Calibration et validation d'un modèle hydraulique en milieu de mangrove en Guinée-Con

Auteur : Moyaux, Vyckie
Promoteur(s) : Degré, Aurore; Archambeau, Pierre
Faculté : Gembloux Agro-Bio Tech (GxABT)
Diplôme : Master en bioingénieur : sciences et technologies de l'environnement, à finalité spécialisée
Année académique : 2019-2020
URI/URL : http://hdl.handle.net/2268.2/10801

Annexe

Annexe 1 – Code Python permettant de changer les variables des fichiers de géométrie

```
# -*- coding: utf-8 -*-
"""
Code créé pour l'automatisation du logiciel HEC-RAS dans le but de mettre
en place une analyse
de sensibilité, une calibration et une validation.

Le code nécessite l'installation préalable des librairies :
- rascontrol (https://github.com/mikebannis/rascontrol.git) permettant de
controller HEC-RAS
- parserasgeo (https://github.com/mikebannis/parserasgeo.git) permet de
modifier les fichiers
  géométrie.
"""


"""
Le premier code complète la librairie parserasgeo et permet de modifier les
coefficients de
déversement latéral, le coefficient de manning des ouvrages, les
coefficients de contraction
et d'expansion.
"""

import parserasgeo_ma.parserasgeo as prg

def change_mann_culv (n, outputfile, fichiergeom):
    """
    Fonction qui change le coefficient de manning des ouvrages
    n = nouveau coefficient de manning ; outputfile = fichier de géométrie
modifié ;
    fichiergeom = fichier de géométrie initial
    """
    model = []

    # Lecture du fichier de géométrie
    file = fichiergeom
    with open (file) as myfile:
        for myline in myfile:
            model.append(myline.rstrip('\n'))

    # Modification du fichier et réécriture vers le nouveau fichier de
géométrie
    with open (outputfile, 'w') as outfile:
        for myline in model:
            if myline.split("=")[0] == "IW Culv":
                carac = myline.split(",")
                #print(float(carac[1])*100)
                outfile.write(','.join([carac[0], carac[1], carac [2],
carac [3]]))
                outfile.write(','+str(float(n))+',')
                outfile.write(','.join([carac[5], carac[6], carac[7],
carac[8], carac[9], carac[10], carac[11], carac[12], carac[13],
carac[14]])+'\n')
            elif myline.split("=")[0] == 'IW Culv Bottom n':
                carac = myline.split('=')
                outfile.write(carac[0]+'=')
```

```
                outfile.write(str(float(n))+'\n')
            else:
                outfile.write(myline+'\n')

def change_entrance_culv (n, outputfile, fichiergeom):
    """
    Fonction qui change le coefficient d'entrée des ouvrages
    n = nouveau coefficient d'entrée des ouvrages
    """
    model = []

    file = fichiergeom
    with open (file) as myfile:
        for myline in myfile:
            model.append(myline.rstrip('\n'))

    with open (outputfile, 'w') as outfile:
        for myline in model:
            if myline.split("=")[0] == "IW Culv":
                carac = myline.split(",")
                #print(float(carac[1])*100)
                outfile.write(','.join([carac[0], carac[1], carac [2],
carac [3], carac [4]]))
                outfile.write(','+str(float(n))+',')
                outfile.write(','.join([carac[6], carac[7], carac[8],
carac[9], carac[10], carac[11], carac[12], carac[13], carac[14]])+'\n')
            else:
                outfile.write(myline+'\n')

def change_exit_culv (n, outputfile, fichiergeom):
    """
    Fonction qui change le coefficient de sortie des ouvrages
    n = nouveau coefficient de sortie
    """
    model = []

    file = fichiergeom
    with open (file) as myfile:
        for myline in myfile:
            model.append(myline.rstrip('\n'))

    with open (outputfile, 'w') as outfile:
        for myline in model:
            if myline.split("=")[0] == "IW Culv":
                carac = myline.split(",")
                #print(float(carac[1])*100)
                outfile.write(','.join([carac[0], carac[1], carac [2],
carac [3], carac [4], carac [5]]))
                outfile.write(','+str(float(n))+',')
                outfile.write(','.join([carac[7], carac[8], carac[9],
carac[10], carac[11], carac[12], carac[13], carac[14]])+'\n')
            else:
                outfile.write(myline+'\n')

def change_lat_coef (n, outputfile, fichiergeom):
    """
    Fonction qui change le coefficient de déverserment latéral des
structures latérales
    n = nouveau coefficient de déversement latéral

    """
```

```
    model = []

    file = fichiergeom
    with open (file) as myfile:
        for myline in myfile:
            model.append(myline.rstrip('\n'))

    with open (outputfile, 'w') as outfile:
        for myline in model:
            if myline.split("=")[0] == "Lateral Weir Coef":
                carac = myline.split('=')
                outfile.write(carac[0]+'=')
                outfile.write(str(float(n))+'\n')
            else:
                outfile.write(myline+'\n')

def change_mann_xs (n, outputfile, fichiergeom):
    """ change valeur de manning dans les cross sections.
