INSTITUT NATIONAL DES SCIENCES APPLIQUÉES ROUEN

Projet de Physique P6
STPI/P6/2016 – n°02

Mouvement de chute libre: Le grand saut de Felix Baumgartner

Etudiants :

Eleonore URIEN **Mona MENADI**
Sara GRAOUNA **Sylvain LECLERC**

Enseignant-responsable du projet :
Ahmad AZZAM

Date de remise du rapport : **13/06/2016**

Référence du projet : ***STPI/P6/2016– 02***

Intitulé du projet : Le ***Grand saut de Felix Baumgartner : Chute Libre***

Type de projet : ***Modélisation***

Objectifs du projet :

L'intégralité de notre projet repose sur la démonstration, en premier lieu, des fonctions qui permettent de décrire le phénomène de chute libre pour ensuite modéliser la trajectoire de Felix Baugmartner.

Mots-clefs du projet : ***Chute libre, démonstration et modélisation***

INSTITUT NATIONAL DES SCIENCES APPLIQUEES DE ROUEN
Département Sciences et Techniques Pour l'Ingénieur
avenue de l'Université - 76801 Saint Etienne du Rouvray - tél : +33 (0)2 32 95 97 00 - fax : +33 (0)2 32 95 98 60

Table des matières

1.Introduction

Le thème de cette étude est «La chute de Felix Baumgartner».

Brièvement, nous pouvons rappeler que le projet de Felix Baumgartner était de réaliser un saut depuis la stratosphère, à une altitude de 40km, au départ d'une capsule suspendue à un ballon gonflé à l'hélium, avec l'intention d'être le premier parachutiste à passer le mur du son.

Nous pouvons ainsi dire, qu'à travers cette expérience, Felix Baumgartner a relevé un défi de taille qui n'avait jusque là jamais été accompli.

Pour étudier sa chute, nous commencerons par définir et modéliser les différents paramètres physiques ayant un impact sur celle-ci. Ensuite nous l'étudierons sans prendre en compte les frottements que le parachutiste subit lors de sa descente. Et enfin nous visualiserons la modélisation informatique de sa chute soumise aux différents frottements qui interviennent.

2.Méthodologie / Organisation du travail

Cette étude a été réalisée par 4 étudiants de STPI 2. Nous avons réalisé ce dossier grâce à l'EC « Projet Physique» enseigné par M.Azzam.

Dans l'optique de ce cours, nous avons réalisé ce projet durant des Travaux Dirigés, mais aussi pendant notre temps personnel.

Pour le déroulement du travail :
En premier lieu nous avons cherché les fonctions nécessaires afin de décrire la représentation de la chute de Felix Baumgartner.
Puis pendant qu'une personne résolvait ses équations grâce à la méthode d'Euler et programmait les fonctions, les autres se sont attelés à démontrer toutes les formules utilisées.
Enfin nous avons exploité les courbes obtenues par programmation et rédigé ce rapport

Pour l'organisation des tâches :
Mona a programmé toutes les fonctions numériquement et a édité les courbes relatives avec le langage Fortran 90.
Eleonore, Sara et Sylvain se sont occupés de rechercher, de démontrer toutes les formules et d'analyser les courbes obtenues.

3.Travail réalisé et résultats

3.1. Étude physique générale :

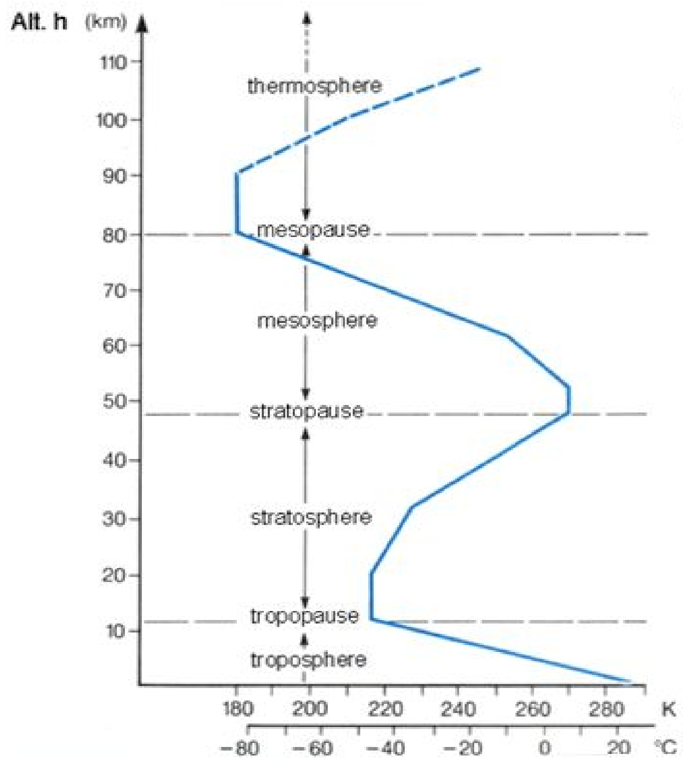
Différents paramètres influent sur la trajectoire et la vitesse du parachutiste en chute libre. Avant de commencer notre étude il est nécessaire de prendre en compte tous ces facteurs.

Nous allons donc étudier, la température en fonction de l'altitude dans le cas d'une atmosphère à gradient constant, la vitesse du son en fonction de l'altitude ainsi que l'intensité de pesanteur en fonction de l'altitude.

3.1.1. Température en fonction de l'altitude (atmosphère à gradient constant):

La température diminue avec l'altitude. Et le gradient de température varie selon les conditions climatiques et météorologiques. Pour notre étude on prendra ce gradient constant . Donc dans ce modèle, on considère que la température T décroît linéairement avec l'altitude:

$$T(z) = To + a*z$$



Courbe de la température atmosphérique en fonction de l'altitude.

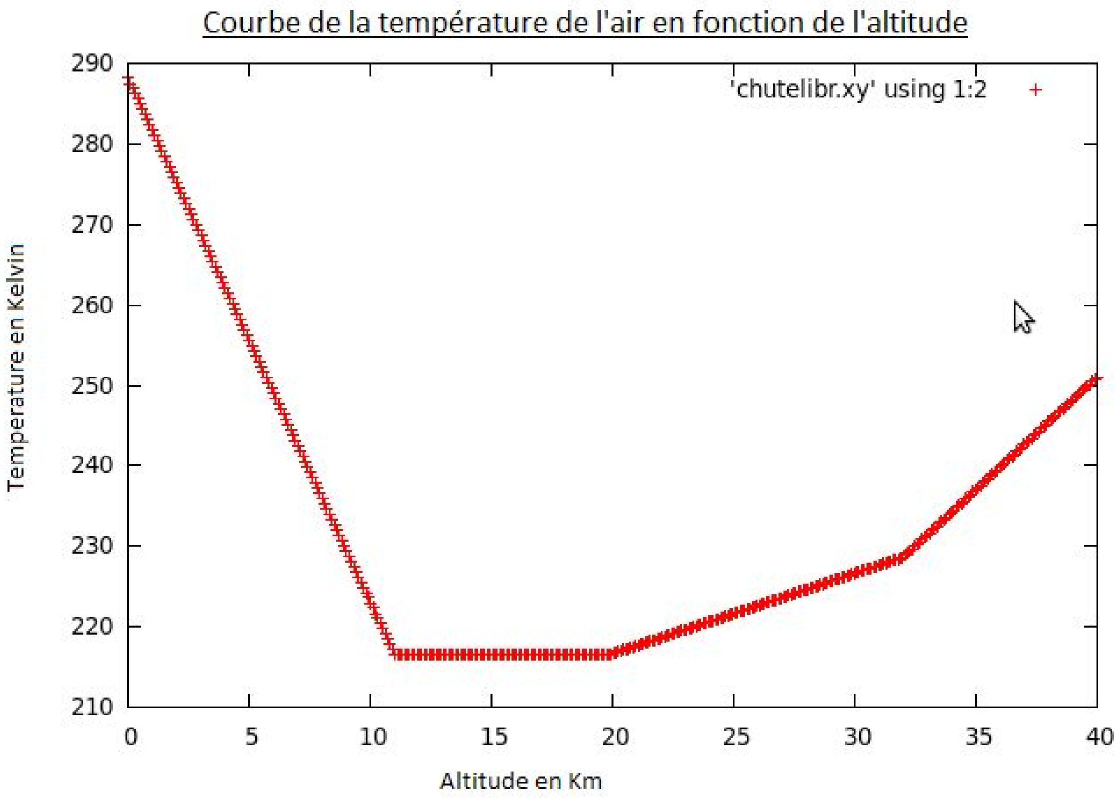
$T(z) = 288,15 - 6,5z \quad z \in [0; 11]$

$T(z) = 216,65 \quad z \in [11; 20]$

$T(z) = 216,65 + z \quad z \in [20; 32]$r

$T(z) = 228,65 + 2,8z \quad z \in [32; 47]$

Ainsi on obtient la courbe suivante:



3.1.2. Vitesse du son en fonction de l'altitude

La vitesse du son est définie comme étant la vitesse de propagation des ondes sonores sans déplacement de matière. Cette valeur dépend de plusieurs facteurs dont la pression, la composition du milieu de propagation du phénomène ondulatoire, la température…

Dans cette partie, nous nous focaliserons sur la variation de la vitesse du son en fonction de la température.

