GNG1506 – Notions fondamentales du traitement de l’information en génie

Projet de cours

*Génie civil - Groupe CVG 4*

Joseph AMOUSSOU-GUENOU et Adam Taktek

Date:

# **1** **L’identification et l’énoncé du problème**

Nous avons un barrage qui bloque l’eau d’un canal en amont. Nous voulons connaître la valeur de la force f exercé par l’eau sur le barrage. Pour faire cela, on veut pour une étude pouvoir choisir les différentes dimensions du barrage, du canal, et puis l’ensemble de valeur (en choisissant un minimum, un maximum et une incrémentation) pour la profondeur d de l’eau.  
 L’étude fera le calcul de la force totale f pour chaque valeur de l’ensemble des valeurs de profondeur d compris entre le minimum et la maximum et distante de la valeur précédente par la valeur de l’incrémentale choisi. Les résultats seront stockés dans un tableau. Puis ces résultats seront représentés par une courbe présentant la variation de la force totale f en fonction de la profondeur. Nous devons aussi calculer dans notre programme la taille du pas utilisé pour la règle trapézoïdale.  
 Le logiciel devra aussi permettre à l’utilisateur de sauvegarder cinq fichiers différents qui ont les valeurs d’entrées sauvegarder. Au démarrage l’utilisateur doit pouvoir choisir soit d’ouvrir l’un des fichiers ou d’entrée des nouvelles valeurs.

Pour le test des valeurs trouvées:  
On doit aussi créer des cas de test avec la solution analytique données, puis comparer avec les valeurs qui sont trouvées par le programme avec la formule de la règle trapézoïdale.

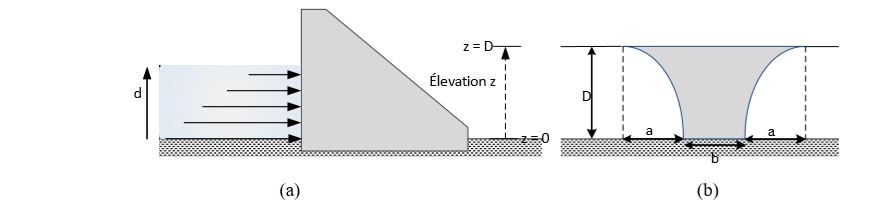
# **2** **La cueillette d’information et la description d’entrées/sorties**

## **2.1** **Sous section 1- Données**

* Constante gravitationnel **G = 9.8 m/s**
* rho - density (ou masse volumique en français ) : **p = 1000 Kg/m^3**

## 

## **2.2** **Sous section 2- Comment trouver la formule de la force exercés**

Scéma du cas:

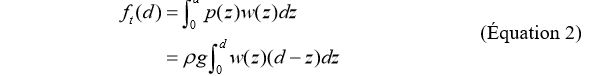
On a les formules préalables nécessaires :

* **Formule de la pression P(z)** en fonction de la distance du fond de l’eau z :   
    
   **P(z)= pG(d-z) (1)  
   P(z)** en N/m^2  
   Avec **z en mètres** Avec **d l’élévation (en m)** de l’eau au-dessus du fond du canal.
* **Et w(z) la largeur de de la face du barrage** (m) à l’élévation z



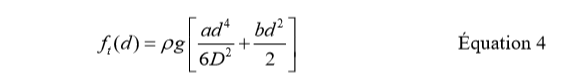
a est la distance du bord de la base du canal à la rive du canal  
 b est la largeur de la base du canal;   
D est la profondeur du canal.  
 **Enfin, la formule de la force exercés:**

* **Formule de la force exercé** sur le barrage par l’eau



* + Où on a **p(z) la formule** **(1)** et **w(z) la formule (3)**

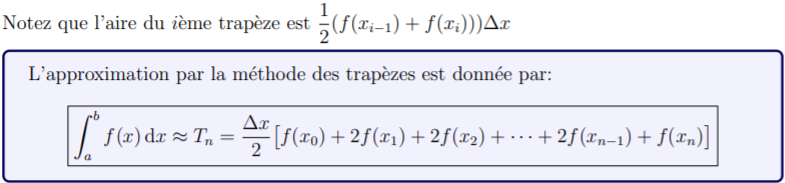
**2.3** **Sous section 3 - Calcul de la force avec la solution analytique**

Avec les formules précédentes on peut trouver la solution analytique.  