    n = nouveau coefficient de manning
    outputfile = fichier de sortie
    fichiergeom = fichier d'entrée (.g**)
    """
    model = fichiergeom
    geo = prg.ParseRASGeo(model)

    # Donne toute les infos du fichier g.** sous le même format
    #xs = geo.return_xs_by_id(1545)
    #print(xs)

    for xs in geo.get_cross_sections() :
        #print(xs) # imprime tout le fichier texte de geométrie concernant
les cross sections
        n_vals = xs.mannings_n.values
        #print (n_vals)
      # n-values are stored as a list of 3-tuples
        new_n = [(station, nv, other) for station, nv, other in n_vals]
        #print(new_n[0])
        if new_n [0] != (0, 0.013, 0) :
            new = [(station, n, other) for station, nv, other in n_vals]
        else:
            new = [(station, nv, other) for station, nv, other in n_vals]
        xs.mannings_n.values = new

    geo.write(outputfile)

"""
Code utilisé dans le cadre du TFE
"""

# Importation des librairies

import rascontrol
import numpy as np
import pandas as pd
import change_var as change
import matplotlib.pyplot as plt
import math


""" Analyse de sensibilité sur le modèle III
Le coefficient de Manning des drains (n) et le coefficient de déversement
```

```
des structures latérales (Cd)
varient de +- 5%
n initial = 0,03
Cd initial = 1,1
"""

# Mise en place de la librairie et choix du dossier
project_RAS =
"C:/Users/MediaMonster/Documents/TFE/Code/hecras/XS_REDUCE.prj"
fichier_geom =
"C:/Users/MediaMonster/Documents/TFE/Code/hecras/XS_REDUCE.g14"
rc = rascontrol.RasController(version='507')
geo = prg.ParseRASGeo(fichier_geom)

# Variation manning drain
for i in np.arange (0.95, 1.05, 0.05):
    n = 0.03*i
    change.change_mann_xs (n, fichier_geom, fichier_geom)
    rc.open_project(project_RAS)

    # Modélisation du modèle après avoir changer le fichier de géométrie
    rc.run_current_plan()
    profs = rc.get_profiles()

    # Récupération des données dans un fichier texte
    with open('outn{0}.txt'.format(i), 'wt') as outfile:
            rivers = rc.get_rivers()
            for riv in rivers:
                for reach in riv.reaches:
                    for node in reach.nodes:
                        if node.node_type == '':
                            outfile.write(','.join([str(riv), str(reach),
node.node_id]))
                            for prof in profs:
                                outfile.write(','+str(node.value(prof, 2)))
                            outfile.write('\n')

# Lecture des fichiers de résultats
output_05 = pd.read_csv('outn0.95.txt', sep =",", header = None)
output_05 = output_05.drop([0,1,3], axis =1)
output_05 = output_05.set_index(2)

output_1 = pd.read_csv('outn1.0.txt', sep =",", header = None)
output_1 = output_1.drop([0,1,3], axis =1)
output_1 = output_1.set_index(2)

output_15 = pd.read_csv('outn1.05.txt', sep =",", header = None)
output_15 = output_15.drop([0,1,3], axis =1)
output_15 = output_15.set_index(2)

# Calcul de l'indice de sensibilité
Delta_S = (abs(output_15 - output_05))/output_1
sensibilite_ndrain = (Delta_S.mean(axis =1)).mean(axis =0)
