Interpolating Lidar Point Clouds

Various interpolation methods of lidar point-cloud data using computative efficient approaches

Bodo Bookhagen (bodo.bookhagen@uni-potsdam.de),

Taylor Smith (tasmith@uni-potsdam.de)

April 2019

Table of Contents

1 Preparing Point Cloud Data

1.1 Ground classification

The original (pre-classified) files from the USGS data / opentopography website were reclassified using LASTools lasground. Note that this is a commercial software that requires a license. Tests indicate that ground-classification with pdal and the Progressive Morphological Filter (PMF) provides similar results. In short, the steps were:

```
cd /raid/lidar_research/lidar_data/usgs_channel_islands/processed/SANTA_CRUZ
mkdir cl2_july2018
cd cl2_july2018

mkdir tiles
wine /opt/LAStools/bin/lastile.exe -set_classification 0 -flag_as_withheld -tile_size  \
    500 -buffer 10 -cores 8 -i ../unclass_og/ARRA*.laz -olaz -odir tiles
```

```
# quick overview by thinning file (keep lowest points)
wine /opt/LAStools/bin/lasthin.exe -sparse -step 30 -lowest -i tiles/*.laz -olaz  \
    -merged -olaz -o SCI_ARRA_noise_30m_lowest.laz
wine /opt/LAStools/bin/blast2dem.exe -hillshade -utm 11N -nad83 -meter  \
    -elevation_meter -merged -step 30 -i SCI_ARRA_5m_lowest.laz -o  \
    dtm_interp/SCI_USGS_UTM11_NAD83_lowest5m_30m_HS.tif
wine /opt/LAStools/bin/blast2dem.exe -utm 11N -nad83 -meter -elevation_meter -merged  \
    -step 30 -i SCI_ARRA_5m_lowest.laz -o  \
    dtm_interp/SCI_USGS_UTM11_NAD83_lowest5m_30m.tif
gdalinfo -hist -stats dtm_interp/SCI_USGS_UTM11_NAD83_lowest5m_30m.tif
gdalinfo -hist -stats dtm_interp/SCI_USGS_UTM11_NAD83_lowest5m_30m_HS.tif

mkdir tilesn
wine /opt/LAStools/bin/lasnoise.exe -cores 12 -i tiles/22*.laz -step_xy 2 -step_z 1  \
    -isolated 5 -olaz -odir tilesn -odix n
wine /opt/LAStools/bin/lasnoise.exe -cores 12 -i tiles/23*.laz -step_xy 2 -step_z 1  \
    -isolated 5 -olaz -odir tilesn -odix n
wine /opt/LAStools/bin/lasnoise.exe -cores 12 -i tiles/24*.laz -step_xy 2 -step_z 1  \
    -isolated 5 -olaz -odir tilesn -odix n
wine /opt/LAStools/bin/lasnoise.exe -cores 12 -i tiles/25*.laz -step_xy 2 -step_z 1  \
    -isolated 5 -olaz -odir tilesn -odix n
wine /opt/LAStools/bin/lasnoise.exe -cores 12 -i tiles/26*.laz -step_xy 2 -step_z 1  \
    -isolated 5 -olaz -odir tilesn -odix n

#Here we are using a custom classification with medium-aggressive settings including  \
    high offset, medium standard dev, and medium spike: CHANNELS do not come out good,  \
    but little vegetation
#Tests indicate that this is LIKELY BEST CANDIDATE for channel extraction. There is  \
    some vegetation, but channels are clear
mkdir ground_overlap
wine /opt/LAStools/bin/lasground.exe -cores 12 -i tilesn/22*n.laz -by_flightline  \
    -wilderness -extra_fine -offset 0.25 -stddev 20 -spike 0.5 -bulge 0.5 -olaz -odir  \
    ground_overlap -odix g 2>&1 | tee lasground_output_22n.out
wine /opt/LAStools/bin/lasground.exe -cores 12 -i tilesn/23*n.laz -by_flightline  \
    -wilderness -extra_fine -offset 0.25 -stddev 20 -spike 0.5 -bulge 0.5 -olaz -odir  \
    ground_overlap -odix g 2>&1 | tee lasground_output_22n.out
wine /opt/LAStools/bin/lasground.exe -cores 12 -i tilesn/24*n.laz -by_flightline  \
    -wilderness -extra_fine -offset 0.25 -stddev 20 -spike 0.5 -bulge 0.5 -olaz -odir  \
    ground_overlap -odix g 2>&1 | tee lasground_output_22n.out
wine /opt/LAStools/bin/lasground.exe -cores 12 -i tilesn/25*n.laz -by_flightline  \
    -wilderness -extra_fine -offset 0.25 -stddev 20 -spike 0.5 -bulge 0.5 -olaz -odir  \
    ground_overlap -odix g 2>&1 | tee lasground_output_22n.out
wine /opt/LAStools/bin/lasground.exe -cores 12 -i tilesn/26*n.laz -by_flightline  \
```

```
    -wilderness -extra_fine -offset 0.25 -stddev 20 -spike 0.5 -bulge 0.5 -olaz -odir  \
    ground_overlap -odix g 2>&1 | tee lasground_output_22n.out

#instead of processing chunks of tiles, one could also use the -lof - list of files  \
    options.
```

For this example, we only process a subset of the data from the Pozo catchment. We clip the Pozo catchment with a shapefile `SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83.shp` and generate an output LAZ file `SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.laz`.

```
cd /raid-everest/lidar_research/lidar_data/usgs_channel_islands/
cd processed/SANTA_CRUZ/cl2_july2018/
ls -1 ground_overlap/*ng.laz > SCI_ground_overlap_filelist.lst
wine /opt/LAStools/bin/lasclip.exe -lof SCI_ground_overlap_filelist.lst -olaz  \
    -drop_withheld -o SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.laz -keep_class  \
    2 -merged -poly SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83.shp
```

1.2 Export to ASCII

```
conda activate PC_py3
```

Convert ground-classified LAS/LAZ to ASCII for GMT processing and compress with `bzip2`, but using parallel bzip2 (pbzip2):

```
wine /opt/LAStools/bin/las2las.exe -i  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.laz -o  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz -oparse xyz
#head -100 SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz  \
    >SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2_100rows.xyz
pbzip2 -7 SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz
```

1.3 Prepare DEM

In order to provide best results and produce overlapping grids, you would want to clip the DEM with a shapefile - this way you ensure that all grids will have the same dimensions. You can clip every output grid with the same shapefile stored in CLIP_SHAPEFILE.

We extract the subset from the interpolated DTM using blast2dem.

```
gdalwarp  \
    /raid-everest/lidar_research/lidar_data/usgs_channel_islands/processed/SANTA_CRUZ/cl2_july2018/dtm_i
    dtm_interp/Pozo_USGS_UTM11_NAD83_g_1m.tif -cutline  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83.shp -cl  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83 -crop_to_cutline -tap -multi -tr 1 1  \
    -t_srs epsg:26911 -co COMPRESS=DEFLATE -co ZLEVEL=7
```

Set `$DATA_BASEDIR` variable:

```
export DATA_BASEDIR=/home/bodo/Dropbox/California/SCI/Pozo/pc_interpolation
```

If the DEM exists already, you can clip it with the shapefile to generate a clipped version that is aligned to integer UTM coordinates (`-tap`):

```
DEM_GRID_IN=$DATA_BASEDIR/dtm_interp/Pozo_USGS_UTM11_NAD83_g_1m.tif
DEM_GRID=$DATA_BASEDIR/dtm_interp/Pozo_USGS_UTM11_NAD83_g_1mc.tif
export CLIP_SHAPEFILE=SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83.shp
gdalwarp -cutline $CLIP_SHAPEFILE -cl SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83  \
    -crop_to_cutline -tap -multi -tr 1 1 -t_srs epsg:26911 $DEM_GRID_IN $DEM_GRID -co  \
    COMPRESS=DEFLATE -co ZLEVEL=7
```

A map of Pozo and the zoom-in area can be generated with GMT5. See the script SCI_Pozo_interpolation_GMT5_plot_DEM_overview_zoom.sh that can be run with

`. gmt5_map_scripts/SCI_Pozo_interpolation_GMT5_plot_DEM_overview_zoom.sh`.

The output figures are stored in the subfolder `figures` and is shown in Figure 1.

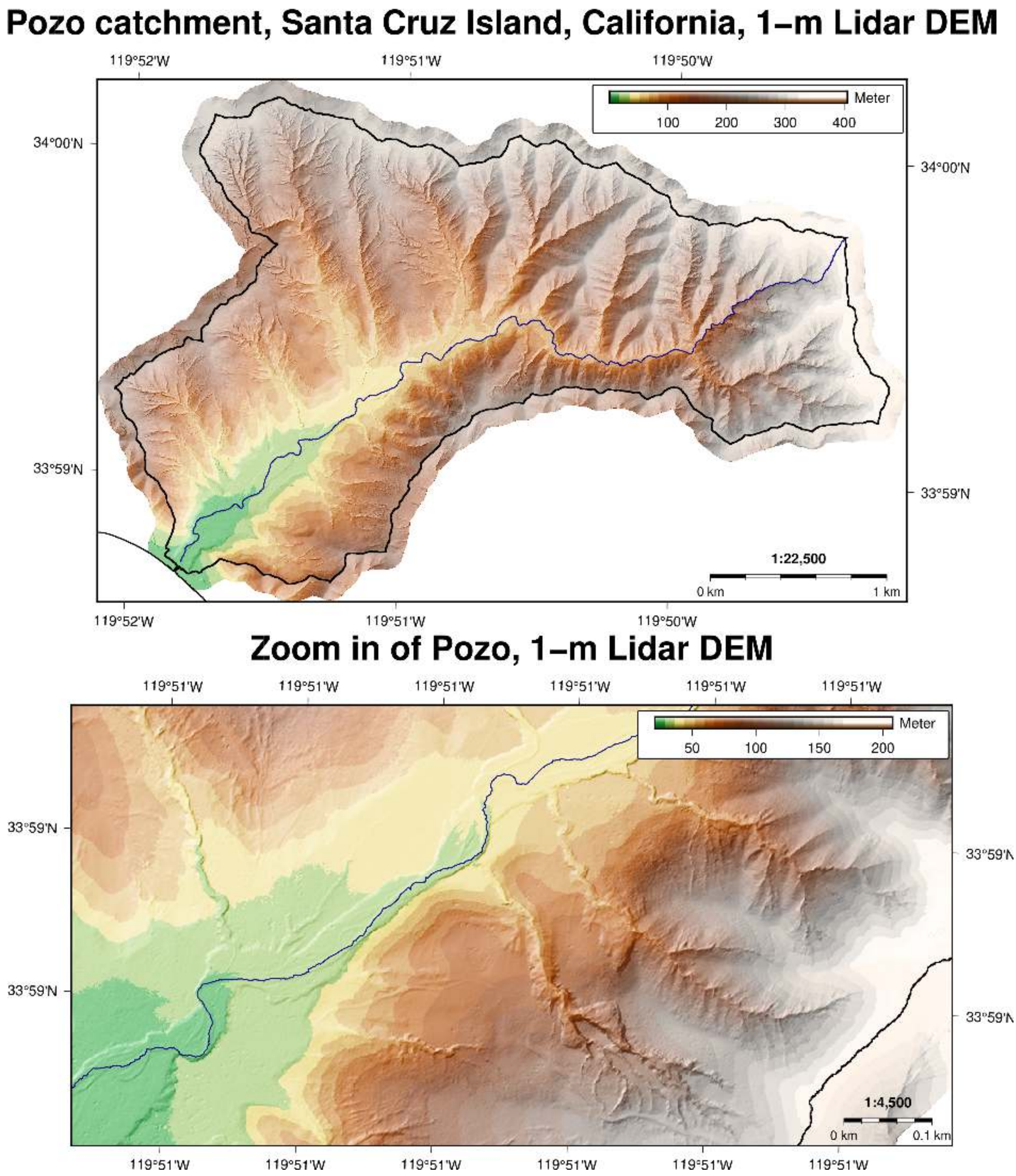


Figure 1: Map view of the Pozo catchment and the zoom-in area in the central part of the catchment.

2 Interpolation of grids with GMT and gdal_grid

2.1 Interpolate with GMT 6

GMT6 has useful interpolation routines with additional output options as compared to GMT5. In addition, you could call these routines directly from Python (not shown in this manual).

GMT6 either has to be compiled from source or installed via anaconda/miniconda. Here, we use miniconda:

```
conda config --prepend channels conda-forge/label/dev
conda create -y -c conda-forge/label/cf201901 -n gmt6 gmt=6* python=3* scipy pandas  \
    numpy matplotlib scikit-image gdal spyder
```

And start the environment:

```
source activate gmt6
```

To keep command lines short, we set the `$DATA_BASEDIR` variable:

```
export DATA_BASEDIR=/home/bodo/Dropbox/California/SCI/Pozo/pc_interpolation
```

Make sure, the DEM exist as NetCDF file:

```
gmt grdconvert $DEM_GRID=gd/1/0/-9999  \
    $DATA_BASEDIR/dtm_interp/Pozo_USGS_UTM11_NAD83_g_1mc.nc
```

2.1.1 GMT blockmean

See http:/gmt.soest.hawaii.edu/doc/5.3.2/blockmean.html

```
mkdir $DATA_BASEDIR/blockmean
```

```
BLOCKMEAN_GRID=$DATA_BASEDIR/blockmean/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_blockmean_1m
pbzip2 -dc SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz.bz2 | gmt blockmean  \
    -R$DEM_GRID -C -G${BLOCKMEAN_GRID}%s.nc -Az,s
```

Convert the NetCDF files to a compress geotiff:

```
cd $DATA_BASEDIR/blockmean
```

```
gdal_translate -of GTIFF  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_blockmean_1mz.nc  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_blockmean_1mz.tif -a_srs  \
    epsg:26911 -co COMPRESS=DEFLATE -co ZLEVEL=7
gdal_translate -of GTIFF  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_blockmean_1ms.nc  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_blockmean_1ms.tif -a_srs  \
    epsg:26911 -co COMPRESS=DEFLATE -co ZLEVEL=7
cd ..
```

A map of the `gmt blockmean` data is generated with gmt5. See the script in the section GMT5 as an example with an output shown in Figures 2 and combined with blockmedian in Figure 4.

