Trabajo de Fin de Grado

Grado en Ingeniería en Tecnologías Industriales

Desarrollo de un software para el cálculo y el análisis de parámetros de rugosidad

ANEXO

Autor: Francisco Vera Pérez
Director: Alejandro Domínguez Fernández

Convocatoria: Septiembre 2018

Escola Tècnica Superior
d'Enginyeria Industrial de Barcelona

Resumen

En este anexo se encuentra el código que conforma el software para el cálculo de rugosidad. Debido a la extensión del programa, no ha sido posible incluirlo en la memoria.

El programa se ha estructurado en dos scripts, el script *Rugosidad_programa.py* y el *Funciones.py*, el primero contiene el grueso del programa, con la interfaz gráfica y todas las funciones, mientras que en el segundo se encuentran funciones destacadas del programa, como la del filtrado, la cual posee un compilador que ayuda a reducir el tiempo de cálculo.

Índice

1. Código del programa

1.1. Rugosidad_programa.py

```
from tkinter import *  # Carga módulo tk (widgets estándar)
from tkinter import ttk # Carga ttk (para widgets nuevos 8.5+)
from tkinter import filedialog
import tkinter
##################################
import os
import numpy as np
import matplotlib.pyplot as plt
import math
import time
import shutil
import funciones
import numba
import scipy
import pandas as pd
from reportlab.pdfgen import canvas
from reportlab.lib.utils import ImageReader
from reportlab.lib.pagesizes import landscape, letter, A4
from matplotlib.backends.backend_pdf import PdfPages
from reportlab.graphics import renderPDF
from svglib.svglib import svg2rlg

def funcion_base():
        starting_point = time.time()
        x , y , z = pon_en_lista()
        altura_nueva = regresion_lineal(x, y, z)
        if comboTipoEstudio.current() == 0:
                y2 = linea_media(x, altura_nueva)
                xprim, yprim, xrugos, yrugos, xond, yond = filtrado(x, y2)
                yprim_media = linea_media(xprim, yprim)
                #yprim_media = regresion_lineal(xprim, yprim, z)
                yrugos_media = linea_media(xrugos, yrugos)
                #yrugos_media = regresion_lineal(xrugos,yrugos,z)
                yond_media = linea_media(xond, yond)
                #yond_media = regresion_lineal(xond, yond, z)

        listPp,listWp,listRp,listRp1,listPv,listWv,listRv,listRv1,listRz,listPt,listWt,listRt,listPa,listWa,listRa,listPq,listWq,listRq,listPsk,listWsk,listRsk,listPkv,listWkv,listRkv,listPsm,listWsm,listRsm =
parametros(yprim_media,yrugos_media,yond_media,xprim,xrugos,xond)
                xabbotP,yabbotP, xabbotR,yabbotR, xabbotW,yabbotW =
curva_abbot(listPp,listWp,listRp1,listPv,listWv,listRv1,listPt,listWt,listRt,xprim,xrugos,xond,yprim_media,yrugos_media,yond_media)
                xampP, xampR, xampW, yampP, yampR, yampW =
distribucion_amplitud(xabbotP,xabbotR,xabbotW,yabbotP,yabbotR,yabbotW)

        lPpk,lPk,lPvk,lPmr,lMr1P,lMr2P,lA1P,lA2P,lRpk,lRk,lRvk,lRmr,lMr1R,lMr2R,lA1R,lA2R,lWpk,lWk,lWvk,lWmr,lMr1W,lMr2W,lA1W,lA2W =
```

```
parametros_abbot(xabbotP,xabbotR,xabbotW,yabbotP,yabbotR,yabbotW,yprim_media,yrugos_media,yond_media,listPt,listWt,
listRt)
                    crea_directorios(listaficheros,lprimarios,londulacion,lrugosidad,lparametros, lnombres1,lnombres)

        muestra_resultados(xprim,xrugos,xond,yprim_media,yrugos_media,yond_media,lprimarios,lrugosidad,londulacion,lis
tPp,listWp,listRp,listPv,listWv,listRv,listRz,listPt,listWt,listRt,listPa,listWa,listRa,listPq,listWq,listRq,listPsk,listWsk,listRsk,listPkv
,listWkv,listRkv,listPsm,listWsm,listRsm,xabbotP,yabbotP,xabbotR,yabbotR,xabbotW,yabbotW,xampP,xampR,xampW,yampP,
yampR,yampW,lPpk,lPk,lPvk,lPmr,lMr1P,lMr2P,lA1P,lA2P,lRpk,lRk,lRvk,lRmr,lMr1R,lMr2R,lA1R,lA2R,lWpk,lWk,lWvk,lWmr,lM
r1W,lMr2W,lA1W,lA2W)

        grafica_resultados_report(xprim,yprim_media,xond,yond_media,xrugos,yrugos_media,xabbotR,yabbotR,lnombres,lis
tRp,listRv,listRz,listRt,listRa,listRq,listRsk,listRkv,listRsm,lRpk,lRk,lRvk,lRmr,lMr1R,lMr2R,lA1R,lA2R)
                    renombrar_y_mover_carpeta()
                    elapsed_time = time.time () - starting_point
                    print('Tiempo Total:',elapsed_time)
        if comboTipoEstudio.current() == 1:
                    xprim, yprim, zprim, xrugos, yrugos, zrugos, xond, yond, zond = filtrado3D(x, y, altura_nueva)
                    #zprim_media = regresion_lineal(xprim, yprim, zprim)
                    listSpP, listSvP, listSzP, listSaP, listSqP, listSskP, listSkvP = parametros3D(xprim, yprim, zprim)
                    xabbotP, yabbotP = curva_abbot3D(listSpP, listSvP, listSzP, zprim)
                    lSpkP,lSkP,lSvkP,lSmrP,lMr1P,lMr2P,lA1P,lA2P = parametros_abbot3D(xabbotP,yabbotP,zprim,listSzP)
                    crea_directorios(listaficheros,lprimarios,londulacion,lrugosidad,lparametros, lnombres1,lnombres)
                    muestra_resultados3D(xprim, yprim, zprim, listSpP, listSvP, listSzP, listSaP, listSqP, listSskP, listSkvP,
xabbotP, yabbotP, lSpkP, lSkP, lSvkP, lSmrP, lMr1P, lMr2P, lA1P, lA2P)
                    grafica_resultados_report3D(xprim, yprim, zprim, listSpP, listSvP, listSzP, listSaP, listSqP, listSskP,
listSkvP, xabbotP, yabbotP, lSpkP, lSkP, lSvkP, lSmrP, lMr1P, lMr2P, lA1P, lA2P)
                    renombrar_y_mover_carpeta()
                    elapsed_time = time.time () - starting_point
                    print('Tiempo Total:',elapsed_time)

listaficheros = []
lprimarios = []
londulacion = []
lrugosidad = []
lparametros = []
lnombres1 = []
lnombres = []

def abrir():
        raiz.filename =  filedialog.askopenfilenames(initialdir = "", title = "Select file",filetypes = (('.txt','*txt'),('.prf','*.prf'),("All
files","*.*")))
        for fichero in raiz.filename:
                    if len(fichero)>0 and fichero not in listaficheros:
                                listaficheros.append(fichero)
                                fichero = fichero[::-1]
                                indice = fichero.find('/')
                                nombre1 = fichero[indice:]
                                nombre1 = nombre1[::-1]
                                punto = fichero.find('.')
                                nombre = fichero[punto+1:indice]
                                nombre = nombre[::-1]
                                nombreprim = nombre + '_primario.txt'
                                nombreond = nombre + '_ondulacion.txt'
                                nombrerugos = nombre + '_rugosidad.txt'
                                nombreparametros = nombre + '_parametros.txt'
```

```
                                lprimarios.append(nombreprim)
                                londulacion.append(nombreond)
                                lrugosidad.append(nombrerugos)
                                lparametros.append(nombreparametros)
                                lnombres1.append(nombre1)
                                lnombres.append(nombre)
        if len(listaficheros) > 0:
                win = Tk()
                win.geometry('250x130')
                win.configure(bg = 'beige')
                lblFicheroExito = Label(win, text = 'El fichero se ha añadido con éxito').place(x = 30 , y = 20)
                lblFicheroContador = Label(win, text = 'Hay '+str(len(listaficheros))+' ficheros para su estudio').place(x = 38
, y = 50)
                botonCerrar = Button(win, text = 'Cerrar', command = win.destroy).place(x = 100 , y = 80)
        print(listaficheros)
        print(lprimarios)
        print(londulacion)
        print(lrugosidad)
        print(lparametros)
        print(lnombres1)
        print(lnombres)
def directorio_salida():
        direc_salida = filedialog.askdirectory()
        print(direc_salida)

def pon_en_lista():
        starting_point = time.time()
        lista_de_x = []
        lista_de_y = []
        lista_de_z = []
        resolucion = ResolucionX.get()
        if ',' in resolucion:
                resolucion = resolucion.replace(',','.')
        resolucion = float(resolucion)
        if comboTipoEstudio.current() == 0:
                for ficheroo in listaficheros:
                        if len(ficheroo) > 0:
                                x = []
                                y = []
                                fichero = open(ficheroo, 'r')
                                if SelecEjes.get() == 2:
                                        resolucion = ResolucionX.get()
                                        count = 0
                                        countResolucion = 0
                                        for linea in fichero:
                                                if count >= LineasEncabezado.get():
                                                        linea = linea.replace(',','.')
                                                        if linea[-1] == '\n':
                                                                linea = linea[:-1]
                                                                linea = float(linea)
                                                        linea = float(linea)
                                                        if comboUnidadesZ.current()    == 0:
                                                                linea = linea*1000
                                                        y.append(linea)
                                                        countResolucion = countResolucion + resolucion
```

```
                                        x.append(countResolucion)
                                count = count + 1
                        else:
                                count = 0

                                for linea in fichero:
                                        if count >= LineasEncabezado.get():
                                                if ElementoSeparador.get() == ',':
                                                        pass
                                                else:
                                                        linea = linea.replace(',','.')
                                                if linea[-1] == '\n':
                                                        linea = linea[:-1]
                                                separador = ElementoSeparador.get()
                                                if len(separador) == 0:
                                                        pass
                                                if len(separador) == 0:
                                                        linea = linea.split()
                                                else:
                                                        linea = linea.split(separador)
                                                linea[0] = float(linea[0])
                                                linea[1] = float(linea[1])
                                                if comboUnidadesX.current() == 1:
                                                        linea[0] = linea[0]      / 1000
                                                if comboUnidadesZ.current()    == 0:
                                                        linea[1] = linea[1]*1000
                                                x.append(linea[0])
                                                y.append(linea[1])
                                        count = count + 1
                        lista_de_x.append(x)
                        lista_de_y.append(y)
                        fichero.close()
if comboTipoEstudio.current() == 1:
        if SelecEjes.get() == 3:
                for ficheroo in listaficheros:
                        if len(ficheroo) > 0:
                                x = []
                                y = []
                                z = []
                                fichero = open(ficheroo, 'r')
                                count = 0

                                for linea in fichero:
                                        if count >= LineasEncabezado.get():
                                                if ElementoSeparador.get() == ',':
                                                        pass
                                                else:
                                                        linea = linea.replace(',','.')
                                                if linea[-1] == '\n':
                                                        linea = linea[:-1]
                                                separador = ElementoSeparador.get()
                                                if len(separador) == 0:
                                                        pass
                                                if len(separador) == 0:
                                                        linea = linea.split()
```

```
                                                    else:
                                                        linea = linea.split(separador)
                                                    linea[0] = float(linea[0])
                                                    linea[1] = float(linea[1])
                                                    linea[2] = float(linea[2])
                                                    if comboUnidadesX.current() == 1:
                                                        linea[0] = linea[0]      / 1000
                                                    if comboUnidadesY.current() == 1:
                                                        linea[1] = linea[1]      / 1000
                                                    if comboUnidadesZ.current()    == 0:
                                                        linea[2] = linea[2]*1000
                                                    x.append(linea[0])
                                                    y.append(linea[1])
                                                    z.append(linea[2])
                                                count = count + 1
                                            lista_de_x.append(x)
                                            lista_de_y.append(y)
                                            lista_de_z.append(z)
                                            fichero.close()
        elapsed_time = time.time () - starting_point
        print('tiempo pon en lista:',elapsed_time)

        return (lista_de_x,lista_de_y,lista_de_z)

def regresion_lineal(x, y, z):
        starting_point = time.time()
        if comboTipoEstudio.current() == 0:
            altura_nueva = []
            for i in range (len(y)):
                n = len(y[i])
                xi = np.array(x[i])
                yi = np.array(y[i])
                sumx = sum(xi)
                sumy = sum(yi)
                sumx2 = sum(xi*xi)
                sumy2 = sum(yi*yi)
                sumxy = sum (xi*yi)
                promx = sumx/n
                promy = sumy/n
                m = (sumx*sumy-n*sumxy)/(sumx**2-n*sumx2)
                b = promy-m*promx
                yparcial = []
                contador = 0
                for elemento in y[i]:
                    yres = y[i][contador] - (m*x[i][contador] + b)
                    yparcial.append(yres)
                    contador = contador + 1
                altura_nueva.append(yparcial)
        if comboTipoEstudio.current() == 1:
            altura_nueva = []
            posit = []
            neg = []
            for i in range(len(y)):
                n = len(x[i])
                xi = np.array(x[i])
```

```
                yi = np.array(y[i])
                zi = np.array(z[i])
                sumx = sum(xi)
                sumy = sum(yi)
                sumz = sum(zi)
                sumx2 = sum(xi*xi)
                sumy2 = sum(yi*yi)
                sumz2 = sum(zi*zi)
                sumxy = sum(xi*yi)
                sumxz = sum(xi*zi)
                sumyz = sum(yi*zi)
                a = (sumxy*sumyz-sumy2*sumxz)/(sumx2*sumy2-sumxy*sumxy)
                b = (sumxy*sumxz-sumx2*sumyz)/(sumx2*sumy2-sumxy*sumxy)
                c = (sumz-a*sumx-b*sumy)/len(x[i])
                zparcial = []
                contador = 0
                for elemento in z[i]:
                        zres = z[i][contador] - (a*x[i][contador] + b*y[i][contador] + c)
                        zparcial.append(zres)
                        contador = contador + 1
                altura_nueva.append(zparcial)
        elapsed_time = time.time () - starting_point
        print('tiempo regresion:',elapsed_time)
        return (altura_nueva)

def linea_media(x, y):
        starting_point = time.time()
        ymedia = []
        for i in range (len(y)):
                Longitud = float(x[i][-1])
                ResolX = x[i][1] - x[i][0]
                ResolX = ResolX*1000
                NumPuntosX = len(x[i])
                ListaPicos = []
                ListaValles = []
                for elemento in y[i]:
                        if elemento > 0:
                                ListaPicos.append(elemento)
                        else:
                                elemento = abs(elemento)
                                ListaValles.append(elemento)
                ListaPicos1 = np.array(ListaPicos)
                ListaValles1 = np.array(ListaValles)
                SumaPicos = sum(ListaPicos1)
                SumaValles = sum(ListaValles1)
                if SumaPicos == 0:
                        Sp = min(ListaValles)
                        Sv = max(ListaValles)
                elif SumaValles == 0:
                        Sp = max(ListaPicos)
                        Sv = min(ListaPicos)
                else:
                        Sp = max (ListaPicos)
                        Sv = max (ListaValles)
                Sz = Sp + Sv
```

```
                    CoefPasoBusq = 500
                    PasoBusq = Sz/CoefPasoBusq
                    ErrBusq = (PasoBusq/2) + (1/CoefPasoBusq)
                    yinter = y[i]
                    yinicial = []
                    starting_point1 = time.time()
                    yinicial, SumaPicos, SumaValles = funciones.bucle_ondulacion(CoefPasoBusq, NumPuntosX, ErrBusq,
yinter, yinicial,SumaPicos, SumaValles, PasoBusq)
                    if len(yinicial) == 0:
                            ymedia.append(yinter)
                    else:
                            ymedia.append(yinicial)
            elapsed_time = time.time () - starting_point
            print('tiempo linea media:',elapsed_time)
            return (ymedia)

def filtrado(x,y):
            starting_point = time.time()
            if comboTipoFiltro.current() == 0:
                    yprimario = []
                    xprimario = []
                    yrugosidad = []
                    xrugosidad = []
                    yondulacion = []
                    xondulacion = []
                    lambdaS = PasaBajos.get()
                    if ',' in lambdaS:
                            lambdaS = lambdaS.replace(',','.')
