Part 1 - Reading original Landsat 8 imagery files

1) Create functions to read imagery files and store it into matrices

$$\mathbf{CWD} := \text{“C:\PhD\2015\00 Papers\Paper 1\Fort Myers FL\03 Landsat 8\”}$$

$$B2(im) := \left| \begin{array}{l} Im \leftarrow READFILE(concat(im, \text{“\_B2.asc”}), \text{“delimited”}) \\ submatrix(Im, 5, rows(Im) - 1, 0, cols(Im) - 1) \end{array} \right.$$

$$B3(im) := \left| \begin{array}{l} Im \leftarrow READFILE(concat(im, \text{“\_B3.asc”}), \text{“delimited”}) \\ submatrix(Im, 5, rows(Im) - 1, 0, cols(Im) - 1) \end{array} \right.$$

$$B4(im) := \left| \begin{array}{l} Im \leftarrow READFILE(concat(im, \text{“\_B4.asc”}), \text{“delimited”}) \\ submatrix(Im, 5, rows(Im) - 1, 0, cols(Im) - 1) \end{array} \right.$$

$$B6(im) := \left| \begin{array}{l} Im \leftarrow READFILE(concat(im, \text{“\_B6.asc”}), \text{“delimited”}) \\ submatrix(Im, 5, rows(Im) - 1, 0, cols(Im) - 1) \end{array} \right.$$

the submatrix is to remove the header of ASCII files.

Mathcad's READ_IMAGE sucks. It transforms the 16-bit images into 8-bits. Create another Mathcad sheet to read all the files listed by a bat file that reports is (see spikedefenestrator code) and create a batch that convert automatically all the GEOTIFF into ASC file, using GDAL code. Than, modify this part of the code so the reading will be directly over the ASC, which preserves all the 16-bits information.

DONE!!!

2) Get the UTM extreme coords based on the metadata file

```
Limits(file) :=
    test ← READTEXT(concat(file, "_MTL.txt"))
    for i ∈ 0 .. rows(test) − 1
        for j ∈ 0 .. cols(test) − 1
            if test_{i,j} = "CORNER_LL_PROJECTION_X_PRODUCT"
                Xinf ← test_{i,j+2}
            else
                if test_{i,j} = "CORNER_UR_PROJECTION_X_PRODUCT"
                    Xsup ← test_{i,j+2}
                else
                    if test_{i,j} = "CORNER_LL_PROJECTION_Y_PRODUCT"
                        Yinf ← test_{i,j+2}
                    else
                        if test_{i,j} = "CORNER_UR_PROJECTION_Y_PRODUCT"
                            Ysup ← test_{i,j+2}
                        else
                            if test_{i,j} = "GEOMETRIC_RMSE_MODEL_Y"
                                rmseY ← test_{i,j+2}
                            else
                                if test_{i,j} = "GEOMETRIC_RMSE_MODEL_X"
                                    rmseX ← test_{i,j+2}
                                else
                                    if test_{i,j} = "RADIANCE_MULT_BAND_2"
                                        ML2 ← test_{i,j+2}
                                    else
                                        if test_{i,j} = "RADIANCE_MULT_BAND_3"
                                            ML3 ← test_{i,j+2}
                                        else
                                            if test_{i,j} = "RADIANCE_MULT_BAND_4"
                                                ML4 ← test
```

$$Xsup(file) := Limits(file)_{0,0} \quad Xinf(file) := Limits(file)_{1,0} \quad Ysup(file) := Limits(file)_{0,1} \quad Yinf(file) := Limits(file)_{1,1}$$

$$PixelX(utmX, file) := \frac{utmX - Xinf(file)}{30} \qquad PixelY(utmY, file) := \frac{Ysup(file) - utmY}{30}$$

$$UtmX(pixelX, file) := pixelX \cdot 30 + Xinf(file) \qquad UtmY(pixelY, file) := Ysup(file) - pixelY \cdot 30$$

```
else
    if test_{i,j} = "RADIANCE_ADD_BAND_2"
        AL2 ← test_{i,j+2}
    else
        if test_{i,j} = "RADIANCE_ADD_BAND_3"
            AL3 ← test_{i,j+2}
        else
            if test_{i,j} = "RADIANCE_ADD_BAND_4"
                AL4 ← test_{i,j+2}
            else
                continue
```

$$\begin{bmatrix} Xsup & Ysup & ML2 & AL2 \\ Xinf & Yinf & ML3 & AL3 \\ rmseX & rmseY & ML4 & AL4 \end{bmatrix}$$

Filtering

$$kernel := \frac{1}{9} \cdot \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$$border := (rows(kernel) - 1) \cdot 0.5 = 1$$

```
Flt(Image, WindowSize) :=
    A ← Image
    for i ∈ 0 + border .. rows(A) − 1 − border
        for j ∈ 0 + border .. cols(A) − 1 − border
            K ← kernel
            for m ∈ 0 .. rows(kernel) − 1
                for n ∈ 0 .. cols(kernel) − 1
                    W_{m,n} ← A_{i − border + m, j − border + n}
            B_{i,j} ← Σ_{m=0}^{rows(W)−1} Σ_{n=0}^{rows(W)−1} K_{m,n} · W_{m,n}
    submatrix(B, border, rows(B) − 1, border, cols(B) − 1)
```

Gaussian kernel to filter image.

$$\frac{1}{256} \cdot \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix} \qquad [1]$$

$$\frac{1}{25} \cdot \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \qquad \frac{1}{16} \cdot \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad \frac{1}{9} \cdot \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$$\sigma\text{filt}(\text{im}) := \left\| \begin{array}{l} A \leftarrow \text{im} \\ \text{for } i \in 0 + \text{border} .. \text{rows}(A) - 1 - \text{border} \\ \quad \left\| \begin{array}{l} \text{for } j \in 0 + \text{border} .. \text{cols}(A) - 1 - \text{border} \\ \quad \left\| \begin{array}{l} K \leftarrow \text{kernel} \\ \text{for } m \in 0 .. \text{rows}(\text{kernel}) - 1 \\ \quad \left\| \begin{array}{l} \text{for } n \in 0 .. \text{cols}(\text{kernel}) - 1 \\ \quad \left\| W_{m,n} \leftarrow A_{i-\text{border}+m,\, j-\text{border}+n} \right. \end{array} \right. \\ B_{i,j} \leftarrow \sqrt{\langle 5\% \rangle^2 \cdot \left(\sum_{m=0}^{\text{rows}(W)-1} \sum_{n=0}^{\text{rows}(W)-1} \left(\left(K_{m,n} \right)^2 \cdot \left(W_{m,n} \right)^2 \right) \right)} \end{array} \right. \\ \text{submatrix}(B, \text{border}, \text{rows}(B) - 1, \text{border}, \text{cols}(B) - 1) \end{array} \right.$$

$$\sigma\text{im}(\text{im}) := \left\| \begin{array}{l} A \leftarrow \text{im} \\ \text{for } i \in 0 + \text{border} .. \text{rows}(A) - 1 - \text{border} \\ \quad \left\| \begin{array}{l} \text{for } j \in 0 + \text{border} .. \text{cols}(A) - 1 - \text{border} \\ \quad \left\| B_{i,j} \leftarrow A_{i,j} \cdot 5\% \right. \end{array} \right. \\ \text{submatrix}(B, 1, \text{rows}(B) - 1, 1, \text{cols}(B) - 1) \end{array} \right.$$

Part 2 - Creating a UTM mask and cutting original images

Define the pixel limits in one image. Then, calculate its UTM coordinates and use it to find the pixel coordinates of the other images relative to the mask defined in the first image.

$x1 := 2854 + 127$ $y1 := 1824 + 55$ $x2 := x1 + 255$ $y2 := y1 + 255$ Reference: *LC80160422015076LGN00*

$\text{UtmXinf} := \text{UtmX}(x1, \text{“LC80160422015076LGN00”})$ $\text{UtmYinf} := \text{UtmY}(y2, \text{“LC80160422015076LGN00”})$

$\text{UtmXsup} := \text{UtmX}(x2, \text{“LC80160422015076LGN00”})$ $\text{UtmYsup} := \text{UtmY}(y1, \text{“LC80160422015076LGN00”})$

$\text{UtmYsup} = 2935830$ $\text{UtmXinf} = 378330$

$\text{UtmYinf} = 2928180$ $\text{UtmXsup} = 385980$

$\text{PrintFactor} := \frac{1}{256}$

$\text{PixelXsup}(\text{file}) := \text{PixelX}(\text{UtmXsup}, \text{file})$ $\text{PixelYsup}(\text{file}) := \text{PixelY}(\text{UtmYsup}, \text{file})$

$\text{PixelXinf}(\text{file}) := \text{PixelX}(\text{UtmXinf}, \text{file})$ $\text{PixelYinf}(\text{file}) := \text{PixelY}(\text{UtmYinf}, \text{file})$

$\text{MB2}(\text{file}) := \text{submatrix}(\text{B2}(\text{file}), \text{PixelYsup}(\text{file}) - \text{border}, \text{PixelYinf}(\text{file}) + \text{border}, \text{PixelXinf}(\text{file}) - \text{border}, \text{PixelXsup}(\text{file}) + \text{border})$

$\text{MB3}(\text{file}) := \text{submatrix}(\text{B3}(\text{file}), \text{PixelYsup}(\text{file}) - \text{border}, \text{PixelYinf}(\text{file}) + \text{border}, \text{PixelXinf}(\text{file}) - \text{border}, \text{PixelXsup}(\text{file}) + \text{border})$

M stands for Mask, MB2 is the mask applied to channel Blue B2. The +1 and -1 increases in size at image matrix are due to filtering, where the extra pixels are added just for coherence during filtering process and then removed.

$$MB4(file) := submatrix(B4(file), PixelYsup(file) - border, PixelYinf(file) + border, PixelXinf(file) - border, PixelXsup(file) + border)$$

$$MB6(file) := submatrix(B6(file), PixelYsup(file) - border, PixelYinf(file) + border, PixelXinf(file) - border, PixelXsup(file) + border)$$

$$address := \text{“C:\PhD\2015\00 Papers\Paper 1\Fort Myers FL\04 Masked images\”}$$

$$PrintMask(file) := \left\| \begin{array}{l} B6 \leftarrow MB6(file) \\ WRITEBMP(concat(address, file, \text{“\_B6\_mask.tif”}), scale(B6, 0, 255)) \end{array} \right.$$

Reading imagery files

$$\mathbf{CWD} := \text{“C:\PhD\2015\00 Papers\Paper 1\Fort Myers FL\”}$$

$$Files := \left\| \begin{array}{l} Log0 \leftarrow READTEXT(\text{“arquivos.log”}) \\ \text{for } i \in 0..rows(Log0) - 1 \\ \quad \left\| \begin{array}{l} Log_i \leftarrow substr\left(Log0_i, 0, strlen\left(Log0_i\right) - 8\right) \\ Log \end{array} \right. \end{array} \right. \qquad Files = [\text{“LC80160422015076LGN00”}]$$

Table of Original Imagery Channels, Masked

Blue Channel (B2)	Green Channel (B3)	Red Channel (B4)	NIR Channel (B6)

$$i3B2 := MB2\left(Files_0\right) \qquad i3B3 := MB3\left(Files_0\right) \qquad i3B4 := MB4\left(Files_0\right) \qquad i3B6 := MB6\left(Files_0\right)$$

