Ice Discrimination using ERS scatterometer

Siebren de Haan and Ad Stoffelen

18 September 2001

Contents

1 Introduction

This paper describes a new method to characterize ice in ERS scatterometer measurements. The objective is to find an improved discrimination of water and ice.

The European Remote Sensing (ERS-1 and -2) satellite were launched in 1991 and 1995. The ERS satellites circles the Earth at a height of 800 km and completes an orbit every 100 minutes crossing both poles in the process. In this so-called polar orbit, the Earth is gradually rotating beneath the spacecraft and so on each subsequent orbit a slightly different part of the ground is seen. Using this type of orbit the ERS-2 scatterometer can observe the surface of the entire globe almost completely in just three days. The only part which is not seen has a latitude higher than approximately 89^o (Artic) and lower than -80^o (Antartic). The region with higher latitude is expected to be covered with either land ice or snow (Antaric) or sea ice (Artic).

Up to February 2001 scatterometer data from the ERS-2 was available for operational use. The ERS scatterometer is a microwave radar operating in C-band, 5.3 Ghz, sequentially illuminating the sea surface, independent of clouds, measuring the normalised radar cross section, called σ_0. The power of the reflected signal is dependent on the sea surface roughness which over the ice-free oceans is created by the wind stress.

Three antennas look in three different directions with radar look angles 45^o, 90^o and 135^o (fore-, mid-, and aft-beam) relative to the flight direction, covering a 500 km wide swath parallel to the sub-satellite track. The ground swath is divided in regular grid nodes which have a 25 km size representing overlapping footprints of size 50 km. A scatterometer node is subsequently illuminated by each antenna resulting in a triplet of measured σ_0. Across track the swath is divided in 19 cells with incidence angles, θ, ranging from 25^o to 57^o for the fore and aft beam and 18^o to 46^o for the mid beam. Over water σ_0 is related to wind speed, relative wind direction and incidence angle through an empirically derived model function. Using the three measurements obtained from three different directions surface wind speed and wind direction can be derived. Over sea ice backscattering turns out to be relatively isotropic in contrast with the strong anisotropic behaviour, which is the basis for wind direction retrieval over water.

Cavenié, et al (1994) suggests a parameter which uses the fact that backscatter over sea ice is relatively isotropic, that is, independent of radar look angle. He defines an anisotropy coefficient

$$A = \left| \frac{\sigma_0^{fore} - \sigma_0^{aft}}{\sigma_0^{fore} + \sigma_0^{aft}} \right|, \tag{1}$$

where σ_0^{fore} and σ_0^{aft} are measured with the same incidence angle but with 90 degrees difference in radar look angle. The anisotropy parameter is solely based on σ_0^{fore} and σ_0^{aft} and not on σ_0^{mid}. For each node across the swath a treshold T_A is defined. Values of A smaller than this treshold indicate ice. Discrimination using this technique is most pronounced for high incidence angles. Problems arise over open water when the wind blows parallel to the sub-satellite ground track, because in these cases the backscatter measurements of fore and aft beam are very close.

Another parameter which can be used for ice discrimination is based on the change of backscatter with incidence angle, that is, the derivative of backscatter with respect to incidence angle (Gohin, et al (1995)). By approximation this derivative can be determined as

$$D = -\frac{\sigma_0^{fore} - \sigma_0^{mid}}{\theta^{fore} - \theta^{mid}}, \tag{2}$$

where σ_0^{fore} and σ_0^{mid} are backscatter measurements observed at the same node with given incidence angles θ^{fore} and θ^{mid}. In analogy to the anisotropy coefficient a node dependent threshold T_D is defined. Using the threshold, this method has the most pronounced sea/ice discrimination for low incidence angles which is opposite in behaviour to the anisotropy coefficient. Again, problems occur when the wind is blowing along or across the track. The change in backscatter with incidence angle can also be used for ice classification into multi-year (low D) and first year (high values of D). The derivative D can also be determined by values of the σ_0^{mid} and σ_0^{aft}.

Stoffelen (1998) suggested an additional parameter to discriminate ice from water. As mentioned above, the two-dimensional ocean wind can be related to a σ_0 triplet through a geophysical model function. This mapping of the two-dimensional wind defines a two-dimensional cone surface in the 3D σ_0 triplet space (Stoffelen (1998)). If this model function was perfect and the observations free of noise, then over water the measured σ_0 triplet would exactly lie on a point on this cone corresponding to the wind. The model function and measurements are not perfect, but generally, over the ocean, the observations are quite near to this cone. Over ice however, this model function relating backscatter to surface wind does not apply, and the distance of the observed σ_0 triplet to this cone may be large, indicating low wind probability. Hence it is suggested to use a threshold of this parameter, so that when the distance to the cone is larger than the threshold, ice is assumed. This method as tested by IFREMER works best for the low and high incidence angles (IFREMER (1996)).

All three methods described above are not optimal with respect to sea/ice discrimination for backscatter measurents observed at the intermediate incidence angle range.

At DNMI an integrated multi-sensor Sea Ice product is developed within the OSI-SAF (O&SI SAF (1998)) framework. This product combines ice products from available sensors (AVHRR, SSM/I, scatterometer, AMSU) to obtain a single reliable product from the various information involving different and possibly contradicting information on ice parameters. A good general tool for combining various data sources containing uncertain information is given by the Bayesian approach. This approach requires knowlegde of the uncertainties in the ice classification of the various instruments. That is, for each measurement it is necessary to specify the probability distribution of each instrument ice estimate given ice, given water and given other ice parameters. The resulting product will be an ice likelihood estimate.

Stoffelen (1998) exploited the concept of 3D measurement space to characterise the wind sensitivity over sea. Similarly, here we define an ice model in the three dimensional measurement space of backscatter triplets. Using this ice model, ice parameter retrieval can be performed in addition to wind parameter retrieval. The purpose of this is to improve wind/ice discrimination. This ice model differs from the previously mentioned anisotropy A parameter and backscatter derivative D in the sense that it fully exploits the information content of the three observed σ_0 at each node rather than reduce two of the three σ_0 to one paramter. The three dimensional ice model may be used in the multi-sensor sea ice product of DNMI since it comprehends both anisotropy parameter and backscatter derivative (Eqs. 1 and 2).

The three dimensional ice model is constructed and explained in the next section. Different cross sections are used for this construction such that direct comparison can be made with all three backscatter ice parameters (i.e. anisotropy A, derivative of the backscatter D and distance to the wind cone). A generalization of the ice model is given in a seperate section. Results of the ice discrimation using this ice model is shown in section 3. From the triplet of backscatter measurements at each node information is derived on the ice or wind properties. Different ice types can be inferred using the ice model. Moreover, when past scatterometer data are used as well, then the combination of several azimuth and incidence angles in a certain area reveals sufficient information for the exclusion of ice points, or the acceptance of ocean points in the wind product. The last section contains the conclusions.

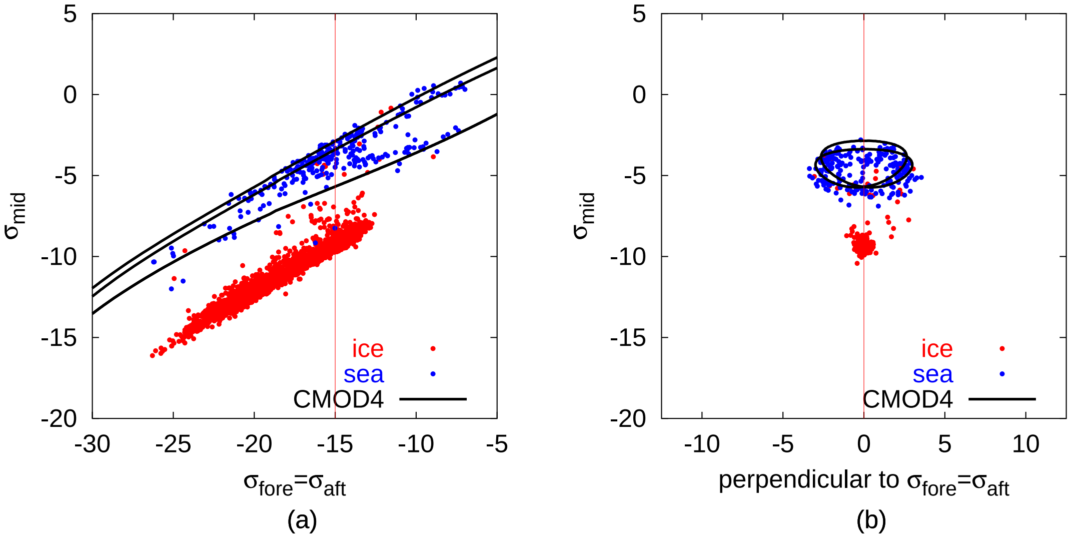


Figure 1: Cross-sections of backscatter triplets measured at node 4, with (a) the $(\sigma_0^{fore}=\sigma_0^{aft})$-plane and (b) perpendicular to $(\sigma_0^{fore}=\sigma_0^{aft})$-plane. The straight line in each sub-figure represents the plane in the other sub-figure.

2 Ice Model

In the first part of this section we describe the derivation of a three dimensional ice model. This model is based on the specific ERS configuration of three beams. A generalization of this model is given in the second part of this section avoiding a σ_0 dependency on θ and ice type. This generalized ice model can be used for ice detection with ASCAT or for comparison with SeaWinds.

2.1 ERS Ice Model

Cross-sections in the 3D measurement space reveal a distinction between ice and sea points. Figure 1(a) and (b) show for the fourth node cross-sections of the backscatter triplets along the $(\sigma_0^{fore}=\sigma_0^{aft})$-plane and a plane perpendicular to this plane. The line of intersection of both planes is shown by the straight line in both figures. The wind cone (CMOD4, Stoffelen (1998)) is also plotted in these graphs. Backscatter triplets on this cone have a high probability to originate from wind driven roughness at sea. The distinction between ice and sea backscatter triplets is made using the IFREMER ice mask. The weekly Arctic ice masks of January 2000 are used to generate a monthly ice mask. Only locations reporting ice during the whole month are labeled as ice, which results in a conservative ice mask. Clearly visible from these cross-sections is the separation between ice and sea backscatter triplets. Note that, despite the conservative ice mask, some ice-classified backscatter triplets reside close to the wind cone indicating high wind probability. Furthermore, due to the conservative monthly ice mask it may happen that backscatter triplets which are over ice are on locations which are not ice covered during the

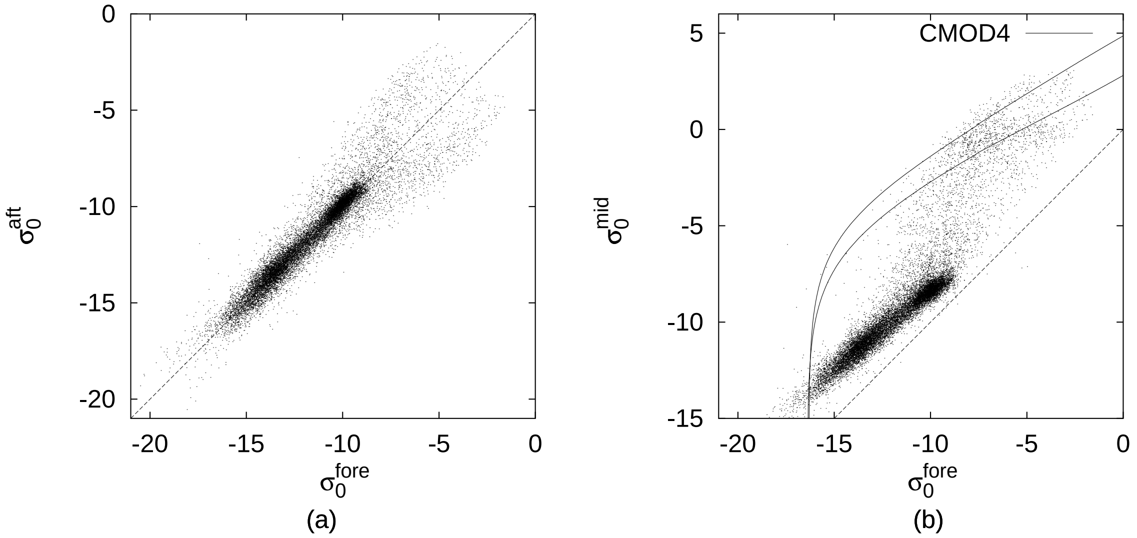


Figure 2: Projection of ice backscatter triplets of node 1 on (a) σ_0^{fore},σ_0^{aft}-plane and (b) σ_0^{fore},σ_0^{mid}-plane. The dashed line in sub-figure (a) and (b) is σ_0^{fore}=σ_0^{aft} and σ_0^{fore}=σ_0^{mid} line respectively. Also shown in sub-figure (b) are the boundaries of the wind cone corresponding to node 1.

whole month of January and therefore are labelled as wind in the ice mask.

From the figure we see that most ice triplets lie on a line; this indicates that one geophysical parameter determines the isotropic scattering from ice. Figure 2(a) shows the projection of ice backscatter triplets of the first node on the (σ_0^{fore},σ_0^{aft})-plane. Figure 2(b) shows the projection on (σ_0^{fore},σ_0^{mid})-plane and the boundaries of the CMOD4 wind cone. As said before, the distinction between ice and sea is based on a very conservative ice mask originating from weekly ice masks produced by IFREMER. Nonetheless, Figure 2(b) shows that some backscatter measurements are close to the wind cone although they were labelled as ice. As in Figure 2(b), in Figure 2(a) these wind points are located mainly in the range above -8 dB. Some more are located between the cone and the great majority of ice points. Perhaps this is due to fractional ice cover. Again the majority is located closely to a line. An indication of the values of the anistropy parameter can be obtained by comparing the shown triplets with the dashed line in Figure 2(a). The large majority of shown ice points have a small anisotropy value, however some ice points with σ_0 values larger than -8 dB have a small anisotropy parameter but are in fact wind points.

The anisotropy parameter A is zero on the $\sigma_0^{fore} = \sigma_0^{aft}$ line shown in 2(a). However, many of these points are very close the the wind cone and are therfore most likely caused by wind backscatter rather than ice backscatter. Using solely A will result in a number of dubious or wrong ice indentifications. Since $\sigma_0^{fore} = \sigma_0^{mid} + D \cdot (\theta^{fore} - \theta^{mid})$ the line of $\sigma_0^{fore} = \sigma_0^{mid}$ in Figure 2(b) provides an indication of the behaviour of the D parameter to discriminate ice. It is clear that the distribution of parallel lines for the range of D including all ice points, also includes many wind points. This is a serious drawback for using D to identify ice.

To find a good representation for the ice line corresponding to this node, fitting the backscatter ice triplets by a least squares linear fit does not result in a optimal fit. This is due to the large number of outliers, probably wind triplets, and the large distance of these outliers to the majority of ice backscatter triplets. The values σ_0^{fore} and σ_0^{aft} of an ice backscatter triplet are distributed nicely around the line σ_0^{fore}=σ_0^{aft} as can be seen from Figure 2(a). The ice line lies therefore most likely in the plane (σ_0^{fore}=σ_0^{aft}). The ice backscatter triplets of other nodes show the same distribution in the σ_0^{fore},σ_0^{aft}-plane although the number of outliers is less compared to the first node. Moreover, the outliers are less extreme.

In the (σ_0^{fore}=σ_0^{aft})-plane using an offset vector $\mathcal{O}$ and a direction vector e_a, the ice line is represented by

$$\mathcal{O} + a\, e_a = \begin{pmatrix} \bar{\sigma}_{fore,aft} \\ \bar{\sigma}_{fore,aft} \\ \bar{\sigma}_{mid} \end{pmatrix} + a \begin{pmatrix} \frac{1}{2}\sqrt{2}\cos\alpha \\ \frac{1}{2}\sqrt{2}\cos\alpha \\ \sin\alpha \end{pmatrix}, \tag{3}$$

where $\bar{\sigma}_{fore,aft}$ and $\bar{\sigma}_{mid}$ are the mean backscatter from ice triplets of the fore and aft beam and mid beam respectively. The parameter a is called the distance along the ice line. The angle α is the angle between the ice line and the σ_0^{mid} axis. Figure 3(a) shows the ice line together with the wind cone. This figure changes slightly for different nodes as will be shown later in this section. There are nodes for which the ice line intersects the wind cone and other nodes have an ice line which lies in the interior of the wind cone.

The ice line can be used to introduce a new coordinate system determined by the following three base vectors e_a, e_b, e_c and origin $\mathcal{O}$, see Figure 3(b)

$$e_a = \begin{pmatrix} \frac{1}{2}\sqrt{2}\cos\alpha \\ \frac{1}{2}\sqrt{2}\cos\alpha \\ \sin\alpha \end{pmatrix}, \; e_b = \begin{pmatrix} \frac{1}{2}\sqrt{2} \\ -\frac{1}{2}\sqrt{2} \\ 0 \end{pmatrix}, \; e_c = \begin{pmatrix} -\frac{1}{2}\sqrt{2}\sin\alpha \\ -\frac{1}{2}\sqrt{2}\sin\alpha \\ \cos\alpha \end{pmatrix} \text{ and}$$

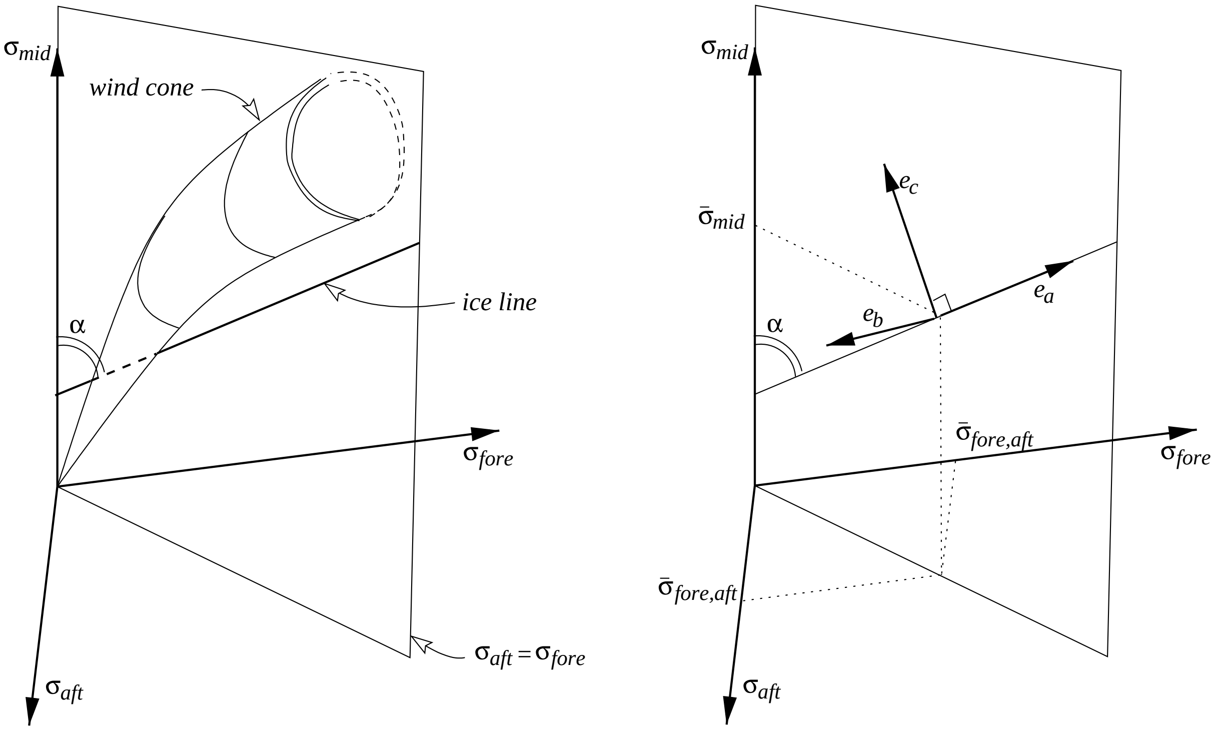
$$\mathcal{O} = \begin{pmatrix} \bar{\sigma}_{fore,aft} \\ \bar{\sigma}_{fore,aft} \\ \bar{\sigma}_{mid} \end{pmatrix}. \tag{4}$$

Then each σ_0 triplet can be written as

$$\sigma_0 = \mathcal{O} + a e_a + b e_b + c e_c. \tag{5}$$

For each node an ice line is determined, that is the position of the origin $\mathcal{O}$ and ice line angle α are determined for each node. The origin $\mathcal{O}$ is determined by the mean of the fore and aft ice backscatter ($\bar{\sigma}_0^{fore,aft}$) and the mean of mid ice backscatter ($\bar{\sigma}_0^{mid}$). As a first guess the ice line angle is set to a constant value. The means of the fore-and-aft and mid beam ice backscatter are fitted using one function, $\sigma_0^{mean}(\theta)$, which is chosen to be cubic in θ.

