Geometric attributes: Program filter_dip_components

IMPROVING DIP ESTIMATES – PROGRAM
filter_dip_components

Contents

Computation flow chart................................................................................................................................................1
Computing mean, median, LUM, and other filtered dip volumes.................................................................................. 2
  Theory: Review of linear and nonlinear filters .................................................................................................................. 6
References: ...................................................................................................................................................................... 13

Computation flow chart

The input to program filter_dip_components is generally computed from program dip3d and includes estimates of the inline and crossline dip components as well as the confidence of these estimates. The confidence computed in program dip3d is simply the semblance along dip of the analytic traces that fall within the (potentially uncentered) Kuwahara window used in the computation. Program filter_dip_components can be run iteratively, whereby the output can be used as input for the next iteration.
Geometric attributes: Program **filter_dip_components**

![Diagram showing geometric attributes](image)

**Computing mean, median, LUM, and other filtered dip volumes**

In the **aaspi_util** GUI, under the **Geometric Attributes** tab, choose program **filter_dip_components**.
Geometric attributes: Program \textit{filter\_dip\_components}

The following window appears:

\textit{filter\_dip\_components} has three input files: (1) inline and (2) crossline components of dip and the (3) confidence (analytic semblance) of the estimate. There will be three output files – the filtered inline and crossline components of dip and an updated confidence estimate. I’ve set the \textit{Suffix} to be ‘1’ indicating that this is the first pass of filtering. The possible filters at present include LUM (lower-upper-middle), MSMTM (multistage median-based modified trimmed mean), median and mean filters. al-Dossary and Marfurt (2007) show the applicability of LUM and MSMTM filters.

Among the parameters, (7) the confidence, \textit{alpha}, is active for all the filters in the list; \textit{alpha} does not work on the values of dip as in an alpha-trimmed mean filter, but rather on the confidence estimate. For the default value of \textit{alpha}=0.5, the values that fall within the analysis window are sorted according to their confidence. If the confidence falls below \textit{alpha}=0.5 of the most confident estimate of dip, we reject it. For those values for which we are quite confident, we take the selected filtered value as our output. The default window size consists of the neighboring traces and samples, in this case +/-25m and +/- 0.02 s.
Geometric attributes: Program filter_dip_components

If you have selected the LUM filter, then the (8) beta value becomes active. If we set beta to be 50%, the result will be the same as using the median filter, where as if we set it to 0%, the result will be as if we had not filtered the data. If we set beta to be between 0 – 50%, for example 20%, then values which fall between 20 – 80 % of the confidence estimate will be kept. Values that fall below 20% of the confidence estimate will be set to the lower threshold 20% confidence value, and values that fall above 80% of the confidence estimate will be set to the upper threshold 80% confidence value. Values that fall below our lower threshold and above our upper threshold will be clipped.

The MSMTM (Multistage median-based modified trimmed mean) filter is able to preserve detail, meaning it acts as an edge preserving filter, a lineament preserving filter and can smoothen noise. The MSMTM is a modified trimmed mean (MTM) filter that implements a multistage median filter (MSM). A data sample’s value is kept if it lies in the range of \([m – q, m + q]\), where \(m\) is calculated using a MSM filter and \(q\) is a user defined range. Larger values of \(q\) result in some smearing of lineaments through higher amplitude “noise” areas, while smaller values of \(q\) better preserve narrow lineaments. For further discussion, please refer to al-Dossary and Marfurt (2007).

The Parallelization parameters panel only asks for the list of nodes and the number of processors per node:
Geometric attributes: Program filter_dip_components

Like all AASPI codes, click Execute and intermediate information will be printed in the xterm from which aaspi_util was launched:

```
3 jobs preloaded
7 jobs preloaded

O: first_line, current_line, last_line, ETA
  105 110 201 0.000 h
O: first_line, current_line, last_line, ETA
  105 120 201 0.000 h
O: first_line, current_line, last_line, ETA
  105 130 201 0.002 h
O: first_line, current_line, last_line, ETA
  105 140 201 0.002 h
O: first_line, current_line, last_line, ETA
  105 150 201 0.000 h
O: first_line, current_line, last_line, ETA
  105 160 201 0.001 h
O: first_line, current_line, last_line, ETA
  105 170 201 0.001 h
O: first_line, current_line, last_line, ETA
  105 180 201 0.001 h
O: first_line, current_line, last_line, ETA
  105 190 201 0.000 h
O: first_line, current_line, last_line, ETA
  105 200 201 0.000 h
```