La vitesse du son dans les gaz peut être modélisée par l'équation suivante :

$$c=\sqrt{\frac{\gamma R T(z)}{M}}$$

Démonstration de cette équation:

Considérons une propagation à une dimension, dans un couloir cylindrique de section S, dans un fluide de masse volumique μ_0 au repos.

Nous allons établir l'équation vérifiée par le déplacement ξ (x, t). ξ est infiniment petit et donc petit devant la longueur d'onde.

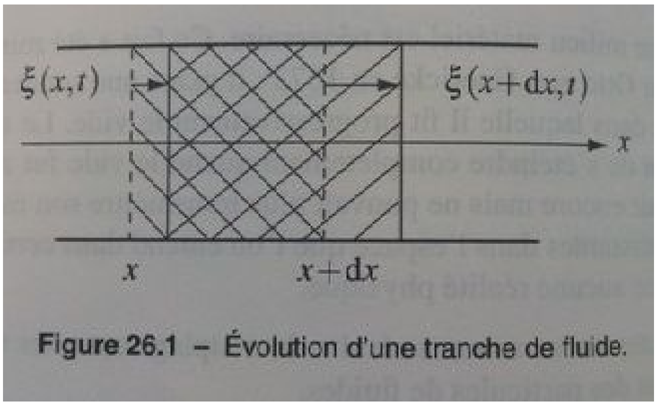


Figure 26.1 – Évolution d'une tranche de fluide.

En effectuant un bilan de masse pour le système fermé entre les plans d'abscisse x et x+dx, nous avons : $dm=\mu_0 Sdx$

En présence de l'onde sonore,

$$dm=\mu(x+dx,t)S\xi(x+dx,t)+\mu(x,t)Sdx-\mu(x,t)S\xi(x,t)$$

$$dm=S\frac{(\partial\mu\xi)}{(\partial x)}(x,t)dx+\mu(x,t)Sdx$$

$$dm=\mu_0\frac{(\partial\xi)}{(\partial x)}(x,t)Sdx+\mu(x,t)Sdx$$

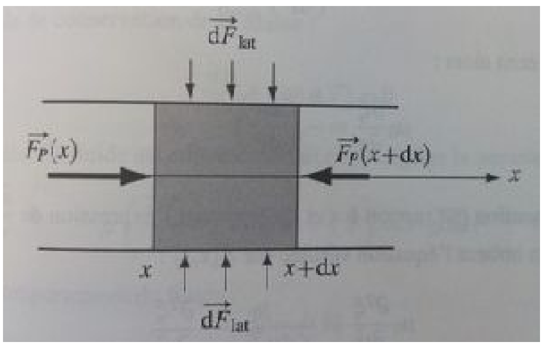
En appliquant la conservation de la masse, on écrit :

$$\mu_0 Sdx=\mu_0\frac{(\partial\xi)}{(\partial x)}(x,t)Sdx+\mu(x,t)Sdx$$

Soit :

$$\mu_0\frac{(\partial\xi)}{(\partial x)}=\mu_0-\mu(x,t)=-\mu_1(x,t) \quad (1)$$

On applique à présent le principe fondamental de la dynamique à ce système.



On note P la pression du fluide.

$\vec{F}_p(x)=(P_0+p_1(x,t))S\vec{u}_x$ à gauche

$\vec{F}_p(x+dx)=-(P_0+p_1(x+dx,t))S\vec{u}_x$ à droite

$d\vec{F}_{lat}$ sur la surface latérale.

Le couloir étant cylindrique, les efforts latéraux se compensent deux à deux.

Donc $d\vec{F}_{lat}=\vec{0}$.

En appliquant le principe fondamental de la dynamique, on obtient :

$$dm\frac{(\partial^2\xi)}{(\partial t^2)}=(P_0+p_1(x,t))S-(P_0+p_1(x+dx,t))S=\frac{(\partial p_1)}{(\partial x)}Sdx$$

Ou encore $dm=\mu_0 dx$:

$$\mu_0\frac{(\partial^2\xi)}{(\partial t^2)}=\frac{-(\partial p_1)}{(\partial x)} \qquad (2)$$

Avec un développement limité sur μ, on obtient : $\mu=f(P_0+p_1)=f(P_0)+f'(P_0)p_1$

$$\mu=\mu_0+\left(\frac{(\partial\mu)}{(\partial P)}\right)_{p_1=0}p_1$$

Soit :

$$\mu_1=\left(\frac{(\partial\mu)}{(\partial P)}\right)_{p_1=0}p_1$$

L'équation (1) s'écrit alors :

$$\mu_0\frac{(\partial\xi)}{(\partial x)}=-\left(\frac{(\partial\mu)}{(\partial P)}\right)_{p_1=0}p_1$$

En dérivant cette équation par rapport à x et en reportant l'expression de $\frac{(\partial p_1)}{(\partial x)}$ obtenue dans (2) on obtient l'équation :

$$\mu_0\frac{(\partial^2\xi)}{(\partial t^2)}=\frac{\mu_0}{\left(\frac{(d\mu)}{(dP)}\right)_{p_1=0}}\frac{(\partial^2\xi)}{(\partial x^2)}$$

Cette équation vérifie l'équation de d'Alembert :

$$\frac{(\partial^2\xi)}{(\partial t^2)}=c^2\frac{(\partial^2\xi)}{(\partial x^2)}$$

Avec c la vitesse de propagation de l'onde (m/s)

Par identification :

$$\frac{1}{c^2}=\left(\frac{(\partial\mu)}{(\partial P)}\right)_{p_1=0}$$

On sait que le coefficient de compressibilité du fluide au repos est :

$$X_0=\frac{1}{\mu_0}*\left(\frac{d\mu}{dP}\right)_{(p_1=0)}$$

Nous pouvons donc définir la célérité des ondes sonores par :

$$c=\frac{1}{\sqrt{\mu_0 X_0}} \quad (1)$$

L'air est considéré comme un fluide parfait, sa viscosité peut donc être négligée. Il subit des transformations adiabatiques réversibles, soient des transformations isentropiques.
Le fluide étant assimilé à un gaz parfait, et son évolution étant adiabatique réversible, on peut la décrire avec la loi de Laplace :

$$PV^{\gamma}=cste \quad \text{où} \quad \gamma=\frac{c_p}{c_v}$$

, rapport des capacités thermiques à pression et volume constants.