This fit, together with a constant value for the ice line angle α is chosen as a first guess ice model. For each ice backscatter triplet from Januari 2000 an ice triplet (a, b, c) is determined using this first guess ice model. The new mean backscatter triplets for each node are determined by ommitting the 5% smallest and largest values of c. For each node the optimal α corresponds to the one resulting in the sharpest distribution of c. The sharpness of a distribution can be determined be calculating the maximum steepness of the cumulative frequency distribution. The resulting values of α are fitted by a cubic function in θ. This procedure is repeated until for each



(a)

(b)

Figure 3: Three dimensional measurement space: (a) ice line and wind cone, (b) local coordinate system determined by the ice line.

node the change in values of mean backscatter and α is small. In Figure 4 the resulting mean values are plotted against the mean incidence angle θ together with the fitted mean backscatter $\sigma_0^{mean}(\theta)$ which is

$$\begin{aligned} \sigma_0^{mean}(\theta) &= -2.15921 - 6.53336 \cdot 10^{-1}\,\theta \\ &\quad + 1.14151 \cdot 10^{-2}\,\theta^2 - 8.55793 \cdot 10^{-5}\,\theta^3. \end{aligned} \tag{6}$$

Note that in the median incidence angle range small discrepancies exist between the fore, aft and mid σ_0. These are due to imperfections in the ice mask, that is, some nodes are reported as ice but are most likely wind nodes, due to some small remaining interbeam biases (Stoffelen (1998), Ocean Calibration), and related to differences in geophysical sampling. We believe that similar effects occur for the mid beam at low incidence angles such that the line is not necessarily a perfect fit. The largest deviations are found in the mid beam, which results in bias in c. A correction for this is discussed later in this section.

The ice line angle is fitted using a cubic function in θ^{fore},

$$\begin{aligned} \alpha(\theta^{fore}) &= 6.45240 \cdot 10^{-1} + 2.97880 \cdot 10^{-2}\,\theta^{fore} \\ &\quad - 7.43522 \cdot 10^{-4}\,(\theta^{fore})^2 + 5.79397 \cdot 10^{-6}\,(\theta^{fore})^3. \end{aligned} \tag{7}$$

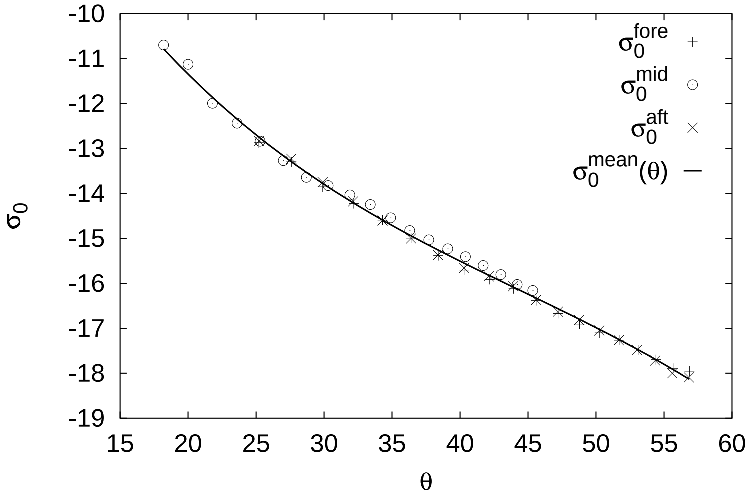


Figure 4: Mean ice backscatter of the fore and aft beam and the mid beam plotted against corresponding the mean incidence angle. The line is the cubic fit of the mean ice backscatter.

The parameter a is a meaningful geophysical parameter, because it determines the dynamic range of σ_0 over ice. Assuming similar scattering conditions at all nodes, the value of a can be made node (i.e incidence angle) independent. As such for a meaningful and unique geophysical interpretation we map a onto some reference node. Here node 10 is chosen as a reference. The mapping function is chosen to be linear in a, with a quadratic dependence on θ^{fore}, that is

$$\tilde{a}(a, \theta^{fore}) = A(\theta^{fore}) + B(\theta^{fore})\, a \tag{8}$$

For each node the mean of a is determined for all ice backscatter observations in January 2000, at positions on the conservative ice mask with a reduced resolution of 50×50 km. For each node the distribution of mean values collocated with the reference node 10 is mapped onto node 10, using equation 8. Next these coefficients, which are node dependent, are fitted using the mean of θ^{fore}. The resulting functions A and B are

$$\begin{aligned} A(\theta^{fore}) &= 6.70077 - 4.20983 \cdot 10^{-1}\, \theta^{fore} \\ &\quad +8.87397 \cdot 10^{-3}\, (\theta^{fore})^2 - 6.21071 \cdot 10^{-5}\, (\theta^{fore})^3 \end{aligned} \tag{9}$$

$$\begin{aligned} B(\theta^{fore}) &= 2.60441 - 8.09739 \cdot 10^{-2}\, \theta^{fore} \\ &\quad +1.36175 \cdot 10^{-3}\, (\theta^{fore})^2 - 8.09724 \cdot 10^{-6}\, (\theta^{fore})^3 \end{aligned} \tag{10}$$

We verified that the distributions of $\tilde{a}$ for collocated nodes indeed overlap, such that the above linear fitting is adequate and $\tilde{a}$ describes the σ_0 dynamic range at all nodes.

Independent of a, the values c for each node are shifted such that the distribution of c for each node attains a maximum around zero, to correct for the observed deviation for the median incidence angles; this is not shown here. Next these node dependent shifts are fitted by a cubic function in θ^{fore}, resulting in

$$\tilde{c}(c, \theta^{fore}) = c - C(\theta^{fore}), \tag{11}$$

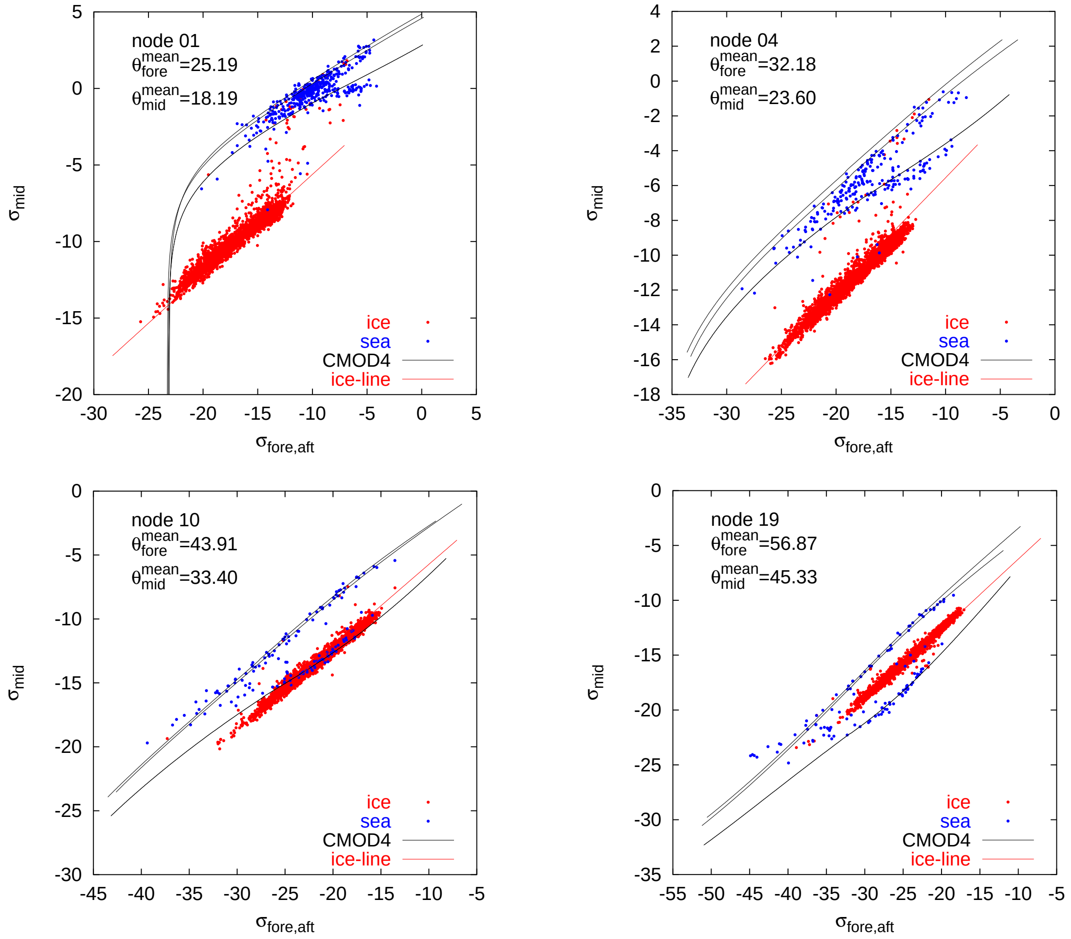


Figure 5: Cross-section of backscatter triplets with along the (σ_0^{fore}=σ_0^{aft})-plane for the nodes 1, 4, 10 and 19 and their corresponding wind cone. Ice line plotted in these figures is determined by $\sigma_0^{mean}(\theta)$.

where

$$\begin{aligned} C(\theta^{fore}) \quad = \quad & 3.76516 - 3.25490 \cdot 10^{-2}\, \theta^{fore} \\ & +8.63233 \cdot 10^{-3}\, (\theta^{fore})^2 - 7.20385 \cdot 10^{-5}\, (\theta^{fore})^3, \end{aligned} \tag{12}$$

where c is the original value and $\tilde{c}$ is the shifted value. For simplicity we now ommit the ˜ in the transformations for a and c.

The distance to the ice line is now defined as

$$d_{ice} = \sqrt{b^2 + c^2}. \tag{13}$$

Figure 5 shows for nodes 1, 4, 10 and 19 the ice line, the cross-section of backscatter triplets with (σ_0^{fore}=σ_0^{aft})-plane and the cross-section of the wind cone in the same plane. Data plotted is again data from January 2000 in the Arctic region. A clear distinction between ice and sea backscatter can be made using the ice line and wind cone for nodes 1 and 4. The ice line

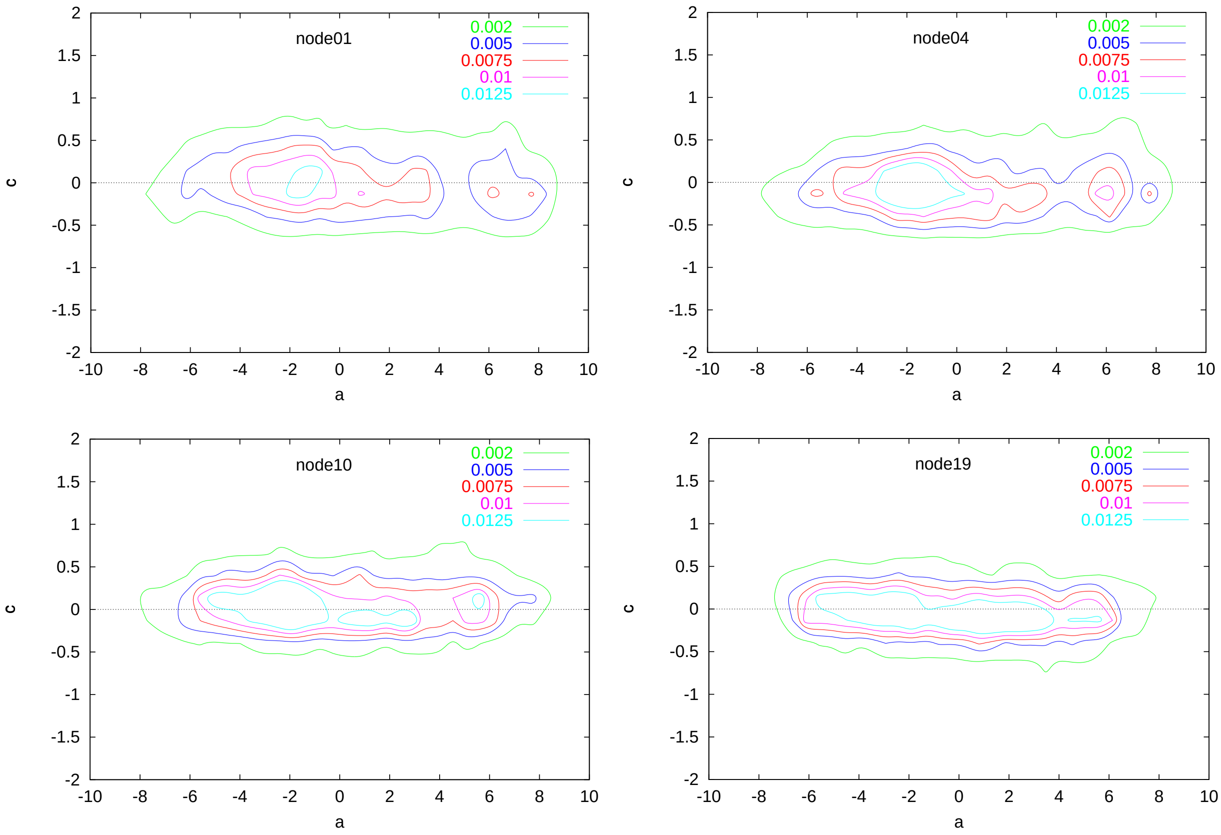


Figure 6: Distributions of (a, c)-pairs for ice triplets of nodes 1,4,10 and 19 in the period Januari 2000 in the Artic region.

overlaps the wind cone for node 10 which implies that an immediate distinction between ice and sea is more difficult when the distance to the ice line is small; see the next section for a solution to this problem. For node 19 the ice line lies in the interior of the wind cone, and therefore backscatter measurements with a small distance to either wind cone or ice line give a good indication whether the sea is frozen or open.

In Figure 6 the two dimensional distributions of a and c are shown for the same four nodes as in Figure 5. Only ice nodes are plotted here. The majority of (a, c)-pairs lies for all nodes in the domain of -8 dB< a <8 dB and -0.5dB< c <0.5 dB. The range of c corresponds closely to the inherent instrument σ_0 accuracy of 0.2 dB. The difference in contour lines is caused by the geophysical distribution of the plotted (a, c)-pairs. For example, node 19 includes backscatter locations which have higher latitude positions than all other nodes; the maximum latitude decreases with decreasing node number. Note that it appears as if there are three maxima in the distribution: near $a \approx -3$ dB and $c \approx 0$ dB, near $a \approx 5$ dB and $c \approx 0$ dB and near $a \approx 1$ dB and $c \approx -0.2$ dB.

2.2 Generalization of the ice model

The ice model presented above depends on the typical ERS scatterometer antenna configuration of the measurement system. The value of the incidence angle of the fore beam is needed for determination of the correct ice line angle, while the incidence angle of the fore and mid beam are used to determine the mean ice backscatter value of the triplet. Other scatterometers, such as ASCAT and QUIKSCAT, or SAR, have a different configuration of radar beams and a generalization of the above defined ice model can therefore be useful.

The ice model is made node independent using the following representation of ice backscatter

$$\sigma(\theta, a) = \bar{\sigma}_0(\theta) + a \cdot \sigma_0^{ice}(\theta), \tag{14}$$

where the parameter a is fixed. The mean ice backscatter $\bar{\sigma}_0$ and along ice line backscatter σ_0^{ice} are fitted by cubic functions in θ. The mean backscatter as a function of θ is defined as

$$\begin{aligned} \bar{\sigma}_0(\theta) &= -4.185896 - 5.221865 \cdot 10^{-1}\,\theta \\ &\quad +8.57813 \cdot 10^{-3}\,(\theta)^2 - 6.54361 \cdot 10^{-5}\,(\theta)^3, \end{aligned} \tag{15}$$

and the along ice line backscatter is defined as

$$\begin{aligned} \sigma_0^{ice}(\theta) &= 1.44728 \cdot 10^{-1} + 1.732199 \cdot 10^{-2}\,\theta \\ &\quad -1.939816 \cdot 10^{-4}\,(\theta)^2 - 8.022119 \cdot 10^{-7}\,(\theta)^3. \end{aligned} \tag{16}$$

In Figure 7 the generalized ice model is plotted against θ for a number of values of the ice parameter a. Also shown in this figure is an empirical backscatter ice model over the Artic region derived by Cavenié, et al (1998). The parameters of this model are determined by minimization of a cost function using a large number of Artic backscatter triplets from January 1993. The dependence of this ice model on the incidence angle θ is also assumed to be cubic. The family of curves vary as a function of the backscatter value coefficient at 40 degrees incidence angle (σ_0^{40}). The general form of this ice model is

$$\sigma_0^{\text{Cavenié}} = A_1T + A_2T^2 + A_3T^3 + \sigma_0^{40}(1 + A_4T + A_5T^2 + A_6T^3), \tag{17}$$

where $T = (\theta - 40)/40$. For more details see Cavenié, et al (1998). The emperical ice model for $\sigma_0^{40} = -15$ has a nearly similar graph as the isotropy ice model for $a = 0$. For other pairs of a and σ_0^{40} for which the curves are relatively close, the models differ at low and high incidence angles. For incidence angles in the middle range, both ice models have similar graphs.

For the ERS configuration the ice line is determined by

$$\begin{pmatrix} \sigma_0^{fore} \\ \sigma_0^{aft} \\ \sigma_0^{mid} \end{pmatrix} = \begin{pmatrix} \bar{\sigma}_0(\theta^{fore}) \\ \bar{\sigma}_0(\theta^{aft}) \\ \bar{\sigma}_0(\theta^{mid}) \end{pmatrix} + a \begin{pmatrix} \sigma_0^{ice}(\theta^{fore}) \\ \sigma_0^{ice}(\theta^{aft}) \\ \sigma_0^{ice}(\theta^{mid}) \end{pmatrix} \tag{18}$$

The ice line, which lies in the (σ_0^{fore}=σ_0^{aft})-plane, determines the e_a vector. This implies that for ice backscatter we require that

$$\sigma_0^{fore} = \sigma_0^{aft}. \tag{19}$$

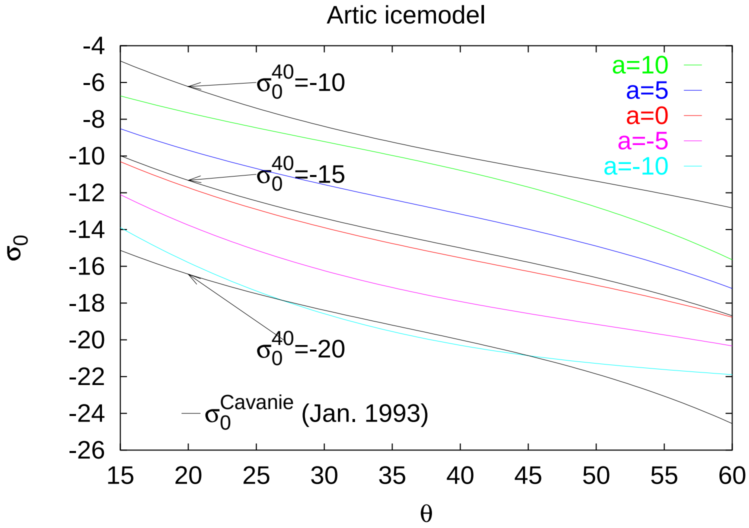


Figure 7: Plot of the generalized ice model for different values of the ice parameter a and an emperical backscatter ice model over artic sea ice Cavenié, et al (1998).

