Normalized Interpolated Convolution from an Adaptive Subgrid documentation

Benjamin Ménétrier

Last update: September 29, 2020

Contents

1 Introduction

The application of a localization matrix $\mathbf{C} \in \mathbb{R}^{n \times n}$ to a vector $\mathbf{x} \in \mathbb{R}^n$ requires n^2 multiplications and $n(n-1)$ additions, which can too expensive for large systems. If the localization functions inside $\mathbf{C}$ are compactly supported, then $\mathbf{C}$ has a lot of zeros and the cost is reduced. However in this case, the cost increases with the size of the localization functions support: the larger the support, the larger the number of points involved and therefore the cost.

The idea of the NICAS method is to apply an equivalent localization operator on a subgrid $\mathcal{G}^s$ that contains a subset of n^s points of the full grid $\mathcal{G}^f$, with $n^s \ll n$. We define the localization matrix $\mathbf{C}^s \in \mathbb{R}^{n^s \times n^s}$ and the linear interpolation operator $\mathbf{S} \in \mathbb{R}^{n \times n^s}$. The NICAS localization matrix is given by:

$$\widetilde{\mathbf{C}} = \mathbf{N}\mathbf{S}\mathbf{C}^s\mathbf{S}^{\mathrm{T}}\mathbf{N}^{\mathrm{T}} \tag{1}$$

where $\mathbf{N} \in \mathbb{R}^{n \times n}$ is a diagonal operator, which ensures that $\widetilde{\mathbf{C}}$ is normalized (i.e. $\widetilde{C}_{ii} = 1$). Indeed, even if $\mathbf{C}^s$ is normalized, $\mathbf{S}\mathbf{C}^s\mathbf{S}^{\mathrm{T}}$ is not normalized.

The ARPEGE model is the operational global model at Météo-France. Most illustrations are drawn from preliminary experiments that have been conducted with ARPEGE on a low-resolution grid (truncation TL149) with 106 levels. Scaling experiments also use a higher-resolution grid (truncation TL399).

2 Convolution subgrid

The choice of the subgrid is directly linked to the interpolation methods used in $\mathbf{S}$. Since a full 3D interpolation might be hard to compute and to apply in a geophysical model, we split $\mathbf{S}$ into three operators:

$$\mathbf{S} = \mathbf{S}^h\mathbf{S}^v\mathbf{S}^s \tag{2}$$

To describe $\mathbf{S}^h$, $\mathbf{S}^v$ and $\mathbf{S}^s$, it is easier to look at the grids they are interpolating to:

- $\mathcal{G}^f$ is the full grid, with a set $\mathcal{S}_0^c$ of n_0^c horizontal points and a set $\mathcal{S}_0^l$ of n_0^l levels,
- $\mathcal{G}^h$ is the first intermediate subgrid, with a subset $\mathcal{S}_1^c$ of n_1^c horizontal points ($n_1^c \leq n_0^c$) and the set $\mathcal{S}_0^l$ of n_0^l levels,
- $\mathcal{G}^v$ is the second intermediate subgrid, with the subset $\mathcal{S}_1^c$ of n_1^c horizontal points and a subset $\mathcal{S}_1^l$ of n_1^l levels ($n_1^l \leq n_0^l$),
- $\mathcal{G}^s$ is the final subgrid, with a subset $\mathcal{S}_2^c$ of n_2^c horizontal points ($n_2^c \leq n_1^c$) and the subset $\mathcal{S}_1^l$ of n_1^l levels, $\mathcal{S}_2^c$ being possibly different for each level of $\mathcal{S}_1^l$.

Thus:

- $\mathbf{S}^s$ is a horizontal interpolation from $\mathcal{S}_2^c$ to $\mathcal{S}_1^c$, possibly different for each level of $\mathcal{S}_1^l$,

- $\mathbf{S}^v$ is a vertical interpolation from $\mathcal{S}_1^l$ to $\mathcal{S}_0^l$, similar for all the points of $\mathcal{S}_1^c$,
- $\mathbf{S}^h$ is a horizontal interpolation from $\mathcal{S}_1^c$ to $\mathcal{S}_0^c$, similar for all the levels of $\mathcal{S}_0^l$.

Grid	Subset of points (number of points)	Subset of levels (number of levels)	Comment
$\mathcal{G}^s$	$\mathcal{S}_2^c$ (n_2^c)	$\mathcal{S}_1^l$ (n_1^l)	$\mathcal{S}_2^c$ depending on the level
Horizontal interpolation $\mathbf{S}^s$ from $\mathcal{S}_2^c$ to $\mathcal{S}_1^c$			
$\mathcal{G}^v$	$\mathcal{S}_1^c$ (n_1^c)	$\mathcal{S}_1^l$ (n_1^l)	$n_2^c \leq n_1^c$
Vertical interpolation $\mathbf{S}^v$ from $\mathcal{S}_1^l$ to $\mathcal{S}_0^l$			
$\mathcal{G}^h$	$\mathcal{S}_1^c$ (n_1^c)	$\mathcal{S}_0^l$ (n_0^l)	$n_1^l \leq n_0^l$
Horizontal interpolation $\mathbf{S}^h$ from $\mathcal{S}_1^c$ to $\mathcal{S}_0^c$			
$\mathcal{G}^f$	$\mathcal{S}_0^c$ (n_0^c)	$\mathcal{S}_0^l$ (n_0^l)	$n_1^c \leq n_0^c$

Table 1: Summary of grids, subsets and interpolations

In practice, horizontal interpolations $\mathbf{S}^s$ and $\mathbf{S}^h$ are bilinear interpolations: their number of operations is bounded by $3n_1^c$ and $3n_0^c$, respectively. The vertical interpolation $\mathbf{S}^v$ is a linear interpolation, with a number of operations bounded by $2n_0^l$.

3 Grid subsampling

3.1 Homogeneous subsampling

On the sphere, a Delaunay mesh of n^c points has $n^t = 2n^c - 2 - n^b$ triangles where n^b is the number of boundary nodes (negligible, so $n^t \approx 2n^c$ hereafter). If we assume that the mesh is homogenous, then these triangles are approximately equilateral with sides of length a. If a is sufficiently small to ignore the sphere surface curvature, the area of each triangle is approximately $(\sqrt{3}/4)a^2$. Thus, the total domain area is given by

$$\mathcal{A} \approx 2n^c \frac{\sqrt{3}a^2}{4} = \frac{\sqrt{3}n^c a^2}{2} \tag{3}$$

Assuming that we want to model a homogeneous convolution function on a support of radius r^h within this framework, we can define the resolution of the function as $\rho^h = r^h/a$. Thus:

$$n^c = \frac{2\mathcal{A}}{\sqrt{3}a^2} = \frac{2\mathcal{A}\left(\rho^h\right)^2}{\sqrt{3}\left(r^h\right)^2} \tag{4}$$

For the vertical direction, the same approach can be applied. The total height $\mathcal{H}$ is split into $n^l - 1$ layers of height h (and thus n^l interface levels). Assuming that we want to model a homogeneous

convolution function on a support of radius r^v within this framework, we can define the resolution of the function, given by $\rho^v = r^v/h$. Thus:

$$n^l = \frac{\mathcal{H}}{h} + 1 = \frac{\mathcal{H}\rho^v}{r^v} + 1 \tag{5}$$

Hereafter, we use the same resolution parameter for both horizontal and vertical directions ($\rho^h = \rho^v = \rho$), and equations 4 and 5 are used abusively in the case of heterogeneous support radii r^h and r^v, assuming that local variations are smooth enough.

3.2 Subgrid selection

The compact support convolution function is provided with locally variable support radii:

- radius $r^h(i^c, i^l)$ for the horizontal direction (dimension: meters),
- radius $r^v(i^c, i^l)$ for the vertical direction (dimension: meters or log(hPa))

where i^c refers to the horizontal point and i^l to the vertical level. The succesive grid subsamplings should reflect the variations of the support radii.

To define the horizontal subsampling from $\mathcal{S}_0^c$ to $\mathcal{S}_1^c$, we define at each horizontal point i^c the minimum support radius over the whole column:

$$r^h_{\min}(i^c) = \min_{i^l \in \mathcal{S}_0^l} r^h(i^c, i^l) \tag{6}$$

The horizontal average of $r^h_{\min}$, computed by

$$\overline{r^h_{\min}} = \frac{1}{n_0^c} \sum_{i^c \in \mathcal{S}_0^c} r^h_{\min}(i^c) \tag{7}$$

is used to define the number of horizontal points n_1^c in the subset $\mathcal{S}_1^c$:

$$n_1^c = \frac{2\mathcal{A}\rho^2}{\sqrt{3}\left(\overline{r^h_{\min}}\right)^2} \tag{8}$$

A trial-and-error algorithm selects n_1^c points of $\mathcal{S}_0^c$ to define $\mathcal{S}_1^c$. It computes the normalized horizontal distance between points i^c and j^c as

$$\widehat{d}^h_{\min}(i^c, j^c) = \frac{d^h(i^c, j^c)}{\sqrt{\frac{1}{2}\left(\left(r^h_{\min}(i^c)\right)^2 + \left(r^h_{\min}(j^c)\right)^2\right)}} \tag{9}$$

where $d^h(i^c, j^c)$ is the horizontal distance between points i^c and j^c. Then, the algorithm tries to maximize the minimum normalized distance $\widehat{d}^h_{\min}$ between all nearest neighbors in $\mathcal{S}_1^c$

