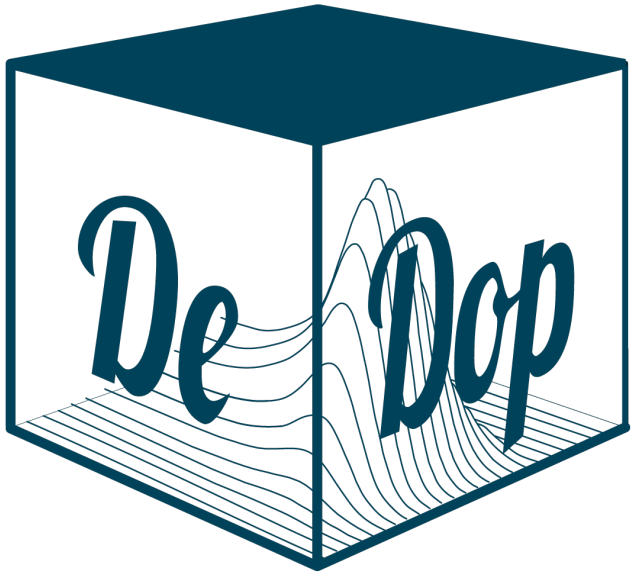


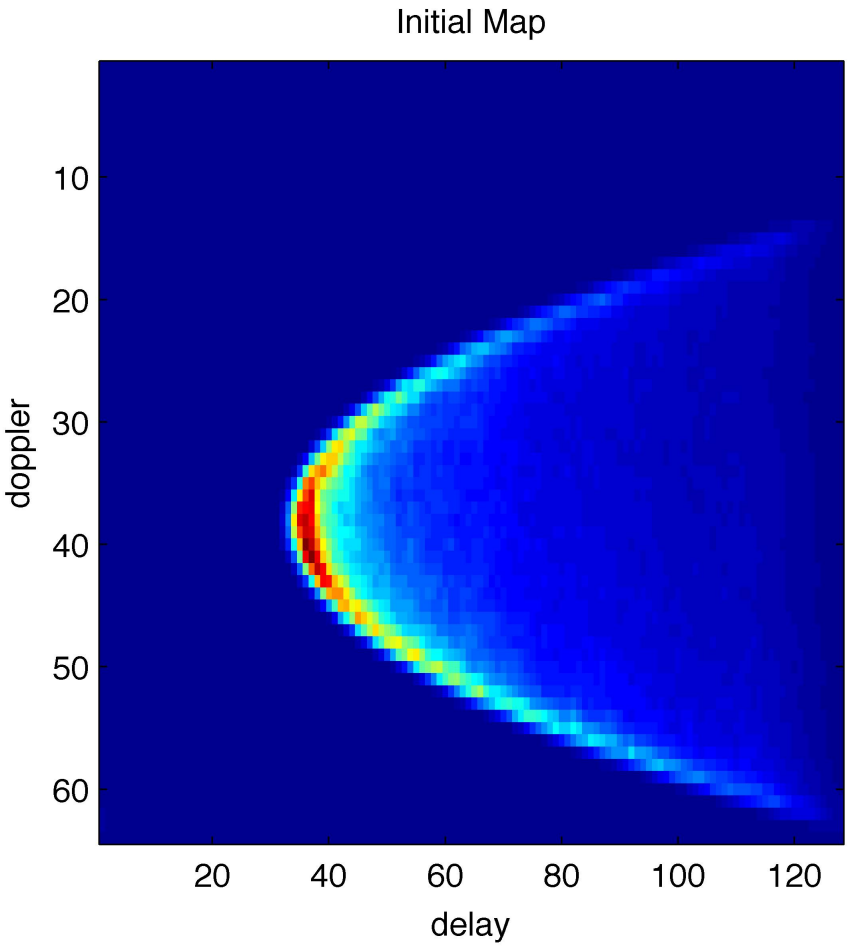
DeDop Case Study: ACDC Stability

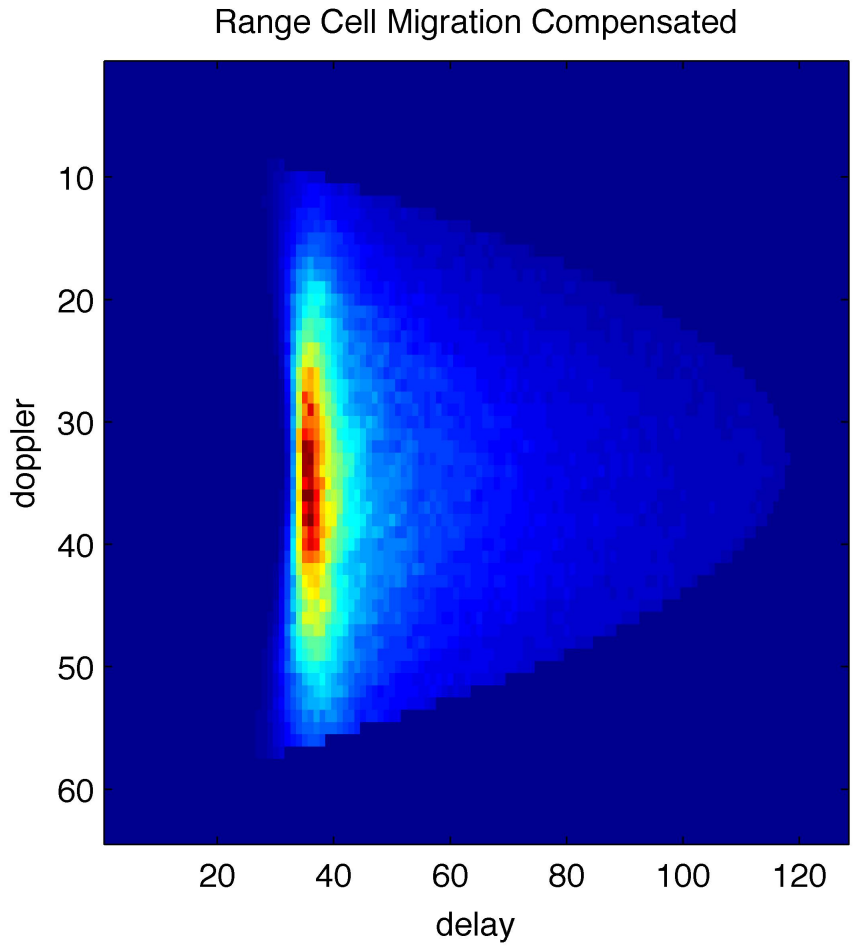
