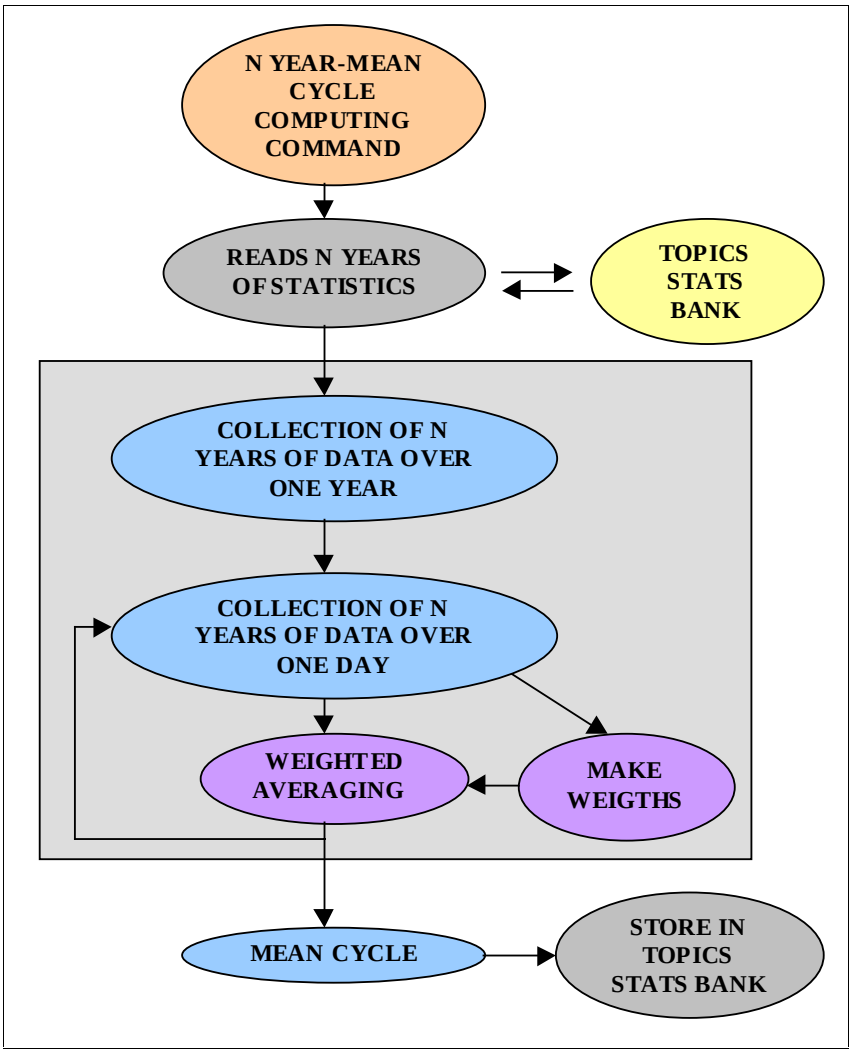
3- An histogram is made with a binsize corresponding to the values quantization (10^{-3}dB and 10^{-4}dB for A and B values respectively); this allows to see and remove artefacts peaks in the collection of backscatter values.
4- We then obtain the good backscatter values on which we compute the mean, standard deviation and median. As explained above the PDF is first computed over the range of these values, and then cut to fit the "useful" range. Original sampling is done so that the final result has the number of samples specified by the user.



statistics computation

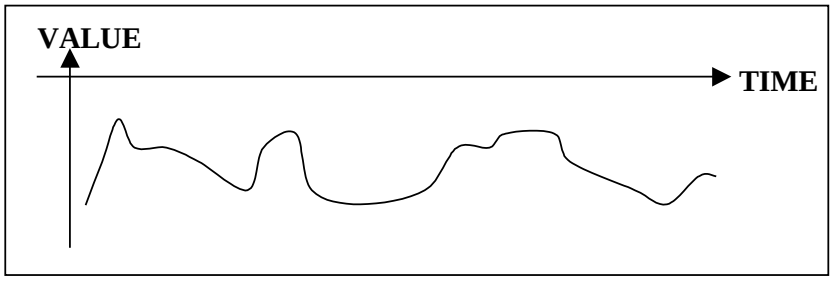
Mean cycles computation

Mean cycles are computed for each type of statistics by averaging several one-year long data pieces. A weighting is done that takes into account the fact that some statistics come from bad or partial images (see routines in annex for details).

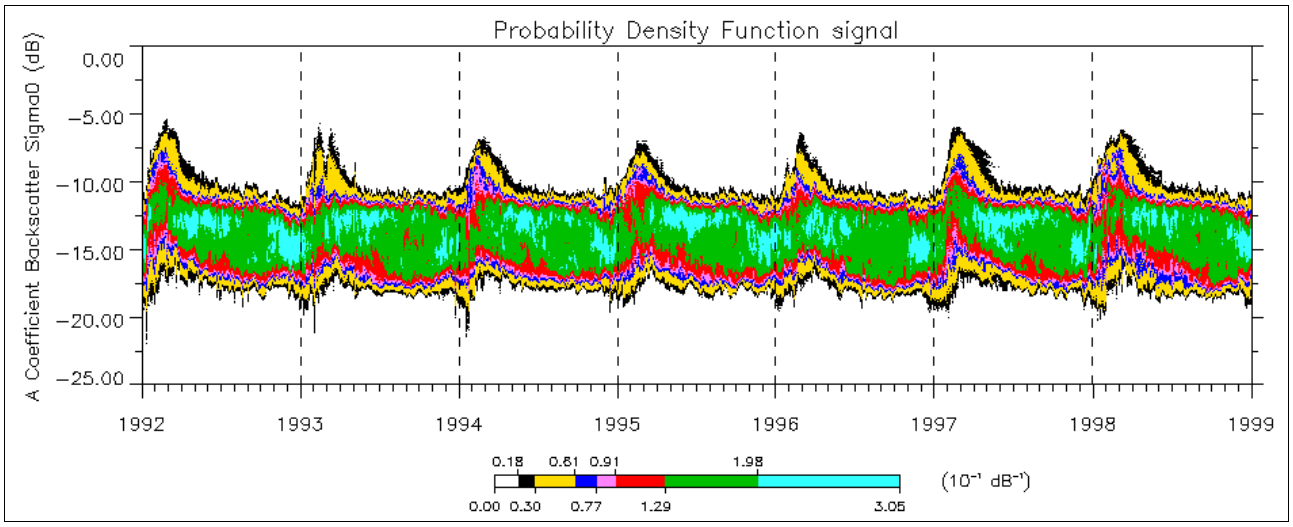


Graphical outputs

Mean, standard deviation and median time series are displayed as 1D-curves, whereas PDF time series, which would logically yield 3D-surfaces, are best viewed projected on a plane, as explained below.



usual graphs for 1D data such as mean, standard deviation or median

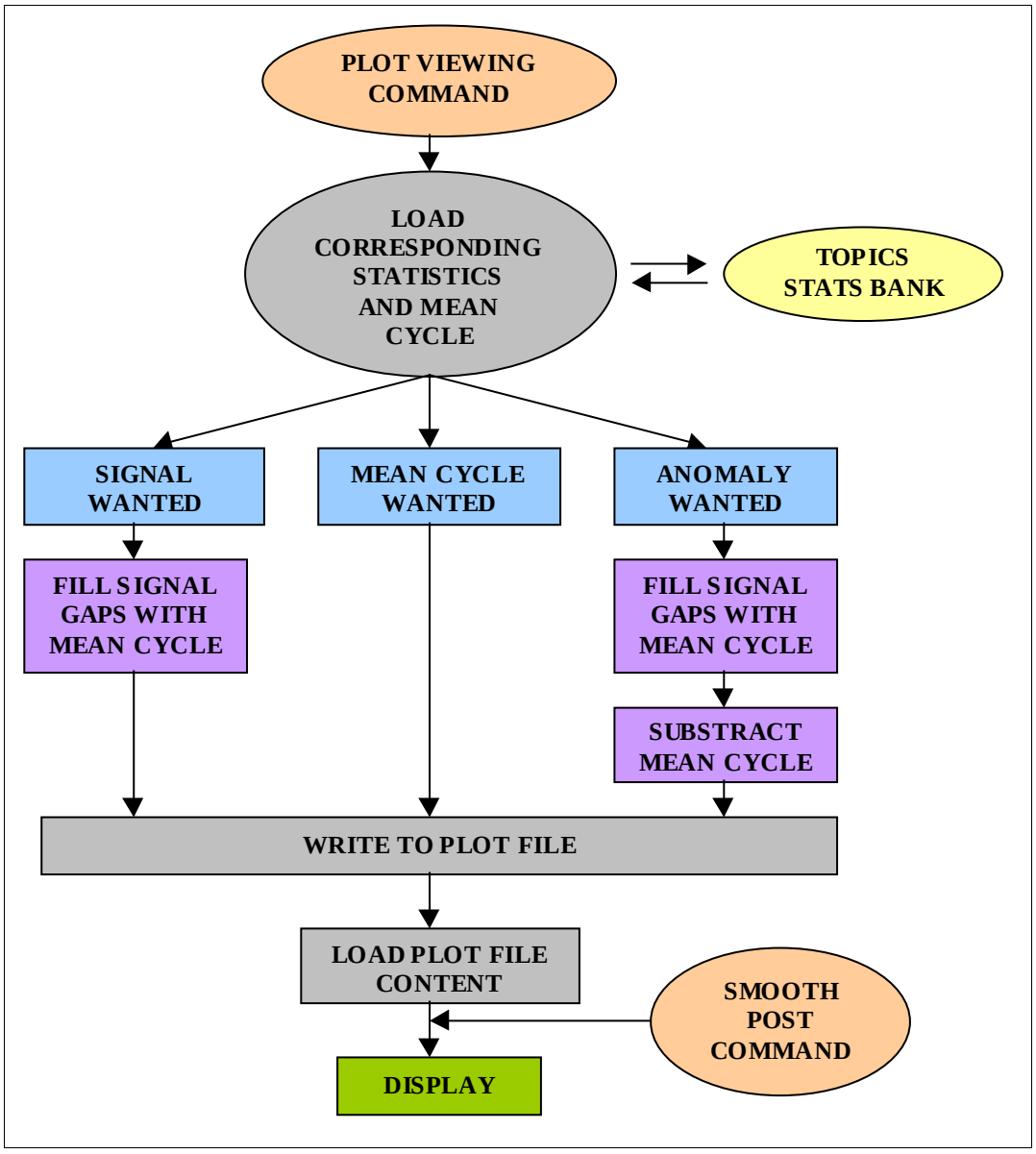


2D plots for the Probability Density Function (each slice of the plot results from the projection of the pdf at a given tine on the plane of the figure, values being coded with colors)

When plotting the data, TOPICS loads the desired statistics from the statistics bank, then :

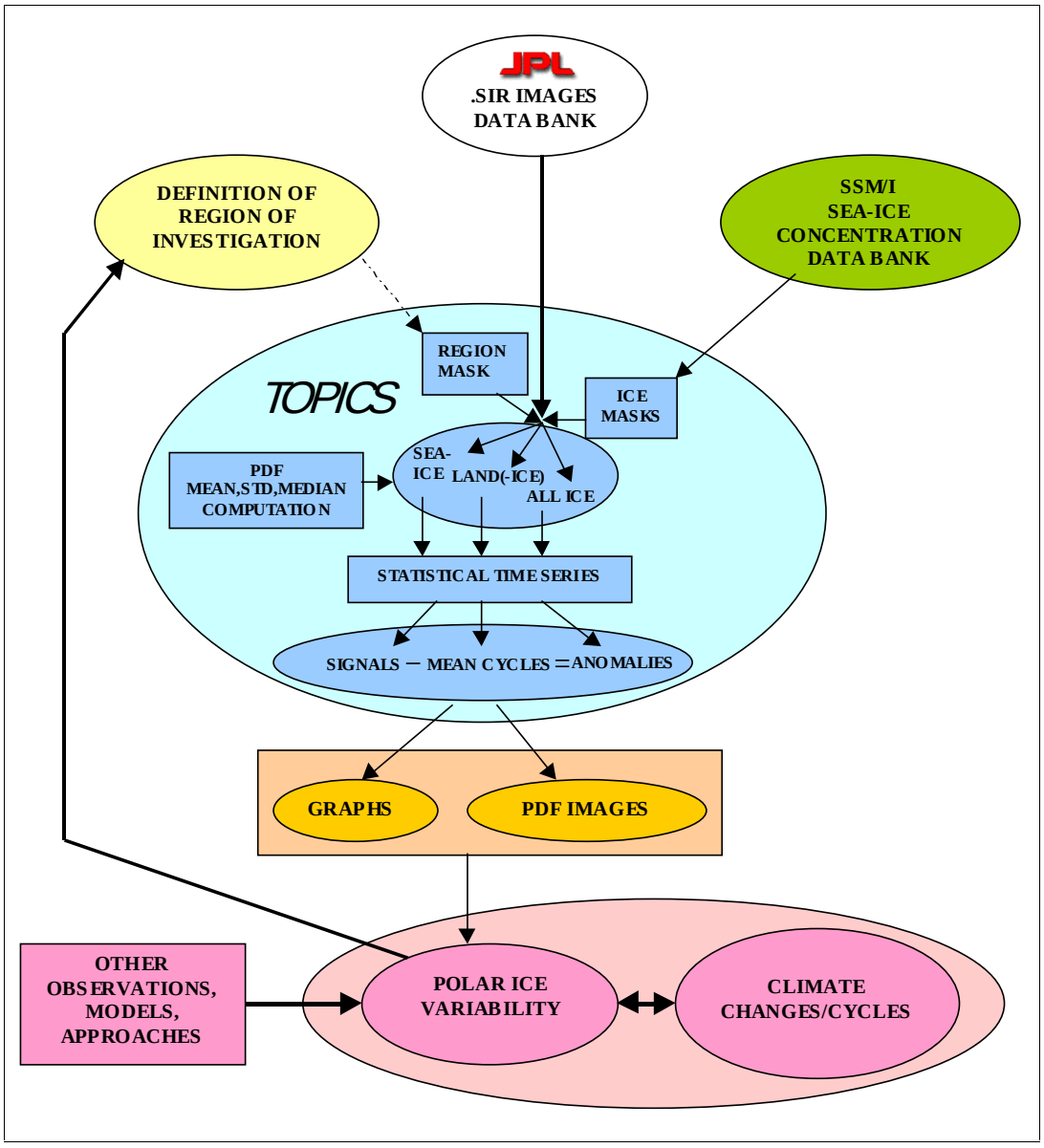
- fills the gaps with the values of the mean cycle if a signal is wanted
- fills the gaps with the values of the mean cycle and substract the mean cycle if an anomaly is wanted

The result is written to a file, which content is then loaded, smoothed if specified, and displayed.



Interpretation of results and feedback

Once output graphs have been interpreted, it is possible to make a feedback by defining an other or a smaller region, so as to see how different areas are related. This provides deeper understanding of the physical processes and trends at stake as well as a temporally dynamical point of view ; for example, it could be possible to see the propagation of certain phenomena around Antarctica due to the circumpolar wave.


.SIR IMAGES DATA BANK
DEFINITION OF REGION OF INVESTIGATION
SSM/I SEA-ICE CONCENTRATION DATA BANK
TOPICS
REGION MASK
ICE MASKS
PDF MEAN,STD,MEDIAN COMPUTATION
SEA-ICE
LAND(-ICE)
ALL ICE
STATISTICAL TIME SERIES
SIGNALS − MEAN CYCLES = ANOMALIES
GRAPHS
PDF IMAGES
OTHER OBSERVATIONS, MODELS, APPROACHES
POLAR ICE VARIABILITY
CLIMATE CHANGES/CYCLES

FOURTH PART : GRAPHICAL INTERFACE DESCRIPTION

This part intends to provide a quick overview on how to use TOPICS' graphical interface.

TOPICS installation and first-time start

To install TOPICS, follow the steps indicated below:

1) Download the directory called TOPICS from the CD. It is recommended not to rename TOPICS main directory.
2) Unzip TOPICS
3) Make sure PVwave is installed on your computer
4) Set your current directory to […]/TOPICS/CODE/TOPICS_STARTUP_ROUTINES
5) Run PVwave
6) Type "@setup_topics" at wave prompt (remark: when asked to enter TOPICS' directory path, type the whole path from the root and finish by "/TOPICS"). This tells PVwave where to find TOPICS' routines, and also compiles all of them.
7) If no error occurred, type "topics" at the prompt line. After 5 to 10 seconds, TOPICS cover window should appear. If an error did occur, you may have mistyped TOPICS' directory path; try step 5 again.
8) Congratulations, you're in! Go to the next section.

Remark: quicker start

If you do not want to type TOPICS' directory path each time you run TOPICS, you can change your Wave startup file so that it creates a common string variable named topics_directory_path containing TOPICS' directory path.

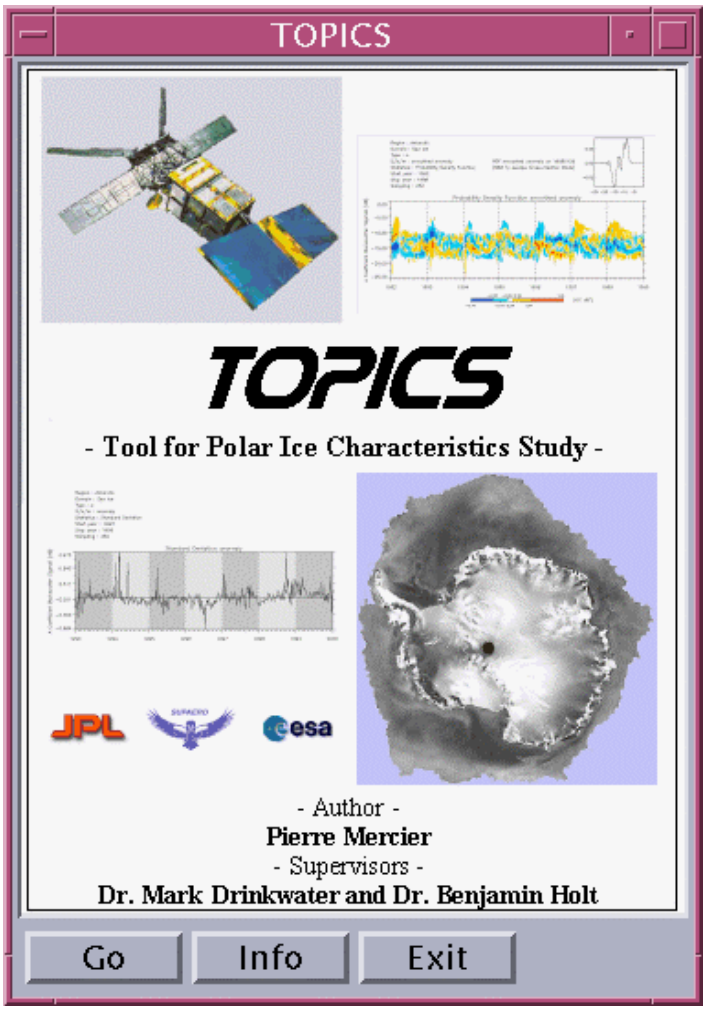
You may also add the path […]/TOPICS/CODE/TOPICS_STARTUP_ROUTINES to !path, which avoids having to set the current directory to that path before you run Wave.

There are many ways to improve TOPICS' start. I deliberately kept it simple so that anyone can customize it the way he wants. If you want to do so, TOPICS' startup code can be found in annex.

The cover window

It is the very first window that appears when you run TOPICS. It offers three options:

1- Go: enters TOPICS; see next section
2- Info: displays general information about TOPICS
3- Quit: exits TOPICS



Rmk: if you have problem with flashing colors, close other applications (Netscape, Mail Tool,…), exit Pvwave, restart it and run TOPICS again.

The main menu bar

The main menu bar appears when you enter TOPICS from the cover window. It is composed of four different menus :

1- Manage Banks: allows to manage images, contours and statistics banks. If you click this label, three submenus appear:

- Images Bank (only one option): takes the census of .sir images bank. An on-line report is displayed on the command line.

- Contours Bank (only one option): allows to update contours bank (see “contours computation window” below).

- Statistics Bank (only one option): allows to compute statistics and mean cycles (see “statistics computation window” below).

2- Customize: If you click on this label, you can choose between the two possibilities:

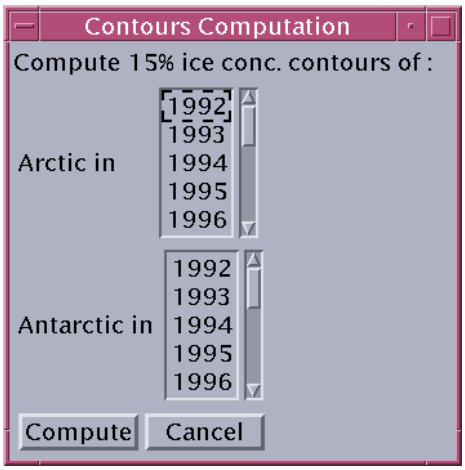
- See Current Regions: displays a map of currently defined regions (see “” below)
- Define New Region: allows the user to define its own regions of investigation (see “define windows” below).

3- View: allows the user to view time series plots. If you click this label, the view window appears (see “view window” below)

4- Quit: exits TOPICS.

The Contours Computation Window

It appears when you click on the label Main Menu Bar > Manage Banks > Contours Bank > Compute Contours. It allows to compute sea-ice contours for a specified year in Arctic and/or Antarctic.

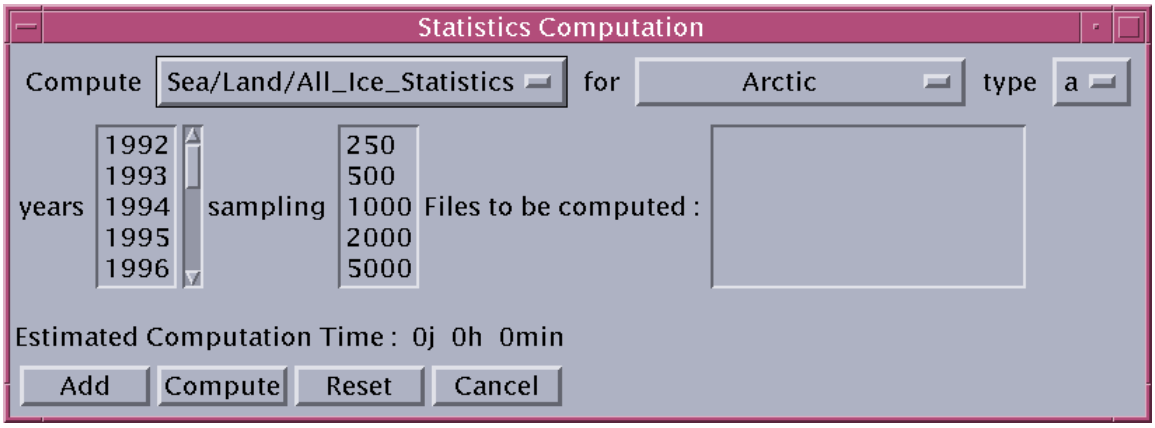


To compute contours, select year(s) with your mouse on the scrolling lists for Arctic and/or Antarctic and click on the « compute » button. Remark : You can select different years and different regions at the same time.

To exit the window or abort, click on “Cancel”.

The Statistics Computation Window

This window appears when you click on the label Main Menu Bar > Manage Banks > Statistics Bank > Compute Statistics. It allows compute statistics or mean cycles computation.



To start a computation, make a selection for the five following parameters:

1- what: 4 options : Sea Ice, Land(-ice) and All ice statistics / Sea Ice Statistics only / Land(-ice) statistics only / mean cycle.
2- region: choose among currently defined regions.
3- type: A or B, corresponding to A and B coefficient for σ^0.
4- years: select one or several years for which statistics will be computed.
5- sampling: the number of samples of the pdf. You can make one or more selections for that field.

Then, click on “Add” button. This causes your computation command to be stored into the memory. Files to be produced by this command appear on the right of the label “Files to be computed : ”.

If you wish to add other commands, proceed the same way. If you want to start the execution of the computation commands, click on the “Compute” button.

In case you made a mistake entering a command, press the “Reset” button. This erases the content of the computation commands memory.

To exit the window or abort, click on “Cancel”.

Remarks :

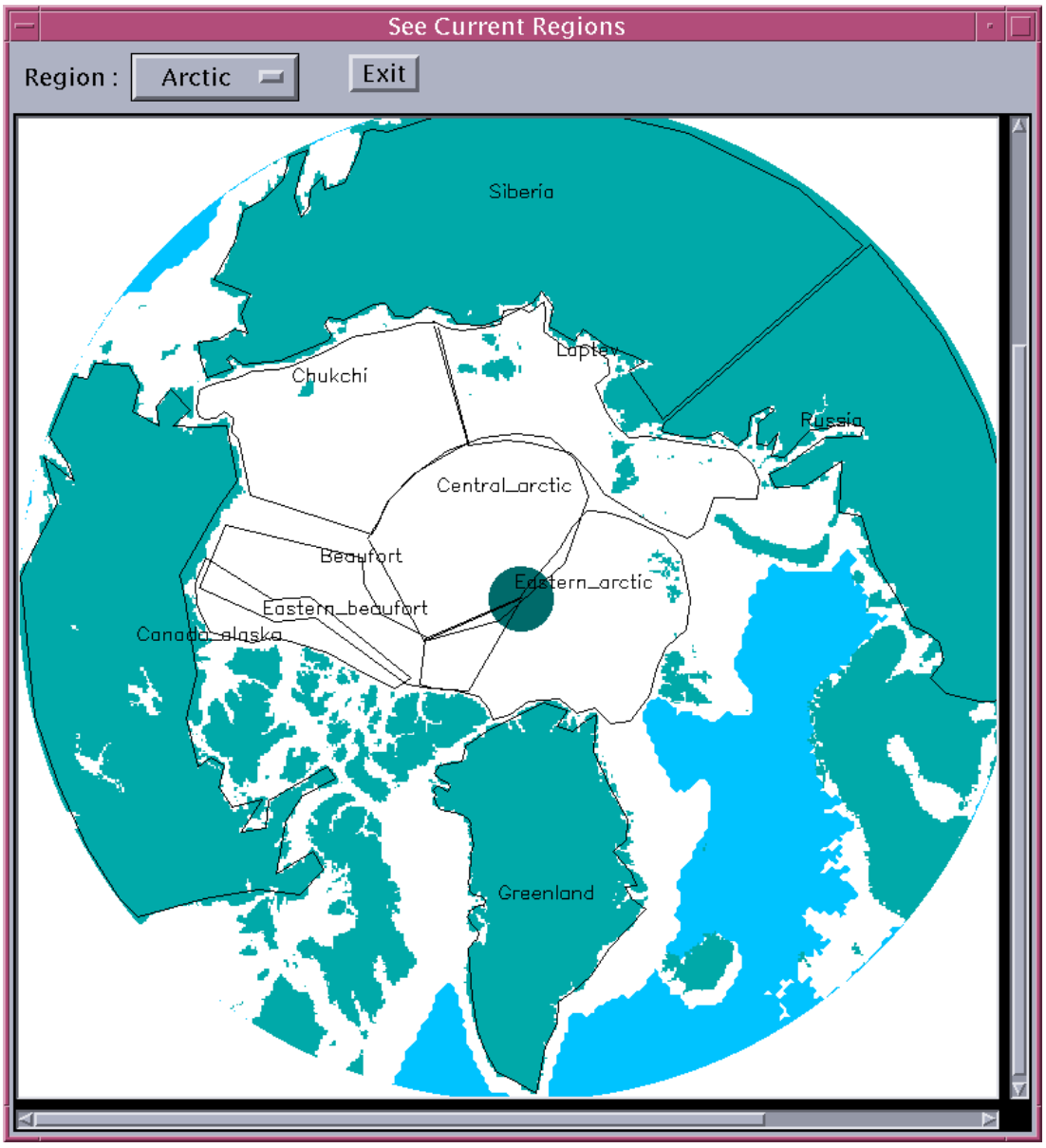
1- An estimation of the time required to execute all the computation commands is indicated just above the “Add” and “Compute” buttons. Depending on how much RAM your computer has, this

estimation may not be accurate. You can correct this by modifying TOPICS' code (section "estimated computation time" in routine config.pro).

2- In case of multiple selections for both years and sampling, TOPICS will work as if you had entered several computation commands with one selection for years and multiple selections for sampling. Ex : if you choose 1992, 1993 for years and 250, 500, 1000 for sampling, then TOPICS will "understand" 1992/250, 1992/500, 1992/1000, 1993/250, 1993/500, 1993/1000.

The See Current Regions window

This window appears when you click on the label Main Menu Bar > Customize > See regions. It allows to see currently defined regions in Arctic and Antarctic.

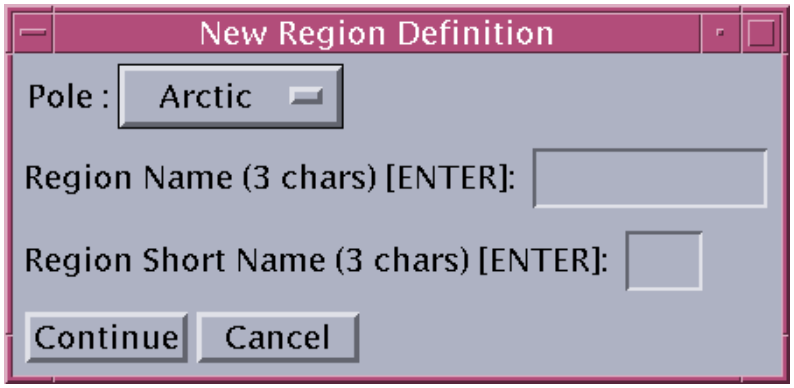


To see currently defined regions, select “Arctic” or “Antarctic” in the option menu. After a few seconds, a map of the selected polar region appears, which shows the contours of the currently defined regions; land appears green, sea blue, and ice white. The green disk in the middlecorresponds to satellite non-visibility areas.

The “Exit” button exits the window and return to the main menu bar.

The New Region Definition window

This window appears when you click on the label Main Menu Bar > Customize > Define New Region. It allows to define a new region of investigation in Arctic or Antarctic.



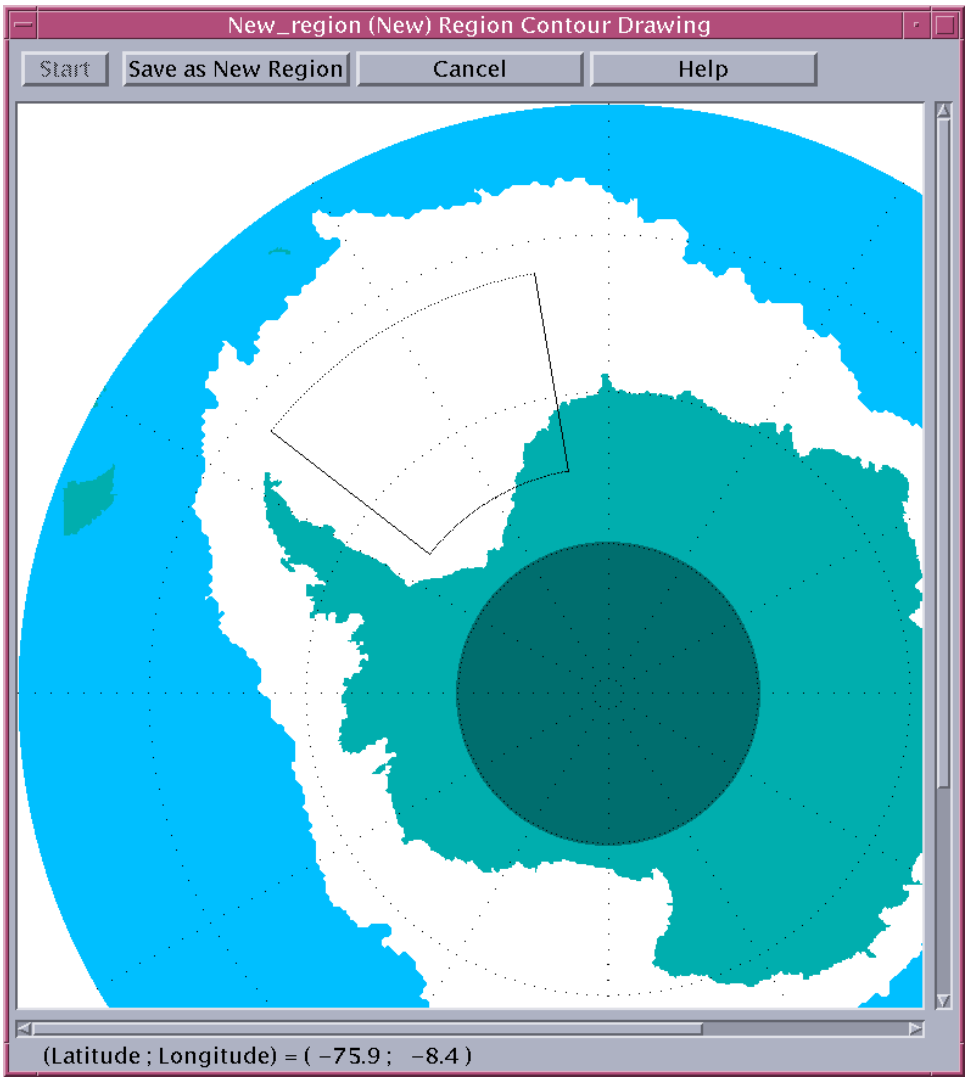
To begin the definition of a new region, specify at which pole it is located. Then type the Region Name (no blanks – use underscore). Don’t forget to hit ENTER, otherwise your entry will not be taken into account, even if it appears in the window. If you forget, an alert box will appear. Finally, enter the Region Short Name, which must be 3 characters long. The region short name is used in filenames and should clearly remind the region name. Same remark as above concerning ENTER.

If your entries are correct and you want to continue the definition, press “Continue”, which will open the Region Contour Drawing window (see “Region Contour Drawing window” below).

To abort or exit, click on “Cancel”.

The Region Contour Drawing window

This window appears when you click on the button « Continue » in the New Region Definition Window. It allows to complete the definition of a new region that was begun in the New Region Definition Window by drawing its contour on a map.



To get some information on how to draw the contour, click on the “Help” button.

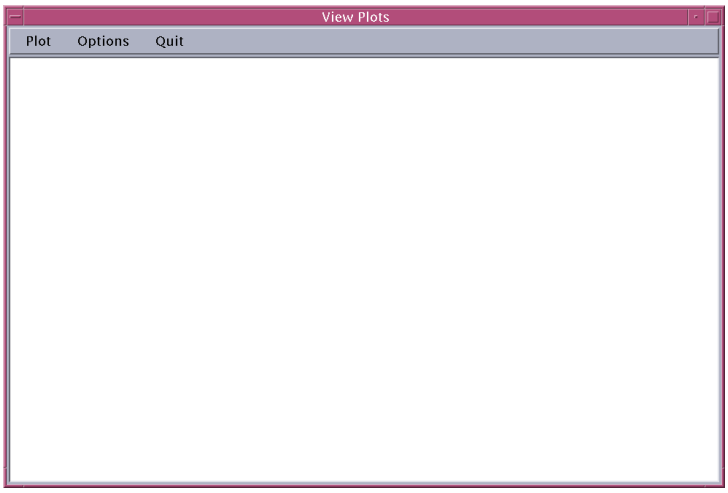
To begin the drawing, press the “Start” button, which enables the “mouse drawing mode” :

- if you move the mouse cursor on the map, the coordinates of the corresponding location will appear in the lower left corner of the window.
- if you press the first button of the mouse, this will either define the first point of the contour or draw a line between the last defined point and the current point.
- If you press the second button of the mouse, this will draw an arc of parallel joining the last defined point P and the projection of the current point on the parallel passing through P.
- Finally, if you press the third mouse button, TOPICS will close the contour and exit the “mouse drawing mode”.

To save the drawn contour and define the new region, press “Save as New Region”. To abort the definition, press “Cancel”.

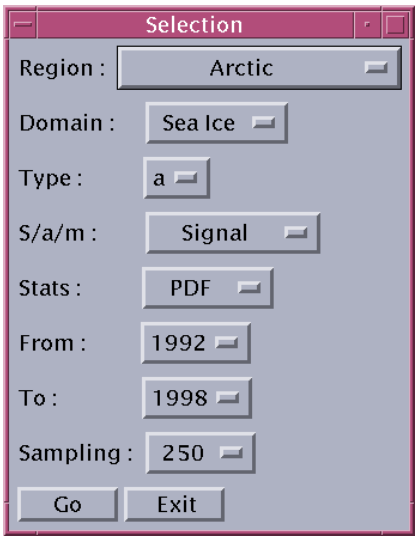
The View Plots window

This window appears when you click on the label Main Menu Bar > View > View Plots. It allows to visualize statistics time series.