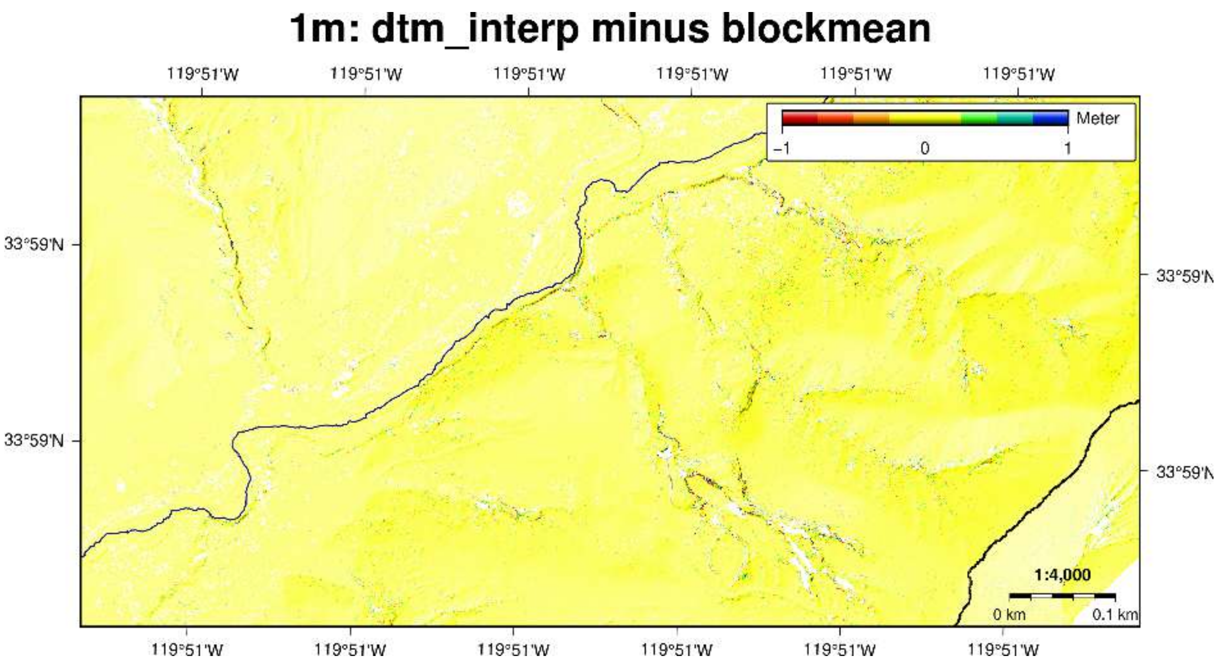


Figure 2: Map view of the LAStools-triangulated minus gmt:blockmean interpolation of the zoomed-in part of the Pozo catchment

2.1.2 GMT blockmedian

http:/gmt.soest.hawaii.edu/doc/5.3.2/blockmedian.html

```
cd $DATA_BASEDIR/
mkdir $DATA_BASEDIR/blockmedian
```

```
DEM_GRID=$DATA_BASEDIR/dtm_interp/Pozo_USGS_UTM11_NAD83_g_1mc.tif
```

```
BLOCKMEDIAN_GRID=$DATA_BASEDIR/blockmedian/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_blockmedi
pbzip2 -dc SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz.bz2 | gmt blockmedian  \
    -R$DEM_GRID -C -G${BLOCKMEDIAN_GRID}%s.nc -Az,s
```

Convert the NetCDF files to a compress geotiff:

```
cd $DATA_BASEDIR/blockmedian
gdal_translate -of GTIFF  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_blockmedian_1mz.nc  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_blockmedian_1mz.tif -a_srs  \
    epsg:26911 -co COMPRESS=DEFLATE -co ZLEVEL=7
gdal_translate -of GTIFF  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_blockmedian_1ms.nc  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_blockmedian_1ms.tif -a_srs  \
    epsg:26911 -co COMPRESS=DEFLATE -co ZLEVEL=7
cd ..
```

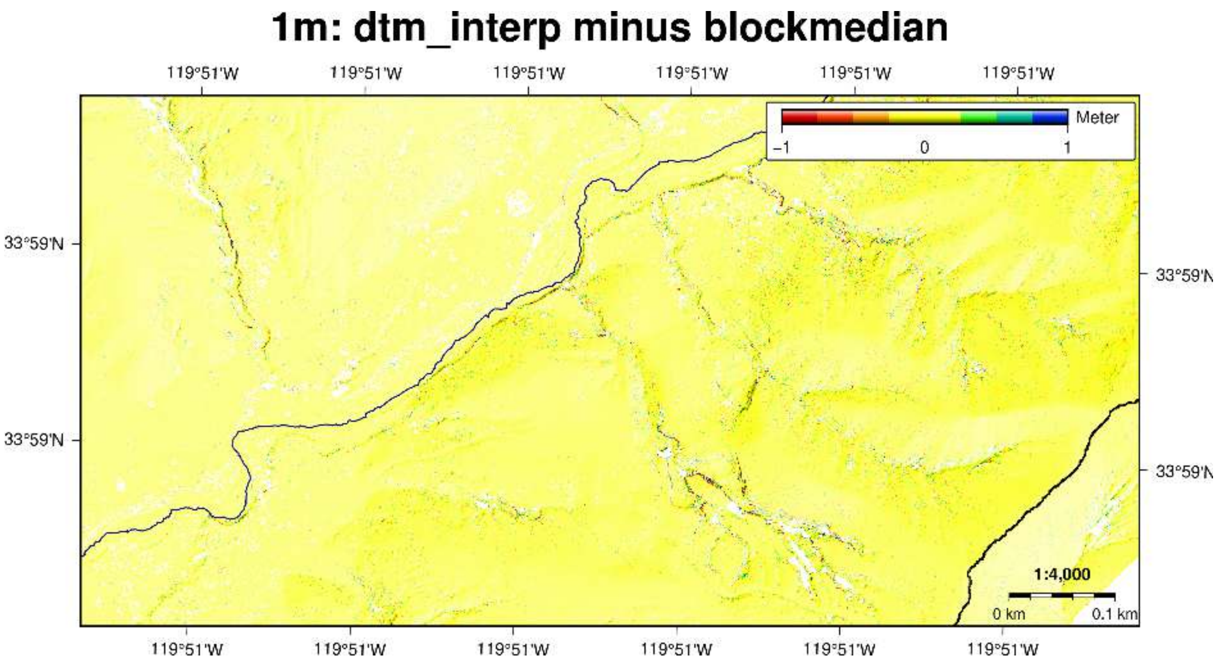


Figure 3: Map view of the LAStools-triangulated minus gmt:blockmedian interpolation of the zoomed-in part of the Pozo catchment.

The gmt:blockmean and gmt:blockmedian interpolated map (see Figure 4).

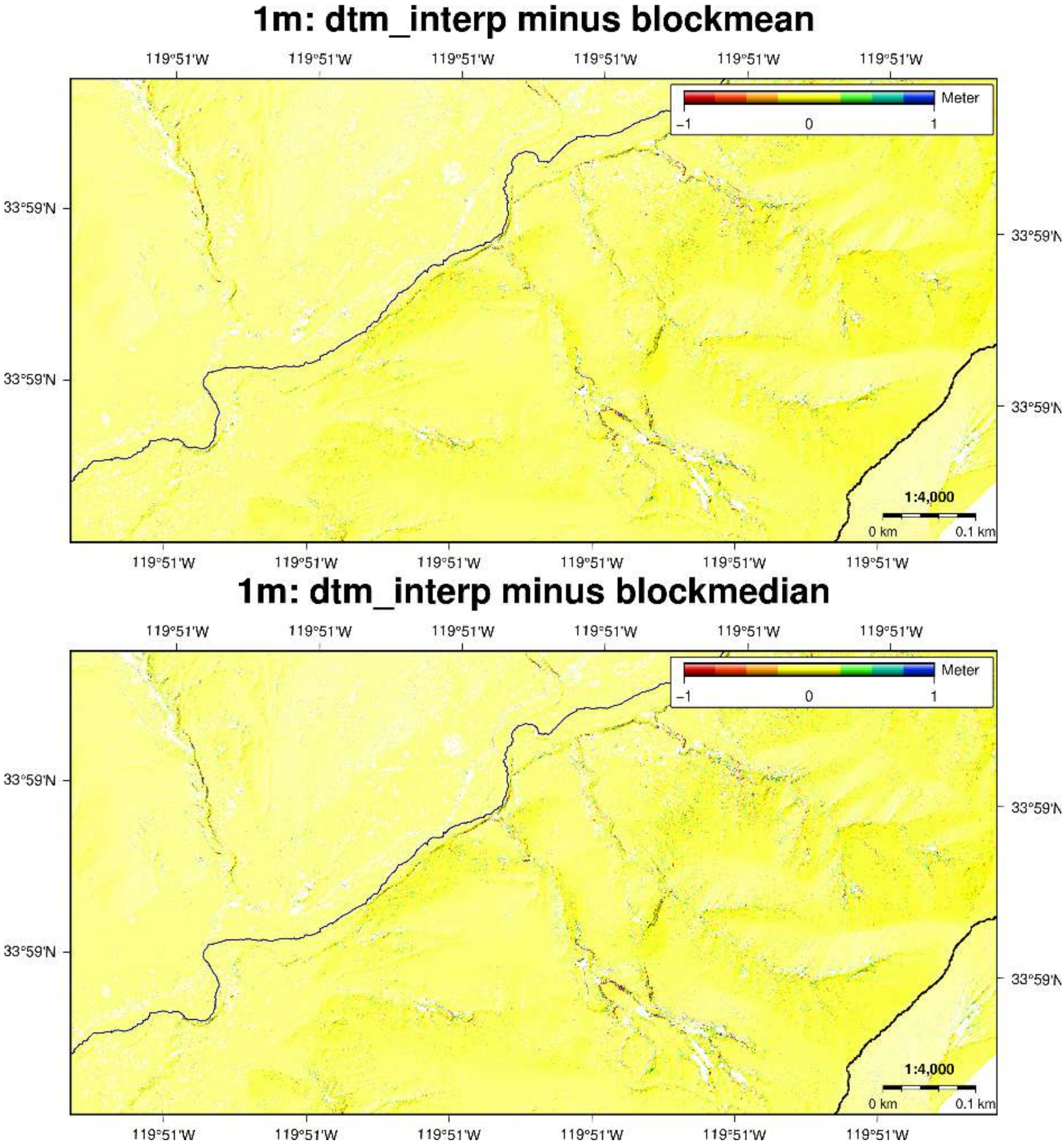


Figure 4: Combined map views of the LAStools-triangulated minus gmt:blockmean and minus gmt:blockmedian interpolation of the zoomed-in part of the Pozo catchment.

2.1.3 GMT Green spline

Not working yet, takes a long time for large points

Greenspline uses the Green's function G(x; x') for the chosen spline and geometry to interpolate data at regular [or arbitrary] output locations. See http:/gmt.soest.hawaii.edu/doc/latest/greenspline.html for more information. Here, we use a minimum curvature spline (`-Sc`) and continuos curvature spline (`-St0.3`) and we only retain the largest eigenvalue when solving the linear system for the spline coefficients by SVD (`-Cn`).

```
cd $DATA_BASEDIR/
mkdir $DATA_BASEDIR/greenspline
```

```
DEM_GRID=$DATA_BASEDIR/dtm_interp/Pozo_USGS_UTM11_NAD83_g_1mc.tif
GREENSPLINE_GRID=$DATA_BASEDIR/greenspline/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_greenspli
pbzip2 -dc SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz.bz2 | gmt greenspline  \
    -R$DEM_GRID -C50+feigenvalue.txt -D1 -Sc -G${GREENSPLINE_GRID}%s.nc

GREENSPLINE_GRID=$DATA_BASEDIR/greenspline/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_greenspli
pbzip2 -dc SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz.bz2 | gmt greenspline  \
    -R$DEM_GRID -Cn -D1 -St0.3 -G${GREENSPLINE_GRID}%s.nc
```

2.1.4 GMT Triangulate

Uses Delauny Triangulation Delaunay, i.e., the algorithm finds how the points should be connected to give the most equilateral triangulation possible. For more information see http:/gmt.soest.hawaii.edu/doc/latest/triangulate.html. This is very similar to the interpolation performed by blast2dem. The actual algorithm used in the triangulations is that of Watson [1982].

```
cd $DATA_BASEDIR/
mkdir $DATA_BASEDIR/triangulation
```

```
DEM_GRID=$DATA_BASEDIR/dtm_interp/Pozo_USGS_UTM11_NAD83_g_1mc.tif
TRIANGULATION_GRID=$DATA_BASEDIR/triangulation/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_trian
pbzip2 -dc SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz.bz2 | gmt triangulate  \
    -R$DEM_GRID -G${TRIANGULATION_GRID}
```

Convert the NetCDF files to a compressed geotiff:

```
cd $DATA_BASEDIR/triangulation
gdal_translate -of GTIFF  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_triangulation_1m.nc  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_triangulation_1m.tif -a_srs  \
    epsg:26911 -co COMPRESS=DEFLATE -co ZLEVEL=7
```

```
cd ..
```

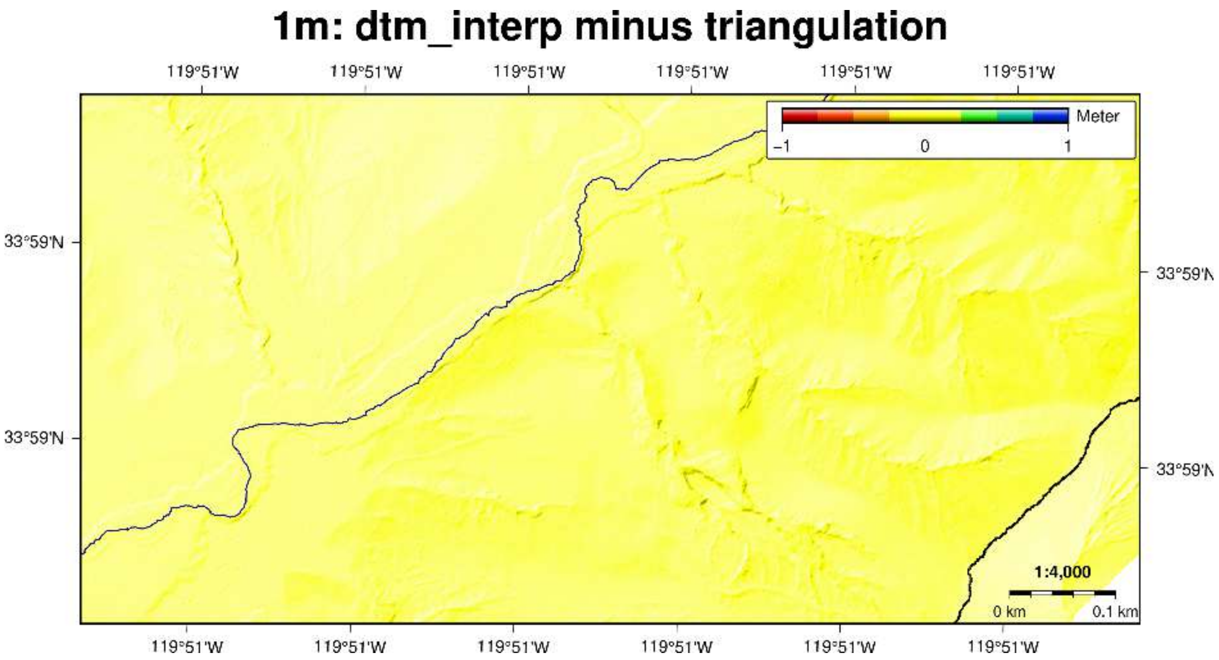


Figure 5: Map view of the LAStools-triangulated minus gmt:blockmedian interpolation of the zoomed-in part of the Pozo catchment.

