Feature tracking in high–resolution regional climate data.

Neil R. Massey[a]

[a] *Environmental Change Institute, University of Oxford, Oxford University Centre for the Environment, South Parks Road, Oxford, OX1 3QY, UK*

Abstract

In this paper, a suite of algorithms are presented which facilitate the identification and tracking of storm–indicative features, such as mean sea-level pressure minima, in high resolution regional climate data. The methods employ a hierarchical triangular mesh, which is tailored to the regional climate data by only subdividing triangles, from an initial icosahedron, within the domain of the data. The regional data is then regridded to this triangular mesh at each level of the grid, producing a compact representation of the data at numerous resolutions. Storm indicative features are detected by first subtracting the background field, represented by a low resolution version of the data, which occurs at a lower level in the mesh. Anomalies from this background field are detected, as feature objects, at a mesh level which corresponds to the spatial scale of the feature being detected and then refined to the highest mesh level. These feature objects are expanded to an outer contour and overlapping objects are merged. The centre points of these objects

Email address: `neil.massey@ouce.ox.ac.uk` (Neil R. Massey)

are tracked across timesteps by applying an optimisation scheme which uses of five hierarchical rules. Objects are added to tracks based on the highest rule in the scheme they pass and, if two objects pass the same rule, the cost of adding the object to the track. An object exchange scheme ensures that adding an object to a track is locally optimal. An additional track optimisation phase is performed which exchanges segments between tracks and merges tracks to obtain a globally optimal track set. To validate the suite of algorithms they are applied to the ERA-Interim reanalysis dataset and compared to other storm–indicative feature tracking algorithms.

Keywords: Storm tracking, Hierarchical mesh, Regional climate modelling, Sparse grid

1. Introduction

The automated tracking of extra tropical cyclones, by identifying and tracing the evolution of storm–indicative features in gridded meteorological data is an active and challenging research topic, with over 15 research groups having developed their own algorithms (Neu et al., 2013). Although these algorithms use different methods to identify and track the features, they all split the problem into two distinct phases. The first is the identification of the storm–indicative features in the gridded data. The second phase is the tracking of these features across timesteps, associating features in one timestep with features in subsequent timesteps so as to track the feature as it moves through time and across a domain, which is known as the corre-

spondence problem (Post et al., 2003). The analysis of the resulting feature tracks, which include acception or rejection of some tracks based on inclusion criteria may be performed before calculating the track statistics.

A common problem facing all tracking algorithms stems from the representation of data on a regular latitude–longitude grid. This causes grid boxes to become smaller toward the poles, in effect increasing their resolution and decreasing the spatial scale that features occur at. In addition, the data exhibits a singularity at the poles, with the first and last row of data having the same value, which makes searching for localised features difficult. Tracking algorithms overcome these problems in a number of ways. To correct the spatial discrepancies, the data may be smoothed by a Cressman filter (Murray and Simmonds, 1991; Sinclair, 1994), transformed to spectral space and truncated at a wavenumber (Hodges, 1994; Benestad and Chen, 2006), convolved with a filter (Hewson and Titley, 2010; Rudeva and Gulev, 2007) or regridded (Serreze, 1995; Wang et al., 2006). In order to account for the pole singularities, the data may be reprojected to a stereographic polar projection (Murray and Simmonds (1991); Hoskins and Hodges (2002, 2005)), at the expense of only being able to track storms in one hemisphere at a time.

Massey (2012) shows that both of these problems can be overcome by regridding the latitude–longitude data to a hierarchical triangular mesh obtained by repeatedly sub–dividing an icosahedron contained within the unit sphere. Constructing the icosahedron so that a centroid of a triangle corresponds to the locations of the poles ensures that there are no singularities.

This paper extends and improves Massey (2012) to enable the regridding of regional climate data to a sparse version of the hierarchical triangular mesh. Improvements to both the identification of storm–indicative features and their tracking are presented and incorporated into a new suite of regridding, feature identification and tracking software. To validate the methods, the software is applied to reanalysis data compared to other tracking algorithms, and is also applied to output from a high–resolution (50km) regional climate model.

2. Regional hierarchical mesh generation

2.1. Generating the mesh

Generation of the mesh commences with the construction of an icosahedron bound by the unit sphere and rotated so the poles lie at the centroid of a triangle. The coordinates of this icosahedron are provided in Table 1 of Massey (2012). The vertices of the triangles are stored in a point cloud and the indices of the point cloud for the vertices of the triangles are stored in an array of 20 quad trees, as in Massey (2012). In this revised mesh generation scheme, the user is required to supply an example of the data they wish to track features within, in the form of a netCDF file with latitude and longitude dimensions. This can be defined for regions including hemispheres or the entire globe, either on a regular or rotated pole grid.

For the purpose of the mesh generation and, later, the regridding of data to this mesh, grid points in the latitude–longitude grid are taken to be at the

centre of the grid box, and the grid box is assumed to have the same value within the grid box. No interpolation to the triangular mesh is attempted and a triangle value is taken to be at the centroid of the triangle and throughout the triangle.

The grid of latitude–longitude coordinates are converted to 3D Cartesian coordinates and then assigned to one of the 20 triangles in the icosahedron by way of a point inclusion test (Equation 5 in Massey (2012)), or a nearest point test if the point inclusion test fails. Each of these 20 triangles is split into 4 sub triangles if it contains a point, by projecting the midpoints of each triangle side onto the unit sphere and adding to the point cloud. The four new triangles are added to the quad tree as children of the initial triangle. Additionally, the point inclusion test is applied to the latitude–longitude coordinates that are within the original triangle to determine which child triangle they should be assigned to. This splitting process continues iteratively until all triangles contain one point or a maximum mesh level is reached. Figure 1 shows this iterative process of subdividing the mesh on a Dymaxion projection (Gray, 1994, 1995) and Figure 2 contains a pseudo–code algorithm of the splitting process. Each subdivision of a triangle is referred to as a level in the mesh, the 20 base triangles are at level 0, a triangle that has been split once is level 1, twice level 2, etc.

For each level in the mesh each triangle has assigned to it the indices of the original gridpoints of the latitude–longitude coordinates which enables very efficient indexing when regridding data and also allows the representation

of the data at different spatial scales. Finally, point and edge adjacency relationships between triangles at the same level in the mesh are calculated as in Massey (2012) for both point adjacency and edge adjacency.