Table of Imagery Channels, Masked and Filtered, Reading

Blue Channel (B2)	Green Channel (B3)	Red Channel (B4)	NIR Channel (B6)

$$im3B2 := Flt(i3B2, 3) \qquad im3B3 := Flt(i3B3, 3) \qquad im3B4 := Flt(i3B4, 3) \qquad im3B6 := Flt(i3B6, 3)$$

$$mean(\sigma filt(i3B2)) = 152.268 \quad mean(\sigma im(i3B2)) = 456.558$$

$$mean(\sigma filt(i3B3)) = 137.224 \quad mean(\sigma im(i3B3)) = 411.342$$

$$mean(\sigma filt(i3B4)) = 115.086 \quad mean(\sigma im(i3B4)) = 344.736$$

$$mean(\sigma filt(i3B6)) = 101.717 \quad mean(\sigma im(i3B6)) = 303.921$$

$$\frac{mean(\sigma im(i3B2))}{mean(\sigma filt(i3B2))} = 2.998 \qquad \frac{mean(\sigma im(i3B3))}{mean(\sigma filt(i3B3))} = 2.998$$

$$\frac{mean(\sigma im(i3B4))}{mean(\sigma filt(i3B4))} = 2.995 \qquad \frac{mean(\sigma im(i3B6))}{mean(\sigma filt(i3B6))} = 2.988$$

```
Mask(im3B2) :=
    for i ∈ 0 .. rows(im3B4) − 1
        for j ∈ 0 .. cols(im3B4) − 1
            if (im3B4[i,j] − im3B6[i,j]) / (im3B4[i,j] + im3B6[i,j]) > 0
                Im[i,j] ← im3B2[i,j]
            else
                Im[i,j] ← −9999
    Im
```

$$Escala := \frac{1}{257} \qquad Amp := 2$$

```
WRITEBMP(concat(address, "im3B6MaxSD_part.tif"), zoom(MaxSDIm(im3B6) • Escala, Amp, Amp))
WRITEBMP(concat(address, "im3B6_part.tif"), zoom(im3B6 • Escala, Amp, Amp))
WRITEBMP(concat(address, "im3B6_filter_part.tif"), zoom(im3B6_Laplace • Escala, Amp, Amp))
WRITERGB(concat(address, "im3B6_uncert_part.tif"), zoom(ImUncert(im3B6, minT3, maxT3) • Escala, Amp, Amp))
```

```
WRITERGB(concat(address, "im3color_part.bmp"), zoom(augment(im3B4, im3B3, im2B2) • Escala, Amp, Amp)) = 0
```

SoundsTotalALB

Table of Imagery Channels, Masked, Filtered, without Land, Reading,

Blue Channel (B2)	Green Channel (B3)	Red Channel (B4)	NIR Channel (B6)
im3B2W := Mask(im3B2)	im3B3W := Mask(im3B3)	im3B4W := Mask(im3B4)	im3B6W := Mask(im3B6)

```
WRITEBMP(concat(address, "im3B6WaterSep_part.tif"), zoom(im3B6W • Escala, Amp, Amp)) = 0
```

Matching WGS84 UTM 19N to Pixel Coords

dim := rows(im3B2W)

CWD := "C:\PhD\2015\00 Papers\Paper 1\Fort Myers FL\06 Smooth Sheets MLLW\"

SOUNDG := reverse(csort(READTEXT("SmoothSheetsMLLW.txt"), 2))

```
SOUNDGALB :=
    CWD ← "C:\PhD\2015\00 Papers\Paper 1\Fort Myers FL\01 LidarMLLW\"
    aa ← READTEXT(concat(CWD, "LidarMLLW.txt"))
    aa ← submatrix(aa, 1, rows(aa) − 1, 0, cols(aa) − 1)
    aaDepth ← aa^⟨2⟩
    aaE ← aa^⟨0⟩
    aaN ← aa^⟨1⟩
    aa ← augment(aaE, aaN, aaDepth)
```

rmseX := Limits(Files_0)_{2,0} = 5.285

rmseY := Limits(Files_0)_{2,1} = 4.281

this procedure allows average the soundings inside a pixel, since the procedure of swapping the soundings file is ordered to store the deeper values and than those can be substituted by shallow ones that are at the same coordinates. Native Sound Sheets dataset presents depth values as negative ones, it was corrected on Sounds determination.
Horizontal RMSE is defined at MTL file, its components in X,Y is used as a cutting off limit to soundings placed at the border of each pixel.

SoundsTotalALB :=
SOUNDG ← SOUNDGALB
σx ← round(1.645 · rmseX, 0)
σy ← round(1.645 · rmseY, 0)
Ss ← identity(dim) · 0
for i ∈ 0 .. rows(SOUNDG) − 1
  if UtmXinf ≤ SOUNDG_{i,0} ≤ UtmXsup ∧ (UtmYinf ≤ SOUNDG_{i,1} ≤ UtmYsup) ∧ −9999 < SOUNDG_{i,2} < 0
    $Y1 \leftarrow \frac{(SOUNDG_{i,1} - UtmYinf)}{30} - trunc\left(\frac{(SOUNDG_{i,1} - UtmYinf)}{30}\right)$
    $X1 \leftarrow \frac{(SOUNDG_{i,0} - UtmXinf)}{30} - trunc\left(\frac{(SOUNDG_{i,0} - UtmXinf)}{30}\right)$
    if $\frac{30-\sigma y}{30} \geq Y1 \geq \frac{\sigma y}{30} \wedge \frac{30-\sigma x}{30} \geq X1 \geq \frac{\sigma x}{30}$
      $row \leftarrow (rows(im3B2W) - 1) - trunc\left(\frac{(SOUNDG_{i,1} - UtmYinf)}{30}\right)$
      $col \leftarrow trunc\left(\frac{(SOUNDG_{i,0} - UtmXinf)}{30}\right)$
      $Ss_{row,col} \leftarrow stack\left(Ss_{row,col}, -(SOUNDG_{i,2}) + (1.521 \cdot 0)\right)$
    else
      $Ss_{row,col} \leftarrow Ss_{row,col}$
  else
    continue
for i ∈ 0 .. rows(Ss) − 1
  for j ∈ 0 .. cols(Ss) − 1
    if $Ss_{i,j} \neq 0$
      c1 ← c1 + 1
      row ← rows($Ss_{i,j}$)
      if row > 1
        $Ss1_{i,j} \leftarrow mean(submatrix(Ss_{i,j}, 1, row - 1, 0, 0))$
      else
        $Ss1_{i,j} \leftarrow Ss_{i,j}$
    else
      $Ss1_{i,j} \leftarrow -9999$
c1
Ss1

SoundsTotalSS :=
σx ← round(1.645 · rmseX, 0)
σy ← round(1.645 · rmseY, 0)
Ss ← identity(dim) · 0
for i ∈ 0 .. rows(SOUNDG) − 1
  if UtmXinf ≤ SOUNDG_{i,0} ≤ UtmXsup ∧ UtmYinf ≤ SOUNDG_{i,1} ≤ UtmYsup ∧ SOUNDG_{i,2} > 0
    $Y1 \leftarrow \frac{(SOUNDG_{i,1} - UtmYinf)}{30} - trunc\left(\frac{(SOUNDG_{i,1} - UtmYinf)}{30}\right)$
    $X1 \leftarrow \frac{(SOUNDG_{i,0} - UtmXinf)}{30} - trunc\left(\frac{(SOUNDG_{i,0} - UtmXinf)}{30}\right)$
    if $\frac{30-\sigma y}{30} \geq Y1 \geq \frac{\sigma y}{30} \wedge \frac{30-\sigma x}{30} \geq X1 \geq \frac{\sigma x}{30}$
      $row \leftarrow (rows(im3B2W) - 1) - trunc\left(\frac{(SOUNDG_{i,1} - UtmYinf)}{30}\right)$
      $col \leftarrow trunc\left(\frac{(SOUNDG_{i,0} - UtmXinf)}{30}\right)$
      $Ss_{row,col} \leftarrow stack\left(Ss_{row,col}, (SOUNDG_{i,2})\right)$
    else
      $Ss_{row,col} \leftarrow Ss_{row,col}$
  else
    continue
for i ∈ 0 .. rows(Ss) − 1
  for j ∈ 0 .. cols(Ss) − 1
    if $Ss_{i,j} \neq 0$
      c1 ← c1 + 1
      row ← rows($Ss_{i,j}$)
      if row > 1
        $Ss1_{i,j} \leftarrow mean(submatrix(Ss_{i,j}, 1, row - 1, 0, 0))$
      else
        $Ss1_{i,j} \leftarrow Ss_{i,j}$
    else
      $Ss1_{i,j} \leftarrow -9999$
c1
Ss1

SoundsTotalALBsup := submatrix(SoundsTotalALB, 0, round(rows(SoundsTotalALB)/2, 0) − 1, 0, cols(SoundsTotalALB) − 1)

SoundsTotalALBinf := submatrix(SoundsTotalALB, round(rows(SoundsTotalALB)/2, 0), rows(SoundsTotalALB) − 1, 0, cols(SoundsTotalALB) − 1)

$$\text{SoundsTotalSSsup} := \text{submatrix}\left(\text{SoundsTotalSS}, 0, \text{round}\left(\frac{\text{rows}(\text{SoundsTotalSS})}{2}, 0\right) - 1, 0, \text{cols}(\text{SoundsTotalSS}) - 1\right)$$

$$\text{SoundsTotalSSinf} := \text{submatrix}\left(\text{SoundsTotalSS}, \text{round}\left(\frac{\text{rows}(\text{SoundsTotalSS})}{2}, 0\right), \text{rows}(\text{SoundsTotalSS}) - 1, 0, \text{cols}(\text{SoundsTotalSS}) - 1\right)$$

$$\text{AreaPerc} := 25\%$$

ALB@SmoothSheets :=

```
ALBTotal ← SoundsTotalALB
SSTotal ← SoundsTotalSS
for i ∈ 0 .. rows(ALBTotal) − 1
    for j ∈ 0 .. cols(ALBTotal) − 1
        if ALBTotal_{i,j} > 0 ∧ SSTotal_{i,j} > 0
            ALB_{i,j} ← ALBTotal_{i,j}
            c1 ← c1 + 1
        else
            ALB_{i,j} ← −9999
c1
ALB
```

SmoothSheets@ALB :=

```
ALBTotal ← SoundsTotalALB
SSTotal ← SoundsTotalSS
for i ∈ 0 .. rows(SSTotal) − 1
    for j ∈ 0 .. cols(SSTotal) − 1
        if ALBTotal_{i,j} > 0 ∧ SSTotal_{i,j} > 0
            SS_{i,j} ← SSTotal_{i,j}
            c1 ← c1 + 1
        else
            SS_{i,j} ← −9999
c1
SS
```

$$\text{ALBsup} := \text{submatrix}(\text{ALB@SmoothSheets}, 0, \text{round}(\text{rows}(\text{ALB@SmoothSheets}) \cdot \text{AreaPerc}, 0) - 1, 0, \text{cols}(\text{ALB@SmoothSheets}) - 1)$$

$$\text{ALBinf} := \text{submatrix}(\text{ALB@SmoothSheets}, \text{round}(\text{rows}(\text{ALB@SmoothSheets}) \cdot \text{AreaPerc}, 0), \text{rows}(\text{ALB@SmoothSheets}) - 1, 0, \text{cols}(\text{ALB@SmoothSheets}) - 1)$$

$$\text{SSsup} := \text{submatrix}(\text{SmoothSheets@ALB}, 0, \text{round}(\text{rows}(\text{ALB@SmoothSheets}) \cdot \text{AreaPerc}, 0) - 1, 0, \text{cols}(\text{SmoothSheets@ALB}) - 1)$$

$$\text{SSinf} := \text{submatrix}(\text{SmoothSheets@ALB}, \text{round}(\text{rows}(\text{ALB@SmoothSheets}) \cdot \text{AreaPerc}, 0), \text{rows}(\text{SmoothSheets@ALB}) - 1, 0, \text{cols}(\text{SmoothSheets@ALB}) - 1)$$