Chris Ray

isardSAT

Background

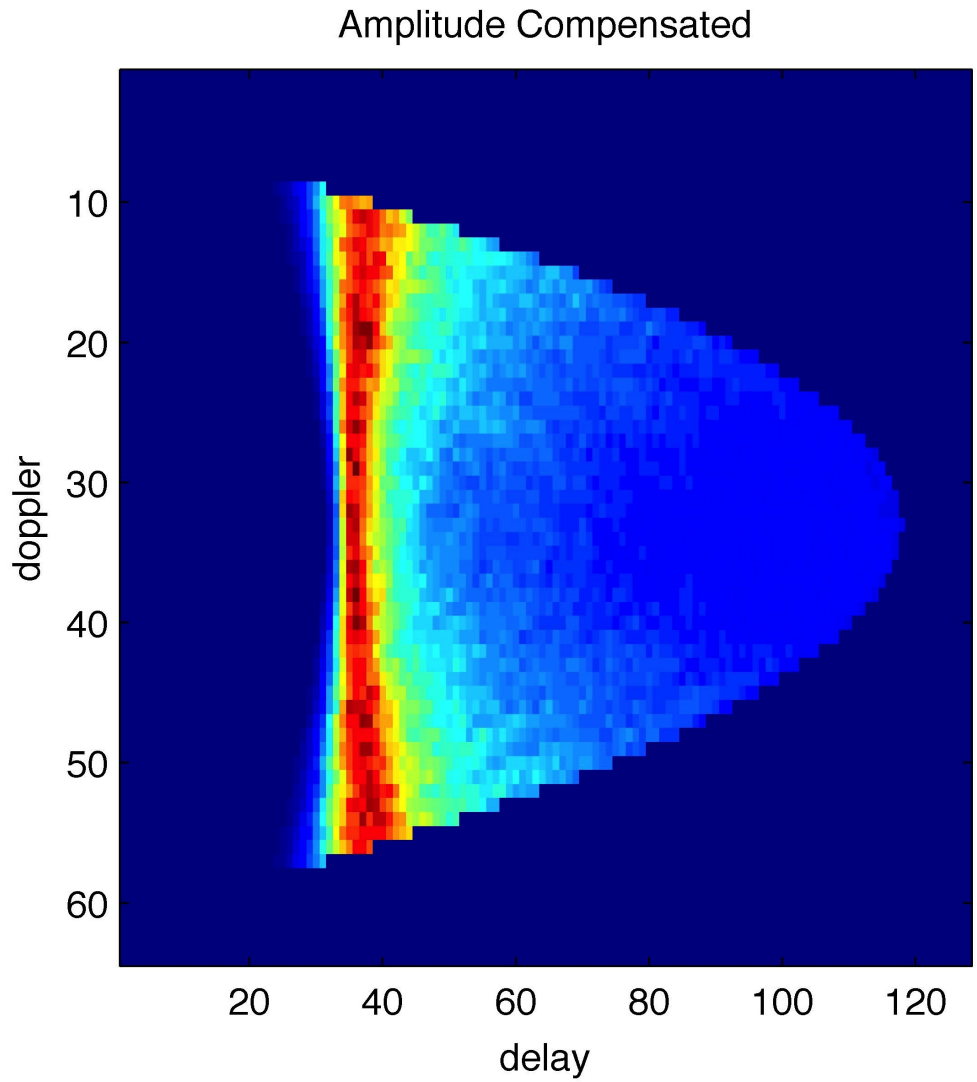
The delay doppler map of the power usually is compensated for range cell migration.