The mesh generation only needs to be performed once for a dataset as the mesh is written to a file and read in by the regridding and feature identification algorithms.

2.2. Labelling scheme

The revised method follows a similar labelling scheme to Massey (2012), but stores the label as a single 64–bit wide integer, rather than a string. Each of the initial 20 triangles is labelled with an integer from 1 to 20. The triangles that result from a subdivision have a 1,2,3 or 4 added to them after multiplication by $10^{(parent\ triangle\ level+2)}$. Using an integer allows for efficient comparisons between labels, especially when checking for label equivalence. Additionally, labels at a higher level in a mesh will be greater than those labels at a lower level in the mesh. Traversing the tree to a triangle label is easily performed by determining which triangle to reference at each level in the mesh by a combination of modulo and division operations. Figure 3 illustrates the labelling scheme for a single initial triangle.

3. Regridding data to the mesh

3.1. Storing the data on the mesh

The mesh will be sparse when applied to regional data and so the indexing scheme of Massey (2012) cannot be used. However, a simpler and more

efficient indexing method can be used where each triangle maintains an index into a data storage array. The 20 initial triangles will have indices of $0 \ldots 19$ and, as the mesh generation algorithm splits triangles, it maintains a count of the total number of triangles that have been produced so far. When a triangle is split into four child triangles the new triangles are assigned the count, count+1, +2, and +3. The count itself is then incremented by 4. To store the regridded data, a two–dimensional array is required, one dimension equal to the number of timesteps that are to be regridded, and the other dimension equal to the number of triangles in the mesh. The mesh is fixed over time and so these indices are used in the regridding algorithm to assign data to a triangle for every timestep.

3.2. Regridding algorithm

Due to the user supplying an example of the data they wish to regrid, the point-inclusion test is performed during the mesh construction phase and at each level each triangle stores the indices of the latitude–longitude points which fall within the triangle. A triangle at level 0 will contain all the points which its children (and grandchildren and great-grandchildren etc.) contain. Regridding the data requires a loop through every triangle in the mesh at every level, recovering the indices of the original gridpoints, reading the data to be regridded using these indices and then assigning the data to the triangle. For triangles which have many grid points, the weighted area mean is taken for the data, as in Massey (2012).

Applying the regridding algorithm at every level of the mesh ensures that the regridded data is available at different spatial scales. This enables the storm–indicative feature detection phase to search for features of a particular spatial scale by searching at a level in the grid whose triangle area corresponds to that scale, and then to refine the search into higher levels of the grid to determine the exact footprint and location of the storm–indicative feature.

4. Objective feature detection on the mesh

Storm–tracking methods are sensitive to the type of background field removed prior to feature identification (Anderson et al., 2003). Massey (2012) uses a hemispherical mean as the background field, which leads to a low spatial density of storm–indicative features. The revised method uses the data at a lower level in the regridded data, which is a lower resolution version of the data at higher levels. This is analogous to the spectral spatial filter of Anderson et al. (2003), which removes the large spatial scale flow, maintains the spatial density of features and preserves the intensities of the features.

4.1. Background field removal

The feature detection process commences with the removal of the background field, which is the data at a lower mesh level (the *background level*) in the regridded mean sea–level pressure (MSLP) data. Subtracting data at this level from the data at the level which features are identified (the *identification level*) produces the anomaly from the large spatial scale flow

and negative values in the resulting data can be identified as low pressure systems.

Subtracting the value in a triangle in the *background level* from triangle values in the *identification level* produces sharp edges in the resulting data which occur at the edge of the triangles in the background level and lead to the erroneous identification of low pressure systems along the edges of those triangles. In order to avoid this edge–effect, the data at the *background level* is smoothed to all higher levels. The smoothed data is then subtracted from the original data at all higher levels and rounded to the nearest H hPa, which aids the feature identification process as MSLP values that are within a H hPa contour are assigned the same value.

4.2. Feature point detection

Features are detected as anomalies from the large spatial scale flow, by using one of the adjacency lists to consider the triangles surrounding each triangle at the *identification level*. At this level, features are identified by comparing the value of a triangle with the values of the adjacent triangles. If the triangle value is less than the surrounding triangle values, and the value is also less than a user defined maximum difference from the large spatial scale flow, then the triangle is considered as a feature. To create the feature object, the labels of the descendants of the initial triangle ($\mathbf{T_i}$) at the highest level in the mesh (the *refinement level*) are added to a list and the location of the object is set to be the centroid of $\mathbf{T_i}$.

Not all of the triangles at the *refinement level* in the list of labels should belong to the object, as they have anomalies that are larger than the largest anomaly in the object ($\left[\mathbf{max}_{anom}\right]$). To account for this, triangle labels are removed from the list of labels if the value of the triangle is not equal to, or less than $\left[\mathbf{max}_{anom}\right]$.

4.3. Feature object detection

At this stage, the feature object identified will not extend beyond the bounds of the triangles at the *identification level*, even though the feature itself may extend into other triangles at the *refinement level*. To counteract this, an object growing routine is employed.

The feature object is grown by examining the values of the triangles surrounding it, and if a surrounding triangle value is within one contour level of $\left[\mathbf{max}_{anom}\right]$ then it is added to the feature object by appending its label to the list of triangle labels. This process is repeated until no extra triangles can be added to the object, as none of the surrounding triangles are within the contour level. An added restriction here is that objects are not allowed to grow to be more than 1000km in radius. This has the consequence that non–closed systems will have an artificial closure 1000km away from the triangle with $\left[\mathbf{max}_{anom}\right]$.

After the object growing routine has completed, any overlapping objects are merged, with the same 1000km restriction as in the feature growing phase. If a triangle in an object is 1000km or less away from the central triangle

of an overlapping object then it is added to the object. If it is more than 1000km away then it is retained in its current object. Objects are deemed to overlap if one or more triangles in an object also occur within the other object, or a triangle in one object is adjacent to a triangle in the other object. Objects are merged by appending triangle labels to one object and deleting the other.

