Part 1 - Reading original Landsat 8 imagery files

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

CWD := "C:\PhD\2015\00 Papers\Paper 3 - Simeonof Island\00 - Landsat\"

B2(im) := Im ← READFILE(concat(im, "_B2.asc"), "delimited")
submatrix(Im, 5, rows(Im) − 1, 0, cols(Im) − 1)

B3(im) := Im ← READFILE(concat(im, "_B3.asc"), "delimited")
submatrix(Im, 5, rows(Im) − 1, 0, cols(Im) − 1)

B4(im) := Im ← READFILE(concat(im, "_B4.asc"), "delimited")
submatrix(Im, 5, rows(Im) − 1, 0, cols(Im) − 1)

B6(im) := Im ← READFILE(concat(im, "_B6.asc"), "delimited")
submatrix(Im, 5, rows(Im) − 1, 0, cols(Im) − 1)

the submatrix is to remove the header of ASCII files.

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] Xinf(file) := Limits(file)[1,0] Ysup(file) := Limits(file)[0,1] 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 · 30 + Xinf(file) UtmY(pixelY, file) := Ysup(file) − pixelY · 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

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

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

Gaussian kernel to filter image.

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

```
Flt(Image) :=
    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} ← round( Σ_{m=0}^{rows(W)−1} Σ_{n=0}^{rows(W)−1} K_{m,n} · W_{m,n} , 0 )
    submatrix(B, border, rows(B) − 1, border, cols(B) − 1)
```

$$\frac{1}{25} \cdot \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

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

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

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

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.

$x1 := 2887$ $y1 := 2593$ $x2 := x1 + 177$ $y2 := y1 + 177$ Reference: *LC80720222016183LGN00*

$UtmXinf := UtmX(x1, \text{“LC80720222016183LGN00”})$ $UtmYinf := UtmY(y2, \text{“LC80720222016183LGN00”})$

$UtmXsup := UtmX(x2, \text{“LC80720222016183LGN00”})$ $UtmYsup := UtmY(y1, \text{“LC80720222016183LGN00”})$

$UtmYsup = 6083010$ $UtmXinf = 476010$

$UtmYinf = 6077700$ $UtmXsup = 481320$

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

$PixelXsup(file) := PixelX(UtmXsup, file)$ $PixelYsup(file) := PixelY(UtmYsup, file)$

$PixelXinf(file) := PixelX(UtmXinf, file)$ $PixelYinf(file) := PixelY(UtmYinf, file)$

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

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

M stands for Mask, MB2 is the mask applied to channel Blue B2. The +1 and -1 increases in

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

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

$$address := \text{“C:\PhD\2015\00 Papers\Paper 3 - Simeonof Island\04 - Masked images\”}$$

$$PrintMask(file) := \left\| \begin{array}{l} B6 \leftarrow MB6(file) \\ WRITEBMP(concat(address, file, \text{“_B6_mask.tif”}), scale(B6, 0, 255)) \end{array} \right.$$

channel Blue B2. The +1 and -1 increases in size at image matrix are due to filtering, where the extra pixel are added just for coherence during filtering process and then removed.

A Comparative Study of Removal Noise from Remote Sensing Image, 2010

Reading imagery files

$$\mathbf{CWD} := \text{“C:\PhD\2015\00 Papers\Paper 3 - Simeonof Island\”}$$

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

$$Files = \begin{bmatrix} \text{“LC08_L1TP_072022_20150426_20170228_01_T1”} \\ \text{“LC08_L1TP_072022_20170602_20170615_01_T1”} \\ \text{“LC80720222016183LGN00”} \end{bmatrix}$$

Table of Original Imagery Channels, Masked

Blue Channel (B2)	Green Channel (B3)	Red Channel (B4)	NIR Channel (B6)
$i3B2 := MB2(Files_1)$	$i3B3 := MB3(Files_1)$	$i3B4 := MB4(Files_1)$	$i3B6 := MB6(Files_1)$

Table of Imagery Channels, Masked and Filtered, Reading

Blue Channel (B2)	Green Channel (B3)	Red Channel (B4)	NIR Channel (B6)
$im3B2 := Flt(i3B2)$	$im3B3 := Flt(i3B3)$	$im3B4 := Flt(i3B4)$	$im3B6 := Flt(i3B6)$

$$\text{Mask}(\text{im3B2}) := \left\| \begin{array}{l} \text{for } i \in 0..\text{rows}(\text{im3B4}) - 1 \\ \quad \text{for } j \in 0..\text{cols}(\text{im3B4}) - 1 \\ \quad\quad \text{if } \dfrac{\text{im3B4}_{i,j} - \text{im3B6}_{i,j}}{\text{im3B4}_{i,j} + \text{im3B6}_{i,j}} > 0 \\ \quad\quad\quad \text{Im}_{i,j} \leftarrow \text{im3B2}_{i,j} \\ \quad\quad \text{else} \\ \quad\quad\quad \text{Im}_{i,j} \leftarrow -9999 \\ \text{Im} \end{array} \right\| \qquad \text{Mask2}(\text{im3B2}) := \left\| \begin{array}{l} \text{for } i \in 0..\text{rows}(\text{im3B4}) - 1 \\ \quad \text{for } j \in 0..\text{cols}(\text{im3B4}) - 1 \\ \quad\quad \text{if } \dfrac{\text{im3B4}_{i,j} - \text{im3B6}_{i,j}}{\text{im3B4}_{i,j} + \text{im3B6}_{i,j}} \leq 0 \\ \quad\quad\quad \text{Im}_{i,j} \leftarrow \text{im3B2}_{i,j} \\ \quad\quad \text{else} \\ \quad\quad\quad \text{Im}_{i,j} \leftarrow -9999 \\ \text{Im} \end{array} \right\|$$

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

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, im3B2) • 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

Table of Imagery Channels, Masked, Filtered, just Land, Reading,

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

im3B2L := Mask2(im3B2) im3B3L := Mask2(im3B3) im3B4L := Mask2(im3B4) im3B6L := Mask2(im3B6)

WRITEBMP(concat(address, "im3B6LandSep_part.tif"), zoom(im3B6L • Escala, Amp, Amp)) = 0

$$\text{im3B4} = \begin{bmatrix} 6032 & 6030 & 6026 & 6021 & 6018 & 6021 & 6027 & 6029 & 6027 & 6026 & 6025 & 6028 \\ 6032 & 6029 & 6024 & 6024 & 6023 & 6024 & 6028 & 6034 & 6034 & 6032 & 6025 & 6024 \\ 6027 & 6024 & 6023 & 6026 & 6024 & 6023 & 6023 & 6028 & 6033 & 6036 & 6033 & 6030 \\ 6020 & 6018 & 6021 & 6031 & 6032 & 6029 & 6027 & 6031 & 6037 & 6038 & 6037 & 6036 \\ 6017 & 6015 & 6021 & 6026 & 6025 & 6021 & 6026 & 6031 & 6038 & 6033 & 6034 & 6034 \\ 6016 & 6016 & 6025 & 6030 & 6032 & 6030 & 6034 & 6035 & 6039 & 6032 & 6030 & 6025 \\ 6021 & 6023 & 6029 & 6029 & 6029 & 6030 & 6035 & 6035 & 6036 & 6031 & 6030 & 6025 \\ 6027 & 6030 & 6031 & 6031 & 6032 & 6036 & 6036 & 6035 & 6038 & 6039 & 6034 & 6027 \\ 6032 & 6034 & 6032 & 6031 & 6027 & 6027 & 6026 & 6028 & 6032 & 6035 & 6032 & 6031 \\ 6025 & 6025 & 6025 & 6029 & 6031 & 6032 & 6031 & 6033 & 6039 & 6041 & 6036 & 6035 \\ 6020 & 6021 & 6021 & 6024 & 6027 & 6031 & 6034 & 6038 & 6040 & 6038 & 6037 & 6040 \\ 6017 & 6016 & 6016 & 6019 & 6023 & 6032 & 6039 & 6044 & 6044 & 6042 & 6042 & 6044 \\ & & & & & & & & & & & \ddots \end{bmatrix}$$

Mapeando uma função linear que mostre a relação entre os valores observados nos canais B3 e B4

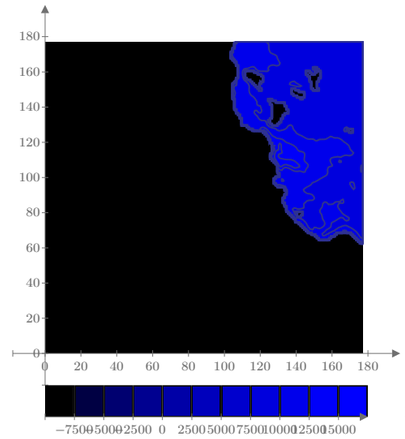
$m := 0 \qquad n := 0$

$$Vector(im3B2L) := \left| \begin{array}{l} im \leftarrow im3B2L \\ \text{for } i \in 0+m \,..\, (rows(im)-1)\cdot 1+m+0 \\ \quad \text{for } j \in 0+n \,..\, (cols(im)-1)\cdot 1+n+0 \\ \quad\quad \text{if } im_{i,j} > 0 \\ \quad\quad\quad Out_k \leftarrow (im_{i,j}) \\ \quad\quad\quad k \leftarrow k+1 \\ Out \end{array} \right.$$

$r2B2B3 := corr(Vector(im3B3L), Vector(im3B4L))^2 = 0.962566$ 0.979729

$r2B3B4 := corr(Vector(im3B3L), Vector(im3B4L))^2 = 0.962566$

$r2B2B4 := corr(Vector(im3B2L), Vector(im3B4L))^2 = 0.942324$



$rotate(im3B2L, 270)$

$VecB := Vector(im3B2L) \qquad VecG := Vector(im3B3L)$

$VecR := Vector(im3B4L)$

$$AGB := \left| \begin{array}{l} im \leftarrow VecG \\ \text{for } i \in 0 \,..\, (rows(im)-1) \\ \quad Out_{i,0} \leftarrow im_i \\ \quad Out_{i,1} \leftarrow 1 \\ Out \end{array} \right. \qquad LGB := VecB$$

$$XGB := (AGB^T \cdot AGB)^{-1} \cdot (AGB^T \cdot LGB) = \begin{matrix} 163.962 \\ 109.717 \end{matrix}$$

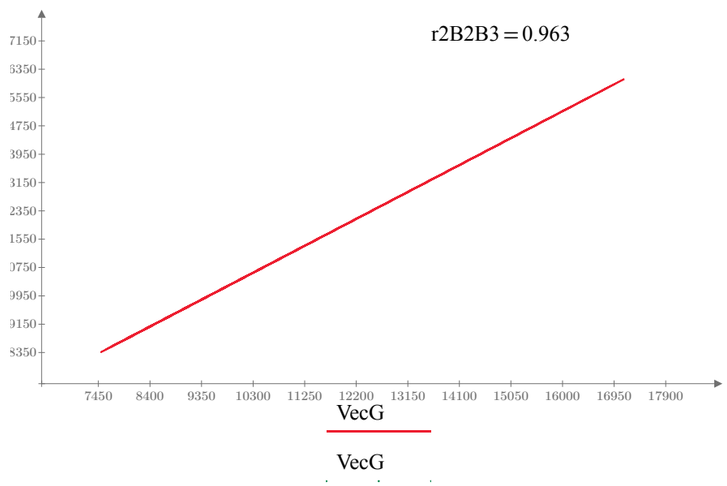
$$VGB := AGB \cdot XGB - LGB \qquad \sqrt{\frac{(VGB^T \cdot VGB)}{rows(VGB)-1}} = 127.093 \qquad \frac{\sum_{i=0}^{rows(VGB)-1} |VGB_i|}{rows(VGB)} = 81.594$$

$min(VecB) = 8456 \qquad max(VecB) = 16445$

$min(VecG) = 7491 \qquad max(VecG) = 17137$

$$Vnorm := \left| \begin{array}{l} \text{for } i \in 0 \,..\, rows(VGB)-1 \\ \quad Out_i \leftarrow |VGB_i| \\ Out \end{array} \right.$$

$r2B2B3 = 0.963$



$VecG_0 \cdot XGB_0 + XGB_1 = 10333.176 \quad VecB_0 = 10502$

$VecG_{150} \cdot XGB_0 + XGB_1 = 10507.603 \quad VecB_{150} = 10361$

A diferença entre o valor estimado de B3 e o observado na imagem B3 é bem inferior à incerteza radiométrica do pixel analizado.

$$Anorm := \left| \begin{array}{l} \text{for } i \in 0 \,..\, rows(AGB^{\langle 0 \rangle}) - 1 \\ \quad Out_i \leftarrow round\left((AGB^{\langle 0 \rangle})_i, 0 \right) \\ Out \end{array} \right.$$

$$\frac{AGB \cdot XGB}{VecB} \qquad XGB = \begin{bmatrix} 0.8 \\ 2361.552 \end{bmatrix}$$

$$csort(augment(Vnorm, Anorm), 0) = \begin{bmatrix} \vdots & \\ 486.016 & 15172 \\ 486.196 & 15011 \\ 486.395 & 15084 \\ 486.762 & 12094 \\ 487.396 & 11758 \\ 487.463 & 13929 \\ 490.456 & 12243 \\ 491.434 & 14769 \\ 491.869 & 13982 \\ 492.305 & 14270 \\ 493.364 & 15339 \\ 495.849 & 14646 \\ & \vdots \end{bmatrix}$$

$min(Anorm) = 7491$

$max(Anorm) = 17137$

$B2(B3) := B3 \cdot XGB_0 + XGB_1 \qquad B2(VecG_0) - VecB_0 = -168.824$

Equação para mapear os valores dos pixels, na área de água, entre os canais B3 e B4. Será usada na determinação dos valores de bottom quando houver a indeterminação pelo modelo que será apresentado mais a frente.

