Earth System
Science
Data

Open Access

*Supplement of*

# Harmonized Soil Database of Ecuador (HESD): data from 2009 to 2015
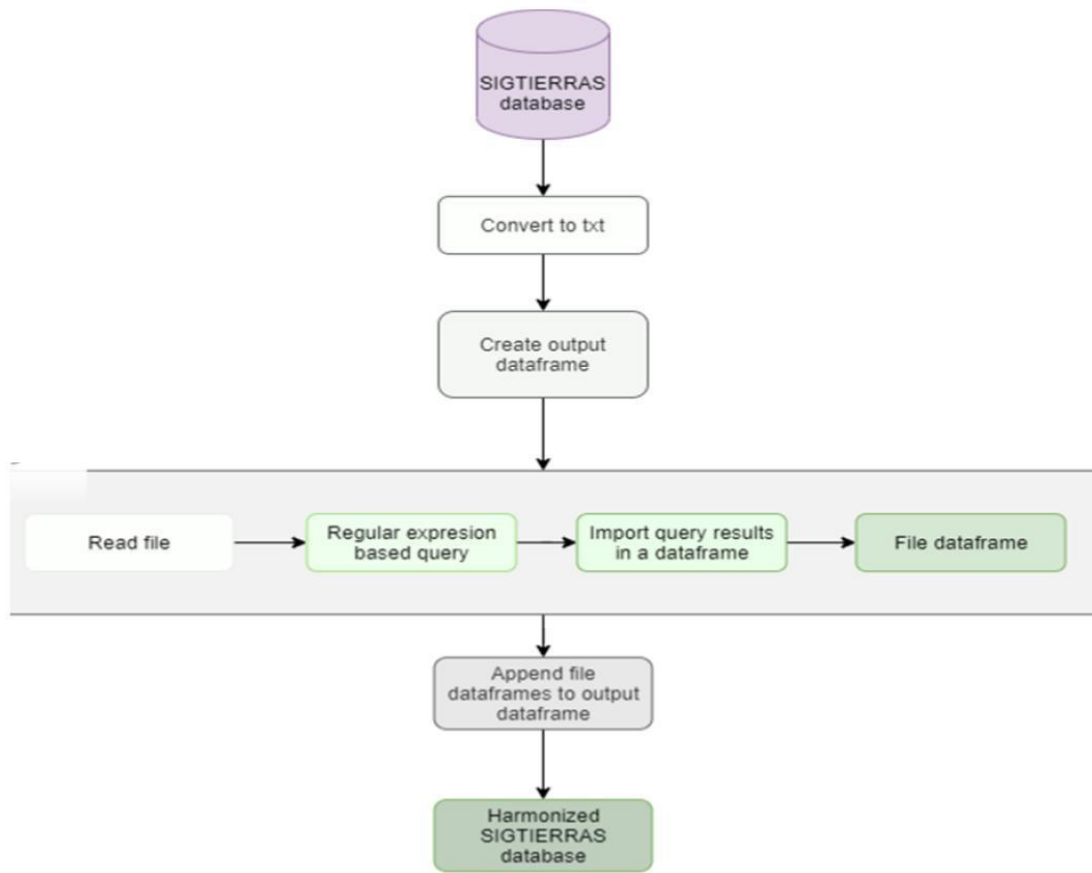
**Daphne Armas et al.**

*Correspondence to:* Cecilio Oyonarte (coyonart@ual.es) and Daphne Armas (daphne.armas@gmail.com)

2 **Supplement A. SIGTIERRAS approach**

3



4

5

6 **Figure S1.** Workflow steps to extract information from SIGTIERRAS´s files.