$$\text{Sounds} := \text{SSsup}$$

```
SizeTotalSS := σx ← round(1.645 • rmseX , 0)
               σy ← round(1.645 • rmseY , 0)
               Ss ← identity(dim) • 0
               for i ∈ 0 .. rows(SOUNDG) − 1
                   if UtmXinf ≤ SOUNDG_{i,0} ≤ UtmXsup ∧ UtmYinf ≤ SOUNDG_{i,1} ≤ UtmYsup ∧ SOUNDG_{i,2} > 0
                       Y1 ← (SOUNDG_{i,1} − UtmYinf) / 30 − trunc((SOUNDG_{i,1} − UtmYinf) / 30)
                       X1 ← (SOUNDG_{i,0} − UtmXinf) / 30 − trunc((SOUNDG_{i,0} − UtmXinf) / 30)
                       if (30 − σy)/30 ≥ Y1 ≥ σy/30 ∧ (30 − σx)/30 ≥ X1 ≥ σx/30
                           row ← (rows(im3B2W) − 1) − trunc((SOUNDG_{i,1} − UtmYinf) / 30)
                           col ← trunc((SOUNDG_{i,0} − UtmXinf) / 30)
                           Ss_{row,col} ← stack(Ss_{row,col}, (SOUNDG_{i,2}))
                       else
                           Ss_{row,col} ← Ss_{row,col}
                   else
                       continue
               for i ∈ 0 .. rows(Ss) − 1
                   for j ∈ 0 .. cols(Ss) − 1
                       if Ss_{i,j} ≠ 0 ∧ SmoothSheets@ALB_{i,j} > 0
                           c1 ← c1 + 1
                           row ← rows(Ss_{i,j})
                           if row > 1
                               Ss1_{i,j} ← row
                           else
                               Ss1_{i,j} ← 1
                       else
                           Ss1_{i,j} ← 0
               c1
               Ss1
```

```
SizeTotalALB := SOUNDG ← SOUNDGALB
                σx ← round(1.645 • rmseX , 0)
                σy ← round(1.645 • rmseY , 0)
                Ss ← identity(dim) • 0
                for i ∈ 0 .. rows(SOUNDG) − 1
                    if UtmXinf ≤ SOUNDG_{i,0} ≤ UtmXsup ∧ UtmYinf ≤ SOUNDG_{i,1} ≤ UtmYsup ∧ −9999 < SOUNDG_{i,2} < 0
                        Y1 ← (SOUNDG_{i,1} − UtmYinf) / 30 − trunc((SOUNDG_{i,1} − UtmYinf) / 30)
                        X1 ← (SOUNDG_{i,0} − UtmXinf) / 30 − trunc((SOUNDG_{i,0} − UtmXinf) / 30)
                        if (30 − σy)/30 ≥ Y1 ≥ σy/30 ∧ (30 − σx)/30 ≥ X1 ≥ σx/30
                            row ← (rows(im3B2W) − 1) − trunc((SOUNDG_{i,1} − UtmYinf) / 30)
                            col ← trunc((SOUNDG_{i,0} − UtmXinf) / 30)
                            Ss_{row,col} ← stack(Ss_{row,col}, (SOUNDG_{i,2}))
                        else
                            Ss_{row,col} ← Ss_{row,col}
                    else
                        continue
                for i ∈ 0 .. rows(Ss) − 1
                    for j ∈ 0 .. cols(Ss) − 1
                        if Ss_{i,j} ≠ 0 ∧ ALB@SmoothSheets_{i,j} > 0
                            c1 ← c1 + 1
                            row ← rows(Ss_{i,j})
                            if row > 1
                                Ss1_{i,j} ← row
                            else
                                Ss1_{i,j} ← 1
                        else
                            Ss1_{i,j} ← 0
                c1
                Ss1
```

Lidar data provided the depth value for each pixel by taking an average value. This will produce an uncertainty, considering each sounding, used to accomplish the average,

```
σSoundsTotalSS :=
    σx ← round(1.645 · rmseX, 0)
    σy ← round(1.645 · rmseY, 0)
    Ss ← identity(dim) · 0
    for i ∈ 0 .. rows(SOUNDG) − 1
        if UtmXinf ≤ SOUNDG_{i,0} ≤ UtmXsup ∧ UtmYinf ≤ SOUNDG_{i,1} ≤ UtmYsup ∧ SOUNDG_{i,2} > 0
            Y1 ← (SOUNDG_{i,1} − UtmYinf)/30 − trunc((SOUNDG_{i,1} − UtmYinf)/30)
            X1 ← (SOUNDG_{i,0} − UtmXinf)/30 − trunc((SOUNDG_{i,0} − UtmXinf)/30)
            if (30 − σy)/30 ≥ Y1 ≥ σy/30 ∧ (30 − σx)/30 ≥ X1 ≥ σx/30
                row ← (rows(im3B2W) − 1) − trunc((SOUNDG_{i,1} − UtmYinf)/30)
                col ← trunc((SOUNDG_{i,0} − UtmXinf)/30)
                Ss_{row,col} ← stack(Ss_{row,col}, (SOUNDG_{i,2}))
            else
                Ss_{row,col} ← Ss_{row,col}
        else
            continue
    for i ∈ 0 .. rows(Ss) − 1
        for j ∈ 0 .. cols(Ss) − 1
            if Ss_{i,j} ≠ 0 ∧ SmoothSheets@ALB_{i,j} > 0
                c1 ← c1 + 1
                row ← rows(Ss_{i,j})
                SmoothSheets ← 2.5
                if row > 1
                    Ss1_{i,j} ← SmoothSheets/√row
                else
                    Ss1_{i,j} ← SmoothSheets
            else
                Ss1_{i,j} ← 0
    c1
    Ss1
```

```
σSoundsTotalALB :=
    SOUNDG ← SOUNDGALB
    σx ← round(1.645 · rmseX, 0)
    σy ← round(1.645 · rmseY, 0)
    Ss ← identity(dim) · 0
    for i ∈ 0 .. rows(SOUNDG) − 1
        if UtmXinf ≤ SOUNDG_{i,0} ≤ UtmXsup ∧ UtmYinf ≤ SOUNDG_{i,1} ≤ UtmYsup ∧ −9999 < SOUNDG_{i,2} < 0
            Y1 ← (SOUNDG_{i,1} − UtmYinf)/30 − trunc((SOUNDG_{i,1} − UtmYinf)/30)
            X1 ← (SOUNDG_{i,0} − UtmXinf)/30 − trunc((SOUNDG_{i,0} − UtmXinf)/30)
            if (30 − σy)/30 ≥ Y1 ≥ σy/30 ∧ (30 − σx)/30 ≥ X1 ≥ σx/30
                row ← (rows(im3B2W) − 1) − trunc((SOUNDG_{i,1} − UtmYinf)/30)
                col ← trunc((SOUNDG_{i,0} − UtmXinf)/30)
                Ss_{row,col} ← stack(Ss_{row,col}, (SOUNDG_{i,2}))
            else
                Ss_{row,col} ← Ss_{row,col}
        else
            continue
    for i ∈ 0 .. rows(Ss) − 1
        for j ∈ 0 .. cols(Ss) − 1
            if Ss_{i,j} ≠ 0 ∧ ALB@SmoothSheets_{i,j} > 0
                c1 ← c1 + 1
                row ← rows(Ss_{i,j})
                ALB ← 0.25
                if row > 1
                    Ss1_{i,j} ← ALB/√row
                else
                    Ss1_{i,j} ← ALB
            else
                Ss1_{i,j} ← 0
    c1
    Ss1
```

σSounds := σSoundsTotalSS

```
Sounds2 := for j ∈ 0 .. rows(Sounds) − 1
             for k ∈ 0 .. cols(Sounds) − 1
               if Sounds[j,k] = 0 ∨ Sounds[j,k] = −9999
                 Sss[j,k] ← −9999
               else
                 Sss[j,k] ← Sounds[j,k]
           Sss
```

takes the Sound Matrix and transforms it into a vector, with the 0 values converted into -9999 according to ASCII format.

```
σSounds2 := for j ∈ 0 .. rows(Sounds) − 1
              for k ∈ 0 .. cols(Sounds) − 1
                if Sounds[j,k] = 0 ∨ Sounds[j,k] = −9999
                  Sss[j,k] ← −9999
                else
                  Sss[j,k] ← σSounds[j,k]
            Sss
```

Recording only available sounding elements on both vectors to perform the adjustment

```
ED0 := 0
```

```
ED1 := 8
```

Extinction depth

```
Im0(imSDB) := k ← 0
              for i ∈ 0 .. rows(Sounds2) − 1
                for j ∈ 0 .. cols(Sounds2) − 1
                  if ED1 > Sounds2[i,j] > ED0 ∧ imSDB[i,j] ≠ −9999
                    Out[k] ← imSDB[i,j]
                    k ← k + 1
                  else
                    continue
              Out
```

This σFiltIm0 represents the uncertainty propagation due to filtering. The variables format show what is the expected names for the variables.

```
σFiltIm0(im#B#W, i#B#) := σ ← σfilt(i#B#)
                          k ← 0
                          for i ∈ 0 .. rows(Sounds2) − 1
                            for j ∈ 0 .. cols(Sounds2) − 1
                              if ED1 > Sounds2[i,j] > ED0 ∧ im#B#W[i,j] ≠ −9999
                                Out[k] ← σ[i,j]
                                k ← k + 1
                              else
                                continue
                          Out
```

```
Soundings0(imSDB) := k ← 0
                     for i ∈ 0 .. rows(Sounds2) − 1
                       for j ∈ 0 .. cols(Sounds2) − 1
                         if ED1 > Sounds2[i,j] > ED0 ∧ imSDB[i,j] ≠ −9999
                           Ss[k] ← Sounds2[i,j]
                           k ← k + 1
                         else
                           continue
                     Ss
```

```
σSoundings0(imSDB) := k ← 0
                      for i ∈ 0 .. rows(Sounds2) − 1
                        for j ∈ 0 .. cols(Sounds2) − 1
                          if ED1 > Sounds2[i,j] > ED0 ∧ imSDB[i,j] ≠ −9999
                            Ss[k] ← σSounds2[i,j]
                            k ← k + 1
                          else
                            continue
                      Ss
```

```
IndexSounding(imSDB, m, n) := S ← Sounds2
                              S2 ← Soundings0(imSDB)
                              m ← m
                              n ← n
                              k ← 0
                              for i ∈ 0 .. rows(Sounds2) − 1
                                for j ∈ 0 .. cols(Sounds2) − 1
                                  if ED1 > S[i,j] > ED0 ∧ imSDB[i,j] ≠ −9999
                                    if i = m ∧ j = n
                                      k2 ← k
                                      out ← 1
                                    k ← k + 1
                                  if out = 1
                                    break
                                if out = 1
                                  break
                              if k2 = 0 ∧ S2[0] ≠ S[m,n]
                                k2 ← −2410
                              else
                                k2 ← k2
                              k2
```

$$\text{Model3}(\text{im1},\text{im2}) := \left\| \begin{array}{l} \text{Im1} \leftarrow \text{Im0}(\text{im1}) \\ \text{Im2} \leftarrow \text{Im0}(\text{im2}) \\ \text{for } i \in 0..\text{rows}(\text{Im1}) - 1 \\ \quad a_i \leftarrow \ln\left(\frac{\text{Im1}_i}{\text{Im2}_i}\right) \\ a \end{array} \right.$$

$$S(\text{im}) := \text{Soundings0}(\text{im})$$

$$\sigma S(\text{im}) := \sigma\text{Soundings0}(\text{im})$$

$bb := 0$

$cc := 0$

Based on Dierssen

$$\text{Model4}(\text{im1},\text{im2}) := \left\| \begin{array}{l} \text{Im1} \leftarrow \text{Im0}(\text{im1}) \\ \text{Im2} \leftarrow \text{Im0}(\text{im2}) \\ \text{for } i \in 0..\text{rows}(\text{Im1}) - 1 \\ \quad a_i \leftarrow \ln\left(\frac{\text{Im1}_i - (bb)}{\text{Im2}_i - (cc)}\right) \\ a \end{array} \right.$$

Based on Stumpf

$$\text{Model3}(\text{im1},\text{im2}) := \left\| \begin{array}{l} \text{Im1} \leftarrow \text{Im0}(\text{im1}) \\ \text{Im2} \leftarrow \text{Im0}(\text{im2}) \\ \text{for } i \in 0..\text{rows}(\text{Im1}) - 1 \\ \quad a_i \leftarrow \frac{\ln(\text{Im1}_i)}{\ln(\text{Im2}_i)} \\ a \end{array} \right.$$

Based on Stumpf

$$\sigma\text{Model3}(\text{im\#B2W},\text{im\#B3W},\text{i\#B2},\text{i\#B3}) := \left\| \begin{array}{l} \sigma 1 \leftarrow \sigma\text{FiltIm0}(\text{im\#B2W},\text{i\#B2}) \\ \sigma 2 \leftarrow \sigma\text{FiltIm0}(\text{im\#B3W},\text{i\#B3}) \\ \text{Im1} \leftarrow \text{Im0}(\text{im\#B2W}) \\ \text{Im2} \leftarrow \text{Im0}(\text{im\#B3W}) \\ \text{for } i \in 0..\text{rows}(\text{Im1}) - 1 \\ \quad a_i \leftarrow \sqrt{\left(\left(\frac{\sigma 1_i}{\text{Im1}_i \cdot \ln(\text{Im2}_i)}\right)^2 + \left(\frac{\sigma 2_i \cdot \ln(\text{Im1}_i)}{\text{Im2}_i \cdot (\ln(\text{Im2}_i))^2}\right)^2\right)} \\ a \end{array} \right.$$