Donc :

$$\frac{P}{\mu^{\gamma}}=cste \Rightarrow \ln\left(\frac{P}{\mu^{\gamma}}\right)=cste \Rightarrow \ln(P)=cste+\gamma\ln(\mu)$$

Soit en dérivant l'équation :

$$\frac{dP}{P_0}=\gamma\frac{d\mu}{\mu_0} \Leftrightarrow \left(\frac{d\mu}{dP}\right)_{(p_1=0)}=\frac{\mu_0}{\gamma P_0}$$

Revenons à l'équation de la célérité des ondes sonores (1):

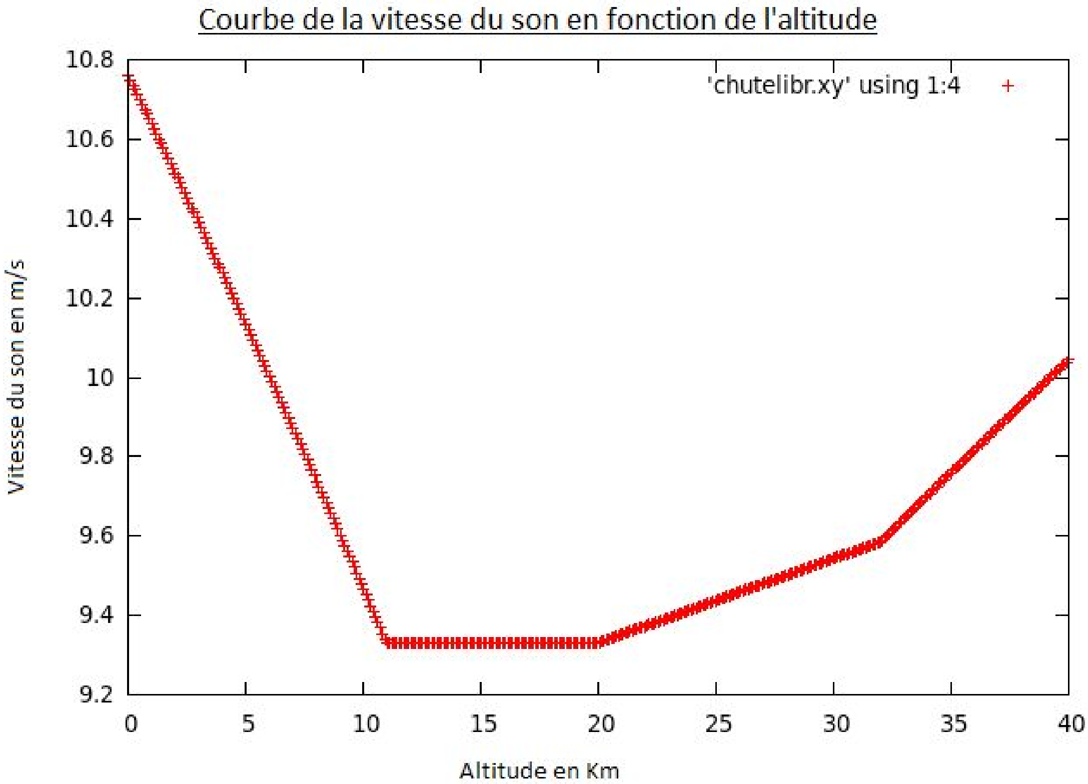
$$c=\frac{1}{\sqrt{\mu_0 X_0}} \quad \text{où} \quad X_0=\frac{1}{\mu_0}*\left(\frac{d\mu}{dP}\right)_{(p_1=0)}=\frac{1}{\gamma P_0}$$

Donc :

$$c=\sqrt{\gamma\frac{P_0}{\mu_0}} \Leftrightarrow c=\sqrt{\frac{\gamma RT(z)}{M}}$$

avec: c en (m/s)= vitesse du son (célérité)
γ = 1.4 le coefficient adiabatique
T en(K) = température
R en(J=K) = constante de Gay Lussac (8,314472J/K)

A partir de l'équation que nous venons de démontrer, nous avons réussi à modéliser la courbe décrivant la variation de la vitesse du son en fonction de l'altitude:



Cette étude nous permettra de déterminer l’apparition des frottements lors de la chute libre du parachutiste placé à 40km d’altitude, soit une fois que sa vitesse aura dépassé celle du son.

3.1.3 Intensité de pesanteur en fonction de l'altitude

Le poids est la résultante de la force de gravitation et de la force d'inertie d'entraînement sur la masse d'un corps.

D’après la deuxième loi de Newton :

$$\sum \vec{F} = \vec{P} + \vec{F} = m\vec{a} = \vec{0}$$

avec $\vec{P} = m\vec{g}$ et $\vec{F} = -G\frac{M_T m}{d^2}\vec{z}$

où G est l'intensité de pesanteur $G = 6{,}67.10^{11} m^3 kg^{-1} s^{-2}$

M_T la masse de la terre $M_T = 5{,}972.10^{24} kg$,

m la masse de l'objet, on choisit 80kg

et $d = R_T + z$ avec R_t le rayon de la terre $R_t = 6371\,km$ et

z l'altitude à partir de la surface de la Terre

Nous considérons que le parachutiste est uniquement soumis à la force de pesanteur terrestre.

Nous pouvons donc poser: $\vec{a} = g \times \vec{z}$.

Ainsi nous obtenons:

$$G\frac{M_T m}{(R_T+z)^2}=mg \Leftrightarrow g(z)=\frac{G*M_T}{R_T^2+2z*R_t+z^2} \Leftrightarrow g(z)=\frac{g_0*R_T^2}{R_T^2*(1+\frac{2z}{R_T}+\frac{z^2}{R_T^2})} \Leftrightarrow g(z)=g_0*(1+2\frac{z}{R_T})^{-2}$$

Or

$$z \ll R_T$$

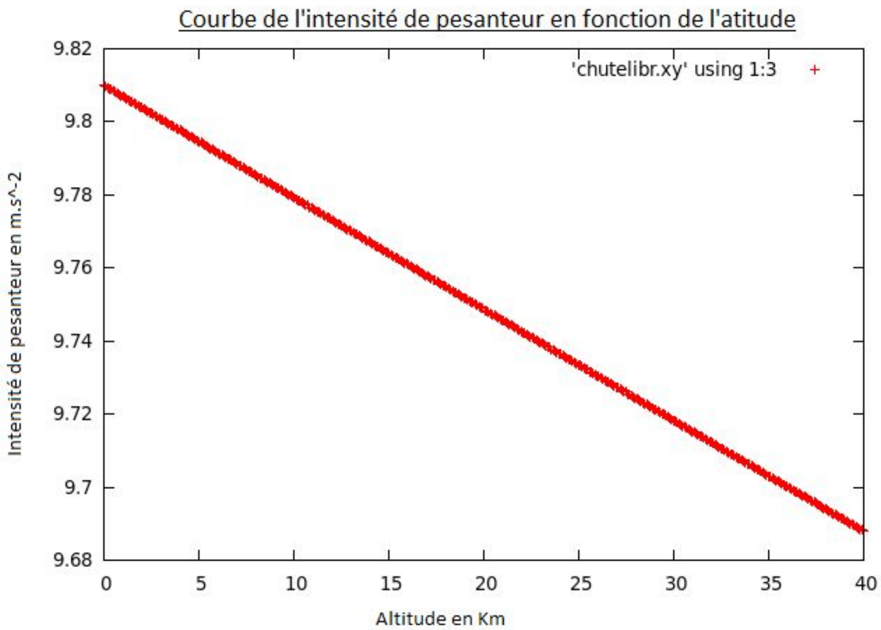
On peut en faire un développement limité sous la forme $(1+x)^{\alpha} \approx 1+\alpha x$

Ici $x=(\frac{z}{R_T})$

On obtient enfin:

$$g(z)=g_0*(1-2\frac{z}{R_T})$$

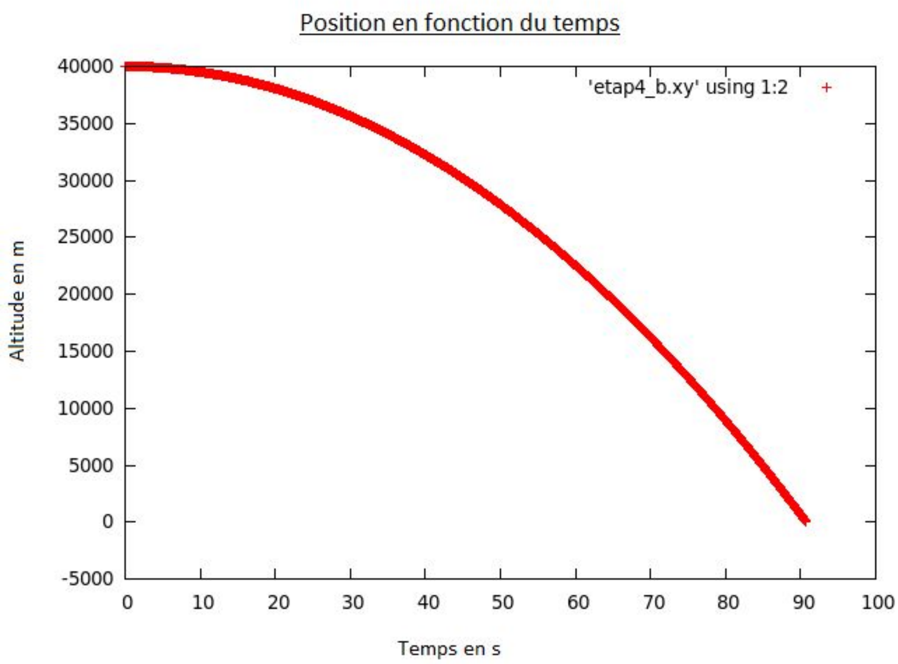
Grâce à notre étude précédente, nous avons pu coder une représentation de l'évolution de l'intensité de pesanteur en fonction de l'altitude. Ceci nous a permis de calculer l'équation différentielle de la vitesse lors de l'étude de la chute libre en elle-même.



