Variable resolution mesh stretching functions

Christine Johnson

March 16, 2026

Regional models require the capability to use a variable-resolution or stretched mesh. This has a high-resolution interior region that is surrounded by a low-resolution outer region, with a stretching region in between. In the stretching region the cell size gradually increases from the high-resolution size to the low resolution size.

Variable resolution meshes allow high-resolution regional models to be nested in low-resolution global models without the need for double nesting (a nested model inside a nested model) even when there is a large ratio between the resolutions of the regional and global models . Without variable resolution meshes, large imbalances between the interior and the LBCs may dominate the solution, creating inaccuracies, instabilities and failure of the model run.

1 Current method

Currently the LFRic mesh tools app includes a planar mesh stretching method that is based on the UM variable resolution technique. This was added in ticket `https://code.metoffice.gov.uk/trac/lfric/ticket/3180` . The cell size of the stretch region is defined as

$$\Delta x_i = r^i \Delta x_{\text{inner}} \tag{1}$$

where i is the cell number in the stretch region and the inflation factor r is calculated such that $r^N \Delta x_{\text{inner}} = \Delta x_{\text{outer}}$ and N is the number of cells in the stretch region.

Advantages:

1. Using a method that is based on the UM technique means that it is possible to create an LFRic variable resolution mesh that matches a UM variable resolution grid. This allows um2lfric to create the initial and LBC data, using a 1-to-1 matching between LFRic and the UM.

Disadvatages:

1. Even though the idea is quite simple, the code in both the UM and LFRic is quite complicated. This makes it difficult to manage and navigate.
2. There are fundamental differences between the UM and LFRic that mean that an exact one-to-one matching is not possible. The UM u-points are half way between the p-points, but in LFRic the cell-centres (equivalent to p-points) are half-way between the cell-nodes (equivalent to u-points). To try to combat this, in LFRic it is possible to choose whether the stretching should be applied to the cell-nodes, cell-centres or p-points. p-points should create the closest mesh to the UM, however there are still some discrepancies.
3. To ensure that the mesh has a uniform resolution in the interior and outer, the cell size does not increase uniformly in the stretch region. This can create unwanted effects in the stretch region, as noted by Dave Lee. See https://code.metoffice.gov.uk/trac/lfric/attachment/ticket/4563/mesh-stretching-comparison.png
4. The stretching is not a true transform. A true transform would take a unit mesh and then apply a calculation to each coordinate to create a new coordinate. In this stretching the coordinates are created by incrementing the coordinate by the cell size. Whilst this does not cause a problem with the final output, it does not give a clean 'mesh pipeline' approach and it is not easy to invert (should that be required).