**La solution analytique** **permet de faire les cas de test pour tester le programme.**

## **2.4** **Sous section 4 - Calcul de la force avec la formule trapézoïdale**

Formule de la règle trapézoïdale



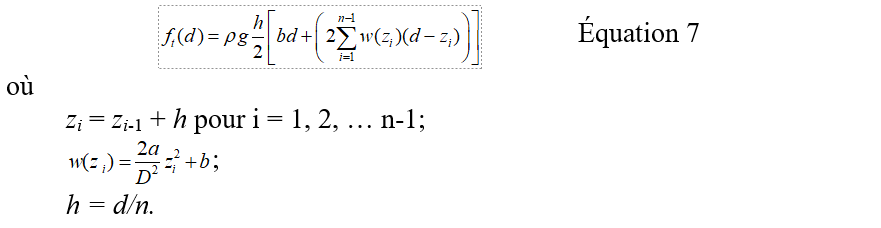
Calcul de la taille du pas

Le pas correspond à la distance que l’on doit prendre par trapèze au niveau de l’axe des x.

Pour le calculer on doit diviser la variation de l'intervalle sur laquelle on veut approximer l’intégrale par le nombre de sous intervalle que l’on souhaite obtenir. Soit la formule qui suit:

##### Soit a le pas et l’intervalle de calcul de l’intégrale est [xi;xf]: a = (xf-xi)/n.

Dans notre cas la formule obtenue pour le calcul de la forc par la méthode trapézoïdale est :



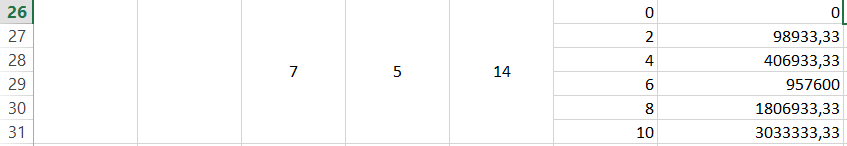
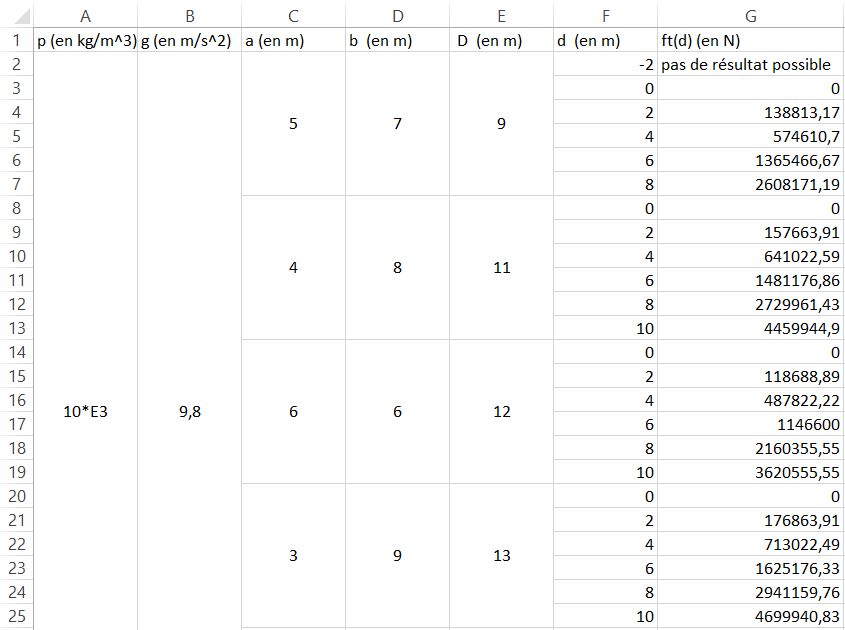
On peut donc utiliser une boucle “for” dans le programme pour calculer la valeur de de la force exercé avec cette formule de récurrence (7).

## **2.4** **Entrées et sorties**

* **Entrées:**
  + **Le choix du type de l’étude** de soit ouvrir un fichier existant ou soit de faire des nouvelles entrées.
  + **Nom du canal**
  + Pour l’**étendue de la profondeur d de l’eau** du canal :  
     (hauteur/profondeur du canal)  
     -Un **Minimum** **de d**   
     -Un **Maximun** **de d** - Une valeur d’**Incrementaiton** **de d**
  + La **valeur a de la distance** du bord de la base du canal à la rive du canal
  + La **valeur b de la largeur de la base** du canal
  + La **valeur D de la profondeur** du canal
* **Sorties:**
  + **Une courbe** présentant la variation de ft(d) en fonction des différentes valeurs **d**
  + **Les valeurs** des différentes entrées de l’utilisateurs
  + **La sauvegarde des données** dans un fichiers

# **3** **Les cas tests et la conception**

## **3.1** **Cas tests**



Le tableur excel nous montre les cas test calculé à l’aide de la formule analytique. Nous remarquons que pour des valeurs de d inférieures à zéro le calcul n’est pas possible. Il en est de même pour D, a, b car toutes ces grandeurs sont des distances et ne peuvent donc pas êtres négatives.