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

**Supplement B. Script files of SIGTIERRAS**

```python
# Author: Fernando Bezares Sanfelip
# Email: fernando.bezares@cesefor.com | fernandobezares95@hotmail.es
import re
import glob, os
import subprocess

# Ecuador soil database generation project
# Pass list of PDFs to R

################################################

# FILES OF SIGTIERRAS

################################################

# Call the R script that will convert to txt
# indicate the .pdf
pdf_files=
glob.glob(os.path.join('C:\\Users\\fbezares\\Documents\\01_Proyectos\\06_Almeria\\SIGTIERRAS\\samp
le','*.pdf'))
print(pdf_files)

# first position indicate where the Rscript.exe is
# second position indicate Converter_pdf_txt_python.R
commands = ['C:\\Program Files\\R\\R-
4.0.2\\bin\\x64\\Rscript.exe','C:\\Users\\fbezares\\Documents\\01_Proyectos\\06_Almeria\\SIGTIERRAS\
\Conversor_pdf_txt_python.R']

subprocess.call(commands + pdf_files, shell=True)
print(commands+pdf_files)

# we indicate where the txt files are (same folder as pdf)
txt_files =
glob.glob(os.path.join('C:\\Users\\fbezares\\Documents\\01_Proyectos\\06_Almeria\\SIGTIERRAS\\samp
le','*.txt'))
print(txt_files)
# open the soil csv file
```

```python
61    encabezado_perfiles_Sig = 'PerID; ID_nac ; COORD_X; COORDY; altitud_m ;Clas_SGST;
62    Clas_GGST; Clas_OST; RTS; RHS; Prof_efec; Geología;Geoforma; pend_local; Uso Tierra; Vegetación;
63    Rocosidad; Pedre; Tipo_erosion; Grado_erosion; Drena; fecha_inte'
64    # create file to write files to
65    arch = open('BD_SUELOS_PERFILES_SIGTIERRAS.csv','a')
66    arch.write('{}\n'.format(encabezado_perfiles_Sig))
67    arch.close()
68    # Go through the txt to apply the regular expressions
69    for i,txt_file in enumerate(txt_files):
70            texto = open(txt_file,'r')
71            string ="
72            for i in texto:
73                    string=string+i
74            arch=open('BD_SUELOS_PERFILES_SIGTIERRAS.csv','a')
75
76            # regular expressions
77            ID_nac=re.findall(r"(?<=[C,c][ó,o]digo [F,f]icha\" \")[^\"]+",string)[0].strip()
78            coord_x=re.findall(r"(?<=[X,x]:\" \")[^\"]+",string)[0]
79            coord_y=re.findall(r"(?<=[Y,y]:\" \")[^\"]+",string)[0]
80            altitud_m=re.findall(r"(?<=[A,a]ltitud:\" \")[^\"]+",string)[0]
81            fecha_inte=re.findall(r"(?<=[F,f]echa descripci[ó,o]n)[^\n]+",string)[0]
82            Geología=re.findall(r"(?<=[F,f]ormaci[ó,o]n [G,g]eol[ó,o]gica:\" \")[^\"]+",string)[0]
83            pend_local=re.findall(r"(?<=[P,p]endiente [L,l]ocal:\" \")[^\%]+",string)[0]
84            Geoforma=re.findall(r"(?<=[G,g]eoforma:\" \")[^\"]+",string)[0]
85            Uso_Tierra=re.findall(r"(?<=[u,U]so de la [T,t]ierra:\" \")[^\"]+",string)[0]
86            Tipo_erosion=re.findall(r"(?<=[C,c]ategor[í,i]a\" \")[^\"]+",string)[0]
87            Grado_erosion=re.findall(r"(?<=[G,g]rado\" \")[^\"]+",string)[0]
88            rts=re.findall(r"(?<=[R,r][é,e]gimen de temperatura del\"\s\")[^\"]+",string)[0]
89            rhs=re.findall(r"(?<=[R,r][é,e]gimen de humedad del suelo\:\"\s\")[^\"]+",string)[0]
90            Prof_efec=re.findall(r"(?<=[v,V]alor:\" \")[^\"]+",string)
91            if len(Prof_efec)<2:
92                    Prof_efec=re.findall(r"(?<=[v,V]alor:\" \")[^\"]+",string)[0]
93            else:
94                    Prof_efec=re.findall(r"(?<=[v,V]alor:\" \")[^\"]+",string)[1]
95
96            Vegetacion=re.findall(r"(?<=[v,V]egetaci[ó,o]n:\" \")[^\"]+",string)[0]
97            Rocosidad=re.findall(r"(?<=Abundancia \(\%\)\" \")[^\"]+",string)[0]
98            Pedre=re.findall(r"(?<=Abundancia \(\%\)\" \")[^\"]+",string)[1]
99            Drena=re.findall(r"(?<=[P,p]ermeabilidad [D,d]renaje [N,n]atural:\" \")[^\"]+",string)[0]
100
```

```python
101             # extraction of local slope values
102
103             if '>' in pend_local:
104                     p=pend_local.split('>')[1].split('-')
105                     pend_local=(float(p[1])+float(p[0]))/2
106             elif '>' not in pend_local:
107                     p=pend_local.split('(')[1].split('-')
108                     pend_local=(float(p[1])+float(p[0]))/2
109
110             # extraction of stoniness values
111             if '<' in Pedre:
112                     Pedre=Pedre.split('<')[1]
113             elif Pedre=='-':
114                     Pedre='0'
115             elif '-'in Pedre:
116                     pe=Pedre.split('-')
117
118                     if pe ==['','','']:
119                             Pedre='NAN'
120                     else:
121                             int_sup=float(pe[1])
122                             int_inf=pe[0].split(' ')[1]
123
124                             Pedre=(int_sup+float(int_inf))/2
125             else:
126                     Pedre=Pedre.split(' ')[1]
127
128             # extraction of rockiness values
129
130             if '<' in Rocosidad:
131                     Rocosidad = Rocosidad.split('<')[1]
132             elif Rocosidad == '-':
133                     Rocosidad = '0'
134             elif '-'in Rocosidad:
135                     pe=Rocosidad.split('-')
136                     if pe ==['','','']:
137                             Pedre='NAN'
138                     else:
139                             int_sup=float(pe[1])
140                             int_inf=pe[0].split(' ')[1]
```

```python
141                         Rocosidad=(int_sup+float(int_inf))/2
142             else:
143                     Rocosidad=Rocosidad.split(' ')[1]
144
145             # exception handling for vegetation and land use
146             if Vegetacion == '-' or Vegetacion == ' ':
147                     Vegetacion = 'NAN'
148             elif Uso_Tierra == '-' or Uso_Tierra == ' ':
149                     Uso_Tierra = 'NAN'
150             else:
151                     pass
152
153
154
155
156
157
158
159             # classification for horizons
160             Orden=re.findall(r"(?<=Orden final:\" \")[^\"]+",string)[0]# orden
161             Sistema_clas=re.findall(r"(?<=Clasificaci[ó,o]n Final\s\()[^\)]+",string)[0]
162             gran_grupo=re.findall(r"(?<=Gran grupo final:\" \")[^\"]+",string)[0]# gran grupo
163             Subgrupo=re.findall(r"(?<=Subgrupo final:\" \")[^\"]+",string)[0]# subgrupo
164
165             # Horizons code
166             codigo_horizonte=re.findall(r"(?<=[S,s][í,i]mbolo: )[^\"]+",string)
167
168             # roots
169             raices_vf=re.findall(r"(?<=VF \([M,m]uy [F,f]ina\)[\",\s])[^\"]+",string)
170             raices_f=re.findall(r"(?<=F \([F,f]ina\)[\",\s]{3})[^\"]+",string)
171             raices_mf=re.findall(r"(?<=M \([M,m]edia\)[\",\s]{3})[^\"]+",string)
172             raices_g=re.findall(r"(?<=C \([G,g]ruesa\)[\",\s]{3})[^\"]+",string)
173
174             # Depending on the code, they will have different root values
175
176             for i in range(len(codigo_horizonte)):
177
178                     ch=codigo_horizonte[i]
179                     r_vf=raices_vf[i]
180                     r_f=raices_f[i]
```
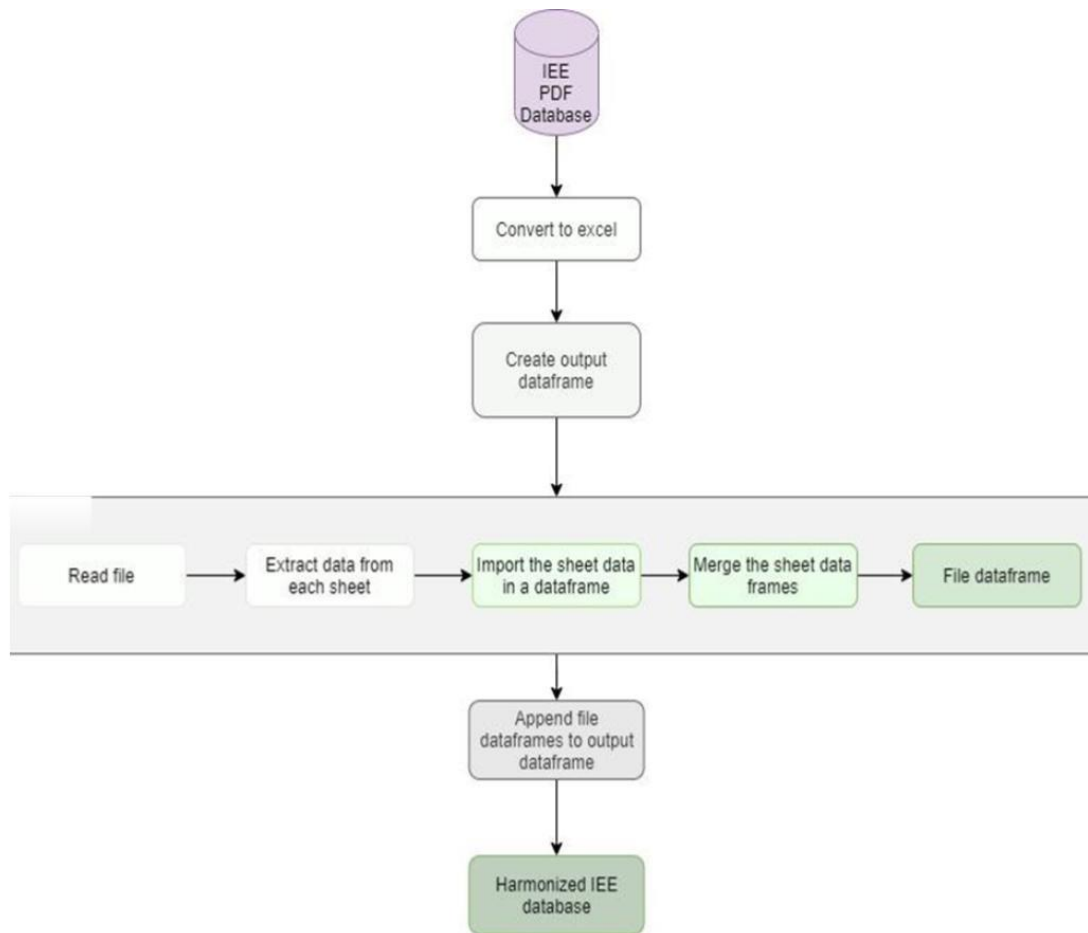
```python
181                    r_m=raices_mf[i]
182                    r_g=raices_g[i]
183
184            # indicate if there are roots in the horizon
185                    if r_vf!='Ninguna' or r_f!='Ninguna'or r_m!='Ninguna'or r_g!='Ninguna':
186                            raices='1'
187                    else:
188                            raices='0'
189                    # 'CODIGO; HORIZONTE; X; Y; SISTEMA CLASIFICACION; ORDEN; GRAN
190    GRUPO; SUBGRUPO; FECHA; RAICES'
191                    #contenido_horizontes= ID_nac+';'+ch+';'+coord_x+';'+ coord_y +';'+ Sistema_clas
192    +';'+Orden+';'+gran_grupo+';'+Subgrupo+';'+fecha_inte+';'+raices
193                    #arch_hori.write('{ }\n'.format(contenido_horizontes))
194
195
196            contenido=' '+';'+ID_nac+';'+coord_x+';'+ coord_y
197    +';'+altitud_m+';'+Subgrupo+';'+gran_grupo+';'+Orden+';'+ rts +';'+rhs+';'+ Prof_efec+';'+ Geología +';'+
198    Geoforma +';'+str(pend_local)+';'+ Uso_Tierra +';'+
199    Vegetacion+';'+str(Rocosidad)+';'+str(Pedre)+';'+Tipo_erosion+';'+Grado_erosion+';'+
200    Drena+';'+fecha_inte
201
202            arch.write('{ }\n'.format(contenido))
203            arch.close()
204            texto.close()
205            os.remove(txt_file)
206
207    #(?<=VF \([M,m]uy [F,f]ina\)[\",\s])[^\"]+
208    #(?<=\"1\" )\"[^\"]+[^\d]+[^\"]+[^\d]+[^\"]+[^\d]+[^\"]+
209    #\"[a-zA-Z]{1,3}[a-z0-9]*\"
210
211
212
213
214
215
216
217
218
219
220
```