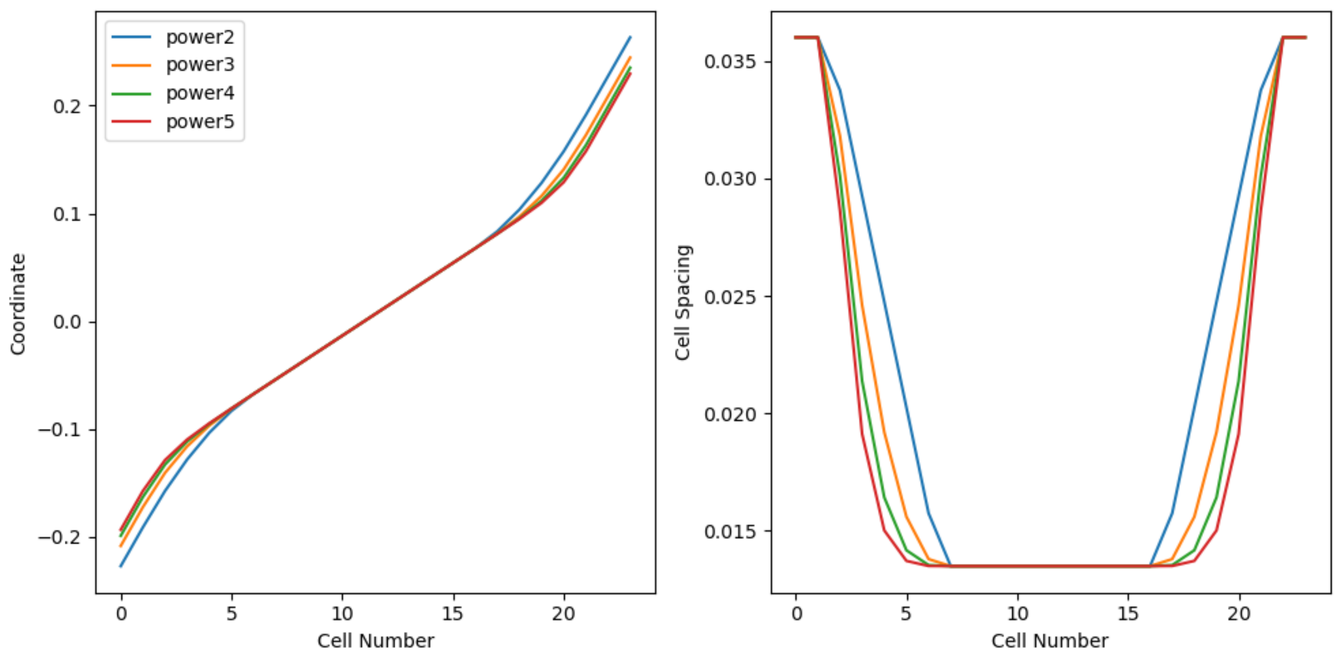

Figure 1: Example of the coordinates and cell spacing for a variety of powers p (poly power in the configuration). All using the same configuration input values (cell size inner 0.0135, cell size outer 0.036, n cells outer 2, n cells stretch 5, edge cells 24).

2 New method

For RAL4, the LFRic regional model will be nested in an LFRic global model, with the initial and LBC data being created using lfric2lfric. This means that there is no longer a requirement for the LFRic mesh to match the UM. And therefore it is possible to use new stretching functions.

Requirements:

1. Simple code - shorter code that is easier to navigate and manage.
2. A true transformation that satisifies $y_i = T(x_i)$ where x_i is the coordinate of the unit mesh, y_i is the coordinate of the stretched mesh in physical coordinates. This allows the stretching to be applied within a series of transformations i.e. A mesh pipeline that starts with a unit mesh, then applies stretching, then applies rotation.(https://code.metoffice.gov.uk/trac/lfric/wiki/LFRicInfrastructure/Mesh/Tools/GeneratorRestructuring).
3. A one-to-one (injective) function that allows the inverse transformation $x_i = T^{-1}(y_i)$ to easily be defined if required. (This may be required for data assimilation with Jedi).
4. The transformation should be applied to the mesh nodes, as currently designed in the mesh tools app.
5. A symmetrical function - so that the stretching at the Eastern edge is the same as the stretching at the Western edge.
6. Use the current inputs: cell size (outer and inner), and total number of cells, and number of cells in outer and stretch regions.

3 Polynomial stretching transformation

We define a stretching transformation that is described using polynomials. In the outer and inner region this is a simple straight line, with the gradient determining the cell size. In the stretch region a higher order polynomial (e.g. a quadratic, or cubic) is added to gradually blend from the inner to outer region. The parameters are chosen so that the cell sizes match at the points between each region, and so that the transformation function is continuous (with no jumps in the coordinates).

Given the following inputs:

1. Inner and outer cell sizes, Δy_{inner} and Δy_{outer}.
2. Total number of cells $2N$.
3. Number of cells in stretch and outer regions, N_s and N_o.
4. The polynomial power (an integer) p.

and a unit mesh with $2N$ cells, x defined on $[-1, 1]$, with cell spacing $\Delta x = 1/N$.

We consider that the mesh is symmetrical, so only consider one side and then apply the same on the transformation on the other side. This means we only need to consider the N points defining $[0, 1]$.

Use the values of N_s and N_o to define the left hand point x_L as the point between the inner and stretch and the right hand point x_R as the point between the stretch and outer.

Define the stretching function transformation T on x in $[0, 1]$

$$\begin{aligned} &\text{Inner:} && y = bx \\ &\text{Stretch:} && y = a(x - x_L)^p + bx \\ &\text{Outer:} && y = y_R + c(x - x_R) \end{aligned} \tag{2}$$

where the parameter y_R is the value of y in the stretch region at the point x_R. $y_R = a(x_R - x_L)^p + bx_R$. (The use of y_R allows the function to be continuous at x_R (only in y and not y' but y is sufficient). And by definition, the function is continuous at x_L in all derivatives.)