$min(VecB) = 8456$

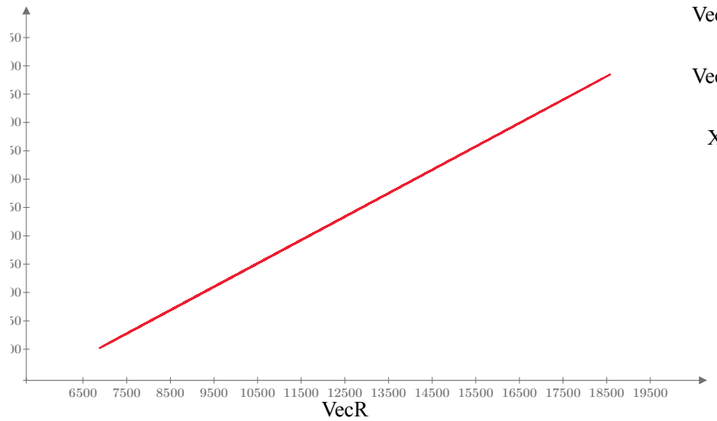
$min(VecG) = 7491 \qquad min(VecR) = 6870$

$$ARB := \left\| \begin{array}{l} im \leftarrow VecR \\ \text{for } i \in 0..(\text{rows}(im)-1) \\ \quad \left\| \begin{array}{l} Out_{i,0} \leftarrow im_i \\ Out_{i,1} \leftarrow 1 \end{array} \right. \\ Out \end{array} \right.$$

$$LRB := VecB \qquad r2B2B4 = 0.94$$

$$XRB := (ARB^T \cdot ARB)^{-1} \cdot (ARB^T \cdot LRB)$$

$$VRB := ARB \cdot XRB - LRB \qquad \sqrt{\frac{(VRB^T \cdot VRB)}{\text{rows}(VRB)-1}} = 304.419 \qquad \frac{\sum_{i=0}^{\text{rows}(VRB)-1} |VRB_i|}{\text{rows}(VRB)} = 223.698$$



$$VecR_0 \cdot XRB_0 + XRB_1 = 9747.929 \qquad VecB_0 = 10502$$

$$VecR_{150} \cdot XRB_0 + XRB_1 = 10420.848 \qquad VecB_{150} = 10361$$

$$XRB = \begin{bmatrix} 0.702 \\ 2905.066 \end{bmatrix}$$

$$\frac{ARB \cdot XRB}{VecB}$$

$$B2(B4) := B4 \cdot XRB_0 + XRB_1 \qquad B2(VecR_0) - VecB_0 = -754.071$$

Equação para mapear os valores dos pixels, na área de água, entre os canais B2 e B4. Será usada na determinação dos valores de bottom quando houver a indeterminação pelo modelo que será apresentado mais a frente.

$$\gamma 23 := XGB_0 \qquad \xi 23 := XGB_1$$

$$\gamma 24 := XRB_0 \qquad \xi 24 := XRB_1$$

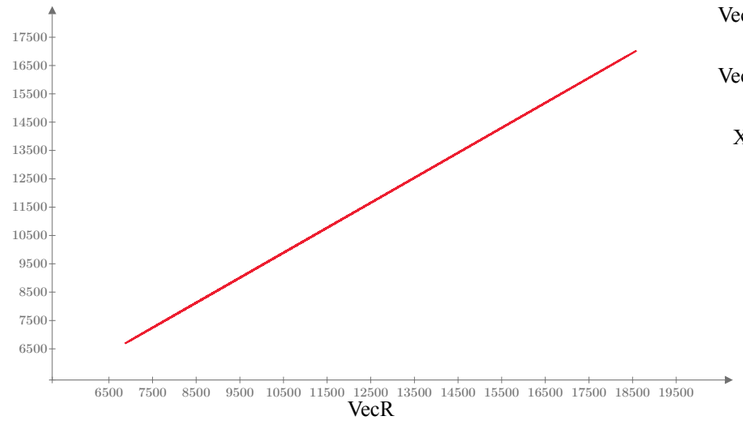
$$\gamma 34 := XRG_0 \qquad \xi 34 := XRG_1$$

$$ARG := \left\| \begin{array}{l} im \leftarrow VecR \\ \text{for } i \in 0..(\text{rows}(im)-1) \\ \quad \left\| \begin{array}{l} Out_{i,0} \leftarrow im_i \\ Out_{i,1} \leftarrow 1 \end{array} \right. \\ Out \end{array} \right.$$

$$LRG := VecG \qquad r2B3B4 = 0.96$$

$$XRG := (ARG^T \cdot ARG)^{-1} \cdot (ARG^T \cdot LRG)$$

$$VRG := ARG \cdot XRG - LRG \qquad \sqrt{\frac{(VRG^T \cdot VRG)}{\text{rows}(VRG)-1}} = 304.966 \qquad \frac{\sum_{i=0}^{\text{rows}(VRG)-1} |VRG_i|}{\text{rows}(VRG)} = 237.771$$



$$VecR_0 \cdot XRG_0 + XRG_1 = 9230.732 \qquad VecG_0 = 9963$$

$$VecR_{150} \cdot XRG_0 + XRG_1 = 10076.454 \qquad VecG_{150} = 10181$$

$$XRG = \begin{bmatrix} 0.882 \\ 630.653 \end{bmatrix}$$

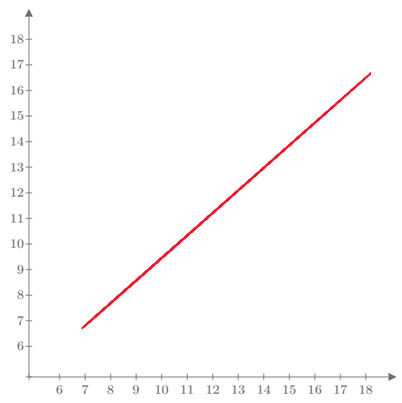
$$\frac{ARG \cdot XRG}{VecG}$$

$$B2(B4) := B4 \cdot XRB_0 + XRB_1 \qquad B2(VecR_0) - VecB_0 = -754.071$$

Equação para mapear os valores dos pixels, na área de água, entre os canais B2 e B4. Será usada na determinação dos valores de bottom quando houver a indeterminação pelo modelo que será apresentado mais a frente.

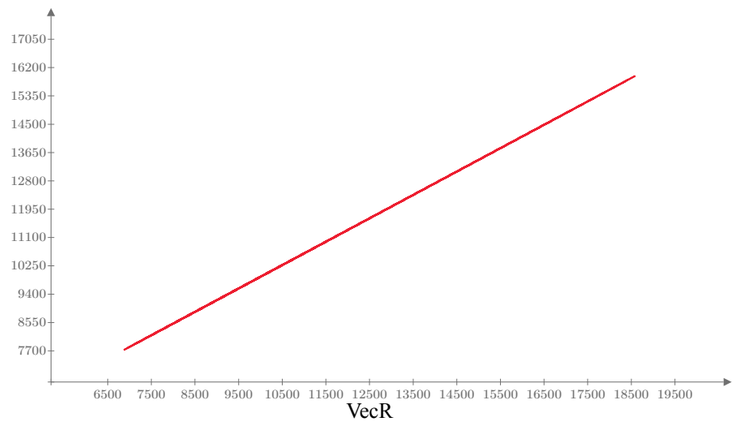
$$\begin{bmatrix} \gamma 23 & \xi 23 \\ \gamma 24 & \xi 24 \\ \gamma 34 & \xi 34 \end{bmatrix} = \begin{bmatrix} 0.8 & 2361.552 \\ 0.702 & 2905.066 \\ 0.882 & 630.653 \end{bmatrix}$$

$$Y1 := \frac{ARG \cdot XRG}{1000} \qquad Y2 := \frac{VecG}{1000} \qquad X1 := \frac{VecR}{1000} \qquad X2 := \frac{VecR}{1000}$$



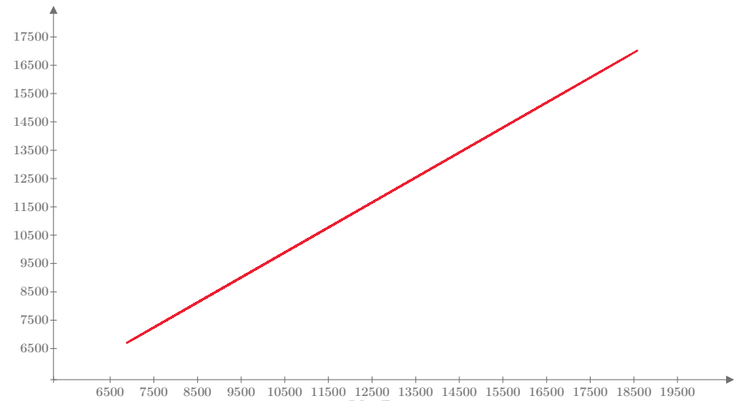
$$\frac{ARB \cdot XRB}{1000}$$

$$\frac{VecB}{1000}$$



$$\frac{ARB \cdot XRB}{VecB}$$

$$\frac{VecR}{VecR}$$



$$\frac{ARG \cdot XRG}{VecG}$$

$$\frac{VecR}{VecR}$$

Matching WGS84 UTM 4N to Pixel Coords

$$dim := rows(im3B2W)$$

$$rmseX := Limits(Files_0)_{2,0} = 3.095$$

$$rmseY := Limits(Files_0)_{2,1} = 3.226$$

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.

```
SOUNDGALB :=
    CWD ← "C:\PhD\2015\00 Papers\Paper 3 - Simeonof Island\03 - MBES\"
    aa ← READTEXT(concat(CWD, "h12103_clipped.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)
```

```
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) ∧ 30 > 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}) + (1.521 · 0))
            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
                c1 ← c1 + 1
                row ← rows(Ss_{i,j})
                if row > 1
```

```
                                                
```

```
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 ∧ 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,