print ("la sensibilité de manning des drains est de ", sensibilite_ndrain)

# Variation du coefficient de déversement latéral
for i in np.arange (0.95, 1.05, 0.05):
    n = i*1.1
    change.change_lat_coef(n, fichier_geom, fichier_geom)
    rc.open_project(project_RAS)
    rc.run_current_plan()
```

```
    profs = rc.get_profiles()

    with open('outCd{0}.txt'.format(i), 'wt') as outfile:
            rivers = rc.get_rivers()
            for riv in rivers:
                for reach in riv.reaches:
                    for node in reach.nodes:
                        if node.node_type == '':
                            outfile.write(','.join([str(reach),
node.node_id]))
                            for prof in profs:
                                outfile.write(','+str(node.value(prof, 2)))
                            outfile.write('\n')

output_05 = pd.read_csv('outCd0.95.txt', sep =",", header = None)
output_05 = output_05.drop([0,1,3], axis =1)
output_05 = output_05.set_index(2)

output_1 = pd.read_csv('outCd1.0.txt', sep =",", header = None)
output_1 = output_1.drop([0,1,3], axis =1)
output_1 = output_1.set_index(2)

output_15 = pd.read_csv('outCd1.05.txt', sep =",", header = None)
output_15 = output_15.drop([0,1,3], axis =1)
output_15 = output_15.set_index(2)

Delta_S = (abs(output_15 - output_05))/output_1
sensibilite_Cd= (Delta_S.mean(axis =1)).mean(axis =0)
print ("la senisibilité des coefficient de déversement latérale est de ",
sensibilite_Cd)

""" Sensibilité sur le modèle réduit de l'ouvrage
    Sensibilité sur le manning de l'ouvrage, le coefficient d'entrée et le
    coefficient de sortie

    Manning ouvrage varie autour de 0.013 (béton)
    Coefficient d'entrée autour de 0.5 (correspond à Box shape sans
arrondissement)
    Coefficient de sortie autour de 0.9 (1 = abrupte et 0.3 = doux)
"""

# Mise en place de la librairie et choix du dossier
project_RAS =
"C:/Users/MediaMonster/Documents/TFE/Code/hecras/KONDEYIRE_OUVRAGEPL.prj"
fichier_geom =
"C:/Users/MediaMonster/Documents/TFE/Code/hecras/KONDEYIRE_OUVRAGEPL.g07"
rc = rascontrol.RasController(version='507')
geo = prg.ParseRASGeo(fichier_geom)

# Variation manning ouvrage

for i in np.arange (0.95, 1.05, 0.05):
    n=0.013*i
    change.change_mann_culv(n, fichier_geom, fichier_geom)
    rc.open_project(project_RAS)

    rc.run_current_plan()
    profs = rc.get_profiles()

    with open('outnc{0}.txt'.format(i), 'wt') as outfile:
            rivers = rc.get_rivers()
```

```
            for riv in rivers:
                for reach in riv.reaches:
                    for node in reach.nodes:
                        if node.node_type == '':
                            outfile.write(','.join([str(reach),
node.node_id]))
                            for prof in profs:
                                outfile.write(','+str(node.value(prof, 2)))
                            outfile.write('\n')

output_05 = pd.read_csv('outnc0.95.txt', sep =",", header = None)
output_05 = output_05.drop([0,1,3], axis =1)
output_05 = output_05.set_index(2)

