The Algorithm

This section walks through the algorithm step by step, so as to explain which methods are used and how they are implemented. The idea is that the code from step `S01_..` on can only accept one well defined structure of data. In earlier versions the approach was to write code that would adapt to different types of data automatically. All of this extra adaptivity turned out to visually and structually clog the code more than it did offer much of a benefit. The concept was therefor reversed. `S00_prep_data` can be altered to produce reuqired output. Yet, there should be no need to adapt any of the later steps in any way. All input paramteres are to be set in `input_vars.m`

0.1 Step S00: Prepare Data

`function S00_prep_data`
Before the actual eddy detection and tracking is performed, SSH-, latitude- and longitude-data is extracted from the given data at desired geo-coordinate bounds and saved as structures in the form needed by the next step (S01).

0.1.1 S00: Set Up

`function [DD]=set_up`
The main purpose of this step is to Find the boundary-indices in the data describing a window that includes the user-chosen geo-coordinate window.

show plot
dow

0.1.2 S00: Parallel Part

`function spmd_body(DD)`
This function distributes chunks of data (i.e. bins of files) to the threads, which then perform the cutting and save their outputs to `'../DATA/CUTS/'`. The code can handle geo-coordinate input that crosses the longitudinal seam of the input data. E.g. say the input data comes in matrices that start and end on some (not necissarily strictly meridional) line straight across the Pacific and it is the Pacific only that is to be analyzed for eddies, the output maps are stitched accordingly. If the desired range is zonally continuous e.g. the entire southern ocean, the first tenth (zonally) of the input data is appended to the end of the output. This ensures that eddies right on the seam can be found and that eddies can be tracked across the seam. This results in eddies being identified twice in `function S03_filter_eddies.m` . Such *phantom* eddies are filtered out in `S04_track_eddies`.

ref

0.2 Step S01: Find Contours

`function S01_contours`

0.3 Step S01b: Find Mean Rossby Radii and Phase Speeds

`function S01b_BruntVaisRossby`

0.4 Step S02: Calculate Geostrophic paramteres

`function S02_infer_fields`

0.5 Step S03: Filter Eddies

```
function S03_filter_eddies
```

0.6 Step S04: Track Eddies

```
function S04_track_eddies
```

0.7 Step S05: Cross Reference Old to New Indices

```
function S05_init_output_maps
```

0.8 Step S06: Make Maps of Mean Parameters

```
function S06_analyze_tracks
```

for now see github for progress

Todo list