## **3.2** **Conception**

### **3.2.1** **Introduction**

Le logiciel utilisera une structure pour obtenir les donnés de l'utilisateur.   
En plus de main, il y aura:  
- une fonction pour obtenir les données de l‘utilisateurs  
- une fonction qui obtient la valeur positive  
  
- une fonction pour remplir le tableau des valeurs des forces avec la solution analytique  
  
- une fonction pour remplir le tableau des valeurs des forces pour la règle trapézoïdales,   
- une fonction calcul de la force exercé avec la règle trapézoïdale,

- une fonction qui trouve la valeur maximale de la force dans un tableau   
- une fonction plot qui trace un graphique avec deux courbes

- Un fonction pour choisi d’ouvrir un fichier ou non  
- Une fonction pour sauvegarder dans un fichier

**Fichier d’en tête:**

#include <stdio.h>  
#include <stdlib.h>  
#include <math.h>  
#include <float.h>  
#include "gng1106plplot.h"

**Constantes:**

#define G 9.81 **:** valeur de la constante gravitationnelle   
#define P 1000 : valeur de la masse volumique de l’eau  
#define FILE1 “Etude1.bin”  
#define FILE2 “Etude2.bin”  
#define FILE3 “Etude3.bin”  
#define FILE4 “Etude3.bin”  
#define FILE5 “Etude5.bin”

**Structures:**

La structure CANAL contiendra:

* nom: Le nom du canal, ou de l’étude
* dmin, dmax : les valeurs de la profondeur minimun et maximun de l’eau entrées par l’utilisateur.
* Inc : Valeur de l’incrémentation de la profondeur de l’eau entrée par l’utilisateur.
* hauteurD : La hauteur du barrage et du canal entrée par l’utilisateur
* largeurCanal : La largeur de la base du canal entrée par l’utilisateur
* distanceCanal : La distance du bord de la base du canal à la rive du canal entrée par l’utilisateur
* nbr : Le nombre de valeur dans les tableaux
* profondeur : tableau contenant les valeurs de la profondeur
* forcetrapezoide, forceanalytique : Les tableaux contenant les valeurs de la force exercés sur le barrage calculé avec respectivement dans l’ordre la solution analytique et la règle trapézoïdale

**Prototypes:**

Void main (void)

Void ObtientEntree (CANAL \*canal)

Double Getvaleurpositive (char \*invitation)

Void RemplirForceAnalytique (CANAL\*canal)

Void RemplirForceTrapezoide (CANAL\*canal)

double CalculForceTrapezoide(double d, double n, CANAL \* canal)

Void Plot (CANAL \*canal)

double getMaxDouble ( n, double tbl[n])

int choixFichier (CANAL \* canal)

int sauvegardeEtude (CANAL \* canal)

**3.2.2** **Fonctions pour l’interaction avec l’utilisateur**

Void main (void)

* **Paramètres:** void
* **Retour:** Void
* **Logique/Algorithme -Instructions:**Cette fonction est celle qui fait les appels à toutes les autres fonctions qui vont permettre l’étude sur la force exercé sur le barrage selon les valeurs de la profondeur entrée par l’utilisateur et permettre à l’utilisateur d’ouvrir un fichier précédemment sauvegardé.
  + Déclare un structure CANAL canal;
  + Déclare int choix;
  + Affecte à choix=ChoixFichier(&canal) pour donner le choix à l’utilisateur de soit ouvrir un fichier et de choisir lequel ou sinon de faire une nouvelle étude
  + Instruction logique if (choix==0)
    - Appel ObtienEntree pour remplir les membres de la variable structure CANAL qui doivent être entrée par l’utilisateur
  + Affecte à la variable canal.nbr la valeur (canal.dmax-canal.dmin)/(canal.Inc)
  + Appel RemplirForceAnalytique pour remplir le tableau des forces exercés sur le barrage par la solution analytique
  + Appel RemplirForceTrapezoide pour remplir le tableau des forces exercés sur le barrage par la règle trapézoïdale
  + Appel Plot pour tracer le graphe avec deux courbes chacune étant soit la fonction de la solution analytique soit la fonction du calcul de la Force avec la règle trapézoïdale
  + Appel comme ceci int inutile= sauvegardeEtude pour choisir de sauvegarder ou non les données.

Void ObtientEntree (CANAL \*canal)

* **Paramètres:**CANAL canal
* **Retour:** Void
* **Logique/Algorithme -Instructions:**
  + Faire une boucle do while :
    - Affecté à canal->dmin, canal->dmax, canal->Inc, canal->hauteurD, canal->largeurCanal,canal->distanceCanal la fonction getValeurpositive
    - Faire une boucle if tel que
      * “Valeurs impossibles” sera imprimés si :

canal->dmin < canal->dmax ;

canal->dmax < canal->hauteurD ;

canal->Inc =! 0,

canal->hauteurD =! 0,

canal->largeurCanal =! 0,

canal->distanceCanal =! 0

* + La boucle do while continue tant que ces conditions

canal->dmin < canal->dmax

canal->dmax < canal->hauteurD

canal->Inc =! 0,

canal->hauteurD =! 0,

canal->largeurCanal =! 0,

canal->distanceCanal =! 0

ne sont pas toutes respectés.