                    lambdaS = float(lambdaS)
                    lambdaC = PasaAltos.get()
                    if ',' in lambdaC:
                            lambdaC = lambdaC.replace(',','.')
                    lambdaC = float(lambdaC)
                    CutOffC = lambdaC #0,8
                    CutOffS = lambdaS #0,0025
                    for i in range(len(y)):
                            listaPrimario = []
                            #Crear vector Gauss para cada fichero
                            if SelecPrimario.get() == 0 or 1:
                                    ConsFiltrado = 0.4697186393*CutOffS
                                    ResolX = (x[i][1]-x[i][0])*1000
                                    Longitud = x[i][-1]
                                    if Longitud < 2*CutOffS:
                                            xprimario.append(x[i])
                                            listaPrimario = y[i]
                                            NumRecEntP = 0
                                            NumRecSalP = 0
                                    elif ResolX >= CutOffS:
                                            xprimario.append(x[i])
                                            listaPrimario = y[i]
                                            NumRecEntP = 0
                                            NumRecSalP = 0
                                    else:
                                            NumValorsEnt = len(y[i])
                                            NumValorsGauss = math.trunc(CutOffS*1000/(2*ResolX))+1
```

```
                                        listv_Gauss = []
                                        for num in range(NumValorsGauss):
                                                xGauss = num*ResolX/1000
                                                v_Gauss = (math.exp(-
math.pi*((xGauss/ConsFiltrado)**2)))/ConsFiltrado
                                                listv_Gauss.append(v_Gauss)
                                        listv_Gauss1 = np.array(listv_Gauss)
                                        AreaGauss = sum(listv_Gauss1)*2
                                        listv_Gauss_norm = []
                                        for elemento in listv_Gauss:
                                                elemento = elemento / AreaGauss
                                                listv_Gauss_norm.append(elemento)
                                        longFiltro = len(listv_Gauss_norm)
                                        #Dimensionar salida patra cada fichero
                                        NumCutOffs = math.trunc(((NumValorsEnt-
2*NumValorsGauss)*ResolX/(CutOffS*1000)))
                                        NumValSalP = math.trunc(NumCutOffs*CutOffS*1000/ResolX)
                                        NumRecEntP = math.trunc((NumValorsEnt-NumValSalP-
2*NumValorsGauss)/2)
                                        NumRecSalP = NumValorsEnt-NumValSalP-2*NumValorsGauss-
NumRecEntP
                                        #Filtrado
                                        ValAFiltrar = NumRecEntP + NumValorsGauss
                                        listaPrimario = funciones.filtro_gaussiano(ValAFiltrar, NumValSalP,
NumValorsGauss, y[i], listv_Gauss_norm)
                                        x_prim = x[i][NumRecEntP+longFiltro:len(x[i])-NumRecSalP-longFiltro+1]
                                        xprimario.append(x_prim)
                        yprimario.append(listaPrimario)
                        listaOndulacion = []
                        if SelecOndulacion.get() ==  0 or 1:
                                ConsFiltrado = 0.4697186393*CutOffC
                                ResolX = (x[i][1]-x[i][0])*1000
                                Longitud = x[i][-1]
                                if Longitud < 2*CutOffC:
                                        x_ond_rug = x[i]
                                        xondulacion.append(x[i])
                                        listaOndulacion = y[i]
                                        NumRecEntO = 0
                                        NumRecSalO = 0
                                elif ResolX >= CutOffC:
                                        x_ond_rug = x[i]
                                        xondulacion.append(x[i])
                                        listaOndulacion = y[i]
                                        NumRecEntO = 0
                                        NumRecSalO = 0
                                else:
                                        NumValorsEnt = len(y[i])
                                        NumValorsGauss = math.trunc(CutOffC*1000/(2*ResolX))+1
                                        listv_Gauss = []
                                        for num in range(NumValorsGauss):
                                                xGauss = num*ResolX/1000
                                                v_Gauss = (math.exp(-
math.pi*((xGauss/ConsFiltrado)**2)))/ConsFiltrado
                                                listv_Gauss.append(v_Gauss)
                                        listv_Gauss1 = np.array(listv_Gauss)
```

```
                                        AreaGauss = sum(listv_Gauss1)*2
                                        listv_Gauss_norm = []
                                        for elemento in listv_Gauss:
                                                elemento = elemento / AreaGauss
                                                listv_Gauss_norm.append(elemento)
                                        longFiltro = len(listv_Gauss_norm)
                                        #Dimensionar salida para cada fichero
                                        NumCutOffs = math.trunc(((NumValorsEnt-
2*NumValorsGauss)*ResolX/(CutOffC*1000)))
                                        NumValSalO = math.trunc(NumCutOffs*CutOffC*1000/ResolX)
                                        NumRecEntO = math.trunc((NumValorsEnt-NumValSalO-
2*NumValorsGauss)/2)
                                        NumRecSalO = NumValorsEnt-NumValSalO-2*NumValorsGauss-
NumRecEntO
                                        #Filtrado
                                        ValAFiltrar = NumRecEntO + NumValorsGauss
                                        listaOndulacion = funciones.filtro_gaussiano(ValAFiltrar, NumValSalO,
NumValorsGauss, y[i], listv_Gauss_norm)
                                        x_ond_rug = x[i][NumRecEntO+longFiltro:len(x[i])-NumRecSalO-
longFiltro+1]
                                        xondulacion.append(x_ond_rug)
                                yondulacion.append(listaOndulacion)
                                listaRugosidad = []
                                if SelecRugosidad.get() == 0 or 1:
                                        if Longitud < 2*CutOffC or ResolX>=CutOffC:
                                                NumValorsEnt = len(y[i])
                                                NumValorsGauss = math.trunc(CutOffC*1000/(2*ResolX))+1
                                                listv_Gauss = []
                                                for num in range(NumValorsGauss):
                                                        xGauss = num*ResolX/1000
                                                        v_Gauss = (math.exp(-
math.pi*((xGauss/ConsFiltrado)**2)))/ConsFiltrado
                                                        listv_Gauss.append(v_Gauss)
                                                listv_Gauss1 = np.array(listv_Gauss)
                                                AreaGauss = sum(listv_Gauss1)*2
                                                listv_Gauss_norm = []
                                                for elemento in listv_Gauss:
                                                        elemento = elemento / AreaGauss
                                                        listv_Gauss_norm.append(elemento)
                                                longFiltro = len(listv_Gauss_norm)
                                        listaOndulacion = listaOndulacion
                                        listaPrimario = listaPrimario[(longFiltro+NumRecEntO-
NumRecEntP):len(listaPrimario)-((longFiltro-1+NumRecSalO)+NumRecSalP)]
                                        for num in range(len(listaPrimario)):
                                                terminoRugosidad = listaPrimario[num] - listaOndulacion[num]
                                                listaRugosidad.append(terminoRugosidad)
                                yrugosidad.append(listaRugosidad)
                                xrugosidad.append(x_ond_rug)
        elapsed_time = time.time () - starting_point
        print('tiempo filtrado 2D:',elapsed_time)
        return (xprimario, yprimario, xrugosidad, yrugosidad, xondulacion, yondulacion)

def filtrado3D(x,y,z):
        if comboTipoFiltro.current() == 0:
                yprimario = []
```

```
                xprimario = []
                zprimario = []
                yrugosidad = []
                xrugosidad = []
                zrugosidad = []
                yondulacion = []
                xondulacion = []
                zondulacion = []
                lambdaS = PasaBajos.get()
                if ',' in lambdaS:
                        lambdaS = lambdaS.replace(',','.')
                lambdaS = float(lambdaS)
                lambdaC = PasaAltos.get()
                if ',' in lambdaC:
                        lambdaC = lambdaC.replace(',','.')
                lambdaC = float(lambdaC)
                CutOffC = lambdaC #0,8
                CutOffS = lambdaS #0,0025
                for i in range(len(y)):
                        if SelecPrimario.get() == 0 or 1:
                                listaPrimario = []
                                listaPrimario = z[i]
                                x_prim = x[i]
                                y_prim = y[i]
                        zprimario.append(listaPrimario)
                        xprimario.append(x_prim)
                        yprimario.append(y_prim)
                        if SelecOndulacion.get() == 0 or 1:
                                pass
        return (xprimario, yprimario, zprimario, xrugosidad, yrugosidad, zrugosidad, xondulacion, yondulacion, zondulacion)

def parametros(yprim, yrugos, yond, xprim, xrugos, xond):
        starting_point = time.time()

        listPp = []
        listWp = []
        listRp = []
        listRp1 = []

        listPv = []
        listWv = []
        listRv = []
        listRv1 = []

        listRz = []

        listPt = []
        listWt = []
        listRt = []

        listPa = []
        listWa = []
        listRa = []

        listPq = []
```

```
listWq = []
listRq = []

listPsk = []
listWsk = []
listRsk = []

listPkv = []
listWkv = []
listRkv = []

listPsm = []
listWsm = []
listRsm = []

for i in range(len(yprim)):
        yprim[i] = np.array(yprim[i])
        Pp = max(yprim[i])
        listPp.append(Pp)
        Pv = min(yprim[i])
        listPv.append(Pv)
        Pt = max(yprim[i]) + abs(min(yprim[i]))
        listPt.append(Pt)
        Pa = sum(abs(yprim[i]))/len(yprim[i])
        listPa.append(Pa)
        Pq = math.sqrt(sum(yprim[i]*yprim[i])/len(yprim[i]))
        listPq.append(Pq)
        Psk = sum(yprim[i]*yprim[i]*yprim[i])/(Pq*Pq*Pq*len(yprim[i]))
        listPsk.append(Psk)
        Pkv = sum(yprim[i]*yprim[i]*yprim[i]*yprim[i])/(Pq*Pq*Pq*Pq*len(yprim[i]))
        listPkv.append(Pkv)
        por_ciento_Pz = 0.1 * Pt
        xPsm = funciones.rsm(xprim[i], yprim[i], por_ciento_Pz)
        if len (xPsm) == 0:
                Psm = ''
                listPsm.append(Psm)
        else:
                Psm = np.mean(xPsm)
                Psm = Psm * 1000
                listPsm.append(Psm)

for i in range(len(yond)):
        yond[i] = np.array(yond[i])
        Wp = max(yond[i])
        listWp.append(Wp)
        Wv = min(yond[i])
        listWv.append(Wv)
        Wt = max(yond[i]) + abs(min(yond[i]))
        listWt.append(Wt)
        Wa = sum(abs(yond[i]))/len(yond[i])
        listWa.append(Wa)
        Wq = math.sqrt(sum(yond[i]*yond[i])/len(yond[i]))
        listWq.append(Wq)
        Wsk = sum(yond[i]*yond[i]*yond[i])/(Wq*Wq*Wq*len(yond[i]))
        listWsk.append(Psk)
```

```
                Wkv = sum(yond[i]*yond[i]*yond[i]*yond[i])/(Wq*Wq*Wq*Wq*len(yond[i]))
                listWkv.append(Wkv)
                por_ciento_Wz = 0.1 * Wt
                xWsm = funciones.rsm(xond[i], yond[i], por_ciento_Wz)
                if len(xWsm) == 0:
                        Wsm = ''
                        listWsm.append(Wsm)
                else:
                        Wsm = np.mean(xWsm)
                        Wsm = Wsm * 1000
                        listWsm.append(Wsm)

        for i in range(len(yrugos)):
                yrugos[i] = np.array(yrugos[i])
                num = int(len(yrugos[i])/5)
                leval = []
                levalx = []

                rugos1x = xrugos[i][:num]
                levalx.append(rugos1x)
                rugos1 = yrugos[i][:num]
                leval.append(rugos1)

                rugos2x = xrugos[i][num:2*num]
                levalx.append(rugos2x)
                rugos2 = yrugos[i][num:2*num]
                leval.append(rugos2)

                rugos3x = xrugos[i][2*num:3*num]
                levalx.append(rugos3x)
                rugos3 = yrugos[i][2*num:3*num]
                leval.append(rugos3)

                rugos4x = xrugos[i][3*num:4*num]
                levalx.append(rugos4x)
                rugos4 = yrugos[i][3*num:4*num]
                leval.append(rugos4)

                rugos5x = xrugos[i][4*num:]
                levalx.append(rugos5x)
                rugos5 = yrugos[i][4*num:]
                leval.append(rugos5)


                Rp1 = max(yrugos[i])
                listRp1.append(Rp1)
                Rv1 = min(yrugos[i])
                listRv1.append(Rv1)
                lRp = []
                for elem in leval:
                        Rp = max(elem)
                        lRp.append(Rp)
                Rp = np.mean(lRp)
                listRp.append(Rp)
                lRv = []
```

```
for elem in leval:
        Rv = min(elem)
        lRv.append(Rv)
Rv = np.mean(lRv)
listRv.append(Rv)
lRz = []
for elem in leval:
        Rz = max(elem) + abs(min(elem))
        lRz.append(Rz)
Rz = np.mean(lRz)
listRz.append(Rz)
Rt = max(yrugos[i]) + abs(min(yrugos[i]))