The Selection window

To select the plot you want to see, go to View Plots window > Plot > Select. The following selection window appears (remark: the label S/a/m means “Signal/anomaly/mean cycle”) :

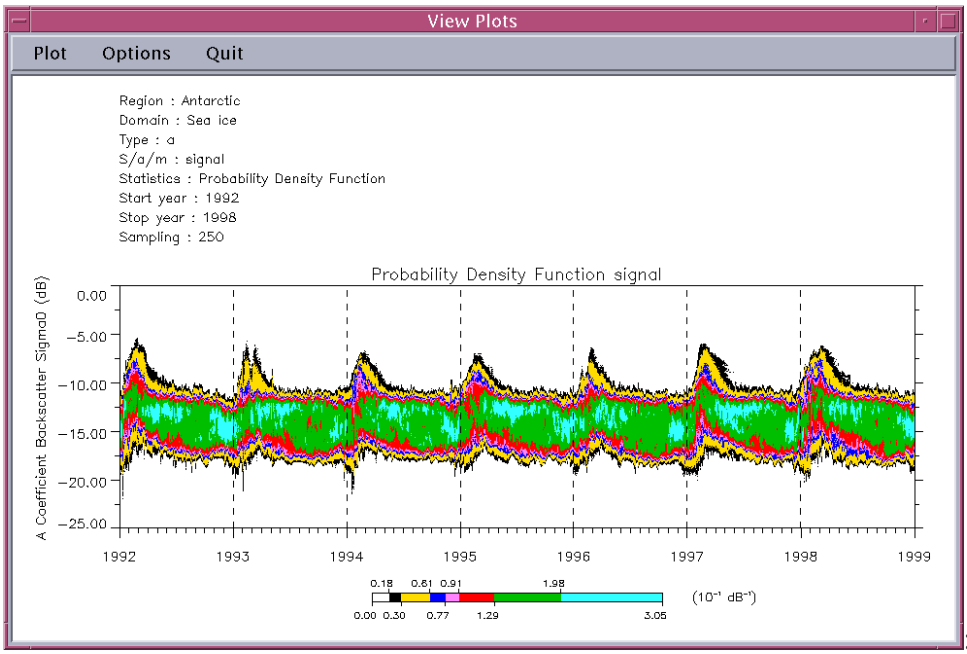


Once your selection is made, click “Go” to validate it or “Exit” to cancel it.

If we assume that the following viewing parameters were selected :

Region	Antarctic
Domain	Sea Ice
Type	a
S/a/m	Signal
Stats	PDF
From	1992
To	1998
Sampling	250

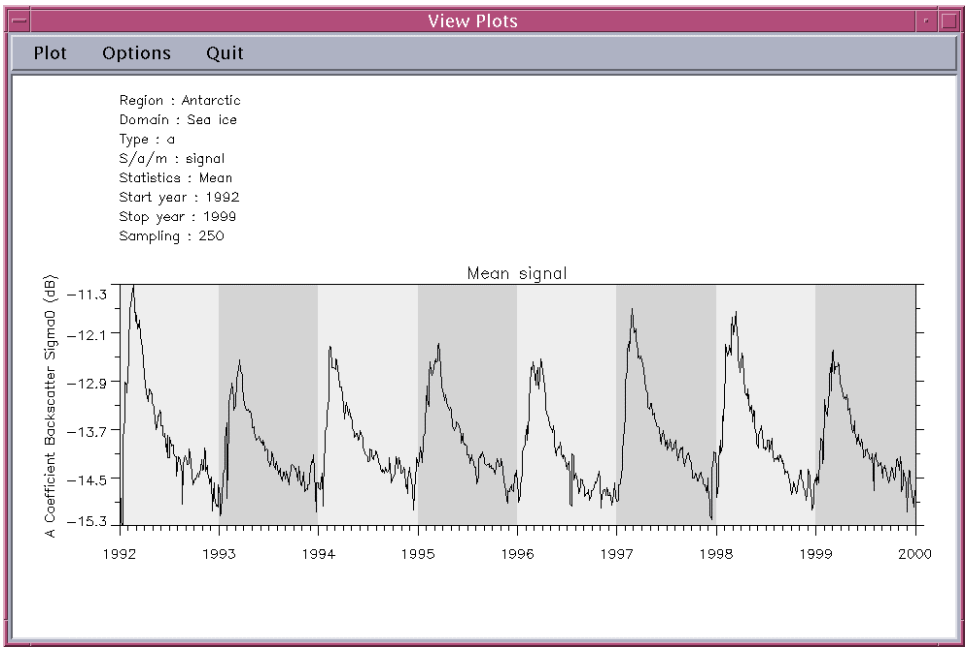
then the View Plots window will display the following plot :



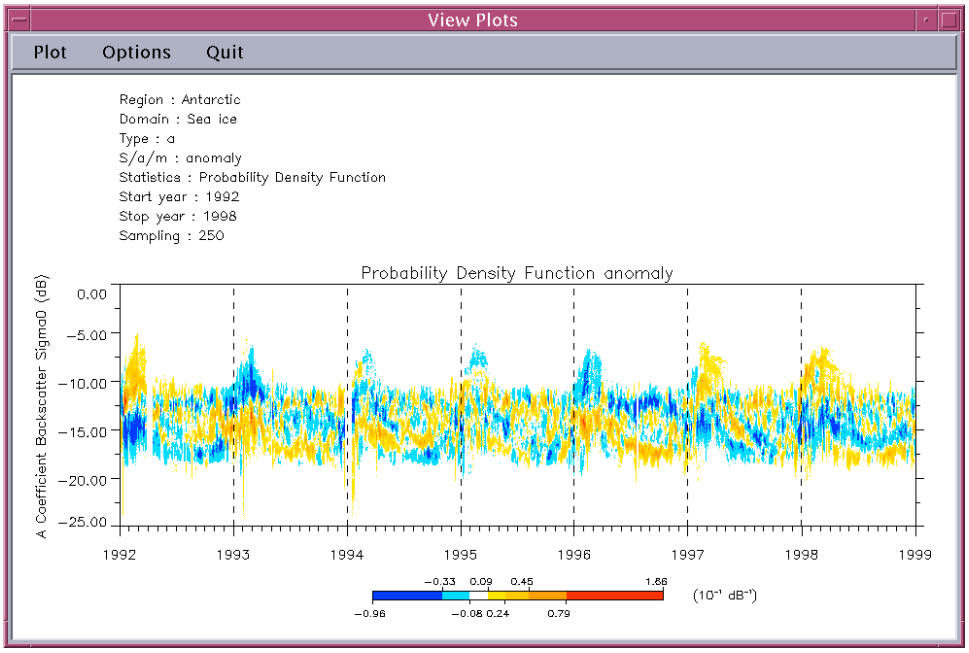
:

You can see that selected parameters are reminded on the upper left corner of the window. By default, the plots are shown in colors. However, we will see that it is possible to change the color table to a gray-level one.

You could also have selected mean/std/median time series; in that case, the plot looks different:

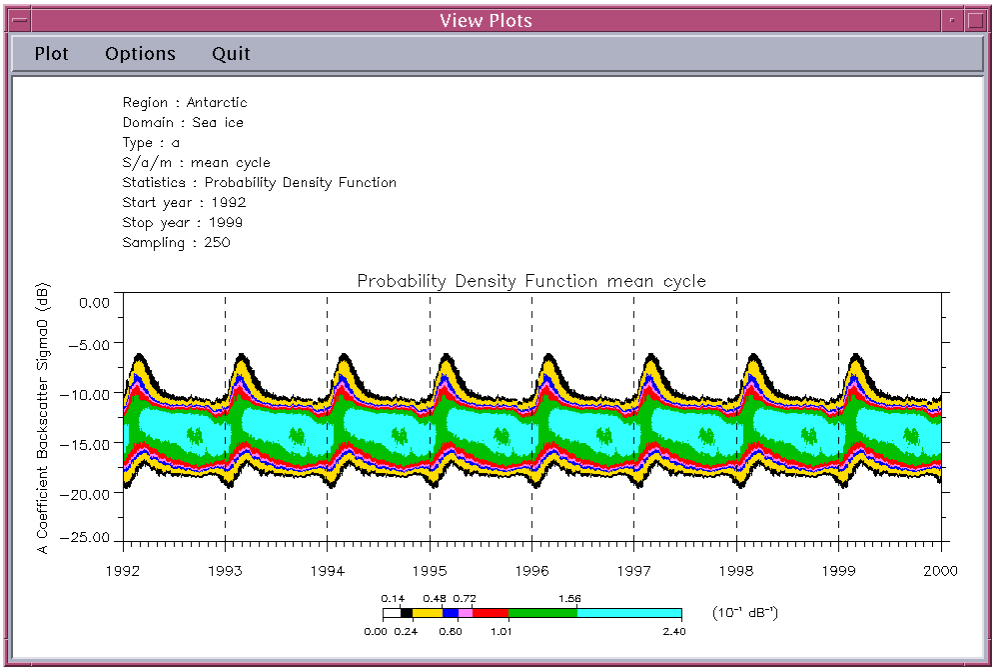


If you go to PLOT > SELECT, the Selection Window will appear again, with the plot parameters fields set to the current values of the plot parameters. If you then change the field “S/a/m” to “anomaly” and press “Go” then we will obtain the following anomaly plot:

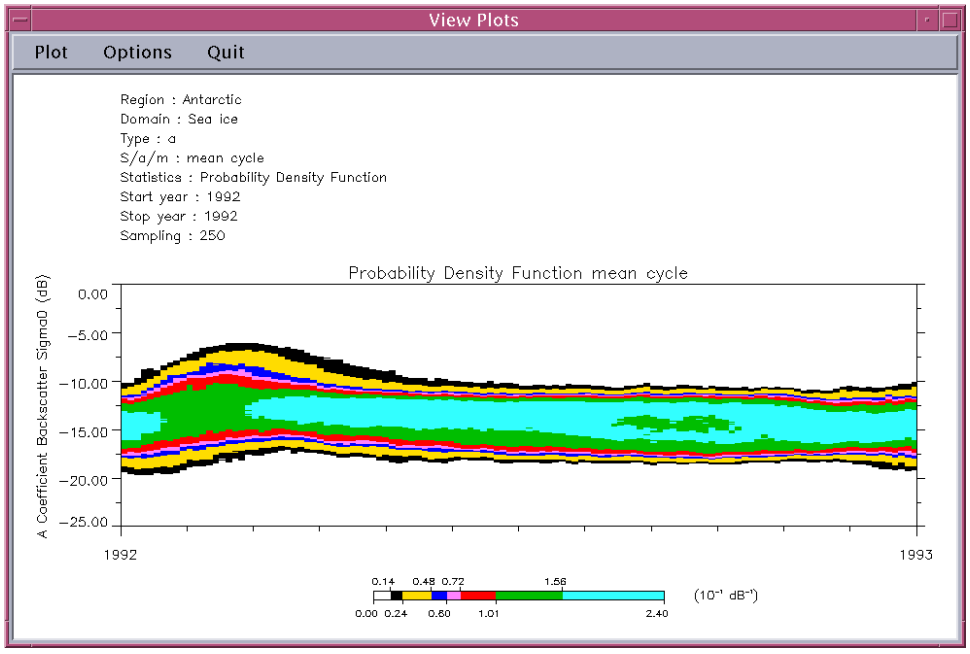


You can notice that the color table changed, so as to be adapted to the range and distribution of the anomaly values. To obtain the mean cycle that was substracted to the signal to get the anomaly,

reopen the selection window and change the field “S/a/m” to “mean cycle”. The View Plots window displays :



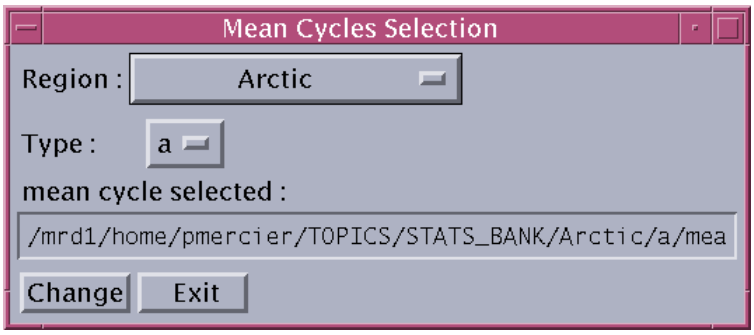
If you want to have a closer look to the mean cycle (which is duplicated on the previous plot), change the field “To” to “1993” in the Selection window; this gives the following annual cycle :



Options

Change Mean Cycle

It is possible to change the mean cycle used to obtain the anomaly. To do so, compute the desired mean cycle (See “Statistics Computation window” above), return to the View Plots window and go to Options > Choose mean Cycle. The Mean Cycles Selection window appears:

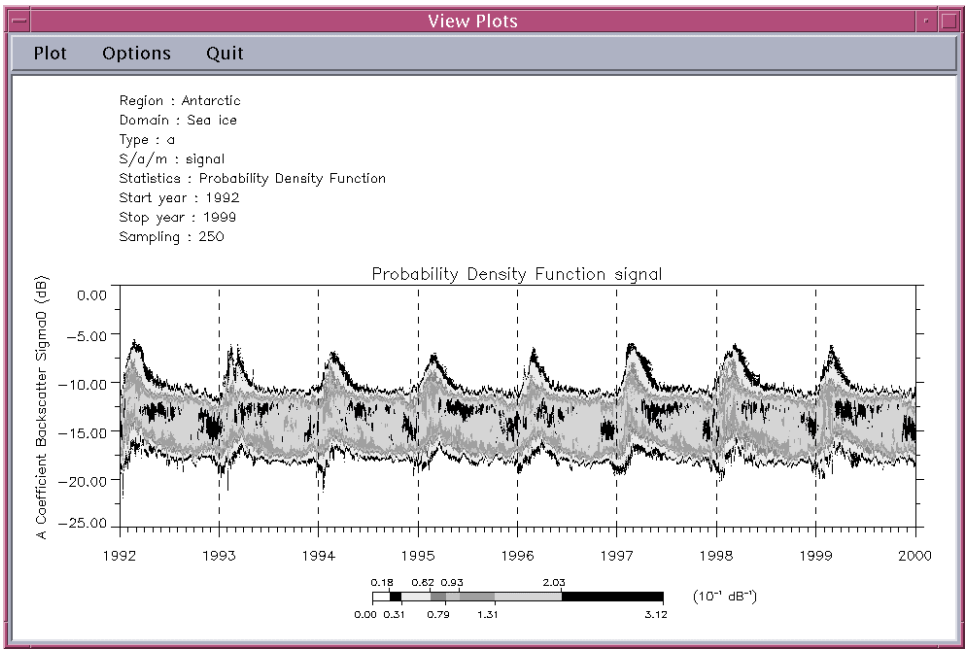


The text field displays the path of the file containing the mean cycle currently selected for the region specified by the first two fields “Region” and “Type”. You can change these two fields to see all currently selected mean cycles. To change the mean cycle, press the “Change” button; a classical file selection window appears that invites you to enter/choose the path of the file containing the new mean cycle. Once done, TOPICS returns to the View Plots window. The mean cycle is now changed, as you can check by opening the Mean Cycles Selection window.

Important remark: the changes done this way will last just for the current TOPICS session. If you quit TOPICS and restart it later, the mean cycles selected to compute anomalies will be initialized to their default values. To make permanent changes, you have to modify the file “mean_cycles_selected_default” located at /TOPICS/PROGRAM_DATA/MEAN_CYCLES_SELECTED/. Be very careful not to remove the first figure at the beginning of that file or add any blanks at the end of it.

Color Table (Only for pdf plots)

By default, plots will appear in color. However, for some reasons (Ex: printing) you may want to have them in gray levels. To do so, go to Options > Color Table > Gray . This changes the color table instantly; TOPICS’ gray level color tables are designed to match exactly the color ones. In the case of the previous example, we obtain the following gray-level plot:

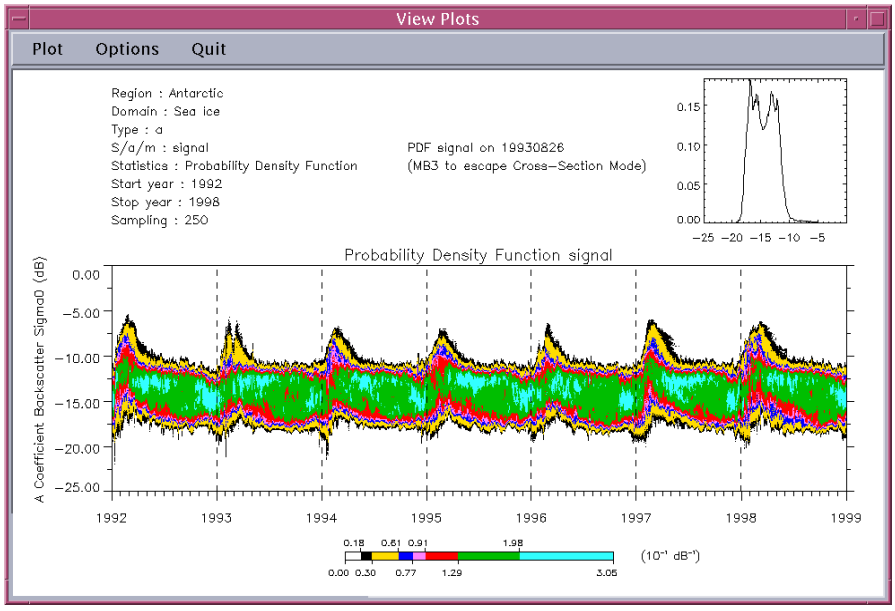


To obtain the color version again, just go to Options > Color Table > Color.

Remark : this option has no effect on plots other than pdf's

Cross-Section (Only for pdf plots)

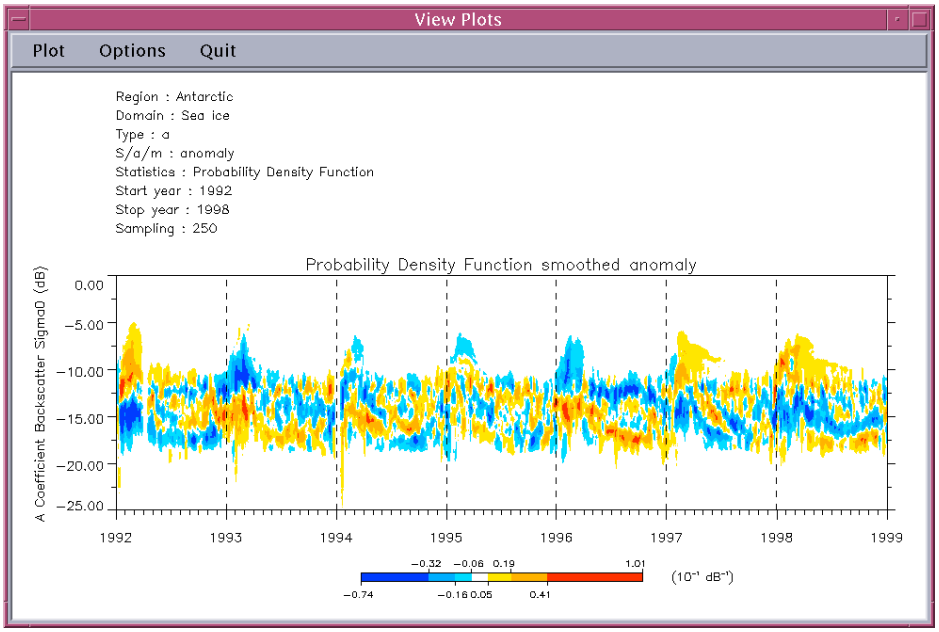
If you go to Options > Cross-section, you will enable the cross-section mode, which displays the cross-section of the plot at the current location of the mouse cursor. This is a continuous display so that you can see the evolution of the pdf with time.



Remarks : - The only mouse-sensitive zone is the plot
- This option has no effect on plots other than pdf's

Smooth (only for pdf plots)

It is sometimes necessary to smooth some plots that do not appear very neat. To do so, go to Options > Smooth. The plot on display will be smoothed using a 3-pixel wide box averaging method. As a result, extremal values may be changed, which will cause the color table to adapt to the new range. Generally, better contrast is an extra benefit when smoothing. This is illustated by the plot below, obtained by smoothing the anomaly we used as an example earlier.



Remark : this option has no effect on plots other than pdf's

Saving plots

To save the plot, go to Plot > Save or Plot > Save as. The first possibility saves the current plot as a TIFF image and places it in TOPICS tiff images directory (/TOPICS/PICS_LIBRARY/TIFFS/ + region directory). The filename format is Rrr_t_dd_sss_k.tiff where :

Rrr is the region short name (Ant/Arc/Grd/….)
t is the type (A/B)
dd is the ice domain (si = sea ice, li = land ice, ai = all ice)
sss is the statistics performed (pdf/mean/std/median)
k is the kind of plot (s = signal, a = anomaly, m = mean cycle)

The second possibility lets you decide the name and the location of the TIFF image produced.

Printing plots

To print the current plot, go to Plot > Print; a dialog box appears in which you can enter the print command you want executed. TOPICS then saves the plots as a temporary TIFF image and has the UNIX system execute the specified print command.

Remark: make sure your computer knows how to send a TIFF image to the printer, otherwise you may have bad surprises…

FIFTH PART : RESULTS

Statistics computed

Signals, anomalies, mean cycles for the Probability Density Function ,the mean, the standard deviation and the median have been successfully computed for the period 1992-2000 in Arctic, Antarctic and the 16 custom regions over the following domains :

	A			B		
	Sea Ice	Land Ice	All Ice	Sea Ice	Land Ice	All Ice
Amundsen-Ross	x			x		
Antarctic	x	x	x	x	x	x
Arctic	x	x	x	x	x	x
Beaufort	x			x		
Bellingshausen	x			x		
Canada-Alaska		x			x	
Central Arctic	x			x		
Chukchi	x			x		
East east Antarctica		x			x	
Eastern Arctic	x			x		
Eastern Beaufort	x			x		
Greenland		x			x	
Laptev	x			x		
Russia		x			x	
Siberia		x			x	
Weddel sea	x			x		
West Antarctica		x			x	
West east Antarctica		x			x	

Problems in the data set

Two remarks immediately arise when looking at plots :

1- first the data of 1992 are very different from the others (see).
2- second, there is clearly a shift in pdf anomalies between ERS-1 data (before the first third of 1996) and those of ERS-2 (after that date) on many anomaly plots (see).

The first problem is likely to be accounted for by poor measurements made by ERS-1 during its first year of operation. It also suggest that this year of data shouldn't be used to compute the mean cycles.

The cause of the second one is still under investigation, however Dr DRINKWATER and Dr LONG, who processes the .sir images at Brigham University, think it may come from a changein ERS-2 calibration or antenna pattern that would not have been taken into account when applying the SIRF algorithm. Indeed, all .sir images (even ERS-2's) are computed assuming that calibration and antenna

pattern are that of ERS-1, which should be the case because the two instruments are identical. Dr LONG has computed .sir images for the period 1992-2000 over the amazon forest, which serves as zero reference for the calibration, and has noticed a similar shift, which indicates that there is definitely a problem of calibration assumption here. However, things might not be that simple, and time is required to check all the steps of the .sir images making. For example, it is not clear yet why the shift appears on some images and not on others.

Does that mean that our results are wrong ? Not really, it just means that we cannot compare ERS-1 and ERS-2 data. But qualitative interpretations of time series over periods where the same satellite was operating remain valid.

General comments about the plots

Considering the fact that I am not a specialist of polar ice, and that plots may be biased for the reason explained above I prefer to let scientist make proper interpretations of the plots that follow. However, here are a few comments I would like to make :

1- As expected, land ice signals are really steady, whereas sea-ice ones change with time because of seasonal melting/freezing.

2- Arctic regions plots are quite different from Antarctic ones, in terms of ranges and evolutions. Indeed, Antartic climate varies much less than Arctic's, because of the influence of the continental sheet.

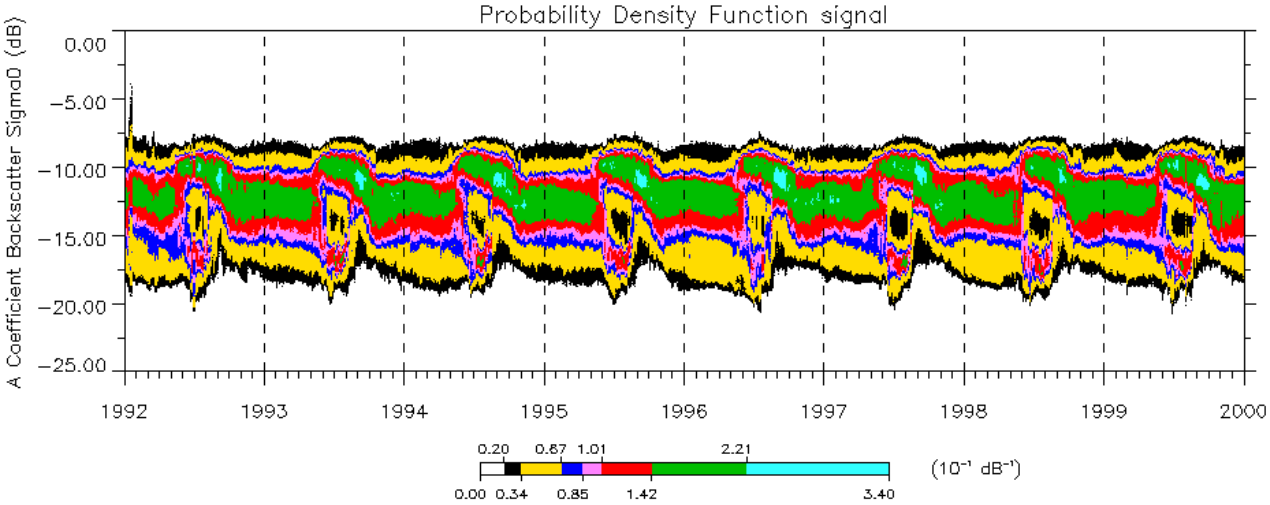
3- More specifically, the PDF is mono or bimodal in Antarctic regions, whereas is can be trimodal in Arctic regions.

4- Some interesting phenomena often occur during El Nino years (1992, 1997,1998). During those years, values are closer to zero than normal, which indicates that less young ice is formed, certainly because of the ocean being warmer under the influence of El Nino.

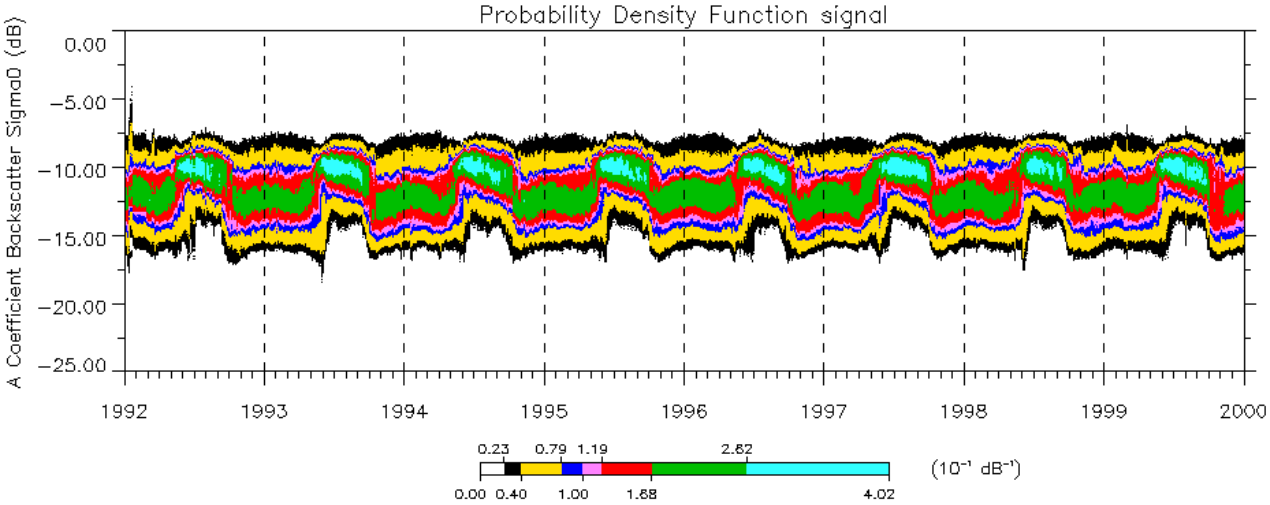
5- Finally, several Greenland plots seem to indicate a drift towards values closer to zero, which is certainly to be linked to human activity impact on environment.

Examples of plots (see CD-rom for other plots)

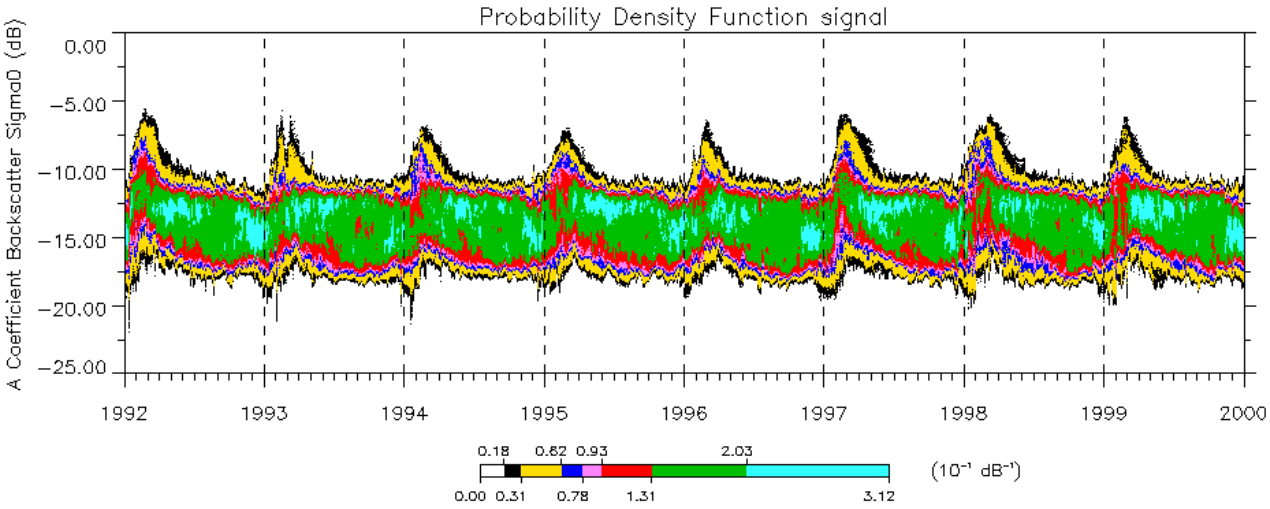
Region : Arctic
Domain : All ice
Type : σ
S/σ/m : signal
Statistics : Probability Density Function
Start year : 1992
Stop year : 1999
Sampling : 250



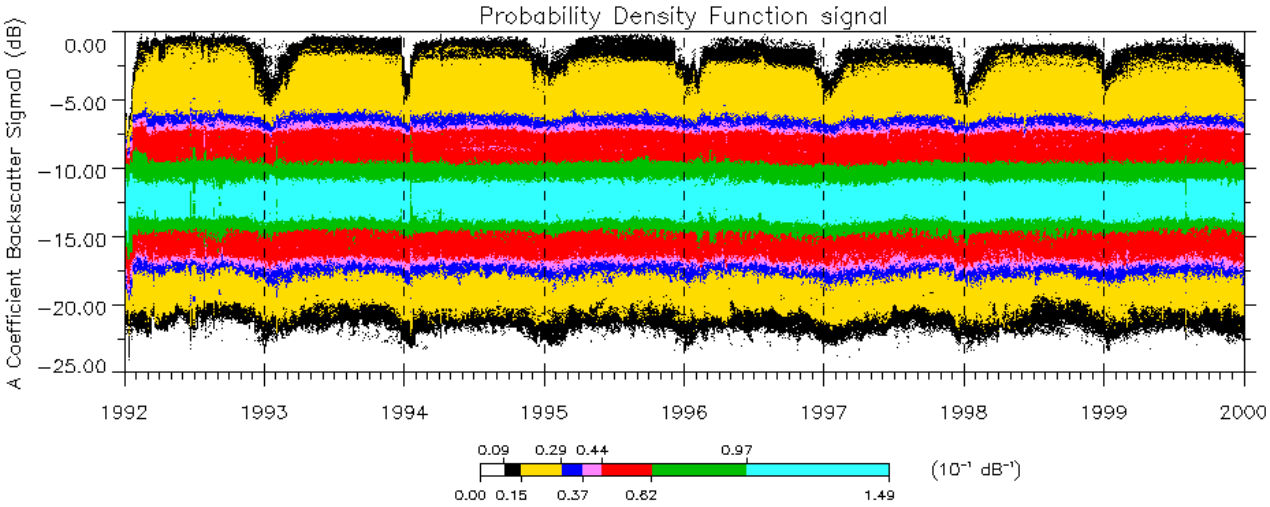
Region : Arctic
Domain : Land(-ice)
Type : σ
S/σ/m : signal
Statistics : Probability Density Function
Start year : 1992
Stop year : 1999
Sampling : 250



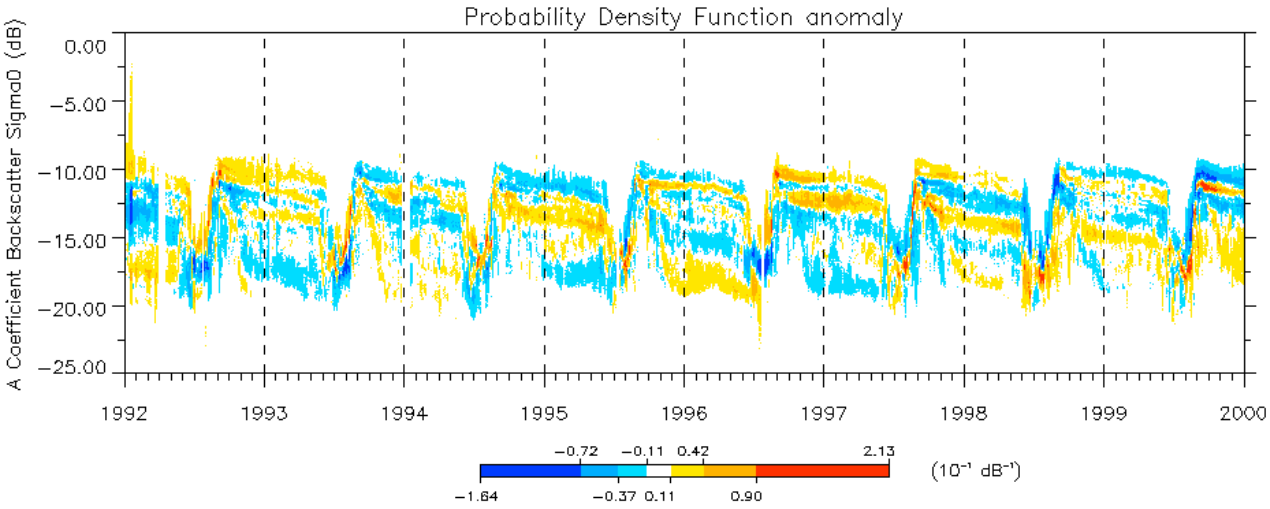
Region : Antarctic
Domain : Sea-ice
Type : a
S/a/m : signal
Statistics : Probability Density Function
Start year : 1992
Stop year : 1999
Sampling : 250



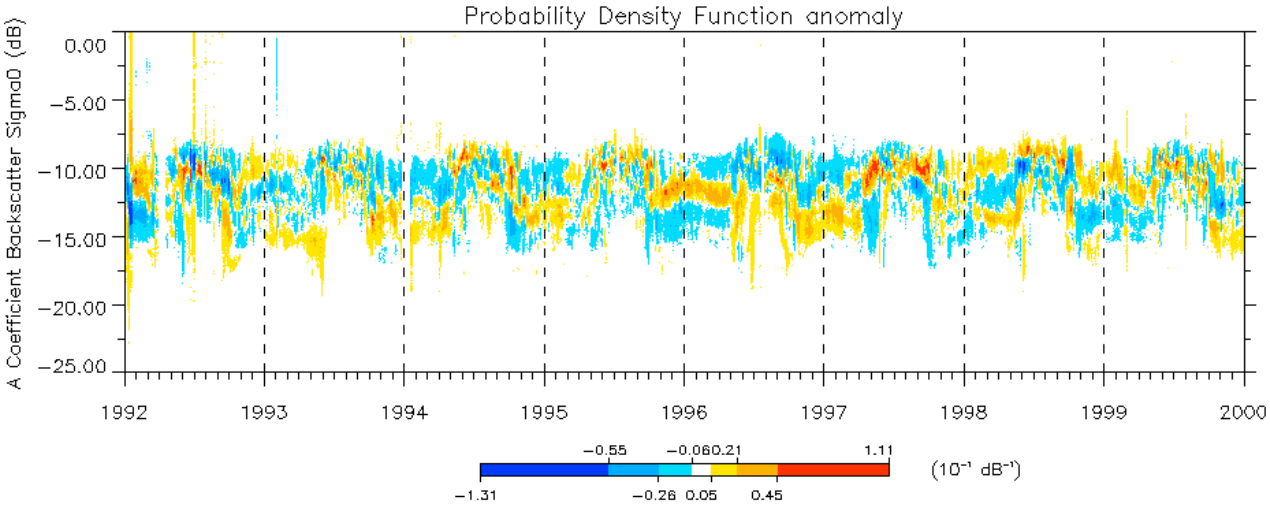
Region : Antarctic
Domain : Land(-ice)
Type : a
S/a/m : signal
Statistics : Probability Density Function
Start year : 1992
Stop year : 1999
Sampling : 250