4.4. Deriving the feature point from the object

The feature tracking algorithm in Section 5 requires a single point, as well as the object definition (the list of triangles in the object), to track the evolution of the feature objects across multiple timesteps. To derive this point a weighted average of the locations of the triangles in the object is used:

$$\begin{aligned} \overline{\boldsymbol{P}} &= \frac{1}{\sum_{i=0}^{n} w_i} \sum_{i=0}^{n} w_i \boldsymbol{P_i} \\ w_i &= 1 - \frac{V_i}{\left[\underset{\boldsymbol{anom}}{\mathbf{max}}\right]} \end{aligned}$$

where $\overline{\boldsymbol{P}}$ is the central feature point of the object, $\boldsymbol{P_i}$ is the location of the centroid of the ith triangle in the object, V_i is the value of the ith triangle in the object, n is the number of triangles in the object and $\left[\underset{\boldsymbol{anom}}{\mathbf{max}}\right]$ is the maximum anomaly in the object. The point $\overline{\boldsymbol{P}}$ is then converted to latitude and longitude coordinates using the standard transformation. The object also records the intensity of the MSLP low, which is $\left[\underset{\boldsymbol{MSLP}}{\mathbf{min}}\right]$, the minimum

MSLP value in the object, and the delta of the MSLP low, which is $\left[\underset{\boldsymbol{MSLP}}{\mathbf{min}}\right] - \left[\underset{\boldsymbol{MSLP}}{\mathbf{max}}\right]$. Figure 4 provides a summary of the feature detection routine.

5. Feature tracking

After the feature identification routine of Section 4 has been applied to the regridded data, a list of feature objects exist for each timestep. It is the job of the feature tracking algorithm to associate these objects across the timesteps to track the progress of the features as they form, move and dissipate. The revised feature tracking method in this paper deviates from that in Massey (2012), which requires the assigning of weights to four terms in a cost function, whereas the new method uses a series of hierarchical rules to assign objects to tracks.

5.1. Tracking rules

The rules to determine whether a feature object should be added to an existing track are based on five criteria, given below. Three of the criteria (5.1.1, 5.1.3, 5.1.5) use the feature point, derived from the feature object in Section 4.4, one (5.1.2) uses the object definition (the list of triangle labels within the feature), and one uses a property of the object (5.1.4).

5.1.1. Distance rule

The user can specify a maximum distance between the last object in a track ($\mathbf{O^{t-1}}$) and the candidate object ($\mathbf{C^{t}}$), which should be based upon the number of hours per timestep in the source data (h) and how far the

object is expected to move within that time period. Extra–tropical cyclones in the Northern Hemisphere have a mean propagation speed of up to $30ms^{-1}$ (Hoskins and Hodges, 2002), so it is suggested to set this search radius to be upwards of $125h$km.

An adaptive constraint (Hodges, 1999) on this search radius is calculated from the geostrophic wind steering vector. If the magnitude of this vector multiplied by h is greater than the user input search radius, then the search radius is set to be this value. The distance between the last track object and the candidate track object is:

$$\Delta = \mathcal{H}(\mathbf{C^t}, \mathbf{O^{t-1}}) \tag{1}$$

where $\mathcal{H}$ is a function that calculates the haversine distance (Sinnott, 1984) between $\mathbf{O^{t-1}}$ and $\mathbf{C^t}$. The haversine distance calculates distances over the skin of a sphere, rather than through it.

5.1.2. Overlap rule

An advantage of identifying feature objects, rather than single feature points, is that their spatial characteristics can be used during the tracking phase. This is useful, in this context, to identify slow moving features which are blocked by high–pressure systems and may change direction sharply (Hodges, 1999). In order to assess this in terms of minimising a cost function

the rule is:

$$\beta = 100 - \mathcal{O}(\mathbf{C^t}, \mathbf{O^{t-1}}) \tag{2}$$

where $\mathcal{O}$ is a function that measures the percentage of overlap between the candidate object $\mathbf{C^t}$ and the object $\mathbf{O^{t-1}}$.

5.1.3. Steering rule

Using a steering vector to predict the motion of a feature based on physical characteristics of the field can reduce the number of erroneous point assignments in a tracking algorithm (Murray and Simmonds, 1991). Here the geostrophic wind in spherical coordinates (Andrews, 2000), approximated by fitting splines to the geopotential height field and taking the derivatives of the splines, is used.

The steering rule uses the geostrophic wind and the central point of $\mathbf{O^{t-1}}$ to predict the position of the next object in the track:

$$\begin{aligned} P^t_\lambda &= O^{t-1}_\lambda + \frac{360 u_g N_H}{F_\phi} \\ P^t_\phi &= O^{t-1}_\phi + \frac{180 v_g N_H}{F_R} \\ F_\phi &= 2\pi E_R \cos(\phi) \end{aligned} \tag{3}$$

where $P^t_{\lambda,\phi}$ is the projected latitude–longitude point, $O^{t-1}_{\lambda,\phi}$ is the central point of $(\mathbf{O^{t-1}})$, (u_g, v_g) is the geostrophic wind vector, F_R is the meridional circum-

ference of the Earth, F_ϕ is the zonal circumference of the Earth at latitude ϕ and E_R is the radius of the Earth.

The cost function is the difference in bearing between the projected point ($P^t_{\lambda,\phi}$) and the last track point ($O^{t-1}_{\lambda,\phi}$) and between $O^{t-1}_{\lambda,\phi}$ and the candidate point ($C^t_{\lambda,\phi}$):

$$\Psi = \angle(O^{t-1}_{\lambda,\phi}, C^t_{\lambda,\phi}) - \angle(O^{t-1}_{\lambda,\phi}, P^t_{\lambda,\phi}) \tag{4}$$

Where $\angle(A, B)$ is the bearing between latitude–longitude coordinates A and B, as defined by Jennings (1994). Objects with bearing differences of $-90°$ to $90°$ are considered to have passed the rule.

5.1.4. Intensity rule

Using the value of the MSLP minimum associated with the object (the intensity) is useful when tracking very deep depressions where smaller depressions are introduced into the domain, as it encourages the correct (deep) depression to be allocated to the track, even if allocating the smaller depressions would meet some of the rules. The intensity cost is defined as:

$$\chi = \mathcal{I}(\mathbf{C^t}) - \mathcal{I}(\mathbf{O^{t-1}}) \tag{5}$$

where $\mathcal{I}$ is a function that returns the intensity value of the object. Objects with intensity differences less than or equal to 100hPa are considered to have passed the rule.

