IAPSO MIO

Dominique Lefèvre (CNRS), dominique.lefevre@mio.osupytheas.fr
Anthony Bosse (Aix-Marseille Université), anthony.bosse@mio.osupytheas.fr

CTD and oxygen adjustment procedure

The general qualification of the CTD data follows the following workflow, where the final .cnv and .btl files contain adjusted salinity and dissolved oxygen data relative to the reference autosal and winkler data.

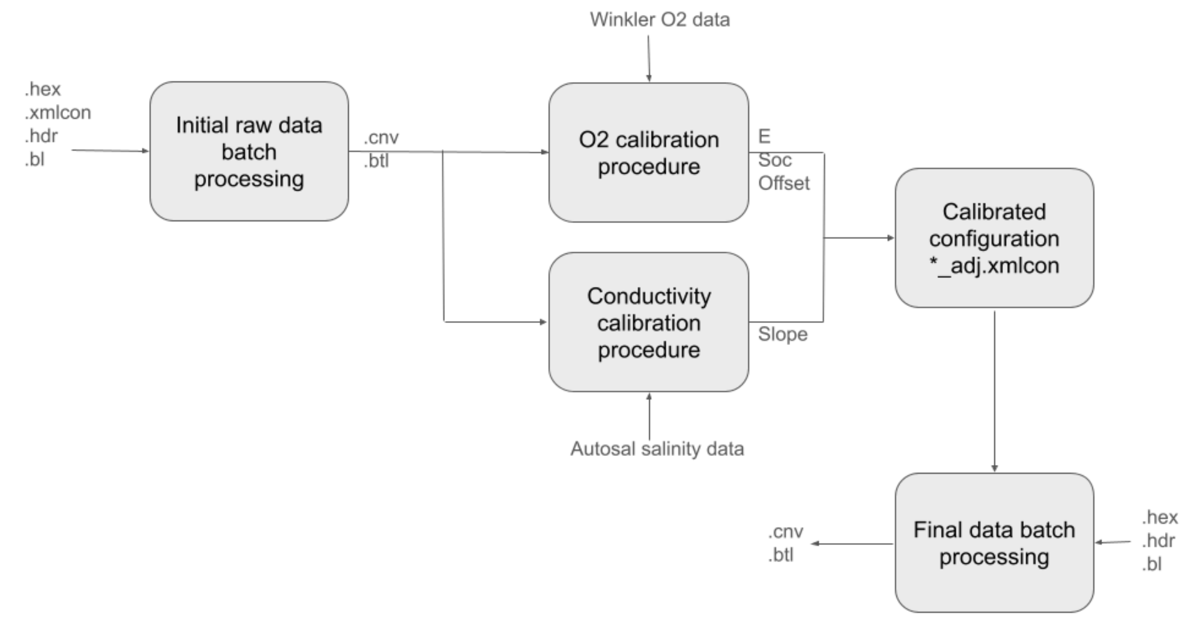


Figure 1: General workflow for the CTDO2 adjustments in delayed mode.

Seabird Processing

Folder architecture: a working directory is setup with the following folders:

SBE-profile-data/raw (containing raw data *.hex, *.bl, *.hdr)
SBE-profile-data/xmlcon (containing original config files *.xmlcon)
SBE-profile-data/cnv (where processed *.cnv files will be stored)
SBE-profile-data/btl (where *.btl will be stored)

SBE_Pprocess/ (with a batch executive file will check for the presence of a cnv file in /cnv, will run “sbebatch” if not, until all files are processed)
SBE_Process/psa_doubleCT (containing the *.psa configuration files for the data processing including traitementctd.txt which gives the order of steps to follow)

Exemple of the different steps done by SBE processing:

datcnv /i..\SBE-profile-data\raw\%1.hex /c..\SBE-profile-data\xmlcon_adj\%1.XMLCON /o..\SBE-profile-data\cnv_adj /ppsa_doubleCT\datcnv.psa /f%1
wildedit /i..\SBE-profile-data\cnv_adj\%1.cnv /o..\SBE-profile-data\cnv_adj /ppsa_doubleCT\wildedit.psa /f%1
wildedit /i..\SBE-profile-data\cnv_adj\%1.cnv /o..\SBE-profile-data\cnv_adj /ppsa_doubleCT\wildedit.psa /f%1
filter /i..\SBE-profile-data\cnv_adj\%1.cnv /o..\SBE-profile-data\cnv_adj /ppsa_doubleCT\filter.psa /f%1
alignctd /i..\SBE-profile-data\cnv_adj\%1.cnv /o..\SBE-profile-data\cnv_adj /ppsa_doubleCT\alignctd.psa /f%1
celltm /i..\SBE-profile-data\cnv_adj\%1.cnv /o..\SBE-profile-data\cnv_adj /ppsa_doubleCT\cellctm.psa /f%1
loopedit /i..\SBE-profile-data\cnv_adj\%1.cnv /o..\SBE-profile-data\cnv_adj /ppsa_doubleCT\loopedit.psa /f%1
derive /i..\SBE-profile-data\cnv_adj\%1.cnv /c..\SBE-profile-data\xmlcon_adj\%1.XMLCON /o..\SBE-profile-data\cnv_adj /ppsa_doubleCT\derive.psa /f%1
binavg /i..\SBE-profile-data\cnv_adj\%1.cnv /o..\SBE-profile-data\cnv_adj /ppsa_doubleCT\binavg.psa /f%1
bottlesum /i..\SBE-profile-data\cnv_adj\%1.ros /c..\SBE-profile-data\xmlcon_adj\%1.XMLCON /o..\SBE-profile-data\btl_adj /ppsa_doubleCT\bottlesum.psa /f%1

The following configuration for each psa is used:

datcnv.psa
datcnv_skipover = 0
datcnv_ox_hysteresis_correction = yes

wildedit.psa
wildedit_pass1_nstd = 2.0
wildedit_pass2_nstd = 10.0
wildedit_pass2_mindelta = 0.000e+000
wildedit_npoint = 100
wildedit_vars = prDM t090C t190C c0S/m c1S/m
wildedit_excl_bad_scans = yes

filter.psa
filter_low_pass_tc_A = 0.030
filter_low_pass_tc_B = 0.150
filter_low_pass_A_vars = turbWETntu0 spar wetStar
filter_low_pass_B_vars = prDM

alignctd.psa
alignctd_adv = sbeox0V 2.000

celltm.psa
celltm_alpha = 0.0300, 0.0300
celltm_tau = 7.0000, 7.0000
celltm_temp_sensor_use_for_cond = primary, secondary

loopedit.psa
loopedit_minVelocity = 0.100
loopedit_surfaceSoak: minDepth = 6.0, maxDepth = 25, useDeckPress = 1
loopedit_excl_bad_scans = yes

derive.psa
derive_time_window_docdt = seconds: 2
derive_ox_tau_correction = yes

binavg.psa
binavg_bintype = decibars
binavg_binsize = 1
binavg_excl_bad_scans = yes
binavg_skipover = 0
binavg_omit = 0

To ease the work, “*.btl” files are merged into one single file using a R script, from which adjustments can be made.

Cf annexe

1. Raw data batch processing

As shown by the general workflow, the first step consists in a batch processing of the raw CTD files, producing .cnv and .blt files.

2. Conductivity-Salinity validation using Salinity bottles

The conductivity correction is deduced from the following steps (see also AUTOSAL_comparison.ipynb).

First, Autosal salinities are plotted against bottle salinities and the presence of outliers stands out clearly (Figure 1).

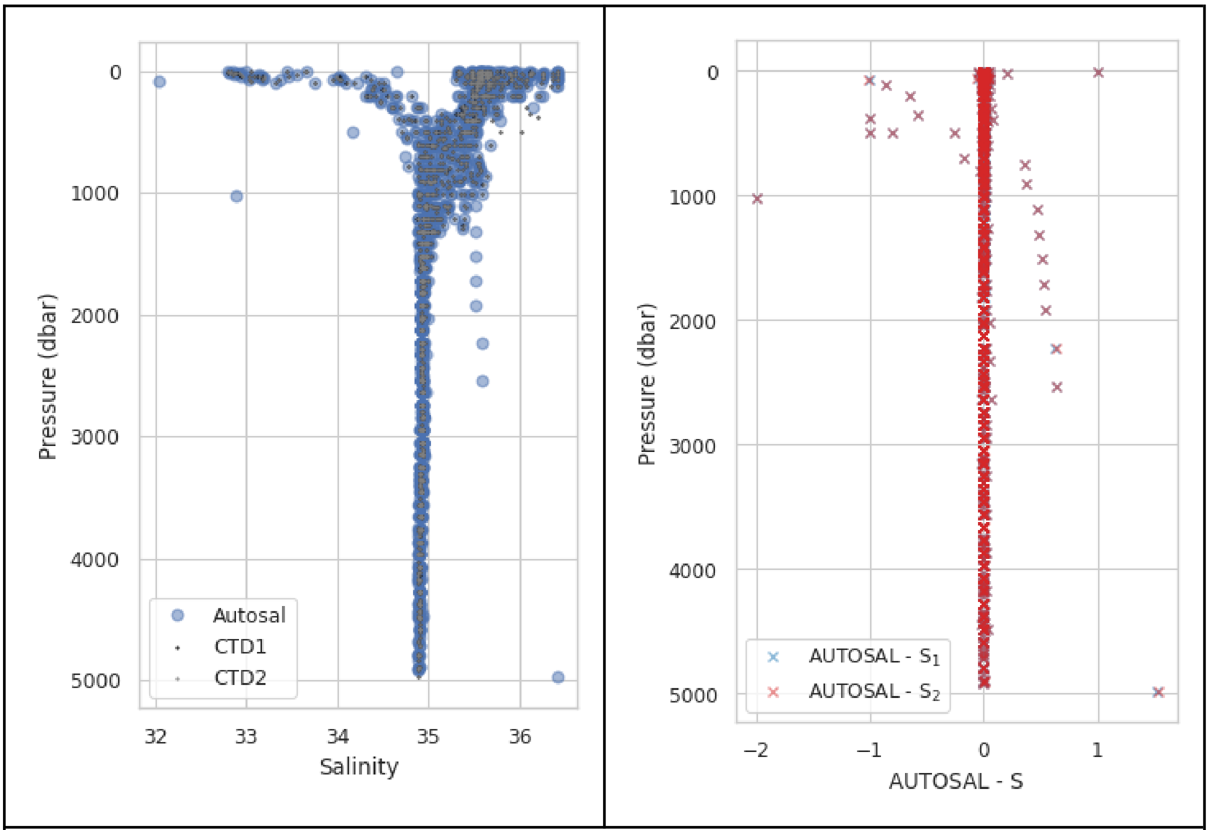


Figure 2: (left) Autosal and bottle salinities versus pressure. (right) Difference of Autosal and bottle salinities against pressure for the two CTD sensors.