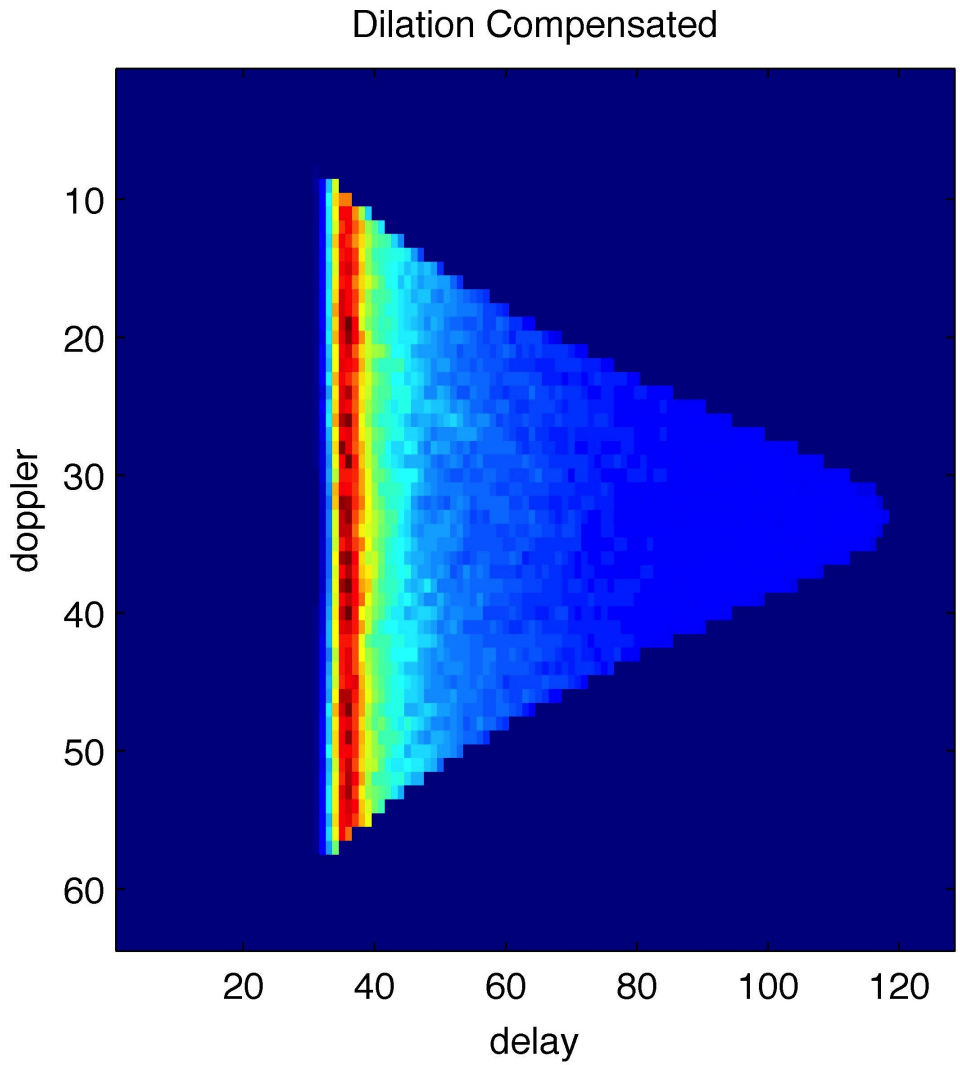
Background: Amplitude Compensation

In addition the amplitude can be compensated.



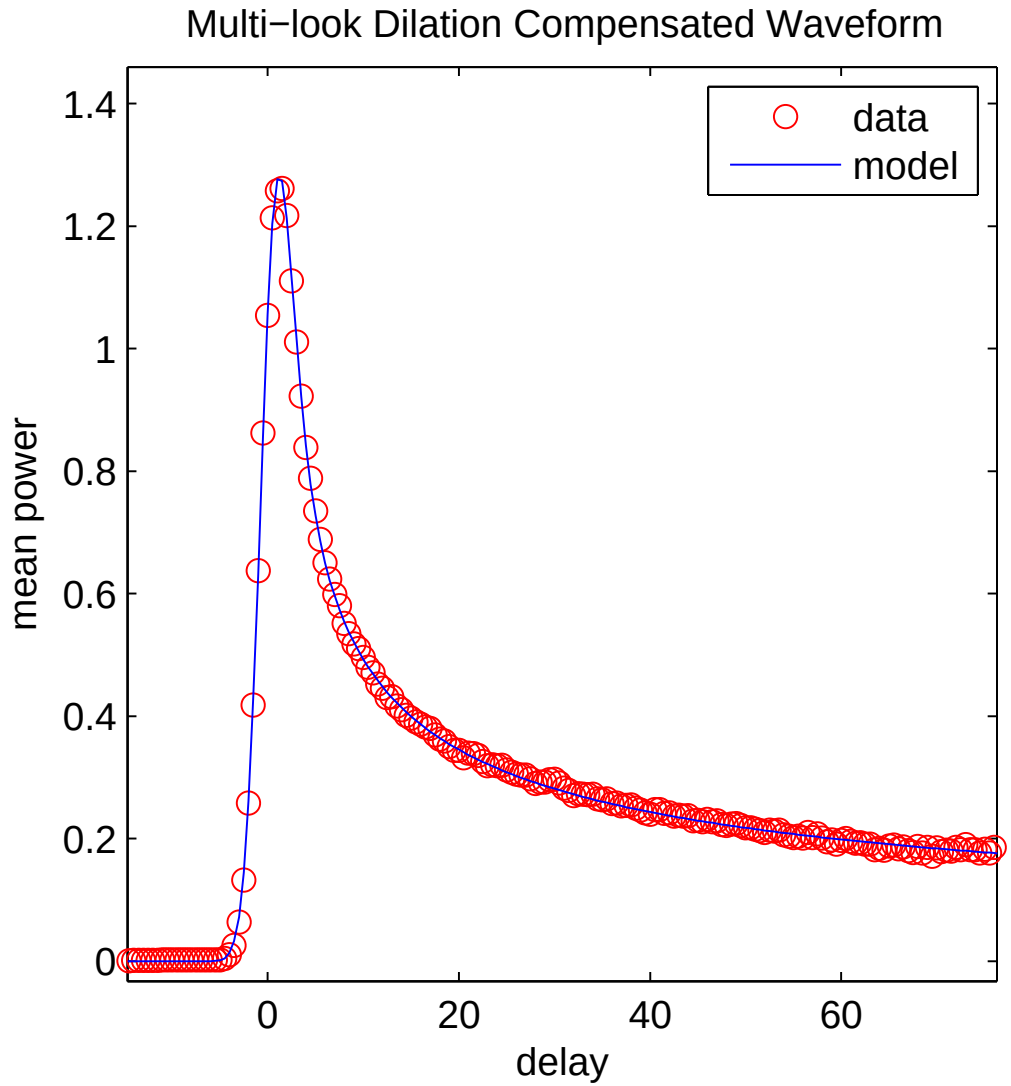
Background: Dilation Compensation

There is also a dilation of the range that can be compensated.
Leading to a DDM that is **the same** for all dopplers.