221 **Supplement C. IEE approach**

222



223

224

225 **Figure S2.** Workflow steps to extract information from IEE's files.

226
227
228
229
230
231
232
233
234
235
236
237
238

**Supplement D. Script files of IEE**

```
# Author: Fernando Bezares Sanfelip
# Email: fernando.bezares@cesefor.com | fernandobezares95@hotmail.es

## DESCRIPTION ##

# The script aims to extract in the most automated way the tabular and text information from a soil-
backup-pdf-format database files converted to excel.
# Each excel file corresponds to a soil profile where profile descritpion and tabular data are provided.

# The logic of the script is to generate a dataframe for each sheet in the excel file. Each sheet corresponds
to a table with chemical or physical description
# for each table the target columns will be identified and pasted in a dictionary, that will form the
dataframe. Due to many inconsistences on the origin of the data,
# fields were not always the same, in the same position or with the same name. After converting pdf to
excel some names or rows were usually pasted together and therefore
# it was hard to find a systematic rule to split them. Therefore, the functions extract_info and
igualar_longitud were desinged to overcome the most frequent cases of
# conversion failures between pdf and excel.

# After creating a dataframe for each table, all the data frames were merged in a common dataframe for
the excel file by the dataframe 2, that holds the information
# of the horizons. Finally, the file dataframe is pasted in a general dataframe that holds teh information
for all the excel files.

# please bare in mind that the goal of this script is to show an example of how data can be extracted from
excel or pdf. Therefore, many rules are taylored just for the
# our data and it is not suitable for other datasets.

# --------------------------------------------------------------------------------------------------------------------
-----------------------------------------

# Import libaries
from openpyxl import Workbook
from openpyxl import load_workbook
import pandas as pd
import glob, os, re

# import data
```

```python
278   excel_files=
279   glob.glob(os.path.join('C:\\Users\\fbezares\\Documents\\01_Proyectos\\06_Almeria\\IEE\\sample','*.xlsx')
280   )
281
282   # DATAFRAME WHERE WE WILL ACCUMULATE ALL THE FINAL INFORMATION AND
283   WILL BE CONVERTED TO EXCEL.
284   print(excel_files)
285   df_final=pd.DataFrame({})
286
287   def extract_info(name,colu,referencia,diccionario,d_ref,n_complemento="):
288       "' looks for information within each column of text. First it checks if the column has a name,
289   otherwise it passes. Then, it checks if it has line breaks, if it does,
290       it generates a list with the information taking into account these line breaks, then it makes sure
291   that the length of the elements introduced in the dictionary are the same when generating the dataframe."'
292       if colu[0]==None:
293           pass
294       elif colu[0]==name:
295           if len(colu)==1:
296               pass
297           # When all the information is in a box, check for line break characters.
298           elif '\n' in str(colu[1]):
299               col=colu[1].split('\n')
300
301               g=[x for x in col]
302
303               # create a list with the information separated by the character "\n" now  check
304   for blank spaces
305               for k, i in enumerate(g):
306                   if r" " in i:
307                       s=i.split(' ')#
308
309                       s=s[::-1] # reverse the list order
310                       g.pop(k) # pop the element that is joined
311                       for j in s:
312                           g.insert(k,j) # We insert the separated value, in the
313   copy of the list, where the value with the joined data was before it.
314                   else:
315                       pass
316               g=col
317               col.insert(0,name)
```

```python
318                         print(name, col)
319
320                         if d_ref.get(referencia)==None:
321                                 print(d_ref.get(referencia))
322                         else:
323                                 while len(col)<(len(d_ref.get(referencia))+1):
324                                         col.append(None)
325
326                         diccionario.update({name+n_complemento:col[1:]})
327
328                 else:
329                         diccionario.update({name:list(colu[1:])})
330         pass
331
332 def igualar_longitud(diccionario,campo):
333         '''Seeks to equalise the length of the dictionary fields once the horizon column is included.'''
334         for k,i in  enumerate(diccionario.items()):
335                 print(i)
336
337                 if i==None:
338                         pass
339                 elif len(i[1])==len(diccionario.get(campo)):
340                         pass
341                 else:
342                         while len(i[1])<len(diccionario.get(campo))and k!=0:
343                                 val=i[1].append(None)
344
345
346
347
348
349 for i,file in enumerate(excel_files):
350
351         ## iterate over the sheets
352         ## save and create the document with the stored information
353         ## access the information in a file
354         try:
355                 wb2 = load_workbook(file)
356         except Exception as e:
357                 pass
```

```python
        ws=wb2.active

        d={
                'Horizonte/ Capa':[None],
                'Profundidad (cm)':[None],
                'Arena':[None],
                'Limo':[None],
                'Arcilla':[None],
                'Clase textural':[None],
                'Da\n3\n(g/cm  )':[None]
                }

        # Create the dictionaries that will be used for the data frames
        dp={}
        d2={}
        d3={}
        d4={}
        d5={}
        d6={}
        d_rp={}

        # dataframe where the data will be pasted in each iteration
        dff2=pd.DataFrame({'Horizonte/ Capa':[None]})


        for sheet in wb2:

                # check table 1
                if sheet.title=='Table 1':
                        # check postion A3 for profile name
                        c = sheet['A3'].value

                        print(c)
                        # alternatively check for A2
                        if c=='' or c==None:
                                c=sheet['A2'].value

                        else:
                                pass
```

```python
398
399                                 #  extract the code by splitting the string by the separators identified
400                                 if ':' in c:
401                                         perfil=c.split(':')[1]
402                                 else:
403                                         perfil=c.split('.')[1]
404
405                                 # try to include in the dataframe, if not it will generate an log document