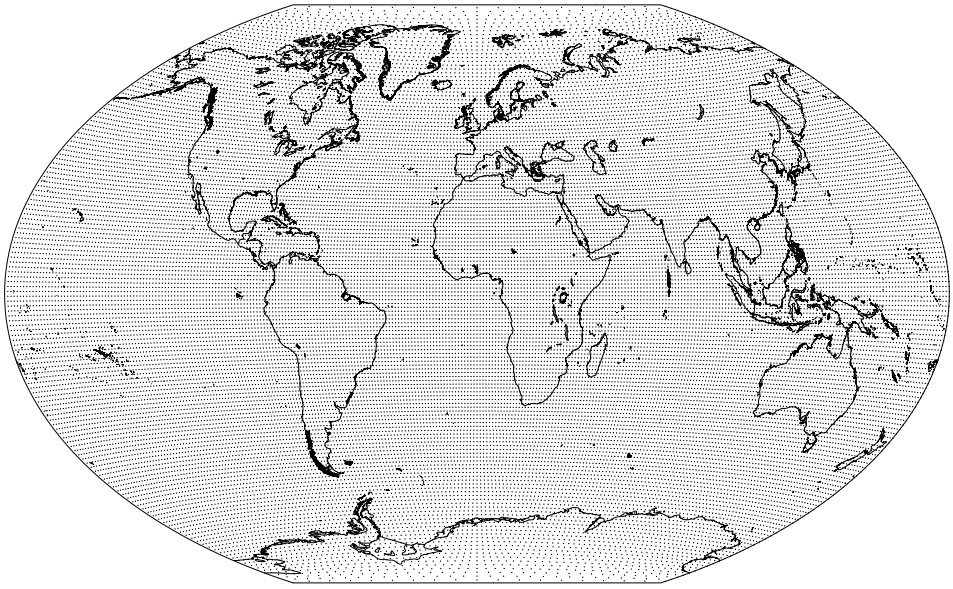

Figure 1: Black dots: ARPEGE grid at truncation TL149 (set $\mathcal{S}_0^c$)

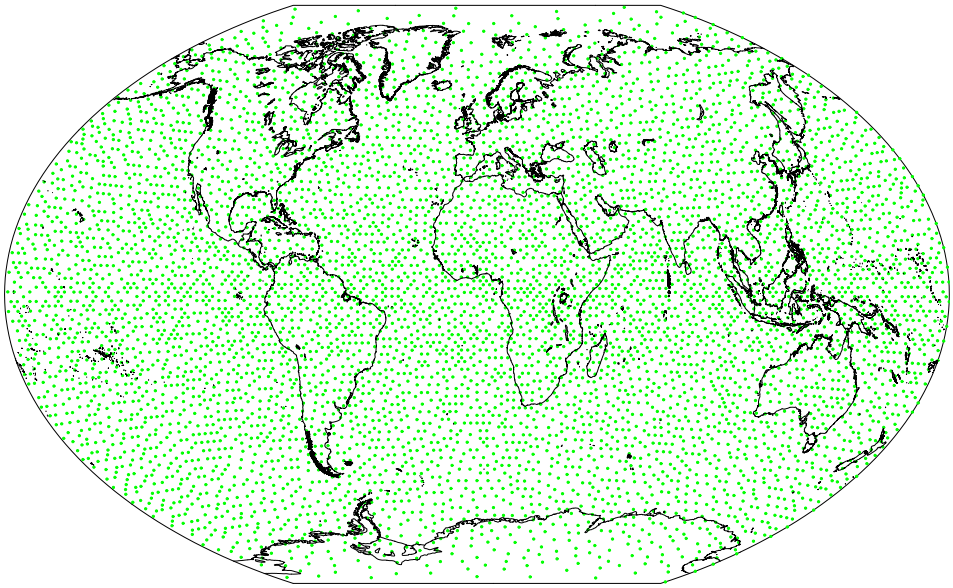

Figure 2: Green dots: subset $\mathcal{S}_1^c$ for a homogeneous support radius r^h

The vertical subsampling from $\mathcal{S}_0^l$ to $\mathcal{S}_1^l$ is defined similarly. At each horizontal point, we define the minimum vertical support radius over the whole level i^l:

$$r^v_{\min}(i^l) = \min_{i^c \in \mathcal{S}_1^c} r^v(i^c, i^l) \tag{10}$$

The normalized distance between two levels i^l and j^l is defined as

$$\widehat{d}^v_{\min}(i^l, j^l) = \frac{d^v(i^l, j^l)}{\sqrt{\frac{1}{2}\left(\left(r^v_{\min}(i^l)\right)^2 + \left(r^v_{\min}(j^l)\right)^2\right)}} \tag{11}$$

where $d^v(i^l, j^l)$ is the vertical distance between levels i^l and j^l (measured in meters or in pressure logarithm for instance). We start from the first level (always in the $\mathcal{S}_1^l$ subset) and look for the next vertical level for which the vertical normalized distance $\widehat{d}^v_{\text{min}}$ with the first level is larger than $1/\rho$. Thus, the distance d^v between these levels is approximately r^v/ρ. Once such a level is reached, the procedure starts again towards the next one. The last level is automatically included in the $\mathcal{S}_1^l$ subset.

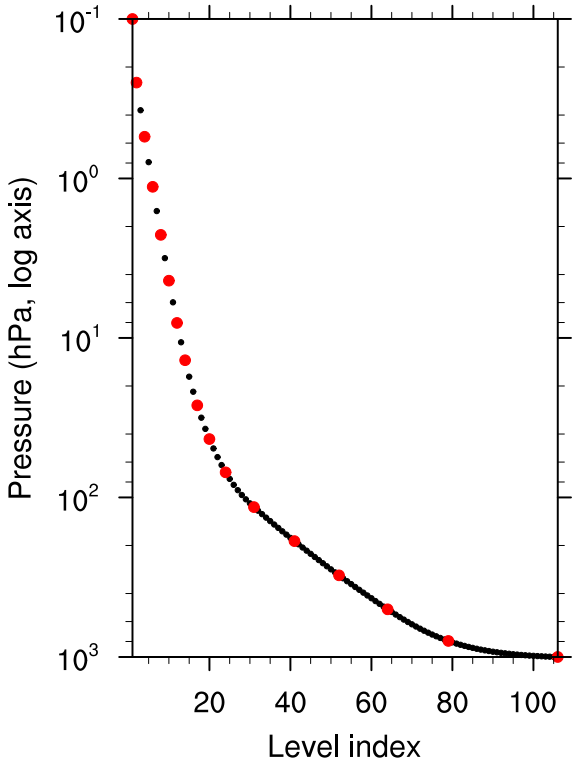

Figure 3: Black dots: ARPEGE levels; red dots: subset $\mathcal{S}_1^l$ with a homogeneous vertical support radius r^v in pressure logarithm coordinate

A final horizontal subsampling is applied for each level of $\mathcal{S}_1^l$ from $\mathcal{S}_1^c$ to $\mathcal{S}_2^c$. The trial-and-error algorithm select $n_2^c(i^l)$ horizontal points at each level $i^l \in \mathcal{S}_1^l$ with

$$n_2^c(i^l) = \frac{2\mathcal{A}\rho^2}{\sqrt{3}\left(\overline{r^h}(i^l)\right)^2} \tag{12}$$

where

$$\overline{r^h}(i^l) = \frac{1}{n_1^c} \sum_{i^c \in \mathcal{S}_1^c} r^h(i^c, i^l) \tag{13}$$

is the horizontal average of r^h at level i^l. The algorithm tries to maximize a normalized horizontal distance, defined between points i^c and j^c at level i^l as

$$\widehat{d}^s(i^c, j^c) = \frac{d^h(i^c, j^c)}{\sqrt{\frac{1}{2}\left(\left(r^h(i^c, i^l)\right)^2 + \left(r^h(j^c, i^l)\right)^2\right)}} \tag{14}$$

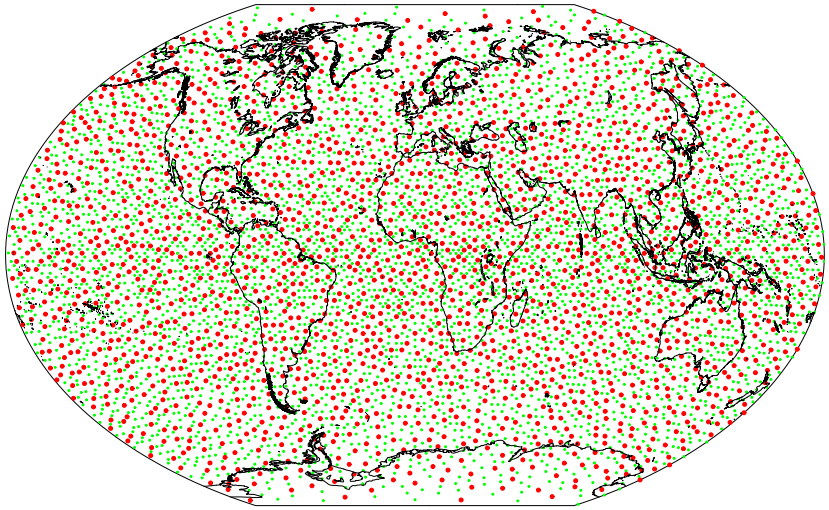

Figure 4: Red dots: subset $\mathcal{S}_2^c$ at a level with a small support radius r^h

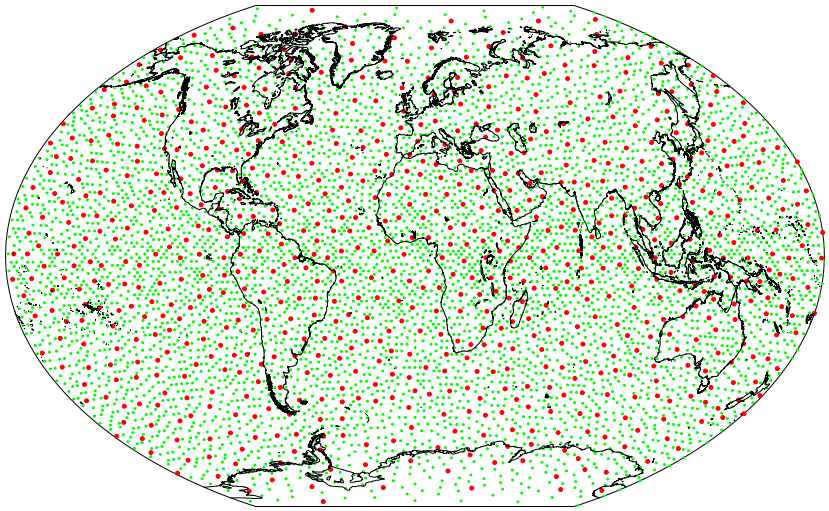

Figure 5: Red dots: subset $\mathcal{S}_2^c$ at a level with a moderate support radius r^h

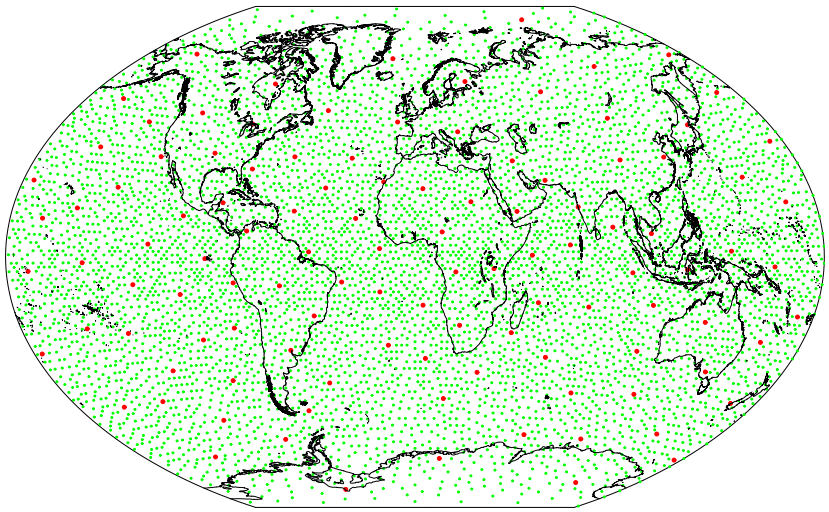

Figure 6: Red dots: subset $\mathcal{S}_2^c$ at a level with a large support radius r^h

The trial-and-error algorithm is able to detect the local variations of support radius r^h to adapt the subgrid, as shown by Figures 7 and 8. The larger the support radius r^h, the coarser the subgrid mesh.