```
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
```

```
                                                
```

ED0 := 0 ED1 := 30 Extinction depth

$$[aaa\ \ lll\ \ ddd] := \left| \begin{array}{l} \text{for } i \in 0..\text{rows}(Sounds2)-1 \\ \quad \text{for } j \in 0..\text{cols}(Sounds2)-1 \\ \quad\quad \text{if } 30 > Sounds2_{i,j} > 0 \wedge im3B2W_{i,j} \neq -9999 \\ \quad\quad\quad L_k \leftarrow Sounds2_{i,j} \\ \quad\quad\quad \left[AA_{k,0}\ \ AA_{k,1} \right] \leftarrow \left[\ln\left(\frac{im3B2W_{i,j}}{im3B3W_{i,j}} \right)\ \ 1 \right] \\ \quad\quad\quad k \leftarrow k+1 \\ \quad\quad \text{else} \\ \quad\quad\quad \text{continue} \\ X \leftarrow \left\langle AA^{T} \cdot AA \right\rangle^{-1} \cdot \left\langle AA^{T} \cdot L \right\rangle \\ DDD \leftarrow AA \cdot X \\ [AA\ \ L\ \ DDD] \end{array} \right|$$

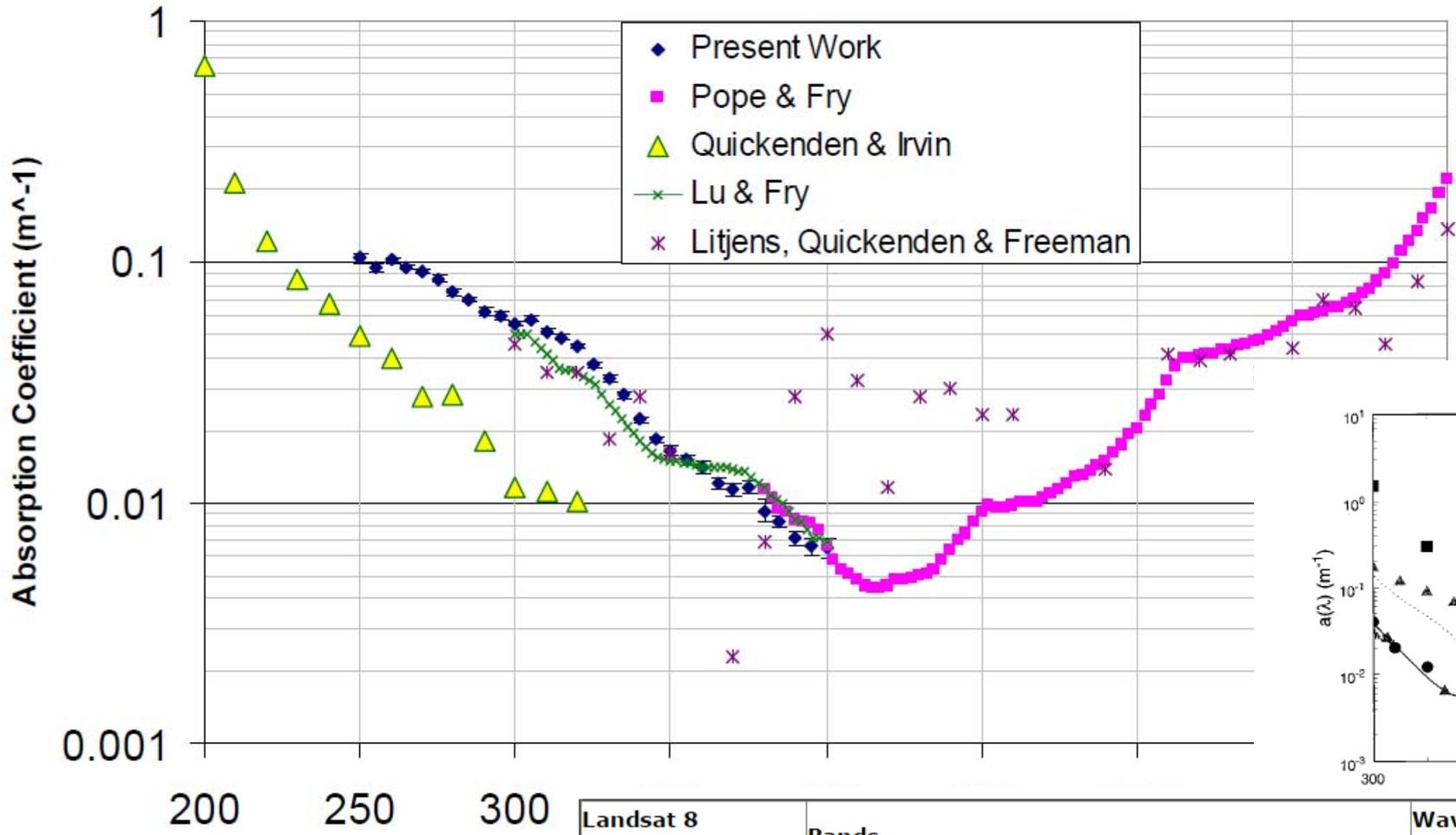
$\text{rows}(Sounds2) = 178$ $\text{rows}(\sigma Sounds2) = 178$

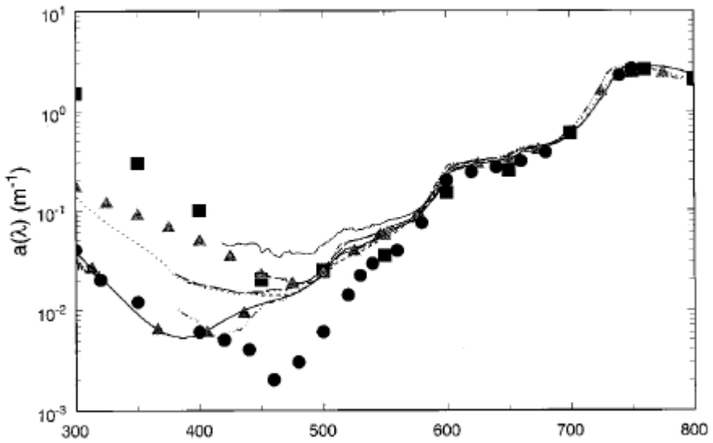
$\text{cols}(Sounds2) = 178$ $\text{cols}(\sigma Sounds2) = 178$

$$xxx := \left\langle aaa^{T} \cdot aaa \right\rangle^{-1} \cdot \left\langle aaa^{T} \cdot lll \right\rangle = \begin{bmatrix} 524.426 \\ -88.795 \end{bmatrix}$$

$DDD := aaa \cdot xxx$ $VVV := DDD - lll$

$$RMSElin := \sqrt{\frac{VVV^{T} \cdot VVV}{\text{rows}(VVV)}}$$





Landsat 8 Operational Land Imager (OLI) and Thermal Infrared Sensor (TIRS)	Bands	Wavelength (micrometers)	Resolution (meters)
	Band 1 - Ultra Blue (coastal/aerosol)	0.43 - 0.45	30
	Band 2 - Blue	0.45 - 0.51	30
	Band 3 - Green	0.53 - 0.59	30
	Band 4 - Red	0.64 - 0.67	30
	Band 5 - Near Infrared (NIR)	0.85 - 0.88	30
	Band 6 - Shortwave Infrared (SWIR) 1	1.57 - 1.65	30
	Band 7 - Shortwave Infrared (SWIR) 2	2.11 - 2.29	30
	Band 8 - Panchromatic	0.50 - 0.68	15
	Band 9 - Cirrus	1.36 - 1.38	30
	Band 10 - Thermal Infrared (TIRS) 1	10.60 - 11.19	100 * (30)
	Band 11 - Thermal Infrared (TIRS) 2	11.50 - 12.51	100 * (30)

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

$ED0 := 0$ $ED1 := 30$ Extinction depth

Continuar modificando, no fim exportará uma matriz de profundidades (baseadas na melhor escolha de par de canais por subset), os parametros Lw associados a cada subset, bem como os Lb, calculados usando a função B4(B3) ou outras de outros canais. Calcular Lw usando full non-linear ajustamento e ver se há necessidade desse processamento adicional. Com base nos valores Lb, convertidos nos diversos subsets para um mesmo canal, usar algum algoritmo de cluster, tipo isodata.

$$F(m0, Lw2, Lw3, m1, B2, B3) := \left| \begin{array}{l} \text{for } i \in 0..\text{rows}(B2) - 1 \\ \quad Out_i \leftarrow m0 \cdot \ln\left(\frac{B2_i - Lw2}{B3_i - Lw3}\right) + m1 \\ Out \end{array} \right.$$

$$\sigma FiltIm0(im\#B\#W, i\#B\#) := \left| \begin{array}{l} \sigma \leftarrow \sigma filt(i\#B\#) \\ k \leftarrow 0 \\ \text{for } i \in 0..\text{rows}(Sounds2) - 1 \\ \quad \text{for } j \in 0..\text{cols}(Sounds2) - 1 \\ \quad\quad \text{if } ED1 > Sounds2_{i,j} > ED0 \wedge im\#B\#W_{i,j} \neq -9999 \\ \quad\quad\quad Out_k \leftarrow \sigma_{i,j} \\ \quad\quad\quad k \leftarrow k + 1 \\ \quad\quad \text{else} \\ \quad\quad\quad \text{continue} \\ Out \end{array} \right.$$

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

$$WeightedCorr2(a, b, w) := \left| \begin{array}{l} lim \leftarrow \text{rows}(a) - 1 \\ W \leftarrow \sum_{i=0}^{lim} w_i \\ \mu a \leftarrow \frac{\sum_{i=0}^{lim} (w_i \cdot a_i)}{W} \\ \mu b \leftarrow \frac{\sum_{i=0}^{lim} (w_i \cdot b_i)}{W} \\ CovAB \leftarrow \frac{\sum_{i=0}^{lim} (w_i \cdot (a_i - \mu a) \cdot (b_i - \mu b))}{W} \\ CovAA \leftarrow \frac{\sum_{i=0}^{lim} (w_i \cdot (a_i - \mu a)^2)}{W} \\ CovBB \leftarrow \frac{\sum_{i=0}^{lim} (w_i \cdot (b_i - \mu b)^2)}{W} \\ \frac{CovAB^2}{CovAA \cdot CovBB} \end{array} \right.$$

This square weighted correlation minimum value is used to provide the Lw1 and Lw2 for each subset. This way, Lw determination is coherent to weighted linear adjustment that later determines m0 and m1.

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

$$AdjustLin2(A, L) := \left| \begin{array}{l} N \leftarrow \begin{bmatrix} \sum_{n=0}^{\text{rows}(A)-1} (A_{n,0})^2 & \sum_{n=0}^{\text{rows}(A)-1} (A_{n,0}) \\ \sum_{n=0}^{\text{rows}(A)-1} (A_{n,0}) & \sum_{n=0}^{\text{rows}(A)-1} 1 \end{bmatrix} \\ U \leftarrow \begin{bmatrix} \sum_{n=0}^{\text{rows}(A)-1} (A_{n,0}) \cdot (L_n) \\ \sum_{n=0}^{\text{rows}(A)-1} (L_n) \end{bmatrix} \\ \begin{bmatrix} N_{1,1} \cdot U_0 - N_{0,1} \cdot U_1 \\ -N_{1,0} \cdot U_0 + N_{0,0} \cdot U_1 \end{bmatrix} \cdot \frac{1}{N_{0,0} \cdot N_{1,1} - N_{1,0} \cdot N_{0,1}} \end{array} \right.$$

$$\text{Subsets}(\text{im3B2W},\text{im3B3W},\text{mrow},\text{Mrow},\text{mcol},\text{Mcol}) :=$$
for $i \in \text{mrow}..\text{Mrow}$
  for $j \in \text{mcol}..\text{Mcol}$
    if $\text{ED1} > \text{Sounds2}_{i,j} > \text{ED0} \wedge \text{im3B2W}_{i,j} \neq -9999$
      $\begin{bmatrix} b2_k & b3_k & L_k & w_k \end{bmatrix} \leftarrow \begin{bmatrix} \text{im3B2W}_{i,j} & \text{im3B3W}_{i,j} & \text{Sounds2}_{i,j} & \sigma\text{Sounds2}_{i,j} \end{bmatrix}$
      $k \leftarrow k+1$
    else
      continue
$\begin{bmatrix} b2 & b3 & L & w \end{bmatrix}$

Definição dos parâmetros para usar no ajustamento, os vetores B2, B3, L e w, baseados na janela de observação escolhida

$$B2(\text{im3B2W},\text{im3B3W},\text{mrow},\text{Mrow},\text{mcol},\text{Mcol}) := \left(\text{Subsets}(\text{im3B2W},\text{im3B3W},\text{mrow},\text{Mrow},\text{mcol},\text{Mcol})^{\langle 0 \rangle}\right)_0$$

$$B3(\text{im3B2W},\text{im3B3W},\text{mrow},\text{Mrow},\text{mcol},\text{Mcol}) := \left(\text{Subsets}(\text{im3B2W},\text{im3B3W},\text{mrow},\text{Mrow},\text{mcol},\text{Mcol})^{\langle 1 \rangle}\right)_0$$

$$L(\text{im3B2W},\text{im3B3W},\text{mrow},\text{Mrow},\text{mcol},\text{Mcol}) := \left(\text{Subsets}(\text{im3B2W},\text{im3B3W},\text{mrow},\text{Mrow},\text{mcol},\text{Mcol})^{\langle 2 \rangle}\right)_0$$

$$w(\text{im3B2W},\text{im3B3W},\text{mrow},\text{Mrow},\text{mcol},\text{Mcol}) := \left(\text{Subsets}(\text{im3B2W},\text{im3B3W},\text{mrow},\text{Mrow},\text{mcol},\text{Mcol})^{\langle 3 \rangle}\right)_0$$

$$A(\text{im3B2W},\text{im3B3W},\text{mrow},\text{Mrow},\text{mcol},\text{Mcol},\text{Lw2},\text{Lw3}) :=$$
for $i \in \text{mrow}..\text{Mrow}$
  for $j \in \text{mcol}..\text{Mcol}$
    if $\text{ED1} > \text{Sounds2}_{i,j} > \text{ED0} \wedge \text{im3B2W}_{i,j} \neq -9999$
      $\begin{bmatrix} A_{k,0} & A_{k,1} \end{bmatrix} \leftarrow \begin{bmatrix} \ln\left(\dfrac{\text{im3B2W}_{i,j} - \text{Lw2}}{\text{im3B3W}_{i,j} - \text{Lw3}}\right) & 1 \end{bmatrix}$
      $k \leftarrow k+1$
    else
      continue
A

$$X(\text{im3B2W},\text{im3B3W},\text{mrow},\text{Mrow},\text{mcol},\text{Mcol},\text{Lw2},\text{Lw3}) :=$$
for $i \in \text{mrow}..\text{Mrow}$
  for $j \in \text{mcol}..\text{Mcol}$
    if $\text{ED1} > \text{Sounds2}_{i,j} > \text{ED0} \wedge \text{im3B2W}_{i,j} \neq -9999$
      $\begin{bmatrix} A_{k,0} & A_{k,1} & L_k \end{bmatrix} \leftarrow \begin{bmatrix} \ln\left(\dfrac{\text{im3B2W}_{i,j} - \text{Lw2}}{\text{im3B3W}_{i,j} - \text{Lw3}}\right) & 1 & \text{Sounds2}_{i,j} \end{bmatrix}$
      $k \leftarrow k+1$
    else
      continue
$(A^T \cdot A)^{-1} \cdot (A^T \cdot L)$

$$V(\text{im3B2W},\text{im3B3W},\text{mrow},\text{Mrow},\text{mcol},\text{Mcol},\text{Lw2},\text{Lw3}) :=$$
for $i \in \text{mrow}..\text{Mrow}$
  for $j \in \text{mcol}..\text{Mcol}$
    if $\text{ED1} > \text{Sounds2}_{i,j} > \text{ED0} \wedge \text{im3B2W}_{i,j} \neq -9999$
      $\begin{bmatrix} A_{k,0} & A_{k,1} & L_k \end{bmatrix} \leftarrow \begin{bmatrix} \ln\left(\dfrac{\text{im3B2W}_{i,j} - \text{Lw2}}{\text{im3B3W}_{i,j} - \text{Lw3}}\right) & 1 & \text{Sounds2}_{i,j} \end{bmatrix}$
      $k \leftarrow k+1$
    else
      continue
$A \cdot (A^T \cdot A)^{-1} \cdot (A^T \cdot L) - L$

$$\text{VtV}(\text{im3B2W},\text{im3B3W},\text{mrow},\text{Mrow},\text{mcol},\text{Mcol},\text{Lw2},\text{Lw3}) := \frac{V \leftarrow V(\text{im3B2W},\text{im3B3W},\text{mrow},\text{Mrow},\text{mcol},\text{Mcol},\text{Lw2},\text{Lw3})}{\sqrt{V^T \cdot V}}$$

$$\text{Corr2}(\text{im3B2W},\text{im3B3W},\text{mrow},\text{Mrow},\text{mcol},\text{Mcol},\text{Lw2},\text{Lw3}) :=$$
for $i \in \text{mrow}..\text{Mrow}$
  for $j \in \text{mcol}..\text{Mcol}$
    if $\text{ED1} > \text{Sounds2}_{i,j} > \text{ED0} \wedge \text{im3B2W}_{i,j} \neq -9999$
      $\begin{bmatrix} A_k & L_k \end{bmatrix} \leftarrow \begin{bmatrix} \ln\left(\dfrac{\text{im3B2W}_{i,j} - \text{Lw2}}{\text{im3B3W}_{i,j} - \text{Lw3}}\right) & \text{Sounds2}_{i,j} \end{bmatrix}$
      $k \leftarrow k+1$
    else
      continue
$\text{corr}(A, L)^2$

by using $(n - M_{i,j}) \cdot (y - X_{i,j}) \leq 0$ we eliminate water column values that could return ln(0) or ln (infinite) as well as ln() values in Complex domain.

$$\text{IsElementOf2}(n, M, y, X) :=$$
$\text{Out} \leftarrow 0$
for $i \in 0..\text{rows}(M) - 1$
  for $j \in 0..\text{cols}(M) - 1$
    if $(n - M_{i,j}) \cdot (y - X_{i,j}) \leq 0$
      $\text{Out} \leftarrow 1$
      break
Out

$$\text{Passo} := 100$$