output_1 = pd.read_csv('outnc1.0.txt', sep =",", header = None)
output_1 = output_1.drop([0,1,3], axis =1)
output_1 = output_1.set_index(2)

output_15 = pd.read_csv('outnc1.05.txt', sep =",", header = None)
output_15 = output_15.drop([0,1,3], axis =1)
output_15 = output_15.set_index(2)

Delta_S = (abs(output_15 - output_05))/output_1
sensibilite_nculv= (Delta_S.mean(axis =1)).mean(axis =0)
print ("la sensibilité du manning de l'ouvrage est de ", sensibilite_nculv)

# Variation coefficient d'entrée ouvrage
for i in np.arange (0.95, 1.05, 0.05):
    kin = 0.5*i
    change.change_entrance_culv(kin, fichier_geom, fichier_geom)
    rc.open_project(project_RAS)

    rc.run_current_plan()
    profs = rc.get_profiles()

    with open('outkin{0}.txt'.format(i), 'wt') as outfile:
            rivers = rc.get_rivers()
            for riv in rivers:
                for reach in riv.reaches:
                    for node in reach.nodes:
                        if node.node_type == '':
                            outfile.write(','.join([str(reach),
node.node_id]))
                            for prof in profs:
                                outfile.write(','+str(node.value(prof, 2)))
                            outfile.write('\n')

output_05 = pd.read_csv('outkin0.95.txt', sep =",", header = None)
output_05 = output_05.drop([0,1,3], axis =1)
output_05 = output_05.set_index(2)

output_1 = pd.read_csv('outkin1.0.txt', sep =",", header = None)
output_1 = output_1.drop([0,1,3], axis =1)
output_1 = output_1.set_index(2)

output_15 = pd.read_csv('outkin1.05.txt', sep =",", header = None)
output_15 = output_15.drop([0,1,3], axis =1)
output_15 = output_15.set_index(2)

Delta_S = (abs(output_15 - output_05))/output_1
sensibilite_kin= (Delta_S.mean(axis =1)).mean(axis =0)
```

```
print ("la sensibilité du coefficient d'entrée est de ", sensibilite_kin)
# Variation coefficient de sortie ouvrage
for i in np.arange (0.95, 1.05, 0.05):
    kex = 1*i
    change.change_exit_culv(kex, fichier_geom, fichier_geom)
    rc.open_project(project_RAS)

    rc.run_current_plan()
    profs = rc.get_profiles()

    with open('outkex{0}.txt'.format(i), 'wt') as outfile:
            rivers = rc.get_rivers()
            for riv in rivers:
                for reach in riv.reaches:
                    for node in reach.nodes:
                        if node.node_type == '':
                            outfile.write(','.join([str(reach),
node.node_id]))
                            for prof in profs:
                                outfile.write(','+str(node.value(prof, 2)))
                            outfile.write('\n')

output_05 = pd.read_csv('outkex0.95.txt', sep =",", header = None)
output_05 = output_05.drop([0,1,3], axis =1)
output_05 = output_05.set_index(2)

output_1 = pd.read_csv('outkex1.0.txt', sep =",", header = None)
output_1 = output_1.drop([0,1,3], axis =1)
output_1 = output_1.set_index(2)

output_15 = pd.read_csv('outkex1.05.txt', sep =",", header = None)
output_15 = output_15.drop([0,1,3], axis =1)
output_15 = output_15.set_index(2)

Delta_S = (abs(output_15 - output_05))/output_1