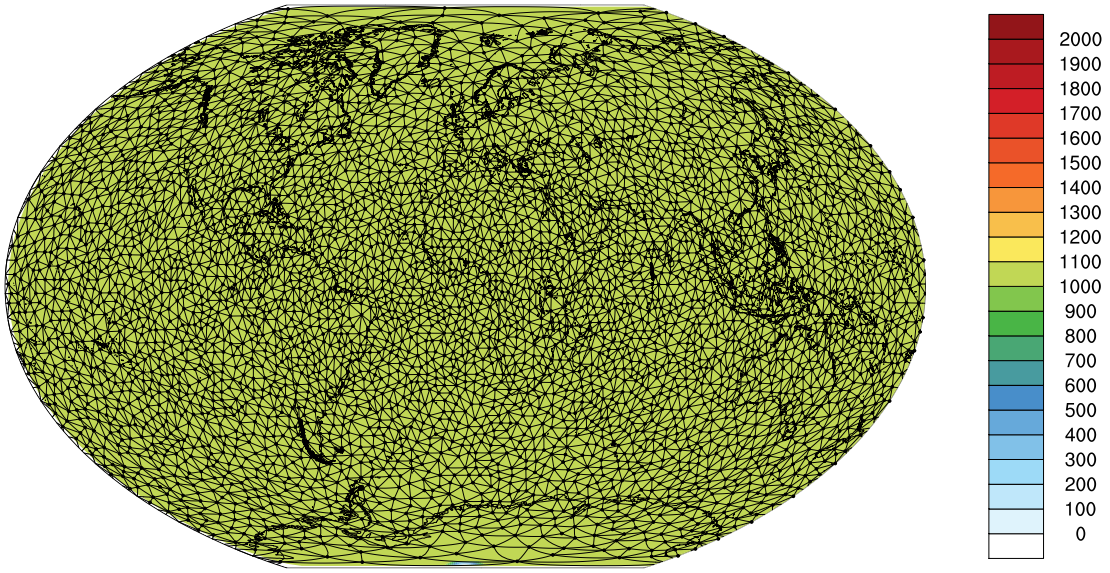

Figure 7: Subset $\mathcal{S}_2^c$ for a homogeneous support radius r^h

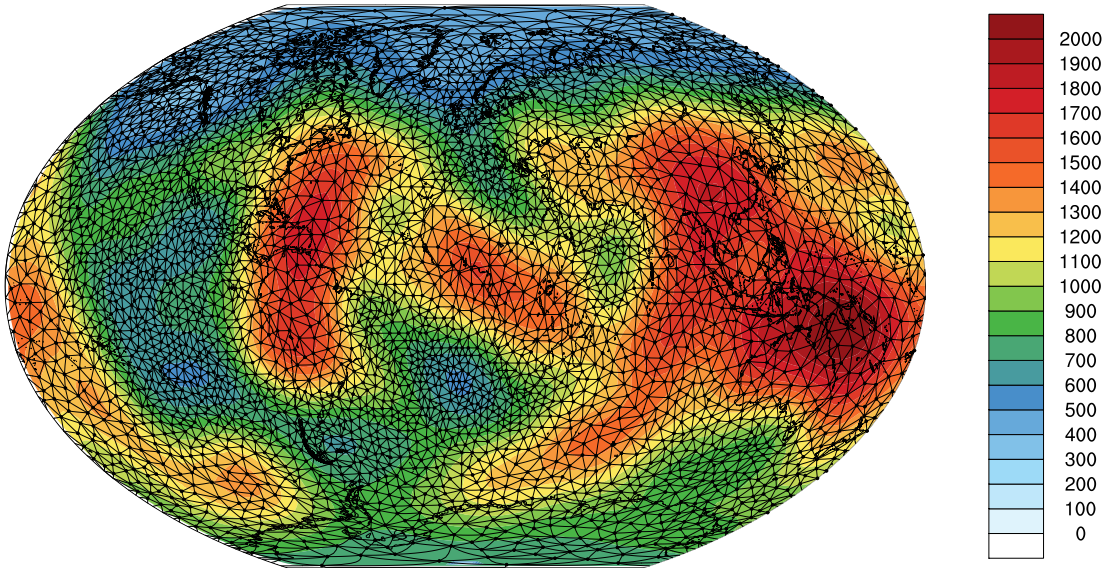

Figure 8: Subset $\mathcal{S}_2^c$ for a smoothly heterogeneous support radius r^h

4 Convolution function

4.1 Distance-based approach

The convolution function used in NICAS is the famous Gaspari and Cohn (1999) function (GC99 hereafter), adapted to a the case where the distance is normalized by the support radius:

$$\mathcal{C}(\widehat{d}) = \begin{cases} 1 - 8\widehat{d}^5 + 8\widehat{d}^4 + 5\widehat{d}^3 - \frac{20}{3}\widehat{d}^2 & \text{if } \widehat{d} \leq \frac{1}{2} \\ \frac{8}{3}\widehat{d}^5 - 8\widehat{d}^4 + 5\widehat{d}^3 + \frac{20}{3}\widehat{d}^2 - 10\widehat{d} + 4 - \frac{1}{3}\widehat{d}^{-1} & \text{if } \frac{1}{2} < \widehat{d} \leq 1 \\ 0 & \text{if } 1 < \widehat{d} \end{cases} \tag{15}$$

where $\widehat{d}$ is a normalized distance (great-circle distance divided by the compact support radius).

The 3D normalized distance between nodes with coordinates (i^c, i^l) and (j^c, j^l) is:

$$\widehat{d}(i^c, i^l; j^c, j^l) = \sqrt{\left(\widehat{d}^h(i^c, i^l; j^c, j^l)\right)^2 + \left(\widehat{d}^v(i^c, i^l; j^c, j^l)\right)^2} \tag{16}$$

where $\widehat{d}^h$ and $\widehat{d}^v$ are the horizontal and vertical normalized distances, respectively given by:

$$\widehat{d}^h(i^c, i^l; j^c, j^l) = \frac{d^h(i^c, j^c)}{\sqrt{\frac{1}{2}\left(\left(r^h(i^c, i^l)\right)^2 + \left(r^h(j^c, j^l)\right)^2\right)}} \tag{17}$$

$$\widehat{d}^v(i^c, i^l; j^c, j^l) = \frac{d^v(i^l, j^l)}{\sqrt{\frac{1}{2}\left(\left(r^v(i^c, i^l)\right)^2 + \left(r^v(j^c, j^l)\right)^2\right)}} \tag{18}$$

This approach is valid for homogeneous and smoothly varying support radii, whose variations length-scale is larger than the convolution function length-scale, as shown by Figures 9 and 10.