A subsample of the salinities to be used is determined by using a standard Interquartile range (IQR) method excluding data outside the interval [Q1-1.5*(Q3-Q1) ; Q3+1.5*(Q3-Q1)] with Q1 and Q3 the 25th and 75th percentile of the data distribution. This reduces the number of samples from 1153 to 1003 and 986, for CTD1 ad CTD2 respectively (Figure 3-left).

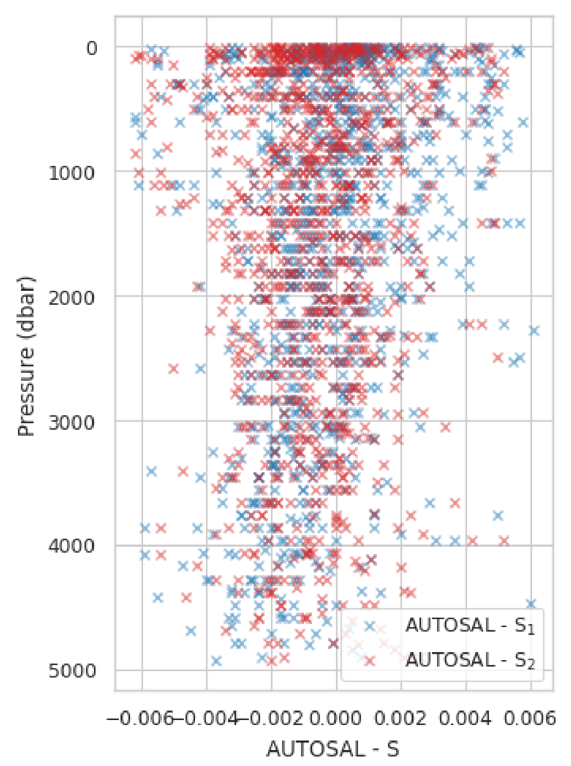


Figure 3: (left) Difference of Autosal and bottle salinities against pressure for the two CTD sensors, after filtering for outliers.

The distribution of differences can be plotted and examined. It shows small medians of -0.003 and -0.006 psu respectively with CTD1 and CTD2 (Figure 4).

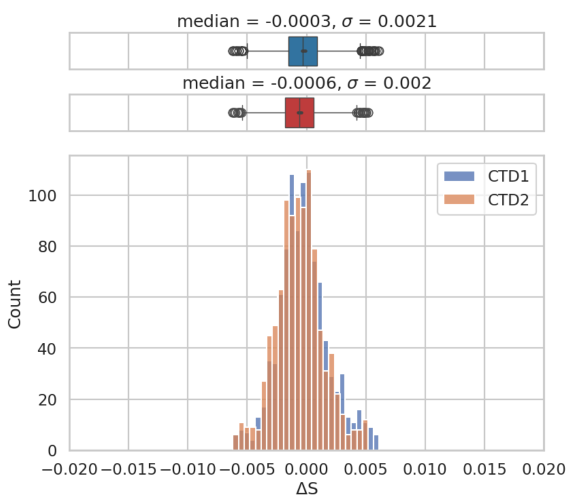


Figure 4: Histogram of the difference between Autosal and bottle salinities before correction.

Note that there seems to be a shift in the distribution with pressure, especially for CTD1. The order of magnitude of this shift is not larger than about 0.001 psu per 1000 dbar, which remains small compared to the variability of the signal. The correction applied here does not correct for any pressure dependance. This might have to be corrected in a second step in order to get more accurate salinity data below 2000 dbar.

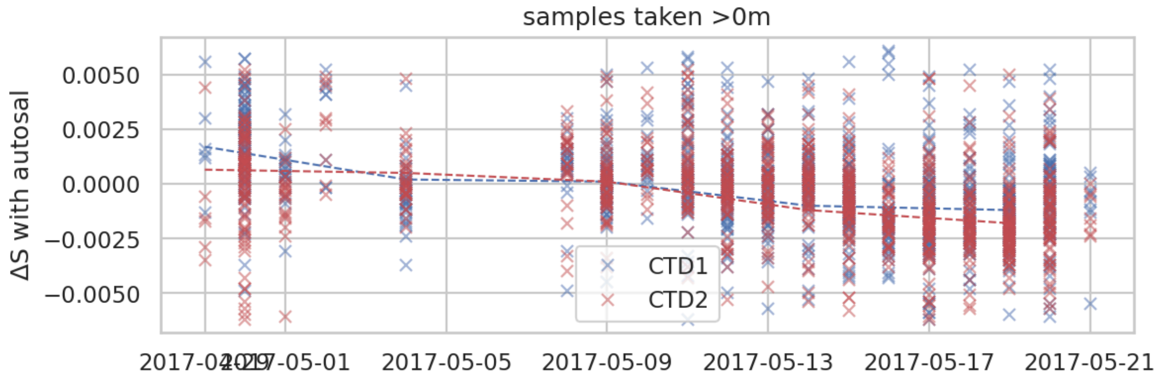


Figure 5: Autosal versus bottle salinity difference as a function of time. The dashed lines show the rolling median over a 5-day window.

Salinity differences are also plotted against time, to check that no temporal drift was present. There is a general decrease in salinity difference of about -0.003psu in the 3 weeks of the sampling.

The next step is the determination of the slope coefficient. For a set of samples, the slope coefficient is calculated as (https://www.seabird.com/cms-portals/seabird_com/cms/documents/training/Module10_DataAccuracyFieldCals.pdf):

$$slope = \frac{\sum_{i=1}^{n} \alpha_i \beta_i}{\sum_{i=1}^{n} \alpha_i \alpha_i}$$

where : n = number of samples, alpha = CTD conductivity, beta = true (bottle sample) conductivity

A general coefficient is calculated for each CTD. In our case, it gives:
Slope correction for CTD1 | CTD2 : 0.999995 | 0.999986

The first option is to set those slope coefficients constant for CTD1 and CTD2 in order to correct the whole data set. But, as shown by Figure 5, the salinity difference seems to vary with time. So we decided to retain a time dependence in the slope coefficients (Figure 6). This means that for each CTD cast, a slope correction coefficient will be interpolated from the slope/time relationship defined by the 5-day rolling median signal of Figure 6.

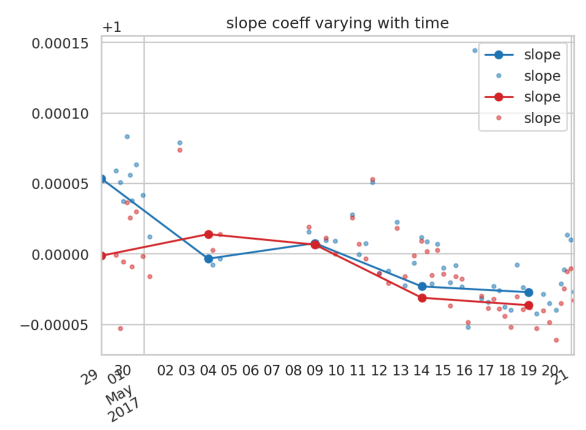


Figure 6: Slope coefficient for the two CTDs according to time. The scattered dots represent slope coefficient computed for each single cast. A 5-day rolling median is then applied to deduce a time-varying slope correction coefficient.

Before reprocessing the whole dataset, conductivity is recalculated using the inferred slope coefficient. The results show better with a median value centered (0.00009/0.00008psu, with standard deviation of 0.0019psu) and no more time variation in salinity difference as expected. (Figure 7 and 8)

Note that the initial difference in salinity was already before correction within the range of precision of the CTD instrument (0.0001 +/- 0.002 psu) and the present correction does not massively increase the precision of the final data set in terms of absolute salinity values.

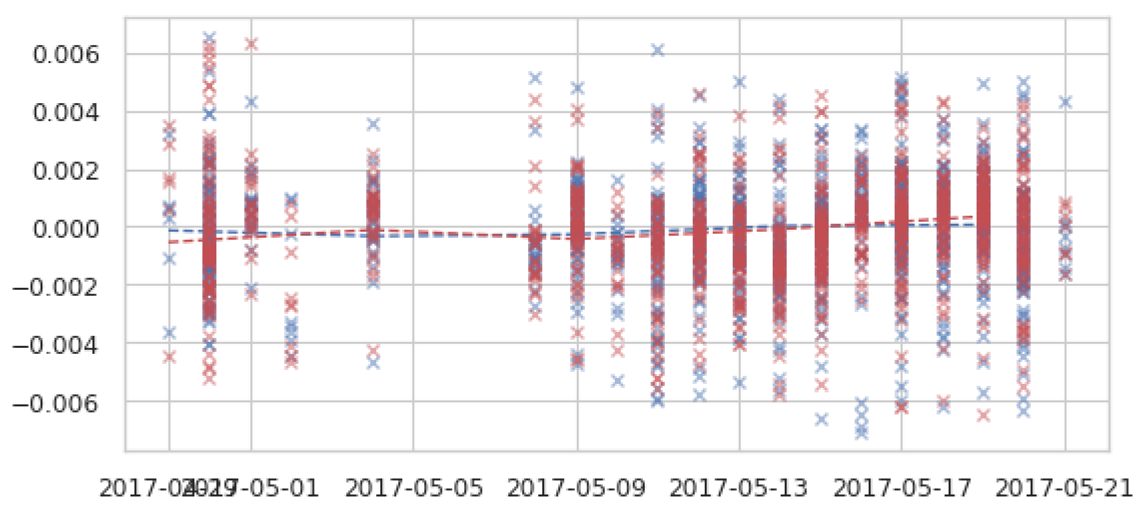


Figure 5: Autosal versus bottle salinity difference as a function of time after correction with a time-varying slope coefficient. The dashed lines show the rolling median over a 5-day window.

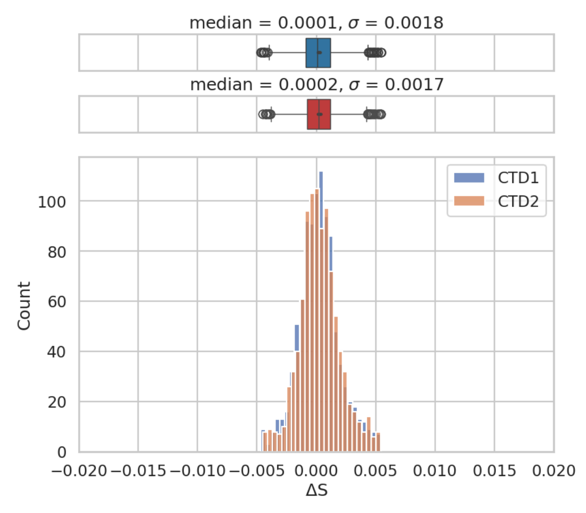


Figure 4: Histogram of the difference between Autosal and bottle salinities after correction with a time-varying slope coefficient.