406                                 try:
407                                         dfp=pd.DataFrame(dp)
408                                 except Exception as e:
409                                         log=open('descripcion_errores.txt','a')
410                                         log.write('error {} en {}'.format(e,sheet.title))
411                                         log.close()
412
413                 # Table 4
414                 if sheet.title=='Table 4':
415
416                                 # check in position A2 if not, then check from the file name
417                                 c = sheet['A2'].value
418
419                                 if c=='' or c==None:
420                                         pass
421                                 else:
422                                         perfil=file.split('\\')[-1]
423                                         perfil=perfil.split('.')[0]
424
425                                 # Paste in the dataframe
426                                 try:
427                                         dfp=pd.DataFrame(dp)
428                                 # generate a log file with the name of the file and error
429                                 except Exception as e:
430                                         log=open('descripcion_errores.txt','a')
431                                         log.write('error {} en {}'.format(e,sheet.title))
432                                         log.close()
433
434
435
436
437                 # table 5 where the descrption for the field is included
```

```python
                    if sheet.title=='Table 5':
                            raiz=[]
                            Pedregosidad=[]

                            for col in sheet.iter_cols(min_row=1,values_only=True):

                                    # if the column is empy then write none in the roots and stoniness
                                    if col[0]==None:
                                            d_rp.update({'Raices':[None]})
                                            d_rp.update({'Pedregosidad':[None]})
                                            df_rp=pd.DataFrame(d_rp)

                                            pass
                                    # If the column is not empty then
                                    elif col[0]=='Horizonte/ Capa' or col[0]=='Horizonte o capa'or
col[0]=='Horizonte':
                                            if col[1]== None:
                                                    pass

                                            # include the column in the data frame
                                            elif '\n' in str(col[1]):
                                                    columna=col[1].split('\n')
                                                    print('Horizonte/ Capa', columna)

                                                    # if there are white spaces then
                                                    if ' ' in columna[1]:
                                                            # select the first value before the split
character and extract the information
                                                            colum=columna[1].split(' ')
                                                            colum.insert(0,columna[0])
                                                            colum.insert(0,'Horizonte/ Capa')
                                                            print('Horizonte', colum)
                                                            d_rp.update({'Horizonte/ Capa':colum[1:]})
                                                    else:
                                                            # extract the information
                                                            col=[x for x in columna]
                                                            col.insert(0,'Horizonte/ Capa')
                                                            print('Horizonte', col)
                                                            d_rp.update({'Horizonte/ Capa':col[1:]})
                                            else:
```

```
478                                                 d_rp.update({'Horizonte/ Capa':list(col[1:])})
479

480                                       # if the column is Caracteristicas descritas
481                                       elif 'Características'in col[0]:
482

483                                 if col[1] == None:
484                                       pass
485

486                             else:
487                                       # makes a list of the column values and for each
488     position it applies regular expressions
489                                       lista=list(col[1:])
490

491                                       for i in lista:
492                                             i=str(i)
493                                             # to extract the roots
494                                             raices=
495     re.findall(r"(?<=[R,r]a[i,í]ces)[^\;,\,]+",i)
496

497                                             # reclassify in 1 or 0
498                                             if len(raices)!=0:
499                                                   raiz.append('1')
500                                             else:
501                                                   raiz.append('0')
502

503                                             # Extract pedregosidad
504

505           pedrego=re.findall(r"(?<=[F,f]ragmentos)[^\;]+",i)
506                                             print(pedrego)
507

508                                             # extract the abundance of the rocks using
509     regular expressions
510                                             if len(pedrego)!=0:
511

512           ninguno=re.findall(r"(?<=[N,n]ingun[o,a])[^\;]+",pedrego[0])
513                                                   mpoco=re.findall(r"(?<=[M,m]uy
514     poc[o,a])[^\;]+",pedrego[0])
515

516           comun=re.findall(r"(?<=com[ú,u]n)[^\;]+",pedrego[0])
```

```
517
518             much=re.findall(r"(?<=much[o,a])[^\;]+",pedrego[0])
519
520             abundante=re.findall(r"(?<=abundante)[^\;]+",pedrego[0])
521
522             dominante=re.findall(r"(?<=dominante)[^\;]+",pedrego[0])
523
524
525
526                                                  # reclass the rock abundance
527     values in numbers
528                                            if len(ninguno)==0:
529                                                Pedregosidad.append('0')
530                                            elif len(mpoco)!=0:
531                                                Pedregosidad.append('1')
532                                            elif len(comun)!=0:
533
534     Pedregosidad.append('3.5')
535                                            elif len(much)!=0:
536                                                print('piedra')