5.1.5. *Curvature rule*

The curvature rule aims to maximise smoothness within tracks. Like the steering rule, it uses the definition of the bearing as defined by Jennings (1994). The curvature between a candidate point and the track is:

$$\mathcal{C} = \angle(O^{t-1}_{\lambda,\phi}, C^{t}_{\lambda,\phi}) - \angle(O^{t-2}_{\lambda,\phi}, O^{t-1}_{\lambda,\phi}) \tag{6}$$

where $O^{t-2}_{\lambda,\phi}$ is the central point of the penultimate object in the existing track. Candidate objects with curvatures of between $-90°$ to $90°$ are considered to have passed the rule, with a cost of:

$$\Xi = \alpha\mathcal{C}\mathcal{D} \tag{7}$$

where $\mathcal{D}$ is the distance cost, as defined in Equation 1 and α is a scalar which ensures the resulting cost is not dominated by the distance term. α is set to be 10^{-3}, so as to convert the distance in metres from Equation 1 into kilometres. This ensures that candidate objects that have a small curvature, but are far away from the last track object, are penalised equally, or more, than those candidate points which are closer but have a larger curvature.

5.1.6. *The hierarchy of rules*

The optimisation routine that forms the core of the tracking algorithm uses a hierarchy of rules:

5 Distance rule

4 Overlap rule

3 Steering rule

2 Intensity rule

1 Curvature rule

All candidate objects have to pass the distance rule requirement first, to ensure that the point is within the search radius before the other rules are considered. Using the hierarchy has the advantage that,during the optimisation process candidate objects can be compared to each other to see which has passed the highest rule. If two objects have passed the same rule, then the costs for both objects are calculated and the object with the lowest cost can be assigned.

5.2. Assigning objects to tracks

The tracking routine commences with constructing a new track for each feature object in the first timestep, i.e. $t = 0$. In subsequent timesteps, objects are added to existing tracks based on whether they pass the inclusion rules and subject to a local optimisation protocol. If an object cannot be be added to an existing track then a new track is created, with that object as the first object in the track.

The local optimisation routine adds feature objects in timestep t to tracks ending in timestep $t-1$. For each timestep t, all feature objects in t are added to a queue of objects. In turn, these objects at the front of the queue are assessed against the tracks that end in timestep $t-1$ ($\mathbb{T}_{0...n}^{a \leq t-1}$), using the rules

in Section 5.1. If an object passes the distance rule and at least one other rule then it is added to track $\mathbb{T}_i^{a \leq t-1}$ as the candidate object. It is possible that an object might pass the rules for multiple tracks, in which case it is added to the track for which it minimises the cost for the highest rule it has passed.

To avoid the ordering of objects influencing the tracking outcome (Hodges, 1999), feature objects that are still in the queue are allowed to replace the current candidate object if they are deemed to be more locally optimal. This is done by following the hierarchy of rules for the new object ($\mathbf{N^t}$) and the current candidate object ($\mathbf{C^t}$). If, after assessing $\mathbf{N^t}$ against $\mathbf{O^{t-1}}$ (the last object in a track), $\mathbf{N^t}$ passes a higher rule than that which $\mathbf{C^t}$ passes, then $\mathbf{N^t}$ becomes the candidate object and $\mathbf{C^t}$ is returned to the queue of feature objects so that it may be assigned to other tracks. If the two objects have both passed the same rule, then the costs for the highest rule are calculated for each object and the object with the lowest value is assigned as the candidate object.

During the optimisation procedure, objects may be assigned to a track, replaced, and then assigned to another track as the procedure continues until there is no more assignment of objects. If an object cannot be assigned to a track (as it does not pass any of the rules, or another object passes the rules with a lower cost) then a new track is started in timestep t, with the feature object as the first object in the track.

5.3. Track optimisation

The procedure in Section 5.2 produces a set of tracks that are locally optimal at each timestep. However, this approach may encounter several problems which prevent a globally optimal solution being found. These include the false assignment of an object to a track, the premature ending of a track where a feature object has not been identified in the next timestep, or the premature ending of a track where subsequent features in the track have erroneously been assigned to another track. To alleviate these problems, the track optimisation routine employs two strategies to find an optimal set of tracks:

- Track merging: extending a track which ends at timestep $t - 1$, by another track, which begins at timestep t or timestep $t + 1$.
- Track reassignment: splitting two tracks, which overlap in time, at timestep t and reconstructing up to three tracks where the combined cost of the tracks is less than the cost of the original two tracks

The optimisation process has the advantage over the track assignment routine of Section 5.2 that it operates on a near optimal set of complete tracks. Many of the tracks will contain more than one timestep and, therefore, a measure of track smoothness over a number of timesteps can be used as the cost function in the optimisation process. This cost is the mean curvature cost of the (up

to) five track objects centred around the timestep t:

$$\overline{\mathcal{C}^t} = \frac{1}{N} \sum_{s=t-2,s>0}^{s=t+2,s\leq L} \alpha \mathcal{C}^s \mathcal{D}^s \tag{8}$$

where $\mathcal{C}^s$ is the curvature at timestep s, as defined in Equation 6, $\mathcal{D}^s$ is the distance between the track object at timestep s as defined in Equation 1, L is the number of timesteps in the track and N is the number of track points in the sum which satisfy the equalities.

5.3.1. Track merging

Given two tracks, $\mathbb{T}_1^{a\leq r}$ and $\mathbb{T}_2^{b\geq s}$, there are two different cases for testing whether tracks can be merged, depending on the timestep in which $\mathbb{T}_1^{a\leq r}$ ends and the timestep in which $\mathbb{T}_2^{b\geq s}$ begins. If the last timestep in track $\mathbb{T}_1^{a\leq r}$ is $r = t - 1$, then the first case occurs when the first timestep in $\mathbb{T}_2^{b\geq s}$ is $s = t$ and the second case occurs when $s = t + 1$.

