Maths Foresees project report: Feature identification and tracking with application to meteorological data

Rob Lamb[1], Ben Powell[2], Jonty Rougier[2], Philip Rudd[3], Kay Shelton[1], and Edward Skevington[2]

[1]JBA Trust, Skipton, UK
[2]University of Bristol
[3]University of Cambridge

March 11, 2016

Abstract

In this report we describe the challenge brought by the JBA Trust to the EPSRC's Maths Foresees Network, and the response to it, developed by the authors during a workshop hosted by the Turing Gateway to Mathematics at the Newton Institute in Cambridge on 21-24 September 2015.

The challenge was to identify coherent meteorological features in time and space from arrays of real or simulated data. Our response proceeded from a formalization of the problem as the identification of connected sets of vertices on a graph, satisfying particular inequalities that qualify them for feature status.

1 Report structure

In section 2, we introduce the challenge brought by the JBA Trust, provide some more general context and describe the prototypical data set on which our numerical experiments have been trialled. In section 3, we step back from the challenge's specifics slightly to discuss its more abstract mathematical setting. This helps us identify certain key issues as well as links from JBA's challenge to those in other fields. In section 4, we zoom back in and describe in detail the algorithms developed during the workshop. Finally, in section 5 we identify a small set of conclusions, both mathematical and methodological, that appear to us to be particularly significant.

2 Context and problem specification

2.1 Context

Most of us are familiar with the depiction of fronts on weather charts (Figure 1); they are a common tool used in everyday life, allowing us to quickly assess the current or a future weather situation with only a basic understanding of the symbols and their interpretation. While these features have long been subjectively hand-analysed on weather charts, recently there has been a push to use computer algorithms to objectively identify the location of weather features. The UK Met Office now uses such techniques to draw the various types of fronts (cold, warm, occluded) and surface pressure troughs on their surface pressure charts. In the research community, the use of computer algorithms to identify weather features objectively in global atmospheric reanalysis datasets (i.e. spatially three dimensional representations of the atmosphere produced using numerical weather prediction models and global climate models) allows the long-term climatology of weather features to be assessed for the first time.

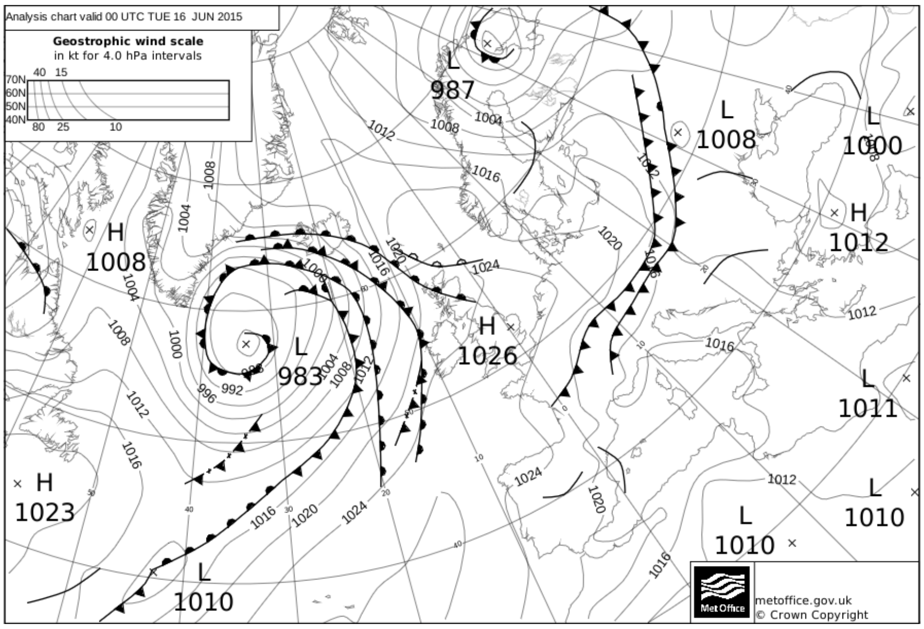

Figure 1: Surface pressure chart for 00 UTC 19 November 2009, produced by the UK Met Office. Obtained from http://www.wetterzentrale.de/topkarten/tkfaxbraar.htm.