Double Getvaleurpositive (char \*invitation)

* **Paramètre** : char \*invitation
* **Retour:** double
* **Logique/Algorithme -Instructions:**

Cette fonction nous permettra de donner à D, d, a et b des valeurs supérieures ou égales à zéro car c’est grandeurs sont des distances.

* Boucle do while qui va :
  + Imprimer invitation
  + Nous demander de donner une valeur qui sera
    - Retourné si elle est supérieur ou égale à 0
    - Refusé si elle est inférieur à 0
  + La boucle continue tant que valeur est inférieur à zéro

### **3.2.3** **Fonctions de calcul**

Void RemplirForceAnalytique (CANAL\*canal)

* **Paramètre** : pointeur à canal. On utilise les variables suivantes
  + canal->profondeur
  + canal ->forceanalytique
  + canal->dmin, canal->dmax, canal->Inc, canal->nbr
  + canal->hauteurD, canal->largeurCanal, canal->distanceCanal,
* **Valeur de retour:** void
* **Logique/Algorithme:**Cette fonction permet de remplir les valeurs de force dans canal->forceanalytique associé à chaque élément de profondeur dans le tableau canal->profondeur avec la formule de la solution analytique.
  + Déclare double et affecte p=canal->dmin
  + Boucle for de i= 0 à canal->nbr-1 avec i incrémenter de 1 à chaque répétition
    - Dans l’élément canal->profondeur[i] affecter la valeur p
    - Dans l’élément canal->forceanalytique[i] entrer la valeur P\*G\*((canal->distanceCanal\*p)/(6\*pow(canal->hauteurD,2)+(canal->largeurCanal\*p\*p)/2)
    - Affecter p=p+canal.Inc
    - Fin de la boucle

Void RemplirForceTrapezoide (CANAL\*canal)

* **Paramètre** : pointeur à canal. On utilise les variables suivantes
  + canal-> profondeur
  + Canal->forcetrapezoide
  + canal->nbr
* **Valeur de retour :** Void
* **Logique/algorithme :**  
  Un appel à la fonction CalculForceTrapezoide  
  + Boucle for allant de ix = 0 à canal->nbr-1 avec ix= ix+1

##### Dans Canal ->forcetrapezoide[ix] affecter la valeur retourne par la fonction CalculForceTrapezoide (canal->profondeur[ix], ix, &canal)

double CalculForceTrapezoide(double d, double n, CANAL \* canal)

* **Paramètre :**
  + canal-> distanceCanal
  + canal -> HauteurD
  + canal -> largeurCanal
  + d
  + n
* **Valeur de retour :** double
* **Logique/algorithme :**
  + Déclare double Ft=0 (variable retourné), Ein = 0 (sigma, somme partiel dans la formule trapézoïdale), h=0 (calcul du pas), w=0 (largeur de la face du barrage a un instant), z=0 (valeur de la profondeur a un instant)
  + h = d/n
  + Boucle for pour i de 1 à n-1
    - z= z+h
    - w = (2\*canal->distanceCanal/(pow(canal->HauteurD, 2))\*z\*z+ canal->largeurCanal
    - Ein = Ein + w\*(d-z)
    - Fin boucle for
  + Ft = P\*G\*(h/2)\*(canal->largeurCanal\*d+(2\*Ein))
  + Retourne la valeur Ft.

**3.2.4** **Fonctions pour la création des graphes**

Void Plot (CANAL \*canal)

* **Paramètre** : CANAL \*canal
* **Retour** : Void
* **Logique/algorithme** :
  + Déclarer une variable double MaxForce représentant la force maximale et faire MaxForce=getMaxDouble(canal->nbr,Canal ->forceanalytique)
  + Déclarer une variable double temporaire représentant la force maximale et faire temporaire = getMaxDouble(canal->nbr,Canal ->forcetrapezoide)
  + Déclarer une variable char etiquette représentant le nom du graphe
  + Faire une boucle if tel que si temporaire>MaxForce on ait temporaire=MaxForce
  + Faire ensuite MaxForce=1.1\*MaxForce
  + Donner à etiquette le nom Force à l’aide de la fonction sprintf
  + Tracer ensuite deux graphes
    - Un de couleur rouge représentant le calcul de la force par la méthode analytique
      * Dans plline mettre pour arguments:Canal ->forceanalytique,canal-> profondeur,canal->nbr
      * Avoir dans plptext comme arguments:
        + 0.1\*(canal->dmax - canal->dmin)+canal->dmin
        + 0.9\*MaxForce
        + 0
        + 0
        + 0
        + “Analytique”
    - Une de couleur bleue représentant le calcul de la force par la méthode des trapèzes
      * Dans plline mettre pour paramètre:Canal ->forcetrapezoide,canal-> profondeur,canal->nbr
      * Avoir dans plptex comme argument
        + 0.3\*(canal->dmax - canal->dmin)+canal->dmin
        + 0.9\*MaxForce
        + 0
        + 0
        + 0
        + “Trapezoidal”

double getMaxDouble ( n, double tbl[n])