listRt.append(Rt)
lRa = []
for elem in leval:
        Ra = sum(abs(elem))/len(elem)
        lRa.append(Ra)
Ra = np.mean(lRa)
listRa.append(Ra)
lRq = []
for elem in leval:
        Rq = math.sqrt(sum(elem*elem)/len(elem))
        lRq.append(Rq)
Rq = np.mean(lRq)
listRq.append(Rq)
lRsk = []
for elem in leval:
        Rsk = sum(elem*elem*elem)/(Rq*Rq*Rq*len(elem))
        lRsk.append(Rsk)
Rsk = np.mean(lRsk)
listRsk.append(Rsk)
lRkv = []
for elem in leval:
        Rkv = sum(elem*elem*elem*elem)/(Rq*Rq*Rq*Rq*len(elem))
        lRkv.append(Rkv)
Rkv = np.mean(lRkv)
listRkv.append(Rkv)
lRsm = []
for elem in range(len(levalx)):
        por_ciento_Rz = 0.1 * lRz[elem]
        xRsm = funciones.rsm(levalx[elem], leval[elem], por_ciento_Rz)
        if len(xRsm) > 1:
                Rsm = np.mean(xRsm)
                lRsm.append(Rsm)
        else:
                pass
if len(lRsm) == 0:
        por_ciento_Rz = 0.1 * Rz
        xRsm = funciones.rsm(xrugos[i], yrugos[i], por_ciento_Rz)
        if len(xRsm) == 0:
                Rsm = ''
                listRsm.append(Rsm)
        else:
                Rsm = np.mean(xRsm)
                Rsm = Rsm * 1000
```

```
                                        listRsm.append(Rsm)
                        else:
                                Rsm = np.mean(lRsm)
                                Rsm = Rsm * 1000
                                listRsm.append(Rsm)
        elapsed_time = time.time () - starting_point
        print('tiempo parametros:',elapsed_time)
        return(listPp, listWp, listRp, listRp1, listPv, listWv, listRv, listRv1, listRz, listPt, listWt, listRt, listPa, listWa,
listRa,listPq, listWq, listRq, listPsk, listWsk, listRsk,listPkv, listWkv, listRkv, listPsm, listWsm, listRsm)

def parametros3D(xprim, yprim, zprim):
        starting_point = time.time()

        listSpP = []
        listSvP = []
        listSzP = []
        listSaP = []
        listSqP = []
        listSskP = []
        listSkvP = []

        for i in range(len(xprim)):
                zprim[i] = np.array(zprim[i])
                SpP = max(zprim[i])
                listSpP.append(SpP)
                SvP = min(zprim[i])
                listSvP.append(SvP)
                SzP = max(zprim[i]) + abs(min(zprim[i]))
                listSzP.append(SzP)
                SaP = sum(abs(zprim[i]))/len(zprim[i])
                listSaP.append(SaP)
                SqP = math.sqrt(sum(zprim[i]*zprim[i])/len(zprim[i]))
                listSqP.append(SqP)
                SskP = sum(zprim[i]*zprim[i]*zprim[i])/(SqP*SqP*SqP*len(zprim[i]))
                listSskP.append(SskP)
                SkvP = sum(zprim[i]*zprim[i]*zprim[i]*zprim[i])/(SqP*SqP*SqP*SqP*len(zprim[i]))
                listSkvP.append(SkvP)
        elapsed_time = time.time () - starting_point
        print('tiempo parametros 3D:',elapsed_time)
        return (listSpP, listSvP, listSzP, listSaP, listSqP, listSskP, listSkvP)


def curva_abbot(listPp,listWp,listRp1,listPv,listWv,listRv1,listPt,listWt,listRt,xprim,xrugos,xond,yprim,yrugos,yond):
        starting_point = time.time()
        xabbotP = []
        yabbotP = []
        xabbotR = []
        yabbotR = []
        xabbotW = []
        yabbotW = []
        if SelecAbbotPrimario.get() == 1:
                xabbotP = []
                yabbotP = []
                for i in range(len(xprim)):
                        #PasoBusq = listPt[i]/1000
```

```
                PasoBusq = 0.1
                NumPuntosX = len(xprim[i])
                Paso_actual = listPp[i]
                EjeY = listPt[i]
                xinter = []
                yinter = []
                while Paso_actual >= listPv[i]:
                        suma = 0
                        for num in yprim[i]:
                                if num >= Paso_actual:
                                        suma = suma + 1
                        xi = (suma/NumPuntosX)*100
                        xinter.append(xi)
                        yinter.append(EjeY)
                        EjeY = EjeY - PasoBusq
                        Paso_actual = Paso_actual - PasoBusq
                xabbotP.append(xinter)
                yabbotP.append(yinter)

if SelecAbbotRugosidad.get() == 1:
        xabbotR = []
        yabbotR = []
        for i in range(len(xrugos)):
                #PasoBusq = listRt[i]/1000
                PasoBusq = 0.1
                NumPuntosX = len(xrugos[i])
                Paso_actual = listRp1[i]
                EjeY = listRt[i]
                xinter = []
                yinter = []
                while Paso_actual >= listRv1[i]:
                        suma = 0
                        for num in yrugos[i]:
                                if num >= Paso_actual:
                                        suma = suma + 1
                        xi = (suma/NumPuntosX)*100
                        xinter.append(xi)
                        yinter.append(EjeY)
                        EjeY = EjeY - PasoBusq
                        Paso_actual = Paso_actual - PasoBusq
                xabbotR.append(xinter)
                yabbotR.append(yinter)
if SelecAbbotOndulacion.get() == 1:
        xabbotW = []
        yabbotW = []
        for i in range(len(xond)):
                #PasoBusq = listWt[i]/1000
                PasoBusq = 0.1
                NumPuntosX = len(xond[i])
                Paso_actual = listWp[i]
                EjeY = listWt[i]
                xinter = []
                yinter = []
                while Paso_actual >= listWv[i]:
                        suma = 0
```

```
                                        for num in yond[i]:
                                                if num >= Paso_actual:
                                                        suma = suma + 1
                                        xi = (suma/NumPuntosX)*100
                                        xinter.append(xi)
                                        yinter.append(EjeY)
                                        EjeY = EjeY - PasoBusq
                                        Paso_actual = Paso_actual - PasoBusq
                                xabbotW.append(xinter)
                                yabbotW.append(yinter)
        elapsed_time = time.time () - starting_point
        print('tiempo curva de Abbott:',elapsed_time)
        return (xabbotP,yabbotP,xabbotR,yabbotR,xabbotW,yabbotW)

def curva_abbot3D(listSpP, listSvP, listSzP, zprim):
        starting_point = time.time()
        if SelecAbbotPrimario.get() == 0 or 1:
                xabbotP = []
                yabbotP = []
                for i in range(len(zprim)):
                        #PasoBusq = listPt[i]/1000
                        PasoBusq = 0.1
                        NumPuntosX = len(zprim[i])
                        Paso_actual = listSpP[i]
                        EjeY = listSzP[i]
                        xinter = []
                        yinter = []
                        while Paso_actual >= listSvP[i]:
                                suma = 0
                                for num in zprim[i]:
                                        if num >= Paso_actual:
                                                suma = suma + 1
                                xi = (suma/NumPuntosX)*100
                                xinter.append(xi)
                                yinter.append(EjeY)
                                EjeY = EjeY - PasoBusq
                                Paso_actual = Paso_actual - PasoBusq
                        xabbotP.append(xinter)
                        yabbotP.append(yinter)
        elapsed_time = time.time () - starting_point
        print('tiempo curva de Abbott 3D:',elapsed_time)
        return (xabbotP,yabbotP)

def distribucion_amplitud(xabbotP,xabbotR,xabbotW,yabbotP,yabbotR,yabbotW):
        starting_point = time.time()
        xampP = []
        yampP = []
        xampR = []
        yampR = []
        xampW = []
        yampW = []
        if SelecAbbotPrimario.get() == 1:
                xampP = []
                yampP = []
                for i in range(len(xabbotP)):
```

```
                    ultimo = len(xabbotP[i])
                    xinter= []
                    for idx in range(1, len(xabbotP[i])-1):
                        xi = ((xabbotP[i][idx+1]-xabbotP[i][idx-1])/2) * 100
                        xinter.append(xi)
                    xampP.append(xinter)
                    yampP.append(yabbotP[i][1:-1])
    if SelecAbbotRugosidad.get() == 1:
        xampR = []
        yampR = []
        for i in range(len(xabbotR)):
            ultimo = len(xabbotR[i])
            xinter= []
            for idx in range(1, len(xabbotR[i])-1):
                xi = ((xabbotR[i][idx+1]-xabbotR[i][idx-1])/2) * 100
                xinter.append(xi)
            xampR.append(xinter)
            yampR.append(yabbotR[i][1:-1])
    if SelecAbbotOndulacion.get() == 1:
        xampW = []
        yampW = []
        for i in range(len(xabbotW)):
            ultimo = len(xabbotW[i])
            xinter= []
            for idx in range(1, len(xabbotW[i])-1):
                xi = ((xabbotW[i][idx+1]-xabbotW[i][idx-1])/2) * 100
                xinter.append(xi)
            xampW.append(xinter)
            yampW.append(yabbotW[i][1:-1])
    elapsed_time = time.time () - starting_point
    print('tiempo distribucion Abbott:',elapsed_time)
    return(xampP, xampR, xampW, yampP, yampR, yampW)

def
parametros_abbot(xabbotP,xabbotR,xabbotW,yabbotP,yabbotR,yabbotW,yprim_media,yrugos_media,yond_media,listPt,listWt,
listRt):
    starting_point = time.time()

    lPpk = []
    lPk = []
    lPvk = []
    lPmr = []
    lMr1P = []
    lMr2P = []
    lA1P = []
    lA2P = []

    lRpk = []
    lRk = []
    lRvk = []
    lRmr = []
    lMr1R = []
    lMr2R = []
    lA1R = []
    lA2R = []
```

```
        lWpk = []
        lWk = []
        lWvk = []
        lWmr = []
        lMr1W = []
        lMr2W = []
        lA1W = []
        lA2W = []
        if SelecAbbotPrimario.get() == 1:
                for i in range(len(xabbotP)):
                        pasoy = yabbotP[i][1] - yabbotP[i][0]
                        lm = []
                        le = []
                        la = []
                        cont = 0
                        for e in range(len(xabbotP[i])):
                                if xabbotP[i][e] >= 40:
                                        for a in range(len(xabbotP[i])):
                                                if (xabbotP[i][e] - xabbotP[i][a]) >= 40 and (xabbotP[i][e] -
xabbotP[i][a+1]) < 40:
                                                        m = (yabbotP[i][e] - yabbotP[i][a]) / (xabbotP[i][e] -
xabbotP[i][a])
                                                        if len(lm) == 0:
                                                                lm.append(m)
                                                if len(lm) >0 and m < lm[0]:
                                                        break
                                                elif len(lm) > 0 and m >= lm[0]:
                                                        lm[0] = m
                                                        if len(le) == 0:
                                                                le.append(e)
                                                        else:
                                                                le[0] = e
                                                        if len(la) == 0:
                                                                la.append(a)
                                                        else:
                                                                la[0] = a
                        h1 = lm[0] * (xabbotP[i][0] - xabbotP[i][la[0]]) + yabbotP[i][la[0]]
                        h2 = lm[0] * (xabbotP[i][-1] - xabbotP[i][la[0]]) + yabbotP[i][la[0]]
                        Pk = h1 - h2
                        for e in range(len(yabbotP[i])):
                                if yabbotP[i][e] <= h1:
                                        Mr1 = xabbotP[i][e-1]
                                        indiceh1 = e-1
                                        break
                        for e in range(len(yabbotP[i])):
                                indiceh2 = 0
                                Mr2 = 0
                                if yabbotP[i][e] <= h2:
                                        Mr2 = xabbotP[i][e-1]
                                        indiceh2 = e-1
                                        break
                        maximo = max(yprim_media[i])
                        minimo = abs(min(yprim_media[i]))
                        por_maximo = 1*maximo
```

```
                por_minimo = 1*minimo
                listapicos = []
                listapicos_resol = []
                listapicos_suma = []
                listavalles = []
                listavalles_resol = []
                listavalles_suma = []
                for e in range(len(xabbotP[i])):
                        if e <= indiceh1 and yabbotP[i][e] <= (minimo + por_maximo):
                                listapicos.append(yabbotP[i][e] - yabbotP[i][indiceh1])
                                resol = xabbotP[i][e+1] - xabbotP[i][e]
                                listapicos_resol.append(resol)
                for e in range(len(xabbotP[i])-1):
                        if e >= indiceh2 and yabbotP[i][e] >= (listPt[i] - maximo - por_minimo):
                                #listavalles.append(yabbotP[i][e] - yabbotP[i][-1])
                                listavalles.append(yabbotP[i][indiceh2] - yabbotP[i][e])
                                resol = xabbotP[i][e+1] - xabbotP[i][e]
                                listavalles_resol.append(resol)
                for e in range(len(listapicos)):
                        valor = listapicos[e] * listapicos_resol[e]
                        listapicos_suma.append(valor)
                for e in range(len(listavalles)):
                        valor = listavalles[e] * listavalles_resol[e]
                        listavalles_suma.append(valor)
                A1 = sum(np.array(listapicos_suma))
                A2 = sum(np.array(listavalles_suma))
                Ppk = (2*A1)/Mr1
                Pvk = (2*A2)/(100 - Mr2)
                Pmr = ''
                for e in range(len(xabbotP[i])):
                        if listPt[i] > 1:
                                if yabbotP[i][e] < (listPt[i] - 1):