Calculate the parameter b so that the gradient of y with respect to x matches the ratio between the prescribed cell spacing in the inner region and that of the unit mesh, $dy/dx = b$. This gives

$$b = \Delta y_{\text{inner}}/\Delta x. \tag{3}$$

Similarly, calculate the parameter c so that the gradient of y matches the ratio between the prescribed cell spacing in the outer region and that of the unit mesh, $dy/dx = c$. This gives

$$c = \Delta y_{\text{outer}}/\Delta x. \tag{4}$$

Calculate the parameter a so that the cell spacing in the stretch region at the point x_R matches the prescribed cell spacing in the outer region (and hence the gradients match too). In the stretch region, the gradient $dy/dx = pa(x - x_L)^{p-1} + b$, which should equal c (i.e. in the outer region) at the point $x = x_R$ so

$$a = \frac{c - b}{p(x_R - x_L)^{p-1}}. \tag{5}$$

4 Implementation

To allow this stretch function to be implemented, the mesh tools app has been modified to create a 'mesh pipeline' process.

1. Create a unit mesh
2. Apply the transformation to physical coordinates - this is a uniform resolution transformation for no stretching, and the stretching transformation otherwise.

The parameters a, b, c are calculated once. And then the coordinates of the unit mesh are looped over, and the transformation is applied to each coordinate, using the symmetrical property of the mesh.

```
for all i
    if x(i) > 0
        y(i) = T(x(i))
    else
        y(i) = -T(-x(i))
```