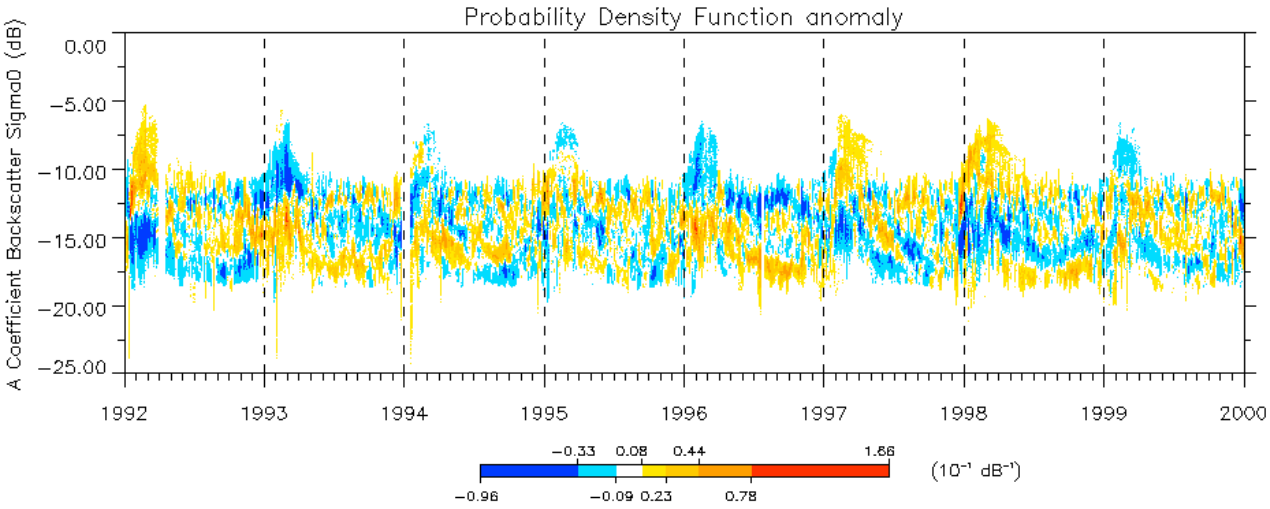
Region : Arctic
Domain : Sea-ice
Type : a
S/a/m : anomaly
Statistics : Probability Density Function
Start year : 1992
Stop year : 1999
Sampling : 250



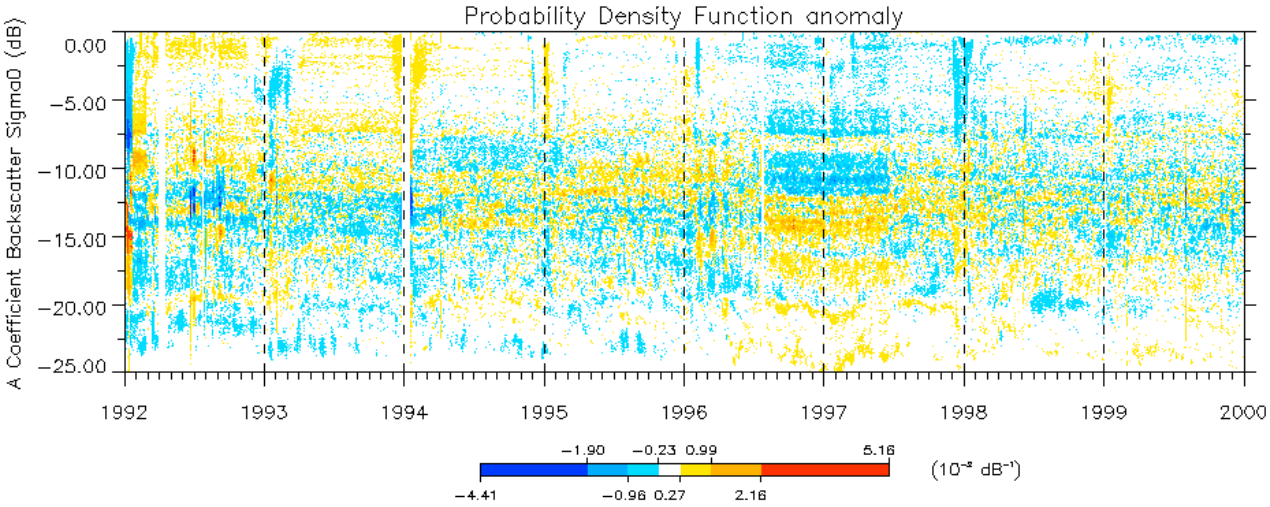
Region : Arctic
Domain : Land(-ice)
Type : a
S/a/m : anomaly
Statistics : Probability Density Function
Start year : 1992
Stop year : 1999
Sampling : 250



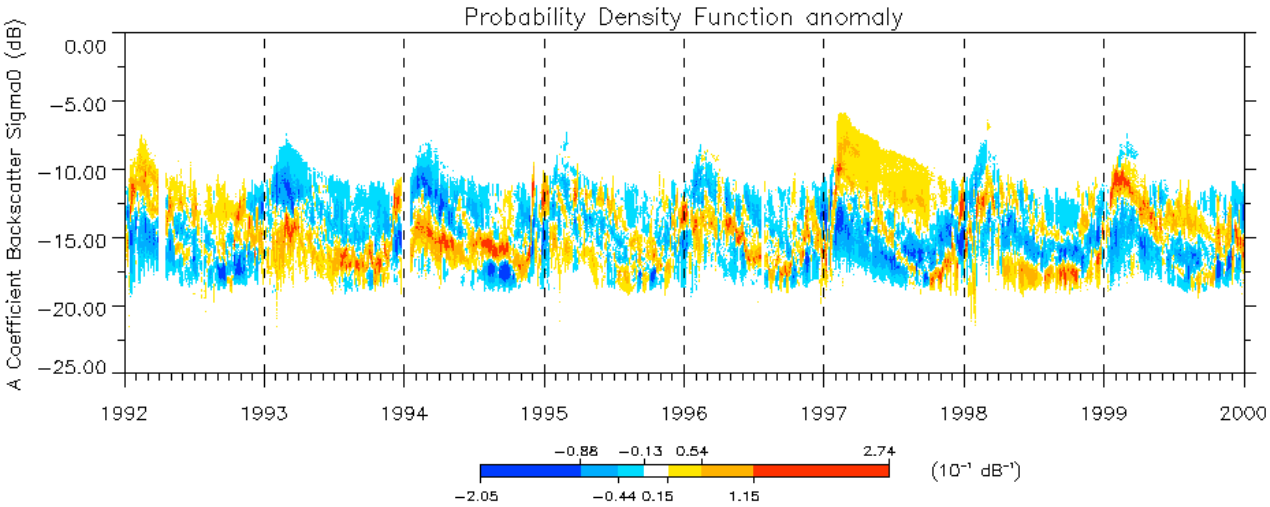
Region : Antarctic
Domain : Sea-ice
Type : σ
S/σ/m : anomaly
Statistics : Probability Density Function
Start year : 1992
Stop year : 1999
Sampling : 250



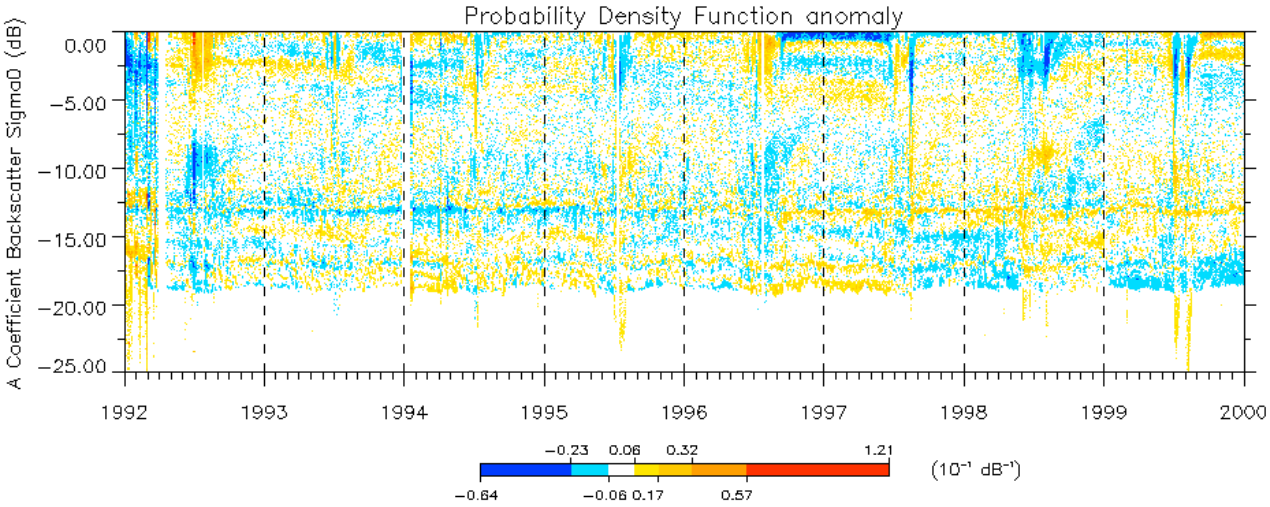
Region : Antarctic
Domain : Land(-ice)
Type : σ
S/σ/m : anomaly
Statistics : Probability Density Function
Start year : 1992
Stop year : 1999
Sampling : 250



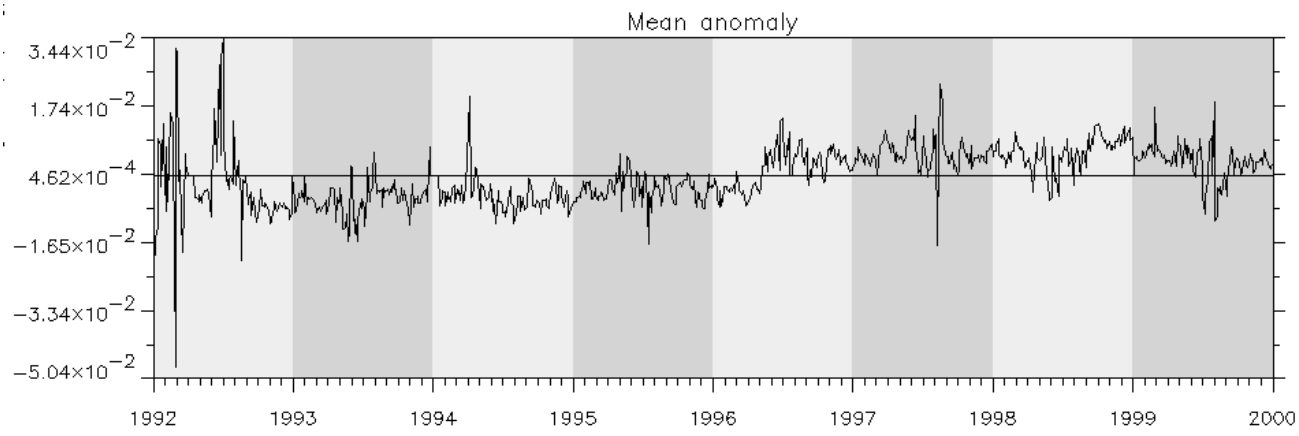
Region : Weddell_sea
Domain : Sea-ice
Type : a
S/a/m : anomaly
Statistics : Probability Density Function
Start year : 1992
Stop year : 1999
Sampling : 250



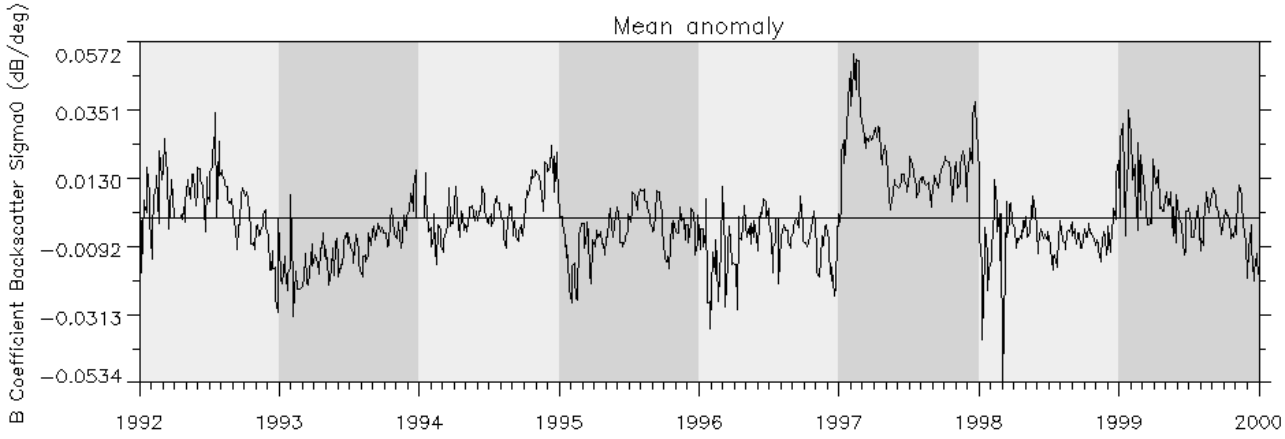
Region : Greenland
Domain : Land(-ice)
Type : a
S/a/m : anomaly
Statistics : Probability Density Function
Start year : 1992
Stop year : 1999
Sampling : 250



Region : Greenland
Domain : Land(−ice)
Type : b
S/a/m : anomaly
Statistics : Mean
Start year : 1992
Stop year : 1999
Sampling : 250



Region : Weddell_sea
Domain : Sea−ice
Type : b
S/a/m : anomaly
Statistics : Mean
Start year : 1992
Stop year : 1999
Sampling : 250



CONCLUSION

TOPICS provides an efficient tool to compute, manage and visualize statistical time series of the normalized cross-section backscatter for different ice domains in polar regions. It has the potential to reveal mid or long term trends as well as cycles in the variability of polar ice, and by allowing the user to define his own regions of investigations enables to localize quite precisely the areas where changes occur.

However, the plots obtained suggest that some changes between ERS-1 and ERS-2 instruments parameters were not taken into account when applying the SIRF algorithm to the EScat data set. This results in some artefacts in the plots which can bias the interpretations and surely makes them more difficult. Therefore, the next step would be to fix those problems so as to get good results that could be fully used.

But yet, the present first and raw version of TOPICS has yielded about four hundred and fifty different plots, most of which do not seem to be concerned by the previous problem. This represents a huge quantity of useful data that just wait to be analysed…

ANNEX :

SET OF COMMENTED ROUTINES

COMMON_USE_ROUTINES

```
;*****************************************************
;*****************************************************
;                        CONFIG.PRO
;              (SEE TOPICS REPORT FOR DETAILS)
;*****************************************************
;*****************************************************
;This routine contains nearly all configuration
;parameters of TOPICS; it may seem very long,
;but once you get used to it, and know how it is
;organized, you benefit from its centralizing TOPICS
;parameters. It is divided in 5 parts, corresponding
;to values of input 'key1':
;
;key1='structures': returns initialized TOPICS
;structure corresponding to parameters given in input
;
;key1='convert' : makes conversion between TOPICS
;data external and internal formats. Ex : The
;region Antarctic is represented by the string
;'Antarctic' outside TOPICS and by the byte 1b inside
;it. thus config('convert','region_long','Antarctic')
;returns 1b
;
;key1='paths' : returns path of file/directory
;corresponding to parameters given in input
;
;key1='data' : returns data about image size, pdf
;parameters, etc depending parameters given in input
;
;key1=interface : returns data needed to build interface
;Ex : list of years for years option menus
;
;
;Input: key1/2 :  strings that describe the kind of
;               result you expect from config.pro
;               datai: inputs needed to return wanted result
;               (figure it out from looking at the right
;               paragraph in the code below)
;
;Output: depends on inputs
;
;Calls: config.pro
;               read_file.pro
;               read_regions_equivalents.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
function config,key1,key2,data1,data2,data3,data4,data5

CASE key1 OF

;*************************************************
;*************************************************
;                        STRUCTURES
;*************************************************
;*************************************************
'structures': CASE key2 OF

;*******************************
;File header
;*******************************
'st1': result={structure1,what:0b,region:0b,type:0b,$
start_year:0,start_day:0,$
stop_year:0,stop_day:0,n_images:0,sampling:0}

;*******************************
;Stats header
;*******************************
'st2': result={structure2,image_available:0b,$
ice_contour_available:0b,region:0b,$
type:0b,year:0,day1:0,satellite:0b,n_avg:0b}

;*******************************
;Stats
;*******************************
'st3': result={,ice_extent:0.0,black_zones_ratio:1.0,$
pdf:fltarr(data1),mean:0.0,std:0.0,median:0.0}

;*******************************
;process_img.pro output
;*******************************
'st4sla': BEGIN
st2=config('structures','st2')
st3=config('structures','st3',data1)
result={,header:st2,si_data:st3,li_data:st3,ai_data:st3}
END

'st4s': BEGIN
st2=config('structures','st2')
st3=config('structures','st3',data1)
result={,header:st2,si_data:st3}
END

'st4l': BEGIN
st2=config('structures','st2')
st3=config('structures','st3',data1)
result={,header:st2,li_data:st3}
END

;*******************************
;
;*******************************
'st5': result={,image_available:fltarr(data2),$
ice_contour_available:fltarr(data2),$
black_zones_ratio:fltarr(data2)}

;*******************************
;variables_plot structure
;*******************************
'st6': result={structure6,region:0b,type:0b,domain:0b,$
stats_type:0b,start_year:0,start_day:0,stop_year:0,$
stop_day:0,sampling:0,sam:0b}

;*******************************
;
;data1 : sampling
;data2 : time_range
;data3 : stats dimension (1/2)
;*******************************
'st7': BEGIN
st5=config('structures','st5',data1,data2)
CASE data3 OF
1: result={,header:st5,stats:fltarr(data2)}
2: result={,header:st5,stats:fltarr(data2,data1)}
ENDCASE
END

;*******************************
;
;*******************************
'st8': result={structure8,pole:0b,region_number:0b,$
region_name:'',region_short_name:''}

;*******************************
; variables_stats structure
;*******************************
'st9': BEGIN
n_years=n_elements(config('interfaces','years_array'))
n_sampling=n_elements(config('interfaces',$
'sampling_array'))
result={structure9,what:0b,region:0b,type:0b,$
years:strarr(n_years),sampling:strarr(n_sampling)}
END

;*******************************
; variables_contours structure
;*******************************
'st10': BEGIN
n_years=n_elements(config('interfaces','years_array'))
result={structure10,years_arc:strarr(n_years),$
years_ant:strarr(n_years)}
END

;*******************************
; variables_mean_cycles structure
;*******************************
'st11': BEGIN
result={structure11,region:0b,type:0b,path:''}
END

;*********************
ENDCASE

;*************************************************
;*************************************************
;                        CONVERT
;*************************************************
;*************************************************

'convert': CASE key2 OF

;*****************
'what': BEGIN
;*****************
data=str(data1)
CASE data OF
'Sea/Land/All_Ice_Statistics': result=1b
'Sea_Ice_Statistics': result=2b
'Land_Ice_Statistics': result=3b
```

```
'Mean_Cycles': result=4b
'1': result='Sea/Land/All Ice Statistics'
'2': result='Sea Ice Statistics'
'3': result='Land Ice Statistics'
'4': result='Mean Cycles'
ENDCASE
END

;*****************
'region': BEGIN
;*****************
read_regions_equivalents,n,numbers,names,short_names
data=str(data1)
FOR i=0,n-1 DO $
CASE data OF
numbers(i): result=short_names(i)
short_names(i): result=numbers(i)
else:
ENDCASE
END

;**********************
'region_long': BEGIN
;**********************
read_regions_equivalents,n,numbers,names,short_names
data=str(data1)
FOR i=0,n-1 DO $
CASE data OF
numbers(i): result=names(i)
names(i): result=numbers(i)
else:
ENDCASE
END

;**********************
'type': BEGIN
;**********************
data=str(data1)
CASE data OF
'a': result=1
'b': result=2
'1': result='a'
'2': result='b'
ENDCASE
END

;**********************
'satellite': BEGIN
;**********************
data=str(data1)
CASE data OF
'ers1': result=1
'ers2': result=2
'1': result='ers1'
'2': result='ers2'
ENDCASE
END

;**************************
'domain': CASE data1 OF
;**************************
1: result='Sea-ice'
2: result='Land(-ice)'
3: result='All ice'
ENDCASE

;**************************
'domain_short': CASE data1 OF
;**************************
1: result='si'
2: result='li'
3: result='ai'
ENDCASE

;**************************
'stats_type': CASE data1 OF
;**************************
0: result='Ice Extent'
1: result='Black Zones Ratio'
2: result='Probability Density Function'
3: result='Mean'
4: result='Standard Deviation'
5: result='Median'
ENDCASE

;**************************
'stats_type_short': CASE data1 OF
;**************************
0: result=''
1: result=''
2: result='pdf'
3: result='mean'
4: result='std'
5: result='median'
ENDCASE


;**************************
'sam': CASE data1 OF
;**************************
1: result='signal'
2: result='anomaly'
3: result='mean cycle'
4: result='smoothed signal'
5: result='smoothed anomaly'
6: result='smoothed mean cycle'
ENDCASE

;*****************
'sam_short': BEGIN
;*****************
data=str(data1)
CASE data OF
'1': result='s'
'2': result='a'
'3': result='m'
'4': result='ss'
'5': result='sa'
'6': result='sm'
ENDCASE
END

;************************************
'units': CASE data1.stats_type OF
;************************************
0: result= 'Mkm2'
1: result= '-'
else : CASE data1.type OF
1: result= 'dB'
2: result= 'dB/deg'
ENDCASE
ENDCASE

;************************************
'title': BEGIN
;************************************
result=config('convert','stats_type',data1.stats_type)+$
' '+config('convert','sam',data1.sam)
END

;************************************
'ytitle': CASE data1.stats_type OF
;************************************
0:result=config('convert','stats_type',data1.stats_type)+$
' '+config('convert','sam',data1.sam)+' '+$
config('convert','units',data1)
1:result=config('convert','stats_type',data1.stats_type)+$
' '+config('convert','sam',data1.sam)+' '+$
config('convert','units',data1)
else: result=strupcase(config('convert','type',data1.type))+$
' Coefficient Backscatter Sigma0 '+'('+$
config('convert','units',data1)+')'
ENDCASE

;************************************************
ENDCASE

;************************************************
;************************************************
;                                PATHS
;************************************************
;************************************************
'paths': BEGIN

COMMON topics_directory_path,topics_directory_path

CASE key2 OF

;*******************************
'extent_grid': BEGIN
;*******************************
result=topics_directory_path+$
'/PROGRAM_DATA/GENERAL_MASKS_AND_GRIDS/EXTENT_GRIDS/'+$
'extent_grid_'+config('convert','region',data1)+'.bin'
END

;*******************************
'circle_sirf': BEGIN
;*******************************
result=topics_directory_path+$
'/PROGRAM_DATA/GENERAL_MASKS_AND_GRIDS/CIRCLE_MASKS/'+$
'circle_sirf_'+config('convert','region',data1)+'.bin'
END

;*******************************
'middle_circle_sirf': BEGIN
```

```
;*******************************
result=topics_directory_path+$
'/PROGRAM_DATA/GENERAL_MASKS_AND_GRIDS/CIRCLE_MASKS/'+$
'middle_circle_sirf_'+config('convert','region',data1)+'.bin'
END

;*******************************
'tmp_dir': BEGIN
;*******************************
result=topics_directory_path+'/PROGRAM_DATA/TMP'
END

;*******************************
'image': BEGIN
;*******************************
header=data1

region=config('convert','region',2-(header.region mod 2))
type=config('convert','type',fix(header.type))
year=str(header.year)
day1=header.day1
satellite=config('convert','satellite',$
fix(header.satellite))
n_avg=header.n_avg

beg_day_str=str(day1)
if (strlen(beg_day_str) eq 1) then $
beg_day_str='00'+beg_day_str
if (strlen(beg_day_str) eq 2) then $
beg_day_str='0'+beg_day_str

END_day_str=str((day1+n_avg-1) mod (365+is_leap(year)+$
(year eq '1998' or year eq '1999')))
if (strlen(end_day_str) eq 1) then $
end_day_str='00'+end_day_str
if (strlen(end_day_str) eq 2) then $
end_day_str='0'+end_day_str

result='/rime2/escat/'+satellite+'-'+type+$
'-'+year+'/'+satellite+'-'+$
type+'-'+region+strmid(year,2,2)+'-'+$
beg_day_str+'-'+end_day_str+'.sir'

END

;*********************
'contours_bank': BEGIN
;*********************
result=topics_directory_path+$
'/PROGRAM_DATA/CONTOURS_BANK'
END

;*******************************
'ice_contour': BEGIN
;*******************************
pole=config('convert','region_long',data1)
year=str(data2)
day1=data3

result=config('paths','contours_bank')+'/'+$
strupcase(pole)+'/'+year+'/'+'con_'+$
strmid(number_to_date(1000l*year+day1),2,6)
END

;*************************
'land_mask': BEGIN
;*************************
region=strlowcase(config('convert','region',data1))

result=topics_directory_path+$
'/PROGRAM_DATA/GENERAL_MASKS_AND_GRIDS/LAND_MASKS/'+$
region+'_land_mask.bin'

END

;*************************
'census': BEGIN
;*************************
region=data1
type=data2

result=topics_directory_path+$
'/STATS_BANK/'+config('convert','region_long',region)+$
'/'+config('convert','type',fix(type))+'/Census_'+$
config('convert','region',region)+'-'+$
config('convert','type',fix(type))

END

;************************
'stats_filename': BEGIN
;************************
region=config('convert','region',data1)
type=config('convert','type',fix(data2))
year=data3
sampling=str(data4)

result=region+'-'+type+'-'+year+'-'+sampling+'.stats'

END

;************************
'stats': BEGIN
;************************
region_long=config('convert','region_long',data1)
type=config('convert','type',fix(data2))
year=data3

result=topics_directory_path+'/STATS_BANK/'+$
region_long+'/'+type+'/'+year+'/'+$
config('paths','stats_filename',data1,data2,data3,data4)

END

;************************
'mean_filename': BEGIN
;************************
region=config('convert','region',data1)
type=config('convert','type',fix(data2))
years_used=data3
sampling=str(data4)

period=''
FOR i=0,n_elements(years_used)-1 DO BEGIN
yy=strmid(years_used(i),2,2)
if (yy ne '') then period=period+yy+'-'
ENDFOR

result=region+'-'+type+'-'+period+sampling+'.mean'

END

;********************
'mean': BEGIN
;********************
region_long=config('convert','region_long',data1)
type=config('convert','type',fix(data2))
result=config('paths','mean_cycle_dir',data1,data2)+$
config('paths','mean_filename',data1,data2,data3,data4)
END

;********************
'mean_cycle_dir': BEGIN
;********************
region_long=config('convert','region_long',data1)
type=config('convert','type',fix(data2))
result=topics_directory_path+$
'/STATS_BANK/'+region_long+'/'+type+'/mean_cycles/'
END

;********************
'mean_cycles_selected': BEGIN
;********************
result=topics_directory_path+$
'/PROGRAM_DATA/MEAN_CYCLES_SELECTED/'+$
'mean_cycles_selected.dat'
END

;********************
'mean_cycles_selected_default': BEGIN
;********************
result=topics_directory_path+$
'/PROGRAM_DATA/MEAN_CYCLES_SELECTED/'+$
'mean_cycles_selected_default.dat'
END

;********************
'mean_selected': BEGIN
;********************
region=data1
type=config('convert','type',fix(data2))
path=config('paths','mean_cycles_selected')

read_file,path,$
'mean_cycles_selected',header,mean_cycles_selected

read_regions_equivalents,n,numbers,names,short_names
w=where(numbers eq region)

COMMON topics_directory_path,topics_directory_path
result=topics_directory_path+$
mean_cycles_selected(2*w(0)+data2-1)

END

;*****************
```

```
'tiff_save': BEGIN
;*****************
region_long=config('convert','region_long',data1.region)
type=config('convert','type',data1.type)
region=config('convert','region',data1.region)
domain_short=config('convert','domain_short',data1.domain)
stats_type_short=$
config('convert','stats_type_short',data1.stats_type)
sam_short=config('convert','sam_short',data1.sam)

result=topics_directory_path+$
'/PICS_LIBRARY/TIFFS/'+region_long+'/'+$
region+'_'+type+'_'+domain_short+'_'+stats_type_short+$
'_'+sam_short+'.tif'

END

;*****************
'plot': BEGIN
;*****************
result=topics_directory_path+$
'/PROGRAM_DATA/PLOT_ON_DISPLAY/plot_on_display.dat'
END

;*****************
'cover_image': BEGIN
;*****************
result=topics_directory_path+$
'/PROGRAM_DATA/COVER_IMAGE/cover_image.tif'
END

;*****************
'regions_equivalents': BEGIN
;*****************
result=topics_directory_path+$
'/PROGRAM_DATA/CUSTOM_REGIONS_EQUIVALENCE_TABLE/'+$
'regions_equivalents.dat'
END

;*********************
'region_contour_and_mask': BEGIN
;*********************
result=topics_directory_path+$
'/PROGRAM_DATA/CUSTOM_REGIONS_MASKS/'+$
config('convert','region_long',data1)+'.bin'
END

;*********************
'session': BEGIN
;*********************
result=topics_directory_path+$
'/PROGRAM_DATA/SESSION/wavesave.pro'
END

;*********************
'info_cover': BEGIN
;*********************
result=topics_directory_path+$
'/PROGRAM_DATA/INFO_COVER/info_cover.txt'
END

;*********************
'ij_ll_grids': BEGIN
;*********************
result=topics_directory_path+$
'/PROGRAM_DATA/GENERAL_MASKS_AND_GRIDS/IJ_LL_GRIDS/'+$
data1+'_ssmi25_'+data2
END


;*********************
'ice_concentration': BEGIN
;*********************
result='/rime2/ssmi/conc/Daily/'
END

;**********************************
ENDCASE

;*********************************************
END

;*************************************************
;*************************************************
;                                  DATA
;*************************************************
;*************************************************

'data': CASE key2 OF

;*************************************
'computation_times_stats': BEGIN
;*************************************
result=[16.5,5.5,5.5]
END


;*************************************
'pdf_params': BEGIN
;*************************************
region=config('convert','region_long',data1)
type=config('convert','type',fix(data2))
CASE (region+' '+type) OF
'Arctic a': result=[-29767,232,-33,1000]
'Arctic b':result=[-12767,-2768,-3,10000]
'Antarctic a': result=[-29767,232,-33,1000]
'Antarctic b': result=[-12767,-2768,-3,10000]
ENDCASE
END

;*****************************************
'pdf_artefacts': BEGIN
;*****************************************
region=config('convert','region_long',data1)
type=config('convert','type',fix(data2))
CASE (region+' '+type) OF
'Arctic a': result=[[-29767,-29764]]
'Arctic b': result=[[-4070,-4065]]
'Antarctic a': result=[[-29767,-29764]]
'Antarctic b': result=[[-4070,-4065]]
ENDCASE
END

;*****************************************
'pdf_range' : BEGIN
;*****************************************
region=config('convert','region_long',data1)
type=config('convert','type',fix(data2))
CASE (region+' '+type) OF
'Arctic a': result=[-25.000,-0.001]
'Arctic b': result=[-0.5000,-0.0001]
'Antarctic a': result=[-25.000,-0.001]
'Antarctic b': result=[-0.5000,-0.0001]
ENDCASE
END

;*****************************************
'image_size': BEGIN
;*****************************************
region=config('convert','region_long',data1)
CASE region OF
'Arctic': result=[770,770]
'Antarctic': result=[970,970]
ENDCASE

END

;*****************************************
'preference': BEGIN
;*****************************************
result=[[2,6],[2,7],[1,6],[1,7]]
END

;*****************************************
'image_ll': BEGIN
;*****************************************
CASE config('convert','region',data1) OF
'Arc': result=[60.0,90.0,-180.0,180.0]
'Ant': result=[-52.0,-90.0,-180.0,180.0]
ENDCASE
END

;*****************************************
'dim_ssmi': BEGIN
;*****************************************
CASE config('convert','region',data1) OF
'Arc': result=[304,448]
'Ant': result=[316,332]
ENDCASE
END

;*****************************************
'badvalues_ssmi': BEGIN
;*****************************************
CASE config('convert','region',data1) OF
'Arc': result=[157,-99]
'Ant': result=[157,-99]
ENDCASE
END

;*****************************************
'exclude': BEGIN
;*****************************************
CASE config('convert','region',data1) OF
'Arc': result=[80,50]
'Ant': result=[80,50]
```

```
ENDCASE
END


;*****************************************
'c_level': BEGIN
;*****************************************
CASE config('convert','region',data1) OF
'Arc': result=15.
'Ant': result=15.
ENDCASE
END

;*****************************************
'cutoff': BEGIN
;*****************************************
CASE config('convert','region',data1) OF
'Arc': result=50
'Ant': result=800
ENDCASE
END

;*****************************************
ENDCASE

;*************************************************
;*************************************************
;                    INTERFACES
;*************************************************
;*************************************************

'interfaces': CASE key2 OF

;*****************************************
'years_array' : $
;*****************************************
result=['1992','1993','1994','1995','1996',$
'1997','1998','1999',$
'2000','2001','2002','2003','2004','2005']

;*****************************************
'sampling_array': $
;*****************************************
result=['250','500','1000','2000','5000']

;*****************************************
ENDCASE

;*************************************************
ENDCASE

;*************************************************
return,result

END


;*************************************************
;*************************************************
;            DATE_TO_NUMBER.PRO
;*************************************************
;*************************************************
;This routine transforms a yyyymmdd format date into
;yyyyddd format date (converse of number_to_date.pro)
;
;Input: number-format string ('yyyyddd')
;
;Output: date-format string ('yyyymmdd')
;
;Calls:      is_leap.pro
;            str.pro
;
;Written by/in: Pierre Mercier 04/00
;
;Last updated: 08/31/00
;*******************************************************
;*******************************************************
FUNCTION date_to_number,date_in

;*****************************************
date=long(date_in)

;*****************************************
yyyy=date/10000
mm=(date mod 10000)/100
dd=(date mod 100)

;*****************************************
IF is_leap(yyyy) $
THEN tab_month=[-1,30,59,90,120,151,181,212,243,273,$
304,334] ELSE tab_month=[-1,30,58,89,119,150,180,$
211,242,272,303,333]

;*****************************************
number=str(tab_month(mm-1)+dd)
IF (strlen(number) eq 1) THEN number ='00'+number
IF (strlen(number) eq 2) THEN number ='0'+number

;*****************************************
yyyy=str(yyyy)
number=yyyy+number

;*****************************************
RETURN,number

;*************************************************
END

;*************************************************
;*************************************************
;                    IS_LEAP.PRO
;*************************************************
;*************************************************
;This routine determines if a year is leap or not
;
;Input: 4 digit year - string or any numerical type
;
;Output: 0b if not leap, 1b otherwise
;
;Calls:      none
;
;Written by/in: Pierre Mercier 04/00
;
;Last updated: 08/31/00
;*******************************************************
;*******************************************************
FUNCTION is_leap,year

;*****************************************
yyyy=fix(year)
boolean=0b

;*****************************************
;A year is leap if and only if (it is a multiple of
;4 and not a multiple of 100) or (it is a multiple
;of 400)
;*****************************************
IF (((yyyy mod 400) eq 0) or (((yyyy mod 100) ne 0) $
and ((yyyy mod 4) eq 0))) THEN boolean=1b

;*****************************************
RETURN, boolean

;*************************************************
END


;*************************************************
;*************************************************
;                LL_TO_SIRF.PRO
;*************************************************
;*************************************************
;This routine converts latitude,longitude coordinates
;into i,j coordinates for sirf images
;
;Input: pole: 1=Arctic, 2=Antarctic
;            lat_deg : latitude in degrees
;            lon_deg : longitude in degrees
;
;Output: i_sirf
;             j_sirf
;
;Calls:      config.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*******************************************************
;*******************************************************
PRO ll_to_sirf,pole,lat_deg,lon_deg,i_sirf,j_sirf

;*******************************************************
;MAKES USEFUL CONSTANTS
;*******************************************************
pi=!dpi
d2r=pi/180

;*******************************************************
;CONVERTS LAT,LON INTO RADIANS
;*******************************************************
lat=abs(lat_deg*d2r)
lon=lon_deg*d2r

;*******************************************************
;LOADS USEFUL PARAMETERS
;*******************************************************
image_ll=config('data','image_ll',pole)
```

```
lat0=abs(image_ll(0))*d2r
image_size=config('data','image_size',pole)
n=image_size(0)

;*****************************************************
;TAKES INTO ACCOUNT THE FACT THAT ARCTIC SIRF IMAGES
;ARE 3PI/4 EASTWARD ROTATED
;*****************************************************
IF pole eq 1 THEN lon=3*pi/4-lon

;*****************************************************
;MAKES THE CONVERSION
;*****************************************************
i_sirf=nint((n-1)*(0.5+tan(pi/4.0-lat/2.0)*$
sin(lon)/(2.0*tan(pi/4.0-lat0/2.0))))
j_sirf=nint((n-1)*(0.5+tan(pi/4.0-lat/2.0)*$
cos(lon)/(2.0*tan(pi/4.0-lat0/2.0))))

;*****************************************************
END


;*****************************************************
;*****************************************************
;          MAKE_CIRCLE_SIRF.PRO
;*****************************************************
;*****************************************************
;This routine makes a file containing the indexes of
;points of a .sir image which are located in the
;useful part (in the circle area, not outside it)
;
;Input: pole : 1=Arctic, 2=Antarctic
;
;Output: file described above
;
;Calls:          config.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
PRO make_circle_sirf,pole

;*****************************************
;LOADS SIZE OF .SIR IMAGE FOR GIVEN POLE
;*****************************************
image_size=config('data','image_size',pole)
n=image_size(0)

;*****************************************
;FINDS INDEXES OF POINTS INSIDE USEFUL CIRCLE
;*****************************************
array=bytarr(n,n)
x_center=0.5*(n-1)
y_center=0.5*(n-1)
r0=0.5*(n-1)
FOR i=0,n-1 do begin
FOR j=0,n-1 do begin
r=sqrt((i-x_center)^2+(j-y_center)^2)
IF (r le r0) THEN array(i,j)=1b
ENDFOR
ENDFOR
w=where(array eq 1b)

;*****************************************
;WRITES INDEXES TO FILE SPECIFIED
;BY CONFIG.PRO
;*****************************************
path=config('paths','circle_sirf',pole)
openw,unit,path,/get_lun
writeu,unit,n_elements(w)
writeu,unit,w
free_lun,unit

;*****************************************
END


;*****************************************************
;*****************************************************
;          MAKE_DIRS_TO_FILE.PRO
;*****************************************************
;*****************************************************
;This routine creates (if they don't already exist)
;directories which occur in the specified file path
;It was written to prevent the user from having
;to create himself the 'complex' organisation of
;directories in STATS/IMAGES/CONTOURS BANKS
;
;Since TOPICS takes some space when running, it is
;possible that calling this routine on a 64 MB or so
;machine will make the program crash (because it
;calls child processes which need space). In that ;case,the user should create the needed
;directories ;himself and rerun the program.
;
;Input: path of file FROM ROOT (string)
;
;Output: creation of directories as explained above
;
;Calls:          config.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
PRO make_dirs_to_file,file_path_from_root

;****************************************
;MAKES TEMPORARY FILE CONTAINING
;ALL TOPICS DIRECTORY SUBDIRECTORIES
;****************************************
COMMON topics_directory_path,topics_directory_path
path=config('paths','tmp_dir')+'/archi.tmp'

spawn,'ls -Rp '+topics_directory_path+' >'+path

;****************************************
;RETRIEVE CONTENT OF THAT FILE,PUTS IT
;INTO WAVE VARIABLE DIRS
;AND DELETE THE FILE
;****************************************
dirs=['']
dir=''
openr,unit,path,/get_lun,/delete
WHILE(not(eof(unit))) DO BEGIN
readf,unit,dir
IF (strmid(dir,strlen(dir)-1,1) eq ':') THEN dirs=$
[dirs,strmid(dir,0,strlen(dir)-1)]
ENDWHILE
free_lun,unit

dirs=dirs(1:*)

;****************************************
;MAKES PATHS FROM ROOT OF DIRECTORIES
;APPEARING IN PATH OF FILE SPECIFIED IN ENTRY
;****************************************
dirs_array=strsplit(file_path_from_root,'/')
dirs_array=dirs_array(1:n_elements(dirs_array)-2)
n=n_elements(dirs_array)
cumul_dir_array=strarr(n)
cumul_dir_array(0)='/'+dirs_array(0)
FOR i=1,n-1 DO $
cumul_dir_array(i)=cumul_dir_array(i-1)+'/'+dirs_array(i)


;****************************************
;CREATES DIRECTORIES APPEARING IN PATH
;OF FILE SPECIFIED IN ENTRY IF THEY
;DON'T ALREADY EXIST
;****************************************
start=where(cumul_dir_array eq topics_directory_path,count)
IF (count ne 0) THEN BEGIN
FOR i=start(0)+1,n-1 DO BEGIN
w=where(dirs eq cumul_dir_array(i),count)
IF (count eq 0) THEN spawn,'mkdir '+cumul_dir_array(i)
ENDFOR

;****************************************
ENDIF ELSE print,'Problem'

;****************************************
END


;*****************************************************
;*****************************************************
;                NUMBER_TO_DATE.PRO
;*****************************************************
;*****************************************************
;This routine transforms a yyyyddd format date into a
;yyyymmdd format date(converse of date_to_number.pro)
;
;Input: number-format string ('yyyyddd')
;
;Output: date-format string ('yyyymmdd')
;
;Calls:          is_leap.pro
;                str.pro
;
;Written by/in: Pierre Mercier 04/00
;
;Last updated: 08/31/00
;*****************************************************
```

```
;*************************************************
FUNCTION number_to_date,number_in


;*****************************************
number=long(number_in)

;*****************************************
yyyy=number/1000
ddd=number mod 1000

;*****************************************
IF is_leap(yyyy) THEN $
tab_month=[0,31,60,91,121,152,182,213,244,274,305,$
335,366] ELSE tab_month=[0,31,59,90,120,151,181,212,$
243,273,304,334,365]

;*****************************************
w=reverse(where((ddd-tab_month) ge 0))
month=w(0)+1
day=ddd-tab_month(w(0))+1

;*****************************************
yyyy=str(yyyy)
mm=str(month)
dd=str(day)
IF (strlen(mm) eq 1) THEN mm='0'+mm
IF (strlen(dd) eq 1) THEN dd='0'+dd

;*****************************************
date=yyyy+mm+dd

;*****************************************
RETURN, date

;*************************************************
END


;*************************************************
;*************************************************
;                    READ_FILE.PRO
;*************************************************
;*************************************************
;This routine reads the file whose 'type' is specified
;by the 'key' input (see comments in program).
;
;Ex: read_file,'...','stats',.. reads a file which
;contains statistics
;
;The structure of data in the file can be deduced
;from the code below (too long to describe here)
;or by looking at TOPICS report
;
;Input: file_path: file path from root (string)
;       key: string describing file content
;
;Output: file_header: file header (usually structure)
;        file_content: file content (usually structure)
;        file_content_bis: part II of file content, if
;        apply (usually structure)
;
;Calls:      config.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*************************************************
;*************************************************
PRO read_file,file_path,key,file_header,$
file_content,file_content_bis

CASE key OF

;*****************************************
;               'LAND_MASK'
;
;READS FILE CONTAINING LAND_MASK OF
;ARCTIC/ANTARCTIC
;*****************************************
'land_mask': BEGIN
openr,unit,file_path,/get_lun
count=0l
readu,unit,count
file_header=count
struc=assoc(unit,lonarr(count))
file_content=struc(0)
free_lun,unit
END

;*****************************************
;               'MEAN_CYCLES_SELECTED'
;
;READS FILE CONTAINING THE PATHS
;OF MEAN CYCLES FILES USED TO COMPUTE
;ANOMALIES
;*****************************************
'mean_cycles_selected': BEGIN
count=0l
file_header=0b
openr,unit,file_path,/get_lun
readf,unit,count
file_content=strarr(count)
path=''
FOR i=0,count-1 DO BEGIN
readf,unit,path
file_content(i)=path
ENDFOR
free_lun,unit
END

;*****************************************
;               'INFO_COVER'
;
;READS TEXT FILE CONTAINING INFORMATION
;ABOUT TOPICS
;*****************************************
'info_cover': BEGIN
file_header=0b
file_content=['']
text_line=''
openr,unit,file_path,/get_lun
WHILE not(eof(unit)) DO BEGIN
readf,unit,text_line
file_content=[file_content,text_line]
ENDWHILE
free_lun,unit
file_content=file_content(1:*)
END


;*****************************************
;               'CENSUS'
;
;READS AN IMAGE BANK CENSUS FILE
;*****************************************
'census': BEGIN
add_content=config('structures','st2')
file_content=[add_content]
openr,unit,file_path,/get_lun,error=err
IF (err ne 0) THEN BEGIN
COMMON mw_menu,mw_menu
census_al=wwalert(mw_menu,['Images Bank Census Missing',$
'Taking it ...'],/working,/noconfirm,title='ERROR',after=4)
make_census
wwalertpopdown,census_al
read_file,file_path,key,file_header,file_content
ENDIF ELSE BEGIN
WHILE not(eof(unit)) DO BEGIN
readu,unit,add_content
file_content=[file_content,add_content]
ENDWHILE
close,unit
file_header=0b
file_content=file_content(1:*)
ENDELSE
END

;*****************************************
;               'REGION_CONTOUR_AND_MASK'
;
;READS FILE CONTAINING THE CONTOUR
;AND THE MASK OF A CUSTOM REGION
;*****************************************
'region_contour_and_mask': BEGIN
n_cont=0l
n_mask=0l
openr,unit,file_path,/get_lun
readu,unit,n_cont
file_content=lonarr(n_cont/2,2)
readu,unit,file_content
readu,unit,n_mask
file_header=[n_cont,n_mask]
file_content_bis=lonarr(n_mask)
readu,unit,file_content_bis
free_lun,unit
END

;*****************************************
;               '.SIR'
;
;READS A .SIR IMAGE (INSPIRED AND SIMPLIFIED
;VERSION OF M. DRINKWATER'S LOADSIR.PRO)
;DO NOT CONVERT VALUES TO FLOAT.
;KEEP THEM AS 2B INTEGERS
```

```
;*****************************************
'.sir': BEGIN

openr,unit,file_path,/get_lun

;*******************************
; read first header
;*******************************
file_header=intarr(256,/Nozero)
readu,unit,file_header

;*******************************
nhtype = file_header(4)
nheader = file_header(40)

;*******************************
; override extra header blocks
; for old header
;*******************************
IF nhtype lt 20 THEN nheader = 1

;*******************************
; skip extra header blocks
;*******************************
header=intarr(256,/Nozero)
FOR i=1,nheader-1 DO readu,unit,header

;*******************************
;read image
;*******************************
file_content=intarr(file_header(0),$
file_header(1),/Nozero)
readu,unit,file_content

free_lun,unit

END


;*****************************************
;                                'STATS'
;
;READS A FILE CONTAINING STATISTICS
;*****************************************
'stats': BEGIN

openr,unit,file_path,/get_lun

file_header=config('structures','st1')
readu,unit,file_header

st4sla=config('structures','st4sla',file_header.sampling)
file_content=replicate(st4sla,file_header.n_images)

CASE file_header.what OF
2: BEGIN
st4s=config('structures','st4s',file_header.sampling)
file_content0=replicate(st4s,file_header.n_images)
readu,unit,file_content0
file_content.header=file_content0.header
file_content.si_data=$
rename_structure(file_content0.si_data,$
file_content.si_data)
END
3: BEGIN
st4l=config('structures','st4l',file_header.sampling)
file_content0=replicate(st4l,file_header.n_images)
readu,unit,file_content0
file_content.header=file_content0.header
file_content.li_data=$
rename_structure(file_content0.li_data,$
file_content.li_data)
END
ELSE: readu,unit,file_content
ENDCASE

free_lun,unit

END

;*****************************************
;                                'PLOT'
;
;READS A FILE CONTAINING A STATISTICS
;PLOT
;*****************************************
'plot': BEGIN
openr,unit,file_path,/get_lun
sampling=0
time_range=0
dim_stats=0
readu,unit,sampling,time_range,dim_stats
file_header=[sampling,time_range,dim_stats]
file_content=config('structures','st7',sampling,$
time_range,dim_stats)
readu,unit,file_content
free_lun,unit
END

;*****************************************
ENDCASE

;*********************************************
END


;*************************************************
;*************************************************
; READ_REGIONS_EQUIVALENTS.PRO
;*************************************************
;*************************************************
;This routine reads the file containing the equivalence
;table between region number, region name and
;region short name .
;
;Rmk: should be integrated into read_file.pro so that
;read_regions_equivalents=read_file('equivalents',...)
;
;Input: none
;
;Output: n : number of regions (long)
;                numbers : string array of regions numbers
;                names : string array of regions names
;                short names : string array of
;                        regions short names
;
;Calls:          config.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*************************************************
;*************************************************
PRO read_regions_equivalents,n,numbers,names,short_names

;*****************************************
;LOADS PATH TO EQUIVALENCE TABLE
;*****************************************
path=config('paths','regions_equivalents')

;*****************************************
;INITIALIZES VARIABLES TO BE READ
;*****************************************
numbers=['']
names=['']
short_names=['']
number=''
name=''
short_name=''
blank=''

;*****************************************
;READS FILE
;*****************************************
openr,unit,path,/get_lun
WHILE not(eof(unit)) do begin
readf,unit,number
readf,unit,name
readf,unit,short_name
readf,unit,blank
numbers=[numbers,number]
names=[names,name]
short_names=[short_names,short_name]
ENDWHILE
free_lun,unit

;*****************************************
;RESIZES ARRAYS (FIRST VALUE NOT WANTED)
;*****************************************
numbers=numbers(1:*)
names=names(1:*)
short_names=short_names(1:*)

;*****************************************
;COMPUTES NUMBER OF REGIONS
;*****************************************
n=n_elements(numbers)

;*****************************************
END


;*************************************************
;*************************************************
;      RENAME_STRUCTURE.PRO
;*************************************************
```

```
;*****************************************************
;This routine creates a copy of a structure,
;changing its reference ('name') to that of
;a compatible structure given in entry
;
;EX: if a={,last_name:'Mark',age:43} is a $1 structure
;and b={,last_name:'George',age:32} is a $2 structure
;then you can't do a=b because the structures
;references are different. This is made possible by doing
;a=rename_structure(b,a) because rename_structure(b,a)
;is a $1 structure with b's content
;
;Input: content_structure : structure
;                name_structure : structure with same
;                content type as content structure
;
;Output: structure whose reference is that of
;                 name_structure and whose content is
;                 that of content_structure
;
;Calls:          none
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
FUNCTION rename_structure,content_structure,name_structure

;****************************************
result=name_structure
FOR i=0,n_tags(result)-1 DO BEGIN
s=size(content_structure.(i))
IF (n_elements(s) eq 4 and s(2) eq 8) THEN $
result.(i)=rename_structure(content_structure.(i),result.(i)) $
ELSE result.(i)=content_structure.(i)
ENDFOR

;****************************************
RETURN,result

;****************************************
END



;*****************************************************
;*****************************************************
;                    SIRF_TO_LL.PRO
;*****************************************************
;*****************************************************
;This routine converts i,j sirf coordinates to
;latitude,longitude coordinates
;
;Input: region : topics region number
;                i,j : sirf coordinates
;
;Output: lat_deg : latitude in degrees
;                 lon_deg : longitude in degrees
;
;Calls:          config.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
PRO sirf_to_ll,region,i,j,lat_deg,lon_deg

;*****************************************************
;GETS POLAR REGION NUMBER
;*****************************************************
pole=2-(region mod 2)

;*****************************************************
;MAKES USEFUL CONSTANTS
;*****************************************************
pi=!dpi
d2r=pi/180

;*****************************************************
;LOADS USEFUL PARAMETERS
;*****************************************************
image_ll=config('data','image_ll',pole)
lat0=abs(image_ll(0))*d2r
image_size=config('data','image_size',pole)
n=image_size(0)

;*****************************************************
;COMPUTES TERMS IN CONVERSION FORMULAS
;AND THEN EXAMINES DIFFERENT CASES
;*****************************************************
coef=(2.0*tan(pi/4.0-lat0/2.0))
tansin=(i/float(n-1))-0.5
tancos=(j/float(n-1))-0.5

;*************************************
;                    SIN EQ 0
;*************************************
IF ((tansin eq 0) and (tancos ne 0)) THEN BEGIN
lon_deg=180.0
lat_deg=2.*(pi/4.-atan(coef*tancos))/d2r
ENDIF

;*************************************
;                    TAN EQ 0
;*************************************
IF ((tansin eq 0) and (tancos eq 0)) THEN BEGIN
lat_deg=90.0
lon_deg=0.0
ENDIF

;*************************************
;   TAN NE 0, COS NE 0, SIN NE 0
;*************************************
IF ((tansin ne 0) and (tancos ne 0)) THEN BEGIN
lon_deg=atan(tansin/tancos)/d2r
lat_deg=2.*(pi/4.-atan(coef*tansin/sin(lon_deg*d2r)))/d2r
ENDIF

;*************************************
;                    COS EQ 0
;*************************************
IF ((tansin ne 0) and (tancos eq 0)) THEN BEGIN
lon_deg=90.0
lat_deg=2.*(pi/4.-atan(coef*tansin))/d2r
ENDIF

;*****************************************************
;TAKES INTO ACCOUNT THE FACT THAT ARCTIC SIRF IMAGES
;ARE 3PI/4 EASTWARD ROTATED
;*****************************************************
IF (pole eq 1) THEN lon_deg=135.-lon_deg

;*****************************************************
;SHIFTS RESULTS INTO GOOD RANGE (MODULO ISSUE)
;*****************************************************
IF (lat_deg gt 90.0) THEN BEGIN
lat_deg=180.-lat_deg
lon_deg=180.+lon_deg
ENDIF

IF (lon_deg gt 180.0) THEN lon_deg=lon_deg-360.
IF (lon_deg lt -180.0) THEN lon_deg=lon_deg+360.

;*************************************
;LATITUDE IS NEGATIVE IF SOUTH HEMI
;*************************************
IF pole eq 2 THEN lat_deg=-lat_deg

;*********************************************
END


;*****************************************************
;*****************************************************
;                    STR.PRO
;*****************************************************
;*****************************************************
;This routine transforms the input into a string
;(whatever its original type), removing all blanks
;
;Input: any scalar type
;
;Output: input converted to string type, no blanks
;
;Calls:          none
;
;Written by/in: Pierre Mercier 04/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
FUNCTION str,data_in

;*****************************************
;IF INPUT IS BYTE,CONVERTS TO INTEGER FIRST
;OTHERWISE STRING CONVERSION MAY NOT GIVE
;EXPECTED RESULT
;*****************************************
s=size(data_in)
IF ((s(0) eq 0) and (s(1) eq 1)) THEN $
data=fix(data_in) ELSE data=data_in

;*****************************************
```

```
;CONVERTS TO STRING, REMOVING ALL BLANKS
;*********************************************
data_out=strcompress(string(data),/remove_all)

;*********************************************
RETURN,data_out

;*********************************************
END


;*****************************************************
;*****************************************************
;                    YY_TO_YYYY.PRO
;*****************************************************
;*****************************************************
;This routine returns a yyyy format year, given
;a yy format one (91<yy<00 : 20st century , 99<yy<92
;: 21st century)
;
;Input: year : yy format year (string or numerical
;                              data type)
;
;Output: yyyy format year (string)
;
;Calls:          none
;
;Written by/in: Pierre Mercier 04/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
FUNCTION yy_to_yyyy,year


;*****************************
yy=fix(year)
IF (yy lt 92) THEN yyyy=2000+yy ELSE yyyy=1900+yy
yyyy=str(yyyy)

;*****************************
RETURN, yyyy

;*****************************
END
```