The other vectors e_b and e_c follow from this vector by demanding that e_b lies in the $\sigma_0^{fore}, \sigma_0^{aft}$ plane and e_c in the $(\sigma_0^{fore}{=}\sigma_0^{aft})$-plane. Write

$$\sigma_{ice}^{f,a} = \frac{1}{2}(\sigma_0^{ice}(\theta^{fore}) + \sigma_0^{ice}(\theta^{aft})) \text{ and } \sigma_{ice}^{m} = \sigma_0^{ice}(\theta^{mid}), \tag{20}$$

then

$$e_a = \left(2(\sigma_{ice}^{f,a})^2 + (\sigma_{ice}^{m})^2)\right)^{-\frac{1}{2}} \begin{pmatrix} \sigma_{ice}^{f,a} \\ \sigma_{ice}^{f,a} \\ \sigma_{ice}^{m} \end{pmatrix} \tag{21}$$

$$e_b = \left(-\sigma_{ice}^{f,a}\sqrt{2}\right)^{-1} \begin{pmatrix} -\sigma_{ice}^{f,a} \\ \sigma_{ice}^{f,a} \\ 0 \end{pmatrix} \tag{22}$$

$$e_c = \left(2(\sigma_{ice}^{f,a})^2 + (\sigma_{ice}^{m})^2\right)^{-\frac{1}{2}} \begin{pmatrix} -\sigma_{ice}^{m} \\ \sigma_{ice}^{m} \\ 2\sigma_{ice}^{f,a} \end{pmatrix} \tag{23}$$

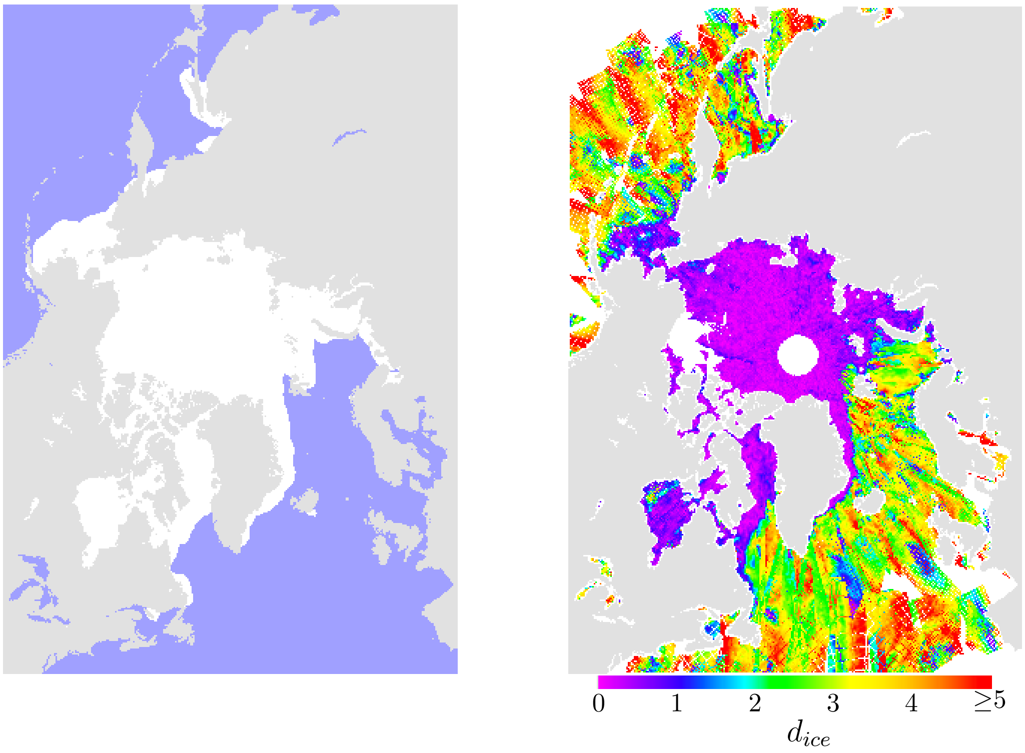
We verified that there is no significant change in the plots of Fig. 4-6 with these new definitions of a, b and c.

3 Results

Figure 8(a) shows the ice map as produced by IFREMER valid for the period 3-9 January 2000. Land is grey, sea is blue and ice is white in this figure. The United States is situated on the left of this image, Spain lies in the lower right corner, in the top right part lies Siberia and below the center of the image lies Greenland. This sea ice backscatter map is computed weekly from the scatterometer data of the C-band Advanced Microwave Instrument. Figure 8(b) shows the distance to the ice line for the same period in January. The ice signal in the ice distance plot is clear when figures (a) and (b) are compared. The IFREMER icemask however, seems to be a little reluctant to report no ice. For example near Spitsbergen, just right of the center, the IFREMER ice mask reports ice in this region while distances to the ice line are between 3 and 4 dB. This discrepancy is also observed near the east coast of Greenland.

On the other hand, there are regions where the distance to the ice line is small and the actual occurance of ice is impossible. This is due to the intersection of the ice line with the wind cone for particular incidence angles and wind directions. This phenomenon is observed at parts of the North Atlantic Ocean.

The ice line parameter b is the offset from the $\sigma_0^{fore}=\sigma_0^{aft}$-plane, see Figure 3. A plot of this parameter is shown in Figure 9(a) for the period as before. A backscatter triplet with a small

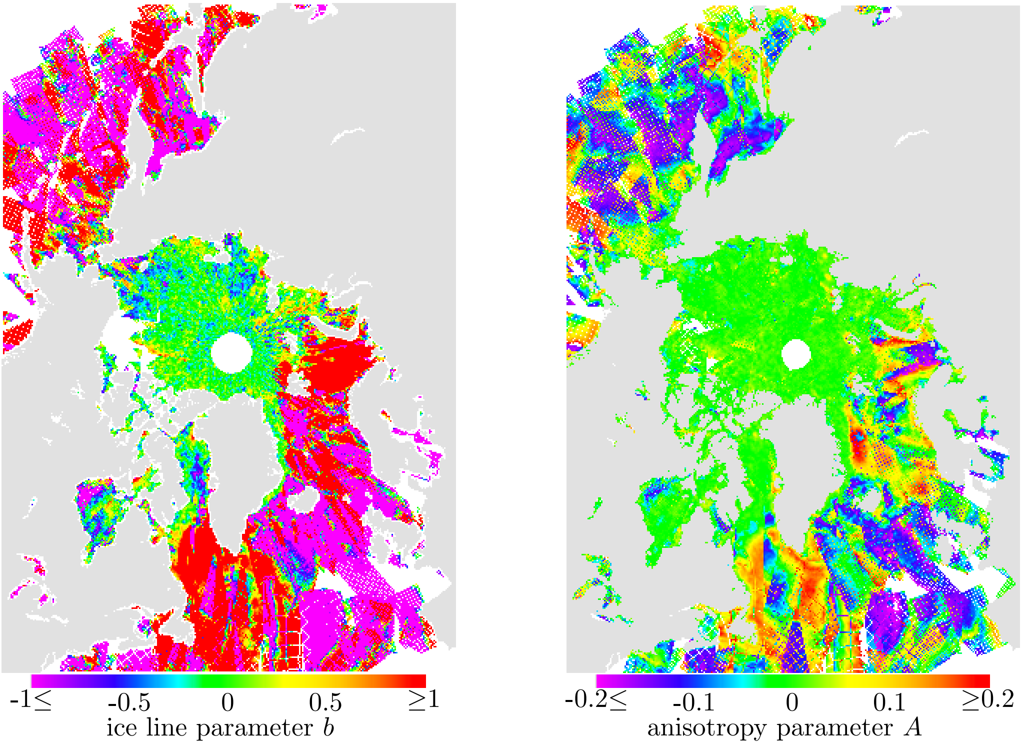


(a) (b)

Figure 8: (a) IFREMER ice map valid for the period 3-9 January 2000. Land is grey, sea is blue and ice is white; (b) distance to the ice line for the same period; no data is white. The legend is given below the image.

distance to the ice line implies that the parameter b is also small. Values of the parameter b show therefore a strong coherence with the distance to the ice line. The parameter b can be compared to the anisotropy coefficient A as defined in Equation 1. The anisotropy values are shown in Figure 9(b). Over the North Atlantic there are a large number of regions for which the anisotropy parameter is small, while there is no ice. The pattern of the swath is visible near the west coast of Greenland. The ice edge near the east coast of Greenland contains a larger area of low anisotropy values A. In this region the values of b are small for locations close to the coast and then show a sudden decrease for positions when moving eastward from the coast of Greenland. When ice is observed the anisotropy parameter A is small; the opposite is not true however.

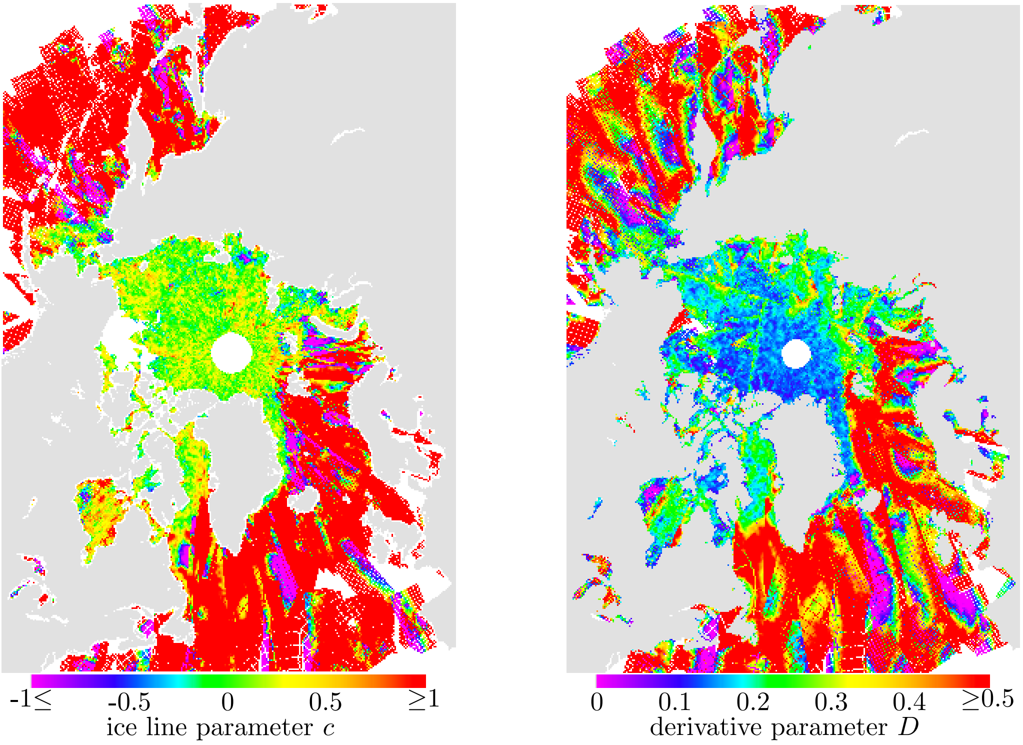
In Figure 10(a) the ice parameter c is plotted for the period in January. The green pixels are related to small values of c; the yellow (blue) pixels are due to small positive (negative) values of c. The parameter c is the offset from the ice line in the σ_0^{fore}=σ_0^{aft}-plane. A large positive value indicates that the σ_0^{mid} backscatter is much larger than σ_0^{fore} or σ_0^{aft}. This parameter can be compared to the derivative coefficient D as defined in Equation 2. The derivative with incidence angle is shown in Figure 10(b). The plot of the derivative shows swath patterns, because the derivative has an incidence angle dependent part. This is the reason for introducing an incidence



(a)

(b)

Figure 9: (a) The ice parameter b for the period 3-9 January 2000. Land is grey, no data is white; (b) anistropy parameter A for the same period. The legend is given below the images.



(a)

(b)

Figure 10: (a) The ice parameter c for the period 3-9 January 2000. Land is grey, no data is white; (b) derivative parameter D for the same period. The legend is given below the images.

angle dependent threshold. These swath patterns over ice are not visible in Fig 10(a). There are regions over the Atlantic where the values of D are small while there is no ice. For many of these regions the value of c is larger than 1 dB, indicating unlikely ice presence indeed.

The parameter D is assumed to reveal ice age. From Figure 10(b) we can distinguish first year and multi year ice, although this is not straight forward due to the incidence angle dependence of the value of D. As a first inventory on ice types one can mark the blue regions, when reported over ice, as multi-year ice and the yellow and green regions as first-year ice. From the plot of the along-the-ice-line parameter a, shown in Figure 11, we see that these regions indeed appear but with different colours. The red and yellow, when over ice, are due to multi-year ice and the green and blue regions are first-year ice surfaces.

A major difference between Figure 10(b) and Figure 11 is that there are no swath patterns in the latter, because the value of a is incidence angle independent by construction. The advantage of using a lies in the fact that a geophysical interpretation of the ice backscatter triplet can be made without reference to node or incidence angle information.

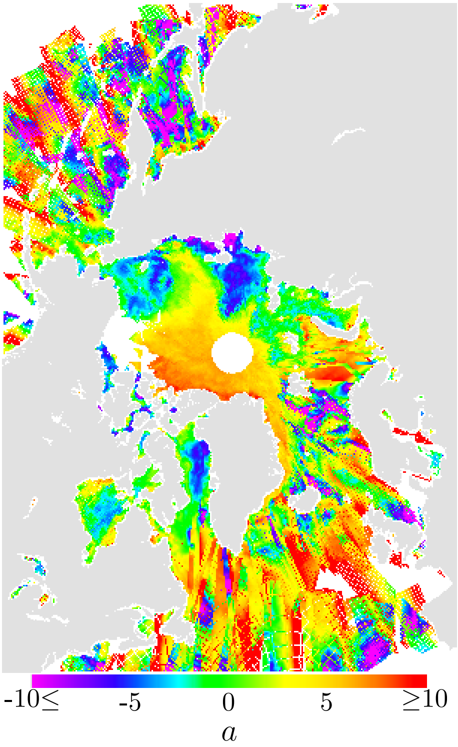


Figure 11: The distance along the ice line a for the period from January 3 to January 9, 2000.

4 Ice Discrimination

Ice probability and wind probability are correlated in the middle of the swath, whereas in the majority of cases wind probability and ice probability can be used to determine respectively wind or ice conditions. When the change in time of ice coverage is compared to change in (for example) wind speed the latter is much more dynamic. As such, ice history may be used to certify whether a condition is ice or water in case of doubt.

Figure 12 shows four locations for which the time series of distance to the ice line, distance to the wind cone, and of the ice line parameter a are discussed below. The four points all have different periods of ice coverage throughout the year. Point A and C are covered with ice only during the Artic winter: point A has a longer winter period than point C. Point B is covered with ice throughout the whole year and point D lies in the open water (i.e. no sea ice at any time) for almost the whole year.

The number of observations per node number is shown in Fig. 13. For all four points the distribution of observations with respect to node number is not constant. Moreover, the total number of observations for the points differs: Point B has twice the number of observations of point A which in turn has twice the number of observations of points C and D. This is due to the orbit of the ERS satellite: the higher the latitude the more passings of the satellite. Point A

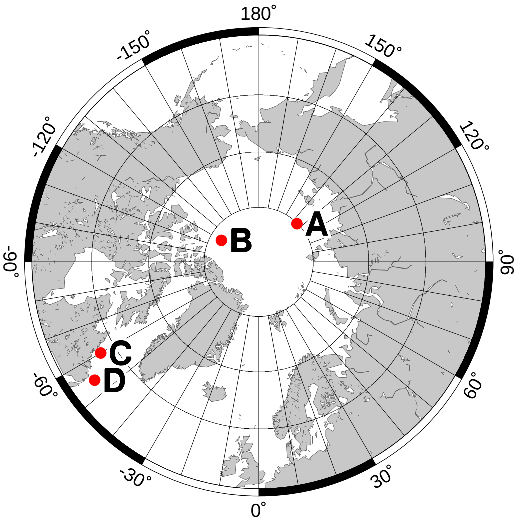


Figure 12: Four locations on the Artic for which the time series are calculated.

has almost no observations from the middle of the swath. Point B has a large amount of middle swath observations and less observations with high node numbers than low node numbers.

Figure 14 shows four panels with the time series of the points A,B,C and D . Each panel shows the time series of distance to the wind cone (top plot), distance to the ice line (middle plot) and the value of the ice parameter a (position along the ice line, bottom plot). Observations from nodes 7 up to 13 are denoted with triangles; open circles are observations from the rest of the swath. The locations correspond to a 25 kilometer squared area.

Point A (see Fig 14a)) is covered with ice from end October 1999 to June 2000. In this period the distance to the ice line is small (middle plot); the value of a (bottom plot) is more or less constant and the distance to the wind cone is large for almost all observations in this period. Between August 1999 and October 1999 ice is unlikely: the distance to the wind cone is small, the distance to the ice line is large and the ice line parameter changes rapidly with time.

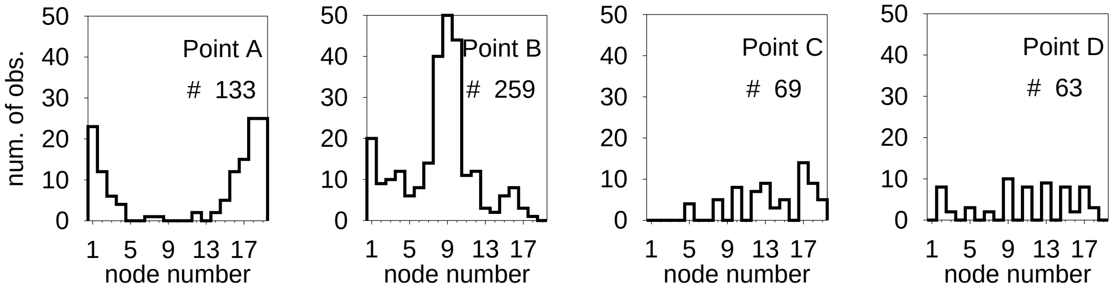
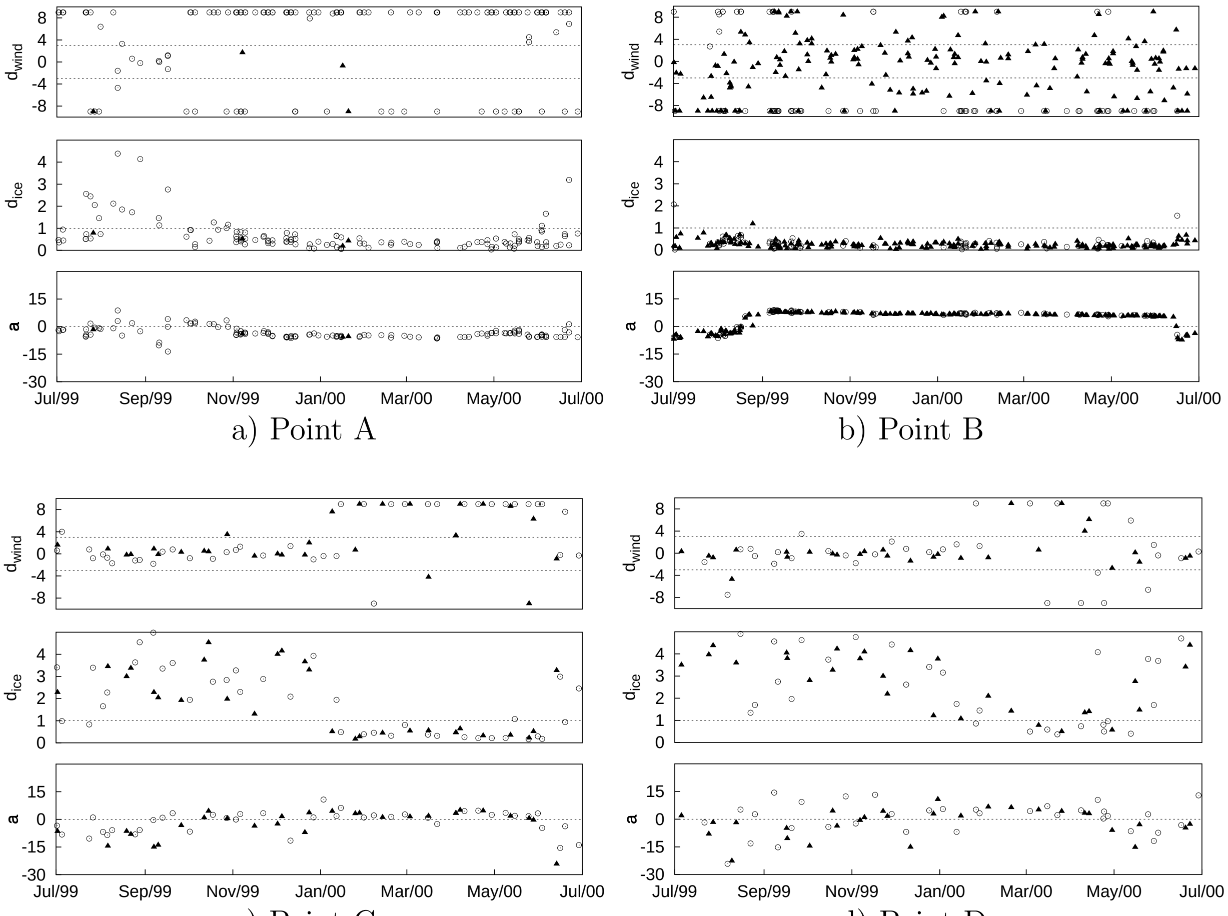


Figure 13: Number of observations for the four points in the period from July 1999 to July 2000.



a) Point A

b) Point B

c) Point C

d) Point D

Figure 14: Time series from July 1999 till July 2000 of the four locations shown in Fig. 12. Each panel shows the time series of distance to the wind cone (top plot), distance to the ice line (middle plot) and the value of the ice parameter a (distance along the ice line, bottom plot). Triangles in the plots denotes an observation from the nodes 7 up to 13 (the middle of the swath); open circles are observations from the rest of the swath.

This indicates that in this period no sea ice was present. In July 1999, the wind probability is generally low, the ice probability substantial, and the ice parameter close to zero, indicating ice. Note that there are a few points with both low wind and ice probability in between July and September 1999.