3.2 . Étude de la chute sans frottements :

Nous allons tout d'abord étudier la chute libre du parachutiste sans considérer les frottements. Nous avons étudié sa position et sa vitesse en fonction du temps puis sa vitesse en fonction de sa position. Puis nous avons comparé nos résultats à ceux obtenus dans l'étude de la vitesse du son.

3.2.1 Position en fonction du temps :



D'après ce graphique, on observe que lorsque l'on considère la chute sans frottement, le parachutiste met 90s pour atteindre la terre.
Or seulement le début du saut peut être considéré sans frottements, soit jusqu'à ce que le parachutiste atteigne la vitesse du son. Nos résultats suivants nous permettent de déterminer le moment et la position où cette vitesse est atteinte, et ainsi de déterminer le moment où les frottements ne sont plus négligeables.

3.2.2 Vitesse en fonction du temps :

La vitesse de l'objet en fonction du temps durant sa chute libre dépend de l'intensité de pesanteur et de la vitesse initiale, ainsi nous obtenons :

$$\frac{dv}{dt}=-g(z)\;et\;\frac{dz}{dt}=v$$

<u>Démontrons cette équation :</u>

Dans le système supposé galiléen, la somme des forces appliquées sur un objet est :

$\sum \vec{F}=m\vec{a}$

Or ici la seule force appliquée est le poids P donc :

$$\vec{P}=m\vec{g}=m\vec{a}\Leftrightarrow\vec{g}=-g\vec{z}=\vec{a}=\frac{d\vec{v}}{dt}$$

avec g l'intensité de pesanteur m/s²
a l'accélération m/s²

Ensuite on intègre cette relation en fonction du temps afin d'obtenir une vitesse :

$$v(t)=-gt+cte$$

Si nous posons une condition initiale :

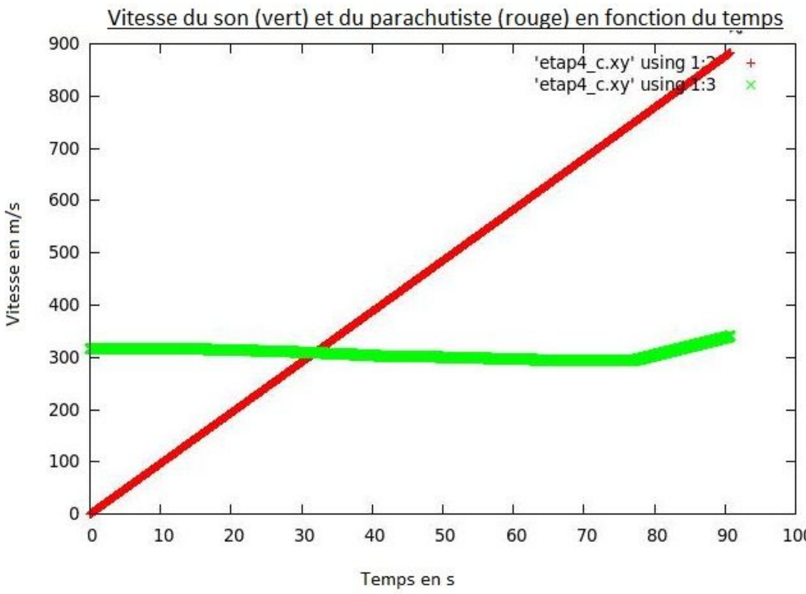
$$v(0)=v_0=0$$

On obtient donc la relation suivante:

$$v=-g*t=\frac{dz}{dt}\Leftrightarrow z=-\frac{1}{2}gt^2+z_0$$

Ceci sont les solutions exactes, nous pouvons également les calculer par la méthode d'Euler.

Nous obtenons le graphe suivant:



On remarque que le parachutiste atteint la vitesse du son au bout de 32s de chute. En combinant les résultats des deux courbes précédentes, nous pouvons supposer que le parachutiste atteint la vitesse du son à 35km d'altitude, soit après 5km de chute. Nous vérifions cette hypothèse dans la partie suivante.

3.2.3 Vitesse en fonction de l'altitude :

Un objet entre en chute libre en partant d'une altitude z avec une vitesse initiale nulle. Sa vitesse à l'arrivée au sol est décrite par l'équation suivante:

$$v(z)=\sqrt{2gz}$$

Démonstration de l'équation:

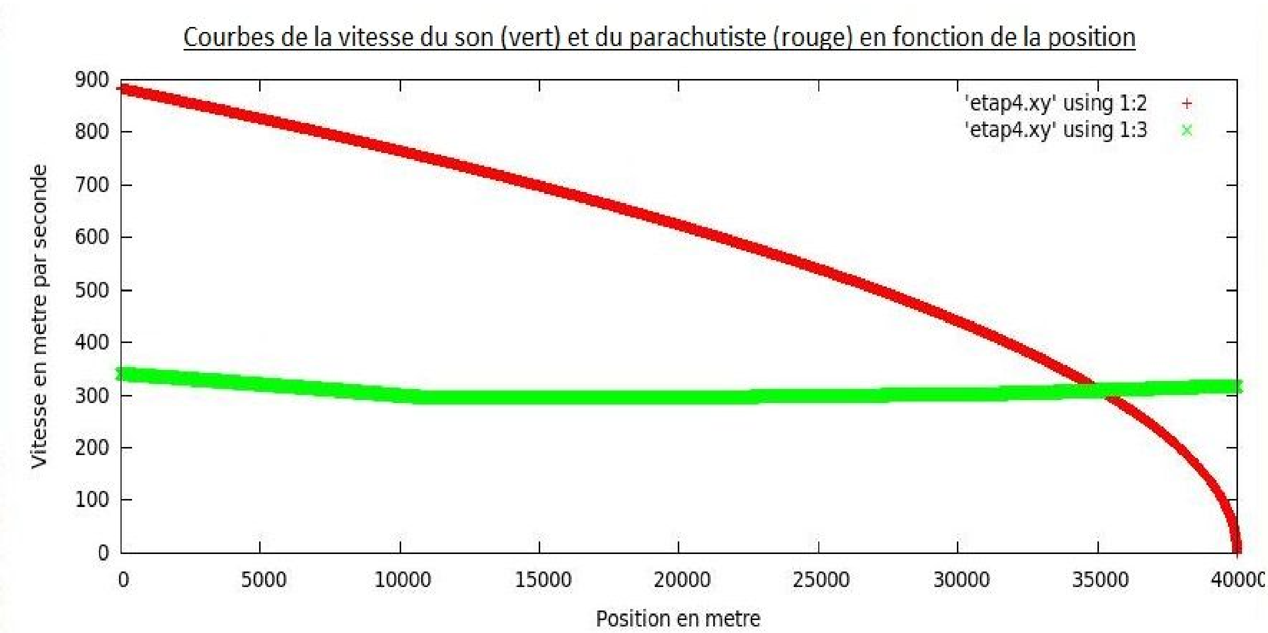
L'énergie cinétique et l'énergie potentielle de pesanteur sont définies telles que:

$$Ep = mgz \quad \text{et} \quad Ec = 1/2mv^2$$

L'énergie mécanique étant conservée lors d'une chute libre, nous obtenons:

$$Ec+Ep=0 \Leftrightarrow v^2=2gz \Leftrightarrow v(z)=\sqrt{2gz}$$

Nous obtenons la modélisation suivante de la vitesse en fonction de l'altitude:



Les courbes représentant mutuellement l'évolution de la vitesse du parachutiste et celle du son se croisent au niveau d'un point d'abscisse 35000. Notre hypothèse précédente, selon laquelle le parachutiste atteint la vitesse du son au bout de 32s et 5 km de chute, est donc vérifiée.

Les représentations précédentes sont réalisées sur la totalité de la chute cependant, elles ne sont valables que sur une certaine distance de chute parcourue.

Nous avons donc effectué une deuxième étude entre le moment où le parachutiste atteint la vitesse du son et le moment où il ouvre son parachute, en prenant en compte les frottements qui interviennent sur cette partie de la chute.