Traditionally, the temporal and spatial variability of weather features, such as fronts and surface pressure troughs, on seasonal and interannual timescales was deduced from just a few years of data on regional scales due to vast amounts of hand-analysis required. With the aid of objective, computer-based analysis, global climatologies of weather features based on decades of data are now possible; for example, the first global climatology of fronts was published in 2011.

Identification of weather features is crucial to both understanding the atmospheric processes occurring and predicting how the atmosphere may evolve. Climatologies of the frequency of occurrence of weather features allow hot-spots of activity to be identified. Studies focused on these hot-spot locations allow us to investigate and understand the dynamical processes occurring in the atmosphere that are responsible for making a particular location favourable or preferential for a particular weather feature.

2.2 Problem specification

The identification of weather features is usually accomplished in two dimensions, e.g. on a single quasi-horizontal surface such as a single altitude or on a pressure surface. However, the majority of weather features exhibit spatially three-dimensional structure, for example the tilt of weather fronts with altitude (Figure 2). In many weather features the vertical structure evolves with time, throughout the feature's lifecycle. Changes in the vertical structure of features can be indicative of where the feature is in its lifecycle; capturing those changes can therefore have practical value in forecasting applications. This information can be lost when the feature is identified in two dimensions only unless the feature identification is performed at multiple vertical levels and some association between similarly located features at different levels is made. Such a process is performed for the identification of tropical and extratropical cyclones in vortex identification and tracking schemes, however it is not generally employed for other types of weather features.

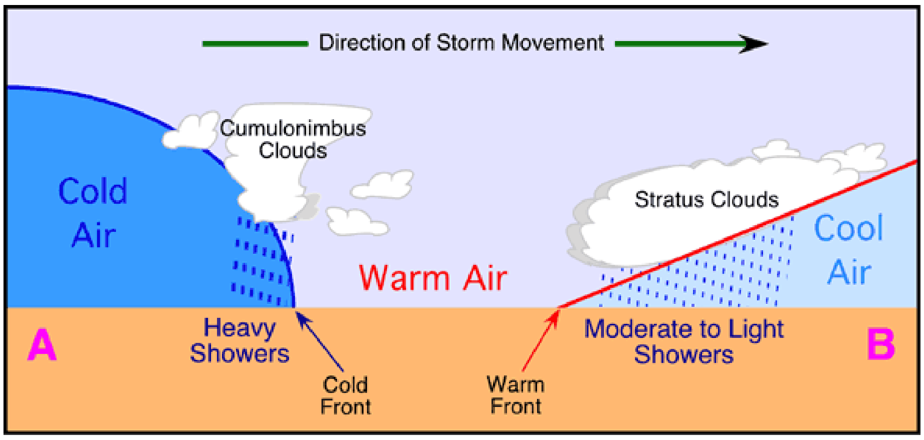

Figure 2: Vertical cross section through a low pressure systems showing the tilt of warm and cold fronts with height. Obtained from http://www.physicalgeography.net.

The identification of weather features in three dimensions presents an interesting problem. The nature of weather features and the representation of the atmosphere using gridded three-dimensional datasets means the form of the features to be identified can be complex and highly variable in both space and time. A first attempt to identify coherent three-dimensional objects requires a simplification of the problem to the most basic of structural features present in the atmosphere. For this purpose, horizontal wind speed is selected as the variable to be used in defining 'features', allowing for the identification of 'jets' within the troposphere.

While the aim of this project is to develop techniques or algorithms to identify weather features in spatially 3D atmospheric datasets, and to investigate their coherence in time (i.e. 4D data), the fundamental mathematically-based concepts underlying many approaches to tackling the problem have much broader applications. The features the resultant algorithms identify do not necessarily need to be atmospheric features; the algorithm could be applied to any type of data stored on some kind of grid. In essence, this could be any type of data that can be represented as a vector or matrix.

For meteorological applications, development of an algorithm to identify features in three dimensions would allow for a more comprehensive representation of them. Given this, we would expect analyses involving the classification of features to be significantly improved too.

2.3 Data

The National Centres for Environmental Prediction/National Center for Atmospheric Research (NCEP/NCAR) Reanalysis 1 dataset, obtained from the Physical Sciences Division of the Earth System Research Laboratory (`http://www.esrl.noaa.gov/psd/data/gridded/data.ncep.reanalysis.html`) is used in this study. This dataset has global coverage with a latitude/longitude grid size of 2.5x2.5, with 17 irregularly-spaced pressure levels. Atmospheric variables are stored at the grid vertices (cell centres) every 6 hours.