##### **Paramètre** : Double tbl[] ,int n

* **Retour** : Double
* **Logique/algorithme** :
  + Déclarer une variable double Max pour stocker la valeur maximale et faire max = -DBL\_MAX
  + Faire une boucle for avec ix = 0 ; ix <n ; ix = ix + 1:

##### Faire une instruction logique if tel que si max < tbl[ix] on a max = tbl[ix]

### 

### **3.2.5** **Fonctions pour manipuler les fichiers**

int choixFichier (CANAL \* canal)

* **Paramètre** : Canal \*canal
* **Retour** : Int
* **Logique/algorithme** :  
  Cette fonction est très compliqué on va alors la détaillé amplement pour la conception:  
  + char a, b, file;
  + int nfichier= 0, erreur=0;
  + FILE \* fptr
  + Double Donnee[6];
  + do {
  + erreur=0;
  + Imprimez “Voulez vous ouvrir un fichier ? (o/n):
  + a=getchar()
  + Instruction logique if pour si a = ‘o’ ou a= ‘O’ faire:
    - Imprimez “Quel fichier voulez vous ouvrir ? (1 - 5 ):
    - Scanner la valeur tapee et entrez la dans nfichier
    - Instruction switch (nfichier)
      * case 1: file = FILE1;
      * break
      * case 2: file = FILE2;
      * break
      * case 3: file = FILE3;
      * break
      * case 4: file = FILE4;
      * break
      * case 5: file = FILE5;
      * break
      * Fin switch
    - fptr = fopen(file, ‘rb’);
    - Instruction if (fptr == NULL)
      * Imprimez “Fichier inexistant, Veuillez faire un nouveau choix ”
      * erreur =1
    - Instruction else if (fptr != NULL)
      * fread(&Donnee,sizeof(Donnee)/sizeof(double),6,fptr)
      * Imprimez “Profondeur minimale: %f”, Donnee[0]
      * Imprimez “Profondeur maximale: %f”, Donnee[1]
      * Imprimer “Increment entre chaque valeur de profondeur: %f”, Donnee[2]
      * Imprimez “Hauteur du barrage:%f”, Donnee[3]
      * Imprimez “largeur du canal: %f”, Donnee[4]
      * Imprimez “distance du canal : %f”, Donne[5]
      * Do {
      * Imprimez “Etes-vous satisfait avec ces donnees? (o/n):”
      * b =getchar ();
      * If (b== ‘o’ ||b== ‘O’)
        + canal->dmin = Donnee[0];
        + canal->dmax = Donnee[1];
        + canal->Inc = Donnee[2];
        + canal->hauteurD = Donnee[3];
        + canal->largeurCanal = Donnee[4];
        + canal->distanceCanal = Donnee[5];
        + Retour 1;
      * If (b==n || b==N)
        + erreur = 1;
        + Imprimez “Vous pouvez alors faire un nouveau choix”
      * while (b != ‘o’ && b != ‘O’ && b != ‘n’ && b != ‘N’ )
  + Else if (a == ‘n’ || a== ‘N’)
    - retour 0;
  + }while (erreur ==1 || a != ‘n’ && a != ‘N’ && a !=‘o’ && a !=‘O’)

int sauvegarder (CANAL \* canal)