3. Oxygen SBE43 configuration adjustment using discrete Winkler samples.

From the merged file (BTL Winkler and SBE43), use the Seabird application note 64.2 (http://ftp.seabird.com/application_notes/AN64-2.htm) to adjust SOC, Offset and E parameter in the following equation:

Based on Seabird algorithm

OX = Soc * [V + Voffset] * OxSOL(Theta,S) *

(1.0 + A*T90C + B* T90C ^2 + C* T90C ^3) * exp(E*P / K)

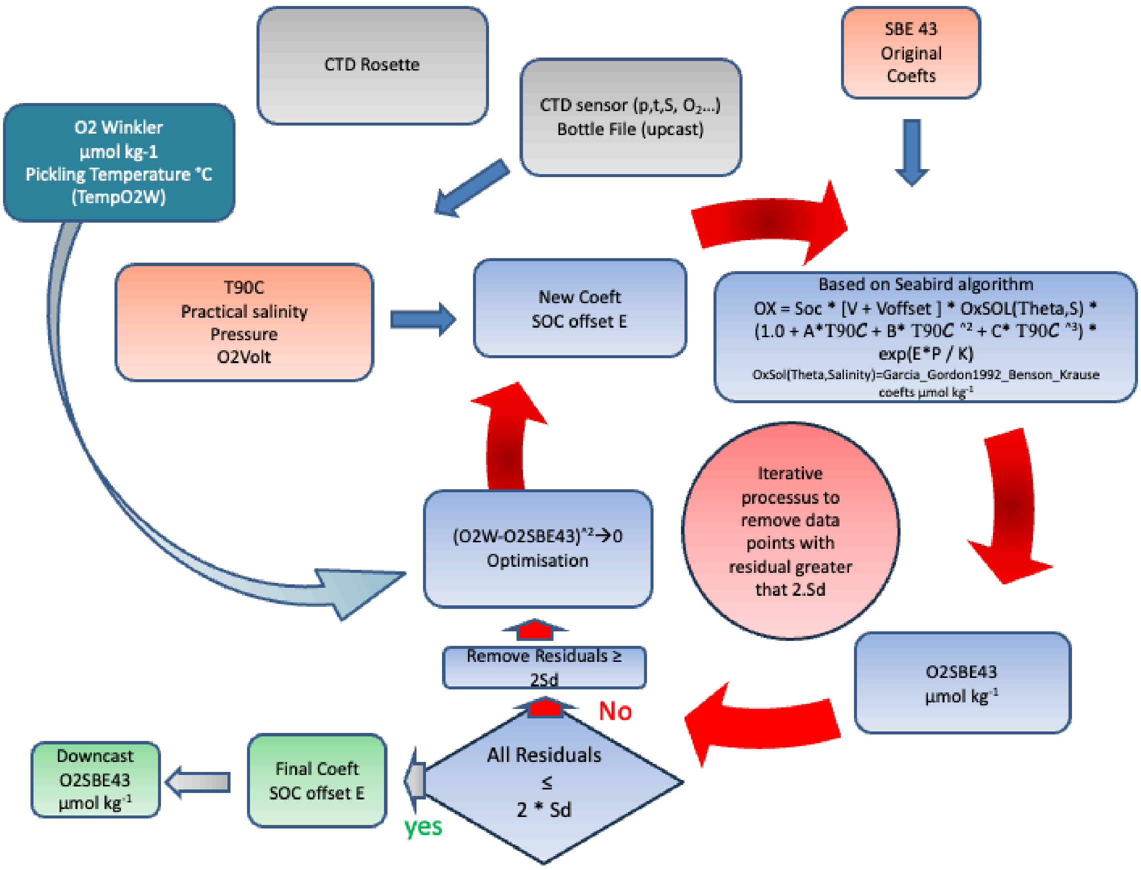
Ox, Oxygen concentration in µmol kg-1
P : Pressure in db
T90, in situ temperature °C
Theta, potential temperature °C
S, salinity
V, SBE43 O2 voltage in volt
OxSol(Theta,Salinity)=Garcia_Gordon1992_Benson_Krause coefts µmol kg^{-1}

*It is to be noticed that there is a mixed usage of in situ and potential temperature in this equation !

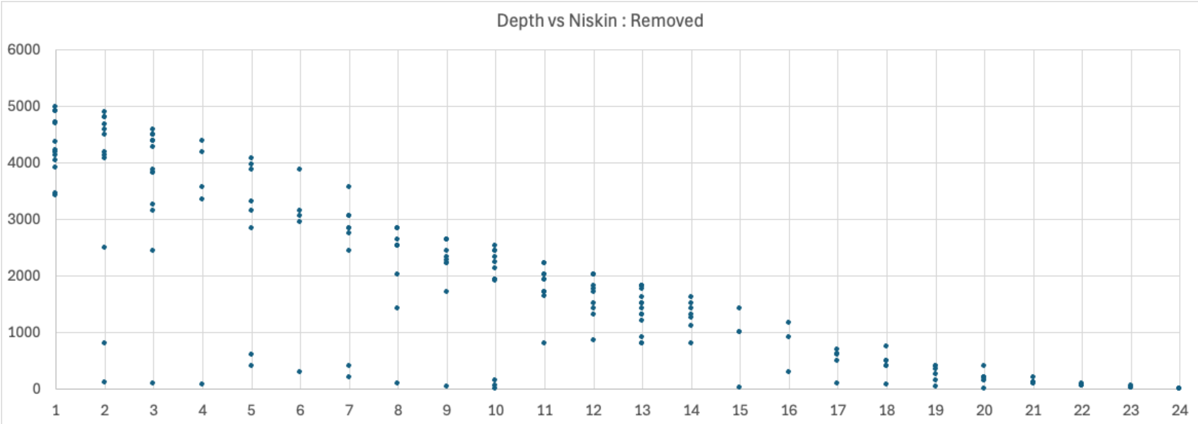
Adjust SBE_Oxygen value (O2_SBE43) is calculated based on this equation for each sample where there is a O2_Winkler sample (O2_W)
The sum of the square distance between each O2_W and O2_SBE_43 is computed.
This sum is then minimised using the evolutionary Excel ® solver function by adjusting the SOC, Offset and E parameters of the Seabird ® equation
For each sample, the value of the distance between the O2W and O2SBE43 is computed and any distance greater that +/-2*Sd is removed.
The solver function is run again until there are no more distance value greater than +/-2*Sd.
It can take several iterations, in this case it took 10 iterations.

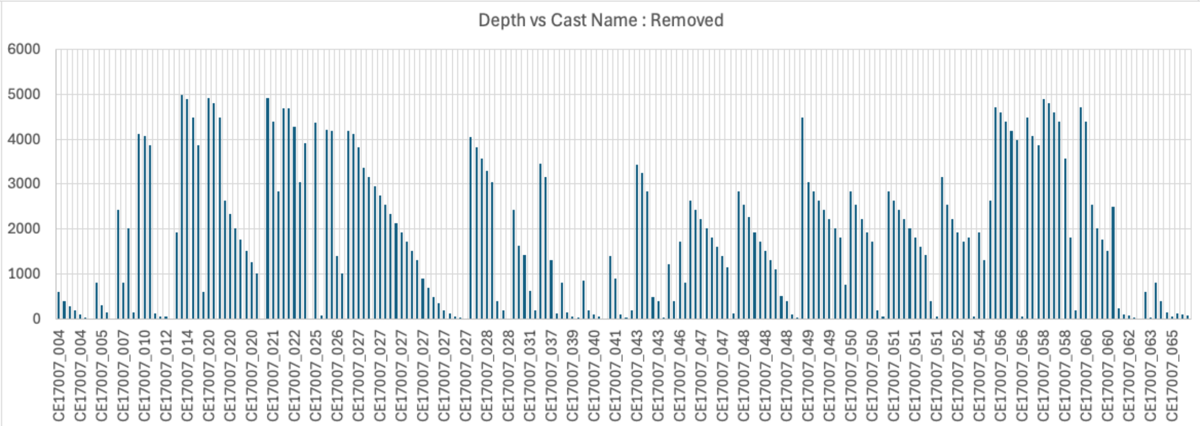
The adjusted SOC, Offset and E value are then used in the xlmcon file to generate the new adjusted profiles.

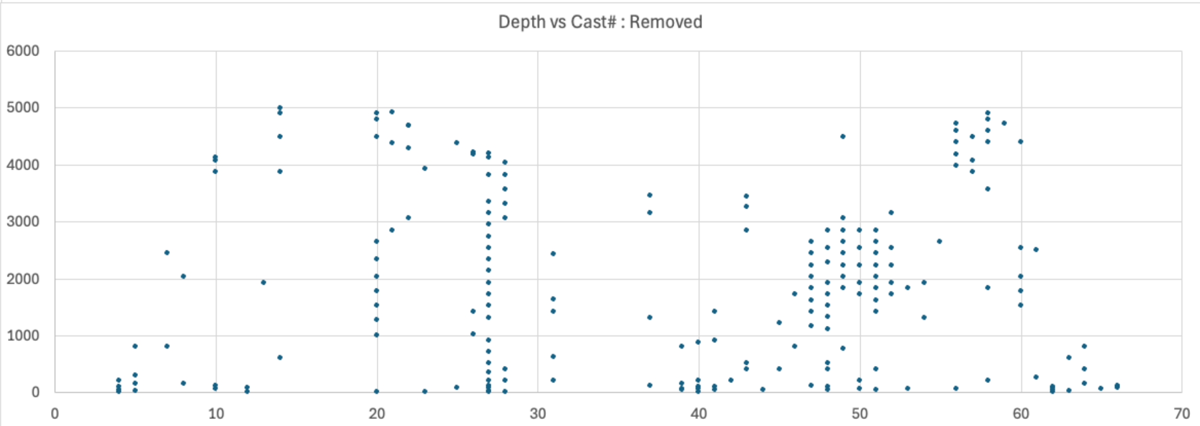
To do this we use Excel® spreadsheet and VBA code under Excel provided in the spreadsheet.
This process is summarised on the following flow chart