Based on Dierssen

$$\sigma\text{Model4}(\text{im\#B2W},\text{im\#B3W},\text{i\#B2},\text{i\#B3}) := \left\| \begin{array}{l} \sigma 1 \leftarrow \sigma\text{FiltIm0}(\text{im\#B2W},\text{i\#B2}) \\ \sigma 2 \leftarrow \sigma\text{FiltIm0}(\text{im\#B3W},\text{i\#B3}) \\ \text{Im1} \leftarrow \text{Im0}(\text{im\#B2W}) \\ \text{Im2} \leftarrow \text{Im0}(\text{im\#B3W}) \\ \text{for } i \in 0..\text{rows}(\text{Im1}) - 1 \\ \quad a_i \leftarrow \sqrt{\left(\left(\frac{\sigma 1_i}{\text{Im1}_i - bb}\right)^2 + \left(\frac{\sigma 2_i}{\text{Im2}_i - cc}\right)^2\right)} \\ a \end{array} \right.$$

$$\text{trunc}(min(\text{im3B2})) = 8329$$

$$\text{Model}(\text{im1},\text{im2}) := \text{Model3}(\text{im1},\text{im2})$$

$$csort\left(augment\left(\frac{round(Model(im1,im2)\cdot 1000,0)}{1000}, S(im1)\right), 0\right)$$

Least Squares Method (Diersen et al., 2003): A0*X=L0, V0=A0*X-L0

$$A(\text{im1},\text{im2}) := \left\| \begin{array}{l} \text{Log} \leftarrow \text{Model}(\text{im1},\text{im2}) \\ \text{for } i \in 0..\text{rows}(\text{Log}) - 1 \\ \quad A_{i,0} \leftarrow \text{Log}_i \\ \quad A_{i,1} \leftarrow 1 \\ A \end{array} \right.$$

$$L(\text{im1}) := S(\text{im1})$$

$$\sigma A(\text{im\#B\#W},\text{im\&B\&W},\text{i\#B\#},\text{i\&B\&}) := \sigma\text{Model3}(\text{im\#B\#W},\text{im\&B\&W},\text{i\#B\#},\text{i\&B\&})$$

$$\sigma L(\text{im\#B\#W}) := \sigma S(\text{im\#B\#W})$$

$$A0(im1, im2) := A(im1, im2) \qquad \sigma A0(im\#B\#W, im\&B\&W, i\#B\#, i\&B\&) := \sigma A(im\#B\#W, im\&B\&W, i\#B\#, i\&B\&)$$

$$L0(im1, im2) := L(im1) \qquad \sigma L0(im\#B\#W, im\&B\&W, i\#B\#, i\&B\&) := \sigma L(im\#B\#W)$$

$$\sigma(im1, im2) := \left\| \begin{array}{l} \sigma L \leftarrow \sigma L(im1) \\ L \leftarrow L0(im1, im2) \\ \text{for } i \in 0 .. \text{rows}(L) - 1 \\ \quad \left\| \sigma_i \leftarrow \left(\frac{1}{\sigma L_i}\right)^2 \right. \\ \sigma \end{array} \right.$$

$$W(im1, im2) := \left\| \begin{array}{l} \sigma L \leftarrow \sigma L(im1) \\ L \leftarrow L0(im1, im2) \\ \text{for } i \in 0 .. \text{rows}(L) - 1 \\ \quad \left\| \sigma_{i,i} \leftarrow \left(\frac{1}{\sigma L_i}\right)^2 \right. \\ \sigma \end{array} \right.$$

$$Aord2(im1, im2) := \left\| \begin{array}{l} Log \leftarrow Model(im1, im2) \\ \text{for } i \in 0 .. \text{rows}(Log) - 1 \\ \quad \left\| \begin{array}{l} A_{i,0} \leftarrow (Log_i)^2 \\ A_{i,1} \leftarrow Log_i \\ A_{i,2} \leftarrow 1 \end{array} \right. \\ A \end{array} \right.$$

$$m0(im1, im2) := \left\| \begin{array}{l} A \leftarrow A0(im1, im2) \\ L \leftarrow L0(im1, im2) \\ W \leftarrow \sigma(im1, im2) \\ AtWA \leftarrow \begin{bmatrix} \sum_{n=0}^{\text{rows}(A)-1} (A_{n,0})^2 \cdot W_n & \sum_{n=0}^{\text{rows}(A)-1} (A_{n,0}) \cdot (A_{n,1}) \cdot W_n \\ \sum_{n=0}^{\text{rows}(A)-1} (A_{n,0}) \cdot (A_{n,1}) \cdot W_n & \sum_{n=0}^{\text{rows}(A)-1} (A_{n,1})^2 \cdot W_n \end{bmatrix} \\ AtWL \leftarrow \begin{bmatrix} \sum_{n=0}^{\text{rows}(A)-1} (A_{n,0}) \cdot (L_n) \cdot W_n \\ \sum_{n=0}^{\text{rows}(A)-1} (A_{n,1}) \cdot (L_n) \cdot W_n \end{bmatrix} \\ ((AtWA)^{-1} \cdot AtWL)_0 \end{array} \right.$$

$$m1(im1, im2) := \left\| \begin{array}{l} A \leftarrow A0(im1, im2) \\ L \leftarrow L0(im1, im2) \\ W \leftarrow \sigma(im1, im2) \\ AtWA \leftarrow \begin{bmatrix} \sum_{n=0}^{\text{rows}(A)-1} (A_{n,0})^2 \cdot W_n & \sum_{n=0}^{\text{rows}(A)-1} (A_{n,0}) \cdot (A_{n,1}) \cdot W_n \\ \sum_{n=0}^{\text{rows}(A)-1} (A_{n,0}) \cdot (A_{n,1}) \cdot W_n & \sum_{n=0}^{\text{rows}(A)-1} (A_{n,1})^2 \cdot W_n \end{bmatrix} \\ AtWL \leftarrow \begin{bmatrix} \sum_{n=0}^{\text{rows}(A)-1} (A_{n,0}) \cdot (L_n) \cdot W_n \\ \sum_{n=0}^{\text{rows}(A)-1} (A_{n,1}) \cdot (L_n) \cdot W_n \end{bmatrix} \\ ((AtWA)^{-1} \cdot AtWL)_1 \end{array} \right.$$

$$Depth(im1, im2) := \left\| \begin{array}{l} A \leftarrow A0(im1, im2) \\ L \leftarrow L0(im1, im2) \\ W \leftarrow \sigma(im1, im2) \\ N \leftarrow \begin{bmatrix} \sum_{n=0}^{\text{rows}(A)-1} (A_{n,0})^2 \cdot W_n & \sum_{n=0}^{\text{rows}(A)-1} (A_{n,0}) \cdot (A_{n,1}) \cdot W_n \\ \sum_{n=0}^{\text{rows}(A)-1} (A_{n,0}) \cdot (A_{n,1}) \cdot W_n & \sum_{n=0}^{\text{rows}(A)-1} (A_{n,1})^2 \cdot W_n \end{bmatrix} \\ U \leftarrow \begin{bmatrix} \sum_{n=0}^{\text{rows}(A)-1} (A_{n,0}) \cdot (L_n) \cdot W_n \\ \sum_{n=0}^{\text{rows}(A)-1} (A_{n,1}) \cdot (L_n) \cdot W_n \end{bmatrix} \\ A \cdot \begin{bmatrix} N_{1,1} \cdot U_0 - N_{0,1} \cdot U_1 \\ -N_{1,0} \cdot U_0 + N_{0,0} \cdot U_1 \end{bmatrix} \cdot \frac{1}{N_{0,0} \cdot N_{1,1} - N_{1,0} \cdot N_{0,1}} \end{array} \right.$$

$A0(im1, im2) := A(im1, im2)$ $L0(im1, im2) := L(im1)$

$$Aord2(im1, im2) := \left\| \begin{array}{l} Log \leftarrow Model(im1, im2) \\ \text{for } i \in 0 .. rows(Log) - 1 \\ \quad \left\| \begin{array}{l} A_{i,0} \leftarrow (Log_i)^2 \\ A_{i,1} \leftarrow Log_i \\ A_{i,2} \leftarrow 1 \end{array} \right. \\ A \end{array} \right.$$

$$\sigma(im1, im2) := \left\| \begin{array}{l} \sigma L \leftarrow \sigma L(im1) \\ L \leftarrow L0(im1, im2) \\ \text{for } i \in 0 .. rows(L) - 1 \\ \quad \left\| \sigma_i \leftarrow \left(\frac{1}{\sigma L_i}\right)^2 \right. \\ \sigma \end{array} \right.$$

$\sigma A0(im\#B2W, im\#B3W, i\#B2, i\#B3) := \sigma Model3(im\#B2W, im\#B3W, i\#B2, i\#B3)$

$\sigma L0(im\#B2W, im\#B3W, i\#B2, i\#B3) := \sigma Soundings0(im\#B2W)$

$a0 := A0(im3B2W, im3B3W)$ $l0 := L0(im3B2W, im3B3W)$

$bbb := a0 \cdot \left(\left(a0^T \cdot a0\right)^{-1} \cdot a0^T \cdot l0\right) - l0$ $\sqrt{\frac{bbb^T \cdot bbb}{rows(bbb) - 2}} = 0.829$

0.118399 0.071527 $\left(\left(a0^T \cdot a0\right)^{-1} \cdot a0^T \cdot l0\right)_0 = 578.943$ 79.039

$\max\left(a0^{\langle 0 \rangle}\right) = 1.016543$ $min\left(a0^{\langle 0 \rangle}\right) = 1.004508$

$$V0(im1, im2) := \left\| \begin{array}{l} m0 \leftarrow m0(im1, im2) \\ m1 \leftarrow m1(im1, im2) \\ \\ X_0 \leftarrow m0 \\ X_1 \leftarrow m1 \\ Ar \leftarrow A0(im1, im2) \\ Lr \leftarrow L0(im1, im2) \\ V \leftarrow Ar \cdot X - Lr \\ V \end{array} \right.$$

$$VTWV(im1, im2) := \left\| \begin{array}{l} \sigma \leftarrow \sigma(im1, im2) \\ V0 \leftarrow V0(im1, im2) \\ \sum_{i=0}^{rows(V0)-1} V0_i \cdot \sigma_i \cdot V0_i \end{array} \right.$$

$$VTV(im1, im2) := \left\| \begin{array}{l} \sigma \leftarrow \sigma(im1, im2) \\ V0 \leftarrow V0(im1, im2) \\ \sum_{i=0}^{rows(V0)-1} V0_i \cdot V0_i \end{array} \right.$$

$$\left\| \begin{array}{l} A \leftarrow A0(im3B2W, im3B3W) \\ L \leftarrow L0(im3B2W, im3B3W) \\ W \leftarrow \sigma(im3B2W, im3B3W) \\ AtWA \leftarrow \begin{bmatrix} \sum_{n=0}^{rows(A)-1} \left(A_{n,0}\right)^2 \cdot W_n & \sum_{n=0}^{rows(A)-1} \left(A_{n,0}\right) \cdot \left(A_{n,1}\right) \cdot W_n \\ \sum_{n=0}^{rows(A)-1} \left(A_{n,0}\right) \cdot \left(A_{n,1}\right) \cdot W_n & \sum_{n=0}^{rows(A)-1} \left(A_{n,1}\right)^2 \cdot W_n \end{bmatrix} \\ \frac{VTWV(im3B2W, im3B3W)}{rows(V0(im3B2W, im3B3W)) - 2} \cdot AtWA^{-1} \end{array} \right. = \begin{bmatrix} 750.499 & -760.259 \\ -760.259 & 770.152 \end{bmatrix}$$

$$\left\| \begin{array}{l} A \leftarrow A0(im3B2W, im3B3W) \\ L \leftarrow L0(im3B2W, im3B3W) \\ W \leftarrow \sigma(im3B2W, im3B3W) \\ AtA \leftarrow \begin{bmatrix} \sum_{n=0}^{rows(A)-1} \left(A_{n,0}\right)^2 & \sum_{n=0}^{rows(A)-1} \left(A_{n,0}\right) \cdot \left(A_{n,1}\right) \\ \sum_{n=0}^{rows(A)-1} \left(A_{n,0}\right) \cdot \left(A_{n,1}\right) & \sum_{n=0}^{rows(A)-1} \left(A_{n,1}\right)^2 \end{bmatrix} \\ \frac{VTV(im3B2W, im3B3W)}{rows(V0(im3B2W, im3B3W)) - 2} \cdot AtA^{-1} \end{array} \right. = \begin{bmatrix} 751.95 & -761.708 \\ -761.708 & 771.598 \end{bmatrix}$$