3.3. Étude après le passage de la vitesse du son :

Comme énoncé précédemment, nous nous intéressons ici à l'évolution de la vitesse et de la position du parachutiste en fonction du temps ainsi qu'à celle de sa vitesse en fonction de sa position, sans prendre en compte la partie de la chute où les frottements sont négligeables. Cependant nous ne considérons toujours pas les frottements dûs à l'ouverture du parachute, cas qui sera traité plus loin dans notre travail.

Une fois la vitesse du son atteinte par le parachutiste, sa vitesse est importante. La force de frottement qui s'exerce est par conséquent proportionnelle au carré de la vitesse:

$$\vec{f} = -kv\vec{v}$$

où k est une constante qui dépend du fluide et des caractéristiques de l'objet tel que:

$$k = 1/2\eta CxS$$

avec: η la viscosité de l'air $\eta = 1.983\ 10^{-5}$

Cx le coefficient de traînée $Cx = 1.1$

S la surface frontale $S = 0.36m^2$

On obtient : k=0.035kg/m

D'après la deuxième loi de Newton :

$$\sum \vec{F} = m\vec{a} \Leftrightarrow \vec{P} + \vec{f} = m\vec{a}$$

et grâce au principe fondamentale de la dynamique, nous obtenons :

$$m\vec{g} - kv\vec{v} = m\vec{a}$$

Afin de faciliter les calculs, nous allons utiliser un axe (O, $\vec{z}$) vertical descendant, et nous obtenons:

$$mg - kv^2 = ma \Leftrightarrow mdv/dt + kv^2 = mg$$

$$\frac{dv}{dt} = g*t - \frac{k}{m}*v^2$$

Nous résolvons cette équation différentielle de façon numérique:

- Méthode d’Euler Explicite:

Tout d'abord nous allons présenter brièvement la méthode d'Euler Explicite:
Pour cette méthode, on trace la tangente à la courbe au point de départ de coordonnées (x_0, y_0) à l’aide de l’équation différentielle. Puis on détermine un pas $h = x0 - x1$ et on définit le point d'arrivée comme le nouveau point de coordonnée (x_1, y_1) . On trace la tangente à la courbe et on la reporte sur (x_1, y_1) . Nous répétons cette opération tout le long de la courbe.
Le développement limité en (x_0, y_0) donne :

$$y_1 = y_0 * (\frac{dy}{dx})_{x=x_0} = y_0 + h * f(x_0, y_0)$$

D'où la formule générale:

$$y_{n+1} = y_n + h * (\frac{dy}{dx})_{x=x_n} \Leftrightarrow y_{n+1} = y_n + h * f(x_n, y_n)$$

Cette méthode n’est néanmoins pas très précise, le champ d’erreur est conséquent.

- Méthode d’Euler Predictor:

Il existe une deuxième méthode plus précise, nommée “Euler Predictor” :
Le principe est le même que la méthode précédente, mais après avoir défini la deuxième tangente à la courbe, la moyenne de ces deux tangentes est calculée afin d’obtenir le point suivant. Ainsi, la tangente permettant de décrire l’évolution de la courbe correspond à la moyenne de la tangente des deux points précédents.
On fait donc un développement limité en (x_0, y_0)

$$y_1 \sim y_0 * (\frac{dy}{dx})_{x=x_0} = y_0 + h * f(x_0, y_0)$$

et au point suivant (x_1, y_1)

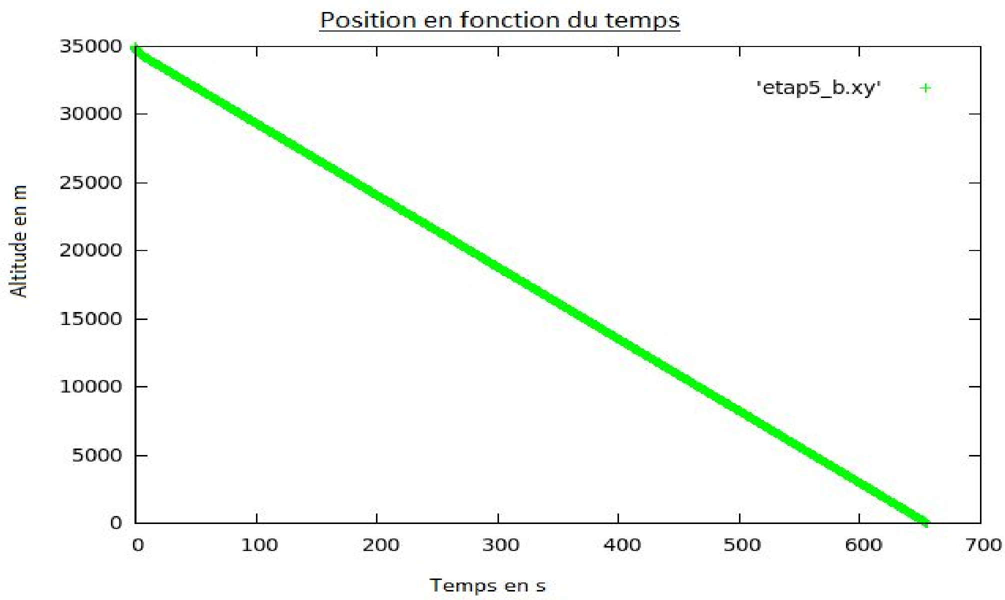
$$y_0 \sim y_1 - h * (\frac{dy}{dx})_{x=x_1} = y_1 + h * f(x_1, y_1)$$

On obtient donc $\frac{(y_1 - y_0)}{h} = \frac{1}{2}(f(x_0, y_0) + f(x_1, y_1))$

Cette méthode permet donc de réduire de moitié le champ d’erreurs.

Nous avons choisi cette dernière méthode pour modéliser l'évolution de la vitesse et de la position du parachutiste en fonction du temps, ainsi que sa vitesse en fonction de sa position.

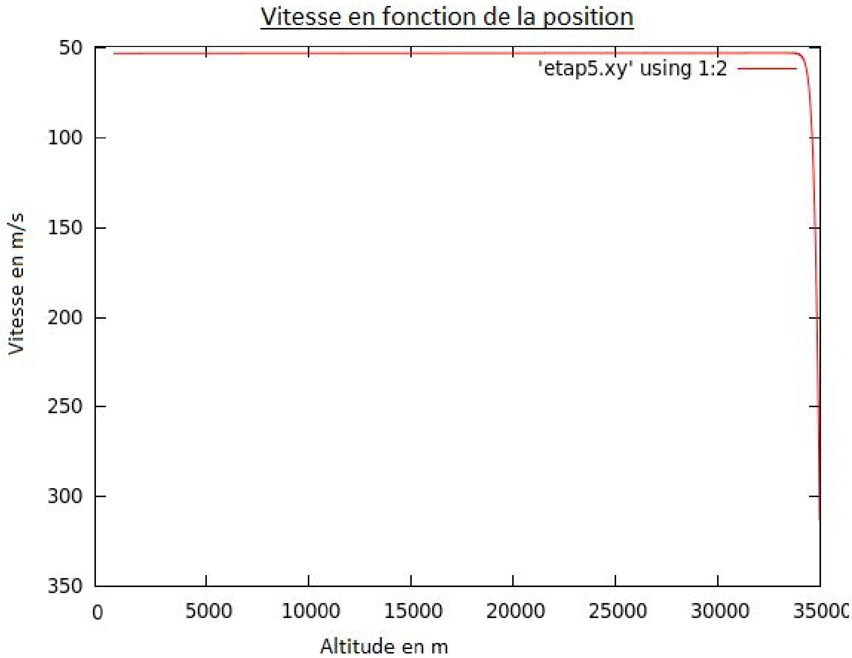
3.3.1 Position en fonction du temps :



Ce graphique commence au moment où le parachutiste vient tout juste de passer le mur du son, soit à 35 000 m d'altitude. A partir de cet instant les frottements ne sont plus négligeables. Le coefficient de frottement k est égale à 0,035kg/m.
On remarque donc qu'il met 625s après avoir franchis le mur du son pour atteindre 1000m, c'est-à-dire le moment où il ouvrira son parachute et où les frottements deviendront plus importants.
Si on rajoute à cela le temps de la chute considéré sans frottement de Felix Baumgartner et celui avec de faibles frottements, on obtient donc en temps total de chute libre de 10min 57s.