The DEM difference gmt:triangulation interpolated map (see Figure 5).

2.1.5 GMT Surface

Gridding points using adjustable tension continuous curvature splines. Surface reads randomly-spaced (x,y,z) triples from standard input [or table] and produces a binary grid file of gridded values z(x,y) by solving: (1 - T) * L (L (z)) + T * L (z) = 0, where T is a tension factor between 0 and 1, and L indicates the Laplacian operator. For more information see http:/gmt.soest.hawaii.edu/doc/latest/surface.html. Here

```
cd $DATA_BASEDIR/
mkdir $DATA_BASEDIR/surface
```

```
DEM_GRID=$DATA_BASEDIR/dtm_interp/Pozo_USGS_UTM11_NAD83_g_1mc.tif
SURFACE_GRID=$DATA_BASEDIR/surface/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_surface_tension02
pbzip2 -dc SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz.bz2 | gmt surface  \
    -R$DEM_GRID -G${SURFACE_GRID} -M1c -T0.25 -C0.1
```

Convert the NetCDF files to a compress geotiff:

```
cd $DATA_BASEDIR/surface
gdal_translate -of GTIFF  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_surface_tension025_c01_1m.nc  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_surface_tension025_c01_1m.tif  \
    -a_srs epsg:26911 -co COMPRESS=DEFLATE -co ZLEVEL=7
cd ..
```

Using Tension=0.35:

```
cd $DATA_BASEDIR
DEM_GRID=$DATA_BASEDIR/dtm_interp/Pozo_USGS_UTM11_NAD83_g_1mc.tif
SURFACE_GRID=$DATA_BASEDIR/surface/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_surface_tension03
pbzip2 -dc SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz.bz2 | gmt surface  \
    -R$DEM_GRID -G${SURFACE_GRID} -M0c -T0.35 -C0.1
```

Convert the NetCDF files to a compress geotiff:

```
cd $DATA_BASEDIR/surface
gdal_translate -of GTIFF  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_surface_tension035_c01_1m.nc  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_surface_tension035_c01_1m.tif  \
    -a_srs epsg:26911 -co COMPRESS=DEFLATE -co ZLEVEL=7
cd ..
```

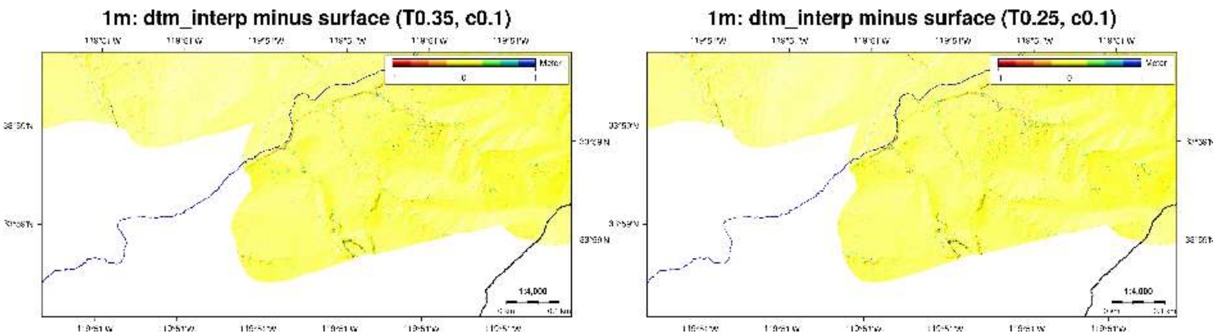


Figure 6: Map view of the LAStools-triangulated minus gmt:surface tension interpolation (t=0.25 and t=0.35) for the Pozo zoom-in area.

The DEM difference of surface tension of the Pozo catchment and the area of interest is shown in Figure 6.

2.1.6 GMT NearestNeighbor interpolation

This is currently not working

http:/gmt.soest.hawaii.edu/doc/latest/nearneighbor.html

The average value is computed as a weighted mean of the nearest point from each sector inside the search radius. The weighting function used is w(r) = 1 / (1 + d ˆ 2), where d = 3 * r / search_radius and r is distance from the node. Distances (-S) are grid-cell size * sqrt(2) / 2 (-`S0.707e`):

```
export DATA_BASEDIR=/home/bodo/Dropbox/California/SCI/Pozo/pc_interpolation
cd $DATA_BASEDIR/
mkdir $DATA_BASEDIR/nearneighbor
```

```
DEM_GRID=$DATA_BASEDIR/dtm_interp/Pozo_USGS_UTM11_NAD83_g_1mc.tif
NEARNEIGHBOR_GRID=$DATA_BASEDIR/nearneighbor/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_gmtnear
pbzip2 -dc SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz.bz2 | gmt nearneighbor  \
    -R$DEM_GRID -G${NEARNEIGHBOR_GRID} -S0.707e -nn -N2+m2
```

Convert the NetCDF files to a compress geotiff:

```
cd $DATA_BASEDIR/nearneighbor
gdal_translate -of GTIFF  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_gmtnearneighbor_1m.nc  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_gmtnearneighbor_1m.tif -a_srs  \
    epsg:26911 -co COMPRESS=DEFLATE -co ZLEVEL=7
cd ..
```

2.2 Interpolate with gdal_grid

2.2.1 NearestNeighbor interpolation using gdal_grid

The above described approach with gmt does not appear to work well. Better to use a gdal_grid with `gdal_grid nearest` approach:

First, you have to set the variables for import/export from gdal:

```
DEM_GRID=$DATA_BASEDIR/dtm_interp/Pozo_USGS_UTM11_NAD83_g_1mc.tif
# get x,y bounds
export minx=`gmt grdinfo -C $DEM_GRID |cut -f 2`
export maxx=`gmt grdinfo -C $DEM_GRID |cut -f 3`
export nx=`gmt grdinfo -C $DEM_GRID |cut -f 10`
```

```
export boundsx="$minx $maxx"
export miny=`gmt grdinfo -C $DEM_GRID |cut -f 4`
export maxy=`gmt grdinfo -C $DEM_GRID |cut -f 5`
export ny=`gmt grdinfo -C $DEM_GRID |cut -f 11`
export boundsy="$miny $maxy"
export boundsyr="$maxy $miny"
```

Next, prepare the file to be read by gdal_grid:

First, add column header with x, y, z to column file containing data:

```
cp -rv SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz  \
    SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.csv
pbzip2 -7 SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz
sed -i '1s/^/x y z\n/' SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.csv
```

Next, Generate a VRT file `SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.vrt` that contains information about the file to be read:

```
<OGRVRTDataSource>
    <OGRVRTLayer name="SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2">
        <SrcDataSource>CSV:SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.csv</SrcDataSource>
        <SrcLayer>SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2</SrcLayer>
        <LayerSRS>EPSG:26911</LayerSRS>
        <GeometryType>wkbPoint</GeometryType>
        <GeometryField encoding="PointFromColumns" x="x" y="y" z="z"/>
    </OGRVRTLayer>
</OGRVRTDataSource>
```

Next, perform the actual interpolation and clip output with `gdalwarp`:

```
PC_IN=SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2
NEARNEIGHBOR_GRID=$DATA_BASEDIR/nearneighbor/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_gdalnea
#We follow the definition of points2grid and assume a radius of spatial resolution *  \
    sqrt(2) / 2
R_M=0.707
export CLIP_SHAPEFILE=SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83.shp
gdal_grid -zfield "z" -a  \
    nearest:radius1=$R_M:radius2=$R_M:min_points=3:max_points=24:nodata=-9999 -txe  \
    $boundsx -tye $boundsyr -outsize $nx $ny -of GTiff -ot Float32 -l ${PC_IN}  \
    ${PC_IN}.vrt ${NEARNEIGHBOR_GRID} -co COMPRESS=DEFLATE -co ZLEVEL=7 --config  \
    GDAL_NUM_THREADS ALL_CPUS --config GDAL_CACHEMAX 2000
```

```
gdalwarp -cutline $CLIP_SHAPEFILE -cl SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83  \
    -crop_to_cutline -tap -multi -tr 1 1 -t_srs epsg:26911 $NEARNEIGHBOR_GRID  \
    ${NEARNEIGHBOR_GRID::-4}_c.tif -co COMPRESS=DEFLATE -co ZLEVEL=7
```

If needed, one can convert to NetCDF GMT grid:

```
gmt grdconvert ${NEARNEIGHBOR_GRID::-4}_c.tif=gd/1/0/-9999 ${NEARNEIGHBOR_GRID}.nc
```

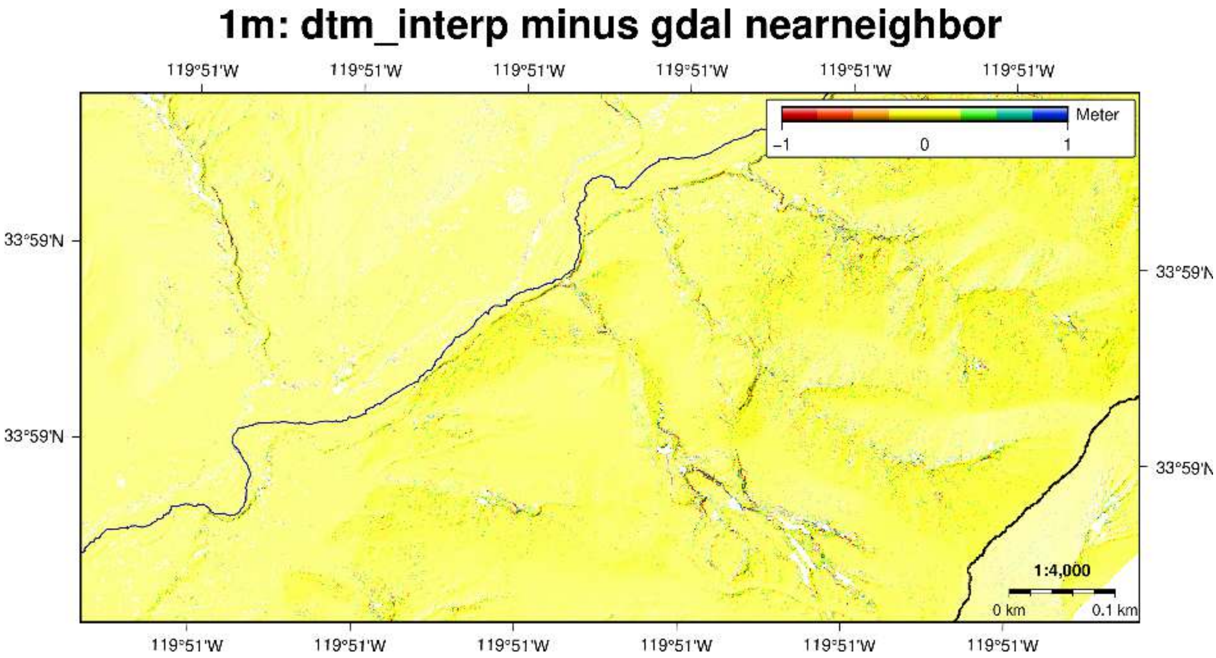


Figure 7: Map view of the LAStools-triangulated minus gdal_grid:nearneighbor interpolation for the Pozo zoom-in area.

The DEM difference of gdal_grid:nearneighbor of the Pozo catchment and the area of interest is shown in Figure 7.

You may want to consider using a larger radius (R=1.414) to avoid nodata areas:

```
PC_IN=SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2
NEARNEIGHBOR_GRID=$DATA_BASEDIR/nearneighbor/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_gdalnea
#We follow the definition of points2grid and assume a radius of spatial resolution *  \
    sqrt(2) / 2
R_M=1.414
export CLIP_SHAPEFILE=SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83.shp
gdal_grid -zfield "z" -a  \
    nearest:radius1=$R_M:radius2=$R_M:min_points=3:max_points=24:nodata=-9999 -txe  \
    $boundsx -tye $boundsyr -outsize $nx $ny -of GTiff -ot Float32 -l ${PC_IN}  \
```

```
    ${PC_IN}.vrt ${NEARNEIGHBOR_GRID} -co COMPRESS=DEFLATE -co ZLEVEL=7 --config  \
    GDAL_NUM_THREADS ALL_CPUS --config GDAL_CACHEMAX 2000

gdalwarp -cutline $CLIP_SHAPEFILE -cl SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83  \
    -crop_to_cutline -tap -multi -tr 1 1 -t_srs epsg:26911 $NEARNEIGHBOR_GRID  \
    ${NEARNEIGHBOR_GRID::-4}_c.tif -co COMPRESS=DEFLATE -co ZLEVEL=7
```

If needed, one can convert to NetCDF GMT grid:

```
gmt grdconvert ${NEARNEIGHBOR_GRID::-4}_c.tif=gd/1/0/-9999 ${NEARNEIGHBOR_GRID}.nc
```

2.2.2 Interpolate IDW using gdal_grid

Interpolate using gdal_grid with `gdal_grid invdistnn`. For details see section "NearestNeighbor interpolation using gdal_grid".