For the first case, a track ending at timestep $t-1$ ($\mathbb{T}_1^{a\leq t-1}$) may be merged with a track beginning at timestep t ($\mathbb{T}_2^{b\geq t}$) if the last feature object in $\mathbb{T}_1^{a\leq t-1}$ ($\mathbf{T}_1^{t-1}$) is within the search radius of the first object in $\mathbb{T}_2^{b\geq t}$ ($\mathbf{T}_2^{t}$) and that, by merging the tracks, the maximum allowed curvature is not exceeded. The curvature is measured using an adaptation of Equation 6:

$$\mathcal{C}'^t = \angle(S_{\lambda,\phi}^{t+1}, S_{\lambda,\phi}^{t}) - \angle(S_{\lambda,\phi}^{t}, T_{\lambda,\phi}^{t-1}) \tag{9}$$

where $S_{\lambda,\phi}^{t+1}$ and $S_{\lambda,\phi}^{t}$ are the points of the first two feature objects in $\mathbb{T}_2^{b\geq t}$.

The second case ($r = t - 1$ in $\mathbb{T}_1^{a \leq r}$, $s = t + 1$ in $\mathbb{T}_2^{b \geq s}$) occurs when a feature has been obscured by another feature, or if the identification routine fails to detect a feature object. Hodges (1994, 1995) use phantom feature points during the track optimisation to enable tracks to be continuous, even if feature points are missing. Here, the idea of phantom feature objects is used to predict a point at timestep t.

Two tracks, $\mathbb{T}_1^{a \leq t-1}$ and $\mathbb{T}_2^{b \geq t+1}$ are merged if a point projected forward from $\mathbb{T}_1^{a \leq t-1}$ to timestep t is within the search radius of a point projected backwards from $\mathbb{T}_2^{b \geq t+1}$. The projection is made using either the geostrophic wind (Equation 3) or a projection based upon the central points of the last (or first) two objects in the track:

$$\begin{aligned} T_\lambda^t &= T_\lambda^{t-a} + a(T_\lambda^{t-a} - T_\lambda^{t-2a}) \\ T_\phi^t &= T_\phi^{t-a} + a(T_\phi^{t-a} - T_\phi^{t-2a}) \end{aligned} \tag{10}$$

where $T_{\lambda,\phi}^t$ is the projected point at timestep t. a controls the direction of the projection – to project forwards, $a = 1$, to project backwards, $a = -1$. Whether the geostrophic wind or the two objects are used in the projection depends on which of the two projected vectors has the largest magnitude. To project backwards using the geostrophic wind, the sign of N_H is reversed in Equation 3.

Merging is achieved by adding a phantom feature object to track $\mathbb{T}_1^{a \leq t-1}$ at timestep t, which has the interpolation of object $\mathbf{T}_1^{t-1}$ and $\mathbf{T}_2^{t+1}$ as values

for the position (latitude and longitude), intensity and delta of the MSLP low, and then appending track $\mathbb{T}_2^{b \geq t+1}$ to the resulting track. Phantom feature objects have no triangle labels associated with them and are deemed to have passed no rules.

It may be the case that track $\mathbb{T}_1^{a \leq t-1}$ can be merged with multiple tracks, as there may be more than one track that begins at $t+1$ and whose projected point is nearer than the user defined search radius. If this occurs then the merged track which has the smallest mean curvature cost ($\overline{\mathcal{C}^t}$ in Equation 8) is selected.

5.3.2. Track reassignment

Due to the nature of the tracking algorithm, feature objects may be assigned to tracks in a sub-optimal way, with respect to the mean curvature cost (Equation 8). This occurs mostly in the first few timesteps of tracks where there are not enough objects in the track to make an optimal choice between two candidate track objects. To counter this a phase of the track optimisation process reassigns sections of track between the set of tracks produced by the tracking algorithm.

Two tracks, $\mathbb{T}_1^{S_1 \leq t \leq E_1}$ and $\mathbb{T}_2^{S_2 \leq t \leq E_2}$ are candidates for reassignment if they overlap in time – i.e. if the range $S_1 \leq t \leq E_1$ shares at least one value of t with the range $S_2 \leq t \leq E_2$. To determine whether a section of $\mathbb{T}_1^{S_1 \leq t \leq E_1}$ should be replaced by a section of $\mathbb{T}_2^{S_2 \leq t \leq E_2}$, for each overlapping value of t, the mean curvature cost of $\mathbb{T}_1^{S_1 \leq t \leq E_1}$ is calculated for t, as well as the mean

curvature cost of replacing the objects at $t, t+1$ and $t+2$ with the objects at those timesteps in track $\mathbb{T}_2^{S_2 \leq t \leq E_2}$. If this mean curvature is less than the previous value, then track $\mathbb{T}_1^{S_1 \leq t \leq E_1}$ is split at point t and three new tracks are created. The first of these is the compound track of $\mathbb{T}_1^{S_1 < t}$ up to timestep t and track $\mathbb{T}_2^{t \leq E_2}$ beyond timestep t. The second is the last part of track $\mathbb{T}_1^{t \leq E_1}$, after timestep t and the third is the first part of track $\mathbb{T}_2^{S_2 < t}$, before timestep t.

Optimising the tracks by splitting them may lead to two tracks being produced which are, in fact, part of the same track. To enable these tracks to be merged, the track optimisation process runs the track merging and track reassignment algorithms repeatedly until no tracks are merged, or no track sections are reassigned, or a maximum number of iterations is reached.

5.4. Post–processing of tracks

After the completion of the tracking algorithm, the tracks may be filtered on various criteria, including a minimum number of timesteps the track persists for, how far the track moves from genesis to lysis, the length of the track or the minimum value of MSLP reached along the track. This filtering is left to the user as post–processing is not directly provided by this suite of algorithms. Note that the splitting of tracks is not explicitly accounted for. If a storm feature splits into two distinct features, then one of these features will be assigned to the original track, whereas a new track will be created starting at the timestep at which the split occurred. There is no indication

in the tracking results that these two tracks are related.

6. Results

Validating a storm tracking algorithm is problematic as there is no universally agreed set of historical storm tracks with which to compare the output from a storm tracking algorithm (Neu et al., 2013). In this Section, the full process of storm–indicative feature tracking detailed in Sections 2.1 to 5 is applied to two different data sets and compared to the output from other tracking algorithms.

