procedure.

Geo Projection `function ProjectedLocations`

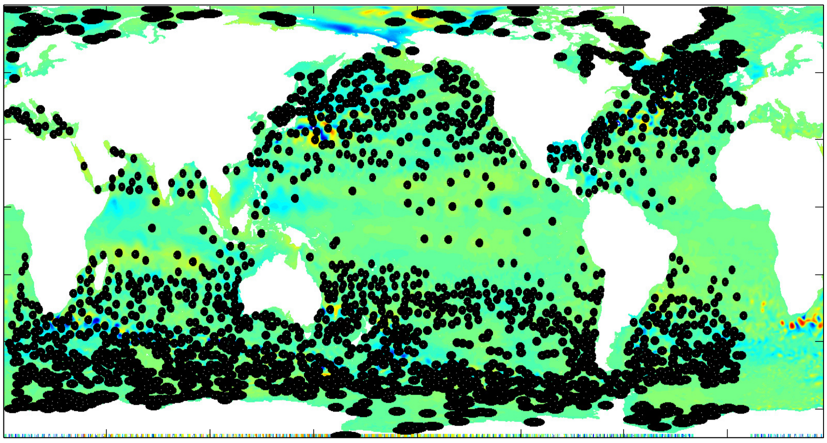

Figure 2.10: Among the saved meta-information for each eddy are also the indices describing the ellipse that defines the eddy's allowed locations with respect to the next time-step.

An optional threshold on the distance an eddy is allowed to travel over one time-step is implemented in the tracking algorithm 2.5. This is a direct adaptation of the ellipse-based constraint described by Chelton *et al.* [3]. The maximum distance in western direction traveled by the eddy within one time-step is limited according to $x_{west} = \alpha c \delta t$ with c as the local long-Rossby-wave phase-speed and *e.g.* $\alpha = 1.75$. In eastern direction the maximum is fixed to a value of *e.g.* $x_{east} = 150$km. This value is also used to put a lower bound on x_{west} and for half the minor axis (y-direction) of the resultant ellipse.

This function builds a mask of eligible geo-coordinates with respect to the next time-step.

2.6 Step S05: Track Eddies

Main Tracking Procedure

`function S04_track_eddies`

Due to the the relatively fine temporal resolution (daily) of the model data, the tracking procedure for this case turns out to be much simpler than the one described by Chelton *et al.* [2]. There is almost no need to project the new position of an eddy, as it generally does not travel further than its own scale in one day. This means that one eddy can usually [7] be tracked unambigiously from one time step to the next as long both time-steps agree on which eddy from the *other* time-step is located the least distance away. The algorithm therefor simply builds an arc-length-distance matrix between all old and all new eddies and then determines the minima of that matrix in both directions *i.e.* one array for the new with respect to the old, and one for the old with respect to the new set. This leads to the following possible situations:

[7] The only exception being the situation when one eddy fades and another emerges simultaneously and in sufficient proximity.

- Old and new agree on a pair. I.e. old eddy O_a has a closest neighbour N_a in the new set and N_a agrees that O_a is the closest eddy from the old set. Hence the eddy is tracked. N_a is O_a at a later time.

- N_a claims O_a to be the closest, but N_b makes the same claim. I.e. two eddies from the new set claim one eddy from the old set to be the closest. In this situation the closer one is decided to be the old one at a later time-step and the other one must be a newly formed eddy.

- At this point all new eddies are either allocated to their respective old eddies or assumed to be *newly born*. The only eddies that have not been taken care of are those from the old set, that *lost* ambiguity claims to another old eddy, that was closer to the same claimed new eddy. I.e. there is no respective new eddy available which must mean that the eddy just *died*. In this case the entire track with all the information for each time step is archived as long as the track-length meets the threshold criterium. If it doesn't, the track is abandoned.

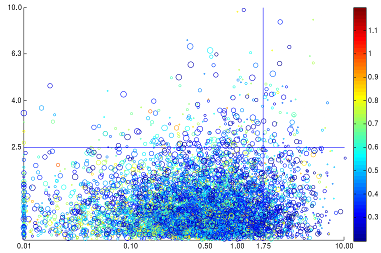

Figure 2.11: Each circle represents one eddy in the new time step. Y-axis: Maximum ratio to closest eddy in old set of either amplitude or σ, where 1 means *identical* and 2 means factor 2 difference. The threshold used for the final runs was 2. X-axis: Ratio of distance to closest eddy from old set divided by δt to local long-Rossby-wave phase-speed. Color-axis: Isoperimetric Quotient. Radius of circles: ratio of σ to local Rossby-radius. All eddies with said ratio larger than 10 are omitted. Note the obvious inverse correlation of scale to IQ, suggesting that all large *eddies* likely represent more than one vortex.

Improvements

The former is the core of the tracking algorithm. It is almost sufficient by itself as long as the temporal resolution is fine enough. The larger the time-step, the more ambiguities arise, which are attempted to be mitigated by flagging elements of the distance matrix not meeting certain thresholds:

- `function checkDynamicIdentity`
  Consider the ambiguous case when there are two eddies N_a and N_b in sufficient proximity to eddy O_a. Let's assume O_a is a relatively solid eddy of rel. large scale with a steep slope *i.e.* large

amplitude and that N_a is merely a subtle blob of an eddy whilst N_b is somewhat similar to O_a but with only half the amplitude. The situation then is clear: N_b is the, apparently slowly dying, O_a at a later time, while N_a could either be a newly formed eddy, an old eddy with its respective representation in the old set something other than O_a, or even just temporary coincidental noise not representative of any significant meso-scale vortex. This interpretation should hold even when O_a sat right between the other two, thereby being much closer to O_a than N_b was.
The purpose of this step is to make such decisions. It does so by comparing the *dynamic* versions of amplitude and scale (*ampToEllipse* and σ) between the time-steps. If either ratio from new to old [8] surpasses a given threshold the pair is flagged as non-eligible. It is important to use the *dynamic* parameters rather than those stemming from the contour line, because as mentioned in 2.5 the contour line itself and the eddy's geometric *character* are hardly correlated at all. One eddy can get detected at different z-levels from one time-step to the next, resulting in completely different amplitudes, scales and shapes with respect to the contour.

[8] In order to compare in both directions equally: $\exp\left(\left|\log\left(v_n/v_o\right)\right|\right)$ where v is either amplitude or scale.

The initial idea was, by assuming Gaussian shapes, to construct a single dimensionless number representing an eddy's geometrical character built upon the contour-related amplitude- and scale values only. Since we have no information about the vertical position of a given contour with respect to assumed Gauss bell, this problem turned out to be intrinsically under-determined and hence useless. The method eventually used, which checks amplitude and scale separately is again very similar to that described by Chelton *et al.* (see Box 1).

- `function nanOutOfBounds`

  This is the second half to the prognostic procedure described in 2.6. It simply flags all pairs of the distance matrix for which the index representing the *new* eddy's geographic location is not among the set of indices describing the ellipse [9] around respective *old* eddy.

[9] see figure 2.10.

- `function checkAmpAreaBounds`

  TODO:relevant only for chelton method!