                                        Pmr = xabbotP[i][e-1]
                                        break
                        else:
                                Pmr = ''
                lPpk.append(Ppk)
                lPk.append(Pk)
                lPvk.append(Pvk)
                lPmr.append(Pmr)
                lMr1P.append(Mr1)
                lMr2P.append(Mr2)
                lA1P.append(A1)
                lA2P.append(A2)
if SelecAbbotRugosidad.get() == 1:
        for i in range(len(xabbotR)):
                pasoy = yabbotR[i][1] - yabbotR[i][0]
                lm = []
                le = []
                la = []
                cont = 0
                for e in range(len(xabbotR[i])):
                        if xabbotR[i][e] >= 40:
                                for a in range(len(xabbotR[i])):
                                        if (xabbotR[i][e] - xabbotR[i][a]) >= 40 and (xabbotR[i][e] -
```

```
xabbotR[i][a+1]) < 40:
                                                m = (yabbotR[i][e] - yabbotR[i][a]) / (xabbotR[i][e] -
xabbotR[i][a])
                                                if len(lm) == 0:
                                                        lm.append(m)
                                        if len(lm) >0 and m < lm[0]:
                                                break
                                        elif len(lm) > 0 and m >= lm[0]:
                                                lm[0] = m
                                                if len(le) == 0:
                                                        le.append(e)
                                                else:
                                                        le[0] = e
                                                if len(la) == 0:
                                                        la.append(a)
                                                else:
                                                        la[0] = a
                        h1 = lm[0] * (xabbotR[i][0] - xabbotR[i][la[0]]) + yabbotR[i][la[0]]
                        h2 = lm[0] * (xabbotR[i][-1] - xabbotR[i][la[0]]) + yabbotR[i][la[0]]
                        Rk = h1 - h2
                        for e in range(len(yabbotR[i])):
                                if yabbotR[i][e] <= h1:
                                        Mr1 = xabbotR[i][e-1]
                                        indiceh1 = e-1
                                        break
                        for e in range(len(yabbotR[i])):
                                if yabbotR[i][e] <= h2:
                                        Mr2 = xabbotR[i][e-1]
                                        indiceh2 = e-1
                                        break
                        maximo = max(yrugos_media[i])
                        minimo = abs(min(yrugos_media[i]))
                        por_maximo = 1*maximo
                        por_minimo = 1*minimo
                        listapicos = []
                        listapicos_resol = []
                        listapicos_suma = []
                        listavalles = []
                        listavalles_resol = []
                        listavalles_suma = []
                        for e in range(len(xabbotR[i])):
                                if e <= indiceh1 and yabbotR[i][e] <= (minimo + por_maximo):
                                        listapicos.append(yabbotR[i][e] - yabbotR[i][indiceh1])
                                        resol = xabbotR[i][e+1] - xabbotR[i][e]
                                        listapicos_resol.append(resol)
                        for e in range(len(xabbotR[i])-1):
                                if e >= indiceh2 and yabbotR[i][e] >= (listRt[i] - maximo - por_minimo):
                                        listavalles.append(yabbotR[i][e] - yabbotR[i][-1])
                                        resol = xabbotR[i][e+1] - xabbotR[i][e]
                                        listavalles_resol.append(resol)
                        for e in range(len(listapicos)):
                                valor = listapicos[e] * listapicos_resol[e]
                                listapicos_suma.append(valor)
                        for e in range(len(listavalles)):
                                valor = listavalles[e] * listavalles_resol[e]
```

```
                        listavalles_suma.append(valor)
                A1 = sum(np.array(listapicos_suma))
                A2 = sum(np.array(listavalles_suma))
                Rpk = (2*A1)/Mr1
                Rvk = (2*A2)/(100 - Mr2)
                Rmr = ''
                for e in range(len(xabbotR[i])):
                        if listRt[i] > 1:
                                if yabbotR[i][e] < (listRt[i] - 1):
                                        Rmr = xabbotR[i][e-1]
                                        break
                        else:
                                Rmr = ''
                lRpk.append(Rpk)
                lRk.append(Rk)
                lRvk.append(Rvk)
                lRmr.append(Rmr)
                lMr1R.append(Mr1)
                lMr2R.append(Mr2)
                lA1R.append(A1)
                lA2R.append(A2)
        if SelecAbbotOndulacion.get() == 1:
                for i in range(len(xabbotW)):
                        pasoy = yabbotW[i][1] - yabbotW[i][0]
                        lm = []
                        le = []
                        la = []
                        cont = 0
                        for e in range(len(xabbotW[i])):
                                if xabbotW[i][e] >= 40:
                                        for a in range(len(xabbotW[i])):
                                                if (xabbotW[i][e] - xabbotW[i][a]) >= 40 and (xabbotW[i][e] -
xabbotW[i][a+1]) < 40:
                                                        m = (yabbotW[i][e] - yabbotW[i][a]) / (xabbotW[i][e] -
xabbotW[i][a])
                                                        if len(lm) == 0:
                                                                lm.append(m)
                                                if len(lm) >0 and m < lm[0]:
                                                        break
                                                elif len(lm) > 0 and m >= lm[0]:
                                                        lm[0] = m
                                                        if len(le) == 0:
                                                                le.append(e)
                                                        else:
                                                                le[0] = e
                                                        if len(la) == 0:
                                                                la.append(a)
                                                        else:
                                                                la[0] = a
                        h1 = lm[0] * (xabbotW[i][0] - xabbotW[i][la[0]]) + yabbotW[i][la[0]]
                        h2 = lm[0] * (xabbotW[i][-1] - xabbotW[i][la[0]]) + yabbotW[i][la[0]]
                        Wk = h1 - h2
                        for e in range(len(yabbotW[i])):
                                if yabbotW[i][e] <= h1:
                                        Mr1 = xabbotW[i][e-1]
```

```
                                indiceh1 = e-1
                                break
                for e in range(len(yabbotW[i])):
                        if yabbotW[i][e] <= h2:
                                Mr2 = xabbotW[i][e-1]
                                indiceh2 = e-1
                                break
                maximo = max(yond_media[i])
                minimo = abs(min(yond_media[i]))
                por_maximo = 1*maximo
                por_minimo = 1*minimo
                listapicos = []
                listapicos_resol = []
                listapicos_suma = []
                listavalles = []
                listavalles_resol = []
                listavalles_suma = []
                for e in range(len(xabbotW[i])):
                        if e <= indiceh1 and yabbotW[i][e] <= (minimo + por_maximo):
                                listapicos.append(yabbotW[i][e] - yabbotW[i][indiceh1])
                                resol = xabbotW[i][e+1] - xabbotW[i][e]
                                listapicos_resol.append(resol)
                for e in range(len(xabbotW[i])-1):
                        if e >= indiceh2 and yabbotW[i][e] >= (listWt[i] - maximo - por_minimo):
                                listavalles.append(yabbotW[i][e] - yabbotW[i][-1])
                                resol = xabbotW[i][e+1] - xabbotW[i][e]
                                listavalles_resol.append(resol)
                for e in range(len(listapicos)):
                        valor = listapicos[e] * listapicos_resol[e]
                        listapicos_suma.append(valor)
                for e in range(len(listavalles)):
                        valor = listavalles[e] * listavalles_resol[e]
                        listavalles_suma.append(valor)
                A1 = sum(np.array(listapicos_suma))
                A2 = sum(np.array(listavalles_suma))
                Wpk = (2*A1)/Mr1
                Wvk = (2*A2)/(100 - Mr2)
                Wmr = ''
                for e in range(len(xabbotW[i])):
                        if listWt[i] >= 1:
                                if yabbotW[i][e] < (listWt[i] - 1):
                                        Wmr = xabbotW[i][e-1]
                                        break
                        else:
                                Wmr = ''
                lWpk.append(Wpk)
                lWk.append(Wk)
                lWvk.append(Wvk)
                lWmr.append(Wmr)
                lMr1W.append(Mr1)
                lMr2W.append(Mr2)
                lA1W.append(A1)
                lA2W.append(A2)
elapsed_time = time.time () - starting_point
print('tiempo parametros Abbott:',elapsed_time)
```

```
	return
(lPpk,lPk,lPvk,lPmr,lMr1P,lMr2P,lA1P,lA2P,lRpk,lRk,lRvk,lRmr,lMr1R,lMr2R,lA1R,lA2R,lWpk,lWk,lWvk,lWmr,lMr1W,lMr2W,lA1
W,lA2W)

def parametros_abbot3D(xabbotP,yabbotP,zprim,listSzP):
	starting_point = time.time()

	lSpkP = []
	lSkP = []
	lSvkP = []
	lSmrP = []
	lMr1P = []
	lMr2P = []
	lA1P = []
	lA2P = []

	if SelecAbbotPrimario.get() == 0 or 1:
		for i in range(len(xabbotP)):
			pasoy = yabbotP[i][1] - yabbotP[i][0]
			lm = []
			le = []
			la = []
			cont = 0
			for e in range(len(xabbotP[i])):
				if xabbotP[i][e] >= 40:
					for a in range(len(xabbotP[i])):
						if (xabbotP[i][e] - xabbotP[i][a]) >= 40 and (xabbotP[i][e] -
xabbotP[i][a+1]) < 40:
							m = (yabbotP[i][e] - yabbotP[i][a]) / (xabbotP[i][e] -
xabbotP[i][a])
							if len(lm) == 0:
								lm.append(m)
						if len(lm) >0 and m < lm[0]:
							break
						elif len(lm) > 0 and m >= lm[0]:
							lm[0] = m
							if len(le) == 0:
								le.append(e)
							else:
								le[0] = e
							if len(la) == 0:
								la.append(a)
							else:
								la[0] = a
			h1 = lm[0] * (xabbotP[i][0] - xabbotP[i][la[0]]) + yabbotP[i][la[0]]
			h2 = lm[0] * (xabbotP[i][-1] - xabbotP[i][la[0]]) + yabbotP[i][la[0]]
			Sk = h1 - h2
			for e in range(len(yabbotP[i])):
				if yabbotP[i][e] <= h1:
					Mr1 = xabbotP[i][e-1]
					indiceh1 = e-1
					break
			for e in range(len(yabbotP[i])):
				if yabbotP[i][e] <= h2:
					Mr2 = xabbotP[i][e-1]
```

```
                                indiceh2 = e-1
                                break
                maximo = max(zprim[i])
                minimo = abs(min(zprim[i]))
                por_maximo = 1*maximo
                por_minimo = 1*minimo
                listapicos = []
                listapicos_resol = []
                listapicos_suma = []
                listavalles = []
                listavalles_resol = []
                listavalles_suma = []
                for e in range(len(xabbotP[i])):
                        if e <= indiceh1 and yabbotP[i][e] <= (minimo + por_maximo):
                                listapicos.append(yabbotP[i][e] - yabbotP[i][indiceh1])
                                resol = xabbotP[i][e+1] - xabbotP[i][e]
                                listapicos_resol.append(resol)
                for e in range(len(xabbotP[i])-1):
                        if e >= indiceh2 and yabbotP[i][e] >= (listSzP[i] - maximo - por_minimo):
                                listavalles.append(yabbotP[i][e] - yabbotP[i][-1])
                                resol = xabbotP[i][e+1] - xabbotP[i][e]
                                listavalles_resol.append(resol)
                for e in range(len(listapicos)):
                        valor = listapicos[e] * listapicos_resol[e]
                        listapicos_suma.append(valor)
                for e in range(len(listavalles)):
                        valor = listavalles[e] * listavalles_resol[e]
                        listavalles_suma.append(valor)
                A1 = sum(np.array(listapicos_suma))
                A2 = sum(np.array(listavalles_suma))
                Spk = (2*A1)/Mr1
                Svk = (2*A2)/(100 - Mr2)
                Smr = ''
                for e in range(len(xabbotP[i])):
                        if listSzP[i] > 1:
                                if yabbotP[i][e] < (listSzP[i] - 1):
                                        Smr = xabbotP[i][e-1]
                                        break
                        else:
                                Smr = ''
                lSpkP.append(Spk)
                lSkP.append(Sk)
                lSvkP.append(Svk)
                lSmrP.append(Smr)
                lMr1P.append(Mr1)
                lMr2P.append(Mr2)
                lA1P.append(A1)
                lA2P.append(A2)
        elapsed_time = time.time () - starting_point
        print('tiempo parametros Abbott 3D:',elapsed_time)
        return (lSpkP,lSkP,lSvkP,lSmrP,lMr1P,lMr2P,lA1P,lA2P)

def crea_directorios(listaficheros,lprimarios,londulacion,lrugosidad,lparametros,lnombres1,lnombres):
        starting_point = time.time()
        os.makedirs(lnombres1[0] + 'Resultados')
```

```
        for i in range (len(listaficheros)):
                os.makedirs(lnombres1[0] + 'Resultados' + '/' + lnombres[i])
                in_primario = lprimarios[i].find('.')
                lprimario = lprimarios[i][:in_primario]
                in_rug = lrugosidad[i].find('.')
                lrugos = lrugosidad[i][:in_rug]
                in_ond = londulacion[i].find('.')