Here, we assume there exists already a CSV and VRT file:

```
export DATA_BASEDIR=/home/bodo/Dropbox/California/SCI/Pozo/pc_interpolation
DEM_GRID=$DATA_BASEDIR/dtm_interp/Pozo_USGS_UTM11_NAD83_g_1mc.tif
# get x,y bounds
export minx=`gmt grdinfo -C $DEM_GRID |cut -f 2`
export maxx=`gmt grdinfo -C $DEM_GRID |cut -f 3`
export nx=`gmt grdinfo -C $DEM_GRID |cut -f 10`
export boundsx="$minx $maxx"
export miny=`gmt grdinfo -C $DEM_GRID |cut -f 4`
export maxy=`gmt grdinfo -C $DEM_GRID |cut -f 5`
export ny=`gmt grdinfo -C $DEM_GRID |cut -f 11`
export boundsy="$miny $maxy"
export boundsyr="$maxy $miny"

PC_IN=SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2
IDW_GRID=$DATA_BASEDIR/idw/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_idwp2_invdist_1m.tif
#We follow the definition of points2grid and assume a radius of spatial resolution *  \
    sqrt(2) / 2
R_M=0.707
```

NOTE that the next command is the standard way to run IDW, but not the most efficient way. Do not use this, unless you have too much time at hand. Please look at option #1 and #2 below to speed up processing for a large number of points

```
gdal_grid -zfield "z" -a  \
    invdist:power=2.0:smoothin=0.0:radius1=$R_M:radius2=$R_M:nodata=-9999 -txe  \
    $boundsx -tye $boundsyr -outsize $nx $ny -of GTiff -ot Float32 -l ${PC_IN}  \
    ${PC_IN}.vrt ${IDW_GRID} -co COMPRESS=DEFLATE -co ZLEVEL=7 --config  \
    GDAL_NUM_THREADS ALL_CPUS --config GDAL_CACHEMAX 2000
```

Not necessary, but just in case:

```
gdalwarp -cutline $CLIP_SHAPEFILE -cl SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83  \
    -crop_to_cutline -tap -multi -tr 1 1 -t_srs epsg:26911 $IDW_GRID  \
    ${IDW_GRID::-4}_c.tif -co COMPRESS=DEFLATE -co ZLEVEL=7
gmt grdconvert ${IDW_GRID::-4}_c.tif=gd/1/0/-9999 ${IDW_GRID::-4}_c.nc
```

Speeding up processing, option #1: Use a maximum point number 24 (adjust this for larger radii):

```
IDW_GRID=$DATA_BASEDIR/idw/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_idwp2_invdist_1m.tif
export  \
    CLIP_SHAPEFILE=/home/bodo/Dropbox/California/SCI/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83.shp
gdal_grid -clipsrc $CLIP_SHAPEFILE -cl SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83  \
    -zfield "z" -a  \
    invdist:power=2.0:smoothin=0.0:radius1=$R_M:radius2=$R_M:min_points=3:max_points=24:nodata=-9999  \
    -txe $boundsx -tye $boundsyr -outsize $nx $ny -of GTiff -ot Float32 -l ${PC_IN}  \
    ${PC_IN}.vrt ${IDW_GRID} -co COMPRESS=DEFLATE -co ZLEVEL=7 --config  \
    GDAL_NUM_THREADS ALL_CPUS --config GDAL_CACHEMAX 2000
```

Not necessary, but just in case:

```
gdalwarp -cutline $CLIP_SHAPEFILE -cl SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83  \
    -crop_to_cutline -tap -multi -tr 1 1 -t_srs epsg:26911 $IDW_GRID  \
    ${IDW_GRID::-4}_c.tif -co COMPRESS=DEFLATE -co ZLEVEL=7
gmt grdconvert ${IDW_GRID::-4}_c.tif=gd/1/0/-9999 ${IDW_GRID::-4}_c.nc
```

Speeding up processing, option #2: Use a maximum point number and the `invdistnn` algorithm and power=1, 2, 3. By defining a higher power value, more emphasis will be put on the nearest points (i.e., their weights are higher). Thus, nearby data to the pixel center will have the most influence.

```
#Power = 1
IDW_GRID=$DATA_BASEDIR/idw/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_idwp1_invdistnn_1m.tif
export  \
    CLIP_SHAPEFILE=/home/bodo/Dropbox/California/SCI/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83.shp
gdal_grid -zfield "z" -a  \
    invdistnn:power=1.0:smoothin=0.0:radius=$R_M:min_points=3:max_points=24:nodata=-9999  \
    -txe $boundsx -tye $boundsyr -outsize $nx $ny -of GTiff -ot Float32 -l ${PC_IN}  \
```

```
    ${PC_IN}.vrt ${IDW_GRID} -co COMPRESS=DEFLATE -co ZLEVEL=7 --config  \
    GDAL_NUM_THREADS ALL_CPUS --config GDAL_CACHEMAX 2000
gdalwarp -cutline $CLIP_SHAPEFILE -cl SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83  \
    -crop_to_cutline -tap -multi -tr 1 1 -t_srs epsg:26911 $IDW_GRID  \
    ${IDW_GRID::-4}_c.tif -co COMPRESS=DEFLATE -co ZLEVEL=7
gmt grdconvert ${IDW_GRID::-4}_c.tif=gd/1/0/-9999 ${IDW_GRID::-4}_c.nc

#Power = 2
IDW_GRID=$DATA_BASEDIR/idw/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_idwp2_invdistnn_1m.tif
export  \
    CLIP_SHAPEFILE=/home/bodo/Dropbox/California/SCI/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83.shp
gdal_grid -zfield "z" -a  \
    invdistnn:power=2.0:smoothin=0.0:radius=$R_M:min_points=3:max_points=24:nodata=-9999  \
    -txe $boundsx -tye $boundsyr -outsize $nx $ny -of GTiff -ot Float32 -l ${PC_IN}  \
    ${PC_IN}.vrt ${IDW_GRID} -co COMPRESS=DEFLATE -co ZLEVEL=7 --config  \
    GDAL_NUM_THREADS ALL_CPUS --config GDAL_CACHEMAX 2000
gdalwarp -cutline $CLIP_SHAPEFILE -cl SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83  \
    -crop_to_cutline -tap -multi -tr 1 1 -t_srs epsg:26911 $IDW_GRID  \
    ${IDW_GRID::-4}_c.tif -co COMPRESS=DEFLATE -co ZLEVEL=7
gmt grdconvert ${IDW_GRID::-4}_c.tif=gd/1/0/-9999 ${IDW_GRID::-4}_c.nc

#Power = 3
IDW_GRID=$DATA_BASEDIR/idw/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_idwp3_invdistnn_1m.tif
export  \
    CLIP_SHAPEFILE=/home/bodo/Dropbox/California/SCI/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83.shp
gdal_grid -zfield "z" -a  \
    invdistnn:power=3.0:smoothin=0.0:radius=$R_M:min_points=3:max_points=24:nodata=-9999  \
    -txe $boundsx -tye $boundsyr -outsize $nx $ny -of GTiff -ot Float32 -l ${PC_IN}  \
    ${PC_IN}.vrt ${IDW_GRID} -co COMPRESS=DEFLATE -co ZLEVEL=7 --config  \
    GDAL_NUM_THREADS ALL_CPUS --config GDAL_CACHEMAX 2000
gdalwarp -cutline $CLIP_SHAPEFILE -cl SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83  \
    -crop_to_cutline -tap -multi -tr 1 1 -t_srs epsg:26911 $IDW_GRID  \
    ${IDW_GRID::-4}_c.tif -co COMPRESS=DEFLATE -co ZLEVEL=7
gmt grdconvert ${IDW_GRID::-4}_c.tif=gd/1/0/-9999 ${IDW_GRID::-4}_c.nc
```

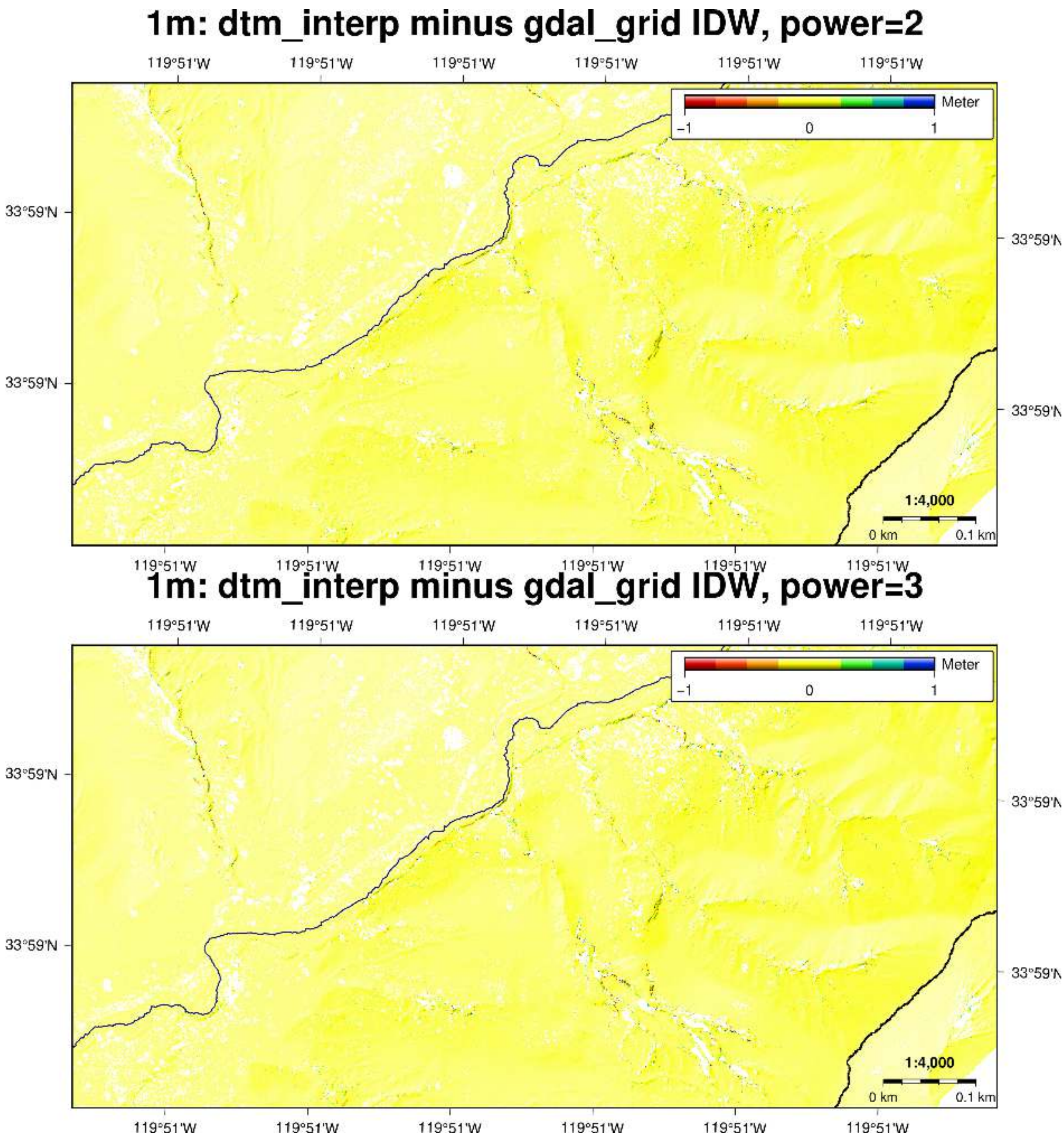


Figure 8: Map view of the LAStools-triangulated minus gdal_grid:idw (power=1, 2 and 3 with radius = 0.707m) interpolation for the Pozo zoom-in area.

The DEM difference of gdal_grid:idw (power=1, 2, 3) of the Pozo catchment and the area of interest is shown in Figure 8.

IDW with larger radii (r=1.414m) to avoid nodata areas

Using the above described approach, we perform the same calculation, but with a larger radius (grid size times sqrt(2), R=1.414):

```
export DATA_BASEDIR=/home/bodo/Dropbox/California/SCI/Pozo/pc_interpolation
DEM_GRID=$DATA_BASEDIR/dtm_interp/Pozo_USGS_UTM11_NAD83_g_1mc.tif
# get x,y bounds
export minx=`gmt grdinfo -C $DEM_GRID |cut -f 2`
export maxx=`gmt grdinfo -C $DEM_GRID |cut -f 3`
export nx=`gmt grdinfo -C $DEM_GRID |cut -f 10`
export boundsx="$minx $maxx"
export miny=`gmt grdinfo -C $DEM_GRID |cut -f 4`
export maxy=`gmt grdinfo -C $DEM_GRID |cut -f 5`
export ny=`gmt grdinfo -C $DEM_GRID |cut -f 11`
export boundsy="$miny $maxy"
export boundsyr="$maxy $miny"
PC_IN=SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2
IDW_GRID=$DATA_BASEDIR/idw/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_idwp2_invdist_1m.tif
R_M=1.414

#Power = 1
IDW_GRID=$DATA_BASEDIR/idw/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_idwp1r1.414_invdistnn_1m.
export  \
    CLIP_SHAPEFILE=/home/bodo/Dropbox/California/SCI/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83.shp
gdal_grid -zfield "z" -a  \
    invdistnn:power=1.0:smoothin=0.0:radius=$R_M:min_points=3:max_points=48:nodata=-9999  \
    -txe $boundsx -tye $boundsyr -outsize $nx $ny -of GTiff -ot Float32 -l ${PC_IN}  \
    ${PC_IN}.vrt ${IDW_GRID} -co COMPRESS=DEFLATE -co ZLEVEL=7 --config  \
    GDAL_NUM_THREADS ALL_CPUS --config GDAL_CACHEMAX 2000
gdalwarp -cutline $CLIP_SHAPEFILE -cl SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83  \
    -crop_to_cutline -tap -multi -tr 1 1 -t_srs epsg:26911 $IDW_GRID  \
    ${IDW_GRID::-4}_c.tif -co COMPRESS=DEFLATE -co ZLEVEL=7
gmt grdconvert ${IDW_GRID::-4}_c.tif=gd/1/0/-9999 ${IDW_GRID::-4}_c.nc

#Power = 2
IDW_GRID=$DATA_BASEDIR/idw/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_idwp2r1.414_invdistnn_1m.
export  \
    CLIP_SHAPEFILE=/home/bodo/Dropbox/California/SCI/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83.shp
gdal_grid -zfield "z" -a  \
    invdistnn:power=2.0:smoothin=0.0:radius=$R_M:min_points=3:max_points=48:nodata=-9999  \
    -txe $boundsx -tye $boundsyr -outsize $nx $ny -of GTiff -ot Float32 -l ${PC_IN}  \
    ${PC_IN}.vrt ${IDW_GRID} -co COMPRESS=DEFLATE -co ZLEVEL=7 --config  \
    GDAL_NUM_THREADS ALL_CPUS --config GDAL_CACHEMAX 2000
gdalwarp -cutline $CLIP_SHAPEFILE -cl SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83  \
```

```
    -crop_to_cutline -tap -multi -tr 1 1 -t_srs epsg:26911 $IDW_GRID  \
    ${IDW_GRID::-4}_c.tif -co COMPRESS=DEFLATE -co ZLEVEL=7
gmt grdconvert ${IDW_GRID::-4}_c.tif=gd/1/0/-9999 ${IDW_GRID::-4}_c.nc

#Power = 3
IDW_GRID=$DATA_BASEDIR/idw/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_idwp3r1.414_invdistnn_1m.
export  \
    CLIP_SHAPEFILE=/home/bodo/Dropbox/California/SCI/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83.shp
gdal_grid -zfield "z" -a  \
    invdistnn:power=3.0:smoothin=0.0:radius=$R_M:min_points=3:max_points=48:nodata=-9999  \
    -txe $boundsx -tye $boundsyr -outsize $nx $ny -of GTiff -ot Float32 -l ${PC_IN}  \
    ${PC_IN}.vrt ${IDW_GRID} -co COMPRESS=DEFLATE -co ZLEVEL=7 --config  \
    GDAL_NUM_THREADS ALL_CPUS --config GDAL_CACHEMAX 2000
gdalwarp -cutline $CLIP_SHAPEFILE -cl SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83  \
    -crop_to_cutline -tap -multi -tr 1 1 -t_srs epsg:26911 $IDW_GRID  \
    ${IDW_GRID::-4}_c.tif -co COMPRESS=DEFLATE -co ZLEVEL=7
gmt grdconvert ${IDW_GRID::-4}_c.tif=gd/1/0/-9999 ${IDW_GRID::-4}_c.nc
```