```
VC(im1, im2, mrow, Mrow, mcol, Mcol) :=
    t1 ← time(1)
    for i ∈ mrow .. Mrow
        for j ∈ mcol .. Mcol
            if ED1 > Sounds2[i,j] > ED0 ∧ im3B2W[i,j] ≠ −9999
                [B2[k]  B3[k]  L[k]] ← [im3B2W[i,j]  im3B3W[i,j]  Sounds2[i,j]]
                k ← k + 1
            else
                continue
    M1 ← submatrix(im1, mrow, Mrow, mcol, Mcol)
    M2 ← submatrix(im2, mrow, Mrow, mcol, Mcol)
    if im1 = im3B2W
        maxIm1 ← max(M1) − 1
        minIm1 ← 5000
    if im1 = im3B3W
        maxIm1 ← max(M1) − 1
        minIm1 ← 5000
    if im2 = im3B3W
        maxIm2 ← max(M2) − 1
        minIm2 ← 5000
    if im2 = im3B4W
        maxIm2 ← max(M2) − 1
        minIm2 ← 5000
    if rows(L) > 2
        s ← Passo
        Corr00 ← Corr2(im1, im2, mrow, Mrow, mcol, Mcol, 0, 0)
        for m ∈ trunc(minIm1 / s) .. trunc(maxIm1 / s)
            for n ∈ trunc(minIm2 / s) .. trunc(maxIm2 / s)
                try
                    kkk ← kkk + 1
                    if IsElementOf2(m · s, M1, n · s, M2) = 0
                        Corr0 ← Corr2(im1, im2, mrow, Mrow, mcol, Mcol, m · s, n · s)
                        if Im(Corr0) = 0
                            if (Corr0 > Corr00) ∧ (m > n)
                                Corr00 ← Corr0
                                [Out[0,0]  Out[0,1]  Out[0,2]] ← [m · s  n · s  Corr00]
                    else
                        continue
                on error
                    continue
        if Out ≠ 0
            Out ← Out
        else
            [Out[0,0]  Out[0,1]  Out[0,2]] ← [0  0  Corr2(im1, im2, mrow, Mrow, mcol, Mcol, 0, 0)]
    else
        Out ← "not enough soundings"
    t2 ← time(2)
    Δt ← t2 − t1
    augment(Out, Δt, kkk)
```

Cálculo dos valores das componentes da coluna d'água para os dois canais escolhidos, com base na definição da janela de análise.

```
IsElementOf(n, M) :=
    Out ← 0
    for i ∈ 0 .. rows(M) − 1
        for j ∈ 0 .. cols(M) − 1
            if (M[i,j] − n) ≤ 0
                Out ← 1
                break
    Out
```

```
m0(im3B2W, im3B3W, mrow, Mrow, mcol, Mcol) :=
    L ← L(im3B2W, im3B3W, mrow, Mrow, mcol, Mcol)
    B2 ← B2(im3B2W, im3B3W, mrow, Mrow, mcol, Mcol)
    B3 ← B3(im3B2W, im3B3W, mrow, Mrow, mcol, Mcol)
    WC ← WC(im3B2W, im3B3W, mrow, Mrow, mcol, Mcol)
    if rows(L) > 2 ∧ WC[0,0] ≠ "not enough soundings"
        Corr00 ← Corr2(im3B2W, im3B3W, mrow, Mrow, mcol, Mcol, 0, 0)
        "m0 e m1 derivados do ajustamento linear:"
        A0 ← A(im3B2W, im3B3W, mrow, Mrow, mcol, Mcol, 0, 0)
        [M0w  M1w] ← (AdjustLin2(A0, L))^T
        Lw20 ← WC[0,0]
        Lw30 ← WC[0,1]
        A1 ← A(im3B2W, im3B3W, mrow, Mrow, mcol, Mcol, Lw20, Lw30)
        [M0  M1] ← (AdjustLin2(A1, L))^T

        X0 ← [M0  Lw20  Lw30  M1]^T
        V0 ← F(X0[0], X0[1], X0[2], X0[3], B2, B3) − L
        RSSE0 ← √((V0)^T • (V0))
        if rows(L) > 4
            try
                for k ∈ 0..1000
                    bug ← 0
                    try
                        L0 ← F(X0[0], X0[1], X0[2], X0[3], B2, B3)
                        for i ∈ 0..rows(B2) − 1
                            A[i,0] ← ln((X0[1] − B2[i]) / (X0[2] − B3[i]))
                            A[i,1] ← X0[0] / (X0[1] − B2[i])
                            A[i,2] ← −X0[0] / (X0[2] − B3[i])
                            A[i,3] ← 1
                    on error
                        break
                    ΔL ← L − L0
                    ΔX ← (A^T • A)^(−1) • (A^T • ΔL)
                    X ← X0 + ΔX
                    V1 ← F(X[0], X[1], X[2], X[3], B2, B3) − L
                    RSSE1 ← √((V1)^T • (V1))
                    try
                        if (min(B2) > X[1] > 5000) ∧ (min(B3) > X[2] > 5000) ∧ (RSSE1 < RSSE0)
                            X0 ← X
                            RSSE0 ← RSSE1
                        else
                            out ← k
                            break
                    on error
                        bug ← 1
```

```
                    bug ← 1
                    try
                        X0 ← X0
                    on error
                        X0 ← X0
                    break
            on error
                X0 ← X0

        [M0  Lw20  Lw30  M1] ← [X0_0  X0_1  X0_2  X0_3]
        for i ∈ mrow .. Mrow
            for j ∈ mcol .. Mcol
                D ← max(M0 · ln((im3B2W_i,j − Lw20) / (im3B3W_i,j − Lw30)) + M1 , 0)
                if D > 0
                    ED ← ((im3B3W_i,j − Lw30) · e^((D − M1)/M0)) / im3B2W_i,j
                else
                    ED ← −9999
                if Im(D) ≠ 0
                    D ← −9999
                    ED ← −9999
                else
                    D ← D
                    ED ← ED
                Depths_(i − mrow , j − mcol) ← D
                EDs_(i − mrow , j − mcol) ← ED
            AD ← Σ_(kkk=0)^(rows(L) − 1) (|F(X0_0 , X0_1 , X0_2 , X0_3 , B2 , B3)_kkk − L_kkk|)
            if rows(V0) > 4
                n ← rows(V0) − 4
            else
                n ← rows(V0) − 2
            N ← rows(V0)
            VtV ← RSSE0^2
        else
            for i ∈ mrow .. Mrow
                for j ∈ mcol .. Mcol
                    Depths_(i − mrow , j − mcol) ← −9999
            VtV ← "no depths"
            n ← "no depths"
        if rows(L) ≤ 2 ∨ WC_0,0 = "not enough soundings"
            Out ← [Depths  VtV  n]
        else
            try
                Out ← [Depths  VtV  n  Lw20  Lw30  e^(−M1/M0)  Lw20 − (Lw30 · e^(−M1/M0))  AD  N  EDs  M0]
```

```
                on error
                    Out ← [Depths  VtV  n]
    Out
```

Em razão dos valores exponenciais, os dois últimos termos estavam dando resultados acima de 10^308 e o Mathcad dava erro. Assim, aproveitamos as sondagens estimadas mas quando ocorrer esses erros de estouro, não haverá estimativa de tipo de fundo.

$ED0 := 0 \qquad ED1 := 30$

```
SubsetsCh(im3B2W, im3B3W, im3B4W, mrow, Mrow, mcol, Mcol)
    for i ∈ mrow .. Mrow
        for j ∈ mcol .. Mcol
            if im3B2W_{i,j} ≠ −9999
                [b2_k  b3_k  b4_k] ← [im3B2W_{i,j}  im3B3W_{i,j}  im3B4W_{i,j}]
                k ← k + 1
            else
                continue
    [b2  b3  b4]
```

$B2(im3B2W, im3B3W, im3B4W, mrow, Mrow, mcol, Mcol)\ (SubsetsCh(im3B2W, im3B3W, im3B4W, mrow, Mrow, mcol, Mcol)^{\langle 0 \rangle})_0$

$B3(im3B2W, im3B3W, im3B4W, mrow, Mrow, mcol, Mcol)\ (SubsetsCh(im3B2W, im3B3W, im3B4W, mrow, Mrow, mcol, Mcol)^{\langle 1 \rangle})_0$

$B4(im3B2W, im3B3W, im3B4W, mrow, Mrow, mcol, Mcol)\ (SubsetsCh(im3B2W, im3B3W, im3B4W, mrow, Mrow, mcol, Mcol)^{\langle 2 \rangle})_0$

```
Jacobiana(ED1) :=
    for i ∈ 0 .. rows(im3B3W) − 1
        for j ∈ 0 .. cols(im3B3W) − 1
            if ED1 > Sounds2_{i,j} > ED0 ∧ im3B3W_{i,j} ≠ −9999
                J_{k,0} ← im3B3W_{i,j}
                J_{k,1} ← 1
                k ← k + 1
            else
                continue
    J
```

Caso Linear

```
ImLin(im3B2W, im3B3W, mrow, Mrow, mcol, Mcol) :=
    L ← L(im3B2W, im3B3W, mrow, Mrow, mcol, Mcol)
    B2 ← B2(im3B2W, im3B3W, mrow, Mrow, mcol, Mcol)
    B3 ← B3(im3B2W, im3B3W, mrow, Mrow, mcol, Mcol)
    if rows(L) > 2
        "m0 e m1 derivados do ajustamento linear:"
        A0 ← A(im3B2W, im3B3W, mrow, Mrow, mcol, Mcol, 0, 0)
        [M0  M1] ← (AdjustLin2(A0, L))^T
        [M0  M1] ← (⟨A0^T • A0⟩^-1 • ⟨A0^T • L⟩)^T
        for i ∈ mrow .. Mrow
            for j ∈ mcol .. Mcol
                D ← max(M0 • ln(im3B2W_i,j / im3B3W_i,j) + M1, 0)
                if Im(D) ≠ 0
                    D ← −9999
                else
                    D ← D
                Depths_(i − mrow, j − mcol) ← D
        n ← rows(L)
        VtV ← (A0 • [M0  M1]^T − L)^T • (A0 • [M0  M1]^T − L)
        AD ← Σ_(kkk=0)^(rows(L) − 1) (|(A0 • [M0  M1]^T)_kkk − L_kkk|)
    else
        for i ∈ mrow .. Mrow
            for j ∈ mcol .. Mcol
                Depths_(i − mrow, j − mcol) ← −9999
        VtV ← "no depths"
        n ← "no depths"
    if rows(L) ≤ 2
        Out ← [Depths  VtV  n]
    else
        try
            Out ← [Depths  VtV  n  AD]
        on error
            Out ← [Depths  VtV  n  AD]
    Out
```

SubsetsYuri(im3B2W, im3B3W, mrow, Mrow, mcol, Mcol) :=