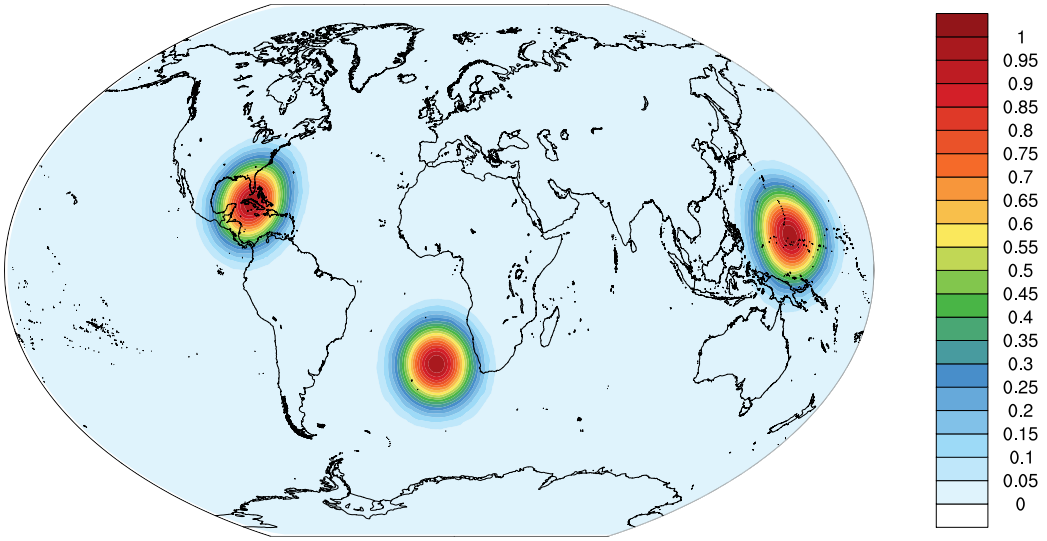

Figure 9: Convolution functions for a homogeneous support radius r^h

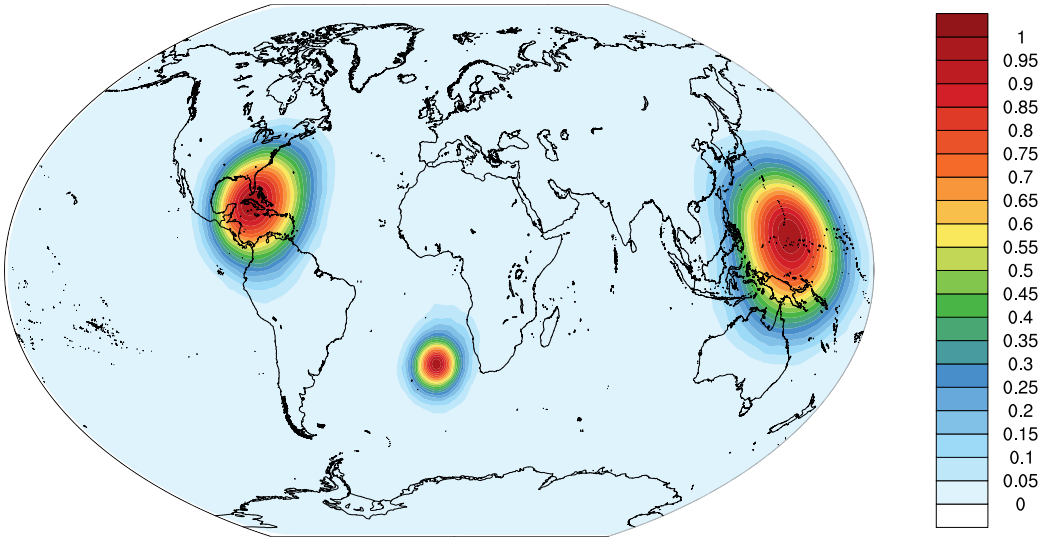

Figure 10: Convolution functions for a smoothly heterogeneous support radius r^h

4.2 Network-based approach

If r^h and r^v are heterogeneous and rapidly varying, the distance-based approach is irrelevant. Figures 12 and 13 show a case with a belt of lower support radius r^h is surrounding the point where the convolution function is applied. Since the values of the convolution function depends on the support radii at the center and at the concerned nodes only, the function values outside this lower radius belt are not affected by it. This is an issue: the signal should be less propagated through a zone a lower radius r^h.

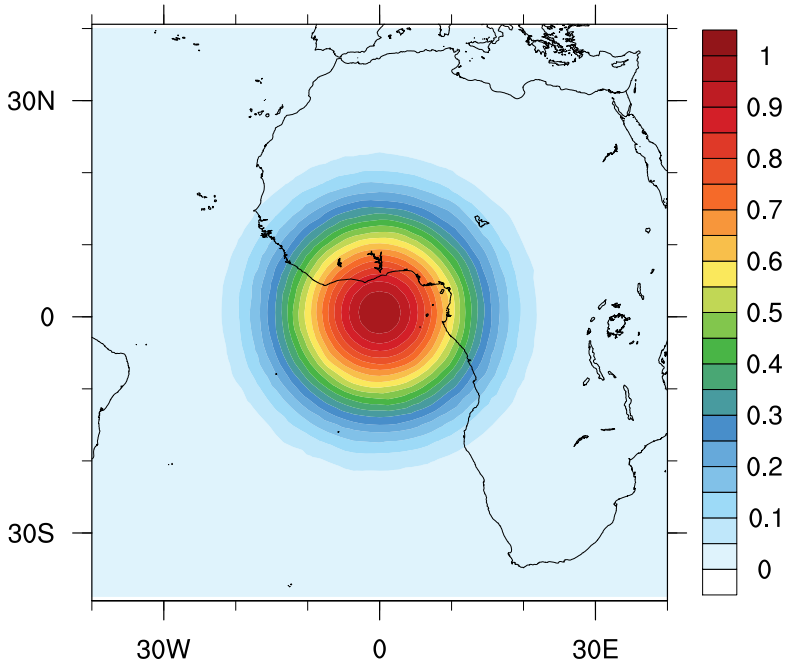

Figure 11: Convolution function for a homogeneous support radius r^h

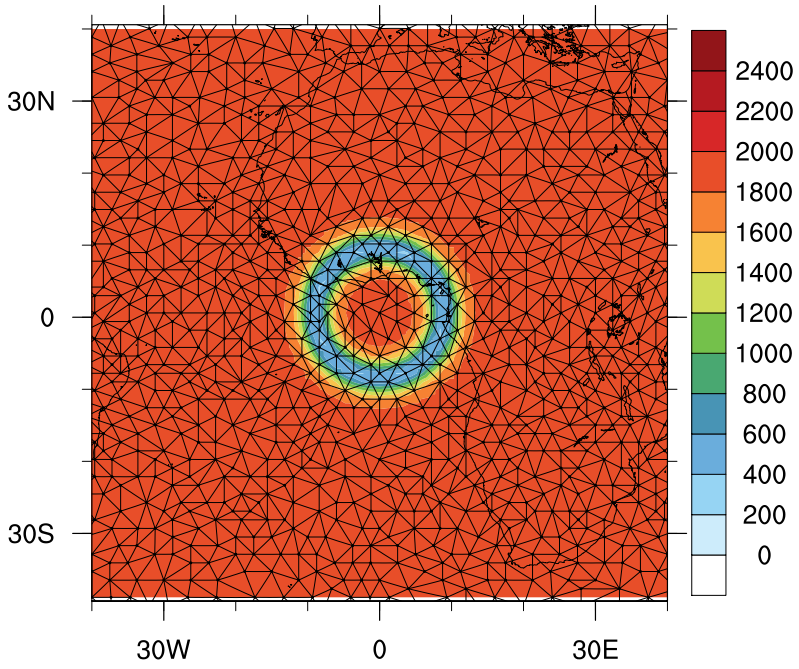

Figure 12: Heterogeneous support radius r^h and subset $\mathcal{S}_2^c$ (black dots)

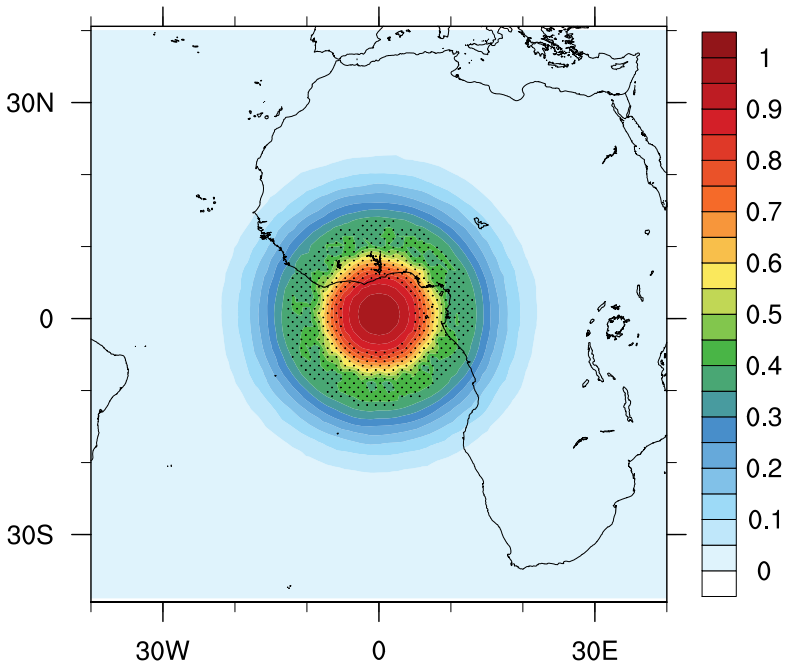

Figure 13: Convolution function for a heterogeneous support radius r^h with the distance-based approach; zone of lower r^h (dotted)

A simple solution is to switch to a network-based approach. First, we compute all the 3D normalized distances between nearest neighbors on the full grid $\mathcal{G}^f$. Then, the normalized distance between to non-neighbor nodes is computed as the shortest integrated normalized distance on a path along $\mathcal{G}^f$ nodes. Figure 14 shows that this method is able to provide the expected result: the signal is less propagated through the belt of lower radius r^h. All the nodes within and beyond that belt are affected.

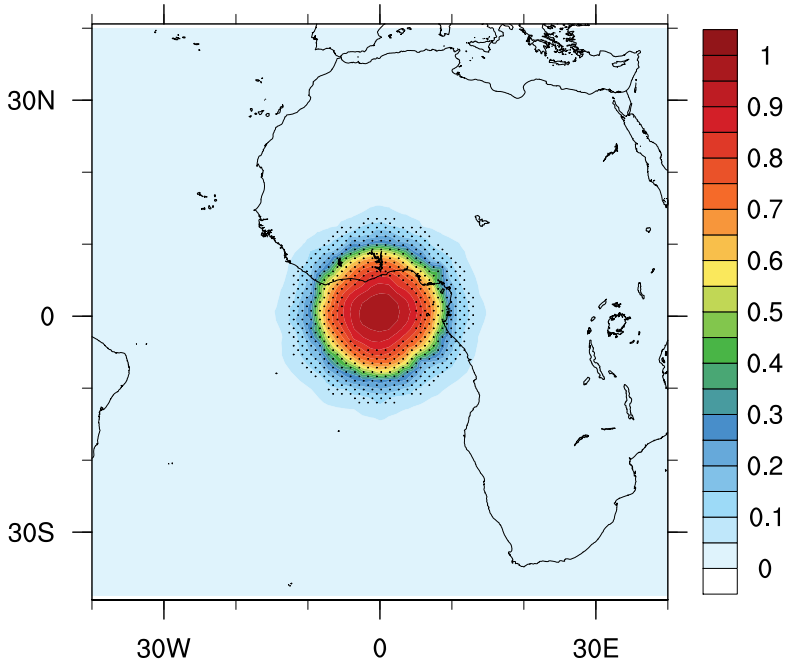

Figure 14: Convolution function for a heterogeneous support radius r^h with the network-based approach; zone of lower r^h (dotted)

This network-based approach can deal with complex boundaries in a better way than the distance-based approach, for which complex boundaries are just a mask applied on the underlying convolution functions. Figure 15 shows for the NEMO grid that the network-based approach is able to contain the convolution function to the West side of the Isthmus of Panama (interpolation has to be adapted so that bilinear interpolation triangles do not cross mask boundaries). This approach can also limit the signal propagation from the West coast to the East coast of Florida. However, it is not a full integrative method as diffusion-based methods. For instance, the width of straits is not taken into account to propagate the signal, only the distance around obstacles matters.