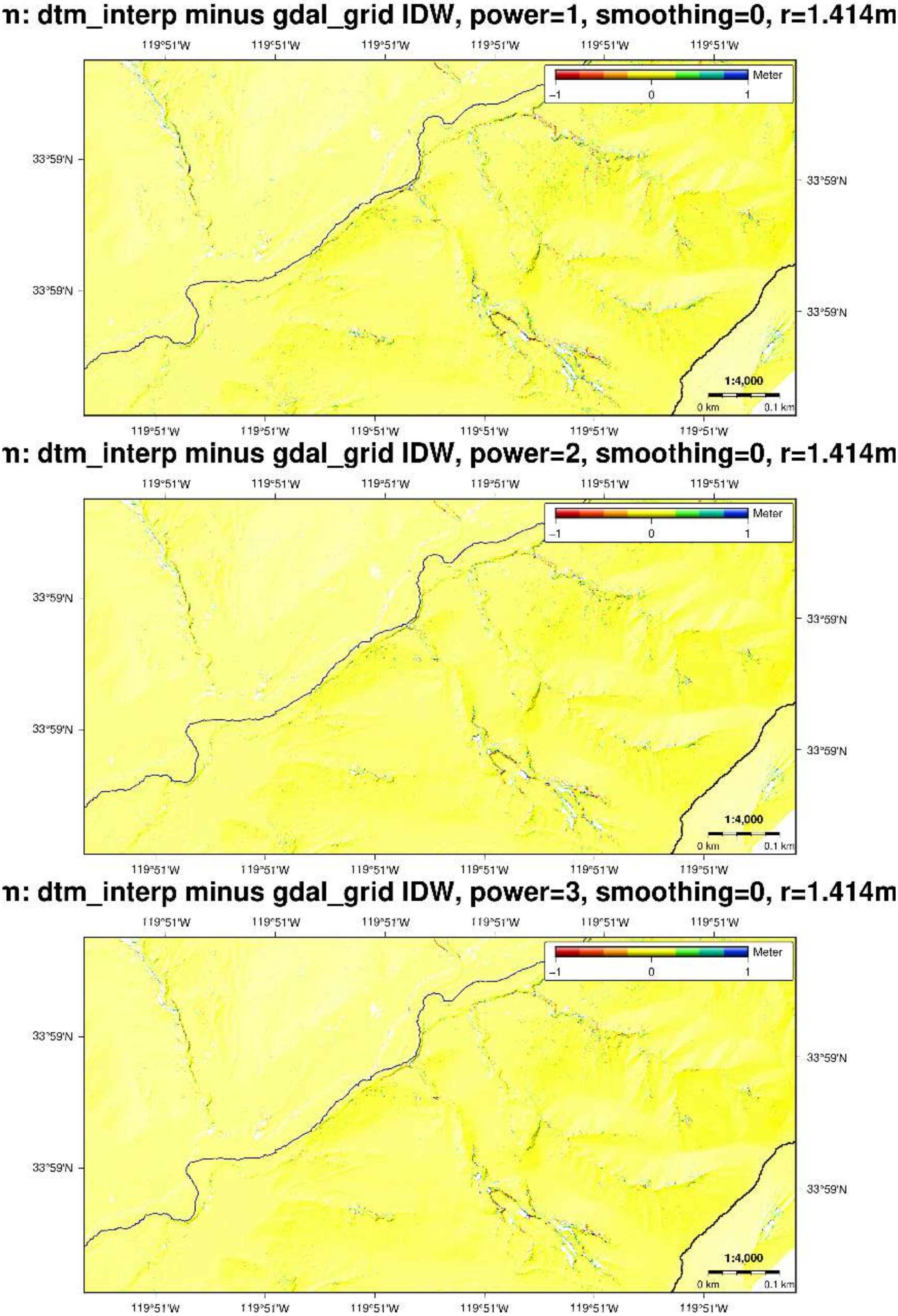


Figure 9: Map view of the LAStools-triangulated minus gdal_grid:idw (power=1, 2, 3) with radius = 1.414m) interpolation for the Pozo zoom-in area.

IDW with larger radii (r=2.828m) to avoid nodata areas

Using the above described approach, we perform the same calculation, but with a larger radius (grid size times sqrt(2) * 2, R=2.828):

```
export DATA_BASEDIR=/home/bodo/Dropbox/California/SCI/Pozo/pc_interpolation
DEM_GRID=$DATA_BASEDIR/dtm_interp/Pozo_USGS_UTM11_NAD83_g_1mc.tif
# get x,y bounds
export minx=`gmt grdinfo -C $DEM_GRID |cut -f 2`
export maxx=`gmt grdinfo -C $DEM_GRID |cut -f 3`
export nx=`gmt grdinfo -C $DEM_GRID |cut -f 10`
export boundsx="$minx $maxx"
export miny=`gmt grdinfo -C $DEM_GRID |cut -f 4`
export maxy=`gmt grdinfo -C $DEM_GRID |cut -f 5`
export ny=`gmt grdinfo -C $DEM_GRID |cut -f 11`
export boundsy="$miny $maxy"
export boundsyr="$maxy $miny"
PC_IN=SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2
IDW_GRID=$DATA_BASEDIR/idw/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_idwp2_invdist_1m.tif
R_M=2.828

#Power = 1
IDW_GRID=$DATA_BASEDIR/idw/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_idwp1r2.828_invdistnn_1m.
export  \
    CLIP_SHAPEFILE=/home/bodo/Dropbox/California/SCI/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83.shp
gdal_grid -zfield "z" -a  \
    invdistnn:power=1.0:smoothin=0.0:radius=$R_M:min_points=3:max_points=96:nodata=-9999  \
    -txe $boundsx -tye $boundsyr -outsize $nx $ny -of GTiff -ot Float32 -l ${PC_IN}  \
    ${PC_IN}.vrt ${IDW_GRID} -co COMPRESS=DEFLATE -co ZLEVEL=7 --config  \
    GDAL_NUM_THREADS ALL_CPUS --config GDAL_CACHEMAX 2000
gdalwarp -cutline $CLIP_SHAPEFILE -cl SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83  \
    -crop_to_cutline -tap -multi -tr 1 1 -t_srs epsg:26911 $IDW_GRID  \
    ${IDW_GRID::-4}_c.tif -co COMPRESS=DEFLATE -co ZLEVEL=7
gmt grdconvert ${IDW_GRID::-4}_c.tif=gd/1/0/-9999 ${IDW_GRID::-4}_c.nc

#Power = 2
IDW_GRID=$DATA_BASEDIR/idw/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_idwp2r2.828_invdistnn_1m.
export  \
    CLIP_SHAPEFILE=/home/bodo/Dropbox/California/SCI/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83.shp
gdal_grid -zfield "z" -a  \
    invdistnn:power=2.0:smoothin=0.0:radius=$R_M:min_points=3:max_points=96:nodata=-9999  \
    -txe $boundsx -tye $boundsyr -outsize $nx $ny -of GTiff -ot Float32 -l ${PC_IN}  \
    ${PC_IN}.vrt ${IDW_GRID} -co COMPRESS=DEFLATE -co ZLEVEL=7 --config  \
```

```
    GDAL_NUM_THREADS ALL_CPUS --config GDAL_CACHEMAX 2000
gdalwarp -cutline $CLIP_SHAPEFILE -cl SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83  \
    -crop_to_cutline -tap -multi -tr 1 1 -t_srs epsg:26911 $IDW_GRID  \
    ${IDW_GRID::-4}_c.tif -co COMPRESS=DEFLATE -co ZLEVEL=7
gmt grdconvert ${IDW_GRID::-4}_c.tif=gd/1/0/-9999 ${IDW_GRID::-4}_c.nc

#Power = 3
IDW_GRID=$DATA_BASEDIR/idw/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_idwp3r2.828_invdistnn_1m.
export  \
    CLIP_SHAPEFILE=/home/bodo/Dropbox/California/SCI/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83.shp
gdal_grid -zfield "z" -a  \
    invdistnn:power=3.0:smoothin=0.0:radius=$R_M:min_points=3:max_points=96:nodata=-9999  \
    -txe $boundsx -tye $boundsyr -outsize $nx $ny -of GTiff -ot Float32 -l ${PC_IN}  \
    ${PC_IN}.vrt ${IDW_GRID} -co COMPRESS=DEFLATE -co ZLEVEL=7 --config  \
    GDAL_NUM_THREADS ALL_CPUS --config GDAL_CACHEMAX 2000
gdalwarp -cutline $CLIP_SHAPEFILE -cl SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83  \
    -crop_to_cutline -tap -multi -tr 1 1 -t_srs epsg:26911 $IDW_GRID  \
    ${IDW_GRID::-4}_c.tif -co COMPRESS=DEFLATE -co ZLEVEL=7
gmt grdconvert ${IDW_GRID::-4}_c.tif=gd/1/0/-9999 ${IDW_GRID::-4}_c.nc
```

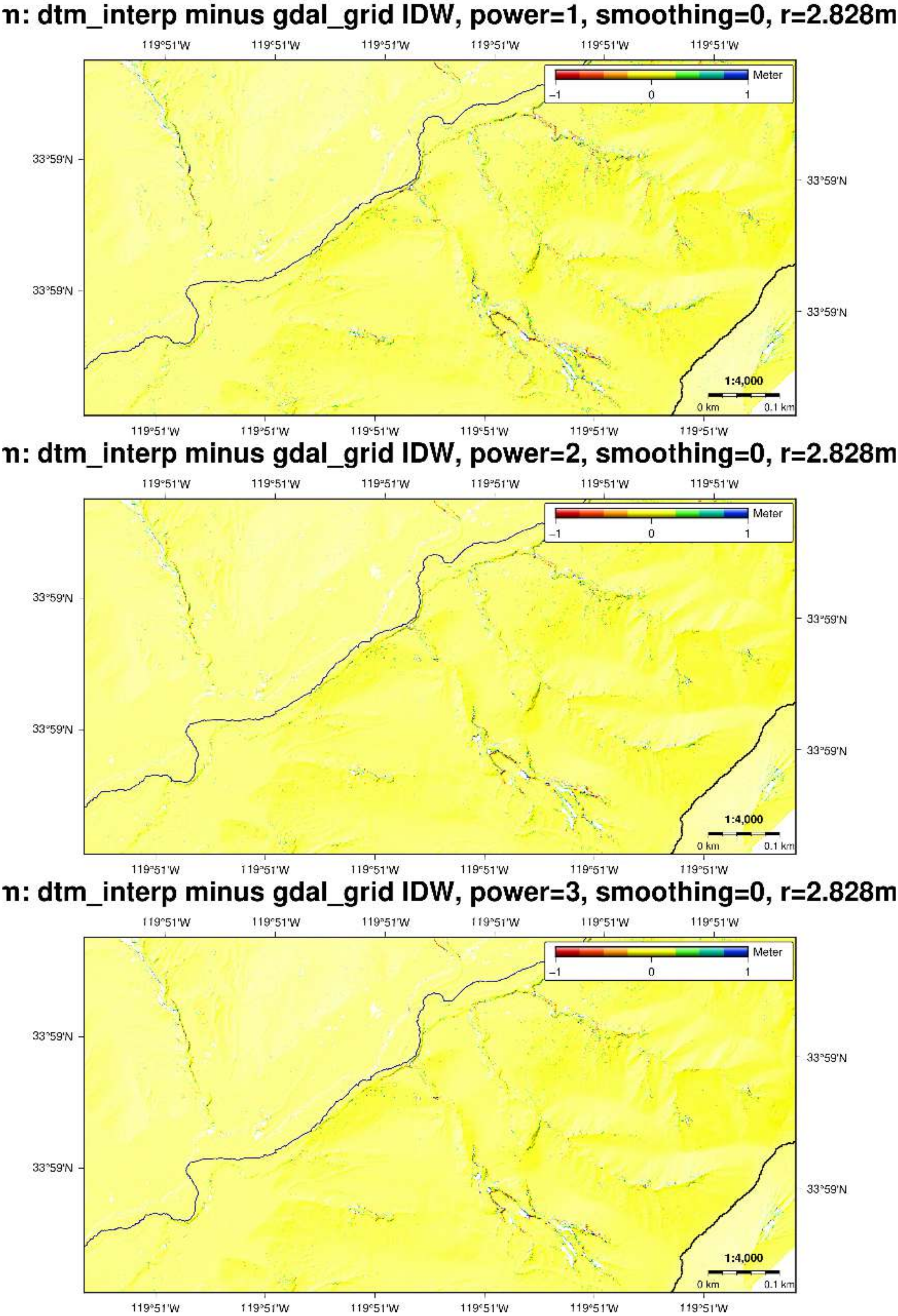


Figure 10: Map view of the LAStools-triangulated minus gdal_grid:idw (power=1, 2, 3) with radius = 2.828m) interpolation for the Pozo zoom-in area.