537                                                Pedregosidad.append('10')
538                                            elif len(abundante)!=0:
539                                                Pedregosidad.append('10')
540                                            elif len(dominante)!=0:
541                                                Pedregosidad.append('60')
542
543                                        else:
544                                                Pedregosidad.append('0')
545
546                                # paste all the information in the dictionary
547                                d_rp.update({
548                                        'Raices':raiz,
549                                        'Pedregosidad':Pedregosidad})
550                                print(d_rp)
551
552                                # paste all in the dataframe if not write the error
553                                try:
554                                        df_rp=pd.DataFrame(d_rp)
555                                except Exception as e:
556                                        df_rp=pd.DataFrame({'Horizonte/ Capa':[None]})
```

```python
                                                    log=open('descripcion_errores.txt','a')
                                                    log.write('error {} en {}'.format(e,sheet.title))
                                                    log.close()




                                    # Extract info from table 7
                            elif sheet.title=='Table 7': # physical description
                                    sheet.delete_rows(2)# drop problematic row
                                    print('\n\tTABLA 7:')
                                    for i, col in enumerate(sheet.iter_cols(min_row=1,values_only=True)):
                                            print(col)

                                            # check the column 'Horizonte/ Capa' and extract the information
                                            if col[0]=='Horizonte/ Capa':
                                                    if col==('Horizonte/ Capa',):
                                                            pass
                                                    elif '\n' in str(col[1]):
                                                            columna=col[1].split('\n')
                                                            print('Horizonte/ Capa', columna)

                                                            if ' ' in columna[1]:

                                                                    colum=columna[1].split(' ')
                                                                    colum.insert(0,columna[0])
                                                                    colum.insert(0,'Horizonte/ Capa')
                                                                    print('Horizonte', colum)
                                                                    d2.update({'Horizonte/ Capa':colum[1:]})
                                                            else:

                                                                    col=[x for x in columna]
                                                                    col.insert(0,'Horizonte/ Capa')
                                                                    print('Horizonte', col)
                                                                    d2.update({'Horizonte/ Capa':col[1:]})
                                            else:
                                                    d2.update({'Horizonte/ Capa':list(col[1:])})
```

```
597
598                                    # extract the different features from the table based on the information
599    of the 'Horizonte/ Capa' field
600                                    extract_info('Profundidad (cm)',col,'Horizonte/ Capa',d2,d2)
601                                    extract_info('Arena',col,'Horizonte/ Capa',d2,d2)
602                                    extract_info('Limo',col,'Horizonte/ Capa',d2,d2)
603                                    extract_info('Arcilla',col,'Horizonte/ Capa',d2,d2)
604                                    #extract_info('Clase textural',col,'Horizonte/ Capa',d2,d2)
605                                    extract_info('Da (g/cm',col,'Horizonte/ Capa',d2,d2)
606                                    extract_info('Porosidad (%)',col,'Horizonte/ Capa',d2,d2)
607                                    extract_info('CC',col,'Horizonte/ Capa',d2,d2)
608                                    extract_info('PMP',col,'Horizonte/ Capa',d2,d2)
609
610
611                                    # create dataframe df2 from dictionary d2 this will hold the
612    information of the Horizons
613                                    try:
614                                            df2=pd.DataFrame(d2)
615                                            df2.insert(0,'ID_Nac',perfil)
616                                    except Exception as e:
617                                            df2=pd.DataFrame({'Horizonte/ Capa':[None]})
618                                            log=open('descripcion_errores.txt','a')
619                                            log.write('error {} en {}'.format(e,sheet.title))
620                                            log.close()
621
622
623
624                             print(d2)
625
626
627               # Chemical composition
628               elif sheet.title=='Table 9':
629                       sheet.delete_rows(2)
630                       print('\n\tTABLA 9:')
631                       for col in sheet.iter_cols(min_row=1,values_only=True):
632                               print(col)
633
634
635                               # Extract information from the field 'Horizonte/ Capa'
636                               if col[0]=='Horizonte/ Capa':
```

```python
                            if col==('Horizonte/ Capa',):
                                pass
                        elif '\n' in str(col[1]):
                                columna=col[1].split('\n')
                                print('Horizonte/ Capa', columna)

                                g=[x for x in columna]


                                for k, i in enumerate(columna):
                                    if r" " in i:
                                        s=i.split(' ')#

                                        s=s[::-1] # reverse the list
                                        g.pop(k) # pop the element that are
joined
                                        for j in s:
                                            g.insert(k,j) # we insert
the separated value in the copy of the list, in the value where the joined data was
                                    else:
                                        pass

                                colum=g
                                colum.insert(0,'Horizonte/ Capa')
                                print('Horizonte', colum)