                lond = londulacion[i][:in_ond]
                if comboTipoEstudio.current() == 0:
                        if SelecPrimario.get() == 1:
                                os.makedirs(lnombres1[0] + 'Resultados' + '/' + lnombres[i] + '/'+ 'Perfiles' + '/' +
lprimario)
                        if SelecRugosidad.get() == 0 or 1:
                                os.makedirs(lnombres1[0] + 'Resultados' + '/' + lnombres[i] + '/' + 'Perfiles' + '/' +
lrugos)
                        if SelecOndulacion.get() == 1:
                                os.makedirs(lnombres1[0] + 'Resultados' + '/' + lnombres[i] + '/' + 'Perfiles' + '/' + lond)
                        if SelecAbbotPrimario.get() == 1:
                                os.makedirs(lnombres1[0] + 'Resultados' + '/' + lnombres[i] + '/'+ 'Abbott' + '/' +
lprimario)
                        if SelecAbbotRugosidad.get() == 1:
                                os.makedirs(lnombres1[0] + 'Resultados' + '/' + lnombres[i] + '/' + 'Abbott' + '/' +
lrugos)
                        if SelecAbbotOndulacion.get() == 1:
                                os.makedirs(lnombres1[0] + 'Resultados' + '/' + lnombres[i] + '/' + 'Abbott' + '/' + lond)
                if comboTipoEstudio.current() == 1:
                        os.makedirs(lnombres1[0] + 'Resultados' + '/' + lnombres[i] + '/'+ 'Perfiles' + '/' + lprimario)
                        os.makedirs(lnombres1[0] + 'Resultados' + '/' + lnombres[i] + '/'+ 'Abbott' + '/' + lprimario)

        elapsed_time = time.time () - starting_point
        print('tiempo crea directorios:',elapsed_time)

def
muestra_resultados(xprim,xrugos,xond,yprim,yrugos,yond,lprimarios,lrugosidad,londulacion,listPp,listWp,listRp,listPv,listWv,list
Rv,listRz, listPt, listWt, listRt,listPa, listWa, listRa,listPq, listWq, listRq, listPsk, listWsk, listRsk,listPkv, listWkv, listRkv, listPsm,
listWsm, listRsm, xabbotP, yabbotP, xabbotR, yabbotR, xabbotW, yabbotW, xampP, xampR, xampW, yampP, yampR,
yampW,lPpk,lPk,lPvk,lPmr,lMr1P,lMr2P,lA1P,lA2P,lRpk,lRk,lRvk,lRmr,lMr1R,lMr2R,lA1R,lA2R,lWpk,lWk,lWvk,lWmr,lMr1W,lM
r2W,lA1W,lA2W):
        starting_point = time.time()
        if SelecPrimario.get() == 1:
                for i in range (len(yprim)):
                        fprim = open(lnombres[i] + '_parametros_primario.txt','w')
                        fprim.write('Fichero de procedencia:' + ' ' + lnombres[i]+'\n')
                        fprim.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
                        fprim.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                        fprim.write('Pp = ' + str(listPp[i])[:8] +' um'  + '\n')
                        fprim.write('Pv = ' + str(listPv[i])[:8] +' um' + '\n')
                        fprim.write('Pt = ' + str(listPt[i])[:8] +' um' + '\n')
                        fprim.write('Pa = ' + str(listPa[i])[:8] +' um' + '\n')
                        fprim.write('Pq = ' + str(listPq[i])[:8] +' um' + '\n')
                        fprim.write('Psk = ' + str(listPsk[i])[:8] +' um' + '\n')
                        fprim.write('Pku = ' + str(listPkv[i])[:8] +' um' + '\n')
                        fprim.write('Psm = ' + str(listPsm[i])[:8] +' um' + '\n')
                        fprim.close()
```

```
                                fprim1 = open(lnombres[i] + '_perfil_primario.txt','w')
                                fprim1.write('Fichero de procedencia:' + ' ' + lnombres[i]+'\n')
                                fprim1.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
                                fprim1.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                                count = 0
                                for num in yprim[i]:
                                        if comboUnidadesZSalida.current() == 0:
                                                num = num/1000
                                        num = str(num)
                                        if comboUnidadesXSalida.current() == 1:
                                                xprim[i][count] = (xprim[i][count])*1000
                                        xprim[i][count] = str(xprim[i][count])
                                        if SelecSeparadorDecimal.get() == 2:
                                                xprim[i][count] = xprim[i][count].replace('.',',')
                                                num = num.replace('.',',')
                                        if SelecEjesSalida.get() == 2:
                                                fprim1.write(num+'\n')
                                        else:
                                                fprim1.write(xprim[i][count]+ElementoSeparadorSalida.get()+num+'\n')
                                        count = count+1
                                fprim1.close()
                                shutil.move(lnombres[i] +
'_parametros_primario.txt',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Perfiles' + '/' +lnombres[i]+'_primario')
                                shutil.move(lnombres[i] +
'_perfil_primario.txt',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Perfiles' + '/' +lnombres[i]+'_primario')

        if SelecOndulacion.get() == 1:
                for i in range (len(yond)):
                                fond = open(lnombres[i] + '_parametros_ondulacion.txt','w')
                                fond.write('Fichero de procedencia:' + ' ' + lnombres[i]+'\n')
                                fond.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
                                if comboTipoFiltro.current() == 0:
                                        fond.write('Tipo de filtro:' + ' ' + 'Gaussiano'+';' + '  '+str(PasaAltos.get()) +'mm'+'\n')
                                fond.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                                fond.write('Wp = ' + str(listWp[i])[:8] +' um' + '\n')
                                fond.write('Wv = ' + str(listWv[i])[:8] +' um' + '\n')
                                fond.write('Wt = ' + str(listWt[i])[:8] +' um' + '\n')
                                fond.write('Wa = ' + str(listWa[i])[:8] +' um' + '\n')
                                fond.write('Wq = ' + str(listWq[i])[:8] +' um' + '\n')
                                fond.write('Wsk = ' + str(listWsk[i])[:8] +' um' + '\n')
                                fond.write('Wku = ' + str(listWkv[i])[:8] +' um' + '\n')
                                fond.write('Wsm = ' + str(listWsm[i])[:8] +' um' + '\n')
                                fond.close()
                                fond1 = open(lnombres[i] + '_perfil_ondulacion.txt','w')
                                fond1.write('Fichero de procedencia:' + ' ' + lnombres[i]+'\n')
                                fond1.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
                                if comboTipoFiltro.current() == 0:
                                        fond1.write('Tipo de filtro:' + ' ' + 'Gaussiano'+';' + '  '+str(PasaAltos.get()) +'mm'+'\n')
                                fond1.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                                count = 0
                                for num in yond[i]:
                                        if comboUnidadesZSalida.current() == 0:
```

```
                                        num = num/1000
                                num = str(num)
                                xond[i][count] = str(xond[i][count])
                                if SelecSeparadorDecimal.get() == 2:
                                        xond[i][count] = xond[i][count].replace('.',',')
                                        num = num.replace('.',',')
                                if SelecEjesSalida.get() == 2:
                                        fond1.write(num+'\n')
                                else:
                                        fond1.write(xond[i][count]+ElementoSeparadorSalida.get()+num+'\n')
                                count = count+1
                        fond1.close()
                        shutil.move(lnombres[i] +
'_parametros_ondulacion.txt',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Perfiles' + '/' +lnombres[i]+'_ondulacion')
                        shutil.move(lnombres[i] +
'_perfil_ondulacion.txt',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Perfiles' + '/' +lnombres[i]+'_ondulacion')

        if SelecRugosidad.get() == 1:
                for i in range (len(yrugos)):
                        frug = open(lnombres[i] + '_parametros_rugosidad.txt','w')
                        frug.write('Fichero de procedencia:' + ' ' + lnombres[i]+'\n')
                        frug.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
                        if comboTipoFiltro.current() == 0:
                                frug.write('Tipo de filtro:' + ' ' + 'Gaussiano'+';' + '  '+str(PasaAltos.get()) +'mm'+'\n')
                        frug.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                        frug.write('Rp = ' + str(listRp[i])[:8] +' um' + '\n')
                        frug.write('Rv = ' + str(listRv[i])[:8] +' um' + '\n')
                        frug.write('Rz = ' + str(listRz[i])[:8] +' um' + '\n')
                        frug.write('Rt = ' + str(listRt[i])[:8] +' um' + '\n')
                        frug.write('Ra = ' + str(listRa[i])[:8] +' um' + '\n')
                        frug.write('Rq = ' + str(listRq[i])[:8] +' um' + '\n')
                        frug.write('Rsk = ' + str(listRsk[i])[:8] +' um' + '\n')
                        frug.write('Rku = ' + str(listRkv[i])[:8] +' um' + '\n')
                        frug.write('Rsm = ' + str(listRsm[i])[:8] +' um' + '\n')
                        frug.close()
                        count = 0
                        frug1 = open(lnombres[i] + '_perfil_rugosidad.txt','w')
                        frug1.write('Fichero de procedencia:' + ' ' + lnombres[i]+'\n')
                        frug1.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
                        if comboTipoFiltro.current() == 0:
                                frug1.write('Tipo de filtro:' + ' ' + 'Gaussiano'+';' + '  '+str(PasaAltos.get()) +'mm'+'\n')
                        frug1.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                        for num in yrugos[i]:
                                if comboUnidadesZSalida.current() == 0:
                                        num = num/1000
                                num = str(num)
                                xrugos[i][count] = str(xrugos[i][count])
                                if SelecSeparadorDecimal.get() == 2:
                                        xrugos[i][count] = xrugos[i][count].replace('.',',')
                                        num = num.replace('.',',')
                                if SelecEjesSalida.get() == 2:
                                        frug1.write(num+'\n')
                                else:
```

```
                                        frug1.write(xrugos[i][count]+ElementoSeparadorSalida.get()+num+'\n')
                                count = count+1
                        frug1.close()
                        shutil.move(lnombres[i] +
'_parametros_rugosidad.txt',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Perfiles' + '/' +lnombres[i]+'_rugosidad')
                        shutil.move(lnombres[i] +
'_perfil_rugosidad.txt',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Perfiles' + '/' +lnombres[i]+'_rugosidad')

        if SelecAbbotPrimario.get() == 1:
                for i in range (len(yabbotP)):
                        fabbotP = open(lnombres[i] + '_parametros_primario_abbott.txt','w')
                        fabbotP.write('Fichero de procedencia:' + ' ' + lnombres[i]+'\n')
                        fabbotP.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
                        if comboTipoFiltro.current() == 0:
                                fabbotP.write('Tipo de filtro:' + ' ' + 'Gaussiano'+';' + '  '+str(PasaAltos.get()) +'mm'+'\n')
                        fabbotP.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                        fabbotP.write('Ppk = ' + str(lPpk[i])[:8] +' um' + '\n')
                        fabbotP.write('Pvk = ' + str(lPvk[i])[:8] +' um' + '\n')
                        fabbotP.write('Pk = ' + str(lPk[i])[:8] +' um' + '\n')
                        fabbotP.write('Mr1 = ' + str(lMr1P[i])[:8] +' %' + '\n')
                        fabbotP.write('Mr2 = ' + str(lMr2P[i])[:8] +' %' + '\n')
                        fabbotP.write('A1 = ' + str(lA1P[i])[:8] +' area' + '\n')
                        fabbotP.write('A2 = ' + str(lA2P[i])[:8] +' area' + '\n')
                        fabbotP.write('Pmr = ' + str(lPmr[i])[:8] +' um' + '\n')
                        count = 0
                        fabbotP1 = open(lnombres[i] + '_perfil_primario_abbott.txt','w')
                        fabbotP1.write('Fichero de procedencia:' + ' ' + lnombres[i]+'\n')
                        fabbotP1.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
                        if comboTipoFiltro.current() == 0:
                                fabbotP1.write('Tipo de filtro:' + ' ' + 'Gaussiano'+';' + '  '+str(PasaAltos.get())
+'mm'+'\n')
                        fabbotP1.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                        for num in yabbotP[i]:
                                if SelecSeparadorDecimal.get() == 2:
                                        num = str(num)
                                        xabbotP[i][count] = str(xabbotP[i][count])
                                        xabbotP[i][count].replace('.',',')
                                        num = num.replace('.',',')

        fabbotP1.write(xabbotP[i][count]+ElementoSeparadorSalida.get()+num+'\n')
                                else:

        fabbotP1.write(str(xabbotP[i][count])+ElementoSeparadorSalida.get()+str(num)+'\n')
                                count = count+1
                        fabbotP.close()
                        fabbotP1.close()
                        shutil.move(lnombres[i] +
'_parametros_primario_abbott.txt',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Abbott' + '/' +lnombres[i]+'_primario')
                        shutil.move(lnombres[i] +
'_perfil_primario_abbott.txt',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Abbott' + '/' +lnombres[i]+'_primario')

        if SelecAbbotOndulacion.get() == 1:
                for i in range (len(yabbotW)):
```

```
                                fabbotW = open(lnombres[i] + '_parametros_ondulacion_abbott.txt','w')
                                fabbotW.write('Fichero de procedencia:' + ' ' + lnombres[i]+'\n')
                                fabbotW.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
                                if comboTipoFiltro.current() == 0:
                                        fabbotW.write('Tipo de filtro:' + ' ' + 'Gaussiano'+';' + '  '+str(PasaAltos.get())
+'mm'+'\n')
                                fabbotW.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                                fabbotW.write('Wpk = ' + str(lWpk[i])[:8] +' um' + '\n')
                                fabbotW.write('Wvk = ' + str(lWvk[i])[:8] +' um' + '\n')
                                fabbotW.write('Wk = ' + str(lWk[i])[:8] +' um' + '\n')
                                fabbotW.write('Mr1 = ' + str(lMr1W[i])[:8] +' %' + '\n')
                                fabbotW.write('Mr2 = ' + str(lMr2W[i])[:8] +' %' + '\n')
                                fabbotW.write('A1 = ' + str(lA1W[i])[:8] +' area' + '\n')
                                fabbotW.write('A2 = ' + str(lA2W[i])[:8] +' area' + '\n')
                                fabbotW.write('Wmr = ' + str(lWmr[i])[:8] +' um' + '\n')