sensibilite_kex= (Delta_S.mean(axis =1)).mean(axis =0)
print ("la sensibilité du coefficient de sortie est de ", sensibilite_kex)


"""
Code pour la calibration de Manning.
Les autres paramètres sont testés de la même manière en adaptant les
valeurs de RMSE
"""
# Recherche des XS dans fichier de sortie
# seulement pour calibration sur ouvrage complet
ouv2 = 7
k1 = 62
k2 = 164
k3 = 237
ouv3 = 255
ouvP = 284
seg = 329

# Ouverture du fichier de données des sondes
sonde_tot = pd.read_excel("sonde_5min.xlsx")
sonde_tot.head(n=1) # Donne noms au colonne
S_ouvplaine = sonde_tot['Ouvrage plaine']
S_ouv2 = sonde_tot['Ouvrage 2 passes']
S_ouv3 = sonde_tot['Ouvrage 3 passes']
S_seg = sonde_tot['Seguema']
```

```
S_k1 = sonde_tot['Kankoussaya 1']
S_k2 = sonde_tot['Kankoussaya 2']
S_k3 = sonde_tot['Kankoussaya 3']
Date_5 = sonde_tot['Date']
Date_k3 = Date_5[1600:]
S_k3 = S_k3[1600:]
S_mer = sonde_tot['Ouvrage mer']

Sonde = np.array([S_ouvplaine, S_ouv2, S_ouv3, S_seg, S_k1, S_k2])
Sonde_k3 = np.array([S_k3, Date_k3])

# Calibration
# Calcul d'un RMSEinf et un RMSEsup pour savoir dans quel sens on varie
# Test sur Manning sur ouvrage plaine

n = 0.03
nsup = n*1.0
ninf = n*0.99
RMSEinf = 0.01
RMSEsup = 1
RMSE=1

while RMSE > 0.02:
    if RMSE > 0.05:
        if RMSEsup<RMSEinf:
            n = nsup
        else:
            n = ninf

        nsup = n*1.1
        ninf = n*0.9
        print ('n supérieur =', round(nsup, 5))
        print ('n inférieur =', round(ninf, 5))

        rc.open_project(project_RAS)
        profs = rc.get_profiles()
        print ('Run 10% de nsup = ', round(nsup, 5))
            # Code avec n Supérieur
            change.change_mann_xs(round(nsup, 6), fichier_geom,
fichier_geom)
            rc.run_current_plan()

            with open('outnsup{0}.txt'.format(round(nsup, 5)), 'wt') as
outfile:
                rivers = rc.get_rivers()
                for riv in rivers:
                    for reach in riv.reaches:
                        for node in reach.nodes:
                            if node.node_type == '':
                                outfile.write(node.node_id)
                                for prof in profs:
                                    outfile.write(','+str(node.value(prof,
2)))
                                outfile.write('\n')

        # Code avec n Inférieur
        rc.open_project(project_RAS)
        profs = rc.get_profiles()
        print ('Run 10% de ninf = ', round(ninf, 5))
```

```
            change.change_mann_xs(round(ninf, 6), fichier_geom,
fichier_geom)
            rc.run_current_plan()

            with open('outninf{0}.txt'.format(round(ninf, 5)), 'wt') as
outfile:
                rivers = rc.get_rivers()
                for riv in rivers:
                    for reach in riv.reaches:
                        for node in reach.nodes:
                            if node.node_type == '':
                                outfile.write(node.node_id)
                                for prof in profs:
                                    outfile.write(','+str(node.value(prof,
2)))