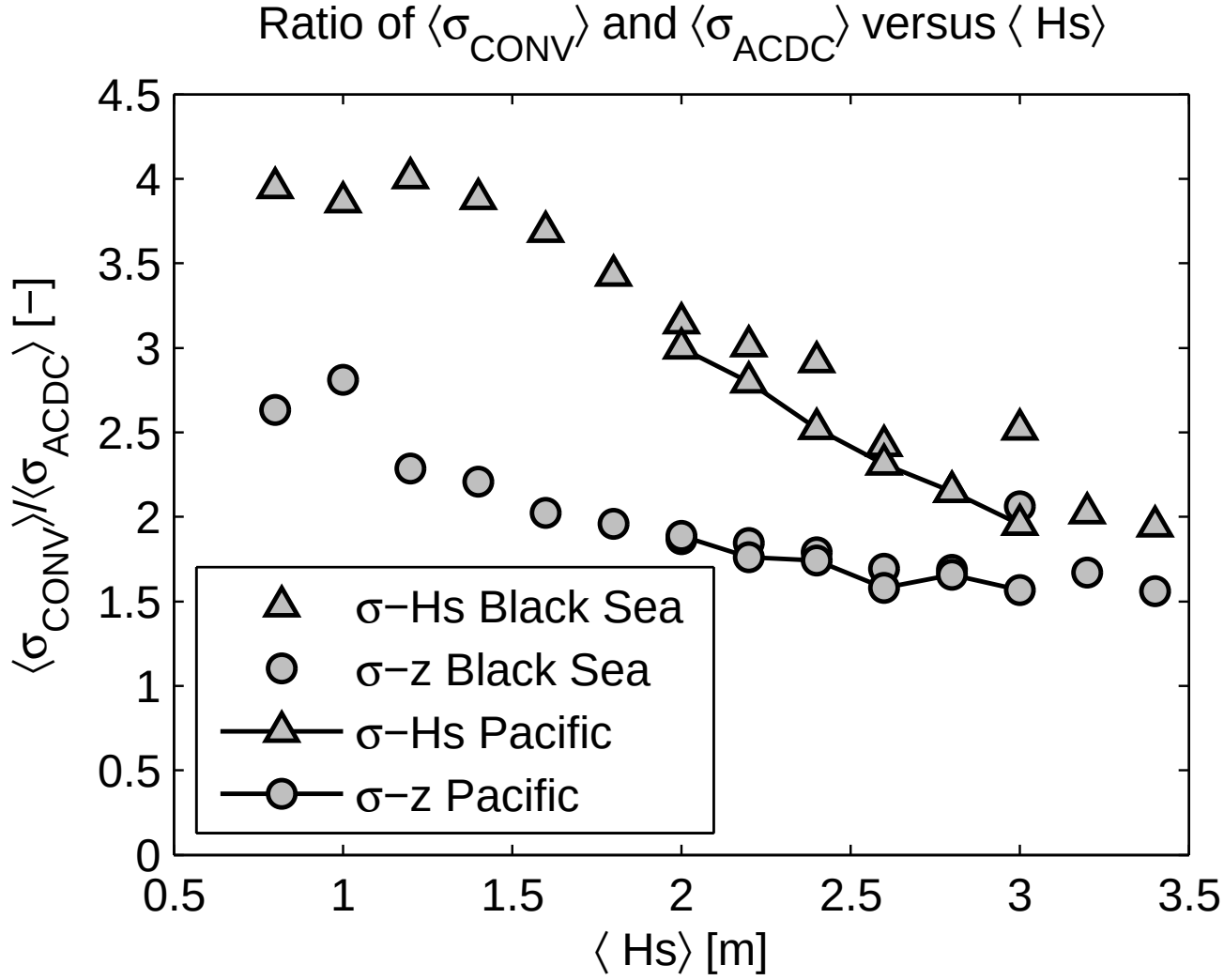
Background: Multi-look waveform

The multi-look waveform is the average of the dilation compensated power, and is fit by simply $P = Af_0(g_0(k - k_0))$.