CONTOURS_MAKING_ROUTINES

```
;*****************************************************
;*****************************************************
;        COMPUTE_CONTOUR.PRO
;*****************************************************
;*****************************************************
;This routine computes the ice edge for Antarctica
;or Arctic at a given time (year,day), averaged over
;6 days. The ice edge is "a priori" based on 15% ice
; concentration (c_level parameter)
;
;There are two key parameters: cutoff(INT) and
;exclude(1xN INTARR). Cutoff specIFies the minimum
;number of elements in each contour saved. Exclude
;is an array (e.g. exclude=[300,244,120]) that is
;used to specIFy contour sizes that should be
;dropped from the list.  For example, IF cutoff=400
;and there is a contour with 420 elements that is not
;a sea ice contour, THEN exclude=[420] will exclude
;that contour from the group.
;
;Several passes over the data are usually necessary
;for averaged_ice_concs in the early and late parts
;of each year when sea ice is smaller in extent.
;cutoff=600 works well for winter months, and
;cutoff=100 with longer exclude arrays
;is effective in early/late parts of the year.
;
;num_cont returns the # of contours retained by the
;sorting routine.;
;
;Input: pole : byte 1=Arctic 2=Antarctic
;              year : yyyy integer specifying year
;              day1 : ddd integer specifying day
;
;Output: File containing ice edge (path specIFied in
;config.pro
;
;Calls:        config.pro
;              read_ice_concentration.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
PRO compute_contour,pole,year,day1

;*******************************************
;LOADS KEY PARAMETERS
;*******************************************
dim_ssmi=config('data','dim_ssmi',pole)
xpix=dim_ssmi(0)
ypix=dim_ssmi(1)
badvalues_ssmi=config('data','badvalues_ssmi',pole)
badval1=badvalues_ssmi(0)
badval2=badvalues_ssmi(1)
c_level=config('data','c_level',pole)
cutoff=config('data','cutoff',pole)
exclude=config('data','cutoff',pole)
num_ex=n_elements(exclude)
ll=config('data','image_ll',pole)
lat_min=ll(0)
image_size=config('data','image_size',pole)
n=image_size(0)

CASE pole of
1:hemi='north'
2:hemi='south'
ENDCASE

;*****************************************************
;READS SSMI GRIDS FOR CONVERSION IJ->LAT,LON
;*****************************************************
lats=fltarr(xpix,ypix)
path=config('paths','ij_ll_grids',hemi,'lats')
openr,unit,path,/get_lun
readu,unit,lats
free_lun,unit

lons=fltarr(xpix,ypix)
path=config('paths','ij_ll_grids',hemi,'lons')
openr,unit,path,/get_lun
readu,unit,lons
free_lun,unit

;*******************************************
;INITIALIZES ARRAYS USED TO COMPUTE ICE EDGE
;*******************************************
averaged_ice_conc = bytarr(xpix,ypix)
ice_conc_data= bytarr(xpix,ypix,6)

;*******************************************
;READS 6 ICE CONCENTRATION IMAGES
;*******************************************
FOR k=0,5 DO BEGIN
day1_k=(day1+k) mod (365+is_leap(year))
IF (day1_k lt day1) THEN year_k=year+1 $
ELSE year_k=year
array=read_ice_concentration(pole,$
day1_k,year_k)
ice_conc_data(*,*,k)=array(*,0:ypix-1)
ENDFOR

;*******************************************
;AVERAGE THE 6 IMAGES
;*******************************************
print,'averaging ice concentration over 6 days...'

FOR l=0,ypix-1 DO BEGIN
FOR m=0,xpix-1 DO BEGIN
a=ice_conc_data(m,l,*)
w=where(( a ne badval1) and (a ne badval2) $
and (a ne 0), count)
IF (count ne 0) THEN averaged_ice_conc(m,l)=$
total(ice_conc_data(m,l,w))/count $
ELSE averaged_ice_conc(m,l)=0
ENDFOR
ENDFOR

;*******************************************
;SELECTS USEFUL PART OF AVERAGED IMAGE
;*******************************************
CASE pole of
1: BEGIN
lat_cut=min(lats(where(lats ge lat_min)))
w=where(lats le lat_cut)
END
2: BEGIN
lat_cut=max(lats(where(lats le lat_min)))
w=where(lats ge lat_cut)
END
ENDCASE

averaged_ice_conc(w)=0b

;*******************************************
;MAKES DESTINATION AND TEMPORARY FILES PATHS
;*******************************************
contour_path=config('paths','ice_contour',$
pole,year,day1)
contour_path_bis=config('paths','tmp_dir')+$
'/contour.tmp'

;*******************************************
;CHECKS EXISTENCE OF DIRECTORIES TO TEMPORARY
;CONTOUR FILE
;*******************************************
openw,unit,contour_path_bis,error=err,/get_lun
IF (err ne 0) THEN BEGIN
make_dirs_to_file,contour_path_bis
openw,unit,contour_path_bis
ENDIF
free_lun,unit

;*******************************************
;FINDS ICE EDGE
;*******************************************
print,'finding 15% ice concentration contour...'

set_plot,'ps'
contour,averaged_ice_conc,level=c_level,$
c_labels=[1],/noerase,/follow,xstyle=1,ystyle=1,$
position=[0,0,xpix-1,ypix-1],/device,$
path_filename=contour_path_bis,background=255,color=0
set_plot,'x'

;*******************************************
;OPENS TEMPORARY FILE
;*******************************************
openr,unit1,contour_path_bis,/get_lun,/delete

;*******************************************
;CHECKS EXISTENCE OF DIRECTORIES TO DESTINATION
;CONTOUR FILE
;*******************************************
openw,unit2,contour_path,error=err,/get_lun
IF (err ne 0) THEN BEGIN
make_dirs_to_file,contour_path
openr,unit2,contour_path
ENDIF
```

```
;*********************************************
;INITIALIZES VARIABLES TO WRITE IN DESTINATION
;FILE
;*********************************************
type = 0b
high = 0b
level=0
num = 0L
value = 0.0
a=0.0
b=0.0
num_cont=0

;*********************************************
;WRITES CONTOURS IN DESTINATION FILE
;*********************************************
WHILE (not EOF(unit1)) DO BEGIN
readu,unit1,type,high,level,num,value
keep=0
IF (num ge cutoff) THEN keep = 1
IF ( (keep eq 1) and (num_ex gt 0)) THEN BEGIN
FOR iik=0,num_ex-1 DO BEGIN
IF num eq exclude(iik) THEN keep = 0
ENDFOR
ENDIF
IF (keep eq 1) THEN BEGIN
num_cont=num_cont+1

;*********************************************
;READS NORMALIZED COORDINATES OF CURRENT
;CONTOUR IN TEMPORARY FILE
;*********************************************
FOR i=0,num-1 DO BEGIN
readu,unit1,a,b

;*********************************************
;DENORMALIZES COORDINATES
;*********************************************
i_ssmi=nint((a-!X.S(0))/!X.S(1))
j_ssmi=nint((b-!Y.S(0))/!Y.S(1))

;*********************************************
;CONVERTS TO LAT,LON
;*********************************************
lat=lats(i_ssmi,j_ssmi)
lon=lons(i_ssmi,j_ssmi)

;*********************************************
;CONVERTS LAT,LON TO I,J FOR SIRF IMAGES
;*********************************************
ll_to_sirf,pole,lat,lon,i_sirf,j_sirf

;*********************************************
;POINT OUTSIDE POLAR AREA IS TRANSFORMED
;INTO POINT ON BORDER OF POLAR AREA
;*********************************************
x_center=0.5*(n-1)
y_center=0.5*(n-1)
r0=0.5*(n-1)
r=sqrt((i_sirf-x_center)^2+(j_sirf-y_center)^2)

WHILE (r gt r0) DO BEGIN
i_sirf=nint(i_sirf+(x_center-i_sirf)/r)
j_sirf=nint(j_sirf+(y_center-j_sirf)/r)
r=sqrt((i_sirf-x_center)^2+(j_sirf-y_center)^2)
ENDWHILE

;*********************************************
;PRINTS OBTAINED COORDINATES IN DESTINATION FILE
;*********************************************
printf,unit2,i_sirf,j_sirf,format='(i3,1x,i3)'

ENDFOR

;*********************************************
;MARKS END OF CONTOUR I
;*********************************************
printf,unit2,999,999,format='(i3,1x,i3)'

ENDIF

;*********************************************
;SKIPS DISCARDED CONTOURS
;*********************************************
IF (keep eq 0) THEN BEGIN
FOR i=0,num-1 DO BEGIN
readu,unit1,a,b
ENDFOR
ENDIF
ENDWHILE


;*********************************************
;CLOSES DESTINATION AND TEMPORARY FILES
;*********************************************
free_lun,unit2
free_lun,unit1

;*********************************************
;WARNS WHEN FINISHED
;*********************************************
print,contour_path,' computed.'
print,''

;*********************************************
END


;*************************************************
;*************************************************
;    INTERFACE_CONTOURS.PRO
;*************************************************
;*************************************************
;This routine sets up graphical interface which
;enables the user to compute contours of Arctic or
;Antarctic regions at a given time
;
;Input: none
;
;Output: graphical interface
;
;Calls:         config.pro
;               compute_contour.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*************************************************
;*************************************************
PRO interface_contours

;*********************************************
;*********************************************
;    INTERFACE WIDGETS DEFINITION
;*********************************************
;*********************************************

;*************************************
;INTERFACE VARIABLES DEFINITION AND
;INITIALIZATION
;*************************************
common variables_contours,variables_contours
variables_contours=config('structures','st10')

;*******************************
;MAIN WINDOW
;*******************************
common mw_contours,mw_contours
common mw_menu,mw_menu
mw_contours=wwmainwindow(mw_menu,$
layout_contours,/vertical,title=$
'Contours Computation',position=[100,100])

;*******************************
;INFO TEXT
;*******************************
tx00=wwtext(layout_contours,text='',/label)
tx01=wwtext(layout_contours,text='Compute 15% '+$
'ice conc. contours of :  ',/label)
tx02=wwtext(layout_contours,text='',/label)

;*******************************
;ARCTIC WIDGETS
;*******************************
ly1=wwlayout(layout_contours)
tx1=wwtext(ly1,text='Arctic in      ',/label)
years_arc_contours_ls=wwlist(ly1,$
config('interfaces','years_array'),$
'years_arc_contours_cb',$
'years_arc_contours_cb',/multi)

;*******************************
;ANTARCTIC WIDGETS
;*******************************
ly2=wwlayout(layout_contours)
tx2=wwtext(ly2,text='Antarctic in ',/label)
years_ant_contours_ls=wwlist(ly2,$
config('interfaces','years_array'),$
'years_ant_contours_cb',$
'years_ant_contours_cb',/multi)
```

```
;*****************************
;COMPUTE/CANCEL BUTTONS
;*****************************
go_contours_bb=wwbuttonbox(layout_contours,$
['Compute','Cancel'],'go_contours_cb')

;*****************************
;DISPLAYS INTERFACE
;*****************************
status=wwsetvalue(mw_contours,/display)

;*****************************
END



;*************************************************
;*************************************************
;                    CALLBACK PROCEDURES
;*************************************************
;*************************************************

;*****************************************
;UPDATES SELECTED YEARS FOR ARCTIC
;*****************************************
PRO years_arc_contours_cb,wid,index
common variables_contours,variables_contours
variables_contours.years_arc(*)=''
variables_contours.years_arc=wwgetvalue(wid)
END

;*****************************************
;UPDATES SELECTED YEARS FOR ANTARCTIC
;*****************************************
PRO years_ant_contours_cb,wid,index
common variables_contours,variables_contours
variables_contours.years_ant(*)=''
variables_contours.years_ant=wwgetvalue(wid)
END

;*****************************************
;MANAGES COMPUTE/CANCEL BUTTONS
;*****************************************
PRO go_contours_cb,wid,index

common variables_contours,variables_contours
common mw_contours,mw_contours

CASE index of

;*****************************
;     COMPUTES CONTOURS
;*****************************
1: BEGIN

;******************
;ALERT BOX
;******************
contours_al=wwalert(mw_contours,$
'Computing contours ...',/noconfirm,$
/working,title='INFO',nowait=1)

print,''
print,'*****************************'
print,'Beginning Contours computation'
print,'*****************************'
print,''

;*****************************************
;ARCTIC CONTOURS COMPUTATION
;*****************************************
w=where(variables_contours.years_arc ne '',$
n_years_arc)
FOR i=0,n_years_arc-1 DO $
FOR day=0,363,3 DO compute_contour,1b,$
variables_contours.years_arc(i),day

;*****************************************
;ANTARCTIC CONTOURS COMPUTATION
;*****************************************
w=where(variables_contours.years_ant ne '',$
n_years_ant)
FOR i=0,n_years_ant-1 DO $
FOR day=0,363,3 DO compute_contour,2b,$
variables_contours.years_ant(i),day

;******************
;CLOSES ALERT BOX
;******************
wwalertpopdown,contours_al

print,'*****************************'
print,'     Contours computed'
print,'*****************************'
print,''


;******************
;EXITS INTERFACE
;******************
status=wwsetvalue(mw_contours,/close)

;******************
END

;*****************************
;EXITS INTERFACE
;*****************************
2: status=wwsetvalue(mw_contours,/close)

;*****************************
ENDCASE

;*****************************
END


;*************************************************
;*************************************************
;READ_ICE_CONCENTRATION.PRO
;*************************************************
;*************************************************
;This routine reads Ice Concentration data from
;a location specified by config.pro. This routine
;is directly inspired from one of Mark Drinwater's
;and thus is not optimized for TOPICS. Some things
;can be made faster, others are not relevant here.
;So, modifications are welcome.
;
;Input: ihem: 1=Arctic 2=Antarctic
;               day1: ddd integer specifying day
;               year: yyyy integer specifying year
;
;Output: ice concentration grid for the specified
;                year+day
;
;Calls:         config.pro
;               number_to_date.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*************************************************
;*************************************************
FUNCTION read_ice_concentration,ihem,day1,year

@hdf_common

;*************************************************
;INITIALIZES VARIABLES
;*************************************************

start = ''
stop  = ''
hem   = ''
ctype  = ''
mo    = ''
chan  = ''
directory = ''
path = ''
type = ''

w     = 0
days  = 0
hemis = 0
err   = 0
itype = 0
pal   = bytarr(3,256)

start=strmid(number_to_date(year*1000l+day1),2,6)
stop=start

;*************************************************
;ALGORITHM (1=NASA TEAM)
;                (2=COMISO)
;*************************************************
ialgo = 1

;*************************************************
;ICE TYPE (1=TOTAL)
;                (2=MULTI-YEAR, FOR NASATEAM NORTHERN ONLY)
;*************************************************
itype=1

;*************************************************
;SPECIFIES LOCATION OF ICE CONCENTRATION DATA
```

```
;*****************************************************
directory=config('paths','ice_concentration')+$
string(year,format='(I4)')

;*****************************************************
;DETERMINES VARIABLES FOR LATER USE
;*****************************************************
IF(ihem eq 1)THEN BEGIN
xdim = 304
ydim = 448
hem  = 'N'
ENDIF  ;hemi eq 1
IF(ihem eq 2)THEN BEGIN
xdim = 316
ydim = 332
hem  = 'S'
ENDIF  ;ihem eq 2
IF(ialgo eq 1)THEN algo = 'N'
IF(ialgo eq 2)THEN algo = 'C'
IF(itype eq 1)THEN type = 'T'
IF(itype eq 2)THEN type = 'M'

;*****************************************************
;EXTRACTS INFORMATION FROM INPUT
;*****************************************************
yy      = fix(strmid(start,0,2))
yyst    = strmid(start,0,2)
mmst    = strmid(start,2,2)
ddst    = strmid(start,4,2)
yysp    = strmid(stop,0,2)
mmsp    = strmid(stop,2,2)
ddsp    = strmid(stop,4,2)

iday    = fix(strmid(start,4,2))
imonth  = fix(strmid(mmst,0,2))
immst   = fix(strmid(mmst,0,2))
immsp           = fix(strmid(mmsp,0,2))

;*****************************************************
;SETS UP ARRAYS
;*****************************************************
days=[31,28,31,30,31,30,31,31,30,31,30,31]
IF(yy eq 92 or yy eq 96)THEN BEGIN
days=[31,29,31,30,31,30,31,31,30,31,30,31]
ENDIF

;*****************************************************
; DETERMINES THE TOTAL NUMBER OF DAYS
;*****************************************************
n_o_d  = 1

;*****************************************************
;INITIALIZES ARRAYS
;*****************************************************
titles  = 52
title           = intarr(xdim,52)
ice             = 0
ice             = bytarr(xdim,ydim+titles,n_o_d)
tmp             = bytarr(xdim,ydim)

;*****************************************************
; STARTS LOOP TO CREATE FILENAMES
;*****************************************************
FOR d = 1,n_o_d DO BEGIN

;*****************************************************
;BUILDS FILENAMES FOR MONTHS LE 9 AND DAYS LE 9
;*****************************************************
IF(imonth le 9)THEN BEGIN
IF(iday le 9)THEN BEGIN
filename = string(format='(i2,a1,i1,a1,i1,a1,3a1)',$
yy,'0',imonth,'0',iday,'.',type,hem,algo)
ENDIF $;iday < 9

;*****************************************************
;BUILDS FILENAMES FOR MONTHS LE 9 AND DAYS GE 10
;*****************************************************
ELSE BEGIN
filename = string(format='(i2,a1,i1,i2,a1,3a1)',$
yy,'0',imonth,iday,'.',type,hem,algo)
ENDELSE  ;iday > 10
ENDIF  ;imonth > 9

;*****************************************************
;BUILDS FILENAMES FOR MONTHS GE 10 AND DAYS LE 9
;*****************************************************
IF(imonth ge 10)THEN BEGIN
IF(iday le 9)THEN BEGIN
filename = string(format='(i2,i2,a1,i1,a1,3a1)',$
yy,imonth,'0',iday,'.',type,hem,algo)
ENDIF $;iday < 9

;*****************************************************
;BUILDS FILENAMES FOR MONTHS GE 10 AND DAYS GE 10
;*****************************************************
ELSE BEGIN
filename = string(format='(i2,i2,i2,a1,3a1)',$
yy,imonth,iday,'.',type,hem,algo)
ENDELSE   ;iday > 10
ENDIF   ;imonth > 10

;*****************************************************
;BUILDS FULL PATH TO THE DATA FILE
;*****************************************************
path = string(format='(a,a1,a)',directory,$
'/',filename)

;*****************************************************
;ATTEMPTS TO OPEN DATA FILES. EXISTENCE CHECK ONLY
;*****************************************************
openr,1,path,error=err

;*****************************************************
;IF FOUND THEN CLOSES AND EXTRACTS USING
;THE IDL-HDF READ
;*****************************************************
IF (err eq 0) THEN BEGIN
close,1

;*****************************************************
;READS AN HDF DATA FILE
;*****************************************************
status=$
DFR8GETIMAGE(path,tmp,xdim,ydim,pal)
tvlct,pal(0,*),pal(1,*),pal(2,*)
tmp=rotate(tmp,7)
ice(*,0:ydim-1,d-1)=tmp
ice(*,ydim:ydim+titles-1,d-1)=title
ENDIF  ;err eq 0

;*****************************************************
;IF FILE NOT FOUND, TELLS USER AND MARKS THE ENTIRE
;DAY AS MISSING (157)
;*****************************************************
IF(err ne 0) THEN BEGIN
close,1
print,'File not found ... ',filename
ice(*,0:ydim-1,d-1)=157
ice(*,ydim:ydim+titles-1,d-1)=title
ENDIF  ;err ne 0
iday=iday+1

;*****************************************************
;CHECKS FOR A NEW MONTH, IF SO, INCREMENTS MONTH
;*****************************************************
IF(iday gt days(imonth-1))THEN BEGIN
imonth=imonth+1
iday=1

;*****************************************************
;CHECKS FOR A NEW YEAR, IF SO, INCREMENT YEAR
;*****************************************************
IF(imonth gt 12)THEN BEGIN
imonth=1
yy=yy+1
ENDIF ;imonth gt 12
ENDIF ;iday gt days
IF(yy eq 92 or yy eq 96) THEN BEGIN
days=[31,29,31,30,31,30,31,31,30,31,30,31]
ENDIF
ENDFOR

;*****************************************************
return, ice

;*****************************************************
END
```