Point B (see Fig 14b)) is covered with ice during the whole period. From September 1999 to mid June 2000 the value of a is not changing substantially. Before September 1999 and after mid June 2000 the value of a has an opposite sign to the 'winter' period, indicating a different backscatter signal in the two periods. The cause of this different value of a may lie in subsequently melting and freezing of the top layer of the ice cover and the formation of puddles.

Table 1: Four classes of a single backscatter triplet.

(a)	probably sea when	$\|d_{wind}\| < 3$ and $\tilde{d}_{ice} > 1$
(b)	probably ice when	$\|d_{wind}\| > 3$ and $\tilde{d}_{ice} < 1$
(c)	mixed when	$\|d_{wind}\| < 3$ and $\tilde{d}_{ice} < 1$
(d)	no ice nor wind signal when	$\|d_{wind}\| > 3$ and $\tilde{d}_{ice} > 1$

Note also that the value of a is less constant indicating temporal changes in surface reflectivity, possibly caused by melting and freezing effects. This change of a is remarkable; in the beginning of September and mid June an outlier in the distance to the ice line with respect to the average is observed. The corresponding distance to the wind cone for the outlier in mid June is large indicating a low wind probability and its value of a fits in the sequence.

Note that d_{wind} behaves clearly different in panels a) and b), where low absolute values are much more likely in panel b). This can be explained by the set of nodes that differs in a) and b) as depicted in Fig. 13. Point A has relatively many observations from the outer swath (circles in Fig 14) and point B in the middle swath B (triangles).

Position C (see Fig 14c)) has a small distance to the ice line and a relative constant values of a from January 2000 to June 2000. Before and after this period the values of a are variable, the distance to the ice line is large and the distance to the wind cone is small. Note the mix of ice and wind points in July.

The time series of point D, shown in Fig 14d) has no clear signal in distance to the ice line nor in the parameter a. As expected, the distance to the wind cone is small for almost all observations (except in spring 2000).

The above four time series are used as a guide to build an ice screening algorithm.

4.1 Ice Discrimination Algorithm

The ice discrimination algorithm to flag a certain location as ice or no ice (or unknown) is based on the normalized distance ($\tilde{d}_{ice}$) to the ice line and distance to the wind cone (d_{wind}). A single backscatter triplet is classified into four different types marking the possible sea/ice state. These classification are then used to derive an ice map together with previous observed values a. The four classes of a single backscatter triplet are presented in Tab. 1.

Statistics of all backscatter measurements over the months July 1999 and November 1999 show that by a carefully chosen node dependent normalization coefficient the fraction of backscatter triplets which are incorrectly classified as probably sea (class (a)) (according to the IFREMER icemask) is smaller than 2% (independent of the node) (see Appendix A). note that these 2% do include uncertainties in the IFREMER ice mask. The normalization coefficients are shown in Table 2.

Generalization of this coefficient is given by

$$n_{ice}(\theta^{mid}) = \begin{cases} 3.978 - 6.981 \cdot 10^{-2}\theta^{mid} + 0.4\cos((\theta^{mid} - 18)/2.6) & \theta^{mid} < 40 \\ 1.0 & \theta^{mid} > 40 \end{cases} \tag{24}$$

Table 2: Normalization coefficients for distance to the ice line.

Node	1	2	3	4	5	6	7	8	9	10	11	12	13	14 - 19
Coef.	3.0	3.0	2.5	2.0	1.8	1.7	1.7	1.8	2.0	2.0	1.9	1.7	1.3	1.0

Table 3: Subclasses of classes a) and b) from Tab. 1.

(a1)	probably sea when	at least one of the three last observations (in time) was not classified as (a)
(a2)	sea when	the last 3 observations (in time) are classified as (a)
(b1)	probably ice (1) when	the number of previously reported values of a is smaller than 5
(b2)	probably ice (2) when	the standard deviation of minimal 5 and maximal 10 values of a is larger than 3
(b3)	ice when	the standard deviation of minimal 5 and maximal 10 values of a is smaller than 3

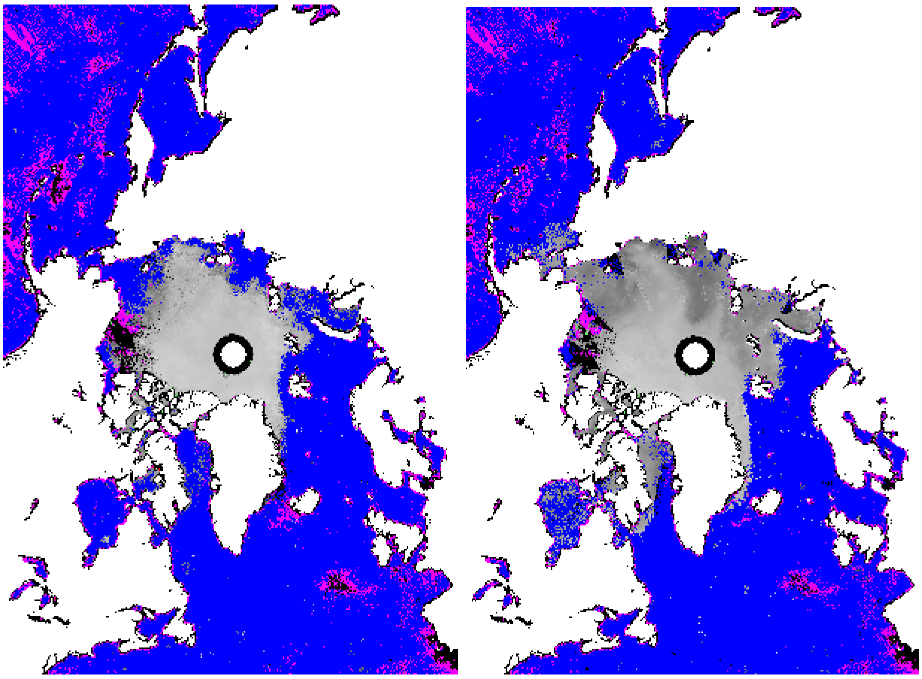
This coefficient maybe instrument or configuration dependent.

The normalization of the distance to the ice line did not result in a decrease of the fraction of incorrectly classified ice observations.

The above classification is solely based on single observations. To better distinguish between open sea and ice, a history of past observations must be used. Moreover to obtain a good coverage of the area of interest data from a few days is necessary. Each of the the classes (a) and (b) both split up into subclasses using in case of class (a) previously observed classes, and for the case of class (b) previously observed values of a. The classification is performed on a horizontal grid of 25×25 km, equivalent to the one used by IFREMER IFREMER (1996). The classification and observed values of a from 8 surrounding pixels and the central pixel are used to classify the sea/ice state of the central pixel.

The classes (a) and (b) are split up into 2 and 3 subclasses as shown in Tab. 3. Using the above subclasses for classes (a) and (b) the scores of erroneous ice classification reduces to smaller than 2% for all nodes. Local classes c) and d) remain undetermined.

The algorithm has been run on one year of data from July 1999 to June 2000. The results are shown in Fig. 15 for the dates 1 October 1999, 2 January 2000, 30 March 2000 and 30 June 2000. In Appendix B the first 9 ice maps are shown to illustrate how the icemaps develop from scratch. The blue pixels are open sea pixels (class (a2)), the purple pixels are class (a1) pixels, the grey pixels represent sea ice (class (b3)), with in grey scales the mean of a in a neighborhood of the pixel under consideration, black pixels are related to class (d) and white is no data (or land). No other classes were present. The mean of a is calculated using minimal 5 and maximal

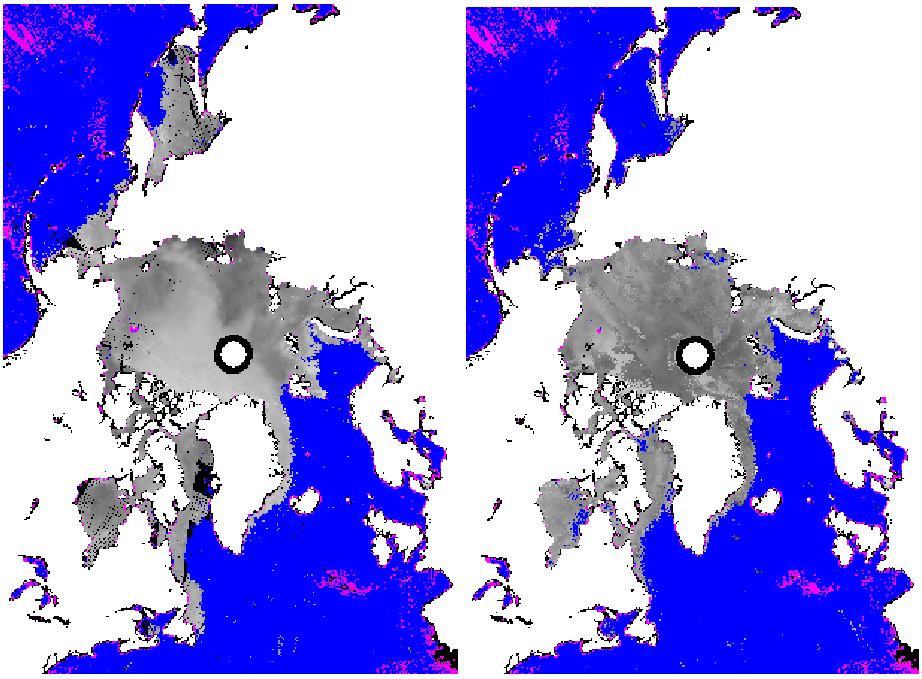


a) 1999/10/01 b) 2000/01/02

Figure 15: The results of the ice screening algorithm for the dates 1 October 1999, 2 January 2000, 30 March 2000 and 30 June 2000. The blue pixels are open sea pixels (class (a2)), the purple pixels are class (a1) pixels, the grey pixels represent sea ice (class (b3)), with in grey scales the mean of a, black pixels are class (d) and white is no data (or land).

10 values of a.

The icemaps in Fig. 15 show no ice in unrealistic places. Almost all purple (class (a1), probably sea) lie in open sea areas. The number of observations used to create the icemaps is shown in Fig 16. The number of observations is highly fluctuating throughout the period under consideration. Low numbers can be due to errors in the messages or the fact that the scatterometer was switched off. One message is a square of 19×19 backscatter triplets. When a complete message is over land this message is ommited in the data set. Differences in the number of observations are also due to the amount of land sampled in a give time period.



c) 2000/03/30 d) 2000/06/30

Figure 15: Continued.

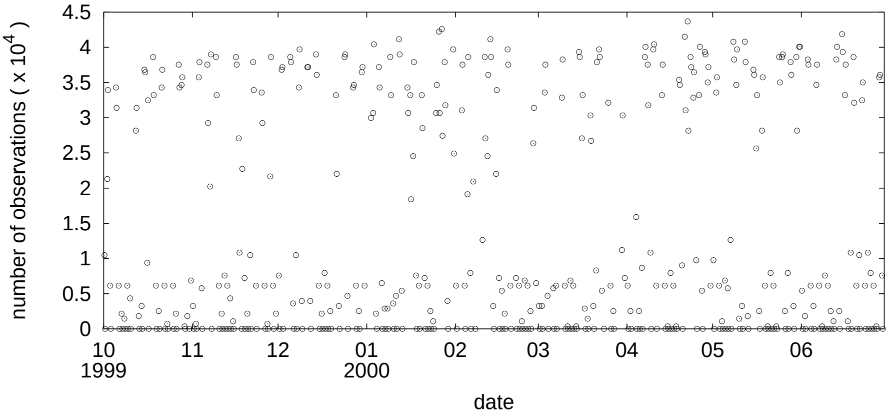


Figure 16: Number of observations in bins of 12 hours used to create the icemaps shown in Fig. 15.

5 Conclusions

In this text we constructed an ice model based on solely scatterometer data. The construction is based on the fact that backscatter measurements from sea ice are isotropic and therefore lie on a line in the 3D measurement space. An incidence angle dependent mapping of this line onto a reference incidence angle is constructed such that this distance to this line and distance along this line can be interpreted generically. Moreover, the construction of the ice line is generalized and a C-band ice model derived for future use with other scatterometers, such as ASCAT, with slightly different antenna configurations.

The distance to the ice line shows to be a good first guess of ice probability. Furthermore, the horizontal distributions of the along-the-ice-line parameter a is shown to contain information on the geophysical condition of the sea ice.

An ice discrimination algorithm was constructed based on the slower dynamical behaviour of sea ice compared to open sea. This algorithm is able to reproduce an ice map and proved to be successful in determining the sea ice coverage of the Artic. The sea ice is reported consistently with its presumed location. Sea ice appears with anticipated seasonal coverage and dynamics.

The algorithm was tested using ERS-2 scatterometer data and appears reliable in more than 98% of the cases. When two or more scatterometers are operated continuously ice maps can be updated more frequently, resulting in a futher improved ice map.

References

Cavenié A., F. Gohin, Y. Quilfen and P. Lecomte., Identification of Sea Ice Zones using AMI Wind: Physical Bases and Applications to the FDP and CERSAT Processing Chains, *Proceedings Second ERS-1 Symposium, Hamburg 11-14 October 1993*, ESA SP-361, January 1994.

Cavenié, A., An emperical C-band backscatter model over Artic Sea Ice from ERS-1 AMI-wind data, March 1998.

Gohin, F., Some active and passive microwave signatures of Antartic sea ice from mid-winter to spring 1991, *Int. J. Remote Sensing*, **vol 16**, No. 11, 2031-2054, 1995

IFREMER Scatterometer Polar Ice Grids User Manual. *IFREMER Report C2-MUTW-03-IF*, 1/12/1996

Ocean and Sea Ice SAF project team, “Ocean and Sea Ice Satellite Application Facility Software Requirements Document”, *EUMETSAT*, Darmstadt, Germany, December 1998

Stoffelen, A. C. M. D. L. T. Anderson, ERS-1 scatterometer data characteristics and wind retrieval skill, *Proceedings of the First ERS-1 Symposium, Cannes, France November 1992, European Space Agency Special Publication*, ESA SP-359, pp 41-47, 1993

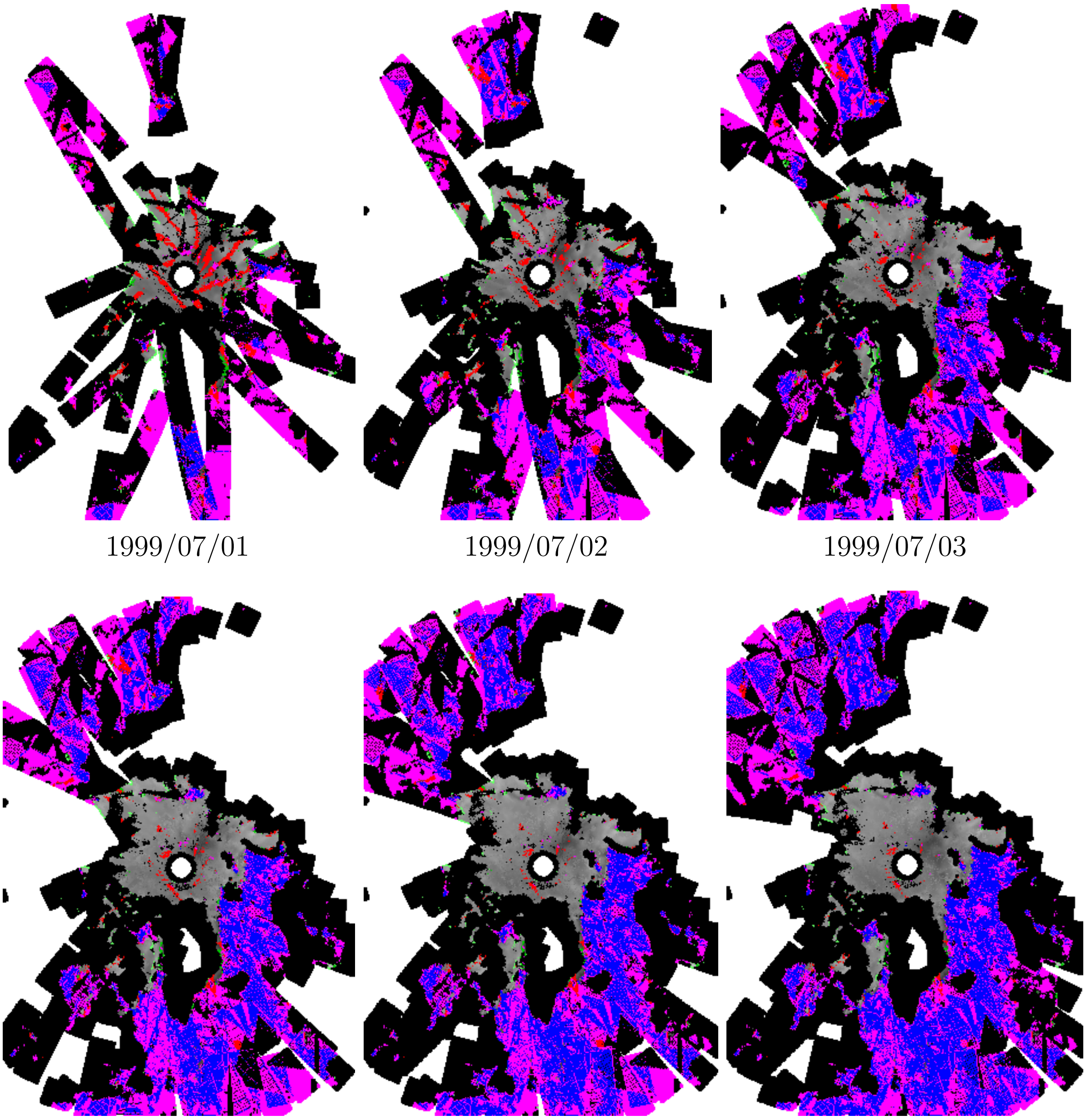
Stoffelen, Ad, Error Modeling and calibration; towards the true surface wind speed, *J. Geophys. Res.*, **103** (C4) 7755-7766, 1998

