Technical Report

Location of observation points on a grid and Test of different interpolation schemes

Nicolas Daget

Rapport Technique TR/CMGC/03/79
CERFACS - GLOBC

Contents

Chapter 1

Introduction

Variational assimilation systems need to estimate the ocean state vector in the observation points.
There is also an observation operator H_i at t_i such as $H_i(\mathbf{w}(t_i)) = G_i(\mathbf{w}(t_0))$ with G_i which is in the cost function J.

$$J(\delta\mathbf{w}) = \frac{1}{2}\delta\mathbf{w}^T\mathbf{B}^{-1}\delta\mathbf{w} + \frac{1}{2}\sum_{i=0}^{n}\left[\mathbf{G}_i\delta\mathbf{w} - \mathbf{d}_i\right]^T\mathbf{R}_i^{-1}\left[\mathbf{G}_i\delta\mathbf{w} - \mathbf{d}_i\right]$$

The first issue is to locate the cells which contain the observation points. As the ORCA grid is streched (figure 1.1), it is necessary to use a special algorithm.
The second issue is to do a good interpolation, despites the grid is streched (figure 1.1).
Differents interpolation methods were implemented in ORCAVAR which permit to choose the best operator H.

- Distance-weighted interpolation scheme 1 ;
- Distance-weighted interpolation scheme 2 ;
- Bilinear interpolation on a geographical grid ;
- General bilinear remapping interpolation ;
- Polynomial interpolation.

These methods will be presented and compared in this report.