Zonal (east/west directed) wind during January 2006 is used as the initial atmospheric variable for development and testing of the 3D feature identification system. By applying a threshold to the wind speeds, for example 50 ms^{-1}, regions with westerly (eastward directed) winds exceeding this threshold are isolated in the data as coherent stretched blobs as demonstrated by the cross-sections in Figure 3.

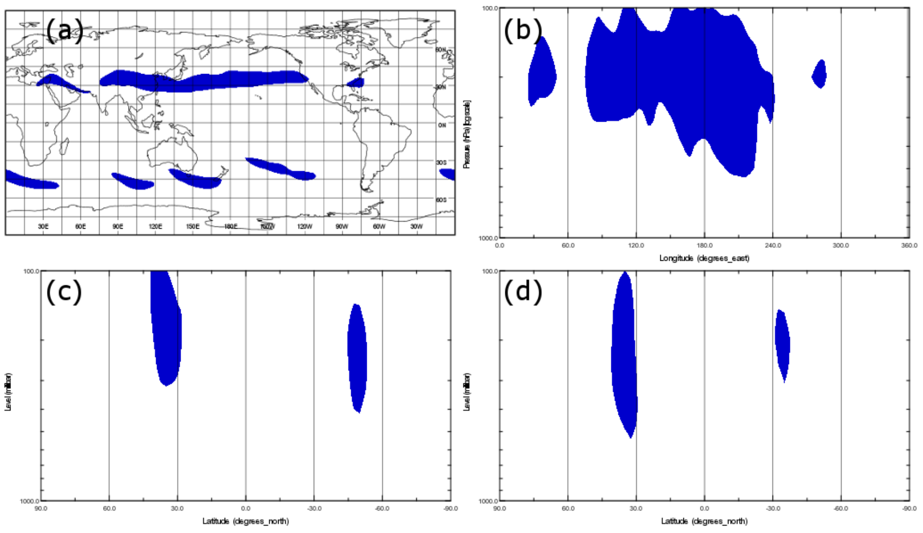

Figure 3: Cross-sections zonal wind exceeding 50 ms^{-1} (shaded) along (a) the 200-hPa pressure surface, (b) the 32.5N line of latitude, (c) the 105E line of longitude, and (d) the 210E line of longitude. For the vertical cross-sections, the lower coordinate is 1000 hPa and the upper coordinate 100 hPa; the axis is logarithmic.

The spatial scales of these features can vary greatly. For the larger-scale elongated tube-like structures forming the mid-latitude westerly jets, they can almost encircle the globe, spanning over 10,000 km in the longitudinal direction, while the latitudinal width may be only around 1,500 km. The vertical scale of the features can also vary greatly and has the added complication of using pressure as a vertical coordinate in the NCEP/NCAR dataset.

Smaller horizontal scale coherent zonal wind features also exist, often termed 'jet streaks', these often exhibit less longitudinally-elongated structures. If considering jets in the total (vector) wind field, it is important to bear in mind that the major axis of a jet will not be aligned longitudinally, it may form 'U' or 'S' shapes. Hence isolating these features may need to consider the local major and minor axis of the feature.

3 Mathematical background

As alluded to in section 2, the task of identifying connected features from arrays of observations is already receiving attention from the community of atmospheric and climate scientists. A key element to the challenge brought by the JBA Trust, however, lies in the application of feature identification methods to four-dimensional spatio-temporal arrays.

The difficulty in extending physical or statistical models to high dimensional settings is often referred to as the 'curse of dimensionality'. The general idea is that in a high-dimensional space there are more directions in which any two points can move away from each other, and so one can fill a space with far more points at distinct locations than in a lower-dimensional space. We say that neighbourhoods becomes smaller and domains become larger. The first part of the problem is that this number of points in a volume scales exponentially with dimension, since adding a dimension means multiplying the number of points we had previously by a given factor rather than, say, adding to it. The second part is that conventional matrix calculations implicitly allow for linear dependencies between all points, and hence typically scale with the cube of the number of locations. Together, these scaling laws compound and quickly make for overwhelmingly demanding computations.