A Ice Classifications scores

		sea		*ice*		*both*		*none*		
July 1999		*class (a)*		*class (b)*		*class (c)*		*class (d)*		
Node	IFR	N	(%)	N	(%)	N	(%)	N	(%)	total
1	sea	7544	(0.66)	87	(0.01)	2	(0.00)	3775	(0.33)	11408
	ice	288	(0.02)	7438	(0.62)	0	(0.00)	4306	(0.36)	12032
2	sea	7540	(0.66)	105	(0.01)	4	(0.00)	3794	(0.33)	11443
	ice	226	(0.02)	7788	(0.64)	2	(0.00)	4131	(0.34)	12147
3	sea	7226	(0.63)	63	(0.01)	16	(0.00)	4153	(0.36)	11458
	ice	170	(0.01)	7758	(0.63)	20	(0.00)	4376	(0.36)	12324
4	sea	7119	(0.62)	53	(0.00)	10	(0.00)	4313	(0.38)	11495
	ice	174	(0.01)	7583	(0.61)	7	(0.00)	4680	(0.38)	12444
5	sea	7379	(0.64)	42	(0.00)	10	(0.00)	4071	(0.35)	11502
	ice	209	(0.02)	7721	(0.61)	0	(0.00)	4655	(0.37)	12585
6	sea	7348	(0.64)	48	(0.00)	25	(0.00)	4105	(0.36)	11526
	ice	288	(0.02)	8015	(0.63)	1	(0.00)	4370	(0.34)	12674
7	sea	7217	(0.63)	67	(0.01)	74	(0.01)	4151	(0.36)	11509
	ice	412	(0.03)	8153	(0.64)	63	(0.00)	4178	(0.33)	12806
8	sea	6978	(0.61)	47	(0.00)	255	(0.02)	4242	(0.37)	11522
	ice	338	(0.03)	7599	(0.59)	778	(0.06)	4251	(0.33)	12966
9	sea	6406	(0.56)	37	(0.00)	700	(0.06)	4391	(0.38)	11534
	ice	174	(0.01)	5717	(0.44)	2741	(0.21)	4400	(0.34)	13032
10	sea	6105	(0.53)	34	(0.00)	862	(0.08)	4480	(0.39)	11481
	ice	152	(0.01)	2691	(0.21)	5739	(0.44)	4521	(0.35)	13103
11	sea	5952	(0.52)	43	(0.00)	909	(0.08)	4606	(0.40)	11510
	ice	164	(0.01)	1400	(0.11)	6997	(0.53)	4587	(0.35)	13148
12	sea	5926	(0.52)	46	(0.00)	805	(0.07)	4708	(0.41)	11485
	ice	175	(0.01)	1901	(0.14)	6413	(0.48)	4800	(0.36)	13289
13	sea	6071	(0.53)	22	(0.00)	525	(0.05)	4897	(0.43)	11515
	ice	211	(0.02)	3564	(0.27)	4503	(0.34)	5140	(0.38)	13418
14	sea	6170	(0.54)	20	(0.00)	269	(0.02)	5046	(0.44)	11505
	ice	277	(0.02)	5816	(0.43)	1847	(0.14)	5530	(0.41)	13470
15	sea	6022	(0.52)	47	(0.00)	229	(0.02)	5234	(0.45)	11532
	ice	244	(0.02)	7015	(0.52)	722	(0.05)	5600	(0.41)	13581
16	sea	5994	(0.52)	64	(0.01)	189	(0.02)	5324	(0.46)	11571
	ice	182	(0.01)	6682	(0.49)	240	(0.02)	6525	(0.48)	13629
17	sea	5846	(0.50)	103	(0.01)	158	(0.01)	5506	(0.47)	11613
	ice	172	(0.01)	6412	(0.47)	50	(0.00)	7024	(0.51)	13658
18	sea	5710	(0.49)	104	(0.01)	123	(0.01)	5743	(0.49)	11680
	ice	177	(0.01)	6163	(0.45)	18	(0.00)	7342	(0.54)	13700
19	sea	5403	(0.46)	129	(0.01)	77	(0.01)	6066	(0.52)	11675
	ice	179	(0.01)	5922	(0.43)	11	(0.00)	7640	(0.56)	13752

		sea		*ice*		*both*		*none*		
November 1999		*class (a)*		*class (b)*		*class (c)*		*class (d)*		
Node	IFR	N	(%)	N	(%)	N	(%)	N	(%)	total
1	sea	7164	(0.66)	158	(0.01)	0	(0.00)	3541	(0.33)	10863
	ice	151	(0.01)	8798	(0.67)	0	(0.00)	4118	(0.32)	13067
2	sea	7198	(0.67)	122	(0.01)	22	(0.00)	3418	(0.32)	10760
	ice	168	(0.01)	8857	(0.67)	0	(0.00)	4218	(0.32)	13243
3	sea	6714	(0.63)	63	(0.01)	0	(0.00)	3949	(0.37)	10726
	ice	167	(0.01)	8899	(0.66)	0	(0.00)	4355	(0.32)	13421
4	sea	6422	(0.60)	31	(0.00)	0	(0.00)	4252	(0.40)	10705
	ice	160	(0.01)	8912	(0.66)	0	(0.00)	4479	(0.33)	13551
5	sea	6734	(0.63)	48	(0.00)	34	(0.00)	3830	(0.36)	10646
	ice	158	(0.01)	8950	(0.65)	3	(0.00)	4599	(0.34)	13710
6	sea	6709	(0.63)	53	(0.00)	153	(0.01)	3737	(0.35)	10652
	ice	145	(0.01)	8997	(0.65)	11	(0.00)	4657	(0.34)	13810
7	sea	6476	(0.61)	74	(0.01)	330	(0.03)	3719	(0.35)	10599
	ice	155	(0.01)	9007	(0.64)	79	(0.01)	4744	(0.34)	13985
8	sea	6284	(0.60)	34	(0.00)	554	(0.05)	3686	(0.35)	10558
	ice	137	(0.01)	7474	(0.53)	1599	(0.11)	4914	(0.35)	14124
9	sea	5946	(0.56)	38	(0.00)	819	(0.08)	3755	(0.36)	10558
	ice	134	(0.01)	1265	(0.09)	7851	(0.55)	5009	(0.35)	14259
10	sea	5819	(0.55)	53	(0.01)	859	(0.08)	3807	(0.36)	10538
	ice	115	(0.01)	1534	(0.11)	7520	(0.52)	5218	(0.36)	14387
11	sea	5739	(0.54)	57	(0.01)	847	(0.08)	3973	(0.37)	10616
	ice	119	(0.01)	5647	(0.39)	3338	(0.23)	5377	(0.37)	14481
12	sea	5683	(0.54)	63	(0.01)	790	(0.07)	4079	(0.38)	10615
	ice	133	(0.01)	7640	(0.52)	1341	(0.09)	5472	(0.38)	14586
13	sea	5936	(0.56)	53	(0.00)	462	(0.04)	4205	(0.39)	10656
	ice	167	(0.01)	8415	(0.57)	457	(0.03)	5663	(0.39)	14702
14	sea	6210	(0.58)	48	(0.00)	148	(0.01)	4276	(0.40)	10682
	ice	182	(0.01)	8521	(0.58)	105	(0.01)	5944	(0.40)	14752
15	sea	6229	(0.58)	71	(0.01)	117	(0.01)	4325	(0.40)	10742
	ice	149	(0.01)	8527	(0.58)	21	(0.00)	6101	(0.41)	14798
16	sea	6188	(0.57)	80	(0.01)	86	(0.01)	4419	(0.41)	10773
	ice	126	(0.01)	7585	(0.51)	4	(0.00)	7090	(0.48)	14805
17	sea	6167	(0.57)	99	(0.01)	57	(0.01)	4491	(0.42)	10814
	ice	100	(0.01)	7152	(0.48)	3	(0.00)	7536	(0.51)	14791
18	sea	6141	(0.57)	113	(0.01)	33	(0.00)	4560	(0.42)	10847
	ice	81	(0.01)	6823	(0.46)	2	(0.00)	7886	(0.53)	14792
19	sea	6071	(0.56)	91	(0.01)	16	(0.00)	4680	(0.43)	10858
	ice	79	(0.01)	6556	(0.44)	5	(0.00)	8160	(0.55)	14800

B Icemaps of 1-9 July 1999



1999/07/01

1999/07/02

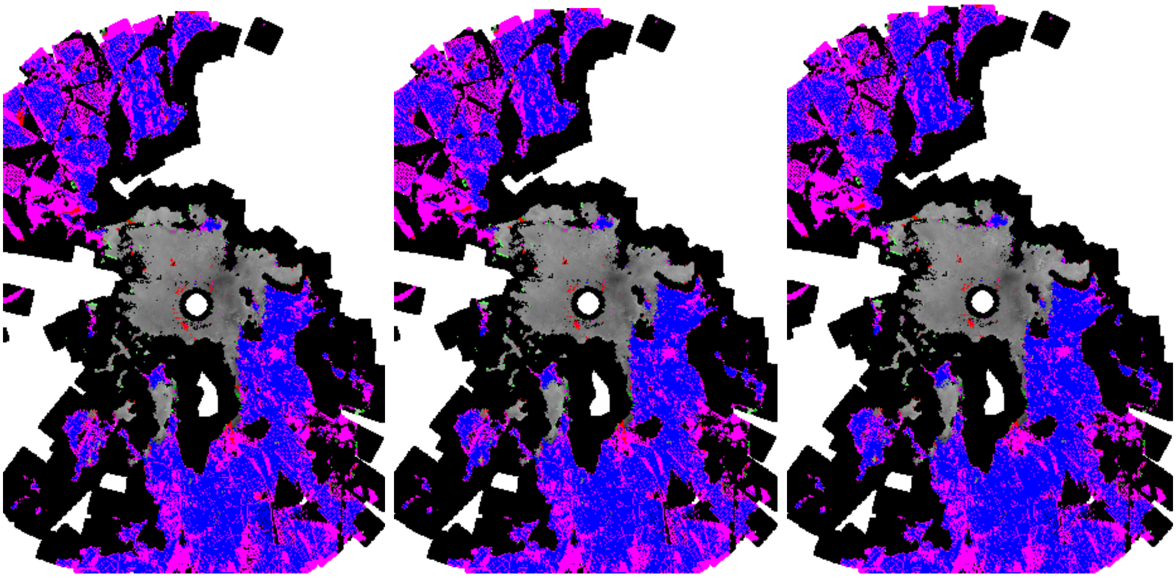
1999/07/03

1999/07/04

1999/07/05

1999/07/06

Figure 17: The results of the ice screening algorithm. See text below for explanation of the colors.



1999/07/07 1999/07/08 1999/07/09

Figure 17: Continued.

Explanation of the colors in the figures:

White	No Data
Purple	Sea but not all last three are sea, class (a1)
Blue	Sea as defined by class (a2)
Light Green	Ice but number of observations smaller than 5, class (b1)
Dark Green	Ice but standard deviation too large, class (b2)
Grey	Ice as defined by class (b3)
Red	Ice and Wind as defined by class (c)
Black	No wind no Ice as defined by class (d)

C Distributions of d_{ice} and d_{wind}

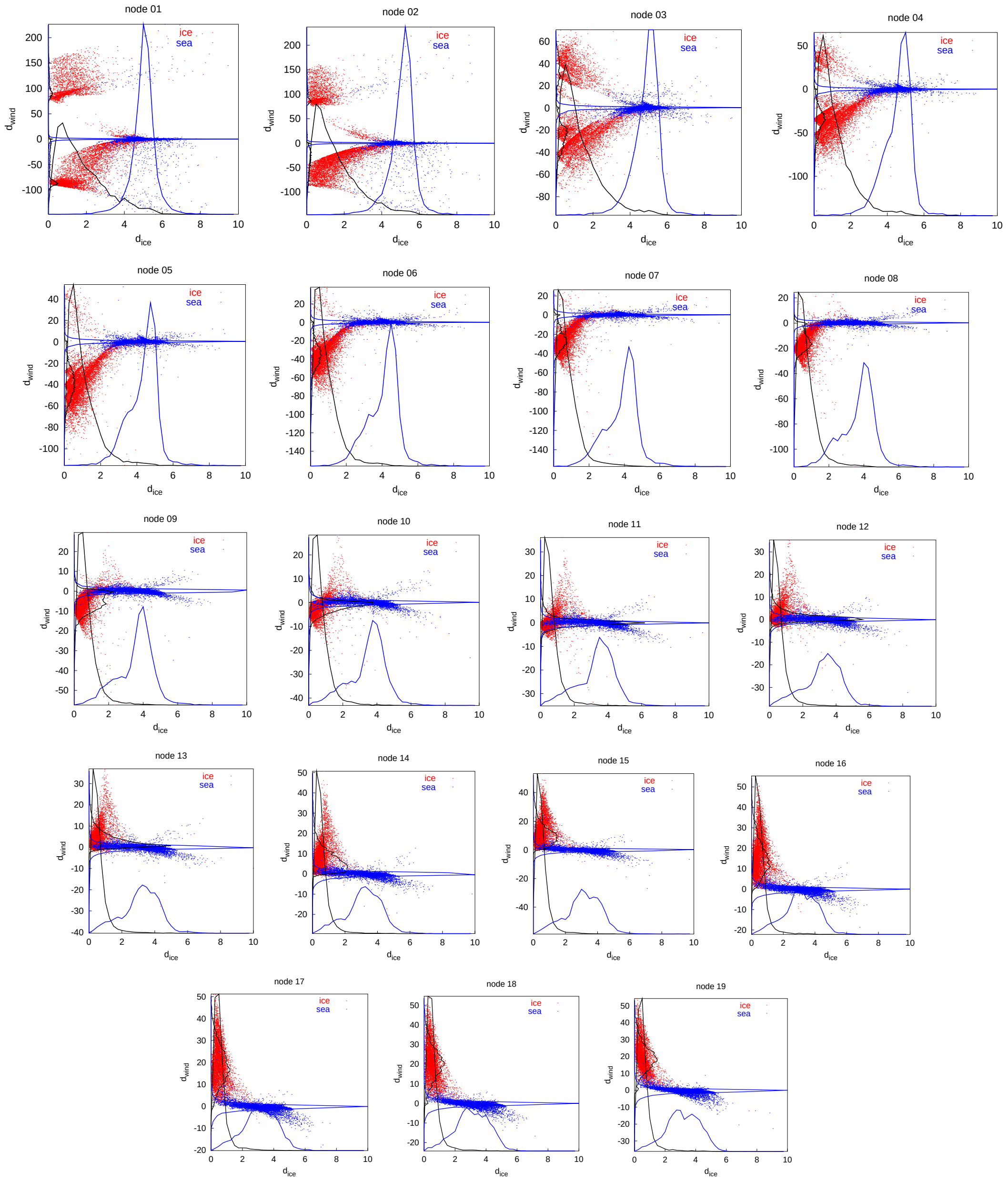


Figure 18: Distributions of d_{ice} (black line) and d_{wind} (blue line) for ice (red) and sea (blue) points. ERS data from July 1999. Sea/ice discrimination based upon conservative IFREMER icemask from July 1999.

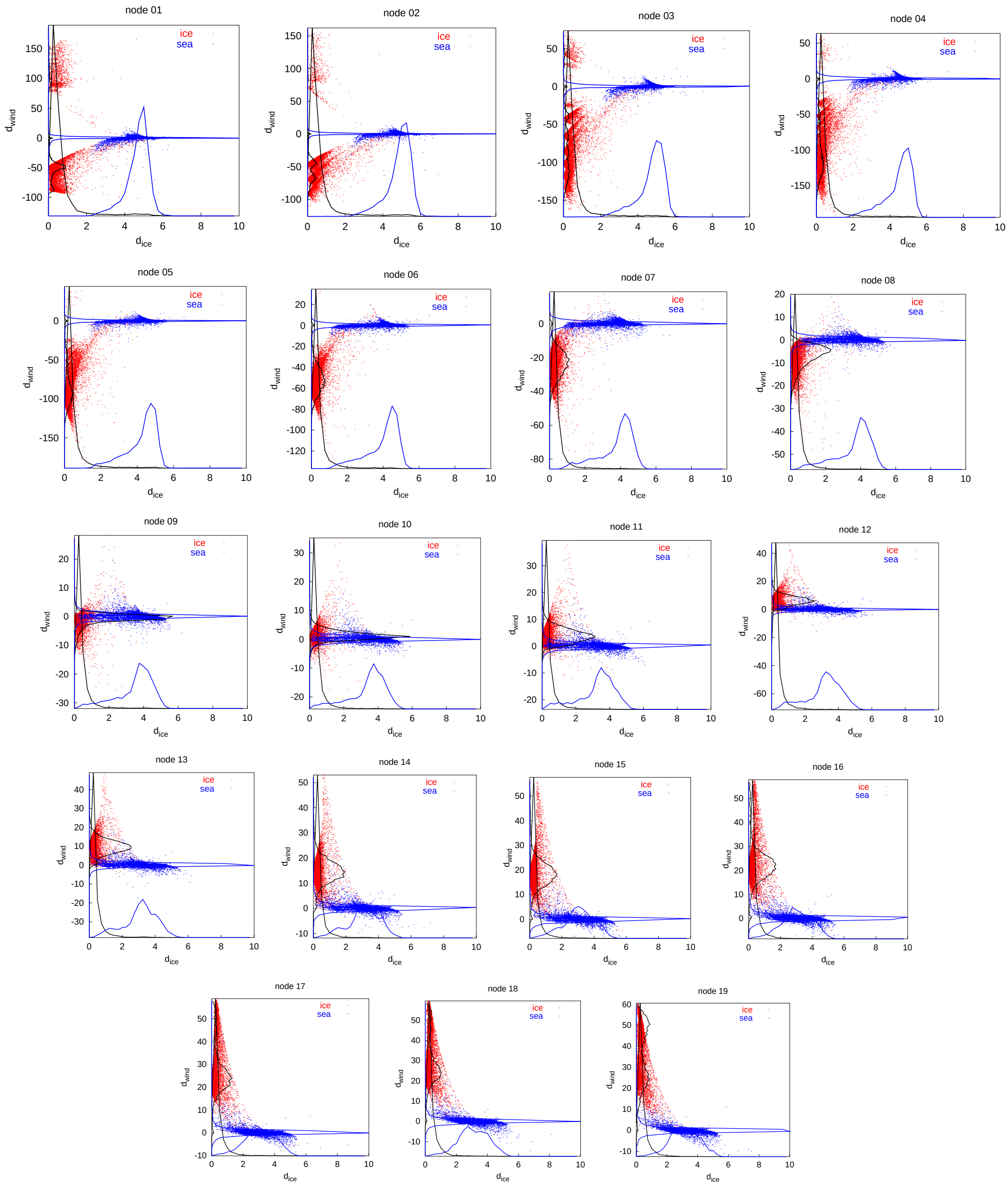
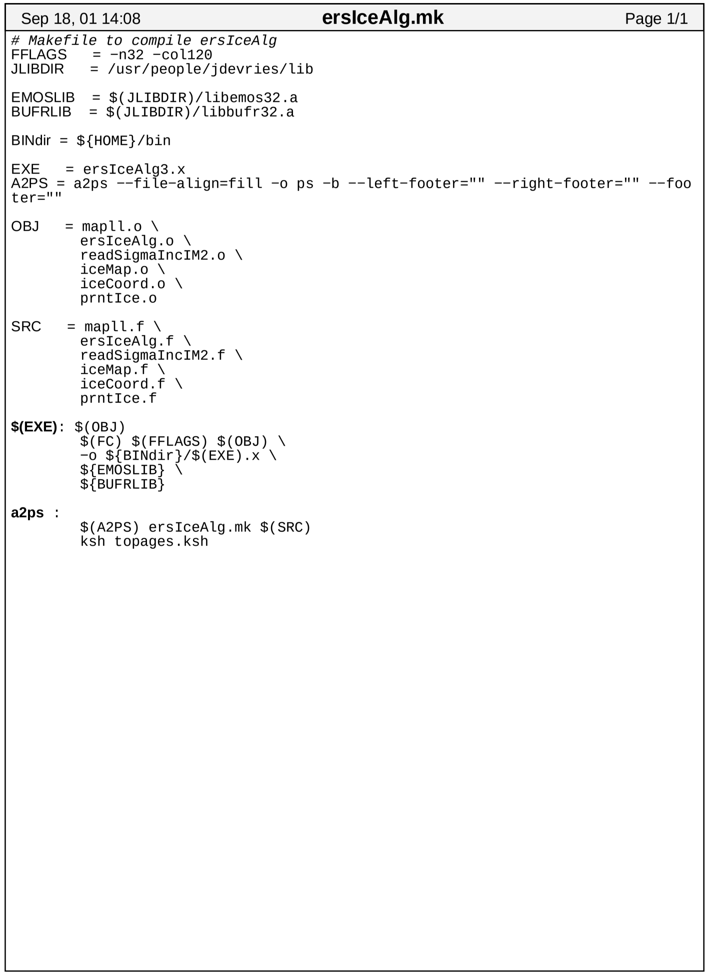


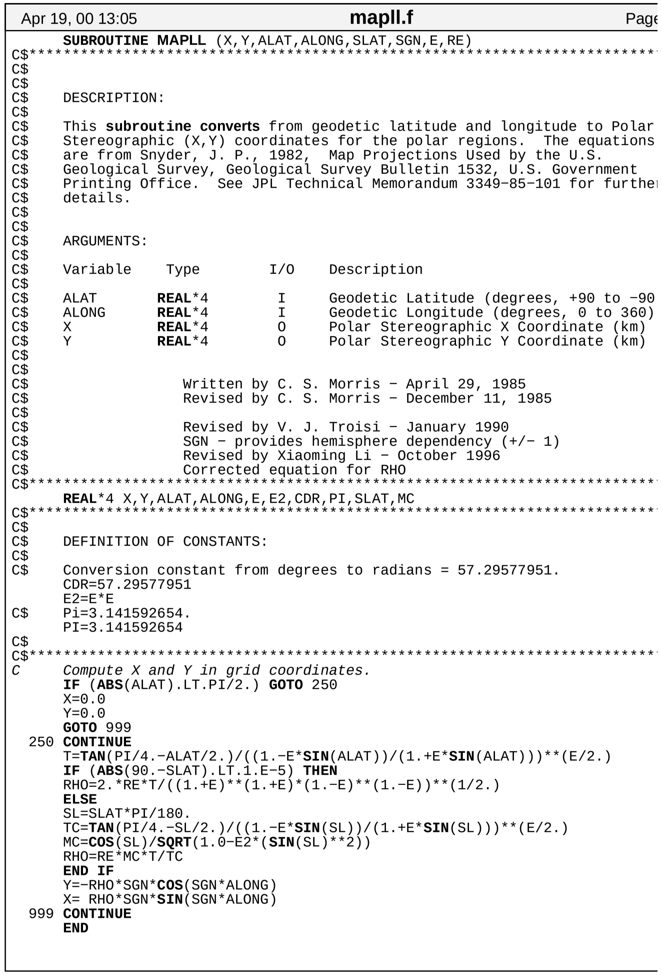
Figure 19: Distributions of d_{ice} (black line) and d_{wind} (blue line) for ice (red) and sea (blue) points. ERS data from November 1999. Sea/ice discrimination based upon conservative IFREMER icemask from November 1999.