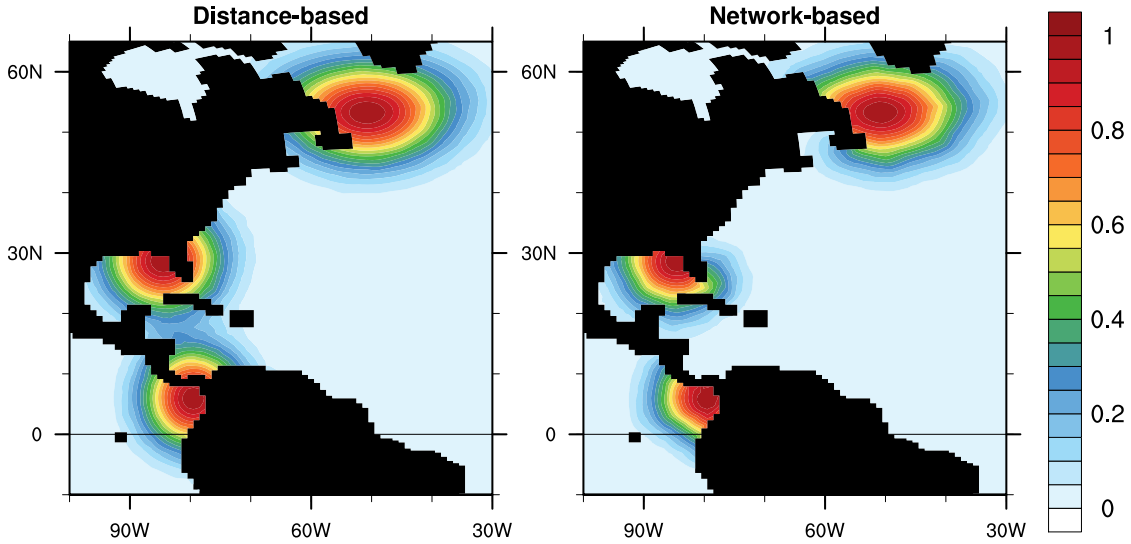

Figure 15: Convolution functions with complex boundaries of the NEMO grid, for a distance-based approach (left) and a network-based approach (right)

5 Normalization

As mentioned in the introduction, even if the subgrid convolution matrix $\mathbf{C}^s$ is normalized, $\mathbf{S}\mathbf{C}^s\mathbf{S}^{\mathrm{T}}$ is not normalized everywhere (only on the nodes of $\mathcal{G}^s$). As a consequence, a normalization operator $\mathbf{N}$ (diagonal) is applied on both sides of $\mathbf{S}\mathbf{C}^s\mathbf{S}^{\mathrm{T}}$, to ensure that $\widetilde{\mathbf{C}}$ is exactly normalized.

The simpler method to compute $\mathbf{N}$ would be to apply $\mathbf{S}\mathbf{C}^s\mathbf{S}^{\mathrm{T}}$ to a Dirac vector $\boldsymbol{\delta}_i$:

$$N_{ii} = \left(\boldsymbol{\delta}_i^{\mathrm{T}}\mathbf{S}\mathbf{C}^s\mathbf{S}^{\mathrm{T}}\boldsymbol{\delta}_i\right)^{-1/2} \tag{19}$$

where $\boldsymbol{\delta}_i \in \mathbb{R}^n$ is a vector of zeros with only the i^{th} coefficient equal to one. However, the cost of applying the full $\mathbf{S}\mathbf{C}^s\mathbf{S}^{\mathrm{T}}$ to Dirac vectors at each node of the full grid $\mathcal{G}^f$ is prohibitive.

If we look at the details of the previous operations though, the number of relevant nodes involved in the computation of N_{ii} is very limited:

- 1 node of $\mathcal{G}^f$ is non-zero in $\boldsymbol{\delta}_i$,
- up to 3 nodes of $\mathcal{G}^h$ are non-zero in $\mathbf{S}^{h\mathrm{T}}\boldsymbol{\delta}_i$ (bilinear interpolation),
- up to 6 nodes of $\mathcal{G}^v$ are non-zero in $\mathbf{S}^{v\mathrm{T}}\mathbf{S}^{h\mathrm{T}}\boldsymbol{\delta}_i$ (linear interpolation),
- up to 18 nodes of $\mathcal{G}^s$ are non-zero in $\mathbf{S}^{s\mathrm{T}}\mathbf{S}^{v\mathrm{T}}\mathbf{S}^{h\mathrm{T}}\boldsymbol{\delta}_i$ (bilinear interpolation),

Thus, the convolution of these 18 non-zero nodes only has an impact on N_{ii}. An affordable procedure is the following:

1. computing the 18 non-zero values of $\mathbf{S}^{s\mathrm{T}}\mathbf{S}^{v\mathrm{T}}\mathbf{S}^{h\mathrm{T}}\boldsymbol{\delta}_i$ and storing them in a vector $\boldsymbol{\delta}' \in \mathbb{R}^{18}$,
2. applying the relevant block of the subgrid convolution matrix $\mathbf{C}^s$ to the coefficients of $\boldsymbol{\delta}'$ and storing only the result for these coefficients in a vector $\boldsymbol{\delta}'' \in \mathbb{R}^{18}$,
3. computing the final result as the inverse square-root of a scalar product: $N_{ii} = \langle \boldsymbol{\delta}', \boldsymbol{\delta}'' \rangle^{-1/2}$.

The computational cost of this procedure is far lower than applying the full $\mathbf{S}\mathbf{C}^s\mathbf{S}^{\mathrm{T}}$. Besides, the procedure can be applied in parallel to all nodes, dividing the elapsed time by the number of parallel threads.

This normalization procedure does not rely on an explicit formula that would be derived in a continuous framework. Thus, it is exact for both homogeneous and heterogeneous support radii, and also in the presence of complex boundaries.

6 Square-root formulation

6.1 Approximate solution

Data assimilation solvers often require a square-root $\mathbf{U}$ of the localization matrix $\mathbf{C}$, such that $\mathbf{C} = \mathbf{U}\mathbf{U}^{\mathrm{T}}$. For the NICAS method:

$$\widetilde{\mathbf{U}} = \mathbf{N}\mathbf{S}\mathbf{U}^s \tag{20}$$

is a square-root of $\widetilde{\mathbf{C}}$ if $\mathbf{U}^s$ is a square-root of $\mathbf{C}^s$. However, computing an exact square-root of $\mathbf{C}^s$ is not affordable with explicit methods (ex. Cholesky decomposition) if n^s is too large. An approximate square-root can be obtained by noting that:

- the (convolution) square-root of a Gaussian function with a homogeneous length-scale L is also a Gaussian function with a homogeneous length-scale $L/\sqrt{2}$,
- the GC99 function has a Gaussian-like shape.

Thus, an approximate square-root of $\mathbf{C}^s$ can be modeled as follows:

$$\mathbf{U}^s = \mathbf{N}'\mathbf{V}^s \tag{21}$$

where:

- $\mathbf{V}^s \in \mathbb{R}^{n^s \times n^s}$ is convolution matrix on the same subgrid $\mathcal{G}^s$ as $\mathbf{C}^s$, but with support radii divided by $\sqrt{2}$,
- $\mathbf{N}' \in \mathbb{R}^{n^s \times n^s}$ is a diagonal normalization matrix ensuring that $\mathbf{U}^s\mathbf{U}^{s^{\mathrm{T}}} = \mathbf{N}'\mathbf{V}^s\mathbf{V}^{s^{\mathrm{T}}}\mathbf{N}'^{\mathrm{T}}$ is exactly normalized (as $\mathbf{C}^s$ is).

Thus, an approximate square-root of $\widetilde{\mathbf{C}}$ is given by $\widetilde{\mathbf{U}} \in \mathbb{R}^{n \times n^s}$:

$$\widetilde{\mathbf{U}} = \mathbf{N}\mathbf{S}\mathbf{N}'\mathbf{V}^s \tag{22}$$

The control vector state corresponding to this square-root has a size n^s, which is significantly smaller than n.

The difference between applying $\widetilde{\mathbf{C}}$ or $\widetilde{\mathbf{U}}\widetilde{\mathbf{U}}^{\mathrm{T}}$ with $\widetilde{\mathbf{U}}$ being an approximate square-root is small, as shown by Figure 16 (to be compared with Figure 9) for a homogeneous support radius and Figure 17 (to be compared with Figure 8) for a heterogenous support radius.

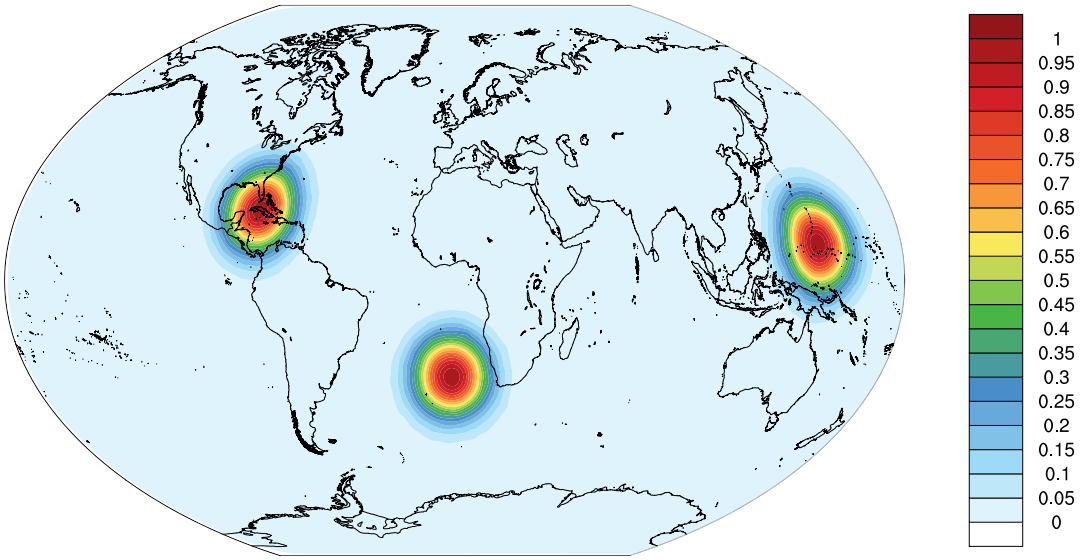

Figure 16: Convolution functions for a homogeneous support radius r^h, applied with the square-root formulation $\widetilde{\mathbf{U}}\widetilde{\mathbf{U}}^{\mathrm{T}}$