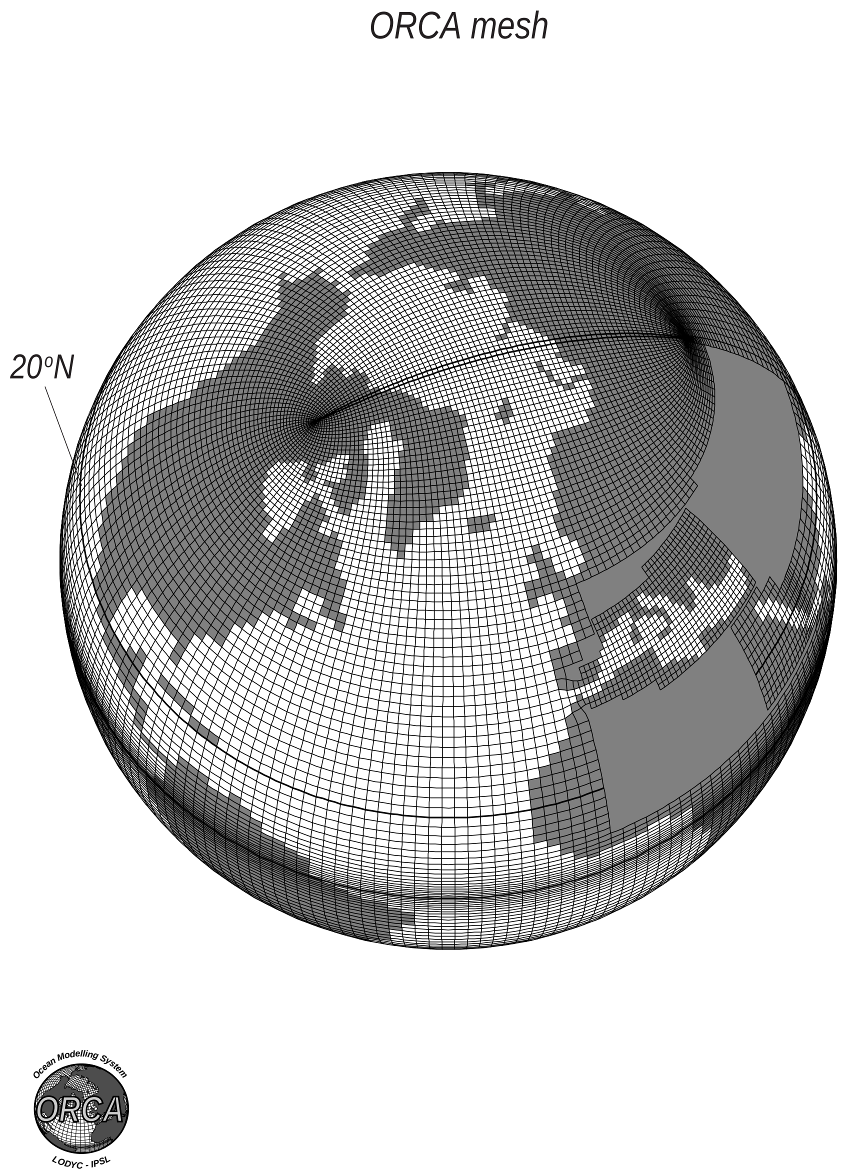

Figure 1.1: ORCA grid

Chapter 2

Location of observation points on the grid

2.1 Background

As the orca grid is so streched (figure 1.1), the indices are not monotonous functions of latitude and longitude and any cells are very big (like in Africa). So, the algorithm must be very robust (hardy).

2.2 Algorithm

2.2.1 Main algorithm

The main part of the algorithm can be decomposed in 3 parts.
To begin, we use a method wich looks like a dichotomy. Nevertheless, instead to cut the domain in two parts and choose one, the domain is cut in 10. Indeed, with so big cells, it is impossible to do a good job, because the cells are not always quadrangles.
Next, the domains is enlarged to avoid several minor problems and all cells of this domain are checked.
If the two first parts are failed, all cells of the grid are checked.

2.2.2 Optimization

The algorithm is not very fast. Nevertheless, the observation points follow them in many case (sattelite data, boat data...). To optimize drastically the algoritm, it need only to remenber the indices of the previous observation point and to check the cells around.

2.3 The test

The purpose is to determine whether a point (x,y) lies within or on the boundary of a quadrangle (figure 2.1) of any shape on a plane.

The solution is to check if the next vectorial products are all negative.

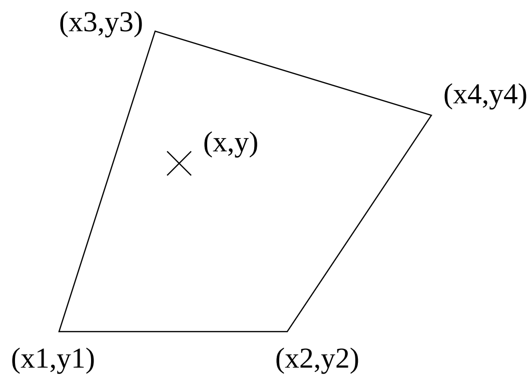

Figure 2.1: Nomenclature