6.1. Application to ERA-Interim data

Neu et al. (2013) analyse the disparities between 15 different storm tracking procedures by applying the algorithms to the ERA-Interim reanalysis (Dee et al., 2011). Total cyclone centre densities for all 15 methods are compared, along with histograms of the track metrics. To compare its performance the procedure detailed in this paper is applied to the Northern hemisphere winter (DJF) ERA-Interim data from 1st January 1989 to 28th February 2009. The mean sea level pressure (MSLP) variable is used to identify features, with the geopotential height at 500 hPa (Z500) used to determine the steering vector.

After applying the tracking procedure to ERA-Interim, 43,310 tracks were identified of which 14,351 had a persistence of 24 hours or more. This is directly comparable to the number of tracks produced by the 15 algorithms

in Table 2 of Neu et al. (2013), where the number of tracks identified in DJF ranges between 5,700 and 20,500, with a mean of 12,393 and standard deviation of 4,504.

Figure 5 shows the density of cyclone centres per timestep and 1000km^2. This figure applies the same method of calculating the cyclone density as Figure 1 in Neu et al. (2013), and so is directly comparable with the results presented there. The storm tracking procedure performs as well as the 15 methods in Neu et al. (2013), qualitatively agreeing with the identification of the areas of high storm activity east of Greenland, around the Northern edge of the Scandinavian peninsula, the two centres in the North Pacific and on the leeward side of the Rocky mountains. The procedure presented here also identifies a large frequency of occurrence of storms over the Mediterranean, which Neu et al. (2013) note has a large discrepancy between methods. However, as noted in Neu et al. (2013) there is no attempt in their study to standardise the depression depth or intensity that a cyclone is required to pass the threshold of each individual algorithm. The algorithm presented here is somewhat inclusive, requiring only a depression of 2 hPa from the spatial scale flow to register as a storm–indicative feature.

Figure 6 shows the relative frequencies of three storm track metrics. This figure is directly comparable to Figure 3 in Neu et al. (2013), and the method presented here compares favourably with the 15 studied there. For intensity, the fraction of unit for each bin is within the standard deviation of the 15 other tracking methods. For lifetime, the same is true, even though the

standard deviations have a much narrower range than the intensity. Again, for propagation speed the method compares well with the 15 other methods, except here there is a higher occurrence of tracks with a speed between 0 and 10 km/hr.

7. Conclusion

This paper presents a suite of algorithms which regrid regional climate data to a sparse version of the hierarchical triangular mesh. This facilitates the identification of storm–indicative features by first removing the large spatial scale flow and then searching for anomalies from the background field. These anomalies are then tracked across timesteps to produce storm tracks. The use of a sparse mesh produces a very compact representation of the data and the hierarchical nature of the mesh allows the large spatial scale flow to be extracted directly from the data. Additionally, the feature objects can be refined to the highest level in the mesh in order to obtain the footprint of the storm–indicative feature. Applying the algorithms to the ERA-Interim reanalysis shows that the method produces comparable results to 15 other tracking procedures. The procedures described in this paper are particularly suitable for regional climate model output.

8. Obtaining the code

The algorithms are written in C++ and can be downloaded from GitHub by using the command:

```
git clone https://github.com/nrmassey/tri_tracker.git --branch CAGEO-version
```


Acknowledgements

This work was supported by a grant from the Risk Prediction Initiative of the Bermuda Institute of Ocean Sciences.

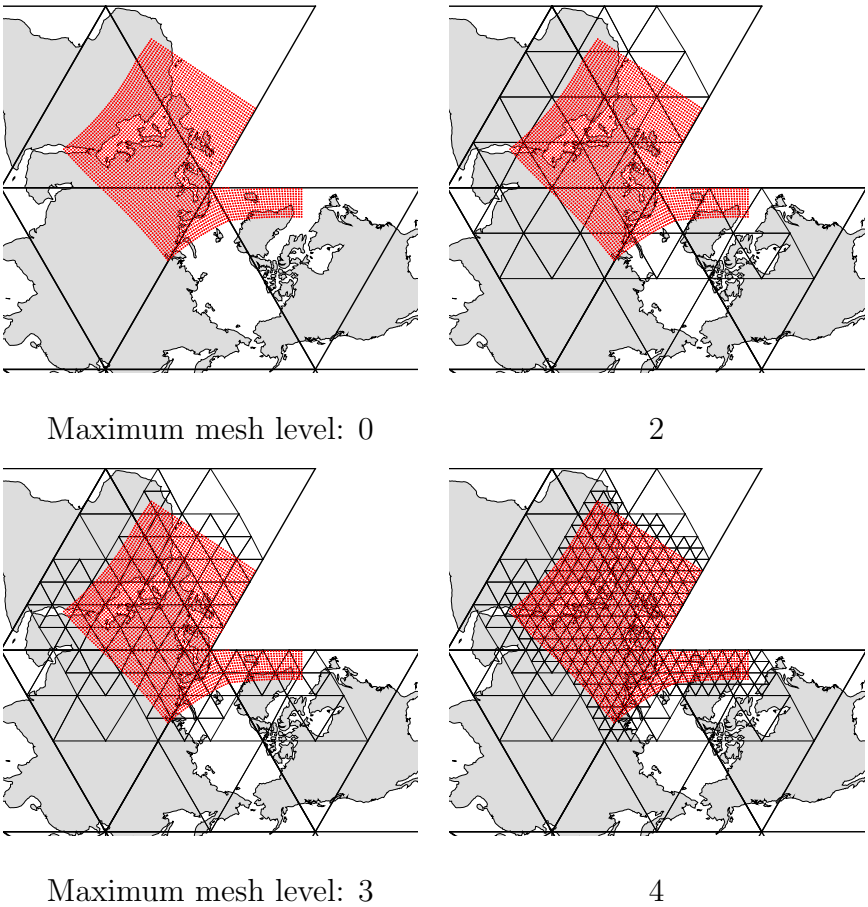

Figure 1: The repeated subdivision of the triangular mesh based on the grid from the HadRM3P regional climate model over Europe. Red dots show the latitude–longitude points of the HadRM3P grid and each panel shows the triangles at the maximum mesh level after the subdivision. Only triangles with at least one latitude–longitude point are subdivided per iteration. The triangular mesh is shown on a Dymaxion projection (Gray, 1994, 1995), which represents the world map on an icosahedron and so is ideal for displaying the hierarchical triangular mesh.

```
function split_triangle(T)
    if T.number_of_points == 0:
        return
                        create the three mid–points along the triangle edges
    P,Q,R = find_midpoints(A,B,C)
                        add the four child triangles (only first triangle shown)
    C_1 = T.add_child(A,R,P)
                                        distribute the points to the children
    for O_{φ,λ} in T.points:
        for i in 1...4:
            if C_i.point_in_triangle(O_{φ,λ}):
                C_i.add_point(O_{φ,λ})
                check whether each child should be split again (recursively)
    for i in 1...4:
        if C_i.number_of_points != 0:
            split_triangle(C_i)
end function

for O_{φ,λ} in input_grid:                add points to initial 12 triangles
    for T in initial_triangles:
        if T.point_in_triangle(O_{φ,λ}):
            T.add_point(O_{φ,λ})

for T in initial_triangles:               recursively subdivide the triangles
    split_triangle(T)
```