Background: Retracking Improvements

Comparing the retracking results with the conventional Delay-Doppler retracking the noise levels are significantly lower.



ACDC is iterative in nature

In order to form the ACDC waveform ψ an initial estimate of the sea state parameters p_0 is needed. From fitting the ACDC waveform we get a new estimate p_1.

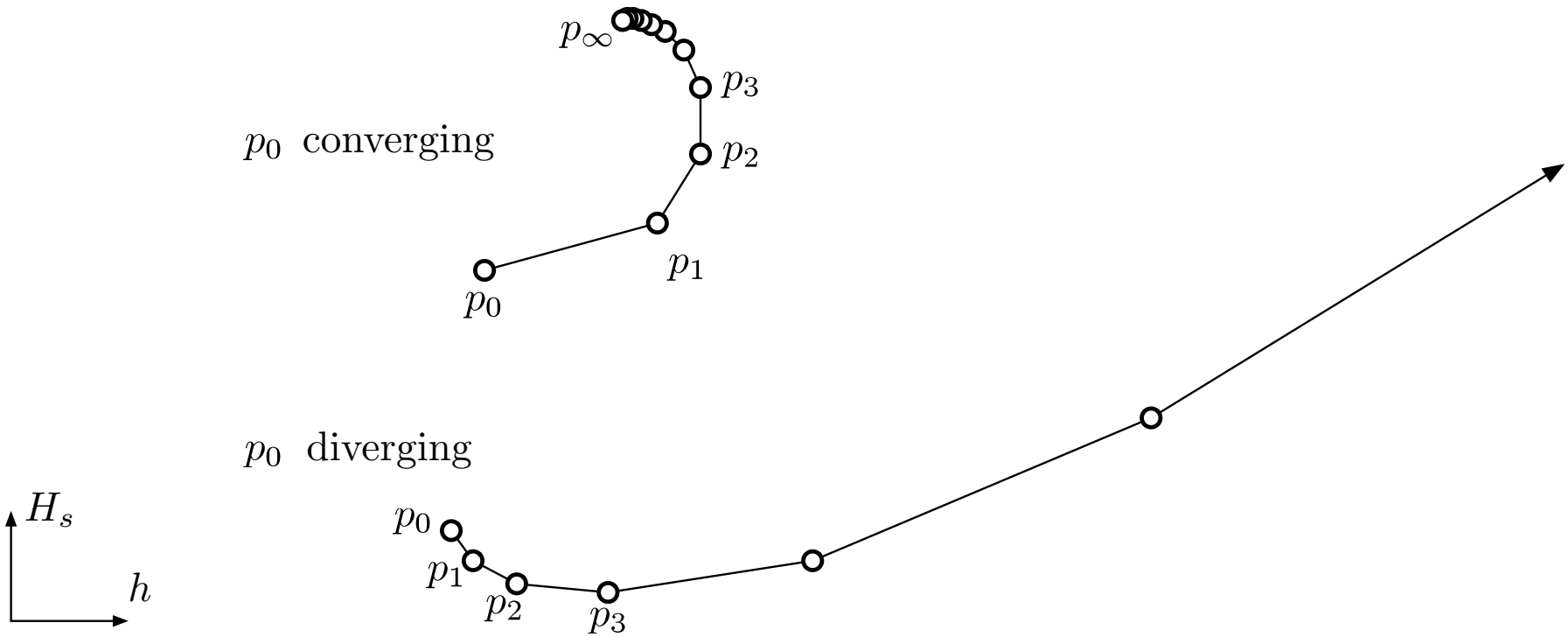
$$p_1 = \Xi(p_0)$$

ACDC is stable if p_1 is a better estimate than p_0.