$$\begin{cases} A = (x - x_1)(y - y_3) - (x - x_3)(y - y_1) \\ B = (x - x_3)(y - y_4) - (x - x_4)(y - y_3) \\ C = (x - x_4)(y - y_2) - (x - x_2)(y - y_4) \\ D = (x - x_2)(y - y_1) - (x - x_1)(y - y_2) \end{cases}$$

So, the test is below :

$$(A \times B \geq 0) \text{ AND } (B \times C \geq 0) \text{ AND } (C \times D \geq 0) \text{ AND } (D\ timesA \geq 0)$$

2.4 360 degrees ambiguity

To avoid errors and bugs, it is very important to manage the 360 degrees ambiguity. Indeed, the longitude must be in the same reference. (For instance : [0:360] or [-180:180]). This is easy to get the longitude in the same reference. Nevertheless, there are several cases more complex. For example, when the observation point longitude is near 0 degree for a [0:360] degrees reference. In this case, the four points around the observation point are not contiguous. So, the longitude of two of them are near 0 and the longitude of the two others are near 360. To avoid this problem, it is necessary to control the shape of the cell. If lenght of the cell is much greater than the height, all the longitude are translated near an extremity of the the referece (near 360 for example). In our case, the test of the shape is below :

$$if \left(\frac{\lambda_{max} - \lambda_{min}}{\phi_{max} - \phi_{min}} > 10 \right) then$$

$$\forall \qquad \lambda < \frac{\lambda_{max}}{3} \qquad ; \qquad \lambda = \lambda + 360$$

Next, we check if the observation point is in the cell. If the test failed and if the observation point is near the other extremity of the reference (near 0 for example), the observation point is translated and an other test is made.