```
for i ∈ mrow .. Mrow
    for j ∈ mcol .. Mcol
        if ED1 > Sounds2[i,j] > ED0 ∧ im3B2W[i,j] ≠ −9999
            [b2[k] b3[k] L[k] II[k] JJ[k]] ← [im3B2W[i,j] im3B3W[i,j] Sounds2[i,j] i j]
            k ← k + 1
        else
            continue
[b2 b3 L II JJ]
```

rowmin1 := 0 = 0

rowmax1 := rows(im3B2W) − 1 = 177

colmin1 := 0 = 0

colmax1 := cols(im3B2W) − 1 = 177

BB2(a, b, c, d) := (SubsetsYuri(im3B2W, im3B3W, a, b, c, d)^⟨0⟩)_0

LL(a, b, c, d) := (SubsetsYuri(im3B2W, im3B3W, a, b, c, d)^⟨2⟩)_0

BB3(a, b, c, d) := (SubsetsYuri(im3B2W, im3B3W, a, b, c, d)^⟨1⟩)_0

II(a, b, c, d) := (SubsetsYuri(im3B2W, im3B3W, a, b, c, d)^⟨3⟩)_0

JJ(a, b, c, d) := (SubsetsYuri(im3B2W, im3B3W, a, b, c, d)^⟨4⟩)_0

```
a ← "YuriData1.txt"
WRITETEXT(a, augment(BB2(rowmin1, rowmax1, colmin1, colmax1), BB3(rowmin1, rowmax1, colmin1, colmax1), LL(rowmin1, rowmax1, colmin1, colmax1), JJ(rowmin1, rowmax1, colmin1, colmax1), II(rowmin1, rowmax1, colmin1, colmax1)))
```

CWD = "C:\PhD\2015\00 Papers\Paper 3 - Simeonof Island\"

```
DepthsLin(step) := minR ← 0
                   maxR ← round(rows(Sounds2) / step)
                   minC ← 0
                   maxC ← round(cols(Sounds2) / step)
                   m ← 0
                   mm ← 0
                   mmm ← 0
                   for i ∈ minR .. (maxR)
                     for j ∈ minC .. (maxC)
                       check ← 0
                       if ((i · step < rows(Sounds2) − 1))
                         minrow ← i · step
                         maxrow ← min(i · step + step − 1 , rows(Sounds2) − 1)
                       else
                         if ((i · step = rows(Sounds2) − 1))
                           minrow ← rows(Sounds2) − 1
                           maxrow ← rows(Sounds2) − 1
                         else
                           check ← "error"
                       if ((j · step < cols(Sounds2) − 1))
                         mincol ← j · step
                         maxcol ← min(j · step + step − 1 , cols(Sounds2) − 1)
                       else
                         if ((j · step = cols(Sounds2) − 1))
                           mincol ← cols(Sounds2) − 1
                           maxcol ← cols(Sounds2) − 1
                         else
                           check ← "error"
                     if check = 0
                       Img ← submatrix(im3B3W , minrow , maxrow , mincol , maxcol)
                       try
                         IM23 ← ImLin(im3B2W , im3B3W , minrow , maxrow , mincol , maxcol)
                         M23 ← ⟨IM23^⟨0⟩⟩_0
                         Error23 ← ⟨IM23^⟨1⟩⟩_0
                       on error
                         IM23 ← "error"
                       try
                         IM34 ← ImLin(im3B3W , im3B4W , minrow , maxrow , mincol , maxcol)
                         M34 ← ⟨IM34^⟨0⟩⟩_0
                         Error34 ← ⟨IM34^⟨1⟩⟩_0
                       on error
                         IM34 ← "error"
                       try
                         IM24 ← ImLin(im3B2W , im3B4W , minrow , maxrow , mincol , maxcol)
                         M24 ← ⟨IM24^⟨0⟩⟩_0
                         Error24 ← ⟨IM24^⟨1⟩⟩_0
```

$$xxx := \left(aaa^{T} \cdot aaa\right)^{-1} \cdot \left(aaa^{T} \cdot lll\right) = \begin{bmatrix} 524.426 \\ -88.795 \end{bmatrix}$$

on error

IM24 ← "error"

if Error23 ≠ "no depths"

VtVpart23$_{m}$ ← Error23

npart23$_{m}$ ← $\langle IM23^{\langle 2 \rangle} \rangle_{0}$

ADpart23$_{m}$ ← $\langle IM23^{\langle 3 \rangle} \rangle_{0}$

m ← m + 1

if Error34 ≠ "no depths"

VtVpart34$_{mm}$ ← Error34

npart34$_{mm}$ ← $\langle IM34^{\langle 2 \rangle} \rangle_{0}$

ADpart34$_{mm}$ ← $\langle IM34^{\langle 3 \rangle} \rangle_{0}$

mm ← mm + 1

if Error24 ≠ "no depths"

VtVpart24$_{mmm}$ ← Error24

npart24$_{mmm}$ ← $\langle IM24^{\langle 2 \rangle} \rangle_{0}$

ADpart24$_{mmm}$ ← $\langle IM24^{\langle 3 \rangle} \rangle_{0}$

mmm ← mmm + 1

for k ∈ 0 .. rows(M23) − 1

for l ∈ 0 .. cols(M23) − 1

if $Img_{k,l} > -9999$

$D23_{minrow+k,mincol+l} \leftarrow M23_{k,l}$

$D34_{minrow+k,mincol+l} \leftarrow M34_{k,l}$

$D24_{minrow+k,mincol+l} \leftarrow M24_{k,l}$

else

$D23_{minrow+k,mincol+l} \leftarrow -9999$

$D34_{minrow+k,mincol+l} \leftarrow -9999$

$D24_{minrow+k,mincol+l} \leftarrow -9999$

else

continue

$$\begin{bmatrix} D23 \\ D34 \\ D24 \end{bmatrix} \begin{bmatrix} \sqrt{\frac{\sum_{o=0}^{\text{rows}(VtVpart23)-1} VtVpart23_o}{\sum_{o=0}^{\text{rows}(VtVpart23)-1} npart23_o}} \\ \sqrt{\frac{\sum_{o=0}^{\text{rows}(VtVpart34)-1} VtVpart34_o}{\sum_{o=0}^{\text{rows}(VtVpart34)-1} npart34_o}} \\ \sqrt{\frac{\sum_{o=0}^{\text{rows}(VtVpart24)-1} VtVpart24_o}{\sum_{o=0}^{\text{rows}(VtVpart24)-1} npart24_o}} \end{bmatrix} \begin{bmatrix} \frac{\sum_{o=0}^{\text{rows}(ADpart23)-1} ADpart23_o}{\sum_{o=0}^{\text{rows}(VtVpart23)-1} npart23_o} \\ \frac{\sum_{o=0}^{\text{rows}(ADpart34)-1} ADpart34_o}{\sum_{o=0}^{\text{rows}(VtVpart34)-1} npart34_o} \\ \frac{\sum_{o=0}^{\text{rows}(ADpart24)-1} ADpart24_o}{\sum_{o=0}^{\text{rows}(VtVpart24)-1} npart24_o} \end{bmatrix}$$

Caso não Linear

```
DepthsNonLin(step) :=
    minR ← 0
    maxR ← round(rows(Sounds2) / step)
    minC ← 0
    maxC ← round(cols(Sounds2) / step)
    m ← 0
    mm ← 0
    mmm ← 0
    for i ∈ minR .. (maxR)
        for j ∈ minC .. (maxC)
            check ← 0
            if ((i • step < rows(Sounds2) − 1))
                minrow ← i • step
                maxrow ← min(i • step + step − 1 , rows(Sounds2) − 1)
            else
                if ((i • step = rows(Sounds2) − 1))
                    minrow ← rows(Sounds2) − 1
                    maxrow ← rows(Sounds2) − 1
                else
                    check ← "error"
            if ((j • step < cols(Sounds2) − 1))
                mincol ← j • step
                maxcol ← min(j • step + step − 1 , cols(Sounds2) − 1)
            else
                if ((j • step = cols(Sounds2) − 1))
                    mincol ← cols(Sounds2) − 1
                    maxcol ← cols(Sounds2) − 1
                else
                    check ← "error"
            if check = 0
                try
                    IM23 ← Im0(im3B2W , im3B3W , minrow , maxrow , mincol , maxcol)
                    M23 ← (IM23^⟨0⟩)_0
                    Error23 ← (IM23^⟨1⟩)_0
                on error
                    IM23 ← "error"
                try
                    IM34 ← Im0(im3B3W , im3B4W , minrow , maxrow , mincol , maxcol)
                    M34 ← (IM34^⟨0⟩)_0
                    Error34 ← (IM34^⟨1⟩)_0
                on error
                    IM34 ← "error"
                try
                    IM24 ← Im0(im3B2W , im3B4W , minrow , maxrow , mincol , maxcol)
                    M24 ← (IM24^⟨0⟩)_0
```

Error24 ← $\langle IM24^{(1)} \rangle_0$

on error

IM24 ← "error"

if Error23 ≠ "no depths" ∧ cols(IM23) = 11

VtVpart23$_m$ ← Error23

npart23$_m$ ← $\langle IM23^{(2)} \rangle_0$

Npart23$_m$ ← $\langle IM23^{(8)} \rangle_0$

ADpart23$_m$ ← $\langle IM23^{(7)} \rangle_0$

m ← m + 1

if Error34 ≠ "no depths" ∧ cols(IM34) = 11

VtVpart34$_{mm}$ ← Error34

npart34$_{mm}$ ← $\langle IM34^{(2)} \rangle_0$

Npart34$_{mm}$ ← $\langle IM34^{(8)} \rangle_0$

ADpart34$_{mm}$ ← $\langle IM34^{(7)} \rangle_0$

mm ← mm + 1

if Error24 ≠ "no depths" ∧ cols(IM24) = 11

VtVpart24$_{mmm}$ ← Error24

npart24$_{mmm}$ ← $\langle IM24^{(2)} \rangle_0$

Npart24$_{mmm}$ ← $\langle IM24^{(8)} \rangle_0$

ADpart24$_{mmm}$ ← $\langle IM24^{(7)} \rangle_0$

mmm ← mmm + 1

AdjustError ← 0

AdjustError2 ← 0

if (IM23 ≠ "error") ∧ (IM34 ≠ "error") ∧ (IM24 ≠ "error")

try

$$A \leftarrow \begin{bmatrix} 1 & -\langle IM23^{(5)} \rangle_0 & 0 \\ 0 & 1 & -\langle IM34^{(5)} \rangle_0 \\ 1 & 0 & -\langle IM24^{(5)} \rangle_0 \\ -1 & \gamma 23 & 0 \\ 0 & -1 & \gamma 34 \\ -1 & 0 & \gamma 24 \end{bmatrix}$$

$$L \leftarrow \begin{bmatrix} \langle IM23^{(6)} \rangle_0 \\ \langle IM34^{(6)} \rangle_0 \\ \langle IM24^{(6)} \rangle_0 \\ -\xi 23 \\ -\xi 34 \\ -\xi 24 \end{bmatrix}$$

$X \leftarrow (A^T \cdot A)^{-1} \cdot (A^T \cdot L)$

on error

AdjustError ← 1

try

$$A2 \leftarrow \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \\ 1 & 0 & -1 \end{bmatrix}$$

$\begin{bmatrix} (2 \cdot \langle IM23^{(10)} \rangle)^{-1} \end{bmatrix}$