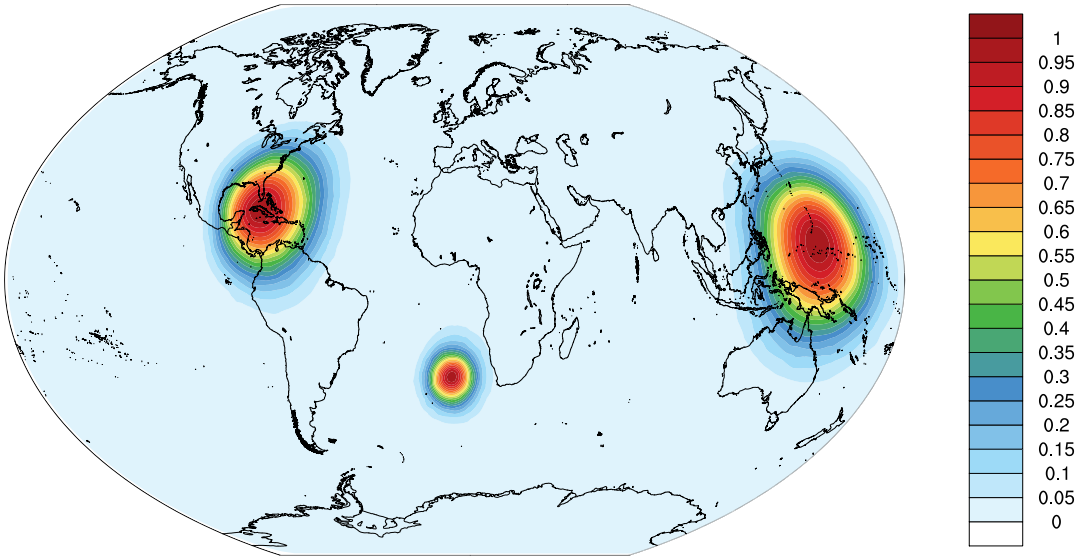

Figure 17: Convolution functions for a smoothly heterogeneous support radius r^h, applied with the square-root formulation $\widetilde{\mathbf{U}}\widetilde{\mathbf{U}}^{\mathrm{T}}$

6.2 Impact on the normalization procedure

Since the square-root defined in the previous section is only approximate, the normalization procedure has to be updated.

The first step is to compute the internal normalization $\mathbf{N}'$. The basic method is applied: for each node of the subgrid $\mathcal{G}^s$, the convolution adjoint $\mathbf{V}^{s\mathrm{T}}$ is applied to a Dirac vector at the concerned node. Only the non-zero values inside the compact support of reduced size (divided by $\sqrt{2}$) are stored in a vector. The inverse square-root of the euclidean norm of this vector gives the internal normalization diagonal coefficient.

The second step is to compute the external normalization $\mathbf{N}$. The previously described procedure cannot be applied directly but it has to be slightly modified:

1. computing the 18 non-zero values of $\mathbf{S}^{s\mathrm{T}}\mathbf{S}^{v\mathrm{T}}\mathbf{S}^{h\mathrm{T}}\boldsymbol{\delta}_i$ and storing them in a vector $\boldsymbol{\delta}' \in \mathbb{R}^{18}$,
2. applying the relevant **columns** of the subgrid convolution matrix $\mathbf{C}^s$ to the coefficients of $\boldsymbol{\delta}'$ and storing **all non-zero resulting values in** $\mathcal{G}^s$ in a vector $\boldsymbol{\delta}''$,
3. computing the final result as the inverse square-root of the norm of $\boldsymbol{\delta}''$.

6.3 Consistent length-scales specification

It is a usual practice to split the square-root $\mathbf{U}$ of a localization operator $\mathbf{C}$ into horizontal and vertical smoothers, denoted $\mathbf{U}_h$ and $\mathbf{U}_v$ respectively, which are applied sequentially. Two strategies are available:

- $\mathbf{U} = \mathbf{U}_h\mathbf{U}_v$ leading to $\mathbf{C} = \mathbf{U}_h\mathbf{U}_v\mathbf{U}_v^{\mathrm{T}}\mathbf{U}_h^{\mathrm{T}}$.
- $\mathbf{U} = \mathbf{U}_v\mathbf{U}_h$ leading to $\mathbf{C} = \mathbf{U}_v\mathbf{U}_h\mathbf{U}_h^{\mathrm{T}}\mathbf{U}_v^{\mathrm{T}}$.

If either $\mathbf{U}_h$ or $\mathbf{U}_v$ is heterogeneous, then it is impossible to define a precise link between the horizontal and vertical length-scales of $\mathbf{U}_h$ and $\mathbf{U}_v$, and the horizontal and vertical length-scales of $\mathbf{C}$.

With the NICAS method, the convolution function is fully 3D, even if the interpolation is performed with successive horizontal and vertical steps. Thus, the method is able to produce localization functions whose horizontal and vertical support radii are consistent with the values provided by the user, even in heterogeneous cases.

7 Parallelization

For high dimensional systems, the domain is split into multiple tiles, each one handled by a different task. Data are exchanged between tasks via the MPI protocol. These communications have a cost, which can dominate the computational cost in some cases. A particular care must be taken when designing the parallel implementation.

7.1 Halos

Four different halo zones are defined:

- **[F]** zone: full grid $\mathcal{G}^f$ nodes on a given task (usually provided by the model itself),
- **[A]** zone: subgrid $\mathcal{G}^s$ nodes on a given task (define by the subsampling on **[F]**),
- **[B]** zone: subgrid $\mathcal{G}^s$ nodes involved in the interpolation ($\mathbf{S}$ goes from **[B]** to **[F]** and $\mathbf{S}^{\mathrm{T}}$ from **[F]** to **[B]**)
- **[C]** zone: subgrid $\mathcal{G}^s$ nodes involved in the convolution ($\mathbf{C}^s$ goes from **[C]** to **[C]**).

7.2 Single communication method

A single communication step can be required between the convolution and the interpolation, from halo zone **[C]** to halo zone **[B]**, via halo zone **[A]**. In the following expression, symbol $\leftarrow\!\!\prec$ stands for a communication between halo zones of different tasks, whereas $\leftarrow$ stands for a simple copy from a halo zone to another one on the same task (no communication). Equation 1 is written here with additional subscripts indicating the working halo zones:

$$\widetilde{\mathbf{C}} = {}_{[\mathbf{F}]}\mathbf{N}\,{}_{[\mathbf{F}]}\mathbf{S}\,{}_{[\mathbf{B}]\leftarrow\!\!\prec[\mathbf{A}]\leftarrow\!\!\prec[\mathbf{C}]}\mathbf{C}^s\,{}_{[\mathbf{C}]\leftarrow[\mathbf{B}]}\mathbf{S}^{\mathrm{T}}\,{}_{[\mathbf{F}]}\mathbf{N}^{\mathrm{T}}\,{}_{[\mathbf{F}]} \tag{23}$$

The nodes involved in the convolution $\mathbf{C}^s$ are copied from halo zone **[B]** to halo zone **[C]** before the convolution. Thus, the halo zone **[C]** must include all the nodes involved in the convolution of nodes in the halo zone **[B]**.

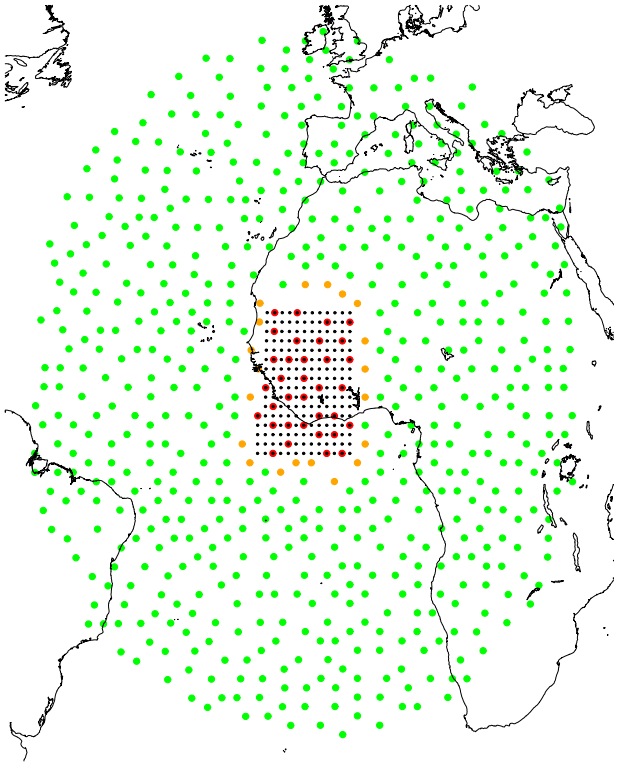

Figure 18: On a given task: halo zones **[F]**, [A], [B] and [C] for the single communication method

7.3 Double communication method

Another option is to add a communication step between the adjoint interpolation and the convolution, from halo zone **[B]** to halo zone **[A]**:

$$\widetilde{\mathbf{C}} = {}_{[\mathbf{F}]}\mathbf{N}_{[\mathbf{F}]}\,\mathbf{S}_{[\mathbf{B}]\nleftarrow[\mathbf{A}]\nleftarrow[\mathbf{C}]}\,\mathbf{C}^{s}_{[\mathbf{C}]\leftarrow[\mathbf{A}]\nleftarrow[\mathbf{B}]}\,\mathbf{S}^{\mathrm{T}}_{[\mathbf{F}]}\,\mathbf{N}^{\mathrm{T}}_{[\mathbf{F}]} \tag{24}$$

The nodes involved in the convolution are copied from halo zone **[A]** to halo zone **[C]** before the convolution. Thus, the halo zone **[C]** must include all the nodes involved in the convolution of nodes in the halo zone **[A]**.