D Source Code



Sep 18, 01 14:08 **ersIceAlg.mk** Page 1/1

```
# Makefile to compile ersIceAlg
FFLAGS   = -n32 -col120
JLIBDIR  = /usr/people/jdevries/lib

EMOSLIB  = $(JLIBDIR)/libemos32.a
BUFRLIB  = $(JLIBDIR)/libbufr32.a

BINdir = ${HOME}/bin

EXE   = ersIceAlg3.x
A2PS = a2ps --file-align=fill -o ps -b --left-footer="" --right-footer="" --foo
ter=""

OBJ   = mapll.o \
        ersIceAlg.o \
        readSigmaIncIM2.o \
        iceMap.o \
        iceCoord.o \
        prntIce.o

SRC   = mapll.f \
        ersIceAlg.f \
        readSigmaIncIM2.f \
        iceMap.f \
        iceCoord.f \
        prntIce.f

$(EXE): $(OBJ)
        $(FC) $(FFLAGS) $(OBJ) \
        -o ${BINdir}/$(EXE).x \
        ${EMOSLIB} \
        ${BUFRLIB}

a2ps :
        $(A2PS) ersIceAlg.mk $(SRC)
        ksh topages.ksh
```



Apr 19, 00 13:05 **mapll.f** Page

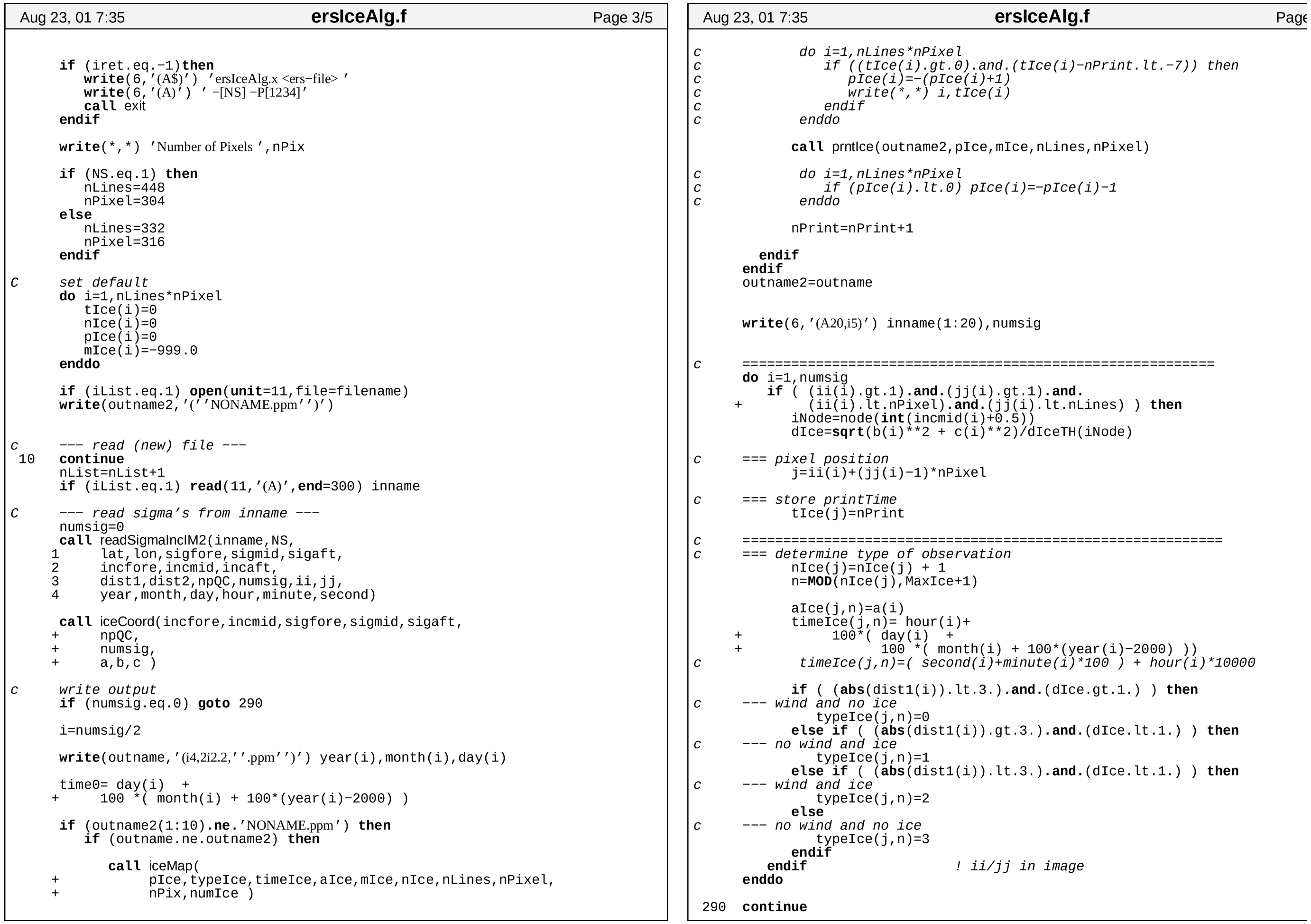
```
      SUBROUTINE MAPLL (X,Y,ALAT,ALONG,SLAT,SGN,E,RE)
C$*************************************************************************
C$
C$
C$    DESCRIPTION:
C$
C$    This subroutine converts from geodetic latitude and longitude to Polar
C$    Stereographic (X,Y) coordinates for the polar regions.  The equations
C$    are from Snyder, J. P., 1982,  Map Projections Used by the U.S.
C$    Geological Survey, Geological Survey Bulletin 1532, U.S. Government
C$    Printing Office.  See JPL Technical Memorandum 3349-85-101 for furthe
C$    details.
C$
C$
C$    ARGUMENTS:
C$
C$    Variable    Type        I/O    Description
C$
C$    ALAT       REAL*4        I     Geodetic Latitude (degrees, +90 to -90
C$    ALONG      REAL*4        I     Geodetic Longitude (degrees, 0 to 360)
C$    X          REAL*4        O     Polar Stereographic X Coordinate (km)
C$    Y          REAL*4        O     Polar Stereographic Y Coordinate (km)
C$
C$
C$                 Written by C. S. Morris - April 29, 1985
C$                 Revised by C. S. Morris - December 11, 1985
C$
C$                 Revised by V. J. Troisi - January 1990
C$                 SGN - provides hemisphere dependency (+/- 1)
C$                 Revised by Xiaoming Li - October 1996
C$                 Corrected equation for RHO
C$*************************************************************************
      REAL*4 X,Y,ALAT,ALONG,E,E2,CDR,PI,SLAT,MC
C$*************************************************************************
C$
C$    DEFINITION OF CONSTANTS:
C$
C$    Conversion constant from degrees to radians = 57.29577951.
      CDR=57.29577951
      E2=E*E
C$    Pi=3.141592654.
      PI=3.141592654
C$
C$*************************************************************************
C     Compute X and Y in grid coordinates.
      IF (ABS(ALAT).LT.PI/2.) GOTO 250
      X=0.0
      Y=0.0
      GOTO 999
  250 CONTINUE
      T=TAN(PI/4.-ALAT/2.)/((1.-E*SIN(ALAT))/(1.+E*SIN(ALAT)))**(E/2.)
      IF (ABS(90.-SLAT).LT.1.E-5) THEN
      RHO=2.*RE*T/((1.+E)**(1.+E)*(1.-E)**(1.-E))**(1/2.)
      ELSE
      SL=SLAT*PI/180.
      TC=TAN(PI/4.-SL/2.)/((1.-E*SIN(SL))/(1.+E*SIN(SL)))**(E/2.)
      MC=COS(SL)/SQRT(1.0-E2*(SIN(SL)**2))
      RHO=RE*MC*T/TC
      END IF
      Y=-RHO*SGN*COS(SGN*ALONG)
      X= RHO*SGN*SIN(SGN*ALONG)
  999 CONTINUE
      END
```

Aug 23, 01 7:35 **ersIceAlg.f** Page 1/5

```
      program ersIceAlg
      implicit none
C     only latest ice model is used
c     functions
      real rfmt

      integer nargs, iargc

      character*128 inname
      character*128 outname
      character*128 outname2
      character*128 outname3
      character*128 filename
      character*128 iceName
      character*4 dummy
      character*128 dummy2
      character  dummy1

      integer kvals              ! max number of values
      parameter (kvals=80000)

c       --- ers data ---
      real sigfore(kvals)        ! sigma0 fore beam
      real sigmid(kvals)         ! sigma0 mid beam
      real sigaft(kvals)         ! sigma0 aft beam
      real dist1(kvals)          ! 1st distance to wind cone
      real dist2(kvals)          ! 2nd distance -> distance to ice line
      real incfore(kvals)        ! incidence angle fore beam
      real incmid(kvals)         ! incidence angle mid beam
      real incaft(kvals)         ! incidence angle aft beam
      real lat(kvals),lon(kvals)
      integer year(kvals)
      integer month(kvals)
      integer day(kvals)
      integer hour(kvals)
      integer minute(kvals)
      integer second(kvals)

      integer ii(kvals),jj(kvals) ! pixel locations of backscatter node

      integer npQC(kvals)
C       from : ~driesena/pp_work/source/INVERT.f
C       Inversion only performed if rVal(npQC,i) ne rMDI and
C       rVal(npFb,i) ne rMDI
C       1.           : Distance to cone too big, wind untouched
C       2.           : Low speed , bogus winds as solution
C       4.           : 1 solution, low speed , bogus 2nd solution added
C       8.           : 1 solution, high speed, no wind solution
C       16.          : High speed, no wind solution
      integer numsig             ! number of sigma's read
      integer NS                 ! North/South pole
      integer size

      integer nLines,nPixel      ! number of lines and pixels


      real dWind,dIce

      real a(kvals),b(kvals),c(kvals) ! ice coordinates

c     --- ice history
      integer MaxIce              ! max number of ice hist. values
      parameter (MaxIce=10)
      real aIce(458*314,MaxIce)
      real mIce(458*314)
```

Aug 23, 01 7:35 **ersIceAlg.f** Page

```
      integer nIce(458*314)
      integer pIce(458*314)
      integer typeIce(458*314,MaxIce)
      integer timeIce(458*314,MaxIce)
      integer tIce(458*314)

      integer iNode

      integer nPix

c     --- int(incidence angle +0.5) -> node number ---
      integer node(18:45)
      data node / 1, 1, 2, 0, 3, 4, 4, 5, 5, 6, 0, 7, 8, 0, 9,
     +     10, 10, 11, 12, 0, 13, 14, 15, 15, 16, 17, 18, 19/
c     --- distance to ice threshold
      real dIceTH(19)
      data dIceTH
     +     / 3.0, 3.0, 2.5, 2.0, 1.8, 1.7, 1.7, 1.8, 2.0,
     +       2.0, 1.9, 1.7, 1.3, 1.0, 1.0, 1.0, 1.0, 1.0,
     +       1.0 /

c     --- locals ---
      integer iret,iList,nList,nPrint
      integer i,j
      integer ii0,jj0
      integer k,n
      integer n1,n2
      real dIce
      real x,x2
      integer ir,ib,ig
      integer oldday

      integer time0

      integer numSea
      integer numIce

C     --- defaults
      numIce=10
      nPix=1
      nPrint=1
C     --- read arguments
      nargs = iargc()
      if (nargs.lt.3) then
         iret=-1
      else
         call getarg(1,dummy)
         if (dummy(1:1).eq.'-') then
            if (dummy(1:2).eq.'-f') then
               call getarg(2,fileName)
               iList=1
            endif
         else
            call getarg(1,inname)
         endif
      endif
      call getarg(iList+2,dummy)
      if (dummy(1:2).eq.'-N') NS=1
      if (dummy(1:2).eq.'-S') NS=2

      call getarg(iList+3,dummy)
      if (dummy(1:3).eq.'-P1') nPix=1
      if (dummy(1:3).eq.'-P2') nPix=5
      if (dummy(1:3).eq.'-P3') nPix=9
      if (dummy(1:3).eq.'-P4') nPix=13
```

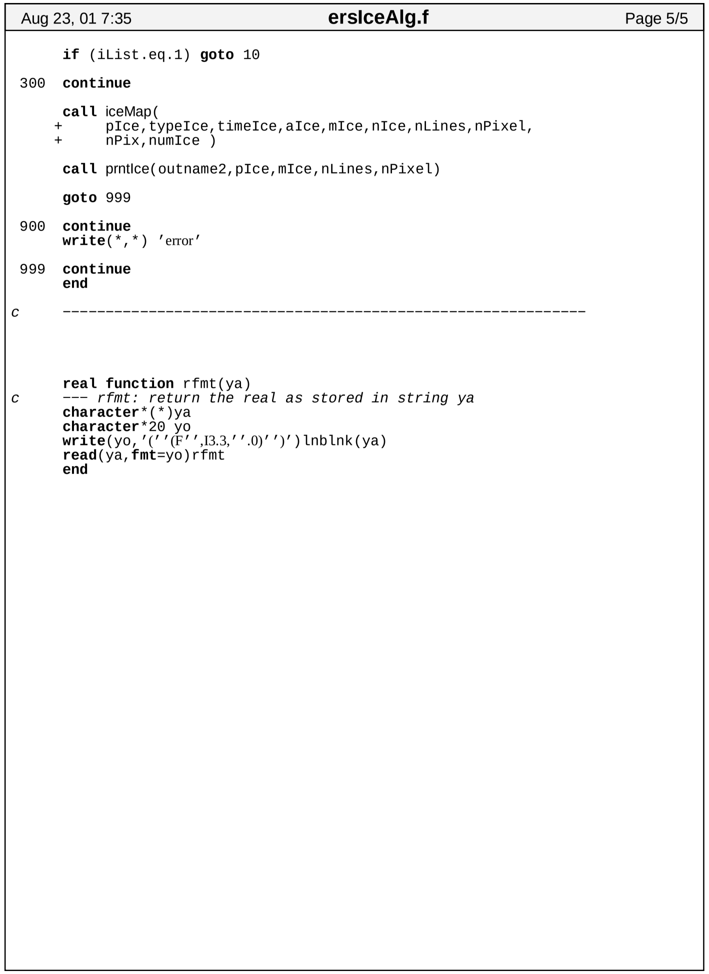


Aug 23, 01 7:35 **ersIceAlg.f** Page 3/5

```
      if (iret.eq.-1)then
         write(6,'(A$)') 'ersIceAlg.x <ers-file> '
         write(6,'(A)') ' -[NS] -P[1234]'
         call exit
      endif

      write(*,*) 'Number of Pixels ',nPix

      if (NS.eq.1) then
         nLines=448
         nPixel=304
      else
         nLines=332
         nPixel=316
      endif

C     set default
      do i=1,nLines*nPixel
         tIce(i)=0
         nIce(i)=0
         pIce(i)=0
         mIce(i)=-999.0
      enddo

      if (iList.eq.1) open(unit=11,file=filename)
      write(outname2,'(''NONAME.ppm'')')


c     --- read (new) file ---
 10   continue
      nList=nList+1
      if (iList.eq.1) read(11,'(A)',end=300) inname

C     --- read sigma's from inname ---
      numsig=0
      call readSigmaIncIM2(inname,NS,
     1     lat,lon,sigfore,sigmid,sigaft,
     2     incfore,incmid,incaft,
     3     dist1,dist2,npQC,numsig,ii,jj,
     4     year,month,day,hour,minute,second)

      call iceCoord(incfore,incmid,sigfore,sigmid,sigaft,
     +     npQC,
     +     numsig,
     +     a,b,c )

c     write output
      if (numsig.eq.0) goto 290

      i=numsig/2

      write(outname,'(i4,2i2.2,''.ppm'')') year(i),month(i),day(i)

      time0= day(i)  +
     +     100 *( month(i) + 100*(year(i)-2000) )

      if (outname2(1:10).ne.'NONAME.ppm') then
         if (outname.ne.outname2) then

            call iceMap(
     +           pIce,typeIce,timeIce,aIce,mIce,nIce,nLines,nPixel,
     +           nPix,numIce )
```

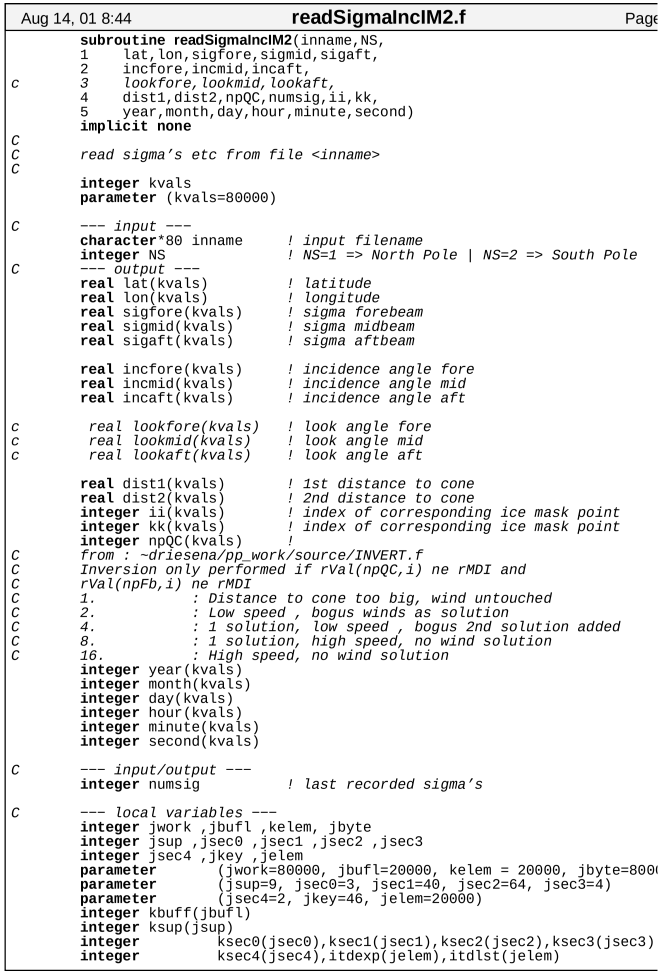
Aug 23, 01 7:35 **ersIceAlg.f** Page

```
c            do i=1,nLines*nPixel
c               if ((tIce(i).gt.0).and.(tIce(i)-nPrint.lt.-7)) then
c                  pIce(i)=-(pIce(i)+1)
c                  write(*,*) i,tIce(i)
c               endif
c            enddo

           call prntIce(outname2,pIce,mIce,nLines,nPixel)

c            do i=1,nLines*nPixel
c               if (pIce(i).lt.0) pIce(i)=-pIce(i)-1
c            enddo

           nPrint=nPrint+1

         endif
      endif
      outname2=outname


      write(6,'(A20,i5)') inname(1:20),numsig


c     ==========================================================
      do i=1,numsig
         if ( (ii(i).gt.1).and.(jj(i).gt.1).and.
     +        (ii(i).lt.nPixel).and.(jj(i).lt.nLines) ) then
           iNode=node(int(incmid(i)+0.5))
           dIce=sqrt(b(i)**2 + c(i)**2)/dIceTH(iNode)

c     === pixel position
           j=ii(i)+(jj(i)-1)*nPixel

c     === store printTime
           tIce(j)=nPrint

c     ===========================================================
c     === determine type of observation
           nIce(j)=nIce(j) + 1
           n=MOD(nIce(j),MaxIce+1)

           aIce(j,n)=a(i)
           timeIce(j,n)= hour(i)+
     +          100*( day(i)  +
     +                100 *( month(i) + 100*(year(i)-2000) ))
c            timeIce(j,n)=( second(i)+minute(i)*100 ) + hour(i)*10000

           if ( (abs(dist1(i)).lt.3.).and.(dIce.gt.1.) ) then
c     --- wind and no ice
              typeIce(j,n)=0
           else if ( (abs(dist1(i)).gt.3.).and.(dIce.lt.1.) ) then
c     --- no wind and ice
              typeIce(j,n)=1
           else if ( (abs(dist1(i)).lt.3.).and.(dIce.lt.1.) ) then
c     --- wind and ice
              typeIce(j,n)=2
           else
c     --- no wind and no ice
              typeIce(j,n)=3
           endif
         endif                   ! ii/jj in image
      enddo

 290  continue
```