```
        L2 ← [ -(2 · (IM34^(10))_0)^-1 ]
             [ -(2 · (IM24^(10))_0)^-1 ]
        X2 ← geninv(A2^T · A2) · (A2^T · L2)
      on error
        AdjustError2 ← 1
    else
      AdjustError ← 1
      AdjustError2 ← 1

    for k ∈ 0 .. rows(M23) − 1
      for l ∈ 0 .. cols(M23) − 1
        D23[minrow + k , mincol + l] ← M23[k , l]
        D34[minrow + k , mincol + l] ← M34[k , l]
        D24[minrow + k , mincol + l] ← M24[k , l]
        if AdjustError = 0
          if (8000 < X_0 < 20000) ∧ (cols(IM23) > 3) ∧ AdjustError = 0
            Bottom2[minrow + k , mincol + l] ← round(X_0 , 0)
          else
            Bottom2[minrow + k , mincol + l] ← −9999
          if (7000 < X_1 < 20000) ∧ (cols(IM23) > 3) ∧ AdjustError = 0
            Bottom3[minrow + k , mincol + l] ← round(X_1 , 0)
          else
            Bottom3[minrow + k , mincol + l] ← −9999
          if (6000 < X_2 < 20000) ∧ (cols(IM23) > 3) ∧ AdjustError = 0
            Bottom4[minrow + k , mincol + l] ← round(X_2 , 0)
          else
            Bottom4[minrow + k , mincol + l] ← −9999
        else
          Bottom2[minrow + k , mincol + l] ← −9999
          Bottom3[minrow + k , mincol + l] ← −9999
          Bottom4[minrow + k , mincol + l] ← −9999
        if AdjustError2 = 0
          if (0 < X2_0 < 1) ∧ (cols(IM23) > 3) ∧ AdjustError2 = 0
            KKK2[minrow + k , mincol + l] ← round(X2_0 , 3)
          else
            KKK2[minrow + k , mincol + l] ← −9999
          if (0 < X2_1 < 1) ∧ (cols(IM23) > 3) ∧ AdjustError2 = 0
            KKK3[minrow + k , mincol + l] ← round(X2_1 , 3)
          else
            KKK3[minrow + k , mincol + l] ← −9999
```

if $\left(0 < X2_2 < 1\right) \wedge \left(\text{cols}\left(IM23\right) > 3\right) \wedge AdjustError2 = 0$

$KKK4_{minrow + k\,,\,mincol + 1} \leftarrow \text{round}\left(X2_2\,,\,3\right)$

else

$KKK4_{minrow + k\,,\,mincol + 1} \leftarrow -9999$

else

$KKK2_{minrow + k\,,\,mincol + 1} \leftarrow -9999$

$KKK3_{minrow + k\,,\,mincol + 1} \leftarrow -9999$

$KKK4_{minrow + k\,,\,mincol + 1} \leftarrow -9999$

else

continue

$$\left[\text{csort}\left(\begin{bmatrix} D23 & \sqrt{\frac{\sum_{o=0}^{\text{rows}(VtVpart23)-1} VtVpart23_o}{\sum_{o=0}^{\text{rows}(VtVpart23)-1} npart23_o}} \\ D34 & \sqrt{\frac{\sum_{o=0}^{\text{rows}(VtVpart34)-1} VtVpart34_o}{\sum_{o=0}^{\text{rows}(VtVpart34)-1} npart34_o}} \\ D24 & \sqrt{\frac{\sum_{o=0}^{\text{rows}(VtVpart24)-1} VtVpart24_o}{\sum_{o=0}^{\text{rows}(VtVpart24)-1} npart24_o}} \end{bmatrix}, 1\right)_{0\,,\,0} \quad \begin{bmatrix} \sqrt{\frac{\sum_{o=0}^{\text{rows}(VtVpart23)-1} VtVpart23_o}{\sum_{o=0}^{\text{rows}(VtVpart23)-1} npart23_o}} \\ \sqrt{\frac{\sum_{o=0}^{\text{rows}(VtVpart34)-1} VtVpart34_o}{\sum_{o=0}^{\text{rows}(VtVpart34)-1} npart34_o}} \\ \sqrt{\frac{\sum_{o=0}^{\text{rows}(VtVpart24)-1} VtVpart24_o}{\sum_{o=0}^{\text{rows}(VtVpart24)-1} npart24_o}} \end{bmatrix} \quad \begin{bmatrix} \frac{\sum_{o=0}^{\text{rows}(ADpart23)-1} ADpart23_o}{\sum_{o=0}^{\text{rows}(VtVpart23)-1} Npart23_o} \\ \frac{\sum_{o=0}^{\text{rows}(ADpart34)-1} ADpart34_o}{\sum_{o=0}^{\text{rows}(VtVpart34)-1} Npart34_o} \\ \frac{\sum_{o=0}^{\text{rows}(ADpart24)-1} ADpart24_o}{\sum_{o=0}^{\text{rows}(VtVpart24)-1} Npart24_o} \end{bmatrix} \quad Bottom2 \quad Bottom3 \quad Bottom4 \quad \begin{bmatrix} D23 \\ D34 \\ D24 \end{bmatrix} \quad KKK2 \quad KKK3 \quad KKK4 \right]$$

$$\text{NonLinStep3} := \text{DepthsNonLin}(3)$$

$$\text{NonLinStep4DepthsNonLin}(4)$$

$$\text{NonLinStep5DepthsNonLin}(5)$$

$$\text{NonLinStep6DepthsNonLin}(6)$$

$$\text{NonLinStep7DepthsNonLin}(7)$$

$$\frac{B2}{B3} \quad \frac{B3}{B4} \quad \frac{B2}{B4}$$

$$\frac{B2}{B3} \quad \frac{B3}{B4} \quad \frac{B2}{B4}$$

$$\text{DepthNonLinStep3} := \left\langle \text{NonLinStep3}^{\langle 0 \rangle} \right\rangle_0 \qquad \text{RMSEsStep3} := \left\langle \text{NonLinStep3}^{\langle 1 \rangle} \right\rangle_0^{\mathrm{T}} \qquad \text{RMSEsStep3} = \begin{bmatrix} 0.65 & 0.69 & 0.65 \end{bmatrix} \quad \begin{bmatrix} 0.67 & 0.69 & 0.66 \end{bmatrix} \qquad \text{MADsStep3} := \left\langle \text{NonLinStep3}^{\langle 2 \rangle} \right\rangle_0^{\mathrm{T}} \qquad \text{MADsStep3} = \begin{bmatrix} 0.29 & 0.31 & 0.3 \end{bmatrix}$$

$$\text{BottomNonLin3} := \left\langle \text{NonLinStep3}^{\langle 3 \rangle} \right\rangle_0 \qquad \text{BottomNonLin3_B2} := \left\langle \text{NonLinStep3}^{\langle 3 \rangle} \right\rangle_0 \qquad \text{BottomNonLin3_B3} := \left\langle \text{NonLinStep3}^{\langle 4 \rangle} \right\rangle_0 \qquad \text{BottomNonLin3_B4} := \left\langle \text{NonLinStep3}^{\langle 5 \rangle} \right\rangle_0$$

$$\text{KNonLinB2} := \left\langle \text{NonLinStep3}^{\langle 7 \rangle} \right\rangle_0 \qquad \text{KNonLinB3} := \left\langle \text{NonLinStep3}^{\langle 8 \rangle} \right\rangle_0 \qquad \text{KNonLinB4} := \left\langle \text{NonLinStep3}^{\langle 9 \rangle} \right\rangle_0$$

KNonLinB2 KNonLinB3 KNonLinB4

$$\text{r2B2B3Step3} := \left\| \begin{array}{l} \text{SDB} \leftarrow \left(\left\langle \text{NonLinStep3}^{\langle 6 \rangle} \right\rangle_0 \right)_0 \\ \text{for } i \in 0 \,..\, \text{rows}(\text{SDB}) - 1 \\ \quad \text{for } j \in 0 \,..\, \text{cols}(\text{SDB}) - 1 \\ \quad\quad \text{if } \left(\text{SDB}_{i,j} \neq -9999 \right) \wedge \left(\text{Sounds2}_{i,j} \neq -9999 \right) \\ \quad\quad\quad \text{Out1}_k \leftarrow \text{SDB}_{i,j} \\ \quad\quad\quad \text{Out2}_k \leftarrow \text{Sounds2}_{i,j} \\ \quad\quad\quad k \leftarrow k + 1 \\ \text{corr}(\text{Out1}, \text{Out2})^2 \end{array} \right. = 0.99608$$