                                d3.update({'Horizonte/ Capa':colum[1:]})


                        else:
                                d3.update({'Horizonte/ Capa':list(col[1:])})




                        # extract the features from the table based on a reference field 'Horizonte/
Capa'

                    extract_info('pH', col,'Horizonte/ Capa',d3,d3)
                    extract_info('N',col,'Horizonte/ Capa',d3,d3)
                    extract_info('P',col,'Horizonte/ Capa',d3,d3)
```

```
677                    extract_info('K',col,'Horizonte/ Capa',d3,d3)
678                    extract_info('Ca',col,'Horizonte/ Capa',d3,d3)
679                    extract_info('Mg',col,'Horizonte/ Capa',d3,d3)
680                    extract_info('S',col,'Horizonte/ Capa',d3,d3)
681                    extract_info('Zn',col,'Horizonte/ Capa',d3,d3)
682                    extract_info('Cu',col,'Horizonte/ Capa',d3,d3)
683                    extract_info('Fe',col,'Horizonte/ Capa',d3,d3)
684                    extract_info('Mn',col,'Horizonte/ Capa',d3,d3)
685                    extract_info('B',col,'Horizonte/ Capa',d3,d3)
686
687
688
689
690            # try to match long of the elements in the dataframe and paste to df3
691            try:
692                    igualar_longitud(d3,'Horizonte/ Capa')
693                    df3 = pd.DataFrame(d3)
694            except Exception as e:
695                    df3 = pd.DataFrame({'Horizonte/ Capa':[None]})
696                    log = open('descripcion_errores.txt','a')
697                    log.write('\nArchivo: {} {}.\n Error {}'.format(file,sheet.title,e))
698                    log.close()
699                    print(df3)
700
701
702
703
704
705        # Second table of chemical composition
706        elif sheet.title == 'Table 10':
707                print('\n\tTABLA 10:')
708                for col in sheet.iter_cols(min_row=1,values_only=True):
709                        print(col)
710
711                        # extract info from the Horizon field
712                        if col[0] =='Horizonte/ Capa':
713                                if col[1] == None:
714                                        pass
715                                elif '\n' in str(col[1]):
716                                        columna=col[1].split('\n')
```

```
717                                              print('Horizonte/ Capa', columna)

719                                              g=[x for x in columna]


722                                              for k, i in enumerate(columna):
723                                                      if r" " in i:
724                                                          s=i.split(' ')#

726                                                          s=s[::-1] # reverse the list
727                                                          g.pop(k) # extraxct the element of
728  the lsit that is joint
729                                                          for j in s:
730                                                              g.insert(k,j) #  insert the
731  separated value in the copy of the list in the value where the data was together
732                                                      else:
733                                                          pass

735                                              colum=g
736                                              colum.insert(0,'Horizonte/ Capa')
737                                              print('Horizonte', colum)
738                                              d4.update({'Horizonte/ Capa':colum[1:]})




742                                      else:
743                                              d4.update({'Horizonte/ Capa':list(col[1:])})


746                                      # extract information from each of the columnso of interest in the
747  layer using as reference the dictionary 4
748                                      extract_info('CE\n(ds/m)',col,'Horizonte/ Capa',d4,d4,'(ds/m)')
749                                      extract_info('MO (%)',col,'Horizonte/ Capa',d4,d_ref=d4)
750                                      extract_info('CO (%)',col,'Horizonte/ Capa',d4,d_ref=d4)
751                                      extract_info('Nitrógeno Total (%)',col,'Horizonte/ Capa',d4,d_ref=d4)
752                                      extract_info('C / N',col,'Horizonte/ Capa',d4,d_ref=d4)
753                                      extract_info('Ca/Mg',col,'Horizonte/ Capa',d4,d_ref=d4)
754                                      extract_info('Mg/K',col,'Horizonte/ Capa',d4,d_ref=d4)
755                                      extract_info('[Ca+Mg]/K',col,'Horizonte/ Capa',d4,d_ref=d4)
756                                      extract_info('Al+H',col,'Horizonte/ Capa',d4,d_ref=d4)
```

```
757
758
759                                    # try convert the dictionary into a dataframe
760                                    try:
761                                            igualar_longitud(d4,'Horizonte/ Capa')
762                                            df4=pd.DataFrame(d4)
763
764                                    # otherwise convert rise a log error file
765                                    except Exception as e:
766                                            df4=pd.DataFrame({'Horizonte/ Capa':[None]})
767                                            log=open('descripcion_errores.txt','a')
768                                            log.write('\nArchivo: {} {}.\n Error
769      {}'.format(file,sheet.title,e))
770                                            log.close()
771
772
773                            # Third table of checmial composition
774                            # this one qas morestable so the columns could be extracted directly
775                            elif sheet.title=='Table 11':
776                                    sheet.delete_rows(2)
777                                    print('\n\tTABLA 11:')
778                                    for col in sheet.iter_cols(min_row=1,values_only=True):
779                                            print(col)
780                                            if col[0]=='Horizonte/ Capa':
781                                                    d5.update({'Horizonte/ Capa':list(col[1:])})
782                                            if col[0]=='Na':
783                                                    d5.update({'Na_t9':list(col[1:])})
784                                            elif col[0]=='K':
785                                                    d5.update({'K_t9':list(col[1:])})
786                                            elif col[0]=='Ca':
787                                                    d5.update({'Ca_t9':list(col[1:])})
788                                            elif col[0]=='Mg':
789                                                    d5.update({'Mg_t9':list(col[1:])})
790                                            elif col[0]=='Suma Bases':