First Question

We can form a sequence of estimates p_n by letting

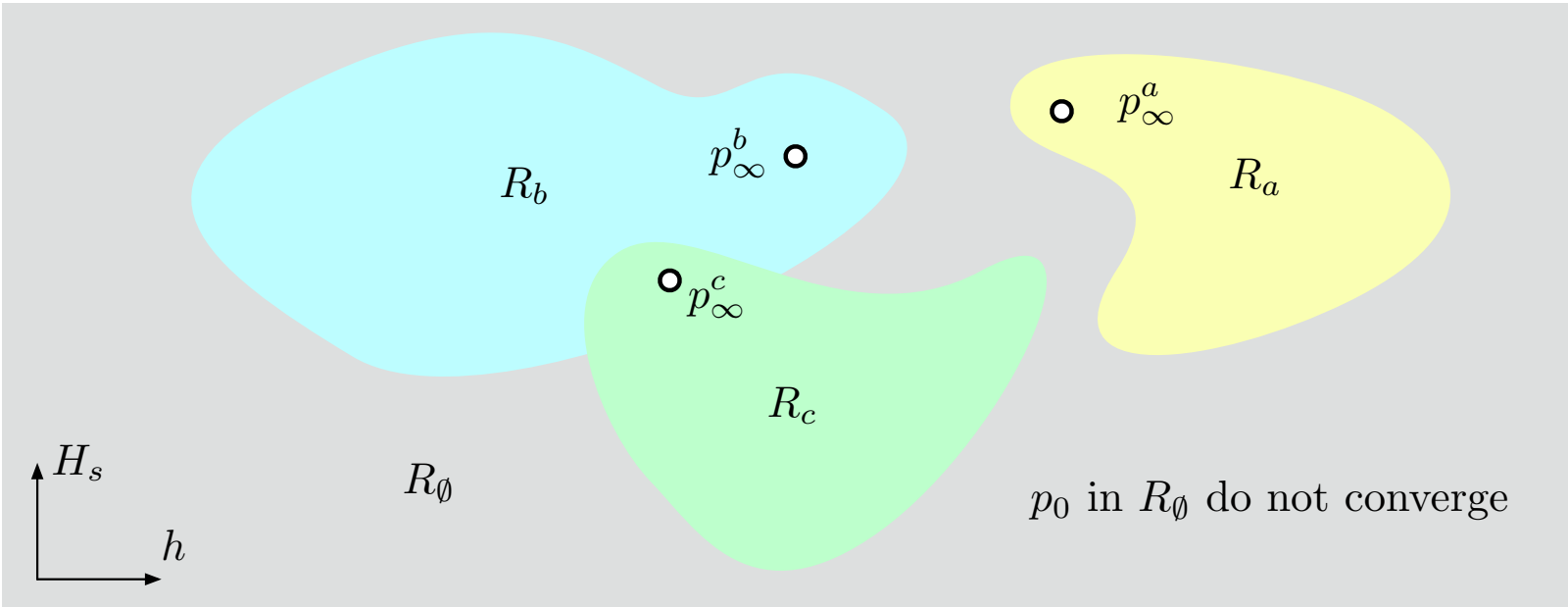
$$p_n = \Xi(p_{n-1}).$$



Question 1: For what initial values p_0 will p_n converge?