                                count = 0
                                fabbotW1 = open(lnombres[i] + '_perfil_ondulacion_abbott.txt','w')
                                fabbotW1.write('Fichero de procedencia:' + ' ' + lnombres[i]+'\n')
                                fabbotW1.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
                                if comboTipoFiltro.current() == 0:
                                        fabbotW1.write('Tipo de filtro:' + ' ' + 'Gaussiano'+';' + '  '+str(PasaAltos.get())
+'mm'+'\n')
                                fabbotW1.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                                for num in yabbotW[i]:
                                        if SelecSeparadorDecimal.get() == 2:
                                                num = str(num)
                                                xabbotW[i][count] = str(xabbotW[i][count])
                                                xabbotW[i][count].replace('.',',')
                                                num = num.replace('.',',')

                fabbotW1.write(xabbotW[i][count]+ElementoSeparadorSalida.get()+num+'\n')
                                        else:

                fabbotW1.write(str(xabbotW[i][count])+ElementoSeparadorSalida.get()+str(num)+'\n')
                                        count = count+1
                                fabbotW1.close()
                                fabbotW.close()
                                shutil.move(lnombres[i] +
'_parametros_ondulacion_abbott.txt',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Abbott' + '/' +lnombres[i]+'_ondulacion')
                                shutil.move(lnombres[i] +
'_perfil_ondulacion_abbott.txt',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Abbott' + '/' +lnombres[i]+'_ondulacion')

                if SelecAbbotRugosidad.get() == 1:
                        for i in range (len(yabbotR)):
                                fabbotR = open(lnombres[i] + '_parametros_rugosidad_abbott.txt','w')
                                fabbotR.write('Fichero de procedencia:' + ' ' + lnombres[i]+'\n')
                                fabbotR.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
                                if comboTipoFiltro.current() == 0:
                                        fabbotR.write('Tipo de filtro:' + ' ' + 'Gaussiano'+';' + '  '+str(PasaAltos.get()) +'mm'+'\n')
                                fabbotR.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                                fabbotR.write('Rpk = ' + str(lRpk[i])[:8] +' um' + '\n')
                                fabbotR.write('Rvk = ' + str(lRvk[i])[:8] +' um' + '\n')
```

```
                                fabbotR.write('Rk = ' + str(lRk[i])[:8] +' um' + '\n')
                                fabbotR.write('Mr1 = ' + str(lMr1R[i])[:8] +' %' + '\n')
                                fabbotR.write('Mr2 = ' + str(lMr2R[i])[:8] +' %' + '\n')
                                fabbotR.write('A1 = ' + str(lA1R[i])[:8] +' area' + '\n')
                                fabbotR.write('A2 = ' + str(lA2R[i])[:8] +' area'+ '\n')
                                fabbotR.write('Rmr = ' + str(lRmr[i])[:8] +' um' + '\n')
                                count = 0
                                fabbotR1 = open(lnombres[i] + '_perfil_rugosidad_abbott.txt','w')
                                fabbotR1.write('Fichero de procedencia:' + ' ' + lnombres[i]+'\n')
                                fabbotR1.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
                                if comboTipoFiltro.current() == 0:
                                        fabbotR1.write('Tipo de filtro:' + ' ' + 'Gaussiano'+';' + '  '+str(PasaAltos.get())
+'mm'+'\n')
                                fabbotR1.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                                for num in yabbotR[i]:
                                        if SelecSeparadorDecimal.get() == 2:
                                                num = str(num)
                                                xabbotR[i][count] = str(xabbotR[i][count])
                                                xabbotR[i][count].replace('.',',')
                                                num = num.replace('.',',')

        fabbotR1.write(xabbotR[i][count]+ElementoSeparadorSalida.get()+num+'\n')
                                        else:

        fabbotR1.write(str(xabbotR[i][count])+ElementoSeparadorSalida.get()+str(num)+'\n')
                                        count = count+1
                                fabbotR.close()
                                fabbotR1.close()
                                shutil.move(lnombres[i] +
'_parametros_rugosidad_abbott.txt',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Abbott' + '/' +lnombres[i]+'_rugosidad')
                                shutil.move(lnombres[i] +
'_perfil_rugosidad_abbott.txt',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Abbott' + '/' +lnombres[i]+'_rugosidad')

        if SelecAbbotPrimario.get() == 1:
                for i in range (len(xampP)):
                                count = 0
                                fampP = open(lnombres[i] + '_perfil_primario_abbott_distribucion_amplitud.txt','w')
                                fampP.write('Fichero de procedencia:' + ' ' + lnombres[i]+'\n')
                                fampP.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
                                if comboTipoFiltro.current() == 0:
                                        fampP.write('Tipo de filtro:' + ' ' + 'Gaussiano'+';' + '  '+str(PasaAltos.get()) +'mm'+'\n')
                                fampP.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                                for num in yampP[i]:
                                        if SelecSeparadorDecimal.get() == 2:
                                                num = str(num)
                                                xampP[i][count] = str(xampP[i][count])
                                                xampP[i][count].replace('.',',')
                                                num = num.replace('.',',')
                                                fampP.write(xampP[i][count]+ElementoSeparadorSalida.get()+num+'\n')
                                        else:

        fampP.write(str(xampP[i][count])+ElementoSeparadorSalida.get()+str(num)+'\n')
                                        count = count+1
```

```
                                fampP.close()
                                shutil.move(lnombres[i] +
'_perfil_primario_abbott_distribucion_amplitud.txt',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Abbott' +
'/'+lnombres[i]+'_primario')

        if SelecAbbotOndulacion.get() == 1:
                for i in range (len(xampW)):
                                count = 0
                                fampW = open(lnombres[i] + '_perfil_ondulacion_abbott_distribucion_amplitud.txt','w')
                                fampW.write('Fichero de procedencia:' + ' ' + lnombres[i]+'\n')
                                fampW.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
                                if comboTipoFiltro.current() == 0:
                                        fampW.write('Tipo de filtro:' + ' ' + 'Gaussiano'+';' + '  '+str(PasaAltos.get()) +'mm'+'\n')
                                fampW.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                                for num in yampW[i]:
                                        if SelecSeparadorDecimal.get() == 2:
                                                num = str(num)
                                                xampW[i][count] = str(xampW[i][count])
                                                xampW[i][count].replace('.',',')
                                                num = num.replace('.',',')
                                                fampW.write(xampW[i][count]+ElementoSeparadorSalida.get()+num+'\n')
                                        else:

        fampW.write(str(xampW[i][count])+ElementoSeparadorSalida.get()+str(num)+'\n')
                                        count = count+1
                                fampW.close()
                                shutil.move(lnombres[i] +
'_perfil_ondulacion_abbott_distribucion_amplitud.txt',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Abbott' + '/'
+lnombres[i]+'_ondulacion')

        if SelecAbbotRugosidad.get() == 1:
                for i in range (len(xampR)):
                                count = 0
                                fampR = open(lnombres[i] + '_perfil_rugosidad_abbott_distribucion_amplitud.txt','w')
                                fampR.write('Fichero de procedencia:' + ' ' + lnombres[i]+'\n')
                                fampR.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
                                if comboTipoFiltro.current() == 0:
                                        fampR.write('Tipo de filtro:' + ' ' + 'Gaussiano'+';' + '  '+str(PasaAltos.get()) +'mm'+'\n')
                                fampR.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                                for num in yampR[i]:
                                        if SelecSeparadorDecimal.get() == 2:
                                                num = str(num)
                                                xampR[i][count] = str(xampR[i][count])
                                                xampR[i][count].replace('.',',')
                                                num = num.replace('.',',')
                                                fampR.write(xampR[i][count]+ElementoSeparadorSalida.get()+num+'\n')
                                        else:

        fampR.write(str(xampR[i][count])+ElementoSeparadorSalida.get()+str(num)+'\n')
                                        count = count+1
                                fampR.close()
                                shutil.move(lnombres[i] +
'_perfil_rugosidad_abbott_distribucion_amplitud.txt',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Abbott' + '/'
```

```
+lnombres[i]+'_rugosidad')

        if SelecRugosidad.get() == 0 or 1:
                for i in range (len(yrugos)):
                        if i == 0:
                                fparam = open('parametros_resumen.txt','w')
                                fparam.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
                                if comboTipoFiltro.current() == 0:
                                        fparam.write('Tipo de filtro:' + ' ' + 'Gaussiano'+';' + '  '+str(PasaAltos.get())
+'mm'+'\n')
                                fparam.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                                fparam.write(': Ensayo  (ISO 4287)     Rp          Rv          Rz          Rt          Ra
    Rq          Rsk          Rku          Rsm'+'\n')
                                fparam.write(': '+lnombres[i]+' '*(19-len(lnombres[i]))+' '+str(listRp[i])[:8]+' um'+'
'+str(listRv[i])[:8]+' um'+'   '+str(listRz[i])[:8]+' um'+'   '+str(listRt[i])[:8]+' um'+'   '+str(listRa[i])[:8]+' um'+'   '+str(listRq[i])[:8]+' um'+'
  '+str(listRsk[i])[:8]+'      '+str(listRkv[i])[:8]+'      '+str(listRsm[i])[:8]+' um'+'\n')
                        else:
                                fparam = open('parametros_resumen.txt','a')
                                fparam.write(': '+lnombres[i]+' '*(19-len(lnombres[i]))+' '+str(listRp[i])[:8]+' um'+'
'+str(listRv[i])[:8]+' um'+'   '+str(listRz[i])[:8]+' um'+'   '+str(listRt[i])[:8]+' um'+'   '+str(listRa[i])[:8]+' um'+'   '+str(listRq[i])[:8]+' um'+'
  '+str(listRsk[i])[:8]+'      '+str(listRkv[i])[:8]+'      '+str(listRsm[i])[:8]+' um'+'\n')
                                fparam.close()
                if SelecAbbotRugosidad.get() == 1:
                        for i in range (len(yabbotR)):
                                if i == 0:
                                        fparam = open('parametros_resumen.txt','a')
                                        fparam.write('\n')
                                        fparam.write('\n')
                                        fparam.write(': Ensayo (ISO 13565-2)   Rk          Rpk          Rvk          Mr1
        Mr2          A1          A2'+'\n')
                                        fparam.write(': '+lnombres[i]+' '*(19-len(lnombres[i]))+' '+str(lRk[i])[:8]+'
um'+'   '+str(lRpk[i])[:8]+' um'+'   '+str(lRvk[i])[:8]+' um'+'   '+str(lMr1R[i])[:8]+' %'+'    '+str(lMr2R[i])[:8]+' %'+'    '+str(lA1R[i])[:8]+'
  '+str(lA2R[i])[:8]+'\n')
                                else:
                                        fparam = open('parametros_resumen.txt','a')
                                        fparam.write(': '+lnombres[i]+' '*(19-len(lnombres[i]))+' '+str(lRk[i])[:8]+'
um'+'   '+str(lRpk[i])[:8]+' um'+'   '+str(lRvk[i])[:8]+' um'+'   '+str(lMr1R[i])[:8]+' %'+'    '+str(lMr2R[i])[:8]+' %'+'    '+str(lA1R[i])[:8]+'
  '+str(lA2R[i])[:8]+'\n')
                        for i in range (len(yabbotR)):
                                if i == 0:
                                        fparam = open('parametros_resumen.txt','a')
                                        fparam.write('\n')
                                        fparam.write('\n')
                                        fparam.write(': Parámetro de relación de material     Rmr'+'\n')
                                        fparam.write(': '+lnombres[i]+' '*(34-len(lnombres[i]))+' '+str(lRmr[i])[:8]+' %'
+ '   1um bajo el pico maximo' +'\n')
                                else:
                                        fparam = open('parametros_resumen.txt','a')
                                        fparam.write(': '+lnombres[i]+' '*(34-len(lnombres[i]))+' '+str(lRmr[i])[:8]+' %'
+ '   1um bajo el pico maximo' +'\n')
                                fparam.close()
                        shutil.move('parametros_resumen.txt',lnombres1[i]+'/'+'Resultados')
                elapsed_time = time.time () - starting_point
                print('tiempo muestra resultados:',elapsed_time)
```

```
def muestra_resultados3D(xprim, yprim, zprim, listSpP, listSvP, listSzP, listSaP, listSqP, listSskP, listSkvP, xabbotP, yabbotP,
lSpkP, lSkP, lSvkP, lSmrP, lMr1P, lMr2P, lA1P, lA2P):
        starting_point = time.time()
        if SelecPrimario.get() == 0 or 1:
                for i in range (len(yprim)):
                        fprim = open(lnombres[i] + '_parametros_primario.txt','w')
                        fprim.write('Fichero de procedencia:' + ' ' + lnombres[i]+'\n')
                        fprim.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
                        fprim.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                        fprim.write('Sp = ' + str(listSpP[i])[:8] +' um' + '\n')
                        fprim.write('Sv = ' + str(listSvP[i])[:8] +' um' + '\n')
                        fprim.write('Sz = ' + str(listSzP[i])[:8] +' um' + '\n')
                        fprim.write('Sa = ' + str(listSaP[i])[:8] +' um' + '\n')
                        fprim.write('Sq = ' + str(listSqP[i])[:8] +' um' + '\n')
                        fprim.write('Ssk = ' + str(listSskP[i])[:8] +' um' + '\n')
                        fprim.write('Skv = ' + str(listSkvP[i])[:8] +' um' + '\n')
                        fprim.close()
                        fprim1 = open(lnombres[i] + '_perfil_primario.txt','w')
                        fprim1.write('Fichero de procedencia:' + ' ' + lnombres[i]+'\n')
                        fprim1.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
                        fprim1.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                        count = 0