$\sigma A0(im3B2W, im3B3W, i3B2, i3B3)_0 = 0.003$ $m0(im3B2W, im3B3W) = 580.159$

$\sigma A0(im3B2W, im3B3W, i3B2, i3B3)_0 \cdot m0(im3B2W, im3B3W) = 1.53$

$$\sqrt{797.251 \cdot A(im3B2W, im3B3W)_{0,0}^{\ 2} + 818.135 \cdot (1)^2 + 2 \left\langle -807.622 \right\rangle \cdot A(im3B2W, im3B3W)_{0,0} \cdot (1)} = 0.092$$

$$\sqrt{797.251 \cdot A(im3B2W, im3B3W)_{0,0}^{\ 2} + 818.135 \cdot (1)^2 + \sigma A0(im3B2W, im3B3W, i3B2, i3B3)_0^{\ 2} \cdot m0(im3B2W, im3B3W)^2 + 2 \left\langle -807.622 \right\rangle \cdot A(im3B2W, im3B3W)_{0,0} \cdot (1)} = 1.53$$

$\frac{0.092}{1.53} = 6.01\%$

$$X(im1, im2) := \begin{bmatrix} m0(im1, im2) \\ m1(im1, im2) \end{bmatrix}$$

$$V(im1, im2) := A(im1, im2) \cdot X(im1, im2) - L(im1)$$

$$X2 := \left(\left(A0(im3B2W, im3B3W)^{T} \cdot A0(im3B2W, im3B3W) \right)^{-1} \cdot A0(im3B2W, im3B3W)^{T} \cdot L0(im3B2W, im3B3W) \right) = \begin{bmatrix} 578.943 \\ -582.276 \end{bmatrix}$$

$$r2X2(im1, im2) := \left\| \begin{array}{l} AX \leftarrow A0(im1, im2) \cdot X2 \\ \mu L \leftarrow mean(L0(im1, im2)) \\ L \leftarrow L0(im1, im2) \\ 1 - \dfrac{\sum_{k=0}^{rows(A0(im1, im2)) - 1} \left((AX_k - L_k)^2 \right)}{\sum_{k=0}^{rows(A(im1, im2)) - 1} \left((\mu L - L_k)^2 \right)} \end{array} \right.$$

$$r2X2(im3B2W, im3B3W) = 0.802077555186$$

$$Depth2(im1, im2) := \left\| \begin{array}{l} A \leftarrow A0(im1, im2) \\ L \leftarrow L0(im1, im2) \\ W \leftarrow \sigma(im1, im2) \\ Eq \leftarrow rows(A) \\ c1 \leftarrow \left(\sum_{n=0}^{Eq-1} W_n \right) \\ c2 \leftarrow \left(\sum_{n=0}^{Eq-1} A_{n,0} \cdot L_n \cdot W_n \right) \\ c3 \leftarrow \left(\sum_{n=0}^{Eq-1} A_{n,0} \cdot W_n \right) \\ c4 \leftarrow \left(\sum_{n=0}^{Eq-1} L_n \cdot W_n \right) \\ c5 \leftarrow \left(\sum_{n=0}^{Eq-1} W_n \cdot (A_{n,0})^2 \right) \\ m0 \leftarrow \dfrac{(c1 \cdot c2 - c3 \cdot c4)}{c1 \cdot c5 - (c3)^2} \\ m1 \leftarrow \dfrac{\langle -c3 \cdot c2 + c5 \cdot c4 \rangle}{c1 \cdot c5 - (c3)^2} \\ \langle A^{\langle 0 \rangle} \rangle \cdot m0 + \langle A^{\langle 1 \rangle} \rangle \cdot m1 \end{array} \right.$$

$$m0(im1, im2) := \left\| \begin{array}{l} A \leftarrow A0(im1, im2) \\ L \leftarrow L0(im1, im2) \\ W \leftarrow \sigma(im1, im2) \\ Eq \leftarrow rows(A) \\ c1 \leftarrow \left(\sum_{n=0}^{Eq-1} W_n \right) \\ c2 \leftarrow \left(\sum_{n=0}^{Eq-1} A_{n,0} \cdot L_n \cdot W_n \right) \\ c3 \leftarrow \left(\sum_{n=0}^{Eq-1} A_{n,0} \cdot W_n \right) \\ c4 \leftarrow \left(\sum_{n=0}^{Eq-1} L_n \cdot W_n \right) \\ c5 \leftarrow \left(\sum_{n=0}^{Eq-1} W_n \cdot (A_{n,0})^2 \right) \\ m0 \leftarrow \dfrac{(c1 \cdot c2 - c3 \cdot c4)}{c1 \cdot c5 - (c3)^2} \end{array} \right.$$

$$m1(im1, im2) := \left\| \begin{array}{l} A \leftarrow A0(im1, im2) \\ L \leftarrow L0(im1, im2) \\ W \leftarrow \sigma(im1, im2) \\ Eq \leftarrow rows(A) \\ c1 \leftarrow \left(\sum_{n=0}^{Eq-1} W_n \right) \\ c2 \leftarrow \left(\sum_{n=0}^{Eq-1} A_{n,0} \cdot L_n \cdot W_n \right) \\ c3 \leftarrow \left(\sum_{n=0}^{Eq-1} A_{n,0} \cdot W_n \right) \\ c4 \leftarrow \left(\sum_{n=0}^{Eq-1} L_n \cdot W_n \right) \\ c5 \leftarrow \left(\sum_{n=0}^{Eq-1} W_n \cdot (A_{n,0})^2 \right) \\ m1 \leftarrow \dfrac{\langle -c3 \cdot c2 + c5 \cdot c4 \rangle}{c1 \cdot c5 - (c3)^2} \end{array} \right.$$

$$\left(A0(im3B2W, im3B3W)^{T} \cdot W(im3B2W, im3B3W) \cdot A0(im3B2W, im3B3W) \right)^{-1} \cdot \left(A0(im3B2W, im3B3W)^{T} \cdot W(im3B2W, im3B3W) \cdot L0(im3B2W, im3B3W) \right) = \begin{bmatrix} 580.159198359936 \\ -583.501815967378 \end{bmatrix}$$

$$EstDepth_{i,j} \neq 2410 \wedge ED1 > SoundsTotalALB_{i,j} > ED0$$

```
DepthFinal(im#B2W, im#B3W) :=
    m0 ← m0(im#B2W, im#B3W)
    m1 ← m1(im#B2W, im#B3W)
    maxLog ← max(A0(im#B2W, im#B3W)^⟨0⟩)
    minLog ← min(A0(im#B2W, im#B3W)^⟨0⟩)
    for i ∈ 0 .. rows(im#B2W) − 1
        for j ∈ 0 .. (cols(im#B2W) − 1)
            try
                t1 ← 1
                t2 ← im#B2W_{i,j} > 0
                t3 ← minLog < ln(im#B2W_{i,j} − bb) / ln(im#B3W_{i,j} − cc) < maxLog
                t4 ← ED1 > SoundsTotalALB_{i,j} > ED0
                if t1 = 1 ∧ t2 = 1 ∧ t3 = 1 ∧ t4 = 1
                    D_{i,j} ← (ln(im#B2W_{i,j} − bb) / ln(im#B3W_{i,j} − cc)) · m0 + m1
                else
                    D_{i,j} ← 2410
            on error
                D_{i,j} ← 2410
    D
```

```
the := 3
```

```
DepthFinalOrd2(im#B2W, im#B3W) :=
    A ← Aord2(im#B2W, im#B3W)
    L ← L0(im#B2W, im#B3W)
    m0 ← ((A^T · A)^{-1} · (A^T · L))_0
    m1 ← ((A^T · A)^{-1} · (A^T · L))_1
    m2 ← ((A^T · A)^{-1} · (A^T · L))_2
    maxLog ← max(A0(im#B2W, im#B3W)^⟨0⟩)
    minLog ← min(A0(im#B2W, im#B3W)^⟨0⟩)
    for i ∈ 0 .. rows(im#B2W) − 1
        for j ∈ 0 .. (cols(im#B2W) − 1)
            try
                if j > 72 ∧ (72 < j < 100 ∧ i > 128) = 0 ∧ im#B2W_{i,j} > 0 ∧ minLog < ln(im#B2W_{i,j} − bb) / ln(im#B3W_{i,j} − cc) < maxLog
                    D_{i,j} ← (ln(im#B2W_{i,j} − bb) / ln(im#B3W_{i,j} − cc))^2 · m0 + (ln(im#B2W_{i,j} − bb) / ln(im#B3W_{i,j} − cc)) · m1 + m2
                else
                    D_{i,j} ← 2410
            on error
                D_{i,j} ← 2410
    D
```

```
:DepthOpt(im#B2W, im#B3W, i#B2, i#B3) :=
    the ← the
    DF ← DepthFinal(im#B2W, im#B3W)
    A ← A0(im#B2W, im#B3W)
    maxLog ← max(A0(im#B2W, im#B3W)^⟨0⟩)
    minLog ← min(A0(im#B2W, im#B3W)^⟨0⟩)
    σA ← σA0(im#B2W, im#B3W, i#B2, i#B3)
    L ← L0(im#B2W, im#B3W)
    σL ← σL0(im#B2W, im#B3W, i#B2, i#B3)
    W ← σ(im#B2W, im#B3W)
    Eq ← rows(A)
    c1 ← (Σ_{n=0}^{Eq−1} W_n)
    c2 ← (Σ_{n=0}^{Eq−1} A_{n,0} · L_n · W_n)
    c3 ← (Σ_{n=0}^{Eq−1} A_{n,0} · W_n)
    c4 ← (Σ_{n=0}^{Eq−1} L_n · W_n)
    c5 ← (Σ_{n=0}^{Eq−1} W_n · (A_{n,0})^2)
    D ← c1 · c5 − (c3)^2
    F1 ← (c1 · c2 − c3 · c4)
    F2 ← ⟨−c3 · c2 + c5 · c4⟩
    σfilt2 ← σfilt(i#B2)
    σfilt3 ← σfilt(i#B3)
    (Σ_{k=0}^{Eq−1} ((c1 · L_k · W_k − c4 · W_k   F1 · (c1 · 2 · W_k · A_{k,0} − 2 · c3 · W_k)) ...)^2)
```

$$R1a \leftarrow \sum_{k=0}^{Eq-1} \left(\left(\left(\frac{c1 \cdot W_k - c4 \cdot W_k}{D} - \frac{F1 \cdot \left(c1 \cdot 2 \cdot W_k \cdot A_{k,0} - 2 \cdot c3 \cdot W_k \right)}{D^2} \right) \cdot \sigma A_k \right) \right)$$

$$R1b \leftarrow \sum_{k=0}^{Eq-1} \left(\left(\left(\left(\frac{c1 \cdot L_k \cdot W_k - c4 \cdot W_k}{D} - \frac{F1 \cdot \left(c1 \cdot 2 \cdot W_k \cdot A_{k,0} - 2 \cdot c3 \cdot W_k \right)}{D^2} \right) \cdot \left(\frac{-c2 \cdot W_k - c3 \cdot L_k \cdot W_k + 2 \cdot c4 \cdot W_k \cdot A_{k,0}}{D} - \frac{F2 \cdot \left(c1 \cdot 2 \cdot W_k \cdot A_{k,0} - 2 \cdot c3 \cdot W_k \right)}{D^2} \right) \right) \cdot \left(\sigma A_k \right)^2 \right) \right)$$

$$R1c \leftarrow \sum_{k=0}^{Eq-1} \left(\left(\left(\frac{-c2 \cdot W_k - c3 \cdot L_k \cdot W_k + 2 \cdot c4 \cdot W_k \cdot A_{k,0}}{D} - \frac{F2 \cdot \left(c1 \cdot 2 \cdot W_k \cdot A_{k,0} - 2 \cdot c3 \cdot W_k \right)}{D^2} \right) \cdot \sigma A_k \right)^2 \right)$$

$$R2a \leftarrow \sum_{k=0}^{Eq-1} \left(\left(\frac{c1 \cdot A_{k,0} \cdot W_k - c3 \cdot W_k}{D} \cdot \sigma L_k \right)^2 \right)$$

$$R2b \leftarrow \sum_{k=0}^{Eq-1} \left(\frac{-c3 \cdot A_{k,0} \cdot W_k + c5 \cdot W_k}{D} \cdot \sigma L_k \cdot \frac{c1 \cdot A_{k,0} \cdot W_k - c3 \cdot W_k}{D} \cdot \sigma L_k \right)$$

$$R2c \leftarrow \sum_{k=0}^{Eq-1} \left(\left(\frac{-c3 \cdot A_{k,0} \cdot W_k + c5 \cdot W_k}{D} \cdot \sigma L_k \right)^2 \right)$$

for $i \in 0 .. rows(im\#B2W) - 1$

  for $j \in 0 .. (cols(im\#B2W) - 1)$

    try

      if $DF_{i,j} \neq 2410$

        $An \leftarrow \frac{\ln(im\#B2W_{i,j} - bb)}{\ln(im\#B3W_{i,j} - cc)}$

        $\sigma 2 \leftarrow \sigma filt2_{i,j}$

        $\sigma 3 \leftarrow \sigma filt3_{i,j}$

        $\sigma An \leftarrow \sqrt{\left(\left(\frac{\sigma 2}{im\#B2W_{i,j} \cdot \ln(im\#B3W_{i,j})} \right)^2 + \left(\frac{\sigma 3 \cdot \ln(im\#B2W_{i,j})}{im\#B3W_{i,j} \cdot \left(\ln(im\#B3W_{i,j}) \right)^2} \right)^2 \right)}$