$$test$$

$$if \qquad (test = FALSE \qquad AND \qquad \lambda_{obs} < cellmax) \qquad then \qquad \lambda_{obs} = \lambda_{obs} + 360$$

$$test$$

Chapter 3

Presentation of the different interpolation schemes

3.1 Distance-weighted interpolation scheme 1

The main idea of this scheme is to calculate the distances on the sphere between the observation point and the four others points of the cell.
The weight of each cell's point is then the product of the distances between the observation point and the three others cell's points.
To calculate the distance on a sphere, the formula below is used :

$$s(AB) = arccos(sin(\phi_A)sin(\phi_B) + cos(\phi_A)cos(\phi_B)cos(\lambda_B - \lambda_A))$$

The weight (p_i) of each cell's angle is then :

$$p_i = \prod_{j \neq i} s_j$$

3.2 Distance-weighted interpolation scheme 2

The first Distance-weighted interpolation scheme is interesting, nevertheless the arccos (or the arcsin if we use the formula $arcos(x) = arcsin(\sqrt{1 - x^2})$) is hard to compute.
So, the second Distance-weighted interpolation scheme use a small-angle approximation.
Then :

$$ds(AB) = \sqrt{(\phi_B - \phi_A)^2 + ((\lambda_B - \lambda_A)cos(\phi_B))^2}$$

3.3 Bilinear interpolation on a geographical grid

The interpolation is split into two 1D interpolations in the longitude and latitude directions, respectively.

3.4 General bilinear remapping interpolation

This is an iterative scheme that involves first mapping a quadrilateral cell into a cell with coordinates (0,0), (1,0), (0,1) and (1,1). Let the latitude-longitude coordinates of point 1 be $(\theta(i,j), \phi(i,j))$, the coordinates of point 2 be $(\theta(i+1,j), \phi(i+1,j))$, etc. Now let α and β be continuous local coordinates such that the coordinates (α, β) of point 1 are (0,0), point 2 are (1,0), point 3 are (1,1) and point 4 are (0,1). If point P lies inside the cell formed by the four points above, the function f at point P can be approximated by

$$\begin{aligned} f_P &= (1-\alpha)(1-\beta)f(i,j) + \alpha(1-\beta)f(i+1,j) + \alpha\beta f(i+1,j+1) + (1-\alpha)\beta f(i,j+1) \\ &= w_1 f(i,j) + w_2 f(i+1,j) + w_3 f(i+1,j+1) + w_4 f(i,j+1) \end{aligned} \tag{3.1}$$

The remapping weights must therefore be computed by finding α and β at point P.
The latitude-longitude coordinates (θ, ϕ) of point P are known and can also be approximated by

$$\begin{aligned} \theta &= (1-\alpha)(1-\beta)\theta_1 + \alpha(1-\beta)\theta_2 + \alpha\beta\theta_3 + (1-\alpha)\beta\theta_4 \\ \phi &= (1-\alpha)(1-\beta)\phi_1 + \alpha(1-\beta)\phi_2 + \alpha\beta\phi_3 + (1-\alpha)\beta\phi_4 \end{aligned} \tag{3.2}$$

Because (3.2) is nonlinear in α and β, we must linearize and iterate towar a solution. Differentiating (3.2) results in

$$\begin{bmatrix} \delta\theta \\ \delta\phi \end{bmatrix} = A \begin{bmatrix} \delta\alpha \\ \delta\beta \end{bmatrix} \tag{3.3}$$

where

$$A = \begin{bmatrix} (\theta_2-\theta_1)+(\theta_1-\theta_4+\theta_3-\theta_2)\beta & (\theta_4-\theta_1)+(\theta_1-\theta_4+\theta_3-\theta_2)\alpha \\ (\phi_2-\phi_1)+(\phi_1-\phi_4+\phi_3-\phi_2)\beta & (\phi_4-\phi_1)+(\phi_1-\phi_4+\phi_3-\phi_2)\alpha \end{bmatrix} \tag{3.4}$$