1 read loop over lines
1 number of traces processed: 1449
process task time (hr) time/trace (s)
1: read data 0.000 0.000
1: send data via MPI 0.000 0.000
1: receive data via MPI 0.000 0.000
1: send results via MPI 0.000 0.000
1: receive results via MPI 0.000 0.000
1: calculate attributes 0.000 0.000
1: write results to disk 0.000 0.000
1: total time 0.004 0.004

1: memory residing only on slaves deallocated
1: attempt to deallocate g.out
1: attempt to deallocate q.out
1: attempt to deallocate conf.out
1: attempt to deallocate line_index

in memory deallocated
lag_index deallocated
lag_interp deallocated
lag_interp, start cdp, end cdp deallocated
1: shared group residing on both master and slave deallocated
8: read loop over lines
8 number of traces processed: 1552
process task time (hr) time/trace (s)
0: read data 0.000 0.000
0: send data via MPI 0.000 0.000
0: receive data via MPI 0.000 0.000

Once the job is completed, typing ls –litr at the terminal prompt shows that the following files were created:

```
-rw-r--r-- 1 kmaefurt aaspi  51 Aug  3 16:10 2000rocessor_list
-rw-r--r-- 1 kmaefurt aaspi  52 Aug  3 16:10 inline_dip_median_filt_bonavouille_1.BAS
-rw-r--r-- 1 kmaefurt aaspi  52 Aug  3 16:10 inline_dip_median_filt_bonavouille_1.H
-rw-r--r-- 1 kmaefurt aaspi  52 Aug  3 16:10 dip_magnitude_median_filt_bonavouille_1.BAS
-rw-r--r-- 1 kmaefurt aaspi  52 Aug  3 16:10 dip_magnitude_median_filt_bonavouille_1.H
-rw-r--r-- 1 kmaefurt aaspi  53 Aug  3 16:10 dip_magnitude_median_filt_bonavouille_1.BAS
-rw-r--r-- 1 kmaefurt aaspi  53 Aug  3 16:10 dip_magnitude_median_filt_bonavouille_1.H
-rw-r--r-- 1 kmaefurt aaspi  53 Aug  3 16:10 dip_magnitude_median_filt_bonavouille_1.BAS
-rw-r--r-- 1 kmaefurt aaspi  53 Aug  3 16:10 dip_magnitude_median_filt_bonavouille_1.H
```

Note that we have created filtered versions of the inline dip and crossline dip components. The part of the name median_filt denotes the kind of filter that was applied. Had we applied a LUM filter, we would see lum_filt instead. Program filter_dip_components also generates new versions of dip magnitude and dip azimuth computed from the filtered dip component volumes.
**Theory: Review of linear and nonlinear filters**

Let’s assume we have $J$ voxels that fall within a 2D or 3D analysis window. There are several linear and nonlinear filters that can be applied.

**The mean filter**

The mean filter is the simplest, where the mean $\mu$ of $J$ samples $d_j$ is defined as:

$$\mu = \frac{1}{J} \sum_{j=1}^{J} d_j.$$  \hspace{1cm} (1)

The mean filter is a smoothing filter, and may not only smooth across faults but smooth in erroneous spikes into the output.

**The median filter**

The first step of the median filter is to sort the data vector, $d$, into a new vector $u$ where $u_k \leq u_{k+1}$:

$$u = \text{sort}\{d_1, d_2, \ldots, d_j, \ldots, d_{J-1}, d_J\}.$$  \hspace{1cm} (2)

Then the median, $m$, is defined as:

$$m = u_{(J+1)/2}.$$  \hspace{1cm} (3)

The median filter is an edge-preserving filter and will preserve changes in dips across faults. It also rejects erroneous spikes in the input data.