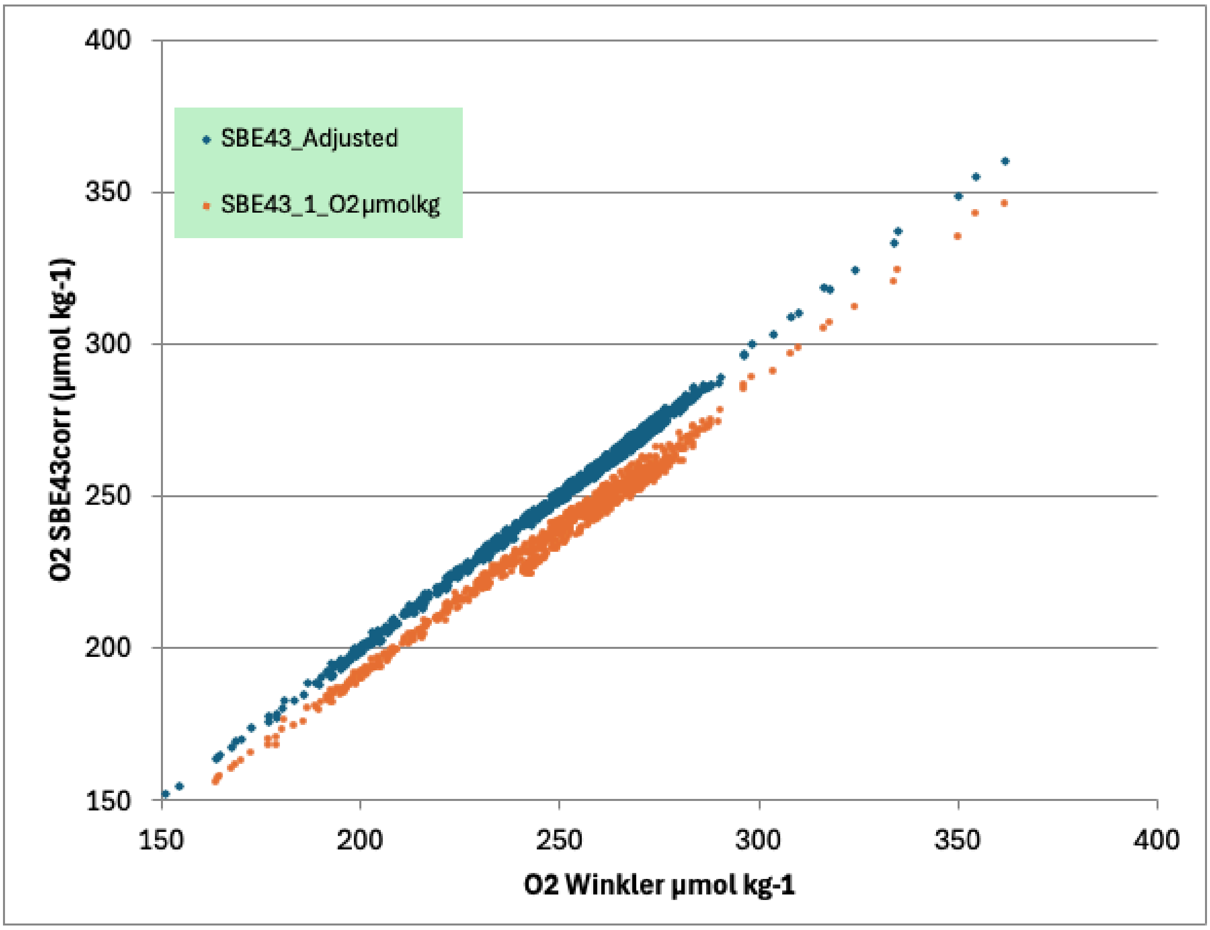
Using this itérative process, 212 points have been removed. 921 samples were kept for the adjustment of the SBE43 with winkler samples.



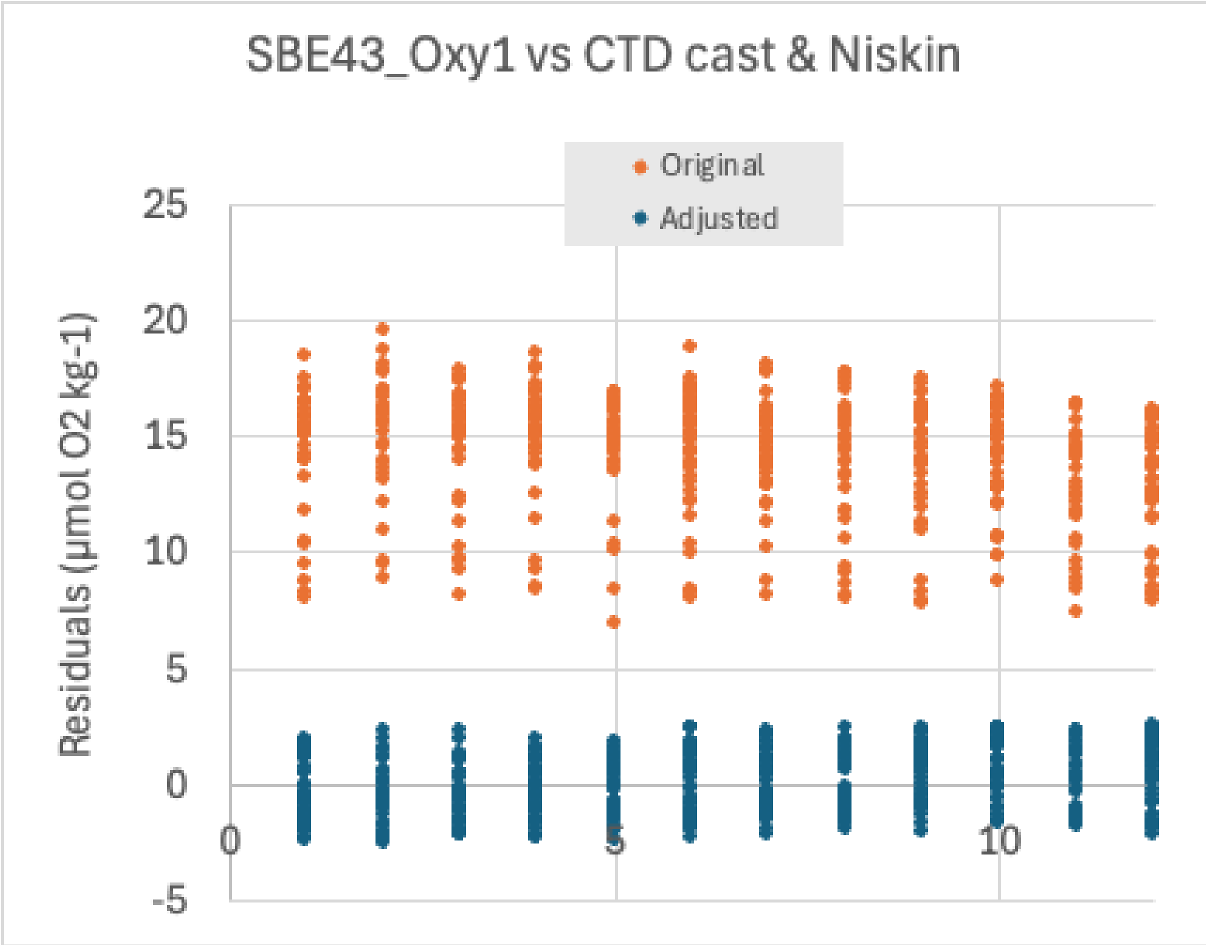




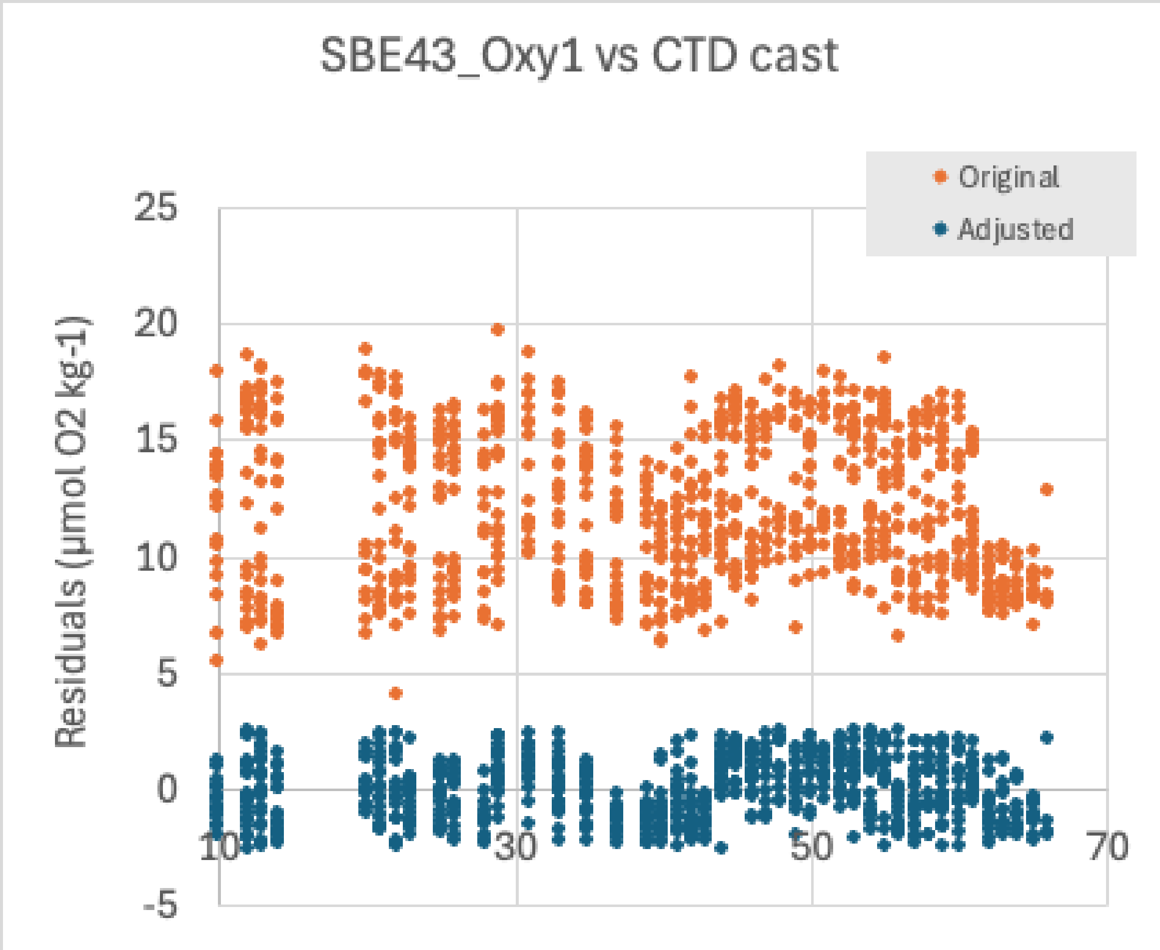
These 3 charts are showing the distribution of the removed data points (Winkler samples) to assess any potential bias such as faulty Niskin bottles. These points are randomly distributed and do not exhibit any obvious bias.



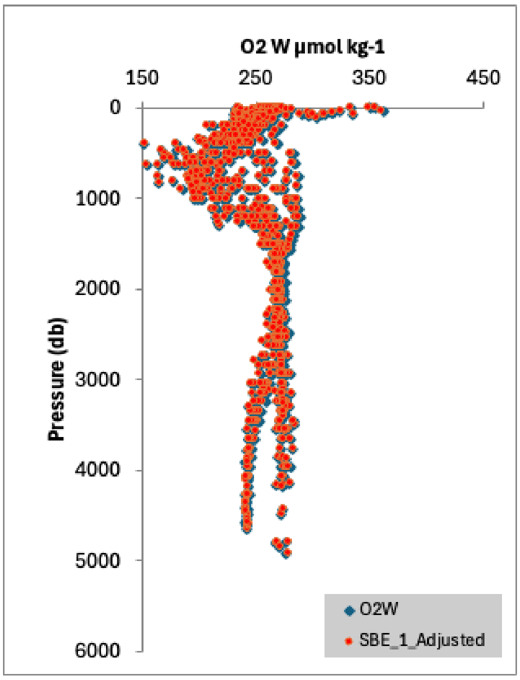
Plot property of the SBE43 vs Winkler samples before adjustment and after adjustment. The effect of “E” pressure coefficient can be identified from the disappearance of the scattering.