The double communication method has an extra communication step, but the halo zone **[C]** is slightly reduced compared to the single communication method. Surprisingly, preliminary tests show that the double communication method is always faster than the single communication method.

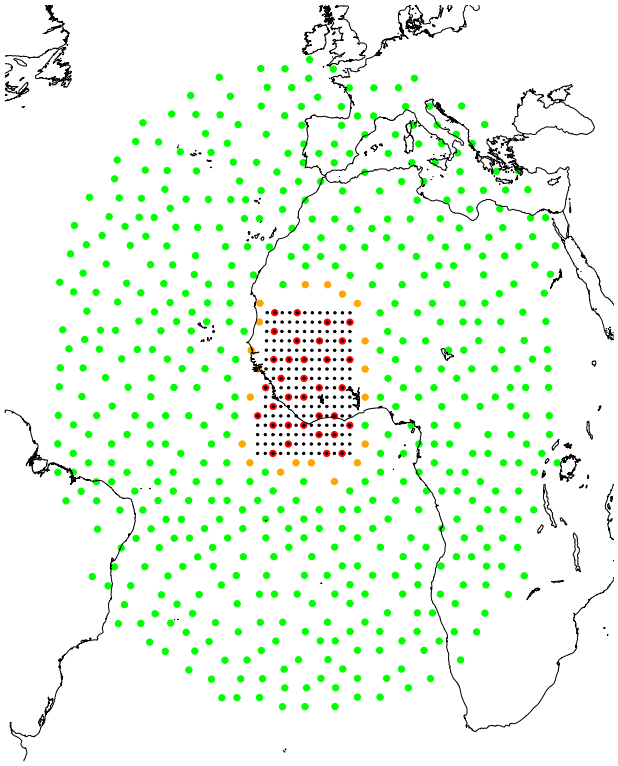

Figure 19: On a given task: halo zones **[F]**, [A], [B] and [C] for the double communication method

8 Numerical efficiency

8.1 Theoretical computational cost

The cost of the direct and adjoint interpolations is directly linked to the kind of horizontal and vertical interpolation used. In NICAS, we use a bilinear interpolation for the horizontal and a linear interpolation for the vertical:

- the number of operations in $\mathbf{S}^h$ is bounded by $3n_0^c n_0^l = 3n$,
- the number of operations in $\mathbf{S}^v$ is bounded by $2n_0^l n_1^c$,
- the number of operations in $\mathbf{S}^s$ is bounded by $3\sum_{i^l \in \mathcal{S}_1^l} n_2^c(i^l) \leq 3n_1^c n_1^l$

Thus, the dominant cost is in the horizontal interpolation $\mathbf{S}^h$, and the number of operations is bounded by $n^{\text{interp}} = 3n$. In a parallel implementation with p tasks, the cost per task becomes

$$n_{\text{MPI}}^{\text{interp}} = 3\frac{n}{p} \tag{25}$$

The interpolation cost is independent from the convolution parameters and is only proportional to the number of $\mathcal{G}^f$ nodes per task.

We can estimate the size of the subgrid $\mathcal{G}^s$ in the context of homogeneous support radii r^h and r^v. Equation 4 and 5 lead to an approximate value of n^s:

$$n^s = n^c n^l \approx \frac{2\mathcal{V}\rho^3}{\sqrt{3}\left(r^h\right)^2 r^v} \tag{26}$$

where $\mathcal{V} = \mathcal{A}\mathcal{H}$ is the domain volume. The 3D compact support of the convolution function is an ellipsoid of volume:

$$\mathcal{V}_{\text{cs}} = \frac{4}{3}\pi\left(r^h\right)^2 r^v \tag{27}$$

Thus, n^s can be expressed as:

$$n^s = \frac{8\pi}{3\sqrt{3}}\frac{\mathcal{V}}{\mathcal{V}_{\text{cs}}}\rho^3 \tag{28}$$

The density of nodes $\mathcal{D}$ is:

$$\mathcal{D} = \frac{n_s}{\mathcal{V}} = \frac{8\pi}{3\sqrt{3}}\frac{\rho^3}{\mathcal{V}_{\text{cs}}} \tag{29}$$

so the number of points in the 3D compact support of the convolution function is:

$$m^s = \mathcal{D}\mathcal{V}_{\text{cs}} = \frac{8\pi\rho^3}{3\sqrt{3}} \tag{30}$$

As a consequence, the number of operations for the convolution is approximately:

$$n^{\text{convol}} = n^s m^s = \frac{64\pi^2}{27}\frac{\mathcal{V}}{\mathcal{V}_{\text{cs}}}\rho^6 = \frac{64\pi^2}{27}n\frac{v}{\mathcal{V}_{\text{cs}}}\rho^6 \tag{31}$$

where v is the average volume of a cell of $\mathcal{G}^f$. In a parallel implementation with p tasks, the cost per task becomes

$$n^{\text{convol}}_{\text{MPI}} = \frac{64\pi^2}{27}\frac{n}{p}\frac{v}{\mathcal{V}_{\text{cs}}}\rho^6 \tag{32}$$

Thus, the convolution cost is proportional to:

- the number of $\mathcal{G}^f$ nodes per task,
- the inverse of the ratio $\mathcal{V}_{\text{cs}}/v$, which gives an indication of the convolution function support volume compared to a grid cell volume, i.e. the smoothing intensity of the convolution,
- the resolution ρ to the power 6.

8.2 Theoretical communication cost

We consider a parallel implementation with p tasks where:

- all levels of a given column are handled by the same task,
- each task handles points on a square of side $r_{[\mathbf{A}]}$ (area $\mathcal{A}_{[\mathbf{A}]} = r_{[\mathbf{A}]}^2$),
- the repartition among tasks is even: $\mathcal{A}_{[\mathbf{A}]} = \mathcal{A}/p$
- p is large enough so that the halo zone **[C]** is not overlapping itself.

In this case, the area of the halo zone **[C]** is given by:

$$\mathcal{A}_{[\mathbf{C}]} = \mathcal{A}_{[\mathbf{A}]} + 4r_{[\mathbf{A}]}r^h + \pi\left(r^h\right)^2 \tag{33}$$

so that the area in the halo zone **[C]** but not in the halo zone **[A]** is

$$\overline{\mathcal{A}_{[\mathbf{C}]}} = \mathcal{A}_{[\mathbf{C}]} - \mathcal{A}_{[\mathbf{A}]} = 4r_{[\mathbf{A}]}r^h + \pi\left(r^h\right)^2 \tag{34}$$

Thus, the number of communications between the halo zone **[C]** and the halo zone **[A]**, which is approximately equal to the number of points in the volume $\overline{\mathcal{V}_{[\mathbf{C}]}} = \overline{\mathcal{A}_{[\mathbf{C}]}}\mathcal{H}$, is given by:

$$\begin{aligned} n_{\text{com}} &= \mathcal{D}\overline{\mathcal{V}_{[\mathbf{C}]}} \\ &= \frac{8\pi}{3\sqrt{3}}\frac{\rho^3}{\mathcal{V}_{\text{cs}}}\left(4r_{[\mathbf{A}]}r^h + \pi\left(r^h\right)^2\right)\mathcal{H} \\ &= \frac{4\pi}{\sqrt{3}}\rho^2 n^l\left(\sqrt{\frac{p_{\text{lim}}}{p}} + 1\right) \end{aligned} \tag{35}$$

where

$$p_{\text{lim}} = \frac{16\mathcal{A}}{\pi^2\left(r^h\right)^2} \tag{36}$$

For instance in a global model:

- with a support radius r^h = 2000 km, $p_{\text{lim}} \approx 207$,
- with a support radius r^h = 1000 km, $p_{\text{lim}} \approx 826$,
- with a support radius r^h = 500 km, $p_{\text{lim}} \approx 3307$.

We notice, as expected, that the number of communications n_{com} is proportional to the number of levels n^l in the subgrid, and to the resolution ρ squared. If $p \ll p_{\text{lim}}$, then n_{com} is also proportional to the inverse square-root of p.

A key aspect of the NICAS method is that communications are performed on the subgrid $\mathcal{G}^s$ and do not depend explicitly on the size of the full grid $\mathcal{G}^f$. However in the present implementation, the distribution of $\mathcal{G}^s$ points among tasks is still defined from the distribution of $\mathcal{G}^f$ points, which can introduce imbalances in the tasks load.

8.3 Resolution impact

The resolution parameter has been introduced to compute the subgrid size. Figures 20, 21 and 22 show the decrease of subgrid density with the resolution parameter.

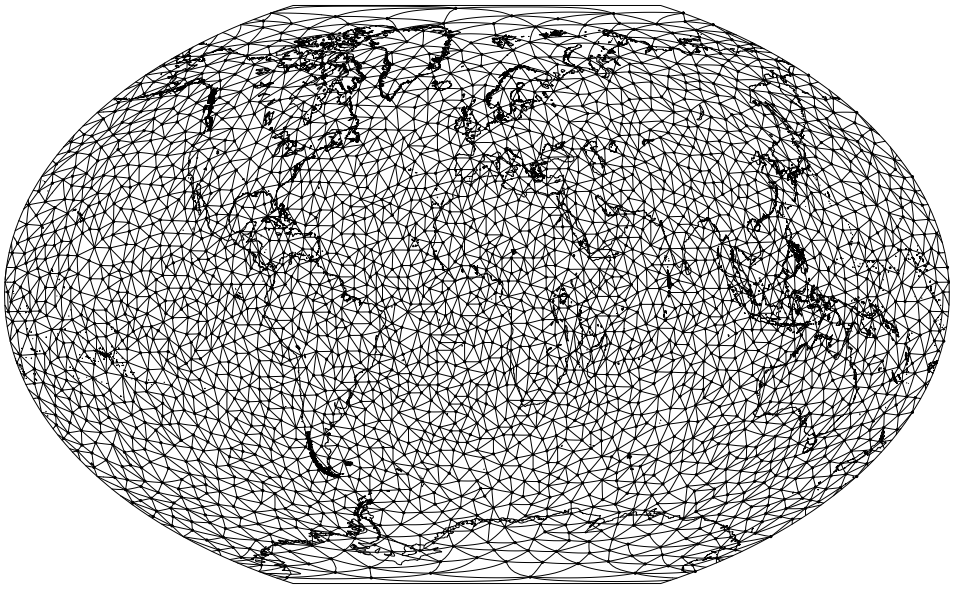

Figure 20: Subgrid $\mathcal{S}_1^c$ with a resolution parameter $\rho = 8$ ($n_1^c = 2827$)