Inverting this system,

$$\delta\alpha = \begin{vmatrix} \delta\theta & (\theta_4-\theta_1)+(\theta_1-\theta_4+\theta_3-\theta_2)\alpha \\ \delta\phi & (\phi_4-\phi_1)+(\phi_1-\phi_4+\phi_3-\phi_2)\alpha \end{vmatrix} \div det(A) \tag{3.5}$$

and

$$\delta\beta = \begin{vmatrix} (\theta_2-\theta_1)+(\theta_1-\theta_4+\theta_3-\theta_2)\beta & \delta\theta \\ (\phi_2-\phi_1)+(\phi_1-\phi_4+\phi_3-\phi_2)\beta & \delta\phi \end{vmatrix} \div det(A) \tag{3.6}$$

Starting with an initial guess for α and β (say $\alpha = \beta = 0$), equations (3.5) and (3.6) can be iterated until $\delta\alpha$ and $\delta\beta$ are suitably small. The weights can then be computed from (3.1). Note that for simple latitude-longitude grids, this iteration will converge in the first iteration.

3.5 Polynomial interpolation

This method permits to interpolate in any quadrangle with shape functions.
The interpolation of a point (x_1, x_2) use 4 nodes with the coordinates such as $m^{(i)} = (x_1^{(i)}, x_2^{(i)})$ with i

from 1 to 4.
The interpolation looks like

$$\widetilde{f}_r(x_1,x_2)=\sum_{i=1}^{4}u^{(i)}P^{(i)}(x_1,x_2)$$

where $P^{(i)}$ are 4 dimensions polynomes with 2 variables. Let $P^{(i)}=a_1^{(i)}+a_2^{(i)}x_1+a_3^{(i)}x_2+a_4^{(i)}x_1x_2$, we built a polinomes $P^{(i)}$ base.
The conditions are also

$$\begin{pmatrix} a_1^{(1)} & a_2^{(1)} & a_3^{(1)} & a_4^{(1)} \\ a_1^{(2)} & a_2^{(2)} & a_3^{(2)} & a_4^{(2)} \\ a_1^{(3)} & a_2^{(3)} & a_3^{(3)} & a_4^{(3)} \\ a_1^{(4)} & a_2^{(4)} & a_3^{(4)} & a_4^{(4)} \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & 1 \\ x_1^{(1)} & x_1^{(2)} & x_1^{(3)} & x_1^{(4)} \\ x_2^{(1)} & x_2^{(2)} & x_2^{(3)} & x_2^{(4)} \\ x_1^{(1)}x_2^{(1)} & x_1^{(2)}x_2^{(2)} & x_1^{(3)}x_2^{(3)} & x_1^{(4)}x_2^{(4)} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The coefficients of the shape functions can be deduce by inversion

$$\begin{pmatrix} a_1^{(1)} & a_2^{(1)} & a_3^{(1)} & a_4^{(1)} \\ a_1^{(2)} & a_2^{(2)} & a_3^{(2)} & a_4^{(2)} \\ a_1^{(3)} & a_2^{(3)} & a_3^{(3)} & a_4^{(3)} \\ a_1^{(4)} & a_2^{(4)} & a_3^{(4)} & a_4^{(4)} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ x_1^{(1)} & x_1^{(2)} & x_1^{(3)} & x_1^{(4)} \\ x_2^{(1)} & x_2^{(2)} & x_2^{(3)} & x_2^{(4)} \\ x_1^{(1)}x_2^{(1)} & x_1^{(2)}x_2^{(2)} & x_1^{(3)}x_2^{(3)} & x_1^{(4)}x_2^{(4)} \end{pmatrix}^{-1}$$

The four shape functions are also

$$\begin{pmatrix} P^{(1)} \\ P^{(2)} \\ P^{(3)} \\ P^{(4)} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ x_1^{(1)} & x_1^{(2)} & x_1^{(3)} & x_1^{(4)} \\ x_2^{(1)} & x_2^{(2)} & x_2^{(3)} & x_2^{(4)} \\ x_1^{(1)}x_2^{(1)} & x_1^{(2)}x_2^{(2)} & x_1^{(3)}x_2^{(3)} & x_1^{(4)}x_2^{(4)} \end{pmatrix}^{-1} \begin{pmatrix} 1 \\ x_1 \\ x_2 \\ x_1x_2 \end{pmatrix}$$