                                outfile.write('\n')

        # Extraction des résultats
        # n Supérieur
        outsup = pd.read_csv('outnsup{0}.txt'.format(round(nsup, 5)), sep
=",", header = None)
        outsup = outsup.drop([1], axis =1)
        outsup = outsup.set_index(0)
        outsup = outsup.T
        outsup.reset_index(level = 0, inplace = True)
        outsup = outsup.drop ('index', axis = 1)
        ouvP_nsup = outsup.iloc[:, ouvP]
        ouv2_nsup = outsup.iloc[:, ouv2]
        ouv3_nsup = outsup.iloc[:, ouv3]
        seg_nsup = outsup.iloc[:, seg]
        k1_nsup = outsup.iloc[:, k1]
        k2_nsup = outsup.iloc[:, k2]
        k3_nsup = outsup.iloc[:, k3]

        nsuperieur = np.array([ouvP_nsup, ouv2_nsup, ouv3_nsup, seg_nsup,
k1_nsup, k2_nsup])
        ndiff_k3 =k3_nsup - S_k3
        ndiffsup = nsuperieur - Sonde
        MSEsup = np.mean(ndiffsup**2)
        MSE_k3 = np.mean(ndiff_k3)
        RMSEsup = math.sqrt(MSEsup)


        #n inférieur
        outinf = pd.read_csv('outninf{0}.txt'.format(round(ninf, 5)), sep
=",", header = None)
        outinf = outinf.drop([1], axis =1)
        outinf = outinf.set_index(0)
        outinf = outinf.T
        outinf.reset_index(level = 0, inplace = True)
        outinf = outinf.drop ('index', axis = 1)
        ouvP_ninf = outinf.iloc[:, ouvP]
        ouv2_ninf = outinf.iloc[:, ouv2]
        ouv3_ninf = outinf.iloc[:, ouv3]
        seg_ninf = outinf.iloc[:, seg]
        k1_ninf = outinf.iloc[:, k1]
        k2_ninf = outinf.iloc[:, k2]
        k3_ninf = outinf.iloc[:, k3]

        ninferieur = np.array([ouvP_ninf, ouv2_ninf, ouv3_ninf, seg_ninf,
k1_ninf, k2_ninf])
```

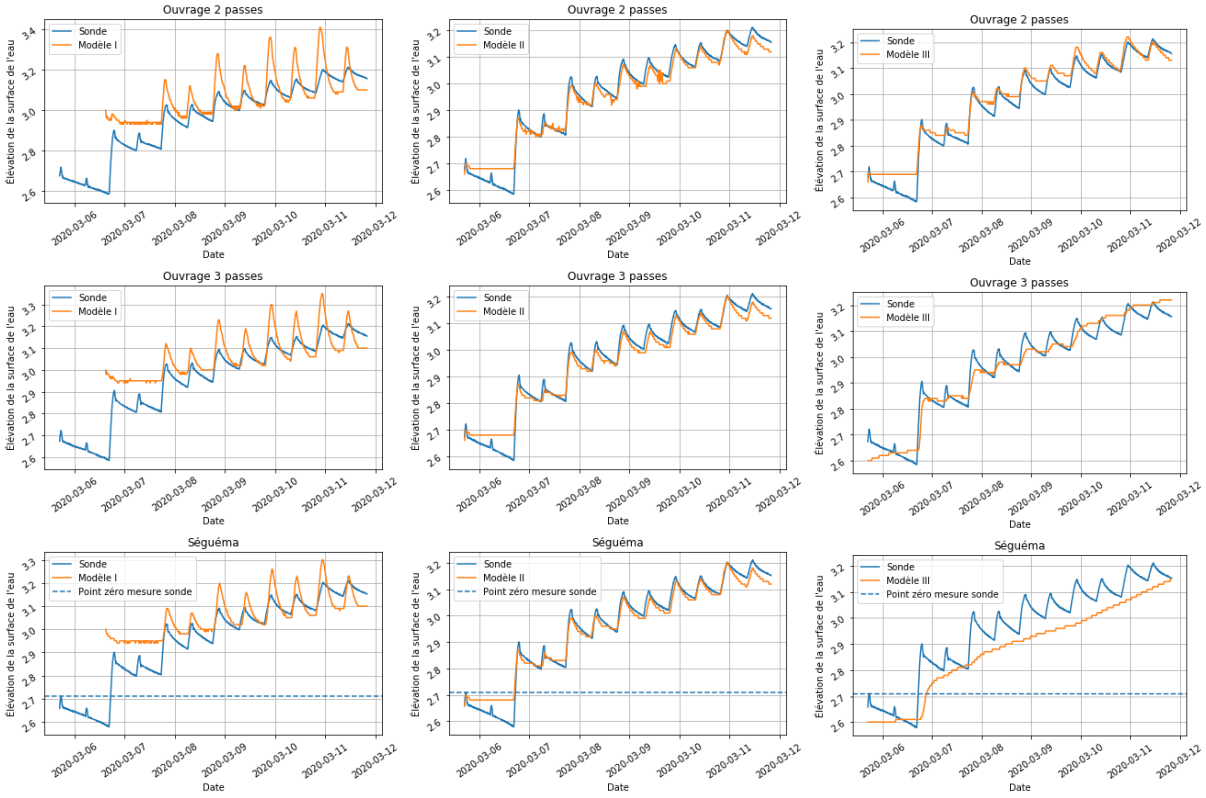
```
        ndiffinf = ninferieur- Sonde
        ndiff_k3 =k3_nsup - S_k3
        MSE_k3 = np.mean(ndiff_k3)
        MSEinf = np.mean(ndiffinf**2)
        RMSEinf = math.sqrt(MSEinf)


        print ('le coefficient de manning supérieur égale à ', round(nsup,
5), 'donne un RMSE de', RMSEsup)
        print ('le coefficient de manninf inférieur égale à ', round(ninf,
5), 'donne un RMSE de', RMSEinf)

        RMSE = min(RMSEsup, RMSEinf)


    elif RMSE > 0.03 :
        if RMSEsup<RMSEinf:
            n = nsup
        else:
            n = ninf

        nsup = n*1.01
        ninf = n*0.99
        print ('n supérieur =', round(nsup, 5))
        print ('n inférieur =', round(ninf, 5))


        # Code avec n Supérieur
        rc.open_project(project_RAS)
        profs = rc.get_profiles()
        print ('Run 1% de nsup = ', round(nsup, 5))

            change.change_mann_xs(round(nsup, 6), fichier_geom,
fichier_geom)
            rc.run_current_plan()

            with open('outnsup{0}.txt'.format(round(nsup, 5)), 'wt') as
outfile:
                rivers = rc.get_rivers()
                for riv in rivers:
                    for reach in riv.reaches:
                        for node in reach.nodes:
                            if node.node_type == '':
                                outfile.write(node.node_id)
                                for prof in profs:
                                    outfile.write(','+str(node.value(prof,
2)))
                                outfile.write('\n')


        # Code avec n Inférieur
        rc.open_project(project_RAS)
        profs = rc.get_profiles()
        print ('Run 1% de ninf = ', round(ninf, 5))

            change.change_mann_xs(round(ninf, 6), fichier_geom,
fichier_geom)
            rc.run_current_plan()

            with open('outninf{0}.txt'.format(round(ninf, 5)), 'wt') as
outfile:
```

```
                rivers = rc.get_rivers()
                for riv in rivers:
                    for reach in riv.reaches:
                        for node in reach.nodes:
                            if node.node_type == '':
                                outfile.write(node.node_id)
                                for prof in profs:
                                    outfile.write(','+str(node.value(prof,
2)))
                                outfile.write('\n')

        # Extraction des résultats
        # n Supérieur
        outsup = pd.read_csv('outnsup{0}.txt'.format(round(nsup, 5)), sep
=",", header = None)
        outsup = outsup.drop([1], axis =1)
        outsup = outsup.set_index(0)
        outsup = outsup.T
        outsup.reset_index(level = 0, inplace = True)
        outsup = outsup.drop ('index', axis = 1)
        ouvP_nsup = outsup.iloc[:, ouvP]
        ouv2_nsup = outsup.iloc[:, ouv2]
        ouv3_nsup = outsup.iloc[:, ouv3]
        seg_nsup = outsup.iloc[:, seg]
        k1_nsup = outsup.iloc[:, k1]
        k2_nsup = outsup.iloc[:, k2]
        k3_nsup = outsup.iloc[:, k3]

        nsuperieur = np.array([ouvP_nsup, ouv2_nsup, ouv3_nsup, seg_nsup,
k1_nsup, k2_nsup])
        ndiffsup = nsuperieur - Sonde
        MSEsup = np.mean(ndiffsup**2)
        RMSEsup = math.sqrt(MSEsup)

        #n inférieur
        outinf = pd.read_csv('outninf{0}.txt'.format(round(ninf, 5)), sep
=",", header = None)
        outinf = outinf.drop([1], axis =1)
        outinf = outinf.set_index(0)
        outinf = outinf.T
        outinf.reset_index(level = 0, inplace = True)
        outinf = outinf.drop ('index', axis = 1)
        ouvP_ninf = outinf.iloc[:, ouvP]
        ouv2_ninf = outinf.iloc[:, ouv2]
        ouv3_ninf = outinf.iloc[:, ouv3]
        seg_ninf = outinf.iloc[:, seg]
        k1_ninf = outinf.iloc[:, k1]
        k2_ninf = outinf.iloc[:, k2]
        k3_ninf = outinf.iloc[:, k3]

        ninferieur = np.array([ouvP_ninf, ouv2_ninf, ouv3_ninf, seg_ninf,
k1_ninf, k2_ninf])
        ndiffinf = ninferieur- Sonde
        ndiff_k3 =k3_nsup - S_k3
        MSE_k3 = np.mean(ndiff_k3)
        MSEinf = np.mean(ndiffinf**2)
        RMSEinf = math.sqrt(MSEinf)


        print ('le coefficient de manning supérieur égale à ', round(nsup,
5), 'donne un RMSE de', RMSEsup)
```

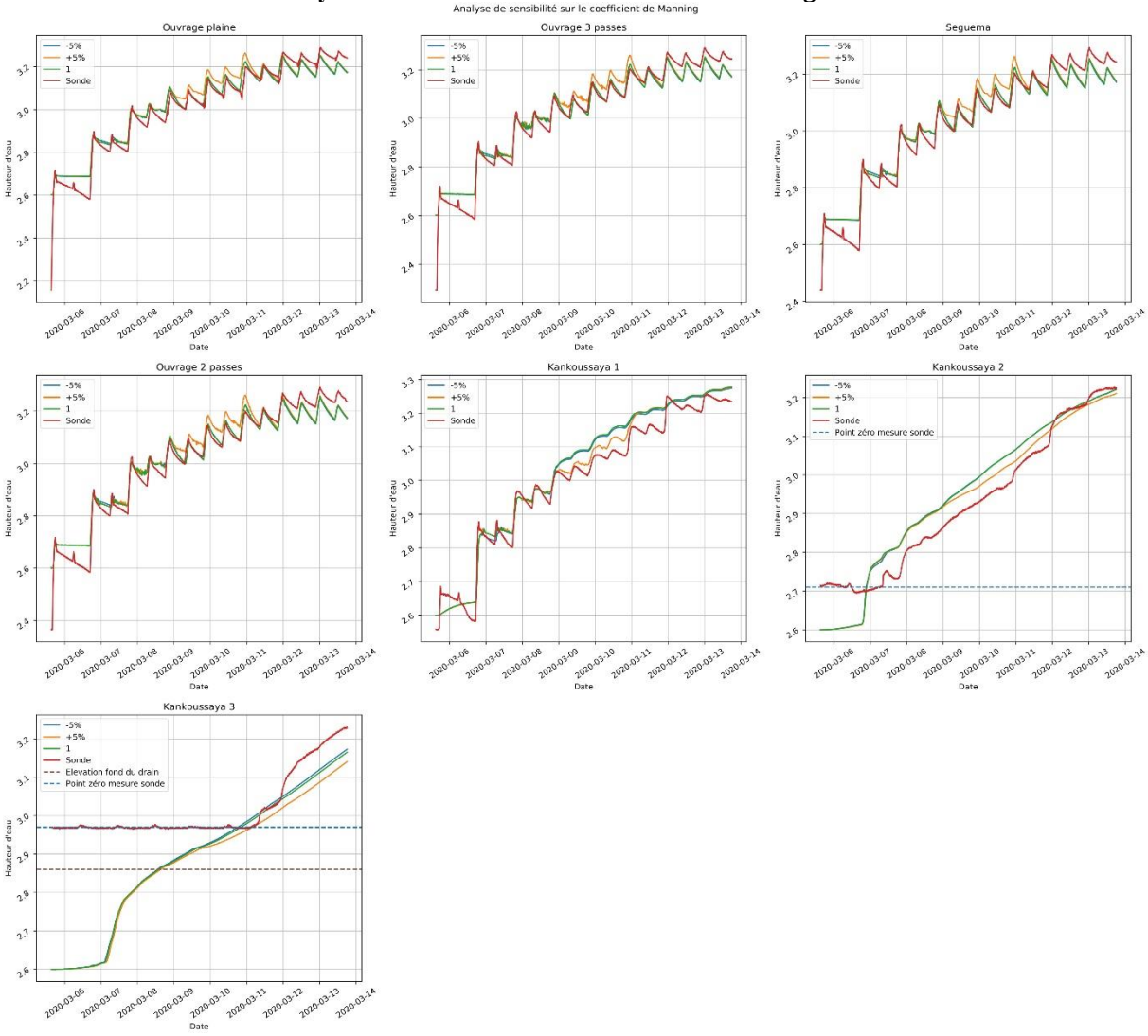
```
        print ('le coefficient de manninf inférieur égale à ', round(ninf,
5), 'donne un RMSE de', RMSEinf)

        RMSE = min(RMSEsup, RMSEinf)

"""
Calcul des TIC pour la validation
Les fichiers de résultats ont été obtenus en utilisant HEC-RAS
manuellement.