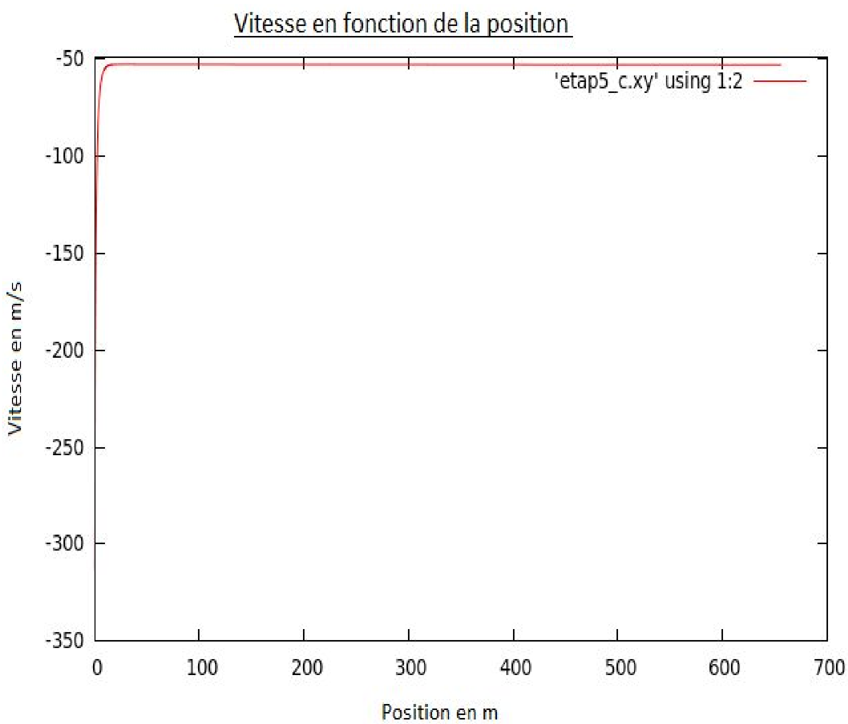
Or la durée réelle totale de la chute libre du saut de Felix Baumgartner est de 4min19s. L'écart important est sûrement dû aux erreurs d'approximation des lectures graphiques, à celles de la méthode d'Euler ainsi qu'à celles des développements limités.

3.3.2 *Vitesse en fonction de la position :*



Grâce à ce graphique on peut observer qu'après le passage du mur du son la vitesse de l'homme chute de 300m/s sans qu'il ne change d'altitude.
Ayant dépassé la vitesse du son, il a subi des frottements qui ont ralenti considérablement sa chute, compensant son poids, lui permettant ainsi de se stabiliser à une vitesse de 50 mètres par seconde.

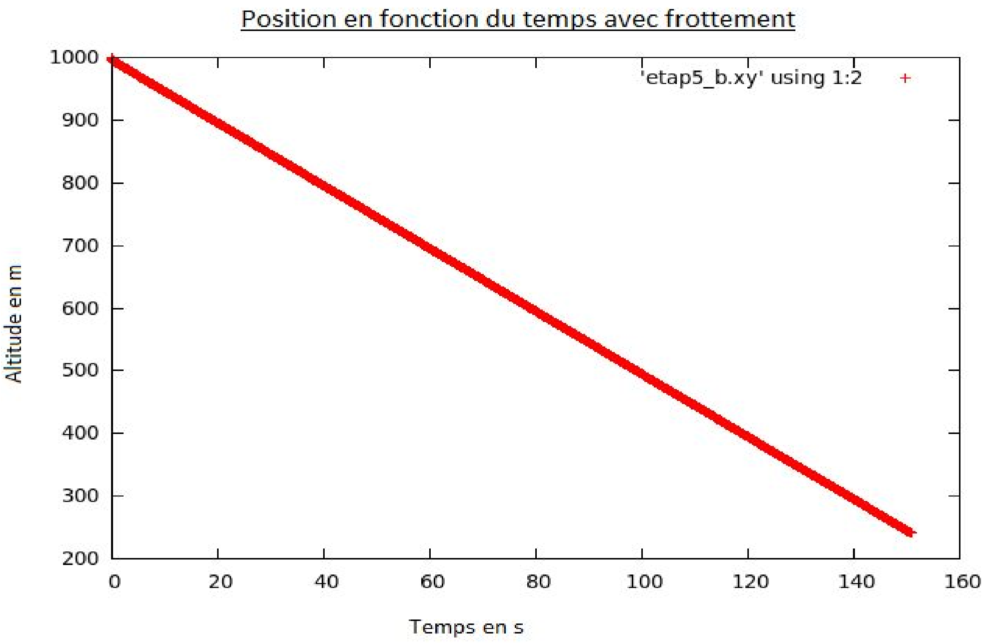
3.3.3 *Vitesse en fonction du temps :*



On remarque que dès que l'homme a passé le mur du son sa vitesse est instantanément ralentie. Cela est dû au fait que les frottements ont augmenté et ne peuvent plus être considérés comme nuls. Sa vitesse se stabilise donc rapidement à 50 mètres par seconde.

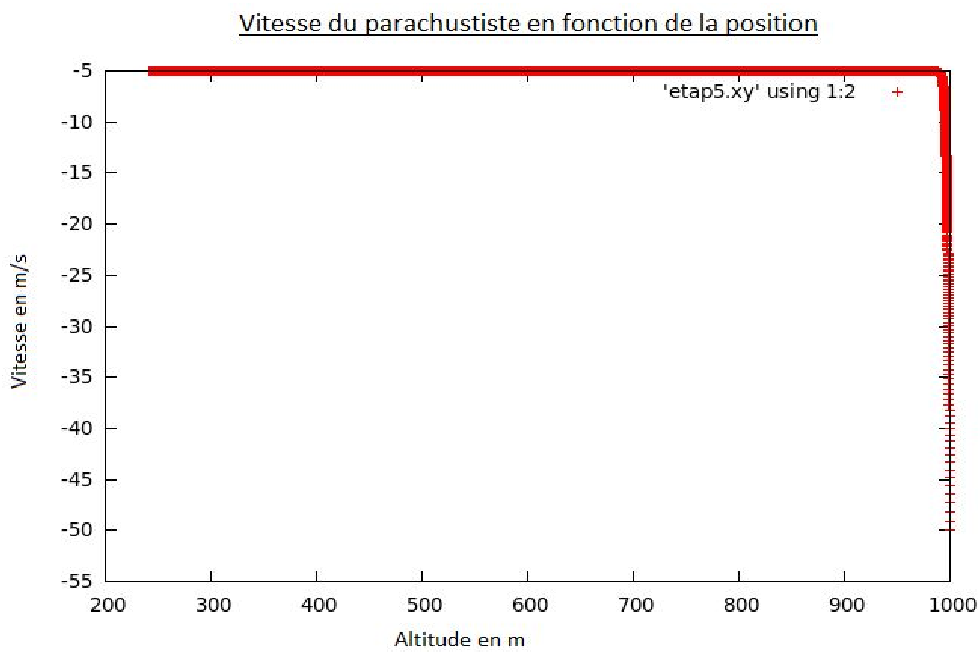
3.4. Étude après ouverture du parachute:

3.4.1 Position en fonction du temps :



Nous pouvons voir sur ce graphique que le parachutiste atteindra le sol 190 secondes après l'ouverture du parachute, soit 3 minutes et 10 secondes environ. Or Felix Baumgartner a mis 4 min 27s pour atteindre le sol. Cet écart s'explique facilement puisque 1000m est la hauteur à laquelle il avait décidé, lors de la préparation du saut , d'ouvrir son parachute, or dans la pratique il l'a ouvert à 2500m .