And the interpolation is so

$$\widetilde{f}_r(x_1,x_2)=u^{(1)}P^{(1)}(x_1,x_2)+u^{(2)}P^{(2)}(x_1,x_2)+u^{(3)}P^{(3)}(x_1,x_2)+u^{(4)}P^{(4)}(x_1,x_2)$$

Chapter 4

Test of the different interpolation schemes

4.1 Methodology

To compare the different interpolation methods, an analytic field will be used. In our experiences, three fields are used (Figure 4.1, figure 4.2 and figure 4.3) :

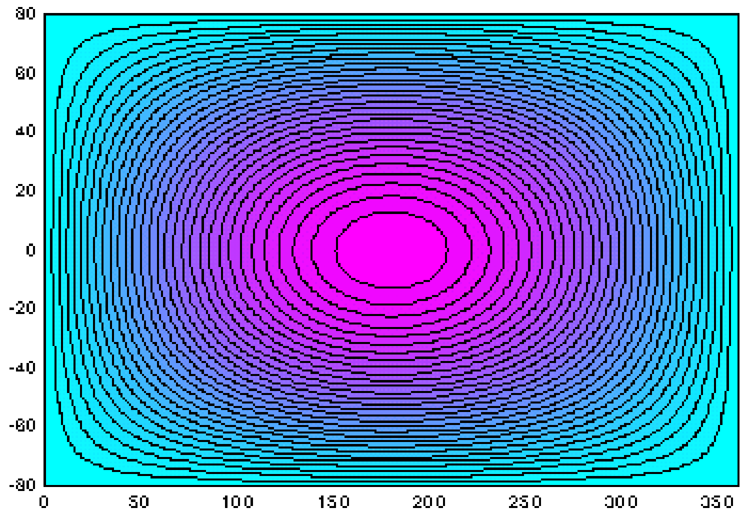

Figure 4.1: Field 1

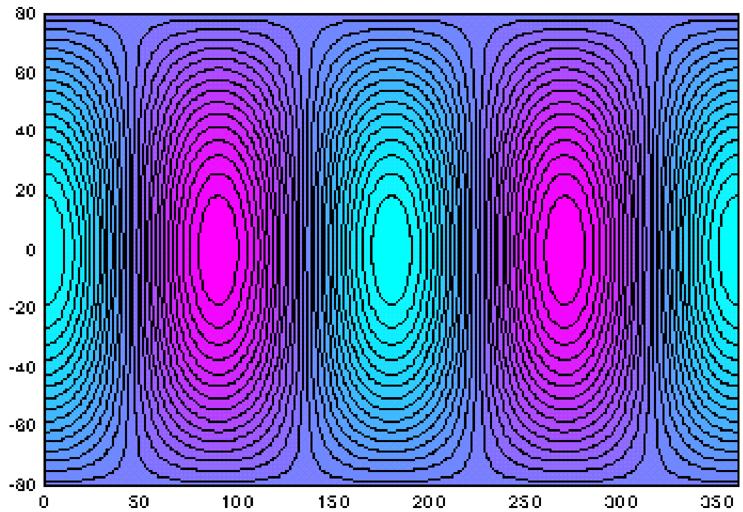

Figure 4.2: Field 2

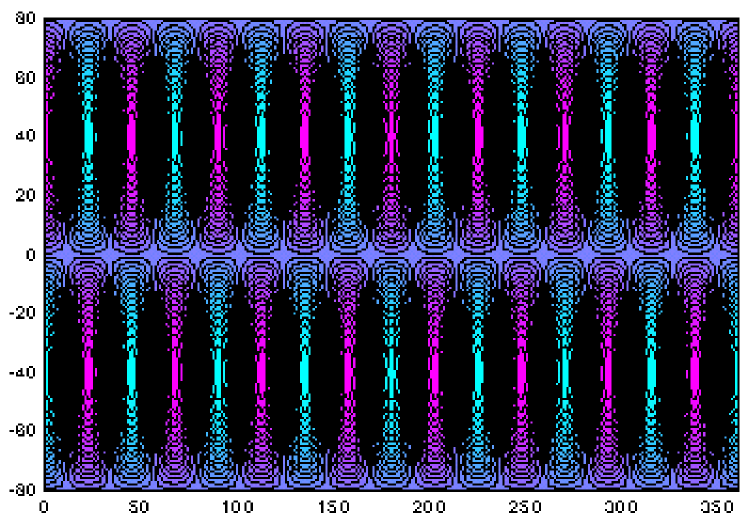

Figure 4.3: Field 3

4.2 Results

The interpolation is made with 100 000 random observation points. The results are summarized in the figures 4.1 and 4.2, where the relative error is given for the different interpolation method.

Interpolation method	Field 1	Field 2	Field 3
Distance-weighted interpolation scheme 1	6.24e-4	1.93e-3	7.80e-3
Distance-weighted interpolation scheme 2	8.05e-4	1.86e-3	5.8e-3
Bilinear interpolation on a geographical grid	1.92e-4	4.99e-4	2.94e-3
General bilinear remapping interpolation	4.88e-4	6.06e-4	7.12e-3
Polynomial interpolation	1.17e-4	4.21e-41	2.56e-3

Table 4.1: Results for the whole domain

Interpolation method	Field 1	Field 2	Field 3
Distance-weighted interpolation scheme 1	7.84e-4	1.73e-3	8.76e-3
Distance-weighted interpolation scheme 2	7.32e-4	2.32e-3	7.80e-3
Bilinear interpolation on a geographical grid	1.33e-4	1.26e-3	3.30e-3
General bilinear remapping interpolation	7.50e-4	4.74e-3	3.11e-2
Polynomial interpolation	1.14e-4	4.22e-4	3.17e-3

Table 4.2: Results for the north part of the domain (latitude between 40 and 80)

These results and the figures below show that the most important criterion is the north part of the domain where the grid is streched and where the results are so bad.
Nevertheless, the general bilinear remapping interpolation and the polynomial interpolation give good enought results. However, the general bilinear remapping interpolation is more interesting because it is a very fast method.
In another hand, the Bilinear interpolation on a geographical grid is very effecient where the grid is regular. The results are very good except in the north part of the domain. So, this method is very usefull for a regular grid.