        $R1 \leftarrow \left(\frac{F1}{D} \cdot \sigma An \right)^2 + An^2 \cdot R1a + 2 \cdot An \cdot R1b + R1c$

        $R2 \leftarrow An^2 \cdot R2a + 2 \cdot An \cdot R2b + R2c$

        $\Sigma_{i,j} \leftarrow \sqrt{R1 + R2}$

      else

        $\Sigma_{i,j} \leftarrow 2410$

    on error

      $\Sigma_{i,j} \leftarrow 2410$

Σ

$t1 := time(the)$

$ErrorDepth \quad \Sigma Depth(im3B2W, im3B3W, i3B2, i3B3)$

$t2 := time(the)$

$t3 := time(the)$

$ErrorDepth2 := \Sigma DepthOpt(im3B2W, im3B3W, i3B2, i3B3)$

$t4 := time(the)$

$$MinTVU := \left| \begin{array}{l} \text{for } i \in 0..rows(ErrorDepth2)-1 \\ \quad \text{for } j \in 0..cols(ErrorDepth2)-1 \\ \quad\quad \text{if } ErrorDepth2_{i,j} \neq 2410 \\ \quad\quad\quad out_{cont} \leftarrow ErrorDepth2_{i,j} \\ \quad\quad\quad cont \leftarrow cont + 1 \\ min(out) \end{array} \right.$$

$$MaxTVU := \left| \begin{array}{l} \text{for } i \in 0..rows(ErrorDepth2)-1 \\ \quad \text{for } j \in 0..cols(ErrorDepth2)-1 \\ \quad\quad \text{if } ErrorDepth2_{i,j} \neq 2410 \\ \quad\quad\quad out_{cont} \leftarrow ErrorDepth2_{i,j} \\ \quad\quad\quad cont \leftarrow cont + 1 \\ \max(out) \end{array} \right.$$

$$MeanTVU := \left| \begin{array}{l} \text{for } i \in 0..rows(ErrorDepth2)-1 \\ \quad \text{for } j \in 0..cols(ErrorDepth2)-1 \\ \quad\quad \text{if } ErrorDepth2_{i,j} \neq 2410 \\ \quad\quad\quad out_{cont} \leftarrow ErrorDepth2_{i,j} \\ \quad\quad\quad cont \leftarrow cont + 1 \\ mean(out) \end{array} \right.$$

Mean absolute difference between Estimated Depths and ALB data

$ED1 = 8 \qquad ED0 = 0$

$$\left| \begin{array}{l} EstDepth \leftarrow DepthFinal(im3B2W, im3B3W) \\ Soundings \leftarrow SoundsTotalALB \\ \text{for } i \in 0..rows(EstDepth)-1 \\ \quad \text{for } j \in 0..cols(EstDepth)-1 \\ \quad\quad \text{if } EstDepth_{i,j} \neq 2410 \wedge ED1 > Soundings_{i,j} > ED0 \\ \quad\quad\quad Res \leftarrow Res + \left(EstDepth_{i,j} - Soundings_{i,j}\right)^2 \\ \quad\quad\quad k \leftarrow k + 1 \\ \left[\left(\sqrt{\frac{Res}{k}}\right)\ k\right] \end{array} \right. = [1.651 \quad 12534]$$

$$MADTotalArea := \left| \begin{array}{l} EstDepth \leftarrow DepthFinal(im3B2W, im3B3W) \\ Soundings \leftarrow SoundsTotalALB \\ \text{for } i \in 0..rows(EstDepth)-1 \\ \quad \text{for } j \in 0..cols(EstDepth)-1 \\ \quad\quad \text{if } EstDepth_{i,j} \neq 2410 \wedge ED1 > Soundings_{i,j} > ED0 \\ \quad\quad\quad Res \leftarrow Res + \left|\left(EstDepth_{i,j} - Soundings_{i,j}\right)\right| \\ \quad\quad\quad k \leftarrow k + 1 \\ \left[\left(\frac{Res}{k}\right)\ k\right] \end{array} \right.$$

$$MADTotalAreaOrd2 := \left| \begin{array}{l} EstDepth \leftarrow DepthFinalOrd2(im3B2W, im3B3W) \\ Soundings \leftarrow SoundsTotalALB \\ \text{for } i \in 0..rows(EstDepth)-1 \\ \quad \text{for } j \in 0..cols(EstDepth)-1 \\ \quad\quad \text{if } EstDepth_{i,j} \neq 2410 \wedge ED1 > Soundings_{i,j} > ED0 \\ \quad\quad\quad Res \leftarrow Res + \left|\left(EstDepth_{i,j} - Soundings_{i,j}\right)\right| \\ \quad\quad\quad k \leftarrow k + 1 \\ \text{if } bb = 0 \wedge cc = 0 \\ \quad \left[\left(\frac{Res}{k}\right)\ k\right] \\ \text{else} \\ \quad \text{“Not Calculated”} \end{array} \right.$$

$AreaPerc = 25\%$

$rows(V0(im3B2W, im3B3W)) = 112$

$$\sqrt{\frac{VTV(im3B2W, im3B3W)}{rows(V0(im3B2W, im3B3W)) - 2}} = 0.83$$

$MADTotalArea = [1.61 \quad 12534.00]$

$1.96 \ MaxTVU = 3.27$

$1.96 \ MeanTVU = 3.05$

$1.96 \ MinTVU = 3.01$

$MADTotalAreaOrd2 = [1.79 \quad 4758]$

$$MADTotalArea2 := \left| \begin{array}{l} EstDepth \leftarrow DepthFinal(im3B2W, im3B3W) \\ Soundings \leftarrow SoundsTotalALB \\ \text{for } i \in 0..rows(EstDepth)-1 \\ \quad \text{for } j \in 0..cols(EstDepth)-1 \\ \quad\quad \text{if } EstDepth_{i,j} \neq 2410 \wedge ED1 > Soundings_{i,j} > ED0 \\ \quad\quad\quad Res_{i,j} \leftarrow \left|\left(EstDepth_{i,j} - Soundings_{i,j}\right)\right| \\ \quad\quad \text{else} \\ \quad\quad\quad Res_{i,j} \leftarrow -9999 \\ Res \end{array} \right.$$

$$\max(m0(im3B2W, im3B3W) \cdot \sigma A0(im3B2W, im3B3W, i3B2, i3B3)) = 1.54$$

```
ControlPoints(AreaPerc) :=
    EstDepth ← DepthFinal(im3B2W, im3B3W)
    ALB ← SoundsTotalALB
    SS ← SoundsTotalSS
    for i ∈ 0 .. rows(ALB) − 1
        for j ∈ 0 .. cols(ALB) − 1
            if (ALB_{i,j} > 0 ∧ SS_{i,j} > 0)
                SS_{i,j} ← SS_{i,j}
    if AreaPerc < 100%
        SS1 ← submatrix(SS, 0, round(rows(SS) · AreaPerc, 0) − 1, 0, cols(SS) − 1)
        for i ∈ round(rows(SS) · AreaPerc, 0) .. rows(SS) − 1
            for j ∈ 0 .. cols(SS) − 1
                SS1_{i,j} ← −9999
        SS ← SS1
    for i ∈ 0 .. rows(EstDepth) − 1
        for j ∈ 0 .. cols(EstDepth) − 1
            if EstDepth_{i,j} ≠ 2410 ∧ (ALB_{i,j} > 0 ∧ SS_{i,j} < 0)
                Res_k ← ALB_{i,j}
                k ← k + 1
    Res
```

```
DEPTHS(AreaPerc) :=
    EstDepth ← DepthFinal(im3B2W, im3B3W)
    ALB ← SoundsTotalALB
    SS ← SoundsTotalSS
    for i ∈ 0 .. rows(ALB) − 1
        for j ∈ 0 .. cols(ALB) − 1
            if (ALB_{i,j} > 0 ∧ SS_{i,j} > 0)
                SS_{i,j} ← SS_{i,j}
            else
                SS_{i,j} ← −9999
    if AreaPerc < 100%
        SS1 ← submatrix(SS, 0, round(rows(SS) · AreaPerc, 0) − 1, 0, cols(SS) − 1)
        for i ∈ round(rows(SS) · AreaPerc, 0) .. rows(SS) − 1
            for j ∈ 0 .. cols(SS) − 1
                SS1_{i,j} ← −9999
        SS ← SS1
    for i ∈ 0 .. rows(EstDepth) − 1
        for j ∈ 0 .. cols(EstDepth) − 1
            if EstDepth_{i,j} ≠ 2410 ∧ (ALB_{i,j} > 0 ∧ SS_{i,j} < 0)
                Res_k ← EstDepth_{i,j}
                k ← k + 1
    Res
```

```
TVUsup(AreaPerc) :=
    EstDepth ← DepthFinal(im3B2W, im3B3W)
    TVU ← ΣDepthOpt(im3B2W, im3B3W, i3B2, i3B3)
    ALB ← SoundsTotalALB
    SS ← SoundsTotalSS
    for i ∈ 0 .. rows(ALB) − 1
        for j ∈ 0 .. cols(ALB) − 1
            if (ALB_{i,j} > 0 ∧ SS_{i,j} > 0)
                SS_{i,j} ← SS_{i,j}
    if AreaPerc < 100%
        SS1 ← submatrix(SS, 0, round(rows(SS) · AreaPerc, 0) − 1, 0, cols(SS) − 1)
        for i ∈ round(rows(SS) · AreaPerc, 0) .. rows(SS) − 1
            for j ∈ 0 .. cols(SS) − 1
                SS1_{i,j} ← −9999
        SS ← SS1
    for i ∈ 0 .. rows(TVU) − 1
        for j ∈ 0 .. cols(TVU) − 1
            if TVU_{i,j} ≠ 2410 ∧ (ALB_{i,j} > 0 ∧ SS_{i,j} < 0)
                Res_k ← EstDepth_{i,j} + 1.96 TVU_{i,j}
                k ← k + 1
    Res
```

```
TVUinf(AreaPerc) :=
    EstDepth ← DepthFinal(im3B2W, im3B3W)
    TVU ← ΣDepthOpt(im3B2W, im3B3W, i3B2, i3B3)
    ALB ← SoundsTotalALB
    SS ← SoundsTotalSS
    for i ∈ 0 .. rows(ALB) − 1
        for j ∈ 0 .. cols(ALB) − 1
            if (ALB_{i,j} > 0 ∧ SS_{i,j} > 0)
                SS_{i,j} ← SS_{i,j}
    if AreaPerc < 100%
        SS1 ← submatrix(SS, 0, round(rows(SS) · AreaPerc, 0) − 1, 0, cols(SS) − 1)
        for i ∈ round(rows(SS) · AreaPerc, 0) .. rows(SS) − 1
            for j ∈ 0 .. cols(SS) − 1
                SS1_{i,j} ← −9999
        SS ← SS1
    for i ∈ 0 .. rows(TVU) − 1
        for j ∈ 0 .. cols(TVU) − 1
            if TVU_{i,j} ≠ 2410 ∧ (ALB_{i,j} > 0 ∧ SS_{i,j} < 0)
                Res_k ← EstDepth_{i,j} − 1.96 TVU_{i,j}
                k ← k + 1
    Res
```