There are (at least) two ways in which computational demand can be brought back under control. The first relies on there being some low dimensional manifold (typically corresponding to a physical law or constraint) embedded in the high dimensional space, on which the most important part of the data lies. Although this strategy can be very difficult when the manifold has an exotic shape, it is, in a sense, cheating since we are explicitly bringing the dimensionality back down.

The second strategy, which is our focus here, is not to deny the high dimensionality of a system but to suppose that it is characterized by a special sort of dependence property whereby a system value at a particular point is only influenced by the values directly surrounding it. In the mathematics literature, this is known as a Markov property and can be understood as corresponding to a rejection of the 'action at a distance' principle, a rejection which is considered applicable for a wide range of classical physical systems.

Markov's explicit formulation of local dependence not only seems intuitively sensible but encourages us to develop computational procedures in which we, or rather our algorithms, look at only small neighbourhoods at a time, and let large scale inferences emerge organically as we complete a tour of the neighbourhoods. To be clear, the Markov property will not help us overcome the exponential scaling of volumes in high dimensions but will allow us to avoid the cubic scaling of computation required to accommodate dependencies between all locations.

To be more explicit, we argue that typical spatial data analysis problems are characterized by a computational cost proportional to the number of interactions per location raised to a small power ($k = 2$ or 3 for example), multiplied by the total number of locations considered. When all interactions are modelled, we arrive at projections for computational demand that look like:

$$\begin{aligned}\text{Number of points} &\propto \exp(d),\\ \text{Interactions per point} &\propto N,\\ \text{Computational cost} &\propto NN^k \propto \exp((k+1)d),\end{aligned}$$

where d is the dimension of the space the data live in. When we take the Markov property into account however we find that powers of d slip out of the exponential, slowing down the explosion in computational demand so that it now looks like:

$$\begin{aligned}\text{Number of points} &\propto \exp(d),\\ \text{Interactions per point} &\propto d,\\ \text{Computational cost} &\propto Nd^k \propto \exp(d)d^k.\end{aligned}$$

Conventionally, graphs are used to keep track of which dependencies are retained in a model. The graph structure immediately relevant to this project is the square lattice, where a node's number of neighbours scales linearly with the dimension.

An additional motivation for embracing graph-based methods for high-dimensional problems, is the way they can help us change how we think about and manipulate high-dimensional datasets in practice. To an algorithm for graph decomposition, for example, we propose that features or regions look almost the same no matter the dimension of the space they live in. A higher dimension will just increase the average connectivity between vertices. Thus, we see an opportunity for developing software capable of interpreting data in any dimension without needing to adapt its subroutines in any way.

Although the the task at hand is hard to frame as statistical inference, the statistical literature on Markov models can provide us with, at least, inspiration for our work. Here, credit is often attributed to [1] for highlighting the potential for the models to assimilate large amounts of spatio-temporal data. More recently [3] have combined Markov and Gaussian assumptions to enable even more ambitious calculations.

Our point is that conversations about the significance of modelling only local dependences are still active and energetic in the statistical community, a fact that is partly manifested in the range of sparse matrix- and graph analysis-packages that have been contributed to the R[2] open-source statistical computing project in recent years. In the numerical analysis community too, Markov models are very important - essentially underlying the finite-difference and finite-element methods used to model a wide range of physical phenomena.

4 Implementation

We now discuss the implementation of our ideas for graph-based analysis to gridded meteorological data. Given the time constraints for the project, code development has remained in the experimental stage. Nevertheless, preliminary results appear highly promising. In appendix 7, we provide some example images generated using our software.

4.1 Overview of the implementation

Our aim is to construct a graph in a computer's memory from data supplied by a user, so that the graph can then be analysed for connected components or features. This can be done in several ways.

One approach we tried during the workshop was to use sparse matrices to store an adjacency matrix, which when raised to a high enough power (at least the base two logarithm of the diameter of the graph) will give a connectivity matrix with entries being 1 if two nodes are connected and 0 if they are not.

However this approach is very inefficient because the connectivity matrix is not sparse, therefore it is recommended instead that a breadth-first or depth-first search is used, or another standard algorithm as will be used by any package that contains graph functions. We therefore adopted an alternative method in a MatLab implementation of a feature identification algorithm, which is outlined below.