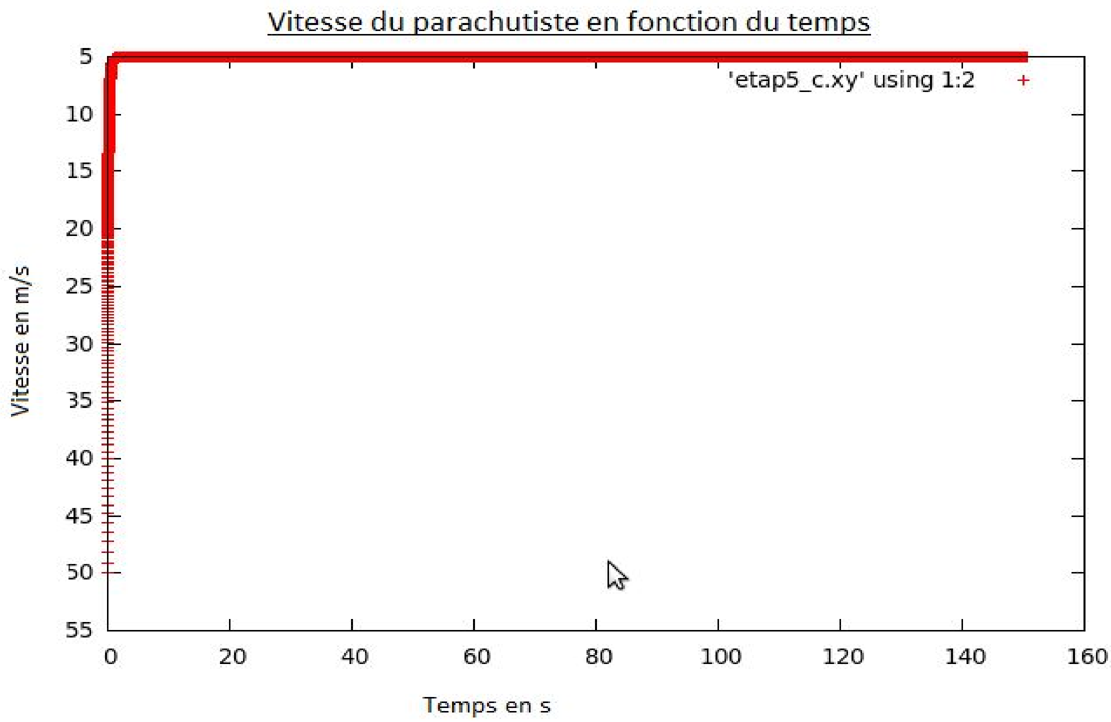
3.4.2 Vitesse en fonction de la position :



On peut remarquer que lors de l'ouverture du parachute l'altitude de l'homme reste constante et sa vitesse diminue fortement jusqu'à atteindre une vitesse constante de 5m/s à laquelle il va finir sa chute.
Cela s'explique par la grande surface de portance du parachute

qui compense fortement la pesanteur.

3.4.3 Vitesse en fonction du temps :



Ce graphique décrit l'évolution de la vitesse du parachutiste, parachute ouvert, en fonction du temps. On remarque que dès l'ouverture du parachute sa vitesse est instantanément ralentie jusqu'à atteindre une vitesse constante de 5 m/s. Cela semble évident puisque la surface du parachute étant très grande, elle induit un forte augmentation des frottements. On a donc un coefficient de frottement k =0,39 kg/m ce qui est 100 fois plus grand que celui après le passage du mur du son.

4.Conclusions et perspectives

Le modèle théorique que nous avons modélisé est plutôt fiable car tous nos résultats, ayant pu être comparés aux mesures du vrai saut en chute libre de Felix Baumgartner, sont similaires à une erreur d'approximation près due aux erreurs de lectures de graphiques, à l'imprécision de la méthode d'Euler utilisée ainsi qu'aux développements limités.

Ce projet nous a paru intéressant car il nous a permis de mettre nos connaissances en mécanique du point en application. En effet, le fait de travailler sur un évènement historique mondial nous a permis de mieux comprendre les enjeux et les caractéristiques de ce saut. Nous avons pu voir que cela nécessitait beaucoup de préparation, non seulement physique pour Felix Baumgartner mais également scientifique afin d'optimiser le saut et ne pas faire d'erreur pouvant avoir des conséquences terribles. La performance de Felix Baumgartner, première en son genre, est mémorable pour la technique et les recherches qui se trouvent derrière.

Ce projet a permis à tout le groupe de découvrir les différents facteurs influant sur une chute libre et d'appliquer la méthode d'Euler, tout comme le langage fortran90.

Il nous a également permis d'appréhender le travail de recherche en sciences physiques, par la recherche bibliographique et sitographique. Les nouveaux outils utilisés, comme la méthode d'Euler, sont un apport à notre culture scientifique. Nous sommes satisfaits d'avoir pu mettre nos connaissances en application pour un projet concret et d'avoir pu voir leur cohérence avec la réalité.

5.Bibliographie

1. M.-N. Santz, “Physique tout en un pour PC/PC*” , 2014
2. http://www.scienceamusante.net/index.php , 25/04/15
3. http://www.web-sciences.com/ , 25/04/15
4. https://fr.wikipedia.org/wiki/Felix_Baumgartner 12/01/15
5. http://wwwens.aero.jussieu.fr/lefrere/master/mni/mncs/te/te2-edo/te2-edo.pdf ,(17/02/2016)
6. http://www.physagreg.fr/mecanique/m12/M12-chute-libre-frottements.pdf
7. http://owl-ge.ch/IMG/pdf/frottement.pdf

6.Annexes

6.1. Code

```
Program chutelibre
implicit none
real:: a, to, t, h, dh, g, g0, Rt, z, dt , h1 , p , p0 , v , v0, pOld,t1,temps,vson,pos, tz,k
integer:: N, hmax, hmin, compteur
real :: temptair, galtitude, vitson, dvdt, dvdt2, dvdt3

N=400
hmax=40000
hmin=0
dt=0.01
V     = V0
!!!!!!!!!!!!!!!!!!!!valeurs paramètres chute sans frottements!!!!!!!!!!!!!!!!!!!!!!!!!!!
V0=0.
P0=40000.
V  = V0
P = P0
dt     = 0.1
temps     = 0.
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

dh=(hmax-hmin)/(real(N)-1)
```

```
open(unit=20,file='chutelibr.xy',form='formatted')
do compteur=1,N
h= hmin+(real(compteur)-1.)*dh
 call Atmosphere_Normalisee(h,T1,a,h1)
write(20,604)h,temptair(h,T1,a,h1),galtitude(h),vitson(temptair(h,T1,a,h1))
enddo
 close(20)
604 format(4(F15.4))


!courbes pour chute sans frottement
open(unit=21,file='etap4_b.xy',form='formatted')
open(unit=22,file='etap4_c.xy',form='formatted')
open(unit=23,file='etap4.xy',form='formatted')
do while  (P>=0)
      POld = p

      call EulerPredicteur_dvdt1(dt,v,p)  ! iteration de un pas de temps via la methode dEuler
Predicteur
      call EulerPredicteur_dpdt1( dt,v,p)

      call Atmosphere_Normalisee(p,T1,a,h1)
         Vson=vitson(temptair(p,T1,a,h1))

write(*,*) temps,P
       write(21,5004) temps,P
       write(22,5004) temps,-V,Vson
       write(23,5004) P,-v,vson




     temps =temps+dt
enddo
      5004 format(4(F15.4))
      close(21)
      close(22)
      close(23)

!!!!!!!!!!!!!!!!!!!valeurs paramètres chute complète avec frottements!!!!!!!!!!!!!!!!!!!!!!!!!
V0=0.
P0=40000.
```

```
V  = V0
P = P0
dt     = 0.1
temps     = 0.
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!courbes pour chute complète avec frottement
open(unit=30,file='etap7_b.xy',form='formatted')
open(unit=31,file='etap7_c.xy',form='formatted')
open(unit=32,file='etap7.xy',form='formatted')
do while  (P>=0)
     POld = p
     call kaltitude(h,k)
     call EulerPredicteur_dvdt1(dt,v,p)  ! iteration de un pas de temps via la methode dEuler
Predicteur
     call EulerPredicteur_dpdt1( dt,v,p)

     call Atmosphere_Normalisee(p,T1,a,h1)
       Vson=vitson(temptair(p,T1,a,h1))

write(*,*) temps,P
      write(30,5004) temps,P
      write(31,5004) temps,-V,Vson
      write(32,5004) P,-v,vson



    temps =temps+dt
enddo
     5012 format(4(F15.4))
     close(30)
     close(31)
     close(32)

!!!!!!!!!!!!!!!!!!!!!valeurs paramètres chute avec frottements vitesse du son!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
V0=350 !vitesse du son dans l'air
P0=35000 !altitude à laquelle le parachutiste atteint la vitesse du son
V  = V0
P = P0
temps=0.
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

```
!courbes chute avec frottement (vitesse du son)
 open(unit=24,file='etap5_b.xy',form='formatted')
open(unit=25,file='etap5_c.xy',form='formatted')
open(unit=26,file='etap5.xy',form='formatted')
do while  (P>=0)
     POld = p

     call EulerPredicteur_dvdt2(dt,v,p) ! iteration de un pas de temps via la methode dEuler
Predicteur
     call EulerPredicteur_dpdt2( dt,v,p)
     call Atmosphere_Normalisee(p,T1,a,h1)
       Vson=vitson(temptair(p,T1,a,h1))

      write(24,5008) temps,P
      write(25,5008) temps,-V,Vson
      write(26,5008) P,-v,vson



    temps =temps+dt
enddo
     5008 format(4(F15.4))
     close(24)
     close(25)
     close(26)

!!!!!!!!!!!!valeurs paramètres chute avec frottements ouverture parachute!!!!!!!!!!!!!!!!!!
V0=51 !vitesse à laquelle va le parachutiste lorsqu'il ouvre son parachute
P0=1000 !altitude a laquelle le parachutiste ouvre son parachute
V  = V0
P = P0
temps=0.
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!