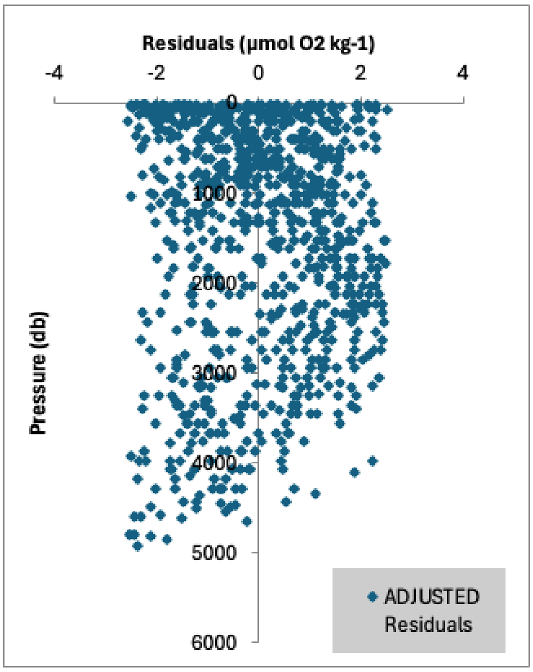


Distribution of the residuals before and after adjustment against Niskin bottle. No Bias can be observed.



Distribution of the residuals before and after adjustment against cast number.
No time drift can be derived from this plot.





Vertical distribution of O2 concentration (left) vs Pressure of the BTL-SBE43 and Winkler samples and O2 residuals (right). residuals are homogeneously distributed around the zero value throughout the water column, providing some confidence in the adjustment procedure of the coefficients.

The adjusted coefficient are

	Original	Adjusted
SOC	5.5183E-01	5.7136E-01
Offset	-5.0930E-01	-5.0233E-01
E	3.6000E-02	3.7649E-02

RMSE = 1.28

4. Reprocessing of CTD files with adjusted configuration files

A final step makes sure that the correction have well been applied using the adjusted *_adj.xmlcon files.

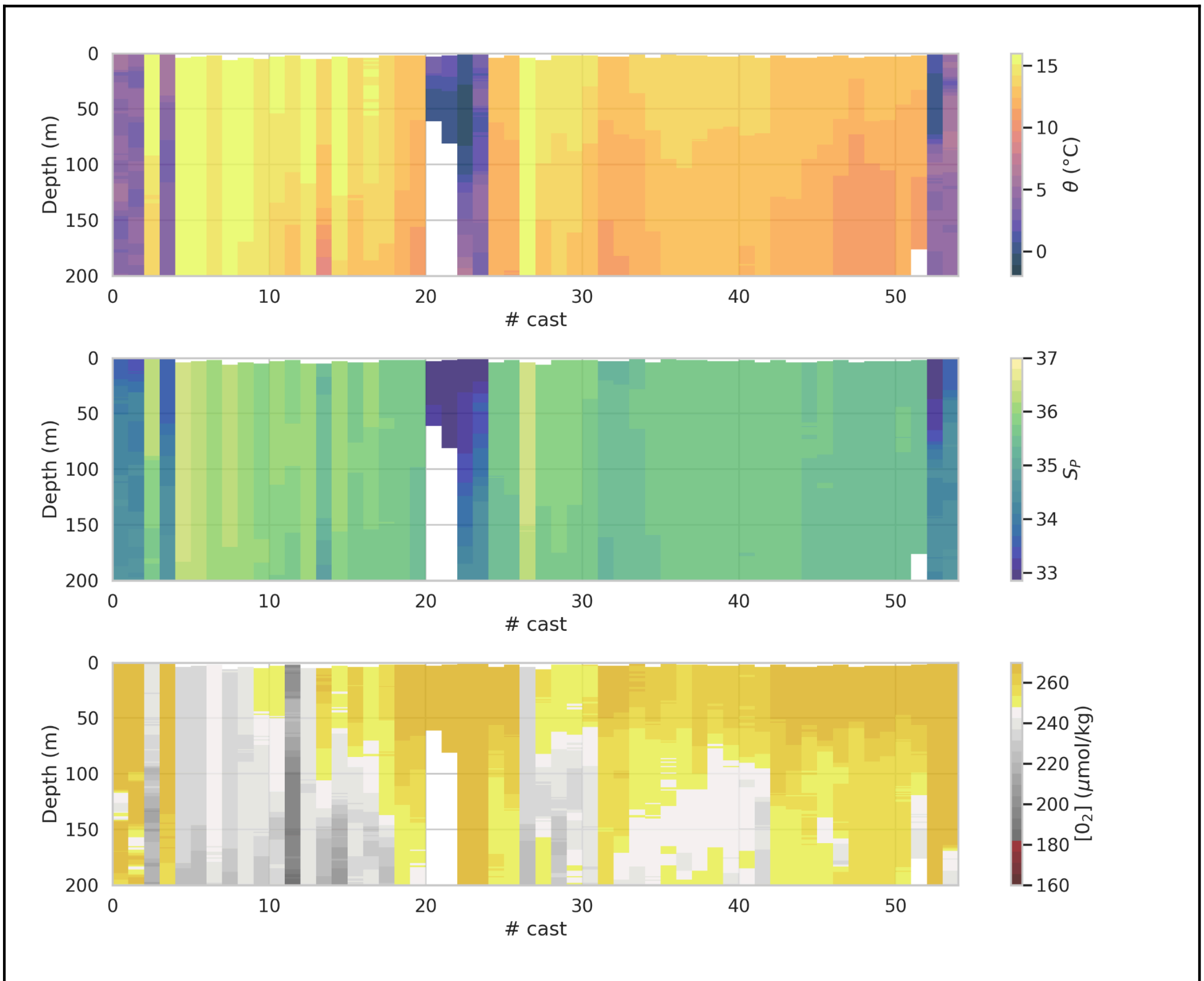


Figure : Overview of the reprocessed CTD files (temperature, salinity and oxygen) in the top 200m.

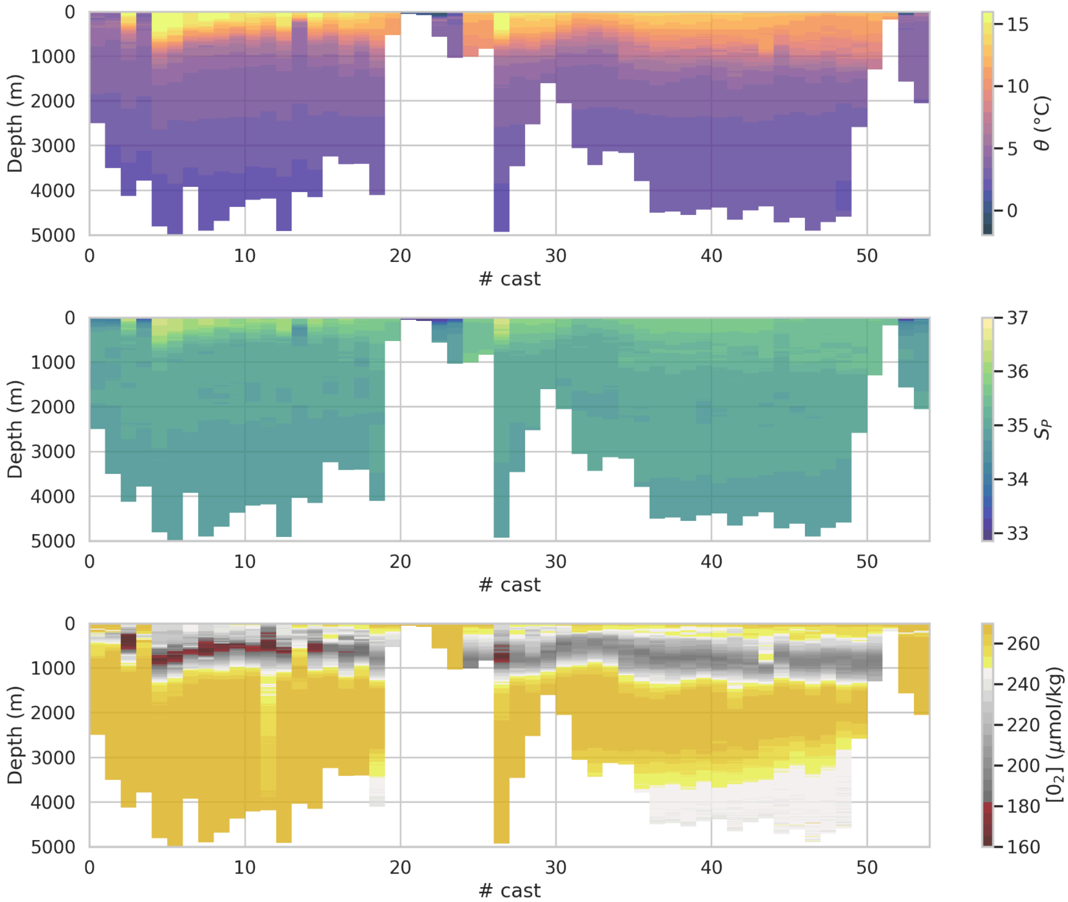


Figure : Overview of the reprocessed CTD files (temperature, salinity and oxygen) with full-depth y-axis.