4.2 Method

Import Data First the data must be imported into some data structure in the program. We used an array where each row is labelled by a 'global node ID', and the columns correspond to longitude, latitude, vertical elevation, time, and the value of the physical parameter at this point. Note that in a problem involving a time dimension, nodes at the same point in space but at different instants of time are considered as different nodes.

Thresholding There is a specified condition on the physical parameter which must be met at a node for it to be part of a feature, e.g. a threshold. The condition is evaluated for every node and, if true, the corresponding global node ID is added to an array of nodes in features. The row number in this array is called the 'nodes ID'.

Adjacency The next thing we want to know is 'are two nodes adjacent?'. Let us take two nodes in the array of nodes in features and assume there is some adjacency function which is 1 if two nodes are adjacent and 0 if they are not. This could be given by a function working on an underlying metric or a simple physical distance measure with a threshold. In both cases it is possible to include more than 'nearest neighbour' adjacencies. We can then construct a matrix where rows and columns are labelled by the nodes ID of the two nodes, respectively, and the value is determined by the adjacency function. This is the adjacency matrix.

Connectivity To find features, the adjacency matrix is passed to a function which indexes the connected components (features) and then returns an array where the row number is the node ID and the value is the feature index. Thus the node IDs for each identified feature can be exported. This algorithm can be given the full set of nodes for which it will return the spatial-temporal features, or it can be given nodes at a single time slice, for which it will return the spatial features, or indeed any subset of the data can be given and the features as seen in that subset alone will be found.

4.3 Visualization

Our aim here is to describe how the features may be visualized to aid the presentation of results. For a problem of two spatial dimensions it is efficient simply to plot filled circles around the centres or squares of each node belonging to a certain feature, plotted on top of a map. Through the inclusion of transparency it is even possible to create a quasi-3D effect. However, for higher dimensions and more detailed analyses such an implementation may not be sophisticated enough. Hence we worked on different ways to present 3D features in plots and movies.

The first method of visualisation we implemented used the convex hull of the cloud of points forming a feature. This turned out to have several disadvantages mainly related to the splitting of point clouds. Firstly, the implemented feature identifier supports splitting and merging for a problem with temporal dimension. Secondly the use of the implemented code in a geophysical field requires a periodic domain, meaning two halves of the same feature can be found on different ends of the domain even in a 2D problem. The convex hull would in both cases connect the (actually separated) clouds of points.

We therefore tried a second approach, which was to define a new field for every feature, with value 1 on every node belonging to that feature and value 0 elsewhere. We then constructed the 0.5-isosurface of each field and plotted the corresponding structures in a distinct colour. This way split features are visualised individually, but are still indicated as belonging together. The described procedure works in any number of dimensions.

4.4 Feature Tracking Graph

The aim of the current work is to not only identify features, but also to track their behaviour as time passes. Potential difficulties here arise when one imagines two spatial features, perhaps regions of high zonal wind, that are created on opposite sides of the planet and migrate around the planet independently, only to finally merge into a single spatial feature. Alternatively, one could imagine two large features that are spatially separated, which interact when one sheds a small feature that is then absorbed by the other. To identify such behaviour, which we may be reluctant to identify as a significant connection between spatial features, we must be able to break down spatio-temporal features into those which are deemed to be transient and those which are deemed to be sustained.

To help with this we propose introducing a new graph, a 'Feature Tracking Graph' (FTG) which has a node at each instant of time for each spatial feature and connects nodes at adjacent instants which are adjacent in the sense of the spatio-temporal feature. One (experimental) method for constructing such a graph is presented below.

Find Spatial Features For each instant of time construct a set of nodes that are contemporary at that instant, and that also satisfy the thresholding condition. Pass this set of nodes through the Adjacency and Connectivity steps to identify the spatial features. Using these spatial features construct an array of structures, where the row index is the spatial feature ID (not the same as the ID index in Connectivity), and each structure contains an array with the nodes that are part of the feature, and also at the prescribed instant of time. Thus we have an array of the spatial features at one 'time slice'.

Find Edges Now take the nodes that are in two-instant slices and perform the Adjacency and Connectivity steps to identify the two-instant features. For each two-instant feature, if any of the nodes in the spatial features at the two instants of times are a member, then their entire corresponding feature must also be a subset, and so we can find the feature IDs of the corresponding spatial features and construct an adjacency matrix for the FTG. Now that we have the graph, it can be used to help investigate the found features.