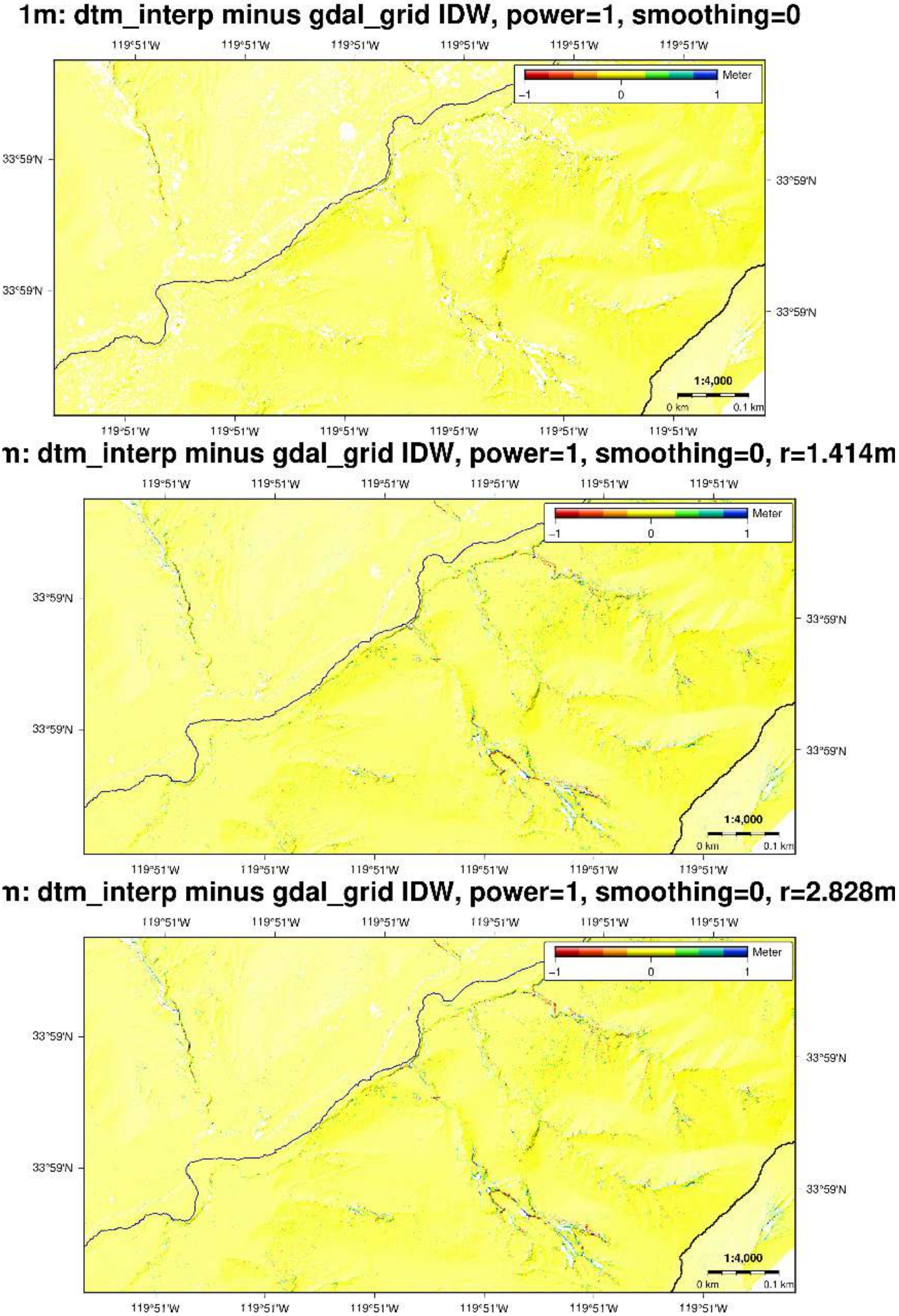


Figure 11: Map view of the LAStools-triangulated minus gdal_grid:idw (power=1, with radii r=0.70m, r=1.41m, r=2.828m) interpolation for the Pozo zoom-in area.

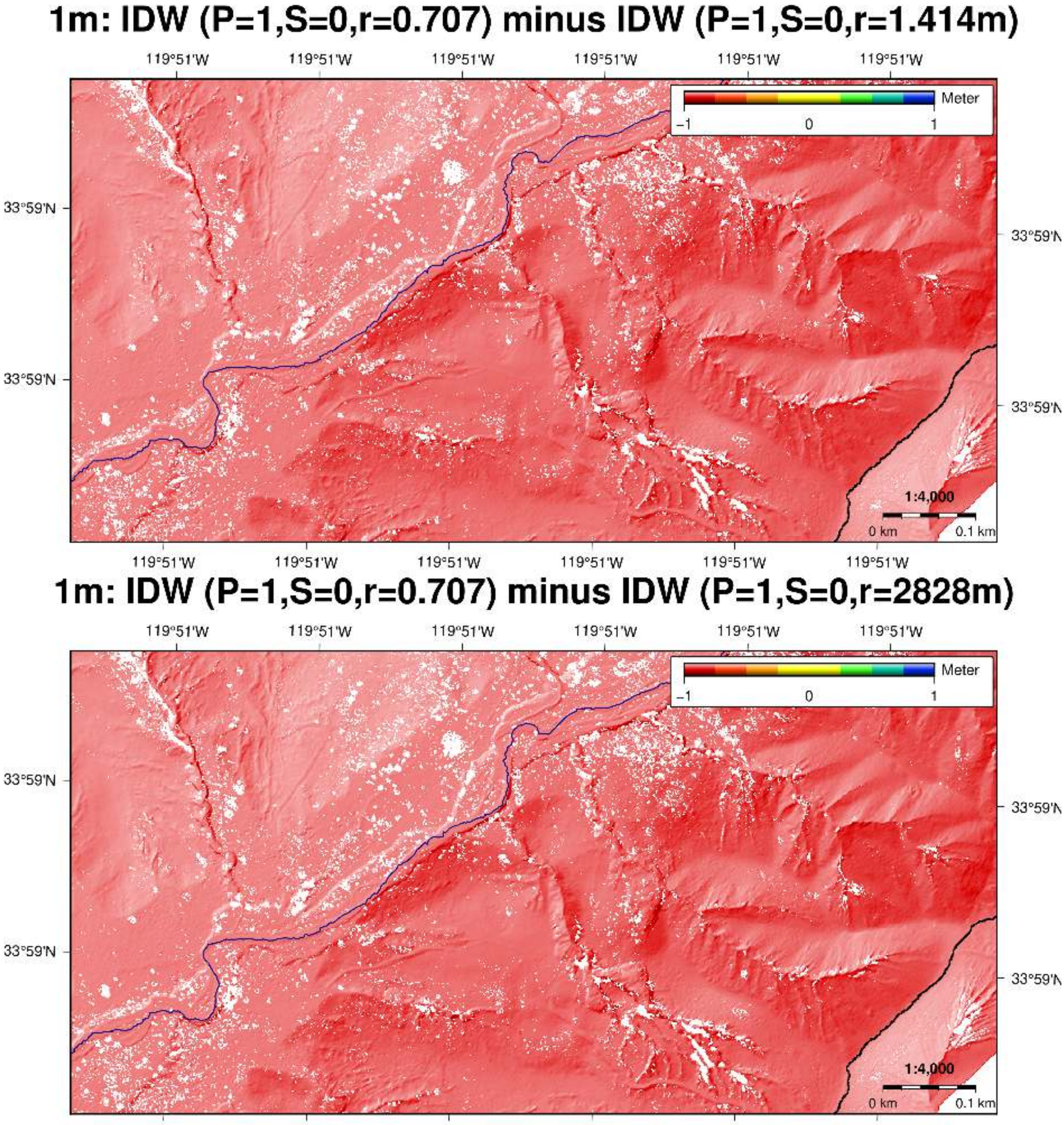


Figure 12: Map view of gdal_grid:idw (power=1, with radius r=0.70m) minus gdal_grid:idw (power=1, with radius r=1.41m) interpolation for the Pozo zoom-in area.

IDW with power=1 and smoothing=1 and 2

In addition to the above example, we explore the smoothing parameter and its effect on the point-cloud data from Pozo. The previous examples rely on no smooth (smoothing=0).

```
#Power = 1 and Smoothing = 1
IDW_GRID=$DATA_BASEDIR/idw/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_idwp1s1_invdistnn_1m.tif
export \
    CLIP_SHAPEFILE=/home/bodo/Dropbox/California/SCI/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83.shp
gdal_grid -zfield "z" -a \
    invdistnn:power=1.0:smoothin=1.0:radius=$R_M:min_points=3:max_points=24:nodata=-9999 \
    -txe $boundsx -tye $boundsyr -outsize $nx $ny -of GTiff -ot Float32 -l ${PC_IN} \
    ${PC_IN}.vrt ${IDW_GRID} -co COMPRESS=DEFLATE -co ZLEVEL=7 --config \
    GDAL_NUM_THREADS ALL_CPUS --config GDAL_CACHEMAX 2000
gdalwarp -cutline $CLIP_SHAPEFILE -cl SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83 \
    -crop_to_cutline -tap -multi -tr 1 1 -t_srs epsg:26911 $IDW_GRID \
    ${IDW_GRID::-4}_c.tif -co COMPRESS=DEFLATE -co ZLEVEL=7
gmt grdconvert ${IDW_GRID::-4}_c.tif=gd/1/0/-9999 ${IDW_GRID::-4}_c.nc

#Power = 1 and Smoothing = 2
IDW_GRID=$DATA_BASEDIR/idw/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.xyz_idwp1s2_invdistnn_1m.tif
export \
    CLIP_SHAPEFILE=/home/bodo/Dropbox/California/SCI/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83.shp
gdal_grid -zfield "z" -a \
    invdistnn:power=1.0:smoothin=2.0:radius=$R_M:min_points=3:max_points=24:nodata=-9999 \
    -txe $boundsx -tye $boundsyr -outsize $nx $ny -of GTiff -ot Float32 -l ${PC_IN} \
    ${PC_IN}.vrt ${IDW_GRID} -co COMPRESS=DEFLATE -co ZLEVEL=7 --config \
    GDAL_NUM_THREADS ALL_CPUS --config GDAL_CACHEMAX 2000
gdalwarp -cutline $CLIP_SHAPEFILE -cl SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83 \
    -crop_to_cutline -tap -multi -tr 1 1 -t_srs epsg:26911 $IDW_GRID \
    ${IDW_GRID::-4}_c.tif -co COMPRESS=DEFLATE -co ZLEVEL=7
gmt grdconvert ${IDW_GRID::-4}_c.tif=gd/1/0/-9999 ${IDW_GRID::-4}_c.nc
```

The comparison of smoothing is shown versus the main DEM (Figure 13) and versus each other (IDW with power=1, smoothing=0 minus IDW with power=1, smoothing=1 and smoothin=2) (Figure {gmt:gdalidw_-p1s0_minus_s12}).

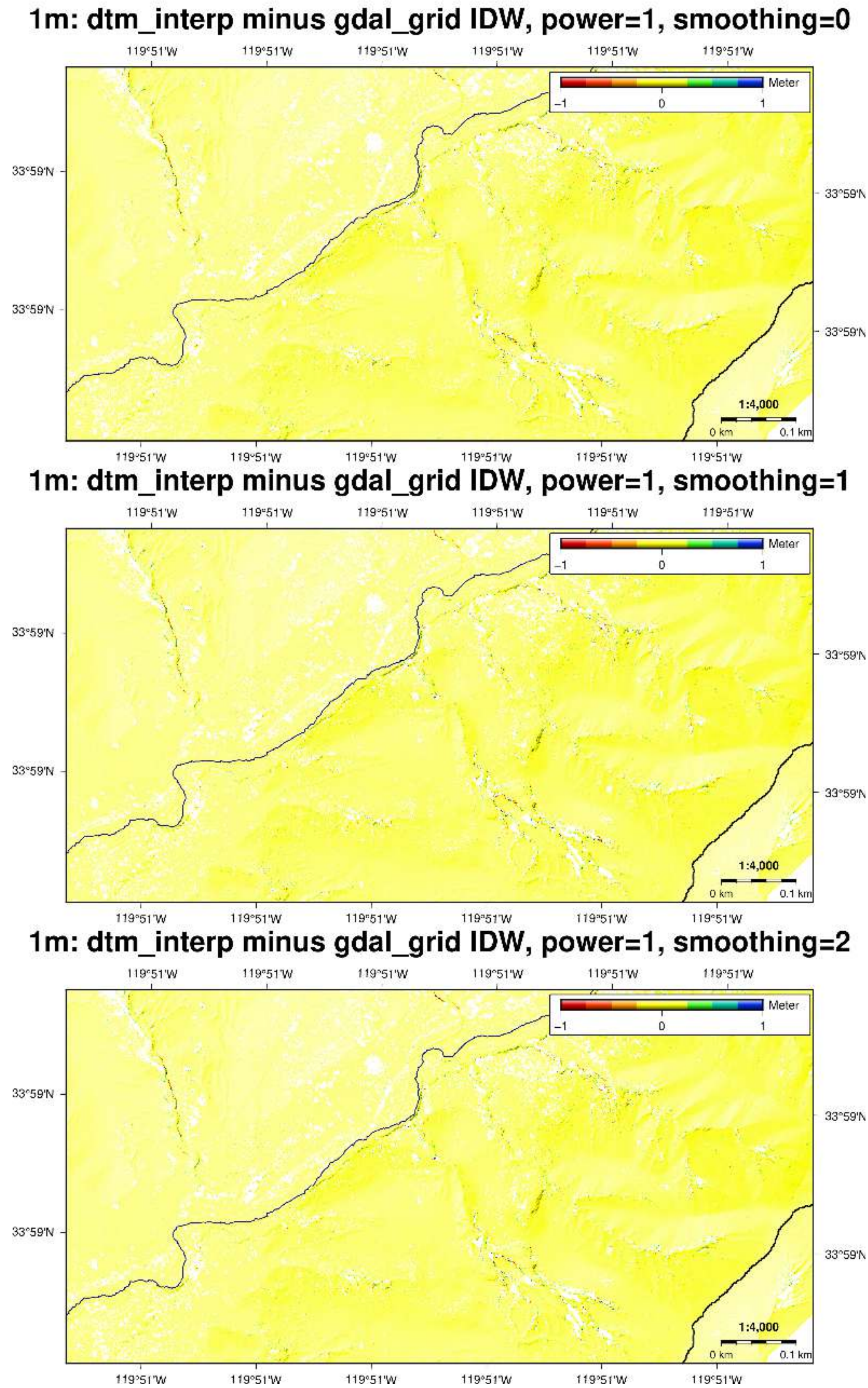
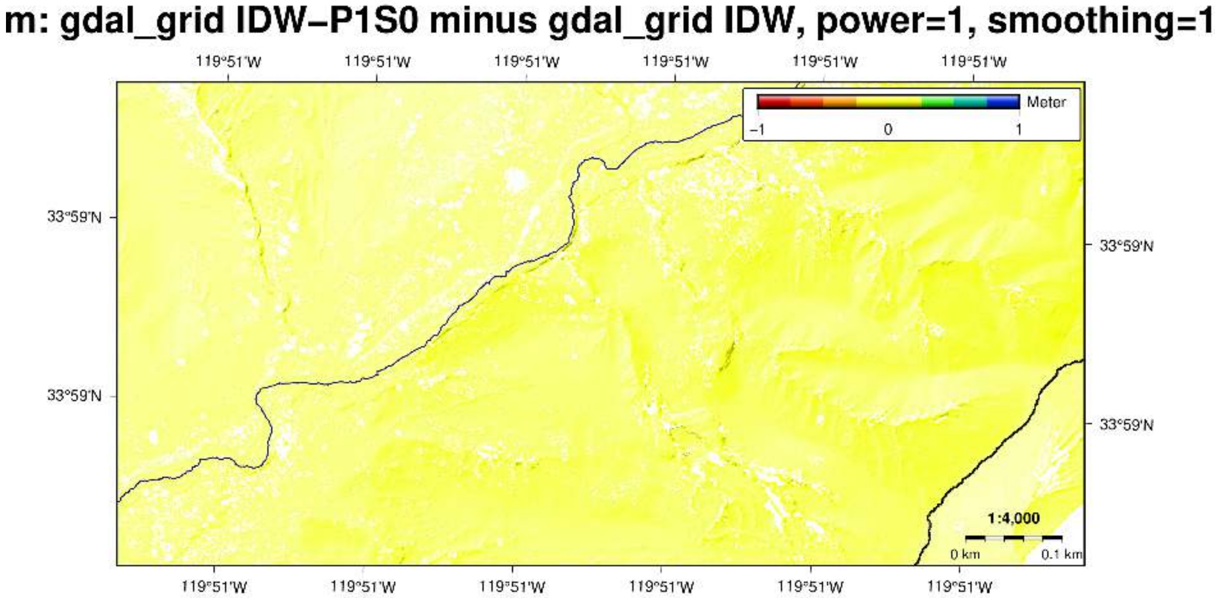