Annexe Code R

Code R to merge single *.btl file into one file to ease the adjustment process.

```
##IAPSO
###HOMEPATH../IAPSO/SBE-profile-data/btl/" ### To be ADJUSTED for each user ####

file_label<-list.files(HOMEPATH)
source("…/IAPSO/CODE/read_btl_info_V_IAPSO.R") # call function
source("…/IAPSO/CODE/read_btl_V_IAPSO.R") # Call function
#	Bottle		Date Potemp090C	Sal00 Potemp190C	Sal11 Sbeox0ML/L
Sbox0Mm/Kg OxsolMm/Kg	PrDM	T090C		T190C	C0S/m	C1S/m		Sbeox0V
	Scan
#header<-c("Bottle","Month","Day","Year","Latitude","Longitude","PrDM","T090C","C0S/m","Sal00","T1
90C","C1S/m","Sal11","FlC","CStarTr0","Sbeox0V","Sbox0Mm/Kg")

header<-c("Bottle","Month","Day","Year",	"Potemp090C",	"Sal00", "Potemp190C",
"Sal11", "Sbeox0ML/L", "Sbox0Mm/Kg", "OxsolMm/Kg",	"PrDM",	"T090C",	"T190C",
	"C0S/m"  ,  "C1S/m"  ,  "Sbeox0V",	"Scan")

header_Out<-c("Bottle","Month","Day","Year","Latitude","Longitude",	"Potemp090C",
"Sal00", "Potemp190C",	"Sal11", "Sbeox0ML/L", "Sbox0Mm/Kg", "OxsolMm/Kg",	"PrDM",
"T090C",	"T190C",	"C0S/m"  ,	"C1S/m"  ,  "Sbeox0V","BTLFILE")
#################################################################
for(j in 1:length(file_label)){
#################################################################
  dfbtl<-read_btl(btl_ori_file=paste0(HOMEPATH,file_label[j]),btl_header=header,btl_file_name =
file_label[j])
  if (j==1) dfbtl_Out<-dfbtl else dfbtl_Out<-rbind(dfbtl_Out,dfbtl)
}
dfbtl_Out<-dfbtl_Out[,header_Out]
write.csv(dfbtl_Out,file=" …/IAPSO/SBE-profile-data/IAPSO_BTL_adj.csv", row.names = FALSE)


########################### FUNCTION READ_BTL
read_btl_info<-function(btl_ori_file,header_max=500){
#################################################################
#################################################################
line2head=0
line2skip2all=0
for (i in 1:header_max){

line<-scan(btl_ori_file,skip=i,nline=1,what="character",sep="",quiet=TRUE,nmax=1)
				if (length(line)==0) break
				if (line=="*") line2head=i
				if (line=="Bottle") line2skip2all=i
				}
				Allline<-i
				n_btl<-(Allline-line2skip2all-2)/2
				print(paste("number of niskin:",n_btl))
for (i in 1:line2head){

line<-scan(btl_ori_file,skip=i,nline=1,what="character",sep="",quiet=TRUE,nmax=3)
```

```
                              if (line[2]=="NMEA" & line[3]=="Latitude") line2LAT<-i
                              if (line[2]=="NMEA" & line[3]=="Longitude") line2LON<-i
                              if (line[2]=="NMEA" & line[3]=="UTC") line2TIME<-i
                              }
if (exists("line2LAT")==FALSE & exists("line2LON")==FALSE & exists("line2TIME")==FALSE) print("no
NMEA lat,lon,time available") else {
lat<-scan(btl_ori_file,skip=line2LAT,nline=1,what="character",sep="=",quiet=TRUE,nmax=4)[2]
lon<-scan(btl_ori_file,skip=line2LON,nline=1,what="character",sep="=",quiet=TRUE,nmax=4)[2]
time<-scan(btl_ori_file,skip=line2TIME,nline=1,what="character",sep="=",quiet=TRUE,nmax=4)[2]
print(paste("Latitude:",lat))
print(paste("Longitude:",lon))
print(paste("Time:",time))
}
##############################################################
#btl<-as.data.frame(matrix(NA,ncol=length(btl_header)))
##############################################################
line2read<-seq(2,n_btl*2,2)
line2skip<-line2skip2all+line2read
headerinfo<-scan(file=btl_ori_file,skip=line2skip2all,nline=1,what="character",sep="",quiet=TRUE)
print("header line is :")
print(headerinfo)
for (k in 1:length(line2read)) {
                    line<-read.table(file=btl_ori_file,skip=line2skip[k],nrows=1)
                    if (k==1) {
                              print("First data line is :")
                              print(line)
                              }
#                   btl[k,]<-line
                    }
}

########################## FUNCTION READ_BTL
read_btl_info<-function(btl_ori_file,header_max=500){
##############################################################
##############################################################
line2head=0
line2skip2all=0
for (i in 1:header_max){

line<-scan(btl_ori_file,skip=i,nline=1,what="character",sep="",quiet=TRUE,nmax=1)
                              if (length(line)==0) break
                              if (line=="*") line2head=i
                              if (line=="Bottle") line2skip2all=i
                              }
                              Allline<-i
                              n_btl<-(Allline-line2skip2all-2)/2
                              print(paste("number of niskin:",n_btl))
for (i in 1:line2head){

line<-scan(btl_ori_file,skip=i,nline=1,what="character",sep="",quiet=TRUE,nmax=3)
                              if (line[2]=="NMEA" & line[3]=="Latitude") line2LAT<-i
                              if (line[2]=="NMEA" & line[3]=="Longitude") line2LON<-i
```

```
                    if (line[2]=="NMEA" & line[3]=="UTC") line2TIME<-i
                    }
if (exists("line2LAT")==FALSE & exists("line2LON")==FALSE & exists("line2TIME")==FALSE) print("no NMEA lat,lon,time available") else {
lat<-scan(btl_ori_file,skip=line2LAT,nline=1,what="character",sep="=",quiet=TRUE,nmax=4)[2]
lon<-scan(btl_ori_file,skip=line2LON,nline=1,what="character",sep="=",quiet=TRUE,nmax=4)[2]
time<-scan(btl_ori_file,skip=line2TIME,nline=1,what="character",sep="=",quiet=TRUE,nmax=4)[2]
print(paste("Latitude:",lat))
print(paste("Longitude:",lon))
print(paste("Time:",time))
}
##############################################################
#btl<-as.data.frame(matrix(NA,ncol=length(btl_header)))
##############################################################
line2read<-seq(2,n_btl*2,2)
line2skip<-line2skip2all+line2read
headerinfo<-scan(file=btl_ori_file,skip=line2skip2all,nline=1,what="character",sep="",quiet=TRUE)
print("header line is :")
print(headerinfo)
for (k in 1:length(line2read)) {
            line<-read.table(file=btl_ori_file,skip=line2skip[k],nrows=1)
            if (k==1) {
                    print("First data line is :")
                    print(line)
                    }
#           btl[k,]<-line
            }
}
```

```
################
Code VBA Excel
################
Sub MergeBTL_Chemistry_IAPSO()


 File = "IAPSO_BTL_Merged.xlsm"
    O2W_S = "CE17007BottleODV"
    BTL = "IAPSO_BTL"
    O2 = "Oxygen"
    Sal = "Salinity"

i = 4 ' Index of F_O2
j = 3 ' Index of F_OUT
k = 2
M = 2
'Reading source of chemistry
    For i = 2 To 1196
    Cast = Workbooks(File).Worksheets(O2W_S).Cells(i, 1)
    ' Ensure Niskin are ranked othewise, test check NiskinNumber?
```

```
            Niskin = Workbooks(File).Worksheets(O2W_S).Cells(i, 8)

' Check if Sal or O2W exist
            O2W = Workbooks(File).Worksheets(O2W_S).Cells(i, 14)
            Salinity = Workbooks(File).Worksheets(O2W_S).Cells(i, 12)
            If O2W <> "" And O2W <> -999 Then
            'get BTL info
            For j = 2 To 1208
                        CastBtl = Workbooks(File).Worksheets(BTL).Cells(j, 20)
                        NiskinBtl = Workbooks(File).Worksheets(BTL).Cells(j, 1)
                        If CastBtl = Cast And NiskinBtl = Niskin Then
                        Workbooks(File).Worksheets(O2).Cells(k, 1) = Cast
                        Workbooks(File).Worksheets(O2).Cells(k, 2) = Niskin
                        Workbooks(File).Worksheets(O2).Cells(k, 3) = O2W
                        'Get BTL values  to complete
'                       Temp = Theta
'                       Salinity = Salinity
'                       SolO2_Theta_molkg = SolO2µmolkg(Temp, Salinity)
                        Workbooks(File).Worksheets(O2).Cells(k, 4) =
Workbooks(File).Worksheets(BTL).Cells(j, 2)                       'Month
                        Workbooks(File).Worksheets(O2).Cells(k, 5) =
Workbooks(File).Worksheets(BTL).Cells(j, 3)  'Day
                        Workbooks(File).Worksheets(O2).Cells(k, 6) =
Workbooks(File).Worksheets(BTL).Cells(j, 4)  'Year
                        Workbooks(File).Worksheets(O2).Cells(k, 7) =
Workbooks(File).Worksheets(BTL).Cells(j, 5)  'Latitude
                        Workbooks(File).Worksheets(O2).Cells(k, 8) =
Workbooks(File).Worksheets(BTL).Cells(j, 6)  'Longitude
                        Workbooks(File).Worksheets(O2).Cells(k, 9) =
Workbooks(File).Worksheets(BTL).Cells(j, 7)  'Potemp090C
                        Workbooks(File).Worksheets(O2).Cells(k, 10) =
Workbooks(File).Worksheets(BTL).Cells(j, 8)  'Sal00
                        Workbooks(File).Worksheets(O2).Cells(k, 11) =
Workbooks(File).Worksheets(BTL).Cells(j, 9)  'Potemp190C
                        Workbooks(File).Worksheets(O2).Cells(k, 12) =
Workbooks(File).Worksheets(BTL).Cells(j, 10)  'Sal11
                        Workbooks(File).Worksheets(O2).Cells(k, 13) =
Workbooks(File).Worksheets(BTL).Cells(j, 11)  'Sbeox0ML/L
                        Workbooks(File).Worksheets(O2).Cells(k, 14) =
Workbooks(File).Worksheets(BTL).Cells(j, 12)  'Sbox0Mm/Kg
                        Workbooks(File).Worksheets(O2).Cells(k, 15) =