5 Conclusions and remarks

The following conclusions distil our findings from the project, based on our development and testing over four days at the Foresees workshop:

- Large-scale feature tracking with 4-D lattice data is possible with desktop computers.
- The curse of dimensionality is significantly mitigated by reducing the number of node interactions.
- Sparse matrix methods are a convenient intermediate step for encoding dependence structures but are not practical for really large problems.
- Highly tuned packages for the manipulation and analysis graph/lattice problems already exist for R and Matlab. We must take full advantage of these.
- On long enough time scales, coalescence of features makes for very large, irregularly shaped mega-features. This is just one way for computational demand to creep upwards. In further applications of the methods described here it will be helpful to identify limits on the spatial and temporal extents of meaningful features so that the upwards creep may be capped.
- The lattice structure is a convenient artifice for dealing with 'medium-sized' problems, and graph-based methods help us postpone the point at which 'medium-sized' becomes 'too large', but it may be worthwhile looking already for alternative data structures that could scale-up even further.

With regard to the authors' experience working together during the workshop, the following points were notable:

- It was very valuable for the mathematical team to have scientists on call to provide realistic test data sets and detailed explanation of the subtleties in the data, and the underlying physical processes. This close exchange allowed rapid progress to be made with algorithm development.

- Considerable skill, experience and rigour is required to code effectively. The project highlighted the interplay between low-level computing and high-level theory.

6 Notes and acknowledgements

At the time of writing, a video presentation on the authors' progress at the end of the workshop is available for viewing at `http://sms.cam.ac.uk/media/2079871`, while further details on other aspects of the workshop are available at `http://www.turing-gateway.cam.ac.uk/mfsg_sep2015.shtml`. The original problem specification supplied by JBA Trust is available at `http://www.turing-gateway.cam.ac.uk/documents/mfsg-sept2015/Industrial%20Challenge%20-%20JBATrust%20Sept%202015.pdf`.

We would like to thank the Turing Gateway to Mathematics and Maths Foresees team for organizing this stimulating workshop. The support of the EPSRC Living with Environmental Change - Maths Foresees Network and the NERC Probability, Uncertainty and Risk in the Environment (PURE) Network is gratefully acknowledged.

7 Appendix

References

[1] Julian Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 192–236, 1974.

[2] R Development Core Team. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria, 2009. ISBN 3-900051-07-0.

[3] H. Rue and L. Held. *Gaussian Markov Random Fields: Theory and Applications.* Chapman & Hall/CRC Monographs on Statistics & Applied Probability. CRC Press, 2005.

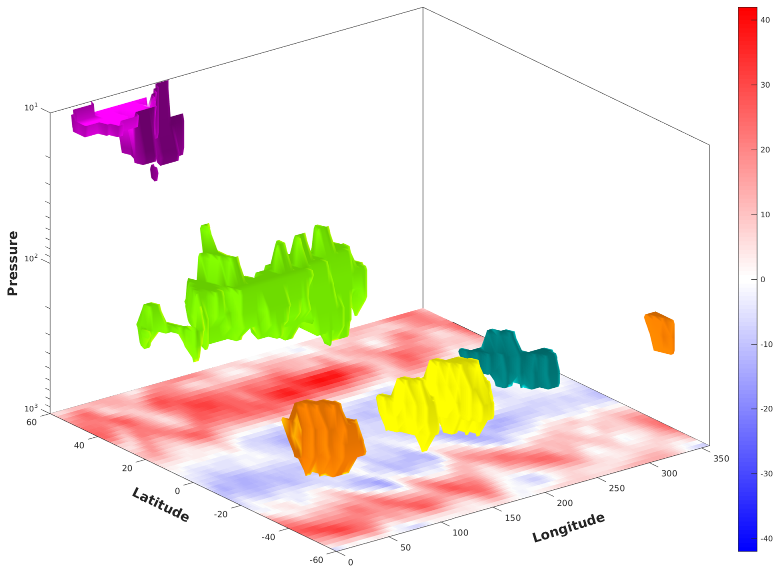

Figure 4: A still from a video produced by a prototype version of the graph-based feature tracking software developed by the authors. The coloured objects show individual features with zonal winds exceeding 50m/s at the given time. The colour field shows the column mean to give an idea of high velocity regions in the horizontal.