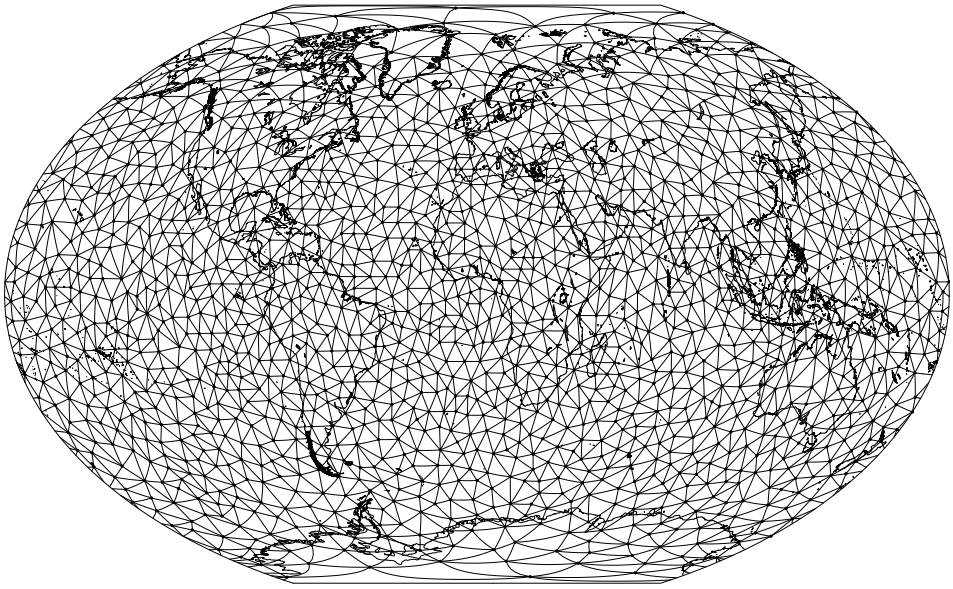

Figure 21: Subgrid $\mathcal{S}_1^c$ with a resolution parameter $\rho = 6$ ($n_1^c = 1590$)

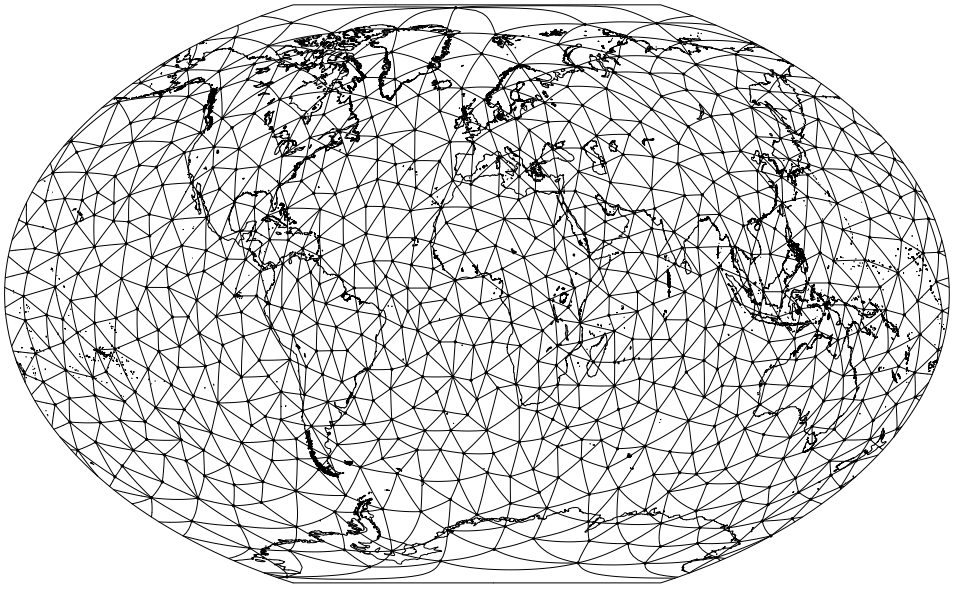

Figure 22: Subgrid $\mathcal{S}_1^c$ with a resolution parameter $\rho = 4$ ($n_1^c = 706$)

A coarser grid results in larger interpolation errors, so that the convolution function on the full grid $\mathcal{S}^f$ looks more "bumpy". Figures 23 and 24 show a comparison of convolution functions for a spectral method and for the NICAS method with a decreasing resolution.

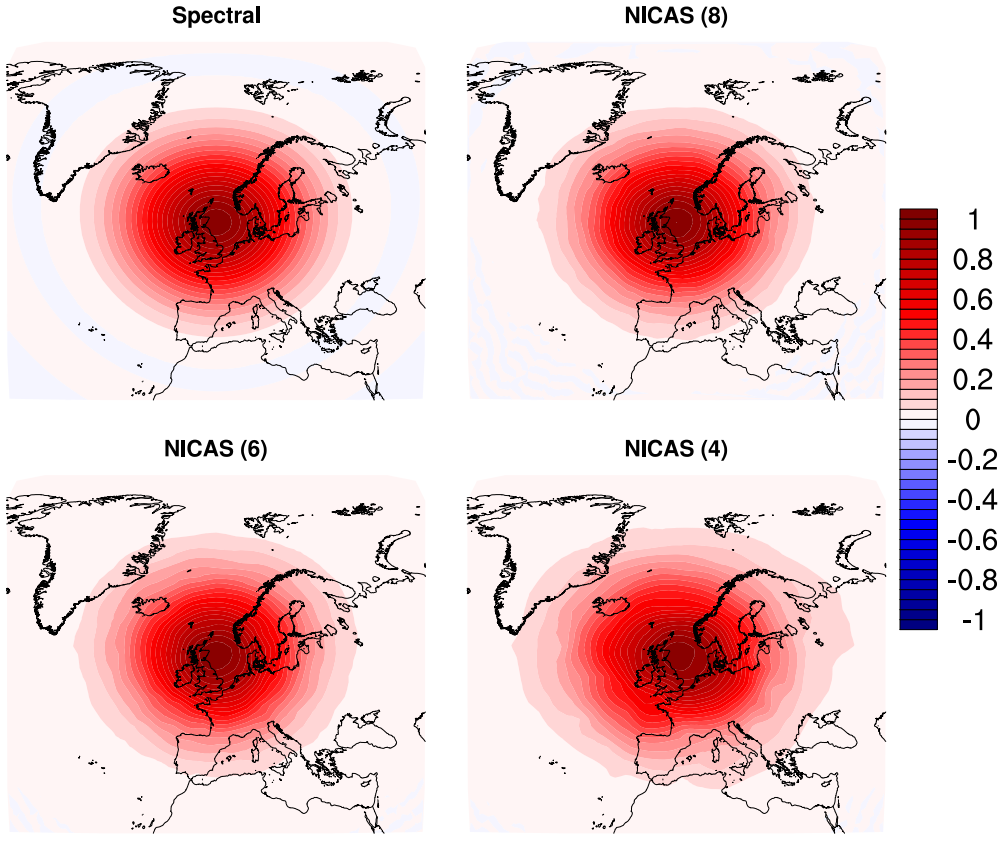

Figure 23: Horizontal convolution function for a spectral method and for the NICAS method with a decreasing resolution ($\rho = 8$, 6 and 4).

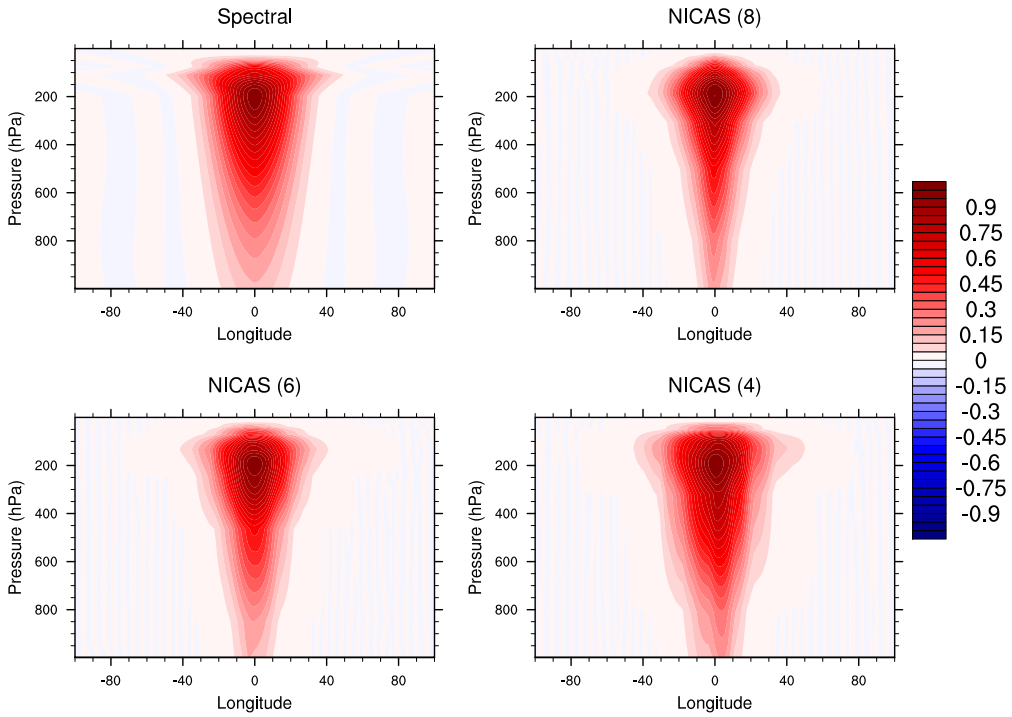

Figure 24: Vertical convolution function for a spectral method and for the NICAS method with a decreasing resolution ($\rho = 8$, 6 and 4).

References

Gaspari G, Cohn SE. 1999. Construction of correlation functions in two and three dimensions. *Quarterly Journal of the Royal Meteorological Society* **125**(554): 723–757.