* **Paramètre** : Canal \*canal
* **Retour** : int
* **Logique/algorithme** :  
  Cette fonction est très compliqué on va alors la détaillé amplement pour la conception
  + char a, b;
  + Int nfichier;
  + Int reussi = 0, manqueMemoire=0;
  + FILE \*fptr;
  + Int i =1 ;
  + Double Donnee [6];
  + Do {
  + Imprimez “Voulez-vous sauvegarder dans un fichier? (o/n):”
  + a=getchar();
  + If ( a == ‘o’ || a== ‘O’)
    - Donnee[0]=canal->dmin;
    - Donnee[1]=canal->dmax;
    - Donnee[2]=canal->Inc;
    - Donnee[3]=canal->hauteurD;
    - Donnee[4] = canal->largeurCanal;
    - Donnee[5] = canal->distanceCanal;
    - Do {
    - switch (i):
    - case 1 : fptr=fopen ( FILE1, rb)
    - If (fptr ==NULL)
      * Reussi =1
    - else
      * fclose(fptr)
    - Break;
    - case 2 : fptr = fopen (FILE2, rb)
    - If (fptr ==NULL)
      * Reussi =1
    - else
      * fclose(fptr)
    - Break;
    - case 3 : fptr = fopen (FILE3, rb)
    - If (fptr ==NULL)
      * Reussi =1
    - else
      * fclose(fptr)
    - Break;
    - case 4 : fptr = fopen (FILE4, rb)
    - If (fptr ==NULL)
      * Reussi =1
    - else
      * fclose(fptr)
    - Break;
    - case 5 : fptr = fopen (FILE5, rb)
    - If (fptr ==NULL)
      * Reussi =1
    - else
      * fclose(fptr)
    - Break;
    - If (i >5)
      * Imprimez “Vous n’avez plus de memoire!”
      * manqueMemoire = 1
    - i += 1;
    - }while (reussi==0 && manqueMemoire== 0)
    - If (manqueMemoire==0)
      * fwrite (Donnee, sizeof(Donne)/sizeof(double), 6, fptr);
      * fclose(fptr)
    - else if (manqueMemoire==1)
      * Do {
      * Imprimez “Voulez -vous remplacer un fichier? (o/n):”
      * b=getchar()
      * if (b ==o || b ==O)
        + Imprimez “lequel voulez-vous remplacer? (1-5):”
        + Scanner la valeur taper dans nfichier
        + Instruction switch (nfichier)

case 1: file = FILE1;

break

case 2: file = FILE2;

break

case 3: file = FILE3;

break

case 4: file = FILE4;

break

case 5: file = FILE5;

break

Fin switch

If ( nfichier <=5 && nfichier >=1)

fptr = fopen(file, ‘wb’);

If (fptr ==NULL)

Imprimez “impossible de creer le fichier”

Fwrite(Donnee, sizeof(Donnee)/sizeof(double),6, fptr)

fclose(fptr)

Else

a= ‘e’

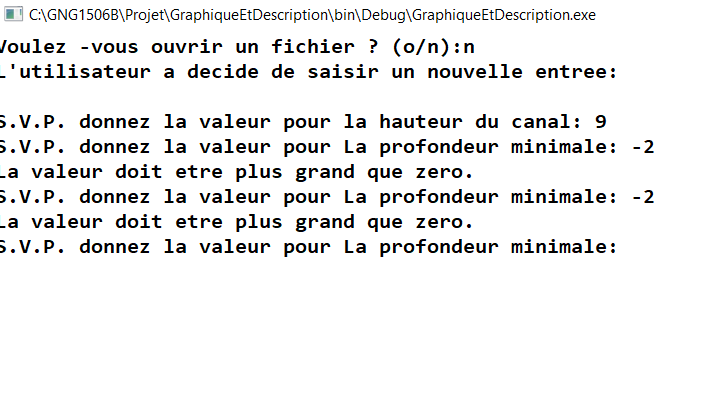
* + - * }while (b != ‘o’ && b != ‘O’ && b!= ‘n’ && b!= ‘N’)
  + }while ( a != o && a!= O && a != n && a!= N)
  + retour 0