                        for num in yprim[i]:
                                if comboUnidadesZSalida.current() == 0:
                                        num = num/1000
                                num = str(num)
                                if comboUnidadesXSalida.current() == 1:
                                        xprim[i][count] = (xprim[i][count])*1000
                                xprim[i][count] = str(xprim[i][count])
                                if comboUnidadesYSalida.current() == 1:
                                        yprim[i][count] = (yprim[i][count])*1000
                                yprim[i][count] = str(yprim[i][count])
                                if SelecSeparadorDecimal.get() == 2:
                                        xprim[i][count] = xprim[i][count].replace('.',',')
                                        yprim[i][count] = yprim[i][count].replace('.',',')
                                        num = num.replace('.',',')
                                if SelecEjesSalida.get() == 3:

        fprim1.write(xprim[i][count]+ElementoSeparadorSalida.get()+xprim[i][count]+ElementoSeparadorSalida.get()+num+'\n
')
                                        count = count+1
                        fprim1.close()
                        shutil.move(lnombres[i] +
'_parametros_primario.txt',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Perfiles' + '/' +lnombres[i]+'_primario')
                        shutil.move(lnombres[i] +
'_perfil_primario.txt',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Perfiles' + '/' +lnombres[i]+'_primario')
        if SelecAbbotPrimario.get() == 0 or 1:
                for i in range (len(yabbotP)):
                        fabbotP = open(lnombres[i] + '_parametros_primario_abbott.txt','w')
                        fabbotP.write('Fichero de procedencia:' + ' ' + lnombres[i]+'\n')
                        fabbotP.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
```

```
                                fabbotP.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                                fabbotP.write('Spk = ' + str(lSpkP[i])[:8] +' um' + '\n')
                                fabbotP.write('Svk = ' + str(lSvkP[i])[:8] +' um' + '\n')
                                fabbotP.write('Sk = ' + str(lSkP[i])[:8] +' um' + '\n')
                                fabbotP.write('Mr1 = ' + str(lMr1P[i])[:8] +' %' + '\n')
                                fabbotP.write('Mr2 = ' + str(lMr2P[i])[:8] +' %' + '\n')
                                fabbotP.write('A1 = ' + str(lA1P[i])[:8] +' area' + '\n')
                                fabbotP.write('A2 = ' + str(lA2P[i])[:8] +' area' + '\n')
                                fabbotP.write('Smr = ' + str(lSmrP[i])[:8] +' um' + '\n')
                                count = 0
                                fabbotP1 = open(lnombres[i] + '_perfil_primario_abbott.txt','w')
                                fabbotP1.write('Fichero de procedencia:' + ' ' + lnombres[i]+'\n')
                                fabbotP1.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
                                fabbotP1.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                                for num in yabbotP[i]:
                                        if SelecSeparadorDecimal.get() == 2:
                                                num = str(num)
                                                xabbotP[i][count] = str(xabbotP[i][count])
                                                xabbotP[i][count].replace('.',',')
                                                num = num.replace('.',',')

        fabbotP1.write(xabbotP[i][count]+ElementoSeparadorSalida.get()+num+'\n')
                                        else:

        fabbotP1.write(str(xabbotP[i][count])+ElementoSeparadorSalida.get()+str(num)+'\n')
                                        count = count+1
                                fabbotP.close()
                                fabbotP1.close()
                                shutil.move(lnombres[i] +
'_parametros_primario_abbott.txt',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Abbott' + '/' +lnombres[i]+'_primario')
                                shutil.move(lnombres[i] +
'_perfil_primario_abbott.txt',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Abbott' + '/' +lnombres[i]+'_primario')
        if SelecPrimario.get() == 0 or 1 and SelecAbbotPrimario.get() == 0 or 1:
                for i in range (len(zprim)):
                        if i == 0:
                                fparam = open('parametros_resumen_primario.txt','w')
                                fparam.write('Fecha y hora:' + ' ' + time.strftime("%d/%m/%y") + '--' +
time.strftime("%H:%M:%S")+'\n')
                                if comboTipoFiltro.current() == 0:
                                        fparam.write('Tipo de filtro:' + ' ' + 'Gaussiano'+';' + '  '+str(PasaAltos.get())
+'mm'+'\n')
                                fparam.write('Comentario:'+ ' ' + Comentario.get() + '\n')
                                fparam.write(': Ensayo          Sp          Sv          Sz          Sa          Sq
Ssk        Sku'+'\n')
                                fparam.write(': '+lnombres[i][0:17]+'    '*(19-len(lnombres[i]))+' '+str(listSpP[i])[:8]+'
um'+'   '+str(listSvP[i])[:8]+' um'+'   '+str(listSzP[i])[:8]+' um'+'   '+str(listSaP[i])[:8]+' um'+'   '+str(listSqP[i])[:8]+' um'+'
'+str(listSskP[i])[:8]+'    '+str(listSkvP[i])[:8]+'\n')
                        else:
                                fparam = open('parametros_resumen_primario.txt','a')
                                fparam.write(': '+lnombres[i][0:17]+'    '*(19-len(lnombres[i]))+' '+str(listSpP[i])[:8]+'
um'+'   '+str(listSvP[i])[:8]+' um'+'   '+str(listSzP[i])[:8]+' um'+'   '+str(listSaP[i])[:8]+' um'+'   '+str(listSqP[i])[:8]+' um'+'
'+str(listSskP[i])[:8]+'    '+str(listSkvP[i])[:8]+'\n')
                for i in range (len(yabbotP)):
                        if i == 0:
```

```
                                        fparam = open('parametros_resumen_primario.txt','a')
                                        fparam.write('\n')
                                        fparam.write('\n')
                                        fparam.write(': Ensayo              Sk          Spk          Svk          Mr1          Mr2
 A1          A2'+'\n')
                                        fparam.write(': '+lnombres[i][0:17]+'    '*(19-len(lnombres[i]))+' '+str(lSkP[i])[:8]+' um'+'
 '+str(lSpkP[i])[:8]+' um'+'   '+str(lSvkP[i])[:8]+' um'+'   '+str(lMr1P[i])[:8]+' %'+'    '+str(lMr2P[i])[:8]+' %'+'    '+str(lA1P[i])[:8]+'
'+str(lA2P[i])[:8]+'\n')
                                else:
                                        fparam = open('parametros_resumen_primario.txt','a')
                                        fparam.write(': '+lnombres[i][0:17]+'    '*(19-len(lnombres[i]))+' '+str(lSkP[i])[:8]+' um'+'
 '+str(lSpkP[i])[:8]+' um'+'   '+str(lSvkP[i])[:8]+' um'+'   '+str(lMr1P[i])[:8]+' %'+'    '+str(lMr2P[i])[:8]+' %'+'    '+str(lA1P[i])[:8]+'
'+str(lA2P[i])[:8]+'\n')
                        fparam.close()
                shutil.move('parametros_resumen_primario.txt',lnombres1[i]+'/'+'Resultados')
                elapsed_time = time.time () - starting_point
                print('tiempo muestra resultados:',elapsed_time)

def grafica_resultados_report(xprim, yprim, xond ,yond, xrugos, yrugos, xabbotR, yabbotR,
lnombres,listRp,listRv,listRz,listRt,listRa,listRq,listRsk,listRkv,listRsm,lRpk,lRk,lRvk,lRmr,lMr1R,lMr2R,lA1R,lA2R):
                starting_point = time.time()
                if SelecPrimario.get() == 1:
                        for i in range (len(yprim)):
                                plt.figure(figsize=(10, 3), dpi = 80)
                                plt.title(lnombres[i]+' Perfil primario')
                                plt.plot(yprim[i],linewidth=1)
                                plt.xticks([])
                                plt.xlabel('Longitud ('+str(xprim[i][0])+'mm - '+str(xprim[i][-1])+'mm)')
                                plt.ylabel('Altura (micras)')
                                plt.grid()
                                plt.savefig(lnombres[i]+'_perfil_primario.pdf')
                                plt.close()
                                shutil.move(lnombres[i] +
'_perfil_primario.pdf',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Perfiles' + '/' +lnombres[i]+'_primario')
                if SelecOndulacion.get() == 1:
                        for i in range (len(yond)):
                                plt.figure(figsize=(10, 3), dpi = 80)
                                plt.title(lnombres[i]+' Perfil de Ondulación')
                                plt.plot(yond[i],linewidth=1)
                                plt.xticks([])
                                plt.xlabel('Longitud ('+str(xond[i][0])+'mm - '+str(xond[i][-1])+'mm)')
                                plt.ylabel('Altura (micras)')
                                plt.grid()
                                plt.savefig(lnombres[i]+'_perfil_ondulacion.pdf')
                                plt.close()
                                shutil.move(lnombres[i] +
'_perfil_ondulacion.pdf',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Perfiles' + '/' +lnombres[i]+'_ondulacion')
                for i in range (len(yrugos)):
                        c = canvas.Canvas(lnombres[i]+'_report.pdf')
                        c.setPageSize(landscape(A4))
                        if SelecRugosidad.get() == 0 or 1:
                                if SelecAbbotRugosidad.get() == 1:
                                        plt.figure(figsize=(10, 3), dpi = 80)
                                        plt.title(lnombres[i]+' Perfil de Rugosidad')
                                        plt.plot(yrugos[i],linewidth=1)
```

```
                                    plt.xticks([])
                                    plt.xlabel('Longitud ('+str(xrugos[i][0])+'mm - '+str(xrugos[i][-1])+'mm)')
                                    plt.ylabel('Altura (micras)')
                                    plt.grid()
                                    plt.savefig(lnombres[i]+'_perfil_rugosidad.pdf')
                                    plt.savefig(lnombres[i]+'_perfil_rugosidad.tiff')
                                    logo = ImageReader(lnombres[i]+'_perfil_rugosidad.tiff')
                                    c.drawImage(logo, 30, 340, mask='auto')
                                    c.drawString(50, 300,'Parámetros de amplitud')
                                    a = str(listRp[i])[:8].replace('.',',')
                                    c.drawString(50, 270,'Rp = ' + a +' um')
                                    a = str(listRv[i])[:8].replace('.',',')
                                    c.drawString(50, 250,'Rv = ' + a +' um')
                                    a = str(listRz[i])[:8].replace('.',',')
                                    c.drawString(50, 230,'Rz = ' + a +' um')
                                    a = str(listRt[i])[:8].replace('.',',')
                                    c.drawString(50, 210,'Rt = ' + a +' um')
                                    a = str(listRa[i])[:8].replace('.',',')
                                    c.drawString(50, 190,'Ra = ' + a +' um')
                                    a = str(listRq[i])[:8].replace('.',',')
                                    c.drawString(50, 170,'Rq = ' + a +' um')
                                    a = str(listRsk[i])[:8].replace('.',',')
                                    c.drawString(50, 150,'Rsk = ' + a +' um')
                                    a = str(listRkv[i])[:8].replace('.',',')
                                    c.drawString(50, 130,'Rku = ' + a +' um')
                                    a = str(listRsm[i])[:8].replace('.',',')
                                    c.drawString(50, 90,'Parámetro de espaciamiento')
                                    c.drawString(50, 60,'Rsm = ' + a +' um')
                                    plt.close()
                                    shutil.move(lnombres[i] +
'_perfil_rugosidad.pdf',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Perfiles' + '/' +lnombres[i]+'_rugosidad')
                        if SelecRugosidad.get() == 0 or 1:
                                if SelecAbbotRugosidad.get() == 0:
                                    plt.figure(figsize=(10, 3), dpi = 80)
                                    plt.title(lnombres[i]+' Perfil de Rugosidad')
                                    plt.plot(yrugos[i],linewidth=1)
                                    plt.xticks([])
                                    plt.xlabel('Longitud ('+str(xrugos[i][0])+'mm - '+str(xrugos[i][-1])+'mm)')
                                    plt.ylabel('Altura (micras)')
                                    plt.grid()
                                    plt.savefig(lnombres[i]+'_perfil_rugosidad.pdf')
                                    plt.savefig(lnombres[i]+'_perfil_rugosidad.tiff')
                                    logo = ImageReader(lnombres[i]+'_perfil_rugosidad.tiff')
                                    c.drawImage(logo, 30, 340, mask='auto')
                                    c.drawString(250, 300,'Parámetros de amplitud')
                                    a = str(listRp[i])[:8].replace('.',',')
                                    c.drawString(250, 270,'Rp = ' + a +' um')
                                    a = str(listRv[i])[:8].replace('.',',')
                                    c.drawString(250, 250,'Rv = ' + a +' um')
                                    a = str(listRz[i])[:8].replace('.',',')
                                    c.drawString(250, 230,'Rz = ' + a +' um')
                                    a = str(listRt[i])[:8].replace('.',',')
                                    c.drawString(250, 210,'Rt = ' + a +' um')
                                    a = str(listRa[i])[:8].replace('.',',')
                                    c.drawString(250, 190,'Ra = ' + a +' um')
```

```
                                        a = str(listRq[i])[:8].replace('.',',')
                                        c.drawString(450, 270,'Rq = ' + a +' um')
                                        a = str(listRsk[i])[:8].replace('.',',')
                                        c.drawString(450, 250,'Rsk = ' + a +' um')
                                        a = str(listRkv[i])[:8].replace('.',',')
                                        c.drawString(450, 230,'Rkv = ' + a +' um')
                                        a = str(listRsm[i])[:8].replace('.',',')
                                        c.drawString(450, 210,'Rsm = ' + a +' um')
                                        plt.close()
                                        shutil.move(lnombres[i] +
'_perfil_rugosidad.pdf',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Perfiles' + '/' +lnombres[i]+'_rugosidad')
                        if SelecAbbotRugosidad.get() == 1:
                                plt.plot(xabbotR[i],yabbotR[i],linewidth=2)
                                plt.title(lnombres[i]+' Curva de Abbott-Firestone')
                                plt.xlabel('Porcentaje de material (%)')
                                plt.ylabel('Altura (micras)')
                                plt.grid()
                                plt.savefig(lnombres[i]+'_perfil_rugosidad_abbott.pdf')
                                plt.savefig(lnombres[i]+'_Curva_Abbott.tiff')
                                logo = ImageReader(lnombres[i]+'_Curva_Abbott.tiff')
                                c.drawImage(logo, 460, 15, mask='auto', width = 320, height = 320)
                                c.drawString(250, 300,'Parámetros de Abbott')
                                c.drawString(250, 280,'*Válidos si la curva tiene forma de S')