$$\text{ContaAcertos}(\text{AreaPerc}) := \left\| \begin{array}{l} [\text{CP}\ \ D\ \ \Sigma\ \ \sigma] \leftarrow [\text{ControlPoints}(\text{AreaPerc})\ \ \text{DEPTHS}(\text{AreaPerc})\ \ \text{TVUsup}(\text{AreaPerc})\ \ \text{TVUinf}(\text{AreaPerc})] \\ \text{for } i \in 0..\text{rows}(\text{CP}) - 1 \\ \quad \left\| \begin{array}{l} \text{if } \left(\text{CP}_i > \Sigma_i\right) \vee \left(\text{CP}_i < \sigma_i\right) \\ \quad \left\| \begin{array}{l} \text{cont} \leftarrow \text{cont} + 1 \end{array} \right. \end{array} \right. \\ 1 - \dfrac{\text{cont}}{\text{rows}(\text{CP})} \end{array} \right.$$

$$\text{SoundingsTot}(\text{imSDB}) := \left\| \begin{array}{l} k \leftarrow 0 \\ \text{for } i \in 0..\text{rows}(\text{Sounds2}) - 1 \\ \quad \left\| \begin{array}{l} \text{for } j \in 0..\text{cols}(\text{Sounds2}) - 1 \\ \quad \left\| \begin{array}{l} \text{if } \text{EDmax} > \text{Sounds2}_{i,j} > 0 \wedge \text{imSDB}_{i,j} \neq -9999 \\ \quad \left\| \begin{array}{l} \text{Ss}_k \leftarrow \text{Sounds2}_{i,j} \\ k \leftarrow k + 1 \end{array} \right. \\ \text{else} \\ \quad \left\| \begin{array}{l} \text{continue} \end{array} \right. \end{array} \right. \end{array} \right. \\ \text{Ss} \end{array} \right.$$

$$\text{Stumpf}(\text{im1}, \text{im2}) := \left\| \begin{array}{l} k \leftarrow 0 \\ \text{for } i \in 0..\text{rows}(\text{Sounds2}) - 1 \\ \quad \left\| \begin{array}{l} \text{for } j \in 0..\text{cols}(\text{Sounds2}) - 1 \\ \quad \left\| \begin{array}{l} \text{if } \text{EDmax} > \text{Sounds2}_{i,j} > 0 \wedge \text{im1}_{i,j} \neq -9999 \\ \quad \left\| \begin{array}{l} \text{Ss}_k \leftarrow \dfrac{\ln\left(\text{im1}_{i,j}\right)}{\ln\left(\text{im2}_{i,j}\right)} \\ k \leftarrow k + 1 \end{array} \right. \\ \text{else} \\ \quad \left\| \begin{array}{l} \text{continue} \end{array} \right. \end{array} \right. \end{array} \right. \\ \text{Ss} \end{array} \right.$$

$$\text{EDmax} := 20$$

$$\text{SoundingsTot8}(\textit{im3B2W}, \textit{im3B3W}) := \left\| \begin{array}{l} k \leftarrow 0 \\ \text{for } i \in 0..\text{rows}(\text{SoundsTotalALB}) - 1 \\ \quad \left\| \begin{array}{l} \text{for } j \in 0..\text{cols}(\text{SoundsTotalALB}) - 1 \\ \quad \left\| \begin{array}{l} \text{if } 8 > \text{SoundsTotalALB}_{i,j} > 0 \wedge \text{im3B2W}_{i,j} \neq -9999 \\ \quad \left\| \begin{array}{l} \text{Ss}_{k,0} \leftarrow \text{SoundsTotalALB}_{i,j} \\ \text{Ss}_{k,1} \leftarrow \dfrac{\ln\left(\text{im3B2W}_{i,j}\right)}{\ln\left(\text{im3B3W}_{i,j}\right)} \\ k \leftarrow k + 1 \end{array} \right. \\ \text{else} \\ \quad \left\| \begin{array}{l} \text{continue} \end{array} \right. \end{array} \right. \end{array} \right. \\ \text{corr}\left(\text{Ss}^{\langle 0 \rangle}, \text{Ss}^{\langle 1 \rangle}\right)^2 \\ \text{Ss} \end{array} \right.$$

$$\begin{bmatrix} \text{Xx1} & \text{Xx2} & \text{Xx3} & \text{Xx4} \\ \text{Yy1} & \text{Yy2} & \text{Yy3} & \text{Yy4} \end{bmatrix} := \begin{bmatrix} \text{DEPTHS}(\text{AreaPerc}) & \text{DEPTHS}(\text{AreaPerc}) & \text{DEPTHS}(\text{AreaPerc}) & \text{DEPTHS}(\text{AreaPerc}) \\ \text{ControlPoints}(\text{AreaPerc}) & \text{TVUsup}(\text{AreaPerc}) & \text{TVUinf}(\text{AreaPerc}) & \text{DEPTHS}(\text{AreaPerc}) \end{bmatrix}$$

$$f(x, \mu, \sigma) := \frac{1}{\sqrt{2 \cdot \pi \cdot \sigma^2}} \cdot e^{-\frac{1}{2} \cdot \left(\frac{x - \mu}{\sigma}\right)^2}$$

$$\begin{bmatrix} \text{Xx1} & \text{Xx2} & \text{Xx3} & \text{Xx4} \\ \text{Yy1} & \text{Yy2} & \text{Yy3} & \text{Yy4} \end{bmatrix} := \begin{bmatrix} \text{csort}(\text{Xx1}, 0) & \text{csort}(\text{Xx1}, 0) & \text{csort}(\text{Xx1}, 0) & \text{csort}(\text{Xx1}, 0) \\ \text{csort}(\text{augment}(\text{Xx1}, \text{Yy1}), 0)^{\langle 1 \rangle} & \text{csort}(\text{augment}(\text{Xx1}, \text{Yy2}), 0)^{\langle 1 \rangle} & \text{csort}(\text{augment}(\text{Xx1}, \text{Yy3}), 0)^{\langle 1 \rangle} & \text{csort}(\text{augment}(\text{Xx1}, \text{Yy4}), 0)^{\langle 1 \rangle} \end{bmatrix}$$

$$\text{MM} := \text{Stumpf}(\text{im3B2W}, \text{im3B3W}) \quad \text{LL} := \text{SoundingsTot}(\text{im3B2W})$$

$$\text{AreaPerc} = 25\%$$

$$\text{MM} := \text{csort}(\text{augment}(\text{LL}, \text{MM}), 0)^{\langle 1 \rangle} \quad \text{LL} := \text{csort}(\text{LL}, 0) \quad \text{corr}(\text{MM}, \text{LL})^2 = 0.802$$

$$\text{ContaAcertos}(\text{AreaPerc}) = 99.3\%$$

$$\text{ED} := \left\| \begin{array}{l} \text{for } i \in 0..\text{rows}(\text{LL}) - 1 \\ \quad \left\| \begin{array}{l} \text{Out}_i \leftarrow 4 \end{array} \right. \\ \text{Out} \end{array} \right. \qquad \text{ED} := \text{csort}(\text{augment}(\text{LL}, \text{ED}), 0)^{\langle 1 \rangle}$$

$$\text{Predicted} := \text{ContaAcertos}(\text{AreaPerc}) = 99.3\%$$

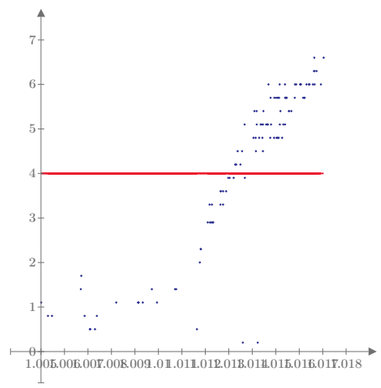
$$\max(\text{SoundingsTot}(\text{im3B2W})) = 6.6$$

$$\max(\text{Sounds2}) = 6.6$$

$$\max(\text{SoundsTotalALB}) = 7.883$$

$$\max(\text{SoundsTotalSS}) = 8.7$$

$$\text{PlotaErros}(\text{AreaPerc}) := \left\| \begin{array}{l} [\text{CP}\ \ D\ \ \Sigma\ \ \sigma] \leftarrow [\text{ControlPoints}(\text{AreaPerc})\ \ \text{DEPTHS}(\text{AreaPerc})\ \ \text{TVUsup}(\text{AreaPerc})\ \ \text{TVUinf}(\text{AreaPerc})] \\ \text{EstDepth} \leftarrow \text{DepthFinal}(\text{im3B2W}, \text{im3B3W}) \\ \text{for } i \in 0..\text{rows}(\text{CP}) - 1 \\ \quad \left\| \begin{array}{l} \text{if } \left(\text{CP}_i > \Sigma_i\right) \vee \left(\text{CP}_i < \sigma_i\right) \\ \quad \left\| \begin{array}{l} \text{Res}_{k,0} \leftarrow \text{CP}_i \\ \text{Res}_{k,1} \leftarrow D_i \\ k \leftarrow k + 1 \end{array} \right. \end{array} \right. \\ \text{csort}(\text{Res}, 1) \end{array} \right.$$



LL

ED

MM

MM

$$\text{LL} = \begin{bmatrix} \vdots \\ 1.4 \\ 1.4 \\ 1.7 \\ 2 \\ 2.3 \\ 2.3 \\ 2.9 \\ 2.9 \\ 2.9 \\ 2.9 \\ 2.9 \\ 3.3 \\ \vdots \end{bmatrix}$$

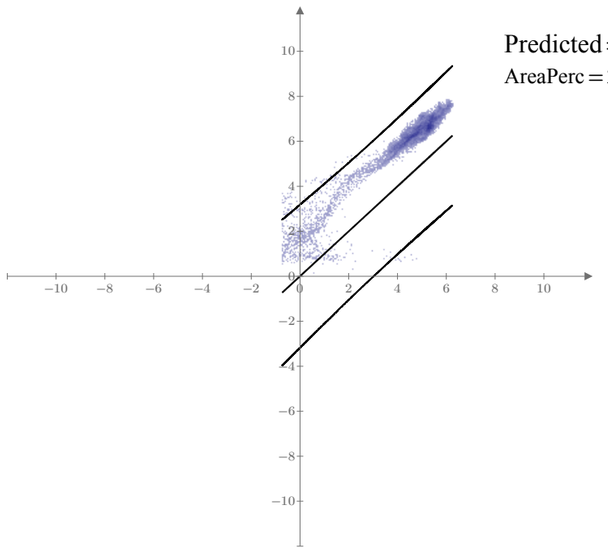
$$\text{Mini} := -10$$

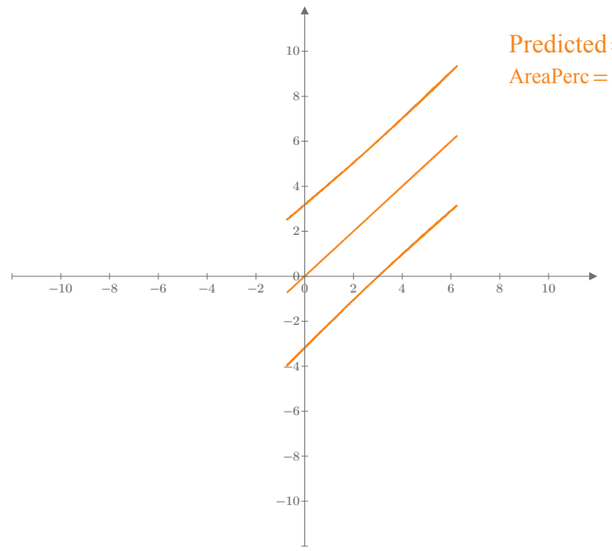
$$\text{Maxi} := 10$$

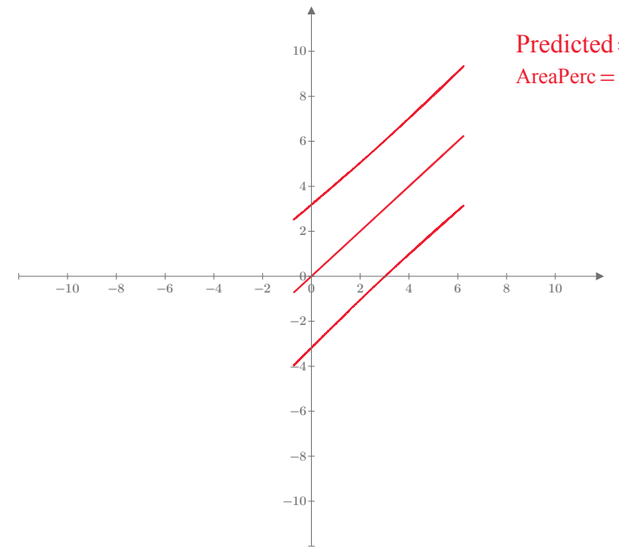
$$\text{Step} := \text{Mini} + 1 \qquad \text{Step} = -9$$

$$min(\blacksquare)$$

42.19%







$\max(L(\text{im3B2W})) = 6.6$

$\min(L(\text{im3B2W})) = 0.2$

$\text{corr}(A(\text{im3B2W}, \text{im3B3W})^{\langle 0 \rangle}, L(\text{im3B2W}))^2 = 0.802$