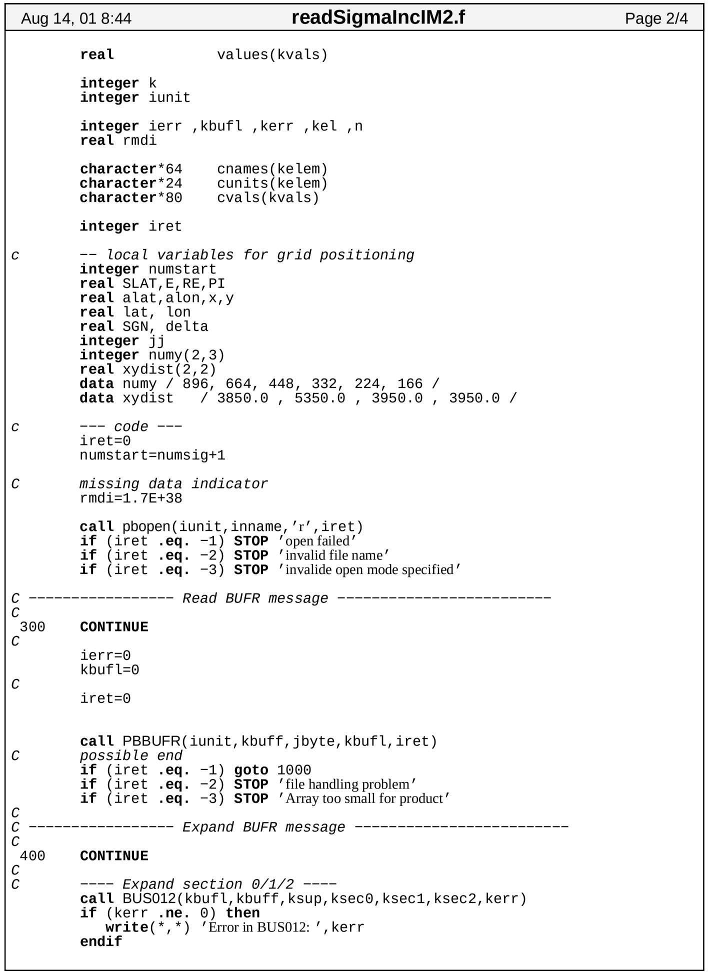
Aug 23, 01 7:35 **ersIceAlg.f** Page 5/5

```
      if (iList.eq.1) goto 10

 300  continue

      call iceMap(
     +     pIce,typeIce,timeIce,aIce,mIce,nIce,nLines,nPixel,
     +     nPix,numIce )

      call prntIce(outname2,pIce,mIce,nLines,nPixel)

      goto 999

 900  continue
      write(*,*) 'error'

 999  continue
      end

c     -------------------------------------------------------------



      real function rfmt(ya)
c     --- rfmt: return the real as stored in string ya
      character*(*)ya
      character*20 yo
      write(yo,'(''(F'',I3.3,''.0)'')')lnblnk(ya)
      read(ya,fmt=yo)rfmt
      end
```



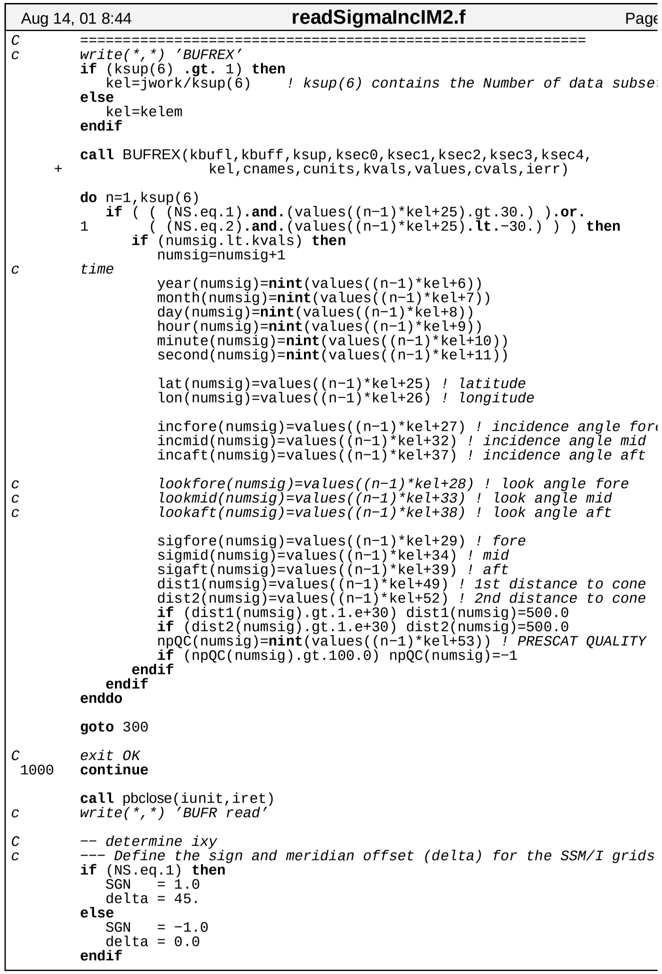
Aug 14, 01 8:44 **readSigmaIncIM2.f** Page

```
        subroutine readSigmaIncIM2(inname,NS,
        1    lat,lon,sigfore,sigmid,sigaft,
        2    incfore,incmid,incaft,
c       3    lookfore,lookmid,lookaft,
        4    dist1,dist2,npQC,numsig,ii,kk,
        5    year,month,day,hour,minute,second)
        implicit none
C
C       read sigma's etc from file <inname>
C
        integer kvals
        parameter (kvals=80000)

C       --- input ---
        character*80 inname     ! input filename
        integer NS              ! NS=1 => North Pole | NS=2 => South Pole
C       --- output ---
        real lat(kvals)         ! latitude
        real lon(kvals)         ! longitude
        real sigfore(kvals)     ! sigma forebeam
        real sigmid(kvals)      ! sigma midbeam
        real sigaft(kvals)      ! sigma aftbeam

        real incfore(kvals)     ! incidence angle fore
        real incmid(kvals)      ! incidence angle mid
        real incaft(kvals)      ! incidence angle aft

c        real lookfore(kvals)   ! look angle fore
c        real lookmid(kvals)    ! look angle mid
c        real lookaft(kvals)    ! look angle aft

        real dist1(kvals)       ! 1st distance to cone
        real dist2(kvals)       ! 2nd distance to cone
        integer ii(kvals)       ! index of corresponding ice mask point
        integer kk(kvals)       ! index of corresponding ice mask point
        integer npQC(kvals)     !
C       from : ~driesena/pp_work/source/INVERT.f
C       Inversion only performed if rVal(npQC,i) ne rMDI and
C       rVal(npFb,i) ne rMDI
C       1.            : Distance to cone too big, wind untouched
C       2.            : Low speed , bogus winds as solution
C       4.            : 1 solution, low speed , bogus 2nd solution added
C       8.            : 1 solution, high speed, no wind solution
C       16.           : High speed, no wind solution
        integer year(kvals)
        integer month(kvals)
        integer day(kvals)
        integer hour(kvals)
        integer minute(kvals)
        integer second(kvals)

C       --- input/output ---
        integer numsig          ! last recorded sigma's

C       --- local variables ---
        integer jwork ,jbufl ,kelem, jbyte
        integer jsup ,jsec0 ,jsec1 ,jsec2 ,jsec3
        integer jsec4 ,jkey ,jelem
        parameter       (jwork=80000, jbufl=20000, kelem = 20000, jbyte=800
        parameter       (jsup=9, jsec0=3, jsec1=40, jsec2=64, jsec3=4)
        parameter       (jsec4=2, jkey=46, jelem=20000)
        integer kbuff(jbufl)
        integer ksup(jsup)
        integer         ksec0(jsec0),ksec1(jsec1),ksec2(jsec2),ksec3(jsec3)
        integer         ksec4(jsec4),itdexp(jelem),itdlst(jelem)
```



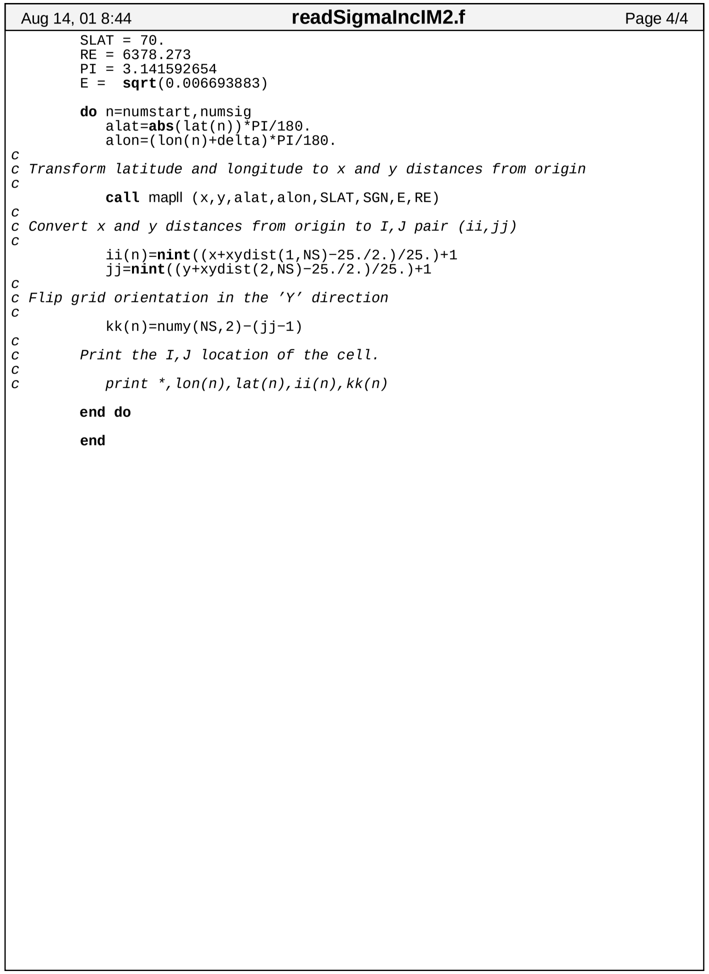
Aug 14, 01 8:44 **readSigmaIncIM2.f** Page 2/4

```
        real            values(kvals)

        integer k
        integer iunit

        integer ierr ,kbufl ,kerr ,kel ,n
        real rmdi

        character*64    cnames(kelem)
        character*24    cunits(kelem)
        character*80    cvals(kvals)

        integer iret

c       -- local variables for grid positioning
        integer numstart
        real SLAT,E,RE,PI
        real alat,alon,x,y
        real lat, lon
        real SGN, delta
        integer jj
        integer numy(2,3)
        real xydist(2,2)
        data numy / 896, 664, 448, 332, 224, 166 /
        data xydist   / 3850.0 , 5350.0 , 3950.0 , 3950.0 /

c       --- code ---
        iret=0
        numstart=numsig+1

C       missing data indicator
        rmdi=1.7E+38

        call pbopen(iunit,inname,'r',iret)
        if (iret .eq. -1) STOP 'open failed'
        if (iret .eq. -2) STOP 'invalid file name'
        if (iret .eq. -3) STOP 'invalide open mode specified'

C ----------------- Read BUFR message -------------------------
C
 300    CONTINUE
C
        ierr=0
        kbufl=0
C
        iret=0


        call PBBUFR(iunit,kbuff,jbyte,kbufl,iret)
C       possible end
        if (iret .eq. -1) goto 1000
        if (iret .eq. -2) STOP 'file handling problem'
        if (iret .eq. -3) STOP 'Array too small for product'
C
C ----------------- Expand BUFR message -------------------------
C
 400    CONTINUE
C
C       ---- Expand section 0/1/2 ----
        call BUS012(kbufl,kbuff,ksup,ksec0,ksec1,ksec2,kerr)
        if (kerr .ne. 0) then
           write(*,*) 'Error in BUS012: ',kerr
        endif
```



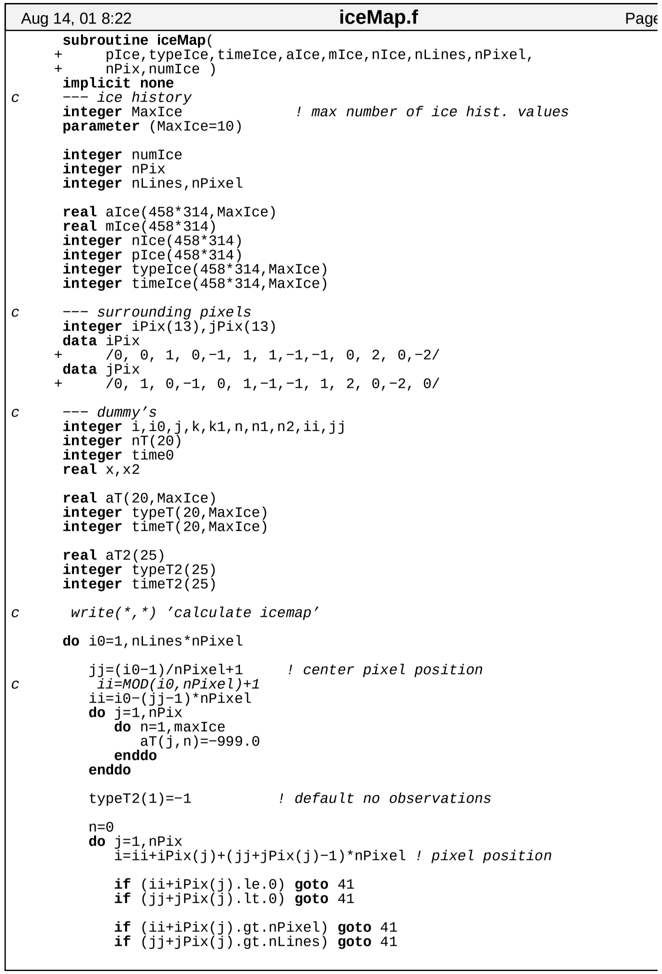
Aug 14, 01 8:44 **readSigmaIncIM2.f** Page

```
C       ===========================================================
c       write(*,*) 'BUFREX'
        if (ksup(6) .gt. 1) then
           kel=jwork/ksup(6)    ! ksup(6) contains the Number of data subse
        else
           kel=kelem
        endif

        call BUFREX(kbufl,kbuff,ksup,ksec0,ksec1,ksec2,ksec3,ksec4,
     +               kel,cnames,cunits,kvals,values,cvals,ierr)

        do n=1,ksup(6)
           if ( ( (NS.eq.1).and.(values((n-1)*kel+25).gt.30.) ).or.
     1        ( (NS.eq.2).and.(values((n-1)*kel+25).lt.-30.) ) ) then
              if (numsig.lt.kvals) then
                 numsig=numsig+1
c       time
                 year(numsig)=nint(values((n-1)*kel+6))
                 month(numsig)=nint(values((n-1)*kel+7))
                 day(numsig)=nint(values((n-1)*kel+8))
                 hour(numsig)=nint(values((n-1)*kel+9))
                 minute(numsig)=nint(values((n-1)*kel+10))
                 second(numsig)=nint(values((n-1)*kel+11))

                 lat(numsig)=values((n-1)*kel+25) ! latitude
                 lon(numsig)=values((n-1)*kel+26) ! longitude

                 incfore(numsig)=values((n-1)*kel+27) ! incidence angle for
                 incmid(numsig)=values((n-1)*kel+32) ! incidence angle mid
                 incaft(numsig)=values((n-1)*kel+37) ! incidence angle aft

c                lookfore(numsig)=values((n-1)*kel+28) ! look angle fore
c                lookmid(numsig)=values((n-1)*kel+33) ! look angle mid
c                lookaft(numsig)=values((n-1)*kel+38) ! look angle aft

                 sigfore(numsig)=values((n-1)*kel+29) ! fore
                 sigmid(numsig)=values((n-1)*kel+34) ! mid
                 sigaft(numsig)=values((n-1)*kel+39) ! aft
                 dist1(numsig)=values((n-1)*kel+49) ! 1st distance to cone
                 dist2(numsig)=values((n-1)*kel+52) ! 2nd distance to cone
                 if (dist1(numsig).gt.1.e+30) dist1(numsig)=500.0
                 if (dist2(numsig).gt.1.e+30) dist2(numsig)=500.0
                 npQC(numsig)=nint(values((n-1)*kel+53)) ! PRESCAT QUALITY
                 if (npQC(numsig).gt.100.0) npQC(numsig)=-1
              endif
           endif
        enddo

        goto 300

C       exit OK
 1000   continue

        call pbclose(iunit,iret)
c       write(*,*) 'BUFR read'

C       -- determine ixy
c       --- Define the sign and meridian offset (delta) for the SSM/I grids
        if (NS.eq.1) then
           SGN   = 1.0
           delta = 45.
        else
           SGN   = -1.0
           delta = 0.0
        endif
```



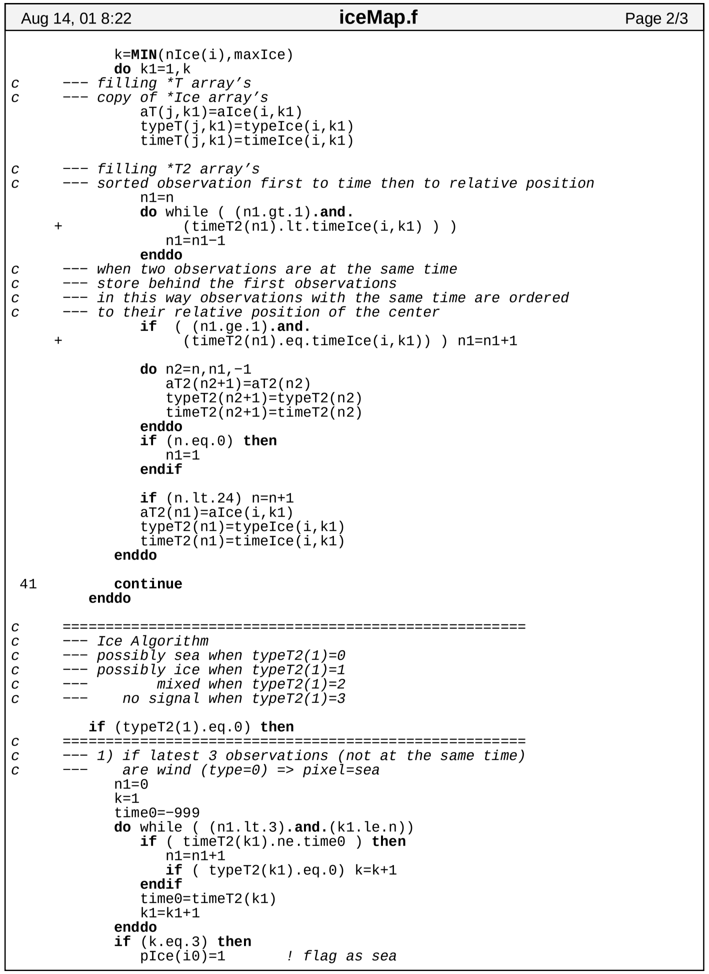
Aug 14, 01 8:44 **readSigmaIncIM2.f** Page 4/4

```
        SLAT = 70.
        RE = 6378.273
        PI = 3.141592654
        E =  sqrt(0.006693883)

        do n=numstart,numsig
           alat=abs(lat(n))*PI/180.
           alon=(lon(n)+delta)*PI/180.
c
c Transform latitude and longitude to x and y distances from origin
c
           call mapll (x,y,alat,alon,SLAT,SGN,E,RE)
c
c Convert x and y distances from origin to I,J pair (ii,jj)
c
           ii(n)=nint((x+xydist(1,NS)-25./2.)/25.)+1
           jj=nint((y+xydist(2,NS)-25./2.)/25.)+1
c
c Flip grid orientation in the 'Y' direction
c
           kk(n)=numy(NS,2)-(jj-1)
c
c       Print the I,J location of the cell.
c
c          print *,lon(n),lat(n),ii(n),kk(n)

        end do

        end
```



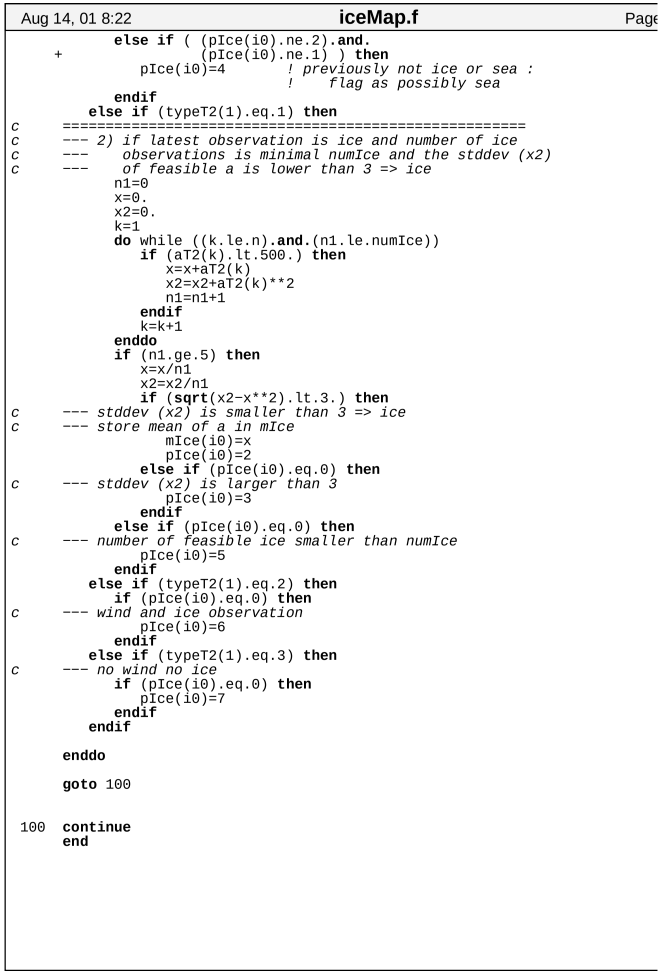
Aug 14, 01 8:22 **iceMap.f** Page

```
      subroutine iceMap(
     +     pIce,typeIce,timeIce,aIce,mIce,nIce,nLines,nPixel,
     +     nPix,numIce )
      implicit none
c     --- ice history
      integer MaxIce             ! max number of ice hist. values
      parameter (MaxIce=10)

      integer numIce
      integer nPix
      integer nLines,nPixel

      real aIce(458*314,MaxIce)
      real mIce(458*314)
      integer nIce(458*314)
      integer pIce(458*314)
      integer typeIce(458*314,MaxIce)
      integer timeIce(458*314,MaxIce)

c     --- surrounding pixels
      integer iPix(13),jPix(13)
      data iPix
     +     /0, 0, 1, 0,-1, 1, 1,-1,-1, 0, 2, 0,-2/
      data jPix
     +     /0, 1, 0,-1, 0, 1,-1,-1, 1, 2, 0,-2, 0/

c     --- dummy's
      integer i,i0,j,k,k1,n,n1,n2,ii,jj
      integer nT(20)
      integer time0
      real x,x2

      real aT(20,MaxIce)
      integer typeT(20,MaxIce)
      integer timeT(20,MaxIce)

      real aT2(25)
      integer typeT2(25)
      integer timeT2(25)

c      write(*,*) 'calculate icemap'

      do i0=1,nLines*nPixel

         jj=(i0-1)/nPixel+1     ! center pixel position
c         ii=MOD(i0,nPixel)+1
         ii=i0-(jj-1)*nPixel
         do j=1,nPix
            do n=1,maxIce
               aT(j,n)=-999.0
            enddo
         enddo

         typeT2(1)=-1          ! default no observations

         n=0
         do j=1,nPix
            i=ii+iPix(j)+(jj+jPix(j)-1)*nPixel ! pixel position

            if (ii+iPix(j).le.0) goto 41
            if (jj+jPix(j).lt.0) goto 41

            if (ii+iPix(j).gt.nPixel) goto 41
            if (jj+jPix(j).gt.nLines) goto 41
```