CUSTOM_ROUTINES

```
;*****************************************************
;*****************************************************
;        INTERFACE_CUSTOM1.PRO
;*****************************************************
;*****************************************************
;This routine sets up graphical interface which
;enables the user to define its own regions
;of investigation for making statistics. It calls
;interface_custom2.pro as a child interface
;
;Input: none
;
;Output: displays graphical interface
;
;Calls:          config.pro
;                read_regions_equivalents.pro
;                interface_custom2.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*********************************************************
;*********************************************************
PRO interface_custom1

;*****************************************
;*****************************************
;    INTERFACE WIDGETS DEFINITION
;*****************************************
;*****************************************

;*************************************
;INTERFACE VARIABLES DEFINITION AND
;INITIALIZATION
;*************************************
COMMON variables_custom,variables_custom
variables_custom=config('structures','st8')
variables_custom.pole=1

;*********************************
;                MAIN WINDOW
;*********************************
COMMON mw_custom1,mw_custom1
COMMON mw_menu,mw_menu
mw_custom1=wwmainwindow(mw_menu,layout_define,$
title='New Region Definition',$
/vertical,position=[50,100])

;*****************************
; POLAR REGION OPTION MENU
;*****************************
pole_om=wwoptionmenu(layout_define,'Pole :',$
{,callback:'pole_custom1_cb',$
button:'Arctic',button:'Antarctic'})

;*****************************
;  REGION NAME WIDGETS
;*****************************
region_name_ly=wwlayout(layout_define,/horizontal)
region_name_lb=wwtext(region_name_ly,$
/label,text='Region Name (3 chars) [ENTER]: ')
region_name_tx=wwtext(region_name_ly,$
'region_name_custom1_cb',cols=12)

;*****************************
; REGION SHORT NAME WIDGETS
;*****************************
region_short_name_ly=wwlayout(layout_define,/horizontal)
region_short_name_lb=wwtext(region_short_name_ly,$
/label,text='Region Short Name (3 chars) [ENTER]: ')
region_short_name_tx=wwtext(region_short_name_ly,$
'region_short_name_custom1_cb',cols=3)

;*****************************
; CONTINUE/CANCEL BUTTONS
;*****************************
go_define_bb=wwbuttonbox(layout_define,$
['Continue','Cancel'],'go_custom1_cb')

;*****************************
;    DISPLAYS INTERFACE
;*****************************
status=wwsetvalue(mw_custom1,/display)

;*****************************
END


;*****************************************
;*****************************************
;                CALLBACK PROCEDURES
;*****************************************
;*****************************************

;*****************************
;UPDATES POLAR_REGION
;*****************************
PRO pole_custom1_cb,wid,index
COMMON variables_custom,variables_custom
variables_custom.pole=index
END

;*****************************
;UPDATES REGION NAME
;*****************************
PRO region_name_custom1_cb,wid,parent
COMMON variables_custom,variables_custom
a=wwgetvalue(wid)
variables_custom.region_name=$
strupcase(strmid(a,0,1))+$
strlowcase(strmid(a,1,strlen(a)-1))
END

;*****************************
;UPDATES REGION SHORT NAME
;*****************************
PRO region_short_name_custom1_cb,wid,parent
COMMON variables_custom,variables_custom
a=wwgetvalue(wid)
variables_custom.region_short_name=$
strupcase(strmid(a,0,1))+$
strlowcase(strmid(a,1,strlen(a)-1))
END

;*****************************
;MANAGES CONTINUE/CANCEL BUTTONS
;*****************************
PRO go_custom1_cb,wid,index

COMMON mw_custom1,mw_custom1

CASE index OF

;*****************************
;                CONTINUE
;*****************************
1: BEGIN

COMMON variables_custom,variables_custom

;*************************
;CHECKS IF ENTRIES ARE OK.
;IF NOT, DISPLAYS ALERT BOX
;AND GOTO 'ALERT' LABEL
;*************************
IF (variables_custom.region_name eq '' or $
variables_custom.region_short_name eq '' or $
strlen(variables_custom.region_short_name) ne 3) $
THEN BEGIN
alert_custom1=wwalert(mw_custom1,title='ALERT',$
"You didn't type ENTER or at least one field "+$
"is blank or short name is not 3 chars long ",$
['OK'],error=1)
GOTO,alert
ENDIF

;*************************
;CHOOSE TOPICS NUMBER
;FOR THE REGION
;*************************
read_regions_equivalents,n,numbers,names,short_names
region_number=2-(variables_custom.pole mod 2)
REPEAT BEGIN
region_number=region_number+2
w=where(numbers eq str(region_number),count)
ENDREP until (count eq 0)
variables_custom.region_number=region_number

;*************************
;EXITS INTERFACE
;*************************
status=wwsetvalue(mw_custom1,/close)

;*************************
;ALERT BOX : MAP LOADING
;*************************
COMMON mw_menu,mw_menu
alert_custom1=wwalert(mw_menu,'Map loading (~1/2 min)...',$
after=5,title='INFO',noconfirm=1,working=1)
interface_custom2

;*****************************
```

```
alert:

;*****************************
END

;*****************************
;                CANCEL
;*****************************
2: status=wwsetvalue(mw_custom1,/close)

;*****************************
ENDCASE

;*****************************
END


;**************************************************
;**************************************************
;       INTERFACE_CUSTOM2.PRO
;**************************************************
;**************************************************
;This routine sets up graphical interface which
;enables the user to define its own regions
;of investigation for making statistics. It is called
;by interface_custom1.pro (parent interface)
;
;Input: none
;
;Output: displays graphical interface
;
;Calls:         plot_contour.pro
;               ll_to_sirf.pro
;               update_x_y_region.pro
;               config.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;**************************************************
;**************************************************
PRO interface_custom2

;*******************************************
;*******************************************
;   INTERFACE WIDGETS DEFINITION
;*******************************************
;*******************************************

COMMON variables_custom,variables_custom

;*********************************
;                MAIN WINDOW
;*********************************
COMMON mw_custom2,mw_custom2
COMMON mw_menu,mw_menu
mw_custom2=wwmainwindow(mw_menu,layout_custom2,$
title=variables_custom.region_name+' ('+$
variables_custom.region_short_name+')'+$
' Region Contour Drawing',/vertical,position=[370,100])

;*********************************
;START/SAVE/CANCEL/HELP BUTTONS
;*********************************
ly=wwlayout(layout_custom2,/horizontal)
COMMON enable_mouse_bb,enable_mouse_bb
enable_mouse_bb=wwbuttonbox(ly,$
[' Start '],'enable_mouse_custom2_cb')
go_custom2_bb=wwbuttonbox(ly,$
['Save as New Region','Cancel','Help'],$
'go_custom2_cb')

;*********************************
;LOADS MAP / MAP DRAWING AREA
;*********************************
COMMON pole_map,pole_map
loadct,11,/silent
invert_ct
CASE variables_custom.pole OF
1: BEGIN
plot_contour,1,1992,0,770,pole_map,/coord
pole_map_dw=wwdrawing(layout_custom2,1,$
'pole_map_custom2_cb',[774,774],[770,770])
END
2: BEGIN
plot_contour,2,1992,183,970,pole_map,/coord
pole_map_dw=wwdrawing(layout_custom2,1,$
'pole_map_custom2_cb',[774,774],[970,970])
END
ENDCASE

;*********************************
;       COORDINATES LABEL
;*********************************
COMMON coord_lb,coord_lb
coord_lb=wwtext(layout_custom2,text=$
'    (Latitude ; Longitude) = ',/label)

;*********************************
;       DISPLAYS INTERFACE
;*********************************
status=wwsetvalue(mw_custom2,/display)

;*********************************
END



;*******************************************
;*******************************************
;                CALLBACK ROUTINES
;*******************************************
;*******************************************

;*********************************
;ENABLES MOUSE DRAWING
;*********************************
PRO enable_mouse_custom2_cb,wid,index
COMMON mw_custom2,mw_custom2
status=wwsetvalue(wid,/nonsensitive)
status=wwsetvalue(mw_custom2,/update)
sketch
END

;*********************************
;CLOSES CONTOURS
;*********************************
PRO close_contour_custom2_cb,wid,index
update_x_y_region,/stop_mouse
END

;*********************************
;DRAWS POLAR REGION MAP
;*********************************
PRO pole_map_custom2_cb,wid,index
COMMON pole_map,pole_map
tv,pole_map
END

;*********************************
;MANAGES SAVE/CANCEL/HELP BUTTONS
;*********************************
PRO go_custom2_cb,wid,index

COMMON mw_custom2,mw_custom2
COMMON x_region,x_region
COMMON y_region,y_region

CASE index OF

;*************************
;                SAVE REGION
;*************************
1: BEGIN

COMMON variables_custom,variables_custom

;******************
;ALERT BOX
;******************
alert_custom2=wwalert(mw_custom2,'The Region '+$
variables_custom.region_name+' ('+$
variables_custom.region_short_name+$
') is going to be added'+$
' to the list of regions used by TOPICS',$
['OK','Cancel'],$
title='Confirmation',warning=1)

IF (alert_custom2 eq 1) THEN BEGIN

;*********************
;USER ACCEPTS SAVING
;*********************

;******************
;UPDATES EQUIVALENCE
;TABLE
;******************
path=config('paths','regions_equivalents')
openu,unit,path,/append,/get_lun
printf,unit,str(variables_custom.region_number)
printf,unit,variables_custom.region_name
printf,unit,variables_custom.region_short_name
printf,unit,''
free_lun,unit
```

```
;*****************
;WRITES CONTOUR
;AND MASK IN A FILE
;*****************
region_mask_path=config('paths','region_mask',$
variables_custom.region_number)
openw,unit,region_mask_path,/get_lun
image_size=$
config('data','image_size',variables_custom.pole)
n=image_size(0)
a=polyfillv(x_region,y_region,n,n)
cont=[[x_region],[y_region]]
writeu,unit,n_elements(cont)
writeu,unit,cont
writeu,unit,n_elements(a)
writeu,unit,a
free_lun,unit

;********************
;USER ABORTS SAVING
;********************
ENDIF ELSE alert_custom2=wwalert(mw_custom2,$
'Region Definition Aborted',$
title='Abort',noconfirm=1,after=3)

;********************
;EXITS INTERFACE
;********************
status=wwsetvalue(mw_custom2,/close)
END

;**************************
;                CANCEL
;**************************
2: BEGIN
alert_custom2=wwalert(mw_custom2,$
'Region Definition Aborted',$
title='Abort',noconfirm=1,after=3)
status=wwsetvalue(mw_custom2,/close)
END

;**************************
;                HELP
;**************************
3: help_al=wwalert(mw_custom2,$
['Press Start : enables mouse',$
'Press first mouse button : draws a point',$
'Hold first mouse button : draws a curve',$
'Press second mouse button : draws arc of parallel'+$
' between latest selected point ',$
'and the point whose longitude is equal to '+$
'that of the present point',$
'Press third mouse button : disables mouse'],$
title='Help',['OK'])

;**************************
ENDCASE

;***********************************
END


;*****************************************************
;*****************************************************
;                INTERFACE_SEE_REGIONS.PRO
;*****************************************************
;*****************************************************
;This routine sets up graphical interface which
;enables the user to see regions currently defined.
;
;Input: none
;
;Output: displays graphical interface
;
;Calls:         config.pro
;               plot_contour.pro
;               read_regions_equivalents.pro
;               read_file.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
PRO interface_see_regions

;*********************************************
;*********************************************
;    INTERFACE WIDGETS DEFINITION
;*********************************************
;*********************************************

;*****************************

;MAIN WINDOW
;*****************************
COMMON mw_see_regions,mw_see_regions
COMMON mw_menu,mw_menu
mw_see_regions=wwmainwindow(mw_menu,layout_see_regions,$
title='See Current Regions',/vertical,position=[370,100])

;*****************************
;TOP LAYOUT
;*****************************
ly=wwlayout(layout_see_regions)

;*****************************
;POLAR REGION OPTION MENU
;*****************************
regions_see_regions_om=$
wwoptionmenu(ly,'Region : ',$
{,callback:'region_see_regions_cb',$
button:'Arctic',button:'Antarctic'})

;*****************************
;EXIT BUTTON
;*****************************
tx=wwtext(ly,text='     ',/label)
go_see_regions_bb=wwbuttonbox(ly,$
[' Exit '],'go_see_regions_cb',/right)

;*****************************
;MAP DRAWING AREA
;*****************************
loadct,11,/silent
invert_ct
COMMON drawing_see_regions_dw,drawing_see_regions_dw
drawing_see_regions_dw=wwdrawing(layout_see_regions,$
1,'null_cb',[780,780],[970,970],background=255b)


;*****************************
;DISPLAYS INTERFACE
;*****************************
status=wwsetvalue(mw_see_regions,/display)

;*****************************
END



;*********************************************
;*********************************************
;                CALLBACK PROCEDURES
;*********************************************
;*********************************************

;**********************************
;MANAGES POLAR REGION OPTION MENU
;**********************************
PRO region_see_regions_cb,wid,index

;************************
;ALERT BOX : MAP LOADING
;************************
COMMON mw_see_regions,mw_see_regions
see_regions_al=$
wwalert(mw_see_regions,'Map loading...',$
/working,/noconfirm,title='INFO',nowait=1)

;**********************************
;LOADS USEFUL PARAMETERS
;**********************************
pole=index
image_size=config('data','image_size',index)
map_size=image_size(0)
ddd=183*(pole-1)

;**********************************
;MAKES POLAR AREA MAP
;**********************************
plot_contour,pole,1992,$
ddd,map_size,pole_map

;**********************************
;RESIZES DRAWING AREA
;**********************************
COMMON drawing_see_regions_dw,$
drawing_see_regions_dw
status=wwsetvalue(drawing_see_regions_dw,[map_size,map_size])
status=wwsetvalue(drawing_see_regions_dw,/update)

;**********************************
;PLOTS MAP
;**********************************
erase
tv,pole_map
```

```
;************************************
;CLOSES ALERT BOX
;************************************
wwalertpopdown,see_regions_al

;************************************
;DRAWS REGIONS CONTOURS
;************************************
read_regions_equivalents,n,numbers,names,short_names
x=0
y=0
tx=''
FOR i=2,n-1 DO $
IF ((numbers(i) mod 2) eq (pole mod 2)) $
THEN BEGIN

read_file,config('paths','region_contour_and_mask',$
numbers(i)),'region_contour_and_mask',header,$
contour_coord,mask

FOR k=0,n_elements(contour_coord)/2-2 DO $
plots,[contour_coord(k,0),contour_coord(k+1,0)],$
[contour_coord(k,1),contour_coord(k+1,1)],$
color=255b,/device

;************************************
;GETS COORD OF REGIONS LABELS
;************************************
x=[x,(max(contour_coord(*,0))+$
min(contour_coord(*,0)))/2]
y=[y,(max(contour_coord(*,1))+$
min(contour_coord(*,1)))/2]
tx=[tx,names(i)]

ENDIF

x=x(1:*)-30
y=y(1:*)
tx=tx(1:*)

;************************************
;LABELS REGIONS, AVOIDING LABELS
;OVERLAPPING
;************************************
w=sort(y)
x=[-100,x(w)]
y=[-100,y(w)]
tx=['',tx(w)]
FOR i=1,n_elements(y)-1 DO BEGIN
diff=y(i)-y(i-1)
IF diff ge 20 THEN $
xyouts,x(i),y(i),tx(i),charsize=1.3,$
color=255b,/device $
ELSE BEGIN
y(i:*)=y(i:*)+20-diff
xyouts,x(i),y(i),tx(i),charsize=1.3,$
color=255b,/device
ENDELSE
ENDFOR

;************************************
END

;************************************
;MANAGES EXIT BUTTON
;************************************
PRO go_see_regions_cb,wid,index
COMMON mw_see_regions,mw_see_regions
status=wwsetvalue(mw_see_regions,/close)
END


;*****************************************************
;*****************************************************
;                    INVERT_CT.PRO
;*****************************************************
;*****************************************************
;This routine makes the background appear white
;and the foreground black by changing end values
;of current color_table
;
;Input: none
;
;Output: none
;
;Calls:          none
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************

PRO invert_ct

;*********************************
tvlct,R,G,B,/get

;*********************************
R(0)=255b
G(0)=255b
B(0)=255b

;*********************************
R(!d.n_colors-1)=0b
G(!d.n_colors-1)=0b
B(!d.n_colors-1)=0b

tvlct,R,G,B

;*********************************
END


;*****************************************************
;*****************************************************
;                  PLOT_CONTOUR.PRO
;*****************************************************
;*****************************************************
;This routine produces a map of Arctic/Antarctic
;with land, sea and ice. Depending on keywords,
;the map is just returned, plotted in a .ps file,
;or on the screen. It is also possible to ask for
;meridiens and parallels.
;
;Input: pole : 1=Arctic 2=Antarctic
;               year : yyyy integer
;               day1 : ddd integer
;               size : integer giving size of desired
;               output window if keyword /x specified
;
;Output: contour : an array containing the desired
;                  contour
;
;Calls:          config.pro
;               ll_to_sirf.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
PRO plot_contour,pole,year,day1,size,contour,$
x=keyword_x,ps=keyword_ps,file_ps=keyword_file_ps,$
coord=keyword_coord

;*****************************************
;LOADS NICE COLOR TABLE AND MAKES SURE
;BACKGROUND IS WHITE, WRITING IS BLACK
;*****************************************
loadct,11,/silent
invert_ct

;*****************************************
;LOADS USEFUL PARAMETERS
;*****************************************
image_size=config('data','image_size',pole)
n=image_size(0)
contour=replicate(0b,n,n)

;*****************************************
;READS AND APPLY SIRF AREA MASK
;*****************************************
path=config('paths','circle_sirf',pole)
count=0l
openr,unit,path,/get_lun
readu,unit,count
struc=assoc(unit,lonarr(count))
w_sea=struc(0)
free_lun,unit

contour(w_sea)=55b

;*****************************************
;READS AND APPLY ICE MASK
;*****************************************
contour_path=config('paths','ice_contour',pole,year,day1)
openr,1,contour_path
x=0
y=0
x_vector=intarr(1)
y_vector=intarr(1)
WHILE (not(eof(1))) DO BEGIN
readf,1,x,y,format='(i3,1x,i3)'
IF ((x eq 999) and (y eq 999)) THEN BEGIN
x_vector=x_vector(1:*)
```

```
y_vector=y_vector(1:*)
ice_mask=polyfillv(x_vector,y_vector,n,n)
contour(ice_mask)=0b
x_vector=intarr(1)
y_vector=intarr(1)
ENDIF ELSE BEGIN
x_vector=[x_vector,x]
y_vector=[y_vector,y]
ENDELSE
ENDWHILE
close,1

;*****************************************
;READS AND APPLY LAND MASK
;*****************************************
land_mask_path=config('paths','land_mask',pole)
read_file,land_mask_path,'land_mask',count,land_mask
contour(land_mask)=30b

;*****************************************
;READS AND APPLY MIDDLE CIRCLE MASK
;*****************************************
path=config('paths','middle_circle_sirf',pole)
count=0l
openr,unit,path,/get_lun
readu,unit,count
struc=assoc(unit,lonarr(count))
w_middle=struc(0)
free_lun,unit

contour(w_middle)=20b

;*********************************************
;IF SPECIFIED BY KEYWORD, PLOTS PARALLELS
;AND MERIDIENS
;*********************************************
IF keyword_set(keyword_coord) THEN BEGIN

;*****************************************
;LOADS INFO ABOUT POLAR_REGION
;*****************************************
image_ll=config('data','image_ll',pole)
lat0=image_ll(0)
l=image_ll(1)
latinf=lat0<l
latsup=lat0>l

;*************************************
;PLOTS PARALLELS
;*************************************
CASE pole OF
1: BEGIN
FOR j=latsup,latinf,-10 DO BEGIN
FOR i=-180,180,2 DO BEGIN
ll_to_sirf,pole,j,i,x,y
contour(x,y)=255b
ENDFOR
ENDFOR
END
2: BEGIN
FOR j=latinf,latsup,10 DO BEGIN
FOR i=-180,180,2 DO BEGIN
ll_to_sirf,pole,j,i,x,y
contour(x,y)=255b
ENDFOR
ENDFOR
END
ENDCASE

;*************************************
;PLOTS MERIDIENS
;*************************************
FOR j=-180,180,30 DO BEGIN
FOR i=latinf,latsup DO BEGIN
ll_to_sirf,pole,i,j,x,y
contour(x,y)=255b
ENDFOR
ENDFOR

;*********************************************
ENDIF

;*************************************
;WRITES CONTOUR TO .PS FILE IF /PS
;KEYWORD IS SET
;*************************************
IF (keyword_set(keyword_ps)) THEN BEGIN
destination_file=file_ps
set_plot,'ps'
device, file=destination_file,bits_per_pixel=8,/color,$
xsize=21.6,ysize=21.6,$
xoffset=5,yoffset=5,scale_factor=1
loadct,2
tvscl,contour,/normal,xsize=0.5,ysize=0.5
device,/close
set_plot,'x'
ENDIF

;*************************************
;DISPLAYS CONTOURS IN X WINDOW IF
;/X KEYWORD IS SET
;*************************************
IF (keyword_set(keyword_x)) THEN BEGIN
loadct,0
tv,contour,xsize=size,ysize=size
ENDIF

;*************************************
END


;*************************************************
;*************************************************
;                    SKETCH.PRO
;*************************************************
;*************************************************
;This routine is used by interface_custom2.pro
;together with update_x_y_region.pro to allow the
;user to define its own regions of investigations
;by drawing contours on polar maps. Sketch.pro
;manages the drawing of the contour, whereas
;update_x_y_region.pro to update the contour in_line.
;
;These two routines work but could easily be
;simplified. So if you feel like...
;
;Input: none
;
;Output: none (update_x_y_region.pro produces
;                the contour)
;
;Calls:          sirf_to_ll.pro
;                update_x_y_region.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*******************************************************
;*******************************************************
PRO sketch

;*******************************
;LOADS/INITIALIZES USEFUL
;PARAMETERS/VARIABLES
;
;first: boolean =1 if first point
;x_region: x coordinates of contour points
;y_region: y coordinates of contour points
;x_first,y_first: coord of first point
;x_new,ynew: coord of current point
;xold,yold: coord of last point
;*******************************
common first,first
common x_region,x_region
common y_region,y_region
common xfirst,xfirst
common yfirst,yfirst
common xnew,xnew
common ynew,ynew
common coord_lb,coord_lb
common xold,xold
common yold,yold
xold=0
yold=0

first=1b
x_region=[0]
y_region=[0]
xfirst=0
yfirst=0

common variables_custom,variables_custom
pole=2-(variables_custom.region_number mod 2)

;*******************************
;BEGINS LOOP
;*******************************
REPEAT BEGIN

;*******************************
;LOADS CURSOR LOCATION
;IN XNEW, YNEW
;*******************************
cursor,xnew,ynew,0,/device

;*******************************
```

```
;EXITS LOOP IF MOUSE BUTTON 3
;IS PRESSED (LAST POINT SIGNAL)
;*****************************
IF (!err eq 4) THEN goto,go_on

;*****************************
;IF CURSOR MOVED AND LOCATED
;IN DRAWING AREA ...
;*****************************
IF (not(xnew eq -1 or ynew eq -1) and $
((xnew ne xold) and (ynew ne yold))) $
THEN BEGIN

;*****************************
;...UPDATES COORDINATES LABEL
;AND ...
;*****************************
sirf_to_ll,pole,xnew,ynew,lat,lon
text_coord='    (Latitude ; Longitude) = ( '+$
string(lat,format='(f5.1)')+'; '+$
string(lon,format='(f6.1)')+')'
status=wwsetvalue(coord_lb,text_coord)
status=wwsetvalue(coord_lb,/update)

;*****************************
;...UPDATES CONTOUR BY ADDING
;LINE IF MB1 PRESSED, AN ARC
;OF PARALLEL IF MB2 PRESSED
;*****************************
IF (!err eq 1) THEN update_x_y_region
IF (!err eq 2) THEN update_x_y_region,/parallel
ENDIF

;*****************************
;CONTINUES LOOP
;*****************************
ENDREP UNTIL 0b

;*****************************
;LAST POINT: CLOSES CONTOUR
;*****************************
go_on : update_x_y_region,/stop_mouse

;*****************************
END

;*****************************************************
;*****************************************************
;                  UPDATE_X_Y_REGION.PRO
;*****************************************************
;*****************************************************
;This routine is used by interface_custom2.pro
;together with sketch.pro to allow the user to define
;its own regions of investigations by drawing contours
;on polar maps. Sketch.pro manages the drawing of
;the contour, whereas update_x_y_region.pro to update
;the contour in_line.
;
;These two routines work but could easily be
;simplified. So if you feel like...
;
;Input: none
;
;Output: produces x and y coord of contour points
;                  as common variables
;
;Calls:          sirf_to_ll.pro
;                ll_to_sirf.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
PRO update_x_y_region,stop_mouse=keyword_stop,$
parallel=keyword_parallel

;***************************************
;LOADS USEFUL VARIABLES
;***************************************
COMMON variables_custom,variables_custom
pole=2-(variables_custom.region_number mod 2)

COMMON xold,xold
COMMON yold,yold
COMMON xnew,xnew
COMMON ynew,ynew
COMMON x_region,x_region
COMMON y_region,y_region
COMMON first,first
COMMON xfirst,xfirst
COMMON yfirst,yfirst

;***************************************
;CASE LAST POINT : CLOSES CONTOUR
;***************************************
IF (keyword_set(keyword_stop)) THEN BEGIN
xnew=xfirst
ynew=yfirst
IF (n_elements(x_region) gt 1) THEN BEGIN
x_region=x_region(1:*)
y_region=y_region(1:*)
ENDIF
ENDIF

;***************************************
;CASE FIRST POINT : UPDATE X AND Y
;KEEP LOCATION FIRST POINT IN MIND
;TO CLOSE CONTOUR AT LAST POINT
;***************************************
IF first THEN BEGIN
xfirst=xnew
yfirst=ynew
xold=xnew
yold=ynew
first=0b
ENDIF ELSE BEGIN

;***************************************
;CASE NOT FIRST POINT :
;***************************************

;*********************************
;ADDS ARC OF PARALLEL TO CONTOUR
;IF KEYWORD SET
;*********************************
IF (keyword_set(keyword_parallel)) THEN BEGIN
sirf_to_ll,pole,xold,yold,latold,lonold
sirf_to_ll,pole,xnew,ynew,latnew,lonnew

IF (lonnew lt 0) THEN lonnew1=lonnew+360.0 ELSE $
lonnew1=lonnew
IF (lonold lt 0) THEN lonold1=lonold+360.0 ELSE $
lonold1=lonold

IF lonnew1 ge lonold1 THEN step=0.5 ELSE step=-0.5
FOR lon=lonold1+step,lonnew1,step DO BEGIN
lon1=lon
IF (lon gt 180.0) THEN lon1=lon-360.0
ll_to_sirf,pole,latold,lon1,xi,yi
plots,[xi,xi],[yi,yi],/device,color=255
x_region=[x_region,xi]
y_region=[y_region,yi]
ENDFOR

ll_to_sirf,pole,latold,lonnew,xold,yold

;*********************************
;OTHERWISE ADDS A LINE TO CONTOUR
;*********************************
ENDIF ELSE BEGIN

plots,[xold,xnew],[yold,ynew],/device,color=255
step=-1+2*(xnew ge xold)
FOR xi=xold+step,xnew,step DO BEGIN
x_region=[x_region,xi]
y_region=[y_region,nint(((ynew-yold)/float(xnew-xold))*$
(xi-xnew)+ynew)]
ENDFOR

xold=xnew
yold=ynew

;*********************************
ENDELSE

;*****************************************
ENDELSE

;*********************************************
END
```