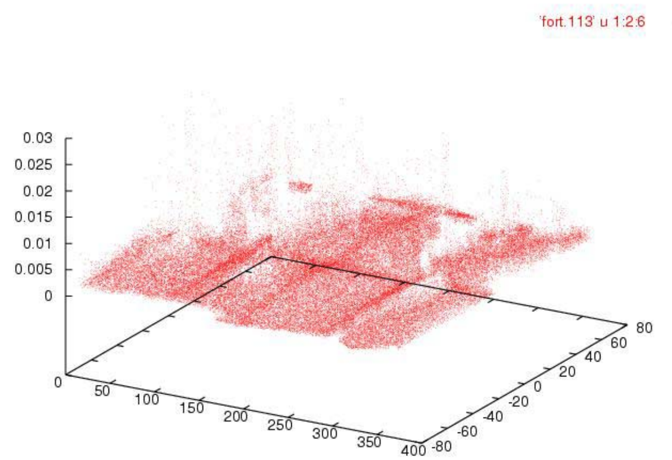

Figure 4.4: Distance-weighted interpolation scheme 1 on field 2

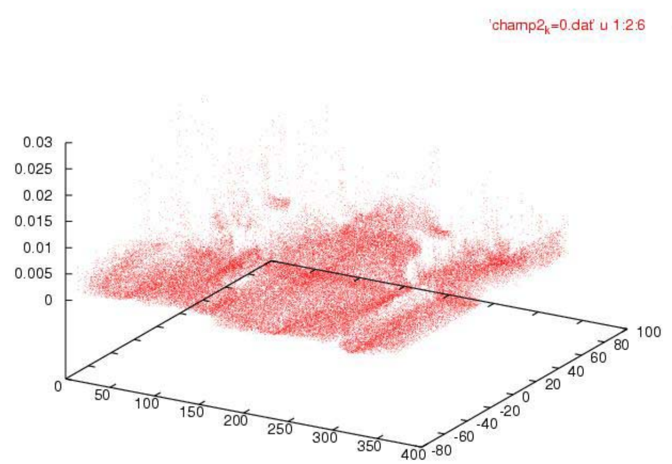

Figure 4.5: Distance-weighted interpolation scheme 2 on field 2

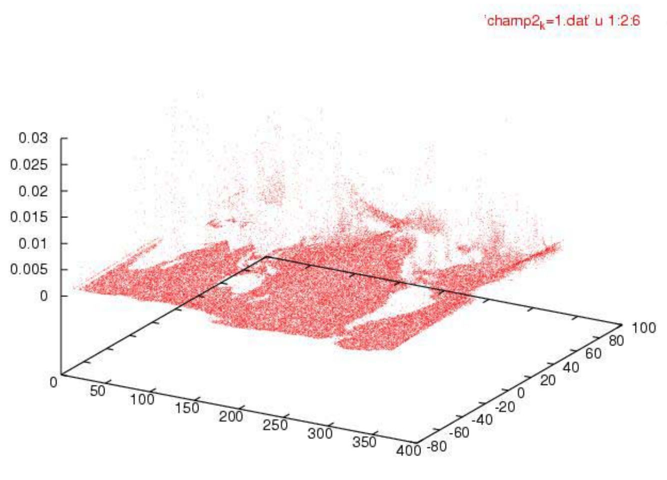

Figure 4.6: Bilinear interpolation on a geographical grid on field 2

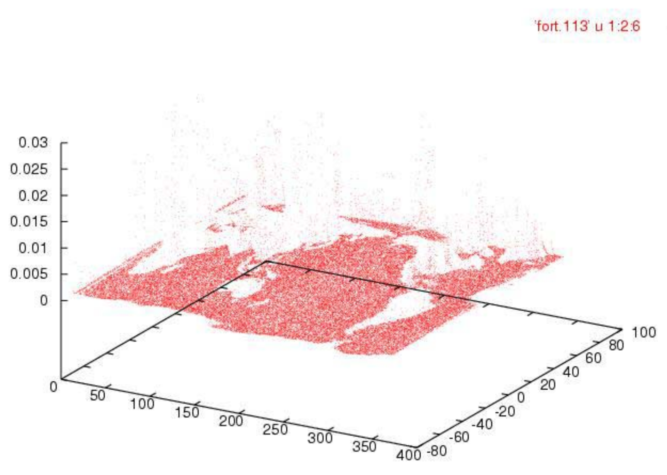

Figure 4.7: General bilinear remapping interpolation on field 2

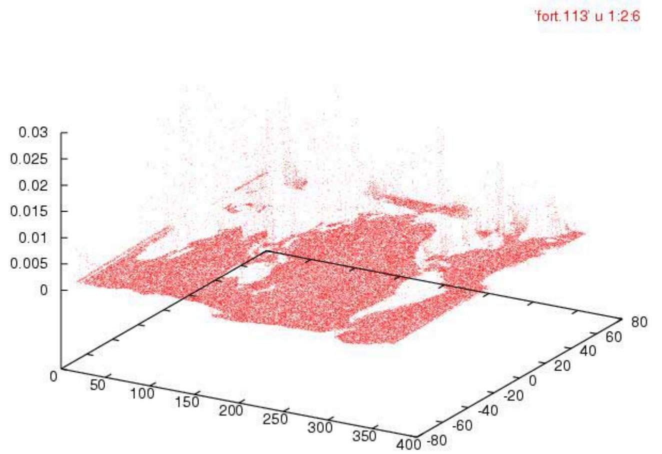

Figure 4.8: Polynomial interpolation on field 2

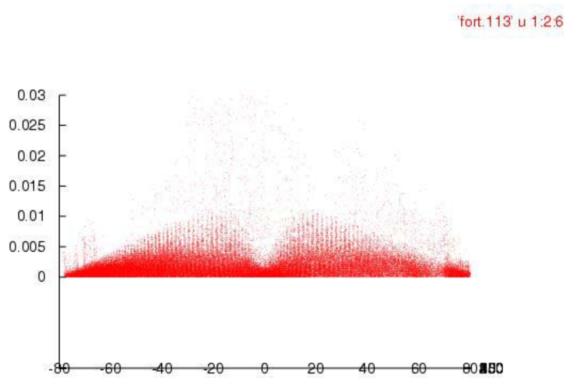

Figure 4.9: Distance-weighted interpolation scheme 1 on field 2 (lateral view)