Figure 2: Outline of the algorithm used to generate the sparse triangular mesh from climate data. See Figure 1 in Massey (2012) for references to triangle vertices in the splitting algorithm.

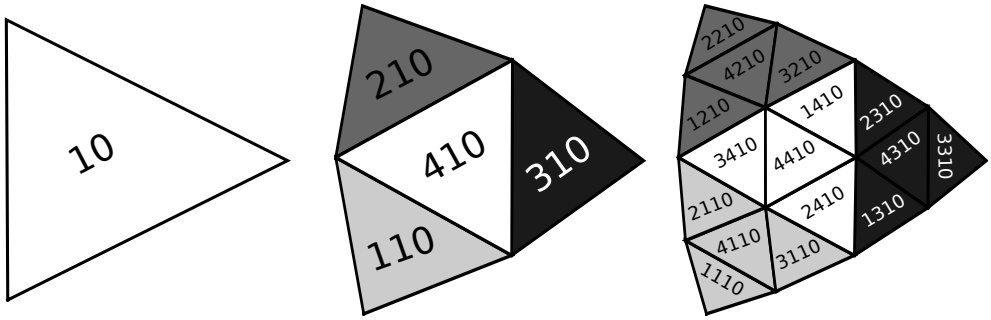

Figure 3: Illustrative example for the triangle labelling scheme for a single initial triangle. The 10th initial triangle is subdivided into 4 triangles and the label is added to the initial label according to Section 2.2. These triangles are further subdivided and labelled.

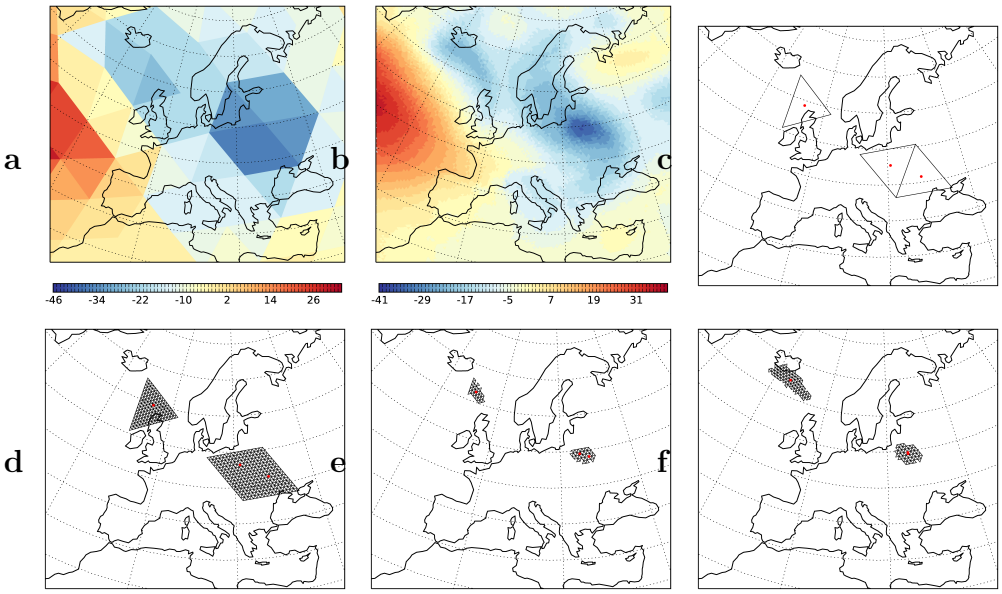

Figure 4: Illustrative depiction of feature identification routine. **(a)** MSLP anomaly from the background field at a single timestep at mesh level 3. **(b)** MSLP anomaly from the background field at mesh level 7. **(c)** MSLP minima detected at mesh level 3. **(d)** Feature object composed of triangles at level 7 descended from those triangles in **c** identified as minima at level 3. **(e)** Refinement of the triangles in the feature object in **d** by discarding triangles whose values are not within one contour level of the maximum anomaly in the object. **(f)** Feature objects after the growing of the objects to encompass all triangles whose values are within one contour level of the maximum anomaly. The objects are also merged. Note that two objects are detected in this example and that the red dots show the centroid of the triangles and (in **(f)**) the feature point position.

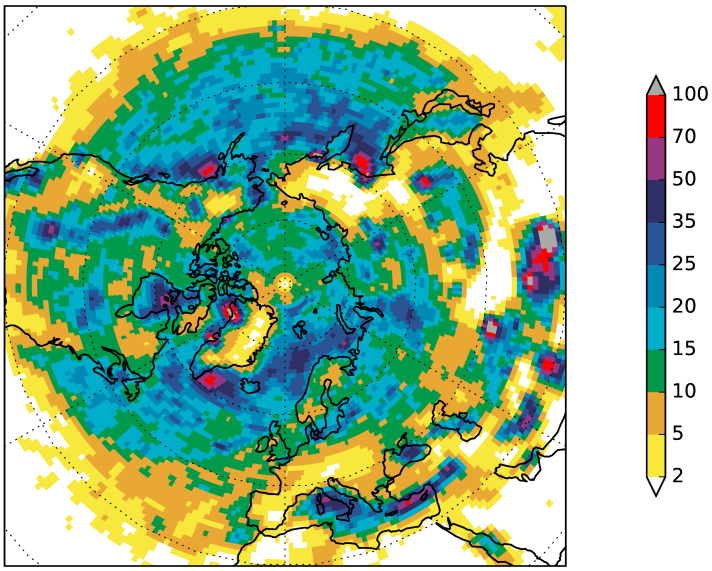

Figure 5: Track cyclone centre density from applying the tracking procedure to the northern hemisphere ERA-Interim reanalysis data for DJF, 1989 to 2009. Cyclone tracks have to last 24 hours or more. Density shows the percentage of cyclone occurrence per timestep and area of 1000km^2.