!courbes chute avec frottement (parachute)
 open(unit=27,file='etap6_b.xy',form='formatted')
open(unit=28,file='etap6_c.xy',form='formatted')
open(unit=29,file='etap6.xy',form='formatted')
do while  (P>=0)
     POld = p
```

```
     call EulerPredicteur_dvdt3(dt,v,p) ! iteration de un pas de temps via la methode dEuler
Predicteur
     call EulerPredicteur_dpdt3 (dt,v,p)
     call Atmosphere_Normalisee(p,T1,a,h1)
       Vson=vitson(temptair(p,T1,a,h1))

      write(27,5010) temps,P
      write(28,5010) temps,-V,Vson
      write(29,5010) P,-v,vson



    temps =temps+dt
enddo
     5010 format(4(F15.4))
     close(27)
     close(28)
     close(29)

!!!!!!!!!!!!!!!!!!!!valeurs paramètres chute complète avec frottements!!!!!!!!!!!!!!!!!!!!!!!!!!
V0=0.
P0=40000.
V  = V0
P = P0
dt     = 0.1
temps     = 0.
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!courbes pour chute complète avec frottement
open(unit=30,file='etap7_b.xy',form='formatted')
open(unit=31,file='etap7_c.xy',form='formatted')
open(unit=32,file='etap7.xy',form='formatted')
do while  (P>=0)
     POld = p
     call kaltitude(h,k)
     call EulerPredicteur_dvdt1(dt,v,p)  ! iteration de un pas de temps via la methode dEuler
Predicteur
     call EulerPredicteur_dpdt1( dt,v,p)

     call Atmosphere_Normalisee(p,T1,a,h1)
       Vson=vitson(temptair(p,T1,a,h1))

write(*,*) temps,P
```

```
       write(30,5004) temps,P
       write(31,5004) temps,-V,Vson
       write(32,5004) P,-v,vson



     temps =temps+dt
enddo
      5012 format(4(F15.4))
      close(30)
      close(31)
      close(32)



end program  chutelibre


! subroutine paramètrant la courbe de la température en fonction de l'altitude
subroutine Atmosphere_Normalisee(h,T1,a,h1)
implicit none
real, intent(in)::h
!local declaration
real,intent(out)::T1
real,intent(out)::a
real,intent(out)::h1

if (h>=0 .AND.h<11000) then
T1=288.15; a=-6.5e-3; h1=0
else if (h>=11000 .AND.h<20000) then
T1=216.65; a=0; h1=11000
else if (h>=20000 .AND.h<32000) then
T1=216.65; a=1e-3; h1=20000
else if (h>=32000 .AND.h<40000) then
T1=228.65; a=2.8e-3; h1=32000
endif
end subroutine Atmosphere_Normalisee



!subroutine caractérisant l'évolution de la vitesse d'une chute sans frottement
subroutine EulerPredicteur_dvdt1(dt,V,pos)
     implicit none
```

```
    real, intent(in)    :: dt , pos
    real, intent(inout)   :: V

    real :: dvdt
    real :: d1 , d2 , vpred
    real :: k

    d1= dvdt(pos)
    Vpred = V + dt*d1
    d2= dvdt(pos)
    V = V + dt*(d1+d2)/2.

end subroutine EulerPredicteur_dvdt1



!subroutine caractérisant l'évolution de la vitesse d'une chute avec frottement (vitesse du son atteinte)
subroutine EulerPredicteur_dvdt2(dt,V,pos)
    implicit none
    real, intent(in)    :: dt , pos
    real, intent(inout)   :: V

    real :: dvdt2
    real :: d1 , d2 , vpred
    real :: K
    k=0.0035
    d1= dvdt2(pos)-k*v**2
    Vpred = V + dt*d1
    d2= dvdt2(pos)-k*vpred**2
    V = V + dt*(d1+d2)/2.

end subroutine EulerPredicteur_dvdt2

!subroutine caractérisant l'évolution de la vitesse d'une chute avec frottement (parachute)
subroutine EulerPredicteur_dvdt3(dt,V,pos)
    implicit none
    real, intent(in)    :: dt , pos
    real, intent(inout)   :: V

    real :: dvdt3
    real :: d1 , d2 , vpred
    real :: K
```

```
    k=0.39
    d1= dvdt3(pos)-k*v**2
    Vpred = V + dt*d1
    d2= dvdt3(pos)-k*vpred**2
    V = V + dt*(d1+d2)/2.

end subroutine EulerPredicteur_dvdt3



!subroutine caractérisant l'évolution de la position d'une chute sans frottement
subroutine EulerPredicteur_dpdt1(dt,V,Pos)
    implicit none
    real, intent(in)    :: dt,V
    real, intent(inout) :: Pos
    !local declaration
    real :: d1,d2   ! les deux pentes de EulerPredicteur
    real :: Ppred
    d1 = V
    Ppred = Pos+ dt*d1
    d2 = V
    pos = pos + 0.5*dt*(d1+d2)
    !
    !
end subroutine EulerPredicteur_dpdt1


!subroutine caractérisant l'évolution de la position d'une chute avec frottement (vitesse du
son)
subroutine EulerPredicteur_dpdt2(dt,V,Pos)
    implicit none
    real, intent(in)    :: dt,V
    real, intent(inout) :: Pos
    !local declaration
    real :: d1,d2   ! les deux pentes de EulerPredicteur
    real :: Ppred
    d1 = -V
    Ppred = Pos+ dt*d1
    d2 = -V
    pos = pos + 0.5*dt*(d1+d2)
    !
    !
end subroutine EulerPredicteur_dpdt2
```

```
!subroutine caractérisant l'évolution de la position d'une chute avec frottement (parachute)
subroutine EulerPredicteur_dpdt3(dt,V,Pos)
    implicit none
    real, intent(in)    :: dt,V
    real, intent(inout) :: Pos
    !local declaration
    real :: d1,d2   ! les deux pentes de EulerPredicteur
    real :: Ppred
    d1 = -V
    Ppred = Pos+ dt*d1
    d2 = -V
    pos = pos + 0.5*dt*(d1+d2)
    !
    !
end subroutine EulerPredicteur_dpdt3

!subroutine attribuant un k selon l'altitude
subroutine kaltitude (h,k)
 implicit none
real, intent(in)::h
!local declaration
real,intent(out)::k

if (h>=0 .AND.h<1000) then
k=0.39
else if (h>=1000 .AND.h<35000) then
k=0.0035
else if (h>=35000 .AND.h<40000) then
k=0
endif
end subroutine kaltitude


!fonction de la température en fonction de l'altitude
real function temptair(h,T1,a,h1)
implicit none
real, intent(in) ::a, h1, T1,h
temptair=T1+a*(h-h1)
return
end function temptair

!fonction de la pesanteur en fonction de l'altitude
```

```
real function galtitude(h)
implicit none
real, intent(in) :: h
real:: g0, Rt
g0=-9.81
Rt=6371000
galtitude=g0*(1-((2*h)/RT))
return
end function galtitude

!fonction de la vitesse du son en fonction de la température
real function vitson(Tz)
implicit none
real,intent(in)::Tz
real :: gama , r
gama =1.4
  r = 287.
  vitson=sqrt(gama*r*Tz)
return
end function vitson

! équation différentielle de la vitesse chute sans frottement
real function dvdt(pos)
implicit none
real , intent(in) :: pos
real :: galtitude
dvdt = galtitude(pos)

return
end function dvdt

! équation différentielle de la vitesse chute avec frottement (vitesse du son)
real function dvdt2(pos)
implicit none
real , intent(in) :: pos
real :: galtitude
dvdt2 = -galtitude(pos)

return
end function dvdt2

! équation différentielle de la vitesse chute avec frottement (parachute)
real function dvdt3(pos)
```

```
implicit none
real , intent(in) :: pos
real :: galtitude
dvdt3 = -galtitude(pos)

return
end function dvdt3
```