791                                                    d5.update({'Suma Bases':list(col[1:])})
792                                            try:
793                                                    if 'CIC' in col[0]:
794                                                            d5.update({'CIC':list(col[1:])})
795                                                    elif 'SB' in col[0]:
796                                                            d5.update({'SB(%)':list(col[1:])})
```

```python
                                    except Exception as e:
                                        if col[0]=='CIC':
                                            d5.update({'CIC':list(col[1:])})
                                        elif col[0]=='SB':
                                            d5.update({'SB(%)':list(col[1:])})

                                # convert into a dataframe
                                try:
                                        df5=pd.DataFrame(d5)
                                # rise a log exception
                                except Exception as e:
                                        df5=pd.DataFrame({'Horizonte/ Capa':[None]})
                                        log=open('descripcion_errores.txt','a')
                                        log.write('\nArchivo: {} {}.\nError
{}'.format(file,sheet.title,e))
                                        log.close()

                        # table for saline chemical determinations
                        elif sheet.title=='Table 12':
                                sheet.delete_rows(2)
                                print('\n\tTABLA 12:')
                                # values are extracted directly from the column values and converted into a
directionary, then they are turned into a dataframe
                                for col in sheet.iter_cols(min_row=1,values_only=True):
                                        print(col)
                                        if col[0]=='Horizonte/ Capa':
                                                d6.update({'Horizonte/ Capa':list(col[1:])})
                                        if col[0]=='pH':
                                                d6.update({'pH_sal':list(col[1:])})
                                        if col[0]=='Na':
                                                d6.update({'Na_sal':list(col[1:])})
                                        elif col[0]=='K':
                                                d6.update({'K_sal':list(col[1:])})
                                        elif col[0]=='Ca':
                                                d6.update({'Ca_sal':list(col[1:])})
                                        elif col[0]=='Mg':
                                                d6.update({'Mg_sal':list(col[1:])})
                                        elif col[0]=='Suma':
                                                d6.update({'Suma_Bases_sal':list(col[1:])})
                                        elif col[0]=='Cl':
```

```python
                                        d6.update({'Cl':list(col[1:])})
                                elif col[0]=='SB(%)':
                                        d6.update({'SB(%)':list(col[1:])})
                                elif col[0]=='CO 3':
                                        d6.update({'CO3':list(col[1:])})
                                elif col[0]=='SO 4':
                                        d6.update({'SO4':list(col[1:])})
                                elif col[0]=='RAS':
                                        d6.update({'RAS':list(col[1:])})
                                elif col[0]=='PSI':
                                        d6.update({'PSI':list(col[1:])})
                                try:
                                        if 'C.E'in col[0]:
                                                d6.update({'CE_sal':list(col[1:])})
                                except Exception as e:
                                        if col[0]=='C.E':
                                                d6.update({'CE_sal':list(col[1:])})


                                # convert into data frame or
                                try:
                                        df6=pd.DataFrame(d6)
                                except Exception as e:
                                        df6=pd.DataFrame({'Horizonte/ Capa':[None]})
                                        log=open('descripcion_errores.txt','a')
                                        log.write('\nArchivo: {} {}.\n Error
{}'.format(file,sheet.title,e))
                                        log.close()


        # Merge the data frame with the horizon information from the file (df2) with the dataframe the
whole file (dff2)
        # the horizons obtained in the df2 will be the reference field for joining the other dataframes
(df3,df4,df5,df6,df_rp)
        print(d2,d3,d4,d5,d6)
        try:
                t=df2.merge(dff2, on='Horizonte/ Capa', how='left')
        except Exception as e:
                log=open('descripcion_errores.txt','a')
                log.write('\nArchivo: {} {}.\n Error {}'.format(file,sheet.title,e))
```

```
877                        log.close()
878                        #print(t)
879            # Merge the data frames that correspond to the information extracted from each layer.
880            # If the length of the dataframe is zero, joining that dataframe is discarded.
881            for i in [df3,df4,df5,df6,df_rp]:
882                    print(i)
883                    if len(i)==0:
884                            pass
885                    else:
886                            try:
887                                    t=t.merge(i, on='Horizonte/ Capa', how='left')
888                            except Exception as e:
889                                    t=''
890                                    log=open('descripcion_errores.txt','a')
891                                    log.write('\nArchivo: {}.\nFallo en merge: {}\n'.format(file,e))
892                                    log.close()
893
894
895        # merge all the dataframes into the daframe with all the fields.
896        # the dataframe for exporting the data
897        try:
898                df_final=df_final.append(t)
899                log_completos=open('estado_archivos_completos.txt','a')
900                log_completos.write('Archivo: {}\n tabla 9 \n{}\ntabla10\n{}\n'.format(file,df3,df4))
901                log_completos.close()
902        except Exception as e:
903                log=open('descripcion_errores.txt','a')
904                log.write('\nArchivo {}.Error en append: {}\n'.format(file,e))
905                log.close()
906                log_resumen=open('archivos_problematicos.txt','a')
907                log_resumen.write('Archivo: {}\n'.format(file))
908                log_resumen.close()
909
910
911
912
913  # Create a excel file from the final dataframe
914  df_final.to_excel("BD_IEE_Merge_1.xlsx")  # doctest: +SKIP
915
916
```