STATS_MAKING_ROUTINES

```
;*************************************************
;*************************************************
;                COMPUTE_MEAN_CYCLES.PRO
;*************************************************
;*************************************************
;This routine computes the mean cycles of the
;statistics according to the parameters given in
;input
;
;Input: region : region number
;                type: 1=a, 2=b
;                years_used: yyyy integers/string array
;                containing years used to process mean cycles
;                sampling: sampling of pdf
;                output_file_path: path from root of file
;                                  to write output to
;
;Output: file containing mean cycles, path
;                 given in input
;
;Calls: read_file.pro
;                config.pro
;                rename_structure.pro
;                make_weights.pro
;                make_dirs_to_file.pro
;                read_regions_equivalents.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*******************************************************
;*******************************************************
PRO compute_mean_cycles,region,type,$
years_used,sampling,output_file_path

;*************************************
;MAKES USEFUL PARAMETERS AND CREATE
;VARIABLES
;*************************************
years=years_used(where(years_used ne ''))
n_years=n_elements(years)
read_file,config('paths','stats',region,type,years(0),$
sampling),'stats',file_header,file_data
what=file_header.what
st=file_data(0)
sorted_data=replicate(st,n_years,122)
mean_cycles=replicate(st,122)

;*************************************
;LOADS STATS FOR EACH YEAR AND STORES
;THEM IN AN ARRAY
;*************************************
FOR i=0,n_years-1 DO BEGIN

file_path=config('paths','stats',region,$
type,years(i),sampling)
print,'loading '+file_path
read_file,file_path,'stats',file_header,file_data0
file_data=replicate(st,122)
FOR j=0,121 DO file_data(j)=$
rename_structure(file_data0(j),st)

sorted_data(i,*)=reform(file_data,1,122)

ENDFOR

;*************************************
;FILLS MEAN CYCLES HEADER (USEFUL ?)
;*************************************
mean_cycles.header.image_available=1b
mean_cycles.header.ice_contour_available=1b

;*************************************
;MAKES WEIGHTED MEAN OF LOADED STATS
;WEIGHTS GIVEN BY MAKE_WEIGHTS.PRO
;*************************************
FOR i=0,121 DO BEGIN
FOR j=1,3 DO BEGIN
weights_array=make_weights(sorted_data(*,i).(j))
FOR k=0,5 DO BEGIN
weighted_array=sorted_data(*,i).(j).(k)
IF (k eq 2) THEN BEGIN
FOR l=0,n_years-1 DO weighted_array(*,l)=$
weighted_array(*,l)*weights_array(l)
mean_cycles(i).(j).(k)=avg(weighted_array,1)
ENDIF ELSE BEGIN
FOR l=0,n_years-1 DO weighted_array(l)=$
weighted_array(l)*weights_array(l)
mean_cycles(i).(j).(k)=avg(weighted_array)
ENDELSE
ENDFOR
ENDFOR
ENDFOR

;*************************************
;MAKES OUPUT FILE HEADER
;*************************************
output_file_header=config('structures','st1')
output_file_header.what=what+3b
output_file_header.region=region
output_file_header.type=type
output_file_header.n_images=122
output_file_header.sampling=sampling

;*************************************
;CHECKS IF DESTINATION FILE PARENT
;DIRECTORIES EXIST. IF NOT, CREATE THEM
;WRITES HEADER AND MEAN CYCLES INTO FILE
;*************************************
openw,unit,output_file_path,/get_lun,error=err
IF (err ne 0) THEN BEGIN
make_dirs_to_file,output_file_path
openw,unit,output_file_path,/get_lun
ENDIF

writeu,unit,output_file_header
writeu,unit,mean_cycles
free_lun,unit

;*************************************
;WARNS WHEN COMPUTATION FINISHED
;*************************************
print,output_file_path+' computed'
print,''

;*********************************************
END


;*******************************************************
;*******************************************************
;                COMPUTE_STATS.PRO
;*******************************************************
;*******************************************************
;This routine computes the probability density
;function,mean, standard deviation and median for
;the values given in input
;
;Input: -backscat_values: 2B integers from .sir image
;                (may not be all values because of region mask)
;                VALUES SHOULD NOT BE CONVERTED TO FLOATS
;               -header: header containing info about .sir image
;                (see 'structures' in config.pro)
;               -sampling: desired sampling for PDF
;
;Output: -stats: structure containing stats
;                 (see 'structures in config.pro)
;                -success: byte indicating succes or
;                 failure of computation (0=failure, 1=success)
;
;Calls: config.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*******************************************************
;*******************************************************
PRO compute_stats,backscat_values,header,$
sampling,stats,success

;*********************************
;LOADS USEFUL PARAMETERS ABOUT PDF
;AND .SIR DATA CONVERSION PARAMETERS
;FROM iNTEGERS TO FLOATS
;*********************************
pdf_params=config('data','pdf_params',$
2-(header.region mod 2),header.type)
pdf_artefacts=config('data','pdf_artefacts',$
2-(header.region mod 2),header.type)
pdf_range=config('data','pdf_range',$
2-(header.region mod 2),header.type)

offset=float(pdf_params(2))
scale_factor=float(pdf_params(3))
min_good_value=pdf_params(0)
max_good_value=pdf_params(1)

range_min=pdf_range(0)
range_max=pdf_range(1)

;*********************************
;PREPARES FILTER FOR "BAD VALUES"
;*********************************
```

```
w=where((backscat_values ge min_good_value) and $
(backscat_values le max_good_value),count)


;*************************************
;IF NO GOOD VALUES, ABORTS
;*************************************
IF (count eq 0) THEN BEGIN
success=0
goto,pdf_failed
ENDIF

;*************************************
;OTHERWISE CONTINUE AND
;FILTERS BAD VALUES
;*************************************
good_backscat_values=backscat_values(w)

;*************************************
;COMPUTES SAMPLING ON FULL RANGE
;TO GET DESIRED SAMPLING ON RANGE
;SPECIFIED IN CONFIG.PRO
;*************************************
sampling_full_range=$
fix(sampling*(max_good_value-min_good_value+1.)/$
((range_max-range_min+1.0/scale_factor)*scale_factor))

;*************************************
;REMOVES ARTEFACTS VALUES
;(~ DIRACS IN FIRST HISTOGRAM)
;*************************************
FOR j=0,n_elements(pdf_artefacts)/2-1 DO BEGIN
artefact_interval=pdf_artefacts(0:1,j)
w1=where((good_backscat_values lt artefact_interval(0)) or $
(good_backscat_values gt artefact_interval(1)),count1)
good_backscat_values=good_backscat_values(w1)
w2=where((good_backscat_values ge (artefact_interval(0)-50)) $
and (good_backscat_values le (artefact_interval(1)+50)),count2)
FOR k=0, artefact_interval(1)-artefact_interval(0) DO $
IF (count2/100 ne 0) THEN good_backscat_values=$
[good_backscat_values,$
replicate(artefact_interval(0)+k,count2/100)]
ENDFOR

;*********************************
;MAKES HISTOGRAM FOR FULL RANGE
;WITH RIGHT SAMPLING
;*********************************
n=(max_good_value-min_good_value+1)/sampling_full_range
h_sampling_full_range=$
histogram(good_backscat_values,min=min_good_value,$
max=max_good_value,binsize=n)

;*********************************
;NORMALIZES HISTOGRAM TO GET PDF
;*********************************
norm_factor=total(h_sampling_full_range)*$
((max_good_value-min_good_value+1)/scale_factor)/$
(sampling_full_range-1)
pdf=h_sampling_full_range/norm_factor

;*********************************
;RESIZES PDF TO GET GOOD RANGE/SAMPLING
;*********************************
u=(fix((range_min-offset-32767.0/scale_factor)*$
scale_factor)-min_good_value)/n
pdf=pdf(u:u+sampling-1)

;*********************************
;COMPUTES BLACK ZONES RATIO
;*********************************
n_good_pix=n_elements(good_backscat_values)
black_zones_ratio=1.0-n_good_pix/$
float(n_elements(backscat_values))

;*********************************
;COMPUTES MEAN, STD, MEDIAN
;OVER GOOD VALUES CONVERTED TO FLOATS
;*********************************
good_backscat_values=(good_backscat_values+32767.0)/$
scale_factor+offset
mean=avg(good_backscat_values)
std=sigma(good_backscat_values)
median=median(good_backscat_values)

;*****************************************
; STORES STATS IN A STRUCTURE
;*****************************************
stats=config('structures','st3',sampling)
stats.black_zones_ratio=black_zones_ratio
stats.pdf=pdf
stats.mean=mean
stats.std=std
stats.median=median

;*************************************
;SAYS THAT STATS COMPUTATION IS A SUCCESS
;*************************************
success=1

;*************************************
;CASE PDF FAILED
;*************************************
pdf_failed:

;*************************************************
END


;*************************************************
;*************************************************
;                INTERFACE_STATS.PRO
;*************************************************
;*************************************************
;This routine sets up graphical interface which
;enables the user to compute statistics as well as
;mean cycles on the primary data (ERS images)
;
;Input: none
;
;Output: displays graphical interface
;
;Calls:         config.pro
;               read_regions_equivalents.pro
;               create_om_structure.pro
;               compute_mean_cycles.pro
;               process_list.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*************************************************
;*************************************************
PRO interface_stats

;*****************************************
;*****************************************
;    INTERFACE WIDGETS DEFINITION
;*****************************************
;*****************************************

;*************************************
;INTERFACE VARIABLES DEFINITION AND
;INITIALIZATION
;*************************************
COMMON variables_stats,variables_stats
st9=config('structures','st9')
variables_stats=st9

;*************************************
;OTHERS COMMON VARIABLES DEFINITION
;*************************************
COMMON etc_stats,etc_stats
etc_stats=0.0
COMMON computation_times_stats,computation_times_stats
computation_times_stats=$
config('data','computation_times_stats')
COMMON files_to_compute,files_to_compute
files_to_compute=replicate(st9,1)
COMMON files_to_compute_str,files_to_compute_str
files_to_compute_str=['']

;*********************************
;               MAIN WINDOW
;*********************************
COMMON mw_stats,mw_stats
COMMON mw_menu,mw_menu
mw_stats=wwmainwindow(mw_menu,layout_stats,/vertical,$
title='Statistics Computation',position=[100,100])

;*********************************
;               FIRST LAYOUT
;*********************************
ly1=wwlayout(layout_stats)

;*****************************
;               WHAT OPTION MENU
;*****************************
COMMON what_om,what_om
what_om=wwoptionmenu(ly1,'Compute ',$
{,callback:'what_stats_cb',button:$
'Sea/Land/All_Ice_Statistics',$
button:'Sea_Ice_Statistics',button:$
'Land_Ice_Statistics',$
button:'Mean_Cycles'})
```

```
;****************************
;                REGION OPTION MENU
;****************************
COMMON region_om,region_om
read_regions_equivalents,n,numbers,names,short_names
region_om_structure=$
create_om_structure(['region_stats_cb',names])
region_om=wwoptionmenu(ly1,'for ',region_om_structure)

;****************************
;                TYPE OPTION MENU
;****************************
COMMON type_om,type_om
type_om=wwoptionmenu(ly1,'type ',$
{,callback:'type_stats_cb',button:'a',button:'b'})

;*********************************
;                SECOND LAYOUT
;*********************************
ly2=wwlayout(layout_stats)

;****************************
;                YEARS LIST
;****************************
tx21=wwtext(ly2,text='years ',/label)
COMMON years_ls,years_ls
years_ls=wwlist(ly2,config('interfaces',$
'years_array'),'years_stats_cb','null_cb',/multi)

;****************************
;                SAMPLING LIST
;****************************
tx22=wwtext(ly2,text='sampling ',/label)
COMMON sampling_ls,sampling_ls
sampling_ls=wwlist(ly2,config('interfaces',$
'sampling_array'),'sampling_stats_cb','null_cb',/multi)

;*********************************
;                THIRD LAYOUT
;*********************************
ly3=wwlayout(layout_stats)

;****************************
;                FILES LIST
;****************************
tx31=wwtext(ly2,text='Files to be computed : ',/label)
COMMON files_ls,files_ls
files_ls=wwlist(ly2,$
['                              '],$
'null_cb','null_cb')

;****************************
;ESTIMATED COMPUTATION
;TIME LABEL
;****************************
COMMON tx4,tx4
tx4=wwtext(layout_stats,text=$
'Estimated Computation Time :  0j  0h  0min',/label)
tx5=wwtext(layout_stats,text='',/label)

;****************************
;ADD/COMPUTE/RESET/CANCEL BUTTONS
;****************************
go_stats_bb=wwbuttonbox(layout_stats,$
['Add','Compute','Reset','Cancel'],'go_stats_cb')

;*********************************
;INITIALIZES WIDGETS
;*********************************
status=wwsetvalue(what_om,1)
status=wwsetvalue(region_om,1)
status=wwsetvalue(type_om,1)

;*********************************
;DISPLAYS INTERFACE
;*********************************
status=wwsetvalue(mw_stats,/display)

;*********************************
END


;*****************************************************
;*****************************************************
;                CALLBACK PROCEDURES
;*****************************************************
;*****************************************************

;*********************************************
;                NULL CALLBACK
;*********************************************
PRO null_cb,wid,index
END

;*********************************************
;                UPDATES 'WHAT'
;*********************************************
PRO what_stats_cb,wid,index
COMMON variables_stats,variables_stats
variables_stats.what=config('convert','what',$
wwgetvalue(wid))
END

;*********************************************
;                UPDATES REGION
;*********************************************
PRO region_stats_cb,wid,index
COMMON variables_stats,variables_stats
variables_stats.region=config('convert',$
'region_long',wwgetvalue(wid))
END

;*********************************************
;                UPDATES TYPE
;*********************************************
PRO type_stats_cb,wid,index
COMMON variables_stats,variables_stats
variables_stats.type=config('convert','type',$
wwgetvalue(wid))
END

;*********************************************
;                UPDATES YEARS
;*********************************************
PRO years_stats_cb,wid,index
COMMON variables_stats,variables_stats
variables_stats.years(*)=''
variables_stats.years=wwgetvalue(wid)
END

;*********************************************
;                UPDATES SAMPLING
;*********************************************
PRO sampling_stats_cb,wid,index
COMMON variables_stats,variables_stats
variables_stats.sampling(*)=''
variables_stats.sampling=wwgetvalue(wid)
END

;*********************************************
;                MANAGES ADD/COMPUTE/RESET/CANCEL
;*********************************************
PRO go_stats_cb,wid,index

COMMON mw_stats,mw_stats
COMMON variables_stats,variables_stats
COMMON files_to_compute,files_to_compute
COMMON files_to_compute_str,files_to_compute_str

CASE index of

;*************************************
;                ADDS FILES
;*************************************
1: BEGIN

files_to_compute=[files_to_compute,variables_stats]

w=where(variables_stats.years ne '',n_years)
w=where(variables_stats.sampling ne '',n_sampling)

;*********************************
;CHECKS IF ENTRIES ARE VALID
;OTHERWISE GO AT END OF PROGRAM
;(PB LABEL)
;*********************************
IF (n_years eq 0 or n_sampling eq 0) THEN BEGIN
compute_al=wwalert(mw_stats,'Selection Not Valid',$
['OK'],/error,title='ERROR')
go_stats_cb,wid,3
goto,pb
ENDIF

region=variables_stats.region
type=variables_stats.type

;*********************************
CASE variables_stats.what of

;**********************
;    MEAN CYCLES :
;    UPDATES NAMES
;**********************
4:BEGIN
FOR k=0,n_sampling-1 DO BEGIN
sampling=variables_stats.sampling(k)
```

```
path=config('paths','mean_filename',region,type,$
variables_stats.years,sampling)
files_to_compute_str=[files_to_compute_str,path]
ENDFOR
END

;**********************
;   STATISTICS
;**********************
ELSE: BEGIN

;************
;UPDATES NAMES
;OF FILES
;TO COMPUTE
;************
FOR j=0,n_years-1 DO BEGIN
FOR k=0,n_sampling-1 DO BEGIN
year=variables_stats.years(j)
sampling=variables_stats.sampling(k)
path=config('paths','stats_filename',$
region,type,year,sampling)
files_to_compute_str=[files_to_compute_str,path]
ENDFOR
ENDFOR

;****************
;UPDATES COMPUTATION TIME
;****************
IF (variables_stats.what gt 0 and $
variables_stats.what le 3) THEN BEGIN
COMMON etc_stats,etc_stats
COMMON computation_times_stats,computation_times_stats

add_time=n_years*n_sampling*$
computation_times_stats(variables_stats.what-1)
etc_stats=etc_stats+add_time

j=string(etc_stats/(24*60),format='(i2)')
h=string((etc_stats mod (24*60l))/60,format='(i2)')
min=string(etc_stats mod 60,format='(i2)')

etc_stats_str='Estimated Computation Time : '+$
j+'j '+h+'h '+min+'min'

COMMON tx4,tx4
status=wwsetvalue(tx4,etc_stats_str)
ENDIF

;************
;REINITIALIZES
;VARIABLES_STATS
;************
variables_stats.years(*)=''
variables_stats.sampling(*)=''

END

;*********************************
ENDCASE

;************************
;   UPDATES WIDGETS
;************************
COMMON files_ls,files_ls
COMMON years_ls,years_ls
COMMON sampling_ls,sampling_ls

status=wwsetvalue(files_ls,files_to_compute_str(1:*))
status=wwsetvalue(years_ls,$
config('interfaces','years_array'))
status=wwsetvalue(sampling_ls,$
config('interfaces','sampling_array'))
status=wwsetvalue(mw_stats,/update)

END

;***************************************
;     COMPUTES SPECIFIED FILES
;***************************************
2: BEGIN

;**************
;ALERT BOX
;**************
compute_al=wwalert(mw_stats,$
['Statistics Being Processed...'],$
title='INFO',/noconfirm,working=1,nowait=1)

;**************
files_to_compute=files_to_compute(1:*)
n_files=n_elements(files_to_compute)

;**************
print,''
print,'***********************************'
print,'Beginning Statistics Computation
print,'***********************************'
print,''

;***********************************
;BEGINS COMPUTATION LOOP
;***********************************
FOR i=0,n_files-1 DO BEGIN

;********************
;USEFUL PARAMETERS
;********************
n_years=$
n_elements(where(files_to_compute(i).years ne ''))
n_sampling=$
n_elements(where(files_to_compute(i).sampling ne ''))
region=files_to_compute(i).region
type=files_to_compute(i).type

;******************************
CASE files_to_compute(i).what of

;******************************
;MEAN_CYCLES COMPUTATION
;******************************
4:BEGIN
FOR k=0,n_sampling-1 DO BEGIN
sampling=files_to_compute(i).sampling(k)
path=config('paths','mean',region,type,$
files_to_compute(i).years,sampling)
compute_mean_cycles,region,type,$
files_to_compute(i).years,sampling,path
ENDFOR
END

;******************************
;STATISTICS COMPUTATION
;******************************
ELSE: BEGIN
FOR j=0,n_years-1 DO BEGIN
FOR k=0,n_sampling-1 DO BEGIN
year=files_to_compute(i).years(j)
sampling=files_to_compute(i).sampling(k)
path=config('paths','stats',region,type,year,sampling)
process_list,files_to_compute(i).what,$
region,type,year,'0101',year,'1231',sampling,path
ENDFOR
ENDFOR
END

;******************************
ENDCASE

;***********************************
;     END OF COMPUTATION
;***********************************
ENDFOR

print,'***********************************'
print,'      Statistics Computed
print,'***********************************'
print,''

wwalertpopdown,compute_al
status=wwsetvalue(mw_stats,/close)

END

;****************************************
; RESET FILES TO COMPUTE
;****************************************
3: BEGIN
variables_stats=config('structures','st9')
files_to_compute=replicate(variables_stats,1)
files_to_compute_str=['']

COMMON what_om,what_om
COMMON region_om,region_om
COMMON type_om,type_om
COMMON years_ls,years_ls
COMMON sampling_ls,sampling_ls
COMMON files_ls,files_ls
COMMON tx4,tx4

status=wwsetvalue(what_om,1)
status=wwsetvalue(region_om,1)
status=wwsetvalue(type_om,1)
status=wwsetvalue(years_ls,$
config('interfaces','years_array'))
status=wwsetvalue(sampling_ls,$
```

```
config('interfaces','sampling_array'))
status=wwsetvalue(files_ls,$
[' ']) 
status=wwsetvalue(tx4,$
'Estimated Computation Time :  0j  0h  0min')
status=wwsetvalue(mw_stats,/update)
END

;*************************************
;          EXITS INTERFACE
;*************************************
4: status=wwsetvalue(mw_stats,/close)

;*************************************
ENDCASE

;**********************************
;FILE ENTRIES WRONG
;**********************************
pb:

;**********************************
END


;*****************************************************
;*****************************************************
;                    MAKE_CENSUS.PRO
;*****************************************************
;*****************************************************
;This routine takes the census of primary .sir
;images/ice contours bank. It returns an array of
;structures describing the images and their states
;(availability, ice contour availability,region,type,
;year,day1 of acquisition,satellite,number of days of
;average)
;
;Rmk: Since this routine calls the FINDFILE procedure
;which needs a lot of space, it is likely not to
;work on a less than 128MB RAM machine
;
;Input: none
;
;Output: file containing census structures array
;
;Calls: config.pro
;       make_dirs_to_file.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
PRO make_census

;*************************************
;ALERT BOX : TAKING CENSUS
;*************************************
COMMON mw_menu,mw_menu
census_al=wwalert(mw_menu,['Taking Census.',$
'See real-time report on command line ...'],$
title='IMAGES BANK CENSUS',/noconfirm,/working,/nowait)

;*************************************
;BEGINS CENSUS : LOOP ON
;ARCTIC/ANTARCTIC AND A/B
;*************************************
FOR region=1,2 DO $
FOR type=1,2 DO BEGIN

region_str=config('convert','region',region)
type_str=config('convert','type',type)

print,' '
print,'***********************************************'
print, 'Beginning census '+region_str+'-'+type_str
print,'***********************************************'
print,' '

;*************************************
;GETS PATHS OF IMAGES IN IMAGES BANK
;*************************************
print,'Getting existing images paths ...'
a=findfile('/rime2/escat/*-'+type_str+'-*')
b=strpos(a,region_str)
images_paths=a(where(b ne -1))

;*************************************
;TRANSFORMS PATHS TO EQUALLY
;DESCRIPTIVE HEADER
;*************************************
print,'Transforming paths to headers ...'

n=n_elements(images_paths)
headers=replicate(config('structures','st2'),n)

FOR k=0,n-1 DO BEGIN
ssss=strmid(images_paths(k),0,4)
t=strmid(images_paths(k),5,1)
Rrr=strmid(images_paths(k),7,3)
yy=strmid(images_paths(k),10,2)
bbb=fix(strmid(images_paths(k),13,3))
eee=fix(strmid(images_paths(k),17,3))

headers(k).region=config('convert','region',Rrr)
headers(k).type=config('convert','type',t)
headers(k).year=fix(yy_to_yyyy(yy))
headers(k).day1=bbb
headers(k).satellite=config('convert','satellite',ssss)
IF (eee ge bbb) THEN headers(k).n_avg=eee-bbb+1 ELSE $
headers(k).n_avg=366+is_leap(headers(k).year)+eee-bbb

ENDFOR

headers.image_available=1b

;*************************************
;REMOVES HEADERS OF IMAGES WHOSE DAY1
;IS NOT A MULTIPLE OF 3
;*************************************
print,'Removing weird data ...'
w_good=where((headers.day1 mod 3) eq 0)
headers=headers(w_good)

;*************************************
;SORTS HEADERS BY DATES AND REMOVES
;DUPLICATE ONES (SOMETIMES SEVERAL
;IMAGES FOR SAME DAY EX : AVERAGED 6/7
;DAYS, TAKEN BY ERS1/ERS2, ETC...)
;*************************************
print,'Sorting headers by dates and removing double data ...'

;********************
;SORTS
;********************
dates_array=headers.year*1000l+headers.day1
w=sort(dates_array)

result=[0l]
k=0
end_w=n_elements(w)-1

;********************
;MAKE DUPLICATE GROUPS
;********************
REPEAT BEGIN
IF (dates_array(w(k)) ne dates_array(w(k+1))) THEN BEGIN
result=[result,w(k)]
k=k+1
ENDIF ELSE BEGIN
multi=[w(k),w(k+1)]
k=k+2
WHILE (dates_array(w(k)) eq dates_array(w(k-1))) DO BEGIN
multi=[multi,w(k)]
k=k+1
ENDWHILE

;********************
;CHOOSE ONE AMONG
;EACH DUPLICATE GROUP
;CHOICE POLICY SPECIFIED
;IN CONFIG.PRO
;********************
preference=config('data','preference')
sat_and_n_avg_array=[transpose(headers(multi).satellite),$
transpose(headers(multi).n_avg)]

i=0
REPEAT BEGIN
index=where((sat_and_n_avg_array(0,*) eq preference(0,i)) $
and (sat_and_n_avg_array(1,*) eq preference(1,i)),count)
i=i+1
ENDREP until (count ne 0)

result=[result,multi(index)]
ENDELSE

ENDREP until (k ge end_w)

result=[result,w(end_w)]

headers=headers(result(1:*))

;*************************************
;FILLS THE GAPS IN THE LIST OF HEADERS
;WITH 'BLANK' HEADERS
```

```
;*************************************
print,'Filling the gaps ...'

fill_in_header=config('structures','st2')

fill_in_header.image_available=0b
fill_in_header.region=headers(0).region
fill_in_header.type=headers(0).type
fill_in_header.satellite=0b
fill_in_header.n_avg=0b

years=unique(headers.year)

FOR k=0,n_elements(years)-1 DO BEGIN
year=years(k)
w1=where(headers.year eq year)
day1_array=headers(w1).day1
FOR j=0,121 DO BEGIN
w2=where(day1_array eq 3*j,count2)
IF (count2 eq 0) THEN BEGIN
fill_in_header.year=year
fill_in_header.day1=3*j
headers=[headers,fill_in_header]
ENDIF
ENDFOR
ENDFOR

dates_array=headers.year*1000l+headers.day1
headers=headers(sort(dates_array))

;*************************************
;CHECKS ICE CONTOUR AVAILABILITY
;AND UPDATES CORRESPONDING FIELD
;FOR EACH HEADER
;*************************************
print,'Checking ice contour availability ...'

ice_contours_available_20th=$
findfile(config('paths','contours_bank')+'/'+$
strupcase(config('convert','region_long',region))+$
'/19*')
ice_contours_available_21th=$
findfile(config('paths','contours_bank')+'/'+$
strupcase(config('convert','region_long',region))+$
'/2*',count=ctr)

ice_contours_available_20th=ice_contours_available_20th($
where(strpos(ice_contours_available_20th,'con_') ne -1))

IF (ctr ne 0) THEN $
ice_contours_available_20th=ice_contours_available_21st($
where(strpos(ice_contours_available_21st,'con_') ne -1))

IF (ctr eq 0) THEN ice_contours_available=$
ice_contours_available_20th ELSE $
ice_contours_available=$
[ice_contours_available_20th,ice_contours_available_21th]

;************************
;CASE ICE CONTOURS BANK
;EMPTY : PROBLEM
;************************
IF (ice_contours_available(0) eq '') THEN BEGIN
wwalertpopdown,census_al
print,''
print,'*****************************************'
print,'*****************************************'
print,'Contours found: None'
print,'Re run TOPICS and compute ice contours
print,'*****************************************'
print,'*****************************************'
print,''
goto,problem
ENDIF

l=strlen(ice_contours_available(0))
FOR k=0,n_elements(ice_contours_available)-1 DO BEGIN
ice_contour_yyyyddd=date_to_number(yy_to_yyyy($
strmid(ice_contours_available(k),l-6,2))+$
strmid(ice_contours_available(k),l-4,4))
ice_contour_yyyy=fix(strmid(ice_contour_yyyyddd,0,4))
ice_contour_ddd=fix(strmid(ice_contour_yyyyddd,4,3))
w=where((headers.year eq ice_contour_yyyy) and $
(headers.day1 eq ice_contour_ddd),count)
IF (count ne 0) THEN headers(w).ice_contour_available=1b

;*************************************************
;END OF CENSUS LOOP
;*************************************************
ENDFOR

;*************************************
;WRITES HEADER ARRAY TO FILE
;SPECIFIED BY CONFIG.PRO
;*************************************
census_path=config('paths','census',region,type)

openw,unit,census_path,/get_lun,error=err
if err ne 0 then begin
make_dirs_to_file,census_path
openw,unit,census_path,/get_lun
endif
writeu,unit,headers
free_lun,unit

;*************************************
;MAKES CENSUS REPORT ON COMMAND LINE
;*************************************
print,'Census '+region_str+'-'+type_str+' completed'
print,''
print,'*************************************************'
print,' years images contours'
print,'*************************************************'

years=unique(headers.year)
FOR k=0,n_elements(years)-1 DO BEGIN
year=years(k)
w1=where(headers.year eq year,count1)
w2=where((headers.year eq year) and $
(headers.ice_contour_available eq 1),count2)
print,year,count1,count2
ENDFOR

print,'*************************************************'
print,'Total :',n_elements(headers),' images'
print,'*************************************************'
ENDFOR

print,''
print,'*************************************************'
print,'Census completed, Data bank updated'
print,'*************************************************'
print,''

;*************************************
;CLOSES ALERT BOX
;*************************************
wwalertpopdown,census_al

;*************************************
;ALERT BOX : CENSUS DONE
;*************************************
census_al=wwalert(mw_menu,'Images Bank Census Done',$
title='INFO',/noconfirm,after=3)

;*************************************************
problem:

;*************************************************************
END


;**************************************************
;**************************************************
;       MAKE_EXTENT_GRID.PRO
;**************************************************
;**************************************************
;This routine doesn't give good results so far, so it
;is not used by TOPICS. However, I think it would be
;rewarding to check what's wrong here, and use it in
;TOPICS.
;
;The goal of this routine is to make a grid for .SIR
;images, with the value of a given pixel being the
;area on Earth whose polar stereographic projection
;is the (assumed) square pixel on SIR images. This
;is not a trivial pb, since the polar stereo proj.
;is not an equal area-one, and because of the fact
;that line_coordinates in the projection plane
;originate from circles on the globe passing by the
;south pole and included in the plane formed by the
;south pole and the line in the projection plane
;
;Definitely, the formulas below are not good, but
;may serve as a work basis. They were obtained by
;differential calculus followed by double integration.
;
;Rmq: pp means projection plane
;     c is for center of .SIR image
;     dx_pp is the pixel-size(km) in the pp
;     i/j is the pixel array-coordinates
;     lat0 is the latitude of the border of the
;     polar region
;     theta is the angle between the Earth
;     rotation axis and the line joining the south
;     pole to a given point
```

```
;    subscripts 1/2 refer to the x=cte
;    and y=cte pix borders
;
;Input: pole number (1=Arc 2=Ant)
;
;Does: compute grid described above, with a size adapted
;                to the input polar region
;
;Calls:          none
;
;Written by/in: Pierre Mercier 07/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
PRO make_extent_grid,pole

;*********************************
;Computing useful values
;*********************************
image_size=config('data','image_size',pole)
n=image_size(0)
extent_grid=fltarr(n,n)

pi=!DPI
Rt=6378.273
image_ll=config('data','image_ll',pole)
lat0=pi*image_ll(0)/180.

dx_pp=2*Rt*tan(pi/4-lat0/2)/n
ic_pp=0.5*(n-1)
jc_pp=0.5*(n-1)

;*************************************
;Computes grid value for each pixel
;*************************************
for i=0,n-1 do begin
for j=0,n-1 do begin

x1_pp=((i-ic_pp)-0.5)*dx_pp
x2_pp=((i-jc_pp)+0.5)*dx_pp

y1_pp=((j-ic_pp)-0.5)*dx_pp
y2_pp=((j-jc_pp)+0.5)*dx_pp

theta_x1=atan(x1_pp/(2.0*Rt))
theta_x2=atan(x2_pp/(2.0*Rt))

theta_y1=atan(y1_pp/(2.0*Rt*cos(theta_x1)))
theta_y2=atan(y2_pp/(2.0*Rt*cos(theta_x2)))

ds_sphere=4*Rt^2.*abs(theta_y2-theta_y1)*$
abs((theta_x2-theta_x1)+(sin(2.*theta_x2)-$
sin(2.*theta_x2))/2.)

extent_grid(i,j)=ds_sphere

endfor
endfor

;*********************************
;Writes result in a file
;*********************************
path=config('paths','extent_grid',pole)
openw,unit,path,/get_lun
writeu,unit,extent_grid
free_lun,unit

;*********************************
END



;*****************************************************
;*****************************************************
;                    MAKE_MASK.PRO
;*****************************************************
;*****************************************************
;This routine makes a mask for .sir images which
;enables to locate sea ice,land ice (and land)
;for a given region and at a given time
;
;
;Input: region : region number
;                year : yyyy integer specifying year
;                day1: ddd integer specifying day
;                what: 1=Sea/Land/All ice investigation
;                    (program returns wsi,wli,wai)
;                    2=Sea ice investigation
;                    (program returns only wsi)
;                    3=Land ice investigation
;                    (program returns only wli)
;                keyword /already_prepared avoids loading
;                region mask if an already prepared mask
;                is provided (ex: call from process_list.pro)
;
;Output: mask (byte array) 1=land,2=sea ice,3=land ice
;                 wsi,wli,wai if what eq 1
;                 wsi if what eq 2
;                 wli if what eq 3
;
;Calls:          config.pro
;                read_file.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
PRO make_mask,region,year,day1,what,mask,$
wsi,wli,wai,mask_already_prepared=keyword_mask

;*************************************
;COMPUTES POLE NUMBER
;*************************************
pole=2-(region mod 2)

;*************************************
;LOADS .SIR IMAGE SIZE
;*************************************
image_size=config('data','image_size',pole)
n=image_size(0)

;*************************************
;IF MASK NOT PREVIOUSLY PREPARED
;BY CALLING ROUTINE,...
;*************************************
IF not(keyword_set(keyword_mask)) THEN BEGIN

;*********************************
;...LOADS AND APPLY REGION MASK
;*********************************
IF (region gt 2) THEN BEGIN
region_mask_path=config('paths','region_mask',region)
read_file,region_mask_path,'region_contour_and_path',$
header,contour,region_mask
mask=replicate(4b,n,n)
mask(region_mask)=0b
ENDIF ELSE mask=bytarr(n,n)

;*********************************
;...LOADS AND APPLY LAND MASK
;*********************************
land_mask_path=config('paths','land_mask',pole)
read_file,land_mask_path,'land_mask',count,land_mask
mask(land_mask)=1

ENDIF

;*****************************************
;LOADS AND APPLY ICE MASK
;*****************************************
ice_contour_path=config('paths','ice_contour',$
pole,year,day1)
openr,unit,ice_contour_path,/get_lun

;************************
x=0
y=0
x_vector=intarr(1)
y_vector=intarr(1)

;************************
WHILE (not(eof(unit))) DO BEGIN

;************************
readf,unit,x,y,format='(i3,1x,i3)'

;************************
IF ((x eq 999) and (y eq 999)) THEN BEGIN
x_vector=x_vector(1:*)
y_vector=y_vector(1:*)
ice_mask=polyfillv(x_vector,y_vector,n,n)
mask(ice_mask)=mask(ice_mask)+2
x_vector=intarr(1)
y_vector=intarr(1)

;************************
ENDIF ELSE BEGIN
x_vector=[x_vector,x]
y_vector=[y_vector,y]
ENDELSE

;************************
ENDWHILE
```

```
;*************************
free_lun,unit


;****************************************
;RETURNS LOCATION OF CERTAIN TYPES
;OF ICE, DEPENDING ON VALUE OF 'WHAT'
;****************************************
CASE what of
1: BEGIN
wsi=where(mask eq 2)
wli=where(mask eq 3)
wai=[wsi,wli]
END
2: wsi=where(mask eq 2)
3: wli=where(mask eq 3)
ENDCASE

;****************************************
END


;**************************************************
;**************************************************
;                MAKE_WEIGHTS.PRO
;**************************************************
;**************************************************
;This routine makes the weights used in mean cycles
;computation
;
;Input: slice_to_average: structure array , each
;                 structure containing labeled statistics for
;                 the same day in year but for different years
;
;Output: float array -same size as input array-
;                 containing normalized weights
;                 i.e. sum(weights)=1.0 (except if all data
;                 are unacceptable, in which case all
;                 weights=0.0)
;
;Calls: none
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;**************************************************
;**************************************************
FUNCTION make_weights,slice_to_average

w=where(slice_to_average.black_zones_ratio ne 1.0,count)

;*******************************
;CASE AT LEAST ONE YEAR OF DATA
;IS OK
;*******************************
IF (count ne 0) THEN BEGIN

;*******************************
;MAKES ARRAYS OF PARAMETERS
;WHICH INFLUENCE WEIGHTING:
;- BLACK ZONES RATIO (E)
;- DEVIATION TO MEAN MEAN (M)
;- DEVIATION TO MEAN STD (S)
;*******************************
E=slice_to_average.black_zones_ratio

Emin=min(E)
w=where(E le Emin+0.1)

average_mean=avg((slice_to_average.mean)(w))
M=abs((slice_to_average.mean-average_mean)/average_mean)

average_std=avg((slice_to_average.std)(w))
S=abs((slice_to_average.std-average_std)/average_std)

;*******************************
;MAKES WEIGHTS
;(THE FORMULA BELOW IS AN A PRIORI
;ONE WHICH DOESN'T HAVE A DRAMATIC
;EFFECT COMPARED TO THE CASE WHERE
;ALL WEIGHTS ARE EQUAL. HOWEVER
;IT GIVES A GOOD IDEA ABOUT
;HOW WEIGHTING SHOULD TAKE
;BLACK ZONES RATIO AND MEAN/STD
;DEVIATION INTO ACCOUNT):
;THE HIGHER THE BLACK ZONES RATIO,
;THE LOWER THE WEIGHT.THE HIGHER
;THE STD/MEAN DEVIATION, THE LOWER
;THE WEIGHT, THE EFFECT INCREASING
;AS E IS BIG ((x+0.25)^2 DEPENDENCE).
;LIKE THIS, 'TOO DARK' STATS AND 'TOO
;STRONGLY MARKED' YEARS DO NOT
;INFLUENCE THE MEAN CYCLE MUCH.
;IF YOU FEEL LIKE IMPROVING THE
;FORMULA FEEL FREE TO DO SO.
;*******************************
weights_array=1.0-(E+(1+(E+0.25)^2)*(S+M))

;*******************************
;IF WEIGHT NEGATIVE, THEN SETS
;IT TO 0.0 i.e. STATS NOT TAKEN
;INTO ACCOUNT BECAUSE IMAGE TOO
;DARK(E TOO HIGH) AND/OR DEVIATION
;OF MEAN/STD TO MEAN VALUE TOO HIGH
;*******************************
w=where(weights_array lt 0.0,count)
IF (count ne 0) THEN weights_array(w)=0.0

;*******************************
;NORMALIZES WEIGHTS ARRAY SO
;THAT SUM=N_YEARS
;*******************************
n_years=n_elements(slice_to_average)
weights_array=weights_array*(n_years/total(weights_array))

;*******************************
;CASE NO YEAR IS OK : NULL WEIGHTS
;*******************************
ENDIF ELSE weights_array=$
replicate(0.0,n_elements(slice_to_average))

;*******************************
;RETURNS RESULT
;*******************************
RETURN,weights_array

;****************************************
END



;**************************************************
;**************************************************
;                PROCESS_IMAGE.PRO
;**************************************************
;**************************************************
;This routine manages statistics computation for a
;.sir image
;
;Input: -stats_header : statistics header previously
;                 created containing useful info about .sir
;                 image for statistics computation
;                -what: byte specifying on which ice domains
;                 statistics are to be computed : 1=sea/land/all
;                 ice 2=sea ice only 3=land ice only
;                -sampling : desired sampling for PDF
;                -common mask : mask common to all the images
;                 in a list, prepared by process_list.pro and
;                 passed as a reference to speed up computation
;
;Output: structure containing the computed statistics :
;                 probability density function, mean, standard
;                 deviation, median (for more details, see
;                 config.pro), of each desired ice domain,
;
;Calls:          config.pro
;                read_file.pro
;                make_mask.pro
;                compute_stats.pro
;
;Written by/in: Pierre Mercier 07/00
;
;Last updated: 08/31/00
;**************************************************
;**************************************************
FUNCTION process_image,stats_header,$
what,sampling,common_mask

;****************************************
;MAKES STRUCTURE TO CONTAIN STATS AND
;MAKES PARAMETERS DEFINING WHICH ICE
;DOMAINS ARE TO BE COMPUTED
;****************************************
CASE what OF
1: BEGIN
stats=config('structures','st4sla',sampling)
i_beg=1
i_end=3
END
2: BEGIN
stats=config('structures','st4s',sampling)
i_beg=1
i_end=1
END
3: BEGIN
stats=config('structures','st4l',sampling)
```

```
i_beg=2
i_end=2
END
ENDCASE

stats.header=stats_header

;*****************************************
;PROCESSES IMAGE ONLY IF IMAGE AND
;CONTOUR AVAILABLE
;*****************************************
IF not(stats_header.image_available) THEN BEGIN
IF not(stats_header.ice_contour_available) THEN $
print,'Image and contour missing' else $
print,'Image missing'
goto,not_processed
ENDIF
IF not(stats_header.ice_contour_available) THEN BEGIN
print,'Contour missing'
goto,not_processed
ENDIF

;*****************************************
;LOADS IMAGE
;*****************************************
image_path=config('paths','image',stats_header)
read_file,image_path,'.sir',info,image

;*****************************************
;FINISHES MAKING OF ALREADY
;PREPARED MASK FOR ICE DOMAINS
;DIFFERENTIATION
;*****************************************
mask=common_mask
make_mask,stats_header.region,stats_header.year,$
stats_header.day1,what,mask,wsi,wli,wai,$
/mask_already_prepared

;*****************************************
;BEGINS A LOOP OVER THE DESIRED ICE
;DOMAINS
;*****************************************
FOR i=i_beg,i_end DO BEGIN

;*****************************************
;MAKES BACKSCAT VALUES VECTOR
;*****************************************
CASE i OF
1: w=wsi
2: w=wli
3: w=wai
ENDCASE

;***************************
;STOPS IF NO BACKSCAT VALUES
;***************************
IF (w(0) ne -1) THEN BEGIN
backscat_values=image(w)

;*****************************
;ATTEMPT TO COMPUTE ICE EXTENT
;(DOESN'T WORK : SEE
;MAKE_EXTENT_GRID.PRO) SO THE
;VALUE OF ICE EXTENT IS 0.0
;*****************************
;image_size=config('data','image_size',$
;2-(stats_header.region mod 2))
;n=image_size(0)
;extent_grid=fltarr(n,n)
;extent_grid_path=config('paths','extent_grid',$
;2-(stats_header.region mod 2))
;openr,unit,extent_grid_path,/get_lun
;readu,unit,extent_grid
;free_lun,unit
;ice_extent=total(extent_grid(w))
;partial_ice_data.ice_extent=ice_extent

;*****************************
; COMPUTES STATISTICS
;*****************************
compute_stats,backscat_values,stats_header,$
sampling,partial_ice_data,success

;*****************************
;CASE STATS COMPUTATION FAILED
;*****************************
IF (success eq 0) THEN BEGIN
CASE what OF
1: stats=config('structures','st4sla',sampling)
2: stats=config('structures','st4s',sampling)
3: stats=config('structures','st4l',sampling)
ENDCASE
stats.header=stats_header
stats.header.image_available=0b
print, $
'Pdf failed : Domain may be too much black or too small'
CASE what OF
1: goto,not_processed_go_next
else: goto,not_processed
ENDCASE
ENDIF

;*************************************
;STORES STATISTICS IN STRUCTURE
;*************************************
CASE what OF
1: stats.(i)=rename_structure(partial_ice_data,stats.(i))
2: stats.(1)=rename_structure(partial_ice_data,stats.(1))
3: stats.(1)=rename_structure(partial_ice_data,stats.(1))
ENDCASE

ENDIF

;*************************************
;IMAGE NOT PROCESSED : NEXT ICE DOMAIN
;*************************************
not_processed_go_next:

ENDFOR

;*************************************
;IMAGE NOT PROCESSED : STOP
;*************************************
not_processed:

;*************************************

RETURN,stats

END



;*************************************************
;*************************************************
;                PROCESS_LIST.PRO
;*************************************************
;*************************************************
;This routine computes statistics over .sir images
;for given ice domains region, type, and time range.
;Several statistical parameters must also be
;specified (see input)
;
;Input: what: byte specifying on which ice domains
;          statistics are to be computed : 1=sea/land/all
;          ice 2=sea ice only 3=land ice only
;          region: region number
;          type: 1=a, 2=b
;          start_year: yyyy integer /string
;          start_day: mmdd integer/string
;          stop_year: yyyy integer /string
;          stop_day: mmdd integer/string
;          sampling: number of samples used to compute
;                    the probability density function
;                    (usually between 100 and 1000)
;          output_file_path : output file path
;
;Output: .stats unformatted file(s) composed of a header
;          and a list of structures containing the
;          statistics (see config.pro
;          for further details)
;
;Calls:    read_file.pro
;          config.pro
;          make_dirs_to_file.pro
;          process_image.pro
;
;Written by/in: Pierre Mercier 07/00
;
;Last updated: 08/31/00
;*************************************************
;*************************************************
PRO process_list,what,region,type,start_year,start_day,$
stop_year,stop_day,sampling,output_file_path

;*****************************************
;SELECTS IMAGES TO BE PROCESSED BY
;REFERING TO IMAGES BANK CENSUS
;*****************************************
read_file,config('paths','census',2-(region mod 2),type),$
'census',info,census

numbers=census.year*1000l+census.day1

start=long(date_to_number(start_year+start_day))
stop=long(date_to_number(stop_year+stop_day))
```

```
w=where((numbers ge start) and (numbers lt stop))
headers_list_of_images_to_process=census(w)

headers_list_of_images_to_process.region=region

n_images=n_elements(headers_list_of_images_to_process)

;*****************************************
;MAKES OUTPUT FILE HEADER (SEE CONFIG.PRO)
;*****************************************
file_header=config('structures','st1')
file_header.what=what
file_header.region=region
file_header.type=type
file_header.start_year=long(start_year)
file_header.start_day=long(start_day)
file_header.stop_year=long(stop_year)
file_header.stop_day=long(stop_day)
file_header.n_images=n_images
file_header.sampling=sampling

;*****************************************
;OPENS OUTPUT FILE
;*****************************************
openw,unit,output_file_path,/get_lun,error=err
IF (err ne 0) THEN BEGIN
make_dirs_to_file,output_file_path
openw,unit,output_file_path,/get_lun
ENDIF

;*****************************************
;WRITES HEADER IN IT
;*****************************************
writeu,unit,file_header

;*****************************************
;PREPARES MASK USED TO DIFFERENTIATE
;ICE DOMAINS (MAKES PART WHICH IS COMMON
;TO ALL IMAGES IN LIST)
;*****************************************
image_size=config('data','image_size',2-(region mod 2))
nx=image_size(0)
ny=image_size(1)

;******************************
;ADDS REGION MASK
;******************************
IF (region gt 2) THEN BEGIN
read_file,$
config('paths','region_contour_and_mask',region),$
'region_contour_and_mask',info,$
region_contour,region_mask
common_mask=replicate(4b,nx,ny)
common_mask(region_mask)=0b
ENDIF ELSE common_mask=bytarr(nx,ny)

;******************************
;ADDS LAND MASK
;******************************
IF (what ne 2) THEN BEGIN
land_mask_path=$
config('paths','land_mask',2-(region mod 2))
read_file,land_mask_path,'land_mask',count,land_mask
common_mask(land_mask)=common_mask(land_mask)+1b
ENDIF

;*****************************************
;MAKES STATS COMPUTATION LOOP
;*****************************************
print,'Beginning computation of stats file '+$
output_file_path
print,''
FOR i=0,n_images-1 DO BEGIN

stats=process_image(headers_list_of_images_to_process(i),$
what,sampling,common_mask)
writeu,unit,stats

;************************************
;PRINTS CURRENT PERCENTAGE OF WORK DONE
;************************************
percentage_work_done=(100l*(i+1))/n_images
print,strcompress(string((stats.header.year mod 100)*$
1000l+stats.header.day1),/remove_all)+' '+$
str(percentage_work_done)+'% images processed.'

;*****************************************
;END OF LOOP
;*****************************************
ENDFOR

;*****************************************
;CLOSES OUTPUT FILE
;*****************************************
free_lun,unit

;*****************************************
; WARNS WHEN FINISHED
;*****************************************
print,'Stats file '+output_file_path+' created.'
print,''
print,''

;************************************************
END
```