# **4 Implémentation**

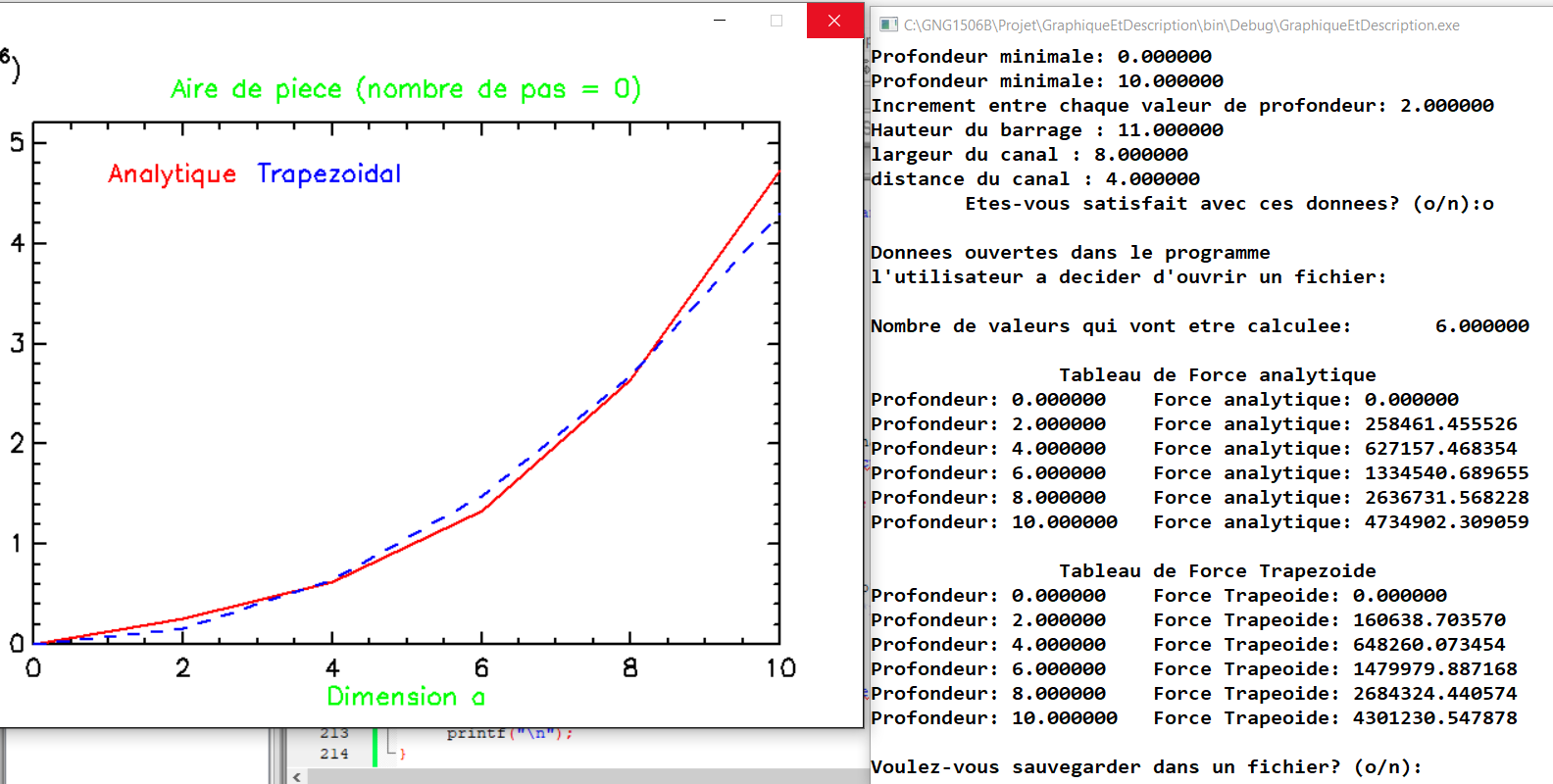
(voir le fichier de code c ci-joint)