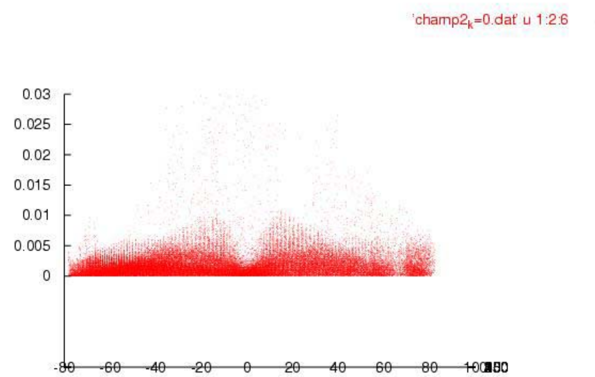

Figure 4.10: Distance-weighted interpolation scheme 2 on field 2 (lateral view)

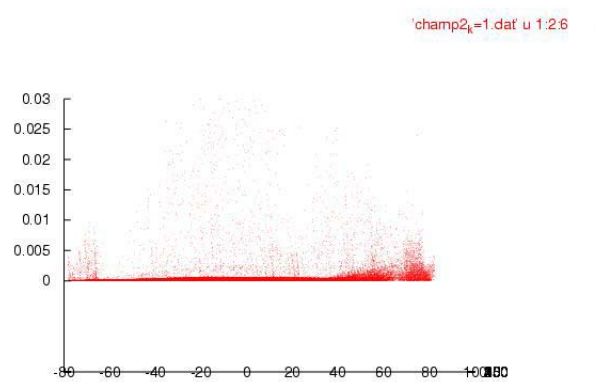

Figure 4.11: Bilinear interpolation on a geographical grid on field 2 (lateral view)

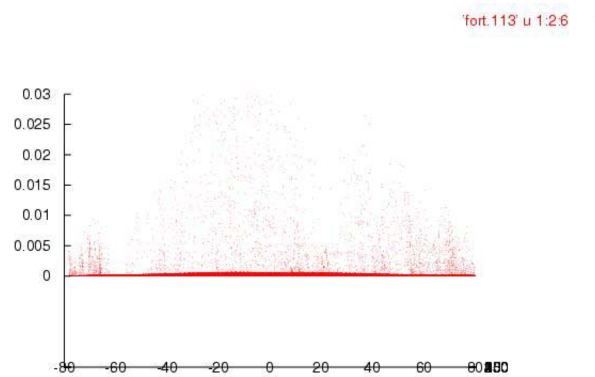

Figure 4.12: General bilinear remapping interpolation on field 2 (lateral view)

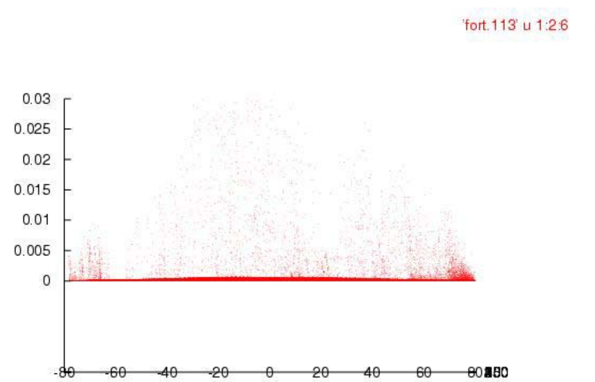

Figure 4.13: Polynomial interpolation on field 2 (lateral view)

Chapter 5

Conclusion

The location algorithm is very efficient and permit to find all the observation points tested. It is a good improvment because the old one has some problems when the grid is strched.
For the interpolation method, the best one is the general bilinear remapping interpolation. Indeed, it is a very powerful method, which is very accurate when the grid is streched and very fast. The rapidity of this algorithm is the main advantage compared to the polynomial interpolation.