                                a = str(lRpk[i])[:8].replace('.',',')
                                c.drawString(250, 250,'Rpk = ' + a +' um')
                                a = str(lRvk[i])[:8].replace('.',',')
                                c.drawString(250, 230,'Rvk = ' + a +' um')
                                a = str(lRk[i])[:8].replace('.',',')
                                c.drawString(250, 210,'Rk = ' + a +' um')
                                a = str(lMr1R[i])[:8].replace('.',',')
                                c.drawString(250, 190,'Mr1 = ' + a +' %')
                                a = str(lMr2R[i])[:8].replace('.',',')
                                c.drawString(250, 170,'Mr2 = ' + a +' %')
                                a = str(lA1R[i])[:8].replace('.',',')
                                c.drawString(250, 150,'A1 = ' + a +' area')
                                a = str(lA2R[i])[:8].replace('.',',')
                                c.drawString(250, 130,'A2 = ' + a +' area')
                                a = str(lRmr[i])[:8].replace('.',',')
                                c.drawString(250, 110,'Rmr = ' + a +' %  1um bajo Rt')
                                plt.close()
                                shutil.move(lnombres[i] +
'_perfil_rugosidad_abbott.pdf',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Abbott' + '/' +lnombres[i]+'_rugosidad')
                        c.save()
                        shutil.move(lnombres[i] + '_report.pdf',lnombres1[i]+'/'+'Resultados/'+lnombres[i])
                elapsed_time = time.time () - starting_point
                print('tiempo grafica resultados y report:',elapsed_time)

def grafica_resultados_report3D(xprim, yprim, zprim, listSpP, listSvP, listSzP, listSaP, listSqP, listSskP, listSkvP, xabbotP,
yabbotP, lSpkP, lSkP, lSvkP, lSmrP, lMr1P, lMr2P, lA1P, lA2P):
                starting_point = time.time()
                if SelecPrimario.get() == 0 or 1  and SelecAbbotPrimario.get() == 0 or 1:
                        for i in range (len(yprim)):
                                plt.plot(xabbotP[i],yabbotP[i],linewidth=2)
                                plt.title(lnombres[i][0:25]+' Curva de Abbott-Firestone')
                                plt.xlabel('Porcentaje de material (%)')
```

```
                                plt.ylabel('Altura (micras)')
                                plt.grid()
                                plt.savefig(lnombres[i]+'_perfil_primario_abbott.pdf')
                                shutil.move(lnombres[i] +
'_perfil_primario_abbott.pdf',lnombres1[i]+'/'+'Resultados/'+lnombres[i]+'/'+'Abbott' + '/' +lnombres[i]+'_primario')
        elapsed_time = time.time () - starting_point
        print('tiempo grafica resultados y report:',elapsed_time)

def renombrar_y_mover_carpeta():
        starting_point = time.time()
        if len(LugarSalida.get()) > 1:
                shutil.move(lnombres1[0]+'Resultados',LugarSalida.get())
                os.rename(LugarSalida.get()+'Resultados',LugarSalida.get()+'Resultados'+' '+ time.strftime("%y-%m-%d")
+ '--' + time.strftime("%Hh %M'%S'"))
        else:
                os.rename(lnombres1[0]+'Resultados',lnombres1[0]+'Resultados'+' '+ time.strftime("%y-%m-%d") + '--' +
time.strftime("%Hh %M'%S'"))
        elapsed_time = time.time () - starting_point
        print('tiempo renombrar y mover carpeta:',elapsed_time)


raiz = Tk() #ventana principal de la aplicación
raiz.geometry('750x600') #dimensiones de la ventana (ancho x alto)
raiz.configure(bg = 'beige') #color de fondo de la ventana
raiz.title('Programa de rugosidad') #asigna título a la ventana

######################
#Fichero fuente
######################
lblFicheroOrigen = Label(text = 'Fichero/s origen', font = 'Helvetica 10 bold').place(x = 5 , y = 15)
#Tipo de estudio
lblTipoEstudio = Label(text = 'Tipo de estudio:').place(x = 5, y = 45)
comboTipoEstudio = ttk.Combobox(raiz, width = 4)
comboTipoEstudio.place(x=100 , y=45)
comboTipoEstudio['values'] = ('2D', '3D')
comboTipoEstudio.current(0)


#Seleccionar fichero
lblSeleccionarFichero = Label(text = 'Seleccionar fichero/s:').place(x=5,y=75)
botonBuscar = Button(raiz, text = 'Buscar...', command =abrir).place(x=130,y=75)

#Formato del fichero
lblFormatoFichero = Label(text = 'Formato del fichero:').place(x = 5, y = 110)
comboFormatoFichero = ttk.Combobox(raiz, width = 4)
comboFormatoFichero.place(x=120,y=110)
comboFormatoFichero['values'] = ['.txt','.prf']
comboFormatoFichero.current(0)

######################
#Formato de entrada
######################
lblFormatoEntrada = Label(text = 'Formato de entrada', font = 'Helvetica 10 bold').place(x = 5, y = 156)
#Ejes y unidades de los ejes
lblEjes = Label(text = 'Ejes:').place(x=5,y=180)
```

```
SelecEjes = IntVar()
XY = Radiobutton(raiz, text = 'x , z', value=1, variable = SelecEjes).place(x=40, y=180)
X = Radiobutton(raiz, text = 'z', value=2, variable = SelecEjes).place(x=100,y=180)
XYZ = Radiobutton(raiz, text = 'x , y , z  (solo 3D)', value=3, variable = SelecEjes).place(x=145, y=180)
SelecEjes.set(1)
lblUnidadesX = Label(text = 'Unidades x:').place(x=5 , y=215)
lblUnidadesY = Label(text = 'Unidades y:').place(x=140, y=215)
lblUnidadesZ = Label(text = 'Unidades z:').place(x=5, y=245)
comboUnidadesX = ttk.Combobox(raiz, width = 6)
comboUnidadesX.place(x=75,y=215)
comboUnidadesX['values'] = ['mm','micras']
comboUnidadesX.current(0)
comboUnidadesY = ttk.Combobox(raiz, width = 6)
comboUnidadesY.place(x=210,y=215)
comboUnidadesY['values'] = ['mm','micras']
comboUnidadesY.current(0)
comboUnidadesZ = ttk.Combobox(raiz, width = 6)
comboUnidadesZ.place(x=75,y=245)
comboUnidadesZ['values'] = ['mm','micras']
comboUnidadesZ.current(1)

#Resolución X
lblResoluciónX = Label(text = 'Resolución x (si solamente z) [mm]:').place(x=5,y=275)
ResolucionX = StringVar(value = '0,00025')
entryResolucionX = Entry(raiz, textvariable = ResolucionX, justify = CENTER, width = 10).place(x=200,y=275)

#Elemento separador
lblElementoSeparador = Label(text = 'Elemento separador:').place(x=5,y=305)
ElementoSeparador = StringVar(value = ';')
entryElementoSeparador = Entry(raiz, width = 4, textvariable = ElementoSeparador, justify = CENTER).place(x = 125, y =305)

#Numero de lineas del encabezado
lblLineasEncabezado = Label(text = 'Nº lineas del encabezado:').place(x=5,y=335)
LineasEncabezado = IntVar(value = 0)
entryLineasEncabezado = Entry(raiz, width = 4, textvariable = LineasEncabezado, justify = CENTER).place(x = 150, y =335)


#########################
#Parámetros de filtrado
#########################
lblFiltro = Label(text = 'Filtro y parámetros de filtrado', font = 'Helvetica 10 bold').place(x = 5 , y = 380)
lblTipoFiltro = Label(text = 'Tipo de filtro:').place(x=5,y=410)
comboTipoFiltro = ttk.Combobox(raiz)
comboTipoFiltro.place(x=90,y=410)
comboTipoFiltro['values'] = ['Gaussiano','otros']
comboTipoFiltro.current(0)
lblPasaAltos = Label(text = 'Pasa Altos (λc):').place(x = 5, y = 440)
PasaAltos = StringVar(value = '0,8')
entryPasaAltos = Entry(raiz, textvariable = PasaAltos, justify = CENTER).place(x = 95, y = 440)
lblPasaBajos = Label(text = 'Pasa bajos (λs):').place(x = 5, y = 470)
PasaBajos = StringVar(value = '0,0025')
entryPasaBajos = Entry(raiz, textvariable = PasaBajos, justify = CENTER).place(x = 95, y = 470)


#########################
```

```
#Tipo de perfil
#########################
lblTipos = Label(text = 'Elementos de salida', font = 'Helvetica 10 bold').place(x = 320 , y = 15)
SelecPrimario = IntVar(value = 1)
checkPrimario = Checkbutton(raiz, text = 'Primario', variable = SelecPrimario, onvalue = 1, offvalue = 0).place(x = 320, y = 45)
SelecRugosidad = IntVar(value = 1)
checkRugosidad = Checkbutton(raiz, text = 'Rugosidad', variable = SelecRugosidad, onvalue = 1, offvalue = 0).place(x = 430, y
= 45)
SelecOndulacion = IntVar(value = 1)
checkOndulacion = Checkbutton(raiz, text = 'Ondulación', variable = SelecOndulacion, onvalue = 1, offvalue = 0).place(x = 560,
y = 45)
SelecAbbotPrimario = IntVar(value = 1)
checkAbbotPrimario = Checkbutton(raiz, text = 'Abbott Primario', variable = SelecAbbotPrimario, onvalue = 1, offvalue =
0).place(x = 320, y = 80)
SelecAbbotRugosidad = IntVar(value = 1)
checkAbbotRugosidad = Checkbutton(raiz, text = 'Abbott Rugosidad', variable = SelecAbbotRugosidad, onvalue = 1, offvalue =
0).place(x = 430, y = 80)
SelecAbbotOndulacion = IntVar(value = 1)
checkAbbotOndulacion = Checkbutton(raiz, text = 'Abbott Ondulacion', variable = SelecAbbotOndulacion, onvalue = 1, offvalue
= 0).place(x = 560, y = 80)
SelecProbPrimario = IntVar(value = 0)
checkProbPrimario = Checkbutton(raiz, text = 'Prob. Primario', variable = SelecProbPrimario, onvalue = 1, offvalue = 0).place(x
= 320, y = 110)
SelecProbRugosidad = IntVar(value = 0)
checkProbRugosidad = Checkbutton(raiz, text = 'Prob. Rugosidad', variable = SelecProbRugosidad, onvalue = 1, offvalue =
0).place(x = 430, y = 110)
SelecProbOndulacion = IntVar(value = 0)
checkProbOndulacion = Checkbutton(raiz, text = 'Prob. Ondulacion', variable = SelecProbOndulacion, onvalue = 1, offvalue =
0).place(x = 560, y = 110)

############################
#Formato de salida
############################
#Ejes y unidades de los ejes
lblFormatoSalida = Label(text = 'Formato de salida', font = 'Helvetica 10 bold').place(x = 320 , y = 155)
lblEjesSalida = Label(text = 'Ejes:').place(x=320,y=180)
SelecEjesSalida = IntVar()
XYSalida = Radiobutton(raiz, text = 'x , z', value=1, variable = SelecEjesSalida).place(x=360, y=180)
XSalida = Radiobutton(raiz, text = 'z', value=2, variable = SelecEjesSalida).place(x=420,y=180)
XYZSalida = Radiobutton(raiz, text = 'x , y , z  (solo 3D)', value=3, variable = SelecEjesSalida).place(x=480, y=180)
SelecEjesSalida.set(1)
lblUnidadesXSalida = Label(text = 'Unidades x:').place(x=320 , y=215)
lblUnidadesYSalida = Label(text = 'Unidades y:').place(x=460 , y=215)
lblUnidadesZSalida = Label(text = 'Unidades z:').place(x=320, y=245)
comboUnidadesXSalida = ttk.Combobox(raiz, width = 6)
comboUnidadesXSalida.place(x=390,y=215)
comboUnidadesXSalida['values'] = ['mm','micras']
comboUnidadesXSalida.current(0)
comboUnidadesYSalida = ttk.Combobox(raiz, width = 6)
comboUnidadesYSalida.place(x=530,y=215)
comboUnidadesYSalida['values'] = ['mm','micras']
comboUnidadesYSalida.current(0)
comboUnidadesZSalida = ttk.Combobox(raiz, width = 6)
comboUnidadesZSalida.place(x=390,y=245)
comboUnidadesZSalida['values'] = ['mm','micras']
```

```
comboUnidadesZSalida.current(1)

#Separador Decimal
lblSeparadorDecimal = Label(text = 'Separador decimal:').place(x=320,y=275)
SelecSeparadorDecimal = IntVar()
punto = Radiobutton(raiz, text = 'punto', value=1, variable = SelecSeparadorDecimal).place(x=430, y=275)
coma = Radiobutton(raiz, text = 'coma', value=2, variable = SelecSeparadorDecimal).place(x=500,y=275)
SelecSeparadorDecimal.set(1)

#Elemento Separador
lblElementoSeparadorSalida = Label(text = 'Elemento separador:').place(x=320,y=305)
ElementoSeparadorSalida = StringVar(value = ';')
entryElementoSeparadorSalida = Entry(raiz, width = 4, textvariable = ElementoSeparadorSalida, justify = CENTER).place(x =
440, y =305)

#Comentario
lblComentario = Label(text = 'Comentario en fichero de salida:').place(x=320,y=335)
Comentario = StringVar()
entryComentario = Entry(raiz, textvariable = Comentario, justify = CENTER, width = 30).place(x=500,y=335)

#Lugar salida
lblLugarSalida = Label(text = 'Ruta de salida (por defecto el directorio de donde procede el fichero):').place(x=320,y=365)
LugarSalida = StringVar(value ='')
entryLugarSalida = Entry(raiz, width = 45, textvariable = LugarSalida, justify = CENTER).place(x=320,y=395)


botonEjecutar = Button(raiz, text = 'Ejecutar programa', command = funcion_base, font = 'Helvetica 10
bold').place(x=430,y=460)

raiz.mainloop()
```

1.2. Funciones.py

```
import time
import math
from numba import jit
import numpy as np
@jit
def filtro_gaussiano(ValAFiltrar, NumValSal, NumValorsGauss, y, listv_Gauss_norm):
        lista = []
        for ii in range (NumValSal+1):
                AlturaFiltrada = 0
                for j in range(NumValorsGauss):
                        AlturaFiltrada = AlturaFiltrada + (y[ValAFiltrar + ii - j] * listv_Gauss_norm[j])
                for j in range(1,NumValorsGauss):
                        AlturaFiltrada = AlturaFiltrada + (y[ValAFiltrar + ii + j] * listv_Gauss_norm[j])
                lista.append(AlturaFiltrada)
        return (lista)

def bucle_ondulacion(CoefPasoBusq, NumPuntosX, ErrBusq, yinter, yinicial,SumaPicos, SumaValles, PasoBusq):
```

```
        for i in range(NumPuntosX):
                if abs((SumaPicos - SumaValles)/NumPuntosX) > ErrBusq:
                        ListaPicos = []
                        ListaValles = []
                        if len (yinicial) == 0:
                                if SumaPicos > SumaValles:
                                        for indice,elem in enumerate(yinter):
                                                elem = elem - PasoBusq
                                                yinicial.append(elem)
                                else:
                                        for indice,elem in enumerate(yinter):
                                                elem = elem + PasoBusq
                                                yinicial.append(elem)
                        else:
                                if SumaPicos > SumaValles:
                                        for indice,elem in enumerate(yinicial):
                                                elem = elem - PasoBusq
                                                yinicial[indice] = elem
                                else:
                                        for indice,elem in enumerate(yinicial):
                                                elem = elem + PasoBusq
                                                yinicial[indice] = elem
                        for elemento in yinicial:
                                if elemento > 0:
                                        ListaPicos.append(elemento)
                                else:
                                        elemento = abs(elemento)
                                        ListaValles.append(elemento)
                        ListaPicos1 = np.array(ListaPicos)
                        ListaValles1 = np.array(ListaValles)
                        SumaPicos = sum(ListaPicos1)
                        SumaValles = sum(ListaValles1)
                        if SumaPicos == 0:
                                Sp = min(ListaValles)
                                Sv = max(ListaValles)
                        elif SumaValles == 0:
                                Sp = max(ListaPicos)
                                Sv = min(ListaPicos)
                        else:
                                Sp = max (ListaPicos)
                                Sv = max (ListaValles)
                        Sz = Sp + Sv
                        PasoBusq = Sz/CoefPasoBusq
                        ErrBusq = (PasoBusq/2) + (1/CoefPasoBusq)
                else:
                        break
        return (yinicial, SumaPicos, SumaValles)

def rsm(x, y, alt):
        longitud = 0.01 * (x[-1]-x[0])
        xi = []
        xii = []
        altura = False
        for i in range(1, len(y)-1):
                if y[i] >= 0 and y[i] >= y[i+1] and y[i] >= y[i-1]:
```

```
                    if y[i] >= alt:
                        altura = True
            if y[i] >= 0 and y[i+1] < 0 and altura == True:
                    if len(xi) == 0:
                        xi.append(x[i])
                    num = abs(xi[-1] - x[i])
                    if num >= longitud:
                        xi.append(x[i])
                    altura = False
for i in range(len(xi)-1):
        xii.append(abs(xi[i+1]-xi[i]))
return (xii)
```