STATS_VIEWING_ROUTINES

```
;*************************************************
;*************************************************
;                  INTERFACE_MEAN_CYCLES.PRO
;*************************************************
;*************************************************
;
;This routine sets up graphical interface which
;enables the user to choose the mean cycles used
;to compute anomaly signals.
;
;Input: none
;
;Output: displays graphical interface
;
;Calls:          config.pro
;                read_regions_equivalents.pro
;                create_om_structure.pro
;                read_file.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*******************************************************
;*******************************************************
PRO interface_mean_cycles

;*******************************************
;*******************************************
;    INTERFACE WIDGETS DEFINITION
;*******************************************
;*******************************************

;*************************************
;INTERFACE VARIABLES DEFINITION AND
;INITIALIZATION
;*************************************
COMMON variables_mean_cycles,$
variables_mean_cycles
variables_mean_cycles=$
config('structures','st11')
variables_mean_cycles.region=1
variables_mean_cycles.type=1

;*********************************
;MAIN WINDOW
;*********************************
COMMON mw_mean_cycles,$
mw_mean_cycles
COMMON mw_view,mw_view
mw_mean_cycles=$
wwmainwindow(mw_view,layout_mean_cycles,$
title='Mean Cycles Selection',/vertical,position=[50,100])

;*********************************
;REGION OPTION MENU
;*********************************
read_regions_equivalents,n,numbers,names,short_names
region_om_structure=create_om_structure($
['region_mean_cycles_cb',names])
region_om=wwoptionmenu(layout_mean_cycles,'Region :',$
region_om_structure)

;*********************************
;TYPE OPTION MENU
;*********************************
type_om=wwoptionmenu(layout_mean_cycles,$
'Type :      ',{,callback:$
'type_mean_cycles_cb',button:'a',button:'b'})

;*********************************
;MEAN CYCLE SELECTED TEXT
;*********************************
COMMON path_tx,path_tx
path_lb=wwtext(layout_mean_cycles,/label,$
text=' mean cycle selected : ')
path_tx=wwtext(layout_mean_cycles,$
'null_cb',cols=50,/read)

;*********************************
;CHANGE/EXIT BUTTONS
;*********************************
go_mean_cycles_bb=$
wwbuttonbox(layout_mean_cycles,$
['Change',' Exit '],'go_mean_cycles_cb')

;*********************************
;INITIALIZES WIDGETS
;*********************************
status=wwsetvalue(region_om,1)

;*********************************
;DISPLAYS INTERFACE
;*********************************
status=wwsetvalue(mw_mean_cycles,/display)

;*********************************
END



;*******************************************
;*******************************************
;                  CALLBACK PROCEDURES
;*******************************************
;*******************************************

;***************************************
;MANAGES REGION OPTION MENU
;***************************************
PRO region_mean_cycles_cb,wid,index

COMMON path_tx,path_tx
COMMON variables_mean_cycles,$
variables_mean_cycles

;************************
;UPDATES REGION
;************************
variables_mean_cycles.region=config('convert',$
'region_long',wwgetvalue(wid))

;************************
;READS MEAN_CYCLE
;CURRENTLY SELECTED
;************************
read_file,config('paths','mean_cycles_selected'),$
'mean_cycles_selected',header,mean_cycles_selected
read_regions_equivalents,n,numbers,names,short_names
w=where(numbers eq variables_mean_cycles.region)

;*******************************
;UPDATES MEAN_CYCLE_SELECTED TEXT
;*******************************
IF mean_cycles_selected(0) eq 'empty' THEN GOTO,pb
COMMON topics_directory_path,topics_directory_path
path=topics_directory_path+$
mean_cycles_selected(2*w(0)+$
variables_mean_cycles.type-1)
status=wwsetvalue(path_tx,path)
status=wwsetvalue(path_tx,/update)

RETURN

pb: BEGIN
status=wwsetvalue(path_tx,'empty')
status=wwsetvalue(path_tx,/update)
END

END

;***************************************
;MANAGES TYPE OPTION MENU
;***************************************
PRO type_mean_cycles_cb,wid,index

COMMON path_tx,path_tx
COMMON variables_mean_cycles,$
variables_mean_cycles

;************************
;UPDATES TYPE
;************************
variables_mean_cycles.type=index

;************************
;READS MEAN_CYCLE
;CURRENTLY SELECTED
;************************
read_file,config('paths','mean_cycles_selected'),$
'mean_cycles_selected',header,mean_cycles_selected
read_regions_equivalents,n,numbers,names,short_names
w=where(numbers eq variables_mean_cycles.region)

;*******************************
;UPDATES MEAN_CYCLE_SELECTED TEXT
;*******************************
IF mean_cycles_selected(0) eq 'empty' THEN GOTO,pb
COMMON topics_directory_path,topics_directory_path
path=topics_directory_path+$
mean_cycles_selected(2*w(0)+$
variables_mean_cycles.type-1)
status=wwsetvalue(path_tx,path)
status=wwsetvalue(path_tx,/update)
```

```
;*****************************
RETURN

;*****************************
pb: BEGIN
status=wwsetvalue(path_tx,'empty')
status=wwsetvalue(path_tx,/update)
END

;*****************************
END

;*************************************
;MANAGES CHANGE/EXIT BUTTONS
;*************************************
PRO go_mean_cycles_cb,wid,index

COMMON mw_mean_cycles,mw_mean_cycles
COMMON variables_mean_cycles,$
variables_mean_cycles

;***************************
CASE index OF

;***************************
;CHANGE MEAN CYCLE_SELECTED
;***************************
1: fs_mean_cycles=wwfileselection($
mw_mean_cycles,title='MEAN CYCLE SELECTION',$
'ok_fs_mean_cycles_cb',$
'cancel_fs_mean_cycles_cb',$
dir=config('paths','mean_cycle_dir',$
variables_mean_cycles.region,$
variables_mean_cycles.type))

;***************************
;EXIT
;***************************
2: status=wwsetvalue(mw_mean_cycles,/close)

;***************************
ENDCASE

;***************************
END

;*****************************************
;FILE_SELECTION WIDGET CHANGE BUTTON (CHANGE CYCLE)
;*****************************************
PRO ok_fs_mean_cycles_cb,wid,shell

COMMON variables_mean_cycles,$
variables_mean_cycles
COMMON topics_directory_path,topics_directory_path
path=strmid(wwgetvalue(wid),strlen(topics_directory_path),$
strlen(wwgetvalue(wid))-strlen(topics_directory_path))
variables_mean_cycles.path=path
region=variables_mean_cycles.region
type=variables_mean_cycles.type
add_path=variables_mean_cycles.path

;*****************************
;CHANGES MEAN_CYCLE_SELECTED
;*****************************
path=config('paths','mean_cycles_selected')
read_file,path,'mean_cycles_selected',header,$
mean_cycles_selected
read_regions_equivalents,n,numbers,names,short_names

a=replicate('not_defined',2*n)
a(0:n_elements(mean_cycles_selected)-1)=$
mean_cycles_selected(*)
w=where(numbers eq region)
a(2*w(0)+type-1)=add_path

openw,unit,path,/get_lun
printf,unit,n_elements(a)
FOR i=0,n_elements(a)-1 DO printf,unit,a(i)
free_lun,unit

;***********************
;ALERT BOX
;***********************
COMMON mw_mean_cycles,mw_mean_cycles
al=wwalert(mw_mean_cycles,$
'Mean cycle selected changed',$
['OK'],title='INFO')

;***********************
;EXITS FILE SELECTION
;WIDGET AND INTERFACE
;***********************
status=wwsetvalue(shell,/close)
status=wwsetvalue(mw_mean_cycles,/close)


;***********************
END

;*****************************************
;FILE_SELECTION WIDGET CANCEL BUTTON
;*****************************************
PRO cancel_fs_mean_cycles_cb,wid,shell
status=wwsetvalue(shell,/close)
END


;*************************************************
;*************************************************
;                    INTERFACE_SELECT.PRO
;*************************************************
;*************************************************
;This routine sets up graphical interface which
;enables the user to select the kind of plot he
;wants to view
;
;Input: none
;
;Output: displays graphical interface
;
;Calls:      read_regions_equivalents.pro
;            create_om_structure.pro
;            config.pro
;            write_plot.pro
;            plot_stats.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*************************************************
;*************************************************
PRO interface_select

;*********************************************
;*********************************************
;    INTERFACE WIDGETS DEFINITION
;*********************************************
;*********************************************

;*************************************
;INTERFACE COMMON VARIABLES DEFINITION
;AND INITIALIZATION
;*************************************
COMMON variables_plot,variables_plot
COMMON variables_select,variables_select
variables_select=variables_plot

;*****************************
;MAIN WINDOW
;*****************************
COMMON mw_select,mw_select
COMMON mw_view,mw_view
mw_select=wwmainwindow(mw_view,$
layout_select,title='Selection',/vertical,$
position=[200,200])

;*****************************
;REGION OPTION MENU
;*****************************
read_regions_equivalents,n,numbers,names,short_names
region_om_structure=$
create_om_structure(['region_select_cb',names])
region_om=wwoptionmenu(layout_select,'Region : ',$
region_om_structure)

;*****************************
;DOMAIN OPTION MENU
;*****************************
domain_om=wwoptionmenu(layout_select,'Domain :    ',$
{,callback:'domain_select_cb',button:'Sea-Ice',$
button:'Land(-Ice)',button:'All Ice'})

;*****************************
;TYPE OPTION MENU
;*****************************
type_om=wwoptionmenu(layout_select,'Type :        ',$
{,callback:'type_select_cb',button:'a',button:'b'})

;*****************************
;S/A/M OPTION MENU
;*****************************
sam_om=wwoptionmenu(layout_select,$
'S/a/m :      ',{,callback:'sam_select_cb',$
button:'Signal',button:'Anomaly',$
button:'Mean Cycle'})
```

```
;*******************************
;STATS TYPE OPTION MENU
;*******************************
stats_type_om=wwoptionmenu(layout_select,$
'Stats :         ',$
{,callback:'stats_type_select_cb',$
button:'PDF',button:'Mean',$
button:'Std',button:'Median'})

;*******************************
;START YEAR OPTION MENU
;*******************************
start_year_om_structure=create_om_structure($
['start_year_select_cb',$
config('interfaces','years_array')])
start_year_om=wwoptionmenu(layout_select,$
'From :        ',start_year_om_structure)

;*******************************
;STOP YEAR OPTION MENU
;*******************************
stop_year_om_structure=create_om_structure($
['stop_year_select_cb',$
config('interfaces','years_array')])
stop_year_om=wwoptionmenu(layout_select,$
'To :             ',stop_year_om_structure)

;*******************************
;SAMPLING OPTION MENU
;*******************************
sampling_om_structure=create_om_structure($
['sampling_select_cb',$
config('interfaces','sampling_array')])
sampling_om=wwoptionmenu(layout_select,'Sampling : ',$
sampling_om_structure)

;*******************************
;GO/EXIT BUTTONS
;*******************************
go_bb=wwbuttonbox(layout_select,['    Go    ',$
'  Exit  '],'go_select_cb')

;*******************************
;WIDGETS INITIALIZATION
;*******************************
w=where(numbers eq variables_select.region)
status=wwsetvalue(region_om,w(0)+1)
status=wwsetvalue(type_om,$
variables_select.type)
status=wwsetvalue(domain_om,$
variables_select.domain)
status=wwsetvalue(stats_type_om,$
variables_select.stats_type-1)
status=wwsetvalue(start_year_om,$
variables_select.start_year-1991)
status=wwsetvalue(stop_year_om,$
variables_select.stop_year-1991)
status=wwsetvalue(sampling_om,$
variables_select.sampling/250)
sam=variables_select.sam
IF sam gt 3 THEN sam=sam-3
status=wwsetvalue(sam_om,sam)

;*******************************
;DISPLAYS INTERFACE
;*******************************
status=wwsetvalue(mw_select,/display)

;*******************************
END



;*********************************************
;*********************************************
;                CALLBACK PROCEDURES
;*********************************************
;*********************************************

;*************************************
;MANAGES REGION OPTION MENU
;*************************************
PRO region_select_cb,wid,index
COMMON variables_select,variables_select
variables_select.region=config('convert',$
'region_long',wwgetvalue(wid))
END

;*************************************
;MANAGES DOMAIN OPTION MENU
;*************************************
PRO domain_select_cb,wid,index
COMMON variables_select,variables_select
variables_select.domain=index
END

;*************************************
;MANAGES TYPE OPTION MENU
;*************************************
PRO type_select_cb,wid,index
COMMON variables_select,variables_select
variables_select.type=index
END

;*************************************
;MANAGES S/A/M OPTION MENU
;*************************************
PRO sam_select_cb,wid,index
COMMON variables_select,variables_select
variables_select.sam=index
END

;*************************************
;MANAGES STATS_TYPE OPTION MENU
;*************************************
PRO stats_type_select_cb,wid,index
COMMON variables_select,variables_select
variables_select.stats_type=index+1
END

;*************************************
;MANAGES START_YEAR OPTION MENU
;*************************************
PRO start_year_select_cb,wid,index
COMMON variables_select,variables_select
variables_select.start_year=fix(wwgetvalue(wid))
END

;*************************************
;MANAGES STOP_YEAR OPTION MENU
;*************************************
PRO stop_year_select_cb,wid,index
COMMON variables_select,variables_select
variables_select.stop_year=fix(wwgetvalue(wid))
END

;*************************************
;MANAGES SAMPLING OPTION MENU
;*************************************
PRO sampling_select_cb,wid,index
COMMON variables_select,variables_select
variables_select.sampling=fix(wwgetvalue(wid))
END

;*************************************
;MANAGES OK/EXIT BUTTONS
;*************************************
PRO go_select_cb,wid,index

COMMON mw_select,mw_select

;*******************************
CASE index of

;*********************
;                OK
;*********************
1: BEGIN $
COMMON variables_plot,variables_plot
COMMON variables_select,variables_select
variables_plot=variables_select
status=wwsetvalue(mw_select,/close)
write_plot
plot_stats
END

;*********************
;                EXIT
;*********************
2: status=wwsetvalue(mw_select,/close)

;*******************************
ENDCASE

;*******************************
END

;*************************************************
;*************************************************
;                INTERFACE_VIEW.PRO
;*************************************************
;*************************************************
;This routine sets up graphical interface which
;enables the user to view statistics plots and
;offers several options too (change color scale,
```

```
;smooth plot, etc...)
;
;Input: none
;
;Output: displays graphical interface
;
;Calls:          config.pro
;                topics_color_tables.pro
;                interface_mean_cycles.pro
;                plot_stats.pro
;                save_plot_to.pro (located end of this file)
;                write_to_tiff.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
PRO interface_view

;*******************************************
;*******************************************
;    INTERFACE WIDGETS DEFINITION
;*******************************************
;*******************************************

;*********************************
;INTERFACE VARIABLES DEFINITIION
;AND INITIALIZATION
;*********************************
COMMON variables_plot,variables_plot
variables_plot=config('structures','st6')
variables_plot.region=1
variables_plot.type=1
variables_plot.domain=1
variables_plot.stats_type=2
variables_plot.start_year=1992
variables_plot.stop_year=1999
variables_plot.sampling=250
variables_plot.sam=1

COMMON min_plot,min_plot
COMMON max_plot,max_plot
min_plot=-1.0
max_plot=1.0

;*********************************
;MAIN WINDOW
;*********************************
COMMON mw_view,mw_view
COMMON mw_menu,mw_menu
mw_view=wwmainwindow(mw_menu,layout_view,$
title='View Plots',position=[100,100],/vertical)

;*********************************
;MENUS DEFINITIONS
;*********************************
color_table_menu={,callback:'color_table_view_cb',$
button:'gray',button:'color'}

plot_menu={,callback:'plot_view_cb',button:'Select',$
button:'Save',button:'Save as ...',button:'Print'}

options_menu={,callback:'options_view_cb',$
button:'Choose mean cycle',$
menubutton:'Color Table',menu:color_table_menu,$
button:'Cross Section',$
button:'Smooth'}

quit_menu={,callback:'quit_view_cb',button:'Quit     '}

;*********************************
;MENU BAR
;*********************************
menus={,callback:'',menubutton:'  Plot  ',menu:plot_menu,$
menubutton:'  Options  ',menu:options_menu,$
menubutton:'  Quit  ',menu:quit_menu}

menu_bar=wwmenubar(layout_view,menus)

;*********************************
;PLOT AREA + LOADS COLOR TABLE
;*********************************
topics_color_tables,1,1,0.,1.
COMMON draw_area,draw_area
draw_area=wwdrawing(layout_view,1,$
'null_cb',[974,584],[970,580],background=0b)

;*********************************
;DISPLAYS INTERFACE
;*********************************
status=wwsetvalue(mw_view,/display)

;*********************************
END



;*******************************************
;*******************************************
;                CALLBACK PROCEDURES
;*******************************************
;*******************************************

;*****************************************
;MANAGES PLOT MENU
;*****************************************
PRO plot_view_cb,wid,index
COMMON variables_plot,variables_plot
CASE index OF

;************************
;VIEW PLOT
;************************
1: interface_select

;************************
;SAVE PLOT
;************************
2: BEGIN
path=config('paths','tiff_save',variables_plot)
save_plot_to,path
END

;************************
;SAVE PLOT AS
;************************
3: BEGIN
view_fs=wwfileselection(wid,'fs_ok_view_cb',$
'fs_cancel_view_cb',title='Destination File')
END

;************************
;PRINT PLOT
;************************
4: BEGIN
view_dg=wwdialog(wid,'Enter print command : ',$
title='PRINT','dg_ok_view_cb','null_cb',$
text='lpr -s -Pncp')
END

;************************
ENDCASE

;************************
END

;*****************************************
;MANAGES COLOR TABLE MENU
;*****************************************
PRO color_table_view_cb,wid,index
COMMON variables_plot,variables_plot
COMMON min_plot,min_plot
COMMON max_plot,max_plot
sam=variables_plot.sam
IF sam gt 3 THEN sam=sam-3
IF variables_plot.stats_type eq 2 THEN $
topics_color_tables,sam,index-1,min_plot,max_plot
END

;*****************************************
;MANAGES OPTIONS MENU
;*****************************************
PRO options_view_cb,wid,index
CASE index OF

;*****************************
;SEE/CHANGE MEAN_CYCLES_SELECTED
;*****************************
1: interface_mean_cycles

;*****************************
;COLOR SCALE (see above)
;*****************************
2:

;*****************************
;CROSS SECTION MODE
;*****************************
3: plot_stats,/cross_section

;*****************************
;SMOOTH PLOT
;*****************************
4: plot_stats,/smooth
```

```
;***************************
ENDCASE

;***************************
END

;****************************************
;MANAGES QUIT MENU
;****************************************
PRO quit_view_cb,wid,index
COMMON mw_view,mw_view
status=wwsetvalue(mw_view,/close)
loadct,10,/silent
END

;****************************************
;MANAGES SAVE_AS SUB-MENU
;FILE SELECTION WIDGET (OK)
;****************************************
PRO fs_ok_view_cb,wid,shell
path=wwgetvalue(wid)
status=wwsetvalue(shell,/close)
save_plot_to,path
END

;****************************************
;MANAGES SAVE_AS SUB-MENU
;FILE SELECTION WIDGET (CANCEL)
;****************************************
PRO fs_cancel_view_cb,wid,shell
status=wwsetvalue(shell,/close)
END

;****************************************
;MANAGES PRINT SUBMENU
;DIALOG BOX (OK)
;****************************************
PRO dg_ok_view_cb,wid,text
com=wwgetvalue(wid)
path=config('paths','tmp_dir')+'/tmp.tiff'
save_plot_to,path,/print
spawn,com+' '+path
COMMON mw_view,mw_view
dg_al=wwalert(mw_view,'Printing done',title='INFO',['OK'])
END

;****************************************
;SUBROUTINE THAT SAVES PLOT TO A TIFF FILE
;****************************************
PRO save_plot_to,path,print=keyword_print

;**********************
;ALERT BOX
;**********************
COMMON mw_view,mw_view
IF not(keyword_set(keyword_print)) THEN $
plot_save_al=wwalert(mw_view,['Saving plot to : ',$
path],/noconfirm,title='SAVING PLOT',nowait=1,working=1)

;**********************
;READS PLOT
;**********************
image=tvrd(0,0,!d.x_size,!d.y_size)

;****************************
;TRIES TO OPEN DESTINATION FILE
;IF PARENT DIRECTORIES MISSING,
;CREATES THEM
;****************************
openw,unit,path,/delete,/get_lun,error=err
IF (err ne 0) THEN BEGIN
make_dirs_to_file,path
openw,unit,path,/delete,/get_lun
ENDIF
free_lun,unit

;****************************
;CONVERTS PLOT TO TIFF IMAGE
;****************************
write_to_tiff,image,path

;*********************************
;CLOSES ALERT BOX,OPENS A NEW ONE
;*********************************
if not(keyword_set(keyword_print)) THEN BEGIN
wwalertpopdown,plot_save_al
plot_save_al=wwalert(mw_view,'Plot successfully saved',$
/noconfirm,title='INFO',['OK'])
ENDIF

;*********************************
END
```

```
;*************************************************
;*************************************************
;                  LABEL_STATS.PRO
;*************************************************
;*************************************************
;This routine loads statistics corresponding to
;parameters entered and adds a header containing
;information about image/contour availability
;as well as black zones ratio of original images
;
;Input: region : region number
;               type: 1=a 2=b
;               domain: 1=sea ice, 2=land ice, 3= all ice
;               stats_type: 0=ice_extent
;                           1=black_zones_ratio
;                           2=pdf
;                           3=mean
;                           4=std
;                           5=median
;               start_year: yyyy integer
;               stop_year: yyyy integer
;               sampling: integer or string
;               key_str: 'stats' or 'mean' whether signal or
;                        mean cycle is desired
;
;Output: structure, type st7, 4 tags:
;                image_available: 0 or 1
;                ice_contour_available: 0 or 1
;                black_zones_ratio: float between 0.0 and 1.0
;                stats: loaded statistics
;
;Calls:          config.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
FUNCTION label_stats,region,type,domain,$
stats_type,start_year,stop_year,sampling,key_str

;************************************
;MAKES USEFUL PARAMETERS
;************************************
n_years=fix(stop_year)-fix(start_year)+1
dim_stats=(stats_type eq 2)+1
time_range=122*(stop_year-start_year+1)

;************************************
CASE key_str OF

;************************************
;CASE SIGNAL STATS
;************************************
'stats': BEGIN

;**************************
;INITIALIZING VARIABLES
;**************************
image_available=[0b]
ice_contour_available=[0b]
black_zones_ratio=[0.0]

if (stats_type eq 2) then $
stats=replicate(0.0,1,sampling) $
else stats=[0.0]

;**************************
;BEGINS LOOP ON YEAR
;**************************
FOR i=0,n_years-1 DO BEGIN

;********************
;LOADS/ADDS STATS
;OF YEAR
;********************
year_str=str(fix(start_year)+i)
file_path=config('paths','stats',region,$
type,year_str,sampling)
read_file,file_path,'stats',file_header,file_data
if (stats_type eq 2) then stats=$
[stats,transpose(file_data.(domain).(stats_type))] $
else stats=[stats,file_data.(domain).(stats_type)]

;********************
;LOADS/ADDS LABELS
;OF YEAR
;********************
```

```
image_available=[image_available,$
file_data.header.image_available]
ice_contour_available=[ice_contour_available,$
file_data.header.ice_contour_available]
black_zones_ratio=[black_zones_ratio,$
file_data.(domain).black_zones_ratio]

ENDFOR

;****************************
;SELECTS USEFUL PARTS OF
;ARRAYS (SEE INITIALIZATION)
;****************************
image_available=image_available(1:*)
ice_contour_available=ice_contour_available(1:*)
black_zones_ratio=black_zones_ratio(1:*)
if (stats_type eq 2) then stats=stats(1:*,*) else $
stats=stats(1:*)

;****************************
END


;**********************************
;CASE MEAN CYCLE STATS
;**********************************
'.mean': BEGIN

;**********************
;LOADS MEAN CYCLE
;**********************
mean_cycle_path=config('paths','mean_selected',$
region,type,sampling)
read_file,mean_cycle_path,'stats',file_header,file_data
if (stats_type eq 2) then stats=$
transpose(file_data.(domain).(stats_type)) $
else stats=file_data.(domain).(stats_type)

;********************
;LOADS LABELS
;********************
image_available=file_data.header.image_available
ice_contour_available=$
file_data.header.ice_contour_available
black_zones_ratio=file_data.(domain).black_zones_ratio

;********************
;DUPLICATE LABELS
;AND MEAN CYCLE
;********************
image_available_plus=image_available
ice_contour_available_plus=ice_contour_available
black_zones_ratio_plus=black_zones_ratio
stats_plus=stats

FOR k=2,n_years DO BEGIN
image_available=[image_available,image_available_plus]
ice_contour_available=[ice_contour_available,$
ice_contour_available_plus]
black_zones_ratio=$
[black_zones_ratio,black_zones_ratio_plus]
stats=[stats,stats_plus]
ENDFOR

;*********************************
END

;***************************************
ENDCASE

;*************************************
;MAKES OUTPUT STRUCTURE
;*************************************
output=config('structures','st7',$
sampling,time_range,dim_stats)
output.header.image_available=image_available
output.header.ice_contour_available=ice_contour_available
output.header.black_zones_ratio=black_zones_ratio
output.stats=stats

;*************************************
RETURN, output

;*****************************************
END


;*****************************************************
;*****************************************************
;                    PLOT_STATS.PRO
;*****************************************************
;*****************************************************
;This routine reads the statistics written in
;dedicated file and plots them in the drawing area of
; interface_view. It may smooth or make a
;cross-section of the plot too if keywords are set
;
;Input: /smooth keyword to smooth
;               /cross_section keyword to plot cross section too
;
;Output: plot on interface view drawing area
;
;Calls:         read_file.pro
;               config.pro
;               topics_color-tables.pro
;               write_plot.pro
;               number_to_date.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
PRO plot_stats,smooth=keyword_smooth,$
cross_section=keyword_cross_section

;**********************************
;DEVICE=X WINDOWS
;**********************************
set_plot,'x'

;**********************************
;READS STATS TO PLOT
;**********************************
read_file,config('paths','plot'),$
'plot',info,stats_and_header
stats_to_plot=stats_and_header.stats

;**********************************
;MAKES USEFUL PLOT PARAMETERS
;w=width
;h=height
;window=whole drawing area
;plot=stats plot drawing area
;1=lower left corner
;2=upper right corner
;**********************************
COMMON variables_plot,variables_plot
start_year=variables_plot.start_year
stop_year=variables_plot.stop_year

n_years= stop_year- start_year+1

w_window=970
h_window=580

x1_plot=110
y1_plot=115

w_plot=w_window-(x1_plot+40)
h_plot=250

x2_plot=x1_plot+w_plot-1
y2_plot=y1_plot+h_plot-1

COMMON min_plot,min_plot
COMMON max_plot,max_plot

min_plot=min(stats_to_plot)
max_plot=max(stats_to_plot)

;**********************************
CASE variables_plot.stats_type OF

;*****************************************************
;*****************************************************
; CASE I : PDF IMAGES
;*****************************************************
;*****************************************************
2: BEGIN

;****************************
;LOADS RIGHT COLOR TABLE
;****************************
sam=variables_plot.sam
IF sam gt 3 THEN sam=sam-3
topics_color_tables,sam,1,min_plot,max_plot

;****************************
;WRITES DESCRIPTION LABELS
;ON PLOT
;****************************
erase

xyouts,/device,x1_plot,h_window-30,'Region : '+$
config('convert','region_long',$
```

```
variables_plot.region),charsize=1.2

xyouts,/device,x1_plot,h_window-50,'Domain : '+$
config('convert','domain',$

variables_plot.domain),charsize=1.2
xyouts,/device,x1_plot,h_window-70,'Type : '+$
config('convert','type',$
variables_plot.type),charsize=1.2

xyouts,/device,x1_plot,h_window-90,'S/a/m : '+$
config('convert','sam',$
variables_plot.sam),charsize=1.2

xyouts,/device,x1_plot,h_window-110,'Statistics : '+$
config('convert','stats_type',$
variables_plot.stats_type),charsize=1.2

xyouts,/device,x1_plot,h_window-130,'Start year : '+$
str( start_year),charsize=1.2

xyouts,/device,x1_plot,h_window-150,'Stop year : '+$
str( stop_year),charsize=1.2

xyouts,/device,x1_plot,h_window-170,'Sampling : '+$
str(variables_plot.sampling),charsize=1.2

!p.noerase=1

;*******************************
;SMOOTH STATS IF DESIRED
;AND RECONFIG PLOT PARAMETERS
;*******************************
IF keyword_set(keyword_smooth) THEN BEGIN
stats_to_plot=smooth(stats_to_plot,3)
stats_and_header.stats=stats_to_plot
variables_plot.sam=sam+3
write_plot,stats_and_header
print,'Smoothing done'
min_plot=min(stats_to_plot)
max_plot=max(stats_to_plot)
topics_color_tables,sam,1,min_plot,max_plot
ENDIF

;*******************************
;MAKE A COPY OF STATS,
;SCALES IT TO FIT LOADED
;COLOR TABLE AND STRETCH IT
;TO FIT PLOT AREA
;*******************************
scaled_stats=(stats_to_plot-min_plot)/$
(max_plot-min_plot)*(!d.n_colors-2-1)+1
scaled_stats=congrid(scaled_stats,w_plot,h_plot)

;*******************************
;DISPLAYS THIS SCALED COPY
;*******************************
tv,scaled_stats,x1_plot,y1_plot

;*******************
;LOADS Y AXIS RANGE
;*******************
y=config('data','pdf_range',2-$
(variables_plot.region mod 2),variables_plot.type)

;*******************************
;DRAWS AXIS
;*******************************
plot,[0],y,/device,$
yrange=y,xrange=[ start_year, stop_year+1],$
xticks=n_years,xticklen=-.04,$
xstyle=9,ystyle=1,yticklen=-.01,xtickformat='(i4)',$
xminor=12,yminor=2,yticks=5,ytickformat='(f5.2)',$
ytitle=config('convert','ytitle',variables_plot),$
title=config('convert','title',variables_plot),$
position=[x1_plot,y1_plot,x2_plot,y2_plot],charsize=1.2

;*******************************
;DRAWS DASHED LINE FOR EACH YEAR
;*******************************
FOR i= start_year+1, stop_year DO BEGIN
plots,[i,i],[y(0),y(1)],/data,linestyle=2
ENDFOR

;*******************************
;DRAWS LINE ON TOP OF PLOT
;NECESSARY BECAUSE XSTYLE=9
;*******************************
plots,[x1_plot,x2_plot],[y2_plot,y2_plot],/device


;*****************************************************
;                        MAKES COLOR BAR
;*****************************************************


;****************************
;COLOR BAR PLOT PARAMETERS
;****************************
w_cs=300
h_cs=8
x1_cs=370
x2_cs=x1_cs+w_cs-1
y1_cs=40
y2_cs=y1_cs+h_cs-1

;*******************************
;    PLOTS COLOR BAR
;*******************************
color_scale=byte(indgen(w_cs)/(w_cs-1.)*$
(!d.n_colors-2-1)+1)
FOR i=0,2 DO color_scale=[[color_scale],[color_scale]]
tv,color_scale,x1_cs,y1_cs

;*******************************
; MAKES BAR'S AXIS LABELING
;*******************************
a=alog10(max_plot-min_plot)
a=fix(a)-(((a-fix(a) ne 0.0) and a lt 0.0))

;*********************
;DETECTS COLOR LEVELS
;AND LABELS AXIS
;*********************
tvlct,r,g,b,/get
r=r(1:!d.n_colors-2)
g=g(1:!d.n_colors-2)
b=b(1:!d.n_colors-2)
diff_r=r-shift(r,1)
diff_g=g-shift(g,1)
diff_b=b-shift(b,1)
diff=bytarr(!d.n_colors-2)
diff(*)=(diff_r(*) ne 0) or $
(diff_g(*) ne 0) or (diff_b(*) ne 0)
diff(0)=1b
diff(n_elements(diff)-1)=1b
w=where(diff eq 1b)
x=(w/float(!d.n_colors-3)*$
(max_plot-min_plot)+min_plot)/10.0^a
w=nint(x1_cs+w/float(!d.n_colors-3)*(w_cs-1.))


FOR i=0,n_elements(x)-1 DO BEGIN
plots,[w(i),w(i)],[(y1_cs-5)+(h_cs+5)*(i mod 2),$
(y1_cs-1)+(h_cs+5)*(i mod 2)],/device
value=str(string(nint(100*x(i))/100.,format='(f5.2)'))
xyouts,/device,w(i)-18,(y1_cs-18)+(h_cs+25)*$
(i mod 2),value,charsize=0.9
ENDFOR

;*********************
;DRAWS BORDER BOX
;*********************
plots,[x1_cs,x2_cs],[y1_cs-1,y1_cs-1],/device
plots,[x2_cs,x2_cs],[y1_cs-1,y2_cs+1],/device
plots,[x2_cs,x1_cs],[y2_cs+1,y2_cs+1],/device
plots,[x1_cs,x1_cs],[y2_cs+1,y1_cs-1],/device

;*******************************
;PLOTS UNIT OF PDF
;*******************************
units=config('convert','units',variables_plot)
IF variables_plot.type eq 2 THEN units='('+units+')'
IF a eq 0 THEN $
xyouts,/device,(x1_cs+w_cs+30),y1_cs,$
'('+units+'!E-1'+'!N)',charsize=0.8 $
ELSE $
xyouts,/device,(x1_cs+w_cs+30),y1_cs,$
'('+'10!E'+str(a)+'!N '+$
units+'!E-1'+'!N)',charsize=1.1

;*****************************************
;CROSS SECTION MODE (IF DESIRED)
;*****************************************
IF keyword_set(keyword_cross_section) THEN BEGIN

;*******************************
;CROSS-SECTION PLOT PARAMETERS
;w=width
;h=heigth
;1=ll corner
;2=ur corner
;sw=small window (cross-section)
;*******************************
w_sw=160
```

```
h_sw=160
x1_sw=w_window-200
x2_sw=x1_sw+w_sw-1
y2_sw=h_window-10
y1_sw=h_window-10-h_sw+1

;*************************
;CROSS-SECTION PLOT X POINTS
;*************************
y_range=config('data','pdf_range',$
2-(variables_plot.region mod 2),$
variables_plot.type)
x_cross=indgen(variables_plot.sampling)/$
float(variables_plot.sampling-1)*$
(y_range(1)-y_range(0))+y_range(0)

;*************************
;CROSS-SECTION PLOT Y POINTS
;*************************
cross_section=fltarr(variables_plot.sampling)

;*************************
;CURSOR INITIALIZATION
;*************************
xold=-1
yold=-1

;*************************
;WARNS CROSS_SECTION
;MODE IS ON
;*************************
print,'Cross-Section Mode off'
print,'Press right mouse button '+$
'to escape Cross-Section Mode'

;*****************************
;BEGINS LOOP
;*****************************
REPEAT BEGIN

cursor,X,Y,0,/device

;*****************************
;CHECKS IF CURSOR ON STATS PLOT
;*****************************
IF ((X ge x1_plot) and (X le (x2_plot)) and $
(Y ge y1_plot) and (Y le (y2_plot)) and $
((X ne xold) or (Y ne yold))) THEN BEGIN
p=fix(((X-x1_plot)/(w_plot-1.))*(122*n_years-1))
a=stats_to_plot(p,*)
cross_section(*)=a(*)

;*****************************
;ERASE PREVIOUS CROSS-SECTION
;*****************************
polyfill,[x1_sw-335,x2_sw+20,x2_sw+20,x1_sw-335],$
[y1_sw-25,y1_sw-25,y2_sw+5,y2_sw+5],$
color=0b,/device

;*******************************
;PLOTS NEW ONE IF NOT ON A GAP
;*******************************
w=where(cross_section ne 0.0,count)
IF count ne 0 THEN plot,x_cross,cross_section,$
position=[x1_sw,y1_sw,x2_sw,y2_sw],$
xrange=y_range,xstyle=1,ystyle=1,/device

;*******************************
;MAKES CROSS-SECTION DATE
;*******************************
ddd=str(3*(p mod 122))
CASE strlen(ddd) OF
1: ddd='00'+ddd
2: ddd='0'+ddd
3:
ENDCASE
date=number_to_date(str(start_year+p/122)+ddd)

;*******************************
;PLOTS DATE LABEL
;*******************************
s='PDF '+config('convert','sam',variables_plot.sam)+$
' on '+date
xyouts,x1_sw-330,h_window-90,s,/device,charsize=1.2
xyouts,x1_sw-330,h_window-110,$
'(MB3 to escape Cross-Section Mode)',$
/device,charsize=1.2

;*******************************
ENDIF

;*******************************
;UPDATE OLD CURSOR POSITION
;*******************************
xold=X
yold=Y

;*******************************
;EXITS LOOP IF PRESS MOUSE
;BUTTON 3
;*******************************
ENDREP until (!mouse.button eq 4)

;*******************************
;ERASE CROSS-SECTION PLOTS
;*******************************
polyfill,[x1_sw-335,x2_sw+20,x2_sw+20,x1_sw-335],$
[y1_sw-25,y1_sw-25,y2_sw+5,y2_sw+5],$
color=0b,/device

;*******************************
;CROSS-SECTION MODE OFF
;*******************************
print,'Cross-Section Mode off'

;*******************************
ENDIF

;*******************************
!p.noerase=0

;*******************************
END

;*************************************************
;*************************************************
; CASE II : GRAPHS (mean,standard deviation,median)
;*************************************************
;*************************************************
ELSE: BEGIN

;*************************************
;LOADS GRAPH-ADAPTED COLOR TABLE
;*************************************
loadct,16,/silent

;*******************************
;WRITES DESCRIPTION LABELS
;ON PLOT
;*******************************
erase

xyouts,/device,x1_plot,h_window-30,'Region : '+$
config('convert','region_long',$
variables_plot.region),charsize=1.2

xyouts,/device,x1_plot,h_window-50,'Domain : '+$
config('convert','domain',$
variables_plot.domain),charsize=1.2

xyouts,/device,x1_plot,h_window-70,'Type : '+$
config('convert','type',$
variables_plot.type),charsize=1.2

xyouts,/device,x1_plot,h_window-90,'S/a/m : '+$
config('convert','sam',$
variables_plot.sam),charsize=1.2

xyouts,/device,x1_plot,h_window-110,'Statistics : '+$
config('convert','stats_type',$
variables_plot.stats_type),charsize=1.2

xyouts,/device,x1_plot,h_window-130,'Start year : '+$
str( start_year),charsize=1.2

xyouts,/device,x1_plot,h_window-150,'Stop year : '+$
str( stop_year),charsize=1.2

xyouts,/device,x1_plot,h_window-170,'Sampling : '+$
str(variables_plot.sampling),charsize=1.2

!p.noerase=1

;*******************************
;MAKE A COPY OF STATS STRETCH
;TO FIT IN DRAWING AREA
;*******************************
scaled_stats=fltarr(w_plot)
a=[[stats_to_plot],[stats_to_plot]]
b=congrid(a,w_plot,2)
scaled_stats(*)=b(*,0)

;*******************************
;PLOTS AXIS ONLY (TO SET UP
;DATA COORDINATES FOR NEXT
;COMMAND : POLYFILL)
```

```
;*****************************
x=indgen(w_plot)/(w_plot-1.)*n_years+ start_year
plot,x,scaled_stats,/device,/nodata,$
xrange=[start_year, stop_year+1],$
yrange=[min_plot,max_plot],$
xstyle=9,ystyle=1,yticklen=-.01,xticklen=-0.04,$
yminor=2,yticks=5,xticks=n_years,$
ytitle=config('convert','ytitle',variables_plot),$
title=config('convert','title',variables_plot),$
position=[x1_plot,y1_plot,x2_plot,y2_plot],charsize=1.2

;*****************************
;DRAWS MOD 2 COLOR-FILLED
;BACKGROUND
;*****************************
FOR year= start_year, stop_year DO BEGIN
polyfill,/data,[year,year+1,year+1,year],$
[min_plot,min_plot,max_plot,max_plot],$
color=12+(year mod 2 )*18
ENDFOR

;*****************************
;PLOTS MODIFIED STATS
;*****************************
plot,x,scaled_stats,/device,$
xrange=[start_year,stop_year+1],xminor=12,$
yrange=[min_plot,max_plot],$
xstyle=9,ystyle=1,yticklen=-.01,xticklen=-0.04,$
yminor=2,yticks=5,xticks=n_years,$
ytitle=config('convert','ytitle',variables_plot),$
title=config('convert','title',variables_plot),$
position=[x1_plot,y1_plot,$
(x2_plot),(y2_plot)],charsize=1.2

;*****************************
;DRAWS LINE ON TOP OF PLOT
;NECESSARY BECAUSE XSTYLE=9
;*****************************
plots,[x1_plot,x2_plot],[y2_plot,(y2_plot)],/device

;*****************************
;DRAWS LINE ON TOP OF PLOT
;NECESSARY BECAUSE XSTYLE=9
;*****************************
IF variables_plot.sam eq 2 THEN $
plots,[start_year,stop_year+1],[0.0,0.0],/data

;*****************************
!p.noerase=0

;*****************************
END

;*********************************************
ENDCASE

;*************************************************
END

;*************************************************
;*************************************************
;                    TOPICS_COLOR_TABLES.PRO
;*************************************************
;*************************************************
;This routine creates gray or color level color tables
;for TOPICS plots enhancement:
;
;7-level gray color table for anomaly plots :
;linear contrast,non-linear intervals
;
;7-level blue/red color table for anomaly plots :
;non-linear contrast,non-linear intervals
;
;10-level gray color table for signal/mean cycle plots:
;custom contrats,custom intervals
;
;8-level contrasted color table for signal/mean cycle plots
;custom contrats,custom intervals
;
;Input: sam: byte 1=signal 2=mean cycle 3=anomaly
;       in_color: boolean 0=gray levels 1=color levels
;       mini,maxi: floats : values range of graph/plot
;
;Output: new color table loaded
;
;Calls:      funct.pro (see at end of file)
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*************************************************
;*************************************************
PRO topics_color_tables,sam,in_color,mini,maxi

CASE sam OF


;*************************************************
;*************************************************
;                    ANOMALY PLOTS
;*************************************************
;*************************************************
2: BEGIN

;*********************************
;MAKES USEFUL PARAMETERS
;*********************************
n_levels=7
range=float(maxi-mini)
zero_extent=2*(0.15*(abs(maxi)<abs(mini)))<(2*0.05*abs(maxi))
ratio=(maxi-0.5*zero_extent)/(range-zero_extent)
n_colors_plus=nint(ratio*(n_levels-1))
n_colors_minus=(n_levels-1)-n_colors_plus
RGB=bytarr(3,!d.n_colors)
n=!d.n_colors-2

;*********************************
;MAKES NEGATIVE VALUES LEVELS
;*********************************
start=1b
FOR i=0,n_colors_minus-1 DO BEGIN

;***************************
;NON LINEAR INTERVALS
;***************************
delta=byte((-0.5*zero_extent-mini)*$
((funct(i+1)-funct(i))/funct(n_colors_minus))/range*n)
stop=start+delta-1

IF in_color THEN BEGIN
;***************************
;COLOR: NON LINEAR LEVELS
;***************************
value=60+160*(funct(i)/funct(n_colors_minus-1))
FOR j=start,stop DO RGB(*,j)=[0b,value,255b]
ENDIF ELSE BEGIN
;***************************
;GRAY: LINEAR LEVELS
;***************************
start_color_minus=100
step_color_minus=40
value=start_color_minus+step_color_minus*i
FOR j=start,stop DO RGB(*,j)=[value,value,value]
ENDELSE

start=stop+1

ENDFOR

;*********************************
;MAKES SURE THERE IS NO GAP
;BETWEEN RIGHT END AND BEGINNING OF
;ZERO LEVEL (DUE TO APPROXIMATIONS)
;*********************************
stop=byte((-0.5*zero_extent-mini)/range*n)
IF start le stop THEN FOR j=start,stop DO $
RGB(*,j)=RGB(*,start-1)

;*********************************
;MAKES ZERO LEVEL
;*********************************
start=stop+1
stop=byte((0.5*zero_extent-mini)/range*n)
FOR j=start,stop DO RGB(*,j)=[255b,255b,255b]

;*********************************
;MAKES POSITIVE VALUES LEVELS
;*********************************
start=stop+1
FOR i=0,n_colors_plus-1 DO BEGIN

;***************************
;NON LINEAR INTERVALS
;***************************
k=n_colors_plus-1-i
delta=byte((maxi-0.5*zero_extent)*$
((funct(k+1)-funct(k))/funct(n_colors_plus))/range*n)
stop=start+delta-1

IF in_color THEN BEGIN
;***************************
;COLOR: NON LINEAR LEVELS
;***************************
value=230-180*(1.0-funct(k)/funct(n_colors_plus-1))
FOR j=start,stop DO RGB(*,j)=[255b,value,0b]
ENDIF ELSE BEGIN
```

```
;*************************
;GRAY: LINEAR LEVELS
;*************************
start_color_plus=240
step_color_plus=-40
value=start_color_plus+step_color_plus*i
FOR j=start,stop DO RGB(*,j)=[value,value,value]
ENDELSE

start=stop+1

ENDFOR

;*********************************
;MAKES SURE THERE IS NO GAP
;BETWEEN END OF ZERO LEVEL AND
;LEFT END OF POSITIVE LEVELS
;*********************************
stop=n
IF start le stop THEN FOR j=start,stop DO $
RGB(*,j)=RGB(*,start-1)

;*********************************
;FIRST VALUE OF COLOR SCALE
;SET TO WHITE, LAST SET TO BLACK
;*********************************
RGB(*,0)=255b
RGB(*,n+1)=0b

;*********************************
;LOADS COLOR TABLE
;*********************************
tvlct,RGB(0,*),RGB(1,*),RGB(2,*)

;****************************************
END


;*********************************************
;*********************************************
;              SIGNAL/MEAN CYCLE PLOTS
;*********************************************
;*********************************************
ELSE: BEGIN

CASE in_color OF

;*********************************************
;                  GRAY LEVEL SIGNAL
;*********************************************
0: BEGIN

RGB=bytarr(3,!d.n_colors-2)

;*********************************
;LEVELS RANGES CUSTOM DEFINITION
;*********************************
tab=[6,10,20,25,30,42,65]
tab=[0,tab,100]

;*********************************
;COLORS CUSTOM DEFINITION
;*********************************
WHITE=[255b,255b,255b]
GRAY1=replicate(235b,3)
GRAY2=replicate(213b,3)
GRAY3=replicate(193b,3)
GRAY4=replicate(160b,3)
GRAY5=replicate(135b,3)
BLACK=[0b,0b,0b]

;*********************************
;PUTS COLORS IN ORDERS
;*********************************
color_tab=[[WHITE],[BLACK],[GRAY1],[GRAY5],$
[GRAY3],[GRAY4],[GRAY2],[BLACK]]

;*********************************
;MAKES COLOR SCALE
;*********************************
FOR i=0,n_elements(tab)-2 DO BEGIN
start=byte((tab(i))/100.*(!d.n_colors-2))
stop=byte((tab(i+1))/100.*(!d.n_colors-2))-1
FOR j=start,stop DO RGB(0:2,j)=color_tab(0:2,i)
ENDFOR

RGB=[[WHITE],[RGB],[BLACK]]

;*********************************
;LOADS COLOR TABLE
;*********************************
tvlct,RGB(0,*),RGB(1,*),RGB(2,*)

;****************************************
END


;*********************************************
;                  COLOR LEVEL SIGNAL
;*********************************************
1: BEGIN

RGB=bytarr(3,!d.n_colors-2)

;*********************************
;LEVELS RANGES CUSTOM DEFINITION
;*********************************
tab=[6,10,20,25,30,42,65]
tab=[0,tab,100]

;*********************************
;COLORS CUSTOM DEFINITION
;*********************************
WHITE=[255b,255b,255b]
RED=[255b,0b,0b]
ORANGE=[255b,122b,0b]
YELLOW=[255b,220b,0b]
GREEN=[0b,190b,0b]
BLUE=[0b,0b,255b]
CYAN=[50b,255b,255b]
PINK=[255b,170b,170b]
BLACK=[0b,0b,0b]
MAGENTA=[255b,130b,255b]

;*********************************
;PUTS COLORS IN ORDERS
;*********************************
color_tab=[[WHITE],[BLACK],[YELLOW],$
[BLUE],[MAGENTA],[RED],[GREEN],[CYAN]]

;*********************************
;MAKES COLOR SCALE
;*********************************
FOR i=0,n_elements(tab)-2 DO BEGIN
start=byte((tab(i))/100.*(!d.n_colors-2))
stop=byte((tab(i+1))/100.*(!d.n_colors-2))-1
FOR j=start,stop DO RGB(0:2,j)=color_tab(0:2,i)
ENDFOR

RGB=[[WHITE],[RGB],[BLACK]]

;*********************************
;LOADS COLOR TABLE
;*********************************
tvlct,RGB(0,*),RGB(1,*),RGB(2,*)

;*********************************
END

;*************************************
ENDCASE

;*************************************
END

;*********************************************
ENDCASE

;*************************************************
END



;*****************************************
;FUNCTION THAT IS USED TO MAKE NON LINEAR
;LEVEL/INTERVAL COLOR SCALES
;FACTOR EXAGERATES DIFFERENCE TO LINEAR CASE
;FACTOR~0 : LINEAR
;THE BIGGER FACTOR, THE LESS LINEAR
;*****************************************
FUNCTION funct,x
factor=3.25
y=alog(x*factor+1.)
return,y
END


;*************************************************
;*************************************************
;                  WRITE_PLOT.PRO
;*************************************************
;*************************************************
;This routine writes the labeled statistics specified
;by the common variable variables_plot or those given
;in input to a file specified by config.pro
;
```

```
;Input: none or labeled stats to write
;
;Output: file containing labeled statistics, path
;                 given by config.pro
;
;Calls: label_stats.pro
;               config.pro
;               rename_structure.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
PRO write_plot,data_in

;***********************************
;LOADS/MAKES USEFUL PARAMETERS
;***********************************
COMMON variables_plot,variables_plot

sampling=variables_plot.sampling
stop_year=variables_plot.stop_year
start_year=variables_plot.start_year
stats_type=variables_plot.stats_type
region=variables_plot.region
type=variables_plot.type
domain=variables_plot.domain
sam=variables_plot.sam

dim_stats=(stats_type eq 2)+1
time_range=122*(stop_year-start_year+1)

;***********************************
;CASE NO INPUT
;***********************************
IF (n_params() eq 0) THEN BEGIN

;*****************************
;MAKES LABELED SIGNAL
;*****************************
a=label_stats(region,type,domain,stats_type,start_year,$
stop_year,sampling,'.stats')

header=a.header
stats=a.stats

;*****************************
;MAKES LABELED MEAN CYCLE
;*****************************
b=label_stats(region,type,domain,stats_type,$
start_year,stop_year,sampling,'.mean')
mean_cycle=b.stats

CASE sam OF
;*****************************
;IF SIGNAL WANTED, FILLS GAPS
;IN SIGNAL USING MEAN CYCLE
;*****************************
1: BEGIN
w=where((header.ice_contour_available eq 0b) or $
(header.image_available eq 0b),count)
IF (count ne 0) THEN stats(w,*)=mean_cycle(w,*)
END
;*****************************
;IF ANOMALY WANTED, DOES SAME
;AS ABOVE AND REMOVE MEAN CYCLE
;*****************************
2:BEGIN
w=where((header.ice_contour_available eq 0b) or $
(header.image_available eq 0b),count)
IF (count ne 0) THEN stats(w,*)=mean_cycle(w,*)
stats=stats-mean_cycle
END
;*****************************
;IF MEAN CYCLE WANTED, JUST
;LOADS IT
;*****************************
3: stats=mean_cycle
ENDCASE

;*****************************
;MAKES STRUCTURE TO WRITE
;*****************************
labeled_stats_to_write=config('structures',$
'st7',sampling,time_range,dim_stats)
labeled_stats_to_write.header=$
rename_structure(header,labeled_stats_to_write.header)
labeled_stats_to_write.stats=stats

;***********************************
;CASE THERE IS AN INPUT :
;STRUCTURE TO WRITE=INPUT
;***********************************
ENDIF ELSE labeled_stats_to_write=data_in


;***********************************
;WRITES STRUCTURE TO FILE
;SPECIFIED BY CONFIG.PRO
;***********************************
path=config('paths','plot')
openw,unit,path,/get_lun
writeu,unit,sampling,time_range,dim_stats
writeu,unit,labeled_stats_to_write
free_lun,unit

;*********************************************
END


;*************************************************
;*************************************************
;                 WRITE_TO_TIFF.PRO
;*************************************************
;*************************************************
;This routine writes an array  as a tiff image
;
;Input: image : 2D byte array
;               path : path from root of destination file
;
;Output: tiff file containing specified image
;
;Calls:         none
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
PRO write_to_tiff,image,path

;*******************************************
;READS PAL COLOR TABLE FOR USE IN RGB VERSION
;*******************************************
tvlct,r,g,b,/get
pal=transpose([[r],[g],[b]])

;*******************************************
;MAKES NEW IMAGE WITH RIGHT FORMAT
;*******************************************
image=reverse(image,2)
siz=size(image)
newimage=bytarr(siz(1),siz(2),3)
FOR k=0,2 DO newimage(*,*,k)=pal(k,image(*,*))

;*******************************************
;WRITES NEW IMAGE TO FILE SPECIFIED BY PATH
;*******************************************
status=dc_write_tiff(path,newimage,class='RGB')

;*********************************************
END
```