**The α-trimmed mean filter**

The α-trimmed filter is an extension of the median filter. First, the algorithm sorts the data in ascending order as in equation 2. Then one defines a fraction (usually defined as a percentage) of the data that falls within the range $0 \leq \alpha \leq \frac{1}{2}$.

$$\alpha$$

The filter rejects $\alpha J$ “outliers” on each end of the data vector and computes the mean of the values of $u_j$ with indices $1+\alpha J \leq j \leq J-\alpha (J-1)$:

$$u_{\alpha-\text{trim}} = \frac{1}{J-2\alpha (J-1)} \sum_{j=1+\alpha (J-1)}^{J-\alpha (J-1)} u_j.$$  \hspace{1cm} (5)

The alpha-trimmed mean filter thus rejects outliers and smooths the remaining values. As such it may still smooth changes in dip across faults.

**The Lower-Upper-Median (LUM) filter:**

The LUM filter is the default filter in `filter_dip_components` and acts in the following manner:

$$u_{\text{LUM}} = \text{median}\left(\{u_{1+\alpha (J-1)}, u^*, u_{J-\alpha (J-1)}\}\right) = \begin{cases} u_{1+\alpha (J-1)} & u^* < u_{1+\alpha (J-1)} \\ u_{J-\alpha (J-1)} & u^* > u_{J-\alpha (J-1)} \\ u^* & 0 \leq \alpha \leq 0.5 \\
\text{otherwise} & \end{cases}$$  \hspace{1cm} (6)

Like the alpha-trimmed mean filter, the LUM filter rejects high and low amplitude “outliers”. Instead of taking the mean of the remaining samples, it compares the dip value at the center of the analysis window $u^*$ to the upper and lower percentiles. If $u^*$ falls beyond these percentiles, it clips the value to the upper or lower percentile; otherwise, it leaves the value alone. In this manner, the LUM filter preserves detailed variation, but rejects erroneous values.
Geometric attributes: Program **filter_dip_components**

The results of the median filter look like this (time slice, t = 1.1 sec):

![Median filtered inline dip](image)

Here we see what the result of the LUM filter looks like (time slice, t = 1.1 sec):

![LUM filtered inline dip](image)
And here is the result of the MSMTM filter with $q = 4$ (time slice, $t = 1.1$ sec)

We note that the median filtered image is overall less noisy and smoother, with a little less N-S acquisition footprint. However, it also has somewhat lower resolution than the input image shown previously. In comparison to the median filter, the LUM filtered image shows more acquisition footprint, but it has enhanced the collapse features as well. The MSMTM filter improves in regards to the footprint and shows better details near the collapse features.

Let’s now plot the filtered dip magnitude. Return to the main aaspi_util GUI and select the tab titled ‘AASPI QC Plotting’:
First select the file `dip_magnitude_median_filt_boonsville_1.H` to plot. Dip magnitude will be strictly positive, so we will want to (2) set the *All positive?* option to *y*. Finally (3) we will want to plot dip magnitude against a white-gray-black colorbar so that flat dips appear as white. The resulting image looks like this (see next page):
Geometric attributes: Program filter_dip_components

The very steep dip (black areas) corresponds to collapse features. Let’s now plot the dip azimuth. Our AASPI QC Plotting GUI looks like following image (see next page):
Enter the (1) file name `dip_azimuth_median_filt_boonsville_1.H` as the AASPI Input. Then (2) choose the cyclic.sep color bar so that $-180^\circ$ will plot with the same color as $+180^\circ$ (yellow for this colorbar). (3) Turn the Auto – Scaling? to be Off to turn off the histogram scaling and instead use explicit clipping. The ranges of these attributes are (4) $-180^\circ$ to $+180^\circ$. The result will look like the following image (see next page):
A drawback of dip azimuth is that it is meaningless when the dip magnitude is very close to 0.0 (the white areas in the dip magnitude image). We can better visualize these areas by using transparency and blending the two images:

However, this image is somewhat disappointing in that the areas of greater dip where the dip azimuth estimates are accurate are now blackened out. We can ameliorate this problem by
Geometric attributes: Program \texttt{filter\_dip\_components}

plotting the dip magnitude against a black-gray-white color bar, thereby rendering the strong dip-magnitude areas more pastel:

![Image of filtered dip azimuth](image)

However, the image is still less than ideal.

References