0.097

$\text{corr}(A(\text{im3B3W}, \text{im3B4W})^{\langle 0 \rangle}, L(\text{im3B2W}))^2 = 0.103$

0.407

$\text{corr}(A(\text{im3B2W}, \text{im3B4W})^{\langle 0 \rangle}, L(\text{im3B2W}))^2 = 0.512$

0.394

$$Yy1 = \begin{bmatrix} \vdots \\ 6.501 \\ 6.085 \\ 6.356 \\ 6.333 \\ 6.056 \\ 6.142 \\ 6.321 \\ 6.243 \\ 5.866 \\ 6.24 \\ 6.349 \\ 6.365 \\ \vdots \end{bmatrix} \quad Yy2 = \begin{bmatrix} \vdots \\ 6.479 \\ 6.484 \\ 6.484 \\ 6.483 \\ 6.482 \\ 6.484 \\ 6.483 \\ 6.488 \\ 6.49 \\ 6.486 \\ 6.492 \\ 6.494 \\ \vdots \end{bmatrix} \quad Yy3 = \begin{bmatrix} \vdots \\ 0.449 \\ 0.444 \\ 0.444 \\ 0.447 \\ 0.449 \\ 0.448 \\ 0.451 \\ 0.447 \\ 0.447 \\ 0.452 \\ 0.45 \\ 0.453 \\ \vdots \end{bmatrix} \quad Yy4 = \begin{bmatrix} \vdots \\ 3.464 \\ 3.464 \\ 3.464 \\ 3.465 \\ 3.465 \\ 3.466 \\ 3.467 \\ 3.467 \\ 3.469 \\ 3.469 \\ 3.471 \\ 3.473 \\ \vdots \end{bmatrix}$$

$$Xx1 = \begin{bmatrix} -0.727 \\ -0.726 \\ -0.724 \\ -0.724 \\ -0.721 \\ -0.719 \\ -0.719 \\ -0.717 \\ -0.717 \\ -0.717 \\ -0.717 \\ -0.713 \\ -0.713 \\ \vdots \end{bmatrix} \quad Xx2 = \begin{bmatrix} -0.727 \\ -0.726 \\ -0.724 \\ -0.724 \\ -0.721 \\ -0.719 \\ -0.719 \\ -0.717 \\ -0.717 \\ -0.717 \\ -0.717 \\ -0.713 \\ -0.713 \\ \vdots \end{bmatrix} \quad Xx3 = \begin{bmatrix} \vdots \\ 4.743 \\ 4.743 \\ 4.743 \\ 4.743 \\ 4.744 \\ 4.744 \\ 4.744 \\ 4.744 \\ 4.744 \\ 4.744 \\ 4.744 \\ 4.744 \\ \vdots \end{bmatrix} \quad Xx4 = \begin{bmatrix} \vdots \\ 4.743 \\ 4.743 \\ 4.743 \\ 4.743 \\ 4.744 \\ 4.744 \\ 4.744 \\ 4.744 \\ 4.744 \\ 4.744 \\ 4.744 \\ 4.744 \\ \vdots \end{bmatrix}$$

$\text{corr}(Yy1, Yy4)^2 = 0.916$

```
Residuals := DF ← DepthFinal(im3B2W, im3B3W)
             ST ← SoundsTotalALB
             for i ∈ 0..rows(ST) − 1
                 for j ∈ 0..cols(ST) − 1
                     if ED1 > ST_{i,j} > ED0 ∧ DF_{i,j} ≠ 2410
                         Res_{i,j} ← |DF_{i,j} − ST_{i,j}|
                     else
                         Res_{i,j} ← 2410
             Res
```

```
MADTotalArea := EstDepth ← DepthFinal(im3B2W, im3B3W)
                Soundings ← SoundsTotalALB
                for i ∈ 0..rows(EstDepth) − 1
                    for j ∈ 0..cols(EstDepth) − 1
                        if EstDepth_{i,j} ≠ 2410 ∧ ED1 > Soundings_{i,j} > ED0
                            Res ← Res + |(EstDepth_{i,j} − Soundings_{i,j})|
                            k ← k + 1
                [(Res / k) k]
```

```
TVUvsAbsDiff := TVU ← ErrorDepth2
                DF ← DepthFinal(im3B2W, im3B3W)
                ST ← SoundsTotalALB
                for i ∈ 0..rows(ST) − 1
                    for j ∈ 0..cols(ST) − 1
                        if ED1 > ST_{i,j} > ED0 ∧ DF_{i,j} ≠ 2410
                            if |DF_{i,j} − ST_{i,j}| ≤ 1.96 • TVU_{i,j}
                                Out_{i,j} ← 1
                            else
                                Out_{i,j} ← 0
                        else
                            Out_{i,j} ← 2410
                Out
```

```
PercTVUvsAbsDiff := In ← TVUvsAbsDiff                          = 99.29
                    for i ∈ 0..rows(In) − 1
                        for j ∈ 0..cols(In) − 1
                            if In_{i,j} ≠ 2410
                                contT ← contT + 1
                                if In_{i,j} = 1
                                    cont1 ← cont1 + 1
                    (cont1 / contT) • 100
```

```
SoundsTotal := EstDepth ← DepthFinal(im3B2W, im3B3W)
               k ← 0
               for i ∈ 0..rows(EstDepth) − 1
                   for j ∈ 0..cols(EstDepth) − 1
                       if EstDepth_{i,j} ≠ 2410 ∧ ED1 > SoundsTotalALB_{i,j} > ED0
                           SS_{i,j} ← SoundsTotalALB_{i,j}
                           k ← k + 1
                       else
                           SS_{i,j} ← 2410
               SS
```

```
Depths3D := DF ← DepthFinal(im3B2W, im3B3W)
            Out_0 ← concat("ncols ", num2str(cols(DF)))
            Out_1 ← concat("nrows ", num2str(rows(DF)))
            Out_2 ← concat("xllcorner ", num2str(UtmXinf))
            Out_3 ← concat("yllcorner ", num2str(UtmYinf))
            Out_4 ← "cellsize 30"
            Out_5 ← "nodata_value 2410"
            for i ∈ 0 .. rows(DF) − 1
                v0 ← num2str(round(DF_{i,0}, 3))
                for j ∈ 1 .. cols(DF) − 1
                    v0 ← concat(v0, " ", num2str(round(DF_{i,j}, 3)))
                Out_{cont+6} ← v0
                cont ← cont + 1
            Out
```

```
Residuals3D := DF ← Residuals
               Out_0 ← concat("ncols ", num2str(cols(DF)))
               Out_1 ← concat("nrows ", num2str(rows(DF)))
               Out_2 ← concat("xllcorner ", num2str(UtmXinf))
               Out_3 ← concat("yllcorner ", num2str(UtmYinf))
               Out_4 ← "cellsize 30"
               Out_5 ← "nodata_value 2410"
               for i ∈ 0 .. rows(DF) − 1
                   v0 ← num2str(round(DF_{i,0}, 3))
                   for j ∈ 1 .. cols(DF) − 1
                       v0 ← concat(v0, " ", num2str(round(DF_{i,j}, 3)))
                   Out_{cont+6} ← v0
                   cont ← cont + 1
               Out
```

```
TVUxAbsDiff := DF ← TVUvsAbsDiff
               Out_0 ← concat("ncols ", num2str(cols(DF)))
               Out_1 ← concat("nrows ", num2str(rows(DF)))
               Out_2 ← concat("xllcorner ", num2str(UtmXinf))
               Out_3 ← concat("yllcorner ", num2str(UtmYinf))
               Out_4 ← "cellsize 30"
               Out_5 ← "nodata_value 2410"
               for i ∈ 0 .. rows(DF) − 1
                   v0 ← num2str(DF_{i,0})
                   for j ∈ 1 .. cols(DF) − 1
                       v0 ← concat(v0, " ", num2str(round(DF_{i,j}, 3)))
                   Out_{cont+6} ← v0
                   cont ← cont + 1
               Out
```

```
Sounds3DTotal := DF ← SoundsTotal
                 Out_0 ← concat("ncols ", num2str(cols(DF)))
                 Out_1 ← concat("nrows ", num2str(rows(DF)))
                 Out_2 ← concat("xllcorner ", num2str(UtmXinf))
                 Out_3 ← concat("yllcorner ", num2str(UtmYinf))
                 Out_4 ← "cellsize 30"
                 Out_5 ← "nodata_value 2410"
                 for i ∈ 0 .. rows(DF) − 1
                     v0 ← num2str(round(DF_{i,0}, 3))
                     for j ∈ 1 .. cols(DF) − 1
                         v0 ← concat(v0, " ", num2str(round(DF_{i,j}, 3)))
                     Out_{cont+6} ← v0
                     cont ← cont + 1
                 Out
```

```
TVU3D := DF ← ErrorDepth2
         Out_0 ← concat("ncols ", num2str(cols(DF)))
         Out_1 ← concat("nrows ", num2str(rows(DF)))
         Out_2 ← concat("xllcorner ", num2str(UtmXinf))
         Out_3 ← concat("yllcorner ", num2str(UtmYinf))
         Out_4 ← "cellsize 30"
         Out_5 ← "nodata_value 2410"
         for i ∈ 0 .. rows(DF) − 1
             v0 ← num2str(round(DF_{i,0}, 3))
             for j ∈ 1 .. cols(DF) − 1
                 v0 ← concat(v0, " ", num2str(round(DF_{i,j}, 3)))
             Out_{cont+6} ← v0
             cont ← cont + 1
         Out
```

CWD := “C:\PhD\2015\00 Papers\Paper 1\Fort Myers FL\Outputs\ASCII new\”

CompArquivo := “LinStumpf5x5AvgALB100.txt”

PercTVUvsAbsDiff = 99.29

a ← concat(“Depths”, CompArquivo)
WRITETEXT(a, Depths3D) =
[“ncols 256”
“nrows 256”
“xllcorner 378330”
⋮]

a ← concat(“AbsDiff”, CompArquivo)
WRITETEXT(a, Residuals3D) =
[“ncols 256”
“nrows 256”
“xllcorner 378330”
⋮]

a ← concat(“TVU”, CompArquivo)
if bb = 0 ∧ cc = 0
  WRITETEXT(a, TVU3D)
else
  0 =
[“ncols 256”
“nrows 256”
“xllcorner 378330”
⋮]

a ← “TotalSoundsALB.txt”
WRITETEXT(a, Sounds3DTotal) =
[“ncols 256”
“nrows 256”
“xllcorner 378330”
⋮]

a ← concat(“TVUvsAbsDiff”, CompArquivo)
if bb = 0 ∧ cc = 0
  WRITETEXT(a, TVUxAbsDiff)
else
  0 =
[“ncols 256”
“nrows 256”
“xllcorner 378330”
⋮]

WRITEBMP(concat(address, “SoundsTotalALB.tif”), zoom(SoundsTotalALB • Escala, Amp, Amp)) = 0

WRITEBMP(concat(address, “SoundsTotalALB.tif”), zoom(SoundsTotal • Escala, Amp, Amp)) = 0

WRITEBMP(concat(address, “SoundsTotalSS.tif”), zoom(SoundsTotalSS • Escala, Amp, Amp)) = 0

WRITEBMP(concat(address, “SoundsSS.tif”), zoom(SSsup • Escala, Amp, Amp)) = 0