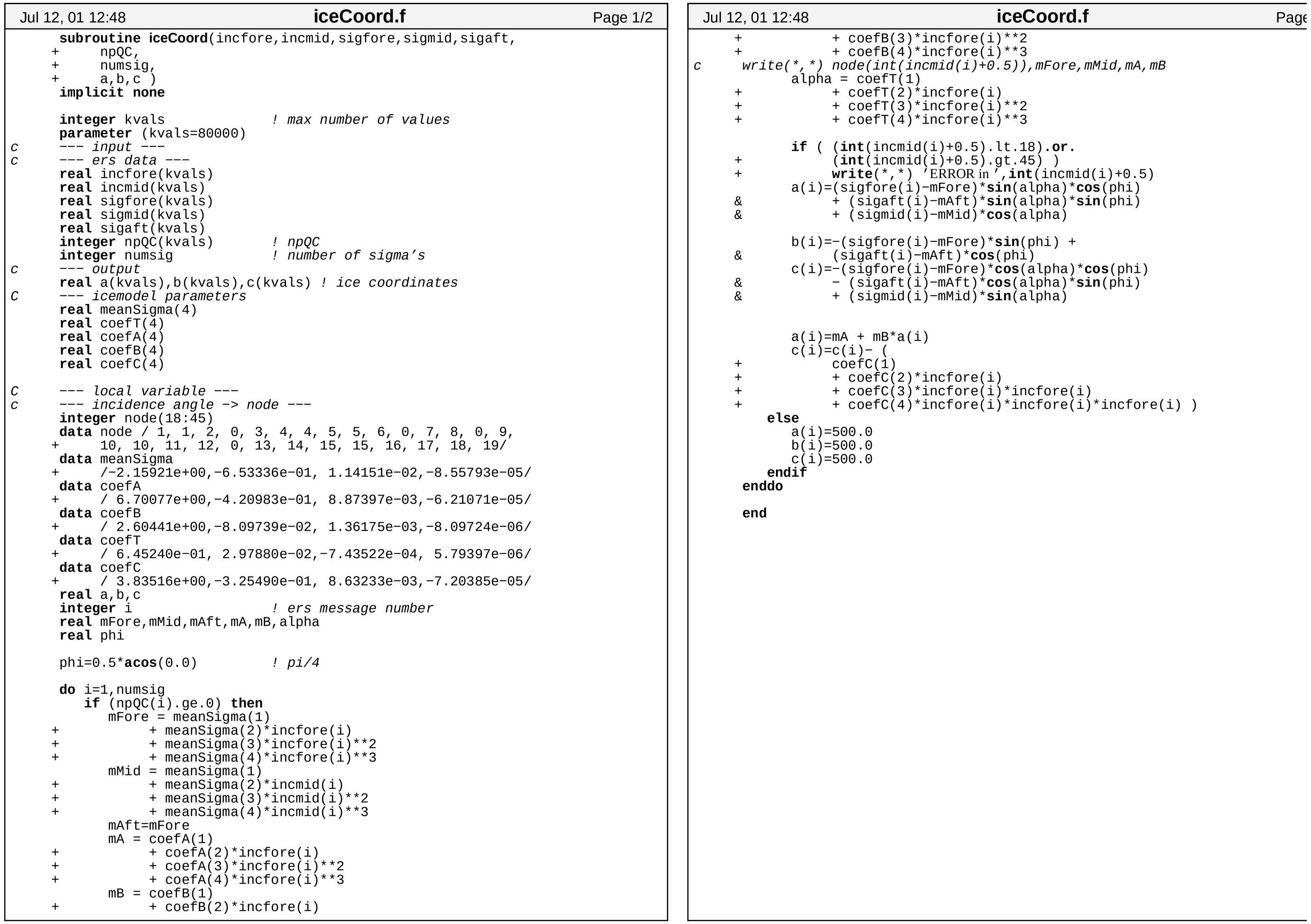
Aug 14, 01 8:22 **iceMap.f** Page 2/3

```
            k=MIN(nIce(i),maxIce)
            do k1=1,k
c     --- filling *T array's
c     --- copy of *Ice array's
               aT(j,k1)=aIce(i,k1)
               typeT(j,k1)=typeIce(i,k1)
               timeT(j,k1)=timeIce(i,k1)

c     --- filling *T2 array's
c     --- sorted observation first to time then to relative position
               n1=n
               do while ( (n1.gt.1).and.
     +              (timeT2(n1).lt.timeIce(i,k1) ) )
                  n1=n1-1
               enddo
c     --- when two observations are at the same time
c     --- store behind the first observations
c     --- in this way observations with the same time are ordered
c     --- to their relative position of the center
               if  ( (n1.ge.1).and.
     +              (timeT2(n1).eq.timeIce(i,k1)) ) n1=n1+1

               do n2=n,n1,-1
                  aT2(n2+1)=aT2(n2)
                  typeT2(n2+1)=typeT2(n2)
                  timeT2(n2+1)=timeT2(n2)
               enddo
               if (n.eq.0) then
                  n1=1
               endif

               if (n.lt.24) n=n+1
               aT2(n1)=aIce(i,k1)
               typeT2(n1)=typeIce(i,k1)
               timeT2(n1)=timeIce(i,k1)
            enddo

 41         continue
         enddo

c     =====================================================
c     --- Ice Algorithm
c     --- possibly sea when typeT2(1)=0
c     --- possibly ice when typeT2(1)=1
c     ---       mixed when typeT2(1)=2
c     ---   no signal when typeT2(1)=3

         if (typeT2(1).eq.0) then
c     =====================================================
c     --- 1) if latest 3 observations (not at the same time)
c     ---   are wind (type=0) => pixel=sea
            n1=0
            k=1
            time0=-999
            do while ( (n1.lt.3).and.(k1.le.n))
               if ( timeT2(k1).ne.time0 ) then
                  n1=n1+1
                  if ( typeT2(k1).eq.0) k=k+1
               endif
               time0=timeT2(k1)
               k1=k1+1
            enddo
            if (k.eq.3) then
               pIce(i0)=1       ! flag as sea
```



Aug 14, 01 8:22 **iceMap.f** Page

```
            else if ( (pIce(i0).ne.2).and.
     +                (pIce(i0).ne.1) ) then
               pIce(i0)=4       ! previously not ice or sea :
                                !    flag as possibly sea
            endif
         else if (typeT2(1).eq.1) then
c     =====================================================
c     --- 2) if latest observation is ice and number of ice
c     ---    observations is minimal numIce and the stddev (x2)
c     ---    of feasible a is lower than 3 => ice
            n1=0
            x=0.
            x2=0.
            k=1
            do while ((k.le.n).and.(n1.le.numIce))
               if (aT2(k).lt.500.) then
                  x=x+aT2(k)
                  x2=x2+aT2(k)**2
                  n1=n1+1
               endif
               k=k+1
            enddo
            if (n1.ge.5) then
               x=x/n1
               x2=x2/n1
               if (sqrt(x2-x**2).lt.3.) then
c     --- stddev (x2) is smaller than 3 => ice
c     --- store mean of a in mIce
                  mIce(i0)=x
                  pIce(i0)=2
               else if (pIce(i0).eq.0) then
c     --- stddev (x2) is larger than 3
                  pIce(i0)=3
               endif
            else if (pIce(i0).eq.0) then
c     --- number of feasible ice smaller than numIce
               pIce(i0)=5
            endif
         else if (typeT2(1).eq.2) then
            if (pIce(i0).eq.0) then
c     --- wind and ice observation
               pIce(i0)=6
            endif
         else if (typeT2(1).eq.3) then
c     --- no wind no ice
            if (pIce(i0).eq.0) then
               pIce(i0)=7
            endif
         endif

      enddo

      goto 100


 100  continue
      end
```

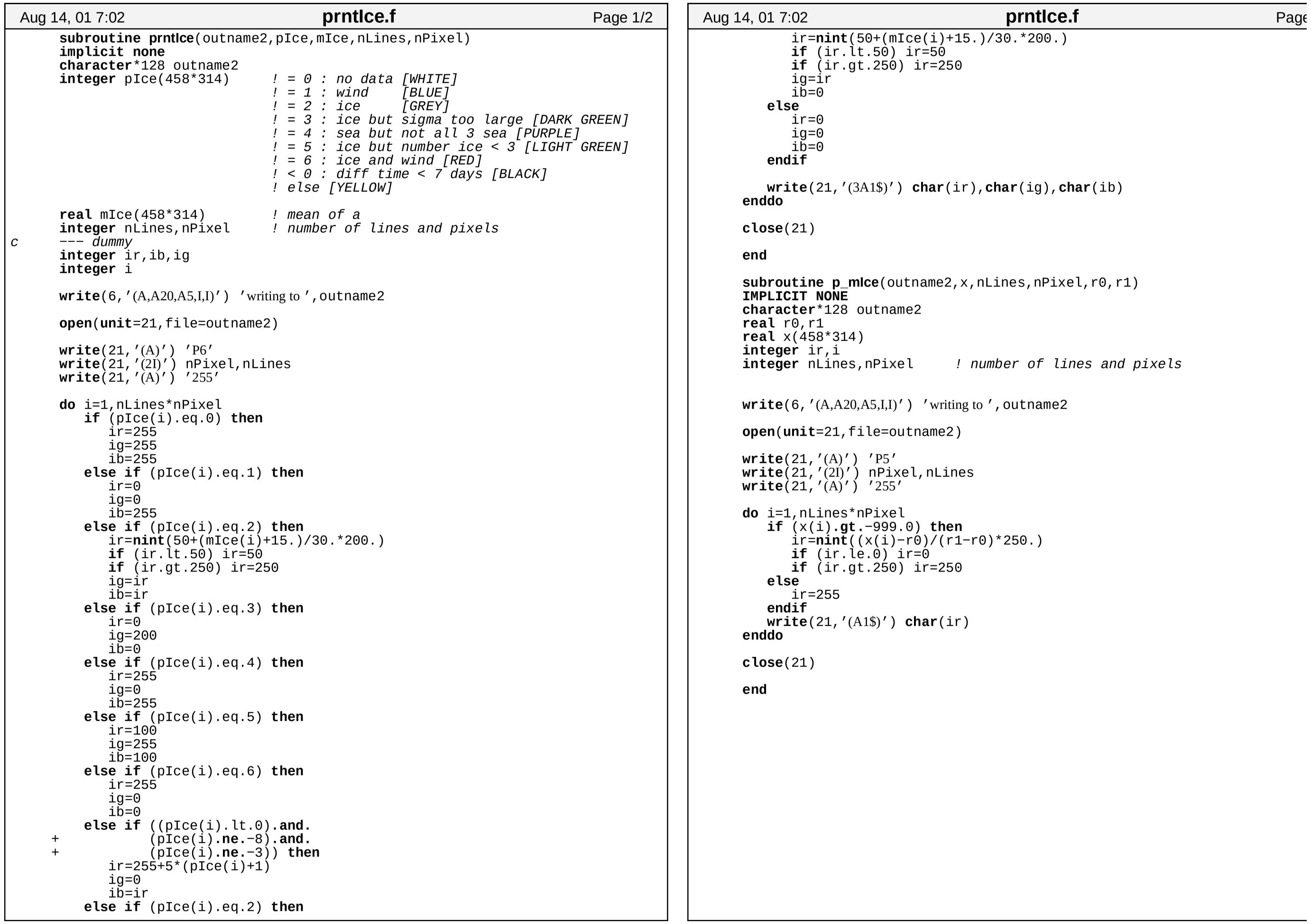


Jul 12, 01 12:48 **iceCoord.f** Page 1/2

```fortran
      subroutine iceCoord(incfore,incmid,sigfore,sigmid,sigaft,
     +     npQC,
     +     numsig,
     +     a,b,c )
      implicit none

      integer kvals             ! max number of values
      parameter (kvals=80000)
c     --- input ---
c     --- ers data ---
      real incfore(kvals)
      real incmid(kvals)
      real sigfore(kvals)
      real sigmid(kvals)
      real sigaft(kvals)
      integer npQC(kvals)       ! npQC
      integer numsig            ! number of sigma’s
c     --- output
      real a(kvals),b(kvals),c(kvals) ! ice coordinates
C     --- icemodel parameters
      real meanSigma(4)
      real coefT(4)
      real coefA(4)
      real coefB(4)
      real coefC(4)

C     --- local variable ---
c     --- incidence angle -> node ---
      integer node(18:45)
      data node / 1, 1, 2, 0, 3, 4, 4, 5, 5, 6, 0, 7, 8, 0, 9,
     +     10, 10, 11, 12, 0, 13, 14, 15, 15, 16, 17, 18, 19/
      data meanSigma
     +     /−2.15921e+00,−6.53336e−01, 1.14151e−02,−8.55793e−05/
      data coefA
     +     / 6.70077e+00,−4.20983e−01, 8.87397e−03,−6.21071e−05/
      data coefB
     +     / 2.60441e+00,−8.09739e−02, 1.36175e−03,−8.09724e−06/
      data coefT
     +     / 6.45240e−01, 2.97880e−02,−7.43522e−04, 5.79397e−06/
      data coefC
     +     / 3.83516e+00,−3.25490e−01, 8.63233e−03,−7.20385e−05/
      real a,b,c
      integer i                 ! ers message number
      real mFore,mMid,mAft,mA,mB,alpha
      real phi

      phi=0.5*acos(0.0)         ! pi/4

      do i=1,numsig
         if (npQC(i).ge.0) then
            mFore = meanSigma(1)
     +           + meanSigma(2)*incfore(i)
     +           + meanSigma(3)*incfore(i)**2
     +           + meanSigma(4)*incfore(i)**3
            mMid = meanSigma(1)
     +           + meanSigma(2)*incmid(i)
     +           + meanSigma(3)*incmid(i)**2
     +           + meanSigma(4)*incmid(i)**3
            mAft=mFore
            mA = coefA(1)
     +           + coefA(2)*incfore(i)
     +           + coefA(3)*incfore(i)**2
     +           + coefA(4)*incfore(i)**3
            mB = coefB(1)
     +           + coefB(2)*incfore(i)
```

Jul 12, 01 12:48 **iceCoord.f** Page

```fortran
     +           + coefB(3)*incfore(i)**2
     +           + coefB(4)*incfore(i)**3
c     write(*,*) node(int(incmid(i)+0.5)),mFore,mMid,mA,mB
            alpha = coefT(1)
     +           + coefT(2)*incfore(i)
     +           + coefT(3)*incfore(i)**2
     +           + coefT(4)*incfore(i)**3

            if ( (int(incmid(i)+0.5).lt.18).or.
     +           (int(incmid(i)+0.5).gt.45) )
     +           write(*,*) ’ERROR in ’,int(incmid(i)+0.5)
            a(i)=(sigfore(i)−mFore)*sin(alpha)*cos(phi)
     &           + (sigaft(i)−mAft)*sin(alpha)*sin(phi)
     &           + (sigmid(i)−mMid)*cos(alpha)

            b(i)=−(sigfore(i)−mFore)*sin(phi) +
     &           (sigaft(i)−mAft)*cos(phi)
            c(i)=−(sigfore(i)−mFore)*cos(alpha)*cos(phi)
     &           − (sigaft(i)−mAft)*cos(alpha)*sin(phi)
     &           + (sigmid(i)−mMid)*sin(alpha)


            a(i)=mA + mB*a(i)
            c(i)=c(i)− (
     +           coefC(1)
     +           + coefC(2)*incfore(i)
     +           + coefC(3)*incfore(i)*incfore(i)
     +           + coefC(4)*incfore(i)*incfore(i)*incfore(i) )
         else
            a(i)=500.0
            b(i)=500.0
            c(i)=500.0
         endif
      enddo

      end
```



Aug 14, 01 7:02 **prntIce.f** Page 1/2

```
      subroutine prntIce(outname2,pIce,mIce,nLines,nPixel)
      implicit none
      character*128 outname2
      integer pIce(458*314)     ! = 0 : no data [WHITE]
                                ! = 1 : wind    [BLUE]
                                ! = 2 : ice     [GREY]
                                ! = 3 : ice but sigma too large [DARK GREEN]
                                ! = 4 : sea but not all 3 sea [PURPLE]
                                ! = 5 : ice but number ice < 3 [LIGHT GREEN]
                                ! = 6 : ice and wind [RED]
                                ! < 0 : diff time < 7 days [BLACK]
                                ! else [YELLOW]

      real mIce(458*314)        ! mean of a
      integer nLines,nPixel     ! number of lines and pixels
c     --- dummy
      integer ir,ib,ig
      integer i

      write(6,'(A,A20,A5,I,I)') 'writing to ',outname2

      open(unit=21,file=outname2)

      write(21,'(A)') 'P6'
      write(21,'(2I)') nPixel,nLines
      write(21,'(A)') '255'

      do i=1,nLines*nPixel
         if (pIce(i).eq.0) then
            ir=255
            ig=255
            ib=255
         else if (pIce(i).eq.1) then
            ir=0
            ig=0
            ib=255
         else if (pIce(i).eq.2) then
            ir=nint(50+(mIce(i)+15.)/30.*200.)
            if (ir.lt.50) ir=50
            if (ir.gt.250) ir=250
            ig=ir
            ib=ir
         else if (pIce(i).eq.3) then
            ir=0
            ig=200
            ib=0
         else if (pIce(i).eq.4) then
            ir=255
            ig=0
            ib=255
         else if (pIce(i).eq.5) then
            ir=100
            ig=255
            ib=100
         else if (pIce(i).eq.6) then
            ir=255
            ig=0
            ib=0
         else if ((pIce(i).lt.0).and.
     +            (pIce(i).ne.-8).and.
     +            (pIce(i).ne.-3)) then
            ir=255+5*(pIce(i)+1)
            ig=0
            ib=ir
         else if (pIce(i).eq.2) then
```

Aug 14, 01 7:02 **prntIce.f** Page

```
            ir=nint(50+(mIce(i)+15.)/30.*200.)
            if (ir.lt.50) ir=50
            if (ir.gt.250) ir=250
            ig=ir
            ib=0
         else
            ir=0
            ig=0
            ib=0
         endif

         write(21,'(3A1$)') char(ir),char(ig),char(ib)
      enddo

      close(21)

      end

      subroutine p_mIce(outname2,x,nLines,nPixel,r0,r1)
      IMPLICIT NONE
      character*128 outname2
      real r0,r1
      real x(458*314)
      integer ir,i
      integer nLines,nPixel     ! number of lines and pixels


      write(6,'(A,A20,A5,I,I)') 'writing to ',outname2

      open(unit=21,file=outname2)

      write(21,'(A)') 'P5'
      write(21,'(2I)') nPixel,nLines
      write(21,'(A)') '255'

      do i=1,nLines*nPixel
         if (x(i).gt.-999.0) then
            ir=nint((x(i)-r0)/(r1-r0)*250.)
            if (ir.le.0) ir=0
            if (ir.gt.250) ir=250
         else
            ir=255
         endif
         write(21,'(A1$)') char(ir)
      enddo

      close(21)

      end
```