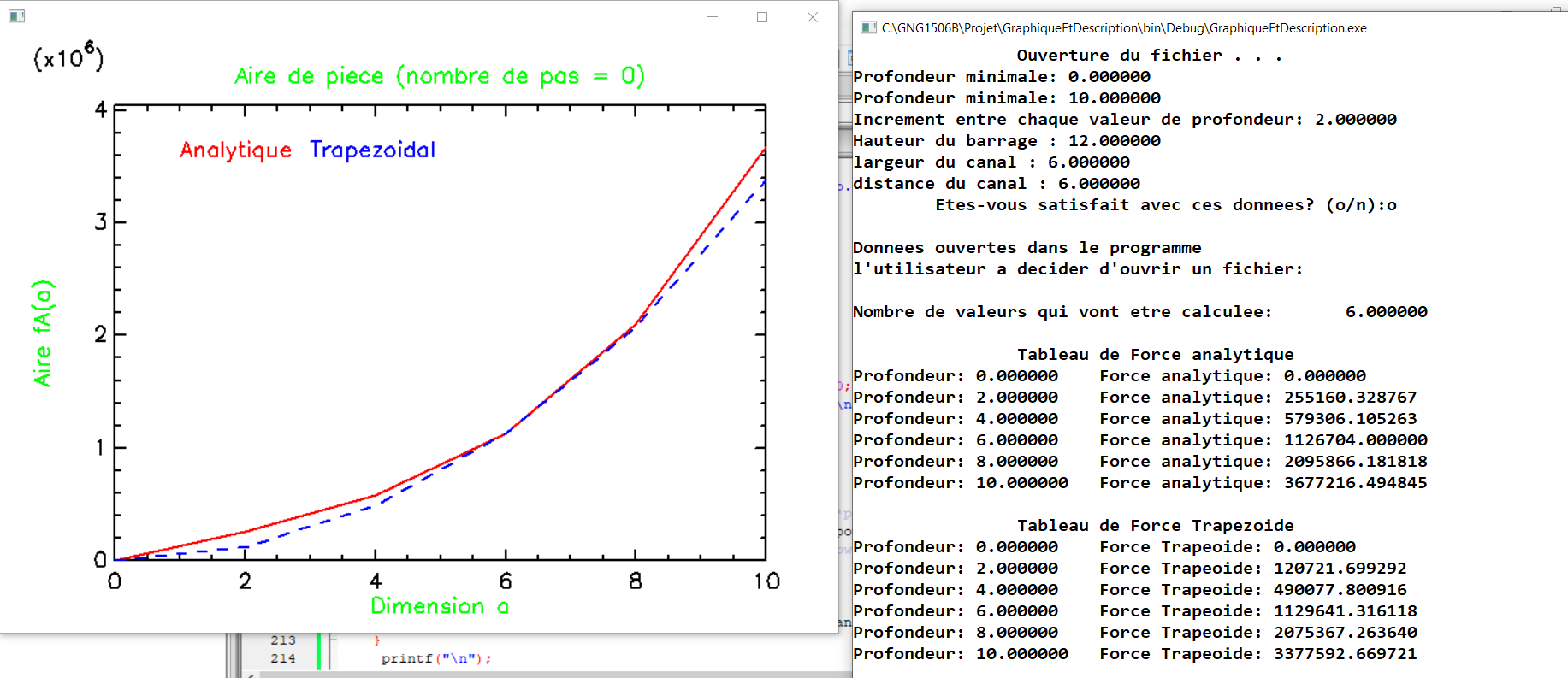
# **5** **Tests et vérification**

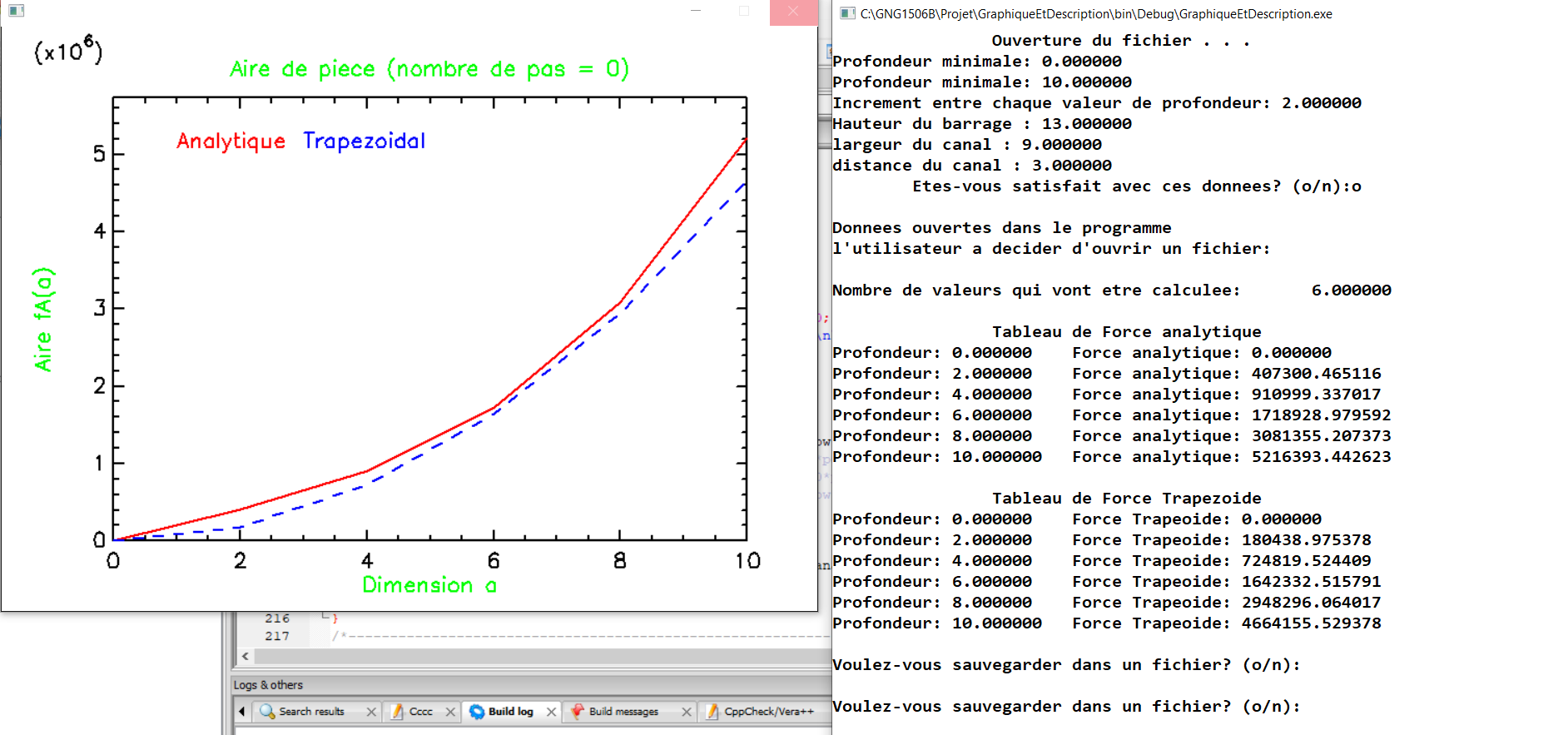
Consoles -TEST:

1-

Pour ce test il est impossible d’avoir de valeur, ce qui est conforme au cas de test trouvé avec excel.

2- 

3-

4-

5-