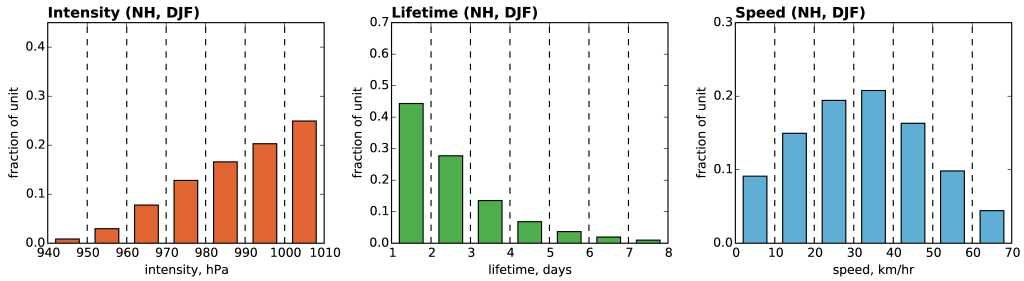

Figure 6: Relative frequencies of track metrics in the tracks identified by applying the storm tracking procedure to the Northern hemisphere ERA-Interim reanalysis data for DJF from 1989 to 2009. Left to right: Cyclone intensity (minimum pressure in feature object), lifetime of the cyclone in days and mean propagation speed of the cyclone in km/hr.

References

Anderson, D., Hodges, K. I., Hoskins, B. J., 2003. Sensitivity of feature-based analysis methods of storm tracks to the form of background field removal. Monthly Weather Review 131, 565–573.

Andrews, D. G., 2000. An Introduction to Atmospheric Physics. Cambridge University Press.

Benestad, R., Chen, D., 2006. The use of a calculus-based cyclone identification method for generating storm statistics. Tellus A 58 (4), 473–486.

Dee, D. P., Uppala, S. M., Simmons, A. J., Berrisford, P., Poli, P., Kobayashi, S., Andrae, U., Balmaseda, M. A., Balsamo, G., Bauer, P., Bechtold, P., Beljaars, A. C. M., van de Berg, L., Bidlot, J., Bormann, N., Delsol, C., Dragani, R., Fuentes, M., Geer, A. J., Haimberger, L., Healy, S. B., Hersbach, H., Hlm, E. V., Isaksen, L., Kllberg, P., Khler, M., Matricardi, M., McNally, A. P., Monge-Sanz, B. M., Morcrette, J.-J., Park, B.-K., Peubey, C., de Rosnay, P., Tavolato, C., Thpaut, J.-N., Vitart, F., 2011. The ERA-Interim reanalysis: configuration and performance of the data assimilation system. Quarterly Journal of the Royal Meteorological Society 137 (656), 553–597.
URL `http://dx.doi.org/10.1002/qj.828`

Gray, R. W., 1994. Fullers Dymaxion map. Cartography and Geographic Information Systems 21 (4), 243–246.

Gray, R. W., 1995. Exact transformation equations for Fullers world map. Cartographica: The International Journal for Geographic Information and Geovisualization 32 (3), 17–25.

Hewson, T. D., Titley, H. A., 2010. Objective identification, typing and tracking of the complete life-cycles of cyclonic features at high spatial resolution. Meteorological Applications 17 (3), 355–381.

Hodges, K. I., 1994. A general method for tracking analysis and its application to meteorological data. Monthly Weather Review 122, 2573–2586.

Hodges, K. I., 1995. Feature tracking on the unit sphere. Monthly Weather Review 123, 3458–3465.

Hodges, K. I., 1999. Adaptive constraints for feature tracking. Monthly Weather Review 127, 1362–1373.

Hoskins, B. J., Hodges, K. I., 2002. New perspectives on the Northern Hemisphere winter storm tracks. Journal of Atmospheric Sciences 59, 1041–1061.

Hoskins, B. J., Hodges, K. I., 2005. A new perspective on Southern Hemisphere storm tracks. Journal of Climate 18, 4108–4129.

Jennings, G. A., 1994. Modern Geometry With Applications. Springer.

Massey, N., 2012. Feature tracking on the hierarchical equal area triangular mesh. Computers and Geosciences 44, 42–51.

Murray, R. J., Simmonds, I., 1991. A numerical scheme for tracking cyclone centres from digital data. Part I:development and operation of the scheme. Australian Meteorological Magazine 39, 155–166.

Neu, U., Akperov, M. G., Bellenbaum, N., Benestad, R., Blender, R., Caballero, R., Cocozza, A., Dacre, H. F., Feng, Y., Fraedrich, K., et al., 2013. IMILAST: A community effort to intercompare extratropical cyclone detection and tracking algorithms. Bulletin of the American Meteorological Society 94 (4), 529–547.

Post, F. H., Vrolijk, B., Hauser, H., Laramee, R. S., Doleisch, H., 2003. The state of the art in flow visualisation: Feature extraction and tracking. Computer Graphics Forum 22 (4), 775–792.

Rudeva, I., Gulev, S. K., 2007. Climatology of cyclone size characteristics and their changes during the cyclone life cycle. Monthly Weather Review 135 (7), 2568–2587.

Serreze, M. C., 1995. Climatological aspects of cyclone development and decay in the Arctic. Atmosphere-Ocean 33 (1), 1–23.

Sinclair, M. R., 1994. An objective cyclone climatology for the Southern Hemisphere. Monthly Weather Review 122, 2239–2256.

Sinnott, C. W., 1984. Virtues of the haversine. Sky and Telescope 68 (2), 159.

Wang, X. L., Swail, V. R., Zwiers, F. W., 2006. Climatology and changes of extratropical cyclone activity: Comparison of ERA–40 with NCEP–NCAR reanalysis for 1958-2001. Journal of Climate 19 (13), 3145–3166.