"""

# Ouverture du fichier de données des sondes

sonde_tot = pd.read_excel("sonde_valid.xlsx")
sonde_tot.head(n=1) # Donne noms au colonne

S_ouvplaine = sonde_tot['Ouvrage plaine']
S_ouv2 = sonde_tot['Ouvrage 2 passes']
S_ouv3 = sonde_tot['Ouvrage 3 passes']
S_seg = sonde_tot['Seguema']
S_k1 = sonde_tot['Kankoussaya 1']
S_k2 = sonde_tot['Kankoussaya 2']
S_k3 = sonde_tot['Kankoussaya 3']
Date_5 = sonde_tot['Date']
S_mer = sonde_tot['Ouvrage mer']
Sonde = np.array([S_ouvplaine, S_ouv2, S_ouv3, S_seg, S_k1, S_k2])
RMSESonde = math.sqrt(np.mean(Sonde**2))

# Extraction des données de modélisation calibrée
file = "valid_c.csv"

outsup = pd.read_csv(file, sep = ",", header = None)
outsup = outsup.set_index(0)
outsup = outsup.T
ouvP_n_05 = outsup[1545]
ouv2_n_05 = outsup[2880]
ouv3_n_05 = outsup[153]
seg_n_05 = outsup[861]
k1_n_05 = outsup[2190]
k2_n_05 = outsup[1245]
k3_n_05 = outsup[150]

nsuperieur = np.array([ouvP_n_05, ouv2_n_05, ouv3_n_05, seg_n_05, k1_n_05,
k2_n_05])
ndiffsup = nsuperieur - Sonde
MSEsup = np.mean(ndiffsup**2)
RMSEsup = math.sqrt(MSEsup)
print ('RMSE = ', RMSEsup)

RMSEModel = math.sqrt(np.mean(nsuperieur**2))
TIC_cal = RMSEsup/(RMSESonde + RMSEModel)

# Extraction des données de modélisation non calibrées
file = "valid_nc.csv"

outsup = pd.read_csv(file, sep = ",", header = None)
outsup = outsup.set_index(0)
outsup = outsup.T
ouvP_n_15 = outsup[1545]
ouv2_n_15 = outsup[2880]
ouv3_n_15 = outsup[153]
```

```
seg_n_15 = outsup[861]
k1_n_15 = outsup[2190]
k2_n_15 = outsup[1245]
k3_n_15 = outsup[150]

nsuperieur = np.array([ouvP_n_05, ouv2_n_05, ouv3_n_05, seg_n_05, k1_n_05,
k2_n_05])
ndiffsup = nsuperieur - Sonde
MSEsup = np.mean(ndiffsup**2)
RMSEsup = math.sqrt(MSEsup)
print ('RMSE = ', RMSEsup)

RMSEModel = math.sqrt(np.mean(nsuperieur**2))
TIC_ncal = RMSEsup/(RMSESonde + RMSEModel)
```

Annexe 2 – Comparaison entre la simulation des trois différents modèles et l'élévation de l'eau mesurée par les sondes



Annexe 3 – Résultat de l'analyse de sensibilité sur le coefficient de Manning



Annexe 4 – Résultat de l'analyse de sensibilité sur le coefficient de déversement latéral