Workbooks(File).Worksheets(BTL).Cells(j, 13)  'OxsolMm/Kg
                        Workbooks(File).Worksheets(O2).Cells(k, 16) =
Workbooks(File).Worksheets(BTL).Cells(j, 14)  'PrDM
                        Workbooks(File).Worksheets(O2).Cells(k, 17) =
Workbooks(File).Worksheets(BTL).Cells(j, 15)  'T090C
                        Workbooks(File).Worksheets(O2).Cells(k, 18) =
Workbooks(File).Worksheets(BTL).Cells(j, 16)  'T190C
```

```
                    Workbooks(File).Worksheets(O2).Cells(k, 19) =
Workbooks(File).Worksheets(BTL).Cells(j, 17)  'C0S/m
                    Workbooks(File).Worksheets(O2).Cells(k, 20) =
Workbooks(File).Worksheets(BTL).Cells(j, 18)  'C1S/m
                    Workbooks(File).Worksheets(O2).Cells(k, 21) =
Workbooks(File).Worksheets(BTL).Cells(j, 19)  'Sbeox0V
                    k = k + 1
                    End If

          Next j
          End If
          If Salinity <> "" And Salinity <> -999 Then
          'get BTL info
          For j = 2 To 1208
                    CastBtl = Workbooks(File).Worksheets(BTL).Cells(j, 20)
                    NiskinBtl = Workbooks(File).Worksheets(BTL).Cells(j, 1)
                    If CastBtl = Cast And NiskinBtl = Niskin Then
                    Workbooks(File).Worksheets(Sal).Cells(M, 1) = Cast
                    Workbooks(File).Worksheets(Sal).Cells(M, 2) = Niskin
                    Workbooks(File).Worksheets(Sal).Cells(M, 3) = Salinity
                    'Get BTL values  to complete
'                   Temp = Theta
'                   Salinity = Salinity
'                   SolO2_Theta_molkg = SolO2µmolkg(Temp, Salinity)
                    Workbooks(File).Worksheets(Sal).Cells(M, 4) =
Workbooks(File).Worksheets(BTL).Cells(j, 2)                    'Month
                    Workbooks(File).Worksheets(Sal).Cells(M, 5) =
Workbooks(File).Worksheets(BTL).Cells(j, 3)  'Day
                    Workbooks(File).Worksheets(Sal).Cells(M, 6) =
Workbooks(File).Worksheets(BTL).Cells(j, 4)  'Year
                    Workbooks(File).Worksheets(Sal).Cells(M, 7) =
Workbooks(File).Worksheets(BTL).Cells(j, 5)  'Latitude
                    Workbooks(File).Worksheets(Sal).Cells(M, 8) =
Workbooks(File).Worksheets(BTL).Cells(j, 6)  'Longitude
                    Workbooks(File).Worksheets(Sal).Cells(M, 9) =
Workbooks(File).Worksheets(BTL).Cells(j, 7)  'Potemp090C
                    Workbooks(File).Worksheets(Sal).Cells(M, 10) =
Workbooks(File).Worksheets(BTL).Cells(j, 8)  'Sal00
                    Workbooks(File).Worksheets(Sal).Cells(M, 11) =
Workbooks(File).Worksheets(BTL).Cells(j, 9)  'Potemp190C
                    Workbooks(File).Worksheets(Sal).Cells(M, 12) =
Workbooks(File).Worksheets(BTL).Cells(j, 10)  'Sal11
                    Workbooks(File).Worksheets(Sal).Cells(M, 13) =
Workbooks(File).Worksheets(BTL).Cells(j, 11)  'Sbeox0ML/L
                    Workbooks(File).Worksheets(Sal).Cells(M, 14) =
Workbooks(File).Worksheets(BTL).Cells(j, 12)  'Sbox0Mm/Kg
                    Workbooks(File).Worksheets(Sal).Cells(M, 15) =
Workbooks(File).Worksheets(BTL).Cells(j, 13)  'OxsolMm/Kg
```

```
                    Workbooks(File).Worksheets(Sal).Cells(M, 16) =
Workbooks(File).Worksheets(BTL).Cells(j, 14)  'PrDM
                    Workbooks(File).Worksheets(Sal).Cells(M, 17) =
Workbooks(File).Worksheets(BTL).Cells(j, 15)  'T090C
                    Workbooks(File).Worksheets(Sal).Cells(M, 18) =
Workbooks(File).Worksheets(BTL).Cells(j, 16)  'T190C
                    Workbooks(File).Worksheets(Sal).Cells(M, 19) =
Workbooks(File).Worksheets(BTL).Cells(j, 17)  'C0S/m
                    Workbooks(File).Worksheets(Sal).Cells(M, 20) =
Workbooks(File).Worksheets(BTL).Cells(j, 18)  'C1S/m
                    Workbooks(File).Worksheets(Sal).Cells(M, 21) =
Workbooks(File).Worksheets(BTL).Cells(j, 19)  'Sbeox0V
                    M = M + 1
                    End If


          Next j
          End If
          Next i
End Sub

Function SolO2mLL(Temp, Salinity)
Rem : calculate oxygen % saturation @ theta
                    Rem  H.E. GARCIA and L.I. GORDON, limnol. Oceanogr., 37(6), 1992,
1307-1312, Benson and Krause constants (ml/l)
                    aox0 = 2.00907
                    aox1 = 3.22014
                    aox2 = 4.0501
                    aox3 = 4.94457
                    aox4 = -0.256847
                    aox5 = 3.88767
                    box0 = -0.00624523
                    box1 = -0.00737614
                    box2 = -0.010341
                    box3 = -0.00817083
                    cox0 = -4.88682E-07
                    Ts = Log((298.15 - Temp) / (273.15 + Temp))
                    Aox = aox0 + aox1 * Ts + aox2 * Ts ^ 2 + aox3 * Ts ^ 3 + aox4 * Ts ^ 4 + aox5
* Ts ^ 5
                    Box = (box0 + box1 * Ts + box2 * Ts ^ 2 + box3 * Ts ^ 3) * Salinity
                    Cox = Aox + Box + cox0 * Salinity ^ 2
                    C4 = Exp(Cox) 'Solubility in ml/l
                    SolO2mLL = C4
                    'C4 = C4 * 1000 / 22.3916 ' O2 solubility in umol/l
End Function

Function SolO2µmolkg(Temp, Salinity)
          'function [conc_O2] = O2sol(S,T)
```

```
'% O2sol   Solubility of O2 in sea water

'%=====================================================================
=====
'% O2sol Version 1.1 4/4/2005
'%       Author: Roberta C. Hamme (Scripps Inst of Oceanography)
'%
'% USAGE:  concO2 = O2sol(S,T)
'%
'% DESCRIPTION:
'%       Solubility (saturation) of oxygen (O2) in sea water
'%       at 1-atm pressure of air including saturated water vapor
'%
'% INPUT:  (if S and T are not singular they must have same dimensions)
'%   S = salinity          [PSS]
'%   T = temperature [degree C]
'%
'% OUTPUT:
'%   concO2 = solubility of O2  [umol/kg]
'%
'% AUTHOR:  Roberta Hamme (rhamme@ucsd.edu)
'%
'% REFERENCE:
'%       Hernan E. Garcia and Louis I. Gordon, 1992.
'%       "Oxygen solubility in seawater: Better fitting equations"
'%       Limnology and Oceanography, 37, pp. 1307-1312.
'%
'% DISCLAIMER:
'%       This software is provided "as is" without warranty of any kind.

'%=====================================================================
=====

'% CALLER: general purpose
'% CALLEE: sw_dens_0.m

'%----------------------
'% Check input parameters
'%----------------------
'if nargin ~=2
'   error('O2sol.m: Must pass 2 parameters')
'end %if

'% Check S,T dimensions and verify consistent
'[ms,ns] = size(S);
'[mt,nt] = size(T);
```

```
'
'% Check that T&S have the same shape or are singular
'if ((ms~=mt) | (ns~=nt)) & (ms+ns>2) & (mt+nt>2)
'   error('O2sol: S & T must have same dimensions or be singular')
'end %if

'%------
'% BEGIN
'%------



'% convert T to scaled temperature
Temp_S = Log((298.15 - Temp) / (273.15 + Temp))

'% constants from Table 1 of Garcia & Gordon for the fit to Benson and Krause
(1984)
A0_O2 = 5.80871
A1_O2 = 3.20291
A2_O2 = 4.17887
A3_O2 = 5.10006
A4_O2 = -0.0986643
A5_O2 = 3.80369
B0_O2 = -0.00701577
B1_O2 = -0.00770028
B2_O2 = -0.0113864
B3_O2 = -0.00951519
C0_O2 = -2.75915E-07

' Corrected Eqn (8) of Garcia and Gordon 1992
Aox = A0_O2 + A1_O2 * Temp_S + A2_O2 * Temp_S ^ 2 + A3_O2 * Temp_S ^ 3 +
A4_O2 * Temp_S ^ 4 + A5_O2 * Temp_S ^ 5
Box = Salinity * (B0_O2 + B1_O2 * Temp_S + B2_O2 * Temp_S ^ 2 + B3_O2 *
Temp_S ^ 3)
Cox = Aox + Box + C0_O2 * Salinity ^ 2
SolO2µmolkg = Exp(Cox)

End Function


Function Density(Temp, Salinity)

'https://unesdoc.unesco.org/ark:/48223/pf0000188170
'Definitions cstes
'Density Constants
DSB0 = 0.824493
DSB1 = -0.0040899
DSB2 = 7.6438E-05
```

```
        DSB3 = -8.2467E-07
        DSB4 = 5.3875E-09

        DSC0 = -0.00572466
        DSC1 = 0.00010227
        DSC2 = -1.6546E-06

        DSD0 = 0.00048314

        DSA0 = 999.842594
        DSA1 = 0.06793952
        DSA2 = -0.00909529
        DSA3 = 0.0001001685
        DSA4 = -1.120083E-06
        DSA5 = 6.536332E-09
        'Secant bulk modulus (K) of seawater : contanst
        'Validity of EOS80 is valid for S=0 to 42; t=-2 to 40 fiC, p=0 to 10000 db

        DensityWaterTemp = DSA0 + DSA1 * Temp + DSA2 * Temp ^ 2 + DSA3 * Temp ^ 3 +
DSA4 * Temp ^ 4 + DSA5 * Temp ^ 5
        DensityTemp = DensityWaterTemp + (DSB0 + DSB1 * Temp + DSB2 * Temp ^ 2 +
DSB3 * Temp ^ 3 + DSB4 * Temp ^ 4) * Salinity + (DSC0 + DSC1 * Temp + DSC2 * Temp ^
2) * Salinity ^ 1.5 + DSD0 * Salinity ^ 2
        Density = DensityTemp / 1000

End Function
Function ThetaPot(Pressure, TIS, Salinity)
        'Add references
        'Definitions cstes
        'Density Constants
        DSB0 = 0.824493
        DSB1 = -0.0040899
        DSB2 = 7.6438E-05
        DSB3 = -8.2467E-07
        DSB4 = 5.3875E-09

        DSC0 = -0.00572466
        DSC1 = 0.00010227
        DSC2 = -1.6546E-06

        DSD0 = 0.00048314

        DSA0 = 999.842594
        DSA1 = 0.06793952
        DSA2 = -0.00909529
        DSA3 = 0.0001001685
        DSA4 = -1.120083E-06
        DSA5 = 6.536332E-09
```