Figure 13: Map view of the LAStools-triangulated minus gdal_grid:idw (power=1, smoothing=0, 1, 2)) interpolation for the Pozo zoom-in area.



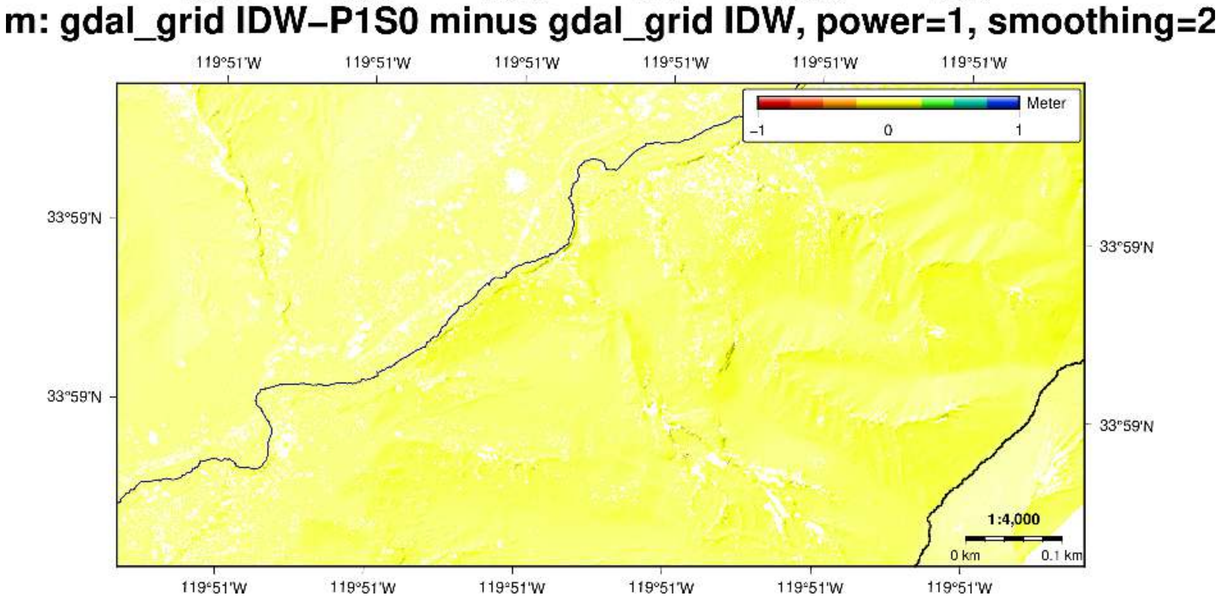


Figure 14: Map view of the DEM interpolated with gdal_grid:idw (power=1, smoothing=0) minus gdal_grid:idw (power=1, smoothing=1 and smoothing=2).

The generation of this maps is shown in GMT5 script gmt5_map_scripts/SCI_Pozo_interpolation_-GMT5_plot_DEM_diff_IDW.sh (see also the Table in the last section).

2.2.3 IDW Interpolation via pdal with writers.gdal

This uses writers.gdal following the Points2Grid approach. We set the radius to resolution * sqrt(2) / 2 (0.707 m) to generate comparable results to the `gdal_grid` approach described above.

Note that this implementation of points2grid uses a 3x3 moving window to fill in voids/NaNs and thus does not have NaN cells as the `gdal_grid` approach described above.

The advantage of the points2grid implementation is that it can read and pipe a large number of points.

Generate a pipeline along these lines:

```
mkdir $DATA_BASEDIR/idw
```

File `SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2_idw_1m_pipeline.json`:

```
{
  "pipeline":[
    "SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2.laz",
    {
      "resolution": 1,
      "radius": 0.707
      "gdaldriver": "GTiff",
      "gdalopts": "COMPRESS=DEFLATE, ZLEVEL=7, GDAL_NUM_THREADS=ALL CPUS",
      "data_type": "float",
      "output_type": "mean, idw, count, stdev",
      "filename":"idw/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2_idw_1m.tif"
    }
  ]
}
```

Run with:

```
pdal pipeline SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2_idw_1m_pipeline.json
```

You will need to clip the file to have the same size as the input file:

```
CLIP_SHAPEFILE=SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83.shp
DEM_GRID=$DATA_BASEDIR/dtm_interp/Pozo_USGS_UTM11_NAD83_g_1mc.tif
# get x,y bounds
export minx=`gmt grdinfo -C $DEM_GRID |cut -f 2`
export maxx=`gmt grdinfo -C $DEM_GRID |cut -f 3`
export nx=`gmt grdinfo -C $DEM_GRID |cut -f 10`
```

```
export miny=`gmt grdinfo -C $DEM_GRID |cut -f 4`
export maxy=`gmt grdinfo -C $DEM_GRID |cut -f 5`
export ny=`gmt grdinfo -C $DEM_GRID |cut -f 11`
export boundste="$minx $miny $maxx $maxy"

#-cutline $CLIP_SHAPEFILE -cl SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83  \
    -crop_to_cutline
gdalwarp -multi -te $boundste -ts $nx $ny -t_srs epsg:26911  \
    idw/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2_idw_1m.tif  \
    idw/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2_idw_1m_c.tif -co  \
    COMPRESS=DEFLATE -co ZLEVEL=7

gdal_translate -b 2 idw/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2_idw_1m_c.tif  \
    idw/SCI_Pozo_100m_buffer_catchment_UTM11N_NAD83_cl2_idw_1m_c2.tif
```

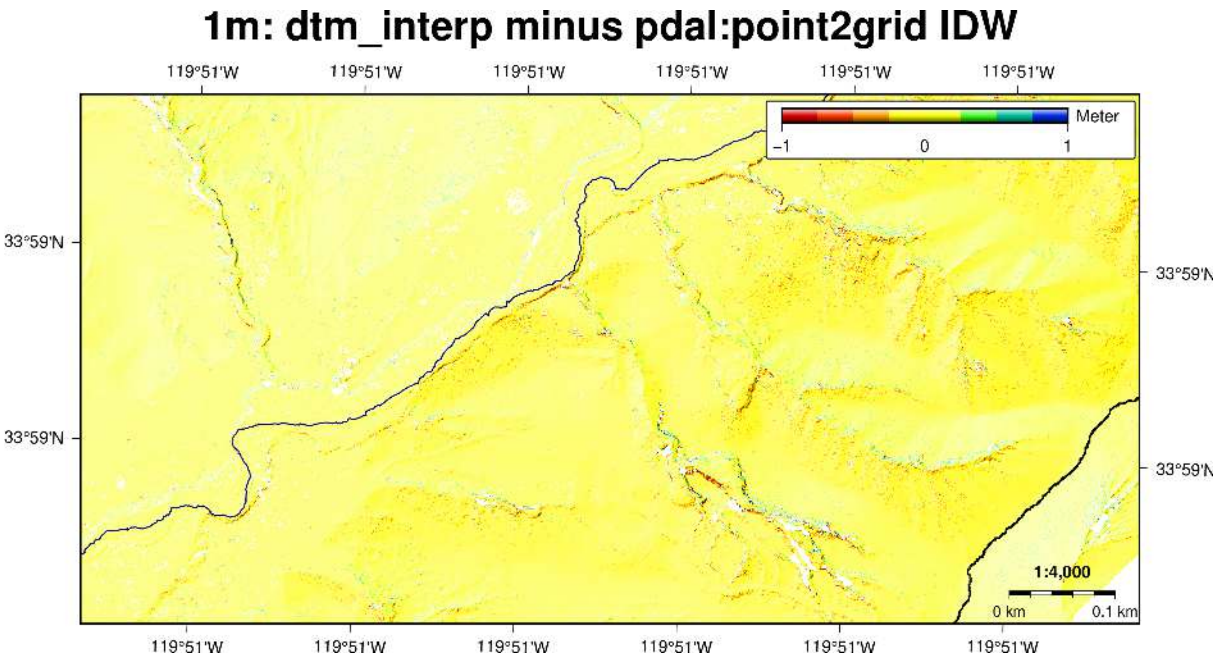


Figure 15: Map view of the LAStools-triangulated minus pdal:Points2Grid interpolation for the Pozo zoom-in area.

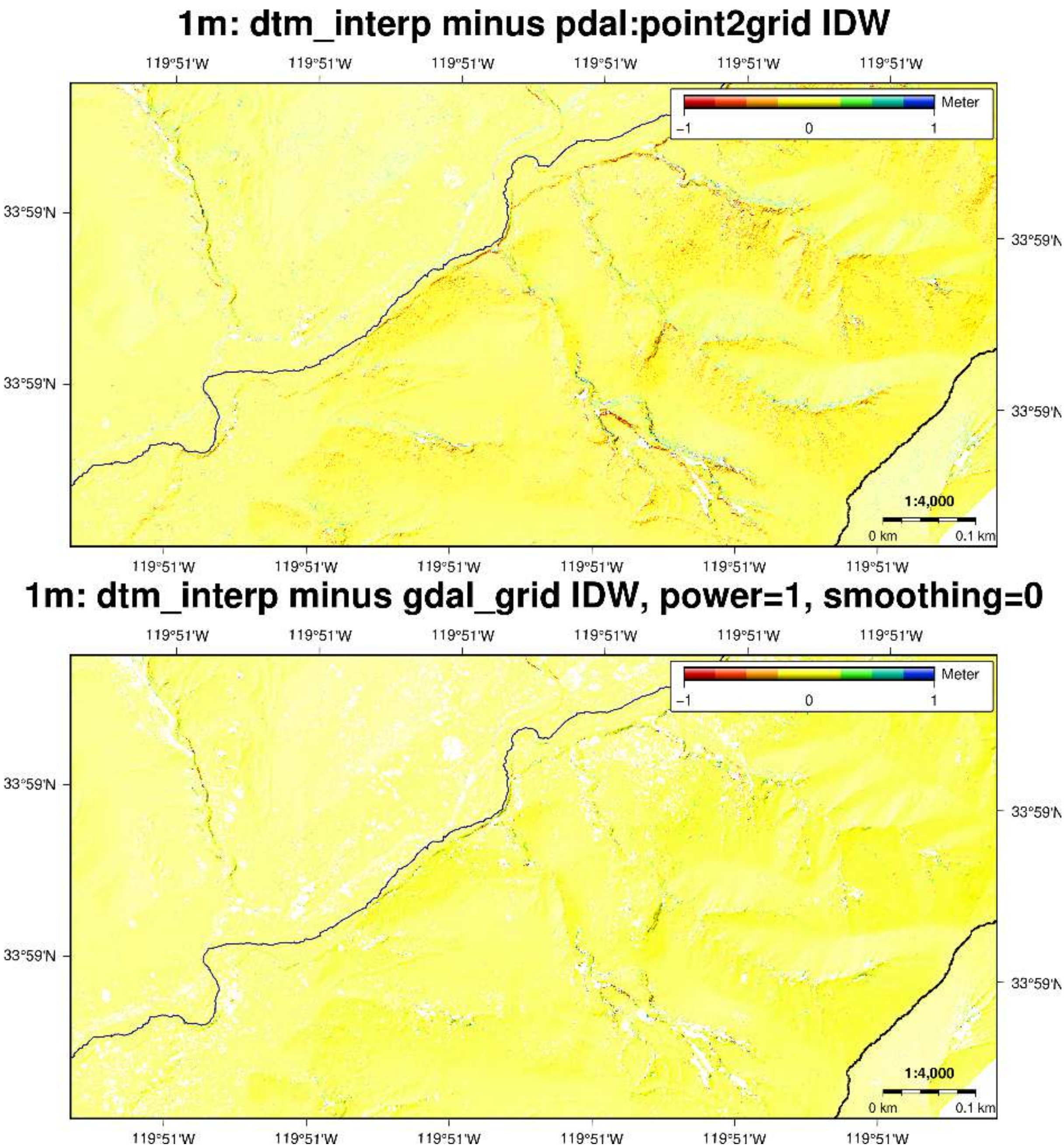


Figure 16: Map view of the LAStools-triangulated minus pdal:Points2Grid and gdal_grid (power=1, smoothing=0) interpolation for the Pozo zoom-in area.