$$\text{r2B3B4Step3} := \left\| \begin{array}{l} \text{SDB} \leftarrow \left(\left\langle \text{NonLinStep3}^{\langle 6 \rangle} \right\rangle_0 \right)_1 \\ \text{for } i \in 0 \,..\, \text{rows}(\text{SDB}) - 1 \\ \quad \text{for } j \in 0 \,..\, \text{cols}(\text{SDB}) - 1 \\ \quad\quad \text{if } \left(\text{SDB}_{i,j} \neq -9999 \right) \wedge \left(\text{Sounds2}_{i,j} \neq -9999 \right) \\ \quad\quad\quad \text{Out1}_k \leftarrow \text{SDB}_{i,j} \\ \quad\quad\quad \text{Out2}_k \leftarrow \text{Sounds2}_{i,j} \\ \quad\quad\quad k \leftarrow k + 1 \\ \text{corr}(\text{Out1}, \text{Out2})^2 \end{array} \right. = 0.99564$$

$$\text{r2B2B4Step3} := \left\| \begin{array}{l} \text{SDB} \leftarrow \left(\left\langle \text{NonLinStep3}^{\langle 6 \rangle} \right\rangle_0 \right)_2 \\ \text{for } i \in 0 \,..\, \text{rows}(\text{SDB}) - 1 \\ \quad \text{for } j \in 0 \,..\, \text{cols}(\text{SDB}) - 1 \\ \quad\quad \text{if } \left(\text{SDB}_{i,j} \neq -9999 \right) \wedge \left(\text{Sounds2}_{i,j} \neq -9999 \right) \\ \quad\quad\quad \text{Out1}_k \leftarrow \text{SDB}_{i,j} \\ \quad\quad\quad \text{Out2}_k \leftarrow \text{Sounds2}_{i,j} \\ \quad\quad\quad k \leftarrow k + 1 \\ \text{corr}(\text{Out1}, \text{Out2})^2 \end{array} \right. = 0.99613$$

r2B2B3Lin :=
SDB ← ((DepthsLin(178)^⟨0⟩))_0
for i ∈ 0 .. rows(SDB) − 1
  for j ∈ 0 .. cols(SDB) − 1
    if (SDB_{i,j} ≠ −9999) ∧ (Sounds2_{i,j} ≠ −9999)
      Out1_k ← SDB_{i,j}
      Out2_k ← Sounds2_{i,j}
      k ← k + 1
corr(Out1, Out2)^2
= 0.37442

r2B3B4Lin :=
SDB ← ((DepthsLin(178)^⟨0⟩))_1
for i ∈ 0 .. rows(SDB) − 1
  for j ∈ 0 .. cols(SDB) − 1
    if (SDB_{i,j} ≠ −9999) ∧ (Sounds2_{i,j} ≠ −9999)
      Out1_k ← SDB_{i,j}
      Out2_k ← Sounds2_{i,j}
      k ← k + 1
corr(Out1, Out2)^2
= 0.16221

r2B2B4Lin :=
SDB ← ((DepthsLin(178)^⟨0⟩))_2
for i ∈ 0 .. rows(SDB) − 1
  for j ∈ 0 .. cols(SDB) − 1
    if (SDB_{i,j} ≠ −9999) ∧ (Sounds2_{i,j} ≠ −9999)
      Out1_k ← SDB_{i,j}
      Out2_k ← Sounds2_{i,j}
      k ← k + 1
corr(Out1, Out2)^2
= 0.01353

DepthsLin(178)

Depthslin := (DepthsLin(178)^⟨0⟩)_0

BottomPasso1 := READEXCEL("BottomNonLinPasso1.xlsx")

DepthsPasso1 := READEXCEL("DepthsNonLinearPasso1.xlsx")

DepthsNonLin

[DepthsL23; DepthsL34; DepthsL24] :=
D23 ← Depthslin_0
D34 ← Depthslin_1
D24 ← Depthslin_2
for i ∈ 0 .. rows(Sounds2) − 1
  for j ∈ 0 .. cols(Sounds2) − 1
    if 30 > Sounds2_{i,j} > 0 ∧ im3B2W_{i,j} ≠ −9999
      Out23_k ← D23_{i,j}
      Out34_k ← D34_{i,j}
      Out24_k ← D24_{i,j}
      k ← k + 1
    else
      continue
[Out23; Out34; Out24]

RMSENonLin := concat(num2str(round(RMSEsStep3_{0,0}, 2)), " m") = "0.65 m"

RMSETradLin := concat(num2str(round(RMSElin_0, 2)), " m") = ?

DepthsNL :=
D ← DepthNonLinStep3
for i ∈ 0 .. rows(Sounds2) − 1
  for j ∈ 0 .. cols(Sounds2) − 1
    if 30 > Sounds2_{i,j} > 0 ∧ im3B2W_{i,j} ≠ −9999
      Out_k ← D_{i,j}
      k ← k + 1
    else
      continue
Out

DepthsNonLin := ((DepthsNonLin(3)^⟨6⟩))_0 = [178 × 178; 178 × 178; 178 × 178]

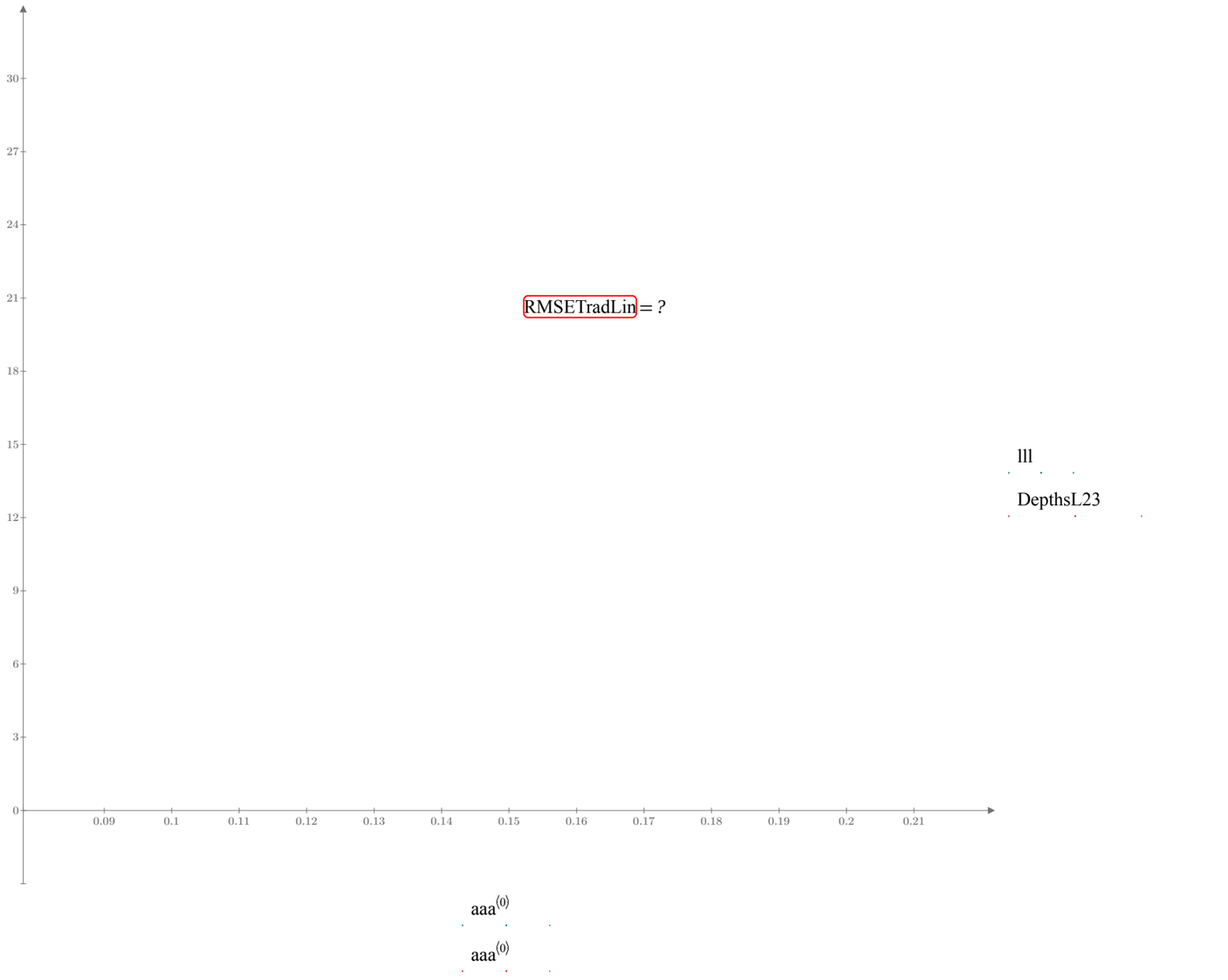
[DepthsNL23; DepthsNL34; DepthsNL24] :=
D23 ← DepthsNLin_0
D34 ← DepthsNLin_1
D24 ← DepthsNLin_2
for i ∈ 0 .. rows(Sounds2) − 1
  for j ∈ 0 .. cols(Sounds2) − 1
    if 30 > Sounds2_{i,j} > 0 ∧ im3B2W_{i,j} ≠ −9999
      Out23_k ← D23_{i,j}
      Out34_k ← D34_{i,j}
      Out24_k ← D24_{i,j}
      k ← k + 1
    else
      continue
[Out23; Out34; Out24]

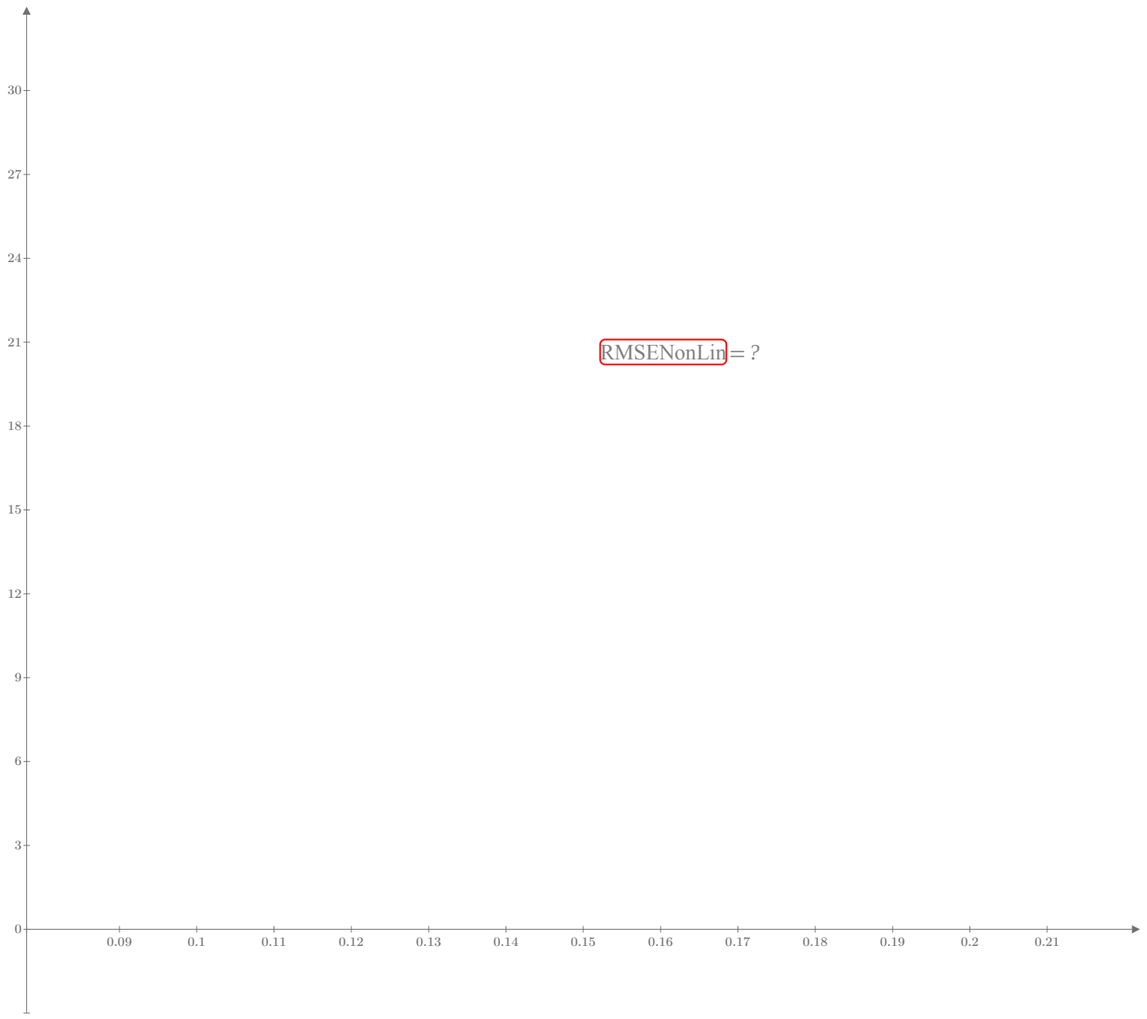
DepthsNonLin(step)

DepthsLin(3) = [[3 × 1] [3 × 1] [3 × 1]]

[0.206 1]

$$aaa = \begin{bmatrix} 0.206 & 1 \\ 0.206 & 1 \\ 0.206 & 1 \\ 0.206 & 1 \\ 0.206 & 1 \\ 0.205 & 1 \\ 0.205 & 1 \\ 0.205 & 1 \\ 0.205 & 1 \\ 0.206 & 1 \\ 0.206 & 1 \\ & \vdots \end{bmatrix} \quad aaa^{\langle 0 \rangle} = \begin{bmatrix} 0.206 \\ 0.206 \\ 0.206 \\ 0.206 \\ 0.206 \\ 0.206 \\ 0.205 \\ 0.205 \\ 0.205 \\ 0.205 \\ 0.206 \\ 0.206 \\ \vdots \end{bmatrix} \quad (\text{DepthsL}) = ? \quad DDD - DepthsL23 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \vdots \end{bmatrix} \quad DDD = \begin{bmatrix} \vdots \\ 19.316 \\ 19.105 \\ 18.789 \\ 18.076 \\ 18.09 \\ 17.965 \\ 18.468 \\ 18.971 \\ 19.529 \\ 19.716 \\ 19.798 \\ 19.366 \\ \vdots \end{bmatrix} \quad DepthsL23 = \begin{bmatrix} \vdots \\ 19.316 \\ 19.105 \\ 18.789 \\ 18.076 \\ 18.09 \\ 17.965 \\ 18.468 \\ 18.971 \\ 19.529 \\ 19.716 \\ 19.798 \\ 19.366 \\ \vdots \end{bmatrix}$$





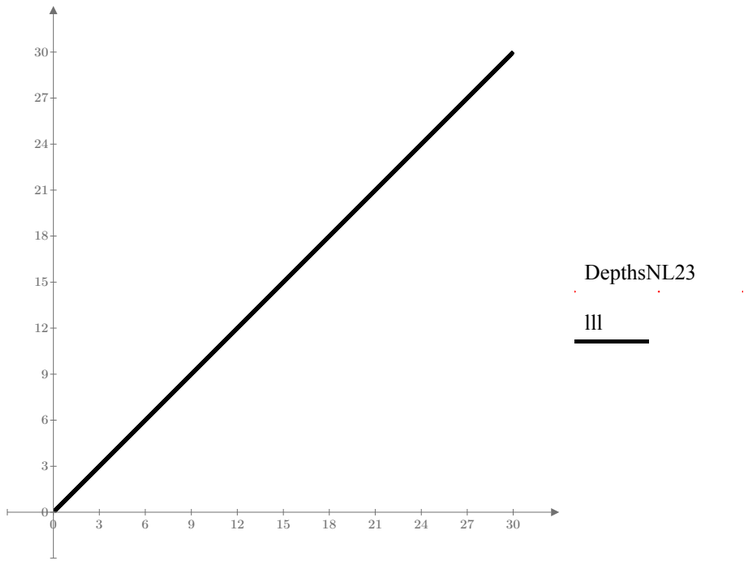
lll

DepthsNL

$aaa^{\langle 0 \rangle}$

$aaa^{\langle 0 \rangle}$

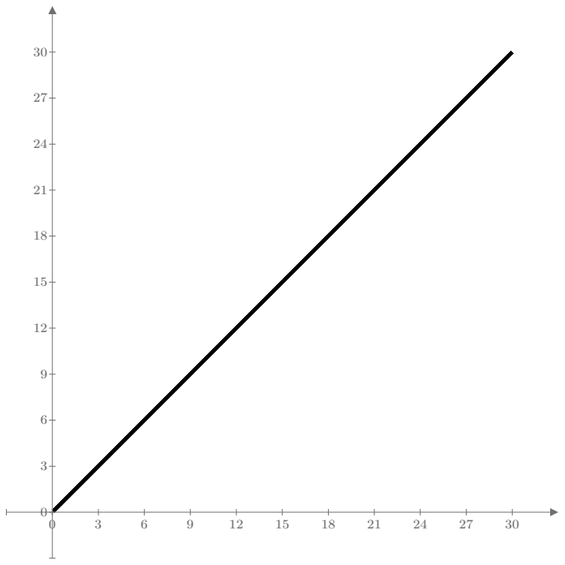
$$\text{Ident} := \left\| \begin{array}{l} \text{for } i \in 0 \,..\, \text{rows}(\text{lll}) - 1 \\ \quad \left\| \begin{array}{l} \text{Out}_i \leftarrow 1 \end{array} \right. \\ \text{Out} \end{array} \right.$$



DepthsNL23

lll

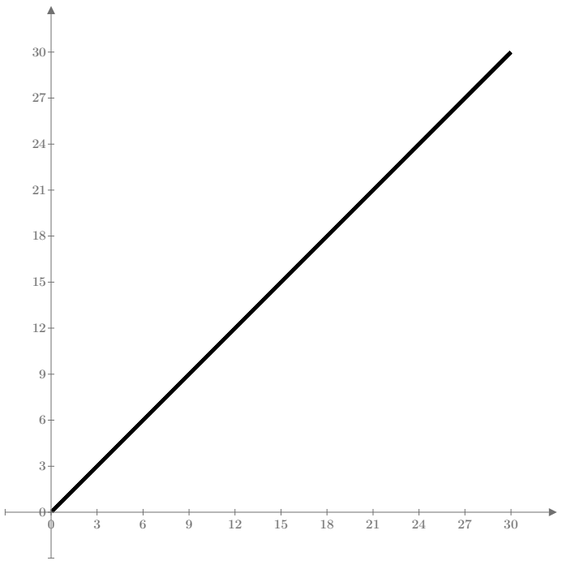
lll



DepthsNL34

lll

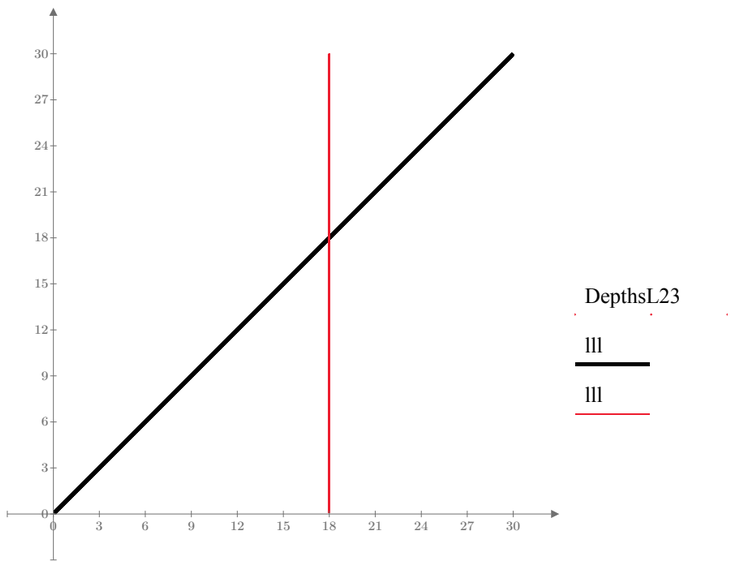
lll



DepthsNL24

lll

lll

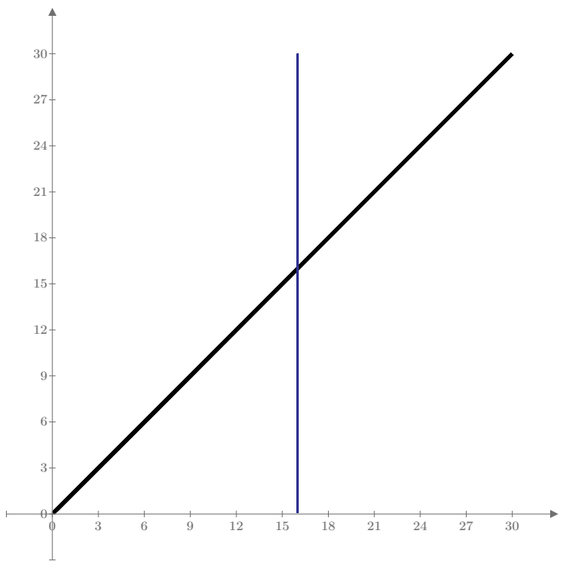


DepthsL23

lll

lll

lll

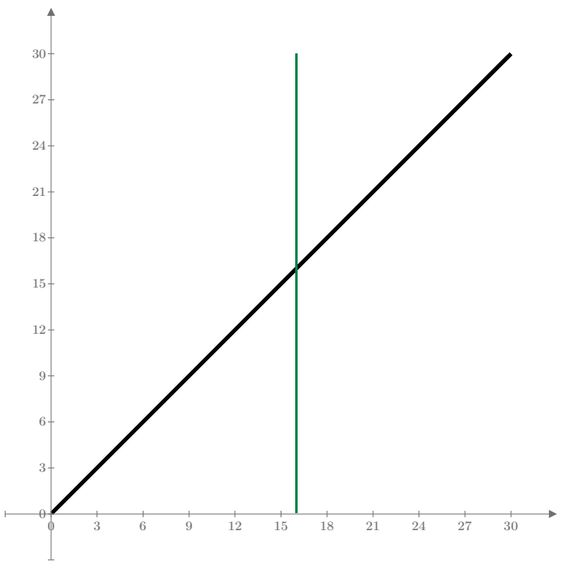


DepthsL34

lll

lll

lll



DepthsL24

lll

lll

lll

lll

18 • Ident

lll

16 • Ident

lll

16 • Ident

CWD := “C:\PhD\2015\00 Papers\Paper 3 - Simeonof Island\Mathcad Outputs\”

DepthsNL3 :=
DF ← DepthNonLinStep3
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 -9999”
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

a ← “NonLinSDBPasso100Step3.txt”
WRITETEXT(a, DepthsNL3) =

$$\begin{bmatrix} \text{“ncols 178”} \\ \text{“nrows 178”} \\ \text{“xllcorner 476010”} \\ \vdots \end{bmatrix}$$

KNonLinB2 KNonLinB3 KNonLinB4