'Secant bulk modulus (K) of seawater : contanst
'Validity of EOS80 is valid for S=0 to 42; t=-2 to 40 fiC, p=0 to 10000 db
KF0 = 54.6746
KF1 = -0.603459
KF2 = 0.011
KF3 = -6.17E-05

KG0 = 0.0794
KG1 = 0.0165
KG2 = -0.00053

KI0 = 0.00228
KI1 = -1.1E-05
KI2 = -1.61E-06

KJ0 = 0.000191

KM0 = -9.93E-07
KM1 = 2.08E-08
KM2 = 9.17E-10

KE0 = 19652.21
KE1 = 148.4206
KE2 = -2.327105
KE3 = 0.0136
KE4 = -5.16E-05

KH0 = 3.239908
KH1 = 0.00144
KH2 = 0.000116
KH3 = -5.78E-07

KK0 = 8.51E-05
KK1 = -6.12E-06
KK2 = 5.28E-08
'Adiabatic correction for theta pot
ATG0 = -2.1687E-16
ATG1 = 1.8676E-14
ATG2 = -4.6206E-13
ATG3 = 2.7759E-12
ATG4 = -1.1351E-10
ATG5 = -5.4481E-14
ATG6 = 8.733E-12
ATG7 = -6.7795E-10
ATG8 = 1.8741E-08
ATG9 = -4.2393E-08
ATG10 = 1.8932E-06

```
        ATG11 = 6.6228E-10
        ATG12 = -6.836E-08
        ATG13 = 8.5258E-06
        ATG14 = 3.5803E-05


        Sal0 = 35
        Pr0 = 0

        TempIS = TIS
        If Pressure = "" Then GoTo nexti
        If TempIS = "" Then GoTo nexti
        If Salinity = "" Then GoTo nexti
        'Density pure water
        DensityWaterIS = DSA0 + DSA1 * TempIS + DSA2 * TempIS ^ 2 + DSA3 * TempIS ^
3 + DSA4 * TempIS ^ 4 + DSA5 * TempIS ^ 5
        DensityIS = DensityWaterIS + (DSB0 + DSB1 * TempIS + DSB2 * TempIS ^ 2 +
DSB3 * TempIS ^ 3 + DSB4 * TempIS ^ 4) * Salinity + (DSC0 + DSC1 * TempIS + DSC2 *
TempIS ^ 2) * Salinity ^ 1.5 + DSD0 * Salinity ^ 2

        'Calculation Theta
        H = Pr0 - Pressure
        ATG0_SP0T0 = (((ATG0 * TempIS + ATG1) * TempIS + ATG2) * Pressure + ((ATG3 *
TempIS + ATG4) * (Salinity - Sal0) + ((ATG5 * TempIS + ATG6) * TempIS + ATG7) * TempIS
+ ATG8)) * Pressure + (ATG9 * TempIS + ATG10) * (Salinity - Sal0) + ((ATG11 * TempIS +
ATG12) * TempIS + ATG13) * TempIS + ATG14
        XK0 = H * ATG0_SP0T0
        T1 = TempIS + 0.5 * XK0
        Q0 = XK0
        P1 = Pressure + 0.5 * H
        AGT1_SP1T1 = (((ATG0 * T1 + ATG1) * T1 + ATG2) * P1 + ((ATG3 * T1 + ATG4) *
(Salinity - Sal0) + ((ATG5 * T1 + ATG6) * T1 + ATG7) * T1 + ATG8)) * P1 + (ATG9 * T1 +
ATG10) * (Salinity - Sal0) + ((ATG11 * T1 + ATG12) * T1 + ATG13) * T1 + ATG14
        XK1 = H * AGT1_SP1T1
        T2 = T1 + 0.29289322 * (XK1 - Q0)
        Q1 = 0.58578644 * XK1 + 0.121320344 * Q0
        ATG2_SP1T2 = (((ATG0 * T2 + ATG1) * T2 + ATG2) * P1 + ((ATG3 * T2 + ATG4) *
(Salinity - Sal0) + ((ATG5 * T2 + ATG6) * T2 + ATG7) * T2 + ATG8)) * P1 + (ATG9 * T2 +
ATG10) * (Salinity - Sal0) + ((ATG11 * T2 + ATG12) * T2 + ATG13) * T2 + ATG14
        XK2 = H * ATG2_SP1T2
        P2 = P1 + 0.5 * H
        T3 = T2 + 1.707106781 * (XK2 - Q1)
        ATG3_SP2T3 = (((ATG0 * T3 + ATG1) * T3 + ATG2) * P2 + ((ATG3 * T3 + ATG4) *
(Salinity - Sal0) + ((ATG5 * T3 + ATG6) * T3 + ATG7) * T3 + ATG8)) * P2 + (ATG9 * T3 +
ATG10) * (Salinity - Sal0) + ((ATG11 * T3 + ATG12) * T3 + ATG13) * T3 + ATG14
        Q2 = 3.414213562 * XK2 - 4.121320344 * Q1
        XK3 = ATG3_SP2T3 * H
        Theta = (XK3 - 2 * Q2) / 6 + T3
```

```
            ThetaPot = Theta
nexti:

End Function
Function Conductivity_Salinity(TIS, P, Conductivity)
                        'Conversion conductivity Salinity
                        'Cste definition
                        A1 = 2.07E-05
                        A2 = -6.37E-10
                        A3 = 3.989E-15
                        B1 = 0.03426
                        B2 = 0.0004464
                        B3 = 0.4215
                        B4 = -0.003107
                        C0 = 0.6766097
                        C1 = 0.0200564
                        C2 = 0.0001104259
                        C3 = -6.9698E-07
                        C4 = 1.0031E-09
                        Dim aa(6)
                        aa(0) = 0.008
                        aa(1) = -0.1692
                        aa(2) = 25.3851
                        aa(3) = 14.0941
                        aa(4) = -7.0261
                        aa(5) = 2.7081
                        Dim bb(6)
                        bb(0) = 0.0005
                        bb(1) = -0.0056
                        bb(2) = -0.0066
                        bb(3) = -0.0375
                        bb(4) = 0.0636
                        bb(5) = -0.0144
                        'P in decibar,T in deg C,C en S/m
                        '// C = conductivity S/m, T = temperature deg C ITPS-68, P = pressure in
decibars
                        If Conductivity <= 0# Then
                        Salinity = 0#
                        Else
                        Conductivity = 10# * Conductivity '/* convert Siemens/meter to mmhos/cm */
                        R = Conductivity / 42.914
                        Value = 1 + B1 * TIS + B2 * TIS * TIS + B3 * R + B4 * R * TIS
                        RP = 1 + (P * (A1 + P * (A2 + P * A3))) / Value
                        Value = RP * (C0 + (TIS * (C1 + TIS * (C2 + TIS * (C3 + TIS * C4)))))
                        RT = R / Value
                        If (RT <= 0#) Then RT = 1E-06
                        sum1 = 0
                        sum2 = 0
```

```
                    For ii = 0 To 5
                              Temp = RT ^ (ii / 2#)
                              sum1 = sum1 + aa(ii) * Temp
                              sum2 = sum2 + bb(ii) * Temp
                    Next ii
                    Value = 1# + 0.0162 * (TIS - 15#)
                    Salinity = sum1 + sum2 * (TIS - 15#) / Value
                    Conductivity_Salinity = Salinity
                    End If
End Function


*****************
Sub FILTER_OUT()


File = "IAPSO_BTL_Merged.xlsm"
          SBE43_O2W = "O2WSBE43_Ox0_v02"
          OUT = "Removed"
          Start_i = 2
          End_i = 1136
          Start_k = 4
          End_k = 1136
          P = 1
          MyLoop = 1
Do_It_Again:
          For i = Start_i To End_i
          flag = Workbooks(File).Worksheets(SBE43_O2W).Cells(i, 26)
          If flag = "OUT" Then
                    'Select Line
                    MyRange = "(d" & i & ":Y" & i & ")"
                    MyFormula = MyRange
                    Range(MyFormula).Select
                    'Erase line
'                   Selection.Cut
'                   Worksheets(Removed).Activate'
'                   Workbooks(File).Worksheets(OUT).Cells(P, 1) =
Workbooks(File).Worksheets(SBE43_O2W).Cells(i, 1)
'                   Workbooks(File).Worksheets(OUT).Cells(P, 2) =
Workbooks(File).Worksheets(SBE43_O2W).Cells(i, 2)
'                   Workbooks(File).Worksheets(OUT).Cells(P, 1) =
Workbooks(File).Worksheets(SBE43_O2W).Cells(i, 3)

                    Selection.ClearContents
          End If
          Next i
          SolverOk SetCell:="$V$2", MaxMinVal:=2, ValueOf:=0, ByChange:="$AA$1:$AA$3",
Engine:=3, EngineDesc:="Evolutionary"
```

```
        'SolverSolve
        SolverSolve UserFinish:=True
        SolverFinish KeepFinal:=1
        For j = Start_i To End_i
                Flag2 = Workbooks(File).Worksheets(SBE43_O2W).Cells(j, 26)
                If Flag2 = "OUT" Then
                MyLoop = MyLoop + 1
                GoTo Do_It_Again
                End If
        Next j
        Workbooks(File).Worksheets(SBE43_O2W).Cells(32, 28) = MyLoop
        For k = Start_k To End_k
                If Workbooks(File).Worksheets(SBE43_O2W).Cells(k, 4) = "" Then
                Workbooks(File).Worksheets(OUT).Cells(P, 1) =
Workbooks(File).Worksheets(SBE43_O2W).Cells(k, 1)
                Workbooks(File).Worksheets(OUT).Cells(P, 2) =
Workbooks(File).Worksheets(SBE43_O2W).Cells(k, 2)
                Workbooks(File).Worksheets(OUT).Cells(P, 3) =
Workbooks(File).Worksheets(SBE43_O2W).Cells(k, 3)
                P = P + 1
                End If
        Next k
End Sub

****************
Sub Salinity_FILTER_OUT()

        'Salinity_SBE04_00_1
        '




File = "IAPSO_BTL_Merged.xlsm"
        SBE43_O2W = "Salinity_SBE04_00_1"
        OUT = "Removed_Salinity"
        Start_i = 4
        End_i = 1151
        Start_k = 4
        End_k = 1151
        P = 1
        MyLoop = 1
        Flag2 = ""
Do_It_Again:
        For i = Start_i To End_i
        flag = Workbooks(File).Worksheets(SBE43_O2W).Cells(i, 13)
        If flag = "OUT" Then
                'Select Line
                MyRange = "(d" & i & ":M" & i & ")"
```

```
                    MyFormula = MyRange
                    Range(MyFormula).Select
                    'Erase line
'                   Selection.Cut
'                   Worksheets(Removed).Activate'
'                   Workbooks(File).Worksheets(OUT).Cells(P, 1) =
Workbooks(File).Worksheets(SBE43_O2W).Cells(i, 1)
'                   Workbooks(File).Worksheets(OUT).Cells(P, 2) =
Workbooks(File).Worksheets(SBE43_O2W).Cells(i, 2)
'                   Workbooks(File).Worksheets(OUT).Cells(P, 1) =
Workbooks(File).Worksheets(SBE43_O2W).Cells(i, 3)

                    Selection.ClearContents
          End If
          Next i
          SolverOk SetCell:="$V$2", MaxMinVal:=2, ValueOf:=0,
ByChange:="$NN1$1:$NN$3", Engine:=3, EngineDesc:="Evolutionary"
          'SolverSolve
          SolverSolve UserFinish:=True
          SolverFinish KeepFinal:=1
          For j = Start_i To End_i
                    Flag2 = Workbooks(File).Worksheets(SBE43_O2W).Cells(j, 26)
                    If Flag2 = "OUT" Then
                    MyLoop = MyLoop + 1
                    GoTo Do_It_Again
                    End If
          Next j
          Workbooks(File).Worksheets(SBE43_O2W).Cells(20, 15) = MyLoop
          For k = Start_k To End_k
                    If Workbooks(File).Worksheets(SBE43_O2W).Cells(k, 4) = "" Then
                    Workbooks(File).Worksheets(OUT).Cells(P, 1) =
Workbooks(File).Worksheets(SBE43_O2W).Cells(k, 1)
                    Workbooks(File).Worksheets(OUT).Cells(P, 2) =
Workbooks(File).Worksheets(SBE43_O2W).Cells(k, 2)
                    Workbooks(File).Worksheets(OUT).Cells(P, 3) =
Workbooks(File).Worksheets(SBE43_O2W).Cells(k, 3)
                    P = P + 1
                    End If
          Next k
End Sub
```