Second Question

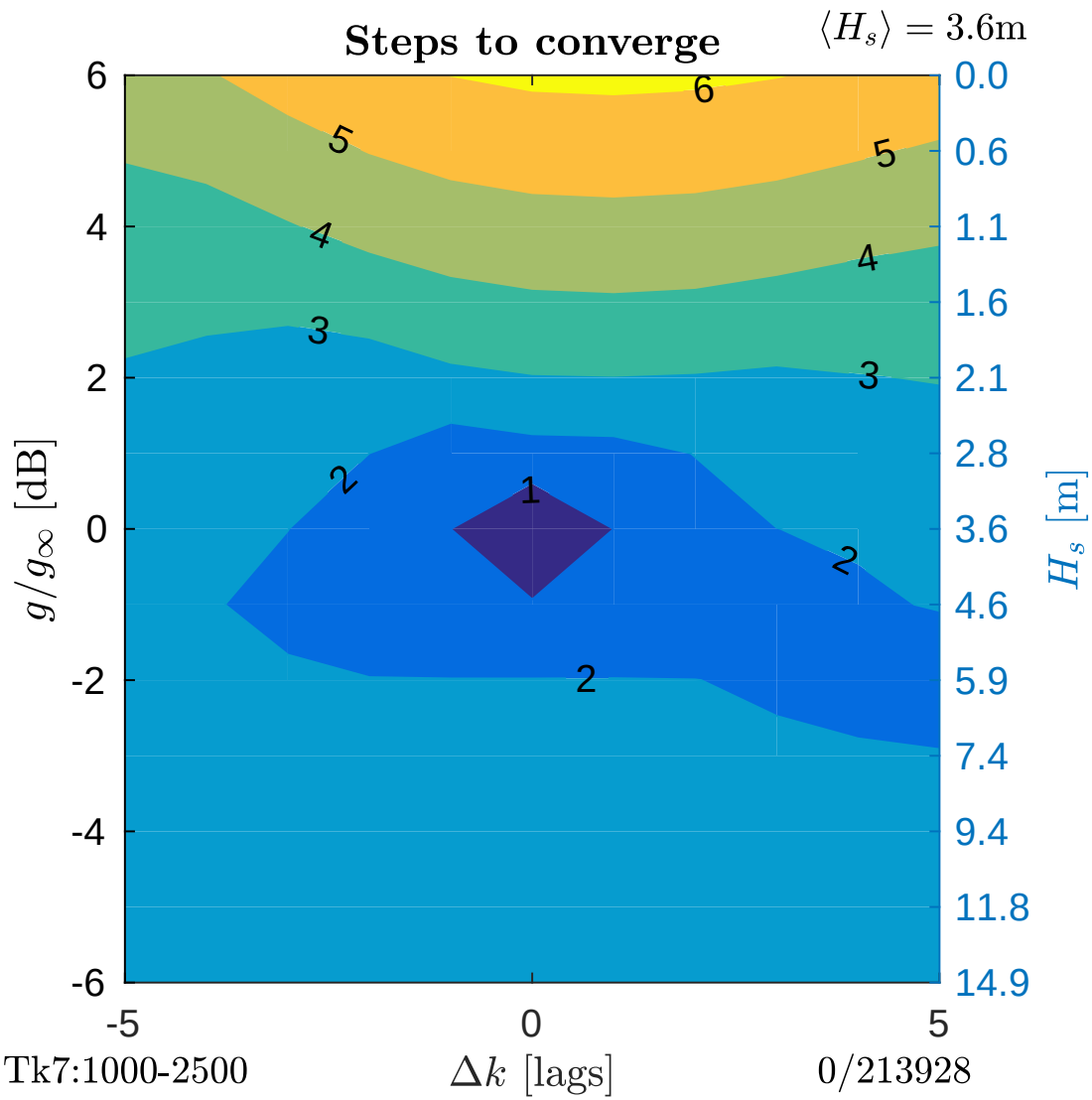
Question 2: For those p_0 that do converge, do they converge to the same value of p_∞?



We can divide the parameter space into mutually exclusive regions, all those points that converge to the same point form a region.

Results

Below is a contour graph of the mean number of steps to converge over a range of initial parameter values.



Conclusions regarding ACDC Stability

- There is only one region, all p_0 lead to the same p_∞. ACDC is stable.

- Sea states with low significant wave height converge more slowly than those with large significant wave heights.

DeDop User Experience

Because the ACDC method is fundamentally different the signal processing needs to be different from L1b. Thus a nonstandard processor would be needed.

The intent in doing this case study was to see if an "outsider" would be able to alter the DeDop base system to give it new functionality.

The test was a failure, due to the users inability to understand the structure of the source code.

Conclusions Regarding User Experience

- The current DeDop documentation is sufficient to install DeDop tool from binary and process a track with the default settings. The configuration panel of DeDop Studio is nice, but without documentation on what the effect of the various flags will be, it is of limited use to anyone who did not write the code. One can of course try running with different flags set and see the results, but without understanding what is being done, the consequences of the action have little significance.

- Following the instructions provided in the DeDop installation guide and the associated installation guides for Miniconda and Github it was possible to install DeDop from source without too much fuss, except when the installation guide got out of sync with the evolving DeDop.

Conclusions Regarding User Experience

- There are quite a number of independent systems with which the user needs to be proficient in order to be able to extend the DeDop tool: Miniconda, Github, Python, a Python editor, Jupyter, and DeDop. Even with excellent documentation it takes some time to find one's bearings in a new environment. To this user all of the systems were unknown.

- The auto-generated documentation of the DeDop code is not particularly helpful in understanding the code. In browsing the code I was able to make very little headway in understanding the structure. I was new to Python when coming to this project, even so I have worked extensively in many languages. Perhaps to someone with previous Python experience the structure of the DeDop code would be self evident, but I still expect it would be difficult. If it was desired that others would extend DeDop I think human written documentation would be essential.


De Dop