The DEM difference of pdal:Points2Grid of the Pozo catchment and the area of interest is shown in Figure 15 and 16. The comparison between gdal_grid:idwP1S0 and pdalidw is shown in Figure 17.

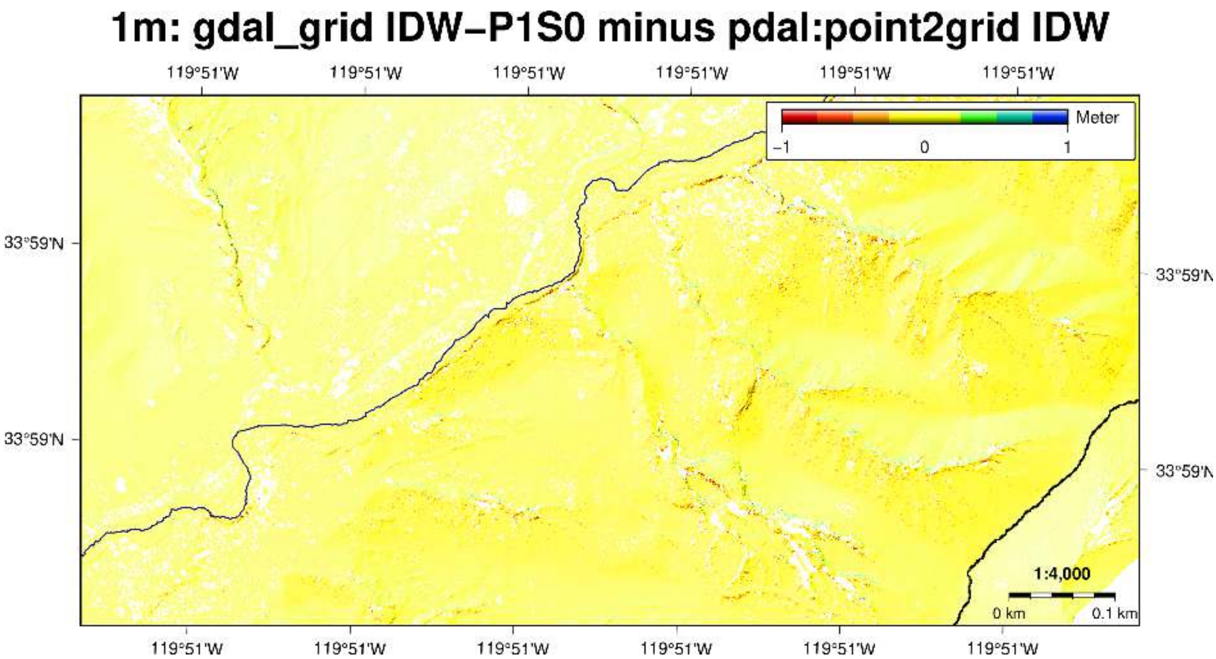


Figure 17: Map view of the gdal_grid:IDW (power=1, smoothing=0) minus pdal:Points2Grid interpolation for the Pozo zoom-in area.

3 Plot with GMT5

Below, we provide a simple GMT Version 5 (GMT5) shell script to plot the DEM difference data. There are several GMT5 scripts for different purposes.

GMT5 Script and Link	Comments	Output Map
SCI_Pozo_interpolation_GMT5_-plot_DEM_overview_zoom.sh	Plot Pozo overview DEM and zoom-in area	Figure 1
SCI_Pozo_interpolation_GMT5_-plot_DEM_diff_blockmean_-blockmedian.sh	GMT:Blockmean and GMT:Blockmedian processing. Simple, but robust interpolation.	Figures 2, 3
SCI_Pozo_interpolation_GMT5_-plot_DEM_diff_surfacetension.sh	Splines with tensions	Figure 6
SCI_Pozo_interpolation_GMT5_-plot_DEM_diff_IDW.sh	IDW-based interpolation with various parameters.	Figures 8, 9, 10, 11, 12, 13, 14, 15

GMT5 Script and Link	Comments	Output Map
SCI_Pozo_interpolation_GMT5_-plot_DEM_diff_nearneighbor.sh		Figure *note done yet*
SCI_Pozo_interpolation_GMT5_-plot_DEM_diff_triangulation.sh	Delauny Triangulation	Figure 5

As an example, the GMT5 script for plotting the DEM and overview is shown here:

```
#!/bin/bash
### GMT V 5 file!
gmt gmtset MAP_FRAME_PEN    1
gmt gmtset MAP_FRAME_WIDTH    0.1
gmt gmtset MAP_FRAME_TYPE     plain
gmt gmtset FONT_TITLE    Helvetica-Bold
gmt gmtset FONT_LABEL    Helvetica-Bold 14p
gmt gmtset PS_PAGE_ORIENTATION    landscape
gmt gmtset PS_MEDIA    A4
gmt gmtset FORMAT_GEO_MAP    D
gmt gmtset MAP_DEGREE_SYMBOL degree
gmt gmtset PROJ_LENGTH_UNIT cm
gmt gmtset MAP_FRAME_AXES WESNZ

# MAP Parameters
#

#data are in /home/bodo/Dropbox/California/SCI/Pozo

#Pozo_USGS_UTM11_NAD83_g_05m.tif
#Pozo_USGS_UTM11_NAD83_g_5m.tif
#Pozo_USGS_UTM11_NAD83_g_10m.tif
#Pozo_USGS_UTM11_NAD83_g_30m.tif
#cd /home/bodo/Dropbox/California/SCI/Pozo/dtm_interp/
#convert to compressed NetCDF format (GMT)
#gdal_translate -co COMPRESS=DEFLATE -of NetCDF Pozo_USGS_UTM11_NAD83_g_05m.tif  \
    Pozo_USGS_UTM11_NAD83_g_05m.nc
#gdal_translate -co COMPRESS=DEFLATE -of NetCDF Pozo_USGS_UTM11_NAD83_g_1m.tif  \
    Pozo_USGS_UTM11_NAD83_g_1m.nc
#gdal_translate -co COMPRESS=DEFLATE -of NetCDF Pozo_USGS_UTM11_NAD83_g_5m.tif  \
    Pozo_USGS_UTM11_NAD83_g_5m.nc
#gdal_translate -co COMPRESS=DEFLATE -of NetCDF Pozo_USGS_UTM11_NAD83_g_10m.tif  \
```

```
    Pozo_USGS_UTM11_NAD83_g_10m.nc
#gdal_translate -co COMPRESS=DEFLATE -of NetCDF Pozo_USGS_UTM11_NAD83_g_30m.tif  \
    Pozo_USGS_UTM11_NAD83_g_30m.nc

POZO_DEM=dtm_interp/Pozo_USGS_UTM11_NAD83_g_1mc.nc
POZO_DEM_HS=${POZO_DEM::-3}_HS.nc
gmt grd2cpt $POZO_DEM -E25 -Cdem2 > dem2_color.cpt
#additional color tables are: -Cdem1, -Cdem3, -Cdem4
if [ ! -e $POZO_DEM_HS ]
then
    echo "generate hillshade $DEM_GRID_HS"
    #more fancy hillshading:
    gmt grdgradient $POZO_DEM -Em315/45+a -Ne0.8 -G$POZO_DEM_HS
fi

#Boundary (polygon) of SCI:  \
    /home/bodo/Dropbox/California/SCI/SCI_boundary_clip_UTM11N_NAD83.shp
#convert to GMT format
#ogr2ogr -f GMT SCI_boundary_clip_UTM11N_NAD83.gmt  \
    /home/bodo/Dropbox/California/SCI/SCI_boundary_clip_UTM11N_NAD83.shp
SCI_BOUNDARY=/raid-cachi/bodo/Dropbox/California/SCI/SCI_boundary_clip_UTM11N_NAD83.gmt

#Pozo catchment
#ogr2ogr -f GMT SCI_Pozo_catchment_UTM11N_NAD83.gmt  \
    /home/bodo/Dropbox/California/SCI/SCI_Pozo_catchment_UTM11N_NAD83.shp
POZO_BOUNDARY=/raid2/bodo/Dropbox/California/SCI/SCI_Pozo_catchment_UTM11N_NAD83.gmt

#Preparing stream network:
#extracted stream from Matlab scripts (Neely et al., 2017) stored in  \
    SCI_1m_noveg_DTM_UTM11_NAD83_shapefiles.zip
#unzip  SCI_1m_noveg_DTM_UTM11_NAD83_shapefiles.zip
#SCI_FAC=shapefiles/SCI_1m_noveg_DTM_UTM11_NAD83_all_MS_proj.shp

### Image-specific definitions
#For an example see: http://gmt.soest.hawaii.edu/doc/5.4.2/gallery/ex28.html#example-28

#width of map in cm:
OVERVIEW_WIDTH=10
OVERVIEW_SCALE=1:22500
OVERVIEW_REGION=$POZO_DEM
#OVERVIEW_REGION=236652.03/237152.03/3764517.98/3765017.98
OVERVIEW_XSTEPS=0.04
OVERVIEW_YSTEPS=0.04
```

```
echo "Creating map for Pozo"
POSTSCRIPT1=figures/Pozo_catchment_topo_overview_map.ps
TITLE="Pozo catchment, Santa Cruz Island, California, 1-m Lidar DEM"
CPT="dem2_color.cpt"
gmt grdimage -Q -R$OVERVIEW_REGION $POZO_DEM -I$POZO_DEM_HS -C$CPT -Jx$OVERVIEW_SCALE  \
    -V -K --COLOR_BACKGROUND=white > $POSTSCRIPT1
# Overlay geographic data and coregister by using correct region and gmt projection  \
    with the same scale
#add shoreline from Lidar data
gmt psxy -Wthin,darkblue -R -J < profile-xy-trace_long_profile.txt -O -K >> $POSTSCRIPT1
gmt psxy -Wthin,black -R$OVERVIEW_REGION -Jx$OVERVIEW_SCALE $SCI_BOUNDARY -O -K >>  \
    $POSTSCRIPT1
gmt psxy -Wthick,black -R -J $POZO_BOUNDARY -O -K >> $POSTSCRIPT1
gmt pscoast -R -Ju11S/$OVERVIEW_SCALE -V -N1 -K -O -Df -Bx1m -By1m  \
    --FONT_ANNOT_PRIMARY=10p --FORMAT_GEO_MAP=ddd:mmF >> $POSTSCRIPT1
gmt psbasemap -R -J -O -K -B+t"$TITLE" --FONT_ANNOT_PRIMARY=9p  \
    -LjRB+c19:23N+f+w1k+l1:22,500+u+o0.2i --FONT_LABEL=10p >> $POSTSCRIPT1
gmt psscale -R -V -J -DjTRC+o1.5c/0.3c/+w6c/0.3c+h -C$CPT -I -F+gwhite+r1p+pthin,black  \
    -Bx100 -By+lMeter --FONT=10p --FONT_ANNOT_PRIMARY=10p -O >> $POSTSCRIPT1
#convert to pdf and PNG
#convert -rotate 90 -quality 100 -density 300 $POSTSCRIPT1 ${POSTSCRIPT1::-3}.pdf
convert -rotate 90 -quality 100 -density 300 -flatten -fuzz 1% -trim +repage  \
    $POSTSCRIPT1 ${POSTSCRIPT1::-3}.png

### Creating second map showing focus area in Pozo
OVERVIEW_WIDTH=10
OVERVIEW_SCALE=1:4500
OVERVIEW_REGION=236000/237000/3764000/3764500
OVERVIEW_XSTEPS=0.04
OVERVIEW_YSTEPS=0.04
echo "Creating zoom-in map for Pozo"
POSTSCRIPT2=figures/Pozo_catchment_topo_zoom_map.ps
TITLE="Zoom in of Pozo, 1-m Lidar DEM"
CPT="dem2_color_zoom.cpt"
gmt grd2cpt $POZO_DEM -R$OVERVIEW_REGION -E25 -Cdem2 > $CPT
gmt grdimage -Q -R$OVERVIEW_REGION $POZO_DEM -I$POZO_DEM_HS -C$CPT -Jx$OVERVIEW_SCALE  \
    -V -K --COLOR_BACKGROUND=white > $POSTSCRIPT2
gmt psxy -Wthin,darkblue -R -J < profile-xy-trace_long_profile.txt -O -K >> $POSTSCRIPT2
gmt psxy -Wthick,black -R -J $POZO_BOUNDARY -O -K >> $POSTSCRIPT2
gmt pscoast -R -Ju11S/$OVERVIEW_SCALE -V -N1 -K -O -Df -Bx0.1m -By0.1m  \
    --FONT_ANNOT_PRIMARY=10p --FORMAT_GEO_MAP=ddd:mmF >> $POSTSCRIPT2
gmt psbasemap -R -J -O -K -B+t"$TITLE" --FONT_ANNOT_PRIMARY=9p  \
    -LjRB+c19:23N+f+w0.1k+l1:4,500+u+o0.2i --FONT_LABEL=10p >> $POSTSCRIPT2
```

```
gmt psscale -R$OVERVIEW_REGION -V -J -DjTRC+o1.5c/0.3c/+w6c/0.3c+h -C$CPT -I  \
    -F+gwhite+r1p+pthin,black -Bx50 -By+lMeter --FONT=10p --FONT_ANNOT_PRIMARY=10p -O  \
    -K >> $POSTSCRIPT2
convert -rotate 90 -quality 100 -density 300 -flatten -fuzz 1% -trim +repage  \
    $POSTSCRIPT2 ${POSTSCRIPT2::-3}.png

convert -quality 100 -density 300 ${POSTSCRIPT1::-3}.png ${POSTSCRIPT2::-3}.png  \
    -splice 0x25 -background "#ffffff" -append  \
    figures/Pozo_catchment_topo_overview_zoom_map.png
```