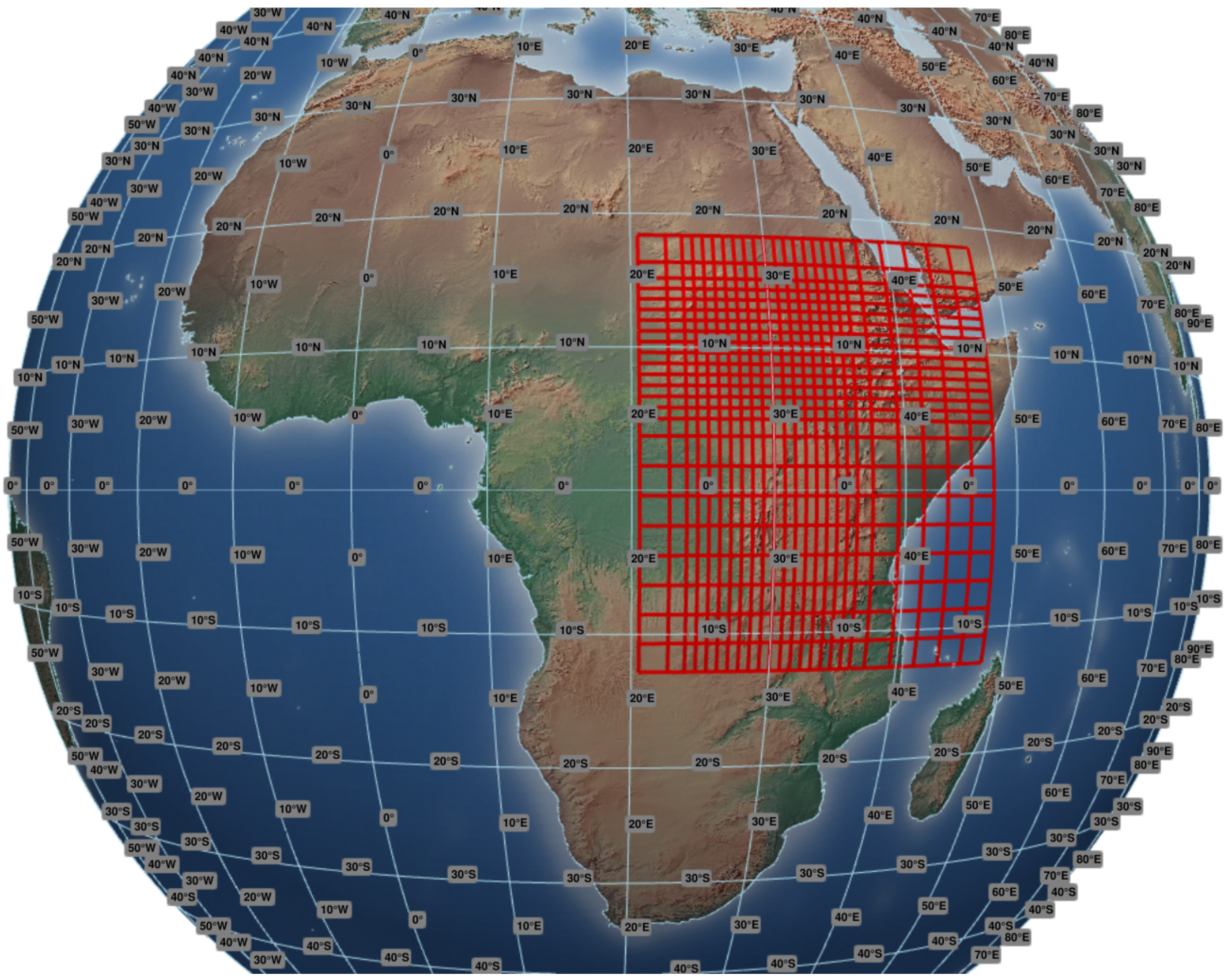

Figure 2: A non-symmetrical variable-resolution mesh.

5 Extension to Non-symmetrical meshes

To extend the implementation to non-symmetrical meshes, the idea is to apply the transformation on each boundary (North, South, East, West).

The number of cells in the stretch and outer regions are defined for each boundary. (`n_cells_outer_nsew` and `n_cells_stretch_nsew`).

For each boundary, first define the stretch transformation parameters. Then apply the transformation on the coordinates from the origin to the boundary. For the East boundary, this is all the coordinates $x(i) > 0$. For the West boundary, this is all the coordinates $x(i) < 0$.

After the stretch transformation is applied, the mesh is recentred so that the centre of the high-resolution interior is at (0,0). This is achieved by adding an 'offset' where the offset is calculated as:

$$\text{offset} = \frac{dx}{2}\left(\left(N^O(E) + N^S(E)\right) - \left(N^O(W) + N^S(W)\right)\right) \tag{6}$$

where $N^O(E)$ and $N^O(W)$ are the number of cells in the outer region for the East and West boundaries, and $N^S(E)$ and $N^S(W)$ are the number of cells in the stretch region for the East and West boundaries.

After the mesh is recentred on (0,0), it is recentered on the domain centre provided in the configuration - as would be done for a uniform mesh.

An example non-symmetrical mesh is shown in Fig. 2, which uses `edge_cells_x = 24`, `edge_cells_y = 24`, `domain_centre = 30.0, 10.0`, `cell_size_inner = 0.0135,0.0135`, `cell_size_outer = 0.036,0.036`, `n_cells_outer_nsew = 1, 8, 5, 1`, `n_cells_stretch_nsew = 1, 1, 1, 1`, `poly_power = 3` .

6 Inverse Transformation

For completeness, it is useful to examine the inverse transformation. This is not required at the moment and so not coded. But it may be required in the future.

To create the inverse of the transformation one needs to find a formula for x in terms of y. This should be possible analytically for $p = 2$ - but would be harder for higher powers. The relevant parameters such as where the stretch region is y_L, y_R and the parameters a,b,c would be required metadata to be stored in the mesh file. This is similar to the rotation metadata that is currently stored in the mesh files to allow the inverse of the rotation transformation to be calculated.

For example, for $p = 2$ in the stretch region

$$\begin{aligned} y &= a(x - x_L)^2 + bx \\ &= a(x^2 - 2x_L x + x_L^2) + bx \\ 0 &= x^2 + (b/a - 2x_L)x + (x_L^2 - y/a) \end{aligned} \tag{7}$$

Using the quadratic formula, and taking the positive root,

$$\begin{aligned} x &= \frac{1}{2}\left(-(b/a - 2x_L) \pm \sqrt{(b/a - 2x_L)^2 - 4(x_L^2 - y/a)}\right) \\ &= -(\frac{b}{2a} - x_L) \pm \sqrt{(\frac{b}{2a} - x_L)^2 - (x_L^2 - y/a)} \\ &= x_L - \frac{b}{2a} + \sqrt{(\frac{b}{2a})^2 - \frac{b}{a}x_L + \frac{y}{a}} \end{aligned} \tag{8}$$

And in the inner region

$$x = y/b \tag{9}$$

and outer region

$$x = x_R + \frac{1}{c}(y - y_R) \tag{10}$$