```
K_B2 := for i ∈ 0 .. rows(KNonLinB2) − 1
          for j ∈ 0 .. cols(KNonLinB2) − 1
            t2 ← KNonLinB2[i,j] > −9999
            t3 ← KNonLinB3[i,j] > −9999
            t4 ← KNonLinB4[i,j] > −9999
            if (t2 = 1) ∧ (t3 = 1) ∧ (t4 = 1)
              A[i,j] ← KNonLinB2[i,j]
            else
              A[i,j] ← −9999
          A
```

```
K_B3 := for i ∈ 0 .. rows(KNonLinB3) − 1
          for j ∈ 0 .. cols(KNonLinB3) − 1
            t2 ← KNonLinB2[i,j] > −9999
            t3 ← KNonLinB3[i,j] > −9999
            t4 ← KNonLinB4[i,j] > −9999
            if (t2 = 1) ∧ (t3 = 1) ∧ (t4 = 1)
              A[cont,0] ← KNonLinB3[i,j]
              cont ← cont + 1
          A
```

```
K_B4 := for i ∈ 0 .. rows(KNonLinB4) − 1
          for j ∈ 0 .. cols(KNonLinB4) − 1
            t2 ← KNonLinB2[i,j] > −9999
            t3 ← KNonLinB3[i,j] > −9999
            t4 ← KNonLinB4[i,j] > −9999
            if (t2 = 1) ∧ (t3 = 1) ∧ (t4 = 1)
              A[cont,0] ← KNonLinB4[i,j]
              cont ← cont + 1
          A
```

```
K_B2 := DF ← KNonLinB2
        Out0 ← concat("ncols ", num2str(cols(DF)))
        Out1 ← concat("nrows ", num2str(rows(DF)))
        Out2 ← concat("xllcorner ", num2str(UtmXinf))
        Out3 ← concat("yllcorner ", num2str(UtmYinf))
        Out4 ← "cellsize 30"
        Out5 ← "nodata_value -9999"
        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
```

```
K_B3 := DF ← KNonLinB3
        Out0 ← concat("ncols ", num2str(cols(DF)))
        Out1 ← concat("nrows ", num2str(rows(DF)))
        Out2 ← concat("xllcorner ", num2str(UtmXinf))
        Out3 ← concat("yllcorner ", num2str(UtmYinf))
        Out4 ← "cellsize 30"
        Out5 ← "nodata_value -9999"
        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
```

```
a ← "NonLinKPasso100Step6B2_2017.txt"
WRITETEXT(a, K_B2) =
```

$$\begin{bmatrix} \text{"ncols 178"} \\ \text{"nrows 178"} \\ \text{"xllcorner 476010"} \\ \vdots \end{bmatrix}$$

```
B_B2 :=
for i ∈ 0 .. rows(BottomNonLin3_B2) − 1
  for j ∈ 0 .. cols(BottomNonLin3_B2) − 1
    t2 ← BottomNonLin3_B2[i,j] > −9999
    t3 ← BottomNonLin3_B3[i,j] > −9999
    t4 ← BottomNonLin3_B4[i,j] > −9999
    if (t2 = 1) ∧ (t3 = 1) ∧ (t4 = 1)
      A[cont,0] ← BottomNonLin3_B2[i,j]
      cont ← cont + 1
A
```

```
B_B3 :=
for i ∈ 0 .. rows(BottomNonLin3_B3) − 1
  for j ∈ 0 .. cols(BottomNonLin3_B3) − 1
    t2 ← BottomNonLin3_B2[i,j] > −9999
    t3 ← BottomNonLin3_B3[i,j] > −9999
    t4 ← BottomNonLin3_B4[i,j] > −9999
    if (t2 = 1) ∧ (t3 = 1) ∧ (t4 = 1)
      A[cont,0] ← BottomNonLin3_B3[i,j]
      cont ← cont + 1
A
```

```
B_B4 :=
for i ∈ 0 .. rows(BottomNonLin3_B4) − 1
  for j ∈ 0 .. cols(BottomNonLin3_B4) − 1
    t2 ← BottomNonLin3_B2[i,j] > −9999
    t3 ← BottomNonLin3_B3[i,j] > −9999
    t4 ← BottomNonLin3_B4[i,j] > −9999
    if (t2 = 1) ∧ (t3 = 1) ∧ (t4 = 1)
      A[cont,0] ← BottomNonLin3_B4[i,j]
      cont ← cont + 1
A
```

BottomNonLin3_B2 := (DepthsNonLin(3)^⟨3⟩)_0 BottomNonLin3_B3 := (DepthsNonLin(3)^⟨4⟩)_0 BottomNonLin3_B4 := (DepthsNonLin(3)^⟨5⟩)_0

BottomNonLin4_B2 (DepthsNonLin(4)^⟨3⟩)_0 BottomNonLin4_B3 (DepthsNonLin(4)^⟨4⟩)_0 BottomNonLin4_B4 (DepthsNonLin(4)^⟨5⟩)_0

BottomNonLin5_B2 (DepthsNonLin(5)^⟨3⟩)_0 BottomNonLin5_B3 (DepthsNonLin(5)^⟨4⟩)_0 BottomNonLin5_B4 (DepthsNonLin(5)^⟨5⟩)_0

BottomNonLin6_B2 (DepthsNonLin(6)^⟨3⟩)_0 BottomNonLin6_B3 (DepthsNonLin(6)^⟨4⟩)_0 BottomNonLin6_B4 (DepthsNonLin(6)^⟨5⟩)_0

BottomNonLin7_B2 (DepthsNonLin(7)^⟨3⟩)_0 BottomNonLin7_B3 (DepthsNonLin(7)^⟨4⟩)_0 BottomNonLin7_B4 (DepthsNonLin(7)^⟨5⟩)_0

```
BottomNL3_B2 :=
for i ∈ 0 .. rows(BottomNonLin3_B2) − 1
  for j ∈ 0 .. cols(BottomNonLin3_B2) − 1
    t2 ← BottomNonLin3_B2[i,j] > −9999
    t3 ← BottomNonLin3_B3[i,j] > −9999
    t4 ← BottomNonLin3_B4[i,j] > −9999
    if (t2 = 1) ∧ (t3 = 1) ∧ (t4 = 1)
      A[i,j] ← BottomNonLin3_B2[i,j]
    else
      A[i,j] ← −9999
A
```

```
BottomNL3_B3 :=
for i ∈ 0 .. rows(BottomNonLin3_B3) − 1
  for j ∈ 0 .. cols(BottomNonLin3_B3) − 1
    t2 ← BottomNonLin3_B2[i,j] > −9999
    t3 ← BottomNonLin3_B3[i,j] > −9999
    t4 ← BottomNonLin3_B4[i,j] > −9999
    if (t2 = 1) ∧ (t3 = 1) ∧ (t4 = 1)
      A[i,j] ← BottomNonLin3_B3[i,j]
    else
      A[i,j] ← −9999
A
```

```
BottomNL3_B4 :=
for i ∈ 0 .. rows(BottomNonLin3_B4) − 1
  for j ∈ 0 .. cols(BottomNonLin3_B4) − 1
    t2 ← BottomNonLin3_B2[i,j] > −9999
    t3 ← BottomNonLin3_B3[i,j] > −9999
    t4 ← BottomNonLin3_B4[i,j] > −9999
    if (t2 = 1) ∧ (t3 = 1) ∧ (t4 = 1)
      A[i,j] ← BottomNonLin3_B4[i,j]
    else
      A[i,j] ← −9999
A
```

```
Bottom2 := | DF ← BottomNL3_B2
           | 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 -9999"
           | 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

| a ← "NonLinBottomPasso100Step3B2.txt" | = | "ncols 178"
| WRITETEXT(a, Bottom2)                 |   | "nrows 178"
                                          | "xllcorner 476010"
                                          | ⋮
```

```
Bottom3 := | DF ← BottomNL3_B3
           | 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 -9999"
           | 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

| a ← "NonLinBottomPasso100Step3B3.txt" | = | "ncols 178"
| WRITETEXT(a, Bottom3)                 |   | "nrows 178"
                                          | "xllcorner 476010"
                                          | ⋮
```

```
Bottom4 := | DF ← BottomNL3_B4
           | 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 -9999"
           | 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

| a ← "NonLinBottomPasso100Step3B4.txt" | = | "ncols 178"
| WRITETEXT(a, Bottom4)                 |   | "nrows 178"
                                          | "xllcorner 476010"
                                          | ⋮
```