TOPICS_STARTUP_ROUTINES

```
;*****************************************************
;*****************************************************
;                        COMPILE.PRO
;*****************************************************
;*****************************************************
;This routine compiles all topics routines.
;this is a batch file : calling sequence : @compile
;at the prompt
;
;Input: none
;
;Output: none
;
;Calls:          none
;
;Written by/in: Pierre Mercier 07/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************

;****************************************
;COMPILES COMMON_USE_ROUTINES
;****************************************
.r config
.r create_om_structure
.r date_to_number
.r is_leap
.r ll_to_sirf
.r make_circle_sirf
.r make_dirs_to_file
.r number_to_date
.r read_file
.r read_regions_equivalents
.r rename_structure
.r sirf_to_ll
.r str
.r yy_to_yyyy

;****************************************
;COMPILES CONTOURS_MAKING_ROUTINES
;****************************************
.r compute_contour
.r interface_contours
.r read_ice_concentration

;****************************************
;COMPILES CUSTOM_MENU_ROUTINES
;****************************************
.r interface_custom1
.r interface_custom2
.r interface_see_regions
.r invert_ct
.r plot_contour
.r sketch
.r update_x_y_region

;****************************************
;COMPILES STATS_MAKING_ROUTINES
;****************************************
.r compute_mean_cycles
.r compute_stats
.r interface_stats
.r make_census
.r make_extent_grid
.r make_mask
.r make_weights
.r process_image
.r process_list

;****************************************
;COMPILES STATS_VIEWING_ROUTINES
;****************************************
.r interface_mean_cycles
.r interface_select
.r interface_view
.r label_stats
.r plot_stats
.r topics_color_tables
.r write_plot
.r write_to_tiff

;****************************************
;COMPILES TOPICS_STARTUP_ROUTINES
;****************************************
.r interface_cover
.r interface_menu
.r set_path
.r topics
```

```
;*****************************************************
;*****************************************************
;                INTERFACE_COVER.PRO
;*****************************************************
;*****************************************************
;This routine sets up TOPICS welcome graphical
;interface
;
;Input: none
;
;Output: displays graphical interface
;
;Calls:          config.pro
;                read_file.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
PRO interface_cover

;*********************************************
;*********************************************
;    INTERFACE WIDGETS DEFINITION
;*********************************************
;*********************************************

;****************************
; TOP-LEVEL TOPICS WIDGET
;****************************
common top_cover,top_cover
top_cover=wwinit('TOPICS','Examples',$
layout_cover,/vertical,position=[300,20])

;****************************
; COVER IMAGE AND TITLE
;****************************
cover_image_dw=wwdrawing(layout_cover,1,$
'cover_image_cover_cb',[366,469],[362,465])

;****************************
; GO/INFO/EXIT BUTTONS
;****************************
go_cover_bb=wwbuttonbox(layout_cover,$
[' Go ',' Info ',' Exit '],$
'go_cover_cb')

;*************************
;  DISPLAYS INTERFACE
;*************************
status=wwsetvalue(top_cover,/display)

;*************************
;MANAGES INTERFACE EVENTS
;*************************
wwloop

;*************************
END


;*****************************************************
;*****************************************************
;                              CALLBACK PROCEDURES
;*****************************************************
;*****************************************************


;*************************************
;READS AND DISPLAYS COVER IMAGE
;*************************************
PRO cover_image_cover_cb,wid,index
print,' '
print,'IGNORE ERROR MESSAGES BELOW'
print,' '
print,'-------------------------------------------'
cover_image_path=config('paths','cover_image')
cover_image=image_read(cover_image_path)
image_display,cover_image,/quiet,/wset
print,'-------------------------------------------'
print,' '
END

;*************************************
;MANAGES GO/INFO/EXIT BUTTONS
;*************************************
PRO go_cover_cb,wid,index
```

```
common top_cover,top_cover
common go_cover,go_cover
go_cover=0b

CASE index of

;*******************************
;     GO : ENTERS TOPICS
;*******************************
1: BEGIN

;************************
;RESET MEAN CYCLES SELECTED
;TO DEFAULT REMARK :
;SHORTER VERSION WHICH TAKES
;TO MUCH SPACE TO RUN ON A
;64MB MACHINE :
;spawn,'cp '+config('paths',$
;'mean_cycles_selected_default')+' '+$
;config('paths','mean_cycles_selected_default')
;************************
openr,unit,config('paths',$
'mean_cycles_selected_default'),/get_lun
count=0l
path=''
readf,unit,count
a=strarr(count)
FOR i=0,count-1 DO BEGIN
readf,unit,path
a(i)=path
ENDFOR
free_lun,unit

openw,unit,config('paths','mean_cycles_selected')
printf,unit,count
FOR i=0,count-1 DO printf,unit,a(i)
free_lun,unit

;*******************************
;CLOSES COVER INTERFACE, RETURN
;TO TOPICS MAIN PROGRAM
;WHICH WILL RUN INTERFACE_MENU
;BECAUSE GO_COVER=1b
;*******************************
go_cover=1b
status=wwsetvalue(top_cover,/close)

;*******************************
END

;*******************************
;                                INFO :
;     READS INFO TEXT FILE
;       AND DISPLAYS IT
;*******************************
2: BEGIN
read_file,config('paths','info_cover'),$
'info_cover',header,info_cover
cover_al=wwalert(top_cover,info_cover,$
['OK'],title='INFORMATION ABOUT TOPICS')
END

;*******************************
;                                EXIT
;*******************************
3: status=wwsetvalue(top_cover,/close)

;*******************************
ENDCASE

;*******************************
END



;*****************************************************
;*****************************************************
;                INTERFACE_MENU.PRO
;*****************************************************
;*****************************************************
;This routine sets up TOPICS' main menu
;graphical interface
;
;Input: none
;
;Output: displays graphical interface
;
;Calls:          none
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
PRO interface_menu

;*********************************************
;*********************************************
;    INTERFACE WIDGETS DEFINITION
;*********************************************
;*********************************************

;************************
;MAIN WINDOW
;************************
COMMON mw_menu,mw_menu
mw_menu=wwinit('TOPICS','',layout_menu,$
/vertical,position=[5,5])

;************************
;MENUS DEFINITIONS
;************************
bank_menu1={,callback:'bank_menu1_cb',$
button:'Make Census'}
bank_menu2={,callback:'bank_menu2_cb',$
button:'Compute Contours'}
bank_menu3={,callback:'bank_menu3_cb',$
button:'Compute Statistics'}
banks_menu={,callback:'null_cb',$
menubutton:'Images Bank',menu:bank_menu1,$
menubutton:'Contours Bank',menu:bank_menu2,$
menubutton:'Statistics Bank',menu:bank_menu3}

custom_menu={,callback:'custom_menu_cb',$
button:'See Current Regions',$
button:'Define New Region'}

view_menu={,callback:'view_menu_cb',button:'View plots'}

quit_menu={,callback:'quit_menu_cb',button:'Quit '}

;************************
;MENU BAR
;************************
menus={,callback:'',$
menubutton:'Manage Banks',menu:banks_menu,$
menubutton:'Customize',menu:custom_menu,$
menubutton:' View   ',menu:view_menu,$
menubutton:'Quit',menu:quit_menu}

menu_bar=wwmenubar(layout_menu,menus)

;************************
;DISPLAYS INTERFACE
;************************
status=wwsetvalue(mw_menu,/display)

;************************
;TOPICS INTERFACES
;EVENTS MANAGER
;************************
wwloop

;************************
END

;*********************************************
;*********************************************
;                CALLBACK ROUTINES
;*********************************************
;*********************************************

;***********************************
;MANAGES IMAGES BANK MENU
;***********************************
PRO bank_menu1_cb,wid,index
CASE index OF
1: make_census
else:
ENDCASE
END

;***********************************
;MANAGES CONTOURS BANK MENU
;***********************************
PRO bank_menu2_cb,wid,index
CASE index OF
1: interface_contours
else:
ENDCASE
END

;***********************************
;MANAGES STATS BANK MENU
;***********************************
```

```
PRO bank_menu3_cb,wid,index
CASE index OF
1: interface_stats
else:
ENDCASE
END

;************************************
;MANAGES CUSTOM MENU
;************************************
PRO custom_menu_cb,wid,index
CASE index OF
1: interface_see_regions
2: interface_custom1
ENDCASE
END

;************************************
;MANAGES VIEW MENU
;************************************
PRO view_menu_cb,wid,index
CASE index OF
1: interface_view
else:
ENDCASE
END

;************************************
;MANAGES QUIT MENU
;************************************
PRO quit_menu_cb,wid,index
print,'Bye'
print,''
COMMON mw_menu,mw_menu
status=wwsetvalue(mw_menu,/close)
END

;*****************************************************
;*****************************************************
;                    SET_PATH.PRO
;*****************************************************
;*****************************************************
;This routine adds to wave variable !path the paths
;of the directories containing routines that needs
;to be compiled to use TOPICS
;
;Input: ask the user TOPICS directory path
;                at the prompt line
;
;Output: none
;
;Calls:
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
PRO set_path

;****************************************
;ASKS FOR TOPICS DIRECTORY PATH
;AT THE PROMPT LINE AND MAKES IT A COMMON
;VARIABLES FOR FUTURE USE BY CONFIG.PRO
;****************************************
COMMON topics_directory_path,topics_directory_path
topics_directory_path=''
read,'Enter TOPICS directory path from root : ',$
topics_directory_path

;****************************************
;ADDS TO !PATH THE PATHS OF THE DIRS CONTAINING
;TOPICS ROUTINES
;****************************************
!path=topics_directory_path+'/CODE/COMMON_USE_ROUTINES:'+$
topics_directory_path+'/CODE/CONTOURS_MAKING_ROUTINES:'+$
topics_directory_path+'/CODE/CUSTOM_MENU_ROUTINES:'+$
topics_directory_path+'/CODE/STATS_MAKING_ROUTINES:'+$
topics_directory_path+'/CODE/STATS_VIEWING_ROUTINES:'+$
topics_directory_path+'/CODE/TOPICS_STARTUP_ROUTINES:'+$
!path

;****************************************
END

;*****************************************************
;*****************************************************
;                 SET_UP_TOPICS.PRO
;*****************************************************
;*****************************************************
;This routine adds to wave variable !path the paths
;of directories containing TOPICS routines to
;be compiled. This is a batch file -> calling sequence
;is @set_up_topics at the prompt line
;
;Input: none
;
;Output: none
;
;Calls:          set_path.pro
;                compile.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************

;****************************************
;INSTALS HDF
;****************************************
@hdf_startup

;****************************************
;UPDATE !PATH (SEE SET_PATH.PRO)
;****************************************
set_path

;****************************************
;COMPILE TOPICS ROUTINES (SEE COMPILE.PRO)
;****************************************
@compile

;****************************************
;WARNS WHEN FINISHED
;****************************************
print,''
print,'Topics paths added to !PATH, Topics routines compiled.'
print,''
print,'Type "topics" at the prompt to begin.'
print,''

;*****************************************************
;*****************************************************
;                      TOPICS.PRO
;*****************************************************
;*****************************************************
;This routine is TOPICS main program, though it is
;one of TOPICS' shortest routines. It just calls
;interface_cover.pro to set_up the welcome interface
;and depending on the user's choice, enters or exits
;TOPICS
;
;The reason for making two different programs is that
;I wanted to make a clear distinction between
;the different kind of routines - in that case
;the main program and interface routines. If you
;prefer to rename interface_cover 'TOPICS' and include
;the following lines in it, feel free to do so.
;
;Input: none
;
;Output: TOPICS welcome graphical interface
;
;Calls:          interface_cover.pro
;                interface_menu.pro
;
;Written by/in: Pierre Mercier 08/00
;
;Last updated: 08/31/00
;*****************************************************
;*****************************************************
PRO topics

;****************************************
;MAKES SUTE THE PLOT DEVICE IS X WINDOWS,
;RESERVES WINDOWA 0,1,2 AND CLOSES
;EVERYTHING BEFORE RUNNING TOPICS ROUTINES
;****************************************
set_plot,'x'
wdelete,0,1,2
close,/all

;****************************************
;CALLS INTERFACE_COVER.PRO TO SET UP
;WELCOME GRAPHICAL INTERFACE
;****************************************
COMMON go_cover,go_cover
go_cover=0b
interface_cover

;****************************************
;CALLS INTERFACE_MENU.PRO TO SET UP
;MAIN MENU GRAPHICAL INTERFACE IF
;USER ASKED TO ENTER TOPICS, OTHERWISE
;EXITS TOPICS
```

```
;*****************************************
if (go_cover eq 1) then interface_menu
;*****************************************
END
```

Acknowldegments

I would like to thank Dr Mark RDRINWATER and Dr Ben HOLT for their welcoming me in the Ocean Science department at the Jet Propulsion Laboratory and their supervising my work during five months there. I also woul like to acknowledge Dr David LONG from Brigham Young University for providing me with up-to-date SIRF processed EScat data, Xiang LIU, Dr Denis BOURRAS and Dr Andy BINGHAM for their advice, and Peter WOYCESHYN for his help with TOPICS color tables. Last but not least, thanks a lot to Anita LACROIX and Cherry AKBARRI for doing a great job with the administrative stuff, and to I-Lin Tang for dealing with all the computer-related problems.

Enhanced EScat data were kindly provided by David Long at Brigham Young University. SSM/I ice concentration data were dowloaded from the National Snow and Ice Data Center (NSIDC) website www.nsidc.com.

References

BENDAT & PIERSOL, Random data - analysis and measurement procedures , 1986, ISBN 0-471-04000-2

DRINKWATER, Satellite microwave radar observations of Antarctic sea ice, Chapter 8, Analysis of SAR data of the polar oceans, Edited by Tsatsoulis & Kwok, 1998

DRINKWATER, Microwave remote sensing of Weddel Sea ice, Antarctic Reasearch Series, Vol. 74, page 187, 1998

DRINWATER, C-Band backscatter measurements of winter sea-ice in the Weddel Sea, Antarctica, Int. J. Remote Sensing, 1995, vol. 16, no. 17, 3365-3389

DRINKWATER, Satellite microwave radar observations of climate related sea-ice anomalies, pdf article on http://polar.jpl.nasa.gov

DRINKWATER & BINGHAM, Recent changes in the microwave scattering properties of the Antarctic ice sheet, Geoscience and Remote Sensing, vol. 38, July 2000, ISSN 0196-2892

DRINKWATER, LIU, LOW, WADHAMS, Interannual variability in Weddel Sea ice from ERS wind scatterometer, pdf article on http://polar.jpl.nasa.gov

DRINKWATER & LONG, Enhanced resolution scatterometer imaging of Southern Ocean sea ice, ESA J. 17 :307-322, 1993

HOLT, The effect of a storm on the 1992 summer sea ice cover of the Beaufort, Chukchi and East Siberian seas, April 2000

LEDREW, BARBER, AGNEW & DUNLOP, Candian sea ice atlas from microwave remotely sensed imagery : July 1987 to June 1990, Climatological Studies Number 44, U.D.C 551.311.18

ROTHROCK, YU & MAYKUT, Thinning of the Arctic sea ice cover, Geophysical Research Letters, vol.26, no. 23, pages 3469-3472, Dec. 1999

STEFFEN, Atlas of the sea ice types, Zürcher Geographische Schriften, 1986, ISBN 3 7281 1622 X